LatticeCORE

DDR & DDR2 SDRAM Controller IP Cores User's Guide

February 2012
ipug35_05.0

Table of Contents

Chapter 1. Introduction 5
Quick Facts 5
Features 6
Chapter 2. Functional Description 7
Command Decode Logic 7
Configuration Interface 8
sysCLOCK PLL 8
Data Path Logic 8
Initialization State Machine 8
Command Application Logic 8
DDR I/O Modules 8
Signal Descriptions 9
Using the Local User Interface 10
Initialization and Auto-Refresh Control 10
Command and Address 11
Data Write 13
Data Read 14
Read/Write with Auto Precharge 14
Local-to-Memory Address Mapping 14
Mode Register Programming 15
Memory Interface 18
Chapter 3. Parameter Settings 19
Mode Tab 21
Type Tab 22
Select Memory 22
Clock 22
Memory Data Bus Size 22
Configuration 22
Data_rdy to Write Data Delay 23
Clock Width 23
CKE Width 23
DIMM Selection 23
Fixed Memory Timing 23
Command Burst Enable 23
Use Differential DQS 23
Setting Tab 24
Row Size 24
Column Size 24
Bank Size 24
Chip Select Width 24
User Slot Size 24
EMR Prog During Init 25
Auto Refresh Burst Count 25
External Auto Refresh Port 25
Mode Register Initial Setting 25
Timing Tab 25
DQS Pin Selection Tab (DDR2 Only) 27
Synthesis & Simulation Tools Option Tab 27
Info Tab 28

© 2012 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal. All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

Chapter 4. IP Core Generation 29
Licensing the IP Core 29
Getting Started 29
IPexpress-Created Files and Top Level Directory Structure 31
Generated Files 32
DDR Memory Controller Core Structure 34
Top-level Wrapper 34
Encrypted Netlist 34
I/O Modules 34
Clock Generator 34
Parameter File 35
Core Header File 35
Preference Files 35
Evaluation Project Files 35
Simulation Files for Core Evaluation 35
Testbench Top 36
Obfuscated Core Simulation Model 36
Command Generator 36
Monitor 36
TB Configuration Parameter 36
Memory Model 36
Memory Model Parameter 37
Evaluation Script File 37
Hardware Evaluation 37
Enabling Hardware Evaluation in Diamond 37
Enabling Hardware Evaluation in ispLEVER 37
Updating/Regenerating the IP Core 37
Regenerating an IP Core in Diamond 37
Regenerating an IP Core in ispLEVER 38
Chapter 5. Application Support 39
Core Implementation 39
Understanding Preferences 39
Preference Localization 39
VREF Assignments 40
DLL Allocation 40
I/O Types for DDR 41
Skew Treatment 42
Data Valid Generation 42
Dummy Logic Removal 43
Read Data Auto-Alignment Logic 43
PCB Routing Delay Compensation 43
Setting read_pulse_tap 44
DQS_PIO_READ Locate Constraints 44
Obtaining Location Values in Diamond Software 44
Obtaining Location Values in ispLEVER Software 45
Troubleshooting 47
Chapter 6. Core Verification 48
Chapter 7. Support Resources 49
Lattice Technical Support 49
Online Forums 49
Telephone Support Hotline 49
E-mail Support 49
Local Support 49

Internet 49
References 49
LatticeECP, LatticeXP 49
LatticeECP2/M 49
LatticeXP2 49
LatticeSC/M 50
LatticeECP3 50
Revision History 50
Appendix A. Resource Utilization 52
LatticeECP/EC Devices 52
Ordering Part Number 52
LatticeECP20 and LatticeEC20 Devices 52
LatticeECP2/S FPGAs 53
Ordering Part Number 53
LatticeECP2M/S FPGAs 53
Ordering Part Number 53
LatticeECP3 FPGAs 54
Ordering Part Number 54
LatticeXP Devices 54
Ordering Part Number 54
LatticeXP2 Devices 54
Ordering Part Number 54
LatticeSC/M FPGAs 55
Ordering Part Number 55
Appendix B. Errata 56

Introduction

The Double Data Rate (DDR) Synchronous Dynamic Random Access Memory (SDRAM) Controller is a general-purpose memory controller that interfaces with industry standard DDR/DDR2 memory devices/modules and provides a generic command interface to user applications. This core reduces the efforts required to integrate the DDR/DDR2 memory controller with the remainder of the application and minimizes the need to deal with the DDR/DDR2 memory interface. This core utilizes dedicated DDR input and output registers in the Lattice FPGA devices to meet the requirements for high-speed double data rate transfers. The timing parameters for a memory device or module can be set through the signals that are input to the core as a part of the configuration interface. This capability enables effortless switching among different memory devices by updating the timing parameters to suit the application without generating a new core configuration.

Throughout this guide, the term 'DDR' is used to represent the first-generation DDR memory. Since this document covers both the Lattice DDR and DDR2 memory controller IP cores, use of the term 'DDR' indicates both DDR and DDR2.

Quick Facts

Table 1-1 gives quick facts about the DDR IP core for LatticeEC™, LatticeECP™, LatticeECP2™, LatticeECP2M™, LatticeECP3™, LatticeECP3EA, LatticeXP™, LatticeXP2™, LattticeSC™, and LatticeSCM™ devices.

Table 1-1. DDR IP Core Quick Facts

		DDR IP Configuration									
		x32 Ics	**x64 Ics**	**x72 Ics**	**x32 Ics**	**x64 Ics**	**x72 Ics**	**x32 Ics**	**x64 Ics**	**x32 Ics**	**x64 Ics**
Core Requirements	FPGA Family Support	LatticeECP/EC & LatticeXP			LatticeECP2/M, LatticeXP2, LatticeECP3E & LatticeECP3EA					LatticeSC/M	
	Minimal Device Needed	LFEC3E-3T144C	LFECP6E-3T144C	LFXP3E-3T144C	LFE2-6E-5F256C	LFE2M20E-5F256C	LFXP2-5E-5TN144C	LFE3-17E-6FN484CES	LFE3-17EA-6FN484CES	LFSC3GA15E-5F256C	LFSCM3GA15EP1-5F256C
Resource Utilization	Target Device	LFEC33E-5F672C	LFECP33E-5F672C	LFXP20E-5F484C	LFE2-50E-6F672C	LFE2M35E-6F672C	LFXP2-17E-6F484CES	LFE3-95E-7FN1156CES	LFE3-95EA-7FN1156CES	LFSC3GA25E-6F900C	LFSCM3GA25EP1-6F900C
	Data Path Width	32	64	72	32	64	72	32	64	32	64
	LUTs	1400	2000	2100	1500	2000	2000	1400	1900	1300	1500
	sysMEM EBRs	0	0	0	0	0	0	0	0	0	0
	Registers	1800	2700	2900	1600	2300	2500	1600	2300	1600	2100
Design Tool Support	Lattice Implementation	Lattice Diamond® 1.2 or ispLEVER® 8.1 SP1									
	Synthesis	Synopsys® Synplify™ Pro for Lattice D-2009.12L-1									
		Mentor Graphics® Precision™ RTL									
	Simulation	Aldec® Active-HDL™ 8.2 Lattice Edition									
		Mentor Graphics ModelSim™ 6.3f SE									

Table 1-2 gives quick facts about the DDR2 IP CORE for LatticeECP2/M, LatticeECP3, LatticeECP3EA, LatticeXP, LatticeXP2, and LattticeSC/M devices.

Table 1-2. DDR2 IP CORE Quick Facts

		DDR2 IP Configuration						
		x32 Ics	**x64 Ics**	**x72 Ics**	**x32 Ics**	**x64 Ics**	**x32 Ics**	**x64 Ics**
Core Requirements	FPGA Families Supported	LatticeECP2/M, LatticeXP2, LatticeECP3E & LatticeECP3EA					LatticeSC/M	
	Minimal Device Needed	LFE2-6E-5F256C	LFE2M20E-5F256C	LFXP2-5E-5TN144C	LFE3-70E-6FN484CES	LFE3-17EA-6FN484CES	LFSC3GA15E-5F256C	LFSCM3GA15EP1-5F256C
Resource Utilization	Targeted Device	LFE2-50E-6F672C	LFE2M35E-6F672C	LFXP2-17E-6F484C	LFE3-95E-7FN1156CES	LFE3-95EA-7FN1156CES	LFSC3GA25E-6F900C	LFSCM3GA25EP1-6F900C
	Data Path Width	32	64	72	32	64	32	64
	LUTs	1450	1750	1850	1400	1700	1500	1700
	sysMEM EBRs	0						
	Registers	1550	2100	2250	1600	2150	1550	1950
Design Tool Support	Lattice Implementation	Diamond 1.2 or ispLEVER 8.1 SP1						
	Synthesis	Synopsys Synplify Pro for Lattice D-2009.12L-1						
		Mentor Graphics Precision RTL						
	Simulation	Aldec Active-HDL 8.2 Lattice Edition						
		Mentor Graphics ModelSim 6.3f SE						

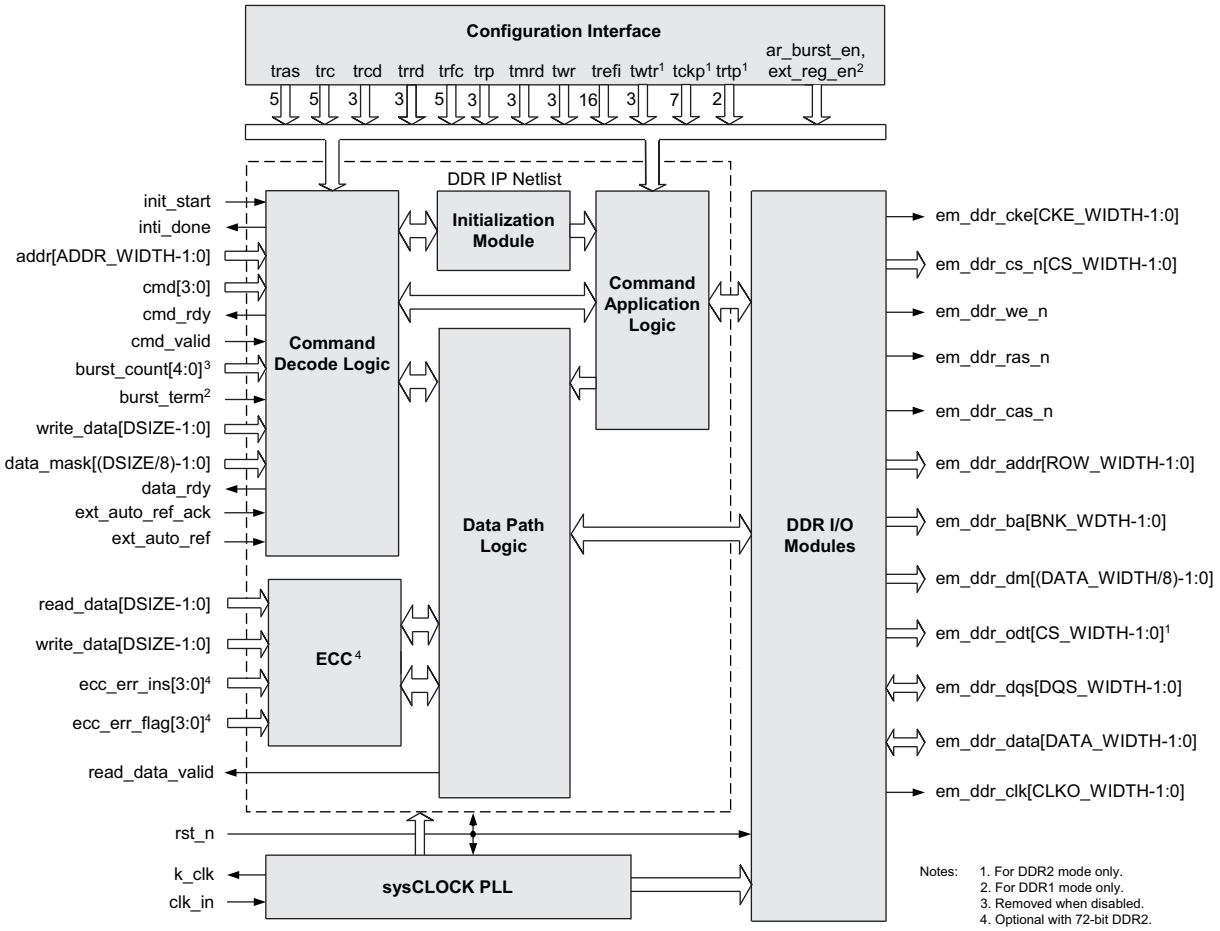
Features

- Interfaces to industry standard DDR/DDR2 SDRAM devices and modules
- High-performance DDR 400/333/266/200/133 operation for LatticeECP3, LatticeECP2/M, LatticeECP2/MS and LatticeSC/M devices; DDR 333/266/200/133 operation for LatticeECP/EC devices; and DDR 266/200/133 operation for LatticeXP devices

  Note: *Actual DDR memory specifications may not support the core's slower speed DDR operation (133 for DDR, 200/133 for DDR2). For the LatticeSC/M devices, see also TN1099,* LatticeSC/M DDR/DDR2 SDRAM Memory Interface User's Guide.
- High-performance DDR2 533/400/333/266/200/133 – operation for LatticeECP3, LatticeECP2/M, LatticeECP2/MS and LatticeSC/M devices; and DDR2 400/333/266/200/133 – operation for LatticeXP2 devices
- Programmable burst lengths of 2, 4 or 8 for DDR and 4 or 8 for DDR2
- Programmable CAS latency of 2 or 3 cycles for DDR and 3, 4, 5 or 6 cycles for DDR2
- Intelligent bank management to optimize performance by minimizing ACTIVE commands
- Supports all JEDEC standard DDR commands
- Two-stage command pipeline to improve throughput
- Supports both registered and unbuffered DIMM
- Command burst function with dynamic burst size control
- Supports all common memory configurations
  - SDRAM data path widths of 8, 16, 24, 32, 40, 48, 56, 64 and 72 bits
  - Variable address widths for different memory devices
  - Up to eight (DDR) or four (DDR2) chip selects for multiple SO/DIMM support
  - Programmable memory timing parameters
  - Byte-level writing through data mask signals

The DDR memory controller consists of two major parts, the encrypted netlist and I/O modules. The encrypted netlist comprises several internal blocks, as shown in Figure 2-1. The device architecture-dependent I/O modules are provided in RTL form. This section briefly describes the operation of each of these blocks.

Figure 2-1. DDR SDRAM Controller Block Diagram



Command Decode Logic

The Command Decode Logic (CDL) block accepts the user commands from the local interface. The accepted command is decoded to determine how the core will act to access the memory. When an accepted command is decoded as a write command, the CDL block asks the user logic to provide the write data. Once it receives the write data from the user logic, the CDL block delivers a write command to the Command Application Logic (CAL) block and the data is sent to the Data Path Logic (DPL) block. Similarly, when the accepted command is a read command, the CDL block sends a read command to the DPL block to generate a read command on the memory interface. The data read from memory is presented to the local user interface.

The CDL block also provides the command burst function that automatically repeats a user command up to the number of times specified via a user input. Intelligent bank management logic tracks the open/close status of every bank and stores the row address of every open bank. This information is used to reduce the number of PRECHARGE and ACTIVE commands issued to the memory. The controller also utilizes two pipelines to improve throughput. One command in the queue is decoded while another is presented at the memory interface.

Configuration Interface

The Configuration Interface (CI) block provides the DDR memory controller with the core reconfiguration capability for the memory timing parameters and other core configuration inputs. The configuration interface for the memory timing parameters can be enabled or disabled via a user parameter. When enabled, the DDR memory controller core can be reconfigured with an updated set of the memory timing parameters in the parameter file without generating a new IP core. When disabled, the reconfiguration logic is permanently removed from the core. It is generally expected that the IP core performance will be improved due to a lower utilization.

sysCLOCK PLL

The sysCLOCK™ PLL block generates the clocks used in all blocks in the memory controller core and provides the system clock to the user logic. If an external clock generator is to be used, it is possible to remove this block from the IP core structure.

Data Path Logic

The DPL block interfaces with the DDR I/O modules and is responsible for generation of the read data and read data valid signal in the read operation mode. This block implements the logic to ensure that the data read from the memory is transferred to the local user interface in a deterministic and coherent manner. The write data does not go through the DPL block; it is directly transferred to the Command Application Logic (CAL) block for the write operation mode. The implementation of the DPL block is also device dependent.

Initialization State Machine

The Initialization State Machine (ISM) block performs the DDR memory initialization sequence defined by JEDEC. Although the memory initialization must be done after the power-up, it is the user's responsibility to provide a user input to the block to start the memory initialization sequence. The ISM block provides an output that indicates the completion of the sequence to the local user interface.

Command Application Logic

The CAL block accepts the decoded commands from the Command Decode Logic on two separate queues. These commands are translated to the memory commands in a way that meets the timing requirements of the memory device. The CI block provides the memory timing parameters to the CAL block so that the timing requirements are satisfied during the command translations. Commands in the two stage queues are pipelined to maximize the throughput on the memory interface. The CDL and the CAL blocks work in parallel to fill and empty the queues respectively.

DDR I/O Modules

The DDR I/O modules are directly connected to the memory interface providing all required DDR ports for memory access. They convert the single data rate (SDR) data to DDR data for write operations and perform the DDR to SDR conversion for read operations. The I/O modules utilize the dedicated FPGA DDR I/O logic and are designed to reliably drive and capture the data on the memory interface.

Signal Descriptions

Table 2-1 describes the user interface and memory interface signals at the top level.

Table 2-1. DDR SDRAM Memory Controller Top-Level I/O List

Port Name	Active State	I/O	Description
Local User Interface			
clk_in	N/A	Input	Reference clock. It is connected to the PLL input. This port is only for DDR1 mode.
k_clk	N/A	Input	System clock. It is connected to the PLL output. This port is only for DDR2 mode.
k1_clk	N/A	Input	System clock, 270-degree phase shift from k_clk. It is connected to the PLL output. This port is only for LatticeECP3 devices in DDR2 mode.
k4_clk	N/A	Input	System clock, 90-degree phase shift from k_clk. It is connected to the PLL output. This port is only for LatticeSC/SCM devices in DDR2 mode.
rst_n	Low	Input	Asynchronous reset. It resets the entire core when asserted.
init_start	High	Input	Initialization start. It should be asserted at least 200µs after the power-on reset to initiate the memory initialization.
cmd[3:0]	N/A	Input	User command input to the memory controller.
cmd_valid	High	Input	Command and address valid input. When asserted, the addr, cmd and burst_count inputs are validated.
addr[ADDR_WIDTH –1:0]	N/A	Input	User address input to the memory controller.
burst_count[4:0]	N/A	Input	Command burst count input. It indicates the number of repeats of a given read or write command. The command burst feature is enabled when BRST_CNT_EN is defined. If not defined, burst_count is removed from the core.
burst_term	High	Input	Burst termination. This port is only for the DDR1 mode.
write_data[DSIZE –1:0]	N/A	Input	Write data input from user logic to the memory controller.
data_mask[(DSIZE/8) –1:0]	High	Input	Data mask input for write_data. Each bit masks the corresponding byte on the write_data bus, in order.
ext_auto_ref	High	Input	User auto-refresh control input. This port is enabled when EXT_AUTO_REF is defined.
k_clk	N/A	Output	System clock output. The user logic uses this as a system clock unless an external clock generator is used. This port is only for DDR1 mode.
k_clk_out	N/A	Output	System clock output. This port is only for DDR2 mode.
init_done	High	Output	Initialization done output. It is asserted for one clock cycle when the core completes the memory initialization routine.
ext_auto_ref_ack	High	Output	User auto-refresh control acknowledge output. This port is enabled when EXT_AUTO_REF is defined.
cmd_rdy	High	Output	Command ready output. When asserted, it indicates the core is ready to accept the next command and address.
data_rdy	High	Output	Data ready output. When asserted, it indicates the core is ready to receive the write data.
read_data[DSIZE –1:0]	N/A	Output	Read data output from the memory to the user logic.
read_data_valid	High	Output	Read data valid output. When asserted, it indicates the data on the read_data bus is valid.
read_pulse_tap	N/A	Input	This signal is used for PCB routing delay compensation, to guarantee that the falling edge of "dqs_pio_read" signal is placed in the preamble of the incoming DQS signal.

Table 2-1. DDR SDRAM Memory Controller Top-Level I/O List (Continued)

Port Name	Active State	I/O	Description
uddcntl_n	N/A	Output	Update control. It is connected to the DQSDLL input. This port is only for LatticeECP2/ECP2S/ECP2M/ECP2MS/ECP3/XP2 devices in DDR2 mode.
dqsdel	High	Input	DQS delay. It is connected to the DQSDLL output. This port is only for LatticeECP2/ECP2S/ECP2M/ECP2MS/ECP3/XP2 devices in DDR2 mode.
DDR SDRAM Memory Interface			
em_ddr_clk[CLKO_WIDTH –1:0]	N/A	Output	DDR memory clock generated by the memory controller.
em_ddr_cke[CKE_WIDTH –1:0]	High	Output	DDR memory clock enable generated by the memory controller.
em_ddr_addr[ROW_WIDTH –1:0]	N/A	Output	DDR memory address. It has the multiplexed row and column address for the memory.
em_ddr_ba[BNK_WDTH –1:0]	N/A	Output	DDR memory bank address.
em_ddr_data[DATA_WIDTH –1:0]	N/A	In/Out	DDR memory bi-directional data bus.
em_ddr_dm[(DATA_WIDTH/8) –1:0]	High	Output	DDR memory write data mask. It is used to mask the byte lanes for byte level write control.
em_ddr_dqs[(DQS_WIDTH –1:0]	N/A	In/Out	DDR memory bi-directional data strobe. This strobe signal is associated with either 4 or 8 data pads.
em_ddr_cs_n[CS_WIDTH –1:0]	Low	Output	DDR memory chip select.
em_ddr_cas_n	Low	Output	DDR memory column address strobe.
em_ddr_ras_n	Low	Output	DDR memory row address strobe.
em_ddr_we_n	Low	Output	DDR memory write enable.
em_ddr_odt[CS_WIDTH –1:0]	High	Output	DDR memory on-die termination control. This output is present only in the DDR2 mode.

Using the Local User Interface

The local user interface of the DDR memory controller IP core consists of four independent functional groups:

- Initialization and Auto-Refresh Control
- Command and Address
- Data Write
- Data Read

Each functional group and its associated local interface signals are listed in Table 2-2.

Table 2-2. Local User Interface Functional Groups

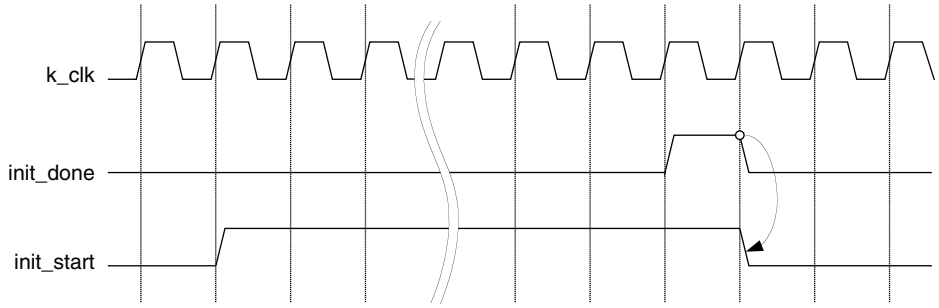
Functional Group	Signals
Initialization and Auto-Refresh Control	init_start, init_done, ext_auto_ref, ext_auto_ref_ack
Command and Address	addr, cmd, cmd_rdy, cmd_valid
Data Write	data_rdy, write_data, data_mask
Data Read	read_data, read_data_valid

Initialization and Auto-Refresh Control

The DDR memory devices must be initialized before the memory controller can access them. The memory controller starts the memory initialization sequence when the init_start signal is asserted by the user interface. The user must wait at least 200 µs after the power-up cycle is completed and the system clock is stabilized, and then generate the initialization start input to the core. Once asserted, the init_start signal needs to be held high until the initial-

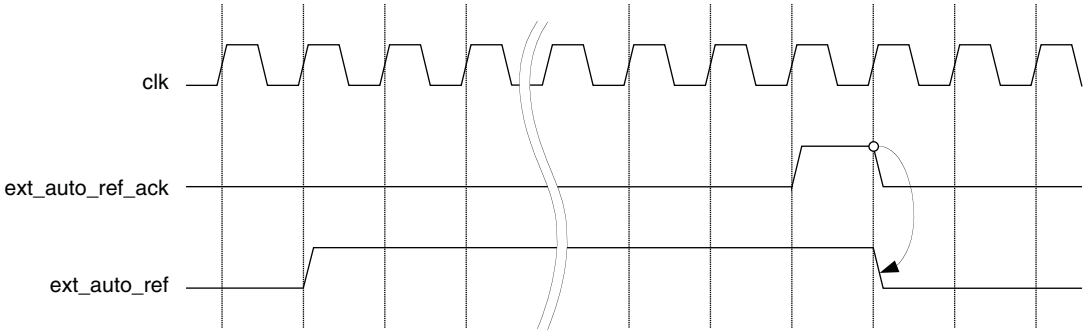
ization process is completed. The init_done signal is asserted high for one clock cycle when the core has completed the initialization sequence and is now ready to access the memory. The init_start signal must be deasserted as soon as init_done is asserted. The memory initialization is required only once after the system reset. Note that the core will operate with the default memory configuration initialized in this process if the user does not program the MR and/or EMR registers. Figure 2-2 shows the timing diagram of the initialization control signals.

Figure 2-2. Timing of Memory Initialization Control



The memory controller core provides the user auto-refresh control feature. This feature can be enabled by the External Auto Refresh Port option. It is a useful function for applications that need to have a complete control on the DDR interface in order to avoid unwanted intervention caused by the memory refresh operations. Once enabled, ext_auto_ref is asserted by a user to force the core to generate a set of Refresh commands in a burst. The number of Refresh commands in a burst is defined by the Auto Refresh Burst Count option. The ext_auto_ref_ack signal is asserted high for one clock cycle to indicate that the core has generated the Refresh commands. The ext_auto_ref signal can be deasserted once the acknowledge signal is detected as shown in Figure 2-3.

Figure 2-3. Timing of External Auto Refresh Control



Command and Address

Once the memory initialization is done, the core waits for user commands that will access the memory. The user logic needs to provide the command and address to the core along with the control signals. The commands and addresses are delivered to the core using the procedure described below:

1. The memory controller core tells the user logic that it is ready to receive a command by asserting the cmd_rdy signal for one clock cycle.

2. If the core finds the cmd_valid signal asserted by the user logic while it is asserting cmd_rdy, it takes the cmd input as a valid user command. The core also accepts the addr input as a valid start address or mode register programming data depending on the command type. If cmd_valid is not asserted, the cmd and addr inputs become invalid and the core ignores them.

3. The cmd, addr and cmd_valid inputs become “don’t care” while cmd_rdy is deasserted.

4. The cmd_rdy signal is asserted again to take the next command.

Note: The first read command issued on the local user interface is a dummy transaction used to initialize the receive physical layer logic. Data will not be returned from the memory until the second read request.

When the core is in the command burst operation, it extensively occupies the data bus. While the core is operating in the command burst mode, it can keep maximum throughput by internally repeating the command. The command burst can be enabled by checking the "Command Burst Enable" box in the Type tab of the IPexpress GUI. (Refer to “Type Tab” on page 22.)

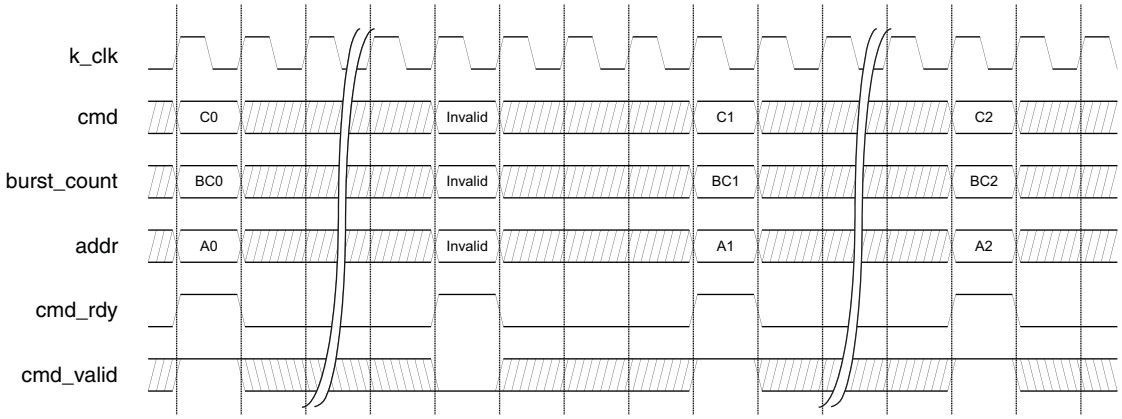
When command burst is enabled, the memory controller can repeat the given READ or WRITE command up to 31(32 in DDR2 mode) times. The burst_count[4:0] input port is used to set the number of repeats of the given command, it indicates the number of memory BL(DDR/2 memory burst length), for example, if DQ width is 8bit, burst_count is 2, BL is 4, then the data length for each write or read command from user should be
DQ width * burst_count * BL = 8*2*4=64 bit= 8 byte.

The core allows the command burst function to access the memory addresses within the current page. When the core reaches the boundary of the current page while accessing the memory in the command burst mode, the next address that the core will access will be the beginning of the same page, causing it to overwrite the contents of the location or read unexpected data. Therefore, the user must track the accessible address range in the current page while the command burst operation is performed. If an application requires a fixed command burst size, use of 2-, 4-, 8-, or 16-burst (32-burst in DDR2 mode) it is recommended to ensure that the command burst accesses do not cross the page boundary. Note that the value “00000” stands for “32” in DDR2 mode, but it is invalid in DDR1 mode.

When command burst is disabled, the burst_count port is removed from the core, and the core will always consider the burst_count as “1” internally.

The burst_count input is sampled the same way as cmd. The timing of the Command and Address group is shown in Figure 2-4. The timing for burst count in Figure 2-4 shows only the sampling time of the bus. When burst_count is sampled with a value other than “00001”, the core will prevent cmd_rdy from being asserted when two queues in CDL are both occupied, if any queue in CDL is empty, the cmd_rdy will be asserted, whatever the command burst is completed or not.

Figure 2-4. Timing of Command and Address



Each command on the cmd bus must be a valid command. Lattice defines the valid memory commands as shown in Table 2-3. All other values are reserved and considered invalid.

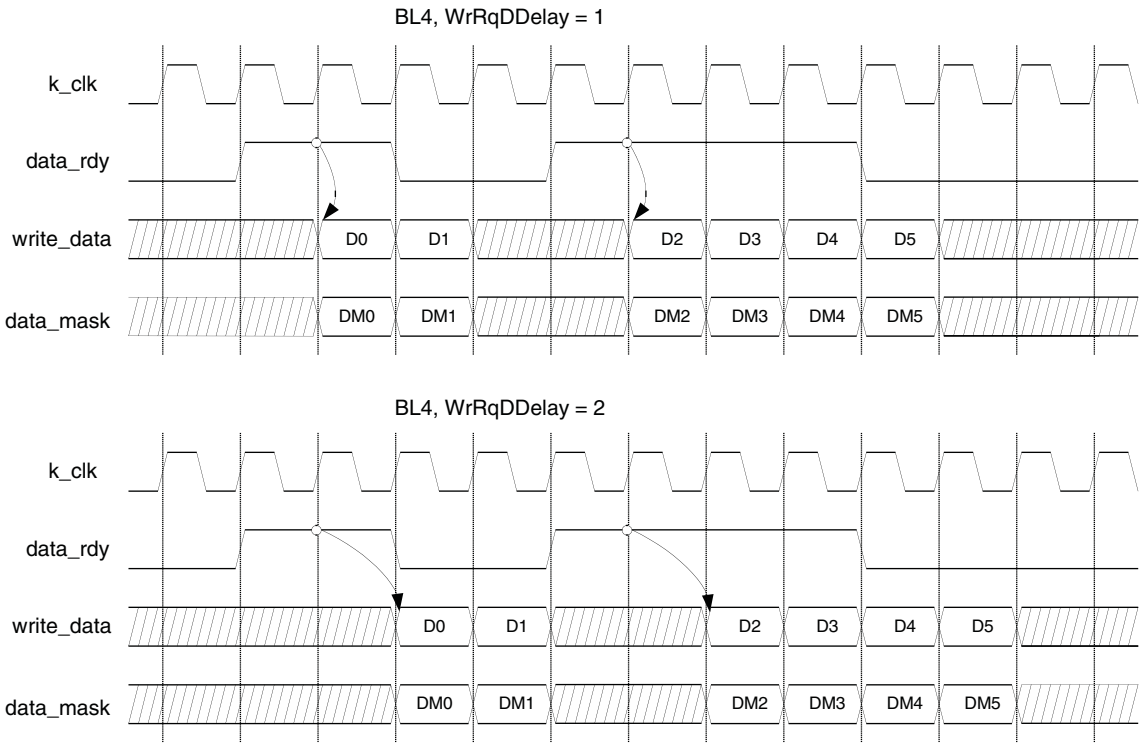
Table 2-3. Defined User Commands

Command	Mnemonic	cmd[3:0]
Read	READ	0001
Write	WRITE	0010
Read with Auto Precharge	READA	0011
Write with Auto Precharge	WRITEA	0100
Powerdown	PDOWN	0101
Load Mode Register	LOAD_MR	0110
Self Refresh	SELF_REF	0111

Data Write

After the WRITE command is accepted, the memory controller core asserts the data_rdy signal when it is ready to receive the write data from the user logic to be written into the memory. Since the duration from the time a write command is accepted to the time the data_rdy signal is asserted is not fixed, the user logic needs to monitor the data_rdy signal to detect when it is asserted. Once data_rdy is asserted, the core expects valid data on the write_data bus one or two clock cycles after the data_rdy signal is asserted. The write data delay is programmable by the user parameter, WrRqDDelay, providing flexible back-end application support. For example, setting WrRqD-Delay = 2 ensures that the core takes the write data out in proper time when the local user interface of the core is connected to a synchronous FIFO module inside the user logic. Figure 2-5 shows two examples of the local user interface data write timing. Both cases are in the BL4 mode. The upper diagram shows the case of one clock cycle delay of write data, while the lower one displays a two clock-cycle delay case. The memory controller considers D0, DM0 through D5, DM5 valid write data.

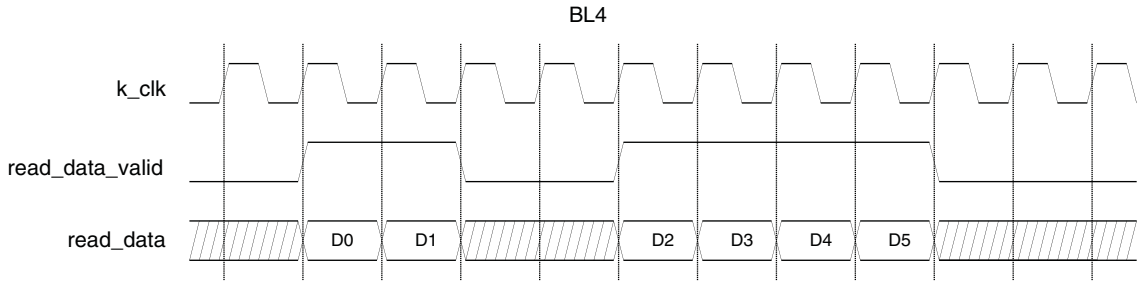
Figure 2-5. One-Clock vs. Two-Clock Write Data Delay



Data Read

When the READ command is accepted, the memory controller core accesses the memory to read the addressed data and brings it back to the local user interface. Once the read data is available on the local user interface, the memory controller core asserts the read_data_valid signal to tell the user logic that the valid read data is on the read_data bus. The read data timing on the local user interface is shown in Figure 2-6.

Figure 2-6. Read Data Timing on Local User Interface



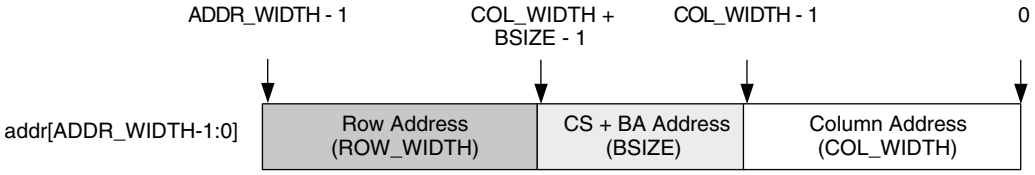
Read/Write with Auto Precharge

The DDR2 IP core automatically closes (precharges) and opens rows according to the user memory address accesses. Therefore, the READA and WRITEA commands are not used for most applications. The commands are provided to comply to the JEDEC DDR2 specification.

Local-to-Memory Address Mapping

Mapping local addresses to memory addresses is an important part of a system design when a memory controller function is implemented. Users must know how the local address lines from the memory controller connect to those address lines from the memory because proper local-to-memory address mapping is crucial to meet the system requirements in applications such as a video frame buffer controller. Even for other applications, careful address mapping is generally necessary to optimize the system performance. On the memory side, the address (A), bank address (BA) and chip select (CS) inputs are used for addressing a memory device. Users can obtain this information from the memory device data sheet. Figure 2-7 shows the local-to-memory address mapping of the Lattice DDR memory controller cores.

Figure 2-7. Local-to-Memory Address Mapping for Memory Access



ADDR_WIDTH is calculated by the sum of COL_WIDTH, ROW_WIDTH and BSIZE. BSIZE is determined by the sum of the bank address size and chip select address size. For 4- or 8-Bank DDR2 devices, the bank address size is 2 or 3, respectively. When the number of chip select is 1, 2 or 4, the chip select address size becomes 0, 1, or 2, respectively. An example of the address mapping is shown in Table 2-4 and Figure 2-8.

Table 2-4. An Example of Address Mapping

User Selection Name	User Value	Parameter Name	Parameter Value	Actual Line Size	Local Address Map
Row Size	14	ROW_WIDTH	14	14	addr[28:15]
Column Size	11	COL_WIDTH	11	11	addr[10:0]
Bank Size	8	BNK_WDTH	3	3	addr[13:11]
Chip Select Width	2	CS_WIDTH	2	1	addr[14]
Total Local Address Line Size		ADDR_WIDTH	29	29	addr[28:0]

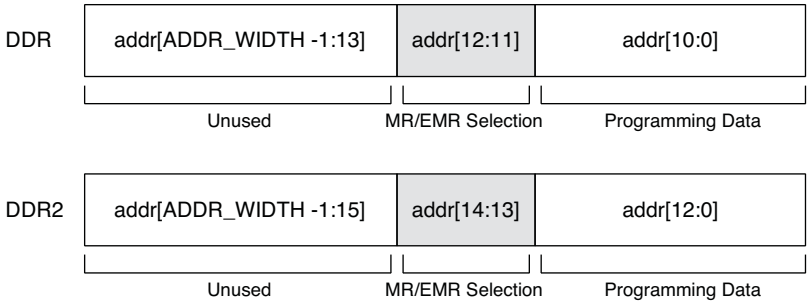
Figure 2-8. Mapped Address for the Example

Mode Register Programming

The DDR SDRAM memory devices are programmed using the mode register (MR) and extended mode registers (EMR). The bank address bus (em_ddr_ba) is used for choosing one of the MR or EMR registers, while the programming data is delivered through the address bus (em_ddr_addr). The memory data bus cannot be used for the MR/EMR programming.

The Lattice DDR memory controller core uses the local address bus, addr, to program these registers. It uses different address mapping from the address mapping for memory accesses. The core accepts a user command, LOAD_MR, to initiate the programming of MR/EMR registers. When LOAD_MR is applied on the cmd bus, the user logic must provide the information for a target mode register and the programming data on the addr bus. When the target mode register is programmed, the memory controller core is also configured to support the new memory setting. Figure 2-9 shows how the local address lines are allocated for the programming of memory registers.

Figure 2-9. Local-to-Memory Address Mapping for MR/EMR Programming



The register programming data is provided through the lower side of the addr bus starting from the bit 0 for LSB. The programming data requires eleven (DDR1 mode) or thirteen (DDR2 mode) bits of the local address lines. Two more bits are needed to choose a target register as listed in Table 2-5. All other upper address lines are unused during the command patch cycle for the LOAD_MR command.

Table 2-5. Mode Register Selection Using Bank Address

Mode Register	Local Address	
	DDR (addr[12:11])	DDR2 (addr[14:13])
MR	00	00
EMR	01	01
EMR2[1]	-	10
EMR3[1]	-	11

1. DDR2 mode only

Figure 2-10 shows the use of local address for typical DDR2 memory configurations. The DDR memory configuration is accomplished the same way, except that it accesses only two registers, MR and EMR. Starting from DDR/DDR2 version 6.7, some of the registers such as Burst Type, Burst Length, CAS Latency and others can be configured directly from the IPexpress GUI for custom initialization.

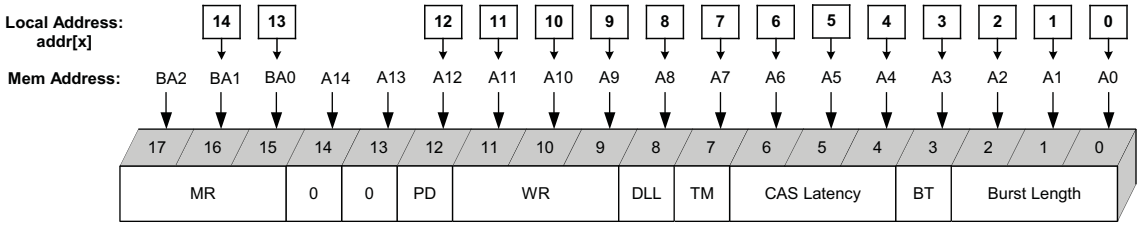
The initialization default values for all mode registers are listed in Table 2-6.

Table 2-6. Initialization Default Values for DDR/DDR2 Mode Registers

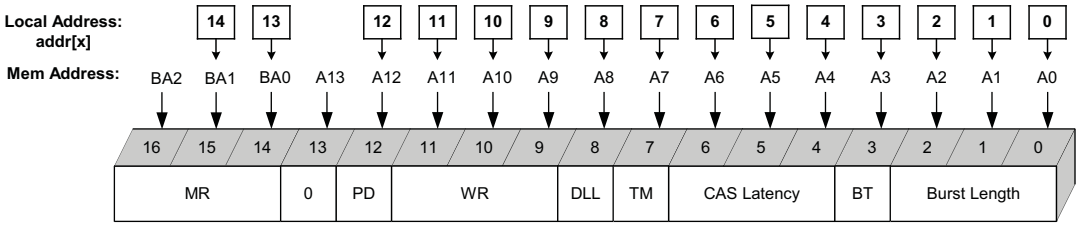
Type	Registers	Value	Description	Local address
DDR MR (BA[1:0] = 00)	Burst Length[1]	3'b010	BL = 4	addr[2:0]
	Burst Type[1]	1'b0	Sequential	addr[3]
	Cas Latency[1]	3'b010	CL = 2 Cycles	addr[6:4]
	Test Mode	1'b0	Normal	addr[7]
	DLL Reset	1'b1	DLL Reset = Yes	addr[8]
	All Others	0		addr[ROW_WIDTH-1:8]
DDR EMR (BA[1:0] = 01)	DLL	1'b0	DLL Enable	addr[0]
	Drive Strength	1'b0	Normal	addr[1]
	All Others	0		addr[ROW_WIDTH-1:2]
DDR2 MR (BA[1:0] = 00 or BA[2:0]=000)	Burst Length[1]	3'b010	BL = 4	addr[2:0]
	Burst Type[1]	1'b0	Sequential	addr[3]
	Cas Latency[1]	3'b100	CL = 4 Cycles	addr[6:4]
	Test Mode	1'b0	Normal	addr[7]
	DLL Reset	1'b1	DLL Reset = Yes	addr[8]
	WR Recovery[1]	3'b010	3 Cycles	addr[11:9]
	Power Down Exit[1]	1'b0	Fast	addr[12]
	All Others	0		addr[ROW_WIDTH-1:13]
DDR2 EMR (BA[1:0] = 01 or BA[2:0]=001)	DLL	1'b0	DLL Enable	addr[0]
	Drive Strength	1'b0	Normal	addr[1]
	RTT0[1]	1'b0	Disabled with RTT1=0	addr[2]
	Additive Latency[1]	3'b011	3 Cycles	addr[5:3]
	RTT1[1]	1'b0	Disabled with RTT0=0	addr[6]
	OCD	3'b000	OCD Not Applicable	addr[9:7]
	DQS Mode	1'b1	Differential Disabled	addr[10]
	RDQS	1'b0	Disable	addr[11]
	Outputs	1'b0	Enable	addr[12]
	All Others			addr[ROW_WIDTH-1:13]
DDR2 EMR2 (BA[1:0] = 10 or BA[2:0]=010)	All	0		addr[ROW_WIDTH-1:0]
DDR2 EMR3 (BA[1:0] = 11 or BA[2:0]=011)	All	0		addr[ROW_WIDTH-1:0]

1. This register can be initialized with a custom value through the IPexpress GUI.

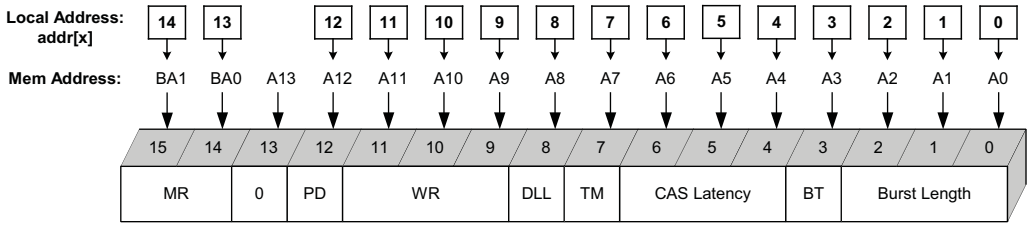
Figure 2-10. Local Address Mapping for MR Programming (Typical DDR2 Memory Configurations)



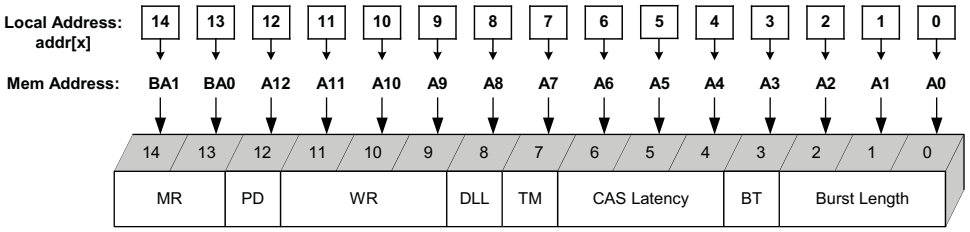
Row Size = 15, Bank Size = 8 (2Gb)



Row Size = 14, Bank Size = 8 (1Gb)



Row Size = 14, Bank Size = 4 (512Mb)



Row Size = 13, Bank Size = 4 (256Mb)

Memory Interface

Table 2-7 lists the connections of the DDR interface between the Lattice DDR memory controller core and memory.

Table 2-7. DDR Interface Signal Connections to DDR Memory

Core Port Name	Memory Port Name[1]	Width	VREF[2]		I/O Type	
			DDR	DDR2	DDR	DDR2
em_ddr_clk[3]	CK, CK#	CLKO_WIDTH	—	—	SSTL25D_II	SSTL18D_II
em_ddr_cke	CKE	CKE_WIDTH	—	—	SSTL25_II	SSTL18_II
em_ddr_odt[4]	ODT	CS_WIDTH	—	—	N/A	SSTL18_II
em_ddr_cs_n	CS#	CS_WIDTH	—	—	SSTL25_II	SSTL18_II
em_ddr_ras_n	RAS#	1	—	—	SSTL25_II	SSTL18_II
em_ddr_cas_n	CAS#	1	—	—	SSTL25_II	SSTL18_II
em_ddr_we_n	WE#	1	—	—	SSTL25_II	SSTL18_II
em_ddr_addr	A	ROW_WIDTH	—	—	SSTL25_II	SSTL18_II
em_ddr_ba	BA	BNK_WDTH	—	—	SSTL25_II	SSTL18_II
em_ddr_data	DQ	DATA_WIDTH	1.25V	0.9V	SSTL25_II	SSTL18_II
em_ddr_dm	DM	DATA_WIDTH/8	—	—	SSTL25_II	SSTL18_II
em_ddr_dqs	DQS	DQS_WIDTH	1.25V	0.9V or none[5]	SSTL25_II	SSTL18_II or SSTL18D_II[6]

1. The listed DDR memory port names are from the Micron DDR memory data sheet.
2. In the banks with multiple VREFs, only VREF1 is used for DDR memory applications. VREF = VCCIO/2.
3. Lattice DDR memory controller core defines only the positive-end signal for the memory clock. The negative-end pad is allocated by the implementation software when a differential I/O type is assigned.
4. The ODT ports are available only in the DDR2 mode.
5. If DQS uses a differential pair, VREF is not required. However, VREF1 is still used for the DQS preamble detection.
6. In the DDR2 mode, either single-ended or differential type of DQS can be selected.

Parameter Settings

The IPexpress™ tool is used to create IP and architectural modules in the Diamond or ispLEVER software. Refer to "IP Core Generation" on page 29 for a description on how to generate the IP.

Table 3-1 provides the list of user configurable parameters for the DDR/DDR2 IP core. The parameter settings are specified using the DDR/DDR2 IP core Configuration GUI in IPexpress. The numerous PCI Express parameter options are partitioned across multiple GUI tabs as shown in this chapter.

Table 3-1. DDR SDRAM Memory Controller Parameters

Parameters	Range/Options	Default Value
Mode		
Add SMI Port Interface for PLL and DLL[1]	Disable, Enable	Disable
Type		
Select Memory - DDR2	Micron DDR2 256Mb -5E Micron DDR2 512Mb -5E Micron DDR2 1Gb -5E Micron DDR2 2Gb -5E Micron DDR2 256Mb -37E Micron DDR2 512Mb -37E Micron DDR2 1Gb -37E Micron DDR2 2Gb -37E Custom	Micron DDR2 256Mb -5E
Select Memory - DDR	Micron DDR 512Mb -5E Micron DDR 512Mb -6E Micron DDR 512Mb -75E Custom	Micron DDR 512Mb -5E
Clock	DDR - 133-200 DDR2 - 166 -333	200
Memory Data Bus size	8, 16, 24, 32, 40, 48, 56, 64, 72	32
Configuration	x4, x8, x16[2]	x8
Data_rdy to Write Data Delay	1, 2	1
Clock Width	1, 2	1
CKE Width	1, 2	1
DIMM Selection	Registered DIMM, Unbuffered Memory	Unbuffered Memory
Fixed Memory Timing	Disable, Enable	Disable
Command Burst Enable	Disable, Enable	Disable
Use Differential DQS[3]	Disable, Enable	Enable
Setting		
Address		
Row Size	13 - 16	13
Column Size	9 - 11	10
Bank Size	4, 8[4]	4
Chip Select width	1, 2, 4	1
User Slot Size	1, 2	1
EMR Prog During Init[5]	Disable, Enable	Enable
Auto Refresh Control		
Auto Refresh Burst Count	2-8	8

Table 3-1. DDR SDRAM Memory Controller Parameters (Continued)

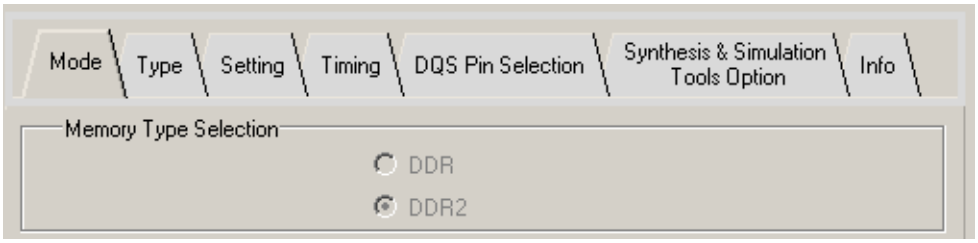
Parameters	Range/Options	Default Value
External Auto Refresh Port	Disable, Enable	Disable
Mode Register Initial Setting - DDR2		
Burst Length	4, 8	4
CAS Latency	2 - 6	4
Additive Latency	0 - 4	3
Write Recovery	2 - 6	3
RTT_Nom(ohm)	50 ohm, 75 ohm, 150 ohm, Disable	Disable
Burst Type	Sequential, Interleaved	Sequential
DLL Control for PD	Fast End, Slow Exit	Fast Exit
Mode Register Initial Setting - DDR		
Burst Length	2, 4, 8	4
CAS Latency	1.5, 2, 2.5, 3	2
Burst Type	Sequential, Interleaved	Sequential
Timing		
TRCD	1 - 7	3
TRAS	1 - 31	8
TRFC	1 - 63	DDR: 14 DDR2: 21
TMRD	1 - 7	2
TRP	1 - 7	3
TRRD	1 - 7	2
TRC	1 - 31	11
TREFI	1 - 65536	1563
TWTR	1 - 7	2
TRTP	1 - 4	2
Synthesis & Simulation Tools Option		
Support Synplify, Support Precision, Support ModelSim, Support ALDEC	Enable, Disable	Enable

Notes:
1. LatticeSC/SCM only.
2. x16 is not available for data bus size 8/24/40/56/72 in DDR2 mode.
3. DDR2 only.
4. DDR has 4 only.
5. The EXT_REG_EN parameter is effective only in the DDR1 mode.

Mode Tab

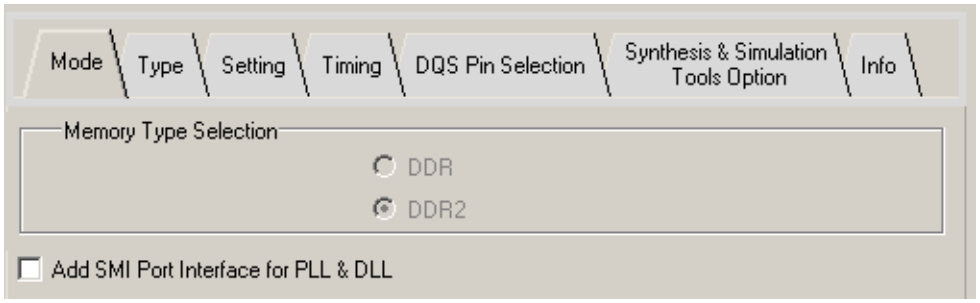
The Memory Type Selection field is not a user option but is selected by IPexpress when a DDR memory controller core is selected from the IPexpress IP core list. Figure 3-1 shows the contents of the Mode tab.

Figure 3-1. Mode Tab



When configuring a DDR memory controller core for LatticeSC/LatticeSCM, the “Add SMI Port Interface for PLL & DLL” is available. Figure 3-2 shows the contents of the Mode tab (LatticeSC/LatticeSCM only).

Figure 3-2. Mode Tab (LatticeSC/LatticeSCM Only)



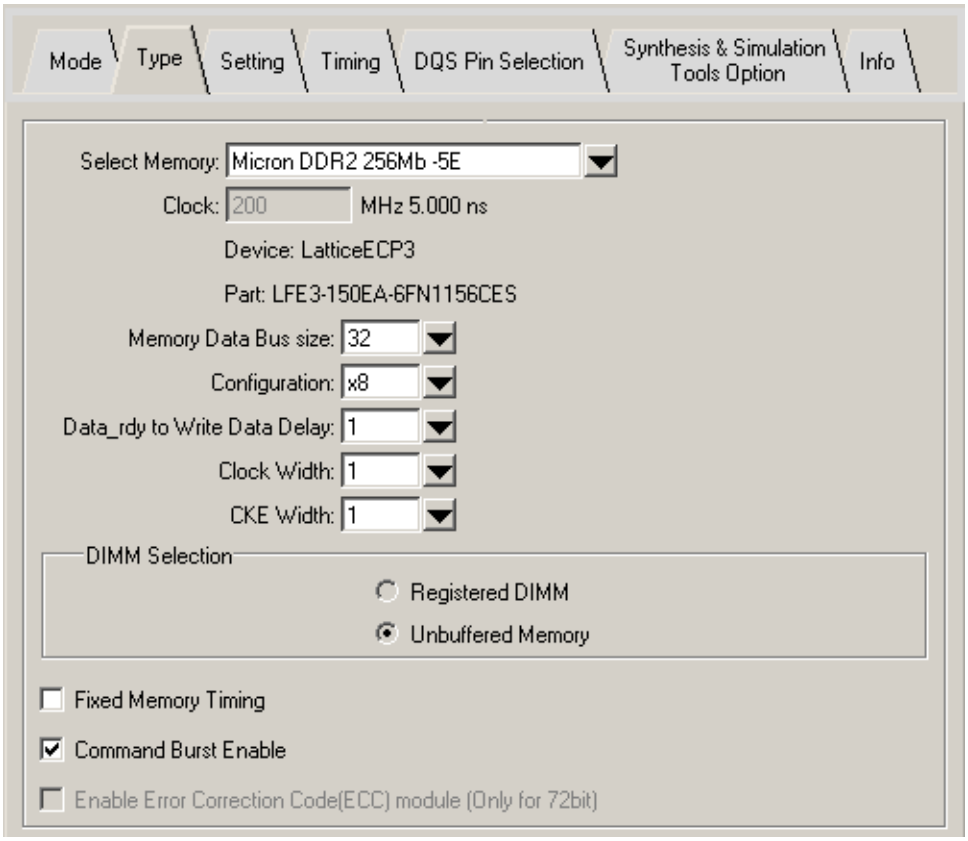
Add SMI Port Interface for PLL & DLL

The DDR memory controller cores for the LatticeSC/M devices have an option to add the SMI Port Interface. This option is not available for all other target devices. The SMI port is used to dynamically configure the PLL and DLL modules used in the LatticeSC/M DDR memory controllers. The ddr_pll90 (PLL1), ddr_rdpll (PLL2) and ddr_trdll (DLL2) blocks have the offset address 0x410, 0x420 and 0x430, respectively. See TN1085, *LatticeSC MPI/System Bus,* for details on the use of the SMI port Interface.

Type Tab

The Type tab enables users to select various configuration options for the target memory device/ module and the core functional features. Figure 3-3 shows the contents of the Type tab.

Figure 3-3. Type Tab



Select Memory

A predefined DDR memory device is selected by default. The timing parameters for the selected memory are listed in the Timing tab. One or more different speed grade memory devices are also available for easy selection. If the desired memory device requires different timing parameters from the pre-defined ones, the Custom option should be selected.

Clock

When a predefined memory is selected, the Clock field displays the corresponding clock speed, and it cannot be modified. With the Custom option selected, a user target speed must be provided.

Memory Data Bus Size

This option means the memory data bus width to which the memory controller core is connected. If a memory module that has a wider data bus than required is to be used, only the required data width has to be selected.

Configuration

This option is used to select the device configuration of the DIMM or on-board memory. Device configurations x4, x8, and x16 are supported.

Data_rdy to Write Data Delay

This option is selected according to the user local back-end application's requirement. The user logic can send the write data to the core with either one-clock cycle or two-clock cycle delay.

Clock Width

This option sets the number of clocks with which the memory controller drives the memory. The IPexpress tool can generate either one or two memory clocks. Once a DDR memory controller core is generated, more memory clocks can be manually instantiated for those applications that need more than two memory clocks.

CKE Width

The number of memory clock enable signals is configured using this option. More clock enable signals can also be instantiated by the user.

DIMM Selection

Registered memory modules have different timing requirements from unbuffered ones. Lattice memory controller cores support both types. A proper type must be selected before the core generation because only one type is supported once a core is generated. DIMM Selection is available when **Custom** is chosen in the Select Memory drop-down. (Refer to "Select Memory" on page 22.)

Fixed Memory Timing

This option disables the memory timing reconfiguration feature for a generated core. When disabled, the IP core only supports the timing parameter set applied at the time of the core generation. This option may provide somewhat improved performance with lower resource utilization by removing the reconfiguration logic from the core. This option should not be selected if it is necessary to support different memory timing parameters without regenerating the core. This option is unchecked by default.

Command Burst Enable

This option enables the core's command burst feature. With this option enabled, the core automatically repeats a memory-read or memory-write command the number of times specified by the burst_count bus. If unchecked, the command burst function is disabled and the burst_count bus is removed from the core. Disabling this option may provide somewhat improved performance with lower utilization by removing the burst_count bus logic from the core. The default of this option is "Enabled." For more detail about this feature, see "Command and Address" on page 11.

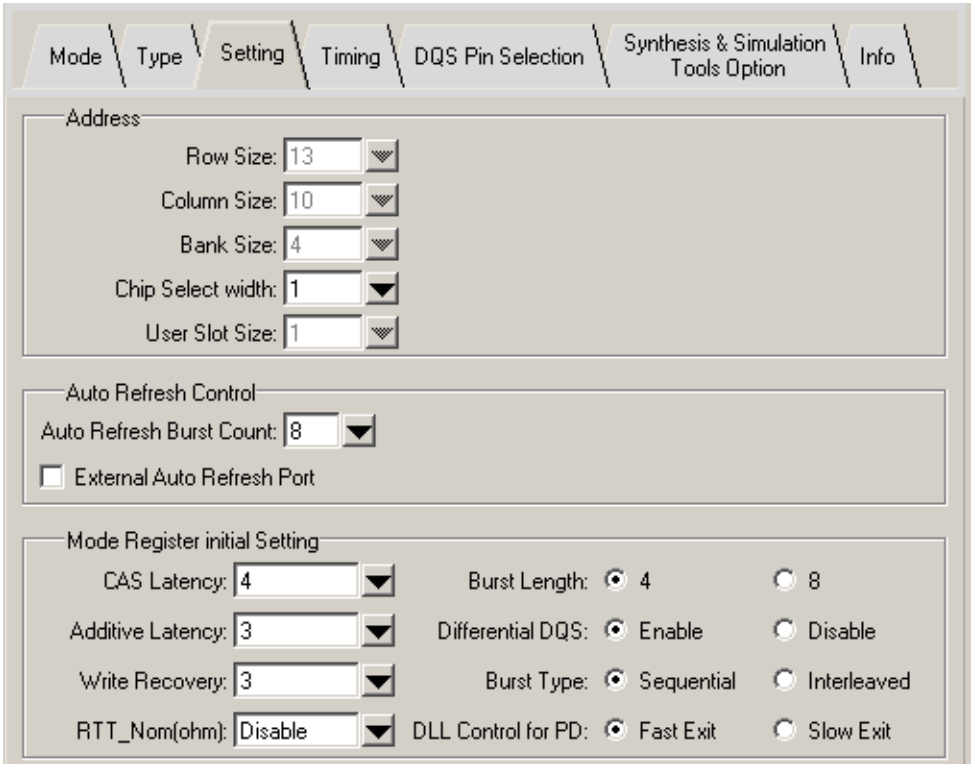
Use Differential DQS

This option will enable the Differential DQS pins in DDR2 IP. (The DDR IP does not have this feature). When checked, the DQS will be paired with complementary signals to provide differential pair signaling to the system during both reads and writes. When unchecked, the DQS will be used in single ended mode.

Setting Tab

The target memory size and addressing scheme are determined in the Setting tab. The memory initialization and auto-refresh configurations are also covered in this tab. Figure 3-4 shows the contents of the Setting tab.

Figure 3-4. Setting Tab



Row Size

This option indicates the row address size for the memory ranging from 13 to 16, which is found in the memory data sheet.

Column Size

This option indicates the column address size for the memory ranging from 9 to 11, which is found in the memory data sheet.

Bank Size

This option indicates the bank address size for the memory. Either 4 or 8 is selected depending on the size and type of the memory to be used with the core.

Chip Select Width

The Lattice memory controller cores follow the JEDEC specifications for DDR/DDR2 SDRAM memories supporting two different sets of chip select signaling. The DDR1 mode provides up to eight chip selects supporting up to four memory modules. The DDR2 mode provides up to four chip selects supporting up to two DDR2 memory modules. Note that this option does not indicate the number of memory modules but the number of actual chip select signals.

User Slot Size

This option indicates the number of physical DIMM or SODIMM slots. The information of user slot number is used for the memory controller core to properly drive the ODT (On-Die Termination) signals to the DDR2 memory mod-

ules. Since DDR memory does not have ODT, this option is available only to the DDR2 mode with a choice of one or two slots.

EMR Prog During Init (For DDR Only)

Once disabled, the core does not program the EMR during the initialization; it is programmed after the initialization process is complete. The default value for this option is Program.

Auto Refresh Burst Count

This option indicates the number of Refresh commands that the memory controller core generates at once. DDR memories have at least an 8-deep Refresh command queue following the JEDEC specification and Lattice DDR memory controller cores support up to eight Refresh commands in one burst. It is recommended that the maximum number be used if the DDR interface throughput is a major concern of the system. If it is set to 8, for example, the core will send a set of eight consecutive Refresh commands to the memory at once when it reaches the time period of the eight refresh intervals (t_{REFI} x 8). Bursting refresh cycles increases the DDR bus throughput because it helps keep core intervention to a minimum.

External Auto Refresh Port

This option provides users with the capability of controlling the memory refresh command generation. If this option is disabled, the core takes control of the Refresh command generation according to the memory timing parameter, TREFI. Once enabled, the core adds the external auto refresh control ports to the local user interface with which users can take full control of the Refresh command generation.

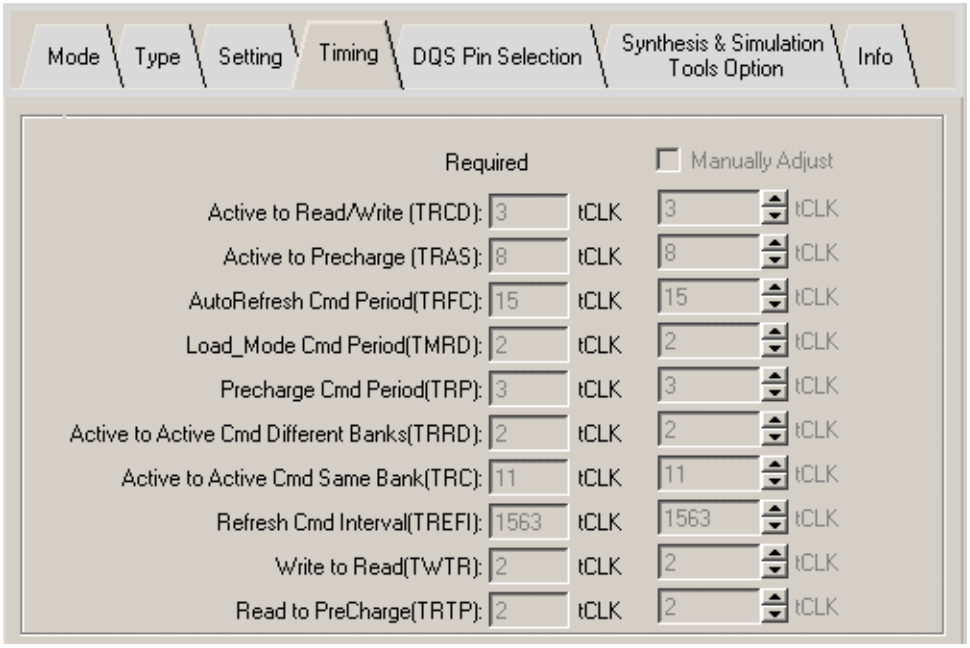
Mode Register Initial Setting

This option allows the user to program the mode registers during the core initialization process. Not all mode register bits are initialized from this option. Only the mode register configuration bits that are used for normal DDR operations are programmed using this setting. See Table 3-1 on page 19 for the list of the covered mode register settings. The user does not need to program the mode registers after the core initialization is finished if the mode register is properly configured as desired.

Timing Tab

If the user did not select a Micron memory listed in Select Memory dropdown in the Type tab of the IPexpress GUI, the DDR/DDR2 memory controller core allows the user to customize the memory timing parameters by selecting “Custom” in the Select Memory dropdown (refer to “Type Tab” on page 22). The "Manually Adjust" box in the Timing tab must be checked to adjust these timing parameters.Two timing parameters, TWTR and TRTP, are for the DDR2 mode only while all others are common to both modes. The numbers in the parameter boxes are decimal values indicating the number of clock cycles (t_{CLK}). Since the timing numbers available in the memory vendors’ data sheets usually are actual time based, conversions from time numbers to clock numbers should be properly made. The conversion is easily made by dividing the time number by the clock period. When a timing parameter is found to be a minimum value in the data sheet, the calculated number, if not a whole integer, should be the next whole integer to be safe. If it is a maximum value, then only the whole part is taken, and the decimal part is discarded. Figure 3-5 shows the contents of the Timing tab.

Figure 3-5. Timing Tab



The memory timing parameters are listed in Table 3-2.

Note: There is a timing parameter that is not shown in the Timing tab. The TCKP parameter is not a memory timing parameter but a memory controller core parameter used only in the DDR2 mode. It provides the wait cycles during the DDR2 memory initialization. The DDR2 specification requires a minimum of 400 ns wait before the PRE-CHARGE ALL command is executed. This parameter is found in the core parameter file with the default number `d107, which ensures 400ns of wait up to 266MHz speed. Although the wait time can be increased or decreased by adjusting the TCKP parameter, it may not be necessary to modify this parameter in most applications.

Table 3-2. Memory Timing Parameters for DDR Memory Controller

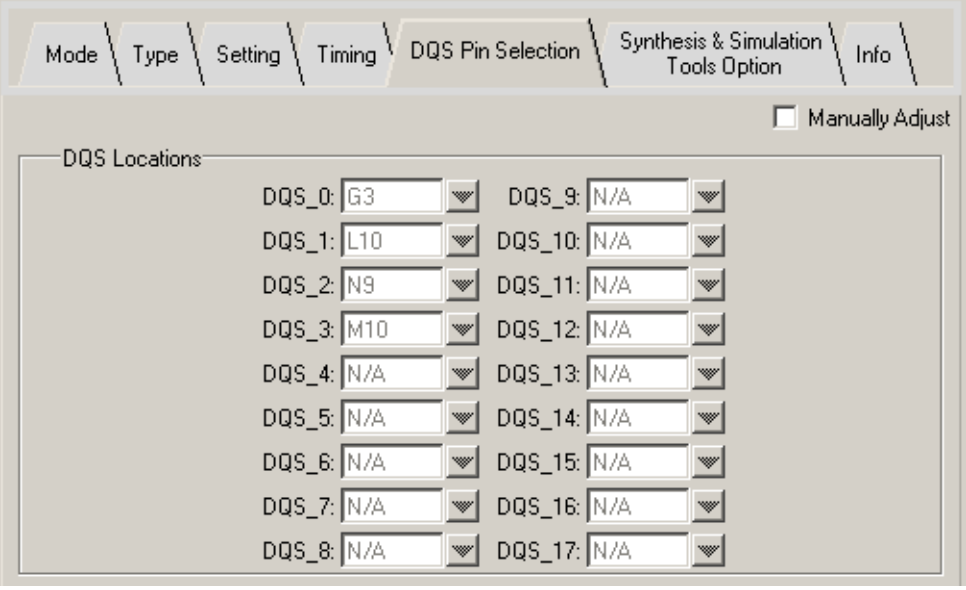
Signal Name	Description
tras[4:0]	ACTIVE to PRECHARGE command delay in clock cycles
trc[4:0]	ACTIVE to ACTIVE/AUTO REFRESH delay in clock cycles
trcd[2:0]	ACTIVE to READ/WRITE delay in clock cycles
trrd[2:0]	ACTIVE bank A to ACTIVE bank B delay in clock cycles
trfc[5:0]	REFRESH command period in clock cycles
trp[2:0]	PRECHARGE command period in clock cycles
tmrd[2:0]	LOAD MODE REGISTER command period in clock cycles
trefi[15:0]	Refresh Interval in clock cycles
trtp[1:0]	READ to PRECHARGE delay, DDR2 mode only
twtr[2:0]	WRITE to READ delay, DDR2 mode only
tckp[6:0][1]	Wait before PRECHARGE ALL during initialization, DDR2 mode only

1. Not available in the IPexpress GUI.

DQS Pin Selection Tab (DDR2 Only)

The DQS Pin Selection tab enables users to select DQS Pin locations of each device in DDR2 mode (for LatticeECP2/ECP2S/ECP2M/ECP2MS/ECP3/XP2 devices). When the Manually Adjust box is checked, the user can find and pick all available DQS Pins in the dropdown list. Figure 3-7 shows the contents of the DQS Pin Selection tab.

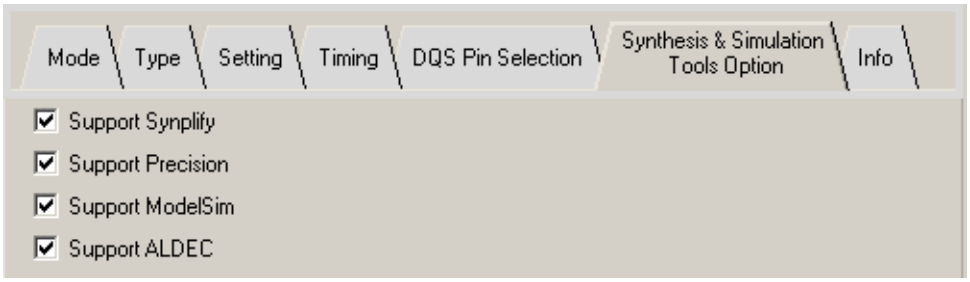
Figure 3-6. DQS Pin Selection Tab



Synthesis & Simulation Tools Option Tab

The Lattice DDR memory controller cores support multiple synthesis and simulation tool flows. This tab allows users to deselect the unwanted flow supports. Figure 3-7 shows the contents of the Synthesis & Simulation Tools Option tab.

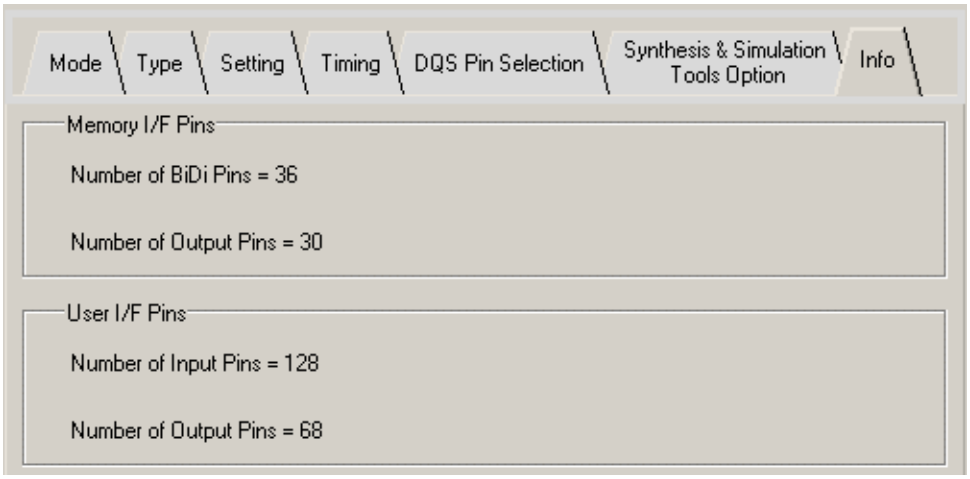
Figure 3-7. Synthesis & Simulation Tools Option Tab



Info Tab

The number of pins required on the DDR bus and the local user interface are reported in the Info tab. Figure 3-8 shows the contents of the Info tab.

Figure 3-8. Info Tab



Memory I/F Pins

The numbers displayed indicate the total required number of DDR bus I/O pads.

User I/F Pins

The numbers displayed indicate the total required number of local user interface signals. Although these signals usually do not use I/O pads in user applications, this information can indicate whether or not the evaluation project will insert the dummy logic. Note that all local user interface signals also use I/O pads in the core evaluation project.

IP Core Generation

This chapter provides information on licensing the DDR/DDR2 IP core, generating the core using the IPexpress tool, running functional simulation, and including the core in a top-level design. The Lattice DDR/DDR2 IP core can be used in LatticeECP3, LatticeECP2/M, LatticeXP2 and LatticeSC/M device families. The Lattice DDR IP core can be used in LatticeECP and LatticeXP device families.

Licensing the IP Core

An IP license is required to enable full, unrestricted use of the DDR/DDR2 IP core in a complete, top-level design. An IP license that specifies the IP core (DDR/DDR2) and device family (ECP3, ECP2/M, ECP, XP2, XP or SC/M) is required to enable full use of the DDR/DDR2 IP core in LatticeECP3, LatticeECP2/M, LatticeXP2 or LatticeSC/M devices and the use of the DDR IP core in LatticeECP or LatticeXP devices. Instructions on how to obtain licenses for Lattice IP cores are given at:

http://www.latticesemi.com/products/intellectualproperty/aboutip/isplevercoreonlinepurchas.cfm

Users may download and generate the DDR/DDR2 IP core and fully evaluate the core through functional simulation and implementation (synthesis, map, place and route) without an IP license. The DDR/DDR2 IP core also supports Lattice's IP hardware evaluation capability, which makes it possible to create versions of the IP core that operate in hardware for a limited time (approximately four hours) without requiring an IP license. See "Hardware Evaluation" on page 37 for further details. However, a license is required to enable timing simulation, to open the design in the Diamond or ispLEVER EPIC tool, and to generate bitstreams that do not include the hardware evaluation timeout limitation.

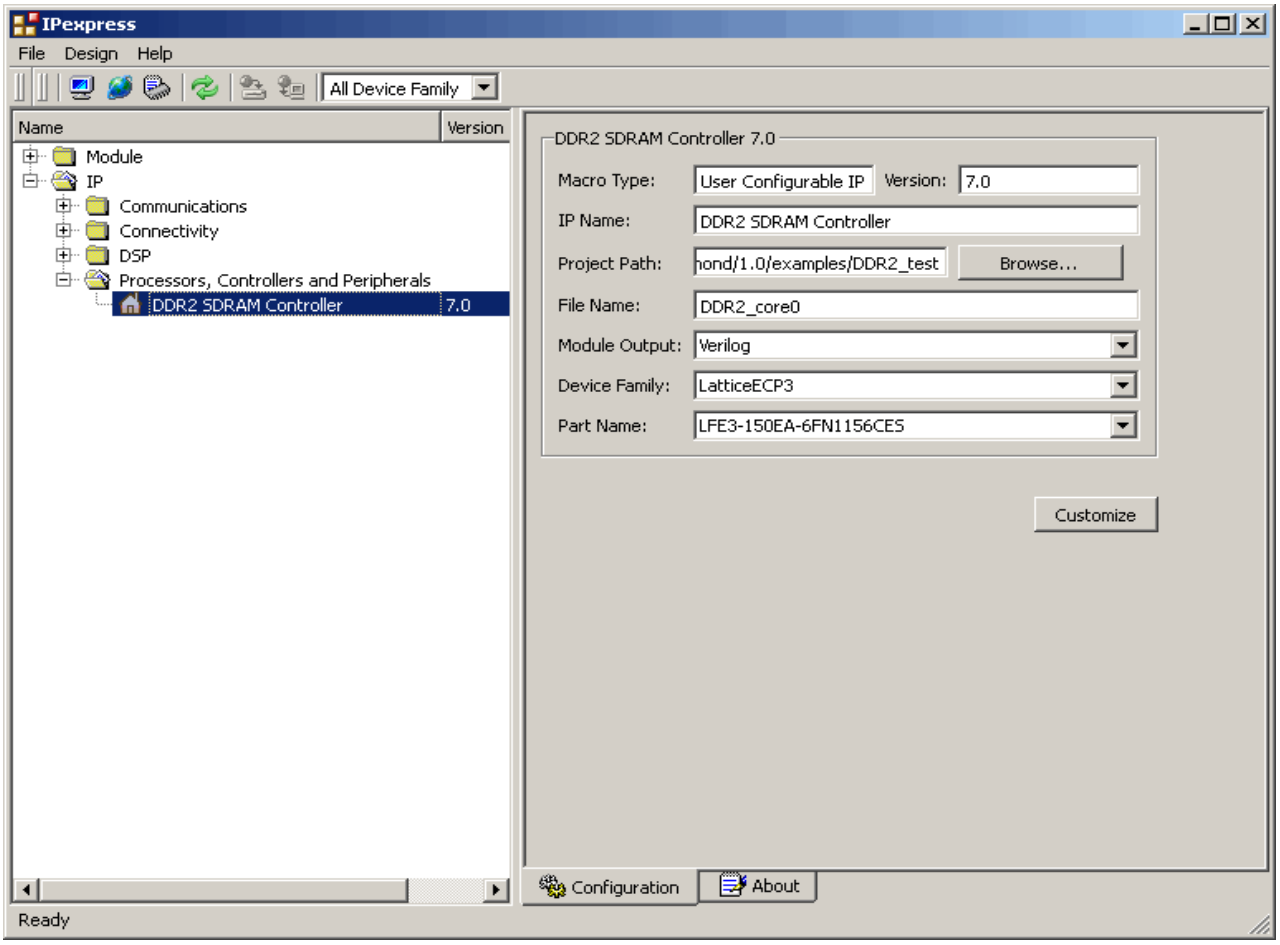
Getting Started

The DDR/DDR2 IP core is available for download from the Lattice's IP Server using the IPexpress tool. The IP files are automatically installed using ispUPDATE technology in any customer-specified directory. After the IP core has been installed, the IP core will be available in the IPexpress GUI dialog box shown in Figure 4-1.

The IPexpress tool GUI dialog box for the DDR/DDR2 IP core is shown in Figure 4-1. To generate a specific IP core configuration the user specifies:

- **Project Path** – Path to the directory where the generated IP files will be loaded.
- **File Name** – "username" designation given to the generated IP core and corresponding folders and files.
- **(Diamond) Module Output** – Verilog or VHDL.
- **(ispLEVER) Design Entry Type** – Verilog HDL or VHDL
- **Device Family** – Device family to which IP is to be targeted (e.g. LatticeSCM, Lattice ECP2M, LatticeECP3, etc.). Only families that support the particular IP core are listed.
- **Part Name** – Specific targeted part within the selected device family.

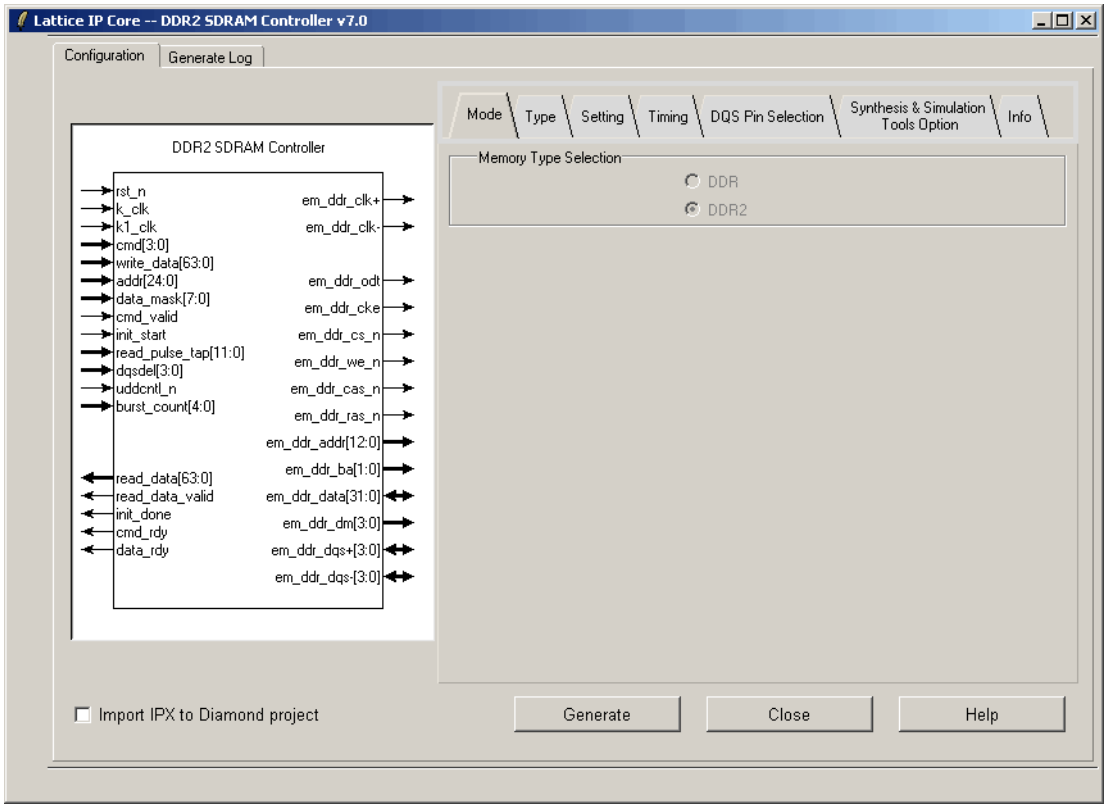
Figure 4-1. IPexpress Dialog Box (Diamond Version)



Note that if the IPexpress tool is called from within an existing project, Project Path, Design Entry, Device Family and Part Name default to the specified project parameters. Refer to the IPexpress tool online help for further information.

To create a custom configuration, the user clicks the **Customize** button in the IPexpress tool dialog box to display the DDR/DDR2 SDRAM IP core Configuration GUI, as shown in Figure 4-2. From this dialog box, the user can select the IP parameter options specific to their application. Refer to "Parameter Settings" on page 19 for more information on the DDR/DDR2 parameter settings.

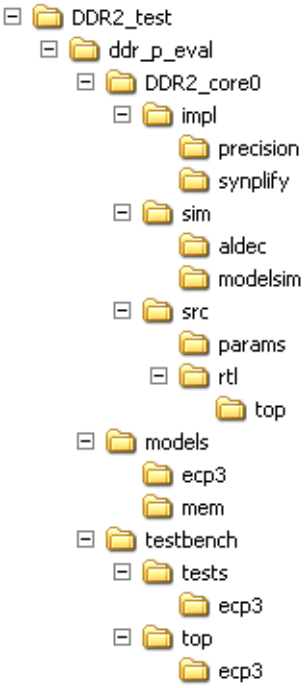
Figure 4-2. Configuration Dialog Box (Diamond Version)



IPexpress-Created Files and Top Level Directory Structure

When the user clicks the **Generate** button in the IP Configuration dialog box, the IP core and supporting files are generated in the specified “Project Path” directory. The directory structure of the generated files is shown in Figure 4-3.

Figure 4-3. LatticeECP3 DDR/DDR2 Core Directory Structure



Generated Files

This section describes the structure of the DDR/DDR2 memory controller core that is generated by IPexpress as per user configuration. It also explains how the generated files are used in the structure. Understanding the core structure is an important step of a system design using the core. The summary of the files of the core for simulation for DDR are listed in Table 4-1, and the summary of the files of the core for simulation for DDR2 are listed in Table 4-2.

Table 4-1. Files for Simulation and Implementation (DDR)

File	Location	Modules	S[1]	P[2]
Top-level wrapper	.\ddr_p_eval\[core_name]\src\rtl\top\[device]\	ddr_sdram_mem_top		X
Top-level wrapper	.\ddr_p_eval\[core_name]\sim	ddr_sdram_mem_top	X	
Encrypted netlist	.\	[core_name].ngo		X
Core header[3]	.\	[core_name]_bb.v		X
I/O modules	.\ddr_p_eval\models\[device]\	ddr_sdram_mem_io_top and its submodules	X	X
Clock generator	.\ddr_p_eval\models\[device]\	pmi_pll_fp	X	X
Parameter file	.\ddr_p_eval\[core_name]\src\params\	ddr_sdram_mem_params	X	X
Preference files[4]	.\ddr_p_eval\[core_name]\impl\[synthesis]\	[core_name]_eval.lpf post_route_trace.prf		X
Evaluation project (GUI)[4]	.\ddr_p_eval\[core_name]\impl\[synthesis]\	[core_name]_eval.syn		X
Evaluation project (Command-line)[4]	.\ddr_p_eval\[core_name]\impl\synplify\syn\ .\ddr_p_eval\[core_name]\impl\synplify\par\	runsyn_[core_name]_top.cmd runpar_[core_name]_top.cmd		X
Testbench top	.\ddr_p_eval\testbench\top\[device]\	test_mem_ctrl	X	
Obfuscated core simulation model	.\	[core_name]_beh	X	
Stimulus generator	.\ddr_p_eval\testbench\tests\[device]\	cmd_gen, test_case	X	
Monitor	.\ddr_p_eval\testbench\top\[device]\	monitor, odt_watchdog	X	

Table 4-1. Files for Simulation and Implementation (DDR) (Continued)

File	Location	Modules	S[1]	P[2]
TB configuration parameter	.\ddr_p_eval\testbench\tests\[device]\	tb_config_params	X	
Memory model	.\ddr_p_eval\models\mem\	ddr1 (with DIMM modules)	X	
Memory model parameter	.\ddr_p_eval\models\mem\	ddr1_parameters.vh	X	
Evaluation script[4]	.\ddr_p_eval\[core_name]\sim\modelsim\ .\ddr_p_eval\[core_name]\sim\aldec\	[core_name]_eval.do	X	
Simulation script[4]	.\ddr_p_eval\[core_name]\sim\modelsim\ .\ddr_p_eval\[core_name]\sim\aldec\	[core_name]_eval_timing_precision.do [core_name]_eval_timing_synplify.do	X	

1. S = Simulation.
2. P = Synthesis/Place and Route.
3. Not required for the VHDL flow.
4. Files are generated according to the Synthesis & Simulation Tools Option tab selection. See "Synthesis & Simulation Tools Option Tab" on page 27.

Table 4-2. Files for Simulation and Implementation (DDR2)

File	Location	Modules	S[1]	P[2]	R[3]
Top-level wrapper	.\ddr_p_eval\[core_name]\src\rtl\top\[device]\	ddr_sdram_mem_top	X		
Top-level wrapper	.\ddr_p_eval\[core_name]\impl	ddr_sdram_mem_top		X	
Encrypted netlist	.\	[core_name].ngo		X	
Core header[4]	.\	[core_name]_bb.v		X	
Instantiation template	.\	[core_name]_inst.v			X
I/O modules	.\ddr_p_eval\models\[device]\	ddr_sdram_mem_io_top and its submodules			X
Clock generator	.\ddr_p_eval\models\[device]\	pmi_pll_fp(ECP2(S)/ECP2M(S)/ ECP3/XP2) ddr_pll90(SC/SCM)	X	X	
Parameter file	.\ddr_p_eval\[core_name]\src\params\	ddr_sdram_mem_params	X	X	
Preference files[5]	.\ddr_p_eval\[core_name]\impl\[synthesis]\	[core_name]_eval.lpf post_route_trace.prf		X	
Evaluation project (GUI)[5]	.\ddr_p_eval\[core_name]\impl\[synthesis]\	[core_name]_eval.syn		X	
Testbench top	.\ddr_p_eval\testbench\top\[device]\	test_mem_ctrl	X		
Obfuscated core simulation model	.\	[core_name]	X		
Stimulus generator	.\ddr_p_eval\testbench\tests\[device]\	cmd_gen, test_case	X		
Monitor	.\ddr_p_eval\testbench\top\[device]\	monitor, odt_watchdog	X		
TB configuration parameter	.\ddr_p_eval\testbench\tests\[device]\	tb_config_params	X		
Memory model	.\ddr_p_eval\models\mem\	ddr2,(plus with DIMM modules)	X		
Memory model parameter	.\ddr_p_eval\models\mem\	ddr2_parameters.vh	X		
Evaluation script[5]	.\ddr_p_eval\[core_name]\sim\(modelsim|aldec)\	[core_name]_eval.do	X		
Simulation script[5]	.\ddr_p_eval\[core_name]\sim\(modelsim|aldec)\	[core_name]_eval_gatesim_precision.do [core_name]_eval_gatesim_synplify.do	X		

1. S = Simulation.
2. P = Synthesis/Place and Route.
3. R = Not used in Simulation/Synthesis/Place and Route, only for reference.
4. Not required for the VHDL flow.
5. Files are generated according to the Synthesis & Simulation Tools Option tab selection. See "Synthesis & Simulation Tools Option Tab" on page 27.

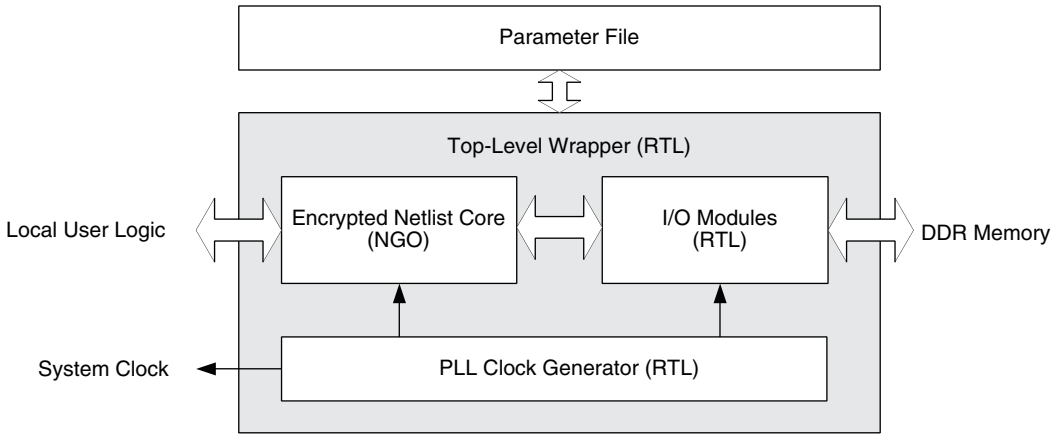
DDR Memory Controller Core Structure

The DDR memory controller core consists of the following five major functional blocks:

- Top-level wrapper (RTL)
- Encrypted memory controller block (NGO)
- I/O module block (RTL)
- Clock generator (RTL)
- Parameter file

All of these blocks are required to implement the core on the target FPGA device. Figure 4-4 shows the interconnection among those blocks.

Figure 4-4. Structure of DDR Memory Controller Core



Top-level Wrapper

The encrypted netlist core, I/O modules, and the clock generator blocks are instantiated in the top-level wrapper. When a system design is made with the Lattice DDR memory controller core, this wrapper must be instantiated. The wrapper is fully parameterized by the generated parameter file.

Encrypted Netlist

The encrypted netlist contains the memory controller function that interfaces with the local user logic and the I/O modules that communicate with the DDR memory. The encrypted netlist must be located in the implementation project directory. IPexpress may generate another netlist for a PMI function when the core is generated. If this is the case, the PMI netlist must also be present along with the core netlist. The name of the PMI netlist is determined by IPexpress with the “pmi_xx..xx.ngo” form.

I/O Modules

The I/O module block provides device dependant DDR I/O functions. This block consist of one I/O module top file and several sub-modules that handle the DDR data (DQ), data mask (DM) and data strobe (DQS) signals. Note that the I/O modules are integrated into an NGO block when the core is generated for the VHDL flow. The simulation will continue to use the Verilog RTL modules to model the I/O Modules block behavior.

Clock Generator

The DDR memory controller core is designed to provide the system clock from the inside of the core. The clock output (k_clk) from the clock generator is used to drive the whole core logic as well as the external user logic. If a system that uses the DDR memory controller core is required to have a clock generator that is external to the core, the incorporated clock generator block can be removed from the core. The connections between the top-level wrapper

and the clock generator are fully RTL based, and therefore, it is possible to modify the structure and connection of the core for the clock distribution following the system's need.

Parameter File

The IPexpress tool generates the parameter file based on the selected user options. The parameter file parameterizes the top-level wrapper and I/O modules. Note that the encrypted netlist (.ngo) file is created using the generated parameter file but is not a parameterized module. Therefore, the parameter definitions must not be altered. Otherwise, there will be connection problems between the netlist and other parameterized RTL modules.

Core Header File

The encrypted netlist is regarded as a black box during synthesis in the Verilog design environment. The header file that represents the netlist module must be included to bind the netlist to the wrapper in the Verilog flow. This file has a suffix "_bb" following the core name and is required only for the synthesis process.

Preference Files

The generated core contains two sets of preference files. One set is for the Synplify synthesis flow, while the other is for the Precision synthesis flow. Each set has two preference files. The implementation preference file ([core_name]_eval.lpf) contains a complete set of timing and physical preferences to force the implementation software to get a better performance margin. The trace preference file (post_route_trace.prf) is used to validate the timing results after the implementation is completed.

Refer to "Core Implementation" on page 39 for more information about understanding preferences, preference localization, VREF assignments, DLL allocation, I/O types for DDR, skew treatment, data valid generation, dummy logic removal, read data auto-alignment logic, PCB routing delay compensation, and DQS_PIO_READ locate constraints.

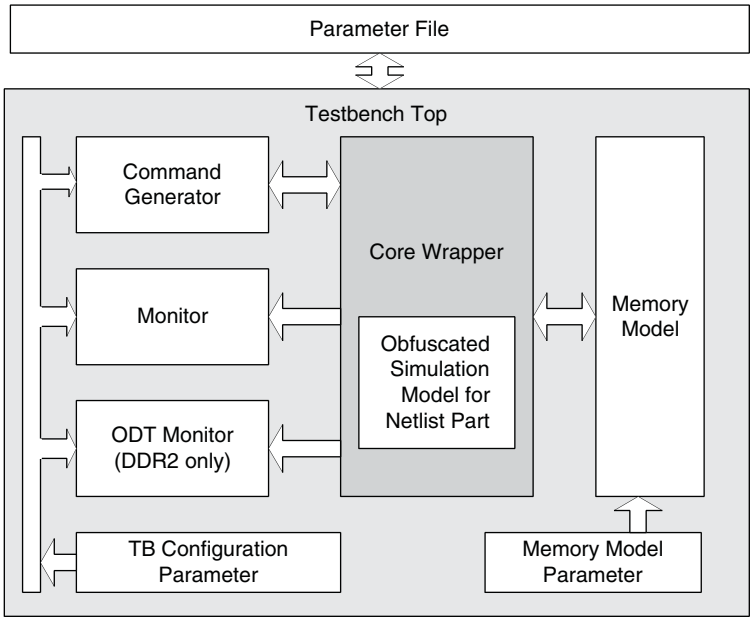
Evaluation Project Files

Several project files for implementation of the IP core are included for instant evaluation of the implementation result. A project file for Project Navigator is provided for the GUI-based flow, while a synthesis command script and a Place and Route (PAR) command script are included for the evaluation with the command-line flow. All required files for synthesis and PAR processes are imported into the project files. These project files can be used as a starting point of a user application design.

Simulation Files for Core Evaluation

Once a DDR memory controller core is generated, it contains a complete set of testbench files that can be used to simulate some core activities for evaluation. This simulation structure for the DDR memory controller core is shown in Figure 4-5. This structure can be reused by system designers to accelerate their system validation. When a DDR memory controller core is simulated in VHDL, the core wrapper is provided in VHDL while other parts of the simulation structure are still in Verilog. Therefore, a simulation tool that has the mixed language capability such as the full version of ModelSim or an Aldec HDL simulator is required.

Figure 4-5. Simulation Structure for DDR Memory Controller Core Evaluation



Testbench Top

The testbench top includes the core under test, memory model, stimulus generator and monitor blocks. It is parameterized by the core parameter file.

Obfuscated Core Simulation Model

The simulation model for the netlist part of the core is provided in the form of obfuscated RTL. This core model represents the functionality of the encrypted netlist and must be included in the simulation that contains the memory controller core.

Command Generator

The command generator generates stimuli for the core. The core initialization and command generation activities are predefined in the provided test case module. It is possible to customize the test case module to see the desired activities of the core

Monitor

The monitor blocks monitor both the local user interface and DDR interface activities and generate a warning or an error if any violation is detected. It also validates the core data transfer activities by comparing the read data with the written data.

TB Configuration Parameter

The TB configuration parameter provides the parameters for testbench files. These parameters are derived from the core parameter file and are not required to configure them separately. For those users who need a special memory configuration, however, modifying this parameter set might provide a support for the desired configuration.

Memory Model

The DDR memory controller core contains a bus functional memory simulation model provided by one of the most popular memory vendors. If a different memory model is required, it can be done by simply replacing the instantiation of the model from the memory configuration modules located in the same folder.

Memory Model Parameter

This memory parameter file comes with the bus functional memory simulation model. It contains the parameters that the memory simulation model needs. It is not necessary for users to change any of these parameters.

Evaluation Script File

The functional and timing simulation macro script files are included for instant evaluation of the core. All required files for simulation are included in the macro script. These simulation scripts can be used as a starting point of a user simulation project. The generated scripts are based on the selection in the Synthesis & Simulation Tool Option tab (see "Synthesis & Simulation Tools Option Tab" on page 27).

Hardware Evaluation

The DDR/DDR2 IP core supports Lattice's IP hardware evaluation capability, which makes it possible to create versions of the IP core that operate in hardware for a limited period of time (approximately four hours) without requiring the purchase of an IP license. It may also be used to evaluate the core in hardware in user-defined designs.

Enabling Hardware Evaluation in Diamond

Choose **Project > Active Strategy > Translate Design Settings**. The hardware evaluation capability may be enabled/disabled in the Strategy dialog box. It is enabled by default.

Enabling Hardware Evaluation in ispLEVER

In the Processes for Current Source pane, right-click the **Build Database** process and choose **Properties** from the dropdown menu. The hardware evaluation capability may be enabled/disabled in the Properties dialog box. It is enabled by default.

Updating/Regenerating the IP Core

By regenerating an IP core with the IPexpress tool, you can modify any of its settings including: device type, design entry method, and any of the options specific to the IP core. Regenerating can be done to modify an existing IP core or to create a new but similar one.

Regenerating an IP Core in Diamond

To regenerate an IP core in Diamond:

1. In IPexpress, click the **Regenerate** button.
2. In the Regenerate view of IPexpress, choose the IPX source file of the module or IP you wish to regenerate.
3. IPexpress shows the current settings for the module or IP in the Source box. Make your new settings in the T**arget** box.
4. If you want to generate a new set of files in a new location, set the new location in the **IPX Target File** box. The base of the file name will be the base of all the new file names. The IPX Target File must end with an .ipx extension.
5. Click **Regenerate.** The module's dialog box opens showing the current option settings.
6. In the dialog box, choose the desired options. To get information about the options, click **Help**. Also, check the About tab in IPexpress for links to technical notes and user guides. IP may come with additional information. As the options change, the schematic diagram of the module changes to show the I/O and the device resources the module will need.
7. To import the module into your project, if it's not already there, select **Import IPX to Diamond Project** (not available in stand-alone mode).
8. Click **Generate**.

9. Check the Generate Log tab to check for warnings and error messages.

10. Click **Close**.

The IPexpress package file (.ipx) supported by Diamond holds references to all of the elements of the generated IP core required to support simulation, synthesis and implementation. The IP core may be included in a user's design by importing the .ipx file to the associated Diamond project. To change the option settings of a module or IP that is already in a design project, double-click the module's .ipx file in the File List view. This opens IPexpress and the module's dialog box showing the current option settings. Then go to step 6 above.

Regenerating an IP Core in ispLEVER

To regenerate an IP core in ispLEVER:

1. In the IPexpress tool, choose **Tools > Regenerate IP/Module**.
2. In the Select a Parameter File dialog box, choose the Lattice Parameter Configuration (.lpc) file of the IP core you wish to regenerate, and click **Open**.
3. The Select Target Core Version, Design Entry, and Device dialog box shows the current settings for the IP core in the Source Value box. Make your new settings in the Target Value box.
4. If you want to generate a new set of files in a new location, set the location in the LPC Target File box. The base of the .lpc file name will be the base of all the new file names. The LPC Target File must end with an .lpc extension.
5. Click **Next**. The IP core's dialog box opens showing the current option settings.
6. In the dialog box, choose desired options. To get information about the options, click **Help**. Also, check the About tab in the IPexpress tool for links to technical notes and user guides. The IP core might come with additional information. As the options change, the schematic diagram of the IP core changes to show the I/O and the device resources the IP core will need.
7. Click **Generate**.
8. Click the **Generate Log** tab to check for warnings and error messages.

This chapter provides application support information for the DDR/DDR2 IP core.

Core Implementation

This section describes the major factors that are important for a successful DDR memory controller implementation. The LatticeSC/M devices have a different DDR I/O structure, and the descriptions shown here are not applicable to them. See TN1099, *LatticeSC DDR/DDR2 SDRAM Memory Interface User's Guide,* for the LatticeSC/M implementation issues.

Understanding Preferences

The following preferences are found in the provided logical preference files (.lpf):

- **FREQUENCY**
  The DDR memory controller core is normally 10% over-constrained for obtaining optimal f_{MAX} results. The post-route trace preference file contains the preferences that have the real performance targets, and it should be used to validate the timing results.
- **MAXDELAY NET**
  The MAXDELAY NET preference ensures that the net for the READ input to the DQSBUF block has a minimal net delay and falls within the data valid clearing window. Since it is highly over-constrained, the post-route trace preference file should be used to validate the timing results.
- **MULTICYCLE / BLOCK PATH**
  These preferences are used to avoid an overruled performance report from the static timing results. They are not considered critical in terms of the core operability but still important for a correct static timing report.
- **IOBUF**
  The IOBUF preference assigns the required I/O types to the DDR I/O pads. See "I/O Types for DDR" on page 41 for details.
- **LOCATE**
  Only the em_ddr_dqs pads are located in the provided preference file per user selection. Note that not all I/O pads can be associated with a DQS (em_ddr_dqs) pad in a bank. Since there is a strict DQ-to-DQS association rule in each Lattice FPGA device, it is strongly recommended the DQ-to-DQS associations of the selected pin-outs be validated using the implementation software before the PCB routing task is started. The DQ-to-DQS pad associations for a target FPGA device can be found in the datasheet or pinout table of the target device.

  – In DDR1 mode, the user needs to edit the DQS pad locations in the provided preference file manually.
  – In DDR2 mode, the DQS pad locations selected in the GUI will appear in the provided preference file accordingly.

  Refer to "DQS_PIO_READ Locate Constraints" on page 44 for the procedure to locate DQS_PIO_READ pgroups for DDR2 IP.

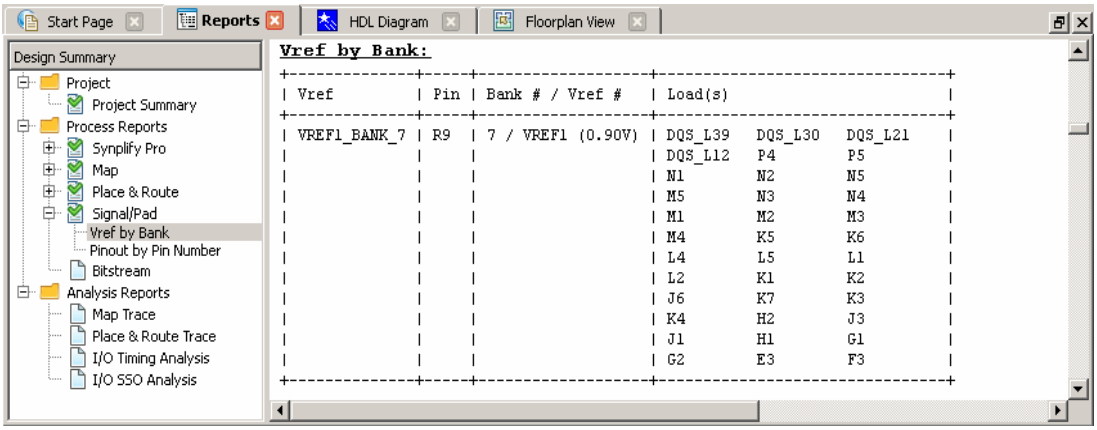
Preference Localization

Due to the nature of high-speed DDR operations, some of the internal nets must be constrained in order to achieve the functional and performance goal. However, the hierarchy structure and name of an internal net is subject to change when there are changes in the design or when a different version of a synthesis tool is used. It is the user's responsibility to track these changes and update them in the preference file. Since the FREQUENCY, MAXDELAY NET and LOCATE PGROUP preferences affect the functionality and performance of the core, it is good to pay close attention to tracking them after each run of the synthesis process. The updated net and path names can be found in the map report file (.mrp).

VREF Assignments

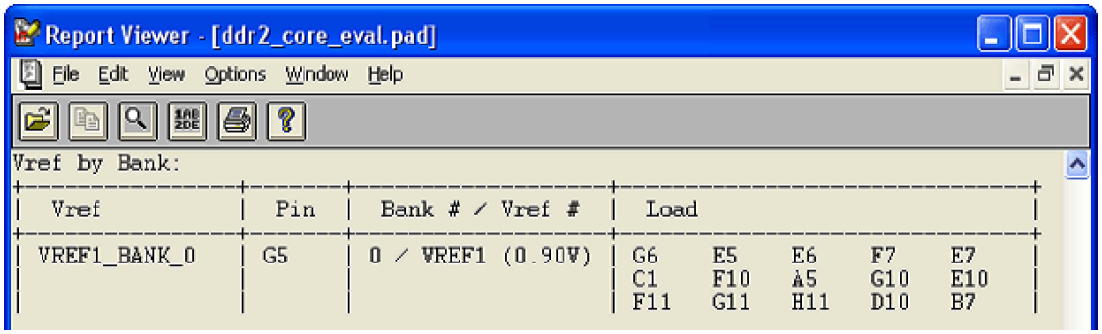
An SSTL I/O type pad requires a reference voltage input when it is operating as a receiving end. In the DDR design in an FPGA device, data and data strobe signals are bi-directional, and each of the banks that contain these bi-directional DDR interface signals must have a connection to the external reference voltage resource. Otherwise, the proper input level will not be detected. This can be done by connecting the VREF1 pad of the bank to the external reference voltage source. Note that when a bank has two VREF pads, only the VREF1 is used for memory DDR applications. The other reference voltage pad, VREF2, should not be tied to the same power rail of VREF1, otherwise the default pull-up on the VREF2 will corrupt the VREF1's voltage. The VREF1 pad and its associated DDR input or bi-directional pads are listed in the pad report file (.pad). An example of the PAD report file in the Diamond software is shown in the example in Figure 5-1.

Figure 5-1. Example of VREF1 Connection Report in Diamond (DDR2)



An example of the PAD report file in the ispLEVER software is shown in the example in Figure 5-2.

Figure 5-2. Example of VREF1 Connection Report in ispLEVER (DDR2)



DLL Allocation

Lattice FPGA devices have dedicated DDR register structures in the input and output for read and write operations. The DQS delay block is required in order to correctly capture data at the input register. Since the data strobe signal (DQS) from the DDR memory is not free-running, a calibrated DLL function is required to precisely delay the incoming DQS. The DQSDLL block is used to generate the delay value on the dqs_del signal. The DQSBUFx block uses dqs_del to generate the 90-degree shifted DQS signal for read operations. Accuracy of this delay is crucial to maximize the capturing window for the read data. The calibration bus, UDDCNTL, controls the update and hold functions of the DLL to compensate for temperature, voltage and process variations. The DQSDLL block is updated when the core is not in the read mode. Note that UDDCNTL is an active-low update enable signal.

The FPGA device has two DQSDLLs on opposite sides of the device. Each DLL compensates DQS delays in its half of the device. The LatticeECP/EC and LatticeXP families have them in the top and bottom places, supporting

one-half in the top banks and the other half in the bottom banks. The LatticeECP2/M, LatticeECP3, and LatticeXP2 families have one in the left side and the other in the right side of the chip. If a user system design requires that DDR data pads be placed across this boundary, it will require both DQSDLL blocks used. When a DDR memory controller is generated, IPexpress instantiates either one or two DQSDLL blocks, depending on the user configuration and the target device size. When the final pinouts are determined and require the relocations of the DQS groups from the original locations in the preference file, each connection from a DQS group to a DQSDLL block must be properly re-established in the top-level wrapper in order to avoid the routing violations. (In DDR2 mode, the DQSDLL block and all related connections are automatically implemented, the re-establishment is not necessary). For example, the final pinouts may require the use of both DQSDLL blocks, if the DQ pads are to be placed across the DLL boundary while the original core uses only one. The code below is an excerpt in Verilog from the top-level wrapper that addresses the DLL allocation. When both DLLs are to be used, IPexpress enables the use of two DLLs as shown in the code. This example shows that only one DLL is used by IPexpress. The DLL allocations are initially made by IPexpress as shown in lines 11 through 18 from the code. If the final DDR pin locations require use of both DLLs, the connection from each DQS group to a DLL should be reallocated by reassigning the lines 11 through 18 according to the pinout requirements.

```
 1: `ifdef USE_TWO_DLL
 2: DQSDLL U0_DQSDLL (.CLK(k_clk), .RST(rst_acth), .UDDCNTL(~update_cntl),
 3:                   .DQSDEL(dqsdel_0), .LOCK());
 4: DQSDLL U1_DQSDLL (.CLK(k_clk), .RST(rst_acth), .UDDCNTL(~update_cntl),
 5:                  .DQSDEL(dqsdel_1), .LOCK());
 6: `else
 7: DQSDLL U0_DQSDLL (.CLK(k_clk), .RST(rst_acth), .UDDCNTL(~update_cntl),
 8:                   .DQSDEL(dqsdel_0), .LOCK());
 9: `endif
10:
11: assign dqsdel[0] = dqsdel_0;
12: assign dqsdel[1] = dqsdel_0;
13: assign dqsdel[2] = dqsdel_0;
14: assign dqsdel[3] = dqsdel_0;
15: assign dqsdel[4] = dqsdel_0;
16: assign dqsdel[5] = dqsdel_0;
17: assign dqsdel[6] = dqsdel_0;
18: assign dqsdel[7] = dqsdel_0;
```

The following lines show a reassignment example when the DQS4, DQS5, DQS6 and DQS7 are located in the other side.

```
11: assign dqsdel[0] = dqsdel_0;
12: assign dqsdel[1] = dqsdel_0;
13: assign dqsdel[2] = dqsdel_0;
14: assign dqsdel[3] = dqsdel_0;
15: assign dqsdel[4] = dqsdel_1;
16: assign dqsdel[5] = dqsdel_1;
17: assign dqsdel[6] = dqsdel_1;
18: assign dqsdel[7] = dqsdel_1;
```

For systems that need two DDR memory controllers, follow the rule that one core is implemented in one half so that the other one can take the other half without crossing the DLL support boundary.

I/O Types for DDR

In the DDR1 mode, the SSTL25_II I/O type is used for all the DDR interface signals except the memory clock pads, em_ddr_clk, which need a differential I/O type, SSTL25D_II. When a DDR2 memory controller core is generated, both em_ddr_clk and em_ddr_dqs take the SSTL18D_II I/O type by default. Since the DQS mode is programmable

in DDR2 memories, the I/O type for em_ddr_dqs can be easily replaced with the single-ended type, SSTL18_II, in the preference file. All other DDR2 interface pads use SSTL18_II.

Note that the local interface signals in the generated core are also assigned with the same I/O type as the DDR interface signals by default. Since the local interface signals are normally embedded inside the FPGA once a system-level design is completed, the IOBUF preferences for them should be removed to avoid unnecessary preference warnings. If any of the local interface signals need to take the I/O pad including the clock and reset inputs, a proper I/O type for the signal must be selected to comply with the system requirement.

Skew Treatment

Lattice DDR memory controller is designed to use dedicated DDR I/O registers in order to minimize the skew among the DDR data and data strobe signals. Excessive skew between any two DDR data signals can be a major contributor to performance degradation. The skew of the DDR control signals among them and with respect to the DDR data is also crucial for high-speed implementation. The best way to minimize the skew of the DDR address and control signals is to implement all of them into the PIO registers instead of taking the registers inside the FPGA fabric.

The IPexpress tool inserts the synthesis directives to push out those signals into the PIOs in the top-level wrapper when the core is generated. The implementation results can be found in the Design Summary section of the map report, which shows whether those signals are inside the PIO or not. The required I/O resource for each DDR interface signal is listed in Table 5-1. The table includes both the PIO and the dedicated DDR register resources. If the PIO register number in the map report matches the total number of PIO registers calculated from the table, all of them are properly implemented into the PIO registers. The utilized IDDR/ODDR and PIO resources are reported separately in the Design Summary section.

Table 5-1. Required I/O Registers for Each DDR Interface Signals

DDR Pad	ODDR	IDDR	PIO Register	Total Required
em_ddr_dq	2	1	-	DATA_WIDTH x 3
em_ddr_dm	1	-	-	DATA_WIDTH / 8
em_ddr_dqs	2 (4)	1 (2)	-	DQS_WIDTH x 3
em_ddr_clk	1 (2)	-	-	CLK_WIDTH
em_ddr_odt	1	-	-	CS_WIDTH
em_ddr_addr	-	-	1	ROW_WIDTH
em_ddr_ba	-	-	1	BNK_WDTH
em_ddr_ras	-	-	1	1
em_ddr_cas	-	-	1	1
em_ddr_we	-	-	1	1
em_ddr_cs	-	-	1	CS_WIDTH
em_ddr_cke	-	-	1	CKE_WIDTH

Note: The numbers in parenthesis indicates that a differential pair is used. These are not included in the reported total.

Data Valid Generation

The read_data bus in the local user interface provides the read data from the memory to the user logic. The data on this bus is valid only while the read_data_valid signal is asserted. The timing of this validation is determined either by core logic or by a dedicated hardware block, depending on the target device. The LatticeECP and LatticeXP families use the LUT-based data valid generation logic inside the core, while the LatticeECP2/M and LatticeXP2 families utilize the dedicated hardware block for the data valid generation. The parameter file generated by IPexpress automatically includes the IO_DATA_VAL parameter to enable the use of the hardware block when the target device supports this hardware feature. The data valid generation logic is considered sensitive in terms of the compatibility of the given memory device or module. The implementation of this block should be carefully made

when the LUT-based logic is to be used. When a core that uses the LUT-based logic is generated, the IPexpress tool creates the preference file, which includes the LOCATE preferences for those blocks to the closest locations of the corresponding DQS pad as shown in the following example below:

```
LOCATE COMP  "em_ddr_dqs_0"  SITE  "PL16A";

LOCATE PGROUP
"U1_ddr_sdram_mem_io_top/U1_ddr_dqs_io/u_0__bidi_dqs/U1_pio_dvalid_gen/u1_data_
valid_macro1/p_data_valid_macro1" SITE "R16C2D";

LOCATE PGROUP
"U1_ddr_sdram_mem_io_top/U1_ddr_dqs_io/u_0__bidi_dqs/U1_pio_dvalid_gen/u1_data_
valid_macro2/p_data_valid_macro2" SITE "R16C3D"
```

The data_valid_macro1 and data_valid_macro2 blocks are defined in the core as physical groups. The above example shows that the DQS pad is located to “PL16A” and two physical groups are located to its closest slices, which are “R16C2” and “R16C3”. It ensures that the internal net delays inside and between two blocks are minimal, keeping the shortest routing distance to the originating DQS pad. Pay attention to the path hierarchies for the macro blocks because they are subject to change, as mentioned in the Preference Localization section. The up-to-date PGROUP paths are found in the schematic section of the processed preference file.

Note that the core that has the hardware data valid generation can be switched to the LUT-based logic by removing the IO_DATA_VAL parameter from the parameter file. The core does not have to be regenerated. When a memory turns out to be incompatible (although it is rarely happening), switching to the LUT-based way might provide a resolution because the timing characteristics of the hardware-based approach are fixed. Note that the IPexpress tool still includes the LOCATE PGROUP preferences but comments them out for future reference in case the switching is necessary. The LatticeECP3 families always use only the hardware block to generate the data valid signal.

Dummy Logic Removal

When a DDR IP core is generated, IPexpress assigns all the signals from both the DDR and local user interfaces to the I/O pads. The number of user interface signals is normally more than four times than that of the DDR interface. It makes the core impossible to be evaluated if the selected device does not have enough I/O pad resource. To facilitate core evaluation with smaller package devices, IPexpress inserts dummy logic to decrease the I/O pad counts by reducing the local read_data and write_data bus sizes by half. With the dummy logic, a core can be successfully evaluated even with smaller pad counts. The PAR process can be completed without a resource violation so that one can evaluate the performance and utilization of the core. However, the synthesized netlist will not function correctly because of the inserted dummy logic. The core with dummy logic, therefore, must be used only for evaluation purposes. The dummy logic parameter, DUMMY_LOGIC, is inserted in the top-level wrapper file if the generated core needs the dummy logic.

After a backend user logic design is attached to the core, most of the user interface signals are embedded. It is important to know that the DUMMY_LOGIC parameter should be removed from the top-level wrapper file before synthesizing the project. Another option is to use the simulation top-level wrapper file for synthesis.

Read Data Auto-Alignment Logic

Lattice FPGA devices have a dedicated DDR support circuitry that allows reliable capture of the read data from each DQS group with respect to the internal core clock. Because of possible PCB trace length differences among all DQS groups, the captured data from a DQS group may not be aligned with the ones from other DQS groups by one clock cycle. The data alignment logic in the DDR memory controller automatically aligns the read data received from the multiple DQS groups and presents it on the user interface.

PCB Routing Delay Compensation

After a read burst operation is completed, the read data valid generation logic must be initialized before the current read burst operation is started. The memory controller uses the incoming DQS signal (dqsi) from the memory for

this operation. The valid timing window of this initialization is strictly defined to be within the preamble period ($t_{PREAMBLE}$). The DQS signal from the memory arrives after the round-trip delay that includes the following delay factors:

- Memory clock output delay from FPGA
- Memory clock travel delay from FPGA to memory through the routed PCB lines
- Memory internal delay from clock to DQS
- DQS travel delay back to FPGA
- FPGA setup delay to the data valid generation logic

Since the timing calculation software cannot know the delays added on the PCB lines, they must be manually compensated in order to guarantee high performance read operations.

The Lattice DDR memory controller provides the read_pulse_tap port to compensate the round-trip delay.

Setting read_pulse_tap

Each DQS group (associated with either 4 DQs or 8 DQs) has its own read pulse tap setting because the PCB routing delay cannot be the same for all DQS groups. If a core is configured as 64-bit with eight DQS groups, for example, the total size of read_pulse_tap port becomes 24 bits (8 groups x 3 bits each).

The Lattice DDR memory controller is designed to be operated with the read_pulse_tap = 000 setting for most cases that use reasonably short PCB trace lines between FPGA and the memory. For eval simulation, read_pulse_tap = 000 is used for all DQS groups.

Table 5-2. Effective Tap Delay for Read Pulse Tap Values

Effect Tap Delay	read_pulse_tap
1 Clock	000
1.5 Clock	001
2 Clock	010
2.5 Clock	011
3 Clock	100
3.5 Clock	101

DQS_PIO_READ Locate Constraints

The DDR/DDR2 IP has a few critical macro-like blocks called as DQS_PIO_READ pgroups that require specific placement locations. This placement is done by adding a “LOCATE” constraint in the .lpf file for each pgroup.

The user must manually update these constraints in the .lpf file by adding location values obtained as follows. In DDR2 mode, all locations of DQS pins are user selectable and the corresponding DQS_PIO_READ macros are automatically implemented. The following steps are for DDR1 mode, or for when the user changes DDR2 DQS pin locations after core generation.

Obtaining Location Values in Diamond Software

***Note**: Refer to Diamond online help for more information about using the Diamond software.*

1. With the Eval project open in Diamond, run the **Place & Route** process without locating the DQS_PIO_READ pgroup in the .lpf file.
2. Enable the **Floorplan View**.
3. In the Floorplan View, find the location of the DQS pin (N9 as shown in the example in Figure 5-3).

a. Find the nearby DQSBUF block.

b. In the .lpf file, locate the DQS0_PIO_READ pgroup in the row and column of the closest SLICE (R30C2D as shown in the example in Figure 5-3) from this DQSBUF block.

4. Repeat Step 3 for each DQS pin of the design.

5. Once the .lpf file is updated for all DQS pins, re-run the **Place & Route** process using the updated .lpf file.

Note: If there is a MAXDELAY violation on any of the constraints listed below, locate the corresponding DQS_PIO_READ pgroup in the adjacent SLICE and re-run PAR.

MAXDELAY TO CELL "/pio_read_neg*"*

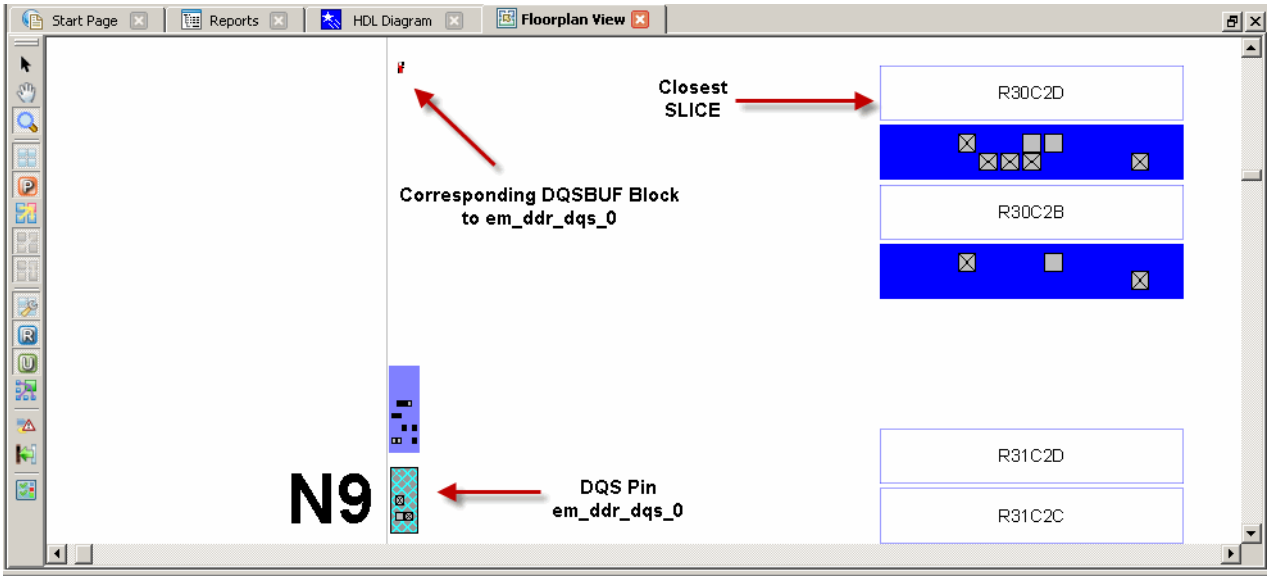
MAXDELAY NET "/pio_read_pos"*

MAXDELAY NET "/pio_read_neg*"*

MAXDELAY NET "dqs_pio_read"*

Figure 5-3 is an example Diamond Floorplan View showing typical locations of the DQS pin (N9), the corresponding DQSBUF block, and the closest SLICE (R30C2D) in a LatticeECP3 device.

Figure 5-3. Diamond Floorplan View



Obtaining Location Values in ispLEVER Software

***Note**: Refer to ispLEVER online help for more information about using the ispLEVER software.*

1. With the Eval project open in the ispLEVER Project Navigator, run the **Place & Route** process without locating the DQS_PIO_READ pgroup in the .lpf file.

2. Open the **Design planner [Pre-Map]** for this design and enable **Floorplan View**.

3. In the Floorplan View, find the location of the dqs0 pin (for example: K4, M23, etc.).

a. Find the nearby DQSBUF block (example: DQS_L48, DQS_R48,.etc.).

b. In the .lpf file, locate the DQS0_PIO_READ pgroup in the row and column of the closest SLICE (for example R48C2D, R48C181D,.etc.) from this DQSBUF.

4. Repeat Step 3 for each DQS pin of the design.

5. Once the .lpf file is updated for all DQS pins, re-run the **Place & Route** process using the updated .lpf file.

Note: If there is a MAXDELAY violation on any of the constraints listed below, locate the corresponding DQS_PIO_READ pgroup in the adjacent SLICE and re-run PAR.

MAXDELAY TO CELL "/pio_read_neg*"*

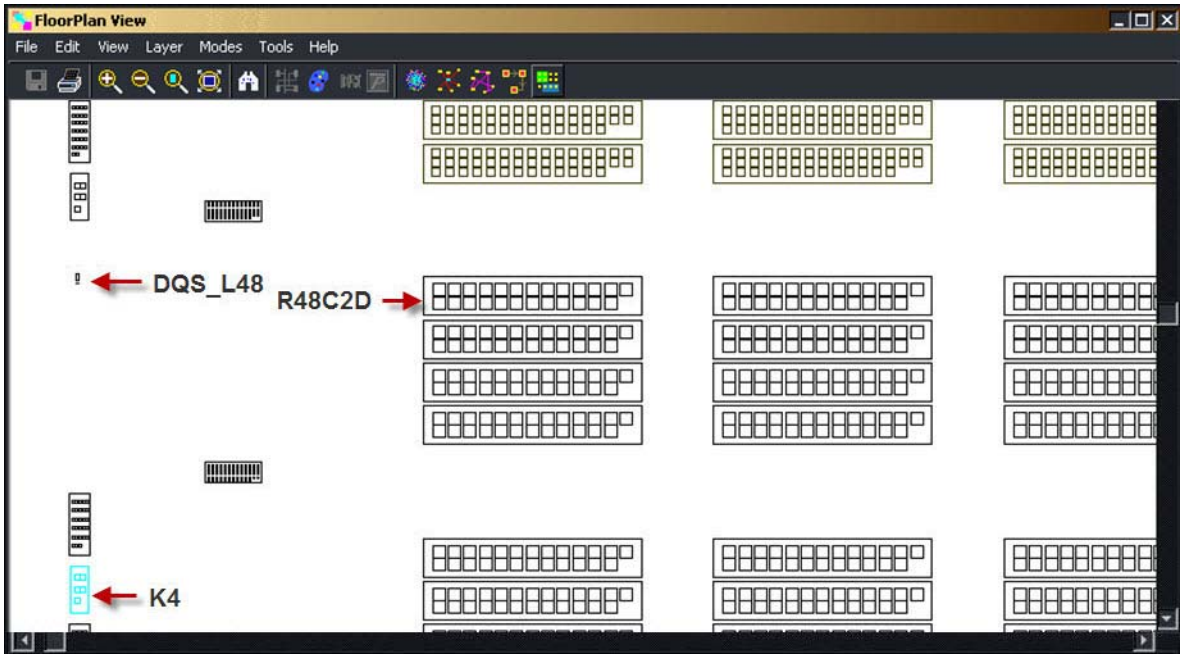
MAXDELAY NET "/pio_read_pos"*

MAXDELAY NET "/pio_read_neg*"*

MAXDELAY NET "dqs_pio_read"*

Figure 5-4 is an example ispLEVER Floorplan View showing typical locations of the DQS pin (K4), DQSBUF block (DQS_L48), and the closest SLICE (R48C2D).

Figure 5-4. ispLEVER Floorplan View



Troubleshooting

When a Lattice DDR memory controller-based system does not work as expected, there could be numerous reasons for the failure. Table 5-3 summarizes some approaches for troubleshooting during DDR system implementation.

Table 5-3. Troubleshooting of DDR Memory Controller Implementation

Symptom	Possible Reason	Troubleshooting
No read data received	Incoming DQS failure	Monitor the DQS signal from the memory. If no DQS is detected during the read operation, it may be a memory failure. Replace the memory.
	VREF connection not established	Monitor the PRMBDET signal. If DQS comes in properly and PRMBDET is not detected or malfunctions, this suggests that VREF is not properly connected to the banks. The VCCIO/2 reference voltage must be connected to the VREF1 pin in all the banks in which DDR inputs are implemented.
Corrupted Read data	Incorrect read data valid timing	Check whether the READ input (to DQSBUF) has been properly constrained and implemented (MAXDELAY NET preference). The READ net delay should be less than half of t_{CLK}.
		Adjust the READ_PULSE TAP value for each DQS group.
Data corruption in a specific frequency range	Data valid timing alignment failure	Although rare, it is possible for this to happen if the memory module is incompatible with the implemented core. Try different memory modules.
		If the symptom persists with the hardware data valid generation, switch to the LUT-based data valid generation logic. It may remove or move the failure range.
Read data is shifted in simulation while the hardware system is working	Clock delta delay	If a design has one or more clocks assigned from the original clock source, the design may have the clock delta delay issue when both the original and assigned clocks are used in the design. Bring the internal clock generator block out to the top-level of the system and distribute it to the rest of the sub-design blocks.
Unexpectedly low performance	Incorrect read data valid timing	See the description for “Corrupt read data”.
	Un-terminated memory control signals	If a system uses only a part of a DDR memory module and the unused memory control signals (such as chip select and clock enable) and the module remains unterminated, they may make the memory module sensitive to noise. Unused control signals must be terminated to their inactive state.
	Excessive skew on memory control signals	Excessive skew between any DDR interface signals can degrade performance. All address and control signals on the DDR interface must be implemented in the PIO registers to minimize the skew.

Core Verification

The functionality of the Lattice DDR and DDR2 IP cores have been verified via simulation and hardware testing in a variety of environments, including:

- Simulation environment verifying proper DDR and DDR2 functionality using Lattice's proprietary verification environment
- Hardware validation of the IP implemented on Lattice FPGA evaluation boards. Specific testing has included:
  - Verifying proper DDR and DDR2 protocol functionality
  - Verifying DDR and DDR2 electrical compliance.
- In-house interoperability testing with multiple DIMM modules

Support Resources

This chapter contains information about Lattice Technical Support, additional references, and document revision history.

Lattice Technical Support

There are a number of ways to receive technical support.

Online Forums

The first place to look is Lattice Forums (http://www.latticesemi.com/support/forums.cfm). Lattice Forums contain a wealth of knowledge and are actively monitored by Lattice Applications Engineers.

Telephone Support Hotline

Receive direct technical support for all Lattice products by calling Lattice Applications from 5:30 a.m. to 6 p.m. Pacific Time.

- For USA & Canada: 1-800-LATTICE (528-8423)
- For other locations: +1 503 268 8001

In Asia, call Lattice Applications from 8:30 a.m. to 5:30 p.m. Beijing Time (CST), +0800 UTC. Chinese and English language only.

- For Asia: +86 21 52989090

E-mail Support

- techsupport@latticesemi.com
- techsupport-asia@latticesemi.com

Local Support

Contact your nearest Lattice Sales Office.

Internet

www.latticesemi.com

References

LatticeECP, LatticeXP

- TN1050, *LatticeECP/EC and LatticeXP DDR Usage Guide*

LatticeECP2/M

- HB1003, *LatticeECP2M Family Handbook*
- TN1105, *LatticeECP2/M High-Speed I/O Interface*
- TN1114, *Electrical Recommendations for Lattice SERDES*

LatticeXP2

- TN1138, LatticeXP2 High-Speed I/O Interface

LatticeSC/M

- DS1004, *LatticeSC/M Family Data Sheet*
- DS1005, *LatticeSC/M Family flexiPCS Data Sheet*
- TN1099, *LatticeSC/M DDR/DDR2 SDRAM Memory Interface User's Guide*

LatticeECP3

- HB1009, *LatticeECP3 Family Handbook*
- TN1180, *LatticeECP3 High-Speed I/O Interface*

Revision History

Date	Document Version	IP Core Version	Change Summary
June 2006	02.4	6.3	Updated to include information for DDR2.
September 2006	02.5	6.3	Updated to included LatticeECP2 DDR2 data, timing diagram for initialization, and graphical representation of read_pulse_tap timing parameters.
October 2006	02.6	6.3	Added paragraph regarding the use of a dummy write immediately before the Command Application Logic section.
November 2006	02.7	6.3	Updated to include IPexpress version of DDR1 for LatticeECP/EC, LatticeSC, LatticeECP2, and LatticeECP2M.
			Updated to include DDR2 for LatticeECP2M.
			Removed non-IPexpress versions of DDR1
March 2007	02.8	6.3	Updated to included recent improvement to 533 Mbps DDR2 data rate.
April 2007	02.9	6.3	Updated Features bullets to include support for LatticeSCM and LatticeECP2M.
May 2007	03.0	6.3	Updated appendices. Added appendix for LatticeXP2 FPGA family.
September 2007	04.0	6.3	New user guide released replacing 03.0 version.
December 2007	04.1	6.4	Updated for the changes made on the DDR1/DDR2 V6.4 cores.
June 2008	04.2	6.5	Updated for the changes made on the DDR1/DDR2 V6.5 cores.
			Updated appendices for ispLEVER 7.1 software release
			Changed document title from "DDR1 & DDR2 SDRAM Controller IP Cores (Pipelined Version)" to "DDR1 & DDR2 SDRAM Controllers IP Cores (Pipelined Versions)".
July 2008	04.3	6.5	Added Read/Write with Auto Precharge text section.
June 2009	04.4	6.6	Updated appendices and added support for the LatticeECP3 device family.
November 2009	04.5	6.7	Updated Table "Initialization Default Values for DDR1/DDR2 Mode Registers."
			Updated appendices for ispLEVER 8.0.
May 2010	04.6	6.7	Divided document into chapters.
			Updated Figure 2-1.
			Added Chapter 6, Core Verification.
			Miscellaneous text changes throughout.

Date	Document Version	IP Core Version	Change Summary
August 2010	04.7	7.0	Added Diamond software support throughout.
			Updated Table 1-2 on page 6.
			Updated Figure 2-1 on page 7.
			Added "DQS Pin Selection Tab (DDR2 Only)" on page 27.
			Added "Obtaining Location Values in Diamond Software" on page 44, Figure 2-1 on page 7, and Figure 5-3 on page 45.
			Updated "Resource Utilization" on page 52.
January 2011	04.8	6.8 (DDR) 7.0 (DDR2)	Added Appendix B.
			Updated for Lattice Diamond 1.1 design software.
June 2011	04.9	6.9 (DDR) 7.1 (DDR2)	Updated "Command and Address" on page 11 with note about initialization read of the physical layer.
			Added information about t_{WPST} length for "LatticeECP3 FPGAs" on page 54.
			Updated for Lattice Diamond 1.2 design software.
February 2012	05.0	6.9 (DDR) 7.1 (DDR2)	Updated document with new corporate logo.

Resource Utilization

This appendix gives resource utilization information for Lattice FPGAs using the DDR/DDR2 IP core. The IP configurations shown in this chapter were generated using the IPexpress software tool. IPexpress is the Lattice IP configuration utility, and is included as a standard feature of the Diamond and ispLEVER design tools. Details regarding the usage of IPexpress can be found in the IPexpress and Diamond and ispLEVER help systems. For more information on the Diamond or ispLEVER design tools, visit the Lattice web site at: www.latticesemi.com/software.

LatticeECP/EC Devices

Table A-1. Performance and Resource Utilization[1]

IP Core	Parameter Settings[2]	SLICEs	LUTs	Registers	I/O	f_{MAX} (MHz)
DDR	Table 3-1 on page 19 parameter defaults	1295	1367	1761	249	166 MHz (333 DDR)

1. Performance and utilization characteristics are generated using LFECP33-5F672C with Lattice Diamond 1.2 software. Performance may vary when using this IP core in a different density, speed or grade within the LatticeECP/EC family.
2. SDRAM data path width of 32 bits

Ordering Part Number

The Ordering Part Number (OPN) for the Pipelined DDR SDRAM Controller IP on LatticeECP/EC devices is DDRCT-GEN-E2-U6.

LatticeECP20 and LatticeEC20 Devices

This section only applies when using the 20K LUT density of the LatticeECP/EC device families.

DQS Postamble Handling

The em_ddr_dqs signal is tri-stated 1/2 clock cycle (postamble) after the last transition at the end of a read cycle. Since the net is pulled to $V_{TT/2}$ by the termination resistor, the internal DQS signal (not shown in diagrams) goes into oscillation since the termination voltage is the same as V_{REF}. This can cause the last data that is read from the memory to be overwritten before it is transferred to a free running clock.

Preventing the em_ddr_dqs signal from going to a tri-state before the last data is read solves this problem. This is achieved by issuing an extra read to the memory chip. The data from this unwanted read is not propagated to the user. The circumstances under which this read is issued are given in Table on page 13.

Table A-2. DQS Postamble Solutions

Current Command	Next Command	Action
Read (Row x, Bank y)	Read (Row x, Bank y)	None.
Read (any address)	NOP (no operation)	Issue another Read consecutive to the current read command.
Read (Row x, Bank y)	Read (Row n, Bank y)	Issue another Read to (Row x, Bank y) consecutive to current command
Read (Row x, Bank y)	Read (Row x, Bank n)	If the Row x, Bank n was open, do nothing. Else, issue a read command to Row x, Bank y
Read (any address)	Write/LOAD_MR	Issue another Read consecutive to the current read command.

This solution comes with a limitation where the IP will not be supporting the Read with Auto Precharge command.

LatticeECP2/S FPGAs

Table A-3. Performance and Resource Utilization[1]

IP Core	Parameter Settings[2]	SLICEs	LUTs	Registers	I/O	f_{MAX} (MHz)$_3$
DDR	Table 3-1 on page 19 parameter defaults	1195	1386	1558	249	200 MHz (400 DDR)
DDR2	Table 3-1 on page 19 parameter defaults	1241	1435	1538	258	266 MHz (533 DDR)

1. Performance and utilization characteristics are generated using LFECP2-50E-7F672C with Lattice Diamond 1.2 software. Performance may vary when using this IP core in a different density, speed or grade within the LatticeECP2/S family.
2. SDRAM data path width of 32 bits.
3. The DDR2 IP core can operate at 266 MHz (533 DDR2) in the fastest speed-grade (-7) when the data width is 64 bits or less and 2 or fewer chip selects are used. For help with designs running at 266 MHz, contact your local sales office.

Ordering Part Number

The Ordering Part Number (OPN) for the Pipelined DDR SDRAM Controller IP on LatticeECP2/S devices is DDRCT-GEN-P2-U6 and for the Pipelined DDR2 SDRAM Controller IP it is DDR2-P-P2-U6.

LatticeECP2M/S FPGAs

Table A-4. Performance and Resource Utilization[1]

IP Core	Parameter Settings[2]	SLICEs	LUTs	Registers	I/O	f_{MAX} (MHz)[3]
DDR	Table 3-1 on page 19 parameter defaults	1195	1386	1558	249	200 MHz (400 DDR)
DDR2	Table 3-1 on page 19 parameter defaults	1241	1435	1538	258	266 MHz (533 DDR)

1. Performance and utilization characteristics are generated usingLFECP2M-35E-7F672C with LatticeDiamond 1.2 software. Performance may vary when using this IP core in a different density, speed or grade within the LatticeECP2M/S family.
2. SDRAM data path width of 32 bits.
3. The DDR2 IP core can operate at 266 MHz (533 DDR2) in the fastest speed-grade (-7) when the data width is 64 bits or less and 2 or fewer chip selects are used. For help with designs running at 266 MHz, contact your local sales office.

Ordering Part Number

The Ordering Part Number (OPN) for the Pipelined DDR SDRAM Controller IP on LatticeECP2M/S devices is DDRCT-GEN-PM-U6 and for the Pipelined DDR2 SDRAM Controller IP it is DDR2-P-PM-U6.

LatticeECP3 FPGAs

Table A-5. Performance and Resource Utilization[1]

IP Core	Parameter Settings[2]	SLICEs	LUTs	Registers	I/O	f_{MAX} (MHz)[3]
DDR	Table 3-1 on page 19 parameter defaults	1175	1403	1594	249	200 MHz (400 DDR)
DDR2	Table 3-1 on page 19 parameter defaults	1192	1391	1568	258	266 MHz (533 DDR)

1. Performance and utilization characteristics are generated using LFE3-95E-8FN1156C with Lattice Diamond 1.2 software. Performance may vary when using this IP core in a different density, speed or grade within the LatticeECP3 family.
2. SDRAM data path width of 32 bits.
3. The DDR2 IP core can operate at 266 MHz (533 DDR2) in the fastest speed-grade (-8) when the data width is 64 bits or less and 2 or fewer chip selects are used. For help with designs running at 266 MHz, contact your local sales office.

Ordering Part Number

The Ordering Part Number (OPN) for the Pipelined DDR SDRAM Controller IP on LatticeECP3 devices is DDRCT-GEN-E3-U6 and for the Pipelined DDR2 SDRAM Controller IP it is DDR2-P-E3-U6.

DQS Postamble Handling

The em_ddr_dqs signal is tri-stated 1-1/2 clock cycles (postamble) after the last transition at the end of a write cycle. The JEDEC DDR SDRAM specification for this time period is labeled as t_{WPST}. The LatticeECP3 t_{WPST} is longer compared to the standard specification, however the value of t_{WPST} can be device specific.

LatticeXP Devices

Table A-6. Performance and Resource Utilization[1]

IP Core	Parameter Settings[2]	SLICEs	LUTs	Registers	I/O	f_{MAX} (MHz)
DDR	Table 3-1 on page 19 parameter defaults	1295	1367	1761	249	133 MHz (266 DDR)

1. Performance and utilization characteristics are generated using LFXP20E-5F484C with Lattice Diamond 1.2 software. Performance may vary when using this IP core in a different density, speed or grade within the LatticeXP family.
2. SDRAM data path width of 32 bits.

Ordering Part Number

The Ordering Part Number (OPN) for the Pipelined DDR SDRAM Controller IP on LatticeXP devices is DDRCT-GEN-XM-U6.

LatticeXP2 Devices

Table A-7. Performance and Resource Utilization[1]

IP Core	Parameter Settings[2]	SLICEs	LUTs	Registers	I/Os	f_{MAX} (MHz)
DDR	Table 3-1 on page 19 parameter defaults	1193	1384	1558	249	200 MHz (400 DDR)
DDR2	Table 3-1 on page 19 parameter defaults	1239	1433	1538	258	200 MHz (400 DDR)

1. Performance and utilization characteristics are generated using LFXP2-17E-6F484C with LatticeDiamond 1.2 software. Performance may vary when using this IP core in a different density, speed or grade within the LatticeXP2 family.
2. SDRAM data path width of 32 bits.

Ordering Part Number

The Operating Part Number (OPN) for the Pipelined DDR SDRAM Controller IP on LatticeXP2 devices is DDRCT-GEN-X2-U6 and for the Pipelined DDR2 SDRAM Controller IP is DDR2-P-X2-U6.

LatticeSC/M FPGAs

Table A-8. Performance and Resource Utilization[1]

IP Core	Parameter Settings[2]	SLICEs	LUTs	Registers	I/O	f_{MAX} (MHz)
DDR	Table 3-1 on page 19 parameter defaults	1111	1277	1517	237	200 MHz (400 DDR)
DDR2	Table 3-1 on page 19 parameter defaults	1252	1480	1532	246	266 MHz (533 DDR2)

1. Performance and utilization characteristics are generated using LFSC3GA25E-6F900C with Lattice Diamond 1.2 software. Performance may vary when using this IP core in a different density, speed or grade within the LatticeSC/M family.
2. SDRAM data path width of 32 bits.

Ordering Part Number

The Ordering Part Number (OPN) for the Pipelined DDR SDRAM Controller IP on LatticeSC/M devices is DDRCT-GEN-SC-U6 and the OPN for the DDR2 SDRAM Controller IP on LatticeSC/M devices is DDR2-P-SC-U6.

1. In DDR1 mode, simulation fails when the chip select is 4 and the data width is larger than 32 bits.

Общество с ограниченной ответственностью «МосЧип» ИНН 7719860671 / КПП 771901001
Адрес: 105318, г.Москва, ул.Щербаковская д.3, офис 1107

Данный компонент на территории Российской Федерации

Вы можете приобрести в компании MosChip.

Для оперативного оформления запроса Вам необходимо перейти по данной ссылке:

http://moschip.ru/get-element

Вы можете разместить у нас заказ для любого Вашего проекта, будь то серийное производство или разработка единичного прибора.

В нашем ассортименте представлены ведущие мировые производители активных и пассивных электронных компонентов.

Нашей специализацией является поставка электронной компонентной базы двойного назначения, продукции таких производителей как XILINX, Intel (ex.ALTERA), Vicor, Microchip, Texas Instruments, Analog Devices, Mini-Circuits, Amphenol, Glenair.

Сотрудничество с глобальными дистрибьюторами электронных компонентов, предоставляет возможность заказывать и получать с международных складов практически любой перечень компонентов в оптимальные для Вас сроки.

На всех этапах разработки и производства наши партнеры могут получить квалифицированную поддержку опытных инженеров.

Система менеджмента качества компании отвечает требованиям в соответствии с ГОСТ Р ИСО 9001, ГОСТ РВ 0015-002 и ЭС РД 009

Офис по работе с юридическими лицами:

105318, г.Москва, ул.Щербаковская д.3, офис 1107, 1118, ДЦ «Щербаковский»

Телефон: +7 495 668-12-70 (многоканальный)

Факс: +7 495 668-12-70 (доб.304)

E-mail: info@moschip.ru

Skype отдела продаж:
moschip.ru
moschip.ru_4
moschip.ru_6
moschip.ru_9