IP CORE MANUAL

Periodic Signal Counter IP

px_periodic_sig_cntr

PENTEK

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

Manual Part Number: 807.48429 Rev: 1.0 – October 31, 2018

Manual Revision History

Date	Version	Comments
10/31/18	1.0	Initial Release

Legal Notices

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

Copyright

Trademarks

Pentek, Jade, and Navigator are trademarks or registered trademarks of Pentek, Inc.

ARM and AMBA are registered trademarks of ARM Limited. PCI, PCI Express, PCIe, and PCI–SIG are trademarks or registered trademarks of PCI–SIG. Xilinx, Kintex UltraScale, Vivado, and Platform Cable USB are registered trademarks of Xilinx Inc., of San Jose, CA.

Printed in the United States of America.

Table of Contents

Page

Table of Contents

Page

Chapter 4: Register Space

Chapter 5: Designing with the Core

Table of Contents

Page

Chapter 6: Design Flow Steps

Table of Contents

Page

This page is intentionally blank

IP Facts

Description

Pentek's Navigator™ Periodic Signal Counter Core provides a count of clock cycles between rising–edge events of an input signal.

This core complies with the ARM® AMBA® *AXI4 Specification* and also provides a control/status register interface. This manual defines the hardware interface, software interface, and parameterization options for the Periodic Signal Counter Core.

Features

- Provides a count of clock cycles between rising–edge events on the input signal
- Register access through AXI4–Lite interface

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 2018.2 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

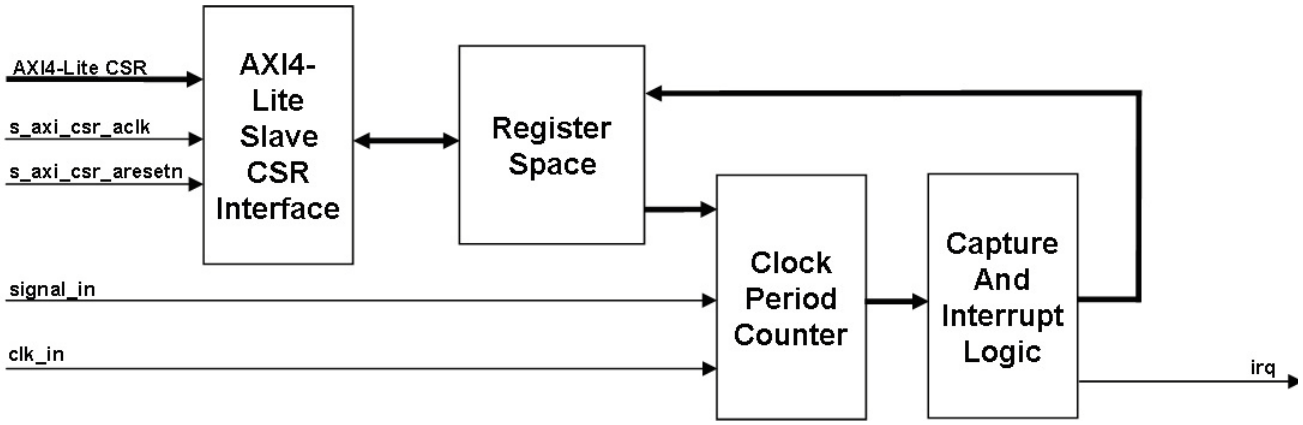
Chapter 1: Overview

1.1 Functional Description

The Periodic Signal Counter Core provides a count of clock cycles between rising−edge events on the input signal. The value of the counter is available to the user via an AXI−Lite CSR interface, as is a counter reset.

Figure 1−1 is a top−level block diagram of the Pentek Periodic Signal Counter Core. The modules within the block diagram are explained in the later sections of this manual.

Figure 1−1: Periodic Signal Counter Core Block Diagram



- **AXI4−Lite Slave CSR Interface:** This module implements a 32−bit AXI4−Lite Slave interface to access the Control and Status Register Space. For additional details about the AXI4−Lite Interface, refer to Section 3.1.

- **Register Space:** This module contains control and status registers, including Interrupt Enable, Interrupt Status and Interrupt Flag registers. Registers are accessed through the AXI4−Lite Slave CSR Interface. For register map details see Chapter 4.

- **Clock Period Counter:** This module contains the clock cycle counter and associated logic.

- **Capture and Interrupt Logic:** This module contains the logic associated with capturing the counter value as well as the interrupt logic.

1.2 Applications

The Periodic Signal Counter Core can be used for determining the periodicity of a signal 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

Chapter 2: General Product Specifications

2.1 Standards

The Periodic Signal Counter Core has bus interfaces that comply with the *ARM AMBA AXI4–Lite Protocol Specification*.

2.2 Performance

The performance of the Periodic Signal Counter Core is limited 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 Periodic Signal Counter Core has two incoming clock signals. The input clock (**`clk_in`**) has a maximum frequency of 500 MHz while the clock for the AXI4–Lite Interface (**`s_axi_csr_aclk`**) has a maximum frequency of 250 MHz on a Kintex Ultrascale –2 speed grade FPGA. Note that 250 MHz is typically the PCI Express (PCIe®) AXI bus clock frequency.

2.3 Resource Utilization

The resource utilization of the Periodic Signal Counter Core is shown in Table 2–1. Resources have been estimated for the Kintex Ultrascale XCKU060 –2e speed grade device. These values were generated using the Vivado Design Suite.

Table 2–1: Resource Usage and Availability

Resource	# Used
LUTs	94
Flip–Flops	409

NOTE: Actual utilization may vary based on the user design in which the Periodic Signal Counter Core is incorporated.

2.4 Limitations and Unsupported Features

This section is not applicable to this IP core.

2.5 Generic Parameters

This section is not applicable to this IP core.

Chapter 3: Port Descriptions

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

- AXI4–Lite Core Interfaces
- I/O Signals

3.1 AXI4–Lite Core Interfaces

The Periodic Signal Counter Core uses the Control/Status Register (CSR) interface to access the control, status and interrupt registers 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, status, and interrupt registers in the Periodic Signal Counter Core. Table 3–1 defines the ports in the CSR Interface. See Chapter 4 for the 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 value will reset all control/status registers to their initial states.
s_axi_csr_awaddr	Input	12	**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 Periodic Signal Counter Core.
s_axi_csr_awprot	Input	3	**Protection:** The Periodic Signal Counter 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 Periodic Signal Counter 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 Periodic Signal Counter 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 Periodic Signal Counter 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 Periodic Signal Counter 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.

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

Port	Direction	Width	Description
s_axi_csr_bvalid	Output	1	**Write Response Valid:** This signal is asserted by the Periodic Signal Counter 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.
s_axi_csr_araddr	Input	12	**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 Periodic Signal Counter Core.
s_axi_csr_arprot	Input	3	**Protection:** These bits are ignored by the Periodic Signal Counter 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 **s_axi_csr_araddr**. The 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 Periodic Signal Counter 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 Periodic Signal Counter 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 Periodic Signal Counter 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.
irq	Output	1	**Interrupt:** This is an active high, edge–type interrupt output.

3.2 I/O Signals

The I/O port/signal descriptions of the top–level module of the Periodic Signal Counter Core are provided in Table 3–2.

Table 3–2: I/O Signal Descriptions

Port/Signal Name	Direction	Width	Description
signal_in	Input	1	**Signal Input:** Input timing signal, synchronized to **`clk_in`**. If the counter is stopped when a rising–edge occurs on this signal, the counter will start. If the counter is already running when a rising–edge occurs on this signal, the current value of the counter is latched into the status register, the counter is reset and counting begins again.
clk_in	Input	1	**Clock In:** This is the clock which drives the counter.

Chapter 4: Register Space

This chapter provides the memory map and register descriptions for the register space of the Periodic Signal Counter Core. The memory map is provided in Table 4–1.

Table 4–1: Register Space Memory Map

Register Name	Address (Base Address +)	Access	Description
Reset	0x00	R/W	This register provides a manual reset for the core.
Period Count	0x04	RO	This register provides the last latched counter value.
Interrupt Enables	0x08	R/W	This register provides enables for the interrupts.
Interrupt Status	0x0C	RO	This register provides current status of the interrupts.
Interrupt Flag	0x10	R/CLR	This register provides status of the interrupt flags.

4.1 Reset Register

This register is used to control the counter reset. It is illustrated in Figure 4–1 and described in Table 4–2.

Figure 4–1: Reset Register

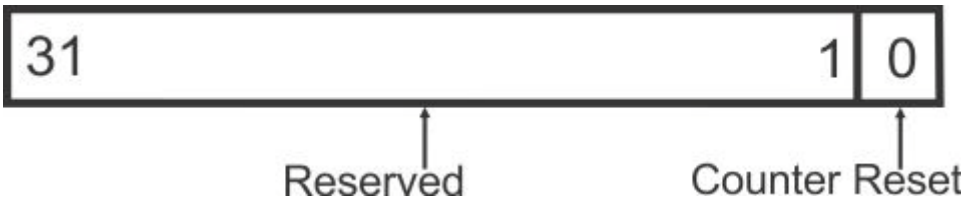


Table 4–2: Reset Register (Base Address + 0x00)

Bits	Field Name	Default Value	Access Type	Description
31:1	Reserved	N/A	N/A	**Reserved**
0	Counter Reset	0	R/W	Counter Reset: This bit is used to reset the clock–cycle counter. 0 = Run 1 = Reset

4.2 Period Count Register

This register provides the clock–cycle count of the last latched value. The clock–cycle count is latched on the rising–edge of the **signal_in** while the counter is running. It is illustrated in Figure 4–2 and described in Table 4–3.

Figure 4–2: Period Count Register

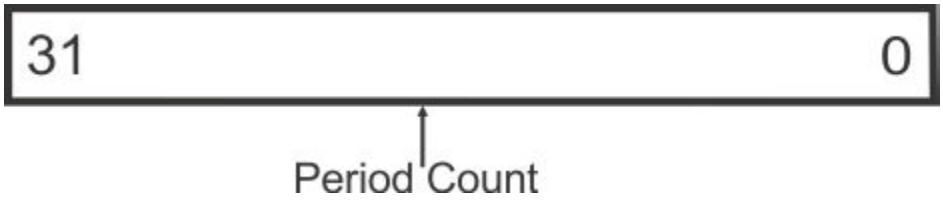


Table 4–3: Period Count Register (Base Address + 0x04)

Bits	Field Name	Default Value	Access Type	Description
31:0	Period Count	0x0000 0000	RO	**Period Count:** This is the clock–cycle count value that was latched on the last rising–edge transition of the **signal_in** input. It represents the number of clock cycles between rising–edge transitions of the **signal_in** input.

4.3 Interrupt Enable Register

The bits in the interrupt enable register are used to enable (or disable) the generation of interrupts based on the condition of certain circuit elements, known as interrupt sources. When a bit in this register associated with a given interrupt source is High, an interrupt will be generated by the rising edge of that source's Interrupt Status Register bit (See Section 4.4). It is illustrated in Figure 4–3 and described in Table 4–4.

Figure 4–3: Interrupt Enable Register

Table 4–4: Interrupt Enable Register (Base Address + 0x08)

Bits	Field Name	Default Value	Access Type	Description
31:1	Reserved	–	–	Reserved
0	Count Value Available	0	R/W	**Count Value Available:** This bit enables/disables the count value available interrupt source. The count value available interrupt source indicates that a clock–cycle count value has been captured. 0 = Disable interrupt 1 = Enable interrupt

4.4 Interrupt Status Register

The Interrupt Status Register has read–only access associated with each interrupt condition. A status bit changes to '1' when the source interrupt occurs. When a status bit in this register changes to '1' the corresponding flag bit in the Interrupt Flag Register is set to '1'. A status bit in this register clears to '0' when that interrupt condition clears, whereas the associated flag bit in the Interrupt Flag Register remains at logic '1' until it is explicitly cleared by the user.

Some of the interrupt sources are transient and so may not appear in the Interrupt Status Register at the time it is read. In such cases, use the Interrupt Flag Register to see the interrupt conditions that have occurred. It is illustrated in Figure 4–4 and described in Table 4–5.

Figure 4–4: Interrupt Status Register

Table 4–5: Interrupt Status Register (Base Address + 0x0C)

Bits	Field Name	Default Value	Access Type	Description
31:1	Reserved	–	–	Reserved
0	Count Value Available	0	RO	**Count Value Available:** This bit indicates the status of the count value available interrupt source. The count value available interrupt source indicates that a clock–cycle count value has been captured. 0 = No interrupt 1 = Interrupt condition asserted

4.5 Interrupt Flag Register

The Interrupt Flag Register has read/clear access associated with each interrupt condition. When reset, this register has all bits set to '0' (cleared). Each flag bit in this register latches an interrupt occurrence. A '1' in any flag bit in this register indicates that an interrupt has occurred.

Note that when any status bit in the Interrupt Status Register, changes to '1' the corresponding flag bit in this register will also be set to '1'. However, when a status bit in the Interrupt Status Register clears from '1' to '0, the corresponding latched flag bit in this register does not clear, but remains at '1'. To clear the flag bits, write '1's to the desired bits. The flags are not affected by the Interrupt Enable Register. It is illustrated in Figure 4–5 and described in Table 4–6.

Figure 4–5: Interrupt Flag Register

Table 4–6: Interrupt Flag Register (Base Address + 0x10)

Bits	Field Name	Default Value	Access Type	Description
31:1	Reserved	–	–	Reserved
0	Count Value Available	0	R/CLR	**Count Value Available:** This bit indicates the status of the count value available interrupt flag. The count value available interrupt source indicates that a clock–cycle count value has been captured. **Read:** 0 = No interrupt 1 = Interrupt latch **Clear:** '1' = Clear latch

Chapter 5: Designing with the Core

This chapter provides guidelines and additional information to facilitate designing with the Periodic Signal Counter Core.

5.1 General Design Guidelines

The Periodic Signal Counter Core provides the required logic to measure the time between rising−edge events on the input signal in terms of clock−cycles of an input clock. This IP core supports an AXI4−Lite user interface.

5.2 Clocking

Counter Clock: `clk_in`. This clock is used for the counter and associated logic.

CSR Clock: `s_axi_csr_aclk`. This clock is the input AXI4−Lite interface clock to the core.

5.3 Resets

Main Reset: `s_axi_csr_aresetn`. This is an active low synchronous reset associated with **`s_axi_csr_aclk`**. When asserted, all of the logic in the Periodic Signal Counter Core is reset and the registers are returned to their initial settings.

5.4 Interrupts

This core has an edge type (rising edge−triggered) interrupt output (**`irq`**), which is synchronous with **`s_axi_csr_aclk`**. On the rising edge of any interrupt signal, a one clock cycle wide pulse is output from the core on the irq output. Each interrupt event is stored in two registers, accessible on the **`s_axi_csr`** bus. The Interrupt Status Register always reflects the current state of the interrupt condition, which may have changed since the generation of the interrupt. The Interrupt Flag Register latches the occurrence of each interrupt, in a bit that retains its state until explicitly cleared. The interrupt flags can be cleared by writing '1' to the associated bit's location. All interrupt sources that are enabled (via the Interrupt Enable Register) are "OR'ed" onto the **`irq`** output.

NOTE: All interrupt sources are latched in the Interrupt Flag Register, even when an interrupt source is not enabled (via the Interrupt Enable Register).

NOTE: Because this core uses edge−triggered interrupts, an interrupt condition which remains active after servicing will not cause the generation of a new interrupt. A new interrupt will only be generated by another rising edge on an interrupt source.

5.5 Interface Operation

- **CSR Interface:** This is the Control/Status Register Interface and is associated with **`s_axis_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. For more details about this interface, refer to Section 3.1.

5.6 Programming Sequence

This section briefly describes the programming sequence for the Periodic Signal Counter Core.

1) Ensure that the Interrupt Flag Register is cleared.

2) Set the interrupt enable bit based on the user design requirement.

3) Reset the counter as needed using the Counter Reset Register.

4) When a counter value has been captured, read the value from the Period Count Register.

5) When done check the Interrupt Flag Register and clear the interrupts.

5.7 Timing Diagrams

The timing diagram for the Periodic Signal Counter Core is obtained by running the simulation of the test bench for the core in the Vivado VSim environment. For more details about the test bench, refer to Section 6.5.

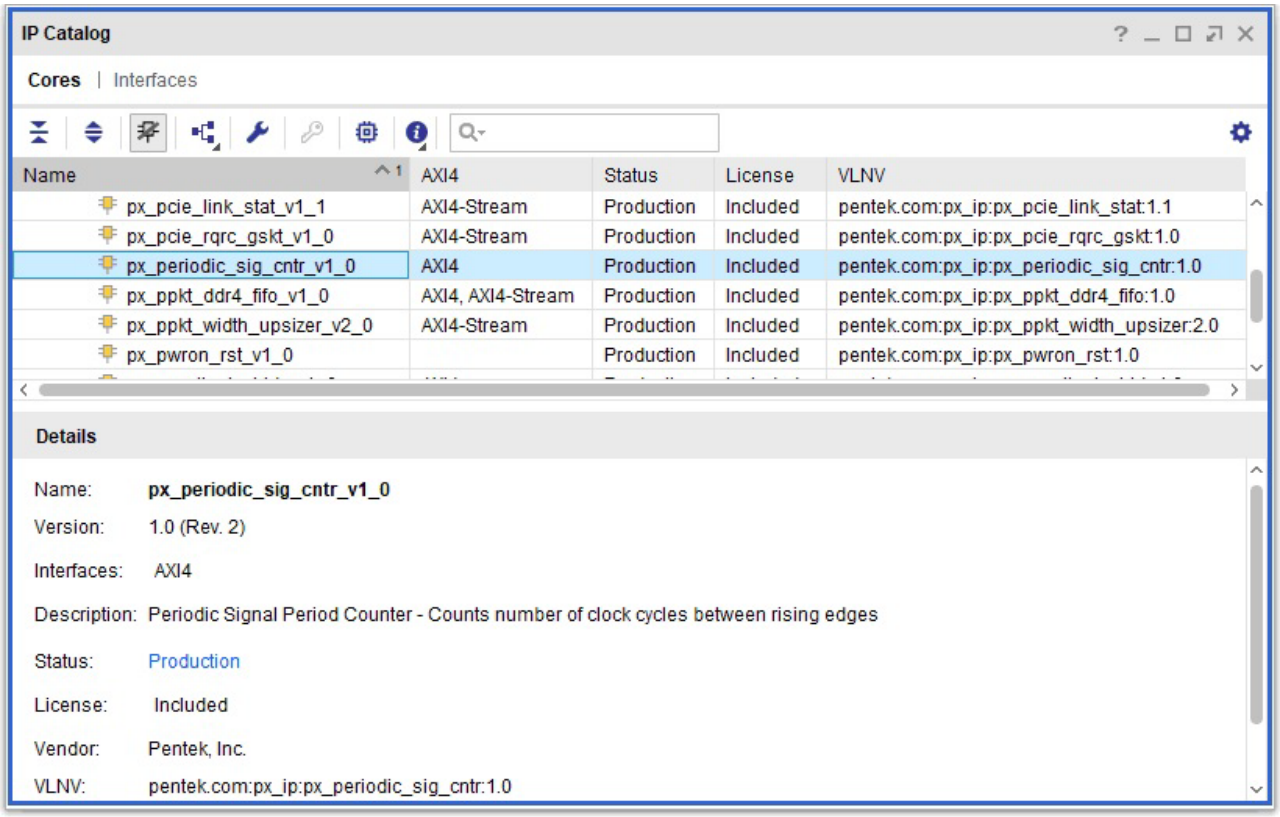
The timing diagram depicts the functionality of the core for a single timing event. The timing diagram for the simulation can be observed in Figure 6−3.

Chapter 6: Design Flow Steps

6.1 Pentek IP Catalog

This chapter describes customization and generation of the Pentek Periodic Signal Counter 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_periodic_sig_cntr_v1_0`** as shown in Figure 6–1.

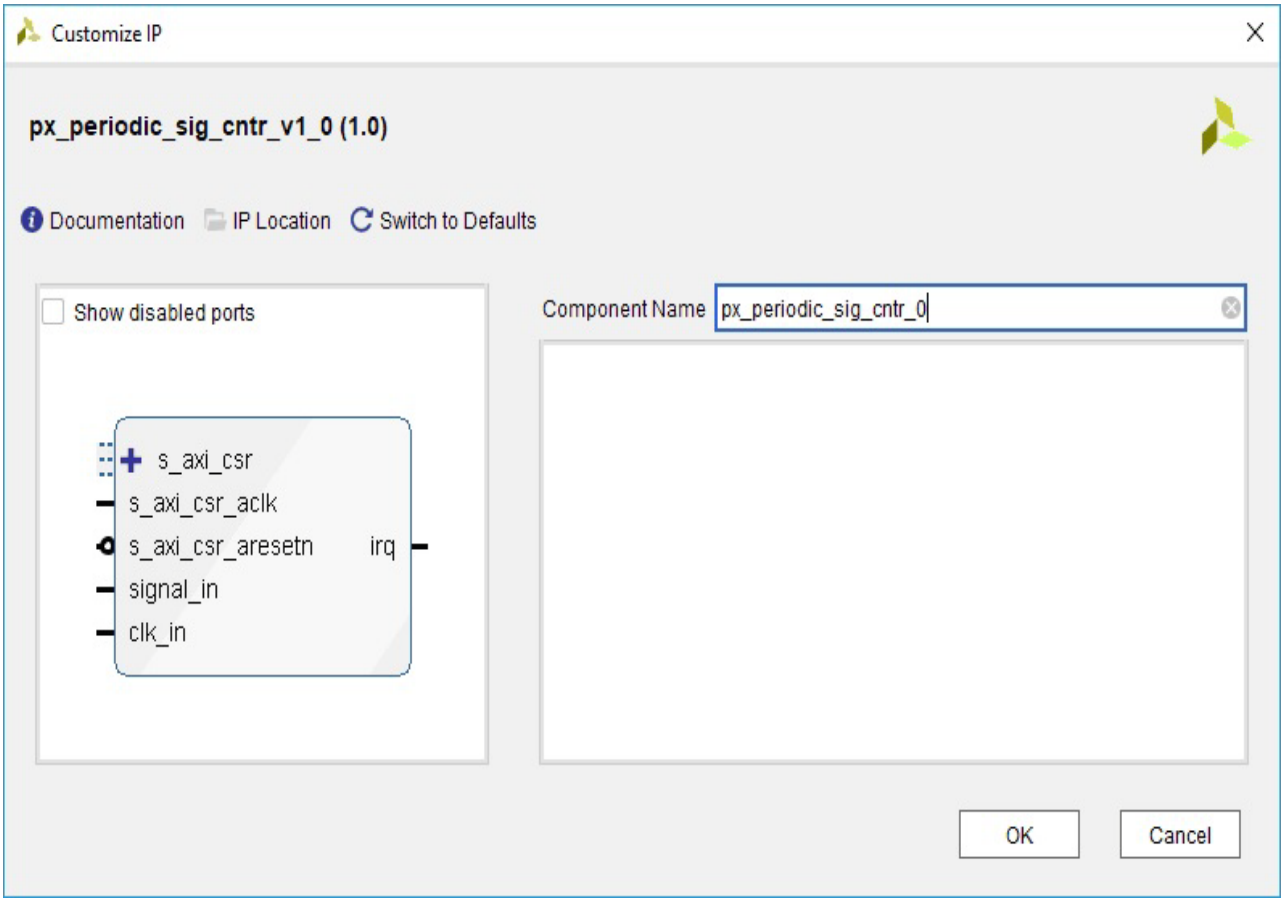
Figure 6–1: Periodic Signal Counter Core in Pentek IP Catalog



6.1 Pentek IP Catalog (continued)

When you select the **px_periodic_sig_cntr_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: Periodic Signal Counter Core IP Symbol



6.2 User Parameters

This section is not applicable to this IP core.

6.3 Output Generation

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 Periodic Signal Counter Core in the Vivado Design Suite.

Required Constraints

The XDC constraints are not provided with the Periodic Signal Counter Core. The nec−essary 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`**) has a maximum operating frequency of 250 MHz and the input clock (**`clk_in`**) has 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

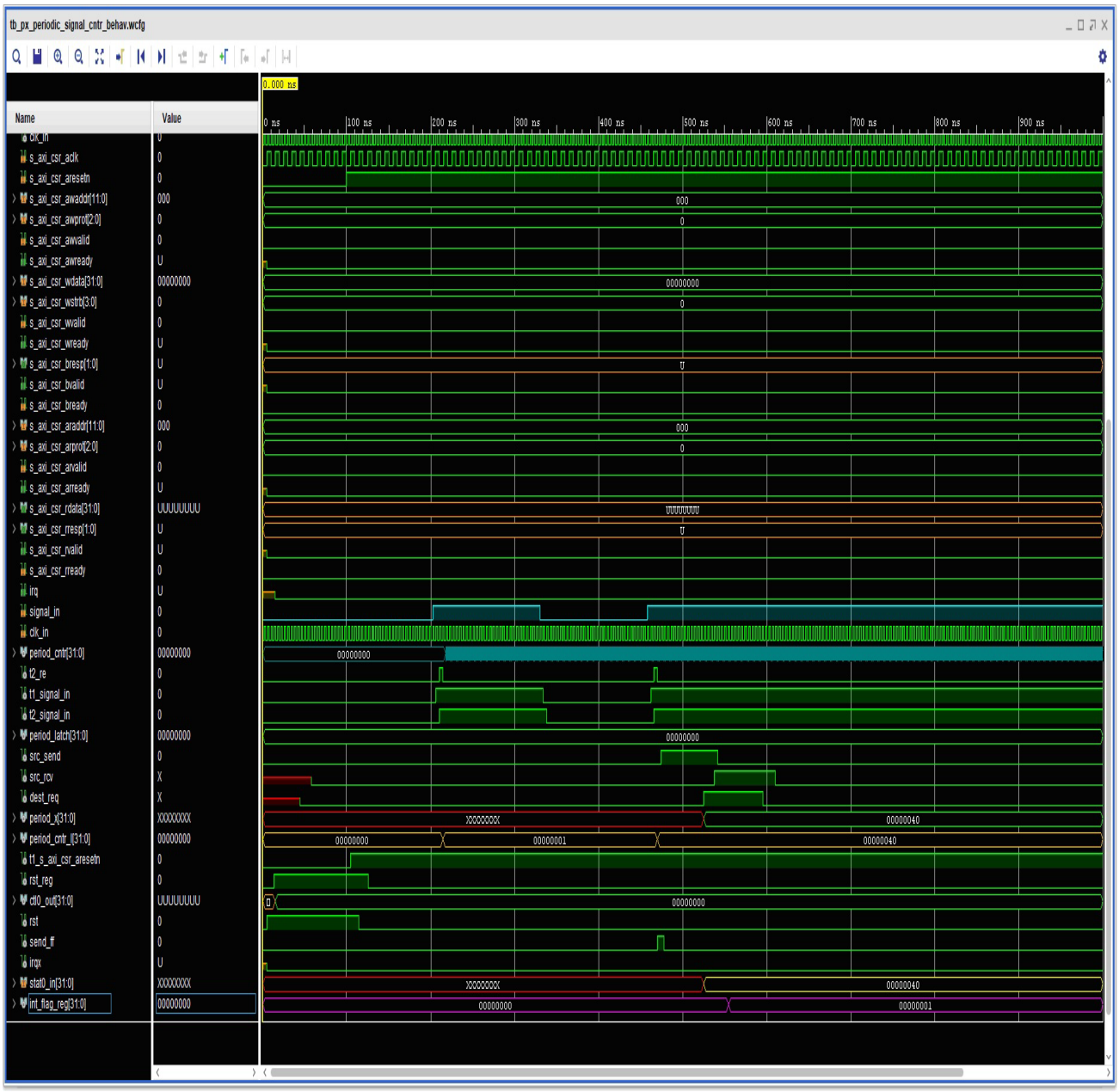
The Periodic Signal Counter Core has a test bench which measures the number of clock cycles between two **signal_input** rising−edge events. The clocks in the test bench are driven at 500MHz for the **clk_in** and 250MHz for the **s_axi_csr_aclk**.

The programming procedure is the same as described in Section 5.6.

The simulation starts with the reset (**s_axi_csr_aresetn**) being applied and released. Shortly after the reset is released, the **signal_input** is pulsed high which starts the counter (**period_cntr[31:0]**). On the next rising−edge of the signal_input the value of the counter is captured on the **period_cntr_1[31:0]** signal, and is passed to the status register (**stat0_in[31:0]**). The interrupt flag is then set.

When run, the simulation produces the results shown in Figure 6−3.

Figure 6–3: Periodic Signal Counter Core Timing Diagram



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