



# 25G Ethernet Intel® Stratix® 10 FPGA IP Design Example User Guide

Updated for Intel® Quartus® Prime Design Suite: **19.1**



[Subscribe](#)

[Send Feedback](#)

**UG-20110 | 2019.05.10**

Latest document on the web: [PDF](#) | [HTML](#)



## Contents

---

<b>1. 25G Ethernet Intel® FPGA IP Quick Start Guide.....</b>	<b>4</b>
1.1. Directory Structure.....	5
1.2. Generating the Design Example.....	6
1.2.1. Design Example Parameters.....	7
1.3. Simulating the 25G Ethernet Intel FPGA IP Design Example Testbench.....	7
1.3.1. Procedure.....	7
1.4. Compiling and Configuring the Design Example in Hardware.....	8
1.4.1. Procedure.....	8
1.5. Testing the 25G Ethernet Intel FPGA IP Design in Hardware.....	9
1.5.1. Procedure.....	9
<b>2. 10G/25G Ethernet Single-Channel Design Example for Intel Stratix 10 Devices.....</b>	<b>10</b>
2.1. Features.....	10
2.2. Hardware and Software Requirements.....	10
2.3. Functional Description.....	10
2.3.1. Design Components.....	13
2.4. Simulation.....	14
2.4.1. Testbench.....	14
2.4.2. Simulation Design Example Components.....	15
2.4.3. Test Case—Design Example Without the IEEE 1588v2 Feature.....	15
2.4.4. Test Case—Design Example with the IEEE 1588v2 Feature.....	17
2.5. Compilation.....	20
2.6. Hardware Testing.....	20
2.6.1. Test Procedure—Design Example Without the IEEE 1588v2 Feature.....	21
2.6.2. Test Procedure—Design Example with the IEEE 1588v2 Feature.....	24
<b>3. 25G Ethernet Single-Channel Design Example for Intel Stratix 10 Devices.....</b>	<b>25</b>
3.1. Features.....	25
3.2. Hardware and Software Requirements.....	25
3.3. Functional Description.....	25
3.3.1. Design Components.....	28
3.4. Simulation.....	29
3.4.1. Testbench.....	29
3.4.2. Simulation Design Example Components.....	30
3.4.3. Test Case—Design Example Without the IEEE 1588v2 Feature.....	30
3.4.4. Test Case—Design Example with the IEEE 1588v2 Feature.....	31
3.5. Compilation.....	34
3.6. Hardware Testing.....	35
3.6.1. Test Procedure—Design Example With and Without the IEEE 1588v2 Feature... ..	35
<b>4. 25G Ethernet Multi-Channel Design Example for Intel Stratix 10 Devices.....</b>	<b>36</b>
4.1. Features.....	36
4.2. Hardware and Software Requirements.....	36
4.3. Functional Description.....	36
4.3.1. Design Components.....	37
4.4. Simulation.....	38
4.4.1. Testbench.....	38
4.4.2. Simulation Design Example Components.....	39

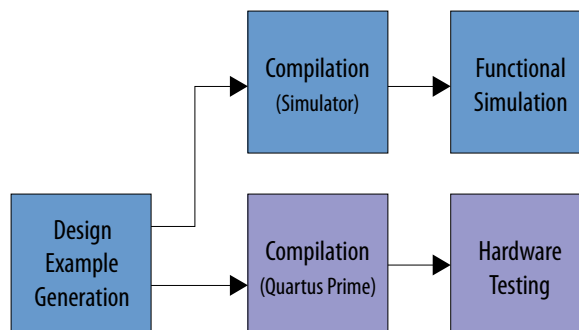


- 4.4.3. Test Case..... 39
- 4.5. Compilation.....42
- 4.6. Hardware Testing..... 43
  - 4.6.1. Test Procedure.....43
- 5. 25G Ethernet Intel FPGA IP Design Example References..... 44**
  - 5.1. Design Example Interface Signals..... 44
  - 5.2. Design Example Registers..... 45
- 6. 25G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide Archives..... 47**
- 7. Document Revision History for the 25G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide.....48**

## 1. 25G Ethernet Intel® FPGA IP Quick Start Guide

The 25G Ethernet (25GbE) Intel® FPGA IP core for Intel Stratix® 10 devices provides the capability of generating design examples for selected configurations.

**Figure 1. Development Stages for the Design Example**



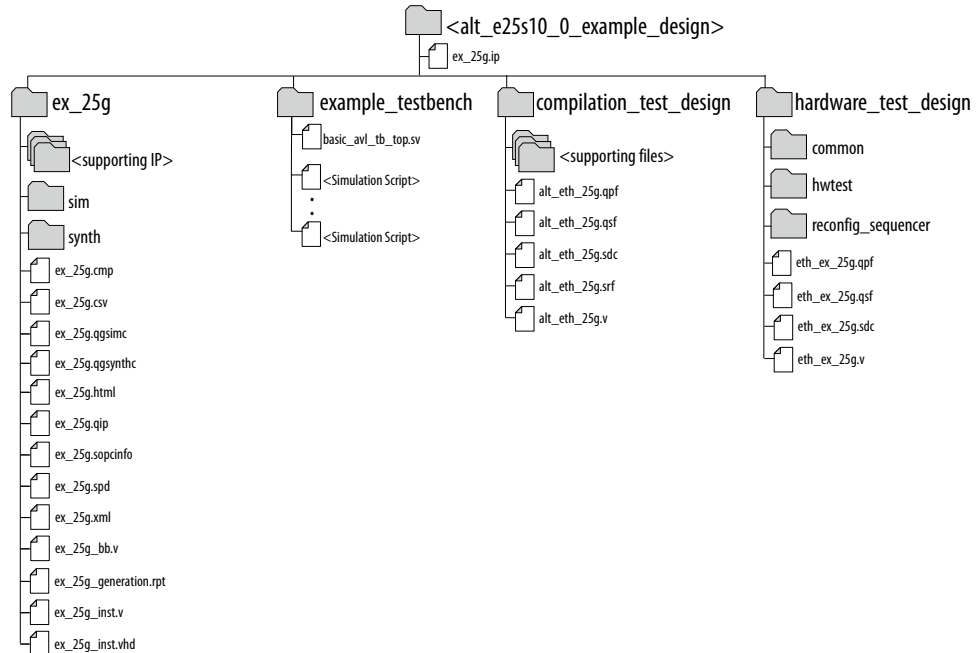
### Related Information

- [10G/25G Ethernet Single-Channel Design Example for Intel Stratix 10 Devices](#) on page 10  
Provides details for the 10G/25G Ethernet single-channel design example.
- [25G Ethernet Single-Channel Design Example for Intel Stratix 10 Devices](#) on page 25  
Provides details for the 25G Ethernet single-channel design example.
- [25G Ethernet Multi-Channel Design Example for Intel Stratix 10 Devices](#) on page 36  
Provides details for the 25G Ethernet multi-channel design example.



## 1.1. Directory Structure

Figure 2. Directory Structure for the 25G and 10G/25G Ethernet Design Examples



- The simulation files (testbench for simulation only) are located in <design\_example\_dir>/example\_testbench.
- The compilation-only design example is located in <design\_example\_dir>/compilation\_test\_design.
- The hardware configuration and test files (the design example in hardware) are located in <design\_example\_dir>/hardware\_test\_design.

Table 1. Directory and File Descriptions

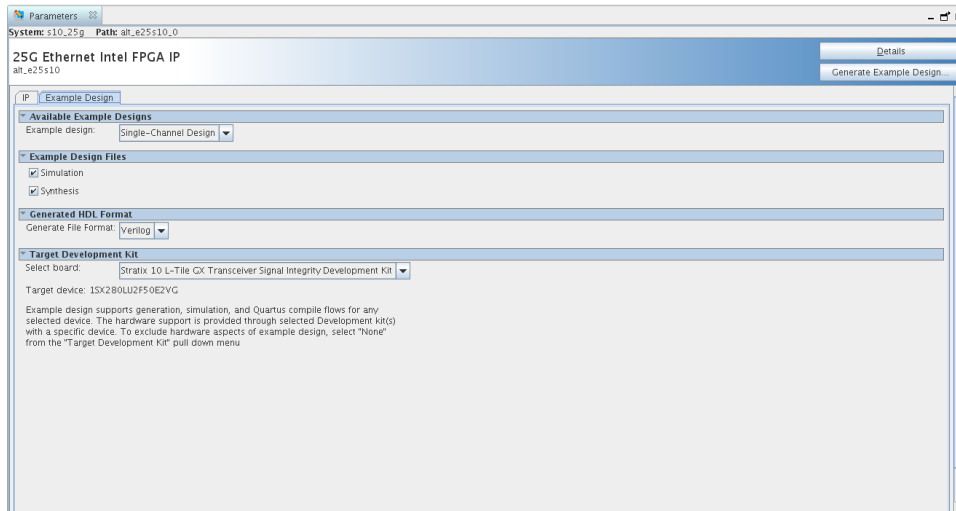
File Names	Description
eth_ex_25g.qpf	Intel Quartus® Prime project file.
eth_ex_25g.qsf	Intel Quartus Prime project settings file.
eth_ex_25g.sdc	Synopsys Design Constraints file. You can copy and modify this file for your own 25GbE Intel FPGA IP core design.
eth_ex_25g.v	Top-level Verilog HDL design example file. Single-channel design uses Verilog file.
common/	Hardware design example support files.
hwtest/main.tcl	Main file for accessing System Console.

## 1.2. Generating the Design Example

Figure 3. Procedure



Figure 4. Example Design Tab in the 25G Ethernet Intel FPGA IP Parameter Editor



Follow these steps to generate the hardware design example and testbench:

1. In the Intel Quartus Prime Pro Edition software, click **File > New Project Wizard** to create a new Quartus Prime project, or **File > Open Project** to open an existing Quartus Prime project. The wizard prompts you to specify a device.
2. In the IP Catalog, locate and select **25G Ethernet Intel FPGA IP**. The **New IP Variation** window appears.
3. Specify a top-level name for your IP variation and click **OK**. The parameter editor adds the top-level .ip file to the current project automatically. If you are prompted to manually add the .ip file to the project, click **Project > Add/Remove Files in Project** to add the file.
4. In the Intel Quartus Prime Pro Edition software, you must select a specific Intel Stratix 10 device in the **Device** field, or keep the default device that the Intel Quartus Prime software proposes.
 

*Note:* The hardware design example overwrites the selection with the device on the target board. You specify the target board from the menu of design example options in the **Example Design** tab (Step 8).
5. Click **OK**. The parameter editor appears.
6. On the **IP** tab, specify the parameters for your IP core variation.
7. On the **Example Design** tab, for **Example Design Files**, select the **Simulation** option to generate the testbench, and select the **Synthesis** option to generate the hardware design example. Only Verilog HDL files are generated.



*Note:* A functional VHDL IP core is not available. Specify Verilog HDL only, for your IP core design example.

8. For **Target Development Kit**, select the **Intel Stratix 10 L-Tile GX Transceiver Signal Integrity Development Kit**.
9. Click **Generate Example Design**. The **Select Example Design Directory** window appears.
10. If you want to modify the design example directory path or name from the defaults displayed (`alt_e25s10_0_example_design`), browse to the new path and type the new design example directory name (`<design_example_dir>`).
11. Click **OK**.

### 1.2.1. Design Example Parameters

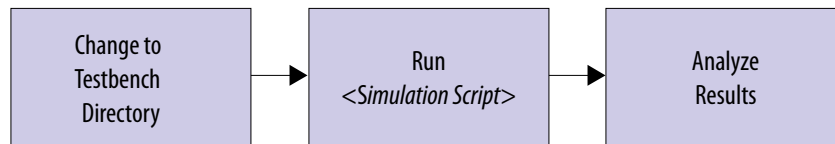
**Table 2. Parameters in the Example Design Tab**

Parameter	Description
<b>Example Design</b>	Available example designs for the IP parameter settings.
<b>Example Design Files</b>	The files to generate for the different development phase. <ul style="list-style-type: none"> <li>• Simulation—generates the necessary files for simulating the example design.</li> <li>• Synthesis—generates the synthesis files. Use these files to compile the design in the Intel Quartus Prime Pro Edition software for hardware testing and perform static timing analysis.</li> </ul>
<b>Generate File Format</b>	The format of the RTL files for simulation—Verilog.
<b>Select Board</b>	Supported hardware for design implementation. When you select an Intel FPGA development board, the <i>Target Device</i> is the one that matches the device on the Development Kit. If this menu is not available, there is no supported board for the options that you select. <b>Intel Stratix 10 L-Tile GX Transceiver Signal Integrity Development Kit:</b> This option allows you to test the design example on the selected Intel FPGA IP development kit. This option automatically selects the <i>Target Device</i> to match the device on the Intel FPGA IP development kit. If your board revision has a different device grade, you can change the target device. <b>None:</b> This option excludes the hardware aspects for the design example.

## 1.3. Simulating the 25G Ethernet Intel FPGA IP Design Example Testbench

### 1.3.1. Procedure

You can compile and simulate the design by running a simulation script from the command prompt.



1. At the command prompt, change the working directory to <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.

**Table 3. Steps to Simulate the Testbench**

Simulator	Instructions
ModelSim*	In the command line, type <code>vsim do run_vsim.do</code> <i>Note:</i> The ModelSim-AE and ModelSim-ASE simulators cannot simulate this IP core. You must use another supported ModelSim simulator such as ModelSim SE.
VCS*	In the command line, type <code>sh run_vcs.sh</code>
NCSim	In the command line, type <code>sh run_ncsim.sh</code>
Xcelium*	In the command line, type <code>sh run_xcelium.sh</code>

A successful simulation ends with the following message:

```
Simulation Passed.
```

or

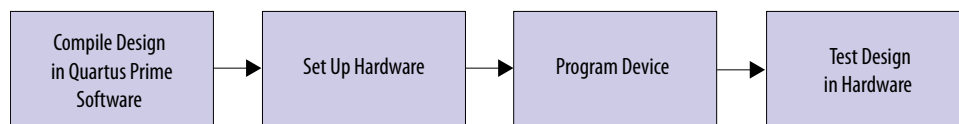
```
Testbench complete.
```

After successful completion, you can analyze the results.

## 1.4. Compiling and Configuring the Design Example in Hardware

The 25G Ethernet Intel FPGA IP core parameter editor allows you to compile and configure the design example on a target development kit.

### 1.4.1. Procedure



To compile and configure a design example on hardware, follow these steps:

1. Launch the Intel Quartus Prime Pro Edition software and select **Processing > Start Compilation** to compile the design.
2. After you generate an SRAM object file `.sof`, 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 GX board to your Intel Quartus Prime Pro Edition session.
  - 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**.

*Note:* This design targets the Intel Stratix 10 device. Please contact your Intel FPGA representative to inquire about a platform suitable to run this hardware example.

#### Related Information

- [Incremental Compilation for Hierarchical and Team-Based Design](#)
- [Programming Intel FPGA Devices](#)

## 1.5. Testing the 25G Ethernet Intel FPGA IP Design in Hardware

### 1.5.1. Procedure

After you compile the 25G 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.

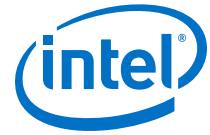
To turn on the System Console and test the hardware design example, follow these steps:

1. In the Intel Quartus Prime Pro Edition software, select **Tools > System Debugging Tools > System Console** to launch the system console.
2. In the Tcl Console pane, type `cd hwtest` to change directory to / hardware\_test\_design/hwtest.
3. Type `source main.tcl` to open a connection to the JTAG master.

Follow the test procedure in the *Hardware Testing* section of the design example and observe the test results in the System Console.

#### Related Information

[Analyzing and Debugging Designs with System Console](#)



## 2. 10G/25G Ethernet Single-Channel Design Example for Intel Stratix 10 Devices

---

The 10G/25G Ethernet single-channel design example demonstrates an Ethernet solution for Intel Stratix 10 devices using the 25G Ethernet Intel FPGA IP core.

Generate the design example from the **Example Design** tab of the 25G Ethernet Intel FPGA IP parameter editor. You can choose to generate the design with or without the IEEE 1588v2 feature. You can also choose to generate the design with or without the Reed-Solomon Forward Error Correction (RS-FEC) feature.

### 2.1. Features

- Supports single Ethernet channel operating at either 10G or 25G.
- Generate design example with IEEE 1588v2 feature.
- Generate design example with RS-FEC feature.
- Generates design example separately from Intel Stratix 10 Transceiver Native PHY.
- Provides testbench and simulation script.

### 2.2. Hardware and Software Requirements

Intel uses the following hardware and software to test the design example in a Linux system:

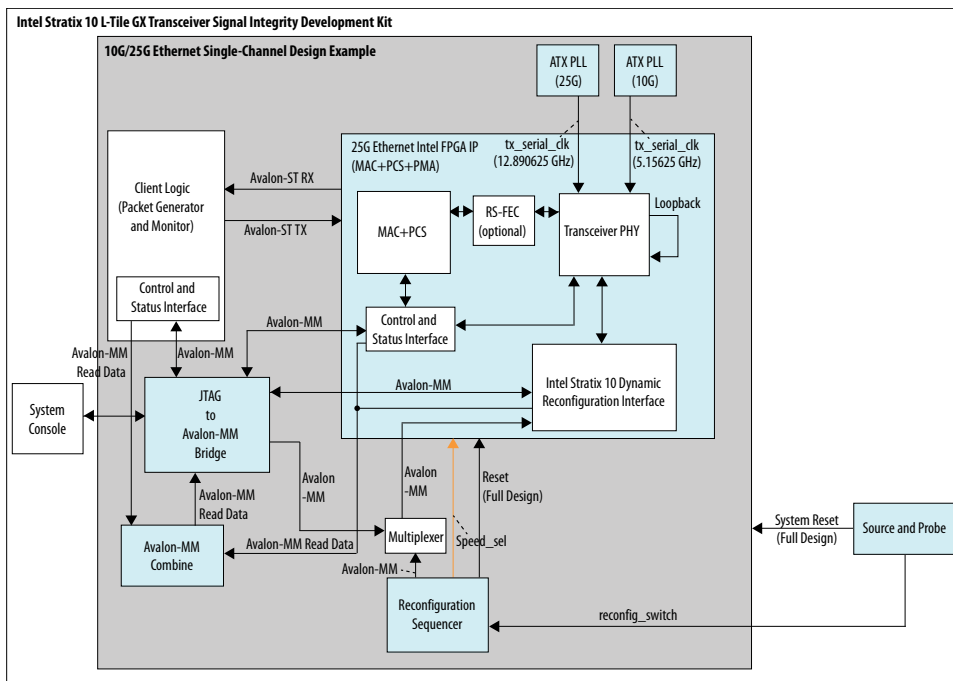
- Intel Quartus Prime Pro Edition software.
- ModelSim-SE, NCSim (Verilog only), VCS, and Xcelium simulator.
- Intel Stratix 10 L-Tile GX Transceiver Signal Integrity Development Kit (1SX280LU2F50E2VG) for hardware testing.

### 2.3. Functional Description

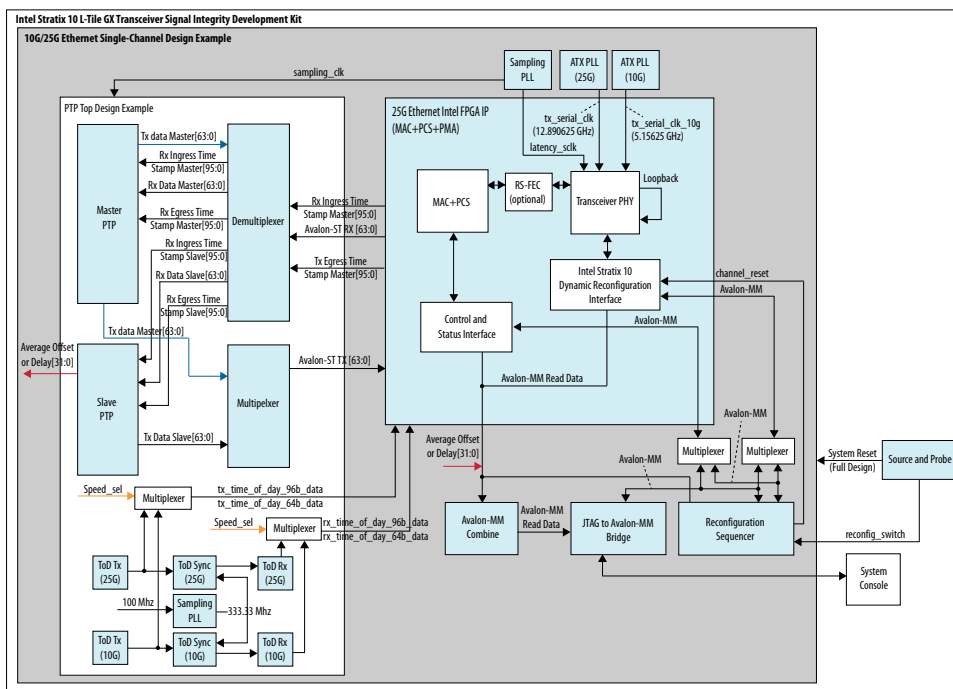
The 10G/25G Ethernet single-channel design example consists of two core variants—MAC+PCS+PMA and MAC+PCS. The following block diagrams show the design components and the top-level signals of the two core variants in the 10G/25G Ethernet single-channel design example.



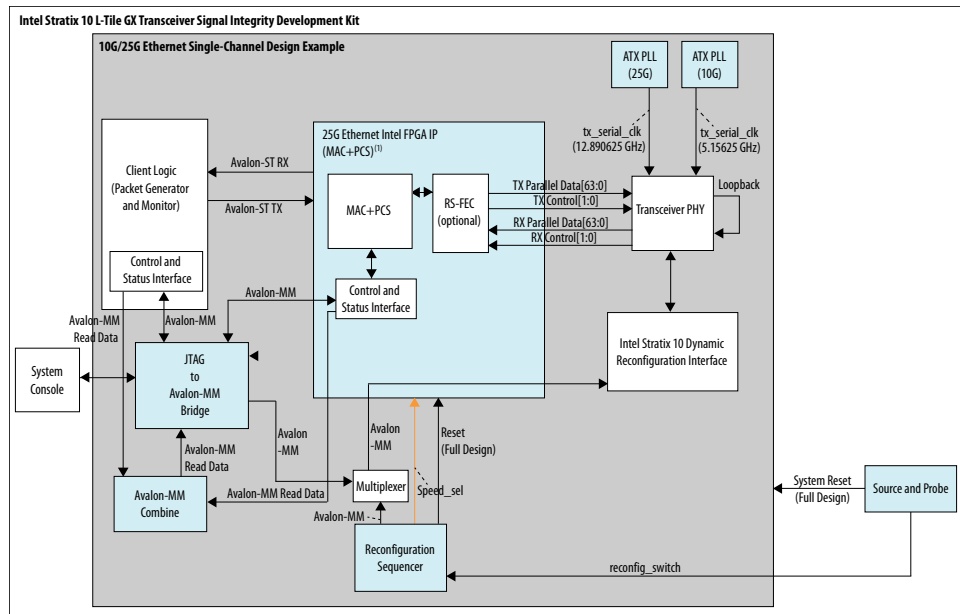
**Figure 5. Block Diagram—10G/25G Ethernet Single-Channel Design Example (MAC +PCS+PMA Core Variant) Without the IEEE 1588v2 Feature**



**Figure 6. Block Diagram—10G/25G Ethernet Single-Channel Design Example (MAC +PCS+PMA Core Variant) with the IEEE 1588v2 Feature**

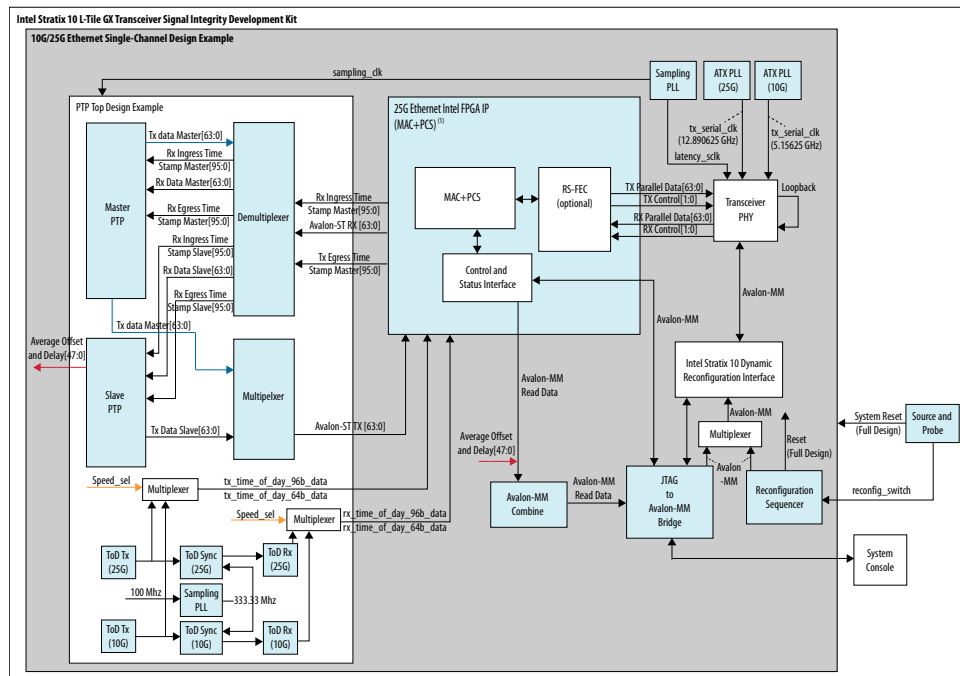


**Figure 7. Block Diagram—10G/25G Ethernet Single-Channel Design Example (MAC +PCS Core Variant) Without the IEEE 1588v2 Feature**



Note:  
1. Components outside of this block are part of the design example only.

**Figure 8. Block Diagram—10G/25G Ethernet Single-Channel Design Example (MAC +PCS Core Variant) with the IEEE 1588v2 Feature**



Note:  
1. Components outside of this block are part of the design example only.



### 2.3.1. Design Components

**Table 4. Design Components**

Component	Description
25G Ethernet Intel FPGA IP	<p>Consists of MAC, PCS, and Transceiver PHY, with the following configuration:</p> <ul style="list-style-type: none"> <li>• <b>Core Variant:</b> MAC+PCS+PMA, MAC+PCS</li> <li>• <b>Enable flow control:</b> Optional</li> <li>• <b>Enable link fault generation:</b> Optional</li> <li>• <b>Enable preamble passthrough:</b> Optional</li> <li>• <b>Enable statistics collection:</b> Optional</li> <li>• <b>Enable MAC statistics counters:</b> Optional</li> <li>• <b>Enable 10G/25G dynamic rate switching:</b> Selected</li> <li>• <b>Enable Native PHY Debug Master Endpoint (NPDME):</b> Optional</li> <li>• <b>Reference clock frequency:</b> 644.531250/322.265625</li> </ul> <p>For the design example with the IEEE 1588 feature, the following additional parameters are configured:</p> <ul style="list-style-type: none"> <li>• <b>Enable IEEE 1588:</b> Selected</li> <li>• <b>Time of day format:</b> Enable 96-bit timestamp format <sup>(1)</sup></li> </ul> <p>For the design example with the RS-FEC feature, the following additional parameter is configured:</p> <ul style="list-style-type: none"> <li>• <b>Enable RS-FEC:</b> Selected</li> </ul>
Reconfiguration Sequencer	Reconfigures the transceiver channel speed from 10 Gbps to 25 Gbps, and vice versa.
ATX PLL	Generates TX serial clocks for the 10G and 25G transceivers.
Client logic	<p>Consists of:</p> <ul style="list-style-type: none"> <li>• Traffic generator, which generates burst packets to the 25G Ethernet Intel FPGA IP core for transmission.</li> <li>• Traffic monitor, which receives burst packets from the 25G Ethernet Intel FPGA IP core.</li> </ul>
Source and Probe	Source and probe signals, including system reset input signal, which you can use for debugging.
<b>Design Components for the IEEE 1588v2 Feature</b>	
Sampling PLL	<p>Generates the clocks for the IEEE 1588v2 design components.</p> <ul style="list-style-type: none"> <li>• <code>latency_sclk</code>: 156.25 MHz for latency measurement.</li> <li>• <code>sampling_clk</code>: 250 MHz for ToD synchronization</li> </ul>
Time-of-day (ToD) Sync	Synchronizes the 10G and 25G ToDs.
ToD Tx	ToD for transmit paths for the 10G and 25G transceivers.
ToD Rx	ToD for receive paths for the 10G and 25G transceivers.
Master Precision Time Protocol (PTP)	<p>Master PTP consists of a packet generator and a packet receiver.</p> <ul style="list-style-type: none"> <li>• Packet generator: Obtains timestamp information from the 25G Ethernet Intel FPGA IP core and generates Avalon<sup>®</sup>-ST packets such as Sync packet and Delay Response packet.</li> <li>• Packet receiver: Obtains the delay request packet information from the 25G Ethernet Intel FPGA IP core and produces timestamp values.</li> </ul>
Slave PTP	Slave PTP consists of a packet generator, a packet receiver, and packet compute.
<i>continued...</i>	

<sup>(1)</sup> The 10G/25G Ethernet single-channel design example with IEEE 1588v2 feature only supports 96-bit timestamp format.

Component	Description
	<ul style="list-style-type: none"> <li>Packet generator: Obtains timestamp information from the 25G Ethernet Intel FPGA IP core and generates Avalon-ST packets such as Delay Request packet.</li> <li>Packet receiver: Obtains the Sync and Delay Response packets information from the 25G Ethernet Intel FPGA IP core and produces timestamp values.</li> <li>Packet compute: Calculates and produces the delay and offsets value based on the timestamp values.</li> </ul>

## 2.4. Simulation

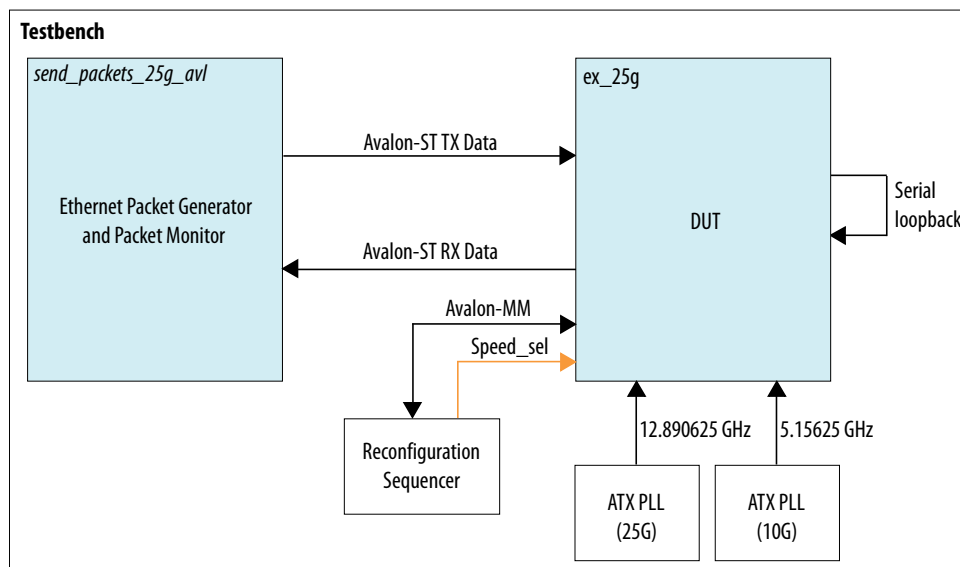
The testbench sends traffic through the IP core, exercising the transmit side and receive side of the IP core.

### Related Information

[Simulating the 25G Ethernet Intel FPGA IP Design Example Testbench on page 7](#)

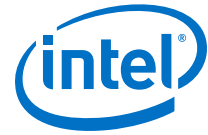
### 2.4.1. Testbench

**Figure 9. Block Diagram of the 10G/25G Ethernet Single-Channel Design Example Simulation Testbench**



**Table 5. Testbench Components**

Component	Description
Device under test (DUT)	The 25G Ethernet Intel FPGA IP core.
Reconfiguration Sequencer	Reconfigures the transceiver channel speed from 10 Gbps to 25 Gbps, and vice versa.
Ethernet Packet Generator and Packet Monitor	<ul style="list-style-type: none"> <li>Packet generator generates frames and transmit to the DUT.</li> <li>Packet Monitor monitors TX and RX datapaths and displays the frames in the simulator console.</li> </ul>
ATX PLL	Generates a TX serial clock for the Intel Stratix 10 10G/25G transceiver which is wrapped in the 25G Ethernet Intel FPGA IP core.



**Note:** For the 10G/25G Ethernet single-channel design example with IEEE 1588v2 feature simulation testbench, refer to [Figure 6](#) on page 11.

## 2.4.2. Simulation Design Example Components

**Table 6. 10G/25G Ethernet Single-Channel Design Example Testbench File Descriptions**

File Name	Description
<b>Testbench and Simulation Files</b>	
basic_avl_tb_top..sv	Top-level testbench file. The testbench instantiates the DUT, performs Avalon-MM configuration on design components and client logic, and sends and receives packet to or from .
<b>Testbench Scripts</b>	
run_vsim.do	The ModelSim script to run the testbench.
run_vcs.sh	The Synopsys VCS script to run the testbench.
run_ncsim.sh	The Cadence NCSim script to run the testbench.
run_xcelium.sh	The Xcelium script to run the testbench.

## 2.4.3. Test Case—Design Example Without the IEEE 1588v2 Feature

The simulation test case performs the following actions:

1. Instantiates 25G Ethernet Intel FPGA IP and ATX PLL.
2. Starts up the design example with an operating speed of 25G.
3. Waits for RX clock and PHY status signal to settle.
4. Prints PHY status.
5. Sends and receives 10 valid data on 25G speed.
6. Performs channel reset and switches to 10G speed.
7. Waits for RX clock and PHY status signal to settle.
8. Prints PHY status.
9. Sends and receives another 10 valid data on 10G speed.
10. Performs channel reset and switches to 25G speed.
11. Waits for RX clock and PHY status signal to settle.
12. Prints PHY status.
13. Sends and receives another 10 valid data on 25G speed.
14. Analyzes the results. The successful testbench displays "Simulation PASSED."

The following sample output illustrates a successful simulation test run:

```
Waiting for RX alignment
RX deskew locked
RX lane alignmnet locked
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...
** Received Packet     1...
** Received Packet     2...
** Sending Packet      9...
** Sending Packet     10...
** Received Packet     3...
** Received Packet     4...
** Received Packet     5...
** Received Packet     7...
** Received Packet     8...
** Received Packet     9...
** Received Packet    10...
Switching to 10G mode: 10G Reconfig start
Switching to 10G mode: 10G Reconfig End
Waiting for RX alignment
RX deskew locked
RX lane alignment locked
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...
** Received Packet     1...
** Received Packet     2...
** Sending Packet      9...
** Sending Packet     10...
** Received Packet     3...
** Received Packet     4...
** Received Packet     5...
** Received Packet     7...
** Received Packet     8...
** Received Packet     9...
** Received Packet    10...
Switching to 25G mode: 25G Reconfig start
Switching to 25G mode: 25G Reconfig End
Waiting for RX alignment
RX deskew locked
RX lane alignment locked
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...
** Received Packet     1...
** Received Packet     2...
** Sending Packet      9...
** Sending Packet     10...
** Received Packet     3...
** Received Packet     4...
** Received Packet     5...
** Received Packet     7...
** Received Packet     8...
** Received Packet     9...
** Received Packet    10...
**
** Testbench complete.
**
```





## 2.4.4. Test Case—Design Example with the IEEE 1588v2 Feature

**Note:** For 10G/25G Ethernet single-channel design example with IEEE 1588v2 feature simulation testbench, refer to [Figure 6](#) on page 11.

The simulation test case performs the following actions:

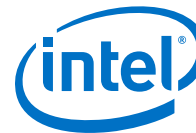
1. Instantiates 25G Ethernet Intel FPGA IP, ATX PLL, and IO PLL (sampling PLL).
2. Starts up the design example with an operating speed of 25G.
3. Waits for RX clock and PHY status signal to settle.
4. Prints PHY status.
5. Checks for 10 valid data on 25G speed.
6. Switches to 10G speed.
7. Waits for RX clock and PHY status signal to settle.
8. Prints PHY status.
9. Checks for another 10 valid data on 10G speed.
10. Switches to 25G speed once all 10 valid data passes.
11. Waits for RX clock and PHY status signal to settle.
12. Prints PHY status.
13. Checks for another 10 valid data on 25G speed.
14. Analyzes the results. The successful testbench displays "Simulation PASSED." when the PTP delay and offset data is within the threshold value.

The following sample output illustrates a successful simulation test run:

```
# Running at 25G mode...
#
#
# Waiting for RX alignment...
# iatpg_pipeline_global_en is set
# iatpg_pipeline_global_en is set
# RX deskew locked.
# RX lane alignment locked.
#
# Sending packets...
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000000064457
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000000064bb4
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000000ffff8a2
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000000643b5
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000000000520
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000000634fb
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000000063f3b
```



```
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000000063a1a
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000006445a
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000000063e95
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000000648d5
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000000643b5
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000000000520
# Offset within tolerance range.
#
#
#
# Finished sending packets.
#
#
# Switching to 10G mode: 10G Reconfig starts...
# Switching to 10G mode: 10G Reconfig End.
#
# Waiting for RX alignment...
# RX deskew locked.
# RX lane alignment locked.
#
# Configuring 1588 period...
# Configuring 1588 period done.
#
# Sending packets...
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000e5a7d
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000002013
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000e0764
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000ffff99a0
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000e0764
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000fffffa66d
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000dfa97
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000006660
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000e1431
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000005993
# Offset within tolerance range.
#
#
```



```
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000e2db7
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000ffffffccc0
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000e1431
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000ffffff8006
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000e60e4
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000ffffff320
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000dfa97
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000000005993
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000e874b
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000000000ce0
# Offset within tolerance range.
#
#
# Finished sending packets.
#
#
# Switching to 25G mode: 25G Reconfig start...
# Switching to 25G mode: 25G Reconfig end.
#
# Waiting for RX alignment...
# RX deskew locked.
# RX lane alignment locked.
#
# Configuring 1588 period...
# Configuring 1588 period done.
#
# Sending packets...
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000063c58
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000063c58
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000006502f
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000006502f
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000006554d
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000000064b10
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000000000000
# Offset within tolerance range.
#
```



```
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000064b10
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000064bb4
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000064bb4
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000065a6c
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000000000
# Offset within tolerance range.
#
#
#
# Finished sending packets.
#
# **
# ** Testbench complete.
# **
```

## 2.5. Compilation

Follow the procedure in [Compiling and Configuring the Design Example in Hardware](#) on page 8 to compile and configure the design example in the selected hardware.

You can estimate resource utilization and Fmax using the compilation-only design example. You can compile your design using the **Start Compilation** command on the **Processing** menu in the Intel Quartus Prime Pro Edition software. A successful compilation generates the compilation report summary.

For more information, refer to *Design Compilation* in the *Compiler User Guide: Intel Quartus Prime Pro Edition*.

### Related Information

#### [Design Compilation](#)

In Compiler User Guide: Intel Quartus Prime Pro Edition

## 2.6. Hardware Testing

In the hardware design example, you can program the IP core in internal serial loopback mode and generate traffic on the transmit side that loops back through the receive side.

Follow the procedure at the provided related information link to test the design example in the selected hardware.

### Related Information

#### [Testing the 25G Ethernet Intel FPGA IP Design in Hardware](#) on page 9

More information on the procedure and hardware setup.



## 2.6.1. Test Procedure—Design Example Without the IEEE 1588v2 Feature

Follow these steps to test the design examples in hardware using PMA serial loopback:

**Note:** The design example starts with default data rate of 25G.

1. Perform data rate switching to 10G:
  - a. In Intel Quartus Prime Pro Edition software, go to **Tools > In-System Sources & Probes Editor** tool to open the default source and probe GUI.
  - b. Set the source bit[1] in source and probe to 1.
2. Perform data rate switching to 25G:
  - a. In Intel Quartus Prime Pro Edition software, go to **Tools > In-System Sources & Probes Editor** tool to open the default source and probe GUI.
  - b. Set the source bit[1] in source and probe to 0.
3. Perform system reset release after executing the data rate reconfiguration:
  - a. Click **Tools > In-System Sources & Probes Editor** tool for the default Source and Probe GUI.
  - b. Toggle the system reset signal (`Source[0]`) from 0 to 1 to apply the reset and return the system reset signal back to 0 to release the system from the reset state.
  - c. Monitor the Probe signals and ensure that the status is valid.
4. Run the following commands in the system console to start the serial loopback test:

**Table 7. Command Parameters**

Parameter	Description	Example Usage
<code>chkphy_status &lt;link num&gt;</code>	Displays the clock frequencies and PHY lock status.	<code>% chkphy_status 0 # Check status of link 0</code>
<code>chkmac_stats &lt;link num&gt;</code>	Displays the values in the MAC statistics counters.	<code>% chkmac_stats 1 # Checks mac statistics counter of link 1</code>
<code>clear_all_stats &lt;link num&gt;</code>	Clears the IP core statistics counters.	<code>% clear_all_stats 1 # Clears statistics counter of link 1</code>
<code>start_gen &lt;link num&gt;</code>	Starts the packet generator.	<code>% start_gen 1 # Begin packet generation on link 1</code>
<code>stop_gen &lt;link num&gt;</code>	Stops the packet generator.	<code>% stop_gen 1 # Stop packet generation on link 1</code>
<code>loop_on &lt;link num&gt;</code>	Turns on internal serial loopback.	<code>% loop_on 2 # Turn on internal loopback on link 2</code>
<code>loop_off &lt;link num&gt;</code>	Turns off internal serial loopback.	<code>% loop_off 2 # Turn off internal loopback on link 2</code>
<code>reg_read &lt;addr&gt;</code>	Returns the IP core register value at <code>&lt;addr&gt;</code> .	<code>% reg_read 0x302 # Read IP CSR register at address 302 of link 0</code>

*continued...*



Parameter	Description	Example Usage
		% reg_read 0x4542 # Read transceiver reconfiguration register at address 4542 of link 0
reg_write <addr> <data>	Writes <data> to the IP core register at address <addr>.	% reg_write 0x30301 0x1 # Write 0x1 to IP CSR scratch register at address 301 of link 3 % reg_write 0x34542 0x0 # Write 0x0 to transceiver reconfiguration register at address 4542 of link 3

- Note:
- a. For single-channel design, <link num> is always 0.
  - b. For multi-channel design, <link num> is the channel number. The valid channel number range is 0 to 3.
- a. Type `loop_on <link num>` to turn on the internal serial loopback mode.
  - b. Type `chkphy_status <link num>` to check the status of the PHY. The TXCLK, RXCLK, and RX status should have the same values shown below for a stable link:

Figure 10. System Console Example Printout

```
Link: 0 Addr: 300
RX PHY Register Access: Checking Clock Frequencies (KHZ)

REFCLK      :0 (KHZ)
TXCLK       :39065 (KHZ)
RXCLK       :39064 (KHZ)

Link: 0 Addr: 300
RX PHY Status Polling

Rx Frequency Lock Status  0x00000001

Mac Clock in OK Condition? 0x00000007

Rx Frame Error  0x00000000

Rx PHY Fully Aligned?  0x00000001
```

- c. Type `clear_all_stats <link num>` to clear TX and RX statistics registers.
- d. Type `start_gen <link num>` to begin packet generation.
- e. Type `stop_gen <link num>` to stop packet generation.
- f. Type `chkmac_stats <link num>` to read the TX and RX statistics counters. Make sure that:
  - i. The transmitted packet frames match the received packet frames.
  - ii. No error frames are received.
- g. Type `loop_off <link num>` to turn off the internal serial loopback.



*Note:* The above configuration is applied to the default 25G mode for the first time.

**Figure 11. Sample Test Output—TX and RX Statistics Counters**

```
=====
STATISTICS FOR BASE 2304 (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 : 375474
65 - 127 Byte Frames : 187736
128 - 255 Byte Frames : 0
256 - 511 Byte Frames : 0
512 - 1023 Byte Frames : 0
1024 - 1518 Byte Frames : 0
1519 - MAX Byte Frames : 0
> MAX Byte Frames : 0
Rx Frame Starts : 0
Multicast data OK Frame : 0
Broadcast data OK Frame : 0
Unicast data OK Frames : 563210
Multicast Control Frames : 0
Broadcast Control Frames : 0
Unicast Control Frames : 0
Pause Control Frames : 0
Payload Octets OK : 27409548
Frame Octets OK : 37547328
Link: 0 Addr: 800
=====
STATISTICS FOR BASE 2048 (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 : 375474
65 - 127 Byte Frames : 187736
128 - 255 Byte Frames : 0
256 - 511 Byte Frames : 0
512 - 1023 Byte Frames : 0
1024 - 1518 Byte Frames : 0
1519 - MAX Byte Frames : 0
> MAX Byte Frames : 0
Tx Frame Starts : 0
Multicast data OK Frame : 0
Broadcast data OK Frame : 0
Unicast data OK Frames : 563210
Multicast Control Frames : 0
Broadcast Control Frames : 0
Unicast Control Frames : 0
Pause Control Frames : 0
Payload Octets OK : 27409548
Frame Octets OK : 37547328
```

### Related Information

[Design Example Registers](#) on page 45

## 2.6.2. Test Procedure—Design Example with the IEEE 1588v2 Feature

Follow these steps to test the design examples in hardware using PMA serial loopback:

**Note:** The design example starts with default data rate of 25G.

1. Perform data rate switching to 10G:
  - a. In Intel Quartus Prime Pro Edition software, go to **Tools > In-System Sources & Probes Editor** tool to open the default source and probe GUI.
  - b. Set the source bit[1] in source and probe to 1.
  - c. In the System Console panel, type the following commands as below to set the correct clock period for the required TX and RX MAC clock frequency in 10G speed mode:

```
reg_write 0xA05 0x666666  
reg_write 0xB05 0x666666
```

2. Perform data rate switching to 25G:
  - a. In Intel Quartus Prime Pro Edition software, go to **Tools > In-System Sources & Probes Editor** tool to open the default source and probe GUI.
  - b. Set the source bit[1] in source and probe to 0.
  - c. In the System Console panel, type the following commands as below to set the correct clock period for the required TX and RX MAC clock frequency in 25G speed mode:

```
reg_write 0xA05 0x28F5C  
reg_write 0xB05 0x28F5C
```

**Note:** 0xA05 is register that configure TX\_PTP\_CLK\_PERIOD. 0xB05 is register that configure RX\_PTP\_CLK\_PERIOD.

3. Perform system reset release after executing the data rate reconfiguration:
  - a. Click **Tools > In-System Sources & Probes Editor** tool for the default Source and Probe GUI.
  - b. Toggle the system reset signal (`Source[0]`) from 0 to 1 to apply the reset and return the system reset signal back to 0 to release the system from the reset state.
  - c. Monitor the Probe signals and ensure that the status is valid.
4. To perform internal serial loopback test, refer to the *Test Procedure—Design Example Without the IEEE 1588v2 Feature* section of this chapter.

### Related Information

- [Design Example Registers](#) on page 45
- [Test Procedure—Design Example Without the IEEE 1588v2 Feature](#) on page 21





## 3. 25G Ethernet Single-Channel Design Example for Intel Stratix 10 Devices

---

The 25G Ethernet single-channel design example demonstrates an Ethernet solution for Intel Stratix 10 devices using the 25G Ethernet Intel FPGA IP core.

Generate the design example from the **Example Design** tab of the 25G Ethernet Intel FPGA IP parameter editor. You can choose to generate the design with or without the IEEE 1588v2 feature. You can also choose to generate the design with or without the Reed-Solomon Forward Error Correction (RS-FEC) feature.

### 3.1. Features

- Supports single Ethernet channel operating at 25G.
- Generates design example with IEEE 1588v2 feature.
- Generates design example with RS-FEC feature.
- Generates design example separately from Intel Stratix 10 Transceiver Native PHY.
- Provides testbench and simulation script.

### 3.2. Hardware and Software Requirements

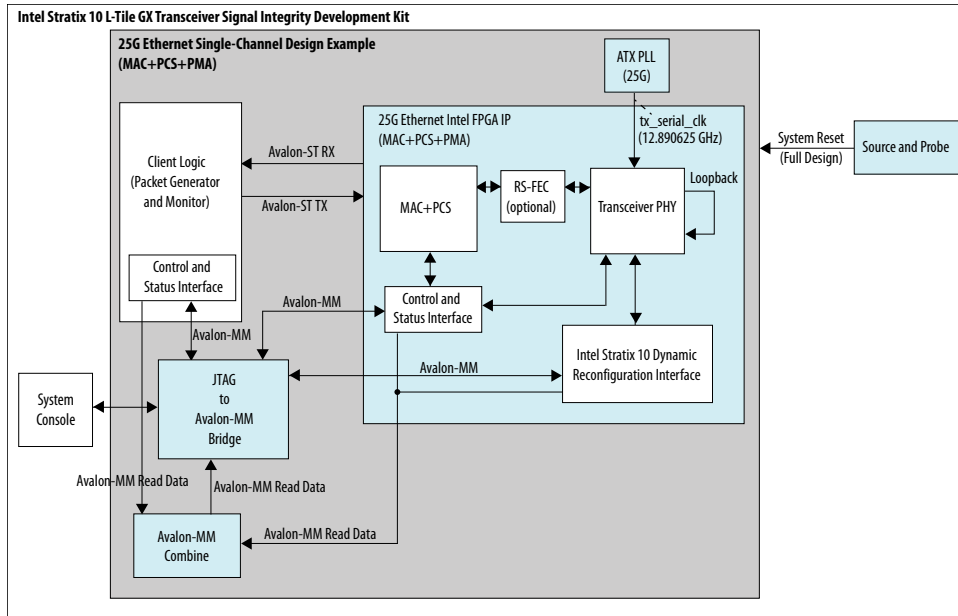
Intel uses the following hardware and software to test the design example in a Linux system:

- Intel Quartus Prime Pro Edition software.
- ModelSim-SE, NCSim (Verilog only), VCS, and Xcelium simulator.
- Intel Stratix 10 L-Tile GX Transceiver Signal Integrity Development Kit (1SX280LU2F50E2VG) for hardware testing.

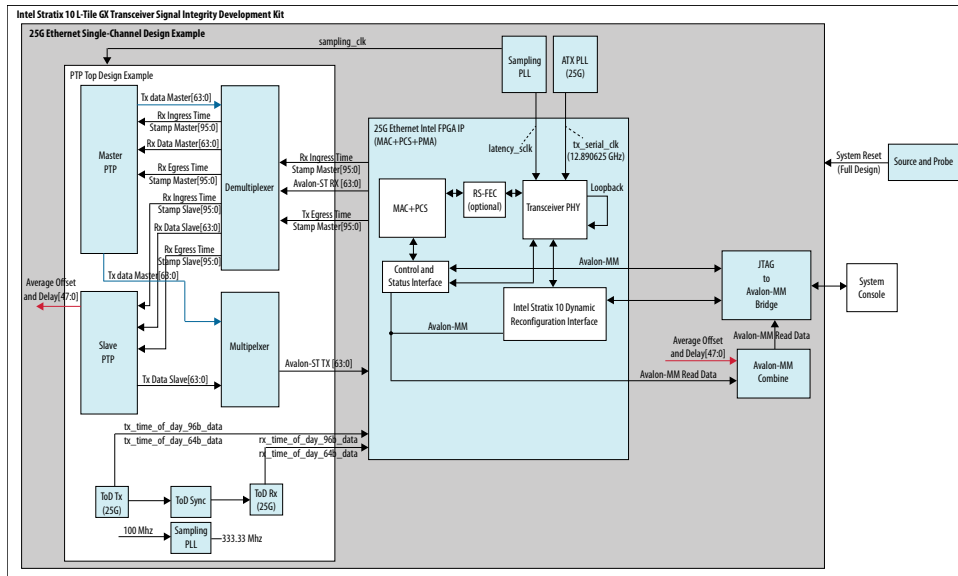
### 3.3. Functional Description

The 25G Ethernet single-channel design example consists of two core variants—MAC+PCS+PMA and MAC+PCS. The following block diagrams show the design components and the top-level signals of the two core variants in the 25G Ethernet single-channel design example.

**Figure 12. Block Diagram—25G Ethernet Single-Channel Design Example (MAC+PCS +PMA Core Variant) Without the IEEE 1588v2 Feature**

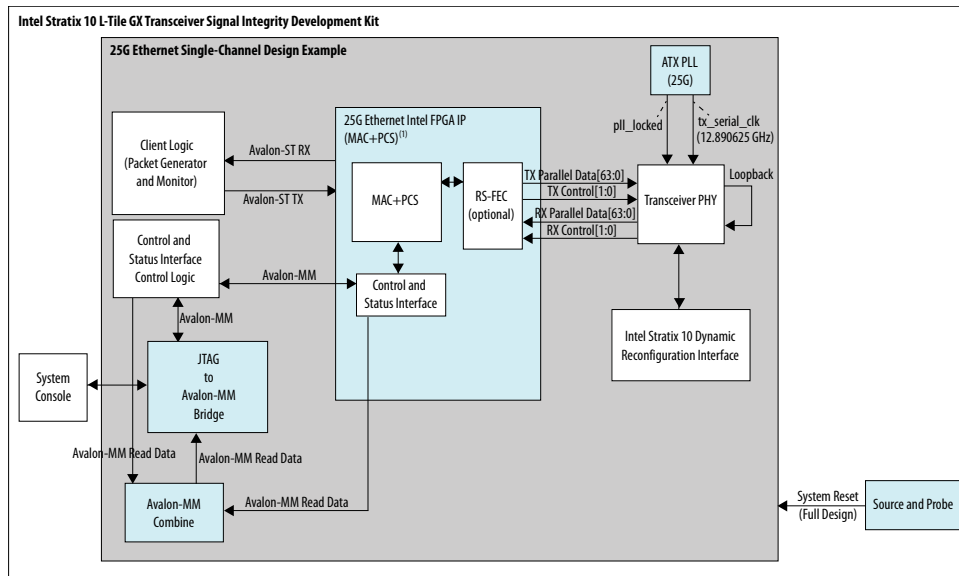


**Figure 13. Block Diagram—25G Ethernet Single-Channel Design Example (MAC+PCS +PMA Core Variant) with the IEEE 1588v2 Feature**



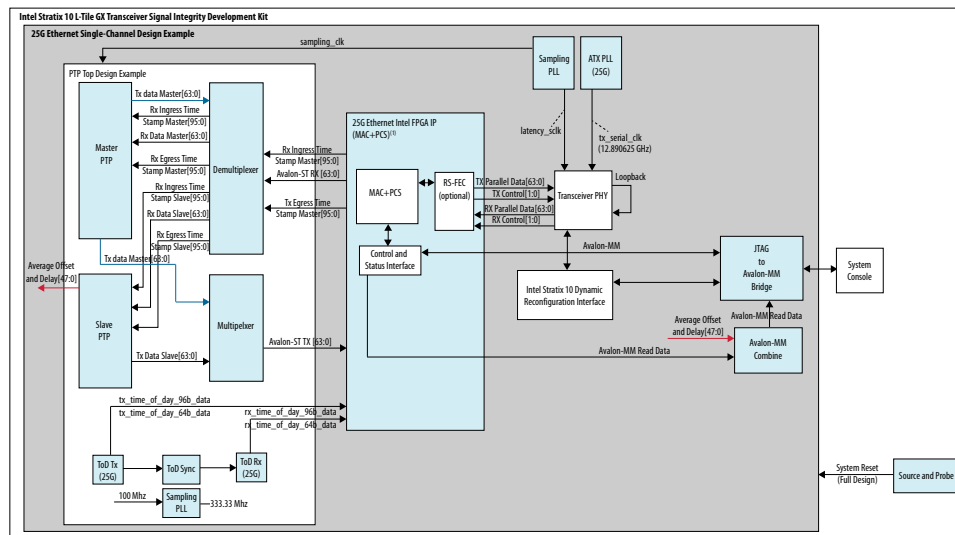


**Figure 14. Block Diagram—25G Ethernet Single-Channel Design Example (MAC+PCS Core Variant) Without the IEEE 1588v2 Feature**



Note:  
 1. Components outside of this block are part of the design example only.

**Figure 15. Block Diagram—25G Ethernet Single-Channel Design Example (MAC+PCS Core Variant) with the IEEE 1588v2 Feature**



Note:  
 1. Components outside of this block are part of the design example only.



### 3.3.1. Design Components

**Table 8. Design Components**

Component	Description
25G Ethernet Intel FPGA IP	<p>Consists of MAC, PCS, and Transceiver PHY, with the following configuration:</p> <ul style="list-style-type: none"> <li>• <b>Core Variant:</b> MAC+PCS+PMA, MAC+PCS</li> <li>• <b>Enable flow control:</b> Optional</li> <li>• <b>Enable link fault generation:</b> Optional</li> <li>• <b>Enable preamble passthrough:</b> Optional</li> <li>• <b>Enable statistics collection:</b> Optional</li> <li>• <b>Enable MAC statistics counters:</b> Optional</li> <li>• <b>Enable 10G/25G dynamic rate switching:</b> Not selected</li> <li>• <b>Enable Native PHY Debug Master Endpoint (NPDME):</b> Optional</li> <li>• <b>Reference clock frequency:</b> 644.531250/322.265625</li> </ul> <p>For the design example with the IEEE 1588 feature, the following additional parameters are configured:</p> <ul style="list-style-type: none"> <li>• <b>Enable IEEE 1588:</b> Selected</li> <li>• <b>Time of day format:</b> Enable 96-bit timestamp format <sup>(2)</sup></li> </ul> <p>For the design example with the RS-FEC feature, the following additional parameter is configured:</p> <ul style="list-style-type: none"> <li>• <b>Enable RS-FEC:</b> Selected</li> </ul>
ATX PLL	Generates TX serial clocks for the 25G transceiver.
Client logic	<p>Consists of:</p> <ul style="list-style-type: none"> <li>• Traffic generator, which generates burst packets to the 25G Ethernet Intel FPGA IP core for transmission.</li> <li>• Traffic monitor, which monitors burst packets that are coming from the 25G Ethernet Intel FPGA IP core.</li> </ul>
Source and Probe	Source and probe signals, including system reset input signal, which you can use for debugging.
<b>Design Components for the IEEE 1588v2 Feature</b>	
Sampling PLL	<p>Generates the clocks for the IEEE 1588v2 design components.</p> <ul style="list-style-type: none"> <li>• <code>latency_sclk</code>: 156.25 MHz for latency measurement.</li> <li>• <code>sampling_clk</code>: 250 MHz for ToD synchronization</li> </ul>
Time-of-day (ToD) Sync	Synchronizes the 25G ToD.
ToD Tx	ToD for transmit paths for the 25G transceiver.
ToD Rx	ToD for receive paths for the 25G transceiver.
Master Precision Time Protocol (PTP)	<p>Master PTP consists of a packet generator and a packet receiver.</p> <ul style="list-style-type: none"> <li>• Packet generator: Obtains timestamp information from the 25G Ethernet Intel FPGA IP core and generates Avalon-ST packets such as Sync packet and Delay Response packet.</li> <li>• Packet receiver: Obtains the delay request packet information from the 25G Ethernet Intel FPGA IP core and produces timestamp values.</li> </ul>
Slave PTP	<p>Slave PTP consists of a packet generator, a packet receiver, and a packet compute.</p> <ul style="list-style-type: none"> <li>• Packet generator: Obtains timestamp information from the 25G Ethernet Intel FPGA IP core and generates Avalon-ST packets such as Delay Request packet.</li> <li>• Packet receiver: Obtains the Sync and Delay Response packets information from the 25G Ethernet Intel FPGA IP core and produces timestamp values.</li> <li>• Packet compute: Calculates and produces the delay and offsets value based on the timestamp values.</li> </ul>

<sup>(2)</sup> The 25G Ethernet single-channel design example with IEEE 1588v2 feature only supports 96-bit timestamp format.



### 3.4. Simulation

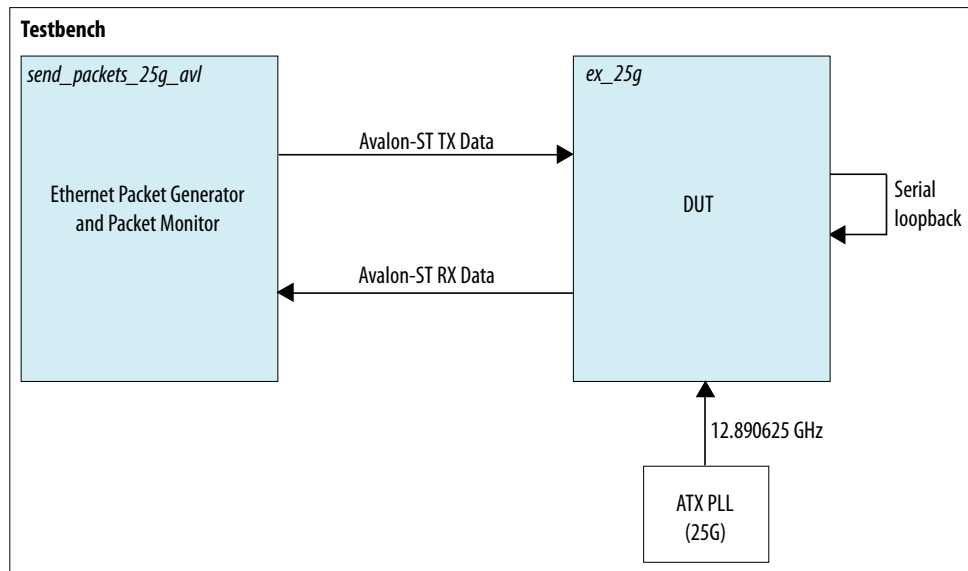
The testbench sends traffic through the IP core, exercising the transmit side and receive side of the IP core.

#### Related Information

Simulating the 25G Ethernet Intel FPGA IP Design Example Testbench on page 7

#### 3.4.1. Testbench

**Figure 16. Block Diagram of the 25G Ethernet Single-Channel Design Example Simulation Testbench**



**Table 9. Testbench Components**

Component	Description
Device under test (DUT)	The 25G Ethernet Intel FPGA IP core.
Reconfiguration Sequencer	Reconfigures the transceiver channel speed from 10 Gbps to 25 Gbps, and vice versa.
Ethernet Packet Generator and Packet Monitor	<ul style="list-style-type: none"> <li>Packet generator generates frames and transmit to the DUT.</li> <li>Packet Monitor monitors TX and RX datapaths and displays the frames in the simulator console.</li> </ul>
ATX PLL	Generates a TX serial clock for the Intel Stratix 10 10G/25G transceiver which is wrapped in the 25G Ethernet Intel FPGA IP core.

**Note:** For the 25G Ethernet single-channel design example with IEEE 1588v2 feature simulation testbench, refer to [Figure 13](#) on page 26.



### 3.4.2. Simulation Design Example Components

**Table 10. 25G Ethernet Single-Channel Design Example Testbench File Descriptions**

File Name	Description
<b>Testbench and Simulation Files</b>	
basic_avl_tb_top.v	Top-level testbench file. The testbench instantiates the DUT, performs Avalon-MM configuration on design components and client logic, and sends and receives packet to or from .
<b>Testbench Scripts</b>	
run_vsim.do	The ModelSim script to run the testbench.
run_vcs.sh	The Synopsys VCS script to run the testbench.
run_ncsim.sh	The Cadence NCSim script to run the testbench.
run_xcelium.sh	The Xcelium script to run the testbench.

### 3.4.3. Test Case—Design Example Without the IEEE 1588v2 Feature

The simulation test case performs the following actions:

1. Instantiates and ATX PLL.
2. Waits for RX clock and PHY status signal to settle.
3. Prints PHY status.
4. Analyzes the results. The successful testbench sends ten packets, receives ten packets, and displays "Testbench complete."



**Figure 17. Sample Simulation Output for Design Example Without the IEEE 1588v2 Feature**

```
#rx_pcs_ready[ 0]
#RX deskew locked
#RX lane alignment locked
#TX enabled
*** Link          0  Sending Packet          1...
*** Link          0  Sending Packet          2...
*** Link          0  Sending Packet          3...
*** Link          0  Sending Packet          4...
*** Link          0  Sending Packet          5...
*** Link          0  Sending Packet          6...
*** Link          0  Sending Packet          7...
*** Link          0  Sending Packet          8...
*** Link          0  Received Packet         1...
*** Link          0  Received Packet         2...
*** Link          0  Sending Packet          9...
*** Link          0  Sending Packet         10...
*** Link          0  Received Packet         3...
*** Link          0  Received Packet         4...
*** Link          0  Received Packet         5...
*** Link          0  Received Packet         6...
*** Link          0  Received Packet         7...
*** Link          0  Received Packet         8...
*** Link          0  Received Packet         9...
*** Link          0  Received Packet        10...
***
*** Testbench complete.
***
```

### 3.4.4. Test Case—Design Example with the IEEE 1588v2 Feature

*Note:* For 25G Ethernet single-channel design example with IEEE 1588v2 feature simulation testbench, refer to [Figure 13](#) on page 26.

The simulation test case performs the following actions:

1. Instantiates , ATX PLL, and IO PLL (sampling PLL).
2. Waits for RX clock and PHY status signal to settle.
3. Prints PHY status.
4. Checks for 10 valid data.
5. Analyzes the results. The successful testbench displays "Testbench complete." when the PTP delay and offset data are within the threshold values.



The following sample output illustrates a successful simulation test run:

```
#
# Waiting for RX alignment...
# iatpg_pipeline_global_en is set
# iatpg_pipeline_global_en is set
# RX deskew locked.
# RX lane alignment locked.
#
# Sending packets...
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000064457
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000064bb4
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000ffffffffff8a2
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000643b5
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000000520
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000634fb
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000063f3b
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000063a1a
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000006445a
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000063e95
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000648d5
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000643b5
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000000520
# Offset within tolerance range.
#
#
# Finished sending packets.
#
# **
# ** Testbench complete.
# **
```





**Figure 18. Sample Simulation Output for Design Example with the IEEE 1588v2 Feature (Part 1 of 2)**

```
#
# Waiting for RX alignment...
# iatpg_pipeline_global_en is set
# iatpg_pipeline_global_en is set
# RX deskew locked.
# RX lane alignment locked.
#
# Sending packets...
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000003c3b7
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000003d2c2
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000ffffffffffb34
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000003cdf7
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000003cdf7
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000000003c363
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000009ec
# Offset within tolerance range.
```

**Figure 19. Sample Simulation Output for Design Example with the IEEE 1588v2 Feature (Part 2 of 2)**

```
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000003c883
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000004cc
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000003c883
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000004cc
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000003cd50
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000000000
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000003c883
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x00000000000000fffffffb33
# Offset within tolerance range.
#
#
# Delay (sec[95:48],ns[47:16],fns[15:0]): 0x000000000000000000003d2c3
# Offset(sec[95:48],ns[47:16],fns[15:0]): 0x0000000000000000000004cc
# Offset within tolerance range.
#
#
# Finished sending packets.
#
# **
# ** Testbench complete.
# **
# *****
# ** Note: $finish      : ./basic_av1_tb_top.sv(129)
#      Time: 150355 ns  Iteration: 2  Instance: /basic_av1_tb_top
```

## 3.5. Compilation

Follow the procedure in [Compiling and Configuring the Design Example in Hardware](#) on page 8 to compile and configure the design example in the selected hardware.

You can estimate resource utilization and Fmax using the compilation-only design example. You can compile your design using the **Start Compilation** command on the **Processing** menu in the Intel Quartus Prime Pro Edition software. A successful compilation generates the compilation report summary.

For more information, refer to *Design Compilation* in the *Compiler User Guide: Intel Quartus Prime Pro Edition*.



### Related Information

#### Design Compilation

In Compiler User Guide: Intel Quartus Prime Pro Edition

## 3.6. Hardware Testing

In the hardware design example, you can program the IP core in internal serial loopback mode and generate traffic on the transmit side that loops back through the receive side.

Follow the procedure at the provided related information link to test the design example in the selected hardware.

### Related Information

[Testing the 25G Ethernet Intel FPGA IP Design in Hardware](#) on page 9

More information on the procedure and hardware setup.

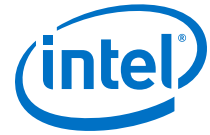
### 3.6.1. Test Procedure—Design Example With and Without the IEEE 1588v2 Feature

Follow these steps to test the design example in hardware:

1. Before you run the hardware testing for this design example, you must reset the system:
  - a. Click **Tools** ► **In-System Sources & Probes Editor** tool for the default Source and Probe GUI.
  - b. Toggle the system reset signal (`Source[0]`) from 0 to 1 to apply the reset and return the system reset signal back to 0 to release the system from the reset state.
  - c. Monitor the Probe signals and ensure that the status is valid.
2. To perform internal serial loopback test, refer to the *Test Procedure—Design Example Without the IEEE 1588v2 Feature* section of the *10G/25G Ethernet Single-Channel Design Example for Intel Stratix 10 Devices* chapter.

### Related Information

- [Design Example Registers](#) on page 45
- [Test Procedure—Design Example Without the IEEE 1588v2 Feature](#) on page 21



## 4. 25G Ethernet Multi-Channel Design Example for Intel Stratix 10 Devices

---

The 25G Ethernet multi-channel design example demonstrates an Ethernet solution for Intel Stratix 10 devices using the 25G Ethernet Intel FPGA IP core.

Generate the design example from the **Example Design** tab of the 25G Ethernet Intel FPGA IP parameter editor.

### 4.1. Features

- Supports up to four Ethernet channels operating at 25G.
- Provides testbench and simulation script.

### 4.2. Hardware and Software Requirements

Intel uses the following hardware and software to test the design example in a Linux system:

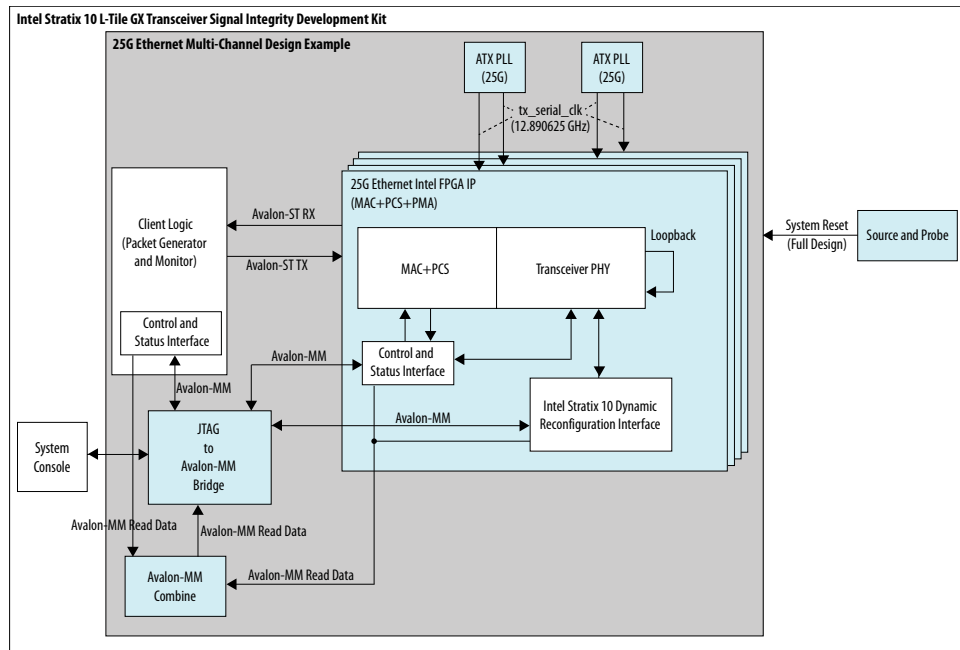
- Intel Quartus Prime Pro Edition software.
- ModelSim-SE, NCSim (Verilog only), VCS, and Xcelium simulator.
- Intel Stratix 10 L-Tile GX Transceiver Signal Integrity Development Kit (1SX280LU2F50E2VG) for hardware testing.

### 4.3. Functional Description

The 25G Ethernet multi-channel design example consists of various components. The following block diagram shows the design components and the top-level signals of the design example.



**Figure 20. Block Diagram—25G Ethernet Multi-Channel Design Example (MAC+PCS+PMA Core Variant)**



### 4.3.1. Design Components

**Table 11. Design Components**

Component	Description
25G Ethernet Intel FPGA IP	Consists of MAC, PCS, and Transceiver PHY, with the following configuration: <ul style="list-style-type: none"> <li>• <b>Core Variant:</b> MAC+PCS+PMA</li> <li>• <b>Enable RS-FEC:</b> Not selected</li> <li>• <b>Enable flow control:</b> Optional</li> <li>• <b>Enable link fault generation:</b> Optional</li> <li>• <b>Enable preamble passthrough:</b> Optional</li> <li>• <b>Enable statistics collection:</b> Optional</li> <li>• <b>Enable MAC statistics counters:</b> Optional</li> <li>• <b>Enable IEEE 1588:</b> Not selected</li> <li>• <b>Enable 10G/25G dynamic rate switching:</b> Not selected</li> <li>• <b>Enable Native PHY Debug Master Endpoint (NPDME):</b> Optional</li> <li>• <b>Reference clock frequency:</b> 644.531250/322.265625</li> </ul>
ATX PLL	Generates TX serial clocks for the 25G transceiver.
Client logic	Consists of: <ul style="list-style-type: none"> <li>• Traffic generator, which generates burst packets to the 25G Ethernet Intel FPGA IP core for transmission.</li> <li>• Traffic monitor, which receives burst packets from the 25G Ethernet Intel FPGA IP core.</li> </ul>
Source and Probe	Source and probe signals, including system reset input signal, which you can use for debugging.

## 4.4. Simulation

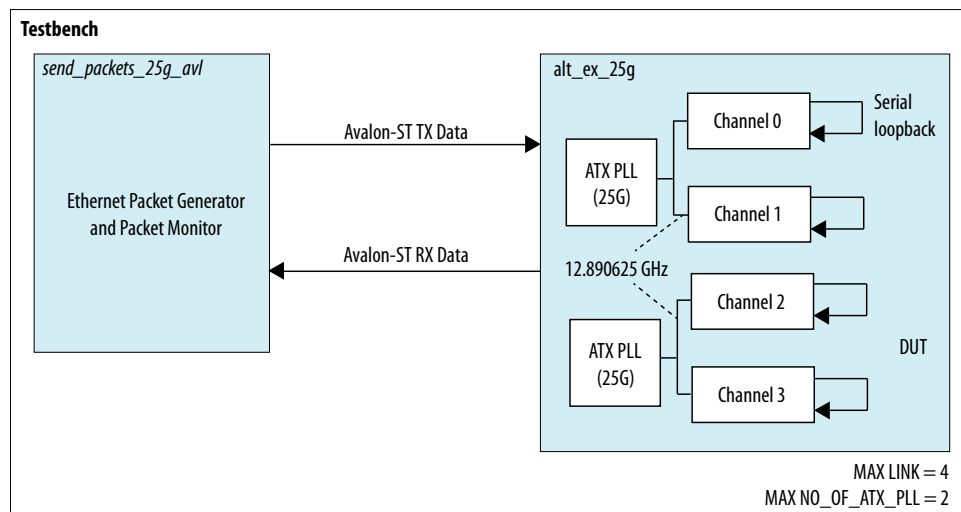
The testbench sends traffic through the IP core, exercising the transmit side and receive side of the IP core.

### Related Information

Simulating the 25G Ethernet Intel FPGA IP Design Example Testbench on page 7

### 4.4.1. Testbench

**Figure 21. Block Diagram of the 25G Ethernet Multi-Channel Design Example Simulation Testbench**



**Table 12. Testbench Components**

Component	Description
Device under test (DUT)	The 25G Ethernet Intel FPGA IP core.
Reconfiguration Sequencer	Reconfigures the transceiver channel speed from 10 Gbps to 25 Gbps, and vice versa.
Ethernet Packet Generator and Packet Monitor	<ul style="list-style-type: none"> <li>Packet generator generates frames and transmit to the DUT.</li> <li>Packet Monitor monitors TX and RX datapaths and displays the frames in the simulator console.</li> </ul>
ATX PLL	Generates a TX serial clock for the Intel Stratix 10 10G/25G transceiver which is wrapped in the 25G Ethernet Intel FPGA IP core.



## 4.4.2. Simulation Design Example Components

**Table 13. 25G Ethernet Multi-Channel Design Example Testbench File Descriptions**

File Name	Description
<b>Testbench and Simulation Files</b>	
basic_avl_tb_top.v	Top-level testbench file. The testbench instantiates the DUT, performs Avalon-MM configuration on design components and client logic, and sends and receives packet to or from the 25G Ethernet Intel FPGA IP.
<b>Testbench Scripts</b>	
run_vsim.do	The ModelSim script to run the testbench.
run_vcs.sh	The Synopsys VCS script to run the testbench.
run_ncsim.sh	The Cadence NCSim script to run the testbench.
run_xcelium.sh	The Xcelium script to run the testbench.

## 4.4.3. Test Case

The simulation test case performs the following steps:

1. Instantiates 25G Ethernet Intel FPGA IP and ATX PLL.
2. Waits for PHY status signal to settle.
3. Prints PHY status.
4. Analyzes the results. The successful testbench sends and receives packets, and displays "Testbench complete."



**Figure 22. Sample Simulation Output when Ethernet Channel is Configured to 1**

This figure shows a successful simulation test run when the Ethernet channel (i.e., LINK) is configured to 1.

```
#rx_pcs_ready[ 0]
#RX deskew locked
#RX lane alignment locked
#TX enabled
#** Link      0  Sending Packet      1...
#** Link      0  Sending Packet      2...
#** Link      0  Sending Packet      3...
#** Link      0  Sending Packet      4...
#** Link      0  Sending Packet      5...
#** Link      0  Sending Packet      6...
#** Link      0  Sending Packet      7...
#** Link      0  Sending Packet      8...
#** Link      0  Received Packet     1...
#** Link      0  Received Packet     2...
#** Link      0  Sending Packet      9...
#** Link      0  Sending Packet     10...
#** Link      0  Received Packet     3...
#** Link      0  Received Packet     4...
#** Link      0  Received Packet     5...
#** Link      0  Received Packet     6...
#** Link      0  Received Packet     7...
#** Link      0  Received Packet     8...
#** Link      0  Received Packet     9...
#** Link      0  Received Packet    10...
#**
#** Testbench complete.
#**
```





**Figure 23. Sample Simulation Output when Ethernet Channel is Configured to 4 (Part 1 of 2)**

This figure shows a successful simulation test run when the Ethernet channel (i.e., LINK) is configured to 4.

```
rx_pcs_ready[      0]
rx_pcs_ready[      1]
rx_pcs_ready[      2]
rx_pcs_ready[      3]
RX deskew locked
RX lane alignment locked
TX enabled
** Link      0  Sending Packet      1...
** Link      0  Sending Packet      2...
** Link      0  Sending Packet      3...
** Link      0  Sending Packet      4...
** Link      0  Sending Packet      5...
** Link      0  Sending Packet      6...
** Link      0  Sending Packet      7...
** Link      0  Sending Packet      8...
** Link      0  Received Packet     1...
** Link      0  Received Packet     2...
** Link      0  Sending Packet      9...
** Link      0  Sending Packet     10...
** Link      0  Received Packet     3...
** Link      0  Received Packet     4...
** Link      0  Received Packet     5...
** Link      0  Received Packet     6...
** Link      0  Received Packet     7...
** Link      0  Received Packet     8...
** Link      0  Received Packet     9...
** Link      0  Received Packet    10...
** Link      1  Sending Packet      1...
** Link      1  Sending Packet      2...
** Link      1  Sending Packet      3...
** Link      1  Sending Packet      4...
** Link      1  Sending Packet      5...
** Link      1  Sending Packet      6...
** Link      1  Sending Packet      7...
** Link      1  Sending Packet      8...
** Link      1  Received Packet     1...
** Link      1  Received Packet     2...
** Link      1  Received Packet     9...
** Link      1  Received Packet    10...
** Link      1  Received Packet     3...
** Link      1  Received Packet     4...
** Link      1  Received Packet     5...
** Link      1  Received Packet     6...
** Link      1  Received Packet     7...
** Link      1  Received Packet     8...
** Link      1  Received Packet     9...
** Link      1  Received Packet    10...
```



**Figure 24. Sample Simulation Output when Ethernet Channel is Configured to 4 (Part 2 of 2)**

This figure shows a successful simulation test run when the Ethernet channel (i.e., LINK) is configured to 4.

```
** Link      2  Sending Packet      1...
** Link      2  Sending Packet      2...
** Link      2  Sending Packet      3...
** Link      2  Sending Packet      4...
** Link      2  Sending Packet      5...
** Link      2  Sending Packet      6...
** Link      2  Sending Packet      7...
** Link      2  Sending Packet      8...
** Link      2  Received Packet     1...
** Link      2  Received Packet     2...
** Link      2  Sending Packet      9...
** Link      2  Sending Packet     10...
** Link      2  Received Packet      3...
** Link      2  Received Packet      4...
** Link      2  Received Packet      5...
** Link      2  Received Packet      6...
** Link      2  Received Packet      7...
** Link      2  Received Packet      8...
** Link      2  Received Packet      9...
** Link      2  Received Packet     10...
** Link      3  Sending Packet      1...
** Link      3  Sending Packet      2...
** Link      3  Sending Packet      3...
** Link      3  Sending Packet      4...
** Link      3  Sending Packet      5...
** Link      3  Sending Packet      6...
** Link      3  Sending Packet      7...
** Link      3  Sending Packet      8...
** Link      3  Received Packet      1...
** Link      3  Received Packet      2...
** Link      3  Sending Packet      9...
** Link      3  Sending Packet     10...
** Link      3  Received Packet      3...
** Link      3  Received Packet      4...
** Link      3  Received Packet      5...
** Link      3  Received Packet      6...
** Link      3  Received Packet      7...
** Link      3  Received Packet      8...
** Link      3  Received Packet      9...
** Link      3  Received Packet     10...
**
** Testbench complete.
**
*****
```

### 4.5. Compilation

Follow the procedure in [Compiling and Configuring the Design Example in Hardware](#) on page 8 to compile and configure the design example in the selected hardware.



You can estimate resource utilization and Fmax using the compilation-only design example. You can compile your design using the **Start Compilation** command on the **Processing** menu in the Intel Quartus Prime Pro Edition software. A successful compilation generates the compilation report summary.

For more information, refer to *Design Compilation* in the *Compiler User Guide: Intel Quartus Prime Pro Edition*.

#### Related Information

[Design Compilation](#)

In Compiler User Guide: Intel Quartus Prime Pro Edition

## 4.6. Hardware Testing

In the hardware design example, you can program the IP core in internal serial loopback mode and generate traffic on the transmit side that loops back through the receive side.

Follow the procedure at the provided related information link to test the design example in the selected hardware.

#### Related Information

[Testing the 25G Ethernet Intel FPGA IP Design in Hardware](#) on page 9

More information on the procedure and hardware setup.

### 4.6.1. Test Procedure

Follow these steps to test the design example in hardware:

1. Before you run the hardware testing for this design example, you must reset the system:
  - a. Click **Tools** ► **In-System Sources & Probes Editor** tool for the default Source and Probe GUI.
  - b. Toggle the system reset signal (`Source[0]`) from 0 to 1 to apply the reset and return the system reset signal back to 0 to release the system from the reset state.
  - c. Monitor the Probe signals and ensure that the status is valid.
2. To perform internal serial loopback test, refer to the *Test Procedure—Design Example Without the IEEE 1588v2 Feature* section of the *10G/25G Ethernet Single-Channel Design Example for Intel Stratix 10 Devices* chapter.

*Note:* `link_num` is valid for 0 to 3 only.

#### Related Information

- [Design Example Registers](#) on page 45
- [Test Procedure—Design Example Without the IEEE 1588v2 Feature](#) on page 21

## 5. 25G Ethernet Intel FPGA IP Design Example References

This section provides information about the 25G Ethernet Intel FPGA IP core interface signals and registers in the design examples.

### 5.1. Design Example Interface Signals

The 25G Ethernet Intel FPGA IP core testbench is self-contained and does not require you to drive any input signals.

**Table 14. Hardware Design Example Interface Signals for 25G Ethernet Intel FPGA IP Core for Intel Stratix 10 Devices**

Signal	Direction	Comments
clk100	Input	Drive at 100 MHz. The intent is to drive this from a 100 Mhz oscillator on the board.
clk_ref	Input	Drive at 644.53125 MHz or 322.265625 MHz from an oscillator on the board.
cpu_resetn	Input	Resets the IP core. Active low. Drives the global hard reset <code>csr_reset_n</code> to the IP core.
tx_serial	Output	Transceiver PHY output serial data.
rx_serial	Input	Transceiver PHY input serial data.
user_led[7:0]	Output	Status signals. The hardware design example connects these bits to drive LEDs on the target board. Individual bits reflect the following signal values and clock behavior: <ul style="list-style-type: none"> <li>[0]: Main reset signal to IP core</li> <li>[1]: Reserved</li> <li>[2]: Divided version of <code>clk50</code></li> <li>[3]: Divided version of 100 MHz status clock</li> <li>[4]: <code>tx_lanes_stable</code></li> <li>[5]: <code>rx_block_lock</code></li> <li>[6]: <code>rx_am_lock</code></li> <li>[7]: <code>rx_pcs_ready</code></li> </ul>

#### Related Information

##### Interfaces and Signal Descriptions

Provides detailed descriptions of the Intel Stratix 10 core signals and the interfaces to which they belong.



## 5.2. Design Example Registers

**Table 15. Hardware Design Example Register Map for 25G Ethernet Intel FPGA IP Core for Intel Stratix 10 Devices**

You access these registers with the `reg_read` and `reg_write` functions in the System Console.

Word Offset	Register Category
<b>Variant: Single-Channel</b>	
0X0000–0X0DFF	Register range to access the Status Registers.
0X4000–0X7FFF	Register range to access the Reconfiguration Registers.
0X10000–0X10001	Register range to access the Reconfiguration Registers module for 10G/25G switching.
0x1020	32-bit <code>average_offset_fnsec_r</code> register: <ul style="list-style-type: none"> <li>Read this register to obtain average offset value [47:16] in fractional nanosecond, which is derived from the offset adjustment data [96:0] of the PTP slave.</li> </ul>
0x1021	32-bit <code>average_offset_fnsec_to_mem</code> register: <ul style="list-style-type: none"> <li>Read this register to obtain average offset value [15:0] in fractional nanosecond, which is derived from the offset adjustment data [96:0] of the PTP slave.</li> <li>Bit [31:16]: Reserved.</li> </ul>
0x1030	32-bit <code>average_delay_fnsec_r</code> register: <ul style="list-style-type: none"> <li>Read this register to obtain average delay value [47:16] in fractional nanosecond, which is derived from the offset adjustment data [191:96] of the PTP slave.</li> </ul>
0x1031	32-bit <code>average_delay_fnsec_to_mem</code> register: <ul style="list-style-type: none"> <li>Read this register to obtain average delay value [15:0] in fractional nanosecond, which is derived from the offset adjustment data [191:96] of the PTP slave.</li> <li>Bit [31:16]: Reserved.</li> </ul>
<b>Variant: Multi-Channel</b>	
0x00000–0x30DFF	For multi-channel design examples, the base address of all channels are incremented with 0x10000. This corresponds to: <ul style="list-style-type: none"> <li>Channel 0 Range: 0x00300–00DFF</li> <li>Channel 1 Range: 0x10300–10DFF</li> <li>Channel 2 Range: 0x20300–20DFF</li> <li>Channel 3 Range: 0x30300–30DFF</li> </ul>
0x04000–0x37FFF	For multi-channel design examples, the base address of all channels are incremented with 0x10000. This corresponds to: <ul style="list-style-type: none"> <li>Channel 0 Range: 0x04000–0x07FFF</li> <li>Channel 1 Range: 0x14000–0x17FFF</li> <li>Channel 2 Range: 0x24000–0x27FFF</li> <li>Channel 3 Range: 0x34000–0x37FFF</li> </ul>

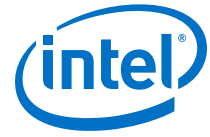


*Note:*

1. For Intel Stratix 10 H-tile production device, disable the background calibration prior to accessing the transceiver core reconfiguration register, as described in the *Disabling Background Calibration* section of the *25G Ethernet Intel FPGA IP User Guide*.
2. Dynamic reconfiguration switching for 10G/25G is not available for multi-channel designs.
3. For single-channel design example, 0x4000 is the base address of the PHY registers. For example, to read the background calibration register, type `reg_read 0x4542`.
4. For multi-channel design example, the base address of the PHY registers is  $0x4000 + (0x10000 * \langle link\ num \rangle)$ . For example, to read the background calibration register at channel 2, type `reg_read 0x24542`.

**Related Information**

- [Control, Status, and Statistics Register Descriptions section of the 25G Ethernet Intel FPGA IP User Guide](#)  
Describes the 25G Ethernet Intel FPGA IP control, status, and statistics registers.
- [Disabling Background Calibration section of the 25G Ethernet Intel FPGA IP User Guide](#)



## 6. 25G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide Archives

---

If an IP core version is not listed, the user guide for the previous IP core version applies.

IP Core Version	User Guide
18.1	<a href="#">25G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide</a>
18.0	<a href="#">25G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide</a>

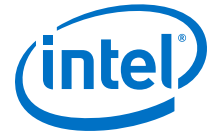
## 7. Document Revision History for the 25G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide

Document Version	Intel Quartus Prime Version	Changes
2019.05.10	19.1	<ul style="list-style-type: none"> <li>Renamed Altera Debug Master Endpoint (ADME) to Native PHY Debug Master Endpoint (NPDME).</li> <li>Updated the hardware testing steps for the following topics: <ul style="list-style-type: none"> <li><i>Test Procedure—Design Example Without the IEEE 1588v2 Feature</i> for 10G/25G Ethernet single-channel design example.</li> <li><i>Test Procedure—Design Example with the IEEE 1588v2 Feature</i> for 10G/25G Ethernet single-channel design example.</li> <li><i>Test Procedure—Design Example With and Without the IEEE 1588v2 Feature</i> for 25G Ethernet single-channel design example.</li> <li><i>Test Procedure</i> for 25G Ethernet multi-channel design example.</li> </ul> </li> <li>Updated the following Figures: <ul style="list-style-type: none"> <li><i>Block Diagram—10G/25G Ethernet Single-Channel Design Example (MAC+PCS+PMA Core Variant) Without the IEEE 1588v2 Feature</i></li> <li><i>Block Diagram—10G/25G Ethernet Single-Channel Design Example (MAC+PCS+PMA Core Variant) with the IEEE 1588v2 Feature</i></li> <li><i>Block Diagram—10G/25G Ethernet Single-Channel Design Example (MAC+PCS Core Variant) Without the IEEE 1588v2 Feature</i></li> <li><i>Block Diagram—10G/25G Ethernet Single-Channel Design Example (MAC+PCS Core Variant) with the IEEE 1588v2 Feature</i></li> </ul> </li> <li>Updated Table: <i>Hardware Design Example Register Map for 25G Ethernet Intel FPGA IP Core for Intel Stratix 10 Devices to include reconfiguration registers (0x04000-0x37FFF)</i>.</li> <li>Updated the note section in the <i>Design Example Registers</i> topic.</li> <li>Made editorial updates throughout the document.</li> </ul>
2019.01.07	18.1	<ul style="list-style-type: none"> <li>Updated the <i>Generating the Design Example</i> topic to correct target development kit in Step 8 from <b>Stratix 10 GX FPGA Development Kit</b> to <b>Intel Stratix 10 L-Tile GX Transceiver Signal Integrity Development Kit</b>.</li> <li>Updated Figure: <i>Block Diagram—10G/25G Ethernet Single-Channel Design Example (MAC+PCS+PMA Core Variant) with the IEEE 1588v2 Feature</i>.</li> <li>Added a note to <i>Design Example Registers</i> topic.</li> </ul>
2018.10.03	18.1	<ul style="list-style-type: none"> <li>Updated Table: <i>Parameters in the Example Design Tab</i> to update the description for <b>Select Board</b>.</li> <li>Updated the <i>Hardware and Software Requirements</i> topics for all design example chapters.</li> <li>Updated the <i>Design Components</i> topics for all design example chapters.</li> </ul>
<i>continued...</i>		



**7. Document Revision History for the 25G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide**

UG-20110 | 2019.05.10



Document Version	Intel Quartus Prime Version	Changes
		<ul style="list-style-type: none"> <li>• Added new Figures:                             <ul style="list-style-type: none"> <li>– Block Diagram—25G Ethernet Single-Channel Design Example (MAC +PCS Core Variant) Without the IEEE 1588v2 Feature</li> <li>– Block Diagram—25G Ethernet Single-Channel Design Example (MAC +PCS Core Variant) Without the IEEE 1588v2 Feature</li> <li>– Block Diagram—25G Ethernet Single-Channel Design Example (MAC +PCS Core Variant) Without the IEEE 1588v2 Feature</li> <li>– Block Diagram—25G Ethernet Single-Channel Design Example (MAC +PCS Core Variant) with the IEEE 1588v2 Feature</li> </ul> </li> <li>• Updated Figures:                             <ul style="list-style-type: none"> <li>– Example Design Tab in the 25G Ethernet Intel FPGA IP Parameter Editor</li> <li>– Block Diagram—25G Ethernet Single-Channel Design Example (MAC +PCS+PMA Core Variant) with the IEEE 1588v2 Feature</li> <li>– Block Diagram—25G Ethernet Single-Channel Design Example (MAC +PCS+PMA Core Variant) Without the IEEE 1588v2 Feature</li> <li>– Block Diagram—25G Ethernet Multi-Channel Design Example (MAC +PCS+PMA Core Variant)</li> <li>– Block Diagram—10G/25G Ethernet Single-Channel Design Example (MAC+PCS+PMA) Without the IEEE 1588v2 Feature</li> <li>– Block Diagram—10G/25G Ethernet Single-Channel Design Example (MAC+PCS+PMA) with the IEEE 1588v2 Feature</li> </ul> </li> <li>• Removed Figure: <i>Sample Simulation Output for Design Example with the IEEE 1588v2 Feature (Part 1 of 2)</i> and <i>Sample Simulation Output for Design Example with the IEEE 1588v2 Feature (Part 2 of 2)</i>.</li> <li>• Updated the simulation sample output of the <i>Test Case—Design Example with the IEEE 1588v2 Feature</i> topic for 25G Ethernet Single-Channel design example.</li> <li>• Updated the simulation sample output of the <i>Test Case—Design Example with the IEEE 1588v2 Feature</i> topic for the 10G/25G Ethernet design example.</li> <li>• Restructured descriptions for <i>Features</i> topics for all design example chapters.</li> <li>• Streamlined the contents and document organization.</li> </ul>
2018.06.25	18.0	Initial release.