LogiCORE™ IP Endpoint v3.7 for PCI Express®

Getting Started Guide

UG430 April 19, 2010

XILINX®

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
2/17/07	3.4	Initial Xilinx release.
5/17/07	3.5	Updated to core version 3.5. Updated for PCI-SIG compliance.
4/19/10	4.0	Updated core to v3.7 and Xilinx ISE to v12.1.

Table of Contents

Appendix A: Additional Design Considerations

About This Guide

The *LogiCORE Endpoint v3.7 for PCI Express® Getting Started Guide* provides information about generating an Endpoint core for PCI Express (PCIe®), customizing and simulating the core using the provided example design, and running the design files through implementation using the Xilinx tools. The Endpoint core supports both Verilog® and VHDL; the example design presented in this guide is provided in Verilog.

Contents

This guide contains the following chapters:

- Preface, "About this Guide," introduces the organization and purpose of this guide, a list of additional resources, and the conventions used in this document.
- Chapter 1, "Introduction," describes the core and related information, including system requirements, recommended design experience, additional resources, technical support, and submitting feedback to Xilinx.
- Chapter 2, "Licensing the Core" describes the available license types and provides instructions for generating a license key.
- Chapter 3, "Quickstart Example Design," provides instructions for quickly generating, simulating, and implementing the example design using the demonstration test bench. In addition, the core directory structure for the Endpoint core and its associated files are defined.
- Appendix A, "Additional Design Considerations," provides additional considerations when implementing the example design.

Additional Resources

To find additional documentation, see the Xilinx website at:

http://www.xilinx.com/support/documentation/index.htm.

To search the Answer Database of silicon, software, and IP questions and answers, or to create a technical support WebCase, see the Xilinx website at:

http://www.xilinx.com/support/mysupport.htm.

Conventions

This document uses the following conventions. An example illustrates each convention.

Typographical

The following typographical conventions are used in this document:

Convention	Meaning or Use	Example
`Courier font`	Messages, prompts, and program files that the system displays	`speed grade: - 100`
`Courier bold`	Literal commands that you enter in a syntactical statement	**`ngdbuild`** *design_name*
Helvetica bold	Commands that you select from a menu	**File → Open**
	Keyboard shortcuts	**Ctrl+C**
Italic font	Variables in a syntax statement for which you must supply values	**`ngdbuild`** *design_name*
	References to other manuals	See the *User Guide* for more information.
	Emphasis in text	If a wire is drawn so that it overlaps the pin of a symbol, the two nets are *not* connected.
Dark Shading	Items that are not supported or reserved	This feature is not supported
Square brackets [ ]	An optional entry or parameter. However, in bus specifications, such as **`bus[7:0]`**, they are required.	**`ngdbuild`** [*option_name*] *design_name*
Braces { }	A list of items from which you must choose one or more	**`lowpwr =`**{**`on`**|**`off`**}
Vertical bar |	Separates items in a list of choices	**`lowpwr =`**{**`on`**|**`off`**}
Angle brackets < >	User-defined variable or in code samples	<directory name>

Vertical ellipsis
.
.
.
	Repetitive material that has been omitted	`IOB #1: Name = QOUT'`
		`IOB #2: Name = CLKIN'`
		.
		.
		.

Horizontal ellipsis ...	Repetitive material that has been omitted	**`allow block`** *block_name loc1 loc2 ... locn;*

Convention	Meaning or Use	Example
Notations	The prefix '0x' or the suffix 'h' indicate hexadecimal notation	A read of address 0x00112975 returned 45524943h.
	An '_n' means the signal is active low	**usr_teof_n** is active low.

Online Document

The following conventions are used in this document:

Convention	Meaning or Use	Example
Blue text	Cross-reference link to a location in the current document	See the section "Additional Resources" for details. Refer to "Title Formats" in Chapter 1 for details.
Blue, underlined text	Hyperlink to a website (URL)	Go to http://www.xilinx.com for the latest speed files.

XILINX®

Chapter 1

Introduction

The LogiCORE Endpoint for PCI Express (PCIe) is a high-bandwidth, scalable, and reliable serial interconnect building block for use with Virtex®-4 and Virtex-5. The core supports both Verilog and VHDL; the example design described in this guide is provided in Verilog.

This chapter introduces the core and provides related information, including system requirements, recommended design experience, additional resources, technical support, and submitting feedback to Xilinx.

About the Core

The Xilinx solutions for PCIe are reliable high-bandwidth scalable serial interconnect intellectual property building blocks for use with the Virtex-4 and Virtex-5 FPGA architectures. The solutions for PCIe support Verilog-HDL and VHDL.

For information about system requirements, installation, and licensing options, see Chapter 2, "Licensing the Core."

The Endpoint for PCIe is available through the Xilinx CORE Generator™. For additional information about the core, see the product page.

Recommended Design Experience

Although the Endpoint for PCIe is a fully verified solution, the challenge associated with implementing a complete design varies depending on the configuration and functionality of the application. For best results, previous experience building high performance, pipelined FPGA designs using Xilinx implementation software and User Constraints Files (UCF) is recommended.

Additional Core Resources

For detailed information and updates about the Endpoint for PCIe, see the following documents, available from the product page, unless otherwise noted.

- *LogiCORE Endpoint for PCI Express Data Sheet*
- *LogiCORE Endpoint for PCI Express User Guide*
- *LogiCORE Endpoint for PCI Express Release Notes file (available from the core directory after generating the core)*

Additional information and resources related to PCI Express technology are available from the following web sites:

- PCI Express at PCI-SIG
- PCI Express Developer's Forum

Technical Support

For technical support, go to www.xilinx.com/support. Questions are routed to a team of engineers with expertise using the PCIe Endpoint core.

Xilinx provides technical support for use of this product as described in the *LogiCORE Endpoint for PCI Express User Guide* and the *LogiCORE Endpoint for PCI Express Getting Started Guide*. Xilinx cannot guarantee timing, functionality, or support of this product for designs that do not follow these guidelines.

Feedback

Xilinx welcomes comments and suggestions about the Endpoint for PCIe core and the accompanying documentation.

Core

For comments or suggestions about the core, please submit a WebCase from www.xilinx.com/support. Be sure to include the following information:

- Product name
- Core version number
- Explanation of your comments

Document

For comments or suggestions about this document, please submit a WebCase from www.xilinx.com/support. Be sure to include the following information:

- Document title
- Document number
- Page number(s) to which your comments refer
- Explanation of your comments

XILINX®

Chapter 2

Licensing the Core

This chapter provides instructions for obtaining a license key for the Endpoint for PCI Express, which you must do before using it in your designs. The core is provided under the terms of the Xilinx LogiCORE Site License Agreement, which conforms to the terms of the SignOnce IP License standard defined by the Common License Consortium.

System Requirements

Windows

- Windows® 2000 Professional with Service Pack 2-4
- Windows XP Professional with Service Pack 1-2

Solaris/Linux

- Sun Solaris® 9/10
- Red Hat® Enterprise Linux 3.0 (32-bit and 64-bit)

Software

- ISE® 12.1

See www.xilinx.com/support/download/index.htm for the latest Xilinx design tools.

Before You Begin

This chapter assumes you have installed the core using either the CORE Generator IP Software Update installer or by performing a manual installation after downloading the core from the web. For information about installing the core, see the product page at www.xilinx.com/pciexpress.

For access to the LogiCORE Endpoint for PCI Express core for Virtex-II Pro, please contact Xilinx Technical Support at www.xilinx.com/support. Starting new designs with the LogiCORE Endpoint for PCI Express core for Virtex-II is discouraged.

License Options

The Endpoint core for PCIe provides three licensing options, described in the following sections. After installing the core, choose a license option.

Simulation Only

The Simulation Only Evaluation license is provided with the Xilinx CORE Generator. This license lets you assess the core functionality with either the provided example design or alongside your own design and demonstrates the various interfaces on the core in simulation. (Functional simulation is supported by a structural model provided by the CORE Generator.)

Full System Hardware Evaluation

The Full System Hardware Evaluation license is available at no cost and lets you fully integrate the core into an FPGA design, place and route the design, evaluate timing, and perform back-annotated gate-level simulation of the core using the demonstration test bench provided or your own test bench.

In addition, the license lets you generate a bitstream from the placed and routed design, which can then be downloaded to a supported device and tested in hardware. The core can be tested in the target device for a limited time before *timing out* (ceasing to function) at which time it can be reactivated by reconfiguring the device.

You can obtain the Full System Evaluation License in one of the following ways, depending on the core:

- By registering on the Xilinx IP Evaluation page and filling out a form to request an automatically generated evaluation license
- By contacting your local Xilinx Field Applications Engineer (FAE) to request a Full System Hardware Evaluation license key

Click Evaluate on the core product page for information about obtaining a Full System Hardware Evaluation License.

Full

The Full license is provided when you purchase the core, and provides full access to all core functionality both in simulation and in hardware, including:

- Functional simulation support
- Back annotated gate-level simulation support
- Full implementation support including place and route and bitstream generation
- Full functionality in the programmed device with no time outs

Obtaining Your License

Obtaining a Full System Hardware Evaluation License

To obtain a Full System Hardware Evaluation license, do the following:

- Navigate to the Endpoint product page at www.xilinx.com/pciexpress.
- Click Evaluate; then click Full System Hardware Evaluation.
- Follow the on-screen instructions to both download the CORE Generator files (delivered as an IP Update) and satisfy any additional requirements associated with the license.

Obtaining a Full License

To obtain a Full license, you must purchase the core. After purchase, you will receive a letter containing a serial number, which is used to register for access to the *lounge*, a secured area of the Endpoint product page.

- From the product page, click Register to register and request access to the lounge.
- Xilinx will review your access request and typically grants access to the lounge in 48 hours. (Contact Xilinx Customer Service if you need faster turnaround.)
- After you receive confirmation of lounge access, click Access Lounge on the Endpoint for PCI Express product page and log in.
- Follow the instructions in the lounge to fill out the license request form; then click Submit to automatically generate the license. An e-mail containing the license and installation instructions will be sent to you immediately.

Installing Your License File

The Simulation Only Evaluation license is provided with the CORE Generator and requires no license file. For the Full System Hardware Evaluation license and the Full license, you will receive an email containing instructions for installing your license file, as well as information about advanced licensing options and technical support.

XILINX®

Chapter 3

Quickstart Example Design

This chapter provides an overview of the Endpoint for PCI Express example design and instructions for generating the core. It also provides instructions for simulating and implementing the example design using the provided demonstration test bench.

Overview

The example simulation design consists of two discrete parts:

- The Endpoint for PCI Express Downstream Port Model, a test bench that generates, consumes, and checks PCIe bus traffic.
- The Programmed Input-Output (PIO) example design, a completer application. The PIO example design responds to PCIe Read and Write requests to its memory space and can be synthesized for testing in hardware.

Simulation Design Overview

For the example simulation design, transactions are sent from the Downstream Port Model to the Endpoint core and processed by the PIO example design. Figure 3-1 illustrates the simulation design provided with the core. For more information about the Downstream Port Model, see Appendix C, "Downstream Port Model Test Bench," in the *LogiCORE Endpoint for PCI Express User Guide*.

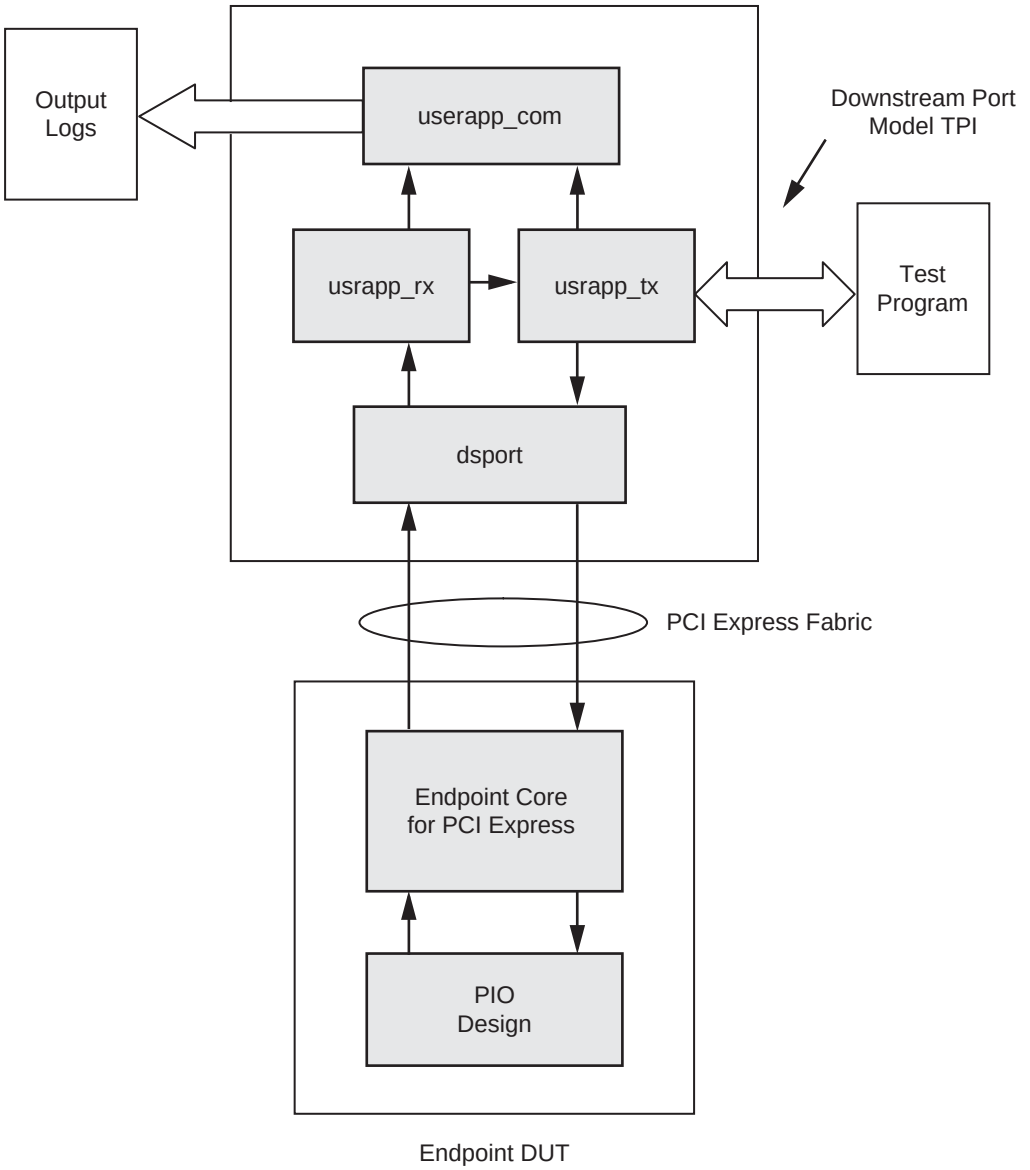

Figure 3-1: **Simulation Example Design Block Diagram**

Implementation Design Overview

The example implementation design consists of a simple programmed IO (PIO) example that can accept read and write transactions and respond to requests, as illustrated in Figure 3-2. Source code for this example is provided with the core. For more information about the PIO example design, see Appendix B, "Programmed Input Output Example Design," in the LogiCORE Endpoint for PCI Express User Guide.

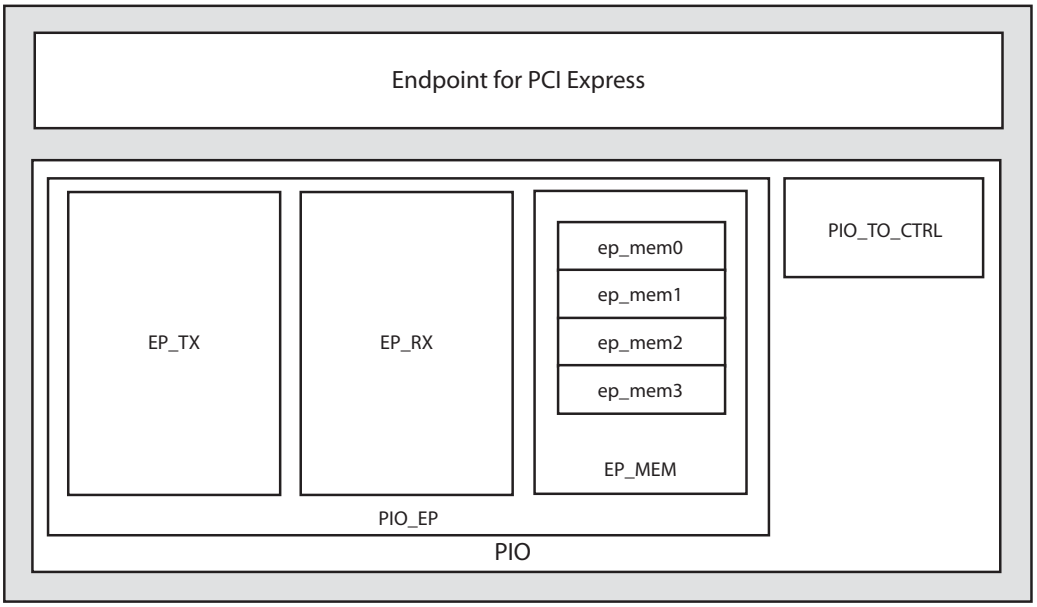

Figure 3-2: **Implementation Example Design Block Diagram**

Example Design Elements

The PIO example design elements include the following:

- Core netlists
- Core simulation models
- An example Verilog HDL wrapper (instantiates the cores and example design)
- A customizable demonstration test bench to simulate the example design

The example design has been tested and verified with Xilinx ISE v12.1 and the following simulators:

- Cadence™ Incisive Enterprise Simulator (IES) v9.2 and above
- Synopsys® VCS and VCS MX 2009.12 and above
- Mentor Graphics® ModelSim® v6.5c and above

Generating the Core

To generate a core using the default values in the CORE Generator Graphical User Interface (GUI), do the following:

1. Start the CORE Generator.

   For help starting and using the CORE Generator, see the Xilinx CORE Generator Guide, available from the ISE documentation web page.

2. Choose File > New Project. The New Project dialog box appears.

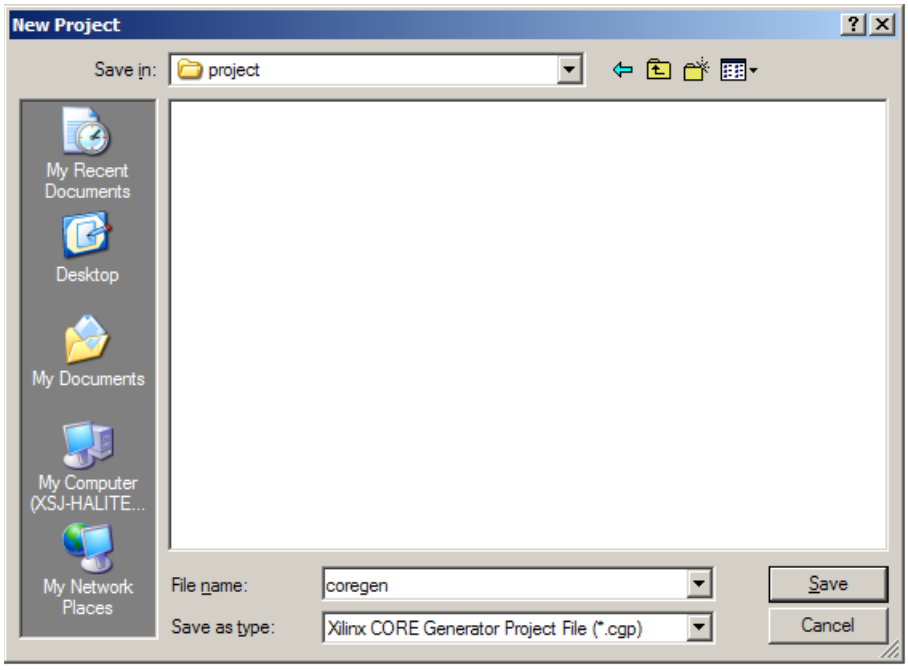

Figure 3-3: **New Project Dialog Box**

3. Enter a project name and location; then click OK. `<project_dir>` is used in this example. The Project Options dialog box appears.

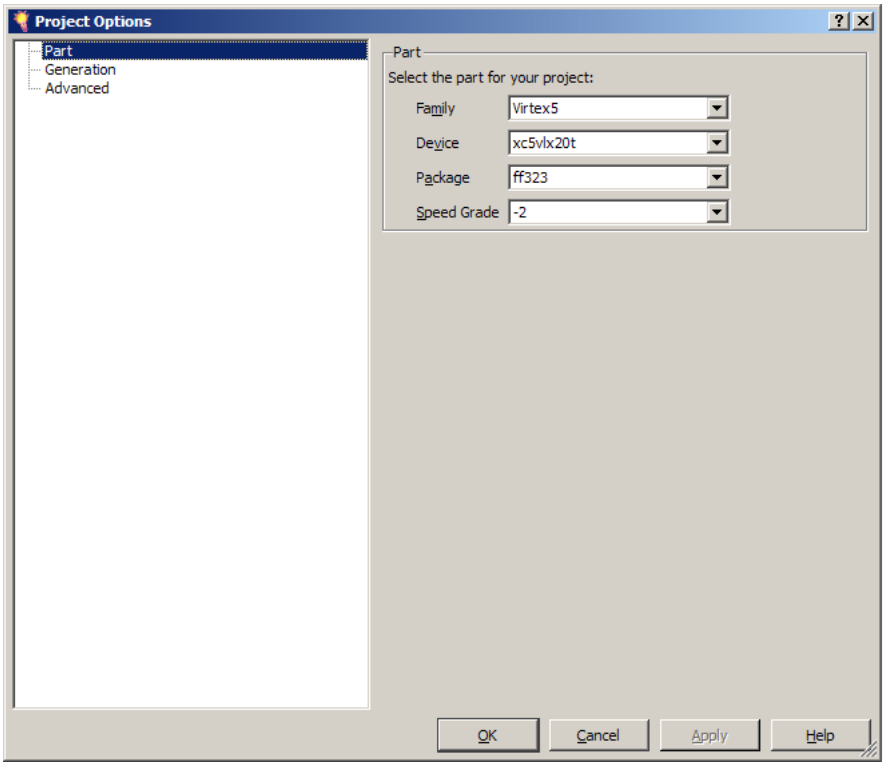

Figure 3-4: **Project Options**

4. Set the project options:

   From the Part tab, select the following options:

   - **Family**: Virtex4
   - **Device**: xc4vfx20
   - **Package**: ff672
   - **Speed Grade**: -10

   Note: If an unsupported silicon family is selected, the core is not available for customization and is dimmed in the list of selections.

   From the Generation tab, select the following parameters; then click OK.

   - **Design Entry**. Select either VHDL or Verilog. (The example design is provided for Verilog only.)
   - **Vendor**. Select Synplicity® or ISE (for XST).

5. Locate the core in the core selection tree under Standard Bus Interfaces/PCI Express. Double-click the core name to display the main Endpoint GUI screen.

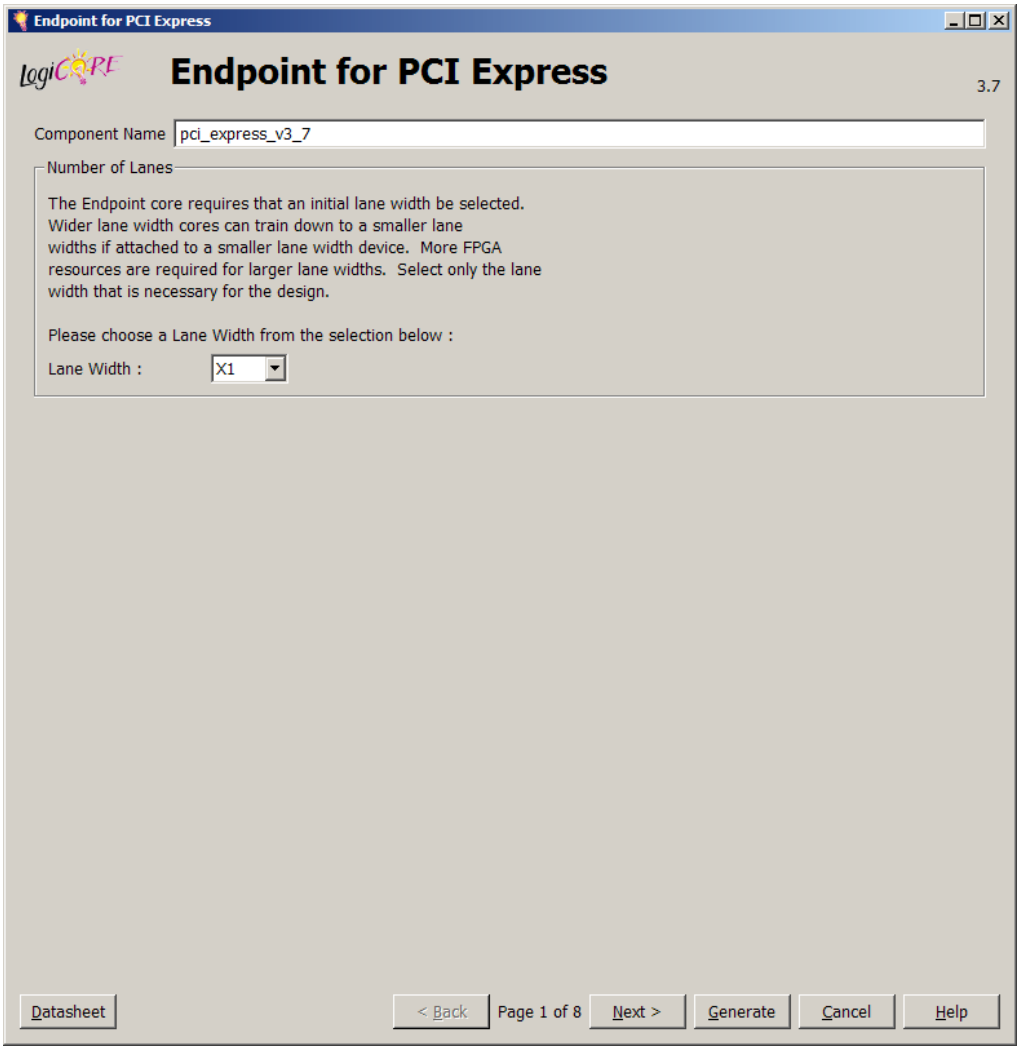

Figure 3-5: **Endpoint for PCI Express Main Screen**

6. In the Component Name field, enter a name for the core. `<component_name>` is used to represent the project name in this example.
7. Click Finish to generate the core using the default parameters. The core and its supporting files, including the PIO example design and Downstream Port Model test bench, are generated in the project directory.

For detailed information about the example design files and directories, see "Directory Structure and File Contents," page 22.

Simulating the Example Design

The example design provides a quick way to simulate the core and observe its behavior. The simulation environment provided with Endpoint core performs simple memory access tests on the PIO example design. Transactions are generated by the Downstream Port Model and responded to by the PIO example design.

- PCI Express Transaction Layer Packets (TLPs) are generated by the test bench transmit user application (`pci_exp_usrapp_tx`). As it transmits TLPs, it also generates a log file, `tx.dat`.
- PCI Express TLPs are received by the test bench receive user application (`pci_exp_usrapp_rx`). As the user application receives the TLPs, it generates a log file, `rx.dat`.

For more information about the test bench, see Appendix C, "Downstream Port Model Test Bench," in the *LogiCORE IP Endpoint for PCI Express User Guide*.

Setting up for Simulation

Simulation scripts are provided for Cadence IUS (VNC), Synopsys VCS, and ModelSim. Configure your environment to run the simulation tool of your choice.

Running the Simulation

The test bench provided with the example design supports pre-implementation mode (RTL) simulations:

- The test bench, along with RTL model of the example design
- The Verilog HDL model of the Endpoint core, created by the CORE Generator

1. To run the simulation, go to the following directory:

   `<project_dir>/<component_name>/simulation/functional`
2. Run the script that corresponds to your simulation tool using one of the following:
   - **VCS**: `simulate_vcs.sh`
   - **Verilog-NC**: `simulate_ncsim.sh`
   - **ModelSim**: `simulate_mti.do`

Implementing the Example Design

After generating the core, the netlists and the example design can be processed by the Xilinx implementation tools. The generated output files include scripts to assist the user in running the Xilinx software.

To implement the example design, open a command prompt or terminal window and type the following commands:

Windows

```
ms-dos> cd <project_dir>\<component_name>\implement
ms-dos> implement.bat
```

UNIX

```
unix-shell% cd <project_dir>/<component_name>/implement
unix-shell% ./implement.sh
```

These commands execute a script that synthesizes, builds, maps, and place-and-routes the example design. The script then generates a post-par simulation model for use in timing simulation. The resulting files are placed in the `results` directory and execute the following processes:

1. Removes data files from the previous runs.
2. Synthesizes the example design using either Synplicity Synplify or XST.
   - The core is instanced as a black box within the example design.
3. `ngdbuild`. Builds a Xilinx design database for the example design.

   Inputs:

   Part-Package-Speed Grade selection:

   XC4VFX20-FF672-10

   Example design UCF:

   xilinx_1_lane_32b_ep-XC4VFX20-FF672-10-COLUMN1.ucf
4. `map`: Maps design to the selected FPGA, using the constraints provided.
5. `par`: Places cells onto FPGA resources and routes connectivity.
6. `trce`: Performs static timing analysis, on design using constraints specified.
7. `netgen`: Generates a logical Verilog HDL representation of the design, and an SDF file, for post-layout verification.
8. `bitgen`: Generates a bitstream file for programming the FPGA.

The following FPGA implementation-related files are generated in the results directory:

- `routed.bit`
  FPGA configuration information
- `routed.v`
  Verilog functional Model
- `routed.sdf`
  Timing model Standard Delay File
- `mapped.mrp`
  Xilinx map report
- `routed.par`
  Xilinx place and route report

- `routed.twr`
  Xilinx timing analysis report

The script file starts from an EDIF/NGC file and results in a bitstream file. Although it is possible to use the Xilinx ISE GUI to implement the example design, the GUI flow is not included in this document.

Directory Structure and File Contents

This section defines the Endpoint for PCI Express core directory structure, including all directories and files associated with the core. Click a directory name below to go to the desired directory and its associated files.

<project directory>
Top-level project directory

<project directory>/<component name>
Core release notes file

<component name>/doc
Product documentation

<component name>/example_design
Verilog design files

<component name>/implement
Implementation script files

implement/results
Results directory, created after implementation scripts are run; contains implement script results.

<component name>/simulation
Simulation scripts

simulation/dsport
Simulation files

simulation/functional
Functional simulation files

simulation/tests
Test command files

<project directory>

The project directory contains all the CORE Generator project files.

Table 3-1: **Project Directory**

Name	Description
<project_dir>	
<component_name>.ngc	Top-level netlist.
<component_name>.v[hd]	Verilog or VHDL simulation model.
<component_name>.xco	CORE Generator project-specific option file; can be used as an input to the CORE Generator.
<component_name>_flist.txt	List of files delivered with core.
<component_name>.{veo|vho}	VHDL or Verilog instantiation template.

Back to Top

<project directory>/<component name>

The <component name> directory contains the release notes file included with the core, which contains last-minute changes and/or updates.

Table 3-2: **Component Name Directory**

Name	Description
<project_dir>/<component_name>	
pci_express_release_notes.txt	Release notes file for the Endpoint core.

Back to Top

<component name>/doc

The doc directory contains the PDF documentation included with the core.

Table 3-3: **Doc Directory**

Name	Description
<project_dir>/<component_name>/doc	
pci_exp_ep_ds506.pdf	*LogiCORE IP Endpoint for PCI Express Data Sheet*
pci_exp_ep_gsg430.pdf	*LogiCORE IP Endpoint for PCI Express Getting Started Guide*
pci_exp_ep_ug185.pdf	*LogiCORE IP Endpoint for PCI Express User Guide*

Back to Top

<component name>/example_design

The example design directory contains all the example design files included with the core.

Table 3-4: **Example Design Directory**

Name	Description
<project_dir>/<component_name>/example_design	
pci_exp_8_lane_64b_ep.v pci_exp_4_lane_32b_ep.v pci_exp_1_lane_32b_ep.v	Verilog top-level example design, applicable to the 8-lane, 4-lane, and 1-lane endpoint design, respectively.
<filename>.ucf	Example design UCF. Filename varies by family, part, and package selected.
xilinx_pci_exp_8_lane_64b_ep_product.v xilinx_pci_exp_4_lane_32b_ep_product.v xilinx_pci_exp_1_lane_32b_ep_product.v	Enables 8-lane, 4-lane, or 1-lane Endpoint in the test bench, respectively.
xilinx_pci_exp_8_lane_64b_ep.v xilinx_pci_exp_4_lane_32b_ep.v xilinx_pci_exp_1_lane_64b_ep.v	Example design wrapper file for 8-lane, 4-lane, and 1-lane example design, respectively.
virtex4_xst.v virtex4fx.v	Black box definitions.
impl_test.v pci_exp_32b_app.v pci_exp_64b_app.v EP_MEM.v PIO.v PIO_32.v PIO_32_RX_ENGINE.v PIO_32_TX_ENGINE.v PIO_64.v PIO_64_RX_ENGINE.v PIO_64_TX_ENGINE.v PIO_EP.v PIO_EP_MEM_ACCESS.v PIO_TO_CTRL.v	PIO example design files.

Back to Top

<component name>/implement

The implement directory contains all the implementation script files.

Table 3-5: **Implement Directory**

Name	Description
<project_dir>/<component_name>/implement	
implement.bat implement.sh	DOS and UNIX/Linux implementation scripts.
synplify.prj	Synplify synthesis script.
xilinx_pci_exp_1_lane_32b_ep.xcf xilinx_pci_exp_4_lane_32b_ep.xcf xilinx_pci_exp_8_lane_64b_ep.xcf	XST constraints files for example designs.
xilinx_pci_exp_1_lane_32b_ep_inc.xst xilinx_pci_exp_4_lane_32b_ep_inc.xst xilinx_pci_exp_8_lane_64b_ep_inc.xst	XST project file for example designs.
xst.scr	XST synthesis script.

Back to Top

implement/results

The results directory is created by the implement script; implement script results are placed in the results directory.

Table 3-6: **Results Directory**

Name	Description
<project_dir>/<component_name>/implement/results	
Created by implement script; implement script results are placed in the results directory.	

Back to Top

<component name>/simulation

The simulation directory contains source files for the provided test bench.

Table 3-7: **Simulation Directory**

Name	Description
<project_dir>/<component_name>/simulation	
board_common.v	Contains test bench definitions.
board.v	Top-level simulation module and loop-back.
sys_clk_gen_ds.v	System differential clock source.
sys_clk_gen.v	System clock source.

Table 3-7: **Simulation Directory *(Cont'd)***

Name	Description
xilinx_pci_exp_cor_ep.f	List of files comprising the design being tested.
xilinx_pci_exp_defines.v	Application macro definitions.

Back to Top

simulation/dsport

The functional directory contains files for the Downstream Port Model test bench.

Table 3-8: **dsport Directory**

Name	Description
<project_dir>/<component_name>/simulation/dsport	
dsport_cfg.v pci_exp_1_lane_64b_dsport.v pci_exp_4_lane_64b_dsport.v pci_exp_expect_tasks.v pci_exp_usrapp_cfg.v pci_exp_usrapp_com.v pci_exp_usrapp_rx.v pci_exp_usrapp_tx.v xilinx_pci_exp_downstream_port.v xilinx_pci_exp_dsport.v	Downstream Port Model files.

Back to Top

simulation/functional

The functional directory contains input files for function simulations.

Table 3-9: **Functional Directory**

Name	Description
<project_dir>/<component_name>/simulation/functional	
board_rtl_x01_32b_v4fx.f board_rtl_x04_32b_v4fx.f board_rtl_x08_64b_v4fx.f board_rtl_x01_32b_v4fx_ncv.f board_rtl_x04_32b_v4fx_ncv.f board_rtl_x08_64b_v4fx_ncv.f	List of files for RTL simulation.
simulate_mti.do	Simulation script for ModelSim.
simulate_ncsim.sh	Simulation script for NCverilog.
simulate_vcs.sh	Simulation script for VCS.
xilinx_lib.f	Xilinx libraries; requires modification to point to the user location of Xilinx libraries.

Table 3-9: **Functional Directory *(Cont'd)***

Name	Description
wave_ncsim.sv	Default list of TRN interface, system, and PCI Express signals available for view through ncsim simulation.
xilinx_lib_mti_v4fx.v xilinx_lib_vcs_v4fx.v xilinx_lib_vnc_v4fx.v	Points to the MGT SmartModel.

Back to Top

simulation/tests

The tests directory contains test definitions for the example test bench.

Table 3-10: **Functional Directory**

Name	Description
<project_dir>/<component_name>/simulation/tests	
pio_tests.v sample_tests1.v tests.v	Test files for the Downstream Port Model testbench.

Back to Top

XILINX®

Appendix A

Additional Design Considerations

Package Constraints

This section discusses design considerations specific to the Endpoint core targeting Virtex-4 and Virtex-5 FPGA devices. Table A-1 defines the smallest supported device and interface combinations for the example design.

Table A-1: **Supported Device and Interface Combinations**

Link Width/Data Bus Width/Speed	Smallest Supported Device/Part Number	Wrapper File
Width: 8-lane Width: 64-bit Port Speed: 250MHz	XC4VFX60 FF1152-11 XC5VLX50T FF1136-1	xilinx_pci_exp_8_lane_64b_ep.v
Width: 4-lane Width: 32-bit Port Speed: 250MHz	XC4VFX60 FF672-11 XC5VLX50T FF1136-1	xilinx_pci_exp_4_lane_32b_ep.v
Width: 1-lane Width: 32-bit Port Speed: 62.5MHz	XC4VFX20 FF672-10 XC5VLX50T FF1136-1	xilinx_pci_exp_1_lane_32b_ep.v

User Constraints Files

The UCF contains various constraints required for the Endpoint core, and must always be used while processing a design and is specific to the target device. Based on the chosen lane width and device, a suitable UCF file is created by the CORE Generator in the `<project_dir>/<component_name>/example_design` directory.

Wrapper File Usage

The wrapper contains an instance of the Endpoint core. When starting a new design, modify this wrapper to include all I/O elements and modules. One of the following files is generated by the CORE Generator based on the selected lane width:

`<project_dir>/<component_name>/example_design/xilinx_pci_exp_8_lane_64b_ep.v`

`<project_dir>/<component_name>/example_design/xilinx_pci_exp_4_lane_32b_ep.v`

`<project_dir>/<component_name>/example_design/xilinx_pci_exp_1_lane_32b_ep.v`