AXI Register Slice IP

IP Version: 1.2.0

User Guide

FPGA-IPUG-02235-1.1

June 2025

Disclaimers

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

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.

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
AMBA	Advanced Microcontroller Bus Architecture
AXI	Advanced extensible Interface
FPGA	Field Programmable Gate Array
GUI	Graphical User Interface
IDE	Integrated Development Environment
IP	Intellectual Property
RTL	Register Transfer Level

1. Introduction

The AXI Register Slice IP core connects one AXI standard manager to one AXI standard subordinate by introducing pipeline stages in between the two to close timing in critical paths. Different configuration options are available. Each AXI channel transfers information in only one direction and the architecture does not require a fixed relationship between the channels. The user can insert a register slice at almost any point in any channel, with an additional cycle of latency.

The AXI Register Slice IP allows:

- A trade-off between cycles of latency and the maximum frequency of operation.
- A direct, fast connection between a processor and high-performance memory, by using simple register slices to isolate the longer path to less performance critical peripherals.

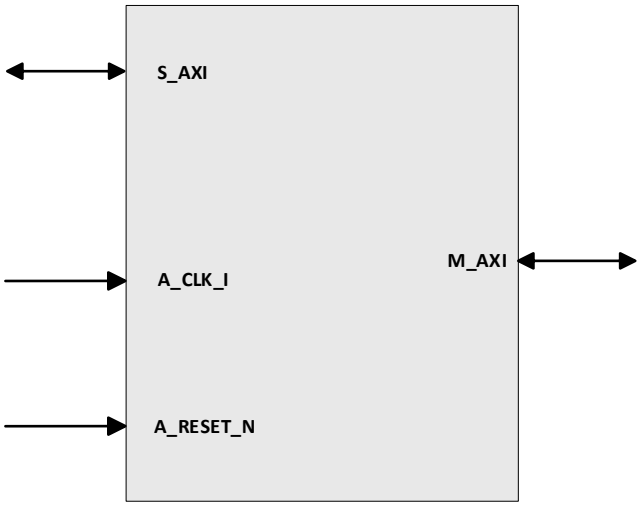


Figure 1.1. AXI Register Slice Block Diagram

In general, this document covers design specifications from RTL to IP packaging and details the procedures for IP generation and integration.

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

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

Supported Devices	IP Configuration and Generation	IP Implementation (Synthesis, Map, Place and Route)
LatticeECP3	Lattice Propel™ Builder software	Lattice Diamond™ software
ECP5	Lattice Propel Builder software	Lattice Diamond software
CrossLink™-NX	Lattice Propel Builder software	Lattice Radiant™ software
Certus™-NX	Lattice Propel Builder software	Lattice Radiant software
MachXO5™-NX	Lattice Propel Builder software	Lattice Radiant software
CertusPro™-NX	Lattice Propel Builder software	Lattice Radiant software
Lattice Avant™	Lattice Propel Builder software	Lattice Radiant software
Certus-N2	Lattice Propel Builder software	Lattice Radiant software

1.1. Features

The key features of the AXI register slice IP include:

- Individually configurable AXI channels.
- Ability to facilitate timing closure by trading-off frequency versus latency.
- One latency cycle per register-slice by default.

- Ability to propagate AXI traffic with no loss in data throughput (without bubble cycles) under all AXI handshake conditions.
- Full-Weight, Light-Weight, and Input Registered modes of operation.
- Support for AXI3, AXI4, and AXI4-Lite modes of the AXI protocol.

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 is active low
- _i are input signals
- _o are output signals
- _io are bi-directional input/output signals

Signal Names that begin with:

- r_ are registers
- w_ are wires
- i_ are input signals
- o_ are output signals
- io_ are bi-directional input/output signals
- s_ are related to subordinate
- m_ are related to manager
- a_ are related to system signals

1.2.3. Attribute

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

2. Functional Description

2.1. Overview

Figure 2.1 shows a functional block diagram of the AXI register slice with two types of ports, manager and subordinate. The subordinate port connects to the AXI manager, and the manager port connects to AXI subordinate. The IP connects the standard AXI subordinate to the standard AXI manager by introducing pipeline stages between the two to help close the timing in critical paths. Figure 2.1 shows the AXI register slice IP with three blocks: an AXI interface to sample vector, an AXI slice, and a sample vector to AXI interface block and three interfaces, M_AXI, S_AXI and clock-reset (A_CLK_I, A_RESET_N).

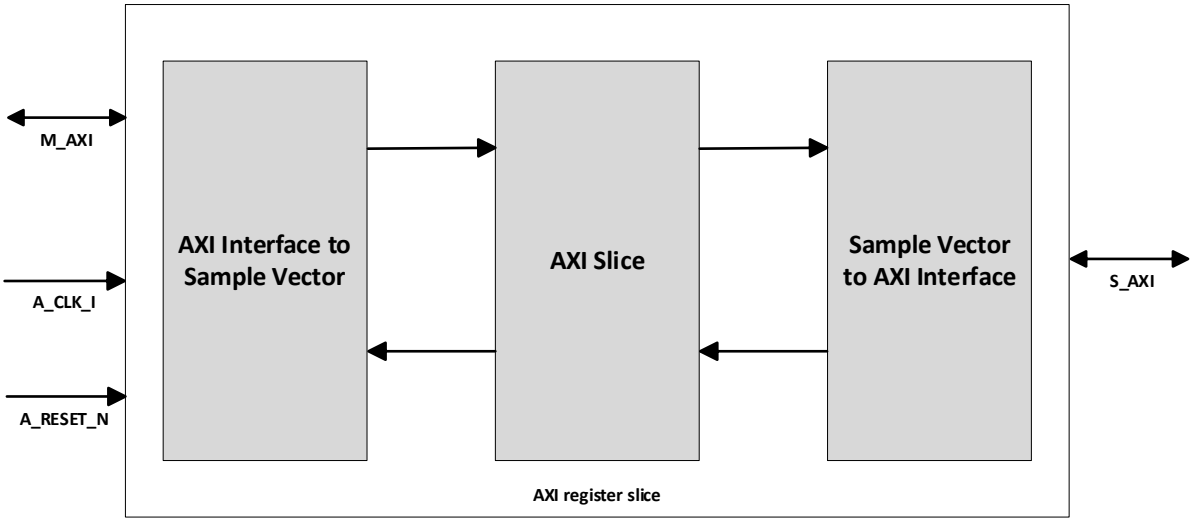


Figure 2.1. AXI Register Slice Functional Block Diagram

The AXI interface to sample vector block performs merging and splitting operations. Each AXI channel has different signals. Signals from each AXI channel are merged into a single large vector before the register operation is performed on the merged vector.

The AXI interface to sample vector block splits the read data and write response (w_m_axi_merge_r/b_ch) signals (from the AXI slice block and sends them to the M_AXI interface).

The block merges the M_AXI interface signals - write address, data, and read address (except valid and ready) signals into different w_s_axi_merge_aw/w/ar_ch signals connected to the AXI slice block and to the S_AXI interface.

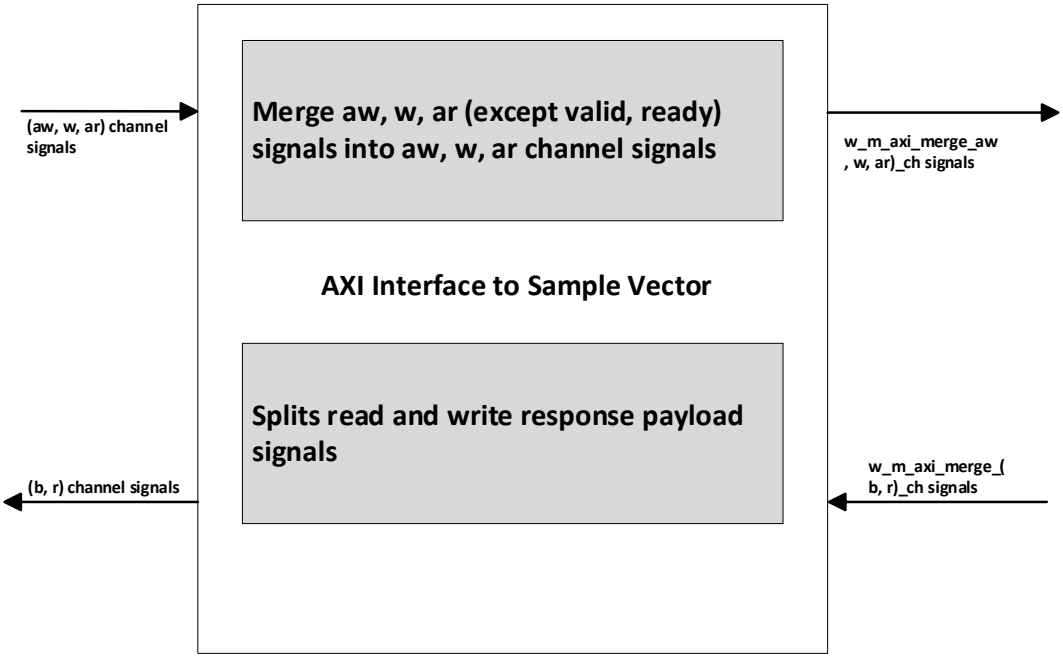


Figure 2.2. AXI Interface to Sample Vector Block Diagram

The AXI slice performs the register slice operation on w_axi_merge_aw/w/b/ar/r_ch signals. It receives the w_axi_merge_*_ch signals from two blocks:

1. AXI interface to sample vector block - Write address, data and read address.
2. Sample vector to AXI interface block - Read data and write response.

The AXI slice operates in three modes: Full-Weight, Light-Weight, and Input Registered. It provides the merged write address, data, and read address signals to the sample vector to AXI interface block, which sends them to the S_AXI interface. The merged read data and write response signals are transmitted to the AXI interface to sample vector block.

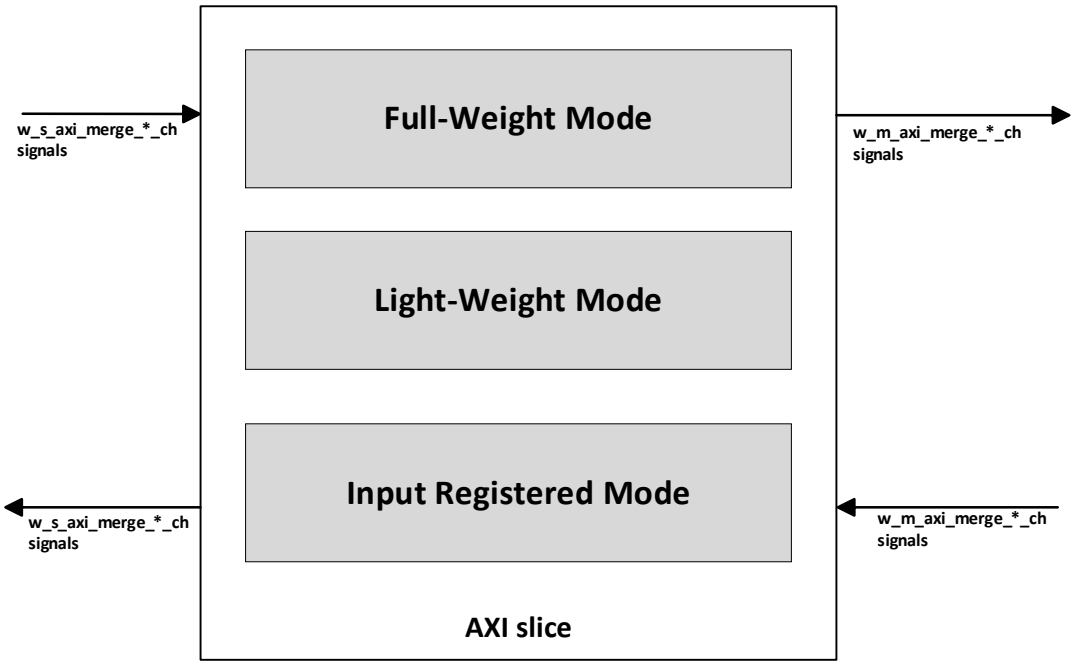


Figure 2.3. AXI Slice Functional Block Diagram

2.1.1. Modes of Slice Operation

2.1.1.1. Full-Weight Mode

The Full-Weight mode is constructed with two-stage pipeline registers. This mode supports back-to-back transfers (100% duty cycle) with one cycle of forward latency (but no bubble cycle), appropriate for write and read channels of AXI4 or AXI3 burst transfers. The Full-Weight mode drives registered output, valid, and ready signals as shown in Figure 2.4. Refer to the Attributes Summary section for details on how to select the Full-Weight mode.

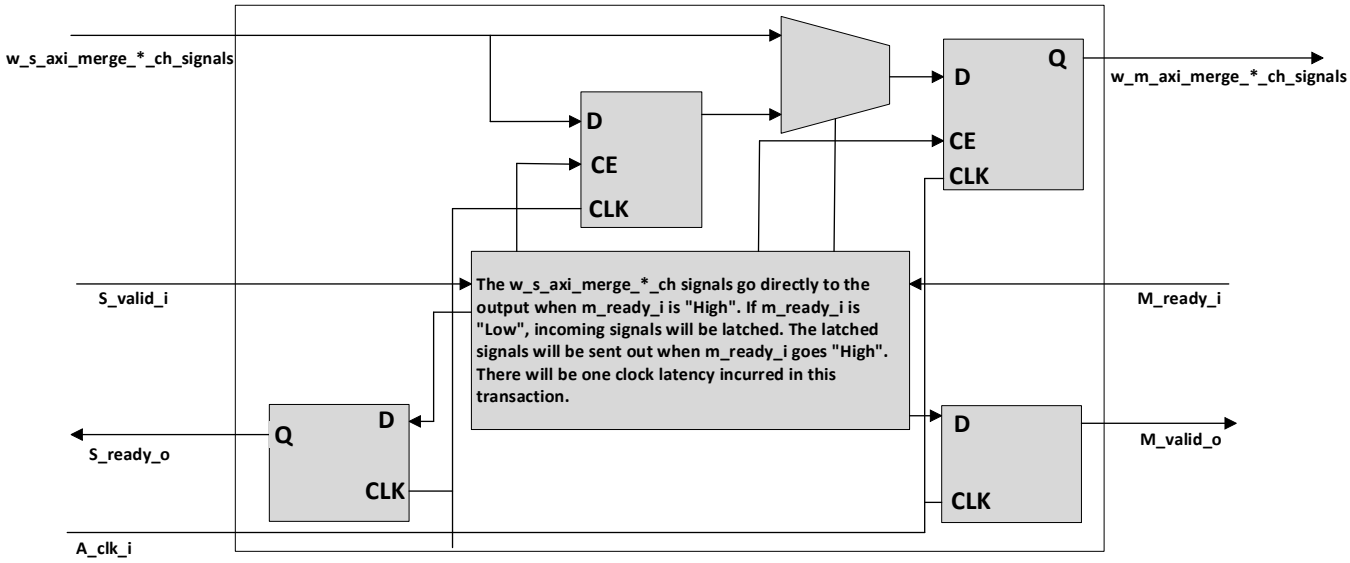


Figure 2.4. Full-Weight Mode

2.1.1.2. Light-Weight Mode

The Light-Weight mode is constructed by a single-stage pipeline register. This mode does not support back-to-back transfers and always incurs one bubble cycle for each transfer (maximum duty cycle of 50%), appropriate for write address, read address, and write response channels. The Light-Weight mode drives registered output, valid, and ready signal as shown in Figure 2.5. Refer to the Attributes Summary section for details on how to select the Light-Weight mode.

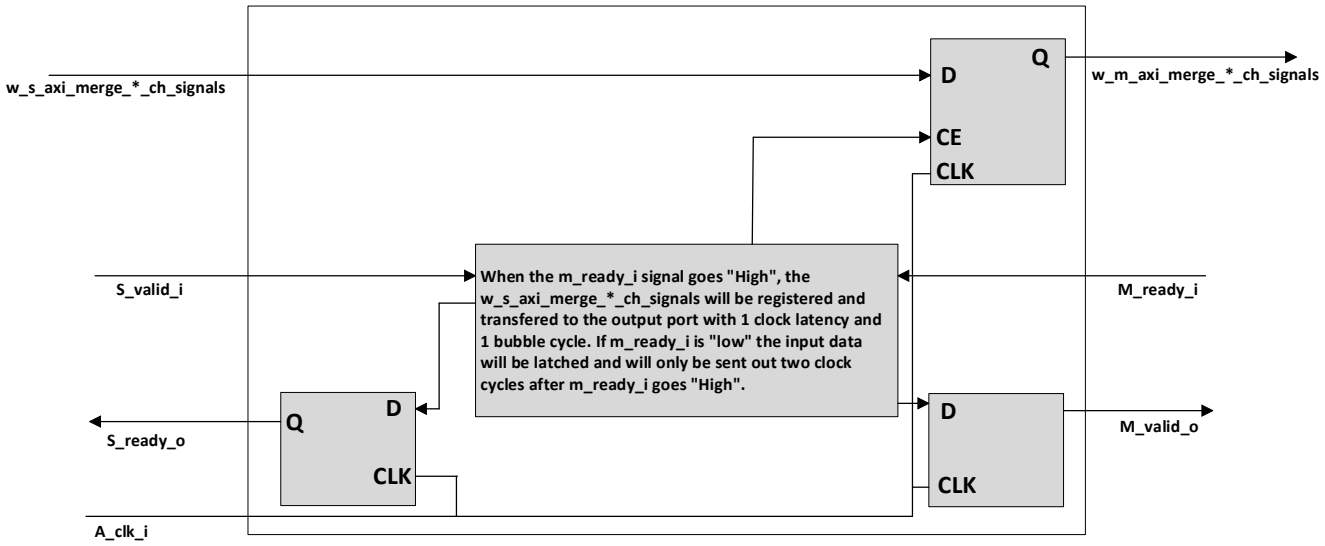


Figure 2.5. Light-Weight Mode

2.1.1.3. Input Registered Mode

The Input Registered mode is constructed with four-stage pipeline registers. This mode registers all input signals at the source before applying any throttling logic. The Input Registered mode supports back-to-back transfers (100% duty cycle) with one cycle of forward latency (but no bubble cycle) in the source-to-destination transfer and two cycles of latency in the destination to source transfer. This mode drives the registered ready output as shown in Figure 2.6. Refer to the Attributes Summary section for details on how to select the Input Registered mode.

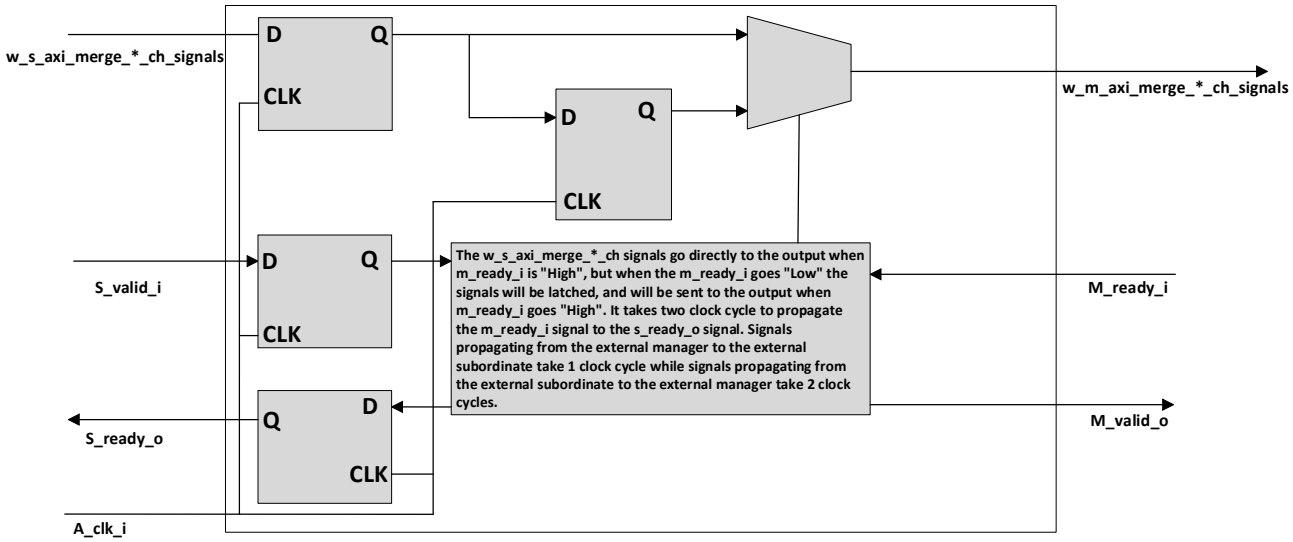


Figure 2.6. Input Registered Mode

Table 2.1. Summary of Latency and Utilization for Different Modes on a CertusPro-NX (LFCPNX-100) Device

Mode	Latency	Supported Protocol	Back-to-Back transfer	Resource Utilization (AXI4)	
				LUT4	PFU Registers
Full-Weight	1 clock cycle	AXI4, AXI3, AXI4-Lite	Supported	247	444
Light-Weight	2 clock cycles	AXI4, AXI3, AXI4-Lite	Not supported	20	227
Input Registered	1 clock cycle for manager to subordinate and 2 clock cycles for subordinate to manager.	AXI4, AXI3, AXI4-Lite	Supported	963	1121

Table 2.2. Summary of Latency and Utilization for Different Modes on a Lattice Avant-E Device (LAV-AT-E70ES1-1LFG676I)

Mode	Latency	Supported Protocol	Back-to-Back transfer	Resource Utilization (AXI4)	
				LUT4	PFU Registers
Full-Weight	1 clock cycle	AXI4, AXI3, AXI4-Lite	Supported	247	441
Light-Weight	2 clock cycle	AXI4, AXI3, AXI4-Lite	Not supported	20	226
Input Registered	1 clock cycle for manager to subordinate and 2 clock cycle for subordinate to manager.	AXI4, AXI3, AXI4-Lite	Supported	635	998

The sample vector to AXI interface performs merging and splitting operations similar to the AXI interface to sample vector block. The merged signals from the AXI interface to the sample vector block are split as the read data and write response signals are merged. The w_axi_merge_aw/w/ar_ch signals are split (which was merged in the AXI interface to sample vector block) into write address, data and read address, and are connected to the S_AXI interface. The read data and write response signals are merged into w_axi_merge_r/b_ch and are connected to the AXI slice block.

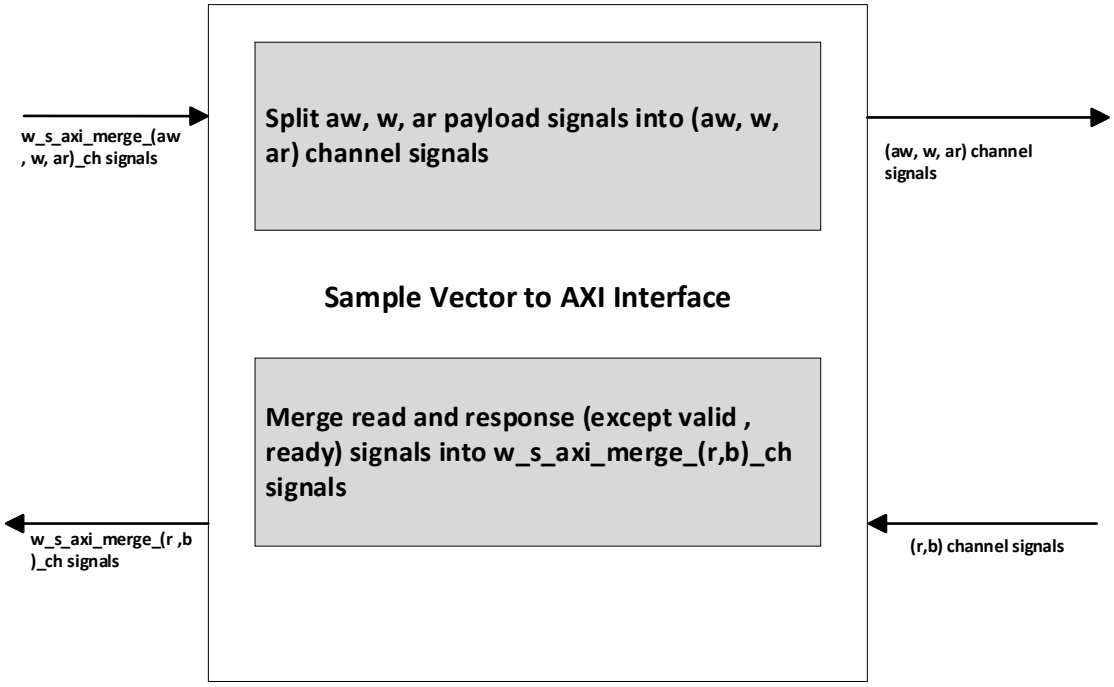


Figure 2.7. Sample Vector to AXI Interface Block Diagram

2.1.2. Clocking

The S_AXI and M_AXI interfaces are synchronized with the A_CLK_I input. The AXI Register Slice IP does not perform clock conversion.

The IP is synthesized in the Lattice Radiant software. Table 2.3, Table 2.4, and Table 2.5 list the maximum frequency of each mode for the different performance grades and AXI Protocols.

Table 2.3. Generated Fmax List for AXI4 on CertusPro-NX (LFCPNX100-7ASG256I)

7_High-Performance_1.0V (Fmax)		
Full-Weight Mode	Light-Weight mode	Input Registered Mode
403.22 MHz	403.22 MHz	253.16 MHz
8_High-Performance_1.0V (Fmax)		
Full-Weight Mode	Light-Weight mode	Input Registered Mode
454.54 MHz	500.00 MHz	285.71 MHz
9_High-Performance_1.0V (Fmax)		
Full-Weight Mode	Light-Weight mode	Input Registered Mode
500.00 MHz	555.55 MHz	344.82 MHz
7_Low-Power_1.0V (Fmax)		
Full-Weight Mode	Light-Weight mode	Input Registered Mode
312.5 MHz	317.46 MHz	227.27 MHz
8_Low-Power_1.0V (Fmax)		
Full-Weight Mode	Light-Weight mode	Input Registered Mode
357.14 MHz	370.37 MHz	238.09 MHz
9_Low-Power_1.0V (Fmax)		
Full-Weight Mode	Light-Weight mode	Input Registered Mode
400 MHz	408.16 MHz	294.12 MHz

Table 2.4. Generated Fmax List for AXI3 on CertusPro-NX (LFCPNX100-7ASG256I)

7_High-Performance_1.0V (Fmax)		
Full-Weight Mode	Light-Weight mode	Input Registered Mode
344.83 MHz	350.88 MHz	263.15 MHz
8_High-Performance_1.0V (Fmax)		
Full-Weight Mode	Light-Weight mode	Input Registered Mode
434.78 MHz	444.44 MHz	344.83 MHz
9_High-Performance_1.0V (Fmax)		
Full-Weight Mode	Light-Weight mode	Input Registered Mode
500.00 MHz	487.80 MHz	384.62 MHz
7_Low-Power_1.0V (Fmax)		
Full-Weight Mode	Light-Weight mode	Input Registered Mode
333.33 MHz	327.86 MHz	200.00 MHz
8_Low-Power_1.0V (Fmax)		
Full-Weight Mode	Light-Weight mode	Input Registered Mode
357.14 MHz	384.62 MHz	227.27 MHz
9_Low-Power_1.0V (Fmax)		
Full-Weight Mode	Light-Weight mode	Input Registered Mode
400.00 MHz	434.78 MHz	285.71 MHz

Table 2.5. Generated Fmax List for AXI4-Lite on CertusPro-NX (LFCPNX100-7ASG256I)

7_High-Performance_1.0V (Fmax)		
Full-Weight Mode	Light-Weight mode	Input Registered Mode
357.14 MHz	454.54 MHz	263.16 MHz
8_High-Performance_1.0V (Fmax)		
Full-Weight Mode	Light-Weight mode	Input Registered Mode
434.78 MHz	476.19 MHz	344.83 MHz
9_High-Performance_1.0V (Fmax)		
Full-Weight Mode	Light-Weight mode	Input Registered Mode
526.32 MHz	555.55 MHz	416.66 MHz
7_Low-Power_1.0V (Fmax)		
Full-Weight Mode	Light-Weight mode	Input Registered Mode
312.50 MHz	322.58 MHz	232.56 MHz
8_Low-Power_1.0V (Fmax)		
Full-Weight Mode	Light-Weight mode	Input Registered Mode
357.14 MHz	400.00 MHz	270.27 MHz
9_Low-Power_1.0V (Fmax)		
Full-Weight Mode	Light-Weight mode	Input Registered Mode
416.66 MHz	425.53 MHz	312.50 MHz

2.1.3. Reset

The A_RESET_N input can be tied to High (inactive) if "soft" reset is not required. AXI handshake outputs remain deasserted until the device recovers from power-on reset, at which time handshake inputs are sampled. The behavioral simulation initializes to the same state as the device following power-on.

The A_RESET_N input, if used, must be synchronized with A_CLK_I.

The IP deasserts all valid and ready outputs after A_RESET_N is sampled active, and for the duration of the A_RESET_N pulse. The AXI protocol requires that the connected manager also deasserts all valid outputs during reset (until after

A_RESET_N is sampled inactive). Subordinates must not assert response-channel valid outputs until after they receive a command from a manager. It is strongly recommended to deassert the ready outputs of the target IP until after reset to avoid inadvertently signaling a transfer completion in case a connected IP recovers from reset during an earlier cycle and asserts are valid.

In compliance with the AXI standard, the reset can be asynchronously deasserted, but the deassertion of the reset is synchronous. Hence, there is an internal synchronous mechanism that aligns the deassertion edge with the clock.

2.2. Signal Description

The port list is given in the following table:

Table 2.6. AXI Register Slice Signal Description

Port name	IO	Width	Description
Global Clock Reset Signals			
a_clk_i	Input	1	Design clock and register interface clock.
a_reset_n_i	Input	1	Asynchronous active low reset.
Subordinate IO Signals			
s_axi_awid_i	Input	*AXI_ID_WIDTH*	Write address id. This signal is the identification tag for the write address group of signals.
s_axi_awaddr_i	Input	*AXI_ADDR_WIDTH*	Write address. The write address gives the address of the first transfer in a write burst transaction.
s_axi_awlen_i	Input	AXI4: 8 AXI3: 4	Burst length. The burst length gives the exact number of transfers in a burst. This information determines the number of data transfers associated with the address. This changes between AXI3 and AXI4.
s_axi_awsize_i	Input	3	Burst size. This signal indicates the size of each transfer in the burst. See burst size.
s_axi_awburst_i	Input	2	Burst type. The burst type and the size information, determine how the address for each transfer within the burst is calculated.
s_axi_awlock_i	Input	AXI4: 1 AXI3: 2	Lock type. Provides additional information about the atomic characteristics of the transfer. This changes between AXI3 and AXI4.
s_axi_awcache_i	Input	4	Memory type. This signal indicates how transactions are required to progress through a system.
s_axi_awprot_i	Input	3	Protection type. This signal indicates the privilege and security level of the transaction, and whether the transaction is a data access or an instruction access.
s_axi_awqos_i	Input	4	Quality of service, qos. The qos identifier is sent for each write transaction. Implemented only in AXI4.
s_axi_awregion_i	Input	4	Region identifier. Permits a single physical interface on a target to be used for multiple logical interfaces. Implemented only in AXI4.
s_axi_awuser_i	Input	*AXI_AWUSER_WIDTH*	User signal. Optional user-defined signal in the write address channel. Supported only in AXI4.
s_axi_awvalid_i	Input	1	Write address valid. This signal indicates that the channel is signaling valid write address and control information.
s_axi_awready_o	Output	1	Write address ready. This signal indicates that the target is ready to accept an address and associated control signals.
s_axi_wid_i	Input	4	Write id tag. This signal is the id tag of the write data transfer. Supported only in AXI3.
s_axi_wdata_i	Input	*AXI_DATA_WIDTH*	Write data.
s_axi_wstrb_i	Input	*AXI_DATA_WIDTH/8*	Write strobes. This signal indicates which byte lanes hold valid data. There is one write strobe bit for each eight bits of the write data bus.

Port name	IO	Width	Description
s_axi_wlast_i	Input	1	Write last. This signal indicates the last transfer in a write burst.
s_axi_wuser_i	Input	*AXI_WUSER_WIDTH*	User signal. Optional user-defined signal in the write data channel. Supported only in AXI4.
s_axi_wvalid_i	Input	1	Write valid. This signal indicates that valid write data and strobes are available.
s_axi_wready_o	Output	1	Write ready. This signal indicates that the target can accept the write data.
s_axi_bid_o	Output	*AXI_ID_WIDTH*	Response id tag. This signal is the id tag of the write response.
s_axi_bresp_o	Output	2	Write response. This signal indicates the status of the write transaction.
s_axi_buser_o	Output	*AXI_BUSER_WIDTH*	User signal. Optional user-defined signal in the write response channel.
s_axi_bvalid_o	Output	1	Write response valid. This signal indicates that the channel is signaling a valid write response.
s_axi_bready_i	Input	1	Response ready. This signal indicates that the leader can accept a write response.
s_axi_arid_i	Input	*AXI_ID_WIDTH*	Read address id. This signal is the identification tag for the read address group of signals.
s_axi_araddr_i	Input	*AXI_ADDR_WIDTH*	Read address. The read address gives the address of the first transfer in a read burst transaction.
s_axi_arlen_i	Input	AXI4: 8 AXI3: 4	Burst length. This signal indicates the exact number of transfers in a burst. This changes between AXI3 and AXI4.
s_axi_arsize_i	Input	3	Burst size. This signal indicates the size of each transfer in the burst.
s_axi_arburst_i	Input	2	Burst type. The burst type and the size information determine how the address for each transfer within the burst is calculated.
s_axi_arlock_i	Input	AXI4: 1 AXI3: 2	Lock type. This signal provides additional information about the atomic characteristics of the transfer. This changes between AXI3 and AXI4.
s_axi_arcache_i	Input	4	Memory type. This signal indicates how transactions are required to progress through a system.
s_axi_arprot_i	Input	3	Protection type. This signal indicates the privilege and security level of the transaction, and whether the transaction is a data access or an instruction access.
s_axi_arregion_i	Input	4	Region identifier. Permits a single physical interface on a target to be used for multiple logical interfaces. Implemented only in AXI4.
s_axi_arqos_i	Input	4	Quality of service, qos. The qos identifier is sent for each read transaction. Implemented only in AXI4.
s_axi_aruser_i	Input	*AXI_ARUSER_WIDTH*	User signal. Optional user-defined signal in the read address channel. Supported only in AXI4.
s_axi_arvalid_i	Input	1	Read address valid. This signal indicates that the channel is signaling valid read address and control information.
s_axi_arready_o	Output	1	Read address ready. This signal indicates that the target is ready to accept an address and associated control signals.
s_axi_rid_o	Output	*AXI_ID_WIDTH*	Read id tag. This signal is the identification tag for the read data group of signals generated by the target.
s_axi_rdata_o	Output	*AXI_DATA_WIDTH*	Read data.

Port name	IO	Width	Description
s_axi_rresp_o	Output	2	Read response. This signal indicates the status of the read transfer.
s_axi_rlast_o	Output	1	Read last. This signal indicates the last transfer in a read burst.
s_axi_ruser_o	Output	*AXI_RUSER_WIDTH*	User signal. Optional user-defined signal in the read data channel.
s_axi_rvalid_o	Output	1	Read valid. This signal indicates that the channel is signaling the required read data.
s_axi_rready_i	Input	1	Read ready. This signal indicates that the leader can accept the read data and response information.
Manager IO Signals			
m_axi_awid_o	Output	*AXI_ID_WIDTH*	Write address id. This signal is the identification tag for the write address group of signals.
m_axi_awaddr_o	Output	*AXI_ADDR_WIDTH*	Write address. The write address gives the address of the first transfer in a write burst transaction.
m_axi_awlen_o	Output	AXI4: 8 AXI3: 4	Burst length. The burst length gives the exact number of transfers in a burst. This information determines the number of data transfers associated with the address. This changes between AXI3 and AXI4.
m_axi_awsize_o	Output	3	Burst size. This signal indicates the size of each transfer in the burst. See burst size.
m_axi_awburst_o	Output	2	Burst type. The burst type and the size information, determine how the address for each transfer within the burst is calculated.
m_axi_awlock_o	Output	AXI4: 1 AXI3: 2	Lock type. Provides additional information about the atomic characteristics of the transfer. This changes between AXI3 and AXI4.
m_axi_awcache_o	Output	4	Memory type. This signal indicates how transactions are required to progress through a system.
m_axi_awprot_o	Output	3	Protection type. This signal indicates the privilege and security level of the transaction, and whether the transaction is a data access or an instruction access.
m_axi_awqos_o	Output	4	Quality of service, qos. The qos identifier is sent for each write transaction. Implemented only in AXI4.
m_axi_awregion_o	Output	4	Region identifier. Permits a single physical interface on a target to be used for multiple logical interfaces. Implemented only in AXI4.
m_axi_awuser_o	Output	*AXI_AWUSER_WIDTH*	User signal. Optional user-defined signal in the write address channel. Supported only in AXI4.
m_axi_awvalid_o	Output	1	Write address valid. This signal indicates that the channel is signaling valid write address and control information.
m_axi_awready_i	Input	1	Write address ready. This signal indicates that the target is ready to accept an address and associated control signals.
m_axi_wid_o	Output	4	Write id tag. This signal is the id tag of the write data transfer.
m_axi_wdata_o	Output	*AXI_DATA_WIDTH*	Write data.
m_axi_wstrb_o	Output	*AXI_DATA_WIDTH*/8	Write strobes. This signal indicates which byte lanes hold valid data. There is one write strobe bit for each eight bits of the write data bus.
m_axi_wlast_o	Output	1	Write last. This signal indicates the last transfer in a write burst.
m_axi_wuser_o	Output	*AXI_WUSER_WIDTH*	User signal. Optional user-defined signal in the write data channel. Supported only in AXI4.

Port name	IO	Width	Description
m_axi_wvalid_o	Output	1	Write valid. This signal indicates that valid write data and strobes are available.
m_axi_wready_i	Input	1	Write ready. This signal indicates that the target can accept the write data.
m_axi_bid_i	Input	*AXI_ID_WIDTH*	Response id tag. This signal is the id tag of the write response.
m_axi_bres_i	Input	2	Write response. This signal indicates the status of the write transaction.
m_axi_buser_i	Input	*AXI_BUSER_WIDTH*	User signal. Optional user-defined signal in the write response channel.
m_axi_bvalid_i	Input	1	Write response valid. This signal indicates that the channel is signaling a valid write response.
m_axi_bready_o	Output	1	Response ready. This signal indicates that the leader can accept a write response.
m_axi_arid_o	Output	*AXI_ID_WIDTH*	Read address id. This signal is the identification tag for the read address group of signals.
m_axi_araddr_o	Output	*AXI_ADDR_WIDTH*	Read address. The read address gives the address of the first transfer in a read burst transaction.
m_axi_arlen_o	Output	AXI4: 8 AXI3: 4	Burst length. This signal indicates the exact number of transfers in a burst. This changes between AXI3 and AXI4.
m_axi_arsize_o	Output	3	Burst size. This signal indicates the size of each transfer in the burst.
m_axi_arburst_o	Output	2	Burst type. The burst type and the size information determine how the address for each transfer within the burst is calculated.
m_axi_arlock_o	Output	AXI4: 1 AXI3: 2	Lock type. This signal provides additional information about the atomic characteristics of the transfer. This changes between AXI3 and AXI4.
m_axi_arcache_o	Output	4	Memory type. This signal indicates how transactions are required to progress through a system.
m_axi_arprot_o	Output	3	Protection type. This signal indicates the privilege and security level of the transaction, and whether the transaction is a data access or an instruction access.
m_axi_arregion_o	Output	4	Region identifier. Permits a single physical interface on a target to be used for multiple logical interfaces. Implemented only in AXI4.
m_axi_arqos_o	Output	4	Quality of service, qos. Qos identifier is sent for each read transaction. Implemented only in AXI4.
m_axi_aruser_o	Output	*AXI_ARUSER_WIDTH*	User signal. Optional user-defined signal in the read address channel. Supported only in AXI4.
m_axi_arvalid_o	Output	1	Read address valid. This signal indicates that the channel is signaling valid read address and control information.
m_axi_arready_i	Input	1	Read address ready. This signal indicates that the target is ready to accept an address and associated control signals.
m_axi_rid_i	Input	*AXI_ID_WIDTH*	Read id tag. This signal is the identification tag for the read data group of signals generated by the target.
m_axi_rdata_i	Input	*AXI_DATA_WIDTH*	Read data.
m_axi_rresp_i	Input	2	Read response. This signal indicates the status of the read transfer.
m_axi_rlast_i	Input	1	Read last. This signal indicates the last transfer in a read burst.

Port name	IO	Width	Description
m_axi_ruser_i	Input	*AXI_RUSER_WIDTH*	User signal. Optional user-defined signal in the read data channel.
m_axi_rvalid_i	Input	1	Read valid. This signal indicates that the channel is signaling the required read data.
m_axi_rready_o	Output	1	Read ready. This signal indicates that the leader can accept the read data and response information.

2.3. Attribute Summary

The configurable attributes of the AXI register slice IP core are shown in Table 2.7.

These attributes can be configured through the IP Catalog's Module/IP wizard of the Lattice Propel/Radiant Builder.

Table 2.7. Attributes Table

Attribute	Select-Able Values	Default	Allowable Values
General			
AXI_PROTOCOL	Editable	0	0 => AXI4 1 => AXI3 2 => AXI4-Lite
AXI_ID_WIDTH	Editable	4	Width of all id signals propagated by the core. (0-32)
AXI_ADDR_WIDTH	Editable	32	For AXI4 or AXI3: Integer (12-64); For AXI4-Lite: Integer (1-64)
AXI_DATA_WIDTH	Editable	32	For AXI4 or AXI3: Integer (32, 64, 128, 256, 512, 1024); For AXI4-Lite: Integer (32, 64)
AXI_SUPPORTS_USER_SIGNALS	Editable	0	User signals only support in AXI4. 1 for AXI4 and 0 for others.
AXI_AWUSER_WIDTH	Editable	1	Write address user signal (only supports in AXI4).
AXI_ARUSER_WIDTH	Editable	1	Read address user signal (only supports in AXI4).
AXI_WUSER_WIDTH	Editable	1	Write data user signal (only supports in AXI4).
AXI_RUSER_WIDTH	Editable	1	Read data user signal (only supports in AXI4).
AXI_BUSER_WIDTH	Editable	1	Write response user signal (only supports in AXI4).
REG_CONFIG_AW	Editable	0	Mode of operation 0 => Full-Weight Mode 1 => Light-Weight Mode 2 => Input Registered Mode
REG_CONFIG_W	Editable	0	Mode of operation 0 => Full-Weight Mode 1 => Light-Weight Mode 2 => Input Registered Mode
REG_CONFIG_B	Editable	0	Mode of operation 0 => Full-Weight Mode 1 => Light-Weight Mode 2 => Input Registered Mode
REG_CONFIG_AR	Editable	0	Mode of operation 0 => Full-Weight Mode 1 => Light-Weight Mode 2 => Input Registered Mode

Attribute	Select-Able Values	Default	Allowable Values
REG_CONFIG_R	Editable	0	Mode of operation 0 => Full-Weight Mode 1 => Light-Weight Mode 2 => Input Registered Mode

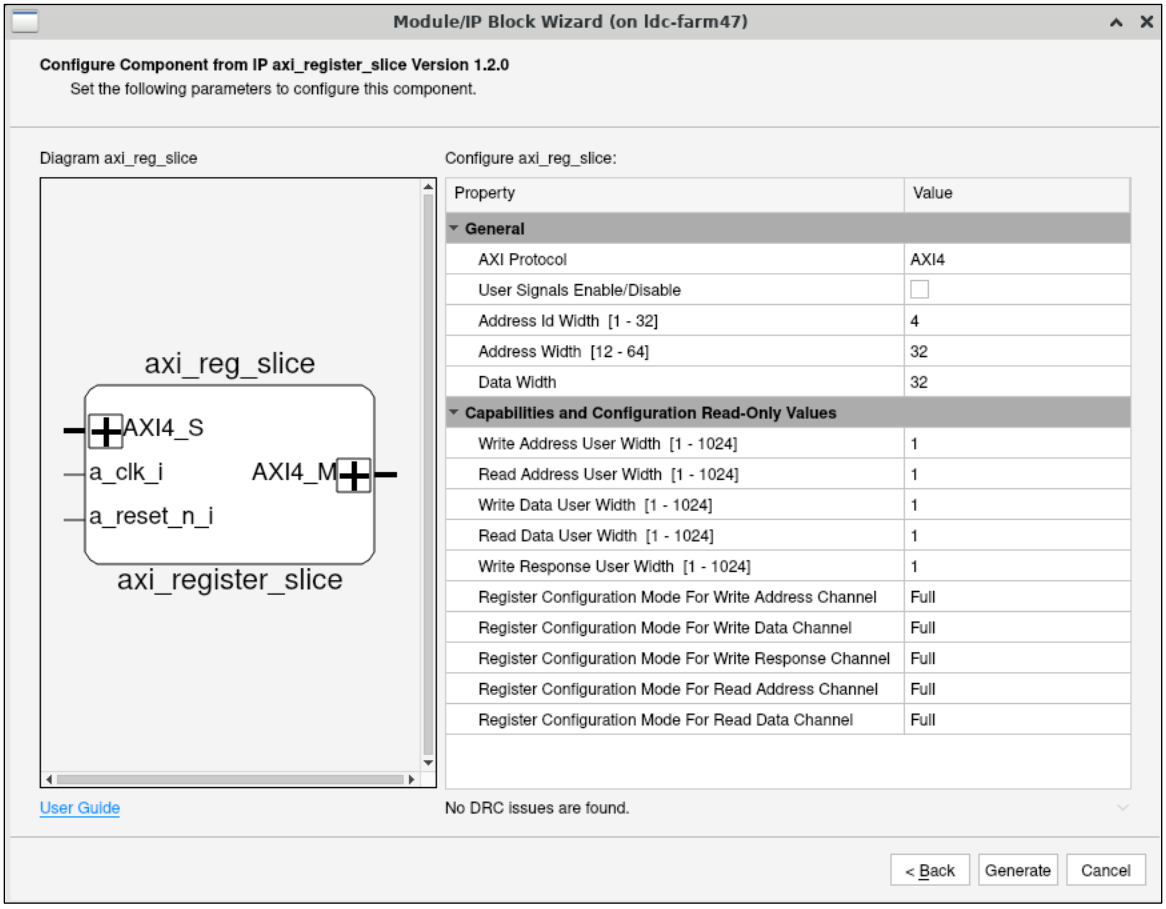


Figure 2.8. AXI Register Slice IP View

2.4. Data Format

The AXI3, AXI4, and AXI4-L interfaces support standard data format as defined in the AMBA (Advanced Microcontroller Bus Architecture) specification.

Refer to relevant user guides from ARM AMBA for more information on the protocol.

2.5. Timing Diagrams

2.5.1. Full-Weight Mode

The Full-Weight mode supports back-to-back transfers (100% duty cycle) with one cycle of forward latency (no bubble cycle), appropriate for write and read channels of AXI4 or AXI3 burst transfers.

Figure 2.9. AXI Write Transfer in Full-Weight Mode

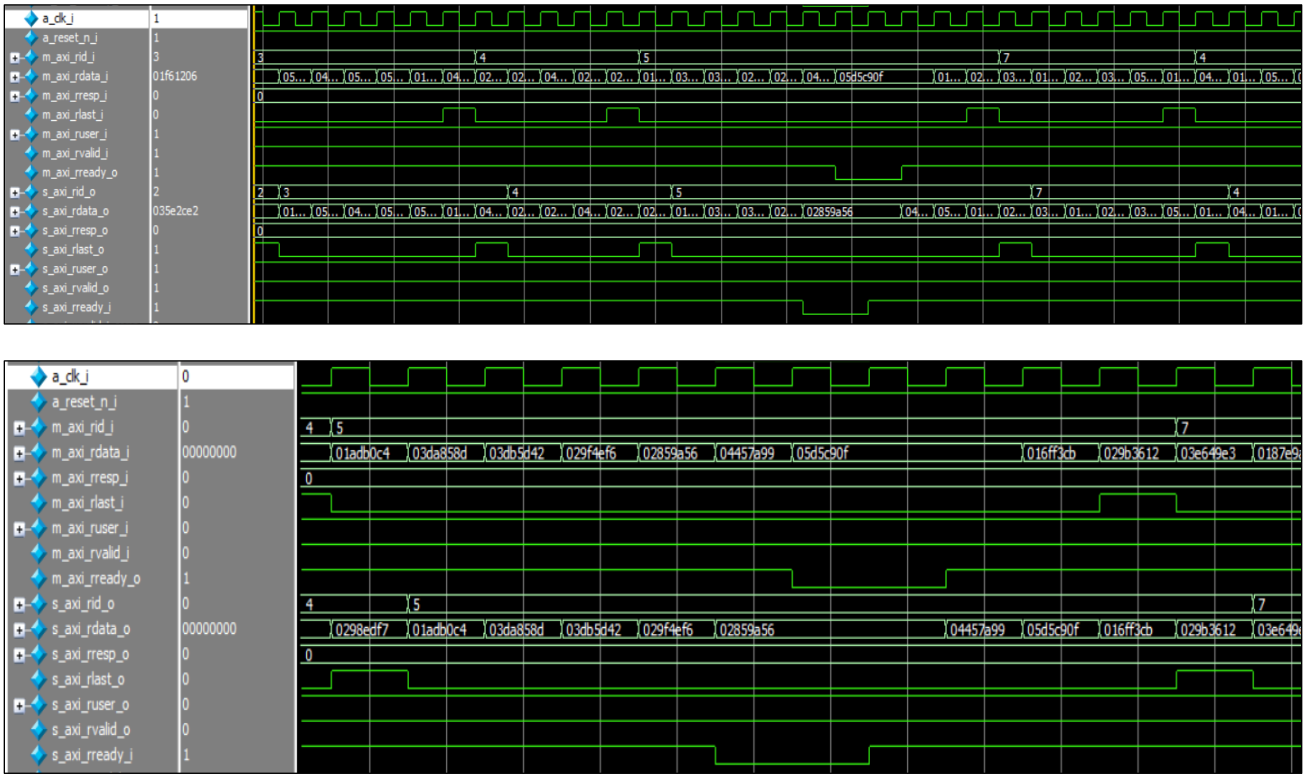


Figure 2.10. AXI Read Transfer in Full-Weight Mode

2.5.2. Light-Weight Mode

The Light-Weight mode does not support back-to-back transfers and always incurs one bubble cycle of each transfer (maximum duty cycle of 50%). Appropriate for write address, read address, and write response channels.

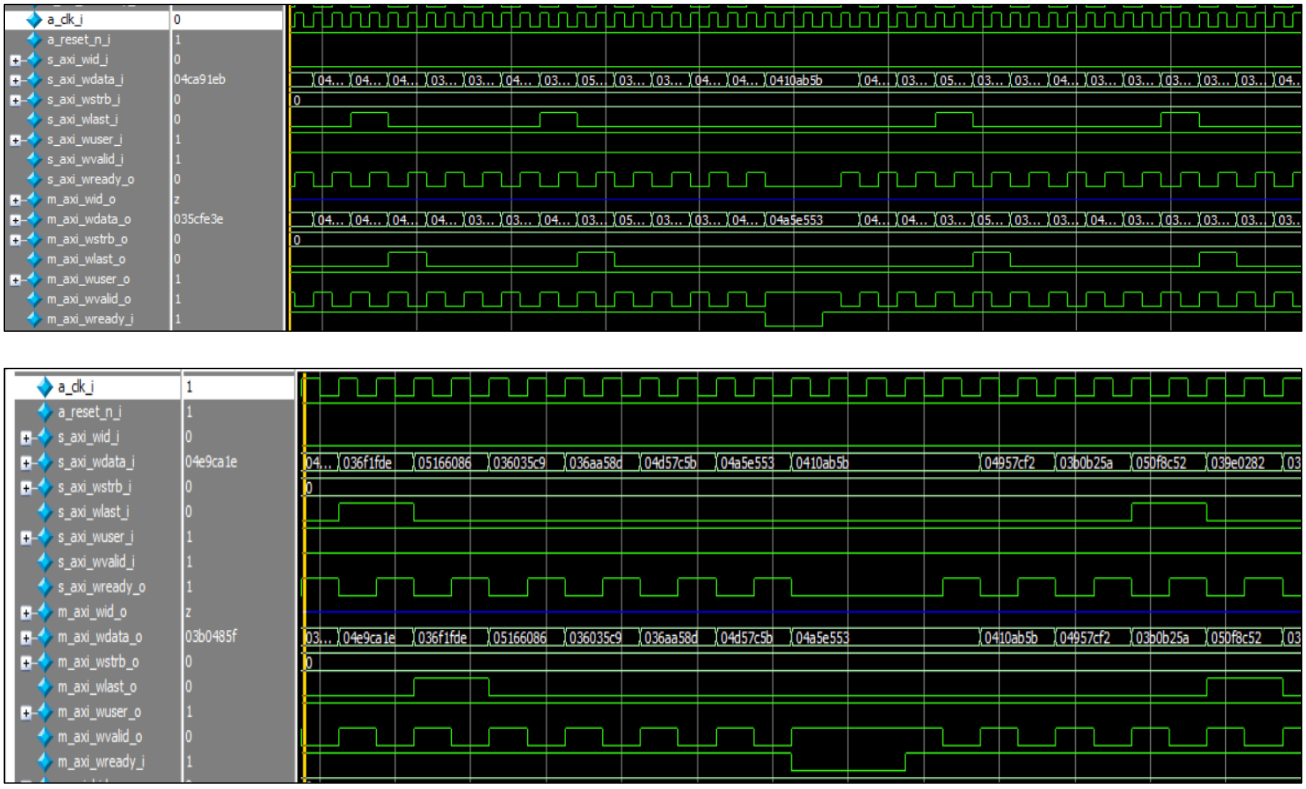
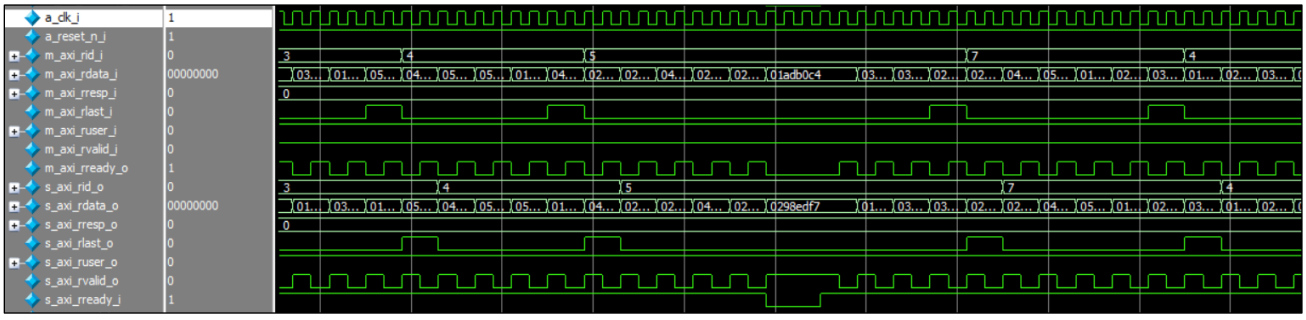


Figure 2.11. AXI Write Transfer in Light-Weight Mode



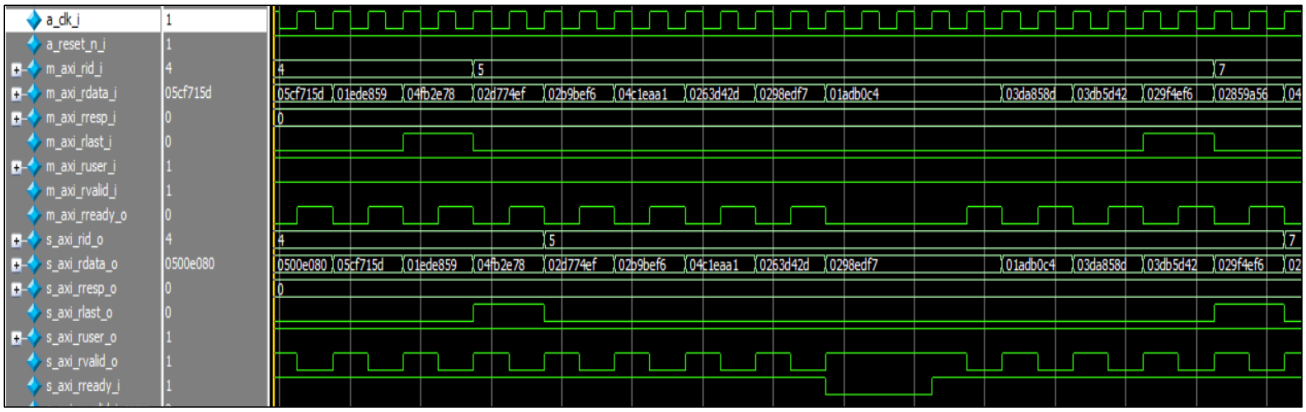


Figure 2.12. AXI Read Transfer in Light-Weight Mode

2.5.3. Input Registered Mode

The Input Registered mode is similar to the Full-Weight mode. The mode supports back-to-back transfers (100% duty cycle) with one cycle of forward latency (but no bubble cycle) in the manager-to-subordinate transfer and two cycles of latency in the subordinate-to-manager transfer.

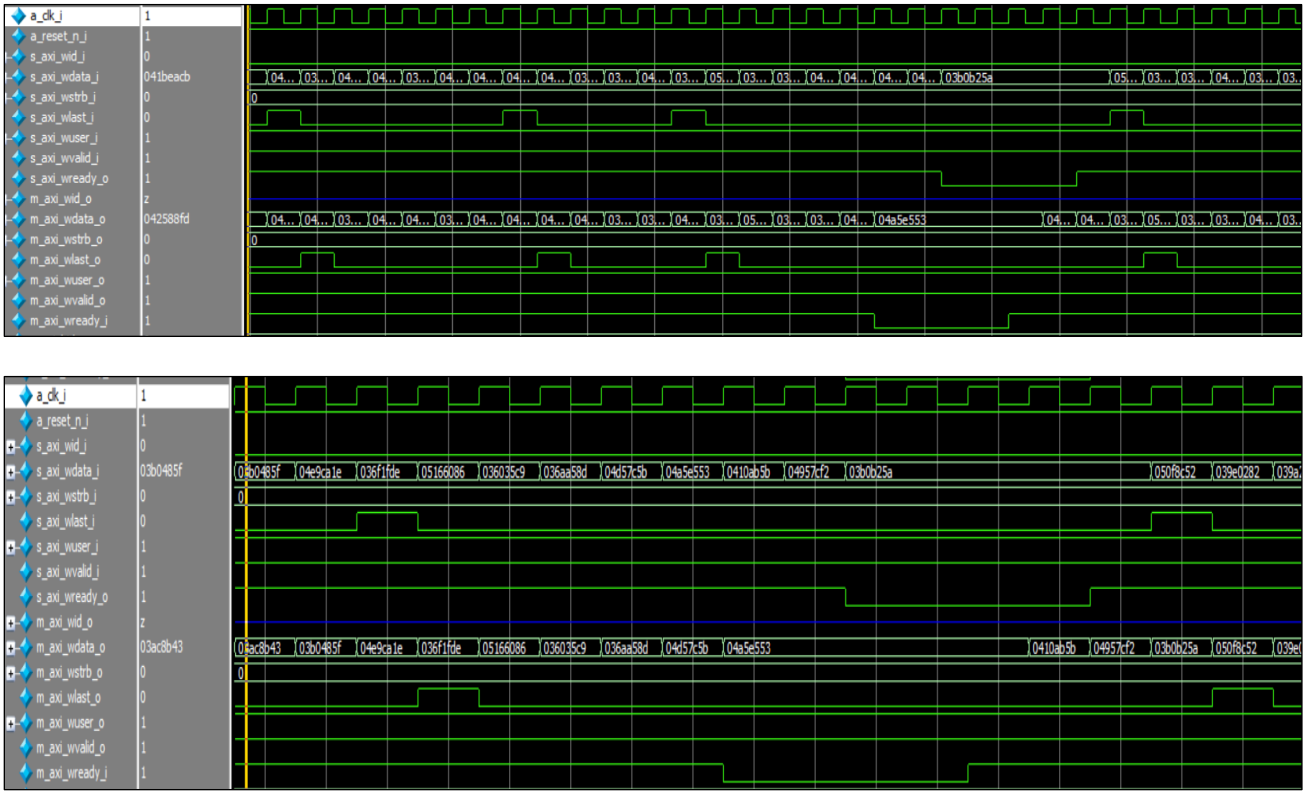


Figure 2.13. AXI Write Transfer in Input Registered Mode

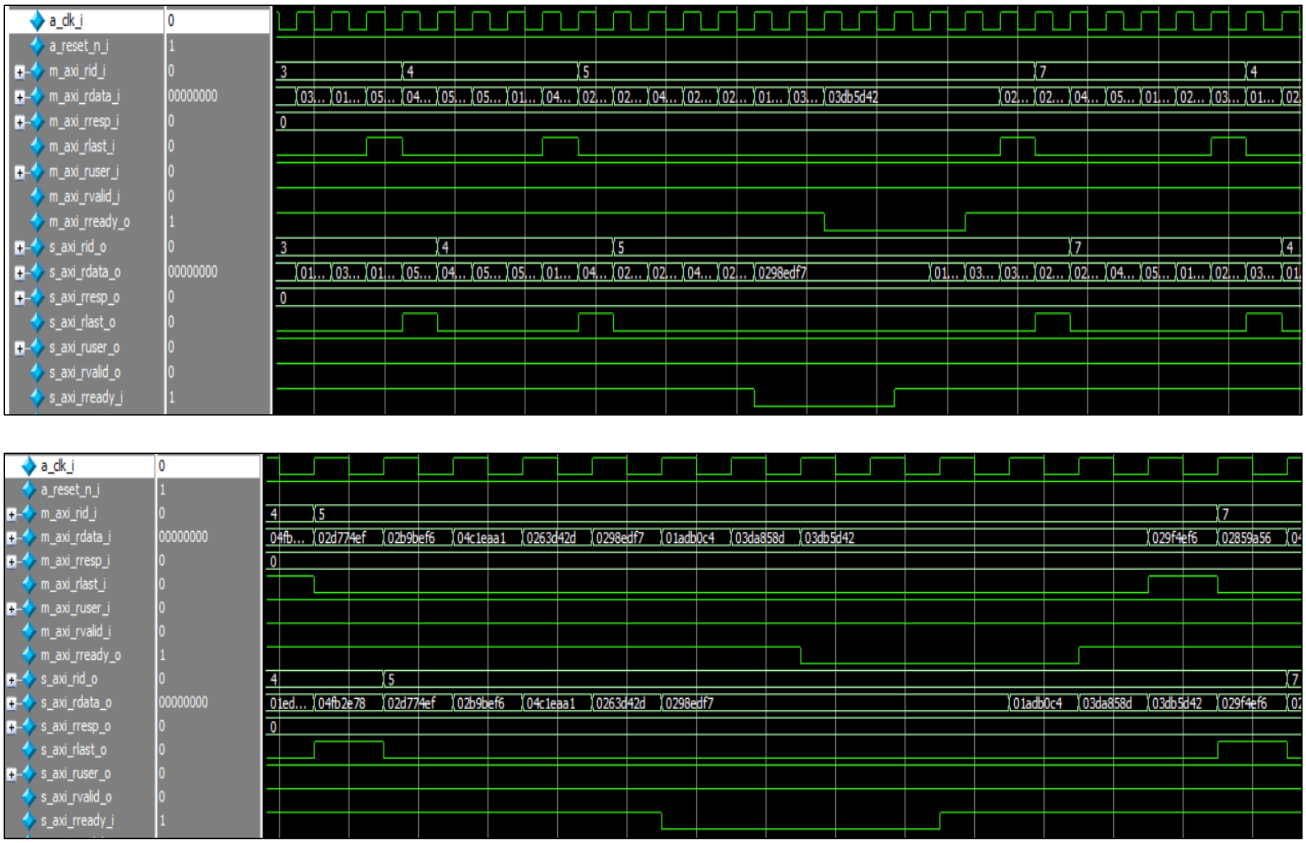
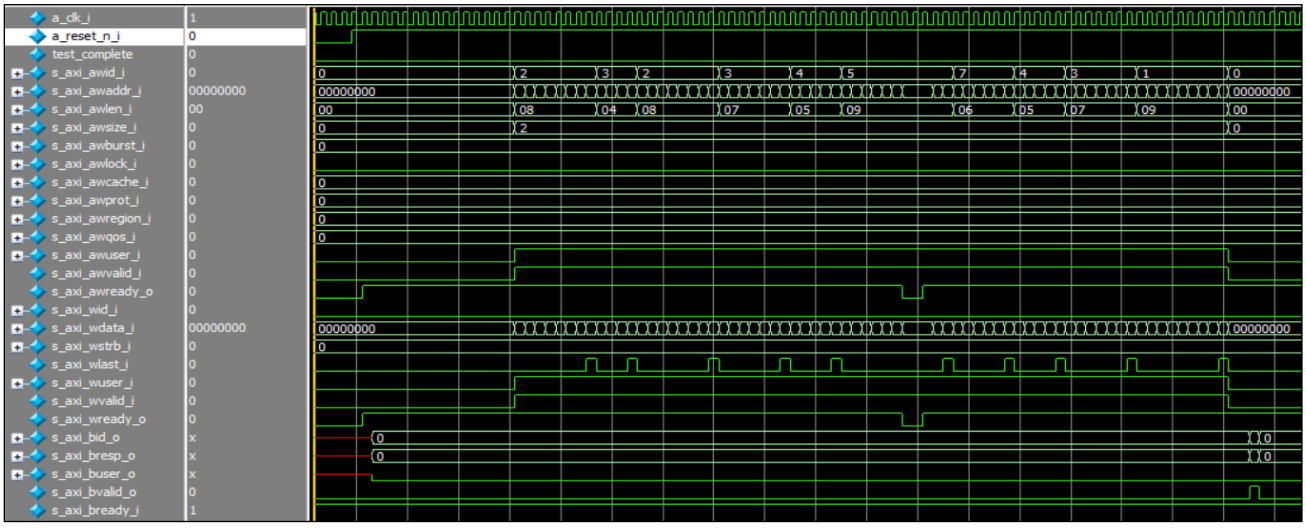


Figure 2.14. AXI Read Transfer in Input Registered Mode

2.5.4. Data Flow

The data flow is explained below:

1. Write address goes with the write data through subordinate port.
2. Write response from the manager port.
3. Read the address through subordinate port.
4. Data and response through manager port.



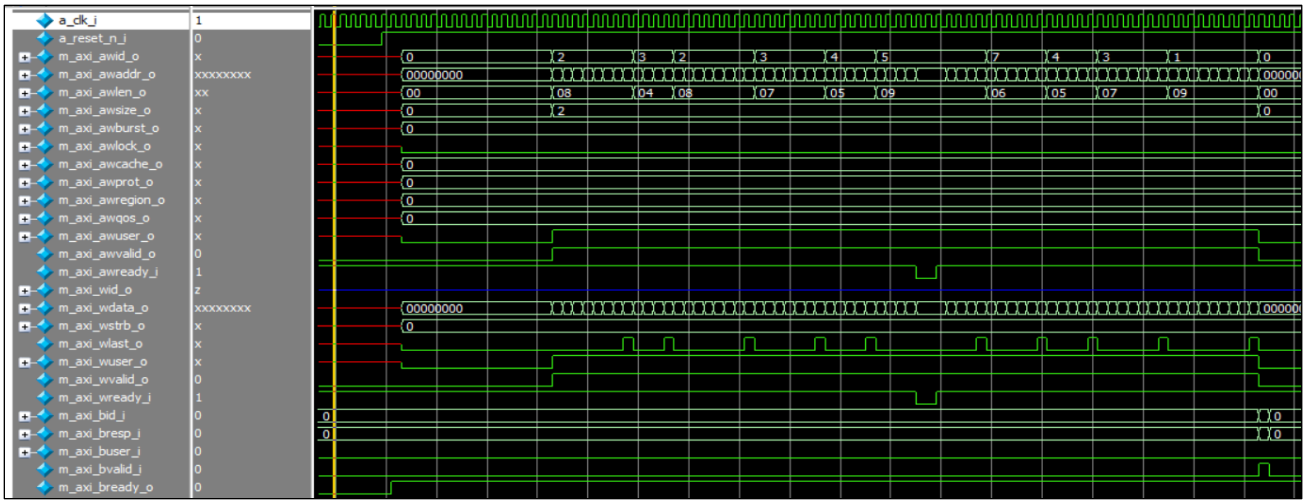
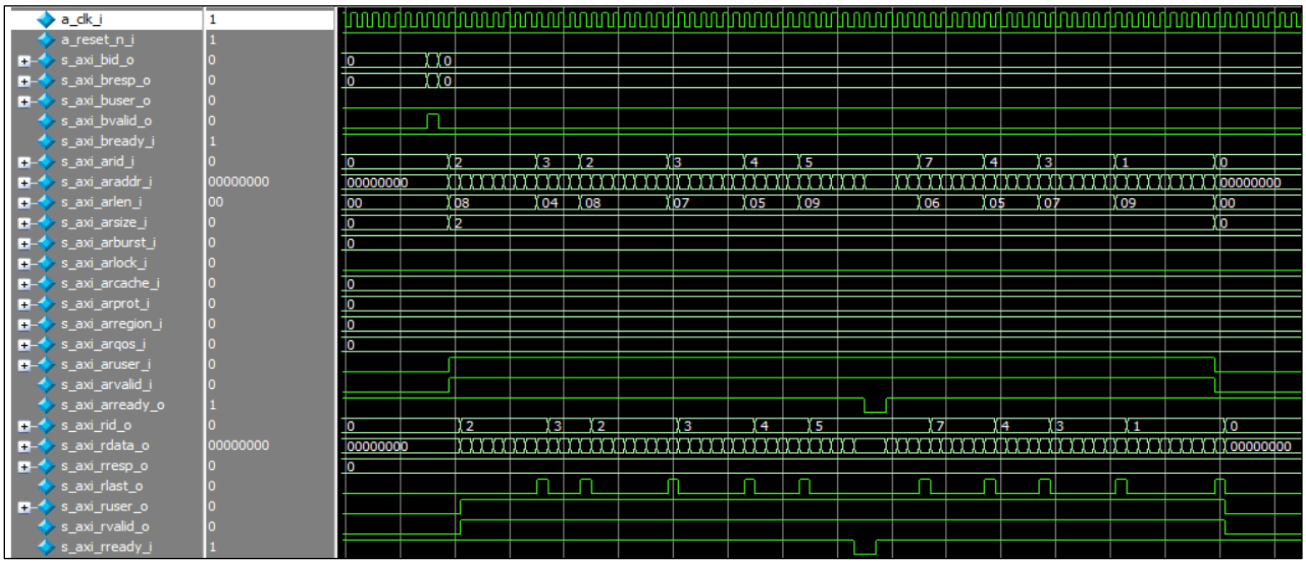


Figure 2.15. Basic AXI AW, W, B Channel Data flow



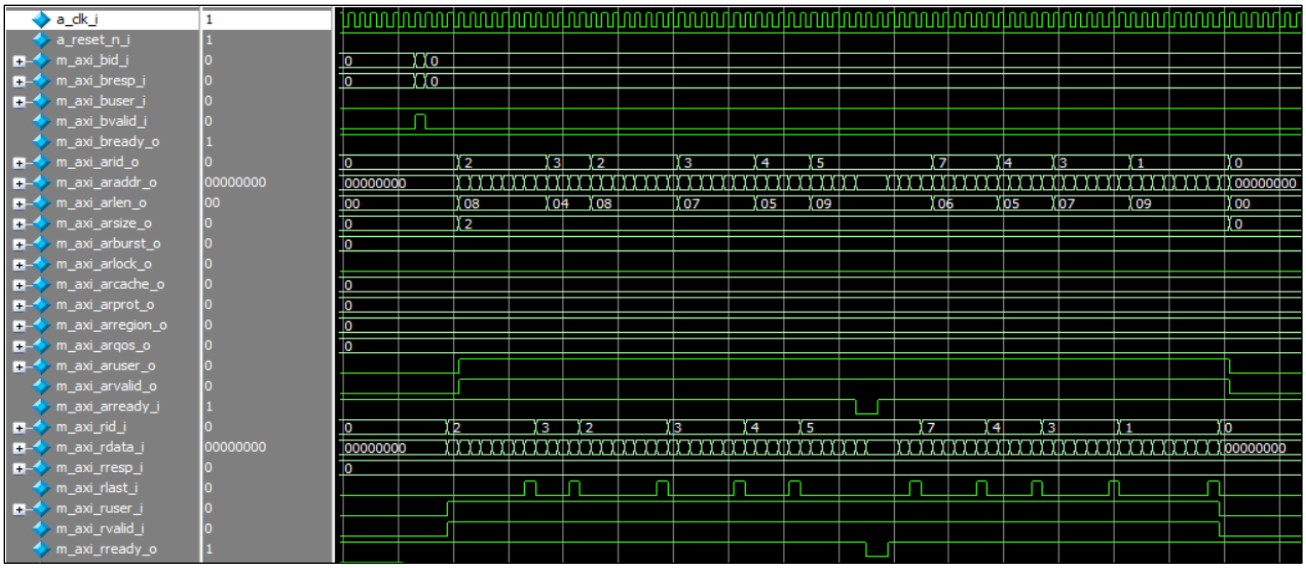


Figure 2.16. Basic AXI AR, R Channel Data Flow

2.6. IP Generation, Simulation, and Validation

This section provides information on how to generate the AXI register slice IP using the Lattice Propel Builder software, and how to run simulation. For more details on the Lattice Propel Builder software, refer to the Lattice Propel Builder user guide.

2.6.1. Generating the IP and Running Functional Simulation

The Lattice Propel Builder software allows user to customize and generate modules and IPs and integrate them into a system. Simulation will be launched through the Lattice Radiant software. This IP cannot be generated using the Lattice Radiant software.

To generate this IP using the Propel Builder software, launch the Propel Builder and create a new project:

1. Click on **New Design**, select SoC Project as **Type** and fill up the **Name** and **Location** fields accordingly.

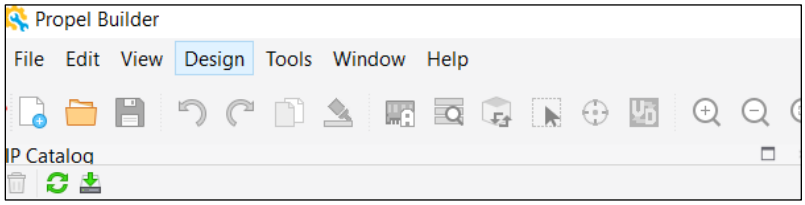


Figure 2.17. Toolbar

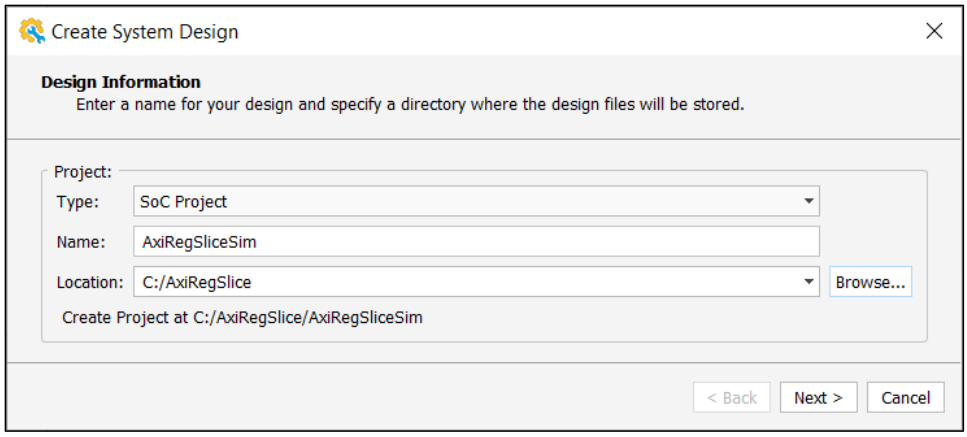


Figure 2.18. Creating New Design

2. Click **Next**.
3. Select **Empty Project** and click **Next**.

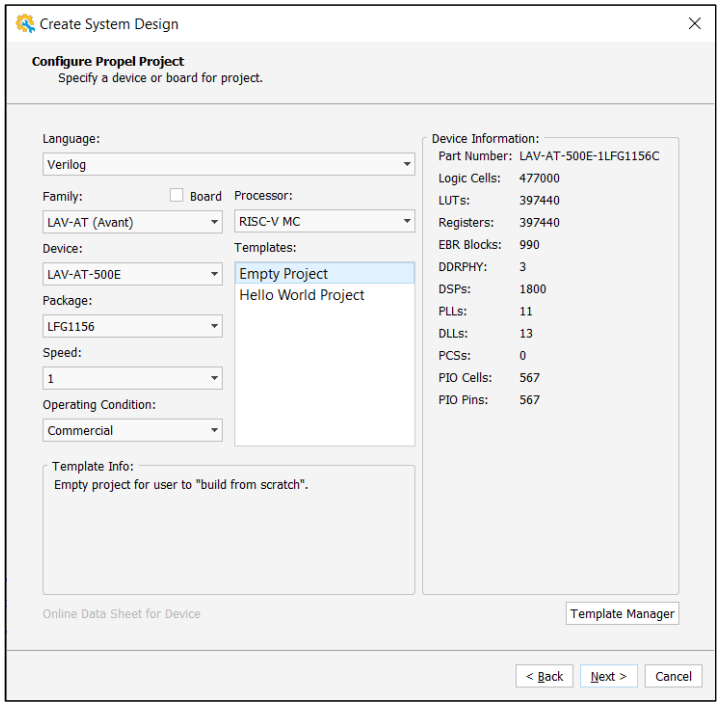


Figure 2.19. Configuring Propel Project Settings

4. Then, click **Finish**.
5. The user can see a blank design canvas. Delete clk_i and rstn_i ports.

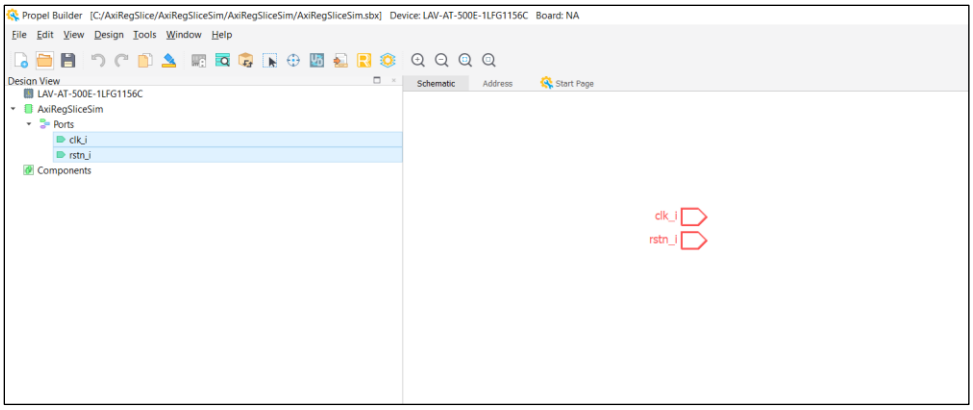


Figure 2.20. AXI Register Slice

6. Click on the **IP Catalog** tab and double click on **AXI Register Slice** to bring up the configuration dialog as shown in Figure 2.21.

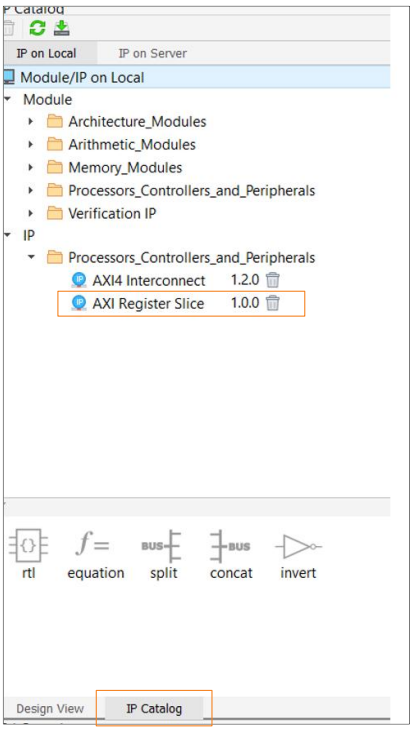


Figure 2.21. IP Catalog Settings

7. Enter a name and configure the IP accordingly.

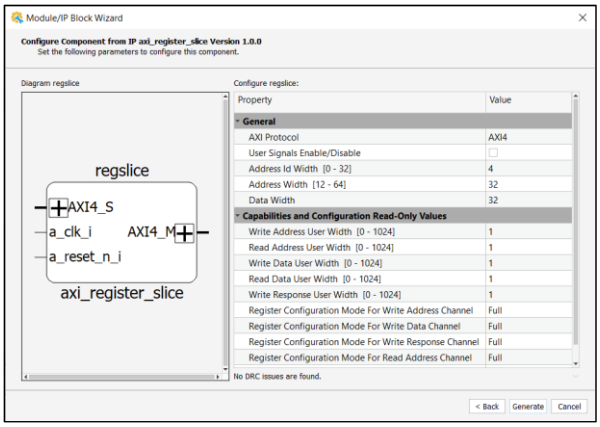


Figure 2.22. Configuring the IP

8. Click **Generate** followed by **Finish** to generate the IP according to the selected configuration.
9. The configured IP is now inserted into the design canvas as shown in Figure 2.23. Click on the **Generate** button to generate the design.

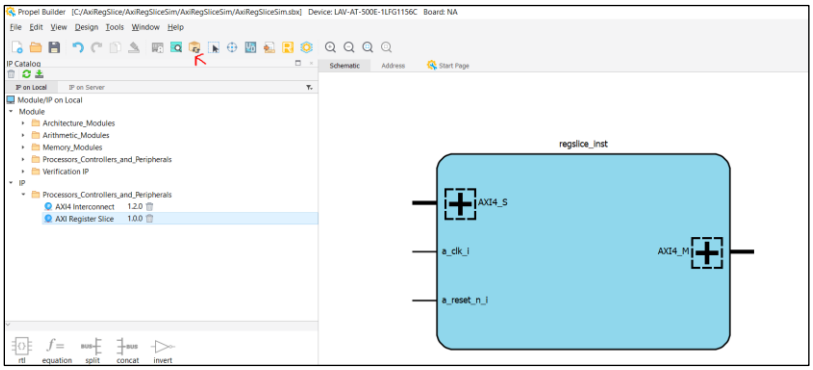


Figure 2.23. Generating the Design

10. Launch Radiant by clicking on the **Radiant** button.

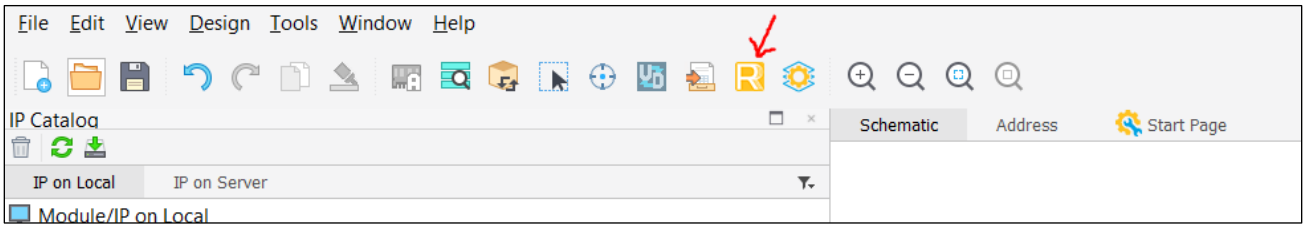


Figure 2.24. Launching Radiant

11. This will bring up the Radiant IDE. Launch the simulation by clicking on the **Simulation Wizard** button.

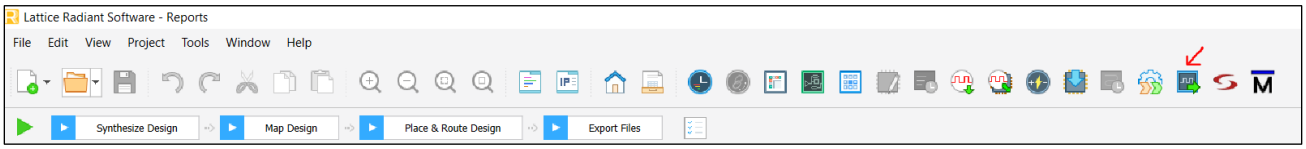


Figure 2.25. Launching Simulation

12. Click **Next** and enter the simulation project name, then **Next**. When prompted to create the simulation directory, click **Yes**.

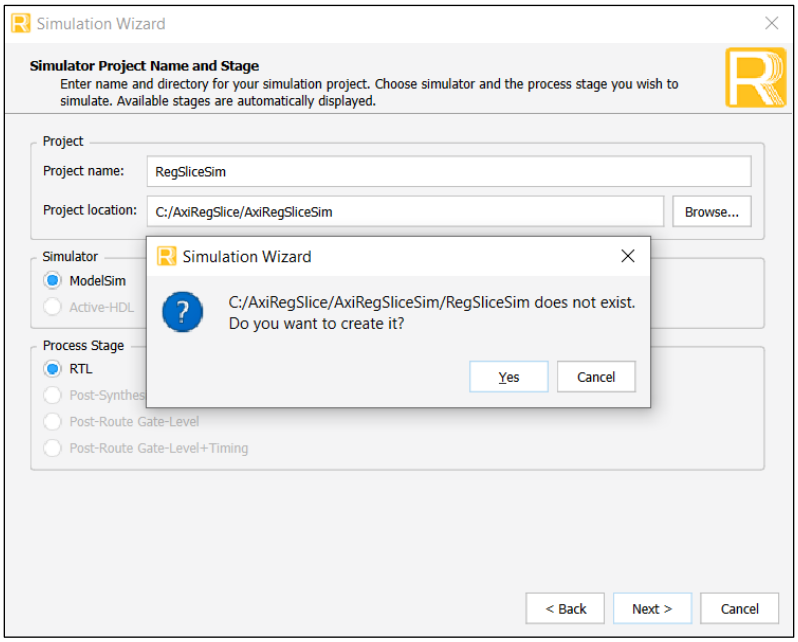


Figure 2.26. Simulator Project Name and Stage Settings

13. Since this IP was generated through Propel and not directly from Radiant, we will need to add the simulation files manually.
14. Remove the automatically added files and then click on the **Add simulation source files(s)** button as shown in Figure 2.27.

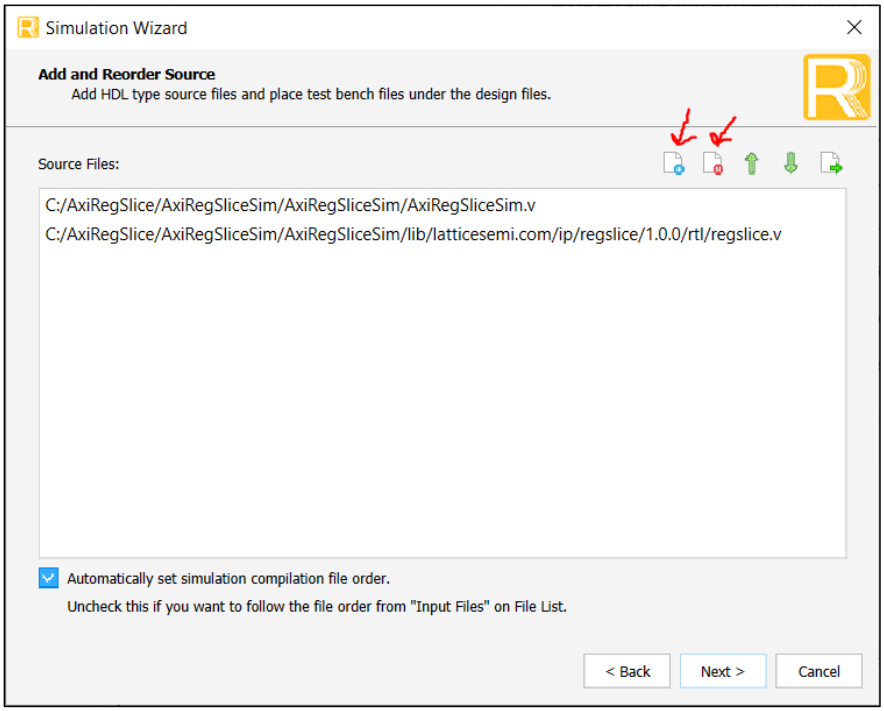


Figure 2.27. Adding and Reorder Source

15. This will bring up the project root directory.

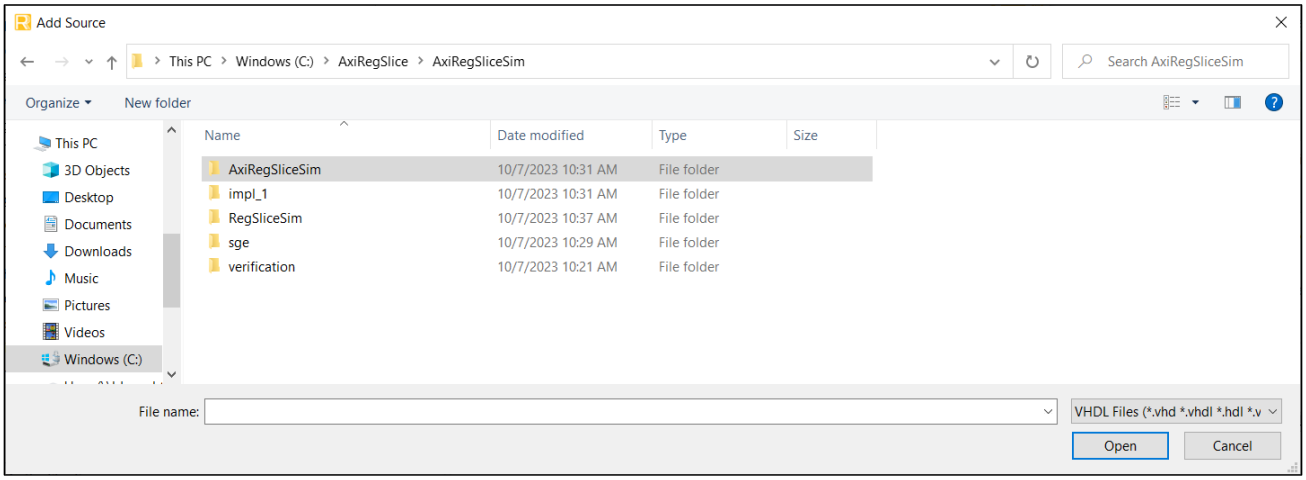


Figure 2.28. Adding Simulation Files

16. Change the file type to **Verilog**.
17. Navigate to the source RTL file, select and click **Open** as shown in Figure 2.29. Do not select the _bb version.

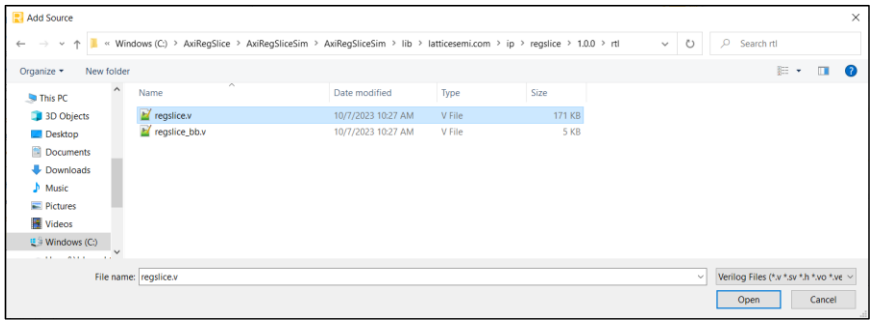


Figure 2.29. Adding Source RTL File

18. Add the files for the testbench files as shown in Figure 2.30.

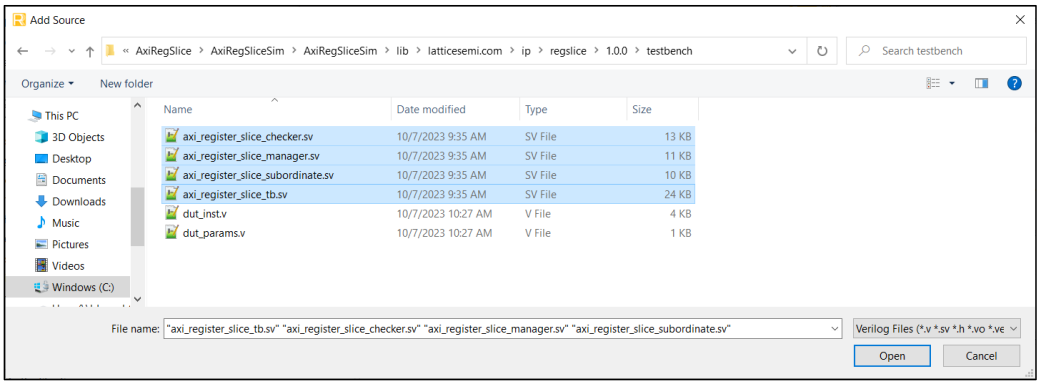


Figure 2.30. Adding SV Files to Project Root Path

19. Click **Open**, followed by **Next**. Leave the “Automatically set simulation compilation file order” selection checked.
20. In the next dialog, make sure to check and confirm the “Simulation Top Module” is axi_register_slice_tb.

21. Click **Next**, then **Finish**.
22. This will launch Modelsim, and simulation will begin. The default simulation time is 100 ns, but this simulation requires more than 100 ns. The user can either enter 0 to indicate run -all in step 19 before clicking **Finish**, or in the Modelsim GUI, key in **run -all** then click **Enter** in the transcript window. Simulation time takes approximately 5.5 us. Transcript file can directly print the Pass/Fail status for the various test cases.

Note: Test case is auto generated based on IP configuration. For example, if the user configures Full-Weight AXI-4 mode in IP then the testbench will automatically be configured to generate the stimulus for AXI-4 in Full-Weight mode.

2.6.2. IP Evaluation

The IP Core supports Lattice's IP evaluation capability when used in the supported FPGA family and targeted device. The IP evaluation capability can be enabled/disabled in the Strategy dialog box. It is disabled by default. To change this setting, go to **Strategies > Strategy1 (active strategy)> Bitstream**.

This IP has been validated using CertusPro-NX LFCPNX-100 9LFG672I Device.

This IP has not been validated using Lattice Avant device.

2.6.3. Hardware Validation

This IP has been validated using a Lattice CertusPro-NX device.

Appendix A. Resource Utilization

Table A.1. Resource Summary with LFCPNX-100

Bus	Mode	Registers	LUTs	EBRs	Target Device	Synthesis Tools
AXI4	Full-Weight	444	247	0	LFCPNX100-7ASG256I	Synplify Pro 2022.1
AXI4	Light-Weight	227	20	0	LFCPNX100-7ASG256I	Synplify Pro 2022.1
AXI4	Input Registered	1121	963	0	LFCPNX100-7ASG256I	Synplify Pro 2022.1
AXI3	Full-Weight	390	220	0	LFCPNX100-7ASG256I	Synplify Pro 2022.1
AXI3	Light-Weight	200	20	0	LFCPNX100-7ASG256I	Synplify Pro 2022.1
AXI3	Input Registered	986	857	0	LFCPNX100-7ASG256I	Synplify Pro 2022.1
AXI4-Lite	Full-Weight	306	185	0	LFCPNX100-7ASG256I	Synplify Pro 2022.1
AXI4-Lite	Light-Weight	152	21	0	LFCPNX100-7ASG256I	Synplify Pro 2022.1
AXI4-Lite	Input Registered	756	667	0	LFCPNX100-7ASG256I	Synplify Pro 2022.1

Table A.2. Resource Summary with LAV-AT-E70ES1-1LFG676I

Bus	Mode	Registers	LUTs	EBRs	Target Device	Synthesis Tools
AXI4	Full-Weight	436	228	0	LAV-AT-E70ES1-1LFG676I	Synplify Pro 2025.1
AXI4	Light-Weight	224	11	0	LAV-AT-E70ES1-1LFG676I	Synplify Pro 2025.1
AXI4	Input Registered	961	699	0	LAV-AT-E70ES1-1LFG676I	Synplify Pro 2025.1
AXI3	Full-Weight	392	206	0	LAV-AT-E70ES1-1LFG676I	Synplify Pro 2025.1
AXI3	Light-Weight	202	11	0	LAV-AT-E70ES1-1LFG676I	Synplify Pro 2025.1
AXI3	Input Registered	866	633	0	LAV-AT-E70ES1-1LFG676I	Synplify Pro 2025.1
AXI4-Lite	Full-Weight	296	158	0	LAV-AT-E70ES1-1LFG676I	Synplify Pro 2025.1
AXI4-Lite	Light-Weight	154	11	0	LAV-AT-E70ES1-1LFG676I	Synplify Pro 2025.1
AXI4-Lite	Input Registered	666	494	0	LAV-AT-E70ES1-1LFG676I	Synplify Pro 2025.1

References

- AXI Register Slice IP Release Notes (FPGA-RN-02049)
- Lattice Propel 2.1 Builder User Guide (FPGA-UG-02143)
- Lattice Radiant Timing Constraints Methodology (FPGA-AN-02059)
- AMBA AXI Specification
- LatticeECP3 web page
- ECP5 web page
- CrossLink-NX web page
- CertusPro-NX web page
- Certus-NX web page
- Certus-N2 web page
- MachXO5-NX web page
- Avant-E web page
- Avant-G web page
- Avant-X web page
- Lattice Radiant Software web page
- Lattice Propel Design Environment web page
- Lattice Diamond Software web page
- Lattice Insights for Lattice Semiconductor training courses and learning plans

Technical Support Assistance

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

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

Revision History

Revision 1.1, IP v1.2.0, June 2025

Section	Change Summary
All	• Renamed document from *AXI Register Slice IP Core - Lattice Radiant/Propel Builder* to *AXI Register Slice IP*. • Updated Lattice Avant device name from LAV-AT-500E to LAV-AT-E70. • Performed minor formatting and typo edits.
Disclaimers	Updated disclaimers.
Introduction	• Updated description. • Added Table 1.1. FPGA Software for IP Configuration, Generation, and Implementation.
Functional Description	Updated Figure 2.8. AXI Register Slice IP View.
Resource Utilization	Updated Table A.2. Resource Summary with LAV-AT-E70ES1-1LFG676I.
References	Updated references.

Revision 1.0, July 2023

Section	Change Summary
All	Initial release.

www.latticesemi.com