LatticeECP3 PCS PIPE IP Core User's Guide

July 2010
IPUG83_01.2

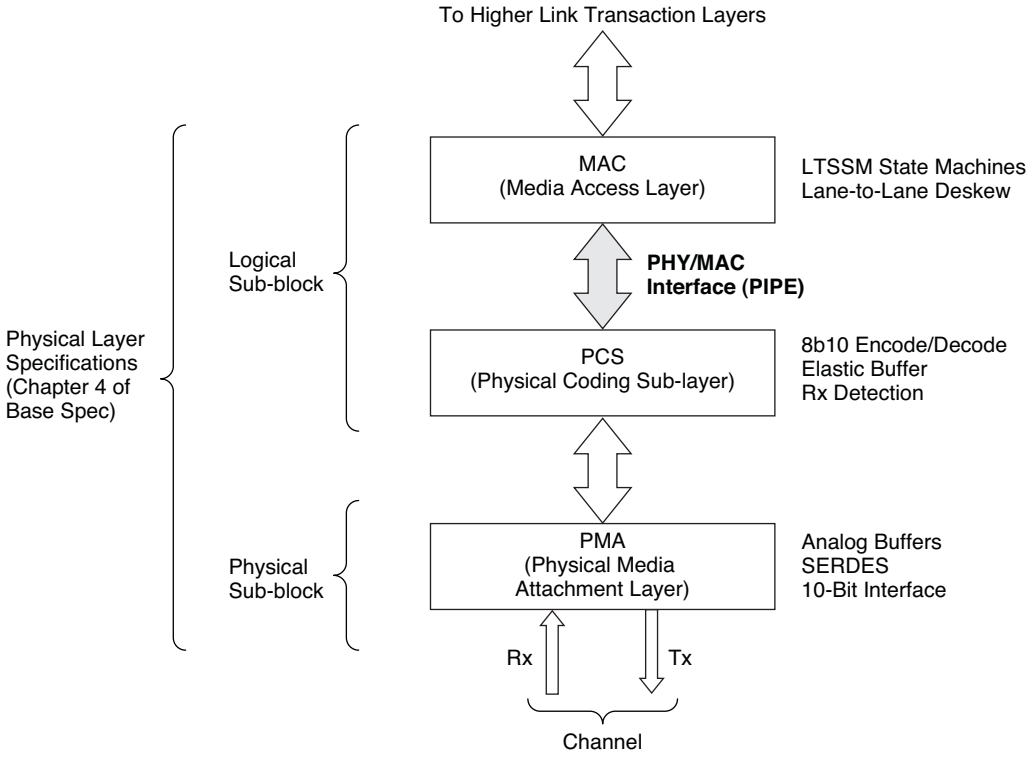
Table of Contents

Intel defined the PHY Interface for PCI Express (PIPE) as a standard interface between a PHY device and the Media Access (MAC) layer for PCI Express (PCIe) applications. The PIPE interface allows the PCI Express PHY device and the MAC layer to be implemented in discrete form (using an off-the-shelf PHY device) or in integrated form. The partitioning of the PCI Express Physical Layer shown Figure 1-1 illustrates this flexibility.

The Lattice PCS PIPE IP core offers PCI Express PHY device functionality, compliant to the Intel PIPE Architecture Draft Version 1.00 (PIPE Ver_1.00), to any endpoint solutions. The PCS PIPE IP core utilizes the SERDES/PCS integrated in LatticeECP3 FPGAs. The Lattice PCS PIPE IP core can be configured to support a link with one or more lanes.

The PCS PIPE IP core is available at no charge and may be directly downloaded from the Lattice IP Server using the IPexpress tool in the Diamond or ispLEVER software.

Figure 1-1. PHY Layer Partitioning



Quick Facts

Table 1-1 gives quick facts about the PCS PIPE IP core.

Table 1-1. LatticeECP3 PCS PIPE IP Core Quick Facts

		LatticeECP3 PCS PIPE IP Core Configuration			
Core Requirements	FPGA Families Supported	LatticeECP3			
	Minimal Device Needed	LFE3-17EA-6TTN256			
Resource Utilization	Targeted Device	LFE3-95E-7FN1156CES			
	Configuration	x1	x4	x1	x4
	Data Path Width	8	8	16	16
	LUTs	125	350	125	500
	Registers	200	400	200	400
	EBRs	0			
Design Tool Support	Lattice Implementation	Diamond® 1.0 or ispLEVER® 8.1			
	Synthesis	Synopsys® Synplify® Pro for Lattice D-2009.12L-1			
		Mentor Graphics® Precision® 2009a 87			
	Simulation	Aldec Active® HDL 8.1 Lattice Edition			
		Mentor Graphics ModelSim® SE 6.5F			

Features

The LatticeECP3 PIPE IP core supports the following features.

PIPE Selection

- Fully compliant to PIPE Rev 1.00 specification
- Standard PCI Express PHY interface allows for multiple IP sources
- Selectable 8-bit or 16-bit interface to transmit and receive PCI Express data
- Holding registers/FIFO for staging transmit and receive data
- Multiple x1 channel support

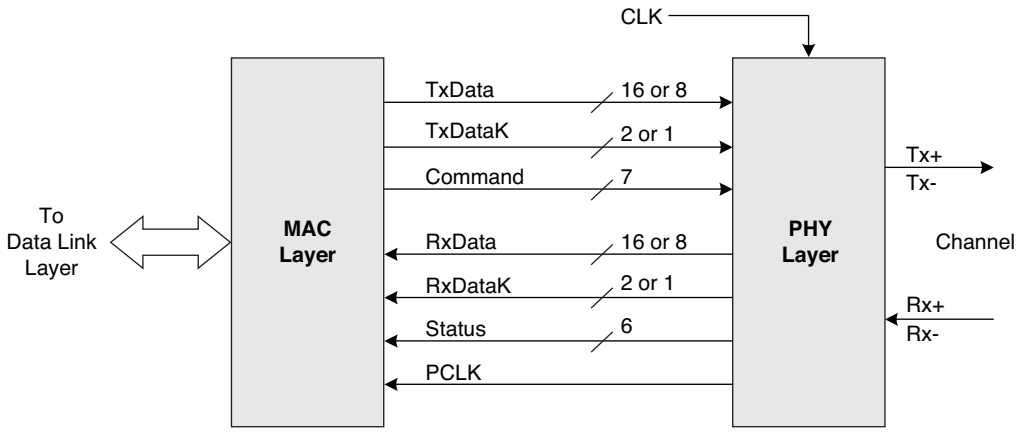
SERDES/PCS Selection

- Selectable SERDES quad location for LatticeECP3 devices
- Selectable x1, multiple x1or x4 PCI Express implementations
- Selectable SERDES channel for PCI Express x1 mode
- Clock/data recovery from the serial stream
- Direct disparity control for use in transmitting compliance pattern
- 8b10b encoder/decoder and error indication
- Receiver detection
- 2.5GT/s full-duplex rate per channel

This chapter provides a functional description of the PCS PIPE IP core.

Figure 2-1 shows the signals between the PHY and the MAC layers as defined by the PIPE Ver.100 document. Table provides a listing and description of the signals.

Figure 2-1. PIPE / MAC Interface



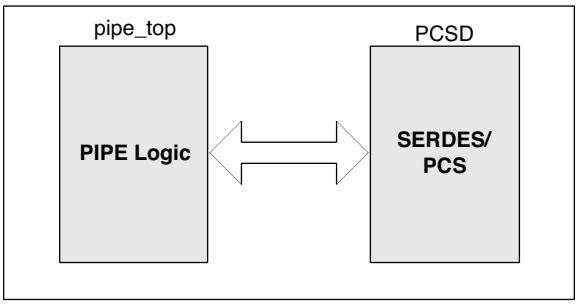
Block Diagram

The PCS PIPE IP core, as an implementation of a PIPE-compliant PHY device, consists of two main functional blocks: the SERDES/PCS quad module and the pipe_top module.

The pipe_top module contains the logic required to transmit parallel data from the MAC, transfer received data to the MAC layer and other functions conforming to the PIPE specifications.

Figure 2-2 shows a high-level block diagram that illustrates the two main functional blocks.

Figure 2-2. PCS PIPE IP Block Diagram



Signal Descriptions

Table 2-1 lists the ports and their descriptions for the PCS PIPE IP core. Note that the signal direction “In/Out” is referenced from the PHY device. Thus, an “Out” signal is driven by the PHY and an “In” signal is input to the PHY.

For more information on the operational timing involving these signals, please refer to the PIPE Ver_1.00 specification.

Table 2-1. PCS PIPE IP Port List

PIPE Signal	Direction	Description
Command Interface Signals		
TxDetectRx/ Loopback	In	Used to determine the operational mode of the PHY. 1 = Request transmitter to do far-end receiver detection 0 = Normal data operation (loopback)
TxElecIdle	In	Forces Tx Output to electrical idle when asserted in all power states. 1 = Force SERDES transmitter to output Electrical Idle 0 = Normal operation When de-asserted while in P0 (operational) (as indicated by PowerDown signals), indicates that there is valid data present on the TxData[..] and TxDataK [..] pins and that the data should be transmitted. When de-asserted in P2 (lowest power state) (as indicated by the PowerDown signals), indicates that the PHY should begin transmitting beacon signals. In this case the signal is asynchronous. TxElecIdle must always be asserted while in power states P0s and P1 (as indicated by the PowerDown signals). This is discussed in more detail later in this section.
TxCompliance	In	Sets the running disparity to negative. Used when transmitting the compliance pattern.
RxPolarity	In	Tells PHY to do a polarity inversion on the received data. 1 = PHY does polarity inversion 0 = PHY does no polarity inversion
PowerDown [1:0]	In	**The SerDes_PCS block is never powered down**. The relative power saving in relation to the rest of the FPGA is minimal. The PowerDown[1:0] signals are used to control some of the other signals going into the block but are not directly connected. Power States (DOES NOT SUPPORT ALL POWER MANAGEMENT STATES) [1] [0] Description 0 0 P0, Normal Operation 0 1 P0s, Power Saving, low recovery latency (Normal Operation) 1 0 P1, Lower Power, Longer recovery latency (64us) (Normal) 1 1 P2, Lowest Power (Power Down)
Status Interface Signals		
RxValid[3:0]	Out	Indicates symbol lock and valid data on RxData and RxDataK. 1 = Lane is synchronized to commas 0 = Lane has not found comma
RxElecIdle[3:0]	Out	Indicates receiver detection of an electrical idle. This is an asynchronous signal. Loss of Signal detection for each channel. Includes a high value and a low value, but only the low value will be sent to the FPGA interface. Register bits are used to set the low value and high value thresholds for these loss-of-signal pins.
PhyStatus[3:0]	Out	Used to indicate completion of several PHY functions including power management state transitions, and receiver detection. When this signal transitions during entry and exit from P2 and PCLK is not running, this signal is asynchronous. This signal needs special treatment in the PIPE logic as discussed later

Table 2-1. PCS PIPE IP Port List (Continued)

PIPE Signal	Direction	Description
RxStatus[2:0]	Out	Encodes receiver status and error codes for the received data stream and receiver detection. [2] [1] [0] Description 0 0 0 Received data OK 0 0 1 1 SKP added 0 1 0 1 SKP removed 0 1 1 Receiver detected 1 0 0 8B/10B decode error 1 0 1 Elastic Buffer Overflow 1 1 0 Elastic Buffer Underflow 1 1 1 Receive disparity error Note: A priority is required. This is discussed later.
Transmit Data Interface Signals		
TxData[15:0] for 16-bit interface TxData[7:0] for 8-bit interface	In	Parallel PCI Express data input bus. For the 16-bit interface, 16 bits represents two symbols of transmit data. Bits [7:0] are the first symbol to be transmitted; bits [15:8] are the second symbol.
TxDataK[1:0] for 16-bit interface TxDataK for 8-bit interface	In	Data/Control for the symbols of transmit data. For 16-bit interfaces, Bit 0 corresponds to the low byte of TxData; Bit 1 to the upper byte. A value of zero indicates a data byte; a value of 1 indicates a control byte.
Receive Data Interface Signals		
RxData[15:0] for 16-bit interface RxData[7:0] for 8-bit interface	Out	Parallel PCI Express data output bus. For the 16-bit interface, 16 bits represents two symbols of receive data. Bits [7:0] are the first symbol to be received, and bits [15:8] are the second symbol.
RxDataK[1:0] for 16-bit interface RxDataK for 8-bit interface	Out	Data/Control for the symbols of receive data. For 16-bit interfaces, Bit 0 corresponds to the low byte of RxData, Bit 1 to the upper-byte. A value of zero indicates a data byte; a value of 1 indicates a control byte.

SCI Bus

The SERDES Client Interface (SCI) allows the SERDES/PCS quad to be controlled by registers as opposed to the configuration memory cells. It is a simple register configuration interface. For further information on this interface, refer to the LatticeECP3 Family Data Sheet.

The PCS/SERDES memory map will be initialized during bitstream configuration using the pcs_pipe_ecp3.txt file provided with the IP core.

Clocking Scheme

The PCI Express IP core requires a 250 MHz reference clock. This reference clock is used to clock the SERDES block of the physical layer as well as the remainder of the PCI Express protocol stack. Figure 2-3 and Figure 2-4 provide a block diagram of the clocking scheme.

Figure 2-3. PCS_PIPE IP Core Clocking Scheme for x4, 8-Bit Configuration

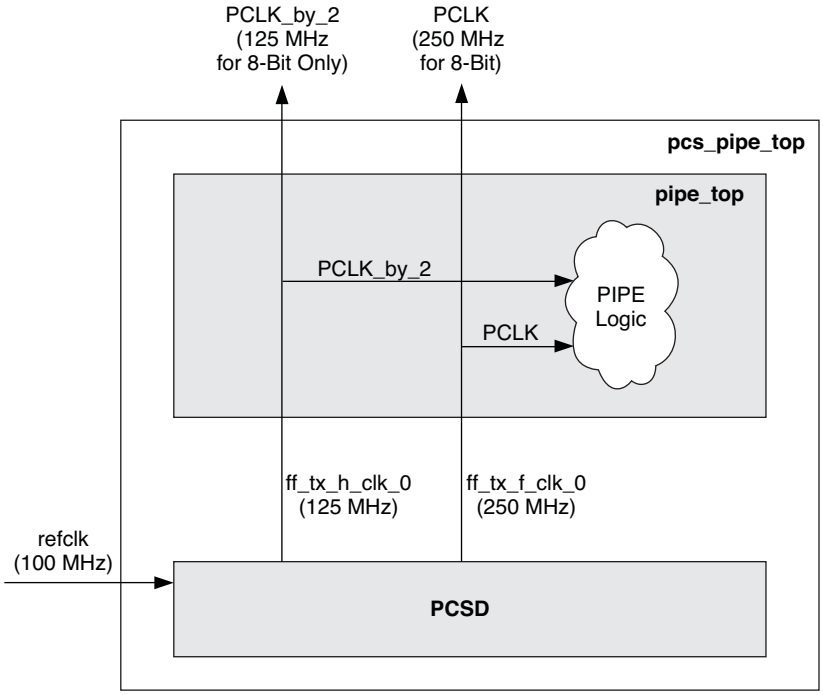
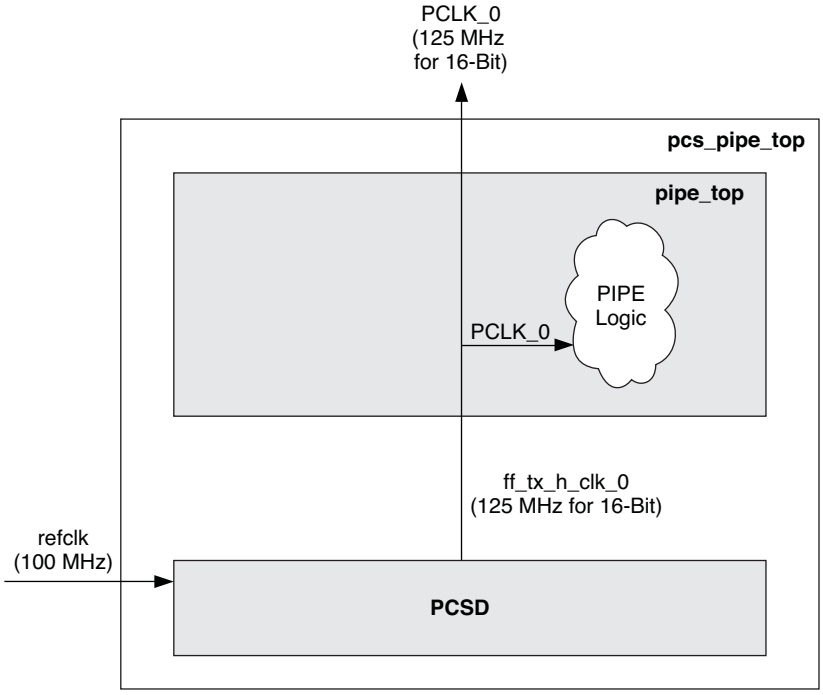


Figure 2-4. PIPE IP Core Clocking Scheme for x1, 16-Bit with Channel_0 Enabled



Typically, PCI Express provides a 100MHz reference clock. This clock must be supplied as a refclk to PCS.

The IPexpress tool is used to create IP and architectural modules in the Diamond or ispLEVER software. Refer to "IP Core Generation" on page 11 for a description on how to generate the IP.

Table 3-1 provides the list of user configurable parameters for the PCS PIPE IP core. The parameter settings are specified using the PCS PIPE IP core Configuration GUI in IPexpress.

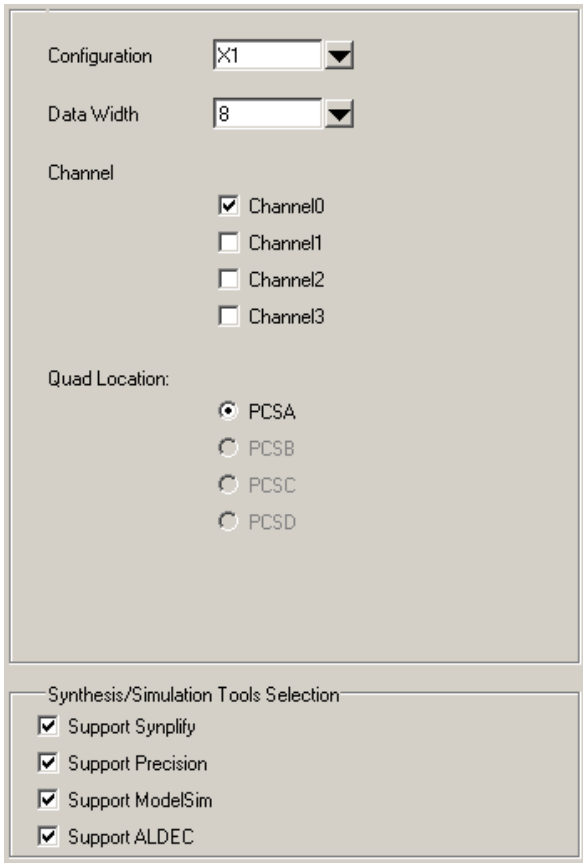
Table 3-1. PCS PIPE IP Core Configuration Parameters

Parameter	Range/Options	Default
Configuration	x1, x4	x1
Data Width	8, 16	8
Channel	Channel0, Channel1, Channel2, Channel3	Channel0
Quad Location	PCSA, PCSB, PCSC, PCSD	PCSA
Synthesis/Simulation Tools Selection		
Support Synplify, Support Precision, Support ModelSim, Support ALDEC	Enable, Disable	Enable

PCS PIPE Configuration Dialog Box

Figure 3-1 shows the PCS PIPE Configuration dialog box.

Figure 3-1. PCS PIPE Configuration Dialog Box



Parameter Descriptions

This section describes the available parameters for the PCS PIPE IP core.

Configuration

Specifies the number of PCI Express lanes that the PIPE interface needs to support. The LatticeECP3 PCS PIPE IP core supports the generation of x1 and x4 configurations.

Datawidth

Specifies the data width of each lane at the PCS_PIPE interface. It supports 8-bit and 16-bit configurations.

Channel

This option is enabled only for x1 mode. If this mode is selected, the user can choose any one, or a combination, of the four SERDES channels in the selected PCS quad.

Quad Location

Specifies the physical location of the SERDES/PCS block. Depending on the device selected, the available SERDES/PCS blocks are shown.

Synthesis/Simulation Tools Selection

The PCS PIPE IP core evaluation capability supports multiple synthesis and simulation tool flows. These options allow the user to select desired tool support.

IP Core Generation

This chapter provides information on how to generate the PCS PIPE IP core using the Diamond or ispLEVER software IPexpress tool, and how to include the core in a top-level design.

This version of the PCS PIPE IP core can be used in LatticeECP3 device families.

Licensing

The Lattice PCS PIPE IP core is available free of charge and does not require a license.

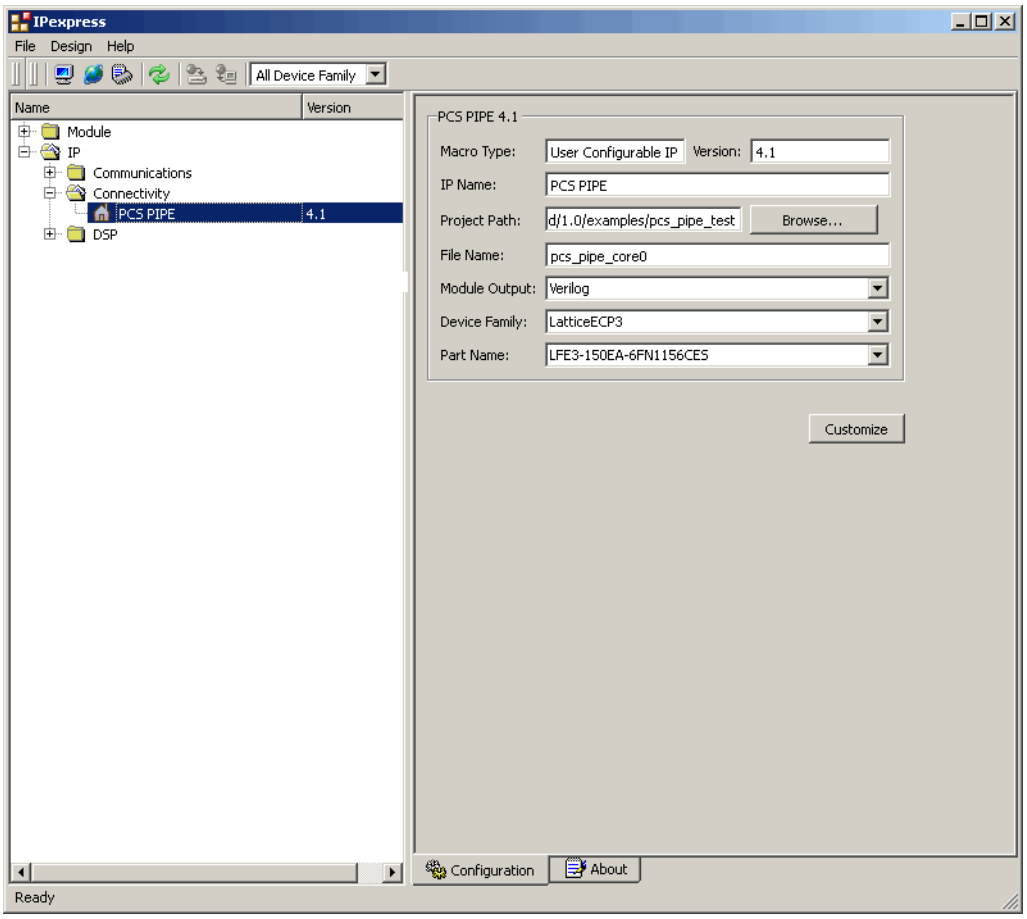
Getting Started

The PCS PIPE IP core is available for download from the Lattice IP Server using the IPexpress tool. The IP files are automatically installed using ispUPDATE technology in any customer-specified directory. After the IP core has been installed, the IP core will be available in the IPexpress GUI dialog box shown in Figure 4-1.

The IPexpress tool GUI dialog box for the PCS PIPE IP core is shown in Figure 4-1. To generate a specific IP core configuration the user specifies:

- **Project Path** – Path to the directory where the generated IP files will be located.
- **File Name** – “username” designation given to the generated IP core and corresponding folders and files.
- **(Diamond) Module Output** – Verilog or VHDL.
- **(ispLEVER) Design Entry Type** – Verilog HDL or VHDL.
- **Device Family** – Device family to which IP is to be targeted (e.g. LatticeECP3). Only families that support the particular IP core are listed.
- **Part Name** – Specific targeted part within the selected device family.

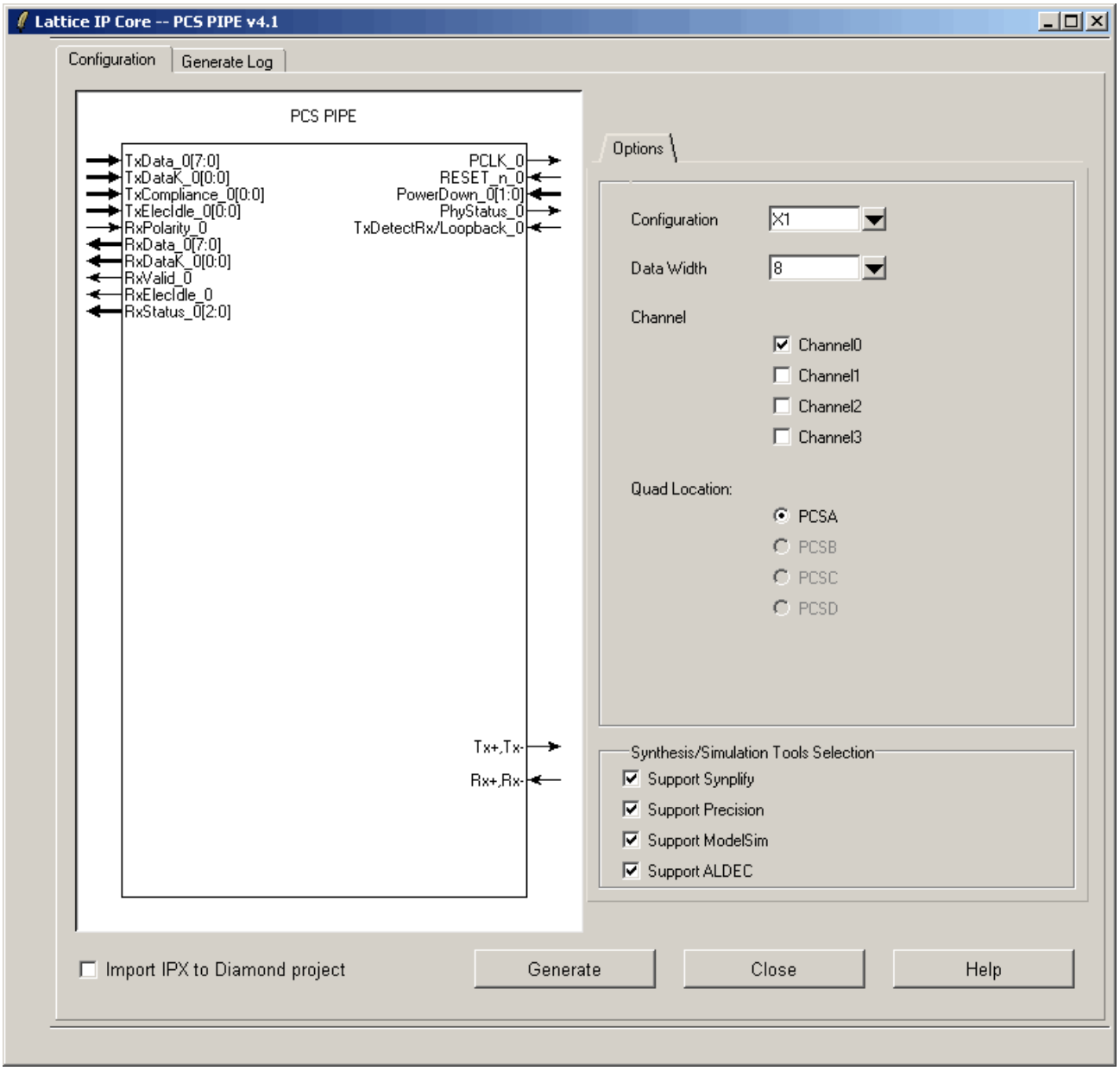
Figure 4-1. IPexpress Tool Dialog Box (Diamond Version)



Note that if the IPexpress tool is called from within an existing project, Project Path, Module Output (Design Entry in ispLEVER), Device Family and Part Name default to the specified project parameters. Refer to the IPexpress tool online help for further information.

To create a custom configuration, the user clicks the **Customize** button in the IPexpress tool dialog box to display the PCS PIPE IP core Configuration GUI, as shown in Figure 4-2. From this dialog box, the user can select the IP parameter options specific to their application. Refer to "Parameter Settings" on page 9 for more information on the PCS PIPE IP parameter settings.

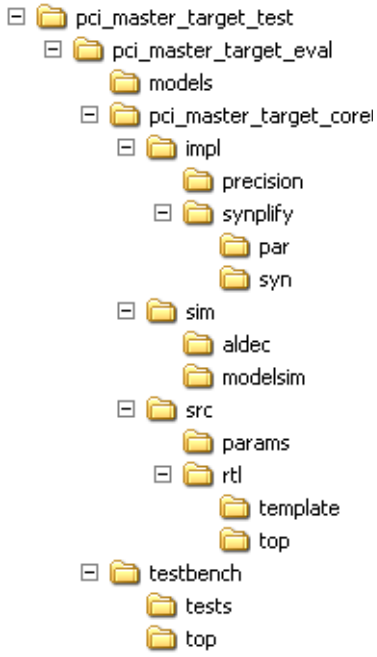
Figure 4-2. PCS PIPE IP Configuration GUI (Diamond Version)



IPexpress-Created Files and Top Level Directory Structure

When the user clicks the **Generate** button in the IP Configuration dialog box, the IP core and supporting files are generated in the specified "Project Path" directory. The directory structure of the generated files is shown in Figure 4-3.

Figure 4-3. LatticeECP3 PCS PIPE IP Core Directory Structure



The design flow for IP created with IPexpress uses a pre-synthesized module (NGO) for synthesis and a model for simulation. The pre-synthesized module is customized and created during IPexpress generation. Table 4-1 provides a list of files created by IPexpress and how they are used.

Table 4-1. File List

File	Description
<*username*>_inst.v	This file provides an instance template for the IP core.
<*username*>.v	This file provides the top-level IP wrapper for simulation.
pcs_pipe_params.v	This file provides the user options of the IP for the simulation model.
<*username*>_bb.v	This file provides the synthesis black box for the user's synthesis.
<*username*>.ngo	This file provides the synthesized IP core.
pcs_pipe_<config>.txt	This file contains the PCS/SERDES memory map initialization. This file is copied in to the simulation directory as well as the Diamond or ispLEVER project directory.
<*username*>.lpf	This file contains Diamond or ispLEVER preferences to control place and route. These preferences should be included in the user's preference file.
<*username*>.lpc	This file contains the IPexpress options used to recreate or modify the core in IPexpress.
<username>.ipx	The IPX file holds references to all of the elements of an IP or Module after it is generated from the IPexpress tool (Diamond version only). The file is used to bring in the appropriate files during the design implementation and analysis. It is also used to re-load parameter settings into the IP/Module generation GUI when an IP/Module is being re-generated.
pcs_pipe_eval	This directory contains a sample design. This design is capable of performing a simulation and running through Diamond or ispLEVER.

Instantiating the Core

The generated PCS PIPE IP core package includes black-box (<*username*>_bb.v) and instance (<*username*>_inst.v) templates that can be used to instantiate the core in a top-level design.

An example RTL top-level reference source file that can be used as an instantiation template for the IP core is provided in \<*project_dir*>\pcs_pipe_eval\<*username*>\src\rtl\top\<family>. Users may also use this top-level reference as the starting template for the top-level for their complete design.

For VHDL entry, only the top-level file is generated which already contains the core instantiation within.

Implementing the Core in a Top-Level Design

As described previously, the top-level file pcs_pipe_eval.v(vhd) provided in \<*project_dir*>\pcs_pipe_eval\<*username*>\src\rtl\top\<*family*> supports the ability to implement just the PCS PIPE Block.

Push-button top-level implementation of this top-level is supported via the project file <*username*>_eval.ldf (Diamond) or .syn (ispLEVER) located in \<*project_dir*>\ pcs_pipe_eval\<*username*>\impl\(synplify or precision).

This design is intended only to provide an accurate indication of the device utilization associated with the core itself and should not be used as an actual implementation example.

To use this project file in Diamond:

1. Choose **File > Open > Project**.
2. Browse to \<*project_dir*>\pcs_pipe_eval\<*username*>\impl\<synplify|precision> in the Open Project dialog box.
3. Select and open <*username*>.ldf. At this point, all of the files needed to support top-level synthesis and implementation will be imported to the project.
4. Select the **Process** tab in the left-hand GUI window.
5. Implement the complete design via the standard Diamond GUI flow.

To use this project file in ispLEVER:

1. Choose **File > Open Project.**
2. Browse to \<*project_dir*>\pcs_pipe_eval\<*username*>\impl\<synplify|precision> in the Open Project dialog box.
3. Select and open <username>.syn. At this point, all of the files needed to support top-level synthesis and implementation will be imported to the project.
4. Select the device top-level entry in the left-hand GUI window.
5. Implement the complete design via the standard ispLEVER GUI flow.

Linux/Solaris

For the Linux/Solaris platform, the top-level synthesis needs to be run separately with a stand-alone synthesis tool, like Synplify Pro, since the Diamond or ispLEVER Linux version only accepts EDIF entry. A synthesis tcl file called pcs_pipe_eval.tcl will be generated in the directory \<*project_dir*>\pcs_pipe_eval\<*username*>\impl \<synplify|precision>.

Users can use the following command to synthesize the top-level file:

```
synpwrap -options -batch pcs_pipe_eval.tcl.
```

Special Instructions for Using ModelSim Libraries

For this IP core, the following zip file will need to be unzipped in order to compile with ModelSim SE:

- pcsd-mti_6.0-V1-1.zip (for LatticeECP3)

Unzip the file under the <*diamond_dir or isplever_dir*>\cae_library\simulation\blackbox, so you will see the subdirectory pcsc(d)_work.

In some cases, the library is obsolete and you will need to refresh it in the ModelSim GUI interface:

1. Type
   `vmap pcsd_work [DIAMOND_INSTALL_PATH or ISPLEVER_INSTALL_PATH]/cae_library/simulation/blackbox/pcsd_work` in the Transcript window.
2. In the Library Manager panel, right-click pcsd_work and select **Refresh**.
3. The PCS pre-compile library is now ready.

Running Functional Simulation

The functional simulation includes a configuration-specific behavioral model of the PCS PIPE, which is instantiated in an FPGA top level along with some test logic (PLLs and registers with the read/write interface). This FPGA top is instantiated in an evaluation testbench that configures FPGA test logic registers and PCS PIPE IP core registers.

The generated IP core package includes the configuration-specific behavior source files (in .\pcs_pipe_eval\< username>\srt\rtl\soft\) for functional simulation. ModelSim simulation is supported via testbench files provided in \<*project_dir*>\pcs_pipe_eval\<*username*>\testbench\top. Models required for simulation are provided in the corresponding \models\<Family> folder.

Users may run the evaluation simulation by doing the following with ModelSim (Windows or Linux):

1. Open ModelSim.
2. Under the File tab, select **Change Directory**.
3. Set the directory to \<*project_dir*>\pcs_pipe_eval\<*username*>\sim\modelsim\rtl.
4. Select **OK**.
5. Under the Tools tab, select **TCL**, then select **Execute Macro in ..\scripts**.
6. Select the file eval_beh_rtl_se.do for ModelSim SE version

Users may run the evaluation simulation by doing the following with Active-HDL (Windows only):

1. Open Active-HDL.
2. Under the **Tools** tab, select **Execute Macro**.

3. Select the file \<*project_dir*>\pcs_pipe_eval\<*username*>\sim\aldec\scripts\eval_beh_rtl.do.

4. Select **OK**.

Note: Simulation will only be available for the default configuration. VHDL timing simulation is not supported in the evaluation package.

Updating/Regenerating the IP Core

By regenerating an IP core with the IPexpress tool, you can modify any of its settings including: device type, design entry method, and any of the options specific to the IP core. You can also update older IP cores to the latest version. Regenerating can be done to modify an existing IP core or to create a new but similar one.

Regenerating an IP Core in Diamond

To regenerate an IP core in Diamond:

1. In IPexpress, click the **Regenerate** button.

2. In the Regenerate view of IPexpress, choose the IPX source file of the module or IP you wish to regenerate.

3. IPexpress shows the current settings for the module or IP in the Source box. Make your new settings in the T**arget** box.

4. If you want to generate a new set of files in a new location, set the new location in the **IPX Target File** box. The base of the file name will be the base of all the new file names. The IPX Target File must end with an .ipx extension.

5. Click **Regenerate.** The module's dialog box opens showing the current option settings.

6. In the dialog box, choose the desired options. To get information about the options, click **Help**. Also, check the About tab in IPexpress for links to technical notes and user guides. IP may come with additional information. As the options change, the schematic diagram of the module changes to show the I/O and the device resources the module will need.

7. To import the module into your project, if it's not already there, select **Import IPX to Diamond Project** (not available in stand-alone mode).

8. Click **Generate**.

9. Check the Generate Log tab to check for warnings and error messages.

10. Click **Close**.

The IPexpress package file (.ipx) supported by Diamond holds references to all of the elements of the generated IP core required to support simulation, synthesis and implementation. The IP core may be included in a user's design by importing the .ipx file to the associated Diamond project. To change the option settings of a module or IP that is already in a design project, double-click the module's .ipx file in the File List view. This opens IPexpress and the module's dialog box showing the current option settings. Then go to step 6 above.

Regenerating an IP Core in ispLEVER

To regenerate an IP core in ispLEVER:

1. In the IPexpress tool, choose **Tools > Regenerate IP/Module**.

2. In the Select a Parameter File dialog box, choose the Lattice Parameter Configuration (.lpc) file of the IP core you wish to regenerate, and click **Open**.

3. The Select Target Core Version, Design Entry, and Device dialog box shows the current settings for the IP core in the Source Value box. Make your new settings in the Target Value box.

4. If you want to generate a new set of files in a new location, set the location in the LPC Target File box. The base of the .lpc file name will be the base of all the new file names. The LPC Target File must end with an .lpc extension.

5. Click **Next**. The IP core's dialog box opens showing the current option settings.

6. In the dialog box, choose desired options. To get information about the options, click **Help**. Also, check the About tab in the IPexpress tool for links to technical notes and user guides. The IP core might come with additional information. As the options change, the schematic diagram of the IP core changes to show the I/O and the device resources the IP core will need.

7. Click **Generate**.

8. Click the **Generate Log** tab to check for warnings and error messages.

Core Verification

The functionality of the Lattice PCI PIPE IP cores have been verified via simulation and hardware testing in a variety of environments, including:

- Simulation environment verifying proper PCI PIPE IP functionality using Lattice's proprietary verification environment
- Hardware validation of the IP implemented on Lattice FPGA evaluation boards. Specific testing includes the following categories of tests:
  - Linkup Test
  - Configuration Test
  - Protocol Test
  - Inter-operability Test

Support Resources

TThis chapter contains information about Lattice Technical Support, additional references, and document revision history.

Lattice Technical Support

There are a number of ways to receive technical support.

Online Forums

The first place to look is Lattice Forums (http://www.latticesemi.com/support/forums.cfm). Lattice Forums contain a wealth of knowledge and are actively monitored by Lattice Applications Engineers.

Telephone Support Hotline

Receive direct technical support for all Lattice products by calling Lattice Applications from 5:30 a.m. to 6 p.m. Pacific Time.

- For USA & Canada: 1-800-LATTICE (528-8423)
- For other locations: +1 503 268 8001

In Asia, call Lattice Applications from 8:30 a.m. to 5:30 p.m. Beijing Time (CST), +0800 UTC. Chinese and English language only.

- For Asia: +86 21 52989090

E-mail Support

- techsupport@latticesemi.com
- techsupport-asia@latticesemi.com

Local Support

Contact your nearest Lattice Sales Office.

Internet

www.latticesemi.com

PCIe Solutions Web Site

Lattice provides customers with low-cost and low-power programmable PCIe solutions that are ready to use right out of the box. A full suite of tested and interoperable PCIe solutions is available that includes development kits with evaluation boards and hardware and software reference designs. These solutions are valuable resources to jump start PCIe applications from a board design and FPGA design perspective. For more information on Lattice's PCIe solutions, visit:

http://www.latticesemi.com/solutions/technologysolutions/pciexpresssolutions.cfm?source=topnav

PCI-SIG Website

The Peripheral Component Interconnect Special Interest Group (PCI-SIG) website contains specifications and documents referred to in this user's guide. The PCi-SIG URL is:

http://www.pcisig.com.

References

LatticeECP3

- HB1009, *LatticeECP3 Family Handbook*

Revision History

Date	Document Version	IP Core Version	Change Summary
August 2009	01.0	4.0	Initial release.
May 2010	01.1	4.0	Divided document into chapters.
			Added Table 1-1 on page 4.
			Added "Core Verification" on page 19.
July 2010	01.2	4.1	Added support for Diamond sotware throughout.

Appendix A:

Resource Utilization

This appendix gives resource utilization information for Lattice FPGAs using the Lattice PCS PIPE IP core.

The IP configurations shown in this chapter were generated using the IPexpress software tool. IPexpress is the Lattice IP configuration utility, and is included as a standard feature of the Diamond and ispLEVER design tools. Details regarding the usage of IPexpress can be found in the IPexpress and Diamond or ispLEVER help system. For more information on the Diamond or ispLEVER design tools, visit the Lattice web site at: www.latticesemi.com/software.

LatticeECP3 FPGAs

Table A-1. Performance and Resource Utilization [1]

Configuration	Data Width	Active Channels	Quad	Slices	LUTs	Registers	EBR	f_{MAX} (MHz) PCLK
x1	8	0	PCSB	117	105	172	0	263
x1	16	0	PCSB	132	125	192	0	177
x4	8	0-3	PCSB	312	315	475	0	256
x4	16	0-3	PCSB	395	491	558	0	155

1. Performance and utilization data are targeting an LFE-95E-7FN672CES device using Diamond 1.0 and Synplify Pro D-2009.12L-1 software. Performance may vary when using a different software version or targeting a different device density or speed grade within the LatticeECP3 family.