HyperRAM Memory Controller

Reference Design

FPGA-RD-02236-1.0

August 2021

Disclaimers

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

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
EBR	Embedded Block RAM
FPGA	Field-Programmable Gate Array
I/O	Input/Output
IP	Intellectual Property
VVML	Voice and Vision Machine Learning

1. Introduction

The HyperRAM controller is designed to enable the CrossLink™-NX device to interface with HyperRAM devices that uses the HyperBus interface. It provides you with two separate standard interfaces (AXI-4) for sending write and read commands to the HyperRAM devices, as well as for accessing the Control Registers of the design.

2. Features and Limitations

The following specifications are the features and limitations of the HyperRAM Controller:

- Supports First-Generation HyperRAM specifications
- Supports up to two HyperRAM devices
- Uses two separate AXI-4 buses for Memory Control and Register Control
- Supports Nexus™ Platform devices*
- Supports Lattice Radiant™ version 2.2 only
- HyperBus clock speed is up to 162 MHz only
- Supports Single HyperBus Clock only

*__Note:__ Reference design is tested on CrossLink-NX device only.

3. Functional Block Diagram

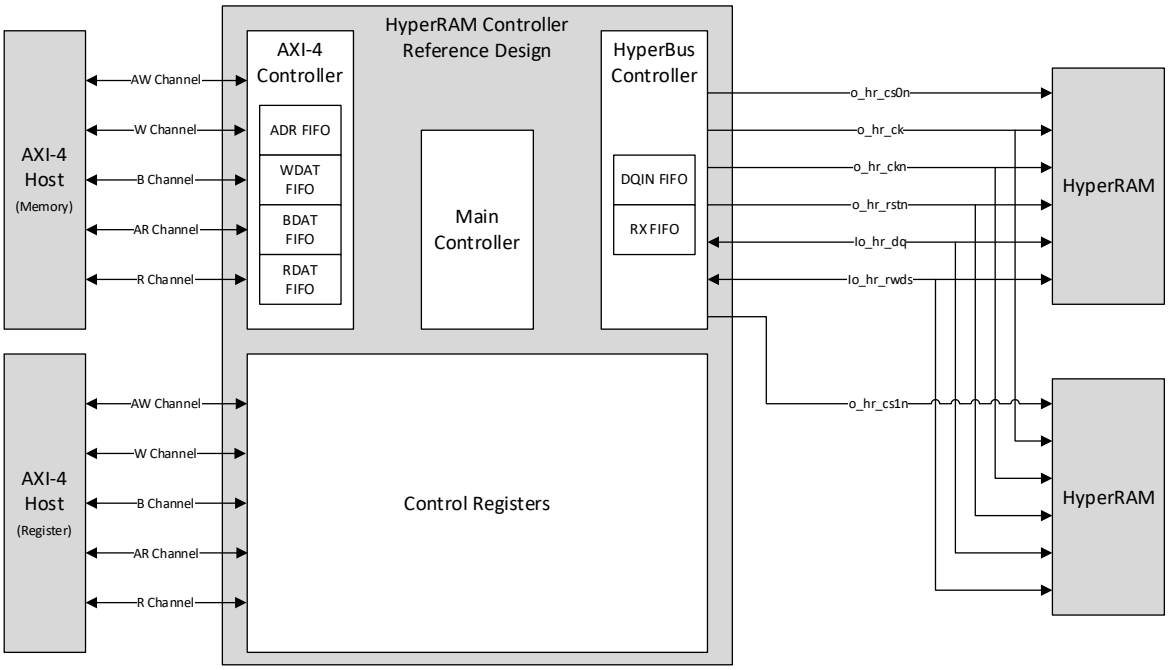

Figure 3.1. Block Diagram

3.1. Control Registers

Table 3.1 shows the summary of the register file that can be accessed through the AXI-4 Register Bus. Table 3.2 describes the notation used for the bit attributes for each register.

Table 3.1. Register File

Address	Register Name
0x00	Controller Status Register (CSR)
0x10	CS0 Memory Base Address Register (MBAR0)
0x14	CS1 Memory Base Address Register (MBAR1)
0x20	CS0 Memory Configuration Register (MCR0)
0x24	CS1 Memory Configuration Register (MCR1)
0x30	CS0 Memory Timing Register (MTR0)
0x34	CS1 Memory Timing Register (MTR1)

Table 3.2. Bit Attributes Notation

Attribute	Notation
Reading behavior specifier	R0: Always read '0'.
	R1: Always read '1'.
	RN: Never read. Reading always results to bus error response.
	R: Readable.
Writing behavior specifier	WX: Writing has no effect.
	W0: Must write '0' when writing. Writing '1' results unknown behavior.
	W1: Must write '1' when writing. Writing '0' results unknown behavior.
	W0C: Writing '0' clears the bit to '0'. Writing '1' has no effect.
	W1C: Writing '1' clears the bit to '0'. Writing '0' has no effect.
	W1S: Writing '1' sets the bit to '1'. Writing '0' has no effect.
	WN: Never write. Writing always results to bus error response.
	W: Writable.
Relation specifier	/: Reading always returns written value.
	/+: Similar to "/". However, the bit can be also set or cleared via dedicated set/clear registers or software reset bit.
	,: The read value is different from the written value.

3.1.1. Controller Status Register (CSR)

Table 3.3. CSR Bit Summary

Bit Index	Bit Name	Bit Attribute	Initial Value	Description
31:27	RFU	R0,WX	0	Reserved for Future Use
26	WRSTOERR	R,WX	0	RSTO Error in Write Transaction: This bit indicates whether HyperBus memory is under reset state in the latest write transaction. When this bit is set, AXI SLVERR occurs. [0] = Normal Operation [1] = HyperBus memory under reset
25	WTRSERR	R,WX	0	Transaction Error in Write Transaction: This bit indicates whether AXI protocol is acceptable by HyperBus CTRL IP in the latest write transaction. When this bit is set, AXI SLVERR occurs. [0] = Normal Operation [1] = This protocol is not supported
24	WDECERR	R,WX	0	Transaction Error in Write Transaction: This bit indicates whether access address is acceptable in the latest write transaction. When this bit is set, AXI DECERR occurs. [0] = Normal Operation [1] = Access address is not reachable
23:17	RFU	R0,WX	0	Reserved for Future Use
16	WACT	R,WX	0	Write Transaction Active: This bit indicates whether write transaction is in progress. [0] = Write transaction is idle [1] = Write transaction is active
15:12	RFU	R0,WX	0	Reserved for Future Use

Bit Index	Bit Name	Bit Attribute	Initial Value	Description
11	RDSSTALL	R,WX	0	RDS Stall Error in Read Transaction: This bit indicates whether read data transfer from HyperBus memory is stalled (RDS remains LOW) in the latest read transaction. When this bit is set, AXI SLVERR occurs. [0] = Normal Operation [1] = RDS is stalled
10	RRSTOERR	R,WX	0	RSTO Error in Read Transaction: This bit indicates whether HyperBus memory is under reset state in the latest read transaction. When this bit is set, AXI SLVERR occurs. [0] = Normal Operation [1] = HyperBus memory under reset
9	RTRSERR	R,WX	0	Transaction Error in Read Transaction: This bit indicates whether AXI protocol is acceptable by HyperBus CTRL IP in the latest read transaction. When this bit is set, AXI SLVERR occurs. [0] = Normal Operation [1] = This protocol is not supported
8	RDECERR	R,WX	0	Transaction Error in Read Transaction: This bit indicates whether access address is acceptable in the latest read transaction. When this bit is set, AXI DECERR occurs. [0] = Normal Operation [1] = Access address is not reachable
7:1	RFU	R0,WX	0	Reserved for Future Use
0	RACT	R,WX	0	Read Transaction Active: This bit indicates whether read transaction is in progress. [0] = Read transaction is idle [1] = Read transaction is active

3.1.2. Memory Base Address Register (MBAR0 and MBAR1)

Table 3.4. MBAR Bit Summary

Bit Index	Bit Name	Bit Attribute	Initial Value	Description
31:24	A[31:24]	R,W	0	Offset address to differentiate CS0 and CS1
23:0	A[23:0]	R0,WX	0	Fixed to 0 since up to 16M bytes can be addressed for each HyperRAM device

Note: MBAR1 must be greater than MBAR0 to enable support of two HyperRAM devices.

3.1.3. Memory Configuration Register (MCR0 and MCR1)

Table 3.5. MCR Bit Summary

Bit Index	Bit Name	Bit Attribute	Initial Value	Description
31	MAXEN	R/W	0	Maximum Length Enable: This bit enables maximum access data length control for restricting HyperBus CS assertion time by MAXLEN bit. [0] = Disable maximum access data length control [1] = Enable maximum access data length control
30:27	RFU	R0,WX	0	Reserved for Future Use
26:18	MAXLEN[8:0]	R/W	000h	Maximum Length: These bits are set as maximum access data length in order to restrict HyperBus CS assertion time. This is ignored when MAXEN bit is 0. [000h] = 2 Byte (1 CK) [001h] = 4 Byte (2 CK) [002h] = 8 Byte (3 CK) : : [1FFh] = 1024 Byte (512 CK)
17	TCMO	R,WX	0	True Continuous Merging Option: This bit is set when the wrap transaction and subsequent continuous transaction can be merged. Note that this function can be used with the HyperFlash memory with specific function. Please confirm whether it is corresponding HyperFlash memory before enabling this function. [0] = No merging WRAP and INCR [1] = Merging WRAP and INCR
16	ACS	R,WX	0	Asymmetry Cache System Support: This bit is set when the different wrap size (cache size) is required by multi core in system. This function should be disabled if the HyperBus memory itself supports the asymmetry cache system. [0] = No asymmetry cache system support [1] = Asymmetry cache system support
15:6	RFU	R0,WX	0	Reserved for Future Use
5	CRT	R/W	0	Asymmetry Cache System Support: This bit indicates whether access is to memory space or register space. This bit is mapped to CA [46] bit in command/address cycle to HyperRAM device. When using HyperFlash memory, this bit should be set to 0. [0] = Memory Space [1] = Configuration Register Space
4	DEVTYPE	R/W	0	Device Type: This bit is set as a device type of connected memory. [0] = HyperFlash [1] = HyperRAM
3:2	RFU	R0,WX	0	Reserved for Future Use

Bit Index	Bit Name	Bit Attribute	Initial Value	Description
1:0	WRAPSIZE[1:0]	R/W	11b	Wrapped Burst Size: This bit is set as wrap burst length which was set to HyperBus memory. This bit is ignored when the ACS bit is 0. When the ACS bit is 1, this bit should be set the same as wrap size of configuration register in HyperBus memory. [00] = Reserved [01] = 64 Bytes [10] = 16 Bytes [11] = 32 Bytes

3.1.4. Memory Timing Register (MTR0 and MTR1)

Table 3.6. MCR Bit Summary

Bit Index	Bit Name	Bit Attribute	Initial Value	Description
31:28	RCSHI[3:0]	R/W	0h	Read Chip Select High Between Operations: Before the read access, this bit inserts the CK cycles to the chip select high period. [0h] = 1.5 CK [1h] = 2.5 CK [2h] = 3.5 CK : : [Fh] = 16.5 CK
27:24	WCSHI[3:0]	R/W	0h	Write Chip Select High Between Operations: Before the write access, this bit inserts the CK cycles to the chip select high period. [0h] = 1.5 CK [1h] = 2.5 CK [2h] = 3.5 CK : : [Fh] = 16.5 CK
23:20	RCSS[3:0]	R/W	0h	Read Chip Select Setup to next CK Rising Edge: In the read access, this bit inserts the CK cycles between the falling edge of chip select and the rising edge of first CK. [0h] = 1 CK [1h] = 2 CK [2h] = 3 CK : : [Fh] = 16 CK
19:16	WCSS[3:0]	R/W	0h	Write Chip Select Setup to next CK Rising Edge: In the write access, this bit inserts the CK cycles between the falling edge of chip select and the rising edge of first CK. [0h] = 1 CK [1h] = 2 CK [2h] = 3 CK : : [Fh] = 16 CK

Bit Index	Bit Name	Bit Attribute	Initial Value	Description
15:12	RCSH[3:0]	R/W	0h	Read Chip Select Setup to next CK Falling Edge: In the read access, this bit inserts the CK cycles between the falling edge of last CK and the rising edge of chip select. [0h] = 1 CK [1h] = 2 CK [2h] = 3 CK : : [Fh] = 16 CK
11:8	WCSH[3:0]	R/W	0h	Write Chip Select Setup to next CK Falling Edge: In the write access, this bit inserts the CK cycles between the falling edge of last CK and the rising edge of chip select. [0h] = 1 CK [1h] = 2 CK [2h] = 3 CK : : [Fh] = 16 CK
7:4	RFU	R0,WX	0	Reserved for Future Use
3:0	LATENCY[3:0]	R/W	0	Latency Cycle for HyperRAM: When using HyperRAM memory, this bit should be set to the same value as the read latency in configuration register of HyperRAM memory. This bit is ignored when the DEVTYPE in HYPERBUS_CTRL_MCRi register is chosen to the HyperFlash memory. [0h] = 5 CK Latency [1h] = 6 CK Latency [2h] = Reserved : : [Eh] = Reserved [Fh] = 4 CK Latency

4. Pin Descriptions

Table 4.1. Pin Descriptions

Signal	Width	Type	Description
Clocks and Reset			
i_clk0	1	Input	System clock. $f_{i\_clk0\_MAX}$ = 162 MHz.
i_clk270	1	Input	System clock, shifted by 270 degrees with respect to i_clk0. $f_{i\_clk270} = f_{i\_clk0}$.
i_clk_axi	1	Input	AXI-4 clock reference. $f_{i\_clk\_axi} \geq f_{i\_clk0}$.
i_resetn	1	Input	Active low system reset.
AXI-4 Signals: Write Address (AW) Channel			
i_axi*_awid	4	Input	Write Address ID.
i_axi*_awaddr	32	Input	Write Address. This is the starting address where data will be written to the HyperRAM.
i_axi*_awlen	8	Input	Number of transfers per burst/transaction. Total bytes to be written is equal to (i_axi_awlen + 1) × $2^{i\_axi\_awsize}$. *Not applicable for Register Control*.
i_axi*_awsize	3	Input	Number of bytes per transfer. *Not applicable for Register Control*.
i_axi*_awvalid	1	Input	This signal indicates that the data on the AW bus are valid.
o_axi*_awready	1	Output	This signal indicates that the HyperRAM Controller is ready to accept data on the AW channel.
AXI-4 Signals: Write Data (W) Channel			
i_axi*_wdata	64 or 32	Input	Write data. 64 bits for Memory Control and 32 bits for Register Control.
i_axi*_wlast	1	Input	Active High. Indicates that the data on i_axi_wdata is the last for that burst/transaction.
I_axi*_wstrb	8 or 4		Byte strobe. 8 bits for Memory Control and 4 bits for Register Control.
i_axi*_wvalid	1	Input	Active High. Indicates that the data on i_axi_wdata is valid.
o_axi*_wready	1	Output	This signal indicates that the HyperRAM Controller is ready to accept data on the W channel.
AXI-4 Signals: Write Response (B) Channel			
o_axi*_bid	4	Output	Write Response ID. This signal should match the data on i_axi_awid when the i_axi_awvalid was asserted.
o_axi*_bresp	2	Output	Write Response. [00] = OKAY, the success of a normal access [01] = EXOKAY, not applicable for this design [10] = SLVERR, an unsuccessful transaction [11] = DECERR, the interconnect cannot successfully decode a slave access
o_axi*_bvalid	1	Output	Indicates that the data on o_axi_bresp and o_axi_bid are valid.
i_axi*_bready	1	Input	Response Ready signal. This indicates that the master can accept a write response.
AXI-4 Signals: Read Address (AR) Channel			
i_axi*_arid	4	Input	Read Address ID.
i_axi*_araddr	32	Input	Read Address. This is the starting address where data will be read from the HyperRAM.
i_axi*_arlen	8	Input	Number of transfers per burst/transaction. Total bytes to be read is equal to (i_axi_arlen + 1) × 64. *Not applicable for Register Control*.
i_axi*_arvalid	1	Input	This signal indicates that the data on the AR bus are valid.
o_axi*_arready	1	Output	This signal indicates that the HyperRAM Controller is ready to accept data on the AR channel.

Signal	Width	Type	Description
AXI-4 Signals: Read Data (R) Channel			
o_axi*_rdata	64 or 32	Output	Read data. 64 bits for Memory Control and 32 bits for Register Control.
o_axi*_rlast	1	Output	Indicates that the data on o_axi_rdata is the last for that burst/transaction when at 1.
o_axi*_rid	4	Output	Read Response ID. This signal should match the data on i_axi_arid when the i_axi_arvalid was asserted.
o_axi*_rresp	2	Output	Read Response. [00] = OKAY, the success of a normal access [01] = EXOKAY, not applicable for this design [10] = SLVERR, an unsuccessful transaction [11] = DECERR, the interconnect cannot successfully decode a slave access
o_axi*_rvalid	1	Output	Indicates that the data on o_axi_rdata, o_axi_rresp and o_axi_rid are valid.
i_axi*_rready	1	Input	Response Ready signal. This indicates that the master can accept a read response.
Register Control Signals: Write Address (AW) Channel			
i_axir_awid	4	Input	Write Address ID.
i_axir_awaddr	32	Input	Write Address. This is the starting address where data will be written to the HyperRAM.
i_axi*_awvalid	1	Input	This signal indicates that the data on the AW bus are valid.
o_axi*_awready	1	Output	This signal indicates that the HyperRAM Controller is ready to accept data on the AW channel.
Register Control Signals: Write Data (W) Channel			
i_axi*_wdata	64 or 32	Input	Write data. 64 bits for Memory Control and 32 bits for Register Control.
i_axi*_wlast	1	Input	Active High. Indicates that the data on i_axi_wdata is the last for that burst/transaction.
I_axi*_wstrb	8 or 4		Byte strobe. 8 bits for Memory Control and 4 bits for Register Control.
i_axi*_wvalid	1	Input	Active High. Indicates that the data on i_axi_wdata is valid.
o_axi*_wready	1	Output	This signal indicates that the HyperRAM Controller is ready to accept data on the W channel.
Register Control Signals: Write Response (B) Channel			
o_axi*_bid	4	Output	Write Response ID. This signal should match the data on i_axi_awid when the i_axi_awvalid was asserted.
o_axi*_bresp	2	Output	Write Response. [00] = OKAY, the success of a normal access [01] = EXOKAY, not applicable for this design [10] = SLVERR, an unsuccessful transaction [11] = DECERR, the interconnect cannot successfully decode a slave access
o_axi*_bvalid	1	Output	Indicates that the data on o_axi_bresp and o_axi_bid are valid.
i_axi*_bready	1	Input	Response Ready signal. This indicates that the master can accept a write response.
Register Control Signals: Read Address (AR) Channel			
i_axi*_arid	4	Input	Read Address ID.
i_axi*_araddr	32	Input	Read Address. This is the starting address where data will be read from the HyperRAM.
i_axi*_arlen	8	Input	Number of transfers per burst/transaction. Total bytes to be read is equal to (i_axi_arlen + 1) × 64. *Not applicable for Register Control*.
i_axi*_arvalid	1	Input	This signal indicates that the data on the AR bus are valid.
o_axi*_arready	1	Output	This signal indicates that the HyperRAM Controller is ready to accept data on the AR channel.

Signal	Width	Type	Description
Register Control Signals: Read Data (R) Channel			
o_axi*_rdata	64 or 32	Output	Read data. 64 bits for Memory Control and 32 bits for Register Control.
o_axi*_rlast	1	Output	Indicates that the data on o_axi_rdata is the last for that burst/transaction when at 1.
o_axi*_rid	4	Output	Read Response ID. This signal should match the data on i_axi_arid when the i_axi_arvalid was asserted.
o_axi*_rresp	2	Output	Read Response. [00] = OKAY, the success of a normal access [01] = EXOKAY, not applicable for this design [10] = SLVERR, an unsuccessful transaction [11] = DECERR, the interconnect cannot successfully decode a slave access
o_axi*_rvalid	1	Output	Indicates that the data on o_axi_rdata, o_axi_rresp and o_axi_rid are valid.
i_axi*_rready	1	Input	Response Ready signal. This indicates that the master can accept a read response.
HyperBus Interface[1]			
o_hr_ck[2]	1	Output	Clock_Pos[2]
o_hr_ckn[2]	1	Output	Clock_Neg[2]
o_hr_cs0n	1	Output	Chip Select 0. Active Low
o_hr_cs1n	1	Output	Chip Select 1. Active Low
o_hr_rstn	1	Output	Hardware Reset. Active Low
io_hr_dq	8	Inout	Data Input/Output
io_hr_rwds	1	Inout	Read Write Data Strobe

Notes:

1. These signals are fully controlled by the HyperRAM Controller design. Any activity on this interface is a direct result of either an AXI-4 Write or Read.
2. For 1.8 V operation, both signals are used and information on the interface is with respect to the crossing between the two clock signals. For 3.3 V, only o_hr_ck is used.

5. Parameters

Table 5.1 shows the global parameters used for the HyperRAM Controller reference design.

Table 5.1. Top-level Parameters*

Parameter	Default	Remarks
C_AXI_MEM_ID_WIDTH	'd4	Bus length of i_axim_awid, o_axim_bid, i_axim_arid and i_axim_rid
C_AXI_MEM_ADDR_WIDTH	'd32	Bus length of i_axim_awaddr and i_axim_araddr
C_AXI_MEM_DATA_WIDTH	'd64	Bus length of i_axim_wdata and i_axim_rdata
C_AXI_MEM_LEN_WIDTH	'd8	Bus length of i_axim_awlen and i_axim_arlen
C_AXI_MEM_DATA_INTERLEAVING	'd0	Internal Use Only
C_MEM_AW_FIFO_ADDR_BITS	'd4	Internal Use Only
C_MEM_AR_FIFO_ADDR_BITS	'd4	Internal Use Only
C_MEM_WDAT_FIFO_ADDR_BITS	'd8	Internal Use Only
C_MEM_RDAT_FIFO_ADDR_BITS	'd7	Internal Use Only
C_AXI_REG_ID_WIDTH	'd4	Bus length of i_axir_awid, o_axir_bid, i_axir_arid and i_axir_rid
C_AXI_REG_ADDR_WIDTH	'd32	Bus length of i_axir_awaddr and i_axir_araddr
C_AXI_REG_DATA_WIDTH	'd32	Bus length of i_axir_wdata and i_axir_rdata
C_AXI_REG_LEN_WIDTH	'd8	Internal Use Only
C_AXI_REG_BASEADDR	'h00000000	Internal Use Only
C_AXI_REG_HIGHADDR	'h0000004F	Internal Use Only
C_RX_FIFO_ADDR_BITS	'd8	Address Bits of the RX FIFO
DPRAM_MACRO	0	Internal Use Only
DPRAM_MACRO_TYPE	0	Internal Use Only

*__Note:__ These parameters are only tested for the default values.

6. Detailed Description

The HyperRAM Controller provides two AXI-4 access ports, which are composed of a Memory Control Port to access the HyperRAM, and a Register Control Port to access the control registers of the HyperRAM Controller.

The Memory Control Port supports AXI-4 protocols with burst-based transactions.

The Register Control Port supports the AXI4-Lite protocol with burst length always as 1 (that is, register-style interface).

6.1. Timing Diagram

6.1.1. AXI-4 Write Timing Diagram (AW, W, and B Channels)

Figure 6.1 and Table 6.1 describe how a User Logic (Host) should control the HyperRAM Controller (Controller) for an AXI-4 Write Transaction.

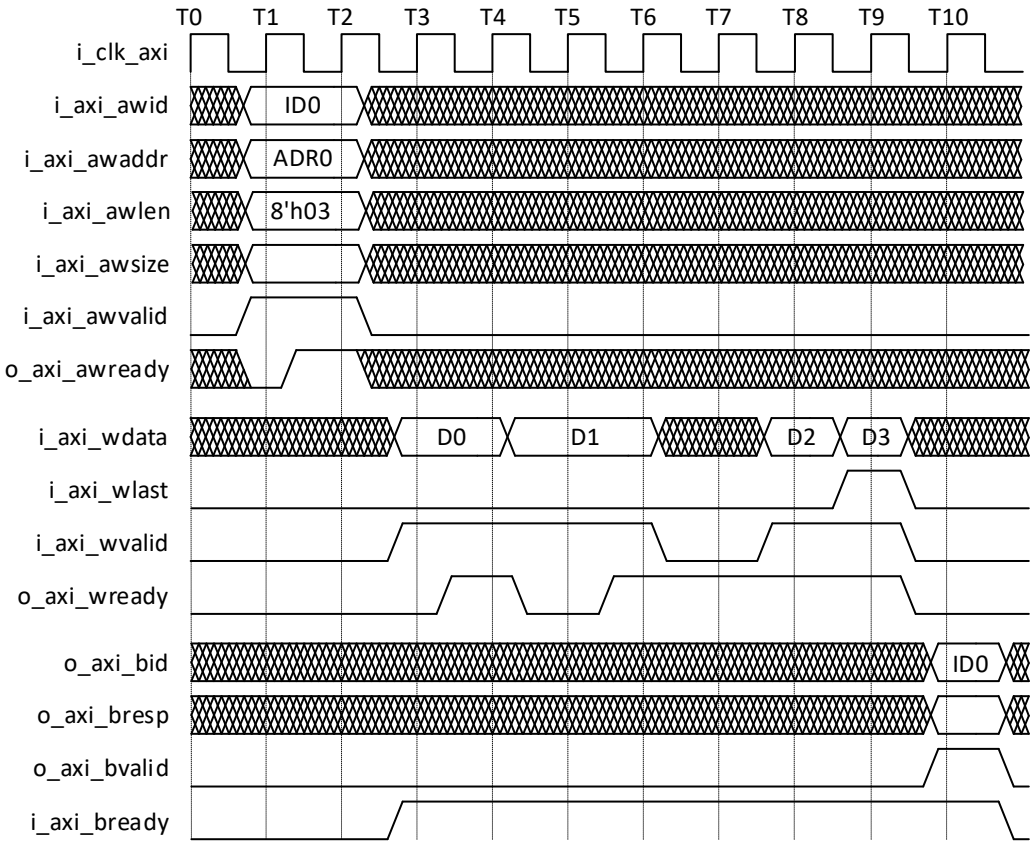

Figure 6.1. AXI-4 Write Timing

Table 6.1. AXI-4 Write Timing Diagram Description

State	Description
T1	The Host sends valid inputs on the AW channel. However, the Host needs to hold these inputs since o_axi_awready is still 0.[1]
T2	The Controller sets the o_axi_awready to 1. At this state, the Controller fetches all inputs on the AW channel since i_axi_awvalid is also 1.
T3	The Host asserts i_axi_bready. The Host can also start to assert i_axi_wvalid and write data on i_axi_wdata but needs to hold until o_axi_wready is 1. For Register Control, i_axi_wlast should also be set to 1.
T4	The Controller sets the o_axi_wready to 1. At this state, the Controller fetches data on i_axi_wdata since i_axi_wvalid is also 1.

State	Description
T5	The Host changes the data on i_axi_wdata. The Host needs to hold until o_axi_wready is 1 again.
T6	The Controller fetches D1 on i_axi_wdata. This is similar to T4.
T7	The Host deasserts i_axi_wvalid since there is no valid write data on i_axi_wdata. This prevents the Controller from fetching write data, given that o_axi_wready is 1.
T8	The Controller fetches D2 on i_axi_wdata since the Host set i_axi_wvalid to 1.
T9	The Controller fetches D3 on i_axi_wdata and is prompted that this is the last write data since the Host set i_axi_wlast to 1.[2]
T10	The Controller returns the response ID and write response, indicated by the assertion of o_axi_bvalid.[3]

Notes:

1. For this example, only four transfers are shown but this can be up to 256, depending on the value of i_axi_awlen.
2. Even when defined through i_axi_awlen, how many transfers there are in total, i_axi_wlast still needs to be asserted on the last transfer.
3. The value on o_axi_bid should be equal to the value on i_axi_awid when i_axi_awvalid is asserted.

6.1.2. AXI-4 Read Timing Diagram (AR and R Channels)

Figure 6.2 and Table 6.2 describe how a User Logic (Host) should control the HyperRAM Controller (Controller) for an AXI-4 Read Transaction.

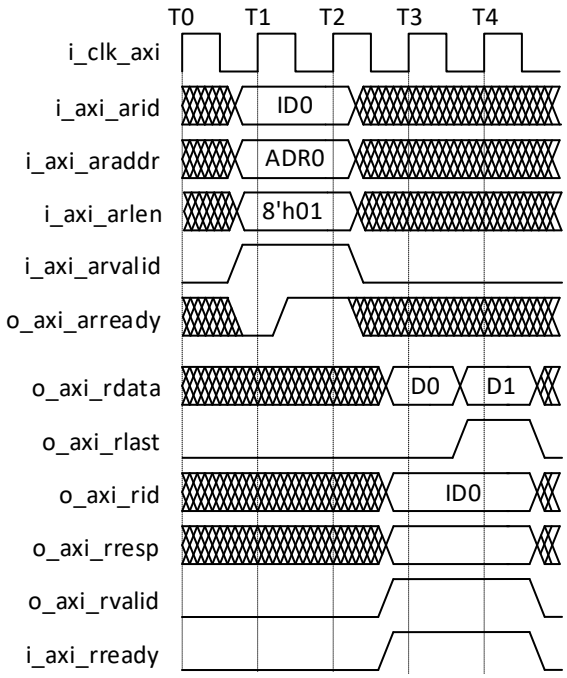

Figure 6.2. AXI-4 Read Timing

Table 6.2. AXI-4 Write Timing Diagram Description

State	Description
T1	The Host sends valid inputs on the AR channel. However, the Host needs to hold these inputs since o_axi_arready is still 0.[1]
T2	The Controller sets the o_axi_arready to 1. At this state, the Controller fetches all inputs on the AR channel since i_axi_arvalid is also 1.
T3	The Host asserts i_axi_rready. The Controller can also start to assert i_axi_rvalid and give valid output on the R channel.2.[3]
T4	The Controller sends the last set of valid output on the R channel, indicated by the assertion of o_axi_rlast.[3]

Notes:

1. For this example, only two transfers are shown but this can be up to 256, depending on the value of i_axi_arlen.
2. The first valid read data is not available immediately after T2. However, even when the read data is ready, if i_axi_rready is not asserted, the HC does not send the output on the R channel.
3. The value on o_axi_rid should be equal to the value on i_axi_arid when i_axi_arvalid was asserted.

7. HDL Simulation and Verification

7.1. Software Validation

The product is validated using the Lattice Radiant version 2.2.

7.2. Simulation

The HyperRAM Controller is simulated using a testbench file, tb.v, which acts as the AXI-4 Host for both Memory and Register Control mentioned in Figure 3.1. It makes use of the task command of Verilog to allow you to call the fixed functions for Register Control Write and Read, Memory Control Write, and Memory Control Read.

The two connected HyperRAM devices are referred to on the testbench as HyperRAM_A (connected to o_hr_cs0n) and HyperRAM_B (connected to o_hr_cs1n).

The default setup on the testbench file is as described in Table 7.1. Figure 7.1 to Figure 7.13 show the images for each step.

Table 7.1. Default Simulation Flow

Step	Description
1 – Reset	The HyperRAM Controller is reset for at least 300 µs.
2 – Set Base Address for HyperRAM devices	The default address offset (MBAR0 and MBAR1) is 0. During this step, the address offset for HyperRAM_B is set to 0x10000000 while the address offset for HyperRAM_A is kept on default. This allows the controller to distinguish which HyperRAM is being written/read to.
3 – Set Latency Cycle to 5 CK	The Latency Cycle (MTR0 [3:0] and MTR1 [3:0]) for both HyperRAM devices is set to 5 CK.
4 – Set Device Type to HyperRAM	The Device Type (MCR0 [4] and MCR1 [4]) for both HyperRAM devices is set to HyperRAM.
5 – Write Data on HyperRAM_A	HyperRAM_A is written with a valid data (0xA8A7A6A5A4A3A2A1A0) with an ID = 0xA
6 – Read Data on HyperRAM_A	HyperRAM_A is read, expecting 0xA8A7A6A5A4A3A2A1A0 with an ID = 0xA
7 – Write Data on HyperRAM_B	HyperRAM_B is written with a valid data (0xB8B7B6B5B4B3B2B1B0) with an ID = 0xB
8 – Read Data on HyperRAM_B	HyperRAM_B is read, expecting 0xB8B7B6B5B4B3B2B1B0 with an ID = 0xB

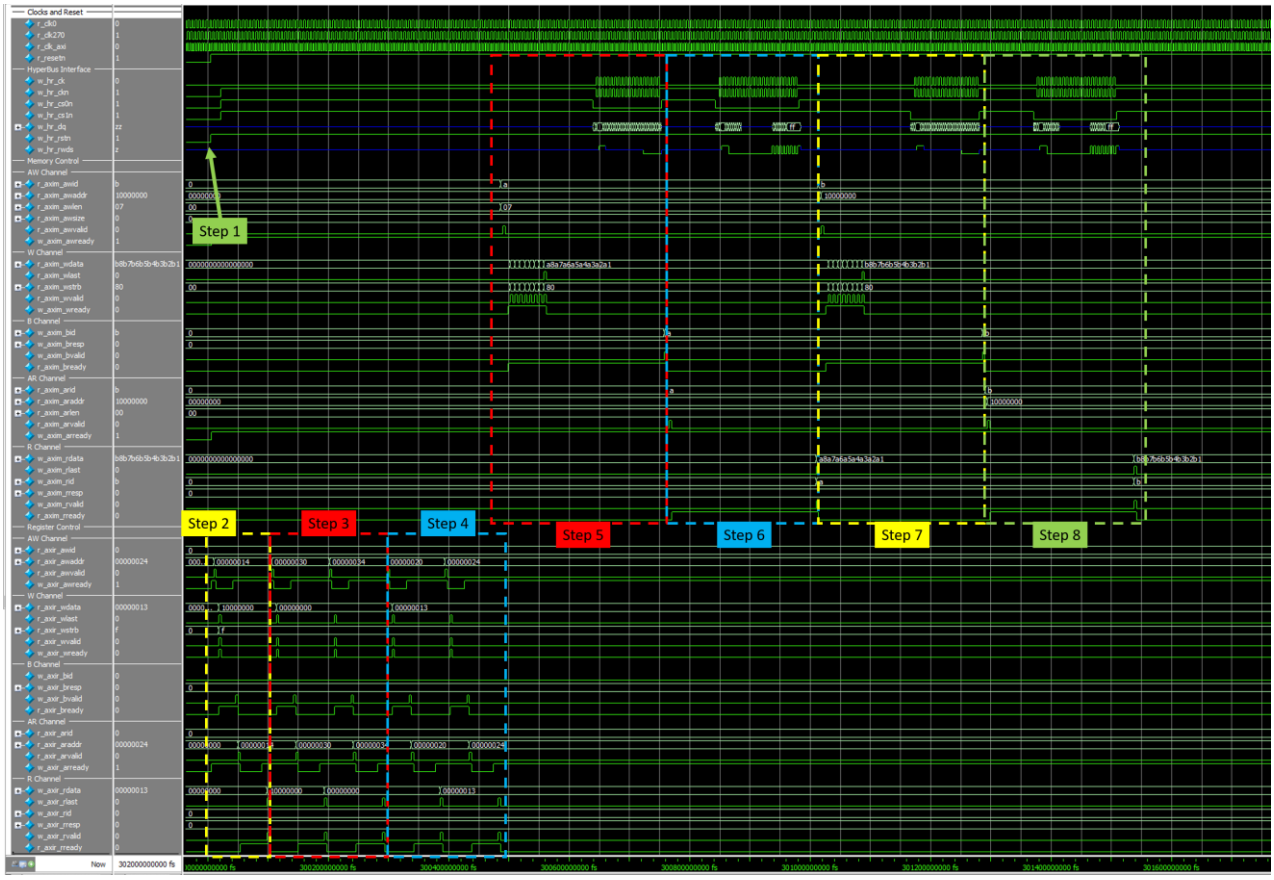

Figure 7.1. Simulation Overview

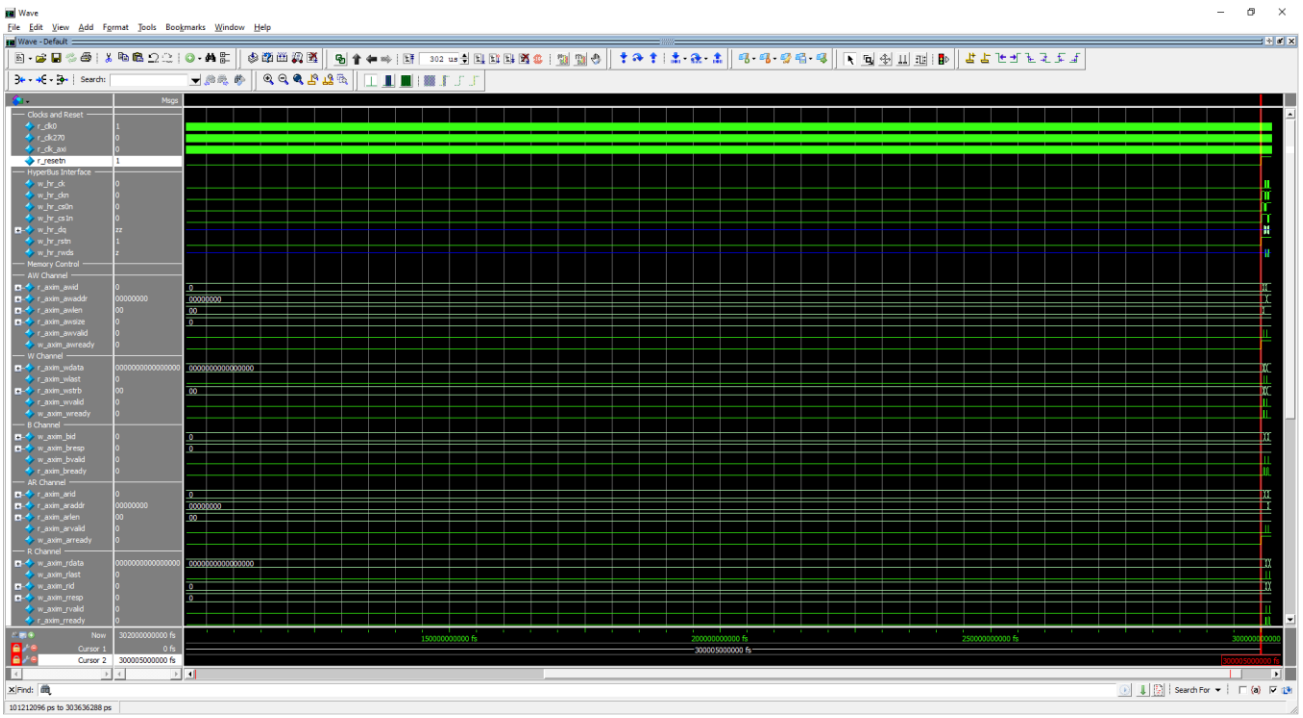

Figure 7.2. Step 1 – Reset

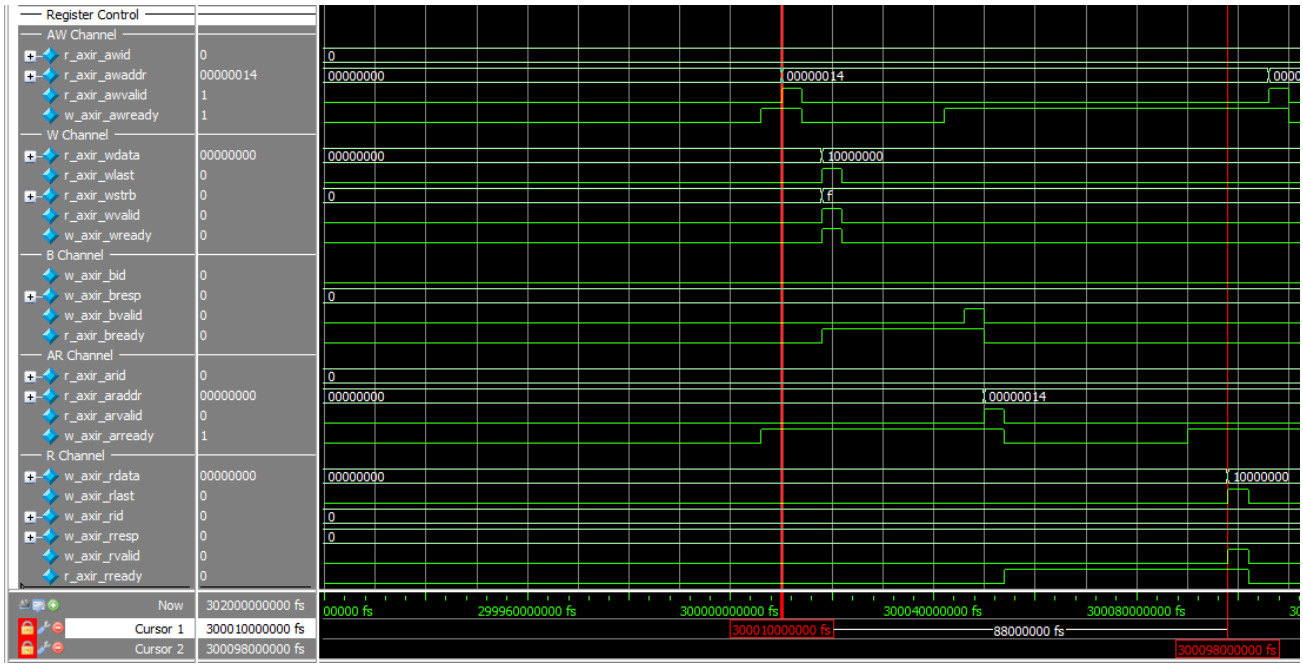

Figure 7.3. Step 2 – Set Base Address for HyperRAM Devices

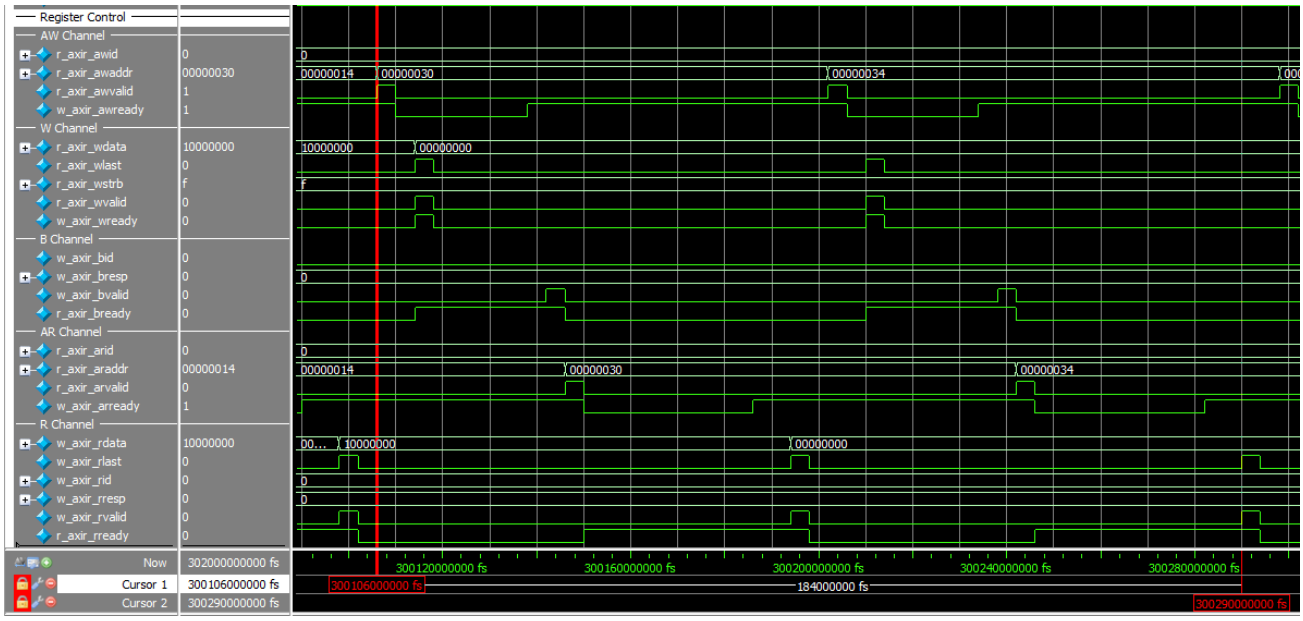

Figure 7.4. Step 3 – Set Latency Cycle to 5 CK

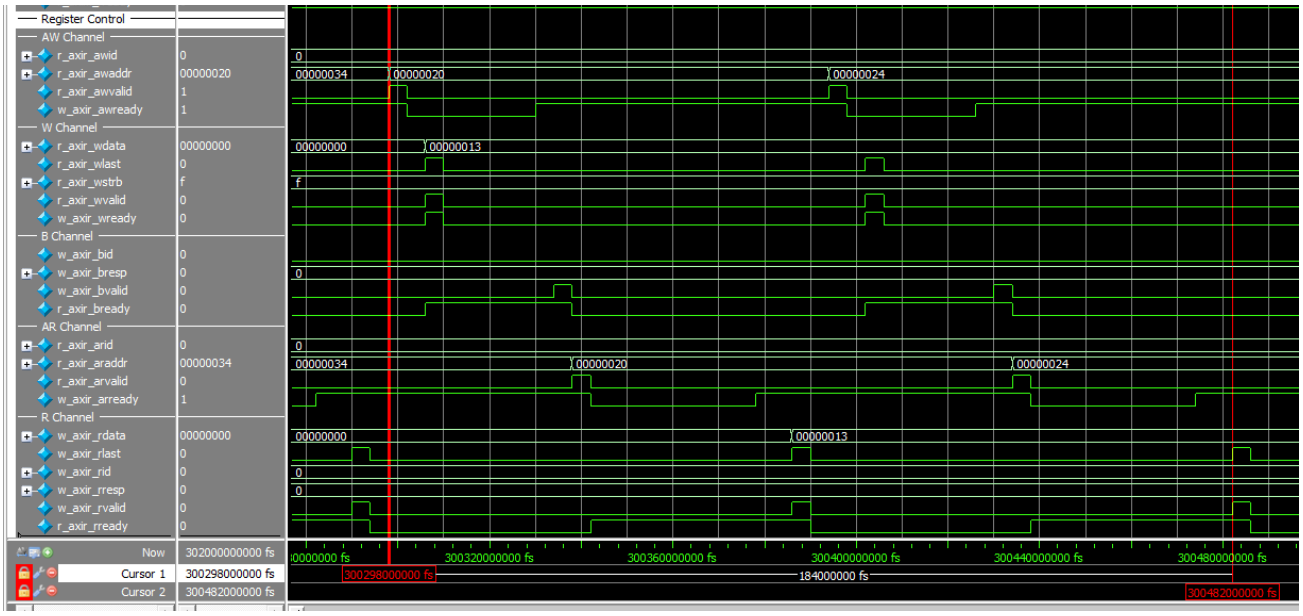

Figure 7.5. Step 4 – Set Device Type to HyperRAM

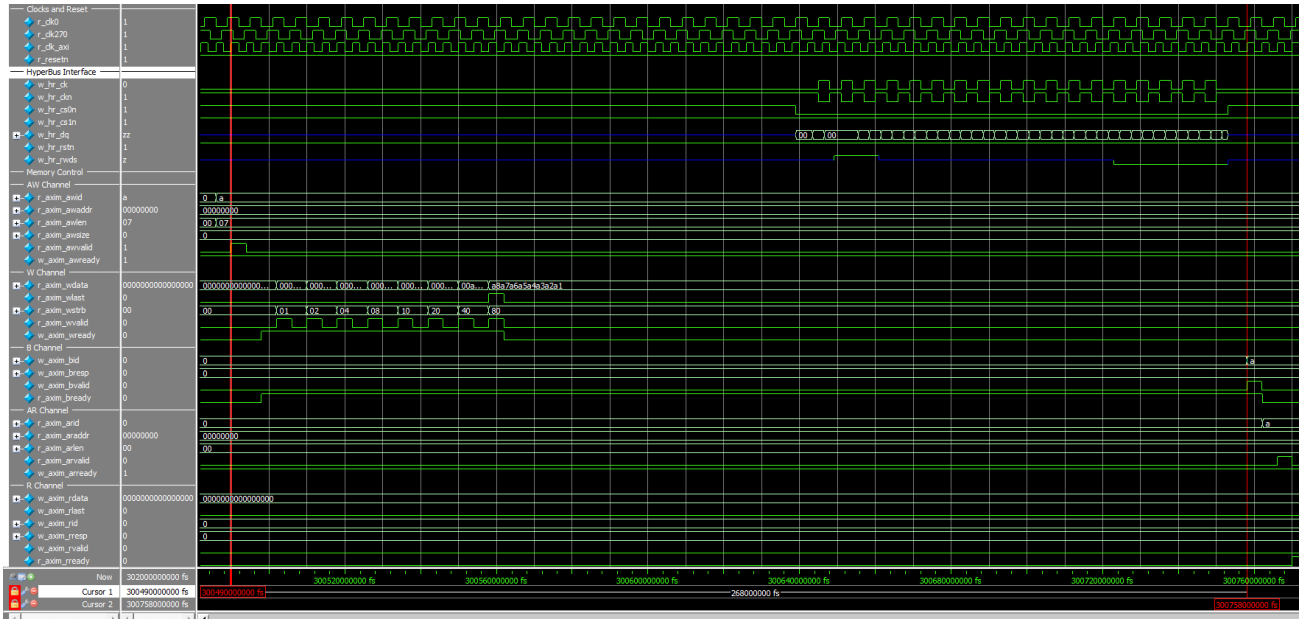

Figure 7.6. Step 5 – Write Data on HyperRAM_A (AXI-4 Signals)

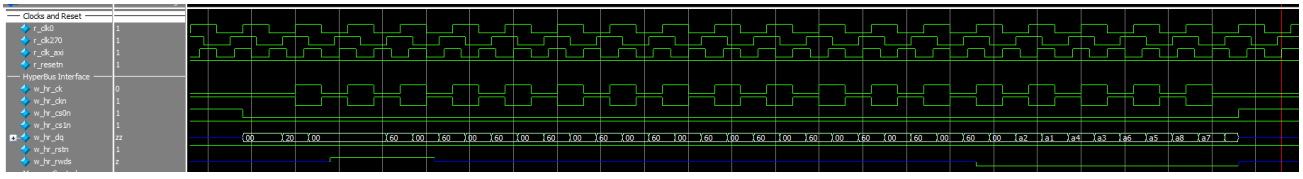

Figure 7.7. Step 5 – Write Data on HyperRAM_A (HyperBus Signals)

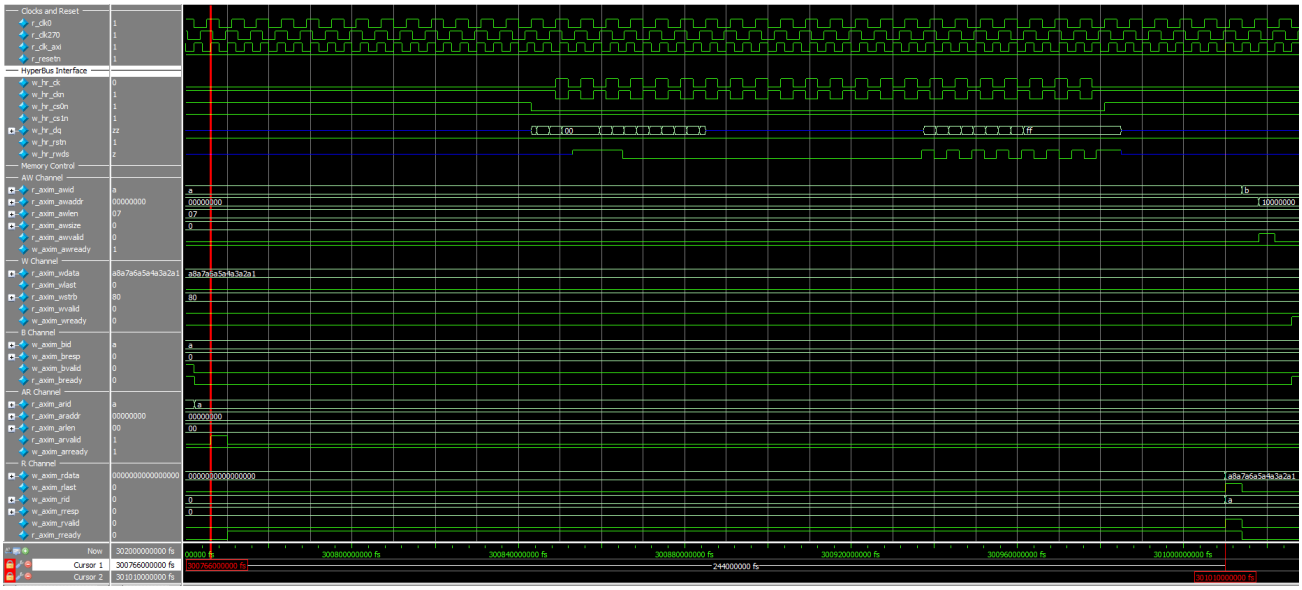

Figure 7.8. Step 6 – Read Data on HyperRAM_A (AXI-4 Signals)

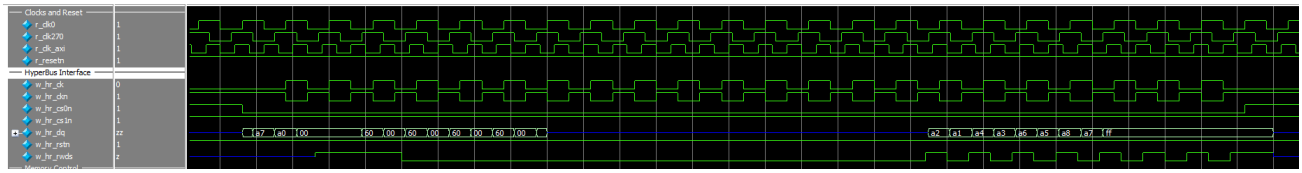

Figure 7.9. Step 6 – Read Data on HyperRAM_A (HyperBus Signals)

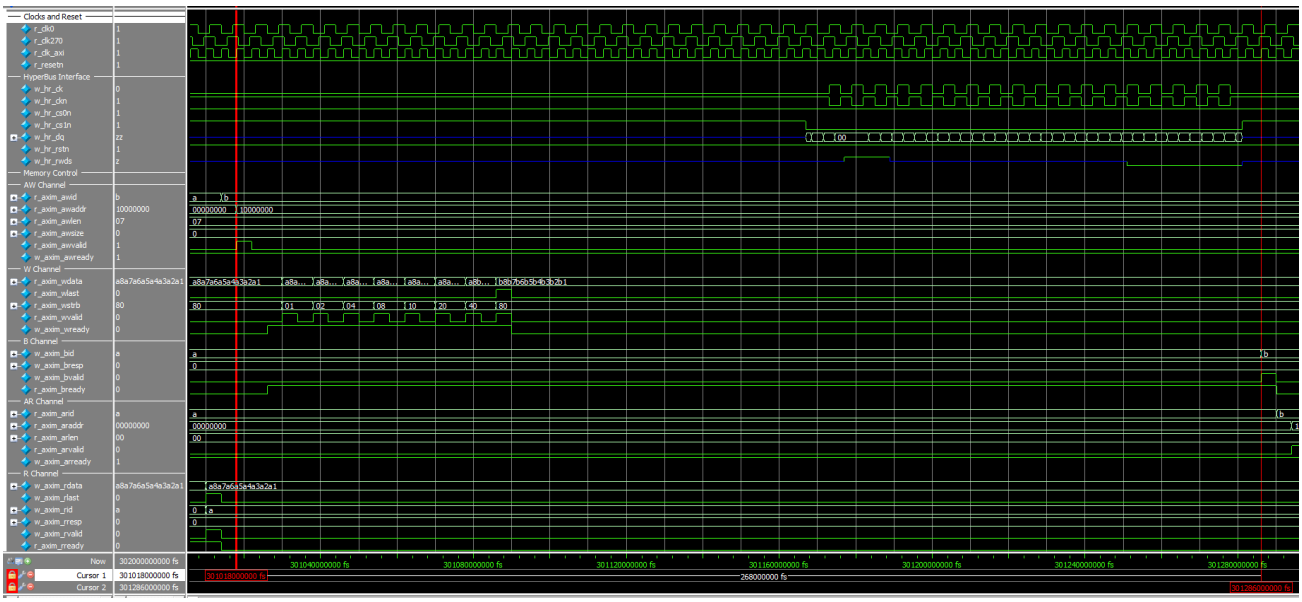

Figure 7.10. Step 7 – Write Data on HyperRAM_B (AXI-4 Signals)

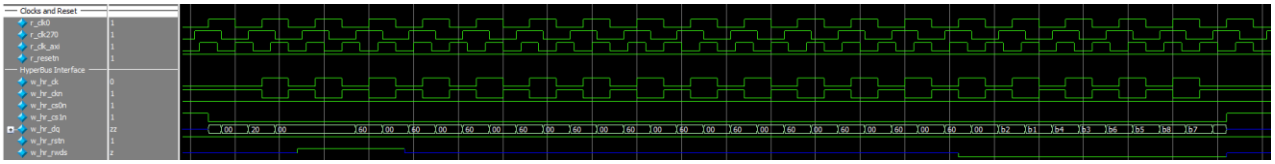

Figure 7.11. Step 7 – Write Data on HyperRAM_B (HyperBus Signals)

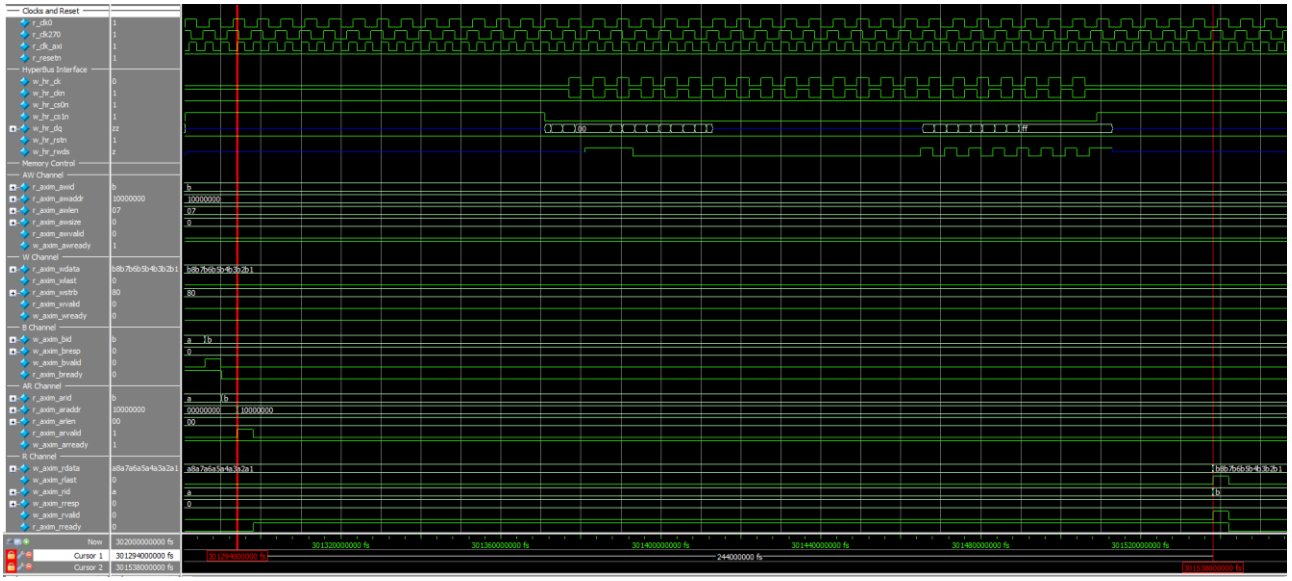

Figure 7.12. Step 8 – Read Data on HyperRAM_B (AXI-4 Signals)

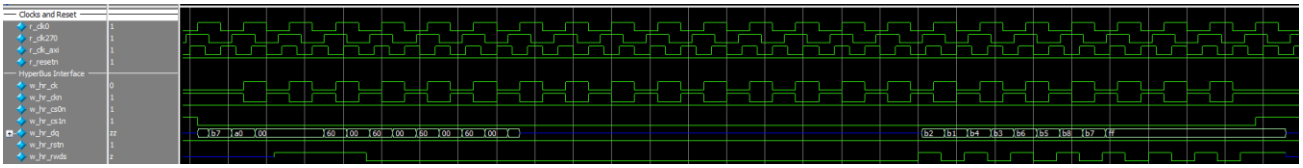

Figure 7.13. Step 8 – Read Data on HyperRAM_B (HyperBus Signals)

8. Packaged Design

The reference design folder (HyperRAM_Controller) contains six subfolders: Docs, IP, Project, Simulation, Source, and Testbench. The details of each subfolder are as follows:

- Bitstream – contains the Companion demo bitstream (VVML_demo.bit), Human Count demo bitstream (HC_demo.bit) and the Lattice SensAI™ Firmware (Firmware.mcs).
- Docs – contains this file and the demo documentation.
- Project – contains the project files for CrossLink-NX.
- Simulation – contains the simulation file (*.do) used to run RTL simulation for ModelSim.
- Source – contains all the HyperRAM Controller RTL files.
- Testbench – contains the testbench files and the simulation model for a HyperRAM device.

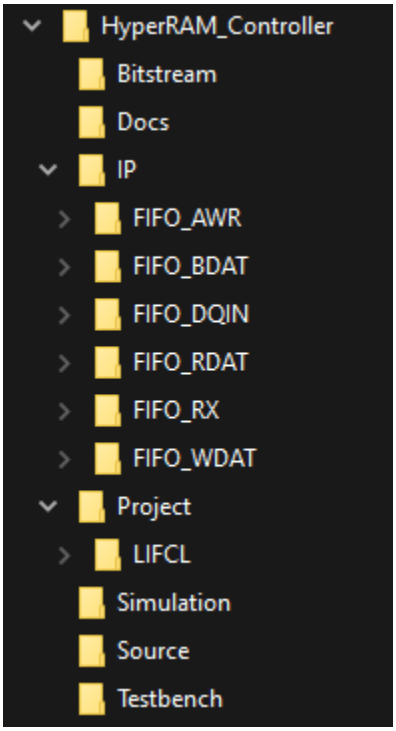

Figure 8.1. Packaged Design Directory Structure

9. Hardware Validation

This reference design was hardware validated using a CrossLink-NX Voice and Vision Machine Learning Board (LIFCL-EVN-VVML), through integration on the Human Counting Demo. The sample bitstream for this demo is included on the Bitstream folder, as well as the documentation on the Docs folder.

9.1. Companion Demo

A companion demo was also created to allow you to perform actual hardware validation on the aforementioned board, without using the Lattice SensAI demo. The demo utilizes the switches and pushbuttons of the board to write and read data from the HyperRAM module.

9.1.1. Block Diagram

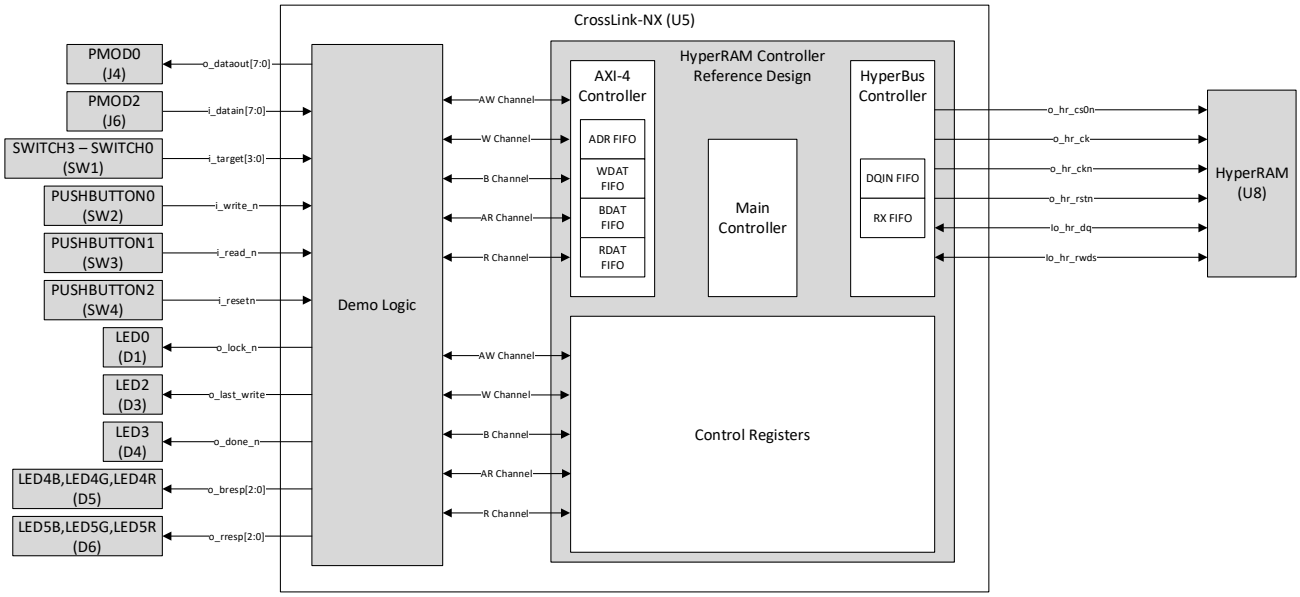

Figure 9.1. Block Diagram of the Companion Demo

9.1.2. Pin Mapping to the VVML Board

This section defines the connection of the Demo Logic ports to the VVML board.

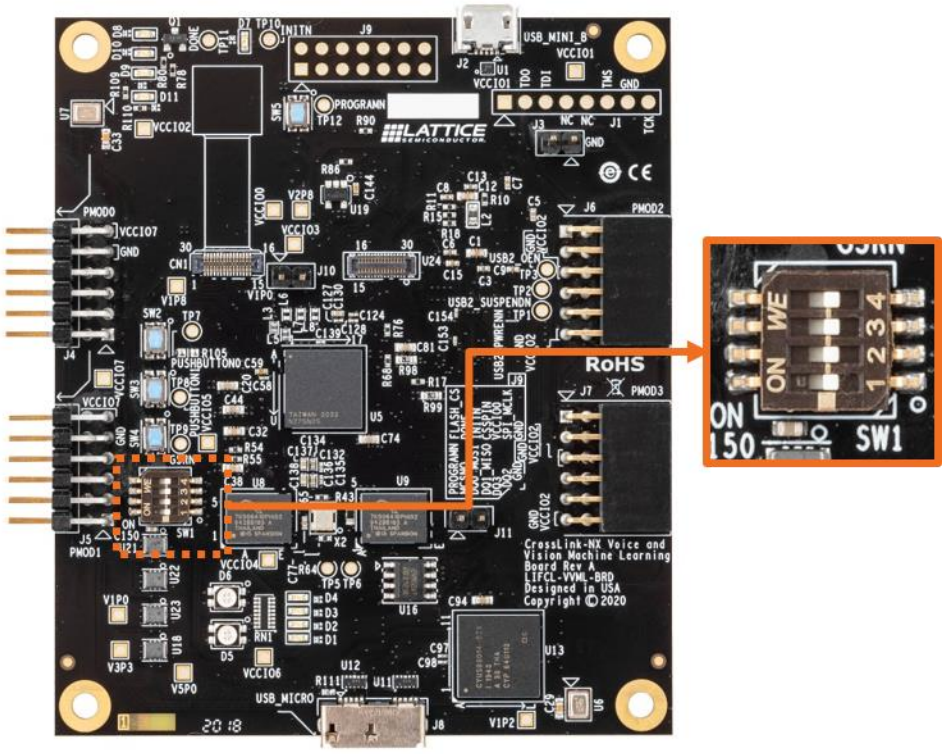

Figure 9.2. Board Location – Control Target

Table 9.1. Command Target

i_target[3] (SW1.3)	i_target[2] (SW1.2)	i_target[1] (SW1.1)	i_target[0] (SW1.0)	Command Target
0	0	0	0	WRITE_READ
0	0	0	1	ADDRESS0
0	0	1	0	ADDRESS1
0	0	1	1	ADDRESS2
0	1	0	0	RC_DATA0
0	1	0	1	RC_DATA1
0	1	1	0	RC_DATA2
0	1	1	1	RC_DATA3
1	0	0	0	MC_DATA0
1	0	0	1	MC_DATA1
1	0	1	0	MC_DATA2
1	0	1	1	MC_DATA3
1	1	0	0	MC_DATA4
1	1	0	1	MC_DATA5
1	1	1	0	MC_DATA6
1	1	1	1	MC_DATA7

Notes:

1. ADDRESSx refers to the 8-bit sections of the 32-bit data register, r_address. Only the lower 24-bits of the register can be modified since:

a. Memory Control only needs a 24-bit address for the HyperRAM and the highest byte (r_address[31:24]) is only used to distinguish addresses that are for cs0n and cs1n. This function is not tested on the board due to the current layout, thus only cs0n is used and the highest byte is kept at 0 after reset.
b. Register Control only needs the lowest byte to access its register file.

2. RC_DATAx refers to the 8-bit sections of the 32-bit data registers, r_rc_wdata and r_rc_rdata.
3. MC_DATAx refers to the 8-bit sections of the 64-bit data register, r_mc_wdata and r_mc_rdata

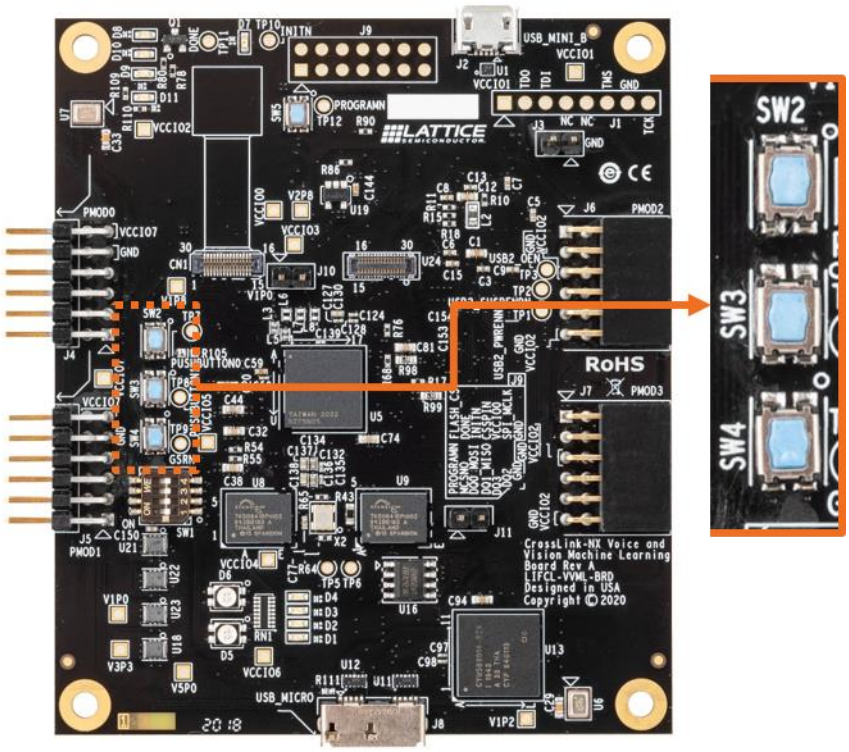

Figure 9.3. Board Location – Control Triggers and Reset

Table 9.2. Command Triggers and Reset

Signal Name	Switch	Description
i_write_n	PUSHBUTTON0: SW2 (K2)	• For i_target = 0x0 (WRITE_READ), starts the write transaction on the AXI-4 Interface. The interface to be written depends on the value of o_last_write (0 = Register Control, 1 = Memory Control). • For i_target = 0x1 to 0x3 (ADDRESSx), the value on i_datain is written on the r_address register. This register is connected directly to the AWADDR and ARADDR buses of both AXI-4 Interfaces. • For i_target = 0x4 to 0x7 (RC_DATAx), the value on i_datain is written on the r_rc_wdata register. This register is connected directly to the WDATA bus of the Register Control AXI-4 Interface. • For i_target = 0x8 to 0xF (MC_DATAx), the value on i_datain is written on the r_mc_wdata register. This register is connected directly to the WDATA bus of the Memory Control AXI-4 Interface.

Signal Name	Switch	Description
i_read_n	PUSHBUTTON1: SW3 (L1)	• For i_target = 0x0 (WRITE_READ), starts the read transaction on the AXI-4 Interface. The interface to be read depends on the value of o_last_write (0 = Register Control, 1 = Memory Control). • For i_target = 0x1 to 0x3 (ADDRESSx), the value on the r_address register is written to o_dataout. This register is connected directly to the AWADDR and ARADDR buses of both AXI-4 Interfaces. • For i_target = 0x4 to 0x7 (RC_DATAx), the value on the r_rc_wdata register is written to o_dataout. This register is connected directly to the WDATA bus of the Register Control AXI-4 Interface. • For i_target = 0x8 to 0xF (MC_DATAx), the value on i_datain is written on the r_mc_wdata register. This register is connected directly to the WDATA bus of the Memory Control AXI-4 Interface.
i_resetn	GSRN: SW4 (L2)	Active low asynchronous system reset.

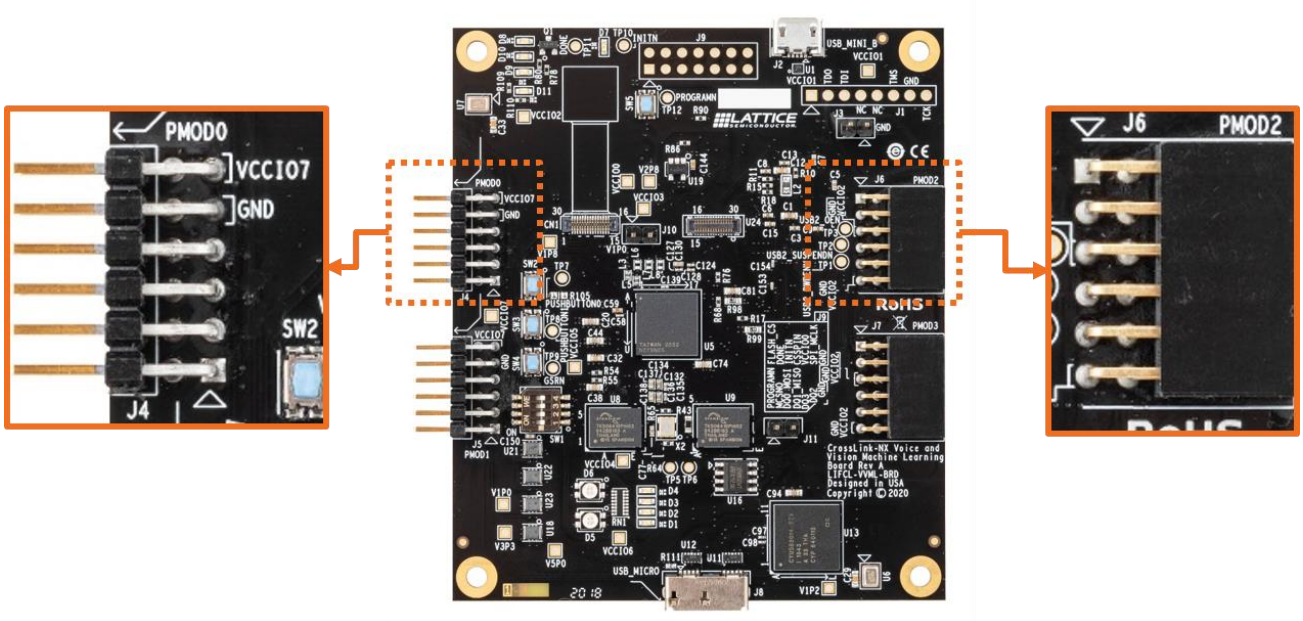

Figure 9.4. Board Location – Data I/O

Table 9.3. Output Data

Signal Name	Location
o_dataout[0]	PMOD0_1
o_dataout[1]	PMOD0_2
o_dataout[2]	PMOD0_3
o_dataout[3]	PMOD0_4
o_dataout[4]	PMOD0_7
o_dataout[5]	PMOD0_8
o_dataout[6]	PMOD0_9
o_dataout[7]	PMOD0_10

Table 9.4. Input Data

Signal Name	Location
i_datain[0]	PMOD2_1
i_datain[1]	PMOD2_2
i_datain[2]	PMOD2_3
i_datain[3]	PMOD2_4
i_datain[4]	PMOD2_7
i_datain[5]	PMOD2_8
i_datain[6]	PMOD2_9
i_datain[7]	PMOD2_10

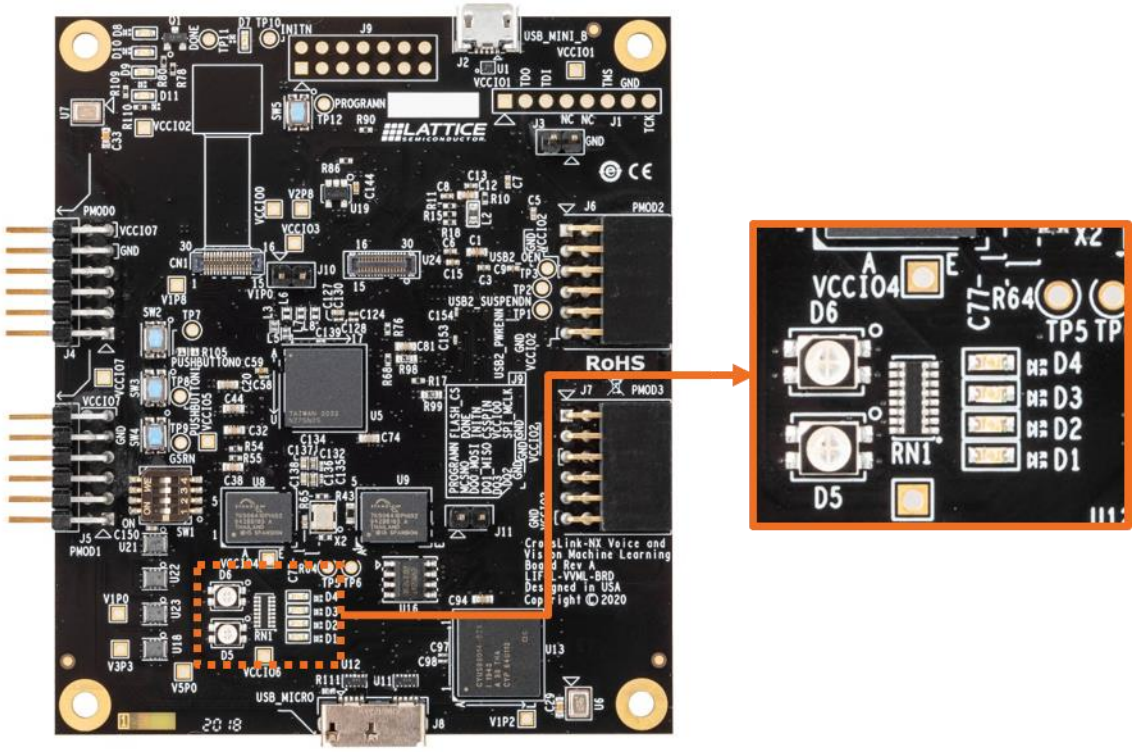

Figure 9.5. Board Location – Status Indicators

Table 9.5. Status Indicators

LED	Net Name	Signal Name	Command Target
D1	LED0	o_lock_n	PLL Lock. 0 = Lock, 1 = No Lock
D3	LED2	o_last_write	0 = Register Control, 1 = Memory Control Indicates which bus is the target of the WRITE_READ (i_target = 4'b0000) command. Changes value when the data register for Register Control or for Memory Control is written/read.
D4	LED3	o_done_n	0 = Write or Read Done, either through WRITE_READ or data registers.

LED	Net Name	Signal Name	Command Target
D5	LED4R	o_bresp[0]	Write Response. [00] = OKAY, the success of a normal access. [01] = EXOKAY, not applicable for this design. [10] = SLVERR, an unsuccessful transaction. [11] = DECERR, the interconnect cannot successfully decode a slave access. 0 = Register Control, 1 = Memory Control. Indicates which bus is gave the BRESP output.
	LED4G	o_bresp[1]	
	LED4B	o_bresp[2]	
D6	LED5R	o_rresp[0]	Read Response. [00] = OKAY, the success of a normal access. [01] = EXOKAY, not applicable for this design. [10] = SLVERR, an unsuccessful transaction. [11] = DECERR, the interconnect cannot successfully decode a slave access.
	LED5G	o_rresp[1]	
	LED5B	o_rresp[2]	

9.1.3. Demo Procedure

9.1.3.1. Initialization of the Control Registers

This step demonstrates how to initialize the control registers of the reference design.

To start, you need to set the DEVTYPE to HyperRAM on MCR0. The value of MCR0[4] is set to 1, indicating that the device connected to CS0 is a HyperRAM device. To set the DEVTYPE, perform the steps below:

1. Set i_target = ADDRESS0.
2. Set i_datain = 0x20.
3. Press SW2.
4. Set i_target = RC_DATA0.
5. Set i_datain = 0x13.
6. Press SW2.
7. Set i_target = WRITE_READ.
8. Press SW2.
9. Wait for D4 to light up.

To set the LATENCY to 5 CK on MTR0, perform the steps below:

1. Set i_target = ADDRESS0
2. Set i_datain = 0x30
3. Press SW2
4. Set i_target = RC_DATA0
5. Set i_datain = 0x00
6. Press SW2
7. Set i_target = WRITE_READ
8. Press SW2
9. Wait for D4 to light up

9.1.3.2. Writing Data to the HyperRAM Device (U8)

This step demonstrates how to write an 8-byte data to the HyperRAM device.
To set the address where the data is written:

1. Set i_target = ADDRESS0.
2. Set i_datain = user_address_byte.
   Note: The resulting address is {ADDRESS2, ADDRESS1, ADDRESS0}.
3. Press SW2.
4. Repeat step 1 to step 3 for ADDRESS1 and ADDRESS2.

To set the data to be written:

1. Set i_target = MC_DATA0.
2. Set i_datain = user_data_byte.
   Note: The resulting data is {MC_DATA7, MC_DATA6, ... , MC_DATA1, MC_DATA0}.
3. Press SW2.
4. Repeat step 1 to step 3 for MC_DATA1 to MC_DATA7.

To execute the HyperRAM Write:

1. Set i_target = WRITE_READ.
2. Press SW2.
3. Wait for D4 to light up.

9.1.3.3. Reading Data from the HyperRAM Device (U8)

This step demonstrates how to read an 8-byte data from the HyperRAM device.
To set the address where the data is read:

1. Set i_target = ADDRESS0.
2. Set i_datain = user_address_byte.
   Note: The resulting address is {ADDRESS2, ADDRESS1, ADDRESS0}.
3. Press SW2.
4. Repeat step 1 to step 3 for ADDRESS1 and ADDRESS2.

To execute the HyperRAM Read:

1. Set i_target = WRITE_READ.
2. Press SW3.
3. Wait for D4 to light up.

To select the data to be read:

1. Set i_target = MC_DATA0.
   Note: The data is read as {MC_DATA7, MC_DATA6, ... , MC_DATA1, MC_DATA0}.
2. Press SW3.
3. Repeat step 1 and step 2 for MC_DATA1 to MC_DATA7.

9.1.4. Simulation Results

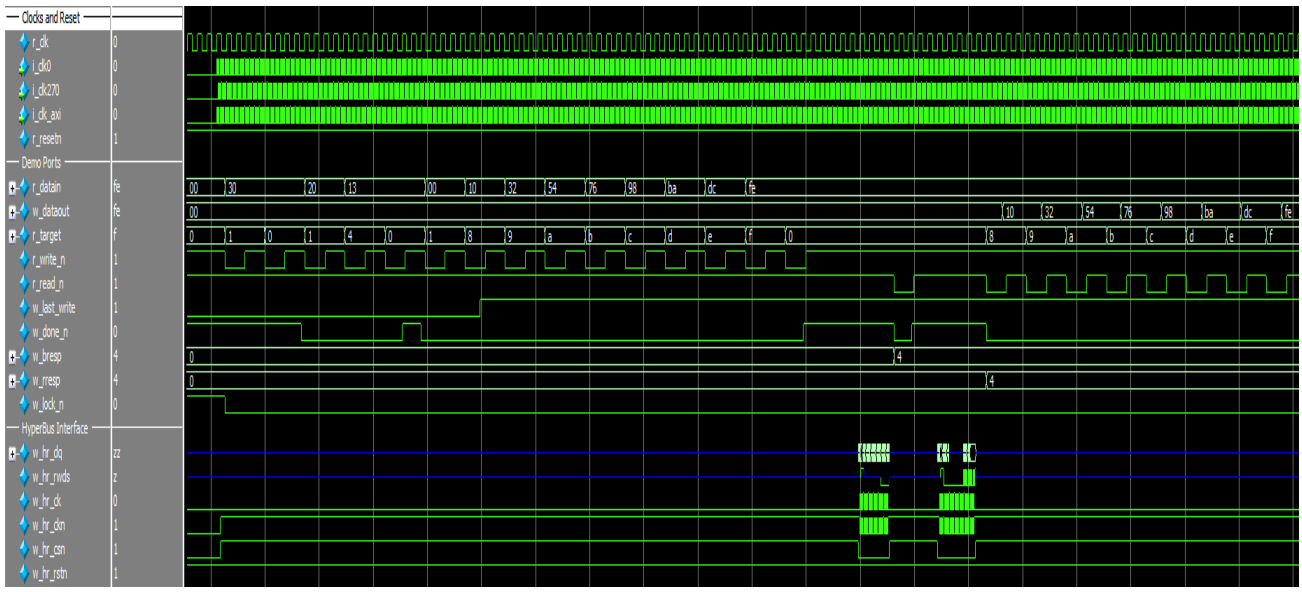

Figure 9.6. Simulation Results – Demo Logic and HyperBus Signals

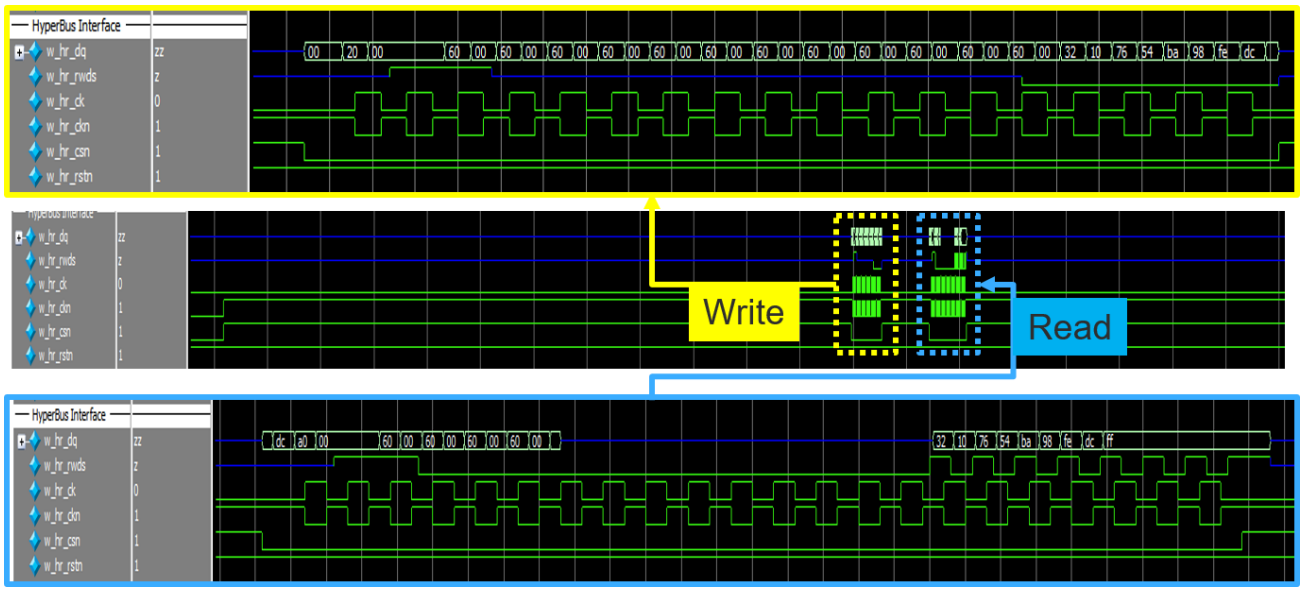

Figure 9.7. Simulation Results –HyperBus Operations (Write and Read)

10. Resource Utilization

The Resource Utilization depends on the configuration. Table 10.1 shows the utilization from using the HyperRAM Controller Reference Design on the Human Count Demo. Utilization may vary depending per design.

Table 10.1. Resource Utilization Example

LUT	PFU Registers	EBR	HP I/O	WR I/O
2179	1590	14	15	0

References

For more information, refer to the following documents:

- HyperBus Specification
- Getting Started with HyperRAM
- CrossLink-NX Voice and Vision Machine Learning Board Evaluation Board User Guide (FPGA-EB-02039)

Technical Support Assistance

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

Revision History

Revision 1.0, August 2021

Section	Change Summary
All	Initial release.

www.latticesemi.com