



AXI4-Stream Async FIFO (Beta Release)

Version 0.1

May 16, 2023

Copyright

Copyright © 2021 Rapid Silicon. All rights reserved. This document may not, in whole or part, be reproduced, modified, distributed, or publicly displayed without prior written consent from Rapid Silicon ("Rapid Silicon").

Trademarks

All Rapid Silicon trademarks are as listed at www.rapidsilicon.com. Synopsys and Synplify Pro are trademarks of Synopsys, Inc. Aldec and Active-HDL are trademarks of Aldec, Inc. Modelsim and Questa are trademarks or registered trademarks of Siemens Industry Software Inc. or its subsidiaries in the United States or other countries. All other trademarks are the property of their respective owners.

Disclaimers

NO WARRANTIES: THE INFORMATION PROVIDED IN THIS DOCUMENT IS "AS IS" WITHOUT ANY EXPRESS OR IMPLIED WARRANTY OF ANY KIND INCLUDING WARRANTIES OF ACCURACY, COMPLETENESS, MERCHANTABILITY, NONINFRINGEMENT OF INTELLECTUAL PROPERTY, OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL RAPID SILICON OR ITS SUPPLIERS BE LIABLE FOR ANY DAMAGES WHATSOEVER (WHETHER DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL, INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OF OR INABILITY TO USE THE INFORMATION PROVIDED IN THIS DOCUMENT, EVEN IF RAPID SILICON HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. BECAUSE SOME JURISDICTIONS PROHIBIT THE EXCLUSION OR LIMITATION OF CERTAIN LIABILITY, SOME OF THE ABOVE LIMITATIONS MAY NOT APPLY TO YOU.

Rapid Silicon may make changes to these materials, specifications, or information, or to the products described herein, at any time without notice. Rapid Silicon makes no commitment to update this documentation. Rapid Silicon reserves the right to discontinue any product or service without notice and assumes no obligation to correct any errors contained herein or to advise any user of this document of any correction if such be made. Rapid Silicon recommends its customers obtain the latest version of the relevant information to establish that the information being relied upon is current and before ordering any products.

Contents

IP Summary	3
Introduction	3
Features	3
Overview	4
AXIS Async FIFO	4
IP Specification	5
Standards	5
IP Support Details	6
Parameters	6
Port List	7
Resource Utilization	8
Design Flow	9
IP Customization and Generation	9
Parameters Customization	10
Synthesis and PR	10
Test Bench	11
Release	12
Release History	12

IP Summary

Introduction

An AXI Streaming Asynchronous FIFO is a type of FIFO (First-In-First-Out) buffer that is used to transfer data between two different clock domains in a System-on-Chip (SoC) design. The basic operation of an AXI Streaming Asynchronous FIFO involves writing data into the FIFO at one clock domain and reading data from the FIFO at another clock domain. The two clock domains may have different clock frequencies and are typically asynchronous to each other.

An AXI Stream FIFO is a commonly used module in digital design that acts as a data mover between two AXI Stream interfaces at different clock domains. It provides a buffer to store a stream of data items coming from the input interface, and allows them to be read out in the same order from the output interface. The FIFO also have various options and configurations to control its behavior, such as the depth of the buffer, the type of synchronization mechanism used, and the way overflow or underflow situations are handled. The AXI Stream interface is a widely used standard for streaming data in digital systems, particularly in the context of FPGA and ASIC designs. It consists of two unidirectional channels, a data channel and a control channel, both of which have a fixed format and timing. The data channel carries a continuous stream of data items, each of which can have a fixed or variable width, and the control channel includes a few signals that indicate the start and end of a transfer, as well as any error or flow control conditions.

Features

- The FIFO supports data transfer between two different clock domains.
- The FIFO supports data transfer widths of 8, 16, 32, or 64 bits.
- The FIFO can be configured with a depth of up to 2^{16} words.
- The FIFO includes a programmable flag that indicates when the FIFO is empty or full.
- The FIFO includes a programmable flag that indicates when an error condition has occurred, such as a data overrun or underrun.
- The FIFO supports AXI4-Stream interfaces.

Overview

AXIS Async FIFO

An AXI streaming asynchronous FIFO is an IP that allows for the transfer of large, continuous data streams between two components of a larger digital system that uses the AXI streaming protocol. The AXI streaming asynchronous FIFO is designed specifically for data streams that consist of a large number of continuous data elements, such as video or audio signals, which are transferred in a continuous, sequential manner. This IP provides a buffering mechanism that ensures the reliable transfer of data between the write-side and read-side interfaces. An AXIS Async FIFO can be used in a wide range of digital systems, including multimedia systems, network switches, and digital signal processing (DSP) systems, where efficient and reliable data transfer is essential for proper system operation. A block diagram for the AXIS Async FIFO IP is shown in Figure 1.

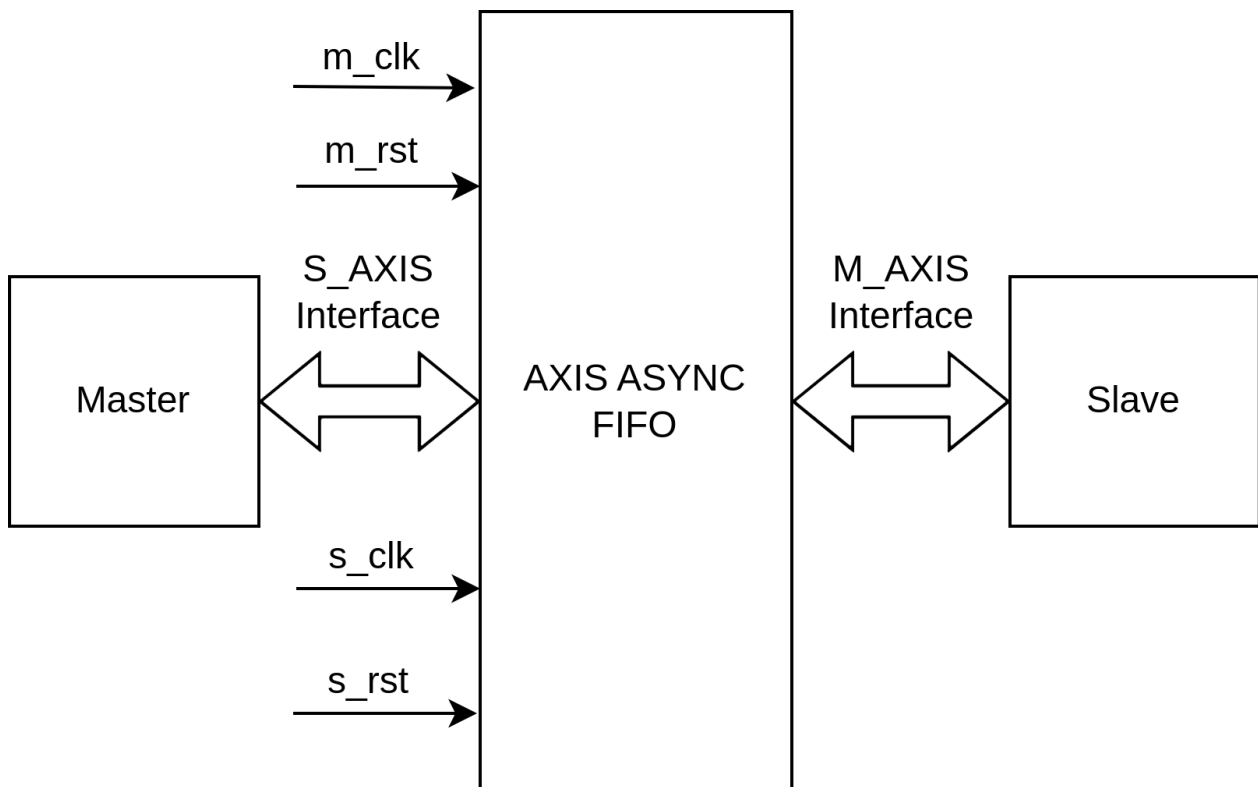


Figure 1: AXIS Async FIFO Block Diagram

IP Specification

The IP has several status signals to indicate the state of the FIFO. The `status_overflow` signal is used to indicate when the FIFO has overflowed. The `status_bad_frame` signal is used to indicate when a frame marked as bad has been detected. The `status_good_frame` signal is used to indicate when a good frame has been detected. The `status_full` signal is used to indicate when the FIFO is full. The `status_empty` signal is used to indicate when the FIFO is empty. An internal block diagram can be seen in Figure 2. The Asynchronous FIFO has a variety of different parameters that makes it a very versatile FIFO to be used in a variety of systems that require the use of independent clocks and resets for both the master and the slave to ensure the maximum throughput is achieved. The data transfer in an asynchronous FIFO is based on handshaking signals. When the FIFO is empty, the read pointer sends a signal to the write pointer to request data. The write pointer then fills the FIFO with data and sends a signal to the read pointer to indicate that data is available. The read pointer then retrieves the data from the FIFO.

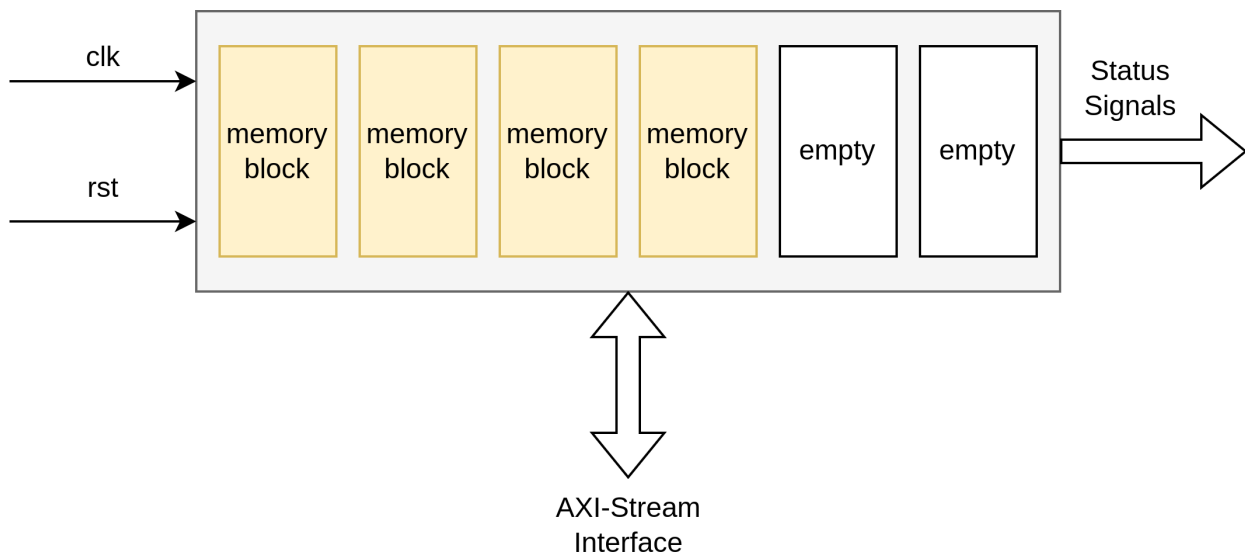


Figure 2: AXIS Async FIFO Internal Diagram

Standards

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

IP Support Details

The Table 1 gives the support details for AXIS Async FIFO.

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	-	Python	Cocotb	-	Raptor (Verific)	Raptor (Icarus)	Raptor

Table 1: IP Details

Parameters

Table 2 lists the parameters of the AXIS Async FIFO.

Parameter	Values	Default Value	Description
DEPTH	16,32,...,32768	4096	FIFO DEPTH
DATA WIDTH	8,16,32,...,1024	8	Width of AXI stream data in bits
LAST ENABLE	0 / 1	1	Propagate tlast signal
ID ENABLE	0 / 1	1	Propagate tid signal
ID WIDTH	1-16	8	tid signal width
DEST ENABLE	0 / 1	1	Propagate tdest signal
DEST WIDTH	1 - 8	8	tdest signal width
USER ENABLE	0 / 1	1	Propagate tuser signal
RAM PIPELINE	1 - 9	1	Number of pipeline registers
FRAME FIFO	0 / 1	1	Frame FIFO mode - operate on frames instead of cycles, when set, m_axis_tvalid will not be deasserted within a frame
USER BAD FRAME VALUE	1 - 99	1	FIFO bad frame marker for value
USER BAD FRAME MASK	1 - 99	1	FIFO Mask for bad frame marker
DROP BAD FRAME	0 / 1	1	Drop frames marked bad
DROP WHEN FULL	0 / 1	1	Drop incoming frames when full

Table 2: Parameters

Port List

Table 3 lists the top interface ports of the AXIS Async FIFO.

Signal Name	I/O	Description
AXI-Stream Slave Interface		
s_clk	I	AXI4-Stream Clock
s_rst	I	AXI4-Stream Asynchronous RESET
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-Stream Master Interface		
m_clk	I	AXI4-Stream Clock
m_rst	I	AXI4-Stream Asynchronous RESET
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
Status Signals		
status_overflow	O	AXI4-Stream user defined sideband information
status_bad_frame	O	AXI4-Stream user defined sideband information
status_good_frame	O	AXI4-Stream user defined sideband information
status_full	O	AXI4-Stream user defined sideband information
status_empty	O	AXI4-Stream user defined sideband information

Table 3: AXIS Async FIFO Interface

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 Utilized	
	Options	Configuration	Resources	Utilized
Minimum Resource	DATA WIDTH	8	LUTs	102
	DEPTH	6		
	LAST EN	0		
	RAM PIPELINE	0	Registers	120
	FRAME FIFO	0		
	OUT FIFO EN	0		
	BAD FRAME VALUE	0	BRAM	1
	DROP BAD FRAME	0		
	DROP WHEN FULL	0		
Maximum Resource	Options	Configuration	Resources	Utilized
	LAST EN	1	LUTs	3413
	DEPTH	32768		
	DATA WIDTH	1024		
	ID WIDTH	8		
	DEST WIDTH	8		
	USER WIDTH	1024	Registers	2327
	RAM PIPELINE	1		
	FRAME FIFO	1		
	OUT FIFO EN	1	BRAM	124
	BAD FRAME VALUE	1		
	DROP BAD FRAME	1		
	DROP WHEN FULL	1		

Table 4: Resource Utilization

Design Flow

IP Customization and Generation

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

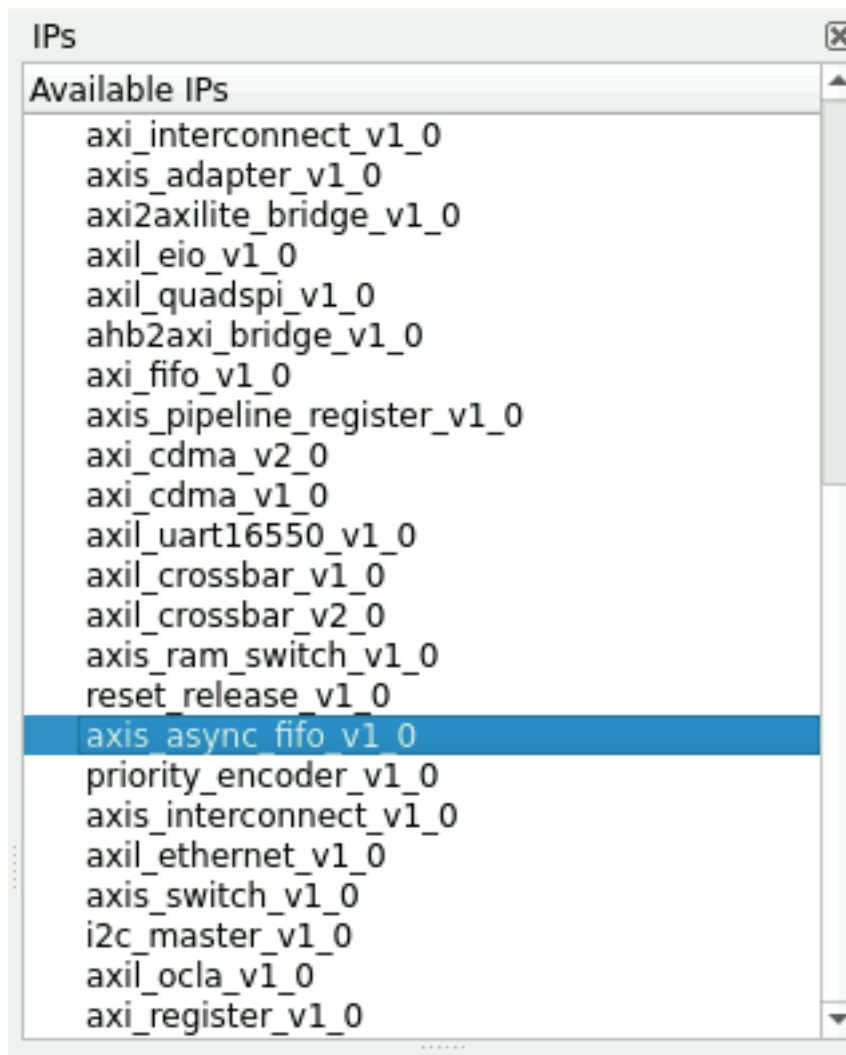
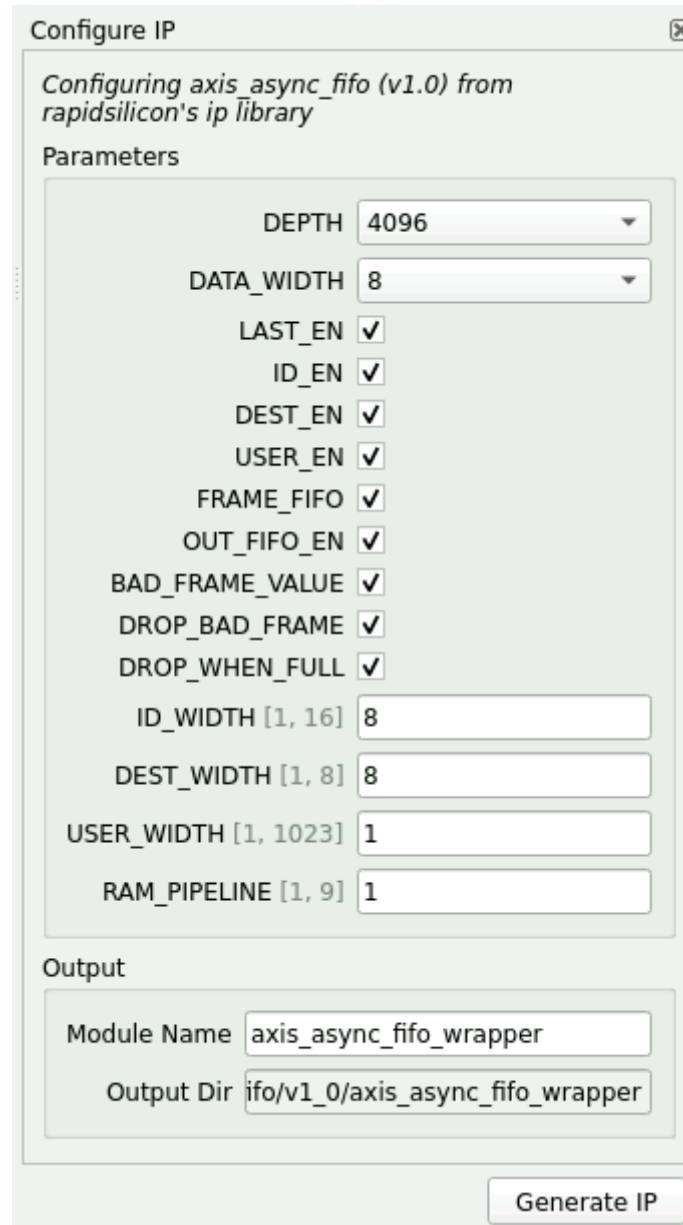


Figure 3: IP list

Parameters Customization

From the IP configuration window, the parameters of AXIS Async FIFO can be configured and IP features can be enabled for generating a customized AXIS Async FIFO 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 Async FIFO.



Configure IP

Configuring *axis_async_fifo* (v1.0) from *rapidsilicon's ip library*

Parameters

DEPTH 4096

DATA_WIDTH 8

LAST_EN ☒

ID_EN ☒

DEST_EN ☒

USER_EN ☒

FRAME_FIFO ☒

OUT_FIFO_EN ☒

BAD_FRAME_VALUE ☒

DROP_BAD_FRAME ☒

DROP_WHEN_FULL ☒

ID_WIDTH [1, 16] 8

DEST_WIDTH [1, 8] 8

USER_WIDTH [1, 1023] 1

RAM_PIPELINE [1, 9] 1

Output

Module Name axis_async_fifo_wrapper

Output Dir ifo/v1_0/axis_async_fifo_wrapper

Generate IP

Figure 4: IP Configuration

Synthesis and PR

Raptor Suite is armed with tools for Synthesis along with Post and Route capabilities and the generated post-synthesis and post-route and place net-lists can be viewed and analyzed from within the Raptor. The generated bit-stream 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. The AXIS Async FIFO simulation is based on Cocotb. It has a complete environment that extensively tests AXIS Async FIFO as a DUT. This test environment can be simulated with any Verilog HDL simulator of choice e.g., Verilator or Icarus. It has 25 tests in total, 12 write tests, 12 read tests and a stress test that stimulates all the interfaces in many different conditions to ensure complete coverage. The simulation can be run from Raptor IP Catalog. User can interact with the waveform from within Raptor

Release

Release History

Date	Version	Revisions
May 16, 2023	0.1	Initial version AXIS Async FIFO User Guide Document