LogiCORE™ PCI Express® Endpoint Block Plus v1.2

Getting Started Guide

UG343 February 15, 2007

Xilinx does not assume any liability arising out of the application or use of the Specification; nor does Xilinx convey any license under its patents, copyrights, or any rights of others. You are responsible for obtaining any rights you may require for your use or implementation of the Specification. Xilinx reserves the right to make changes, at any time, to the Specification as deemed desirable in the sole discretion of Xilinx. Xilinx assumes no obligation to correct any errors contained herein or to advise you of any correction if such be made. Xilinx will not assume any liability for the accuracy or correctness of any engineering or technical support or assistance provided to you in connection with the Specification.

THE SPECIFICATION IS PROVIDED “AS IS" WITH ALL FAULTS, AND THE ENTIRE RISK AS TO ITS FUNCTION AND IMPLEMENTATION IS WITH YOU. YOU ACKNOWLEDGE AND AGREE THAT YOU HAVE NOT RELIED ON ANY ORAL OR WRITTEN INFORMATION OR ADVICE, WHETHER GIVEN BY XILINX, OR ITS AGENTS OR EMPLOYEES. XILINX MAKES NO OTHER WARRANTIES, WHETHER EXPRESS, IMPLIED, OR STATUTORY, REGARDING THE SPECIFICATION, INCLUDING ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE, AND NONINFRINGEMENT OF THIRD-PARTY RIGHTS.

IN NO EVENT WILL XILINX BE LIABLE FOR ANY CONSEQUENTIAL, INDIRECT, EXEMPLARY, SPECIAL, OR INCIDENTAL DAMAGES, INCLUDING ANY LOST DATA AND LOST PROFITS, ARISING FROM OR RELATING TO YOUR USE OF THE SPECIFICATION, EVEN IF YOU HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THE TOTAL CUMULATIVE LIABILITY OF XILINX IN CONNECTION WITH YOUR USE OF THE SPECIFICATION, WHETHER IN CONTRACT OR TORT OR OTHERWISE, WILL IN NO EVENT EXCEED THE AMOUNT OF FEES PAID BY YOU TO XILINX HEREUNDER FOR USE OF THE SPECIFICATION. YOU ACKNOWLEDGE THAT THE FEES, IF ANY, REFLECT THE ALLOCATION OF RISK SET FORTH IN THIS AGREEMENT AND THAT XILINX WOULD NOT MAKE AVAILABLE THE SPECIFICATION TO YOU WITHOUT THESE LIMITATIONS OF LIABILITY.

The Specification is not designed or intended for use in the development of on-line control equipment in hazardous environments requiring fail-safe controls, such as in the operation of nuclear facilities, aircraft navigation or communications systems, air traffic control, life support, or weapons systems (“High-Risk Applications”). Xilinx specifically disclaims any express or implied warranties of fitness for such High-Risk Applications. You represent that use of the Specification in such High-Risk Applications is fully at your risk.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
10/23/06	1.1	Initial Xilinx release.
2/15/07	2.0	Update core to version 1.2; Xilinx tools 9.1i.

Table of Contents

Preface: About This Guide

Chapter 1: Introduction

Chapter 2: Licensing the Core

Chapter 3: Quickstart Example Design

Appendix: Additional Design Considerations

About This Guide

The *PCI Express® Endpoint Block Plus Getting Started Guide v1.2* provides information about generating a PCI Express Endpoint Block Plus (PCIe® Block Plus) core, customizing and simulating the core using the provided example design, and running the design files through implementation using the Xilinx tools.

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" provides instructions for selecting a license option for the core.
- Chapter 3, "Quickstart Example Design," provides instructions for quickly generating, simulating, and implementing the example design using the demonstration test bench.
- Appendix, "Additional Design Considerations," defines additional considerations when implementing the example design.

Additional Resources

For additional information, go to www.xilinx.com/support. The following table lists some of the resources you can access from this website or by using the provided URLs.

Resource	Description/URL
Tutorials	Tutorials covering Xilinx design flows, from design entry to verification and debugging. www.xilinx.com/support/techsup/tutorials/index.htm
Answer Browser	Database of Xilinx solution records. www.xilinx.com/xlnx/xil_ans_browser.jsp
Data Sheets	Device-specific information about Xilinx device characteristics, including readback, boundary scan, configuration, length count, and debugging. www.xilinx.com/xlnx/xweb/xil_publications_index.jsp

Resource	Description/URL
Problem Solvers	Interactive tools that allow you to troubleshoot your design issues. www.xilinx.com/support/troubleshoot/psolvers.htm
Tech Tips	Latest news, design tips, and patch information for the Xilinx design environment. www.xilinx.com/xlnx/xil_tt_home.jsp

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 you enter in a syntactical statement	**`ngdbuild`** `design_name`
Italic font	Variables in a syntax statement for which you must supply values	See the *Development System Reference Guide* for more information.
	References to other manuals	See the *User Guide* for details.
	Emphasis in text	If a wire is drawn so that it overlaps the pin of a symbol, the two nets are *not* connected.
<text in brackets>	User-defined variable for directory names.	<component_name>
Dark Shading	Items that are not supported or reserved	Unsupported feature
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}`**

Convention	Meaning or Use	Example

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

Horizontal ellipsis . . .	Omitted repetitive material	**`allow block`** `block_name loc1 loc2 ... locn;`
Notations	The prefix '`0x`' or the suffix '`h`' indicate hexadecimal notation	A read of address `0x00112975` returned `45524943`h.
	An '`_n`' means the signal is active low	`usr_teof_n` is active low.

Online Document

The following linking 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. See "Title Formats" in Chapter 1 for details.
Blue, underlined text	Hyperlink to a website (URL)	Go to www.xilinx.com for the latest speed files.

Chapter 1

Introduction

The PCIe Block Plus core is a high-bandwidth, scalable, and reliable serial interconnect building block for use with Virtex™-5 FPGA devices. This core supports 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 PCIe Block Plus core is a Xilinx CORE Generator™ IP core, included in the latest IP Update on the Xilinx IP Center. For additional information about the core, see the PCIe Block Plus product page. For information about obtaining a license for the core, see Chapter 2, "Licensing the Core."

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™ 9.1i with applicable Service Pack

See the PCIe Block Plus release notes file for the required Service Pack; ISE Service Packs can be downloaded from www.xilinx.com/xlnx/xil_sw_updates_home.jsp?update=sp.

Recommended Design Experience

Although the PCIe Block Plus core 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 PCIe Block Plus core, see the following documents, available from the PCIe Block Plus product page unless otherwise noted.

- *PCI Express Endpoint Block Plus Data Sheet*
- *PCI Express Endpoint Block Plus User Guide*
- *PCI Express Endpoint Block Plus Release Notes* (available from the core directory after generating the core)
- *Virtex-5 PCI Express Endpoint Block User Guide* (UG197)

Additional information and resources related to the 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 Block Plus core.

Xilinx provides technical support for use of this product as described in the *PCI Express Endpoint Block Plus User Guide* and the *PCI Express Endpoint Block Plus 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 PCIe Block Plus core and the accompanying documentation.

Core

For comments or suggestions about the PCIe Block Plus 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

Chapter 2

Licensing the Core

This chapter provided licensing options for the PCIe Block Plus core, which you must do before using the core in your designs. The PCIe Block Plus 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.

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 additional information about installing the core, see the PCIe Block Plus product page.

License Options

The PCIe Block Plus core provides two license options: a Simulation Only Evaluation license and a Full license. The Simulation Only Evaluation license is provided by default with the CORE Generator. The Full license is also free of charge, but you must register for it on the PCIe Block Plus product page. See "Obtaining a Full License" below.

Simulation Only

The Simulation Only Evaluation license, provided by default with CORE Generator, lets you assess the core functionality with either the provided example design or alongside your own design, and demonstrates the various interfaces to the core in simulation. (Functional simulation is supported by a dynamically generated gate-level netlist).

Full

The Full license provides full access to all core functionality both in simulation and in hardware, including:

- Gate-level 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

Simulation Only Evaluation License

The Simulation Only Evaluation license is provided with the CORE Generator system and requires no license file.

Obtaining a Full License

To obtain a Full license, you must register for access to the *lounge*, a secured area of the PCIe Block Plus product page.

- From the product page, click Register to register and request access to the lounge. Access to the lounge is automatic and granted immediately.
- After you receive confirmation of lounge access, click Access Lounge from the product page and log in.
- Click Access Lounge on the product lounge page and fill out the license request form linked from this location; then click Submit to automatically generate the license. An e-mail containing the license and installation instructions will be sent immediately to the email address you specified.

Installing Your License File

If you select the Full License option, you will receive an email that includes instructions for installing your license file. In addition, information about advanced licensing options and technical support is provided.

Chapter 3

Quickstart Example Design

This chapter provides an overview of the PCIe Block Plus example design and instructions for generating the core. It also covers simulating and implementing the example design using the provided demonstration test bench.

Overview

The example simulation design consists of two discrete parts:

- The PCI Express Downstream Port Model, a test bench that generates, consumes, and checks PCI Express bus traffic.
- The Programmed Input-Output (PIO) example design, a PCIe 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 simulation design, transactions are sent from the Downstream Port Model to the PCIe Block Plus core and processed by the PIO example design. Figure 3-1 illustrates the simulation example design provided with the PCIe Block Plus core. For more information about the Downstream Port Model, see Appendix D, "PCI Express Endpoint Downstream Model Test Bench," in the *PCI Express Endpoint Block Plus User Guide*.

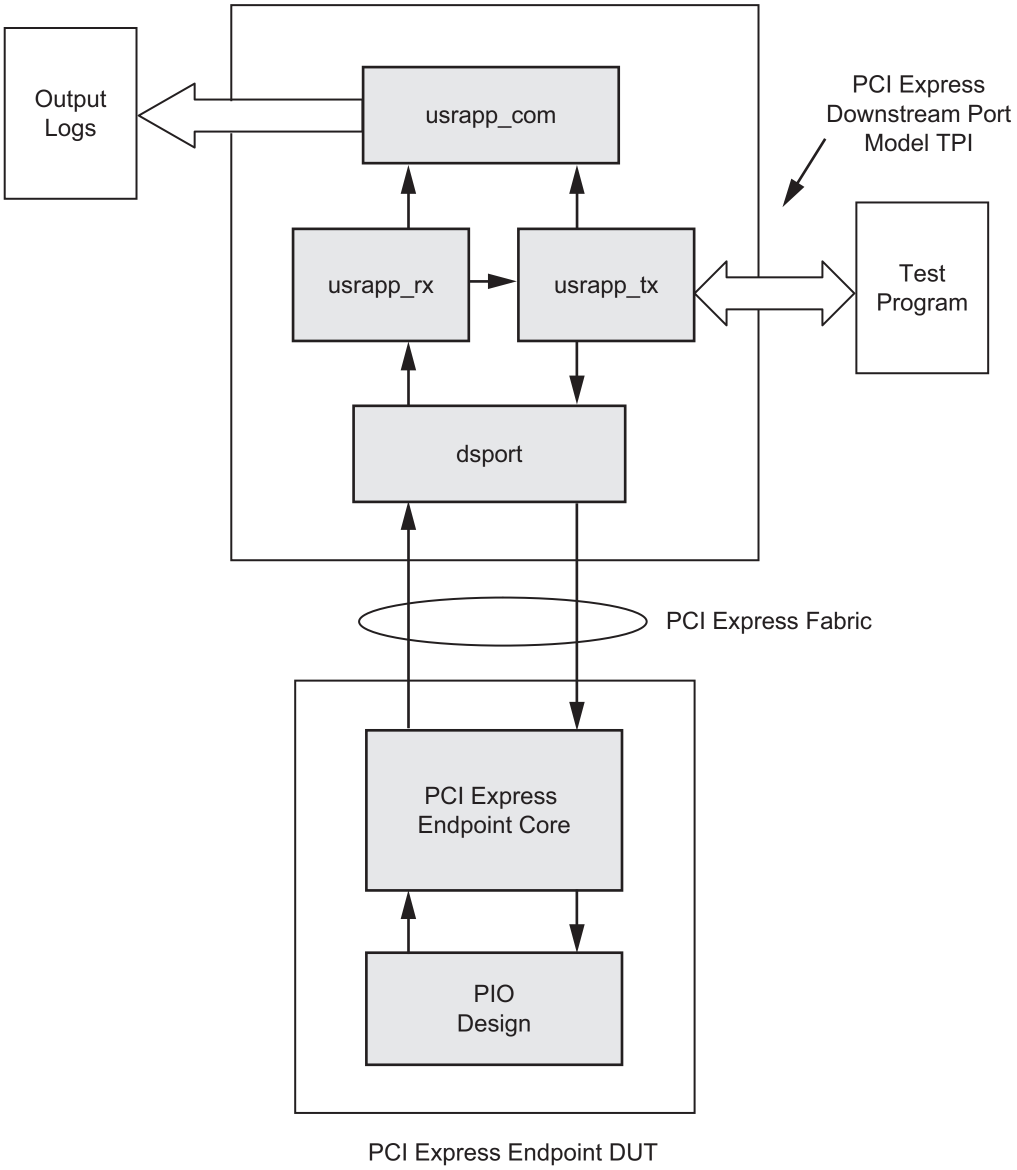


Figure 3-1: **Simulation Example Design Block Diagram**

Implementation Design Overview

The 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 C, "PCI Express Endpoint PIO Example Design," in the *PCI Express Endpoint Block Plus User Guide*.

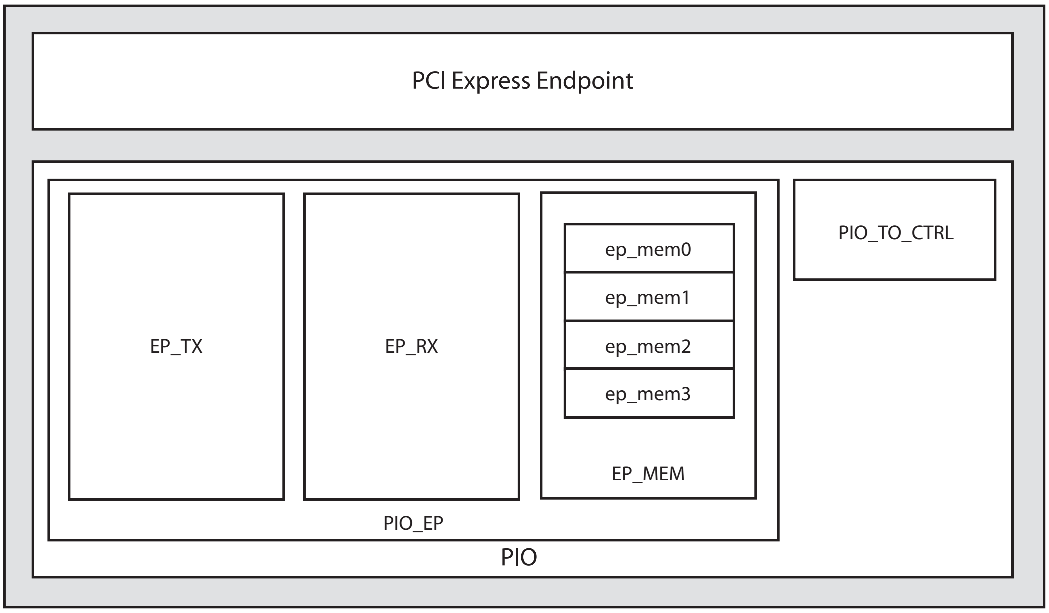


Figure 3-2: **Implementation Example Design Block Diagram**

Example Design Elements

The PCIe Block Plus 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 PCIe Block Plus example design has been tested and verified with Xilinx ISE v8.2i and the following simulators:

- Cadence® IUS 5.7 or higher
- Synopsys® VCS X2006.06
- MTI ModelSim® v6.1e

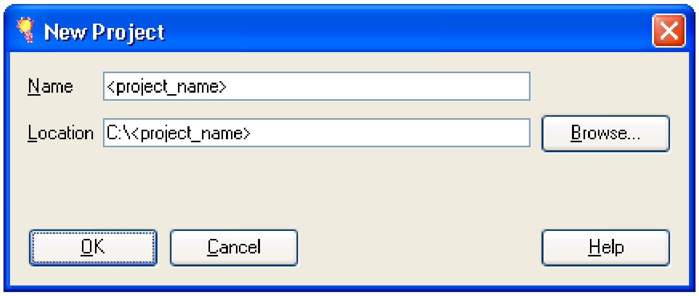
Generating the Core

To generate a PCIe Block Plus 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.

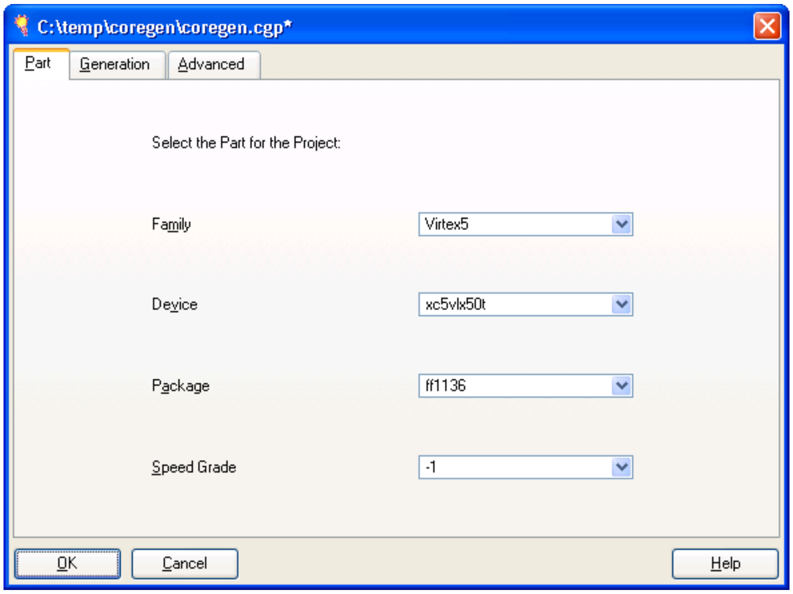
2. Choose File > New Project. The New Project dialog box appears.



Name: <project_name>
Location: C:\<project_name>

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.



Select the Part for the Project:

Family: Virtex5
Device: xc5vlx50t
Package: ff1136
Speed Grade: -1

Figure 3-4: **Project Options**

4. Set the project options:

   From the Part tab, select the following options:

   - **Family**: Virtex5
   - **Device**: xc5vlx50t
   - **Package**: ff1136

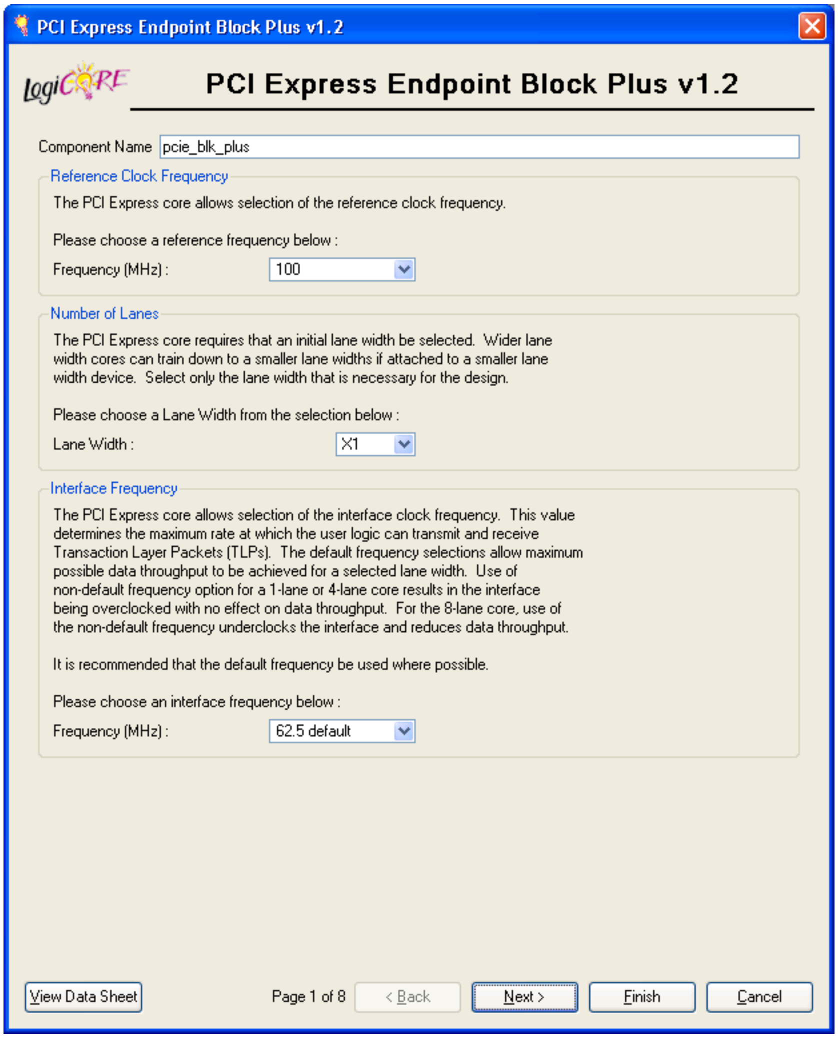
- **Speed Grade**: -1

Note: If an unsupported silicon device is selected, the PCIe Block Plus core is not available for customization and is dimmed in the list of cores.

From the Generation tab, select the following parameters:

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

5. Click OK.
6. Locate the PCIe Block Plus core in the core selection tree under Standard Bus Interfaces/PCI Express. Double-click the core name to display the PCIe Block Plus main screen.



PCI Express Endpoint Block Plus v1.2

PCI Express Endpoint Block Plus v1.2

Component Name pcie_blk_plus

Reference Clock Frequency

The PCI Express core allows selection of the reference clock frequency.

Please choose a reference frequency below :

Frequency (MHz) : 100

Number of Lanes

The PCI Express core requires that an initial lane width be selected. Wider lane width cores can train down to a smaller lane widths if attached to a smaller lane width device. Select only the lane width that is necessary for the design.

Please choose a Lane Width from the selection below :

Lane Width : X1

Interface Frequency

The PCI Express core allows selection of the interface clock frequency. This value determines the maximum rate at which the user logic can transmit and receive Transaction Layer Packets (TLPs). The default frequency selections allow maximum possible data throughput to be achieved for a selected lane width. Use of non-default frequency option for a 1-lane or 4-lane core results in the interface being overclocked with no effect on data throughput. For the 8-lane core, use of the non-default frequency underclocks the interface and reduces data throughput.

It is recommended that the default frequency be used where possible.

Please choose an interface frequency below :

Frequency (MHz) : 62.5 default

Figure 3-5: **PCI Express Endpoint Block Plus Main Screen**

7. In the Component Name field, enter a name for the core. `<component_name>` is used in this example.

8. Click Finish to generate the PCIe Block Plus 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 20.

Simulating the Example Design

The example design provides a quick way to simulate and observe the behavior of the core. The simulation environment provided with PCIe Block Plus 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 D, "PCI Express Endpoint Downstream Model Test Bench," in the *PCI Express Endpoint Block Plus User Guide*.

Setting up for Simulation

Simulation scripts are provided for Cadence NC Verilog (VNC), Synopsys VCS, and Mentor Graphics ModelSim (MTI). Set 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 PCIe Block Plus 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`
   - **Cadence IUS**: `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 PCIe Block Plus example design, open a command prompt or terminal window and type the following:

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 PCIe Block Plus example design using either Synplicity Synplify Pro or XST.

   - The PCIe Block Plus 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:

   XC5VLX50T-FF1136-1

   Example design UCF:

   `xilinx_pci_exp_blk_plus_1_lane_ep-XC5VLX50T-FF1136-1.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. It is possible to use the Xilinx ISU GUI to implement the example design. However, the GUI flow is not presented in this document.

Directory Structure and File Contents

The PCIe Block Plus core directories and their associated files are defined in the sections that follow. Click a directory name to go to the desired directory and its associated files.

- **<project directory>**
  Top-level project directory; name is user-defined
  - <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, and 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 provided with the core, which may include last-minute changes and updates, and information about required tools.

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

Name	Description
`<project_dir>/<component_name>`	
`pcie_blk_plus_release_notes.txt`	PCIE Block Plus release notes file.

Back to Top

<component name>/doc

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

Table 3-3: **Doc Directory**

Name	Description
`<project_dir>/<component_name>/doc`	
`pcie_blk_plus_ds551.pdf`	*PCI Express Endpoint Block Plus Data Sheet*
`pcie_blk_plus_gsg343.pdf`	*PCI Express Endpoint Block Plus Getting Started Guide*
`pcie_blk_plus_ug341.pdf`	*PCI Express Endpoint Block Plus User Guide*

Back to Top

<component name>/example_design

The example design directory contains the example design files provided 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_64b_ep.v pci_exp_1_lane_64b_ep.v	Verilog top-level example design, applicable to the 8-lane, 4-lane, and 1-lane endpoint design, respectively.
xilinx_pci_exp_blk_plus_1_lane_ep-XC5VLX50T-FF1136-1.ucf	PCIe Block Plus 1-lane example design UCF.
xilinx_pci_exp_blk_plus_1_lane_ep-XC5VLX110T-FF1136-1.ucf	PCIe Block Plus 1-lane example design UCF.
xilinx_pci_exp_blk_plus_1_lane_ep-XC5VLX330T-FF1738-1.ucf	PCIe Block Plus 1-lane example design UCF.
xilinx_pci_exp_blk_plus_4_lane_ep-XC5VLX50T-FF1136-1.ucf	PCIe Block Plus 4-lane example design UCF.
xilinx_pci_exp_blk_plus_4_lane_ep-XC5VLX110T-FF1136-1.ucf	PCIe Block Plus 4-lane example design UCF.
xilinx_pci_exp_blk_plus_4_lane_ep-XC5VLX330T-FF1738-1.ucf	PCIe Block Plus 4-lane example design UCF.
xilinx_pci_exp_blk_plus_8_lane_ep-XC5VLX50T-FF1136-1.ucf	PCIe Block Plus 8-lane example design UCF.
xilinx_pci_exp_blk_plus_8_lane_ep-XC5VLX110T-FF1136-1.ucf	PCIe Block Plus 8-lane example design UCF.
xilinx_pci_exp_blk_plus_8_lane_ep-XC5VLX330T-FF1738-1.ucf	PCIe Block Plus 8-lane example design UCF.
xilinx_pci_exp_8_lane_ep_product.v xilinx_pci_exp_4_lane_ep_product.v xilinx_pci_exp_1_lane_ep_product.v	Enables 8-lane, 4-lane, and 1-lane PCIe Block Plus, respectively, in the test bench.
xilinx_pci_exp_8_lane_ep.v xilinx_pci_exp_4_lane_ep.v xilinx_pci_exp_1_lane_ep.v	Top-level PIO example design files for 8-lane, 4-lane, and 1-lane cores.
pci_exp_64b_app.v EP_MEM.v PIO.v PIO_EP.v PIO_EP_MEM_ACCESS.v PIO_TO_CTRL.v PIO_64.v PIO_64_RX_ENGINE.v PIO_64_TX_ENGINE.v	PIO example design files.

Back to Top

<component name>/implement

The implement directory contains the core implementation script files.

Table 3-5: **Implement Directory**

Name	Description
<project_dir>/<component_name>/implement	
xst.scr	XST synthesis script.
implement.bat implement.sh	DOS and UNIX/Linux implementation scripts.
synplify.prj	Synplify synthesis script.
xilinx_pci_exp_1_lane_ep_inc.xst xilinx_pci_exp_4_lane_ep_inc.xst xilinx_pci_exp_8_lane_ep_inc.xst	XST project file for 1-lane, 4-lane, and 8-lane example design, respectively.

Back to Top

implement/results

The results directory is created by the implement script, after which the implement script results are placed in the results directory.

Table 3-6: **Results Directory**

Name	Description
<project_dir>/<component_name>/implement/results	
Implement script result files.	

Back to Top

<component name>/simulation

The simulation directory contains the simulation source files provided with the core.

Table 3-7: **Simulation Directory**

Name	Description
<project_dir>/<component_name>/simulation	
board_common.v	Contains PCI Express 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.
xilinx_pci_exp_cor_ep.f	List of files comprising the design being tested.

Table 3-7: **Simulation Directory *(Continued)***

Name	Description
xilinx_pci_exp_defines.v	PCI Express application macro definitions.

Back to Top

simulation/dsport

The dsport directory contains the data stream simulation scripts provided with the core.

Table 3-8: **dsport Directory**

Name	Description
<project_dir>/<component_name>/simulation/dsport	
dsport_cfg.v pci_exp_expect_tasks.v pci_exp_1_lane_64b_dsport.v pci_exp_4_lane_64b_dsport.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 functional simulation scripts provided with the core.

Table 3-9: **Functional Directory**

Name	Description
<project_dir>/<component_name>/simulation/functional	
board_rtl_x01_v4fx.f board_rtl_x04_v4fx.f board_rtl_x08_v4fx.f board_rtl_x01_v4fx_ncv.f board_rtl_x04_v4fx_ncv.f board_rtl_x08_v4fx_ncv.f	List of files for RTL simulations.
simulate_mti.do	Simulation script for ModelSim.
simulate_ncsim.sh	Simulation script for NCverilog.
simulate_vcs.sh	Simulation script for VCS.

Table 3-9: **Functional Directory *(Continued)***

Name	Description
xilinx_lib_mti.v xilinx_lib_vcs.v xilinx_lib_vnc.v	Points to the required SmartModel.

Back to Top

simulation/tests

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

Table 3-10: **tests Directory**

Name	Description
<project_dir>/<component_name>/simulation/tests	
pio_tests.v sample_tests1.v tests.v	Test definitions for example test bench.

Back to Top

Appendix

Additional Design Considerations

Package Constraints

This section discusses design considerations specific to the PCIe Block Plus core. Table A-1 lists the smallest supported device and interface combinations for the PCI Express Endpoint Block Plus core.

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

Smallest Supported Device/Part Number	Data Bus Width/Speed	Wrapper File
XC5VLX50T FF1136-1	Width: 64-bit Port Speed: 62.5 MHz	xilinx_pci_exp_1_lane_ep.v

User Constraints Files

The user constraints file (UCF) contains various constraints required for the PCIe Block Plus core. The user constraints file must always be used while processing a design and is specific to the target device. Based on the chosen lane width, a subset of the files listed below are created by CORE Generator.

```
<project_dir>/<component_name>/example_design/xilinx_pci_exp_blk_plus
_1_lane_ep-XC5VLX50T-FF1136-1.ucf
<project_dir>/<component_name>/example_design/xilinx_pci_exp_blk_plus
_1_lane_ep-XC5VLX110T-FF1136-1.ucf
<project_dir>/<component_name>/example_design/xilinx_pci_exp_blk_plus
_1_lane_ep-XC5VLX330T-FF1738-1.ucf
<project_dir>/<component_name>/example_design/xilinx_pci_exp_blk_plus
_4_lane_ep-XC5VLX50T-FF1136-1.ucf
<project_dir>/<component_name>/example_design/xilinx_pci_exp_blk_plus
_4_lane_ep-XC5VLX110T-FF1136-1.ucf
<project_dir>/<component_name>/example_design/xilinx_pci_exp_blk_plus
_4_lane_ep-XC5VLX330T-FF1738-1.ucf
<project_dir>/<component_name>/example_design/xilinx_pci_exp_blk_plus
_8_lane_ep-XC5VLX50T-FF1136-1.ucf
<project_dir>/<component_name>/example_design/xilinx_pci_exp_blk_plus
_8_lane_ep-XC5VLX110T-FF1136-1.ucf
<project_dir>/<component_name>/example_design/xilinx_pci_exp_blk_plus
_8_lane_ep-XC5VLX330T-FF1738-1.ucf
```

Wrapper File Usage

The wrapper contains an instance of the PCIe Block Plus core. When starting a new design, modify this wrapper to include all I/O elements and modules. One of the following files will be generated by CORE Generator based on the chosen lane width.

```
<project_dir>/<component_name>/example_design/xilinx_pci_exp_1_lane_ep.v
<project_dir>/<component_name>/example_design/xilinx_pci_exp_4_lane_ep.v
<project_dir>/<component_name>/example_design/xilinx_pci_exp_8_lane_ep.v
```