RapidSilicon

AXI4-Stream Broadcast (Beta Release)

Version 0.1

February 29, 2024

Contents

IP Summary

Introduction

A Broadcast is a communication protocol for connecting different components of a system in a parallel and synchronized manner. Broadcast allows multiple slaves to receive the same data from a single master at the same time. This is useful in situations where multiple slaves need to receive the same information, such as when updating the contents of a shared memory. It also improves the system performance and reduces power consumption by avoiding the data duplication and multiple transfers from the master to multiple slaves. This is an AXI-Stream compliant Broadcast IP for easy integration with other AXI based systems.

Features

- Duplicates one input stream across multiple output streams.
- Support for up to 16 Masters.
- Configurable data width, destination width, ID width and user width.
- Configurable AXI Stream Signals for better control.

Overview

AXIS Broadcast

The AXI Broadcast feature is a way for the master to communicate with multiple slaves simultaneously by sending a single transaction to a special broadcast address. This address is usually the highest address in the address space of the system. When the AXI interconnect receives the transaction with the broadcast address, it distributes the transaction to all slaves that are configured to respond to the broadcast address. This feature can be particularly useful in scenarios where multiple slaves need to be updated or accessed simultaneously. For example, in a cache coherency protocol, a processor core may need to invalidate the cache lines of all other cores when it updates a shared memory location. Using the AXI Broadcast feature, the core can send a single transaction with the broadcast address to invalidate all other caches simultaneously. Similarly, in a multicast communication scenario, a master can use the AXI Broadcast feature to send the same data to multiple slaves, such as in a video or audio streaming application. However, it's important to note that using the AXI Broadcast feature can introduce additional complexity and potential contention issues in the system. For example, if multiple masters use the broadcast address to send transactions simultaneously, it can cause contention and delay in the system. Also, not all AXI implementations support the broadcast feature, so it's important to check the specifications of the specific AXI interconnect or IP block being used. A block diagram for the Broadcast IP is shown in Figure 1.

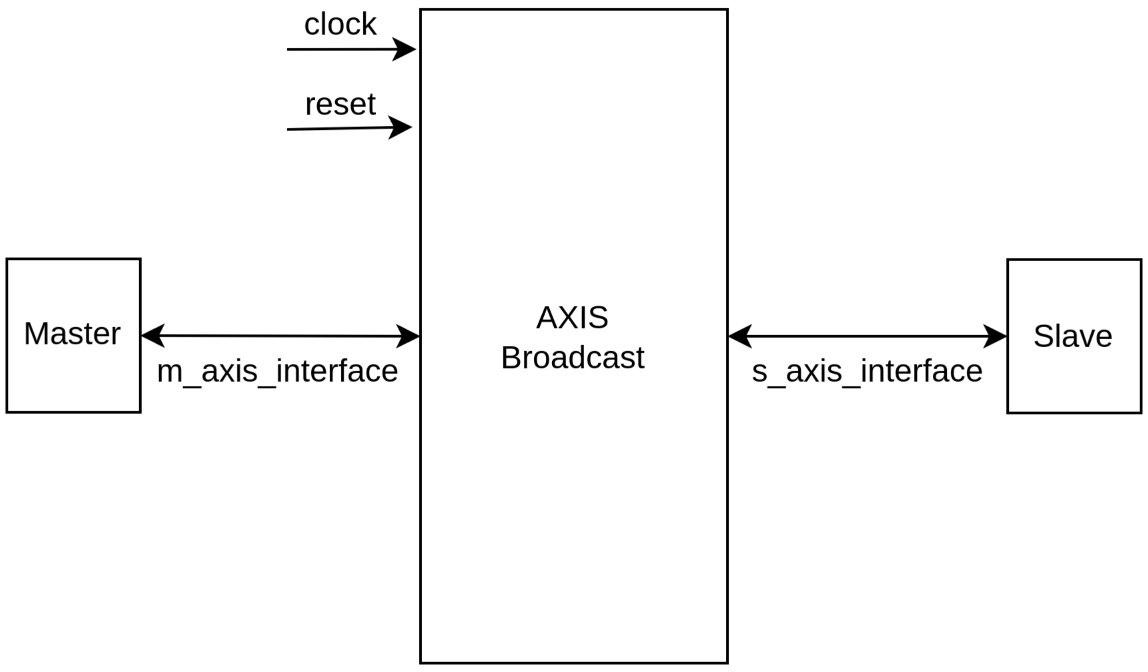


Figure 1: AXIS Broadcast Block Diagram

IP Specification

In a typical AXI4 Stream transfer, the master device sends a sequence of data packets to the slave device. Each packet includes a data payload and some control information, such as a packet identifier, data length, and end-of-packet marker. The slave device reads the data packets from the master device and processes them according to its own logic.
In AXI4 Stream Broadcast, the master device sends a single stream of data to multiple slave devices simultaneously. This is accomplished by using a broadcast channel, which is a special type of channel that replicates the data stream to multiple output channels.
To implement AXI4 Stream Broadcast, the master device sends the data stream to the broadcast channel, which then replicates the data stream to multiple output channels. Each output channel connects to a separate slave device, which reads the data stream from the channel and processes it according to its own logic.
One of the key benefits of AXI4 Stream Broadcast is that it can help to reduce the amount of data traffic on a system bus, since multiple devices can receive the same data without requiring multiple transfers. This can be especially useful in applications such as video processing, where multiple display devices need to receive the same video stream.
However, there are some challenges associated with implementing AXI4 Stream Broadcast. One challenge is managing the bandwidth and latency of the broadcast channel. Since the channel must replicate the data stream to multiple output channels, it can be more bandwidth-intensive than a regular AXI4 Stream transfer. Additionally, since the output channels may have different processing latencies, it is important to ensure that all slave devices receive the data stream correctly and in a timely manner. This Broadcast IP supports up to 16 masters with much customization suited for consumer needs in an FPGA friendly fashion.
Overall, AXI4 Stream Broadcast is a useful protocol for streaming data to multiple devices in a digital system. It can help to reduce system complexity and data traffic, while enabling efficient and reliable data transfers. The internal block diagram can be seen in Figure 2.

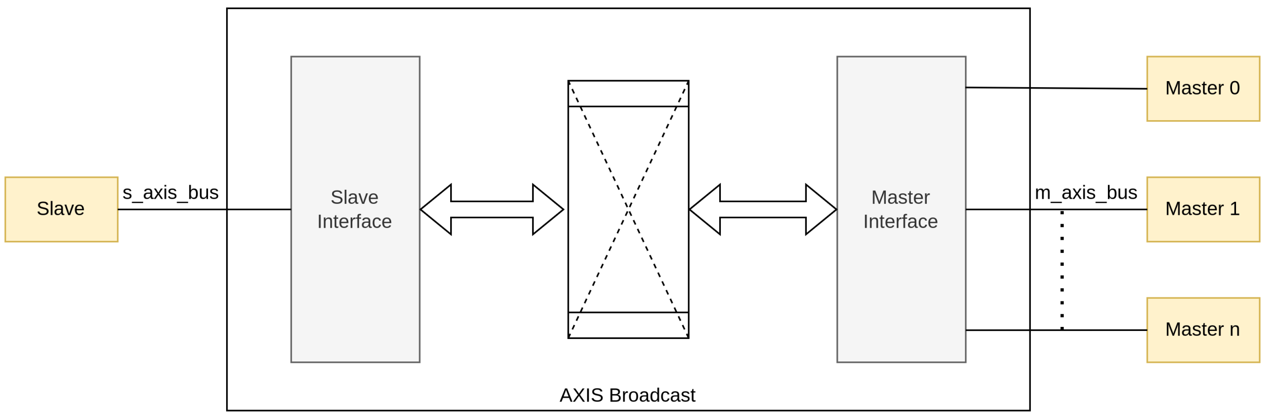


Figure 2: AXIS Broadcast Internal Diagram

Standards

The AXI4-Stream interface is compliant with the AMBA® AXI Protocol Specification.

IP Support Details

The Table 1 gives the support details for AXIS Broadcast.

Compliance		IP Resources				Tool Flow		
Device	Interface	Source Files	Constraint Files	Testbench	Simulation Model	Analyze and Elaboration	Simualtion	Synthesis
Gemini	AXI4-Stream	Verilog	-	Python	Cocotb	Surelog (Raptor)	Icarus (Raptor)	Raptor

Table 1: IP Details

Port List

Table 2 lists the top interface ports of the AXIS Broadcast.

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_tready	O	AXI4-Stream transfer ready
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_tready	I	AXI4-Stream transfer ready
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

Table 2: AXIS Broadcast Interface

Parameters

Table 3 lists the parameters of the AXIS Broadcast.

Parameter	Values	Default Value	Description
M COUNT	2 - 16	4	Number of Master Interfaces
DATA WIDTH	8, 16, 32, 64, 128, 256, 512, 1024	8	Data Width for each transfer
USER WIDTH	1 - 1024	1	Data Width for user defined sideband information
DEST WIDTH	1 - 8	1	Destination Width
ID WIDTH	1 - 16	2	ID Width
ID ENABLE	0 / 1	0	ID Enable
USER ENABLE	0 / 1	1	User Data Enable
DEST ENABLE	0 / 1	1	TDEST Signal Enable
LAST ENABLE	0 / 1	1	TLAST Signal Enable
IP TYPE	-	ASBR	Type of Peripheral
IP VERSION	-	<ip_version>	Version of Peripheral
IP ID	-	<date_and_time>	Date and Time of the generated Peripheral

Table 3: Parameters

Resource Utilization

The parameters for computing the maximum and minimum resource utilization are given in Table 4, remaining parameters have been kept at their default values.

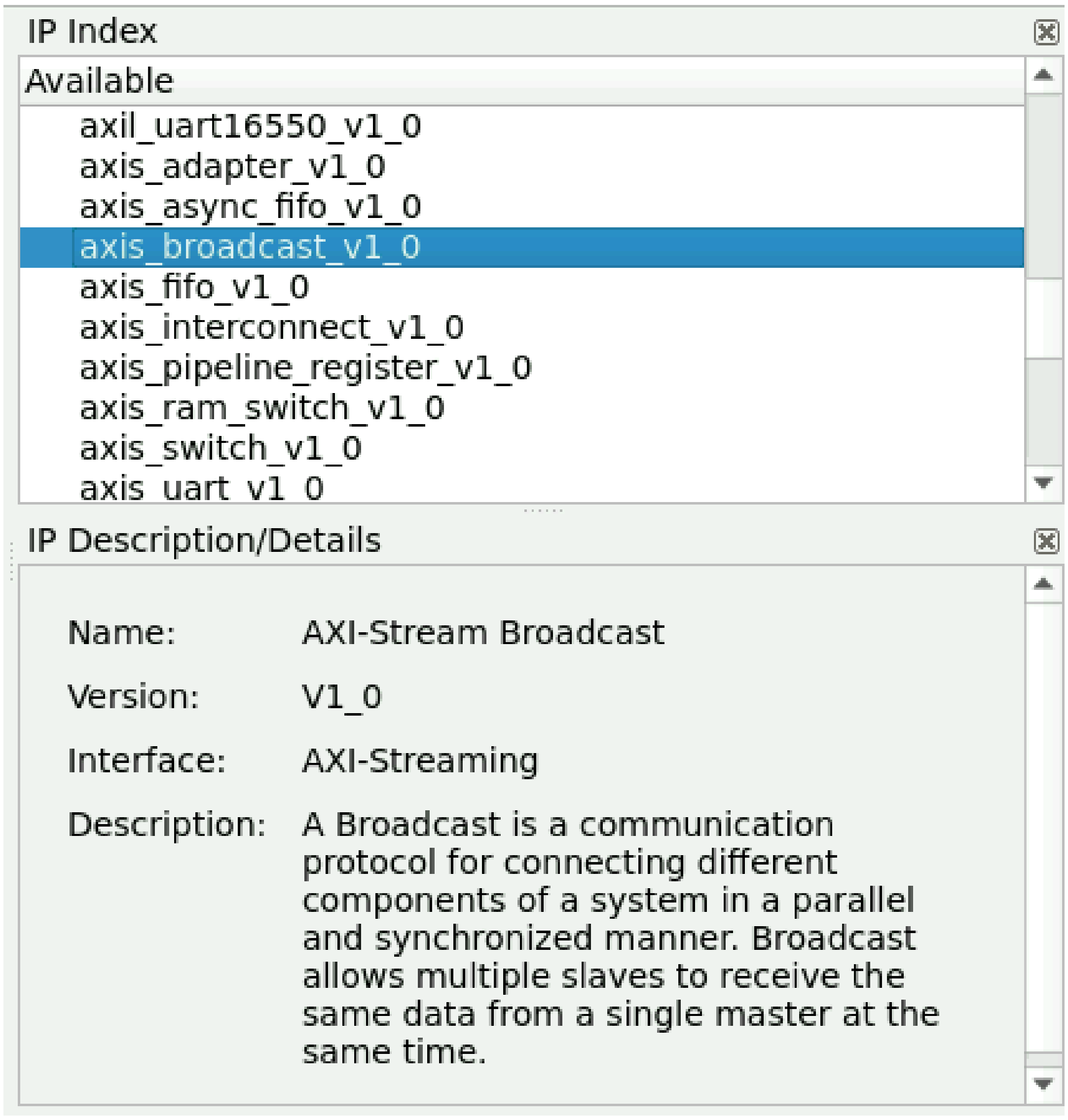
Tool	Raptor Design Suite			
FPGA Device	GEMINI			
Configuration			**Resource Utilization**	
Minumum Resource	**Options**	**Configuration**	**Resource**	**Utilized**
	M COUNT	2	LUTs	17
	DATA WIDTH	8		
	ID ENABLE	0		
	DEST ENABLE	0	Registers	20
	USER ENABLE	0		
	LAST ENABLE	0		
Maximum Resource	**Options**	**Configuration**	**Resource**	**Utilized**
	M COUNT	16	LUTs	2207
	DATA WIDTH	1024		
	ID WIDTH	16		
	DEST WIDTH	8	Registers	4394
	USER WIDTH	1024		
	LAST ENABLE	1		

Table 4: Resource Utilization

Design Flow

IP Customization and Generation

AXIS Broadcast IP core is a part of the Raptor Design Suite Software. A customized AXIS Broadcast can be generated from the Raptor's IP configurator window as shown in Figure 3.



IP Index

Available

axil_uart16550_v1_0
axis_adapter_v1_0
axis_async_fifo_v1_0
axis_broadcast_v1_0
axis_fifo_v1_0
axis_interconnect_v1_0
axis_pipeline_register_v1_0
axis_ram_switch_v1_0
axis_switch_v1_0
axis_uart_v1_0

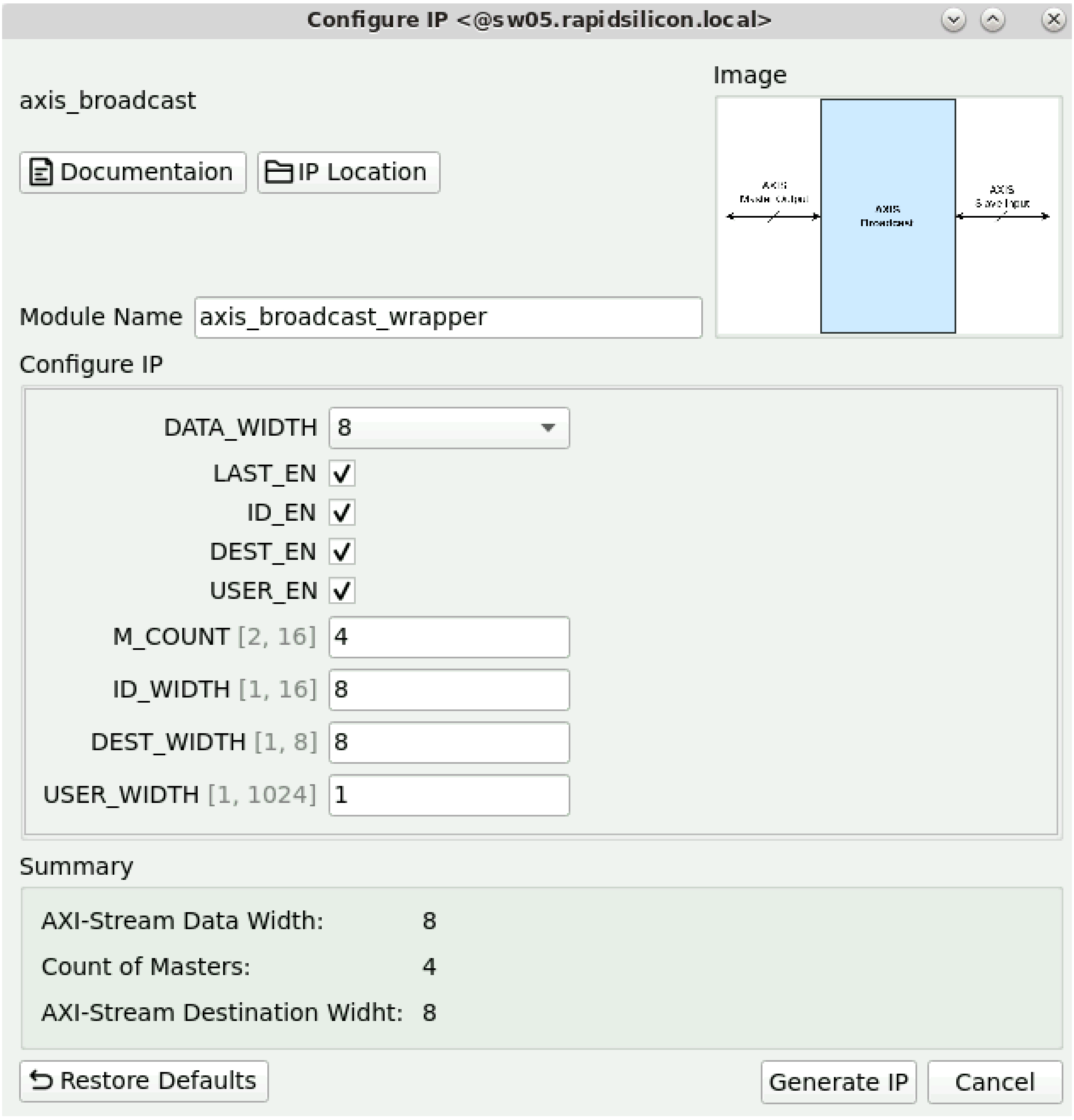
IP Description/Details

Name:	AXI-Stream Broadcast
Version:	V1_0
Interface:	AXI-Streaming
Description:	A Broadcast is a communication protocol for connecting different components of a system in a parallel and synchronized manner. Broadcast allows multiple slaves to receive the same data from a single master at the same time.

Figure 3: IP list

Parameters Customization

From the IP configuration window, the parameters of the Broadcast can be configured and Broadcast features can be enabled for generating a customized Broadcast IP core that suits the user application requirement as shown in Figure 4. After IP Customization, all the source files are made available to the user with a top wrapper that instantiates a parameterized instance of the AXIS Broadcast.



axis_broadcast

Module Name: axis_broadcast_wrapper

Configure IP

DATA_WIDTH	8
LAST_EN	✓
ID_EN	✓
DEST_EN	✓
USER_EN	✓
M_COUNT [2, 16]	4
ID_WIDTH [1, 16]	8
DEST_WIDTH [1, 8]	8
USER_WIDTH [1, 1024]	1

Summary

AXI-Stream Data Width:	8
Count of Masters:	4
AXI-Stream Destination Widht:	8

Figure 4: IP Configuration

Example Design

Overview

This AXIS Broadcast IP can be utilized in any system that has multiple masters and there is a need to forward the same data to all of these masters from one slave. This helps reduce redundancy of similar data by providing all masters with the same data traffic. One such example design of this AXIS Broadcast can be visualized in Figure 5.

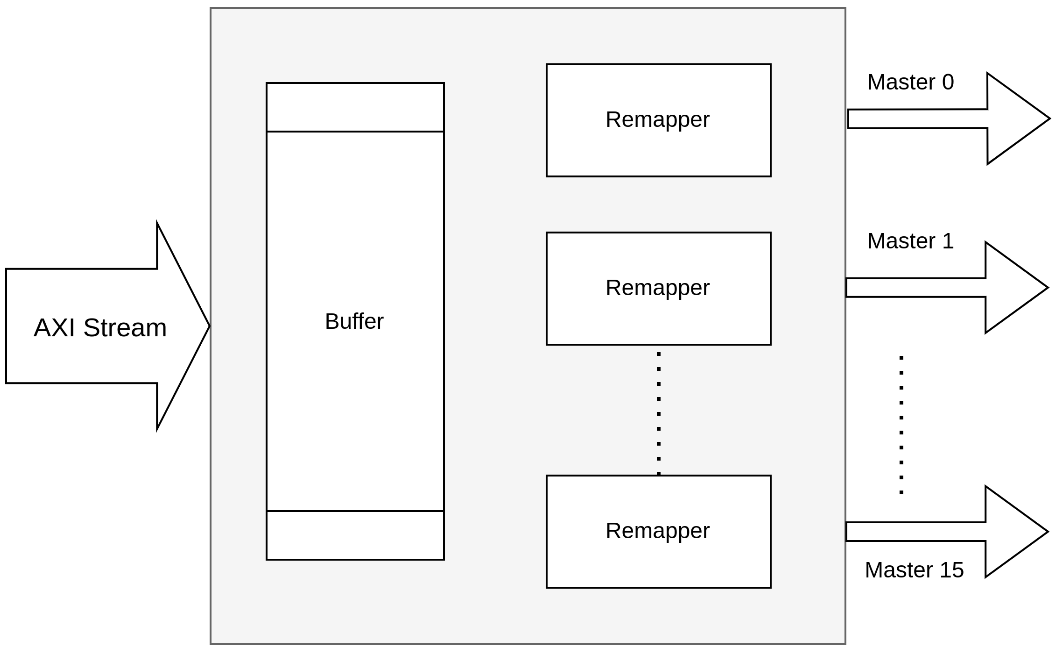


Figure 5: AXIS Broadcast replicating one into multiple Streams

Simulating the Example Design

The IP being Verilog HDL, can be simulated via a bunch of industry standard stimulus. For instance, it could be simulated via writing a Verilog Test-bench, or incorporating a soft processor that can stimulate this Broadcast IP. The bundled example design is stimulated via a Coco-tb based environment that iteratively stimulates all the master/slave pairs while also stress testing the data routing between them.

Synthesis and PR

Raptor Suite is armed with tools for Synthesis along with Post and Route capabilities and the generated postsynthesis and post-route and place netlists can be viewed and analyzed from within the Raptor. The generated bitstream can then be uploaded on an FPGA device to be utilized in hardware applications.

Test Bench

A Coco-tb based test bench can be found in the **/sim** repository formed after the generation of the IP. It generates a Broadcast IP with the required number of Masters. This test environment can be simulated with any Verilog HDL simulator of choice e.g., Verilator or Icarus. This simulates the generated IP under various test conditions including stimulating all slave interfaces individually and then some stress test cases where all the interfaces are stimulated together. This makes up for a total of 4 tests upon passing of which the IP can be verified functionally. The simulation can be easily run by clicking the "Simulate IP" button as shown in figure 6.

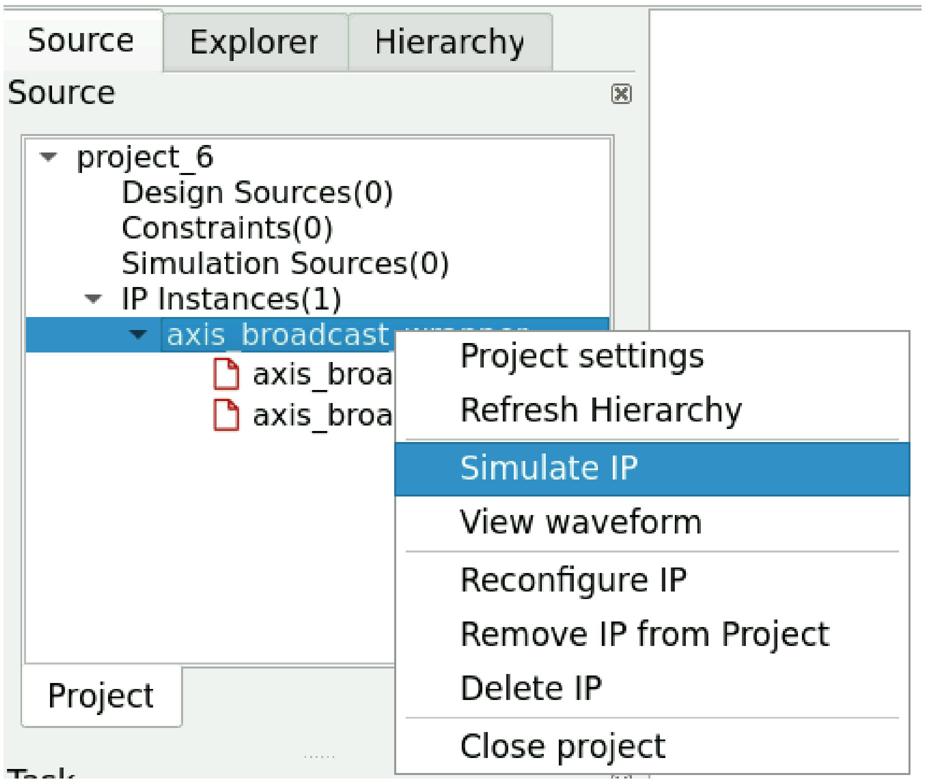


Figure 6: Simulate IP Window

The waveforms are also dumped in-depth analysis of the whole operation which can be seen by clicking the "View Waveform" button as shown in 7.

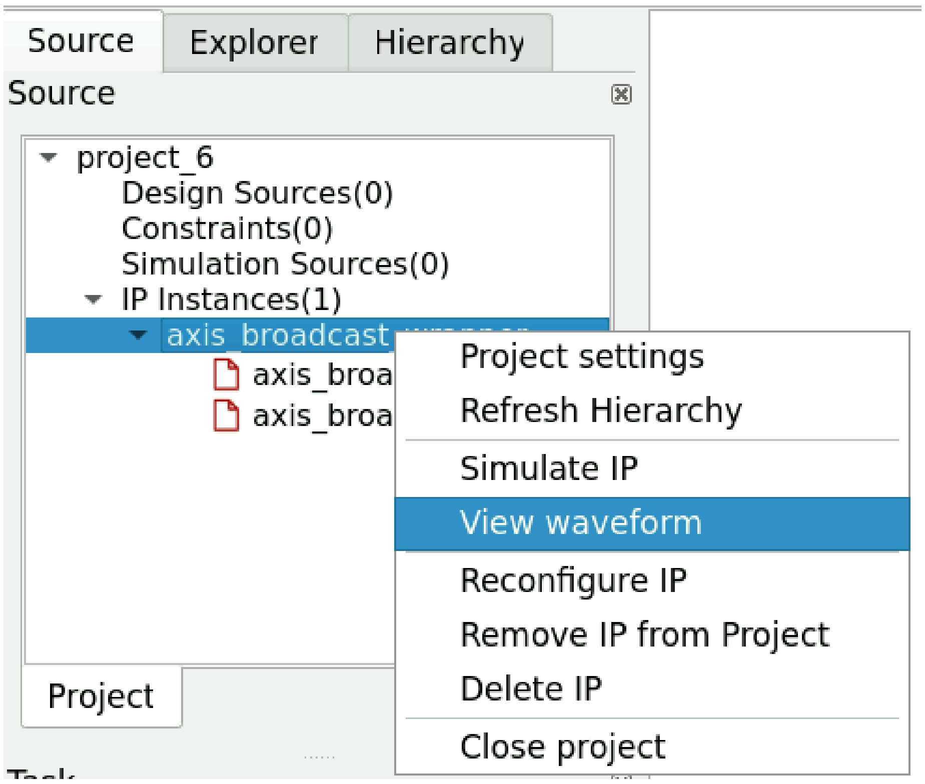
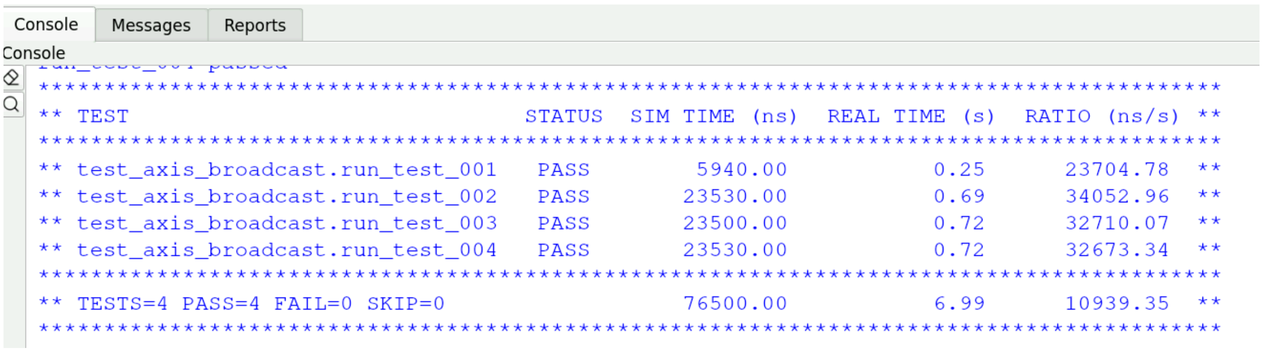


Figure 7: View Waveform Window

The simulation results are also displayed in the console window a glimpse of which can be seen in figure 8.



```
*********************************************************************************************
** TEST                                    STATUS  SIM TIME (ns)  REAL TIME (s)  RATIO (ns/s) **
*********************************************************************************************
** test_axis_broadcast.run_test_001         PASS       5940.00          0.25      23704.78  **
** test_axis_broadcast.run_test_002         PASS      23530.00          0.69      34052.96  **
** test_axis_broadcast.run_test_003         PASS      23500.00          0.72      32710.07  **
** test_axis_broadcast.run_test_004         PASS      23530.00          0.72      32673.34  **
*********************************************************************************************
** TESTS=4 PASS=4 FAIL=0 SKIP=0                       76500.00          6.99      10939.35  **
*********************************************************************************************
```

Figure 8: Simulation Results

Release

Release History

Date	Version	Revisions
February 29, 2024	0.1	Initial version AXI Stream Broadcast User Guide Document