LatticeCORE

MachXO2™ LPDDR SDRAM Controller IP Core User's Guide

February 2014
IPUG92_01.3

Table of Contents

Introduction

Introduction

The LPDDR Synchronous Dynamic Random Access Memory (SDRAM) Controller is a general-purpose memory controller that interfaces with industry standard LPDDR memory devices/modules compliant with JESD209B, LPDDR SDRAM Standard, and provides a generic command interface to user applications. This IP core reduces the effort required to integrate the LPDDR memory controller with the remainder of the application and minimizes the need to directly deal with the LPDDR memory interface.

Quick Facts

Table 1-1 gives quick facts about the LPDDR SDRAM Controller IP core for MachXO2™ devices.

Table 1-1. LPDDR SDRAM Controller IP Core Quick Facts

		LPDDR IP Configuration			
		16-bit, Generic	8-bit, Generic	16-bit, Wishbone	8-bit, Wishbone
Core Requirements	FPGA Families Supported	MachXO2			
	Minimum Device Needed	LCMXO2-2000 CABGA256	LCMXO2-2000 TQFP144	LCMXO2-2000 CABGA256	LCMXO2-2000 CSBGA132
Resource Utilization	Target Device	LCMXO2-7000HE-6FG484C			
	LUTs	1684	1620	1523	1389
	sysMEM EBRs	0	0	0	
	Registers	880	849	933	834
Design Tool Support	Lattice Implementation	Lattice Diamond® 3.1			
	Synthesis	Lattice Synthesis Engine (LSE)			
		Synopsys Synplify Pro® for Lattice I-2013.09L			
	Simulation	Aldec Active-HDL™ 9.3 Lattice Edition			
		Mentor Graphics ModelSim® SE 6.6 or later			

Features

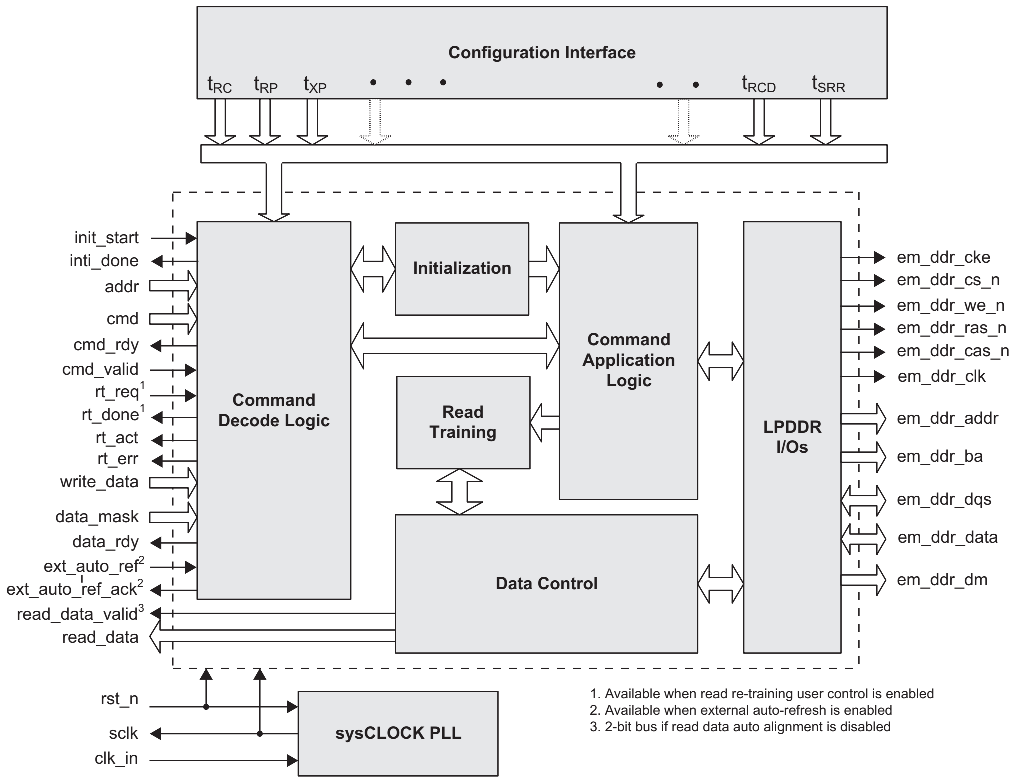
- Interfaces to industry standard LPDDR SDRAM devices compliant to JESD209B
- Double-data rate architecture: two data transfers per clock cycle
- Bi-directional data strobe per byte of data (DQS)
- Programmable auto refresh support
- Data mask support – one mask per byte
- Power-down and deep-power-down support
- Power-on initialization support
- Re-initialization after a deep-power-down support
- Dynamic I/O training after initialization
- Periodic I/O retraining after an auto-refresh burst
- Dynamic memory clock power off during self refresh, power-down and deep-power-down operations
- Single-port operation support
- Full, half and quarter array self refresh support
- Status register read support
- TCSR programmability through MRS

Functional Description

Overview

The LPDDR memory controller consists of two major parts: the controller core logic module and the I/O logic module. This section briefly describes the operation of each of these modules. Figure 2-1 provides a high-level block diagram illustrating the main functional blocks and the technology used to implement the LPDDR SDRAM Controller IP core functions.

Figure 2-1. LPDDR SDRAM Controller Block Diagram



The core module has several functional sub-modules: Initialization Block, Command Decode Logic Block, Command Application Logic Block, Data Control Block and Read Training Block. LPDDR I/O module implements the physical interface (PHY) to the memory device. This block mostly consists of MachXO2 device I/O primitives which are compliant to LPDDR electrical and timing requirements.

Initialization Block

The Initialization Block performs the LPDDR memory initialization sequence as defined by the JEDEC protocol. After power-on or a normal reset of the LPDDR controller, memory device must be initialized before any command is sent to the controller. It is user's responsibility to assert the init_start input to the LPDDR controller to start the memory initialization sequence. The completion of initialization is indicated by the init_done output signal.

This module automatically re-initializes the memory device upon exiting a deep-power-down operation.

Read Training Block

The Read Training Block adjusts the MachXO2 I/Os for write/read operations. It is automatically activated at the end of an initialization sequence and at the end of every Nth auto refresh burst cycle where N is a user configurable value to set a read re-training interval provided in IPexpress GUI. Alternatively, a user may choose to fully control the timing of read re-training by selecting the Read Re-training User Control option from the GUI.

Data Control Block

The Data Control Block generates the control signals for the I/Os for write/read operations. This block implements all the logic needed to ensure that the data write/read to and from the memory is transferred to the local user interface in a deterministic and coherent manner.

LPDDR I/Os

The LPDDR I/O modules are MachXO2 device primitives that directly connect to the LPDDR memory. These primitives implement all the interface signals required for memory access. They convert the single data rate (SDR) data to double rate LPDDR data for write operations and perform the LPDDR to SDR conversion in read mode.

Command Decode Logic Block

The Command Decode Logic (CDL) Block accepts user commands from the local interface and decodes them to generate a sequence of internal memory commands depending on the current command and the status of current bank and row. It tracks the open/close status of every bank and stores the row address of every open bank. The controller implements a command pipeline enabling the next command in the queue to be decoded while the current command is presented at the memory interface.

Command Application Logic Block

The Command Application Logic (CAL) Block accepts and processes the internal command sequences from the Command Decode Logic. It translates each sequence into memory commands that comply with the JEDEC protocol sequences and timing requirements of the LPDDR memory device. It is the module responsible for protocol compliance.

Signal Descriptions

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

Table 2-1. LPDDR SDRAM Memory Controller Top-Level I/O List for Generic Interface

Port Name	Active State	I/O	Description
Local User Interface			
clk_in	N/A	Input	Reference clock. It is connected to the PLL input.
rst_n	Low	Input	Asynchronous reset. It resets the entire IP core when asserted.
init_start	High	Input	Initialization start request. Should be asserted to initiate memory initialization either right after the power-on reset or before sending the first user command to the memory controller
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 and cmd inputs are valid.
addr[ADDR_WIDTH-1:0]	N/A	Input	User read or write address input to the memory controller.
write_data[DSIZE-1:0]	N/A	Input	Write data input from user logic to the memory controller. The user side write data width is two times the memory data bus
data_mask[(DSIZE/8)-1:0]	N/A	Input	Data mask input for write_data.
rt_req	High	Input	Read training request. Available when the read re-training user control option is enabled. When asserted, the core is requested to perform read training.

Table 2-1. LPDDR SDRAM Memory Controller Top-Level I/O List for Generic Interface (Continued)

Port Name	Active State	I/O	Description
rt_err	High	Output	Read training error output. When asserted, it indicates that the read training process has been failed.
sclk	N/A	Output	System clock output. The user logic uses this as a system clock unless an external clock generator is used.
init_done	High	Output	Initialization done output. It is asserted for one clock cycle when the core completes the memory initialization routine.
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 [DQS_WIDTH-1:0] /read_data_valid	High	Output	Read data valid output. When asserted, it indicates the data on the read_data bus is valid. Each bit is an independent output of the corresponding DQS group when the read data auto-alignment option is disabled. When enabled, the single-bit read_data_valid validates the auto-aligned read data.
rt_done	High	Output	Read training done output. Available when the read re-training user control option is enabled. When asserted, the core indicates that the requested read training has been completed.
rt_act	High	Output	Read training active output. When asserted, it indicates that the core is in the read training process.
ext_auto_ref	High	Input	User external auto refresh request. Available when the external auto refresh option is enabled. When asserted, the core is requested to perform an auto refresh burst.
ext_auto_ref_ack	High	Output	User external auto refresh acknowledge output. Available when the external auto refresh option is enabled. When asserted, the core indicates that the requested user auto-refresh burst has been completed.

Table 2-2 describes the user interface signals at the top level I/O for Wishbone Interface.

Table 2-2. LPDDR SDRAM Memory Controller Top-Level I/O List for Wishbone Interface

Port Name	I/O	Description
S0_ADR_I[31:0], S1_ADR_I[31:0]	Input	The address input array used to pass a binary address. For LPDDR memory of: 2 G: S0_ADR_I[26:0] = valid address S0_ADR_I[31:27] = unused 1 G: S0_ADR_I[25:0] = valid address S0_ADR_I[31:26] = unused 512 M: S0_ADR_I[24:0] = valid address S0_ADR_I[31:25] = unused 256 M: S0_ADR_I[23:0] = valid address S0_ADR_I[31:24] = unused 128 M: S0_ADR_I[22:0] = valid address S0_ADR_I[31:23] = unused
S0_DAT_I[31:0], S1_DAT_I[31:0]	Input	The data input array used for write data
S0_SEL_I[4:0], S1_SEL_I[4:0]	Input	The select input array indicates where valid data is placed on the DAT_I signal array during WRITE cycles and where it should be present on the DAT_O signal array during READ cycles.
S0_WE_I, S1_WE_I	Input	The write enable Input WE_I indicates whether the current local bus cycle is a READ or WRITE cycle. The signal is negated during READ cycles and is asserted during WRITE cycles
S0_CYC_I[2:0], S1_CYC_I[2:0]	Input	The Cycle Input CYC_I, when asserted, indicates that a valid bus cycle is in progress. The signal is asserted for the duration of all bus cycles. The CYC_I signal is asserted during the first data transfer and remains asserted until the last data transfer.
CLK0_I, CLK1_I	Input	Wishbone input clocks
RST0_I, RST1_I	Input	Wishbone reset
S0_CTI_I, S1_CTI_I	Input	Cycle type identifier input. It provides additional information about the current cycle.
S0_BTE_I, S1_BTE_I	Input	Burst type extension input. It provides additional information about the current burst.
S0_LOCK_I, S1_LOCK_I	Input	Lock input signal. When asserted, it indicates that the current cycle is uninterruptible.
S0_STB_I, S1_STB_I	Input	Strobe input. When asserted, it indicates that the core can respond to other Wishbone signals.
S0_DAT_O[31:0], S1_DAT_O[31:0]	Output	The data Output array used for read data
S0_ACK_O, S1_ACK_O	Output	The acknowledge output ACK_O, when asserted, indicates the termination of a normal bus cycle by the slave
S0_ERR_O, S1_ERR_O	Output	The error output ERR_O indicates an abnormal cycle termination by the slave.

Table 2-2. LPDDR SDRAM Memory Controller Top-Level I/O List for Wishbone Interface (Continued)

Port Name	I/O	Description
S0_RTY_O, S1_RTY_O	Output	The retry output RTY_O indicates that the slave interface is not ready to accept or send data
S0_INT_O, S1_INT_O	Output	Interrupt output. It is for a future use and can be disregarded.

Using the Local User Interface

The local user interface of the LPDDR memory controller IP core consists of five independent functional groups:

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

Each functional group and its associated local interface signals as listed in Table 2-3.

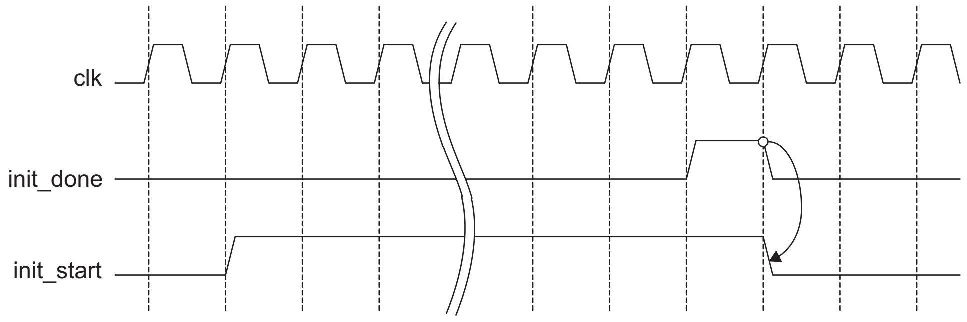
Table 2-3. Local User Interface Functional Groups

Functional Group	Signals
Initialization and Training	init_start, init_done, rt_req, rt_done, rt_act, rt_err
Command and Address	addr, cmd, cmd_rdy, cmd_valid
Data Write	data_rdy, write_data, data_mask
Data Read	read_data, read_data_valid
External Auto Refresh	ext_auto_ref, ext_auto_ref_ack

Initialization and Training

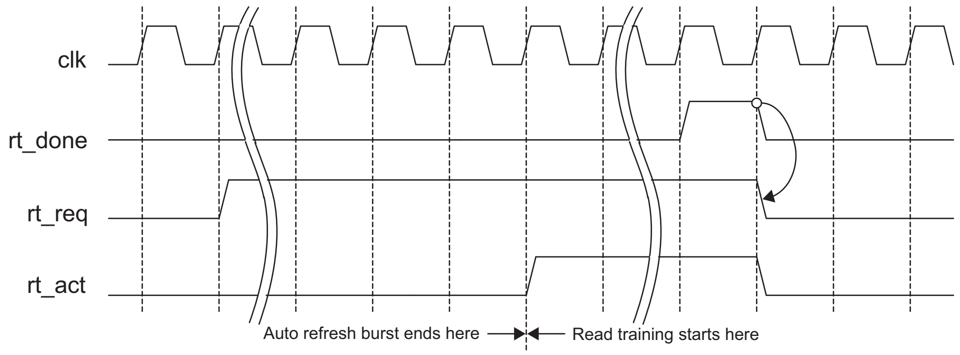
LPDDR memory devices must be initialized before the memory controller can access them. The memory controller starts the memory initialization sequence when the init_start signal is asserted by the user interface. Once asserted, the init_start signal needs to be held high until the initialization process is completed. The output signal init_done is asserted high for one clock cycle indicating that the core has completed the initialization sequence and is now ready to access the memory. The init_start signal must be de-asserted as soon as init_done is sampled high at the rising edge of sclk. If the init_start is left high at the next rising edge of sclk, the memory controller takes it as another request for initialization and starts the initialization process again. Memory initialization is required immediately after the system reset or after a deep-power-down exit. Re-initialization after a deep-power-down exit is automatically handled by the core.

Figure 2-2. Timing of Memory Initialization Control



LPDDR memory devices do not use DLL, and therefore the controller needs to periodically compensate for process, voltage and temperature (PVT) variations. Read training is initially performed immediately following the memory initialization process. Once the initialization is done, the core automatically performs read re-training during the normal operations with a user selected training interval. Since a re-training process takes over the LPDDR bus until it is finished, the core allows re-training only after an auto-refresh burst cycle is finished in order to minimize interruption of user transactions. The training interval is defined in terms of the number of auto refresh burst cycles. The user can select every 8th, 64th, 256th, 512th or 1024th auto refresh burst cycle to initiate a read re-training process. If a longer period is needed or the training is to be initiated by the user, the Read Re-training User Control option can be enabled. Once enabled, the rt_req and rt_done signals are added to the user interface. The rt_req signal can be asserted at any time during a normal operation, but the actual training will start at the end of next auto refresh burst cycle. The rt_req signal must be de-asserted as soon as rt_done is sampled high at the rising edge of sclk. Figure 2-3 shows the timing of user read re-training.

Figure 2-3. Timing of User Read Training



Since LPDDR memory devices do not provide a training pattern, the controller must reserve a user address space to train the bus. The location of the training address space is user configurable through the Address Space for Training option. The address range of training space is 16 and reserved for all core configurations. If the address space is set to 23'h003000, for example, the range of the space becomes 23'h003000 to 23'h00300F. The starting address must be modulo of 16. For example, address 23'h003000 or 23'h003010 is a valid starting address for the training space. This address space should be avoided in user transactions. If the entire range of addressable memory space is to be accessed by user application, the user can select the Read Re-training User Control option to fully take over the control of read training.

Command and Address

Once the memory initialization is done, the core waits for user commands in order to set up and/or access the memory. User logic needs to provide command and address to the core and drive the control signals appropriately. The commands and addresses are delivered to the core using the following procedure:

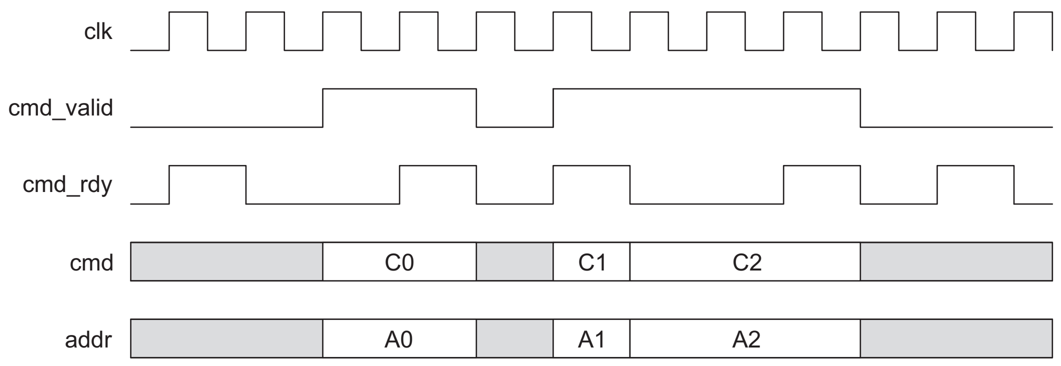
1. The memory controller asserts the signal cmd_rdy for one clock cycle indicating its readiness to receive a command and address.

2. User logic asserts the signal cmd_valid indicating that the command and address on the bus are valid data. When the cmd_rdy is high, user logic must de-assert cmd_valid in the next clock cycle if there is no new data, or keep it high if the bus contains new, valid command and address.

3. When both the cmd_rdy and cmd_valid are high, the core accepts the command and address data on the cmd and addr buses. User logic must de-assert cmd_valid in the next clock cycle if there is no new valid data on the cmd and addr buses, or keep it high if the bus contains new, valid command and address as shown in Figure 2-4.

The assertions of cmd_rdy by the controller and cmd_valid by user logic are independent of each other.

Figure 2-4. Timing of Command and Address



User Commands

The user initiates a request to the memory controller by loading a specific command code in cmd input along with other information such as the memory address. The command on the cmd bus must be a valid command. Lattice LPDDR IP defines a set of valid memory commands as shown in Table 2-4. All other values should not be used.

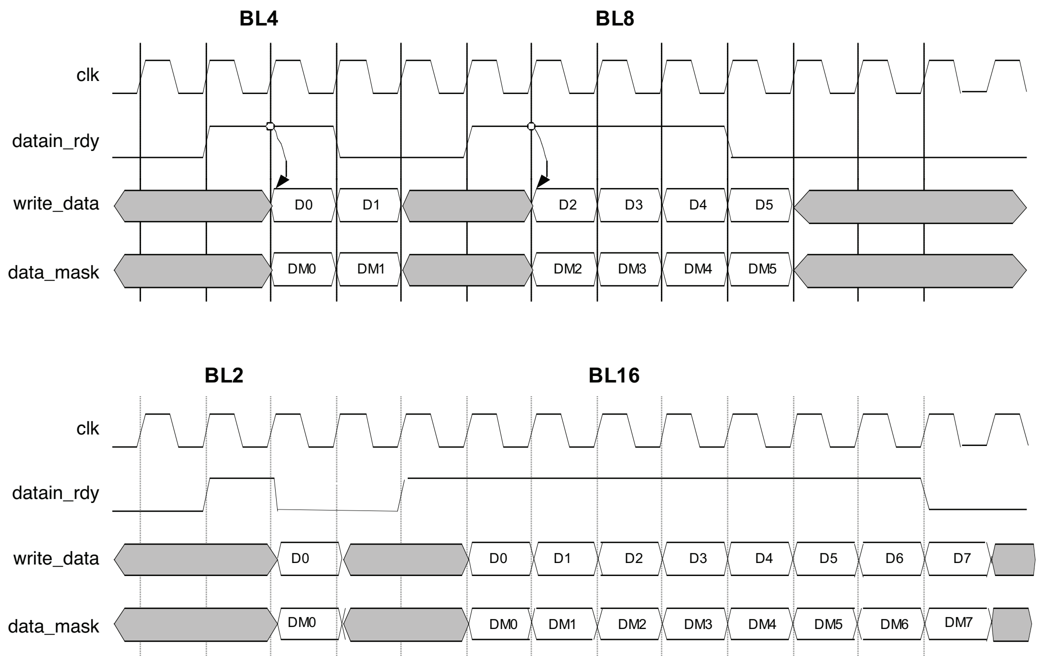
Table 2-4. Defined User Commands for Generic Interface

Command	Mnemonic	cmd[3:0]
Read	RD	0001
Write	WR	0010
Read with Auto Precharge	RDA	0011
Write with Auto Precharge	WRA	0100
Powerdown Entry	PDE	0101
Load Mode Register	LMR	0110
Status Register Read	SRR	0111
Self Refresh Entry	SRE	1000
Self Refresh Exit	SRX	1001
Powerdown Exit	PDE	1010
Deep Powerdown Entry	DPDE	1011
Deep Powerdown Exit	DPDX	1100

WRITE

The user initiates a memory write operation by asserting cmd_valid along with the WRITE or WRITEA command and the address. After the WRITE command is accepted, the memory controller core asserts the datain_rdy signal when it is ready to receive the write data from the user logic to write into the memory. Since the duration from the time a write command is accepted to the time the datain_rdy signal is asserted is not fixed, the user logic needs to monitor the datain_rdy signal. Once datain_rdy is asserted, the core expects valid data on the write_data bus one clock cycle after the datain_rdy signal is asserted. Figure 2-5 shows an example of the local user interface data write timing. The controller decodes the addr input to extract the current row and current bank addresses and checks if the current row in the memory device is already opened. If there is no opened row in the current bank an ACTIVE command is generated by the controller to the memory to open the current row first. Then the memory controller issues a WRITE command to the memory. If there is already an opened row in the current bank and the current row address is different from the opened row, a PRECHARGE command is generated by the controller to close opened row in the bank. This is followed with an ACTIVE command to open the current row. Then the memory controller issues a WRITE command to the memory. If the current row is already open, only a WRITE command (without any ACTIVE or PRECHARGE commands) is sent to the memory.

Figure 2-5. One-Clock Write Data Delay



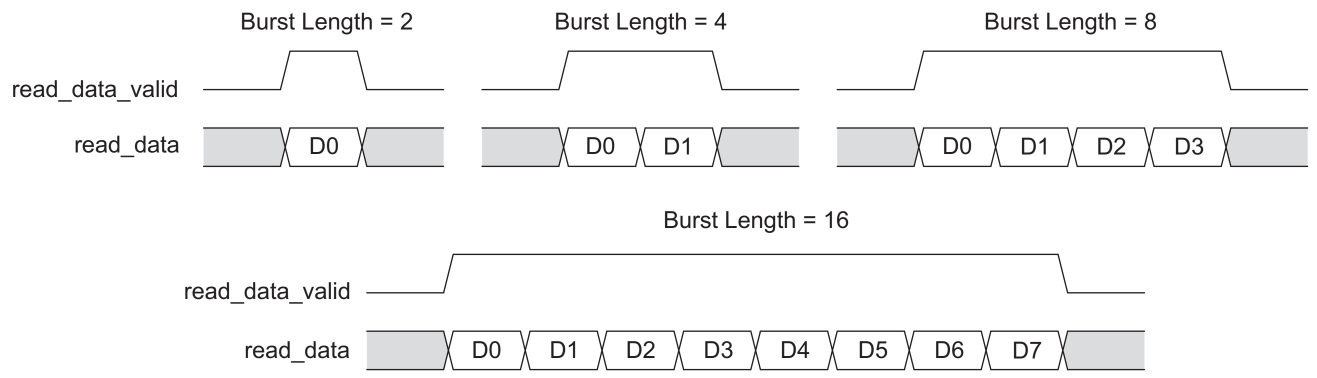
WRITEA

WRITEA is treated in the same way as a WRITE command except that the core issues a Write with Auto Precharge command to the memory instead of just a WRITE command. This causes the memory to automatically close the current row after completing the WRITE operation.

READ

When the READ command is accepted, the memory controller core accesses the memory to read the addressed data and brings the data 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 indicate 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. The READ operation follows the same row status checking scheme as mentioned in the WRITE operation. Depending on the current row status the memory controller generates ACTIVE and PRECHARGE commands as required. Refer to the WRITE operation for more detail.

Figure 2-6. User-Side Read Operation



READA

READA is treated in the same way as a READ command except that the core issues a Read with Auto Precharge command to the memory instead of a READ command. This makes the memory automatically close the current row after completing the read operation.

AUTO REFRESH

Since LPDDR memories have at least an 8-deep Auto Refresh command queue as per the JEDEC specification, the Lattice LPDDR memory controller core supports up to eight Auto Refresh commands in one burst. The core has an internal Auto Refresh Generator that sends out a set of consecutive Auto Refresh commands to the memory at once when it reaches the time period of the refresh intervals (t_{REFI}) times the Auto Refresh burst count selected in the IPexpress GUI. It is recommended that the maximum number be used if the LPDDR interface throughput is a major concern of the system. If it is set to eight, for example, the core will send a set of eight consecutive Auto Refresh commands to the memory once it reaches the time period of the eight refresh intervals (t_{REFI} x 8). Bursting refresh cycles increases the LPDDR bus throughput because it helps keep core intervention to a minimum.

SELF REFRESH

The self refresh command comes as a set of two in compliance to JEDEC protocol: self refresh entry and self refresh exit. The user should always use them as a set, a self refresh exit should always follow a self refresh entry. To minimize power, the user has the option to turn the memory clock off during self refresh operations. This is a user-programmable parameter, and when set, the controller will automatically turn off the memory clock. To minimize the power even further, the controller will refresh half or a quarter of the memory if it is set through an MRS command.

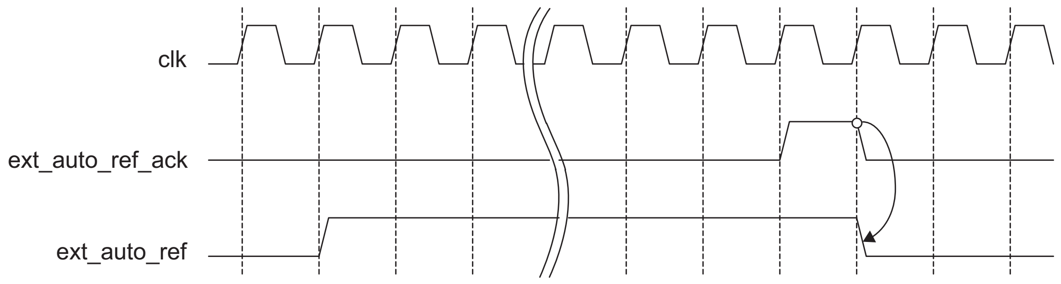
Power Down and Deep Power Down

The power-down commands come as a set of two in compliance to JEDEC protocol: entry and exit. The user should always use them as a set an exit should always follow an entry. To minimize power, the user has the option to turn the memory clock off during power-down operations. For deep power entry, the controller will re-initialize the memory and performs read re-training upon exiting the deep-power-down mode.

EXTERNAL AUTO REFRESH

Alternatively, the user can enable the External Auto Refresh function which will add an input signal ext_auto_ref and an output signal ext_auto_ref_ack to the core. In this case the internal auto refresh generator is disabled and the core sends out a burst of refresh commands, as directed by Auto refresh burst count, every time the ext_auto_ref is asserted. Completion of refresh burst is indicated by the output signal ext_auto_ref_ack. The ext_auto_ref signal must be de-asserted as soon as ext_auto_ref_ack is sampled high at the rising edge of sclk. Figure 2-7 shows the timing of external user auto refresh. If there is an application where explicit memory refresh is not necessary, the user still needs to enable External Auto Refresh and keep the minimum refresh effort so that the read re-training can be initiated from the auto refresh bursts.

Figure 2-7. Timing of External User Auto Refresh



User Commands for Wishbone Interface

The LPDDR controller has a GUI selectable interface compliant to two port WISHBONE bus. Port_0 is used for read/writes and port_1 for programming the mode register and executing special commands like power down, deep power down or self refresh. Once the WISHBONE option is enabled, only the BL2 (Burst Length=2) mode is supported. When the port_1 WISHBONE bus is used to program the mode register, BL2 should be selected for Burst Length.

The memory command mapping of the port_1 wishbone bus are described in Table 2-5.

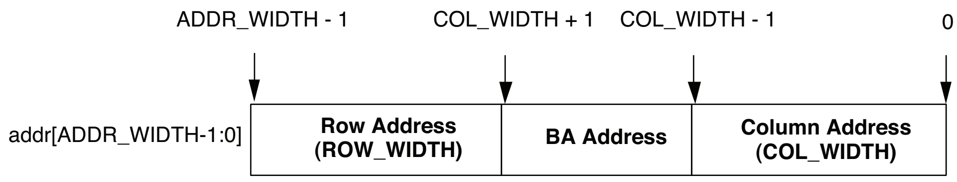
Table 2-5. Memory Command Mapping of the port_1 Wishbone Bus

S1_ADR_I	S1_DAT_I	Command	Description
32‘h1	Valid data =[B6, … B0]	MRS	Bits [B2,B1,B0] = Burst length Bits [B3] = Burst type Bits [B6,B5,B4] = CAS latency -------------------------------- Bits [31, … B7] = Don’t Care
32‘h2	Valid data =[B7, … B0]	EMRS	Bits [B2,B1,B0] = PASR Bits [B4,B3] = TCSR Bits [B7,B6,B5] = Drive Strength -------------------------------- Bits [31, … B8] = Don’t Care
32’h4	Don’t Care	PDE	Power Down Entry
32’h8	Don’t Care	PDX	Power Down Exit
32’h10	Don’t Care	DPDE	Deep Power Down Entry
32’h20	Don’t Care	DPDX	Deep Power Down Exit
32’h40	Don’t Care	SRE	Self Refresh Entry
32’h80	Don’t Care	SRX	Self Refresh Exit
32’h100	Don’t Care	SRR	Status Register Read

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 a memory device's address lines because proper local-to-memory address mapping is crucial to meet the system requirements in applications such as video frame buffering. Even for other applications, careful address mapping is generally necessary to optimize system performance. On the memory side, the address (A) and bank address (BA) inputs are used for addressing a memory device. Users can obtain this information from the device data sheet. Figure 2-8 shows the local-to-memory address mapping of the Lattice LPDDR memory controller core.

Figure 2-8. Local-to-Memory Address Mapping for Memory Access



Mode Register Programming

The LPDDR SDRAM memory devices are programmed using the mode registers MRS and EMRS. The bank address bus (em_ddr_ba) is used to choose one of the Mode registers, while the programming data is delivered through the address bus (em_ddr_addr). The memory data bus cannot be used for mode register programming. The Lattice LPDDR memory controller core uses the local address bus, addr, to program these registers. The core accepts a user command, LMR, to initiate the programming of mode registers. When LMR is applied on the cmd bus, the user logic must provide the information for the targeted 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. Table 2-6 shows how the local address lines are allocated for selecting mode registers. The addr[7:0] lines are used to provide the contents for the selected mode register.

Table 2-6. Mode Register Selection Using Bank Address Bits

Mode Register	(addr[9:8])
MRS	00
EMRS	10

Parameter Settings

The IPexpress™ tool is used to create IP and architectural modules in the Diamond design software. Table 3-1 provides a list of user-configurable parameters for this IP core. The parameter settings are specified using the LPDDR SDRAM Controller IP core Configuration GUI in IPexpress.

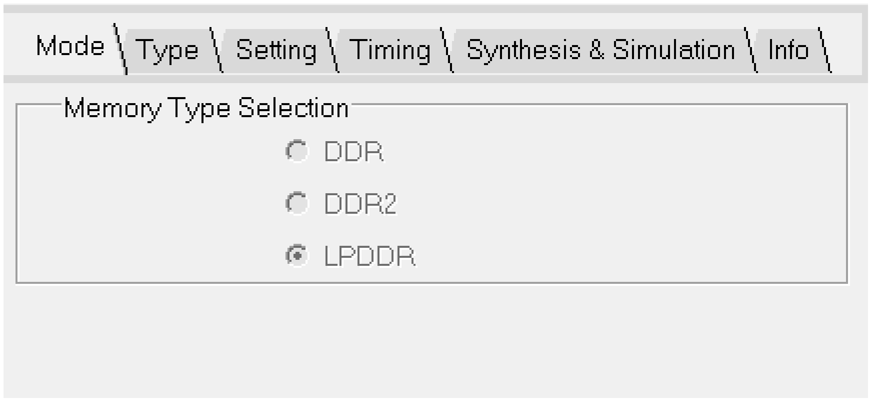
Table 3-1. LPDDR Core Configuration Parameters

Parameters in GUI	Range/Options	Default
Select Memory	MT46H8M16LF	MICRON MT46H8M16LF
	MT46H16M16LF	
	MT46H32M16LF	
	MT46H64M16LF	
	MT46H128M16LF	
	CUSTOM	
Clock	10~133	133MHz (Wishbone off, -6 SP grade) 100 MHz (All other cases)
Memory Data Bus Size	8, 16	16
Clock Width	1	1
Wishbone Bus	Disable, One Port, Two Ports	Disable
Row Size	12-14	12
Column Size	9-11	9
Bank Size	2	2
Chip Select Width	1	1
Auto refresh burst count	2-8	8
External Auto Refresh (User Refresh Control)	Selected/ Unselected	Unselected
Read Re-training	Auto Re-training User Control	Auto Re-training
Re-training Period (N)	8, 64, 256, 512, 1024	1024
Address Space for Training	‘h0 to {`ADDR_WIDTH{1’b1}} -4’hf	23`h000000
Read Data Auto Alignment	Selected/ Unselected	Unselected
Partial Array Self Refresh	Full, Half, Quarter	Full
Memory Clock OFF	Selected/ Unselected	Unselected
Burst Length	2, 4, 8, 16 with Generic, 2 with Wishbone	8 with Generic, 2 with Wishbone
Burst Type	Sequential, Interleave	Sequential
CAS Latency	3	3
Active to Read/Write (TRCD)	3~7	3
Active to Precharge (TRAS)	6~15	6
AutoRefresh Cmd Period (TRFC)	15~31	15
Load_Mode Cmd Period (TMRD)	2~3	2
Precharge Cmd Period (TRP)	3	3
Active to Active Cmd (TRC)	9~15	10
Avr. Refresh Cmd Interval (TREFI)	Max = Int(7.8xClock)	Int(7.8xClock)
Write to Read (TWTR)	1~3	1
Power Down to Next Cmd (TXP)	1~7	2
Min CKE High/Low (TCKE)	4~7	4
Exit Self Ref to Next Cmd (TXSR)	11~63	27
SRR to Read (TSRR)	2~3	2
SRR to Next Cmd (TSRC)	4~7	4
Write Recovery (TWR)	2~3	2

Mode Tab

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

Figure 3-1. Mode Options in the IPexpress Tool



Memory Type Selection

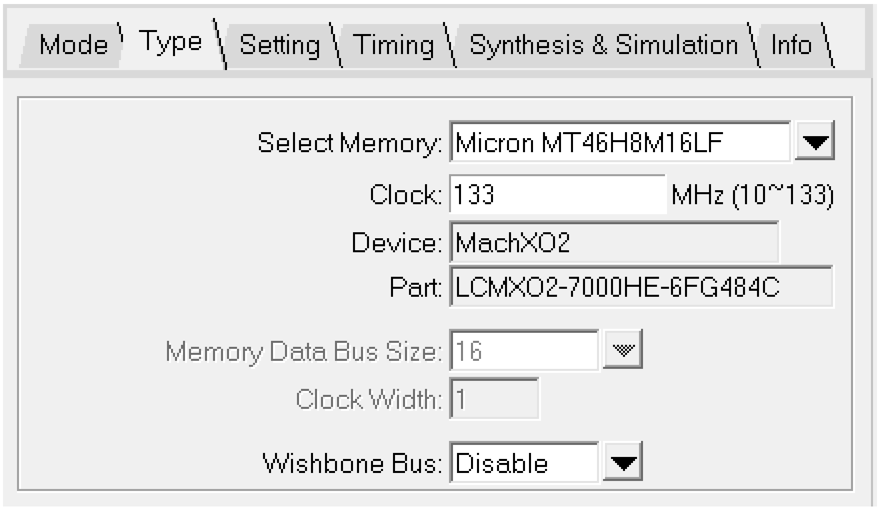
DDR [ ] DDR2 [✓ LPDDR]

Type Tab

The Type tab allows the user to select the LPDDR controller configuration for the target memory device and the core functional features. These parameters are considered as static parameters since the values for these parameters can only be set in the GUI. The LPDDR controller must be regenerated to change the value of any of these parameters.

Figure 3-2 shows the contents of the Type tab.

Figure 3-2. Type Options in the IPexpress Tool



Select Memory: Micron MT46H8M16LF

Clock: 133 MHz (10~133)

Device: MachXO2

Part: LCMXO2-7000HE-6FG484C

Memory Data Bus Size: 16

Clock Width: 1

Wishbone Bus: Disable

Select Memory

The LPDDR 128Mb x16 from Micron Technology® is provided as the default LPDDR memory. The timing parameters of this memory are listed in the Memory Device Timing tab as default values.

Clock

This parameter specifies the frequency of the memory clock to the on-board memory. The supported frequency range is from 10MHz up to 100MHz or 133MHz depending on the interface mode and the device speed grade.

Memory Data Bus Size

The MachXO2 device family provides a 8- or 16-bit I/O interface to the LPDDR memory.

Clock Width

The controller provides one differential clock for the LPDDR memory.

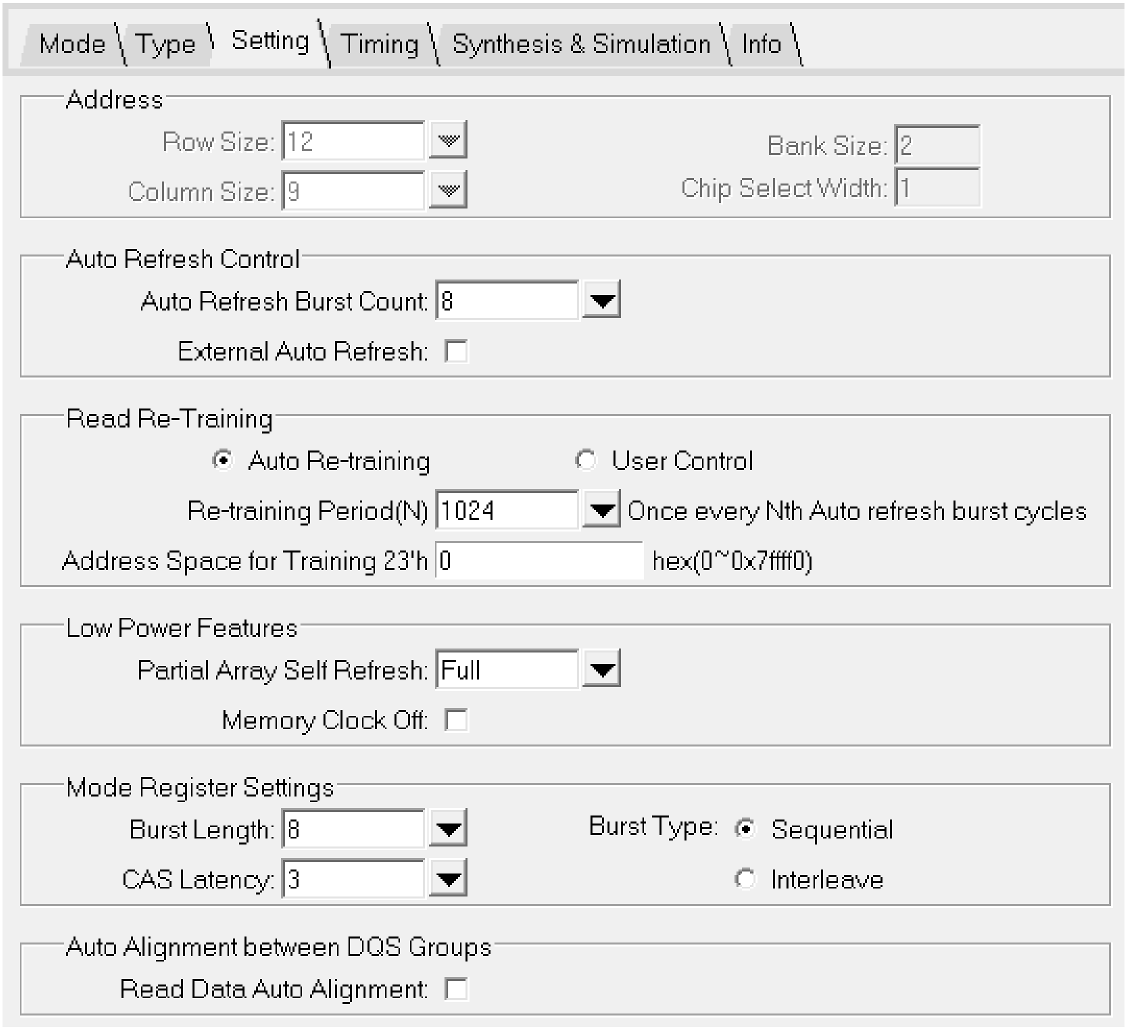
Wishbone Bus

The controller supports both the Lattice generic user interface and Wishbone bus. One Port or Two Ports option is selected to use the Wishbone bus.

Setting Tab

Figure 3-3 shows the contents of the Setting tab.

Figure 3-3. Setting Options in the IPexpress Tool



Address
Row Size: 12
Column Size: 9
Bank Size: 2
Chip Select Width: 1

Auto Refresh Control
Auto Refresh Burst Count: 8
External Auto Refresh: ☐

Read Re-Training
[✓ Auto Re-training] User Control
Re-training Period(N) 1024 Once every Nth Auto refresh burst cycles
Address Space for Training 23'h 0 hex(0~0x7fff0)

Low Power Features
Partial Array Self Refresh: Full
Memory Clock Off: ☐

Mode Register Settings
Burst Length: 8
CAS Latency: 3
Burst Type: [✓ Sequential] Interleave

Auto Alignment between DQS Groups
Read Data Auto Alignment: ☐

Row Size

This option indicates the Row Address size used in the selected memory configuration.

Column Size

This option indicates the Column Address size used in the selected memory configuration.

Auto Refresh Burst Count

This option indicates the number of Auto Refresh commands that the memory controller core is set to send in a single burst.

External Auto Refresh

This option, if selected, allows the user logic to generate an auto refresh request to the controller. If this option is not selected, the controller automatically generates refresh commands to the memory at the interval defined by the memory refresh timing requirement (TREFI) and Auto Refresh Burst Count.

Read Re-training

This option selects whether the periodic read re-training is governed by the controller or the user. If the User Control option is selected, two additional ports, rt_req and rt_done, are added to the user interface. If the Auto Re-training is selected, the core automatically initiates read re-training requests following the user selected training interval, Re-training Period.

Re-training Period

This option defines a training interval in terms of the number of auto refresh burst cycles. This option is available when the Auto Re-training option is selected with the 8th, 64th, 256th, 512th or 1024th auto refresh burst cycles to choose from. The controller will initiate read re-training once every selected number of auto refresh burst cycles.

Address Space for Training

This option assigns the location of training address space to the user address map. The size of training address space is determined by the selected burst length, and only the start address needs to be entered. BL2, BL4, BL8 or BL16 use 2, 4, 8 or 16 address locations respectively.

Read Data Auto Alignment

This option selects whether the controller automatically deskews and aligns the read data and read data valid signals or not. Once enabled, the read_data_valid signal becomes a single bit output fully aligned with the read data also deskewed between two DQS groups. This option should be enabled when dynamic skew is observed on the read data and two-bit read_data_valid between two DQS groups. It is generally expected that an implemented LPDDR controller core does not show a skew in most cases. If there is skew observed on the read data, however, assertion timing differences by one clock between two read_data_valid bits will also be dynamically observed. This dynamic skew can occur when a specific length of round trip delay of a system makes the incoming DQS edges aligned with the system clock, and the jitter on the LPDDR bus dynamically changes the clock relationships. When this dynamic skew is observed, the core needs to be regenerated with the Read Data Auto Alignment option enabled. The auto alignment mode removes the dynamic skew and aligns the read data with the read data valid signal. Since the auto alignment mode adds about three clocks of additional latency with increased lut counts, it is recommended that this option be enabled only when dynamic skew is observed. When disabled, either of two read_data_valid bits can be used to validate the read data because they are identical when there is no skew observed.

Partial Array Self Refresh

For power saving purposes, the controller has the capability to refresh a quarter, a half or full memory, if the user is accessing only a quarter, a half or full memory, during self refresh operations.

Memory Clock

This is an on/off option and if enabled, the controller will turn the memory clock off during self refresh, power down and deep power down operations.

Burst Length

This option sets the burst length value in the mode register during initialization. When Wishbone is selected, only the BL2 mode is supported.

CAS Latency

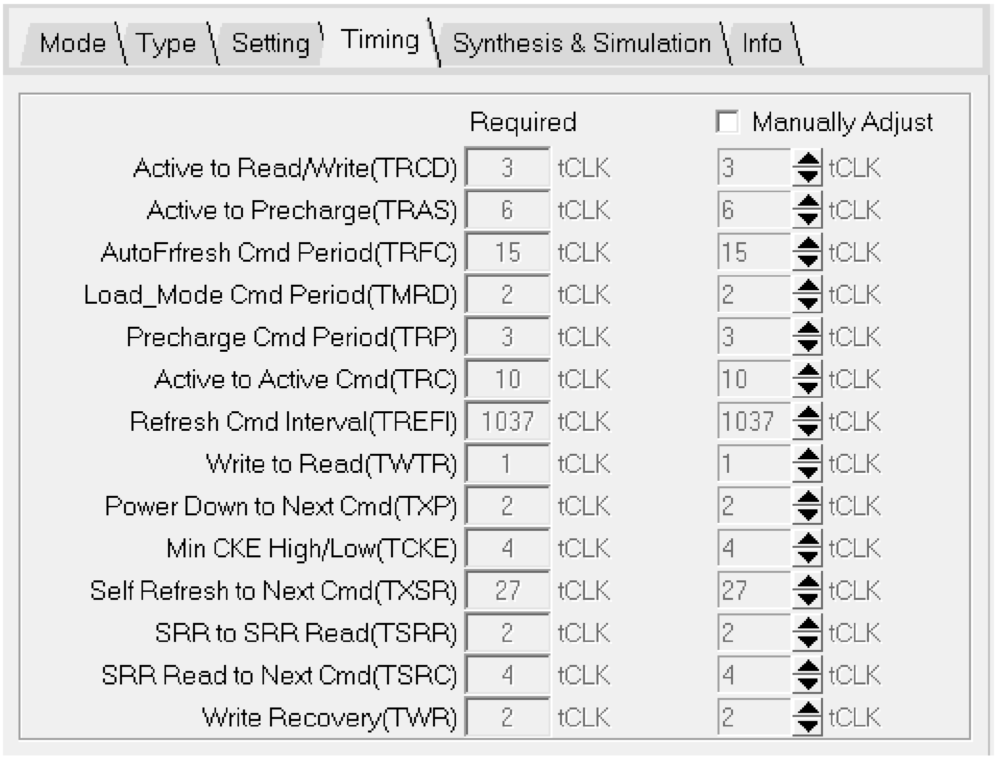
This option sets the CAS latency value in the mode register during initialization.

Burst Type

This option sets the burst type value in the mode register during initialization.

Memory Device Timing Tab

Figure 3-4 shows the contents of the Memory Device Timing tab.



	Required		☐ Manually Adjust	
Active to Read/Write(TRCD)	3	tCLK	3	tCLK
Active to Precharge(TRAS)	6	tCLK	6	tCLK
AutoFrfresh Cmd Period(TRFC)	15	tCLK	15	tCLK
Load_Mode Cmd Period(TMRD)	2	tCLK	2	tCLK
Precharge Cmd Period(TRP)	3	tCLK	3	tCLK
Active to Active Cmd(TRC)	10	tCLK	10	tCLK
Refresh Cmd Interval(TREFI)	1037	tCLK	1037	tCLK
Write to Read(TWTR)	1	tCLK	1	tCLK
Power Down to Next Cmd(TXP)	2	tCLK	2	tCLK
Min CKE High/Low(TCKE)	4	tCLK	4	tCLK
Self Refresh to Next Cmd(TXSR)	27	tCLK	27	tCLK
SRR to SRR Read(TSRR)	2	tCLK	2	tCLK
SRR Read to Next Cmd(TSRC)	4	tCLK	4	tCLK
Write Recovery(TWR)	2	tCLK	2	tCLK

Figure 3-4. Memory Device Timing Options in the IPexpress Tool

Manually Adjust

Checking this box allows users to manually set (via increment/decrement) any of the memory timing parameters. If the user needs to change any of the default values, the Manually Adjust checkbox must be checked. This selection will enable the user to increment/decrement the memory timing parameters.

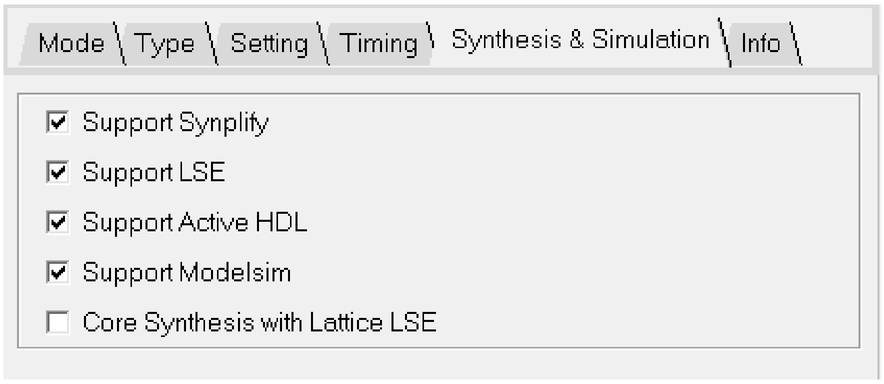
tCLK – Memory Clock

This is a notation signifying that the memory timing parameters shown in this tab are specified in terms of tCLK LPDDR memory clock cycles.

Synthesis and Simulation Tab

The Synthesis and Simulation tab enables the user to select the simulation and synthesis tools to be used for generating a design. Figure 3-5 shows the contents of the Design Tools Options and Info tab.

Figure 3-5. Synthesis and Simulation Options in the IPexpress Tool



☑ Support Synplify
☑ Support LSE
☑ Support Active HDL
☑ Support Modelsim
☐ Core Synthesis with Lattice LSE

The tab supports the following parameters:

Support Synplify

If selected, IPexpress generates evaluation scripts and other associated files required to synthesize the top-level design using the Synopsys Synplify synthesis tool.

Support LSE

If selected, IPexpress generates evaluation scripts and other associated files required to synthesize the top-level design using LSE.

Support ModelSim

If selected, IPexpress generates evaluation script and other associated files required to synthesize the top-level design using the Mentor Graphics ModelSim simulator.

Support Active HDL

If selected, IPexpress generates evaluation script and other associated files required to synthesize the top-level design using the Aldec® Active-HDL® simulator.

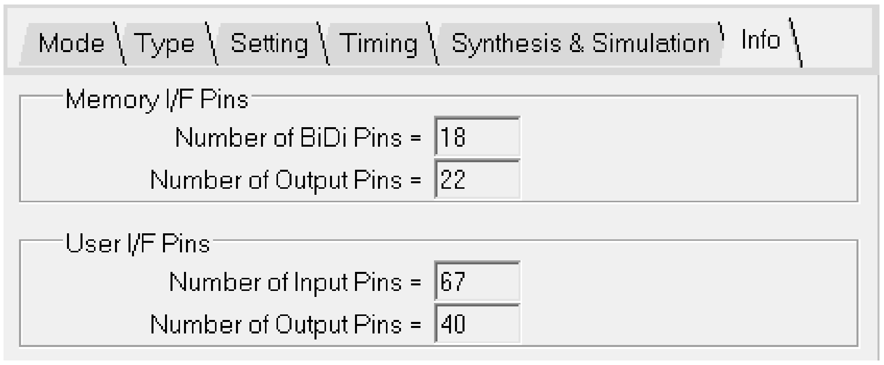
Core Synthesis with Lattice LSE

If selected, IPexpress generates the encrypted netlist for the core using LSE.

Info Tab

This tab provides information about the pinout resources used. Figure 3-6 shows the contents of the Info tab.

Figure 3-6. Info Tab in the IPexpress Tool



Memory I/F Pins

Number of BiDi Pins = 18

Number of Output Pins = 22

User I/F Pins

Number of Input Pins = 67

Number of Output Pins = 40

Memory I/F Pins

This section displays information for the LPDDR memory interface pins.

Number of BiDi Pins
This is a notification of the number of bi-directional pins used in the memory side interface for the selected configuration. Bi-directional pins are used for Data (DQ) and Data Strobe (DQS) signals only.

Number of Output Pins
This is a notification of the number of output-only pins used in the memory side interface for the selected configuration. Output-only pins are used for LPDDR Address, Command and Control signals.

User I/F Pins

This section displays the information for the user interface pins.

Number of Input Pins
This is a notification of the number of input pins used in the user interface for the selected configuration.

Number of Output Pins
This is a notification of the number of output pins used in the user interface for the selected configuration.

IP Core Generation

This chapter provides information on how to generate the LPDDR SDRAM IP core using the Diamond IPexpress tool, and how to include the core in a top-level design.

The LPDDR SDRAM IP core can be used in the MachXO2 device family. For example information and known issues on this IP core see the Lattice LPDDR IP ReadMe document. This file is available once the core is installed in Diamond. The document provides information on creating an evaluation version of the core for use in simulation.

Users may download and generate the LPDDR SDRAM IP core and fully evaluate the core through functional simulation and implementation (synthesis, map, place and route) without an IP license. The LPDDR SDRAM 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 30 for further details. However, a license is required to enable timing simulation, to open the design in the Diamond EPIC tool, and to generate bitstreams that do not include the hardware evaluation timeout limitation.

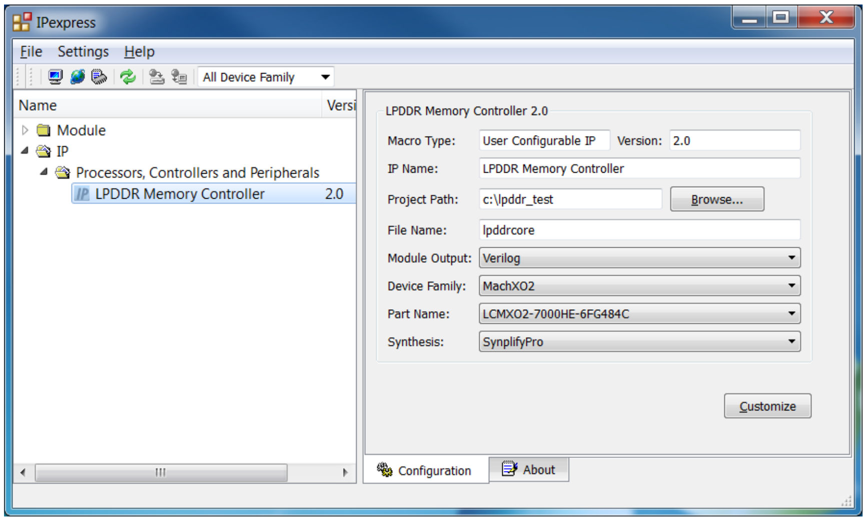
Getting Started

The LPDDR SDRAM 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 LPDDR SDRAM 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.
- **Module Output** – Verilog or VHDL.
- **Device Family** – Device family to which IP is to be targeted. 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



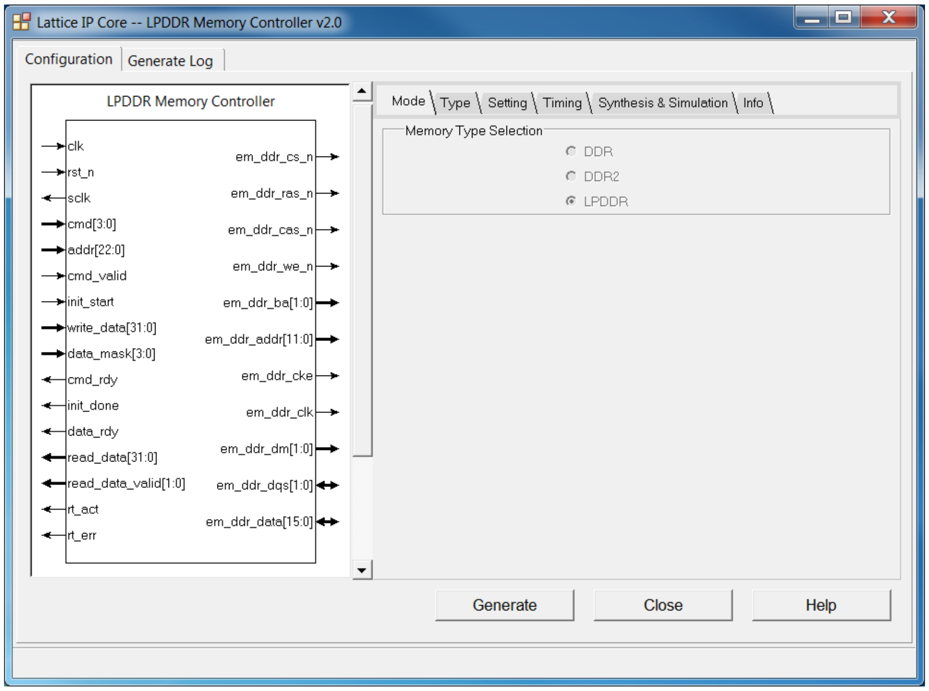
LPDDR Memory Controller 2.0

Macro Type: User Configurable IP   Version: 2.0
IP Name: LPDDR Memory Controller
Project Path: c:\lpddr_test
File Name: lpddrcore
Module Output: Verilog
Device Family: MachXO2
Part Name: LCMXO2-7000HE-6FG484C
Synthesis: SynplifyPro

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 LPDDR 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

Figure 4-2. LPDDR SDRAM IP Configuration GUI



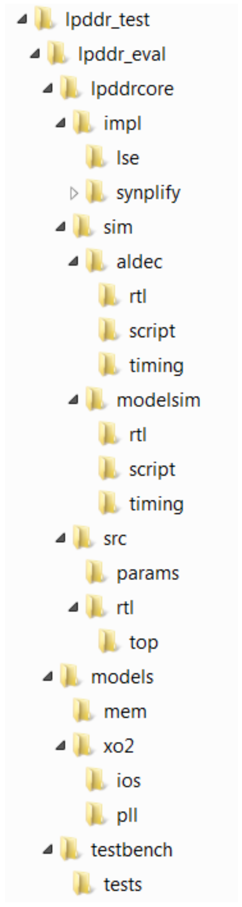
Memory Type Selection

DDR
DDR2
[✓ LPDDR]

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. MachXO2 LPDDR Core Directory Structure



```
lpddr_test
  lpddr_eval
    lpddrcore
      impl
        lse
        synplify
      sim
        aldec
          rtl
          script
          timing
        modelsim
          rtl
          script
          timing
      src
        params
        rtl
          top
    models
      mem
      xo2
        ios
        pll
    testbench
      tests
```

LPDDR SDRAM Controller IP Core File Structure

Table 4-1 provides a list of key files and directories created by the IPexpress tool and how they are used. The IPexpress tool creates several files that are used throughout the design cycle. The names of most of the created files are customized to the user's module name specified in the IPexpress tool.

Table 4-1. File List

File	Description
<*username*>.v	This file provides the LPDDR SDRAM core for simulation.
<*username*>_beh.v	This file provides a behavioral simulation model for the LPDDR SDRAM core.
<*username*>_bb.v	This file provides the synthesis black box for the user's synthesis.
<*username*>.ngo	The ngo files provide the synthesized IP core.
<*username*>.lpc	This file contains the IPexpress tool options used to recreate or modify the core in the IPexpress tool.
<*username*>.ipx	The IPX file holds references to all of the elements of an IP or Module after it is generated from the IPexpress tool. The file is used to bring in the appropriate files during the design implementation and analysis. It is also used to re-load parameter settings into the IP/Module generation GUI when an IP/Module is being re-generated.

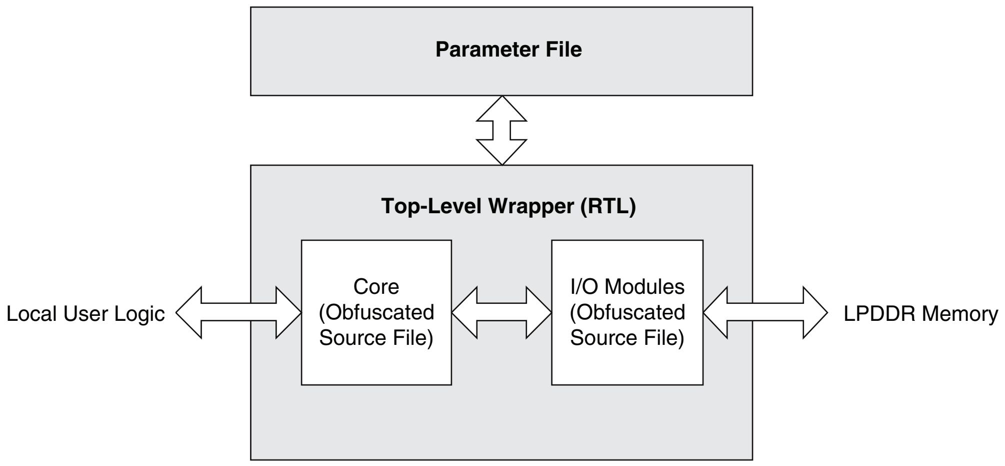
Table 4-1. File List (Continued)

File	Description
<*username*>_top.[v,vhd]	This file provides a module which instantiates the LPDDR SDRAM core. This file can be easily modified for the user's instance of the LPDDR SDRAM core. This file is located in the `lpddr_eval/<username>/src` directory.
<*username*>_generate.tcl	This file is created when GUI "Generate" button is pushed. This file may be run from command line.
<*username*>_generate.log	This is the IPexpress scripts log file.
<*username*>_gen.log	This is the IPexpress IP generation log file
<*username*>.v	This file provides the LPDDR SDRAM core for simulation.

The LPDDR SDRAM Controller IP core consists of the following four major functional blocks:

- Top-level wrapper (RTL)
- Obfuscated memory controller core and I/O modules for simulation and encrypted netlist memory controller core for synthesis
- Clock generator (RTL for simulation and Verilog flow synthesis or netlist for VHDL flow synthesis) All of these blocks are required to implement the IP core on the target device. Figure 4-4 depicts the interconnection among those blocks.

Figure 4-4. File Structure of LPDDR SDRAM Controller IP Core


Parameter File
Top-Level Wrapper (RTL)
Local User Logic
Core (Obfuscated Source File)
I/O Modules (Obfuscated Source File)
LPDDR Memory

Top-level Wrapper

The obfuscated core, obfuscated I/O modules, and the clock generator block are instantiated in the top-level wrapper. When a system design is made with the LPDDR SDRAM Controller IP Core, this wrapper must be instantiated. The wrapper is fully parameterized by the generated parameter file.

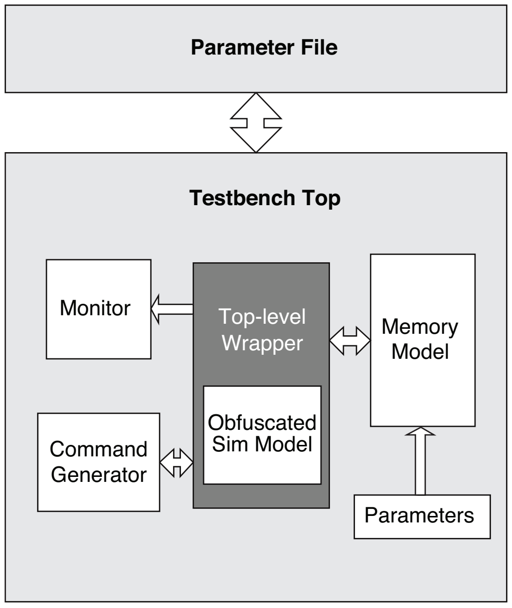
Obfuscated Module for the Core and I/O Modules

Obfuscated RTL files are used for simulation. The obfuscated core contains the memory controller function and I/O modules that interface with the user logic on one side and with the LPDDR memory on the other side. An equivalent netlist of the core is provided for synthesis.

Simulation Files for IP Core Evaluation

Once a LPDDR SDRAM Controller IP core is generated, it contains a complete set of test bench files to simulate a few example core activities for evaluation. The simulation environment for the LPDDR memory controller IP is shown in Figure 4-5. This structure can be reused by system designers to accelerate their system validation.

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



Test Bench Top

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

Obfuscated Core and I/O Simulation Models

The obfuscated simulation model for the core and I/O modules are instantiated in the top-level wrapper. This obfuscated simulation model must be included in the simulation.

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 block monitors both the local user interface and LPDDR interface activities and generates 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.

Memory Model

The LPDDR SDRAM Controller test bench uses a memory simulation model provided by one of the most popular memory vendors. If a different memory model is required, it can be used 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 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

ModelSim and Aldec Active-HDL simulation macro script files are included for instant evaluation of the IP core. All required files for simulation are included in the macro script. This simulation script can be used as a starting point of a user simulation project.

Instantiating the Core

The generated LPDDR SDRAM IP core package includes black-box (<*username*>_bb.v) and instance (<*username*>_inst.v) templates that can be used to instantiate the core in a top-level design. An example RTL top-level reference source file that can be used as an instantiation template for the IP core is provided in `\<project_dir>\lpddr_eval\<username>\src\rtl\top`. Users may also use this top-level reference as the starting template for the top-level for their complete design.

Running Functional Simulation

Simulation support for the LPDDR SDRAM IP core is provided for Aldec and ModelSim simulators. The LPDDR SDRAM core simulation model is generated from the IPexpress tool with the name <*username*>.v. This file calls <*username*>_beh.v which contains the obfuscated simulation model. An obfuscated simulation model is Lattice's unique IP protection technique which scrambles the Verilog HDL while maintaining logical equivalence. VHDL users will use the same Verilog model for simulation.

The ModelSim environment is located in `\<project_dir>\lpddr_eval\<username>\sim\modelsim`. Users can run the ModelSim simulation by performing the following steps:

1. Open ModelSim.
2. Under the File tab, select **Change Directory** and choose folder `\<project_dir>\lpddr_eval\<username>\sim\modelsim`.
3. Under the Tools tab, select **Tcl > Execute Macro** and execute the ModelSim "do" script shown.

The Aldec Active-HDL environment is located in `\<project_dir>\lpddr_eval\<username>\sim\aldec`. Users can run the Aldec evaluation simulation by performing the following steps:

1. Open Active-HDL.
2. Under the Tools tab, select **Execute Macro**.
3. Browse to the directory `\<project_dir>\lpddr_eval\<username>\sim\aldec` and execute the Active-HDL "do" script shown.

Synthesizing and Implementing the Core in a Top-Level Design

The LPDDR SDRAM IP core itself is synthesized and provided in NGO format when the core is generated through the IPexpress tool. You can combine the core in your own top-level design by instantiating the core in your top level file as described in "Instantiating the Core" on page 29 and then synthesizing the entire design with LSE or Synplify RTL Synthesis.

Push-button implementation of this top-level design with LSE or Synplify RTL Synthesis is supported via the project files <username>_eval.ldf located in the `\<project_dir>\lpddr_eval\<username>\impl\LSE (or synplify)` directory.

To use this project file:

1. Choose **File** > **Open** > **Project**.
2. Browse to `\<project_dir>\lpddr_eval\<username>\impl\LSE (or synplify)` in the Open Project dialog box.
3. Select and open <username>_eval.ldf. At this point, all of the files needed to support top-level synthesis and implementation will be imported to the project.
4. Select the **Process** tab in the left-hand GUI window.
5. Implement the complete design via the standard Diamond GUI flow.

Hardware Evaluation

The LPDDR SDRAM 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

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.

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

To regenerate an IP core:

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.

Chapter 5:

Application Support

This chapter provides application support information for the MachXO2 LPDDR SDRAM Controller IP core.

Core Implementation

This section describes the major factors that are important for a successful LPDDR memory controller implementation.

Understanding Preferences

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

- **FREQUENCY** – The LPDDR SDRAM Controller IP core is normally over-constrained for obtaining optimal f_{MAX} results.
- **IOBUF** – The IOBUF preference assigns the required I/O types to the LPDDR I/O pads.
- **LOCATE** – For all MachXO2 devices the memory data and data strobe pins are located at the right side of the device. For this reason these I/Os are grouped and locked to Bank 1.

Core Verification

The functionality of the MachXO2 LPDDR SDRAM Controller IP core has been verified via simulation with LPDDR simulation models from Micron Technology, including:

- Simulation environment verifying proper LPDDR functionality using Lattice's proprietary verification environment.

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.

E-mail Support

techsupport@latticesemi.com

Local Support

Contact your nearest Lattice Sales Office.

Internet

www.latticesemi.com

References

- DS1035, *MachXO2 Family Data Sheet*

JEDEC Web Site

The JEDEC website contains specifications and documents referred to in this user's guide. The JEDEC URL is: www.jedec.org.

Revision History

Date	Document Version	IP Core Version	Change Summary
November 2010	01.0	1.0	Initial release.
December 2011	01.1	1.0	Updated Quick Facts table.
			Updated LPDDR Core Configuration Parameters table.
			Appendix A – Updated Performance and Resource Utilization table for MachXO2.
			Removed references to ispLEVER® design software.
October 2012	01.2	1.0	Updated document with new corporate logo.
			LPDDR SDRAM Memory Controller Top-Level I/O List for Generic Interface – Clarified description for the read_data_valid[1:0] port.
February 2014	01.3	2.0	Added description for the read data auto deskew/alignment function.
			Updated signal descriptions table: - Added ports for user read re-training and user refresh control. - Signal name changes (datain, dmsel -> write_data, data_mask).
			Updated diagrams for new and revised functions.
			Added LSE synthesis flow.
			Updated descriptions for the core generation GUI.
			Updated Lattice Technical Support information.

Resource Utilization

This appendix gives resource utilization information for Lattice FPGAs using the LPDDR SDRAM Controller IP core. IPexpress is the Lattice IP configuration utility, and is included as a standard feature of the Diamond design software. Details regarding the use of IPexpress can be found in the IPexpress and Diamond help systems. For more information on the Diamond design software, visit the Lattice web site at: www.latticesemi.com.

MachXO2 Devices

Table A-1 lists performance and resource utilization data for four LPDDR SDRAM configurations.

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

IP Core	Parameter Settings	Slices	LUTs	Registers	I/O	f_{MAX} (MHz)[2]
LPDDR 16-bit, Generic	Default parameters	883	1678	880	147	133
LPDDR 8-bit, Generic		859	1622	849	102	133
LPDDR 16-bit, Wishbone 1 Port		787	1524	933	143	100
LPDDR 8-bit, Wishbone 1 Port		722	1384	835	99	100

1. Performance and utilization data are generated using a LCMXO2-7000HE-6FG484C in Lattice Diamond 3.1 design software. Performance may vary when using this IP core in a different density, speed or grade within the MachXO2 family.
2. The LPDDR IP core can operate up to 133 MHz (266Mbps LPDDR) in the fastest speed-grade device (-6) when the Lattice generic interface is used.

Ordering Part Number

The Ordering Part Number (OPN) for the LPDDR SDRAM Controller IP on MachXO2 devices is:
LPDDRCT-WB-M2-U