AXI Stream Interconnect v1.0

IP User Guide (Beta Release)

November 20, 2023

Contents

IP Summary

Introduction

AXI Stream Interconnect IP is a digital logic block that facilitates communication between various IP blocks in a System-on-Chip (SoC) design, specifically for streaming data. It acts as a bridge between the different IP blocks, allowing them to transfer streaming data and information in a fast and efficient manner. The AXI Stream Interconnect IP implements the AXI Stream protocol, which is a specialized version of the AXI (Advanced eXtensible Interface) protocol. This protocol is optimized for the transfer of continuous data streams and is commonly used in high-speed video and audio processing applications. The IP block can support multiple AXI Stream ports and allows for configurable routing of data streams between different IP blocks, making it flexible and scalable for various SoC designs. AXI Stream Interconnect IP is commonly used in FPGA and ASIC designs to simplify the on-chip communication infrastructure for streaming data.

Features

- Supports parameterized AXI Stream Interface.
- Supports multiple masters and multiple slaves.
- Support separate select signals for each master.

Overview

AXI Stream Interconnect

AXI Stream Interconnect is an IP Core which used for the communication of various IP blocks in a System-on-Chip (SoC) design. This IP Core supports multiple ports, allowing for the routing of data streams between IP blocks in a flexible and scalable manner. The figure 1 shows the block diagram of AXIS Interconnect.

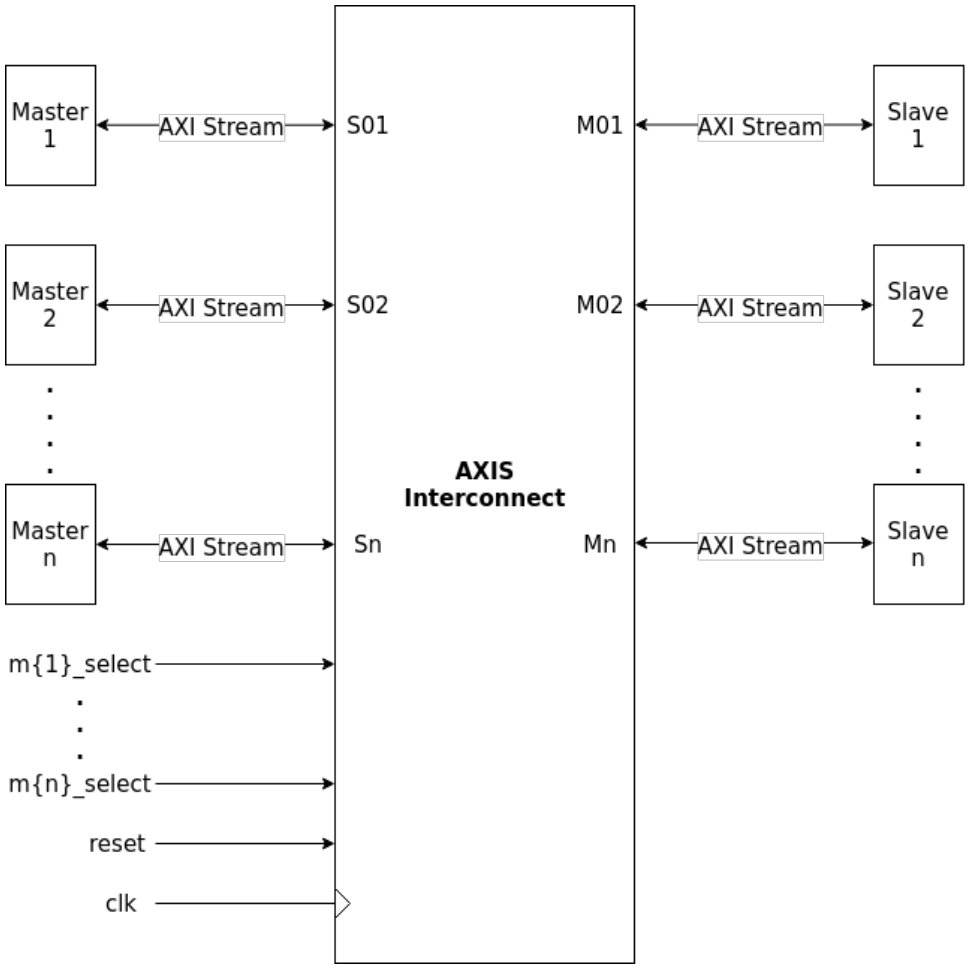

Figure 1. AXIS Interconnect Block Diagram

Licensing

IP Support Details

The Table 1 gives the support details for AXIS Interconnect.

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

Table 1. Support Details

Resource Utilization

The parameters for computing the maximum and the minimum resource utilization are given in Table 2, remaining parameters have been kept at their default values.

Tool	Raptor Design Suite			
FPGA Device	GEMINI			
Configuration			**Resource Utilization**	
Minimum Resource	**Options**	**Configuration**	**Resources**	**Utilized**
	S_COUNT	2	LUTS	48
	M_COUNT	1	REGISTERS	144
	DATA_WIDTH	32	CELLS	192
	ID_WIDTH	8	-	-
Maximum Resource	**Options**	**Configuration**	**Resources**	**Utilized**
	S_COUNT	16	LUTS	5772
	M_COUNT	16	REGISTERS	2368
	DATA_WIDTH	32	CELLS	8140
	ID_WIDTH	32	-	-

Table 2. Resource Utilization

Ports

Table 3 lists the top interface ports of the AXI Stream Interconnect.

Signal Name	I/O	Description
AXI Clock and Reset		
clk	I	AXI4-Stream Clock
rst	I	AXI4-Stream RESET
AXI Slave Interface		
s_axis_tdata	I	AXI4-Stream data
s_axis_tkeep	I	AXI4-Stream keep data qualifier
s_axis_tvalid	I	AXI4-Stream valid transfer
s_axis_tlast	I	AXI4-Stream boundary of transfer packet
s_axis_tid	I	AXI4-Stream data stream identifier
s_axis_tdest	I	AXI4-Stream data routing information
s_axis_tuser	I	AXI4-Stream user defined sideband information
AXI Master Interface		
m_axis_tdata	O	AXI4-Stream data
m_axis_tkeep	O	AXI4-Stream keep data qualifier
m_axis_tvalid	O	AXI4-Stream valid transfer
m_axis_tlast	O	AXI4-Stream boundary of transfer packet
m_axis_tid	O	AXI4-Stream data stream identifier
m_axis_tdest	O	AXI4-Stream data routing information
m_axis_tuser	O	AXI4-Stream user defined sideband information
Other Signals		
select	I	Select line for master selection

Table 3. AXI Stream Interface

Parameters

Table 4 lists the parameters of the AXIS Interconnect.

Parameter	Values	Default Value	Description
S_COUNT	2-16	4	Number of Slave Interfaces
M_COUNT	1-16	4	Number of Master Interfaces
DATA_WIDTH	1-4096	8	Data Width of Stream Interface
ID_WIDTH	1-32	8	Width of Transaction ID fields
DEST_WIDTH	1-32	8	Width of DEST fields
USER_WIDTH	1-4096	1	Width of USER fields
LAST_EN	True/False	True	Last enable
ID_EN	True/False	True	ID enable
DEST_EN	True/False	True	DEST enable
USER_EN	True/False	True	USER enable

Table 4. Parameters

Design Flow

IP Customization and Generation

AXI Stream Interconnect IP core is a part of the Raptor Design Suite Software. A customized memory can be generated from the Raptor's IP configurator window as shown in figure 2.

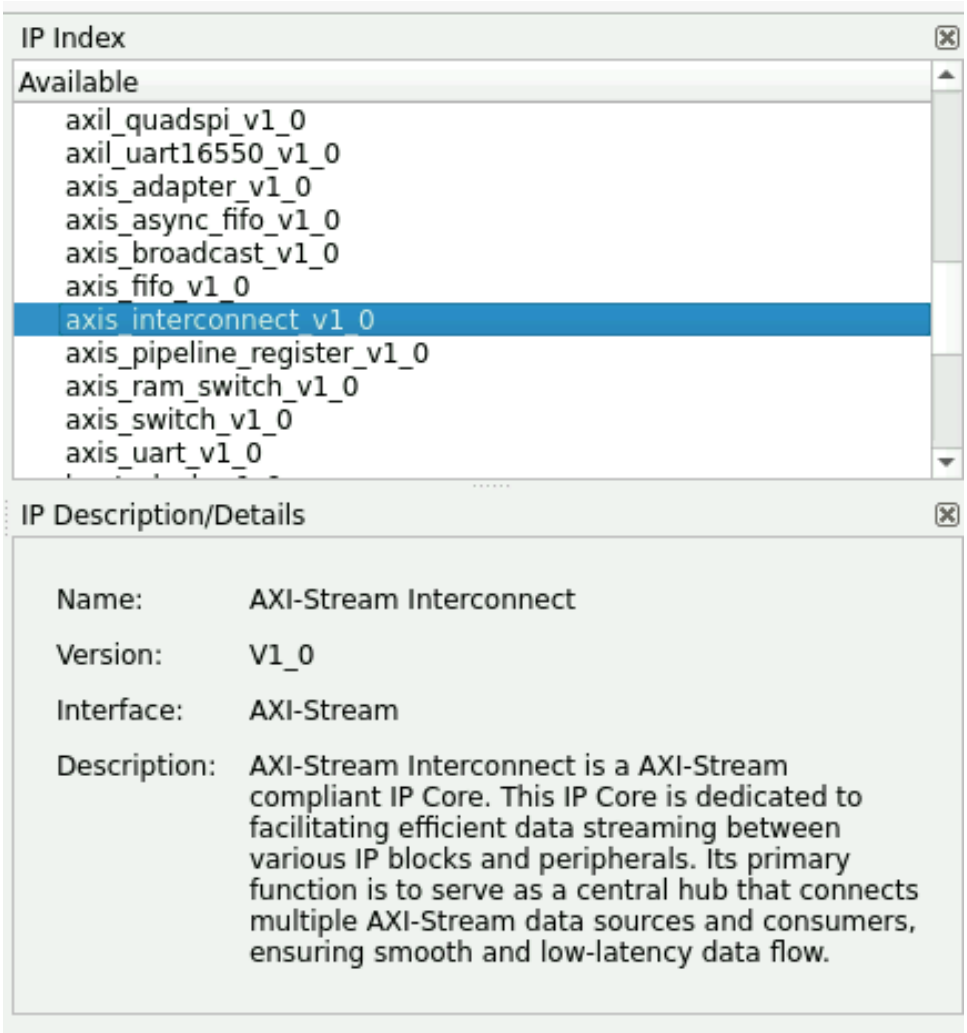

Figure 2. IP list

Parameters Customization

From the IP configuration window, the parameters of the can be configured and it's features can be enabled for generating a customized IP core that suits the user application requirements. All parameters are shown in figure 3.

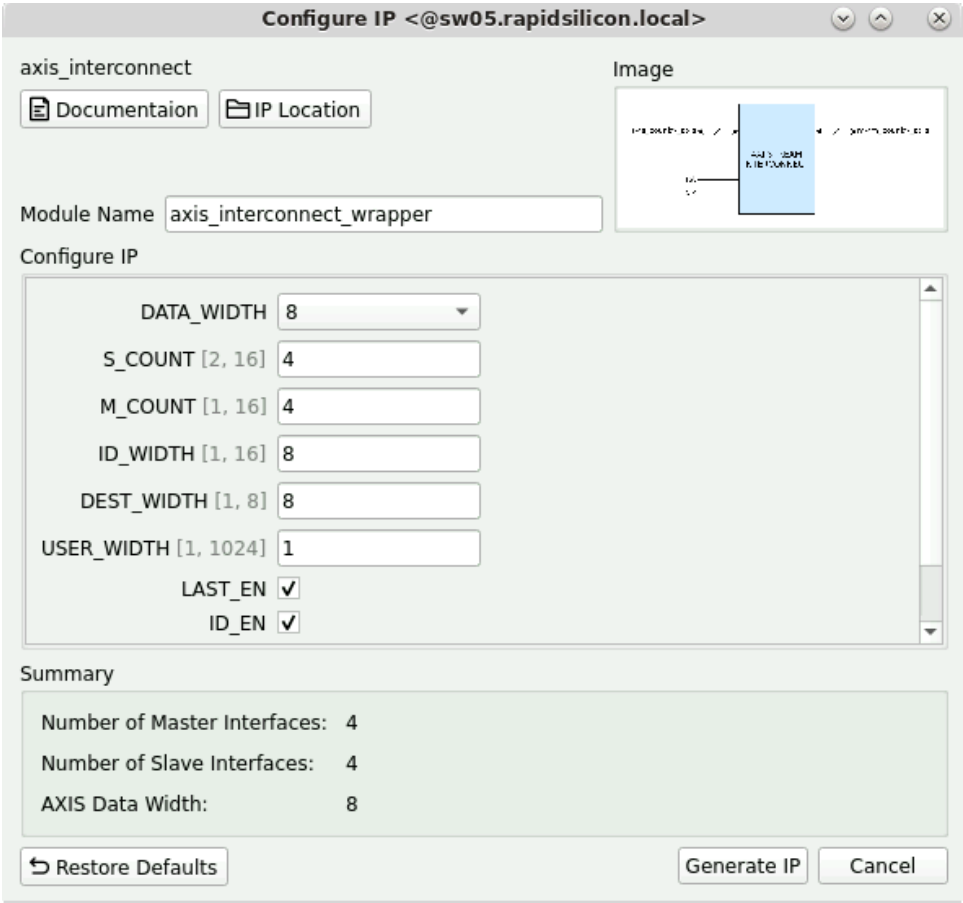

Figure 3. IP Configuration

Test Bench

The testbench included with AXI Stream Interconnect is myhdl based testbench. For simulation, right click on generated IP Instance and then click "Simulate IP" as shown in Figure 4.

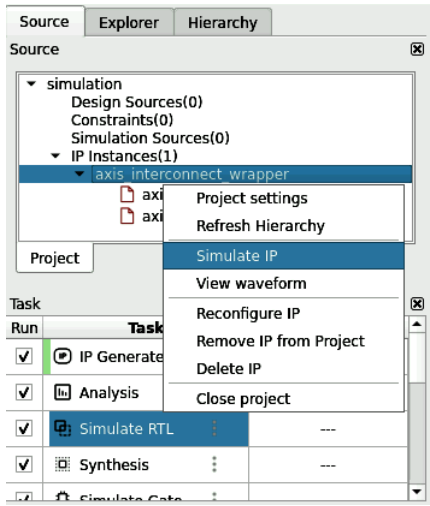

Figure 4. Simulate IP

In this test, multiple masters performs multiple transcations to multiple slaves. It contains four masters and four slaves connected to AXI Stream Interconnect. Multiple masters sends data frames to multiple slaves through AXI Stream Interconnect. After running the simulation, you'll get pass/ fail status on console. The status of test is shown in Figure 5.

```
Running test...
test 1: 0123 -> 0123
[source_0] Sending frame AXIStreamFrame(data=bytearray(b'\x01\x00\x00\xff\x01\x02\x03\x04'), keep=None, id=0, dest=0, user=None, last_cycle_user=None)
[source_1] Sending frame AXIStreamFrame(data=bytearray(b'\x01\x01\x01\xff\x01\x02\x03\x04'), keep=None, id=1, dest=1, user=None, last_cycle_user=None)
[source_2] Sending frame AXIStreamFrame(data=bytearray(b'\x01\x02\x02\xff\x01\x02\x03\x04'), keep=None, id=2, dest=2, user=None, last_cycle_user=None)
[source_3] Sending frame AXIStreamFrame(data=bytearray(b'\x01\x03\x03\xff\x01\x02\x03\x04'), keep=None, id=3, dest=3, user=None, last_cycle_user=None)
[sink_3] Got frame AXIStreamFrame(data=bytearray(b'\x01\x03\x03\xff\x01\x02\x03\x04'), keep=[255], id=[3], dest=[3], user=[0], last_cycle_user=0)
[sink_2] Got frame AXIStreamFrame(data=bytearray(b'\x01\x02\x02\xff\x01\x02\x03\x04'), keep=[255], id=[2], dest=[2], user=[0], last_cycle_user=0)
[sink_1] Got frame AXIStreamFrame(data=bytearray(b'\x01\x01\x01\xff\x01\x02\x03\x04'), keep=[255], id=[1], dest=[1], user=[0], last_cycle_user=0)
[sink_0] Got frame AXIStreamFrame(data=bytearray(b'\x01\x00\x00\xff\x01\x02\x03\x04'), keep=[255], id=[0], dest=[0], user=[0], last_cycle_user=0)
test 2: 0123 -> 3210
[source_0] Sending frame AXIStreamFrame(data=bytearray(b'\x02\x00\x03\xff\x01\x02\x03\x04'), keep=None, id=0, dest=3, user=None, last_cycle_user=None)
[source_1] Sending frame AXIStreamFrame(data=bytearray(b'\x02\x01\x02\xff\x01\x02\x03\x04'), keep=None, id=1, dest=2, user=None, last_cycle_user=None)
[source_2] Sending frame AXIStreamFrame(data=bytearray(b'\x02\x02\x01\xff\x01\x02\x03\x04'), keep=None, id=2, dest=1, user=None, last_cycle_user=None)
[source_3] Sending frame AXIStreamFrame(data=bytearray(b'\x02\x03\x00\xff\x01\x02\x03\x04'), keep=None, id=3, dest=0, user=None, last_cycle_user=None)
[sink_3] Got frame AXIStreamFrame(data=bytearray(b'\x02\x00\x03\xff\x01\x02\x03\x04'), keep=[255], id=[0], dest=[3], user=[0], last_cycle_user=0)
[sink_2] Got frame AXIStreamFrame(data=bytearray(b'\x02\x01\x02\xff\x01\x02\x03\x04'), keep=[255], id=[1], dest=[2], user=[0], last_cycle_user=0)
[sink_1] Got frame AXIStreamFrame(data=bytearray(b'\x02\x02\x01\xff\x01\x02\x03\x04'), keep=[255], id=[2], dest=[1], user=[0], last_cycle_user=0)
[sink_0] Got frame AXIStreamFrame(data=bytearray(b'\x02\x03\x00\xff\x01\x02\x03\x04'), keep=[255], id=[3], dest=[0], user=[0], last_cycle_user=0)
test 3: 0000 -> 0123
[source_0] Sending frame AXIStreamFrame(data=bytearray(b'\x03\x00\xff\xff\x01\x02\x03\x04'), keep=None, id=0, dest=0, user=None, last_cycle_user=None)
[sink_3] Got frame AXIStreamFrame(data=bytearray(b'\x03\x00\xff\xff\x01\x02\x03\x04'), keep=[255], id=[0], dest=[0], user=[0], last_cycle_user=0)
[sink_2] Got frame AXIStreamFrame(data=bytearray(b'\x03\x00\xff\xff\x01\x02\x03\x04'), keep=[255], id=[0], dest=[0], user=[0], last_cycle_user=0)
[sink_1] Got frame AXIStreamFrame(data=bytearray(b'\x03\x00\xff\xff\x01\x02\x03\x04'), keep=[255], id=[0], dest=[0], user=[0], last_cycle_user=0)
[sink_0] Got frame AXIStreamFrame(data=bytearray(b'\x03\x00\xff\xff\x01\x02\x03\x04'), keep=[255], id=[0], dest=[0], user=[0], last_cycle_user=0)
TEST 2 PASSED
```

Figure 5. Simulation Results

You can view waveform of the results. To view waveform, right click on generated IP Instance and then click "View waveform" as shown in Figure 6.

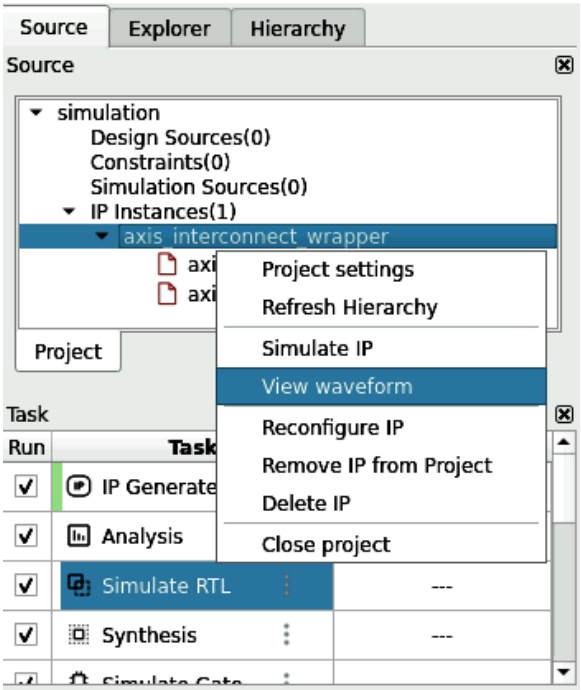

Figure 6. View Waveform

Revision History

Date	Version	Revisions
November 20, 2023	0.01	Initial version AXI Stream Interconnect User Guide Document