



# **LPDDR4 Memory Controller for Nexus Devices**

## **User Guide**

FPGA-IPUG-02127-1.5

September 2023

## 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 associated risk the responsibility entirely of the Buyer. Buyer shall not rely on any data and performance specifications or parameters provided herein. 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. No Lattice products should be used in conjunction with mission- or safety-critical or any other application in which the failure of Lattice's product could create a situation where personal injury, death, severe property or environmental damage may occur. 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
Acronyms in This Document .....	7
1. Introduction .....	8
1.1. Quick Facts .....	8
1.2. Features.....	8
1.3. Licensing and Ordering Information.....	10
1.4. IP Validation Summary .....	11
1.5. Minimum Device Requirements.....	11
1.6. Naming Conventions .....	11
1.6.1. Nomenclature.....	11
1.6.2. Signal Names .....	11
2. Functional Description.....	12
2.1. IP Architecture.....	12
2.1.1. Soft Memory Controller .....	12
2.1.2. Soft PHY .....	13
2.1.3. Soft Training Engine.....	14
2.2. Clocking and Reset .....	15
2.3. User Interfaces .....	16
2.3.1. Data Interface Protocols.....	16
2.3.2. Configuration Interface Protocol.....	17
2.4. LPDDR4 Calibration .....	18
2.4.1. Initialization and Training Sequence .....	18
2.4.2. Initialization and Training without APB Interface.....	21
2.5. LPDDR4 Operation Description .....	21
2.5.1. Write and Read Data Access.....	21
2.5.2. Auto Refresh Support .....	21
2.5.3. Power Saving Feature .....	21
2.5.4. Periodic ZQ Calibration .....	22
2.5.5. Temperature Tracking and Extended Temperature Support .....	22
3. IP Parameter Description.....	23
3.1. General.....	23
3.2. Memory Device Timing .....	25
3.3. Training Settings.....	27
4. Signal Description .....	30
4.1. Clock and Reset .....	30
4.2. Interrupts and Initialization/Training .....	30
4.3. AHB-Lite Data Interface.....	31
4.4. AXI4 Data Interface .....	31
4.5. APB Register Interface.....	32
4.6. LPDDR4 Memory Interface.....	33
5. Register Description.....	34
5.1. Feature Control Register (FEATURE_CTRL_REG) (0x00) .....	34
5.2. Reset Register (RESET_REG) (0x04) .....	36
5.3. Settings Register (SETTINGS_REG) (0x08) .....	36
5.4. Interrupt Status Register (INT_STATUS_REG) (0x10) .....	37
5.5. Interrupt Enable Register (INT_ENABLE_REG) (0x14) .....	37
5.6. Interrupt Set Register (INT_SET_REG) (0x18) .....	38
5.7. Training Operation Register (TRN_OP_REG) (0x20) .....	38
5.8. Status Register (STATUS_REG) (0x24) .....	39
6. LPDDR4 Memory Controller Example Design .....	41
6.1. Overview .....	41
6.2. Synthesis Example Design .....	41

6.3.	Simulation Example Design .....	43
7.	Designing and Simulating the IP .....	44
7.1.	Generating the IP .....	44
7.1.1.	Creating a Radiant Project.....	44
7.1.2.	Configuring and Generating the IP .....	46
7.2.	Design Implementation .....	49
7.2.1.	Pin Placement.....	49
7.2.2.	Constraints .....	50
7.3.	Example Design Hardware Evaluation .....	51
7.3.1.	Preparing the Bitstream .....	52
7.3.2.	Running on Hardware.....	53
7.4.	Example Design Simulation .....	55
8.	Debugging .....	59
8.1.	Debug with the Example Design .....	59
8.2.	Debug with Reveal Analyzer.....	59
8.2.1.	Command Bus Training (CBT) .....	61
8.2.2.	Write Leveling.....	61
8.2.3.	Read Training.....	62
8.2.4.	Write Training.....	63
	Appendix A. Resource Utilization .....	64
	References .....	65
	Technical Support Assistance .....	66
	Revision History .....	67

## Figures

Figure 1.1. Enabling Bitstream for IP Evaluation .....	10
Figure 2.1. Memory Controller IP Core Functional Diagram.....	12
Figure 2.2. Lattice PHY .....	14
Figure 2.3. Training Engine .....	15
Figure 2.4. Shortened Initialization Sequence Simulation Waveform (TRN_OP_REG[0]=0) .....	18
Figure 2.5. Command Bus Training .....	19
Figure 2.6. Write Leveling .....	19
Figure 2.7. Read Training .....	20
Figure 2.8. Write Training .....	20
Figure 6.1. Memory Controller Example Design Functional Diagram.....	42
Figure 7.1. Creating a New Radiant Project .....	44
Figure 7.2. New Project Settings.....	45
Figure 7.3. Project Device Settings .....	45
Figure 7.4. Project Synthesis Tool Selection .....	46
Figure 7.5. IP Instance Settings .....	47
Figure 7.6. IP Generation Result .....	48
Figure 7.7. Add Existing File Dialog Box .....	52
Figure 7.8. Radiant Programmer.....	53
Figure 7.9. Serial Terminal Settings .....	54
Figure 7.10. Simulation Wizard.....	56
Figure 7.11. Adding and Reordering Simulation Source Files .....	56
Figure 7.12. Parsing Simulation HDL Files.....	57
Figure 7.13. Simulation Summary.....	57
Figure 7.14. Simulation Result Waveform .....	58
Figure 8.1. Reveal Example: LPDDR4 Training Passes.....	61
Figure 8.2. Reveal Example: Command Bus Training Failure.....	61
Figure 8.3. Command Bus Training Simulation Waveform .....	61
Figure 8.4. Command Bus Training Reveal Capture.....	61
Figure 8.5. Write Leveling Simulation Waveform .....	62
Figure 8.6. Write Leveling Reveal Capture.....	62
Figure 8.7. Read Gate Training Simulation Waveform .....	62
Figure 8.8. Read Gate Training Reveal Capture .....	62
Figure 8.9. Read Deskew Training Simulation Waveform.....	63
Figure 8.10. Read Deskew Training Reveal Capture .....	63
Figure 8.11. Write Training Simulation Waveform .....	63
Figure 8.12. Write Training Reveal Capture.....	63

## Tables

Table 1.1. Quick Facts .....	8
Table 1.2. Features Overview .....	9
Table 1.3. Ordering Part Number .....	10
Table 1.4. IP Validation Summary .....	11
Table 1.5. Minimum Device Requirements .....	11
Table 2.1. MCE Request Handling .....	13
Table 2.2. Supported AHB-Lite Transactions .....	16
Table 2.3. Supported AXI4 Transactions .....	17
Table 3.1. General Attributes .....	23
Table 3.2. Clock Settings Attributes .....	23
Table 3.3. Memory Configuration Attributes .....	24
Table 3.4. Local Data Bus Attributes .....	24
Table 3.5. General Definitions .....	24
Table 3.6. Memory Device Timing Setting Attributes .....	26
Table 3.7. Periodic Event Setting Attributes .....	26
Table 3.8. Memory Device Timing Definitions .....	26
Table 3.9. Training Settings Attributes .....	27
Table 3.10. Trained Values Attributes .....	28
Table 3.11. Training Settings Definitions .....	28
Table 4.1. Clock and Reset Port Definitions .....	30
Table 4.2. Interrupts and Initialization/Training Port Definitions .....	30
Table 4.3. AHB-Lite Interface Port Definitions .....	31
Table 4.4. AXI4 Interface Port Definitions .....	31
Table 4.5. APB Interface Port Definitions .....	32
Table 4.6. LPDDR4 Interface Port Definitions .....	33
Table 5.1. Summary of LPDDR4 Memory Controller IP Registers .....	34
Table 5.2. Register Access Type Definitions .....	34
Table 5.3. Feature Control Register .....	34
Table 5.4. Address Mapping for addr_translation=0 .....	35
Table 5.5. Address Mapping Example .....	36
Table 5.6. Reset Register .....	36
Table 5.7. Settings Register .....	36
Table 5.8. Interrupt Status Register .....	37
Table 5.9. Interrupt Enable Register .....	37
Table 5.10. Interrupt Set Register .....	38
Table 5.11. Training Operation Register .....	38
Table 5.12. Status Register .....	39
Table 6.1. Supported Example Design Configurations .....	41
Table 6.2. Simulation Runtime Summary .....	43
Table 7.1. Memory Controller Attribute Guidelines .....	47
Table 7.2. Generated File List .....	48
Table 7.3. Project Constraints .....	50
Table 7.4. Contents of eval/traffic_gen .....	51
Table 8.1. Reveal Analyzer Signal Definitions .....	59

## Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
AHB-L	Advanced High-Performance Bus Lite
APB	Advanced Peripheral Bus
AXI4	Advanced eXtensible Interface 4
BL	Burst Length
CA	Command and Address
CS	Chip Select
CBT	Command Bus Training
DBI	Data Bus Inversion
DDR	Double Data Rate
DFI	DDR PHY
DM	Data Mask
DQ	Data
DQS	Data Strobe
ECC	Error Correction Code
ECLK	Edge Clock
FPGA	Field Programmable Gate Array
I/F	Interface
JEDEC	Joint Electron Device Engineering Council
JTAG	Joint Test Action Group
LPDDR4	Low Power Double Data Rate Generation 4
LVSTL	Low Voltage Swing Terminated Logic
MC	Memory Controller
MR	Mode Register
MRS	Mode Register Set
ODT	On-Die Termination
PRBS	Pseudorandom Binary Sequence
PVT	Process, Voltage, and Temperature
RTL	Register Transfer Level
SCLK	System Clock
SDR	Single Data Rate
SDRAM	Synchronous Dynamic Random Access Memory
SSN	Simultaneous Switching Noise
TCL	Tool Command Language
VREF	Voltage Reference

# 1. Introduction

The Lattice Semiconductor LPDDR4 Memory Controller for Nexus Devices provides a turnkey solution consisting of a controller, DDR PHY, and associated clocking and training logic to interface with LPDDR4 SDRAM. The IP Core is implemented in System Verilog HDL using the Lattice Radiant™ software integrated with the Lattice Synthesis Engine (LSE) and Synplify Pro® synthesis tools. The LPDDR4 Memory Controller simplifies the interfacing of CertusPro™-NX and MachXO5™-NX devices with external LPDDR4 memory for user applications.

## 1.1. Quick Facts

The following table presents a summary of the LPDDR4 Memory Controller for Nexus Devices.

**Table 1.1. Quick Facts**

<b>IP Requirements</b>	Supported FPGA Family	LPDDR4 mode – CertusPro-NX LPDDR4 mode – MachXO5T-NX
<b>Resource Utilization</b>	Targeted Device	LFCPNX-100 and LFCPNX-50 (IP Core v1.x.x and v2.x.x) LFMXO5-55T and LFMXO5-100T (in IP Core v2.1.x)
	Supported User Interfaces	AHB-L for data access in IP Core v1.x.x only AXI4 for data access in IP Core v2.x.x only APB for configuration access in IP Core v1.x.x and v2.x.x
	Resources	Refer to <a href="#">Table A.1</a> and <a href="#">Table A.2</a>
<b>Design Tool Support</b>	Lattice Implementation	Lattice Radiant software 2023.1
	Synthesis	Lattice Synthesis Engine Synopsys® Synplify Pro for Lattice
	Simulation	For a list of supported simulators, refer to the <a href="#">Lattice Radiant Software User Guide</a>

## 1.2. Features

The LPDDR4 Memory Controller for Nexus Devices supports the following key features:

- LPDDR4 SDRAM protocol, compliant to [LPDDR4 JEDEC Standard](#)
  - LPDDR4 SDRAM speeds of:
    - 300 MHz (600 Mbps)
    - 350 MHz (700 Mbps)
    - 400 MHz (800 Mbps)
    - 533 MHz (1066 Mbps)
- LPDDR4 Memory Controller features:
  - Component support for interface data widths of x16, x32, and x64
  - Up to 16 Gb per channel density support
  - x16 LPDDR4 device support (8:1 DQ:DQS ratio)
  - Burst length of BL16 and BL32, including On-The-Fly (OTF)
  - 8:1 gearing mode (LPDDR4:FPGA logic interface clock ratio)
  - Read DBI support only
  - Configurable CAS latencies for Reads and Writes based on target interface speed
  - Configurable Address widths to support various memory densities
- AHB-Lite data interface support (in IP Core v1.x.x only):
  - SINGLE, INCR1, and INCR8 for read/write operations
  - Aligned addressing only



- AXI4 data interface support (in IP Core v2.x.x only):
  - INCR with AxLEN = 0-63 for read/write operations (in IP Core v2.1.x)
  - Unaligned transfer using byte strobes (in IP Core v2.1.x)
  - Narrow transfers (in IP Core v2.1.x)
  - Narrow AXI widths of 32, 64, and 128
  - Aligned addressing to AxSIZE only
- APB configuration interface support (in IP Core v1.x.x and v2.x.x):
  - Automatic LPDDR4 SDRAM initialization
  - Dynamic valid window optimization for Read/Write paths
  - DQ-DQS skew optimization for Write training
- Periodic training support featuring:
  - Temperature tracking
  - Adaptive/Derate refresh rate for extended temperature support
  - ZQ calibration (ZQCAL START and LATCH only)
- Automatic detection of idle triggering Self-Refresh with Power-down entry
- Polling and Out-of-band interrupt support for error and extended temperature support

**Table 1.2. Features Overview**

Key Features	LPDDR4 Support Details
Device Format	Component
Data Widths	x16, x32, x64
Data User Interface	AHB-L, AXI4
Configuration Interface	APB
Maximum Command Rate	533 Mbps
Maximum Data Rate	1066 Mbps
<b>HW Managed Periodic Events</b>	
Refresh	All bank auto refresh
ZQ Calibration	Yes for ZQ start and latch, No for ZQ reset
Low Power Features	Self-refresh with power-down
<b>Other Features<sup>1</sup></b>	
Error Correction Code (ECC)	Not yet supported
Dual-rank	No
Data Bus Inversion (DBI)	Yes for reads, No for writes
Temperature Tracking	Yes
Refresh Adaptation (derate) to Temperature Variation	Yes
On-Die Termination (ODT)	Yes for DQ, No for CA
<b>Training<sup>1</sup></b>	
Initialization	Yes
Command Training	Yes
Write Leveling	Yes
Read Training	Yes
Write Training	Yes
VREF Training	Not yet supported

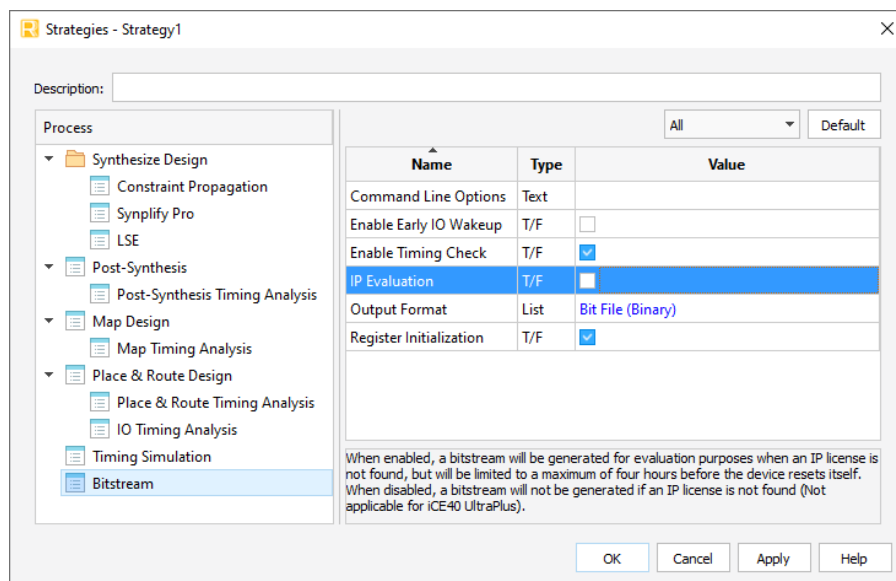
Notes:

1. **Yes** implies that a configurable option exists to enable or disable the feature. **No** implies that the feature is currently not supported and will not be supported in the future.

### 1.3. Licensing and Ordering Information

The LPDDR4 Memory Controller for Nexus Devices supports Lattice's IP evaluation capability for supported FPGA families and target devices. A bitstream can be generated for evaluation purposes without the purchase of an IP license, allowing hardware operation for a limited period of time (maximum of four hours), before the device resets itself. The IP evaluation setting is disabled by default and can be enabled/disabled within the Lattice Radiant software by performing the following steps:

1. Launch the Lattice Radiant software and select Project > Active Strategy > Bitstream Settings. This will open the Strategies dialog box.
2. Enable bitstream generation for IP evaluation purposes by setting IP Evaluation to True (checked) or disable this feature by setting IP Evaluation to False (unchecked).



**Figure 1.1. Enabling Bitstream for IP Evaluation**

An IP specific license string is required to enable full use of the LPDDR4 Memory Controller for Nexus IP in a complete design. For more information about pricing and availability of the LPDDR4 Memory Controller for Nexus IP, contact your local [Lattice Sales Office](#).

**Table 1.3. Ordering Part Number**

Device Family	Part Number	
	Single Machine Annual	Multi-site Perpetual
CertusPro-NX	LPDDR4-MC-CPNX-US	LPDDR4-MC-CPNX-UT
MachXO5-NX	LPDDR4-MC-XO5-US	LPDDR4-MC-XO5-UT

## 1.4. IP Validation Summary

The LPDDR4 Memory Controller for Nexus Devices supports both CertusPro-NX and MachXO5T-NX devices. The following table summarizes the compilation, simulation, and hardware validation for the LPDDR4 Memory Controller IP.

**Table 1.4. IP Validation Summary**

Device/Mode	Compilation	Simulation	Hardware
CertusPro-NX (LPDDR4 mode)	Yes	Yes	Yes <sup>1</sup>
MachXO5T-NX (LPDDR4 mode)	Yes	Yes	No

Note:

- For x16 and x32 data widths only. Data width of x64 is not HW validated.

## 1.5. Minimum Device Requirements

The LPDDR4 Memory Controller for Nexus Devices supports both CertusPro-NX and MachXO5T-NX devices. The following table summarizes the minimum device requirements for the Memory Controller IP Core.

**Table 1.5. Minimum Device Requirements**

LPDDR4 Interface Speed	LPDDR4 Data Width	Supported Speed Grades
300 MHz (600 Mbps)	x16, x32, x64	7, 8, 9
350 MHz (700 Mbps)	x16, x32, x64	7, 8, 9
400 MHz (800 Mbps)	x16, x32, x64	7, 8, 9
533 MHz (1066 Mbps)	x16, x32, x64	8, 9

## 1.6. Naming Conventions

This section provides information regarding terminology used within this document.

### 1.6.1. Nomenclature

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

### 1.6.2. Signal Names

Signal Names that end with:

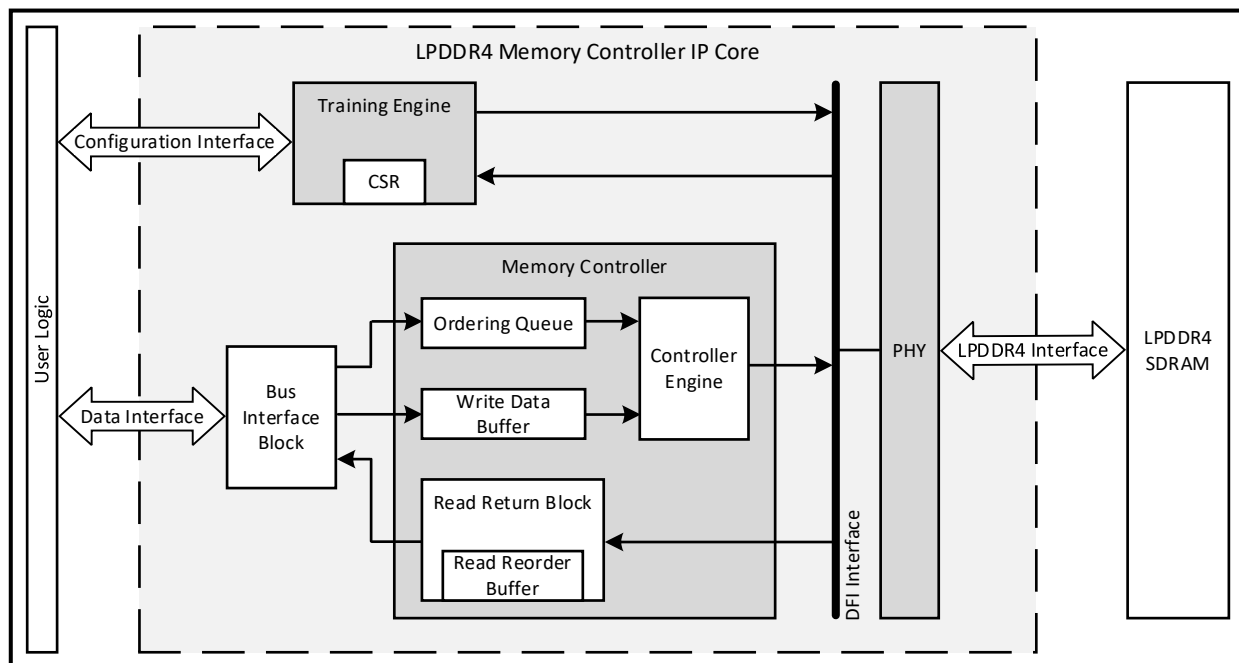
- `_n` are active low
- `_i` are input signals
- `_o` are output signals
- `_io` are bi-directional input/output signals

## 2. Functional Description

This section provides a detailed functional description of the LPDDR4 Memory Controller for Nexus Devices. Including information regarding clock and reset handling, available user data and configuration interfaces, the LPDDR4 calibration sequence, and LPDDR4 operation descriptions.

### 2.1. IP Architecture

The LPDDR4 Memory Controller for Nexus Devices consists of three main blocks: the Memory Controller, PHY, and Training Engine. The following figure represents the LPDDR4 Memory Controller submodules and its connectivity.



**Figure 2.1. Memory Controller IP Core Functional Diagram**

The data interface allows users to initiate LPDDR4 command/address/control and read/write operations to the external LPDDR4 SDRAM. The configuration interface provides access to the Training Engine and the Configuration Set Registers (CSRs), which configure the Memory Controller and perform the LPDDR4 training sequences. The LPDDR4 interface allows the selected Lattice FPGA to communicate with the external LPDDR4 memory. For more information on the data and configuration interfaces, refer to the [User Interfaces](#) section of this User Guide.

#### 2.1.1. Soft Memory Controller

The Soft Memory Controller consists of the following submodules:

- Bus Interface Block
- Controller Engine
- Read Return Block

##### 2.1.1.1. Bus Interface Block

The Bus Interface Block is responsible for accepting user-initiated operations on the selected data interface and translating them for processing within the Memory Controller. The data provided to the Controller Engine consists of:

- Address interface size: determined based on the selected LPDDR4 SDRAM device density and LPDDR4 data width
- Read and Write interface sizes: determined based on the selected LPDDR4 data width

The ordering queue is responsible for queuing the address, command, and control information coming from the Bus Interface Block. It prioritizes the execution of commands targeting an already opened bank and maintains correct order of execution when multiple requests target the same bank in LPDDR4 SDRAM. The write data buffer is responsible for buffering the incoming data for write commands.

#### 2.1.1.2. Controller Engine

The Controller Engine accepts the received requests from the Bus Interface and is responsible for translating the address to row, column, and bank format. The Controller Engine supports outstanding writes and reads, which is a write/read request that is entered into a queue and arbitrated for processing. The following table describes the three different types of requests and how they are processed.

**Table 2.1. MCE Request Handling**

Request type	Prioritization and Definition	Comments
Read	Varies depending on selected data interface protocol. Refer to the <a href="#">Data Interface Protocols</a> section of this User Guide for details.	The read request size is equal to the supported LPDDR4 burst length.
Write	Write requests are defined as a write, where the entire data needs to be written for the supported LPDDR4 burst length.	Example: For a 32-bit LPDDR4 data bus width, with burst length 16 (BL16), a full write would be $16 \times 4B = 64B$ .
Partial write	Partial write requests are defined as a write, where the entire data <i>is not</i> written.	Example: For a 32-bit LPDDR4 data bus width, a partial write would be a write request that is less than 32B. This is equivalent to a masked write.

All write requests may be processed out of order if a data dependency is not present. For example, an incoming request tied to an already opened page in LPDDR4 SDRAM can be prioritized if it is not dependent on the requests already in the queue. Read requests may also be processed out of order if a data dependency is not present depending on the selected data interface protocol. Refer to the [Data Interface Protocols](#) section of this User Guide for information regarding outstanding write/read support.

The Controller Engine contains bank management logic to track all the opened and closed pages within each bank, along with timers for each bank and rank. This allows issued memory commands to meet the operational sequence and timing requirements of the LPDDR4 SDRAM. The last layer of logic within the Controller Engine handles handshaking between the PHY and external LPDDR4 memory. The Controller Engine is also responsible for supporting periodic events compliant with the LPDDR4 SDRAM requirements, such as temperature tracking, adaptive/derate refresh rates, and ZQ calibration. For more information on supported periodic features, refer to the [LPDDR4 Operation Description](#) section of this User Guide.

#### 2.1.1.3. Read Return Block

The Read Return Block is responsible for responding to the Controller Engine issued read requests. Execution of commands can be out of order, so the Read Return Block handles the reordering to ensure that the read data is sent to the Bus Interface in order. The Read Return Block also handles Data Bus Inversion (DBI), which is an I/O signaling technique that reduces power consumption and improves signal integrity.

### 2.1.2. Soft PHY

The soft PHY is comprised of logic in the FPGA fabric to convert SDR to DDR, and vice-versa. The DFI interface operates in SDR mode, whereas the LPDDR4 interface operates in DDR mode. [Figure 2.2](#) represents a high-level block diagram of the Soft PHY and its major submodules. The Training Engine and Memory Controller are able to access both the Command/Address path and Data Input/Output paths.

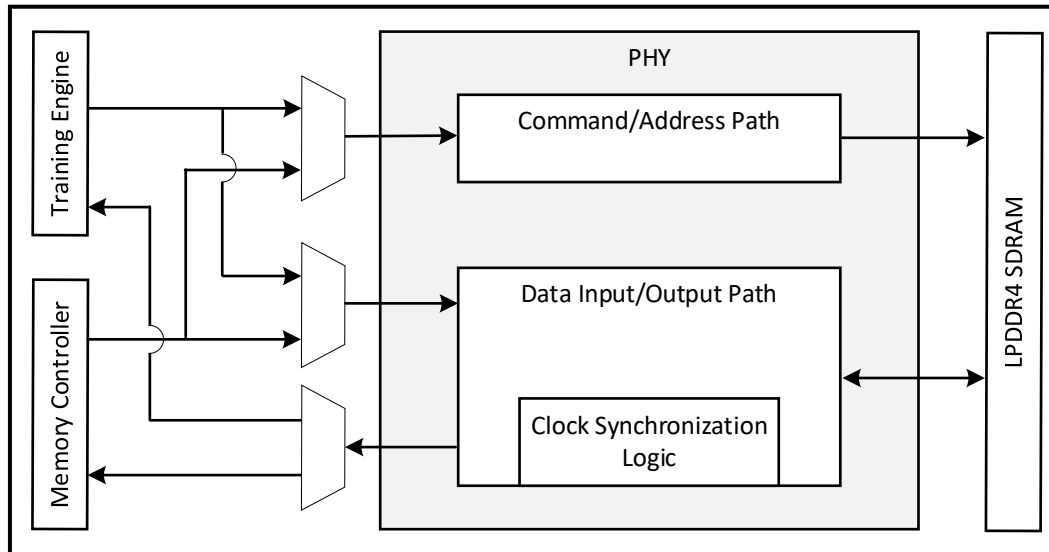


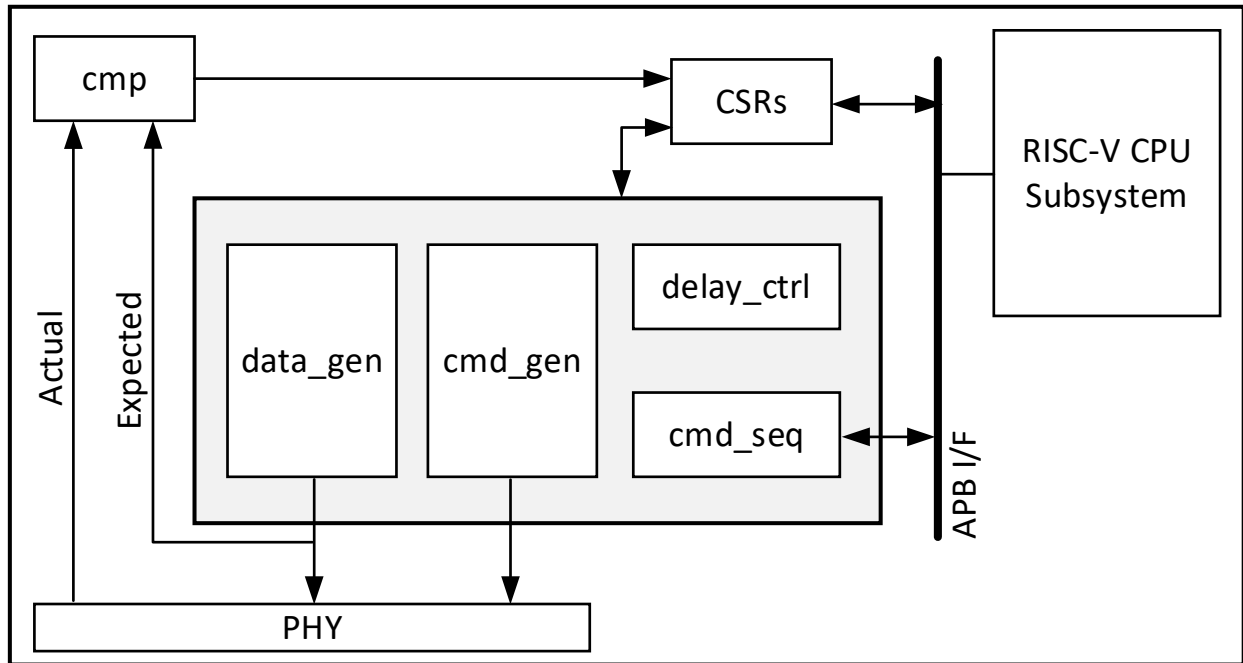
Figure 2.2. Lattice PHY

The PHY utilizes hardened logic, called I/O modules or hardware primitives, to implement clock synchronization logic and the command/address and data input/output paths. The clock synchronization logic handles Clock Domain Crossing (CDC) between the Edge Clock (ECLK) and System Clock (SCLK). ECLK is an internal clock used to clock the DDR primitives and SCLK is used to clock the Memory Controller IP Core. Without proper synchronization, the bit order on individual elements might become desynchronized, causing the data bus to become scrambled.

The LPDDR4 Memory Controller operates in 8:1 gearing mode, which means that ECLK operates at the same frequency as the LPDDR4 interface and that SCLK operates at a quarter of the LPDDR4 memory clock. In other words, for a 400 MHz LPDDR4 interface, ECLK operates at 400 MHz and SCLK operates at 100 MHz. This clocking ratio and DDR transfer relationship means that the user data bus is eight times the width of the LPDDR4 memory data bus. For example, a 32-bit LPDDR4 memory interface would require a 256-bit write and read data bus on the DDR PHY (DFI) side, which is an interface protocol between the Memory Controller and PHY. For more information on the soft PHY, hardware primitives, and clock synchronization logic, refer to the [LPDDR4 Memory Interface Module User Guide](#).

### 2.1.3. Soft Training Engine

The Training Engine is responsible for initializing and training the external LPDDR4 memory. It provides a configuration interface from user logic to the Configuration Set Registers (CSR). Users can issue commands to the Training Engine to perform reads and writes to these registers, allowing users to set desired programmable options, handle interrupts, and obtain details during the LPDDR4 memory training sequence. It consists of command and data generators, a comparator, user-accessible registers, delay logic, a command sequencer, and a RISC-V CPU subsystem.



**Figure 2.3. Training Engine**

The command generator (cmd\_gen) is responsible for generating LPDDR4 commands for all training procedures. The data generator (data\_gen) is a pseudo-random (PRBS) number generator that generates the expected data for read training. During write training, it generates both the write data and expected read data. The compare (cmp) block is responsible for comparing the actual read data with the expected read data and sending the result to the registers within the CSR block. The command sequencer (cmd\_seq) is responsible for dictating the command and data generation during the various training stages. The delay control (delay\_ctrl) block adjusts the delay of the target LPDDR4 memory signals during training. The CSR block consists of various user-accessible registers outlined in the [Register Description](#) section of this User Guide.

The RISC-V subsystem writes to the CSRs and command sequencer block in order to initialize and train the LPDDR4 interface. It is also responsible for programming the pattern and seed in the command and data generator blocks prior to the execution of the training procedure.

## 2.2. Clocking and Reset

The LPDDR4 Memory Controller IP requires a 100 MHz LVSTL input reference clock: pll\_rst\_n\_i. Users need to provide this clock via an external source and route it to an appropriate pin on the FPGA (externally sourced). The provided input clock is then routed through a dedicated PLL within the Memory Controller IP to generate the Edge Clock (ECLK) and System Clock (SCLK). The ECLK signal is used internally to the IP Core to clock the I/O modules and SCLK is used to clock the IP Core. The Primary Clock (PCLK) signal is used to implement clock synchronization logic to synchronize SCLK and ECLK. For additional details regarding clock synchronization, refer to the [LPDDR4 Memory Interface Module](#) User Guide.

The LPDDR4 Memory Controller IP contains an asynchronous active low reset: rst\_n\_i. When asserted, the Memory Controller and LPDDR4 SDRAM are reset to their default values. The IP Core contains internal logic to synchronously de-assert the internal reset once rst\_n\_i is de-asserted, so users do not need to worry about implementing their own deassertion logic.

The configuration interface for the LPDDR4 Memory Controller operates off of the pclk\_i signal and is reset via the preset\_n\_i signal. The clock for the configuration interface is the same one used to implement clock synchronization for SCLK and ECLK. The preset\_n\_i signal is an asynchronous active low reset, where users must ensure it is synchronously de-asserted to pclk\_i. Refer to the [Data Interface Protocols](#) section of this User Guide for information regarding the clocks and resets for the LPDDR4 Memory Controller supported data interfaces.

## 2.3. User Interfaces

This section describes the supported protocols for data and configuration interfaces available to the user and supported by the LPDDR4 Memory Controller.

### 2.3.1. Data Interface Protocols

The data interface allows users to initiate read and write operations to external LPDDR4 SDRAM. The LPDDR4 Memory Controller supports two protocols for data interfacing: AHB-Lite or AXI4.

#### 2.3.1.1. AHB-Lite

The AHB-Lite I/F is available in IP Core v1.x.x and is a single channel bus intended for high-performance and high-frequency applications. The AHB-Lite I/F operates off of the `sclk_o` signal and is reset via the `rst_n_i` signal. Refer to the [Clocking and Reset](#) section of this User Guide for more details.

The Memory Controller IP supports the following AHB-Lite types of burst transfers:

- Burst Type (HBURST):
  - SINGLE: single burst (1 beat)
  - INCR: incrementing burst that does not wrap at address boundaries (4 or 8 beats)
- Burst Size (HSIZE):
  - [8, 16, 32, 64, 128, 256] Bytes
- Burst Address (HADDR):
  - Unaligned addressing is not supported so all transfers must be aligned to the address boundary
  - SINGLE: address should be aligned to HSIZE
  - INCR4: address should be aligned to  $(\text{BUS\_WIDTH} / 8) \times 16$
  - INCR8: address should be aligned to  $(\text{BUS\_WIDTH} / 8) \times 32$

The AHB-Lite protocol can only support a single outstanding transaction per bus master. As a result, when AHB-Lite is configured as the user data interface, the Memory Controller can support up to 4 outstanding writes and only 1 outstanding read, where the next request cannot be started until the current one completes. For AHB-Lite implementation, reads are prioritized over writes unless there is a dependent write to be serviced. For more information regarding the AHB-Lite protocol, refer to the [AMBA AHB Protocol Specification](#).

**Table 2.2. Supported AHB-Lite Transactions**

Transaction Type	HBURST	HSIZE[2:0]	Comment
Write	SINGLE	[0, 1, 2, 3, 4, 5] <sup>1</sup>	[1-32] Byte write
Write	INCR4	[0, 1, 2, 3, 4, 5] <sup>1</sup>	[4-128] Byte write
Write	INCR8	[0, 1, 2, 3, 4, 5] <sup>1</sup>	[8-256] Byte write
Read	SINGLE	[0, 1, 2, 3, 4, 5] <sup>1</sup>	[1-32] Byte read
Read	INCR4	[3, 4, 5] <sup>1</sup>	[32-128] Byte read
Read	INCR8	[3, 4, 5] <sup>1</sup>	[64-256] Byte read

Notes:

1. For HSIZE=3, only BUS\_WIDTH=16 is supported. For HSIZE=4, only BUS\_WIDTH=32 is supported. For HSIZE=5, only BUS\_WIDTH=64 is supported.

#### 2.3.1.2. AXI4

The AXI4 I/F is available from IP Core v2.0.0 and is intended for high-bandwidth and low-latency operation. The AXI4 I/F is a multi-channel bus consisting of 5 independent channels: write address, read address, write data, read data, and write response channels (read response is sent along with the read data). The AXI4 I/F operates off of the `ack_i` signal when the *Enable Local Bus Clock* attribute is checked. The AXI4 I/F is reset via the `areset_n_i` signal, which is an asynchronous active low reset, where users must ensure it is synchronously de-asserted to `ack_i`.



The Memory Controller IP supports the following AXI4 types of burst transfers:

- Burst Type (AxBURST):
  - INCR: incrementing burst that does not wrap at address boundaries
- Burst Size (AxSIZE):
  - [1, 2, 3, 4, 8, 16, 32, 64] Bytes
- Burst Length (AxLEN):
  - 1-64 beat burst (65-256 is not yet supported)
- Burst Address (AxADDR):
  - Unaligned addressing is not supported so all transfers must be aligned to AxSIZE
  - Unaligned transfer is supported using byte strobes
  - For maximum burst length of 64, address is required to be aligned to  $(DDR\_WIDTH / 8) \times 32$
- Write Strobes (WSTRB):
  - Only the first and last beats of a burst can have incomplete byte strobes (WSTRB cannot go low in middle of burst)

The AXI4 protocol supports out of order transaction completion and contains support for multiple outstanding transactions per bus master. As a result, when AXI4 is configured as the user data interface, the Memory Controller can support up to 8 outstanding writes and 8 outstanding reads, where both reads and writes are of the same priority. For more information regarding the AXI4 protocol, refer to the [AMBA AXI Protocol Specification](#).

**Table 2.3. Supported AXI4 Transactions**

Transaction Type	AxBURST	AxLEN[3:0]	AxSIZE[2:0]	Comment
Write	INCR	[0-63]	[0, 1, 2, 3, 4, 5, 6] <sup>1</sup>	[1-4096] Byte write
Read	INCR	[0-63]	[0, 1, 2, 3, 4, 5, 6] <sup>1</sup>	[1-4096] Byte read

Notes:

1. For AxSIZE=4, only DDR\_WIDTH=16 is supported. For AxSIZE=5, only DDR\_WIDTH=32 is supported. For AxSIZE=6, only DDR\_WIDTH=64 is supported.

## 2.3.2. Configuration Interface Protocol

The configuration interface allows users to initialize and train the LPDDR4 memory interface. The Memory Controller IP Core utilizes the APB protocol for its configuration interface.

### 2.3.2.1. APB

The APB I/F is available in both IP Core v1.x.x and v2.x.x and is a low-power protocol intended for accessing programmable control registers. It is not pipelined and is a synchronous protocol with a single address bus and two data busses: write and read. For more information regarding the APB protocol, please refer to the [AMBA APB Protocol Specification](#).

The Memory Controller leverages this protocol to initialize and train the interface between FPGA logic and LPDDR4 SDRAM. The APB I/F operates off of the `pclk_i` signal and is reset via the `preset_n_i` signal. Refer to the [Clocking and Reset](#) section of this User Guide for more details.

The APB I/F is not available when the *Enable APB I/F* attribute is unchecked. For details on how to initialize and train LPDDR4 SDRAM without the APB I/F, refer to the [Initialization and Training without APB Interface](#) section of this User Guide.

## 2.4. LPDDR4 Calibration

To ensure proper device functionality, the external LPDDR4 memory must be initialized and trained before the Memory Controller can perform data accesses. A detailed explanation of the initialization and training sequence is outlined in the [LPDDR4 JEDEC Standard](#).

Upon device power-up, the soft RISC-V CPU located inside the Training Engine is held in reset. To start the initialization and training sequence of the LPDDR4 SDRAM device, the user should execute the following steps via the configuration interface:

- Enable initialization and training by writing 8'h1F to the Training Operation Register (TRN\_OP\_REG). This will enable initialization, command bus training, write leveling, read training, and write training sequences to execute. For simulation purposes, it is recommended to shorten the initialization and training sequences by writing 8'h00 to TRN\_OP\_REG. For more information, refer to the [Register Description](#) section of this User Guide.
- Pull the CPU and Training Engine out of reset by writing 2'h3 to the Reset Register (RESET\_REG). This begins the initialization and training sequence.
- Wait until initialization and training completes using one of the following methods:
  - Poll the Status Register (STATUS\_REG) until the write\_trn\_done signal is asserted (STATUS\_REG[4]=1). This indicates that write training has completed, which is the final stage in the initialization and training process.
  - Wait for the trn\_done\_int signal (INT\_STATUS\_REG[0]=1) or the trn\_error\_int signal (INT\_STATUS\_REG[1]=1) to assert in the Interrupt Status Register (INT\_STATUS\_REG). This method requires the trn\_done\_en signal (INT\_ENABLE\_REG[0]=1) and the trn\_err\_en signal (INT\_ENABLE\_REG[1]=1) to be asserted in the Interrupt Enable Register (INT\_ENABLE\_REG)

After the above steps complete, the control of the PHY is transferred to the Memory Controller and the user can now start accessing the LPDDR4 memory through the data interface. Once init\_done\_o asserts, the RISC-V CPU and Training Engine enter reset to save power. Refer to the [Initialization and Training Sequence](#) section of this User Guide for more details regarding the LPDDR4 calibration sequence.

### 2.4.1. Initialization and Training Sequence

The LPDDR4 Memory Controller IP performs initialization according to the LPDDR4 JEDEC Standard. This section describes the steps that occur during the LPDDR4 initialization and training sequence.

Once the CPU and Controller Engine is pulled out of reset, the ddr\_reset\_n\_o signal is set low for a period of ~200  $\mu$ s. This is followed by the ddr\_cke\_o signal asserting high for ~2 ms before the LPDDR4 clock (ddr\_ck\_o) activates and begins toggling. These steps take a very long time to simulate, which is why it is recommended to set the init\_en signal low (TRN\_OP\_REG[0]=0) to shorten the ddr\_reset\_n\_o and ddr\_cke\_o assert times to a few clock cycles. Refer to the Simulation Example Design section of this User Guide for simulation runtimes of different LPDDR4 configurations.

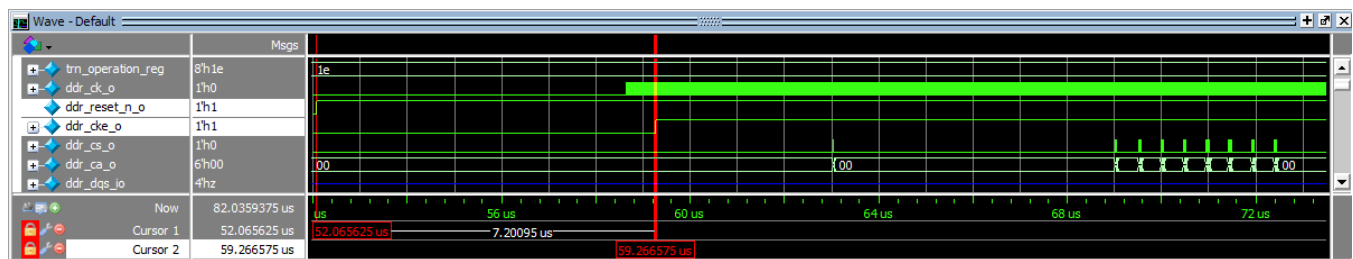
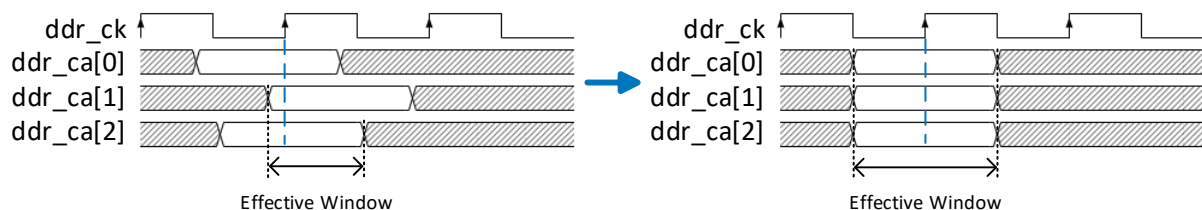


Figure 2.4. Shortened Initialization Sequence Simulation Waveform (TRN\_OP\_REG[0]=0)

#### 2.4.1.1. Command Bus Training (CBT)

The goal of Command and Address (CA) calibration is to delay the command and address signals as necessary to optimize the CA window. When setting the cbt\_en signal high (TRN\_OP\_REG[1]=1), the Memory Controller performs Command Bus Training (CBT) according to the [LPDDR4 JEDEC Standard](#). This centers the entire CA bus relative to ddr\_ck\_o by aligning the rising edge of CK to the middle of the CA valid window. The LPDDR4 memory then provides feedback to the user through the DQ line, which is used by the Memory Controller to determine if additional adjustment is required.

During this time, the DDR clock, `ddr_ck_o`, will stop before and after CBT. This is expected as the clock is switching between low frequency (50 MHz) and high frequency (LPDDR4 interface speed) operation, as outlined in the [LPDDR4 JEDEC Standard](#). Upon successful completion of CBT, `ddr_ca_o` should be centered to the eye of `ddr_ck_o`, ensuring correct behavior during high frequency operation. Refer to the [Command Bus Training Debug](#) section of this User Guide for additional details regarding how CBT is performed by the LPDDR4 Memory Controller.

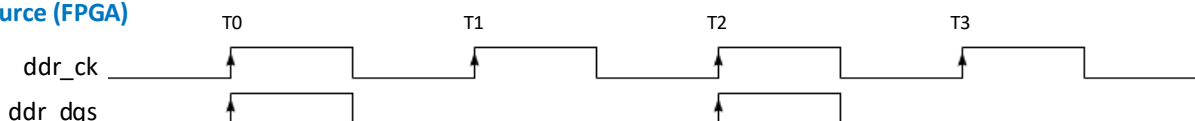


**Figure 2.5. Command Bus Training**

#### 2.4.1.2. Write Leveling

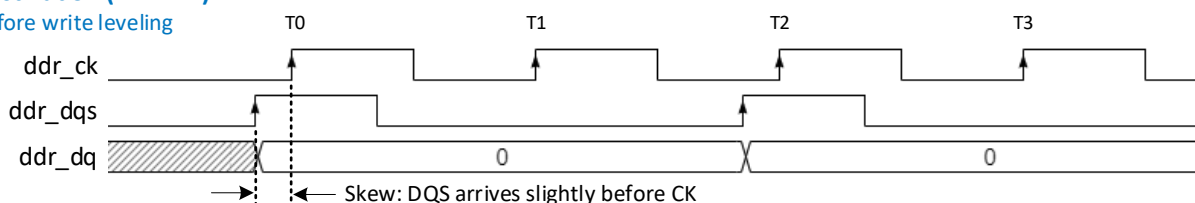
The purpose of write leveling is to delay each Data Strobe (DQS) relative to the DDR clock during write operations so they are edge-aligned. This addresses the CK to DQS timing skew that is introduced with the adoption of Fly-By Topology, which reduces Simultaneous Switching Noise (SSN) by allowing different memory components to receive write commands at different times. Setting the `write_lvl_en` signal high (`TRN_OP_REG[2]=1`), enables the Memory Controller to perform write leveling on all available ranks. During this process, the Memory Controller delays the `ddr_dqs_o` signal until a 0-to-1 transition on `ddr_ck_o` is captured by the `dqs` rising edge of the LPDDR4 SDRAM device. The LPDDR4 memory then provides feedback of the captured clock signal value through the `DQ` line, which is used by the Memory Controller to determine if additional adjustment is required. Refer to the [Write Leveling Debug](#) section of this User Guide for additional details regarding how write leveling is performed by the LPDDR4 Memory Controller.

##### Source (FPGA)



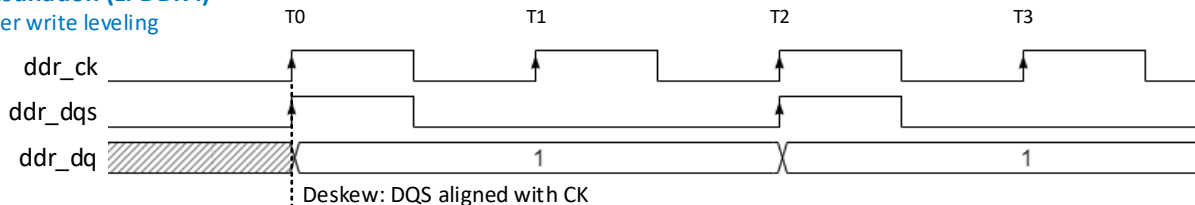
##### Destination (LPDDR4)

Before write leveling



##### Destination (LPDDR4)

After write leveling

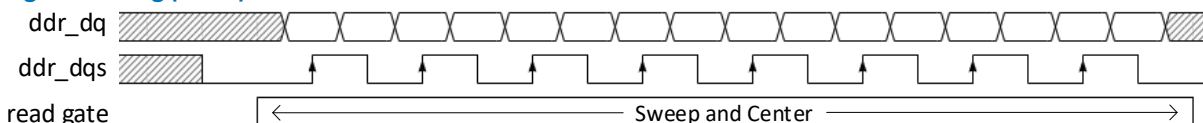


**Figure 2.6. Write Leveling**

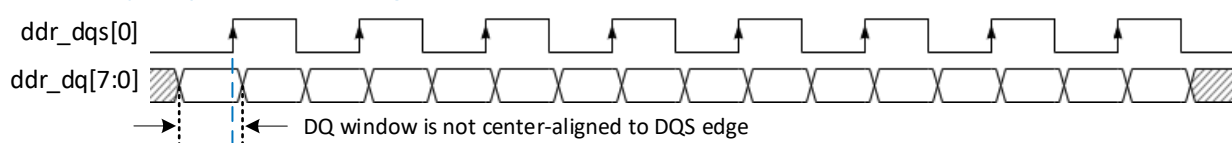
### 2.4.1.3. Read Training

The purpose of read training is to delay each DQS relative to the DDR clock during read operations to ensure a full burst can be captured correctly. Read Training also center-aligns DQS relative to the DQ window. Setting the read\_trn\_en signal high (TRN\_OP\_REG[3]=1), enables the Memory Controller to perform read training. During this process, the Memory Controller issues burst reads and delays the ddr\_dqs\_o signal until a full burst is captured correctly. In other words, the Memory Controller delays its read gate to ensure all data sent by the external LPDDR4 SDRAM is received optimally. Once the DQS is captured properly for the entire read burst, read per-byte deskew is performed to optimize the read DQ window to its associated DQS. Refer to the [Read Training Debug](#) section of this User Guide for additional details regarding how read training is performed by the LPDDR4 Memory Controller.

#### Read gate training (FPGA)



#### Before read per-byte deskew training (FPGA)



#### After read per-byte deskew training (FPGA)

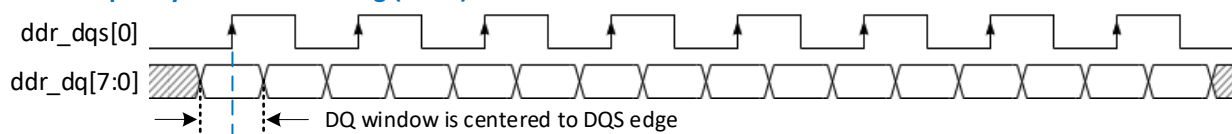
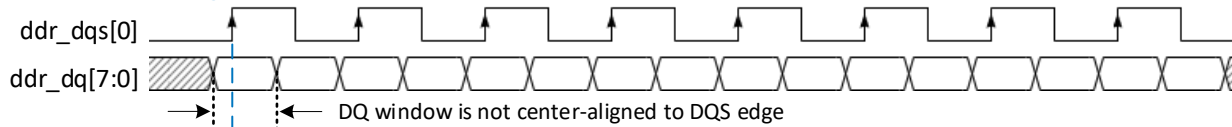


Figure 2.7. Read Training

### 2.4.1.4. Write Training

The purpose of write training is to delay each DQS relative to DQ during write operations to optimize the data window. Setting the write\_trn\_en signal high (TRN\_OP\_REG[4]=1), enables the Memory Controller to perform write training. This entails aligning the rising-edge of DQS to the center of the DQ valid window per byte on the LPDDR4 memory side. Refer to the Write Training Debug section of this User Guide for additional details regarding how write training is performed by the LPDDR4 Memory Controller.

#### Before write training (LPDDR4)



#### After write training (LPDDR4)

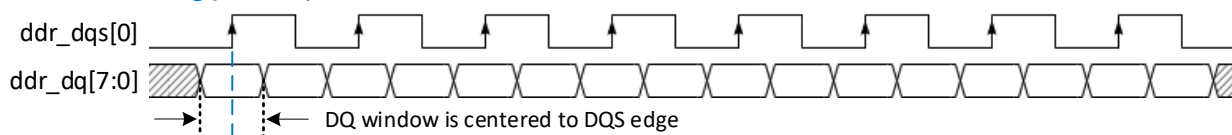


Figure 2.8. Write Training

### 2.4.2. Initialization and Training without APB Interface

The APB I/F will not be available when the *Enable APB I/F* attribute is unchecked. In this case, the `init_start_i` and `trn_opr_i` signals will be available for starting and configuring the LPDDR4 initialization and training. The `trn_opr_i` sets the value of the Training Operation Register (TRN\_OP\_REG), thus controlling which specific training steps to perform based on the asserted bits. For hardware implementation, it is recommended to set `trn_opr_i=0x1F` to perform: command bus training, write leveling, read training, and write training procedures. For simulation purposes, it is recommended to shorten the initialization and training sequences by setting `trn_opr_i=0x00`. Refer to the [Register Description](#) section of this User Guide for more details.

The user should execute the following steps to start the initialization and training sequence of the LPDDR4 SDRAM:

1. Set `init_start_i=0` while the Memory Controller IP is in reset or while `pll_lock_o=0`
2. Set `init_start_i=1` and maintain this value until `init_done_o` asserts
3. Once `init_done_o=1`, set `init_start_i=0` to hold the training CPU in reset to save power

## 2.5. LPDDR4 Operation Description

This section provides details on various operations and features supported by the LPDDR4 Memory Controller IP.

### 2.5.1. Write and Read Data Access

Once the `init_done_o` signal asserts, users can initiate write and read accesses. The types of data accesses supported by the Memory Controller IP depends on the selected data interface protocol. Refer to [Table 2.2](#) and [Table 2.3](#) for a list of all supported AHB-Lite and AXI4 data accesses.

The Memory Controller decodes the memory-mapped address to check if the row and bank addresses are already opened in the memory device. When a WRITE/READ command is issued to LPDDR4 memory the following commands are issued:

- If the target row and bank are not open, an ACTIVATE command is issued by the Memory Controller to the LPDDR4 SDRAM to open the row. This is followed by the WRITE/READ command.
- If there is an opened row in the current bank, and the target row address is different from the opened row, a PRECHARGE command is issued by the Memory Controller to close the opened row. This is followed by an ACTIVATE command to open the target row and then the WRITE/READ command.
- If the target row is already opened, only a WRITE/READ command is issued.

The Memory Controller does not immediately close a row after a WRITE/READ command, it issues a separate PRECHARGE command as needed.

### 2.5.2. Auto Refresh Support

Ideally, REFRESH commands should be issued every refresh interval, specified by the *Refresh Period* attribute. To improve efficiency in scheduling and switching between tasks, the LPDDR4 memory allows a maximum of 8 REFRESH commands to be postponed throughout the LPDDR4 operation. The Memory Controller has an internal auto refresh generator that sends out a set of consecutive AUTO REFRESH commands to the memory once when it reaches the time specified in the following calculation: *Refresh Period* × *No. of Outstanding Refresh*.

For high performance applications, it is recommended to set *No. of Outstanding Refresh* to the maximum value. This increases the LPDDR4 bus throughput by minimizing Memory Controller intervention.

### 2.5.3. Power Saving Feature

The Memory Controller supports power-saving when the *Enable Power Down* attribute is set. The Memory Controller IP tracks the period of inactivity on the local data bus by monitoring the System Clock (SCLK), which is the main clock used by the Memory Controller IP. When the period of inactivity on the bus reaches the value set in the *Number of SCLK to enter Self-Refresh from no traffic* attribute, the Memory Controller will issue SELF REFRESH and Power-Down Entry commands. This will put the memory into Power-Down mode to save power until a new request is received on the local data bus, at which the Memory Controller will issue SELF REFRESH and Power-Down Exit commands, followed by an additional REFRESH command.

#### 2.5.4. Periodic ZQ Calibration

ZQ Calibration calibrates the output driver impedance across PVT. There are two ZQ Calibration modes initiated with the Multi-Purpose Command (MPC): ZQCAL START and ZQCAL LATCH. The ZQCAL START command initiates the SDRAM's calibration procedure, and the ZQCAL LATCH command captures the result and loads it into the SDRAM's drivers.

The Memory Controller IP periodically performs ZQ Calibration as the voltage and temperature may fluctuate during operation. The user can configure the frequency at which ZQ Calibration is performed and its duration via the *ZQ Calibration Period* and *ZQ Calibration Start to Latch* attributes. When SETTINGS\_REG[16]=0, ZQ Calibration is set to Short, whereas when SETTINGS\_REG[16]=1, ZQ Calibration is set to Long. Refer to the [Register Description](#) section of this User Guide for more details.

#### 2.5.5. Temperature Tracking and Extended Temperature Support

LPDDR4 SDRAM devices support temperature tracking, where the device's refresh rate can change based on the operational temperature. When the memory device's temperature is within normal operating conditions, the rate is set to 1x refresh, which is the default case. As the temperature fluctuates below or above the default case, the Memory Controller will decrease or increase the frequency of refreshes respectively. The required refresh rate according to temperature is specified in MR4 within LPDDR4 SDRAM. When the Temperature Update Flag (TUF) is set in MR4, it indicates that the refresh rate has changed from the last read issued to MR4.

LPDDR4 SDRAM devices also support operation for extended temperature. When the operational temperature increases beyond normal operation conditions, the refresh frequency begins to increase. When the refresh rate in MR4 is set to 3'b110, timing derating is required for certain commands, slowing performance. For more information regarding temperature derating timing requirements, refer to the [LPDDR4 JEDEC Standard](#).

The Memory Controller IP reads MR4 periodically to track the temperature of the LPDDR4 memory device. Based on the refresh rate set in MR4, the Memory Controller IP issues REFRESH commands at the required frequency. Note that when the refresh rate in MR4 is set to either 3'b000 or 3'b111, the LPDDR4 memory may not work as expected due to exceeding of low/high temperature operating limits.

When the refresh rate is set to 3'b110 in MR4, the Memory Controller accommodates the temperature derating timing requirements when issuing commands to the memory. The user can change the frequency at which the memory's MR4 register is read through the *Temperature Check Period* attribute. When reading MR4, the Memory Controller IP sets STATUS\_REG[18:16] to the read refresh rate value, and INT\_STATUS\_REG[4]=1 when TUF is set in MR4. Refer to the [Register Description](#) section of this User Guide for more details.

## 3. IP Parameter Description

The configurable attributes of the LPDDR4 Memory Controller for Nexus Devices are shown and described in the following tables. These attributes can be configured through the IP Catalog's Module/IP Block Wizard of the Lattice Radiant software. Refer to the [Designing and Simulating the IP](#) section of this User Guide for information on how to configure and generate the Memory Controller IP.

### 3.1. General

The following section describes the parameters available in the General tab of the IP parameter editor.

**Table 3.1. General Attributes**

Attribute	Selectable Values	Default	Dependency on Other Attributes
DDR Interface Type	LPDDR4	LPDDR4	Display only
I/O Buffer Type	LVSTL_I, LVSTL_II	LVSTL_I	—
DDR Command Frequency (MHz) <sup>1</sup>	300, 350, 400, 533	533	—
Gearing Ratio	8:1	8:1	Display only
Enable ECC	Checked, Unchecked	Unchecked	Display only (ECC is not yet supported)
Enable Power Down	Checked, Unchecked	Unchecked	—
Enable DBI	Checked, Unchecked	Unchecked	—
Read Latency	Calculated	N/A	Display only Calculated based off selection for <i>DDR Command Frequency</i>
Write Latency	Calculated	N/A	Display only Calculated based off selection for <i>DDR Command Frequency</i>
Enable Internal RISC-V CPU	Checked, Unchecked	Checked	Display only (Disabling of Internal RISC-V CPU is not yet supported)

Notes:

- Only device speed grades of 8 and 9 can support DDR command frequencies up to 533MHz.

**Table 3.2. Clock Settings Attributes**

Attribute	Selectable Values	Default	Dependency on Other Attributes
Enable PLL	Checked	Checked	Display only
PLL Reference Clock from Pin	Checked, Unchecked	Checked	—
I/O Standard for Reference Clock	LVSTL_I, LVSTL_II	LVSTL_I	Display only Equal to selection for <i>I/O Buffer Type</i>
RefClock (MHz)	—	100	Display only
DDR Command Actual Frequency (MHz)	Calculated	N/A	Display only Based off selection for <i>DDR Command Frequency</i>



**Table 3.3. Memory Configuration Attributes**

Attribute	Selectable Values	Default	Dependency on Other Attributes
DDR Density (per Channel)	2 Gb, 4 Gb, 8 Gb, 16 Gb	4 Gb	—
DDR Bus Width(BUS_WIDTH)	16, 32, 64	32	—
Number of Ranks	1	1	Display only (Only single rank is supported)
Number of DDR Clocks (CK_WIDTH)	1	1	Display only
Number of Chip Selects (CS_WIDTH)	1	1	Display only

**Table 3.4. Local Data Bus Attributes**

Attribute	Selectable Values	Default	Dependency on Other Attributes
Local Data Bus Type	AHBL	AHB-Lite	For IP Core v1.x.x
	AXI4	AXI4	For IP Core v2.x.x
Address Width	Calculated	N/A	Display only Calculated based off selection for <i>DDR Density</i>
Data Width (AHBL_DATA_WIDTH or AXI_DATA_WIDTH)	Calculated	N/A	Display only Calculated based off selection for <i>Local Data Bus Type</i>
ID Width	2, 3, 4, 5, 6, 7, 8	4	For IP Core v2.x.x
Number of Outstanding Writes	1, 2, 3, 4	4	For IP Core v1.x.x
Write Ordering Queues	1, 2, 3, 4, 5, 6, 7, 8	4	For IP Core v2.x.x
Number of Outstanding Reads	1	1	Display only For IP Core v1.x.x
Read Ordering Queues	1, 2, 3, 4, 5, 6, 7, 8	4	For IP Core v2.x.x
Enable Local Bus Clock	Checked/Unchecked	Unchecked	Display only For IP Core v1.x.x
		Checked	For IP Core v2.x.x
Enable APB I/F	Checked/Unchecked	Checked	For IP Core v2.x.x

**Table 3.5. General Definitions**

Attribute	Description
<b>General Group</b>	
DDR Interface Type	Specifies the SDRAM Memory interface: LPDDR4
I/O Buffer Type	I/O Standard for the memory interface signals
DDR Command Frequency (MHz)	Speed at which the memory controller will issue commands to the memory device
Gearing Ratio	Specifies the ratio relationship between the DDR data speed and the memory controller speed
Enable ECC	Enables error-correction code (ECC) for single-bit error correct and double-bit error detection (not yet supported)
Enable Power Down	Enables memory controller automatically placing the memory device into power-down mode after a specified number of idle controller clock cycles
Enable DBI	Enables data bus inversion (DBI) for better signal integrity and read/write margins
Read Latency	Specifies the delay from issuing of a read command to receiving of read data from SDRAM
Write Latency	Specifies the delay from issuing of a write command to providing of write data to SDRAM
Enable Internal RISC-V CPU	Enables RISC-V subsystem to support initialization and training of the memory device (disabling is not yet supported)



Attribute	Description
<b>Clock Settings Group</b>	
Enable PLL	Enables PLL
PLL Reference Clock from Pin	Indicates if provided PLL reference clock originates from a pin
I/O Standard for Reference Clock	Specifies the I/O standard for the PLL reference clock
RefClock (MHz)	Indicates the PLL reference clock speed (100 MHz)
DDR Command Actual Frequency (MHz)	Specifies the actual operating frequency of the memory interface (calculated by the PLL – includes tolerance)
<b>Memory Configuration Group</b>	
DDR Density (per Channel)	Density of SDRAM (# of chips on memory module). Only DDR Density with power of 2 is supported.
DDR Bus Width	Total number of data pins in memory interface
Number of Ranks	Specifies number of ranks in memory interface (only single rank is supported)
Number of DDR Clocks	Specifies the number of CK/CK# clock pairs to be driven to SDRAM (multiple clocks is not yet supported)
Number of Chip Selects	Specifies the number of chip select (CS) signals to be driven to SDRAM (multiple chip selects is not yet supported)
<b>Local Data Bus Group</b>	
Local Data Bus Type	Indicates bus for local data interface: IP Core v1.x.x: AHB-Lite IP Core v2.x.x: AXI4
Address Width	Specifies number of address pins in memory interface, dependent on the SDRAM density.
Data Width	Indicates data width for local data bus: BUS_WIDTH × 4 if using AHB-Lite BUS_WIDTH × 8 if using AXI4
ID Width	Indicates bit width of AXI4 ID Only for IP Core v2.x.x
Number of Outstanding Writes / Write Ordering Queues	The number of outstanding writes/reads and the write/read ordering queues process write and read data requests. The higher the value, the better bandwidth on the interface since gaps between accesses is lessened at the cost of consuming more LUTs.
Number of Outstanding Reads / Read Ordering Queues	
Enable Local Bus Clock	Enables clock domain crossing logic for the local data bus. IP Core v1.x.x: disabled by default (fixed) IP Core v2.x.x: enabled by default (configurable)
Enable APB I/F	Enables APB interface. Should be enabled if target application includes a CPU. When disabled, initialization and training of SDRAM should be handled via init_start_i and trn_opr_i signals. Please refer to the <a href="#">Initialization and Training without APB Interface</a> section of this User Guide for more information.

## 3.2. Memory Device Timing

The following section describes the parameters available in the Memory Device Timing tab of the IP parameter editor. The default value for the attributes listed under Memory Device Timing Setting Group is different for each DDR Command Frequency value, and is based off what is specified in the JESD209-4C SDRAM standard. These values may need to be manually adjusted based on the selected LPDDR4 device.

**Table 3.6. Memory Device Timing Setting Attributes**

Attribute	Selectable Values	Default	Dependency on Other Attributes
Manual Timing Adjust Enable	Checked/Unchecked	Unchecked	—
TRCD (tCLK)	4–65536	Calculated	Enabled when <i>Manually Adjust</i> is Checked
TRAS (tCLK)	4–65536	Calculated	Enabled when <i>Manually Adjust</i> is Checked
TRPPB (tCLK)	4–65536	Calculated	Enabled when <i>Manually Adjust</i> is Checked
TWR (tCLK)	4–65536	Calculated	Enabled when <i>Manually Adjust</i> is Checked
TRTP (tCLK)	8–65536	Calculated	Enabled when <i>Manually Adjust</i> is Checked
TCCD (tCLK)	8–65536	Calculated	Enabled when <i>Manually Adjust</i> is Checked
MWR2MWR (tCLK)	32–65536	Calculated	Enabled when <i>Manually Adjust</i> is Checked
TRRD (tCLK)	6–65536	Calculated	Enabled when <i>Manually Adjust</i> is Checked
TRFC (tCLK)	24–28080	Calculated	Enabled when <i>Manually Adjust</i> is Checked
TFAW (tCLK)	40–65536	Calculated	Enabled when <i>Manually Adjust</i> is Checked
TZQCAL (tCLK)	4–65536	Calculated	Enabled when <i>Manually Adjust</i> is Checked
TMRR (tCLK)	8–65536	Calculated	Enabled when <i>Manually Adjust</i> is Checked
TMRD (tCLK)	10–65536	Calculated	Enabled when <i>Manually Adjust</i> is Checked
TRPAB (tCLK)	4–65536	Calculated	Enabled when <i>Manually Adjust</i> is Checked
TRTW (tCLK)	21–65536	Calculated	Enabled when <i>Manually Adjust</i> is Checked
TDQSS (tCLK)	Integer	1	Enabled when <i>Manually Adjust</i> is Checked
TRD2PRE (tCLK)	4–65536	Calculated	Enabled when <i>Manually Adjust</i> is Checked
TWR2PRE (tCLK)	4–65536	Calculated	Enabled when <i>Manually Adjust</i> is Checked
TXP (tCLK)	4–65536	Calculated	Enabled when <i>Manually Adjust</i> is Checked
TXPSR (tCLK)	4–65536	Calculated	Enabled when <i>Manually Adjust</i> is Checked

**Table 3.7. Periodic Event Setting Attributes**

Attribute	Selectable Values	Default	Dependency on Other Attributes
No of SCLK to enter Self-Refresh from no traffic	Integer	1000	Enabled when <i>Enable Power Down</i> is Checked
Refresh Period (tSCLK)	Calculated	Calculated	Default value is calculated based off selection for <i>DDR Command Frequency</i>
No. of Outstanding Refresh	1	1	Display only For IP Core v1.x.x
	1, 2, 3, 4, 5, 6, 7, 8	8	For IP Core v2.x.x
ZQ Calibration Period (sec)	Integer	32	—
ZQ Calibration Start to Latch (usec)	Integer	1	—
Temperature Check Period (sec)	Integer	32	—

**Table 3.8. Memory Device Timing Definitions**

Attribute	Description
<b>Memory Device Timing Setting Group<sup>1</sup></b>	
Manual Timing Adjust Enable	Enables user to manually set any of the memory timing parameters
TRCD	Indicates the delay between the ACTIVATE command (RAS) and the internal access to data (CAS)
TRAS	Indicates how long memory must wait after an ACTIVATE command before a PRECHARGE command can be issued to close the row
TRPPB	Row PRECHARGE time for a single bank

Attribute	Description
TWR	Specifies the amount of clock cycles needed to complete a WRITE before a PRECHARGE command can be issued
TRTP	Internal READ to PRECHARGE delay
TCCD	Minimum time between two READ/WRITE (CAS) commands (burst length / 2)
MWR2MWR	Masked WRITE to masked WRITE
TRRD	Minimum time interval between two ACTIVATE commands to different banks
TRFC	Indicates how long memory must wait after a REFRESH command before an ACTIVATE command can be accepted by memory
TFAW	Specifies the period duration during which only four banks can be active
TZQCAL	ZQ calibration time from START to LATCH command
TMRR	Mode register READ command period
TMRD	Minimum time between two MRS commands
TRPAB	Row PRECHARGE time for all banks
TRTW	READ to WRITE command delay
TDQSS	Describes skew between the output data strobe with respect to the memory clock for writes (DQS to CK)
TRD2PRE	READ to PRECHARGE time
TWR2PRE	WRITE to PRECHARGE time
TXP	Power-down exit latency to next valid command
TXPSR	Power-down exit latency to next self-refresh
<b>Periodic Event Setting Group</b>	
Number of SCLK to enter Self-Refresh from no traffic	The memory controller puts the memory in self-refresh when there is no traffic for the specified number of SCLK cycles
Refresh Period	Specifies the number of SCLK cycles between refresh commands
Number of Outstanding Refresh	Specifies the maximum number of outstanding refresh commands. Refer to the <a href="#">Auto Refresh Support</a> section of this User Guide for more information.
ZQ Calibration Period	Indicates period for performing ZQ Calibration in seconds
ZQ Calibration Start to Latch	Indicates time from start to latch during ZQ Calibration in microseconds
Temperature Check Period	Indicates period for reading the temperature register (MR4) in LPDDR4 memory in seconds

Notes:

- The memory device timing parameters listed under the Memory Device Timing tab are defined according to the JESD209-4C SDRAM standard. Refer to the memory device data sheet for detailed descriptions and allowed values for these parameters.

### 3.3. Training Settings

The following section describes the parameters available in the Training Settings tab of the IP parameter editor, which is only applicable for IP Core v2.x.x.

**Table 3.9. Training Settings Attributes**

Attribute	Selectable Values	Default	Dependency on Other Attributes
DDR Clock Delay Value	0-127	64	—
Initial MC DQ_VREF Value	0-127	70	—
Initial Memory CA_VREF Value	0-127	80	—
Initial Memory DQ_VREF Value	0-127	80	—
DQ_ODT Value	Disable, RZQ/1, RZQ/2, RZQ/3, RZQ/4, RZQ/5, RZQ/6	Disable	—

**Table 3.10. Trained Values Attributes**

Attributes	Selectable Values	Default	Dependency on Other Attributes
DQS0 Trained Write Leveling Delay	0-127	68	—
DQS1 Trained Write Leveling Delay	0-127	68	—
DQS2 Trained Write Leveling Delay	0-127	68	Visible only when <i>DDR Bus Width</i> >= 32
DQS3 Trained Write Leveling Delay	0-127	68	Visible only when <i>DDR Bus Width</i> >= 32
DQS<4,5,6,7> Trained Write Leveling Delay	0-127	68	Visible only when <i>DDR Bus Width</i> == 64
DQS0 Trained RDCLKSEL <sup>1</sup>	0-15	Calculated	—
DQS1 Trained RDCLKSEL <sup>1</sup>	0-15	Calculated	—
DQS2 Trained RDCLKSEL <sup>1</sup>	0-15	Calculated	Visible only when <i>DDR Bus Width</i> >= 32
DQS3 Trained RDCLKSEL <sup>1</sup>	0-15	Calculated	Visible only when <i>DDR Bus Width</i> >= 32
DQS<4,5,6,7> Trained RDCLKSEL <sup>1</sup>	0-15	Calculated	Visible only when <i>DDR Bus Width</i> == 64
Trained CS Delay	0-127	64	—
Trained CA Delay	0-127	64	—
Trained DQSBUF Read Delay	0-127	24	—
Trained DQSBUF Read Sign	0, 1	0	—
Trained Write DQ/DBI delay	0-127	56	—
Trained Write Latency	0-18	7	—
Trained Read Latency	0-40	12	—

Notes:

1. The default value of *DQS<0,1,2,3,4,5,6,7> Trained RDCLKSEL* is different for each *DDR Command Frequency* value. Each step adjusts the internally generated read DQS by a ~45° phase shift from the incoming DQS.

**Table 3.11. Training Settings Definitions**

Attribute	Description
<b>Training Settings Group<sup>1</sup></b>	
DDR Clock Delay Value	Specifies the DDR clock delay, so that the first DQS toggle is at CK = 0 during write leveling. It is recommended to increase this value if write leveling fails
Initial MC DQ_VREF Value	Initial DQ VREF value for the FPGA I/O. This value is translated to an internal reference voltage where the input DQ/DBI signal will be compared to determine a 0 and 1 value
Initial Memory CA_VREF Value	Initial CA VREF value that is written to MR12 in LPDDR4 memory
Initial Memory DQ_VREF Value	Initial DQ VREF value that is written to MR14 in LPDDR4 memory
DQ_ODT Value	Specifies the DQ ODT value that is written to MR11 in LPDDR4 memory
<b>Trained Values Group<sup>2</sup></b>	
DQS<0,1,2,3,4,5,6,7> Trained Write Leveling Delay	Specifies delay value to be programmed to the PHY when TRN_OP_REG[2] = 0. Each step adjusts the write DQS delay by ~12.5 ps.
DQS<0,1,2,3,4,5,6,7> Trained RDCLKSEL	Specifies the delay value to be programmed to the PHY when TRN_OP_REG[3] = 0. Each step adjusts the internal read DQS delay by a 45° phase shift from the incoming DQS.
Trained CS Delay	Specifies the CS delay. It is recommended to use the default value and to only adjust when CBT fails after already adjusting the <i>Initial MC DQ_VREF Value</i> . Each step adjusts the CS delay by ~12.5 ps.
Trained CA Delay	Specifies the delay value to be programmed to the PHY when TRN_OP_REG[1] = 0. Each step adjusts the CA delay by ~12.5 ps.
Trained DQSBUF Read Delay	Specifies the delay value to be programmed to the PHY when TRN_OP_REG[3] = 0. Each step adjusts the internal read DQS delay by a small delay based on the DDR clock frequency.
Trained Write DQ/DBI Delay	Specifies the delay value to be programmed to the PHY when TRN_OP_REG[4] = 0. Each step adjusts the write DQ/DBI delay by ~12.5 ps.

Attribute	Description
Trained Write Latency	Specifies the write latency setting of the PHY that passes write training. Ideally, this is equal to <i>Write Latency</i> + 1.
Trained Read Latency	Specifies the read latency setting of the PHY that passes read training. Ideally, this is equal to <i>Read Latency</i> + 1.

## Notes:

1. These attributes should only be modified when an error is encountered during LPDDR4 memory training.
2. These attributes are used only when skipping LPDDR4 training during simulation. It is also possible to skip training on hardware when set correctly, but this is not recommended as the delays/VREFs will not be adjusted according to environmental temperature variations.

## 4. Signal Description

The input and output signals of the LPDDR4 Memory Controller for Nexus Devices are covered in the following section. The signals available are based off the selected configuration of the LPDDR4 Memory Controller IP.

### 4.1. Clock and Reset

The following section describes the interface ports for clock and reset in the LPDDR4 Memory Controller IP.

**Table 4.1. Clock and Reset Port Definitions**

Port Name	I/O	Width	Description
pll_refclk_i	In	1	PLL reference clock input
pll_rst_n_i	In	1	PLL reset active low
pclk_i	In	1	Clock for APB interface, training CPU and control logic of the internal PLL. This clock is independent of sclk_o, since sclk_o stops during clock frequency changes.
preset_n_i	In	1	Asynchronous active low reset for APB interface. This reset must be de-asserted synchronous to pclk_i.
pll_lock_o	Out	1	PLL lock output indicating when PLL is locked
sclk_o	Out	1	System clock. This is ¼ of the DDR clock frequency and is the main clock of the LPDDR4 Memory Controller IP.
rst_n_i	In	1	Asynchronous active low reset. When asserted, output ports and registers are forced to their reset values. The LPDDR4 Memory Controller IP implements logic to de-assert the internal reset synchronous to the internal clocks after rst_n_i de-asserts.
ack_i	In	1	AXI4 I/F clock. Available in IP Core v2.x.x This is only available when <i>Enable Local Bus Clock</i> attribute is checked
areset_n_i	In	1	Asynchronous active low reset for AXI4 I/F. Available in IP Core v2.x.x This reset must be de-asserted synchronous to ack_i. This is only available when <i>Enable Local Bus Clock</i> attribute is checked.

### 4.2. Interrupts and Initialization/Training

The following section describes the interface ports for interrupts and initialization/training control in the LPDDR4 Memory Controller IP.

**Table 4.2. Interrupts and Initialization/Training Port Definitions**

Port Name	I/O	Width	Description
irq_o	Out	1	Interrupt signal Reset value is 1'b0
init_start_i	In	1	Starts the memory initialization and training according to trn_opr_i. This signal is available when <i>Enable APB I/F</i> attribute is unchecked.
init_done_o	Out	1	Indicates completion of initialization and training and the memory is available for access through the data interface.
trn_opr_i	In	8	Sets the TRN_OP_REG, which specifies the training steps to perform. Refer to the <a href="#">Training Operation Register (TRN_OP_REG)</a> (0x20) section of this User Guide for more information. This signal is available when <i>Enable APB I/F</i> attribute is unchecked.
trn_err_o	Out	1	Indicates failure in training

### 4.3. AHB-Lite Data Interface

The following section describes the data interface port when the *Local Data Bus Type* attribute is set to AHB-Lite in the LPDDR4 Memory Controller IP. These ports are available only when using IP Core v1.x.x.

Refer to the [AMBA AHB Protocol Specification](#) for a description of these signals. The transactions allowed on the AHB-Lite interface to the Memory Controller are described in [Table 2.2](#).

**Table 4.3. AHB-Lite Interface Port Definitions**

Port Name	I/O	Width	Description
ahbl_hsel_i	In	1	Start transaction
ahbl_hready_i	In	1	Ready signal from AHB-Lite interconnect. When connecting to an AHB-Lite manager directly, connect this to ahbl_hreadyout_o or set to 1.
ahbl_haddr_i <sup>1</sup>	In	AHBL_ADDR_WIDTH	Request address
ahbl_hburst_i	In	3	Burst type
ahbl_hsize_i	In	3	Indicates the size of the transfer
ahbl_hmastlock_i	In	1	AHB-Lite HMASTLOCK signal – not used for this IP
ahbl_hprot_i	In	4	AHB-Lite HPROT signal – not used for this IP
ahbl_htrans_i	In	1	Transfer type IDLE, BUSY, NONSEQ, SEQ
ahbl_hwrite_i	In	1	1 = Write, 0 = Read
ahbl_hwdata_i <sup>1</sup>	In	AHBL_DATA_WIDTH	The write data for write transactions
ahbl_hreadyout_o	Out	1	Read response valid
ahbl_hrdata_o <sup>1</sup>	Out	AHBL_DATA_WIDTH	Read data response
ahbl_hresp_o	Out	1	Read response state

Notes:

1. The bit width of ahbl\_haddr\_i/ahbl\_hwdata\_i/ahbl\_hrdata\_o is calculated based on the *DDR density* and *DDR Bus Width* attributes in [Table 3.3](#). Refer to [Table 5.4](#) for the address mapping.

### 4.4. AXI4 Data Interface

The following section describes the data interface port when the *Local Data Bus Type* attribute is set to AXI4 in the LPDDR4 Memory Controller IP. These ports are available only when using IP Core v2.x.x.

Refer to the [AMBA AXI Protocol Specification](#) for a description of these signals. The transactions allowed on the AXI4 interface to the Memory Controller are described in [Table 2.3](#).

**Table 4.4. AXI4 Interface Port Definitions**

Port Name	I/O	Width	Description
axi_arid_i	In	AXI_ID_WIDTH	AXI4 read address channel: Read address ID signal
axi_araddr_i <sup>1</sup>	In	AXI_ADDR_WIDTH	AXI4 read address channel: Read address signal
axi_arlen_i	In	8	AXI4 read address channel: Burst length signal Supports up to burst length 64 only, it is prohibited to issue more than this
axi_arsize_i	In	3	AXI4 read address channel: Burst size signal
axi_arburst_i	In	2	AXI4 read address channel: Burst type signal Only INCR is supported
axi_arqos_i	In	4	AXI4 read address channel: Quality of service signal This signal is currently unused
axi_arvalid_i	In	1	AXI4 read address channel: Read address valid signal
axi_arready_o	Out	1	AXI4 read address channel: Read address ready signal
axi_rid_o	Out	AXI_ID_WIDTH	AXI4 read data channel: Read ID tag signal
axi_rdata_o <sup>1</sup>	Out	AXI_DATA_WIDTH	AXI4 read data channel: Read data signal
axi_rresp_o	Out	2	AXI4 read data channel: Read response signal

Port Name	I/O	Width	Description
axi_rlast_o	Out	1	AXI4 read data channel: Read last signal
axi_rvalid_o	Out	1	AXI4 read data channel: Read valid signal
axi_rready_i	In	1	AXI4 read data channel: Read ready signal
axi_awid_i	In	AXI_ID_WIDTH	AXI4 write address channel: Write address ID signal
axi_awaddr_i <sup>1</sup>	In	AXI_ADDR_WIDTH	AXI4 write address channel: Write address signal
axi_awlen_i	In	8	AXI4 write address channel: Burst length signal
axi_awsz_i	In	3	AXI4 write address channel: Burst size signal
axi_awburst_i	In	2	AXI4 write address channel: Burst type signal
axi_awqos_i	In	4	AXI4 write address channel: Quality of service signal
axi_awvalid_i	In	1	AXI4 write address channel: Write address valid signal
axi_awready_o	Out	1	AXI4 write address channel: Write address ready signal
axi_wdata_i <sup>1</sup>	In	AXI_DATA_WIDTH	AXI4 write data channel: Write data signal
axi_wstrb_i	In	AXI_STRB_WIDTH	AXI4 write data channel: Write strobe signal
axi_wlast_i	In	1	AXI4 write data channel: Write last signal
axi_wvalid_i	In	1	AXI4 write data channel: Write valid signal
axi_wready_o	Out	1	AXI4 write data channel: Write ready signal
axi_bid_o	Out	AXI_ID_WIDTH	AXI4 write response channel: Response ID tag signal
axi_bresp_o	Out	2	AXI4 write response channel: Write response signal
axi_bvalid_o	Out	1	AXI4 write response channel: Write response valid signal
axi_bready_i	In	1	AXI4 write response channel: Response ready signal

Notes:

1. The bit width of axi\_araddr\_i/axi\_rdata\_o/axi\_awaddr\_i/axi\_wdata\_i is calculated based on the *DDR density* and *DDR Bus Width* attributes in Table 3.3. Refer to Table 5.4 for the address mapping.

## 4.5. APB Register Interface

The following section describes the configuration interface port when the *Enable APB I/F* attribute is checked in the LPDDR4 Memory Controller IP. Refer to the [AMBA APB Protocol Specification](#) for a description of these signals.

**Table 4.5. APB Interface Port Definitions**

Port Name	I/O	Width	Description
apb_psel_i	In	1	APB Select signal Indicates that the subordinate device is selected and a data transfer is required
apb_paddr_i	In	12	APB Address signal
apb_pwdata_i	In	32	APB Write data signal Bits [31:8] are not used
apb_pwrite_i	In	1	APB Direction signal 1 = Write, 0 = Read
apb_penable_i	In	1	APB Enable signal Indicates the second and subsequent cycles of an APB transfer
apb_pready_o	Out	1	APB Ready signal Indicates transfer completion. Subordinate uses this signal to extend an APB transfer. Reset value is 1'b0.
apb_pslverr_o	Out	1	APB Error signal Indicates a transfer failure. This signal is tied to 1'b0.
apb_prdata_o	Out	32	APB Read data signal



## 4.6. LPDDR4 Memory Interface

The following section describes the interface ports for LPDDR4 SDRAM.

**Table 4.6. LPDDR4 Interface Port Definitions**

Port Name	I/O	Width	Description
ddr_ck_o <sup>1</sup>	Out	CK_WIDTH	LPDDR4 CK signal
ddr_cke_o <sup>1</sup>	Out	CK_WIDTH	LPDDR4 CKE signal
ddr_cs_o <sup>1</sup>	Out	CS_WIDTH	LPDDR4 CS signal
ddr_ca_o	Out	6	LPDDR4 CA signal
ddr_reset_n_o	Out	1	Memory reset signal
ddr_dq_io <sup>1</sup>	In/Out	BUS_WIDTH	LPDDR4 DQ signal
ddr_dqs_io <sup>1</sup>	In/Out	DQS_WIDTH	LPDDR4 DQS signal
ddr_dmi_io <sup>1</sup>	In/Out	DQS_WIDTH	LPDDR4 DMI signal

Notes:

1. The bit width of SDRAM Memory Interface signals is defined based on the attributes listed in [Table 3.3](#).

## 5. Register Description

This section describes the user accessible registers in the LPDDR4 Memory Controller IP. Some registers are marked reserved for internal CPU usage and should not be written to. Writing to these registers may cause failures during the initialization and training operations.

**Table 5.1. Summary of LPDDR4 Memory Controller IP Registers**

Offset	Register Name	Access Type	Description
0x00	FEATURE_CTRL_REG	RW	Feature Control Register
0x04	RESET_REG	RW	Reset Register
0x08	SETTINGS_REG	RW	Settings Register
0x0C	Reserved	—	Reserved
0x10	INT_STATUS_REG	RW1C	Interrupt Status Register
0x14	INT_ENABLE_REG	RW	Interrupt Enable Register
0x18	INT_SET_REG	WO	Interrupt Set Register
0x1C	For internal use	—	Reserved for use by the internal CPU
0x20	TRN_OP_REG	RW	Training Operation Register
0x24	STATUS_REG	RW	Status Register
0x28 to 0xA4	For internal use	—	Reserved for use by the internal CPU
0xA8 onwards	Reserved	—	Reserved

**Table 5.2. Register Access Type Definitions**

Access Type	Behavior on Read Access	Behavior on Write Access
RO	Returns register value	Ignores write access
WO	Returns 0	Updates register value
RW	Returns register value	Updates register value
RW1C	Returns register value	Writing 1'b1 on register bit clears the bit to 1'b0 Writing 1'b0 on register bit is ignored
RSVD	Returns 0	Ignores write access

### 5.1. Feature Control Register (FEATURE\_CTRL\_REG) (0x00)

The Feature Control Register reflects the modes of operation specified according to the attributes selected during IP configuration. These attributes are set during IP configuration and cannot be modified during run-time. The CPU reads this register to identify the modes of operation.

**Table 5.3. Feature Control Register**

Field	Name	Access	Width	Reset
[31:17]	reserved	RSVD	15	—
[16]	num_ranks	RO	1	Number of Ranks
[15:12]	ddr_width	RO	4	DDR Bus Width
[11:8]	ddr_type	RO	4	DDR Interface Type
[7:4]	addr_translation	RO	4	0
[3]	gear_ratio	RO	1	Gearing Ratio
[2]	pwr_down_en	RO	1	Enable Power Down
[1]	dbi_en	RO	1	Enable DBI
[0]	reserved	RSVD	1	—

#### num\_ranks

- The num\_ranks is fixed at value 0, which corresponds to single rank.

#### ddr\_width

- The ddr\_width specifies the bit width of the DDR data bus as follows:
  - 0 – *DDR Bus Width* = 8 bits
  - 1 – *DDR Bus Width* = 16 bits
  - 2 – reserved
  - 3 – *DDR Bus Width* = 32 bits
  - 4 – reserved
  - 5 – reserved
  - 6 – reserved
  - 7 – *DDR Bus Width* = 64 bits

#### ddr\_type

- The ddr\_type is fixed at value 1, which corresponds to LPDDR4 as the standard implemented by the Memory Controller IP Core.

#### addr\_translation

- The address translation specifies the mapping of the address bits of the local memory-mapped address and the memory address in terms of row, column, bank, and rank. Only 1 address translation scheme (addr\_translation=0) is currently supported; refer to [Table 5.4](#) and [Table 5.5](#) for details.

#### gear\_ratio

- The gear\_ratio is fixed at value 1, which corresponds to 8:1 gearing and specifies the ratio of the DDR clock domain and system clock domain as ddr\_ck\_o frequency = 4 × sclk\_o frequency.

#### pwr\_down\_en

- The pwr\_down\_en enables the power saving mode by putting the memory in self-refresh when there is no traffic for sclk\_o cycles as specified in the *No. of SCLK to enter Self-Refresh from no traffic* attribute.

#### dbi\_en

- The dbi\_en enables the Data Bus Inversion function to reduce the toggling of DDR data signals, thus improving the signal integrity and reducing dynamic power consumption as specified in the *Enable DBI* attribute.

**Table 5.4. Address Mapping for addr\_translation=0**

Memory Address	Size	Local Address Map	
Row	ROWW = refer to memory device data sheet	ROW_H = ROW_L + ROWW - 1 ROW_L = BANK_H + 1	Addr[ROW_H: ROW_L]
Bank	BANKW = 3	BANK_H = COL_H + BANKW - 1 BANK_L = COL_H + 1	Addr[BANK_H: BANK_L]
Column	COLW = 10	COL_H = COL_L + COLW - 1 COL_L = OFFSETW	Addr[COL_H: COL_L]
Offset	If <i>DDR Bus Width</i> == 64: OFFSETW = 3 If <i>DDR Bus Width</i> == 32: OFFSETW = 2 If <i>DDR Bus Width</i> == 16: OFFSETW = 1 If <i>DDR Bus Width</i> == 8: OFFSETW = 0	N/A	

**Table 5.5. Address Mapping Example**

Memory Address	Example User Value	Actual Line Size	Local Address Map
Offset	<i>DDR Bus Width = 32</i>	2	*addr_i[1:0]
Column Width (COLW)	10 (Fixed for LPDDR4)	10	*addr_i[11:2]
Bank Width (BANKW)	3 (Fixed for LPDDR4)	3	*addr_i[14:12]
Row Width (ROWW)	<i>DDR Density = 4</i>	15	*addr_i[29:15]
Rank Width (RANKW)	<i>Number of Ranks = 1</i>	0	--
Total Local Address Line Size		30	*addr_i[29:0]

## 5.2. Reset Register (RESET\_REG) (0x04)

The Reset Register controls the reset of the internal CPU and Training Engine, where both are reset at power-on. The host should de-assert the reset to the internal CPU and Training Engine to begin memory initialization and training. Upon training completion, the internal CPU sets the trn\_eng\_rst\_n signal low (RESET\_REG[0]=0), to place the Training Engine in reset to save power. When the *Enable APB I/F* attribute is unchecked, the init\_start\_i signal controls the internal CPU reset (RESET\_REG[1]). This register does not reset the Configuration Set Registers (CSRs).

**Table 5.6. Reset Register**

Field	Name	Access	Width	Reset
[31:2]	reserved	RSVD	30	—
[1]	cpu_rst_n	RW	1	0
[0]	trn_eng_rst_n	RW	1	0

## 5.3. Settings Register (SETTINGS\_REG) (0x08)

The Settings Register controls the DDR write and read latencies according to the attributes selected during IP configuration.

**Table 5.7. Settings Register**

Field	Name	Access	Width	Reset
[31:17]	reserved	RSVD	16	—
[16]	zq_cal_sel	RW	1	1
[15:12]	reserved	RSVD	4	—
[11:8]	read_latency	RW	4	<i>Read Latency</i>
[7:4]	reserved	RSVD	4	—
[3:0]	write_latency	RW	4	<i>Write Latency</i>

### zq\_cal\_sel

- The zq\_cal\_sel specifies the ZQ calibration operation for periodic ZQ calibration:
  - 0 – ZQ Calibration Short
  - 1 – ZQ Calibration Long

### read\_latency

- The read\_latency specifies the number of DDR clock cycles from read command to the first read data.

### write\_latency

- The write\_latency specifies the number of DDR clock cycles from write command to the first write data.

## 5.4. Interrupt Status Register (INT\_STATUS\_REG) (0x10)

The following table lists all the pending interrupts supported in the Memory Controller IP. When an interrupt bit asserts, it remains asserted until it is cleared by the host writing 1'b1 to the corresponding bit.

The interrupt status bits are independent of the interrupt enable bits. In other words, status bits may indicate pending interrupts, even though those interrupts are disabled in the Interrupt Enable Register. User logic that handles interrupts should mask (bitwise and logic) the contents of INT\_STATUS\_REG and INT\_ENABLE\_REG to determine which interrupts to service. The irq\_o interrupt signal is asserted whenever both an interrupt status bit and the corresponding interrupt enable bits are set.

**Table 5.8. Interrupt Status Register**

Field	Name	Access	Width	Reset
[31:5]	reserved	RSVD	27	—
[4]	temp_change_int	RW1C	1	0
[3:2]	reserved	RSVD	2	—
[1]	trn_err_int	RW1C	1	0
[0]	trn_done_int	RW1C	1	0

### temp\_change\_int

- Temperature Change Interrupt. The Memory Controller periodically reads MR4 of the LPDDR4 memory device according to the *Temperature Check Period* attribute. This interrupt bit asserts when MR4 indicates a temperature change, and the refresh rate is not equal to 1x refresh.
  - 0 – No interrupt
  - 1 – Interrupt pending

### trn\_err\_int

- Training Error Interrupt. This Interrupt bit asserts when the Training Engine encounters an error during training. The user should read STATUS\_REG to determine the specific error.
  - 0 – No interrupt
  - 1 – Interrupt pending

### trn\_done\_int

- Training Done Interrupt. This Interrupt bit asserts when initialization and training is completed successfully.
  - 0 – No interrupt
  - 1 – Interrupt pending

## 5.5. Interrupt Enable Register (INT\_ENABLE\_REG) (0x14)

The Interrupt Enable Register lists all configurable interrupts within the Memory Controller IP.

**Table 5.9. Interrupt Enable Register**

Field	Name	Access	Width	Reset
[31:5]	reserved	RSVD	27	—
[4]	temp_change_en	RW	1	0
[3:2]	reserved	RSVD	1	—
[1]	trn_err_en	RW	1	0
[0]	trn_done_en	RW	1	0

### temp\_change\_en

- Temperature Change Interrupt Enable.
  - 0 – Interrupt disabled
  - 1 – Interrupt enabled

#### trn\_err\_en

- Training Error Interrupt Enable.
  - 0 – Interrupt disabled
  - 1 – Interrupt enabled

#### trn\_done\_en

- Training Done Interrupt Enable.
  - 0 – Interrupt disabled
  - 1 – Interrupt enabled

## 5.6. Interrupt Set Register (INT\_SET\_REG) (0x18)

The following table shows a summary of the Interrupt Set Register. Writing 1'b1 to a register bit in this register asserts the corresponding interrupt status bits in INT\_STATUS\_REG. Writing 1'b0 is ignored.

**Table 5.10. Interrupt Set Register**

Field	Name	Access	Width	Reset
[31:5]	reserved	RSVD	27	—
[4]	temp_change_set	WO	1	0
[3:2]	reserved	RSVD	2	—
[1]	trn_err_set	WO	1	0
[0]	trn_done_set	WO	1	0

#### temp\_change\_set

- Temperature Change Interrupt Set.
  - 0 – No Action
  - 1 – Assert the temp\_change\_int signal (INT\_STATUS\_REG[4]=1)

#### trn\_err\_set

- Training Error Interrupt Set.
  - 0 – No Action
  - 1 – Assert the trn\_err\_int signal (INT\_STATUS\_REG[1]=1)

#### trn\_done\_set

- Training Done Interrupt Set.
  - 0 – No Action
  - 1 – Assert the trn\_done\_int signal (INT\_STATUS\_REG[0]=1)

## 5.7. Training Operation Register (TRN\_OP\_REG) (0x20)

The following table shows a summary of the Training Operation Register. This register controls the memory initialization and training. It is recommended to set these register bits to 1'b0 during simulation to shorten the initialization and training procedure. When *Enable APB I/F* is unchecked, TRN\_OP\_REG[7:0] is set to the value of the trn\_opr\_i input signal.

**Table 5.11. Training Operation Register**

Field	Name	Access	Width	Reset
[31:8]	reserved	RSVD	24	—
[7:5]	reserved	RSVD	3	—
[4]	write_trn_en	RW	1	1
[3]	read_trn_en	RW	1	1
[2]	write_lvl_en	RW	1	1
[1]	cbt_en	RW	1	1
[0]	init_en	RW	1	1

#### write\_trn\_en

- Enables write training during initialization and training. Write training optimizes the write DQ delay with respect to the write DQS to improve the data valid window for write operations. If write training fails for a certain latency, the Training Engine will increase the latency setting by 1 and retry write training.
  - 0 – The Training Engine programs the *Trained Write DQ/DBI delay* and *Trained Write Latency to PHY* attributes, and checks that write and read FIFO access is equal.
  - 1 – The Training Engine performs write training.

#### read\_trn\_en

- Enables read training during initialization and training. Read training tunes the PHY to capture the incoming read DQS burst according to the read latency setting. If read training fails for a certain latency setting, the Training Engine will increase the latency setting by 1 and retry read training. Once the read DQS burst is captured properly, the read DQS-DQ timing is trained to improve the read data valid window.
  - 0 – The Training Engine programs the *DQS<0,1,2,3,4,5,6,7> Trained RDCLKSEL*, *Trained DQSBUF Read Delay/Sign*, and *Trained Read Latency* attributes to the PHY and checks that read access will be successful.
  - 1 – The Training Engine performs read training.

#### write\_lvl\_en

- Enables write leveling during initialization and training. Write leveling compensates for CK-DQS timing skews. The Training Engine performs write leveling according to the JEDEC standard.
  - 0 – The Training Engine programs the *DQS<0,1,2,3,4,5,6,7> Trained Write Leveling Delay* attributes to the PHY and checks that the DQ feedback after DQS pulse is high.
  - 1 – The Training Engine performs write leveling.

#### cbt\_en

- Enables Command Bus Training (CBT) during initialization and training. CBT performs CA\_VREF programming and aligns the CS/CA and CK for high frequency operation.
  - 0 – The Training Engine will shorten the command bus training. Instead of iterating through the different CA delays to find the optimal delay value, it will only program the *Trained CA Delay* attribute to PHY.
  - 1 – The Training Engine performs command bus training to find the optimal CA delay value.

#### init\_en

- Provides ability to shorten initialization for simulation purposes.
  - 0 – Initialization is greatly reduced. For example, reset time and CKE low time is greatly shortened.
  - 1 – Initialization is performed according to the [LPDDR4 JEDEC Standard](#).

## 5.8. Status Register (STATUS\_REG) (0x24)

The following table shows a summary of the Status Register. The internal CPU writes to this register to communicate the status to the host CPU.

**Table 5.12. Status Register**

Field	Name	Access	Width	Reset
[31:19]	reserved	RSVD	13	—
[18:16]	refresh_rate	RO	3	—
[15:14]	reserved	RSVD	2	—
[13:12]	error_on_rank	RW	2	0
[11]	write_trn_err	RW	1	0
[10]	read_trn_err	RW	1	0
[9]	write_lvl_err	RW	1	0
[8]	cbt_err	RW	1	0
[7:6]	reserved	RSVD	2	—
[5]	in_self_refresh	RO	1	0
[4]	write_trn_done	RW	1	0

Field	Name	Access	Width	Reset
[3]	read_trn_done	RW	1	0
[2]	write_lvl_done	RW	1	0
[1]	cbt_done	RW	1	0
[0]	phy_ready	RO	1	0

#### refresh\_rate

- Reflects the value of the refresh rate set in MR4 within LPDDR4 memory. It specifies the required refresh period based on the temperature of the LPDDR4 device.
  - 3'b000: SDRAM Low temperature operating limit exceeded
  - 3'b001: 4x refresh
  - 3'b010: 2x refresh
  - 3'b011: 1x refresh (default)
  - 3'b100: 0.5x refresh
  - 3'b101: 0.25x refresh, no de-rating
  - 3'b110: 0.25x refresh, with de-rating
  - 3'b111: SDRAM High temperature operating limit exceeded

#### error\_on\_rank

- When error\_on\_rank is set to 2'bx1, it indicates a training error has occurred on rank 0.

#### write\_trn\_err

- Asserts when a failure occurs during write training.

#### read\_trn\_err

- Asserts when a failure occurs during read training. If the read\_trn\_done signal is low (STATUS\_REG[3]=0), the training failed to find a setting that properly captures the read DQS burst. If the read\_trn\_done signal is high (STATUS\_REG[3]=1), the training failed to find an optimal read DQS-DQ delay for capturing the read data.

#### write\_lvl\_err

- Asserts when a failure occurs during write leveling.

#### cbt\_err

- Asserts when a failure occurs during Command Bus Training.

#### in\_self\_refresh

- Asserts when the memory is in self-refresh.

#### write\_trn\_done

- Asserts when write training has completed.

#### read\_trn\_done

- Asserts when read training has completed.

#### write\_lvl\_done

- Asserts when write leveling has completed.

#### cbt\_done

- Asserts when Command Bus Training has completed.

#### phy\_ready

- Asserts when PHY initialization of the PHY is complete and ready for operation.



## 6. LPDDR4 Memory Controller Example Design

This section describes the LPDDR4 Memory Controller Example Design that is available to users for synthesis and simulation after successful IP generation.

### 6.1. Overview

The following table summarizes the IP parameter configurations that are reflected in the LPDDR4 Memory Controller Example Design. For steps on how to generate the Memory Controller IP Core, refer to the [Designing and Simulating the IP](#) section of this User Guide.

**Table 6.1. Supported Example Design Configurations**

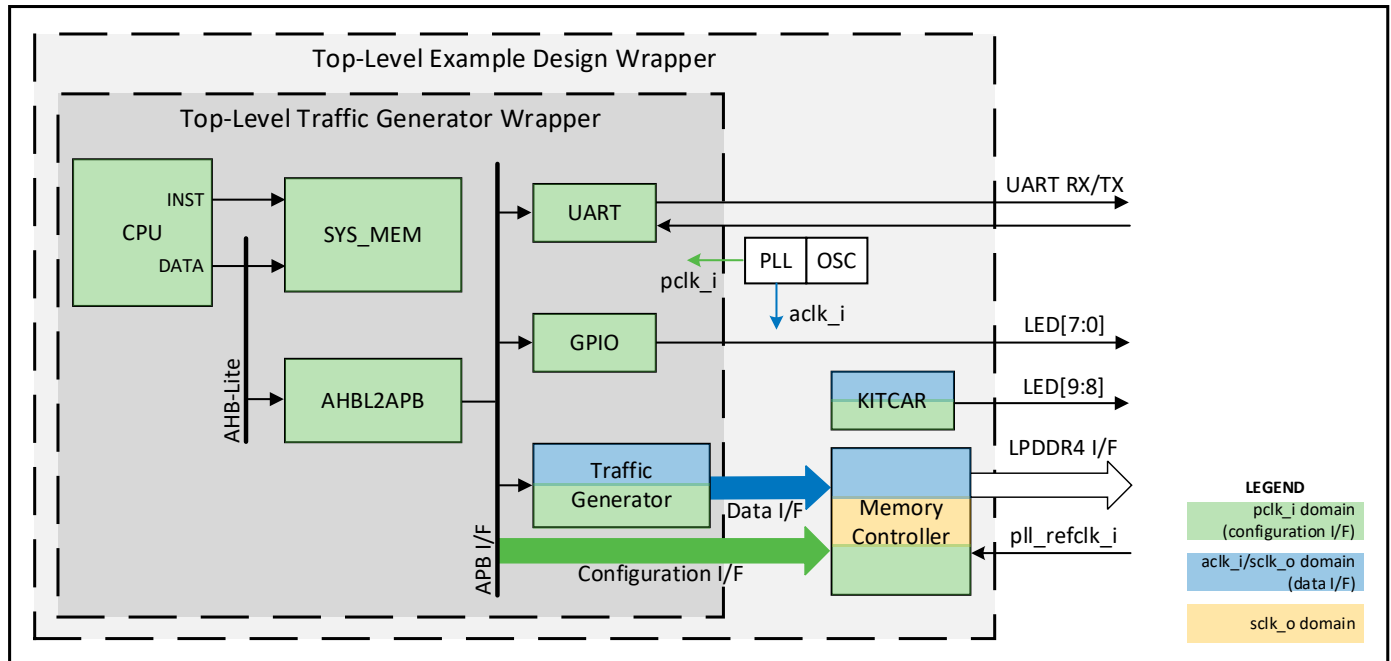
Attribute <sup>1</sup>	Supported Setting
I/O Buffer Type	LVSTL_I, LVSTL_II
DDR Command Frequency	300, 350, 400, 533
Enable Power Down	Checked, Unchecked
Enable DBI	Checked, Unchecked
PLL Reference Clock from Pin	Checked, Unchecked
DDR Bus Width	16, 32, 64
Local Data Bus Type	AHBL, AXI
ID Width	All
Number of Outstanding Writes/Reads	All
Write/Read Ordering Queues	All
Enable Local Bus Clock	Checked, Unchecked
Enable APB I/F	Checked, Unchecked
Memory Device Timing Tab	All
Training Settings Tab	All

Notes:

1. All configurable attributes are supported in the LPDDR4 Memory Controller Example Design.

### 6.2. Synthesis Example Design

After successful generation of the LPDDR4 Memory Controller for Nexus Devices, a synthesizable example design becomes available to users. This design contains a test program that allows users to evaluate the Memory Controller IP on hardware. [Figure 6.1](#) represents a block diagram of the LPDDR4 Memory Controller Example Design.



**Figure 6.1. Memory Controller Example Design Functional Diagram**

The main blocks that make up the synthesizable example design include the following:

- RISC-V CPU subsystem – handles initialization and training of the LPDDR4 interface and performs data access checks.
- System Memory (SYS\_MEM) – stores the instruction code for the example design test program.
- AHBL2APB bridge – converts the CPU data interface (AHB-Lite) to the configuration interface (APB).
- UART block – allows users to interface with the test program and prints out results via a serial terminal connection.
- GPIO block – allows users to verify the result of LPDDR4 training sequences and the functionality of pclk\_i and aclk\_i clock signals.
- Traffic generator block – implements a series of pseudo-random (PRBS) writes and reads, and compares the read data to the expected data, displaying the result over a UART connection.
- LPDDR4 Memory Controller IP – provides an interface to external LPDDR4 memory to issue reads and writes.
- Oscillator (OSC) – generates a 90 MHz clock for the configuration interface (pclk\_i).
- PLL – included only in IP Core v2.x.x, and takes the oscillator output as an input reference clock to the PLL. The PLL generates both the configuration interface clock (pclk\_i) and the user data interface AXI clock (aclk\_i).

The synthesizable example design consists of a test program that is stored in system memory and is fetched by the CPU instruction port. The CPU data port accesses system memory and connects to the following: Configuration Set Registers (CSRs), UART, GPIO, traffic generator and LPDDR4 Memory Controller. The test program associated with the example design performs the following:

- Waits for user input over a serial terminal connection upon the assertion of the reset signal: rstn\_i
- Configures the Memory Controller to perform complete reset, initialization, and training (TRN\_OP\_REG=0x1F) of the LPDDR4 interface.
- Performs a series of loop-back data access checks
- Allows users to change their VREF settings
- Calculates performance of the LPDDR4 interface

The top-level example design wrapper file, eval\_top.sv, provides ports for a PLL reference clock, LPDDR4 interface, UART interface, and a 10-bit LED output. LED[7:0] corresponds to bits STATUS\_REG[8:1], whereas toggling on LED[8] signifies aclk\_i is functioning and toggling on LED[9] signifies pclk\_i is functioning. For information on the example design test program and instructions on how to generate and perform hardware evaluation of the LPDDR4 Memory Controller Example Design, refer to the [Designing and Simulating the IP](#) section of this User Guide.

## 6.3. Simulation Example Design

The simulation example design is similar to the synthesizable example design, apart from the following changes:

- Disables UART connection
- External LPDDR4 memory is replaced with LPDDR4 memory model

The UART connection is disabled in simulation since it takes a long time to simulate and the LPDDR4 memory model allows simulation of user-initiated operations to LPDDR4 SDRAM. For instructions on how to simulate the LPDDR4 Memory Controller, including the example design, refer to the [Designing and Simulating the IP](#) section of this User Guide.

The following table summarizes the simulation runtime of the LPDDR4 Memory Controller for various configurations.

**Table 6.2. Simulation Runtime Summary**

LPDDR4 Configuration	Typical Initialization Time	Typical Training Time
x16, 533 MHz	2251 $\mu$ s	328 $\mu$ s
x32, 533 MHz	2251 $\mu$ s	439 $\mu$ s
x64, 533 MHz	2251 $\mu$ s	543 $\mu$ s
x16, 400 MHz	2251 $\mu$ s	430 $\mu$ s
x32, 350 MHz	2251 $\mu$ s	536 $\mu$ s
x64, 300 MHz	2251 $\mu$ s	753 $\mu$ s

The initialization time is measured from the moment the CPU and Controller Engine is pulled out of reset (RESET\_REG) to the assertion of the LPDDR4 Clock Enable signal (ddr\_cke\_o). The training time is measured from ddr\_cke\_o assertion to the assertion of init\_done\_o, which signifies the completion of the initialization and training sequence. For more details on the initialization and training sequences, refer to the LPDDR4 Calibration section of this User Guide.

## 7. Designing and Simulating the IP

This section describes the steps required within Lattice Radiant software to configure and generate the LPDDR4 Memory Controller for Nexus Devices. This section also provides information regarding design implementation and hardware evaluation of the synthesizable example design.

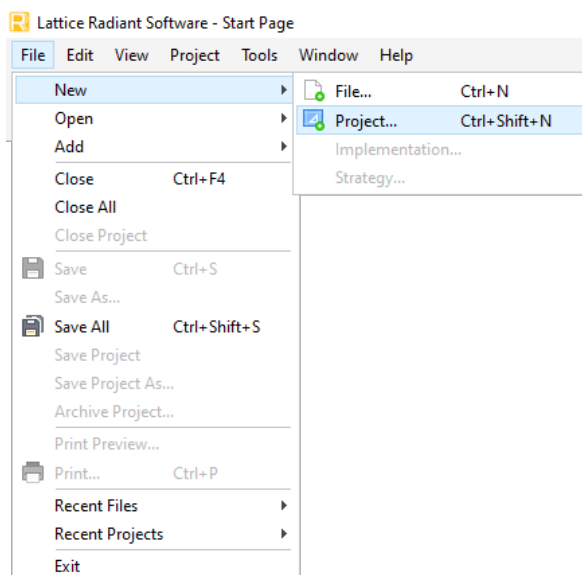
### 7.1. Generating the IP

This section describes the steps required to create, configure, and generate an instance of the LPDDR4 Memory Controller for Nexus Devices.

#### 7.1.1. Creating a Radiant Project

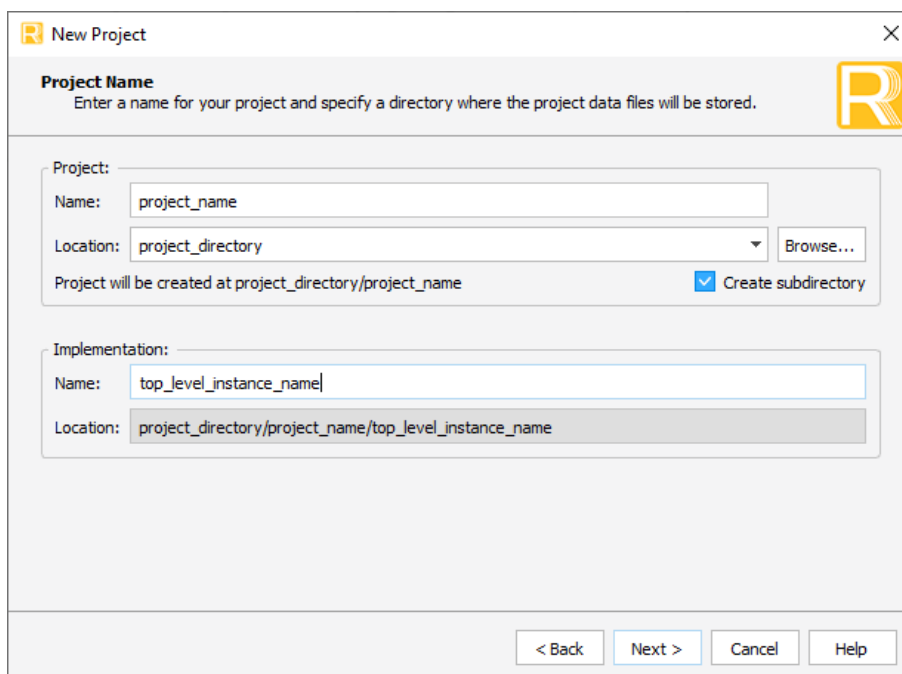
In order to generate an instance of the Memory Controller IP, a Lattice Radiant project must first be created.

1. Launch the Lattice Radiant software and select **File > New > Project**. This will open the **New Project** dialog box. Click **Next**.



**Figure 7.1. Creating a New Radiant Project**

2. Specify a name (<project\_name>) for the Lattice Radiant project, a directory (<project\_directory>) to store the project files, and a top-level design implementation name (<top\_level\_instance\_name>). Click **Next** two times.



**New Project**

**Project Name**  
Enter a name for your project and specify a directory where the project data files will be stored.

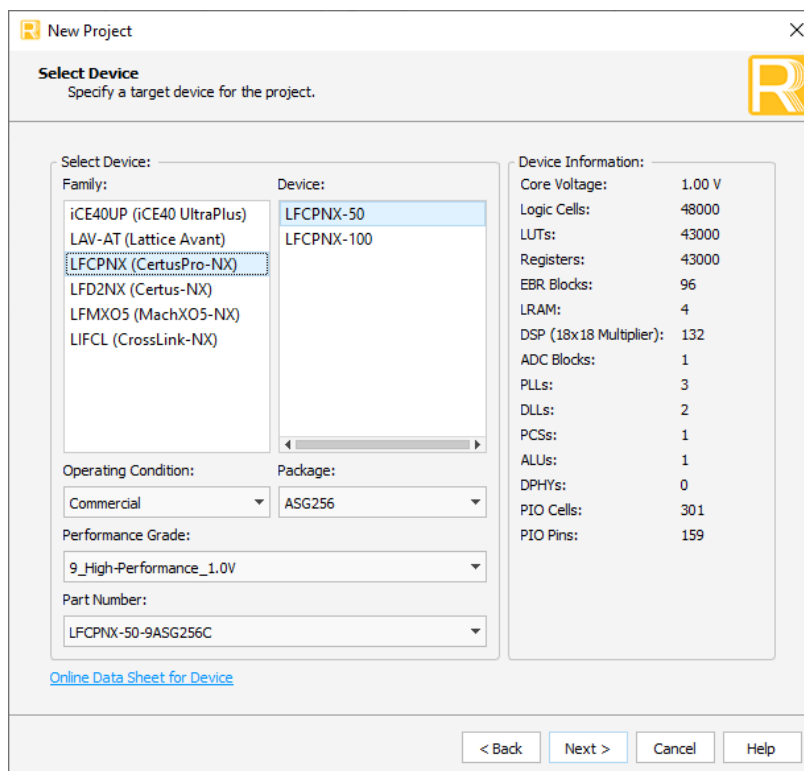
Project:  
Name:   
Location:    
Project will be created at project\_directory/project\_name ☒ Create subdirectory

Implementation:  
Name:   
Location:

< Back Next > Cancel Help

Figure 7.2. New Project Settings

- Under **Family**, select CertusPro-NX or MachXO5T-NX. Under **Device, Package, Operating Condition, and Performance Grade**, make the appropriate selections representative of the selected device part number. Click **Next**.



**Select Device**  
Specify a target device for the project.

Select Device:  
Family:  
iCE40UP (iCE40 UltraPlus)  
LAV-AT (Lattice Avant)  
**LFCPNX (CertusPro-NX)**  
LFD2NX (Certus-NX)  
LFMXO5 (MachXO5-NX)  
LIFCL (CrossLink-NX)  
Device:  
LFCPNX-50  
LFCPNX-100

Operating Condition:  Package:   
Performance Grade:   
Part Number:

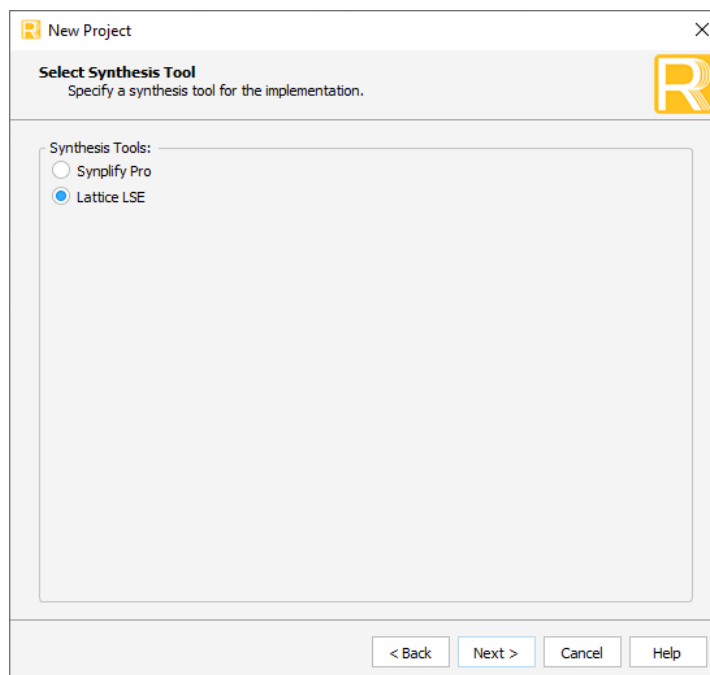
[Online Data Sheet for Device](#)

Device Information:  
Core Voltage: 1.00 V  
Logic Cells: 48000  
LUTs: 43000  
Registers: 43000  
EBR Blocks: 96  
LRAM: 4  
DSP (18x18 Multiplier): 132  
ADC Blocks: 1  
PLLs: 3  
DLLs: 2  
PCSs: 1  
ALUs: 1  
DPHYs: 0  
PIO Cells: 301  
PIO Pins: 159

< Back Next > Cancel Help

Figure 7.3. Project Device Settings

4. Specify the desired synthesis tool for implementation of the Lattice Radiant project. Click **Next** and **Finish**.

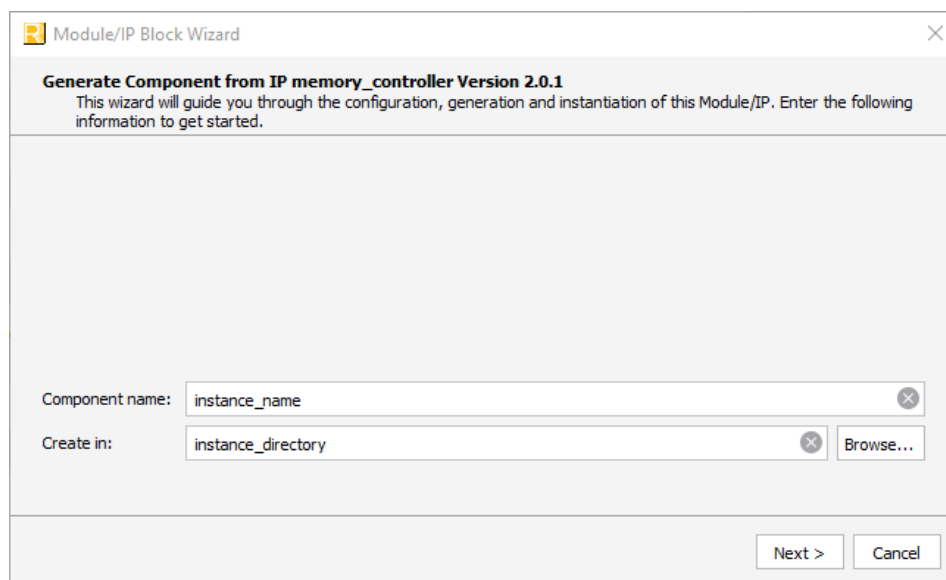


**Figure 7.4. Project Synthesis Tool Selection**

### 7.1.2. Configuring and Generating the IP

The following steps illustrate how to generate the Memory Controller IP Core in Lattice Radiant software.

1. Under the **IP Catalog** (Tools > IP Catalog), locate and double-click on the desired Memory Controller IP listed under **IP > Processors\_Controllers\_and\_Peripherals**.
  - a. If no Memory Controller IPs are installed on the system, select the **IP on Server** tab within the IP Catalog.
  - b. Click on **Download from Lattice IP Server** icon next to the desired Memory Controller IP for installation. This will open an **IP License Agreement** dialog box.
  - c. Click **Accept** and then click on the **Refresh IP Catalog** icon.
  - d. Locate the installed Memory Controller IP under the **IP on Local** tab within the IP Catalog and double-click.
    - IP Core v1.x.x = Memory Controller
    - IP Core v2.0.1 = Memory Controller for CertusPro-NX
    - IP Core v2.0.x = LPDDR4 Memory Controller for Nexus
2. The **Module/IP Block Wizard** dialog box will open. Provide a name (<instance\_name>) and directory (<instance\_directory>) for the Memory Controller IP, where the default directory is set to <project\_directory>/<project\_name>. Click **Next**.



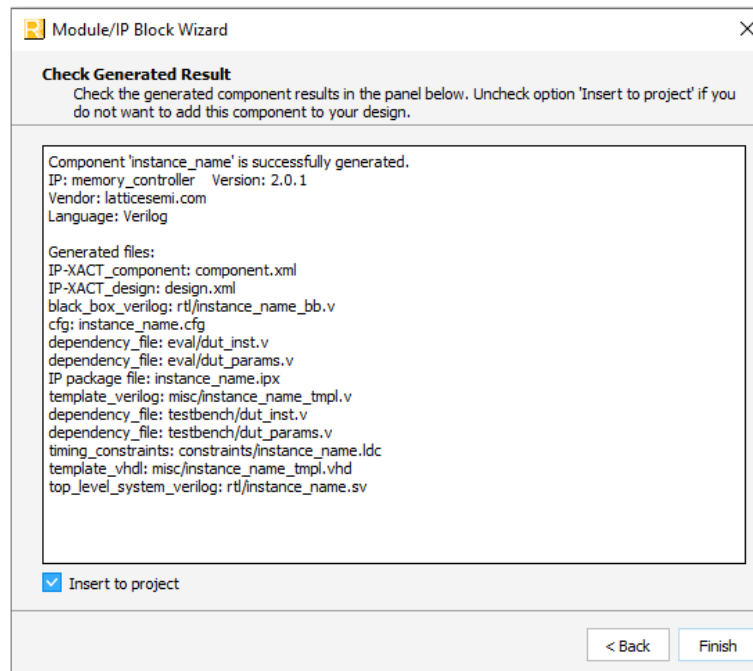
**Figure 7.5. IP Instance Settings**

- The Memory Controller IP editor contains multiple tabs that needs to be configured according to the desired LPDDR4 memory interface implementation. The following table provides high-level guidance for configuring the tabs in the Memory Controller IP editor. For detailed information on the individual attributes, refer to the [IP Parameter Description](#) section of this User Guide.

**Table 7.1. Memory Controller Attribute Guidelines**

Memory Controller IP Attribute Tab	Guidelines <sup>1</sup>
General	- Ensure that the Memory clock frequency ( <i>DDR Command Frequency</i> ) is entered correctly
Memory Device Timing	- Refer to the datasheet for the selected LPDDR4 memory device to modify the default timing parameters as needed
Training Settings (IP Core v2.x.x)	<ul style="list-style-type: none"> <li>- MC DQS VREF should only be modified if an error has occurred during read training and/or data access</li> <li>- Memory CA VREF should only be modified if an error has occurred during command bus training</li> <li>- Memory DQ VREF should only be modified if an error has occurred during write training and/or data access</li> </ul>

- Click **Generate**. The **Check Generating Result** dialog box opens, showing design block messages and results as shown in [Figure 7.6](#). Click **Finish**.



**Figure 7.6. IP Generation Result**

- All the generated files are placed under the <instance\_directory>/<instance\_name> directory path. The generated files under the <instance\_directory>/<instance\_name> directory are listed in the following table.

**Table 7.2. Generated File List**

File Name	Description
component.xml	Contains the ipxact:component information of the IP
design.xml	Lists the set parameters of the IP in IP-XACT 2014 format
<instance_name>.cfg	Lists only the configured/changed parameter values set during IP configuration
<instance_name>.ipx	Lists the files associated with IP generation
constraints/<instance_name>.ldc	Defines the I/O standard for LPDDR4 memory interface signals
eval/apb2init.sv	Implements handshaking between APB accesses and internal RISC-V CPU for initialization of LPDDR4 memory in LPDDR4 Memory Controller Example Design
eval/clock_constraint.sdc	Pre-synthesis constraint for setting the PLL reference clock frequency of the LPDDR4 Memory Controller Example Design
eval/constraint.pdc	Post-synthesis constraints for the LPDDR4 Memory Controller Example Design
eval/dut_inst.v	Instantiation of generated IP core in eval_top.sv for LPDDR4 Memory Controller Example Design
eval/dut_params.v	Defines local parameters for eval_top based on parameter values set during IP configuration for LPDDR4 Memory Controller Example Design
eval/eval_top.sv	Top-level RTL file for the LPDDR4 Memory Controller Example Design
eval/kitcar.v	Counter that drives LEDs to indicate that internal clocks (pclk and aclk) are active in LPDDR4 Memory Controller Example Design
eval/pll_aclk_pclk.v	PLL responsible for generating APB clock (pclk) and AXI4 (aclk) in LPDDR4 Memory Controller Example Design



File Name	Description
eval/select_protocol.py	Script that defines LPDDR4 protocol for eval_top and tb_top files in LPDDR4 Memory Controller Example Design
eval/axi_bridge/	Contains RTL for AXI bus interface to Native I/F
eval/traffic_gen/	Contains RTL files for the LPDDR4 Memory Controller Example Design
misc/<instance_name>.tpl.v misc/<instance_name>.vhd	These files provide instance templates for the IP core
rtl/<instance_name>.sv	Example RTL top-level file that instantiates the IP core
rtl/<instance_name>_bb.v	Example synthesizable RTL black box file that instantiates the IP core
testbench/debug_c_code.sv	For internal use only
testbench/dut_inst.v	Template instance files
testbench/dut_params.v	List of parameters based on user IP configurations
testbench/tb_top.sv	Top level testbench file
testbench/lpddr4/	Contains LPDDR4 memory model and instances for simulation

## 7.2. Design Implementation

This section describes the steps required to properly run a LPDDR4 Memory Controller for Nexus IP design on hardware.

### 7.2.1. Pin Placement

Typically, all external memory interfaces require the following FPGA resources:

- Data, data mask, and data strobe signals
- Command, address, and control signals
- PLL and clock network signals
- RZQ and VREF signals
- Other FPGA resources

In Lattice CertusPro-NX and MachXO5T-NX FPGA devices, external memory interfaces are supported in the High Performance I/O (HPIO) banks located at the bottom of the device (banks 3, 4, 5). These banks are labeled as HIGHSPD in the device pinout tables. Each of these banks consists of 3-4 HPIO DQS groups, but depending on the device package, the number of HPIO DQS groups within each bank could be fewer. These HPIO DQS groups are labeled as DQx and DQSx/DQSNx in the device pinout tables, where 'x' represents the assigned HPIO DQS group number. Dedicated clock routing within HPIO banks is represented as GPLL or PCLK, and dedicated reference voltage pins are represented as VREF, in the device pinout tables. Refer to the High-Speed I/O User Guide and Pinout files located on the CertusPro-NX and MachXO5T-NX web pages on [www.latticesemi.com](http://www.latticesemi.com) for more information.

Observe the following guidelines when placing pins for external memory interfaces:

- Ensure that pins for external memory interfaces reside within HPIO banks at the bottom of the device.
- An external memory interface can occupy one or more banks. When an interface occupies multiple banks, it is recommended to occupy banks that are adjacent to one other in order to minimize timing and routing. Banks 5 and 3 are adjacent to bank 4, but bank 5 is not adjacent to bank 3.
- The DQS signals are fixed to specific locations (DQS/DQSN) within each HPIO bank. The DQS\_P signal is required to be placed at these locations within the HPIO DQS group. The DQS\_N signal will be auto placed and should not be manually assigned.
- All associated Data (DQ) and Data Mask (DM) signals belonging to a Data Strobe (DQS), must be placed in the same HPIO DQS group. Typically, a LPDDR4 DQS group consists of 8 DQ signals, 1 DM signal, and 1 DQS/DQSN pair. This means a HPIO DQS group needs 11 pins to support an LPDDR4 DQS group.

- The input reference clock to the PLL must be assigned to use dedicated clock routing (GPLL or PCLK). It is recommended to place the input reference clock on a dedicated PLL pin (GPLL) within the HPIO bank for better performance and to minimize jitter and routing.
- At least one VREF pin per HPIO bank that is used to implement an external memory interface, must be available and used as a reference voltage input.

Proposed pinouts should always be tested in Lattice Radiant software with correct I/O standards before finalizing.

## 7.2.2. Constraints

To ensure proper design coverage and hardware functionality, users must include the following necessary constraints in their LPDDR4 Memory Controller for Nexus IP project.

**Table 7.3. Project Constraints**

File Name	Description	Action Required
Memory Controller IP LDC file: constraints/<instance_name>.ldc	Sets the I/O type for each of the ports necessary to interface with LPDDR4 SDRAM	No
Clock Constraint SDC file: eval/clock_constraint.sdc	Contains an example constraint for the input PLL reference clock	Yes – user needs to include a create_clock constraint based on the frequency for the input PLL reference clock. This can be placed in a user-created SDC or PDC file.
Memory Controller IP PDC file: eval/constraint.pdc	Contains generated constraints based on IP configuration	Yes – user needs to copy the constraints listed in this file directly into their top-level PDC file. An explanation for each group of constraints to be copied is included within the eval/constraint.pdc file.

### 7.2.2.1. Clock

The provided clock\_constraint.sdc file contains a single create\_clock constraint for the PLL reference clock (pll\_refclk\_i). It is recommended that users copy this constraint into their own SDC or PDC file since the provided clock\_constraint.sdc file will be overwritten every time the Memory Controller IP Core is regenerated.

For clocks that are generated from the user's design, external to the IP, create\_generated\_clock constraints may be needed. In some cases, clocks or generated clocks don't need to be defined/constrained since they are automatically-generated (i.e. for output of PLL or OSC). For additional information on how to implement constraints, refer to the [Lattice Radiant Timing Constraints Methodology](#) User Guide.

### 7.2.2.2. Memory Controller IP

The provided constraint.pdc file contains two sets of constraints:

- IP constraints – specific to the Memory Controller IP Core
- Eval constraints – specific to the Memory Controller Example Design

For implementation of the generated Memory Controller Example Design, users need to copy the IP and Eval constraints into their top-level user PDC file. However, if users are implementing their own Memory Controller design, only the IP constraints need to be copied into the top-level user PDC file. It is recommended that users copy these constraints into their own PDC file since the provided constraint.pdc file will be overwritten everytime the Memory Controller IP Core is regenerated.

The IP constraints are composed of create\_generated\_clock, set\_false\_path, ldc\_create\_group, and set\_max\_delay constraints. The eval constraints are composed of set\_false\_path, set\_max\_delay, and ldc\_create\_group constraints and should only be copied if running the provided Memory Controller Example Design. Eval constraints are located below the following comment in the provided constraint.pdc file:

```
#####
```

```
# Below are the constraints for eval design, you dont need these if you are not using the eval
```

Refer to the [Lattice Radiant Timing Constraints Methodology User Guide](#) for details regarding the implementation of constraints.

### 7.3. Example Design Hardware Evaluation

After successfully configuring and generating the Memory Controller IP Core, the included synthesis example design can be used for hardware evaluation of the LPDDR4 Memory Controller. For a detailed description of the LPDDR4 Memory Controller Example Design, refer to the [Synthesis Example Design](#) section of this User Guide. The traffic generator files are located under the eval/traffic\_gen directory and are described in the following table.

**Table 7.4. Contents of eval/traffic\_gen**

File List	Description
ahbl0.v	AHBL_1x2. It routes CPU data access to SYS_MEM or AHBL2APB
ahbl2apb0.v	AHBL to APB bridge
apb0.v	APB_1x4. It routes CPU data access via AHBL2APB going to each module's CSR
cpu0.v	RISC-V CPU
gpio0.v	GPIO module
ahbl_tragen.v	RTL files for AHB-Lite traffic generator (IP Core v1.x.x)
lsccl_ahbl_traffic_gen.v,	
lsccl_ahb_master.v	
lsccl_traffic_gen_csr.v	
lsccl_lfsr.v	
memc_traffic_gen.v	RTL files for AXI4 traffic generator (IP Core v2.x.x)
ctrl_fifo.v	
lsccl_axi4_traffic_gen.v	
lsccl_axi4_m_csr.v	
lsccl_axi4_m_rd.v	
lsccl_axi4_m_wr.v	
lsccl_axi4_perf_calc.v	
mc_axi4_traffic_gen.v	
lsccl_osc.v	RTL files for OSC module
osc0.v	
lsccl_ram_dp_true.v	Copy of Lattice Radiant RAM_DP_True Foundational IP (needed by SYS_MEM)
memc_apb.v	RTL file for APB configuration interface
sysmem0.v	The SYS_MEM for hardware validation, enabled when eval_top.SIM=0 (Implementation)
sysmem0_sim.v	The SYS_MEM for RTL simulation, enabled when eval_top.SIM=1 (Simulation)
uart0.v	The UART module

### 7.3.1. Preparing the Bitstream

After configuring and generating the Memory Controller IP Core, all associated example design files would have been created under the eval directory. Refer to [Table 7.2](#) for more details. The following steps illustrate how to prepare the Memory Controller Example Design project and generate the associated bitstream.

1. After generating the Memory Controller IP Core, the Radiant project should contain the <instance\_name>.ipx under the project's input files. If not, right-click on **Input Files** and select **Add > Existing File** under the **File List** tab in the lower-left corner of the Radiant window. This will open an **Add Existing File** dialog box. Navigate to the <instance\_directory>/<instance\_name> directory and select the <instance\_name>.ipx file. Click **Add**.
2. To add the top-level example design file to the project, right-click on **Input Files** and select **Add > Existing File**. This will open an **Add Existing File** dialog box. Navigate to the eval directory and select the eval\_top.sv file. Click **Add**.

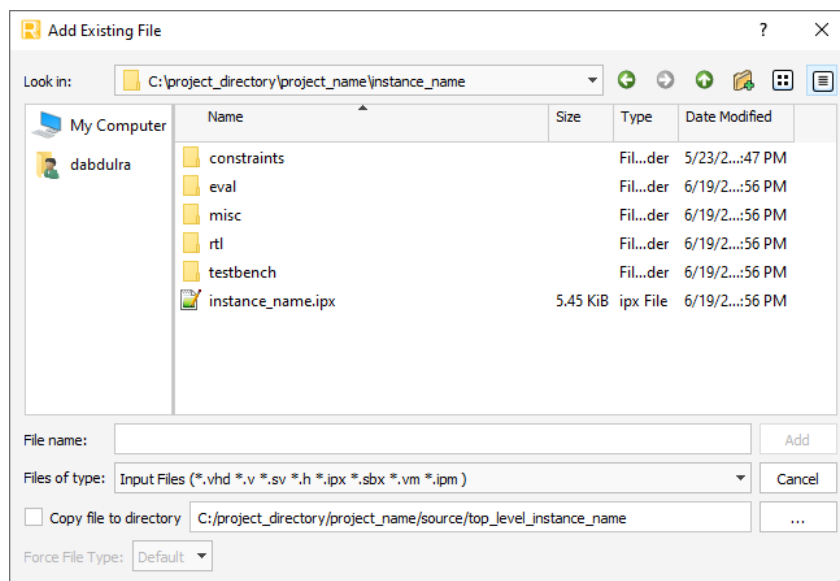


Figure 7.7. Add Existing File Dialog Box

3. To add the pre-synthesis constraint file to the project, right-click on **Pre-Synthesis Constraint Files** and select **Add > Existing File**. In the **Add Existing File** dialog box, check the **Copy file to directory** option. This ensures any user modifications is not overwritten when the Memory Controller IP Core is regenerated. Navigate to the eval directory and select the clock\_constraint.sdc file. Click **Add**.
4. To add the post-synthesis constraint file to the project, right-click on **Post-Synthesis Constraint Files** and select **Add > Existing File**. In the **Add Existing File** dialog box, check the **Copy file to directory** option. Navigate to the eval directory and select the constraint.pdc file. Click **Add**.
5. Modify the constraint.pdc to add the pin assignment for the CertusPro-NX or MachXO5T-NX board. Users can accomplish this by either modifying the constraint.pdc file directly or first synthesizing the Radiant project by clicking on the **Synthesize Design** button, and then adding pinouts via the Device Constraint Editor under **Tools > Device Constraint Editor**. Note that when assigning the UART pins, the UART TX signal is connected to the UART RX on the FPGA side and that the UART RX signal is connected to the UART TX signal on the FPGA side.
6. Before running the example design on hardware, a bitstream needs to be generated. This can be accomplished by clicking on the button. This will generate a <project\_name>\_<top\_level\_instance\_name>.bit file under the <project\_directory>/<project\_name>/<top\_level\_instance\_name> directory specified in Step 2.


Users may sometimes encounter timing failures during Place & Route due to unconstrained paths specific to their design. For details on implementing constraints, refer to the [Constraints](#) section of this User Guide.

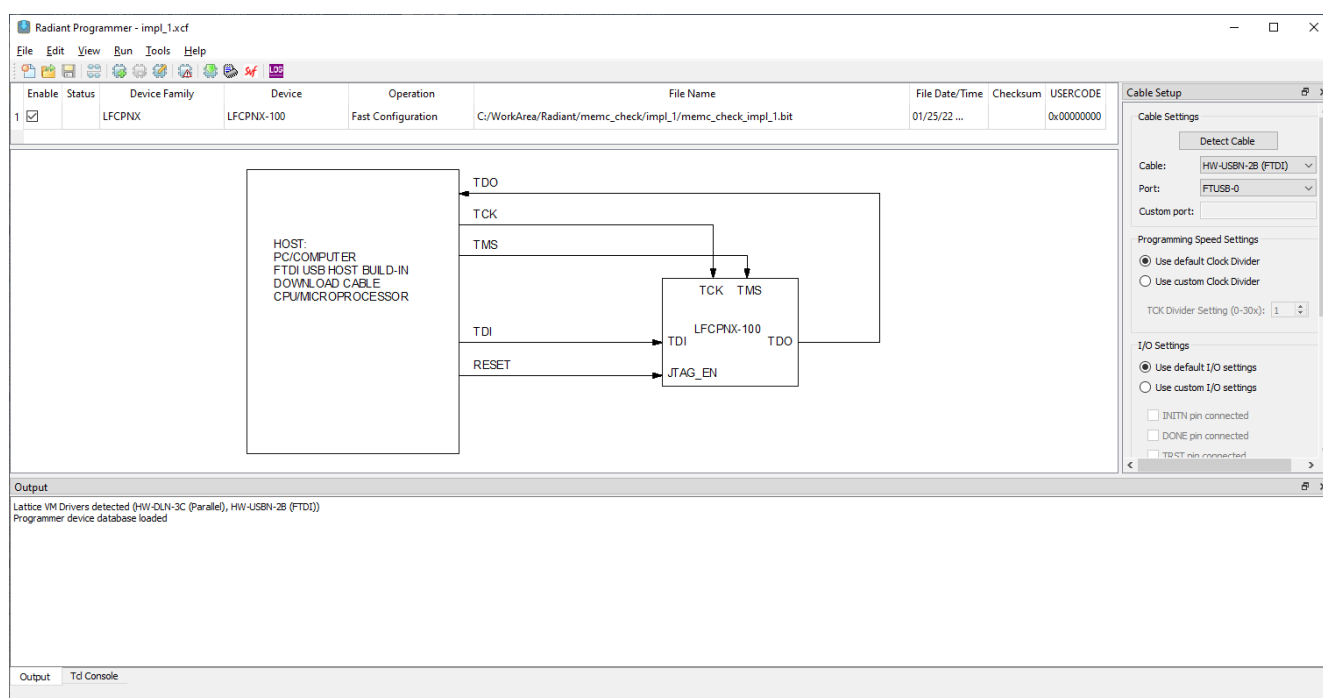
### 7.3.2. Running on Hardware

To perform hardware evaluation of the LPDDR4 Memory Controller Example Design, the following are needed:



- CertusPro-NX or MachXO5T-NX FPGA board with UART connection
- Associated power supply and programming cable
- Personal computer running Lattice Radiant software 2023.1 or later
- Lattice Propel™ software 1.0 or later, or any terminal that supports serial communication

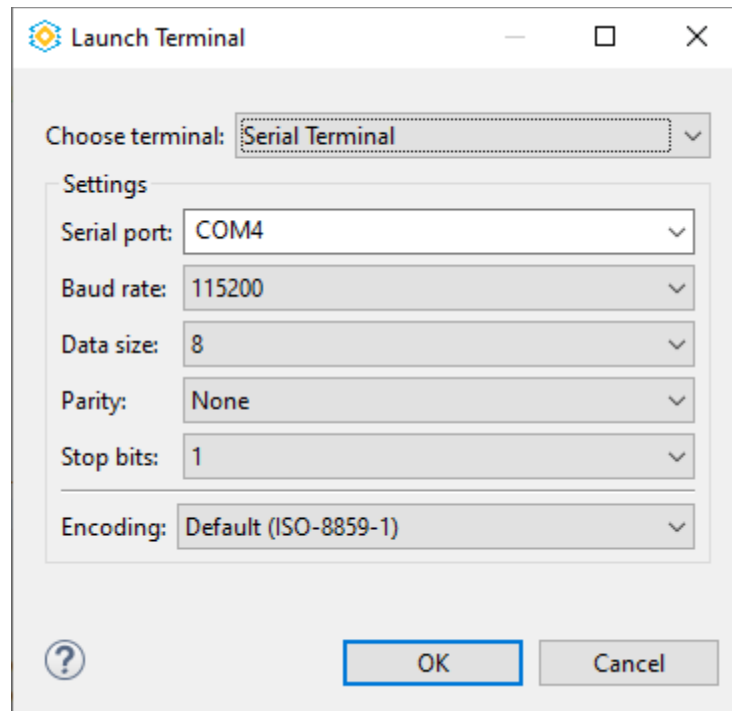
To run the example design on hardware, a bitstream file is required. To generate the .bit file, refer to the [Preparing the Bitstream](#) section of this User Guide. The following steps illustrate how to program the FPGA board with the example design.

1. Connect the FPGA board to the computer and power on the board.
2. Run the Lattice Radiant Programmer by clicking on the  button. This will launch the Lattice Radiant Programmer which will scan for devices and configure the programmer automatically.



**Figure 7.8. Radiant Programmer**

3. Click under the **File Name** field and then click on ... to the right of the field in order to launch the **Open File** dialog box. Navigate to the bitstream file generated in Step 15 and click **Open**.
4. Program the Lattice FPGA device by clicking on the  button. Upon successful programming, the **Output** pane at the bottom of the Programmer window will display the following message:
  - After programming the Lattice FPGA with the example design bitstream, a serial terminal needs to be launched. For users wishing to use their own serial communication terminal, skip to Step 25. For users wishing to use the Lattice Propel terminal, continue with Step 23.
5. Launch the Lattice Propel software and select **Launch**. This will open the default workspace.
6. To open a terminal, click on the  button.
7. Configure the terminal settings to be a **Serial Terminal** with the appropriate **Baud rate**, **Data size**, **Parity**, **Stop bits**, and **Encoding**. Click **OK** to launch the **Terminal** pane at the bottom of the Propel window. Note that the **Serial port** will vary depending on the computer setup.



**Figure 7.9. Serial Terminal Settings**

To run the example design, assert the `rstn_i` signal. This will prompt the following message to appear in the terminal. If no message is received, close the current serial terminal and open a new one with a different **Serial port**.

Press 1 to enter initial Vref values or press any key to proceed Training

Pressing 1 allows users to try different VREF values for their board/design since VREF training is not yet supported in the Memory Controller IP Core. This feature is available to users to help debug and test various VREF settings. For more information on how to debug a VREF-related failure, refer to the [Debug with the Example Design](#) section of this User Guide. The VREF values set here directly correlate to the VREF attributes defined in [Table 3.9](#). Pressing any other key results in the default VREF values being set for these attributes as depicted below.

Enter MC DQS Grp Vref (3'd): 70

Enter Memory CA Vref (3'd): 80

Enter Memory DQ Vref (3'd): 80

MC DQS Grp Vref = 70, Memory CA Vref = 80, Memory DQ Vref = 80

After the user provides input over the serial terminal, training of the LPDDR4 SDRAM will begin followed by 7 different data access checks:

- Test 0: single write followed by single read
- Test 1: 2-beat incrementing burst write followed by burst read
- Test 2: 4-beat incrementing burst write followed by burst read
- Test 3: 8-beat incrementing burst write followed by burst read
- Test 4: 8-beat incrementing burst write followed by a delay before issuing burst read
- Test 5: 64-beat incrementing burst write followed by burst read
- Test 6: 64-beat incrementing parallel burst write and burst read to measure performance

The results from the data access checks and performance measurement are then printed over the serial connection.

```
Training Passed.
Starting Data Access Check.
0 1 2 3 4 5 6
Performance Values :
Bus_efficiency : 80 Perf_MBps : 27261
```

```
Data Access Check Pass.
```

The Bus\_efficiency value represents the efficiency percentage of the LPDDR4 bus utilization, whereas the Perf\_MBps value represents the bandwidth of the LPDDR4 interface. The following formulas are used to calculate the efficiency and bandwidth:

Efficiency = (Total number of bytes transferred) / (# of DDR clock cycles × (DDR\_WIDTH / 8) × gearing ratio)

Bandwidth = (Total number of bytes transferred) / (# of DDR clock cycles × DDR clock period)

The bandwidth can also be calculated using the following equation: LPDDR4 Data Width × LPDDR4 Data Rate × (LPDDR4 Bus Efficiency / 100), where the LPDDR4 Data Rate is in Mbps.

In the case that a failure is encountered during training, a message will be sent over the serial connection notifying the user of the particular stage that has failed. This will also abort the loop-back data access checks.

```
Write Training Failed!
Aborting Data Access Check...
```

## 7.4. Example Design Simulation

After successfully configuring and generating the Memory Controller IP Core, the included example design can be used to simulate the LPDDR4 Memory Controller. All associated simulation files are located under the testbench directory. Refer to [Table 7.2](#) for more details. The following steps illustrate how to prepare the Memory Controller Example Design project for simulation.

1. Before simulating the example design, steps 13-14 under the Preparing the Bitstream section of this User Guide must be completed. To add the top-level testbench file to the project, select **File > Add > Existing Simulation File**. This will open an **Add Existing Simulation File** dialog box. Navigate to the testbench directory and select the tb\_top.sv file. Click **Add**.
2. Before creating the simulation environment, it is recommended to set the SIM parameter in eval\_top.sv to 1. This parameter shortens the initialization sequence of the LPDDR4 interface and disables the UART interface for simulation. Users should not modify the SIM parameter for hardware implementation.

When SIM is set to 1, it programs 0x1E to the Training Operation Register (TRN\_OP\_REG) to speed up the simulation runtime by reducing the reset and CKE initialization time. Users can configure the simulation of reset and the initialization and training sequences by forcing the value of TRN\_OP\_REG by locating the following comment in tb\_top.sv:

```
// This force shortens the initialization, vref trainings are also skipped
// force tb_top.u_eval_top.u_lp4mc_0.lscclpddr4_mc_inst.u_trn_eng.i_csr.trn_operation_reg
= 8'h1E;
```

It is also recommended to skip the training sequences by writing 8'h00 to TRN\_OP\_REG/trn\_opr\_i. This will reduce simulation time by programming the verified trained values to the PHY. Refer to the [Simulation Example Design](#) section of this User Guide for more details.

To create the simulation environment to simulate the LPDDR4 Memory Controller Example Design, the following steps should be taken:

1. Launch the simulation wizard in Radiant by selecting **Tools > Simulation Wizard**. This will open the **Simulation Wizard** dialog box. Click **Next** and provide a name (<sim\_name>) and directory (<sim\_directory>) for the simulation project, where the default directory is set to <project\_directory>/<project\_name>. Click **Next**.

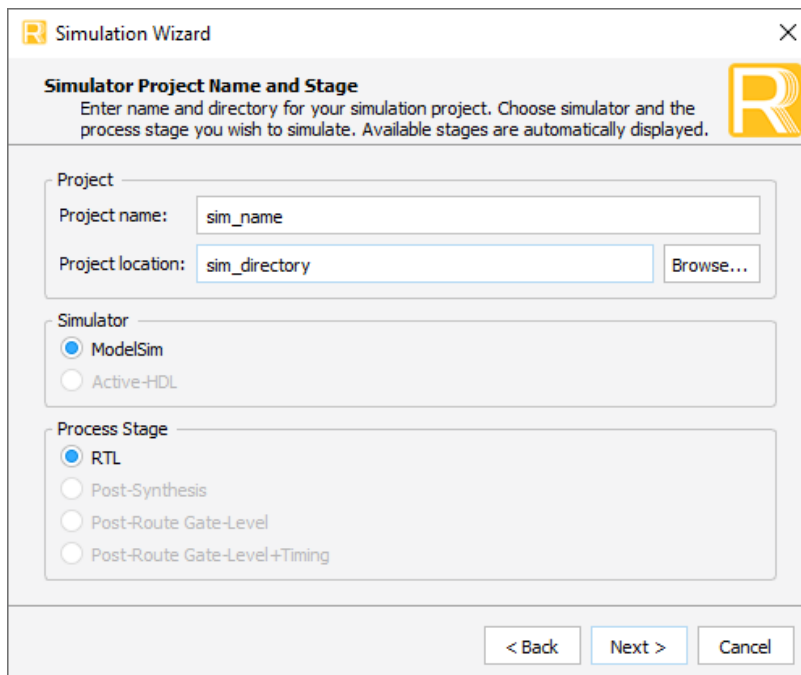


Figure 7.10. Simulation Wizard

2. Under the **Add and Reorder Source** window, notice that the **Source Files** only contains the top-level evaluation (eval\_top.sv) and top-level testbench (tb\_top.sv) files. Click **Next**.

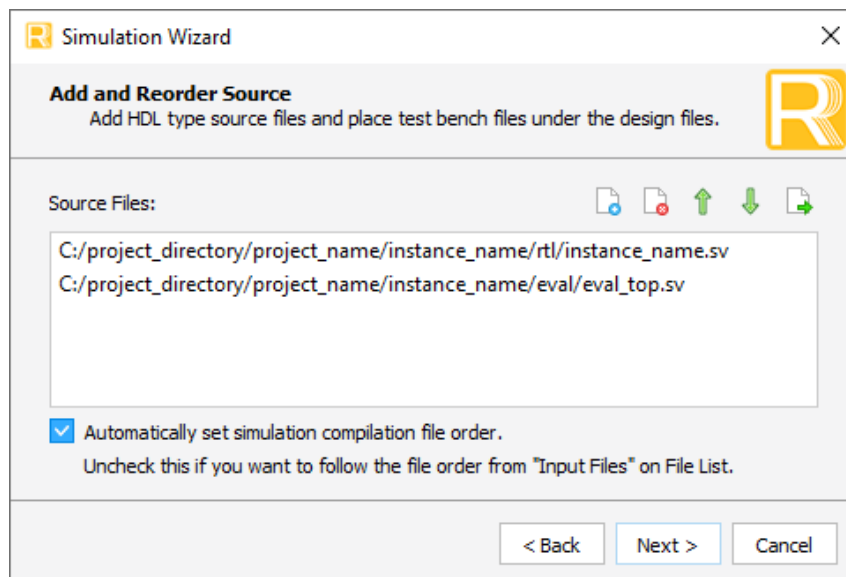
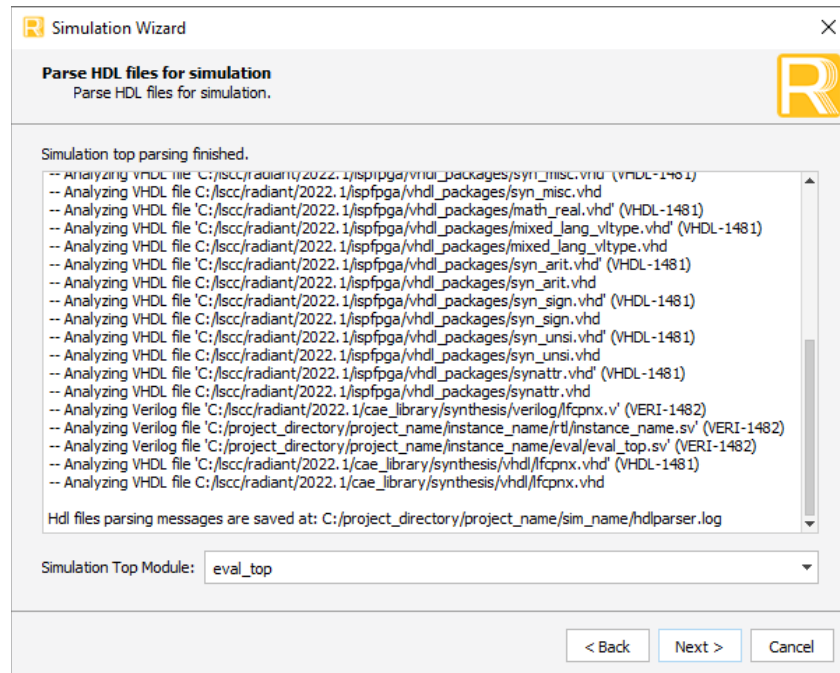


Figure 7.11. Adding and Reordering Simulation Source Files

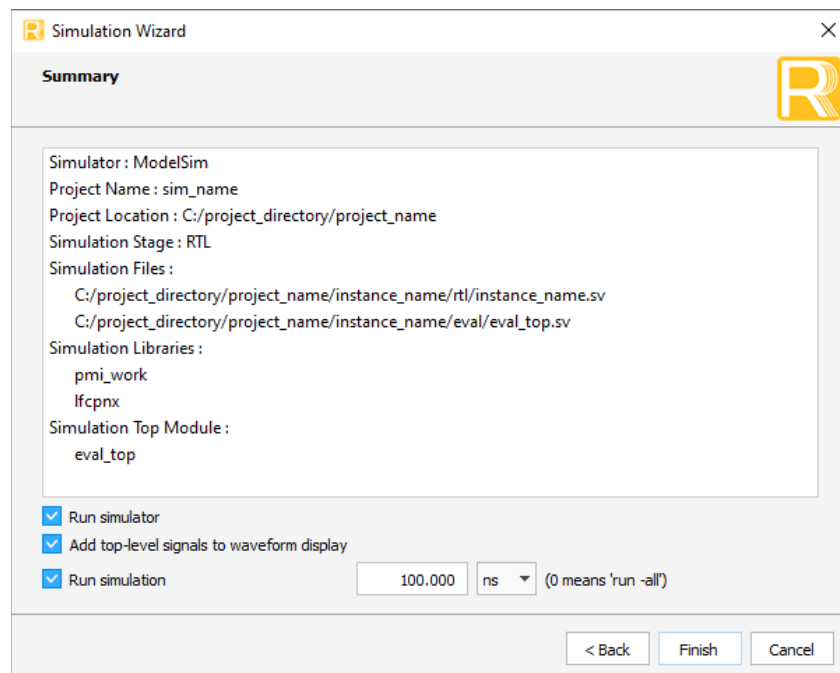


- Under the **Parse HDL files for simulation** window, notice that the **Simulation Top Module** is set to eval\_top. Click **Next**.



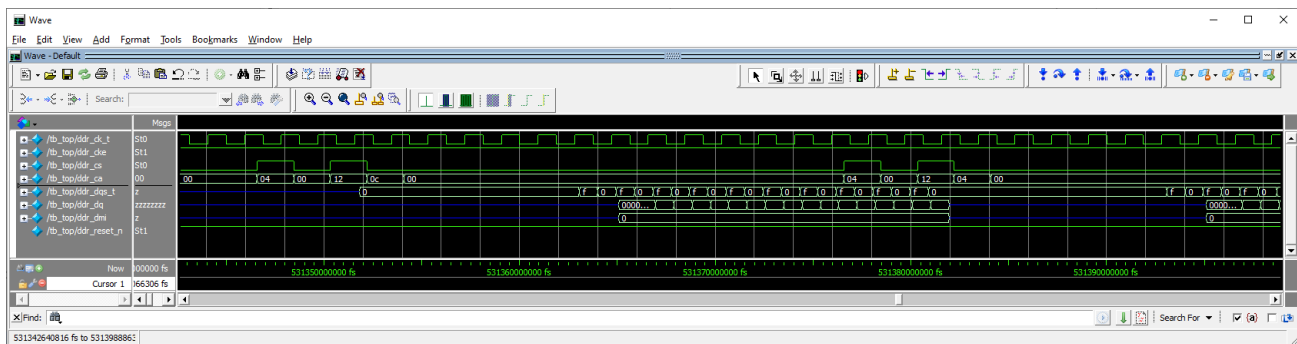
**Figure 7.12. Parsing Simulation HDL Files**

- This opens the **Summary** window. By default the simulation will run for 100  $\mu$ s, which allows users to configure the waveform to log signals of interest in the Modelsim simulator before continuing. Users can then enter the following TCL command in Modelsim to run the simulation until completion: **run -all**. Alternatively, if users wish to run the simulation with the default top-level signals, users can change the 100  $\mu$ s value to 0  $\mu$ s in order to execute the simulation completely.



**Figure 7.13. Simulation Summary**

The results of the Memory Controller IP simulation design are shown in the following figure.



**Figure 7.14. Simulation Result Waveform**

The following error messages are expected in the Modelsim simulation log and should be disregarded. These messages are due to the violation of timing requirements regarding the LPDDR4 simulation model as a consequence of shortening the reset and CKE initialization:

```
# tb_top.LP4MEM_00.mem_x16_00.ins_1ch.<protected> 26083125000 : ### lpddr4_debug
RESET_n high input
# tb_top.LP4MEM_01.mem_x16_01.ins_1ch.<protected> 26083125000 : ### lpddr4_debug
RESET_n high input
# Error: tb_top.LP4MEM_01.mem_x16_01.ins_1ch.<protected>.<protected> 26083125000 tINIT1
Error.
# Error: tb_top.LP4MEM_00.mem_x16_00.ins_1ch.<protected>.<protected> 26083125000 tINIT1
Error.
# tb_top.LP4MEM_00.mem_x16_00.ins_1ch.<protected> 32804075000 : ### lpddr4_debug CKE
high input
# tb_top.LP4MEM_01.mem_x16_01.ins_1ch.<protected> 32804075000 : ### lpddr4_debug CKE
high input
# Error: tb_top.LP4MEM_01.mem_x16_01.ins_1ch.<protected>.<protected> 32804075000 tINIT3
Error.
# Error: tb_top.LP4MEM_00.mem_x16_00.ins_1ch.<protected>.<protected> 32804075000 tINIT3
Error.
```

## 8. Debugging

This section discusses tools and strategies available to users to assist with debugging their LPDDR4 memory interface.

### 8.1. Debug with the Example Design

The provided LPDDR4 example design can serve to help debug functional issues regarding the training of external LPDDR4 memory or data accesses issued by the Memory Controller. For a description of the example design and instructions on how to run the test program for hardware evaluation, refer to the [Synthesis Example Design](#) and [Running on Hardware](#) sections of this User Guide.

Currently, VREF training is not yet supported in the LPDDR4 Memory Controller for Nexus Devices. As a result, it is possible that the default values could lead to training and data access failures due to board and PVT variations. In the case that a failure is encountered, users may rerun the test program included in the example design for different VREF values. This can be achieved by asserting the `rstn_i` signal and running the test program for different values until the example design reliably passes on the board. Once optimal VREF settings are found, it is suggested that users regenerate the Memory Controller IP Core with the new settings in order to make them the new defaults. For guidance on the correlation between the different VREF attributes and functional issue, refer to the following:

- MC DQS Grp Vref setting: can help to resolve read training and data access failures
- Memory DQ Vref setting: can help to resolve write training and data access failures
- Memory CA Vref setting: can help to resolve command bus training failures, although highly unlikely since the max command bus does not operate at a high frequency

### 8.2. Debug with Reveal Analyzer

The LPDDR4 Memory Controller for Nexus Devices introduced reveal signals to assist users in debugging issues regarding the training sequence of the LPDDR4 memory interface using Reveal Analyzer. This feature was introduced in IP Core v2.1.x and exists within the `rtl/<instance_name>.sv` file following successful IP generation. When the user data interface is set to the AHB-Lite protocol, the reveal signals can be located under the following comment in the generated IP RTL file:

```
// For reveal debugging
```

When the user data interface is set to the AXI4 protocol, the reveal signals can be located under the following comment in the generated IP RTL file:

```
// Reveal tap points for AXI INTERFACE
```

Refer to the [Debugging with Reveal Usage Guidelines and Tips Application Note](#), for information regarding the setup and usage of Reveal Analyzer. The following table covers the list of reveal signals available to users for debug.

**Table 8.1. Reveal Analyzer Signal Definitions**

Signal Name	Description
<code>rsl_cs_r</code>	Chip select
<code>rsl_cbt_en_r</code>	Asserted high during Command Bus Training (CBT)
<code>rsl_ca_adj_cout_r</code>	CA delay per bit adjustment cout signal
<code>rsl_ca_adj_dir_r</code>	CA delay per bit adjustment direction signal
<code>rsl_ca_adj_load_n_r</code>	CA delay per bit adjustment load signal
<code>rsl_ca_adj_move_r</code>	CA delay per bit adjustment signal
<code>rsl_cbt_dq_fbak_r</code>	DQ[13:8] output pins to feedback captured value to the Memory Controller during CBT
<code>rsl_wrlvl_en_r</code>	Asserted high during Write Leveling
<code>rsl_wrlvl_load_en_r</code>	Asynchronously resets the delay code to the default value for CK-DQS skew compensation during Write Leveling

Signal Name	Description
rvl_wrlvl_move_r	At rising edge, it changes the delay code ( $\pm 1$ ) according to the direction set by wrlvl_dir_i for CK-DQS skew compensation during Write Leveling
rvl_wrlvl_dir_r	Controls the direction of delay code change for CK-DQS skew compensation during Write Leveling (0 = increase delay, 1 = decrease delay)
rvl_wrlvl_cout_r	Margin test output flag to indicate the under-flow or over-flow for CK-DQS skew compensation during Write Leveling
rvl_burst_det_sclk_r	Clock generated using burst_det_o
rvl_rd_clkssel_r	Used to select read clock source and polarity control: [1:0]: sets phase shift from DQS write delay cell to 0, 45, 90, or 135 degrees (2'b00 to 2'b11) [2] = 0: use inverted clock [2] = 1: use non-inverted clock (adds 180 degree phase shift) [3] = 0: bypasses the register in the read enable path [3] = 1: selects the register in the read enable path
rvl_read_dqs_ie_r	Read enable signal for capturing the incoming read BL16. Each bit captures the DQ/DMI for the corresponding ddr_ck_o cycle.
rvl_pause_r	Set to 1 to stop the DQSBUF-generated internal clocks when updating rvl_rd_clkssel_r and the delay codes. This is to avoid metastability.
rvl_dqwl_r2	Data output of Write Leveling
rvl_dqdm_i_load_n_r	DMI input delay per bit adjustment load signal
rvl_dqdm_i_move_r	DMI input delay per bit adjustment move signal
rvl_dqdm_i_dir_r	DMI input delay per bit adjustment direction signal
rvl_dqdm_i_cout_r	DMI input delay per bit adjustment cout signal
rvl_dqdm_o_load_n_r	DMI output delay per bit adjustment load signal
rvl_dqdm_o_move_r	DMI output delay per bit adjustment move signal
rvl_dqdm_o_dir_r	DMI output delay per bit adjustment direction signal
rvl_dqdm_o_cout_r	DMI output delay per bit adjustment cout signal
rvl_rd_comp_result_r	Results of compared data for each DQS group: [0] = 1: DQS group 0 comparison passes [1] = 1: DQS group 1 comparison passes [2:7] = 1: DQS group 2/3/4/5/6/7 comparison passes This is only valid when rvl_rd_comp_done_r is asserted.
rvl_rd_comp_done_r	Asserted high when compared data is done.
rvl_trn_stat_done	{write_trn_done, read_trn_done, write_lvl_done, cbt_done, phy_ready}
rvl_trn_stat_err	{write_trn_err, read_trn_err, write_lvl_err, cbt_err}
rvl_scratch_0_r	Registers used by the training CPU to write debug information
rvl_scratch_1_r	Registers used by the training CPU to write debug information

For general purpose debug, the rvl\_trn\_stat\_done and rvl\_trn\_stat\_err signals can be used to determine if the training of the LPDDR4 interface has passed or failed. The rvl\_trn\_stat\_done signal contains multiple status signals that are concatenated, to indicate that a particular stage of training has completed. The rvl\_trn\_stat\_err signal contains multiple status signals that are concatenated, to indicate whether an error has occurred during a particular stage of training. Examples are included in [Figure 8.1](#) and [Figure 8.2](#).

Bus/Signal	Data	0:1024	0:2048	0:3072	0:4096	0:5120	0:6144	0:7168	0:8192	0:9216	0:10240	0:11264	0:12288	0:13312	0:14336	0:15360
...inst/rvl_trn_stat_done	1F															
...inst/rvl_trn_stat_err	0000															

Figure 8.1. Reveal Example: LPDDR4 Training Passes

Bus/Signal	Data	0:1024	0:2048	0:3072	0:4096	0:5120	0:6144	0:7168	0:8192	0:9216	0:10240	0:11264	0:12288	0:13312	0:14336	0:15360
...mc_inst/rvl_trn_stat_done	01															
...r4_mc_inst/rvl_trn_stat_err	0001															

Figure 8.2. Reveal Example: Command Bus Training Failure

### 8.2.1. Command Bus Training (CBT)

The Memory Controller begins Command Bus Training (CBT) by sending CA pattern 0x19 to the LPDDR4 memory device, delaying the CA bus, and using DQ[13:8] to provide feedback on the captured value. CBT is performed when the `rvl_cbt_en_r` signal is asserted high. The upper byte of the `rvl_dqwl_r2` signal acts as the feedback mechanism, and the lower byte correlates to the CA VREF level and range setting. DQ[13:8] is mapped to the `rvl_cbt_dq_fbak_r` signal, which notifies the Memory Controller whether CBT has passed for the current delay setting. Once 0x19 is read on the `rvl_cbt_dq_fbak_r` signal, it indicates CBT has passed. This sequence is repeated until the optimal setting, providing the best margin, is found. For more information on CBT, refer to the [Command Bus Training \(CBT\) Description](#) section of this User Guide.

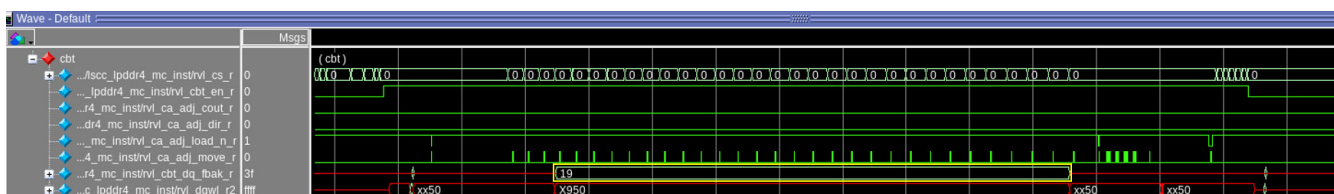


Figure 8.3. Command Bus Training Simulation Waveform

Bus/Signal	Data	0:1024	0:2048	0:3072	0:4096	0:5120	0:6144	0:7168	0:8192	0:9216	0:10240
...lpddr4_mc_inst/rvl_cs_r	0000	0000	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX
...dr4_mc_inst/rvl_cbt_en_r	1										
...c_inst/rvl_ca_adj_cout_r	0										
...mc_inst/rvl_ca_adj_dir_r	0										
...inst/rvl_ca_adj_load_n_r	1										
...c_inst/rvl_ca_adj_move_r	0										
...c_inst/rvl_cbt_dq_fbak_r	00	00	19	3B							
...dr4_mc_inst/rvl_dqwl_r2	0000	0050	1950	3850	3850						

Figure 8.4. Command Bus Training Reveal Capture

### 8.2.2. Write Leveling

Write leveling begins with the LPDDR4 memory device sampling the DDR clock with the rising edge of DQS, and asynchronously providing feedback to the Memory Controller via the DQ pins. Write leveling is performed when `rvl_wrlvl_en_r` is asserted high. The `rvl_dqwl_r2` signal acts as the feedback mechanism to indicate whether write leveling has passed for the current delay setting. Write leveling starts with a coarse delay adjustment for all DQS groups, followed by a fine delay adjustment for each of DQS group. When all bits in the `rvl_dqwl_r2` signal are set high, it indicates write leveling has passed. Note that if the first sample of the `rvl_dqwl_r2` signal has all bits set high, the user should increase the setting for the *DDR Clock Delay Value* attribute. For more information on write leveling, refer to the [Write Leveling Description](#) section of this User Guide.

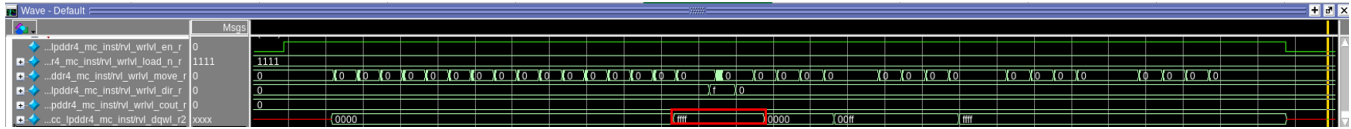


Figure 8.5. Write Leveling Simulation Waveform

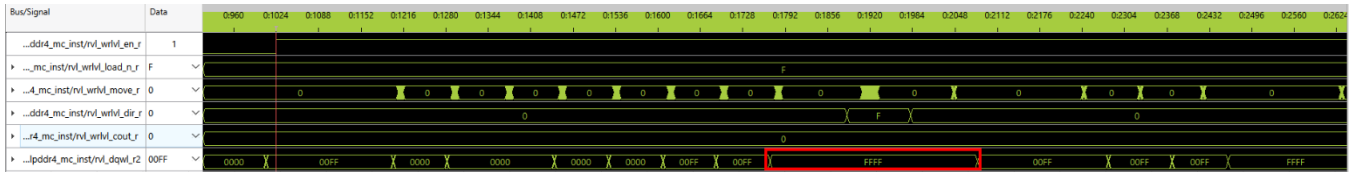


Figure 8.6. Write Leveling Reveal Capture

### 8.2.3. Read Training

Read training consists of two stages: read gate training (DQS) and read deskew (DQ).

Read gate training sweeps all the read DQS signals relative to the DDR clock until a full burst is able to be captured. The read\_dqs\_ie\_i signal corresponds to the DQS signals and the rd\_clkssel\_i signal corresponds to the DDR clock. Once the read preamble and eight DQS pulses are captured, it indicates read gate training was successful. This is indicated when the burst\_det\_sclk signal is set high for four occurrences in a row. The final setting for the rd\_clkssel\_i signal is calculated as the average of the burst\_det\_sclk assertion window.

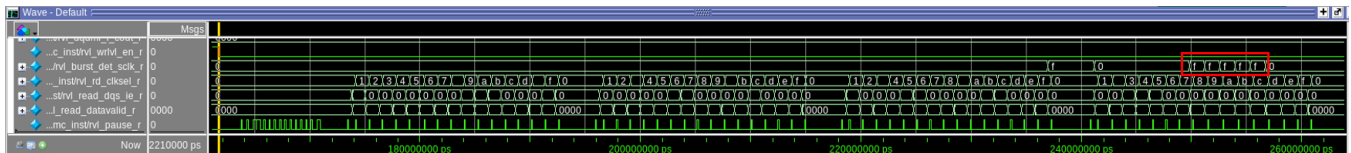


Figure 8.7. Read Gate Training Simulation Waveform

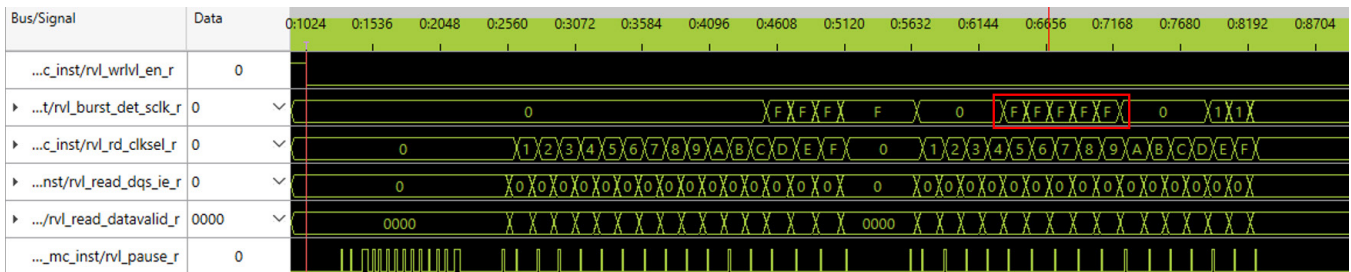


Figure 8.8. Read Gate Training Reveal Capture

Read deskew training begins with Multi-Purpose Command: MPC[READ DQ CALIBRATION]. It issues Mode Register Write (MRW) commands to MR32 and MR40 and then issues READ commands to see if the content read back on DQ is correct. If the DQ signal does not capture the correct data, it is delayed and the process is repeated. Once the read data is correct, the rvl\_rd\_comp\_result\_r signal is asserted high. The final DQ delay value is calculated after a passing window is achieved. For more information on read training, refer to the [Read Training Description](#) section of this User Guide.

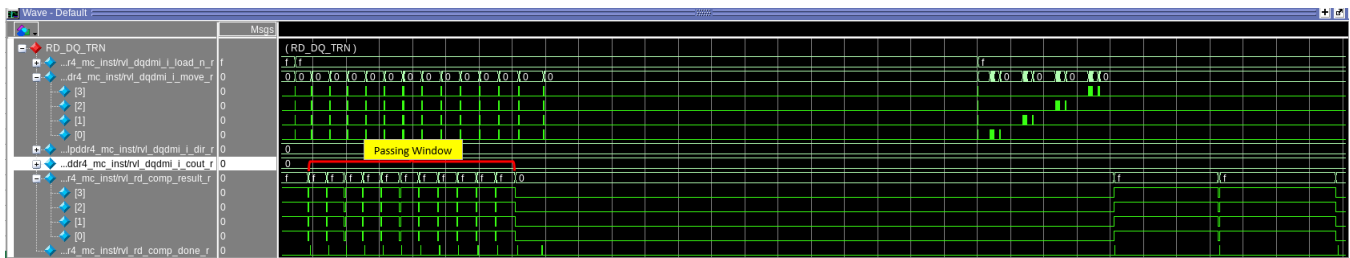


Figure 8.9. Read Deskew Training Simulation Waveform

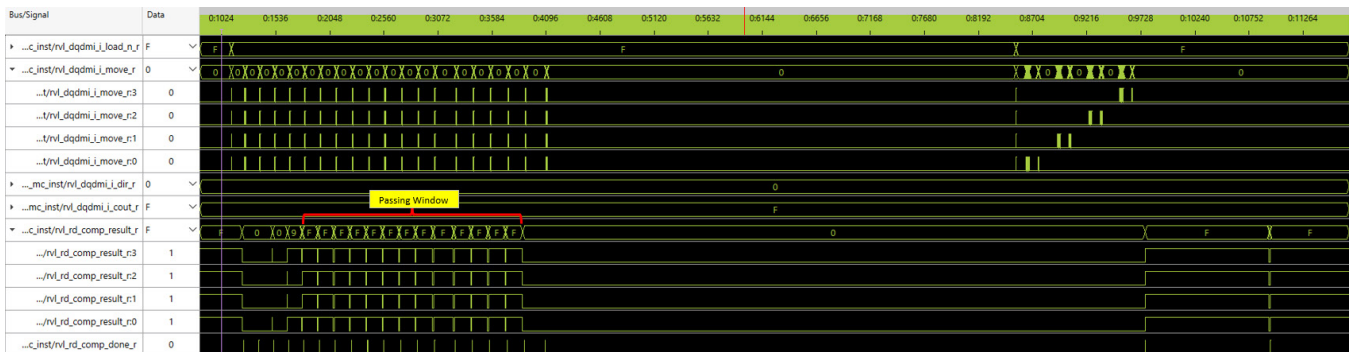


Figure 8.10. Read Deskew Training Reveal Capture

## 8.2.4. Write Training

Write training begins with Multi-Purpose Command: MPC[WRITE-FIFO]. It issues a set of 5 MPC[WRITE-FIFO] commands and 5 MPC[READ-FIFO] commands. Once data is written to the external LPDDR4 SDRAM, the data is read back on the DQ signal and compared with the “expected” data to determine if the read data is correct. If the DQ signal does not capture the correct data, it is delayed and the process is repeated. Once the read data is correct, the rvl\_rd\_comp\_result\_r signal is asserted high. The final DQ delay value is calculated after a passing window is achieved. For more information on write training, refer to the [Write Training Description](#) section of this User Guide.

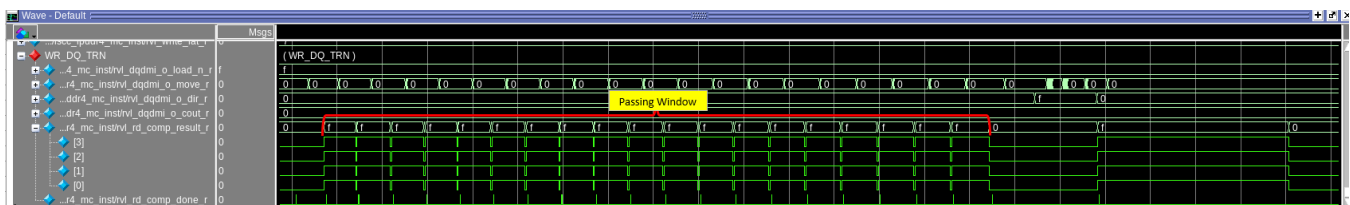


Figure 8.11. Write Training Simulation Waveform



Figure 8.12. Write Training Reveal Capture



## Appendix A. Resource Utilization

The following table shows the configuration and resource utilization for IP Core v1.3.0 implemented for LFCPNX-100-9FFG672I using LSE of Lattice Radiant software 2022.1.

**Table A.1 Resource Utilization for IP Core v1.3.0**

Configuration	sclk_o Fmax <sup>1</sup> (MHz)	Registers	LUTs	EBR	IDDR/ODDR/TDDR
Interface Type = AHBL Others = Default	143.761	6935	10396	18	121 (32+49+40)
Interface Type = AHBL, Enable DBI = Checked, Enable Power Down = Checked, Others = Default	141.423	7023	10969	18	125 (36+49+50)
Interface Type = AHBL, Enable Internal RISC-V CPU = Unchecked, Others = Default	140.905	6234	8987	8	121 (32+49+40)
Interface Type = AHBL, DDR Bus Width = 16, Others = Default	134.246	5296	8061	14	65 (16+29+20)
Interface Type = AHBL, DDR Bus Width = 64, Others = Default	124.719	10228	15526	26	233 (64+89+80)

Note:

1. The sclk\_o Fmax is generated using the top-level example design wrapper file, eval\_top.sv, that is described in the [Synthesis Example Design](#) section of this User Guide. These values may increase when the IP Core is used with the user logic.

The following table shows the configuration and resource utilization for IP Core v2.1.0 implemented for LFCPNX-100-9FFG672C using Synplify Pro of Lattice Radiant software 2023.1.

**Table A.2. Resource Utilization for IP Core v2.1.0**

Configuration	aclk_i Fmax (MHz)	sclk_o Fmax <sup>1</sup> (MHz)	Registers	LUTs	EBR	IDDR/ODDR/TDDR
DDR Bus Width = 16, Others = Default	219.202	154.273	7465	9725	25	65 (16+29+20)
Default	186.359	154.631	8927	10515	33	121 (32+49+40)
DDR Bus Width = 64, Others = Default	164.123	157.332	11894	12345	50	233 (64+89+80)
DDR Bus Width = 64, Enable Power Down = Checked, Enable DBI = Checked, Others = Default	159.591	169.693	12017	12864	50	241 (72+89+80)

Note:

1. The sclk\_o Fmax is generated using the top-level example design wrapper file, eval\_top.sv, that is described in the [Synthesis Example Design](#) section of this User Guide. These values may increase when the IP Core is used with the user logic.



## References

For more information refer to:

- [Lattice Radiant](#) FPGA design software
- [Lattice Insights](#) for Lattice Semiconductor training courses and learning plans
- [Lattice Sales Office](#) for more information about pricing and availability of the LPDDR4 Memory Controller for Nexus Devices
- [CertusPro-NX](#) web page
- [MachXO5-NX](#) web page
- [AMBA AHB Protocol Specification](#)
- [AMBA AXI Protocol Specification](#)
- [AMBA APB Protocol Specification](#)
- [LPDDR4 JEDEC Standard](#)
- [Lattice Radiant Timing Constraints Methodology User Guide \(FPGA-AN-02059\)](#)
- [Debugging with Reveal Usage Guidelines and Tips Application Note \(FPGA-AN-02060\)](#)
- [LPDDR4 Memory Interface Module User Guide \(FPGA-IPUG-02154\)](#)

## Technical Support Assistance

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

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

## Revision History

### Revision 1.5, September 2023

Section	Change Summary
All	<ul style="list-style-type: none"> <li>Changed the document title from <i>Memory Controller IP Core - Lattice Radiant Software</i> to <i>LPDDR4 Memory Controller for Nexus Devices</i>.</li> <li>Reworked document content and structure for clarity.</li> </ul>
Acronyms in this document	<ul style="list-style-type: none"> <li>Reworked section contents.</li> </ul>
Introduction	<ul style="list-style-type: none"> <li>Reworked section contents.</li> <li>Reworked <i>section 4 Ordering Part Number</i> and renamed to <i>subsection 1.3 Licensing and Ordering Information</i>.</li> <li>Added IP Validation Summary and Minimum Device Requirements subsection.</li> <li>Reworked <i>subsection 1.3 Conventions</i> and renamed to <i>subsection 1.6 Naming Conventions</i>.</li> </ul>
Functional Description	<ul style="list-style-type: none"> <li>Reworked <i>subsection 2.5. Submodules description</i> and renamed to <i>subsection 2.1 IP Architecture</i>.</li> <li>Added subsection <i>2.2 Clocking and Reset</i>.</li> <li>Reworked <i>subsection 2.2.1 AHB-Lite Interface</i> and moved under <i>subsection 2.3 User Interfaces</i>.</li> <li>Reworked <i>subsection 2.2.4 AXI4 Interface</i> and moved under added <i>subsection 2.3 User Interfaces</i>.</li> <li>Reworked <i>subsection 2.6. Initialization and Training</i> and renamed to <i>subsection 2.4 LPDDR4 Calibration</i>.</li> <li>Reworked <i>subsection 2.7. Operations Details</i> and renamed to <i>subsection 2.5 LPDDR4 Operation Description</i>.</li> </ul>
IP Parameter Description	Reworked <i>subsection 2.3 Attributes Summary</i> and moved this under <i>IP Parameter Description</i> section.
Signal Description	Reworked <i>subsection 2.2 Signal Description</i> and converted it to <i>Signal Description</i> section.
Register Description	Reworked <i>subsection 2.4 Register Description</i> and converted it to <i>Register Description</i> section.
LPDDR4 Memory Controller Example Design	Added this section.
Designing and Simulating the IP	Reworked <i>section 3 IP Generation, Simulation, and Verification</i> and renamed to this main section.
Debugging	Added this section.
Appendix A. Resource Utilization	Reworked section contents.
References	Reworked section contents.
Technical Support Assistance	Added FAQ link in this section.

### Revision 1.4, December 2022

Section	Change Summary
Acronyms in This Document	Added AXI4 and revised CBI.
Introduction	<ul style="list-style-type: none"> <li>Added AXI4 interface.</li> <li>In Table 1.1: <ul style="list-style-type: none"> <li>Removed Performance Grade</li> <li>Updated Supported User Interfaces and Resources in</li> </ul> </li> <li>In Features section: <ul style="list-style-type: none"> <li>Updated Supported Transactions and Command Frequency.</li> <li>Indicated future enhancements.</li> <li>Removed Dynamic On-Die Termination (ODT) controls.</li> <li>Added AXI4 I/F in Table 1.2.</li> </ul> </li> </ul>

Section	Change Summary
Functional Description	<ul style="list-style-type: none"> <li>In Overview section: <ul style="list-style-type: none"> <li>Removed <i>ODT for generating ODT</i>.</li> <li>Added AXI4 I/F.</li> </ul> </li> <li>In Table 2.1: <ul style="list-style-type: none"> <li>Added AXI4 I/F.</li> <li>Updated Clock and Reset group and Other Signals group.</li> <li>Corrected ahbl_htrans_i I/O.</li> </ul> </li> <li>Updated AHB-Lite Interface and Native Interface sections to indicate availability in IP Core v1.x.x.</li> <li>Added AXI4 Interface section.</li> <li>Added new attributes for IP Core v2.x.x in Table 2.6 and Table 2.7.</li> <li>Removed RefClock (MHz) Selectable Values from Table 2.6.</li> <li>In Register Description section: <ul style="list-style-type: none"> <li>Improve description of addr_translation field in FEATURE_CTRL_REG. Also indicated that num_ranks 1-Dual Rank is not supported. <ul style="list-style-type: none"> <li>Updated COLW Local Address map in Table 2.12.</li> <li>Updated description of RESET_REG and TRN_OP_REG due to Enable APB I/F attribute.</li> <li>Updated subsection headings.</li> </ul> </li> </ul> </li> <li>Indicated future enhancements in the Training Operation Register (TRN_OP_REG) (0x20) section.</li> <li>Removed the On-Die Termination Control section.</li> <li>Added Initialization and Training without APB I/F section.</li> <li>Updated interface in the Write and Read Data Access section. Also added reference to Table 2.5.</li> </ul>
IP Generation, Simulation, and Validation	<ul style="list-style-type: none"> <li>Updated the outline of this section.</li> <li>Updated Figure 3.1, Figure 3.2, Figure 3.3, Figure 3.4, and Figure 3.5 for IP Core v 2.0.0.</li> <li>Added Constraining the IP section.</li> <li>Updated Hardware Validation section for IP Core v 2.0.0. Updated step 8 in the procedure for running hardware evaluation and added information on VREF training support.</li> </ul>
Appendix A. Resource Utilization	Added Resource Utilization for IP Core v 2.0.0.
References	Added link to Lattice Radiant Software User Guide.
Technical Support Assistance	Added reference to the Lattice Answer Database on the Lattice website.
All	<ul style="list-style-type: none"> <li>Replaced <i>slave</i> with <i>subordinate</i> and <i>master</i> with <i>manager</i> when appropriate.</li> <li>Minor adjustments in formatting and style.</li> </ul>

### Revision 1.3, January 2022

Section	Change Summary
All	Removed DDR3 features across the document.
Introduction	<ul style="list-style-type: none"> <li>Updated Lattice Radiant software version in Table 1.1.</li> <li>Updated Features to add Native I/F, Memory DQ_VREF Training, Memory Controller DQ_VREF training, and updated Command frequency.</li> </ul>
Functional Description	<ul style="list-style-type: none"> <li>Updated Table 2.1 to remove hclk_i and hreset_n_i, add Native Interface, and update table note.</li> <li>Added Native Interface and Native Interface to AXI4 Bridge sections.</li> <li>Updated Error Log Register (ERROR_LOG_REG), Training Operation Register (TRN_OP_REG), and Status Register (STATUS_REG) section.</li> <li>Updated the following in Table 2.5: <ul style="list-style-type: none"> <li>DDR Command Frequency (MHz)</li> <li>Local Data Bus Type</li> </ul> </li> </ul>

Section	Change Summary
	<ul style="list-style-type: none"> <li>Data Width</li> <li>Periodic Event Setting Group</li> <li>Updated Local Interface group in Table 2.6.</li> <li>Separated Initialization and Training section from Operation Details section.</li> <li>Split REFRESH Support section into Auto Refresh Support and Power Saving Feature.</li> <li>Added Periodic ZQ Calibration and Temperature Tracking and Extended Temperature Support sections.</li> </ul>
Core Generation, Simulation, and Validation	<ul style="list-style-type: none"> <li>Updated Figure 3.1, Figure 3.2, and Figure 3.3.</li> <li>Updated Table 3.1.</li> <li>Updated Running Functional Simulation section to add the simulation support.</li> <li>Replaced Hardware Evaluation section with Hardware Validation.</li> </ul>
Ordering Part Number	Updated content to add DDR4 part number with one year subscription license and remove part numbers for DDR3.
Appendix A. Resource Utilization	<ul style="list-style-type: none"> <li>Updated Lattice Radiant software version to 3.1.</li> <li>Updated values in Table A.1.</li> </ul>

## Revision 1.2, June 2021

Section	Change Summary
All	Minor adjustments in formatting.
Introduction	Updated Features section content to correct acronym from INC4 and INC8 to <i>INCR4</i> and <i>INCR8</i> .
Functional Description	Improved description of <code>rst_n_i</code> in Table 2.1.
Core Generation, Simulation, and Validation	Updated Figure 3.1, Figure 3.2, and Figure 3.3.
Ordering Part Number	Added this section.
Appendix A. Resource Utilization	Updated Lattice Radiant software version to 3.0.
References	Updates section content to add CertusPro-NX webpage.

## Revision 1.1, March 2021

Section	Change Summary
Introduction	Updated Features section content to correct acronym from INC4 and INC8 to <i>INCR4</i> and <i>INCR8</i> .
Functional Description	<ul style="list-style-type: none"> <li>Updated Table 2.1 to add hreset and preset port name.</li> <li>Updated Table 2.2 to correct acronym from INC4 and INC8 to <i>INCR4</i> and <i>INCR8</i>.</li> <li>Updated Table 2.3 to correct Selectable Values, Default, and Dependency on Other Attributes.</li> <li>Updated Feature Control Register (FEATURE_CTRL_REG) section to correct num_rank bullet and [31:7] and [16] in Table 2.7.</li> <li>Updated Interrupt Status Register (INT_STATUS_REG) section to correct temp_change_int bullet and [4] in Table 2.12.</li> <li>Updated Interrupt Enable Register (INT_ENABLE_REG) section to correct temp_change_en bullet and [4] in Table 2.13.</li> <li>Updated Interrupt Set Register (INT_SET_REG) section to correct temp_change_set bullet and [4] in Table 2.14.</li> </ul>
Core Generation, Simulation, and Validation	Updated Figure 3.1 and Figure 3.2.
Appendix A. Resource Utilization	Updated values in Table A.1.

**Revision 1.0, February 2021**

Section	Change Summary
All	Initial release



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