LIFCL Large RAM Module - Lattice Radiant Software

User Guide

FPGA-IPUG-02068-1.0

December 2019

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 and with all faults, and all risk associated with such information is entirely with Buyer. Buyer shall not rely on any data and performance specifications or parameters provided herein. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. No Lattice products should be used in conjunction with mission- or safety-critical or any other application in which the failure of Lattice's product could create a situation where personal injury, death, severe property or environmental damage may occur. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
AW	Address Width
DW	Data Width
ECC	Error Correcting Code
FPGA	Field-Programmable Gate Array
LMMI	Lattice Memory Mapped Interface
LRAM	Large Random-Access Memory
NB	Number of Bytes
SECDED	Double Error Detection
SRAM	Single-Port Random Access Memory

1. Introduction

The Lattice Semiconductor Large Random-Access Memory (LRAM) Module is designed to work as Single-Port RAM, Dual-Port RAM, Pseudo Dual-Port RAM, and Read-Only Memory (ROM). The architecture of these devices provides many resources for memory-intensive applications.

LRAM Module can be targeted to all CrossLink-NX™ FPGA devices. The design is implemented in Verilog HDL using Lattice Radiant® Software and Route tool integrated with Synplify Pro® synthesis tool. Attributes available for configuration from the user interface are shown in Table 2.3, Table 2.5, Table 2.7, and Table 2.9.

1.1. Features

Large RAM Module is designed to work as a Single-Port RAM, a Dual-Port RAM, a Pseudo Dual-Port RAM, and as a ROM.

The main features of the LRAM Module are:

- 524288 bits (0.5 M) per LRAM core (LRAM Module can have up to two cores)
- Single-Port Memory and Dual-Port Memory with shared clock
- Read Only Memory
- Data width from 1 to 64 bits
- Error Correcting Code (ECC) or Byte Enable selection (mutually exclusive)
- Sleep Mode support with optional memory data retention
- Output data registers (by passable)
- Output register retention
- Unaligned read data (for RISC-V compressed instruction)
- Width conversion
- Support three types of Write Modes: Normal, Write Through and Read Before Write.

This Module supports the following power management:

- Power down module logic
- Power down SRAM periphery logic
- Power down SRAM memory array when retention is not enabled

1.2. Conventions

1.2.1. Nomenclature

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

1.2.2. Signal Names

Signal names that end with:

- _*n* are active low
- _*i* are input signals
- _*o* are output signals

2. Memory Modules

2.1. Overview

In addition to basic memory functions, the Large RAM module can perform the following operations, and features. Each special feature is discussed below.

2.1.1. Error Correction Code and Byte Enable

Error-Correcting Code is a system that adds redundant data, or parity data, to a message, such that a receiver can recover it even when a number of errors were introduced, either during the process of transmission, or on storage. This Module design implements an ECC module for the Lattice FPGA families that can be applied to increase memory reliability in critical applications. The ECC module provides Single Error Correction – Double Error Detection (SECDED) capability based on a class of optimal minimum odd weight error parity codes that provides better performance than typical Hamming-based SECDED codes. ECC syndrome is calculated over all 4 bytes of data. If you incorrectly enable byte write together with ECC, the inner ECC is disabled and byte write still works correctly.

Byte Enable feature enables you to mask the bytes written in the RAM. The Byte Enable control can be per 8-bits; the selection can be made in Module/IP Block Wizard, while generating the module.

Byte Enable and ECC are mutually exclusive and cannot be used together.

2.1.2. Unaligned Read

The large RAM IP supports RISC-V compressed instruction chunks of data and shift them. If RISC-V sometimes needs to read 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. Unaligned Read enables this feature; Table 2.1 shows the possible combinations for Unaligned Read shifting. The unaligned read pins are shared with the ben[x]_i ports; there are no dedicated unaligned read ports for the Large RAM module. With this given, 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 2.1. Shift functions of unaligned read

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}

DOx refers to DOA/DOB output port of the Large RAM.

Unaligned Read is only supported if DATA_WIDTH = 32 for all ports.

2.1.3. Using Various Data Widths on Various Ports

When LRAM memory is configured as True Dual-Port or Pseudo Dual-Port memory, it has separate Data Width (DW) and Address Width (AW) parameters for ports A and B. The parameters can be configured independently; however, there are a few constraints for their values.

- Data Width of the wide port should be the multiple of narrow port's Data Width. The ratio can be 1, 2, or 4.
- Full memory space for both ports should be the same:

  (2^AW_A)*DW_A = (2^AW_B)*DW_B,

  where AW_X is Address Width of port X and DW_X is Data Width of port X. If Data Widths are equal, Address Widths should be the same. If Data Widths ratio is two, Address Widths difference should be one. Finally, if Data Widths ratio is four, Address Widths difference should be two.
- To handle different Data Widths, the byte enable signals are used inside wrapper. This is not visible to you. However, when Data Widths are not the same, the ECC is disabled even if you do not check the Byte Enable.
- Number of Bytes (NB) used for each port can be calculated using following formulas:

For the Narrow Port: NB = ceil(Data Width/8).

For the Wide Port: NB = (Data Widths Ratio) * (NB of Narrow port).

For example, if:

DW_A = 2 and DW_B = 8,

then:

NB_A = 1 and NB_B = 4

If Byte Enable is set for a port, then its width (in bits) is equal to the Number of Bytes for that port.

Byte Enable can be set only if the Number of Bytes is greater than one on the corresponding port and the number of bits is a multiple of 8 on both ports.

2.1.4. Write Mode Attribute

Any port with both Read and Write access has Write Mode attribute. This attribute is available for Port A in the Single Port Mode and for both Port A and Port B in True Dual-Port Mode. In Pseudo Dual-Port and ROM Modes, no Write Mode attribute is available as there are no ports with both Read and Write access.

There are three possible values for Write Mode attribute: Normal, Write Through, and Read Before Write. All three modes are supported in Single Port Large RAM, while only the first two are supported in the True Dual-Port Large RAM.

- In Normal mode, the output data does not change nor is it updated during the write operation.
- In Write Through mode, the output data is updated with the input data during the write cycle.
- In Read Before Write mode, the output data port is updated with the existing data stored in the write address, during a write cycle. This mode is supported only in the Single Port LRAM.

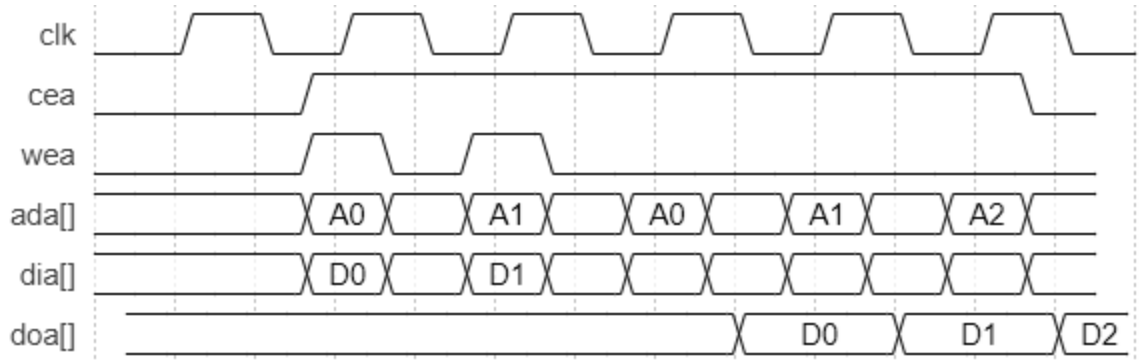


Figure 2.1. Single-Port LRAM Timing Diagram in Normal Mode (Output Register Disabled)

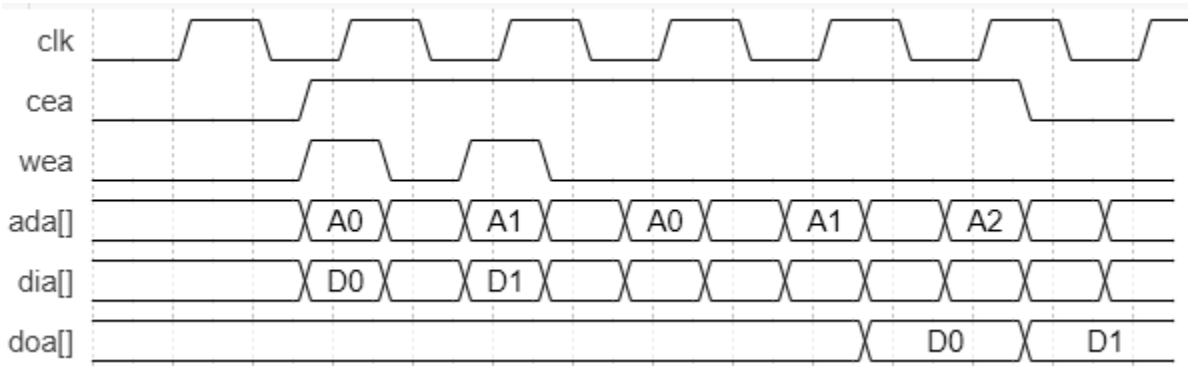


Figure 2.2. Single-Port LRAM Timing Diagram in Normal Mode (Output Register Enabled)

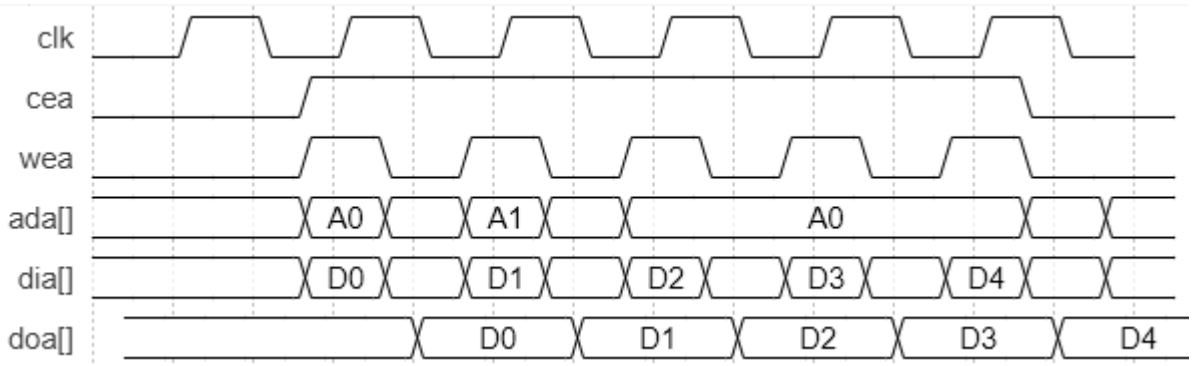


Figure 2.3. Single-Port LRAM Timing Diagram in Write Through Mode (Output Register Disabled)

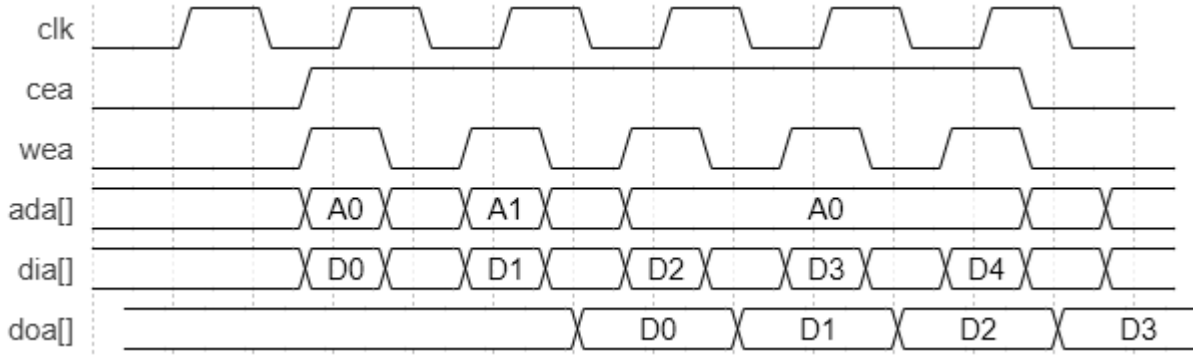


Figure 2.4. Single-Port LRAM Timing Diagram in Write Through Mode (Output Register Enabled)

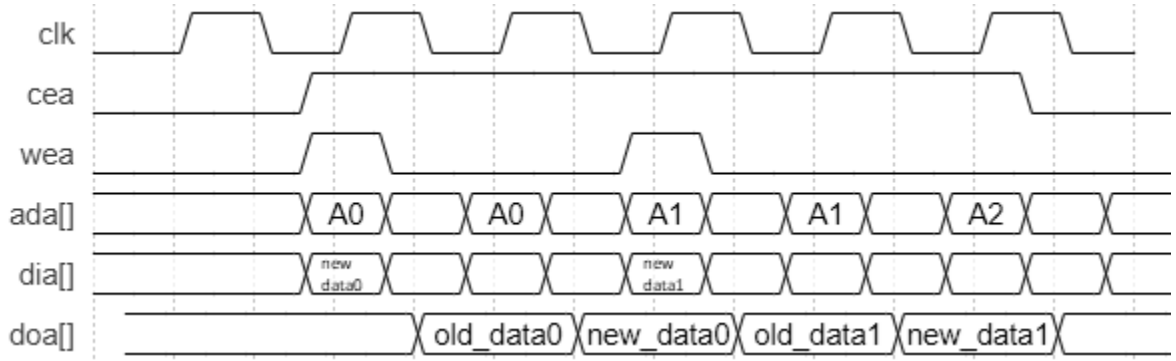


Figure 2.5. Single-Port LRAM Timing Diagram in Read Before Write Mode (Output Register Disabled)

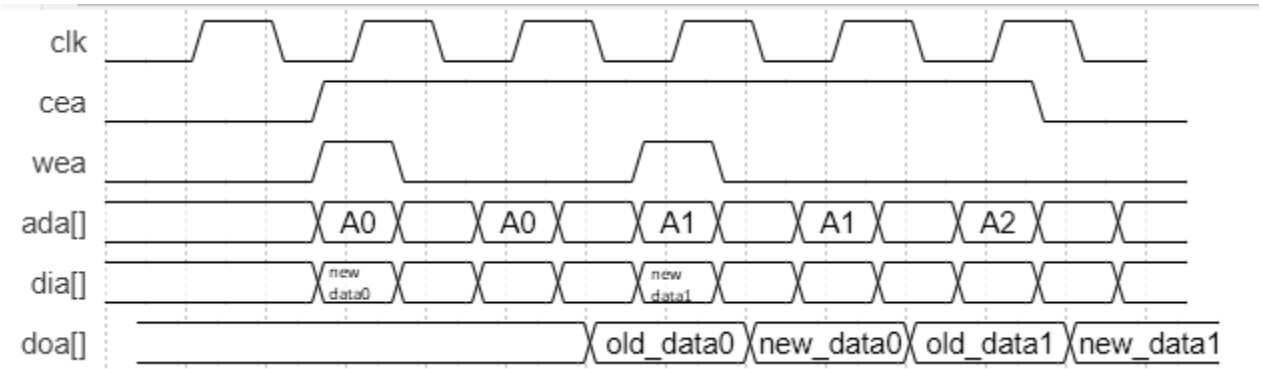
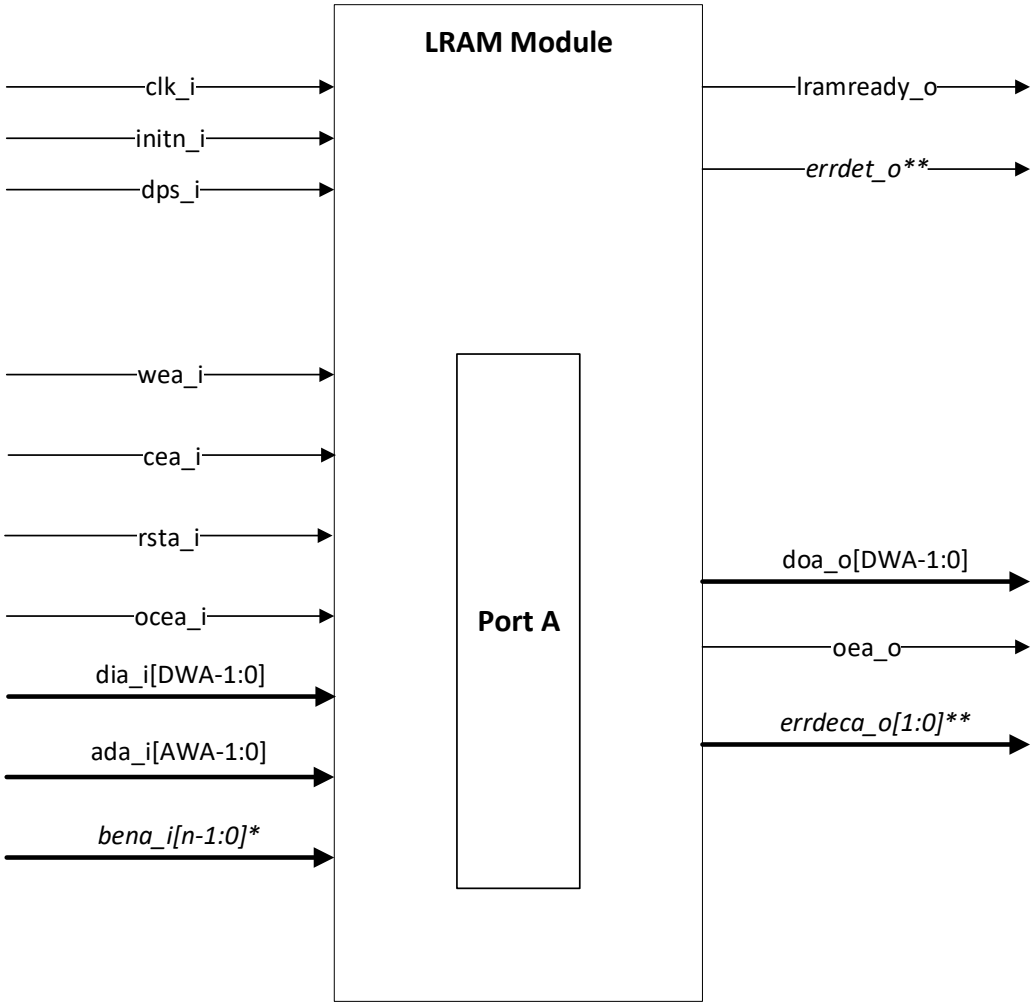


Figure 2.6. Single-Port LRAM Timing Diagram in Read Before Write Mode (Output Register Enabled)

2.2. Single-Port Large RAM

2.2.1. Functional Description

In the Single-Port Mode, only one port is used to write and read. Input can be configured as register in and output can be configured as register out. The SRAM enclosed in the Large RAM Module is synchronous. A functional block diagram for the Single Port Mode Large RAM is shown in Figure 2.7.



** optional, Byte Enable input is added in case "Provide Byte Enable" option is checked from GUI's General tab;*
*** optional, both signals are added in case "Enable ECC" option is checked from GUI's General tab*

Figure 2.7. Single-Port Mode Functional Diagram (Port A used as example)

DWA (Input Data Width for Port A) varies from 1–64;

AWA (Address Width for Port A) varies from 8–17;

n (the size of Byte Enable for Port A) takes values from 2–4.

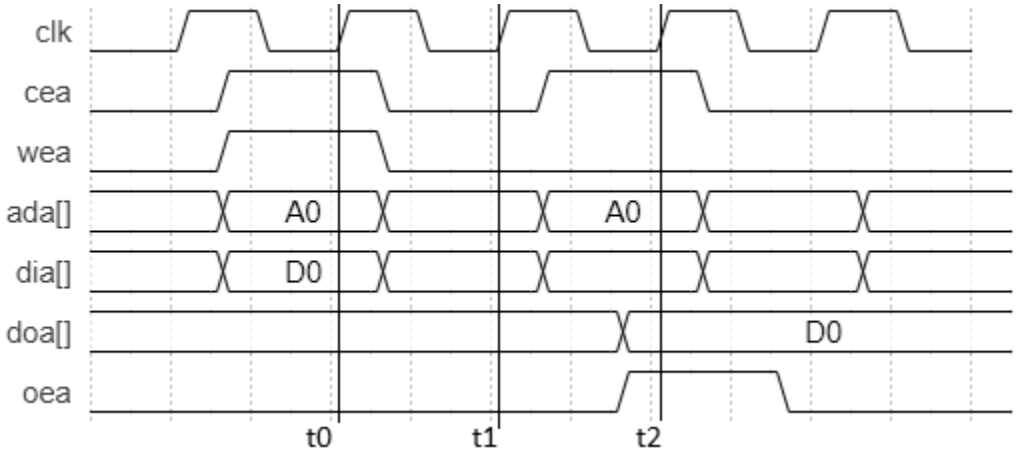


Figure 2.8. Single-Port Mode Timing Diagram (Output Register Disabled)

As shown in Figure 2.8, data flow is as follows:

- ada and dia are clocked in the SRAM at t0.
- When you read data, you must set cea and wea port values after t1.
- You get the read back data at t2.

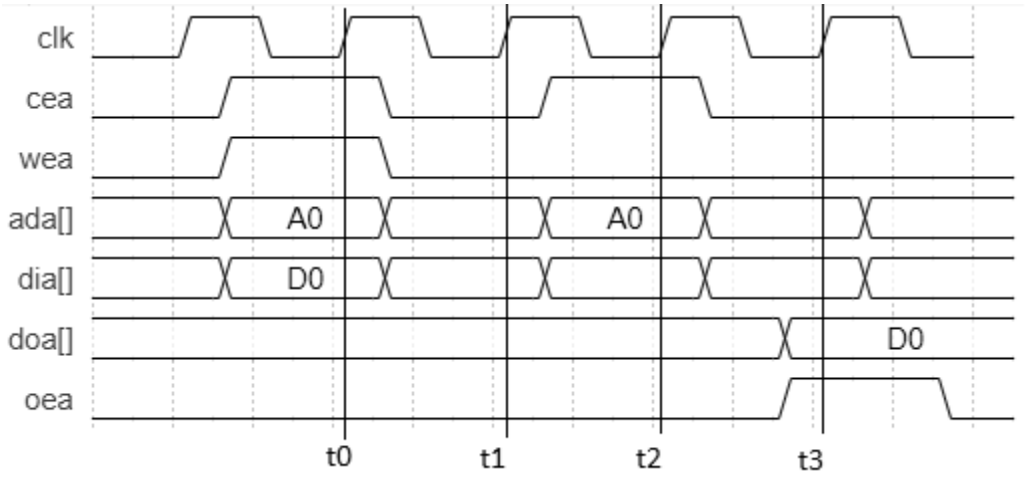


Figure 2.9. Single-Port Mode Timing Diagram (Output Register Enabled)

As shown in Figure 2.9, data flow is as follows:

- First, ada and dia are clocked in the SRAM at t0 in your view;
- When you read data, you must set cea and wea port values after t1;
- Large RAM connects those signals to SRAM; SRAM clocks in ada and dia, data D0 gets ready;
- Large RAM registers the output data with output register after t2 and connects it to output port doa;
- You get the read back data at t3.

For more information on various Write modes and corresponding timing diagrams, refer to Write Mode Attribute section.

2.2.2. Signal Description

Table 2.2. Single-Port Mode Signals

Name	Direction	Description
Clock and Reset		
clk_i	Input	Clock for Port A
Port A		
dia_i[DWA-1:0]	Input	Input Data Port A (1 – 64 bits)
ada_i[AWA-1:0]	Input	Port A Address (10 – 16 bits)
wea_i	Input	Port A Write Enable
cea_i	Input	Port A Clock Enable
bena_i[n-1:0]	Input	Port A Byte Enable (n takes values from 1 to 4). Optional signal for each bit position, *0* means the corresponding byte should be written, *1* – should not. Also an alternative pin for Unaligned Read (see Unaligned Read)
rsta_i	Input	Port A Logic Reset
ocea_i	Input	Port A Output Register Clock Enable
oea_o	Output	Output Enable Port A
doa_o[DWA-1:0]	Output	Output Data Port A
errdeca_o[1:0]	Output	Output ECC Error Indicator Port A [0]: One-bit error detected and corrected [1]: Two or more bits error detected
General		
dps_i	Input	Dynamic Power Select
lramready_o	Output	Large RAM Module ready indicator
errdet_o	Output	Large RAM Module error status

Notes:

- DWA (Input Data Width for Port A) varies from 1 – 64;
- AWA (Address Width for Port A) varies from 8 – 17;

2.2.3. Attribute Summary

Table 2.3 provides the list of user-configurable attributes for the Single Port Large RAM. Attributes are specified using Single Port Large RAM Core Configuration user interface in Lattice Radiant.

Table 2.3. Attribute Summary for Single-Port Mode

User Interface Configuration Tab		Attribute	Selectable Values	Default	Dependency on Other Attributes
General	General	LRAM Type	Single Port; True Dual Port; Pseudo Dual Port; ROM	Single Port	None
		Preserve Array Enable	Unchecked; Checked	Unchecked	
		Global Reset Enable	Unchecked; Checked	Checked	
		Provide Byte Enables	Unchecked; Checked	Unchecked	If Data Width is multiple of 8 and is greater than 8. Adds ByteEn input
		Unaligned Read Enable	Unchecked; Checked	Unchecked	Data Width = 32
		Enable ECC	Unchecked; Checked	Unchecked	Provide Byte Enables is unchecked
	Reset Mode	Assertion	Async; Sync	Sync	None

User Interface Configuration Tab		Attribute	Selectable Values	Default	Dependency on Other Attributes
		Release	Async; Sync	Sync	Reset Mode - Assertion set to Async
	Initialization	Memory Initialization	None; Initialize to all 0's; Initialize to all 1's; Memory File	None	None
		Memory File	Button; File browser	Unselected	
		Memory File Format	Binary; Hex; Addressed Hex	Binary	Memory File selected
A Port		Data Width	1 – 64	64	None

		Address Depth	256-137072	16384	If (*Actual Data Width = 64)
					Max (Address Depth) = 16384 (14 bit)
					else if (Actual Data Width = 32)
					Max (Address Depth) = 32768 (15 bit)
					else if (Actual Data Width = 16)
					Max (Address Depth) = 65536 (16 bit)
					else
					Max (Address Depth) = 131072 (17 bit)

		Address Width	8 - 17	14	Calculated, display only
		Write Mode	Normal; Write Through; Read Before Write	Normal	None
		Output Register	Unchecked; Checked	Unchecked	

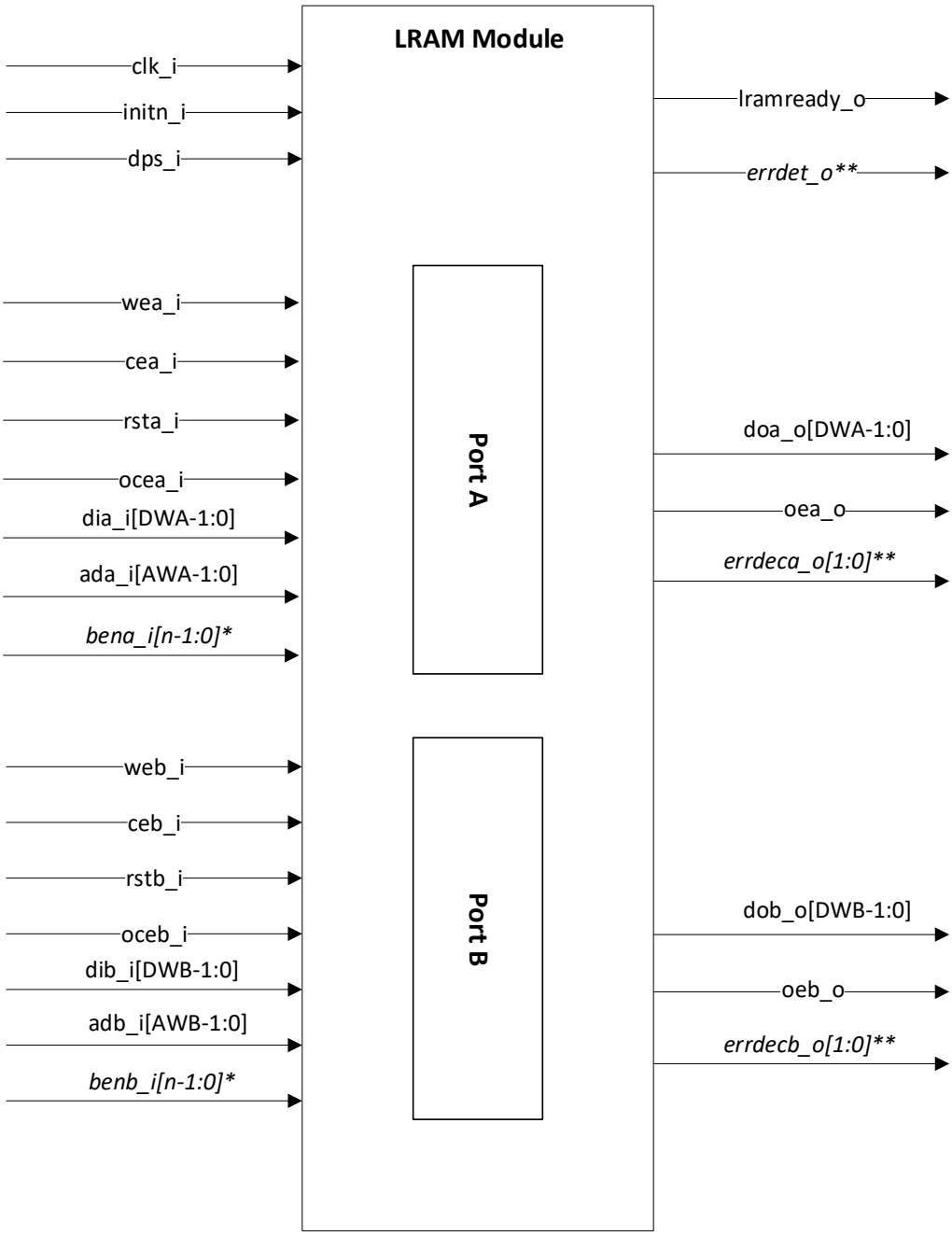
Note: For the Single-Port configuration, *Actual Data Width* is minimum power of two value that is greater or equal to 8 and is no less than *Data Width*. For example, if A_Width = 2, then A_Actual_Width = 8.

2.3. True Dual-Port Large RAM

2.3.1. Functional Description

Dual-Port Mode is implemented from Single-Port SRAM Model. To accommodate the requests from both ports at the same time, the enclosed Single-Port RAM runs the clock with doubled frequency.

In the Dual-Port Mode, if reading and writing operations come at one CIB clock cycle, those operations are mapped to the Single-Port SRAM model. A functional block diagram for the Dual-Port Mode Large RAM is shown in Figure 2.10.



** optional, Byte Enable input is added in case "Provide Byte Enable" is checked from GUI General tab;*
*** optional signals, added in case "Enable ECC" option is checked from GUI's General tab*

Figure 2.10. True Dual-Port Mode Functional Diagram

DWA (Input Data Width for Port A) varies from 1 – 64; AWA (the Address Width for Port A) varies from 8 - 17;

DWB (Input Data Width for Port B) varies from 1 – 64; AWB (the Address Width for Port B) varies from 8 - 17;

n (the size of Byte Enables for Ports A and B) takes values from 1 to 4.

In the Dual-Port Mode with both port A and port B reading data from SRAM, when submitted to SRAM in one cycle, Port B reads back the data with a one clock delay.

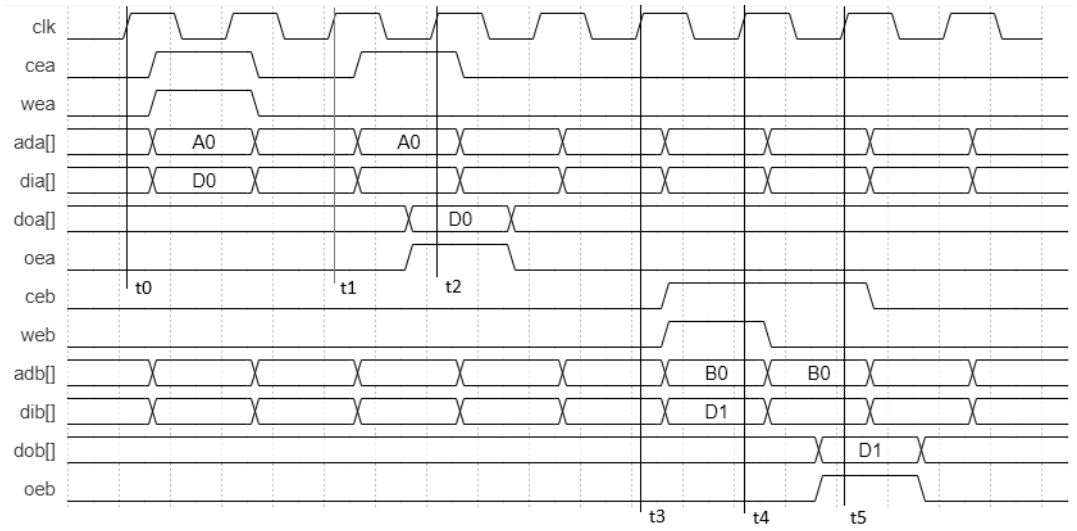


Figure 2.11. Dual-Port Mode Timing Diagram with Port A and Port B Working in Different Cycles (Output Registers Disabled for Both Ports)

As shown in Figure 2.11, data flow is as follows:

For Port A:

- You prepare data at t0;
- Large RAM clocks in cea, wea, ada, and dia to SRAM;
- When you read data, you must set cea and wea port values after t1. Large RAM connects those signals to SRAM;
- SRAM clocks in ada and dia, output data gets ready;
- Large RAM connects it to doa, and you get the read back data at t2.

For Port B:

- You prepare data at t3;
- Large RAM clocks in ceb, web, adb, and dib to SRAM;
- When you read data, you must set ceb and web port values. Large RAM connects those signals to SRAM at t4.
- SRAM clocks in adb and dib, output data gets ready.
- Large RAM connects it to dob, and you get the read back data at t5.

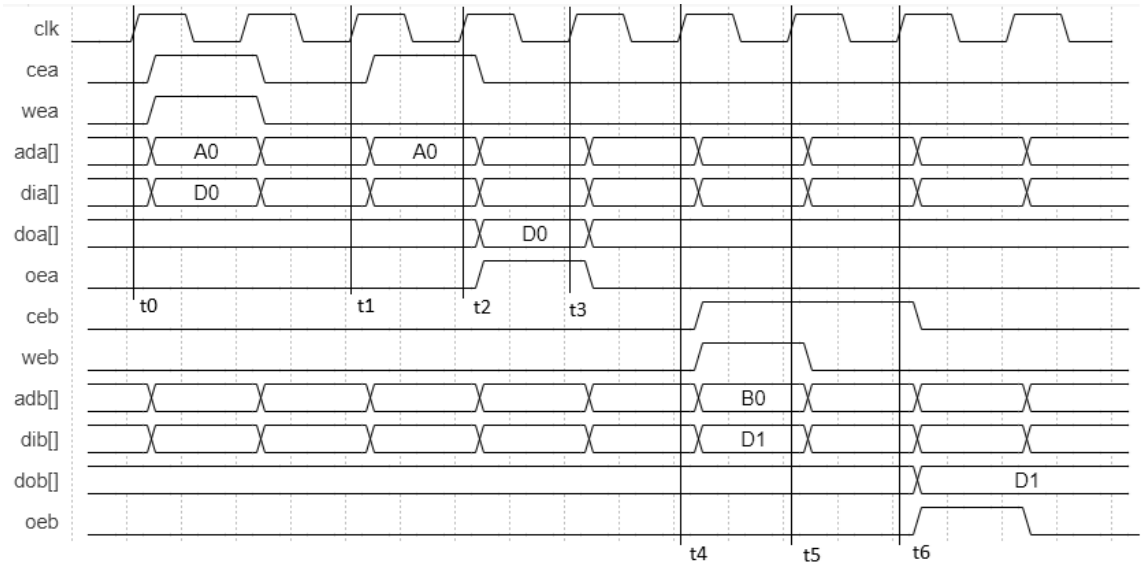


Figure 2.12. Dual-Port Mode Timing Diagram with Ports A and B Working in Different Cycles (Output Register Enabled for Both Ports)

As shown in Figure 2.12, data flow is as follows:

For Port A:

- You prepare data at t0;
- Large RAM clocks in cea, wea, ada and dia SRAM;
- When you read data, you must set cea and wea port values after t1. Large RAM connects those signals to SRAM.
- SRAM clocks in ada and dia, output data gets ready.
- Large RAM registers the output data with output register after t2 and connects it to output port doa.
- You get the read back data at t3.

For Port B:

- You prepare data at t4;
- Large RAM clocks in ceb, web, adb, and dib to SRAM at t4;
- When you read data, you must set ceb and web port values. Large RAM connects those signals to SRAM.
- SRAM clocks in adb and dib, output data gets ready after t5.
- Large RAM registers the output data with output register after t6 and connects it to output port dob.
- You get the read back data at the next positive edge of the clock.

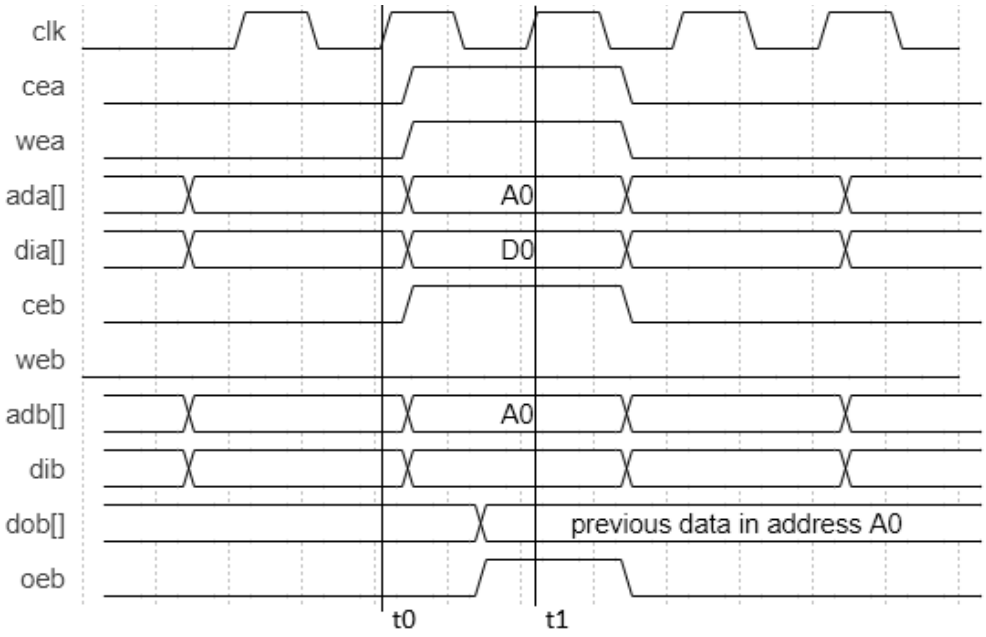


Figure 2.13. Dual-Port Mode Timing Diagram with Port A and Port B Working in the Same Cycle (Output Registers Disabled for Both Ports)

As shown in Figure 2.13, data flow as follows:

- Port A writes address A0 and Port B reads address A0 at the same clock cycle;
- At t0, Port B address is clocked into SRAM, output data is ready after t0;
- At t1, Port A's address and data are clocked into SRAM for writing;
- You get Port B read back data at t1.

Note: When both ports are writing and reading the same address, reading takes precedence over writing in one cycle, and the output of reading operation is previous data in address.

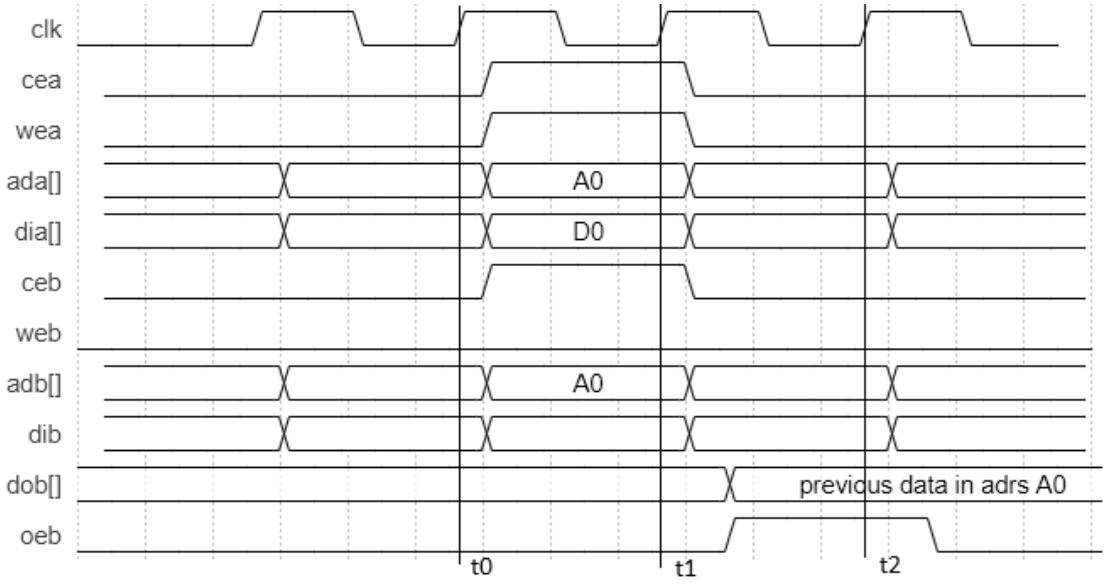


Figure 2.14. Dual-Port Mode Timing Diagram with Ports A and B Working in the Same Cycle (Output Register Enabled for Both Ports)

As shown in Figure 2.14, data flow is as follows:

- Port A writes address A0 and Port B reads address A0 at the same clock cycle;
- Port B address is clocked into SRAM, and output data is ready after t0;
- At t1, Port A's address and data are clocked into SRAM for writing;
- Large RAM registers the output data from Port B with the output register after t1 and connects it to output port dob.
- You get the Port B read back data at t2.

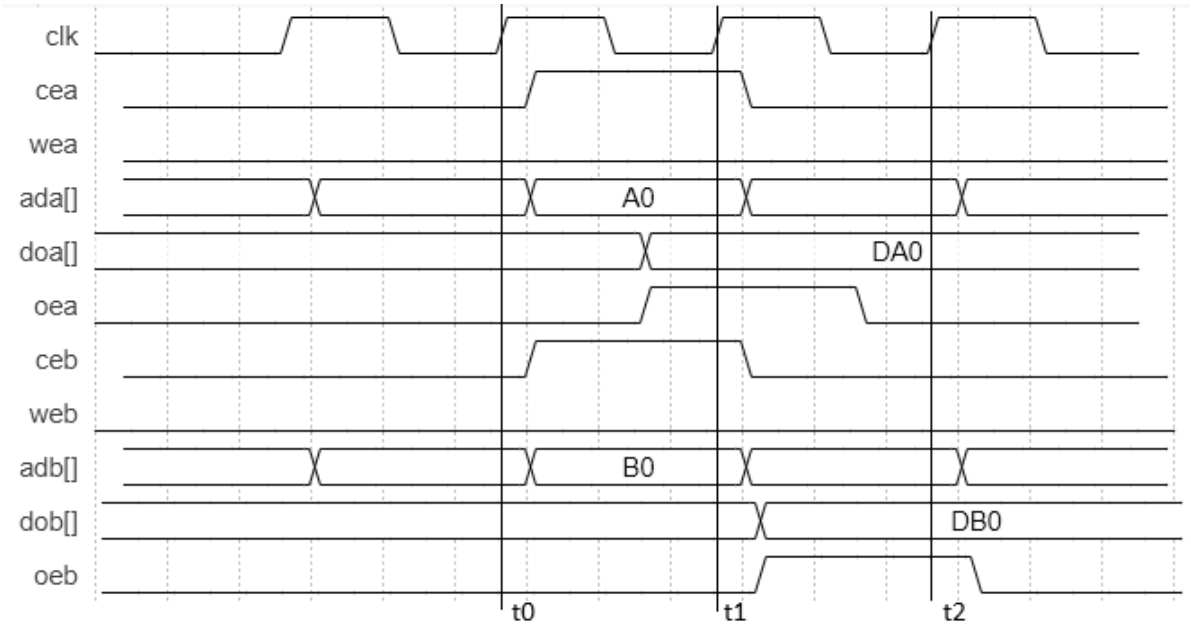


Figure 2.15. Dual-Port Mode Timing Diagram with Ports A and B Reading in the Same Cycle (Output Registers Disabled for Both Ports)

As shown in Figure 2.15, data flow is as follows:

- Both Port A and Port B read different address at the same clock cycle;
- Port A address is clocked into SRAM, and output data DA0 is ready after t0;
- You get the Port A read back data at t1.
- Port B address is clocked into SRAM, and output data DB0 is ready after t1;
- You get the Port B read back data at t2.

Note: When reading from both ports in the same cycle but from various addresses, data for port B comes with one clock delay. This is because actual LRAM primitive has just one port and it is impossible to read both addresses without delay.

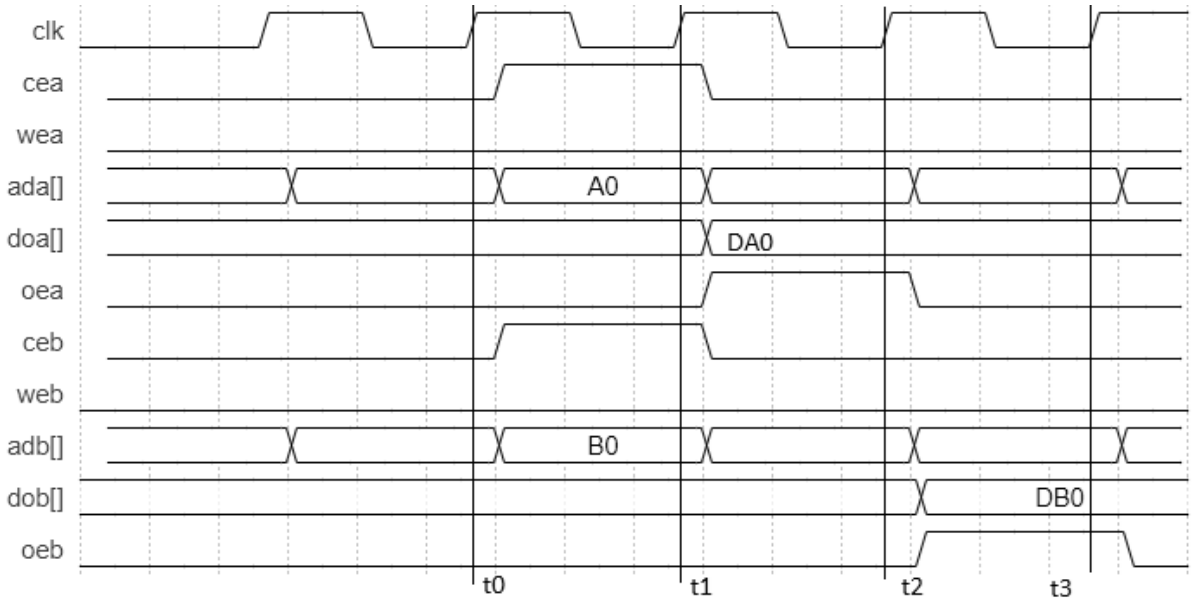


Figure 2.16. Dual-Port Mode Timing Diagram with Ports A and B Working in the Same Cycle (Output Register Enabled for Both Ports)

As shown in Figure 2.16, data flow as follows:

- Both Port A and Port B read different address at the same clock cycle;
- Port A address is clocked into SRAM, and output data DA0 is ready after t0;
- Large RAM registers output data DA0 with output register and connects it to output port doa.
- You get Port A read back data after t1.
- Port B address is clocked into SRAM, and output data DB0 is ready after t2;
- Large RAM registers the output data DB0 with output register after t2 and connects it to output port dob.
- You get Port B read back data at t3.

2.3.2. Signal Description

Table 2.4. True Dual-Port Mode Signals

Name	Direction	Description
Clock and Reset		
clk_i	Input	Clock for Port A and Port B
Port A		
dia_i[DWA-1:0]	Input	Input Data Port A (1 – 64 bits)
ada_i[AWA-1:0]	Input	Port A Address (10 – 16 bits)
wea_i	Input	Port A Write Enable
cea_i	Input	Port A Clock Enable
bena_i[n-1:0]	Input	Port A Byte Enable (n takes values from 1 to 4). Optional signal. For each bit position, *0* means the corresponding byte should be written, *1* mean should not. Also an alternative pin for Unaligned Read (see Unaligned Read)
rsta_i	Input	Port A Logic Reset
ocea_i	Input	Port A Output Register Clock Enable
oea_o	Output	Output Enable Port A
doa_o[DWA-1:0]	Output	Output Data Port A
errdeca_o[1:0]	Output	Output ECC Error Indicator Port A [0]: One-bit error detected and corrected [1]: Two or more bits error detected
Port B		
dib_i[DWB-1:0]	Input	Input Data Port B (1 – 64 bits)
adb_i[AWB-1-1:0]	Input	Port B Address (10 – 16 bits)
web_i	Input	Port B Write Enable
ceb_i	Input	Port B Clock Enable
benb_i[n-1:0]	Input	Port B Byte Enable (n takes values from 1 to 4). Optional signal For each bit position, “0” means the corresponding byte should be written, “1” – should not. Also an alternative pin for Unaligned Read (see Unaligned Read)
rstb_i	Input	Port B Logic Reset
oceb_i	Input	Port B Output Register
oeb_o	Output	Output Enable Port B
dob_o[DWB-1:0]	Output	Output Data Port B
errdecb_o[1:0]	Output	Output ECC Error Indicator Port B [0]: One-bit error detected and corrected [1]: Two or more bits error detected
General		
dps_i	Input	Dynamic Power Select
lramready_o	Output	Large RAM Module ready indicator
errdet_o	Output	Large RAM Module error status

Notes:

- DWA (Input Data Width for Port A) varies from 1 – 64; AWA (the Address Width for Port A) varies from 8 – 17.
- DWB (Input Data Width for Port B) varies from 1 – 64; AWB (the Address Width for Port B) varies from 8 – 17.

2.3.3. Attribute Summary

Table 2.5 provides the list of user configurable attributes for the True Dual-Port Large RAM. Attributes are specified using True Dual-Port Large RAM Core Configuration user interface in Lattice Radiant.

Table 2.5. Attribute Summary for True Dual-Port Mode

User Interface Configuration Tab		Attribute	Selectable Values	Default	Dependency on Other Attributes
General	General	LRAM Type	Single Port; True Dual Port; Pseudo Dual Port; ROM	True Dual Port	None
		Preserve Array Enable	Unchecked; Checked	Unchecked	
		Global Reset Enable	Unchecked; Checked	Checked	
		Provide Byte Enables	Unchecked; Checked	Unchecked	If both Data Widths are multiple of 8 and at least one is greater than 8. Adds ByteEn input.
		Unaligned Read Enable	Unchecked; Checked	Unchecked	Data Width = 32
		Enable ECC	Unchecked; Checked	Unchecked	If both Data Widths are > 16 and Provide Byte Enables is unchecked.
	Reset Mode	Assertion	Async; Sync	Sync	None
		Release	Async; Sync	Sync	Reset Mode – Assertion set to Async
	Initialization	Memory Initialization	None; Initialize to all 0's; Initialize to all 1's; Memory File	None	None
		Memory File	Button; File browser	Unselected	
		Memory File Format	Binary; Hex; Addressed Hex	Binary	Memory File selected
A Port		Data Width	1 – 64	64	Address Depth * Data Width must be the same for both ports and Data Width division of 2 ports must be 1, 2 or 4.

		Address Depth	256-131072	16384	If (*Actual Data Width = 64)
					Max (Address Depth) = 16384 (14 bit)
					else if (Actual Data Width = 32)
					Max (Address Depth) = 32768 (15 bit)
					else if (Actual Data Width = 16)
					Max (Address Depth) = 65536 (16 bit)
					else
					Max (Address Depth) = 131072 (17 bit)

		Address Width	8 - 17	14	Calculated, display only
		Write Mode	Normal; Write Through	Normal	None
		Output Register	Unchecked; Checked	Unchecked	

User Interface Configuration Tab	Attribute	Selectable Values	Default	Dependency on Other Attributes
B Port	Data Width	1 – 64	64	Address Depth * Data Width must be the same for both ports and Data Width division of 2 ports must be 1, 2 or 4

	Address Depth	256-131072	16384	If (*Actual Data Width = 64)
				Max (Address Depth) = 16384 (14 bit)
				else if (Actual Data Width = 32)
				Max (Address Depth) = 32768 (15 bit)
				else if (Actual Data Width = 16)
				Max (Address Depth) = 65536 (16 bit)
				else
				Max (Address Depth) = 131072 (17 bit)

	Address Width	8 - 17	14	Calculated, display only
	Write Mode	Normal; Write Through	Normal	None
	Output Register	Unchecked; Checked	Unchecked	

Note: For the True Dual-Port configuration:

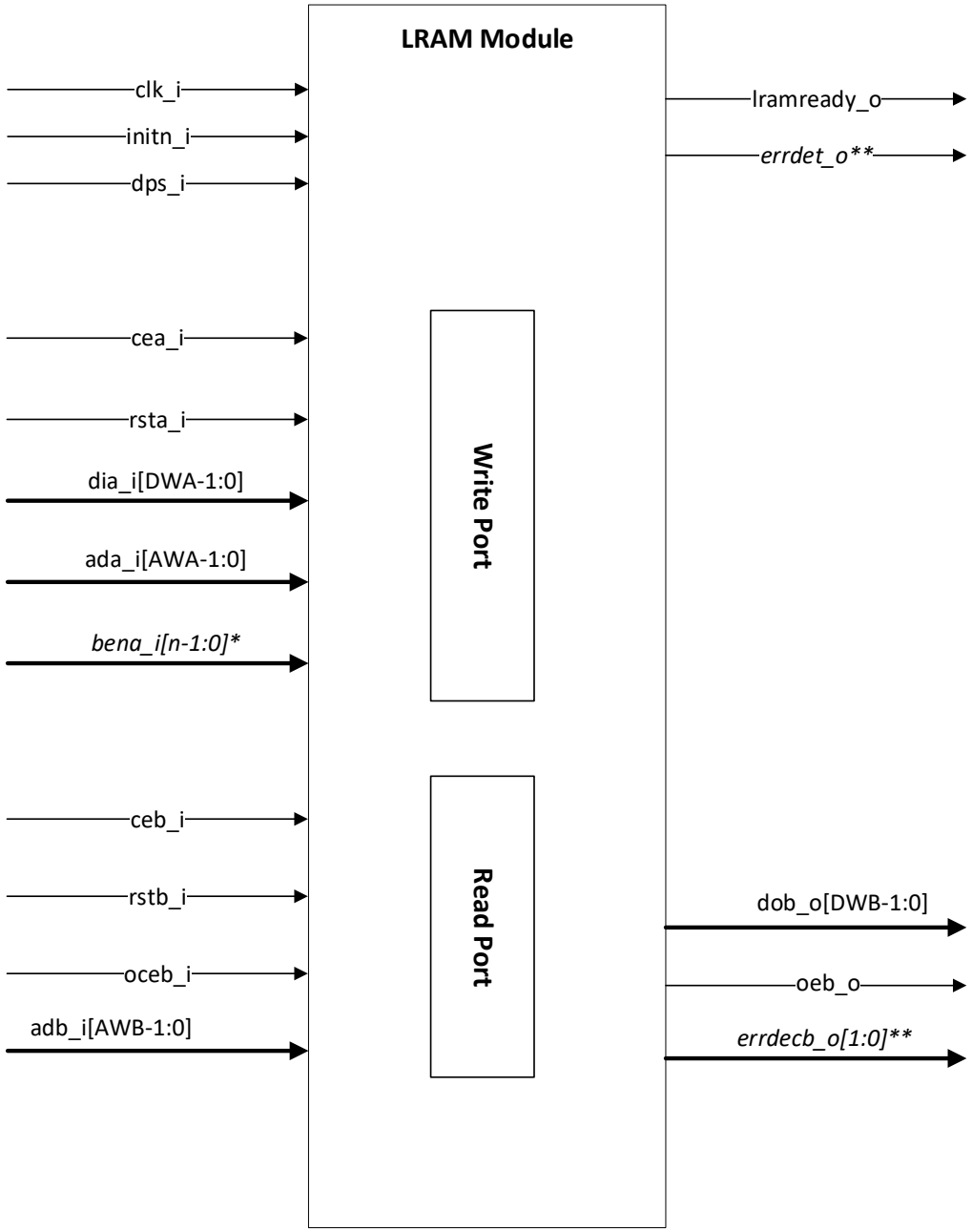
- For the narrow port, *Actual Data Width* is minimum power of two value that is greater or equal to 8 and is no less than *Data Width* value for that port;
- For the wide port, *Actual Data Width* is width_ratio times greater than the *Actual Data Width* value for the narrow port. For example, if:

  A_Width=2 and B_Width=8, then
  A_Actual_Width=8 and B_Actual_Width=8*4=32.

2.4. Pseudo Dual-Port Large RAM

2.4.1. Functional Description

In Pseudo Dual-Port Mode, Port A works as a writing port and Port B works as a reading port. Input and output register bypass mode is supported in the Single Clock Pseudo Dual-Port Mode. In this mode, both ports are writing to and reading from the same address. The reading takes precedence over writing in one cycle, so the output of reading is the previous data in the address. A functional block diagram for the Dual-Port Mode Large RAM is shown in Figure 2.17.



** optional, Byte Enable input is added in case "Provide Byte Enable" is checked from GUI General tab;*
*** optional, both signals are added in case "Enable ECC" option is checked from GUI's General tab*

Figure 2.17. Pseudo Dual-Port Mode Functional Diagram

For relevant timing diagrams, please refer to Figure 2.13 and Figure 2.14.

2.4.2. Signal Description

Table 2.6. Pseudo Dual-Port Mode Signals

Name	Direction	Description
Clock and Reset		
clk_i	Input	Clock for Write Port and Read Port
Write Port		
dia_i[DWW-1:0]	Input	Input Data Write Port (1 – 64 bits)
ada_i[AWW-1:0]	Input	Write Port Address (10 – 16 bits)
cea_i	Input	Write Port Clock Enable
bena_i[n-1:0]	Input	Write Port Byte Enable (n takes values from 1 to 4). Optional signal For each bit position, *0* means the corresponding byte should be written, *1* means should not. Also an alternative pin for Unaligned Read (see Unaligned Read)
rsta_i	Input	Write Port Logic Reset
errdeca_o[1:0]	Output	Output ECC Error Indicator Write Port [0]: One-bit error detected and corrected [1]: Two or more bits error detected
Read Port		
adb_i[AWR-1-1:0]	Input	Read Port Address (10 – 16 bits)
ceb_i	Input	Read Port Clock Enable
rstb_i	Input	Read Port Logic Reset
oceb_i	Input	Read Port Output Register
oeb_o	Output	Output Enable Read Port
dob_o[DWR-1:0]	Output	Output Data Read Port
errdecb_o[1:0]	Output	Output ECC Error Indicator Read Port [0]: One-bit error detected and corrected [1]: Two or more bits error detected
General		
dps_i	Input	Dynamic Power Select
lramready_o	Output	Large RAM Module ready indicator
errdet_o	Output	Large RAM Module error status

Notes:

- DWW (Input Data Width for Write Port) varies from 1 – 64; AWW (Address Width for Write Port) varies from 8 – 17.
- DWR (Input Data Width for Read Port) varies from 1 – 64; AWB (Address Width for Read Port) varies from 8 – 17.

2.4.3. Attribute Summary

Table 2.7 provides the list of user-configurable attributes for the Pseudo Dual-Port Large RAM. Attributes are specified using Pseudo Dual-Port Large RAM Core Configuration user interface in Lattice Radiant.

Table 2.7. Attribute Summary for Pseudo Dual-Port Mode

User Interface Configuration Tab		Attribute	Selectable Values	Default	Dependency on Other Attributes
General	General	LRAM Type	Single Port; True Dual Port; Pseudo Dual Port; ROM	Pseudo Dual Port	None
		Preserve Array Enable	Unchecked; Checked	Unchecked	
		Global Reset Enable	Unchecked; Checked	Checked	
		Provide Byte Enables	Unchecked; Checked	Unchecked	If both Data Widths are multiple of 8 and at least one is greater than 8. Adds ByteEn input.
		Unaligned Read Enable	Unchecked; Checked	Unchecked	Data Width = 32
		Enable ECC	Unchecked; Checked	Unchecked	If both Data Widths are > 16 and Provide Byte Enables is unchecked.
	Reset Mode	Assertion	Async; Sync	Sync	None
		Release	Async; Sync	Sync	Reset Mode - Assertion set to Async
	Initialization	Memory Initialization	None; Initialize to all 0's; Initialize to all 1's; Memory File	None	None
		Memory File	Button; File browser	Unselected	
		Memory File Format	Binary; Hex; Addressed Hex	Binary	Memory File selected
A Port		Data Width	1 – 64	64	Address Depth * Data Width must be the same for both ports and Data Width division of 2 ports must be 1, 2 or 4.

		Address Depth	256 - 131072	16384	If (*Actual Data Width = 64)
					Max (Address Depth) = 16384 (14 bit)
					else if (Actual Data Width = 32)
					Max (Address Depth) = 32768 (15 bit)
					else if (Actual Data Width = 16)
					Max (Address Depth) = 65536 (16 bit)
					else
					Max (Address Depth) = 131072 (17 bit)

		Address Width	8 - 17	14	Calculated, display only
		Write Mode	Normal	Normal	Greyed Out, display only
		Output Register	Unchecked	Unchecked	Check, Unchecked
B Port		Data Width	1 – 64	64	Address Depth * Data Width must be the same for both ports and Data Width division of 2 ports must be 1, 2 or 4.

User Interface Configuration Tab	Attribute	Selectable Values	Default	Dependency on Other Attributes

	Address Depth	256 - 131072	16384	If (*Actual Data Width = 64)
				Max (Address Depth) = 16384 (14 bit)
				else if (Actual Data Width = 32)
				Max (Address Depth) = 32768 (15 bit)
				else if (Actual Data Width = 16)
				Max (Address Depth) = 65536 (16 bit)
				else
				Max (Address Depth) = 131072 (17 bit)

	Address Width	8 - 17	14	Calculated, display only
	Write Mode	Normal	Normal	Greyed Out, display only
	Output Register	Unchecked; Checked	Unchecked	Check, Unchecked

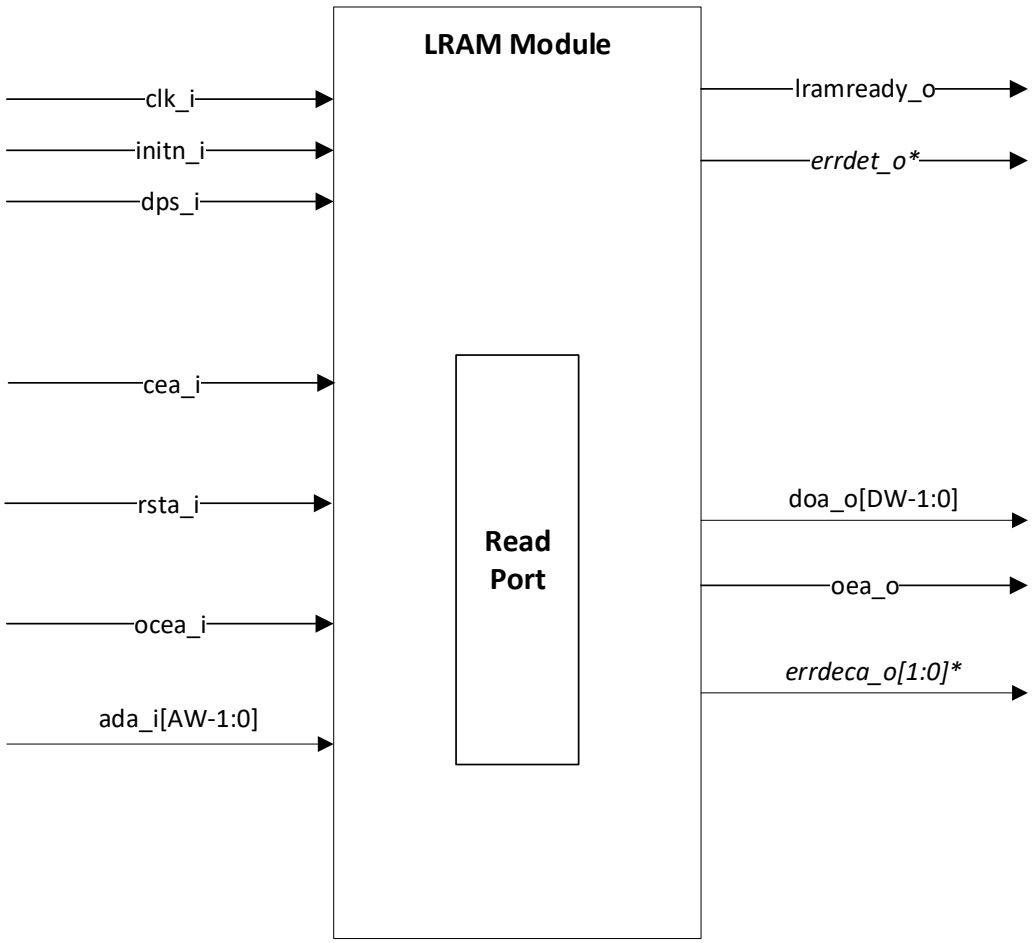
Note: For the Pseudo-Dual-Port configuration:

- For the narrow port, *Actual Data Width* is minimum power of two value that is greater or equal to 8 and is no less than *Data Width* value for that port;
- For the wide port, *Actual Data Width* is width_ratio times greater than the *Actual Data Width* value for the narrow port. For example, if: A_Width=2 and B_Width=8, then A_Actual_Width=8 and B_Actual_Width=8*4=32.

2.5. Large RAM Read Only Memory

2.5.1. Functional Description

When used as ROM, only one port is used to read. Input can be configured as register in and output can be configured as register out. The SRAM enclosed in the Large RAM Module is synchronous. A functional block diagram for the Dual-Port Mode Large RAM is shown in Figure 2.18.



** optional, both signals are added in case "Enable ECC" option is checked from GUI's General tab*

Figure 2.18. ROM Functional Diagram

The address is clocked into the SRAM when Clock Enable selection is enabled. In case the output registers are bypassed, the new data is available right after the rising edge of the same clock cycle on which read address is clocked into the SRAM with Clock Enable selection enable.

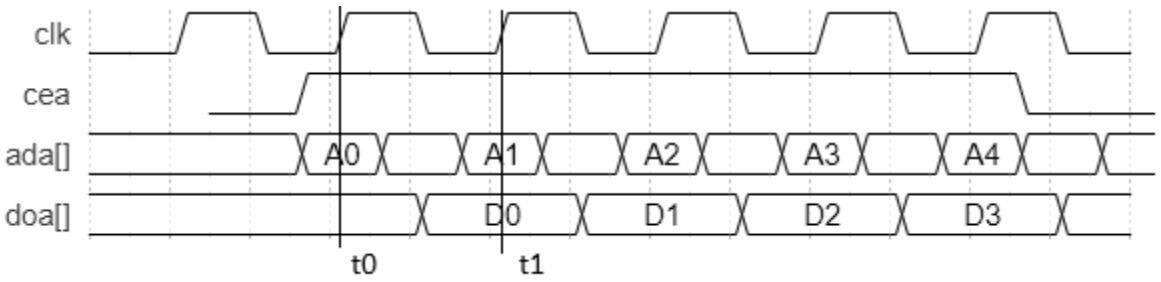


Figure 2.19. ROM Timing Diagram (Output Register Disabled)

As shown in Figure 2.19, data flow is as follows:

- ada is clocked in the SRAM at t0.
- You must set cea port value and gets the read back data at t1.

2.5.2. Signal Description

Table 2.8. ROM Signals

Name	Direction	Description
Clock and Reset		
clk_i	Input	Clock for Read Port
Read Port		
ada_i[AW-1:0]	Input	Read Port Address (10 – 16 bits)
cea_i	Input	Read Port Clock Enable
bena_i	Input	Byte Enable Port functioning as Unaligned Read (see Unaligned Read)
rsta_i	Input	Read Port Logic Reset
ocea_i	Input	Read Port Output Register Clock Enable
oea_o	Output	Output Enable Read Port
doa_o[DW-1:0]	Output	Output Data Read Port
errdeca_o[1:0]	Output	Output ECC Error Indicator Port A [0]: One-bit error detected and corrected [1]: Two or more bits error detected
General		
dps_i	Input	Dynamic Power Select
lramready_o	Output	Large RAM Module ready indicator
errdet_o	Output	Large RAM Module error status

2.5.3. Attribute Summary

Table 2.9 provides the list of user-configurable attributes for the Read Only Memory Large RAM. Attributes are specified using Read Only Memory Large RAM Core Configuration user interface in Lattice Radiant.

Table 2.9. Attribute Summary for ROM

User Interface Configuration Tab		Attribute	Selectable Values	Default	Dependency on Other Attributes
General	General	LRAM Type	Single Port; True Dual Port; Pseudo Dual Port; ROM	ROM	None
		Preserve Array Enable	Unchecked; Checked	Unchecked	
		Global Reset Enable	Unchecked; Checked	Checked	
		Unaligned Read Enable	Unchecked; Checked	Unchecked	Data Width = 32
		Enable ECC	Unchecked; Checked	Unchecked	
	Reset Mode	Assertion	Async; Sync	Sync	None
		Release	Async; Sync	Sync	Reset Mode – Assertion set to Async
	Initialization	Memory Initialization	None; Initialize to all 0's; Initialize to all 1's; Memory File	None	None
		Memory File	Button; File browser	Unselected	
		Memory File Format	Binary; Hex; Addressed Hex	Binary	Memory File selected
A Port		Data Width	1 - 64	64	None

		Address Depth	256-131072	16384	If (*Actual Data Width = 64)
					Max (Address Depth) = 16384 (14 bit)
					else if (Actual Data Width = 32)
					Max (Address Depth) = 32768 (15 bit)
					else if (Actual Data Width = 16)
					Max (Address Depth) = 65536 (16 bit)
					else
					Max (Address Depth) = 131072 (17 bit)

		Address Width	8 - 17	14	Calculated, display only
		Output Register	Unchecked; Checked	Unchecked	None

Note: For the ROM configuration, *Actual Data Width* is minimum power of two value that is greater or equal to 8 and is no less than *Data Width.* For example, if A_Width = 2, then A_Actual_Width = 8.

3. Memory Initialization File Format

There are three possible formats for memory initialization file:

- Binary format,
- Hex format, and
- Addressed Hex format.

On each line the most significant symbol (bit or hexadecimal symbol) is the first one (the furthermost left) and the last significant symbol is the last one (the furthermost right). Each next line contains data from higher addresses.

For example, if we have the following lines:

1234
5678
ABCD
EF90,

then the final initialization vector is *EF90ABCD56781234* where *34* is the data for address 0. Above mentioned is correct for all formats. Each format is described separately in the following sections.

3.1. Binary Format

For this format, only *0* and *1* symbols are allowed. The best practice is to have one word per line.

Although the number of bits on each line might be different (IP generation engine collects all lines and split them according to Data Width), it is recommended to have one word per line to have a user friendly format.

For example, if Data Width on port A is 14, then each line of initialization file contains 14 binary symbols (*0* or *1*).

3.2. Hex format

For this format, all digits from *0* to *9* as well as letters A, B, C, D, E, F, a, b, c, d, e and f are allowed.

Each line contains a string of the above-mentioned hexadecimal digits without any spaces. At the beginning of each string, the *0x* may or may not be attached.

If Data Width in bits for port A is multiple of 4, then Data_Width/4 hexadecimal symbols are dedicated for each word. If Data Width in bits for port A is not multiple of 4, then ceil(Data_Width/4) hexadecimal symbols are dedicated for each word.

For example, if number of bits in word is 6, then 2 hexadecimal symbols are dedicated per word. First two bits from 8 is cropped in this case.

3.3. Addressed Hex Format

This format is the same as Hex Format with a possibility to jump to the desired address.

If a line begins with an @ symbol, it means that it contains the jump address instead of data.

For example, the following record:

0x1234
0x3456
@0x100
0x7890,

means that after writing first two words (0x1234 and 0x3456) from the beginning of memory, it needs to jump to address 0x100, then write word 0x7890.

There is no limit on the number of jump commands in the initialization file. Furthermore, it can have the first line as a jump command.

4. IP Generation and Evaluation

This section provides information on how to generate and synthesize LRAM Module using Lattice Radiant Software. For more on Lattice Radiant Software, please refer to the Lattice Radiant Software 2.0 User Guide and relevant Lattice tutorials.

4.1. Generating and Synthesizing the Module

The Lattice Radiant Software allows you to customize and generate modules and IPs and integrate them into device architecture.

To generate the LRAM Module in Lattice Radiant Software:

1. In the **Module/IP Block Wizard**, create a new Lattice Radiant Software project for the LRAM Module.
2. In the dialog box of the **Module/IP Block Wizard** window, customize LRAM Module using drop-down menus and check boxes as desired. As a sample configuration, see Figure 4.1. For configuration options, see Table 2.3, Table 2.5, Table 2.7, and Table 2.9.

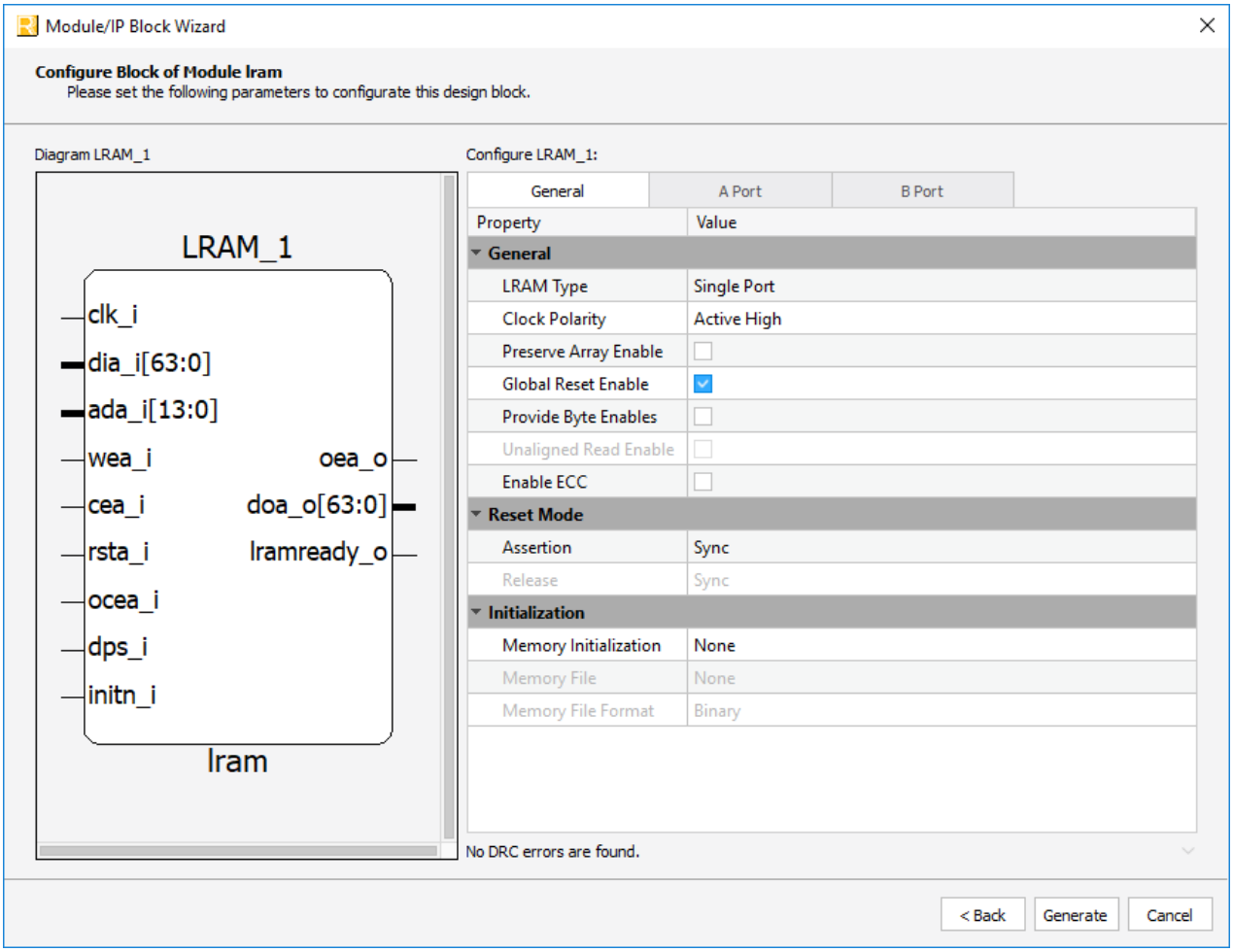


Figure 4.1. Configure Block of LRAM Module

3. Click **Generate**. The **Check Generating Result** dialog box opens, showing the design block messages and results as shown in Figure 4.2.

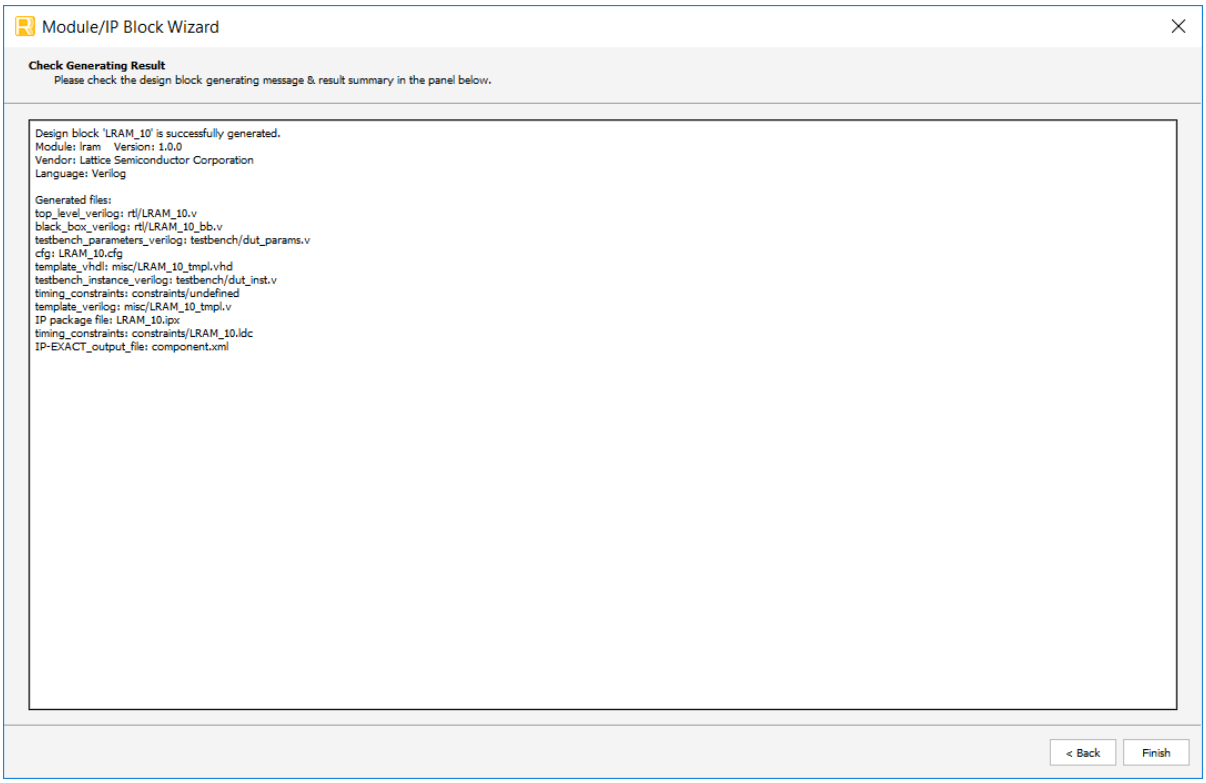


Figure 4.2. Check Generating Result

4. Click **Finish** to generate the Verilog file. Upon generating your desired design, you can synthesize it by pressing **Synthesize Design** on the top left corner of the screen, as shown in Figure 4.3.

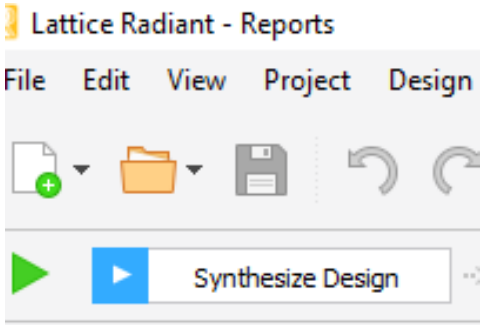


Figure 4.3. Synthesizing Design

4.2. Running Functional Simulation

To run the Verilog simulation:

1. Using the Lattice Radiant Software tcl console, go to the newly created *testbench* directory and run the command *source createDefines.tcl* to generate Verilog defines file.
2. Click the  button located on the Toolbar to initiate the Simulation Wizard shown in Figure 4.4

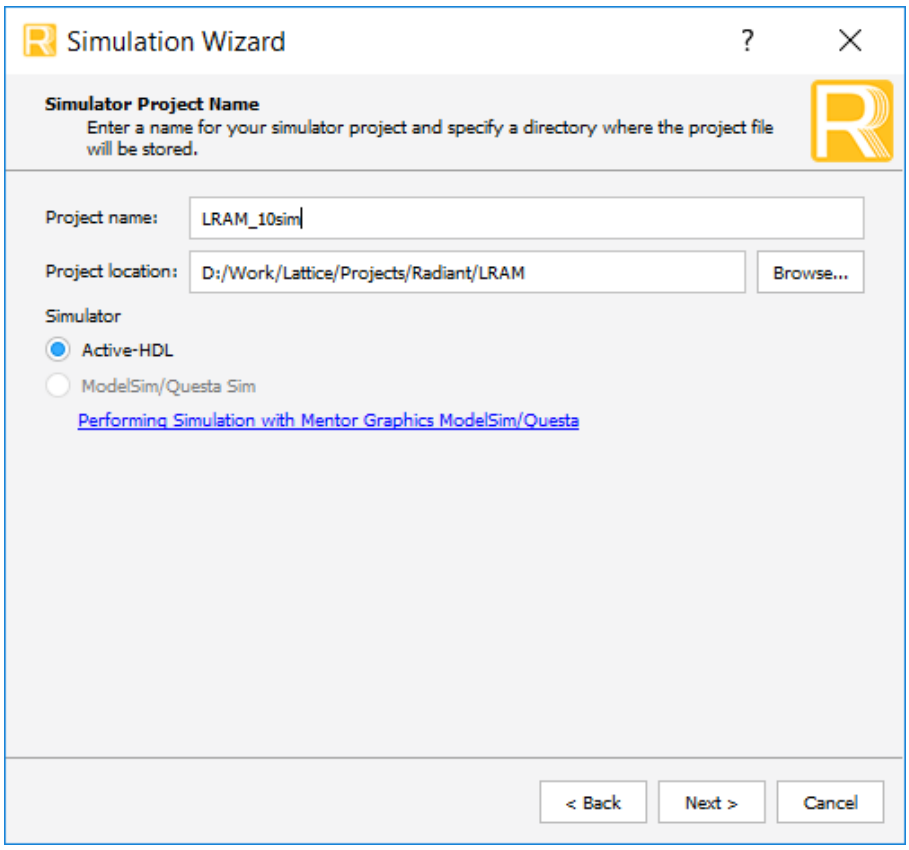


Figure 4.4. Simulation Wizard

3. Double-click **Next** to open the Add and Reorder Source window as shown in Figure 4.5.

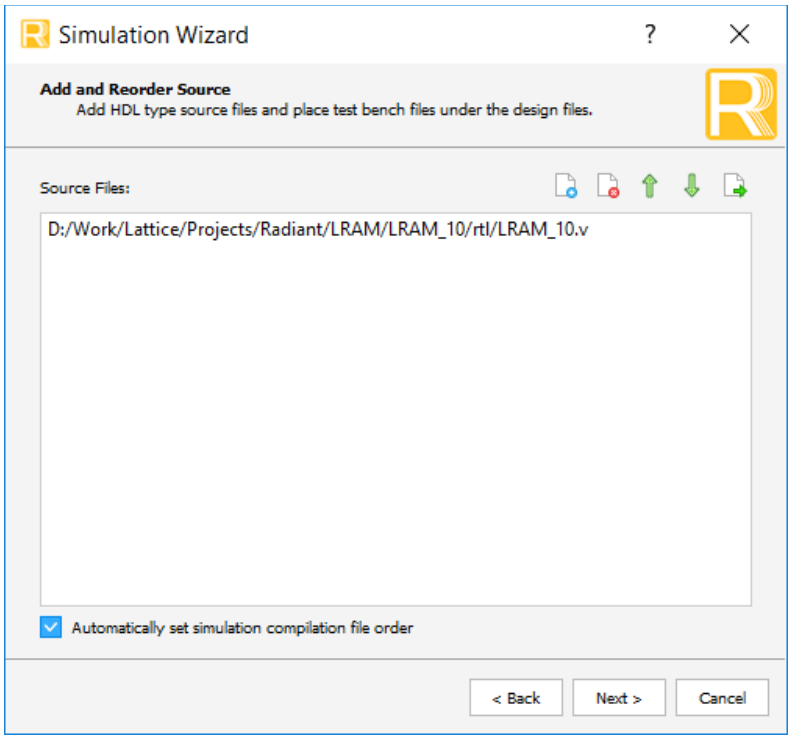


Figure 4.5. Adding and Reordering Source

4. Add *tb_top.v* file from *testbench* directory as shown in Figure 4.6.

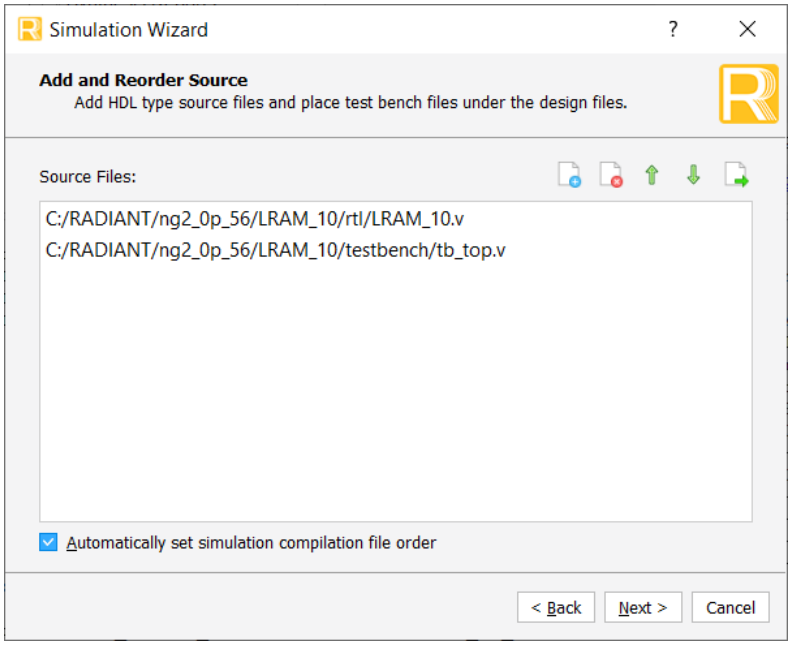


Figure 4.6. Adding File from Testbench Directory

5. Click **Next** to run simulation.

References

For complete information on Lattice Radiant Project-Based Environment, Design Flow, Implementation Flow and Tasks, as well as on the Simulation Flow, see the Lattice Radiant Software 2.0 User Guide.

Technical Support

For assistance, submit a technical support case at www.latticesemi.com/techsupport.

Revision History

Document Revision 1.0, Lattice Radiant SW version 2.0, December 2019

Section	Change Summary
All	Changed document status from Preliminary to final.
Acronyms in This Document	Added this section.
IP Generation and Evaluation	• Updated source file in Running Functional Simulation procedure to *tb_top.v*. • Updated Figure 4.6. Adding File from Testbench Directory.
All	Minor editorial changes

Section	Change Summary
All	Changed document status from Preliminary to final.

Document Revision 0.80, Lattice Radiant SW version 0.80, October 2019

Section	Change Summary
All	Preliminary release

www.latticesemi.com