# AXIS Switch v1.0

---

## *IP User Guide (Beta Release)*


December 5, 2023

RapidSilicon

# Contents

**IP Summary** **2**
**Overview** **3**
AXIS Switch . . . 3
Licensing . . . 4
**IP Specification** **5**
Standards . . . 5
IP Support Details . . . 6
Resource Utilization . . . 6
Port List . . . 7
Parameters . . . 8
**Design Flow** **9**
IP Customization and Generation . . . 9
**Example Design** **11**
Overview . . . 11
Simulating the Example Design . . . 11
Synthesis and PnR . . . 11
**Test Bench** **12**
**Release** **14**
Revision History . . . 14

# IP Summary

## Introduction

The AXIS Switch core is an AXI4-Streaming compliant customizable switch that is designed to be used in applications that require configurable routing between masters and slaves with multiple arbitration options.
The TDEST based routing uses RTL parameters configured before synthesis to control the routing by assigning a base TDEST pair to each master interface that is then used to generate a decode table which is then decoded by the respective slave interface based on the valid TDEST value. This transfer routes a request to an arbiter of one of the master interfaces.
The arbiter responds with a grant based on input from a priority encoder, and then the slave proceeds with the transfer. Arbitration can be performed at either transfer level or the transaction level. The TDEST based routing requires that the signal has at least $log_2$ number of bits.

## Features

- Supports 1-16 slaves.
- Supports 1-16 masters.
- Supports Round-Robin and LSB High Priority arbitration choices.
- Supports routing based on TDEST base/high pairs.
- Configurable data width.
- Configurable slave/master register types.
- Configurable offset in address arbitration.

# Overview

## AXIS Switch

AXIS Switch is configured via its AXI-streaming slave interface. It captures the number of slaves and masters and their respective addressing offsets and generate address configuration for each master-slave pair. The arbitration occurs via an embedded priority encoder. The Figure 1 shows the AXIS Switch core connected with multiple masters and slaves routing data in between them.

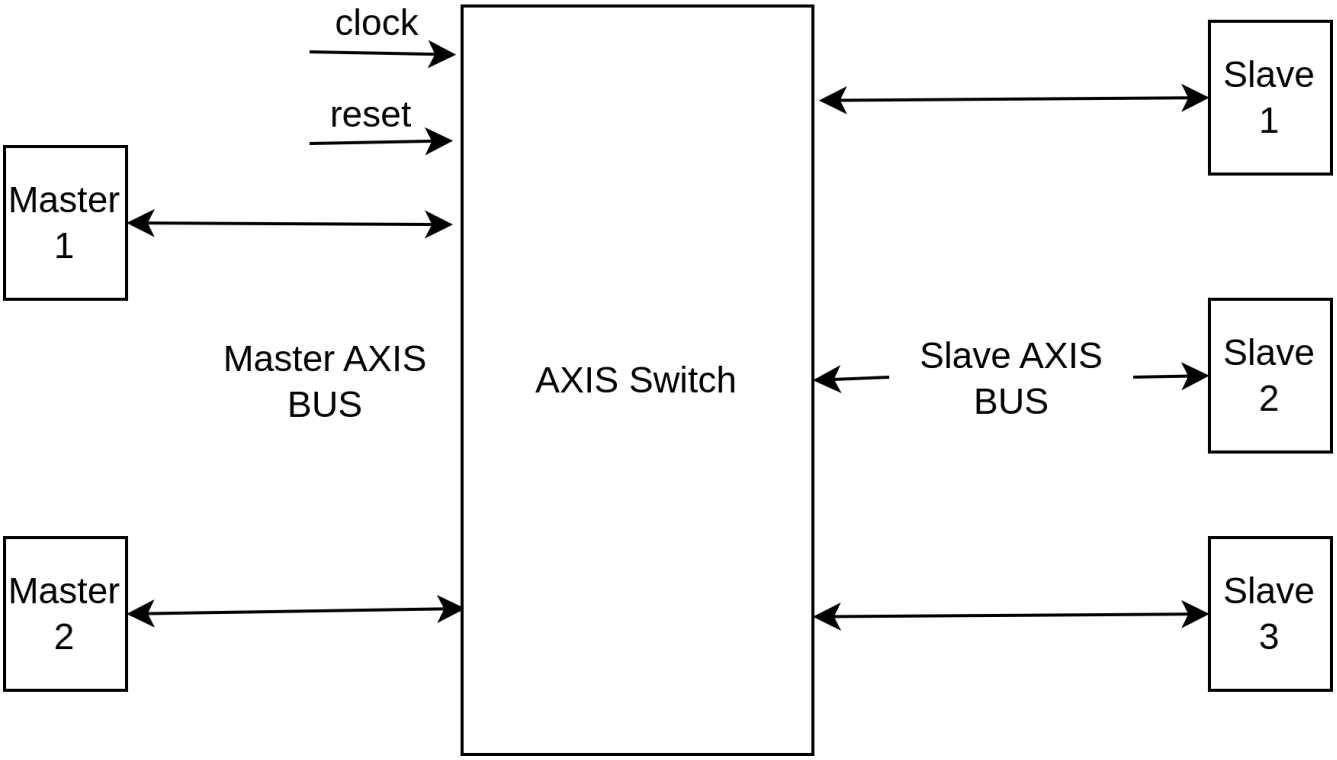



**Figure 1.** AXIS Switch

# Licensing

COPYRIGHT TEXT:

---



---

# IP Specification

The Figure 2 shows the internal block diagram of AXIS Switch. It consists of a priority encoder, an arbiter and an axis register module to be used on both slave and master sides.
The AXIS Switch is configured from the AXI interface on the rising edge of the AXI bus clock. All the modules interfaced with this AXIS switch run on the same AXI bus clock and hence follow a pipelined structure. The Switch generates the address configuration based on the configuration and then the arbitration module routes the traffic between the slave-master pairs. This Switch forms the base for an AXIS Interconnect.
The whole module combines sequential and combinational logic to make sure it operates effectively without breaking.

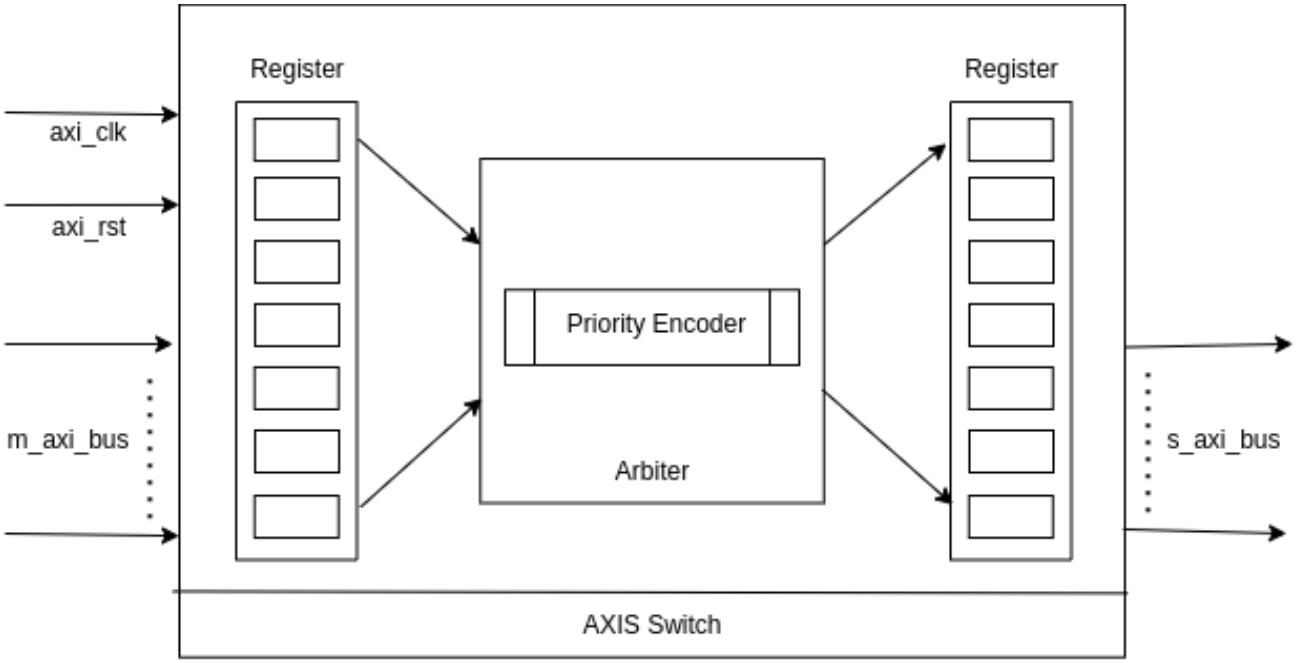



**Figure 2.** Top Module

## Standards

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

## IP Support Details

The Table 1 gives the support details for AXIS Switch.

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

**Table 1.** IP Details

## Resource Utilization

The parameters for computing the maximum and 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** |
| | Master Interfaces | 1 | LUT | 5 |
| | Slave Interfaces | 1 | Registers | 2 |
| | S ID Width | 1 | - | - |
| | M Reg Type | 0 | - | - |
| | LSB High Priority | 0 | - | - |
| | Type Round Robin | 0 | - | - |
| | Data Width | 8 | - | - |
| Maximum Resource | **Options** | **Configuration** | **Resources** | **Utilized** |
| | Master Interfaces | 3 | LUT | 521 |
| | Slave Interfaces | 8 | Registers | 200 |
| | ID En | ON | DSP | 3 |
| | TID En | ON | - | - |

**Table 2.** AXIS Switch Resource Utilization

## Ports

Table 3 lists the top interface ports of the AXIS Switch.

| 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 3.** AXIS Switch Interface

## Parameters

Table 4 lists the parameters of the AXIS Switch.

| Parameter | Values | Default Value | Description |
|---|---|---|---|
| S COUNT | 1 - 16 | 4 | Number of Slave Interfaces |
| M COUNT | 1 - 16 | 4 | Number of Master Interfaces |
| DATA WIDTH | 1 - 1024 | 8 | Data Width for each transfer |
| USER WIDTH | 1 - 1024 | 1 | Data Width for user defined sideband information |
| S ID WIDTH | 1 - 16 | 8 | Slave side ID Width |
| M DEST WIDTH | 1 - 8 | 1 | Master side Destination Width |
| M REG TYPE | Bypass, Simple Buffer, Skid Buffer | Skid Buffer | Master side Register Type |
| S REG TYPE | Bypass, Simple Buffer, Skid Buffer | Bypass | Slave side Register Type |
| M BASE | 0 - 15 | 0 | Address configuration for Master Interface from Base |
| M TOP | 0 - 15 | 0 | Address configuration for Master Interface from Top |
| ID EN | 0 / 1 | 0 | ID Enable |
| USER EN | 0 / 1 | 1 | User Data Enable |
| LSB HIGH PRIORITY | 0 / 1 | 1 | LSB Priority Selection |
| TYPE ROUND ROBIN | 0 / 1 | 1 | Round Robin Architecture |
| TID | 0 / 1 | 0 | Enable update of Transfer ID |
| IP TYPE | - | AXISSWTH | Type of Peripheral |
| IP VERSION | - | <ip_version> | Version of Peripheral |
| IP ID | - | <date_and_time> | Date and Time of the generated Peripheral |

**Table 4.** Parameters

# Design Flow

## IP Customization and Generation

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

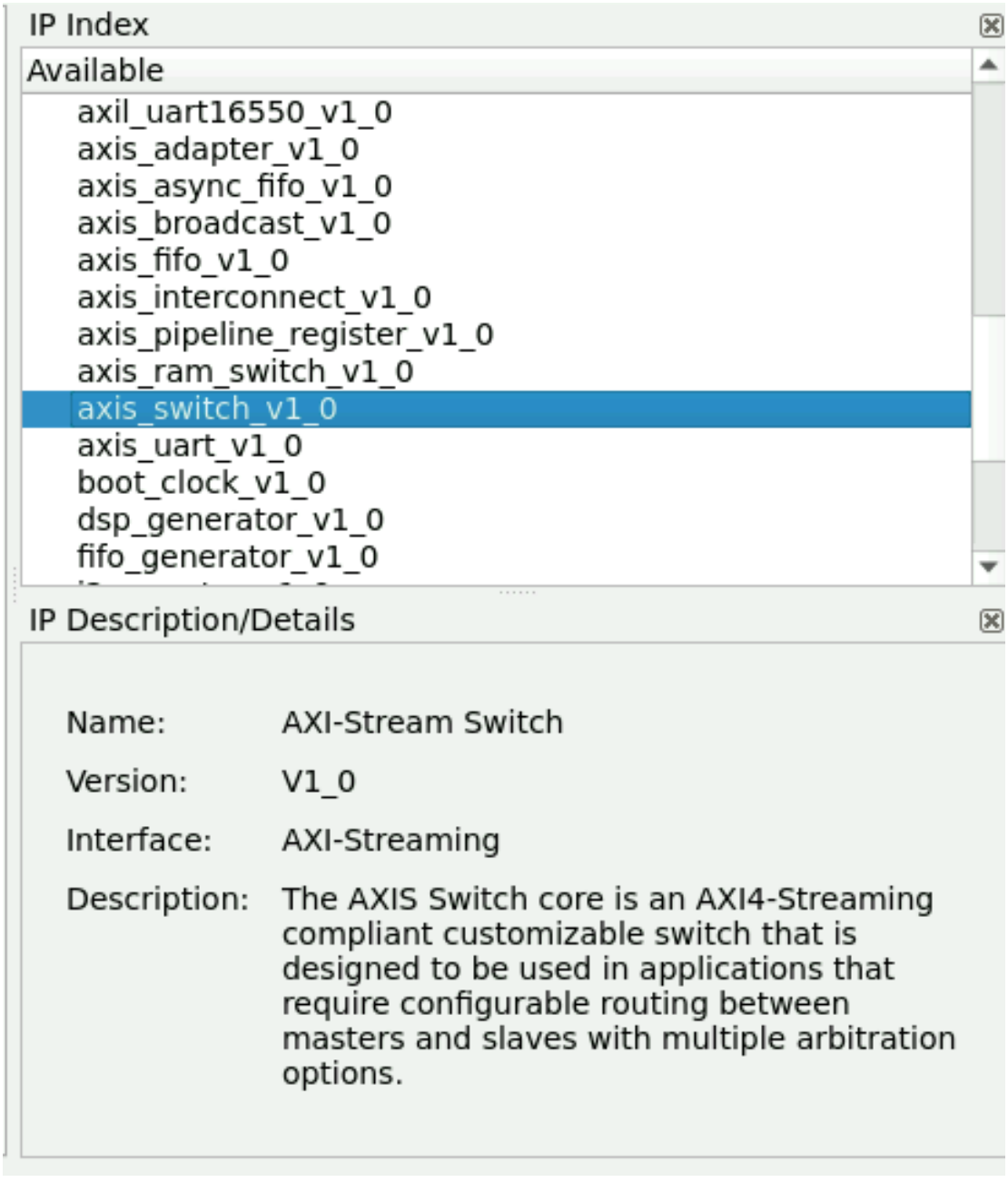



**Figure 3.** IP list

**Parameters Customization:** From the IP configuration window, the parameters of the Switch can be configured and Switch features can be enabled for generating a customized Switch 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 Switch.

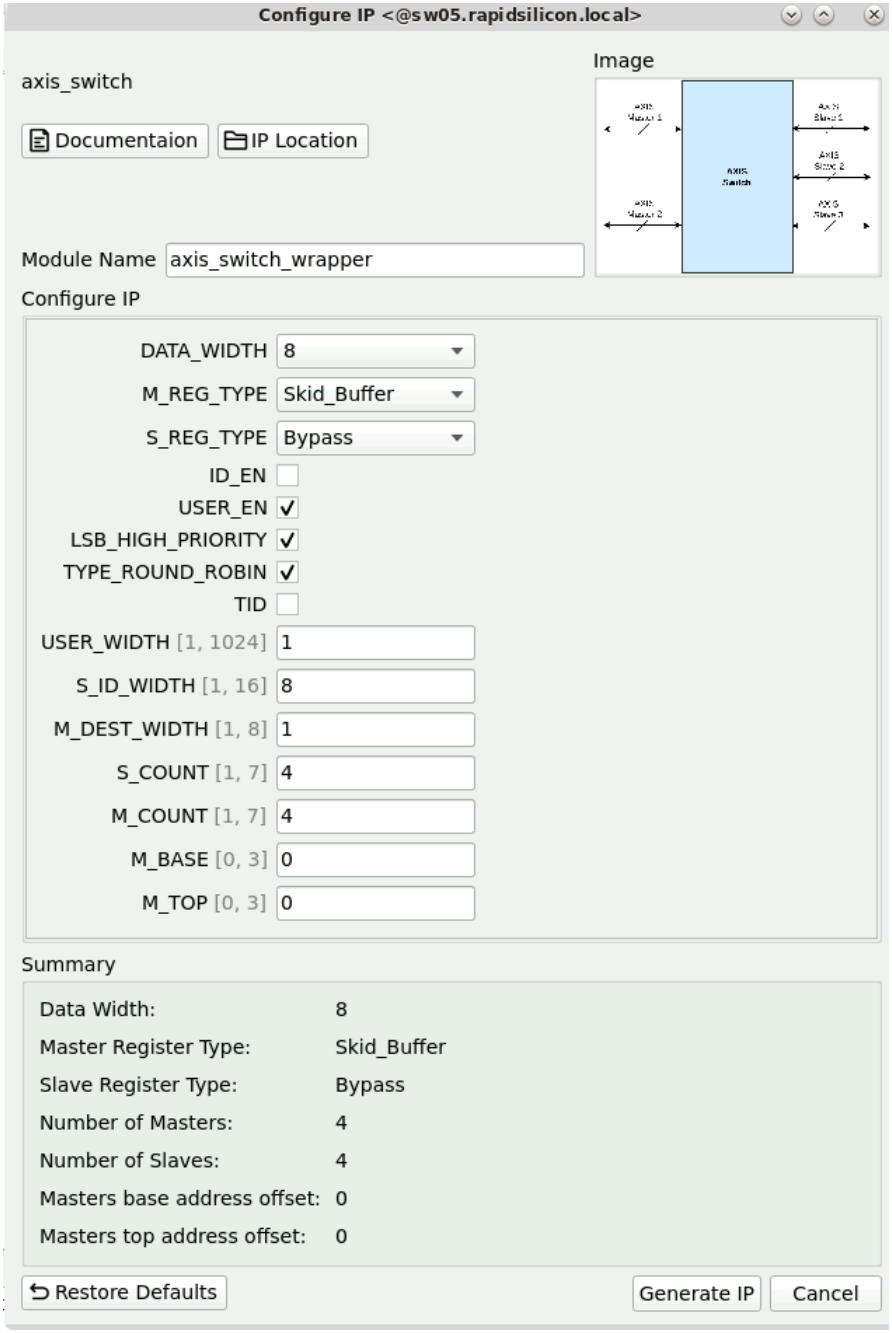



**Figure 4.** IP Configuration

# Example Design

## Overview

This AXIS RAM Switch can be utilized in any system that has multiple master and slave pairs and there is a need to route the traffic between them. This RAM Switch can also be utilized to make an Interconnect by adding several required FIFOs and data width adapters. In its native form, it acts as a minimalist data routing switch with the smallest footprint. One such interpretation of this AXIS Switch utilized in the AXIS Interconnect can be visualized in Figure 5.

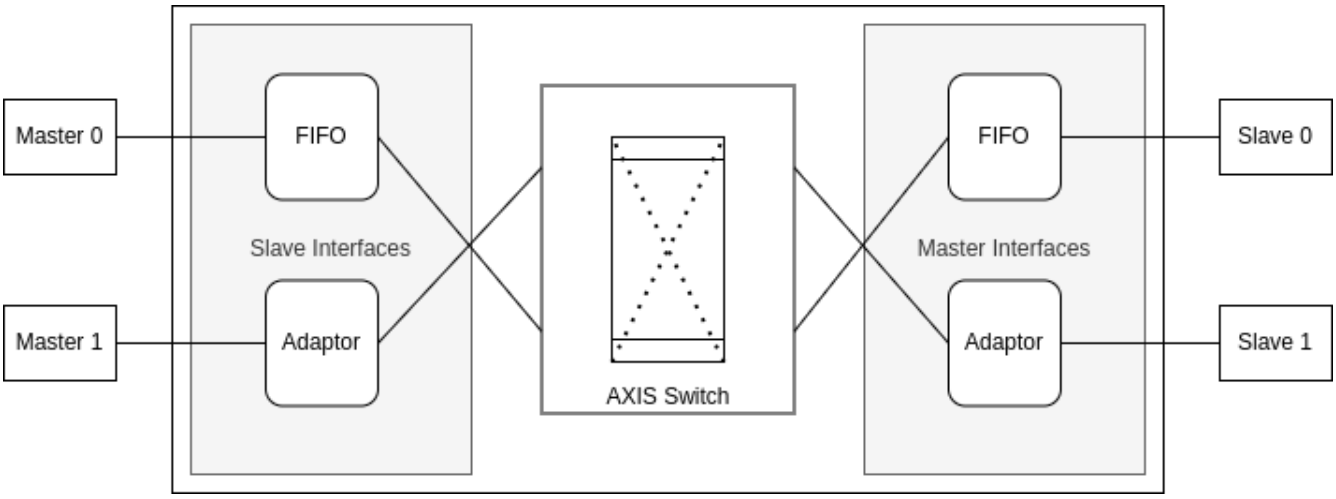



**Figure 5.** AXIS Switch in Interconnect

## 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 RAM Switch. 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 PnR

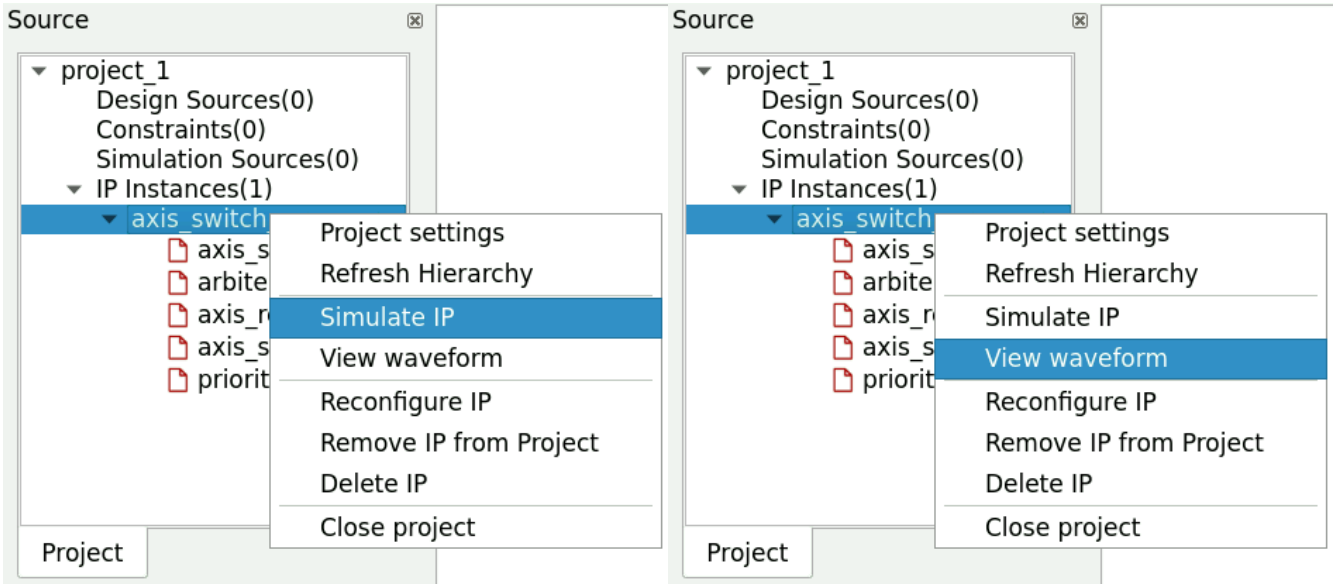
Raptor Suite is armed with tools for **Synthesis** along with **Post and Route** capabilities and the generated post-synthesis and post-route and place netlists can be viewed and analyzed from within the Raptor. The generated bitstream can then be uploaded on an FPGA device to be utilized in hardware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

- ID EN = 1
- TID = 1
- S ID WIDTH = 16
- M DEST WIDTH = 8
- S COUNT = 4
- M COUNT = 4

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 master and slave interfaces individually and then some stress test cases where all the interfaces are stimulated together. This makes up for a total of 25 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 6a. The waveform can also be viewed via the built-in wave-veiwer by clicking the "View waveform" button as shown in figure 6b.




**(a)** Simulate IP Window **(b)** View Waveform Window

**Figure 6.** IP Source Window

The simulation results are also displayed in the console window a glimpse of which can be seen in figure 7.

```
**************************************************************************
** TEST                                          STATUS  SIM TIME (ns)
**************************************************************************
** test_axis_switch.run_test_001                 PASS        7320.00
** test_axis_switch.run_test_002                 PASS        7330.00
** test_axis_switch.run_test_003                 PASS        7330.00
** test_axis_switch.run_test_004                 PASS        7330.00
** test_axis_switch.run_test_005                 PASS       23530.00
** test_axis_switch.run_test_006                 PASS       23530.00
** test_axis_switch.run_test_007                 PASS       23530.00
** test_axis_switch.run_test_008                 PASS       23530.00
** test_axis_switch.run_test_009                 PASS       23520.00
** test_axis_switch.run_test_010                 PASS       23520.00
** test_axis_switch.run_test_011                 PASS       23520.00
** test_axis_switch.run_test_012                 PASS       23520.00
** test_axis_switch.run_test_013                 PASS       23530.00
** test_axis_switch.run_test_014                 PASS       23530.00
** test_axis_switch.run_test_015                 PASS       23530.00
** test_axis_switch.run_test_016                 PASS       23530.00
** test_axis_switch.run_test_tuser_assert_001    PASS         430.00
** test_axis_switch.run_test_tuser_assert_002    PASS         430.00
** test_axis_switch.run_test_tuser_assert_003    PASS         430.00
** test_axis_switch.run_test_tuser_assert_004    PASS         430.00
** test_axis_switch.run_arb_test_001             PASS         930.00
** test_axis_switch.run_stress_test_001          PASS       20380.00
** test_axis_switch.run_stress_test_002          PASS       72370.00
** test_axis_switch.run_stress_test_003          PASS       67260.00
** test_axis_switch.run_stress_test_004          PASS       69970.00
**************************************************************************
** TESTS=25 PASS=25 FAIL=0 SKIP=0                          544260.03
**************************************************************************
```

**Figure 7.** Simulation Results

# Revision History

| Date | Version | Revisions |
|---|---|---|
| December 5, 2023 | 0.01 | Initial version AXIS Switch User Guide Document |