



XAPP454 (v2.1) January 20, 2009

DDR2 SDRAM Interface for Spartan-3 Generation FPGAs

Author: Samson Ng

Summary

This application note describes a DDR2 SDRAM interface implementation in a Spartan[®]-3 generation FPGA, interfacing with a Micron DDR2 SDRAM device. This document provides a brief overview of the DDR2 SDRAM device features, followed by a detailed explanation of the DDR2 SDRAM interface implementation.

DDR2 SDRAM Device Overview

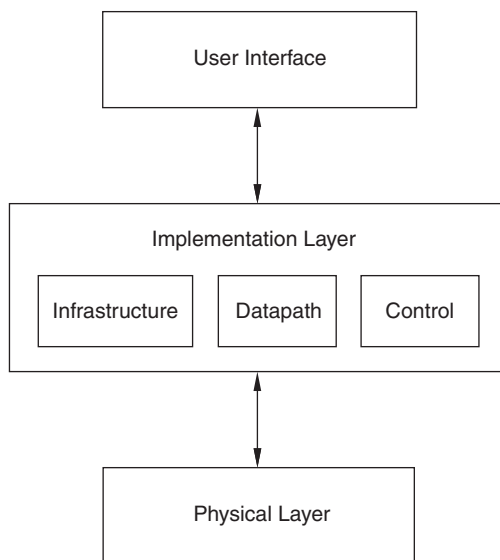
The DDR2 SDRAM interface is source-synchronous, supporting double-data rate transfers like DDR SDRAM, and uses the SSTL 1.8V I/O standard, providing significant system power savings. DDR2 SDRAM is only available in FBGA packages. These packages have greatly reduced parasitics as compared to TSOP packages for DDR SDRAM. The DDR2 SDRAM bus is clocked at twice the speed of DDR SDRAM and uses 4n prefetch instead of 2n prefetch.

DDR2 SDRAM devices achieve high-speed operation by transferring data on both the rising and falling edges of the clock signal. The memory operates using a differential clock provided by the controller. Commands are registered at every positive edge of the clock. A bidirectional data strobe (DQS) is transmitted along with the data for use in data capture at the receiver. During reads, DQS is transmitted by the DDR2 SDRAM device and is edge aligned with data. During writes, DQS is transmitted by the controller and is center aligned with data.

Read and write accesses to the DDR2 SDRAM device are burst oriented. Accesses begin with the registration of an active command and are then followed by a read or write command. The address bits registered coincident with the active command are used to select the bank and row to be accessed. The address bits registered with the read or write command are used to select the bank and starting column location for the burst access.

Interface Model

The DDR2 SDRAM interface is layered to simplify the design and make it modular. [Figure 1](#) shows the layered memory interface. The three layers consist of an application layer, an implementation layer, and a physical layer.

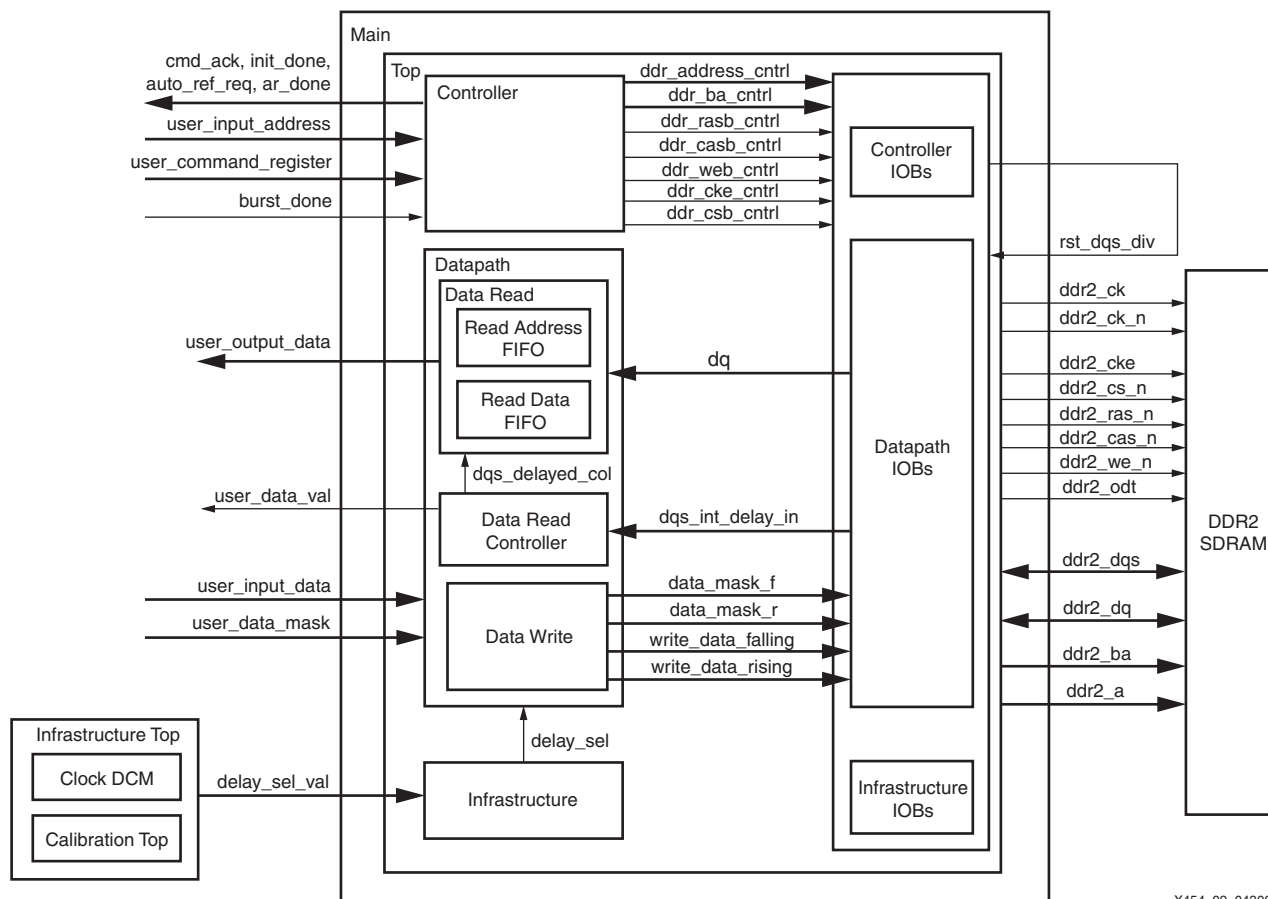


X454_01_032008

Figure 1: Interface Layering Model

DDR2 SDRAM Controller Modules

Figure 2 is a block diagram of the DDR2 SDRAM interface for the Spartan-3 generation FPGA. All four blocks shown in this figure are subblocks of the ddr2_top module. The function of each block is explained in the following sections.



X454_02_043008

Figure 2: DDR2 SDRAM Interface Modules

Controller

The controller supports burst lengths of four and eight, and a CAS latency of three. The controller initializes the EMR(2) and EMR(3) registers during the Load Mode command and also generates differential data strobes.

The controller accepts user commands, decodes these user commands, and generates read, write, and refresh commands to the DDR2 SDRAM. The controller also generates signals for other modules.

Datapath

The datapath module is responsible for transmitting data to and receiving data from the memories. Major functions include:

- Writing data to the memory
- Reading data from the memory
- Transferring the read data from the memory clock domain to the FPGA clock domain

The write data and strobe are clocked out of the FPGA. The strobe is center aligned with respect to the data during writes. For DDR2 SDRAM memories, the strobe is non-free running. To meet these requirements, the write data is clocked out using a clock that is shifted 90° (CLK90) and 270° (CLK270) from the primary clock going to the memory. CLK270 is generated locally from CLK90 through clock inversion. The data strobes are generated out of primary clocks going to the memory.

Memory read data is edge aligned with a source-synchronous clock. The data is received using the non-free running strobe and transferred to the FPGA clock domain. The input side of the data uses resources similar to the input side of the strobe. This ensures matched delays on data and strobe signals until the strobe is delayed in the strobe delay circuit.

Read Data Capture

During a read transaction, the DDR2 SDRAM device sends the DQS strobe and associated data to the FPGA. DQS is edge aligned with the data. Capturing the data is a challenging task in source-synchronous interfaces that run at high frequencies. The data changes at every edge of DQS and the strobe is not free running. Read data from the memory device is captured directly into the FPGA logic using a delayed DQS. The DQS delay mechanism is explained in [“Delay Circuits,” page 8](#).

Instead of registering the data in the input/output blocks (IOBs), a look-up table (LUT)-based dual-port distributed RAM is used for data capture. This is a simpler method of data capture with no need of second stage recapturing to the system clock domain. The LUT RAM is configured as a pair of FIFOs, and each data bit is input into both FIFOs, as shown in [Figure 3](#). These 16-entry-deep FIFOs are asynchronous and have independent read and write ports.

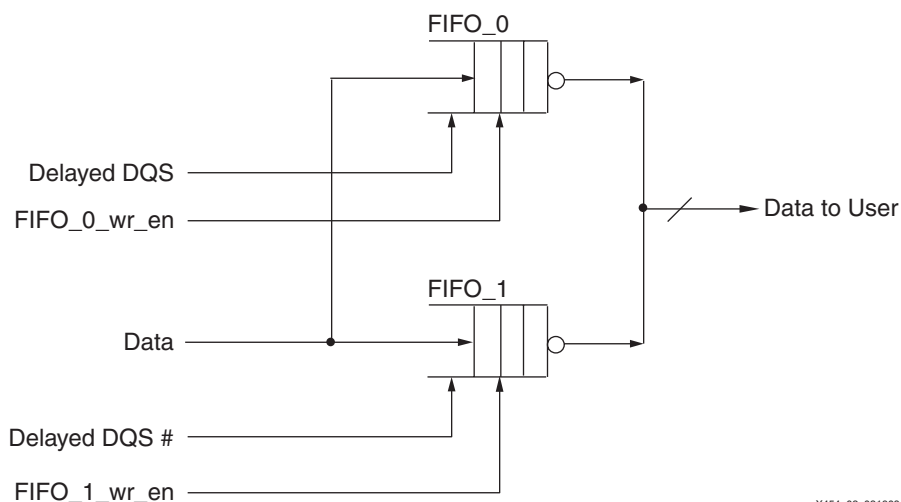


Figure 3: FIFO Block Diagram

Read Data Clocking

Read data from the memory is written into FIFO_0 on the rising edge of the delayed DQS, and into FIFO_1 on the falling edge of the delayed DQS. Data can be read out of both FIFO_0 and FIFO_1 simultaneously.

The FIFO write pointer is clocked using the delayed DQS. FIFO read pointers are clocked in the FPGA internal clock domain. The FIFOs are written when the FIFO write enable signal is asserted. Generation of the FIFO write enable signal is explained in detail in [“Write Enable Generation,”](#) page 5.

The FIFO write enable signal is generated from a signal named `rst_dqs_div`, which is asserted at any time during the preamble for the DQS. This signal is deasserted at any time during the last two DQS phases. [Figure 4](#) illustrates the idea behind `rst_dqs_div`.

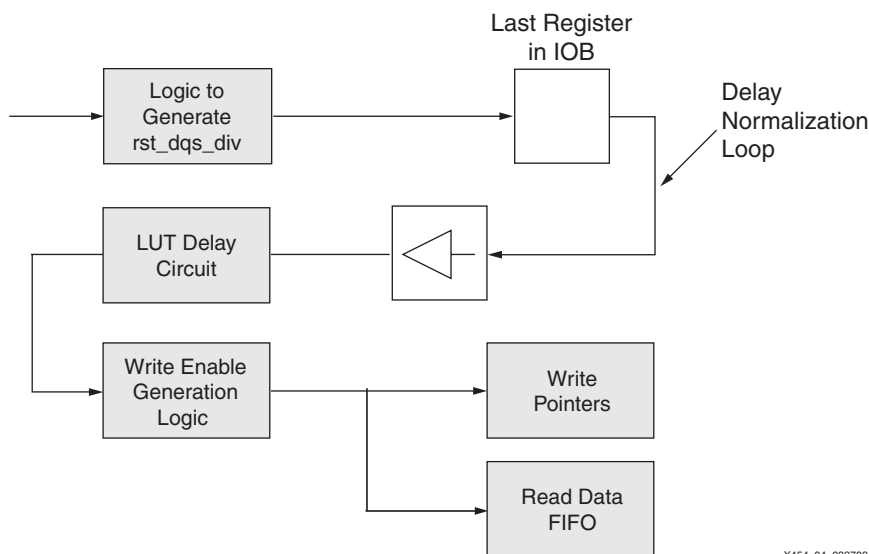


Figure 4: Circuit Illustrating `rst_dqs_div` Signal

The `rst_dqs_div` signal is driven to an IOB as an output and is then taken as an input through the input buffer. This technique normalizes the IOB and trace delays between `rst_dqs_div` and the DQS clock signals. The `rst_dqs_div` from the input pad of the FPGA uses identical routing

resources as the DQS before it enters the LUT delay circuit. The trace delay of the loop should be the sum of the trace delays of the clock forwarded to the memory and the DQS.

The LUT delay circuit shown in Figure 4 is identical to the LUT delay circuit used for the delayed DQS. This ensures that `rst_dqs_div` and the delayed DQS signals take identical paths and have similar delays before being sent to the FIFO write pointers and FIFO write enable inputs.

Write Enable Generation

Figure 5 shows the circuit for write enable generation. The FIFO_0 write enable signal is the logical OR of `rst_dqs_div` and the registered version of the `rst_dqs_div` output. FIFO_1 is enabled after the first positive transition of the delayed DQS signal. This logic eliminates false latching of data into the FIFOs. It also eliminates false incrementing of the write pointers during the preamble period of the delayed DQS 3-state-to-Low transition and the postamble period of the delayed DQS High-to-3-state transition.

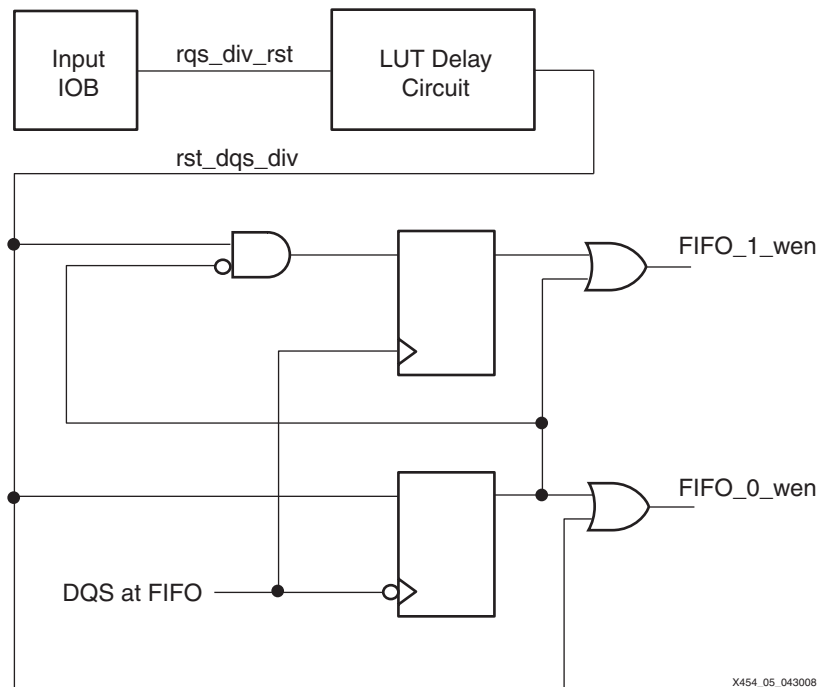


Figure 5: FIFO Write Enable Generation

The registered output enables the FIFOs and FIFO write pointers during the postamble period when `rst_dqs_div` is deasserted. The registered `rst_dqs_div` flag is cleared on the last trailing edge of the delayed DQS, and the FIFOs and FIFO write pointers are disabled.

Figure 6 shows the timing diagram for `dqs_div_rst`, `rst_dqs_div`, and the FIFO write enable signals. The total delay of `rst_dqs_div` and the FIFO write enable signals should not exceed one memory clock cycle. The Memory Interface Generator (MIG) tool generates the necessary constraints for `dqs_div_rst`, `rst_dqs_div`, and the FIFO write enable signals in the user constraints file (UCF) with the MAXDELAY constraint to ensure that the write enable signals meet the setup time.

Data is latched into the FIFOs when the delayed DQS toggles. The FIFO_0 and FIFO_1 write pointers are enabled when `rst_dqs_div` is asserted. Data is latched into FIFO_0 on the rising edge of the delayed DQS, and the FIFO_0 write pointer increments on the same edge. Data is latched into FIFO_1 on the falling edge of the delayed DQS, and the FIFO_1 write pointer increments on the same edge.

The timing diagrams of Figure 6 show how data is captured into FIFO_0, FIFO_1, and FIFO Write Enable.

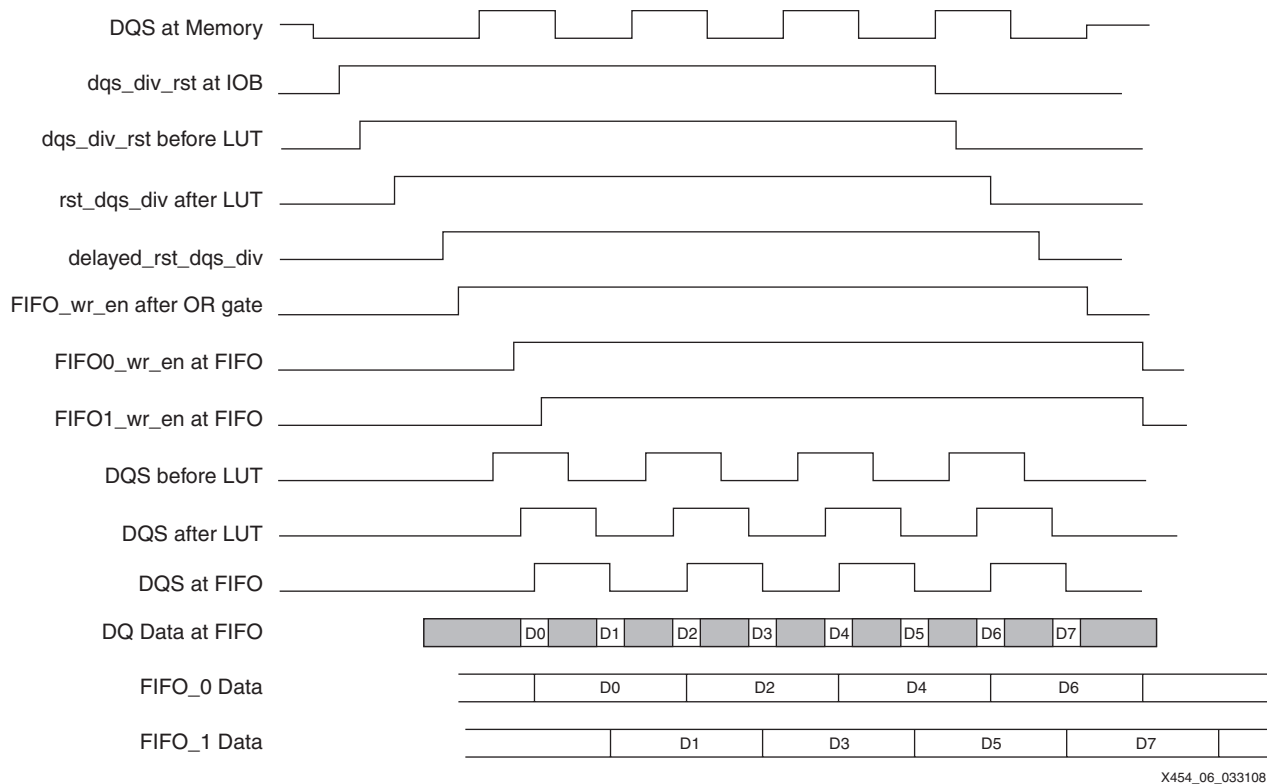


Figure 6: Timing Diagrams of FIFO_0, FIFO_1, and FIFO Write Enable

Infrastructure

The top-level Infrastructure module generates the FPGA clocks and reset signals. A Digital Clock Manager (DCM) is used to generate CLK0 and CLK90. A delay calibration circuit is also implemented in this module.

Delay Calibration Circuit

Several factors can affect the centering of DQS in the DQ data valid window. Due to process, voltage, and temperature variations, the LUT delay can vary from 250 ps to 625 ps in the Spartan-3 family. The process variation is calibrated by the delay calibration circuit to select the correct number of taps in the DQS tap delay circuit at reset and power-up. As temperature or voltage drift up or down, the delay calibration circuit continuously calibrates to position the DQS in the center of the DQ data valid window.

The LUT delays in the delay circuit are measured using the tap delay circuit shown in [Figure 7](#). Each inverter in the tap delay circuit is created using a LUT. Similarly, each multiplexer in [Figure 9](#) is built using a LUT. This calibration circuit does not require additional DCMs or global buffers (BUFGs).

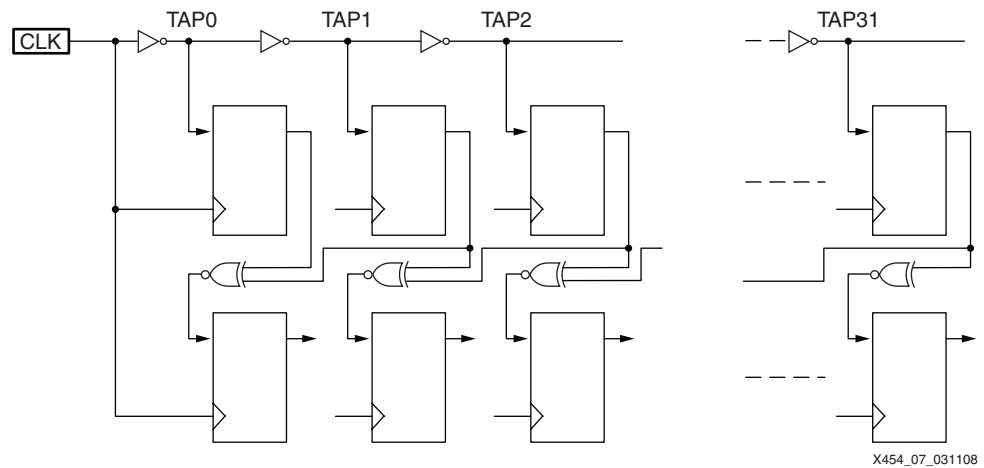


Figure 7: Tap Delay Circuit Built Using LUTs

[Figure 8](#) shows the waveforms generated by the tap delay circuit.

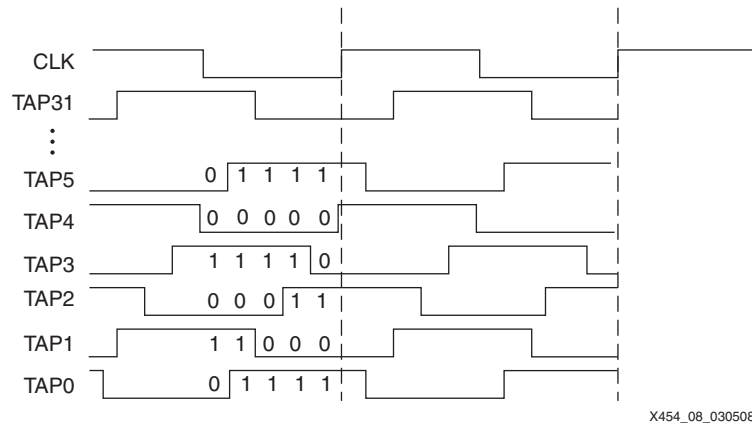


Figure 8: Tap Delay Waveforms

Each of the tap delays in [Figure 8](#) has its input inverted. The nominal LUT delay in the XC3S700A-5FG484 device is 620 ps. For a 166 MHz design, there are five or six LUTs in a clock phase. The edge pattern (10101101011010 or 0101001010010) is generated from the tap circuit. The patterns are registered at the rising clock boundary through a chain of flip-flops. The tap delay circuit creates two rising edges that are then counted by a phase counter using the same clock to count how many taps are in a clock phase. The calibrated tap values are then used to determine how many MUXs are needed to position the DQS strobe line.

The number of taps required for a clock phase is determined by the specific FPGA used. If a faster FPGA is chosen, the number of LUT delays in the strobe path is increased by the delay calibration logic to ensure that the strobe is still within the valid data window. If the Spartan-3 generation FPGA becomes faster due to process variations or any other reason, the number of tap delays for the same clock phase is automatically increased by the delay calibration logic.

The delay calibration circuit selects the number of delay elements used to delay the strobe lines with respect to the read data. The delay calibration circuit calculates the delay of a circuit that is identical in all respects to the strobe delay circuit. All aspects of the delay are considered for

calibration, including all the component and route delays. The calibration circuit selects the number of delay elements for any given time. After the calibration is done, the calibration circuit adjusts the select lines for the delay circuit.

Delay Circuits

The total strobe delay must fall within a data valid window. Therefore, the strobe is delayed using internal delay elements. The delay elements consist of LUTs and other resources in the FPGA fabric. Selecting the elements and routing resources used within those elements defines the delay values precisely.

Figure 9 illustrates the manner in which LUTs are configured to create delay elements. The nominal delay for each LUT is 620 ps in a Spartan-3A device of -5 speed grade.

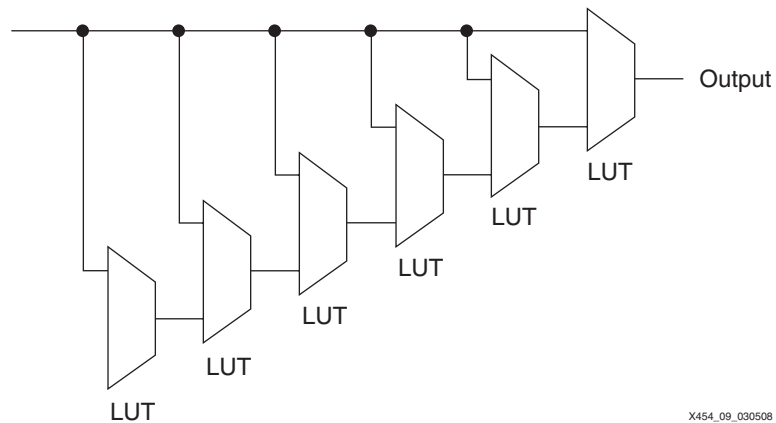


Figure 9: Building Delay Elements Using LUTs in a Xilinx FPGA

Delay Circuit Timing Analysis

All timing analysis is done using a -5 speed grade XC3S700A-5FG484 device at 166 MHz. Table 1 illustrates the timing analysis for the FIFO-based data capture scheme described in “Read Data Capture,” page 3. On the left and right banks, the local clock net delay is relatively small. A single vertical Full Hex (VFULLHEX) line is used for local clock distribution. Because of this, it is possible to easily control the overall strobe delay to fall within the calculated window. As described in “Delay Circuits,” the overall strobe delay is controlled with the delay circuit. The performance of designs in which the DDR2 is implemented on the left or right side of the FPGA is 166 MHz or 333 Mb/s on a -5 speed grade device.

Table 1: Read Data Timing

Parameter	Value (ps)	Leading Edge Uncertainties (ps)	Trailing Edge Uncertainties (ps)	Meaning
T _{CLOCK}	6024			Clock period (1/f).
T _{CLOCK_PHASE}	3012			Clock phase is half of T _{CLOCK} .
T _{CLOCK_DUTY_CYCLE_DIST}	410	0	0	Duty cycle distortion of clock to memory.
T _{DATA_PERIOD}	2602	0	0	Total data period, which is equal to T _{CLOCK_PHASE} – T _{CLOCK_DUTY_CYCLE_DIST} .
T _{DQSQ}	300	300		Strobe to data distortion from <i>Micron MT47H16M16BG-37E DDR2 SDRAM Data Sheet [Ref 1]</i> .

Table 1: Read Data Timing (Cont'd)

Parameter	Value (ps)	Leading Edge Uncertainties (ps)	Trailing Edge Uncertainties (ps)	Meaning
T _{QHS}	400		400	Hold skew factor for DQ from <i>Micron MT47H16M16BG-37E DDR2 SDRAM Data Sheet</i> .
T _{DS}	-70	-70	0	Setup time for RAM16X1D from <i>Spartan-3A FPGA Family: Data Sheet [Ref 2]</i> for -5 speed grade.
T _{DH}	130	0	130	Hold time of FIFO. This value is taken from <i>Spartan-3A FPGA Family: Data Sheet [Ref 2]</i> for -5 speed grade.
T _{PACKAGE_SKEW}	60	60	60	Worst-case package skew.
T _{LOCAL_CLOCK_SKEW}	65	65	65	Worst-case local clock line skew.
T _{PCB_LAYOUT_SKEW}	50	50	50	Skew between data and strobes on the board.
Total Uncertainties		405	705	Total uncertainties for leading edge and trailing edge uncertainties.
Data Valid Window Boundaries		405	1897	
Data Valid Window	1492			Data valid window is T _{DATA_PERIOD} - (Leading edge uncertainties + Trailing edge uncertainties).

LUT Selected Banks

Table 2 shows the LUT selection table when left and right banks are used for data and strobe pins in an XC3S700A-5FG484 device. At normal temperature, two multiplexers of the delay circuit are selected by the calibration circuit to delay the strobe.

The number of multiplexers in the DQS path depends only on the number of LUTs in a clock phase. This mapping is constant across frequencies up to 166 MHz. Thus, the entire design is frequency-independent.

Table 2: LUT Selection for Left and Right Implementations

Parameter (ps)	Nominal	90%	80%	70%	60%	50%	40%
DQ Data Delay	384	3456	307	269	230	192	154
Local Clock Route Delay	397	357	286	200	120	60	24
DQS Delay IOB to LUT	450	405	324	227	136	68	27
LUT Delay	620	558	496	434	372	310	248
Number of LUTs in a Clock Phase	5	5	6	7	8	10	13
Number of LUTs to Delay DQS	2	2	3	4	3	4	4
Total DQS Delay	1467	1320	1670	1376.9	1624	1354	1083
Total Extra DQS Delay	1083	1424.7	1362	1192	1394	1162	929
Data Valid Window (Lower Bound)	405	400	394	389	383	378	372

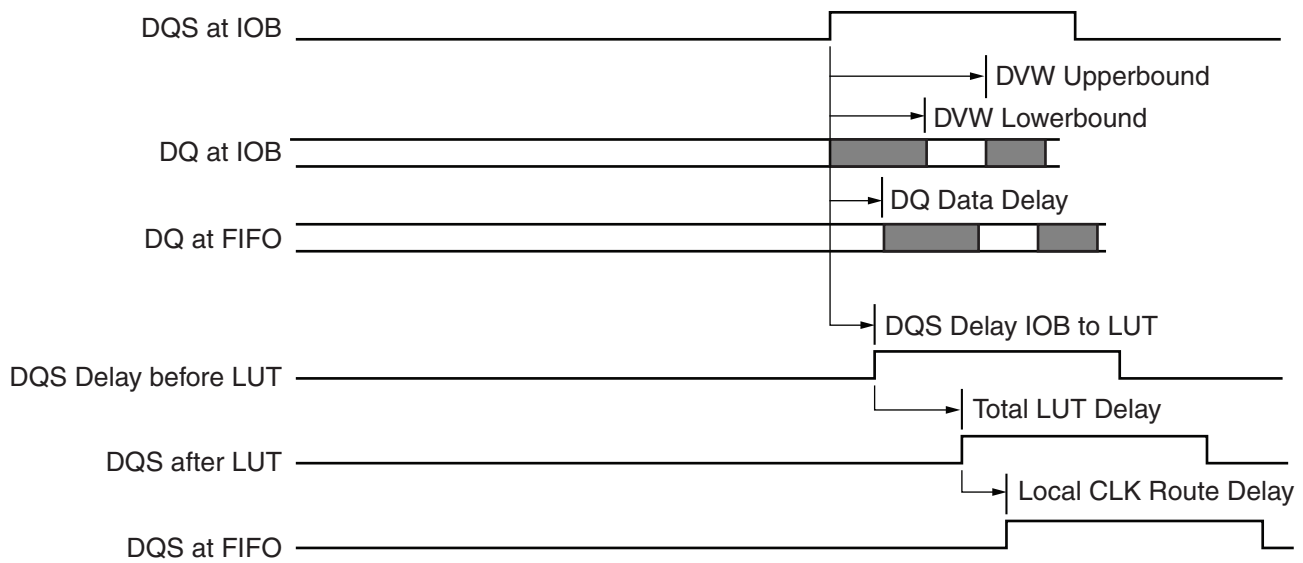
Table 2: LUT Selection for Left and Right Implementations (Cont'd)

Parameter (ps)	Nominal	90%	80%	70%	60%	50%	40%
Data Valid Window (Upper Bound)	1897	1923	1948	1974	1999	2025	2050

Notes:

1. In this table, the clock frequency is 166 MHz and the clock period is 6024 ps. The implementation is targeted to the XC3S700A-5FG484 device and interfaced to the Micron MT47H16M16XX-37E DDR2 SDRAM.
2. The total DQS delay in the table is generated from the MIG tool, version 2.1. DQS delay is the total delay for the route taken by the DQS from the input pad to the delay circuit plus the total number of LUT delays and the local clock route delay. The total extra DQS delay is the total DQS delay minus the data delay. This value should always fall within the data valid window.
3. Timing analysis is provided for the absolute minimum delay across process, temperature, and voltage variation.

Figure 10 shows the delay path of the DQ and DQS read capturing scheme. DQS at the FIFO is well inside the data valid window of DQ at the FIFO.



X454_10_040808

Figure 10: DQ and DQS Captured Read Timing

Table 3 shows the timing budget of the output data generation scheme when the Spartan-3A FPGA transmits data to the memory device using DDR output flip-flops.

Table 3: Write Data Timing

Parameter	Value (ps)	Leading Edge Uncertainties (ps)	Trailing Edge Uncertainties (ps)	Meaning
Clock Frequency (MHz)	166			Clock frequency (f).
T _{CLOCK}	6024			Clock period.
T _{CLOCK_PHASE}	3012			Clock phase is half of T _{CLOCK} .
T _{CLOCK_DUTY_CYCLE_DIST}	410			CLKOUT_DUTY_CYCLE_DLL parameter from <i>Spartan-3A FPGA Family: Data Sheet</i> [Ref 2].
T _{DATA_PERIOD}	2602			Data period is T _{CLOCK_PHASE} - T _{CLOCK_DUTY_CYCLE_DIST} .
T _{DSa} (REF)	350	350		DQ and DM logic level input setup time relative to DQS from <i>MT47H16M16BG-37E Data Sheet</i> [Ref 1].

Table 3: Write Data Timing (Cont'd)

Parameter	Value (ps)	Leading Edge Uncertainties (ps)	Trailing Edge Uncertainties (ps)	Meaning
T_{DHa} (REF)	350	0	350	DQ and DM logic level hold time relative to DQS from <i>MT47H16M16BG-37E Data Sheet</i> .
$T_{PACKAGE_SKEW}$	60	60	60	Package skew.
$T_{CLOCK_TREE_SKEW}$	50	50	50	Clock tree skew.
T_{CLKOUT_PHASE}	210	210	210	Phase offset between DCM outputs from <i>Spartan-3A FPGA Family: Data Sheet [Ref 2]</i> .
T_{JITTER}	0	0	0	The FPGA and DDR2 devices have a common clock. Thus, the effective jitter is zero.
$T_{PCB_LAYOUT_SKEW}$	50	50	50	Skew between data and strobes on the board.
Total Uncertainties		720	720	<ul style="list-style-type: none"> Leading edge uncertainties is the sum of T_{DS}, $T_{PACKAGE_SKEW}$, $T_{CLOCK_TREE_SKEW}$, T_{CLKOUT_PHASE}, T_{JITTER}, and $T_{PCB_LAYOUT_SKEW}$. Trailing edge uncertainties is the sum of T_{DH}, $T_{PACKAGE_SKEW}$, $T_{CLOCK_TREE_SKEW}$, T_{CLKOUT_PHASE}, T_{JITTER}, and $T_{PCB_LAYOUT_SKEW}$.
Margin	1162			Margin is equal to $T_{DATA_PERIOD} -$ (Leading edge uncertainties + Trailing edge uncertainties).

Table 4 shows the timing budget of the output address and control generation scheme when the Spartan-3A FPGA accesses the memory device using single data rate (SDR) output flip-flops.

Table 4: Address and Command Data Timing

Parameter	Value (ps)	Leading edge uncertainties	Trailing edge uncertainties	Meaning
Clock frequency (MHz)	166			Clock frequency (f).
T_{CLOCK}	6024			Clock period (1/f).
T_{IS}	500	500	0	Address and control input setup time from <i>MT47H16M16BG-37E Data Sheet [Ref 1]</i> .
T_{IH}	500	0	500	Address and control input hold time from <i>MT47H16M16BG-37E Data Sheet</i> .
$T_{PACKAGE_SKEW}$	60	60	60	Package skew.
T_{JITTER}	0	0	0	The FPGA and DDR2 devices have a common clock. Thus, the effective jitter is zero.
$T_{CLOCK_TREE_SKEW}$	50	50	50	Clock tree skew.

Table 4: Address and Command Data Timing (Cont'd)

Parameter	Value (ps)	Leading edge uncertainties	Trailing edge uncertainties	Meaning
$T_{PCB_LAYOUT_SKEW}$	50	50	50	Skew between address and control signals on the board.
T_{CLKOUT_PHASE}	210	210	210	Phase offset between DCM outputs from <i>Spartan-3A FPGA data sheet</i> [Ref 2].
Total Uncertainties		870	870	<ul style="list-style-type: none"> Leading edge uncertainties is the sum of the T_{IS}, $T_{PACKAGE_SKEW}$, T_{JITTER}, $T_{CLOCK_TREE_SKEW}$, $T_{PCB_LAYOUT_SKEW}$, and T_{CLKOUT_PHASE}. Trailing edge uncertainties is the sum of the T_{IH}, $T_{PACKAGE_SKEW}$, T_{JITTER}, $T_{CLOCK_TREE_SKEW}$, $T_{PCB_LAYOUT_SKEW}$, and T_{CLKOUT_PHASE}.
Margin	4284			Margin is equal to $T_{CLOCK} - (\text{Leading edge uncertainties} + \text{Trailing edge uncertainties})$.

The loopback timing data shown in Table 5 is related to FIFO write enable generation for the data input capture scheme and supplements the data in Table 1.

Table 5: Loopback Timing

Parameter	Leading edge delays (ps)	Trailing edge delays (ps)	Meaning
T_{CLOCK}	6024		Master clock period.
T_{CLKOUT_PHASE}	3012		Clock phase (half period).
Delay Details for DQS			
DQS Delay from IOB to LUT	450	450	
DQS Local Clock Route Delay	397	397	DQS line delay from output of LUT delay element.
Total DQS Delay	847	847	The LUT delay on DQS is not considered because both DQS and <code>rst_dqs_div</code> signals are by delayed the same amount.
Delay Details for Loopback			
<code>rst_dqs_div</code> Delay from IOB to LUT	388	388	Constrained using MAXDELAY constraints. Value from PAR report.
<code>delayed_rst_dqs_div</code> Delay to OR Gate	1290	1290	Constrained using MAXDELAY constraints. Value from PAR report.
OR Gate Delay	620	620	Implemented in a single LUT.
<code>fifo_1_wen</code> Delay from OR Gate	854	854	Constrained using MAXDELAY constraints. Value from PAR report.
Total Loopback Signal Delay	3152	3152	Sum of all delays listed.
Margin	3719	3719	

Table 6 shows the timing budget and evaluates the path from the system clock source, through the Spartan-3A FPGA, to the clock input on the memory device.

Table 6: Clock-to-Memory Timing

Parameter	Leading edge delays (ps)	Trailing edge delays (ps)	Meaning
T_{CLOCK}	6024		Master clock period.
$T_{\text{CLKOUT_PHASE}}$	3012		Clock phase (half period).
CLKIN_CYC_JITT_DLL_HF	150	150	Cycle-to-cycle jitter of the oscillator. Maximum value is set to CLKIN_CYC_JITT_DLL_HF from <i>Spartan-3A FPGA data sheet</i> [Ref 2].
CLKOUT_DUTY_CYCLE_DLL	250	250	DCM and BUFG duty cycle distortion from CLKOUT_DUTY_CYCLE_DLL parameter (1% of CLKIN period [T_{CLOCK}] plus 190 ps) for CLK0 and CLK90.
Clock Phase from DCM and BUFG	2687	3337	Derived after duty cycle distortion and half of the jitter values are subtracted from $T_{\text{CLKOUT_PHASE}}$.
Memory Clock Jitter	125	125	Taken from <i>MT47H16M16BG-37E Data Sheet</i> [Ref 1].
Memory Duty Cycle Distortion	2711	3313	Taken from <i>MT47H16M16BG-37E Data Sheet</i> .
Memory Input Clock Timing	2643	3396	Derived after jitter and duty cycle distortion parameters are applied to the input clock.
Margin	44	59	

Local Clocking Resources

The delayed strobe in this design uses the local clocking resources available in the device for the clock routing. Full hex lines (spanning six configurable logic blocks) that have low skew are located throughout the device.

The left and right implementations use VFULLHEX lines for local clock routing. The top and bottom implementations use vertical long (VLONG), VFULLHEX, and horizontal Full Hex (HFULLHEX) lines for local clock routing. This route is more complex than the left and right sides. The delay and skew of this local clock route is higher than the left and right local clock routes. This results in a limit of 133 MHz memory clock speed in a top and bottom implementation.

IOBS

All FPGA input and output signals are implemented in the IOBS module. All address and control signals are registered going into and coming out of the IOBS module.

User Interface Signals

For a detailed description of the user interface signals and a write/read diagram, refer to the *Xilinx Memory Interface Generator (MIG) User Guide* [Ref 3].

DDR2 SDRAM Auto Refresh

The DDR2 SDRAM must be refreshed once every 7.8 μs with an auto_refresh command. The auto_refresh command is asserted with SYS_CLK. The ar_done signal is asserted by the DDR2 SDRAM controller upon completion of the auto_refresh command.

Reference Design

The DDR2 SDRAM controller reference design is integrated into the MIG tool. This tool has been integrated with the Xilinx CORE Generator™ software. For the latest version of the design, download the IP update from:

<http://www.xilinx.com/support/download/index.htm>.

Table 7 shows the reference design matrix.

Table 7: Reference Design Matrix

Parameter	Description
General	
Developer Name	Xilinx
Target Devices (Stepping Level, ES, Production, Speed Grades)	Spartan-3A DSP FPGA XC3SD3400A-4FG676, Spartan-3A FPGA XC3S700A-4FG484
Source Code Provided?	HDL code is generated by MIG tool
Source Code Format	VHDL, Verilog
Design Uses Code or IP from Existing Reference Design, Application Note, 3rd party, or CORE Generator Software?	Yes
Simulation	
Functional Simulation Performed?	Yes
Timing Simulation Performed?	Yes
Testbench Provided for Functional and Timing Simulations?	Yes
Testbench Format	VHDL, Verilog
Simulator Software and Version	ModelSim 6.3c
SPICE/IBIS Simulations?	No
Implementation	
Synthesis Software Tools and Version	XST, version 10.1
Implementation Software Tools and Version	ISE® software, version 10.1
Static Timing Analysis Performed?	Yes
Hardware Verification	
Hardware Verified?	Yes
Hardware Platform Used for Verification	Spartan-3A FPGA Starter Kit, Spartan-3A DSP FPGA 3400A Development Board

References

This document uses the following references:

1. Micron MT47H16M16BG-37E DDR2 SDRAM Data Sheet
<http://download.micron.com/pdf/datasheets/dram/ddr2/256MbDDR2.pdf>
2. [DS529](#), *Spartan-3A FPGA Family: Data Sheet*
3. [UG086](#), *Xilinx Memory Interface Generator (MIG) User Guide*

Additional Resources

The following resources provide additional information useful to this application note:

1. [XAPP458](#), *Implementing DDR2-400 Memory Interfaces in Spartan-3A FPGAs*
2. [DS610](#), *Spartan-3A DSP FPGA Family: Data Sheet*

Conclusion

It is possible to implement a high-performance DDR2 SDRAM interface for Spartan-3 generation FPGAs. This design has been simulated, synthesized in Xilinx Synthesis Technology (XST) and Synplicity software, and taken through the Xilinx Project Navigator flow.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
12/06/04	1.0	Initial Xilinx release.
06/07/07	1.1	Revised to update references to MIG tool.
06/11/07	1.1.1	Trademark update.
05/09/08	2.0	Major revision. Added detailed descriptions of the read data capture, read data clocking, and delay calibration circuits used in the design. Added Reference Design Matrix (Table 7).
01/20/09	2.1	Updated " Delay Calibration Circuit ," page 6 and Table 6 .

Notice of Disclaimer

Xilinx is disclosing this Application Note to you "AS-IS" with no warranty of any kind. This Application Note is one possible implementation of this feature, application, or standard, and is subject to change without further notice from Xilinx. You are responsible for obtaining any rights you may require in connection with your use or implementation of this Application Note. XILINX MAKES NO REPRESENTATIONS OR WARRANTIES, WHETHER EXPRESS OR IMPLIED, STATUTORY OR OTHERWISE, INCLUDING, WITHOUT LIMITATION, IMPLIED WARRANTIES OF MERCHANTABILITY, NONINFRINGEMENT, OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT WILL XILINX BE LIABLE FOR ANY LOSS OF DATA, LOST PROFITS, OR FOR ANY SPECIAL, INCIDENTAL, CONSEQUENTIAL, OR INDIRECT DAMAGES ARISING FROM YOUR USE OF THIS APPLICATION NOTE.