IP CORE MANUAL

PCI Express Interrupt Controller IP

px_pcie_irq_ctlr

PENTEK

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

Manual Part Number: 807.48326 Rev: 1.0 - December 09, 2016

Manual Revision History

Date	Version	Comments
12/09/16	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

IP Facts

Description....5
Features....5
Table 1-1: IP Facts Table....5

Chapter 1: Overview

1.1 Functional Description....7
Figure 1-1: PCI Express Interrupt Controller Core Block Diagram....7
1.2 Applications....8
1.3 System Requirements....8
1.4 Licensing and Ordering Information....8
1.5 Contacting Technical Support....8
1.6 Documentation....8

Chapter 2: General Product Specifications

2.1 Standards....9
2.2 Performance....9
2.2.1 Maximum Frequencies....9
2.3 Resource Utilization....9
Table 2-1: Resource Usage and Availability....9
2.4 Limitations and Unsupported Features....9
2.5 Generic Parameters....10
Table 2-2: Generic Parameters....10

Chapter 3: Port Descriptions

3.1 AXI4-Lite Core Interfaces....11
3.1.1 Control/Status Register (CSR) Interface....11
Table 3-1: Control/Status Register (CSR) Interface Port Descriptions....11
3.2 I/O Signals....14
Table 3-2: I/O Signals....14

Table of Contents

Page

Chapter 4: Register Space

Table 4-1: Register Space Memory Map 17
4.1 Interrupt Enable Register 17
Table 4-2: Interrupt Enable Register (Base Address + 0x0) 17
4.2 Interrupt Flag Register 18
Table 4-3: Interrupt Flag Register (Base Address + 0x4) 18

Chapter 5: Designing with the Core

5.1 General Design Guidelines 19
5.2 Clocking 19
5.3 Resets 19
5.4 Interrupts 19
5.5 Interface Operation 20
5.6 Programming Sequence 20
5.7 Timing Diagrams 20
Figure 5-1: PCI Express Interrupt Controller Core Simulation Output - Legacy Interrupt Mode ... 21
Figure 5-2: PCI Express Interrupt Controller Core Simulation Output - MSI Mode 21

Chapter 6: Design Flow Steps

Figure 6-1: PCI Express Interrupt Controller Core in Pentek IP Catalog 23
Figure 6-2: PCI Express Interrupt Controller Core IP Symbol 24
6.2 User Parameters 24
6.3 Generating Output 24
6.4 Constraining the Core 25
6.5 Simulation 26
Figure 6-3: PCI Express Interrupt Controller Core Test Bench Simulation Output 26
6.6 Synthesis and Implementation 26

IP Facts

Description

Pentek's Navigator™ PCI Express (PCIe®) PCIe Interrupt Controller Core converts edge-type interrupt inputs from the user design into Legacy Interrupts, and Message Signaled Interrupt (MSI) signals required by the Xilinx® Gen3 Integrated Block for PCI Express IP 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 PCIe Interrupt Controller Core.

Features

- Configurable number of (up to 32) interrupt inputs
- Supports single vector MSI
- Register access through AXI4-Lite Interface
- Supports Xilinx Kintex Ultrascale and Xilinx Virtex-7 design families
- Supports dword addressing mode

Table 1-1: IP Facts Table

Core Specifics	
Supported Design Family[a]	Kintex® Ultrascale and Virtex-7
Supported User Interfaces	AXI4-Lite
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

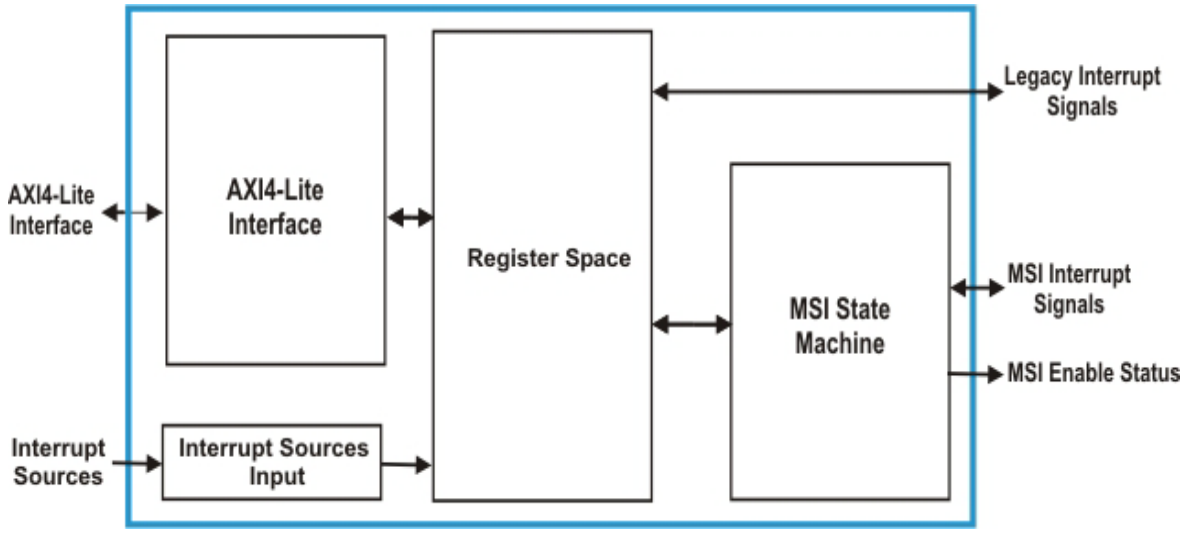
Chapter 1: Overview

1.1 Functional Description

The PCIe Interrupt Controller Core serves as an interrupt controller which combines the interrupts from the other modules of the user design and generates the necessary signals for the Xilinx PCIe core. The core generates MSI or Legacy configuration interface interrupt signals based on the interrupt mode selected. The core also has an output status signal to indicate the interrupt operation mode (Legacy or MSI) of the core. The PCIe Interrupt Controller Core has an AXI4-Lite Slave Interface which is connected to the Register Space as shown in Figure 1-1.

Figure 1-1 is a top-level block diagram of the Pentek PCIe Interrupt Controller Core. The modules within the block diagram are explained in the later sections of this manual.

Figure 1-1: PCI Express Interrupt Controller Core Block Diagram



1.1 Functional Description (continued)

- **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 Interrupt registers like Interrupt Enable and Interrupt Flag registers. Registers are accessed through the AXI4-Lite Interface.
- **MSI Interrupt State Machine:** This state machine is used to generate the configuration interface MSI signals when the MSI Interrupt mode is enabled.

1.2 Applications

This PCIe Interrupt Controller Core can be used for any application that requires combining edge-type interrupts from peripheral devices into one interrupt signal and converting it to legacy and MSI signals for the Xilinx Gen3 Integrated Block for PCIe IP core.

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
4) *Xilinx Gen3 Integrated Block for PCI Express Product Guide*

Chapter 2: General Product Specifications

2.1 Standards

The PCIe Interrupt Controller Core has bus a interface that complies with the *ARM AMBA AXI4-Lite Protocol Specification*.

2.2 Performance

The performance of the PCIe Interrupt Controller 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 PCIe Interrupt Controller 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 PCIe Interrupt Controller 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	113
Flip-Flops	279

NOTE: Actual utilization may vary based on the user design in which the PCIe Interrupt Controller Core is incorporated.

2.4 Limitations and Unsupported Features

- ❑ This IP core does not support the multi-vector MSI operation.
- ❑ This IP core does not support MSI-X interrupt mode.

2.5 Generic Parameters

The generic parameters of the PCIe Interrupt Controller 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
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.
ku		**FPGA Type:** This parameter defines the type of FPGA this core is being used with. 0 - Virtex-7 1 - Kintex Ultrascale

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 PCIe Interrupt Controller 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 PCIe Interrupt Controller 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	4	**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 PCIe Interrupt Controller Core.
s_axi_csr_awprot	Input	3	**Protection:** The PCIe Interrupt Controller 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 PCIe Interrupt Controller 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 PCIe Interrupt Controller 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 PCIe Interrupt Controller 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 PCIe Interrupt Controller 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 PCIe Interrupt Controller 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	4	**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 PCIe Interrupt Controller Core.
s_axi_csr_arprot	Input	3	**Protection:** These bits are ignored by the PCIe Interrupt Controller 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 PCIe Interrupt Controller 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 PCIe Interrupt Controller 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 PCIe Interrupt Controller 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 PCIe Interrupt Controller 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 I/O Signals

The I/O port/signal descriptions of the top-level module of the PCIe Interrupt Controller Core are discussed in the Table 3-2. For more details about Configuration Interrupt Controller Interface of the Xilinx Gen3 Integrated Block for PCI Express IP Core, refer to the Product Guide(*Xilinx Gen3 Integrated Block for PCI Express IP Core Product Guide*).

Table 3-2: I/O Signals

Port/Signal Name	Type	Direction	Description
Interrupt Sources			
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**.
Interrupt Mode Status Signal			
msi_enable_out[3:0]	std_logic _vector	Output	**MSI Mode Status Signal:** This signal indicates if the MSI mode is enabled. **msi_enable_out[0]** = 0 => MSI mode disabled. Legacy mode enabled. **msi_enable_out[0]** = 1 => MSI mode enabled.
Configuration Interface Interrupt Signals			
Legacy Interrupt Signals			
cfg_interrupt_int[3:0]	std_logic _vector	Output	**Configuration INTx vector:** The core uses this signal to indicate an interrupt in Legacy Interrupt mode. These four bits correspond to INTA, INTB, INTC, and INTD. Asserting one of these signals causes the Xilinx PCIe core to send out an **Assert_INTx** message, deassserting this signal causes the Xilinx PCIe core to transmit a **Deassert_INTx** message over the PCIe link.
cfg_interrupt_pending			**Configuration INTx Interrupt Pending:** Indicates a pending interrupt.
cfg_interrupt_sent	std_logic	Input	**Configuration INTx Sent:** A pulse on this input from the Xilinx PCIe core indicates that an **Assert_INTx** or a **Deassert_INTx** message has been sent in response to a change in the **cfg_interrupt_int** output from the core.

Table 3-2: I/O Signals (Continued)

Port/Signal Name	Type	Direction	Description
Configuration Interface Interrupt Signals (continued)			
MSI Signals			
cfg_interrupt_msi_ attr[2:0]	std_logic _vector	Output	**Configuration MSI Interrupt TLP Attribute:** These bits provide the attribute bits setting for the MSI request. Bit 0 - No Snoop Bit 1 - Relaxed ordering Bit 2 - ID-Based ordering
cfg_interrupt_msi _data[31:0]	std_logic _vector	Input	**Configuration Interrupt MSI Data**
cfg_interrupt_msi_ enable			**Configuration Interrupt MSI Fuction Enabled:** This signal indicates that the MSI messaging mode is enabled.
cfg_interrupt_msi_fail	std_logic		**Configuration Interrupt MSI Operation Failed:** This signal from the Xilinx PCIe core indicates that the MSI interrupt messge had been aborted before transmission over the PCIe link. The core must retransmit the MSI interrupt when this signal is High.
cfg_interrupt_msi _function_number	std_logic _vector	Output	**Configuration MSI Function Number:** This indicates the endpoint function number initiating the MSI transaction.
cfg_interrupt_msi_int [63-(ku*32) downto 0]	std_logic _vector		**Configuration MSI Vector:** These bits are used to indicate the interrupt condition associated with the PCI function for an interrupt request. The core must activate only one of these signals for one cycle to transmit an interrupt. After asserting the interrupt the core must wait until an **cfg_interrupt_msi_sent** or **cfg_interrupt_msi_fail** indication from the PCIe core is received before asserting another interrupt.
cfg_interrupt_msi _mask_update	std_logic	Input	**Configuration Interrupt MSI Mask Update:** Asserted for one cycle when any enabled funtions in the MSI Mask Register change value. MSI mask register contains the mask bits for MSI Interrupts.
cfg_interrupt_msi _mmenable	std_logic _vector		**Configuration Interrupt MSI Multiple Message Enable:** This is the value of the Multiple Message Enable field and indicates the number of allocated MSI Interrupt vectors.

Table 3-2: I/O Signals (Continued)

Port/Signal Name	Type	Direction	Description
Configuration Interface Interrupt Signals (continued)			
MSI Signals (continued)			
cfg_interrupt_msi _pending_status	std_logic _vector	Output	**Configuration Interrupt MSI Pending Status:** The core provides the status of pending MSI interrupts through this signal.
cfg_interrupt_msi_pending _status_data_enable	std_logic		**Configuration Interrupt MSI Pending Status Data Enable:** This signal enables the MSI pending status signal.
cfg_interrupt_msi_pending _status_func_num[3:0]	std_logic _vector		**Configuration Interrupt MSI Pending Status Function number:** This signal indicates the function number of the pending MSI interrupt request.
cfg_interrupt_msi _select[3:0]	std_logic _vector	Output	**Configuration Interrupt MSI Select:** These bits indicate the selected function of the MSI interrupt request 0000 - 0001 = Physical Function 0 - 1 0010 - 1111 = Virtual Function 0 - 5
cfg_interrupt_msi_sent	std_logic	Input	**Configuration Interrupt MSI Sent:** A one clock cycle pulse on this input indicates that an MSI interrupt message has been transmitted over the PCIe link by the Xilinx PCIe core.
cfg_interrupt_msi _tph_present		Output	**Configuration Interrupt MSI Transaction Processing Hint (TPH) Present:** Indicates the presence of a TPH in the MSI request. The core asserts this bit while asserting **cfg_interrupt_msi _int**, if it includes a TPH.
cfg_interrupt_msi _tph_st_tag[8:0]	std_logic _vector		**Configuration Interrupt MSI TPH Steering Tag:** When **cfg_interrupt_msi_tph_present** is High, the steering tag associated with the hint must be placed on **cfg_interrupt_ msi_tph_st_tag[7:0]**.
cfg_interrupt_msi _tph_type[1:0]			**Configuration Interrupt MSI TPH Type:** When **cfg_interrupt _msi_tph_present** is High, these bits indicate the two-bit type associated with the hint.
cfg_interrupt_msi _vf_enable		Input	**Configuration Interrupt MSI Virtual Function Enable:** Indicates MSI messaging is enabled, per virtual function.

Chapter 4: Register Space

This chapter provides the memory map and register descriptions for the register space of the PCIe Interrupt Controller 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	0x0	R/W	Interrupt enable bits
Interrupt Flag Register	0x4	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 will be generated by the rising edge of that interrupt source. This register is described in Table 4-2.

Table 4-2: Interrupt Enable Register (Base Address + 0x0)

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 their default values. 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 (the same bit association as the Interrupt Enable Register). 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. To clear the flag bits, write '1's to the desired bits. This register is described on Table 4-3.

Table 4-3: Interrupt Flag Register (Base Address + 0x4)

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 their default values. 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 PCIe Interrupt Controller Core.

5.1 General Design Guidelines

The PCIe Interrupt Controller core provides the required logic to generate the Legacy and MSI interrupt request signals required by the Xilinx Gen3 PCIe core. The interrupt sources from the user design are combined to generate an interrupt control for the PCIe core. For details on the configuration interrupt controller interface of the Xilinx PCIe core, refer to the Product Guide (*Xilinx Gen4 Integrated Block for PCI Express IP Core Product Guide*).

5.2 Clocking

Main Clock: **aclk**

This clock will be the User Interface clock (**user_clk**) output of the Xilinx PCIe core. The Xilinx PCIe core's user clock output is 250MHz. This clock is used to clock all the ports on the PCIe Interrupt Controller core.

5.3 Resets

Main reset: **aresetn**

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

5.4 Interrupts

This core has edge-type (rising edge-triggered) interrupt inputs. They are synchronous with the **aclk**. On the rising edge of any interrupt signal, a one-clock-cycle wide pulse is generated on it's internal interrupt signal. Each interrupt event is stored in the Interrupt Flag register, accessible on the **s_axi_csr** bus.

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 internal interrupt signal. The Legacy or MSI interrupt signals are generated and transmitted to the Xilinx PCIe core based on the internal interrupt signal.

5.4 Interrupts (continued)

NOTE: All interrupt sources are latched in the interrupt flag register, even when an interrupt source is not enabled to create an interrupt.

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 **aclk**. It is a standard AXI4-Lite Slave interface. See Chapter 4 for the Register Space memory map, which provides more details on the registers that can be accessed through this interface.

5.6 Programming Sequence

This section briefly describes the programming sequence of registers in the PCIe Interrupt Controller Core.

1) Ensure that the Interrupt Flag Register is cleared.
2) Enable the required interrupt sources to generate an interrupt.
3) Based on the Interrupt mode, Legacy or MSI signals are transmitted.
4) Clear the Interrupt Flag Register.

5.7 Timing Diagrams

The timing diagrams for the PCIe Interrupt Controller core are shown in Figures 5-1 and 5-2. These timing diagrams were obtained by running the simulation of the test bench of the core in Vivado VSim environment. These timing diagrams depict the functionality of the core for Legacy and MSI Interrupt modes.

5.7 Timing Diagrams (continued)

Figure 5-1: PCI Express Interrupt Controller Core Simulation Output - Legacy Interrupt Mode

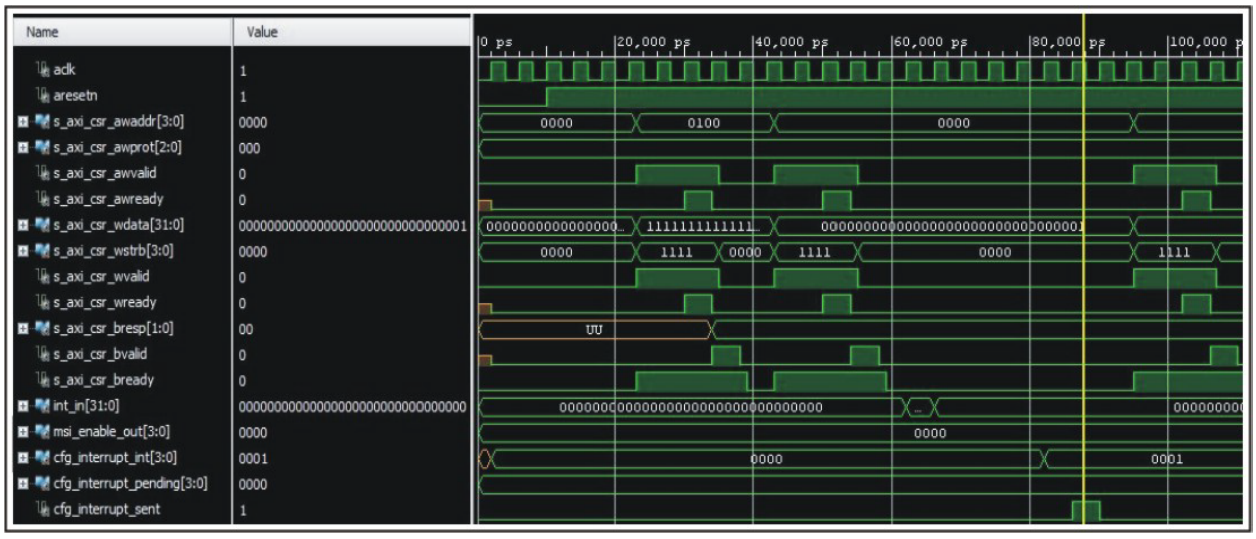
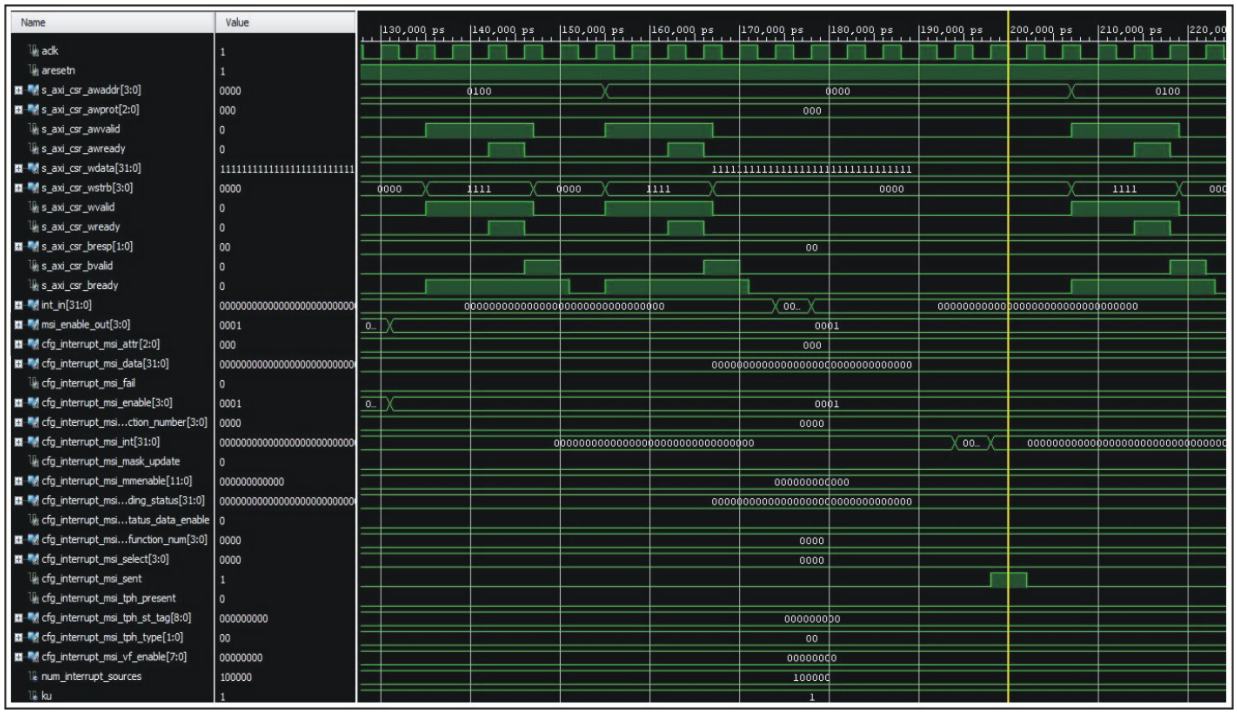


Figure 5-2: PCI Express Interrupt Controller Core Simulation Output - MSI Mode



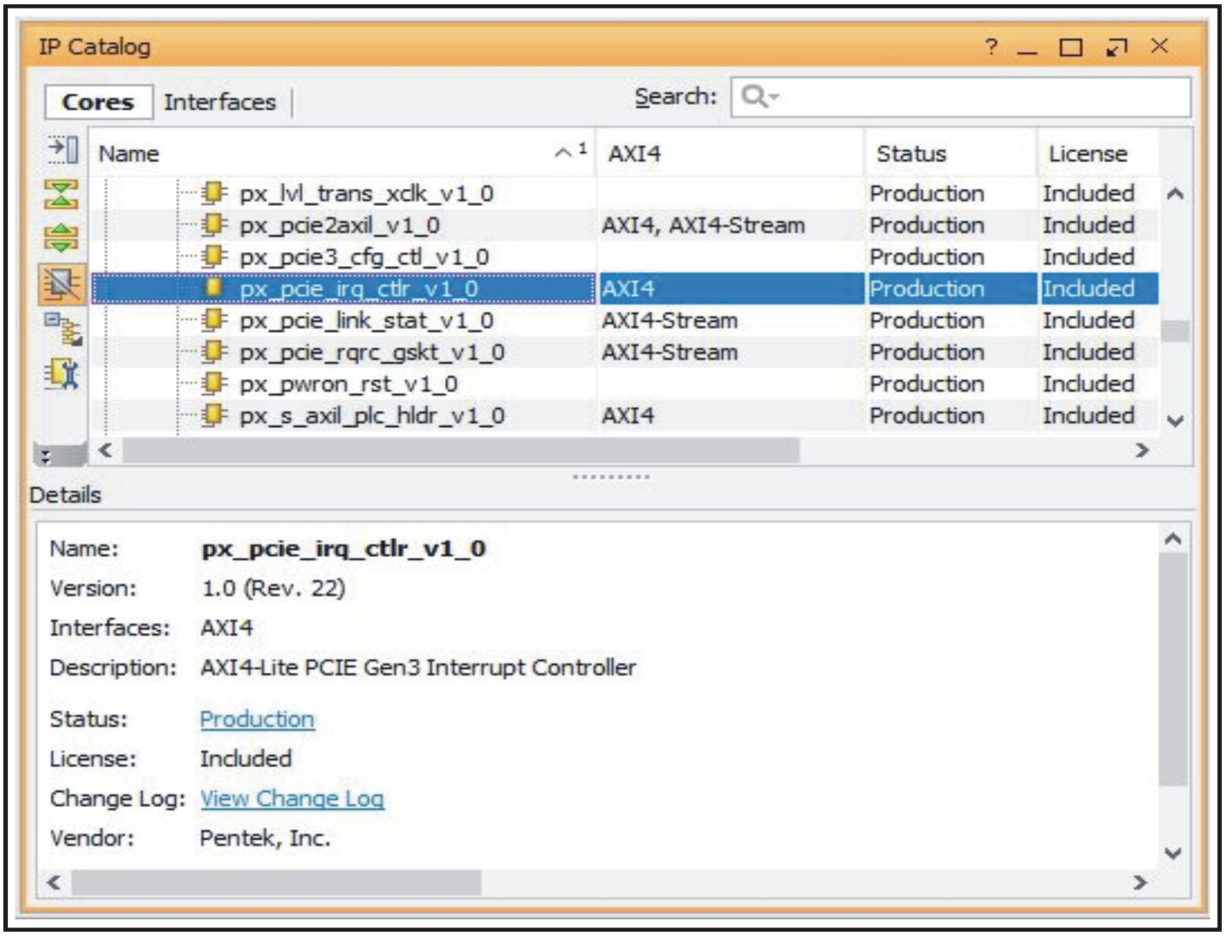
This page is intentionally blank

Chapter 6: Design Flow Steps

6.1 Pentek IP Catalog

This chapter describes customization and generation of the Pentek PCIe Interrupt Controller 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_pcie_irq_ctlr_v1_0** as shown in Figure 6-1.

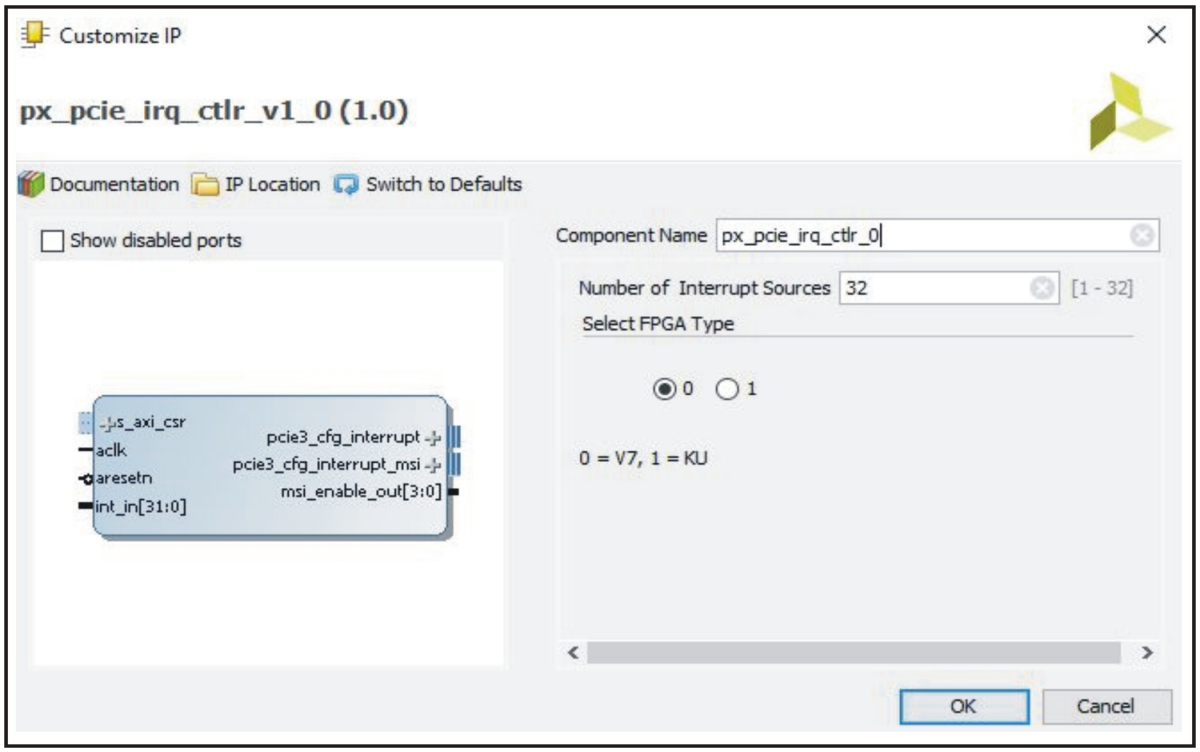
Figure 6-1: PCI Express Interrupt Controller Core in Pentek IP Catalog



6.1 Pentek IP Catalog (continued)

When you select the **px_pcie_irq_ctlr_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: PCI Express Interrupt Controller Core IP Symbol



6.2 User Parameters

The user parameters of this IP core are descibed in 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 PCIe Interrupt Controller Core in Vivado Design Suite.

Required Constraints

The XDC constraints are not provided with the PCIe Interrupt Controller 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 and Virtex-7 FPGAs.

Clock Frequencies

The clock frequency (**aclk**) 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

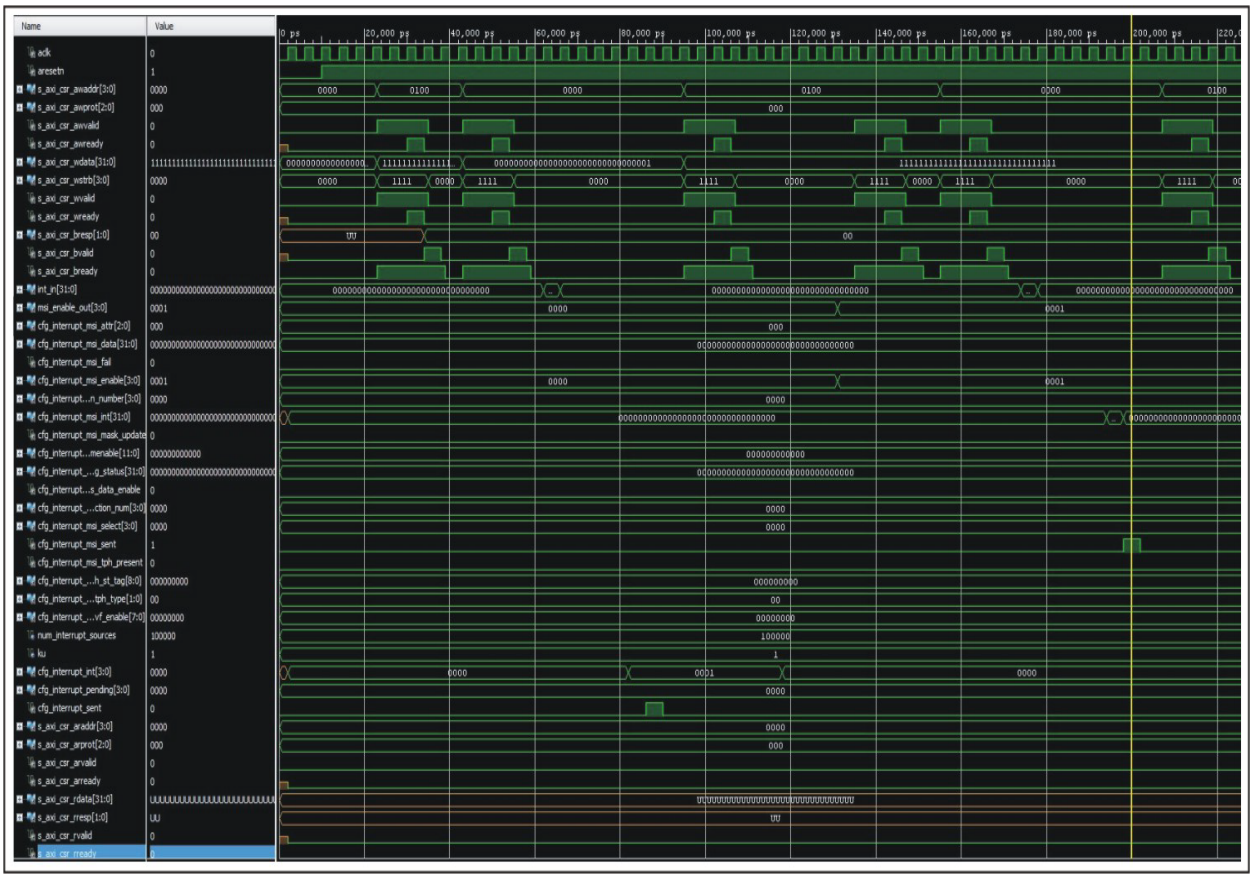
This section is not applicable for this IP core.

6.5 Simulation

The PCIe Interrupt Controller IP has a test bench which generates the output waveforms using Vivado VSim environment. The test bench is designed to run at 250 MHz input clock frequency with 32 interrupt sources. It is designed for a Kintex Ultrascale FPGA.

The test bench first generates Legacy interrupt signals by disabling the MSI mode (**cfg_interupt_msi_enable[0]** = '0') and then generates MSI signals by enabling the MSI mode (**cfg_interupt_msi_enable[0]** = '1'). 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: PCI Express Interrupt Controller 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*.