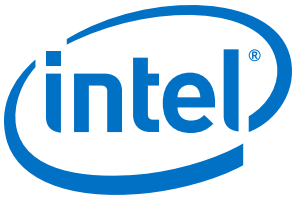


Intel® FPGA DisplayPort Design Example User Guide for Intel® Arria 10 Devices

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

UG-20075 | 2017.11.06
Latest document on the web: **PDF** | **HTML**

Subscribe
Send Feedback

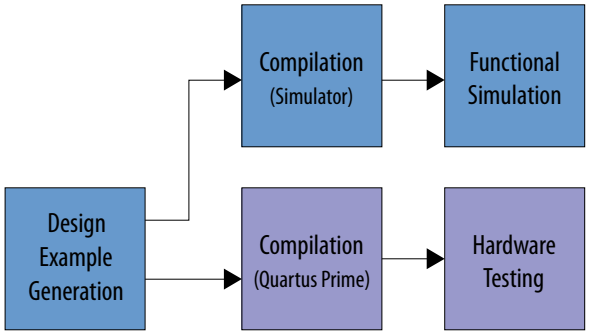
Contents

1 Intel® FPGA DisplayPort Design Example Quick Start Guide for Intel® Arria® 10 Devices

The Intel® FPGA DisplayPort IP core design example for Intel Arria® 10 devices features a simulating testbench and a hardware design that supports compilation and hardware testing.

When you generate a design example, the parameter editor automatically creates the files necessary to simulate, compile, and test the design in hardware.

Figure 1. Development Steps



Related Links

Intel FPGA DisplayPort IP Core User Guide

1.1 Directory Structure

The directories contain the generated files for the DisplayPort design example.

Figure 2. Directory Structure for the Design Example

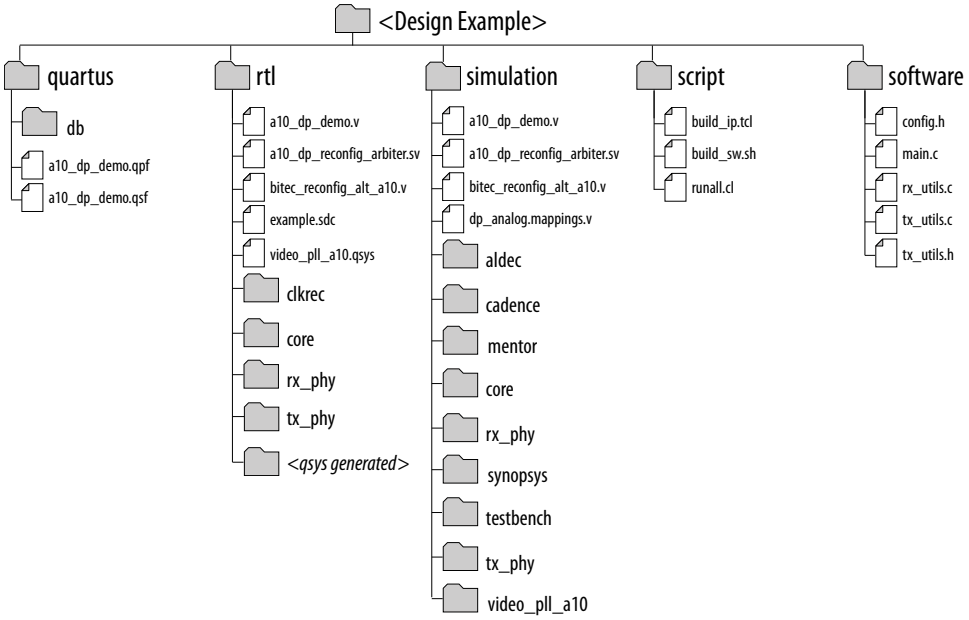


Table 1. Other Generated Files in RTL Folder

Folders	Files
clkrec	/altera_pll_reconfig_core.v
	/altera_pll_reconfig_mif_reader.v
	/altera_pll_reconfig_top.v
	/bitec_clkrec.qip
	/bitec_clkrec.sdc
	/bitec_clkrec.v
	/bitec_dp_add.v
	/bitec_dp_cdc.v
	/bitec_dp_cdc_fifo.v
	/bitec_dp_cdc_pulse.v
	/bitec_dp_cnt.v
	/bitec_dp_dcfifo.v
	/bitec_dp_dd.v
	/bitec_dp_div.v
	/bitec_dp_mult.v
	/bitec_fpll_calc.v
	/bitec_fpll_cntrl.v
	/bitec_fpll_reconf.v

continued...

Folders	Files
	/bitec_loop_cntrl.v
	/bitec_vsyngen.v
	• /clkrec_pll135_a10.qsys (Intel Quartus Prime Standard Edition) • /clkrec_pll135_a10.ip (Intel Quartus Prime Pro Edition)
	• /clkrec_pll_a10.qsys (Intel Quartus Prime Standard Edition) • /clkrec_pll1_a10.ip (Intel Quartus Prime Pro Edition)
	• /clkrec_reset_a10.qsys (Intel Quartus Prime Standard Edition) • /clkrec_reset_a10.ip (Intel Quartus Prime Pro Edition)
	<qsys generated folder>
core	/altera_avalon_i2c
	• /dp_core.qsys (Intel Quartus Prime Standard Edition) • /dp_core.ip (Intel Quartus Prime Pro Edition)
	• /dp_rx.qsys (Intel Quartus Prime Standard Edition) • /dp_rx.ip (Intel Quartus Prime Pro Edition)
	• /dp_tx.qsys (Intel Quartus Prime Standard Edition) • /dp_tx.ip (Intel Quartus Prime Pro Edition)
	<qsys generated folder>
rx_phy	• /gxb_rx.qsys (Intel Quartus Prime Standard Edition) • /gxb_rx.ip (Intel Quartus Prime Pro Edition)
	• /gxb_rx_reset.qsys (Intel Quartus Prime Standard Edition) • /gxb_rx_reset.ip (Intel Quartus Prime Pro Edition)
	/rx_phy_top.v
	<qsys generated folder>
tx_phy	• /gxb_tx.qsys (Intel Quartus Prime Standard Edition) • /gxb_tx.ip (Intel Quartus Prime Pro Edition)
	• /gxb_tx_fpll.qsys (Intel Quartus Prime Standard Edition) • /gxb_tx_fpll.ip (Intel Quartus Prime Pro Edition)
	/tx_phy_top.v
	<qsys generated folder>

Table 2. Other Generated Files in Simulation Folder

Folders	Files
aldec	/aldec.do
	/rivierapro_setup.tcl
cadence	/cds.lib
	/hdl.var
	/ncsim.sh
	/ncsim_setup.sh

continued...

Folders	Files
	<cds_libs folder>
core	/altera_avalon_i2c
	• /dp_core.qsys (Intel Quartus Prime Standard Edition) • /dp_core.ip (Intel Quartus Prime Pro Edition)
	• /dp_rx.qsys (Intel Quartus Prime Standard Edition) • /dp_rx.ip (Intel Quartus Prime Pro Edition)
	• /dp_tx.qsys (Intel Quartus Prime Standard Edition) • /dp_tx.ip (Intel Quartus Prime Pro Edition)
	<qsys generated folder>
mentor	/mentor.do
	/msim_setup.tcl
rx_phy	• /gxb_rx.qsys (Intel Quartus Prime Standard Edition) • /gxb_rx.ip (Intel Quartus Prime Pro Edition)
	/rx_phy_top.v
	• /gxb_rx_reset.qsys (Intel Quartus Prime Standard Edition) • /gxb_rx_reset.ip (Intel Quartus Prime Pro Edition)
	<qsys generated folder>
synopsys	/vcs/filelist.f
	/vcs/vcs_setup.sh
	/vcs/vcs_sim.sh
	/vcsmx/synopsys_sim_setup
	/vcsmx/vcsmx_setup.sh
	/vcsmx/vcsmx_sim.sh
testbench	/a10_dp_harness.sv
	/clk_gen.v
	/freq_check.v
	/rx_freq_check.v
	/tx_freq_check.v
	/vga_driver.v
tx_phy	• /gxb_tx.qsys (Intel Quartus Prime Standard Edition) • /gxb_tx.ip (Intel Quartus Prime Pro Edition)
	<qsys generated folder>
	• /gxb_tx_fpll.qsys (Intel Quartus Prime Standard Edition) • /gxb_tx_fpll.ip (Intel Quartus Prime Pro Edition)
	/tx_phy_top.v

1.2 Hardware and Software Requirements

Intel uses the following hardware and software to test the design example.

Hardware

- Intel Arria 10 GX FPGA Development Kit
- DisplayPort Source (Graphics Processor Unit (GPU))
- DisplayPort Sink (Monitor)
- Bitec DisplayPort FMC daughter card (Revision 5.0)
- DisplayPort cables

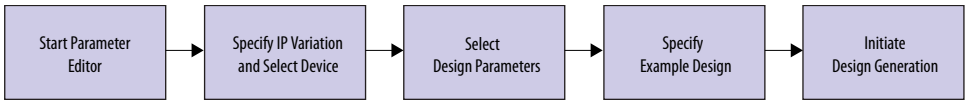
Software

- Intel Quartus Prime (for hardware testing)
- ModelSim* - Intel FPGA Edition, ModelSim - Intel FPGA Edition Starter Edition, NCSim (Verilog only), Riviera-Pro, or VCS (Verilog only)/VCS-MX simulator

1.3 Generating the Design

Use the Intel FPGA DisplayPort parameter editor in the Intel Quartus Prime software to generate the design example.

Figure 3. Generating the Design Flow



1. Click **Tools ➤ IP Catalog**, and select Intel Arria 10 as the target device family.

   Note: The design example only support Intel Arria 10 devices.
2. In the IP Catalog, locate and double-click **Intel FPGA DisplayPort** . The **New IP Variation** window appears.
3. Specify a top-level name for your custom IP variation. The parameter editor saves the IP variation settings in a file named <*your_ip*>.ip or <*your_ip*>.qsys.
4. You may select a specific Intel Arria 10 device in the **Device** field, or keep the default Intel Quartus Prime software device selection.
5. Click **OK**. The parameter editor appears.
6. Configure the desired parameters for both TX and RX.

   Note: The DisplayPort design example generation flow supports only SST. Selecting the **Support MST** parameter prevents you from generating the example design.
7. On the **Design Example** tab, select **DisplayPort SST Parallel Loopback With PCR** or **DisplayPort SST Parallel Loopback Without PCR**.
8. Select **Simulation** to generate the testbench, and select **Synthesis** to generate the hardware design example.

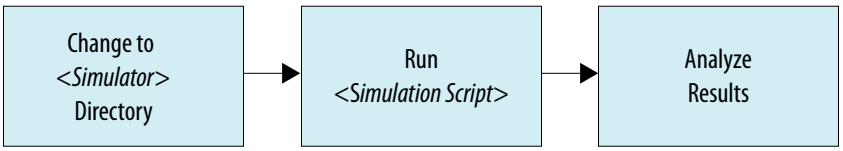
You must select at least one of these options to generate the design example files. If you select both, the generation time is longer.

9. For **Target Development Kit**, select **Arria 10 GX FPGA Development Kit**. If you select the development kit, then the target device (selected in **step 4**) changes to match the device on the development kit. For **Arria 10 GX FPGA Development Kit**, the default device is 10AX115S2F45I1SG.
10. Click **Generate Example Design**.

1.4 Simulating the Design

The DisplayPort design example testbench simulates a serial loopback design from a TX instance to an RX instance. An internal video pattern generator module drives the DisplayPort TX instance and the RX instance video output connects to CRC checkers in the testbench.

Figure 4. Design Simulation Flow



1. Navigate to the simulation folder of your choice.
2. Run the simulation script for the supported simulator. The script compiles and runs the testbench in the simulator.
3. Analyze the results.

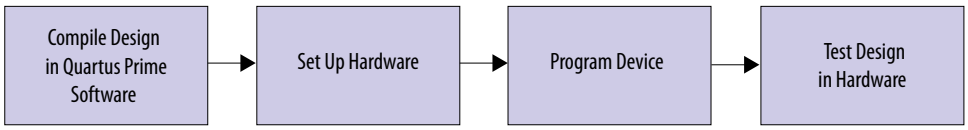
Table 3. Steps to Run Simulation

Simulator	Working Directory	Instructions
Riviera-Pro	/simulation/aldec	In the command line, type `vsim -c -do aldec.do`
NCSim	/simulation/cadence	In the command line, type `source ncsim.sh`
ModelSim	/simulation/mentor	In the command line, type `vsim -c -do mentor.do`
VCS	/simulation/synopsys/vcs	In the command line, type `source vcs_sim.sh`
VCS-MX	/simulation/synopsys/vcsmx	In the command line, type `source vcsmx_sim.sh`

A successful simulation ends with the following message:

```
# SINK CRC_R = ac9c, CRC_G = ac9c, CRC_B = ac9c,
# SOURCE CRC_R = ac9c, CRC_G = ac9c, CRC_B = ac9c,
# Pass: Test Completed
```

1.5 Compiling and Testing the Design



To compile and run a demonstration test on the hardware example design, follow these steps:

1. Ensure hardware example design generation is complete.
2. Launch the Intel Quartus Prime software and open <*project directory*>/quartus/a10_dp_demo.qpf.

   Note: The Bitec DisplayPort FMC daughter card revision 10 has schematic changes compared to revision 8 and earlier. To support all revisions, the design example top level RTL file at <*project directory*>/rtl/a10_dp_demo.v and the software config.h file include a local parameter for you to select the FMC revision.

   ```
   localparam BITEC_DP_CARD_REV = 0;
   // 0 = Bitec FMC DP card rev.4 - 8,
   // 1 = rev.9 or later

   in <project>/software/config.h:
   #define BITEC_DP_CARD_REV 0
   // set to 0 = Bitec FMC DP card rev.4 - 8
   // set to 1 = Bitec FMC DP card rev.9 or later
   ```

   The default value is 0. If the config.h file is updated, you must run build_sw.sh in the script folder before compiling the Intel Quartus Prime project to ensure the software is effective.
3. Click **Processing ➤ Start Compilation**.
4. After successful compilation, the Intel Quartus Prime software generates a .sof file in your specified directory.
5. Connect the DisplayPort RX connector on the Bitec daughter card to an external video source, such as the graphics card on a PC.
6. Connect the DisplayPort TX connector on the Bitec daughter card to a video analyzer or a DisplayPort sink device, such as a PC monitor.
7. Ensure all switches on the development board are in default position.
8. Configure the selected Intel Arria 10 device on the development board using the generated .sof file (**Tools ➤ Programmer**).
9. The DisplayPort sink device displays the video generated from the video source.

Related Links

- Intel Arria 10 FPGA Development Kit User Guide

- AN793: Arria 10 DisplayPort 4Kp60 with Video and Image Processing Pipeline Retransmit Reference Design Provides more information
  Provides more information about Intel's video connectivity, the DisplayPort Sink (RX) and Source (TX) functions using a video loop-through system.

1.5.1 Regenerating ELF File

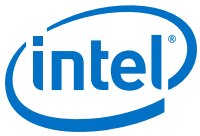
1. Go to <*project directory*>/software and edit the code if necessary.
2. Go to <*project directory*>/script and execute the following build script:
   ```
   source build_sw.sh
   ```
3. Make sure an .elf file is generated in <*project directory*>/software/dp_demo.
4. Download the generated .elf file into the FPGA without recompiling the .sof file by running the following script:
   ```
   nios2-download <project directory>/software/dp_demo/*.elf
   ```
5. Push the reset button on the FPGA board for the new software to take effect.

1.6 Intel FPGA DisplayPort Design Example Parameters

Table 4. DisplayPort Design Example Parameters

These options are available only for Intel Arria 10 devices.

Parameter	Value	Description
Available Design Example		
Select Design	None, DisplayPort SST Parallel Loopback with PCR, DisplayPort SST Parallel Loopback without PCR	Select the design example to be generated. • None: No design example is available for the current parameter selection • DisplayPort SST Parallel Loopback with PCR: This design example demonstrates parallel loopback from DisplayPort sink to DisplayPort source through a Pixel Clock Recovery (PCR) module when you turn off the **Enable Video Input Image Port** parameter. • DisplayPort SST Parallel Loopback without PCR: This design example demonstrates parallel loopback from DisplayPort sink to DisplayPort source without a Pixel Clock Recovery (PCR) module when you turn on the **Enable Video Input Image Port** parameter. *Note:* DisplayPort SST Parallel Loopback without PCR design example is available only in the Intel Quartus Prime Pro Edition.
Design Example Files		
Simulation	On, Off	Turn on this option to generate the necessary files for the simulation testbench.
Synthesis	On, Off	Turn on this option to generate the necessary files for Intel Quartus Prime compilation and hardware demonstration.
Generated HDL Format		
Generate File Format	Verilog, VHDL	Select your preferred HDL format for the generated design example fileset.



Generated HDL Format		
		Note: This option only determines the format for the generated top level IP files. All other files (e.g. example testbenches and top level files for hardware demonstration) are in Verilog HDL format.

Target Development Kit		
Select Board	No Development Kit, Arria 10 GX FPGA Development Kit, Custom Development Kit	Select the board for the targeted design example. • No Development Kit: This option excludes all hardware aspects for the design example. The IP core sets all pin assignments to virtual pins. • Intel Arria 10 GX FPGA Development Kit: This option automatically selects the project's target device to match the device on this development kit. You may change the target device using the **Change Target Device** parameter if your board revision has a different device variant. The IP core sets all pin assignments according to the development kit. • Custom Development Kit: This option allows the design example to be tested on a third-party development kit with an Intel FPGA. You may need to set the pin assignments on your own.

Target Device		
Change Target Device	On, Off	Turn on this option and select the preferred device variant for the development kit.

2 Intel FPGA DisplayPort Design Example Detailed Description

The Intel FPGA DisplayPort IP core design example demonstrates parallel loopback from DisplayPort RX instance to DisplayPort TX instance.

Table 5. Intel FPGA DisplayPort Design Example for Intel Arria 10 Devices

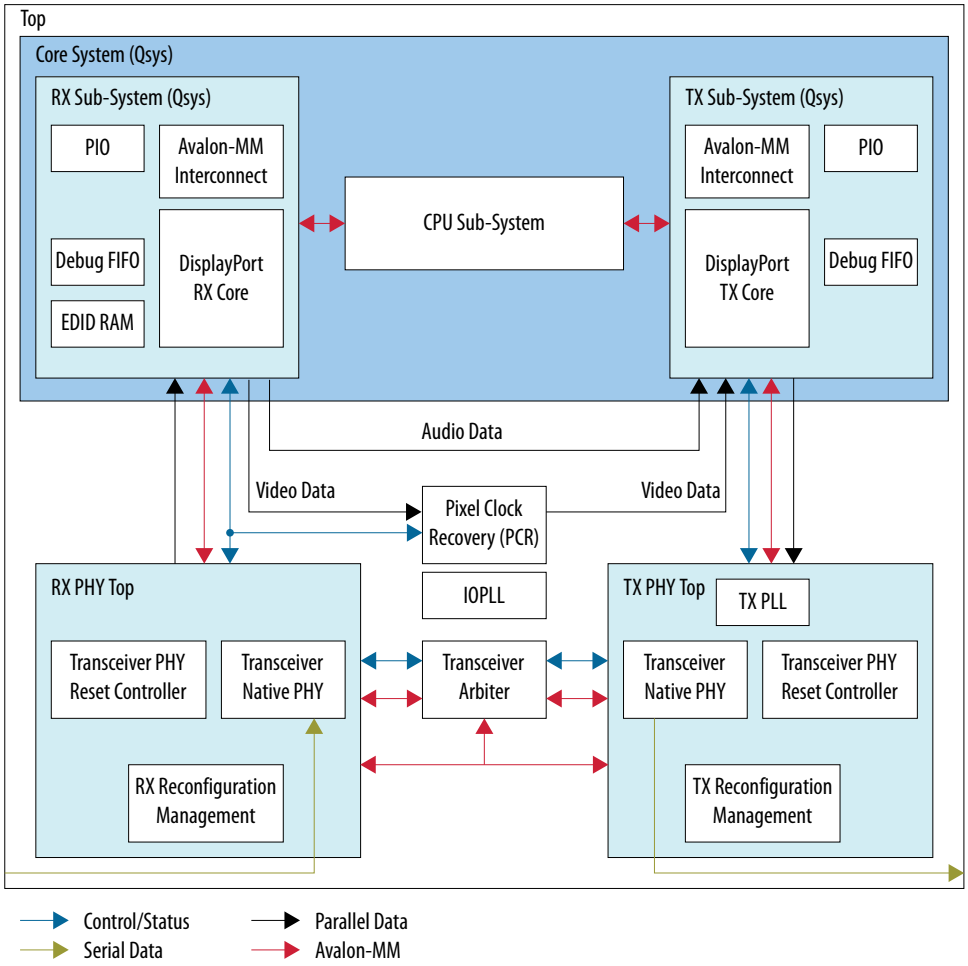
Design Example	Designation	Data Rate	Channel Mode	Loopback Type
DisplayPort SST parallel loopback with PCR	DisplayPort SST	HBR3, HBR2, HBR, and RBR	Simplex	Parallel with PCR
DisplayPort SST parallel loopback without PCR	DisplayPort SST	HBR3, HBR2, HBR, and RBR	Simplex	Parallel without PCR

Note: DisplayPort SST parallel loopback without PCR design example and support for HBR3 are available only in the Intel Quartus Prime Pro Edition software.

2.1 Intel Arria 10 DisplayPort SST Parallel Loopback

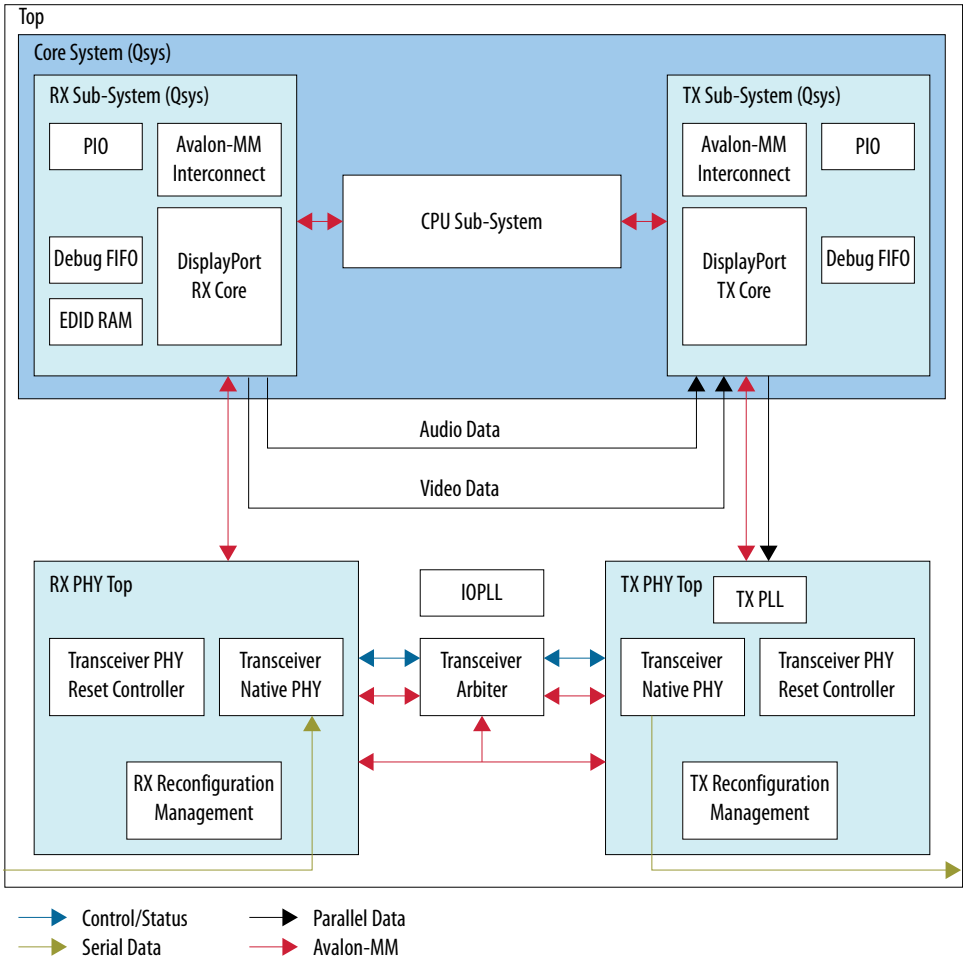
The parallel loopback design examples demonstrate the transmission of a single video stream from DisplayPort sink to DisplayPort source with or without a Pixel Clock Recovery (PCR).

Figure 5. Intel Arria 10 DisplayPort SST Parallel Loopback with PCR



- In this variant, the DisplayPort source's parameter, **TX_SUPPORT_IM_ENABLE**, is turned off and the standard VSYNC/HSYNC/DE video interface is used.
- The DisplayPort sink receives video and or audio streaming from external video source such as GPU and decodes it into parallel video interface.
- The IOPLL drives the video clock at a fixed frequency (in this case, 160 MHz).
- If DisplayPort sink's **MAX_LINK_RATE** is configured to **HBR2** and **PIXELS_PER_CLOCK** is configured to **Dual**, the video clock runs at 300 MHz to support 4Kp60 pixel rate (594/2 = 297 MHz). Otherwise, the video clock runs at 160 MHz.
- The design uses the pixel recovery clock (PCR) to recover the pixel clock according to the received MSA information from the sink and converts the RX parallel video interface to the standard VSYNC/HSYNC/DE interface.
- The PCR output drives the source video interface and encodes to the DisplayPort main link before transmitting to the monitor.
- The recovered clock drives the TX video clock.

Figure 6. Intel Arria 10 DisplayPort SST Parallel Loopback without PCR

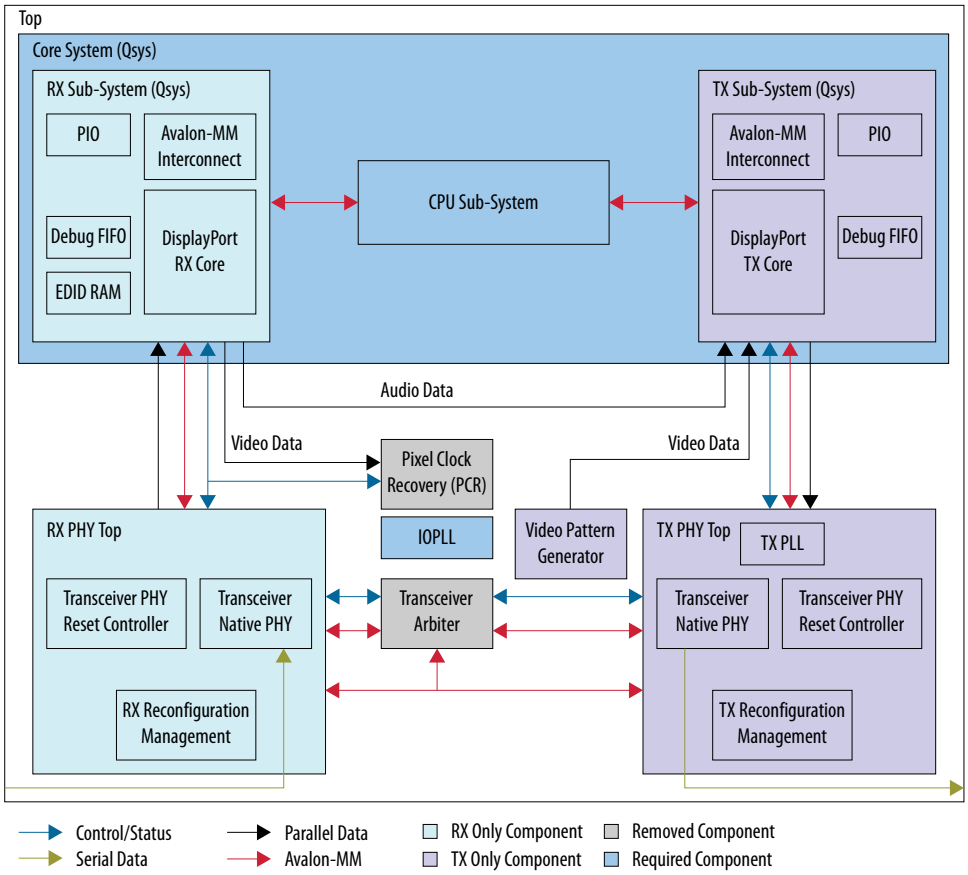


- In this variant, the DisplayPort source's parameter, **TX_SUPPORT_IM_ENABLE**, is turned on ("1") and the video image interface is used.
- The DisplayPort sink receives video and or audio streaming from external video source such as GPU and decodes it into parallel video interface.
- The DisplayPort sink video output directly drives the DisplayPort source video interface and encodes to the DisplayPort main link before transmitting to the monitor.
- The IOPLL drives both the DisplayPort sink and source video clocks at a fixed frequency.
- If DisplayPort sink and source's **MAX_LINK_RATE**parameter is configured to **HBR2** and **PIXELS_PER_CLOCK** is configured to **Dual**, the video clock runs at 300 MHz to support 4Kp60 pixel rate (594/2 = 297 MHz). Otherwise, the video clock runs at 160 MHz.

Table 6. Design Example Variant Comparison

Design Example	PCR Module	Enable Video Image Interface	Adaptive Sync	Video Interface
DisplayPort SST parallel loopback with PCR	Required	No	Not supported	Standard VSYNC/ HSYNC/DE interface (`txN_video_in`)
DisplayPort SST parallel loopback without PCR	Not required	Yes	Supported	Video Image Interface (`txN_video_in_im`)

Figure 7. Components Required for RX or TX Only Design



To use RX or TX only components:

- Remove the irrelevant blocks from the design.
- Edit the `config.h` file in the software folder to specify if DP_SUPPORT_RX and DP_SUPPORT_TX is 1 0r 0. The default setting for both parameters is 1.
  - For TX-only design, set DP_SUPPORT_RX and `BITEC_RX_GPUMODE` to 0.
  - For RX-only design, set DP_SUPPORT_TX to 0.

Table 7. RX-Only and TX-Only Design Requirements

User Requirement	Preserve	Remove	Add
DisplayPort RX Only	RX PHY Top; Core System consists of: • RX sub-system • CPU sub-system	• TX Top • PCR (if not needed) • Transceiver Arbiter	—
DisplayPort TX Only	TX PHY Top; Core System consists of: • TX sub-system • CPU sub-system	• RX Top • PCR • Transceiver Arbiter	Video Pattern Generator

Related Links

Jitter of PLL Cascading or Non-Dedicated Clock Path for Arria 10 PLL Reference Clock
Sourcing reference clock from a cascaded PLL output, global clock, or core clock network will introduce additional jitter to the PLL output. Refer to this solution for workaround if your design clocks experience additional jitter.

2.2 Design Components

The Intel FPGA DisplayPort IP core design example requires these components.

Table 8. Core System Components

Module	Description
Core System (Platform Designer)	The core system consists of the Nios II Processor and its necessary components, DisplayPort RX and TX core sub-systems. This system provides the infrastructure to interconnect the Nios II processor with the Intel FPGA DisplayPort IP core (RX and TX instances) through Avalon Memory Mapped (Avalon-MM) interface within a single Platform Designer system to ease the software build flow. This system consists of: • CPU Sub-System • RX Sub-System • TX Sub-System
RX Sub-System (Platform Designer)	The RX sub-system consists of: • Clock Source—The clock source to the DisplayPort RX core. This sub-system has two clock sources integrated: 100 MHz and 16 MHz. • Reset Bridge—The bridge that connects the external signal to the sub-system. This bridge synchronizes to the respective clock source before it is used. • DisplayPort RX Core—DisplayPort Sink IP core, *VESA DisplayPort Standard version 1.4*. • Debug FIFO—This FIFO captures all DisplayPort RX auxiliary cycles, and prints out in the Nios II Debug terminal. • PIO—The parallel IO that triggers the MSA capture, and prints out when the on-board push button (PB) is pressed. • Avalon-MM Pipeline Bridge—This Avalon-MM bridge interconnects the Avalon-MM interface between components within the RX sub-system to the Nios II processor in the Core sub-system. • EDID—The EDID RAM is only used to store the desired EDID value in the RAM and connect to the DisplayPort Sink IP core. This component is only used when you disable the **Enable GPU Control** option in the RX core.
TX Sub-System (Platform Designer)	The TX sub-system consists of:

continued...

Module	Description
	• Clock Source—The clock source to the DisplayPort TX core. This sub-system has two clock sources integrated: 100 MHz and 16 MHz. • Reset Bridge—The bridge that connects the external signal to the sub-system. This bridge synchronizes to the respective clock source before it is used. • DisplayPort TX Core—DisplayPort Source IP core, *VESA DisplayPort Standard version 1.4*. • Debug FIFO—This FIFO captures all DisplayPort TX auxiliary cycles, and prints out in the Nios II Debug terminal. This component is only used when the TX_AUX_DEBUG parameter is turned on. • PIO—The parallel IO that triggers the DPTX register update in software (`tx_utils.c`). • Avalon-MM Pipeline Bridge—This Avalon-MM bridge interconnects the Avalon-MM interface between components within the TX sub-system to the Nios II processor in the Core sub-system.

Table 9. DisplayPort RX PHY Top and TX PHY Top Components

Module	Description
RX PHY Top	The RX PHY top level consists of the components related to the receiver PHY layer. • Transceiver Native PHY (RX)—The hard transceiver block that receives the serial data from an external video source and deserializes it to 20-bit or 40-bit parallel data to the DisplayPort Sink IP core. This block supports up to 8.1 Gbps (HBR3) data rate with 4 channels. • Transceiver PHY Reset Controller—The RX Reconfiguration Management module triggers the reset input of this controller to generate the corresponding analog and digital reset signals to the Transceiver Native PHY block according to the reset sequencing. • RX Reconfiguration Management—This block reconfigures and recalibrates the Transceiver Native PHY block to receive serial data in the supported data rates (RBR, HBR, HBR2, and HBR3). *Note:* 8.1 Gbps is available only in the Intel Quartus Prime Pro Edition software.
TX PHY Top	The TX PHY top level consists of the components related to the transmitter PHY layer. • Transceiver Native PHY(TX)—The hard transceiver block that receives 20-bit or 40-bit parallel data from the Intel FPGA DisplayPort IP core and serializes the data before transmitting it. This block supports up to 8.1 Gbps (HBR3) data rate with 4 channels. *Note:* You must set the TX channel bonding mode to **PMA and PCS bonding** and the **PCS TX Channel bonding master** parameter to 0 (default is auto). • Transceiver PHY Reset Controller—The TX Reconfiguration Management module triggers the reset input of this controller to generate the corresponding analog and digital reset signals to the Transceiver Native PHY block according to the reset sequencing. • TX Reconfiguration Management—This block reconfigures and recalibrates the Transceiver Native PHY block to transmit serial data in the required data rates (RBR, HBR, HBR2, and HBR3). • TX PLL—The transmitter PLL block provides a fast serial fast clock to the Transceiver Native PHY block. If you need to use the PLL across multiple transceiver channels, you can move the TX PLL out of the TX PHY top module. For the Intel FPGA DisplayPort IP core design example, Intel uses transmitter fractional PLL (FPLL). *Note:* 8.1 Gbps is available only in the Intel Quartus Prime Pro Edition software.

Table 10. Loopback Top Component

Module	Description
Pixel Clock Recovery (PCR)	This module recovers pixel clock (derived from the DisplayPort Sink MSA information). PCR dynamically detects the received video format and recovers the corresponding pixel clock. This module also integrates a DCFIFO as video data buffer from the receiver and transmitter clock domains. This module supports resolutions up to 4Kp60 only. *Note:* Your design may not require PCR if you use your own recovery logic or any of the Video and Image Processing (VIP) IP cores.

Table 11. Top-Level Common Blocks

Module	Description
Transceiver Arbiter	This generic functional block prevents transceivers from recalibrating simultaneously when either RX or TX transceivers within the same physical channel require reconfiguration. The simultaneous recalibration impacts applications where RX and TX transceivers within the same channel are assigned to independent IP implementations. This transceiver arbiter is an extension to the resolution recommended for merging simplex TX and simplex RX into the same physical channel. This transceiver arbiter also assists in merging and arbitrating the Avalon-MM RX and TX reconfiguration requests targeting simplex RX and TX transceivers within a channel as the reconfiguration interface port of the transceivers can only be accessed sequentially. The transceiver arbiter is not required when only either RX or TX transceiver is used in a channel. The transceiver arbiter identifies the requester of a reconfiguration through its Avalon-MM reconfiguration interfaces and ensures that the corresponding `tx_reconfig_cal_busy` or `rx_reconfig_cal_busy` is gated accordingly.
IOPLL	IOPLL generates common source clock: `dp_rx_vid_clkout` and `clk_16` (16 MHz) for the DisplayPort system. • `dp_rx_vid_clkout`—used as RX core video clock of video data stream and PCR video input clock. • `clk_16`—Used as DisplayPort auxiliary clock and PCR reference clock.

2.3 Clocking Scheme

The clocking scheme illustrates the clock domains in the Intel FPGA DisplayPort IP core design example.

Figure 8. Intel FPGA DisplayPort Design Example Clocking Scheme

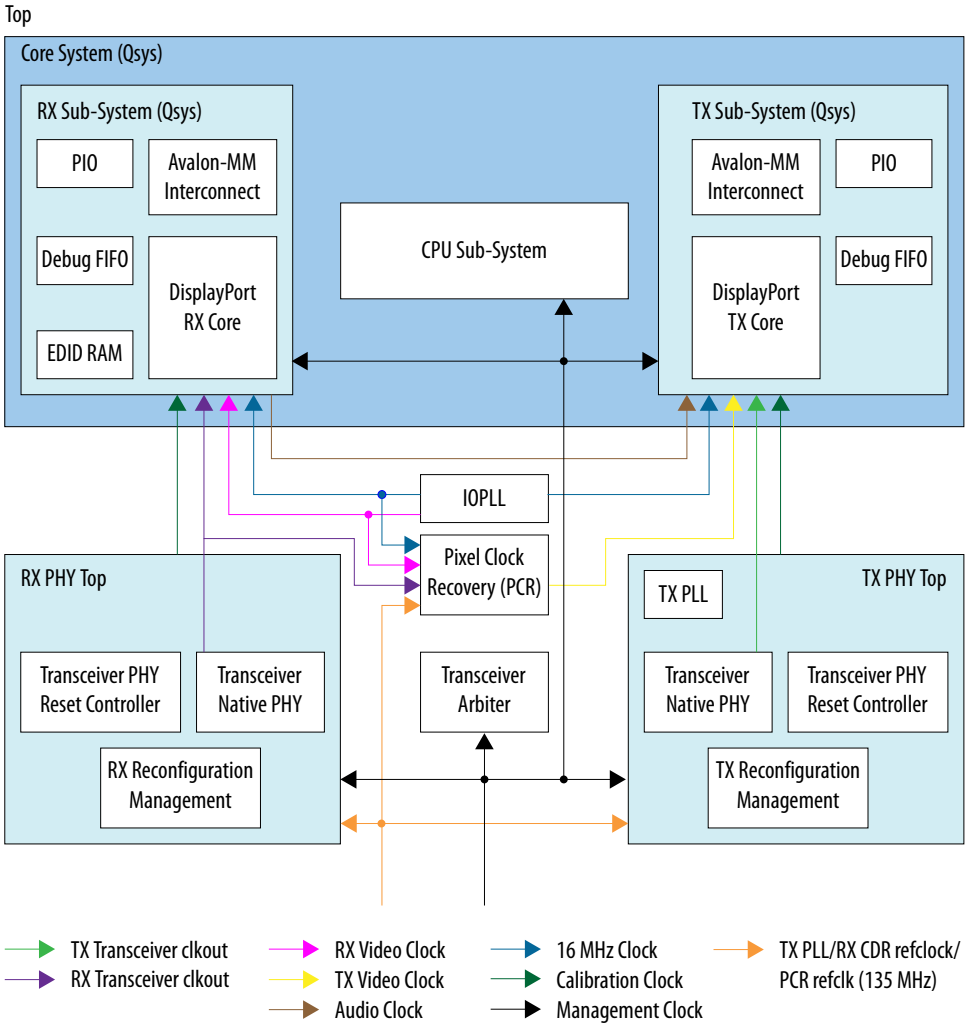


Table 12. Clocking Scheme Signals

Clock	Signal Name in Design	Description		
TX PLL Refclock	`tx_pll_refclk`	135 MHz TX PLL reference clock, that is divisible by the transceiver for all DisplayPort data rates (1.62 Gbps, 2.7 Gbps, and 5.4 Gbps). *Note:* The reference clock source of the TX PLL refclock is located at the HSSI `refclk` pin.		
TX Transceiver Clockout	`gxb_tx_clkout`	TX clock recovered from the transceiver, and the frequency varies depending on the data rate and symbols per clock.		
		Data Rate	**Symbols per Clock**	**Frequency (MHz)**
		RBR (1.62 Gbps)	2 (dual)	81
			4 (quad)	40.5
		HBR (2.7 Gbps)	2 (dual)	135

continued...

<table>
<tr><th>Clock</th><th>Signal Name in Design</th><th colspan="3">Description</th></tr>
<tr><td rowspan="5"></td><td rowspan="5"></td><td></td><td>4 (quad)</td><td>62.5</td></tr>
<tr><td rowspan="3">HBR2 (5.4 Gbps)</td><td>2 (dual)</td><td>270</td></tr>
<tr><td>4 (quad)</td><td>135</td></tr>
<tr><td>HBR3 (8.1 Gbps)</td><td>4 (quad)</td><td>202.5</td></tr>
<tr><td colspan="3"></td></tr>
<tr><td>TX PLL Serial Clock</td><td>gxb_tx_bonding_clocks</td><td colspan="3">Serial fast clock generated by TX PLL. The clock frequency is set based on the data rate.</td></tr>
<tr><td>RX Refclock</td><td>rx_cdr_refclk</td><td colspan="3">135 MHz transceiver clock data recovery (CDR) reference clock, that is divisible by all DisplayPort data rates (1.62 Gbps, 2.7 Gbps, and 5.4 Gbps).
Note: The reference clock source of the RX refclock is located at the HSSI refclk pin.</td></tr>
<tr><td rowspan="11">RX Transceiver Clkout</td><td rowspan="11">gxb_rx_clkout</td><td colspan="3">RX clock recovered from the transceiver, and the frequency varies depending on the data rate and symbols per clock.</td></tr>
<tr><td>Data Rate</td><td>Symbols per Clock</td><td>Frequency (MHz)</td></tr>
<tr><td rowspan="2">RBR (1.62 Gbps)</td><td>2 (dual)</td><td>81</td></tr>
<tr><td>4 (quad)</td><td>40.5</td></tr>
<tr><td rowspan="2">HBR (2.7 Gbps)</td><td>2 (dual)</td><td>135</td></tr>
<tr><td>4 (quad)</td><td>62.5</td></tr>
<tr><td rowspan="2">HBR2 (5.4 Gbps)</td><td>2 (dual)</td><td>270</td></tr>
<tr><td>4 (quad)</td><td>135</td></tr>
<tr><td>HBR3 (8.1 Gbps)</td><td>4 (quad)</td><td>202.5</td></tr>
<tr><td colspan="3"></td></tr>
<tr><td rowspan="5">Management Clock</td><td rowspan="5">rx_rcfg_mgmt_clk
tx_rcfg_mgmt_clk</td><td colspan="3">A free running 100 MHz clock for both Avalon-MM interfaces for reconfiguration and PHY reset controller for transceiver reset sequence.</td></tr>
<tr><td colspan="2">Component</td><td>Required Frequency (MHz)</td></tr>
<tr><td colspan="2">Avalon-MM reconfiguration</td><td>100 – 125</td></tr>
<tr><td colspan="2">Transceiver PHY reset controller</td><td>1 – 500</td></tr>
<tr><td colspan="3"></td></tr>
<tr><td>Audio Clock</td><td>dp_audio_clk</td><td colspan="3">DisplayPort audio clock.</td></tr>
<tr><td>16 MHz Clock</td><td>clk_16</td><td colspan="3">160 MHz clock used to encode and decode auxiliary channel in the Intel FPGA DisplayPort source and sink IP cores. This clock is also used as a reference clock in the Pixel Clock module for fractional calculation.</td></tr>
<tr><td>Calibration Clock</td><td>dp_rx_clk_cal
dp_tx_clk_cal</td><td colspan="3">A 50 MHz calibration clock input that must be synchronous to the Transceiver Reconfiguration module's clock. This clock is used in the Intel FPGA DisplayPort IP core's reconfiguration logic.</td></tr>
<tr><td>RX Video Clock</td><td>dp_rx_vid_clkout</td><td colspan="3">Video clock for DisplayPort sink to clock video data stream.</td></tr>
</table>

continued...

Clock	Signal Name in Design	Description
		If MAX_LINK_RATE = HBR2 and PIXELS_PER_CLOCK = Dual, video clock uses 300 MHz. Otherwise, fixed to 160 MHz.
TX Video Clock	tx_vid_clk	Recovered video clock from the PCR module that reflects the actual video clock frequency. Used when DisplayPort source's TX_SUPPORT_IM_ENABLE = 0.
TX IM Clock	tx_im_clk	Video clock for DisplayPort source to clock video data stream. Must be the same as the RX video clock in this design. Used when DisplayPort source's TX_SUPPORT_IM_ENABLE = 1.

2.4 Interface Signals and Parameter

The tables list the signals and parameter for the Intel FPGA DisplayPort IP core design example.

Table 13. Top-Level Signals

Signal	Direction	Width	Description
On-board Oscillator Signal			
refclk1_p	Input	1	100 MHz clock source used as IOPLL reference clock and Avalon-MM management clock
User Push Buttons and LEDs			
user_pb[0]	Input	1	Push button to trigger MSA print out during debug
cpu_resetn	Input	1	Global reset
user_led_g	Output	8	Green LED display *Note:* Refer to Hardware Setup on page 32 for the on-board user LED functions.
DisplayPort FMC Daughter Card Pins on FMC Port A			
fmca_gbtclk_m2c_p	Input	1	135 MHz dedicated transceiver reference clock from FMC port A
fmca_dp_m2c_p	Input	*N*	DisplayPort RX serial data *Note: N* = RX maximum lane count
fmca_dp_c2m_p	Output	*N*	DisplayPort TX serial data *Note: N* = TX maximum lane count
fmca_la_tx_p_10	Input	1	DisplayPort RX cable detect • 1 = Cable detected • 0 = Cable not detected
fmca_la_rx_n_8	Input	1	DisplayPort RX power detect • 1 = Power not detected • 0 = Power detected
fmca_la_tx_n_9	Input	1	DisplayPort RX Aux In
fmca_la_rx_n_6	Output	1	DisplayPort RX Aux Out
fmca_la_tx_p_9	Output	1	DisplayPort RX Aux OE

continued...

DisplayPort FMC Daughter Card Pins on FMC Port A			
fmca_la_rx_p_6	Output	1	DisplayPort RX HPD • 1 = HPD asserted • 0 = HPD deasserted
fmca_la_rx_n_9	Input	1	DisplayPort TX HPD • 1 = HPD asserted • 0 = HPD deasserted
fmca_la_tx_p_12	Input	1	DisplayPort TX Aux In
fmca_la_rx_p_10	Output	1	DisplayPort TX Aux Out
fmca_la_rx_n_10	Output	1	DisplayPort TX Aux OE
fmca_la_tx_n_12	Output	1	FMC card TX CAD

Interface to Parade Tech PS8460 Retimer			
fmca_la_tx_p_0	Inout	1	PS8460_SDA
fmca_la_tx_n_0	Inout	1	PS8460_SCL
fmca_la_rx_p_0	Output	1	PS8460_EQ0
fmca_la_rx_n_0	Output	1	PS8460_EQ1
fmca_la_tx_p_1	Output	1	PS8460_PDN
fmca_la_tx_n_1	Output	1	PS8460_CFG0
fmca_la_tx_p_2	Output	1	PS8460_CFG1
fmca_la_tx_n_2	Output	1	PS8460_CFG2

Table 14. Intel FPGA DisplayPort IP Core Signals (Platform Designer System)

Signal	Direction	Width	Description
Clock and Reset			
clk_100_in_clk	Input	1	100 MHz clock to CPU sub-system
cpu_reset_bridge_in_reset_n	Input	1	Reset to CPU sub-system (active low)

DisplayPort RX Signals			
dp_rx_reset_bridge_in_reset_n	Input	1	Reset to RX sub-system (active low)
dp_rx_clk_16_in_clk	Input	1	RX Auxiliary clock (16 MHz)
dp_rx_dp_sink_clk_cal	Input	1	RX reconfiguration calibration clock
dp_rx_pio_0_in_port	Input	1	Push button IO for debug purpose
dp_rx_dp_sink_rx_audio_valid	Output	1	RX Audio Interface *Note: M* = RX audio channel
dp_rx_dp_sink_rx_audio_mute	Output	1	

continued...

DisplayPort RX Signals			
dp_rx_dp_sink_rx_audio_infoframe	Output	40	
dp_rx_dp_sink_rx_audio_lpcm_data	Output	*M**32	
dp_rx_dp_sink_rx_aux_in	Input	1	RX auxiliary interface
dp_rx_dp_sink_rx_aux_out	Output	1	
dp_rx_dp_sink_rx_aux_oe	Output	1	
dp_rx_dp_sink_rx_hpd	Output	1	RX HPD
dp_rx_dp_sink_rx_cable_detect	Input	1	RX cable detect (active high)
dp_rx_dp_sink_rx_pwr_detect	Input	1	RX power detect (active high)
dp_rx_dp_sink_rx_msa	Output	217	DisplayPort RX MSA
dp_rx_dp_sink_rx_lane_count	Output	5	DisplayPort RX lane count
dp_rx_dp_sink_rx_link_rate	Output	2	RX Link Rate 2-bit indicator, used in PCR • RBR: 2'b00 • HBR: 2'b01 • HBR2: 2'b10 • HBR3: 2'b11
dp_rx_dp_sink_rx_link_rate_8bits	Output	8	RX Link Rate 8-bit indicator, used in transceiver reconfiguration management • RBR: 0x06 • HBR: 0x0A • HBR2: 0x14 • HBR3: 0x1E
dp_rx_dp_sink_rx_ss_valid	Output	1	DisplayPort RX secondary stream interface
dp_rx_dp_sink_rx_ss_data	Output	160	
dp_rx_dp_sink_rx_ss_sop	Output	1	
dp_rx_dp_sink_rx_ss_eop	Output	1	
dp_rx_dp_sink_rx_ss_clk	Output	1	
dp_rx_dp_sink_rx_stream_valid	Output	1	RX post scrambler stream data. For debug purpose. *Note: S* = RX symbols per clock
dp_rx_dp_sink_rx_stream_clk	Output	1	
dp_rx_dp_sink_rx_stream_data	Output	*S**32	

continued...

DisplayPort RX Signals			
dp_rx_dp_sink_rx_stream_ctrl	Output	*S*4	
dp_rx_dp_sink_rx_vid_clk	Input	1	DisplayPort RX video stream interface. *Note: B* = RX bits per color, *P* = RX pixels per clock
dp_rx_dp_sink_rx_vid_sol	Output	1	
dp_rx_dp_sink_rx_vid_eol	Output	1	
dp_rx_dp_sink_rx_vid_sof	Output	1	
dp_rx_dp_sink_rx_vid_eof	Output	1	
dp_rx_dp_sink_rx_vid_locked	Output	1	
dp_rx_dp_sink_rx_vid_interlace	Output	1	
dp_rx_dp_sink_rx_vid_field	Output	1	
dp_rx_dp_sink_rx_vid_overflow	Output	1	
dp_rx_dp_sink_rx_vid_data	Output	*B*P*3	
dp_rx_dp_sink_rx_vid_valid	Output	*P*	
dp_rx_dp_sink_rx_parallel_data	Input	*N* *S*10	DisplayPort parallel data from RX Native PHY *Note: N* = RX maximum lane count, *S* = RX symbols per clock
dp_rx_dp_sink_rx_std_clkout	Input	*N*	CDR clock out from RX Native PHY *Note: N* = RX maximum lane count
dp_rx_dp_sink_rx_restart	Output	1	Reset signal to RX Native PHY Reset controller when RX data loses alignment. Triggered by the DisplayPort RX core.
dp_rx_dp_sink_rx_reconfig_req	Output	1	Transceiver reconfiguration interface to the RX reconfiguration management module *Note: N* = RX maximum lane count
dp_rx_dp_sink_rx_reconfig_ack	Input	1	
dp_rx_dp_sink_rx_reconfig_busy	Input	1	
dp_rx_dp_sink_rx_bitslip	Output	*N*	
dp_rx_dp_sink_rx_cal_busy	input	*N*	
dp_rx_dp_sink_rx_analogreset	Output	*N*	

continued...

DisplayPort RX Signals			
dp_rx_dp_sink_rx_digitalreset	Output	*N*	
dp_rx_dp_sink_rx_is_lockedtoref	Input	*N*	
dp_rx_dp_sink_rx_is_lockedtodata	Input	*N*	
dp_rx_dp_sink_rx_set_locktoref	Output	*N*	
dp_rx_dp_sink_rx_set_locktodata	Output	*N*	

DisplayPort TX Signals			
dp_tx_reset_bridge_in_reset_n	Input	1	Reset to TX sub-system
dp_tx_clk_16_in_clk	Input	1	TX Auxiliary clock (16 MHz)
dp_tx_dp_source_clk_cal	Input	1	TX reconfiguration calibration clock
dp_tx_dp_source_tx_audio_valid	Input	1	TX audio channel interface *Note: M* = TX audio channel
dp_tx_dp_source_tx_audio_mute	Input	1	
dp_tx_dp_source_tx_audio_lpcm_data	Input	*M**32	
dp_tx_dp_source_tx_audio_clk	Input	1	
dp_tx_dp_source_tx_aux_in	Input	1	TX auxiliary interface
dp_tx_dp_source_tx_aux_out	Output	1	
dp_tx_dp_source_tx_aux_oe	Output	1	
dp_tx_dp_source_tx_hpd	Input	1	TX HPD
dp_tx_dp_source_tx_link_rate	Output	2	TX Link Rate 2-bit indicator, used in transceiver reconfiguration management • RBR: 2'b00 • HBR: 2'b01 • HBR2: 2'b10 • HBR3: 2'b11
dp_tx_dp_source_tx_link_rate_8bits	Output	8	TX Link Rate 8-bit indicator, used in transceiver reconfiguration management • RBR: 0x06 • HBR: 0x0A • HBR2: 0x14 • HBR3: 0x1E

continued...

DisplayPort TX Signals			
dp_tx_dp_source_tx_ss_ready	Output	1	DisplayPort TX secondary stream interface
dp_tx_dp_source_tx_ss_valid	Input	1	
dp_tx_dp_source_tx_ss_data	Input	128	
dp_tx_dp_source_tx_ss_sop	Input	1	
dp_tx_dp_source_tx_ss_eop	Input	1	
dp_tx_dp_source_tx_ss_clk	Output	1	
dp_tx_dp_source_tx_vid_clk	Input	1	DisplayPort TX video stream (VYSNC/HSYNC/DE) interface (only used when TX_SUPPORT_IM_ENABLE = 0) *Note: B* = TX bits per color, *P* = TX pixels per clock.
dp_tx_dp_source_tx_vid_data	Input	*B*P*3	
dp_tx_dp_source_tx_vid_v_sync	Input	*P*	
dp_tx_dp_source_tx_vid_h_sync	Input	*P*	
dp_tx_dp_source_tx_vid_de	Input	*P*	
dp_tx_dp_source_tx_im_clk	Input	1	DisplayPort TX video image interface (only used when TX_SUPPORT_IM_ENABLE = 1) *Note: B* = TX bits per color, *P* = TX pixels per clock.
dp_tx_dp_source_tx_im_sol	Input	1	
dp_tx_dp_source_tx_im_eol	Input	1	
dp_tx_dp_source_tx_im_sof	Input	1	
dp_tx_dp_source_tx_im_eof	Input	1	
dp_tx_dp_source_tx_im_data	Input	*B*P*3	
dp_tx_dp_source_tx_im_valid	Input	1	
dp_tx_dp_source_tx_im_locked	Input	1	
dp_tx_dp_source_tx_im_interlace	Input	1	
dp_tx_dp_source_tx_im_field	Input	1	
dp_tx_dp_source_tx_parallel_data	Output	*N*S*10	DisplayPort parallel data to TX Native PHY *Note: N* = TX maximum lane count, *S* = TX symbols per clock

continued...

DisplayPort TX Signals			
dp_tx_dp_source_tx_std_clkout	Input	*N*	TX Native PHY clock out *Note: N* = TX maximum lane count
dp_tx_dp_source_tx_pll_locked	Input	1	TX PLL locked indicator
dp_tx_dp_source_tx_reconfig_req	Output	1	Transceiver Reconfiguration interface to TX reconfiguration management module *Note: N* = TX maximum lane count
dp_tx_dp_source_tx_reconfig_ack	Input	1	
dp_tx_dp_source_tx_reconfig_busy	Input	1	
dp_tx_dp_source_tx_pll_powerdown	Output	1	
dp_tx_dp_source_tx_analog_reconfig_req	Output	1	
dp_tx_dp_source_tx_analog_reconfig_ack	Input	1	
dp_tx_dp_source_tx_analog_reconfig_busy	Input	1	
dp_tx_dp_source_tx_vod	Output	*N**2	
dp_tx_dp_source_tx_emp	Output	*N**2	
dp_tx_dp_source_tx_analogreset	Output	*N*	
dp_tx_dp_source_tx_digitalreset	Output	*N*	
dp_tx_dp_source_tx_cal_busy	Input	*N*	

Table 15. RX PHY Top-Level Signals

Signal	Direction	Width	Description
rx_cdr_refclk	Input	1	RX Native PHY CDR reference clock. This design example uses 135 MHz.
dp_rx_clk_cal	Output	1	50 MHz DisplayPort RX reconfiguration calibration clock. This clock must be synchronous to rcfg_mgmt_clk.
rx_cdr_resetn	Input	1	RX Native PHY reset (active low)
video_pll_locked	Input	1	This signal indicates that the video PLL (video clock and clk16) is stable and locked. Use as reset to the Intel FPGA DisplayPort IP core and the transceiver.
dp_rx_link_rate_8bits	Input	8	RX link rate indicator, used in transceiver reconfiguration management
rx_rcfg_mgmt_reset	Input	1	RX reconfiguration reset
rx_rcfg_mgmt_clk	Input	1	RX reconfiguration management clock (100 MHz)
rx_rcfg_en	Output	1	RX reconfiguration enable signal

continued...

Signal	Direction	Width	Description
rx_rcfg_write	Output	1	Reconfiguration Avalon-MM interfaces that interact with Transceiver Arbiter *Note: N* = RX maximum lane count (1, 2, or 4)
rx_rcfg_read	Output	1	
rx_rcfg_address	Output	12	
rx_rcfg_writedata	Output	32	
rx_rcfg_readdata	Input	32	
rx_rcfg_waitrequest	Input	1	
rx_rcfg_cal_busy	Input	*N*	
gxb_rx_rcfg_write	Input	*N*	Reconfiguration Avalon-MM interfaces from Transceiver Arbiter *Note: N* = RX maximum lane count (1, 2, or 4)
gxb_rx_rcfg_read	Input	*N*	
gxb_rx_rcfg_address	Input	*N**10	
gxb_rx_rcfg_writedata	Input	*N**32	
gxb_rx_rcfg_readdata	Output	*N**32	
gxb_rx_rcfg_waitrequest	Output	*N*	
gxb_rx_rcfg_cal_busy	Output	*N*	
gxb_rx_clkout	Output	*N*	RX Native PHY CDR clock out *Note: N* = RX maximum lane count (1, 2, or 4)
gxb_rx_serial_data	Input	*N*	DisplayPort Serial Data to RX Native PHY *Note: N* = RX maximum lane count (1, 2, or 4)
dp_rx_parallel_data	Output	*N***S**10	DisplayPort parallel data to DisplayPort RX core *Note: N* = RX maximum lane count (1, 2, or 4), *S* = RX symbols per clock (2 or 4)
dp_rx_restart	Input	1	Reset signal to the RX Native PHY Reset controller when RX data loses alignment. Triggered by the DisplayPort RX core.
dp_rx_rcfg_req	Input	1	Transceiver Reconfiguration interface from the DisplayPort RX core *Note: N* = RX maximum lane count (1, 2, or 4)
dp_rx_rcfg_ack	Output	1	
dp_rx_rcfg_busy	Output	1	
dp_rx_is_lockedtoref	Output	*N*	
dp_rx_is_lockedtodata	Output	*N*	
dp_rx_bitslip	Input	*N*	
dp_rx_cal_busy	Output	1	
dp_rx_set_locktoref	Input	*N*	
dp_rx_set_locktodata	Input	*N*	

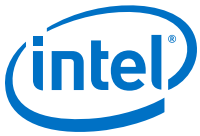


Table 16. TX PHY Top-Level Signals

Signal	Direction	Width	Description
tx_pll_refclk	Input	1	TX transceiver PLL reference clock. This design example uses 135 MHz.
dp_tx_clk_cal	Output	1	50 MHz DisplayPort TX reconfiguration calibration clock. This clock must be synchronous to rcfg_mgmt_clk.
tx_pll_resetn	Input	1	TX transceiver PLL reset (active low)
video_pll_locked	Input	1	This signal indicates that the video PLL (video clock and clk16) is stable and locked. Use as reset to the Intel FPGA DisplayPort IP core and the transceiver.
tx_cad	Output	1	Driven to FMC card TX CAD. Tied to 0.
dp_tx_link_rate_8bits	Input	8	TX Link Rate indicator, used in transceiver reconfiguration management. • RBR: 0x06 • HBR: 0x0A • HBR2: 0x14 • HBR3: 0x1E
tx_rcfg_mgmt_reset	Input	1	TX reconfiguration reset
tx_rcfg_mgmt_clk	Input	1	TX reconfiguration management clock (100 MHz)
tx_rcfg_en	Output	1	TX reconfiguration enable signal
tx_rcfg_write	Output	1	Reconfiguration Avalon-MM interfaces to Transceiver Arbiter *Note: N* = TX maximum lane count (1, 2, or 4)
tx_rcfg_read	Output	1	
tx_rcfg_address	Output	12	
tx_rcfg_writedata	Output	32	
tx_rcfg_readdata	Input	32	
tx_rcfg_waitrequest	Input	1	
tx_rcfg_cal_busy	Input	*N*	
gxb_tx_rcfg_write	Input	*N*	Reconfiguration Avalon-MM interfaces from Transceiver Arbiter *Note: N* = TX maximum lane count (1, 2, or 4)
gxb_tx_rcfg_read	Input	*N*	
gxb_tx_rcfg_address	Input	*N**10	
gxb_tx_rcfg_writedata	Input	*N**32	
gxb_tx_rcfg_readdata	Output	*N**32	
gxb_tx_rcfg_waitrequest	Output	*N*	
gxb_tx_rcfg_cal_busy	Output	*N*	
gxb_tx_clkout	Output	*N*	Transceiver clock out *Note: N* = TX maximum lane count (1, 2, or 4)
gxb_tx_serial_data	Output	*N*	DisplayPort Serial Data from Transceiver *Note: N* = TX maximum lane count
dp_tx_parallel_data	Input	*N***S**10	DisplayPort Parallel Data from DisplayPort TX Core

continued...

Signal	Direction	Width	Description
			Note: N = TX maximum lane count (1, 2, or 4), *S* = TX symbols per clock (2 or 4)
dp_tx_rcfg_req	Input	1	Transceiver Reconfiguration interface from DisplayPort TX Core *Note: N* = TX maximum lane count (1, 2, or 4)
dp_tx_rcfg_ack	Output	1	
dp_tx_rcfg_vod	Input	8	
dp_tx_rcfg_emp	Input	8	
dp_txpll_rcfg_req	Input	1	
dp_txpll_rcfg_ack	Output	1	
dp_tx_rcfg_busy	Output	1	
dp_txpll_powerdown	Input	1	
dp_tx_cal_busy	Output	*N*	
dp_txpll_locked	Output	1	

Table 17. Transceiver Arbiter Signals

Signal	Direction	Width	Description
clk	Input	1	Reconfiguration clock. This clock must share the same clock with the reconfiguration management blocks.
reset	Input	1	Reset signal. This reset must share the same reset with the reconfiguration management blocks.
rx_rcfg_en	Input	1	RX reconfiguration enable signal
tx_rcfg_en	Input	1	TX reconfiguration enable signal
rx_rcfg_ch	Input	2	Indicates which channel to be reconfigured on the RX core. This signal must always remain asserted.
tx_rcfg_ch	Input	2	Indicates which channel to be reconfigured on the TX core. This signal must always remain asserted.
rx_reconfig_mgmt_write	Input	1	Reconfiguration Avalon-MM interfaces from the RX reconfiguration management
rx_reconfig_mgmt_read	Input	1	
rx_reconfig_mgmt_address	Input	10	
rx_reconfig_mgmt_writedata	Input	32	
rx_reconfig_mgmt_readdata	Output	32	
rx_reconfig_mgmt_waitrequest	Output	1	
tx_reconfig_mgmt_write	Input	1	Reconfiguration Avalon-MM interfaces from the TX reconfiguration management
tx_reconfig_mgmt_read	Input	1	

continued...

Signal	Direction	Width	Description
tx_reconfig_mgmt_address	Input	10	
tx_reconfig_mgmt_writedata	Input	32	
tx_reconfig_mgmt_readdata	Output	32	
tx_reconfig_mgmt_waitrequest	Output	1	
reconfig_write	Output	1	Reconfiguration Avalon-MM interfaces to the transceiver
reconfig_read	Output	1	
reconfig_address	Output	10	
reconfig_writedata	Output	32	
rx_reconfig_readdata	Input	32	
rx_reconfig_waitrequest	Input	1	
tx_reconfig_readdata	Input	1	
tx_reconfig_waitrequest	Input	1	
rx_cal_busy	Input	1	Calibration status signal from the RX transceiver
tx_cal_busy	Input	1	Calibration status signal from the TX transceiver
rx_reconfig_cal_busy	Output	1	Calibration status signal to the RX transceiver PHY reset control
tx_reconfig_cal_busy	Output	1	Calibration status signal from the TX transceiver PHY reset control

Table 18. Pixel Clock Recovery Signals

The PCR module in the dynamic generation design example is an enhanced version where 2 Fractional PLLs (FPLLs) are used.

Signal	Direction	Width	Description
areset	Input	1	PCR reset
clk	Input	1	Control loop clock (16 MHz)
clk_135	Input	1	135 MHz clock
rx_link_clk	Input	1	RX Native PHY CDR clock out
rx_link_rate	Input	2	RX link rate 2-bit indicator
rx_msa	Input	217	RX MSA
vidin_clk	Input	1	RX video clock. If MAX_LINK_RATE = HBR2 and PIXELS_PER_CLOCK = Dual, uses 300 MHz. Otherwise, fixed to 160 MHz.
vidin_data	Input	*B***P**3	RX video stream interface from RX core *Note: B* = RX bits per color, *P* = RX pixels per clock.
vidin_valid	Input	1	

continued...

Signal	Direction	Width	Description
vidin_locked	Input	1	
vidin_sof	Input	1	
vidin_eof	Input	1	
vidin_sol	Input	1	
vidin_eol	Input	1	
rec_clk	Output	1	Reconstructed/recovered video clock
rec_clk_x2	Output	1	Reconstructed/recovered video clock (2x faster); not used
vidout	Output	*B***P**3	TX video stream interface *Note: B* = TX bits per color, *P* = TX pixels per clock.
hsync	Output	1	
vsync	Output	1	
de	Output	1	
field2	Output	1	

Table 19. Pixel Clock Recovery Parameters

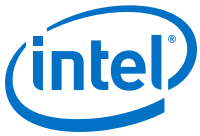
You can use these parameters to configure the clock recovery core.

Parameter	Default Value	Description
PIXELS_PER_CLOCK	1	Specifies how many pixels in parallel (for each clock cycle) are gathered from the DisplayPort RX core (1, 2 or 4).
BPP	24	Specifies the width (in bits) of a single pixel. 1 bit per pixel is equivalent to 3* bits per color.
CLK_PERIOD_NS	10	Specifies the period (in nanoseconds) of the clock signal connected to the port. In this design example, the value used is 62.
DEVICE_FAMILY	Intel Arria 10	Identifies the family of the device used.
FIXED_NVID	0	Specifies the configuration of the DisplayPort RX received video clocking used. • 1 if GPU NVID is fixed to 'h8000 • 0 if GPU NVID is not fixed Select 0 if you require the PCR to inter-operate with any GPU. Select 1 if you want to optimize resources but take note that this option may not work with certain GPUs.

2.5 Hardware Setup

The Intel FPGA DisplayPort design example is 4Kp60 capable and performs a loop-through for a standard DisplayPort video stream.

1. To run the hardware test, connect a DisplayPort-enabled source device to the DisplayPort FMC daughter card sink input.
2. The DisplayPort sink decodes the port into a standard video stream and sends it to the clock recovery core.
3. The clock recovery core synthesizes the original video pixel clock to be transmitted together with the received video data.



Note: You require the clock recovery feature to produce video without using a frame buffer.

4. The clock recovery core then sends the video data to the DisplayPort source and the Transceiver Native PHY TX block.
5. Connect the DisplayPort FMC daughter card source port to a monitor to display the image.

Table 20. On-board User LED Functions

LEDs	Function
USER_LED[0]	This LED indicates that the source is successfully lane-trained. At this point, the IP core asserts `RX0_vid_locked`.
USER_LED[5:1]	These LEDs illuminate design example lane counts. • 4'b0001 = 1 lane • 4'b0010 = 2 lanes • 4'b0100 = 4 lanes
USER_LED[7:6]	These LEDs indicate the RX link rate. • 2'b00 = RBR • 2'b01 = HBR • 2'b10 = HBR2 • 2'b11 = HBR3

2.6 Simulation Testbench

The simulation testbench simulates the DisplayPort TX serial loopback to RX.

Figure 9. Intel FPGA DisplayPort IP Core Simplex Mode Simulation Testbench Block Diagram

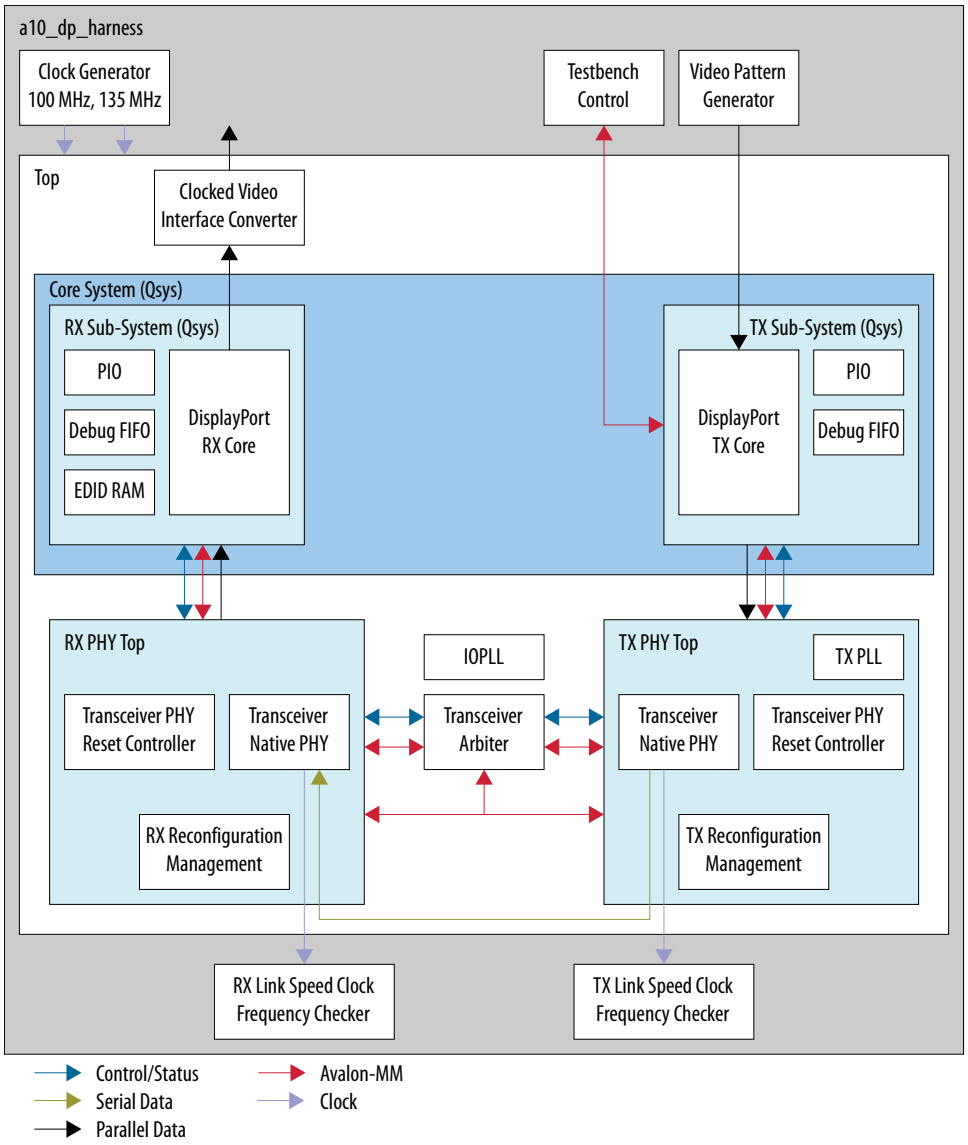


Table 21. Testbench Components

Component	Description
Video Pattern Generator	This generator produces color bar patterns that you can configure. You can parameterize the video format timing.
Testbench Control	This block controls the test sequence of the simulation and generates the necessary stimulus signals to the TX core.

continued...

Component	Description
	The testbench control block also reads the CRC value from both source and sink to make comparisons.
RX Link Speed Clock Frequency Checker	This checker verifies if the RX transceiver recovered clock frequency matches the desired data rate.
TX Link Speed Clock Frequency Checker	This checker verifies if the TX transceiver recovered clock frequency matches the desired data rate.

The simulation testbench does the following verifications:

Test Criteria	Verification
• Link Training sweep across all data rates from HBR3 to HBR2 to HBR and RBR • Read the DPCD registers to check if the DP Status sets and measures both TX and RX Link Speed frequency.	Integrates Frequency Checker to measure the Link Speed clock's frequency output from the TX and RX transceiver.
• Run video pattern from TX to RX. • Verify the CRC for both source and sink to check if they match.	• Connects video pattern generator to the DisplayPort Source to generate the video pattern. • Testbench control next reads out both Source and Sink CRC from DPTX and DPRX registers and compares to ensure both CRC values are identical. *Note:* To ensure CRC is calculated, you must enable the RX/TX_SUPPORT_AUTOMATED_TEST parameter.

A successful simulation ends with the following message:

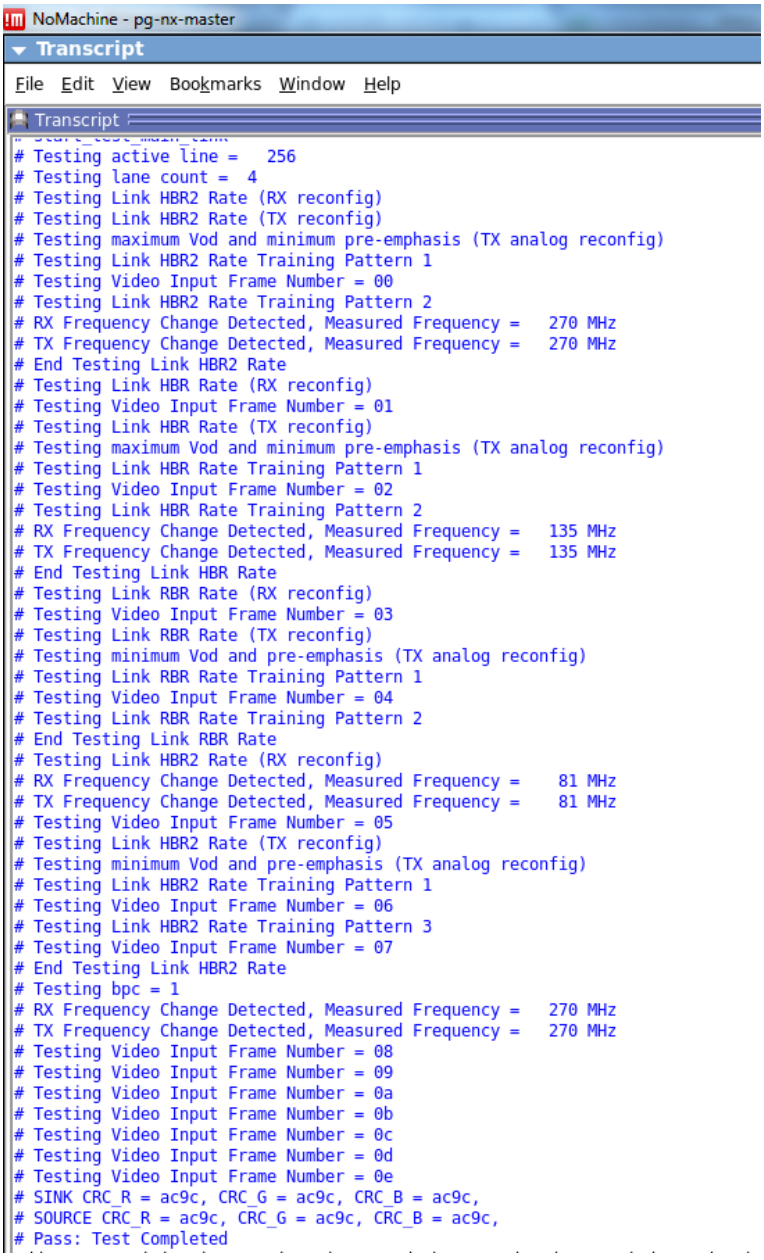


Table 22. DisplayPort Design Example Supporter Simulators

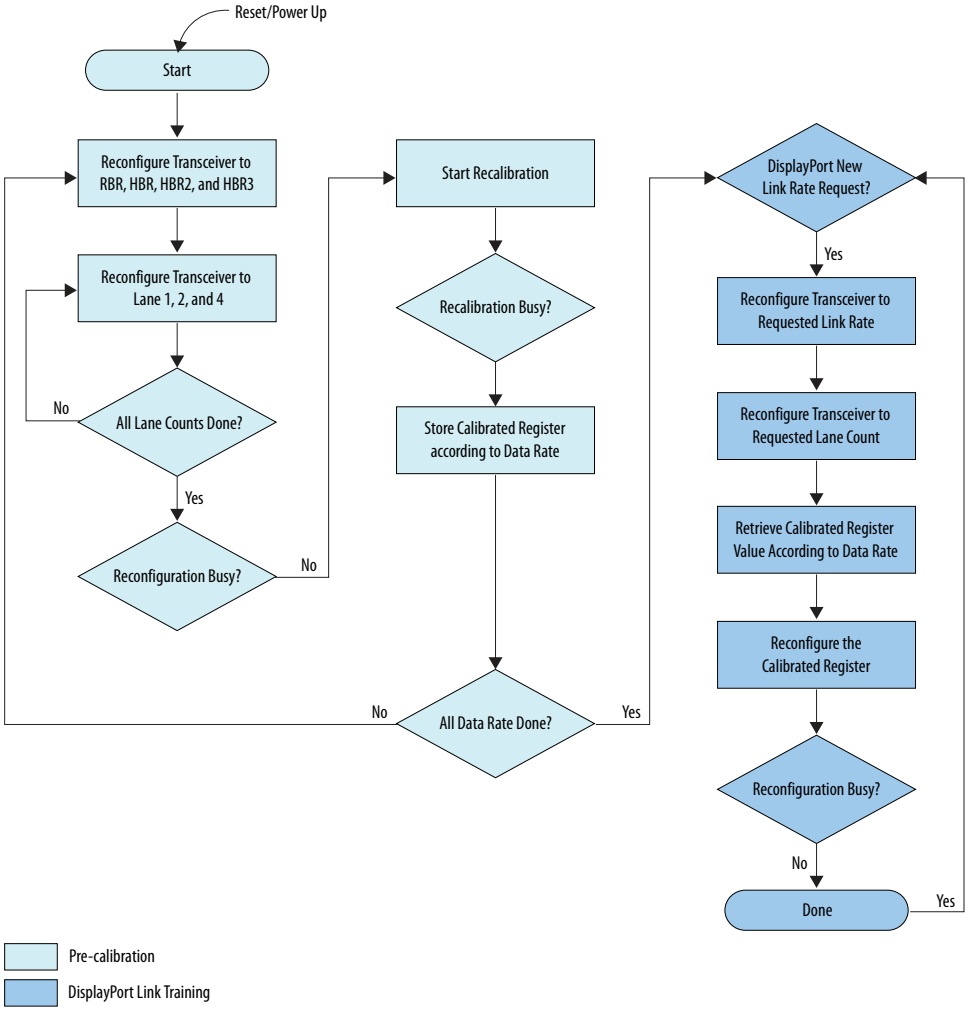
Simulator	Verilog HDL	VHDL
ModelSim	Yes	Yes
VCS/VCS-MX	Yes	Yes
Riviera-Pro	Yes	Yes
NCSim	Yes	No

2.7 DisplayPort Transceiver Reconfiguration Flow

The *VESA DisplayPort Standard version 1.4* supports 4 link rates (8.1 Gbps, 5.4 Gbps, 2.7 Gbps, and 1.62 Gbps). You can dynamically switch from 1 data rate to another. Transceiver reconfiguration is required to support dynamic link rate switching.

The Intel FPGA DisplayPort design examples require some level of reconfiguration and recalibration but with some modification. In these design examples, the pre-calibration method is implemented to reduce the transceiver reconfiguration duration.

Figure 10. Transceiver Reconfiguration Flowchart



The following sequences describe the flow.

1. Upon power up or push button reset, the DisplayPort reconfiguration module initiates the transceiver reconfiguration to sweep across all supported link rate and all lane count.
   a. For TX FPLL, these register offsets are reconfigured:

- 10'h12B (TXPLL M Counter)
- 10'h12C (TXPLL L Counter)

b. For RX FPLL, these register offsets are reconfigured:

- 10'h13a (RX L PFD and PD Counter)
- 10'h13b (RX M Counter)

2. After reconfiguration completes, recalibration initiates per data rate.
3. After calibration completes, the pre-defined calibrated registers will be stored according to the respective data rate.
   a. For TX FPLL, these register offsets are recalibrated:
      - 10'h10A (PLL VCO Frequency Band 0 fix low bits)
      - 10'h10B (PLL VCO Frequency Band 0 dyn)
      - 10'h142 (PLL VCO Frequency Band 0 fix high bits)
      - 10'h123 (PLL VCO Frequency Band 1 fix)
      - 10'h124 (PLL VCO Frequency Band 1 dyn)
      - 10'h125
      - 10'h126

   b. For CDR PLL (RX FPLL), these register offsets are recalibrated:
      - 10'h132 (CDR VCO Speed fix)
      - 10'h133 (Charge Pump Vcc register)
      - 10'h134 (CDR VCO Speed fix)
      - 10'h135 (LF PFD and PD Register)
      - 10'h136 (CDR VCO Speed fix)
      - 10'h137 (CDR VCO Speed fix)
      - 10'h139 (Charge Pump current PFD and PD register)
4. Steps 1 through 3 are repeated until all supported data rates are covered.
5. When the pre-calibration steps complete, the reconfiguration module is ready to start DisplayPort link training.
6. Whenever the Intel FPGA DisplayPort IP core sends a new link rate request, the reconfiguration module initiates reconfiguration to the transceiver.
7. The reconfiguration flow includes retrieving the calibrated register offset value that corresponds to the link rate and reconfigure it to the transceiver. No recalibration is required.
8. When reconfiguration completes, the transceiver is ready to receive the link rate.
9. The DisplayPort reconfiguration module continues to monitor if a new link rate request is detected. If it detects a new request, the module repeats step 5.

A Intel FPGA DisplayPort Design Example User Guide for Intel Arria 10 Devices Archives

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

IP Core Version	User Guide
17.0	Intel Arria 10 DisplayPort IP Core Design Example User Guide
16.1	DisplayPort IP Core Design Example User Guide

B Revision History for DisplayPort IP Core Design Example User Guide

Date	Version	Changes
November 2017	2017.11.06	• Renamed DisplayPort IP core to Intel FPGA DisplayPort as per Intel rebranding. • Changed the term Qsys to Platform Designer. • Renamed the design examples to **DisplayPort SST Parallel Loopback With PCR** and **DisplayPort SST Parallel Loopback Without PCR**. • Updated information that the Intel FPGA DisplayPort IP core now conforms to *VESA DisplayPort Standard version 1.4*. • Added data link rate support for HBR3 (8.10 Gbps). This rate is available only in quad symbols per clock for Intel Arria 10 devices in Intel Quartus Prime Pro Edition. • Added new pins for DisplayPort FMC Daughter Card Pins on FMC Port A. • Added a link for workaround to avoid jitter of PLL cascading or non-dedicated clock paths for Intel Arria 10 PLL reference clock.
May 2017	2017.05.08	• Rebranded as Intel. • Changed the part number. • Added files designated for Intel Quartus Prime Pro Edition. • Added information for a new design example variant: **Arria 10 DP SST Parallel Loopback Without PCR**. • Added information about the new TX video image interface. • Edited the function description for USER_LED[5:1]. The actual lane count should be 4'b0010 = 2 lanes and 4'b0100 = 4 lanes. • Added information about DisplayPort transceiver reconfiguration flow. • Added guidelines to regenerate `.elf` file. • Added link to archived version of the *Arria 10 DisplayPort IP Core Design Example User Guide*.
October 2016	2016.10.31	Initial release.