IP CORE MANUAL

AXI4-Stream Bus Traffic Meter IP

px_axis_traffic_meter

PENTEK

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

Manual Part Number: 807.48316 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 Bus Traffic Meter Core measures the data rate (in bytes-per-second) of the AXI4-Stream traffic through the core.

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 Bus Traffic Meter Core.

Features

- Register access through AXI4-Lite interface
- Supports divide factor to keep the resulting transfer rate to a 32-bit value
- User-programmable initial clock frequency, number of bytes in the AXI4-Stream data, divide factor, and width of AXI4-Stream user data
- Provides register access to control initial clock frequency of the core

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 Bus Traffic Meter Core accepts input data AXI4-Streams and maps them to the output data ports with no delay. The core includes counters to calculate the data rate of the AXI4-Stream data when it is valid and the ready signal, indicating that the user design is ready to accept data, is High.

The clock frequency value at reset can be defined by the user through the generic parameter **init_clk_freq** (see Table 2-2). This value is used to determine the number of clock cycles per second based on which the data rate is calculated. This value is typically set to the frequency of the incoming AXI Stream clock signal.

The AXI4-Stream Bus Traffic Meter core includes a Register Space with control/status registers which can be accessed using an AXI4-Lite Slave interface. The initial clock frequency value can be reset by changing the value of the Frequency Value Control Register (see Section 4.1). The data rate of the AXI Stream can be obtained by reading from the Bytes per Second Reading Status Register (see Section 4.2).

The AXI4-Stream Bus Traffic Meter Core has a generic parameter named divide factor which is used to keep the resultant data rate to a 32-bit value. When the data rate exceeds 4 GBps, the divide factor must be set accordingly in order to divide the rate to obtain a 32-bit value. The value read from the status register must then be multiplied with the divide factor to obtain the true data rate of the AXI4-Stream. The data rate value is updated once every second as determined by the clock frequency value in the control register. The input AXI4-Stream data width can be defined by the user through the generic parameter **num_bytes** (see Table 2-2). The AXI4-Stream Bus Traffic Meter Core has an AXI Clock Converter Core which is connected to the AXI4-Lite Interface in order to operate the Register Space in the AXI Stream Clock domain. The core also has a Clock Domain Crossing Core in order to operate the reset input across the AXI4-Lite interface in the AXI Stream Clock domain.

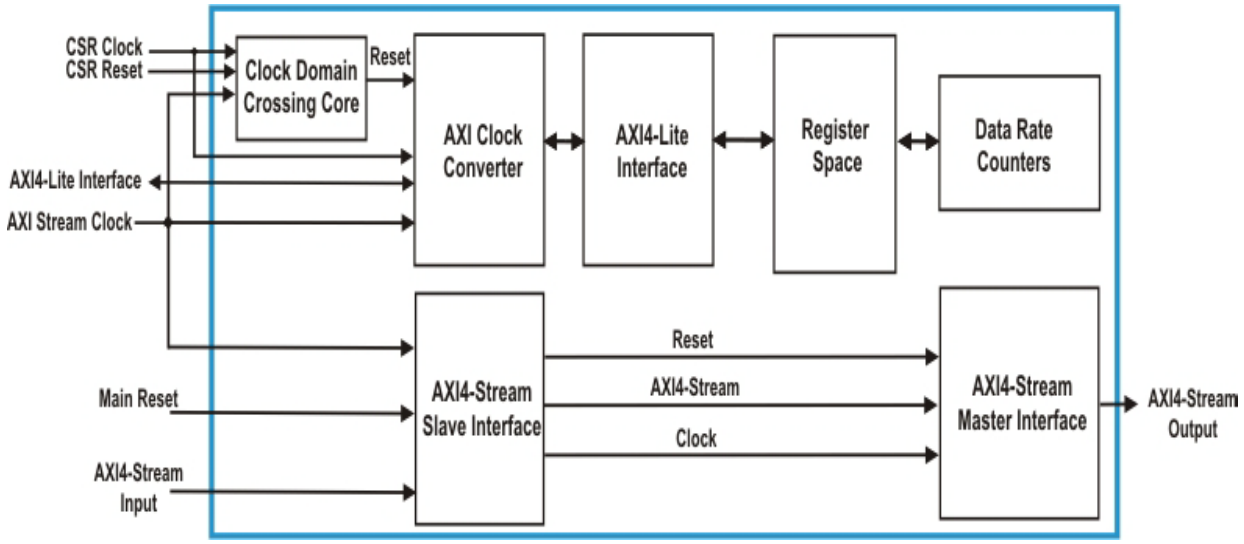
Figure 1-1 is a top-level block diagram of the Pentek AXI4-Stream Bus Traffic Meter Core. The modules within the block diagram are explained in the later sections of this manual.

- **AXI Clock Converter Core:** The AXI Clock Converter Core is included in the Xilinx AXI Interconnect Core and is used to connect one AXI memory-mapped slave to an AXI memory-mapped master which is operating in a different clock domain. In the AXI4-Stream Bus Traffic Meter Core, the AXI Clock Converter is used to operate the Register Space in the AXI Stream clock domain (**axis_aclk**).

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

1.1 Functional Description (continued)

Figure 1-1: AXI4-Stream Bus Traffic Meter Core Block Diagram



- **AXI4-Lite Interface:** This module implements a 32-bit AXI4-Lite Slave interface to access the Register Space. For additional details about the AXI4-Lite Interface, refer to Section 3.1 AXI4-Lite Core Interfaces.
- **Register Space:** This module contains the control and status registers of the core. Registers are accessed through the AXI4-Lite interface.
- **Data Rate Counters:** This module includes two counters used to calculate the data rate of the AXI4-Stream bus.

1.2 Applications

The AXI4-Stream Bus Traffic Meter Core can be used to determine the data rate of AXI4-Stream data and can be incorporated into any Kintex Ultrascale FPGA.

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 Bus Traffic Meter 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 Bus Traffic Meter 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 Bus Traffic Meter Core has two incoming clock signals. The input AXI Stream clock has a maximum frequency of 250MHz and the clock across the AXI4-Lite interface also has a maximum frequency of 250MHz on a Kintex Ultra-scale -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 Bus Traffic Meter 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	391
Flip-Flops	1002
Memory LUTs	68

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

2.4 Limitations and Unsupported Features

- This core does not suport counting the **tkeep** bits (see Table 3-2) to accurately detect only valid bytes. It assumes all bytes are valid when **tvalid** is asserted, which impacts the accuracy of the resultant data rate value.

2.5 Generic Parameters

The generic parameters of the AXI4-Stream Bus Traffic Meter 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
init_clk_freq	Integer	**Initial Clock Frequency at Reset:** This parameter defines the value of the initial clock frequency at reset based on which the number of clock cycles per second is calculated and the data rate is determined. It is defined in Hertz. It can range from 1 MHz to 500 MHz.
div_factor		**Divide Factor:** This parameter defines the divide factor of the core by which the data rate is divided to keep it to a 32-bit value. Set to 2 when the resultant rate is greater than or equal to 4 GBps and set to 4 when it is greater than or equal to 8 GBps.
num_bytes		**Number of Data Bytes:** This parameter defines the width of the data bus of the input AXI4-Stream in bytes. It can take the values 1, 2, 4, 8, 16, 32, and 64.
has_tuser	Boolean	**Has User Data:** This parameter indicates whether the incoming AXI Stream has sideband user information (**s_axis_tuser**).
has_tlast		**Has Data Last:** This parameter is used to indicate whether the incoming AXI Stream has the Data Last (**s_axis_tlast**) signal. Data Last indicates the boundary of the data packet.
has_tkeep		**Has Data Keep:** This parameter is used to indicate whether the incoming AXI Stream has the Data Keep (**s_axis_tkeep**) signal. Data Keep is a byte qualifier which indicates whether the data in the associated byte of AXI Stream data (**s_axis_tdata)** is processed as a part of the data stream.
has_tready		**Has Data Ready:** This parameter is used to indicate whether the incoming AXI Stream has the Data Ready input from the slave receiving the output AXI4-Stream data of this core. It also enables an output data ready signal from the Traffic Meter Core to the user design.
num_user_bits	Integer	**Number of User Bits:** This parameter defines the width (in bits) of the user sideband information (**s_axis_tuser**) that is transmitted alongside the data stream.

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

3.1 AXI4-Lite Core Interfaces

The AXI4-Stream Bus Traffic Meter Core uses the Control/Status Register (CSR) interface to control, and receive status from, the user design.

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 and status registers in the AXI4-Stream Bus Traffic Meter 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	7	**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 Bus Traffic Meter Core.
s_axi_csr_awprot	Input	3	**Protection:** The AXI4-Stream Bus Traffic Meter 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 Bus Traffic Meter 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 Bus Traffic Meter 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 Bus Traffic Meter 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 Bus Traffic Meter 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 Bus Traffic Meter 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	7	**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 Bus Traffic Meter Core.
s_axi_csr_arprot	Input	3	**Protection:** These bits are ignored by the AXI4-Stream Bus Traffic Meter 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 Bus Traffic Meter 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 Bus Traffic Meter 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 Bus Traffic Meter 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 Bus Traffic Meter 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 Traffic Meter core implements two AXI4-Stream core interfaces across the input and output to receive and transfer data streams. An AXI4-Stream Slave Interface at the input is used to receive data streams across the input ports. An AXI4-Stream Master Interface at the output is used to transfer data streams across the output ports. Table 3-2 defines the ports in the AXI4-Stream Slave and Master interfaces of the AXI4-Stream Bus Traffic Meter core. See the *AMBA AXI4-Stream Specification* for more details on the operation of the AXI4-Stream Interface.

Table 3-2: AXI4-Stream Interface Port Descriptions

Port	Direction	Width	Description
AXI4-Stream Slave Interface			
axis_aclk	Input	1	**AXI Stream Clock**
axis_aresetn			**Reset:** Active Low.
s_axis_tdata		depends on the generic parameter **num_bytes**	**Input Data**
s_axis_tvalid		1	**Input Data Valid:** This signal is asserted by the user logic when data is valid on **s_axis_tdata** bus.
s_axis_tready	Output		**Data Ready**: Active High. This is an output **tready** signal generated by the Traffic Meter core indicating that it is ready to accept data. This output can be enabled by setting the generic parameter **has_tready** to True. When **has_tready** is True, data is transferred across the AXI Stream Slave Interface when both **s_axis_tvalid** and **s_axis_tready** are High on the same cycle.
s_axis_tuser	Input	depends on the generic parameter **num_user_bits**	**Sideband Data:** This is input user defined sideband information which is transmitted alongside the data stream. This input when available to the core can be enabled by setting the generic parameter **has_tuser** to True.

Table 3-2: AXI4-Stream Interface Port Descriptions (Continued)

Port	Direction	Width	Description
AXI4-Stream Slave Interface (continued)			
s_axis_tkeep	Input	depends on the generic parameter **num_bytes**	**Data Keep:** This is an byte qualifier signal and is valid only when the generic parameter **has_tkeep** is set to True. Each bit of this signal corresponds to a byte in **s_axis_tdata** i.e., bit 0 corresponds to the least significant byte of **s_axis_tdata** and the most significant bit to the most significant byte. When a bit is asserted, the data on **s_axis_tdata** is considered valid. All **s_axis_tkeep** bits must be '1' contiguously until **s_axis_tlast** is asserted. When **s_axis_tlast** is asserted, and the number of data samples is not a multiple of **tdata** width, **tkeep** its are set to '0' to indicate which data bytes are to be ignored.
s_axis_tlast		1	**Data Last:** This input signal is valid when the generic parameter **has_tlast** is set to True. When asserted, **s_axis_tlast** marks the last data in the current data frame.
AXI4-Stream Master Interface			
m_axis_tdata	Output	depends on the generic parameter **num_bytes**	**Output Data**
m_axis_tvalid		1	**Output Data Valid:** This signal is asserted when data is valid on **m_axis_tdata** bus.
m_axis_tready	Input		**Data Ready**: Active High. This is an input **tready** signal from an AXI Stream slave in the user design indicating that it is ready to accept data. This input can be enabled by setting the generic parameter **has_tready** to True. When **has_tready** is True, data is transferred across the AXI Stream Master interface when both **m_axis_tvalid** and **m_axis_tready** are High on the same cycle. If the slave in the user design deasserts the ready signal when **m_axis_tvalid** is High, the core maintains the data on the bus and keeps the valid signal asserted until the slave has asserted the ready signal.
m_axis_tuser	Output	depends on the generic parameter **num_user_bits**	**Sideband Data:** This is user defined sideband output information transmitted alongside the data stream. This output can be enabled by setting the generic parameter **has_tuser** to True.

Table 3-2: AXI4-Stream Interface Port Descriptions (Continued)

Port	Direction	Width	Description
AXI4-Stream Master Interface (continued)			
m_axis_tkeep	Output	depends on the generic parameter **num_bytes**	**Data Keep:** This is an byte qualifier signal and is valid only when the generic parameter **has_tkeep** is set to True. Each bit of this signal corresponds to a byte in **m_axis_tdata** i.e., bit 0 corresponds to the least significant byte of **m_axis_tdata** and the most significant bit to the most significant byte. When a bit is asserted, the data on **m_axis_tdata** is considered valid. All **m_axis_tkeep** bits must be '1' contiguously until **m_axis_tlast** is asserted. When **m_axis_tlast** is asserted, and the number of data samples is not a multiple of **tdata** width, **tkeep** its are set to '0' to indicate which data bytes are to be ignored.
m_axis_tlast		1	**Data Last:** This signal is valid when the generic parameter **has_tlast** is set to True. When asserted, **m_axis_tlast** marks the last data in the current data frame.

Chapter 4: Register Space

This chapter provides the memory map and register descriptions for the Register Space of the AXI4-Stream Bus Traffic Meter Core. The memory map is provided in Table 4-1.

Table 4-1: Register Space Memory Map

Register Name	Address (Base Address +)	Access	Description
Frequency Value	0x00	R/W	Controls the clock frequency value of the core.
Bytes per Second Reading	0x04	R	Indicates the bytes per second data rate value.
Divide Factor	0x08		Indicates the value of the divide factor of the core.

4.1 Frequency Value Register

This register controls the clock frequency of the AXI4-Stream Bus Traffic Meter Core, based on which the data rate is calculated. The Frequency Value Register is illustrated in Figure 4-1 and described in Table 4-2.

Figure 4-1: Frequency Value Register

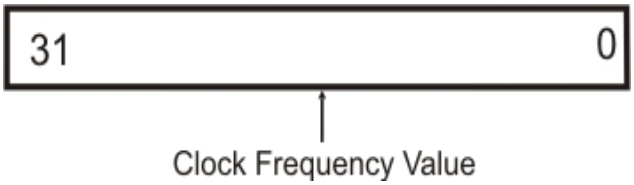


Table 4-2: Frequency Value Register (Base Address + 0x00)

Bits	Field Name	Default Value	Access Type	Description
31:0	freq_val	defined by the generic parameter **init_clk_freq**	R/W	**Clock Frequency Value:** These bits control the clock frequency value of the core used in the calculation of the data rate.

4.2 Bytes per Second Reading Register

This is a status register which indicates the data rate of the AXI4-Stream. This value is updated every second based on the clock frequency defined in the Frequency Value Control Register. The Bytes per Second Reading Register is illustrated in Figure 4-2 and described in Table 4-3.

Figure 4-2: Bytes per Second Reading Register

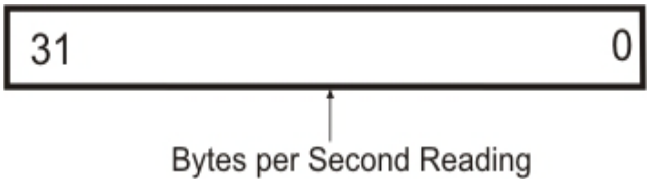


Table 4-3: Bytes per Second Reading Register (Base Address + 0x04)

Bits	Field Name	Default Value	Access Type	Description
31:0	data_rate	0x00000000	R	**Bytes per Second Reading:** These bits indicate the data rate of the AXI4-Stream in bytes per second.

4.3 Divide Factor Register

This is a status register which indicates the divide factor of the core by which the data rate is divided to keep it to a 32-bit value. This value is defined by the user through the generic parameter **div_factor** (see Section 2.5). The Divide Factor Register is illustrated in Figure 4-2 and described in Table 4-3.

Figure 4-3: Divide Factor Register

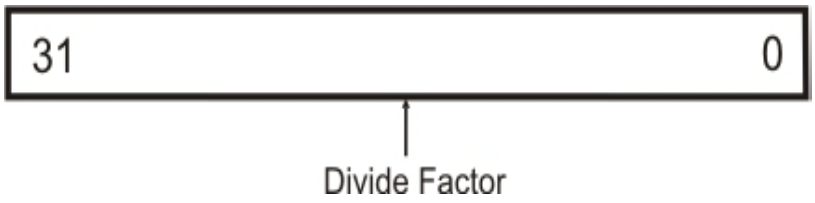


Table 4-4: Divide Factor Register (Base Address + 0x08)

Bits	Field Name	Default Value	Access Type	Description
31:0	div_factor	defined by the generic parameter **div_factor**	R	**Divide Factor:** This is the value of the divide factor by which the data rate is divided to keep it to a 32-bit value.

Chapter 5: Designing with the Core

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

5.1 General Design Guidelines

The AXI4-Stream Bus Traffic Meter Core provides the required logic for calculating data rate of the AXI Stream data. This IP core supports AXI4-Lite and AXI4-Stream user interfaces. The user can customize the core by setting the generic parameters, as described in Section 2.5. The core also has a control register which is used to control the clock frequency value of the core (see Section 4.1).

5.2 Clocking

AXI Stream Clock: **axis_aclk**

This clock is used to clock all ports of the AXI4-Stream Bus Traffic Meter core.

CSR Clock: **s_axi_csr_aclk**

This clock is the input AXI4-Lite interface clock to the core which is converted using the AXI Clock Converter Core to operate the other modules within the Traffic Meter core in the AXI Stream clock domain.

5.3 Resets

Main reset: **axis_aresetn**

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

CSR Reset: **s_axi_csr_aresetn**

This is an active low reset synchronous with **s_axi_csr_clk**. The CSR Interface of the core runs on the CSR reset which is in the AXI Stream clock domain. When asserted, the control/status registers are reset.

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 **axis_aclk**. It is a standard AXI4-Lite Slave Interface. See Chapter 4 for the control/ status register memory map, which provides more details on the registers that can be accessed through this interface.

AXI4-Stream Interfaces: This core has AXI4-Stream Slave and Master Interfaces at the input and output respectively to receive and transfer data streams as described in Section 3.2.

5.6 Programming Sequence

This section briefly describes the programming sequence of registers in the AXI4-Stream Bus Traffic Meter Core.

1) Assign the desired values to the generic paramters.

2) Set the Frequency Value Register with the required value.

3) Read data from the Bytes per Second Reading Register for the data rate value.

5.7 Timing Diagrams

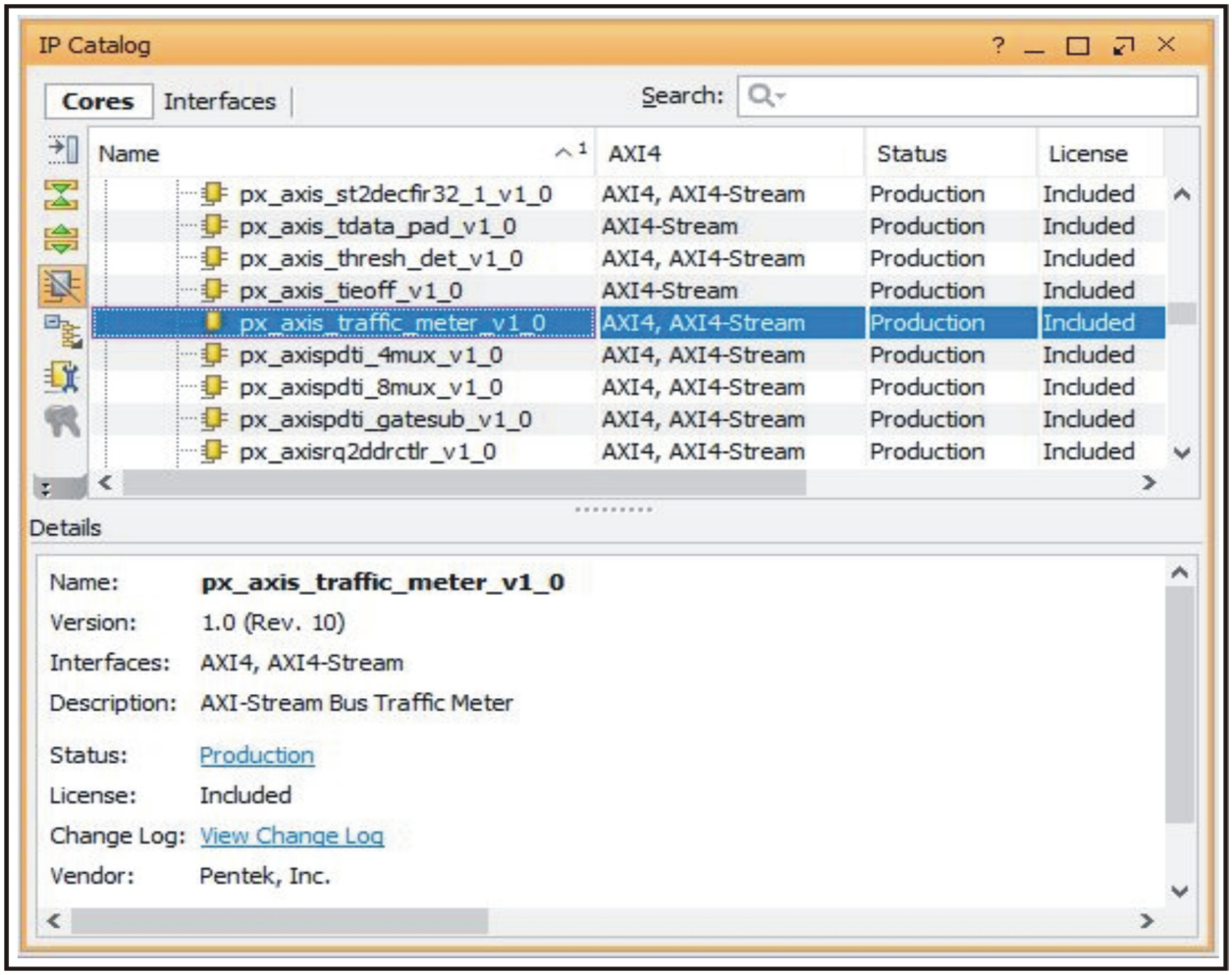
The timing diagrams for the AXI4-Stream Bus Traffic Meter Core are obtained by running the simulation of the test bench of the core in Vivado VSim environment. For more details about the test bench, refer to Section 6.5.

Chapter 6: Design Flow Steps

6.1 Pentek IP Catalog

This chapter describes customization and generation of the Pentek AXI4-Stream Bus Traffic Meter 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_traffic_meter_v1_0** as shown in Figure 6-1.

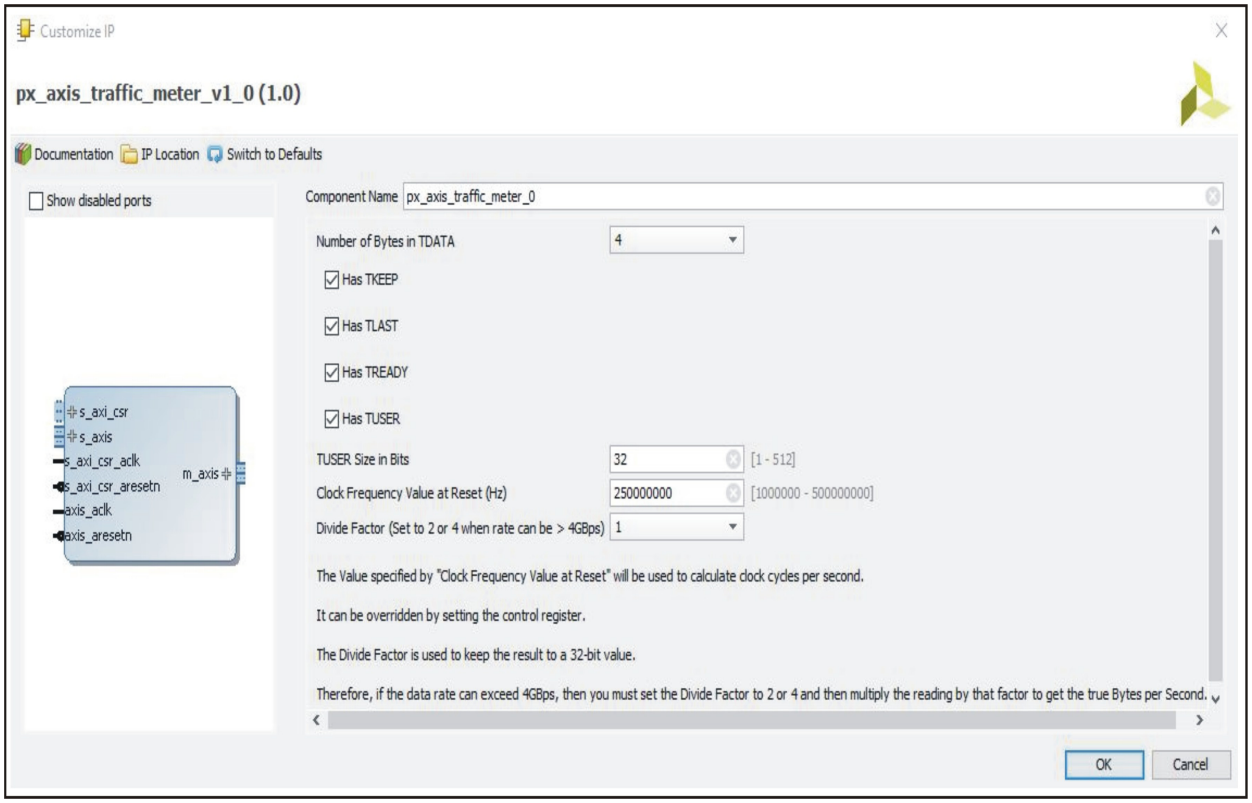
Figure 6-1: AXI4-Stream Bus Traffic Meter Core in Pentek IP Catalog



6.1 Pentek IP Catalog (continued)

When you select the **px_axis_traffic_meter_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 Bus Traffic Meter 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 Bus Traffic Meter Core in Vivado Design Suite.

Required Constraints

The XDC constraints are not provided with the AXI4-Stream Bus Traffic Meter 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 CSR clock (**s_axi_csr_aclk**), and AXI Stream clock (**axis_aclk**) both have maximum frequencies of 250 MHz for this core.

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 AXI Stream Bus Traffic Meter Core has a test bench which generates the output waveforms using the Vivado VSim environment. The test bench is designed to run at 250 MHz CSR clock frequency and 200 MHz AXI Stream clock frequency.

The test bench sets the input data width to 4 bytes, initial clock frequency to 1000 Hz, and the divide factor to a value of 1. In this test bench the AXI Stream clock period is not varied with the generic clock rate value in order to reduce the simulation time to obtain the result.

6.5 Simulation (continued)

The test bench has all AXI4-Stream signals enabled with the user data having a width of 32 bits. Fixed AXI4-Stream data is passed through the core and the data rate is obtained by reading from the Bytes per Second Status Register.

The data rate is displayed in the TCL console of the Vivado VSim environment as shown in Figure 6-4. The programming procedure is the same as described in Section 5.6. When run, the simulation produces the results shown in Figure 6-3.

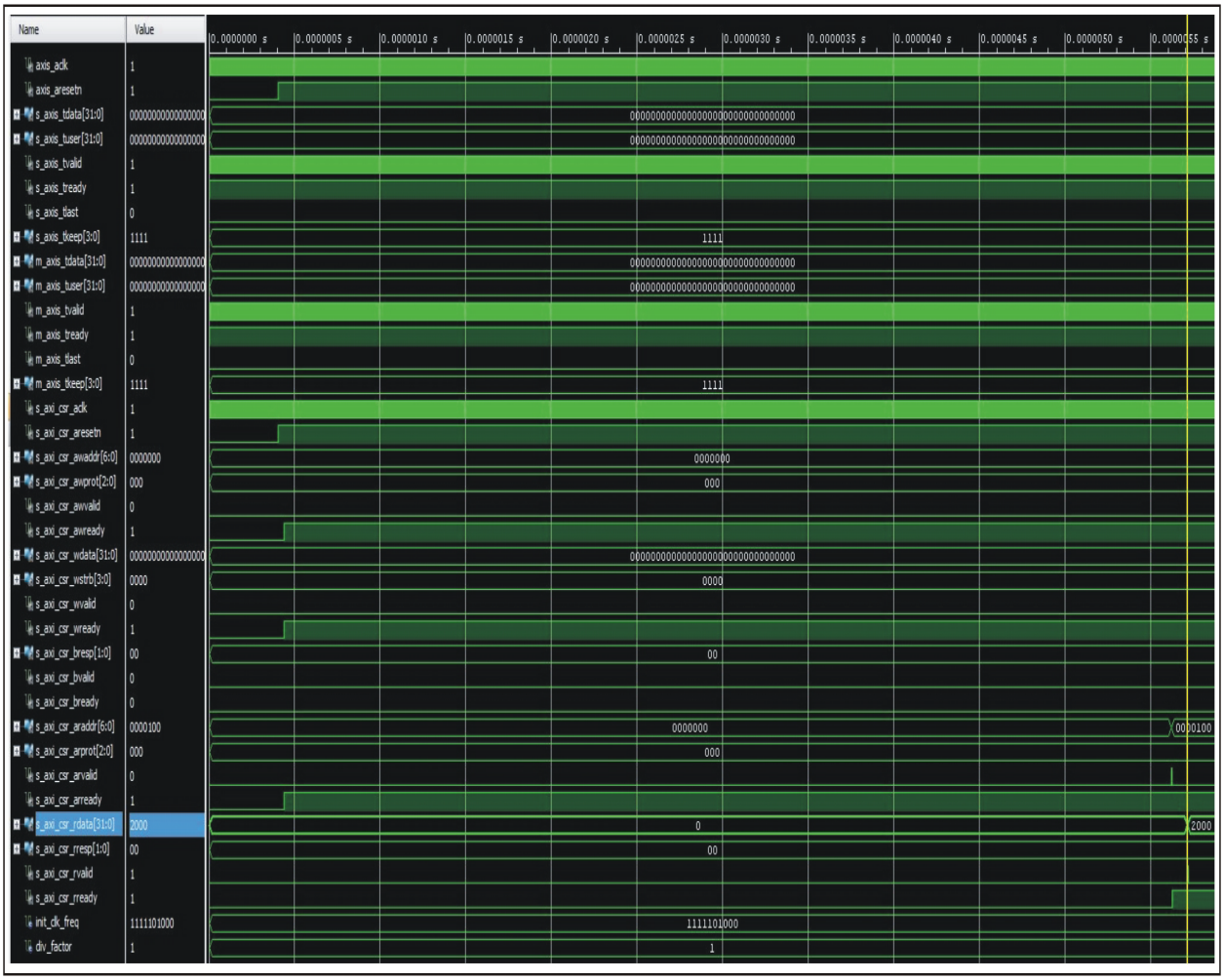


Figure 6-3: AXI4-Stream Bus Traffic Meter Core Test Bench Simulation Output -1

6.5 **Simulation** (continued)

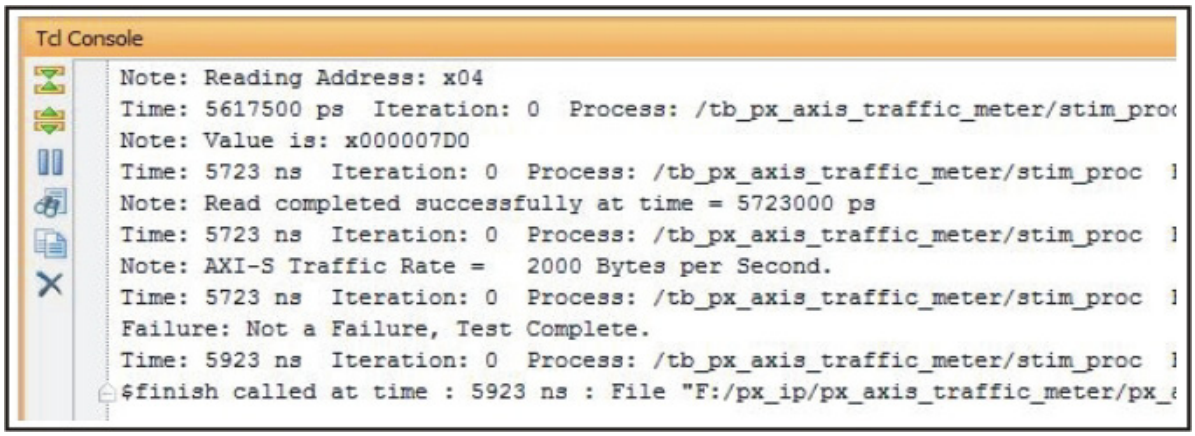


Figure 6-4: AXI4-Stream Bus Traffic Meter Core Test Bench Simulation Output -2

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