



Lattice**CORE**

MachXO2 LPDDR SDRAM Controller IP Core User's Guide

Chapter 1. Introduction	4
Introduction	4
Quick Facts	4
Features	4
Chapter 2. Functional Description	5
Overview	5
Initialization Block	5
I/O Training Block	6
Data Control Block	6
LPDDR I/Os	6
Command Application Logic Block	6
Command Decode Logic Block	6
Signal Descriptions	6
Using the Local User Interface	8
Initialization Control	8
Command and Address	8
User Commands	9
Power Down and Deep Power Down	11
User Commands for Wishbone Interface	11
Local-to-Memory Address Mapping	12
Mode Register Programming	13
Chapter 3. Parameter Settings	14
Mode Tab	14
Type Tab	15
Select Memory	15
Clock	16
Devices	16
Memory Data Bus Size	16
Data_rdy to Write Data Delay	16
Clock Width	16
Setting Tab	16
Row Size	16
Column Size	16
I/O Auto Training	17
Periodic I/O Auto Retraining	17
Partial Array Self Refresh	17
Memory Clock	17
Burst Length	17
CAS Latency	17
Burst Type	17
Memory Device Timing Tab	18
Synthesis and Simulation Tab	18
Support Synplify	19
Support ModelSim	19
Support Aldec	19
Info Tab	19
Memory Interface Pins	19
Number of Bi-directional Pins	19
Number of Output Pins	19

Getting Started	20
Chapter 4. IP Core Generation.....	20
IPexpress-Created Files and Top Level Directory Structure	22
LPDDR SDRAM Controller IP Core File Structure	22
Top-level Wrapper.....	23
Obfuscated Module for the Core	23
Obfuscated Module for the I/O Modules	23
Simulation Files for IP Core Evaluation.....	24
Test Bench Top	24
Obfuscated Core and I/O Simulation Models.....	24
Command Generator	24
Monitor	24
Memory Model	24
Memory Model Parameter.....	24
Evaluation Script File	25
Instantiating the Core	25
Running Functional Simulation	25
Synthesizing and Implementing the Core in a Top-Level Design	25
Hardware Evaluation.....	26
Enabling Hardware Evaluation.....	26
Updating/Regenerating the IP Core	26
Regenerating an IP Core	26
Chapter 5. Application Support.....	28
Core Implementation.....	28
Understanding Preferences	28
Chapter 6. Core Verification	29
Chapter 7. Support Resources	30
Lattice Technical Support.....	30
Online Forums.....	30
Telephone Support Hotline	30
E-mail Support	30
Local Support.....	30
Internet.....	30
References.....	30
JEDEC Web Site.....	30
Revision History	31
Appendix A. Resource Utilization	32
MachXO2 Devices	32
Ordering Part Number	32

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

Core Requirements	FPGA Family	MachXO2			
	Targeted Devices	LCMXO2-7000HE-6BG256C			
	Configuration	Configuration 1	Configuration 2	Configuration 3	Configuration 4
Resource Utilization	Registers	911	804	969	969
	Slices	814	705	729	729
	LUTs	1530	1314	1409	1409
	EBRs	0	0	0	0
	f _{MAX} (MHz)	136.4	136.6	142.9	142.9
Design Tool	Lattice Implementation	Lattice Diamond® 1.3			
Support	Synthesis	Synopsys® Synplify™ Pro for Lattice F-2011.09L			
	Simulation	Mentor Graphics® ModelSim™ SE 6.3F			

Features

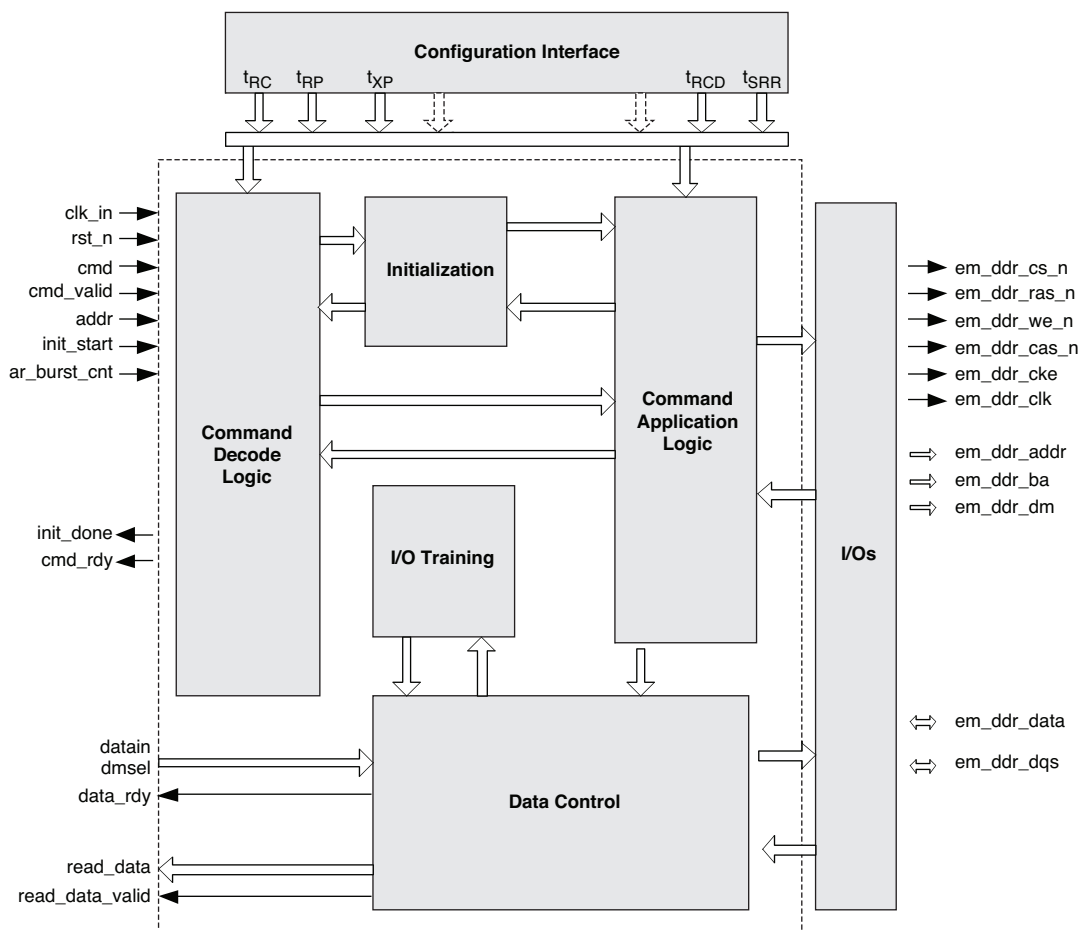
- Interfaces to industry standard LPDDR SDRAM according to JESD209B
- Double-data rate architecture; two data transfers per clock cycle
- Bi-directional data strobe per byte of data (DQS)
- Programmable auto refresh support
- Data mask support – one mask per byte
- Power down and deep power down support
- Supports power-on initialization
- Supports re-initialization after a deep power down
- 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
- Supports single-port operation
- Supports full, half and quarter array self refresh
- 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 I/O Training Block. LPDDR I/O modules provide the PHY interface to the memory device. This block mostly consists of MachXO2 device I/O primitives supporting compliance 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 must be initialized before sending any command to the Controller. It is the 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 provided by this block.

This module is automatically activated at the exit of a deep power down operation and the LPDDR memory is re-initialized.

I/O Training Block

The I/O 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 auto refresh burst. It is an IPexpress GUI-programmable parameter. The user may choose to perform training each time the memory is initialized or periodically at the end of every auto refresh burst. The memory reserved space for training patterns is for addresses 14'h0 - 14'hF of row 0 of bank 0.

Data Control Block

The Data Control Block interfaces with the LPDDR I/O modules and is responsible for generating 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 Application Logic Block

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

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 opened bank. The controller implements a command pipeline to improve throughput. With this capability, the next command in the queue is decoded while the current command is presented at the memory interface.

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	In	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

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

Port Name	Active State	I/O	Description
data_mask[(DSIZE/8)-1:0]	N/A	Input	Data mask input for write_data.
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[1:0]	High	Output	Read data valid output. When asserted, it indicates the data on the read_data bus is valid. The two bits are independent outputs of the two DQSBUFH used in the design. If the LPDDR IO training and alignment is achieved, the bits of the read_data_valid bus should be identical.

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
PORT0_ARRD_I[31:0], PORT1_ARRD_I[31:0]	Input	The address input array used to pass a binary address. For LPDDR memory of: 1G : PORT0_ARRD_I[25:0] = valid address PORT0_ARRD_I[31:26] = unused 512M : PORT0_ARRD_I[24:0] = valid address PORT0_ARRD_I[31:25] = unused 256M : PORT0_ARRD_I[23:0] = valid address PORT0_ARRD_I[31:24] = unused 128M : PORT0_ARRD_I[22:0] = valid address PORT0_ARRD_I[31:23] = unused 64M : PORT0_ARRD_I[21:0] = valid address PORT0_ARRD_I[31:22] = unused
PORT0_DAT_I[31:0], PORT1_DAT_I[31:0]	Input	The data input array used for write data
PORT0_SEL_I[4:0], PORT1_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.
PORT0_WE_I, PORT1_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
PORT0_CYC_I[2:0], PORT1_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.
PORT0_CLK_I, PORT1_CLK_I	Input	Wishbone input clocks
RST_I	Input	Wishbone reset

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

Port Name	I/O	Description
PORT0_DAT_O[31:0], PORT1_DAT_O[31:0]	Output	The data Output array used for read data
PORT0_ACK_O, PORT1_ACK_O	Output	The acknowledge output ACK_O, when asserted, indicates the termination of a normal bus cycle by the slave
PORT0_ERR_O, PORT1_ERR_O	Output	The error output ERR_O indicates an abnormal cycle termination by the slave.
PORT0_RTY_O, PORT1_RTY_O	Output	The retry output RTY_O indicates that the slave interface is not ready to accept or send data

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](#).

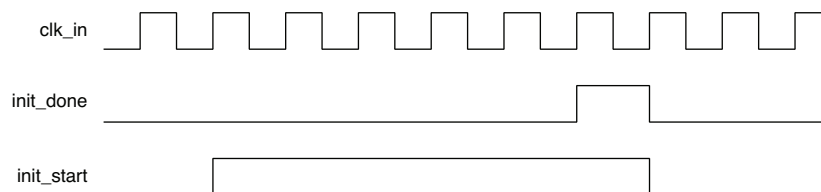
Table 2-3. Local User Interface Functional Groups

Functional Group	Signals
Initialization Control	init_start, init_done
Command and Address	addr, cmd, cmd_rdy, cmd_valid
Data Write	datain_rdy, write_data, data_mask
Data Read	read_data, read_data_valid

Initialization Control

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 only once, immediately after the system reset.

Figure 2-2. Timing of Memory Initialization Control

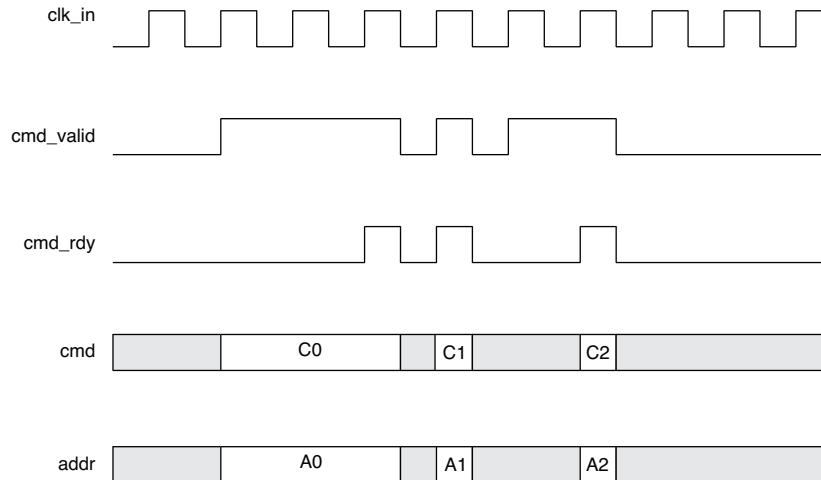


Command and Address

Once the memory initialization is done, the core waits for user commands in order to set up and/or access the memory. 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 following procedure. The memory controller core tells

the user logic that it is ready to receive a command by asserting the cmd_rdy signal for one cycle. If the core finds the cmd_valid signal asserted by the user logic while cmd_rdy is asserted, it takes the cmd input as a valid user command. cmd_valid should be de-asserted at the rising edge of the clock that samples cmd_rdy high. The core also accepts the addr input as a valid start address or mode register programming data depending on the command type. When the core reaches the boundary of the current page while accessing the memory, the next address that the core will access becomes the beginning of the same page. It will cause overwriting of the contents of the location or reading unexpected data. Therefore, the user must track the accessible address range in the current page while the command burst operation is performed.

Figure 2-3. 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 defines a set of valid memory commands as shown in [Table 2-4](#). All other values should not be used.

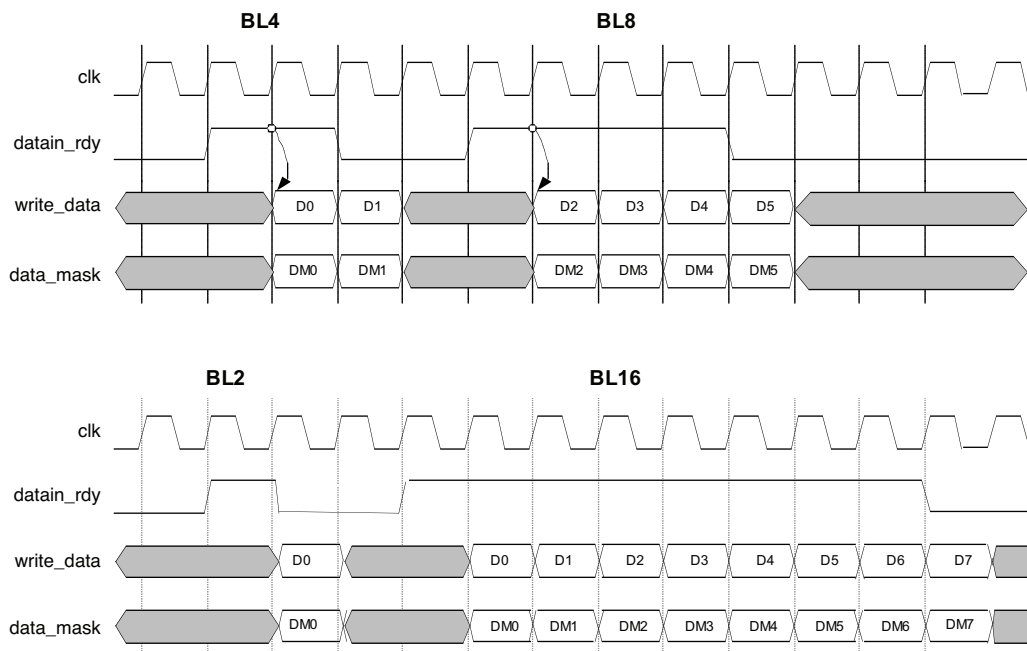
Table 2-4. Defined User Commands for Generic Interface

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 2-4 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 2-4. One-Clock Write Data Delay



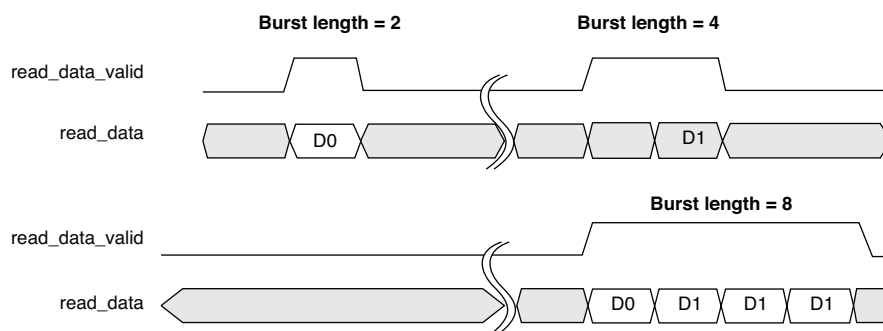
WRITEA

`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 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 2-5. 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 description mentioned in the `WRITE` operation for more detail.

Figure 2-5. 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 Lattices LPDDR memory controller core can support 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 ($t_{REFI} \times 8$). Bursting refresh cycles increases the LPDDR bus throughput because it helps keep core intervention to a minimum. Upon completion of an Auto Refresh burst, the controller will automatically retrain the I/Os if the periodic retraining of the I/Os is selected from the IPexpress GUI.

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 upon exiting the deep power down state, and retrain the I/Os if it is selected from the GUI.

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 memory.

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

PORT1_ADDR_I	PORT1_DATA_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

The memory addresses shown in [Table 2-6](#) are reserved for the controller and should not be used by the Port 0 user.

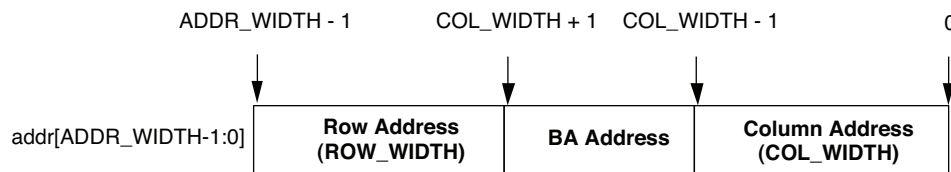
Table 2-6. Reserved Memory Address for Port 0

PORT0_ADDR_I	Description
32'h0 – 32'h10	For Port0, responsible of Read/Writes, the memory addresses 32'h0000_0000 to 32'h0000_0010 are reserved and should not be used by the Port 0 user.

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 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 2-6](#) shows the local-to-memory address mapping of the Lattice LPDDR memory controller core.

Figure 2-6. 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 2-7](#) shows how the local address lines are allocated for the programming of memory registers.

Table 2-7. 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.

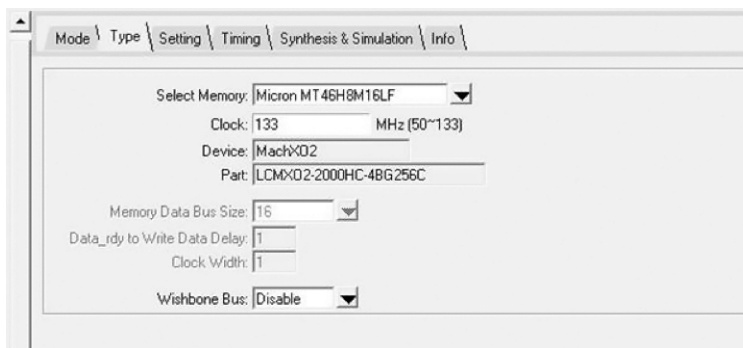
Table 3-1. LPDDR Core Configuration Parameters

Parameter	Configuration			
	Configuration 1	Configuration 2	Configuration 3	Configuration 4
Design Entry	Verilog HDL			
Device Family	MachXO2			
Part Name	LCMXO2-7000HE-6BG256C			
Memory	Micron MT46H64M16LF			
Clock	133 MHz			
Data Width	16			
Wishbone	Disable	Disable	One Port	One Port
Row Width	12	12	12	12
Col Width	9	9	9	9
Auto Refresh Burst Count	8	8	8	2
IO Auto Training	Enable	Disable	Enable	Enable
Periodic IO Auto Retraining	Enable	Disable	Enable	Enable
Partial Array Self Refresh	Full			
Memory Clock off	Disable			
Burst Length	4			
Burst Type	Sequential			
TRCD	3			
TRAS	8			
TRFC	15			
TMRD	2			
TRP	3			
TRC	12			
TREFI	1040			
TWTR	2			
TXP	2			
TCKE	4			
TXSR	27			
TSRR	3			
TSRC	2			
TWR	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



The screenshot shows the 'Mode' tab in the IPexpress Tool. The settings are as follows:

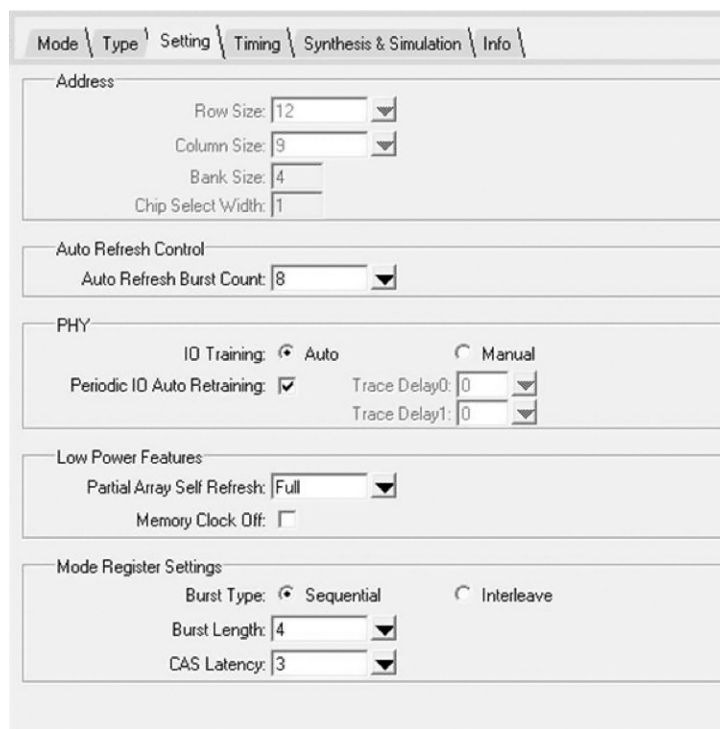
- Select Memory: Micron MT46H8M16LF
- Clock: 133 MHz (50~133)
- Device: MachXO2
- Part: LCMXO2-2000HC-4BG256C
- Memory Data Bus Size: 16
- Data_rdy to Write Data Delay: 1
- Clock Width: 1
- Wishbone Bus: Disable

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



The screenshot shows the 'Type' tab in the IPexpress Tool. The settings are as follows:

- Address**
 - Row Size: 12
 - Column Size: 9
 - Bank Size: 4
 - Chip Select Width: 1
- Auto Refresh Control**
 - Auto Refresh Burst Count: 8
- PHY**
 - IO Training: ☒ Auto ☐ Manual
 - Periodic IO Auto Retraining: ☒
 - Trace Delay0: 0
 - Trace Delay1: 0
- Low Power Features**
 - Partial Array Self Refresh: Full
 - Memory Clock Off: ☐
- Mode Register Settings**
 - Burst Type: ☒ Sequential ☐ Interleave
 - Burst Length: 4
 - CAS Latency: 3

Select Memory

The LPDDR 3 1GB-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 frequency is 100 MHz or less.

Devices

The MachXO2 device family provides a LPDDR JEDEC-compliant PHY and memory controller.

The devices available for evaluation are LCMXO2-2000HC-6BG256CES, LCMXO2-4000HC-6BG256CES and LCMXO2-7000HC-6BG256CES.

Memory Data Bus Size

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

Data_rdy to Write Data Delay

User logic is allowed to send the write data to the controller after a one-clock cycle delay with respect to the datain_rdy signal. Refer to [“WRITE” on page 8](#) for more information.

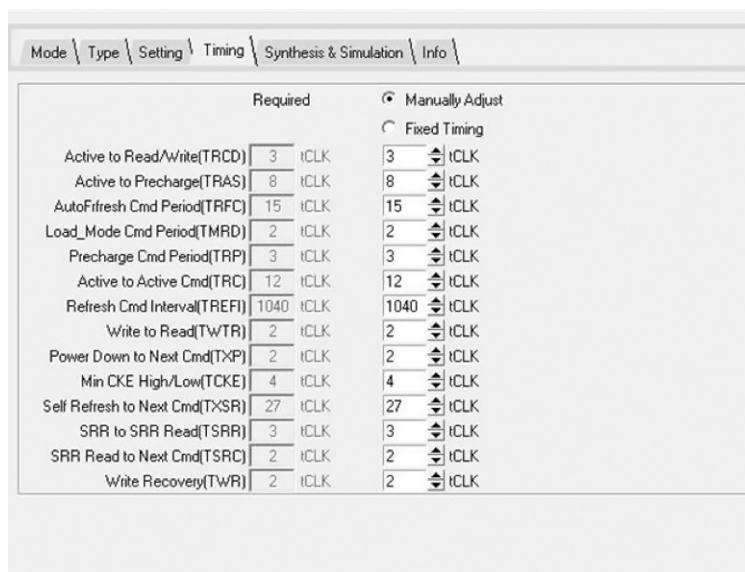
Clock Width

The controller provides one differential clock for the LPDDR memory.

Setting Tab

[Figure 3-3](#) shows the contents of the Setting tab.

Figure 3-3. Setting Options in the IPexpress Tool



	Required	Manually Adjust	Fixed Timing
Active to Read/Write(TRCD)	3 iCLK	3 iCLK	
Active to Precharge(TRAS)	8 iCLK	8 iCLK	
AutoRefresh Cmd Period(TRFC)	15 iCLK	15 iCLK	
Load_Mode Cmd Period(TMRD)	2 iCLK	2 iCLK	
Precharge Cmd Period(TRP)	3 iCLK	3 iCLK	
Active to Active Cmd(TRC)	12 iCLK	12 iCLK	
Refresh Cmd Interval(TREFI)	1040 iCLK	1040 iCLK	
Write to Read(TwTR)	2 iCLK	2 iCLK	
Power Down to Next Cmd(TXP)	2 iCLK	2 iCLK	
Min CKE High/Low(TCKE)	4 iCLK	4 iCLK	
Self Refresh to Next Cmd(TXSR)	27 iCLK	27 iCLK	
SRR to SRR Read(TSRR)	3 iCLK	3 iCLK	
SRR Read to Next Cmd(TSRC)	2 iCLK	2 iCLK	
Write Recovery(TwR)	2 iCLK	2 iCLK	

Row Size

This option indicates the Row Address size used in the selected memory configuration. It corresponds to the 1 Gb LPDDR memory used for this evaluation package.

Column Size

This option indicates the Column Address size used in the selected memory configuration. It corresponds to the 1 Gb LPDDR Memory used for this evaluation package.

I/O Auto Training

This is an on/off option and if enabled, the controller will train the I/Os automatically. Refer to [“I/O Training” on page 4](#) for more information.

Periodic I/O Auto Retraining

This is an on/off option and if enabled, the controller will auto retrain the I/Os at the end of every refresh burst automatically.

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.

CAS Latency

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

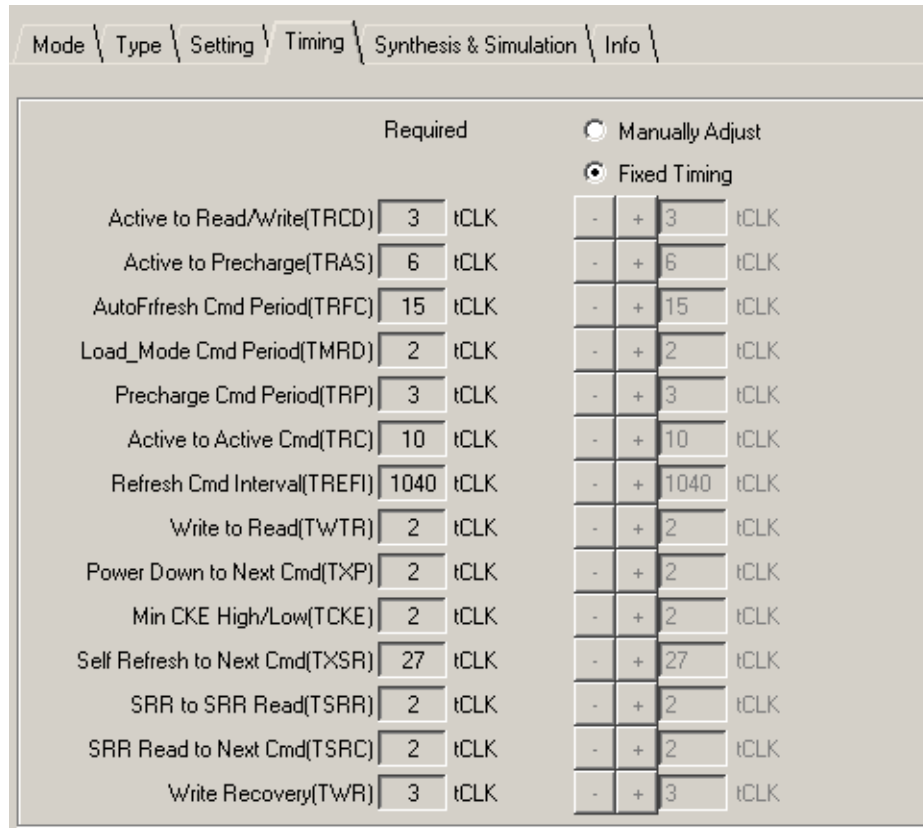
Burst Type

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.

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



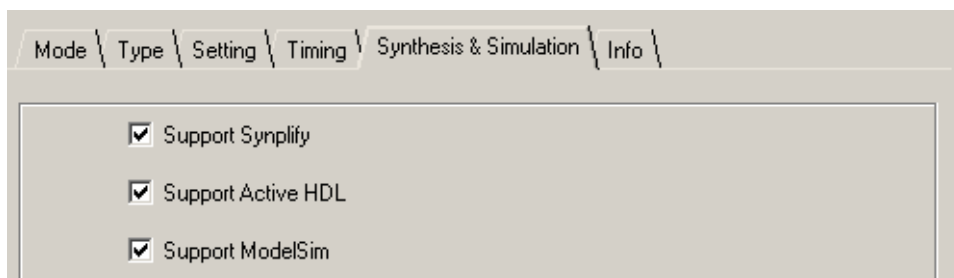
Parameter	Required	Manually Adjust	Fixed Timing
Active to Read/Write(TRCD)	3 tCLK	- + 3 tCLK	- + 3 tCLK
Active to Precharge(TRAS)	6 tCLK	- + 6 tCLK	- + 6 tCLK
AutoRefresh Cmd Period(TRFC)	15 tCLK	- + 15 tCLK	- + 15 tCLK
Load_Mode Cmd Period(TMRD)	2 tCLK	- + 2 tCLK	- + 2 tCLK
Precharge Cmd Period(TRP)	3 tCLK	- + 3 tCLK	- + 3 tCLK
Active to Active Cmd(TRC)	10 tCLK	- + 10 tCLK	- + 10 tCLK
Refresh Cmd Interval(TREFI)	1040 tCLK	- + 1040 tCLK	- + 1040 tCLK
Write to Read(TWTR)	2 tCLK	- + 2 tCLK	- + 2 tCLK
Power Down to Next Cmd(TXP)	2 tCLK	- + 2 tCLK	- + 2 tCLK
Min CKE High/Low(TCKE)	2 tCLK	- + 2 tCLK	- + 2 tCLK
Self Refresh to Next Cmd(TXSR)	27 tCLK	- + 27 tCLK	- + 27 tCLK
SRR to SRR Read(TSRR)	2 tCLK	- + 2 tCLK	- + 2 tCLK
SRR Read to Next Cmd(TSRC)	2 tCLK	- + 2 tCLK	- + 2 tCLK
Write Recovery(TWR)	3 tCLK	- + 3 tCLK	- + 3 tCLK

The default memory timing parameters displayed in this tab are the default values of the Micron Technology LPDDR 1Gb module. Users can adjust these parameters by selecting the Manual Adjust checkbox. It is important that the values in this Memory Device Timing tab are adjusted to the timing parameters of the on-board memory device that the user plans to use in their application.

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 Active HDL
- ☒ Support ModelSim

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 ModelSim

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

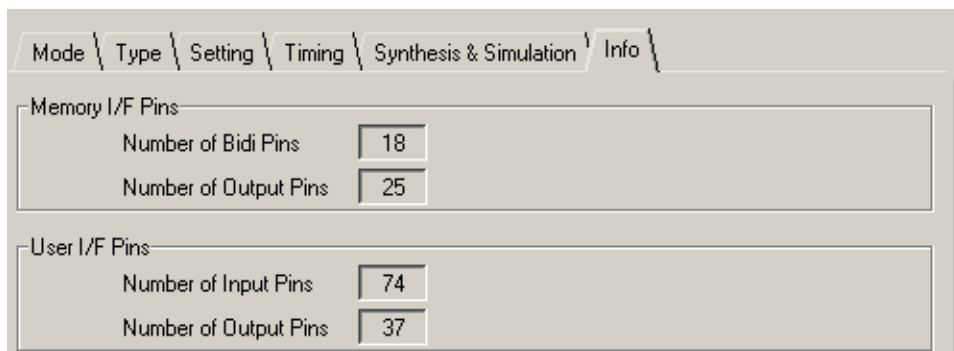
Support Aldec

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

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



Mode	Type	Setting	Timing	Synthesis & Simulation	Info
Memory I/F Pins					
		Number of Bidi Pins	18		
		Number of Output Pins	25		
User I/F Pins					
		Number of Input Pins	74		
		Number of Output Pins	37		

Memory Interface Pins

This section displays the following information:

Number of Bi-directional 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.

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 26](#) 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 time-out limitation.

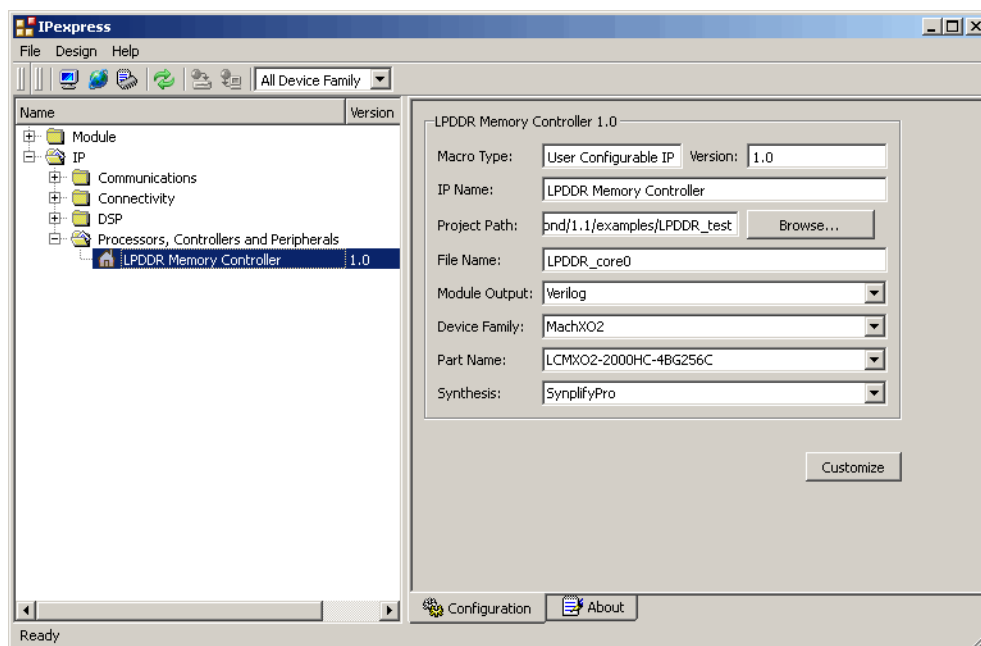
Getting Started

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.

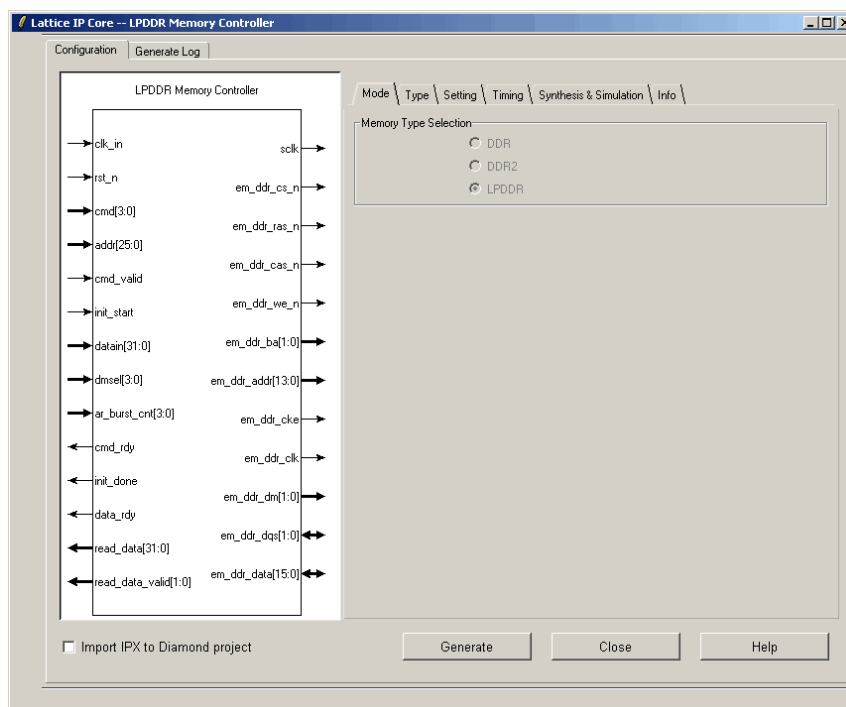
Figure 4-1. IPexpress Dialog Box



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

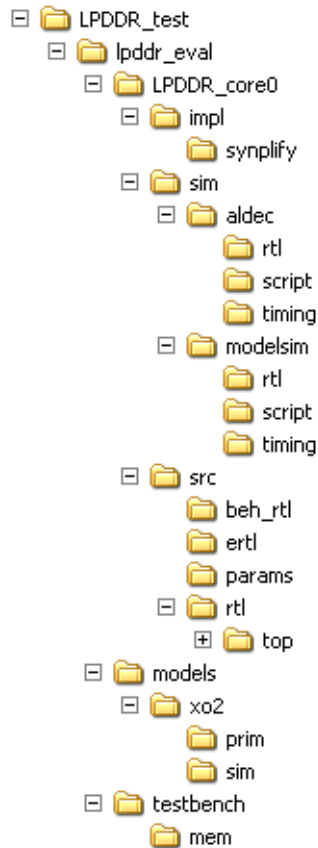
Figure 4-2. LPDDR SDRAM IP Configuration GUI



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 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

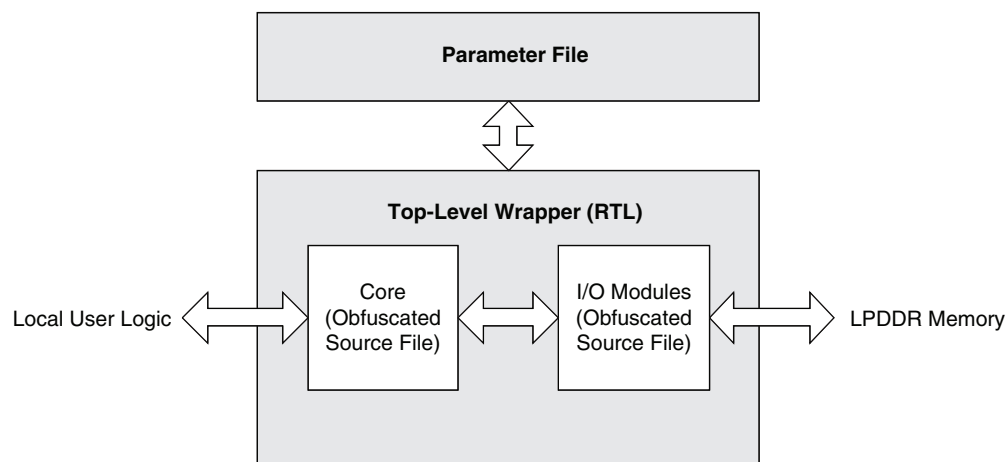
File	Description
<username>.v	This file provides the LPDDR SDRAM core for simulation.
<username>_beh.v	This file provides a behavioral simulation model for the LPDDR SDRAM core.
<username>_bb.v	This file provides the synthesis black box for the user's synthesis.
<username>.ngo	The ngo files provide the 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 for simulation and encrypted netlist memory controller core for synthesis
- Obfuscated I/O module block for simulation and encrypted netlist I/O module block 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


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

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

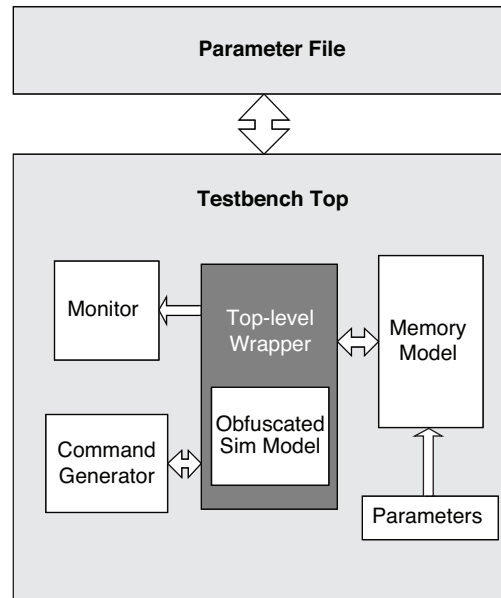
Obfuscated Module for the I/O Modules

The obfuscated I/O module block provides device-dependent LPDDR I/O functions. This block consist of one I/O module top file and several sub-modules that handle the LPDDR data (DQ), data mask (DM) and data strobe (DQS) signals. An equivalent netlist of the I/O modules 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 models for the core and I/O modules are instantiated in the top-level wrapper. These 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\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.

When compiling the LPDDR SDRAM IP core the following files must be compiled with the model.

- `pci_exp_params.v`
- `pci_exp_ddefines.v`

These files provide “define constants” that are necessary for the simulation model.

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 one of the ModelSim “do” scripts shown, depending on which version of ModelSim is used (ModelSim SE or the Lattice OEM version).

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 25](#) and then synthesizing the entire design with Synplify RTL Synthesis.

The top-level file `<username>_eval_top.v` provided in `\<project_dir>\lpddr_eval\<username>\src\top`

supports the ability to implement the LPDDR SDRAMlpddr_eval core in isolation. Push-button implementation of this top-level design with Synplify RTL Synthesis is supported via the project files <username>_eval.ldf located in the \<project_dir>\lpddr_eval\<username>\impl\synplify directory.

To use this project file:

1. Choose **File > Open > Project**.
2. Browse to \<project_dir>\lpddr_eval\<username>\impl\synplify in the Open Project dialog box.
3. Select and open <username>.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 **Target** 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.

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. 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.
- **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.

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 and 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

- [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.

Resource Utilization

This appendix gives resource utilization information for Lattice CPLDs 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/software.

MachXO2 Devices

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

Table A-1. Performance and Resource Utilization¹

IP Core	Parameter Settings ²	SLICES	LUTs	Registers	I/O	PAR Start Point	f _{MAX} (MHz)
LPDDR SDRAM Controller Configuration 1	Table 3-1 on page 14 parameter defaults	814	1530	911	149	1	136.4
LPDDR SDRAM Controller Configuration 2		705	1314	804	149	1	136.6
LPDDR SDRAM Controller Configuration 3		729	1409	969	143	1	142.9
LPDDR SDRAM Controller Configuration 4		729	1409	969	143	1	142.9

1. Performance and utilization data are generated using a LCMXO2-7000HE-6BG256C device in Lattice Diamond 1.3 design software. Performance may vary when using this IP core in a different density, speed or grade within the MachXO2 family.

2. SDRAM data path width of 16 bits.

Ordering Part Number

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