EBR Memory Modules - Lattice Radiant Software

User Guide

FPGA-IPUG-02190-1.5

February 2025

Disclaimers

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

Contents

Figures

Tables

Abbreviations in This Document

A list of abbreviations used in this document.

Abbreviation	Definition
EBR	Embedded Block Random Access Memory
FPGA	Field-Programmable Gate Array
IP	Intellectual Property
LUT	Lookup-Table
PMI	Parameterized Module Instantiation
RTL	Register Transfer Level

1. Introduction

This user guide discusses EBR memory usage for devices supported by Lattice Radiant™ Software. It is intended to be used by design engineers as a guide to integrate EBR memory blocks with their applications.

In addition, behavioral inference for custom memory supported and the Synthesis tool is expected to infer the required components based on synthesis directive.

Designers can utilize the EBR memory primitives using two different methods described below.

- **Using Module/IP Block Wizard** – The Module/IP Block Wizard user interface allows user to specify the required memory type and size. Module/IP Block Wizard takes this specification and instantiates a synthesizable RTL (register transfer level) code and sets the appropriate parameters based on the user interface setting.
- **Using PMI (Parameterized Module Instantiation)** – PMI allows experienced users to skip the graphical user interface and utilize the configurable memory primitives on-the-fly from the Lattice Radiant Software project navigator. The user set the parameters, and the control signals needed in Verilog.

Table 1.1. Memory Modules Support

Modul Name	iCE40 UltraPlus™	Lattice Nexus™ Platform	Lattice Avant™
Single Port Memory (RAM_DQ)	✓	✓	✓
Pseudo Dual Port Memory (RAM_DP)	✓	✓	✓
True Dual Port Memory (RAM_DP True)	✓	✓	✓
Read Only Memory (ROM)	✓	✓	✓
Shift Register (Shift_Reg)	✓	✓	✓
Single Port Large RAM (Large RAM_DQ)	—	✓	—
Pseudo Dual Port Large RAM (Large RAM_DP)	—	✓	—
True Dual Port Large RAM (Large RAM_DP True)	—	✓	—
Large Read Only Memory (Large ROM)	—	✓	—

2. Memory Modules

The following sections discuss the different EBR memory modules available from the IP Catalog, the size of memory that each module can support, and other special options for the module. Module/IP Block Wizard automatically allows user to create memories larger than the width and depth supported for each memory primitive.

Output Register

The output data of the memory module is optionally registered at the output. The user can choose this option by selecting the Enable Output Register check box in Module/IP Block Wizard while customizing the module.

Reset

The memory modules also support the Reset signal. The Reset (or RST) signal only resets output registers of the memory module. It does not reset the contents of the memory module.

Timing

To correctly write into a memory cell in the memory module, the correct address should be registered by the logic. Hence, it is important to note that while running the trace on the memory module, there should be no setup and hold time violations on the address registers (address). Failing to meet these requirements can result in incorrect addressing and, consequently, corruption of memory contents. During a read cycle, a similar issue can occur that involves the correct contents not being read if the address is not correctly registered in the memory.

Unaligned Access Feature (for Large RAM Modules)

The Large RAM Module supports RISC-V compressed instruction chunks of data and shift them. If RISC-V needs to read the upper 16-bit of data in some address, it is very helpful to add support for shifting the upper 16 bits of output into the lower 16 bits, padding the upper bits with to 0. Set 'Enable Unaligned Read' to True to enable this feature.

2.1. Single Port RAM (RAM_DQ) – EBR Based

Lattice FPGAs support all the features of Single Port Memory Module or RAM_DQ. Module/IP Block Wizard allows user to generate the Verilog-HDL for the memory size as per design requirement.

Module/IP Block Wizard generates the memory module, as shown in Figure 2.1.

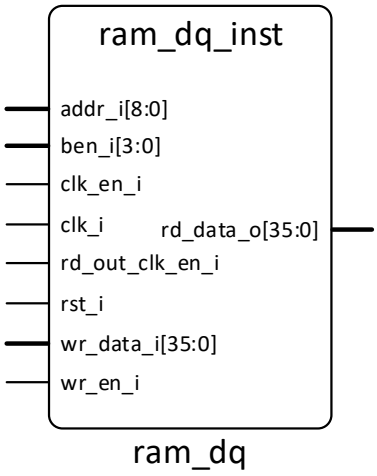

Figure 2.1. Single Port Memory Generated by Module/IP Block Wizard

The various ports and their definitions for Single Port Memory are listed in Table 2.1. The table lists the corresponding ports for the module generated by Module/IP Block Wizard.

Table 2.1. EBR-Based Single Port Memory Port Definitions

Port Name	Direction	Width	Description
addr_i	Input	Address Width	Address Bus
ben_i	Input	Byte Size Width	Byte Enable port, active low (0)
clk_en_i	Input	1	Clock Enable
clk_i	Input	1	Clock
rd_out_clk_en_i	Input	1	Read Output Register Clock Enable
rst_i	Input	1	Reset active high (1)
wr_data_i	Input	Data Width	Data Input
wr_en_i	Input	1	Write Enable
rd_data_o	Output	Data Width	Data Output

The various attributes available for the Single Port Memory (RAM_DQ) are listed in Table 2.2. Some of these attributes are user selectable through the Module/IP Block Wizard interface.

Table 2.2. EBR-Based Single Port Memory Attribute Definitions

Attributes	Description	Values	Default Value
Configuration Attributes			
Address Depth	Address depth of the Read and Write port	2 – 65,536	512
Data Width	Data word width of the Read and Write port	1 – 512	36
Enable Output Register	Data Out port (Q) can be registered or not using this selection	True, False	True
Enable Output ClockEn	Clock Enable for the output clock (this option requires enabling output register)	True, False	False
Reset Assertion	Selection for Reset to be Synchronous or Asynchronous to the Clock	async, sync	sync
Enable Byte Enable[1]	Enables the Byte control port (ben_i)	True, False	False
Initialization Attributes			
Memory Initialization	Allows user to initialize memories to all 1s, 0s, or provide custom initialization through a memory file.	none, Initialize to all 0s, Initialize to all 1s, Memory File	none
Memory File	When Memory File is selected, user can browse to the memory file for custom initialization of RAM.	—	none
Memory File Format	This option allows user to select if the memory file is formatted as Binary or Hex. (See the Initialization File Formats section for details on the different memory file formats.)	binary, hex	hex
Allow update of initialization data	Allow user to update initialization data	True, False	False

Note:

1. Byte-Enable can only be used for data widths ≥ 16 bits.

The Single Port RAM timing waveforms are shown in Figure 2.2 and Figure 2.3.

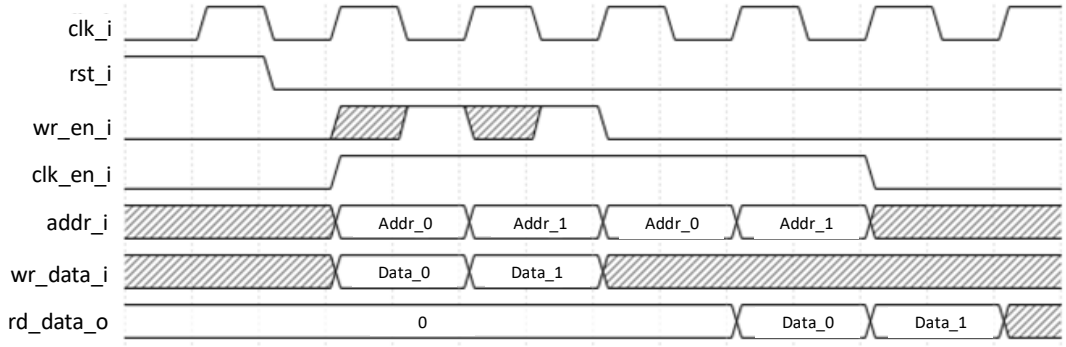

Figure 2.2. Single Port Memory Timing Diagram, Without Output Registers

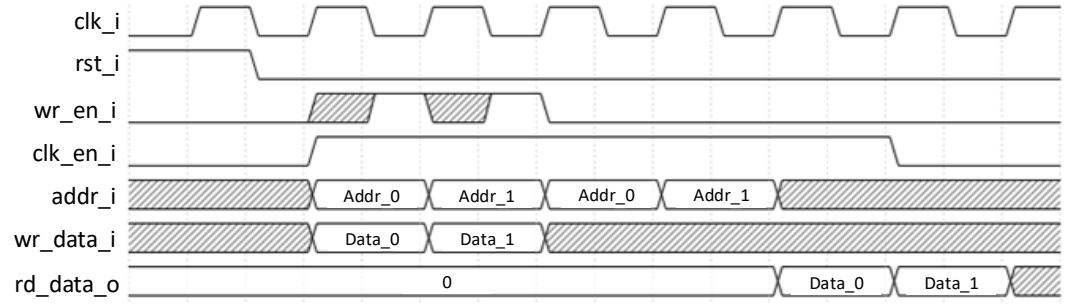

Figure 2.3. Single Port Memory Timing Diagram, With Output Registers

2.2. Pseudo Dual Port RAM (RAM_DP) – EBR Based

Lattice FPGAs support all the features of Pseudo Dual Port Memory Module or RAM_DP. Module/IP Block Wizard allows user to generate the Verilog-HDL for the memory size as per design requirement.

Module/IP Block Wizard generates the memory module shown in Figure 2.4.

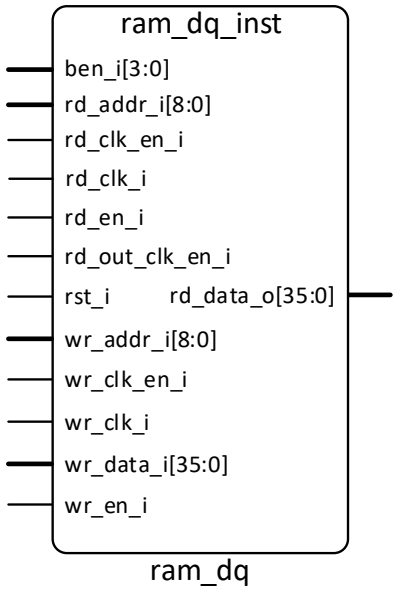

Figure 2.4. Pseudo Dual Port Memory Generated by Module/IP Block Wizard

The various ports and their definitions for Pseudo Dual Port memory are listed in Table 2.3. The table lists the corresponding ports for the module generated by Module/IP Block Wizard.

Table 2.3. EBR-Based Pseudo Dual Port Memory Port Definitions

Port Name	Direction	Width	Description
ben_i	Input	Byte Size Width	Byte Enable port, active low (0)
rd_addr_i	Input	Read Address Width	Read Address
rd_clk_en_i	Input	1	Read Clock Enable
rd_clk_i	Input	1	Read Clock
rd_en_i	Input	1	Read Enable
rd_out_clk_en_i	Input	1	Read Output Register Clock Enable
rst_i	Input	1	Reset active high (1)
wr_addr_i	Input	Write Port Address Width	Write Address
wr_clk_en_i	Input	1	Write Clock Enable
wr_clk_i	Input	1	Write Clock
wr_data_i	Input	Write Port Data Width	Write Data
wr_en_i	Input	1	Write Enable
rd_data_o	Output	Read Port Data Width	Read Data
one_err_det_o	Output	1	ECC 1-bit error detect
two_err_det_o	Output	1	ECC 2-bit error detect

The various attributes available for the Pseudo Dual Port Memory (RAM_DP) are listed in Table 2.4. Some of these attributes are user-selectable through the Module/IP on Local.

Table 2.4. EBR-Based Pseudo Dual Port Memory Attribute Definitions

Attribute	Description	Values	Default Value
General Attributes			
Write Port Address Depth	Write Port Address depth	2 – 65,536	512
Write Port Data Width[1,2]	Write Port Data word width	1 – 256	36
Read Port Address Depth	Read Port Address depth	2 – 65,536	512
Read Port Data Width[1,2]	Read Port Data word width	1 - 256	36
Enable Output Register	Data Out port (Q) can be registered or not using this selection.	True, False	True
Enable Output ClockEn	Clock Enable for the output clock (this option requires enabling output register)	True, False	False
Reset Assertion	Selection for the Reset to be Synchronous or Asynchronous to the Clock	async, sync	sync
Enable Byte Enable[3, 4, 5]	Enables the Byte control port (ben_i)	True, False	False
Enable ECC [5, 6]	Enable the ECC function for the EBR	True, False	False
Initialization Attributes			
Memory Initialization [7]	Allows user to initialize their memories to all 1s, 0s, or providing custom initialization through a memory file.	none, Initialize to all 0s, Initialize to all 1s, Memory File	none
Memory File	When Memory file is selected, user can browse to the memory file for custom initialization of RAM.	—	none

Attribute	Description	Values	Default Value
Memory File Format	This option allows user to select if the memory file is formatted as Binary or Hex. (See the Initialization File Formats section for details on the different formats.)	binary, hex	hex
Allow update of initialization data	Allow user to update initialization data	True, False	False

Notes:

1. For mixed width configurations, the total bit size must be equal (that is, WDEPTH × WDATA = RDEPTH × RDATA)
2. For mixed width configuration, the ratio between max DATA and min DATA must be power of 2 (that is. if WDATA > RDATA, then 2^n = (WDATA/RDATA), where n must be a positive integer and $2^n \leq 64$).
3. Byte-Enable with mixed-width: the ratio between max DATA and min DATA must be 2, 4 or 8.
4. Byte-Enable can only be used for data widths ≥ 16 bits.
5. ECC and Byte-Enable cannot be used together.
6. ECC is available only if WDATA_WIDTH = RDATA_WIDTH
7. Initialization files should be provided with respect to WADDR_DEPTH × WDATA_WIDTH.
8. If writing and reading at the same address simultaneously, the behavior is not guaranteed (that is, either the read or write may happen first).

The user has the option to enable the output registers for Pseudo Dual Port RAM (RAM_DP). The internal timing waveforms for Pseudo Dual Port RAM (RAM_DP) with these options are shown in Figure 2.5 and Figure 2.6. A mixed width timing with no output registers is also provided in Figure 2.7.

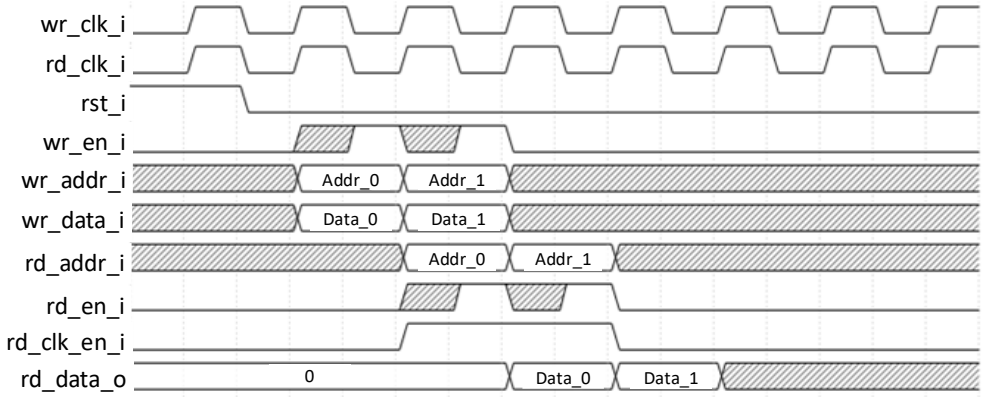

Figure 2.5. Pseudo Dual Port Memory Timing Diagram, Without Output Registers

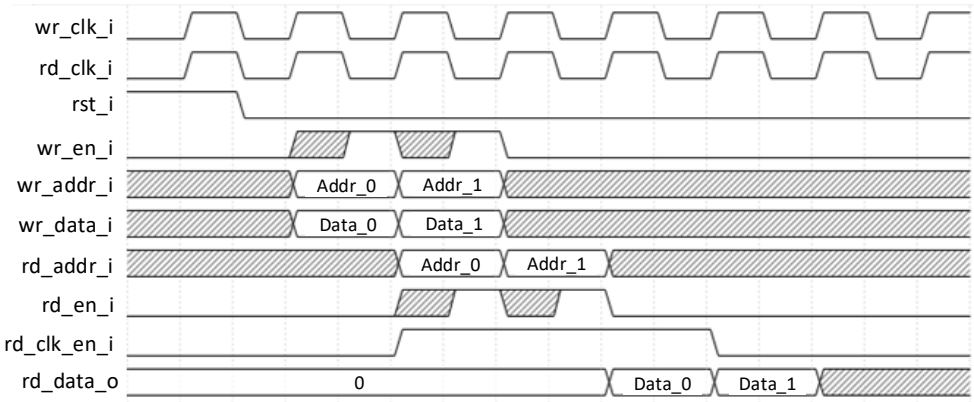

Figure 2.6. Pseudo Dual Port Memory Timing Diagram, With Output Registers

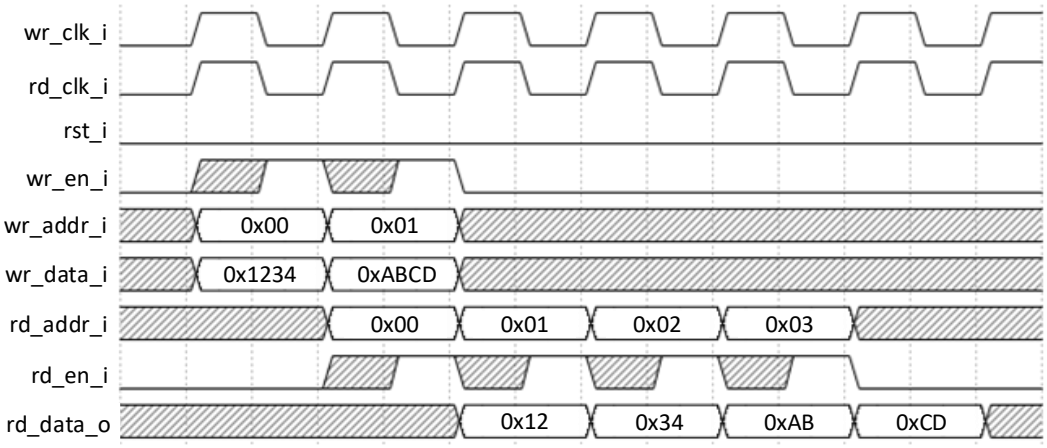

Figure 2.7. Pseudo Dual Port Memory Timing Diagram, Mixed-Width Without Output Registers

2.3. True Dual Port RAM (RAM_DP True) – EBR Based

Lattice FPGA devices built on the Lattice Nexus platform support all the features of True-Dual Port Memory Module or RAM_DP True. Module/IP Block Wizard allows user to generate the Verilog-HDL for the memory size as per design requirement.

Module/IP Block Wizard generates the memory module shown in Figure 2.8.

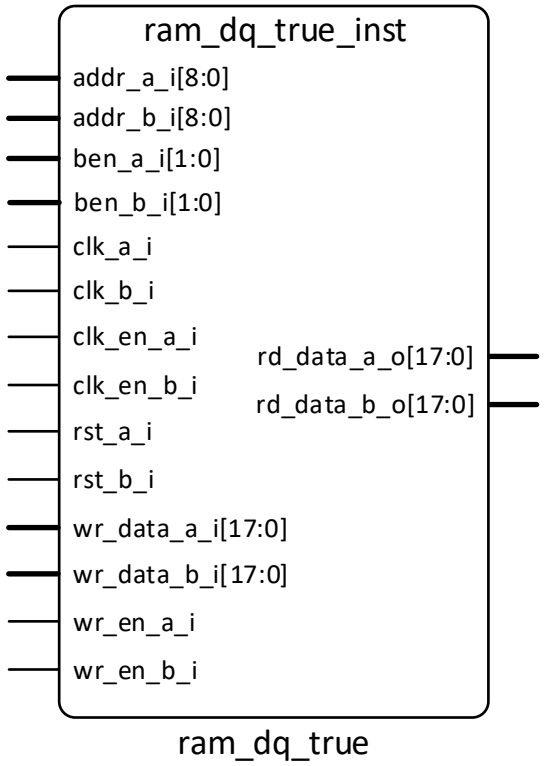

Figure 2.8. True Dual Port Memory Generated by Module/IP Block Wizard

The various ports and their definitions for True Dual Port memory are listed in Table 2.5. The table lists the corresponding ports for the module generated by Module/IP Block Wizard.

Table 2.5. EBR-Based True Dual Port Memory Port Definitions

Port Name	Direction	Width	Description
addr_a_i	Input	Port A Address Width	Port A address
addr_b_i	Input	Port B Address Width	Port B address
ben_a_i	Input	Port A Byte Enable Width	Port A Byte Enable port
ben_b_i	Input	Port B Byte Enable Width	Port B Byte Enable port
clk_a_i	Input	1	Port A Clock
clk_b_i	Input	1	Port B Clock
clk_en_a_i	Input	1	Port A Clock Enable
clk_en_b_i	Input	1	Port B Clock Enable
rst_a_i	Input	1	Port A Reset active high (1)
rst_b_i	Input	1	Port B Reset active high (1)
wr_data_a_i	Input	Port A Data Width	Port A write data
wr_data_b_i	Input	Port B Data Width	Port B write data
wr_en_a_i	Input	1	Port A Write Enable
wr_en_b_i	Input	1	Port B Write Enable
rd_data_a_o	Output	Port A Data Width	Port A Output Data
rd_data_b_o	Output	Port B Data Width	Port B Output Data

The various attributes available for the True Dual Port Memory (RAM_DP True) are listed in Table 2.6. Some of these attributes are user-selectable through the Module/IP on Local.

Table 2.6. EBR-Based True Dual Port Memory Attribute Definitions

Attribute	Description	Values	Default Value
General Attributes			
Port A Address Depth	Port A Address Depth	2 – 65,536	512
Port A Data Width	Port A Data Width	1 – 512	18
Port B Address Depth	Port B Address depth.[1, 2]	2 – 65,536	512
Port B Data Width	Port B Data word width.[1, 2]	1 – 512	18
Enable Output Register (Port A)	Data Out port A (Q) can be registered or not using this selection.	True, False	True
Enable Output Register (Port B)	Data Out port B (Q) can be registered or not using this selection.	True, False	True
Reset Assertion (Port A)	Selection for the Port A Reset to be Synchronous or Asynchronous to the Clock	async, sync	sync
Reset Assertion (Port B)	Selection for the Port B Reset to be Synchronous or Asynchronous to the Clock	async, sync	sync
Reset Deassertion (Port A)	Selection for the Port A Reset Release to be Synchronous or Asynchronous to the Clock	async, sync	sync
Reset Deassertion (Port B)	Selection for the Port A Reset Release to be Synchronous or Asynchronous to the Clock	async, sync	sync
Enable Byte Enable (Port A) [3, 4]	Enables the Byte control port (ben_i) for port A's write port.	True, False	False

Attribute	Description	Values	Default Value
Enable Byte Enable (Port B) [3, 4]	Enables the Byte control port (ben_i) for port B's write port.	True, False	False
Initialization Attributes			
Memory Initialization	Allows user to initialize their memories to all 1s, 0s, or providing custom initialization through a memory file.	none, Initialize to all 0s, Initialize to all 1s, Memory File	none
Memory File	When Memory file is selected, user can browse to the memory file for custom initialization of RAM.	—	none
Memory File Format	This option allows user to select if the memory file is formatted as Binary or Hex. (See the Initialization File Formats section for details on the different formats.)	binary, hex	hex
Allow update of initialization data	Allow user to update initialization data	True, False	False

Notes:

1. For mixed width configurations total bit size must be equal (that is, ADEPTH × ADATA = BDEPTH × BDATA)
2. For mixed width configuration the ratio between max DATA and min DATA must be power of 2 (that is, if ADATA > BDATA, then 2^n = (ADATA/BDATA), where n must be a positive integer and $2^n \leq 32$).
3. Byte-enable for mixed width configuration is applicable only when max DATA = 2 × min DATA (that is, if ADATA > BDATA, then FACTOR = 2 = (ADATA / BDATA)).
4. Initialization files should be provided with respect to ADDR_DEPTH_A × DATA_WIDTH_A.
5. If writing and reading at the same address simultaneously, the behavior is not guaranteed (that is, either the read or write may happen first).

The user has the option to enable the output registers for True Dual Port RAM (RAM_DP_TRUE). The internal timing waveforms for True Dual Port RAM (RAM_DP_TRUE) with these options are shown in Figure 2.9 and Figure 2.10. A mixed width timing with no output registers is also provided in Figure 2.11.

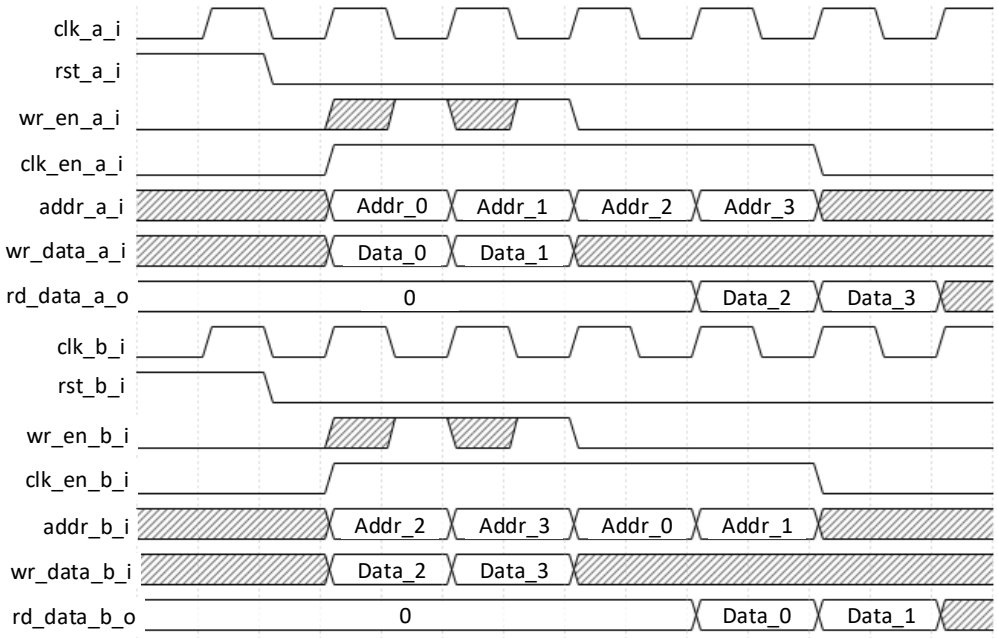

Figure 2.9. True Dual Port Memory Timing Diagram, Without Output Registers

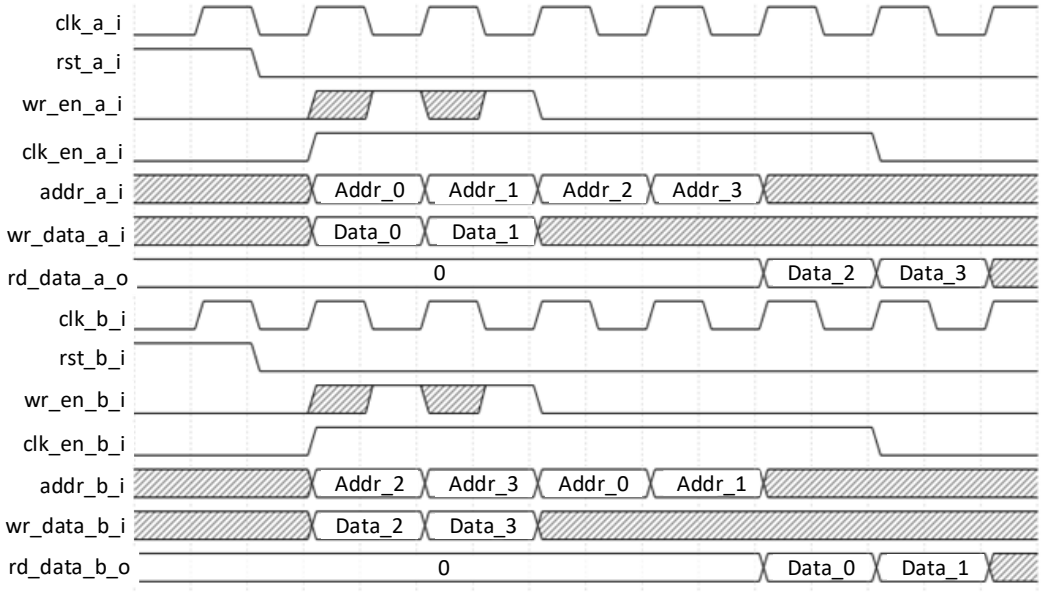

Figure 2.10. True Dual Port Memory Timing Diagram, With Output Registers

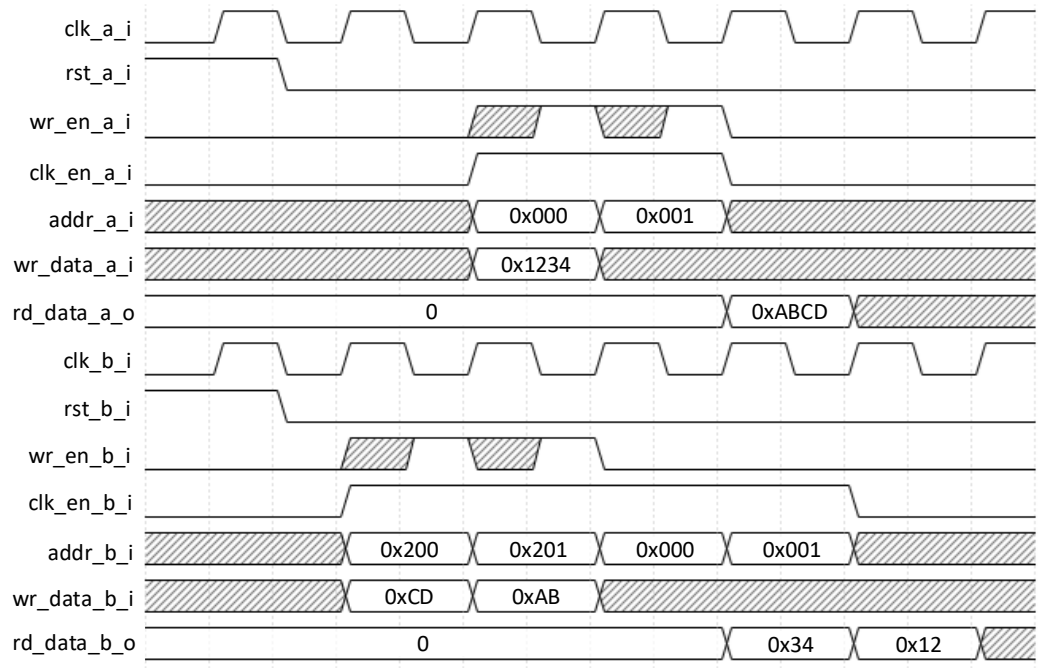

Figure 2.11. True Dual Port Memory Timing Diagram, Mixed Width Without Output Registers

2.4. Read Only Memory (ROM) – EBR Based

Lattice FPGAs support all the features of Read Only Memory Module or ROM. Module/IP Block Wizard allows user to generate the Verilog-HDL for the memory size as per design requirement.

Module/IP Block Wizard generates the memory module shown in Figure 2.12.

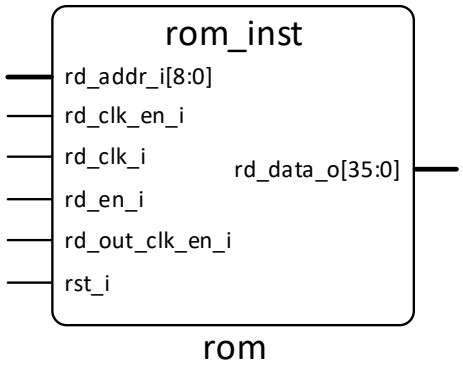

Figure 2.12. Read Only Memory Generated by Module/IP Block Wizard

The various ports and their definitions for Read Only Memory are listed in Table 2.7. The table lists the corresponding ports for the module generated by Module/IP Block Wizard.

Table 2.7. EBR-Based Read Only Memory Port Definitions

Port Name	Direction	Width	Description
rd_addr_i	Input	Read Address Width	Read Address
rd_clk_en_i	Input	1	Read Clock Enable
rd_clk_i	Input	1	Read Clock input
rd_en_i	Input	1	Read Enable
rd_out_clk_en_i	Input	1	Read Out Clock Enable
rst_i	Input	1	Reset active high (1)
rd_data_o	Output	Read Data Width	Read Data

The various attributes available for the Read Only Memory (ROM) are listed in Table 2.8. Some of these attributes are user-selectable through the Module/IP on Local.

Table 2.8. EBR-Based Read Only Memory Attribute Definitions

Attribute	Description	Values	Default Value
General Attributes			
Read Port Address Depth	Read Port Address Depth	2 – 65,536	512
Read Port Data Width	Read Port Data Width	1 - 512	36
Enable Output Register	Data Out (Q) can be registered or not using this selection.	True, False	True
Enable Output ClockEn	Clock Enable for the output clock of (this option requires enabling output register)	True, False	False
Reset Assertion	Selection for the Reset to be Synchronous or Asynchronous to the Clock	async, sync	sync
Initialization Attributes			
Memory Initialization	Allows user to initialize the memory by providing a custom initialization through a memory file.	Memory file	Memory file

Attribute	Description	Values	Default Value
Memory File	When Memory file is selected, user can browse to the memory file for custom initialization of RAM.	—	none
Memory File Format	This option allows user to select if the memory file is formatted as Binary or Hex. (See the Initialization File Formats section for details on the different formats.)	binary, hex	hex
Allow update of initialization data	Allow user to update initialization data	True, False	False

The Read Only Memory timing waveforms are shown in Figure 2.13 and Figure 2.14.

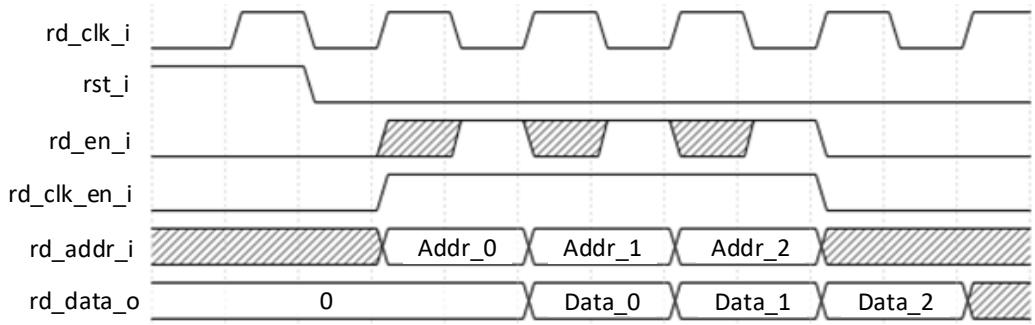

Figure 2.13. Read Only Memory Timing Diagram, Without Output Registers

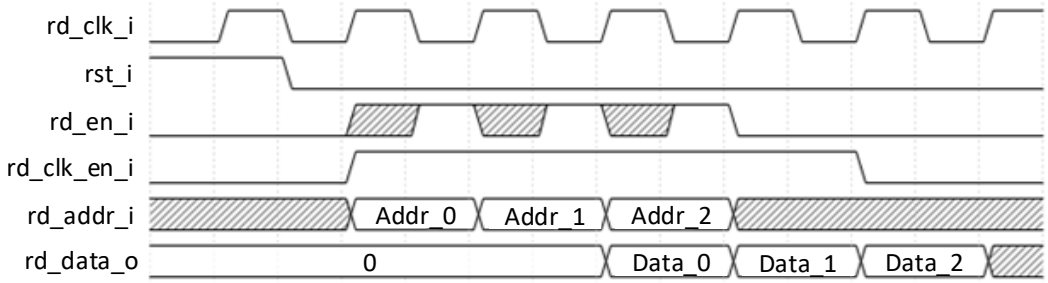

Figure 2.14. Read Only Memory Timing Diagram, With Output Registers

2.5. Shift Register (Shift_Register) – EBR Based or LUT Based

Lattice FPGAs support all the features of Shift Register or Shift_Register. Module/IP Block Wizard allows user to generate the Verilog-HDL for the memory size as per design requirement. The memory for the Shift Register can be implemented using either EBR or LUT, which can be specified during IP generation.

Module/IP Block Wizard generates the memory module shown in Figure 2.15.

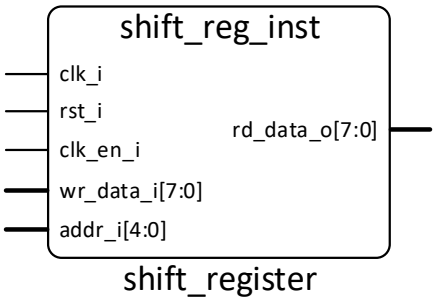

Figure 2.15. Shift Register Generated by Module/IP Block Wizard

The various ports and their definitions for Shift Register are listed in Table 2.9. The table lists the corresponding ports for the module generated by Module/IP Block Wizard.

Table 2.9. Shift Register Port Definitions

Port Name	Direction	Width	Description
clk_i	Input	1	Clock
rst_i	Input	1	Reset active high (1)
clk_en_i	Input	1	Clock Enable
wr_data_i	Input	Data Width	Data Input
addr_i	Input	Max Shift Width	Current amount of shift from the current address. (Requires shift register type to lossless, or variable).
rd_data_o	Output	Data Width	Data Output

The various attributes available for the Shift Register (Shift_Register) are listed in Table 2.10. Some of these attributes are user selectable through the Module/IP Block Wizard interface.

Table 2.10. Shift Register Attribute Definitions

Attribute	Description	Values	Default Value
Configuration Attributes			
Max Shift[1]	The maximum address shifts allowed	2 – 8,192	16
Data Width[1]	Data word width of the Read and Write port	1 - 256	8
Enable Output Register	Data Out port (Q) can be registered or not using this selection	True, False	True
Shift Register Type	Chooses whether, the shift register is operating at: (1) fixed delta from the currently written address, or (2) variable delta from the currently written address	fixed, variable	fixed
Implementation[1]	Chooses how the shift register memory is implemented.	EBR, LUT	EBR

Note:

1. For EBR implementations, this IP requires a minimum size of 30-bits to be properly implemented.

Timing diagrams for Shift Register are shown in Figure 2.16 and Figure 2.17, for non-registered and registered configurations respectively. A timing diagram is also provided for variable shift configuration without registers see Figure 2.18 and a LUT based shift register on Figure 2.19 (Shift Register uses MAX_DEPTH = 4 for fixed configuration and MAX_DEPTH = 8 for variable configuration).

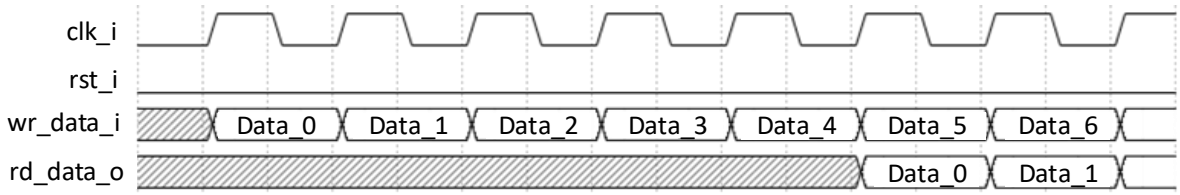

Figure 2.16. Shift Register Without Output Register

Figure 2.17. Shift Register With Output Register

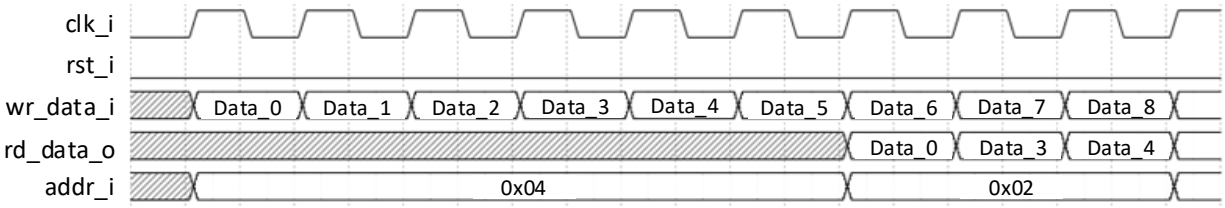

Figure 2.18. Shift Register Variable Configuration and Without Output Register

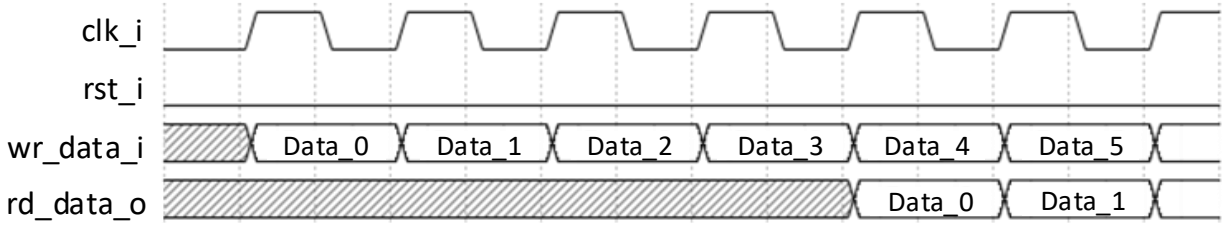

Figure 2.19. Shift Register Without Output Register (LUT Memory, Avant and Nexus Devices)

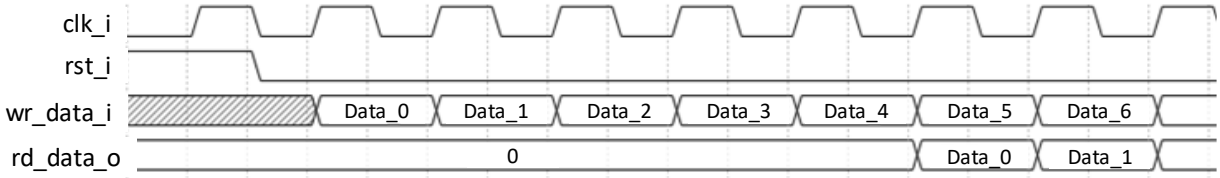

Figure 2.20. Shift Register Without Output Register (LUT Memory, iCE40UP Devices)

2.6. Single Port Large RAM (Large_RAM_SP)

Lattice FPGA devices built on the Lattice Nexus platform support all the features of Single Port Large RAM Module or Large_RAM_SP. Module/IP Block Wizard allows you to generate the Verilog-HDL for the memory size as per design requirement.

The Single Port Large RAM Module/IP Block Wizard is shown in Figure 2.21.

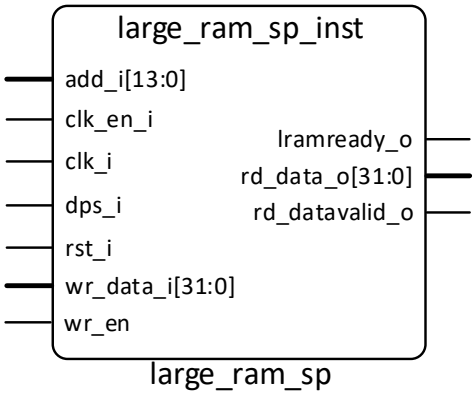

Figure 2.21. Single-Port Large RAM Module Generated by Module/ IP Block Wizards

The various ports and their definitions for Single-Port Memory are listed in Table 2.11. The table lists the corresponding ports for the module generated by Module/IP Block Wizard.

Table 2.11. Large-RAM based Single-Port Memory Port Definitions

Port Name	Direction	Width	Description
clk_i	Input	1	Clock
rst_i	Input	1	Reset active high (1)
dps_i	Input	1	Dynamic Power Switching – when LOW means the LRAM is in normal operation mode, when HIGH means the LRAM is in low power mode, and write/read functions are stopped.
clk_en_i	Input	1	Clock Enable
rdout_clken_i[1]	Input	1	Read Output Register Clock Enable
wr_en_i	Input	1	Write Enable
wr_data_i	Input	Data Width	Data Input
addr_i	Input	Address Width	Address Bus
ben_i[2]	Input	Byte Size Width	Byte Enable port / Unaligned Access shared port, active low (0)
lramready_o	Output	1	When high means that the LRAM is ready for operation
rd_datavalid_o	Output	1	When high means the current output values of rd_data_o and ecc_errdet_o are valid.
rd_data_o	Output	Data Width	Data Output
errdet_o[3]	Output	1	When high, this means an access error occurred since LRAM is in sleep/config mode.
ecc_errdet_o[3]	Output	2	ECC output result: MSB – Two error detect LST – One error detect

Notes:

1. Available when 'Enable Read Out Clock En' is enabled.
2. Available when 'Enable Byte Enable' or 'Enabled Unaligned Read' is enabled.
3. Available when 'Enable ECC' is enabled.

The various attributes available for the Single-Port Memory (Large_RAM_SP) are listed in Table 2.12. Some of these attributes are user selectable through the Module/IP Block Wizard interface.

Table 2.12. Large RAM-Based Single-Port Memory Attribute Definitions

Attributes	Description	Values	Default Value
Configuration			
Address Depth	Address depth of the Read and Write port	2 – 131,072	16,384
Data Width	Data word width of the Read and Write port	1 – 64	32
Enable Output Register	Data Out port (rd_data_o) can be registered or not using this selection	True, False	True
Enable Output ClockEn	Clock Enable for the output clock (this option requires enabling output register)	True, False	False
Reset Assertion	Selection for Reset to be Synchronous or Asynchronous to the Clock	async, sync	Sync
Reset De-Assertion	Selection for Reset De-Assertion to be Synchronous or Asynchronous to the Clock. (This option requires an asynchronous reset assert)	async, sync	Sync
Enable Byte Enable[1, 2, 3]	Enables the Byte control port (ben_i)	True, False	False
Initialization			
Memory Initialization	Allows you to initialize memories to all 1s, 0s, or provide custom initialization through a memory file.	none, all 0, all 1, memory file	none
Memory File	When Memory File is selected, you can browse to the memory file for custom initialization of RAM.	—	none
Memory File Format	This option allows you to select if the memory file is formatted as Binary or Hex. (See the Initialization File Formats section for details on the different memory file formats.)	binary, hex	hex
Miscellaneous Options			
Write/Read Related			
Enable ECC[2, 4]	Enables ECC functionality of the IP	True, False	False
Select Write Mode	Sets the behavior of the output port rd_data_o based on the current write transaction.	“normal”, “write-through”, “read-before-write”	“normal”
Enable Unaligned Read[3, 4]	Enables the unaligned read functionality of the Soft IP which shift the data out based on the current port input at the time of the read command	True, False	False
Others			
Enable Preserve Array	Allows the LRAM to preserve the array when the IP is set to low power mode	True, False	True
Enable GSR	Enables the GSR to reset this IP	True, False	True

Notes:

1. Byte-Enable can only be used for data widths ≥ 16 bits.
2. Byte-Enable CANNOT be used with ECC.
3. Byte-Enable port ben_i is shared with Unaligned Read functionality. The port functions as byte-enable if wr_en_i is HIGH, and as unaligned read when wr_en_i is LOW.
4. ECC and Unaligned Read can only be enabled when DATA_WIDTH = 32.

2.6.1. Unaligned Access Feature

Table 2.13 shows possible combinations for Unaligned Read shifting. The unaligned read pins are shared with the ben_i ports for Single Port Large RAM. Given this, you should be careful in changing the value of these signals. The port functions as byte-enabled during write-access and unaligned read during read-access.

Table 2.13. Unaligned Read shift function (Single Port LRAM)

ben_i[1:0]	ben_i[2] = 0	ben_i[2] = 1
00	DOx=x[31:0] (no shift)	DOx=x[31:0] (no shift)
01	DOx={8'b0,x[31:8]}	DOx={x[23:0],8'b0}
10	DOx={16'b0,x[31:16]}	DOx={x[15:0],16'b0}
11	DOx={24'b0,x[31:24]}	DOx={x[7:0],24'b0}

DOx refers to rd_data_o output port of the Large RAM.

Unaligned Read is only supported if DATA_WIDTH = 32.

Timing diagrams for Large RAM Single Port are shown in Figure 2.22 and Figure 2.23, for non-registered and registered configurations respectively. Additional timing diagrams are provided for write-through mode, Figure 2.24, and read-before-write mode transactions, Figure 2.25.

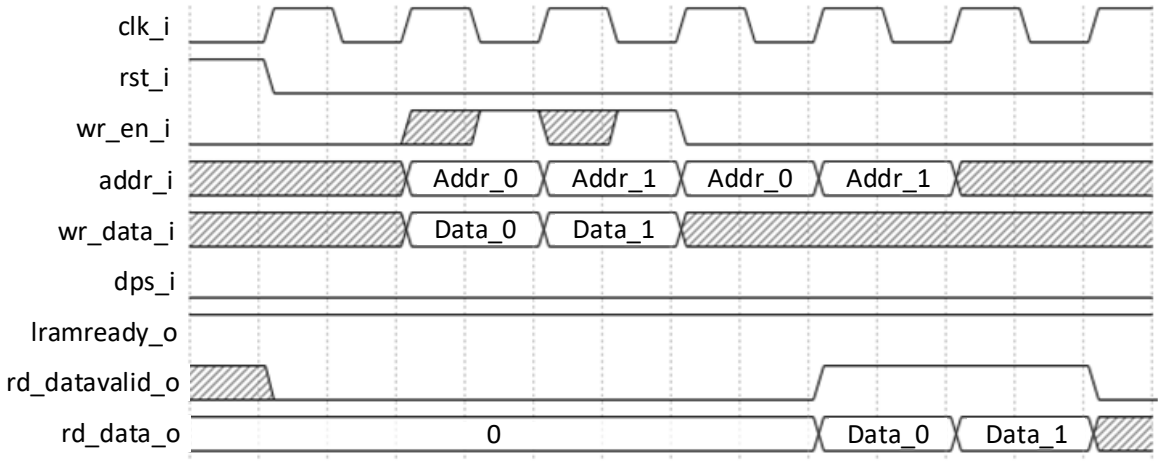

Figure 2.22. Single Port Large RAM Timing Waveform (Normal Mode), Without Output Registers

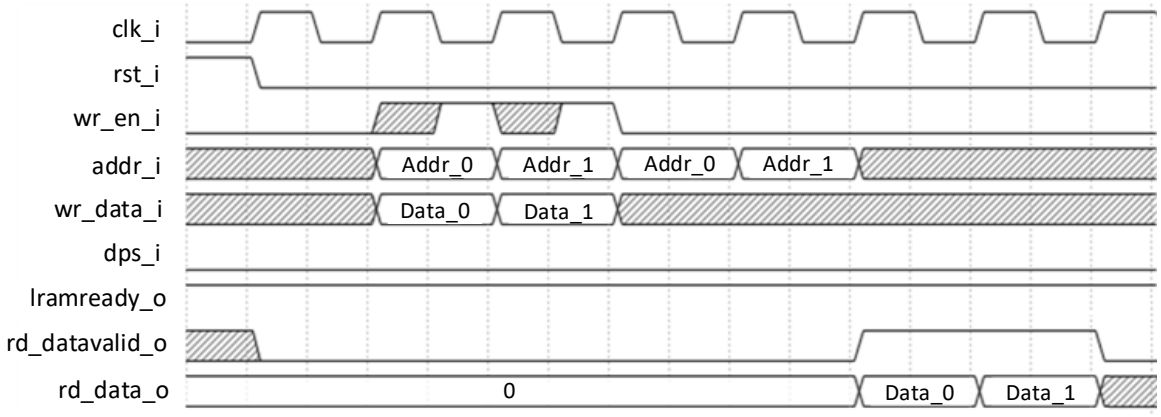

Figure 2.23. Single Port Large RAM Timing Waveform (Normal Mode), With Output Registers

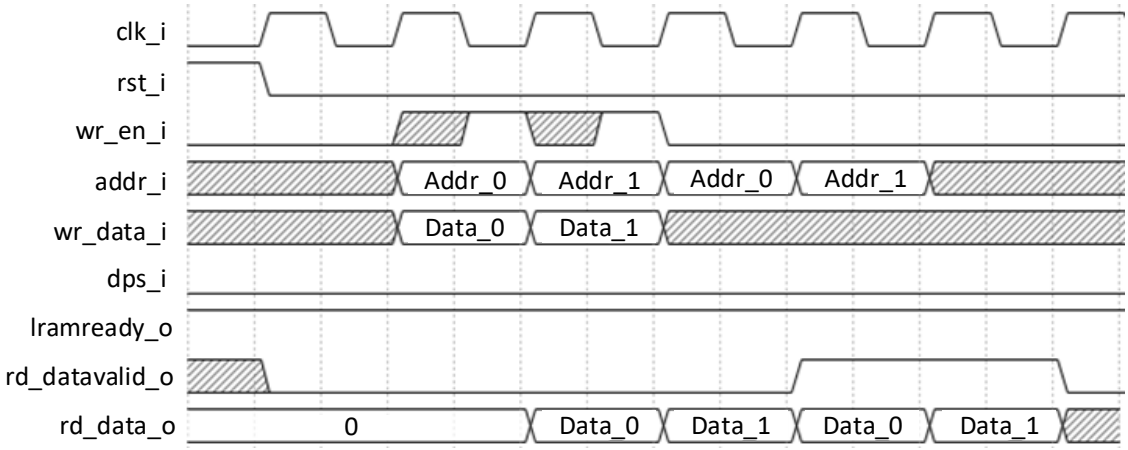

Figure 2.24. Single Port Large RAM Timing Waveform (Write-Through Mode), Without Output Registers

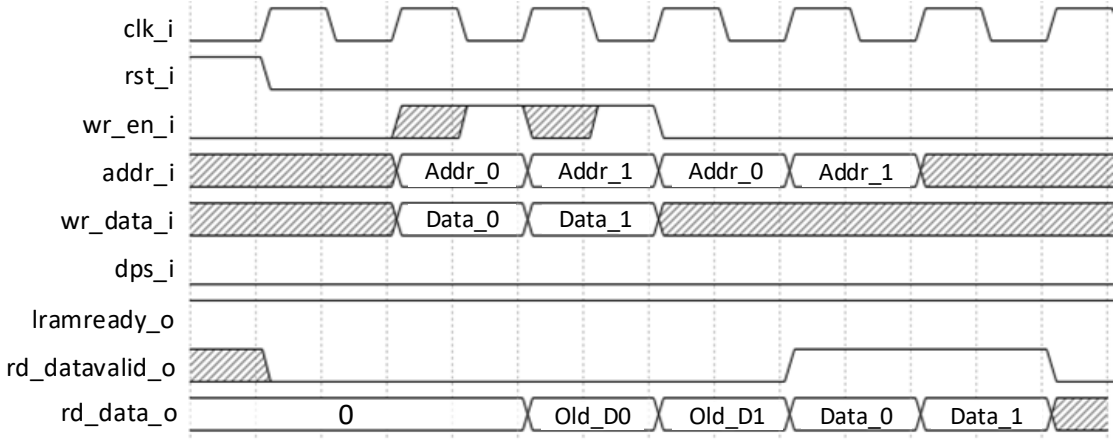

Figure 2.25. Single Port Large RAM Timing Waveform (Read-Before-Write Mode), Without Output Registers

2.7. Pseudo Dual Port Large RAM (Large_RAM_DP)

Lattice FPGA devices built on the Lattice Nexus platform support all the features of Pseudo-Dual Port Large RAM Module or Large RAM_DP. Module/IP Block Wizard allows you to generate the Verilog-HDL for the memory size as per design requirement.

Module/IP Block Wizard generates the memory module shown in Figure 2.26.

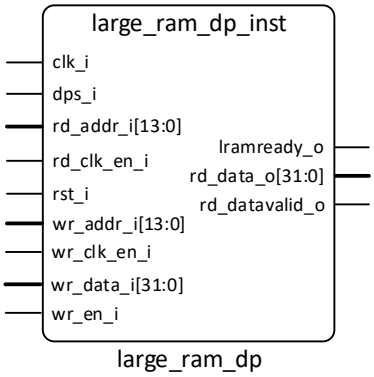

Figure 2.26. Pseudo Dual-Port Large RAM Module Generated by Module/IP Block Wizard

The various ports and their definitions for Pseudo Dual-Port memory are listed in Table 2.14. The table lists the corresponding ports for the module generated by Module/IP Block Wizard.

Table 2.14. Large RAM-Based Pseudo Dual-Port Memory Port Definitions

Port Name	Direction	Width	Description
clk_i	Input	1	Clock
rst_i	Input	1	Reset active high (1)
dps_i	Input	1	Dynamic Power Switching – when LOW means the LRAM is in normal operation mode, when HIGH means the LRAM is in low power mode, and write/read functions are stopped.
wr_clk_en_i	Input	1	Write Clock Enable
rd_clk_en_i	Input	1	Read Clock Enable
rdout_clken_i[1]	Input	1	Read Output Register Clock Enable
wr_en_i	Input	1	Write Enable
wr_data_i	Input	Write Port Data Width	Write Data
wr_addr_i	Input	Write Port Address Width	Write Address
rd_addr_i	Input	Read Address Width	Read Address
ben_i[4]	Input	Byte Size Width	Byte Enable port, active low (0)
unaligned_i[2]	Input	4	Unaligned Access Port
lramready_o	Output	1	When high means that the LRAM is ready for operation
rd_datavalid_o	Output	1	When high means the current output values of rd_data_o and ecc_errdet_o are valid.
rd_data_o	Output	Read Port Data Width	Read Data
errdet_o[3]	Output	1	When high, this means an access error occurred since LRAM is in sleep/config mode.
ecc_errdet_o[3]	Output	2	ECC output result: MSB – Two error detect LST – One error detect

Notes:
1. Available when ‘Enable Read Out Clock En’ is enabled.
2. Available when ‘Enabled Unaligned Read’ is enabled.
3. Available when ‘Enable ECC’ is enabled.
4. Available when ‘Enable Byte Enable’ is enabled.

The various attributes available for the Pseudo Dual-Port Memory (RAM_DP) are listed in Table 2.15. Some of these attributes are user-selectable through the Module/IP on Local.

Table 2.15. Large RAM-Based Pseudo Dual-Port Memory Attribute Definitions

Attribute	Description	Values	Default Value
General Attributes			
Write Port Address Depth[1, 2]	Write Port Address depth	2 – 131,072	16,384
Write Port Data Width[1, 2]	Write Port Data word width	1 – 64	32
Read Port Address Depth[1, 2]	Read Port Address depth	2 – 131,072	16,384
Read Port Data Width[1, 2]	Read Port Data word width	1 – 64	32
Enable Output Register	Data Out port (Q) can be registered or not using this selection.	True, False	True
Enable Output ClockEn	Clock Enable for the output clock (this option requires enabling output register)	True, False	False
Reset Assertion	Selection for the Reset to be Synchronous or Asynchronous to the Clock	async, sync	sync
Enable Byte Enable[3, 4]	Enables the Byte control port (ben_i)	True, False	False

Attribute	Description	Values	Default Value
Initialization Attributes			
Memory Initialization [5]	Allows you to initialize their memories to all 1s, 0s, or providing custom initialization through a memory file.	none, all 0, all 1, Memory file	none
Memory File	When Memory file is selected, you can browse to the memory file for custom initialization of RAM.	—	none
Memory File Format	This option allows you to select if the memory file is formatted as Binary or Hex. (See the Initialization File Formats section for details on the different formats.)	binary, hex	hex
Miscellaneous Options			
Write/Read Related			
Enable ECC[4, 6]	Enables ECC functionality of the IP	True, False	False
Enable Unaligned Read[6]	Enables the unaligned read functionality of the Soft IP which shift the data out based on the current port input at the time of the read command	True, False	False
Others			
Enable Preserve Array	Allows the LRAM to preserve the array when the IP is set to low power mode	True, False	True
Enable GSR	Enables the GSR to reset this IP	True, False	True

Notes:

1. Total Number of bits between write and read ports must match (WADDR_DEPTH × WDATA_WIDTH = RADDR_DEPTH × RDATA_WIDTH).
2. For mixed-width implementations, the allowable factor is 2 or 4 (that is, WDATA_WIDTH = 2 × RDATA_WIDTH, 4 × WDATA_WIDTH = RDATA_WIDTH).
3. Byte-Enable can only be used for data widths ≥ 16 bits.
4. Byte-Enable CANNOT be used with ECC.
5. Memory file is provided with respect to write port (WADDR_DEPTH × WDATA_WIDTH).
6. ECC and/or Unaligned Read can only be used when WDATA_WIDTH = RDATA_WIDTH = 32.

2.7.1. Unaligned Access Feature

Table 2.16 shows combinations for Unaligned Read shifting. The unaligned_i signals are used to implement this feature.

Table 2.16. Unaligned Read shift function (Pseudo Dual Port LRAM)

unaligned_i[1:0]	unaligned_i [2] = 0	unaligned_i [2] = 1
00	DOx=x[31:0] (no shift)	DOx=x[31:0] (no shift)
01	DOx={8'b0,x[31:8]}	DOx={x[23:0],8'b0}
10	DOx={16'b0,x[31:16]}	DOx={x[15:0],16'b0}
11	DOx={24'b0,x[31:24]}	DOx={x[7:0],24'b0}

DOx refers to rd_data_o output port of the Large RAM.

Unaligned Read is only supported if WDATA_WIDTH = RDATA_WIDTH = 32.

User has the option to enable the output registers for Pseudo-Dual Port Large RAM (Large RAM_DP). The internal timing waveforms for Pseudo-Dual Port Large RAM (Large RAM_DP) with these options are shown in Figure 2.27 and Figure 2.28.

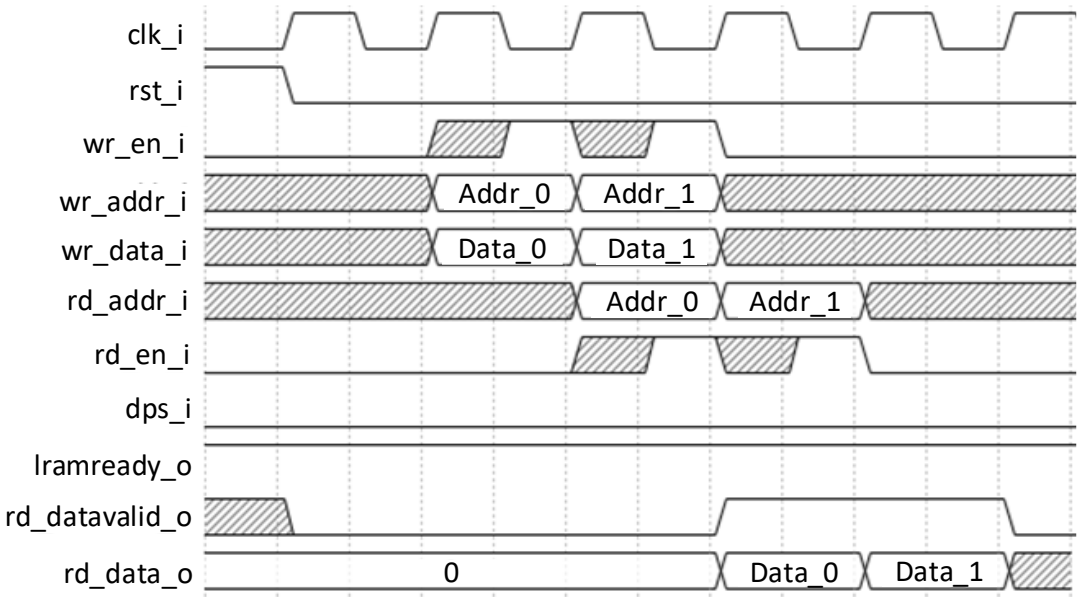

Figure 2.27. Pseudo Dual Port Large RAM Timing Diagram, Without Output Registers

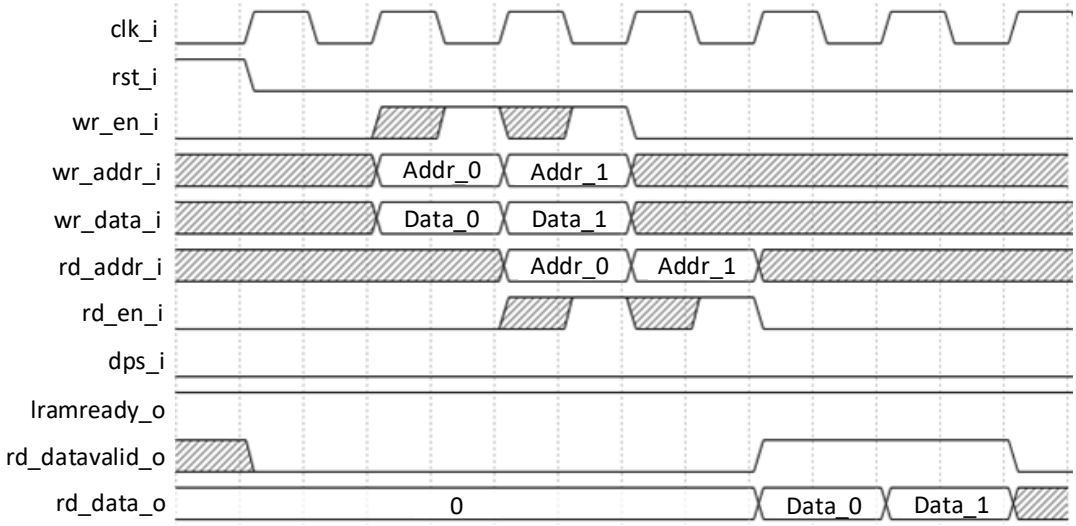

Figure 2.28. Pseudo Dual Port Large RAM Timing Diagram, With Output Registers

2.8. True Dual-Port Large RAM (Large_RAM_DP True)

Lattice FPGA devices built on the Lattice Nexus platform support all the features of True-Dual Port Large RAM or Large RAM_DP_True. Module/IP Block Wizard allows you to generate the Verilog-HDL for the memory size as per design requirement.

The Module/IP Block Wizard generates the memory module shown in Figure 2.29.

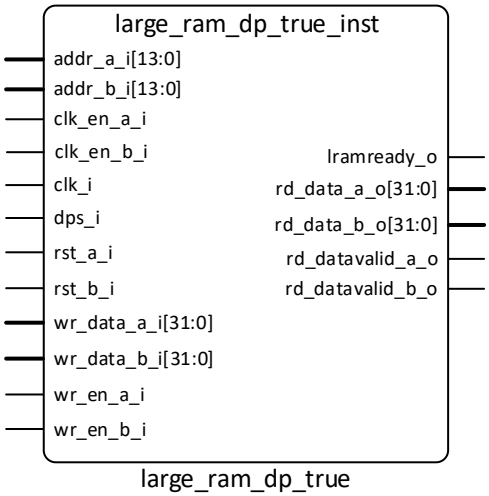

Figure 2.29. True Dual-Port Large RAM Module Generated by Module/IP Block Wizard

The various ports and their definitions for True Dual-Port Large RAM are listed in Table 2.17. The table lists the corresponding ports for the module generated by Module/IP Block Wizard.

Table 2.17. Large RAM-Based True Dual-Port Memory Port Definitions

Port Name	Direction	Width	Description
addr_a_i	Input	Port A Address Width	Port A address
addr_b_i	Input	Port B Address Width	Port B address
wr_data_a_i	Input	Port A Data Width	Port A write data
wr_data_b_i	Input	Port B Data Width	Port B write data
clk_i	Input	1	Clock
dps_i	Input	1	Dynamic Power Switching – when LOW means the LRAM is in normal operation mode, when HIGH means the LRAM is in low power mode, and write/read functions are stopped.
clk_en_a_i	Input	1	Port A Clock Enable
clk_en_b_i	Input	1	Port B Clock Enable
rdout_clken_a_i[1]	Input	1	Port A Read Output Register Clock Enable
rdout_clken_b_i[1]	Input	1	Port B Read Output Register Clock Enable
wr_en_a_i	Input	1	Port A Write Enable
wr_en_b_i	Input	1	Port B Write Enable
rst_a_i	Input	1	Port A Reset active high (1)
rst_b_i	Input	1	Port B Reset active high (1)
ben_a_i[2]	Input	Port A Byte Enable Width	Port A Byte Enable port / Unaligned Access
ben_b_i[2]	Input	Port B Byte Enable Width	Port B Byte Enable port / Unaligned Access
lramready_o	Output	1	When high, this means that the LRAM is ready for operation
rd_datavalid_a_o	Output	1	When high, this means the current output values of rd_data_a_o and ecc_errdet_a_o are valid.
rd_datavalid_b_o	Output	1	When high, this means the current output values of rd_data_b_o and ecc_errdet_b_o are valid.

Port Name	Direction	Width	Description
rd_data_a_o	Output	Port A Data Width	Port A Output Data
rd_data_b_o	Output	Port B Data Width	Port B Output Data
ecc_errdet_a_o[3]	Output	2	Port A ECC output result: MSB – Two error detect LST – One error detect
ecc_errdet_b_o[3]	Output	2	Port B ECC output result: MSB – Two error detect LST – One error detect
errdet_o	Output	1	When high, this means an access error occurred since LRAM is in sleep/config mode.

Notes:
1. Available when 'Enable Read Out Clock En' is enabled.
2. Available when 'Enable Byte Enable' or 'Enabled Unaligned Read' is enabled.
3. Available when 'Enable ECC' is enabled.

The various attributes available for the True Dual-Port Large RAM are listed in Table 2.18. Some of these attributes are user-selectable through the Module/IP on Local.

Table 2.18. Large RAM-Based True Dual-Port Memory Attribute Definitions

Attribute	Description	Values	Default Value
General Attributes			
Port A Address Depth[1, 2]	Port A Address Depth	2 – 131,072	16,384
Port A Data Width[1, 2]	Port A Data Width	1 – 64	32
Port B Address Depth[1, 2]	Port B Address depth.[1, 2]	2 – 131,072	16,384
Port B Data Width[1, 2]	Port B Data word width.[1, 2]	1 – 64	32
Enable Output Register (Port A)	Data Out port A (Q) can be registered or not using this selection.	True, False	True
Enable Output Register (Port B)	Data Out port B (Q) can be registered or not using this selection.	True, False	True
Reset Assertion (Port A)	Selection for the Port A Reset to be Synchronous or Asynchronous to the Clock	async, sync	sync
Reset Assertion (Port B)	Selection for the Port B Reset to be Synchronous or Asynchronous to the Clock.	async, sync	sync
Reset Deassertion (Port A)	Selection for the Port A Reset Release to be Synchronous or Asynchronous to the Clock.	async, sync	sync
Reset Deassertion (Port B)	Selection for the Port A Reset Release to be Synchronous or Asynchronous to the Clock.	async, sync	sync
Enable Byte Enable (Port A) [3, 4]	Enables the Byte control port (ben_i)	True, False	False
Enable Byte Enable (Port B) [3, 4]	Enables the Byte control port (ben_i)	True, False	False
Initialization Attributes			
Memory Initialization[5]	Allows you to initialize their memories to all 1s, 0s, or providing custom initialization through a memory file.	none, all 0, all 1, Memory file	none
Memory File	When Memory file is selected, you can browse to the memory file for custom initialization of RAM.	—	none

Attribute	Description	Values	Default Value
Memory File Format	This option allows you to select if the memory file is formatted as Binary or Hex. (See the Initialization File Formats section for details on the different formats.)	binary, hex	hex
Miscellaneous Options			
Write/Read Related			
Enable ECC[4, 6]	Enables ECC functionality of the IP	True, False	False
Enable Write-Through A	Enable Write-Through mode for Port A	True, False	False
Enable Write-Through B	Enable Write-Through mode for Port B	True, False	False
Enable Unaligned Read[6]	Enables the unaligned read functionality of the Soft IP which shift the data out based on the current port input at the time of the read command.	True, False	False
Others			
Enable Preserve Array	Allows the LRAM to preserve the array when the IP is set to low power mode.	True, False	True
Enable GSR	Enables the GSR to reset this IP	True, False	True

Notes:

1. For mixed width configurations, total bit size must be equal (that is, ADEPTH × ADATA = BDEPTH × BDATA).
2. For mixed width configuration, the ratio between ADATA and BDATA must be 2 or 4.
3. Byte-enable can only be enabed when ADATA ≥ 16 for port A, and/or BDATA ≥ 16 port B.
4. Byte-enable and ECC cannot be used simultaneously.
5. Initialization files should be provided with respect to ADDR_DEPTH_A x DATA_WIDTH_A.
6. ECC and Unaligned Read can only be enabled when ADATA = BDATA = 32.

2.8.1. Unaligned Access Feature

Table 2.19 shows possible combinations for Unaligned Read shifting. The unaligned read pins are shared with the ben_[x]_i ports for True Dual Port Large RAM. Given this, you should be careful in changing the value of these signals. The port functions as byte-enabled during write-access and unaligned read during read-access.

Table 2.19. Unaligned Read shift function (True Dual Port LRAM)

ben_[x]_i[1:0]	ben_[x]_i[2] = 0	ben_[x]_i[2] = 1
00	DOx=x[31:0] (no shift)	DOx=x[31:0] (no shift)
01	DOx={8'b0,x[31:8]}	DOx={x[23:0],8'b0}
10	DOx={16'b0,x[31:16]}	DOx={x[15:0],16'b0}
11	DOx={24'b0,x[31:24]}	DOx={x[7:0],24'b0}

DO[x] refers to rd_data_[x]_o output port of the Large RAM. Where [x] can be port a or b.

Unaligned Read is only supported if DATA_WIDTH_A = DATA_WIDTH_B = 32.

User has the option to enable the output registers for True Dual Port Large RAM. The internal timing waveforms for True Dual Port Large RAM with these options are shown in Figure 2.30 and Figure 2.31. A sample waveform form for write-through without registered output is also provided in Figure 2.32.

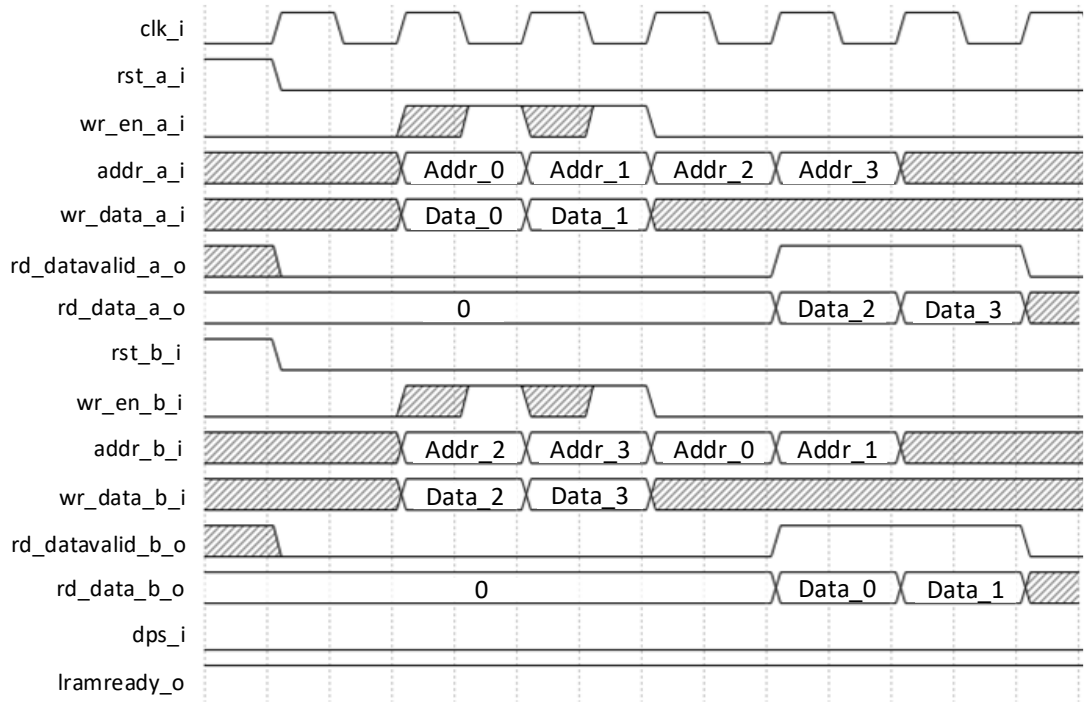

Figure 2.30. True Dual-Port Large RAM Timing Waveform (Normal Mode), Without Output Registers

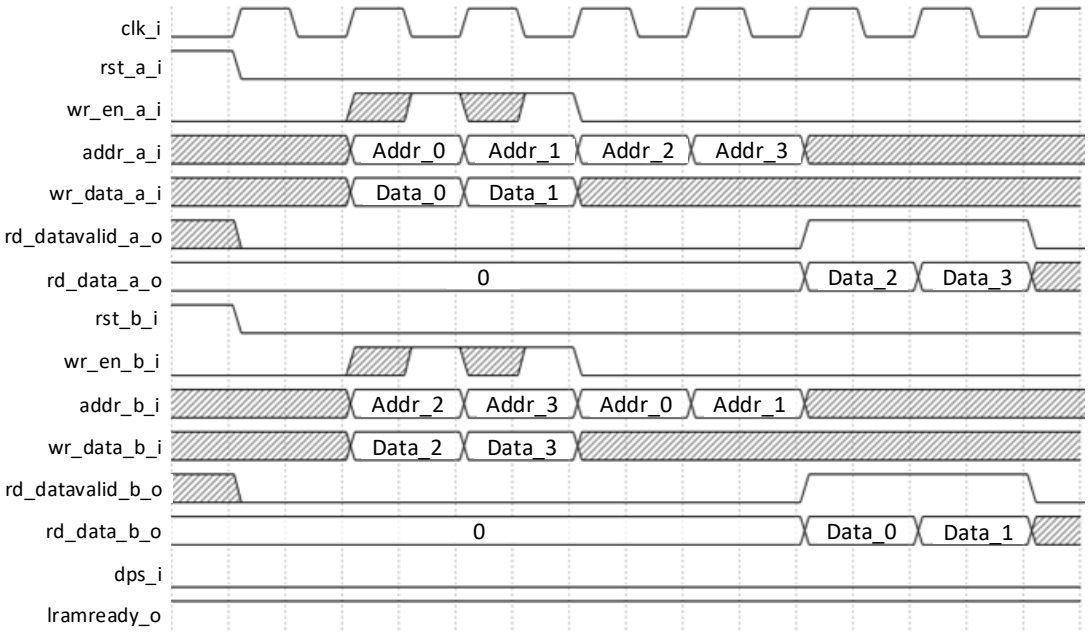

Figure 2.31. True Dual-Port Large RAM Timing Waveform (Normal Mode), With Output Registers

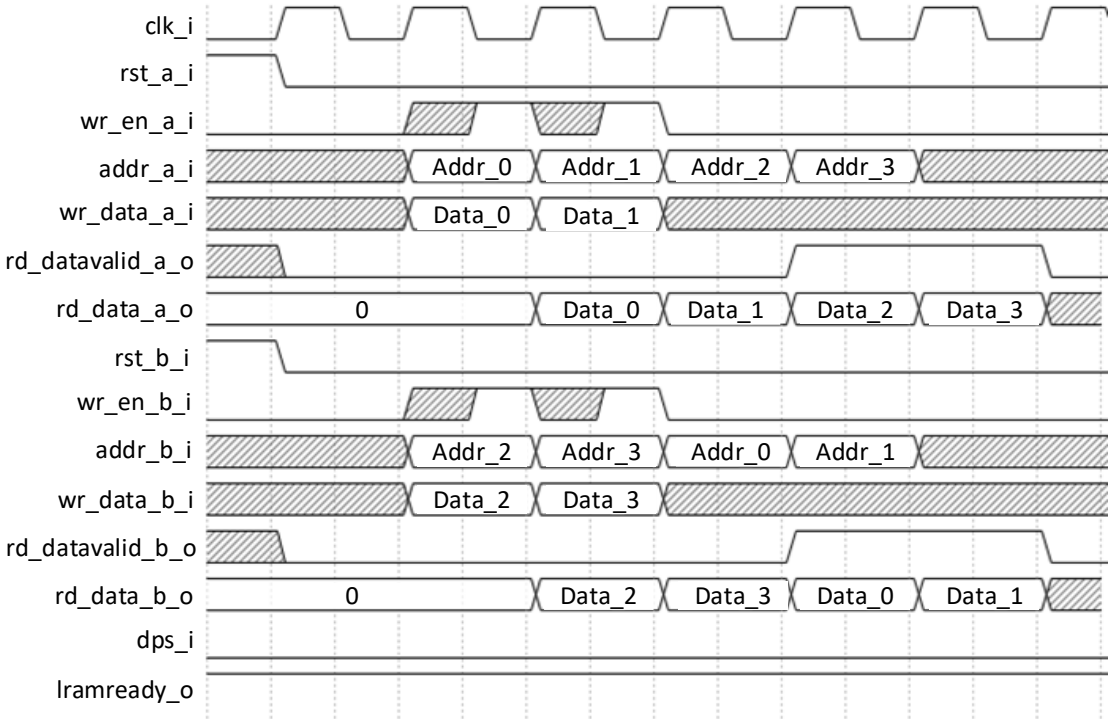

Figure 2.32. True Dual-Port Large RAM Timing Waveform (Write-Through Mode), Without Output Registers

2.9. Large Read-Only Memory (Large_ROM)

Lattice FPGA devices built on the Lattice Nexus platform support all the features of Large Read Only Memory Module. Module/IP Block Wizard allows user to generate the Verilog-HDL for the memory size as per design requirement.

Module/IP Block Wizard generates the memory module shown in Figure 2.33.

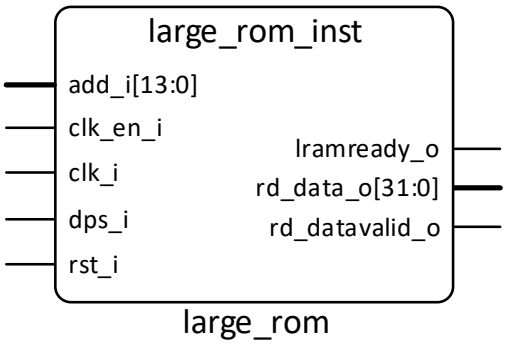

Figure 2.33. Large Read Only Memory Module Generated by Module/IP Block Wizard

The various ports and their definitions for Read Only Memory are listed in Table 2.20. The table lists the corresponding ports for the module generated by Module/IP Block Wizard.

Table 2.20. Large Read Only Memory Port Definitions

Port Name	Direction	Width	Description
clk_i	Input	1	Clock input
rst_i	Input	1	Reset active high (1)
dps_i	Input	1	Dynamic Power Switching – when LOW means the LRAM is in normal operation mode, when HIGH means the LRAM is in low power mode, and read functions are stopped.
clk_en_i	Input	1	Clock Enable
rdout_clken_i[1]	Input	1	Read Out Clock Enable
addr_i	Input	Address Width	Address
unaligned_i[2]	Input	4	Unaligned Access Port
lramready_o	Output	1	When high, this means that the LRAM is ready for operation.
rd_datavalid_o	Output	1	When high, this means the current output values of rd_data_o and ecc_errdet_o are valid.
rd_data_o	Output	Read Data Width	Read Data
errdet_o[3]	Output	1	When high means an access error occurred since LRAM is in sleep/config mode.
ecc_errdet_o[3]	Output	2	ECC output result: MSB – Two error detect LST – One error detect

Notes:
1. Available when 'Enable Read Out Clock En' is enabled.
2. Available when 'Enabled Unaligned Read' is enabled.
3. Available when 'Enable ECC' is enabled.

The various attributes available for the Read Only Memory (ROM) are listed in Table 2.21. Some of these attributes are user-selectable through the Module/IP on Local.

Table 2.21. Large Read Only Memory Attribute Definitions

Attribute	Description	Values	Default Value
General Attributes			
Address Depth	Read Port Address Depth	2–131,072	16,384
Data Width	Read Port Data Width	1–64	32
Enable Output Register	Data Out (Q) can be registered or not using this selection.	True, False	True
Enable Read Output Clock En	Clock Enable for the output clock of (this option requires enabling output register)	True, False	False
Reset Assertion	Selection for the Reset to be Synchronous or Asynchronous to the Clock	async, sync	sync
Initialization Attributes			
Memory File	When Memory file is selected, you can browse to the memory file for custom initialization of RAM.	—	none
Memory File Format	This option allows you to select if the memory file is formatted as Binary or Hex. (See the Initialization File Formats section for details on the different formats.)	binary, hex	hex
Miscellaneous Options			
Write/Read Related			
Enable ECC[1]	Enables ECC functionality of the IP	True, False	False
Enable Unaligned Read[1]	Enables the unaligned read functionality of the Soft IP which shift the data out based on the current port input at the time of the read command	True, False	False

Attribute	Description	Values	Default Value
Others			
Enable Preserve Array	Allows the LRAM to preserve the array when the IP is set to low power mode	True, False	True
Enable GSR	Enables the GSR to reset this IP	True, False	True

Note:

1. ECC and Unaligned Read can only be enabled when RDATA_WIDTH = 32.

2.9.1. Unaligned Access Feature

Figure 2.22 shows combinations for Unaligned Read shifting. The unaligned_i signals are used to implement this feature.

Table 2.22. Unaligned Read Shift Function (LROM)

unaligned_i[1:0]	unaligned_i [2] = 0	unaligned_i [2] = 1
00	DOx=x[31:0] (no shift)	DOx=x[31:0] (no shift)
01	DOx={8'b0,x[31:8]}	DOx={x[23:0],8'b0}
10	DOx={16'b0,x[31:16]}	DOx={x[15:0],16'b0}
11	DOx={24'b0,x[31:24]}	DOx={x[7:0],24'b0}

DOx refers to rd_data_o output port of the Large RAM.

Unaligned Read is only supported if RDATA_WIDTH = 32.

The Large Read Only Memory timing waveforms are shown in Figure 2.34 and Figure 2.35.

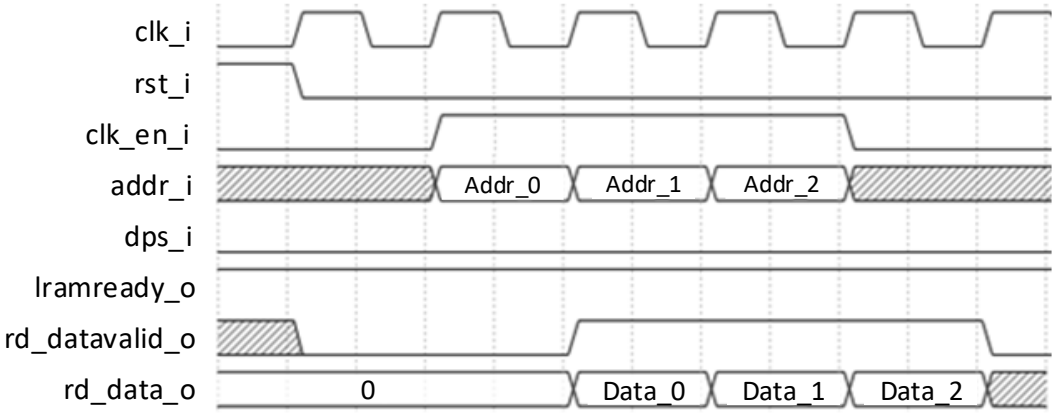

Figure 2.34. Large ROM Timing Diagram, Without Output Registers

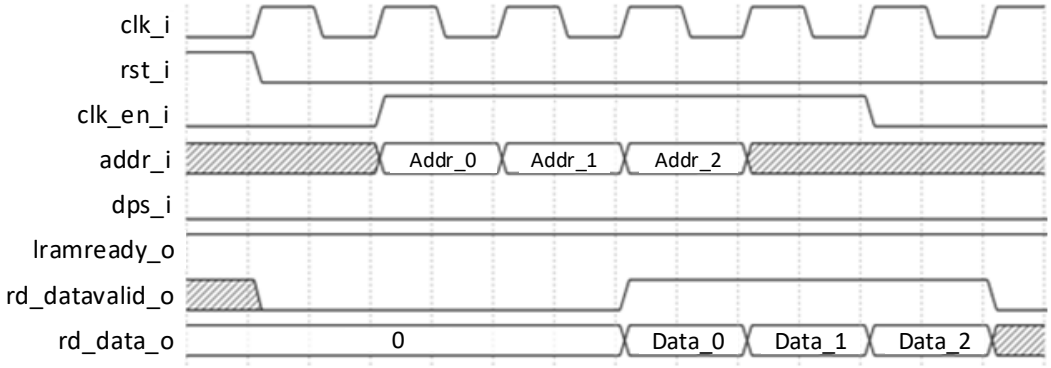

Figure 2.35. Large ROM Timing Diagram, With Output Register

3. IP Generation, Simulation, and Validation

This section provides information on how to generate the EBR Memory Module IP using the Lattice Radiant software and how to run simulation. For more details on the Lattice Radiant software, refer to the Lattice Radiant software user guide.

3.1. Generation

Module/IP Block Wizard in Lattice Radiant Software allows user to generate, create, or open modules for the target device. From the Lattice Radiant Software, select the IP Catalog tab as shown in Figure 3.1.

The left pane of the IP Catalog window displays the module tree. The user can utilize Module/IP on Local to specify a variety of memories in the designs. These modules are constructed using one or more memory primitives along with general purpose routing and LUTs (lookup tables) as required.

The memory modules are categorized as Memory_Modules with EBR_Components as sub-category. The available memory modules in the Lattice Radiant Software IP catalog are shown below.

The right pane of the window shows the description of the selected module and provides links to the documentation.

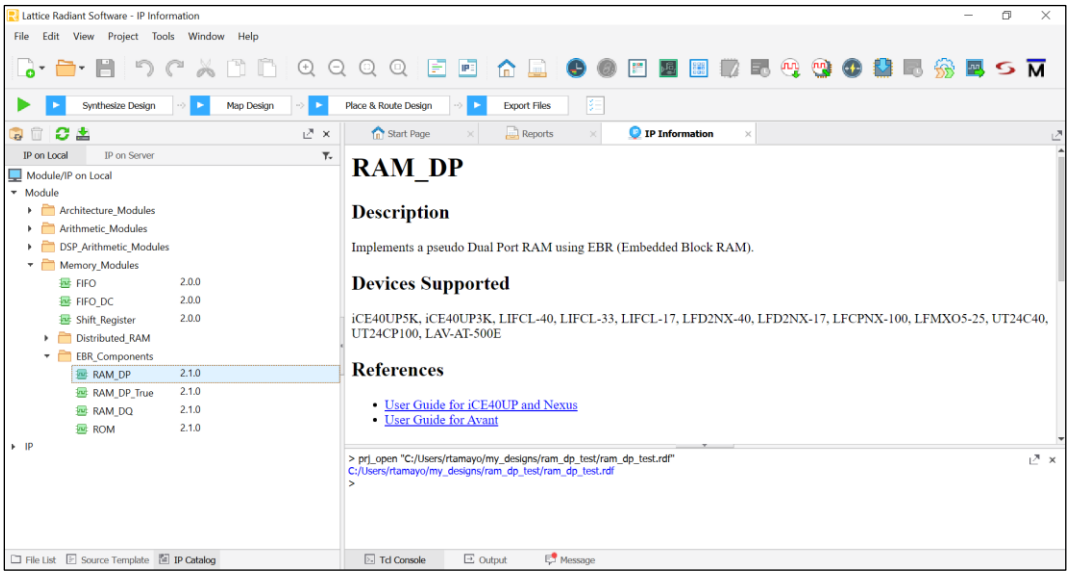

Figure 3.1. Memory Modules Under Module/IP on Local in the Lattice Radiant Software

The following section shows an example of generating an EBR-based Pseudo Dual Port RAM of size 512 × 18.

To generate an EBR-based Pseudo Dual Port RAM of size 512 × 18:

1. Click **Memory_Modules** > **EBR_Components** > **RAM_DP**.
2. The **Module/IP Block Wizard** opens as shown in Figure 3.2. Enter values in the **Component name** and the **Create in** fields then click **Next**.

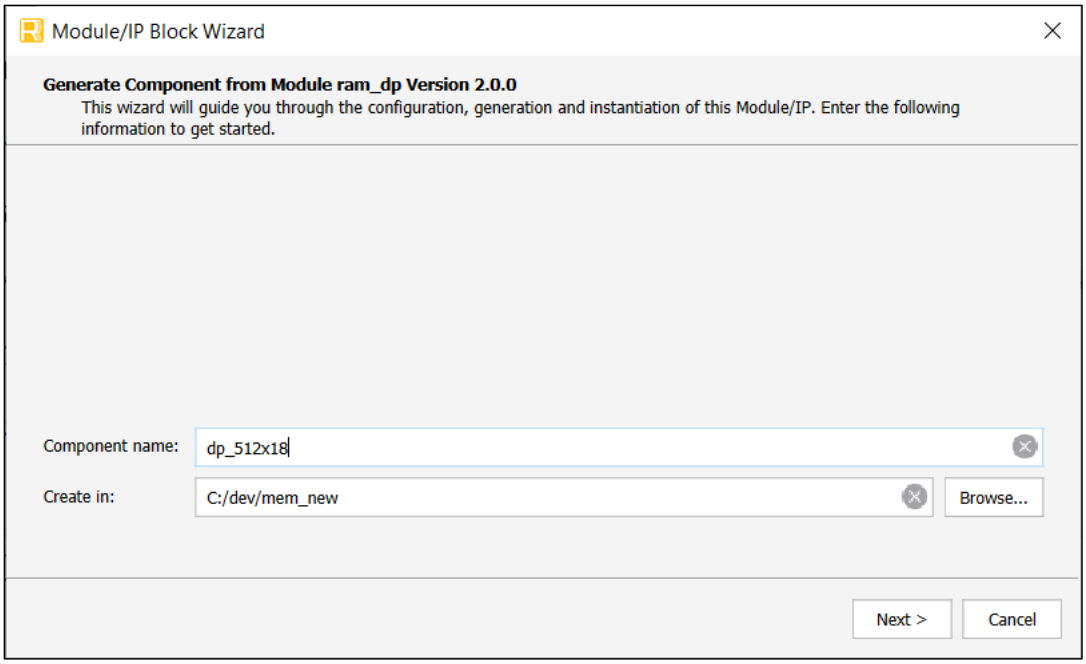

Figure 3.2. Example: Generating RAM_DP Using Module/IP Block Wizard

3. In the module's dialog box of the **Module/IP Block Wizard** window, customize the EBR-based Pseudo Dual Port RAM using drop-down menus and check boxes as shown in Figure 3.3.

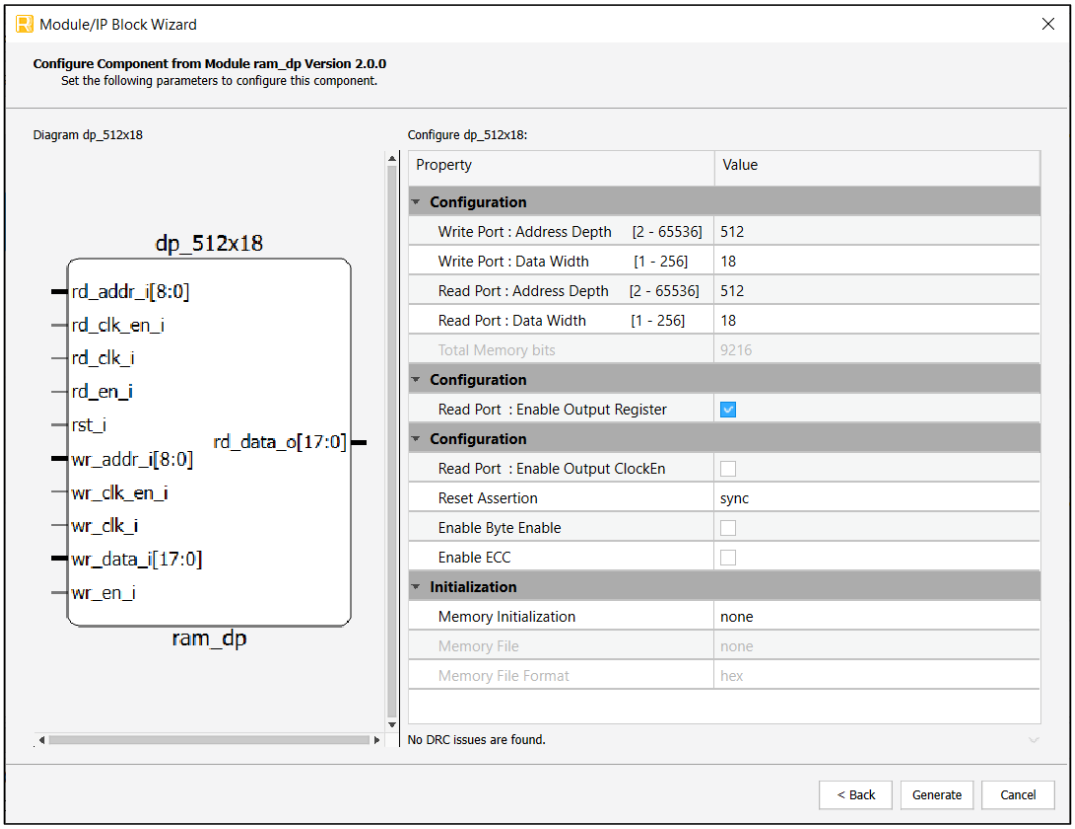

Figure 3.3. Example: Generating RAM_DP Module Customization

4. When all the options are set, click **Generate**. The **Check Generated Result** dialog box opens, showing design block messages and results as shown in Figure 3.4.

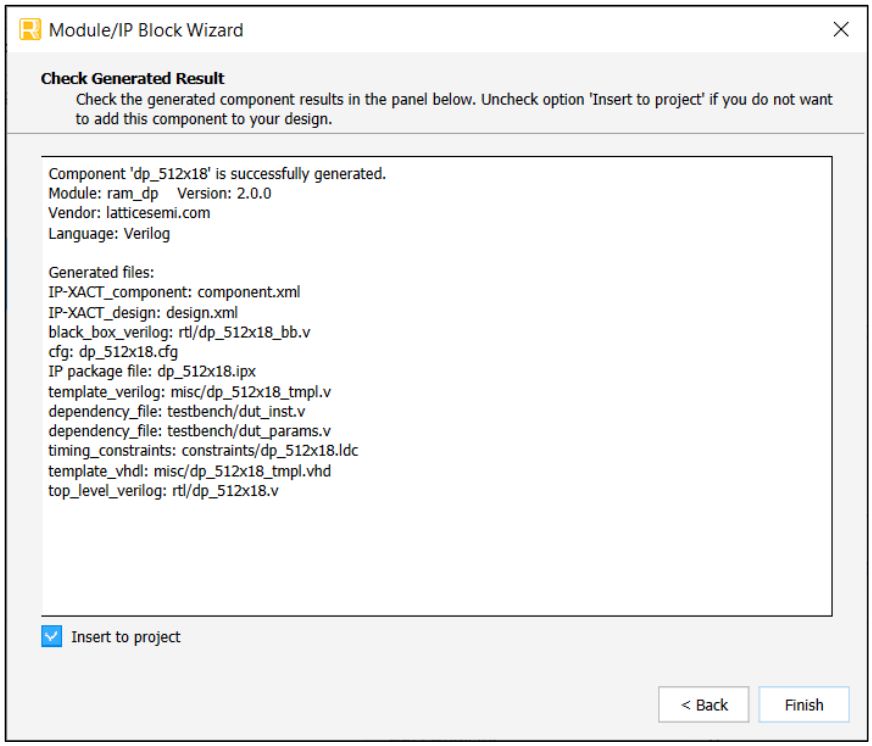

Figure 3.4. Example: Generating RAM_DP Check Generated Result

5. Click the **Finish** button. All the generated files are placed under the directory paths in the **Create in** and the **Component name** fields shown in Figure 3.2.

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

Table 3.1. Generated File List

File	Description
<Component name>.ipx	This file contains the information on the files associated to the generated IP.
<Component name>.cfg	This file contains the attribute values used in IP configuration.
component.xml	Contains the ipxact:component information of the IP.
design.xml	Documents the configuration attributes of the IP in IP-XACT 2014 format.
rtl/<Component name>.v	This file provides an example RTL top file that instantiates the IP core.
rtl/<Component name>_bb.v	This file provides the synthesis black box.
misc/<Component name>_tmpl.v misc /<Component name>_tmpl.vhd	These files provide instance templates for the IP core.

3.2. Running Functional Simulation

1. Click the  button located on the **Toolbar** then click **Next** to initiate the **Simulation Wizard** shown in Figure 3.5.

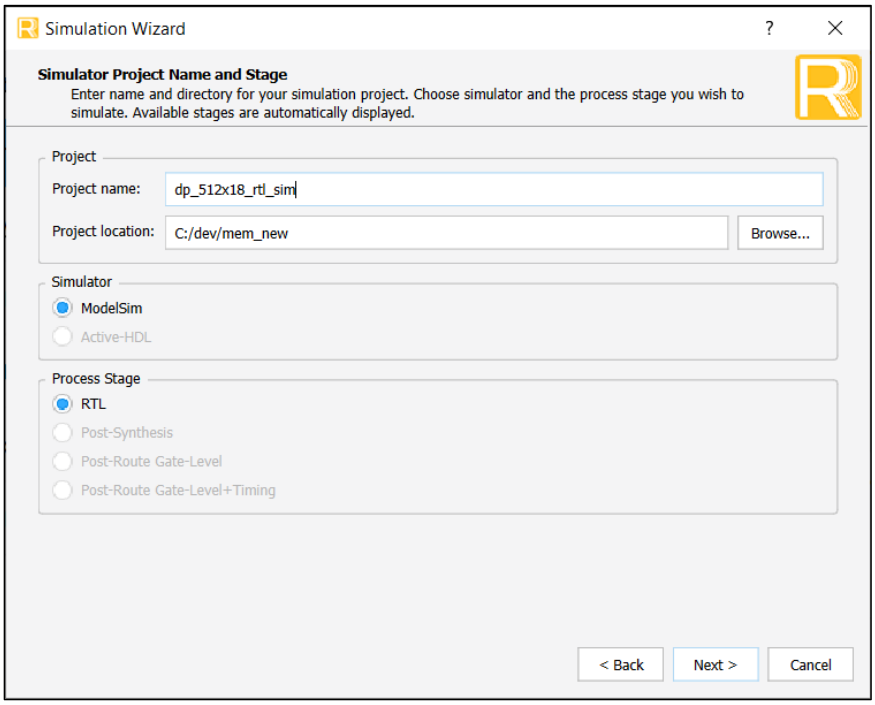

Figure 3.5. Simulation Wizard Simulator Project Name and Stage

2. Click **Next** to open the **Add and Reorder Source** window as shown in Figure 3.6.

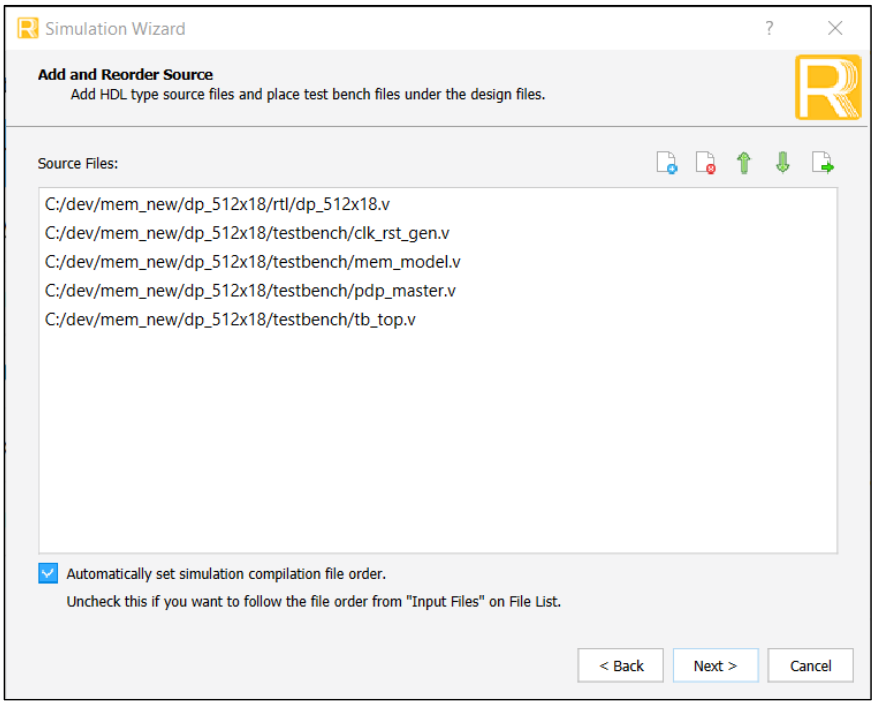

Figure 3.6. Simulation Wizard Add and Reorder Source

3. Click **Next** to open the **Parse HDL files for simulation**. Set **tb_top** as the **Simulation Top Module** as shown in Figure 3.7.

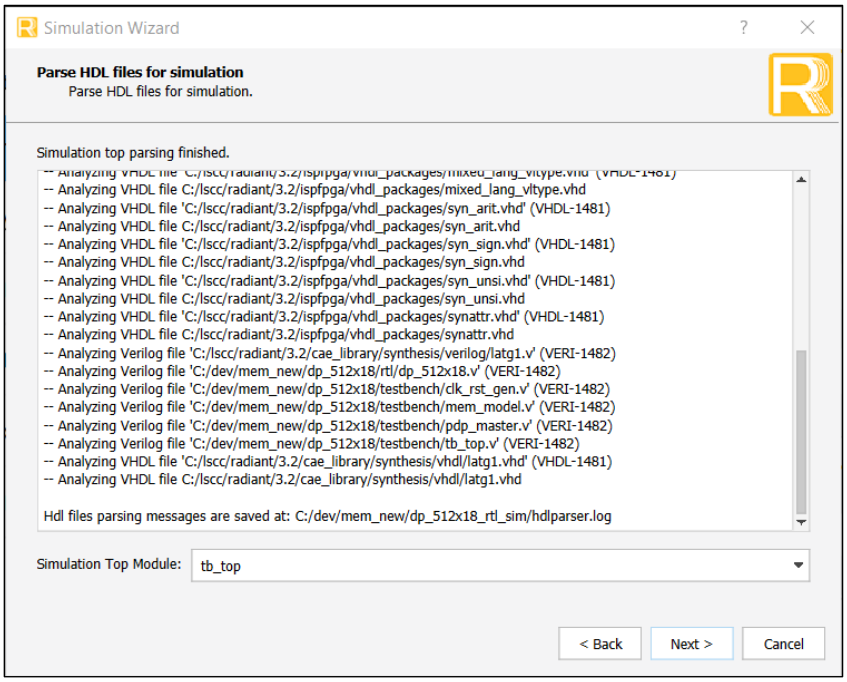

Figure 3.7. Simulation Wizard Parse HDL files for simulation

4. Click **Next**. The **Summary** window is shown. Click **Finish** to run the simulation. The complete waveform results of the simulation in this example are shown in Figure 3.8.

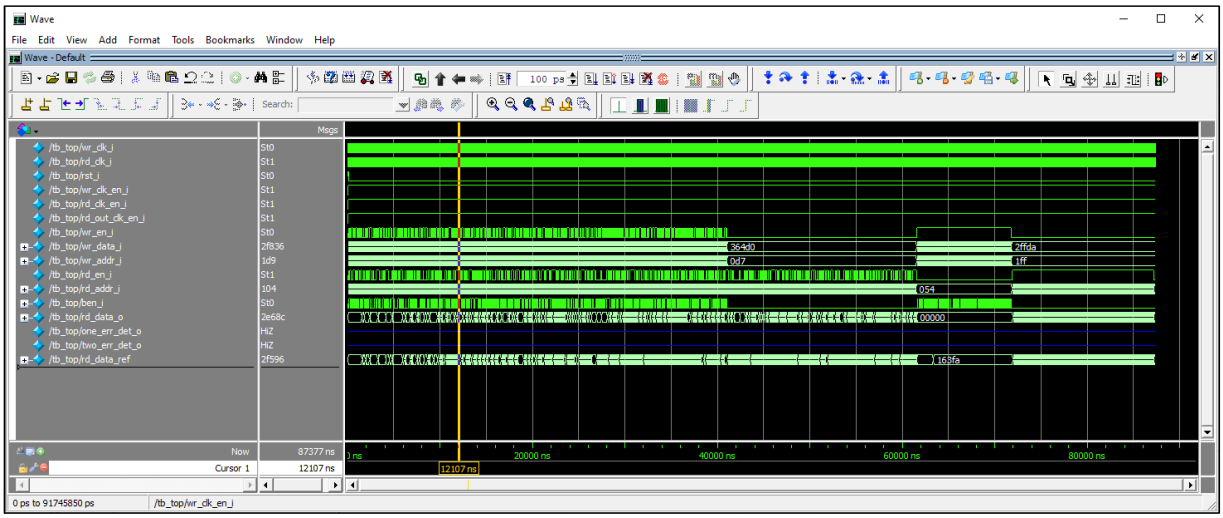

Figure 3.8. Simulation Waveform

3.3. Constraining the IP

To ensure that the design meets its desired performance goals on the FPGA, it is the responsibility of the users to provide proper timing and physical constraints. The contents of the following IP constraint file can be added to the user design constraints:

<IP Instance_Path>/<IP_Instance_Name>/eval/constraint.pdc

The above constraint file has been verified during IP evaluation with the IP instantiated directly in the top-level module. It can be modified but modifications should be made with thorough understanding of the effect of each constraint.

Refer to the Lattice Radiant Timing Constraints Methodology for details on how to constraint user design.

4. PMI Support

The following sections discuss the different PMI modules which can be utilized for entering custom parameters for the single port RAM, pseudo dual port RAM, true dual port RAM, read only memory and shift register. If parameters are left unspecified during module instantiation, default values are used. Some parameters here are unused and are added to maintain backward compatibility with older devices.

4.1. pmi_ram_dq

The following are port descriptions, Verilog and VHDL definitions, and parameter descriptions for pmi_ram_dq (Single Port Memory Module).

Table 4.1. Single Port Memory PMI Port Definitions

Direction	Port Name	Type	Size (Buses Only)
I	Address	Bus	(pmi_addr_width - 1):0
I	Data	Bus	(pmi_data_width - 1):0
I	Clock	Bit	N/A
I	ClockEn	Bit	N/A
I	Reset	Bit	N/A
I	WE	Bit	N/A
O	Q	Bus	(pmi_data_width - 1):0

Table 4.2. Single Port Memory PMI Attribute Definitions

Attributes	Description	Values	Default Value
pmi_addr_depth	Address depth of the read and write port	2–<Max that can fit in the device>	512
pmi_addr_width	Address width of the read and write port	1–<Max that can fit in the device>	9
pmi_data_width	Data word width of the Read and Write port	1–512	18
pmi_regmode	Data Out port (Q) can be registered or not using this selection.	“reg”, “noreg”	“reg”
pmi_gsr[1]	Sets if the global set/reset is enabled.	“enabled”,” disable”	“disable”
pmi_resetmode	Selection for the Reset to be Synchronous or Asynchronous to the Clock.	“async”, “sync”	“sync”
pmi_optimization[1]	Describes the synthesis directive if optimizing for area or speed	“speed”, “area”	“speed”
pmi_init_file	When Memory file is selected, user should set this parameter to the memory file.	<file_path>	“none”
pmi_init_file_format	This option allows the user to select if the memory file is formatted as Binary, Hex. (See the Initializing Memory section for details on different formats.)	“binary”, “hex”	“binary”
pmi_write_mode	Defines the write mode of the module.	“normal”	“normal”
pmi_family[2]	This option selects the product family used.	“iCE40UP”, "LIFCL", “LFD2NX”, “LFCPNX”, “LFMXO5”, “LAV-AT”, “common”	“common”
module_type[1]	Refers to the type of pmi module.	“pmi_ram_dq”	“pmi_ram_dq”

Notes:

1. Unused
2. Use pmi_family = “common” if initialization is needed. pmi_family = “common” requires 30 bits to be properly implemented (pmi_addr_depth × pmi_data_width ≥ 30).

Verilog pmi_ram_dq Definition

```
module pmi_ram_dq
  #(parameter pmi_addr_depth = 512,
    parameter pmi_addr_width = 9
    parameter pmi_data_width = 18,
    parameter pmi_regmode = "reg",
    parameter pmi_gsr = "disable",
    parameter pmi_resetmode = "sync",
    parameter pmi_optimization = "speed",
    parameter pmi_init_file = "none",
    parameter pmi_init_file_format = "binary",
    parameter pmi_write_mode = "normal",
    parameter pmi_family = "common",
    parameter module_type = "pmi_ram_dq")

    (input [(pmi_data_width-1):0] Data,
     input [(pmi_addr_width-1):0] Address,
     input  Clock,
     input  ClockEn,
     input  WE,
     input  Reset,
     output [(pmi_data_width-1):0] Q);

endmodule // pmi_ram_dq
```

VHDL pmi_ram_dq Definition

```
 component pmi_ram_dq is
     generic (
       pmi_addr_depth : integer := 512;
       pmi_addr_width : integer := 9;
       pmi_data_width : integer := 18;
       pmi_regmode : string := "reg";
       pmi_gsr : string := "disable";
       pmi_resetmode : string := "sync";
       pmi_optimization : string := "speed";
       pmi_init_file : string := "none";
       pmi_init_file_format : string := "binary";
       pmi_write_mode : string := "normal";
       pmi_family : string := "EC";
       module_type : string := "pmi_ram_dq"
    );
    port (
     Data : in std_logic_vector((pmi_data_width-1) downto 0);
     Address : in std_logic_vector((pmi_addr_width-1) downto 0);
     Clock: in std_logic;
     ClockEn: in std_logic;
     WE: in std_logic;
     Reset: in std_logic;
     Q : out std_logic_vector((pmi_data_width-1) downto 0)
   );
  end component pmi_ram_dq;
```

4.2. pmi_ram_dq_be

The following are port descriptions, Verilog and VHDL definitions, and parameter descriptions for pmi_ram_dq_be (Single Port Memory Module with Byte-Enable).

Table 4.3. Single Port Memory with Byte-Enable PMI Port Definitions

Direction	Port Name	Type	Size (Buses Only)
I	Address	Bus	(pmi_addr_width - 1):0
I	Data	Bus	(pmi_data_width - 1):0
I	Clock	Bit	N/A
I	ClockEn	Bit	N/A
I	Reset	Bit	N/A
I	WE	Bit	N/A
I	ByteEn	Bus	(pmi_data_width/ pmi_byte_size-1):0
O	Q	Bus	(pmi_data_width - 1):0

Table 4.4. Single Port Memory with Byte-Enable PMI Attribute Definitions

Attributes	Description	Values	Default Value
pmi_addr_depth	Address depth of the read and write port	2–<Max that can fit in the device>	512
pmi_addr_width	Address width of the read and write port	1–<Max that can fit in the device>	9
pmi_data_width	Data word width of the Read and Write port	1–512	18
pmi_regmode	Data Out port (Q) can be registered or not using this selection.	“reg”, “noreg”	“reg”
pmi_gsr[1]	Sets if the global set/reset is enabled.	“enabled”, “disable”	“disable”
pmi_resetmode	Selection for the Reset to be Synchronous or Asynchronous to the Clock.	“async”, “sync”	“sync”
pmi_optimization[1]	Describes the synthesis directive if optimizing for area or speed	“speed”, “area”	“speed”
pmi_init_file[1]	When Memory file is selected, user should set this parameter to the memory file.	<file_path>	“none”
pmi_init_file_format[1]	This option allows the user to select if the memory file is formatted as Binary, Hex. (See the Initializing Memory section for details on different formats.)	“binary”, “hex”	“binary”
pmi_write_mode	Defines the write mode of the module.	“normal”	“normal”
pmi_family[2]	This option selects the product family used.	“iCE40UP”, "LIFCL", “LFD2NX”, “LFCPNX”, “LFMXO5”, “LAV-AT”, “common”	“common”
pmi_byte_size	Byte size. The value should be 9 if pmi_data_width is divisible by 9, else 8.	8, 9	9
module_type[1]	Refers to the type of pmi module.	“pmi_ram_dq_be”	“pmi_ram_dq_be”

Notes:

1. Unused. WARNING: Do not change the value of pmi_init_file, this PMI does not support initialization.
2. Use pmi_family = “<device>”

Verilog pmi_ram_dq_be Definition

```
module pmi_ram_dq_be
  #(parameter pmi_addr_depth = 512,
    parameter pmi_addr_width = 9
    parameter pmi_data_width = 18,
    parameter pmi_regmode = "reg",
    parameter pmi_gsr = "disable",
    parameter pmi_resetmode = "sync",
    parameter pmi_optimization = "speed",
    parameter pmi_init_file = "none",
    parameter pmi_init_file_format = "binary",
    parameter pmi_write_mode = "normal",
    parameter pmi_family = "common",
    parameter pmi_byte_size = 9,
    parameter module_type = "pmi_ram_dq")

    (input [(pmi_data_width-1):0] Data,
     input [(pmi_addr_width-1):0] Address,
     input  Clock,
     input  ClockEn,
     input  WE,
     input  Reset,
     input [(pmi_data_width/pmi_byte_size-1):0] ByteEn,
     output [(pmi_data_width-1):0] Q);

endmodule // pmi_ram_dq_be
```

VHDL pmi_ram_dq_be Definition

```
component pmi_ram_dq_be is
     generic (
       pmi_addr_depth : integer := 512;
       pmi_addr_width : integer := 9;
       pmi_data_width : integer := 18;
       pmi_regmode : string := "reg";
       pmi_gsr : string := "disable";
       pmi_resetmode : string := "sync";
       pmi_optimization : string := "speed";
       pmi_init_file : string := "none";
       pmi_init_file_format : string := "binary";
       pmi_write_mode : string := "normal";
       pmi_family : string := "common";
       pmi_byte_size : integer := 9;
       module_type : string := "pmi_ram_dq"
    );
    port (
     Data : in std_logic_vector((pmi_data_width-1) downto 0);
     Address : in std_logic_vector((pmi_addr_width-1) downto 0);
     Clock: in std_logic;
     ClockEn: in std_logic;
     WE: in std_logic;
     Reset: in std_logic;
     ByteEn : in std_logic_vector((pmi_data_width/pmi_byte_size -1) downto 0);
```

```
    Q : out std_logic_vector((pmi_data_width-1) downto 0)
  );
 end component pmi_ram_dq_be;
```

4.3. pmi_ram_dp

The following are port descriptions, Verilog and VHDL definitions, and parameter descriptions for pmi_ram_dp (Pseudo Dual Port Memory Module).

Table 4.5. Pseudo Dual Port Memory PMI Port Definitions

Direction	Port Name	Type	Size (Buses Only)
I	WrAddress	Bus	(pmi_wr_addr_width - 1):0
I	RdAddress	Bus	(pmi_rd_addr_width - 1):0
I	Data	Bus	(pmi_wr_data_width - 1):0
I	RdClock	Bit	N/A
I	RdClockEn	Bit	N/A
I	Reset	Bit	N/A
I	WrClock	Bit	N/A
I	WrClockEn	Bit	N/A
I	WE	Bit	N/A
O	Q	Bus	(pmi_rd_data_width - 1):0

Table 4.6. Pseudo Dual Port Memory PMI Attribute Definitions

Attributes	Description	Values	Default Value
pmi_wr_addr_depth	Write Port Address depth.	2–<Max that can fit in the device>	512
pmi_wr_addr_width	Write Port Address width. Equal to $\log_2$(pmi_wr_addr_depth)	1–<Max that can fit in the device>	9
pmi_wr_data_width	Write Port Data word width.	1–512	18
pmi_rd_addr_depth	Read Port Address depth.	2–<Max that can fit in the device>	512
pmi_rd_addr_width	Read Port Address width. Equal to $\log_2$(pmi_rd_addr_depth)	1–<Max that can fit in the device>	9
pmi_rd_data_width	Read Port Data word width.	1–512	18
pmi_regmode	Data Out port (Q) can be registered or not using this selection.	“reg”, “noreg”	“reg”
pmi_gsr[1]	Sets if the global set/reset is enabled.	“enabled”, ”disable”	“disable”
pmi_resetmode	Selection for the Reset to be Synchronous or Asynchronous to the Clock.	“async”, “sync”	“sync”
pmi_optimization[1]	Describes the synthesis directive if optimizing for area or speed	“speed”, “area”	“speed”
pmi_init_file	When Memory file is selected, user should set this parameter to the memory file.	<file_path>	“none”
pmi_init_file_format	This option allows users to select if the memory file is formatted as Binary, Hex. (Check Sec.5 for details on different formats)	“binary”, “hex”	“binary”

Attributes	Description	Values	Default Value
pmi_family[2]	Defines the FPGA family being used in the module	“iCE40UP”, "LIFCL", “LFD2NX”, “LFCPNX”, “LFMXO5”, “LAV-AT”, “common”	“common”
module_type[1]	Refers to the type of pmi module.	“pmi_ram_dp”	“pmi_ram_dp”

Notes:

1. Unused.
2. Use pmi_family = “common” if initialization is needed. pmi_family = “common” requires 30 bits to be properly implemented (pmi_wr_addr_depth x pmi_wr_data_width ≥ 30) and does not support Mixed width. Mixed width + initialization is NOT supported.

Verilog `pmi_ram_dp` Definition

```
module pmi_ram_dp
  #(parameter pmi_wr_addr_depth = 512,
    parameter pmi_wr_addr_width = 9,
    parameter pmi_wr_data_width = 18,
    parameter pmi_rd_addr_depth = 512,
    parameter pmi_rd_addr_width = 9,
    parameter pmi_rd_data_width = 18,
    parameter pmi_regmode = "reg",
    parameter pmi_gsr = "disable",
    parameter pmi_resetmode = "sync",
    parameter pmi_optimization = "speed",
    parameter pmi_init_file = "none",
    parameter pmi_init_file_format = "binary",
    parameter pmi_family = "common",
    parameter module_type = "pmi_ram_dp")

    (input [(pmi_wr_data_width-1):0] Data,
     input [(pmi_wr_addr_width-1):0] WrAddress,
     input [(pmi_rd_addr_width-1):0] RdAddress,
     input  WrClock,
     input  RdClock,
     input  WrClockEn,
     input  RdClockEn,
     input  WE,
     input  Reset,
     output [(pmi_rd_data_width-1):0]  Q);

endmodule // pmi_ram_dp
```

VHDL pmi_ram_dp Definition

```
component pmi_ram_dp is
     generic (
       pmi_wr_addr_depth : integer := 512;
       pmi_wr_addr_width : integer := 9;
       pmi_wr_data_width : integer := 18;
       pmi_rd_addr_depth : integer := 512;
       pmi_rd_addr_width : integer := 9;
       pmi_rd_data_width : integer := 18;
       pmi_regmode : string := "reg";
       pmi_gsr : string := "disable";
       pmi_resetmode : string := "sync";
       pmi_optimization : string := "speed";
       pmi_init_file : string := "none";
       pmi_init_file_format : string := "binary";
       pmi_family : string := "common";
       module_type : string := "pmi_ram_dp"
    );
    port (
     Data : in std_logic_vector((pmi_wr_data_width-1) downto 0);
     WrAddress : in std_logic_vector((pmi_wr_addr_width-1) downto 0);
     RdAddress : in std_logic_vector((pmi_rd_addr_width-1) downto 0);
     WrClock: in std_logic;
     RdClock: in std_logic;
     WrClockEn: in std_logic;
     RdClockEn: in std_logic;
     WE: in std_logic;
     Reset: in std_logic;
     Q : out std_logic_vector((pmi_rd_data_width-1) downto 0)
   );
  end component pmi_ram_dp;
```

4.4. pmi_ram_dp_be

The following are port descriptions, Verilog and VHDL definitions, and parameter descriptions for pmi_ram_dp_be (Pseudo Dual Port Memory Module).

Table 4.7. Pseudo Dual Port Memory with Byte Enable PMI Port Definitions

Direction	Port Name	Type	Size (Buses Only)
I	WrAddress	Bus	(pmi_wr_addr_width - 1):0
I	RdAddress	Bus	(pmi_rd_addr_width - 1):0
I	Data	Bus	(pmi_wr_data_width - 1):0
I	RdClock	Bit	N/A
I	RdClockEn	Bit	N/A
I	Reset	Bit	N/A
I	WrClock	Bit	N/A
I	WrClockEn	Bit	N/A
I	WE	Bit	N/A
I	ByteEn	Bus	(pmi_data_width/ pmi_byte_size-1):0
O	Q	Bus	(pmi_rd_data_width - 1):0

Table 4.8. Pseudo Dual Port Memory with Byte Enable PMI Attribute Definitions

Attributes	Description	Values	Default Value
pmi_wr_addr_depth	Write Port Address depth.	2–<Max that can fit in the device>	512
pmi_wr_addr_width	Write Port Address width. Equal to $\log_2$(pmi_wr_addr_depth)	1–<Max that can fit in the device>	9
pmi_wr_data_width	Write Port Data word width.	1–512	18
pmi_rd_addr_depth	Read Port Address depth.	2–<Max that can fit in the device>	512
pmi_rd_addr_width	Read Port Address width. Equal to $\log_2$(pmi_rd_addr_depth)	1–<Max that can fit in the device>	9
pmi_rd_data_width	Read Port Data word width.	1–512	18
pmi_regmode	Data Out port (Q) can be registered or not using this selection.	“reg”, “noreg”	“reg”
pmi_gsr[1]	Sets if the global set/reset is enabled.	“enabled”, ”disable”	“disable”
pmi_resetmode	Selection for the Reset to be Synchronous or Asynchronous to the Clock.	“async”, “sync”	“sync”
pmi_optimization[1]	Describes the synthesis directive if optimizing for area or speed	“speed”, “area”	“speed”
pmi_init_file[1]	When Memory file is selected, user should set this parameter to the memory file.	<file_path>	“none”
pmi_init_file_format[1]	This option allows users to select if the memory file is formatted as Binary, Hex. (Check Sec.5 for details on different formats)	“binary”, “hex”	“binary”
pmi_family[2]	Defines the FPGA family being used in the module	“iCE40UP”, "LIFCL", “LFD2NX”, “LFCPNX”, “LFMXO5”, “LAV-AT”, “common”	“common”
pmi_byte_size	Byte size. The value should be 9 if pmi_data_width is divisible by 9, else 8.	8, 9	9
module_type[1]	Refers to the type of pmi module.	“pmi_ram_dp_be”	“pmi_ram_dp_be”

Notes:

1. Unused. WARNING: Do not change the value of pmi_init_file, this PMI does not support initialization.
2. Use pmi_family = “<device>”. Mixed-width byte-enable only applies to W/R ≤ 8 for Avant devices, W/R ≤ 4 for Nexus devices, and W/R ≤ 2 for iCE40UP device.

Verilog pmi_ram_dp_be Definition

```
module pmi_ram_dp_be
  #(parameter pmi_wr_addr_depth = 512,
    parameter pmi_wr_addr_width = 9,
    parameter pmi_wr_data_width = 18,
    parameter pmi_rd_addr_depth = 512,
    parameter pmi_rd_addr_width = 9,
    parameter pmi_rd_data_width = 18,
    parameter pmi_regmode = "reg",
    parameter pmi_gsr = "disable",
    parameter pmi_resetmode = "sync",
    parameter pmi_optimization = "speed",
    parameter pmi_init_file = "none",
    parameter pmi_init_file_format = "binary",
    parameter pmi_family = "common",
    parameter pmi_byte_size = 9,
    parameter module_type = "pmi_ram_dp")
```

```
    (input [(pmi_wr_data_width-1):0] Data,
     input [(pmi_wr_addr_width-1):0] WrAddress,
     input [(pmi_rd_addr_width-1):0] RdAddress,
     input  WrClock,
     input  RdClock,
     input  WrClockEn,
     input  RdClockEn,
     input  WE,
     input  Reset,
     input [(pmi_wr_data_width/pmi_byte_size)-1:0] ByteEn,
     output [(pmi_rd_data_width-1):0]  Q);

endmodule // pmi_ram_dp_be
```

VHDL pmi_ram_dp_be Definition

```
component pmi_ram_dp_be is
     generic (
       pmi_wr_addr_depth : integer := 512;
       pmi_wr_addr_width : integer := 9;
       pmi_wr_data_width : integer := 18;
       pmi_rd_addr_depth : integer := 512;
       pmi_rd_addr_width : integer := 9;
       pmi_rd_data_width : integer := 18;
       pmi_regmode : string := "reg";
       pmi_gsr : string := "disable";
       pmi_resetmode : string := "sync";
       pmi_optimization : string := "speed";
       pmi_init_file : string := "none";
       pmi_init_file_format : string := "binary";
       pmi_family : string := "common";
       pmi_byte_size : integer := 9;
       module_type : string := "pmi_ram_dp_be"
    );
    port (
     Data : in std_logic_vector((pmi_wr_data_width-1) downto 0);
     WrAddress : in std_logic_vector((pmi_wr_addr_width-1) downto 0);
     RdAddress : in std_logic_vector((pmi_rd_addr_width-1) downto 0);
     WrClock: in std_logic;
     RdClock: in std_logic;
     WrClockEn: in std_logic;
     RdClockEn: in std_logic;
     WE: in std_logic;
     Reset: in std_logic;
     ByteEn : in std_logic_vector((pmi_wr_data_width/pmi_byte_size -1) downto 0);
     Q : out std_logic_vector((pmi_rd_data_width-1) downto 0)
   );
  end component pmi_ram_dp_be;
```

4.5. pmi_rom

The following are port descriptions, Verilog and VHDL definitions, and parameter descriptions for pmi_rom (Read Only Memory Module).

Table 4.9. Read Only Memory PMI Port Definitions

Direction	Port Name	Type	Size (Buses Only)
I	Address	Bus	(pmi_ addr_width - 1):0
I	OutClock	Bus	N/A
I	OutClockEn	Bus	N/A
I	Reset	Bit	N/A
O	Q	Bus	(pmi_data_width - 1):0

Table 4.10. Read Only Memory PMI Attribute Definitions

Attributes	Description	Values	Default Value
pmi_addr_depth	Address depth.	2–<Max that can fit in the device>	512
pmi_addr_width	Address width. Equal to $\log_2$(pmi_wr_addr_depth)	1–<Max that can fit in the device>	9
pmi_data_width	Data word width.	1–512	18
pmi_regmode	Data Out port (Q) can be registered or not using this selection.	“reg”, “noreg”	“reg”
pmi_gsr[1]	Sets if the global set/reset is enabled.	“enabled”, ”disable”	“disable”
pmi_resetmode	Selection for the Reset to be Synchronous or Asynchronous to the Clock.	“async”, “sync”	“sync”
pmi_optimization[1]	Describes the synthesis directive if optimizing for area or speed	“speed”, “area”	“speed”
pmi_init_file	When Memory file is selected, user should set this parameter to the memory file.	<file_path>	“none”
pmi_init_file_format	This option allows users to select if the memory file is formatted as Binary, Hex. (Check Sec.5 for details on different formats)	“binary”, “hex”	“binary”
pmi_family[2]	Defines the FPGA family being used in the module	“iCE40UP”, "LIFCL", “LFD2NX”, “LFCPNX”, “LFMXO5”, “LAV-AT”, “common”	“common”
module_type[1]	Refers to the type of pmi module.	“pmi_rom”	“pmi_rom”

Notes:

1. Unused
2. Use only pmi_family = “common”. This pmi requires 30 bits to be properly implemented (pmi_addr_depth x pmi_data_width >= 30).

Verilog `pmi_rom` Definition

```
module pmi_rom # (
    parameter pmi_addr_depth = 512,
    parameter pmi_addr_width = 9
    parameter pmi_data_width = 8,
    parameter pmi_regmode = "reg",
    parameter pmi_gsr = "disable",
    parameter pmi_resetmode = "sync",
    parameter pmi_optimization = "speed",
    parameter pmi_init_file = "none",
```

```
    parameter pmi_init_file_format = "binary",
    parameter pmi_family = "common",
    parameter module_type = "pmi_rom")

    (input [(pmi_addr_width-1):0] Address,
     input OutClock,
     input OutClockEn,
     input Reset,
     output [(pmi_data_width-1):0] Q)
;

endmodule // pmi_rom
```

VHDL `pmi_rom` Definition

```
component pmi_rom is
     generic (
       pmi_addr_depth : integer := 512;
       pmi_addr_width : integer := 9;
       pmi_data_width : integer := 8;
       pmi_regmode : string := "reg";
       pmi_gsr : string := "disable";
       pmi_resetmode : string := "sync";
       pmi_optimization : string := "speed";
       pmi_init_file : string := "none";
       pmi_init_file_format : string := "binary";
       pmi_family : string := "common";
       module_type : string := "pmi_rom"
    );
    port (
     Address : in std_logic_vector((pmi_addr_width-1) downto 0);
     OutClock: in std_logic;
     OutClockEn: in std_logic;
     Reset: in std_logic;
     Q : out std_logic_vector((pmi_data_width-1) downto 0)
   );
  end component pmi_rom;
```

4.6. pmi_ram_dp_true

The following are port descriptions, Verilog and VHDL definitions, and parameter descriptions for pmi_ram_dp_true (True Dual Port Memory Module).

Table 4.11. True Dual Port Memory PMI Port Definitions

Direction	Port Name	Type	Size (Buses Only)
I	AddressA	Bus	(pmi_addr_width_a - 1):0
I	AddressB	Bus	(pmi_addr_width_b - 1):0
I	DataInA	Bus	(pmi_data_width_a – 1):0
I	DataInB	Bus	(pmi_data_width_b – 1):0
I	ClockA	Bit	N/A
I	ClockB	Bit	N/A
I	ClockEnA	Bit	N/A
I	ClockEnB	Bit	N/A
I	WrA	Bit	N/A
I	WrB	Bit	N/A
I	ResetA	Bit	N/A
I	ResetB	Bit	N/A
O	QA	Bus	(pmi_ data_width_a - 1):0
O	QB	Bus	(pmi_ data_width_b - 1):0

Table 4.12. True Dual Port Memory PMI Attribute Definitions

Attributes	Description	Values	Default Value
pmi_addr_depth_a	Port A Address depth.	2–<Max that can fit in the device>	512
pmi_addr_width_a	Port A Address width. Equal to $\log_2$(pmi_addr_depth_a).	1–<Max that can fit in the device>	9
pmi_data_width_a	Port A Data word width.	1–512	18
pmi_addr_depth_b	Port B Address depth.	2–<Max that can fit in the device>	512
pmi_addr_width_b	Port B Address width. Equal to $\log_2$(pmi_addr_depth_b).	1–<Max that can fit in the device>	9
pmi_data_width_b	Port B Data word width.	1–512	18
pmi_regmode_a	Data Out for port A (Q) can be registered or not using this selection.	“reg”, “noreg”	“reg”
pmi_regmode_b	Data Out for port B (Q) can be registered or not using this selection.	“reg”, “noreg”	“reg”
pmi_gsr[1]	Sets if the global set/reset is enabled.	“enabled”, ”disable”	“disable”
pmi_resetmode	Selection for the Reset to be Synchronous or Asynchronous to the Clock.	“async”, “sync”	“sync”
pmi_optimization[1]	Describes the synthesis directive if optimizing for area or speed	“speed”, “area”	“speed”
pmi_init_file	When Memory file is selected, user should set this parameter to the memory file.	<file_path>	“none”
pmi_init_file_format	This option allows users to select if the memory file is formatted as Binary, Hex. (Check Sec.5 for details on different formats)	“binary”, “hex”	“binary”
pmi_write_mode_a	Defines the write mode of the module’s port A.	“normal”	“normal”

Attributes	Description	Values	Default Value
pmi_write_mode_b	Defines the write mode of the module's port B.	"normal"	"normal"
pmi_family[2]	Defines the FPGA family being used in the module	"iCE40UP", "LIFCL", "LFD2NX", "LFCPNX", "LFMXO5", "LAV-AT", "common"	"common"
module_type[1]	Refers to the type of pmi module.	"pmi_ram_dp_true"	"pmi_ram_dp_true"

Notes:

1. Unused
2. Use pmi_family = "common" if initialization is needed. pmi_family = "common" requires 30 bits to be properly implemented (pmi_ addr_depth_a x pmi_data_width_a ≥ 30) and does not support Mixed width. Mixed width + initialization is NOT supported.

Verilog pmi_ram_dp_true Definition

```
module pmi_ram_dp_true
  #(parameter pmi_addr_depth_a = 512,
    parameter pmi_addr_width_a = 9,
    parameter pmi_data_width_a = 18,
    parameter pmi_addr_depth_b = 512,
    parameter pmi_addr_width_b = 9,
    parameter pmi_data_width_b = 18,
    parameter pmi_regmode_a = "reg",
    parameter pmi_regmode_b = "reg",
    parameter pmi_gsr = "disable",
    parameter pmi_resetmode = "sync",
    parameter pmi_optimization = "speed",
    parameter pmi_init_file = "none",
    parameter pmi_init_file_format = "binary",
    parameter pmi_write_mode_a = "normal",
    parameter pmi_write_mode_b = "normal",
    parameter pmi_family = "common",
    parameter module_type = "pmi_ram_dp_true")

    (input [(pmi_data_width_a-1):0]    DataInA,
     input [(pmi_data_width_b-1):0]    DataInB,
     input [(pmi_addr_width_a-1):0] AddressA,
     input [(pmi_addr_width_b-1):0] AddressB,
     input  ClockA,
     input  ClockB,
     input  ClockEnA,
     input  ClockEnB,
     input  WrA,
     input  WrB,
     input  ResetA,
     input  ResetB,
     output [(pmi_data_width_a-1):0]  QA,
     output [(pmi_data_width_b-1):0]  QB);

endmodule // pmi_ram_dp_true
```

VHDL pmi_ram_dp_true Definition

```
component pmi_ram_dp_true is
     generic (
       pmi_addr_depth_a : integer := 512;
       pmi_addr_width_a : integer := 9;
       pmi_data_width_a : integer := 18;
       pmi_addr_depth_b : integer := 512;
       pmi_addr_width_b : integer := 9;
       pmi_data_width_b : integer := 18;
       pmi_regmode_a : string := "reg";
       pmi_regmode_b : string := "reg";
       pmi_gsr : string := "disable";
       pmi_resetmode : string := "sync";
       pmi_optimization : string := "speed";
       pmi_init_file : string := "none";
       pmi_init_file_format : string := "binary";
       pmi_write_mode_a : string := "normal";
       pmi_write_mode_b : string := "normal";
       pmi_family : string := "common";
       module_type : string := "pmi_ram_dp_true"
    );
    port (
     DataInA : in std_logic_vector((pmi_data_width_a-1) downto 0);
     DataInB : in std_logic_vector((pmi_data_width_b-1) downto 0);
     AddressA : in std_logic_vector((pmi_addr_width_a-1) downto 0);
     AddressB : in std_logic_vector((pmi_addr_width_b-1) downto 0);
     ClockA: in std_logic;
     ClockB: in std_logic;
     ClockEnA: in std_logic;
     ClockEnB: in std_logic;
     WrA: in std_logic;
     WrB: in std_logic;
     ResetA: in std_logic;
     ResetB: in std_logic;
     QA : out std_logic_vector((pmi_data_width_a-1) downto 0);
     QB : out std_logic_vector((pmi_data_width_b-1) downto 0)
   );
  end component pmi_ram_dp_true;
```

4.7. pmi_distributed_shift_reg

The following are port descriptions, Verilog and VHDL definitions, and parameter descriptions for pmi_distributed_shift_reg (Shift Register Memory Module).

Table 4.13. Shift Register Memory PMI Port Definitions

Direction	Port Name	Type	Size (Buses Only)
I	Addr	Bus	(pmi_max_width - 1):0
I	Clock	Bit	N/A
I	ClockEn	Bit	N/A
I	Reset	Bit	N/A
I	Q	Bus	(pmi_data_width – 1):0

Table 4.14. Shift Register Memory PMI Attribute Definitions

Attributes	Description	Values	Default Value
pmi_data_width	Data width for shift register.	2–<Max that can fit in the device>	8
pmi_max_shift	Maximum shift register size.	2–<Max that can fit in the device>	16
pmi_max_width	Maximum shift register width size. Equal to $\log_2$(pmi_max_shift + 1).	1–512	5
pmi_num_shift	Maximum amount, which can be shifted.	2–<Max that can fit in the device>	16
pmi_num_width	Maximum shift amount width size. Equal to $\log_2$(pmi_num_shift + 1).	1–512	5
pmi_regmode	Data Out can be registered or not using this selection.	“reg”, “noreg”	“reg”
pmi_shiftreg_type	Type of shift register.	“fixed”, “variable”	“fixed”
pmi_family	Defines the FPGA family being used in the module	"LIFCL", “LFD2NX”, “LFCPNX”, “LFMXO5”, “LAV-AT”, “common”	“common”
module_type	Refers to the type of pmi module.	“pmi_distributed_shift_reg”	“pmi_distributed_shift_reg”

Verilog pmi_distributed_shift_reg reg Definition

```
module pmi_distributed_shift_reg
  #(parameter pmi_data_width = 8,
    parameter pmi_regmode = "reg",
    parameter pmi_shiftreg_type = "fixed",
    parameter pmi_num_shift = 16,
    parameter pmi_num_width = 4,
    parameter pmi_max_shift = 16,
    parameter pmi_max_width = 4,
    parameter pmi_init_file = "none",
    parameter pmi_init_file_format = "binary",
    parameter pmi_family = "common",
    parameter module_type = "pmi_distributed_shift_reg")

    (
     input [(pmi_data_width-1):0] Din,
     input [(pmi_max_width-1):0]  Addr,
     input Clock,
     input ClockEn,
     input Reset,
```

```
    output [(pmi_data_width-1):0] Q);

endmodule // pmi_distributed_shift_reg
```

VHDL pmi_distributed_shift_reg Definition

```
  component pmi_distributed_shift_reg is
     generic (
       pmi_data_width : integer := 8;
       pmi_regmode : string := "reg";
       pmi_shiftreg_type : string := "fixed";
       pmi_num_shift : integer := 16;
       pmi_num_width : integer := 4;
       pmi_max_shift : integer := 16;
       pmi_max_width : integer := 4;
       pmi_init_file : string := "none";
       pmi_init_file_format : string := "binary";
       pmi_family : string := "common";
       module_type : string := "pmi_distributed_shift_reg"
    );
    port (
     Din : in std_logic_vector((pmi_data_width-1) downto 0);
     Addr : in std_logic_vector((pmi_max_width-1) downto 0);
     Clock: in std_logic;
     ClockEn: in std_logic;
     Reset: in std_logic;
     Q : out std_logic_vector((pmi_data_width-1) downto 0)
   );
  end component pmi_distributed_shift_reg;
```

5. Initializing Memory

In each memory mode, it is possible to specify the power-on state of each bit in the memory array. This allows the memory to be used as ROM if desired. Each bit in the memory array can have a value of 0 or 1.

5.1. Initialization File Formats

The initialization file is an ASCII file, which user can create or edit using any ASCII editor. Module/IP Block Wizard supports the binary and hex memory file formats.

Each row includes the value to be stored in a particular memory location. The number of characters (or the number of columns) represents the number of bits for each address (or the width of the memory module). The file used for the memory initialization could be any text file, regardless of file extension, which follows the format described above.

The memory initialization can be static or dynamic. In the case of static initialization, the memory values are stored in the bitstream. Dynamic initialization of memories involves memory values stored in the external flash and can be updated by user logic knowing the memory address locations. The size of the bitstream (bit or rbt file) is larger due to static values stored in them.

The initialization file is primarily used for configuring the ROMs. RAMs can also use the initialization file to preload memory contents.

5.2. Binary File

The binary file is a text file of 0 and 1. The rows indicate the number of words, and the columns indicate the width of the memory.

Memory Size 20 × 32

```
00100000010000000010000001000000
00000001000000010000000100000001
00000010000000100000001000000010
00000011000000110000001100000011
00000100000001000000010000000100
00000101000001010000010100000101
00000110000001100000011000000110
00000111000001110000011100000111
00001000010010000000100001001000
00001001010010010000100101001001
00001010010010100000101001001010
00001011010010110000101101001011
00001100000011000000110000001100
00001101001011010000110100101101
00001110001111100000111000111110
00001111001111110000111100111111
00010000000100000001000000010000
00010001000100010001000100010001
00010010000100100001001000010010
00010011000100110001001100010011
```

5.3. Hex File

The hex file is a text file of hexadecimal characters in a similar row-column arrangement. The number of rows in the file is the same as the number of address locations, with each row indicating the content of the memory location.

Memory Size 8 × 16

```
A001
0B03
1004
CE06
0007
040A
0017
02A4
```

Appendix A. Resource Utilization

Table A.1. Device and Tool Tested

	Value
Lattice Radiant Software Version	Radiant Software 2022.1 (for Windows)
Device Used	LAV-AT-E70-2LFG1156C
Synthesis Tool	Synplify Pro (R) S-2021.09LR-SP2, Build 164R, Oct 18 2022

Table A.2. EBR-Based Single Port Memory Utilization

Memory Size	REG_EN	Byte-EN	Synthesis Tool	Register	LUTs	EBR
512 × 36	“reg”	No	Synplify Pro	0	2	1
16,384 × 32	“reg”	Yes	Synplify Pro	0	5	16
4,096 × 128	“noreg”	No	Synplify Pro	0	2	15

Table A.3. EBR-Based Pseudo Dual Port Memory Utilization

Write CONFIG	Read CONFIG	REG_EN	Byte-EN	Synthesis Tool	Register	LUTs	EBR
512 × 36	512 x 36	“reg”	No	Synplify Pro	1	2	1
16,384 × 32	4,096 x 128	“reg”	Yes	Synplify Pro	6	281	16
2,048 × 256	16,384 x 32	“noreg”	No	Synplify Pro	0	0	16

Table A.4. EBR-Based True Dual Port Memory Utilization

Port A CONFIG	Read CONFIG	REG A	REG B	Byte-EN A	Byte-EN B	Synthesis Tool	Register	LUTs	EBR
512 × 18	512 x 18	“reg”	“reg”	No	No	Synplify Pro	0	0	1
16,384 × 16	8,192 x 32	“reg”	“reg”	Yes	Yes	Synplify Pro	8	114	8
8,192 × 32	32,768 x 8	“noreg”	“noreg”	No	No	Synplify Pro	0	0	8

Table A.5. EBR-Based Read-Only Memory Utilization

Memory Size	REG_EN	Synthesis Tool	Register	LUTs	EBR
1,024 × 32	“reg”	Synplify Pro	0	2	1
16,384 x 16	“noreg”	Synplify Pro	0	2	8

Table A.6. Shift Register Utilization

Memory Size	REG_EN	SHIFT TYPE	Memory Type	Synthesis Tool	Register	LUTs	EBR
16 × 8	“reg”	fixed	EBR	Synplify Pro	5	12	1
8,192 × 24	“noreg”	variable	EBR	Synplify Pro	28	42	12
256 × 16	“noreg”	fixed	LUT	Synplify Pro	9	1029	0

Table A.7. Device and Tool Tested

	Value
Lattice Radiant Software Version	Radiant Software 2023.2 (for Windows)
Device Used	LIFCL-40-8BG400C
Synthesis Tool	Synplify Pro (R) U-2023.03LR-SP1-Beta2, Build 167R, Aug 29 2023

Table A.8. Single Port Large RAM (Large_RAM_SP) Utilization

Memory Size	REG_EN	Byte-EN	Synthesis Tool	Register	LUT4s	Large RAMs
16,384 × 32	“reg”	Yes	Synplify Pro	2	1	1
16,384 × 64	“noreg”	No	Synplify Pro	1	2	2

Table A.9.Pseudo Dual Port Large RAM (Large_RAM_DP) Utilization

Memory Size	REG_EN	Byte-EN	Synthesis Tool	Register	LUT4s	Large RAMs
16,384 × 32	“reg”	Yes	Synplify Pro	2	1	1
16,384 × 64	“noreg”	No	Synplify Pro	1	2	2

Table A.10. True Dual Port Large RAM (Large_RAM_DP_True) Utilization

Memory Size	REG_EN	Byte-EN	Synthesis Tool	Register	LUT4s	Large RAMs
16,384 × 32	“reg”	Yes	Synplify Pro	4	259	1
16,384 × 64	“noreg”	No	Synplify Pro	2	260	2

Table A.11. Single Port Large RAM (Large_RAM_SP) Utilization

Memory Size	REG_EN	Byte-EN	Synthesis Tool	Register	LUT4s	Large RAMs
16,384 × 32	“reg”	Yes	Synplify Pro	2	1	1
16,384 × 64	“noreg”	No	Synplify Pro	2	2	2

Appendix B. Limitations

Here are the identified limitations:

- PMI_* initialization can only be supported when using pmi_family = “common”.
- PMI_RAM_DP and PMI_RAM_DP_TRUE mixed-width configurations can only be supported when using pmi_family = “LAV-AT”.
- For RAM_DP and RAM_DP_True, the behavior is not guaranteed if writing and reading at the same address simultaneously (that is, either the read or write may happen first).
- For RAM_DP of Nexus devices, the simulation fails due to simulation model limitation. The signal one_err_det_o is always high when set to ECC enabled.

References

- Lattice Avant Platform web page
- Lattice Nexus Platform web page
- Lattice iCE40 UltraPlus web page
- Lattice Solutions IP Cores web page
- Lattice Radiant Timing Constraints Methodology
- Lattice Radiant FPGA design software
- Lattice Insights web page for Lattice Semiconductor training courses and learning plans

Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.

For frequently asked questions, refer to the Lattice Answer Database at www.latticesemi.com/Support/AnswerDatabase.

Revision History

Revision 1.5, February 2025

Section	Change Summary
All	Minor editorial fixes.

Memory Modules	• Updated the tables below to add the description, *active low (0)* for ben_i signal.
	  • Table 2.1. EBR-Based Single Port Memory Port Definitions
	  • Table 2.3. EBR-Based Pseudo Dual Port Memory Port Definitions
	  • Table 2.11. Large-RAM based Single-Port Memory Port Definitions
	  • Table 2.14. Large RAM-Based Pseudo Dual-Port Memory Port Definitions
	• Updated Figure 2.9. True Dual Port Memory Timing Diagram, Without Output Registers.
	• Updated Figure 2.23. Single Port Large RAM Timing Waveform (Normal Mode), With Output Registers.

Revision 1.4, October 2024

Section	Change Summary
Abbreviations in This Document	Changed *acronyms* to *abbreviations*.
Memory Modules	Updated the statement, *The Reset (RST) signal only resets input and output registers of the memory module* to *The Reset (RST) signal only resets output registers of the memory module*.

Revision 1.3, February 2024

Section	Change Summary
Introduction	Replaced *technical note* with user guide and removed *the LAV-AT Family of* from the sentence: *This user guide discusses EBR memory usage for devices supported by Lattice Radiant™ Software.*

PMI Support	• Added *"iCE40UP"*, *"LIFCL"*, *"LFD2NX"*, *"LFCPNX"*, and *"LFMXO5"* values to *pmi_family* attributes to the following tables:
	  • Table 4.2. Single Port Memory PMI Attribute Definitions
	  • Table 4.4. Single Port Memory with Byte-Enable PMI Attribute Definitions
	  • Table 4.6. Pseudo Dual Port Memory PMI Attribute Definitions
	  • Table 4.8. Pseudo Dual Port Memory with Byte Enable PMI Attribute Definitions
	  • Table 4.10. Read Only Memory PMI Attribute Definitions
	  • Table 4.12. True Dual Port Memory PMI Attribute Definitions
	  • Table 4.14. Shift Register Memory PMI Attribute Definitions
	• Updated *Note 2* of the following tables:
	  • Table 4.4. Single Port Memory with Byte-Enable PMI Attribute Definitions
	  • Table 4.6. Pseudo Dual Port Memory PMI Attribute Definitions
	  • Table 4.8. Pseudo Dual Port Memory with Byte Enable PMI Attribute Definitions
	  • Table 4.12. True Dual Port Memory PMI Attribute Definitions

References	Added this section.

Revision 1.2, November 2023

Section	Change Summary
Disclaimers	Updated this section.

Memory Modules	• Updated below figures:
	  • Figure 2.2. Single Port Memory Timing Diagram, Without Output Registers
	  • Figure 2.3. Single Port Memory Timing Diagram, With Output Registers
	  • Figure 2.5. Pseudo Dual Port Memory Timing Diagram, Without Output Registers
	  • Figure 2.6. Pseudo Dual Port Memory Timing Diagram, With Output Registers
	  • Figure 2.9. True Dual Port Memory Timing Diagram, Without Output Registers

Section	Change Summary
	• Figure 2.10. True Dual Port Memory Timing Diagram, With Output Registers • Figure 2.11. True Dual Port Memory Timing Diagram, Mixed Width Without Output Registers • Figure 2.13. Read Only Memory Timing Diagram, Without Output Registers • Figure 2.14. Read Only Memory Timing Diagram, With Output Registers • Added below sections: • Single Port Large RAM (Large_RAM_SP) • Pseudo Dual Port Large RAM (Large_RAM_DP) • True Dual-Port Large RAM (Large_RAM_DP True) • Large Read-Only Memory (Large_ROM)
IP Generation, Simulation, and Validation	• Updated title of the section from *IP Generation* to *IP Generation, Simulation, and Validation*. • Added Constraining the IP section.
Appendix A. Resource Utilization	• Added Table A.7. Device and Tool Tested, Table A.8. Single Port Large RAM (Large_RAM_SP) Utilization, Table A.9.Pseudo Dual Port Large RAM (Large_RAM_DP) Utilization, Table A.10. True Dual Port Large RAM (Large_RAM_DP_True) Utilization, and Table A.11. Single Port Large RAM (Large_RAM_SP) Utilization. • Replaced *LAV-AT-500E-2LFG1156C* with *LAV-AT-E70-2LFG1156C* in Table A.1. Device and Tool Tested.
Appendix B. Limitations	Added sentence *For RAM_DP of Nexus devices, the simulation fails due to simulation model limitation. The signal one_err_det_o is always high when set to ECC enabled*.

Revision 1.1, June 2023

Section	Change Summary
Memory Modules	Added another row after Memory File Format Attributes in the following tables. • Table 2.2. EBR-Based Single Port Memory Attribute Definitions • Table 2.4. EBR-Based Pseudo Dual Port Memory Attribute Definitions • Table 2.6. EBR-Based True Dual Port Memory Attribute Definitions • Table 2.8. EBR-Based Read Only Memory Attribute Definitions
Technical Support Assistance	Added reference to the Lattice Answer Database on the Lattice website.

Revision 1.0, November 2022

Section	Change Summary
All	Initial release

www.latticesemi.com