

AXI Virtual FIFO Controller v2.0

LogiCORE IP Product Guide

PG038 November 18, 2015

Table of Contents

IP Facts

Chapter 1: Overview

AXI4-Stream Interface Controller	6
Per-Channel S2MM/MM2S DRAM Address	6
Memory Mapped Interface Controller	6
Per-Channel Flag Generation	7
MM2S Weighted Round Robin Arbiter	7
Applications	8
Unsupported Features	9
Licensing and Ordering	9

Chapter 2: Product Specification

Standards Compliance	10
Performance	10
Resource Utilization	11
I/O Signals	12

Chapter 3: Designing with the Core

General Design Guidelines	24
Clocking	24
Resets	25
Protocol Description	25

Chapter 4: Customizing and Generating the Core

Vivado Integrated Design Environment (IDE)	27
Design Parameters	30
Output Generation	33

Chapter 5: Constraining the Core

Chapter 6: Simulation

Chapter 7: Synthesis and Implementation

Chapter 8: Detailed Example Design

Appendix A: Verification, Compliance, and Interoperability

Simulation	38
Hardware Testing	38

Appendix B: Migrating and Upgrading

Migrating to the Vivado Design Suite	39
Upgrading in the Vivado Design Suite	39

Appendix C: Debugging

Finding Help on Xilinx.com	40
Debug Tools	41

Appendix D: Additional Resources

Xilinx Resources	42
References	42
Revision History	43
Notice of Disclaimer	43

Introduction

The Xilinx LogiCORE™ IP AXI Virtual FIFO Controller core (VFIFO) is a high performance core that implements multiple AXI4-Stream FIFOs. The memory storage for data contained in the FIFOs comes from an attached AXI4 slave memory controller. The VFIFO core manages multiple sets of read and write address pointers to emulate the behavior of multiple independent FIFOs. An AXI4 slave memory controller with external memory can provide large depths of external SRAM or DDR memory. VFIFO is useful in applications using PCIe, video, or Ethernet that require FIFOs deeper than can be otherwise constructed from on chip block RAM memory.

Features

- Supports up to eight channels
- Supports single clock operations
- Supports DRAM address management
- Configurable DRAM base address
- Supports AXI4-Stream and AXI4 Memory Mapped (MM) interfaces
- Supports 32-to-1024 AXI4-Stream data width
- AXI MM data width same as AXI4-Stream Data width
- Supports configurable 512-4096 bytes AXI MM bursts
- Generates Per-Channel FIFO Full signals
- Generates Per-Channel FIFO Empty signals
- Supports MM2S Weighted Round Robin (WRR) Arbitration
- Provides AXI MM response error interrupts

LogiCORE IP Facts Table	
Core Specifics	
Supported Device Family ⁽¹⁾	UltraScale+™ Families, UltraScale™ Architecture, Zynq®-7000, 7 Series
Supported User Interfaces	AXI4 and AXI4-Stream
Resources	See Table 2-3 .
Provided with Core	
Design Files	Encrypted RTL
Example Design	VHDL
Test Bench	VHDL
Constraints File	XDC
Simulation Model	Verilog and/or Source HDL
Supported S/W Driver	N/A
Tested Design Flows⁽²⁾	
Design Entry	Vivado® Design Suite
Simulation	For supported simulators, see the Xilinx Design Tools: Release Notes Guide .
Synthesis	Vivado Synthesis
Support	
Provided by Xilinx at the Xilinx Support web page	

Notes:

1. For a complete list of supported devices, see Vivado IP catalog.
2. For the supported versions of the tools, see the [Xilinx Design Tools: Release Notes Guide](#).

Overview

AXI Virtual FIFO controller supports a true multi-channel architecture allowing per-clock-cycle channel arbitration. The AXI4-Stream t_{dest} signal provided to the AXI4-Stream slave interface is used as a channel identifier. t_{dest} can be kept unique for each packet allowing packet switching operations. Figure 1-1 shows the block diagram for AXI Virtual FIFO Controller (VFIFO) IP.

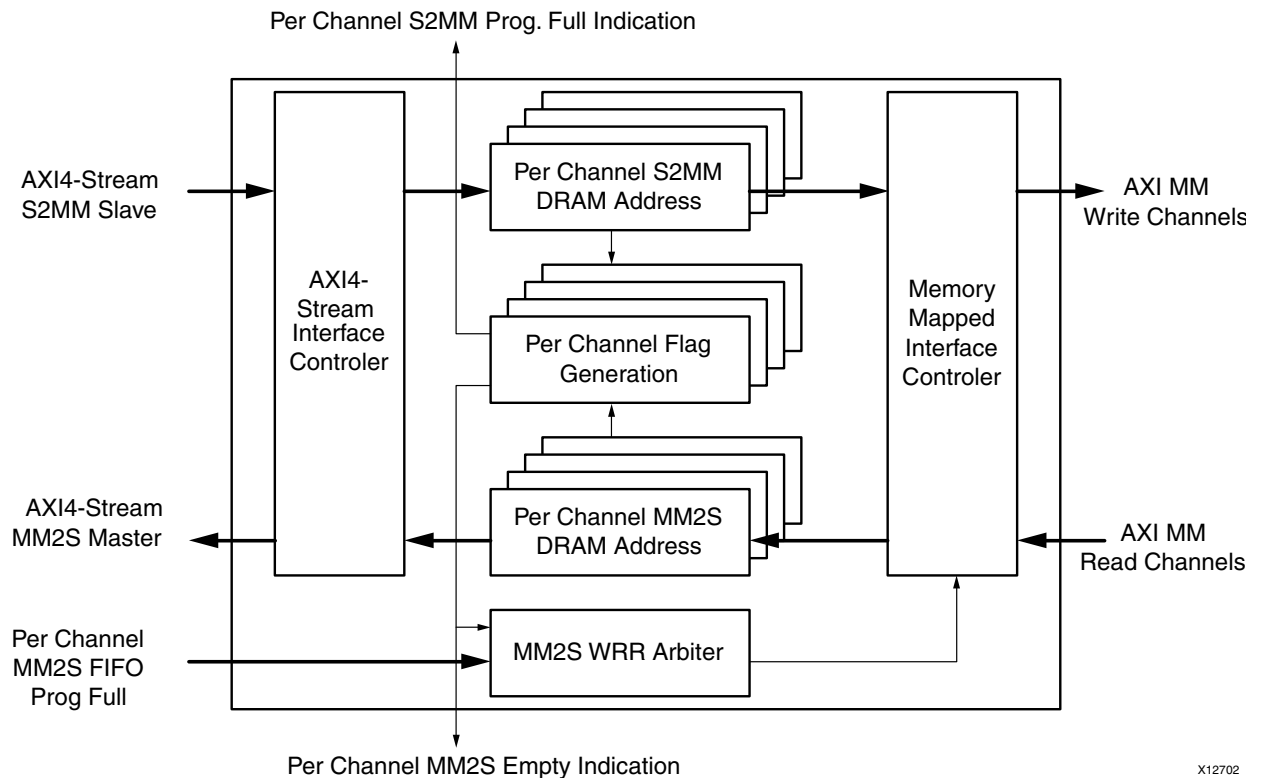


Figure 1-1: AXI Virtual FIFO Controller Block Diagram

AXI Virtual FIFO operations can be broadly categorized into four modules: AXI4-Stream Interface Controller, Per-Channel S2MM/MM2S DRAM Address, Memory Mapped Interface Controller and MM2S Weighted Round Robin Arbiter. This chapter describes these modules.

AXI4-Stream Interface Controller

This module handles the AXI4-Stream interface and complies with the AXI4-Stream protocol specification. It provides a Slave Stream Interface to receive data beats or packets from the AXI4-Stream Interconnect IP block, and provides a Master Stream Interface to send data beats or packets to the AXI4-Stream Interconnect. VFIFO is usually used in conjunction with the AXI4-Stream Interconnect to multiplex and demultiplex AXI4-Stream channels to and from the VFIFO controller. AXI4-Stream Interconnect provides buffering, data steering, width conversion, clock conversion, and arbitration features to complement VFIFO when building a system.

- Supports parameterizable data width (32, 64, 128, 256, 512 and 1024)
- Supports up to eight channels
- Supports up to 16-bit optional `tuser` signal (`tuser` is valid only with first beat of packet)
- Supports `tid` Source Stream Identifier (`tid` is valid only with first beat of packet)
- Supports `tlast` as packet delimiter
- Supports `tkeep` to identify packet remainder (`tkeep` is valid only with `tlast`)

Per-Channel S2MM/MM2S DRAM Address

This module handles all requirements for converting AXI MM space into a FIFO. It manages the DRAM addresses and generates per-channel FIFO status signals.

- Supports up to 256MB DDR space
- User specifies DDR memory base address, and the DRAM's start and end addresses are automatically calculated by the core
- Tracks each channel's write and read address pointers to the DDR
- Generates per-channel FIFO full and FIFO empty signals

Memory Mapped Interface Controller

This module handles the AXI Memory Mapped (MM) interface and complies with AXI MM protocol specifications. It provides AXI MM write channels to perform writes to the DDR memory, and provides AXI MM read channels to read data from the DDR memory.

- AXI-MM data width is same as configured AXI4-Stream data width
- Maps to a single MM ID, tied to zero
- Supports programmable burst size of 512, 1024, 2048 and 4096 bytes
- Same burst size for read and write transfers
- Supports only incremental bursts
- Burst size is broken under the following conditions:
 - When MM address crosses 4 K boundary.
 - On determining channel change within the burst
 - On detecting a timeout (`tvalid` deasserted for 256-clocks for the burst)

Per-Channel Flag Generation

AXI Virtual FIFO controller generates per-channel Stream to Memory Mapped (S2MM) FIFO full indication. Full indication is used to apply back pressure on AXI4-Stream Masters. To avoid blocking other channels accessing the VFIFO Controller, ensure the following conditions are met:

- When the VFIFO Controller is full for an active channel (channel which is being used), the external arbiter completes the current packet up to 32 KB or completes the current TDM/burst cycle. The burst cycle for the stream switch should be less than or equal to the programmed burst configured for VFIFO Controller.
- When the VFIFO Controller is full for a new channel, the external arbiter should not select the channel indicating full.

MM2S Weighted Round Robin Arbiter

The VFIFO Controller implements an internal Weighted Round Robin (WRR) Arbiter to transfer data from DDR memory to the per-channel FIFOs in the stream interconnect switch. The Arbiter ensures data is read only when the MM2S Stream FIFO has enough space to accept the data

The MM2S Arbiter initiates the data transfer (AXI AR transactions) from DDR memory to the channel-specific FIFOs in the AXI4-Stream interconnect. AR transactions are initiated only for the channels indicating a programmable amount of space available in the FIFO.

To avoid blocking other channels from accessing the VFIFO Controller, AR weight programming in the VFIFO should always amount to the data transfer less than or equal to

the available space in Stream FIFOs, that is, available space with programmable full deasserted should be always greater than "AR Weight x Burst Size" in the VFIFO Controller.

After initiating AR Transactions, the VFIFO Controller checks for FIFO status only after transferring the configured amount of data to the stream interconnect FIFOs.

The latencies from the VFIFO to the FIFO in the MM2S switch can result in one additional burst transfer (AR Weight x Burst Size) to the stream FIFOs. This space needs to be additionally accounted for in AXI4-Stream interconnect FIFOs.

For more information, see [Per Channel AR Channel Weight Allocation in Chapter 4](#).

Applications

Figure 1-2 shows an outline of the VFIFO use case in which 1 to 8 endpoint IPs send and receive streams of variable-length packets to a VFIFO Controller. The VFIFO Controller is an interface to a DRAM that implements a FIFO access scheme instead of a random access scheme to the memory device. The controller partitions the memory into 1–8 regions (one partition per "virtual" FIFO) and keeps track of the start and end address pointers into the memory when data is read or written from one of the VFIFO partitions. When an endpoint IP writes a packet to the controller, the controller writes that packet data into one of the VFIFO partitions in the DRAM. Similarly, when the controller detects that there is data in one of the Virtual FIFOs, it reads that data and pushes it out to a particular endpoint IP. Each endpoint can write into any one partition and read from one (different) partition.

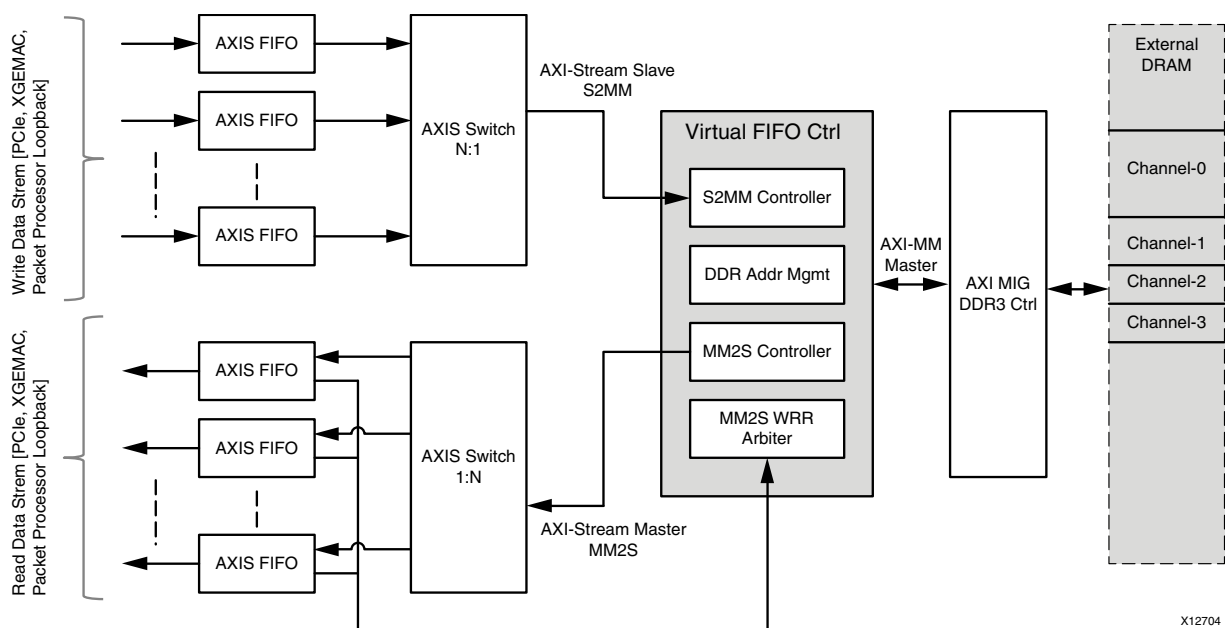


Figure 1-2: Application Block Diagram

X12704

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](#). The core may be accessed through the Vivado Design Suite.

Information about this and other Xilinx LogiCORE IP modules is available at the [Xilinx Intellectual Property](#) page. For information about pricing and availability of other Xilinx LogiCORE modules and software, please contact your [local Xilinx sales representative](#).

Product Specification

This chapter includes details on performance and interfaces.

Standards Compliance

This core complies with the following industry specifications:

- AMBA® AXI4-Stream Protocol Specification
- AMBA AXI4 Protocol Specification

Performance

This section details the performance information for various core configurations.

Maximum Frequencies

Maximum frequencies achieved for various devices and speed grades based on the F_{MAX} margin system are shown in [Table 2-1](#).

Note: Frequencies for UltraScale™ and Zynq®-7000 devices are expected to be similar to frequencies for 7 series devices.

Table 2-1: Maximum Frequencies for Supported Devices

Family	Speed Grade	F_{MAX} (MHz)	
		AXI4	AXI4-Lite
Virtex-7	-1	200	180
Kintex-7		200	180
Artix-7		150	120
Virtex-7	-2	240	200
Kintex-7		240	200
Artix-7		180	140

Table 2-1: Maximum Frequencies for Supported Devices (Cont'd)

Family	Speed Grade	F _{MAX} (MHz)	
		AXI4	AXI4-Lite
Virtex-7	-3	280	220
Kintex-7		280	220
Artix-7		200	160

Latency

The VFIFO Controller takes about eight clock cycles to initiate an AW transaction on AXI MM side after receiving a data beat on AXI4-Stream side.

On the MM2S path, the VFIFO Controller takes about 12 clock cycles to initiate the AR transaction on the AXI MM side if a given channel has the data in DDR and `vfifo_mm2s_channel_full[active_channel]` is low. In addition, the VFIFO takes about seven clock cycles to present the received RDATA on AXI4-Stream side.

Table 2-2 shows the data path latencies from Streaming to Memory Map and vice versa.

Table 2-2: Core Latency

Data Path Direction	Latency
Streaming to Memory Map	8
Memory Map to Streaming	7

Throughput

The VFIFO Controller was configured for four channels and was independently tested for a data width of 512 bits and 64 bits. The data width of 512 bits with a 4096 burst size was benchmarked for 80 Gb/s throughput across four channels.

Resource Utilization

Because the AXI Virtual FIFO Controller core is used with other design pieces in an FPGA, the utilization and timing numbers reported in this section are estimates. As the VFIFO core is combined with other pieces of the FPGA design, the utilization of FPGA resources and timing of the VFIFO core design will vary from the results reported here. The core benchmark is shown in Table 2-3. These values were generated using the Vivado Design Suite. They are derived from post-synthesis reports, and might change during MAP and PAR.

Configuration used to obtain the below resource utilization is as follows:

- Number of Channel = 4
- tuser width = 16

Table 2-3: Resource Utilization

Burst Size	Data Width	Block RAMs	DSP48	Slices	Slice Flip-Flops	LUTs
512	32	4	18	621	1665	1170
	64	4	18	723	1938	1169
	128	5	18	845	2485	1173
	256	7	18	1161	3541	1142
1024	32	4	18	616	1644	1167
	64	4	18	603	1917	1168
	128	5	18	850	2464	1172
	256	7	18	951	3520	1175

I/O Signals

Signals for the VFIFO Controller are listed in [Table 2-4](#).

Table 2-4: VFIFO Controller Pin Table

Signal Name	Direction	Width	Default	Description
Global Signals				
aclk	IN	1	'0'	Global Interface Clock: All signals on the interface must be synchronous to aclk.
aresetn	IN	1	'0'	Global reset: This signal is active-Low. aresetn must be asserted at for least for one clock cycle.

Table 2-4: VFIFO Controller Pin Table (Cont'd)

Signal Name	Direction	Width	Default	Description
AXI4-Stream Input Signals From Switch to AXI Virtual FIFO Controller				
s_axis_tvalid	IN	1	'0'	<p>tvalid: Indicates that the master is driving a valid transfer. A transfer takes place when both tvalid and tready are asserted.</p> <p>The VFIFO Controller determines FIFO underrun caused on:</p> <ul style="list-style-type: none"> • Data Transfer is initiated for a Channel (tdest, by asserting s_axis_tvalid) • Without any Channel ID change, s_axis_tvalid is de-asserted for eight clock cycles <p>The above condition will result in a transaction on AXI Memory Mapped interface.</p>
s_axis_tready	OUT	1	'0'	<p>tready: Indicates that the slave can accept a transfer in the current cycle.</p>
s_axis_tdata	IN	DATA_WIDTH	0	<p>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.</p> <p>Supported tdata widths include: 32, 64, 128, 256, 512 and 1024 bits.</p>
s_axis_tstrb	IN	DATA_WIDTH/8	0	<p>tstrb: Byte qualifier that indicates whether the content of the associated byte of tdata is processed as a data byte or a position byte. For a 64-bit DATA, bit 0 corresponds to the least significant byte on DATA, and bit 7 corresponds to the most significant byte.</p> <p>Note: s_axis_tstrb input is not used in Virtual FIFO Controller.</p>

Table 2-4: VFIFO Controller Pin Table (Cont'd)

Signal Name	Direction	Width	Default	Description
s_axis_tkeep	IN	DATA_WIDTH/8	0	<p>tkeep: The byte qualifier that indicates whether the content of the associated byte of tdata is processed as part of the data stream.</p> <p>tkeep is active-High and should be contiguous and LSB aligned. For a 64-bit DATA, bit 0 corresponds to the least significant byte on DATA, and bit 7 corresponds to the most significant byte.</p> <p>tkeep will be ignored when tlast = 0. tkeep will only be valid in a data beat where tlast = 1.</p> <p>Deasserted tkeep bits are contiguous and MSB aligned in the data beat where tlast = 1.</p>
s_axis_tlast	IN	1	'0'	<p>tlast: Indicates the boundary of a packet.</p>
s_axis_tid	IN	LOG2(NUM_OF_Channels)	0	<p>tid: The data stream identifier that indicates different streams of data. tid is a Source AXI4-Stream identifier and can be optional. tid is considered valid with only first beat of a packet.</p>
s_axis_tdest	IN	LOG2(NUM_OF_Channels)	0	<p>tdest: Channel Identifier that provides routing information for the data stream.</p> <p>tdest is a Destination AXI4-Stream identifier and should be kept valid or same for the entire packet duration.</p>
s_axis_tuser	IN	USER_WIDTH	0	<p>tuser: The user-defined sideband information that can be transmitted alongside the data stream.</p> <p>tuser is considered valid only in a first data beat of the Packet and can carry packet control information.</p>
AXI Virtual FIFO Controller Output Signals To AXI4-Stream Switch				
m_axis_tvalid	OUT	1	'0'	<p>tvalid: Indicates that the master is driving a valid transfer.</p> <p>A transfer takes place when both tvalid and tready are asserted.</p>
m_axis_tready	IN	1	'0'	<p>tready: Indicates that the slave can accept a transfer in the current cycle.</p>

Table 2-4: VFIFO Controller Pin Table (Cont'd)

Signal Name	Direction	Width	Default	Description
m_axis_tdata	OUT	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, 512 and 1024 bits
m_axis_tstrb	OUT	DATA_WIDTH/8	1	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. For a 64-bit DATA, bit 0 corresponds to the least significant byte on DATA, and bit 7 corresponds to the most significant byte. Note: m_axis_tstrb output is tied to all 1's and should not be used for any logical operations of interconnecting IPs.
m_axis_tkeep	OUT	DATA_WIDTH/8	0	tkeep: The byte qualifier that indicates whether the content of the associated byte of tdata is processed as part of the data stream. tkeep is active-High and should be contiguous and LSB aligned. For a 64-bit DATA, bit 0 corresponds to the least significant byte on DATA, and bit 7 corresponds to the most significant byte. tkeep will only be valid in a data beat where tlast = 1. Deasserted tkeep bits are contiguous and MSB aligned in the data beat where tlast = 1.
m_axis_tlast	OUT	1	'0'	tlast: Indicates the boundary of a packet.
m_axis_tid	OUT	LOG2(NUM_CHANNEL)	0	tid: The data stream identifier that indicates different streams of data. tid is a Source AXI4-Stream Identifier and can be optional, tid is Valid with only first Beat of Packet and carries tid as provided by Source AXI4-Stream.

Table 2-4: VFIFO Controller Pin Table (Cont'd)

Signal Name	Direction	Width	Default	Description
m_axis_tdest	OUT	LOG2(NUM_CHANNEL)	0	tdest: Channel Identifier and Provides routing information for the data stream. tdest is a Destination AXI4-Stream Identifier and will be kept Valid or same for the entire Packet duration.
m_axis_tuser	OUT	USER_WIDTH	0	tuser: The user-defined sideband information that can be transmitted alongside the data stream tuser will be valid only in a first data beat of the Packet and can carries Packet Control Information as provided by Source AXI4-Stream
Write Address Channel Signals				
m_axi_awid	OUT	LOG2(NUM_CHANNEL)	0	Write Address ID: Identification tag for the write address group of signals. AWID is always set to zero, all configured channels access to a single address MAP in Memory mapped interconnect.
m_axi_awaddr	OUT	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
m_axi_awlen	OUT	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.
m_axi_awsize	OUT	3	0	Burst Size: Indicates the size of each transfer in the burst. Byte lane strobes indicate exactly which byte lanes to update. Burst Size is always set based on configured data width of the interface.
m_axi_awburst	OUT	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 transfers.

Table 2-4: VFIFO Controller Pin Table (Cont'd)

Signal Name	Direction	Width	Default	Description
m_axi_awlock	OUT	1	0	Lock Type: This signal provides additional information about the atomic characteristics of the transfer. awlock is always set to zero.
m_axi_awcache	OUT	4	3	Cache Type: Indicates the bufferable, cacheable, writethrough, write-back, and allocate attributes of the transaction. awcache is always set to zero.
m_axi_awprot	OUT	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. AWPROT is always set to zero.
m_axi_awqos	OUT	4	0	Quality of Service (QoS): Sent on the write address channel for each write transaction. awqos is always set to zero.
m_axi_awregion	OUT	4	0	Region Identifier: Sent on the write address channel for each write transaction. awregion is always set to zero.
m_axi_awuser	OUT	USER_WIDTH	0	Write Address Channel User: AWUSER is always set to zero
m_axi_awvalid	OUT	1	'0'	Write Address Valid: Indicates that valid write address and control information are available: <ul style="list-style-type: none"> • 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.
m_axi_awready	IN	1	'0'	Write Address Ready: Indicates that the slave is ready to accept an address and associated control signals: <ul style="list-style-type: none"> • 1 = Slave ready • 0 = Slave not ready.
Write Data Channel Signals				
m_axi_wdata	OUT	DATA_WIDTH	0	Write Data: The write data bus can be 8, 16, 32, 64, 128, 256, 512 or 1024 bits wide.

Table 2-4: VFIFO Controller Pin Table (Cont'd)

Signal Name	Direction	Width	Default	Description
m_axi_wstrb	OUT	DATA_WIDTH / 8	0	<p>Write Strobes: Indicates which byte lanes to update in memory. There is one write strobe for each eight bits of the write data bus.</p> <p>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:</p> <ul style="list-style-type: none"> • STROBE[0] = 1b, DATA[7:0] is valid • STROBE[7] = 0b, DATA[63:56] is not valid
m_axi_wlast	OUT	1	'0'	Write Last: Indicates the last transfer in a write burst.
m_axi_wuser	OUT	USER_WIDTH	0	Write Data Channel User: wuser is always set to zero
m_axi_wvalid	OUT	1	'0'	<p>Write Valid: Indicates that valid write data and strobes are available:</p> <ul style="list-style-type: none"> • 1 = Write data and strobes available. • 0 = Write data and strobes not available
m_axi_wready	IN	1	'0'	<p>Write Ready: Indicates that the slave can accept the write data:</p> <ul style="list-style-type: none"> • 1 = Slave ready. • 0 = Slave not ready.
Write Response Channel Signals				
m_axi_bid	IN	LOG2(NUM_OF_Channels)	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.
m_axi_bresp	IN	2	0	Write Response: Indicates the status of the write transaction. The allowable responses are OKAY, EXOKAY, SLVERR, and DECERR.
m_axi_buser	IN	USER_WIDTH	0	Write Response Channel User: BUSER is not used in Virtual FIFO controller.
m_axi_bvalid	IN	1	'0'	<p>Write Response Valid: Indicates that a valid write response is available:</p> <ul style="list-style-type: none"> • 1 = Write response available. • 0 = Write response not available.

Table 2-4: VFIFO Controller Pin Table (Cont'd)

Signal Name	Direction	Width	Default	Description
m_axi_bready	OUT	1	'1'	Response Ready: Indicates that the master can accept the response information. <ul style="list-style-type: none"> • 1 = Master ready. • 0 = Master not ready.
Read Address Channel Signals				
m_axi_arid	OUT	LOG2(NUM_CHANNEL)	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 Map interconnect.
m_axi_araddr	OUT	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. The control signals that are issued alongside the address detail how the address is calculated for the remaining transfers in the burst.
m_axi_arlen	OUT	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.
m_axi_arsize	OUT	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.
m_axi_arburst	OUT	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.
m_axi_arlock	OUT	1	0	Lock Type: This signal provides additional information about the atomic characteristics of the transfer. Lock type is always set to zero.
m_axi_arcache	OUT	4	3	Cache Type: This signal provides additional information about the cacheable characteristics of the transfer. Cache Type is always set to zero.

Table 2-4: VFIFO Controller Pin Table (Cont'd)

Signal Name	Direction	Width	Default	Description
m_axi_arprot	OUT	3	0	Protection Type: This signal provides protection unit information for the transaction. Protection Type is always set to zero.
m_axi_arqos	OUT	4	0	Quality of Service (QoS): Sent on the read address channel for each read transaction. Quality of service is always set to zero.
m_axi_arregion	OUT	4	0	Region Identifier: Sent on the read address channel for each read transaction. Region Identifier is always set to zero.
m_axi_aruser	OUT	USER_WIDTH	0	Read Address Channel User: User is always set to zero.
m_axi_arvalid	OUT	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. <ul style="list-style-type: none"> • 1 = Address and control information valid. • 0=Address and control information not valid.
m_axi_arready	IN	1	'0'	Read Address Ready: Indicates that the slave is ready to accept an address and associated control signals: <ul style="list-style-type: none"> • 1 = Slave ready. • 0 = Slave not ready.
Read Data Channel Signals				
m_axi_rid	IN	LOG2(NUM_CHANNEL)	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. RID is not used in VFIFO Controller.
m_axi_rdata	IN	DATA_WIDTH	0	Read Data: Can be 32, 64, 128, 256, 512 or 1024 bits wide.
m_axi_rresp	IN	2	0	Read Response: Indicates the status of the read transfer. The allowable responses are OKAY, EXOKAY, SLVERR, and DECERR.
m_axi_rlast	IN	1	'0'	Read Last: Indicates the last transfer in a read burst.

Table 2-4: VFIFO Controller Pin Table (Cont'd)

Signal Name	Direction	Width	Default	Description
m_axi_ruser	IN	USER_WIDTH	0	Read Data Channel User: RUSER is not used in VFIFO Controller.
m_axi_rvalid	IN	1	'0'	Read Valid: Indicates that the required read data is available and the read transfer can complete: <ul style="list-style-type: none"> • 1 = Read data available. • 0 = Read data not available
m_axi_rready	OUT	1	'0'	Read Ready: Indicates that the master can accept the read data and response information: <ul style="list-style-type: none"> • 1= Master ready. • 0 = Master not ready.
Stream Interface Control Handshake Signals				
vfifo_mm2s_channel_full	IN	NUM_CHANNEL	0	<p>AXI4-Stream MM2S FIFO Programmable Full Indication:</p> <ul style="list-style-type: none"> • 1 = Indicates external MM2S FIFO cannot accept Read Data from DDR Memory Space • 0 = Indicates external MM2S FIFO can accept N consecutive transactions from DDR Memory Space • N = Space in MM2S FIFO (in bytes) / Configured Burst Size in Bytes <p>For example:</p> <ul style="list-style-type: none"> • Space in the FIFO when Programmable Full in Set is 4096 bytes. • Configured Burst Size in Bytes is 512 bytes • AR Weight for the channel can be set to $4096/512 = 8$. <p>Note: Full indication will be checked again only after N transactions are sent on MM2S interface.</p>

Table 2-4: VFIFO Controller Pin Table (Cont'd)

Signal Name	Direction	Width	Default	Description
vfifo_s2mm_channel_full	OUT	NUM_CHANNEL	0	Virtual FIFO Programmable Full Indication for the Channel: <ul style="list-style-type: none"> 1: <ul style="list-style-type: none"> For active channel, indicates Virtual FIFO can accept only 8 more beats of data from S2MM AXI4-Stream interface. For non-active channel, Arbiter should not select the Channel when Full indication is set. 0: Indicates Virtual FIFO can accept Stream data for the channel.
vfifo_mm2s_channel_empty	OUT	NUM_CHANNEL	1	Virtual FIFO Programmable Full Indication for the Channel: <ul style="list-style-type: none"> 1: Indicates Virtual FIFO Empty condition, when set indicates Virtual FIFO does not have any transactions for the corresponding channel. 0: Indicates Virtual FIFO Non Empty condition. When reset, indicates Virtual FIFO has at least one transaction for the corresponding channel.
Interrupt Signals				
vfifo_mm2s_rresp_err_intr	OUT	1	'0'	Read Response Error: This signal is clear on reset, indicating RRESP error received on AXI Memory Mapped interface.
vfifo_s2mm_bresp_err_intr	OUT	1	'0'	Write Response Error: This signal is clear on reset, indicating WRESP error received on AXI Memory Mapped interface.

Table 2-4: VFIFO Controller Pin Table (Cont'd)

Signal Name	Direction	Width	Default	Description
fifo_s2mm_overrun_err_intr	OUT	1	'0'	DDR Memory Space Overrun Condition: This signal is clear on reset, indicating more than 8 beats of data written after full indication is asserted on S2MM AXI4-Stream interface.
vfifo_idle	OUT	NUM_CHANNEL	1	<ul style="list-style-type: none"> • 1: Indicates that the Virtual FIFO controller does not have any data either in the data path or there is no pending outstanding write/read transaction to/from DDR. • 0: Indicates that the Virtual FIFO controller has one or more data either in the data path or there is a pending outstanding write/read transaction to/from DDR.

Designing with the Core

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

General Design Guidelines

The customizable AXI Virtual FIFO Controller core provides multiple Stream and Memory Mapped interface options. The Burst Size in Bytes option can be set to configure the maximum burst size on the AXI4 Memory Mapped interface. Maximizing the Burst Size on the AXI4 Memory Mapped interface can improve the overall system performance. Channel switch or a timeout detected within the configured burst size creates smaller transactions and can result in lowering the performance. For the Virtual FIFO Controller to generate transactions based on the configured burst size, the arbitration done in the N:1 stream multiplexing should also match burst size configuration in Virtual FIFO controller core.

AR weight can be used for maximizing the performance from external memory to the Stream Interface. Per Channel AR Channel Weight Allocation determines the maximum number of AXI MM outstanding read requests generated for the channel. This configuration option allows different channels to carry different bandwidths of data. Based on applications, the user can determine use of the packet (for Ethernet) or normal FIFOs (for PCIe) to be used in the 1:N Stream Switch. The depth of these FIFOs should be set based on the calculation provided in [Per Channel AR Channel Weight Allocation in Chapter 4](#). Maintaining deeper FIFOs in Stream Switch with larger AR weight programming in the Virtual FIFO can improve the overall external memory to Stream Interface performance.

Clocking

The AXI Virtual FIFO Controller core operates on a single clock (`ac1k`) and all input and output interface signals of AXI4-Stream and AXI4 interfaces are in synchronization with this clock.

Resets

The AXI Virtual FIFO Controller core uses a single asynchronous reset ($\overline{\text{ARESETN}}$) that is synchronized inside the core. The core is in reset state for 16 clock cycles on application of reset. The core enters into the reset state asynchronously but comes out of reset synchronously.

Protocol Description

The AXI Virtual FIFO Controller core uses the industry standard AMBA® AXI4-Stream and AXI4 Protocol Specification.

Figure 3-1 shows the timing diagram of AXI4-Stream interface where INFORMATION represents all AXI4-Stream signals except t_{valid} / t_{ready} .

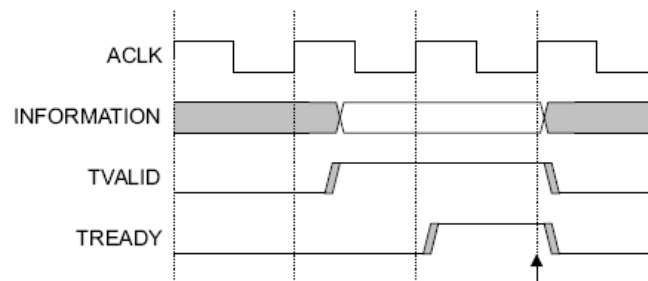


Figure 3-1: AXI4-Stream Interface

Figure 3-2 shows the timing diagram of AXI4 Write burst transaction, and Figure 3-3 shows the timing diagram of AXI4 Read burst transaction.

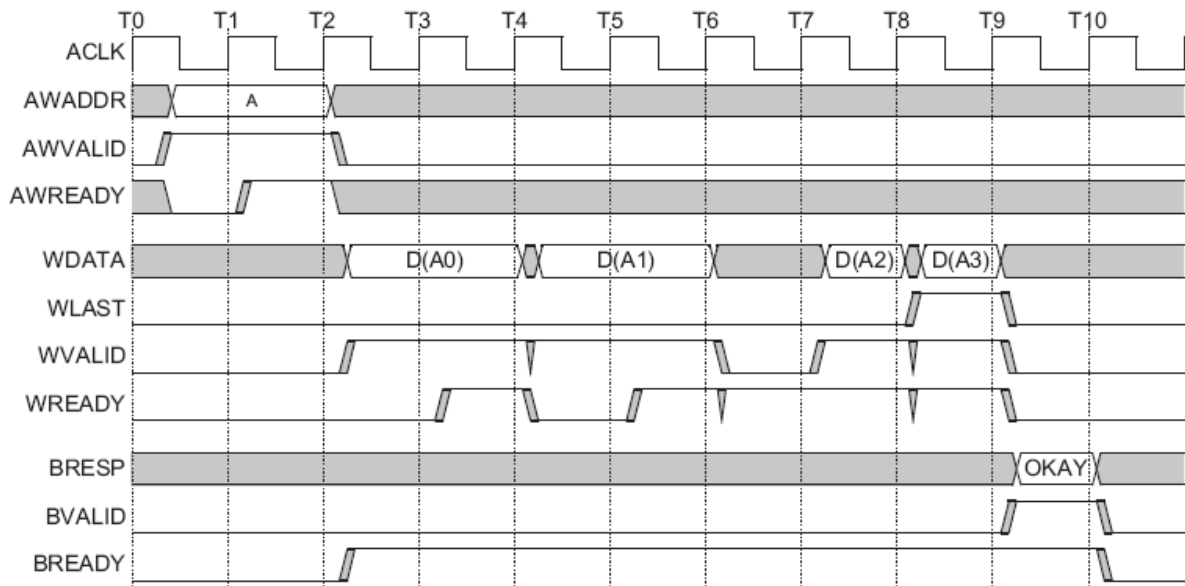


Figure 3-2: AXI4 Write Burst Transaction

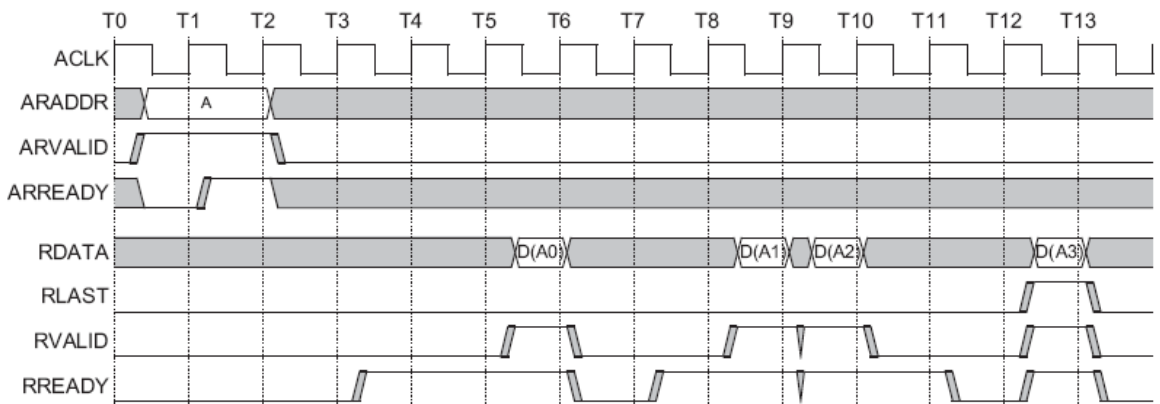


Figure 3-3: AXI4 Read Burst Transaction

Customizing and Generating the Core

This chapter includes information on using Xilinx tools to customize and generate the core.

Vivado Integrated Design Environment (IDE)

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.
2. Double-click the selected IP or select the Customize IP command from the toolbar or popup menu.

For details, see the sections, “Working with IP” and “Customizing IP for the Design” in the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 5] and the “Working with the Vivado IDE” section in the *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 8].

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

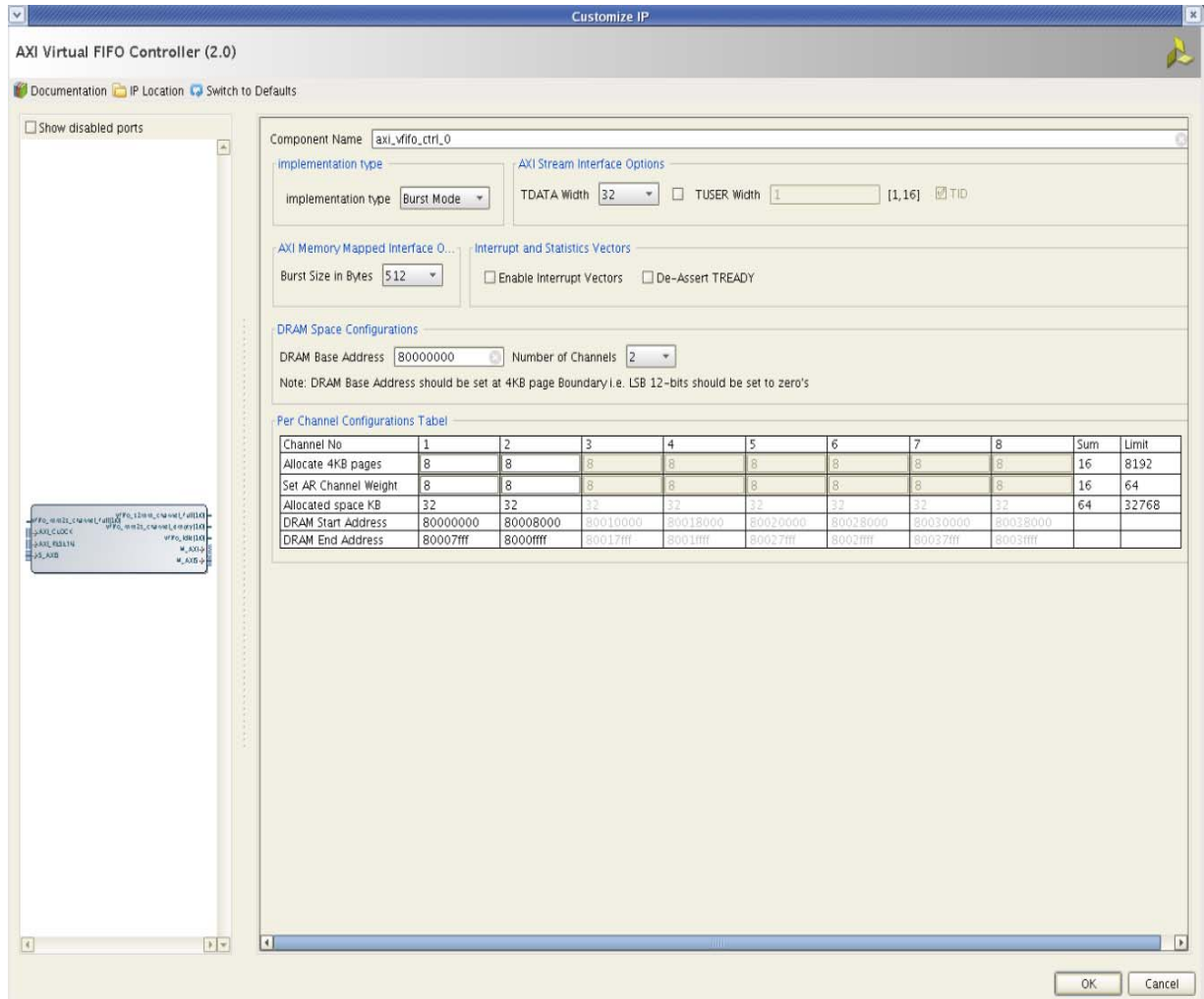


Figure 4-1: AXI Virtual FIFO Controller in Vivado IP Catalog

TDATA Width

`tdata` is the primary payload that provides the data on the AXI4-Stream interface. The TDATA Width user option configures the AXI4-Stream data width. The valid range for TDATA Width is 32, 64, 128, 256, 512 and 1024 bits. AXI MM DATA width is the same as the AXI4-Stream data width.

TUSER Width

`tuser` defines sideband information that can be transmitted alongside the data stream. For the Virtual FIFO controller, `tuser` is valid only for the first beat of the packet and can carry packet length or packet control information. The TUSER Width user option configures the `tuser` signal width. The valid range for TUSER Width is from 1 to 16 bits.

Burst Size in Bytes

The Burst Size in Bytes user option configures the maximum burst size on the AXI Memory Mapped interface. The valid range for Burst Size is 512, 1024, 2048 or 4096 bytes.

DRAM Base Address

DRAM Base Address configures the 32-bit base address of the Virtual FIFO Controller in the DDR Memory. The DRAM Base Address should always set at 4 KB page boundary, that is the LSB 12 bits of the DRAM Base Address should be set to all 0s.

Number of Channels

The Number of Channels user configuration defines the number of channels to be used in the Virtual FIFO controller. The valid range for Number of Channels is 2 to 8. The AXI4-Stream `tid` and `tdest` interface signal width is determined by the Number of Channels user configuration. `tid` and `tdest` width equals LOG_2 [number of channel].

Per Channel 4 KB Page Allocation

Per channel 4 KB Page Allocation defines the DRAM space to be used by the channel. The minimum number of pages allowed for any channel is 64. The maximum number of pages for a channel is derived from the total DDR space available for all channels. The sum of pages allocated across all channels should not exceed the total DDR space available for all channels. The maximum pages or DDR space allocation is also a function of "Burst Size in Byte" configuration and is listed in [Table 4-1](#).

Table 4-1: Maximum Number of Pages

Burst Size in Bytes	Maximum Available 4 KB Pages	Maximum Available DRAM Space (MB)
512	8192	32
1024	16384	64
2048	32768	128
4096	65536	256

Note: Each channel has a 8,192 limit.

Per Channel AR Channel Weight Allocation

Per Channel AR Channel Weight Allocation determines the maximum number of AXI MM outstanding read requests generated for the channel. AR Channel Weight configuration should be done by considering the programmable full threshold set in MM2S Stream Interconnect FIFO, as shown in [Table 4-2](#).

For example:

- Configured Burst Size in Bytes = 512 bytes
- Configured AR Weight = 4
- AR Burst = 512 x 4 = 2048 bytes
- Space available in FIFO when Prog Full is deasserted should be at least 2 x AR Burst = 4096 bytes

Table 4-2: Example FIFO Settings and AR Weight Configuration

Interface Width	FIFO Width x Depth	FIFO Prog Full Threshold	Virtual FIFO Burst Size	Example AR Weight	Space in FIFO vs AR Burst
64	64x1024 [8 K bytes]	512 [4 K bytes]	2048	1	4 K / 2 K
64	64x2048 [16 K bytes]	512 [4 K bytes]	2048	3	12 K / 6 K
512	512x512 [32 K bytes]	256 [16 K bytes]	4096	2	16 K / 8 K
512	512x1024 [64 K bytes]	256 [16 K bytes]	4096	6	48 K / 24 K

Design Parameters

Certain features of the core are customizable and enable a design that uses only the necessary resources and runs at the best possible performance. Table 4-3 lists the VFIFO Controller design parameters.

Table 4-3: Design Parameters

Parameter Name	VHDL Type	Allowable Values	Default Value	Description
C_FAMILY	String	See the IP Facts table for supported devices.	Virtex7	Carries the family information.
C_DRAM_BASE_ADDR	String	32 bit hexadecimal starting at 4KB boundary	"80000000"	Determines the base address of the DRAM. LSB 12 bits should be zero for the address.
C_AXI_ADDR_WIDTH	Integer	32	32	Determines the DRAM address width. Width of C_DRAM_BASE_ADDR = C_AXI_ADDR_WIDTH.
C_AXIS_TDATA_WIDTH	Integer	32, 64, 128, 256, 512, 1024	32	Determines the data width.

Table 4-3: Design Parameters (Cont'd)

Parameter Name	VHDL Type	Allowable Values	Default Value	Description
C_AXI_BURST_SIZE	Integer	512, 1024, 2048, 4096	512	Determines the burst size of a transaction. <ul style="list-style-type: none"> If axis_tdata_width is 32, burst sizes of 512 and 1024 are allowed. If axis_tdata_width is 64, burst sizes of 512, 1024, and 2048 are allowed. If axis_tdata_width is 128 or 256, burst sizes of 512, 1024, 2048, and 4096 are allowed. If axis_tdata_width is 512, burst sizes of 1024, 2048, and 4096 are allowed. If axis_tdata_width is 1024, burst sizes of 2048 and 4096 are allowed.
C_HAS_AXIS_TUSER	Integer	0 - 1	0	<ul style="list-style-type: none"> 0: Core does not have USER interface signal. 1: Core has USER interface signal.
C_HAS_AXIS_TLAST	Integer	0 - 1	0	<ul style="list-style-type: none"> 0: Core does not have LAST interface signal. 1: Core has LAST interface signal.
C_AXIS_TUSER_WIDTH	Integer	1 - 16	1	Determines the USER width. If C_HAS_AXIS_TUSER is 1, then C_AXIS_TUSER_WIDTH can be anything between 1 and 16, otherwise C_AXIS_TUSER_WIDTH is 1
C_NUM_CHANNEL	Integer	2 - 8	2	Determines the number of channels.
C_IMPLEMENTATION_TYPE	Integer	0-2	1	Implementation Type: <ul style="list-style-type: none"> 0: Packet Mode (Not currently supported) 1: Burst Mode 2: Packet Buffer Mode (Not currently supported)
C_HAS_AXIS_TID	integer	0-1	1	<ul style="list-style-type: none"> 0: Core does not have an ID interface signal 1: Core has an ID interface signal
C_ENABLE_INTERRUPT	integer	0-1	0	Enable interrupt port mode.
C_DEASSERT_TREADY	integer	0-1	0	Deasserts S_AXIS_TREADY on any channel's Full indicated by VFIFO_S2MM_CHANNEL_FULL.
C_ARB_WEIGHT_CH0	integer	1-8	8	Channel 0 arbitration weight for read arbitrator. Determines the maximum number of consecutive read requests allowed for the channel.

Table 4-3: Design Parameters (Cont'd)

Parameter Name	VHDL Type	Allowable Values	Default Value	Description
C_ARB_WEIGHT_CH1	integer	1-8	8	Channel 1 arbitration weight for read arbitrator. Determines the maximum number of consecutive read requests allowed for the channel.
C_ARB_WEIGHT_CH2	integer	1-8	8	Channel 2 arbitration weight for read arbitrator. Determines the maximum number of consecutive read requests allowed for the channel.
C_ARB_WEIGHT_CH3	integer	1-8	8	Channel 3 arbitration weight for read arbitrator. Determines the maximum number of consecutive read requests allowed for the channel.
C_ARB_WEIGHT_CH4	integer	1-8	8	Channel 4 arbitration weight for read arbitrator. Determines the maximum number of consecutive read requests allowed for the channel.
C_ARB_WEIGHT_CH5	integer	1-8	8	Channel 5 arbitration weight for read arbitrator. Determines the maximum number of consecutive read requests allowed for the channel.
C_ARB_WEIGHT_CH6	integer	1-8	8	Channel 6 arbitration weight for read arbitrator. Determines the maximum number of consecutive read requests allowed for the channel.
C_ARB_WEIGHT_CH7	integer	1-8	8	Channel 7 arbitration weight for read arbitrator. Determines the maximum number of consecutive read requests allowed for the channel.
C_AXIS_TID_WIDTH	Integer	1 - 3	1	Determines the ID width. Should be rounded-up value of $\log_2(C_NUM_CHANNEL)$
C_NUM_PAGE_CH0	Integer	8 - 8192	8	Determines the number of 4 K pages for channel 0.
C_NUM_PAGE_CH1	Integer	8 - 8192	8	Determines the number of 4 K pages for channel 1.
C_NUM_PAGE_CH2	Integer	8 - 8192	8	Determines the number of 4 K pages for channel 2.
C_NUM_PAGE_CH3	Integer	8 - 8192	8	Determines the number of 4 K pages for channel 3.
C_NUM_PAGE_CH4	Integer	8 - 8192	8	Determines the number of 4 K pages for channel 4.
C_NUM_PAGE_CH5	Integer	8 - 8192	8	Determines the number of 4 K pages for channel 5.

Table 4-3: Design Parameters (Cont'd)

Parameter Name	VHDL Type	Allowable Values	Default Value	Description
C_NUM_PAGE_CH6	Integer	8 - 8192	8	Determines the number of 4 K pages for channel 6.
C_NUM_PAGE_CH7	Integer	8 - 8192	8	Determines the number of 4 K pages for channel 7.

Output Generation

For details, see "Generating IP Output Products" in the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 5].

Constraining the Core

There are no constraints associated with this core other than the IP-level constraints that are added to the project when the IP is generated from the Vivado software.

Simulation

This chapter contains information about simulating in the Vivado® Design Suite environment. For details, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900) [\[Ref 3\]](#).

Synthesis and Implementation

This chapter contains information about synthesis and implementation in the Vivado® Design Suite environment.

For details about synthesis and implementation, see “Synthesizing IP” and “Implementing IP” in the *Vivado Design Suite User Guide: Designing with IP* (UG896) [\[Ref 5\]](#).

Detailed Example Design

This chapter provides detailed information about the example design, including a description of files and the directory structure generated by the Xilinx Vivado® Design Suite.

The AXI Virtual FIFO Controller example design consists of the following:

- HDL wrapper that instantiates the AXI Virtual FIFO Controller netlist
- AXI Virtual FIFO Controller constraint file

The AXI Virtual FIFO Controller example design has been tested with the Vivado® Design Suite.

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 AXI Virtual FIFO Controller has been tested with Vivado Design Suite, Xilinx XSIM, and Mentor Graphics Questa SIM simulator.

Hardware Testing

The AXI Virtual FIFO Controller has been hardware validated at 200 MHz on KC705 board using Kintex-7 -2 speed grade device (325T). The VFIFO Controller was configured for four channels and was independently tested for a data width of 512-bits and 64-bits. The data width of 512-bits with 4096 burst size was benchmarked for 80 Gb/s throughput across four channels.

Migrating and 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) [\[Ref 6\]](#).

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.

Port Changes

There are no port changes.

Debugging

This appendix includes details about resources available on the Xilinx Support website and debugging tools.

Finding Help on Xilinx.com

To help in the design and debug process when using the AXI Virtual FIFO Controller, 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 AXI Virtual FIFO Controller. 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 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.

Master Answer Record for the AXI Virtual FIFO Controller

AR: [54809](#)

Technical Support

Xilinx provides technical support in 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](#).

Note: Access to WebCase is not available in all cases. Please login to the WebCase tool to see your specific support options.

Debug Tools

There are many tools available to address AXI Virtual FIFO Controller design issues. It is important to know which tools are useful for debugging various situations.

Example Design

The AXI Virtual FIFO Controller is delivered with an example design that can be synthesized, complete with functional test benches. Information about the example design can be found in [Chapter 8, Detailed Example Design](#).

Vivado Design Suite Debug Feature

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

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

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

Additional Resources

Xilinx Resources

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

www.xilinx.com/support.

References

These documents provide supplemental material:

1. [AMBA AXI4-Stream Protocol Specification](#)
2. [AMBA AXI4 Protocol Specification](#)
3. *Vivado Design Suite User Guide: Logic Simulation* ([UG900](#))
4. *Vivado Design Suite User Guide: Implementation* ([UG904](#))
5. *Vivado Design Suite User Guide: Designing with IP* ([UG896](#))
6. *ISE to Vivado Design Suite Migration Methodology Guide* ([UG911](#))
7. *Vivado Design Suite User Guide: Programming and Debugging* ([UG908](#))
8. *Vivado Design Suite User Guide: Getting Started* ([UG910](#))

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
11/18/2015	2.0.1	Added support for UltraScale+ families.
11/11/2014	2.0.1	Documentation only. Modified Table 4-1 title and added a note to the table.
12/18/2013	2.0	Added support for UltraScale™ architecture.
10/02/2013	2.0	Added details about customizing the core in Vivado Integrated Design Environment (IDE) in Chapter 4.
03/20/2013	2.0	Updated core to v2.0. Removed support for ISE Design Suite. Added Appendix B, Debugging.
05/25/2012	1.0	Initial Xilinx release.
04/24/2012	Draft	Xilinx Confidential DRAFT. Approved for external release under NDA only.

Notice of Disclaimer

The information disclosed to you hereunder (the "Materials") is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of the Limited Warranties which can be viewed at <http://www.xilinx.com/warranty.htm>; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in Critical Applications: <http://www.xilinx.com/warranty.htm#critapps>.

© Copyright 2012-2015 Xilinx, Inc. Xilinx, the Xilinx logo, Artix, ISE, Kintex, Spartan, Virtex, Vivado, Zynq, and other designated brands included herein are trademarks of Xilinx in the United States and other countries. All other trademarks are the property of their respective owners.