



# **LPDDR2 SDRAM Controller Lite IP Core - Lattice Radiant Software**

IP Version: v1.4.0

## **User Guide**

FPGA-IPUG-02089-1.6

December 2025

## Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS, with all faults, and all associated risk is the responsibility entirely of the Buyer. The information provided herein is for informational purposes only and may contain technical inaccuracies or omissions, and may be otherwise rendered inaccurate for many reasons, and Lattice assumes no obligation to update or otherwise correct or revise this information. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. LATTICE PRODUCTS AND SERVICES ARE NOT DESIGNED, MANUFACTURED, OR TESTED FOR USE IN LIFE OR SAFETY CRITICAL SYSTEMS, HAZARDOUS ENVIRONMENTS, OR ANY OTHER ENVIRONMENTS REQUIRING FAIL-SAFE PERFORMANCE, INCLUDING ANY APPLICATION IN WHICH THE FAILURE OF THE PRODUCT OR SERVICE COULD LEAD TO DEATH, PERSONAL INJURY, SEVERE PROPERTY DAMAGE OR ENVIRONMENTAL HARM (COLLECTIVELY, "HIGH-RISK USES"). FURTHER, BUYER MUST TAKE PRUDENT STEPS TO PROTECT AGAINST PRODUCT AND SERVICE FAILURES, INCLUDING PROVIDING APPROPRIATE REDUNDANCIES, FAIL-SAFE FEATURES, AND/OR SHUT-DOWN MECHANISMS. LATTICE EXPRESSLY DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY OF FITNESS OF THE PRODUCTS OR SERVICES FOR HIGH-RISK USES. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

# Contents

|  |    |
|--|----|
| Contents .....   | 3  |
| Abbreviations in This Document.....  | 5  |
| 1. Introduction .....  | 6  |
| 1.1. Quick Facts .....   | 6  |
| 1.2. Features .....  | 6  |
| 1.3. Conventions .....   | 7  |
| 1.3.1. Nomenclature.....   | 7  |
| 1.3.2. Signal Names .....  | 7  |
| 1.3.3. Attribute .....   | 7  |
| 2. Functional Description.....   | 8  |
| 2.1. Overview .....  | 8  |
| 2.2. Signal Description.....   | 9  |
| 2.3. Attributes Summary .....  | 11 |
| 2.4. Submodules Description .....  | 15 |
| 2.4.1. LPDDR2 Memory Controller Module.....                                      | 15 |
| 2.4.2. LPDDR2 PHY Module .....   | 15 |
| 2.4.3. Clock Synchronization Module .....  | 16 |
| 2.5. Operations Details.....   | 17 |
| 2.5.1. Initialization Control .....  | 17 |
| 2.5.2. Command and Address .....   | 18 |
| 2.5.3. User Commands .....   | 19 |
| 2.5.4. WRITE .....   | 20 |
| 2.5.5. WRITEA.....   | 21 |
| 2.5.6. READ .....  | 21 |
| 2.5.7. READA.....  | 21 |
| 2.5.8. REFRESH Support.....  | 21 |
| 2.6. Local-to-Memory Address Mapping .....                                       | 22 |
| 2.7. Mode register Programming.....  | 23 |
| 3. Core Generation, Simulation, and Validation .....                             | 24 |
| 3.1. Licensing the IP.....   | 24 |
| 3.2. Generation and Synthesis .....  | 24 |
| 3.2.1. Required Post-Synthesis Constraints.....                                  | 27 |
| 3.3. Running Functional Simulation .....   | 27 |
| 4. Ordering Part Number .....  | 29 |
| 5. Known Issue.....  | 30 |
| 5.1. Timing Constraints Not Recognized in Lattice Radiant 2023.1 and Later ..... | 30 |
| 5.1.1. Root Cause .....  | 30 |
| 5.1.2. Workaround .....  | 30 |
| Appendix A. Resource Utilization .....   | 31 |
| References .....   | 32 |
| Technical Support Assistance .....   | 33 |
| Revision History .....   | 34 |

## Figures

|  |    |
|--|----|
| Figure 2.1. LPDDR2 SDRAM Controller Lite IP Core Functional Diagram .....        | 8  |
| Figure 2.2. Timing of Memory Initialization Control.....                         | 17 |
| Figure 2.3. Timing Diagram for Read Retraining.....                              | 18 |
| Figure 2.4. Timing of Command and Address.....                                   | 19 |
| Figure 2.5. Timing One-Clock vs. Two-Clock Write Data Delay.....                 | 20 |
| Figure 2.6. User-Side Read Operation .....                                       | 21 |
| Figure 2.7. Local-to-Memory Address Mapping for Memory Access .....              | 22 |
| Figure 2.8. Mapped Address for the Example .....                                 | 22 |
| Figure 2.9. User-to-Memory Address Mapping for MR Programming .....              | 23 |
| Figure 3.1. Module/IP Block Wizard .....   | 24 |
| Figure 3.2. Module/IP Block Wizard of LPDDR2 SDRAM Controller Lite IP Core ..... | 25 |
| Figure 3.3. Check Generating Result.....   | 26 |
| Figure 3.4. Simulation Wizard.....   | 27 |
| Figure 3.5. Adding and Reordering Source .....                                   | 28 |
| Figure 3.6. Simulation Waveform Hardware Evaluation .....                        | 28 |

## Tables

|  |    |
|--|----|
| Table 1.1. Quick Facts .....   | 6  |
| Table 2.1. LPDDR2 SDRAM Controller Lite IP Core Signal Description ..... | 9  |
| Table 2.2. Attributes Table .....  | 11 |
| Table 2.3. Attributes Descriptions .....                                 | 13 |
| Table 2.4. Native Interface Functional Groups .....                      | 17 |
| Table 2.5. Defined User Commands .....                                   | 19 |
| Table 2.6. Address Mapping Example.....                                  | 22 |
| Table 2.7. Mode Register Selection Using Address Bits.....               | 23 |
| Table 2.8. Initialization Default Values for Mode Register Setting ..... | 23 |
| Table 3.1. Generated File List .....                                     | 26 |
| Table A.1. Resource Utilization.....                                     | 31 |

## Abbreviations in This Document

A list of abbreviations used in this document.

| Abbreviation | Definition                                |
|--------------|---|
| CAL          | Command Application Logic                 |
| CDL          | Command Decode Logic                      |
| CSM          | Clock Synchronization Module              |
| DM           | Data Mask                                 |
| DPL          | Data Path Logic                           |
| DQS          | Data Strobe                               |
| FPGA         | Field Programmable Gate Array             |
| JEDEC        | Joint Electron Device Engineering Council |
| LPDDR2       | Low Power Double Data Rate Type 2         |
| LSE          | Lattice Synthesis Engine                  |
| LUT          | Look-Up Table                             |
| PASR         | Partial Array Self-Refresh                |
| PLL          | Phase-Locked Loop                         |
| RTL          | Register Transfer Level                   |
| SDRAM        | Synchronous Dynamic Random Access Memory  |

# 1. Introduction

The Lattice Low Power Double Data Rate Type 2 (LPDDR2) Synchronous Dynamic Random-Access Memory (SDRAM) Controller Lite IP Core is a general-purpose memory controller that interfaces with industry standard LPDDR2 memory devices compliant with JESD209-2B, LPDDR2 SDRAM Standard, and provides a generic command interface to user applications. LPDDR2 SDRAM is the next-generation Low Power SDRAM memory technology which offers a higher data rate, higher density, greater bandwidth and power efficiency. This core reduces the effort required to integrate the LPDDR2 memory controller with the user application design and minimizes the need to directly deal with the LPDDR2 memory interface.

## 1.1. Quick Facts

[Table 1.1](#) presents a summary of the LPDDR2 SDRAM Controller Lite IP Core.

**Table 1.1. Quick Facts**

|                             |                          |  |
|-----------------------------|--------------------------|--|
| <b>IP Requirements</b>      | Supported Devices        | CrossLink™-NX (LIFCL-40, LIFCL-17), Certus™-NX (LFD2NX-40, LFD2NX-17), CertusPro™-NX (LFCPNX-100). |
| <b>Resource Utilization</b> | Supported User Interface | Native   |
|                             | Resources                | See <a href="#">Table A.1</a>  |
| <b>Design Tool Support</b>  | Lattice Implementation   | IP Core v1.4.0 – Lattice Radiant software 3.0  |
|                             | Synthesis                | Lattice Synthesis Engine (LSE)<br>Synopsys® Synplify Pro® for Lattice                              |
|                             | Simulation               | For a list of supported simulators, see the Lattice Radiant software user guide.                   |

## 1.2. Features

The key features of LPDDR2 SDRAM Controller Lite IP Core include:

- Interfaces to industry standard LPDDR2 SDRAM components and modules compliant with JESD209-2B, LPDDR2 SDRAM Standard
- Interfaces to LPDDR2 SDRAM at speeds up to 400 MHz/800 Mbps
- Supports memory data path width of 16
- Supports x16 device configuration
- Supports single rank of an LPDDR2 device (one chip select)
- Supports burst lengths of eight (fixed)
- Programmable read and write latency set
- Supports automatic LPDDR2 SDRAM initialization and refresh
- Automatic read training for each DQS
- Supports Deep Power Down mode and Power down mode
- Supports Self-Refresh operations on all Banks
- Partial Array Self-Refresh (PASR) operations on Bank and Segment
- Automatic programmable interval refresh or user-initiated refresh
- Supports Burst or Distributed auto refresh
- Supports all LIFCL devices

**Note:** This Lite version IP Core only supports features and commands that LPDDR2 have in common with LPDDR3. Burst lengths of 8 is supported, but burst lengths of 4 and 16 are not supported. Also, flash memory commands (LPDDR2-N) are not supported.

## 1.3. Conventions

### 1.3.1. Nomenclature

The nomenclature used in this document is based on Verilog HDL.

### 1.3.2. Signal Names

Signal Names that end with:

- `_n` are active low
- `_i` are input signals
- `_o` are output signals
- `_io` are bi-directional input/output signals

### 1.3.3. Attribute

The names of attributes in this document are formatted in title case and italicized (*Attribute Name*).

## 2. Functional Description

### 2.1. Overview

The LPDDR2 SDRAM Controller Lite IP Core consists of three submodules: Memory Controller (MC) module, Physical Interface (PHY) module and Clock Synchronization Module (CSM). It takes the requests on its Local Interface and converts into LPDDR2 SDRAM Memory Interface. It formats request and organizes flow control on the data buses of to/from the LPDDR2 SDRAM Memory.

The [Submodules Description](#) section briefly describes the operation of each of these submodules. [Figure 2.1](#) provides a high-level block diagram illustrating the main functional blocks used to implement the LPDDR2 SDRAM Controller Lite IP Core functions.

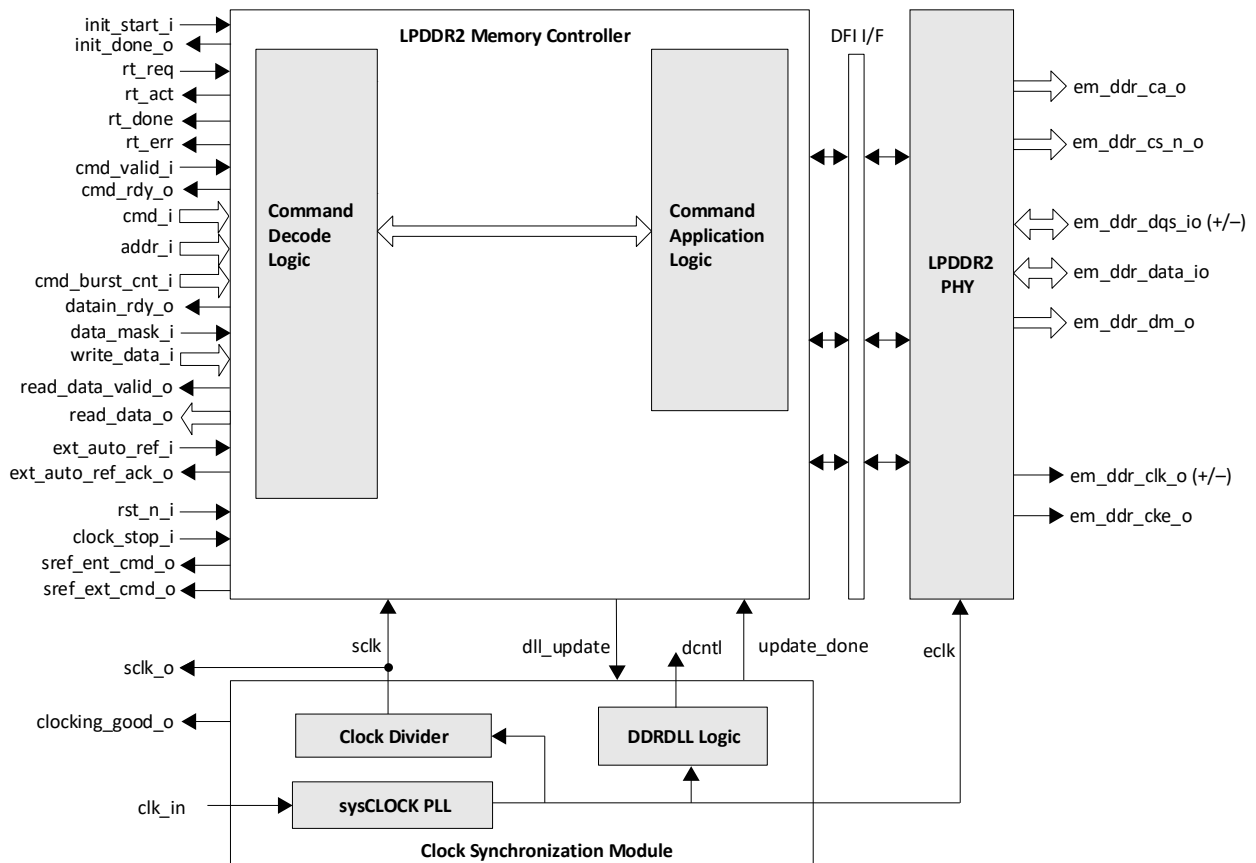


Figure 2.1. LPDDR2 SDRAM Controller Lite IP Core Functional Diagram

## 2.2. Signal Description

Table 2.1 lists the input and output signals for LPDDR2 SDRAM Controller Lite IP Core.

**Table 2.1. LPDDR2 SDRAM Controller Lite IP Core Signal Description**

| Port Name   | I/O    | Width        | Description   |
|---|--------|--------------|---|
| <b>Clock and Reset</b>                            |        |              |   |
| clk_i   | In     | 1            | Reference clock for PLL   |
| clocking_good_o                                   | Out    | 1            | Signal from LPDDR2 SDRAM PHY indicating stable clock condition  |
| sclk_o  | Out    | 1            | System clock used by LPDDR2 Memory Controller Module. You may use this clock for LPDDR2 controller interface logic.   |
| clock_stop_i                                      | In     | 1            | Stops the LPDDR2 SDRAM memory clock<br>The memory clock stays Low while this signal is asserted High.   |
| rst_n_i   | In     | 1            | Active low asynchronous reset signal for the entire IP Core   |
| <b>LPDDR2 SDRAM Memory Interface*<sup>1</sup></b> |        |              |   |
| em_ddr_clk_o                                      | Out    | CLKO_WIDTH   | Differential pair memory clock generated by the controller.<br>This signal is internally single ended and is converted to differential pair at the FPGA pin.  |
| em_ddr_cke_o                                      | Out    | CKE_WIDTH    | Memory clock enable generated by the controller   |
| em_ddr_ca_o                                       | Out    | CA_WIDTH     | Memory Command and Address (CA) signal – Multiplexed row address, column address and command to the memory.   |
| em_ddr_data_io                                    | In/Out | DATA_WIDTH   | Memory bi-directional data signal   |
| em_ddr_dm_o                                       | Out    | DATA_WIDTH/8 | LPDDR2 memory write data mask is used to mask the byte lanes for byte-level write.  |
| em_ddr_dqs_io                                     | In/Out | DATA_WIDTH/8 | Memory bi-directional, differential pair data strobe  |
| em_ddr_cs_n_o                                     | Out    | CS_WIDTH     | Memory chip select  |
| <b>Native Interface</b>                           |        |              |   |
| init_start_i                                      | In     | 1            | Initialization start request<br>Assert this signal to start memory initialization either right after the power-on reset or before sending the first user command to the memory controller.                                      |
| init_done_o                                       | Out    | 1            | Initialization done output. Asserts for one clock period after the core completes memory initialization. When sampled high, the input signal init_start must be immediately de-asserted at the same edge of the sampling clock. |
| rt_req_i  | In     | 1            | Read pulse training request from the user   |
| rt_act_o  | Out    | 1            | Signal to indicate that the read pulse training is active   |
| rt_done_o   | Out    | 1            | Signal to indicate that the read pulse training has been completed  |
| rt_err_o  | Out    | 1            | Signal to indicate a failure during the read pulse training   |
| cmd_valid_i                                       | In     | 1            | Command and address valid input<br>When asserted, the addr, cmd and cmd_burst_cnt inputs are considered valid.  |
| cmd_rdy_o   | Out    | 1            | Command ready output<br>When asserted, indicates that the core is ready to accept the next command and the corresponding address.   |
| cmd_i   | In     | 4            | User command input to the memory controller   |
| addr_i <sup>2</sup>                               | In     | ADDR_WIDTH   | User read or write address input to the memory controller   |
| cmd_burst_cnt_i                                   | In     | 1            | Command burst count input<br>Indicates the number of times a given read or write command is to be repeated by the controller automatically.   |
| datain_rdy_o                                      | Out    | 1            | Data ready output<br>When asserted, indicates the core is ready to receive the write data.  |
| data_mask_i <sup>3</sup>                          | In     | DSIZE/8      | Data mask input for write data<br>Each bit masks a corresponding byte of local write data.  |

| Port Name                 | I/O | Width | Description   |
|---------------------------|-----|-------|---|
| write_data_i <sup>3</sup> | In  | DSIZE | Write data input from user logic to the memory controller<br>The user side writes data width is four times the memory data signal.  |
| read_data_valid_o         | Out | 1     | Read data valid output<br>When asserted, indicates the data on the read_data_o signal is valid.   |
| read_data_o <sup>*3</sup> | Out | DSIZE | Read data output from memory controller to the user logic   |
| ext_auto_ref_i            | In  | 1     | Refresh request from user<br>This signal is available only when the <i>External Auto Refresh Port</i> attribute is Checked.   |
| ext_auto_ref_ack_o        | Out | 1     | Completion of memory refresh in response to ext_auto_ref_i signal assertion. This signal is available only when the <i>External Auto Refresh Port</i> attribute is Checked. |
| sref_ent_cmd_o            | Out | 1     | Asserts when the IP Core enters Self refresh operation. This occurs when a Self-Refresh Entry command is received.  |
| sref_ext_cmd_o            | Out | 1     | Asserts when the IP Core exits Self refresh operation. This occurs when a Self-Refresh Exit command is received.  |

**Notes:**

1. The bit width of LPDDR2 SDRAM Memory Interface signals are set by the attributes, refer to [Table 2.3](#) for the description of these attributes.
2. The bit width of addr\_i is set by ADDR\_WIDTH which is defined in [Local-to-Memory Address Mapping](#) section.
3. The bit width of write\_data\_i, data\_mask\_i and read\_data\_o are set by DSIZE which is 4 × DATA\_WIDTH.

## 2.3. Attributes Summary

The configurable attributes of the LPDDR2 SDRAM Controller Lite IP Core are shown in [Table 2.2](#) and are described in [Table 2.3](#). The attributes can be configured through the IP Catalog's Module/IP wizard of the Lattice Radiant software. The attributes are arranged into tabs and related attributes are collected into groups. The three tabs are as follows:

- **General Tab**  
The General tab contains the attributes for configuring the target memory device and the IP Core features. These attributes are static; they can only be set in the Module/IP Block wizard. The LPDDR2 SDRAM Controller Lite IP Core must be regenerated to change the features set by these attributes.
- **Memory Device Setting Tab**  
The Memory Device Setting Tab contains the attributes for configuring the target memory device/module. The attributes under Mode Register Initial Setting Group are dynamic, which means, reset values are set from Module/IP Block Wizard and are dynamically changeable using MRW user commands. Refer to JESD209-2B, LPDDR2 SDRAM Standard, for allowed values.
- **Memory Device Timing Tab**  
The attribute default displayed in this tab are the default values of the Micron LPDDR2 4Gb -25 memory module. These attributes can be modified by checking the *Manual Adjust* attribute. It is important that the attribute values in this tab are adjusted to the timing parameters of the memory device for the target application. The LPDDR2 SDRAM Controller Lite IP Core also uses these timing parameters when generating memory commands.

**Table 2.2. Attributes Table**

| Attribute                         | Selectable Values   | Default                         | Dependency on Other Attributes   |
|-----------------------------------|---|---------------------------------|--|
| <b>General Tab</b>                |   |                                 |  |
| <b>Device Information Group</b>   |   |                                 |  |
| Select Memory                     | Micron LPDDR2 1Gb-25 16x,<br>Micron LPDDR2 4Gb-25 32x,<br>Custom  | Micron<br>LPDDR2 1Gb-<br>25 16x | The <i>Micron LPDDR2 4Gb-25 32x</i> is not available when the target FPGA device cannot support 32-bit data. |
| <b>Clock Settings Group</b>       |   |                                 |  |
| Enable PLL                        | Checked   | Checked                         | Display only   |
| PLL Reference Clock from Pin      | Checked, Unchecked  | Checked                         | —  |
| I/O Standard for Reference Clock  | SLVS, HSUL12, HSUL12D,<br>LVTT133, LVCMOS33,<br>LVCMOS25, LVCMOS18,<br>LVCMOS15, LVCMOS12,<br>LVCMOS10, LVCMOS10R | SLVS                            | Enabled when <i>PLL Reference Clock from Pin</i> is checked  |
| RefClock (MHz)                    | 25.0, 33.3, 50, 66.6, 75.0,<br>100.0, 125.0, 150.0, 200.0   | 50                              | —  |
| MemClock (MHz)                    | 50.0, 66.6, 150.0, 166.6,<br>200.0, 250.0, 266.6, 300.0,<br>333.3, 350.0, 366.6, 400.0                            | 150                             | —  |
| MemClock Tolerance (%)            | 0.0, 0.1, 0.2, 0.5, 1.0, 2.0, 5.0,<br>10.0  | 0.0                             | —  |
| MemClock Actual Frequency (MHz)   | Calculated  | N/A                             | Display Information only   |
| <b>Memory Configuration Group</b> |   |                                 |  |
| Memory Type                       | On-board Memory   | On-board<br>Memory              | —  |

| Attribute                                  | Selectable Values                  | Default   | Dependency on Other Attributes   |
|--|------------------------------------|-----------|--|
| Memory Data Bus Size                       | 16, 32                             | 16        | Depends on <i>Select Memory</i> <ul style="list-style-type: none"> <li>Micron LPDDR2 1Gb-25 16x: 16</li> <li>Micron LPDDR2 4Gb-25 32x: 32</li> <li>Custom: 16, 32</li> </ul> The 32-bit data configuration is not available if the target FPGA device cannot support it. |
| Configuration                              | x16, x32                           | x16       | Depends on <i>Memory Data Bus Size</i> <ul style="list-style-type: none"> <li>16: x16</li> <li>32: x32</li> </ul>  |
| Rank Size                                  | Single                             | Single    | Display Information only   |
| Clock Width                                | 1                                  | 1         | Display Information only   |
| CKE Width                                  | 1                                  | 1         | Display Information only   |
| <b>Local Interface</b>                     |                                    |           |  |
| Local Bus Type                             | Native                             | Native    | Display Information only   |
| <b>Additional Configuration Group</b>      |                                    |           |  |
| Data ready to Write Data delay             | 1, 2, 3                            | 1         | —  |
| <b>Memory Device Setting Tab</b>           |                                    |           |  |
| <b>Address Group</b>                       |                                    |           |  |
| Bank Address Width                         | 2,3                                | 3         | Editable when <i>Select Memory</i> = Custom  |
| Row Size                                   | 12,13,14,15                        | 13        | Default value is set by <i>Select Memory</i> : <ul style="list-style-type: none"> <li>Micron LPDDR2 1Gb-25 16x: 13</li> <li>Micron LPDDR2 4Gb-25 32x: 14</li> <li>Custom: 12</li> </ul> Editable when <i>Select Memory</i> = Custom                                      |
| Column Size                                | 8, 9, 10, 11, 12                   | 10        | Editable when <i>Select Memory</i> = Custom  |
| <b>Auto Refresh Control Group</b>          |                                    |           |  |
| Refresh Type                               | Burst, Distributed                 | Burst     |  |
| Refresh Bank                               | All-Bank                           | All-Bank  | Display Information only   |
| Auto Refresh Burst Count                   | 8, 32, 512, 1024, 2048, 4096, 8192 | 32        | —  |
| External Auto Refresh Port                 | Checked/Unchecked                  | Unchecked | —  |
| <b>Mode Register Initial Setting Group</b> |                                    |           |  |
| Burst Length                               | Fixed 8                            | Fixed 8   | —  |
| Write Recovery                             | 3, 4, 5, 6, 7, 8                   | 6         | —  |
| Read Latency                               | 3, 4, 5, 6, 7, 8                   | 6         | —  |
| Write Latency                              | 1, 2, 3, 4                         | 3         | —  |
| Drive Strength (Ohm)                       | [34.3, 40, 48, 60, 80, 12] PD/PU   | 40 PD/PU  | —  |
| <b>Memory Device Timing Tab</b>            |                                    |           |  |
| <b>Command and Address Timing Group</b>    |                                    |           |  |
| Manually Adjust                            | Checked/Unchecked                  | Unchecked | —  |
| TRTP(tCLK)                                 | 2–65536                            | 3         | Enabled when <i>Manually Adjust</i> is Checked   |
| TWTR(tCLK)                                 | 2–65536                            | 4         | Enabled when <i>Manually Adjust</i> is Checked   |
| TMRW(tCLK)                                 | 5–65536                            | 10        | Enabled when <i>Manually Adjust</i> is Checked   |
| TMRR(tCLK)                                 | 4–65536                            | 4         | Enabled when <i>Manually Adjust</i> is Checked   |
| TRCD(tCLK)                                 | 3–65536                            | 10        | Enabled when <i>Manually Adjust</i> is Checked   |
| TRP_AB(tCLK)                               | 3–65536                            | 10        | Enabled when <i>Manually Adjust</i> is Checked   |
| TRP_PB(tCLK)                               | 3–65536                            | 10        | Enabled when <i>Manually Adjust</i> is Checked   |
| TRC(tCLK)                                  | 6–65536                            | 27        | Enabled when <i>Manually Adjust</i> is Checked   |

| Attribute   | Selectable Values | Default | Dependency on Other Attributes                 |
|---|-------------------|---------|--|
| TRAS(tCLK)  | 3–65536           | 17      | Enabled when <i>Manually Adjust</i> is Checked |
| TRRD(tCLK)  | 2–65536           | 4       | Enabled when <i>Manually Adjust</i> is Checked |
| <b>Refresh, Reset and Power Down Timing Group</b> |                   |         |  |
| TRFC_AB(tCLK)                                     | 13–65536          | 52      | Enabled when <i>Manually Adjust</i> is Checked |
| TRFC_PB(tCLK)                                     | 6–65536           | 24      | Enabled when <i>Manually Adjust</i> is Checked |
| TREFI_AB(tCLK)                                    | 8–4160            | 3120    | Enabled when <i>Manually Adjust</i> is Checked |
| TREFI_PB(tCLK)                                    | 8–4160            | 390     | Enabled when <i>Manually Adjust</i> is Checked |
| TCKE(tCLK)  | 3–65536           | 3       | Enabled when <i>Manually Adjust</i> is Checked |
| TCKESR(tCLK)                                      | 3–65536           | 6       | Enabled when <i>Manually Adjust</i> is Checked |
| TXP(tCLK)   | 2–4160            | 4       | Enabled when <i>Manually Adjust</i> is Checked |
| TXSR(tCLK)  | 14                | 56      | Enabled when <i>Manually Adjust</i> is Checked |

**Table 2.3. Attributes Descriptions**

| Attribute                         | Description  |
|-----------------------------------|--|
| <b>General Tab</b>                |  |
| <b>Device Information Group</b>   |  |
| Select Memory                     | Some attribute default values are dependent on this attribute. Micron LPDDR2 1Gb-25 16x is provided as the default LPDDR2 SDRAM device, the timing parameters of this memory device are listed in the Memory Device Timing tab as default values.  |
| <b>Clock Settings</b>             |  |
| Enable PLL                        | Shows that PLL is instantiated inside the LPDDR2 SDRAM Controller Lite IP Core.  |
| PLL Reference Clock from Pin      | When checked the reference clock of internal PLL is connected to an FPGA pin. This option allows you to specify the <i>I/O Standard for Reference Clock</i> . If Unchecked, PLL expects that the reference clock is connected to the FPGA fabric.  |
| I/O Standard for Reference Clock  | Specifies the I/O standard of the FPGA pin that is used for input reference clock.   |
| RefClock (MHz)                    | Specifies the reference input clock to PLL which generates the system clock (sclk_o) and memory clock (em_ddr_clk_o).  |
| MemClock (MHz)                    | Specifies the frequency of the memory clock to memory device. The default value is linked to the speed grade of Lattice device selected. This is the PLL output frequency which depends on the corresponding value of RefClock. For example, for MemClock value of 333 MHz the PLL RefClock should be set to 111 MHz |
| MemClock Tolerance (%)            | Enables the acceptance of selected percent difference between <i>MemClock</i> attribute and the calculated <i>MemClock Actual Frequency</i> .  |
| MemClock Actual Frequency (MHz)   | Shows the calculated actual DDR memory clock frequency.  |
| <b>Memory Configuration Group</b> |  |
| Memory Type                       | This attribute is for information only. Only On-board Memory type is supported.  |
| Memory Data Bus Size (DATA_WIDTH) | Specifies the bit width of LPDDR2 SDRAM data signal (em_ddr_data_io). If the memory module has a wider data than required, only the required data width should be selected.  |
| Configuration                     | Selects the device configuration of the on-board memory. The memory controller supports device configurations x16, and x32.  |
| Rank Size (CS_WIDTH)              | Select the number of Chip selects (em_ddr_cs_n_o) required. Only single Rank is supported.   |
| Clock Width (CLKO_WIDTH)          | Specifies the number of clocks signals (em_ddr_clk_o) with which the IP Core drives the memory. Only 1 clock output. The clocks signals are converted to differential pair in the FPGA pins.<br>Note that the differential pair signals are not shown in simulation.   |
| CKE Width (CKE_WIDTH)             | Specifies the number of Clock Enable (CKE) signals (em_ddr_cke_o) with which the IP Core drives the memory. Only one cke output is supported.  |
| <b>Local Interface</b>            |  |
| Local Bus Type                    | Specifies the user interface in FPGA fabric side. Only Native Interface is currently supported.  |

| Attribute                                  | Description   |
|--|---|
| <b>Additional Configuration Group</b>      |   |
| Data ready to Write Data delay             | This option is for information only. User logic is allowed to send the write data to the controller after a one-clock cycle delay with respect to datain_rdy_o signal.  |
| <b>Memory Device Setting Tab</b>           |   |
| <b>Address Group</b>                       |   |
| Bank Address Width                         | Indicates the size of Bank Address.   |
| Row Size (ROW_WIDTH)                       | Indicates the default Row Address size used in the selected memory configuration.   |
| Column Size                                | Indicates the default Column Address size used in the selected memory configuration.  |
| <b>Auto Refresh Control Group</b>          |   |
| Refresh Type                               | This option allows you to select either Burst auto refresh or Distributed auto refresh.<br>Burst: IP Core performs a series of refresh cycles one after another until all the rows have been refreshed, after which normal memory accesses occur until the next refresh<br>Distributed: Refresh cycles are performed at regular intervals, interspersed with memory accesses.   |
| Refresh Bank                               | Indicates that refresh operation is performed for all banks. Per bank auto refresh is not supported.  |
| Auto Refresh Burst Count                   | Indicates the number of Auto Refresh commands that the LPDDR2 SDRAM Controller Lite IP Core is set to send in a single burst. Refer to <a href="#">REFRESH Support</a> for more details.  |
| External Auto Refresh Port                 | Specifies the generation of refresh commands to the memory.<br>If Unchecked: the controller automatically generates refresh commands to the memory at the interval defined by the <i>Auto Refresh Burst Count</i> and memory refresh timing requirement.<br>If Checked: The user logic is allowed to generate a Refresh request to the controller through the ext_auto_ref_i signal. Refer to <a href="#">REFRESH Support</a> for more details. |
| <b>Mode Register Initial Setting Group</b> |   |
| Burst Length                               | Sets the Burst length value in Mode Register 1 during initialization. Only Burst Length 8 is currently supported.   |
| READ Latency                               | Sets the READ Latency value in Mode Register 1 during initialization. This value remains until you write a different value to the Mode Register.  |
| Write Latency                              | This only displays the corresponding Write latency for the selected Read Latency. This value remains until you write a different Read Latency value to the Mode Register.   |
| Write Recovery                             | Sets the Write Recovery value in Mode Register 1 during initialization. The value is in terms of Memory clock. This value remains until you write a different value to the Mode Register.   |
| Drive Strength ( $\Omega$ )                | This option sets the Output Driver Strength Control value in Mode Register 3 during initialization. This value remains until you write a different value to the Mode Register. A list of Pull-down (PD) and Pull-up (PU) values are provided as defined by the LPDDR2's Mode Register 3.  |
| <b>Memory Device Timing Tab*</b>           |   |
| <b>Command and Address Timing Group</b>    |   |
| Manually Adjust                            | Checking this box allows you to manually set any of the memory timing parameters. If you need to change any of the default values, the Manual Adjust checkbox must be checked. This selection enables you to modify the memory timing parameters.   |
| TRTP(tCLK)                                 | Internal Read to Precharge Command Delay  |
| TWTR(tCLK)                                 | Internal Write to Read Command Delay  |
| TMRW(tCLK)                                 | MODE Register Write Command Period  |
| TMRR(tCLK)                                 | Mode Register Read Command Period   |
| TRCD(tCLK)                                 | RAS to CAS Delay  |
| TRP_AB(tCLK)                               | Row Precharge Time (all banks)  |
| TRP_PB(tCLK)                               | Row Precharge Time (single bank)  |
| TRC(tCLK)                                  | ACTIVE to ACTIVE Command Period   |
| TRAS(tCLK)                                 | Row Active Time   |
| TRRD(tCLK)                                 | Active Bank A to Active Bank B  |

| Attribute   | Description  |
|---|--|
| <b>Refresh, Reset and Power Down Timing Group</b> |  |
| TRFC_AB(tCLK)                                     | Refresh Cycle time   |
| TRFC_PB(tCLK)                                     | Per Bank Refresh Cycle Time  |
| TREFI_AB(tCLK)                                    | Average time between REFRESH commands (all banks)                              |
| TREFI_PB(tCLK)                                    | Average time between REFRESH commands (single bank)                            |
| TCKE(tCLK)  | CKE min. pulse width (high and low pulse width)                                |
| TCKESR(tCLK)                                      | CKE min. pulse width during Self-Refresh (low pulse width during Self-Refresh) |
| TXP(tCLK)   | Exit power down to next valid command delay                                    |
| TXSR(tCLK)  | Self-refresh exit to next valid command delay                                  |

**\*Note:** The Memory Device Timing parameters listed in this tab are standard parameters as defined in JESD209-2B, LPDDR2 SDRAM Standard. Refer to the memory device data sheet for detailed descriptions and allowed values of these parameters.

## 2.4. Submodules Description

### 2.4.1. LPDDR2 Memory Controller Module

The LPDDR2 Memory Controller module contains Command Decode Logic (CDL) block and Command Application Logic (CAL) block.

#### 2.4.1.1. Command Decode Logic

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. The intelligent bank management logic tracks the open/close status of every bank and stores the row address of every opened bank. The controller implements a command pipeline of depth 2 to improve throughput. With this capability, the next command in the queue is decoded while the current command is presented at the memory interface.

#### 2.4.1.2. Command Application Logic

The Command Application Logic (CAL) block accepts the decoded internal command sequence from the Command Decode Logic and translates each sequence into appropriate memory commands that meet the operational sequence and timing requirements of the memory device. The CDL and CAL blocks work in parallel to fill and empty the command queue respectively.

### 2.4.2. LPDDR2 PHY Module

The Physical Interface (PHY) module implements a set of soft logic in the FPGA fabric for initialization, read training and read/write paths as well as MDDR Controller IP for 1:2 clock gearing and LPDDR2 memory interface. MDDR Controller IP includes LIFCL FPGA Family device hardware primitives including I/O modules that directly interface with the LPDDR2 memory. These primitives implement all interface signals required for memory access. They convert the single data rate (SDR) data from you to double rate LPDDR2 data for the write operation and perform the DDR to SDR conversion in the read mode.

#### 2.4.2.1. Initialization

The Initialization block performs the LPDDR2 memory initialization sequence as defined by LPDDR2 JEDEC protocol. After power on or a normal reset of the LPDDR2 controller, the memory must be initialized before sending any command to the controller. It is your responsibility to assert the `init_start` input to the LPDDR2 controller to initiate the memory initialization sequence. The completion of initialization is indicated by the `init_done` field of the status register.

#### 2.4.2.2. Read Training

For every read operation, the LPDDR2 I/O primitives of the LIFCL device must be initialized at the appropriate time to identify the incoming DQS preamble. Upon a proper detection of the preamble, the primitive DQSBUF extracts a clean signal out of the incoming DQS signal from the memory and generates the `DATAVALID` output signal that indicates the correct timing window of the valid read data. The memory controller generates a positioning signal, `READ[2:0]`, to the

primitive DQSBUF that is used for the above-mentioned operation. In addition to the READ[2:0] input, another fine control input signal READCLKSEL[3:0] and an output signal, BURSTDET, of the DQSBUF block are provided to the controller to accomplish the READ signal positioning. Due to the DQS round trip delay that includes PCB routing and I/O pad delays, proper internal positioning of the READ signal with respect to the incoming preamble is crucial for successful read operations. The LIFCL DQSBUF block supports a dynamic READ signal positioning function called read training that enables the memory controller to position the internal READ signal within an appropriate timing window by progressively shifting the READ signal and monitoring the positioning result. During the read training, the memory controller generates the READ[2:0] pulse as a coarse positioning control. READCLKSEL[3:0] is also generated to provide a finer position control (1/4 T per step). The BURSTDET output of DQSBUF is used to monitor the result of the current position. The READ[2:0] signal is needed to be set high at least 5.5T before the read preamble starts. The internal READ is progressively shifted by READ[2:0] and READCLKSEL[3:0] once the read training routine starts. When the internal READ signal is properly positioned, the BURSTDET signal is asserted high to indicate that the preamble has been detected correctly. This guarantees that the generated DATAVALID signal is indicating the correct read valid time window. The READ[2:0] and READSELCLK[2:0] signals are generated in the system clock (SCLK) domain and required to stay asserted for the total burst length of the read operation.

The memory controller determines the proper position alignment when there is not a single failure on BURSTDET assertions during the multiple trials. If there is any failure, the memory controller shifts the READCLKSEL[3:0] and READ[2:0] signals accordingly to position the internal READ to a safer position and tries again until it detects no BURSTDET failure during the entire read training process. The memory controller stores the delay value of the successful position of the READCLKSEL[3:0] and READ[2:0] signal for each DQS group. It uses these delay values during a normal read operation to correctly detect the preamble first, followed by the generation of DATAVALID signal.

#### 2.4.2.3. PHY Control Block

The PHY Control Block includes the data path control and PHY timing control functions. The data path control function interfaces with the LPDDR2 I/O modules and is responsible for generating the read data and read data valid signals during the read operations. This block also implements all the logic that are needed to ensure that the LPDDR2 command/controls and data write/read to and from the memory are properly transferred to the local user interface in a deterministic and coherent manner.

#### 2.4.2.4. Data Path Logic

The Data Path Logic (DPL) block interfaces with the LPDDR2 I/O modules and is responsible for generating the read data and read data valid signals during 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.

### 2.4.3. Clock Synchronization Module

The Clock Synchronization Module (CSM) generates all the clock signals, such as system clock (sclk\_o) and edge clock (ECLK) for the IP core. The CSM logic ensures that all DDR I/O primitives are synchronized after a system reset. CSM also provides the logic for the DLL update operation. Without proper synchronization, the bit order on different elements might be off-sync with each other and the entire bus is scrambled. The clock synchronization ensures that all DDR components start from exactly the same edge clock cycle.

For 400 MHz LPDDR2 SDRAM device support, the MC module operates with a 200 MHz system clock (SCLK), the I/O logic works with a 400 MHz edge clock (ECLK). The combination of this operating clock ratio and the double data rate transfer leads to a user side data bus that is four times the width of the memory side data bus.

## 2.5. Operations Details

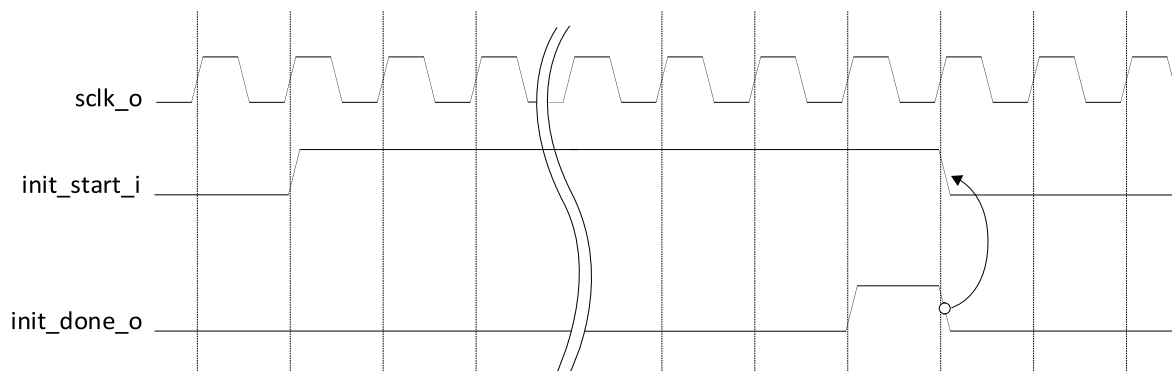
The Native Interface of the LPDDR2 SDRAM Controller Lite IP Core consists of five independent functional groups. Each functional group and its associated local interface signals as listed in [Table 2.4](#).

**Table 2.4. Native Interface Functional Groups**

| Functional Group       | Native Interface Signals   |
|------------------------|--|
| Initialization Control | init_start_i, init_done_o, rt_req_i, rt_act_o, rt_done_o, rt_err_o |
| Command and Address    | cmd_valid_i, cmd_rdy_o, cmd_i, addr_i, cmd_burst_cnt_i             |
| Data Write             | datain_rdy_o, data_mask_i, write_data_i                            |
| Data Read              | read_data_valid_o, read_data_o                                     |
| Auto Refresh           | ext_auto_ref_i, ext_auto_ref_ack_o                                 |

### 2.5.1. Initialization Control

LPDDR2 SDRAM devices must be initialized before they can be accessed by the memory controller. The memory controller starts the memory initialization sequence when external/user logic asserts the init\_start\_i signal. Once asserted, the init\_start signal needs to be held high until the initialization process is completed. The output signal init\_done\_o is asserted for one clock cycle indicating that the core has completed the initialization sequence and is now ready to access the SDRAM device. The external/user logic must negate the init\_start\_i signal as soon as init\_done\_o is sampled high at the rising edge of sclk\_o. 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 is reset. The memory controller ensures a minimum gap of 200  $\mu$ s between em\_ddr\_cke assertion and reset command to the memory. [Figure 2.2](#) shows the timing diagram of the initialization control signals.



**Figure 2.2. Timing of Memory Initialization Control**

The read training is also performed during the initialization process to find the best-read pulse position that detects the incoming read DQS preamble timing. Since LPDDR2 SDRAM devices do not use a DLL function, the clock to DQS driving time can vary significantly with the process, voltage, and temperature (PVT) variations. Due to this reason, periodic retraining of the read pulse position may be necessary for guaranteeing stable read transactions over the PVT variations during the normal operation. The LPDDR2 SDRAM Controller Lite IP core provides you with a function that can perform read retraining for PVT calibration during the normal operation using the following signals:

- rt\_req\_i: read retraining request
- rt\_act\_o: read training active
- rt\_done\_o: read retraining done

External/user logic need to assert the `rt_req_i` signal during the normal operation but only when the LPDDR2 interface is in an idle state. No activity other than NOP command is allowed when the core enters the read retraining mode. Once asserted, `rt_req_i` should remain asserted until the `rt_done_o` signal is sampled High. The `rt_act_o` signal indicates that the IP core is in the read training mode, and the LPDDR2 interface should remain idle with only NOP commands until the retraining process is completed. The `rt_done` signal is asserted only for one clock cycle, and the `rt_req_i` signal must be deasserted immediately after the sampling of the `rt_done` assertion to avoid unnecessary reentering the training. The `rt_err_o` signal is asserted if the read training is not successful. Since a failure during the read training usually causes the IP core unable to transfer the read data from the LPDDR2 SDRAM device to the FPGA fabric, `rt_err_o` should be the first signal that you need to check if the LPDDR2 interface does not work properly.

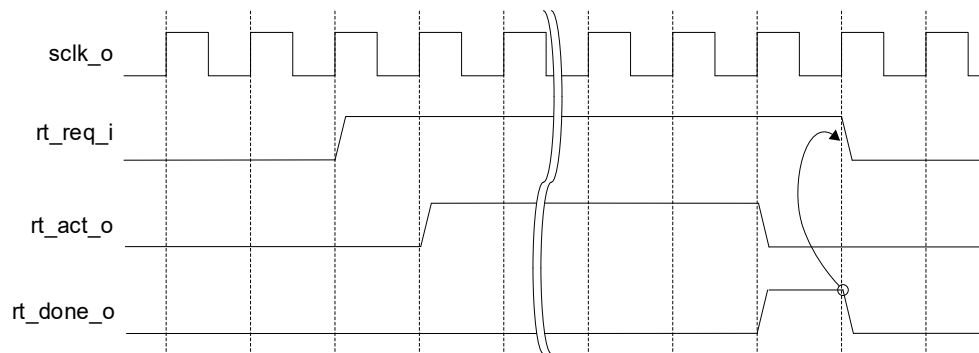


Figure 2.3. Timing Diagram for Read Retraining

### 2.5.2. Command and Address

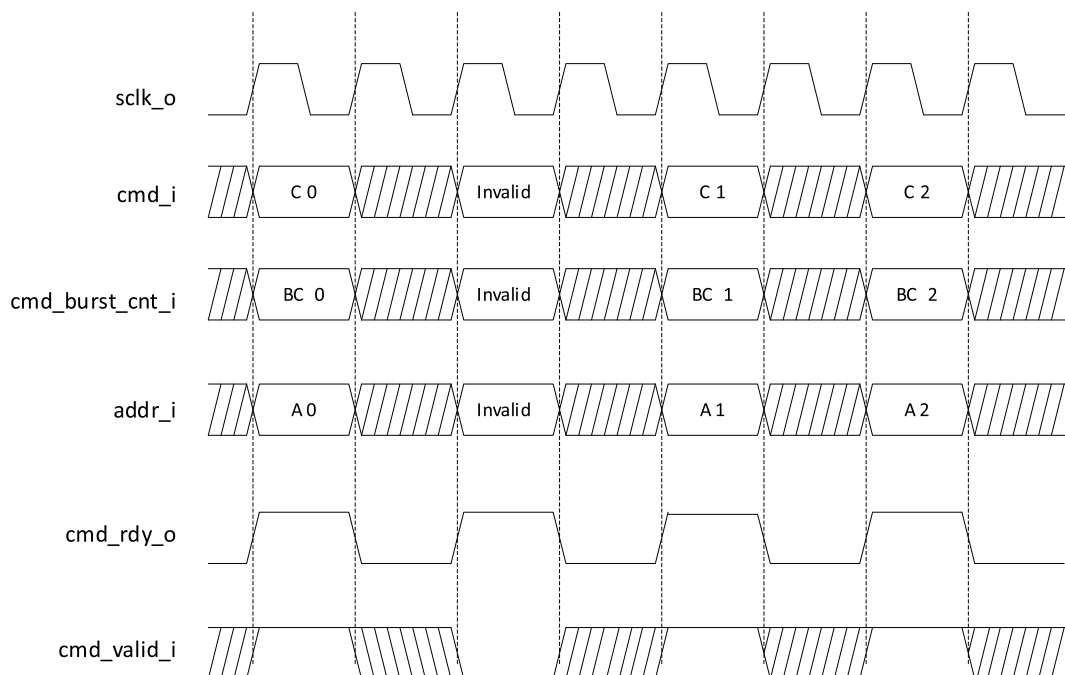
After memory initialization is completed, the IP Core waits for user commands in order to set up or access the SDRAM device. The user logic needs to provide the command and address to the core along with the control signals. The commands and addresses are delivered to the core using the following procedure.

The memory controller core informs the user logic that it is ready to receive a command by asserting the `cmd_rdy_o` signal for one cycle. If the core finds the `cmd_valid_i` signal asserted by the user logic while its `cmd_rdy_o` is asserted, it takes the `cmd` input as a valid user command. Usually, `cmd_valid_i` is deasserted at the rising edge of the clock that samples `cmd_rdy_o` high. The core also accepts the `addr_i` input as a valid start address or mode register programming data depending on the command type. Along with `addr_i` input the core also accepts the signal `cmd_burst_cnt_i`. If `cmd_valid_i` is not asserted, the `cmd_i` and `addr_i` inputs become invalid, and the core ignores them. The `cmd_i`, `addr_i`, `cmd_burst_cnt_i` and `cmd_valid_i` inputs become *don't care* while `cmd_rdy_o` is de-asserted. The `cmd_rdy_o` signal is asserted again to accept the next command.

The core is designed to ensure maximum throughput at a burst length of eight by asserting `cmd_rdy_o` once every two-clock cycle unless the command queue is full or there is an intervention on the memory interface such as Auto-Refresh cycles.

When the core is in the command burst operation, it extensively occupies the data bus. During this time, the core prevents `cmd_rdy_o` from being asserted until the command burst is completed. While the core is operating in the command burst mode, it can keep maximum throughput by internally replicating the command. The memory controller repeats the given READ or WRITE command up to 32 times. The `cmd_burst_cnt_i` input is used to set the number of repeats of the given command. The core allows the command burst function to access the memory addresses within the current page. When the core reaches the boundary of the current page while accessing the memory in the command burst mode, the next address that the core accesses becomes the beginning of the same page. This results in overwriting the contents of the location or reading unexpected data. Therefore, you must track the accessible address range in the current page while the command burst operation is performed. If an application requires a fixed command burst size, use of 2-, 4-, 8-, 16- or 32-burst is recommended to ensure that the command burst accesses do not cross the page boundary. When `cmd_burst_cnt_i` is 0, the controller performs 32 commands (reads or writes). The `cmd_burst_cnt` input is sampled the same way as `cmd` signal. The timing diagram of the Command and Address group is shown in Figure 2.4. The timing for burst count `i` in Figure 2.4 shows only the sampling time of the bus. When `cmd_burst_cnt_i` is sampled with a value greater than 5'b00001 and the command queue

becomes full, the cmd\_rdy\_o signal is not asserted, and the memory address is automatically increased by the core until the current command burst cycle is completed.



**Figure 2.4. Timing of Command and Address**

### 2.5.3. User Commands

You can initiate a request to the memory controller by loading a specific command code in cmd input along with other information such as memory address. The command on the cmd\_i signal must be a valid command. Lattice defines a set of valid memory commands as shown in Table 2.5. All other values are reserved and considered invalid.

**Table 2.5. Defined User Commands**

| Command                   | Mnemonic     | cmd_i[3:0] |
|---------------------------|--------------|------------|
| Read                      | READ         | 4'b0001    |
| Write                     | WRITE        | 4'b0010    |
| Read with Auto Precharge  | READA        | 4'b0011    |
| Write with Auto Precharge | WRITEA       | 4'b0100    |
| Power down Entry          | PDOWN_ENT    | 4'b0101    |
| Load Mode Register        | LOAD_MR      | 4'b0110    |
| Self-Refresh Entry        | SEL_REF_ENT  | 4'b1000    |
| Self-Refresh Exit         | SEL_REF_EXIT | 4'b1001    |
| Power down Exit           | PDOWN_EXIT   | 4'b1011    |
| ZQ Calibration Long       | ZQ_LNG       | 4'b1100    |
| ZQ Calibration Short      | ZQ_SHRT      | 4'b1101    |

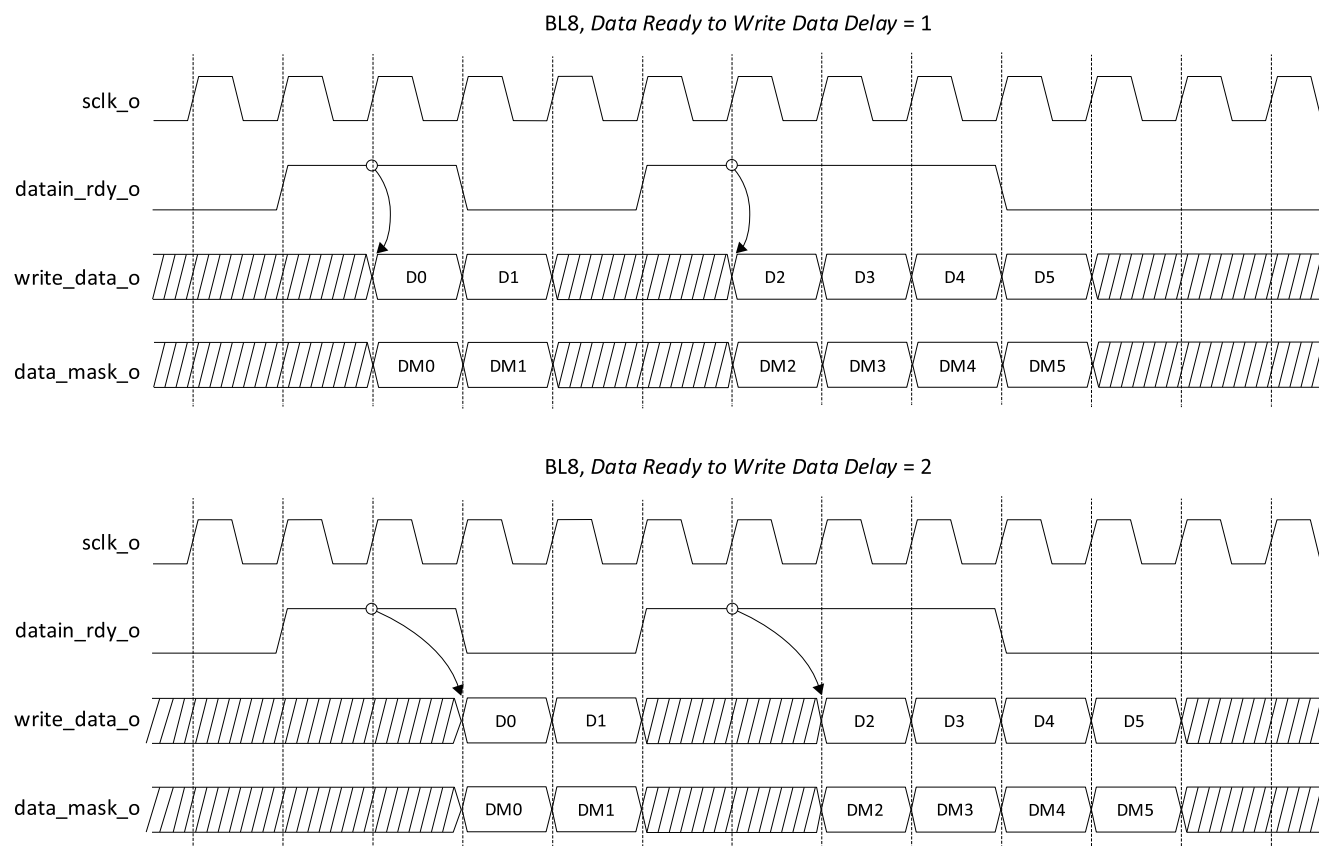
**Notes:**

- The controller accepts only the command codes listed above as legal commands. Any other command code is discarded as invalid command.
- The controller discards Self-Refresh Entry or Power Down Entry command if the memory is already in Self-refresh mode or Power Down mode respectively.
- The controller discards Self-Refresh Exit or Power Down Exit command if the memory is already not in Self-refresh mode or Power Down mode respectively.

## 2.5.4. WRITE

The user initiates a memory write operation by asserting `cmd_valid_i` along with the WRITE or WRITEA command and the address. After the WRITE command is accepted, the memory controller core asserts the `datain_rdy_o` 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_o` signal is asserted is not fixed, the user logic needs to monitor the `datain_rdy_o` signal. Once `datain_rdy_o` is asserted, the core expects valid data on the `write_data_i` signal 1, 2 or 3 clock cycles after the `datain_rdy_o` signal is asserted. The write data delay is programmable by the user, by setting desired value to *Data Ready to Write Data Delay* attribute, providing flexible backend application support. For example, setting the value to 2 ensures that the core takes the write data in proper time when the local user interface of the core is connected to a synchronous FIFO module inside the user logic. Figure 2.5 shows two examples of the local user interface data write timing. Both cases are in BL8 mode. The upper diagram shows the case of one clock cycle delay of write data, while the lower one displays a two clock-cycle delay case. The memory controller considers D0, DM0 through D5, DM5 valid write data.

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 current bank, an ACTIVE command is generated by the controller to the memory to open the current row first. The memory controller then 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. The memory controller then issues a WRITE command to the memory. If current row is already opened, only a WRITE command (without any ACTIVE or PRECHARGE commands) is sent to the memory.



**Figure 2.5. Timing One-Clock vs. Two-Clock Write Data Delay**

### 2.5.5. WRITEA

WRITEA is treated the same way as WRITE command, except that the IP 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 upon completing the write operation.

### 2.5.6. 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_o` signal to tell the user logic that the valid read data is on the `read_data_o` signal. The timing diagram of read data on the local user interface is shown in Figure 2.6.

Read operation follows the same row status checking scheme as mentioned in write operation. Depending on current row status the memory controller generates ACTIVE and PRECHARGE commands as required. Refer to the description mentioned in Write operation for more details.

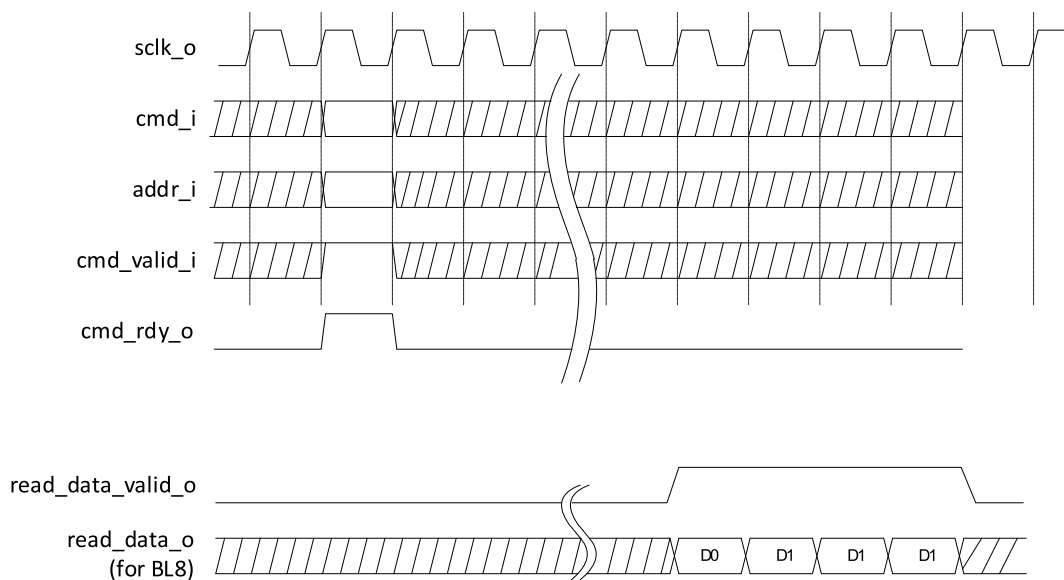


Figure 2.6. User-Side Read Operation

### 2.5.7. READA

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

### 2.5.8. REFRESH Support

Since LPDDR2 SDRAM devices have at most an 8192-deep Auto Refresh command queue as per JEDEC JESD209-2B specification, Lattice's LPDDR2 memory controller core can support up to 8192 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 ( $TREFI\_AB$  attribute) times the *Auto Refresh Burst Count* as selected in the Module/IP Block Wizard.

It is recommended that the optimum number of burst Refresh commands be used if the LPDDR2 interface throughput is a major concern of the system. If the *Auto Refresh Burst Count* attribute is set to  $n$ , for example, the core sends a set of  $n$  consecutive Auto Refresh commands to the memory at once when it reaches the time period of the  $n$  refresh intervals ( $TREFI\_AB \times n \times sclk\_o$  period). Sending out  $n$  number of refresh commands in a burst takes about  $n \times 4 \times TRFC\_AB \times sclk\_o$  period in All bank refresh mode. During this time, no other command is sent to the memory.

When a refresh burst is used, the controller issues a Pre-charge command only for the first Refresh command and the subsequent Refresh commands of the burst are issued without the associated Pre-charge commands. This is to improve the LPDDR2 throughput.

Alternatively, you can enable the *External Auto Refresh Port* attribute which adds an input signal `ext_auto_ref_i` and an output signal `ext_auto_ref_ack_o` 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_i` is asserted. Completion of refresh burst is indicated by the output signal `ext_auto_ref_ack_o`.

In an application where explicit memory refresh is not necessary, you can enable *External Auto Refresh Port* attribute and keep the `ext_auto_ref_i` signal deasserted.

The user logic can use the `sref_ent_cmd_o` and `sref_ext_cmd_o` signal to determine if the IP Core enters and exits self-refresh operation.

## 2.6. 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. You must know how the local address lines from the memory controller connect to those address lines from the memory because proper local-to-memory address mapping is crucial to meet the system requirements in applications such as a video frame buffer controller. Even for other applications, careful address mapping is generally necessary to optimize the system performance. In the memory side, the address (A), bank address (BA) and chip select (CS) inputs are used for addressing a memory device. You can obtain this information from a given data sheet. Figure 2.7 shows the local-to-memory address mapping of the Lattice LPDDR2 SDRAM Controller Lite IP Core.

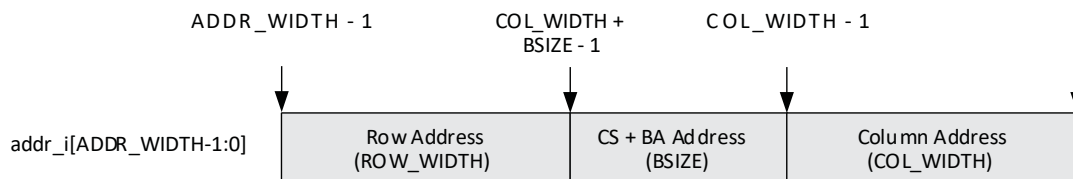


Figure 2.7. Local-to-Memory Address Mapping for Memory Access

`ADDR_WIDTH` is calculated by the sum of `COL_WIDTH`, `ROW_WIDTH` and `BSIZE`. `BSIZE` is determined by the sum of the `BANK_WIDTH` and `CS_WIDTH`. Since the number of chip select is 1, the chip select address size becomes 0. An example of a typical address mapping is shown in Table 2.6 and Figure 2.8.

Table 2.6. Address Mapping Example

| Attribute Name                  | Example User Value | Actual Line Size | Local Address Map          |
|---------------------------------|--------------------|------------------|----------------------------|
| Column Size                     | 11                 | 11               | <code>addr_i[10:0]</code>  |
| Bank Address Width              | 3                  | 3                | <code>addr_i[13:11]</code> |
| Rank Size (or Chip Select Size) | Single             | 0                | —                          |
| Row Size                        | 14                 | 14               | <code>addr_i[27:14]</code> |
| Total Local Address Line Size   |                    | 29               | <code>addr_i[27:0]</code>  |

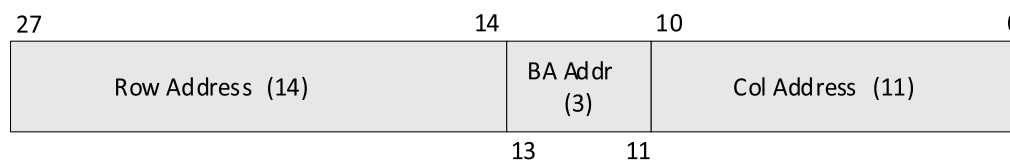
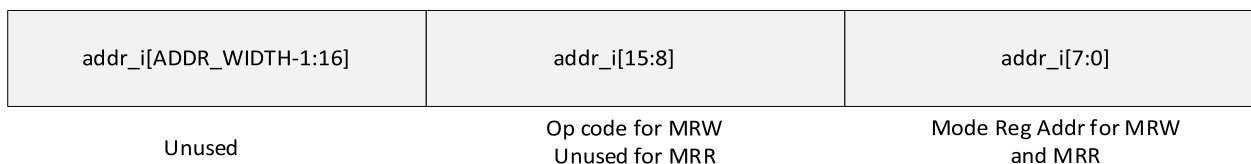


Figure 2.8. Mapped Address for the Example

## 2.7. Mode register Programming

The LPDDR2 SDRAM devices are programmed using a set of up to 256 Mode Registers. Mode Register address and the Opcode to the Mode Register (for MRW only) are sent to the memory device through the `em_dds_ca_o` signal along with the MRW/MRR command. The memory data (`em_dds_data_io`) signal cannot be used for the Mode Register programming.

The Lattice LPDDR2 memory controller core uses the local address signal, `addr_i`, to program these registers. The core accepts a user command, MRW, to initiate the programming of mode registers. When MRW is applied on the `cmd_i` signal, the user logic must provide the information for the targeted mode register and the programming data on the `addr_i` signal. When the target mode register is programmed, the memory controller core is also configured to support the new memory setting. Figure 2.7 and Table 2.7 show the bit mapping of local address signal, `addr_i` for the programming of memory registers.



**Figure 2.9. User-to-Memory Address Mapping for MR Programming**

The register address (8 bits) is provided through the lower side of the `addr_i` signal starting from the bit 0 for LSB. The programming data uses the next 8 bits.

**Table 2.7. Mode Register Selection Using Address Bits**

| Mode Register | ( <code>addr_i[7:0]</code> ) |
|---------------|------------------------------|
| MR0           | 8'h00                        |
| MR1           | 8'h01                        |
| ...           | ...                          |
| MR255         | 8'hFF                        |

The initialization process uses the Mode register initial values selected through the interface. If these registers are not further programmed by the user logic, using MRW user command, they remain in the configurations programmed during the initialization process. Table 2.8 shows the list of available parameters and their initial default values from the interface if they are not changed by the user.

**Table 2.8. Initialization Default Values for Mode Register Setting**

| Mode Register | Register Field         | Default Value | Description | Local Address                      | Module/IP Block Wizard Setting |
|---------------|------------------------|---------------|-------------|------------------------------------|--------------------------------|
| MR1           | Burst Length           | 3'b011        | BL = 8      | <code>addr_i[10:8]</code>          | Yes                            |
|               | WR Recovery            | 3'b100        | 6           | <code>addr_i[15:13]</code>         | Yes                            |
| MR2           | Read and Write Latency | 4'b0100       | RL=6, WL=3  | <code>addr_i[11:8]</code>          | Yes                            |
| MR3           | Drive strength         | 4'b0010       | 40 Ω PD/PU  | <code>addr_i[ROW_WIDTH-1:0]</code> | Yes                            |

## 3. Core Generation, Simulation, and Validation

This section provides information on how to generate the LPDDR2 SDRAM Controller Lite IP Core using the Lattice Radiant software and how to run simulation and synthesis. For more details on the Lattice Radiant software, refer to the [Lattice Radiant Software 2025.1 User Guide](#).

### 3.1. Licensing the IP

An IP core-specific license string is required enable full use of the LPDDR2 SDRAM Controller Lite IP Core in a complete, top-level design. You can fully evaluate the IP core through functional simulation and implementation (synthesis, map, place and route) without an IP license string. This IP core supports Lattice’s IP hardware evaluation capability, which makes it possible to create versions of the IP core, which operate in hardware for a limited time (approximately four hours) without requiring an IP license string. See Hardware Evaluation section for further details. However, a license string is required to enable timing simulation and to generate bitstream file that does not include the hardware evaluation timeout limitation.

### 3.2. Generation and Synthesis

The Lattice Radiant software allows you to customize and generate modules and IPs and integrate them into the device’s architecture. The procedure for generating the LPDDR2 SDRAM Controller Lite IP Core in Lattice Radiant software is described below.

To generate the LPDDR2 SDRAM Controller Lite IP Core:

1. Create a new Lattice Radiant software project or open an existing project.
2. In the **IP Catalog** tab, double-click on **LPDDR2\_SDRAM\_Controller\_Lite** under **IP, Processors, Controllers, and Peripherals** category. The **Module/IP Block Wizard** opens as shown in [Figure 3.1](#). Enter values in the **Component name** and the **Create in** fields and click **Next**.

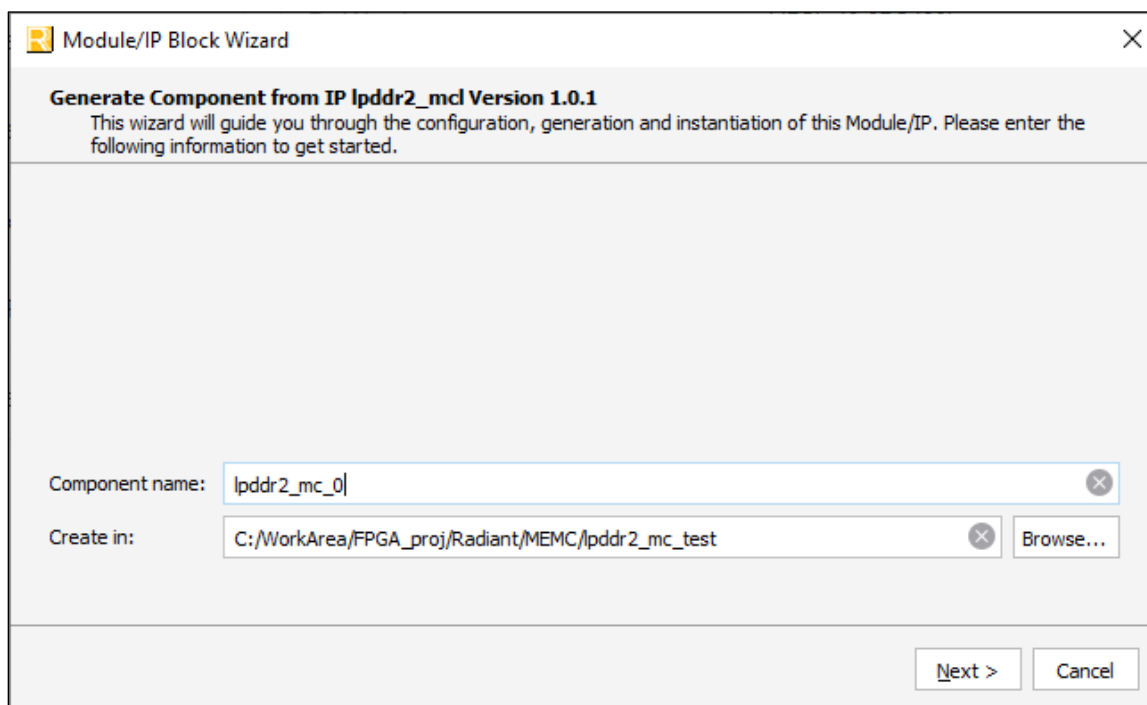
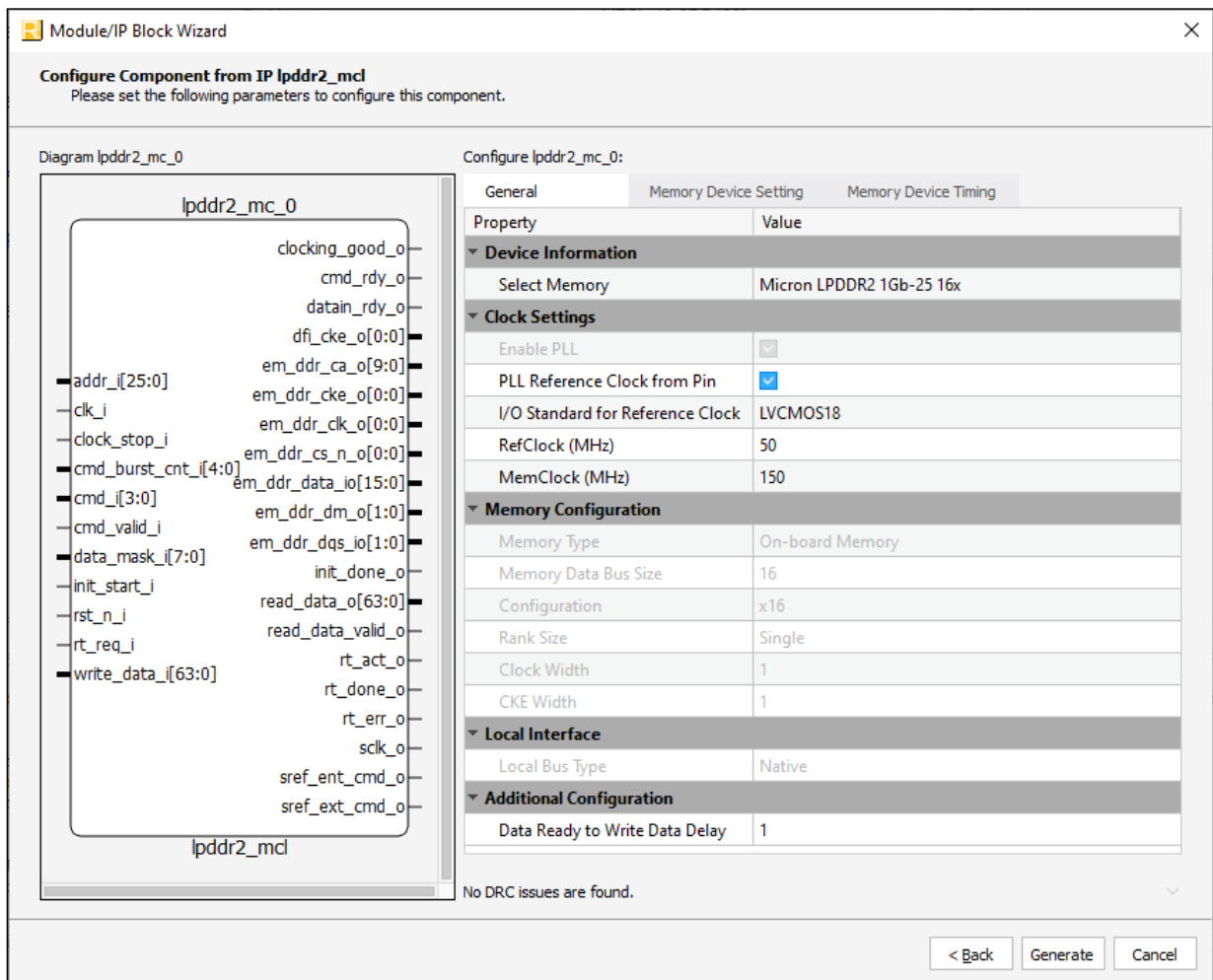


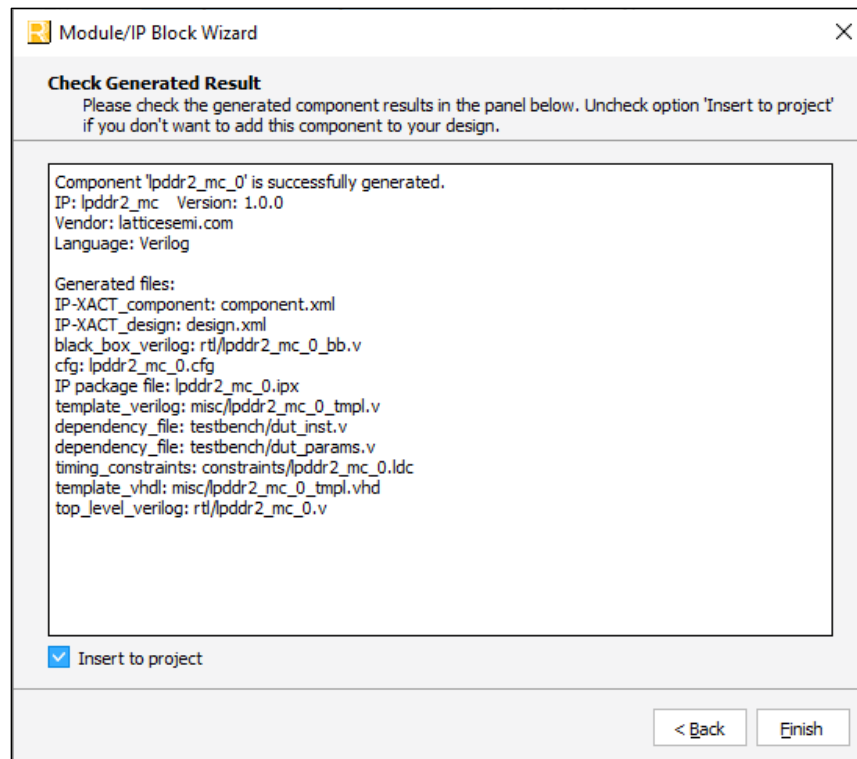
Figure 3.1. Module/IP Block Wizard

3. In the module's dialog box of the **Module/IP Block Wizard** window, customize the selected LPDDR2 SDRAM Controller Lite IP Core using drop-down menus and check boxes. As a sample configuration, see [Figure 3.2](#). For configuration options, see the [Attributes Summary](#) section.



**Figure 3.2. Module/IP Block Wizard of LPDDR2 SDRAM Controller Lite IP Core**

4. Click **Generate**. The **Check Generating Result** dialog box opens, showing design block messages and results as shown in [Figure 3.3](#).



**Figure 3.3. Check Generating Result**

- Click the **Finish** button. All the generated files are placed under the directory paths in the **Create in** and the **Component name** fields shown in [Figure 3.1](#).

The generated LPDDR2 SDRAM Controller Lite IP Core package includes the black box (<Component name>\_bb.v) and instance templates (<Component name>\_tmpl.v/vhd) that can be used to instantiate the core in a top-level design. An example RTL top-level reference source file (<Component name>.v) that can be used as an instantiation template for the IP core is also provided. You may also use this top-level reference as the starting template for the top-level for their complete design. The generated files are listed in [Table 3.1](#).

**Table 3.1. Generated File List**

| Attribute   | Description  |
|---|--|
| <Component name>.ipx  | This file contains the information on the files associated to the generated IP.  |
| <Component name>.cfg  | This file contains the parameter values used in IP configuration.  |
| component.xml   | Contains the ipxact:component information of the IP.   |
| design.xml  | Documents the configuration parameters of the IP in IP-XACT 2014 format.   |
| rtl/<Component name>.v  | This file provides an example RTL top file that instantiates the IP core.  |
| rtl/<Component name>_bb.v                                       | This file provides the synthesis black box.  |
| misc/<Component name>_tmpl.v<br>misc /<Component name>_tmpl.vhd | These files provide instance templates for the IP core.  |
| eval/top_constraint.pdc   | Guide for setting clock constraint in the post-synthesis constraint files  |
| eval/eval_top.v   | Top level RTL files that may be used for running Lattice Radiant software flow check (synthesis to export) on the generated IP. Without this, the Radiant software Map process fails due to not having enough I/O. This is mainly used for checking resource utilization and fmax for the selected IP configuration, this is not for implementation. |
| eval/lscs_simple_lfsr.v   | A simple linear-feedback shift register.   |
| eval/dut_inst.v   | A sample instantiation of the generated IP. This is included by the eval_top.v.  |
| eval/dut_params.v   | Lists the equivalent localparams of the user settings. This is included by the eval_top.v.   |

### 3.2.1. Required Post-Synthesis Constraints

The LPDDR2 SDRAM Controller IP Core has one PLL reference clock input and three internally-generated clocks. You need to constrain these clocks in the post-synthesis constraint file of your Lattice Radiant software project.

The eval/top\_constraint.pdc file described in Table 3.1 is a constraint file that is generated based on the user settings. You need to copy the contents of this file to your post synthesis constraint file for your Lattice Radiant project.

### 3.3. Running Functional Simulation

The initialization of LPDDR2 memories take around 200  $\mu$ s. This takes a long time to simulate. After checking that the initialization works, it is usually desired to bypass (reduce to few clock cycles) these process in the succeeding simulation to save time. Bypassing these requires manual update on the generated IP RTL file and testbench file as follows:

- Modify the <generated\_ip\_path>/rtl/<generated\_ip\_name>.v and set SIMULATION parameter to 1. For example:


```
lpddr2_mcl_0_ipgen_lscclpddr2_mc #(.FAMILY("LIFCL"),
    .SIMULATION(1), // Set SIMULATION=1 for simulation only
    .SELECT_MEMORY("1Gb"),
```

- The file <generated\_ip\_path>/testbench/dut\_params.v contains the user-defined parameters. This is used for configuring the test. You should set the localparam SIMULATION to 1 so that the test knows that you bypass reset and initialization.

```
localparam SIMULATION = 1;
```

You should ensure that the generated IP that you used for bitstream generation has SIMULATION parameter set to 0 so that the IP Core can generate correct initialization period which is necessary for proper LPDDR2 SDRAM operation.

To run the functional simulation:

- Add the top-level testbench file, tb\_top.v in the project as a simulation file, click **File** Tab, then select **Add** in the drop-down menu, then click **Existing Simulation File**. Then select <Component name>/testbench/tb\_top.v.
- Click the  button located on the **Toolbar** to initiate the **Simulation Wizard** shown in Figure 3.4.

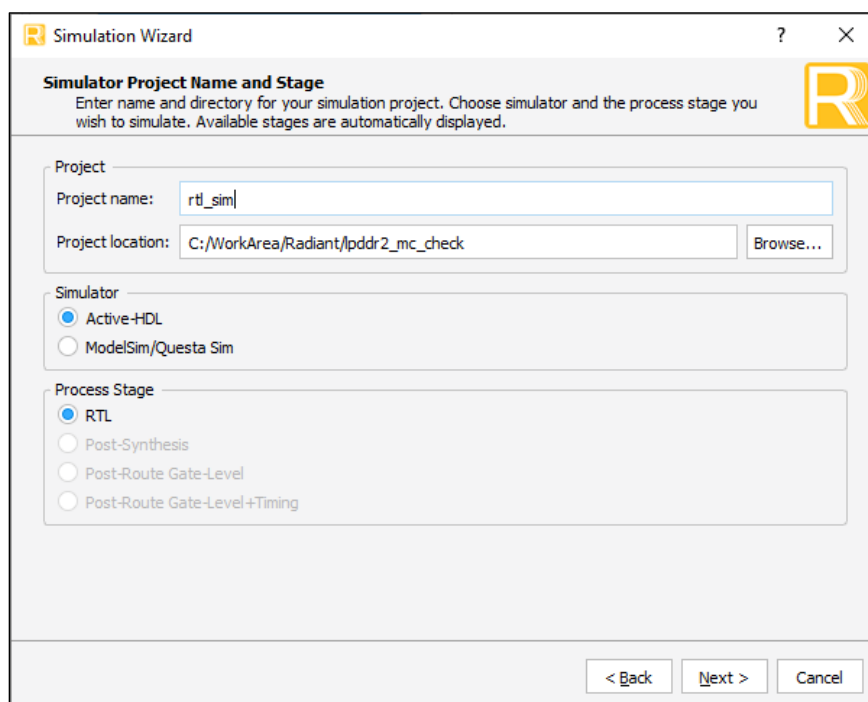


Figure 3.4. Simulation Wizard

- Click **Next** to open the **Add and Reorder Source** window as shown in Figure 3.5. Notice that the **Source Files** area only contain the generated IP (<Component name>.v) and the tb\_top.v, which is added in Step 1. The tb\_top.v includes all the necessary test files for simulation.

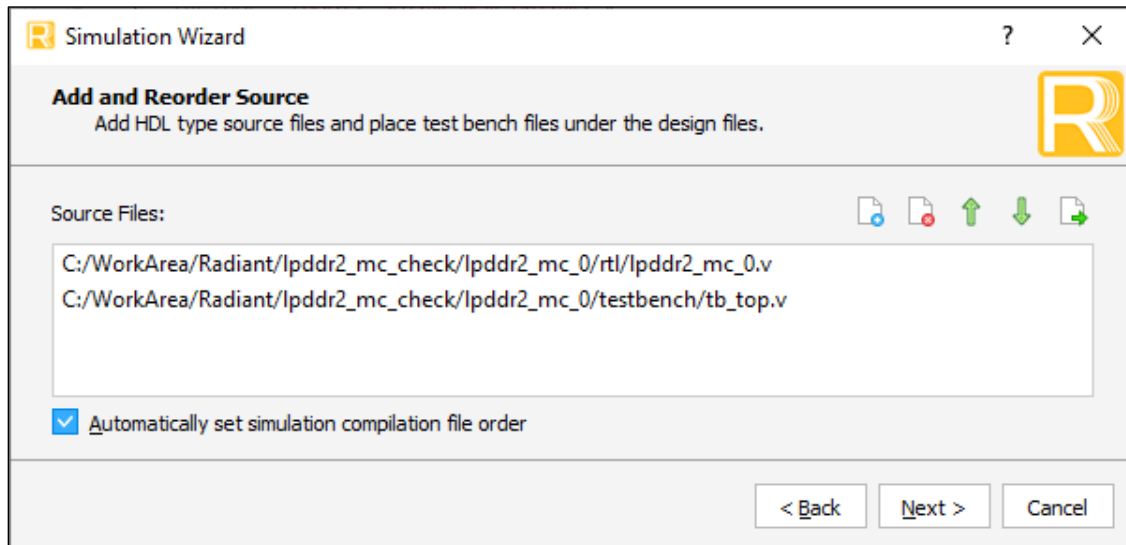


Figure 3.5. Adding and Reordering Source

- Click **Next**. The **Summary** window is shown; click **Finish** to run the simulation.

**Note:** It is necessary to follow the procedure above until it is fully automated in the Lattice Radiant software suite. The results of the simulation in our example are provided in Figure 3.6.

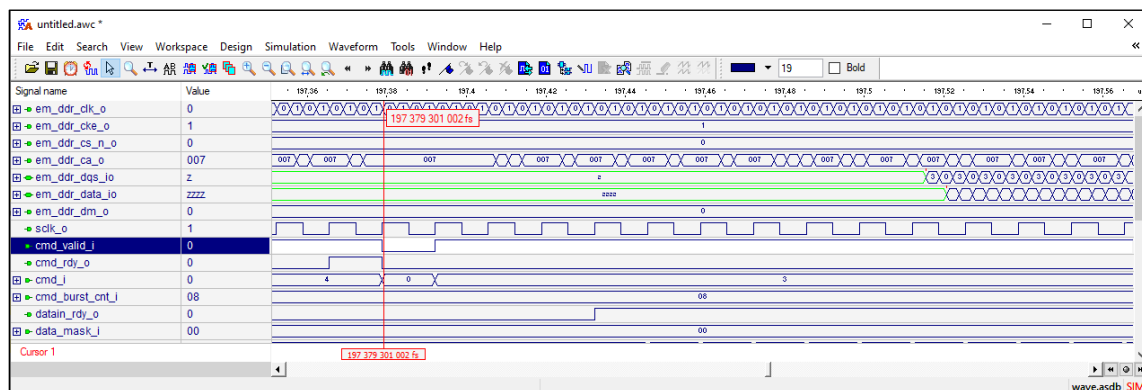


Figure 3.6. Simulation Waveform Hardware Evaluation

The LPDDR2 SDRAM Controller Lite IP Core supports Lattice's IP hardware evaluation capability when used with CrossLink-NX devices. This 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. The hardware evaluation capability may be enabled/disabled in the Strategy dialog box. It is enabled by default. To change this setting, go to Project > Active Strategy > LSE/Synplify Pro Settings.

## 4. Ordering Part Number

The Ordering Part Number (OPN) for this IP Core are the following:

- LPDDR2L-CN-X-U – LPDDR2 SDRAM Controller Lite for CrossLink-NX – Single Design License
- LPDDR2L-CN-X-UT – LPDDR2 SDRAM Controller Lite for CrossLink-NX – Site License
- LPDDR2L-CT-N-X-U – LPDDR2 SDRAM Controller Lite for Certus-NX – Single Design License
- LPDDR2L-CT-N-X-UT – LPDDR2 SDRAM Controller Lite for Certus-NX – Site License
- LPDDR2L-CP-N-X-U – LPDDR2 SDRAM Controller Lite for CertusPro-NX – Single Design License
- LPDDR2L-CP-N-X-UT – LPDDR2 SDRAM Controller Lite for CertusPro-NX – Site License

## 5. Known Issue

This section documents known issues that may affect design behavior, tool functionality, or expected outcomes when using specific versions of Radiant and related synthesis tools.

### 5.1. Timing Constraints Not Recognized in Lattice Radiant 2023.1 and Later

In Radiant version 2023.1 or later, timing constraints defined in eval/top\_constraint.pdc may not be applied during synthesis when using the Synplify Pro Synthesis tool. This can result in timing violations or unexpected behavior in the final implementation.

#### 5.1.1. Root Cause

The issue is caused by updates to the Synplify Pro Synthesis Tool included with Radiant 2023.1 and later versions.

#### 5.1.2. Workaround

To ensure timing constraints are recognized during synthesis, revise the constraint file using the following format:

- #Constraints for PLL reference clock  

```
create_clock -name {clk_i} -period 20 [get_ports clk_i]
```
- #Constraint for the PLL-generated clocks  

```
create_generated_clock -name {clkop} -source [get_pins
*/lssc_lpddr2_mc_inst/*/*/*.PLL_inst/REFCK] -divide_by 1 -multiply_by 3 [get_pins
*/lssc_lpddr2_mc_inst/*/*/*.PLL_inst/CLKOP]
create_generated_clock -name {clkos} -source [get_pins
*/lssc_lpddr2_mc_inst/*/*/*.PLL_inst/REFCK] -divide_by 1 [get_pins
*/lssc_lpddr2_mc_inst/*/*/*.PLL_inst/CLKOS]
```
- # Constrain the clocks generated in the LPDDR2 SDRAM PHY  

```
create_generated_clock -name {eclkout_w} -source [get_pins
*/lssc_lpddr2_mc_inst/*/*/*.PLL_inst/CLKOP] -divide_by 1 [get_pins
*/lssc_lpddr2_mc_inst/*/*/*/*/ECLKOUT]
create_generated_clock -name {sclk_o} -source [get_pins
*/lssc_lpddr2_mc_inst/*/*/*/*/ECLKSYNC_inst/ECLKOUT] -divide_by 2 [get_pins
*/lssc_lpddr2_mc_inst/*/*/*/*/ECLKDIV_inst/DIVOUT]
set_clock_groups -group [get_clocks {clkop eclkout_w sclk_o}] -group [get_clocks
clkos] -asynchronous
set_false_path -from [get_pins -hierarchical
*/lssc_lpddr2_mc_inst/*/*/*.PLL_inst/LOCK]
```

## Appendix A. Resource Utilization

Table A.1 show configuration and resource utilization for LIFCL-40-9BG400I using Synplify Pro of Lattice Radiant software 2.1.

**Table A.1. Resource Utilization**

| Configuration                      | sclk_o Fmax (MHz) | Registers | LUTs | EBR | IDDR/ODDR/TDDR |
|------------------------------------|-------------------|-----------|------|-----|----------------|
| Default (MemClock=150 MHz)         | 177.368           | 1762      | 2104 | 0   | 67             |
| MemClock=300 MHz, Others = Default | 194.477           | 1762      | 2067 | 0   | 67             |
| MemClock=350 MHz, Others = Default | 194.212           | 1762      | 2070 | 0   | 67             |

**Notes:**

- The sclk\_o Fmax is generated using the eval\_top.v that is described in Table 3.1. This design only contains the LPDDR2 SDRAM Controller Lite IP Core and a few linear-feedback shift registers. These values may be reduced when the IP Core is used with the user logic.
- The *distributed RAM* utilization is accounted for in the total LUT4 utilization. The actual LUT4 utilization is distribution among *logic*, *distributed RAM*, and *ripple logic*.

## References

- [Lattice Radiant Software 2025.1 User Guide](#)
- [CrossLink-NX FPGA](#) web page
- [Certus-NX FPGA](#) web page
- [CertusPro-NX FPGA](#) web page
- <https://www.jedec.org>
- [Lattice Radiant Software](#) web page
- [Lattice Propel Design Environment](#) web page
- [Lattice Insights](#) for Lattice Semiconductor training courses and learning plans

## Technical Support Assistance

Submit a technical support case through [www.latticesemi.com/techsupport](http://www.latticesemi.com/techsupport).

For frequently asked questions, refer to the Lattice Answer Database at [www.latticesemi.com/Support/AnswerDatabase](http://www.latticesemi.com/Support/AnswerDatabase)

## Revision History

### Revision 1.6, IP v1.4.0, December 2025

| Section                       | Change Summary  |
|-------------------------------|---|
| All                           | Added IP v1.4.0.  |
| Abbreviations in The Document | <ul style="list-style-type: none"> <li>Replaced <i>Acronyms</i> with <i>Abbreviations</i>.</li> <li>Updated this section.</li> </ul>  |
| Introduction                  | <ul style="list-style-type: none"> <li>Updated the format of <a href="#">Table 1.1. Quick Facts</a>.</li> <li>Updated the Lattice Implementation row under <a href="#">Table 1.1. Quick Facts</a>. <ul style="list-style-type: none"> <li>Added IP Core v1.4.0 - Lattice Radiant software 3.0.</li> <li>Removed IP Core v1.0.0 – Lattice Radiant software 2.0, IP Core v1.1.0 – Lattice Radiant software 2.1, and IP Core v1.3.0 – Lattice Radiant software 3.0.</li> </ul> </li> </ul> |
| Known Issue                   | Added this section.   |
| Technical Support Assistance  | Added reference to the Lattice Answer Database on the Lattice website.  |

### Revision 1.5, June 2021

| Section                | Change Summary   |
|------------------------|--|
| Introduction           | Updated Table 1.1 to add CertusPro-NX support and update IP Core – Lattice Radiant Design version. |
| Ordering Part Number   | Added part number for CertusPro-NX.  |
| Appendix B. Limitation | Removed this section.  |
| References             | Updated this section to add CertusPro-NX web page.   |

### Revision 1.4, October 2020

| Section                                     | Change Summary   |
|---|--|
| Introduction                                | <ul style="list-style-type: none"> <li>Added LIFCL-17 and LFD2NX-17 in the Target Devices.</li> <li>Updated reference to the Lattice Radiant Software User Guide.</li> </ul> |
| Functional Descriptions                     | Updated selectable values in Table 2.2 for the following attributes:<br>I/O Standard for Reference Clock, RefClock (MHz), MemClock (MHz).                                    |
| Core Generation, Simulation, and Validation | Updated reference to the Lattice Radiant Software User Guide.  |
| References                                  | Updated this section.  |

### Revision 1.3, August 2020

| Section                          | Change Summary     |
|----------------------------------|--------------------|
| Appendix A. Resource Utilization | Updated Table A.1. |

### Revision 1.2, June 2020

| Section                                     | Change Summary  |
|---|---|
| All   | Updated Lattice Radiant software user guide references to version 2.1 across the document.  |
| Introduction                                | <ul style="list-style-type: none"> <li>Added Certus-NX and LFD2NX-40 in Supported FPGA Family and Target Devices, respectively, and added IP Core v1.1.x in Lattice Implementation row in Table 1.1.</li> <li>Updated Features to limit memory data path width and device configurations to 16 and x16 respectively due to the limitation.</li> </ul> |
| Functional Descriptions                     | <ul style="list-style-type: none"> <li>Updated Select Memory and Memory Data Bus Size entries in Table 2.2 in the Attributes Summary section.</li> <li>Added MemClock Tolerance and MemClock Actual Frequency attributes in Table 2.2 and Table 2.3.</li> </ul>   |
| Core Generation, Simulation, and Validation | <ul style="list-style-type: none"> <li>Updated Table 3.1 for the added eval folder.</li> <li>Added Required Post-Synthesis Constraints section.</li> </ul>  |
| Ordering Part Number                        | Updated this section.   |

| Section                          | Change Summary        |
|----------------------------------|-----------------------|
| Appendix A. Resource Utilization | Updated Table A.1.    |
| Appendix B. Limitation           | Added this section.   |
| References                       | Updated this section. |

#### Revision 1.1, February 2020

| Section                                     | Change Summary  |
|---|---|
| Functional Description                      | Added sref_ent_cmd_o and sref_ext_cmd_o signals in Figure 2.1.<br>Fixed signal names in Figure 2.2. |
| Core Generation, Simulation, and Validation | Updated Figure 3.1 and Figure 3.2 for IP Core v1.0.1.   |

#### Revision 1.0, December 2019

| Section | Change Summary   |
|---------|------------------|
| All     | Initial release. |



[www.latticesemi.com](http://www.latticesemi.com)