AXI4 Full Cross-bar v2.0

***IP User Guide** (Beta Release)*

November 20, 2023

RapidSilicon

Contents

IP Summary 2
Overview 3
AXI4 Crossbar 3
Licensing 4
IP Specification 5
Overview 5
Standards 7
IP Support Details 9
Resource Utilization 9
Port List 10
Parameters 12
Maximum Performance 12
Design Flow 13
IP Customization and Generation 13
Test Bench 15
Release 16
Revision History 16

IP Summary

Introduction

The AXI4 Full Crossbar is AXI4 compliance IP core that connects one or more AXI memory mapped master devices to more memory mapped slave devices. Each connected master could be a core that originates AXI transactions while each connected slave could be the final target of AXI transactions or a slave interface of a downstream AXI Crossbar core being cascaded.
This core support multiple clock feature, where different masters and slaves running on different clocks can communicate with each other.

Features

- Address width: Up to 64 bits
- Data width: 8, 16, 32, 64, 128, 256 bits
- Multi clock support
- Support Concurrent transactions
- Support all burst types

Overview

AXI4 Crossbar

The IP has Slave and Master interfaces through which different masters and slaves devices communicate with each other. The Slave Interface of AXI Crossbar core can be configured to comprise 1-4 Slave Interface slots to accept transactions from up to 4 connected master devices. The Master interface can be configured to comprise 1-4 Master Interface slots to issue transactions to up to 4 connected slave devices . All master and slave connected through this core can operate on different frequencies while the IP is responsible for clock conversion between these interfaces. The figure 1 shows the crossbar IP connecting multiple masters and slaves.

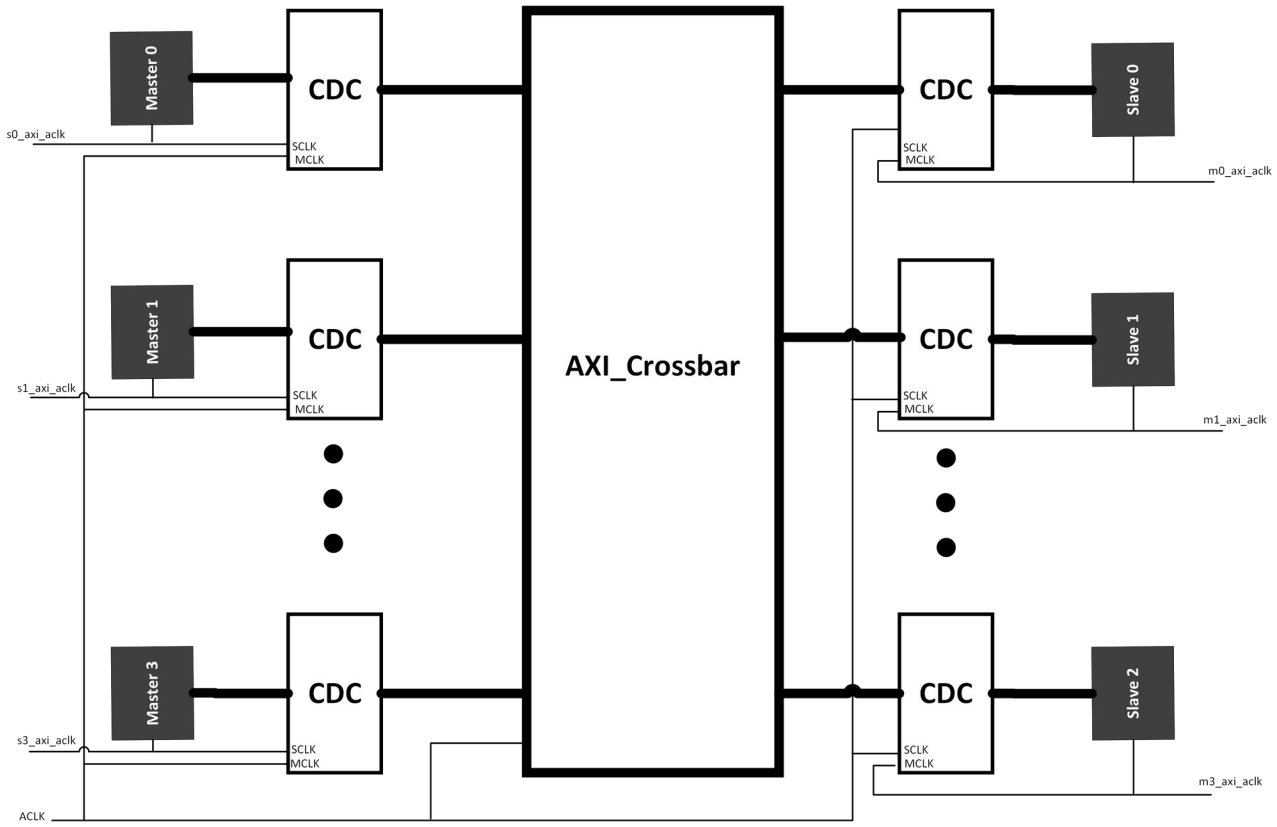


Figure 1. AXI4 Full Crossbar connecting multiples Masters and Slaves

Licensing

IP Specification

Overview

The figure 2 shows the top level block diagram of AXI4 Crossbar. Where multiple masters and slaves are connected. A maximum of 4 masters and 4 slaves can be connected with one crossbar instance. Each interface has its own CDC(Clock Domain Crossing) block for multi clock support. Inside the top the write and read channels has separate instance. All the write channels are connected with AXI Crossbar Write instance while all read channels are connected with AXI Crossbar Read instance. Both the write and read instance consists of a single, vectored AXI Slave Interface, plus a single, vectored AXI Master Interface. Each vectored interface can be configured to connect between 1 and 4 master/slave devices. The pathways connecting to all the master interfaces of the CDC block are merged together to connect to the vectored slave Interface of the write and read instance. The pathways connecting to all the slave interfaces of the CDC block are merged together to connect to the vectored master Interface of the write and read instance.
For each signal comprising a vectored AXI interface on the write and read instance, its natural width is multiplied by the number of devices to which it is connected.

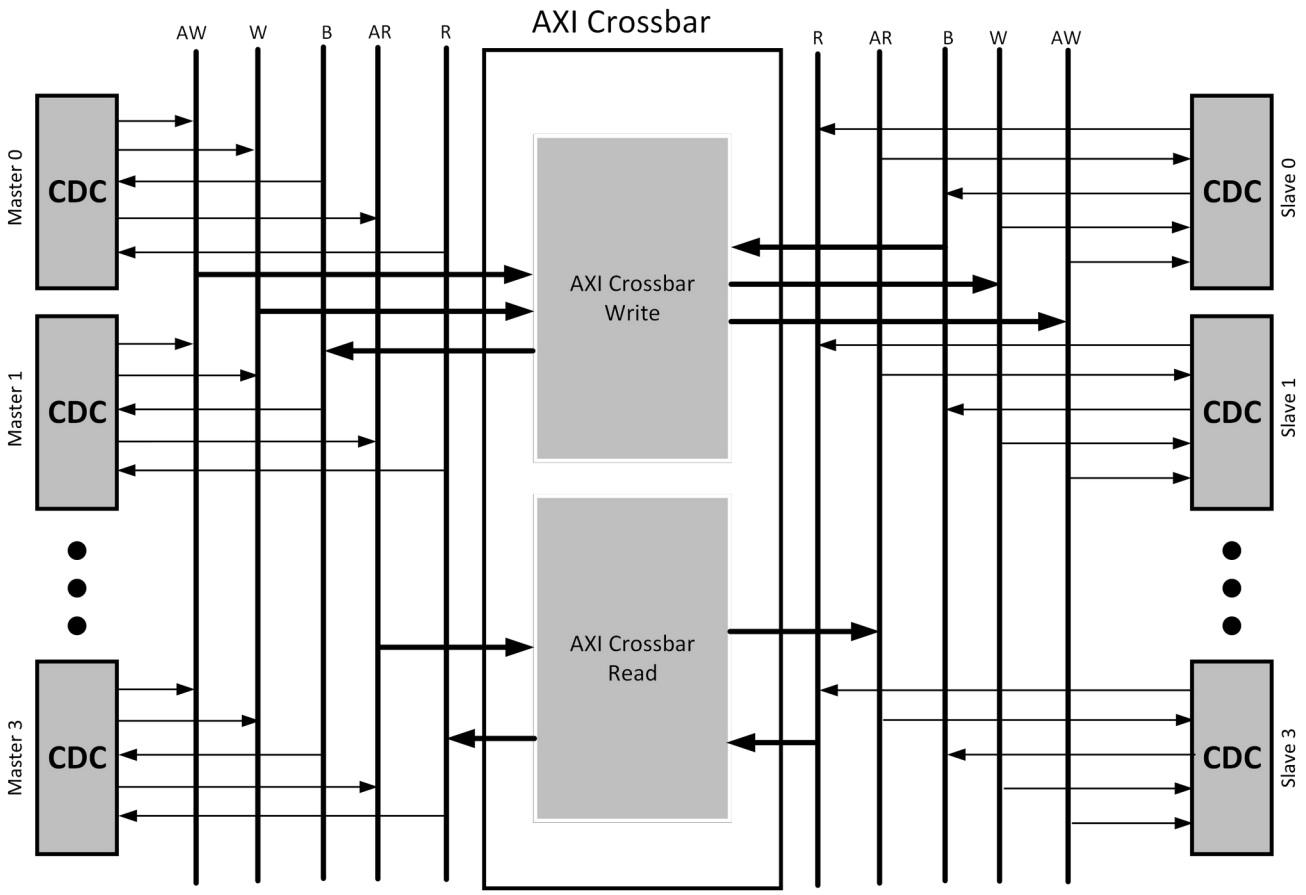


Figure 2. Top Module

Apart from CDC the core has sub modules including address decoder, priority encoder and arbiter. These modules are responsible for address decoding, setting priorities for transactions and arbitration.

Address Decoder

The figure 3 shows the block diagram of Address Decoder. The AXI4 Crossbar core must determine which Master Interface is the target of each transaction by decoding the address of each Address Write channel and Address Read channel transaction from the Slave Interface slot. So the address decoder assign address space to each Master Interface and maintain a table called address table. The address decoder also check the address space configuration to make sure that there is no overlapping of addresses and all addresses are aligned. This address table is then used by the address decoder during address decoding. The address decoder follows certain rules while assigning address space to each master interface.
Whenever a transaction address receive on the Slave Interface does not match any of the ranges being decoded by the Address decoder, the transaction is trapped and handled by a decoded error module. In such conditions the Crossbar generates a protocol compliant response back to the master which originate the transaction with the response code (DECERR), which means that their is no slave available among this address range.

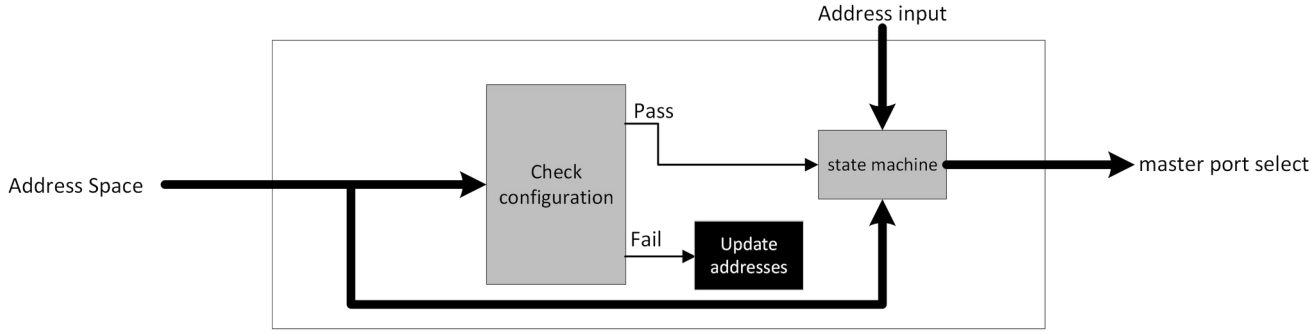


Figure 3. Address Decoder

Arbitration

Both write and read channel instance has arbiters for address and response arbitration. Primarily, arbitration is granted based on the relative priority set by the priority encoder. The arbitration among different transactions is decided by round-robin.
Each slave interface can accept a maximum of 16 concurrent transactions, while each master interface can issue maximum of 4 transactions. A write transaction starts when both AWVALID and AWREADY signal goes high and same transaction are considered to be complete when BVALID/BREADY handshake completes. Based on these handshake signals the counter increment upon the start of transaction and decrement upon the completion of transaction. This way it calculate the total number of write transactions in flight. Similarly for read transactions when transaction starts, both ARVAID and ARREADY signals goes high while transactions are considered to be complete when an RVALID/RREADY handshake completes in which RLAST is asserted. So based on these signals the crossbar count total number of read transactions in flight. So transactions that target the master interface that has reached its issuing limit are disqualified from arbitration and its request is not forwarded to arbiter.

Clock Domain Crossing

The AXI Crossbar top wrapper has one main clock ACLK while each master and slave interface has its own clock signals as shown in figure 1. As each CDC has two clocks, one is master clock and one is slave clock as shown in figure 4. The CDC block at slave interface shares master clock with Crossbar core while the CDC slave clock will be shared with master device. Similarly CDC block at master interface shares slave clock with Crossbar core while the CDC master clock will be shared with slave device. This feature allow that all the masters and slaves connected through the core can operate on different frequencies.

The CDC block is basically based on asynchronous FIFO, which has two seperate clock domain for write and read operations. Each cahnnel has its seperate FIFO instance. All the channel signals are passing through the CDC FIFO, which has two gray coded counters, one for pushing the FIFO in one clock domain while one is for popping from the FIFO in the other clock domain. The handshaking signals (VALID, READY) are used for status, writing to and reading form the FIFO. The figure 4 shows the block diagram of Clock Domain Crossing module.

Standards

The AXI4 Master and Slave interfaces are compliant with the AMBA® AXI Protocol Specification.

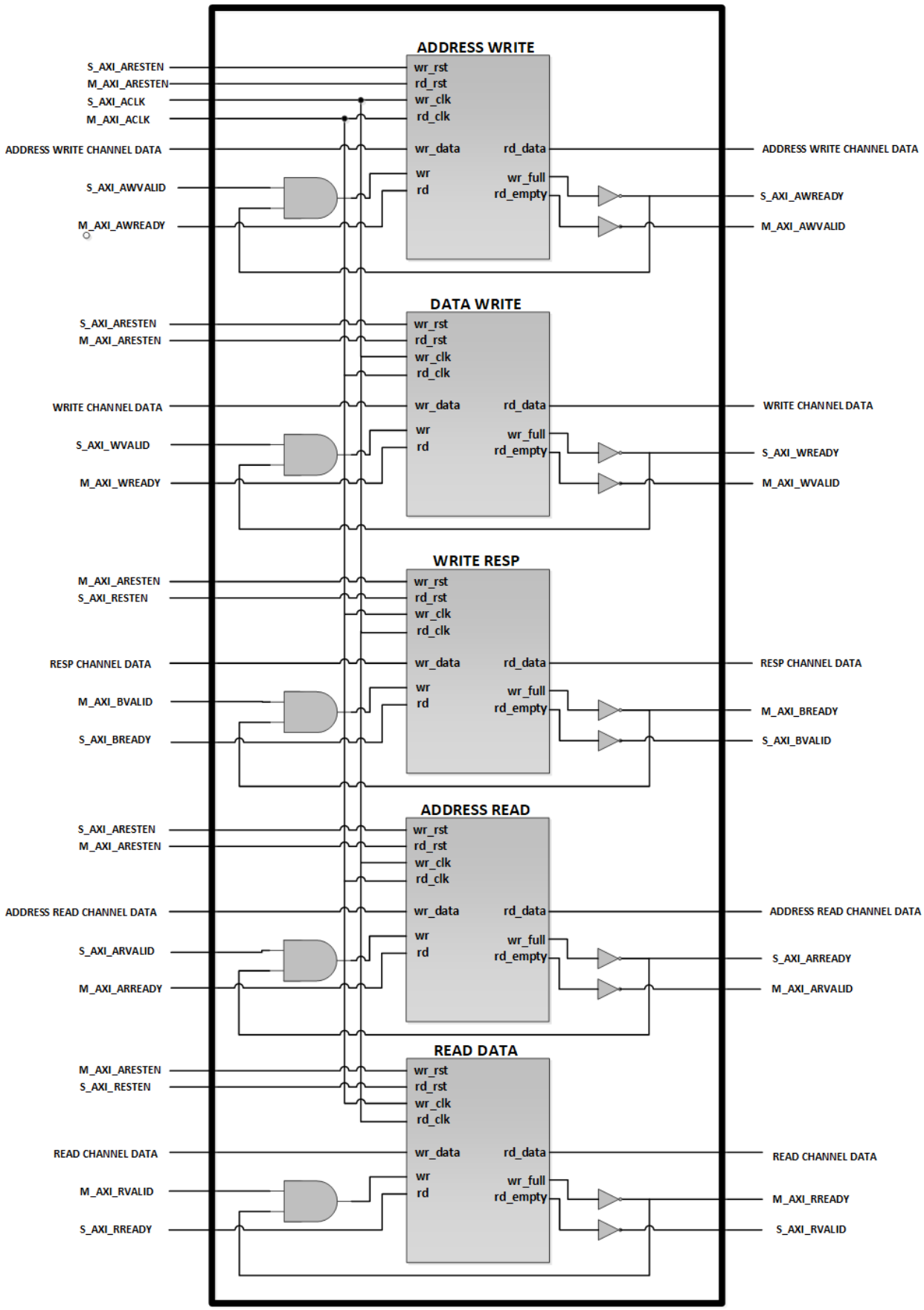


Figure 4. CDC Block Diagram

IP Support Details

Compliance		IP Resources					Tool Flow		
Device	**Interface**	**Source Files**	**Constraint File**	**Testbench**	**Simulation Model**	**Software Driver**	**Analyze and Elaboration**	**Simulation**	**Synthesis**
GEMINI	AXI4	Verilog	SDC	Cocotb	-	-	Raptor	Raptor	Raptor

Resource Utilization

Tool	Raptor Design Suite			
FPGA Device	GEMINI			
Configuration			**Resource Utilization**	
Minimum Resource	**Options**	**Configuration**	**Resources**	**Utilized**
	Number of slave Interface	1	LUT	1062
			DFF	1951
	Number of master Interface	1	BRAM	0
	BRAM Enable	No	DSP	0
Maximum Resource	**Options**	**Configuration**	**Resources**	**Utilized**
	Number of slave Interface	4	LUT	4069
			DFF	7502
	Number of master Interface	4	BRAM	0
	BRAM Enable	No	DSP	0

Ports

Table 2 lists the top Slave Interface ports of the AXI4 Full Crossbar.

Signal Name	I/O	Description
AXI Clock and Reset		
ACLK	I	AXI4 Source Clock
ARESET	I	AXI4 Active High RESET
AXI WRITE ADDRESS CHANNEL		
s<nn>_axi_awid	I	Write address ID
s<nn>_axi_awaddr	I	Write address
s<nn>_axi_awlen	I	Write burst length
s<nn>_axi_awsize	I	Write burst size
s<nn>_axi_awburst	I	Write burst type
s<nn>_axi_awlock	I	Write locking
s<nn>_axi_awcache	I	Write cache handling
s<nn>_axi_awprot	I	Write protection level
s<nn>_axi_awqos	I	Write QoS setting
s<nn>_axi_awvalid	I	Write address valid
s<nn>_axi_awready	O	Write address ready (from slave)
AXI WRITE DATA CHANNEL		
s<nn>_axi_wdata	I	Write data
s<nn>_axi_wstrb	I	Write data strobe (byte select)
s<nn>_axi_wlast	I	Write data last transfer in burst
s<nn>_axi_wvalid	I	Write data valid
s<nn>_axi_wready	O	Write data ready (from slave)
AXI WRITE RESPONSE CHANNEL		
s<nn>_axi_bid	O	Write response ID
s<nn>_axi_bresp	O	Write response
s<nn>_axi_bvalid	O	Write response valid
s<nn>_axi_bready	I	Write response ready (from master)
AXI READ ADDRESS CHANNEL		
s<nn>_axi_arid	I	Read address ID
s<nn>_axi_araddr	I	Read address
s<nn>_axi_arlen	I	Read burst length
s<nn>_axi_arsize	I	Read burst size
s<nn>_axi_arburst	I	Read burst type
s<nn>_axi_arlock	I	Read locking
s<nn>_axi_arcache	I	Read cache handling
s<nn>_axi_arprot	I	Read protection level
s<nn>_axi_arqos	I	Read QoS setting
s<nn>_axi_arvalid	I	Read address valid
s<nn>_axi_arready	O	Read address ready (from slave)
AXI READ DATA CHANNEL		
s<nn>_axi_rid	O	Read data ID
s<nn>_axi_rdata	O	Read data
s<nn>_axi_rresp	O	Read response
s<nn>_axi_rlast	O	Read data last transfer in burst

s<nn>_axi_rvalid	O	Read response valid
s<nn>_axi_rready	I	Read response ready (from master)

Table 2: Crossbar Slave

Interface NOTE: The <nn> shows the number of slave interface.

Table 4 lists the top Master Interface ports of the AXI4 Full Crossbar.

Signal Name	I/O	Description
AXI WRITE ADDRESS CHANNEL		
m<nn>_axi_awid	O	Write address ID
m<nn>_axi_awaddr	O	Write address
m<nn>_axi_awlen	O	Write burst length
m<nn>_axi_awsize	O	Write burst size
m<nn>_axi_awburst	O	Write burst type
m<nn>_axi_awlock	O	Write locking
m<nn>_axi_awcache	O	Write cache handling
m<nn>_axi_awprot	O	Write protection level
m<nn>_axi_awqos	O	Write QoS setting
m<nn>_axi_awvalid	O	Write address valid
m<nn>_axi_awready	I	Write address ready
AXI WRITE DATA CHANNEL		
m<nn>_axi_wdata	O	Write data
m<nn>_axi_wstrb	O	Write data strobe (byte select)
m<nn>_axi_wlast	O	Write data last transfer in burst
m<nn>_axi_wvalid	O	Write data valid
m<nn>_axi_wready	I	Write data ready
AXI WRITE RESPONSE CHANNEL		
m<nn>_axi_bid	I	Write response ID
m<nn>_axi_bresp	I	Write response
m<nn>_axi_bvalid	I	Write response valid
m<nn>_axi_bready	O	Write response ready
AXI READ ADDRESS CHANNEL		
m<nn>_axi_arid	O	Read address ID
m<nn>_axi_araddr	O	Read address
m<nn>_axi_arlen	O	Read burst length
m<nn>_axi_arsize	O	Read burst size
m<nn>_axi_arburst	O	Read burst type
m<nn>_axi_arlock	O	Read locking
m<nn>_axi_arcache	O	Read cache handling
m<nn>_axi_arprot	O	Read protection level
m<nn>_axi_arqos	O	Read QoS setting
m<nn>_axi_arvalid	O	Read address valid
m<nn>_axi_arready	I	Read address ready
AXI READ DATA CHANNEL		
m<nn>_axi_rid	I	Read data ID

m<nn>_axi_rdata	I	Read data
m<nn>_axi_rresp	I	Read response
m<nn>_axi_rlast	I	Read data last transfer in burst
m<nn>_axi_rvalid	I	Read response valid
m<nn>_axi_rready	O	Read response ready (from master)

Table 4: Crossbar Master

Interface NOTE: The <nn> shows the number of master interface.

Parameters

The AXI4 Full Crossbar has set of parameters which are available to user. These parameters include data width, address width, master count, slave count and BRAM. Data width and address width define the size of data and address bus respectively. User can select from 8 up to 256bits data size, wheres address can be configured either 32 bits or 64 bits. The M_Count and S_Count parameters enable the total number of master and slave interfaces.
The BRAM option is enabled by default. This parameter basically control the type of memory used by the CDC block. Table 5 lists the parameters of the AXI Crossbar.

Parameter	Values	Default Value	Description
Data Width	8-256	32	Define size of data for Data channel
Address Width	32, 64	32	Define size of address for Address channel
M Count	1,2,3,4	4	Total number of master ports.
S Count	1,2,3,4	4	Total number of slave ports.
BRAM	1,0	1	Set to 1 to use BRAM. Set to 0 to use Distributed RAM

Table 5: Parameters List

Maximum Performance

This section summarize the estimated maximum performance for AXI4 Full Crossbar with different number of master and slave interfaces. These frequencies are measured after running the desing on Raptor in standalone mode. Table 6 shows the maximum performance results.

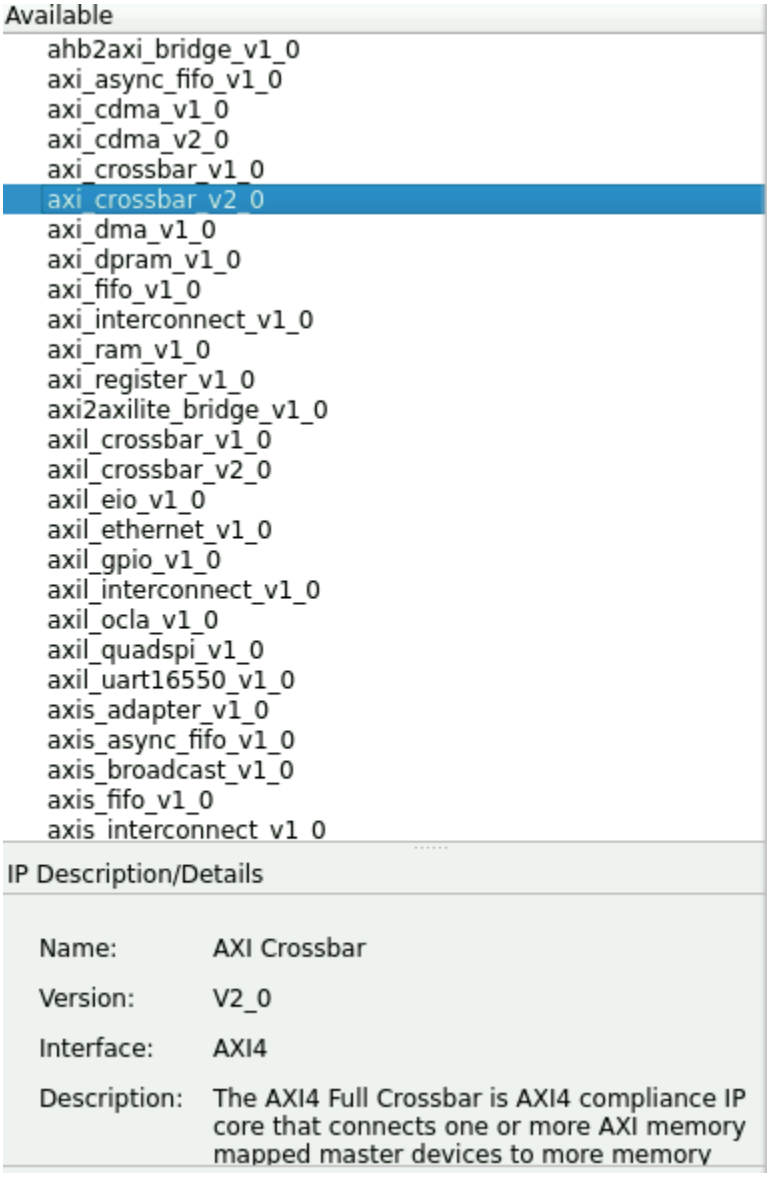
Number of Slave Interface	Number of Master Interface		
	2	3	4
1	200 MHz	—	—
2	200 MHz	—	—
3	197 MHz	180 MHz	—
4	197MHz	—	160 MHz

Table 6: AXI4 Crossbar Performance (MHz) on GEMINI

Design Flow

IP Customization and Generation

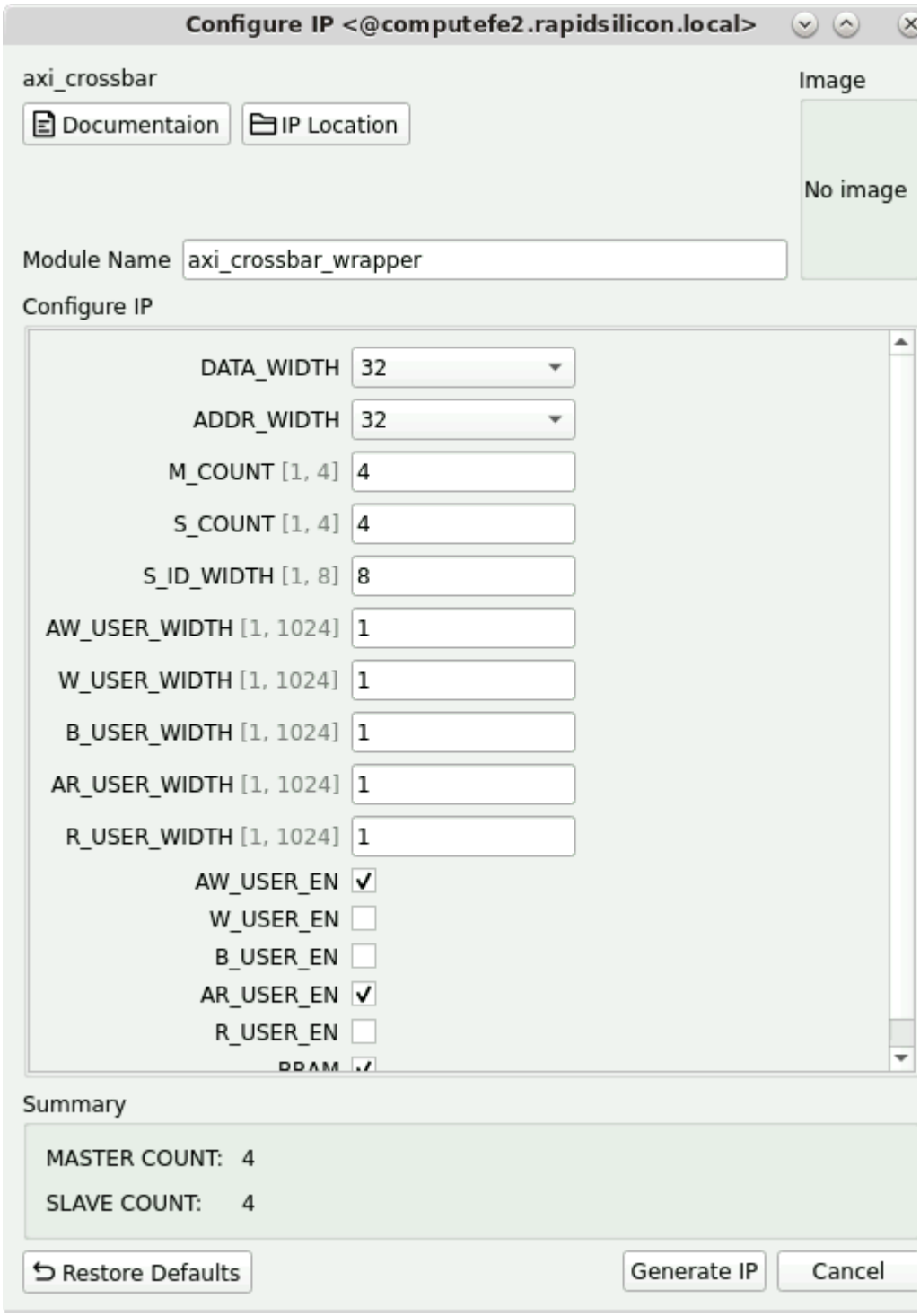
AXI4 Full Crossbar IP is a part of the Raptor Design Suite Software. A customized IP can be generated from the Raptor's IP configurator window. First enable IP Configurator and then select the axi_crossbar_v2_0 IP from the IP list.



IP list

You can also see the IP details including interface and description as shown in the figure above.

Parameters Customization: To customize the IP, double click the IP that will open the IP configurator window as shown in figure below. From the IP configuration window, the parameters of the IP can be configured and AXI4 Full Crossbar features can be enabled for generating a customized IP core that suits the user application requirement.



IP Configuration

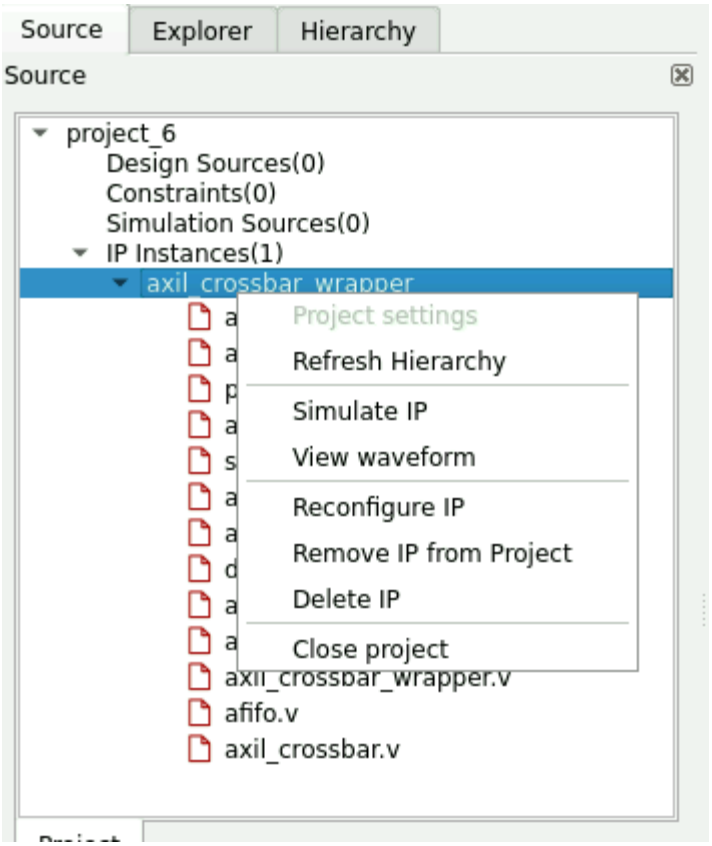
After the IP customization and generation step, a top wrapper plus all source files are made available to the user. Now user can add all the source files to project to use it at system level.

Test Bench

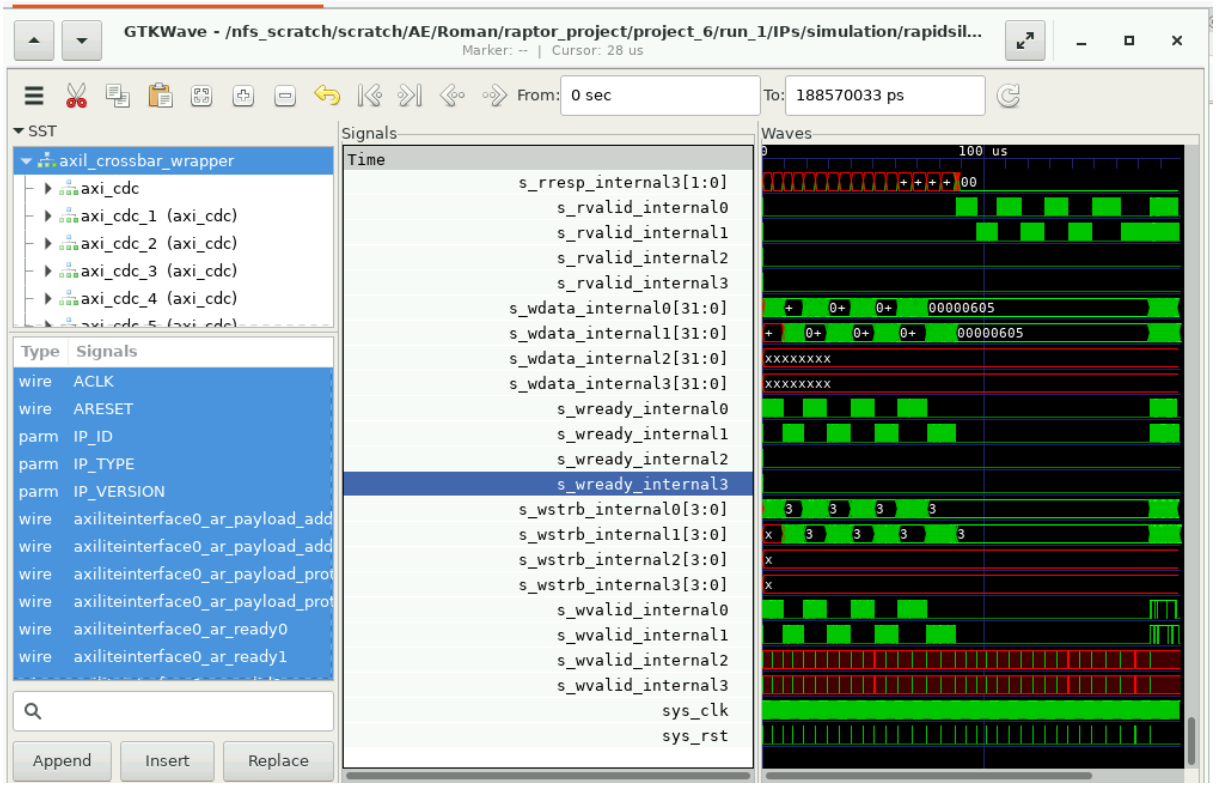
To check the behavior of the IP Core, a cocotb test-bench with basics configuration is available for simulation. Once the IP is generated then test-bench file can be found in the IP directory under sim folder.

To run simulation following these steps:

1. Right click on the IP name within source tab and select Simulate IP as shown in figure below.



2. This will run the simulation and result will dumped into vcd file.
3. To see the waveform, again right click on the IP and select View waveform. The GTKWave will open showing the IP simulation result as shown in figure below.



Simulation Result

Revision History

Date	Version	Revisions
November 20, 2023	0.01	Initial version of AXI4 Full Crossbar User Guide Document