LatticeSC/M DDR/DDR2 SDRAM Memory Interface User's Guide

July 2008 Technical Note TN1099

Introduction

FPGA logic designers are often faced with the need to communicate with external memories, and applications are requiring increasingly large I/O channel bandwidths. In response to these demands, the industry has defined several new memory devices with their associated protocols (e.g., QDR-SRAM, DDR/DDR2 SDRAM, RLDRAM), each being optimized for a particular segment of the high-bandwidth market. This user's guide discusses a memory interface for a Double-Data-Rate SDRAM (DDR/DDR2 SDRAM), implemented in the LatticeSC™ and LatticeSCM™ FPGAs. LatticeSC/M FPGAs support data rates for individual DDR2 devices and SODIMM modules at rates up to 667 Mbps for the embedded MACO block and rates up to 533 Mbps for soft IP cores.

DDR/DDR2 SDRAM

Description

Overview

Double Data Rate (DDR) memory interfaces prefetch two data bits per clock cycle from the memory core to the I/Os and then output the data on both the rising and falling edges of the clock. DDR can transfer two data words per clock cycle, as opposed to SDRAM's one word per clock cycle, effectively doubling the speed at which data is transferred. The data is sampled using a clock that accompanies the data so that skew due to I/O buffers and board trace delays is automatically nulled out.

The second version of DDR (DDR2) employs several enhancements that increase bus bandwidth and simplify operation. Lower supply voltages permit smaller design geometries and hence higher speeds; memory architecture changes permit larger storage capacity per device; posted CAS support simplifies instruction execution; and on-die termination (ODT) reduces support device count.

History

Here is a brief timeline of the DDR2 SDRAM development:

- 09/1998 - Micron samples 64Mm DDR SDRAM
- 06/2000 - JEDEC releases initial DDR SDRAM specification
- 02/2002 - JEDEC releases DDR2 SDRAM specification
- 05/2002 - Samsung develops prototype DDR2 SDRAM
- 08/2002 - Micron demonstrates DDR2 SDRAM application
- 01/2003 - Samsung Introduces first 4Gb DDR memory module
- 10/2003 - Micron samples 1Gb DDR2

Differences Between DDR and DDR2 SDRAM

Although DDR (retroactively referred to as DDR1) and DDR2 are very similar in functionality, there are some important differences, as listed in Table 1

Table 1. Differences Between DDR1 and DDR2

Feature	DDR (DDR1) Mode	DDR2 Mode
CAS Latency	1, 2 or 3 Clocks	2, 3, 4 or 5 Clocks
Write Latency	1 Clock	Read Latency - 1
Burst Length	2, 4, 8 Words	4, 8 Words
Differential DQS Support	No	Supported
Redundant DQS Support	No	Supported
Ability to Interrupt 8-Word Bursts (Write and Read)	No	Supported
Number of Banks per Device	4	4 or 8
On-Die Termination	No	Supported
Posted CAS Additive Latency Mode	No	Supported

Specifications and Performance

The DDR/DDR2 SDRAM is targeted to applications requiring:

- High data density
- Low cost per bit
- Unbalanced read/write ratios (ratio ≠ 1:1)

DDR/DDR2 SDRAM drawbacks (as compared to SRAMs):

- Requires refresh
- Requires initialization
- Greater latency than SRAM architectures

Table 2 lists some of the characteristics of the DDR and DDR2 SDRAM solutions.

Table 2. DDR and DDR2 SDRAM Characteristics

DDR/DDR2 SDRAM Feature	DDR SDRAM Value	DDR2 SDRAM Value	Units
Max. Frequency	200	333	MHz
DLL	Yes (optional)		—
Max. Data Rate	400	667	MWord/s
Max. Data Density/Device	1024	1024	Mbits
Max. Row Address Size	14	14	Bits
Max. Column Address Size	11	11	Bits
Max. Bank Address Size	2	3	Bits
Max. Number Of Chip Selects	1/2/4/8	1/2	—
Data Width per Device	8/16/32/40/64/72	8/16/32/40/64/72	Bits
Supply Voltage	2.5	1.8	Volts
I/O Protocol	SSTL-25	SSTL-18	—
Refresh Cycles Needed?	Yes		—

Implementation Challenges

Among the most difficult aspects of a DDR/DDR2 SDRAM Memory Interface implementation is the clock net design. The LatticeSC/M makes use of the capabilities of one or two PLLs and one DLL to generate the necessary internal clocks and the K/K# clock to the DDR/DDR2 SDRAM, as shown in Figure 1. PLL1 is used for clock multiplication and 90º phase shifting, and is typically shared by all DDR/DDR2 Memory Interfaces on the device that are operating off the same clock. PLL2 is optional and is used to compensate for internal and I/O delay. DLL1 is used to generate a delay value to drive the programmable delays on the DQS input, causing a specific time delay on this signal. A DLL, not a PLL, is used here because DQS is non-periodic. DLL1 is also typically shared by all Memory Interfaces operating off the same clock. All these features enable the Memory Interface to compensate for device variation and board trace delay, and keep the clock for the receive data in the center of the data eye. It is important at data rates of 400Mb/s and above that there be automatic, dynamic compensation for the effects of variations due to device process, supply voltage and operating temperature (PVT).

LatticeSC/M Features That Solve the Implementation Challenges

The LatticeSC/M family provides many features that make high-speed systems easier to build. Among them are:

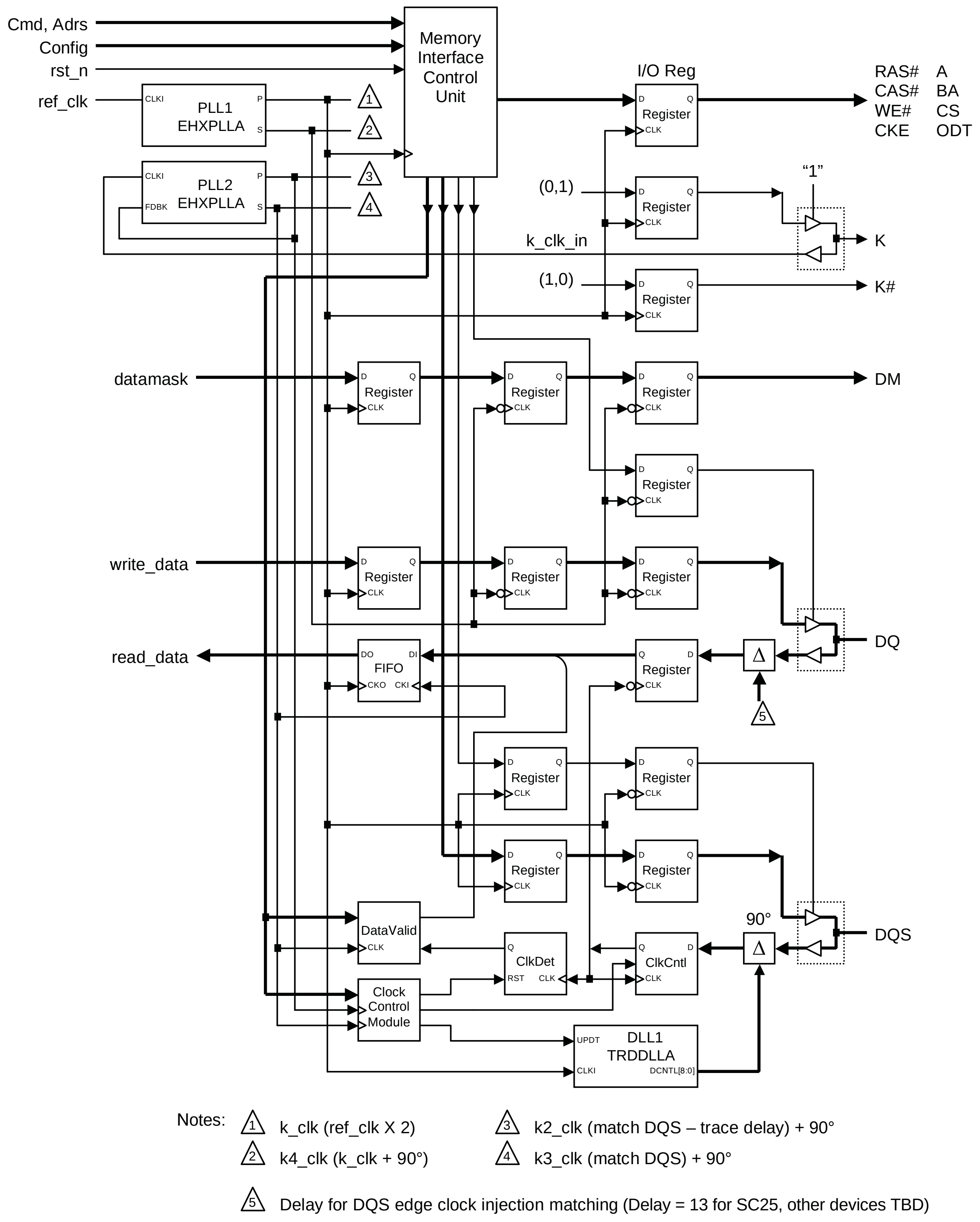
- Per-pin DDR (Double-Data-Rate) capability built in. Usable in both DDR and QDR (Quad-Data-Rate) implementations.
- Per-pin DDR incorporates hardwired serial-to-parallel and parallel-to-serial conversion. Useful in higher-speed implementations.
- I/Os that support SSTL (Stub-Series Terminated Logic), used with high-speed DDR SDRAM devices, and HSTL (High-Speed Transceiver Logic), used with high-speed SDRAM devices.
- Optional delays (INDEL) built into the I/O paths that can be statically or dynamically set on a per-pin or bus basis. The I/O pin delay blocks match the delay blocks that are internal to the DLLs, so that the DLLs can adaptively determine the optimum delay setting, and then apply that setting to the I/O pin delay blocks. In this way, the DLL can compensate for effects of frequency, voltage, temperature and device-to-device variation.
- PLLs that can generate two high-speed, low-jitter clocks that are in quadrature (90º out of phase with each other).
- DLLs that can be used to modify the delays of non-periodic signals, such as the DQS input clocks.
- Special clocking structures that allow the DQS input to be synchronously disabled.
- Special input buffer structure that provides a single-shot output signal to the FPGA fabric upon the first clock edge after a user-defined reset is released.
- Integrated I/O pin output serial terminations that allow output buffers to match the 50Ω traces found on the printed circuit board to reduce reflections. These serial terminations are compensated so as to remain constant over the operating range of the device (PVT compensated).
- Integrated I/O pin parallel terminations that are enabled when bi-directional I/O pins are inputs, but disabled when outputs (for DDR2 SDRAM use). These parallel terminations are also PVT compensated.
- High-speed FPGA fabric.

LatticeSC/M DDR/DDR2 Memory Interface Implementation Details

Block Diagram

Figure 1 shows the block diagram for a typical DDR/DDR2 Memory Interface implemented in the LatticeSC/M FPGA.

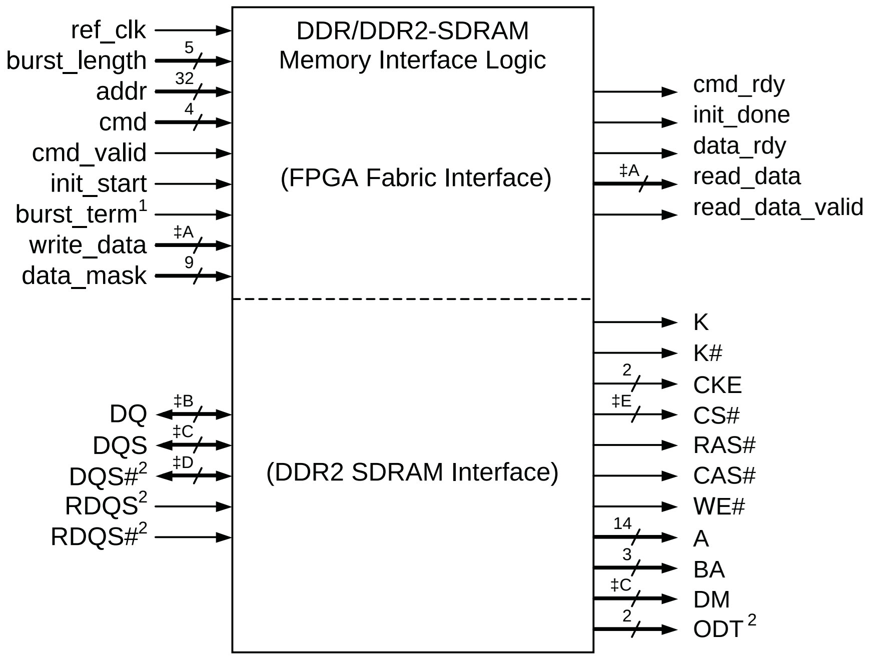
Figure 1. DDR/DDR2 Memory Interface Block Diagram



Pinout

Figure 2 shows the pinout for a typical DDR/DDR2 SDRAM Memory Interface implemented in the LatticeSC/M FPGA. The internal FPGA interface is on the top, and the external SSTL SDRAM interface is on the bottom.

Figure 2. DDR/DDR2 SDRAM Memory Interface Pinout Interface (Typical)



Notes:

1 DDR1 only

2 DDR2 only

‡A 16/32/64/80/128/144 bits wide

‡B 8/16/32/40/64/72 bits wide

‡C 1/2/4/5/8/9 bits wide

‡D 1/2/4/5 bits wide

‡E 8 (DDR), 2 (DDR2)

Signal List

Table 3 lists the DDR/DDR2 SDRAM Memory Interface signals and their descriptions.

Table 3. Interface Signals

Pin Name	Direction	Width	Description
A	Out to SDRAM	14	Address.
BA		3	Bank Address. BA2 is for future expansion.
RAS#		1	Row Address Select (active LO).
CAS#		1	Column Address Select (active LO).
WE#		1	Write Enable (active LO); HI = Read, LO = Write.
CS#		8 (DDR) 2 (DDR2)	Chip Selects; one for each DDR/DDR2 SDRAM device (two per DIMM). When LO/HI, accesses are valid/masked.
K		1	High speed SDRAM Clock (differential pair; must be routed as such). Frequency is typically double the ref_clk frequency.
K#		1	
CKE		1/2	Clock Enable; one for each DDR/DDR2 SDRAM device (two per DIMM). CKE HI activates the SDRAM clock and I/O buffers.
DM		1/2/4/ 5/8/9	Data Mask Bits, one DM bit for each byte of write data.
ODT		1/2	On-Die Termination Control; one for each DDR2 SDRAM device.
DQ	In from / Out to SDRAM	8/16/ 32/40/ 64/72	Data Input/Output: Bidirectional write/read data bus.
DQS		1/2/ 4/5/ 8/9	Data Strobe Input/Output. One DQS strobe bit for each byte of DQ data. Transition edge sent centered in data during write, and received aligned with data transition during read.
DQS#		1/2/ 4/5	Data Strobe Input/Output, complement (DDR2 only). One DQS# strobe bit for each byte of DQ data. Valid only when differential DQS mode is enabled in EMR0; valid for a maximum of five bytes of data.
RDQS	In from SDRAM	1	Redundant DQS (DDR2 only). Valid only for the x8 configuration, and only when RDQS is enabled in EMR.
RDQS#		1	Redundant DQS# (DDR2 only). Valid only for the x8 configuration, and only when RDQS and DQS# are enabled in EMR.
addr	In from FPGA	32	Address.
cmd		4	Command bus. See Table 6 for a listing of valid commands.
cmd_valid		1	Command valid. When active HI, indicates that the FPGA logic has a command ready to execute on bus cmd. If signal cmd_rdy is simultaneously asserted, the command is initiated.
burst_length		5	Burst length. Valid values are 1-31, measured in DDR/DDR2 SDRAM bursts. For example, a length of three indicates the execution of three 4-word or 8-word accesses.
ref_clk		1	Reference clock.
rst_n		1	Reset.
write_data		16/32/ 64/80/ 128/144	Write data bus.
data_mask		1/2/ 4/5/ 8/9	Write data bus byte write mask.
init_start		1	A one-clock-wide active-Hi pulse of this signal initiates an initialization sequence. Initialization completion is signaled by init_done.
burst_term		1	Terminates burst in progress (DDR1 only).

Table 3. Interface Signals (Continued)

Pin Name	Direction	Width	Description
cmd_rdy	Out to FPGA	1	When active HI, indicates that the Memory Interface is ready to accept a new command. If signal cmd_valid is simultaneously asserted, the command on bus cmd is initiated.
init_done		1	A one-clock-wide active-HI pulse of this signal indicates completion of an initialization sequence that was started by signal init_start.
data_rdy		1	Active-HI; accompanies valid data on bus write_data.
read_data		16/32/ 64/80/ 128/144	Read data bus.
read_data_valid		1	Valid to accompany read_data

Note: Signals are actual for SDRAM-side interface, and typical for FPGA-side interface.

DDR/DDR2 SDRAM Mode/Extended Mode Registers Definition

Tables and 5 list the various fields in the Mode and Extended Mode registers, and show their defined values. These registers customize the SDRAM's performance and behavior for the particular application and operating frequency, and are loaded during the initialization procedure.

Table 4. DDR/DDR2 SDRAM Mode Register Definition

BA[1:0]	Name	Bits	Value	Description	
				DDR1	DDR2
00 (MR)	—	A13	—	‡	
	Power-Down	A12	0	‡	Fast Exit
			1	‡	Slow Exit
	Write Recovery	A11, A10, A9	000	‡	
			001	‡	2
			010	‡	3
			011	‡	4
			100	‡	5
			101	‡	6
			110	‡	
			111	‡	
	DLL	A8	0	DLL Disabled	
			1	DLL Enabled	
	Test Mode	A7	0	‡	Normal
			1	‡	Test
	CAS# Latency	A6, A5, A4	000	‡	
			001	‡	
			010	‡	
			011	‡	3
			100	‡	4
			101	‡	5
			110	‡	
			111	‡	
	Burst Type	A3	0	Sequential	
			1	Interleaved	
	Burst Length	A2, A1, A0	000	‡	
			001	2	‡
			010	4	
			011	8	
			100	‡	
			101	‡	
			110	‡	
			111	‡	

Note: ‡ denotes Reserved.

Table 5. DDR/DDR2 SDRAM Extended Mode Registers Definition

BA[1:0]	Name	Bits	Value	Description	
				DDR1	**DDR2**
01 (EMR)	—	A13	—	‡	
	Out	A12	0	‡	Enabled
			1	‡	Disabled
	RDQS	A11	0	‡	Disabled
			1	‡	Enabled
	DQS#	A10	0	‡	Enabled
			1	‡	Disabled
	OCD Program	A9, A8, A7	000	‡	OCD Not Supported
			001	‡	
			010	‡	
			011	‡	
			100	‡	
			101	‡	
			110	‡	
			111	‡	OCD Default State
	Rtt	A6, A2	00	‡	Disabled
			01	‡	75Ω
			10	‡	150Ω
			11	‡	50Ω
	Posted CAS# Additive Latency	A5, A4, A3	000	‡	0
			001	‡	1
			010	‡	2
			011	‡	3
			100	‡	4
			101	‡	
			110	‡	
			111	‡	
	ODS	A1	0	Output Drive Strength: Full (18Ω Target)	
			1	Output Drive Strength: Reduced (40Ω Target)	
	DLL	A0	0	Enable (Normal)	
			1	Disable (Test/Debug)	
10 (EMR2)	Reserved	A13-A0	—	‡	
11 (EMR3)	Reserved	A13-A0	—	‡	

Note: ‡ denotes Reserved.

DDR/DDR2 SDRAM Commands

Table 6 lists the JEDEC-defined commands that the DDR/DDR2 SDRAM recognizes.

Table 6. JEDEC DDR/DDR2 SDRAM Commands

Function	CKE Prev → Current	OpCode[1]	BA[2:0]	A[13:11], A[11,9:0]	A[10]
Load Mode Reg	H	0	BA[3]	MR Value[4]	
Refresh	H	1	X[5]	X[5]	X[5]
Self Refresh Entry	H → L	1			
Self Refresh Exit	L → H	7 or ♦[2]			
Single Bank Precharge	H	2	Bank Address		L
All Banks Precharge	H	2	X[5]		H
Bank Activate	H	3	Bank Address	Row Address	
Write	H	4		Column Address	L
Write with Auto Precharge	H	4			H
Read	H	5			L
Read with Auto Precharge	H	5			H
No Operation	H → X[5]	7	X[5]		
Device Deselect	H → X[5]	♦[2]			
Power-down Entry	H → L	7 or ♦[2]			
Power-down Exit	L → H	7 or ♦[2]			

1. The OpCode consists of CS#, RAS#, CAS# and WE# concatenated to produce a hexadecimal value. The value is sampled on the rising edge of clock CK.
2. The symbol "♦" denotes an OpCode value of H:X:X:X (i.e., CS# = HI)
3. During a Load Mode operation, BA[2:0] specifies the Mode/Extended Mode Register to be loaded:
   - 000 selects MR
   - 001 selects EMR
   - 010 selects EMR2
   - 011 selects EMR3
   - 1XX is undefined
4. During a Load Mode operation, A[13:0] specifies the data to be loaded into the selected Mode/Extended Mode Register.
5. "X" denotes either "H" or "L" (but a defined logic level)

DDR/DDR2 SDRAM Memory Interface FPGA Interface Commands

Table 7 lists the commands available to the FPGA logic for controlling SDRAM operation (example implementation).

Table 7. FPGA Interface Command Summary

Command	Acronym	Commandcmd[3:0]	SDRAM Address
No-Op	—	0 (0000)	—
Read	READ	1 (0001)	Column
Write	WRITE	2 (0010)	Column
Read with Auto Precharge	READA	3 (0011)	Column
Write with Auto Precharge	WRITEA	4 (0100)	Column
Power Down	PWRDN	5 (0101)	—
Load Mode Register	LOAD_MR	6 (0110)	Opcode A15-A0
Self Refresh	SELF_REFRESH	7 (0111)	Column
No-Op	—	8 (1000)	—
Read Interrupt[1]	READ_INT	9 (1001)	Column
Read Interrupt with Auto Precharge[1]	READ_INTA	A (1010)	Column
Write Interrupt[1]	WRITE_INT	B (1011)	Column
Write Interrupt with Auto Precharge[1]	WRITE_INTA	C (1100)	Column
No-Op	—	D (1101)	—
No-Op	—	E (1110)	—
No-Op	—	F (1111)	—

1. DDR2 SDRAM only.

Description of Operation

The DDR/DDR2 SDRAM features high-bandwidth access to a very dense and cost-efficient memory. DDR/DDR2 SDRAM is preferred over QDR (Quad Data Rate) SRAM in situations where:

- Large memory size is required; and
- Low cost per bit is of prime concern.

The limitations of DDR/DDR2 SDRAM are:

- Since writes and reads share one bidirectional data bus, total bandwidth is halved as compared to the QDR architecture (this is significant in situations where the ratio of writes to reads is close to 1:1).
- Interruptions in data transfer are necessary for refresh.
- Access latency is relatively high.
- Requires initialization after power-up, and activation/precharge of rows before/after accesses (Memory Interface simplifies this).

The LatticeSC/M DDR/DDR2 SDRAM Memory Interface simplifies the designer's task when building a DDR/DDR2 SDRAM interface into an FPGA design. The critical timing parameters (listed in Table 11) are provided to the Memory Interface at the time the FPGA design is compiled, and the Memory Interface sees to it that these parameters are observed during device operation. In the typical implementation discussed in this document, this is accomplished via the signal cmd_valid, which is asserted when the Memory Interface is ready to accept a new command; the Memory Interface is thus able to throttle the instruction stream and ensure that timing parameters are met.

The DDR/DDR2 SDRAM Memory Interface allows the FPGA user logic to access a DDR/DDR2 SDRAM with a minimum of overhead activity, while still providing access to the DDR/DDR2 SDRAM's special features.

Any implementation must execute an initialization process (described below), after which reads and writes can be performed in any order, without concern for low-level activities such as refresh, bank activation and precharging, and without concern for adherence to all the SDRAM timing requirements.

Initialization

Before the DDR/DDR2 SDRAM enters normal operational mode, it must undergo an initialization sequence, as detailed below for DDR2 (DDR1 is similar):

1. For a minimum of 200µs after stable power and clock (CK, CK#), apply NOP or DESELECT commands and take CKE high.
2. Wait a minimum of 400ns, then issue a PRECHARGE ALL command.
3. Issue a LOAD MODE command to the EMR2 register. (To issue an EMR2 command, provide LOW to BA0 and BA2, provide HIGH to BA1).
4. Issue a LOAD MODE command to the EMR3 register. (To issue an EMR3 command, provide HIGH to BA0 and BA1, provide LOW to BA2).
5. Issue a LOAD MODE command to the EMR register to enable DLL. (To issue a DLL Enable command, provide LOW to BA1, BA2, and A0, provide HIGH to BA0. Bits E7, E8, and E9 must all be set to 0).
6. Issue a LOAD MODE command for DLL Reset. 200 cycles of clock input is required to lock the DLL. (To issue a DLL reset, provide HIGH to A8 and provide LOW to BA2, BA1, and BA0.) CKE must be HIGH the entire time.
7. Issue PRECHARGE ALL command.
8. Issue two or more REFRESH commands.
9. Issue a LOAD MODE command with LOW to A8 to initialize device operation (i.e., to program operating parameters without resetting the DLL).
10. Issue a LOAD MODE command to the EMR to enable OCD default by setting bits E7, E8, and E9 to 1 and setting all other desired parameters.
11. Issue a LOAD MODE command to the EMR to enable OCD exit by setting bits E7, E8, and E9 to 0 and setting all other desired parameters.

This sequence is complex, specific, and includes time delays. Fortunately, the Memory Interface can execute this task automatically, requiring only that the FPGA user logic initiate the sequence by asserting a signal (init_start in our typical implementation) for one clock cycle. The user logic then waits until initialization completes, as indicated by the assertion of another signal (init_done) for one clock cycle. Initialization can be performed at any time, but must be performed after initial power-up of the DDR/DDR2 SDRAM.

Initialization will place the DDR/DDR2 SDRAM in the default state programmed into the Memory Interface. The FPGA user logic can then overwrite any default parameter value by rewriting the Mode and Extended Mode Registers. For example, the user logic may wish to specify 8-word bursts rather than the default 4-word bursts.

Posted CAS Support (Additive Latency - DDR2 Only)

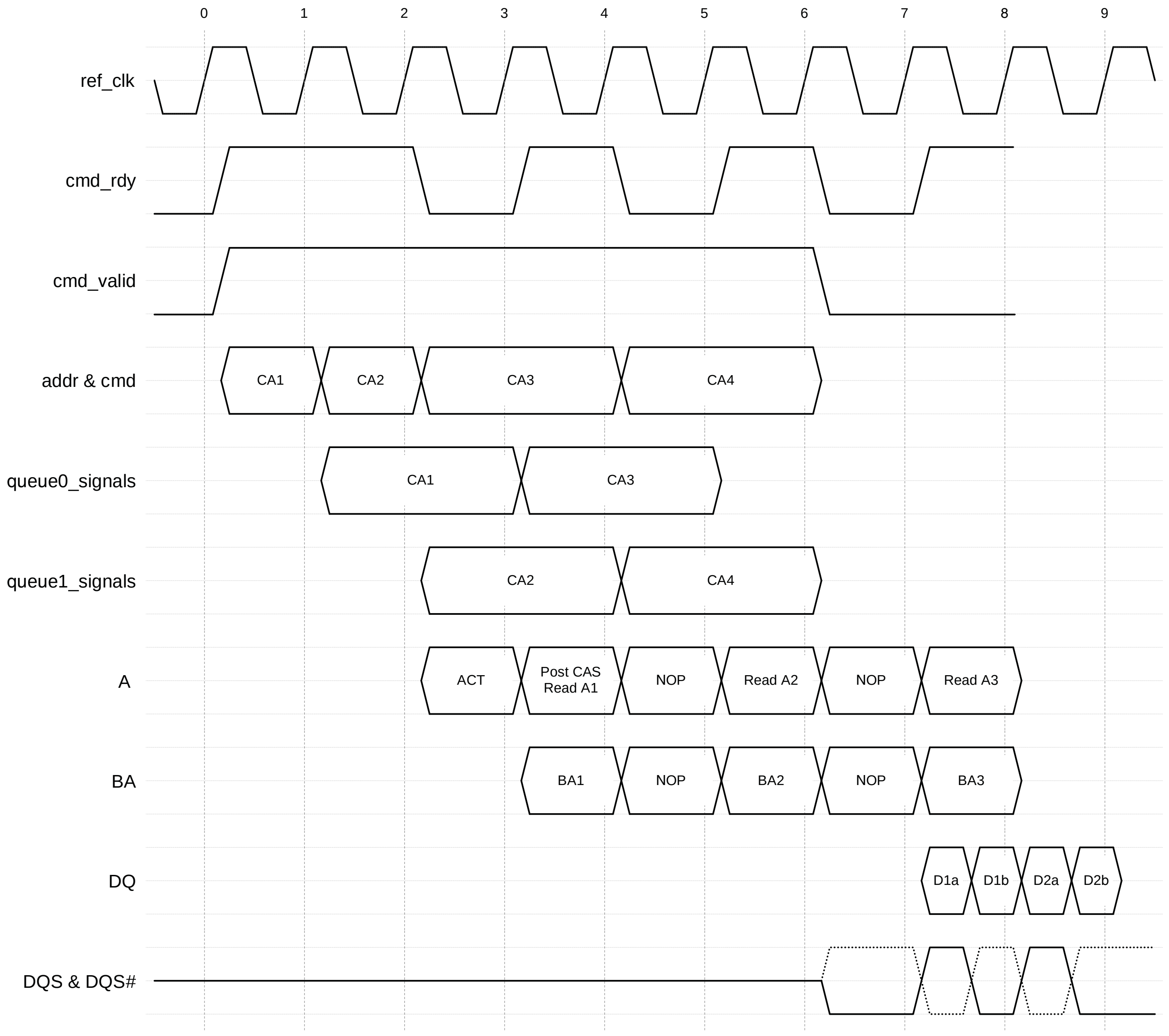
The feature Posted CAS is new to the DDR2 SDRAM specification, and is supported by the DDR2 SDRAM Memory Interface. In Posted CAS operation, read and write operations experience an intentionally added delay called Additive Latency. Additive Latency, whose value is specified by bits E5-E3 of the Extended Mode Register, holds up the internal issue of read and write commands for the specified number of clock cycles, thus increasing the effective latency by that number of clock cycles. Read Latency (RL) now becomes Additive Latency (AL) plus CAS Latency (CS). Write Latency (WL) is still RL -1. Typical Posted CAS timing is shown in Figure 3. In summary,

$$RL = AL + CL \text{ and}$$

$$WL = RL - 1 = AL + CL - 1$$

When the Additive Latency value is properly specified, it allows read and write commands to be placed immediately after their associated activate commands, simplifying command issue and preventing gaps that would otherwise occur because ideal timing would place commands on top of each other. Note that the DDR/DDR2 SDRAM Activate command is generated transparently by the Memory Interface. If the Additive Latency value is not properly selected, the Memory Interface may not be able to generate gapless data transfer streams. Nevertheless, it will still generate proper command sequences and correctly access the SDRAM.

Figure 3. Posted CAS Read Followed by Read (BL=4, AL=2, CL=3) – DDR2 Only



On-Die Termination (ODT)

On-Die Termination (ODT) is a significant feature of DDR2 SDRAMs that is designed to improve signal integrity by allowing the memory device to turn on/off the termination on the input buffers. On the memory device, the DQ, DQS and DM pins are terminated with ODT. Bits 2 and 6 in the Extended Mode Register (EMR), which is programmed using the EMRS command, define the effective termination resistance. This effective resistance can be 75Ω or 150Ω. The Memory Interface provides an ODT pin per device that is used to turn on/off this termination resistance. Off-chip termination is recommended for all the address and control inputs such as A, CS#, WE#, RAS#, CAS#, BA and CKE. The clock pair K and K# (also K copy and K# copy) is treated as a differential pair and differentially terminated on a DIMM.

On-Die Termination (ODT) on the DDR2 SDRAM is supported by the Memory Interface. By default it is inactive, and can be activated via a Load Mode Register command at any time after initialization is performed. The two-bit bus odt[1:0] then provides dynamic control of the ODT input(s) on the DDR2 SDRAM.

The discussion of ODT thus far is of the DDR2 SDRAM ODT capability, and the ability of the Memory Interface to control it. However, the LatticeSC/M I/Os also provide ODT capability on the Memory Interface end of the bidirectional signal; it is automatic and dynamic (deactivated/activated when the I/O buffer is active/tristate). Among the impedance values available are 60, 75 and 150Ω.

DDR1 devices do not support ODT. Consequently, terminations on the memory device end must be handled by discrete on-board terminations. Nevertheless, the LatticeSC/M I/Os still support ODT on their end.

When the Memory Interface is attached to a single DDR2 SDRAM device, control of the ODT signal sent to the DDR2 SDRAM is relatively simple: ODT is enabled during writes and disabled during reads. However, when more than one device is being controlled (JEDEC permits up to four, configured as either two DIMM slots of two ranks each, or four single-rank DIMM slots), the control is more complex.

When two DIMM slots are used, Tables 8 (writes) and 9 (reads) give the JEDEC-recommended ODT control configurations. When four DIMM slots are used, JEDEC recommends that discrete terminations on the motherboard be used in place of ODT. These recommendations are based on simulations of the various possible scenarios.

Table 8. Two-Slot ODT Control for Writes

Configuration		**Write to**	**Target DQ ODT Resistance (RTT)**				
			Memory Interface	**Module in Slot1**		**Module in Slot2**	
Slot1	**Slot2**			**Rank1**	**Rank2**	**Rank1**	**Rank2**
2R	2R	Slot 1	Hi-Z	Hi-Z	Hi-Z	75Ω	Hi-Z
		Slot 2	Hi-Z	75Ω	Hi-Z	Hi-Z	Hi-Z
2R	1R	Slot 1	Hi-Z	Hi-Z	Hi-Z	75Ω	—
		Slot 2	Hi-Z	75Ω	Hi-Z	Hi-Z	—
1R	2R	Slot 1	Hi-Z	Hi-Z	—	75Ω	Hi-Z
		Slot 2	Hi-Z	75Ω	—	Hi-Z	Hi-Z
1R	1R	Slot 1	Hi-Z	Hi-Z	—	75Ω	—
		Slot 2	Hi-Z	75Ω	—	Hi-Z	—
2R	—	Slot 1	Hi-Z	150Ω	Hi-Z	—	—
—	2R	Slot 2	Hi-Z	—	—	150Ω	Hi-Z
1R	—	Slot 1	Hi-Z	150Ω	—	—	—
—	1R	Slot 2	Hi-Z	—	—	150Ω	—

Table 9. Two-Slot ODT Control for Reads

Configuration		Read From	Target DQ ODT Resistance (RTT)				
			Memory Interface	Module in Slot1		Module in Slot2	
Slot1	Slot2			Rank1	Rank2	Rank1	Rank2
2R	2R	Slot 1	150Ω	Hi-Z	Hi-Z	75Ω	Hi-Z
		Slot 2	150Ω	75Ω	Hi-Z	Hi-Z	Hi-Z
2R	1R	Slot 1	150Ω	Hi-Z	Hi-Z	75Ω	—
		Slot 2	150Ω	75Ω	Hi-Z	Hi-Z	—
1R	2R	Slot 1	150Ω	Hi-Z	—	75Ω	Hi-Z
		Slot 2	150Ω	75Ω	—	Hi-Z	Hi-Z
1R	1R	Slot 1	150Ω	Hi-Z	—	75Ω	—
		Slot 2	150Ω	75Ω	—	Hi-Z	—
2R	—	Slot 1	75Ω	Hi-Z	Hi-Z	—	—
—	2R	Slot 2	75Ω	—	—	Hi-Z	Hi-Z
1R	—	Slot 1	75Ω	Hi-Z	—	—	—
—	1R	Slot 2	75Ω	—	—	Hi-Z	—

Output Drive Strength (ODS)

The Output Drive Strength selection feature of the DDR2 SDRAM is available to users of the Memory Interface. The Full Strength mode is active by default; the Reduced Strength (approximately. 60% of Full Strength) mode can be activated via a Load Mode Register command at any time after initialization is performed.

DQS# and RDQS Enable

The DQS# and RDQS Enable features of the DDR2 SDRAM are available to users of the Memory Interface. By default, DQS# is enabled and RDQS is disabled. Either or both can be modified via a Load Mode Register command at any time after initialization is performed.

DQS# is used with DQS to implement differential (instead of single-ended) DQS signaling. Differential signaling offers improved signal timing, reduced crosstalk and reduced SSN (simultaneous switching noise). As an output, DQS/DQS# simply consists of an inverted pair; as an input, the signal is evaluated differentially.

RDQS is useful in systems that employ a mix of nibble-wide (x4) and byte-wide (x8) SDRAMs. For this purpose, it is only useful and available on x8 devices. On the x8 device, DQS can be paired with the DQS signal for one nibble, and the RDQS can be paired with the DQS signal for the other nibble.

Burst Size and Ordering

The DDR/DDR2 SDRAM specification provides the burst size and type options shown in Table 10, and all options are available to users of the DDR/DDR2 SDRAM. By default, sequential 4-word bursts are selected. Any other option can be activated via a Load Mode Register command at any time after initialization is performed.

Table 10. Burst Mode Options

Burst Length	Starting Column Address (A2, A1, A0)	Order of Accesses Within a Burst	
		Sequential (Default)	Interleaved
4 (Default)	X00	0, 1, 2, 3	0, 1, 2, 3
	X01	1, 2, 3, 0	1, 0, 3, 2
	X10	2, 3, 0, 1	2, 3, 0, 1
	X11	3, 0, 1, 2	3, 2, 1, 0
8	000	0, 1, 2, 3, 4, 5, 6, 7	0, 1, 2, 3, 4, 5, 6, 7
	001	1, 2, 3, 0, 5, 6, 7, 4	1, 0, 3, 2, 5, 4, 7, 6
	010	2, 3, 0, 1, 6, 7, 4, 5	2, 3, 0, 1, 6, 7, 4, 5
	011	3, 0, 1, 2, 7, 4, 5, 6	3, 2, 1, 0, 7, 6, 5, 4
	100	4, 5, 6, 7, 0, 1, 2, 3	4, 5, 6, 7, 0, 1, 2, 3
	101	5, 6, 7, 4, 1, 2, 3, 0	5, 4, 7, 6, 1, 0, 3, 2
	110	6, 7, 4, 5, 2, 3, 0, 1	6, 7, 4, 5, 2, 3, 0, 1
	111	7, 4, 5, 6, 3, 0, 1, 2	7, 6, 5, 4, 3, 2, 1, 0

Load Mode Register
During Initialization, the Memory Interface performs an initial loading of the Mode and Extended mode Registers. The values loaded are the values programmed into the logic at design time. The Load Mode Register command allows the FPGA logic to reassign at any time any of the operational modes that are specified by the Mode register or Extended Mode Registers (refer to Tables Figure 3, Figure 4 and Figure 7).

Write and Read
The basic write and read operations are described below. The DDR/DDR2 SDRAM Memory Interface can support all DDR/DDR2 options, including 2/4/8-word burst, sequential/interleaved burst ordering, fast/slow exit power-down, ODT, output drive strength control, and DQS# and RDQS enable.

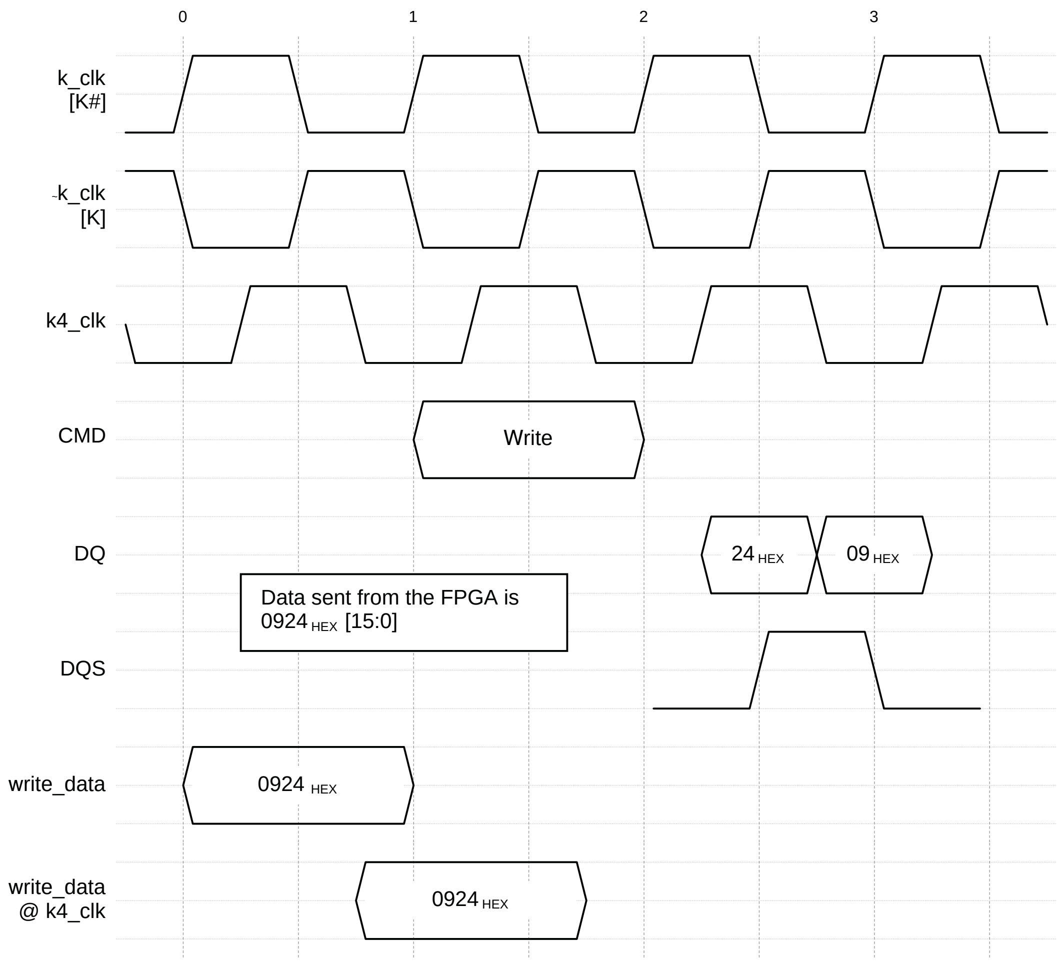
Write and Read with Auto Precharge
Each read or write access can also optionally perform an auto precharge operation to close the currently open row, either in the currently accessed bank or in all banks.

Whether or not to perform an auto precharge is a decision that depends on the characteristics of the system. Of course, if it is known whether or not the next access will be to the current row, that information can be used to decide whether to perform an auto precharge. If accesses are sequential, then the next access is usually to the same row, so it is best not to precharge. If accesses are of random or unknown order, then it is usually best to auto precharge after every access. Making a correct decision or “guess” will result in shorter access times and lower power consumption. However, if the incorrect choice is made, the Memory Interface will still correctly recover, either precharging the open row and activating the new row or re-activating the closed row as necessary, before the next access is attempted.

Write and Read Interrupt, with/without Auto Precharge (DDR2 Only)
When in 8-word burst mode, reads and writes are allowed to be interrupted when they are halfway complete by another instruction of the same type (i.e., reads by reads, writes by writes). The second (interrupting) command must specify the interrupt mode via its opcode, may specify auto precharge, and must occur exactly two clocks after the first (interrupted) command. The first command must not specify auto precharge.

Write Timing Analysis
The timing of a Write operation is shown in Figure 4. Note that during a write, DQS is center-aligned to DQ. This is accomplished by using a version of the system clock (K) to generate the DQS strobe at a 90º phase shift from DQ. A PLL can be used to generate the K clock using a 2x multiple of the ref_clk (or any other multiplication value that is appropriate based on customer need).

Figure 4. Write Data Timing Diagram



DQS Shut-Off (ClkCntrl)

It is important to gate the tristate-able DQS read strobe from the DDR/DDR2 memory for the following reason:

When neither the DDR SDRAM device nor the Memory Interface is driving DQS, DQS is in a high-impedance state. Both DQ and DQS are terminated by resistors that pull the voltage to the termination voltage (typically VCCIO/2). According to the JEDEC JEDS 8-9 specification for the SSTL-2 I/O standard and the HSTL-I standard, this is an indeterminate logic level, neither a logic high nor logic low. During read postamble, any noise on the DQS line can cause invalid data to be registered, thereby corrupting the read data to the user interface.

The DQS shut off logic in the Slayer PIO, called the CLKCNTRL block, is used to turn off DQS during read postamble, and is shown in Figures 8 and 9. In this application, the Q output is tied back to the CLK input. When the control input CE ='1', the internal register is "off", which enables the output, allowing CLK/Q to follow DQS on D. When the control input (CE) is brought to '0', the internal register is synchronously turned "on" at the next falling edge of CLK/Q, causing the stream of pulses on CLK/Q to cleanly stop on '0'. When the control input CE is returned to a '1', the internal register asynchronously clears, and CLK/Q immediately begins to again follow DQS on D.

The use of this block is shown in Figure 5.

Figure 5. The CLKCNTRL Block

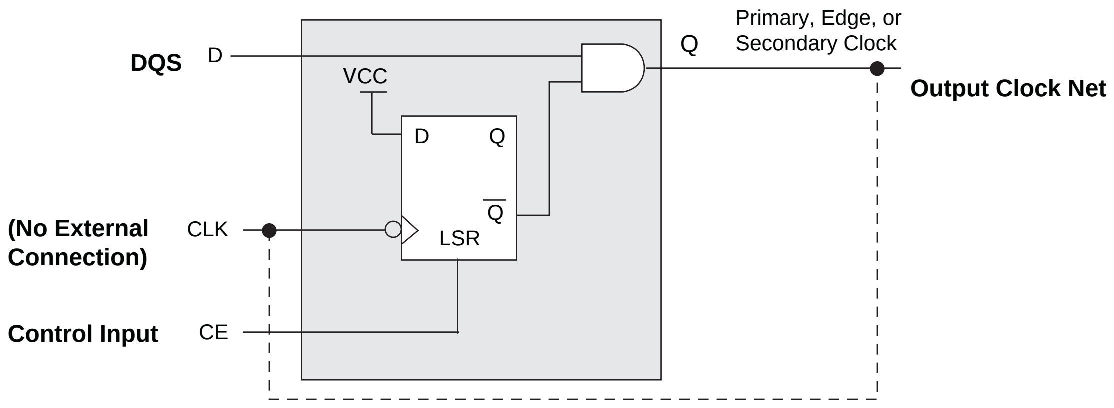
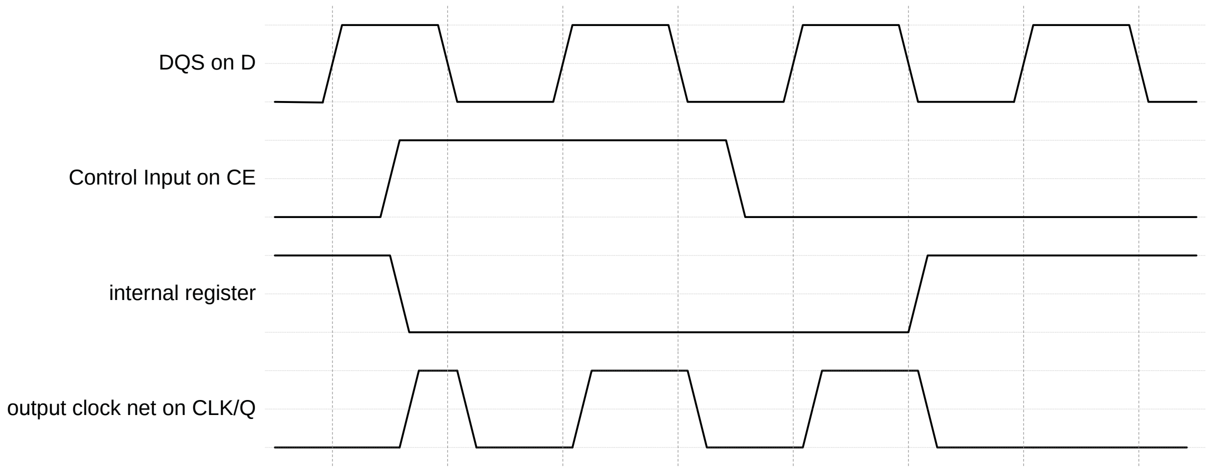


Figure 6. CLKCNTRL Timing



Preamble Edge Detection (ClkDetect)

Every I/O pin in the LatticeSC/M architecture also supports a one-shot edge detection circuit. This circuit is typically placed directly after the DQS shut-off circuitry and is used to detect the first rising edge of a Read Preamble. It is reset before a read begins and thus provides information as to the timing of the preamble, regardless of the board trace length to/from the memory device. The use of this block is shown in Figure 1.

Read Timing Analysis

The Memory Interface core uses the K clock for internal timing, as well as sending it and its complement K# to the DDR/DDR2 SDRAM memory device. This core clock K is fed into DLL1 to produce a T/4 digital control output bus. This 9-bit bus is used to control the INDEL delay cells within the PIOs used for DQS/DQS# read inputs and will provide a 90º time shift for the DQS/DQS# input signals.

As shown in Figures 5 and 6, the read data DQ is captured on the rising or falling edge of the data strobe DQS (DQS# is not shown). Since DQ and DQS are edge-aligned coming from the SDRAM device, DQS needs to be delayed (ideally, centered to DQ) to effectively capture the data. Methods such as using "cycle boosting" delays or pre-setting the INDEL to a given value can be used to delay the DQS with respect to the data, but using the DLL as shown in Figure 1 to control the INDEL to delay the DQS signals by 90º gives the greatest timing margin over variations in process, voltage and temperature (PVT), and is independent of the interface speed. INDEL on the DQ pins can be set to a specific value, since edge-clock injection delays are known and fixed over the devices' specified operating range; a fixed INDEL setting on the DQ inputs will be used to match the captured DQ data to the edge clock injection delay for DQS.

The read data timing diagram in Figure 7 shows the read data captured using DQS at the FPGA I/O, and the relationship between k_clk, k4_clk, k2_clk and k3_clk. It also shows an example for the number of K clock cycles of latency after which read data is available to the FPGA. The data_valid signal generated in the soft IP indicates the start of the read data burst. This is generated by sampling the first rising edge of DQS using the edge detect (Clk-Detect) capability built into the LatticeSC/M PIOs. The naming conventions used in Figure 7 should be used only as a reference.

Figure 7. Read Data DQ Timing at Output of SDRAM Device

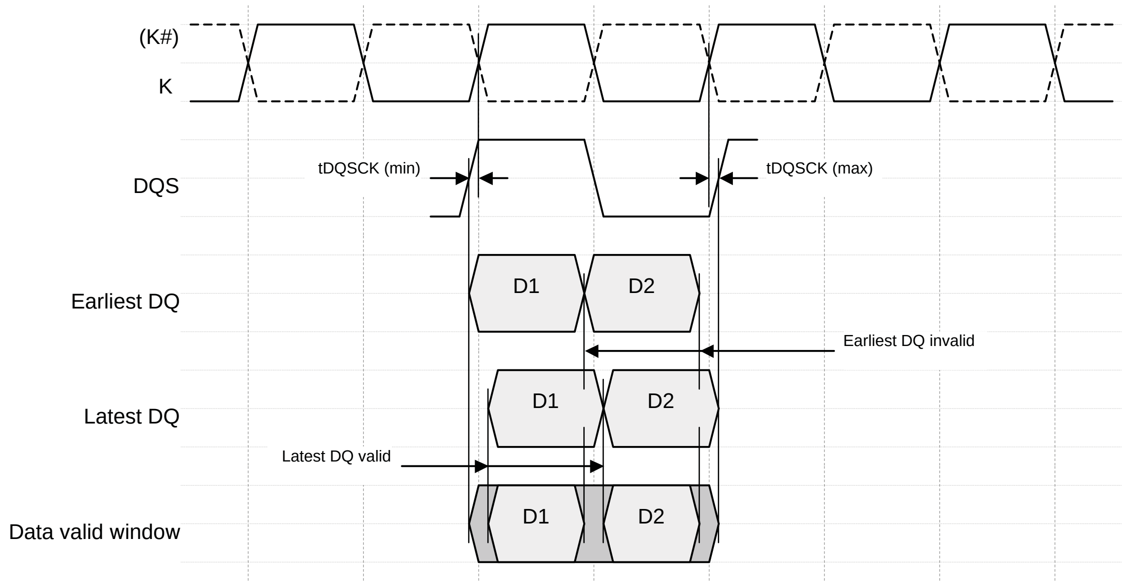


Figure 8. Read Data Capture at SDRAM Memory Interface

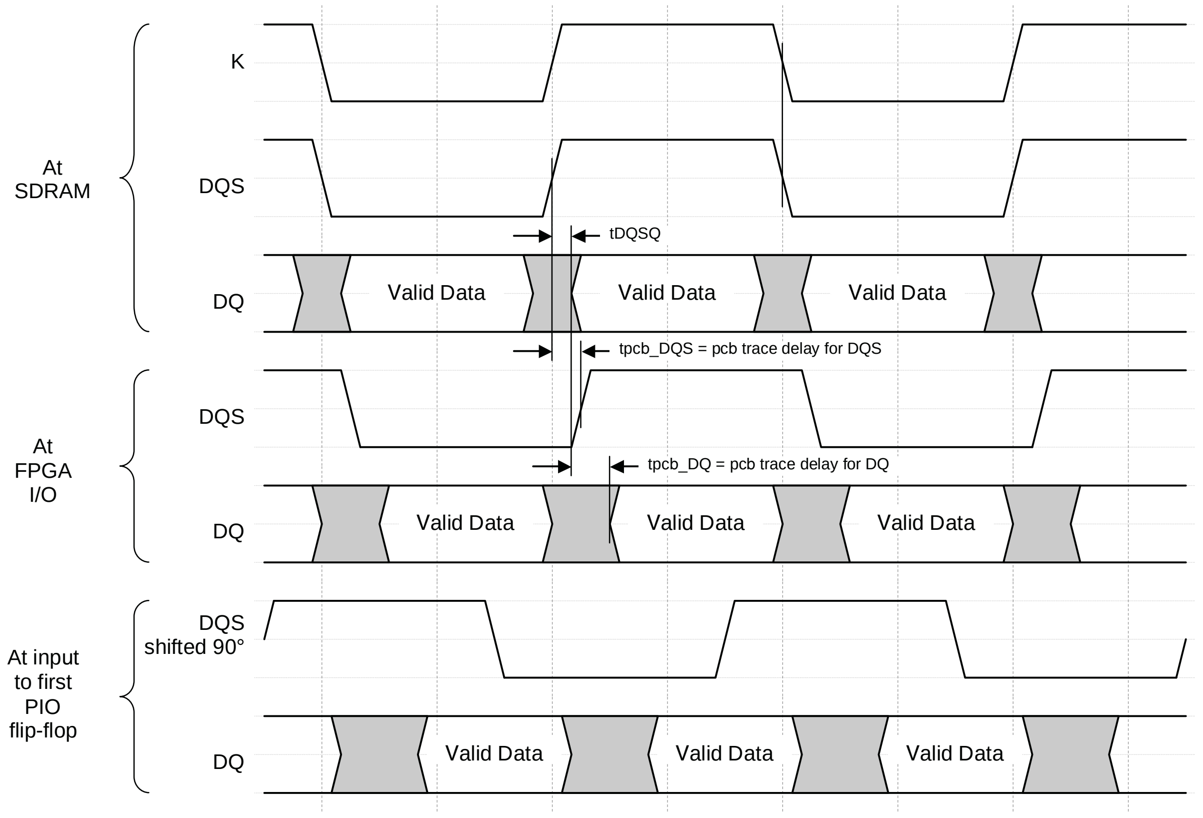
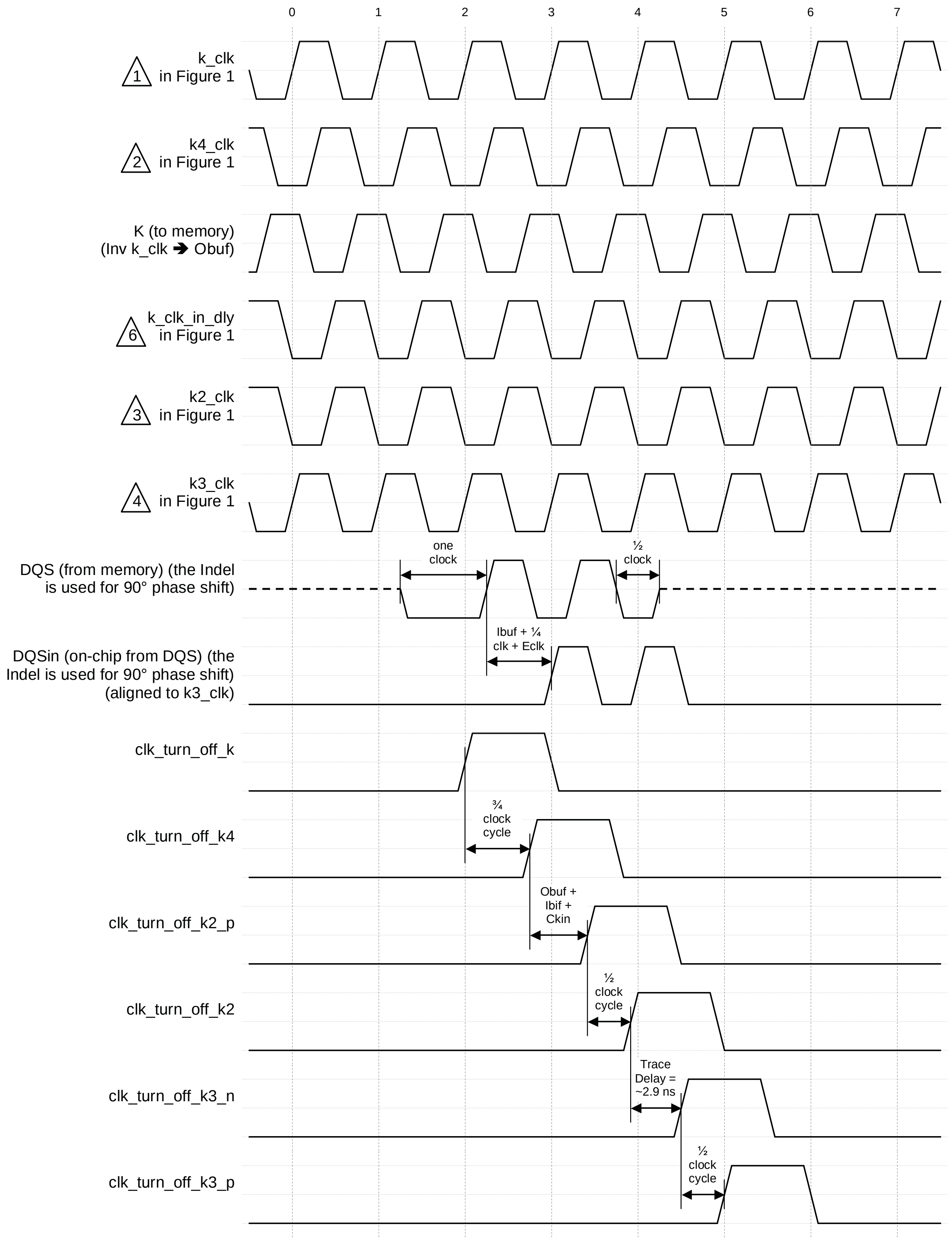


Figure 9. Read Data Timing Diagram



Self Refresh and Refresh

The Self Refresh command causes the DDR/DDR2 SDRAM to enter a data-retaining quiescent state; all inputs except CKE can (but don't) float. The next command that is issued will return the SDRAM to its active state, but a period of 200 clocks (t_{XSRD}) will transpire before that instruction is allowed to execute, in order to allow the SDRAM's DLL to stabilize.

Self refresh is under control of the FPGA user logic, and automatic refresh is performed transparently by the Memory Interface.

Power-down

A power-down command is provided to place the DDR/DDR2 SDRAM in a low-power quiescent state. The DDR/DDR2 SDRAM supports two power-down modes: Fast-Exit and Slow-Exit. Fast-Exit mode, although having a faster recovery, consumes more power. By default, Fast-Exit is selected. Slow-Exit can be activated via a Load Mode Register command at any time after initialization is performed. Even during power-down, the Memory Interface logic will ensure that refreshes are performed as necessary.

Reducing PLL/DLL Usage

In many cases, two or more Memory Interfaces are used on the same device. In these cases, it is possible to reduce the total number of PLLs/DLLs by sharing their functions among the several Memory Interfaces. The following techniques may be employed, and are best implemented in the following order of priority:

1. Reduce the PLLs that are used to multiply the ref_clk to the K clock frequency. Typically, this function can be shared between the two Memory Interfaces, if they are running at the same speed. Sometimes, no PLLs are specifically required for this function, since the needed clock rate is already created elsewhere in the system.
2. Share one DLL to generate the 90º phase shift for the DQS input. This is only possible if both Memory Interfaces are running at the same speed. This is important because only one bus is needed to route to all Memory Interfaces. Due to PVT conditions, it is important to use a DLL for this function, but it is permissible to share one between two Memory Interfaces. Note that there will be less tuning range around 90º available to improve the overall performance, since optimization will be performed for both memory interfaces simultaneously.

LatticeSC/M I/O Buffers

DDR/DDR2 SDRAM inputs and outputs conform to the JEDEC standards SSTL_25 (DDR1) and SSTL_18 (DDR2). LatticeSC/M I/Os can be configured to conform to these standards, among may others.

Additionally, DDR/DDR2 requires DDR I/O operation, and the LatticeSC/M family of devices provides the hardware to support DDR input and output in every I/O pin. This means that data transfer within the device is at SDR (single data rate) speeds, so that the internal FPGA fabric can easily handle the bandwidth.

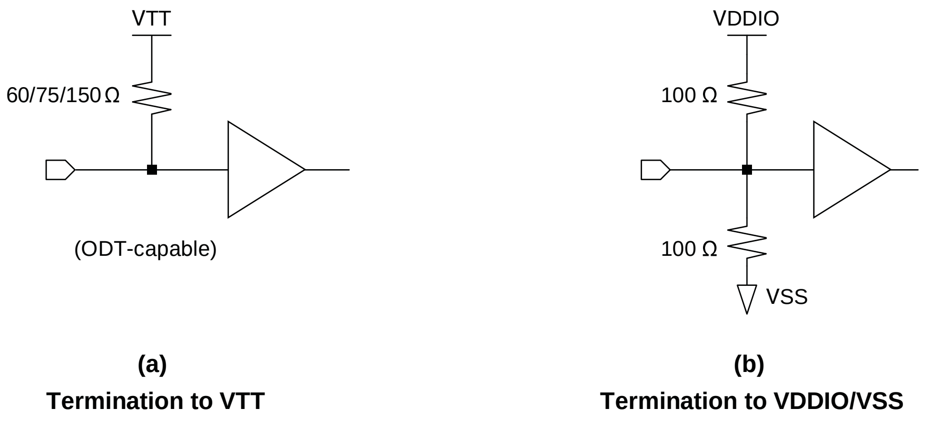
A third feature that is required for optimum DDR2 performance is On-Die Termination (ODT). LatticeSC/M devices offer ODT on a per-pin basis. Two termination configurations are available (Figure 10), and ODT is available in configuration (a). DDR1 SDRAMS do not offer ODT as a feature, but the LatticeSC/M I/O ODT capability can still be utilized on the FPGA end of the signal.

The configurations shown in Figure 10 are:

- Termination directly to VTT via a 60/75/150Ω impedance.
- Termination via a Thevenin-equivalent 50Ω network across VDDIO and VSS.

LatticeSC/M output buffers also offer features important in high-speed designs. Each output buffer includes an optional 25/33/50Ω serial resistor, as specified in some high-speed signaling standards. This serial resistor matches output impedance to board trace impedance, referred to as source-termination. In addition to other advantages such as reducing edge rates and the noise they produce, source-terminated signals can properly drive single loads that are unterminated (stubbed).

Figure 10. LatticeSC/M Input Buffer Configurations



Clocking Challenges and Solutions

Figure 11 illustrates the clocking network. Several unique features of the LatticeSC/M architecture are utilized in this design. A PLL ① is used to perform frequency multiplication of the input clock ref_clk to produce clock k_clk, and at the same time to generate a second clock k4_clk that is 90º lagging, so that the clocks K and K# to the DDR2 SDRAM can transition in the center of the data eye of bus DQ. Both k_clk and k4_clk are typically routed on primary clock nets so that there is very little skew from the ideal 90º offset.

Two DDR I/O registers ② function as a simple buffer (inverter) of the input clock k_clk to produce output signal K (K#). They do this by clocking in a HI (LO) value on the clock's rising edge, and a LO (HI) value on the clock's falling edge. Since the same register and clock are used as are used to transmit RAS#, CAS#, WE, A, BA, CS and ODT, clocks K and K# will be matched in phase with these signals. The DQS, and DQS 3-state enable signals ④ also use a similar clocking structure and have the same timing parameters as the K clock signals when enabled as outputs.

A DLL ③ is used to dynamically generate a delay value that is used in the input buffer ④ to phase-shift by 90º the DQS input. The phase shift places the clocks' transition edges in the center of the data that they capture for DQS and keeps DQS aligned (not counting board routing delay and SDRAM DLL uncertainty, for which compensation will be applied later). This takes advantage of the fact that the DLL and the input buffers contain matching delay blocks, so that the delay selection value generated in the DLL when it generates a 90º shifted clock can also be used in any input buffers to cause the same phase shift.

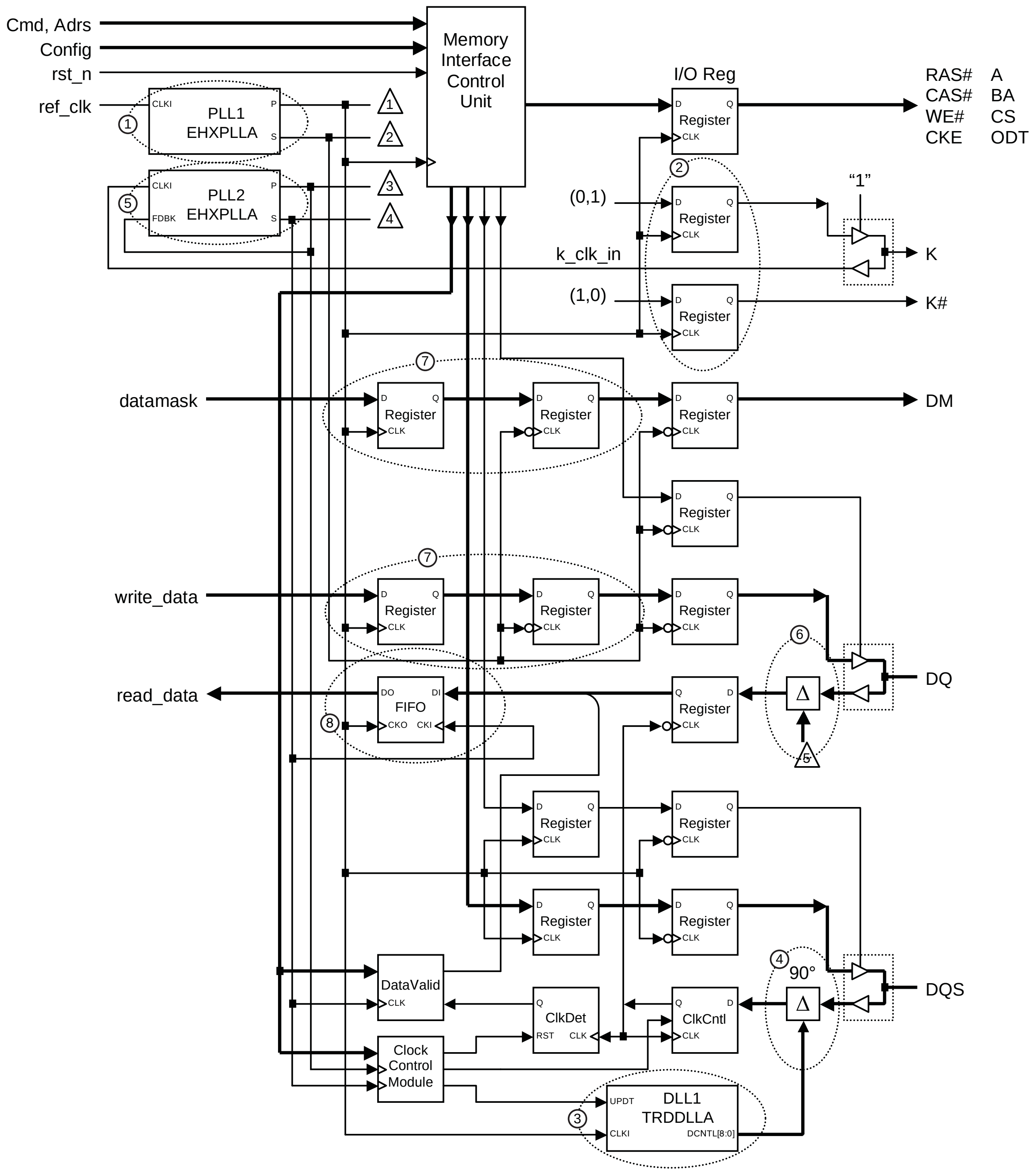
A second PLL ⑤ is used to generate two more internal clocks, k2_clk and k3_clk. The first clock k2_clk is generated from the logic that generates K, so that the delays due to the output buffer, input buffer and the 90º phase shift on the DQS clock input are nullified. Signal k3_clk contains an additional programmed delay, calculated to compensate for the sum of all off-chip delays (such as board trace delay and SDRAM DLL phase match uncertainty) and thus to transfer read data from the DQS input clock back to the continuous k3_clk.

For the DQ data bus, the delay elements ⑥ are used in an "Edge Clock Injection Match" mode. This compensates for the edge clock routing of the DQS input, thereby providing an optimal read data edge. Manual changes to the input delay can also be made to each individual Q input pin to compensate for differences in board trace delays.

The write data and corresponding data mask are sent through a pair of registers ⑦ to effect a clock domain transfer. The first rank is clocked by k_clk, presumably the same clock as used in the FPGA fabric. The second rank is clocked by k4_clk, which shifts the output by 90º so that the clock is centered in the data eye.

The read data undergoes a clock domain change via FIFO ⑧. The read data is clocked in by k3_clk, and clocked out by k_clk.

Figure 11. DDR2 Memory Interface Block Diagram – Clocking



Notes: 1 k_clk (ref_clk X 2)
2 k4_clk (k_clk + 90°)
3 k2_clk (match DQS – trace delay) + 90°
4 k3_clk (match DQS) + 90°
5 Delay for DQS edge clock injection matching (Delay = 13 for SC25, other devices TBD)

Performance

Timing Budgets

The timing on the signals interconnecting the LatticeSC/M Memory Interface and the DDR2 SDRAM can be difficult to manage, since there are several unusual components to the timing budget. These signals fall into two groups: the signals going to the SDRAM under clocking by clock K/K#, and the return signals from the SDRAM under clocking by DQS.

All LatticeSC/M timing specifications apply to all speed grades over process/voltage/ temperature variation. The values are subject to change. See the LatticeSC/M Family Data Sheet for the latest values.

The first group of signals is depicted in Figure 12, and the timing budget is calculated in Table 11. There are eight components to the timing budget: 1 clock period, 2 duty cycle distortion (DCD), 3 PLL cycle-to-cycle jitter, 4 PLL phase error skew, 5 primary clock skew, 6 package skew, 7 board trace skew, and 8 SDRAM receive register input setup/hold.

Parameters marked a are LatticeSC/M device parameters and are valid over all the device's specified operating conditions and speed grades, b is a characteristic of the PC board, and c are DDR2 SDRAM device parameters. Values given for b and c are typical and must be adjusted for the specific PCB and DDR2 SDRAM chosen.

DCD and PLL phase error skew are both significant because they both tend to displace the clocks in the eye, and because of the fact that both edges of the clock are critical for capturing data.

The second group of signals is depicted in Figure 13, and the timing budget is calculated in Table 12 (DDR2 SDRAM DLL enabled). There are nine components to the timing budget: 1 clock period, 2 DDR2 SDRAM duty cycle distortion (DCD), 3 DDR2 SDRAM Clock Output Skew, 4 package skew, 5 board trace skew, 6 edge clock skew, 7 LatticeSC/M receive register input setup/hold, 8 the SDRAM output skew, and 9 the quantization error of the DLL. Item 9 represents the smallest step of delay change in the DLL internal programmable delay.

Parameters marked with an "a" are LatticeSC/M device parameters and are valid over all specified operating conditions and speed grades for the device, "b" is a characteristic of the PC board, and "c" are DDR2 SDRAM device parameters. Values given for b and c are typical and must be adjusted for the specific PCB and DDR2 SDRAM chosen.

Figure 12. Timing Path, DDR Write

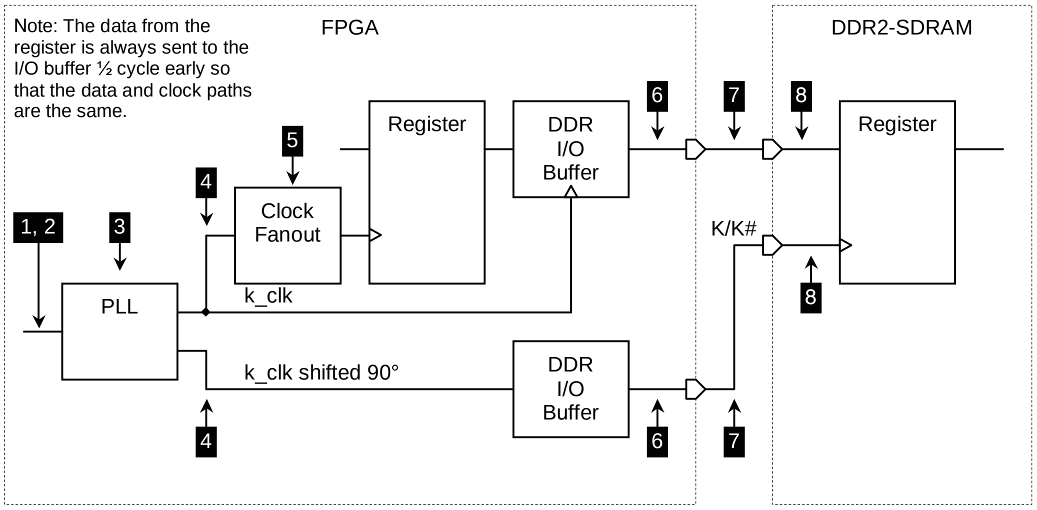

Note: The data from the register is always sent to the I/O buffer ½ cycle early so that the data and clock paths are the same.
FPGA
DDR2-SDRAM
Register
DDR I/O Buffer
Register
Clock Fanout
PLL
k_clk
k_clk shifted 90°
DDR I/O Buffer
K/K#
1, 2
3
4
5
6
7
8
8
4
6
7

Table 11. Timing Budget - DDR Write Data Path (Preliminary)

Parameter		Units	Value	Long Path	Short Path		Description
f_{CLK}		MHz	266				Clock Frequency
t_{CLK}		psec	3750				Clock Period
$t_{CLK-DDR}$	1	psec	1875				DDR Clock Period (1/2 * t_{CLK})
t_{DCD}	2	psec	±95	+95	-95	a	Clock Duty Cycle Distortion (45%-55%)
t_{CTC}	3	psec	±70	+70	-70	a	PLL Cycle-to-Cycle Jitter
t_{PHASE}	4	psec	±70	+70	-70	a	PLL Phase Error Skew
t_{CS}	5	psec	±50	+50	-50	a	Primary Clock Skew (DQS placed near DQ)
t_{PKG}	6	psec	±40	+40	-40	a	LatticeSC/M Package Skew
t_{BOARD}	7	psec	±80	+80	-80	b	Board Trace Skew (K/K# vs. DQ)
t_{SD}	8	psec	+450	+450		c	DDR2 SDRAM Input Setup
t_{HD}	8	psec	-450		-450	c	DDR2 SDRAM Input Hold
Total			+855	-855			Sum of All Components

Figure 13. Timing Path, DDR Read

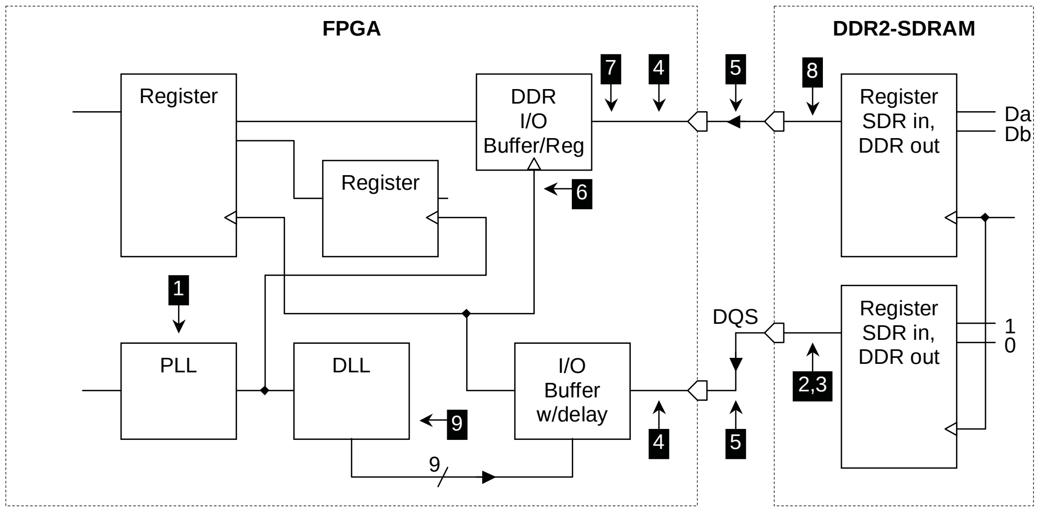


Table 12. Timing Budget - DDR Read Data Path (Preliminary)

Parameter		Units	Value	Long Path	Short Path		Description
f_{CLK}		MHz	266				Clock Frequency
t_{CLK}		psec	3750				Clock Period
$t_{CLK\text{-}DDR}$	1	psec	1875				DDR Clock Period (1/2 * t_{CLK})
t_{DCD}	2	psec	±95	+95	-95	b	Clock Duty Cycle Distortion of DDR2 SDRAM (45%-55%)
T_{CQHQV}	3	psec	+350	+350		b	DDR2 SDRAM DQS High to DQ Valid
T_{CQHQX}	3	psec	-350		-350	b	DDR2 SDRAM DQS High to DQ Invalid
t_{PKG}	4	psec	±40	+40	-40	a	LatticeSC/M Package Skew
t_{BOARD}	5	psec	±80	+80	-80	b	Board Trace Skew
t_{CS}	6	psec	+40	+40		a	Edge Clock Skew
t_{SD}	7	psec	+363	+363		a	LatticeSC/M Input Setup
t_{SD}	7	psec	-279		+279	a	LatticeSC/M Input Hold
$t_{RAMOUNT}$	8	psec	±40	+40	-40	c	DDR2 SDRAM Output Skew
T_{DQE}	9	psec	±70	+70	-70	a	DLL Delay Quantization Error
Total			+1078	-396			Sum of all Components

"Eye" size = (1875 ps) - (1088 ps) - (396 ps) = 391 ps

Usage Overview

The following subsections provide information on usage-related topics.

IPexpress and I/O Assistant

The LatticeSC/M family is supported by ispLEVER®, a suite of tools that facilitates the implementation of complex designs such as those that employ a DDR/DDR2 SDRAM Memory Interface.

ispLEVER provides an integrated tool environment for the entire design flow for the LatticeSC/M. Complex designs contain various Intellectual Property (IP) blocks and/or architecture-specific modules. A key element of a complex design flow is module generation. To facilitate module generation for LatticeSC/M, ispLEVER includes a tool called IPexpress.

IPexpress is a graphically-driven utility that accepts user specifications for pre-optimized logic blocks, and generates all the detailed instructions for synthesizing and simulating that logic. IPexpress provides the designer with the top-level instantiation of the module that the designer can then integrate into the design. This allows the designer to concentrate on the architectural issues of the design while ispLEVER software handles the details.

The LatticeSC/M DDR SDRAM Memory Interface is one of the logic units that IPexpress can generate. Table 13 lists the data typically provided to the IPexpress for instantiation of a DDR/DDR2 SDRAM Memory Interface. This data is provided via a Graphical User Interface (GUI), and the values are programmed into the FPGA design. The output of IPexpress is the top-level netlist for the Memory Interface, and includes all clocking, DQS logic, I/O placement, etc.

I/O Assistant is another utility in ispLEVER. It allows the designer to specify physical design restraints such as pin placement and I/O bank assignment, and to do so early in the design cycle, even before the detailed FPGA design is complete. This is necessary in complex designs where these allocations can have significant impact on the design flow.

The LatticeSC/M family provides a comprehensive solution to the designer's requirements. The ispLEVER design flow, together with IPexpress and I/O Assistant tools to simplify integration of Lattice IP into the design, provide the designer with a complete end-to-end solution. The design will go from conception to tested product in a fraction of the time.

Table 13. Parameters Provided to IPexpress for DDR/DDR2 SDRAM Memory Interface (Typical)

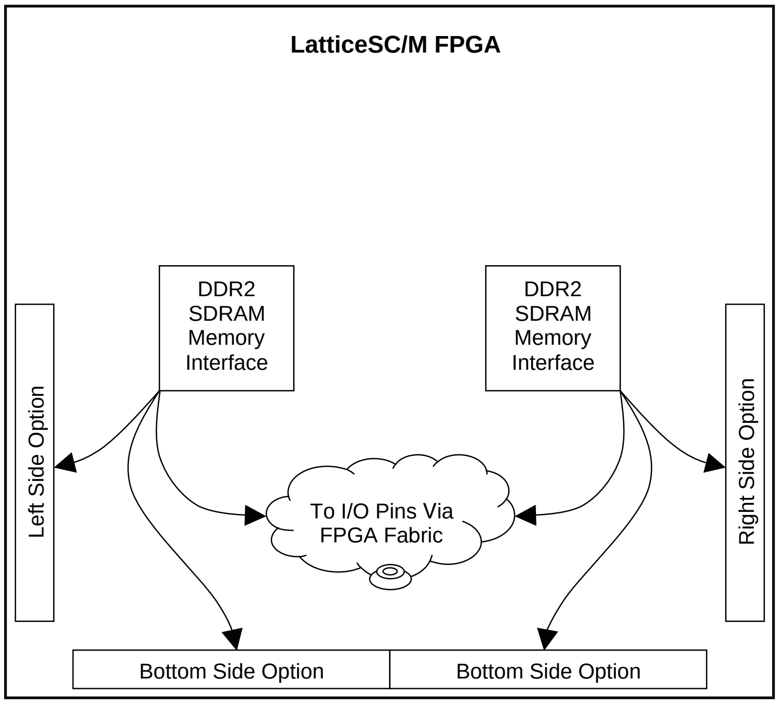
Parameter	Permissible Values	Default Value
DATA_WIDTH	8, 16, 24, 32, 40, 48, 56, 64, 72	32
ROW_WIDTH	Up to 14	13
COL_WIDTH	Up to 13	9
TRAS	‡	40 ns
TRC	‡	60 ns
TRCD	‡	15 ns
TRRD	‡	10 ns
TRFC	‡	75 ns
TRP	‡	15 ns
TMRD	1-7	2 clocks
TWR	‡	15 ns
TREFI	‡	15.625 µs
Number of Auto Refresh Burst Commands	1 - 8	2
Page Size	256, 512, 1024, 2048, 4096, 8192	256 columns
DDR1 Only		
CS_WIDTH	1, 2, 4, 8	1
Enable Ext Base Mode Write	0, 1	1
DDR2 Only		
CS_WIDTH	1, 2	1
No. of DIMM Slots	1, 2	1
INT_BANK	4, 8	4
TRTP	‡	10 ns
TWTR	‡	10 ns
TFAW	‡	40 ns
TCKP	‡	400 ns

Note: ‡ Value is determined by the specific DDR/DDR2 SDRAM device utilized (see spec.).

Pinout Selection and Layout Constraint

In order to achieve maximum operating speed, the DDR/DDR2 protocol requires carefully designed I/O timing on the interface between the LatticeSC/M device and the DDR/DDR2 SDRAM. For this reason, Lattice-supplied modules can provide pre-optimized pinout assignments, as well as assignments for critical internal components such as PLLs and DLLs (a DDR/DDR2 Memory Interface design typically requires two PLLs and one DLL). Where possible, optional alternatives are provided so that the Memory Interface can co-exist with various combinations of other IP. One option places the Memory Interface’s interface pins directly adjacent to the unit’s internal logic on the left or right side of the chip (see Figure 14). A second option places the interface pins on the bottom of the device. This can be useful when other constraints cause the side pins to be unavailable. There is the third option of not utilizing pre-assigned pinouts, but then the timing is not pre-optimized.

Figure 14. DDR2 SDRAM Memory Interface Pinout Options (Typical)



In order to achieve minimum skew between signals, all signals in a bus should be assigned pins within the same I/O bank. If this is not possible, use pins in two banks adjacent to the same corner of the FPGA. These two banks in the LatticeSC device have low skew when driven by a PLL or DLL in that corner. Also bear in mind that since read interface bus clocks typically use edge clock routing, the I/O registers associated with the DQ bus (DQS, DQS# and RDQS clocks) must be connectable by the same edge clock.

It is a good design practice to explicitly assign the input clocks, but allow the associated data signals to be assigned by the placement tool. Clocks frequently drive directly into specific internal elements such as a PLL or DLL. These elements have a dedicated corresponding I/O pin that, when used, provides optimal performance, as shown in Tables 14 and 15. The I/O Assistant application facilitates the process of clock/data pin assignment. Using I/O Assistant, the designer can easily assign resources (pins, PLLs, DLLs, etc.) with whatever specificity is appropriate (pin, I/O bank, etc.), even before the design is complete.

Design Guidelines to Optimize Performance

Master Clock

The master reference clock can be sourced from any clock source, either internal or external to the LatticeSC/M device. It is fed to PLL PLL1 in Figure 11. If the source is external, it should use the direct input pin for that PLL CLKI input (refer to Table 14). Also, minimize clock jitter caused by coupling from noisy neighboring signals (refer to the accompanying discussion, "Selecting a Pin That Has Low Jitter Noise" below). Note that the PLL will filter some of the jitter that exists at the PLL's input.

Table 14. PLL Direct Input Pins (True/Complement Pair)

	F900	**FF1020**	**FC1152**	**FC1704**
ULC PLL A	D3/D2	K25/J25	F30/G30	J37/J38
ULC PLL B	K4/J4	M23/N23	N25/P25	N33/P33
LLC PLL B	AC6/AC7	AC23/AD24	AG29/AG28	AN36/AP36
LLC PLL A	AH1/AJ1	AJ32/AK32	AM33/AN33	AU42/AV42
LRC PLL A	AJ30/AH30	AJ1/AK1	AN2/AM2	AV1/AU1
LRC PLL B	AD26/AC25	AC10/AD9	AG6/AG7	AN7/AP7
URC PLL B	K25/K26	M10/N10	N10/P10	N10/P10
URC PLL A	D28/E28	K8/J8	F5/G5	J6/J5

Table 15. DLL Direct Input Pins (True/Complement Pair)

	F900	**FF1020**	**FC1152**	**FC1704**
ULC DLL C	E3/E2	D32/D31	F31/G31	G40/H40
ULC DLL D	F3/G3	E32/E31	D33/E33	G41/H41
LLC DLL E	AB6/AC5	AE26/AE27	AJ30/AK30	AL37/AM37
LLC DLL F	AF2/AG2	AG32/AG31	AL32/AL31	AR39/AR40
LLC DLL C	AF4/AE5	AF27/AG28	AH29/AJ29	AL33/AL34
LLC DLL D	AG3/AH2	AK31/AL31	AM32/AM31	AU38/AV38
LRC DLL C	AJ29/AH29	AL2/AK2	AM3/AM4	AV2/AW2
LRC DLL D	AG28/AG29	AJ2/AH3	AJ6/AH6	AL10/AL9
LRC DLL F	AF29/AF28	AG1/AG2	AL3/AL4	AR4/AR3
LRC DLL E	AB26/AC26	AE7/AE6	AJ5/AK5	AL6/AM6
URC DLL D	G28/F28	E1/E2	D2/E2	G2/H2
URC DLL C	D29/D30	D1/D2	F4/G4	G3/H3

Note that if there are multiple DDR2 Memory Controllers on the same LatticeSC/M device that operate at the same rate, they can share a common PLL PLL1, in which case the two nets k_clk and k4_clk will also be common among them. This is accomplished by wrapping each memory controller in its own module that contains all logic except PLL1, connecting each module's internal nets k_clk and k4_clk to module inputs, instantiating one copy of PLL1 outside the DDR2 modules, and connecting that PLL's outputs to those inputs on each module.

DQS Strobe

The DQS strobe and its associated DQ and DM signals must all reside in banks served by a common edge clock. The DQS strobe I/O pin must be assigned to a preferred pin, that is, a pin that can directly drive its associated edge clock. There is a single edge clock serving the two banks (2-3 or 6-7) on each of the two sides of the device, right and left. The two banks (4-5) on the bottom edge are each served by a separate edge clock.

The bidirectional DQS strobe to/from the DDR2 device can be implemented as either a single-ended or differential signal.

For differential DQS implementations, the differential pairs must be on A/B pairs (for example, PB17A/PB17B) that are semi-dedicated "PCLKT/C[7:2]_[7:0]" (for example, PCLKT5_2/PCLKC5_2). These pairs feature complementary outputs and differential inputs, and are able to directly drive edge clocks. If IPexpress™ is used to generate the DDR2 memory Controller, the pins assigned will conform to this requirement.

Differential DQS applications also require that the IOBUF preference for the DQS signals have the IO_TYPE changed from SSTL18_II to SSTL18D_II. If the design is generated by IPexpress, this is handled automatically.

K/K# Clocks

The K/K# clock pair (and K_copy/K_copy# pair, if used) must be placed on A/B pairs, since they form a complementary output pair.

In order to minimize skew and noise, the K/K# clock pair must be located on an input pin that drives an edge clock.

It is important to minimize clock jitter caused by coupling from noisy neighboring signals (refer to the accompanying discussion, "Selecting a Pin That Has Low Jitter Noise", below).

PLLs

PLL1 in Figure 11 generates a 90° phase shift between the address/data/control lines to the DDR device and the accompanying K/K# clock. PLL1 also performs optional clock frequency multiplication when necessary. No custom adjustment of PLL1 is needed.

PLL2 in Figure 11 performs a 90° phase shift on k_clk_in, the internally reflected copy of the outbound clock. PLL2 also compensates for the total round-trip delay of the board traces to/from the memory device. The value of this delay is entered into IPexpress as the "DQS Trace Delay Compensation" when the DDR module is generated. In order to achieve optimum performance, especially at high clock rates, this delay value can be tuned for the specific implementation. This tuning need only be performed once, and should be performed using the final board layout. The simplest way to perform this tuning is by iteratively changing the DQS Trace Delay Compensation in IPexpress to determine the range of values that yield correct performance, and then using the "sweet spot" centered in that range. Alternatively, the PLL2 behavior can be modified dynamically by writing the relevant parameters, PHASEADJ and CLKOS_VCODEL, via the System Bus. For details on modifying these parameters, refer to TN1098, *LatticeSC sysCLOCK and PLL-DLL User's Guide*. Note that CLKOS_VCODEL is applied at PLL reset, so a reset must be applied after each change.

DLLs

The DLL (DLL1 in Figure 11) is used to generate a 90° phase shift so that the receive DQ data eye is centered on the receiving DQS clock. It uses TRDDLLA (time reference delay) mode to achieve this result.

To achieve maximum performance, it may be necessary to adjust the DLL's ALU function +/- 1 or more taps in order to center the DQS in the DQ eye. The optimum setting should be determined experimentally, using the final board layout. The parameter to adjust is named DCNTL_ADJVAL, and can be set in the DLL's source code file "ddr_trdll.v". Do this by adding two lines similar to the following, which set DCNTL_ADJVAL to -2:

```
    /* synthesis DCNTL_ADJVAL="-2" */

    // exemplar attribute ddr_trdll_0_0 DCNTL_ADJVAL -2
```

Add each of the lines to the group of similar lines in the code. The value can also be modified dynamically by writing it via the LatticeSC/M SMI Bus, if the DLL has been assigned a unique SMI address (DCNTL_ADJVAL is byte 9). Also, if the device has the ORCAstra interface module implemented in the design, the value can be modified via the ORCAstra GUI.

Since this DLL drives the DCNTL bus, it should be located in the corner of the device adjacent to the DQS signal banks. Note that the clock to the DLL is driven internally, so there is no input pin to be located.

Board Layout and Trace Matching

All DQ, DQS and DM signals within a lane must have PCB board trace lengths matched to within 50 picoseconds, and across lanes, they must be matched to within 150 picoseconds.

All address, control signals, and their clocks (K, K#, K_copy and K_copy#) to the DDR2 device must be matched to within 50 picoseconds.

Lattice recommends simulation of simultaneous switching outputs (SSOs) for the device/package combination for performance targeted to over 200 MHz.

In order to ensure that potential conflicts are resolved and to provide maximum flexibility when assigning resources, Lattice recommends that the LatticeSC/M device design be placed and routed in ispLever before commitment of the board design to manufacture.

Other Board-Level Considerations

All dynamic signal traces must be 50 Ohm transmission lines.

All power signals, including any VTT power, must be supplied by planes, not traces.

Care must be taken to keep reference voltages, such as the DDR2 VREF pin, noise-free.This involves robust, wide-bandwidth decoupling, and isolation of quiet, noise-sensitive signals from noise sources.

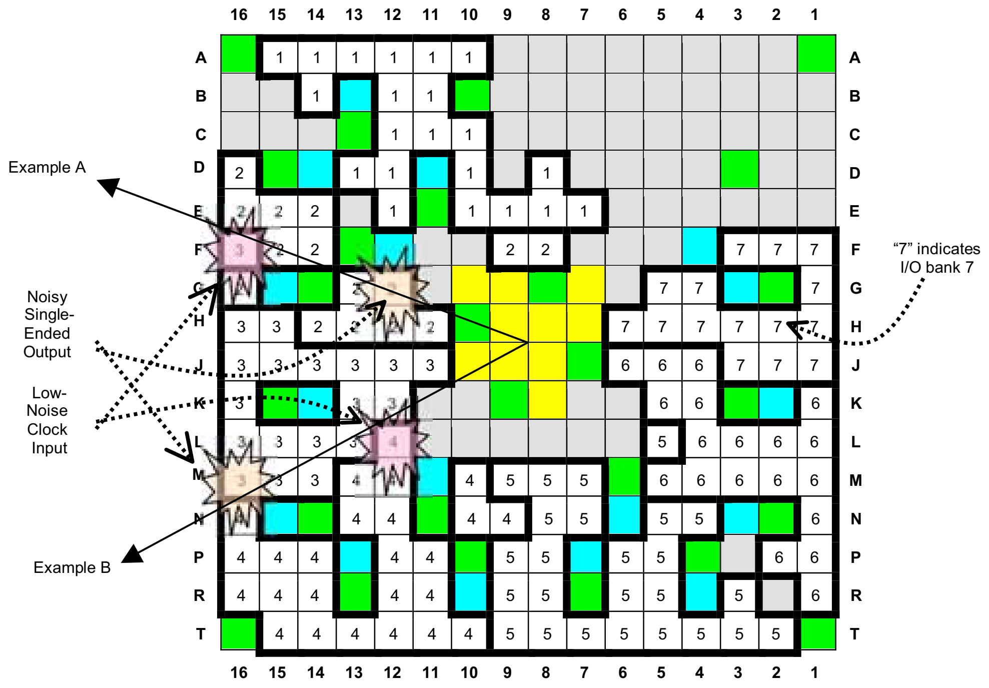
The physical distance between the LatticeSC/M device and the DDR2 memory device needs to be minimized, since trace delays, skews and signal degradation will limit overall speed.

Selecting a Pin That Has Low Jitter Noise

When a signal, such as an input reference clock or the DDR2 clocks K/K# or DQS, needs to be especially quiet with low-jitter, some special design rules can help achieve this goal:

- It is highly preferable to place the pin in a bank that does not also contain single-ended output drivers. Figure 15 shows how bank groups form clusters around the package, in this case for a 256-pin fpBGA.

Figure 15. Selecting a Pin for Low Jitter Noise



- If a quiet bank cannot be used, avoid creating inductively coupled paths linked to noisy signals on the package. These occur when the low-noise signal trace passes through an area on the package substrate from pin to pad that contains noisy signal pins or traces (in particular single-ended outputs, and especially when those single-ended outputs are unterminated). Figure 15 also illustrates this concept. Two examples are shown:
  - Example A shows a noisy output pin (G12, bank 2) that is near the package center, and a low-noise clock pin (F16, bank 3) that is situated radially outward from that pin. In this case, the pin-to-die connection for the clock will route directly past the noisy output pin, resulting in coupled noise. This should be avoided.
  - Example B demonstrates the reverse situation, which is also to be avoided. In this case, a noisy output pin (M16, bank 3) is situated radially outward from a low-noise clock pin (L12, bank 4), so that the noisy output's pad-to-pin connection will pass over the clock pin.
  - In order to minimize this coupling, it is typically better to place noise-sensitive pins toward the center of the package. This reduces the trace length of this signal in the package, thus reducing coupling to this signal.

Noise immunity may be further enhanced by providing extra ground pins around the sensitive signal, by driving adjacent outputs to a constant LO and tying them to signal ground on the PCB. This can enhance noise immunity in two ways: first, it provides extra signal current return paths, and second, it provides a buffer distance to nearby signal pins, thus reducing coupling to their signals. The buffers should be set to the maximum drive strength allowed at the bank's VCCIO voltage.

References

- For more information on the DDR/DDR2 standard:
  - JEDEC (official specification): www.jedec.org
- Wikipedia:
  - www.wikipedia.org/wiki/DDR_SDRAM
  - www.wikipedia.org/wiki/DDR2_SDRAM

- * For information on participating vendors:
  - Micron Technology, Inc.: www.micron.com
  - Samsung Semiconductor: www.samsung.com
- For more information on the LatticeSC/M FPGA family: www.latticesemi.com
  - LatticeSC/M Family Data Sheet
  - LatticeSC/M technical notes
- TN1098, *LatticeSC sysCLOCK and PLL-DLL User's Guide*.

Conclusion

This user's guide discusses an implementation of a DDR/DDR2 Memory Interface to assist designers in determining whether the DDR/DDR2 SDRAM is appropriate for their application, and if so, to provide an understanding of its operation and details for design-in.

Technical Support Assistance

Hotline: 1-800-LATTICE (North America)
+1-503-268-8001 (Outside North America)
e-mail: techsupport@latticesemi.com
Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
February 2006	01.0	Initial release.
April 2007	01.1	Added Design Guidelines to Optimize Performance section.
June 2007	01.2	Updated DDR/DDR2 Memory Interface Block Diagram
July 2007	01.3	Added PLLs section.
July 2008	01.4	Updated information for K/K# Clocks.