Intel® Stratix® 10 Avalon® Streaming (Avalon-ST) IP for PCIe* Design Example User Guide

Updated for Intel® Quartus® Prime Design Suite: **17.1**

Subscribe
Send Feedback

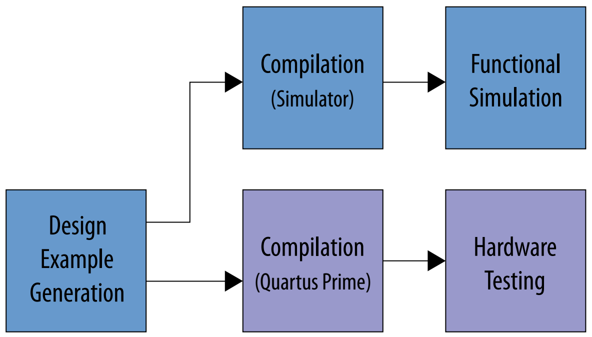
UG-20053 | 2019.12.26
Latest document on the web: PDF | HTML

Contents

1. Quick Start Guide

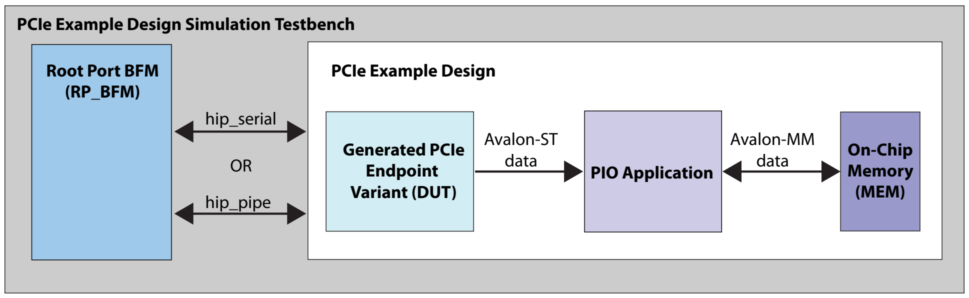
Using Intel® Quartus® Prime software, you can generate a programmed I/O (PIO) design example for the Avalon®-ST Intel Stratix® 10-GX Hard IP for PCI Express* IP core. The generated design example reflects the parameters that you specify. The PIO example transfers data from a host processor to a target device. It is appropriate for low-bandwidth applications. This design example automatically creates the files necessary to simulate and compile in the Intel Quartus Prime software. You can download the compiled design to the Intel Stratix 10-GX FPGA Development Board. To download to custom hardware, update the Intel Quartus Prime Settings File (`.qsf`) with the correct pin assignments .

Figure 1. Development Steps for the Design Example



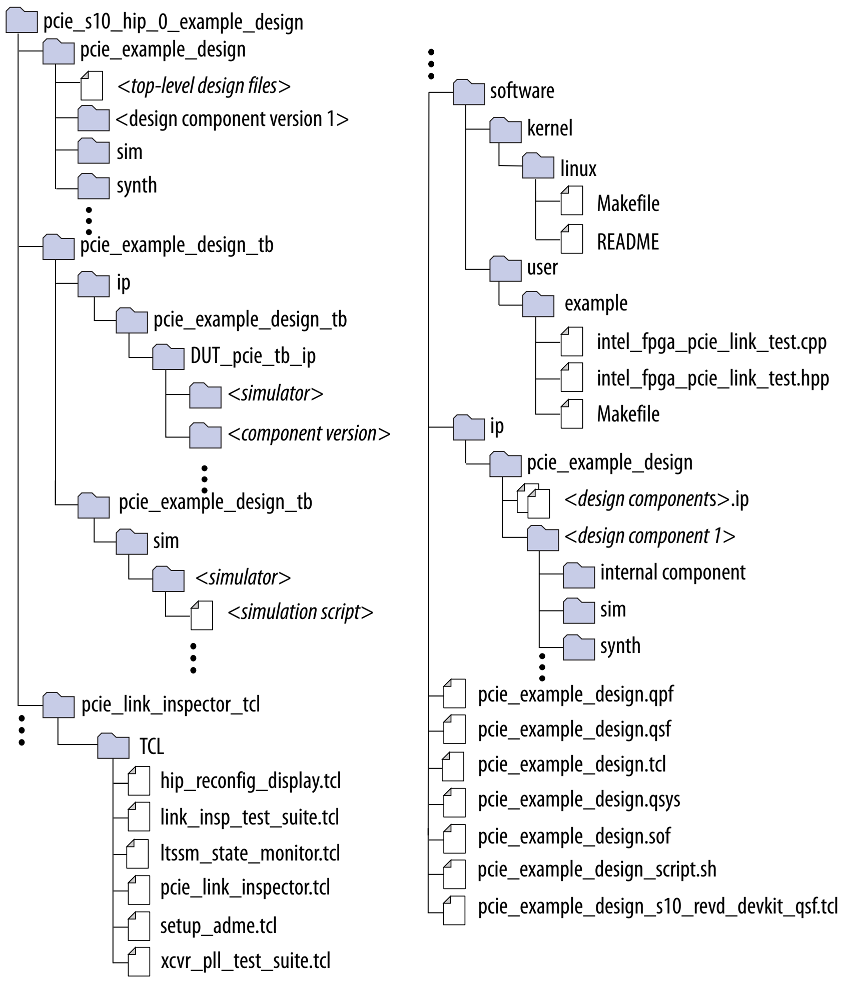
1.1. Design Components

Figure 2. Block Diagram for the Platform Designer PIO Design Example Simulation Testbench



1.2. Directory Structure

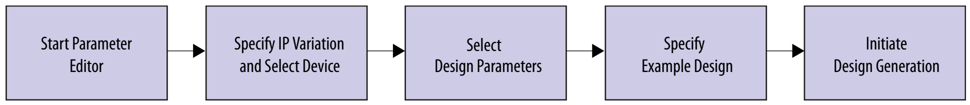
Figure 3. Directory Structure for the Generated Design Example



1.3. Generating the Design Example

Follow these steps to generate your design:

Figure 4. Procedure



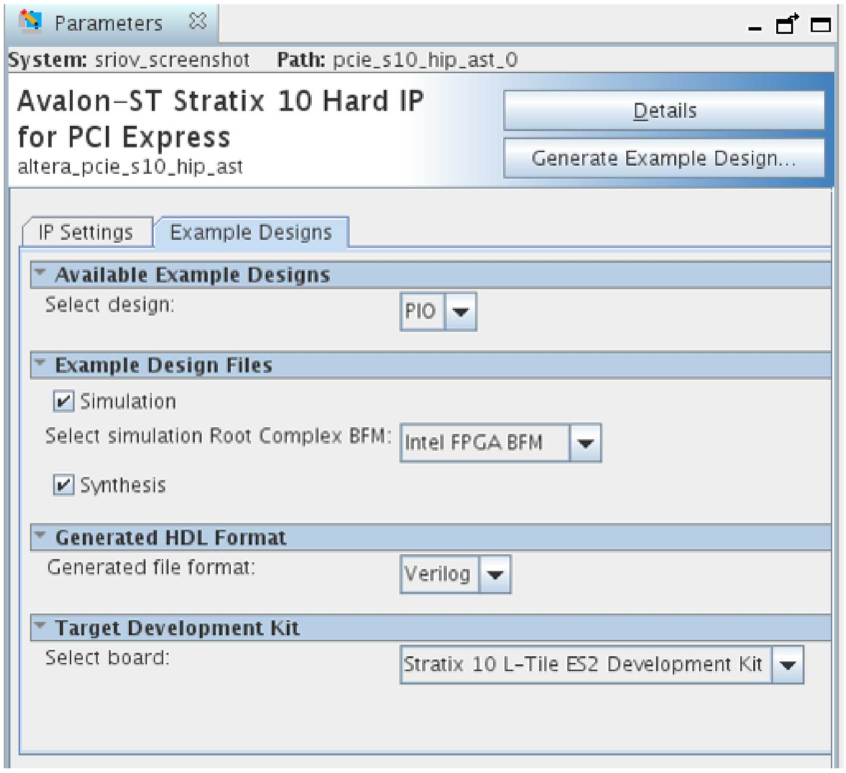
1. In the Intel Quartus Prime Pro Edition software, create a new project (**File ➤ New Project Wizard**).
2. Specify the **Directory, Name**, and **Top-Level Entity.**
3. For **Project Type**, accept the default value, **Empty project**. Click **Next**.
4. For **Add Files** click **Next**.
5. For **Family, Device & Board Settings** under **Family**, select **Intel Stratix 10 (GX/SX/MX/TX)** and the **Target Device** for your design.
6. Click **Finish**.
7. In the IP Catalog locate and add the **Avalon-ST Intel Stratix 10 Hard IP for PCI Express**.
8. In the **New IP Variant** dialog box, specify a name for your IP. Click **Create**.
9. On the **IP Settings** tabs, specify the parameters for your IP variation.
10. On the **Example Designs** tab, make the following selections:
    a. For **Available Example Designs,** select **PIO**.
    b. For **Example Design Files**, turn on the **Simulation** and **Synthesis** options. If you do not need these simulation or synthesis files, leaving the corresponding option(s) turned off significantly reduces the example design generation time.
    c. If you have selected a x16 configuration, for **Select simulation Root Complex BFM**, choose the appropriate BFM:
        - **Intel FPGA BFM**: for all configurations up to Gen3 x8. This bus functional model (BFM) supports x16 configurations by downtraining to x8.
        - **Third-party BFM**: for x16 configurations if you want to simulate all 16 lanes using a third-party BFM. Refer to AN-811: Using the Avery BFM for PCI Express Gen3x16 Simulation on Intel Stratix 10 Devices for information about simulating with the Avery BFM.

    d. For **Generated HDL Format**, only Verilog is available in the current release.
    e. For **Target Development Kit**, select the appropriate option.

    Note: If you select **None**, the generated design example targets the device you specified in Step 5 above. If you intend to test the design in hardware, make the appropriate pin assignments in the `.qsf` file. You can also use the pin planner tool to make pin assignments.
11. Select **Generate Example Design** to create a design example that you can simulate and download to hardware. If you select one of the Intel Stratix 10 development boards, the device on that board overwrites the device previously selected in the Intel Quartus Prime project if the devices are different. When the prompt asks you to specify the directory for your example design, you can accept

the default directory, *<example_design>/*
`pcie_s10_hip_avmm_bridge_0_example_design`, or choose another directory.

Figure 5. Example Design Tab


Parameters
System: sriov_screenshot Path: pcie_s10_hip_ast_0
Avalon-ST Stratix 10 Hard IP for PCI Express
altera_pcie_s10_hip_ast
Details
Generate Example Design...
IP Settings | Example Designs
Available Example Designs
Select design: PIO
Example Design Files
[✓] Simulation
Select simulation Root Complex BFM: Intel FPGA BFM
[✓] Synthesis
Generated HDL Format
Generated file format: Verilog
Target Development Kit
Select board: Stratix 10 L-Tile ES2 Development Kit

When you generate an Intel Stratix 10 example design, a file called `recommended_pinassignments_s10.txt` is created in the directory pcie_s10_hip_avmm_bridge_0_example_design.[1]

12. Click **Finish**. You may save your `.ip` file when prompted, but it is not required to be able to use the example design.
13. The prompt, **Recent changes have not been generated. Generate now?**, allows you to create files for simulation and synthesis of the IP core variation that you specified in Step 9 above. Click **No** if you only want to work with the design example you have generated.
14. Close the dummy project.

[1] This file contains the recommended pin assignments for all the pins in the example design. If you select a development kit option in the pull-down menu for **Target Development Kit**, the pin assignments in the `recommended_pinassignments_s10.txt` file match those that are in the `.qsf` file in the same directory. If you chose **NONE** in the pull-down menu, the `.qsf` file does not contain any pin assignment. In this case, you can copy the pin assignments in the `recommended_pinassignments_s10.txt` file to the `.qsf` file. You can always change any pin assignment in the `.qsf` file to satisfy your design or board requirements.

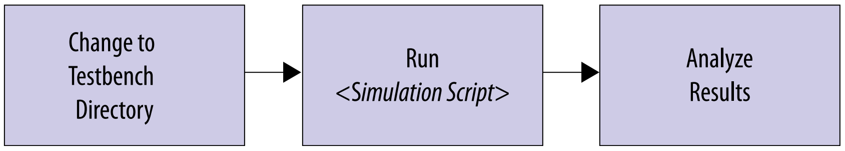
15. Open the example design project.
16. Compile the example design project to generate the `.sof` file for the complete example design. This file is what you download to a board to perform hardware verification.
17. Close your example design project.

Related Information

AN-811: Using the Avery BFM for PCI Express Gen3x16 Simulation on Intel Stratix 10 Devices

1.4. Simulating the Design Example

Figure 6. Procedure



1. Change to the testbench simulation directory, `pcie_example_design_tb`.
2. Run the simulation script for the simulator of your choice. Refer to the table below.
3. Analyze the results.

Table 1. Steps to Run Simulation

Simulator	Working Directory	Instructions
ModelSim*	`<example_design>/pcie_example_design_tb/pcie_example_design_tb/sim/mentor/`	1. Invoke vsim (by typing `vsim`, which brings up a console window where you can run the following commands). 2. `do msim_setup.tcl`

continued...

Simulator	Working Directory	Instructions
		Note: Alternatively, instead of doing Steps 1 and 2, you can type: `vsim -c -do msim_setup.tcl`. 3. `ld_debug` 4. `run -all` 5. A successful simulation ends with the following message, "Simulation stopped due to successful completion!"
VCS*	`<example_design>/pcie_example_design_tb/pcie_example_design_tb/sim/synopsys/vcs`	1. `sh vcs_setup.sh USER_DEFINED_COMPILE_OPTIONS="" USER_DEFINED_ELAB_OPTIONS="-xlrm\ uniq_prior_final" USER_DEFINED_SIM_OPTIONS=""` 2. A successful simulation ends with the following message, "Simulation stopped due to successful completion!"
NCSim*	`<example_design>/pcie_example_design_tb/pcie_example_design_tb/sim/cadence`	1. `sh ncsim_setup.sh USER_DEFINED_SIM_OPTIONS="" USER_DEFINED_ELAB_OPTIONS="-timescale\ 1ns/1ps"` 2. A successful simulation ends with the following message, "Simulation stopped due to successful completion!"
Xcelium* Parallel Simulator	`<example_design>/pcie_example_design_tb/pcie_example_design_tb/sim/xcelium`	1. `sh xcelium_setup.sh USER_DEFINED_SIM_OPTIONS="" USER_DEFINED_ELAB_OPTIONS ="-timescale\ 1ns/1ps\ -NOWARN\ CSINFI"` 2. A successful simulation ends with the following message, "Simulation stopped due to successful completion!"

This testbench simulates up to x8 variants. It supports x16 variants by down-training to x8. To simulate all lanes of a x16 variant, you can create a simulation model using the Platform Designer to use in an Avery testbench. For more information refer to *AN-811: Using the Avery BFM for PCI Express Gen3x16 Simulation on Intel Stratix 10 Devices*.

The simulation reports, "Simulation stopped due to successful completion" if no errors occur.

Related Information

AN-811: Using the Avery BFM for PCI Express Gen3x16 Simulation on Intel Stratix 10 Devices

1.5. Compiling the Design Example and Programming the Device

1. Navigate to `<project_dir>/pcie_s10_hip_avmm_bridge_0_example_design/` and open `pcie_example_design.qpf`.
2. On the Processing menu, select **Start Compilation**.
3. After successfully compiling your design, program the targeted device with the Programmer.

1.6. Installing the Linux Kernel Driver

Before you can test the design example in hardware, you must install the Linux kernel driver. You can use this driver to perform the following tests:

- A PCIe* link test that performs 100 writes and reads
- Memory space DWORD(2) reads and writes
- Configuration Space DWORD reads and writes

In addition, you can use the driver to change the value of the following parameters:

- The BAR being used
- The selects device by specifying the bus, device and function (BDF) numbers for the required device

The driver also allows you to enable SR-IOV for H-Tile devices.

Complete the following steps to install the kernel driver:

1. Navigate to `./software/kernel/linux` under the example design generation directory.
2. Change the permissions on the `install,` `load`, and `unload` files:
   ```
   $ chmod 777 install load unload
   ```
3. Install the driver:
   ```
   $ sudo ./install
   ```
4. Verify the driver installation:
   ```
   $ lsmod | grep intel_fpga_pcie_drv
   ```
   Expected result:
   ```
   intel_fpga_pcie_drv 17792 0
   ```
5. Verify that Linux recognizes the PCIe design example:
   ```
   $ lspci -d 1172:000 -v | grep intel_fpga_pcie_drv
   ```
   Note: If you have changed the Vendor ID, substitute the new Vendor ID for Intel's Vendor ID in this command.

   Expected result:
   ```
   Kernel driver in use: intel_fpga_pcie_drv
   ```

1.7. Running the Design Example Application

1. Navigate to `./software/user/example` under the design example directory.
2. Compile the design example application:
   ```
   $ make
   ```
3. Run the test:

(2) Throughout this user guide, the terms word, DWORD and QWORD have the same meaning that they have in the PCI Express Base Specification. A word is 16 bits, a DWORD is 32 bits, and a QWORD is 64 bits.

```
$ sudo ./intel_fpga_pcie_link_test
```

You can run the Intel FPGA IP PCIe link test in manual or automatic mode.

— In automatic mode, the application automatically selects the device. The test selects the Intel Stratix 10 PCIe device with the lowest BDF by matching the Vendor ID. The test also selects the lowest available BAR.

— In manual mode, the test queries you for the bus, device, and function number and BAR.

For the Intel Stratix 10 GX Development Kit, you can determine the BDF by typing the following command:

```
$ lspci -d 1172
```

4. Here are sample transcripts for automatic and manual modes:

```
Intel FPGA PCIe Link Test - Automatic Mode
Version 2.0
0: Automatically select a device
1: Manually select a device
***************************************************
>0
Opened a handle to BAR 0 of a device with BDF 0x100
***************************************************
0: Link test - 100 writes and reads
1: Write memory space
2: Read memory space
3: Write configuration space
4: Read configuration space
5: Change BAR
6: Change device
7: Enable SR-IOV
8: Do a link test for every enabled virtual function
   belonging to the current device
9: Perform DMA
10: Quit program
***************************************************
> 0
Doing 100 writes and 100 reads . .
Number of write errors:     0
Number of read errors:      0
Number of DWORD mismatches: 0
```

```
Intel FPGA PCIe Link Test - Manual Mode
Version 1.0
0: Automatically select a device
1: Manually select a device
***************************************************
> 1
Enter bus number:
> 1
Enter device number:
> 0
Enter function number:
> 0
BDF is 0x100
Enter BAR number (-1 for none):
> 4
Opened a handle to BAR 4 of a device with BDF 0x100
```

Related Information

PCIe Link Inspector Overview
Use the PCIe Link Inspector to monitor the link at the Physical, Data Link and Transaction Layers.

2. Design Example Description

2.1. Functional Description for PIO Design Example

The testbench illustrates PIO traffic between the host and Endpoint. The PIO design example consists of memory transfers from a host processor to a target device. In this example, the host processor issues single-dword MemRd and MemWr TLPs.

The Endpoint (DUT) and PIO application (APPS) perform the necessary translation between the PCI Express TLPs and simple Avalon-MM reads and writes to memory.

The PIO testbench includes the following components:

- The Root Port BFM that drives downstream TLPs to the Endpoint.

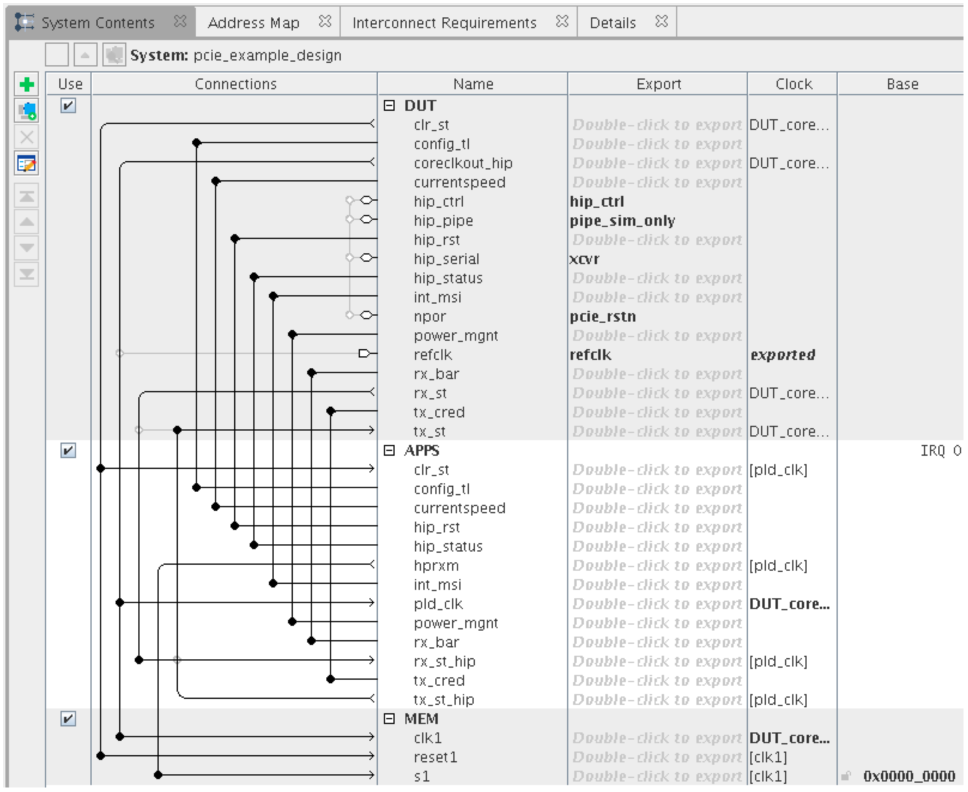
  Note: This Intel Root Port BFM provides a simple method to do basic testing of the Application Layer logic that interfaces to the DUT. However, the testbench and Root Port BFM are not intended to be a substitute for a full verification environment. To thoroughly test your application, obtain commercially available PCI Express verification IP and tools, or do your own extensive hardware testing or both.

- The Generated PCIe Endpoint Variant (DUT) with the parameters you specified. This component drives TLP data received to the PIO application. For more details on this component, refer to the *Intel Stratix 10 Avalon Streaming (Avalon-ST) and Single Root I/O Virtualization (SR-IOV) Interface for PCI Express Solutions User Guide*.
- The PIO Application (APPS) component. Along with some additional logic, it translates Avalon-ST data to Avalon-MM data for writes and reads to the on-chip memory.
- For variants up to Gen3 x8, an on-chip memory (MEM) stores data. Gen3 x16 variants include the memory in the APPs component.

The test program writes and reads back data to 0x00000040 in the on-chip memory. It compares data read to the expected result. The test reports, "Simulation stopped due to successful completion" if no errors occur.

Figure 7. Platform Designer System Contents for Stratix 10 PCI Express PIO Design Example

The Platform Designer generates this design for up to Gen3 x16 variants.



System: pcie_example_design

Use	Name	Export	Clock	Base
✓	DUT			
	clr_st	Double-click to export	DUT_core...	
	config_tl	Double-click to export		
	coreclkout_hip	Double-click to export	DUT_core...	
	currentspeed	Double-click to export		
	hip_ctrl	hip_ctrl		
	hip_pipe	pipe_sim_only		
	hip_rst	Double-click to export		
	hip_serial	xcvr		
	hip_status	Double-click to export		
	int_msi	Double-click to export		
	npor	pcie_rstn		
	power_mgnt	Double-click to export		
	refclk	refclk	exported	
	rx_bar	Double-click to export		
	rx_st	Double-click to export	DUT_core...	
	tx_cred	Double-click to export		
	tx_st	Double-click to export	DUT_core...	
✓	APPS			IRQ 0
	clr_st	Double-click to export	[pld_clk]	
	config_tl	Double-click to export		
	currentspeed	Double-click to export		
	hip_rst	Double-click to export		
	hip_status	Double-click to export		
	hprxm	Double-click to export	[pld_clk]	
	int_msi	Double-click to export		
	pld_clk	Double-click to export	DUT_core...	
	power_mgnt	Double-click to export		
	rx_bar	Double-click to export		
	rx_st_hip	Double-click to export	[pld_clk]	
	tx_cred	Double-click to export		
	tx_st_hip	Double-click to export	[pld_clk]	
✓	MEM			
	clk1	Double-click to export	DUT_core...	
	reset1	Double-click to export	[clk1]	
	s1	Double-click to export	[clk1]	0x0000_0000

Related Information

Intel Stratix 10 Avalon Streaming (Avalon-ST) and Single Root I/O Virtualization (SR-IOV) Interface for PCI Express Solutions User Guide

2.2. Serial Data Signals

This differential, serial interface is the physical link between a Root Port and an Endpoint.

The PCIe IP Core supports 1, 2, 4, 8, or 16 lanes. Each lane includes a TX and RX differential pair. Data is striped across all available lanes.

Table 2. 1-Bit Interface Signals

In the following table <*n*> is the number of lanes.

Signal	Direction	Description
`tx_out[<n>-1:0]`	Output	Transmit output. These signals are the serial outputs of lanes <*n*>-*1*–0.
`rx_in[<n>-1:0]`	Input	Receive input. These signals are the serial inputs of lanes <*n*>-*1*–0.

Refer to *Pin-out Files for Intel Devices* for pin-out tables for all Intel devices in **.pdf**, **.txt**, and **.xls** formats.

Transceiver channels are arranged in groups of six. For GX devices, the lowest six channels on the left side of the device are labeled GXB_L0, the next group is GXB_L1, and so on. Channels on the right side of the device are labeled GXB_R0, GXB_R1, and so on. Be sure to connect the Hard IP for PCI Express on the left side of the device to appropriate channels on the left side of the device, as specified in the *Pin-out Files for Intel Devices*.

Related Information

- Pin-out Files for Intel Devices
- Link Inspector Hardware
  Use the PCIe Link Inspector to monitor the link at the Physical, Data Link and Transaction Layers.

2.3. Registers

There are no control registers for the PIO design example. The *PCI Express Base Specification 3.0* defines a comprehensive set of configuration, control, and status registers to control and debug the design example.

A. Document Revision History for the Intel Stratix 10 Avalon-ST Hard IP for PCIe Design Example User Guide

A.1. Intel Stratix 10 Avalon Streaming (Avalon-ST) IP for PCIe Design Example User Guide Revision History

Date	Software Version	Changes
December 2019	17.1	Added the link to the Avalon Streaming IP for PCIe User Guide.
February 2018	17.1	Rewrote the Design Example Description section to change the description from that of a Simple DMA design to one for a PIO design.
November 2017	17.1	Made the following changes: • Added compilation support. • Added simulation support for NCSim. • Added Linux driver for hardware example. • Revised *Generating the Design* topic to create a single `.ip` for PCIe instead of a complete system design. Generating the testbench creates a design example from the `.ip` . • Added web link to information on using the PCIe Link Inspector.
May 2017	Quartus Prime Pro v17.1 Stratix 10 ES Editions	• Added support for Gen3 x16 Programmer Object File (`*.pof`) generation using a simplified design example. • Corrected minor errors and typos.