LogiCORE IP AXI4-Stream FIFO v3.00b

Product Guide

PG080 December 18, 2012

XILINX®

XILINX

Table of Contents

SECTION I: SUMMARY

Introduction

The AXI4-Stream FIFO core allows memory mapped access to a AXI4-Stream interface. The core can be used to interface to the AXI Ethernet without the complexity or resource utilization of using DMA.

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 signalling. The management of the AXI4-Stream interfaces is transparent to the user.

Features

- 32-bit AXI4-Lite slave interface
- Configurable data interface type (AXI4 or AXI4-Lite)
- Configurable data width of 32 or 64-bit (AXI4 Data Interface only)
- Independent configurable internal TX and RX data FIFOs
- Full duplex operation
- Supports AXI Ethernet basic mode
- Provides interrupts for error and status conditions
- TX cut-through mode

LogiCORE IP Facts Table	
Core Specifics	
Supported Device Family(1)	Zynq™-7000, Artix™-7, Virtex®-7, Kintex™-7, Virtex-6, Spartan®-6
Supported User Interfaces	AXI4, AXI4-Lite, AXI4-Stream
Resources	See Resource Utilization.
Provided with Core	
Design Files	VHDL
Example Design	Not Provided
Test Bench	Not Provided
Constraints File	Not Provided
Simulation Model	Not Provided
Supported S/W Driver(2)	Standalone
Tested Design Flows(3)	
Design Entry	ISE Design Suite Embedded Edition v14.3 Vivado Design Suite v2012.4(4)
Simulation	Mentor Graphics ModelSim
Synthesis	XST 14.3 Vivado Synthesis v2012.4
Support	
Provided by Xilinx @ www.xilinx.com/support	

Notes:

1. For a complete list of supported derivative devices, see Embedded Edition Derivative Device Support.
2. Standalone driver details can be found in the EDK or SDK directory (<*install_directory*>/doc/usenglish/xilinx_drivers.htm). Linux OS and driver support information is available from //wiki.xilinx.com.
3. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.
4. Supports only 7 series devices.

Overview

Figure 1-1 shows the major components in the AXI4-Stream FIFO core — an AXI Interface block with an AXI4/AXI4-Lite Slave interface, Interrupt Controller, a Registers module, a Receive Control Module, a Transmit Control Module, a Receive FIFO for the receive data and length, and a Transmit FIFO for the transmit data and the length.

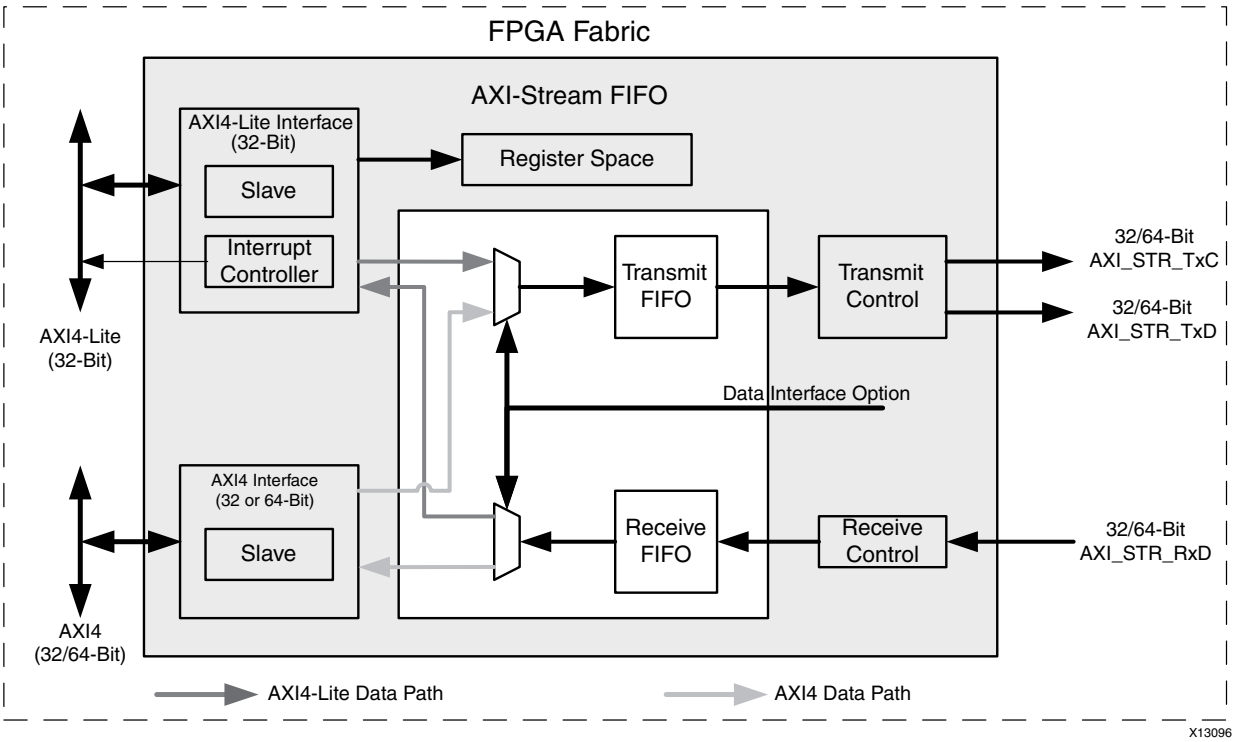

Figure 1-1: **AXI4-Stream FIFO Core Block Diagram**

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 Embedded Development Kit (EDK) to attach the AXI4-Stream FIFO core, AXI Ethernet core, processor, memory, interconnect the buses, clocking, and additional embedded components.

This section briefly 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.

Packets will automatically be transmitted by writing packet data to the transmit data FIFO followed by writing a length in the TLR.

When the TX cut-through option is selected, packets are automatically transmitted as and when a packet is written to the transmit data FIFO. The last data beat, along with TLAST, is transmitted after writing a length in the TLR. This feature is used to transmit a large packet size with the minimum FIFO size.

Receiving packets functions in a manner similar to transmission, except that the steps are reversed. Receiving the ISR receive complete interrupt or polling of the receive data FIFO occupancy register (when not zero) indicates the reception of a packet. Reading the RLR provides the packet length in bytes. Given the number of received packet bytes, the appropriate number of reads of the Read data FIFO will provide the packet data.

Table 1-1 illustrates a power-up read of the registers followed by a transmission and reception of a single packet. See the register definitions for further information and options.

Table 1-1: **Sample TX and RX Usage**

Register	Access	Value	Activity
Power-up Read of Register Values			
ISR	Read Word	0x01800000	Read interrupt status register
ISR	Write Word	0x0FFFFFFFF	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	0x000001FE	Read the transmit FIFO vacancy
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)
TXFIFO_DATA	Write Word	0xFFFFFFFF	4 bytes of data
TXFIFO_DATA	Write Word	0x12345678	4 bytes of data
TXFIFO_DATA	Write Word	0x00010203	4 bytes of data
TXFIFO_DATA	Write Word	0x08090A0B	4 bytes of data
TXFIFO_DATA	Write Word	0x10111213	4 bytes of data
TXFIFO_DATA	Write Word	0x18191A1B	4 bytes of data

Table 1-1: **Sample TX and RX Usage *(Cont'd)***

Register	Access	Value	Activity
TXFIFO_DATA	Write Word	0x20212223	4 bytes of data
TXFIFO_DATA	Write Word	0x28292A2B	4 bytes of data
TDFV	Read Word	0x000001F6	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	0x0FFFFFFF	Write to clear reset done interrupt bits
ISR	Read Word	0x00000000	Read interrupt status register
TDFV	Read Word	0x000001FE	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	0x0FFFFFFF	Write to clear reset done interrupt bits
ISR	Read Word	0x00000000	Read interrupt status register
RDFO	Read Word	0x00000008	Read the receive FIFO occupancy
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)
RDFO	Read Word	0x00000008	Read the receive FIFO occupancy
RXFIFO_DATA	Read Word	0xFFFFFFFF	4 bytes of data
RXFIFO_DATA	Read Word	0x12345678	4 bytes of data
RXFIFO_DATA	Read Word	0x00010203	4 bytes of data
RXFIFO_DATA	Read Word	0x08090A0B	4 bytes of data
RXFIFO_DATA	Read Word	0x10111213	4 bytes of data
RXFIFO_DATA	Read Word	0x18191A1B	4 bytes of data
RXFIFO_DATA	Read Word	0x20212223	4 bytes of data
RXFIFO_DATA	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)

Applications

This core converts AXI4/AXI4-Lite transactions to 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 Information

This Xilinx LogiCORE IP module is provided at no additional cost with the Xilinx Vivado Design Suite and ISE Design Suite Embedded Edition tools 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 the following 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 will vary from the results reported here.

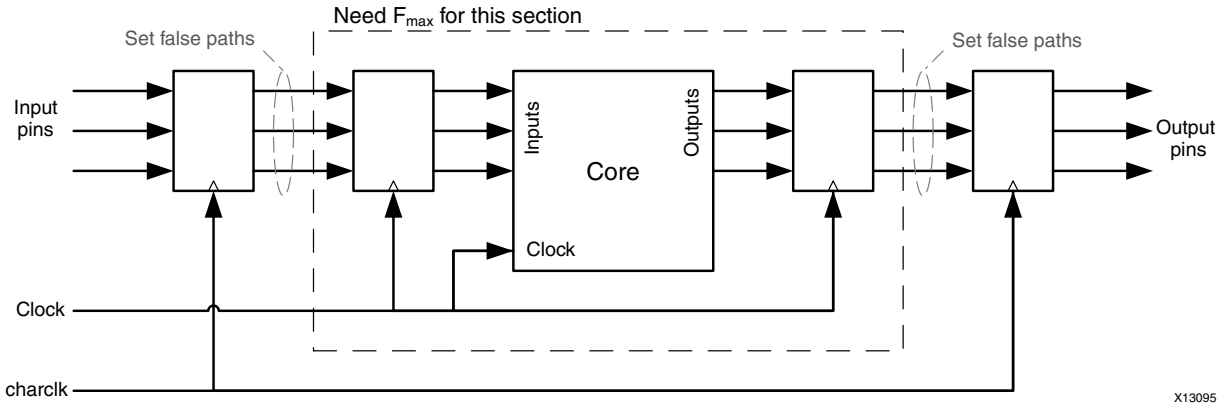

Figure 2-1: **Virtex-7 FPGA with the AXI4-Stream FIFO Core**

Maximum Frequencies

When targeting Virtex-7 devices, the AXI4-Stream FIFO can operate up to 300 MHz with the maximum possible configuration. The maximum possible configurations are:

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

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 packet mode, AXI4-Stream transactions will 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 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 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.

Resource Utilization

Because the AXI4-Stream FIFO core will be used with other design pieces in the FPGA, the utilization and timing numbers reported in this section are just estimates. As the AXI4-Stream FIFO core is combined with other pieces of the FPGA design, the utilization of FPGA resources and timing of the AXI4-Stream FIFO core design will vary from the results reported here.

The benchmarks shown in Table 2-1 were obtained with the Data Interface option set to AXI4-Lite, a data width of 32 and both Transmit FIFO Depth and Receive FIFO Depth of 512 words.

Table 2-1: **Resource Utilization with AXI4-Lite, 32 Data Width and FIFO Depths of 512**

Family	Block RAMs	Slices	Slice Flip-Flops	LUTs	F_{MAX} (MHz)
Virtex7	2	311	692	728	336
Kintex7	2	315	692	727	250
Artix7	2	313	692	729	196

The benchmarks shown in Table 2-2 were obtained with the Data Interface option set to AXI4, a data width of 32 and both Transmit FIFO Depth and Receive FIFO Depth of 512 words.

Table 2-2: **Resource Utilization with AXI4, 32 Data Width and FIFO Depths of 512**

Family	Block RAMs	Slices	Slice Flip-Flops	LUTs	F_{max} (MHz)
Virtex7	2	331	739	813	250
Kintex7	2	337	739	807	334
Artix7	2	326	739	812	226

The benchmarks shown in Table 2-3 were obtained with the Data Interface option set to AXI4, a data width of 64 and both Transmit FIFO Depth and Receive FIFO Depth of 512 words.

Table 2-3: **Resource Utilization with AXI4, 64 Data Width and FIFO Depths of 512**

Family	Block RAMs	Slices	Slice Flip-Flops	LUTs	F_{max} (MHz)
Virtex7	2	375	883	857	320
Kintex7	2	379	883	864	266
Artix7	2	376	883	863	204

Port Descriptions

The AXI4-Stream FIFO core uses a transparent bus EDK format to simplify the connection of signals between the AXI4-Stream interface of the core and AXI4-Stream interfaces of other IP such as the AXI Ethernet core. This technique allows the EDK tools to automatically connect the AXI signals as an entire group.

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 AXI-Stream FIFO core does not support any advanced features and drives constant values on this interface. The AXI-Stream FIFO core follows the handshake requirements as defined by the AXI Ethernet Core. For more details on the handshake requirement, see "Normal Transmit AXI4-Stream Control Words" in DS759, *LogiCORE IP AXI Ethernet Data Sheet*.

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 AXI Stream input clocks of the AXI Ethernet core must use the same clock.

Table 2-4: **I/O Signals**

Signal Name	Dir	Width	Default	Description
Top Level System Signal				
Interrupt	OUT	1	'0'	Global Interface Clock: All signals on Interface must be synchronous to ACLK.
Global Signals				
S_AXI_ACLK	IN	1	'0'	Global Interface Clock: All signals on Interface must be synchronous to ACLK.
S_AXI_ARESETN	IN	1	'0'	Global reset: This signal is active-Low, ARESETN must be asserted at least for one clock cycle.
AXI4-Lite Write Address Channel Signals				
S_AXI_AWADDR	IN	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	IN	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	OUT	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.

Table 2-4: **I/O Signals *(Cont'd)***

Signal Name	Dir	Width	Default	Description
AXI4-Lite Write Data Channel Signals				
S_AXI_WDATA	IN	C_S_AXI_DATA_WIDTH	0	Write Data: The write data bus width is 32-bit only.
S_AXI_WSTRB	IN	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	IN	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.
S_AXI_WREADY	OUT	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	OUT	2	0	Write Response: Indicates the status of the write transaction. The allowable responses are OKAY, EXOKAY, SLVERR, and DECERR.
S_AXI_BVALID	OUT	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	IN	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	IN	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.

Table 2-4: **I/O Signals *(Cont'd)***

Signal Name	Dir	Width	Default	Description
S_AXI_ARVALID	IN	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	OUT	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-Lite Read Data Channel Signals				
S_AXI_RDATA	OUT	C_S_AXI_DATA_WIDTH	0	Read Data: The read data bus width is 32-bit only.
S_AXI_RRESP	OUT	2	0	Read Response: Indicates the status of the read transfer. The allowable responses are OKAY, EXOKAY, SLVERR, and DECERR.
S_AXI_RVALID	OUT	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	IN	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)	IN	1	'0'	ACLK: Clock for the AXI4-Stream Transmit data interface. Presently not used in the core.
MM2S_PRMRY_RESET_OUT_N	OUT	1	'0'	Reset: Reset for the AXI4-Stream Transmit data interface.
AXI_STR_TXD_TVALID	OUT	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	IN	1	'0'	TREADY: Indicates that the slave can accept a transfer in the current cycle.

Table 2-4: **I/O Signals *(Cont'd)***

Signal Name	Dir	Width	Default	Description
AXI_STR_TXD_TDATA	OUT	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 or 64 (AXI4 interface only).
AXI_STR_TXD_TKEEP	OUT	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	OUT	1	'0'	TLAST: Indicates the boundary of a packet.
AXI_STR_TXD_TDEST	OUT	4	0	TDEST: Destination AXI Stream Identifier and Provides routing information for the data stream.
AXI4-Stream Transmit Control Channel Signals				
AXI_STR_TXC_ACLK (1)	IN	1	'0'	ACLK: Clock for the AXI4-Stream Transmit data interface. Presently not used in the core.
MM2S_CNTRL_RESET_OUT_N	OUT	1	'0'	Reset: Reset for the AXI4-Stream Transmit data interface.
AXI_STR_TXC_TVALID	OUT	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	IN	1	'0'	TREADY: Indicates that the slave can accept a transfer in the current cycle
AXI_STR_TXC_TDATA	OUT	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 or 64 (AXI4 Interface only).
AXI_STR_TXC_TKEEP	OUT	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	OUT	1	'0'	TLAST: Indicates the boundary of a packet.
AXI4-Stream Receive Data Channel Signals				
AXI_STR_RXD_ACLK (1)	IN	1	'0'	ACLK: Clock for the AXI4-Stream Transmit data interface. Presently not used in the core.

Table 2-4: **I/O Signals *(Cont'd)***

Signal Name	Dir	Width	Default	Description
S2MM_PRMRY_RESET_OUT_N	OUT	1	'0'	Reset: Reset for the AXI4-Stream Transmit data interface.
AXI_STR_RXD_TVALID	IN	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	OUT	1	'0'	TREADY: Indicates that the slave can accept a transfer in the current cycle.
AXI_STR_RXD_TDATA	IN	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 or 64 (AXI4 interface only).
AXI_STR_RXD_TKEEP	IN	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	IN	1	'0'	TLAST: Indicates the boundary of a packet.
AXI_STR_RXD_TDEST	IN	4	0	TDEST: Destination AXI Stream Identifier and Provides routing information for the data stream.
AXI4 Write Address Channel Signals				
S_AXI4_AWID	IN	C_S_AXI_ID_WIDTH	0	Write Address ID: Identification tag for the write address group of signals
S_AXI4_AWADDR	IN	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	IN	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	IN	3	0	Burst Size: Indicates the size of each transfer in the burst.
S_AXI4_AWBURST	IN	2	0	Burst Type: The burst type, coupled with the size information, details how the address for each transfer within the burst is calculated.

Table 2-4: **I/O Signals *(Cont'd)***

Signal Name	Dir	Width	Default	Description
S_AXI4_AWLOCK	IN	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	IN	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.
S_AXI4_AWPROT	IN	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	IN	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	OUT	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	IN	C_S_AXI4_DATA_WIDTH	0	Write Data: The write data bus can be 32 or 64 bits wide.
S_AXI4_WSTRB	IN	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

Table 2-4: **I/O Signals *(Cont'd)***

Signal Name	Dir	Width	Default	Description
S_AXI4_WLAST	IN	1	'0'	Write Last: Indicates the last transfer in a write burst.
S_AXI4_WVALID	IN	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.
S_AXI4_WREADY	OUT	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	OUT	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	OUT	2	0	Write Response: Indicates the status of the write transaction. The allowable responses are OKAY, EXOKAY, SLVERR, and DECERR.
S_AXI4_BVALID	OUT	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	IN	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	IN	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	IN	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	IN	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 2-4: **I/O Signals *(Cont'd)***

Signal Name	Dir	Width	Default	Description
S_AXI4_ARSIZE	IN	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	IN	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
S_AXI4_ARLOCK	IN	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	IN	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	IN	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	IN	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	OUT	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	OUT	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.
S_AXI4_RDATA	OUT	C_S_AXI4_DATA_WIDTH	0	Read Data: The read data bus can be 32 or 64 bits wide.

Table 2-4: **I/O Signals *(Cont'd)***

Signal Name	Dir	Width	Default	Description
S_AXI4_RRESP	OUT	2	0	Read Response: Indicates the status of the read transfer. The allowable responses are OKAY, EXOKAY, SLVERR, and DECERR.
S_AXI4_RLAST	OUT	1	'0'	Read Last: Indicates the last transfer in a read burst.
S_AXI4_RVALID	OUT	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	IN	1	'0'	Read Ready: Indicates that the master can accept the read data and response information: • 1 = Master ready. • 0 = Master not ready.

1. This port is currently not used.

Register Space

The AXI4-Stream FIFO core contains the registers listed in Table 2-5.

Table 2-5: **Register Names and Descriptions**

Register Name	AXI Address	Access
Interrupt Status Register (ISR)	C_S_AXI_BASEADDR + 0x0	Read/Clear on Write(1)
Interrupt Enable Register (IER))	C_S_AXI_BASEADDR + 0x4	Read/Write
Transmit Data FIFO Reset (TDFR)	C_S_AXI_BASEADDR + 0x8	Write(2)
Transmit Data FIFO Vacancy (TDFV)	C_S_AXI_BASEADDR + 0xC	Read
Transmit Data FIFO 32-bit Wide Data Write Port (TDFD)	C_S_AXI_BASEADDR + 0x10 or C_S_AXI4_BASEADDR + 0x10	Write(3)
Transmit Length FIFO (TLF)	C_S_AXI_BASEADDR + 0x14	Write
Receive Data FIFO reset (RDFR)	C_S_AXI_BASEADDR + 0x18	Write(2)
Receive Data FIFO Occupancy (RDFO)	C_S_AXI_BASEADDR + 0x1C	Read
Receive Data FIFO 32-bit Wide Data Read Port (RDFD)	C_S_AXI_BASEADDR + 0x20 or C_S_AXI4_BASEADDR + 0x20	Read(3)
Receive Length FIFO (RLF)	C_S_AXI_BASEADDR + 0x24	Read
AXI4-Stream Reset (SRR)	C_S_AXI_BASEADDR + 0x28	Write(2)

Table 2-5: **Register Names and Descriptions *(Cont'd)***

Register Name	AXI Address	Access
Transmit Destination Register (TDR)	C_S_AXI_BASEADDR + 0x2C	Write
Receive Destination Register (RDR)	C_S_AXI_BASEADDR + 0x30	Read
Reserved	C_S_AXI_BASEADDR + 0x34 to C_S_AXI_BASEADDR + 0x3C	N/A(4)

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 may be cleared in a single write.
2. Reset if written with 0xA5.
3. C_S_AXI4_BASEADDR should be used only if the Data Interface option is AXI4.
4. If read, these registers will return 0x0. Writing these registers will have no effect.

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.

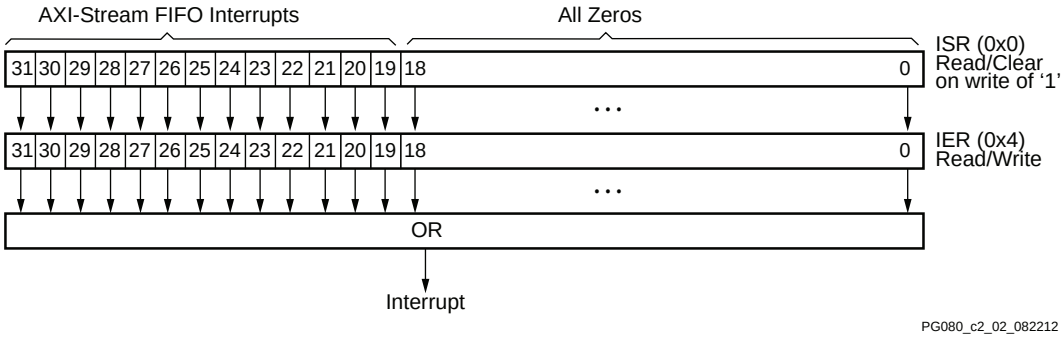

Figure 2-2: **Interrupt Control Structure**

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.

Once an interruptible condition occurs, it will be 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 may be cleared in a single write.

For any bit set in the Interrupt Status Register, a corresponding bit must be also set in the Interrupt Enable Register for the IP2INTC_Irpt signal to be driven active High out of the AXI4-Stream FIFO core.

The Interrupt Status Register bit definitions are detailed in Table 2-6.

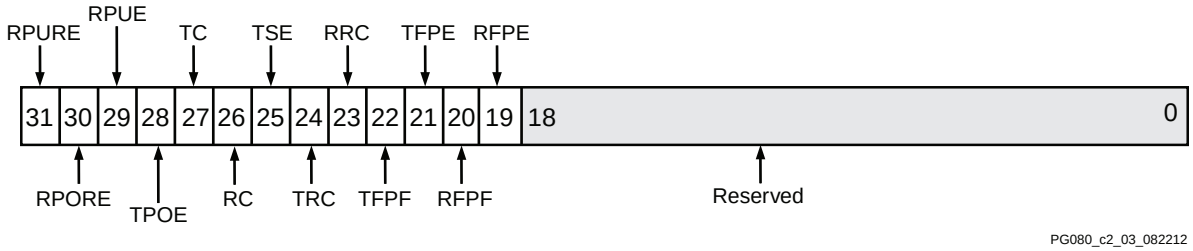

Figure 2-3: **Interrupt Status Register (offset 0x0)**

Table 2-6: **Interrupt Status Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
0-18	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. '0' = No interrupt pending '1' = Interrupt pending
20	RFPF	Read/Clear on Write of "1"	0	**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. '0' = No interrupt pending '1' = Interrupt pending
22	TFPF	Read/Clear on Write of "1"	0	**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"	0	**Receive Reset Complete**: This interrupt indicates that a reset of the receive logic has completed. '0' = No interrupt pending. '1' = Interrupt pending.

Table 2-6: **Interrupt Status Register Bit Definitions *(Cont'd)***

Bit(s)	Name	Core Access	Reset Value	Description
24	TRC	Read/Clear on Write of "1"	0	**Transmit Reset Complete**: This interrupt indicates that a reset of the transmit logic has completed. '0' = No interrupt pending. '1' = Interrupt pending.
25	TSE	Read/Clear on Write of "1"	0	**Transmit Size Error**: This interrupt is generated if the number of 32-bit 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 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 may 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 2-6: **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 may not be empty, the read has gone beyond the current packet and removed 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 "1" in a bit location enables the related interrupt from being propagated, while a value of "0" disables it.

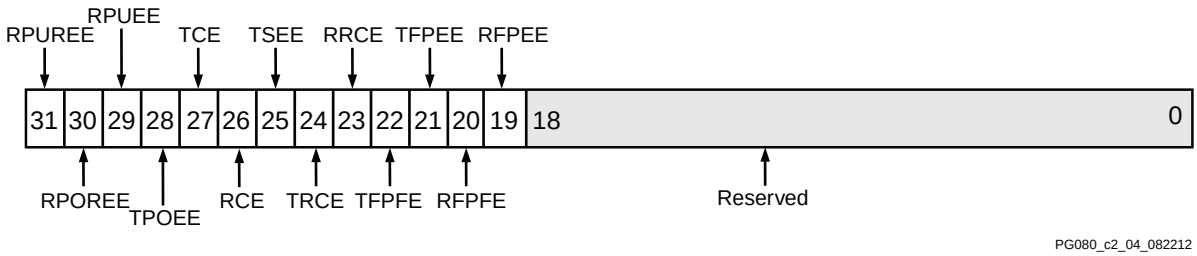

Figure 2-4: **Interrupt Enable Register (offset 0x4)**

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 AXI Stream has completed. The reset can occur only during inactive times on the TX AXI 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.

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.

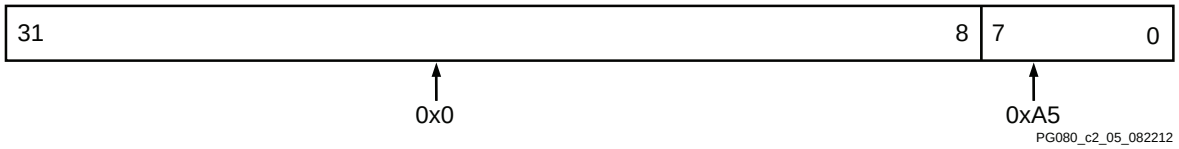

Figure 2-5: **Transmit Data FIFO Reset Register (offset 0x8)**

Table 2-7: **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.

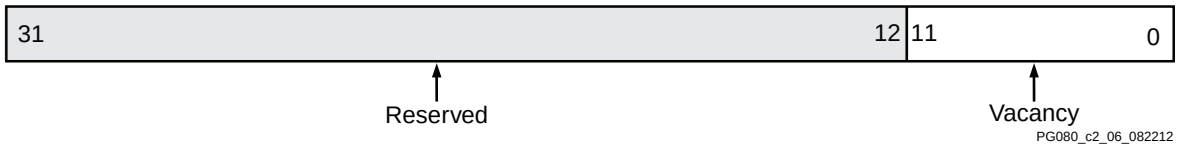

Figure 2-6: **Transmit Data FIFO Vacancy Register (offset 0xC)**

Transmit Data FIFO Data Write Port (TDFD)

The Transmit Data FIFO Data Write Port shown in Figure 2-7 is a 32-bit wide address location for writing data into the Transmit Data FIFO. The smallest packet that may be transmitted is four 32-bit words (include partial final word) which corresponds to 13 to 16 bytes. The maximum packet that may be transmitted is limited by the size of the FIFO which is 508 (include partial final word) words or 2029 to 2032 bytes.

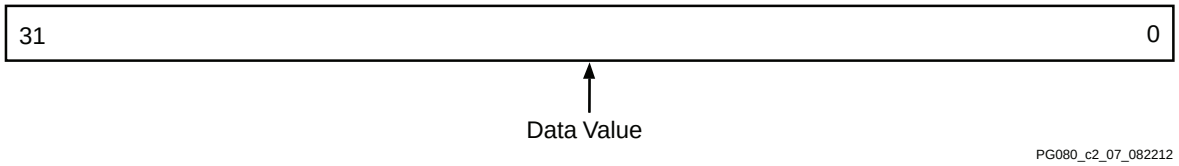

Figure 2-7: **Transmit Data FIFO Data Write Port (offset 0x10)**

Table 2-8: **Transmit Data FIFO Data Write Port Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
31-0	Write Data Value	Write	N/A	**Transmit Data FIFO Write Value**.

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 AXI Stream has completed. Only during inactive times on the RX AXI 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 may cause failure condition in that core.

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. An example transaction is if a packet is received over the AXI4-Stream that exceeds the FIFO size of this core causing the core's destination ready to become inactive in the middle of a transfer. In this case, an S_AXI_ARESETN reset is needed.

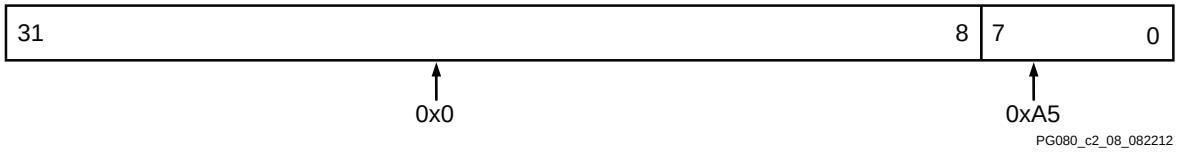

Figure 2-8: **Receive Data FIFO Reset Register (offset 0x18)**

Table 2-9: **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.

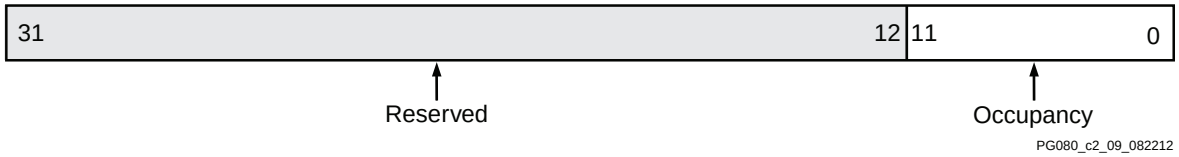

Figure 2-9: **Receive Data FIFO Occupancy Register (offset 0x1C)**

Table 2-10: **Receive Data FIFO Occupancy Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
11-0	Occupancy	Read	0x0	**Receive Data FIFO Occupancy**: This is the unsigned value reflecting a current snapshot of the number of 32-bit wide locations in use for data storage in the receive Data FIFO memory core. 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.
31-12	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 32-bit wide address location for reading data from the Receive Data FIFO. The smallest packet that may be received is a four 32-bit words packet which corresponds to 13 to 16 bytes. The maximum packet that may be received is limited by the size of the FIFO which is 508 words or 2029 to 2032 bytes.

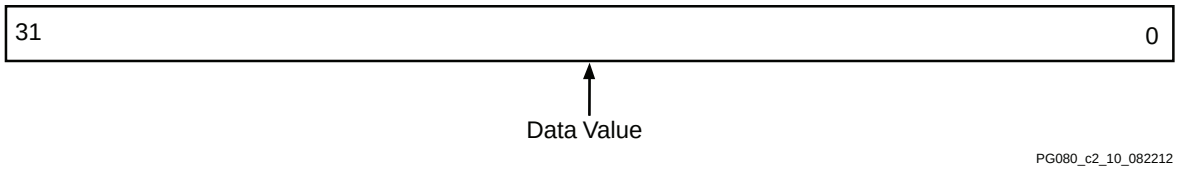

Figure 2-10: **Receive Data FIFO Data Read Port (offset 0x20)**

Table 2-11: **Receive Data FIFO Data Read Port Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
31-0	Read Data Value	Read	N/A	**Receive Data FIFO Read Value**.

Transmit Length Register (TLR)

The Transmit Length Register shown in Figure 2-11 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.

Once the packet length is written to the TLR, it is automatically moved to the Transmit Data FIFO with the packet data freeing up the TLR for another value. The packet length must be written to the TLR after the packet data is written to the transmit data FIFO. It is not valid to

write data for multiple packets to the transmit data FIFO before writing the packet length values.

The action of writing to the Transmit Length Register is used by the AXI4-Stream FIFO core to initiate the processing of transmit packets across the AXI4-Stream interface. This action continues until all of the TLR values that have been stored are processed.

The width of the TLR is wide enough to support packets up to 32 Kbytes in length. The smallest packet that may be transmitted is four 32-bit words (include partial final word) which corresponds to 13 to 16 bytes. The maximum packet that may be transmitted is limited by the size of the FIFO which is C_TX_FIFO_DEPTH-4 words.

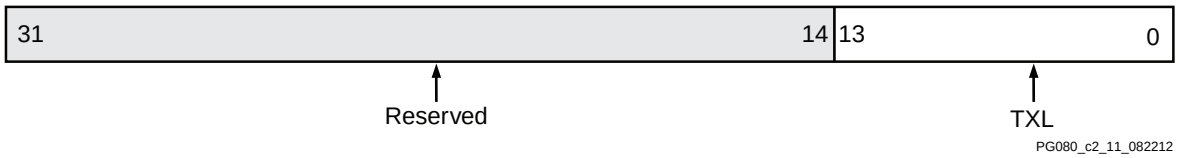

Figure 2-11: **Transmit Length Register (offset 0x14)**

Table 2-12: **Transmit Length Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
31-14	Reserved	N/A	0x0	**Reserved**: These bits are reserved for future definition and will always return all zeros.
13-0	TXL	Write	0x0	**Transmit Length**: The number of bytes of the corresponding transmit packet stored in the transmit data FIFO.

Receive Length Register (RLR)

The receive length register shown in Figure 2-12 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 length is written by the AXI4-Stream FIFO core when the packet is received across the RX AXI4-Stream interface and the receive data FIFO has the adequate number of locations to ensure that all of the packet data has been saved.

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

The RLR values are stored in the receive data FIFO by the AXI4-Stream FIFO core with the data of each packet. The RLR value for the subsequent packet to be processed is moved to the RLR when the previous RLR value has been read.

This register is wide enough to support packets up to 32 Kbytes in length. The smallest packet that may be received is four 32-bit words (include partial final word) which

corresponds to 13 to 16 bytes. The maximum packet that may be received is limited by the size of the FIFO and is C_RX_FIFO_DEPTH-4 words.

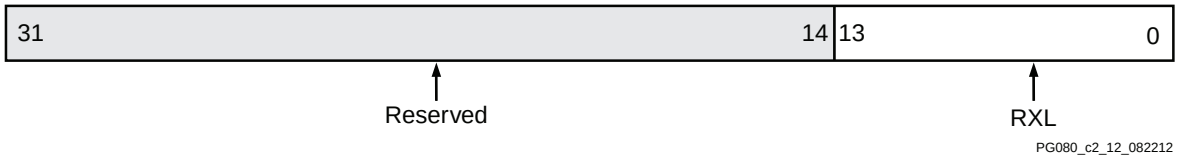

Figure 2-12: **Receive Length Register (offset 0x24)**

Table 2-13: **Receive Length Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
31-14	Reserved	Read	0x0	**Reserved**: These bits are reserved for future definition and will always return all zeros.
13-0	RXL	Read	0x0	**Receive Length**: The number of bytes of the corresponding receive data stored in the receive data FIFO.

AXI4-Stream Reset Register (SRR)

The AXI4-Stream Register shown in Figure 2-13 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.

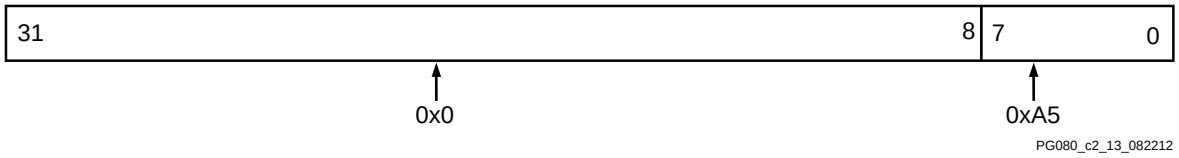

Figure 2-13: **AXI4-Stream Reset Register (offset 0x28)**

Table 2-14: **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 2-14 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.

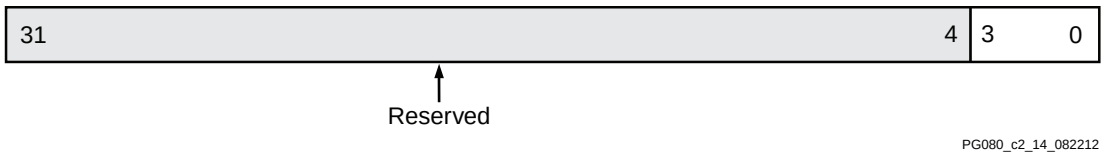

Figure 2-14: **Transmit Destination Register (offset 0x2C)**

Table 2-15: **Transmit Destination Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
31-4	Reserved	N/A	0x0	**Reserved**: These bits are reserved for future definition and will always return all zeros.
3-0	TDEST	Write	0x0	**Transmit Destination**: The destination address of the transmit packet stored in the transmit data FIFO.

Receive Destination Register (RDR)

The Receive Destination Register shown in Figure 2-15 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). Once 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.

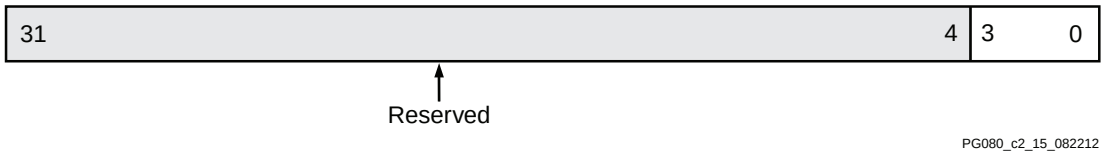

Figure 2-15: **Receive Destination Register (offset 0x30)**

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

Bit(s)	Name	Core Access	Reset Value	Description
31-4	Reserved	N/A	0x0	**Reserved**: These bits are reserved for future definition and will always return all zeros.
3-0	RDEST	Read	0x0	**Receive Destination**: The destination address of the receive packet stored in the receive data FIFO.

Reserved Registers

Reading from reserved registers will return 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

Figure 3-1 shows the AXI4-Stream FIFO core connected to the AXI Ethernet core.

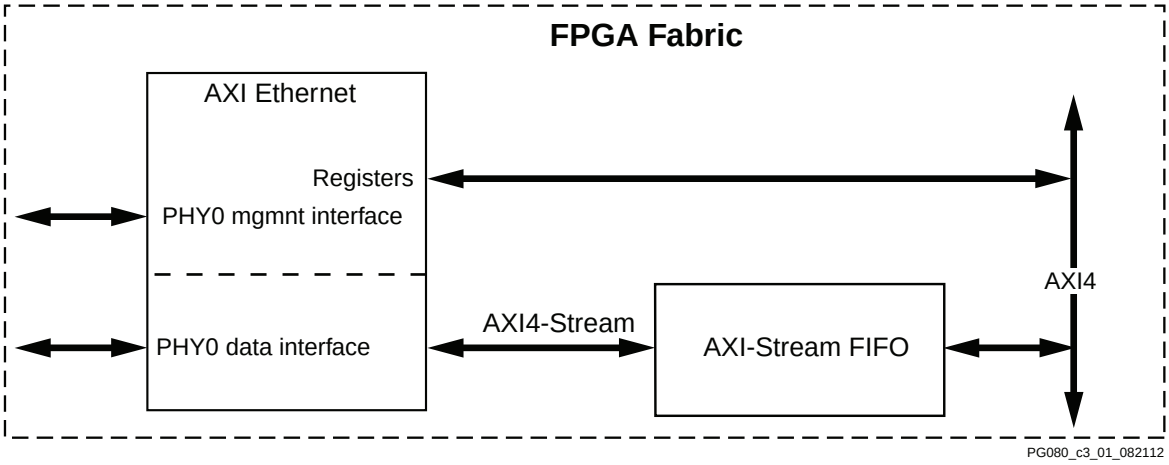

Figure 3-1: **AXI4-Stream FIFO Connected to an AXI Ethernet Core**

Figure 3-2 shows a partial code segment from an EDK MHS file. The file shows an example connection between AXI4-Stream FIFO cores and an AXI Ethernet core.

```
BEGIN axi_fifo_mm_s
 PARAMETER INSTANCE = ETHERNET_fifo
 PARAMETER HW_VER = 3.00.a
 PARAMETER C_INTERCONNECT_S_AXI_MASTERS = microblaze_0.M_AXI_DP
 PARAMETER C_BASEADDR = 0x71a00000
 PARAMETER C_HIGHADDR = 0x71a0ffff
 BUS_INTERFACE S_AXI =         axi_interconnect_memory_mapped_lite_0
 BUS_INTERFACE AXI_STR_RXD = ETHERNET_fifo_rxd
 BUS_INTERFACE AXI_STR_TXD = ETHERNET_fifo_txd
 BUS_INTERFACE AXI_STR_TXC = ETHERNET_fifo_txc
 PORT S_AXI_ACLK = clk_100_0000MHzMMCM0
 PORT INTERRUPT = axi_fifo_INTERRUPT
 PORT AXI_STR_TXD_ACLK = clk_100_0000MHzMMCM0
 PORT AXI_STR_TXC_ACLK = clk_100_0000MHzMMCM0
 PORT AXI_STR_RXD_ACLK = clk_100_0000MHzMMCM0
END

BEGIN axi_ethernet
…
 BUS_INTERFACE S_AXI = axi_interconnect_memory_mapped_lite_0
 BUS_INTERFACE AXI_STR_RXD = ETHERNET_fifo_rxd
 BUS_INTERFACE AXI_STR_TXD = ETHERNET_fifo_txd
 BUS_INTERFACE AXI_STR_TXC = ETHERNET_fifo_txc
 …
 PORT PHY_RST_N = ETHERNET_PHY_RST_N
 PORT GTX_CLK = clk_125_0000MHz
 PORT INTERRUPT = ETHERNET_INTERRUPT
 PORT AXI_STR_RXS_TREADY = net_vcc
 PORT AXI_STR_TXC_ACLK = clk_100_0000MHzMMCM0
 PORT AXI_STR_TXD_ACLK = clk_100_0000MHzMMCM0
 PORT AXI_STR_RXS_ACLK = clk_100_0000MHzMMCM0
 PORT AXI_STR_RXD_ACLK = clk_100_0000MHzMMCM0
END
```

PG080_c3_02_082112

Figure 3-2: **EDK MHS File Code Segment**

Design Tools

The AXI4-Stream FIFO core design is implemented using VHDL code. Xilinx XST and the Vivado Design Suite are the synthesis tools used 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 3-3 details the AXI4-Stream interface where INFORMATION represents all AXI4-Stream signals except TVALID/TREADY.

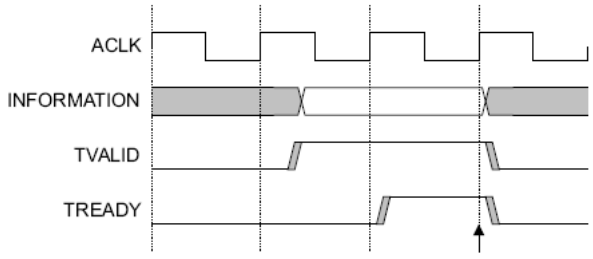

Figure 3-3: **AXI4-Stream Interface Timing Diagram**

Figure 3-4 details the AXI4 Write burst transaction, and Figure 3-5 details the AXI4 Read burst transaction.

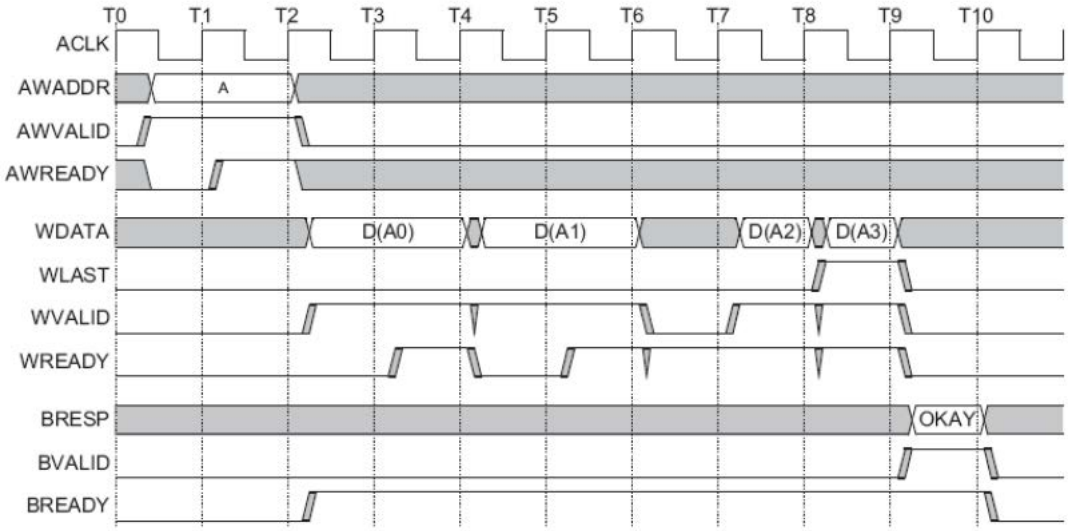

Figure 3-4: **AXI4 Write Burst Transaction**

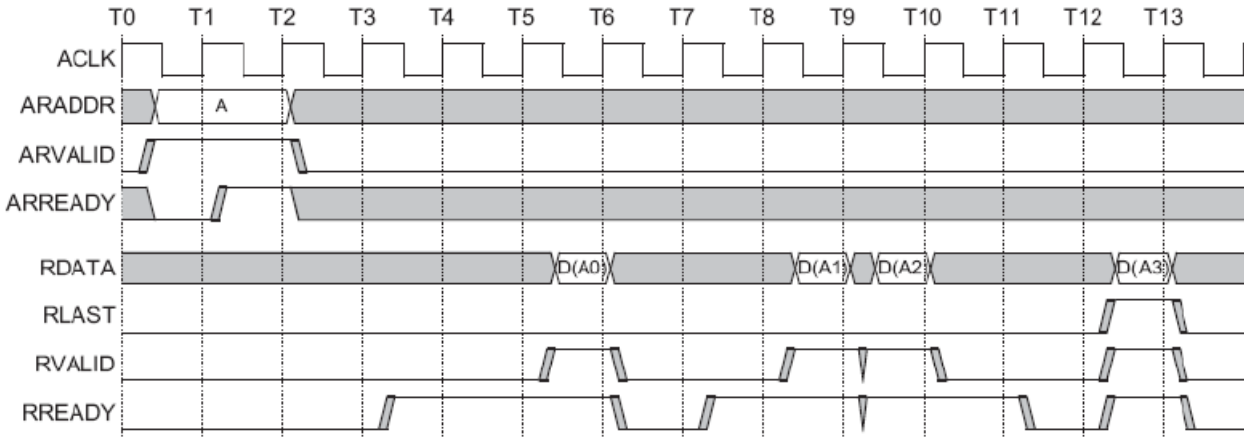

Figure 3-5: **AXI4 Read Burst Transaction**

SECTION II: VIVADO DESIGN SUITE

Customizing and Generating the Core

Constraining the Core

Customizing and Generating the Core

This chapter includes information about using Xilinx tools to customize and generate the core in the Vivado™ Design Suite environment.

GUI

The AXI4-Stream FIFO core is located under AXI Infrastructure in the Vivado IP catalog.

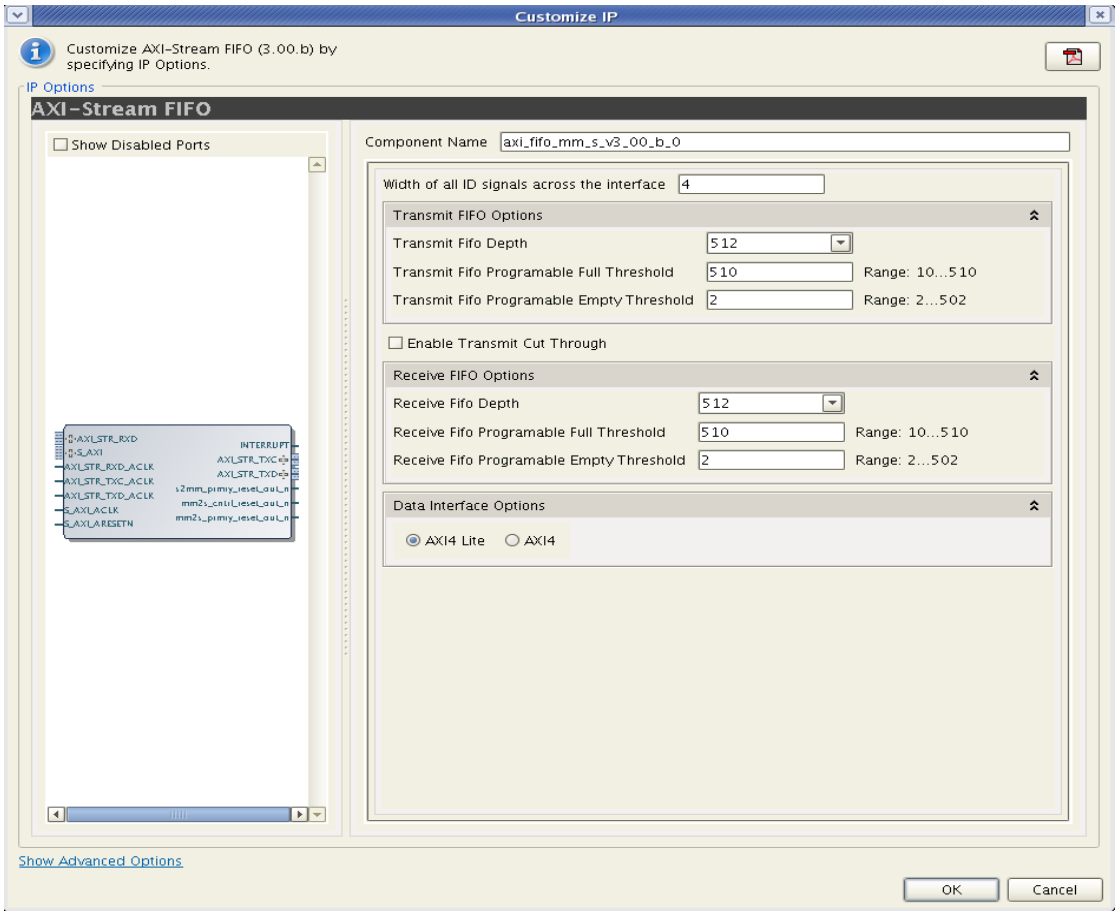

Figure 4-1: **Vivado GUI for AXI4-Stream FIFO**

Output Generation

The output files generated by the Vivado IP catalog are placed in the <project_directory> top-level directory. Depending on the settings, the file output list may include some or all of the following files:

- **<project_directory>/<project_name.data>**
  Contains constraints and file set details
- **<project directory>/<project_name>.src/sources_1/ip/ <component name>**
  Contains the source files, (XCI, XDC, TCL and document files)
  - <component name>/synth
    Contains the source file necessary to synthesize the AXI4-Stream FIFO
  - <component name>/sim
    Contains the source file necessary to simulate the AXI4-Stream FIFO

<project directory>/<project_name>.src/sources_1/ip/ <component name>

This directory contains templates for instantiation of the core, synth, XML and the XCI files.

Table 4-1: **Component Name Directory**

Name	Description
<component_name>.xci	Log file from VIVADO software describing which options were used to generate the AXI4-Stream FIFO core. An XCI file can also be used as an input to the Vivado Design Suite.
<component_name>.{veo|vho}	VHDL or Verilog instantiation template.

<component name>/synth

The synth directory contains the files for synthesizing the core.

Table 4-2: **Synth Directory**

Name	Description
<component_name>.vhd	A VHDL file from the Vivado Design Suite to synthesize the AXI4-Stream FIFO core.

<component name>/sim

The sim directory contains the files for simulating the core.

Table 4-3: **Synth Directory**

Name	Description
<component_name>.vhd	A VHDL file from the Vivado Design Suite to simulate the AXI4-Stream FIFO core.

Constraining the Core

This chapter contains information about constraining the core in the Vivado™ Design Suite environment.

Required Constraints

The AXI4-Stream FIFO does not require any specific constraint other than a clock constraint that is set at the system level.

Device, Package, and Speed Grade Selections

See IP Facts for details about device support.

Clock Frequencies

When targeting a Virtex-7 devices, this core can operate up to 300 MHz with the maximum possible configuration.

Clock Management

The AXI4-Stream FIFO uses a single clock and reset. The core does not require any additional clock management.

Clock Placement

The AXI4-Stream FIFO core does not have any clock placement constraints.

Banking

There are no banking constraints.

Transceiver Placement

There are no transceiver constraints.

I/O Standard and Placement

There are no I/O constraints.

SECTION III: ISE DESIGN SUITE

Customizing and Generating the Core

Constraining the Core

Customizing and Generating the Core

This chapter includes information about using Xilinx tools to customize and generate the core in the ISE® Design Suite environment.

GUI

The CORE Generator GUI for customizing the core is shown in Figure 6-1.

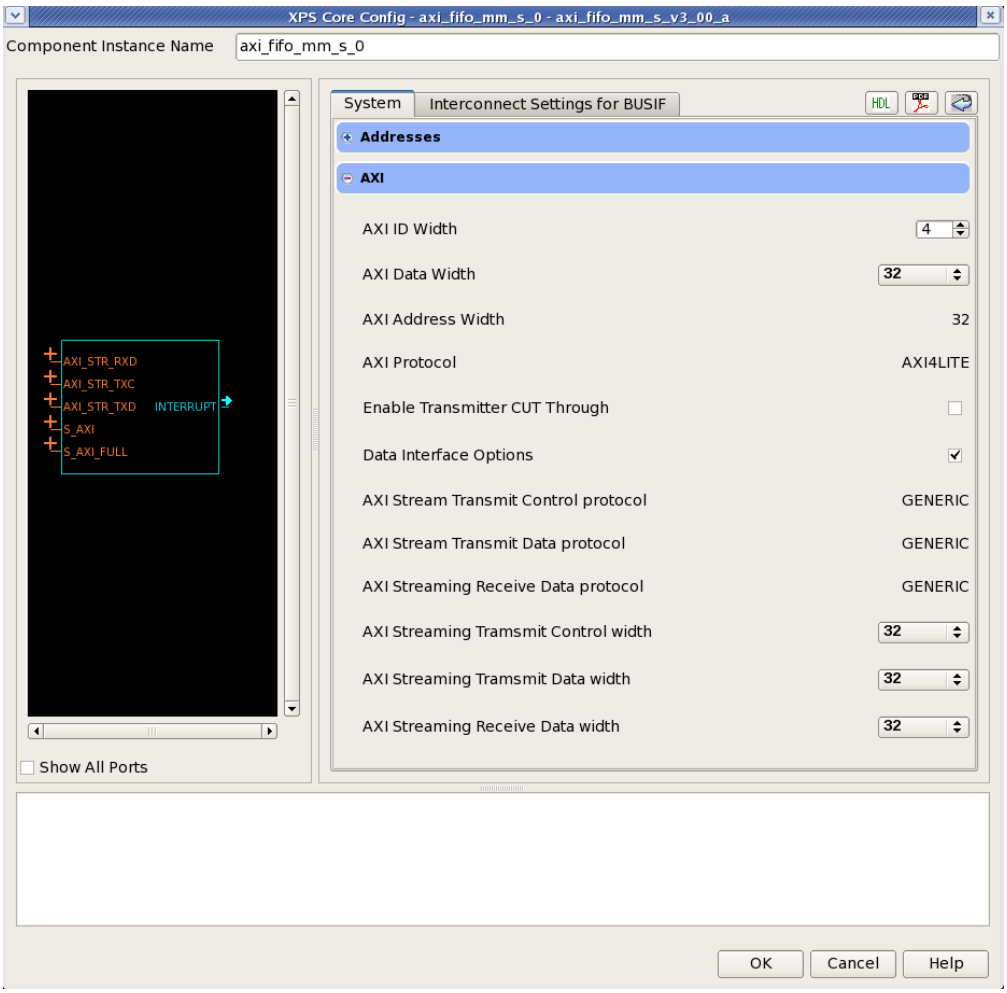

Figure 6-1: **XPS Customization Screen**

Parameter Values in the XCO File

To allow the user to generate a AXI4-Stream FIFO core that is tailored for their system, certain features are parameterizable, thereby resulting in a design that utilizes only the resources required by their system and runs at the best possible performance. The features that are parameterizable in the AXI4-Stream FIFO core design are shown in Table 6-1.

Inferred Parameters

In addition to the parameters listed in Table 6-1, there are parameters that are inferred for each AXI interface in the EDK tools. Through the design, these EDK-inferred parameters control the behavior of the AXI Interconnect. For a complete list of the interconnect settings related to the AXI interface, see DS768, *AXI Interconnect Specification Data Sheet*.

Table 6-1: **Design Parameters**

Feature/Description	Parameter Name	Allowable Values	Default Values	VHDL Type
Device Family	C_FAMILY	zynq7000, artix7, virtex7, kintex7, virtex6, spartan6	virtex7	string
Width of All ID Signals Across The Interface: This is auto computed by the tools.	C_S_AXI_ID_WIDTH	Integer	4	Integer
Address Width of the Interface: Must be 32.	C_S_AXI_ADDR_WIDTH	32	32	Integer
Data Width of the Interface	C_S_AXI_DATA_WIDTH	32	32	Integer
	C_S_AXI4_DATA_WIDTH	32, 64	32	Integer
Base Address of the IP(1)	C_BASEADDR	0x0 - 0xfffff000	0xfffff000	Bit32
High Address of the IP(1)	C_HIGHADDR	0x00000fff - 0xffffffff	0x00000fff	Bit32
Base Address of the IP(1)	C_AXI4_BASEADDR(2)	0x0 - 0xfffff000	0xfffff000	Bit32
High Address of the IP(1)	C_AXI4_HIGHADDR(2)	0x00000fff - 0xffffffff	0x00000fff	Bit32
TX FIFO Depth: FIFO size is C_TX_FIFO_DEPTH* C_S_AXI_DATA_WIDTH/8.	C_TX_FIFO_DEPTH	512, 1024, 2048, 4096	512	Integer
RX FIFO Depth: FIFO size is C_RX_FIFO_DEPTH* C_S_AXI_DATA_WIDTH/8.	C_RX_FIFO_DEPTH	512, 1024, 2048, 4096	512	Integer
TX FIFO Programmable FULL Threshold Value	C_TX_FIFO_PF_THRESHOLD	10 - C_TX_FIFO_DEPTH-2	510	Integer

Table 6-1: **Design Parameters *(Cont'd)***

Feature/Description	Parameter Name	Allowable Values	Default Values	VHDL Type
TX FIFO Programmable EMPTY Threshold Value	C_TX_FIFO_PE_THRESHOLD	2 - C_TX_FIFO_ DEPTH-10	2	Integer
RX FIFO Programmable FULL Threshold Value	C_RX_FIFO_PF_THRESHOLD	10 - C_RX_FIFO_ DEPTH-2	510	Integer
RX FIFO Programmable EMPTY Threshold Value	C_TX_FIFO_PE_THRESHOLD	2 - C_TX_FIFO_ DEPTH-10	2	Integer
TX Cut-Through Mode (Non-packet mode)	C_USE_TX_CUT_THROUGH	0, 1	0	Integer
AXI4 Data Interface Mode	C_DATA_INTERFACE_TYPE	0, 1	0	Integer

Notes:

1. HIGHADDR is required to be at least BASEADDR + 4095 because AXI requires a core to have a minimum 4k address space. For example: C_BASEADDR = 0x00000000 and C_HIGHADDR = 0x00000FFF. The range specified by BASEADDR and HIGHADDR must be a power of 2 in size, and must have BASEADDR aligned to the size.
2. The values of C_AXI4_BASEADDR and C_AXI4_HIGHADDR should be different from C_BASEADDR and C_HIGHADDR for AXI4 Data Interface mode (C_DATA_INTERFACE_TYPE = 1).

Constraining the Core

This chapter contains information about constraining the core in the ISE® Design Suite environment.

Required Constraints

The AXI4-Stream FIFO does not require any specific constraint other than a clock constraint that is set at the system level.

Device, Package, and Speed Grade Selections

See IP Facts for details about device support.

Clock Frequencies

When targeting a Virtex-7 devices, this core can operate up to 300 MHz with the maximum possible configuration.

Clock Management

The AXI4-Stream FIFO uses a single clock and reset. The core does not require any additional clock management.

Clock Placement

The AXI4-Stream FIFO core does not have any clock placement constraints.

Banking

There are no banking constraints.

Transceiver Placement

There are no transceiver constraints.

I/O Standard and Placement

There are no I/O constraints.

SECTION IV: APPENDICES

Verification, Compliance, and Interoperability

Additional Resources

Appendix A

Verification, Compliance, and Interoperability

This appendix includes information about how the IP was tested for compliance with the protocol to which it was designed.

Simulation

The AXI4-Stream FIFO has been tested with Xilinx ISIM, and Mentor Graphics ModelSim 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 Ethernet IP.

Appendix B

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.

The following topics are included in this appendix:

Finding Help on Xilinx.com

To help in the design and debug process when using the AXI4-Stream FIFO core, the Xilinx Support web page (www.xilinx.com/support) contains key resources such as product documentation, release notes, answer records, information about known issues, and links for opening a Technical Support Web Case.

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 (www.xilinx.com/support) or by using the Xilinx Documentation Navigator.

Download the Xilinx Documentation Navigator from the Design Tools tab on the Downloads page (www.xilinx.com/download). For more information about this tool and the features available, open the online help after installation.

Release Notes

Known issues for all cores, including the AXI4-Stream FIFO core are described in the IP Release Notes Guide (XTP025).

Known Issues

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 are listed below, and can also 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.

Contacting Technical Support

Xilinx provides premier technical support for customers encountering issues that require additional assistance.

To contact Xilinx Technical Support:

1. Navigate to www.xilinx.com/support.
2. Open a WebCase by selecting the WebCase link located under Support Quick Links.

When opening a WebCase, include:

- Target FPGA including package and speed grade.
- All applicable Xilinx Design Tools and simulator software versions.
- Additional files based on the specific issue might also be required. See the relevant sections in this debug guide for guidelines about which file(s) to include with the WebCase.

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.

ChipScope Pro Tool

The ChipScope™ Pro tool inserts logic analyzer, bus analyzer, and virtual I/O cores directly into your design. The ChipScope Pro tool allows you to set trigger conditions to capture application and integrated block port signals in hardware. Captured signals can then be analyzed through the ChipScope Pro Logic Analyzer tool. For detailed information for using the ChipScope Pro tool, see www.xilinx.com/tools/cspro.htm.

Reference Boards

Various Xilinx development boards support AXI4-Stream FIFO core. These boards can be used to prototype designs and establish that the core can communicate with the system.

- 7 series evaluation boards
  - KC705
  - KC724

License Checkers

If the IP requires a license key, the key must be verified. The ISE and Vivado tool flows have a number of license check points for gating licensed IP through the flow. If the license check succeeds, the IP may continue generation; otherwise generation halts with an error. License checkpoints are enforced by the following tools:

- ISE flow: XST, NgdBuild, Bitgen
- Vivado flow: Vivado Synthesis, Vivado Implementation, write_bitstream (Tcl command)

IMPORTANT: *IP license level is ignored at checkpoints. The test confirms a valid license exists. It does not check IP license level.*

Simulation Debug

This section provides debug steps for common issues seen in simulation. The following debug information can be used with any simulator.

- If the Transmit Complete interrupt not getting asserted for the AXI4-Stream Transmit Data Interface, then:
  - Check whether AXI_STR_TXD_TREADY is getting asserted from the test bench.
  - Check TPOE and TSE interrupt status register values.
- If the Receive Complete interrupt is not occurring or if the RDFO register value is zero for AXI4-Stream Receive Data Interface, check whether the complete packet has been written to the AXI4-Stream Receive Data side. Receive complete interrupts occur only after a complete packet reception has occurred inside the core.
- If data transfer not working on the AXI4 interface, check the transaction base address for the TDFD and RDFD registers. The base address for the AXI4 interface is different from the AXI4-Lite interface.
- If soft resets are not getting applied, check whether the packet transmission is complete on AXI4-Stream Transmit Data and AXI4-Stream Receive Data interfaces. For a soft reset, core interfaces will not get reset during a packet transfer.
- If AXI_STR_TXD_TVALID is getting de-asserted before the end of a complete packet transfer on the TX side, check if the core is in cut-through mode using GUI parameter C_USE_TX_CUT_THROUGH. In cut through mode, the TX side works in non-packet mode.

Hardware Debug

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

General Checks

Ensure that all the timing constraints were met during implementation.

- Does it work in post-place and route timing simulation? If problems are seen in hardware but not in timing simulation, this could indicate a PCB issue. Ensure that all clock sources are active and clean.
- If using MMCMs in the design, ensure that all MMCMs have obtained lock by monitoring the LOCKED port.
- If your outputs go to 0, check your licensing.
- If the interrupts do not occur, check if the interrupts are enabled.
- If the design is unresponsive, check the interrupt status register and ensure the interrupt is serviced.

Interface Debug

AXI4-Lite Interfaces

Read from a register that does not have all 0s as a default to verify that the interface is functional. Figure B-1 shows an AXI4-Lite read timing diagram.

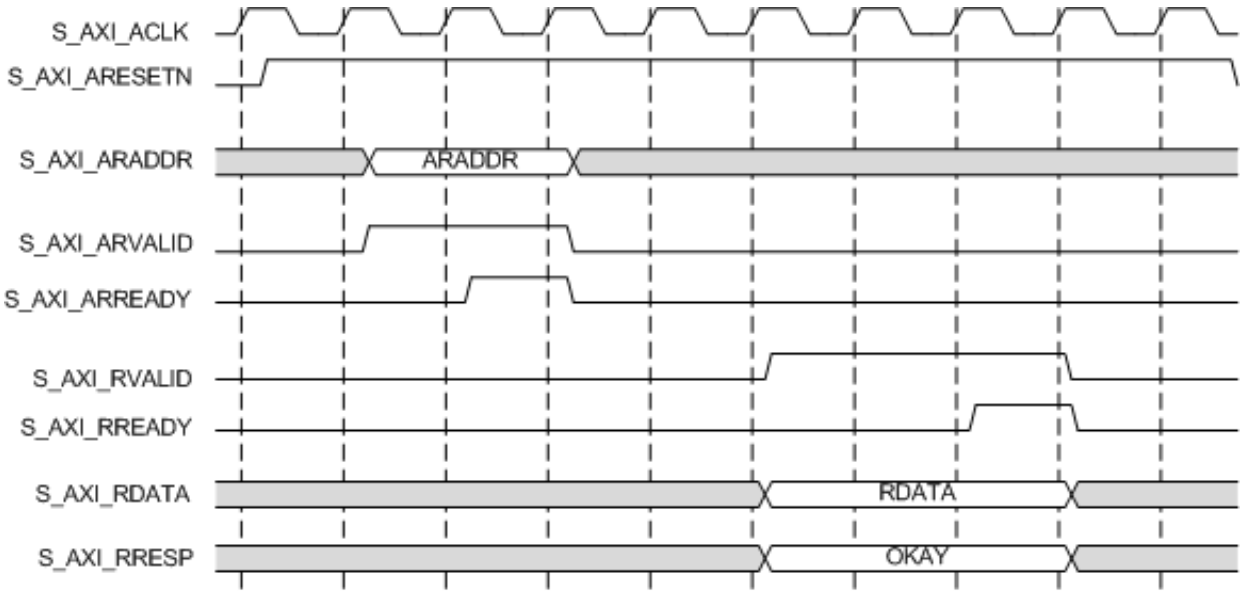

Figure B-1: **AXI4-Lite Read**

Output S_AXI_ARREADY asserts when the core is ready to accept the read address, and output S_AXI_RVALID asserts when the read data/response is valid. If the interface is unresponsive, ensure that the following conditions are met:

- The S_AXI_ACLK input is connected and toggling.
- The interface is not being held in reset, and S_AXI_ARESETN is an active-Low reset.
- If the simulation has been run, verify in simulation and/or a ChipScope tool capture that the waveform is correct for accessing the AXI4-Lite interface.

AXI4-Stream Interfaces

If data is not being transmitted or received, check the following conditions:

- If transmit AXI_STR_TXD_TVALID is stuck low following the AXI_STR_TXD_TREADY input being asserted, the core cannot send data.
- If the receive AXI_STR_RXD_TREADY is stuck low, the core is not receiving data.
- Check that the S_AXI_ACLK inputs are connected and toggling.

- Check that the AXI4-Stream waveforms are being followed. See Figure B-2.

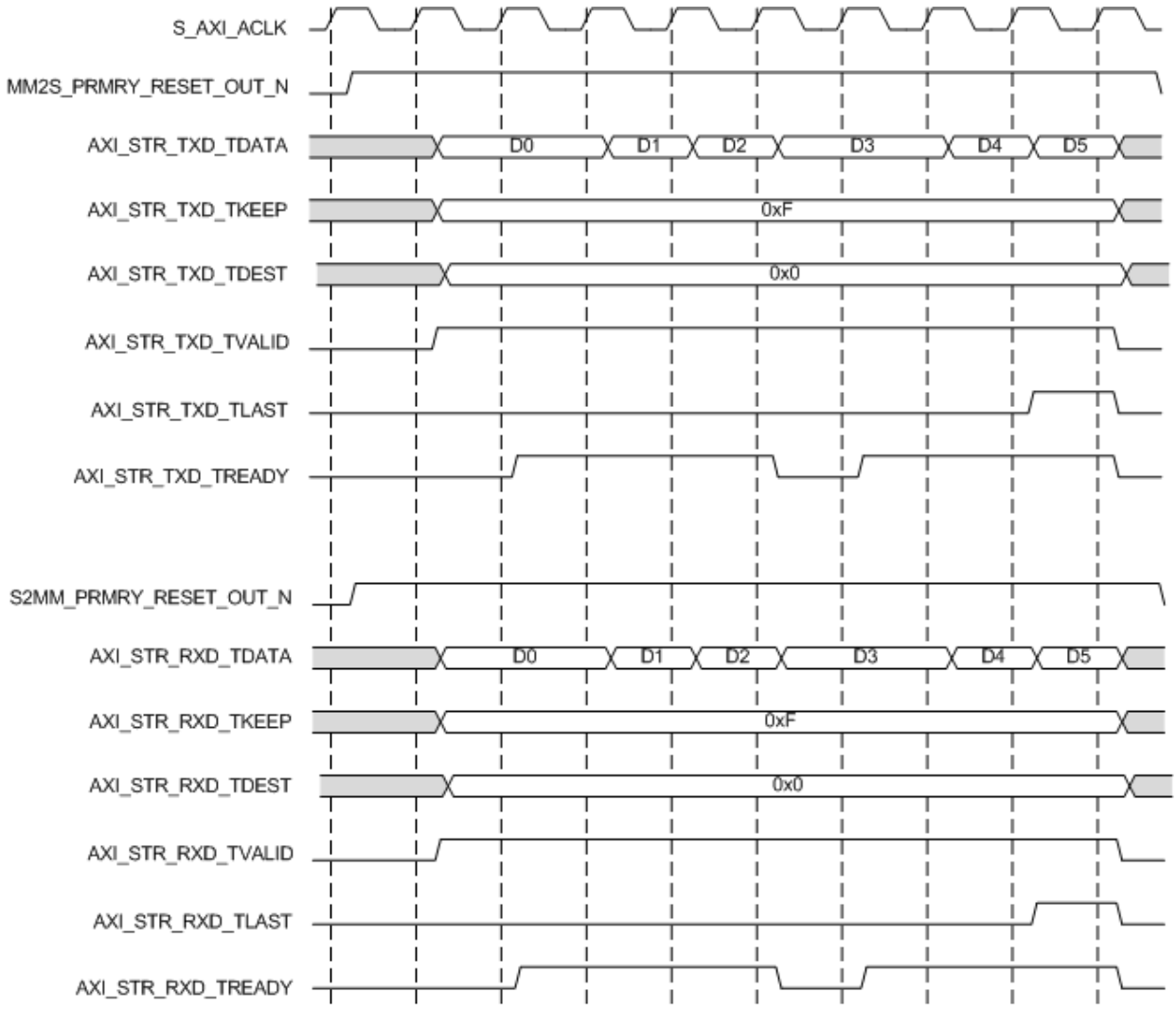

Figure B-2: **AXI4-Stream Transmit**

Appendix C

Additional Resources

Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see the Xilinx Support website at:

www.xilinx.com/support.

For a glossary of technical terms used in Xilinx documentation, see:

www.xilinx.com/company/terms.htm.

References

These documents provide supplemental material useful with this product guide:

1. *AMBA AXI4-Stream Protocol Specification*
2. *AMBA AXI4 Protocol Specification*
3. http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ihi0051a/index.html
4. DS769, *LogiCORE IP AXI Slave Burst*
5. DS160, *Spartan-6 Family Overview*
6. DS150, *Virtex-6 Family Overview*
7. DS180, *7 Series FPGAs Overview*
8. Vivado™ Design Suite user documentation

Technical Support

Xilinx provides technical support at www.xilinx.com/support for this LogiCORE™ IP product when used as described in the product documentation. Xilinx cannot guarantee timing,

functionality, or support of product if implemented in devices that are not defined in the documentation, if customized beyond that allowed in the product documentation, or if changes are made to any section of the design labeled DO NOT MODIFY.

See the Embedded Edition Derivative Device Support web page (www.xilinx.com/ise/embedded/ddsupport.htm) for a complete list of supported derivative devices for this core.

See the IP Release Notes Guide (XTP025) for more information on this core. For each core, there is a master Answer Record that contains the Release Notes and Known Issues list for the core being used. The following information is listed for each version of the core:

- New Features
- Resolved Issues
- Known Issues

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
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.
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 29 and the Receive Length Register (RLR), page 30. • Clarified which ports are not used by the core in Table 2-4.

Notice of Disclaimer

fail-safe performance; you assume sole risk and liability for use of Xilinx products in Critical Applications: http://www.xilinx.com/warranty.htm#critapps.