IP CORE MANUAL

AXI4-Stream Advance IP

px_axis_pdti_adv

PENTEK

Pentek, Inc.
One Park Way
Upper Saddle River, NJ 07458
(201) 818-5900
http://www.pentek.com/

Manual Part Number: 807.48319 Rev: 1.0 - December 09, 2016

Manual Revision History

Date	Version	Comments
12/09/16	1.0	Initial Release

Legal Notices

Copyright

Trademarks

Printed in the United States of America.

Table of Contents

Page

Table of Contents

IP Facts

Description

Pentek's Navigator™ AXI4-Stream Advance Core advances data, timestamp, information from a Combined Data/ Timestamp/ Information AXI4-Stream input to align with the gate/ sync/ PPS timing events. This is implemented by adding delay to the input AXI4-Stream (excluding bits corresponding to gate, sync, PPS positions) using an external Block RAM.

This core complies with the ARM® AMBA® AXI4 Specification and also provides a control/status register interface. This user manual defines the hardware interface, software interface, and parameterization options for the AXI4-Stream Advance Core.

Features

- Software programmable width of input data stream
- Supports up to 32 bytes wide input data streams
- Supports addition of control register which can be accessed through an AXI4-Lite Interface
- Width of delay vector can be user-defined
- Supports up to 16-bit wide delay control vector

Table 1-1: IP Facts Table

Core Specifics	
Supported Design Family[a]	Kintex® Ultrascale
Supported User Interfaces	AXI4-Lite and AXI4-Stream
Resources	See Table 2-1
Provided with the Core	
Design Files	VHDL
Example Design	Not Provided
Test Bench	VHDL
Constraints File	Not Provided[b]
Simulation Model	VHDL
Supported S/W Driver	HAL Software Support
Tested Design Flows	
Design Entry	Vivado® Design Suite 2016.3 or later
Simulation	Vivado VSim
Synthesis	Vivado Synthesis
Support	
Provided by Pentek fpgasupport@pentek.com	

a.For a complete list of supported devices, see the *Vivado Design Suite Release Notes*.
b.Clock constraints can be applied at the top level module of the user design.

This page is intentionally blank

Chapter 1: Overview

1.1 Functional Description

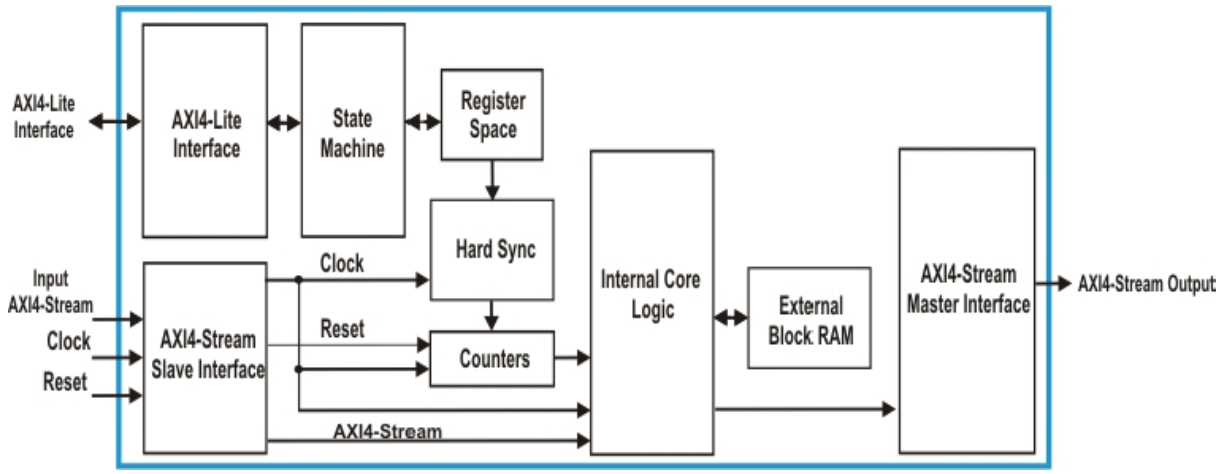
The AXI4-Stream Advance Core accepts a combined Data/ Timestamp/ Information AXI4-Stream, which includes sample data, timestamp with a time-aligned copy of the timing events (gate, sync, PPS), and data information (see Section 3.2). This core advances the data, time-stamp, and data information of the input, by adding delay to them, to align with the timing signals (gate, sync, PPS).

The delay to be introduced to the input can be defined by the user as a delay control input to the core or using a control register which can be accessed through an AXI4-Lite Interface. The generic parameter **has_csr** (see Table 2-2) is used to enable (or disable) the delay control register for delay control.

The AXI4-Stream Advance core also has a Xilinx® Hard Sync Synchronizer IP block which is used to synchronize the delay, from the delay control input or the delay control register, with the input AXI4-Stream clock (**s_axis_aclk**). This core uses an external Xilinx dual port Block RAM (BRAM) to introduce delay and store the data. The AXI4-Stream Advance Core generates the required control signals to the BRAM for read and write operations. The output of the core is obtained by reading from the External BRAM. The output of the core contains an AXI4-Stream with data, timestamp, data information aligned to the gate, sync, PPS events. The width of the input data stream can be defined using the generic parameter **data_byte_width** (see Section 2.5).

Figure 1-1 is a top-level block diagram of the Pentek AXI4-Stream Advance Core. The modules within the block diagram are explained in the later sections of this manual. This diagram shows an external BRAM linked to the core and has the control register enabled. When the control register is disabled the delay control input of the core is directly mapped to the Hard Sync input.

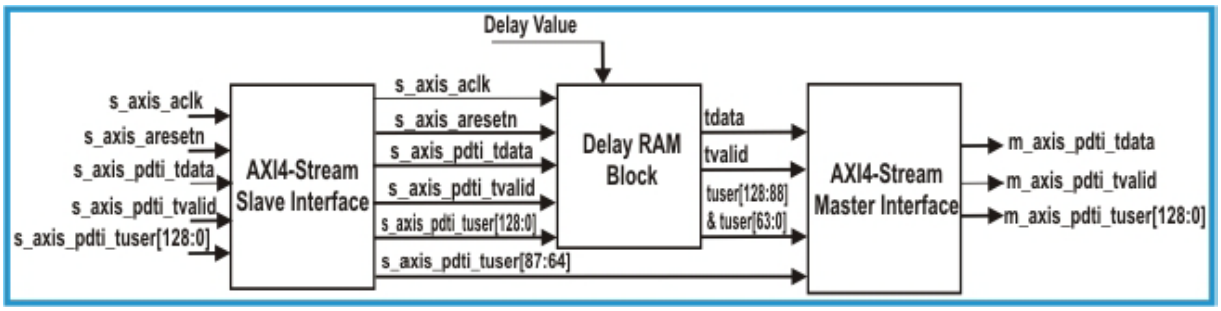
Figure 1-1: AXI4-Stream Advance Core Block Diagram



1.1 Functional Description (continued)

The Figure 1-2 is a more detailed block diagram showing the incoming AXI4-Stream signals and the generated output AXI4-Stream signals after passing through the Delay RAM module. It shows that only the data, timestamp, data information signals are delayed while the gate, sync, and PPS timing signals are directly driven to the output ports to obtain an AXI4-Stream output where data, timestamp and data information are aligned with the timing signals. For a detailed explanation of the AXI4-Stream signals refer to Section 3.2. The bits 87 to 64 of the sideband data (**s_axis_pdti_tuser**) correspond to the gate, sync, and PPS positions (see Table 3-2).The delay value input to the Delay RAM block comes from either the delay control input of the core or the control register based on the generic parameter **has_csr**.

Figure 1-2: AXI4-Stream Advance Core Block Diagram with Signals



- **AXI4-Stream Interface:** The AXI4-Stream Advance Core has two AXI4-Stream Interfaces. At the input, an AXI4-Stream Slave Interface is used to receive input AXI4-Streams and at the output an AXI4-Stream Master Interface is used to transfer AXI4-Streams through the output ports. For more details about the AXI4-Stream Interfaces please refer to **Section 3.2 AXI4-Stream Core Interfaces**.

- **AXI4-Lite Interface:** This is an optional module which implements a 32-bit AXI4-Lite Slave Interface to write and read from the control register within the Register Space of the core. This interface is enabled when the core has a control register, which is defined by the generic parameter **has_csr**. For additional details about the AXI4-Lite Interface, refer to Section 3.1 AXI4-Lite Core Interfaces.

- **Register Space:** The registers within the Register Space of the core can be enabled (or disabled) by the generic parameter **has_csr** (see Section 2.5). This space contains a control register which can be accessed through the AXI4-Lite Interface.

- **Hard Sync:** This is Xilinx Hard Sync Synchronizer IP block which is used to synchronize the delay from the control register or the delay control input of the core, to the AXI4-Stream clock (**s_axis_aclk**).

- **Counters:** This core includes two counters which count based on the delay value and are used to generate the required control outputs to the external BRAM.

1.2 Applications

The AXI4-Stream Advance Core can be incorporated into any Kintex Ultrascale FPGA where the data, timestamp, data information in an AXI4-Stream (PDTI format, see Section 3.2) need to be advanced in relation to the gate, sync and PPS timing events. It can also be used in applications where pre-triggering is required.

1.3 System Requirements

For a list of system requirements, see the Vivado Design Suite Release Notes.

1.4 Licensing and Ordering Information

This core is included with all Pentek Navigator FPGA Design Kits for Pentek Jade series board products. Contact Pentek for Licensing and Ordering Information (www.pentek.com).

1.5 Contacting Technical Support

Technical Support for Pentek's Navigator FPGA Design Kits is available via e-mail (fpgasupport@pentek.com) or by phone (201-818-5900 ext. 238, 9 am to 5 pm EST).

1.6 Documentation

This user manual is the main document for this IP core. The following documents provide supplemental material:

1) *Vivado Design Suite User Guide: Designing with IP*

2) *Vivado Design Suite User Guide: Programming and Debugging*

3) *ARM AMBA AXI4 Protocol Version 2.0 Specification*
http://www.arm.com/products/system-ip/amba-specifications.php

This page is intentionally blank

Chapter 2: General Product Specifications

2.1 Standards

The AXI4-Stream Advance Core has bus interfaces that comply with the A*RM AMBA AXI4-Lite Protocol Specification* and the *AMBA AXI4-Stream Protocol Specification*.

2.2 Performance

The performance of the AXI4-Stream Advance Core is limited only by the FPGA logic speed. The values presented in this section should be used as an estimation guideline. Actual performance can vary.

2.2.1 Maximum Frequencies

The AXI4-Stream Advance Core has two incoming clock signals. The input AXI4-Stream clock has a maximum frequency of 500 MHz while the clock across the AXI4-Lite interface has a maximum frequency of 250 MHz on a Kintex Ultrascale -2 speed grade FPGA. 250 MHz is typically the PCI Express (PCIe®) AXI bus clock frequency.

2.3 Resource Utilization

The resource utilization of the AXI4-Stream Advance Core is shown in Table 2-1. Resources have been estimated for the Kintex Ultrascale XCKU060 -2 speed grade device. These values were generated using the Vivado Design Suite.

Table 2-1: Resource Usage and Availability

Resource	# Used
LUTs	71
Flip-Flops	390
Memory LUTs	78

NOTE: Actual utilization may vary based on the user design in which the AXI4-Stream Advance Core is incorporated.

2.4 Limitations and Unsupported Features

This section is not applicable to this IP core.

2.5 Generic Parameters

The generic parameters of the AXI4-Stream Advance Core are described in Table 2-2. These parameters can be set as required by the user application while customizing the core.

Table 2-2: Generic Parameters

Port/Signal Name	Type	Description
data_byte_width	Integer	**Data Byte Width:** This parameter indicates the width of the input data streams across the AXI4-Stream Slave Interface of the core in bytes. It can range from 1 to 32 bytes.
delay_ctl_width		**Delay Control Width:** This parameter indicates the width of the delay control value of the core. This parameter also defines the BRAM address width since the delay control value is used with the counters to generate the RAM address for read and write operations.
has_csr	Boolean	**Has Control/Status Register:** This parameter is used to enable (or disable) the control register to control the delay of the core. When enabled, the core does not have a delay control input.

Chapter 3: Port Descriptions

This chapter provides details about the port descriptions for the following interface types:

- AXI4-Lite Core Interfaces
- AXI4-Stream Core Interfaces
- I/O Signals

3.1 AXI4-Lite Core Interfaces

The AXI4-Stream Advance Core uses the following AXI4-Lite Interface to access the control register within the core when the generic parameter **has_csr** is set to True.

3.1.1 Control/Status Register (CSR) Interface

The CSR interface is an AXI4-Lite Slave Interface that can be used to access the control register in the AXI4-Stream Advance Core. Table 3-1 defines the ports in the CSR interface. See Chapter 4 for a Control/Status Register memory map and bit definitions. See the *AMBA AXI4-Lite Specification* for more details on operation of the AXI4-Lite interfaces.

Table 3-1: Control/Status Register (CSR) Interface Port Descriptions

Port	Direction	Width	Description
s_axi_csr_aclk	Input	1	**Clock**
s_axi_csr_aresetn	Input	1	**Reset:** Active low. This signal will reset all control registers to their initial states.
s_axi_csr_awaddr	Input	3	**Write Address:** Address used for write operations. It must be valid when **s_axi_csr_awvalid** is asserted and must be held until **s_axi_csr_awready** is asserted by the AXI4-Stream Advance Core.
s_axi_csr_awprot	Input	3	**Protection:** The AXI4-Stream Advance Core ignores these bits.
s_axi_csr_awvalid	Input	1	**Write Address Valid:** This input must be asserted to indicate that a valid write address is available on **s_axi_csr_awaddr**. The AXI4-Stream Advance Core asserts **s_axi_csr_awready** when it is ready to accept the address. The **s_axi_csr_awvalid** must remain asserted until the rising clock edge after the assertion of **s_axi_csr_awready**.

Table 3-1: Control/Status Register (CSR) Interface Port Descriptions (Continued)			
Port	**Direction**	**Width**	**Description**
s_axi_csr_awready	Output	1	**Write Address Ready:** This output is asserted by the AXI4-Stream Advance Core when it is ready to accept the write address.The address is latched when **s_axi_csr_awvalid** and **s_axi_csr_awready** are high on the same cycle.
s_axi_csr_wdata	Input	32	**Write Data:** This data will be written to the address specified by **s_axi_csr_awaddr** when **s_axi_csr_wvalid** and **s_axi_csr _wready** are both asserted. The value must be valid when **s_axi_csr_wvalid** is asserted and held until **s_axi_csr_wready** is also asserted.
s_axi_csr_wstrb	Input	4	**Write Strobes:** This signal, when asserted, indicates the number of bytes of valid data on the **s_axi_csr_wdata** signal. Each of these bits, when asserted, indicate that the corresponding byte of **s_axi_csr_wdata** contains valid data. Bit 0 corresponds to the least significant byte, and bit 3 to the most significant.
s_axi_csr_wvalid	Input	1	**Write Valid:** This signal must be asserted to indicate that the write data is valid for a write operation. The value on **s_axi_csr _wdata** is written into the register at address **s_axi_csr_awaddr** when **s_axi_csr_wready** and **s_axi_csr_wvalid** are high on the same cycle.
s_axi_csr_wready	Output	1	**Write Ready:** This signal is asserted by the AXI4-Stream Advance Core when it is ready to accept data. The value on **s_axi_csr _wdata** is written into the register at address **s_axi_csr_awaddr** when **s_axi_csr_wready** and **s_axi_csr_wvalid** are high on the same cycle, assuming that the address has already or simultaneously been submitted.
s_axi_csr_bresp	Output	2	**Write Response:** The AXI4-Stream Advance Core indicates success or failure of a write transaction through this signal, which is valid when **s_axi_csr_bvalid** is asserted; 00 = Success of normal access 01 = Success of exclusive access 10 = Slave Error 11 = Decode Error Note: For more details about this signal refer to the *AMBA AXI Specification*.
s_axi_csr_bready	Input	1	**Write Response Ready:** This signal must be asserted by the user logic when it is ready to accept the Write Response.
s_axi_csr_bvalid	Output	1	**Write Response Valid:** This signal is asserted by the AXI4-Stream Advance Core when the write operation is complete and the Write Response is valid. It is held until **s_axi_csr_bready** is asserted by the user logic.

Table 3-1: Control/Status Register (CSR) Interface Port Descriptions (Continued)

Port	Direction	Width	Description
s_axi_csr_araddr	Input	3	**Read Address:** Address used for read operations. It must be valid when **s_axi_csr_arvalid** is asserted and must be held until **s_axi_csr_arready** is asserted by the AXI4-Stream Advance Core.
s_axi_csr_arprot	Input	3	**Protection:** These bits are ignored by the AXI4-Stream Advance Core
s_axi_csr_arvalid	Input	1	**Read Address Valid:** This input must be asserted to indicate that a valid read address is available on the **s_axi_csr_araddr**. The AXI4-Stream Advance Core asserts **s_axi_csr_arready** when it ready to accept the Read Address. This input must remain asserted until the rising clock edge after the assertion of **s_axi_csr_arready**.
s_axi_csr_arready	Output	1	**Read Address Ready:** This output is asserted by the AXI4-Stream Advance Core when it is ready to accept the read address. The address is latched when **s_axi_csr_arvalid** and **s_axi_csr_ arready** are high on the same cycle.
s_axi_csr_rdata	Output	32	**Read Data:** This value is the data read from the address specified by the **s_axi_csr_araddr** when **s_axi_csr_arvalid** and **s_axi_csr_arready** are high on the same cycle.
s_axi_csr_rresp	Output	2	**Read Response:** The AXI4-Stream Advance Core indicates success or failure of a read transaction through this signal, which is valid when **s_axi_csr_rvalid** is asserted; 00 = Success of normal access 01 = Success of exclusive access 10 = Slave Error 11 = Decode Error Note: For more details about this signal refer to the *AMBA AXI Specification*.
s_axi_csr_rvalid	Output	1	**Read Data Valid:** This signal is asserted by the AXI4-Stream Advance Core when the read is complete and the read data is available on **s_axi_csr_rdata**. It is held until **s_axi_csr_ rready** is asserted by the user logic.
s_axi_csr_rready	Input	1	**Read Data Ready:** This signal is asserted by the user logic when it is ready to accept the Read Data.

3.2 AXI4-Stream Core Interfaces

The AXI4-Stream Advance Core has the following AXI4-Stream Interfaces, used to receive and transfer data streams.

- Combined Sample Data/ Timestamp/ Information Stream (PDTI) Interface: This core implements two of these AXI4-Stream interfaces across the input and output to receive and transfer AXI4-Streams.

3.2.1 Combined Sample Data/ Timestamp/ Information Streams (PDTI) Interface

The Pentek Jade series board products have AXI4-Streams that follow a combined sample data/ Timestamp/ Information Stream format. This type of data stream combines sample data with its time-aligned timestamp and data information. There is an AXI4-Stream Slave Interface across the input to receive AXI4-Streams and an AXI4-Stream Master Interface across the output to transfer AXI4-Streams.

Table 3-2 defines the ports in the AXI4-Stream Slave and Master Combined Sample Data/ Timestamp/ Information Stream Interfaces. See the *AMBA AXI4-Stream Specification* for more details on the operation of the AXI4-Stream Interface..

Table 3-2: Combined Sample Data/ Timestamp/ Information Streams Interface Port Descriptions

Port	Direction	Width	Description
AXI4-Stream Slave Interface			
s_axis_aclk	Input	1	**AXI4-Stream Clock**
s_axis_aresetn			**Reset:** Active Low.
s_axis_pdti_tdata		depends on the generic parameter **data_byte_width**	**Input Data:** This is the input data stream.

Table 3-2: Combined Sample Data/ Timestamp/ Information Streams Interface Port Descriptions (Continued)

Port	Direction	Width	Description
AXI4-Stream Slave Interface (continued)			
s_axis_pdti_tvalid	Input	1	**Input Data Valid:** Asserted when data is valid on **s_axis_pdti_tdata**.
s_axis_pdti_tuser		128	**Sideband Information:** This is the user defined sideband information received alongside the data stream. **tuser [63:0] - Timestamp[63:0]** **tuser [71:64] - Gate Positions** **tuser [79:72] - Sync Positions** **tuser [87:80] - PPS Positions** **tuser [92:88] - Samples per clock cycle** **tuser [94:93] - Data Format** **=> 0 = 8-bit; 1 = 16-bit;** **2 = 24-bit; 3 = 32-bit** **tuser [95] - Data Type => 0 = Real; 1 = I/Q** **tuser [103:96] - channel [7:0]** **tuser [127:104] - Reserved** Note: The bits [103:96] define the channel number in the user design from where the data is being received.
AXI4-Stream Master Interface			
m_axis_pdti_tdata	Output	depends on the generic parameter **data_byte_width**	**Output Data:** This is the output data from the AXI4-Stream Advance Core.
m_axis_pdti_tvalid		1	**Output Data Valid:** Asserted when data is valid on **m_axis_pdti_tdata**.
m_axis_pdti_tuser		128	**Output Sideband Information:** This is the user defined sideband information transmitted alongside the data stream. **tuser [63:0] - Timestamp[63:0]** **tuser [71:64] - Gate Positions** **tuser [79:72] - Sync Positions** **tuser [87:80] - PPS Positions** **tuser [92:88] - Samples per clock cycle** **tuser [94:93] - Data Format** **=> 0 = 8-bit; 1 = 16-bit;** **2 = 24-bit; 3 = 32-bit** **tuser [95] - Data Type => 0 = Real; 1 = I/Q** **tuser [103:96] - channel [7:0]** **tuser [127:104] - Reserved** Note: The bits [103:96] define the channel number in the user design from where the data is being received.

3.3 I/O Signals

The I/O port/signal descriptions of the top level module of the AXI4-Stream Advance core are discussed in Table 3-3..

Table 3-3: I/O Signals

Port/ Signal Name	Type	Direction	Description
delay_ctl [delay_ctl_width-1:0]	std_logic _vector	Input	**Delay Control Input:** This input defines the delay to be introduced to the input AXI4-Stream. This input is present when the generic parameter **has_csr** is set to False.
bram_wrport_rst	std_logic	Output	**BRAM Write Port Reset:** Active High, write port reset for the Block RAM.
bram_wrport_clk			**BRAM Write Port Clock:** This is the clock signal for the write port of the Block RAM. Connected to **s_axi_aclk.**
bram_wrport_en			**BRAM Write Port Enable:** When set to High, this signal enables the write port of the BRAM.
bram_wrport_we [(data_byte_width +15):0]	std_logic _vector		**BRAM Write Enable:** This signal when set to High, enables write operation across the BRAM write port.
bram_wrport_addr [delay_ctl_width-1:0]			**BRAM Write Port Address:** These bits indicate the memory address within the Block RAM for write operations.
bram_wrport_wrdata [(((data_byte_ width*8)-1)+128):0]	std_logic _vector	Output	**BRAM Write Data:** This is the data output of the AXI4-Stream Advance core to be written into the Block RAM memory at address **bram_wrport_addr**.
bram_rdport_rst	std_logic		**BRAM Read Port Reset:** Active High, read port reset for the Block RAM.
bram_rdport_clk			**BRAM Read Port Clock:** This is the clock signal for the read port of the Block RAM. Connected to **s_axi_aclk.**
bram_rdport_en			**BRAM Read Port Enable:** When set to High, this signal enables the read port of the BRAM.
bram_rdport_addr [delay_ctl_width-1:0]	std_logic _vector		**BRAM Read Port Address:** These bits indicate the memory address within the Block RAM for read operations.
bram_rdport_rddata [(((data_byte_ width*8)-1)+128):0]		Input	**BRAM Read Data:** This is data input of the AXI4-Stream Advance core from the BRAM for a memory read request operation.

Chapter 4: Register Space

This chapter provides the memory map and register descriptions for the register space of the AXI4-Stream Advance Core. The memory map is provided in Table 4-1.

Table 4-1: Register Space Memory Map

Register Name	Address (Base Address +)	Access	Description
Control Register	0x00	R/W	Controls delay to be introduced to the input data.

4.1 Control Register

This register control the delay to be introduced to the input AXI4-Stream of the core. The value in the delay control value bits of this control register is used as the input delay value when the generic parameter **has_csr** is set to True. The Control Register is illustrated in Figure 4-1 and described in Table 4-2.

Figure 4-1: Control Register

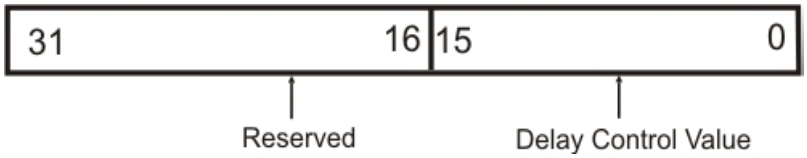


Table 4-2: Control Register (Base Address + 0x00)

Bits	Field Name	Default Value	Access Type	Description
31:16	Reserved	N/A	N/A	**Reserved**
15:0	delay_ctl	0x0000	R/W	**Delay Control Value:** These bits define the delay to be introduced to the input AXI4-Stream.

This page is intentionally blank

Chapter 5: Designing with the Core

This chapter includes guidelines and additional information to facilitate designing with the AXI4-Stream Advance Core.

5.1 General Design Guidelines

The AXI4-Stream Advance Core provides the required logic to generate an AXI4-Stream output, which has data, timestamp and data information aligned to the timing signals (gate, sync and PPS), from incoming AXI PDTI Streams (see Section 3.2). This core works in conjunction with an external Xilinx dual port Block RAM core in the user design. This IP core supports AXI4-Lite and AXI4-Stream user interfaces. The user can customize the core by setting the generic parameters based on the application requirement as described in Section 2.5. A Xilinx dual port BRAM has read latency of 2 clock cycles which should be considered while defining the generic parameters.

5.2 Clocking

AXI4-Stream Clock: **s_axis_aclk**

This clock is used to clock all ports in the AXI4-Stream Advance core.

CSR Clock: **s_axi_csr_aclk**

This clock is used to clock the AXI4-Lite interface of the core.

5.3 Resets

Main reset: **s_axis_aresetn**

This is an active low reset synchronous with **s_axis_aclk**.

CSR Reset: **s_axi_csr_aresetn**

This is an active low reset synchronous with **s_axi_csr_clk**.

5.4 Interrupts

This section is not applicable to this IP core.

5.5 Interface Operation

CSR Interface: This is the Control/Status Register Interface and is associated with **s_axi_csr_aclk**. It is a standard AXI4-Lite Slave Interface. See Chapter 4 for the control register memory map, which provides more details on the registers that can be accessed through this interface.

Combined Sample Data/ Timestamp/ Information Streams (PDTI) Interfaces: This core implements two of these AXI4-Stream interfaces across the input and output to receive, and transfer AXI PDTI streams, and is associated with **s_axis_aclk**. For more details about this interface please refer to Section 3.2.1.

5.6 Programming Sequence

This section briefly describes the programming sequence of registers in the AXI4-Stream Advance Core.

1) Assign desired values to the generic parameters.

2) Set the control register with the required value if the control register is enabled.

3) Include a BRAM and map the AXI4-Stream Advance Core to the BRAM.

4) Observe the outputs across the output ports.

5.7 Timing Diagrams

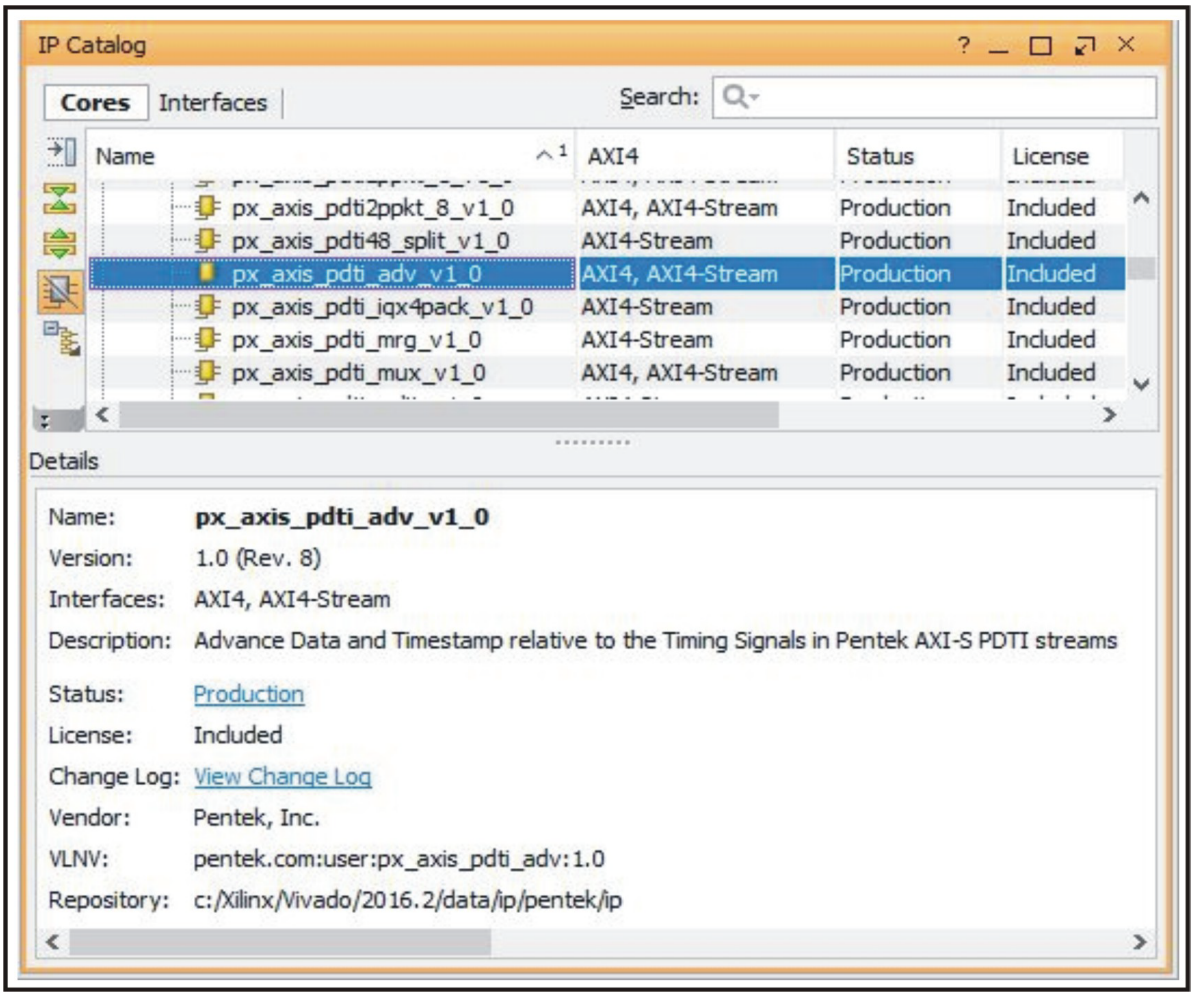
The timing diagrams for the AXI4-Stream Advance Core are obtained by running the simulation of the test bench of the core in Vivado VSim environment. For more details about the test bench, refer to Section 6.5.

Chapter 6: Design Flow Steps

6.1 Pentek IP Catalog

This chapter describes customization and generation of the Pentek AXI4-Stream Advance Core. It also includes simulation, synthesis, and implementation steps that are specific to this IP core. This core can be generated from the Vivado IP Catalog when the Pentek IP Repository has been installed. It will appear in the IP Catalog list as **px_axis_pdti_adv_v1_0** as shown in Figure 6-1.

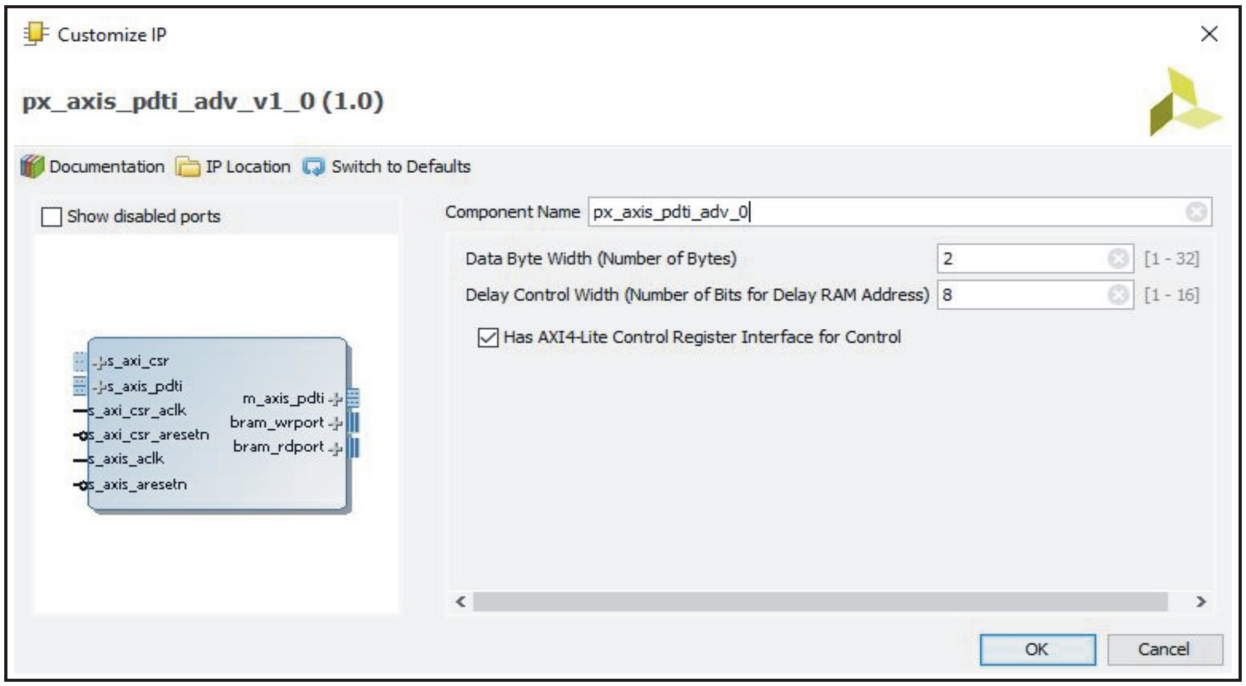
Figure 6-1: AXI4-Stream Advance Core in Pentek IP Catalog



6.1 Pentek IP Catalog (continued)

When you select the **px_axis_pdti_adv_v1_0** core, a screen appears that shows the core's symbol and the core's parameters (see Figure 6-2). The core's symbol is the box on the left side.

Figure 6-2: AXI4-Stream Advance Core IP Symbol



6.2 User Parameters

The user parameters of this IP core are described in Section 2.5 of this user manual.

6.3 Generating Output

For more details about generating and using IP in the Vivado Design Suite, refer to the *Vivado Design Suite User Guide - Designing with IP.*

6.4 Constraining the Core

This section contains information about constraining the AXI4-Stream Advance Core in Vivado Design Suite.

Required Constraints

The XDC constraints are not provided with the AXI4-Stream Advance Core. Clock constraints can be applied in the top-level module of the user design.

Device, Package, and Speed Grade Selections

This IP works for the Kintex Ultrascale FPGAs.

Clock Frequencies

The clock (**s_axi_csr_aclk**) can take frequencies up to 250 MHz. The AXI4-Stream clock (**s_axis_aclk**) has a maximum frequency of 500 MHz.

Clock Management

This section is not applicable for this IP core.

Clock Placement

This section is not applicable for this IP core.

Banking and Placement

This section is not applicable for this IP core.

Transceiver Placement

This section is not applicable for this IP core.

I/O Standard and Placement

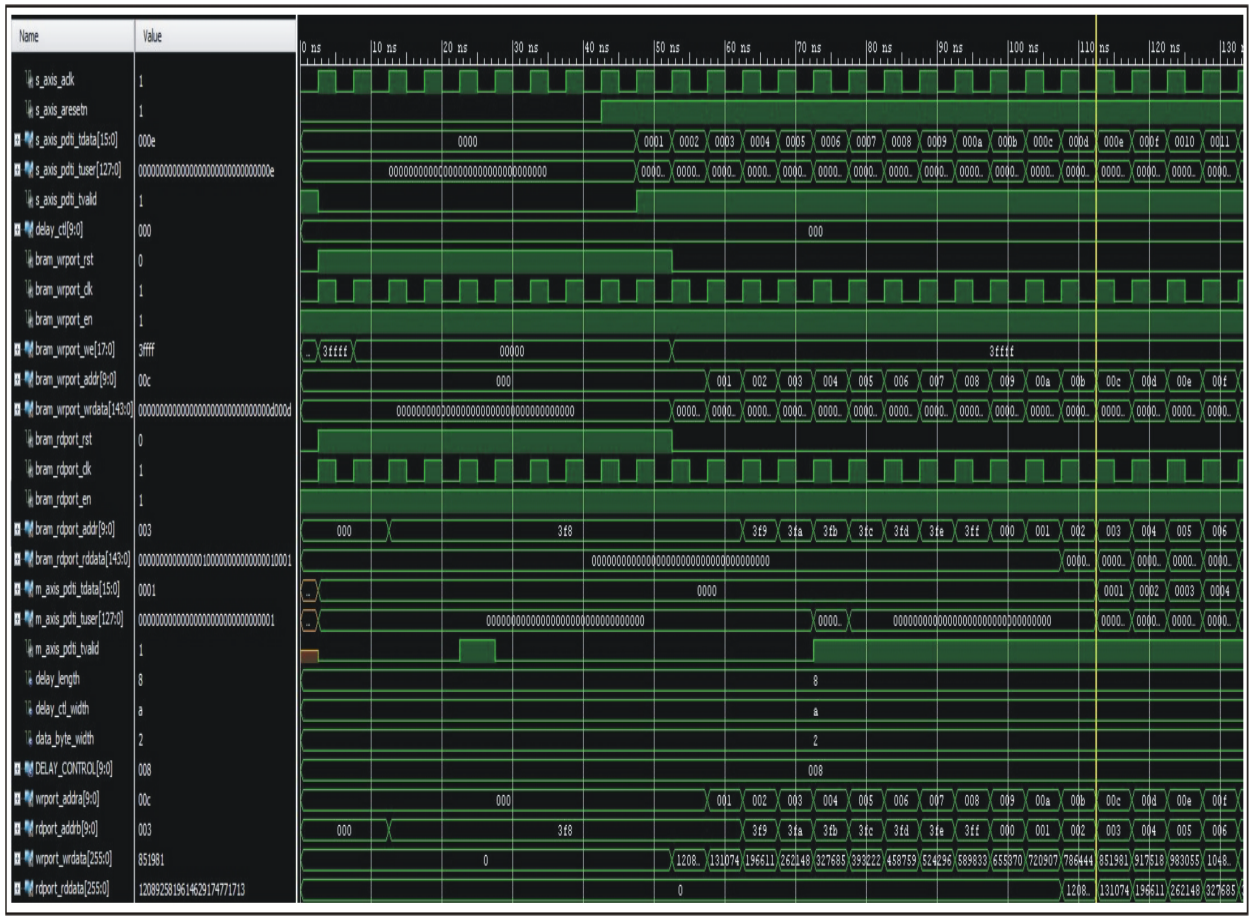
This section is not applicable for this IP core.

6.5 Simulation

The AXI4-Stream Advance Core has a test bench which generates output waveforms using the Vivado VSim environment. The test bench is designed to run at 200 MHz AXI4-Stream clock frequency. It sets a data width of 2 bytes across the input AXI4-Stream Slave Interface and does not have a control register.

It has a delay control input with a width of 10 bits and introduces an integer delay of 10 clock cycles. The test bench generates incoming data stream through an increasing counter and has the least significant 16 bits of sideband data (**s_axis_pdti_tuser**) equal to the input data (**s_axis_pdti_tdata**). The generated BRAM control outputs of the AXI4-Stream Advance Core are mapped to an external dual port BRAM included in the test bench. The BRAM has a 2 clock cycles of read latency. The programming procedure is the same as described in Section 5.6. When run, the simulation produces the results shown in Figure 6-3.

Figure 6-3: AXI4-Stream Advance Core Test Bench Simulation Output



6.6 Synthesis and Implementation

For details about synthesis and implementation see the *Vivado Design Suite User Guide - Designing with IP*.

This page is intentionally blank