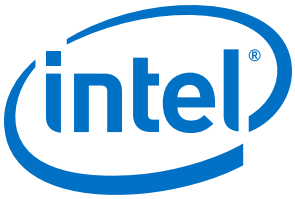


E-tile Hard IP for Ethernet Intel® Stratix® 10 FPGA IP Design Example User Guide

Updated for Intel® Quartus® Prime Design Suite: **18.1.1**

UG-20172 | 2019.02.14
Latest document on the web: **PDF** | **HTML**

Contents

1. Quick Start Guide

The E-Tile Hard IP for Ethernet Intel FPGA IP core provides a simulation testbench and a hardware design example that supports compilation and hardware testing. When you generate the design example, the parameter editor automatically creates the files necessary to simulate, compile, and test the design in hardware.

In addition, you can download the compiled hardware design to the Intel® Stratix® 10 TX Transceiver Signal Integrity Development Kit. Intel provides a compilation-only example project that you can use to quickly estimate IP core area and timing.

Table 1. List of Supported Design Example Variants

Data Rate	Variant	Simulation	Compilation-Only Project	Hardware Design Example
10GE	Single or multi channels Media Access Controller (MAC) + Physical Coding Sublayer (PCS) with optional 1588 Precision Time Protocol (PTP)	√	√	√
	Single channel PCS	√	√	√
	Single channel Optical Transport Network (OTN)	√	√	X
	Single channel Flexible Ethernet (FlexE)	√	√	X
25GE	Single or multi channels MAC + PCS with optional Reed Solomon Forward Error Correction (RSFEC) and optional PTP	√	√	√
	Single channel PCS with optional Reed Solomon Forward Error Correction (RSFEC)	√	√	√
	Single channel OTN with optional RSFEC	√	√	X
	Single channel FlexE with optional RSFEC	√	√	X
100GE	MAC+ PCS with optional:	√	√	√

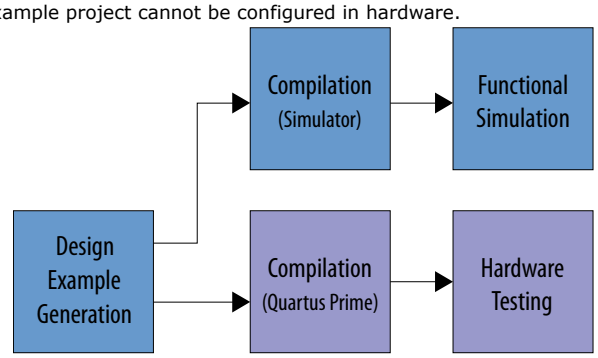
continued...

Data Rate	Variant	Simulation	Compilation-Only Project	Hardware Design Example
	• (528,514) or (544, 514) RSFEC • asynchronous clock			
	PCS with optional (528,514) or (544, 514) RSFEC	√	√	√
	OTN with optional (528,514) or (544, 514) RSFEC	√	√	X
	FlexE with optional (528,514) or (544, 514) RSFEC	√	√	X

Note: The E-Tile Hard IP for Ethernet Intel FPGA IP provides preliminary support for the OTN feature. For further inquiries, contact your nearest Intel sales representative or file an Intel Premier Support (IPS) case on https://www.intel.com/content/www/us/en/programmable/my-intel/mal-home.html.

Figure 1. Development Steps for the Design Example

The compilation-only example project cannot be configured in hardware.



1.1. Directory Structure

The E-Tile Hard IP for Ethernet Intel FPGA design example file directories contain the following generated files for the design examples.

Figure 2. E-Tile Hard IP for Ethernet Intel FPGA 10/25-GE with Optional RSFEC and Optional PTP Design Example Directory Structure

<*datarate*> is either "10" or "25", depending on your IP core variation.

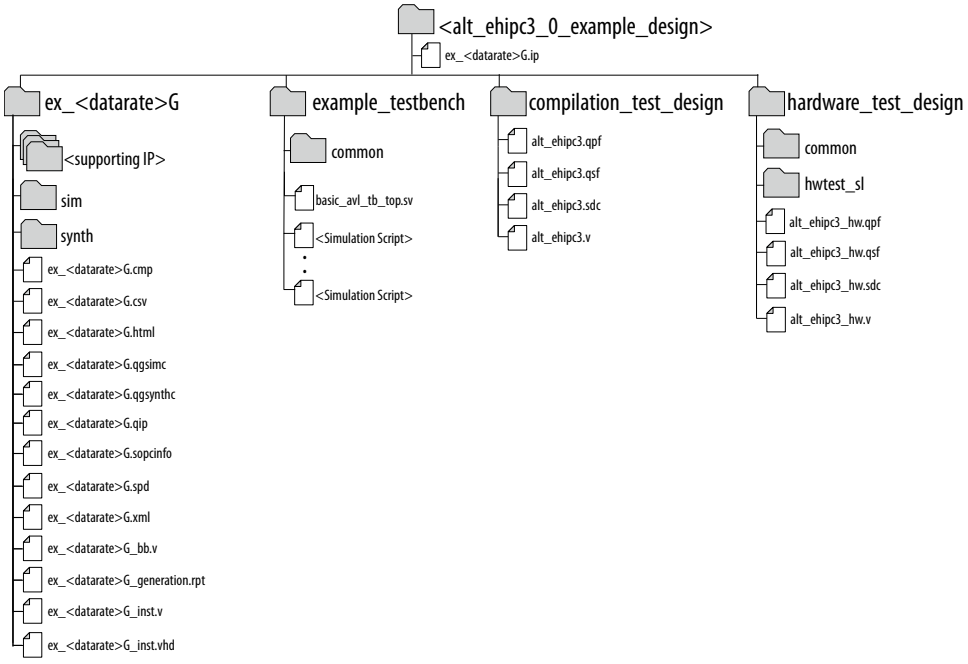


Figure 3. E-Tile Hard IP for Ethernet Intel FPGA 100GE with Optional RSFEC Design Example Directory Structure

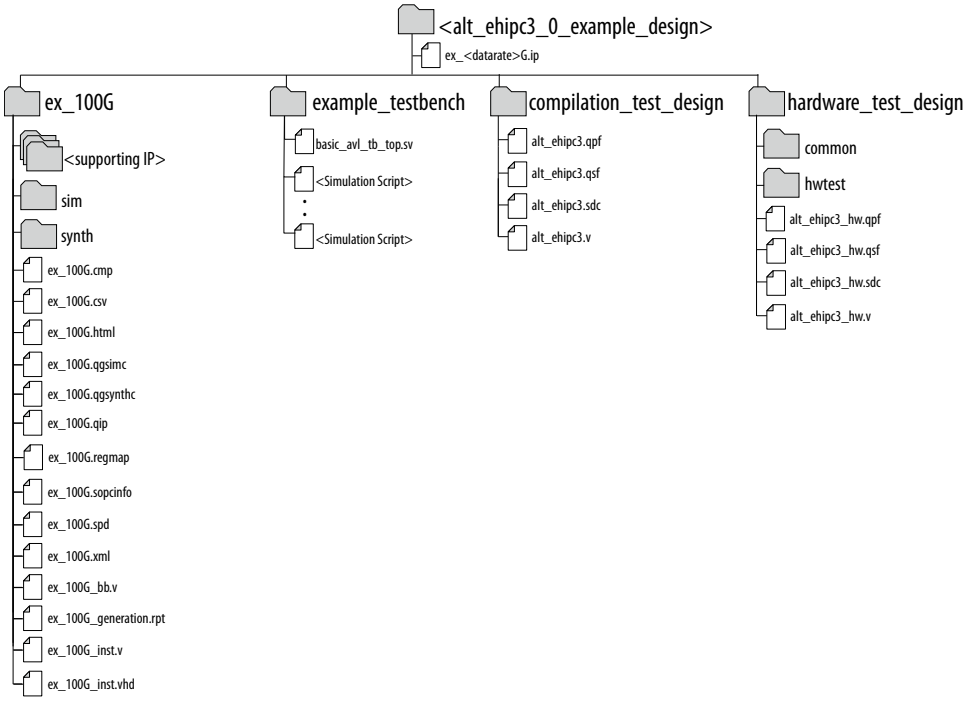


Table 2. E-Tile Hard IP for Ethernet Intel FPGA IP Core Testbench File Descriptions

File Names	Description
Key Testbench and Simulation Files	
`<design_example_dir>/example_testbench/basic_avl_tb_top.sv`	Top-level testbench file. The testbench instantiates the DUT and runs Verilog HDL tasks to generate and accept packets.
Testbench Scripts	
`<design_example_dir>/example_testbench/run_vsim.do`	The Mentor Graphics ModelSim* script to run the testbench.
`<design_example_dir>/example_testbench/run_vcs.sh`	The Synopsys VCS* script to run the testbench.
`<design_example_dir>/example_testbench/run_vcsmx.sh`	The Synopsys VCS MX* script (combined Verilog HDL and SystemVerilog with VHDL) to run the testbench.
`<design_example_dir>/example_testbench/run_ncsim.sh`	The Cadence NCSim* script to run the testbench.
`<design_example_dir>/example_testbench/run_xcelium.sh`	The Xcelium* script to run the testbench.

Table 3. IP Core Hardware Design Example File Descriptions

File Names	Description
`<design_example_dir>/hardware_test_design/alt_ehipc3_hw.qpf`	Intel Quartus® Prime project file
`<design_example_dir>/hardware_test_design/alt_ehipc3_hw.qsf`	Intel Quartus Prime project settings file
`<design_example_dir>/hardware_test_design/alt_ehipc3_hw.sdc`	Synopsys Design Constraints files. You can copy and modify these files for your own design.
`<design_example_dir>/hardware_test_design/alt_ehipc3_hw.v`	Top-level Verilog HDL design example file
`<design_example_dir>/hardware_test_design/common/`	Hardware design example support files
`hwtest_sl/main_script.tcl` (10GE/25GE) `hwtest/main.tcl` (100GE)	Main file for accessing System Console

1.2. Generating the Design

Figure 4. Procedure

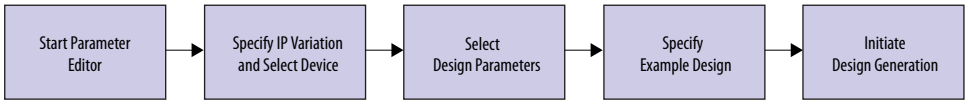
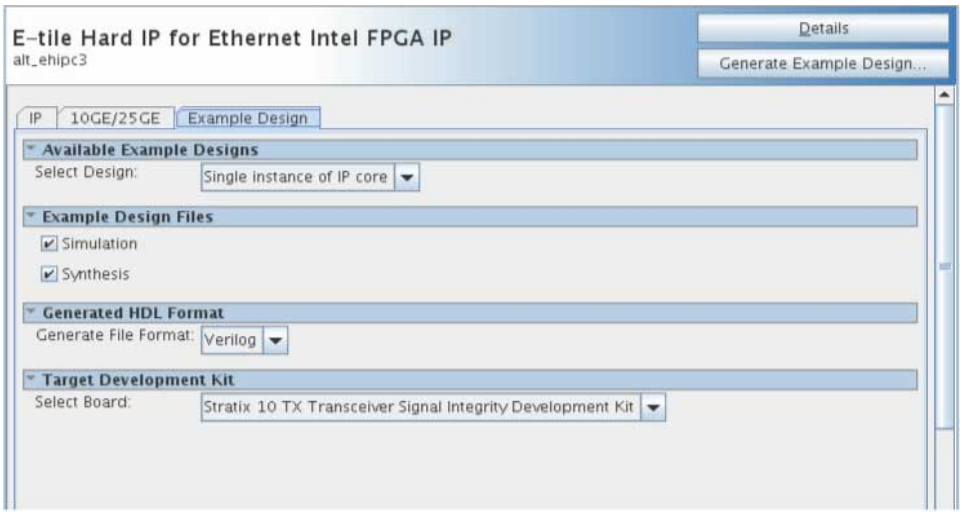


Figure 5. Example Design Tab in the E-Tile Hard IP for Ethernet Intel FPGA Parameter Editor



If you do not already have an Intel Quartus Prime Pro Edition project in which to integrate your E-Tile Hard IP for Ethernet Intel FPGA IP core, you must create one.

1. In the Intel Quartus Prime Pro Edition, click **File ➤ New Project Wizard** to create a new Quartus Prime project, or **File ➤ Open Project** to open an existing Intel Quartus Prime project. The wizard prompts you to specify a device.
2. Specify the device family **Intel Stratix 10** and select a device that meets all of these requirements:
   - Transceiver tile is E-tile
   - Transceiver speed grade is 1 or 2
   - Core speed grade is 1 or 2
3. Click **Finish**.

Follow these steps to generate the E-Tile Hard IP for Ethernet Intel FPGA hardware design example and testbench:

1. In the IP Catalog, locate and select **E-Tile Hard IP for Ethernet Intel FPGA**. The **New IP Variation** window appears.
2. Specify a top-level name <*your_ip*> for your custom IP variation. The parameter editor saves the IP variation settings in a file named <*your_ip*>.ip.
3. Click **OK**. The parameter editor appears.
4. On the **IP**, 100GE, or 10GE/25GE tabs, specify the parameters for your IP core variation.
5. The hardware design examples provided, enable internal serial loopback by default. To run the designs with external loopback mode, select **Enable adaptation load soft IP** parameter in the **PMA Adaptation** tab.
6. Select a PMA adaptation preset for **PMA adaptation Select** parameter.
7. Click **PMA Adaptation Preload** to load the initial and continuous adaptation parameters.

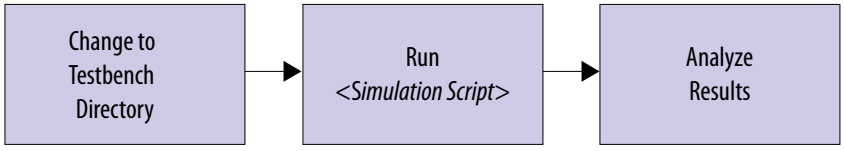
8. Specify the number of PMA configurations to support when multiple PMA configurations are enabled using **Number of PMA conguration** parameter.
9. Select which PMA configuration to load or store using **Select a PMA configuration to load or store**.
10. Click **Load adaptation from selected PMA configuration** to load the selected PMA configuration settings.
11. On the **Example Design** tab, under **Example Design Files**, select the **Simulation** option to generate the testbench and the compilation-only project. Select the **Synthesis** option to generate the hardware design example. You must select at least one of the **Simulation** and **Synthesis** options to generate the design example.
12. On the **Example Design** tab, under **Generated HDL Format**, select **Verilog** HDL or **VHDL**. If you select **VHDL**, you must simulate the testbench with a mixed-language simulator. The device under test in the `ex_<datarate>` directory is a VHDL model, but the main testbench file is a System Verilog file.
13. Under **Target Development Kit**, select the **Stratix 10 TX Transceiver Signal Integrity Development Kit** or select **None**. The compilation-only and hardware design examples target your project device. For the hardware design to functional correctly, you must ensure your project device is the same device on your development kit.
14. Click the **Generate Example Design** button. The **Select Example Design Directory** window appears.
15. If you want to modify the design example directory path or name from the defaults displayed (`alt_ehipc3_0_example_design`), browse to the new path and type the new design example directory name (*<design_example_dir>*).

Related Information

- E-Tile Hard IP for Ethernet Intel FPGA IP Core Parameters
  Provides more information about customizing your IP core.
- Intel Quartus Prime Pro Edition Intel Stratix 10 E-Tile Transceiver PHY User Guide
  More information parameters in **PMA Adaptation** tab.
- Intel Stratix 10 TX Signal Integrity Development Kit Webpage

1.3. Simulating the E-Tile Hard IP for Ethernet Intel FPGA Design Example Testbench

Figure 6. Procedure



Follow these steps to simulate the testbench:

1. Change to the testbench simulation directory `<design_example_dir>/example_testbench`.
2. Run the simulation script for the supported simulator of your choice. The script compiles and runs the testbench in the simulator. Refer to the table *Steps to Simulate the Testbench*.
3. Analyze the results. The successful testbench sends ten or fourteen packets, receives the same number of packets, and displays `"Testbench complete."`

Table 4. Steps to Simulate the Testbench

Simulator	Instructions
Mentor Graphics ModelSim*	In the command line, type `vsim -do run_vsim.do` If you prefer to simulate without bringing up the ModelSim GUI, type `vsim -c -do run_vsim.do` *Note:* The ModelSim - Intel FPGA Edition simulator does not have the capacity to simulate this IP core. You must use another supported ModelSim simulator such as ModelSim SE.
Cadence NCSim*	In the command line, type `sh run_ncsim.sh`
Synopsys VCS*	In the command line, type `sh run_vcs.sh` or `sh run_vcsmx.sh`
Xcelium*	In the command line, type `sh run_xcelium.sh`

1.4. Compiling the Compilation-Only Project

To compile the compilation-only example project, follow these steps:

1. Ensure compilation design example generation is complete.
2. In the Intel Quartus Prime Pro Edition software, open the Intel Quartus Prime Pro Edition project `<design_example_dir>/compilation_test_design/alt_ehipc3.qpf`.
3. On the Processing menu, click **Start Compilation**.

After successful compilation, reports for timing and for resource utilization are available in your Intel Quartus Prime Pro Edition session.

Related Information

Block-Based Design Flows

1.5. Compiling and Configuring the Design Example in Hardware

To compile the hardware design example and configure it on your Intel Stratix 10 device, follow these steps:

1. Ensure hardware design example generation is complete.
2. In the Intel Quartus Prime Pro Edition software, open the Intel Quartus Prime project `<design_example_dir>/hardware_test_design/alt_ehip3.qpf`.
3. On the Processing menu, click **Start Compilation**.
4. After successful compilation, a `.sof` file is available in `<design_example_dir>/hardware_test_design/output_files` directory. Follow these steps to program the hardware design example on the Intel Stratix 10 device:

a. On the **Tools** menu, click **Programmer**.
b. In the Programmer, click **Hardware Setup**.
c. Select a programming device.
d. Select and add the Intel Stratix 10 Transceiver Signal Integrity Development Kit to which your Intel Quartus Prime Pro Edition session can connect.
e. Ensure that **Mode** is set to **JTAG**.
f. Select the Intel Stratix 10 device and click **Add Device**. The Programmer displays a block diagram of the connections between the devices on your board.
g. In the row with your `.sof`, check the box for the `.sof`.
h. Check the box in the **Program/Configure** column.
i. Click **Start**.

Related Information

- Block-Based Design Flows
- Programming Intel FPGA Devices
- Analyzing and Debugging Designs with System Console

1.6. Testing the E-Tile Hard IP for Ethernet Intel FPGA Hardware Design Example

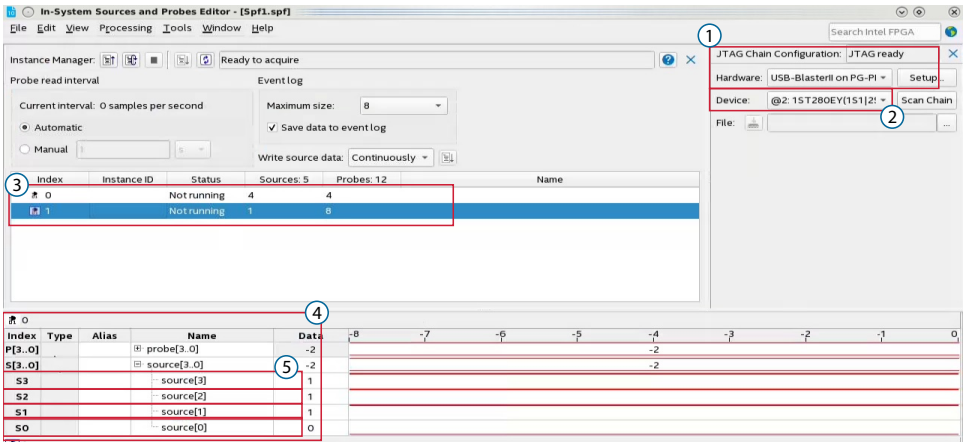
After you compile the E-Tile Hard IP for Ethernet Intel FPGA IP core design example and configure it on your Intel Stratix 10 device, you can use the System Console to program the IP core and its embedded Native PHY IP core registers.

1.6.1. 10GE/25GE MAC + PCS with Optional RSFEC and Optional PTP Hardware Design Example

To turn on the System Console and test the hardware design example, follow these steps:

1. After the hardware design example is configured on the Intel Stratix 10 device, in the Intel Quartus Prime Pro Edition software, on the **Tools** menu, click **In-System Sources and Probes Editor**.

Figure 7. In-System Sources and Probes Editor



1. JTAG Chain Configuration shows USB-BlasterII connection to the development kit.
2. Shows the device used on the development kit.
3. Shows the number of instances connected to the JTAG chain.
4. Shows the number of probes and sources connected to instance 0.
5. S3 connected to sl_csr_rst_n
   S2 connected to sl_tx_rst_n
   S1 connected to sl_rx_rst_n
   S0 connected to i_reconfig_reset

2. In the **JTAG Chain Configuration** window, select the USB connection that is connected to the development kit.
3. Next, from the **Device** list, select the device with 1ST280EY string in the name. The **Ready to acquire** status will appear at the bottom of the **Instance Manager** window if the correct device is selected.
4. A list of instances will be shown once the connection is acquired. There are four sources under index 0. These sources have the following connections:

Source	Signal
source[3]	`sl_csr_rst_n` (active low)
source[2]	`sl_tx_rst_n` (active low)
source[1]	`sl_rx_rst_n` (active low)
source[0]	`i_reconfig_reset` (active high)

5. Assert value of 1 to **source[3]**, **source[2]**, and **source[1]**.
6. Toggle **source[0]** to initiate reset for the transceiver and Ethernet reconfiguration interfaces.

7. Once the reset is initiated, on the **Tools** menu, click **System Debugging Tools ➤ System Console**.
8. In the Tcl Console pane, type `cd hwtest_sl` to change directory to `<design_example_dir>/hardware_test_design/hwtest_sl`.
9. Type `set <command_setting>` to configure the test according to your design configuration:

Command Setting	Description
totalChannel	Set this value according to the value of **Number of Channels of 10GE/25GE** parameter in your design. The default value is 1. Example, in the system console type `set totalChannel 2` to change the number of channels to 2.
jtag_port_id	Set this value to the JTAG port ID that is connected to the development kit. Example, in the system console type `set jtag_port_id 1` to change the JTAG ID to 1.
enablePTP	Set this to 1 if PTP is enabled in the design. Otherwise set the value to 0. The default value is 0. Example, in the system console type `set enablePTP 1` to enable PTP.
speed	Choose the following option according to the design example variation: • `10G` for 10 Gbps data rate • `25G` for 25 Gbps data rate • `25G_fec` for 25 Gbps data rate with RSFEC enabled Example, in the system console type `set speed 25G_fec` to set the data rate to 25G with RSFEC enabled.

10. Type `source main_script.tcl` to enable the internal loopback and run the test.

Configuring the hardware test in System Console:

```
% set totalChannel 2
2
% set jtag_port_id 1
1
% set speed 10G
10G
% source main_script.tcl
Info: Number of Channels = 2
Info: JTAG Port ID       = 1
Info: PTP Enable         = 0
Info: Speed              = 10G
```

Related Information

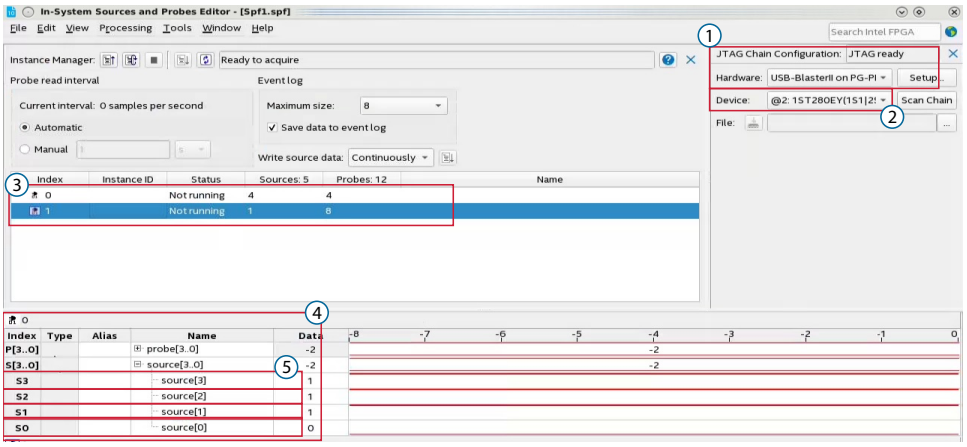
Intel Quartus Prime Pro Edition User Guide: Debug Tools - In-System Sources and Probes

1.6.2. 10GE/25GE PCS Only with Optional RSFEC Hardware Design Example

To turn on the System Console and test the hardware design example, follow these steps:

1. After the hardware design example is configured on the Intel Stratix 10 device, in the Intel Quartus Prime Pro Edition software, on the **Tools** menu, click **In-System Sources and Probes Editor**.

Figure 8. In-System Sources and Probes Editor



1. JTAG Chain Configuration shows USB-BlasterII connection to the development kit.
2. Shows the device used on the development kit.
3. Shows the number of instances connected to the JTAG chain.
4. Shows the number of probes and sources connected to instance 0.
5. S3 connected to sl_csr_rst_n
   S2 connected to sl_tx_rst_n
   S1 connected to sl_rx_rst_n
   S0 connected to i_reconfig_reset

2. In the **JTAG Chain Configuration** window, select the USB connection that is connected to the development kit.
3. Next, from the **Device** list, select the device with 1ST280EY string in the name. The **Ready to acquire** status will appear at the bottom of the **Instance Manager** window if the correct device is selected.
4. A list of instances will be shown once the connection is acquired. There are four sources under instance 0. These sources have the following connections:

Source	Signal
source[3]	`sl_csr_rst_n` (active low)
source[2]	`sl_tx_rst_n` (active low)
source[1]	`sl_rx_rst_n` (active low)
source[0]	`i_reconfig_reset` (active high)

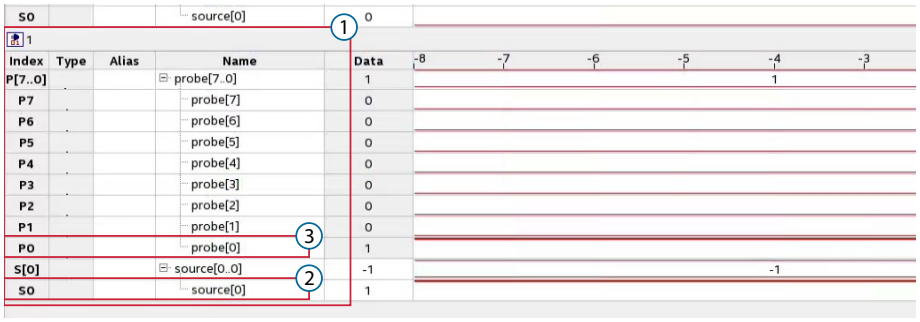
5. Set value of 1 to **source[3]**, **source[2]**, and **source[1]** to bring the ethernet circuit out reset.
6. Toggle **source[0]** to initiate reset for the transceiver and Ethernet reconfiguration interfaces.

7. Click **Tools ➤ System Debugging Tools ➤ System Console**.
8. In the Tcl Console pane, type `cd hwtest_sl` to change directory to `<design_example_dir>/hardware_test_design/hwtest_sl`.
9. Type `set <command_setting>` to configure the test according to your design configuration:

Command Setting	Description
totalChannel	Set this value according to the value of **Number of Channels of 10GE/25GE** parameter in your design. The default value is 1. Example, in the system console type `set totalChannel 2` to change the number of channels to 2. *Note:* E-Tile Hard IP for Ethernet Intel FPGA IP does not support multichannel PCS variation.
jtag_port_id	Set this value to the JTAG port ID that is connected to the development kit. Example, in the system console type `set jtag_port_id 0` to set the JTAG ID to 0.
speed	Choose the following option according to the design example variation: • `10G` for 10 Gbps data rate • `25G` for 25 Gbps data rate • `25G_fec` for 25 Gbps data rate with RSFEC enabled Example, in the system console type `set speed 25G_fec` to set the data rate to 25 Gbps with RSFEC enabled.

```
% set totalChannel 1
1
% set jtag_port_id 1
1
% set speed 25G_fec
25G_fec
```

10. Type `source tests/c3_elane_xcvr_loopback_test.tcl` to enable transceiver internal loopback.
11. In the **In-System Sources and Probes Editor** editor, under index 1, assert **source[0]** to 1. This source is connected to `trigger_to_send_packet` signal. This signal is an active high signal.



1. Shows the number of probes and sources connected to instance 1.
2. S0 connected to trigger_to_send_packet. Assert to start sending PCS66 packets.
3. P0 connected to checker_pass. This signal changes to 1 when the test has passed.

12. The status in the **Instance manager** for instance 1 will show **Not running**.
13. At the **Processing** tab, click **Continuously Read Probe Data** to continuously sample the probe data of the selected instance.
14. **Probe[0]** of instance 1 is connected to `checker_pass` signal. This signal changes value from 0 to 1 when the packets are verified and passed.

Related Information

Intel Quartus Prime Pro Edition User Guide: Debug Tools - In-System Sources and Probes

1.6.3. 100GE MAC + PCS with Optional (528,514) RSFEC and PMA Calibration Hardware Design Example

This hardware design example enables internal serial loopback mode by default. To run the hardware design with external loopback mode, select **Enable adaptation load soft IP** in the parameter editor before generating the design example.

To turn on the System Console and test the hardware design example, follow these steps:

1. After the hardware design example is configured on the Intel Stratix 10 device, in the Intel Quartus Prime Pro Edition software, on the **Tools** menu, click **System Debugging Tools ➤ System Console**.
2. In the Tcl Console pane, type `cd hwtest` to change directory to `<design_example_dir>/hardware_test_design/hwtest`.
3. Type `source main.tcl` to open a connection to the JTAG master.

   You can use the following design example commands to configure the 100GE hardware design example test with internal serial loopback mode. Example, in the system console, type `run_test` and press `Enter`.

   - `run_test`: To run hardware design example tests.
   - `chkphy_status`: Displays the clock frequencies and PHY lock status.
   - `chkmac_stats`: Displays the values in the MAC statistics counters.
   - `clear_all_stats`: Clears the IP core statistics counters.
   - `start_pkt_gen`: Starts the packet generator.

- `stop_pkt_gen`: Stops the packet generator.
- `loop_on`: Turns on internal serial loopback.
- `loop_off`: Turns off internal serial loopback.
- `reg_read <addr>`: Returns the IP core register value at <*addr*>. Example, to read TX datapath PCS ready register, type `reg_read 0x322`.
- `reg_write <addr> <data>`: Writes <*data*> to the IP core register at address <*addr*>. Example, to initiate soft reset on RX PCS, type `reg_write 0x310 0x0004>`

4. To run the MAC+PCS with (528,514) RSFEC and PMA calibration design example in external loopback mode, open `hardware_test_design/hwtest/main.tcl` file and uncomment `start_pma_ical` command. Make sure the **Enable adaptation load soft IP** is selected and the **PMA adaptation Select** is set to**NRZ_28Gbps_LR**, **NRZ_28Gbps_VSR**, or **NRZ_10Gbps** before generating the design example.
5. Disable the internal serial loopback mode by using `loop_off` command.

   You can use the following design example commands to configure the 100GE hardware design example test with external loopback mode.

   - `start_pma_ical`: Performs PMA calibration on external loopback or external devices connection tests.
   - `start_spicorst03`: Performs NRZ transceiver SERDES reset.
   - `ical_16_NoPrbsNoLdEL03`: Performs NRZ PMA calibration using NIOS firmware with default values.

     Important: All the values set in this design example are tested with Intel Stratix 10 TX Transceiver Signal Integrity Development Kit. You may need to customize the PMA adaptation configuration values if you are running this design example on boards other than the Intel Stratix 10 TX Transceiver Signal Integrity Development Kit.
   - `chk_ical_status`: Checks calibration status.
   - `ld_rcp`: Loads PMA configuration settings based on the selection set in the **Select a PMA configuration to load or store** in the parameter editor.

     Important: All the values set in this design example are tested with Intel Stratix 10 TX Transceiver Signal Integrity Development Kit. You may need to customize the PMA adaptation configuration values if you are running this design example on boards other than the Intel Stratix 10 TX Transceiver Signal Integrity Development Kit.
   - `chk_rcp_status`: Checks PMA configuration settings load status and retry if necessary.

Related Information

- Intel Quartus Prime Pro Edition User Guide: Debug Tools - In-System Sources and Probes
- Intel Quartus Prime Pro Edition Intel Stratix 10 E-Tile Transceiver PHY User Guide
  More information on parameters in **PMA Adaptation** tab.

1.6.4. 100GE MAC + PCS with Optional (544,514) RSFEC and PMA Calibration Hardware Design Example

This hardware design example enables internal serial loopback mode by default. To run the hardware design with external loopback mode, select **Enable adaptation load soft IP** in the parameter editor before generating the design example.

To turn on the System Console and test the hardware design example, follow these steps:

1. After the hardware design example is configured on the Intel Stratix 10 device, in the Intel Quartus Prime Pro Edition software, on the **Tools** menu, click **System Debugging Tools ➤ System Console**.
2. In the Tcl Console pane, type `cd hwtest` to change directory to `<design_example_dir>/hardware_test_design/hwtest`.
3. Type `source main.tcl` to open a connection to the JTAG master.

   You can use the following design example commands to configure the 100GE hardware design example test with internal serial loopback mode. Example, in the system console, type `run_test_pam4` and press `Enter`.

   - `run_test_pam4`: To run hardware design example tests without PMA calibration.
   - `start_pma_02_ical`: To perform PMA adaptation calibration using NIOS firmware.
   - `chkphy_status`: Displays the clock frequencies and PHY lock status.
   - `chkmac_stats`: Displays the values in the MAC statistics counters.
   - `clear_all_stats`: Clears the IP core statistics counters.
   - `start_pkt_gen`: Starts the packet generator.
   - `stop_pkt_gen`: Stops the packet generator.
   - `loop_on_pam4`: Turns on internal serial loopback
   - `loop_off`: Turns off internal serial loopback.
   - `reg_read <addr>`: Returns the IP core register value at `<addr>`. Example, to read TX datapath PCS ready register, type `reg_read 0x322`.
   - `reg_write <addr> <data>`: Writes `<data>` to the IP core register at address `<addr>`. Example, to initiate soft reset on RX PCS, type `reg_write 0x310 0x0004>`
   - `chk_ical_status_02`: Checks for PAM4 calibration status.
4. To run the MAC+PCS with (544,514) RSFEC and PMA calibration design example in external loopback mode, make sure the **Enable adaptation load soft IP** is selected and the **PMA adaptation Select** is set to**PAM4_56Gbps_LR** or **PAM4_56Gbps_VSR** before generating the design example.
5. Disable the internal serial loopback mode by using `loop_off` command.

You can use the following design example commands to configure the 100GE hardware design example test with external loopback mode.

- `start_pma_02_ical_ex`: Performs PMA adaptation calibration using NIOS firmware for external loopback mode,
- `start_spicorst02`: Performs PAM4 transceiver SERDES reset.
- `ical_16_NoPrbsNoLdELCntPC02`: Performs PAM4 PMA calibration using NIOS firmware with default values.

  Important: All the values set in this design example are tested with Intel Stratix 10 TX Transceiver Signal Integrity Development Kit. You may need to customize the PMA adaptation configuration values if you are running this design example on boards other than the Intel Stratix 10 TX Transceiver Signal Integrity Development Kit.
- `chk_ical_status_02`: Checks for PAM4 calibration status.

Related Information

- Intel Quartus Prime Pro Edition User Guide: Debug Tools - In-System Sources and Probes
- Intel Quartus Prime Pro Edition Intel Stratix 10 E-Tile Transceiver PHY User Guide
  More information on parameters in **PMA Adaptation** tab.

1.6.5. 100GE PCS Only with Optional (528,514) RSFEC or (544,514) RSFEC, and Optional PTP Hardware Design Example

To turn on the System Console and test the hardware design example, follow these steps:

1. After the hardware design example is configured on the Intel Stratix 10 device, in the Intel Quartus Prime Pro Edition software, on the **Tools** menu, click **System Debugging Tools ➤ System Console**.
2. In the Tcl Console pane, type `cd hwtest` to change directory to `<design_example_dir>/hardware_test_design/hwtest`.
3. Type `source main.tcl` to open a connection to the JTAG master.
4. Type `pcs_only_traffic_test <number of iteration>` to run the specified iteration of PCS only with (528,514) RSFEC hardware design example test. If no value is specified, the test runs only 1 iteration. Each packet generated for every iterations are in random number of frames, size, and types.
5. Type `pcs_only_traffic_test_pam4 <number of interation>` to run the specified iteration of PCS only with (544,514) RSFEC hardware design example test. If no value is specified, the test runs only 1 iteration. Each packet generated for every iterations are in random number of frames, size, and types.

Related Information

Intel Quartus Prime Pro Edition User Guide: Debug Tools - In-System Sources and Probes

2. 10GE/25GE with Optional RSFEC Design Examples

The 10GE/25GE design example demonstrates an Ethernet solution for Intel Stratix 10 devices using the E-Tile Hard IP for Ethernet Intel FPGA IP core with the following variants:

Table 5. Supported Design Example Variants for 10GE/25GE

RSFEC is not supported in 10GE variant.

Number of Channel	Variant				
	MAC + PCS with Optional RSFEC	MAC + PCS with Optional RSFEC and PTP	PCS Only with Optional RSFEC	OTN with Optional RSFEC	FlexE with Optional RSFEC
1	Simulation, compilation-only project, and hardware design example	Simulation, compilation-only project, and hardware design example	Simulation, compilation-only project, and hardware design example	Simulation and compilation-only project	Simulation and compilation-only project
2 - 3	Simulation, compilation-only project, and hardware design example	N/A	N/A	N/A	N/A
4	Simulation, compilation-only project, and hardware design example	Simulation, compilation-only project, and hardware design example	N/A	N/A	N/A

2.1. Simulation Design Examples

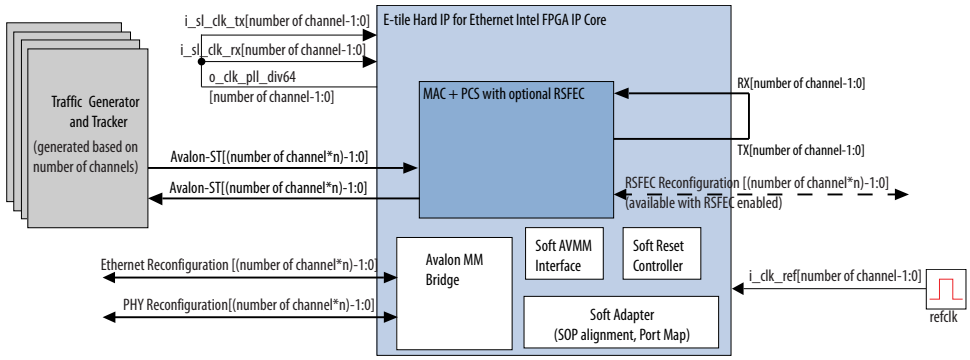
2.1.1. Non-PTP 10GE/25GE MAC+PCS with Optional RSFEC Simulation Design Example

The simulation block diagram below is generated using the following settings:

1. **1 to 4 10GE/25GE with optional RSFEC** or **100GE or 1 to 4 channel 10GE/25GE with optional RSFEC and PTP** as the core variant.
2. **10GE/25GE Channel(s)** as **Active channel(s) at startup** if you choose **100GE or 1 to 4 channel 10GE/25GE with optional RSFEC and PTP** as the core variant.
3. **Enable RSFEC** to use RSFEC feature.
4. **10G** or **25G** as the Ethernet rate.

   Note: RSFEC is not supported in 10GE variant.

Figure 9. Simulation Block Diagram for Non-PTP E-Tile Hard IP for Ethernet Intel FPGA 10GE/25GE MAC+PCS with Optional RSFEC Design Examples



The testbench sends traffic through the IP core, exercising the transmit side and receive side of the IP core.

To speed up simulation, the IP core simulation model sends alignment marker tags at shorter intervals than required by the IEEE Ethernet standard. The standard specifies an alignment marker interval of 16384 words in each virtual lane. The simulation model with the testbench implements an alignment marker interval of 512 words.

The successful test run displays output confirming the following behavior:

1. Waiting for PLL to lock.
2. Waiting for RX transceiver reset to complete.
3. Waiting for RX alignment.
4. Sending 10 packets.
5. Receiving those packets.
6. Displaying `Testbench complete.`

The following sample output illustrates a successful simulation test run for a 25GE, MAC+PCS with RSFEC, non-PTP IP core variation.

```
# Ref clock is 156.25 MHz
# Channel 0 - waiting for EHIP Ready....
# Channel 0 - EHIP READY is 1 at time           2472365000
# Channel 0 - Waiting for RX Block Lock
# Channel 0 - EHIP RX Block Lock  is high at time           2507639043
# Channel 0 - Waiting for RX alignment
# Channel 0 - RX deskew locked
# Channel 0 - RX lane aligmnent locked
# Channel 0 - TX enabled
# ** Sending Packet            1...
# ** Sending Packet            2...
# ** Sending Packet            3...
# ** Sending Packet            4...
# ** Sending Packet            5...
# ** Sending Packet            6...
# ** Sending Packet            7...
# ** Sending Packet            8...
# ** Sending Packet            9...
# ** Sending Packet           10...
# Channel 0 - Received Packet           1...
# Channel 0 - Received Packet           2...
# Channel 0 - Received Packet           3...
# Channel 0 - Received Packet           4...
```

```
# Channel 0 - Received Packet           5...
# Channel 0 - Received Packet           6...
# Channel 0 - Received Packet           7...
# Channel 0 - Received Packet           8...
# Channel 0 - Received Packet           9...
# Channel 0 - Received Packet          10...
# **
# ** Reading KR CSR -C0
# ** Address offset = 000c0, ReadData  = 737d0381
# ** AVMM access CSR registers read/write check for ETH amd XCVR CH0
# ** Address offset = 00301, ReadData  = 00000000
# ** Address offset = 00301, WriteData = c3ec3ec3
# ** Address offset = 00301, ReadData  = c3ec3ec3
# ** Address offset = 00301, WriteData = 00000000
# ** Address offset = 00300, ReadData  = 11112015
# ** Address offset = 00400, ReadData  = 11112015
# ** Address offset = 00a00, ReadData  = 11112015
# ** Address offset = 00b00, ReadData  = 11112015
# ** Address offset = 00836, ReadData  = 0000000a
# ** Address offset = 00936, ReadData  = 0000000a
# ** Address offset = 00804, ReadData  = 00000000
# ** Address offset = 00904, ReadData  = 00000000
# ** Address offset = 00322, ReadData  = 00000001
# ** Address offset = 00084, WriteData = ffffffff
# ** Address offset = 00084, ReadData  = 000000ff
# ** Address offset = 00084, WriteData = 00000000
# ** Address offset = 00230, WriteData = ffffffff
# ** Address offset = 00230, ReadData  = 000000ff
# ** Address offset = 00230, WriteData = 0000007b
# **
# ** AVMM access CSR registers read/write check for ETH RSFEC
# ** Address offset = 10000, ReadData  = 00000001
# ** Address offset = 10000, WriteData = ffffffff
# ** Address offset = 10000, ReadData  = 000000fd
# ** Address offset = 10004, ReadData  = 00000004
# ** Address offset = 10010, ReadData  = 00000061
# ** Address offset = 10011, ReadData  = 00000066
# ** Address offset = 10000, WriteData = 00000001
# ** Check KR CSR Status - C0
# ** Address offset = 000b1, ReadData  = 00040801
# ** Address offset = 000d2, ReadData  = 00000001
# **
# ** Testbench complete.
# **
# ****************************************
# ** Note: $finish    : ./basic_avl_tb_top.sv(415)
#    Time: 2628595 ns  Iteration: 0  Instance: /basic_avl_tb_top
```

Related Information

Simulating the E-Tile Hard IP for Ethernet Intel FPGA Design Example Testbench on page 9

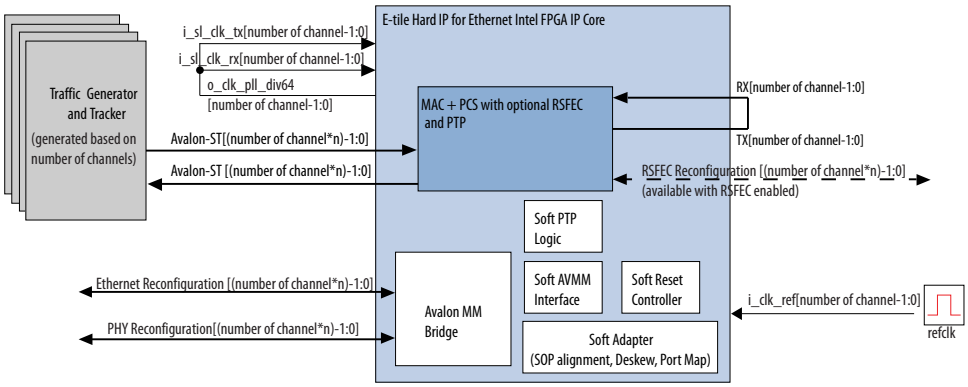
2.1.2. 10GE/25GE MAC+PCS with Optional RSFEC and PTP Simulation Design Example

The simulation block diagram below is generated using the following settings:

1. **100GE or 1 to 4 channel 10GE/25GE with optional RSFEC and PTP** as the core variant.
2. **10G/25GE channels** as **Active channel(s) at startup**.
3. **Enable IEEE 1588 PTP**.
4. **Enable RSFEC** to use RSFEC feature.
5. **10G** or **25G** as the Ethernet rate.

Note: RSFEC is not supported in 10GE variant.

Figure 10. Simulation Block Diagram for E-Tile Hard IP for Ethernet Intel FPGA 10GE/25GE with Optional RSFEC and PTP Design Examples



In this design example, the testbench sends traffic through the IP core, exercising the transmit side and receive side of the IP core.

To speed up simulation, the IP core simulation model sends alignment marker tags at shorter intervals than required by the IEEE Ethernet standard. The standard specifies an alignment marker interval of 16384 words in each virtual lane. The simulation model with the testbench implements an alignment marker interval of 512 words.

The successful test run displays output confirming the following behavior:

1. Waiting for PLL to lock.
2. Waiting for RX transceiver reset to complete.
3. Waiting for RX alignment.
4. Sending 10 packets.
5. Receiving those packets.
6. Displaying `Testbench complete.`

The following sample output illustrates a successful simulation test run for a 25GE, MAC+PCS, RSFEC, PTP IP core variation.

```
# Channel 0 - EHIP Ready is high
# Channel 0 - Waiting for RX Block Lock
# Channel 0 - RX Block Lock is high
# Channel 0 - Waiting for RX alignment
# Channel 0 - RX lane aligmnent locked
# Channel 0 - Waiting for TX PTP Ready
# Channel 0 - TX PTP ready
# Channel 0 - Training RX PTP AIB deskew and waiting for RX PTP ready
# Channel 0 - Sending  Packet          1
# Channel 0 - Received Packet          1
# Channel 0 - Sending  Packet          2
# Channel 0 - Received Packet          2
# Channel 0 - Sending  Packet          3
# Channel 0 - Received Packet          3
# Channel 0 - Sending  Packet          4
# Channel 0 - Received Packet          4
# Channel 0 - RX PTP ready
.
.
(Repeat tests for Channel 1, Channel 2, and Channel 3)
```

```
.
.
# ====> writedata = 00000000
#
# Channel 0 - Configure TX extra latency
# ====> writedata = 0004267a
#
# Channel 0 - Configure RX extra latency
# ====> writedata = 8002d4de
#
# Channel 0 - TX enabled
# Channel 0 - Sending  Packet          1
# Channel 0 - Sending  Packet          2
# Channel 0 - Sending  Packet          3
# Channel 0 - Sending  Packet          4
# Channel 0 - Sending  Packet          5
# Channel 0 - Sending  Packet          6
# Channel 0 - Sending  Packet          7
# Channel 0 - Sending  Packet          8
# Channel 0 - Sending  Packet          9
# Channel 0 - Sending  Packet         10
# Channel 0 - Received Packet          1
# Channel 0 - Received Packet          2
# Channel 0 - Received Packet          3
# Channel 0 - Received Packet          4
# Channel 0 - Received Packet          5
# Channel 0 - Received Packet          6
# Channel 0 - Received Packet          7
# Channel 0 - Received Packet          8
# Channel 0 - Received Packet          9
# Channel 0 - Received Packet         10
# ====> writedata = 00000000
.
.
(Send and receive packets for Channel 1 and Channel 2)
.
.
# ====> writedata = 00000000
#
# Channel 3 - Configure TX extra latency
# ====> writedata = 0004267a
#
# Channel 3 - Configure RX extra latency
# ====> writedata = 800369d0
#
# Channel 3 - TX enabled
# Channel 3 - Sending  Packet          1
# Channel 3 - Sending  Packet          2
# Channel 3 - Sending  Packet          3
# Channel 3 - Sending  Packet          4
# Channel 3 - Sending  Packet          5
# Channel 3 - Sending  Packet          6
# Channel 3 - Sending  Packet          7
# Channel 3 - Sending  Packet          8
# Channel 3 - Sending  Packet          9
# Channel 3 - Sending  Packet         10
# Channel 3 - Received Packet          1
# Channel 3 - Received Packet          2
# Channel 3 - Received Packet          3
# Channel 3 - Received Packet          4
# Channel 3 - Received Packet          5
# Channel 3 - Received Packet          6
# Channel 3 - Received Packet          7
# Channel 3 - Received Packet          8
# Channel 3 - Received Packet          9
# Channel 3 - Received Packet         10
# ***************************************
# ** Testbench complete.
```

```
# ****************************************
# ** Note: $finish    : ./basic_avl_tb_top.sv(484)
#    Time: 473545955 ps  Iteration: 0  Instance: /basic_avl_tb_top
```

Related Information

Simulating the E-Tile Hard IP for Ethernet Intel FPGA Design Example Testbench on page 9

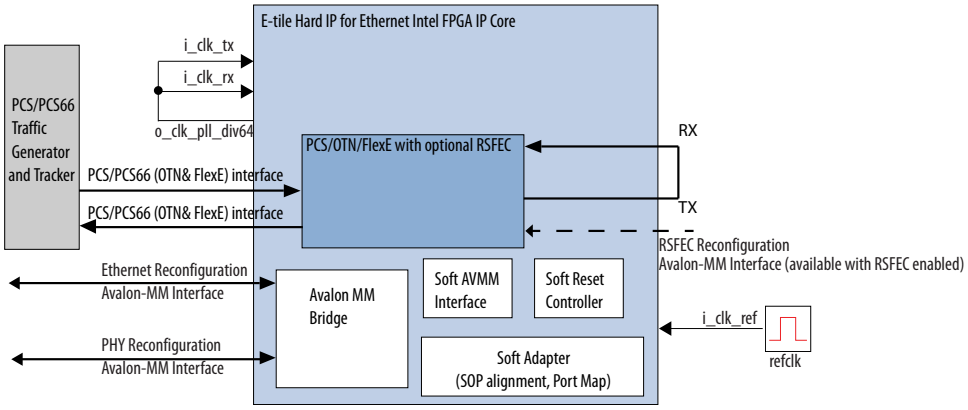
2.1.3. 10GE/25GE PCS Only, OTN, or FlexE with Optional RSFEC Design Example Simulation

The simulation block diagram below is generated using the following settings:

- **1 to 4 10GE/25GE with optional RSFEC** or **100GE or 1 to 4 channel 10GE/25GE with optional RSFEC and PTP** as the core variant.
- **10GE/25GE Channel(s)** as **Active channel(s) at startup** if you choose **100GE or 1 to 4 channel 10GE/25GE with optional RSFEC and PTP** as the core variant.
- **Enable RSFEC** to use RSFEC feature.
- **10G** or **25G** as the Ethernet rate.
- Select **PCS Only**, **OTN**, or **FlexE** as Ethernet IP layers.

  Note: RSFEC is not supported in 10GE variant.

Figure 11. Simulation Block Diagram for E-Tile Hard IP for Ethernet Intel FPGA 10GE/25GE PCS Only, OTN, or FlexE with Optional RSFEC Design Examples



The testbench sends traffic through the IP core, exercising the transmit side and receive side of the IP core.

The successful test run displays output confirming the following behavior:

1. Wait for PLL to lock.
2. Wait for RX transceiver reset to complete.
3. Wait for RX alignment.
4. Send three sets of packet.
5. Receive and verify the packets.
6. Displaying `Testbench complete.`

The following sample output illustrates a successful simulation test run for a 10GE, PCS Only IP core variation.

```
# Ref clock is 322.265625 MHz
# waiting for EHIP Ready....
# EHIP READY is 1 at time             425955000
# Waiting for RX Block Lock
# EHIP RX Block Lock  is high at time             429395673
# Waiting for RX alignment
# RX deskew locked
# RX lane aligmnent locked
# TX enabled
# *** Sending packets ***
# Start frame detected, byteslip 0, time 431948219
# ** RX checker has received packets correctly!
# ** RX checker is reset.
# *** Second attempt of sending packets ***
# Start frame detected, byteslip 0, time 437204752
# ** RX checker has received packets correctly!
# ** RX checker is reset.
# *** Third attempt of sending packets ***
# Start frame detected, byteslip 0, time 442467492
# ** RX checker has received packets correctly!
# ** PASSED
# **
# ***************************************
# ** Note: $finish    : ./basic_avl_tb_top.sv(246)
#    Time: 445329189 ps  Iteration: 0  Instance: /basic_avl_tb_top
# 1
# Break in Module basic_avl_tb_top at ./basic_avl_tb_top.sv line 246
```

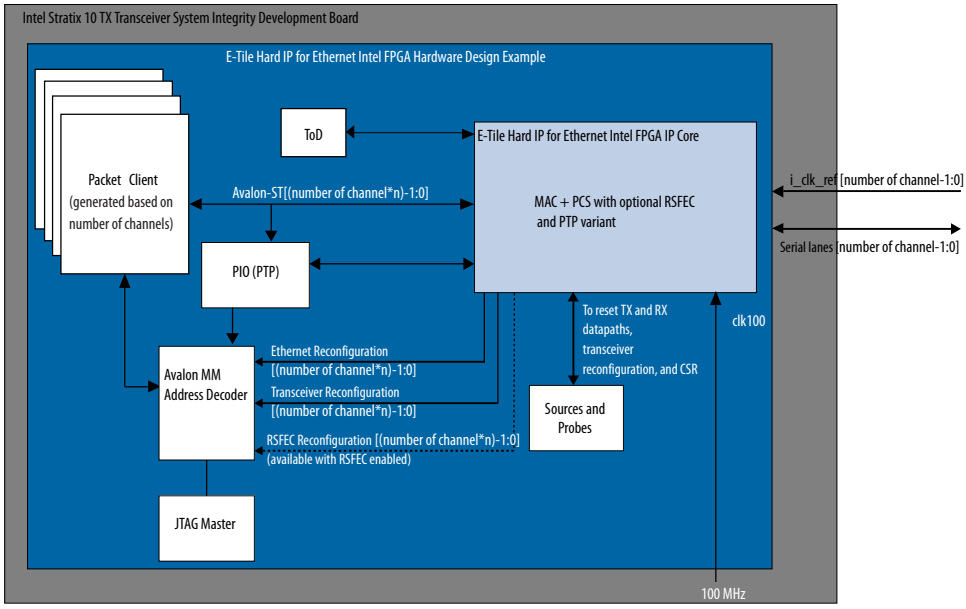
Related Information

Simulating the E-Tile Hard IP for Ethernet Intel FPGA Design Example Testbench on page 9

2.2. Hardware Design Examples

2.2.1. 10GE/25GE MAC + PCS with Optional RSFEC and PTP Hardware Design Example Components

Figure 12. 10GE/25GE MAC + PCS with Optional RSFEC and PTP Hardware Design Examples High Level Block Diagram



The E-Tile Hard IP for Ethernet Intel FPGA hardware design examples include the following components:

- E-Tile Hard IP for Ethernet Intel FPGA IP core.
- Client logic that coordinates the programming of the IP core and packet generation.
- Time-of-day (ToD) module to provide a continuous flow of current time-of-day information to the IP core.
- PIO block to store RX and TX PTP timestamp for accuracy calculation and to send PTP 2-step timestamp request.
- Avalon MM address decoder to decode reconfiguration address space for MAC, transceiver, and RSFEC modules during reconfiguration accesses.
- JTAG controller that communicates with the System Console. You communicate with the client logic through the System Console.

The following sample outputs illustrate a successful hardware test run for a 25GE, MAC+PCS, non-PTP IP core variation. The test results are located at `<design_example_dir>/hardware_test_design/hwtest_sl/c3_elane_xcvr_loopback_test.log` or `<design_example_dir>/hardware_test_design/hwtest_sl/c3_elane_traffic_basic_test.log`.

Result from `c3_elane_xcvr_loopback_test.log` file:

```
Info: Set JTAG Master Service Path

Info: Opened JTAG Master Service

    Test Start time is: 13:12:26
    Test Start date is: 11/12/2018    Successfully Write XCVR Channel 0, CSR
Register offset = 0x84, data = 0x1
    Successfully Write XCVR Channel 0, CSR Register offset = 0x85, data =
0x1
.
.
.

    Successfully Read  XCVR  Channel 0, CSR Register offset = 0x88, data =
0x8

    C3 ELANE XCVR Channel 0 Loopback mode is successfully enabled

    Successfully Write EHIPLANE Channel 0, User Register
phy_ehip_csr_soft_reset, offset = 0x310, data = 0x0
    Successfully Write EHIPLANE Channel 0, User Register
phy_ehip_csr_soft_reset, offset = 0x310, data = 0x1
.
.
.

    Successfully Read  EHIPLANE Channel 0, User Register
phy_ehip_csr_soft_reset, offset = 0x310, data = 0x0

    C3 ELANE Channel 0 System Reset is successfully

    Test End time is: 12:33:54
    Test End date is: 11/12/2018

Info: Closed JTAG Master Service


Info: Test <c3_elane_xcvr_loopback_test> Passed
```

Result from `c3_elane_traffic_basic_test_log` file:

```
Info: Set JTAG Master Service Path


Info: Opened JTAG Master Service

    Test Start time is: 13:12:26
    Test Start date is: 11/12/2018

Info: Read all ELANE CSR registers


    Successfully Read  EHIPLANE Channel 0, User Register phy_revid, offset =
0x300, data = 0x11112015
    Successfully Read  EHIPLANE Channel 0, User Register phy_scratch, offset
= 0x301, data = 0x0
.
.
.
    Successfully Read  EHIPLANE Channel 0, User Register
phy_ehip_csr_soft_reset   , offset = 0x310, data = 0x0

    C3 ELANE Channel 0 System Reset is successfully

    Successfully Write EHIPLANE Channel 0, Traffic GEN/CHK Register
pkt_end_addr_start_addr, offset = 0x8, data = 0x25800040
    Successfully Write EHIPLANE Channel 0, Traffic GEN/CHK Register
pkt_tx_num, offset = 0x9, data = 0xa
```

```
Info: Stopping the traffic generator

   Successfully Write EHIPLANE Channel 0, Traffic GEN/CHK Register
pkt_tx_ctrl, offset = 0x10, data = 0x87

Info: clearing the traffic generator statistics

    Successfully Write EHIPLANE Channel 0, Traffic GEN/CHK Register
pkt_clear_dropped_counter, offset = 0x7, data = 0x3
    Successfully Write EHIPLANE Channel 0, Traffic GEN/CHK Register
pkt_clear_dropped_counter, offset = 0x7, data = 0x0

Info: clearing the statistics

    Successfully Write EHIPLANE Channel 0, User Register cntr_tx_config,
offset = 0x845, data = 0x1
    Successfully Write EHIPLANE Channel 0, User Register cntr_rx_config,
offset = 0x945, data = 0x1

Info: Enabling the statistics

    Successfully Write EHIPLANE Channel 0, User Register cntr_tx_config,
offset = 0x845, data = 0x0
    Successfully Write EHIPLANE Channel 0, User Register cntr_rx_config,
offset = 0x945, data = 0x0

Info: Starting the traffic generator

    Successfully Write EHIPLANE Channel 0, Traffic GEN/CHK Register
pkt_tx_ctrl, offset = 0x10, data = 0x85
    Successfully Read  EHIPLANE Channel 0, User Register
cntr_tx_fragments_lo, offset = 0x800, data = 0x0

Info: Stopping the traffic generator

    Successfully Write EHIPLANE Channel 0, Traffic GEN/CHK Register
pkt_tx_ctrl, offset = 0x10, data = 0x87
    Successfully Read  EHIPLANE Channel 0, Traffic GEN/CHK Register
pkt_tx_ctrl, offset = 0x10, data = 0x87
.
.
.
    Successfully Read  EHIPLANE Channel 0, Traffic GEN/CHK Register
pkt_rx_pkt_cnt, offset = 0x5, data = 0x463ec5

Info: Channel 0 test is completed

    Successfully Read  RSFEC Register rsfec_top_rx_cfg, offset = 0x14, data =
0x1000
    Successfully Read  RSFEC Register arbiter_base_cfg, offset = 0x0, data =
0x1
.
.
.
    Successfully Read  RSFEC Register rsfec_top_tx_cfg, offset = 0x10, data =
0x1666

    Test End time is: 13:12:37
    Test End date is: 11/12/2018

Info: Closed JTAG Master Service


Info: Test <c3_elane_traffic_basic_test> Passed
```

The following sample output illustrate a successful hardware test run for a 25GE, MAC +PCS, with PTP IP core variation. The test result is located at *<design_example_dir>*/hardware_test_design/hwtest_sl/c3_elane_ptp_traffic_basic_test.log.

```
Info: Set JTAG Master Service Path


Info: Opened JTAG Master Service

    Test Start time is: 13:12:37
    Test Start date is: 11/12/2018

    Successfully Write EHIPLANE Channel 0, User Register
phy_ehip_csr_soft_reset, offset = 0x310, data = 0x0
    Successfully Write EHIPLANE Channel 0, User Register
phy_ehip_csr_soft_reset, offset = 0x310, data = 0x1
.
.
.
     Successfully Read  EHIPLANE Channel 0, User Register
phy_ehip_csr_soft_reset , offset = 0x310, data = 0x0

    C3 ELANE Channel 0 System Reset is successfully

Info: Stopping the traffic generator

    Successfully Write EHIPLANE Channel 0, Traffic GEN/CHK Register
pkt_tx_ctrl , offset = 0x10, data = 0x57

Info: clearing the traffic generator statistics

    Successfully Write EHIPLANE Channel 0, Traffic GEN/CHK Register
pkt_clear_dropped_counter  , offset = 0x7, data = 0x3
    Successfully Write EHIPLANE Channel 0, Traffic GEN/CHK Register
pkt_clear_dropped_counter  , offset = 0x7, data = 0x0

Info: clearing the statistics

    Successfully Write EHIPLANE Channel 0, User Register cntr_tx_config ,
offset = 0x845, data = 0x1
    Successfully Write EHIPLANE Channel 0, User Register cntr_rx_config ,
offset = 0x945, data = 0x1

Info: Enabling the statistics

    Successfully Write EHIPLANE Channel 0, User Register cntr_tx_config ,
offset = 0x845, data = 0x0
    Successfully Write EHIPLANE Channel 0, User Register cntr_rx_config ,
offset = 0x945, data = 0x0
.
.
.
     Successfully Read  EHIPLANE Channel 0, User Register
phy_ehip_csr_soft_reset , offset = 0x310, data = 0x0

    C3 ELANE Channel 0 System Reset is successfully

Info: Training PTP RX AIB deskew and waiting for PTP RX ready...

    Successfully Read  EHIPLANE Channel 0, PIO Register, offset = 0x0, data =
0x0
.
.
.
     Successfully Read  EHIPLANE Channel 0, PIO Register, offset = 0x0, data
= 0x7

Info: PTP RX AIB Deskew Done
```

```
Info: clearing the traffic generator statistics

    Successfully Write EHIPLANE Channel 0, Traffic GEN/CHK Register
pkt_clear_dropped_counter  , offset = 0x7, data = 0x3
     Successfully Write EHIPLANE Channel 0, Traffic GEN/CHK Register
pkt_clear_dropped_counter  , offset = 0x7, data = 0x0

Info: clearing the statistics

    Successfully Write EHIPLANE Channel 0, User Register cntr_tx_config,
offset = 0x845, data = 0x1
    Successfully Write EHIPLANE Channel 0, User Register cntr_rx_config,
offset = 0x945, data = 0x1

Info: Enabling the statistics

    Successfully Write EHIPLANE Channel 0, User Register cntr_tx_config,
offset = 0x845, data = 0x0
    Successfully Write EHIPLANE Channel 0, User Register cntr_rx_config ,
offset = 0x945, data = 0x0

Info: Accuracy measurement settings

     Successfully Read  RSFEC Register rsfec_cw_pos_rx_3 , offset = 0x1cc,
data = 0x48

Info: RX slip count = 0x8


Info: UI Value = 0x0009EE01


Info: TX Extra Latency = 0x53c8287


Info: RX Extra Latency = 0x595e090

    Successfully Write EHIPLANE Channel 0, User Register
tx_ptp_extra_latency, offset = 0xa0a, data = 0x53c82
.
.
.
     Successfully Read  EHIPLANE Channel 0, PIO Register, offset = 0xc, data
= 0x101

Info: Iteration = 1 : TX Timestamp = 00000000001001a75b48d452,  RX Timestamp
= 00000000001001a75b457470,  Accuracy Difference = 3.37454224 ns


     Successfully Write EHIPLANE Channel 0, PIO Register, offset = 0xc, data
= 0x0
    Successfully Write EHIPLANE Channel 0, Traffic GEN/CHK Register
pkt_tx_ctrl, offset = 0x10, data = 0x57
    Successfully Write EHIPLANE Channel 0, PIO Register, offset = 0xc, data =
0x102
    Successfully Write EHIPLANE Channel 0, Traffic GEN/CHK Register
pkt_tx_ctrl, offset = 0x10, data = 0x55
    Successfully Read  EHIPLANE Channel 0, User Register cntr_tx_64b_lo,
offset = 0x816, data = 0x2
    Successfully Read  EHIPLANE Channel 0, User Register cntr_rx_64b_lo,
offset = 0x916, data = 0x2
    Successfully Read  EHIPLANE Channel 0, PIO Register, offset = 0x4, data =
0x338872bb
    Successfully Read  EHIPLANE Channel 0, PIO Register, offset = 0x5, data =
0x10031d
    Successfully Read  EHIPLANE Channel 0, PIO Register, offset = 0x6, data =
0x0
    Successfully Read  EHIPLANE Channel 0, PIO Register, offset = 0x8, data =
0x338512d9
    Successfully Read  EHIPLANE Channel 0, PIO Register, offset = 0x9, data =
```

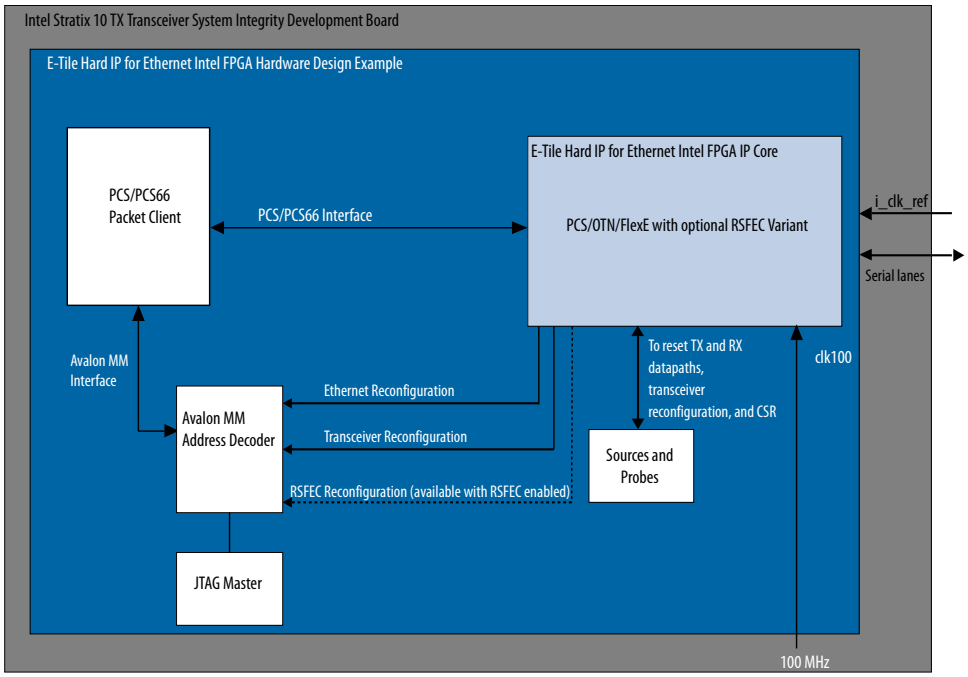
```
0x10031d
    Successfully Read  EHIPLANE Channel 0, PIO Register, offset = 0xa, data =
0x0
    Successfully Read  EHIPLANE Channel 0, PIO Register, offset = 0x7, data =
0x2
    Successfully Read  EHIPLANE Channel 0, PIO Register, offset = 0xc, data =
0x102
.
.
.

Info: Iteration = 1000 : TX Timestamp = 00000000003a1e3439531544,  RX
Timestamp = 00000000003a1e34394f65f2,  Accuracy Difference = 3.68484497 ns

Info: Stopping the traffic generator

    Successfully Write EHIPLANE Channel 0, PIO Register, offset = 0xc, data =
0x0
.
.
.
Successfully Read  EHIPLANE Channel 0, User Register cntr_rx_badlt_hi, offset
= 0x969, data = 0x0

    Test End time is: 13:13:22
    Test End date is: 11/12/2018

Info: Closed JTAG Master Service


Info: Test <c3_elane_ptp_traffic_basic_test> Passed
```

Related Information

- Intel Stratix 10 TX Signal Integrity Development Kit Webpage
- Compiling and Configuring the Design Example in Hardware on page 10
- Testing the E-Tile Hard IP for Ethernet Intel FPGA Hardware Design Example on page 11

2.2.2. 10GE/25GE PCS Only with Optional RSFEC Hardware Design Example Components

Figure 13. 10GE/25GE with Optional RSFEC and PTP Hardware Design Examples High Level Block Diagram



The E-Tile Hard IP for Ethernet Intel FPGA hardware design examples include the following components:

- E-Tile Hard IP for Ethernet Intel FPGA IP core.
- Client logic that coordinates the programming of the IP core and packet generation.
- JTAG controller that communicates with the System Console. You communicate with the client logic through the System Console.

Result from `c3_elane_traffic_basic_test_log` file:

Related Information

- Intel Stratix 10 TX Signal Integrity Development Kit Webpage
- Compiling and Configuring the Design Example in Hardware on page 10
- Testing the E-Tile Hard IP for Ethernet Intel FPGA Hardware Design Example on page 11

2.3. 10GE/25GE Design Example Interface Signals

The following signals are hardware design example signals for all 10GE/25GE variants.

Table 6. 10GE/25GE Hardware Design Example Interface Signals

Signal	Direction	Description
clk100	Input	Drive at 100 to 161.13 MHz. Input clock for CSR access on all the AVMM interfaces.
i_clk_ref	Input	Drive at 322.265625 MHz.
cpu_resetn	Input	Resets the IP core. Active low. Drives the global hard reset csr_reset_n to the IP core.
o_tx_serial[(number of channels-1:0]	Output	Transceiver PHY output serial data.
i_rx_serial[number of channels-1:0]	Input	Transceiver PHY input serial data.

Related Information

E-Tile Hard IP for Ethernet Intel FPGA Interfaces and Signal Descriptions

2.4. 10GE/25GE Design Examples Registers

Table 7. E-Tile Hard IP for Ethernet Intel FPGA Hardware Design Examples Register Map

Lists the memory mapped register ranges for all 10GE/25GE hardware design example variants. You access these registers with the reg_read and reg_write functions in the System Console.

Channel Number	Word Offset	Register Type
0	0x000000	KR4 registers
	0x000300	RX PCS registers
	0x000400	TX MAC registers
	0x000500	RX MAC registers
	0x000800	TX Statistics Counter registers
	0x000900	RX Statistics Counter registers
	0x001000	Packet Client registers
	0x002000	PTP monitoring registers
	0x010000	RSFEC configuration registers
	0x100000	Transceiver registers
1	0x200000	KR4 registers
	0x200300	RX PCS registers
	0x200400	TX MAC registers
	0x200500	RX MAC registers
	0x200800	TX Statistics Counter registers
	0x200900	RX Statistics Counter registers
	0x201000	Packet Client registers
	0x202000	PTP monitoring registers

continued...

Channel Number	Word Offset	Register Type
	0x210000	RSFEC configuration registers
	0x300000	Transceiver registers
2	0x400000	KR4 registers
	0x400300	RX PCS registers
	0x400400	TX MAC registers
	0x400500	RX MAC registers
	0x400800	TX Statistics Counter registers
	0x400900	RX Statistics Counter registers
	0x401000	Packet Client registers
	0x402000	PTP monitoring registers
	0x410000	RSFEC configuration registers
	0x500000	Transceiver registers
3	0x600000	KR4 registers
	0x600300	RX PCS registers
	0x600400	TX MAC registers
	0x600500	RX MAC registers
	0x600800	TX Statistics Counter registers
	0x600900	RX Statistics Counter registers
	0x601000	Packet Client registers
	0x602000	PTP monitoring registers
	0x610000	RSFEC configuration registers
	0x700000	Transceiver registers

Table 8. Packet Client Registers

You can customize the E-Tile Hard IP for Ethernet Intel FPGA hardware design example by programming the packet client registers.

Addr	Name	Bit	Description	HW Reset Value	Access
0x1000	PKT_CL_SCRATCH	[31:0]	Scratch register available for testing.		RW
0x1001	PKT_CL_CLNT	[31:0]	Four characters of IP block identification string "CLNT"		RO
0x1008	Packet Size Configure	[29:0]	Specify the transmit packet size in bytes. These bits have dependencies to PKT_GEN_TX_CTRL register. • Bit [29:16]: Specify the upper limit of the packet size in bytes. This is only applicable to incremental mode. • Bit [13:0]: — For fixed mode, these bits specify the transmit packet size in bytes. — For incremental mode, these bits specify the incremental bytes for a packet.	0x25800040	RW

continued...

Addr	Name	Bit	Description	HW Reset Value	Access
0x1009	Packet Number Control	[31:0]	Specify the number of packets to transmit from the packet generator.	0xA	RW
0x1010	PKT_GEN_TX_CTRL	[7:0]	• Bit [0]: Reserved. • Bit [1]: Packet generator disable bit. Set this bit to the value of 1 to turn off the packet generator, and reset it to the value of 0 to turn on the packet generator. • Bit [2]: Reserved. • Bit [3]: Has the value of 1 if the IP core is in MAC loopback mode; has the value of 0 if the packet client uses the packet generator. • Bit [5:4]: — 00: Random mode — 01: Fixed mode — 10: Incremental mode • Bit [6]: Set this bit to 1 to use 0x1009 register to turn off packet generator based on a fixed number of packets to transmit. Otherwise, bit [1] of PKT_GEN_TX_CTRL register is used to turn off the packet generator. • Bit [7]: — 1: For transmission without gap in between packets. — 0: For transmission with random gap in between packets.	0x6	RW
0x1011	Destination address lower 32 bits	[31:0]	Destination address (lower 32 bits)	0x56780ADD	RW
0x1012	Destination address upper 16 bits	[15:0]	Destination address (upper 16 bits)	0x1234	RW
0x1013	Source address lower 32bits	[31:0]	Source address (lower 32 bits)	0x43210ADD	RW
0x1014	Source address lower 16bits	[15:0]	Source address (upper 16 bits)	0x8765	RW
0x1016	PKT_CL_LOOPBACK_RESET	[0]	MAC loopback reset. Set to the value of 1 to reset the design example MAC loopback.	1'b0	RW

Related Information

E-Tile Hard IP for Ethernet Intel FPGA IP core register descriptions

3. 100GE with Optional RSFEC Design Example

The 100GE design example demonstrates an Ethernet solution for Intel Stratix 10 devices using the E-Tile Hard IP for Ethernet Intel FPGA IP core with the following variants:

Table 9. Supported Design Example Variants for 100GE

Variant	Design Example Support
MAC+PCS with Optional RSFEC (528,514)/(544,514) • For (528,514) RSFEC variant, the design example consists of 4 transceiver channels • For (544,514) RSFEC variant, the design example consists of 2 transceiver channels	Simulation, compilation-only project, and hardware design example
PCS Only with Optional RSFEC (528,514)/(544,514) • For (528,514) RSFEC variant, the design example consists of 4 transceiver channels • For (544,514) RSFEC variant, the design example consists of 2 transceiver channels	Simulation, compilation-only project, and hardware design example
OTN with Optional RSFEC (528,514)/(544,514) • For (528,514) RSFEC variant, the design example consists of 4 transceiver channels • For (544,514) RSFEC variant, the design example consists of 2 transceiver channels	Simulation and compilation-only project
FlexE with Optional RSFEC (528,514)/(544,514) • For (528,514) RSFEC variant, the design example consists of 4 transceiver channels • For (544,514) RSFEC variant, the design example consists of 2 transceiver channels	Simulation and compilation-only project

Note: The E-Tile Hard IP for Ethernet Intel FPGA IP provides preliminary support for the OTN feature. For further inquiries, contact your nearest Intel sales representative or file an Intel Premier Support (IPS) case on https://www.intel.com/content/www/us/en/programmable/my-intel/mal-home.html.

3.1. Simulation Design Examples

3.1.1. E-Tile Hard IP for Ethernet Intel FPGA 100GE MAC + PCS with Optional RSFEC Simulation Design Example

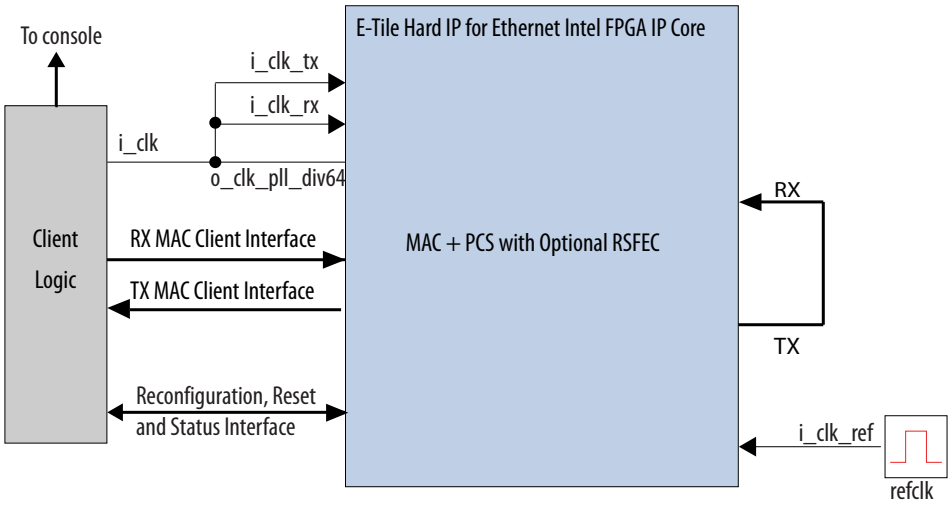
The simulation block diagram below is generated using the following settings:

1. **Single 100GE with optional RSFEC** or **100GE or 1 to 4 channel 10GE/25GE with optional RSFEC and PTP** as the core variant.
2. **100GE Channel** as **Active channel(s) at startup** if you choose **100GE or 1 to 4 channel 10GE/25GE with optional RSFEC and PTP** as the core variant.
3. **Enable RSFEC** to use RSFEC feature.

Note: The RSFEC feature is only available when you select **100GE or 1 to 4 channel 10GE/25GE with optional RSFEC and PTP** as the core variant.

4. **100G** as the Ethernet rate.
5. **MAC + PCS** as **Select Ethernet IP Layers** to use instantiate MAC and PCS layer or **MAC+PCS+(528,514)RSFEC/MAC+PCS+(544,514)RSFEC** to instantiate MAC and PCS with RSFEC feature.

Figure 14. Simulation Block Diagram for E-Tile Hard IP for Ethernet Intel FPGA 100GE MAC + PCS with Optional RSFEC Design Example



The testbench sends traffic through the IP core, exercising the transmit side and receive side of the IP core.

To speed up simulation, the IP core simulation model sends alignment marker tags at shorter intervals than required by the IEEE Ethernet standard. The standard specifies an alignment marker interval of 16384 words in each virtual lane. The simulation model with the testbench implements an alignment marker interval of 512 words.

The successful test run displays output confirming the following behavior:

1. The client logic resets the IP core.
2. Waits for RX datapath to align.
3. Once alignment is complete, client logic transmits a series of packets to the IP core.
4. The client logic receives the same series of packets through RX MAC interface.
5. The client logic then checks the number of packets received and verify that the data matches with the transmitted packets.
6. Displaying `Testbench complete.`

The following sample output illustrates a successful simulation test run for a 100GE, MAC+PCS with optional RSFEC IP core variation.

```
# o_tx_lanes_stable is 1 at time             345651500
# waiting for tx_dll_lock....
# TX DLL LOCK is 1 at time            398849563
# waiting for tx_transfer_ready....
# TX transfer ready is 1 at time             399169435
```

```
# waiting for rx_transfer_ready....
# RX transfer ready is 1 at time             410719813
# EHIP PLD Ready out is 1 at time             410776000
# EHIP reset out is 0 at time             411040000
# EHIP reset ack is 0 at time             412282101
# EHIP TX reset out is 0 at time             413160000
# EHIP TX reset ack is 0 at time             462643731
# waiting for EHIP Ready....
# EHIP READY is 1 at time             462750387
# EHIP RX reset out is 0 at time             463088000
# waiting for rx reset ack....
# EHIP RX reset ack is 0 at time             463283667
# Waiting for RX Block Lock
# EHIP RX Block Lock  is high at time             467376591
# Waiting for AM lock
# EHIP RX AM Lock  is high at time             468643131
# Waiting for RX alignment
# RX deskew locked
# RX lane aligmnent locked
# ** Sending Packet           1...
# ** Sending Packet           2...
# ** Sending Packet           3...
# ** Sending Packet           4...
# ** Sending Packet           5...
# ** Sending Packet           6...
# ** Sending Packet           7...
# ** Received Packet          1...
# ** Sending Packet           8...
# ** Received Packet          2...
# ** Sending Packet           9...
# ** Received Packet          3...
# ** Received Packet          4...
# ** Sending Packet          10...
# ** Received Packet          5...
# ** Received Packet          6...
# ** Received Packet          7...
# ** Received Packet          8...
# ** Received Packet          9...
# ** Received Packet         10...
# ====>MATCH!     ReaddataValid = 1 Readdata = 11112015 Expected_Readdata =
11112015
#
# ====> writedata = ffff0000
#
# ====>MATCH!     ReaddataValid = 1 Readdata = 11112015 Expected_Readdata =
11112015
#
# ====> writedata = 4321abcd
#
# ====>MATCH!     ReaddataValid = 1 Readdata = 4321abcd Expected_Readdata =
4321abcd
#
# ====> writedata = a5a51234
#
# ====>MATCH!     ReaddataValid = 1 Readdata = a5a51234 Expected_Readdata =
a5a51234
#
# ====> writedata = abcda5a5
#
# ====>MATCH!     ReaddataValid = 1 Readdata = abcda5a5 Expected_Readdata =
abcda5a5
#
# ====> writedata = 4321abcd
#
# ====>MATCH!     ReaddataValid = 1 Readdata = 4321abcd Expected_Readdata =
4321abcd
#
# ====> writedata = a5a51234
#
# ====>MATCH!     ReaddataValid = 1 Readdata = a5a51234 Expected_Readdata =
a5a51234
```

```
#
# ====> writedata = abcda5a5
#
# ====>MATCH!     ReaddataValid = 1 Readdata = abcda5a5 Expected_Readdata =
abcda5a5
#
# TX enabled
# **
# ** Testbench complete.
# **
# ****************************************
```

Related Information

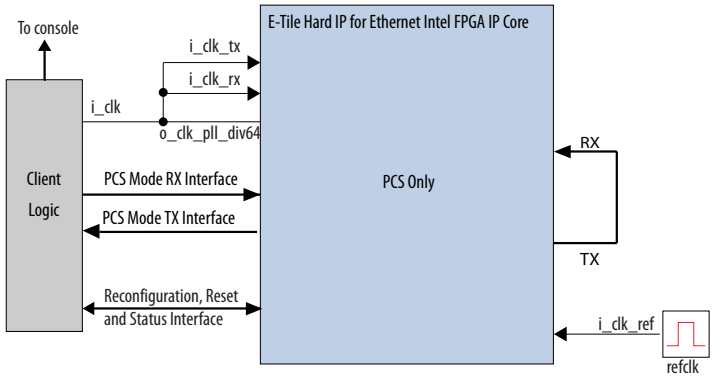
Simulating the E-Tile Hard IP for Ethernet Intel FPGA Design Example Testbench on page 9

3.1.2. E-Tile Hard IP for Ethernet Intel FPGA 100GE PCS Only with Optional RSFEC Simulation Design Example

The simulation block diagram below is generated using the following settings:

1. **Single 100GE with optional RSFEC** or **100GE or 1 to 4 channel 10GE/25GE with optional RSFEC and PTP** as the core variant.
2. **100GE Channel** as **Active channel(s) at startup** if you choose **100GE or 1 to 4 channel 10GE/25GE with optional RSFEC and PTP** as the core variant.
3. **100G** as the Ethernet rate.
4. **PCS_Only**, **PCS+(528,514)RSFEC**, or **PCS+(544,514)RSFEC** as the Ethernet IP layer.

Figure 15. E-Tile Hard IP for Ethernet Intel FPGA 100GE PCS Only Simulation Design Example Block Diagram



The testbench sends traffic through the IP core, exercising the transmit side and receive side of the IP core.

To speed up simulation, the IP core simulation model sends alignment marker tags at shorter intervals than required by the IEEE Ethernet standard. The standard specifies an alignment marker interval of 16384 words in each virtual lane. The simulation model with the testbench implements an alignment marker interval of 512 words.

The successful test run displays output confirming the following behavior:

1. The client logic resets the IP core.
2. Waits for RX datapath to align.
3. Once alignment is complete, client logic transmits a series of packets to the IP core through TX MII interface.
4. A counter drives `i_tx_mii_am` port with alignment marker insertion requests at the correct intervals.
5. The client logic receives the same series of packets through RX MII interface.
6. The client logic then checks the number of packets received.
7. Displaying `Testbench complete.`

The following sample output illustrates a successful simulation test run for a 100GE, PCS only IP core variation.

```
o_tx_lanes_stable is 1 at time             354775000
waiting for tx_dll_lock....
TX DLL LOCK is 1 at time             413726943
waiting for tx_transfer_ready....
TX transfer ready is 1 at time             414046815
waiting for rx_transfer_ready....
RX transfer ready is 1 at time             425122383
EHIP PLD Ready out is 1 at time             425184000
EHIP reset out is 0 at time             425320000
EHIP reset ack is 0 at time             426016853
EHIP TX reset out is 0 at time             426232000
EHIP TX reset ack is 0 at time             476830347
waiting for EHIP Ready....
EHIP READY is 1 at time             476910363
EHIP RX reset out is 0 at time             478680000
waiting for rx reset ack....
EHIP RX reset ack is 0 at time             478777403
Waiting for RX Block Lock
EHIP Rx Block Lock  is high at time             481444603
Waiting for AM lock
EHIP Rx am Lock  is high at time             482711523
Waiting for RX alignment
RX deskew locked
RX lane aligmnent locked
Sending Packets and Receiving Packets
====> writedata = 00000001

====>MATCH!     ReaddataValid = 1 Readdata = 00000053 Expected_Readdata =
00000053

**
** Testbench complete.
**
**************************************
```

Related Information

Simulating the E-Tile Hard IP for Ethernet Intel FPGA Design Example Testbench on page 9

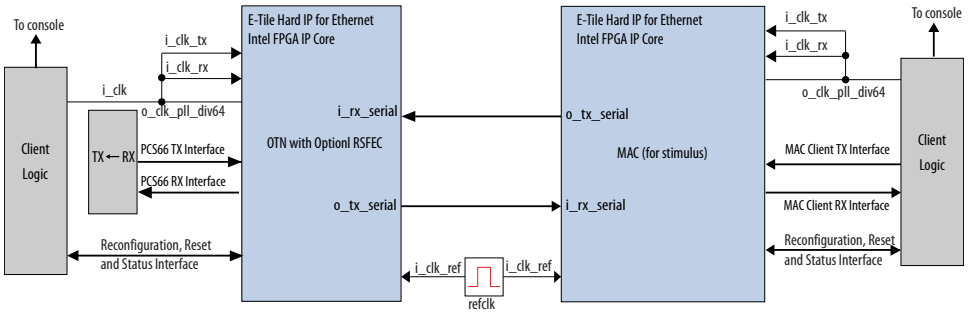
3.1.3. E-Tile Hard IP for Ethernet Intel FPGA 100GE OTN with Optional RSFEC Simulation Design Example

The simulation block diagram below is generated using the following settings:

1. **Single 100GE with optional RSFEC** or **100GE or 1 to 4 channel 10GE/25GE with optional RSFEC and PTP** as the core variant.
2. **100GE Channel** as **Active channel(s) at startup** if you choose **100GE or 1 to 4 channel 10GE/25GE with optional RSFEC and PTP** as the core variant.
3. **100G** as the Ethernet rate.
4. **OTN**, **OTN+(528,514)RSFEC**, or **OTN+(544,514)RSFEC** as the Ethernet IP layer.

Note: The E-Tile Hard IP for Ethernet Intel FPGA IP provides preliminary support for the OTN feature. For further inquiries, contact your nearest Intel sales representative or file an Intel Premier Support (IPS) case on https://www.intel.com/content/www/us/en/programmable/my-intel/mal-home.html.

Figure 16. E-Tile Hard IP for Ethernet Intel FPGA 100GE OTN Simulation Design Example Block Diagram



The testbench sends traffic through the IP core with OTN mode, exercising the transmit side and receive interface using a separate E-Tile Hard IP for Ethernet Intel FPGA MAC as a stimulus generator.

The successful test run displays output confirming the following behavior:

1. The client logic resets both the IP cores.
2. The stimulus client logic waits for the stimulus RX datapath and OTN RX datapath to align.
3. Once alignment is complete, the stimulus client logic transmits a series of packets to the OTN IP core.
4. The OTN IP core receives the series of packets and transmits back to the stimulus MAC IP core.
5. The stimulus client logic then checks the number of packets received and verify that the packets have no errors.
6. Displaying `Testbench complete.`

The following sample output illustrates a successful simulation test run for a 100GE OTN IP core variation.

```
# test_dut: def_100G_o_tx_lanes_stable is 1 at time             345685000
# test_dut: waiting for tx_dll_lock....
# dut: o_tx_lanes_stable is 1 at time             345685000
# dut: waiting for tx_dll_lock....
# dut: TX DLL LOCK is 1 at time             398849563
# dut: waiting for tx_transfer_ready....
```

```
# dut: TX transfer ready is 1 at time             399169435
# dut: waiting for rx_transfer_ready....
# dut: RX transfer ready is 1 at time             410719813
# dut: EHIP PLD Ready out is 1 at time             410776000
# dut: EHIP reset out is 0 at time            411040000
# dut: EHIP reset ack is 0 at time            412282101
# dut: EHIP TX reset out is 0 at time             413160000
# dut: EHIP TX reset ack is 0 at time             462643731
# dut: waiting for EHIP Ready....
# dut: EHIP READY is 1 at time            462750387
# dut: EHIP RX reset out is 0 at time             463088000
# dut: waiting for rx reset ack....
# dut: EHIP RX reset ack is 0 at time             463283667
# dut: Waiting for RX Block Lock
# test_dut: TX DLL LOCK is 1 at time            475452243
# test_dut: waiting for tx_transfer_ready....
# test_dut: TX transfer ready is 1 at time             475772115
# test_dut: waiting for rx_transfer_ready....
# test_dut: RX transfer ready is 1 at time             487164223
# test_dut: EHIP PLD Ready out is 1 at time             487224000
# test_dut: EHIP reset out is 0 at time           487488000
# test_dut: EHIP reset ack is 0 at time           488907771
# test_dut: EHIP TX reset out is 0 at time             489784000
# test_dut: EHIP TX reset ack is 0 at time             539116083
# test_dut: waiting for EHIP Ready....
# test_dut: EHIP READY is 1 at time            539169411
# test_dut: EHIP RX reset out is 0 at time             539512000
# test_dut: waiting for rx reset ack....
# test_dut: EHIP RX reset ack is 0 at time             539702691
# test_dut: Waiting for RX Block Lock
# dut: EHIP RX Block Lock  is high at time             542102451
# dut: Waiting for AM lock
# test_dut: EHIP RX Block Lock  is high at time             542735721
# test_dut: Waiting for AM lock
# dut: EHIP RX AM Lock  is high at time           543368991
# dut: Waiting for RX alignment
# dut: RX deskew locked
# dut: RX lane aligmnent locked
# dut: ************************************
# test_dut: EHIP RX AM Lock  is high at time             549068421
# test_dut: Waiting for RX alignment
# test_dut: RX deskew locked
# test_dut: RX lane aligmnent locked
# test_dut: ** Sending Packet           1...

.
.
.
# test_dut: ** Sending Packet           9...
# test_dut: ** Sending Packet          10...
# test_dut: ** Received Packet          1...

.
.
.
# test_dut: ** Received Packet          9...
# test_dut: ** Received Packet         10...
# test_dut: **
# test_dut: ** Testbench complete.
# test_dut: **
# test_dut: ****************************************
```

Related Information

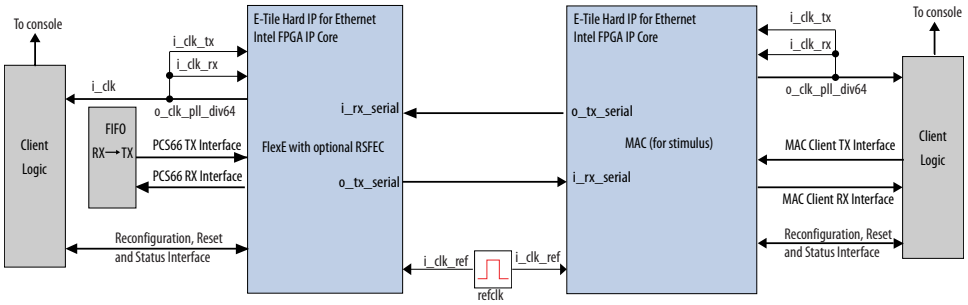
Simulating the E-Tile Hard IP for Ethernet Intel FPGA Design Example Testbench on page 9

3.1.4. E-Tile Hard IP for Ethernet Intel FPGA 100GE FlexE with Optional RSFEC Simulation Design Example

The simulation block diagram below is generated using the following settings:

1. **Single 100GE with optional RSFEC** or **100GE or 1 to 4 channel 10GE/25GE with optional RSFEC and PTP** as the core variant.
2. **100GE Channel** as **Active channel(s) at startup** if you choose **100GE or 1 to 4 channel 10GE/25GE with optional RSFEC and PTP** as the core variant.
3. **100G** as the Ethernet rate.
4. **FlexE**, **FlexE+(528,514)RSFEC**, or **FlexE+(544,514)RSFEC** as the Ethernet IP layer.

Figure 17. E-Tile Hard IP for Ethernet Intel FPGA 100GE FlexE Simulation Design Example Block Diagram



The testbench sends traffic through the IP core, exercising the transmit side and receive side of the IP core.

The successful test run displays output confirming the following behavior:

1. The client logic resets both the IP cores.
2. The stimulus client logic waits for the stimulus RX datapath and FlexE RX datapath to align.
3. Once alignment is complete, the stimulus client logic transmits a series of packets to the FlexE IP core.
4. The FlexE IP core receives the series of packets and transmits back to the stimulus MAC IP core.
5. The stimulus client logic then checks the number of packets received and verify that the packets have no errors.
6. Displaying `Testbench complete.`

The following sample output illustrates a successful simulation test run for a 100GE, FlexE only IP core variation.

```
# test_dut: def_100G_o_tx_lanes_stable is 1 at time            345685000
# test_dut: waiting for tx_dll_lock....
# dut: o_tx_lanes_stable is 1 at time            345685000
# dut: waiting for tx_dll_lock....
# dut: TX DLL LOCK is 1 at time            398849563
# dut: waiting for tx_transfer_ready....
# dut: TX transfer ready is 1 at time            399169435
# dut: waiting for rx_transfer_ready....
```

```
# dut: RX transfer ready is 1 at time             410719813
# dut: EHIP PLD Ready out is 1 at time             410776000
# dut: EHIP reset out is 0 at time            411040000
# dut: EHIP reset ack is 0 at time            412282101
# dut: EHIP TX reset out is 0 at time            413160000
# dut: EHIP TX reset ack is 0 at time            462643731
# dut: waiting for EHIP Ready....
# dut: EHIP READY is 1 at time            462750387
# dut: EHIP RX reset out is 0 at time            463088000
# dut: waiting for rx reset ack....
# dut: EHIP RX reset ack is 0 at time            463283667
# dut: Waiting for RX Block Lock
# test_dut: TX DLL LOCK is 1 at time            475452243
# test_dut: waiting for tx_transfer_ready....
# test_dut: TX transfer ready is 1 at time            475772115
# test_dut: waiting for rx_transfer_ready....
# test_dut: RX transfer ready is 1 at time            487164223
# test_dut: EHIP PLD Ready out is 1 at time             487224000
# test_dut: EHIP reset out is 0 at time           487488000
# test_dut: EHIP reset ack is 0 at time           488907771
# test_dut: EHIP TX reset out is 0 at time            489784000
# test_dut: EHIP TX reset ack is 0 at time            539116083
# test_dut: waiting for EHIP Ready....
# test_dut: EHIP READY is 1 at time            539169411
# test_dut: EHIP RX reset out is 0 at time            539512000
# test_dut: waiting for rx reset ack....
# test_dut: EHIP RX reset ack is 0 at time            539702691
# test_dut: Waiting for RX Block Lock
# dut: EHIP RX Block Lock  is high at time            542102451
# dut: Waiting for AM lock
# dut: EHIP RX AM Lock  is high at time           543368991
# dut: Waiting for RX alignment
# dut: RX deskew locked
# dut: RX lane aligmnent locked
# dut: ****************************************
# test_dut: EHIP RX Block Lock  is high at time            546535341
# test_dut: Waiting for AM lock
# test_dut: EHIP RX AM Lock  is high at time           547801881
# test_dut: Waiting for RX alignment
# test_dut: RX deskew locked
# test_dut: RX lane aligmnent locked
# test_dut: ** Sending Packet          1...
.
.
.
# test_dut: ** Sending Packet          9...
# test_dut: ** Sending Packet         10...
# test_dut: ** Received Packet          1...
.
.
.
# test_dut: ** Received Packet          9...
# test_dut: ** Received Packet         10...
# test_dut: **
# test_dut: ** Testbench complete.
# test_dut: **
# test_dut: ****************************************
```

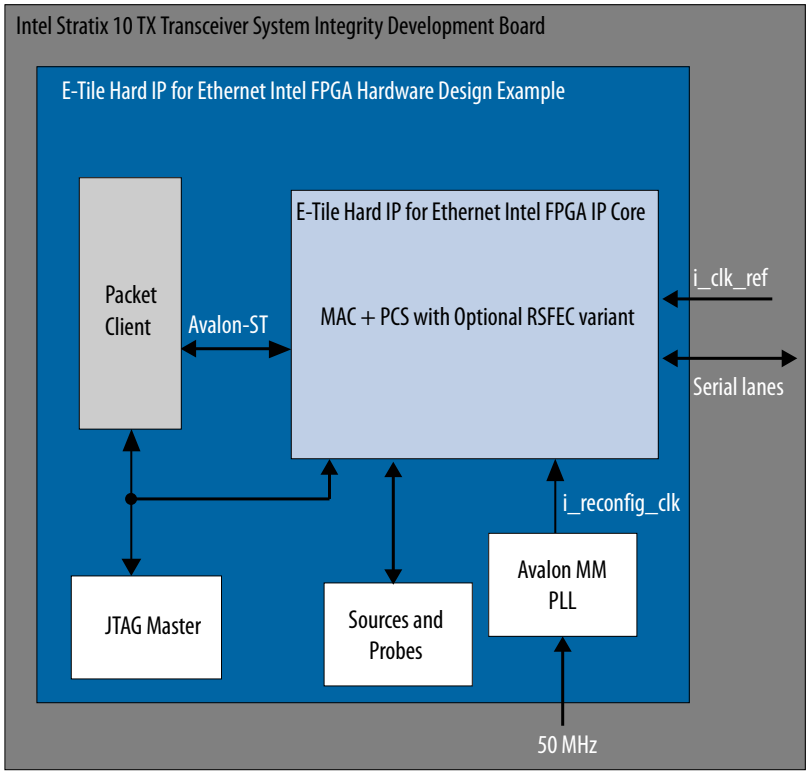
Related Information

Simulating the E-Tile Hard IP for Ethernet Intel FPGA Design Example Testbench on page 9

3.2. Hardware Design Examples

3.2.1. 100GE MAC + PCS with Optional RSFEC and PMA Calibration Hardware Design Example Components

Figure 18. 100GE MAC + PCS with Optional RSFEC Hardware Design Example High Level Block Diagram



The E-Tile Hard IP for Ethernet Intel FPGA hardware design example includes the following components:

- E-Tile Hard IP for Ethernet Intel FPGA IP core. The IP core consists of 4 channels if you select (528,514) RSFEC option, and 2 transceiver channels if you select (544,514) RSFEC option.
- Client logic that coordinates the programming of the IP core and packet generation.
- IOPLL to generate a 100 MHz clock from a 50 MHz input clock to the hardware design example.
- JTAG controller that communicates with the System Console. You communicate with the client logic through the System Console.

The hardware design example uses run_test command to initiate packet transmission from packet generator to the IP core. By default, the internal serial loopback is disabled in this design example. Use the `loop_on` command to enable the internal serial loopback. When you use the `run_test` or the `run_test_pam4` commands to run the hardware test in the design examples, the script enables internal loopback.

When the internal serial loopback is enabled, the IP core receives the packets and transmit to the packet generator. The client logic reads and print out the MAC statistic registers when the packet transmissions are complete.

The following sample outputs illustrate a successful hardware test run for 100GE, MAC +PCS with (528,514) RSFEC variation:

```
% run_test
--- Turning off packet generation ----
---------------------------------------
--------- Enabling loopback ----------
---------------------------------------
--- Wait for RX clock to settle... ---
---------------------------------------
-------- Printing PHY status ---------
---------------------------------------
 RX PHY Register Access: Checking Clock Frequencies (KHz)
    REFCLK         :0 (KHZ)
    TXCLK         :40285  (KHZ)
    RXCLK         :40284  (KHZ)
    TXRSCLK     :0  (KHZ)
    RXRSCLK     :0  (KHZ)
 RX PHY Status Polling
 Rx Frequency Lock Status     0x0000000f
 Mac Clock in OK Condition?   0x00000001
 Rx Frame Error               0x00000000
 Rx PHY Fully Aligned?        0x00000001
 Rx AM LOCK Condition?        0x00000001
 Rx Lanes Deskewed Condition? 0x00000001
---- Clearing MAC stats counters -----
---------------------------------------
--------- Sending packets... ---------
---------------------------------------
----- Reading MAC stats counters -----
---------------------------------------

================================================================================
============
                         STATISTICS FOR BASE 0x000900
(Rx)

================================================================================
============
Fragmented Frames                : 0
Jabbered Frames                  : 0
Any Size with FCS Err Frame      : 0
Right Size with FCS Err Fra      : 0
Multicast data  Err Frames       : 0
Broadcast data Err  Frames       : 0
Unicast data Err  Frames         : 0
Multicast control  Err Frame     : 0
Broadcast control Err  Frame     : 0
Unicast control Err  Frames      : 0
Pause control Err  Frames        : 0
64 Byte Frames                   : 7190
65 - 127 Byte Frames             : 6965
128 - 255 Byte Frames            : 14338
256 - 511 Byte Frames            : 28779
512 - 1023 Byte Frames           : 57548
1024 - 1518 Byte Frames          : 55880
1519 - MAX Byte Frames           : 0
> MAX Byte Frames               : 1669560
Rx Frame Starts                  : 1840260
Multicast data  OK  Frame        : 0
Broadcast data OK   Frame        : 0
Unicast data OK   Frames         : 1836399
Multicast Control Frames         : 0
Broadcast Control Frames         : 0
Unicast Control Frames           : 0
```

```
Pause Control Frames             : 0

================================================================================
============
                        STATISTICS FOR BASE 0x000800
(Tx)

================================================================================
============
Fragmented Frames                : 0
Jabbered Frames                  : 0
Any Size with FCS Err Frame      : 0
Right Size with FCS Err Fra      : 0
Multicast data  Err Frames       : 0
Broadcast data Err  Frames       : 0
Unicast data Err  Frames         : 0
Multicast control  Err Frame     : 0
Broadcast control Err  Frame     : 0
Unicast control Err  Frames      : 0
Pause control Err  Frames        : 0
64 Byte Frames                   : 7190
65 - 127 Byte Frames             : 6965
128 - 255 Byte Frames            : 14338
256 - 511 Byte Frames            : 28779
512 - 1023 Byte Frames           : 57548
1024 - 1518 Byte Frames          : 55880
1519 - MAX Byte Frames           : 0
> MAX Byte Frames                : 1669560
Tx Frame Starts                  : 1840260
Multicast data  OK  Frame        : 0
Broadcast data OK   Frame        : 0
Unicast data OK   Frames         : 1836399
Multicast Control Frames         : 0
Broadcast Control Frames         : 0
Unicast Control Frames           : 0
Pause Control Frames             : 0
```

The following sample outputs illustrate a successful hardware test run for 100GE, MAC +PCS with (544,512) RSFEC variation:

```
% run_test_pam4
--- Turning off packet generation ----
---------------------------------------
--------- Enabling loopback ----------
---------------------------------------
--- Performing PMA adaptation... ---
---------------------------------------
------------ Starting PMA Adaptation ------------
------- Checking PMA Adaptation Status-------
------- PMA Adaptation Done for ch0x0 -------
------- PMA Adaptation Done for ch0x2 -------
------------ Applying TX and RX Reset ----------
 wait for phy lock=50, locked=1
--Iteration:0 - PMA Adaptaion is Successful--
--- Wait for RX clock to settle... ---
---------------------------------------
-------- Printing PHY status ---------
---------------------------------------
 RX PHY Register Access: Checking Clock Frequencies (KHz)
    REFCLK         :0 (KHZ)
    TXCLK         :41504  (KHZ)
    RXCLK         :41505  (KHZ)
    TXRSCLK     :0  (KHZ)
    RXRSCLK     :0  (KHZ)
 RX PHY Status Polling
 Rx Frequency Lock Status     0x0000000f
 Mac Clock in OK Condition?   0x00000001
 Rx Frame Error               0x00000000
 Rx AM LOCK Condition?        0x00000001
 Rx Lanes Deskewed Condition? 0x00000001
```

```
---- Clearing MAC stats counters -----
--------------------------------------
--------- Sending packets... ---------
--------------------------------------
----- Reading MAC stats counters -----
--------------------------------------

================================================================================
============
                          STATISTICS FOR BASE 0x000900
(Rx)

================================================================================
============
Fragmented Frames               : 0
Jabbered Frames                 : 0
Any Size with FCS Err Frame     : 0
Right Size with FCS Err Fra     : 0
Multicast data  Err Frames      : 0
Broadcast data Err  Frames      : 0
Unicast data Err  Frames        : 0
Multicast control  Err Frame    : 0
Broadcast control Err  Frame    : 0
Unicast control Err  Frames     : 0
Pause control Err  Frames       : 0
64 Byte Frames                  : 7114
65 - 127 Byte Frames            : 6925
128 - 255 Byte Frames           : 14418
256 - 511 Byte Frames           : 28563
512 - 1023 Byte Frames          : 57313
1024 - 1518 Byte Frames         : 56067
1519 - MAX Byte Frames          : 0
> MAX Byte Frames               : 1670068
Rx Frame Starts                 : 1840468
Multicast data  OK  Frame       : 0
Broadcast data OK   Frame       : 0
Unicast data OK   Frames        : 1836559
Multicast Control Frames        : 0
Broadcast Control Frames        : 0
Unicast Control Frames          : 0
Pause Control Frames            : 0

================================================================================
============
                          STATISTICS FOR BASE 0x000800
(Tx)

================================================================================
============
Fragmented Frames               : 0
Jabbered Frames                 : 0
Any Size with FCS Err Frame     : 0
Right Size with FCS Err Fra     : 0
Multicast data  Err Frames      : 0
Broadcast data Err  Frames      : 0
Unicast data Err  Frames        : 0
Multicast control  Err Frame    : 0
Broadcast control Err  Frame    : 0
Unicast control Err  Frames     : 0
Pause control Err  Frames       : 0
64 Byte Frames                  : 7114
65 - 127 Byte Frames            : 6925
128 - 255 Byte Frames           : 14418
256 - 511 Byte Frames           : 28563
512 - 1023 Byte Frames          : 57313
1024 - 1518 Byte Frames         : 56067
1519 - MAX Byte Frames          : 0
> MAX Byte Frames               : 1670068
Tx Frame Starts                 : 1840468
Multicast data  OK  Frame       : 0
Broadcast data OK   Frame       : 0
```

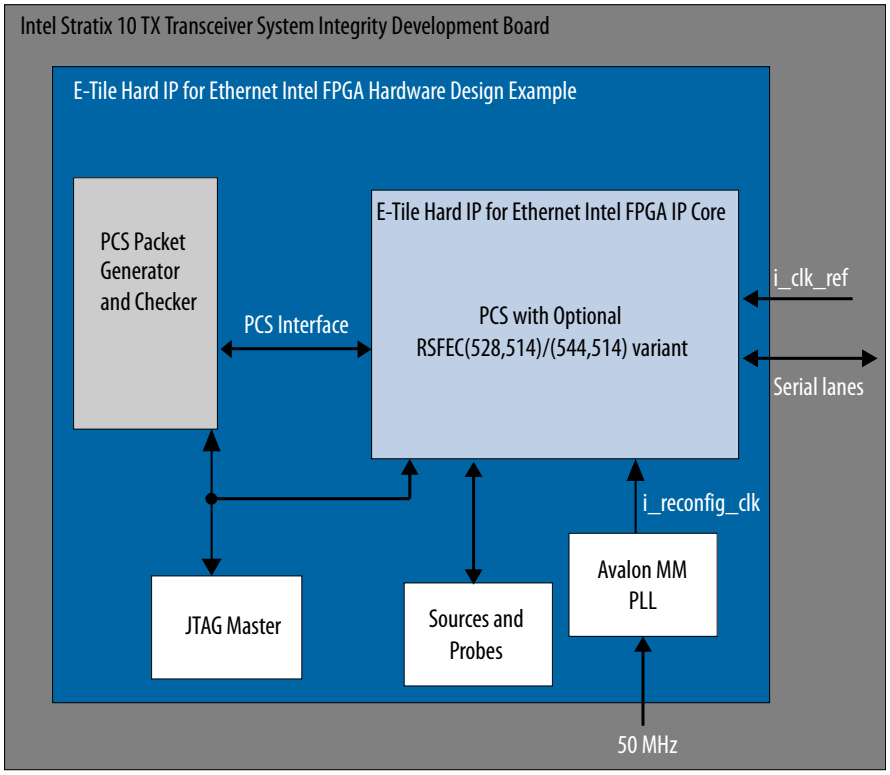
```
Unicast data OK   Frames          : 1836559
Multicast Control Frames          : 0
Broadcast Control Frames          : 0
Unicast Control Frames            : 0
Pause Control Frames              : 0
```

Related Information

- Intel Stratix 10 TX Signal Integrity Development Kit Webpage
- Compiling and Configuring the Design Example in Hardware on page 10
- Testing the E-Tile Hard IP for Ethernet Intel FPGA Hardware Design Example on page 11

3.2.2. 100GE PCS with Optional RSFEC Hardware Design Example Components

Figure 19. 100GE MAC + PCS with Optional RSFEC Hardware Design Example High Level Block Diagram



The E-Tile Hard IP for Ethernet Intel FPGA hardware design example includes the following components:

- E-Tile Hard IP for Ethernet Intel FPGA IP core.
- PCS packet generator and checker that coordinates the programming of the IP core, packet generation, and verify the packets.
- IOPLL to generate a 100 MHz clock from a 50 MHz input clock to the hardware design example.
- JTAG controller that communicates with the System Console. You communicate with the client logic through the System Console.

The hardware design example test initiates media-independent interface (MII) packet transmission from packet generator to the IP core. The packet generator supports incremental packet mode, fixed-size packet mode, and random packet content mode. Once reset is completed, the packet generator generates the number of packets requested to the IP core. The IP core transfers the packets through internal PMA loopback to the packet generator and checker for verification. This test only works with internal PMA loopback mode.

The following sample outputs illustrate a successful hardware test run for 100GE, PCS only with (528,514) RSFEC variation:

```
% pcs_only_traffic_test
Running pcs_only_traffic_test test
 RX PHY Register Access: Checking Clock Frequencies (KHz)

    REFCLK         :2 (KHZ)
    TXCLK         :40284  (KHZ)
    RXCLK         :40284  (KHZ)
    TXRSCLK     :0  (KHZ)
    RXRSCLK     :0  (KHZ)
 RX PHY Status Polling
 Rx Frequency Lock Status     0x0000000f
 Mac Clock in OK Condition?   0x00000001
 Rx Frame Error               0x00000000
Rx PHY Fully Aligned?        0x00000001
 Rx AM LOCK Condition?        0x00000001
 Rx Lanes Deskewed Condition? 0x00000001
Setting Number of frames to 6767
Setting Size of frames to 8588
Setting Size of frames to constant
-------------------------------------
PCS TRAFFIC = 0
pcs_only_traffic_test:pass
0
```

The following sample outputs illustrate a successful hardware test run for 100GE, PCS only with (544,512) RSFEC variations:

```
% % pcs_only_traffic_test_pam4
Running pcs_only_traffic_test_pam4 test
RX PHY Register Access: Checking Clock Frequencies (KHz)
    REFCLK         :1 (KHZ)
    TXCLK         :41504  (KHZ)
    RXCLK         :41505  (KHZ)
    TXRSCLK     :0  (KHZ)
    RXRSCLK     :0  (KHZ)
 RX PHY Status Polling
 Rx Frequency Lock Status     0x0000000f
 Mac Clock in OK Condition?   0x00000001
 Rx Frame Error               0x00000000
 Rx AM LOCK Condition?        0x00000001
 Rx Lanes Deskewed Condition? 0x00000001
-------------------------------------
PCS TRAFFIC = 0
Setting Number of frames to 5340
```

```
Setting Size of frames to 635
Setting Size of frames to random
pcs_only_traffic_test_pam4:pass
```

Related Information

- Intel Stratix 10 TX Signal Integrity Development Kit Webpage
- Compiling and Configuring the Design Example in Hardware on page 10
- Testing the E-Tile Hard IP for Ethernet Intel FPGA Hardware Design Example on page 11

3.3. 100GE MAC + PCS with Optional RSFEC Design Example Interface Signals

The E-Tile Hard IP for Ethernet Intel FPGA testbench is self-contained and does not require you to drive any input signals.

Table 10. 100GE MAC + PCS with Optional RSFEC Hardware Design Example Interface Signals

Signal	Direction	Description
clk50	Input	Drive at 50 MHz. The intent is to drive this from a 50 Mhz oscillator on the board.
i_clk_ref	Input	Drive at 156.25 MHz.
cpu_resetn	Input	Resets the IP core. Active low. Drives the global hard reset csr_reset_n to the IP core.
i_rx_serial[3:0]	Input	Transceiver PHY input serial data.
o_tx_serial[3:0]	Output	Transceiver PHY output serial data.
user_led[3:0]	Output	Status signals. Currently the design example drives all of these signals to a constant value of 0. The hardware design example connects these bits to drive LEDs on the target board.

Related Information

E-Tile Hard IP for Ethernet Intel FPGA Interfaces and Signal Descriptions

3.4. 100GE PCS with Optional RSFEC Design Example Interface Signals

The E-Tile Hard IP for Ethernet Intel FPGA testbench is self-contained and does not require you to drive any input signals.

Table 11. 100GE PCS with Optional RSFEC Hardware Design Example Interface Signals

Signal	Direction	Description
clk50	Input	Drive at 50 MHz. The intent is to drive this from a 50 Mhz oscillator on the board.
i_clk_ref	Input	Drive at 156.25 MHz.
cpu_resetn	Input	Resets the IP core. Active low. Drives the global hard reset csr_reset_n to the IP core.
i_rx_serial[3:0]	Input	Transceiver PHY input serial data.
o_tx_serial[3:0]	Output	Transceiver PHY output serial data.
user_led[3:0]	Output	Status signals. Currently the design example drives all of these signals to a constant value of 0. The hardware design example connects these bits to drive LEDs on the target board.

Related Information

E-Tile Hard IP for Ethernet Intel FPGA Interfaces and Signal Descriptions

3.5. 100GE MAC+PCS with Optional RSFEC Design Example Registers

Table 12. E-Tile Hard IP for Ethernet Intel FPGA Hardware Design Example Register Map

Lists the memory mapped register ranges for the hardware design example. You access these registers with the reg_read and reg_write functions in the System Console.

Word Offset	Register Type
0x000000	KR4 registers
0x000300	RX PCS registers
0x000400	TX MAC registers
0x000500	RX MAC registers
0x000800	TX Statistics Counter registers
0x000900	RX Statistics Counter registers
0x001000	Packet Client registers
0x002000	Packet monitoring registers
0x010000	RSFEC configuration registers
0x100000	Transceiver registers

Table 13. Packet Client Registers

You can customize the E-Tile Hard IP for Ethernet Intel FPGA hardware design example by programming the packet client registers.

Addr	Name	Bit	Description	HW Reset Value	Access
0x1000	PKT_CL_SCRATCH	[31:0]	Scratch register available for testing.		RW
0x1001	PKT_CL_CLNT	[31:0]	Four characters of IP block identification string "CLNT"		RO
0x1008	Packet Size Configure	[29:0]	Specify the transmit packet size in bytes. These bits have dependencies to PKT_GEN_TX_CTRL register. • Bit [29:16]: Specify the upper limit of the packet size in bytes. This is only applicable to incremental mode. • Bit [13:0]: — For fixed mode, these bits specify the transmit packet size in bytes. — For incremental mode, these bits specify the incremental bytes for a packet.	0x25800040	RW
0x1009	Packet Number Control	[31:0]	Specify the number of packets to transmit from the packet generator.	0xA	RW
0x1010	PKT_GEN_TX_CTRL	[7:0]	• Bit [0]: Reserved. • Bit [1]: Packet generator disable bit. Set this bit to the value of 1 to turn off the packet generator, and reset it to the value of 0 to turn on the packet generator. • Bit [2]: Reserved. • Bit [3]: Has the value of 1 if the IP core is in MAC loopback mode; has the value of 0 if the packet client uses the packet generator. • Bit [5:4]: — 00: Random mode — 01: Fixed mode — 10: Incremental mode • Bit [6]: Set this bit to 1 to use 0x1009 register to turn off packet generator based on a fixed number of packets to transmit. Otherwise, bit [1] of PKT_GEN_TX_CTRL register is used to turn off the packet generator. • Bit [7]: — 1: For transmission without gap in between packets. — 0: For transmission with random gap in between packets.	0x6	RW
0x1011	Destination address lower 32 bits	[31:0]	Destination address (lower 32 bits)	0x56780ADD	RW
0x1012	Destination address upper 16 bits	[15:0]	Destination address (upper 16 bits)	0x1234	RW

continued...

Addr	Name	Bit	Description	HW Reset Value	Access
0x1013	Source address lower 32bits	[31:0]	Source address (lower 32 bits)	0x43210ADD	RW
0x1014	Source address lower 16bits	[15:0]	Source address (upper 16 bits)	0x8765	RW
0x1016	PKT_CL_LOOPBACK_RESET	[0]	MAC loopback reset. Set to the value of 1 to reset the design example MAC loopback.	1'b0	RW

Related Information

E-Tile Hard IP for Ethernet Intel FPGA IP core register descriptions

3.6. 100GE PCS with Optional RSFEC Design Example Registers

Table 14. E-Tile Hard IP for Ethernet Intel FPGA Hardware Design Example Register Map

Lists the memory mapped register ranges for the hardware design example. You access these registers with the `reg_read` and `reg_write` functions in the System Console.

Word Offset	Register Type
0x000000	KR4 registers
0x000300	RX PCS registers
0x00F000	Packet Generator and Checker registers
0x010000	RSFEC configuration registers
0x100000	Transceiver registers

Table 15. Packet Generator and Checker Registers

You can customize the E-Tile Hard IP for Ethernet Intel FPGA hardware design example by programming the packet client registers.

Addr	Name	Bit	Description	HW Reset Value	Access
0xF000		[0]	Write 1 to start transmitting PCS packets.	0x0	RWC
0xF001	Control register	[0]	Write 1 to reset the channel.	0x0	RW
0xF002	XGMII Status register	[6:0]	• Bit [0]: value 1 indicates the RX path is ready to receive packet • Bit [1]: Value 1 indicates the packets are verified and passed. • Bit [2]: Value 1 indicates there is an error with the received packets. • Bit [3]: Value 1 indicates the FIFO is full. • Bit [4]: Value 1 indicates the test is completed. • Bit [5]: Value 1 indicates all frames completed transmission and reception. • Bit [6]: value 1 indicates the test has passed. .	0x0	RO

continued...

Addr	Name	Bit	Description	HW Reset Value	Access
0xF003	GMII Status register	[5:0]	• Bit [0]: value 1 indicates the GMII RX path is ready to receive packet • Bit [1]: Value 1 indicates the auto-negotiation completed. • Bit [2]: Value 1 indicates packet generation completed. • Bit [3]: Value 1 indicates packet verification completed. • Bit [4]: Value 1 indicates is an error with the received packets. • Bit [5]: value 1 indicates the test has passed.	0x0	RO
0xF006	max_frame register	[31:0]	Specify the maximum number of frames for transmission.	0x0	RW
0xF007	frame_length register	[31:0]	Specify the packet size.	0x0	RW
0xF008	XGMII_data_match_count	[255:0]	Report the number of XGMII passed packets.	0x0	RO
0xF009	XGMII_data_mismatch_count	[255:0]	Reports the number of XGMII error packets.	0x0	RO
0xF00A	frame_type	[2:0]	• 001: Fixed mode • 010: Incremental mode • 100: Random mode	0x0	RW
0xF00B	PXGMII_client_loopback	[0]	Set the value to 1 to enable XGMII RX loopback to XGMII TX.	0x0	RW

Related Information

E-Tile Hard IP for Ethernet Intel FPGA IP core register descriptions

4. Document Revision History for the E-Tile Hard IP for Ethernet Intel FPGA Design Example User Guide

Document Version	Intel Quartus Prime Version	Changes
2019.02.14	18.1.1	Updated Table: *Steps to Simulate the Testbench* to include instruction for Xcelium simulator.
2019.01.04	18.1.1	• Added information for the following design examples: — 10GE/25GE PCS Only, OTN, and FlexE RSFEC simulation and compilation-only project design examples. — 10GE/25GE PCS Only hardware design example. — Multi channel 10GE/25GE MAC + PCS with optional RSFEC and PTP simulation, compilation-only project, and hardware design examples. — 100GE MAC + PCS with optional RSFEC(544,514) simulation, compilation-only project, and hardware design examples. — 100GE PCS Only with optional RSFEC(528,514) and RSFEC(544,514) simulation, compilation-only, and hardware design examples. • Updated steps to test 10GE/25GE MAC + PCS with optional RSFEC and optional PTP and 100GE MAC+PCS with optional RSFEC and PMA calibration hardware design examples. • Updated result log for 10GE/25GE MAC + PCS with optional RSFEC and optional PTP and 100GE MAC+PCS with optional RSFEC and PMA calibration hardware design examples. • Updated simulation and hardware design example block diagram for 10GE/25GE MAC + PCS with optional RSFEC and non-PTP 10GE/25GE MAC + PCS with optional RSFEC variants. • Updated register map for 10GE/25GE and 100GE design examples.
2018.08.10	18.0	Added a note to clarify that the E-Tile Hard IP for Ethernet Intel FPGA IP provides preliminary support for the OTN feature in the following sections: • *Quick Start Guide* • *100GE with Optional RSFEC Design Example* • *100GE OTN Simulation Design Example*
2018.07.19	18.0	Initial release.