

## Introduction

DDR SDRAM devices are widely used today for a broad range of applications, such as embedded processor systems, image processing, storage, communications and networking. In addition, the universal adoption of DDR SDRAM in PCs made DDR SDRAM memory a compelling, cost effective DRAM solution for high-bandwidth applications.

Stratix® II devices support two modes of DDR SDRAM interfacing: with and without dedicated DQS circuitry. In addition, Stratix II and Stratix II GX device families offer two different data paths or physical interfaces (PHYs) with the dedicated DQS phase-shift circuitry: the legacy integrated static data path and controller (referred to as the legacy controller in this document) and ALTMEMPHY. With dedicated circuitry and ALTMEMPHY, Stratix II devices can interface with 233-MHz DDR SDRAM. Without dedicated circuitry, Stratix II devices can interface with 150-MHz DDR SDRAM.

**Table 1. DDR SDRAM Interface Maximum Clock Frequency Support in Stratix II Devices** *Notes (1), (2), (3)*

Speed Grade	Frequency (MHz)			
	With Dedicated DQS Circuitry		Without Dedicated DQS Circuitry	
	ALTMEMPHY		Legacy PHY	Legacy PHY
	Half-Rate Mode	Full-Rate Mode		
-3	200	233	200	150
-4	200	200	200	133
-5	200	167	200	100

**Notes for Table 1:**

- (1) The supported operating frequencies listed here are memory interface maximums for the FPGA device family. Your design's actual achievable performance is based on design and system specific factors, as well as static timing analysis of the completed design.
- (2) These specifications apply to both commercial and industrial devices.
- (3) These specifications are applicable for both interfacing with DDR SDRAM modules and discrete devices.

This application note describes several systems which interface DDR SDRAM memory with a Stratix II FPGA. It details the electrical and timing analysis for the interface with a Stratix II device.



Use this application note together with the *External Memory Interfaces in Stratix II and Stratix II GX Devices* chapter of the *Stratix II Device Handbook*.

## DDR SDRAM Overview

DDR SDRAM is a 2n prefetch architecture with two data transfers per clock cycle. It uses a strobe, DQS, which is associated with a group of data pins (DQ) for read and write operations. Both the DQS and DQ ports are bidirectional. Address ports are shared for write and read operations.

Write and read operations are sent in bursts, and DDR SDRAM supports burst lengths of 2, 4, and 8. This means that you need to provide 2, 4, or 8 groups of data for each write transaction, and you will receive 2, 4, or 8 groups of data each read transaction. The time between when the read command is clocked into the memory and when the data is presented at the memory pins is called the column address strobe (CAS) latency. DDR SDRAM supports CAS latencies of 2, 2.5, and 3, depending on the operating frequency. Both the burst length and CAS latency are set in the DDR SDRAM mode register.

DDR SDRAM devices use the SSTL-2 I/O standard and can hold between 64 Mb to 1 Gb of data. Each device is divided into four banks, and each bank has a fixed number of rows and columns. Only one row per bank can be accessed at one time. The ACTIVE command opens a row and the PRECHARGE command closes a row.

A DLL inside the DDR SDRAM device edge-aligns the DQ and DQS signals with respect to CK. The DLL must be turned on for normal operation and can be turned off to save power or for debugging purposes. All timing analyses in this document assume that the DLL is on. Some DDR SDRAM devices also have adjustable data-output drive strength. Altera recommends using the highest drive strength for maximum performance.



For more information on the DDR SDRAM specifications, go to [www.jedec.org](http://www.jedec.org).

## ALTMEMPHY and Legacy PHY Brief Overview

The legacy PHY uses a resynchronization clock with a static phase shift that is determined before design compilation, while the ALTMEMPHY implementation features a dynamic resynchronization clock that is calibrated for process (P) variations during initialization and tracks voltage and temperature (VT) variations. The ALTMEMPHY megafunction is available as a stand-alone PHY for use with third-party memory controllers. The ALTMEMPHY is also instantiated by Altera's DDR and DDR2 SDRAM High Performance Controller MegaCore® function. The legacy PHY is embedded in the DDR and DDR2 SDRAM Controller MegaCore function. Unlike ALTMEMPHY, the legacy PHY must be extracted manually from the DDR and DDR2 SDRAM Controller MegaCore function when using a third-party controller. Use the ALTMEMPHY megafunction for all new designs to achieve high performance and optimal resynchronization phase shift. All new device families after Stratix II GX support ALTMEMPHY and may not support legacy PHY. Use legacy PHY when you need a lower latency interface or when you are not using dedicated phase shift circuitry.



For more information on whether to use the legacy PHY or the ALTMEMPHY megafunction, refer to *TB 091: External Memory Interface Options for Stratix II Devices*.



For more information on the ALTMEMPHY megafunction, refer to the *ALTMEMPHY Megafunction User Guide*.



For more information on the DDR and DDR2 SDRAM High Performance Controller MegaCore function, refer to the *DDR and DDR2 SDRAM High Performance Controller MegaCore Function User Guide*.



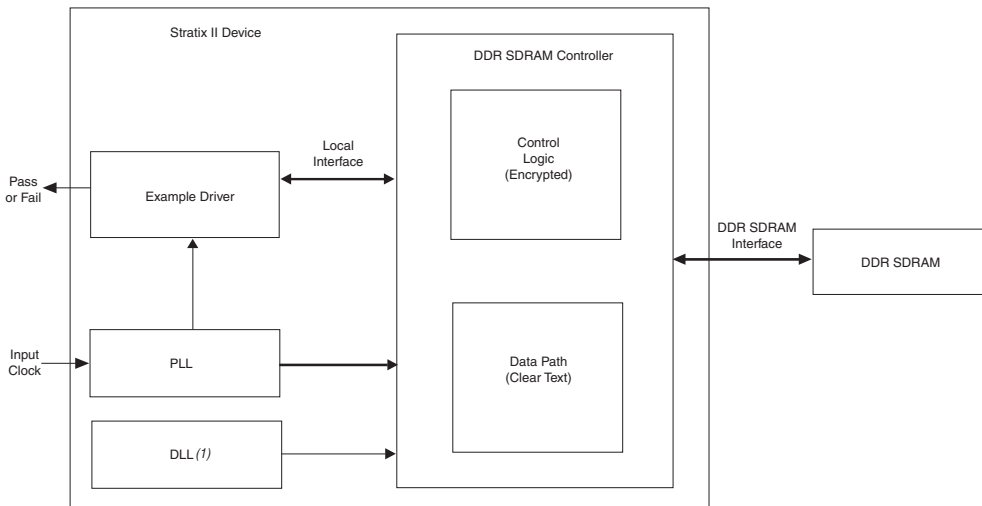
For more information on the legacy PHY and controller, refer to the *DDR and DDR2 SDRAM Controller Compiler User Guide*, respectively.

## Interface Description

This section describes the interface between the FPGA and DDR SDRAM devices. It describes the interface signals and how Altera FPGA pins should be configured to meet the DDR SDRAM electrical and timing requirements. It also lists the number of DQS/DQ pins available in the FPGA. In addition, this section also describes the architecture of the interface between the FPGA and the DDR SDRAM.

Understanding how complicated the interface can be, Altera offers a complete solution that will create the memory controller within minutes. The Altera DDR SDRAM Controller MegaCore<sup>®</sup> function has two components: a clear text data path block and an encrypted controller logic block. This license-free data path block is the recommended physical layer interface that can be used with your own controller or the Altera controller. When you use this provided data path, you are ensured a working system as the DDR SDRAM IP ToolBench constrains your interface pins and data path logic for optimal operation. [Figure 1](#) shows the block diagram for the FPGA to DDR SDRAM interface.

**Figure 1. DDR SDRAM Controller MegaCore System Level Block Diagram**



**Note to [Figure 1](#):**

- (1) When using the dedicated DQS circuitry, the DLL center-aligns the DQS strobe to the DQ data bus during read operations. When not using the dedicated DQS circuitry, a PLL implements this phase shift.

## Interface Signals

Table 2 shows the DDR SDRAM interface pins and how to connect them to Stratix II device pins.

<i>Table 2. DDR SDRAM Interface Pins</i>		
<b>Pin Type</b>	<b>Description</b>	<b>Stratix II Pin Utilization</b>
DQ	Bidirectional read and write data	DQ
DQS	Bidirectional read and write data strobe	DQS
CK	System clock	User I/O pin
CK#	System clock	User I/O pin
FB_CLK	Copy of CK output clock signal fed back to PLL for resynchronization or DLL-based read capture	PLL clock input pin
DM	Write data mask, edge-aligned to DQ during write	User I/O pin
All other	Addresses and commands	User I/O pin

This section provides a description of the clock, command, address, and data signals of a DDR SDRAM memory interface.

### *Clocks Signals*

The DDR SDRAM device uses CK and CK# to clock commands and addresses into the memory. The memory also uses these clock signals to generate the DQS signal during a read operation via a DLL inside the memory. The skew between CK or CK# and the SDRAM-generated DQS signal is specified as  $t_{DQSCX}$  in the DDR SDRAM data sheet.

The DDR SDRAM has a write requirement  $t_{DQSS}$ , which states the positive edge of DQS on writes must be within  $\pm 25\%$  ( $\pm 90^\circ$ ) of the positive edge of the DDR SDRAM clock input. Because of this, you should use the I/O element's (IOE's) DDR register to generate the CK and CK# signals to match with the DQS signal and reduce any process, voltage, and temperature variations.

To improve resynchronization for the 200-MHz DDR SDRAM interfaces, you can route a copy of the CK signal from the memory pin back to the Stratix II device. Refer to [“Round Trip Delay Calculation” on page 35](#) for resynchronization details.

### *Strobes, Data, DM and Optional ECC Signals*

Both DQ and DQS are bidirectional (the same signals are used for both writes and reads). A group of DQ pins is associated with one DQS pin. In  $\times 8$  and  $\times 16$  DDR SDRAM devices, one DQS pin is associated with 8 DQ pins (Stratix II  $\times 8/\times 9$  mode). Use the DQS pins and their associated DQ pins listed in the Stratix II pin table when interfacing with DDR SDRAM from Stratix II I/O banks 3, 4, 7, or 8. These I/O banks also offer DQSn pins, but these pins are not necessary for DDR SDRAM interfaces. Refer to [“Data Path Architecture Using Dedicated DQS Circuitry” on page 11](#) for more information. With dedicated circuitry, Stratix II devices can support up to 200-MHz DDR SDRAM.

Any of the user I/O pins in I/O banks 1, 2, 5, or 6 can also interface with DDR SDRAM memory. However, since these I/O banks do not have dedicated DQS circuitry, they can only support up to 150-MHz DDR SDRAM interfaces. The DQS signal is ignored during reads in this scheme. For more information, refer to [“Data Path Architecture Without Using Dedicated DQS Circuitry” on page 14](#).

The width of the memory data bus supported by Stratix II devices is only limited by the number of DQ/DQS pins available in the device.

Stratix II devices offer DQ/DQS pins on all sides of the device, however the VIO banks support DLL-based higher performance interfaces. For interfaces running at lower speeds a PLL-based implementation is supported on all Stratix II I/O banks.

For high-performance interfaces, use all DQ groups on one VIO side of the device. When all DQ groups on that side are used up, use the DQ groups on the opposite VIO side. Note that the DQS/DLL signals may not be shared between the top and bottom. Hence, each VIO side should have its own DLL instantiation.

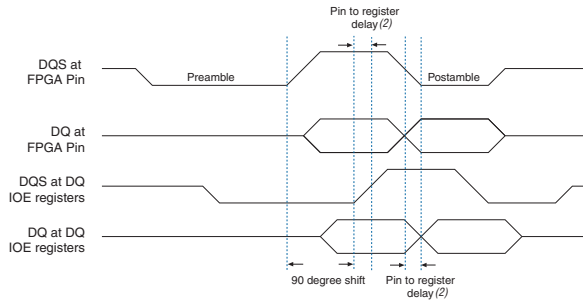
If your interface performance requirements can be met by the Stratix II PLL-based implementation, use DQ groups on one side of the device. When your interface width requirements exceed the number of groups supported on that side, allocate DQ groups on one of the remaining banks. Note that PLLs used for read capture must be placed on the same side as the DQ input pins for best source-synchronous compensation and timing margins.

Refer to [Table 3](#) for the number of DQS/DQ groups supported in Stratix II devices.

<b>Table 3. Number of DQS and DQ Groups in Stratix II Devices</b>	
<b>Package</b>	<b>Number of <math>\times 8/\times 9</math> Groups</b>
484-pin FineLine BGA	4
672-pin FineLine BGA	8
484-pin Hybrid FineLine BGA	4
780-pin FineLine BGA	8
1,020-pin FineLine BGA	18
1,508-pin FineLine BGA	18

The data signals (DQ) are edge-aligned with the DQS signal during a read from the memory and are center-aligned with the DQS signal during a write to the memory. The memory controller shifts the DQS signal by 90° during a write to center align the DQ and DQS and shifts the DQS during a read, so that DQ and DQS are center-aligned at the capture register. Stratix II devices use a PLL to center-align the DQS signal with respect to the DQ signals during writes, and use either the dedicated DQS phase-shift circuitry or a PLL to shift the incoming DQS signal during reads based on the implementation you chose. [Figure 2](#) shows an example where the DQS signal is shifted by 90° during a burst-of-two read. [Figure 3](#) shows an example of the relationship between the data and data strobe during a burst-of-two write.

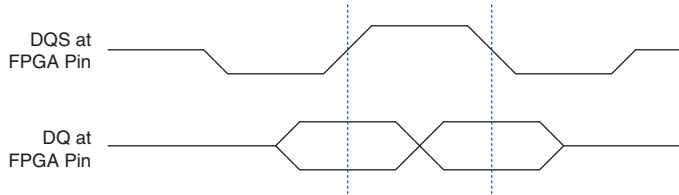
**Figure 2. DQ & DQS Relationship During a DDR SDRAM Read *Note (1)***



**Notes to Figure 2:**

- (1) This is an example of a 90° shift. The shift value read for your system should be based on your timing analysis and may not be 90°.
- (2) The delay from the DQS pin to the capture register and DQ pin to the capture register can be configured to minimize additional skew between these signals at the IOE registers.

**Figure 3. DQ & DQS Relationship During a DDR SDRAM Write**



The memory device's setup ( $t_{DS}$ ) and hold times ( $t_{DH}$ ) for the DQ and DM pins during a write are relative to the edges of DQS write signals and not the CK or CK# clock. These times are equal ( $t_{DS} = t_{DH}$ ) and typically 0.4 ns for a 200-MHz DDR SDRAM device.

The DQS signal is normally generated on the positive edge of system clock (because of the  $t_{DQSS}$  requirement described shortly). The DQ and data mask (DM) signals are clocked using a  $-90^\circ$  shifted clock from the system clock. The edges of DQS are centered on the DQ and DM signals when they arrive at the DDR SDRAM.



The DQS, DQ, and DM board trace lengths should be tightly matched.

The DDR SDRAM uses the data mask (DM) pins during a write. Driving the DM pins low marks that the write is valid. The memory will mask the DQ signals if the DM pins are driven high. You can use any of the I/O pins in the same bank as the associated DQS and DQ pins to generate the DM signal.

The DM timing requirements at the DDR SDRAM input are identical to those for DQ data. The DDR registers, clocked by the  $-90^\circ$  shifted clock, create the DM signals.

Some DDR SDRAM devices support error correction coding (ECC), to detect and automatically correct error in data transmission. The 72-bit DDR SDRAM modules contain eight ECC pins in addition to 64 data pins. Connect the DDR SDRAM ECC pins to a Stratix II device DQS and DQ group. You should create a 72-bit DDR SDRAM memory controller and add logic to decode/encode the ECC bits on the local interface of the controller.

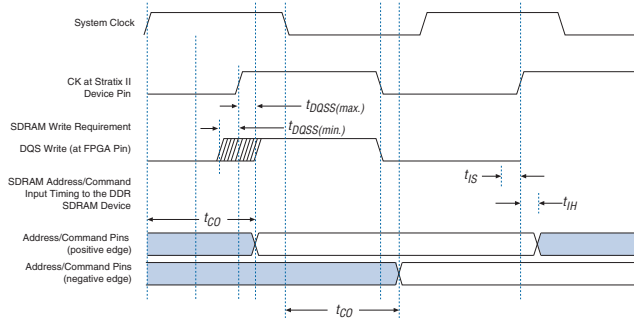
### *Commands & Addresses*

Commands and addresses in DDR SDRAM devices are clocked into the memory using the CK and CK# signal at single data rate using only one clock edge. DDR SDRAM devices have twelve to fourteen address pins, depending on the device capacity. The address pins are multiplexed, so two clock cycles are required to send the row, column, and bank addresses. The CS, RAS, CAS, and CKE pins are DDR SDRAM command pins.

The DDR SDRAM address and command inputs both require the same setup and hold times with respect to the DDR SDRAM clock. The Stratix II device address and command signals change at the same time as the DQS write signal since they are both generated from the system clock. The positive edge of the DDR SDRAM clock, CK, is aligned with DQS to satisfy  $t_{DQSS}$ . If the command and address outputs are generated on the clock's positive edge, they may not meet the setup and hold time requirements (see [Figure 4](#)). Therefore, you should use the negative edge of the system clock or a phase-shifted version of the system clock for the commands and addresses to the DDR SDRAM. This clock edge selection is made based on timing analysis with accurate pin loading information. The timing analysis methodology for address and command signals is very similar to the write timing paths described on [page 31](#). You can use any of the I/O pins for the commands and addresses.

Figure 4 shows the address and command timing and the DDR SDRAM  $t_{DQSS}$ ,  $t_{IS}$ ,  $t_{IH}$  timing requirements.

**Figure 4. DQS vs CK, Address & Command Timing Notes (1), (2)**



**Notes to Figure 4:**

- (1) The address and command timing shown in Figure 4 is applicable for both reads and writes.
- (2) If the board trace lengths for the DQS, CK, address, and command pins are the same, the signal relationships at the Stratix II device pins are maintained at the DDR SDRAM pins.

## Interface Architecture

There are two data paths or physical interface (PHY) available in Stratix II and Stratix II GX devices; the legacy PHY and the ALTMEMPHY megafunction. For highest performance, use the ALTMEMPHY megafunction.



For more information about this implementation, refer to the [ALTMEMPHY Megafunction User Guide](#). This section only describes the legacy PHY architecture.



If you are not sure which PHY to use, refer to [Technical Brief 091; External Memory Interface Options for Stratix II Devices](#).

In addition, the legacy PHY offers two read-side implementations. There are DLL-based mode implementation and PLL-based mode implementation. The DLL-based mode implementation always gives a higher performance than the PLL-based implementation. This is because the PLL-based implementation ignores the DQS strobes during reads. In the DLL-based mode implementation, the Stratix II DLL-based dedicated DQS circuitry to phase-shift the strobe signals (DQS) and center-align the strobe signal with the read data (DQ). While there are up to 144 DQ pins available in the dedicated DQS mode, users who require additional support have the option to generate the phase-shifted clock from a PLL instead of using the dedicated strobe signal (DQS) to capture the read data.

The write-side implementation suggested requires a PLL output two clocks; one to generate the write data and one to generate the write clock using DDR I/Os. This implementation results in matched propagation delays for clock and data signals from the FPGA to the DDR SDRAM, minimizing skew.


#### *Data Path Architecture Using Dedicated DQS Circuitry*

The DDR SDRAM interface implementation using dedicated DQS circuitry uses the following:

- A write-side PLL to generate CK and CK# system clocks and clock out address, command, strobe and data signals.
- A read-side DLL-based phase-shift circuitry to register read data from the memory using strobe signals, DQS.

This implementation is also called DQS mode or DLL-based read implementation.

There is one DQS phase-shift circuit available on the top of the device and one on the bottom of the device. Each DQS phase-shift circuit requires an input reference clock. The DQS phase-shift circuitry shifts the DQS signal to center-align the signal with the DQ signal at the IOE register, ensuring the data will get latched at the IOE register. The DQS signal is then inverted before going to the DQ IOE clock ports as described in the *External Memory Interfaces in Stratix II and Stratix II GX* chapter of the *Stratix II Device Family Handbook*.

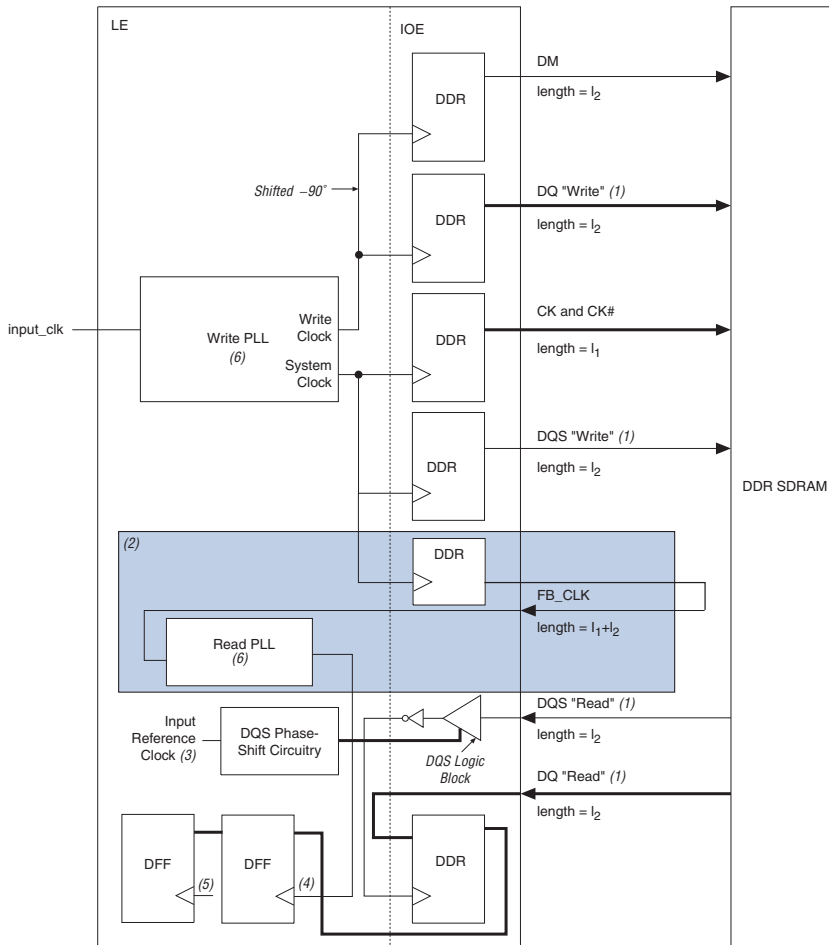
 The `areset` signal of the read PLL must be toggled after power up and the system PLL is locked.

 If the PLL loses lock, you need to reset the PLL.

**Figure 5** shows how the Stratix II device generates the DQ, DQS, CK, and CK# signals. The write PLL generates the system clock and  $-90^\circ$  shifted clock (write clock). If the write PLL input clock and the DDR SDRAM frequencies are different, you must provide the input reference clock to the DQS phase-shift circuitry either from another input clock pin or from either PLL 5 or 6 (see the *External Memory Interfaces in Stratix II and Stratix II GX Devices* chapter of the *Stratix II Device Family Handbook* for details). The system clock and write clock have the same frequency as the DQS frequency. The write clock is  $-90^\circ$  shifted from the system clock.

**Figure 5** also shows a feedback clock (`FB_CLK`) and a second (read) PLL for resynchronization. This feedback clock is optional and is only used for DDR SDRAM interfaces at 200 MHz. The board trace length for the feedback clock from the memory should be the sum of the board trace lengths for the DQ/DQS and CK/CK# signals. In **Figure 5**, `FB_CLK` mirrors the DDR SDRAM CK pin and routed back to the Stratix II device with a board trace length of  $l_2$ , which is the same length as the DQS or DQ board trace.

Figure 5. Stratix II DDR SDRAM Interface Data Path for DLL-Based Read

**Notes to Figure 5:**

- (1) DQ and DQS signals are bidirectional. One DQS signal is associated with a group of DQ signals.
- (2) The feedback clock implementation is optional to help ease resynchronization for 200-MHz interfaces.
- (3) The input reference clock can either be from the `input_clk`, another clock pin, or the PLL 5 or 6 output.
- (4) The clock to the resynchronization register can be from the system clock, write clock, an extra clock output from the write PLL or from a second (read) PLL in the feedback clock implementation. The figure shows this to be from the read PLL when the feedback clock is used.
- (5) The clock to this register can either be the system clock or another clock output of the write PLL. If the design needs another write PLL output, another register is needed to transfer the data back to the system clock domain.
- (6) The read and write PLLs are configured in normal mode.

### *Data Path Architecture Without Using Dedicated DQS Circuitry*

When using the PLL-based read implementation, Stratix II can support up to 150-MHz DDR SDRAM where each interface uses two PLLs for best performance (see [Figure 6 on page 16](#)).

The DDR SDRAM interface implementation without using dedicated DQS circuitry uses the following:

- A write-side PLL to generate CK and CK# system clocks and clock out address, command, strobe and data signals.
- A read-side PLL-based phase-shift to register read data from the memory using a feedback clock, FB\_CLK.

This implementation is also called PLL-based read implementation (or non-DQS mode).

Refer to the Stratix II pin table for the recommended DQS and DQ pins in this mode. The board trace lengths for the DQ, DQS, and DM pins should be tightly matched.

The write PLL generates the system clock, the  $-90^\circ$  shifted clock (`write_clk`), and the feedback clock (`FB_CLK`). The feedback clock is routed outside the FPGA and back into the FPGA. This board trace length should equal the clock trace length from FPGA to the memory plus the DQ trace length from the memory to the FPGA. In [Figure 6](#), assuming that clock trace length =  $l_1$  and DQ trace length =  $l_2$ , the `FB_CLK` needs to have a trace length of  $l_3 = (l_1 + l_2)$ . Matched trace lengths minimize skew across process, voltage, and temperature variations.

The read PLL uses the feedback clock as the input clock and generates the clock needed to capture the DQ during reads. The Stratix II device ignores the DQS signal in this scheme. The read PLL is in source synchronous mode to the IOE register and is matched to the data propagation through the input pin, PLL, and clock network. Because the trace length of the feedback clock is the same as the CK or CK# and DQS trace, FB\_CLK coming into the FPGA will look like the DQS signal with a little bit of skew. The read PLL can then be shifted to compensate for the skew and implement the 90° PLL phase shift required to capture the DQ signals during reads.

You need to ensure that the read PLL is in the same side of the device as the data pins are in because the Quartus II software associates a particular PLL with a particular I/O bank for source synchronous operation. The clock delay to the worst case I/O registers in this I/O bank are fully compensated and result in closely matched data delays and clock delays from the pin to the I/O registers across PVT. When using I/O registers in the non-compensated I/O banks, clock delays, and data delays are less closely matched. Use a Fast PLL for implementing the interface on side I/O banks, and use an Enhanced PLL for implementing the interface on top or bottom I/O banks for best clock and data delay matching. You also need to set the input pin delay to register option to 0 in the Quartus II software.

Since the DQS signal is ignored during reads, the DQS-DQ relationship from the DDR SDRAM device no longer applies. The skew and timing variations on the interface determines the maximum data rate you can achieve with this method.

Figure 6 shows a summary of how Stratix II devices generate the DQ, DQS, CK, and CK# signals. The write PLL generates system clock and write clock. The read clock FB\_CLK from the DDR SDRAM device goes to a PLL input pin which generates the proper phase shift to capture read data.





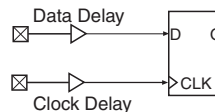
This application note describes Altera's recommended timing analysis methodology using write and read capture timing paths as examples. You should use this methodology for analyzing timing for all applicable timing paths (including address/command, resynchronization, postamble, etc.). To ensure successful operation, the Altera DDR SDRAM Controller MegaCore function performs timing analysis on the read capture, resynchronization, and postamble paths. While these analyses account for all FPGA related timing effects, you should design in adequate margin to account for board level effects such as crosstalk, inter-symbol interference and other noise effects. These effects and their impact on timing margins are best analyzed by performing board level simulations. Simulation results can be used to adjust the board timing requirement parameter,  $t_{EXT}$  for use in margin calculations.

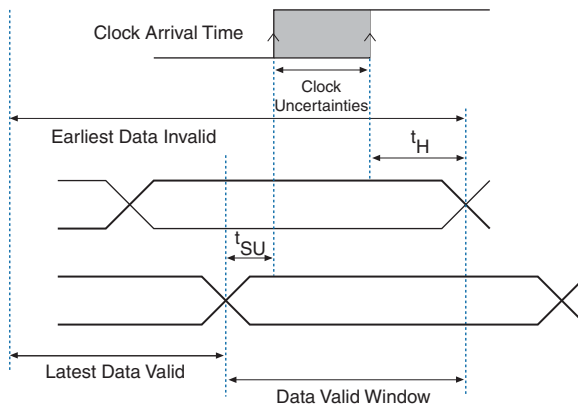
The following presents an analysis of the read and write capture timing margins for the Micron MT9VDDT3272AG-40B DDR SDRAM interface with a EP2S60F1020C3 FPGA. This sample interface shows you the proper methodology for this timing analysis. You will need to include the appropriate timing specifications for your preferred FPGA and memory device.

## Methodology Overview

Timing paths are analyzed by considering the data and clock arrival times at the destination register. In [Figure 7](#) and [Figure 8](#), the setup margin is defined as the time between “earliest clock arrival time” and “latest valid data arrival time” at the register ports. Similarly, hold margin is defined as the time between “earliest invalid data arrival time” and the “latest clock arrival time” at the register ports. These arrival times are calculated based on propagation delay information with respect to a common reference point (such as a DQS edge or system clock edge).

**Figure 7. Simplified Block Diagram for Timing Analysis**



**Figure 8. Data Valid Window Timing Waveform**

### FPGA Timing Information

Since your design needs to work under all conditions, timing margins should be evaluated at all process, voltage, and temperature (PVT) conditions. To facilitate this, Altera provides two device timing models in the Quartus II software: slow corner model and fast corner model.

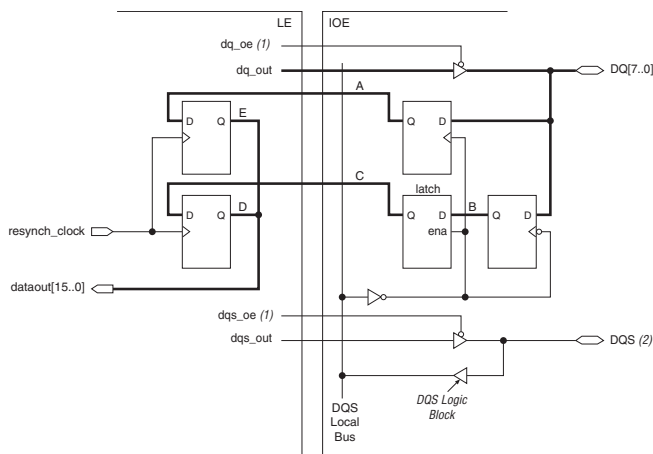
- The slow corner model provides timing delays between two nodes within the FPGA with slow silicon, high temperature, and low voltage. In other words, the model provides the slowest possible delay for that timing path on any device for that particular speed grade.
- The fast corner model provides timing delays between two nodes within the FPGA with fast silicon, low temperature, and high voltage. In other words, the model provides the fastest possible delay for that timing path on any device.

Note that while almost all FPGA timing delays and uncertainties are modeled in the Quartus II software, a limited number of uncertainties that cannot be modeled are published in the FPGA handbook for use in margin calculations. These data sheet specifications are based on device characterization and account for device level effects such as on-chip variation, rise/fall mismatch, and noise. Some examples include clock skew and jitter on PLL and DLL outputs. These timing uncertainties or adder terms, when used in conjunction with the reported timing data reported Quartus II software, provides the most accurate device timing information. The following analysis will detail the use of these timing adder terms.

## Read Timing Margins for DLL-Based Implementation

During read operations, the DDR SDRAM provides a clock strobe (DQS) that is edge-aligned with the data bus (DQ). The memory controller (in the FPGA) is required to shift the clock edge to the center of the data valid window and capture the DQ input data. Figure 2 on page 8 illustrates the timing relationship between the DQS and DQ signals during a read operation. Figure 9 shows a more detailed picture of the Stratix II device read data path for  $\times 8$  mode. The DQS signal goes to the DQS logic block and is phase-shifted. The DQS local bus then inverts the shifted DQS signal before it clocks the DQ at the input registers. The DQ input register outputs then go to the resynchronization register in the logic array. The `resynch_clock` signal clocks the resynchronization register. The `resynch_clock` signal can come from the system clock, the write clock, the write PLL clock, or the second (read) PLL (if you use the feedback clock scheme).

**Figure 9. DDR SDRAM Read Data Path in Stratix II Devices**



**Notes to Figure 9:**

- (1) The output enable registers are not shown here, but `dqs_oe` and `dq_oe` are active low in silicon. However, the Quartus II software implements it as active high and adds the inverter automatically during compilation.
- (2) Figure 9 does not show the DQS postamble circuitry.



For more information on the DQ and DQS paths, refer to *External Memory Interfaces in Stratix II and Stratix II GX Devices* chapter in volume 2 of the *Stratix II Device Handbook*.

Table 5 displays a sample DDR SDRAM read timing margin analysis for the EP2S60F1020C3 over worst case conditions. The board trace variations for the DQ and DQS pins is  $\pm 20$  ps. You can perform a similar timing analysis for your interface with another DDR SDRAM memory by replacing the  $t_{HP}$ ,  $t_{QHS}$ , and  $t_{DQSQ}$  values in Table 5 with those from your memory data sheet.

### *Memory Timing Parameters*

You would start the read timing analysis by obtaining the timing relationship between the DQ and DQS outputs from the DDR SDRAM memory device. Since this example analyzes timing for 200 MHz clock speeds or 400 Mbps data rates, the half clock period is 2250 ps after accounting for duty cycle distortion on the DQS strobe. This is specified as  $t_{HP}$  in the memory data sheet and is 45% of the 5000 ps clock period. Apart from  $t_{HP}$ , the memory also specifies  $t_{DQSQ}$  and  $t_{QHS}$ . The former specifies the maximum time from a DQS edge to the last DQ bit valid and the latter specifies the data hold skew factor.

With these memory timing parameters, the data valid window at the memory can be calculated to be  $t_{HP} - t_{QHS} - t_{DQSQ} = 1350$  ps. Assuming the board trace length variations amongst all DQ and DQS traces are not more than  $\pm 20$  ps, the data valid window present at the FPGA input pins is 1310 ps.

### *FPGA Timing Parameters*

FPGA timing parameters are obtained from two sources: Quartus II software timing analyzer and the Stratix II data sheet. The Quartus II software provides all clock and data propagation delays, and the data sheet specifies all clock uncertainties and skew adder terms.

Stratix II features dedicated DQS phase shift circuitry in the top/bottom IO banks of the device, which will center-align the DQS edge with respect the DQ input signals. This phase shift circuitry has a coarse and fine delay resolution. The coarse delay feature is self-compensating over PVT and has a resolution of 22.5°, 30°, or 36° of the reference clock frequency (based on the DLL mode of operation).



For detailed information on DLL operations, refer to *External Memory Interfaces in Stratix II and Stratix II GX Devices* chapter in volume 2 of the *Stratix II Device Handbook*.

Since the target memory speed is 200 MHz, you can select between DLL modes 1 (medium) and 2 (high). DLL mode 2 provides a coarse phase resolution of 30°, while mode 1 provides a 22.5° resolution. You can further fine tune this phase shift with a DLL offset implemented using uncompensated delay chains.

This analysis assumes a 67.5° phase shift on the DQS strobe (DLL mode 1), knowing that the phase shift (and DLL mode) can always be adjusted at the end of this timing analysis for balanced setup and hold margins on the read capture register.

The DQS phase shift circuitry uses a DLL to provide the self-compensating coarse delay shift. Hence, you have to account for any phase jitter and phase shift error on the DQS signal. The data sheet specifies the  $t_{DQS\_PHASE\_JITTER}$  ( $\pm 45$  ps) and  $t_{DQS\_PSERR}$  ( $\pm 38$  ps) timing parameters for three DLL delay stages.

After encountering the phase shift circuitry, the DQS signal travels on a dedicated local clock bus to the DQ capture registers. The fanout of this local clock bus could range from  $\times 4$  to  $\times 36$ . While Quartus II software provides clock propagation delays to each of these DQ register clock, un-modeled uncertainties are accounted with the  $t_{DQS\_SKEW\_ADDER}$  skew adder term listed in the data sheet. For the  $\times 8$  mode used in this example, the skew adder is  $\pm 35$  ps.

To obtain Quartus II software timing data for the target device, you should instantiate and compile the DDR SDRAM Controller MegaCore. If you are using your own controller logic, you should instantiate the clear-text DDR SDRAM data path instead to obtain timing delays. For the read interface, the MegaCore function extracts and reports timing delays associated with each DQ and DQS pin in the `<core_instance_name>_extraction_data.txt` file located in your project directory. Using this data file and the `extract.tcl` utility, minimum and maximum propagation delays on the clock and data path are extracted and presented in [Table 4](#). This timing extraction is done with both device timing models (fast corner and slow corner). Observe that the difference between minimum and maximum delays is minimal due to the matched routing paths within the die and package.

<b>Table 4. FPGA Timing Delays for EP2S60F1020C3 from the Quartus II Software Note (1)</b>		
	<b>Fast Corner (ns)</b>	<b>Slow Corner (ns) (-3 Speed Grade)</b>
Data Delay (min.)	1.074	1.618
Data Delay (max.)	1.134	1.678
Clock Delay (min.)	1.992	2.532
Clock Delay (max.)	2.012	2.552
Micro Setup (2)	0.068	0.122
Micro Hold (2)	0.037	0.072

**Notes to Table 4:**

- (1) These delays are reported in the `<core_instance_name>_extraction_data.txt` file located in your project directory. Data Delay is the propagation delay from the each DQ pin to the input DDR register and is reported as `dq_2_ddio`. Clock Delay is the propagation delay to the DDR input registers from the corresponding DQS pin, and is calculated as `dqspin_2_dqsclk + dqsclk_2_ddio_resync`.
- (2) The micro setup and micro hold times are specified in the [DC & Switching Characteristics](#) chapter in volume 1 of the *Stratix II Device Handbook*.

**Setup & Hold Margins Calculations**

After obtaining all relevant timing information from the memory, FPGA, and board you can calculate the setup and hold margins at the DQ capture register during read operations.

With extracted timing information from the FPGA slow corner model:

$$\begin{aligned}
 \text{Earliest clock arrival time} &= \text{Minimum clock delay within} \\
 (t_{\text{EARLY\_CLOCK}}) & \quad \text{FPGA – DQS uncertainties} \\
 &= \text{Clock delay (min.)} - t_{\text{DQS\_PHASE\_JITTER}} - \\
 & \quad t_{\text{DQS\_PSERR}} - t_{\text{DQSQINT}} \\
 &= 2532 - 45 - 37.5 - 35 \\
 &= 2414.5 \text{ ps}
 \end{aligned}$$

$$\begin{aligned}
 \text{Latest data valid time} &= \text{Memory DQS-to-DQ valid + max.} \\
 (t_{\text{LATE\_DATA\_VALID}}) & \quad \text{data delay in FPGA} \\
 &= t_{\text{DQSQ}} + \text{Data delay (max.)} \\
 &= 400 + 1678 \\
 &= 2078 \text{ ps}
 \end{aligned}$$

$$\begin{aligned}
 \text{Setup margin} &= \text{Earliest clock arrival} - \text{latest data} \\
 &\quad \text{valid} - \text{micro setup} - \text{board} \\
 &\quad \text{uncertainty} \\
 &= t_{\text{EARLY\_CLOCK}} - t_{\text{LATE\_DATA\_VALID}} - \\
 &\quad \mu t_{\text{SU}} - t_{\text{EXT}} \\
 &= 2414.5 - 2078 - 122 - 20 \\
 &= 194.5 \text{ ps}
 \end{aligned}$$

Hence, the setup margin with the slow corner timing model is 195 ps. Repeating these calculations with the fast corner timing model, the setup margin is calculated to be 253 ps.

$$\begin{aligned}
 \text{Latest clock arrival time} &= \text{Max. clock delay within} \\
 (t_{\text{LATE\_CLOCK}}) &\quad \text{FPGA} + \text{DQS uncertainties} \\
 &= \text{Clock delay (max.)} + t_{\text{DQS\_PHASE\_JITTER}} + \\
 &\quad t_{\text{DQS\_PSERR}} + t_{\text{DQS\_QINT}} \\
 &= 2552 + 45 + 37.5 + 35 \\
 &= 2669.5 \text{ ps}
 \end{aligned}$$

$$\begin{aligned}
 \text{Earliest data invalid time} &= \text{Memory DQS-to-DQ invalid} + \\
 (t_{\text{EARLY\_DATA\_INVALID}}) &\quad \text{minimum data delay in FPGA} \\
 &= (t_{\text{HP}} - t_{\text{QHS}}) + \text{Data delay (min.)} \\
 &= (2250 - 500) + 1618 \\
 &= 3368 \text{ ps}
 \end{aligned}$$

$$\begin{aligned}
 \text{Hold margin} &= \text{Latest clock arrival time} - \text{earliest} \\
 &\quad \text{data invalid time} - \text{micro hold} - \\
 &\quad \text{board uncertainty} \\
 &= t_{\text{EARLY\_DATA\_INVALID}} - t_{\text{LATE\_CLOCK}} \\
 &\quad - \mu t_{\text{H}} - t_{\text{EXT}} \\
 &= 3368 - 2669.5 - 72 - 20 \\
 &= 606.5 \text{ ps}
 \end{aligned}$$

Hence, hold margin with the slow corner timing model is 607 ps. Repeating these calculations with the fast corner model, the hold margin is calculated to be 638 ps. The setup and hold margins can be balanced by using positive delay offset.

**Table 5. Read Timing Analysis for 200-MHz DDR SDRAM Interface in EP2S60F1020C3 using Dedicated DQS Circuitry (Part 1 of 2)**

Parameter	Specification	Fast Corner Model	Slow Corner Model	Description
Memory specifications (1)	$t_{HP}$	2.250	2.250	Half period as specified by the memory data sheet (including memory clock duty cycle distortion)
	$t_{DQSQ}$	0.400	0.400	Skew between DQS and DQ from the memory
	$t_{QHS}$	0.500	0.500	Data hold skew factor as specified by the memory data sheet
FPGA specifications (2), (4), (5)	$t_{DQS\_PHASE\_JITTER}$	0.045	0.045	Phase jitter on DQS output delayed by DLL (three delay stages = $\pm 3 \times 15$ )
	$t_{DQS\_PSERR}$	0.038	0.038	Phase Shift Error on DQS output delayed by DLL (three delay stages)
	$t_{DQS\_SKEW\_ADDER}$	0.035	0.035	Clock Delay Skew Adder for $\times 8$
	Minimum Clock Delay (Input)	1.992	2.532	Minimum DQS pin to IOE register delay from Quartus II software (with 67.5° DLL-based phase shift)
	Max. Clock Delay (Input)	2.012	2.552	Max. DQS pin to IOE register delay from Quartus II software (with 67.5° DLL-based phase shift)
	Min. Data Delay (Input)	1.074	1.618	Min. DQ pin to IOE register delay from Quartus II software
	Max. Data Delay (Input)	1.134	1.678	Max. DQ pin to IOE register delay from Quartus II software
	$\mu t_{SU}$	0.068	0.122	Intrinsic setup time of the IOE register
	$\mu t_{H}$	0.037	0.072	Intrinsic hold time of the IOE register
Board specifications	$t_{EXT}$	0.020	0.020	Board trace variations on the DQ and DQS lines



**Table 5. Read Timing Analysis for 200-MHz DDR SDRAM Interface in EP2S60F1020C3 using Dedicated DQS Circuitry (Part 2 of 2)**

Parameter	Specification	Fast Corner Model	Slow Corner Model	Description
Timing calculations	$t_{\text{EARLY\_CLOCK}}$	1.875	2.415	Earliest possible clock edge after DQS phase-shift circuitry and uncertainties (Min. clock delay - $t_{\text{DQS\_JITTER}}$ - $t_{\text{DQS\_PSERR}}$ - $t_{\text{DQS\_SKEW\_ADDER}}$ )
	$t_{\text{LATE\_CLOCK}}$	2.130	2.670	Latest possible clock edge after DQS phase-shift circuitry and uncertainties (Max clock delay + $t_{\text{DQS\_JITTER}}$ + $t_{\text{DQS\_PSERR}}$ + $t_{\text{DQS\_SKEW\_ADDER}}$ )
	$t_{\text{EARLY\_DATA\_INVALID}}$	2.824	3.368	Time for earliest data to become invalid for sampling at FPGA flop ( $t_{\text{HP}}$ - $t_{\text{QHS}}$ + Min. data delay)
	$t_{\text{LATE\_DATA\_VALID}}$	1.534	2.078	Time for latest data to become valid for sampling at FPGA flop ( $t_{\text{DQSQ}}$ + Max data delay)
Results	Read setup timing margin (3)	0.253	0.195	$t_{\text{EARLY\_CLOCK}} - t_{\text{LATE\_DATA\_VALID}} - \mu t_{\text{SU}} - t_{\text{EXT}}$
	Read hold timing margin (3)	0.638	0.607	$t_{\text{EARLY\_DATA\_INVALID}} - t_{\text{LATE\_CLOCK}} - \mu t_{\text{H}} - t_{\text{EXT}}$
	Total margin	0.890	0.801	Setup margin + Hold margin

**Notes for Table 5:**

- (1) The memory numbers referenced in this table are from a Micron MT9VDDT3272AG-40B clocked at 200 MHz.
- (2) This analysis is performed with FPGA timing parameters for an EP2S60F1020C3. You should use this template to analyze timing for your preferred Stratix II density-package combination. For FPGA specifications, see the *External Memory Interfaces* section in the *DC & Switching Characteristics* chapter in volume 1 of the *Stratix II Handbook*.
- (3) DLL phase shift is adjustable if you need to balance the setup and hold time margin.
- (4) These numbers are from the Quartus II software, version 5.0 using the DDR SDRAM Controller MegaCore 3.3.0.
- (5) Package trace skews are modeled by Quartus II software.

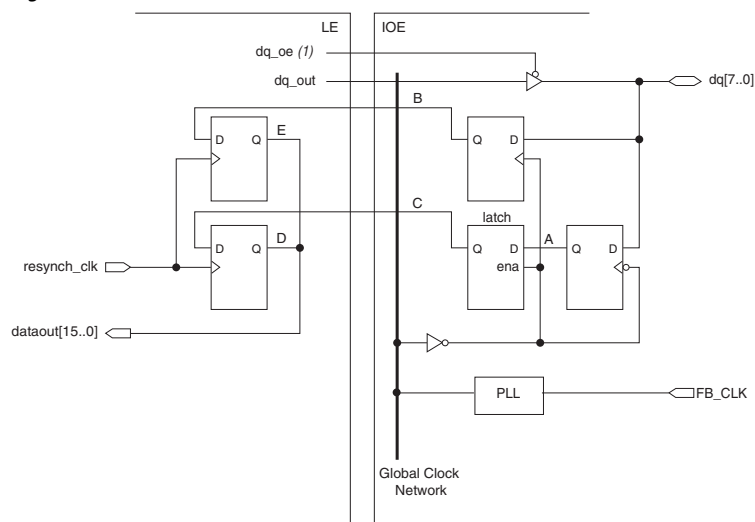
## Read Timing Margins for PLL-Based Implementation

Timing margin analysis for a PLL-based implementation is very similar to the previously described DLL-based implementation. The only differences are the capture clock used and related clock uncertainties. In this mode, a copy of the  $CK$  clock signal is fed-back to a PLL inside the FPGA.

In this example we analyze margins for a DDR SDRAM memory device for a 150 MHz read operation using a PLL.

Figure 10 shows the PLL-based read data path from the Stratix II device. The  $FB\_CLK$  signal goes to the PLL and the PLL generates a phase shifted read clock to capture the read data. The outputs of the DQ registers then go to the LE resynchronization registers. You may need multiple resynchronization registers before data is synchronized with the system clock.

**Figure 10. PLL-Based Read Data Path from the Stratix II Device**



**Note to Figure 10:**

- (1) The  $dq\_oe$  signals are active low in silicon. However, the Quartus II software adds the inverter automatically during compilation.

### Memory Timing Parameters

The timing relationship of data (DQ) with respect to the CK clock is governed by the  $t_{AC}$  parameter. For the DDR-333 memory device under consideration, this timing parameter is  $\pm 700$  ps. This memory parameter replaces the  $t_{DQSQ}$  and  $t_{QHS}$  parameters used in the DLL based implementation.

### FPGA Timing Parameters

When the CK clock is fed-back into the PLL for read capture, uncertainties introduced on this clock include jitter, phase shift error, and compensation error. While jitter and phase shift error parameters have been defined before, the compensation error is a measure of the PLL's ability to regenerate an clock output that tracks the input reference. For the source synchronous mode of the PLL, this parameter is typically  $t_{PLL\_COMP\_ERROR} = \pm 100$  ps.



For more information on *PLL specifications*, refer to *PLLs in Stratix II and Stratix II GX Devices* chapter in volume 2 of the *Stratix II Handbook*.

PLL-based read implementation uses a single global clock network to distribute the phase shifted clock signal to DQ capture registers in the IOE. Differences in clock arrival times to these registers (clock skew) is modeled in Quartus II software, and is reflected in the minimum or maximum propagation delays for the clock. Additionally, Quartus II software models the package trace delays for every pin in the device. Hence, this analysis does not account for such skews separately in the timing margin analysis. The extracted minimum or maximum clock and data delays account for these uncertainties.

To obtain the Quartus II software timing data for the target device, instantiate and compile the DDR Controller MegaCore function. If you are using your own controller logic, you should instantiate the clear text DDR data path instead to obtain timing delays. For the read interface, the Quartus II software reports individual setup and hold times for each DQ pin. Select the "List Paths" option in the timing report to get the data and clock propagation delays for that DQ pin. Select the worst case setup and hold DQ registers to extract the minimum and maximum propagation delays.

A List Paths example on the setup time for DQ[4] is shown below. This path shows propagation delays of 0.979 ns on the DQ pin to register path, and 2.491 ns (-0.419 + 2.910) on the DQS clock pin to register path. Using this approach, minimum and maximum propagation delays on the clock and data path are extracted and presented in [Table 6](#). This timing extraction is done twice, once with each device model (fast model and slow model.)

```
Info: tsu for register
"my_ddr_core:my_ddr_core_ddr_sdram|...|dq_captured_falling[4]" (data pin =
"ddr_dq[4]", clock pin = "feedback_clk_in") is -1.390 ns
    Info: + Longest pin to register delay is 0.979 ns
    Info: + Micro setup delay of destination is 0.122 ns
    Info: - Offset between input clock "feedback_clk_in" and output clock
"ddr_pll_fb_stratixii:g_stratixpll_ddr_feedback_pll_inst|altpll:altpll_component
|_clk0" is -0.419 ns
    Info: - Shortest clock path from clock
"ddr_pll_fb_stratixii:g_stratixpll_ddr_feedback_pll_inst|altpll:altpll_component
|_clk0" to destination register is 2.910 ns
```

Other than the “List Paths” for the read interface, the MegaCore function also extracts and reports timing delays associated with each DQ and DQS pin in the <core\_instance\_name>\_extraction\_data.txt file located in your project directory. Using this data file and the **extract.tcl** utility, minimum and maximum propagation delays on the clock and data path are extracted and presented in [Table 7](#). These timing extraction data are data delay(min.), data delay(max.), clock delay(min.), and clock delay(max.) for both device timing models (fast corner and slow corner). Observe that the difference between minimum and maximum delays is minimal due to the matched routing paths within the die and package.



For the determining of data delay and clock delay, refer to [Note \(1\)](#) of [Table 5](#). FPGA Timing Delays for EP2S60F1020C3 from the Quartus II Software.

[Table 6](#) shows the calculated data valid window at the FPGA pins based on 150 MHz for the timing analysis. The IP ToolBench utility performs a similar timing margin analysis for your DDR SDRAM Controller MegaCore instance.

**Table 6. Read Timing Analysis for 150-MHz DDR SDRAM interface in EP2S60F1020C3 Without Using Dedicated DQS Circuitry (Part 1 of 2)**

Parameter	Specification	150-MHz Fast Model(1)	150-MHz Slow Model(1)	Description
Memory specifications	$t_{HP}$	3.000	3.000	Half period as specified by the memory data sheet
	$t_{AC}$	0.700	0.700	Data (DQ) output access time from memory clock (CK) for DDR 333Mbps device
FPGA specifications	PLL phase shift (2)	1.458	1.458	PLL phase shift to capture data (this is using 79°)
	$t_{PLL\_JITTER}$ (5)	0.125	0.125	Stratix II PLL jitter
	$t_{PLL\_COMP\_ERROR}$	0.100	0.100	PLL Compensation Error (high bandwidth)
	$t_{PLL\_PSERR}$ (5)	0.015	0.015	PLL Phase Shift Error
	$t_{CO\_SKEW}$ (CK and FB_CLK)	0.045	0.046	Variations of the CK and feedback clock signals clock-to-out times from the Quartus II software. (4)
	Min. Clock Delay (3)	0.546	1.033	Min. feedback clock pin to IOE register delay from Quartus II software
	Max. Clock Delay(3)	0.567	1.076	Max. feedback clock pin to IOE register delay from Quartus II software
	Min. Data Delay (3)	0.579	0.883	Min. DQ pin to IOE register delay from Quartus II software
	Max. Data Delay (3)	0.673	0.979	Max. DQ pin to IOE register delay from Quartus II software
	$\mu t_{SU}$	0.068	0.122	Intrinsic setup time of the IOE register
$\mu t_{H}$	0.037	0.072	Intrinsic hold time of the IOE register	
Board specifications	$t_{EXT}$	0.020	0.020	Board trace variations on the DQ and DQS lines

**Table 6. Read Timing Analysis for 150-MHz DDR SDRAM interface in EP2S60F1020C3 Without Using Dedicated DQS Circuitry (Part 2 of 2)**

Parameter	Specification	150-MHz Fast Model(1)	150-MHz Slow Model(1)	Description
Timing calculations	$t_{\text{EARLY\_CLOCK}}$	1.764	2.251	Earliest possible clock edge after DQS phase-shift circuitry and uncertainties (Min clock delay + PLL phase shift - $t_{\text{PLL\_PSERR}}$ - $t_{\text{PLL\_JITTER}}$ - $t_{\text{PLL\_COMP\_ERROR}}$ )
	$t_{\text{LATE\_CLOCK}}$	2.265	2.774	Latest possible clock edge after DQS phase-shift circuitry and uncertainties (Max. clock delay + PLL phase shift + $t_{\text{PLL\_PSERR}}$ + $t_{\text{PLL\_JITTER}}$ + $t_{\text{PLL\_COMP\_ERROR}}$ )
	$t_{\text{EARLY\_DATA\_INVALID}}$	2.834	3.137	Time for earliest data to become invalid for sampling at FPGA flop ( $t_{\text{HP}} - t_{\text{AC}}$ + Min. data delay - $t_{\text{co\_skew}}$ )
	$t_{\text{LATE\_DATA\_VALID}}$	1.418	1.725	Time for latest data to become valid for sampling at FPGA flop ( $t_{\text{AC}}$ + Max Data Delay + $t_{\text{co\_skew}}$ )
Results	Read setup timing margin	0.258	0.384	$t_{\text{EARLY\_CLOCK}} - t_{\text{LATE\_DATA\_VALID}} - \mu t_{\text{SU}} - t_{\text{EXT}}$
	Read hold timing margin	0.512	0.271	$t_{\text{EARLY\_DATA\_INVALID}} - t_{\text{LATE\_CLOCK}} - \mu t_{\text{H}} - t_{\text{EXT}}$
	Total margin	0.770	0.655	Setup + hold time margin

**Notes for Table 6:**

- (1) The memory numbers referenced in this table are from a Micron MT9VDDT3272AG-335 clocked at 150 MHz.
- (2) PLL phase shift is adjustable if you need to balance the setup and hold time margin.
- (3) These numbers are from the Quartus II software, version 5.1 using the DDR SDRAM Controller MegaCore 3.3.0. IOE capture registers are clocked by source synchronous mode PLL output.
- (4) This number represents the difference between the Quartus II software reported  $t_{\text{co}}$  for your CK and feedback clocks. The value used in your analysis can be minimized by choosing pins with closely matched  $t_{\text{co}}$  such as adjacent pins.
- (5) For FPGA PLL specifications, see the PLL Timing Specifications section of the *DC Switching Characteristics* chapter in volume 1 of the *Stratix II Device Handbook*.

## Write Data Timing Analysis

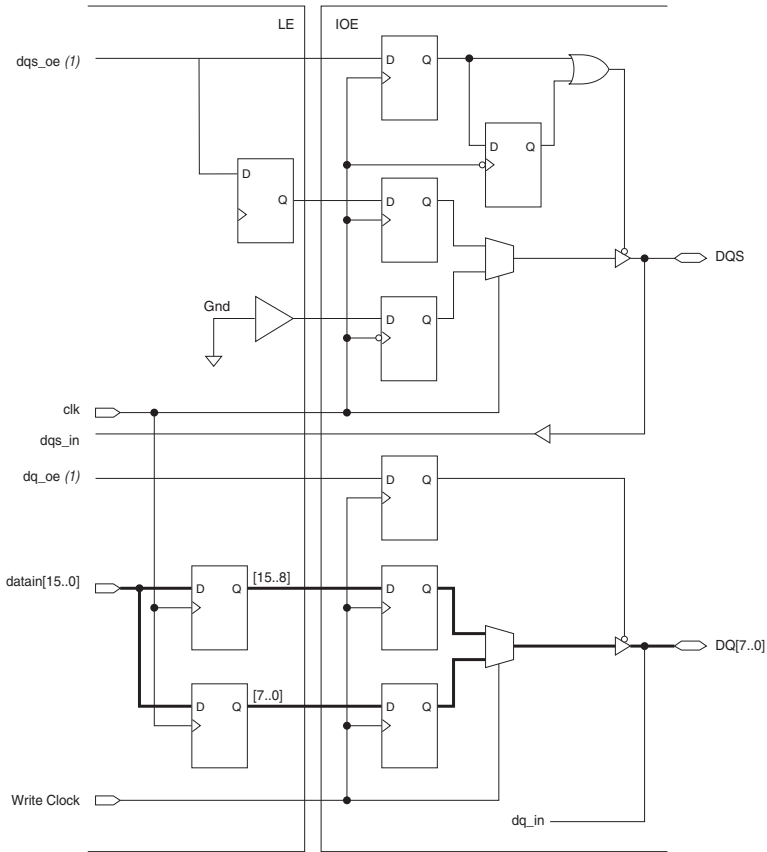
Whether you are using the DQS phase-shift circuitry or the PLL to capture data during a read operation from the DDR SDRAM device, there is only one implementation for the write operation. Timing margin analysis for write data and address/command signals are very similar. This section analyzes timing for the write data signals. You should use the same approach to repeat this for the address/command signals.

For write operations, the DDR SDRAM memory requires the clock strobe (DQS) to be center-aligned with the data bus (DQ). This is implemented in Stratix II using the PLL phase shift feature. Two output clocks are created from the PLL, with a relative 90° phase offset. The leading (−90°) clock edge is used to clock out the DQ write data output pins to the memory, while the lagging (0°) clock edge is used to generate the DQS clock strobe and CK/CK# memory output clocks. [Figure 2 on page 8](#) illustrates the timing relationship between the DQS and DQ inputs required by the memory during a write operation. As shown in [Figure 5](#), the write side uses a PLL to generate the clocks listed in [Table 7](#).

<b>Clock</b>	<b>Description</b>
System clock	This is used for the memory controller and to generate the DQS write, CK, and CK# signals.
Write clock (−90° shifted from system clock)	This is used in the data path to generate the DQ write signals.
Feedback clock	This optional clock is used if you are not using the DQS phase-shift circuitry when reading from the DDR SDRAM device or if you are using the feedback clock scheme for resynchronization.
Resynchronization clock	This optional clock is only used if you are using the DQS phase-shift circuitry and need a different clock phase shift than available for resynchronization.

Figure 11 shows the data path for DDR SDRAM write operations.

**Figure 11. Stratix II DDR SDRAM Write Data Path**



**Note to Figure 11:**

- (1) The  $dqs\_oe$  and  $dq\_oe$  signals are active low in silicon. However, the Quartus II software implements it as active high and adds the inverter automatically during compilation.



### Memory Timing Parameters

When writing to a memory, the FPGA needs to ensure that setup and hold times are met. These specifications ( $t_{DS}$  and  $t_{DH}$ ) are obtained from the data sheet (4000 ps each). Additionally, the FPGA needs to provide a memory clock (CK/CK#) that meets the clock high/low time specifications. And finally, the skew between the DQS output strobe and CK output clock cannot exceed limits set by the memory. While the last parameter does not directly affect timing margins, it needs to be met for successful memory operation.

### FPGA Timing Parameters

The timing paths within the FPGA for the DQ and DQS outputs to memory are matched by data path design. Dedicated clock networks drive DDR IO structures to generate DQ and DQS. This results in minimal skew between these outputs. These skew parameters include: phase-shift error, clock skew, and package skew.

The two clock networks used are driven by the same PLL, however with a  $90^\circ$  relative phase shift. The  $0^\circ$  clock is used to generate DQS, while a  $-90^\circ$  clock is used to generate DQ. Typical PLL uncertainties such as jitter and compensation error, affect both clock networks equally. Hence, these timing parameters do not affect write timing margins. As the clock generating DQ is phase shifted, the PLL phase shift uncertainty ( $t_{PLL\_PSEERR} = \pm 30$  ps, listed in the *DC & Switching Characteristics* chapter in volume 1 of the *Stratix II Device Handbook*) affects DQ arrival times at the memory pins.

Quartus II software models intra-clock skew, that is skew between nodes driven by the same dedicated clock network. However, skew between two such clock networks is not modeled and specified in the data sheet as an adder term. You should add this skew component to the propagation delays extracted from Quartus II software.

For a 72-bit DDR SDRAM interface that spans two IO banks in the top or bottom of the device, the clock skew adder between two clock networks is specified as  $\pm 50$  ps ( $t_{CLOCK\_SKEW\_ADDER}$ ). This uncertainty is used while calculating DQS arrival times at the memory pins.

The final skew component is package skew. As noted earlier, Quartus II software models package trace delay for each pin on the device. Extracted propagation delays reflect any skew between output signals to the memory.

Table 8 shows a sample 200MHz DDR SDRAM write timing margin analysis for an EP2S60F1020C3. The board trace variations allotted for your DQ and DQS pins is  $\pm 20$  ps. You should use this as a template to perform a similar timing analysis for your specific FPGA and DDR SDRAM.

Parameter	Specification	Fast Corner Model	Slow Corner Model	Description
Memory specifications (1)	$t_{DS}$	0.400	0.400	Memory data setup requirement
	$t_{DH}$	0.400	0.400	Memory data hold requirement
FPGA specifications (2), (4), (5)	$t_{HP}$	2.050	2.050	Ideal half period minus 5% duty cycle distortion and half-period jitter
	$t_{PLL\_JITTER}$	0.000	0.000	Does not affect margin as the same PLL generates both write clocks (0° and -90°)
	$t_{PLL\_PSERR}$	0.015	0.015	PLL phase shift error (On -90° clock output)
	$t_{CLOCK\_SKEW\_ADDER}$	0.050	0.050	Clock skew between two dedicated clock networks feeding IO banks on same side of the FPGA
	Min. Clock Delay (Output)	0.938	1.748	Min. DQS $t_{CO}$ from Quartus II software (0° PLL output clock)
	Max. Clock Delay (Output)	0.973	1.793	Max. DQS $t_{CO}$ from Quartus II software (0° PLL output clock)
	Min. Data Delay (Output)	-0.321	0.489	Min. DQ $t_{CO}$ from Quartus II software (-90° PLL output clock)
Max. Data Delay (Output)	-0.167	0.822	Max. DQ $t_{CO}$ from Quartus II software (-90° PLL output clock)	
Board specifications	$t_{EXT}$	0.020	0.020	Board trace variations on the DQ and DQS lines

**Table 8. Write Timing Analysis for 200-MHz DDR SDRAM Interface to an EP2S60F1020C3 (Part 2 of 2)**

Parameter	Specification	Fast Corner Model	Slow Corner Model	Description
Timing calculations	$t_{\text{EARLY\_CLOCK}}$	0.888	1.698	Earliest possible clock edge seen by Memory device (Min. clock delay – $t_{\text{PLL\_JITTER}}$ – $t_{\text{CLOCK\_SKEW\_ADDER}}$ )
	$t_{\text{LATE\_CLOCK}}$	1.023	1.843	Latest possible clock edge seen by Memory device (Max. clock delay + $t_{\text{PLL\_JITTER}}$ + $t_{\text{CLOCK\_SKEW\_ADDER}}$ )
	$t_{\text{EARLY\_DATA\_INVALID}}$	1.714	2.524	Time for earliest data to become invalid for sampling at the Memory input pins ( $t_{\text{HP}}$ + Min. data delay – $t_{\text{PLL\_PSERR}}$ )
	$t_{\text{LATE\_DATA\_VALID}}$	-0.152	0.837	Time for latest data to become valid for sampling at the Memory input pins (Max. data delay + $t_{\text{PLL\_PSERR}}$ )
Results	Read setup timing margin (3)	0.620	0.441	$t_{\text{EARLY\_CLOCK}} - t_{\text{LATE\_DATA\_VALID}} - t_{\text{DS}} - t_{\text{EXT}}$
	Read hold timing margin (3)	0.271	0.261	$t_{\text{EARLY\_DATA\_INVALID}} - t_{\text{LATE\_CLOCK}} - t_{\text{DH}} - t_{\text{EXT}}$
	Total margin	0.891	0.702	Setup margin + Hold Margin

**Notes for Table 8:**

- (1) The memory numbers referenced in this table are from a Micron MT9VDDT3272AG-40B clocked at 200 MHz.
- (2) This analysis is performed with FPGA timing parameters for an EP2S60F1020C3. You should use this template to analyze timing for your preferred Stratix II density-package combination. For more information on FPGA specifications, see the *PLL Timing Specifications and Clock Network Skew Adders* sections of the *DC & Switching Characteristics* chapter in Volume 1 of the *Stratix II Handbook*.
- (3) PLL phase shift is adjustable if you need to balance the setup and hold time margin.
- (4) These numbers are from the Quartus II software, version 5.1 using the Altera IP core 3.3.0.
- (5) Package trace skews are modeled by Quartus II software.

## Round Trip Delay Calculation

Read data is sent into the FPGA DDR registers using the DQS signal as a clock. Therefore, data must be transferred from the DQS clock domain to the system clock domain. This transfer of memory read data from DQS clock domain to the FPGA system clock domain is called resynchronization. You can resynchronize the data with the system clock either with or without a feedback clock. The optional feedback clock requires an extra pin and PLL. Altera recommends using the optional feedback clock for 200-MHz interfaces. If your DDR SDRAM design is running slower than 200 MHz, you do not need a feedback clock.

Analyzing timing for the resynchronization path requires round-trip delay (RTD) calculations for clock and data signals across PVT. When calculating RTD using the fast and slow timing models, a safe resynchronization window might not exist at higher speeds when using the single stage resynchronization implementation. A feedback clock implementation increases timing margins by using delay matching and multi-stage resynchronization. A typical RTD path includes delay components such as  $t_{co}$  of FPGA CK output pin, board trace delay for CK clock, memory propagation delay from CK to DQS, DQS phase shift delay inside the FPGA, and DQ propagation delay from capture register to resynchronization register. Each of these delay components can vary significantly across PVT and result in a data valid window that is severely diminished or non-existent. A feedback clock architecture uses two register stages between the memory clock domain and the system clock to split uncertainties. The clock and data delay paths to the first stage register have matched delays through the FPGA clock output pin, board trace, and FPGA input pin to register. In addition, a PLL-compensated clock network eliminates delay variation across PVT. Therefore, this 2-PLL feedback clock implementation is recommended for higher speeds. The DDR SDRAM MegaCore IP ToolBench provides timing margin analysis for resynchronization paths for both implementations. Review the timing analysis output from Quartus II software or perform a paper analysis (as described below) to select the best resynchronization implementation for your design.

### *Round Trip Delay with the Optional Feedback Clock*

The feedback clock and the second PLL shown in [Figure 5](#) ease the data resynchronization. This feedback clock is taken from the CK signal at the memory and routed back to the Stratix II FPGA to feed a second PLL, called the read PLL. This read PLL needs to be in normal mode so that the output is in phase with the PLL input (if there is no phase-shifting). The PLL input is then offset by  $\pm t_{DQSCK}$  value from the DDR SDRAM plus any board trace skew between DQS, CK, and the `FB_CLK` traces. The PLL can then compensate for the delay between IOE register to the LE register and synchronize the data from the DQS clock domain to the feedback clock domain.

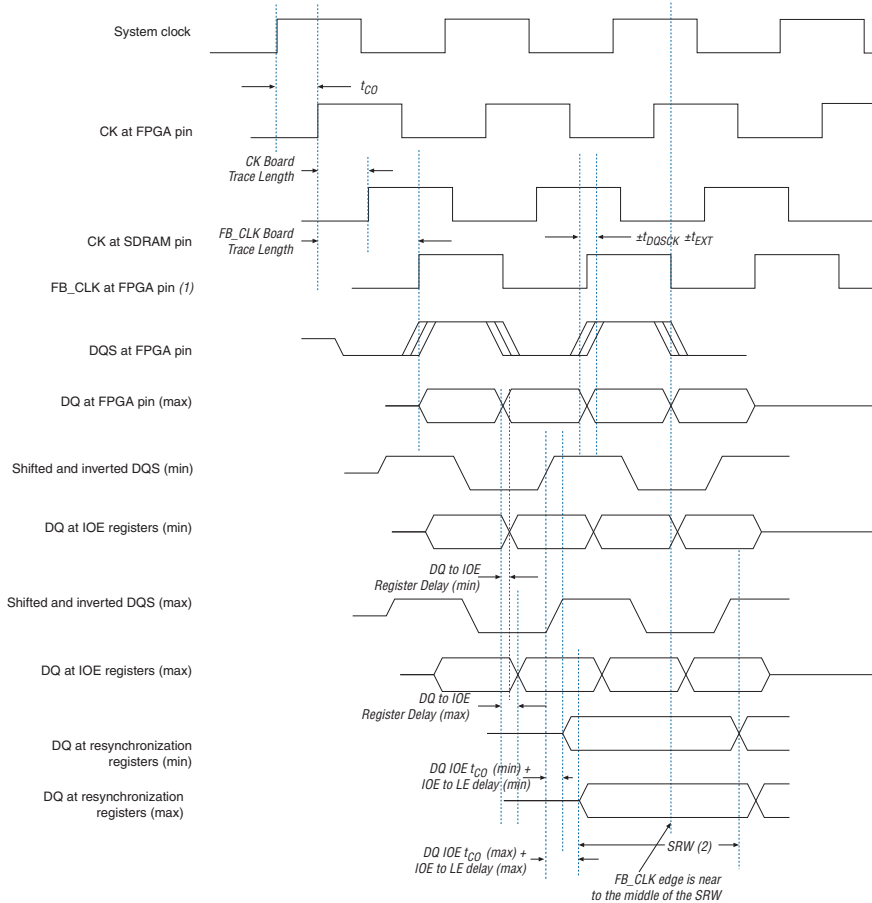
The board trace lengths cause the feedback clock to lag behind the system clock. This causes a delay on the CK and DQS signals ( $l_1 + l_2$ ). You can calculate whether the registers outputs clocked by the feedback clock need another resynchronization stage before getting to the system clock domain. In this calculation, you need to have the maximum and minimum values for the following delays:

- Clock-to-out delays for the CK signal from the Stratix II device
- CK board trace lengths

- DQS board trace lengths
- Register to register delays between the registers in the feedback clock domain and the registers in the system clock domain

Figure 12 shows an example timing waveform when using the optional feedback clock for resynchronization.

**Figure 12. Round Trip Delay Diagram with Feedback Clock Example**



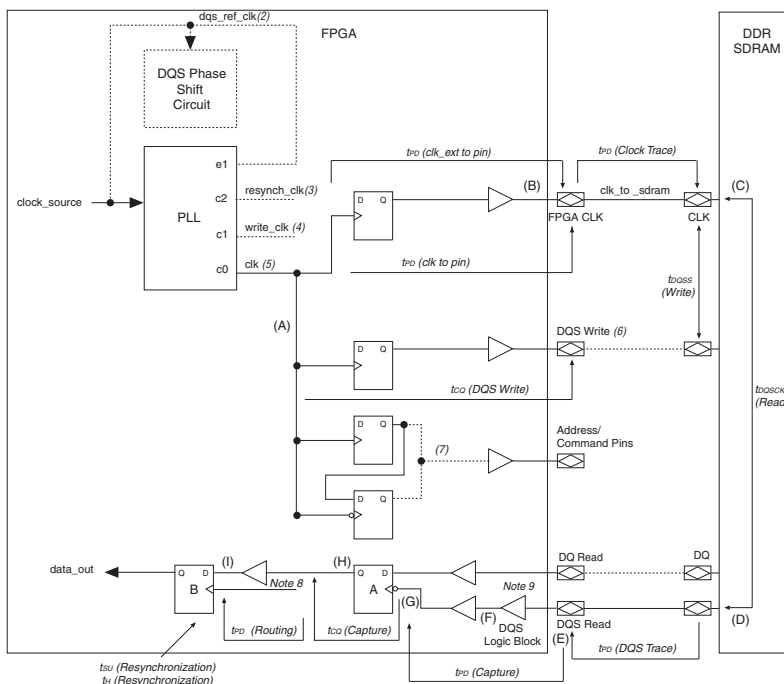
**Notes to Figure 12:**

- (1) The FB\_CLK goes to a PLL whose output can be shifted so that it is centered in the SRW.
- (2) SRW: safe resynchronization window.

### Round Trip Delay Calculation without the Optional Feedback Clock

Figure 13 shows the timing analysis and illustration of the round trip delay when the feedback clock is not used. The round trip delay is the delay from the FPGA clock to the DDR SDRAM and back to the FPGA (input to register B). This analysis is required to reliably transfer data from register A (in the IOE) to register B (in the LE).

**Figure 13. Round Trip Delay Illustration without a Feedback Clock Note (1)**



#### Notes to Figure 13:

- (1) The nodes for the round trip delay (RTD) analysis are marked with letters (A) through (I).
- (2) The `dqs_ref_clk` input for Stratix II devices can be either from an output of the PLL or directly from an input clock pin.
- (3) The `resynch_clk` is optional based on the round trip delay of the system.
- (4) The `write_clk` signal is shown for completeness, but it is not needed in the timing analysis for round-trip delay or address/command timing.
- (5) `clk` is the system clock.
- (6) The DQS signal is bidirectional. DQS write and DQS read are shown as two separate pins for this timing analysis.
- (7) You can clock the address/command register with either a rising edge or falling edge of the `clk` signal.
- (8) Register B's clock input can be `clk`, `write_clk`, or `resynch_clk`. The `clk` and `write_clk` signals can also be inverted at Register B if needed.
- (9) The DQS phase-shift reference circuit controls the 90° phase shift dynamically. The control path is not shown and its operation is user transparent.

Register A in [Figure 13](#) represents the DDR SDRAM capture logic. The Q output from register A represents the point at which the read data has been converted from DDR SDRAM to single data rate (SDR). At the output of register A, the data is already at single data rate, but is still in the DQS clock domain.  $DQ_H$  (DQ data during DQS high) is sampled on the positive edge of the  $90^\circ$  phase-shifted DQS pulse, but re-sampled on the negative edge of the  $90^\circ$  phase-shifted DQS pulse, to align it with  $DQ_L$  (DQ data during DQS low).

Once sampled by the negative edge of the  $90^\circ$  phase-shifted DQS pulse,  $DQ_L$  and  $DQ_H$  are available for resynchronization.

To sample the Q output of register A into register B, you need the time relationship between register B's clock input and the D input, which depends on the phase relationship between the DQS and clk signals and involves the following steps:

1. Calculate the systems round trip delay.
2. Select a resynchronization phase of the system clock or other available clock that reliably samples the Q output of register A, based on the calculated safe resynchronization window (SRW).
3. Apply the correct clock edge for your resynchronization logic in your memory controller.

You can use the `clk`, `write_clk` or `resynch_clk` signals as the register B clock input. You can also invert `clk` and `write_clk` if needed. To determine the timing of data at the D input of register B relative to `clk`, consider the following timing-path dependencies:

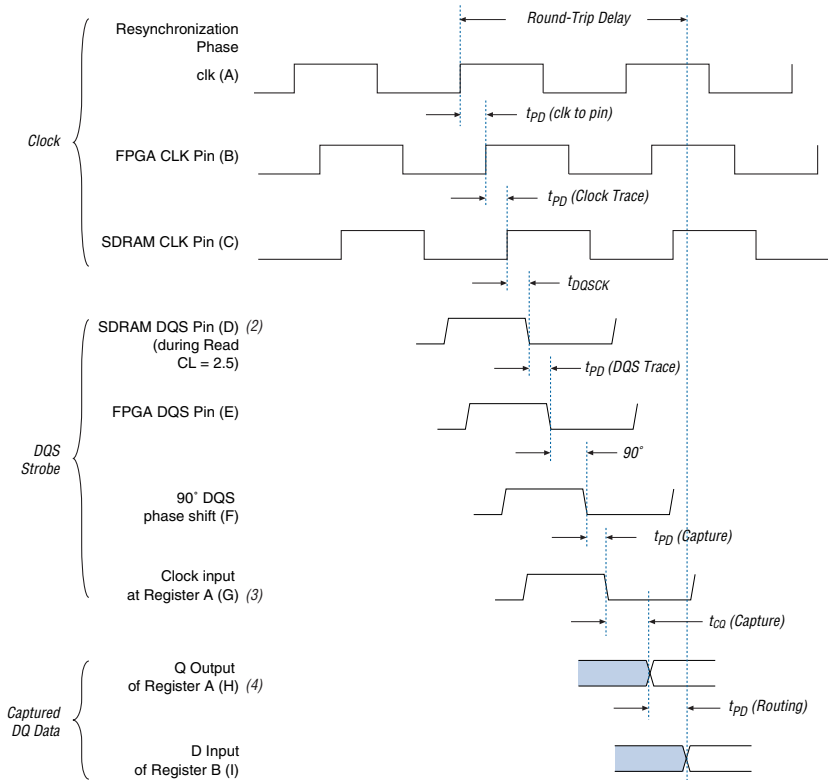
- The DDR SDRAM clock input arrives (a delayed version of `clk`).
- DQS strobe from the DDR SDRAM arrives at the clock input of register A.
- Data arrives at the Q output of register A.
- Data arrives at the D input of register B.

There are therefore three main parts to this path:

- Clock delays—between the FPGA global clock net and the DDR SDRAM clock input.
- DQS strobe delays—between the DDR SDRAM clock input and `dqs`'s arrival at the FPGA capture registers.
- Read data delays—between the output of register A and the input of register B.

Figure 14 shows the timing relationship of the signals for the delays between points (A) to (I) for a CAS latency of 2.5 clock cycles.

**Figure 14. RTD Calculation without a Feedback Clock Note (1)**



**Notes to Figure 14:**

- (1) The letters in parenthesis refer to the letters in Figure 13.
- (2) The DQS strobe edge can be anywhere within  $\pm t_{DQSCK}$  of the DDR SDRAM clock pin edge. For calculating the maximum round trip delay the diagram assumes the strobe occurs  $t_{DQSCK}$  after the clock. For calculating the minimum it has to be assumed it occurs  $t_{DQSCK}$  before.
- (3) The delays in the DQS path from FPGA pin to capture register are matched to those for the DQ path with the exception of the DQS delay chain.
- (4) Although data is initially sampled at a capture register on the positive edge of DQS, it is only on the negative edge that both  $DQ_H$  and  $DQ_L$  are available at the Q outputs of the DDR SDRAM capture logic. At this point they are then single data rate.

To determine the point at which the data can be reliably resynchronized, calculate the minimum and maximum round trip delay. You can then determine what resynchronization logic to use for your system. Remember to take into account PVT variations.



Delay (A) to (B) is the clock-to-out time to generate the clock signals to the DDR SDRAM device.

Delay (B) to (C) is the trace delay for the clock. If there are multiple DIMMs or devices in the system, the one furthest away from the FPGA should be used for the maximum calculation; the closest for the minimum.

Delay (C) to (D) is the relationship between the clock and the DQS strobe timing during reads. This is  $t_{DQSCK}$  in DDR SDRAM specifications, nominally 0, but typically varies by  $\pm 0.75$  ns depending on the DDR SDRAM device-speed grade. The DQS output strobe is only guaranteed to be within  $\pm t_{DQSCK}$  of the clock input. So use  $t_{DQSCK}$  (maximum), typically +0.75 ns, for calculating the maximum round trip delay;  $t_{DQSCK}$  (minimum), typically -0.75 ns, for calculating the minimum delay.

Delay (D) to (E) is the trace delay for dqs, which typically matches the trace delay for the dq signals in the same byte group. To calculate the maximum RTD, use the byte group with the longest trace lengths; for the minimum use the shortest. Similarly, if there are multiple DIMMs or devices in the system, use the one furthest from the FPGA for the maximum calculation and the one closest to the FPGA for the minimum. Trace lengths between different byte groups do not have to be tightly matched, but a difference between the longest and shortest decreases the safe resynchronization window. This increase reduces the window size within which the data can be reliably resynchronized.

PLL jitter and clock duty cycle also affect the RTD. Add each of these delays to the maximum value and subtract from the minimum. PLL jitter and clock duty cycle are not shown in [Figure 13 on page 38](#), but are included in [Table 9 on page 45](#) which shows example RTD calculations.

### *Resynchronization Selections*

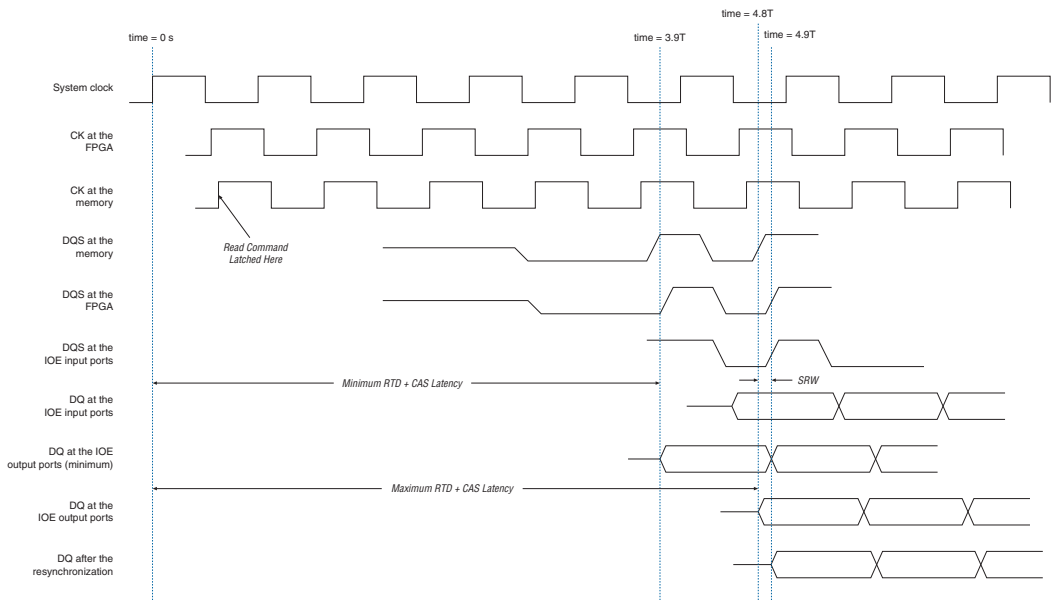
When the DQS signal arrives at the Stratix II device, the dedicated phase-shift circuitry shifts the signal by  $90^\circ$  to capture the DQ signals. The DQ signals are then ready to be synchronized with the system clock. The round trip delay numbers vary depending on the board delay and the device internal delay. Complete a timing analysis to decide whether to use the falling edge or the rising edge of the system clock of the write clock for the synchronization registers. After calculating the maximum and minimum RTD, determine the equivalent number of system clock cycles at your operating frequency to find the point at which the data becomes valid relative to clk. The example maximum delay in [Table 9 on page 45](#) represents 1.8 cycles at 200 MHz; the minimum represents 0.9

cycles. If the CAS latency is included, which is equal to three in [Figure 9 on page 19](#), the example represents a minimum delay of 3.9 cycles and a maximum delay of 4.8 cycles.

The overlap of the minimum and maximum data valid windows defines the data valid window, which comprises the safe resynchronization window and  $t_{SU}$  and  $t_{H}$  of register B.

[Figure 15](#) shows an example timing waveform of the RTD analysis in [Table 9 on page 45](#).

**Figure 15. RTD Diagram without a Feedback Clock Example 1 Note (1)**



**Note to Figure 15:**

(1) T refers to the clock period of the system, which is 5 ns in this example.

The RTD helps you determine the safe resynchronization window (SRW) and how you need to resynchronize the data. Since the shifted DQS signal can go into the LEs, you can use the shifted DQS signal for resynchronization within the LE. This may mean that the DQS clock domain to system clock domain transfer will happen between two LE registers. The timing analysis in this section assumes that the clock domain transfer happens from the IOE register to the LE register.

Assume that time 0 is the time of the clock rising edge. Once the DDR SDRAM device receives this rising edge from the Stratix II device, the read command is clocked into the SDRAM upon receiving this positive edge, and you can calculate the SRW valid time is as follows:

$$\text{Minimum SRW valid time} = \text{maximum RTD} + \text{CAS latency} \times \text{clock period} + \mu t_{\text{SU}}$$

$$\text{Maximum SRW valid time} = \text{minimum RTD} + (\text{CAS latency} + 1) \times \text{clock period} - \mu t_{\text{H}}$$

From the example in [Table 9 on page 45](#), the minimum SRW valid time is 4.8 cycles and the maximum SRW valid time is 4.9 cycles (ignoring  $\mu t_{\text{SU}}$  and  $\mu t_{\text{H}}$ ).

The size of the SRW in the example is then 0.1 cycles, calculated by the following equation:

$$\text{SRW size} = \text{maximum SRW valid time} - \text{minimum SRW valid time}$$

The SRW size should be large enough to accommodate the worst-case clock skew between two PLL output clocks (150 ps).

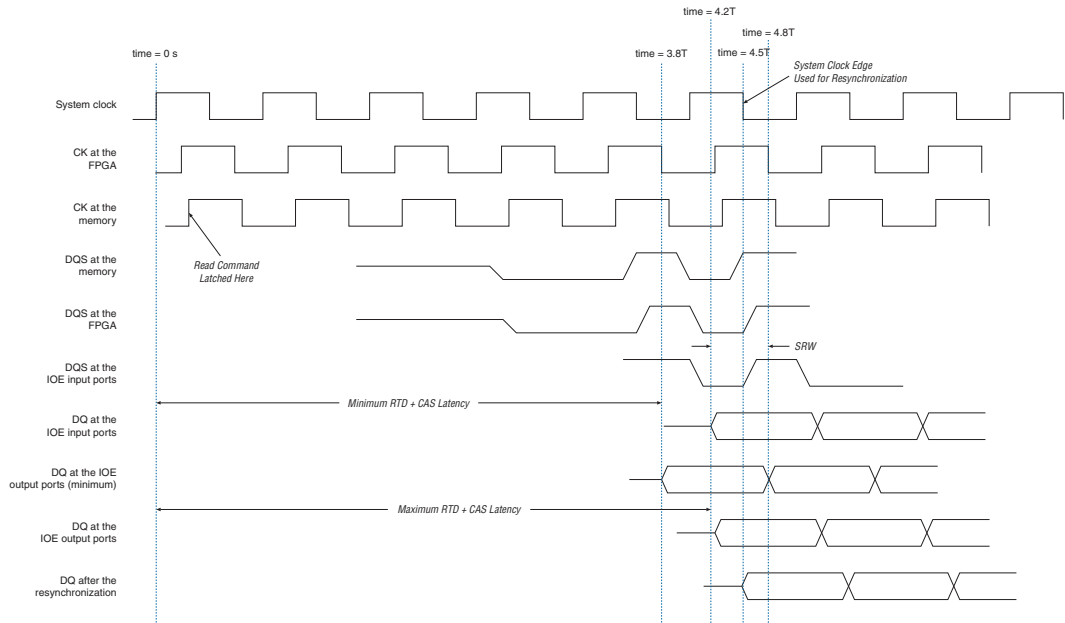
Next, determine how many half clock cycles elapse from time 0 to the minimum SRW valid time (`numcycle`) by calculating the ceiling function of the minimum SRW valid time divided by half a clock cycle. To find out whether the SRW falls within a clock edge, multiply `numcycle` by half a clock cycle. If the result is less than the maximum SRW valid time, then a system clock edge falls within the SRW. Otherwise you need an extra PLL output for your resynchronization clock.

The example in [Table 9 on page 45](#) shows that `numcycle` is equal to 10 and that the SRW does not fall within a system clock edge.

If you do not need a resynchronization clock and `numcycle` is an even number, the active system clock edge for resynchronization is the positive edge. If `numcycle` is odd, the resynchronization system clock edge is the negative edge, and you must determine the resynchronization phase selection.

Figure 16 illustrates an example where the SRW is within a system clock edge. In that example, `numcycle` is equal to 9 (time = 4.5T) and the negative edge of the system clock is used for the resynchronization clock.

Figure 16. RTD Diagram without a Feedback Clock Example 2



Note to Figure 16:

(1) T refers to the clock period of the system, which is 5 ns in this example.

If there is no clock edge available within the safe resynchronization window, you need an extra resynchronization clock, you can shift the system clock from either edge. If `numcycle` is even in this case, the closest system clock edge to the SRW is negative and if `numcycle` is odd, the closest clock edge is positive.

You can calculate the needed phase shift for the resynchronization clock from the following equations:

$$\text{Minimum phase shift} = \text{minimum SRW valid time} + \text{PLL clock skew} \\ (150 \text{ ps}) - (\text{numcycle} - 1) \times t_{\text{CK}}/2$$

$$\text{Maximum phase shift} = \text{minimum SRW valid time} - \text{PLL clock skew} \\ (150 \text{ ps}) - (\text{numcycle} - 1) \times t_{\text{CK}}/2$$

The phase-shift calculation example in [Table 9 on page 45](#) shows that the minimum phase shift is 1.61 ns and the maximum phase shift is 1.52 ps. This is because the SRW is less than 300 ps. You can still choose the median (1.565 ns) for the phase shift of the resynchronization clock.

You then need to convert the results to the equivalent degree phase shifts. If the closest clock edge to the SRW is negative, add or subtract  $180^\circ$  after the conversion to shift the clock from the positive edge. For example, in [Table 9](#), the phase-shift range is between 1.52 to 1.61 ns based on the negative edge clock. The median of this number is 1.565 ns which equates to  $\sim 113^\circ$  (from 200-MHz clock). If you want this clock to be shifted from the positive edge of system clock, you can either use  $293^\circ$  ( $113^\circ + 180^\circ$ ) or  $-67^\circ$  ( $113^\circ - 180^\circ$ ).

**Table 9. Example RTD Calculation** *Note (1)*

Delay	Numbers in Figure 13 & Figure 14	Example Minimum Values (ns)	Example Maximum Values (ns)	Comments
$t_{PD}$ (clk to pin)	(A) to (B)	2.00	3.00	Equal to $t_{CQ}$ (DQS write)
$t_{PD}$ (clock trace)	(B) to (C)	0.33	0.50	2 to 3 inches at 166 ps per inch (2)
$t_{DQSCK}$	(C) to (D)	-0.60	+ 0.60	See DDR SDRAM specifications
$t_{PD}$ (dqs trace)	(D) to (E)	0.33	0.50	2 to 3 inches @ 166 ps per inch (2)
90° phase-shift	(E) to (F)	1.133	1.333	Includes Stratix II DLL jitter and phase-shift error
$t_{PD}$ (capture)	(F) to (G)	0.50	1.00	—
$t_{CQ}$ (capture)	(G) to (H)	0	0.16	—
$t_{PD}$ (routing)	(H) to (I)	1.00	1.50	—
PLL jitter	—	-0.10	+ 0.10	PLL jitter specification
Clock duty cycle	—	-0.25	+ 0.25	45-55% duty at 200 MHz
Round-trip total	(A) to (I)	4.343	8.943	—

**Notes to Table 9:**

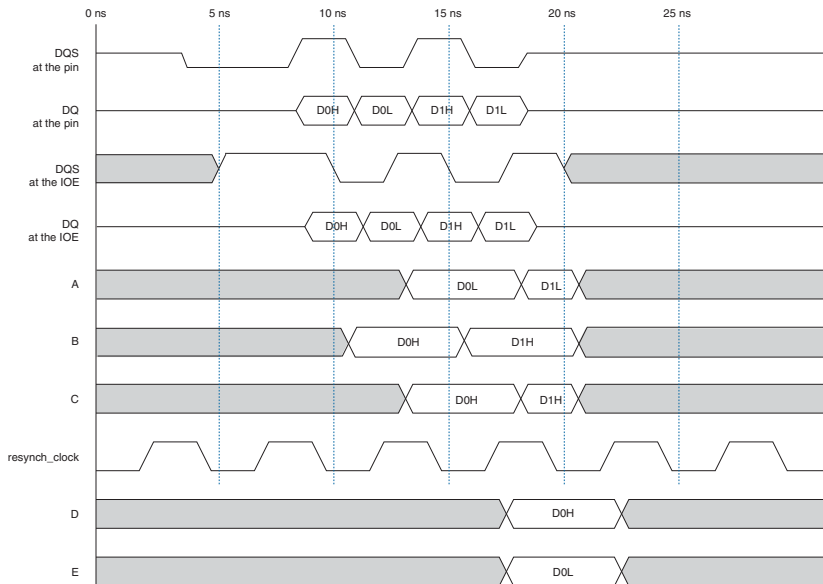
- (1) Numbers quoted here are not taken from a specific system or a specific device.
- (2) To know the exact delay for your system, perform a time domain reflectometry (TDR) analysis on your system.

## DQS Postamble

The DDR SDRAM DQ and DQS pins use the SSTL-2 class II I/O standard. When neither the Stratix II nor DDR SDRAM device drives the DQ and DQS pins, the signals go to a high-impedance state. Since a pull-up resistor terminates both DQ and DQS to  $V_{TT}$  (1.25 V), the effective voltage on the high-impedance line is 1.25 V. According to the JEDEC JESD 8-9 specification for the SSTL-2 class II I/O standard, this is an indeterminate logic level and the input buffer can interpret this as either a logic high or logic low. If there is any noise on the DQS line, the input buffer may interpret that noise as actual strobe edges. Therefore, when the DQS signal goes to tri-state after a read postamble, you should disable the clock to the input registers so that erroneous data does not get latched in and all the data from the memory is resynchronized properly.

Figure 17 shows a read operation example when the DQS postamble could be a problem. Figure 9 shows definitions of A, B, C, D and E. Waveform A shows the output of the active high IOE input register. Waveform B shows the active low register output of the Stratix II IOE input register. The active low register output goes into the latch whose output is illustrated in Waveform C. Waveforms D and E illustrate the output signals after the resynchronization registers.

**Figure 17. Read Example with a DQS Postamble Issue**



The first falling edge of the DQS at the IOE register occurs at 10 ns. At this point, data D0H is clocked in by the active low register (waveform B). At 12.5 ns, data D0L is sampled in by the active high register (waveform A) and data D0H passes through the latch (waveform C). In this example, the positive edge of the `resynch_clock` occurs at 16.5 ns, where both D0H and D0L are sampled by the logic element's (LE's) resynchronization registers. Similarly, data D1H is clocked in by the active low register at 15 ns, while data D1L is clocked in by the active high register and data D1H passes through the latch at 17.5 ns. At 20 ns, assume that noise on the DQS line causes a valid clock edge at the IOE registers such that it changes the value of waveforms A, B, and C. The next rising edge of the `resynch_clock` signal does not occur until 21.5 ns, but data D1L and D1H are not valid anymore at the output of the latch and the active-high input register, so the resynchronization registers do not sample D1L and D1H and may sample the wrong data instead.

Stratix II devices have circuitry to prevent false edge trigger at the end of the DQS postamble. Each Stratix II DQS logic block is connected to a postamble circuitry that consists of an AND gate (see Figure 18). This register is clocked by the shifted DQS signal. Its `SCLRDN` port is connected to GND. The controller needs to include extra logic to tell the reset signal to release the preset signal on the falling DQS edge at the start of the postamble. This disables any glitches that happen right after the postamble.

**Figure 18. Stratix II DQS Postamble Circuitry Connection**

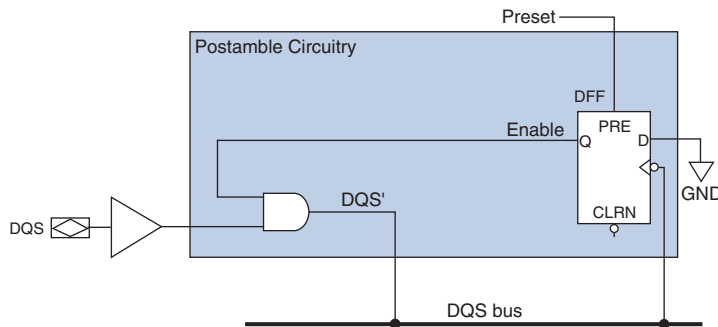
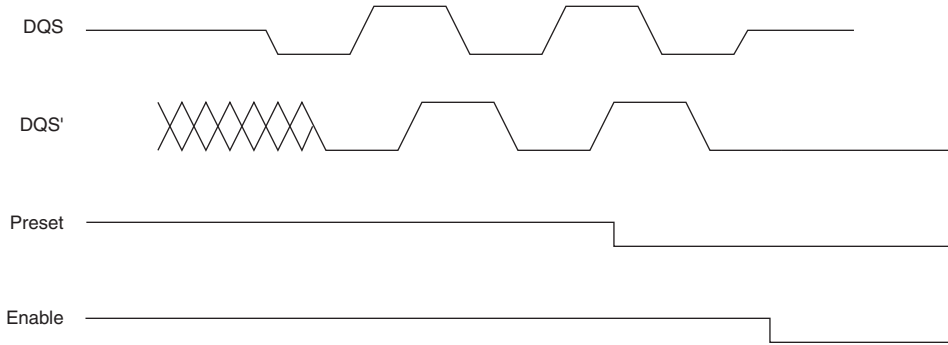
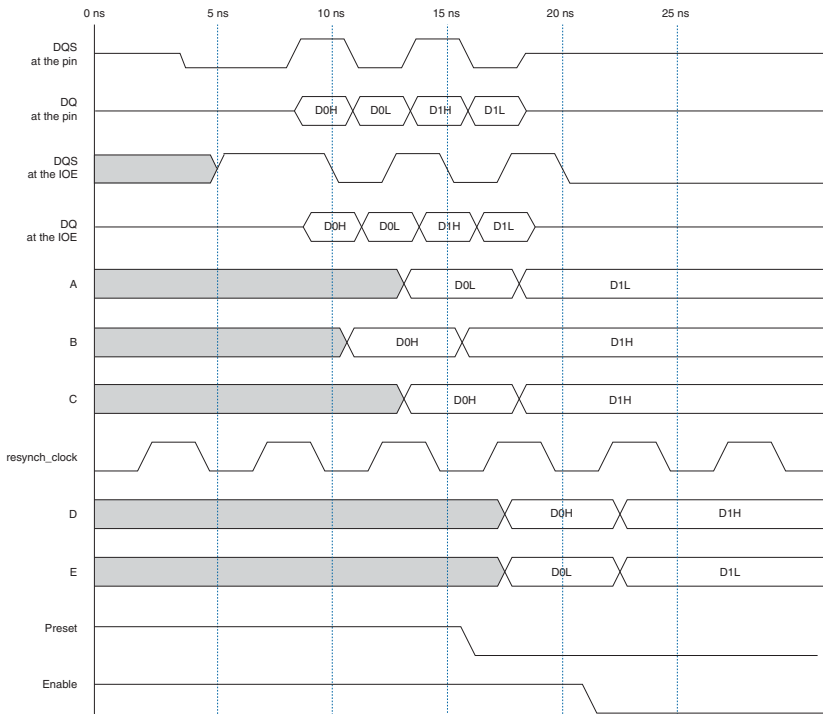


Figure 19 shows the timing waveform for Figure 18. Figure 20 shows the read timing waveform when the Stratix II DQS postamble circuitry is used.

**Figure 19. Stratix II DQS Postamble Circuitry Control Timing Waveform**



**Figure 20. Stratix II DQS Postamble Circuitry Read Timing Waveform**





The Altera DDR SDRAM controller MegaCore function calculates the resynchronization cycle and phase but allows you to override these settings.



For more information on Stratix II DQS postamble circuitry, refer to *DDR MegaFunction User Guide*.

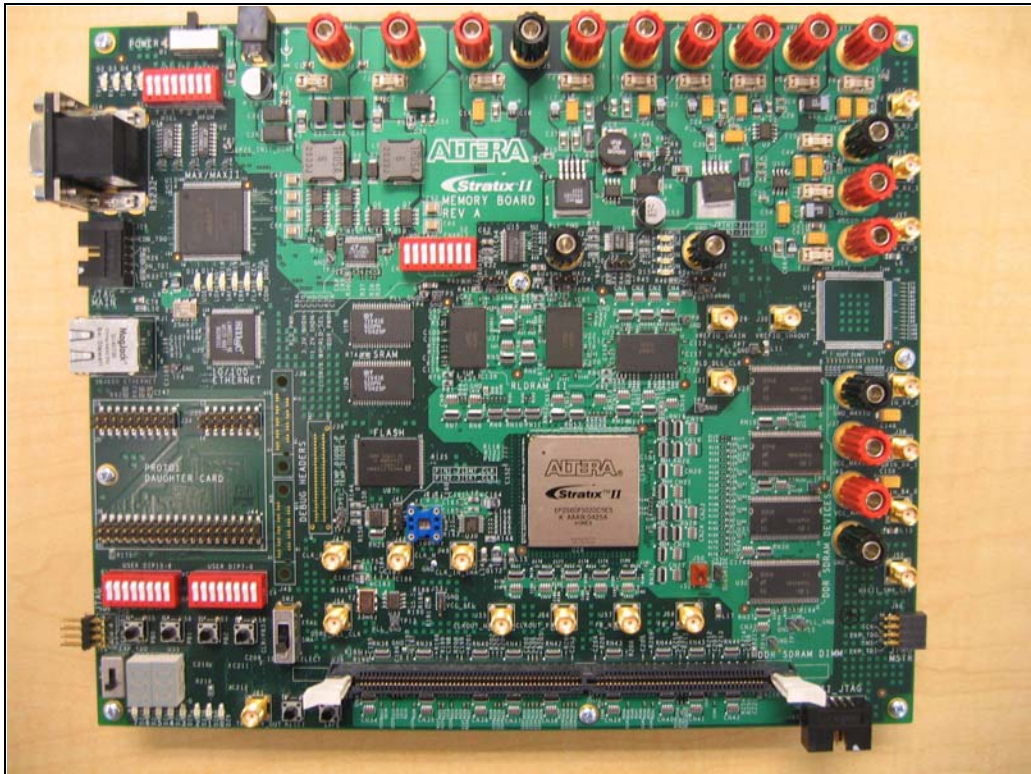
## Stratix-Series Memory Board I

Altera produces the Stratix II memory board I and Stratix memory board I to demonstrate DDR SDRAM and RLD RAM II interfaces with the Stratix-series device family. The Stratix II memory board I includes a Stratix II EP2S60F1020C4 device interfacing with the following external memory devices:

- Four DDR SDRAM  $\times 16$  devices connected to the Stratix II side I/O banks of banks 1 and 2. The boards will use one of the following third-party memory devices: Micron MT46V16M16TG-5B, Infineon HYB25D25616OBT-5A, or Samsung K4H561638F-TCCC.
- One DDR SDRAM module connected to the Stratix II I/O banks 7 and 8. The boards will use one of the following third-party memory devices: Micron MT9VDDT3272AG-40B, Infineon HYS72D32300GU-5-B, or Samsung M381L3223ETM-CCC.
- One RLD RAM-II SIO  $\times 18$  device connected to the Stratix II I/O bank 3. The boards will use the Micron MT49H16M18CFM-2.5 third-party memory device.
- Two RLD RAM-II CIO  $\times 18$  devices connected to the Stratix II I/O bank 4 that support 400 MHz double data rate (DDR). The boards will use one of the following third-party memory devices: Micron MT49H16M18FM-2.5 or Infineon HYB18RL28818AC-2.5.

Figure 21 shows the Stratix II memory board I.

**Figure 21. Stratix II Memory Board I**



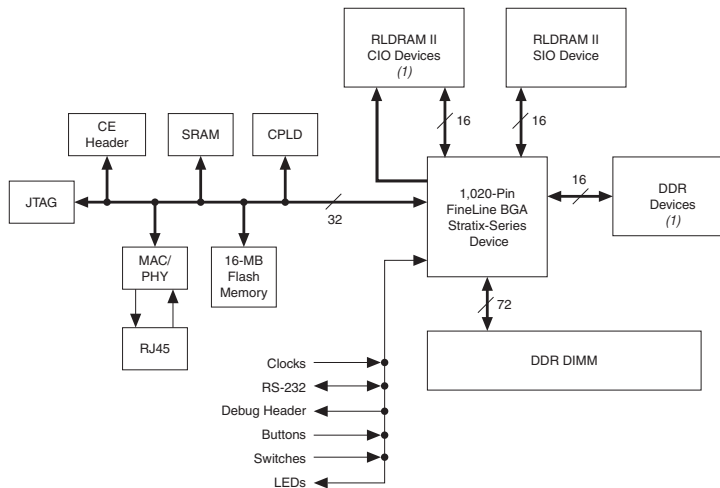
The Stratix-series memory board I is powered by a single DC input with on-board regulators generating the other required lower voltages. In addition to the on-board regulators, fuse-isolated banana jacks will be provided for all unique voltages for characterization purposes. The incoming DC voltage will be regulated down to 3.3-V and 1.2-V using a dual switching power supply in order to efficiently support the fairly large DC drop from the input to the output. All other board voltages are generated from this 3.3-V rail. There are fuse sockets to isolate planes from regulators to allow bench supplies to power these sections using banana jacks.

The following regulators are available on the Stratix II memory board:

- Linear Technology LTC3728—Dual-output regulator for generating the Stratix II device's  $V_{CCINT}$  and 3.3-V outputs.
- Micrel Semiconductor MIC29502BU—High-current low-dropout regulator for generating the power for the memory devices and the Stratix PLL.
- National Semiconductor LP2996MR—DDR SDRAM and RLDRAM II termination regulator for generating the termination voltage ( $V_{TT}$ ) and reference voltage ( $V_{REF}$ ).
- Micrel Semiconductor MIC94300—Low-voltage low-dropout regulator for generating the Stratix II PLL power.
- Linear Technology LTC1872B—Step-up DC/DC controller for generating power for the fan circuit.

Figure 22 shows the Stratix-series memory board I block diagram.

**Figure 22. Stratix-Series Memory Board I Block Diagram**



**Note to Figure 22:**

(1) The Stratix-series memory board I has multiple RLDRAM II CIO and DDR SDRAM devices.

## Conclusion

DDR SDRAM is widely used in FPGA designs and is the most popular architecture of DRAM technology. Stratix II devices offer a proven, flexible, and high-performance interface to DDR SDRAM with consistent timing margins to meet your design needs. A compelling DRAM solution for a wide range of end applications, DDR SDRAM interfaces quickly and easily with Altera's Stratix II devices.

## Referenced Documents

This chapter references the following documents:

- *JEDEC Standard Publication JESD79C, DDR SDRAM Specification*, JEDEC Solid State Technology Association.
- *MT9VDDT3272AG-40B, 184-Pin DDR SDRAM DIMMs Data Sheet*, Micron Technology, Inc.
- [www.jedec.org](http://www.jedec.org).
- *External Memory Interfaces in Stratix II and Stratix II GX Devices* chapter in volume 2 of the *Stratix II Device Handbook*.
- *PLLs in Stratix II and Stratix II GX Devices* chapter in volume 2 of the *Stratix II Handbook*.
- *DDR MegaFunction User Guide*.
- *TB 091: External Memory Interface Options for Stratix II Devices*.
- *ALTMEMPHY Megafunction User Guide*.
- *DDR and DDR2 SDRAM High Performance Controller MegaCore Function User Guide*.
- *DDR and DDR2 SDRAM Controller Compiler User Guide*

## Document Revision History

Table 10 shows the revision history for this document.

<b>Date &amp; Document Version</b>	<b>Changes Made</b>	<b>Summary of Changes</b>
September 2008 v3.2	<ul style="list-style-type: none"> <li>● Updated "Interface Description", "Interface Architecture", "Setup &amp; Hold Margins Calculations", "DQS Postamble", and "Referenced Documents" sections.</li> <li>● Updated Table 5, Table 6, and Table 8.</li> <li>● Updated Figure 18, Figure 19, and Figure 20.</li> </ul>	—
May 2006 v3.1	<ul style="list-style-type: none"> <li>● Updated Figure 18 and Figure 20.</li> </ul>	—
v3.0	<ul style="list-style-type: none"> <li>● Updated "Interface Description", "Interface Signals" "Interface Timing Analysis", "Methodology Overview", "Read Timing Margins for PLL-Based Implementation", and "Round Trip Delay Calculation" sections.</li> <li>● Updated Table 3, Table 5, Table 5, Table 6, and Table 8.</li> <li>● Updated Figure 4, Figure 6, Figure 10, and Figure 11.</li> </ul>	—