LogiCORE IP AXI Virtual FIFO Controller v1.1

Product Guide

PG038 July 25, 2012

Table of Contents

SECTION I: SUMMARY

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 active-Low reset
- 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]	Zynq™-7000[2], Kintex-7, Virtex-7, Artix-7
Supported User Interfaces	AXI4 and AXI4-Stream
Resources	See Table 2-1.
Provided with Core	
Design Files	Netlist
Example Design	VHDL
Test Bench	Not Provided
Constraints File	Vivado: XDC ISE: UCF
Simulation Model	Verilog and/or VHDL Structural
Supported S/W Driver	N/A
Tested Design Flows[3]	
Design Entry	Vivado™ Design Suite v2012.2[4] ISE™ Design Suite v14.2
Simulation	Mentor Graphics ModelSim Xilinx ISIM/XSIM
Synthesis	Vivado Synthesis ISE Design Suite
Support	
Provided by Xilinx @ www.xilinx.com/support	

Notes:

1. For a complete listing of supported devices, see the release notes for this core.
2. Supported in ISE Design Suite implementations only.
3. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.
4. Supports only 7 series devices.

Overview

AXI Virtual FIFO controller supports a true multi-channel architecture allowing per-clock-cycle channel arbitration. The AXI4-Stream TDEST signal provided to the AXI4-Stream slave interface is used as a channel identifier. TDEST 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.

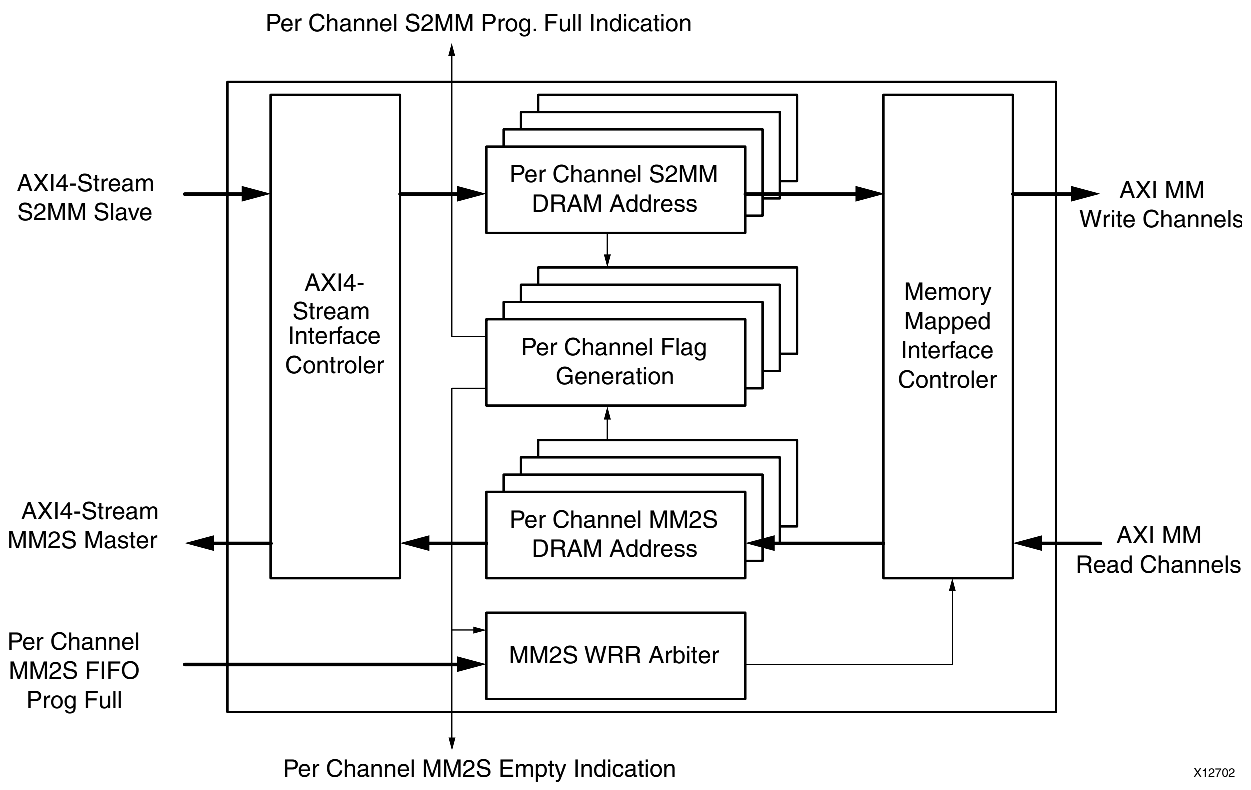

Per Channel S2MM Prog. Full Indication
AXI4-Stream S2MM Slave
AXI4-Stream Interface Controler
Per Channel S2MM DRAM Address
Memory Mapped Interface Controler
AXI MM Write Channels
Per Channel Flag Generation
AXI4-Stream MM2S Master
Per Channel MM2S DRAM Address
AXI MM Read Channels
Per Channel MM2S FIFO Prog Full
MM2S WRR Arbiter
Per Channel MM2S Empty Indication
X12702

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 (WRR) 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 (WRR) 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 details refer to Per Channel AR Channel Weight Allocation in Chapter 7.

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.

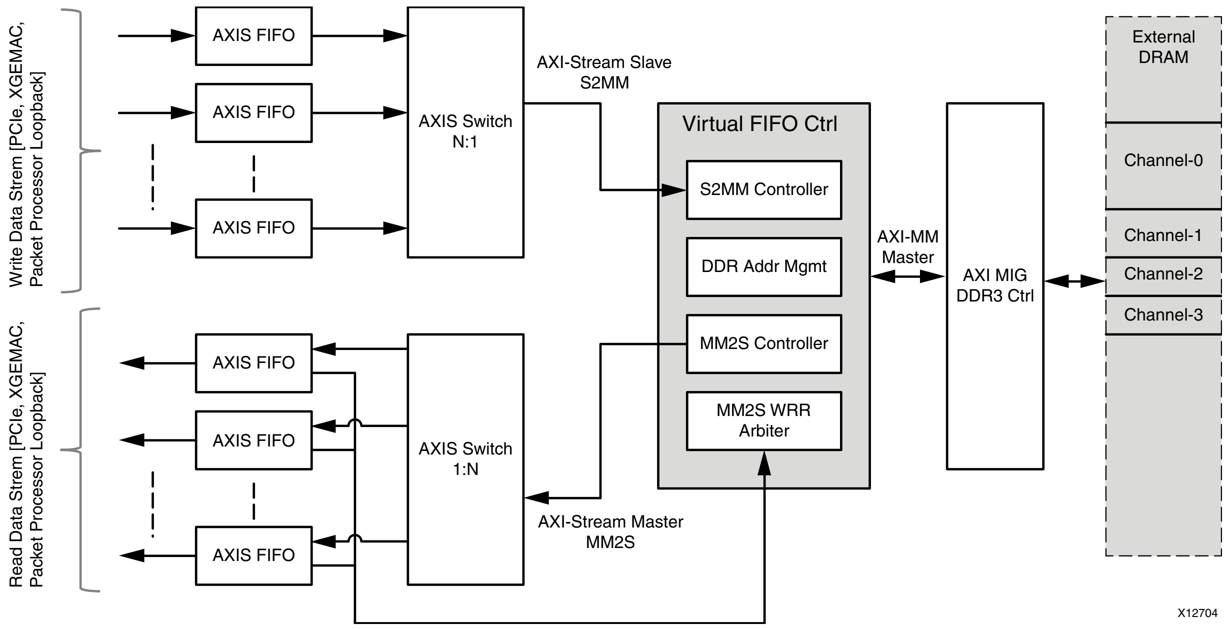


Figure 1-2: **Application Block Diagram**

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 and ISE Design Suite software under the terms of the Xilinx End User License. The core may be accessed through the Vivado Design Suite and ISE CORE Generator IP catalog.

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

AXI Virtual FIFO Controller operates up to 200 MHz on a Kintex™-7 device at -2 and higher speed grades.

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 7 clock cycles to present the received RDATA on AXI4-Stream side.

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 thoughput across four channels.

Resource Utilization

Because the AXI Virtual FIFO Controller core will be 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 2-1 for a Virtex-7 FPGA. These values were generated using the Xilinx® CORE Generator™ tools, v14.2. 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:

- Data Width = 1024
- Burst Size = 4096
- Number of Channel = 4
- TUSER width = 16

Table 2-1: **Resource Utilization for Virtex-7 FPGAs**

Device Resource					F_{MAX}(MHz)
Block RAMs	**DSP48**	**Slices**	**Slice Flip-Flops**	**LUTs**	F_{MAX}
16	18	10827	9929	1702	216

I/O Signals

Signals for the VFIFO Controller are listed in Table 2-2.

Table 2-2: **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.

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

Signal Name	Direction	Width	Default	Description
ARESETN	IN	1	'0'	Global reset: This signal is active-Low. ARESETN must be asserted at for least for one clock cycle.
AXI4-Stream Input Signals From Switch to AXI Virtual FIFO Controller				
S_AXIS_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. The VFIFO Controller determines FIFO underrun cased on: • 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 The above condition will result in a transaction on AXI Memory Mapped interface.
S_AXIS_TREADY	OUT	1	'0'	TREADY: Indicates that the slave can accept a transfer in the current cycle.
S_AXIS_TDATA	IN	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.
S_AXIS_TSTRB	IN	DATA_WIDTH/8	0	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. ***Note:*** S_AXIS_TSTRB input is not used in Virtual FIFO Controller.

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

Signal Name	Direction	Width	Default	Description
S_AXIS_TKEEP	IN	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 be ignored when TLAST = 0. 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.
S_AXIS_TLAST	IN	1	'0'	TLAST: Indicates the boundary of a packet.
S_AXIS_TID	IN	LOG2(NUM_OF _Channels)	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 considered valid with only first beat of a packet.
S_AXIS_TDEST	IN	LOG2(NUM_OF _Channels)	0	TDEST: Channel Identifier that provides routing information for the data stream. TDEST is a Destination AXI4-Stream identifier and should be kept valid or same for the entire packet duration.
S_AXIS_TUSER	IN	USER_WIDTH	0	TUSER: The user-defined sideband information that can be transmitted alongside the data stream. TUSER is considered valid only in a first data beat of the Packet and can carry packet control information.
AXI Virtual FIFO Controller Output Signals To AXI4-Stream Switch				
M_AXIS_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.
M_AXIS_TREADY	IN	1	'0'	TREADY: Indicates that the slave can accept a transfer in the current cycle.

Table 2-2: **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 2-2: **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 2-2: **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	0	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: • 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: • 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 2-2: **VFIFO Controller Pin Table *(Cont'd)***

Signal Name	Direction	Width	Default	Description
M_AXI_WSTRB	OUT	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
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'	Write Valid: Indicates that valid write data and strobes are available: • 1 = Write data and strobes available. • 0 = Write data and strobes not available
M_AXI_WREADY	IN	1	'0'	Write Ready: Indicates that the slave can accept the write data: • 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.

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

Signal Name	Direction	Width	Default	Description
M_AXI_BVALID	IN	1	'0'	Write Response Valid: Indicates that a valid write response is available: • 1 = Write response available. • 0 = Write response not available.
M_AXI_BREADY	OUT	1	'1'	Response Ready: Indicates that the master can accept the response information. • 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.

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

Signal Name	Direction	Width	Default	Description
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	0	Cache Type: This signal provides additional information about the cacheable characteristics of the transfer. Cache Type is always set to zero.
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. • 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: • 1 = Slave ready. • 0 = Slave not ready.

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

Signal Name	Direction	Width	Default	Description
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.
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: • 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: • 1= Master ready. • 0 = Master not ready.

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

Signal Name	Direction	Width	Default	Description
Stream Interface Control Handshake Signals				
VFIFO_MM2S_CHANNEL_FULL	IN	NUM_CHANNEL	0	AXI4-Stream MM2S FIFO Programmable Full Indication: • 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 For example: • 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. ***Note:*** Full indication will be checked again only after N transactions are sent on MM2S interface.
VFIFO_S2MM_CHANNEL_FULL	OUT	NUM_CHANNEL	0	Virtual FIFO Programmable Full Indication for the Channel: • 1: ◦ 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.

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

Signal Name	Direction	Width	Default	Description
VFIFO_MM2S_CHANNEL_EMPTY	OUT	NUM_CHANNEL	1	Virtual FIFO Programmable Full Indication for the Channel: • 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.
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	• 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.

Chapter 3

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 the Per Channel AR Channel Weight Allocation in Chapter 7. 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 (ACLK) 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 (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 TVALID/TREADY.

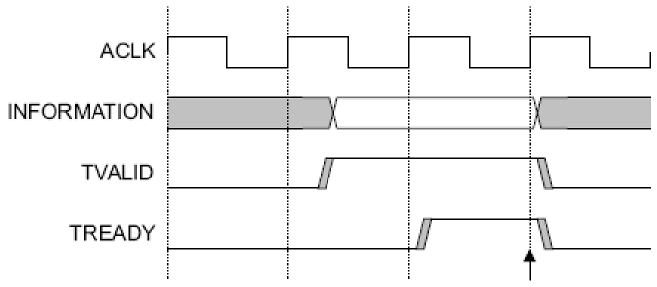


Figure 3-1: **AXI4-Stream Interface**

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.

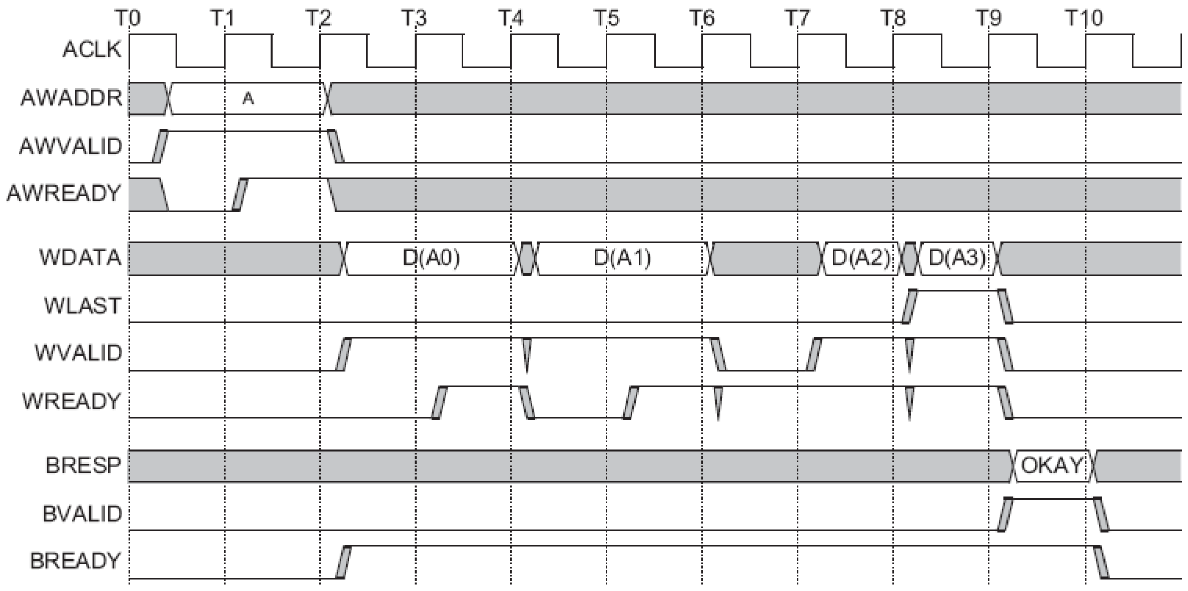


Figure 3-2: **AXI4 Write Burst Transaction**

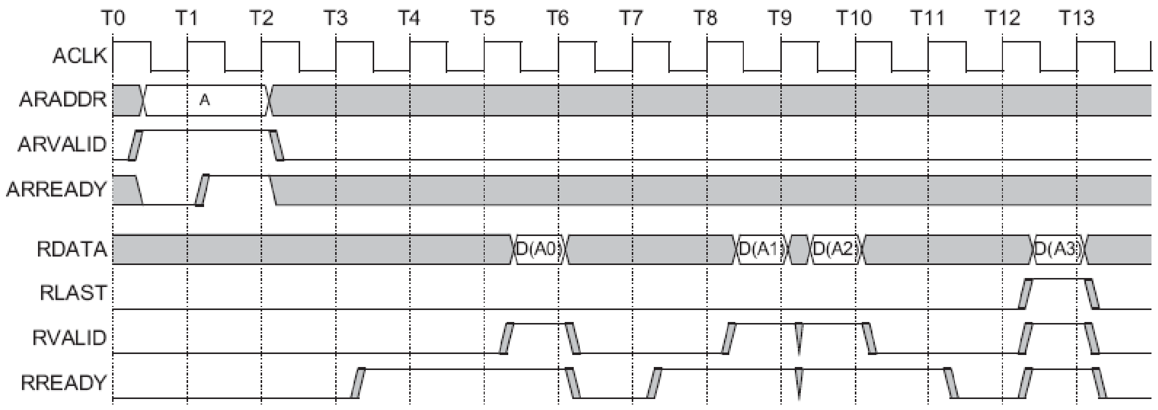


Figure 3-3: **AXI4 Read Burst Transaction**

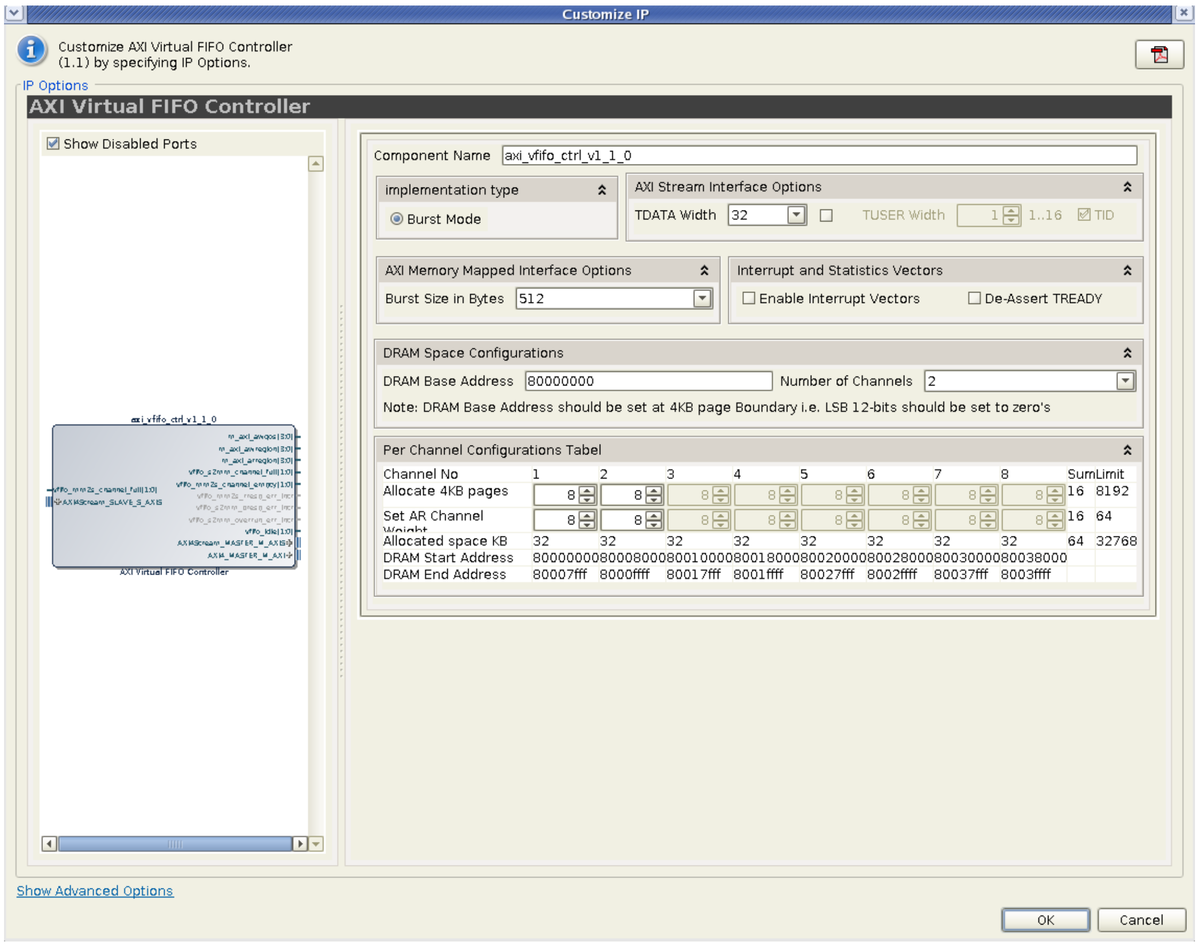
SECTION II: VIVADO DESIGN SUITE

Customizing and Generating the Core

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

GUI

The AXI Virtual FIFO Controller core is located under **AXI Infrastructure** in the Vivado IP catalog.



Customize AXI Virtual FIFO Controller (1.1) by specifying IP Options.

AXI Virtual FIFO Controller

Component Name: axi_vfifo_ctrl_v1_1_0

implementation type: Burst Mode [✓]

AXI Stream Interface Options: TDATA Width 32; TUSER Width 1 1..16; TID ✓

AXI Memory Mapped Interface Options: Burst Size in Bytes 512

Interrupt and Statistics Vectors: ☐ Enable Interrupt Vectors ☐ De-Assert TREADY

DRAM Space Configurations: DRAM Base Address 80000000; Number of Channels 2

Note: DRAM Base Address should be set at 4KB page Boundary i.e. LSB 12-bits should be set to zero's

Per Channel Configurations Tabel

Channel No	1	2	3	4	5	6	7	8	Sum	Limit
Allocate 4KB pages	8	8	8	8	8	8	8	8	16	8192
Set AR Channel Weight	8	8	8	8	8	8	8	8	16	64
Allocated space KB	32	32	32	32	32	32	32	32	64	32768
DRAM Start Address	80000000	80008000	80010000	80018000	80020000	80028000	80030000	80038000		
DRAM End Address	80007fff	8000ffff	80017fff	8001ffff	80027fff	8002ffff	80037fff	8003ffff		

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

Details of the GUI options can be found in Chapter 7, Customizing and Generating the Core.

Output Generation

See Directory and File Contents in Chapter 6 for more details on the files created when the core is generated.

Constraining the Core

This chapter contains details about applicable constraints.

Required Constraints

There are no required constraints.

Device, Package, and Speed Grade Selections

See IP Facts for details about supported devices.

Clock Frequencies

This core can operate up to 200 MHz with the maximum possible configuration on a Virtex-7 device.

Clock Management

The AXI Virtual FIFO Controller uses a single clock and reset. The only constraints needed are clocking constraints.

Clock Placement

The AXI Virtual FIFO Controller core does not have any special requirement on the placement of the clock.

Banking

There are no banking constraints.

Transceiver Placement

There are no transceiver constraints.

I/O Standard and Placement

There are no I/O constraints.

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.

Directory and File Contents

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 sources like XCI, XDC, TCL and document files.
  - 📁 <component name>/synth
    Contains the source file necessary to synthesize the AXI Virtual FIFO Controller
  - 📁 <component name>/sim
    Contains the source file necessary to simulate the AXI Virtual FIFO Controller
  - 📁 <component name>/example_design
    Contains the source file necessary to synthesize the example design
  - 📁 <component name>/simulation
    Contains the source file necessary to simulate the example design
    - 📁 <component name>/simulation/functional
      Contains the simulation script file necessary to launch simulation of core including the example design

The AXI Virtual FIFO Controller core directories and their associated files are defined in the following sections.

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

This directory contains templates for instantiation of the core, example design, synth, XML and the xci files.

Table 6-1: **<component name> Directory**

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

<component name>/synth

The synth directory contains the Virtual FIFO synthesis file.

Table 6-2: **synth Directory**

Name	Description
<component_name>.vhd	A VHDL file from VIVADO software to synthesize the Virtual FIFO core.

<component name>/sim

The sim directory contains the Virtual FIFO simulation wrapper file.

Table 6-3: **sim Directory**

Name	Description
<component_name>.vhd	A VHDL file from VIVADO software to simulate the Virtual FIFO core.

<component name>/example_design

The example design directory contains the example design files provided with the core.

Table 6-4: **Example Design Directory**

Name	Description
<component_name>_exdes.vhd	The VHDL top-level file for the example design; it instantiates the Virtual FIFO core. This file contains entity with the IO's required for the core configuration.
<component_name>_exdes.xdc	Provides an example clock constraint for processing the Virtual FIFO core using the VIVADO implementation tools.

<component name>/simulation

The simulation directory contains the simulation files provided with the core.

Table 6-5: **Simulation Directory**

Name	Description
<component_name>_tb.vhd	The VHDL top-level file for the simulation. It instantiates the example design of Virtual FIFO core. This file contains the Stimulus required for the core simulation.

<component name>/simulation/functional

The functional directory contains the scripts to launch XSIM Simulation with simulation test bench set as the top entity.

Table 6-6: **Functional Directory**

Name	Description
Simulate_xsim.[sh|bat]	Compiles all the required files and launches simulation. The SH file is used in the Linux environment. the BAT file is used in the Windows environment.

Example Design

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 Vivado Design Suite v2012.2.

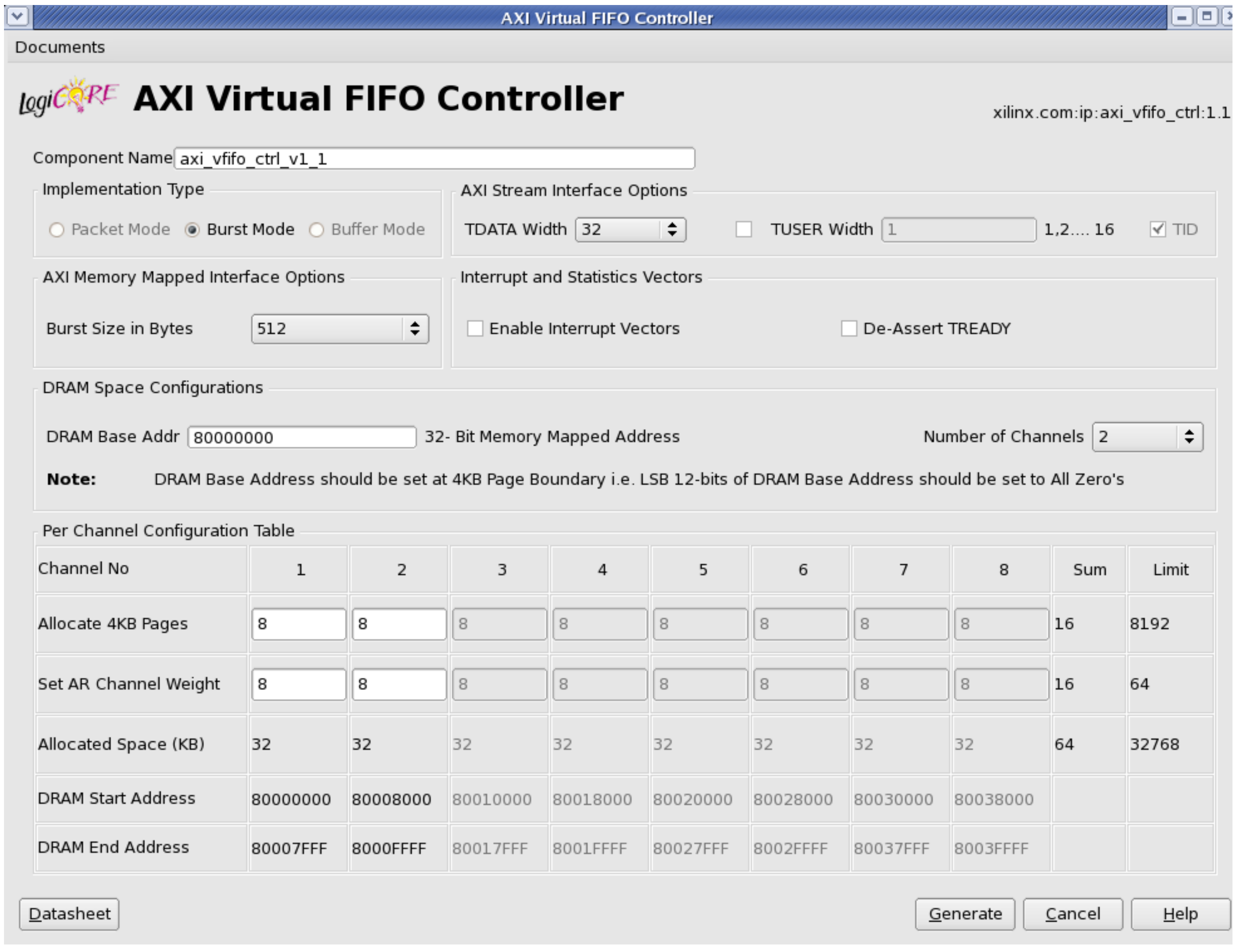
SECTION III: ISE DESIGN SUITE

Customizing and Generating the Core

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

GUI

The CORE Generator GUI is shown in Figure 7-1.



AXI Virtual FIFO Controller

xilinx.com:ip:axi_vfifo_ctrl:1.1

Component Name axi_vfifo_ctrl_v1_1

Implementation Type: Packet Mode [✓ Burst Mode] Buffer Mode

AXI Stream Interface Options: TDATA Width 32; ☐ TUSER Width 1 1,2.... 16; ☑ TID

AXI Memory Mapped Interface Options: Burst Size in Bytes 512

Interrupt and Statistics Vectors: ☐ Enable Interrupt Vectors; ☐ De-Assert TREADY

DRAM Space Configurations: DRAM Base Addr 80000000 32- Bit Memory Mapped Address; Number of Channels 2

Note: DRAM Base Address should be set at 4KB Page Boundary i.e. LSB 12-bits of DRAM Base Address should be set to All Zero's

Per Channel Configuration Table

Channel No	1	2	3	4	5	6	7	8	Sum	Limit
Allocate 4KB Pages	8	8	8	8	8	8	8	8	16	8192
Set AR Channel Weight	8	8	8	8	8	8	8	8	16	64
Allocated Space (KB)	32	32	32	32	32	32	32	32	64	32768
DRAM Start Address	80000000	80008000	80010000	80018000	80020000	80028000	80030000	80038000		
DRAM End Address	80007FFF	8000FFFF	80017FFF	8001FFFF	80027FFF	8002FFFF	80037FFF	8003FFFF		

Figure 7-1: **Customization GUI**

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 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 a 4 KB page boundary, that is the least significant 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 LOG2 [number of channels].

Per Channel 4 KB Page Allocation

The 4 KB page allocation specifies the size of the channel divided into 4 KB pages. 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 7-1.

Table 7-1: **Maximum Number of Pages Per Channel**

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

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 the MM2S Stream Interconnect FIFO to avoid blocking other channels from accessing VFIFO Controller. The settings are listed in Table 7-2.

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 7-2: **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 7-3 lists the VFIFO Controller design parameters.

Table 7-3: **Design Parameters**

Parameter Name	VHDL Type	Allowable Values	Default Value	Description
C_FAMILY	String	All 7 series 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.
C_AXI_BURST_SIZE	Integer	512, 1024, 2048, 4096	512	Determines the burst size of a transaction. • 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	• 0: Core does not have USER interface signal. • 1: Core has USER interface signal.
C_HAS_AXIS_TLAST	Integer	0 - 1	0	• 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: • 0: Packet Mode (Not currently supported) • 1: Burst Mode • 2: Packet Buffer Mode (Not currently supported)
C_HAS_AXIS_TID	integer	0-1	1	• 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.

Table 7-3: **Design Parameters *(Cont'd)***

Parameter Name	VHDL Type	Allowable Values	Default Value	Description
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.
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 log2(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.

Table 7-3: **Design Parameters *(Cont'd)***

Parameter Name	VHDL Type	Allowable Values	Default Value	Description
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.
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

See Directory and File Contents in Chapter 9 for more details on the files created when the core is generated.

Constraining the Core

This chapter contains details about applicable constraints.

Required Constraints

There are no required constraints other than the clock constraint.

Device, Package, and Speed Grade Selections

See IP Facts for details about supported devices.

Clock Frequencies

The AXI Virtual FIFO Generator core can operate up to 200 MHz with the maximum possible configuration on a Virtex-7 device.

Clock Management

The AXI Virtual FIFO Controller uses a single clock and reset. The clock may be from DCM or PLL.

Clock Placement

The AXI Virtual FIFO Controller core does not have any special requirement on the placement of the clock.

Banking

There are no banking constraints.

Transceiver Placement

There are no transceiver constraints.

I/O Standard and Placement

There are no I/O constraints.

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 CORE Generator™ software, the purpose and contents of the provided scripts, and the contents of the example HDL wrappers.

Directory and File Contents

The following folders are included in the example design:

- **<project directory>**
  Top-level project directory for the CORE Generator software. The name is defined by the user.
  - **<project directory>/<component name>**
    Core release notes file.
    - **<component_name>/doc**
      Contains the VFIFO Controller solution PDF documentation.
    - **<component_name>/example_design**
      Contains the Verilog and VHDL design files.
    - **<component_name>/implement**
      Contains the implementation script files.
      - **<component_name>/implement/results**
        Contains the results directory and results that are created after the implementation scripts are run.

The AXI Virtual FIFO Controller core directories and their associated files are defined in the following sections.

<project directory>

The <project directory> contains all the CORE Generator software project files.

Table 9-1: **<project directory> Directory**

Name	Description
<component_name>.ngc	Top-level netlist
<component_name>.v[hd]	Verilog or VHDL simulation model
<component_name>.xco	CORE Generator software project-specific option file. Can be used as an input to the CORE Generator software.
<component_name>_flist.txt	List of files delivered with the core.
<component_name>.{veo|vho}	VHDL or Verilog instantiation template.

<project directory>/<component name>

The <component name> directory contains the release notes file provided with the core, which can include last-minute changes and updates.

Table 9-2: **<component name> Directory**

Name	Description
axi_vfifo_ctrl_v1_1_readme.txt	Core name release notes file.

<component_name>/doc

The doc directory contains the PDF documentation provided with the core.

Table 9-3: **Doc Directory**

Name	Description
pg038_axi_vfifo_ctrl.pdf	LogiCORE IP AXI Virtual FIFO Controller Product Guide.

<component_name>/example_design

The example design directory contains the example design files provided with the core.

Table 9-4: **Example Design**

Name	Description
<component_name>_top.ucf	Provides example constraints necessary for processing the AXI Virtual FIFO Controller core using the Xilinx implementation tools.
<component_name>_top.xdc	Provides example constraints necessary for processing the AXI Virtual FIFO Controller core using the Xilinx implementation tools.

Table 9-4: **Example Design *(Cont'd)***

Name	Description
<component_name>_top.vhd	The VHDL top-level file for the example design; it instantiates the AXI Virtual FIFO Controller core. This file contains entity with the IO's required for the core configuration.
<component_name>_top_wrapper.v[hd]	The VHDL wrapper file for the example design <component_name>_top.vhd file.

<component_name>/implement

The implement directory contains the core implementation script files.

Table 9-5: **Implement Directory**

Name	Description
implement.{bat|sh}	A Windows (.bat) or Linux script that processes the example design.
xst.prj	The XST project file for the example design that lists all of the source files to be synthesized. Only available when the CORE Generator software project option is set to ISE or Other.
xst.scr	The XST script file for the example design used to synthesize the core. Only available when the CORE Generator software Vendor project option is set to ISE or Other.

<component_name>/implement/results

The results directory is created and populated by the implement script.

Table 9-6: **Results Directory**

Name	Description
xst.scr	Implement script result files.

Example Design

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

- AXI Virtual FIFO Controller netlist
- HDL wrapper which instantiates the AXI Virtual FIFO Controller netlist

The AXI Virtual FIFO Controller example design has been tested with Xilinx ISE® software v14.2.

Implementation

The implementation script is either a shell script (SH) or batch file (BAT) that processes the example design through the Xilinx tool flow. It is located at:

Linux

```
<project_dir>/<component_name>/implement/implement.sh
```

Windows

```
<project_dir>/<component_name>/implement/implement.bat
```

The implement script performs these steps:

- Synthesizes the HDL example design files using XST
- Runs NGDBuild to consolidate the core netlist and the example design netlist into the NGD file containing the entire design
- Maps the design to the target technology
- Place-and-routes the design on the target device
- Performs static timing analysis on the routed design using Timing Analyzer (TRCE)
- Generates a bitstream
- Enables Netgen to run on the routed design to generate a VHDL or Verilog netlist (as appropriate for the Design Entry project setting) and timing information in the form of SDF files

The Xilinx tool flow generates several output and report files. These are saved in the following directory created by the implement script:

```
<project_dir>/<component_name>/implement/results
```

SECTION IV: APPENDICES

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 Xilinx ISE® software v14.2, Xilinx ISIM/XSIM, and Mentor Graphics ModelSim 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 thoughput across four channels.

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 user guide:

- *AMBA AXI4-Stream Protocol Specification*
- AMBA AXI4 Protocol Specification

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 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
04/24/12	Draft	Xilinx Confidential DRAFT. Approved for external release under NDA only.
05/25/12	1.0	Initial Xilinx release.

Notice of Disclaimer