FIFO Memory Modules

User Guide

FPGA-IPUG-02182-1.6

March 2026

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 REDUDANCIES, 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.

Inclusive Language

This document was created consistent with Lattice Semiconductor’s inclusive language policy. In some cases, the language in underlying tools and other items may not yet have been updated. Please refer to Lattice’s inclusive language FAQ 6878 for a cross reference of terms. Note in some cases such as register names and state names it has been necessary to continue to utilize older terminology for compatibility.

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

Contents

Contents....3
Abbreviations in This Document....6
1. Introduction....7
1.1. Quick Facts....7
1.2. Features....8
2. Memory Modules....9
2.1. First in First Out Single Clock (FIFO)....9
2.2. First in First Out Dual Clock (FIFO_DC)....12
2.2.1. Write-Once-Read-Many Application....21
3. IP Generation....23
3.1. Generation....23
3.2. Running Functional Simulation....26
3.3. Constraining the IP....28
4. PMI Support....29
4.1. pmi_fifo....29
4.2. pmi_fifo_dc....31
Appendix A. Resource Utilization....35
References....36
Technical Support Assistance....37
Revision History....38

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

Figures

Figure 2.1. FIFO Single Clock Memory Module Generated by Module/IP Block Wizard 9
Figure 2.2. FIFO Single Clock Memory Timing Diagram, Without Registers 11
Figure 2.3. FIFO Single Clock Memory Timing Diagram, With Output Registers 11
Figure 2.4. FIFO Single Clock Memory Timing Diagram, FWFT Enabled 12
Figure 2.5. FIFO Dual Clock Memory Generated by Module/IP Block Wizard 12
Figure 2.6. FIFO Dual Clock Memory Timing Diagram, Without Registers 16
Figure 2.7. FIFO Dual Clock Memory Timing Diagram, With Output Registers 16
Figure 2.8. FIFO Dual Clock Memory Timing Diagram, Mixed-Width Without Registers 16
Figure 2.9. FIFO Dual Clock Memory First-Word Fall-Through Mode Timing Diagram 17
Figure 2.10. FIFO Dual Clock Memory First-Word Fall-Through Mode Timing Diagram (Output Registers Enabled) 17
Figure 2.11. FIFO Dual Clock Memory Timing Diagram of Area-Optimized/HW-Based FIFO (Avant Devices) 17
Figure 2.12. FIFO Dual Clock Memory Timing Diagram, FWFT Mode (Area-Optimized/HW-Based FIFO, Avant Devices) 18
Figure 2.13. FIFO Dual Clock Memory Timing Diagram, FWFT Mode with Output Registers (Area-Optimized/HW-Based FIFO, Avant Devices) 18
Figure 2.14. FIFO Dual Clock Memory Timing Diagram of Area-Optimized/HW-Based FIFO (Nexus Devices) 18
Figure 2.15. FIFO Dual Clock Memory Timing Diagram, FWFT Mode (Area-Optimized/HW-Based FIFO, Nexus Devices) 18
Figure 2.16. FIFO Dual Clock Memory Timing Diagram, FWFT Mode with Output Registers (Area-Optimized/HW-Based FIFO, Nexus Devices) 19
Figure 2.17. rst_i Behavior in Area-Optimized (HW) Timing Diagram (Avant Devices) 20
Figure 2.18. rp_rst_i Behavior in Area-Optimized (HW) Timing Diagram (Avant Devices, with Output Registers) 20
Figure 2.19. rst_i Behavior in Area-Optimized (HW) Timing Diagram (Nexus Devices) 20
Figure 2.20. rp_rst_i Behavior in Area-Optimized (HW) Timing Diagram (Nexus Devices, with Output Registers) 21
Figure 2.21. rst_i Behavior in Feature-Rich (LUT) Controller Timing Diagram 21
Figure 2.22. rp_rst_i Behavior in Feature-Rich (LUT) Controller Timing Diagram 21
Figure 2.23. FIFO Dual Clock Memory Timing Diagram LUT-Based Controller, Read Pointer Reset Usage 22
Figure 2.24. FIFO Dual Clock Timing Diagram, Write-Once-Read-Many (Area-Optimized/HW-Based FIFO, Avant Devices) 22
Figure 2.25. FIFO Dual Clock Timing Diagram, Write-Once-Read-Many (Area Optimized/HW-Based FIFO, Nexus Devices) 22
Figure 3.1. Memory Modules Under Module/IP on Local in the Lattice Radiant Software 23
Figure 3.2. Example: Generating FIFO Using Module/IP Block Wizard 24
Figure 3.3. Example: Generating FIFO Module Customization 24
Figure 3.4. Example: Generating FIFO Check Generated Result 25
Figure 3.5. Simulation Wizard Simulator Project Name and Stage 26
Figure 3.6. Simulation Wizard Add and Reorder Source 26
Figure 3.7. Simulation Wizard Parse HDL files for simulation 27
Figure 3.8. Simulation Waveform 27

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

Tables

Table 1.1. FIFO Memory Modules Quick Facts 7
Table 1.2. FIFO Modules Key Features 8
Table 2.1. First in First Out Single Clock Memory Port Definitions 9
Table 2.2. First in First Out Single Clock Memory Attribute Definitions 10
Table 2.3. First in First Out Dual Clock Memory Port Definitions 12
Table 2.4. First in First Out Dual Clock Memory Attribute Definitions 13
Table 2.5. First in First Out Dual Clock Memory Attribute Definitions for MachXO4 Devices 14
Table 2.6. Feature-Rich (LUT) and Area-Optimized (HW) Reset Behavior 19
Table 3.1. Generated File List 25
Table 4.1. First In First Out Single Clock Memory PMI Port Definitions 29
Table 4.2. First In First Out Single Clock Memory PMI Attribute Definitions 29
Table 4.3. First in First Out Dual Clock Memory PMI Port Definitions 31
Table 4.4. First in First Out Dual Clock Memory PMI Attribute Definitions 32
Table A.1. FIFO Resource Utilization for LAV-AT-E70-2LFG1156C 35
Table A.2. FIFO_DC Resource Utilization for LAV-AT-E70-2LFG1156C 35
Table A.3. FIFO Resource Utilization for LFCPNX-100-8LFG672C 35
Table A.4. FIFO_DC Resource Utilization for LFCPNX-100-8LFG672C 35

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

Abbreviations in This Document

A list of abbreviations used in this document.

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

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

1. Introduction

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

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

- Using Module/IP Block Wizard – The Module/IP Block Wizard user interface allows you to specify the required memory type and size. The 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. You set the parameters and the control signals needed in Verilog.

1.1. Quick Facts

Table 1.1 presents a summary of the FIFO Memory Modules.

Table 1.1. FIFO Memory Modules Quick Facts

IP Requirements	Supported Devices	• IP Core v2.5.0 (FIFO) and IP Core v2.7.0 (FIFO_DC): iCE40 UltraPlus, Lattice Avant™, CertusPro™-NX, Certus™-NX, MachXO5™-NX, CrossLink™-NX, Certus™-NX-RT, CertusPro™-NX-RT • IP Core Version 3.0.1 (FIFO_DC): MachXO4™
	IP Changes[1]	Refer to the FIFO Memory Modules Release Notes (FPGA-RN-02095)
Resource Utilization	Supported User Interface	General purpose I/O (GPIO).
	Resources	Refer to Appendix A. Resource Utilization
Design Tool Support	Lattice Implementation[2]	IP Core v2.5.0 (FIFO) – Lattice Radiant Software 2024.2 IP Core v2.7.0 (FIFO_DC) – Lattice Radiant Software 2025.2 IP Core v3.0.1 (FIFO_DC) – Lattice Radiant Software 2025.2.1
	Synthesis	Lattice Synthesis Engine (LSE) Synopsys Synplify Pro® for Lattice
	Simulation	For a list of supported simulators, see the Lattice Radiant Software User Guide.

Notes:

1. In some instances, the IP may be updated without changes to the user guide. This user guide may reflect an earlier IP version but remains fully compatible with the later IP version. Refer to the IP Release Notes for the latest updates.
2. Lattice Implementation indicates the IP version release coinciding with the software version release. Check the software for IP version compatibility with earlier or later software versions.

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

1.2. Features

Table 1.2 shows key features of the FIFO modules include:

Table 1.2. FIFO Modules Key Features

FIFO (Single Clock)[1]	FIFO_DC (Dual Clocks) [1]	FIFO_DC (Dual Clocks)[2]
• Operates with a single clock for both read and write • Configurable data width and memory depth • Supports *Area-Optimized (HW)* or *Feature-Rich (LUT)* controller implementation • Memory implementation options: EBR or LUT • Optional *First-Word Fall-Through (FWFT)* mode • Optional registered output: *Enable Output Register* • Synchronous or asynchronous reset for controller and registers • Programmable *Almost Full* and *Almost Empty* flags: (static or dynamic thresholds) • Optional Data Count output • High-speed implementation option for HW controller[3]	• Operates with separate read and write clocks (asynchronous domains) • Configurable data width and memory depth • Supports mixed-width configurations with power-of-2 ratio (EBR only) • Supports *Area-Optimized (HW)* or *Feature-Rich (LUT)* controller implementation • Memory implementation options: EBR or LUT • Optional *First-Word Fall-Through (FWFT)* mode • Optional registered output: *Enable Output Register* • Synchronous or asynchronous reset for both read and write pointers • Additional reset for read pointer only • Programmable *Almost Full* and *Almost Empty* flags (static or dynamic thresholds) • Optional Data Count outputs for write and read sides • High-speed implementation option for HW controller[3]	• Operates with separate read and write clocks (asynchronous domains) • Configurable data width and memory depth • Optional registered output: *Enable Output Register* • Synchronous or asynchronous reset for both read and write pointers • Additional reset for read pointer only • Programmable *Almost Full* and *Almost Empty* flags (static or dynamic thresholds) • Optional Data Count outputs for write and read sides

Note:

1. Available only for non-MachXO4 devices.
2. Available only for MachXO4 devices.
3. High-speed implementation option for HW controller is supported only for Nexus devices.

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

2. Memory Modules

The following sections discuss the different FIFO 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 you to create memories larger than the width and depth supported for each memory primitive.

- **Output Register**
  The output data of the memory is optionally registered at the output. You 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 input and output registers of the memory module, and FIFO controller. It does not reset the contents of the memory module.
- **First-Word-Fall Through (FWFT) Mode**
  When First-Word Fall-Through (FWFT) is enabled, a pre-read operation is performed when the first word is written on an empty FIFO, even when you do not assert the read command. This pre-read causes the buffer to increase the size of the FIFO by 1 word and affects the behavior of ALMOST_FULL and ALMOST_EMPTY flags as the first word is popped out.

2.1. First in First Out Single Clock (FIFO)

Lattice FPGAs support all the features of First in First Out Single Clock Memory Module or FIFO. 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 2.1.

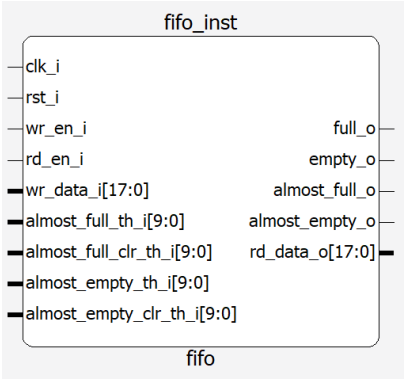

Figure 2.1. FIFO Single Clock Memory Module Generated by Module/IP Block Wizard

The various ports and their definitions for First in First Out Single Clock 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. First in First Out Single Clock Memory Port Definitions

Port Name	Direction	Width	Description
clk_i	Input	1	Clock
rst_i	Input	1	Reset
wr_en_i	Input	1	Write Enable
rd_en_i	Input	1	Read Enable
wr_data_i	Input	Data Width	Data Input
almost_full_th_i[1,2]	Input	Address Width	Almost full threshold value input
almost_full_clr_th_i[2]	Input	Address Width	Almost full clear / de-assert threshold value input
almost_empty_th_i[3,4]	Input	Address Width	Almost empty threshold value input
almost_empty_clr_th_i[4]	Input	Address Width	Almost empty clear / de-assert threshold value input
rd_data_o	Output	Data Width	Data Output
full_o	Output	1	Full Flag

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

Port Name	Direction	Width	Description
empty_o	Output	1	Empty Flag
almost_full_o	Output	1	Almost Full Flag
almost_ empty_o	Output	1	Almost Empty Flag
data_cnt_o	Output	Address Width + 1	Data Count – outputs the current number of written data that can be read in the FIFO

Notes:

1. Available when Almost Full Assertion = *Dynamic – Single Threshold*.
2. Available when Almost Full Assertion = *Dynamic – Dual Threshold*.
3. Available when Almost Empty Assertion = *Dynamic – Single Threshold*.
4. Available when Almost Empty Assertion = *Dynamic – Dual Threshold*.
5. Available when Enable Data Count is enabled.

The various attributes available for the First in First Out (FIFO) Single Clock are listed in Table2.2. Some of these attributes are user selectable through the Module/IP Block Wizard interface.

Table 2.2. First in First Out Single Clock Memory Attribute Definitions

Attribute	Description	Values	Default Value
Configuration Attributes			
Address Depth[1,3]	Address depth of the Read and Write port	2 – 65536[1,2,3]	1024
Data Width	Data word width of the Read and Write port	1 – 256	18
Controller Implementation[1]	Chooses how the FIFO controller is implemented. • Area-Optimized (HW) – uses the device's Hard-IP/EBR implementation of the FIFO logic and memory. • Feature-Rich (LUT) – uses the device's fabric resources to implement the FIFO logic. The memory implementation type can be chosen between EBR or LUT.	Area-Optimized (HW), Feature-Rich (LUT)	Area-Optimized (HW)
Use HIGH Speed Implementation[4]	Available only when Controller Implementation = Area-Optimized (HW). This option can be checked for an aggressive FIFO routing resulting significantly faster Fmax but can consume a large amount of EBR resources.	True, False	False
FWFT Enable	Enables First-Word-Fall-Through	True, False	False
Implementation Type[2]	Chooses how the FIFO memory is implemented.	EBR, LUT	EBR
Enable Output Register	Data Out port (Q) can be registered or not using this selection	True, False	True
Reset Assertion	Selection for Reset to be Synchronous or Asynchronous to the Clock	async, sync	async
Enable Almost Full Flag	Enables or disables the functionality of the almost full flag	enable, disable	enable
Almost Full Assertion	Checks how the almost full flag is implemented. Either dynamic or static, with or without clearing. (requires Enable Almost full flag	Static – Single Threshold, Static – Dual Threshold, Dynamic – Single Threshold,	Static – Single Threshold

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

Attribute	Description	Values	Default Value
	enabled).	Dynamic – Dual Threshold.	
Full Assert Level	The current address when written, flags the almost full flag. (Requires Static – Single / Dual Threshold).	0 < Full Assert Level < Address Depth	1023
Full Deassert Level	The current address when read, clears the almost full flag. (Requires Static – Dual Threshold).	0 < Full Deassert Level < Full Assert Level	1022
Enable Almost Empty Flag	Enables or disables the functionality of the almost empty flag	enable, disable	enable
Almost Empty Assertion	Checks how the almost empty flag is implemented. Either dynamic or static, with or without clearing. (requires Enable Almost empty flag enabled).	Static – Single Threshold, Static – Dual Threshold, Dynamic – Single Threshold, Dynamic – Dual Threshold.	Static – Single Threshold
Empty Assert Level	The current address when read, flags the almost empty flag. (Requires Static – Single / Dual Threshold).	0 < Empty Assert Level < Address Depth	1
Empty Deassert Level	The current address, when written, clears the almost empty flag. (Requires Static – Dual Threshold).	Empty Assert Level < Empty Deassert Level < Address Depth	2
Enable Data Count	Enables counting the number of data written to the FIFO	enable, disable	disable

Notes:

1. For Feature-Rich (LUT) based implementations, the ADDRESS_DEPTH must be a power of 2 (i.e., 2, 4, 8, … 65536).
2. For Implementation Type = LUT, maximum ADDRESS_DEPTH is 8192 only.
3. For Use HIGH Speed Implementation = True, maximum ADDRESS_DEPTH is 16383 only.
4. Available only for Nexus devices.

Timing diagrams for First in First Out Single Clock Memory are shown in Figure 2.2, Figure 2.3, and Figure 2.4 configurations without registers, with output registers, and when FWFT is enabled, respectively (FIFO Configuration uses ADDRESS_DEPTH = 4).

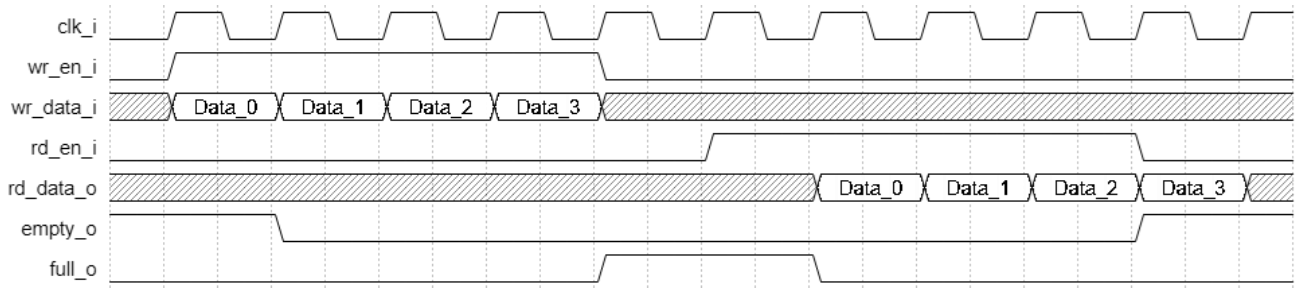

Figure 2.2. FIFO Single Clock Memory Timing Diagram, Without Registers

Figure 2.3. FIFO Single Clock Memory Timing Diagram, With Output Registers

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

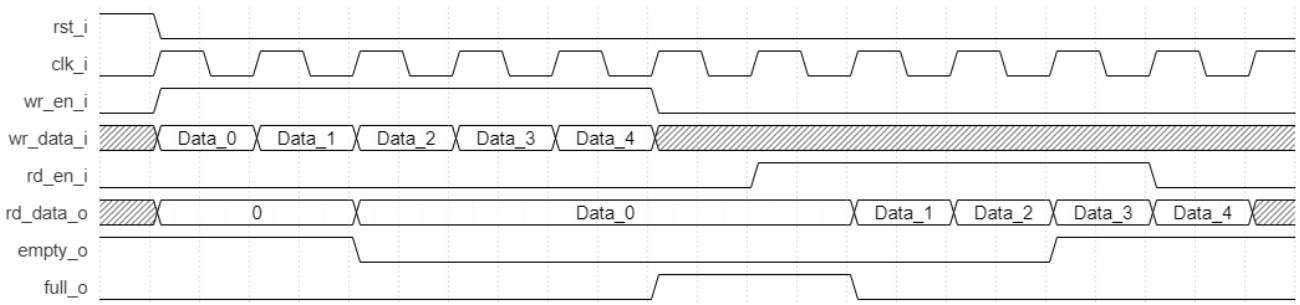

Figure 2.4. FIFO Single Clock Memory Timing Diagram, FWFT Enabled

2.2. First in First Out Dual Clock (FIFO_DC)

Lattice FPGAs support all the features of First in First Out Dual Clock Memory Module or FIFO_DC. 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 2.5.

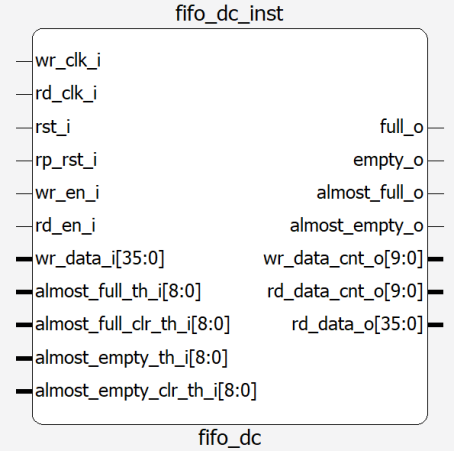

Figure 2.5. FIFO Dual Clock Memory Generated by Module/IP Block Wizard

The various ports and their definitions for First in First Out Dual Clock 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. First in First Out Dual Clock Memory Port Definitions

Port Name	Direction	Width	Description
wr_clk_i	Input	1	Write Clock
rd_clk_i	Input	1	Read Clock
rst_i	Input	1	FIFO Empty Reset – Write and Read pointer reset
rp_rst_i[1]	Input	1	FIFO Full Reset – Read pointer reset
wr_en_i	Input	1	Write Enable
rd_en_i	Input	1	Read Enable
wr_data_i	Input	Write Data Width	Data Input
almost_full_th_i[2,3]	Input	Write Address Width	Almost full threshold value input
almost_full_clr_th_i[3]	Input	Write Address Width	Almost full clear threshold/de-assert threshold input
almost_empty_th_i[4,5]	Input	Read Address Width	Almost empty threshold value input
almost_empty_clr_th_i[5]	Input	Read Address Width	Almost empty clear threshold/de-assert threshold value input

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

Port Name	Direction	Width	Description
rd_data_o	Output	Read Data Width	Data Output
full_o	Output	1	Full Flag
empty_o	Output	1	Empty Flag
almost_full_o	Output	1	Almost Full Flag
almost_ empty_o	Output	1	Almost Empty Flag
wr_data_cnt_o[6]	Output	Write Address Width + 1	Write Data Count – outputs the current number of data written to the FIFO.
rd_data_cnt_o[7]	Output	Read Address Width + 1	Read Data Count – outputs the current number of data that can be read from the FIFO.

Note:

1. rp_rst_i is usually used for Write-Once-Read-Many application. Refer to the Write-Once-Read-Many Application section for more information.
2. Available when Almost Full Assertion = *Dynamic – Single Threshold*.
3. Available when Almost Full Assertion = *Dynamic – Dual Threshold*.
4. Available when Almost Empty Assertion = *Dynamic – Single Threshold*.
5. Available when Almost Empty Assertion = *Dynamic – Dual Threshold*.
6. Available when Enable Data Count (Write) is enabled.
7. Available when Enable Data Count (Read) is enabled.

The various attributes available for the First in First Out Dual Clock (FIFO_DC) are listed in Table 2.4 and Table 2.5. Some of these attributes are user selectable through the Module/IP Block Wizard interface.

Table 2.4. First in First Out Dual Clock Memory Attribute Definitions

Attribute	Description	Values	Default Value
Configuration Attributes			
Write Port Address Depth[1,2,3,4]	Write Port Address depth	2 – 65536	512
Write Port Data Width[1]	Write Port Data word width	1 – 256	18
Read Port Address Depth[1,2,3,4]	Read Port Address depth	2 – 65536	512
Read Port Data Width[1]	Read Port Data word width	1 – 256	18
Controller Implementation	Chooses how the FIFO controller is implemented: • Area-Optimized (HW) – uses the device's Hard-IP / EBR implementation of the FIFO logic and memory. • Feature-Rich (LUT) – uses the device's fabric resources to implement the FIFO logic. The memory implementation type can be chosen between EBR or LUT.	Area-Optimized (HW), Feature-Rich (LUT)	Area-Optimized (HW)
Use HIGH Speed Implementation[5]	Available only when Controller Implementation = Area-Optimized (HW). This option can be checked for an aggressive FIFO routing resulting significantly faster Fmax but can consume a large amount of EBR resources.	True, False	False
FWFT Enable	Enables First-Word-Fall-Through	True, False	False
Implementation	Chooses how the FIFO_DC memory	EBR, LUT	EBR

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

Attribute	Description	Values	Default Value
	is implemented.		
Enable Output Register	Data Out port (Q) can be registered or not using this selection	True, False	True
Reset Assertion	Selection for Reset to be Synchronous or Asynchronous to the Clock	async, sync	async
Enable Almost Full Flag	Enables or disables the functionality of the almost full flag	enable, disable	enable
Almost Full Assertion	Checks how the almost full flag is implemented. Either dynamic or static, with or without clearing. (requires Enable Almost full flag enabled).	Static – Single Threshold, Static – Dual Threshold, Dynamic – Single Threshold, Dynamic – Dual Threshold.	Static – Single Threshold
Full Assert Level	The current address when written, flags the almost full flag (requires Static – Single / Dual Threshold).	0 < Full Assert Level < Write Address Depth	511
Full Deassert Level	The current address when read, clears the almost full flag (requires Static – Dual Threshold).	0 < Full Deassert Level < Full Assert Level	510
Enable Almost Empty Flag	Enables or disables the functionality of the almost empty flag	enable, disable	enable
Almost Empty Assertion	Checks how the almost empty flag is implemented. Either dynamic or static, with or without clearing. (requires Enable Almost empty flag enabled).	Static – Single Threshold, Static – Dual Threshold, Dynamic – Single Threshold, Dynamic – Dual Threshold.	Static – Single Threshold
Empty Assert Level	The current address when read, flags the almost empty flag. (Requires Static – Single / Dual Threshold).	0 < Empty Assert Level < Read Address Depth	1
Empty Deassert Level	The current address when written, clears the almost empty flag. (Requires Static – Dual Threshold).	Empty Assert Level < Empty Deassert Level < Read Address Depth	2
Enable Data Count (Write)	Enables counting the amount of data written to the FIFO_DC	enable, disable	disable
Enable Data Count (Read)	Enables counting the amount of data read from the FIFO_DC	enable, disable	disable

Notes:

1. For mixed width configurations total bit size must be equal (that is, WDEPTH*WDATA = RDEPTH*RDATA). For mixed width configuration the ratio between maximum DATA and minimum 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 <= 64.
2. For Feature-Rich (LUT) based implementations, the ADDRESS_DEPTH must be a power of 2 (i.e., 2, 4, 8, … 65536).
3. Mixed width implementations cannot be used with LUT memory, and maximum ADDRESS_DEPTH is 8192 only.
4. For Use HIGH Speed Implementation = True, maximum ADDRESS_DEPTH is 16383 only.
5. Available only for Nexus devices.

Table 2.5. First in First Out Dual Clock Memory Attribute Definitions for MachXO4 Devices

Attribute	Description	Values	Default Value
FIFO Implementation			
Memory Type	Select the memory type	EBR, LUT	EBR
Write Depth	Specify the write depth	2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384	512
Write Data Width	Specify the write data width	1 – 256	18

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

Attribute	Description	Values	Default Value
Read Depth[1]	Specify the read depth	2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384	512
Read Data Width	Display the read data width, calculated based on the formula of (Write Depth x Write Data Width) / Read Depth	1 – 256	18
Enable Output Register	Specify to enable/disable data output register	Unchecked, Checked	Unchecked
Output Register Mode	Select the mode of the data output register, only available when *Enable Output Register* is checked	No Gating, Read Enable, Output Clock Enable, Output Clock Enable + Read Enable	No Gating
Flag Control			
Enable Almost Empty Flag	Specify to enable/disable almost empty flag	Unchecked, Checked	Checked
Almost Empty Flag Assertion	Select the almost empty flag assertion implementation type	Static - Single Threshold, Static - Dual Threshold, Dynamic - Single Threshold, Dynamic - Dual Threshold	Static - Single Threshold
Almost Empty Flag Assertion Level	The current address when read, flags the almost empty flag. (Requires Static - Single / Dual Threshold).	1 – Read Depth	1
Almost Empty Flag De-assertion Level	The current address when written, clears the almost empty flag. (Requires Static - Dual Threshold).	1 – Read Depth	2
Enable Almost Full Flag	Specify to enable/disable almost full flag	Unchecked, Checked	Checked
Almost Full Flag Assertion	Select the almost full flag assertion implementation type	Static - Single Threshold, Static - Dual Threshold, Dynamic - Single Threshold, Dynamic - Dual Threshold	Static - Single Threshold
Almost Full Flag Assertion Level	The current address when written, flags the almost full flag (requires Static - Single / Dual Threshold).	1 – Write Depth	511
Almost Full Flag De-assertion Level	The current address when read, clears the almost empty flag. (Requires Static - Dual Threshold).	1 – Write Depth	510
Reset Mode			
Reset Assertion Mode	Specify the rest assertion to be asynchronous or synchronous to the clock	Async, Sync	Async
Reset Release Mode	Specify the rest release to be asynchronous or synchronous to the clock	Async, Sync	Sync
Data Counts			
Enable Write Data Count	Specify to enable/disable the counting of data written into the FIFO	Unchecked, Checked	Unchecked
Enable Read Data Count	Specify to enable/disable the counting of data read from the FIFO	Unchecked, Checked	Unchecked
Error Correction Code (ECC)			

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

Attribute	Description	Values	Default Value
Enable ECC (not supported for Data Width > 64) [2]	Specify to enable/disable ECC	Unchecked, Checked	Unchecked

Notes:

1. The read depth is currently grayed-out and will only be visible when mixed width configurations are supported in future Radiant release.
2. The ECC is currently not available and will be supported in future Radiant release.

Timing diagrams for FIFO Dual Clock Memory are shown in Figure 2.6 through Figure 2.22 for different FIFO_DC configurations depending on the device family being used:

- Figure 2.6 to Figure 2.10 show the different FIFO_DC behaviors when the FIFO Controller being used is LUT-based (Feature-Rich), for both Nexus and Avant devices.
- Figure 2.11 to Figure 2.13 show timing diagrams when using Area-Optimized (HW-Based) Dual Clock FIFO with Avant devices.
- Figure 2.14 to Figure 2.16 show timing diagrams when using Area-Optimized (HW-Based) Dual Clock FIFO with Nexus devices.
- Figure 2.17 to Figure 2.22 show the expected reset usage and behavior of the FIFO flags and data output (the configuration shows synchronous reset behavior).

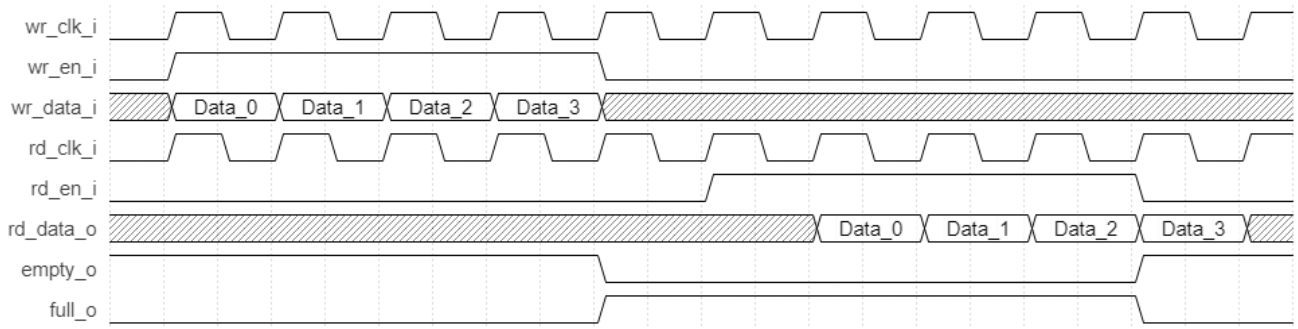

Figure 2.6. FIFO Dual Clock Memory Timing Diagram, Without Registers

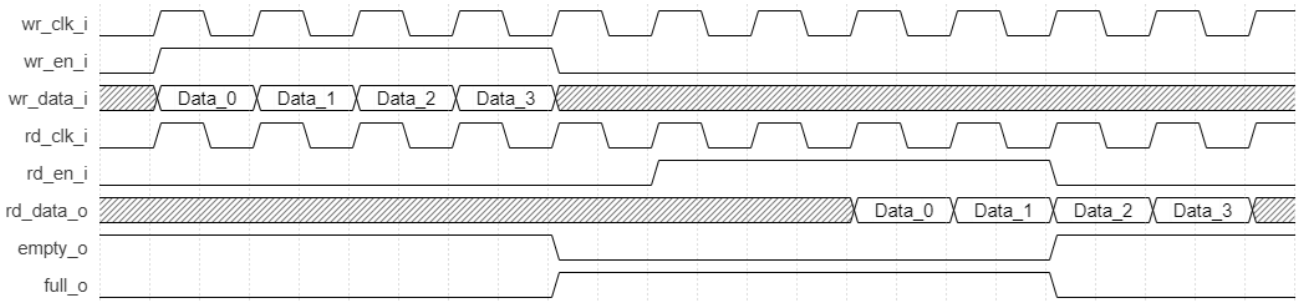

Figure 2.7. FIFO Dual Clock Memory Timing Diagram, With Output Registers

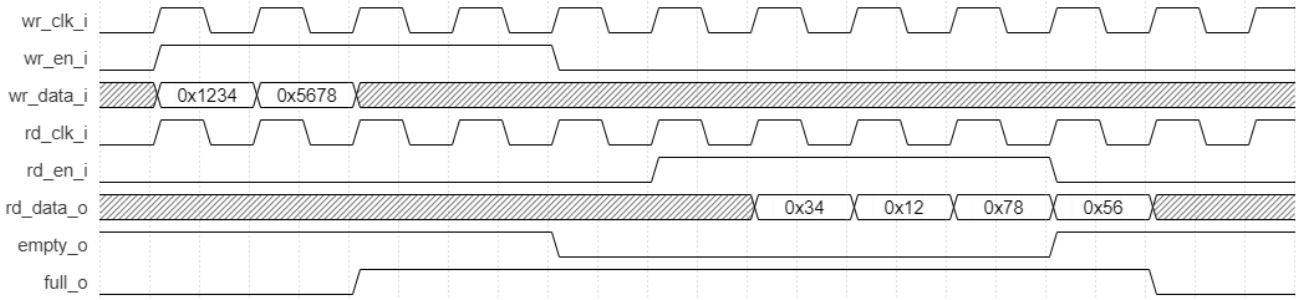

Figure 2.8. FIFO Dual Clock Memory Timing Diagram, Mixed-Width Without Registers

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

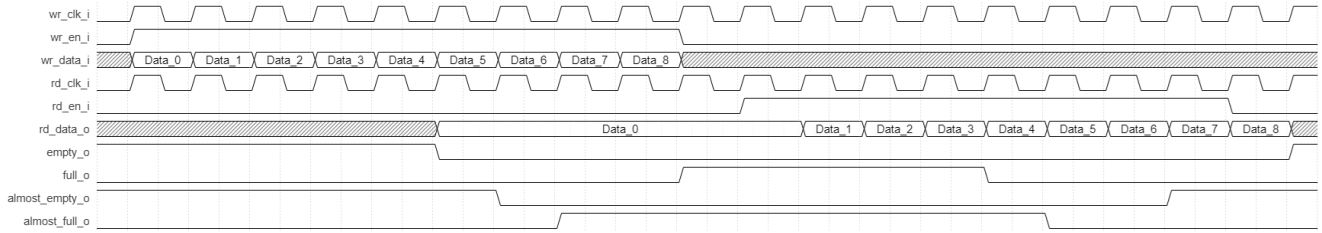

Figure 2.9. FIFO Dual Clock Memory First-Word Fall-Through Mode Timing Diagram

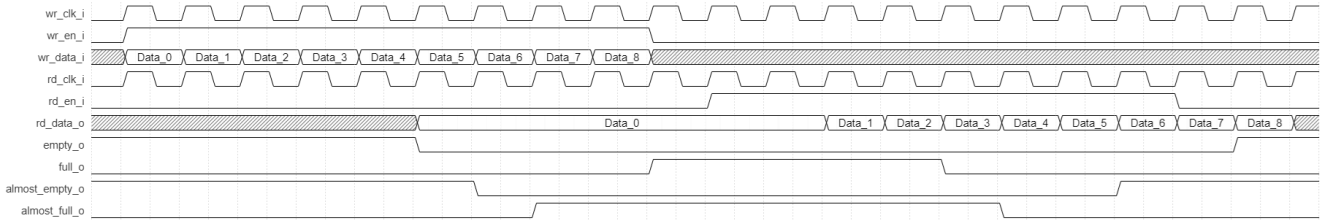

Figure 2.10. FIFO Dual Clock Memory First-Word Fall-Through Mode Timing Diagram (Output Registers Enabled)

Latency Differences between Feature-Rich (LUT) and Area-Optimized (HW) based implementations:

In the LUT based controller, the FIFO controller is implemented on the fabric. Synchronization logic is required to translate the write-clock domain transactions to the read-clock and vice-versa to properly maintain the integrity of the data being passed. This introduces a latency to the system and affects the flags from either side of the domain.

The FULL and ALMOST_FULL flags operating in the write-clock domain waits for three cycles before de-assertion because of a read in the read-clock domain. However, no latency is required for assertion as a write operation operates in the same write-clock domain. Additionally, this implementation is the same for the EMPTY and ALMOST_EMPTY flags which operate in the read-clock domain. There is a three-cycle delay before de-assertion, but no additional latency for the flag assertion as the read operates in the clock domain.

In HW based controllers, a special circuit is utilized to mitigate clock crossing issues and removes this latency altogether.

Similar configurations are used as shown in the following waveforms.

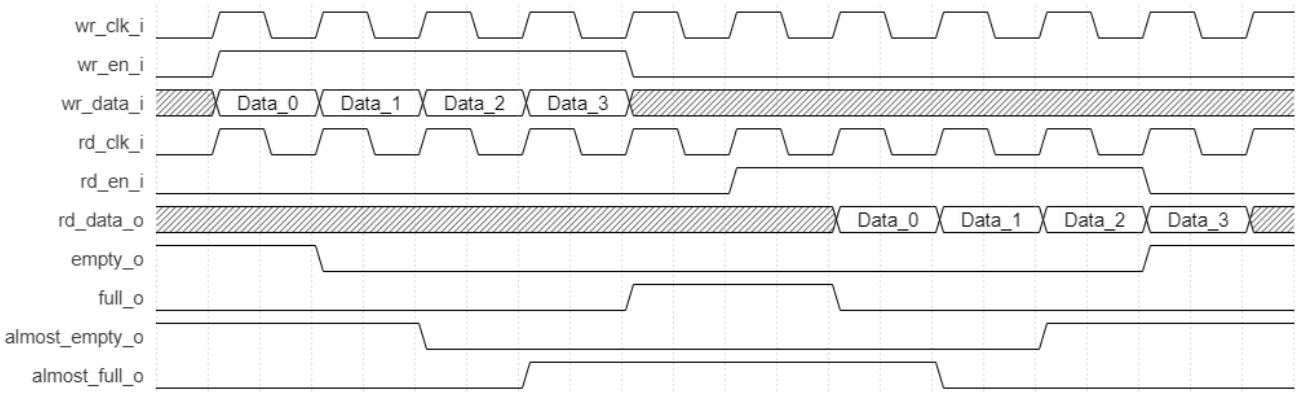

Figure 2.11. FIFO Dual Clock Memory Timing Diagram of Area-Optimized/HW-Based FIFO (Avant Devices)

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

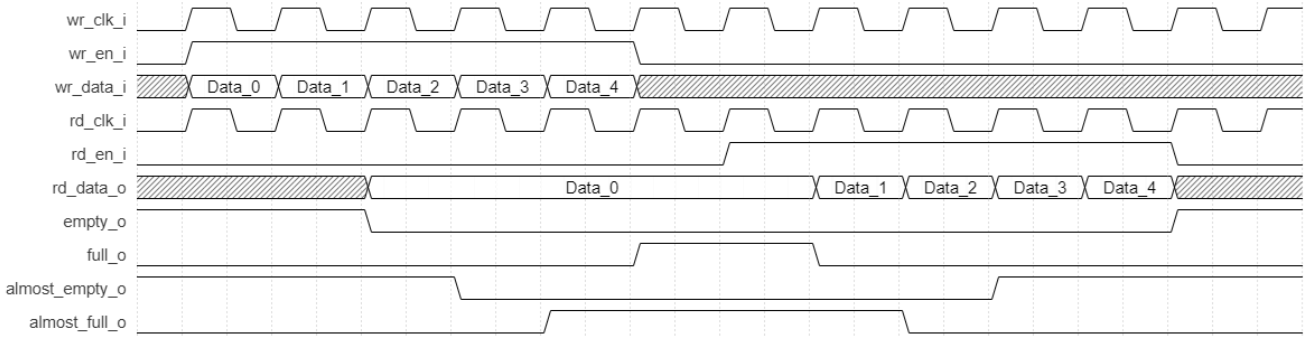

Figure 2.12. FIFO Dual Clock Memory Timing Diagram, FWFT Mode (Area-Optimized/HW-Based FIFO, Avant Devices)

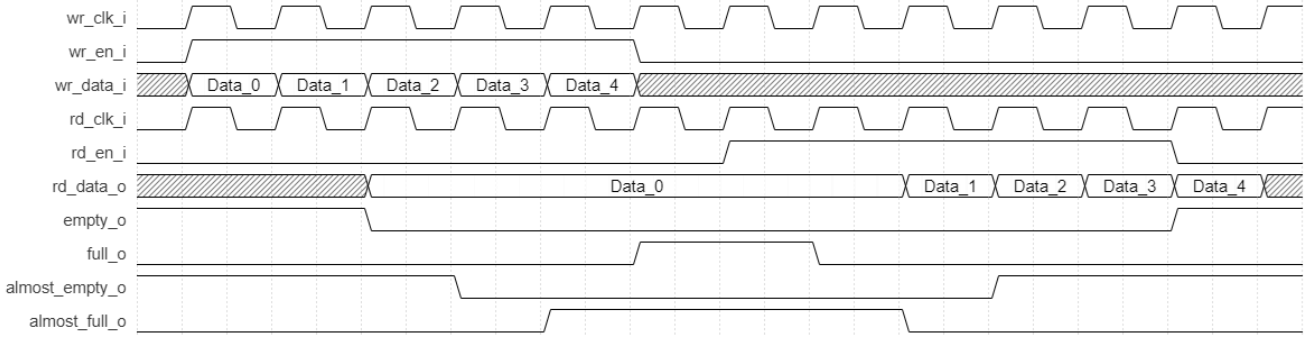

Figure 2.13. FIFO Dual Clock Memory Timing Diagram, FWFT Mode with Output Registers (Area-Optimized/HW-Based FIFO, Avant Devices)

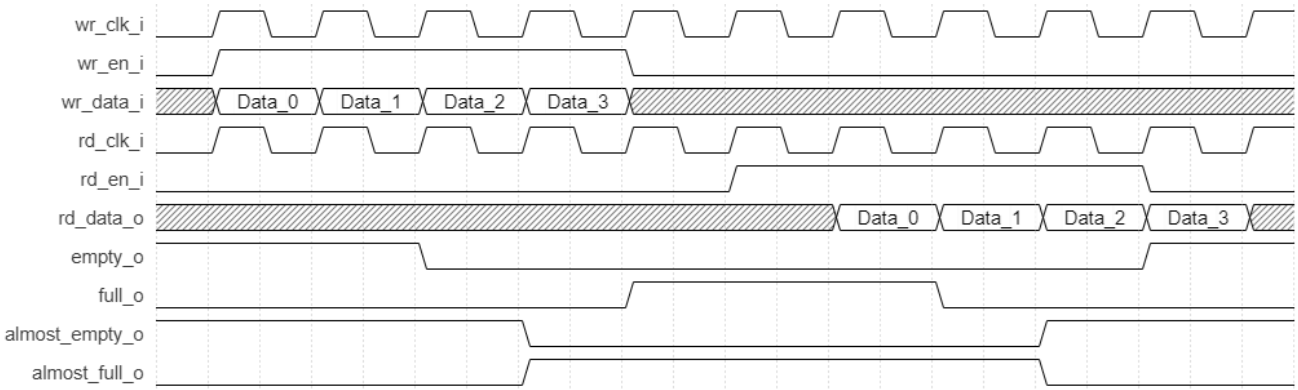

Figure 2.14. FIFO Dual Clock Memory Timing Diagram of Area-Optimized/HW-Based FIFO (Nexus Devices)

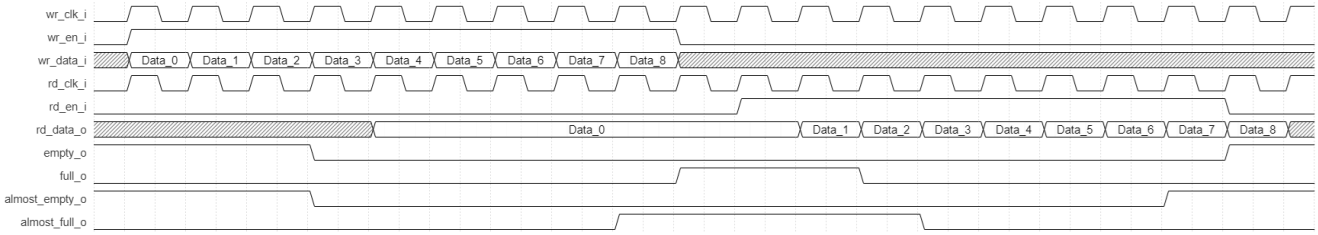

Figure 2.15. FIFO Dual Clock Memory Timing Diagram, FWFT Mode (Area-Optimized/HW-Based FIFO, Nexus Devices)

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

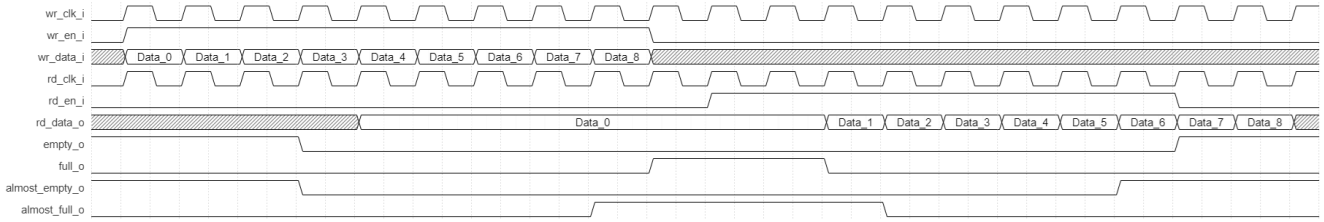

Figure 2.16. FIFO Dual Clock Memory Timing Diagram, FWFT Mode with Output Registers (Area-Optimized/HW-Based FIFO, Nexus Devices)

Reset Differences between Feature-Rich (LUT) and Area-Optimized (HW) based implementations:

The reset behavior of different FIFO items is described in Table 2.6 for both LUT- and HW-based implementations. The timing waveforms for the flags and read data are shown in Figure 2.17 to Figure 2.22. Configuration used are WADDR_DEPTH = 8, RADDR_DEPTH = 8, with output register enabled.

Table 2.6. Feature-Rich (LUT) and Area-Optimized (HW) Reset Behavior

Item	Clock	HW-Controller		LUT-Controller	
		rst_i	rp_rst_i	rst_i	rp_rst_i
Write pointer	wr_clk_i	resets at the rising edge of rst_i	—	resets at the rising edge of rst_i	—
Read pointer[5]	rd_clk_i	resets at the rising edge of rst_i	resets at the rising edge of rp_rst_i	resets at the next positive edge of rd_clk_i	*async*: resets at the rising edge of rp_rst_i *sync*: resets at the next positive edge of rd_clk_i
Write flags (full_o, almost_full_o)	wr_clk_i	**Avant:** updates at the rising edge of rst_i **Nexus:** updates at the next positive edge of wr_clk_i	**Avant**: updates at the next positive edge of wr_clk_i[2] **Nexus**: updates at the rising edge of rp_rst_i	resets at the next positive edge of wr_clk_i	updates at the next positive edge of wr_clk_i after write pointer synchronization[2]
Read flags (empty_o, almost_empty_o)	rd_clk_i	updates at the rising edge of rst_i	**Avant:** updates at the rising edge of rp_rst_i **Nexus:** updates at the next positive edge of rd_clk_i	resets at the next positive edge of rd_clk_i	updates at the next positive edge of rd_clk_i after read pointer resets
Read data (rd_data_o)[1,5]	rd_clk_i	—	*async*: resets at the rising edge of rp_rst_i *sync*: resets at the next positive edge of rd_clk_i	—	*async*: resets at the rising edge of rp_rst_i *sync*: resets at the next positive edge of rd_clk_i
Write data count output[3,5]	wr_clk_i	*Not supported*	*Not supported*	*async*: resets at the rising edge of rst_i *sync*: resets at the next positive edge of wr_clk_i	—
Read data count output[4,5]	rd_clk_i	*Not supported*	*Not supported*	resets at the next positive edge of wr_clk_i	*async*: resets at the rising edge of rp_rst_i *sync*: resets at the next positive edge of rd_clk_i

Notes:

1. If *Enable Output Register* is checked.
2. full_o and almost_full asserts depending on the synchronized value of the write pointer.
3. If *Enable Data Count (Write)* is checked.
4. If *Enable Data Count (Read)* is checked.

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

5. *Reset Mode* is either *sync* or *async*.

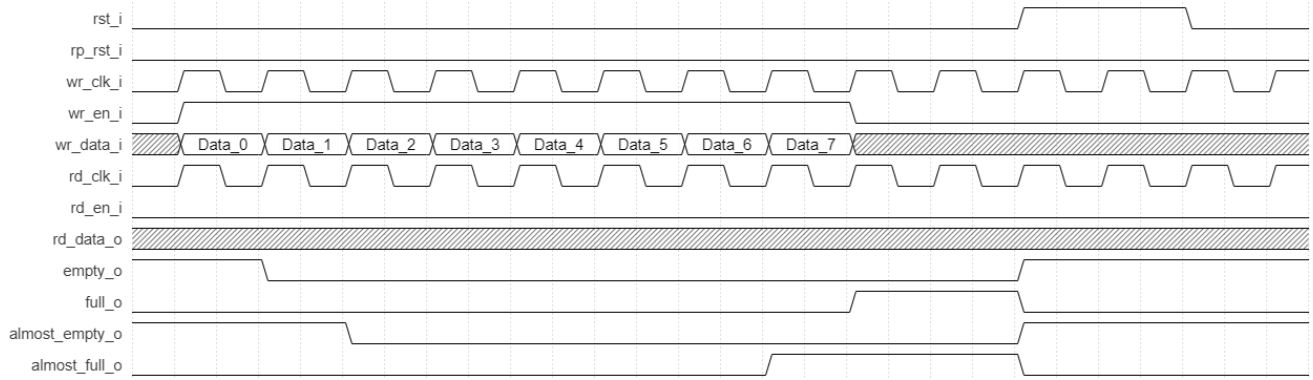

Figure 2.17. rst_i Behavior in Area-Optimized (HW) Timing Diagram (Avant Devices)

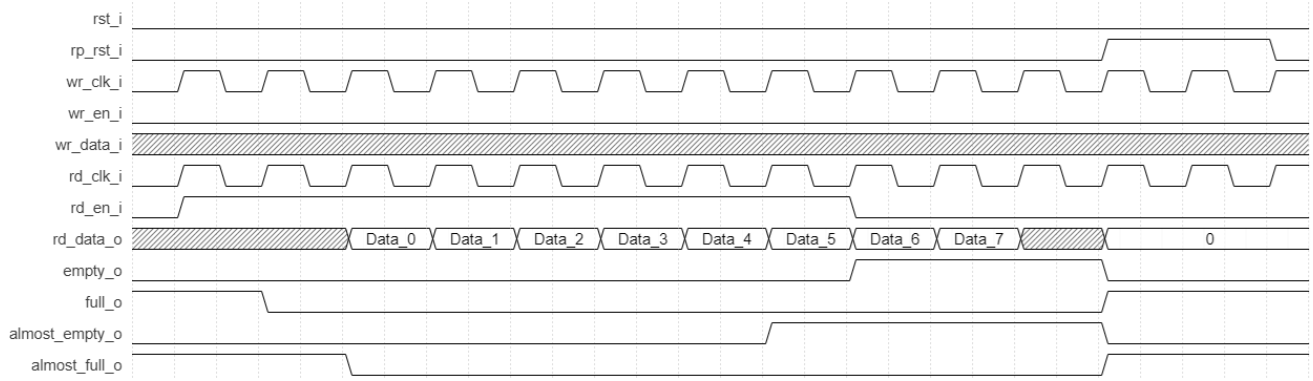

Figure 2.18. rp_rst_i Behavior in Area-Optimized (HW) Timing Diagram (Avant Devices, with Output Registers)

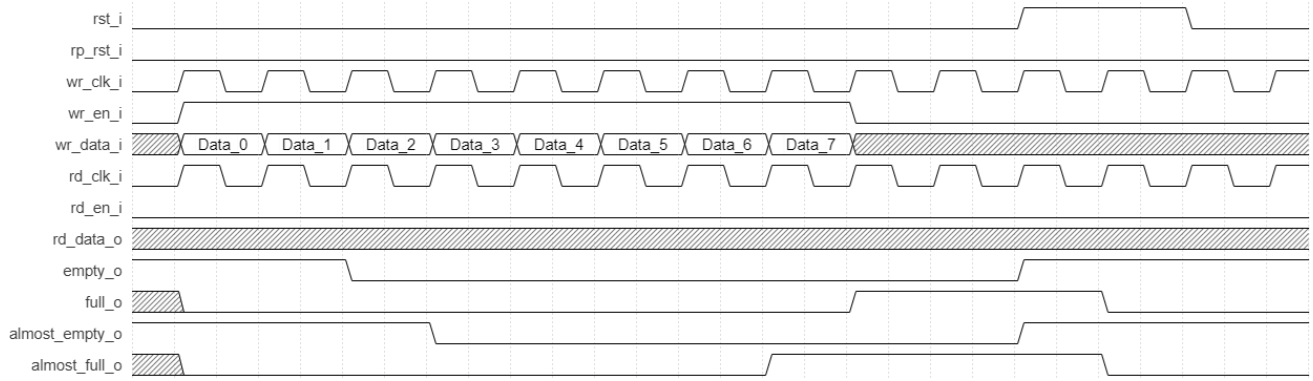

Figure 2.19. rst_i Behavior in Area-Optimized (HW) Timing Diagram (Nexus Devices)

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

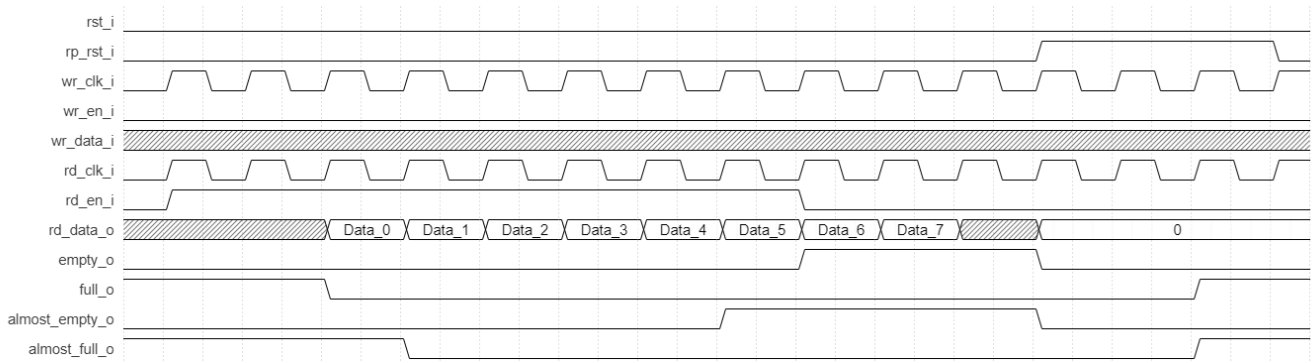

Figure 2.20. rp_rst_i Behavior in Area-Optimized (HW) Timing Diagram (Nexus Devices, with Output Registers)

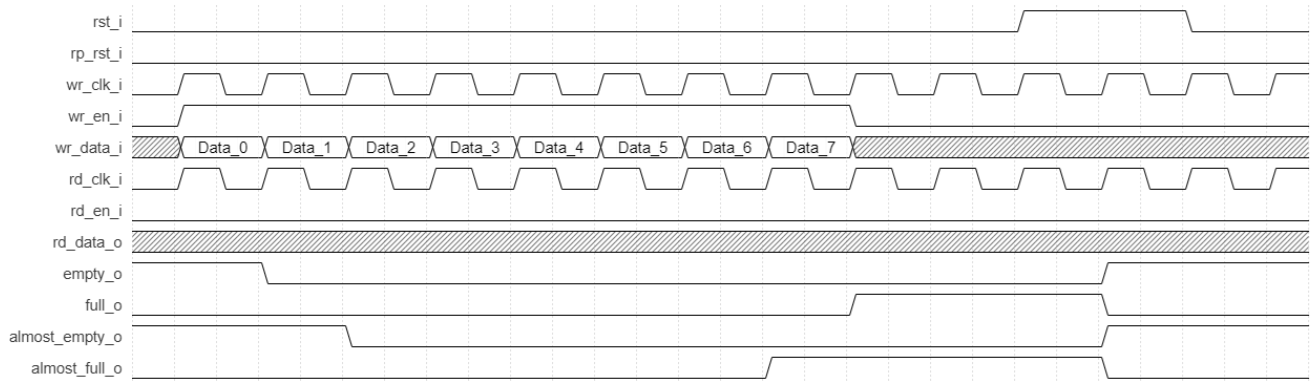

Figure 2.21. rst_i Behavior in Feature-Rich (LUT) Controller Timing Diagram

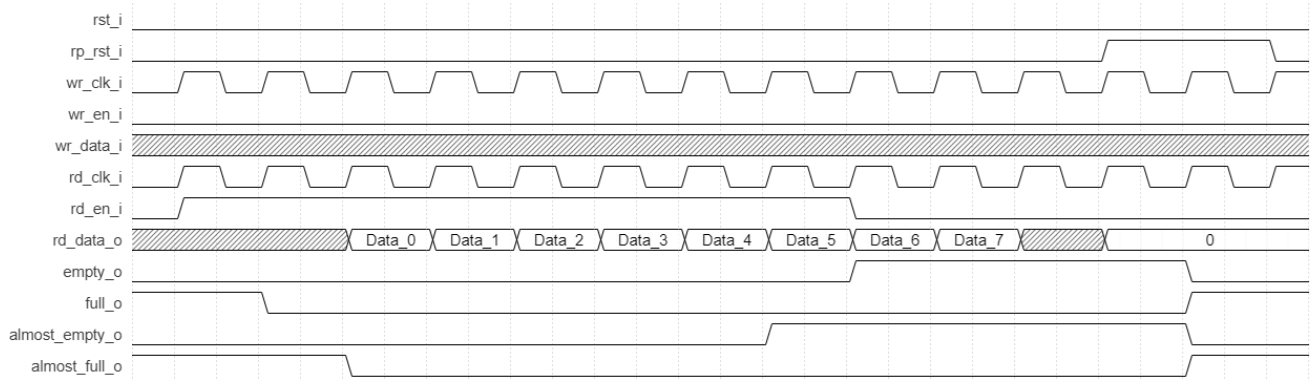

Figure 2.22. rp_rst_i Behavior in Feature-Rich (LUT) Controller Timing Diagram

2.2.1. Write-Once-Read-Many Application

The Read Pointer reset (rp_rst_i) is a read-clock synchronized reset designed to reset the Dual Clock FIFO IP to FULL, and resend the previous set of data, typical for packet transmissions. To use this reset signal, follow these steps:

1. FIFO reset using rst_i.

   Note: rp_rst_i resets the Read Pointer, while rst_i resets both Write and Read Pointers.
2. Write data — you can perform write data until FULL flag is asserted.
3. Read data — you can perform read data until EMPTY flag is asserted.

   Data write directly following a data read is not allowed.
4. Reset using rp_rst_i, and wait for 1 write clock cycle (additional 2 clock cycles are needed for Feature-Rich (LUT) controller implementation).

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

If data was written until FULL flag is asserted in Step 2, the FULL flag asserts again because of the rp_rst_i reset.

If FULL flag is not asserted after rp_rst_i reset, additional Write data is allowed (until FULL flag asserts).

5. Read data — output data set is the same as Step 3, plus the additional write data if any. You can perform reset follows by read sequence (Step 4 to 5) repeatedly.

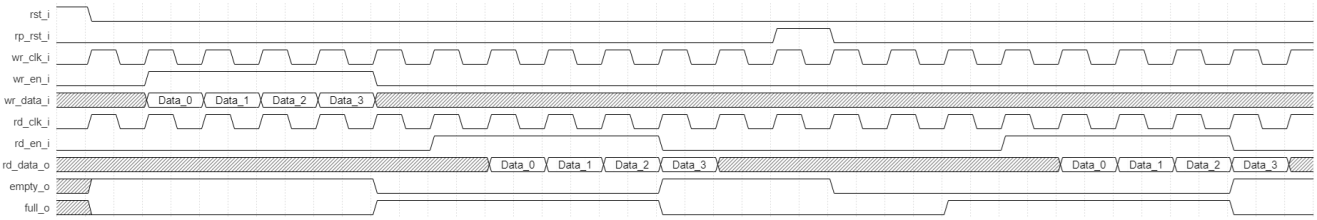

Figure 2.23. FIFO Dual Clock Memory Timing Diagram LUT-Based Controller, Read Pointer Reset Usage

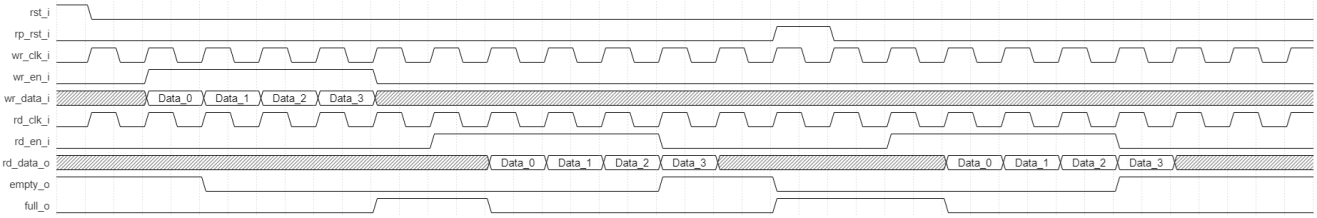

Figure 2.24. FIFO Dual Clock Timing Diagram, Write-Once-Read-Many (Area-Optimized/HW-Based FIFO, Avant Devices)

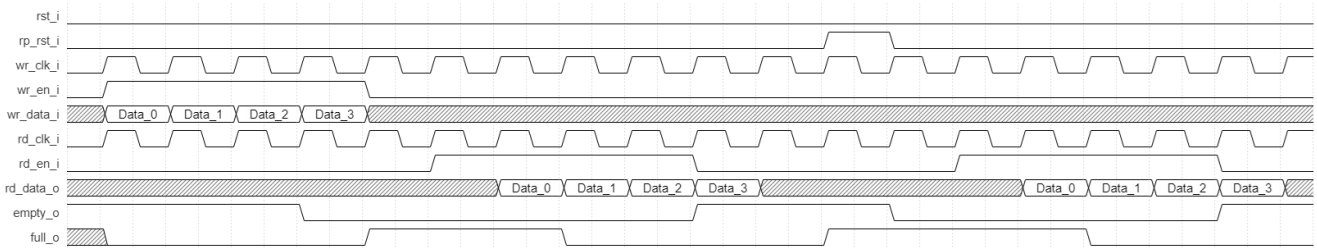

Figure 2.25. FIFO Dual Clock Timing Diagram, Write-Once-Read-Many (Area Optimized/HW-Based FIFO, Nexus Devices)

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

3. IP Generation

This section provides information on how to generate the FIFO 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.

Note: The screenshots provided are for reference only. Details may vary depending on the version of the IP or software being used. If there have been no significant changes to the GUI, a screenshot may reflect an earlier version of the IP.

3.1. Generation

Module/IP Block Wizard in Lattice Radiant Software allows you 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. You can utilize Module/IP on Local to specify a variety of memories in your 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. 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.

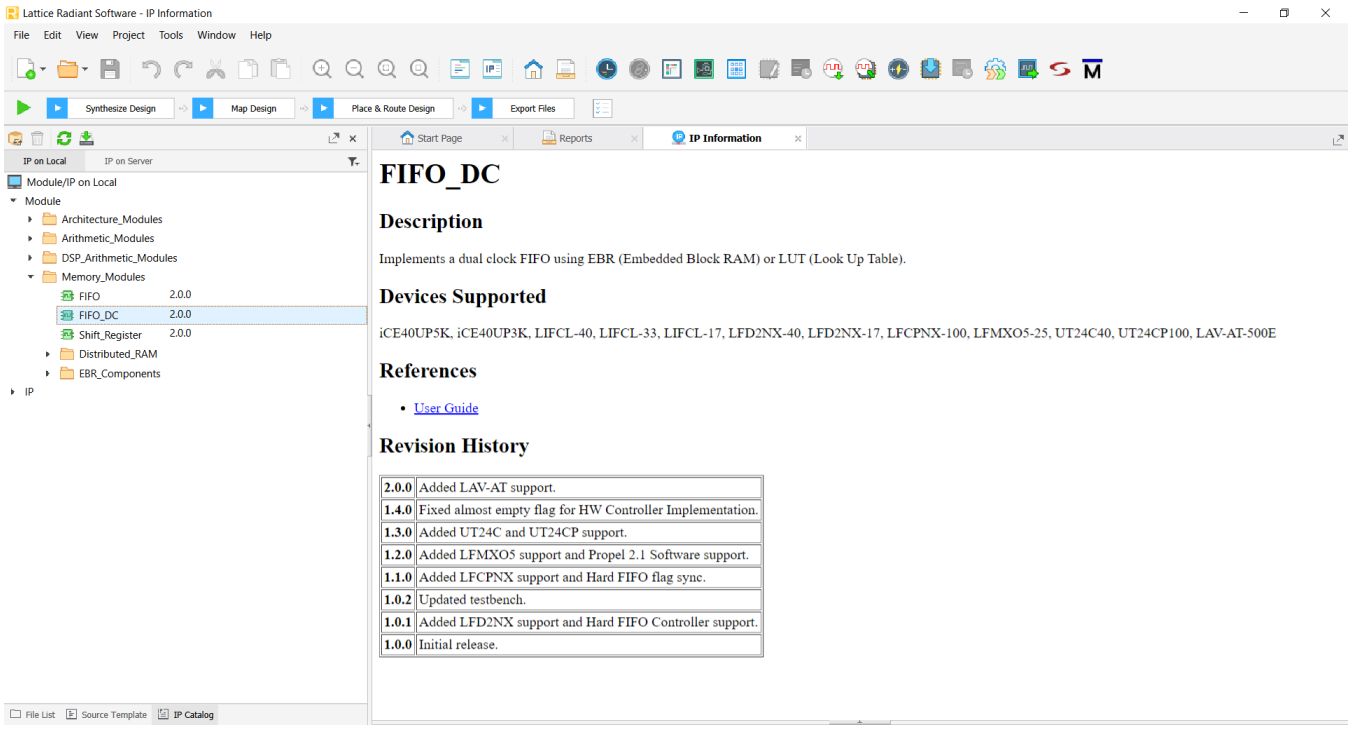

Figure 3.1. Memory Modules Under Module/IP on Local in the Lattice Radiant Software

The following section shows an example of generating a FIFO Single Clock of size 512 x 18.

To generate a FIFO Single Clock of size 512 x 18:

1. Double-click **FIFO** under **Memory_Modules**.
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**.

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

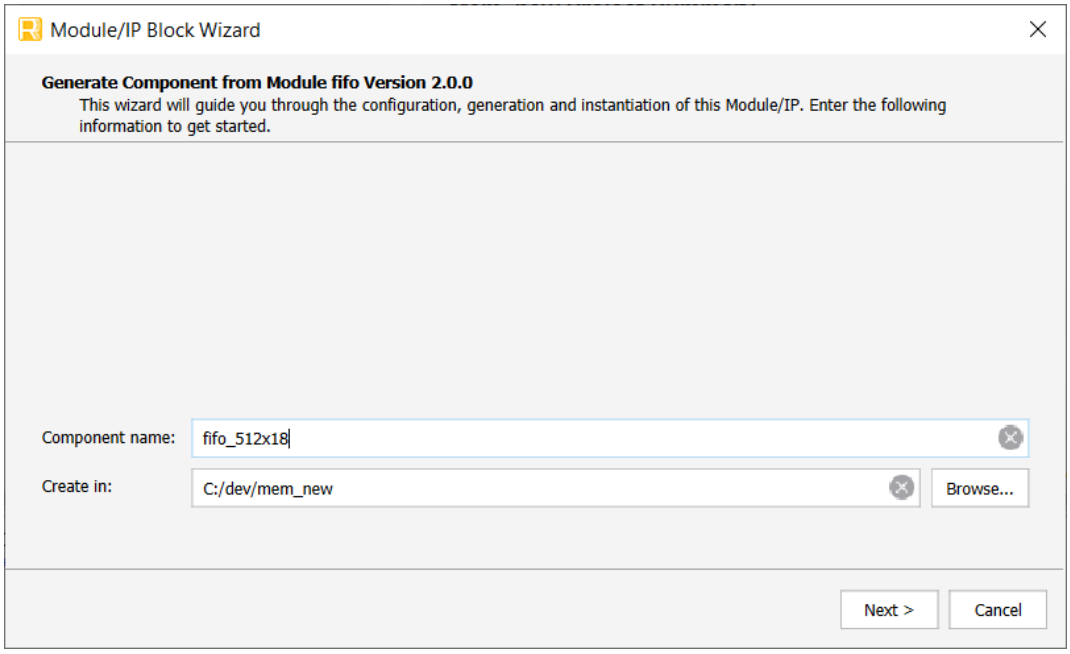

Figure 3.2. Example: Generating FIFO Using Module/IP Block Wizard

3. In the module's dialog box of the **Module/IP Block Wizard** window, customize the FIFO Single Clock using drop-down menus and check boxes as shown in Figure 3.3.

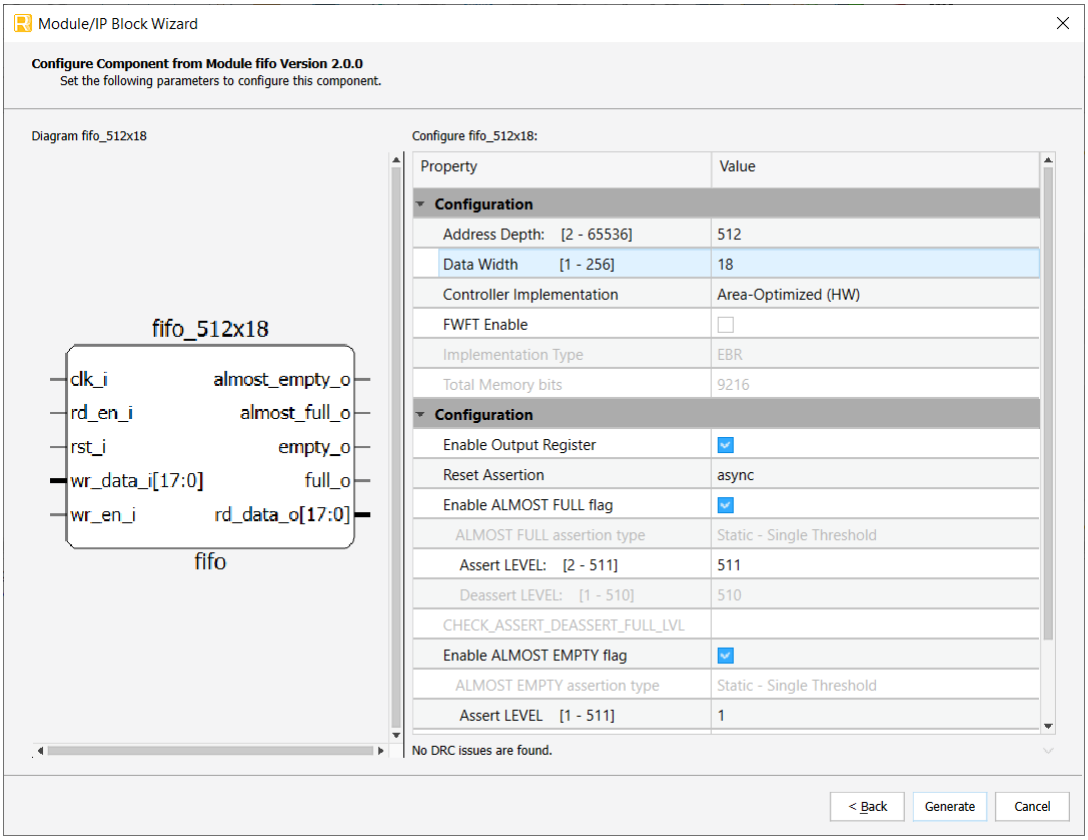

Figure 3.3. Example: Generating FIFO Module Customization

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

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.

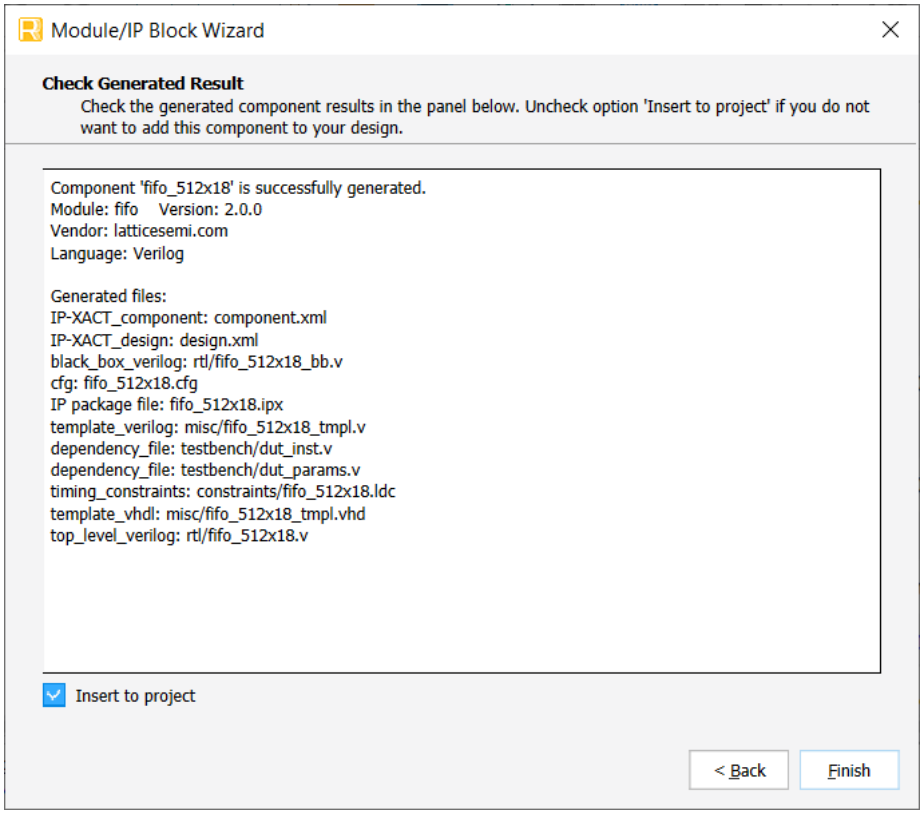

Figure 3.4. Example: Generating FIFO Check Generated Result

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

Table 3.1. Generated File List

File	Description
<Component name>.ipx	Contains the information on the files associated to the generated IP.
<Component name>.cfg	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	Provides an example RTL top file that instantiates the IP core.
rtl/<Component name>_bb.v	Provides the synthesis closed-box.
misc/<Component name>_tmpl.v misc /<Component name>_tmpl.vhd	Provide instance templates for the IP core.

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

3.2. Running Functional Simulation

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

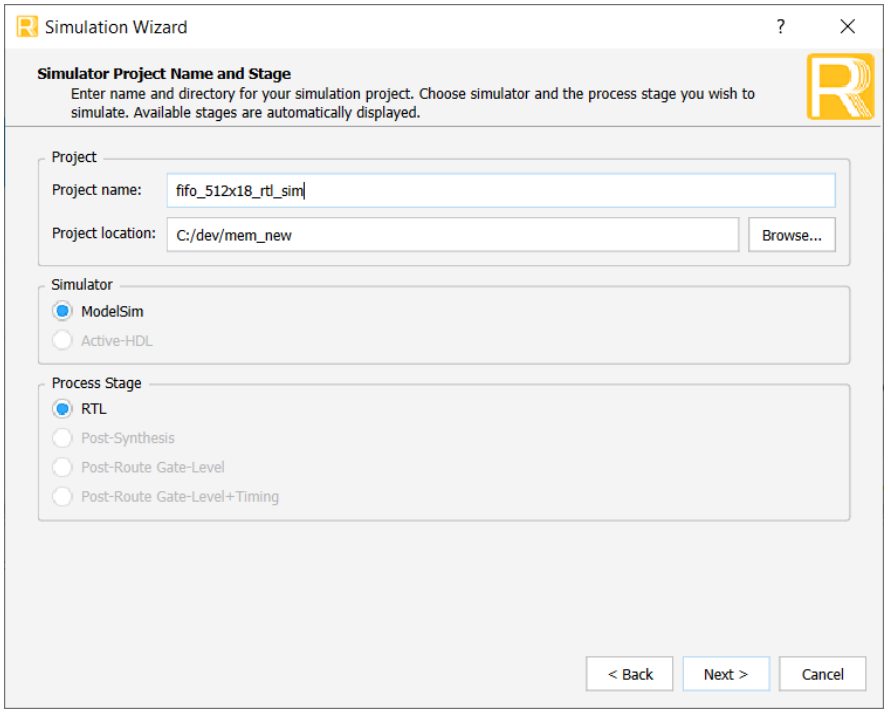

Figure 3.5. Simulation Wizard Simulator Project Name and Stage

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

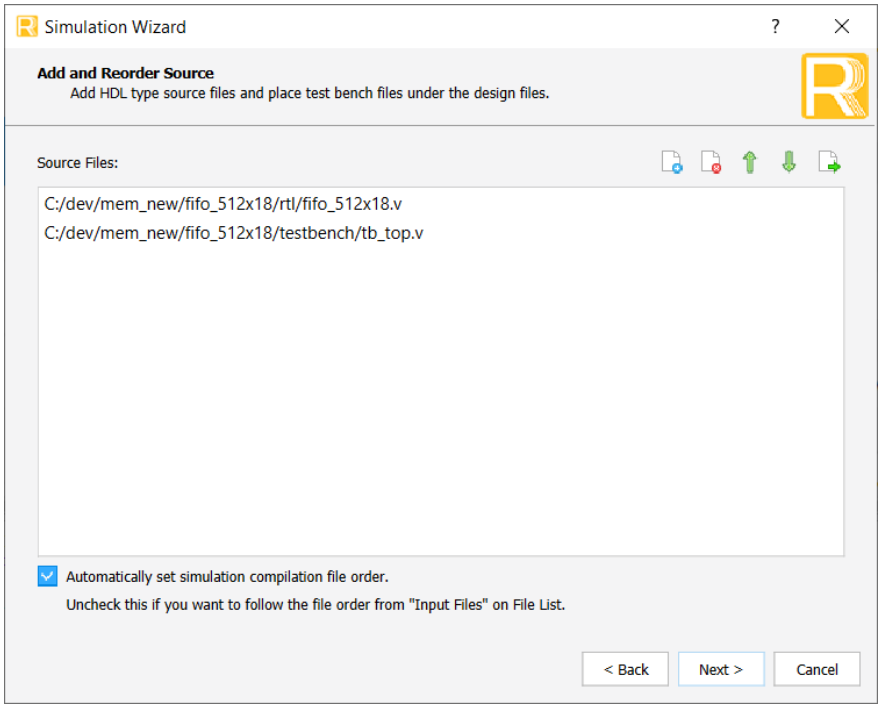

Figure 3.6. Simulation Wizard Add and Reorder Source

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

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

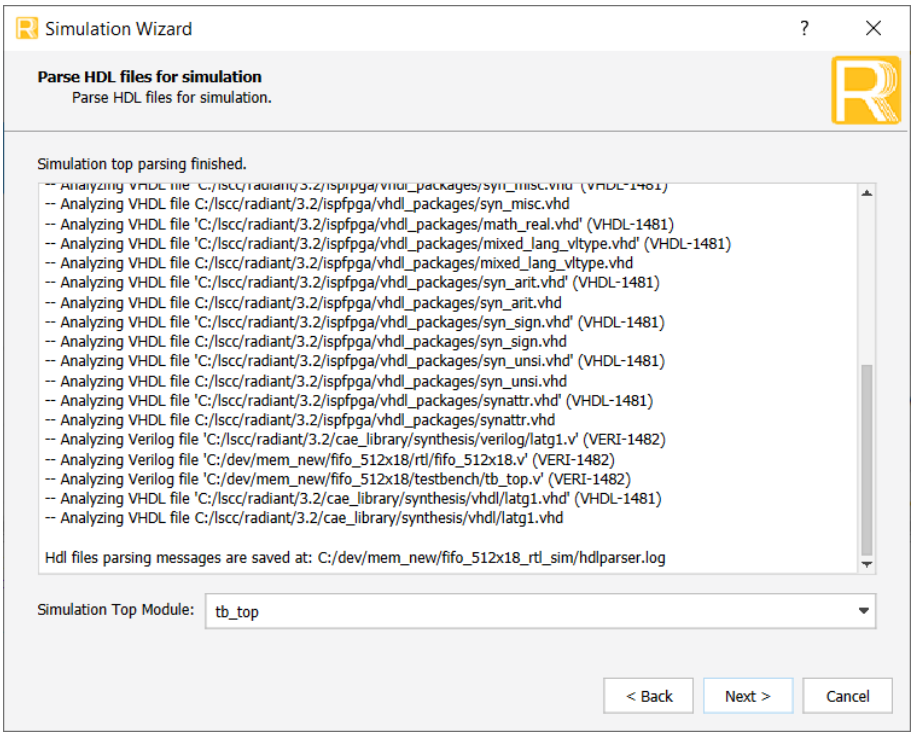

Figure 3.7. Simulation Wizard Parse HDL files for simulation

9. 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.

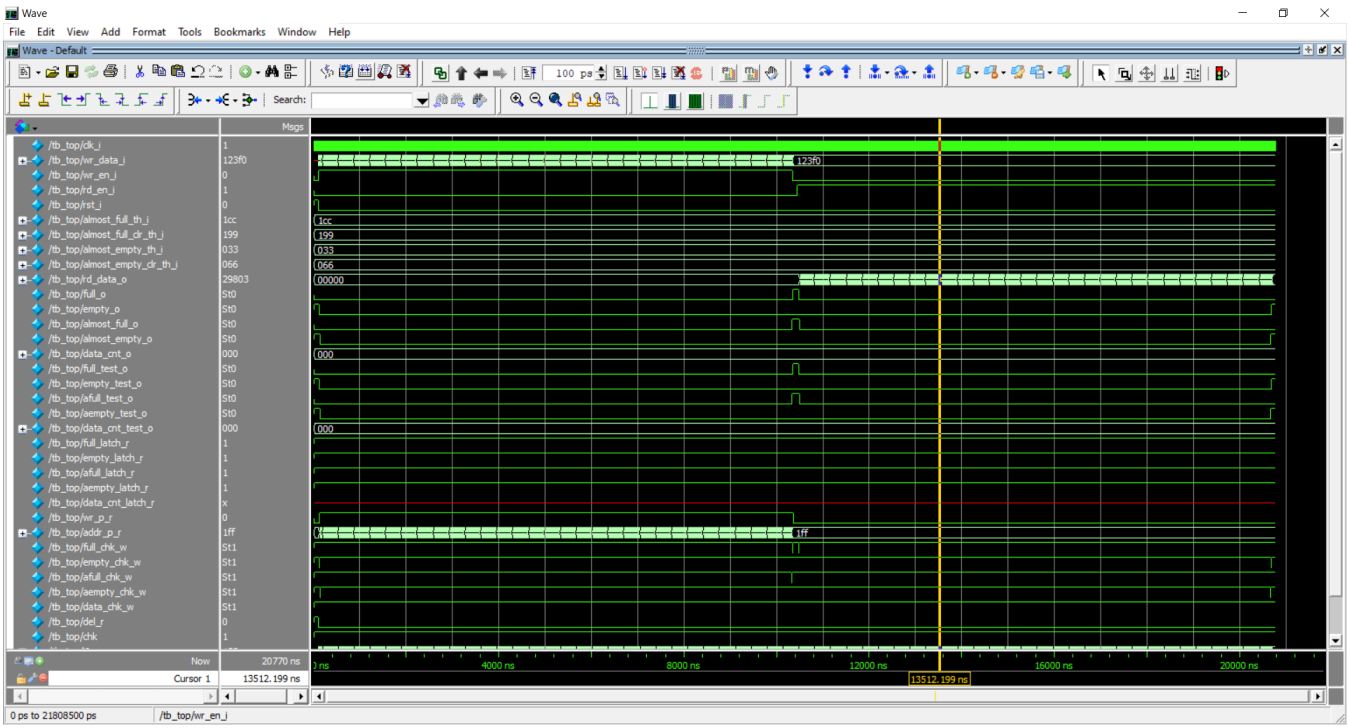

Figure 3.8. Simulation Waveform

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

3.3. Constraining the IP

You need to provide proper timing and physical design constraints to ensure that your design meets the desired performance goals on the FPGA. Add the content of the following IP constraint file to your design constraints:

<IP_Instance_Path>/<IP_Instance_Name>/eval/constraint.pdc

The constraint file has been verified during IP evaluation with the IP instantiated directly in the top-level module. You can modify the constraints in this file with thorough understanding of the effect of each constraint.

To use this constraint file, copy the contents of the *constraint.pdc* to the top-level design constraint for post-synthesis.

Refer to Lattice Radiant Timing Constraints Methodology (FPGA-AN-02095) for details on how to constrain your design.

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

4. PMI Support

The following sections discuss the different PMI modules which can be utilized for entering custom parameters for the FIFO Single Clock memory and FIFO Dual Clock memory. 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_fifo

The following are port descriptions, Verilog and VHDL definitions, and parameter descriptions for pmi_fifo (First In First Out Single Clock Memory Module).

Table 4.1. First In First Out Single Clock Memory PMI Port Definitions

Direction	Port Name	Type	Size (Buses Only)
I	Data	Bus	(pmi_data_width - 1):0
I	Clock	Bit	N/A
I	WrEn	Bit	N/A
I	RdEn	Bit	N/A
I	Reset	Bit	N/A
O	Q	Bus	(pmi_data_width - 1):0
O	Empty	Bit	N/A
O	Full	Bit	N/A
O	AlmostEmpty	Bit	N/A
O	AlmostFull	Bit	N/A

Table 4.2. First In First Out Single Clock Memory PMI Attribute Definitions

Attributes	Description	Values	Default Value
pmi_data_width	FIFO data width.	2–<Max that can fit in the device>	8
pmi_data_depth	FIFO address depth.	1–<Max that can fit in the device>	256
pmi_full_flag[1]	Defines the full flag.	2–<Max that can fit in the device>	256
pmi_empty_flag[1]	Defines the empty flag.	1–<Max that can fit in the device>	0
pmi_almost_full_flag	Defines the lower limit of the almost full flag	2–<Max that can fit in the device>	252
pmi_almost_empty_flag	Defines the upper limit of the almost empty flag.	1–<Max that can fit in the device>	4
pmi_regmode	Data Out for FIFO that can be registered or not using this selection.	*reg*, *noreg*	*reg*
pmi_family	Defines the FPGA family being used in the module	*LAV-AT*, *common*	*common*
pmi_implementation[2]	Refers to the memory used by the FIFO module	*EBR*, *LUT*, *HARD_IP*	*EBR*
module_type [1]	Refers to the type of pmi module.	*pmi_fifo*	*pmi_fifo*

Notes:

1. Unused. *pmi_full_flag* takes the value of *pmi_data_depth*, *pmi_empty_flag* is tied to 0.
2. *pmi_implementation* uses a soft controller unless set to *Hard_IP*.

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

Verilog `pmi_fifo` Definition

```
module pmi_fifo #(
       parameter pmi_data_width = 8,
       parameter pmi_data_depth = 256,
       parameter pmi_full_flag = 256,
       parameter pmi_empty_flag = 0,
       parameter pmi_almost_full_flag = 252,
       parameter pmi_almost_empty_flag = 4,
       parameter pmi_regmode = "reg",
       parameter pmi_family = "common" ,
       parameter module_type = "pmi_fifo",
       parameter pmi_implementation = "EBR"
)(
   input  [pmi_data_width-1:0] Data,
   input Clock,
   input WrEn,
   input RdEn,
   input Reset,
   output [pmi_data_width-1:0] Q,
   output Empty,
   output Full,
   output AlmostEmpty,
   output AlmostFull);

endmodule // pmi_fifo
```

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

VHDL `pmi_fifo` Definition

```
component pmi_fifo is
     generic (
       pmi_data_width : integer := 8;
       pmi_data_depth : integer := 256;
       pmi_full_flag : integer := 256;
       pmi_empty_flag : integer := 0;
       pmi_almost_full_flag : integer := 252;
       pmi_almost_empty_flag : integer := 4;
       pmi_regmode : string := "reg";
       pmi_family : string := "common" ;
       module_type : string := "pmi_fifo";
       pmi_implementation : string := "EBR"
    );
    port (
     Data : in std_logic_vector(pmi_data_width-1 downto 0);
     Clock: in std_logic;
     WrEn: in std_logic;
     RdEn: in std_logic;
     Reset: in std_logic;
     Q : out std_logic_vector(pmi_data_width-1 downto 0);
     Empty: out std_logic;
     Full: out std_logic;
     AlmostEmpty: out std_logic;
     AlmostFull: out std_logic
   );
  end component pmi_fifo;
```

4.2. pmi_fifo_dc

The following are port descriptions, Verilog and VHDL definitions, and parameter descriptions for pmi_fifo_dc (First in First Out Dual Clock Memory Module).

Table 4.3. First in First Out Dual Clock Memory PMI Port Definitions

Direction	Port Name	Type	Size (Buses Only)
I	Data	Bus	(pmi_data_width_w - 1):0
I	WrClock	Bit	N/A
I	RdClock	Bit	N/A
I	WrEn	Bit	N/A
I	RdEn	Bit	N/A
I	Reset	Bit	N/A
I	RPReset	Bit	N/A
O	Q	Bus	(pmi_ data_width_r - 1):0
O	Empty	Bit	N/A
O	Full	Bit	N/A
O	AlmostEmpty	Bit	N/A
O	AlmostFull	Bit	N/A

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

Table 4.4. First in First Out Dual Clock Memory PMI Attribute Definitions

Attributes	Description	Values	Default Value
pmi_data_width_w	FIFO data write port size.	2–<Max that can fit in the device>	18
pmi_addr_depth_w	FIFO address depth write port size.	1–<Max that can fit in the device>	256
pmi_data_width_r	FIFO data read port size.	2–<Max that can fit in the device>	18
pmi_addr_depth_r	FIFO address depth read port size.	1–<Max that can fit in the device>	256
pmi_full_flag[1]	Defines the full flag.	2–<Max that can fit in the device>	256
pmi_empty_flag[1]	Defines the empty flag.	1–<Max that can fit in the device>	0
pmi_almost_full_flag	Defines the lower limit of the almost full flag	2–<Max that can fit in the device>	252
pmi_almost_empty_flag	Defines the upper limit of the almost empty flag.	1–<Max that can fit in the device>	4
pmi_regmode	Data Out for FIFO, if can be registered or not using this selection.	*reg*, *noreg*	*reg*
pmi_resetmode	Selection for the Reset to be Synchronous or Asynchronous to the Clock.	*async*, *sync*	*async*
pmi_family	Defines the FPGA family being used in the module	*LAV-AT*, *common*	*common*
pmi_implementation[2]	Refers to the memory used by the FIFO module	*EBR*, LUT, HARD_IP	*EBR*
module_type	Refers to the type of pmi module.	*pmi_fifo_dc*	*pmi_fifo_dc*

Notes:

1. Unused. *pmi_full_flag* takes the value of *pmi_data_depth*, *pmi_empty_flag* is tied to 0.
2. *pmi_implementation* uses a soft controller unless set to *Hard_IP*.

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

Verilog `pmi_fifo_dc` Definition

```
module pmi_fifo_dc #(
     parameter pmi_data_width_w = 18,
     parameter pmi_data_width_r = 18,
     parameter pmi_data_depth_w = 256,
     parameter pmi_data_depth_r = 256,
     parameter pmi_full_flag = 256,
     parameter pmi_empty_flag = 0,
     parameter pmi_almost_full_flag = 252,
     parameter pmi_almost_empty_flag = 4,
     parameter pmi_regmode = "reg",
     parameter pmi_resetmode = "async",
     parameter pmi_family = "common" ,
     parameter module_type = "pmi_fifo_dc",
     parameter pmi_implementation = "EBR"
     ) (
   input  [pmi_data_width_w-1:0] Data,
   input WrClock,
   input RdClock,
   input WrEn,
   input RdEn,
   input Reset,
   input RPReset,
   output [pmi_data_width_r-1:0] Q,
   output Empty,
   output Full,
   output AlmostEmpty,
   output AlmostFull);
endmodule // pmi_fifo_dc
```

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

VHDL `pmi_fifo_dc` Definition

```
component pmi_fifo_dc is
     generic (
       pmi_data_width_w : integer := 18;
       pmi_data_width_r : integer := 18;
       pmi_data_depth_w : integer := 256;
       pmi_data_depth_r : integer := 256;
       pmi_full_flag : integer := 256;
       pmi_empty_flag : integer := 0;
       pmi_almost_full_flag : integer := 252;
       pmi_almost_empty_flag : integer := 4;
       pmi_regmode : string := "reg";
       pmi_resetmode : string := "async";
       pmi_family : string := "common" ;
       module_type : string := "pmi_fifo_dc";
       pmi_implementation : string := "EBR"

    );
    port (
     Data : in std_logic_vector(pmi_data_width_w-1 downto 0);
     WrClock: in std_logic;
     RdClock: in std_logic;
     WrEn: in std_logic;
     RdEn: in std_logic;
     Reset: in std_logic;
     RPReset: in std_logic;
     Q : out std_logic_vector(pmi_data_width_r-1 downto 0);
     Empty: out std_logic;
     Full: out std_logic;
     AlmostEmpty: out std_logic;
     AlmostFull: out std_logic
   );
  end component pmi_fifo_dc;
```

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

Appendix A. Resource Utilization

The following tables show the resource utilization of the FIFO and FIFO_DC IPs for the LAV-AT-E70-2LFG1156C and LFCPNX-100-8LFG672C devices using Synplify Pro of the Lattice Radiant software 2023.2. Various configurations are used to show the effect of different memory sizes and implementations in the resource usage.

Table A.1. FIFO Resource Utilization for LAV-AT-E70-2LFG1156C

Memory Size	REG_EN	Almost FLAGs	FIFO Controller	Memory Implementation	Register	LUT	EBR
1024 × 18	*reg*	Yes	HARD_IP (HW)	EBR	4	0	1
1024 × 18	*reg*	Yes	FABRIC (LUT)	EBR	178	211	1
1024 × 18	*reg*	Yes	FABRIC (LUT)	LUT	196	2396	0

Table A.2. FIFO_DC Resource Utilization for LAV-AT-E70-2LFG1156C

Write Configuration	Read Configuration	REG_EN	Almost FLAGs	FIFO Controller	Memory Implementation	Register	LUT	EBR
512 × 18	512 × 18	*reg*	Yes	HARD_IP (HW)	EBR	0	40	1
512 × 18	512 × 18	*reg*	Yes	FABRIC (LUT)	EBR	107	181	1
512 × 18	512 × 18	*reg*	Yes	FABRIC (LUT)	LUT	173	1333	0
512 × 18	512 × 18	*noreg*	No	FABRIC (LUT)	EBR	85	136	1
16384 × 16	8192 × 32	*reg*	Yes	HARD_IP (HW)	EBR	0	80	8
16384 × 16	8192 × 32	*reg*	Yes	FABRIC (LUT)	EBR	149	256	8

Table A.3. FIFO Resource Utilization for LFCPNX-100-8LFG672C

Memory Size	REG_EN	Almost FLAGs	FIFO Controller	Memory Implementation	Register	LUT	EBR
1024 × 18	*reg*	Yes	HARD_IP (HW)	EBR	18	1	2
1024 × 18	*reg*	Yes	FABRIC (LUT)	EBR	178	141	1
1024 × 18	*reg*	Yes	FABRIC (LUT)	LUT	196	3323	0

Table A.4. FIFO_DC Resource Utilization for LFCPNX-100-8LFG672C

Write Configuration	Read Configuration	REG_EN	Almost FLAGs	FIFO Controller	Memory Implementation	Register	LUT	EBR
512 × 18	512 × 18	*reg*	Yes	HARD_IP (HW)	EBR	24	37	1
512 × 18	512 × 18	*reg*	Yes	FABRIC (LUT)	EBR	125	196	1
512 × 18	512 × 18	*reg*	Yes	FABRIC (LUT)	LUT	143	1777	0
512 × 18	512 × 18	*noreg*	No	FABRIC (LUT)	EBR	103	139	1
16384 × 16	8192 × 32	*reg*	Yes	HARD_IP (HW)	EBR	211	176	20
16384 × 16	8192 × 32	*reg*	Yes	FABRIC (LUT)	EBR	181	272	16

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

References

- FIFO Memory Modules Release Notes (FPGA-RN-02095)
- Certus-N2 web page
- Certus-NX web page
- CertusPro-NX web page
- CrossLink-NX web page
- iCE40 UltraPlus web page
- MachXO4 web page
- MachXO5-NX web page
- Lattice Avant Platform web page
- Lattice Radiant Timing Constraints Methodology (FPGA-AN-02095)
- Lattice IP Modules web page
- Lattice Radiant Software web page
- Lattice Propel Design Environment web page
- Lattice Insights for Lattice Semiconductor training courses and learning plans

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

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.

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

Revision History

Note: In some instances, the IP may be updated without changes to the user guide. The user guide may reflect an earlier IP version but remains fully compatible with the later IP version. Refer to the IP Release Notes for the latest updates.

Revision 1.6, March 2026

Section	Change Summary
Introduction	Updated the IP and software version for FIFO_DC (MachXO4) in Table 1.1. FIFO Memory Modules Quick Facts: • IP Version: 3.0.1 • Lattice Radiant Software: 2025.2.1

Memory Modules	• Updated the *rst_i* and *rp_rst_i* ports descriptions in Table 2.3. First in First Out Dual Clock Memory Port Definitions
	• Removed table notes associated with limitation previously mentioned in Table 2.5. First in First Out Dual Clock Memory Attribute Definitions for MachXO4 Devices. The functionalities are now fully supported:
	  • Added support for LUT memory type
	  • Added *Static – Dual Threshold*, *Dynamic – Single Threshold*, *and Dynamic – Dual Threshold* support for almost full and almost empty flags in EBR

Revision 1.5, December 2025

Section	Change Summary
All	• Added a note on the IP version in the *Revision History* section. • Minor editorial edits.
Abbreviations in This Document	Changed section title to Abbreviations in This Document.
Introduction	Added sections 1.1. Quick Facts and 1.2. Features.
Memory Modules	Added Table 2.5. First in First Out Dual Clock Memory Attribute Definitions for MachXO4 Devices.
IP Generation	Added note indicating GUI screenshots are only for reference and may vary by IP or software version.
References	Updated references.

Revision 1.4, March 2024

Section	Change Summary
All	• Renamed document from *FIFO Memory Modules - Lattice Radiant Software* to *FIFO Memory Modules*. • Performed minor formatting and typo edits.
Inclusive Language	Added inclusive language boilerplate.
Introduction	Removed mention of Lattice Avant devices as this document supports across devices.

Memory Modules	• Updated the note for rp_rst_i in Table 2.3. First in First Out Dual Clock Memory Port Definitions.
	• Updated description for the following implementations in the First in First Out Dual Clock (FIFO_DC) section.
	  • Latency Differences between Feature-Rich (LUT) and Area-Optimized (HW) based implementations
	  • Reset Differences between Feature-Rich (LUT) and Area-Optimized (HW) based implementations
	• Updated Table 2.5. Feature-Rich (LUT) and Area-Optimized (HW) Reset Behavior.
	• Removed the following diagrams:
	  • FIFO Dual Clock Memory Timing Diagram, Read Pointer Reset Usage
	  • FIFO Dual Clock Memory Timing Diagram, Read Pointer Reset Usage (Synchronous Reset)
	  • FIFO Dual Clock Timing Diagram, rp_rst_i Usage (Area-Optimized/HW-Based FIFO, Avant Devices)
	• Updated the following diagrams:

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

	• Figure 2.17. rst_i Behavior in Area-Optimized (HW) Timing Diagram (Avant Devices) • Figure 2.18. rp_rst_i Behavior in Area-Optimized (HW) Timing Diagram (Avant Devices, with Output Registers) • Figure 2.21. rst_i Behavior in Feature-Rich (LUT) Controller Timing Diagram • Figure 2.22. rp_rst_i Behavior in Feature-Rich (LUT) Controller Timing Diagram • Added the following diagrams: • Figure 2.19. rst_i Behavior in Area-Optimized (HW) Timing Diagram (Nexus Devices) • Figure 2.20. rp_rst_i Behavior in Area-Optimized (HW) Timing Diagram (Nexus Devices, with Output Registers) • Added the Write-Once-Read-Many Application section.
Resource Utilization	Updated resource utilization per the Lattice Radiant software 2023.2.
References	Updated references.

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

Revision 1.3, December 2023

Section	Change Summary
All	Performed minor editorial fixes.
Disclaimers	Updated this section.

Memory Modules	• Updated details on FWFT.
	• Updated the following tables and their respective table notes:
	  • Table 2.1. First in First Out Single Clock Memory Port Definitions.
	  • Table 2.2. First in First Out Single Clock Memory Attribute Definitions.
	  • Table 2.3. First in First Out Dual Clock Memory Port Definitions.
	  • Table 2.4. First in First Out Dual Clock Memory Attribute Definitions.
	• Added the following figures: Figure 2.4, Figure 2.9, Figure 2.10, Figure 2.11, Figure 2.12, Figure 2.14, Figure 2.15, Figure 2.16, Figure 2.17, Figure 2.18, and Figure 2.19.
	• Updated the respective figure captions for the remaining figures in this chapter.
	• Removed section 2.3 First-Word Fall-Through (FWFT) Mode.

IP Generation	Added section 3.3 Constraining the IP.
Appendix A	Updated Table A.1. Device and Tool Tested.
References	Added this section.

Revision 1.2, June 2023

Section	Change Summary
All	Overall fixes and updates
Memory Modules	• Updated details on FWFT • Updated Figure 2.13. FIFO First-Word Fall-Through Mode Timing Diagram (No output register) • Added Figure 2.14. FIFO First-Word-Fall-Through Mode Timing Diagram (With output register)
Technical Support Assistance	Added reference link to the Lattice Answer Database.

Revision 1.1, November 2022

Section	Change Summary
All	Overall updates
Memory Modules	• Added notes on rp_rst sequence cases • Update Figure 2.6. First in First Out Dual Clock Memory Timing Diagram, With Registers • Updated Figure 2.7. First in First Out Dual Clock Memory Timing Diagram, Mixed-Width Without registers • Added Table 2.5 Feature-Rich (LUT) and Area-Optimized (HW) Reset Behavior • Updated Figure 2.9. rst_i Behavior in Hard Controller First Timing Diagram • Updated Figure 2.13. FIFO First-Word Fall-Through Mode Timing Diagram (No output register)
IP Generation	• Updated Figure 3.10. Memory Modules Under Module/IP on Local in the Lattice Radiant Software
PMI Support	• Updated Table 4.2. First In First Out Single Clock Memory PMI Attribute Definitions for the PMI attribute values • Updated Table 4.4. First in First Out Dual Clock Memory PMI Attribute Definitions for the PMI attribute values
Appendix A	• Updated Table A. 1. Device and Tool Tested • Updated Resource Utilization data in Table A. 2. First in First Out Single Clock Memory Utilization and Table A. 3. First in First Out Dual Clock Memory Utilization

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

Revision 1.0, May 2022

Section	Change Summary
All	Initial release

© 2026 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

www.latticesemi.com