Tightly-Coupled Memory - Lattice Propel Builder 2023.1

User Guide

FPGA-IPUG-02231-1.1

July 2023

Disclaimers

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

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
ASCII	American Standard Code for Information Interchange
EBR	Embedded Block RAM
ECC	Error Correction Code
FIFO	First In, First Out
FPGA	Field Programmable Gate Array
RAM	Random Access Memory
ROM	Read-Only Memory
TCM	Tightly-Coupled Memory

1. Introduction

This document provides technical information about the Tightly-Coupled Memory (TCM) IP and aims to provide information essential for IP/system developing, verification, integration, testing, and validation.

The design is implemented in Verilog HDL. The IP can be configured and generated based on Table 1.1.

Table 1.1. FPGA Software for IP Configuration, Generation, and Implementation

Supported FPGA Family	IP Configuration and Generation	IP Implementation (Synthesis, Map, Place and Route)
Lattice Avant™	Lattice Propel™ Builder software	Lattice Radiant™ software
MachXO5™-NX	Lattice Propel Builder software	Lattice Radiant software
CrossLink™-NX	Lattice Propel Builder software	Lattice Radiant software
CertusPro™-NX	Lattice Propel Builder software	Lattice Radiant software
Certus™-NX	Lattice Propel Builder software	Lattice Radiant software

1.1. Features

- Configurable as single-port or dual-port memory
- Core memory can be implemented as EBR, LRAM, or Distributed RAM
- Supports ROM and RAM mode
- Supports byte enable configurable
- Supports byte writes when used with compatible hardware
- Data width configurable
- Memory depth configurable
- Address range configurable for every port
- Uses 32-bit data word transfers
- Unaligned Access Port for every port
- Uses Little-endian bit structure
- Burst write and read feature

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
- _*io* are bi-directional input/output signals

1.2.3. Attribute Names

Attribute names in this document are formatted in title case and italicized (Attribute Name).

2. Functional Descriptions

2.1. Block Diagram

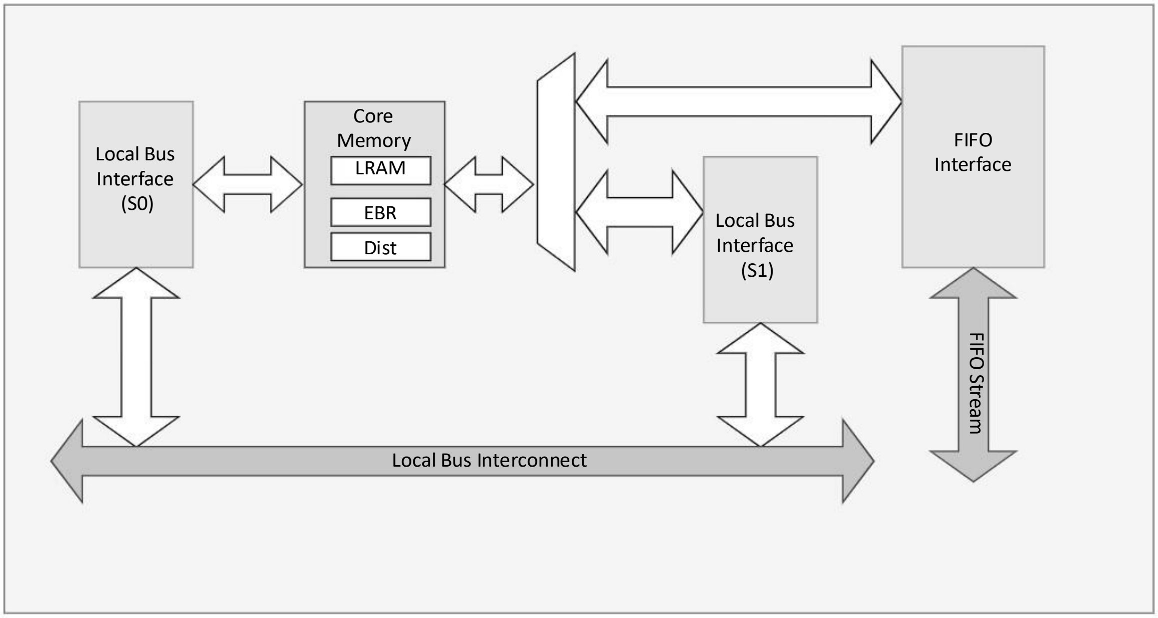

Figure 2.1. Generic TCM IP Block Diagram

2.1.1. Memory Implementation

The TCM Memory Module uses Embedded Block RAMs (EBR) or Distributed Memory in the Lattice Avant family devices. As for CrossLink-NX and Certus-NX, CertusPro-NX, and MachXO5-NX family devices, Large RAMs can also be used. The memory implementation can be configured as true dual-port, single-port, or read-only memory. The number of ports and read/write configuration of the TCM IP Module automatically select the best type of memory for the user-selected application.

Table 2.1. System Core Memory Implementation

Memory Type	Configuration Used[1]	CrossLink-NX,Certus-NX, CertusPro-NX, and MachXO5-NX	Lattice Avant
LRAM	1 port : "W/R" 2 ports: "W/R" + "W/R"	Yes	No
EBR – ram_dp_true	2 ports: "W/R" + "W/R"	Yes	Yes
EBR – ram_sp	1 port : "W/R"	Yes	Yes
Distributed Memory	1 port : "W/R"	Yes	Yes

Note:

1. Byte-Enable cannot be used in conjunction with ECC. LRAM is not present in Lattice Avant family devices.

Table 2.2. Features Supported per Memory Block

Feature	LRAM	EBR	Distributed Memory
Memory Initialization	Yes	Yes	No
Registered Output	Yes	Yes	Yes
Dual-Port Configuration	Yes	Yes	No
Byte-Enable	Yes	Yes	No

2.2. Functional Overview

2.2.1. Local Bus Interface

The TCM Memory Module is designed to be fully compatible with the local bus interface. It can be configured as single-port or dual-port interfaces, depending on if single-port or dual-port memory is needed along with RAM or ROM configuration.

2.2.2. FIFO Interface

There is a dedicated FIFO interface shared with the port S1. This interface is used to inject FIFO data from a FIFO stream. This is typically used to upload firmware values to the core memory. The FIFO starts writing at the designated first byte-addressable data and writes up to the maximum depth of the implemented memory.

Table 2.3. Allowable Combination of Features for TCM

Device	Byte-Enable	ECC	Unaligned Access	Maximum Supported Port Count
LRAM	×	×	×	2
	×	×	×	2
	×	×	✓	2
	×	×	✓	2
	×	✓	×	2
	×	✓	×	Not supported
	×	✓	✓	2
	×	✓	✓	Not supported
	✓	×	×	2
	✓	×	×	2
	✓	×	✓	2
	✓	×	✓	2
	✓	✓	×	Not supported
	✓	✓	×	Not supported
	✓	✓	✓	Not supported
	✓	✓	✓	Not supported
EBR	×	×	N/A	2
	×	×	N/A	2
	×	✓	N/A	2[1]
	×	✓	N/A	Not supported
	✓	×	N/A	2
	✓	×	N/A	2
	✓	✓	N/A	Not supported
	✓	✓	N/A	Not supported

Note:

1. EBR ECC is limited to Single-Port “W/R” or Dual-Port “W/O” + “R/O” combinations only.

2.3. Attribute Summary

Table 2.4. TCM IP Attribute Summary

Parameter Name	Values	Default	Description
ADDR_DEPTH	1–128K	16K	Address depth of the memory in units of DATA_WIDTH. Maximum ADDR_DEPTH value depends on DATA_WIDTH. DATA_WIDTH ADDR_DEPTH 8 1–128K 16 1–64K 32 1–32K
DATA_WIDTH	8, 16, 32	32	Data width of the memory (bits)
MEMORY_TYPE	"EBR" or "Distributed" or "LRAM"	"EBR"	Parameter that selects the type of memory implemented for this instance of TCM IP.
PORT_COUNT	1 or 2	1	Determines if the TCM IP uses a single port or dual ports.
ECC_ENABLE	"True", "False"	"False"	Determines if ECC is used (applies to both ports).
ENABLE FIFO STREAMER	"True", "False"	"False"	Enables the FIFO Streamer interface. If this is used with *PORT_COUNT = 2*, the Streamer interface is muxed with port S1. Otherwise, it has its own dedicated port.
FIFO CLOCK BYPASS	"True", "False"	"False"	When enabled, the FIFO streamer uses the sys_clk as clock source. Otherwise, it has its own dedicated clock. It is editable only when *ENABLE FIFO STREAMER* = '*True*'.
FIFO START ADDRESS	0 to ADDR_DEPTH-1	0	The start address where the FIFO streamer starts to write. Editable only when *ENABLE FIFO STREAMER* = '*True*'.
REGMODE_S0	"True", "False"	"False"	Applies a registered output for S0.
REGMODE_S1[1]	"True", "False"	"False"	Applies a registered output for S1.
RESET_MODE_S0	"Async" or "Sync"	"Async"	Reset mode for Port S0.
RESET_MODE_S1[2]	"Async" or "Sync"	"Async"	Reset mode for Port S1.
BYTE_ENABLE_S0	True, False	True	Determines if Byte Enable for S0 is used.
BYTE_ENABLE_S1[2,3,5]	True, False	True	Determines if Byte Enable for S1 is used.
S0_START_ADDR[2]	0 to ADDR_DEPTH-1	0	Starting memory address offset for port S0.
S1_START_ADDR[2]	0 to ADDR_DEPTH-1	0	Starting memory address offset for port S1.
S0_END_ADDR[2]	0 to ADDR_DEPTH-1	ADDR_DEPTH-1	Ending memory address offset for port S0. It must be greater than *S0_START_ADDR*.
S1_END_ADDR[2]	0 to ADDR_DEPTH-1	ADDR_DEPTH-1	Ending memory address offset for port S1. It must be greater than *S1_START_ADDR*.
ACCESS_TYPE_S0	"R/W", "W/O" or "R/O"	"R/W"	Determines the access for S0.
ACCESS_TYPE_S1[2]	"R/W", "W/O" or "R/O"	"R/W"	Determines the access for S1.
SHIFT_DIRECTION[4]	"right", "left"	"right"	Shift direction of the unaligned read access.
INIT_FILE	<string>	"None"	Selects the initialization file for the TCM IP.
INIT_FILE_FORMAT	"Hex" or "Binary"	"Binary"	Determines the file format of the initialization file.

Notes:

1. Applies only to LRAM and EBR. Dual-port EBR ECC is limited to "W/O" + "R/O" for *ACCESS_TYPE_S0* and *ACCESS_TYPE_S1* respectively.
2. Editable only when *PORT_COUNT = 2*.
3. Byte-Enable is not available when ECC is enabled or when *DATA_WIDTH = 8*.
4. Editable only if *ENABLE_UNALIGNED_ACCESS* is enabled.
5. Byte-Enable and Unaligned Access use the lower address bits as control signals.

2.3.1. START_ADDR and END_ADDR Description and Implementation

These parameters are used to limit the accessible memory address of each port. An offset is generated to apply the effects of START_ADDR and END_ADDR. For example, if the START_ADDR is equal to 0x400, passing a value of 0x10 points to the memory access location 0x410. There is no minimum difference between START_ADDR and END_ADDR.

However, since the TCM specification requires a minimum of one kB address space, an out of bounds error (refer to Table 2.5) is generated if the address goes beyond the specified END_ADDR limit. For example, START_ADDR = 0x400 and END_ADDR = 0x500. The minimum AHB-L address space is one kB, so an address between 0x000 to 0x3FF can be passed. If 0x101 is send to the port, the 0x400 offset is applied totaling to a value 0x501. Since the final address is beyond the END_ADDR, an out of bound error is generated.

2.4. Signal Descriptions

Table 2.5. TCM IP Ports

Pin Name	Direction	Width(bits)	Description
SYS_CLK_I	IN	1	Clock source
SYS_RST_I	IN	1	Active low reset signal
Port 0			
IBUS_CMD_VALID_I	INPUT	1	Indicates that the master is signaling valid write/read address and control information.
IBUS_CMD_READY_O	OUTPUT	1	Indicates that the slave is ready to accept an address and associated control signals.
IBUS_CMD_PAYLOAD_WR_I	INPUT	1	Indicates the transfer direction. When HIGH, this signal indicates a write transfer. When LOW, it indicates a read transfer. It has the same timing as that of the address signals. However, it does not remain constant throughout a burst transfer.
IBUS_CMD_PAYLOAD_UNCACHED_I	INPUT	1	Indicates if the transfer is a cache operation. When HIGH, this signal indicates a cache transfer. When LOW, it indicates an uncached transfer.
IBUS_CMD_PAYLOAD_ADDRESS_I	INPUT	32	Address
IBUS_CMD_PAYLOAD_DATA_I	INPUT	DATA_WIDTH_A	The write data bus transfers data from the master to the slaves during write operations.
IBUS_CMD_PAYLOAD_MASK_I	INPUT	DATA_WIDTH_A/8	Indicates the byte lanes that hold valid data. There is one write mask bit for every eight bits of the write data bus.
IBUS_CMD_PAYLOAD_SIZE_I	INPUT	3	Indicates the size of the transfer, which is typically byte, half-word, or word. 3'b101 indicates an 8-words burst transfer.
IBUS_CMD_PAYLOAD_LAST_I	INPUT	1	Indicates the last transfer in a write burst.
IBUS_RSP_VALID_O	OUTPUT	1	The slave is signaling the required read data.
IBUS_RSP_PAYLOAD_LAST_O	OUTPUT	1	Indicates the last transfer in a read burst.
IBUS_RSP_PAYLOAD_DATA_O	OUTPUT	DATA_WIDTH_A	Read data.
IBUS_RSP_PAYLOAD_ERROR_O	OUTPUT	1	Indicates the status of the read transfer. When HIGH, this signal indicates a successful transfer. When LOW, it indicates a failed transfer.
Port 1			
DBUS_CMD_VALID_I	INPUT	1	Indicates that the master is signaling valid write/read address and control information.
DBUS_CMD_READY_O	OUTPUT	1	Indicates that the slave is ready to accept an address and associated control signals.
DBUS_CMD_PAYLOAD_WR_I	INPUT	1	Indicates the transfer direction. When HIGH, this signal indicates a write transfer. When LOW, it indicates a read transfer. It has the same timing as that of the

Pin Name	Direction	Width(bits)	Description
			address signals. However, it does not remain constant throughout a burst transfer.
DBUS_CMD_PAYLOAD_UNCACHED_I	INPUT	1	Indicates if the transfer is a cache operation. When HIGH, this signal indicates a cache transfer. When LOW, it indicates an uncached transfer.
DBUS_CMD_PAYLOAD_ADDRESS_I	INPUT	32	Address
DBUS_CMD_PAYLOAD_DATA_I	INPUT	DATA_WIDTH_B	The write data bus transfers data from the master to the slaves during write operations.
DBUS_CMD_PAYLOAD_MASK_I	INPUT	DATA_WIDTH_B/8	Indicates that the byte lanes that hold valid data. There is one write mask bit for every eight bits of the write data bus.
DBUS_CMD_PAYLOAD_SIZE_I	INPUT	3	Indicates the size of the transfer, which is typically byte, halfword, or word. 3'b101 indicates an 8-words burst transfer.
DBUS_CMD_PAYLOAD_LAST_I	INPUT	1	Indicates the last transfer in a write burst.
DBUS_RSP_VALID_O	OUTPUT	1	The slave signals the required read data.
DBUS_RSP_PAYLOAD_LAST_O	OUTPUT	1	Indicates the last transfer in a read burst.
DBUS_RSP_PAYLOAD_DATA_O	OUTPUT	DATA_WIDTH_B	Read data.
DBUS_RSP_PAYLOAD_ERROR_O	OUTPUT	1	Indicates the status of the read transfer. When HIGH, this signal indicates a successful transfer. When LOW, it indicates a failed transfer.
FIFO Streamer			
FIFO_CLK_I	INPUT	1	FIFO clock source
FIFO_WR_EN_I	INPUT	1	FIFO write enable
FIFO_WR_DATA_I	INPUT	8	FIFO write data
FIFO_INTERFACE_EN_I	INPUT	1	FIFO enable
FIFO_ADDRESS_RST_I	INPUT	1	FIFO address reset
FIFO_FULL_O	OUTPUT	1	FIFO full status output flag

2.4.1. System Memory Timing Diagrams

Figure 2.2 to Figure 2.4 show the timing diagrams for some configurations of the system memory. Slave Port S0 is given as an example, but Port 1 should have similar timing diagrams. The FIFO Interface Diagram is shown in Figure 2.5.

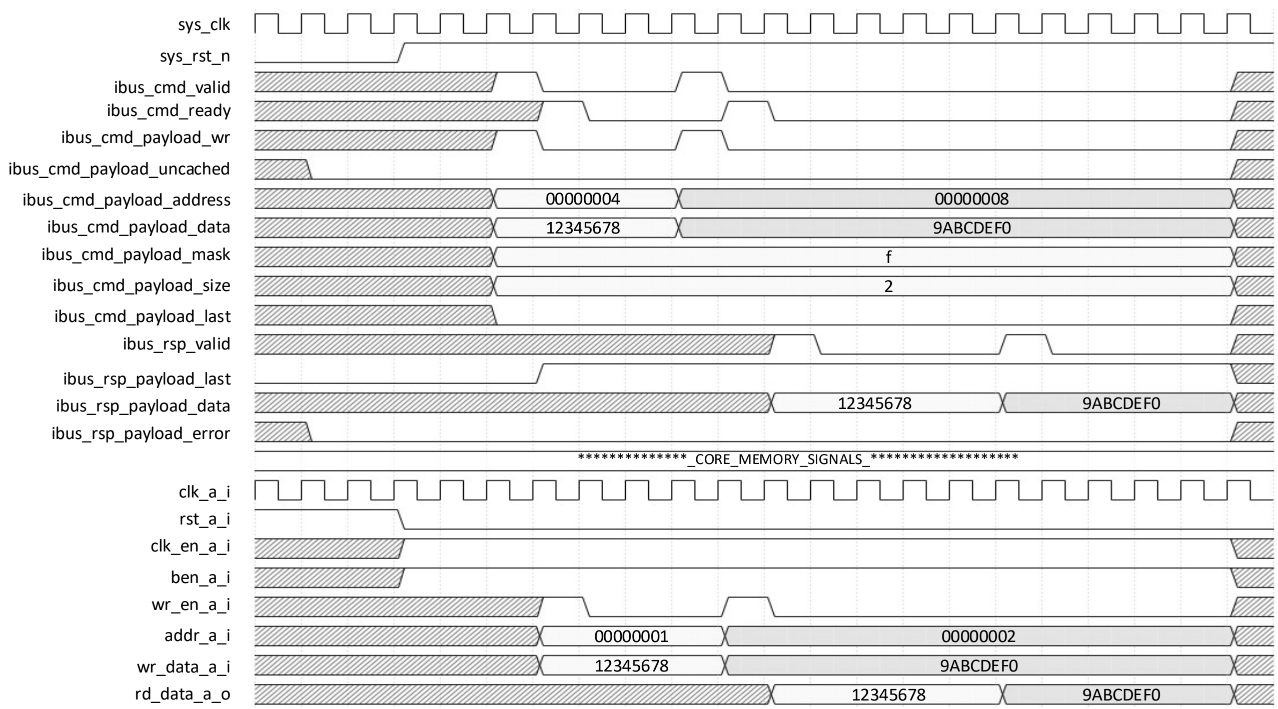

Figure 2.2. Write-Read Operation

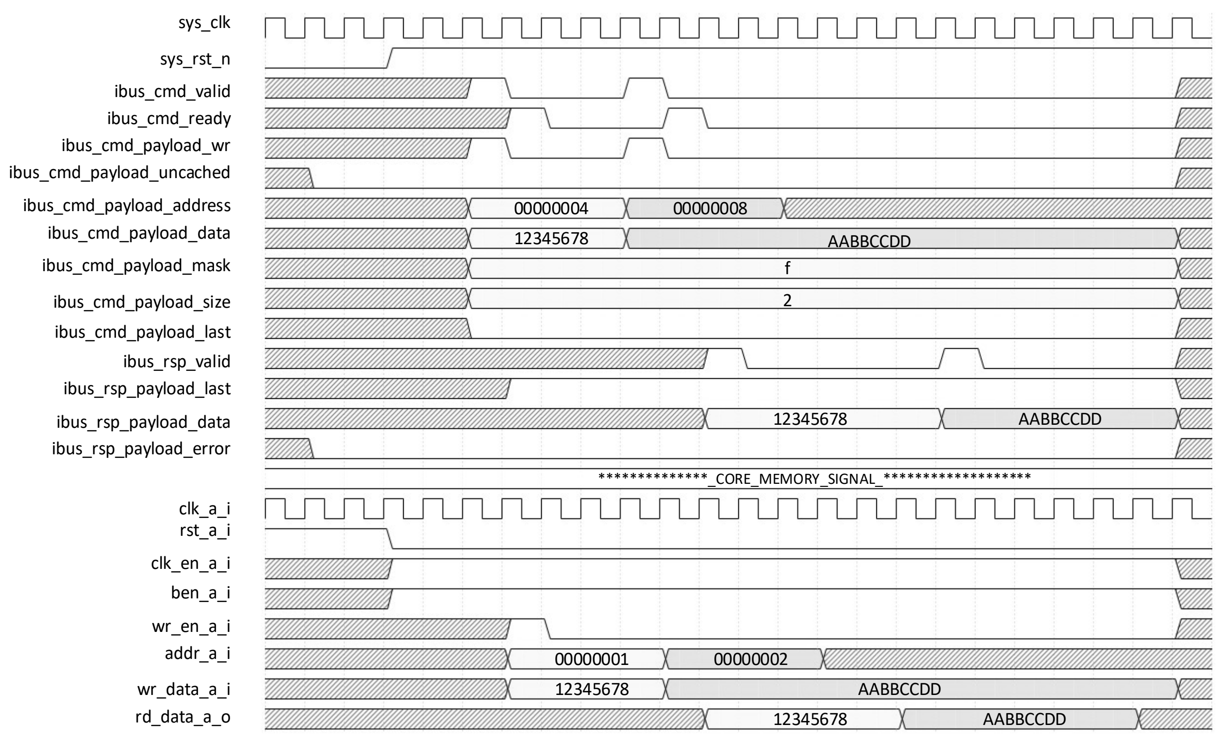

Figure 2.3. Sequential Write/Read

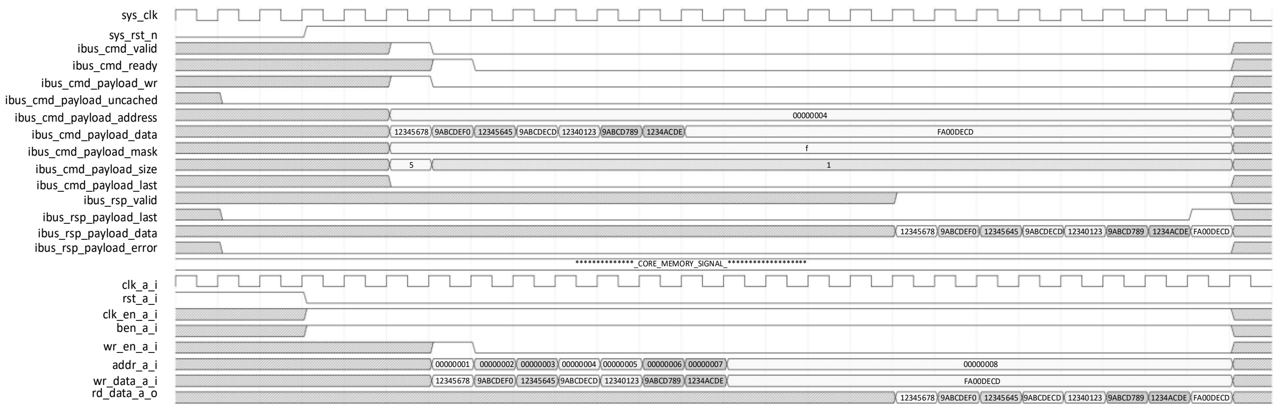

Figure 2.4. Bulk Write-Bulk Read Operation

Figure 2.5 shows the FIFO Interface Timing Diagram. When using the FIFO, you can safely extend the fifo_wr_en_i and fifi_wr_data_i signals even when the fifo_full_o signal has been asserted; but those signals are ignored. Further write transactions are also ignored when fifo_full_o is asserted.

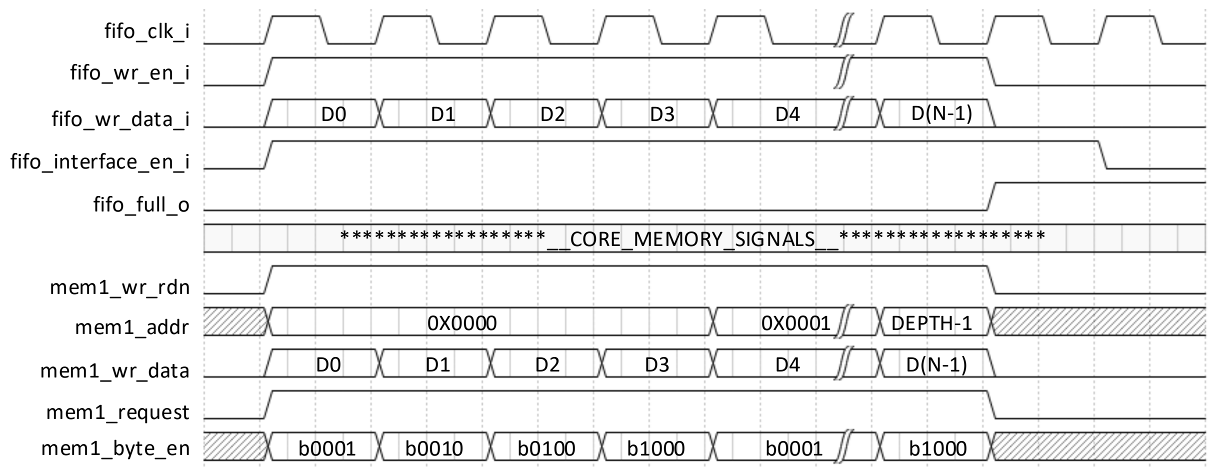

Figure 2.5. FIFO Interface Timing Diagram

2.5. User Accessible Parameters

- **Supported families**
  - Lattice Avant
  - MachXO5-NX
  - CrossLink-NX
  - CertusPro-NX
  - Certus-NX
- **Supported platforms**
  - Lattice Nexus
  - Lattice Avant
- **Supported bus interfaces**
  - Single-Port Memory
  - Dual-Port Memory
- **Supported primitives**
  - EBR

- LRAM
- Distributed RAM

2.6. Unaligned Access Feature

The Large RAM Module supports RISC-V compressed instruction chunks of data and shifts them. If RISC-V needs to read the upper 16 bit of data in some address, it is very helpful to add support for shifting the upper 16 bits of output into the lower 16 bits, padding the upper bits with 0. Unaligned Read enables this feature. Table 2.6 shows possible combinations for Unaligned Read shifting. The unaligned read pins are shared with the ben_i ports for Single-Port Large RAM. Given this, you should be careful in changing the value of these signals. The port functions as Byte-Enable during write-access and Unaligned Read during read-access.

Table 2.6.Unaligned Read shift function (Single-Port LRAM)

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

2.7. Initialization Format

The initialization file is an ASCII file, which the designer can create or edit using any ASCII editor. The Module/IP Block Wizard supports the following memory file formats:

- Binary File
- Hex File

The file name for the memory initialization file is *.mem (<file_name>.mem). Each row includes the value to be stored in a particular memory location. The number of characters (or the number of columns) represents the number of bits for each address (or the width of the memory module).

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

The initialization file is primarily used when the TCM IP is configured as a ROM. In RAM configuration, the initialization file can also be used to preload memory contents.

Binary File

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

Memory Size 20 x 32

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

Hex File

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

Memory Size 8 x 16

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

References

- Avant-E web page
- MachXO5-NX web page
- CrossLink-NX web page
- CertusPro-NX web page
- Certus-NX web page

Technical Support Assistance

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

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

Revision History

Revision 1.1, July 2023

Section	Change Summary
Introduction	In Table 1.1. FPGA Software for IP Configuration, Generation, and Implementation, changed the IP Implementation tool from *Lattice Diamond software* to *Lattice Radiant software* for MachXO5-NX, CrossLink-NX, CertusPro-NX and Certus-NX families.

Functional Descriptions	• In Table 2.4. TCM IP Attribute Summary:
	  • changed the Values of S0_START_ADDR, S1_START_ADDR, S0_END_ADDR, and S1_END_ADDR from *0 to ADDR_DEPTH* to *0 to ADDR_DEPTH-1*;
	  • changed the Default of S0_END_ADDR and S1_END_ADDR from *ADDR_DEPTH* to *ADDR_DEPTH-1*.
	• In Table 2.5. TCM IP Ports:
	  • changed the Width (bits) of IBUS_CMD_PAYLOAD_ADDRESS_I from *ADDR_DEPTH_A* to *32*;
	  • changed the Width (bits) of DBUS_CMD_PAYLOAD_ADDRESS_I from *ADDR_DEPTH_B* to *32*;
	  • changed the Width (bits) of IBUS_RSP_PAYLOAD_DATA_O from *32* to *DATA_WIDTH_A*;
	  • changed the Width (bits) of DBUS_RSP_PAYLOAD_DATA_O from *32* to *DATA_WIDTH_B*.

References	Updated this section.

Revision 1.0, June 2023

Section	Change Summary
All	Production release.

www.latticesemi.com