AXI4-Stream FIFO v4.2

LogiCORE IP Product Guide

Vivado Design Suite

PG080 October 30, 2019

XILINX

Table of Contents

Introduction

The LogiCORE™ IP AXI4-Stream FIFO core allows memory mapped access to a AXI4-Stream interface. The core can be used to interface to AXI4-Stream IPs, similar to the LogiCORE IP AXI Ethernet core, without having to use a full DMA solution.

The principal operation of this core allows the write or read of data packets to or from a device without any concern over the AXI4-Stream interface signaling. You can easily manage the AXI4-Stream interfaces as they are transparent.

Features

- 32-bit AXI4-Lite slave interface
- Configurable data interface type (AXI4 or AXI4-Lite)
- Configurable 32, 64, 128, 256, or 512-bit AXI4-Stream data interface. For AXI4-Lite, the FIFO data width is 32 bits. For AXI4, the FIFO data width matches the AXI4-Stream data width.
- Independently configurable internal TX and RX data FIFOs
- Full duplex operation
- Supports AXI Ethernet basic mode
- Provides interrupts for error and status conditions
- TX and RX cut-through mode

LogiCORE™ IP Facts Table	
Core Specifics	
Supported Device Family(1)	UltraScale+™ families UltraScale™ families Zynq®-7000 SoC 7 series FPGAs
Supported User Interfaces	AXI4, AXI4-Lite, AXI4-Stream
Resources	See Resource Utilization.
Provided with Core	
Design Files	VHDL
Example Design	VHDL
Test Bench	VHDL
Constraints File	Not Provided
Simulation Model	Not Provided
Supported S/W Driver(2)	Standalone
Tested Design Flows(3)	
Design Entry	Vivado® Design Suite
Simulation	For support simulators, see the Xilinx Design Tools: Release Notes Guide.
Synthesis	Vivado Synthesis
Support	
Release Notes and Known Issues	Master Answer Record: 54447
All Vivado IP Change Logs	Master Vivado IP Change Logs: 72775
Xilinx Support web page	

Notes:

1. For a complete list of supported devices, see the Vivado IP catalog.
2. Linux OS and driver support information is available from the Linux AXI4-Stream FIFO Standalone Driver Page.
3. For the supported versions of third-party tools, see the Xilinx Design Tools: Release Notes Guide.

Overview

Figure 1-1 shows the major components in the AXI4-Stream FIFO core that consists of the following:

- AXI Interface block with an AXI4/AXI4-Lite Slave interface
- Interrupt controller
- Registers space
- Receive control module
- Transmit control module
- Receive FIFO for the receive data and length
- Transmit FIFO for the transmit data and the length

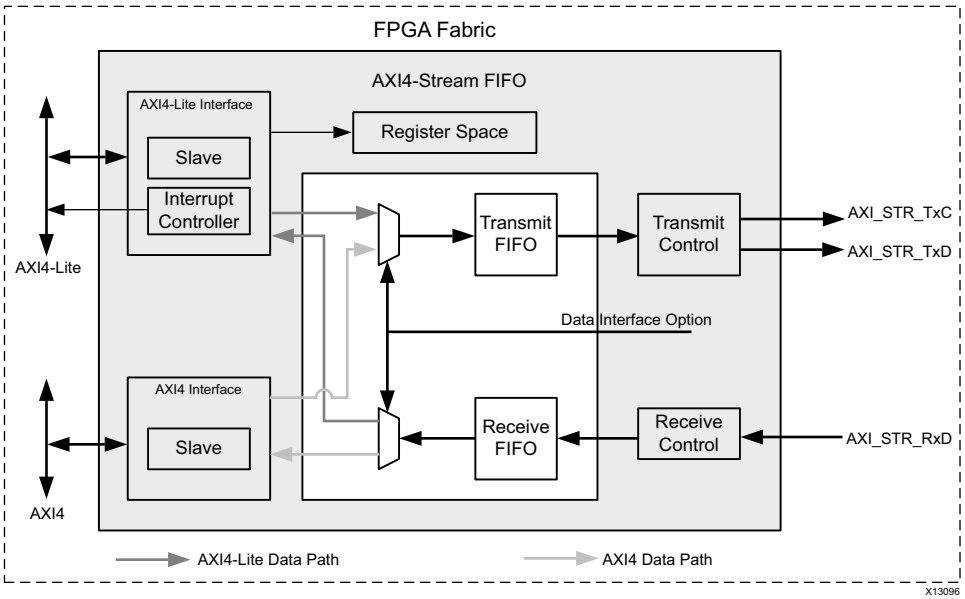

Figure 1-1: **AXI4-Stream FIFO Core Block Diagram**

Note: Supported data widths for **`AXI4_STR_TxC/AXI4_STR_TxD/AXI4_STR_RxD`** are 32/64/128/256/512 bits.

The AXI4-Stream FIFO core was designed to provide memory-mapped access to an AXI4-Stream interface connected to other IP, such as the AXI Ethernet core. Systems must be built through the Vivado® Design Suite to attach the AXI4-Stream FIFO core, AXI Ethernet core, processor, memory, interconnect the buses, clocking, and additional embedded components.

Applications

The AXI4-Stream FIFO core converts AXI4/AXI4-Lite transactions to and from AXI4-Stream transactions, and can be used in Ethernet applications and others that use packet communication.

Unsupported Features

This core does not support asynchronous clock (independent clock) mode.

Licensing and Ordering

This Xilinx® LogiCORE™ IP module is provided at no additional cost with the Xilinx Vivado Design Suite under the terms of the Xilinx End User License.

Information about this and other Xilinx LogiCORE IP modules is available at the Xilinx Intellectual Property page. For information on pricing and availability of other Xilinx LogiCORE IP modules and tools, contact your local Xilinx sales representative.

Product Specification

Standards

This core complies with both the AMBA® AXI4-Stream Protocol Specification and the AMBA AXI4 Protocol Specification.

Performance

To measure the performance (F_{MAX}) of the AXI4-Stream FIFO core, it was added as the Device Under Test (DUT) to a Virtex®-7 FPGA as shown in Figure 2-1.

Because the core is used without other design modules in the FPGA, the utilization and timing numbers reported in this section are estimates only. When this core is combined with other designs in the system, the utilization of FPGA resources and timing of the design can vary from the results reported here.

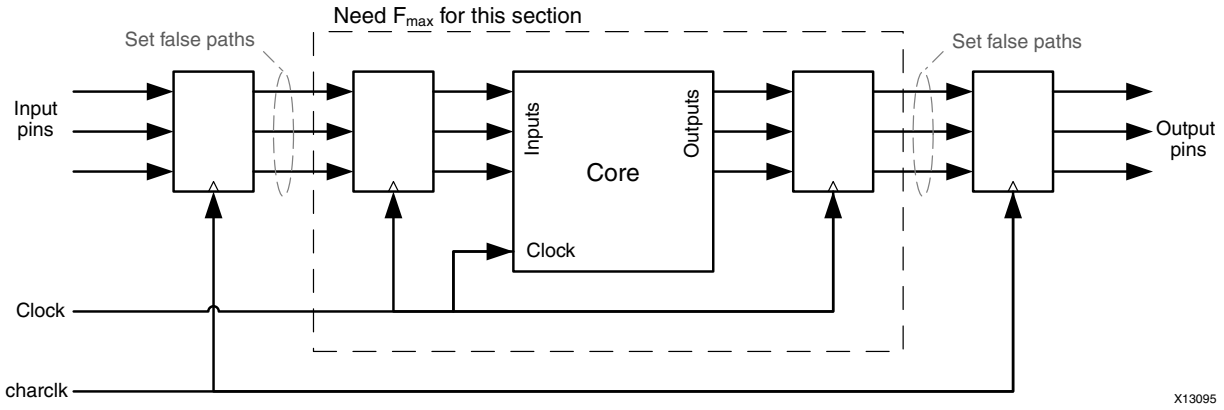

Figure 2-1: **Virtex-7 FPGA with the AXI4-Stream FIFO Core**

Maximum Frequencies

When targeting Virtex-7 devices, the AXI4-Stream FIFO core can operate up to 300 MHz with the following configuration:

- Data Interface Option: AXI4
- AXI4 Data Width: 64
- Transmit/Receive FIFO Depth: 4096 locations

The F_{MAX} is influenced by the exact system and is provided for guidance. It is not a guaranteed value across all systems.

Latency

Depending on the configuration of the core, the latency between AXI4 to AXI4-Stream varies. For example, if the core is configured for store and forward mode, AXI4-Stream transactions might not start until the entire packet is written into the Transmit FIFO from the AXI4-Lite/AXI4 interface. If the core is configured for TX cut-through/RX cut-through mode, the AXI4-Stream transactions start three clocks after WDATA is accepted from the AXI4-Lite/AXI4 interface.

Throughput

The throughput varies depending on the configuration of the AXI4-Stream FIFO core. For example, if the Data Interface is configured as AXI4-Lite, the throughput is less because the core can accept WDATA once in three clock cycles. If the Data Interface is configured as AXI4, the through put is three times more than the AXI4-Lite interface. Table 2-1 and Table 2-2 show the throughput numbers for store-and-forward mode and cut-through mode. Clock frequency is kept constant at 100 MHz to calculate the throughput.

Table 2-1: **AXI4-Stream FIFO Transmit Throughput**

Interface	Frequency (In MHz)	Packet Length (In Bytes)	Maximum Data Throughput (MBytes/sec)	
			Store-and-Forward Mode	Cut-Through Mode
AXI4-Lite	100	8 KB	64.88	77.51
AXI4	100	8 KB	198.49	393.65

Table 2-2: **AXI4-Stream FIFO Receive Throughput**

Interface	Frequency (In MHz)	Packet Length (In Bytes)	Maximum Data Throughput (MBytes/sec)	
			Store-and-Forward Mode	Cut-Through Mode
AXI4-Lite	100	8 KB	66.24	78.99
AXI4	100	8 KB	196.97	370.09

Resource Utilization

For details about resource utilization, visit Performance and Resource Utilization.

Port Descriptions

The AXI4-Stream FIFO has three AXI4-Stream interfaces: one for transmitting data, one for transmit control, and one for receiving data.

When using AXI4-Stream FIFO core with the AXI Ethernet core, connect the three AXI4-Stream interfaces listed:

1. AXI_STR_TXD – AXI4-Stream Transmit Data
2. AXI_STR_TXC – AXI4-Stream Transmit Control
3. AXI_STR_RXD – AXI4-Stream Receive Data

The AXI4-Stream Transmit Control Interface supports the transmit protocol of AXI Ethernet cores. The AXI4-Stream Transmit Control Interface is used by the AXI Ethernet core for partial CSUM off-loading of extended VLAN features. The AXI4-Stream FIFO core does not support any advanced features and drives constant values on this interface. The AXI4-Stream FIFO core follows the handshake requirements as defined by the AXI Ethernet Core. For more details, see *AXI Ethernet Subsystem Product Guide* (PG138) [Ref 4].

The AXI4-Stream FIFO core uses one clock from the AXI4-Lite interface for all clock inputs. When the AXI Ethernet core is used with the AXI4-Stream FIFO core, all the AXI4-Stream input clocks of the AXI Ethernet core must use the same clock.

Table 2-3: **I/O Signals**

Signal Name	I/O	Width	Default	Description
Top-Level System Signal				
Interrupt	O	1	0	Indicates Interrupt status. This signal is asserted if any of the interrupt status bit is set to 1 and interrupt is enabled for that particular bit. By default, the signal is set to 0.
Global Signals				
s_axi_aclk	I	1	0	Global Interface Clock: All signals on Interface must be synchronous to ACLK.
s_axi_aresetn	I	1	0	Global reset: This signal is active-Low, ARESETN must be asserted at least for one clock cycle.

Table 2-3: **I/O Signals *(Cont'd)***

Signal Name	I/O	Width	Default	Description
AXI4-Lite Write Address Channel Signals				
s_axi_awaddr	I	C_S_AXI_ ADDR_WIDTH	0	Write Address: The write address bus gives the address of the first transfer in a write burst transaction. The associated control signals are used to determine the addresses of the remaining transfers in the burst.
s_axi_awvalid	I	1	0	Write Address Valid: Indicates that valid write address and control information are available: • 1 = Address and control information available. • 0 = Address and control information not available. The address and control information remain stable until the address acknowledge signal, AWREADY, goes High.
s_axi_awready	O	1	0	Write Address Ready: Indicates that the slave is ready to accept an address and associated control signals: • 1 = Slave ready • 0 = Slave not ready
AXI4-Lite Write Data Channel Signals				
s_axi_wdata	I	C_S_AXI_DATA_ WIDTH	0	Write Data: The write data bus width is 32-bit only.
s_axi_wstrb	I	C_S_AXI_DATA_ WIDTH / 8	0	Write Strobes: Indicates which byte lanes to update in memory. There is one write strobe for each eight bits of the write data bus. Therefore, WSTRB[n] corresponds to WDATA[(8 × n) + 7:(8 × n)]. For example: • S_AXI_WSTRB[0] = 1, WDATA[7:0] is valid. • S_AXI_WSTRB[3] = 0b, WDATA[31:24] is not valid.
s_axi_wvalid	I	1	0	Write Valid: Indicates that valid write data and strobes are available: • 1 = Write data and strobes available • 0 = Write data and strobes not available

Table 2-3: **I/O Signals *(Cont'd)***

Signal Name	I/O	Width	Default	Description
s_axi_wready	O	1	0	Write Ready: Indicates that the slave can accept the write data: • 1 = Slave ready • 0 = Slave not ready
AXI4-Lite Write Response Channel Signals				
s_axi_bresp	O	2	0	Write Response: Indicates the status of the write transaction. The allowable responses are OKAY, EXOKAY, SLVERR, and DECERR.
s_axi_bvalid	O	1	0	Write Response Valid: Indicates that a valid write response is available: • 1 = Write response available • 0 = Write response not available
s_axi_bready	I	1	1	Response Ready: Indicates that the master can accept the response information. • 1 = Master ready • 0 = Master not ready
AXI4-Lite Read Address Channel Signals				
s_axi_araddr	I	C_S_AXI_ADDR_WIDTH	0	Read Address: The read address bus gives the initial address of a read burst transaction. Only the start address of the burst is provided and the control signals that are issued alongside the address detail how the address is calculated for the remaining transfers in the burst.
s_axi_arvalid	I	1	0	Read Address Valid: When High, indicates that the read address and control information is valid and will remain stable until the address acknowledge signal, ARREADY, is High. • 1 = Address and control information valid • 0 = Address and control information not valid
s_axi_arready	O	1	0	Read Address Ready: Indicates that the slave is ready to accept an address and associated control signals: • 1 = Slave ready • 0 = Slave not ready

Table 2-3: **I/O Signals *(Cont'd)***

Signal Name	I/O	Width	Default	Description
AXI4-Lite Read Data Channel Signals				
s_axi_rdata	O	C_S_AXI_DATA_WIDTH	0	Read Data: The read data bus width is 32-bit only.
s_axi_rresp	O	2	0	Read Response: Indicates the status of the read transfer. The allowable responses are OKAY, EXOKAY, SLVERR, and DECERR.
s_axi_rvalid	O	1	0	Read Valid: Indicates that the required read data is available and the read transfer can complete: • 1 = Read data available • 0 = Read data not available
s_axi_rready	I	1	0	Read Ready: Indicates that the master can accept the read data and response information: • 1 = Master ready • 0 = Master not ready
AXI4-Stream Transmit Data Channel Signals				
axi_str_txd_aclk[(1)]	I	1	0	ACLK: Clock for the AXI4-Stream Transmit data interface. Presently not used in the core.
mm2s_prmry_reset_out_n	O	1	0	Reset: Reset for the AXI4-Stream Transmit data interface.
axi_str_txd_tvalid	O	1	0	TVALID: Indicates that the master is driving a valid transfer. A transfer takes place when both TVALID and TREADY are asserted.
axi_str_txd_tready	I	1	0	TREADY: Indicates that the slave can accept a transfer in the current cycle.
axi_str_txd_tdata	O	C_S_AXI_DATA_WIDTH Or C_S_AXI4_DATA_WIDTH	0	TDATA: The primary payload that is used to provide the data that is passing across the interface. The width of the data payload is an integer number of bytes. Supported TDATA widths include: 32, 64, 128, 256 or 512 (AXI4 interface only).
axi_str_txd_tkeep	O	C_S_AXI_DATA_WIDTH/8 Or C_S_AXI4_DATA_WIDTH/8	0	TKEEP: The byte qualifier that indicates whether the content of the associated byte of TDATA is valid. For a 32-bit DATA, bit 0 corresponds to the least significant byte on DATA, and bit 3 corresponds to the most significant byte.
axi_str_txd_tlast	O	1	0	TLAST: Indicates the boundary of a packet.

Table 2-3: **I/O Signals *(Cont'd)***

Signal Name	I/O	Width	Default	Description
axi_str_txd_tdest	O	C_AXIS_TDEST_ WIDTH	0	TDEST: Destination AXI4-Stream Identifier and Provides routing information for the data stream.
axi_str_txd_tstrb[(1)]	O	C_S_AXI_DATA_ WIDTH/8 Or C_S_AXI4_DATA _WIDTH/8	0	TSTRB: The byte qualifier that indicates whether the content of the associated byte of TDATA is processed as a data byte or a position byte.
axi_str_txd_tid[(1)]	O	C_AXIS_TID_ WIDTH	0	TID: The data stream identifier that indicates different streams of data.
axi_str_txd_tuser[(1)]	O	C_AXIS_TUSER_ WIDTH	0	TUSER: User-defined sideband information that can be transmitted with the data stream.
AXI4-Stream Transmit Control Channel Signals				
axi_str_txc_aclk [(1)]	I	1	0	ACLK: Clock for the AXI4-Stream Transmit data interface. Presently not used in the core.
mm2s_cntrl_reset_out_n	O	1	0	Reset: Reset for the AXI4-Stream Transmit data interface.
axi_str_txc_tvalid	O	1	0	TVALID: Indicates that the master is driving a valid transfer. A transfer takes place when both TVALID and TREADY are asserted
axi_str_txc_tready	I	1	0	TREADY: Indicates that the slave can accept a transfer in the current cycle
axi_str_txc_tdata	O	C_S_AXI_DATA_ WIDTH Or C_S_AXI4_ DATA_WIDTH	0	TDATA: The primary payload that is used to provide the data that is passing across the interface. The width of the data payload is an integer number of bytes. Supported TDATA widths include 32, 64, 128, 256, or 512 (AXI4 Interface only).
axi_str_txc_tkeep	O	C_S_AXI_DATA_ WIDTH/8 Or C_S_AXI4_ DATA_WIDTH/8	0	TKEEP: The byte qualifier that indicates whether the content of the associated byte of TDATA is valid. For a 32-bit DATA, bit 0 corresponds to the least significant byte on DATA, and bit 3 corresponds to the most significant byte.
axi_str_txc_tlast	O	1	0	TLAST: Indicates the boundary of a packet.

Table 2-3: **I/O Signals *(Cont'd)***

Signal Name	I/O	Width	Default	Description
axi_str_txc_tstrb[(1)]	O	C_S_AXI_DATA_WIDTH/8 Or C_S_AXI4_DATA_WIDTH/8	0	TSTRB: The byte qualifier that indicates whether the content of the associated byte of TDATA is processed as a data byte or a position byte.
axi_str_txc_tid[(1)]	O	C_AXIS_TID_WIDTH	0	TID: The data stream identifier that indicates different streams of data.
axi_str_txc_tdest	O	C_AXIS_TDEST_WIDTH	0	TDEST: Provides routing information for the data stream.
axi_str_txc_tuser[(1)]	O	C_AXIS_TUSER_WIDTH	0	TUSER: User-defined sideband information that can be transmitted with the data stream
AXI4-Stream Receive Data Channel Signals				
axi_str_rxd_aclk [(1)]	I	1	0	ACLK: Clock for the AXI4-Stream Transmit data interface. Presently not used in the core.
s2mm_prmry_reset_out_n	O	1	0	Reset: Reset for the AXI4-Stream Transmit data interface.
axi_str_rxd_tvalid	I	1	0	TVALID: Indicates that the master is driving a valid transfer. A transfer takes place when both TVALID and TREADY are asserted.
axi_str_rxd_tready	O	1	0	TREADY: Indicates that the slave can accept a transfer in the current cycle.
axi_str_rxd_tdata	I	C_S_AXI_DATA_WIDTH Or C_S_AXI4_DATA_WIDTH	0	TDATA: The primary payload that is used to provide the data that is passing across the interface. The width of the data payload is an integer number of bytes. Supported TDATA widths include 32, 64, 128, 256, or 512 (AXI4 interface only).
axi_str_rxd_tkeep	I	C_S_AXI_DATA_WIDTH/8 Or C_S_AXI4_DATA_WIDTH/8	0	TKEEP: The byte qualifier that indicates whether the content of the associated byte of TDATA is valid. For a 32-bit DATA, bit 0 corresponds to the least significant byte on DATA, and bit 3 corresponds to the most significant byte.
axi_str_rxd_tlast	I	1	0	TLAST: Indicates the boundary of a packet.
axi_str_rxd_tdest	I	C_AXIS_TDEST_WIDTH	0	TDEST: Destination AXI4-Stream Identifier and provides routing information for the data stream.

Table 2-3: **I/O Signals *(Cont'd)***

Signal Name	I/O	Width	Default	Description
axi_str_rxd_tstrb[1]	O	C_S_AXI_DATA_WIDTH/8 Or C_S_AXI4_DATA_WIDTH/8	0	TSTRB: The byte qualifier that indicates whether the content of the associated byte of TDATA is processed as a data byte or a position byte.
axi_str_rxd_tid[1]	O	C_AXIS_TID_WIDTH	0	TID: The data stream identifier that indicates different streams of data.
axi_str_rxd_tuser[1]	O	C_AXIS_TUSER_WIDTH	0	TUSER: User-defined sideband information that can be transmitted with the data stream.
AXI4 Write Address Channel Signals				
s_axi4_awid	I	C_S_AXI_ID_WIDTH	0	Write Address ID: Identification tag for the write address group of signals.
s_axi4_awaddr	I	C_S_AXI_ADDR_WIDTH	0	Write Address: The write address bus gives the address of the first transfer in a write burst transaction. The associated control signals are used to determine the addresses of the remaining transfers in the burst.
s_axi4_awlen	I	8	0	Burst Length: The burst length gives the exact number of transfers in a burst. This information determines the number of data transfers associated with the address.
s_axi4_awsize	I	3	0	Burst Size: Indicates the size of each transfer in the burst.
s_axi4_awburst	I	2	0	Burst Type: The burst type, coupled with the size information, details how the address for each transfer within the burst is calculated. This core supports incremental burst type only. Core behavior is not guaranteed if unsupported burst type is set.
s_axi4_awlock	I	1	0	Lock Type: This signal provides additional information about the atomic characteristics of the transfer. Presently this signal is not used in the core.
s_axi4_awcache	I	4	0	Cache Type: Indicates the bufferable, cacheable, writethrough, write-back, and allocate attributes of the transaction. Presently this signal is not used in the core.

Table 2-3: **I/O Signals *(Cont'd)***

Signal Name	I/O	Width	Default	Description
s_axi4_awprot	I	3	0	Protection Type: Indicates the normal, privileged, or secure protection level of the transaction and whether the transaction is a data access or an instruction access. Presently this signal is not used in the core.
s_axi4_awvalid	I	1	0	Write Address Valid: Indicates that valid write address and control information are available: • 1 = Address and control information available • 0 = Address and control information not available The address and control information remain stable until the address acknowledge signal, AWREADY, goes High.
s_axi4_awready	O	1	0	Write Address Ready: Indicates that the slave is ready to accept an address and associated control signals: • 1 = Slave ready • 0 = Slave not ready
AXI4 Write Data Channel Signals				
s_axi4_wdata	I	C_S_AXI4_DAT_WIDTH	0	Write Data: The write data bus can be 32 or 64 bits wide.
s_axi4_wstrb	I	C_S_AXI4_DATA_WIDTH/8	0	Write Strobes: Indicates which byte lanes to update in memory. There is one write strobe for each eight bits of the write data bus. Therefore, WSTRB[n] corresponds to WDATA[(8 × n) + 7:(8 × n)]. For a 64-bit DATA, bit 0 corresponds to the least significant byte on DATA, and bit 7 corresponds to the most significant byte. For example: • STROBE[0] = 1b, DATA[7:0] is valid • STROBE[7] = 0b, DATA[63:56] is not valid
s_axi4_wlast	I	1	0	Write Last: Indicates the last transfer in a write burst.
s_axi4_wvalid	I	1	0	Write Valid: Indicates that valid write data and strobes are available: • 1 = Write data and strobes available • 0 = Write data and strobes not available

Table 2-3: **I/O Signals *(Cont'd)***

Signal Name	I/O	Width	Default	Description
s_axi4_wready	O	1	0	Write Ready: Indicates that the slave can accept the write data: • 1 = Slave ready • 0 = Slave not ready
AXI4 Write Response Channel Signals				
s_axi4_bid	O	C_S_AXI_ID_ WIDTH	0	Response ID: The identification tag of the write response. The BID value must match the AWID value of the write transaction to which the slave is responding.
s_axi4_bresp	O	2	0	Write Response: Indicates the status of the write transaction. The allowable responses are OKAY, EXOKAY, SLVERR, and DECERR.
s_axi4_bvalid	O	1	0	Write Response Valid: Indicates that a valid write response is available: • 1 = Write response available. • 0 = Write response not available.
s_axi4_bready	I	1	1	Response Ready: Indicates that the master can accept the response information. • 1 = Master ready. • 0 = Master not ready.
AXI4 Read Address Channel Signals				
s_axi4_arid	I	C_S_AXI_ID_ WIDTH	0	Read Address ID: This signal is the identification tag for the read address group of signals. ARID is always set to zero; all configured channels access to a single address MAP region in Memory mapped interconnect.
s_axi4_araddr	I	C_S_AXI_ADDR_ WIDTH	0	Read Address: The read address bus gives the initial address of a read burst transaction. Only the start address of the burst is provided and the control signals that are issued alongside the address detail how the address is calculated for the remaining transfers in the burst.
s_axi4_arlen	I	8	0	Burst Length: The burst length gives the exact number of transfers in a burst. This information determines the number of data transfers associated with the address.

Table 2-3: **I/O Signals *(Cont'd)***

Signal Name	I/O	Width	Default	Description
s_axi4_arsize	I	3	0	Burst Size: This signal indicates the size of each transfer in the burst. Burst Size is always set based on configured data width of the interface.
s_axi4_arburst	I	2	0	Burst Type: The burst type, coupled with the size information, details how the address for each transfer within the burst is calculated. Burst Type is always set to Incremental. Core behavior is not guaranteed if unsupported burst type is set.
s_axi4_arlock	I	1	0	Lock Type: This signal provides additional information about the atomic characteristics of the transfer. Presently this signal is not used in the core.
s_axi4_arcache	I	4	0	Cache Type: This signal provides additional information about the cacheable characteristics of the transfer. Presently this signal is not used in the core.
s_axi4_arprot	I	3	0	Protection Type: This signal provides protection unit information for the transaction. Presently this signal is not used in the core.
s_axi4_arvalid	I	1	0	Read Address Valid: When High, indicates that the read address and control information is valid and will remain stable until the address acknowledge signal, ARREADY, is High. • 1 = Address and control information valid • 0 = Address and control information not valid
s_axi4_arready	O	1	0	Read Address Ready: Indicates that the slave is ready to accept an address and associated control signals: • 1 = Slave ready • 0 = Slave not ready
AXI4 Read Data Channel Signals				
s_axi4_rid	O	C_S_AXI_ID_ WIDTH	0	Read ID Tag: ID tag of the read data group of signals. The RID value is generated by the slave and must match the ARID value of the read transaction to which it is responding.

Table 2-3: **I/O Signals *(Cont'd)***

Signal Name	I/O	Width	Default	Description
s_axi4_rdata	O	C_S_AXI4_DATA_WIDTH	0	Read Data: The read data bus can be 32 or 64 bits wide.
s_axi4_rresp	O	2	0	Read Response: Indicates the status of the read transfer. The allowable responses are OKAY, EXOKAY, SLVERR, and DECERR.
s_axi4_rlast	O	1	0	Read Last: Indicates the last transfer in a read burst.
s_axi4_rvalid	O	1	0	Read Valid: Indicates that the required read data is available and the read transfer can complete: • 1 = Read data available • 0 = Read data not available
s_axi4_rready	I	1	0	Read Ready: Indicates that the master can accept the read data and response information: • 1 = Master ready • 0 = Master not ready

Notes:
1. This port is currently not used.

Register Space

The AXI4-Stream FIFO core contains the registers listed in Table 2-4.

Table 2-4: **Register Names and Descriptions**

Register Name	AXI Address	Access
Interrupt Status Register (ISR)	C_BASEADDR + 0x0	Read/Clear on Write(1)
Interrupt Enable Register (IER))	C_BASEADDR + 0x4	Read/Write
Transmit Data FIFO Reset (TDFR)	C_BASEADDR + 0x8	Write(2)
Transmit Data FIFO Vacancy (TDFV)	C_BASEADDR + 0xC	Read
Transmit Data FIFO 32-bit Wide Data Write Port (TDFD)	C_BASEADDR + 0x10 or C_AXI4_BASEADDR + 0x0000	Write(3)
Transmit Length Register (TLR)	C_BASEADDR + 0x14	Write
Receive Data FIFO reset (RDFR)	C_BASEADDR + 0x18	Write(2)
Receive Data FIFO Occupancy (RDFO)	C_BASEADDR + 0x1C	Read

Table 2-4: **Register Names and Descriptions *(Cont'd)***

Register Name	AXI Address	Access
Receive Data FIFO 32-bit Wide Data Read Port (RDFD)	C_BASEADDR + 0x20 or C_AXI4_BASEADDR + 0x1000	Read(3)
Receive Length Register (RLR)	C_BASEADDR + 0x24	Read
AXI4-Stream Reset (SRR)	C_BASEADDR + 0x28	Write(2)
Transmit Destination Register (TDR)	C_BASEADDR + 0x2C	Write
Receive Destination Register (RDR)	C_BASEADDR + 0x30	Read
Transmit ID Register(4)	C_BASEADDR + x34	Write
Transmit USER Register(4)	C_BASEADDR + x38	Write
Receive ID Register(4)	C_BASEADDR + x3C	Read
Receive USER Register(4)	C_BASEADDR + x40	Read
Reserved	C_BASEADDR + 0x44 to C_BASEADDR + 0x7C	N/A(5)

Notes:

1. The latched interruptible condition is cleared by writing a 1 to that bit location. Writing a 1 to a bit location that is 0 has no effect. Likewise, writing a 0 to a bit location that is 1 has no effect. Multiple bits can be cleared in a single write.
2. Reset if written with 0xA5.
3. C_AXI4_BASEADDR should be used only if the Data Interface option is AXI4.
4. Not currently supported.
5. If read, these registers will return 0x0. Writing these registers will have no effect.
6. C_BASEADDR is defined by the interconnect when using IP Integrator. When you implement a standalone core (e.g. selecting from the IP Catalog in Vivado), only the address signals **`s_axi_awaddr(5:2)`** and **`s_axi_araddr(5:2)`** are decoded.This results in repeating the address map of the registers every 64 hex locations. In this case, **`C_BASEADDR`** can be considered to be 0.

Interrupt Interface

The interrupt signals generated by the AXI4-Stream FIFO core are managed by the ISR and IER registers. The ISR is combined with the IER register to define the interrupt interface of the AXI4-Stream FIFO core. An overview diagram of the interrupt control structure is shown in Figure 2-2.

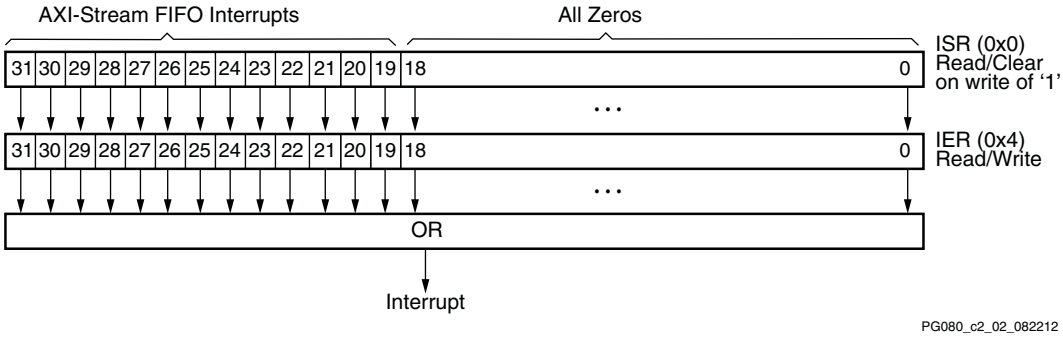

Figure 2-2: **Interrupt Control Structure**

Interrupt Status Register (ISR)

The Interrupt Status Register is shown in Figure 2-3. The Interrupt Status register uses one bit to represent each internal interruptible condition.

After an interruptible condition occurs, it is captured in this register (represented as the corresponding bit being set to 1) even if the condition changes. The latched interruptible condition is cleared by writing a 1 to its bit location. Writing a 1 to a bit location that is 0 has no effect. Likewise, writing a 0 to a bit location that is 1 has no effect. Multiple bits can be cleared in a single write.

For any bit set in the Interrupt Status Register, a corresponding bit must also be set in the Interrupt Enable Register for the Interrupt signal to be driven active High out of the AXI4-Stream FIFO core.

The Interrupt Status Register bit definitions are detailed in Table 2-5.

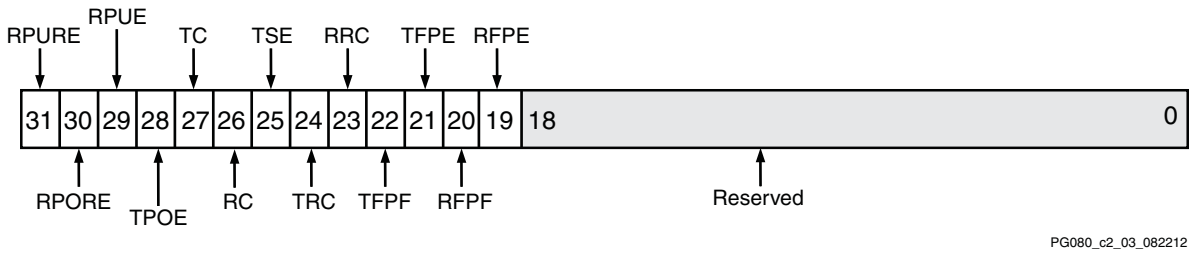

Figure 2-3: **Interrupt Status Register (Offset 0x0)**

Table 2-5: **Interrupt Status Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
18:0	Reserved	Read	0x0	**Reserved**: These bits are reserved for future definition and will always return all zeros.
19	RFPE	Read/Clear on Write of 1	0	**Receive FIFO Programmable Empty**: Generated when the difference between the read and write pointers of the receive FIFO reaches the programmable EMPTY threshold value when the FIFO is being emptied. 0 = No interrupt pending 1 = Interrupt pending
20	RFPF	Read/Clear on Write of 1	1	**Receive FIFO Programmable Full**: This interrupt is generated when the difference between the read and write pointers of the receive FIFO reaches the programmable FULL threshold value. 0 = No interrupt pending 1 = Interrupt pending
21	TFPE	Read/Clear on Write of 1	0	**Transmit FIFO Programmable Empty**: This interrupt is generated when the difference between the read and write pointers of the transmit FIFO reaches the programmable EMPTY threshold value when the FIFO is being emptied. 0 = No interrupt pending 1 = Interrupt pending For lower values of programmable threshold, this flag may toggle during the initial writes because of First Word Fall Through (FWFT) behavior of FIFO. This flag may toggle even though there are no external reads.
22	TFPF	Read/Clear on Write of 1	1	**Transmit FIFO Programmable Full**: This interrupt is generated when the difference between the read and write pointers of the transmit FIFO reaches the programmable FULL threshold value. 0 = No interrupt pending 1 = Interrupt pending
23	RRC	Read/Clear on Write of 1	1	**Receive Reset Complete**: This interrupt indicates that a reset of the receive logic has completed. 0 = No interrupt pending 1 = Interrupt pending
24	TRC	Read/Clear on Write of 1	1	**Transmit Reset Complete**: This interrupt indicates that a reset of the transmit logic has completed. 0 = No interrupt pending 1 = Interrupt pending

Table 2-5: **Interrupt Status Register Bit Definitions *(Cont'd)***

Bit(s)	Name	Core Access	Reset Value	Description
25	TSE	Read/Clear on Write of 1	0	**Transmit Size Error**: This interrupt is generated if the number of words (including partial words in the count) written to the transmit data FIFO does not match the value written to the transmit length register (bytes) divided by 4/8 and rounded up to the higher integer value for trailing byte fractions. Interrupts occur only for mismatch of word count (including partial words). Interrupts do not occur due to mismatch of byte count. 0 = No interrupt pending 1 = Interrupt pending
26	RC	Read/Clear on Write of 1	0	**Receive Complete**: Indicates that at least one successful receive has completed and that the receive packet data and packet data length is available. This signal is not set for unsuccessful receives. This interrupt can represent more than one packet received, so it is important to check the receive data FIFO occupancy value to determine if additional receive packets are ready to be processed. 0 = No interrupt pending 1 = Interrupt pending
27	TC	Read/Clear on Write of 1	0	**Transmit Complete**: Indicates that at least one transmit has completed. 0 = No interrupt pending 1 = Interrupt pending
28	TPOE	Read/Clear on Write of 1	0	**Transmit Packet Overrun Error**: This interrupt is generated if an attempt is made to write to the transmit data FIFO when it is full. A reset of the transmit logic is required to recover. 0 = No interrupt pending 1 = Interrupt pending
29	RPUE	Read/Clear on Write of 1	0	**Receive Packet Underrun Error**: This interrupt occurs when an attempt is made to read the receive FIFO when it is empty. The data read is not valid. A reset of the receive logic is required to recover. 0 = No interrupt pending 1 = Interrupt pending

Table 2-5: **Interrupt Status Register Bit Definitions *(Cont'd)***

Bit(s)	Name	Core Access	Reset Value	Description
30	RPORE	Read/Clear on Write of 1	0	**Receive Packet Overrun Read Error**: This interrupt occurs when more words are read from the receive data FIFO than are in the packet being processed. Even though the FIFO is not empty, the read has gone beyond the current packet and removed the data from the next packet. A reset of the receive logic is required to recover. 0 = No interrupt pending 1 = Interrupt pending
31	RPURE	Read/Clear on Write of 1	0	**Receive Packet Underrun Read Error**: This interrupt occurs when an attempt is made to read the receive length register when it is empty. The data read is not valid. A reset of the receive logic is required to recover. 0 = No interrupt pending 1 = Interrupt pending

Interrupt Enable Register (IER)

The Interrupt Enable Register shown in Figure 2-4 determines which interrupt sources in the Interrupt Status Register are allowed to generate interrupts. Setting to "1" in a bit location enables the related interrupt from being propagated, while a value of "0" disables it.

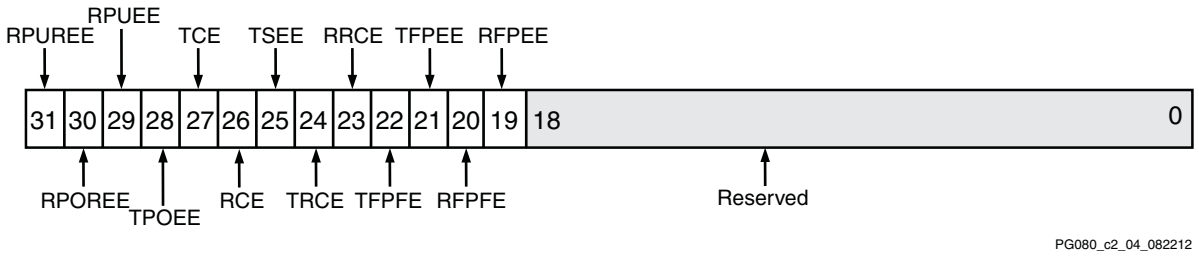

Figure 2-4: **Interrupt Enable Register (Offset 0x4)**

Transmit Data FIFO Reset Register (TDFR)

The Transmit Data FIFO Reset Register shown in Figure 2-5 is not an actual register, but is instead a write-only address, which when written with a specific value, generates a reset for the Transmit Data FIFO. This reset will not occur until transmit activity on the TX AXI4-Stream has completed. The reset can occur only during inactive times on the TX AXI4-Stream and will affect only the transmit circuitry in this core, thereby preventing the core on the other end of the AXI4-Stream from receiving a partial packet which could potentially cause a failure condition in the latter core. The reset is applied only during the inactive times on the TX AXI4-Stream. Writing a TDFR register with other than A5 value will disable the reset.

Because of this mode of operation, it is possible that if the AXI4-Stream becomes unresponsive during an AXI4-Stream transaction, a reset will never occur. For example, this might occur while waiting for the destination ready to go active in the middle of a transfer. In such cases it is necessary to use both the AXI4-Stream Reset and the S_AXI_ARESETN reset.

Note: To apply the AXI4-Stream Reset, you must write the SRR register and then assert S_AXI_ARESETN.

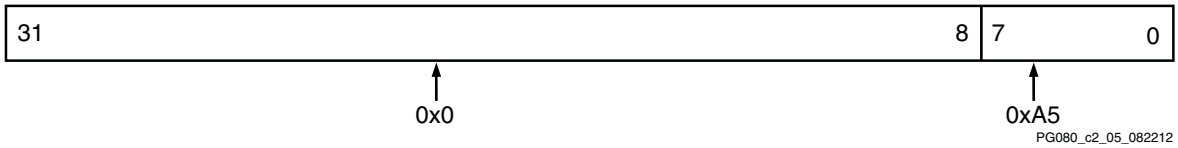

Figure 2-5: **Transmit Data FIFO Reset Register (Offset 0x8)**

Table 2-6: **Transmit Data FIFO Reset Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
31:0	Reset Key	Write	N/A	**Reset Write Value**. "0x000000A5" - Generate a reset. Others - No effect.

Transmit Data FIFO Vacancy Register (TDFV)

The Transmit Data FIFO Vacancy Register shown in Figure 2-6 is a read-only register that gives the vacancy status of the Transmit Data FIFO. This is an unsigned value and reflects the current snapshot of the number of locations free for data storage in the Transmit Data FIFO. The value reflected (N) in this register tells you that you can perform N writes to Transmit FIFO. The value of this register after reset is C_TX_FIFO_DEPTH-4. The register does not decrement for every Transmit Data FIFO Data Write Port (TDFD) write. It decrements by two for every two write locations.

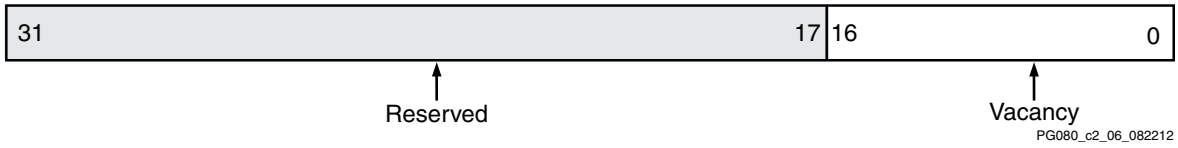

Figure 2-6: **Transmit Data FIFO Vacancy Register (Offset 0xC)**

Transmit Data FIFO Data Write Port (TDFD)

The Transmit Data FIFO Data Write Port shown in Figure 2-7 is an *N*-bit wide address location for writing data into the Transmit Data FIFO. *N* is equal to C_S_AXI_DATA_WIDTH.

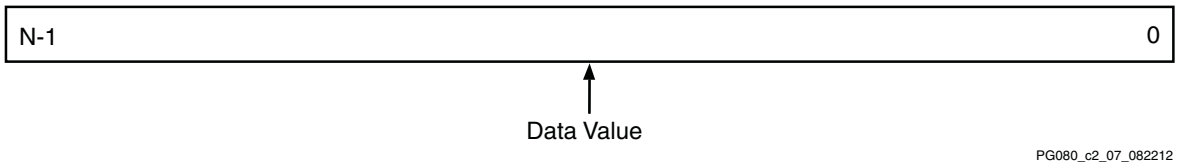

Figure 2-7: **Transmit Data FIFO Data Write Port (Offset 0x10)**

Table 2-7: **Transmit Data FIFO Data Write Port Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
(*N*-1) - 0	Write Data Value	Write	N/A	**Transmit Data FIFO Write Value**. *N* is equal to C_S_AXI_DATA_WIDTH.

Receive Data FIFO Reset Register (RDFR)

The Receive Data FIFO Reset Register shown in Figure 2-8 is not an actual register but, rather a write-only address, which when written with a specific value, generates a reset for the Receive Data FIFO.

This reset will not occur until receive activity on the RX AXI4-Stream has completed. Only during inactive times on the RX AXI4-Stream can a reset occur. It will affect only the receive circuitry in this core. This prevents the core on the other end of the AXI4-Stream from transmitting a partial packet which can cause failure condition in that core. The reset is applied only during the inactive times on the RX AXI4-Stream. Writing a RDFR register with other than A5 value will disable the reset.

Because of this mode of operation, it is possible that if the AXI4-Stream interface becomes unresponsive during an AXI4-Stream transaction, that the reset will never occur. For example, if a packet is received over the AXI4-Stream that exceeds the FIFO size of this core, the core destination is ready to become inactive in the middle of a transfer. In this case, an S_AXI_ARESETN reset is needed.

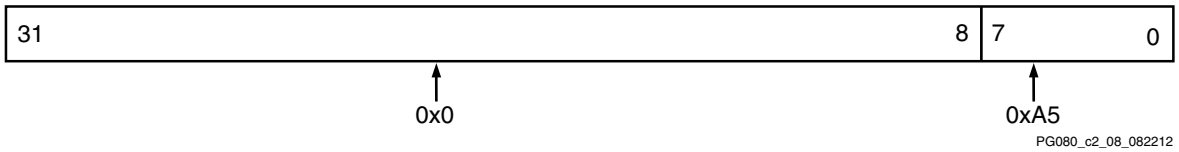

Figure 2-8: **Receive Data FIFO Reset Register (Offset 0x18)**

Table 2-8: **Receive Data FIFO Reset Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
31:0	Reset Key	Write	N/A	**Reset Write Value**: "0x000000A5" - Generate a reset. Others - No effect.

Receive Data FIFO Occupancy Register (RDFO)

The Receive Data FIFO Occupancy Register shown in Figure 2-9 is a read-only register that gives the occupancy status of the Receive Data FIFO.

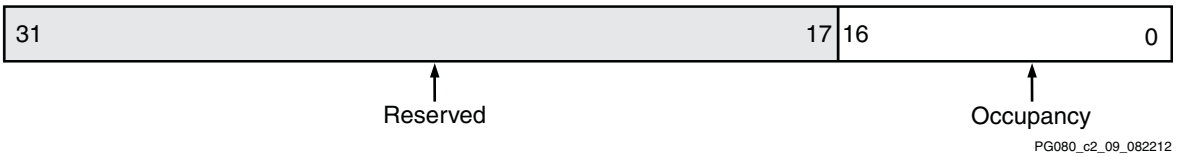

Figure 2-9: **Receive Data FIFO Occupancy Register (Offset 0x1C)**

Table 2-9: **Receive Data FIFO Occupancy Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
16:0	Occupancy	Read	0x0	**Receive Data FIFO Occupancy**: This is the unsigned value reflecting a current snapshot of the number of locations in use for data storage in the receive Data FIFO memory core in the most recent transaction. This value is only updated after a packet is successfully received, and therefore can be used to determine if a receive packet is ready to be processed when a non-0 value is read. If the number of packets received is one, then this register returns the value of the locations occupied. After the FIFO is read, any subsequent read to this register returns the value of 0. If more than one packet is received, a read to this register returns the number of locations occupied by the latest received packet.
31:17	Reserved	Read	0x0	**Reserved**: These bits are reserved for future definition and will always return all zeros.

Receive Data FIFO Data Read Port (RDFD)

The Receive Data FIFO Data Read Port shown in Figure 2-10 is a *N*-bit wide address location for reading data from the Receive Data FIFO. *N* is equal to C_S_AXI_DATA_WIDTH.

IMPORTANT: *The value of this register is not guaranteed if this register is read when Receive Packet Underrun Error (RPUE) interrupt bit in Interrupt Status Register (ISR) is set.*

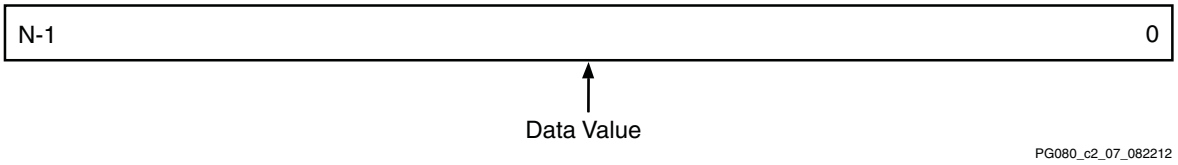

Figure 2-10: **Receive Data FIFO Data Read Port (Offset 0x20)**

Table 2-10: **Receive Data FIFO Data Read Port Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
(N-1) - 0	Read Data Value	Read	N/A	**Receive Data FIFO Read Value**. *N* is equal to C_S_AXI_DATA_WIDTH.

Transmit Length Register (TLR)

The Transmit Length Register shown in Figure 2-11 and Figure 2-12. This register is used to store packet length values (the number of bytes in the packet) corresponding to valid packets ready for transmit. The data for the packet is stored in the transmit Data FIFO. The data is written to the AXI4-Stream FIFO core over the AXI4 interface, typically by a processor or DMA core such the Central DMA (CDMA). When presenting a transmit packet to the AXI4-Stream FIFO core, write the packet data to the Transmit Data FIFO first, then write the length of the packet into the TLR.

It is not valid to write data for multiple packets to the transmit data FIFO before writing the packet length values.

Store-and-Forward Mode

In this mode, packet transmission on the TX AXI4-Stream interface does not start until the TLR is written with a valid packet length value. The width of the TLR is wide enough to support packets up to 8 MBytes-4 in length. The smallest packet that can be transmitted is 1 byte. The maximum packet that can be transmitted is limited by the size of the FIFO, which is (C_TX_FIFO_DEPTH–4)*(data interface width/8) bytes.

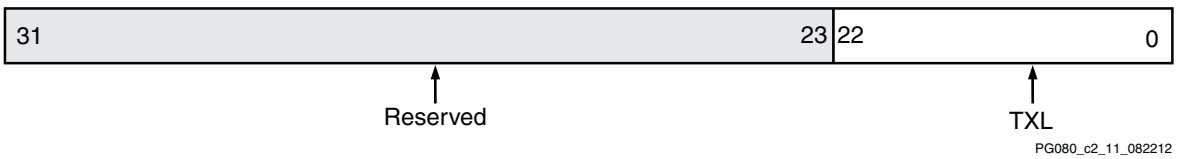

Figure 2-11: **Transmit Length Register: Store-and-Forward Mode (Offset 0x14)**

Table 2-11: **Transmit Length Register (Store-and-Forward Mode) Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
22:0	TXL	Write	0x0	**Transmit Length**: The number of bytes of the corresponding transmit packet stored in the transmit data FIFO.
31:23	Reserved	N/A	0x0	**Reserved**: These bits are reserved for future definition and will always return all zeros.

Cut-Through Mode

In this mode, packet transmission starts on the AXI4-Stream interface when the transmit FIFO is not empty. However, the last beat of the packet is transmitted only when the TLR is written with a valid packet length value.

The width of the TLR is wide enough to support packets up to 8 MBytes–4 (8388604) in length. The smallest packet that can be transmitted is 1 byte. The maximum packet that can be transmitted is 8 MBytes–4 and is independent of the TX FIFO depth selected.

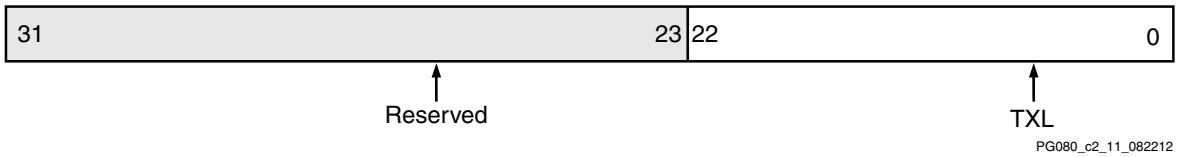

Figure 2-12: **Transmit Length Register: Cut-Through Mode (Offset 0x14)**

Table 2-12: **Transmit Length Register (Cut-Through Mode) Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
22:0	TXL	Write	0x0	**Transmit Length**: The number of bytes of the corresponding transmit packet stored in the transmit data FIFO.
31:23	Reserved	N/A	0x0	**Reserved**: These bits are reserved for future definition and will always return all zeros.

Receive Length Register (RLR)

The receive length register shown in Figure 2-13 and Figure 2-14. This register is used to retrieve packet length values (the number of bytes in the packet) corresponding to valid packets received. The data for the packet is stored in the Receive Data FIFO.

The RLR should only be read when a receive packet is available for processing (the receive occupancy is not zero). After the RLR is read, the receive packet data should be read from the receive data FIFO before the RLR is read again.

Note: RDFO should be read before reading RLR. Reading RLR first will result in the RDFO being reset to zero.

Store-and-Forward Mode

In this mode, the length is written by the AXI4-Stream FIFO core when the complete packet is received across the RX AXI4-Stream interface. The width of the RLR is wide enough to support packets up to 8 MBytes–4 (8388604) in length. The smallest packet that can be received is 1 byte. The maximum packet that can be received is limited by the size of the FIFO, which is (C_RX_FIFO_DEPTH–4)*(data interface width/8) bytes.

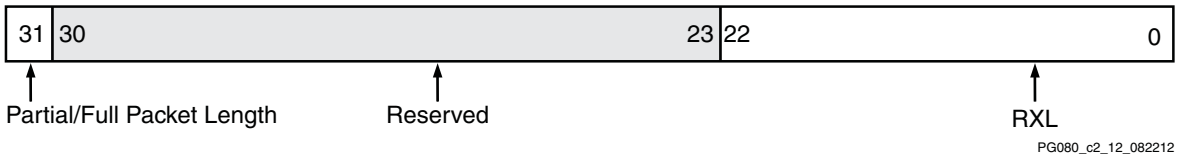

Figure 2-13: **Receive Length Register: Store-and-Forward Mode (Offset 0x24)**

Table 2-13: **Receive Length Register (Store-and-Forward Mode) Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
22:0	RXL	Read	0x0	**Receive Length**: The number of bytes of the corresponding receive data stored in the receive data FIFO.
31:23	Reserved	Read	0x0	**Reserved**: These bits are reserved for future definition and will always return all zeros.

Cut-Through Mode

In this mode, the length is written by the AXI4-Stream FIFO core when the RX FIFO is not empty. Bit 31 of the register is used to indicate whether the length value given in the remaining 30 bits is for a partial or full packet. When bit 31 is 1, the length indicates the amount of partial packet data that can be read. After the last beat of the packet received on the AXI4-Stream side, bit 31 becomes 0 and remaining bits show the complete packet length.

The width of the RLR is wide enough to support packets up to 8 MBytes–4 (8388604) in length. The smallest packet that can be received is 1 byte. The maximum packet that can be received is 8 MBytes–4 and is independent of the RX FIFO depth selected in Vivado IDE.

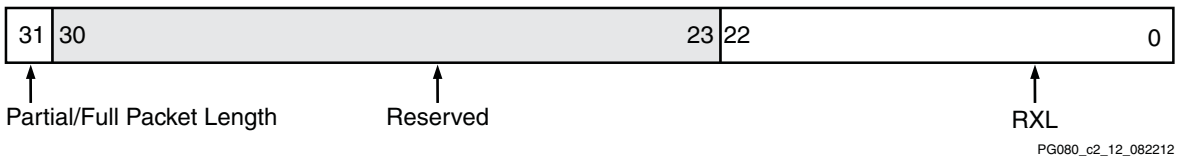

Figure 2-14: **Receive Length Register: Cut-Through Mode (Offset 0x24)**

Table 2-14: **Receive Length Register (Cut-Through Mode) Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
22:0	RXL	Read	0x0	**Receive Length**: The number of bytes of the corresponding receive data stored in the receive data FIFO.
30:23	Reserved	Read	0x0	**Reserved**: These bits are reserved for future definition and will always return all zeros.
31	Partial/Full Packet Length	Read	0x0	**Partial/Full Packet Length Indicator**: When bit 31 is 1, the RXL indicates the amount of partial packet data that can be read. After the last beat of the packet received on the AXI4-Stream side, bit 31 becomes 0 and the RXL shows the complete packet length.

AXI4-Stream Reset Register (SRR)

The AXI4-Stream Register shown in Figure 2-15 is not an actual register. It is a write-only address, which when written with a specific value, generates an immediate reset for the entire core as well as driving a reset on the external outputs, s2mm_prmry_reset_out_n, mm2s_prmry_reset_out_n, and mm2s_cntrl_reset_out_n, which can be used to reset the core on the other end of the AXI4-Stream.

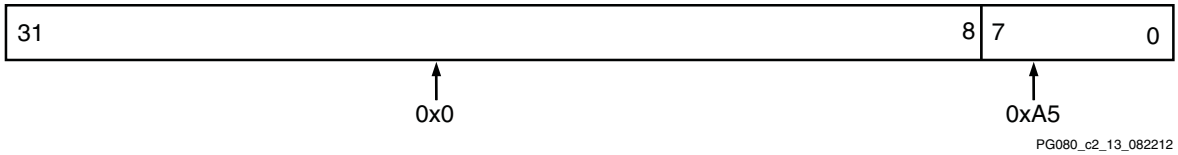

Figure 2-15: **AXI4-Stream Reset Register (Offset 0x28)**

Table 2-15: **AXI4-Stream Reset Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
31:0	Reset Key	Write	N/A	**Reset Write Value**: "0x000000A5" - Generate a reset. Others - No effect.

Transmit Destination Register (TDR)

The Transmit Destination Register shown in Figure 2-16 stores the destination address corresponding to the packet to be transmitted. When presenting a transmit packet to the AXI4-Stream FIFO core, write the destination address into TDR first, write the packet data to the Transmit Data FIFO next, and then write the length of the packet into the TLR.

The destination address must be written to the TDR before the packet data is written to the transmit data FIFO. Writing data for multiple packets to the transmit data FIFO before writing the destination address values is not a valid sequence.

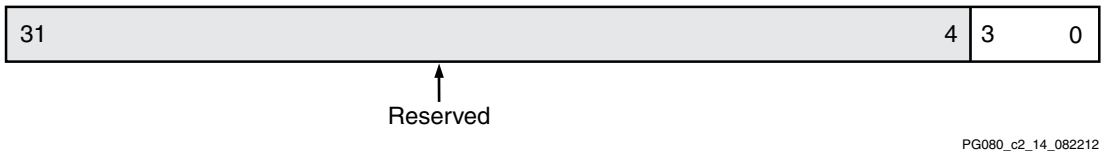

Figure 2-16: **Transmit Destination Register (Offset 0x2C)**

Table 2-16: **Transmit Destination Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
3:0	TDEST	Write	0x0	**Transmit Destination**: The destination address of the transmit packet stored in the transmit data FIFO. ***Note:*** TDEST is optional when generating the core and that TDEST can be specified to be 1 to 4 bits when it is implemented (see TDEST and corresponding "Width" in Figure 4-1, page 44).
31:4	Reserved	N/A	0x0	**Reserved**: These bits are reserved for future definition and will always return all zeros.

Receive Destination Register (RDR)

The Receive Destination Register shown in Figure 2-17 retrieves the destination address corresponding to the valid packet received.

The RDR should only be read when a receive packet is available for processing (the receive occupancy is not zero). After the RDR is read, the receive packet data should be read from the receive data FIFO before the RDR is read again. The RDR values are stored in the receive data FIFO by the AXI4-Stream FIFO core with the data of each packet. The RDR value for the subsequent packet to be processed is moved to the RDR when the previous RDR value has been read.

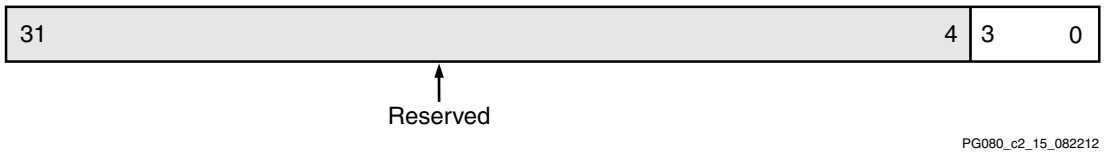

Figure 2-17: **Receive Destination Register (Offset 0x30)**

Table 2-17: **Receive Destination Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
3:0	RDEST	Read	0x0	**Receive Destination**: The destination address of the receive packet stored in the receive data FIFO. ***Note:*** TDEST is optional when generating the core and that TDEST can be specified to be 1 to 4 bits when it is implemented (see TDEST and corresponding "Width" in Figure 4-1, page 44).
31:4	Reserved	N/A	0x0	**Reserved**: These bits are reserved for future definition and will always return all zeros.

Reserved Registers

Reading from reserved registers returns zeros and writing to reserved registers will have no effect. However, any accesses to address offset 0x40 and above causes undefined results.

Designing with the Core

This chapter includes guidelines and additional information to facilitate designing with the core.

General Design Guidelines

The AXI4-Stream FIFO core can be used in applications to interface between an AXI4 memory mapped interface and an AXI4-Stream interface. An example of this application would be the Xilinx® AXI Ethernet IP core which has an AXI4-Lite interface for configuration and control and an AXI4-Stream interface for data transfer. The AXI4-Stream FIFO can be used as a bridge to interface to the AXI4 or AXI4-Lite interfaces as shown in Figure 3-1.

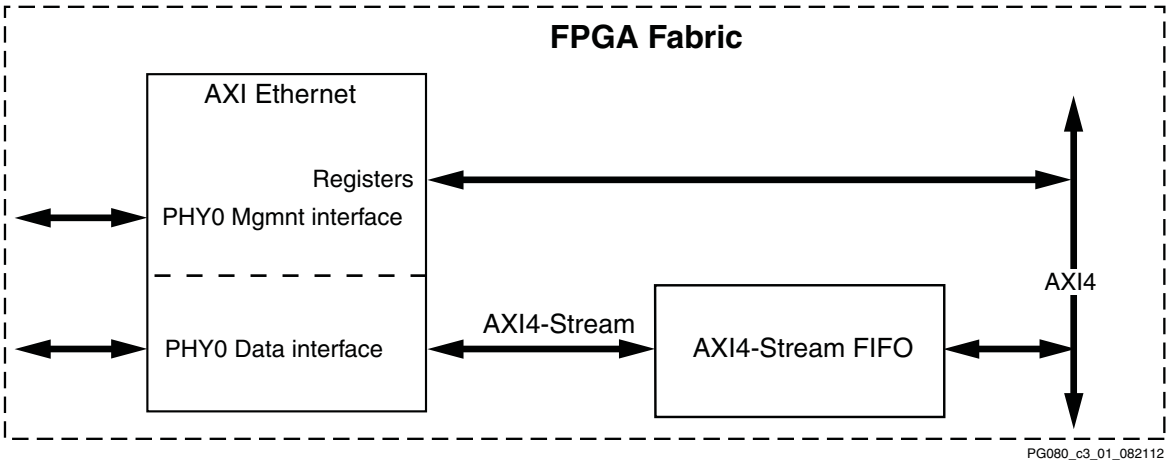

Figure 3-1: **AXI4-Stream FIFO Connected to an AXI Ethernet Core**

Design Tools

The AXI4-Stream FIFO core design is implemented using VHDL code.The Vivado® Design Suite includes a synthesis tool for synthesizing the core.

Target Technology

The target technology is an FPGA listed in the supported device family field of the LogiCORE™ IP Facts Table.

Clocking

The AXI4-Stream FIFO core operates on a single clock (**s_axi_aclk**), and all input and output interface signals of the AXI4-Stream and AXI4-Lite/AXI4 interfaces are synchronized with this clock.

Resets

The AXI4-Stream FIFO core uses a single asynchronous reset (**s_axi_aresetn**). The core stays in a reset state for three clock cycles after the reset applied.

Protocol Description

The AXI4-Stream FIFO core uses the industry standard AMBA® AXI4-Stream and AXI4 Protocol Specification. Figure 3-2 details the AXI4-Stream interface where INFORMATION represents all AXI4-Stream signals except TVALID/TREADY.

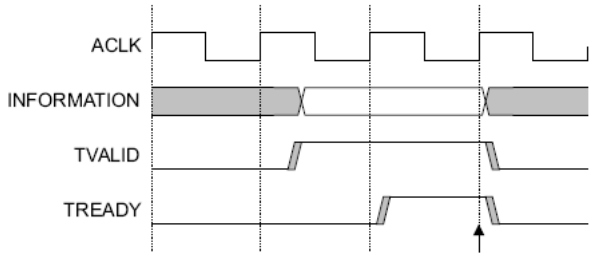

Figure 3-2: **AXI4-Stream Interface Timing Diagram**

Figure 3-3 details the AXI4 Write burst transaction, and Figure 3-4 details the AXI4 Read burst transaction.

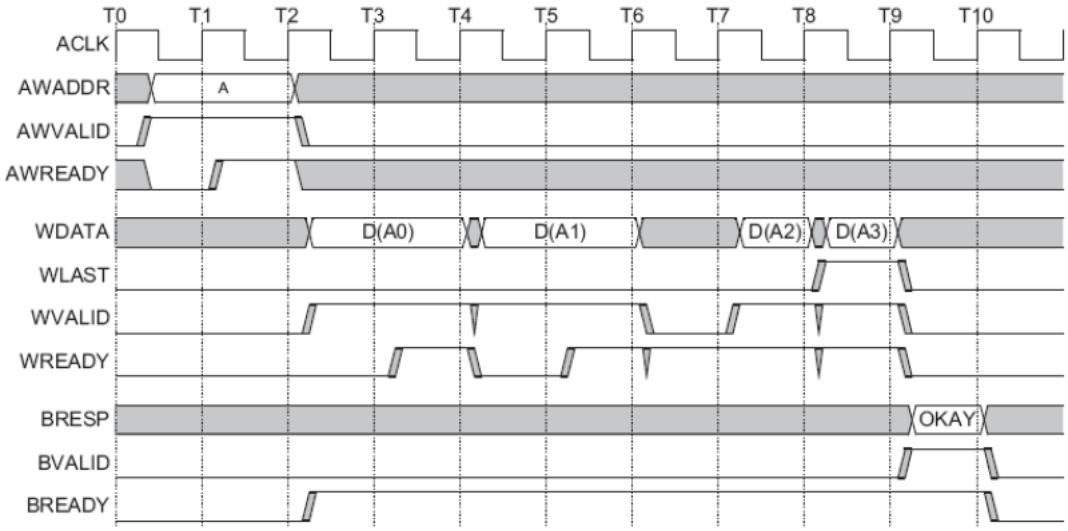

Figure 3-3: **AXI4 Write Burst Transaction**

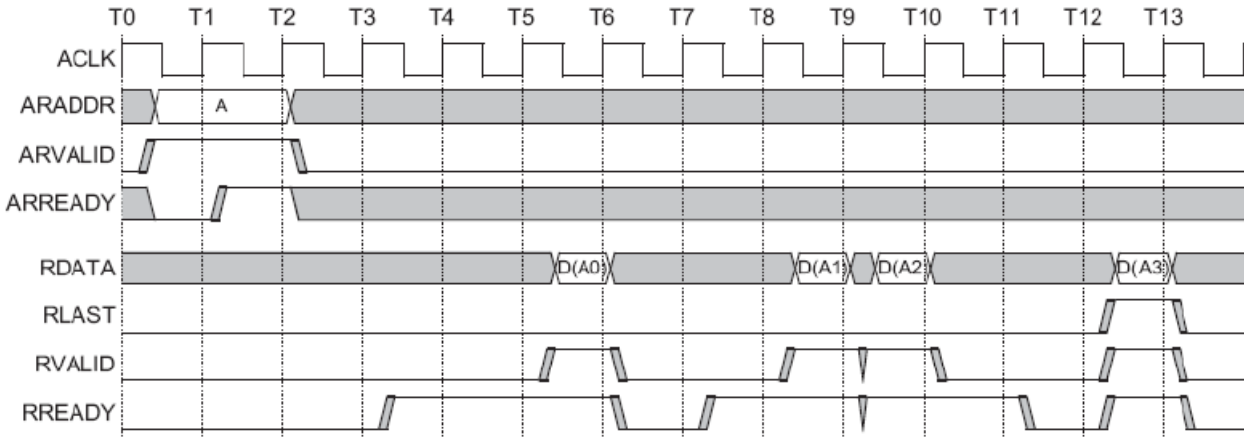

Figure 3-4: **AXI4 Read Burst Transaction**

This section describes the operation of the AXI4-Stream FIFO core through register accesses using the AXI Ethernet core as an example.

Depending on the Data Interface Option, either AXI4 or AXI4-Lite is used for FIFO accesses. When AXI4-Lite is selected, register access and FIFO accesses are handled by the AXI4-Lite interface. When AXI4 is selected, register access is handled by AXI4-Lite interface and FIFO accesses are handled by AXI4 interface. Data bursting is possible when the Data Interface option is AXI4.

The AXI4-Stream FIFO supports two packet transmission modes: store-and-forward mode and cut-through mode.

- In store-and-forward mode, packet transmission begins on the AXI4-Stream interface in the following circumstances:

- when the complete packet is written to the FIFO
- Length of packet is written to TX Length Register.

In this mode, the size of the FIFO must be large enough to hold the complete packet.

- In cut-through mode, packet transmission begins on the AXI4-Stream interface when there is enough data in the FIFO. In this mode, the FIFO does not need to hold the complete packet. However, ensure that the AXI4-Stream interface does not under run.

The AXI4-Stream FIFO supports two packet receiving modes: store-and-forward mode and cut-through mode.

- In store-and-forward mode, the data from the AXI4-Stream interface is completely stored in the FIFO prior to making it available over the AXI4 memory mapped interface. The RX Length Register (RLR) is updated with the length of packet, and the interrupt status registers are updated.
- In cut-through mode, packet reception begins on the AXI4 memory mapped interface when there is enough data in the FIFO. In this mode, the FIFO does not need to hold the complete packet. The RX Length Register (RLR) register is updated with the data count continuously until the packet is completely received. The RLR register value can be used to read the data out of the FIFO. The cut-through mode supports reception of packets that are larger than the FIFO size. However, ensure that AXI4-Stream interface does not over-run the FIFO.

Note: The AXI4-Stream signal **`tlast`** along with **`tvalid`** denotes the end of packet in store-and-forward mode and cut-through mode.

Note: The IP is designed to hold a maximum of (RX FIFO depth / 4) + 2 one word packets. For example, if the RX FIFO depth is 512, then the IP can hold a maximum of 130 one word packets.

Programing Sequence

Programming Sequence Using Direct Register Read/Write

This programming sequence is provided for you to directly perform the read/write operations to the registers and this might not match with the example driver software that is provided along with this IP.

Table 3-1 and Table 3-2 illustrate a power-up read of the registers followed by a programming sequence for transmission and reception of a single packet in store-and-forward mode and cut-through mode using AXI4-Lite interface. See the register definitions for further information and options.

Table 3-1: **Programming Sequence for TX and RX in Store-and-Forward Mode**

Register	Access	Value	Activity
Power-up/Reset Read of Register Values			
ISR	Read Word	01D00000	Read interrupt status register (indicates transmit reset complete and receive reset complete)
ISR	Write Word	0xFFFFFFFF	Write to clear reset done interrupt bits
ISR	Read Word	0x00000000	Read interrupt status register
IER	Read Word	0x00000000	Read interrupt enable register
TDFV	Read Word	0x000001FC	Read the transmit FIFO vacancy (for TX FIFO Depth of 512)
RDFO	Read Word	0x00000000	Read the receive FIFO occupancy
Transmit a Packet			
IER	Write Word	0x0C000000	Enable transmit complete and receive complete interrupts
TDR	Write Word	0x00000002	Transmit Destination address (0x2 = destination device address is 2)
TDFD	Write Word	0xFFFFFFFF	4 bytes of data
TDFD	Write Word	0x12345678	4 bytes of data
TDFD	Write Word	0x00010203	4 bytes of data
TDFD	Write Word	0x08090A0B	4 bytes of data
TDFD	Write Word	0x10111213	4 bytes of data
TDFD	Write Word	0x18191A1B	4 bytes of data
TDFD	Write Word	0x20212223	4 bytes of data
TDFD	Write Word	0x28292A2B	4 bytes of data
TDFV	Read Word	0x000001F4	Read the transmit FIFO vacancy
TLR	Write Word	0x00000020	Transmit length (0x20 = 32bytes), this starts transmission
ISR	Read Word	0x08000000	A typical value after TX Complete is indicated by interrupt
ISR	Write Word	0xFFFFFFFF	Write to clear transmit complete interrupt bits
ISR	Read Word	0x00000000	Read interrupt status register
TDFV	Read Word	0x000001FC	Read the transmit FIFO vacancy
Receive a Packet			
ISR	Read Word	0x04000000	A typical value after RX Complete is indicated by interrupt
ISR	Write Word	0xFFFFFFFF	Write to clear receive complete interrupt bits
ISR	Read Word	0x00000000	Read interrupt status register
RDFO	Read Word	0x00000008	Read the receive FIFO occupancy

Table 3-1: **Programming Sequence for TX and RX in Store-and-Forward Mode *(Cont'd)***

Register	Access	Value	Activity
RLR	Read Word	0x00000020	Receive length (0x20 =32 bytes) indicates number of bytes to read
RDR	Read Word	0x00000002	Receive Destination address (0x2 = destination device address is 2)
RDFD	Read Word	0x00000008	Read the receive FIFO occupancy
RDFD	Read Word	0xFFFFFFFF	4 bytes of data
RDFD	Read Word	0x12345678	4 bytes of data
RDFD	Read Word	0x00010203	4 bytes of data
RDFD	Read Word	0x08090A0B	4 bytes of data
RDFD	Read Word	0x10111213	4 bytes of data
RDFD	Read Word	0x18191A1B	4 bytes of data
RDFD	Read Word	0x20212223	4 bytes of data
RDFD	Read Word	0x28292A2B	4 bytes of packet data and CRC value
RDFO	Read Word	0x00000000	Read the receive FIFO occupancy (no further receive packets to process)

Table 3-2: **Programming Sequence for TX and RX in Cut-Through Mode**

Register	Access	Value	Activity
Power-up Read of Register Values			
ISR	Read Word	0x01D00000	Read interrupt status register (indicates transmit reset complete and receive reset complete)
ISR	Write Word	0xFFFFFFFF	Write to clear reset done interrupt bits
ISR	Read Word	0x00000000	Read interrupt status register
IER	Read Word	0x00000000	Read interrupt enable register
TDFV	Read Word	0x000001FC	Read the transmit FIFO vacancy
RDFO	Read Word	0x00000000	Read the receive FIFO occupancy
Transmit a Packet in Cut-Through Mode			
IER	Write Word	0x0C000000	Enable transmit complete and receive complete interrupts
TDR	Write Word	0x00000002	Transmit Destination address (0x2 = destination device address is 2)
TDFD	Write Word	0xFFFFFFFF	4 bytes of data
TDFD	Write Word	0x12345678	4 bytes of data
TDFD	Write Word	0x00010103	4 bytes of data
TDFD	Write Word	0x08090A0B	4 bytes of data

Table 3-2: **Programming Sequence for TX and RX in Cut-Through Mode *(Cont'd)***

Register	Access	Value	Activity
TDFD	Write Word	0x10111213	4 bytes of data
TDFD	Write Word	0x18191A1B	4 bytes of data
TDFD	Write Word	0x20212223	4 bytes of data
TDFD	Write Word	0x28292A2B	4 bytes of data
TLR	Write Word	0x00000020	Transmit length (0x20 = 32bytes), this starts transmission
ISR	Read Word	0x08000000	A typical value after TX Complete is indicated by interrupt
ISR	Write Word	0xFFFFFFFF	Write to clear transmit complete interrupt bits
ISR	Read Word	0x00000000	Read interrupt status register
TDFV	Read Word	0x000001FC	Read the transmit FIFO vacancy
Receive a Packet in Cut-Through Mode			
IER	Write Word	0x04100000	Enable receive complete and Receive FIFO Programmable Full (RFPF) threshold interrupts
ISR	Read Word	0x00100000	A typical value after RFPF is indicated by interrupt
ISR	Write Word	0x00100000	Reset RFPF interrupt
RLR	Read Word	0x80000010	Read the receive FIFO occupancy. If bit-31 is 1, it indicates that a partial packet is available. 0x80000010 = 16 bytes
RDFD	Read Word	0xFFFFFFFF	Started reading partial packet. 4 bytes of data
RDFD	Read Word	0x21031987	4 bytes of data
RDFD	Read Word	xAEF10011	4 bytes of data
RDFD	Read Word	0x27071985	4 bytes of data
ISR	Read Word	0x04000000	A typical value after RX complete is indicated by interrupt
ISR	Write Word	0x04000000	Reset RX complete interrupt
RLR	Read Word	0x00000020	Receive length (0x20 = 32 bytes) indicates number of bytes to be read
RDR	Read Word	0x00000002	Receive Destination address (0x2 = destination device address is 2)
RDFD	Read Word	0x10101021	Reading remaining 16 bytes. 4 bytes of data
RDFD	Read Word	0x21041987	4 bytes of data

Table 3-2: **Programming Sequence for TX and RX in Cut-Through Mode *(Cont'd)***

Register	Access	Value	Activity
RDFD	Read Word	xAEF10011	4 bytes of data
RDFD	Read Word	0x27061985	4 bytes of data
ISR	Read Word	0x04000000	A typical value after RX complete is indicated by interrupt
ISR	Write Word	0x04000000	Reset RX complete interrupt
RLR	Read Word	0x80000014	Read the receive FIFO occupancy. If bit 31 is 0, it indicates that a full packet is available. 0x80000014 = 20 bytes
RDR	Read Word	0x00000003	Receive Destination address (0x3 = destination device address is 3)
RDFD	Read Word	0x10101010	Reading packet. 4 bytes of data
RDFD	Read Word	0x20041981	4 bytes of data
RDFD	Read Word	xAEF1001F	4 bytes of data
RDFD	Read Word	0x21021958	4 bytes of data
RDFD	Read Word	0x0E0CB231	4 bytes of data
RDFO	Read Word	0x00000000	Read the receive FIFO occupancy

Programming Sequence Using Example Software Driver

Init Sequence

Call the function **`XLlFifo_CfgInitialize`** for the initialization that does a reset of the TX/RX registers and then clear the ISR.

Transmit a Packet

1. Call **`SetupInterruptSystem`** function which does the initialization of the interrupt controller
2. Call the function **`XLlFifo_IntEnable`** to enable all the required interrupts.
3. Call the **`TxSend`** function to write the data to TXFIFO and check TDFV for the FIFO occupancy before writing to the TX FIFO using the **`XLFifo_iTxVacancy`** function.
4. Start transmission by writing to TLR using the **`XLIFifo_iTxSetLen`** function.
5. Wait for the data transmission to complete, then the call **`FifoHandler`** interrupt handler and if it is a TX complete, then call **`FifoSendHandler`** and then clear the ISR.

Receive a Packet

1. On receiving the RX Interrupt, the **`FifoHandler`** interrupt handler is called which in turn calls the **`FifoRecvHandler.`**
2. Read RLR to find the number of bytes.
3. Read the number of bytes till the number of bytes read from RLR while checking the occupancy by using the **`XLlFifo_iRxOccupancy`** function.
4. Clear the ISR.
5. Check if there is an Interrupt pending for RX and then go to step 2.

Design Flow Steps

This chapter describes customizing and generating the core, constraining the core, and the simulation, synthesis and implementation steps that are specific to this IP core. More detailed information about the standard Vivado® design flows and the IP integrator can be found in the following Vivado Design Suite user guides:

- *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 3]
- *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 8]
- *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 9]
- *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 10]

Customizing and Generating the Core

This section includes information about using Xilinx tools to customize and generate the core in the Vivado Design Suite environment. You can customize the IP for use in your design by specifying values for the various parameters associated with the IP core using the following steps:

1. Select the IP from the IP catalog. The AXI4-Stream FIFO core is located under AXI Infrastructure in the Vivado IP catalog.
2. Double-click the selected IP or select the Customize IP command from the toolbar or right-click menu .

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 8] and the *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 9].

Note: Figures in this chapter are illustrations of the Vivado Integrated Design Environment (IDE). This layout might vary from the current version.

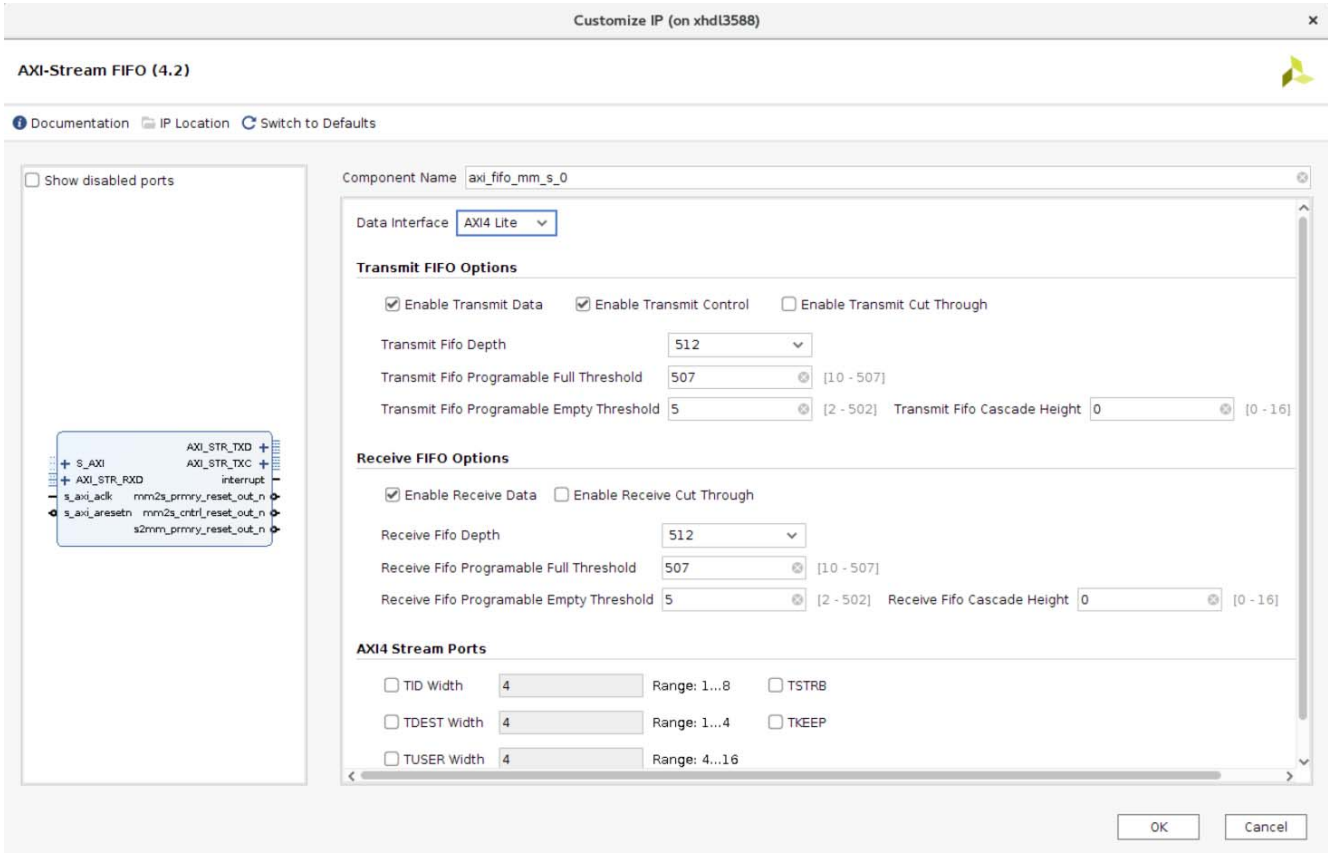

Figure 4-1: **Vivado IDE for AXI4-Stream FIFO**

- **Data Interface**:
  - **AXI4-Lite**: AXI4-Lite interface is for register access and transmit/receive FIFO accesses. Supported data width in this mode is 32.
  - **AXI4**: In this mode, all register accesses, except transmit/receive data FIFO registers, are accessed using the AXI4-Lite interface. Only transmit/receive data FIFO registers are accessed using the AXI4 interface. Data bursting is possible only in the AXI4 mode. The supported data width in this mode is 32, 64, 128, 256, and 512 (only for transmit/receive data FIFO registers).
- **AXI4 Data Width**: The AXI4 data width defines the width of the transmit/receive data FIFO registers. Supported data width is 32, 64, 128, 256, and 512.
- **AXI4 ID Width**: AXI4 ID width defines the width of the AWID/BID/ARID/RID ports. Supported ID width is 0 to 32.

- **Transmit FIFO**:
  - **Enable Transmit Data**: Enables the transmit datapath (from AXI4 interface to AXI4-Stream interface).
  - **Enable Transmit Control**: Enables the transmit control. The AXI4-Stream Transmit Control Interface supports the transmit protocol of AXI Ethernet cores.
  - **Enable Transmit Cut-Through**: Enables the cut-through mode in which packet transmission begins on the AXI4-Stream interface when there is enough data in the FIFO. When this option is not selected, the FIFO operates in Store-and-Forward Mode.
  - **Transmit Fifo Depth**: Valid range of the transmit FIFO depth is 512 to 128 k locations (powers of 2).
  - **Transmit Fifo Programmable Full Threshold:** The valid range for this threshold is provided in the IDE. When the difference between the read and write pointers of the transmit FIFO reaches the programmable FULL threshold value, the TFPF bit in ISR is set.
  - **Transmit Fifo Programmable Empty Threshold**: The valid range for this threshold is provided in the IDE. When the difference between the read and write pointers of the transmit FIFO reaches the programmable EMPTY threshold value, the TFPE bit in ISR is set.
  - **Transmit Fifo Cascade Height**: This parameter specifies the number of block RAMs present in one cascade chain of a Transmit FIFO. To implement a memory that requires more than 1 block RAM, it needs to connect several block RAMs using built-in cascade and/or fabric LUTs (MUXes). When cascade_height = 1, synthesis uses NO cascade at all. This is used for maximum timing performance. Default value is 0 (that is, synthesis has an option to choose cascading or not).
- **Receive FIFO**:
  - **Enable Receive Data**: Enables the receive datapath (from AXI4-Stream interface to AXI4 interface).
  - **Enable Receive Cut-Through**: Enables the cut-through mode in which packet reception begins on the AXI4 interface when there is enough data in the FIFO. When this option is not selected, the FIFO operates in Store-and-Forward Mode.
  - **Receive Fifo Depth**: Valid range of the receive FIFO depth is 512 to 128 k locations (powers of 2).
  - **Receive Fifo Programmable Full Threshold**: The valid range for this threshold is provided in the IDE. When the difference between the read and write pointers of the receive FIFO reaches the programmable FULL threshold value, the RFPF bit in ISR is set.
  - **Receive Fifo Programmable Empty Threshold:** The valid range for this threshold is provided in the IDE. When the difference between the read and write pointers of

the receive FIFO reaches the programmable EMPTY threshold value, the RFPE bit in ISR is set.

- **Receive Fifo Cascade Height**: This parameter specifies the number of block RAMs present in one cascade chain of a Receive FIFO. To implement a memory that requires more than 1 block RAM, it needs to connect several block RAMs using built-in cascade and/or fabric LUTs (MUXes). When cascade_height = 1, synthesis uses NO cascade at all. This is used for maximum timing performance. Default value is 0 (that is, synthesis has an option to choose cascading or not).

Note: The Full and Empty threshold range is different from previous version of the core (v4.1) due to the use of XPM libraries. See the IP GUI for the valid range of values.

- **AXI4-Stream Ports**: The AXI4-Stream FIFO configures the widths for TUSER, TID and TDEST signals. The valid range of these signals is provided in the IDE. For TKEEP and TSTRB signals, the width is determined by the configured DATA width and is internally calculated by using the equation (DATA Width)/8. See Table 2-4, page 19 for the list of supported registers.

User Parameters

Table 4-1 shows the relationship between the GUI fields in the Vivado IDE and the User Parameters (which can be viewed in the Tcl console).

Table 4-1: **GUI Parameter to User Parameter Relationship**

GUI Parameter/Value	User Parameter/Value	Default Value
Component Name	Component_Name	
AXI4 ID Width	C_S_AXI_ID_WIDTH	4
Data Interface	C_DATA_INTERFACE_TYPE	AXI4-Lite
AXI4 Data Width	C_S_AXI4_DATA_WIDTH	32
Transmit FIFO Options		
Enable Transmit Data	C_USE_TX_DATA	True: 1
Enable Transmit Control	C_USE_TX_CTRL	True: 1
Enable Transmit Cut Through	C_USE_TX_CUT_THROUGH	False: 0
Transmit FIFO Depth	C_TX_FIFO_DEPTH	512
Transit FIFO Programmable Full Threshold	C_TX_FIFO_PF_THRESHOLD	507
Transmit FIFO Programmable Empty Threshold	C_TX_FIFO_PE_THRESHOLD	5
Transmit FIFO Cascade Height	C_TX_CASCADE_HEIGHT	0
Receive FIFO Options		
Enable Receive Data	C_USE_RX_DATA	True: 1
Enable Receive Cut Through	C_USE_RX_CUT_THROUGH	False: 0
Receive FIFO Depth	C_RX_FIFO_DEPTH	512

Table 4-1: **GUI Parameter to User Parameter Relationship *(Cont'd)***

GUI Parameter/Value	User Parameter/Value	Default Value
Receive FIFO Programmable Full Threshold	C_RX_FIFO_PF_THRESHOLD	507
Receive FIFO Programmable Empty Threshold	C_RX_FIFO_PE_THRESHOLD	5
Receive FIFO Cascade Height	C_RX_CASCADE_HEIGHT	0
AXI4-Stream Ports		
	C_HAS_AXIS_TUSER	False: 0
	C_HAS_AXIS_TID	False: 0
	C_HAS_AXIS_TDEST	False: 0
TID Width	C_AXIS_TID_WIDTH	4
TUSER Width	C_AXIS_TUSER_WIDTH	4
TDEST Width	C_AXIS_TDEST_WIDTH	4
TSTRB	C_HAS_AXIS_TSTRB	False: 0
TKEEP	C_HAS_AXIS_TKEEP	False: 0

Output Generation

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 8].

Constraining the Core

There are no constraints associated with this core.

Simulation

For comprehensive information about Vivado simulation components, as well as information about using supported third party tools, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 10].

IMPORTANT: *For cores targeting 7 series or Zynq-7000 devices, UNIFAST libraries are not supported. Xilinx IP is tested and qualified with UNISIM libraries only.*

Synthesis and Implementation

For details about synthesis and implementation, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 8].

Appendix A

Verification, Compliance, and Interoperability

This appendix provides details about how this IP core was tested for compliance.

Simulation

The AXI4-Stream FIFO has been tested with Xilinx® ISim, and Mentor Graphics Questa® SIM simulator.

Hardware Testing

The AXI4-Stream FIFO has been hardware validated at 200 MHz on a KC705 board using Kintex®-7 -2 speed grade device (325T). The IP was configured for AXI4-Lite to work with Xilinx AXI Ethernet IP.

Upgrading

This appendix contains information about migrating a design from ISE® to the Vivado® Design Suite, and for upgrading to a more recent version of the IP core. For customers upgrading in the Vivado Design Suite, important details (where applicable) about any port changes and other impact to user logic are included.

Migrating to the Vivado Design Suite

For information about migrating to the Vivado Design Suite, see the *ISE to Vivado Design Suite Migration Guide* (UG911) [Ref 12].

Upgrading in the Vivado Design Suite

This section provides information about any changes to the user logic or port designations that take place when you upgrade to a more current version of this IP core in the Vivado Design Suite.

Parameter Changes

There are no parameter changes from the previous release to this release.

Port Changes

There are no port changes from the previous release to this release.

Debugging

This appendix provides information for using the resources available on the Xilinx Support website, debug tools, and other step-by-step processes for debugging designs that use the AXI4-Stream FIFO core.

Finding Help on Xilinx.com

To help in the design and debug process when using the AXI4-Stream FIFO, the Xilinx Support web page contains key resources such as product documentation, release notes, answer records, information about known issues, and links for obtaining further product support.

Documentation

This product guide is the main document associated with the AXI4-Stream FIFO core. This guide, along with documentation related to all products that aid in the design process, can be found on the Xilinx Support web page or by using the Xilinx Documentation Navigator.

Download the Xilinx Documentation Navigator from the Downloads page. For more information about this tool and the features available, open the online help after installation.

Answer Records

Answer Records include information about commonly encountered problems, helpful information on how to resolve these problems, and any known issues with a Xilinx product. Answer Records are created and maintained daily ensuring that users have access to the most accurate information available.

Answer Records for this core can be located by using the Search Support box on the main Xilinx support web page. To maximize your search results, use proper keywords such as:

- Product name
- Tool message(s)
- Summary of the issue encountered

A filter search is available after results are returned to further target the results.

Master Answer Record for AXI4-Stream FIFO

AR: 54447

Technical Support

Xilinx provides technical support at the Xilinx Support web page for this LogiCORE™ IP product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support if you do any of the following:

- Implement the solution in devices that are not defined in the documentation.
- Customize the solution beyond that allowed in the product documentation.
- Change any section of the design labeled DO NOT MODIFY.

To contact Xilinx Technical Support, navigate to the Xilinx Support web page.

Debug Tools

There are many tools available to address AXI4-Stream FIFO core design issues. It is important to know which tools are useful for debugging various situations.

Vivado Design Suite Debug Feature

The Vivado® Design Suite debug feature inserts logic analyzer and virtual I/O cores directly into your design. The debug feature also allows you to set trigger conditions to capture application and integrated block port signals in hardware. Captured signals can then be analyzed. This feature in the Vivado IDE is used for logic debugging and validation of a design running in Xilinx devices.

The Vivado logic analyzer is used with the logic debug IP cores, including:

- ILA 2.0 (and later versions)
- VIO 2.0 (and later versions)

See the *Vivado Design Suite User Guide: Programming and Debugging* (UG908) [Ref 13].

Simulation Debug

For details about simulating a design in the Vivado® Design Suite, see the *Vivado Logic Simulation User Guide* (UG900) [Ref 10].

Hardware Debug

Hardware issues can range from link bring-up to problems seen after hours of testing. This section provides debug steps for common issues. The Vivado debug feature is a valuable resource to use in hardware debug. The signal names mentioned in the following individual sections can be probed using the debug feature for debugging the specific problems.

General Checks

Ensure that all the timing constraints were met during implementation.

- Ensure that all clock sources are active and clean.
- If the interrupts do not occur, check if the interrupts are enabled.
- If the design is unresponsive, check if the programming sequence is correct.

Appendix D

Additional Resources and Legal Notices

Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see Xilinx Support.

Documentation Navigator and Design Hubs

Xilinx® Documentation Navigator provides access to Xilinx documents, videos, and support resources, which you can filter and search to find information. To open the Xilinx Documentation Navigator (DocNav):

- From the Vivado® IDE, select **Help > Documentation and Tutorials**.
- On Windows, select **Start > All Programs > Xilinx Design Tools > DocNav**.
- At the Linux command prompt, enter **`docnav`**.

Xilinx Design Hubs provide links to documentation organized by design tasks and other topics, which you can use to learn key concepts and address frequently asked questions. To access the Design Hubs:

- In the Xilinx Documentation Navigator, click the **Design Hubs View** tab.
- On the Xilinx website, see the Design Hubs page.

Note: For more information on Documentation Navigator, see the Documentation Navigator page on the Xilinx website.

References

These documents provide supplemental material useful with this product guide:

1. *AMBA AXI4-Stream Protocol Specification*
2. *AMBA AXI4 Protocol Specification*

3. *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994)
4. *AXI 1G/2.5G Ethernet Subsystem Product Guide* (PG138)
5. *LogiCORE IP AXI Slave Burst* (DS769)
6. *Virtex-6 Family Overview* (DS150)
7. *7 Series FPGAs Overview* (DS180)
8. *Vivado Design Suite User Guide: Designing with IP* (UG896)
9. *Vivado Design Suite User Guide: Getting Started* (UG910)
10. *Vivado Design Suite User Guide: Logic Simulation* (UG900)
11. *Vivado Design Suite User Guide: Implementation* (UG904)
12. *ISE to Vivado Design Suite Migration Methodology Guide* (UG911)
13. *Vivado Design Suite User Guide: Programming and Debugging* (UG908)

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
10/30/2019	4.2	• Added families to Supported Device Family and added Release Notes and Known Issues row to IP Facts. • Updated Figure 4-1 and added Transmit/Receive Fifo Cascade Height descriptions. • Added Transmit/Receive Fifo Cascade Height to Table 4-1.
05/22/2019	4.2	• Updated configurable data width in Features. • Added RX FIFO note in Protocol Description. • Added description on inactive reset in Transmit Data FIFO Reset Register (TDFR). • Added description on inactive reset in Receive Data FIFO Reset Register (RDFR). • Updated supported data width for AXI4 and AXI4 Data Width in Customizing and Generating the Core.
12/05/2018	4.2	• Updated standalone driver links in Features. • Removed note in Interrupt Status Register Bit Definitions table. • Updated Figure 4-1. • Updated TX and RX FIFO Programmable Empty Threshold default value to 5 in GUI Parameter to User Parameter Relationship table.
04/06/2016	4.1	• Corrected the interrupt signal definitions. • Corrected configurable data widths. • Corrected the Interrupt Status Register Bit Definitions.

Date	Version	Revision
04/01/2015	4.1	• Updated the read interrupt status register (ISR) value in the Programming Sequence for TX and RX in Cut-Through Mode Table. • Updated the Reset Values for Receive FIFO Programmable Full (RFPF) and Transmit FIFO Programmable Full (TFPF) in the Interrupt Status Register Bit Definitions Table. • Updated the Transmit Length Register (TLR) width to support packets up to 8 Mbytes–4. • Updated the Transmit Length Register (TLR) to support packets up to 8388604 in length.
10/01/2014	4.1	• Added data widths of 128, 256 and 512 for AXI4 Data Interface. • Updated the following register definitions: ◦ Transmit Data FIFO Vacancy Register (TDFV) ◦ Transmit Data FIFO Data Write Port (TDFD) ◦ Receive Data FIFO Occupancy Register (RDFO) ◦ Receive Data FIFO Data Read Port (RDFD) ◦ Transmit Length Register (Store-and-Forward Mode) ◦ Transmit Length Register (Cut-Through Mode) ◦ Receive Length Register (Store-and-Forward Mode) ◦ Receive Length Register (Cut-Through Mode) • Updated AXI4 ID Width to 0-16. • Updated the Transmit and Receive FIFO Depth ranges to 512 - 128 k (powers of 2). • Offset address for TDFD and RDFD registers changed from 0x10 to 0x0000 and 0x20 to 0x1000.
04/02/2014	4.0	• Corrected the value for the Interrupt Status Register (ISR) in Table 1-1 and 1-2. • Changed the name of the Receive Length FIFO (RLF) register to Receive Length register (RLR). • Updated the maximum packet length of Transmit Data FIFO Data Write Port and the Receive Data FIFO Data Read Port. • Added details about Store-and-Forward and Cut-Through modes to the Receive Length and Transmit Length registers.
12/18/2013	4.0	• Added support for UltraScale™ architecture.
10/02/2013	4.0	• Added details to Vivado Integrated Design Environment (IDE) in Chapter 4. • Added selectable transmit and receive path. • Added support for IP integrator.
03/20/2013	3.0	• Updated core to v4.0. • Removed support for ISE Design Suite. • Added Appendix B, Debugging.

Date	Version	Revision
12/18/2012	2.0	Updated core to v3.00b and Vivado Design Suite for 2012.4. • Updated the lengths of the Transmit Length Register (TLR), page 31 and the Receive Length Register (RLR), page 32. • Clarified which ports are not used by the core in Table 2-3.
10/16/2012	1.0	Initial Xilinx release as a product guide. Replaces DS806, *LogiCORE IP AXI4-Stream FIFO Data Sheet*. • Changed the size of the FIFO to 508 words. • Added support for an AXI4 interface, TX cut-through mode, and a 64-bit data path.

Please Read: Important Legal Notices