IP CORE MANUAL

Aggregate Multiple Interrupt Pulses IP

px_irq_pls_aggr

PENTEK

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

Manual Part Number: 807.48325 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™ Aggregate Multiple Interrupt Pulses Core generates an edge-type interrupt output by aggregating multiple input interrupt pulses. This core also provides a register interface to access interrupt registers.

This core complies with the ARM® AMBA® AXI4 Specification. This user manual defines the hardware interface, software interface, and parameterization options for the Aggregate Multiple Interrupt Pulses Core.

Features

- Configurable number of (up to 32) interrupt inputs
- Register access through AXI4-Lite Interface
- Supports Kintex Ultrascale design family

Table 1-1: IP Facts Table

Core Specifics	
Supported Design Family[a]	Kintex® Ultrascale
Supported User Interfaces	AXI4-Lite
Resources	See Table 2-1
Provided with the Core	
Design Files	VHDL
Example Design	Not Provided
Test Bench	Not Provided
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.Necessary constraints can be applied at the top-level module of the user design.

This page is intentionally blank

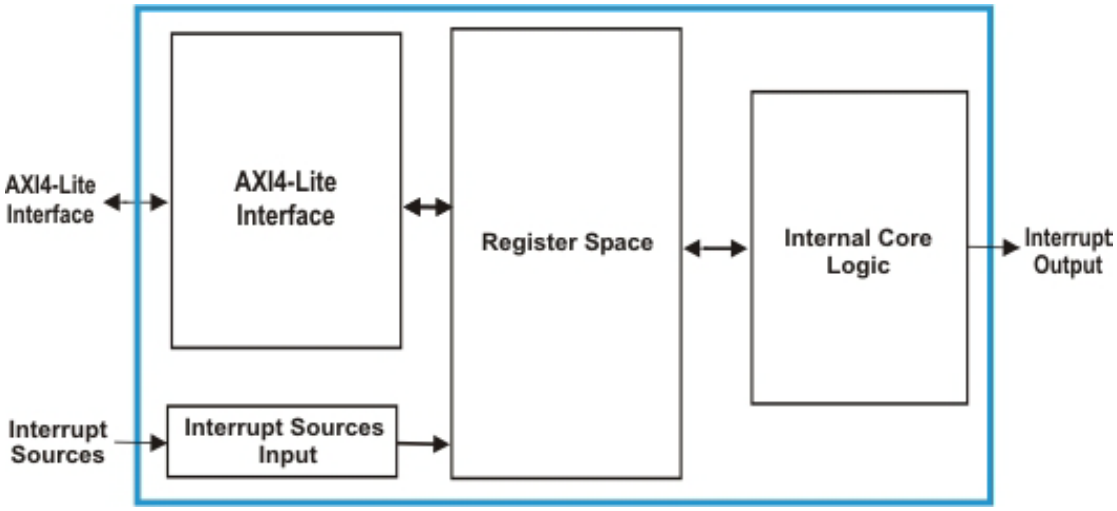
Chapter 1: Overview

1.1 Functional Description

The Aggregate Multiple Interrupt Pulses Core combines the edge-type interrupt inputs from the user design and generates a single interrupt output. The number of interrupt sources to the core can be defined by the user using the generic parameter **num_interrupt_sources** (see Table 2-2). The Aggregate Multiple Interrupt Pulses Core has an AXI4-Lite Slave Interface to access the Register Space as shown in Figure 1-1.

Figure 1-1 is a top-level block diagram of the Aggregate Multiple Interrupt Pulses Core. The modules within the block diagram are explained in other sections of this manual.

Figure 1-1: Aggregate Multiple Interrupt Pulses Core Block Diagram



- **AXI4-Lite Interface:** This module implements a 32-bit AXI4-Lite slave interface to access the register space. For more details about the AXI4-Lite Interface, refer to Section 3.1 AXI4-Lite Core Interfaces.

- **Register Space:** This module contains the interrupt registers like Interrupt Enable and Interrupt Flag registers. Registers are accessed through the AXI4-Lite Interface.

1.2 Applications

This core can be used for any application where multiple edge-type interrupt pulses are to be combined to generate a single interrupt output.

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 Aggregate Multiple Interrupt Pulses Core has a bus interface that complies with the *ARM AMBA AXI4-Lite Protocol Specification*.

2.2 Performance

The performance of the Aggregate Multiple Interrupt Pulses 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 Aggregate Multiple Interrupt Pulses Core is designed to meet a target 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 Aggregate Multiple Interrupt Pulses 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	118
Flip-Flops	271

NOTE: Actual utilization may vary based on the user design in which the Aggregate Multiple Interrupt Pulses Core is incorporated.

2.4 Limitations and Unsupported Features

This section is not applicable to this IP core.

2.5 Generic Parameter

The generic parameter of the Aggregate Multiple Interrupt Pulses Core is described in Table 2-2. This parameter can be set as required by the user application while customizing the core.

Table 2-2: Generic Parameters

Port/Signal Name	Type	Description
num_interrupt_sources	Integer	**Number of Interrupt Sources:** This parameter defines the number of interrupt source inputs to the core. It can range from 1 to 32.

Chapter 3: Port Descriptions

This chapter provides port descriptions for the following interface types:

- AXI4-Lite Core Interfaces
- I/O Signals

3.1 AXI4-Lite Core Interfaces

The Aggregate Multiple Interrupt Pulses 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 interrupt registers in the Aggregate Multiple Interrupt Pulses Core. Table 3-1 defines the ports in the CSR interface. See Chapter 4 for a Register Space 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 (250MHz)**
s_axi_csr_aresetn	Input	1	**Reset:** Active low.
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 core. Note that the Register Space registers occupy an address range of [Base Address + (0x0 to 0x4)].
s_axi_csr_awprot	Input	3	**Protection:** The Aggregate Multiple Interrupt Pulses 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 Aggregate Multiple Interrupt Pulses 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 Aggregate Multiple Interrupt Pulses 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 Aggregate Multiple Interrupt Pulses 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 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_bvalid	Output	1	**Write Response Valid:** This signal is asserted by the 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_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_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 Aggregate Multiple Interrupt Pulses Core.
s_axi_csr_arprot	Input	3	**Protection:** These bits are ignored by the Aggregate Multiple Interrupt Pulses 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 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 Aggregate Multiple Interrupt Pulses 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 Aggregate Multiple Interrupt Pulses 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 core when the read operation 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 I/O Signals

The I/O port/signal descriptions of the top-level module of the Aggregate Multiple Interrupt Pulses Core are described in Table 3-2.

Table 3-2: I/O Signals

Port/Signal Name	Type	Direction	Description
int_in [num_interrupt_ sources-1: 0]	std_logic_vector	Input	**Interrupt Sources:** These are the interrupt source inputs to the core. The width of this signal depends on the generic parameter **num_interrupt_sources**.
irq_out	std_logic	Output	**Interrupt Output:** This is the interrupt output signal of the core obtained from aggregating the interrupt sources.

Chapter 4: Register Space

This chapter provides the memory map and register descriptions for the register space of the Aggregate Multiple Interrupt Pulses Core. The memory map is provided in Table 4-1.

Table 4-1: Register Space Memory Map

Register Name	Address (Base Address +)	Access	Description
Interrupt Enable Register	0x00	R/W	Interrupt enable bits
Interrupt Flag Register	0x04	R/Clr	Interrupt flag bits

4.1 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 output will be generated by the rising edge of that interrupt source.

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

Bits	Field Name	Default Value	Access Type	Description
31:0	int_src	0x00000000	R/W	**Interrupt Sources:** Each of the bits in this register corresponds to an incoming interrupt source. Bit 0 corresponds to interrupt source input[0] and bit 31 to interrupt input [31]. When the number of interrupt sources is less than 32, the remaining bits of this register are assigned with the default value. These bits are used to enable/ disable the corresponding interrupt source. 0 = Disable interrupt 1 = Enable interrupt

4.2 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 when the corresponding interrupt bit is enabled by the Interrupt Enable Register. To clear the flag bits, write '1's to the desired bits.

Table 4-3: Interrupt Flag Register (Base Address + 0x04)

Bits	Field Name	Default Value	Access Type	Description
31:0	int_src	0x00000000	R/Clr	**Interrupt Sources:** Each of the bits in this register corresponds to an incoming interrupt source. Bit 0 corresponds to interrupt source input[0] and bit 31 to interrupt input [31]. When the number of interrupt sources is less than 32, the remaining bits of this register are assigned with the default value. These bits indicate the corresponding interrupt flag. **Read:** 0 = No interrupt 1 = Interrupt latched **Clear:** 1 = Clear latch

Chapter 5: Designing with the Core

This chapter includes guidelines and additional information to facilitate designing with the Aggregate Multiple Interrupt Pulses Core.

5.1 General Design Guidelines

The Aggregate Multiple Interrupt Pulses Core provides the required logic to generate an edge-type interrupt output from the input interrupt sources.

5.2 Clocking

Main Clock: **s_axi_csr_aclk**

This clock is used to clock all the ports on the Aggregate Multiple Interrupt Pulses Core.

5.3 Resets

Main reset: **s_axi_csr_aresetn**

This is an active low synchronous reset associated with the **s_axi_csr_aclk**.

5.4 Interrupts

This core has edge-type (rising-edge-triggered) interrupt inputs. They are synchronous with the **s_axi_csr_aclk**. On the rising edge of any interrupt signal, a one-clock-cycle-wide pulse is generated on its **irq** output when the corresponding interrupt bit is enabled by the Interrupt Enable Register. The Interrupt Flag Register latches the occurrence of each enabled 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 output interrupt signal.

NOTE: Because this core uses edge-triggered interrupts, the fact that an interrupt condition may remain 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 the **s_axi_csr_aclk**. It is a standard AXI4-Lite Interface. See Chapter 4 for the register memory map and more details about the registers that can be accessed through this interface.

5.6 Programming Sequence

This section briefly describes the programming sequence of registers to initiate and complete a transaction on the Aggregate Multiple Interrupt Pulses Core.

1) Ensure that the Interrupt Flag Register is cleared.

2) Enable the required interrupt sources to generate an interrupt output.

3) Observe the output across the **irq_out** port of the core.

4) Clear the Interrupt Flag Register.

5.7 Timing Diagrams

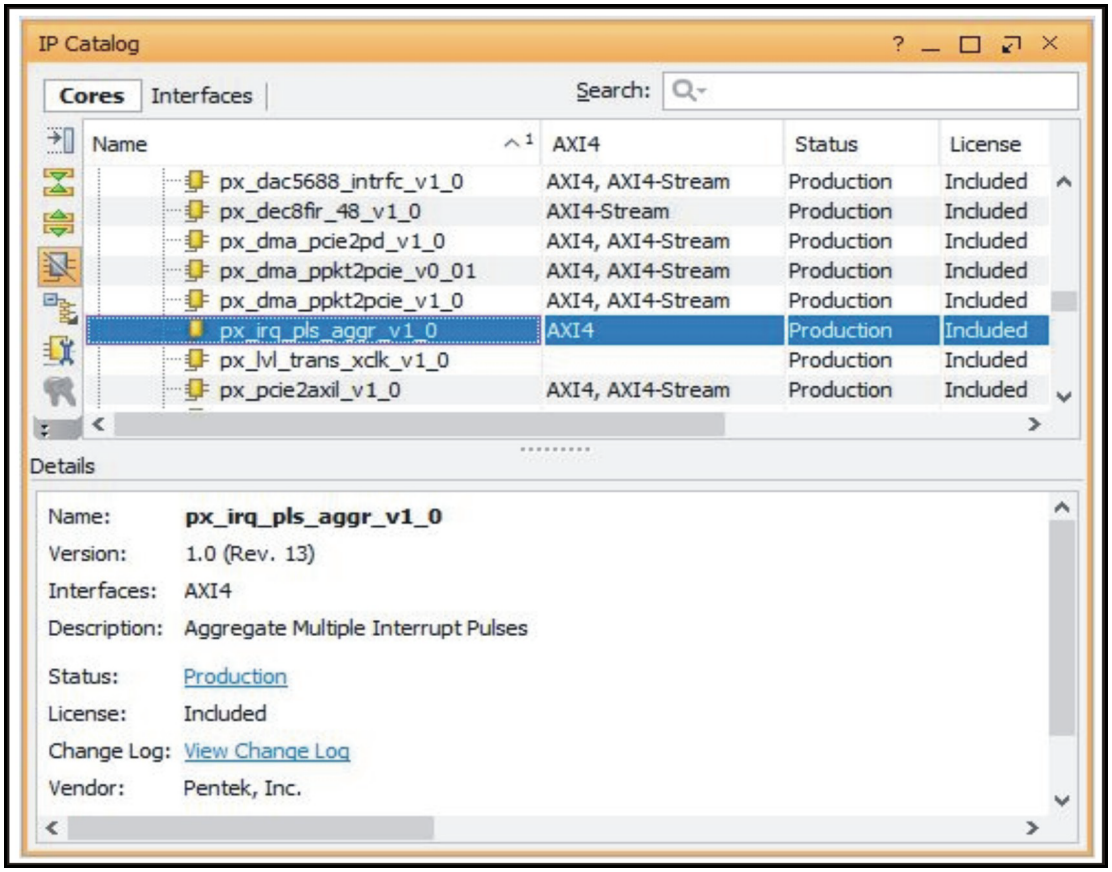
This section is not applicable to this IP core.

Chapter 6: Design Flow Steps

6.1 Pentek IP Catalog

This chapter describes customization and generation of the Pentek Aggregate Multiple Interrupt Pulses 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_irq_pls_aggr_v1_0** as shown in Figure 6-1.

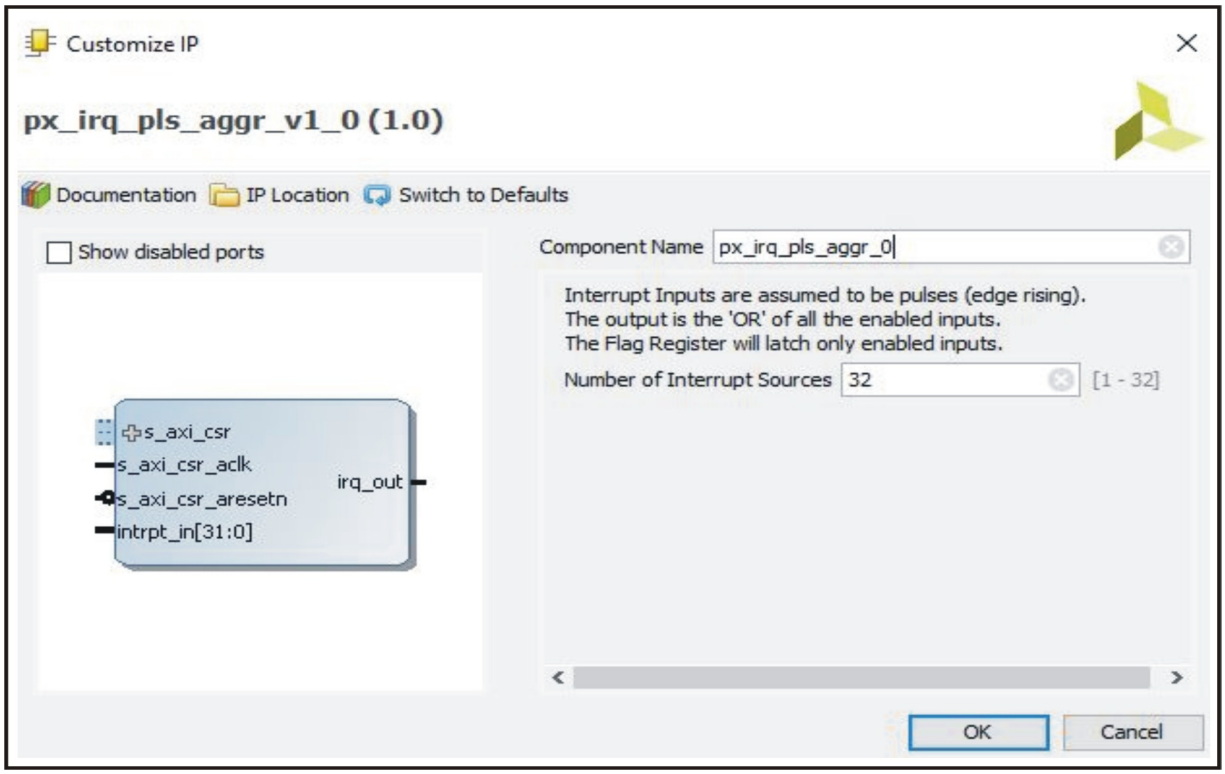
Figure 6-1: Aggregate Multiple Interrupt Pulses Core in Pentek IP



6.1 Pentek IP Catalog (continued)

When you select the **px_irq_pls_aggr_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: Aggregate Multiple Interrupt Pulses Core IP Symbol



6.2 User Parameters

Number of Interrupt Sources: This value indicates the number of interrupt sources to the Aggregate Multiple Interrupt Pulses Core and it can be defined by the user based on the application requirement. It can range from 1 to 32.

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 core in Vivado Design Suite environment.

Required Constraints

The XDC constraints are not provided with this core. Necessary 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 maximum clock frequency (**s_axi_csr_clk**) for this IP core is 250 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

This section is not applicable to this IP core.

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