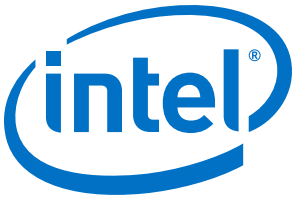


HDMI Intel® Arria 10 FPGA IP Design Example User Guide

Updated for Intel® Quartus® Prime Design Suite: **19.4**

IP Version: **19.3.0**

Subscribe
Send Feedback

UG-20077 | 2020.01.16
Latest document on the web: **PDF** | **HTML**

Contents

1. HDMI Intel® FPGA IP Design Example Quick Start Guide for Intel® Arria® 10 Devices

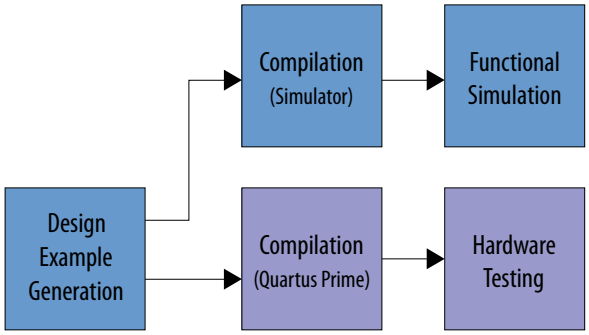
The HDMI Intel® FPGA IP design example for Intel Arria® 10 devices features a simulating testbench and a hardware design that supports compilation and hardware testing.

The HDMI Intel FPGA IP offers the following design examples:

- HDMI 2.1 RX-TX retransmit design with fixed rate link (FRL) mode enabled.
- HDMI 2.0 RX-TX retransmit design with FRL mode disabled.

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



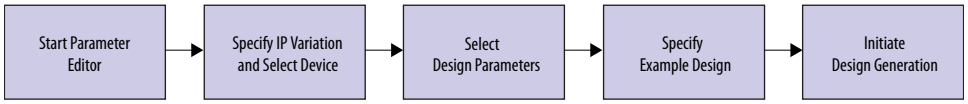
Related Information

Intel FPGA HDMI IP User Guide

1.1. Generating the Design

Use the HDMI Intel FPGA IP parameter editor in the Intel Quartus® Prime software to generate the design examples.

Figure 2. Generating the Design Flow



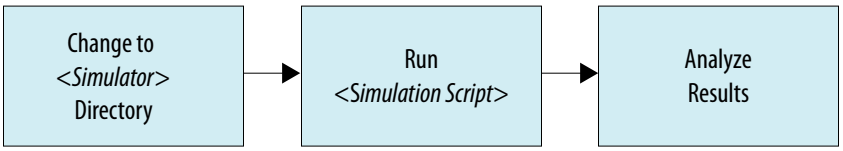
1. Create a project targeting Intel Arria 10 device family and select the desired device.
2. In the IP Catalog, locate and double-click **HDMI Intel FPGA IP**. The **New IP Variant** or **New IP Variation** window appears.
3. Specify a top-level name for your custom IP variation. The parameter editor saves the IP variation settings in a file named <*your_ip*>.ip or <*your_ip*>.qsys.
4. Click **OK**. The parameter editor appears.
5. On the **IP** tab, configure the desired parameters for both TX and RX.
6. Turn on the **Support FRL** parameter to generate the HDMI 2.1 design example in FRL mode. Turn it off to generate the HDMI 2.0 design example without FRL.
7. On the **Design Example** tab, select **Arria 10 HDMI RX-TX Retransmit**.
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 **Generate File Format**, select **Verilog** or **VHDL**.
10. For **Target Development Kit**, select **Intel Arria 10 GX FPGA Development Kit**. If you select a development kit, then the target device (selected in **step 4**) changes to match the device on target board. For **Intel Arria 10 GX FPGA Development Kit**, the default device is 10AX115S2F4I1SG.
11. Click **Generate Example Design**.

1.2. Simulating the Design

The HDMI testbench simulates a serial loopback design from a TX instance to an RX instance. Internal video pattern generator and audio pattern generator modules drive the HDMI TX instance and the serial output from the TX instance connects to the RX instance in the testbench.

Figure 3. Design Simulation Flow



Note: Simulation is available only for **Support FRL** disabled designs.

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

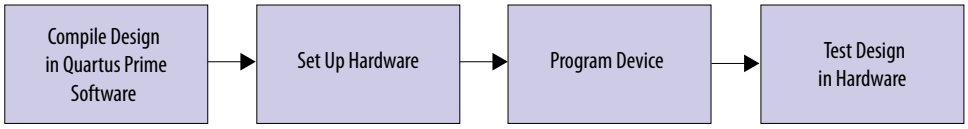
Table 1. Steps to Run Simulation

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`
Xcelium* Parallel	/simulation/xcelium	In the command line, type `source xcelium_sim.sh`

A successful simulation ends with the following message:

```
# SYMBOLS_PER_CLOCK = 2
# VIC               = 0
# AUDIO_CLK_DIVIDE  = 800
# TEST_HDMI_6G      = 1
# Simulation pass
```

1.3. 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 the `.qpf` file. .
   - HDMI 2.1 design example with FRL enabled: `project directory/quartus/a10_hdmi21_frl_demo.qpf`
   - HDMI 2.0 design example with FRL disabled: `project directory/quartus/a10_hdmi2_demo.qpf`
3. Click **Processing ➤ Start Compilation**.
4. After successful compilation, a `.sof` file will be generated in your specified directory.
5. Connect to the on-board FMCB (J2): .

- HDMI 2.1 design example with FRL enabled: Bitec HDMI 2.1 FMC Daughter Card Rev 4
- HDMI 2.0 design example with FRL disabled: Bitec HDMI 2.0 FMC Daughter Card Rev 11

6. Connect TX (P1) of the Bitec FMC daughter card to an external video source.
7. Connect RX (P2) of the Bitec FMC daughter card to an external video sink or video analyzer.
8. Ensure all switches on the development board are in default position.
9. Configure the selected Intel Arria 10 device on the development board using the generated `.sof` file (**Tools ➤ Programmer**).
10. The analyzer should display the video generated from the source.

Related Information

Intel Arria 10 FPGA Development Kit User Guide

1.4. HDMI Intel FPGA IP Design Example Parameters

Table 2. HDMI Intel FPGA IP Design Example Parameters for Intel Arria 10 Devices

These options are available for Intel Arria 10 devices only.

Parameter	Value	Description
Available Design Example		
Select Design	Arria 10 HDMI RX-TX Retransmit	Select the design example to be generated. The generated design example has pre-configured parameter settings. It does not follow user settings.
Design Example Files		
Simulation	On, Off	Turn on this option to generate the necessary files for the simulation testbench. *Note:* Simulation is available only for **Support FRL** disabled designs.
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. *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.

Target Development Kit		
		• No Development Kit: This option excludes all hardware aspects for the design example. The IP core sets all pin assignments to virtual pins. • 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. Detailed Description for HDMI 2.1 Design Example (Support FRL Enabled)

The HDMI 2.1 design example in FRL mode demonstrates one HDMI instance parallel loopback comprising three RX channels and four TX channels.

Table 3. HDMI 2.1 Design Example for Intel Arria 10 Devices

Design Example	Data Rate	Channel Mode	Loopback Type
Arria 10 HDMI RX-TX Retransmit	• 12 Gbps • 10 Gbps (Default) • 8 Gbps • 6 Gbps • 3 Gbps	Simplex	Parallel with FIFO buffer

Features

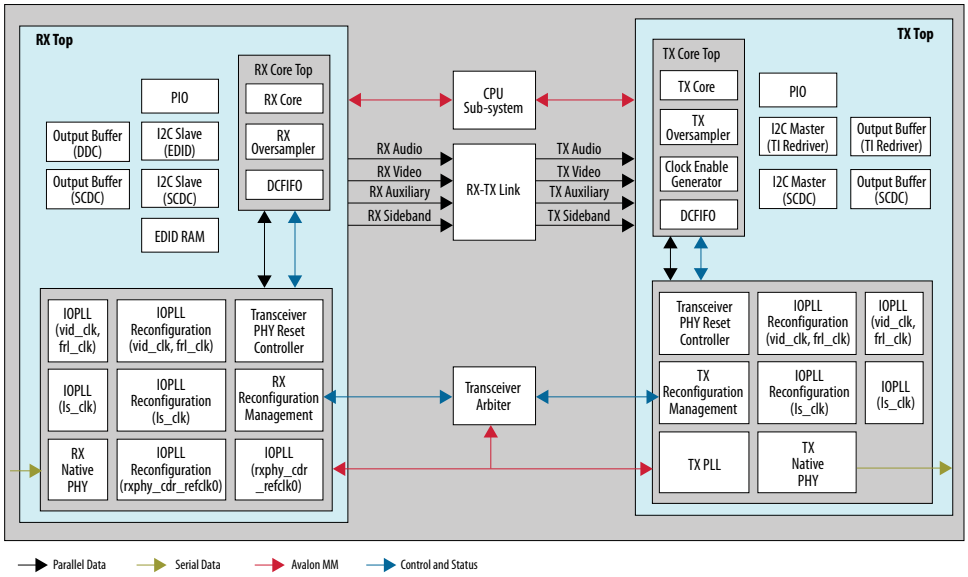
- The design instantiates FIFO buffers to perform a direct HDMI video stream passthrough between the HDMI 2.1 sink and source.
- The design is capable to switch between FRL mode and TMDS mode during run time.
- The design uses LED status for early debugging stage.
- The design comes with HDMI RX and TX instances.
- The design demonstrates the insertion and filtering of Dynamic Range and Mastering (HDR) InfoFrame in RX-TX link module.
- The design negotiates the FRL rate between the sink connected to TX and the source connected to RX. You can set the pre-configured maximum FRL rate for the RX by configuring the `HDMI_RX_MAX_FRL_RATE` parameter in the `software\tx_control_src\global.h` script. The Nios® II processor negotiates the link base on the capability of the sink connected to TX.
- The design includes several debugging features.

The RX instance receives a video source from the external video generator, and the data then goes through a loopback FIFO before it is transmitted to the TX instance. You need to connect an external video analyzer, monitor, or a television with HDMI connection to the TX core to verify the functionality.

2.1. HDMI 2.1 RX-TX Retransmit Design Block Diagram

The HDMI RX-TX retransmit design example demonstrates parallel loopback on simplex channel mode for HDMI 2.1 with **Support FRL** enabled.

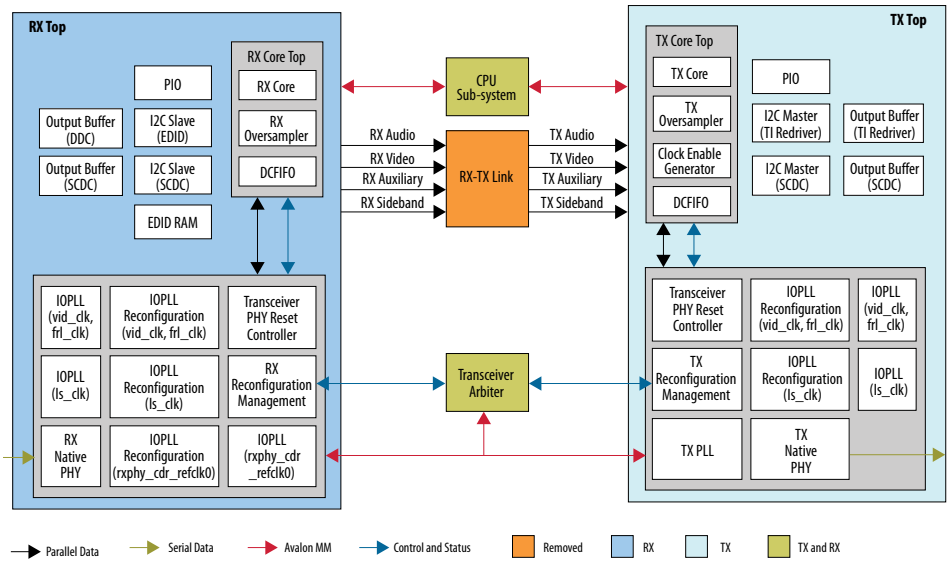
Figure 4. HDMI 2.1 RX-TX Retransmit Block Diagram



2.2. Creating RX-Only or TX-Only Designs

For advanced users, you can use the HDMI 2.1 design to create a TX- or RX-only design.

Figure 5. Components Required for RX-Only or TX-Only Design



To use RX- or TX-only components, remove the irrelevant blocks from the design.

Table 4. RX-Only and TX-Only Design Requirements

User Requirements	Preserve	Remove	Add
HDMI RX only	RX Top	• TX Top • RX-TX Link • CPU Subsystem • Transceiver Arbiter	–
HDMI TX only	• TX Top • CPU Sub-System	• RX Top • RX-TX Link • Transceiver Arbiter	Video Pattern Generator (custom module or generated from the Video and Image Processing (VIP) Suite)

Besides the RTL changes, you need to also edit the `main.c` script.

- For HDMI TX-only designs, decouple the wait for the HDMI RX lock status by removing the following lines and replace with `tx_xcvr_reconfig();`:

```
rx_hdmi_lock = READ_PIO(PIO_IN0_BASE, PIO_RX_LOCKED_OFFSET, PIO_RX_LOCKED_WIDTH);
while (rx_hdmi_lock == 0) {
if (check_hpd_isr()) {
break; }
// rx_vid_lock = READ_PIO(PIO_IN0_BASE, PIO_VID_LOCKED_OFFSET, PIO_VID_LOCKED_WIDTH);
rx_hdmi_lock = READ_PIO(PIO_IN0_BASE, PIO_RX_LOCKED_OFFSET, PIO_RX_LOCKED_WIDTH);
// Reconfig Tx after rx is locked
if (rx_hdmi_lock == 1) {
tx_xcvr_reconfig();
} }
```

- For HDMI RX-only designs, keep only the following lines in the `main.c` script:

```
REDRIVER_INIT();
hdmi_rx_init();
```

2.3. Directory Structure

The directories contain the generated files for the HDMI Intel FPGA IP design example.

Figure 6. Directory Structure for the Design Example

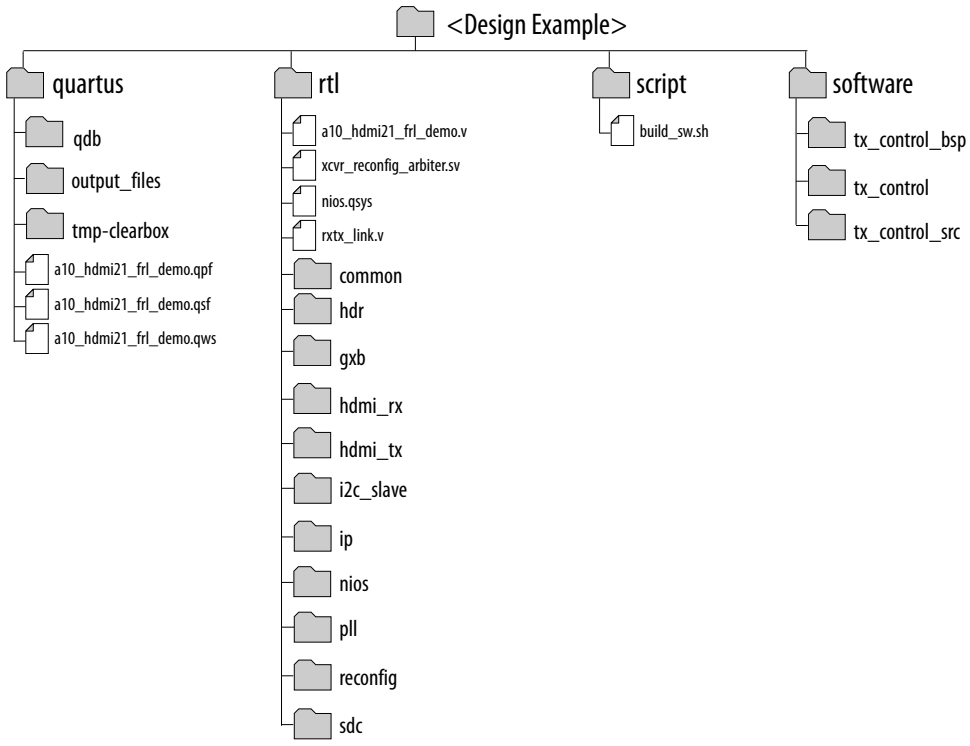


Table 5. Generated RTL Files

Folders	Files/Subfolders
common	clock_control.ip
	clock_crosser.v
	dcfifo_inst.v
	edge_detector.sv
	fifo.ip
	output_buf_i2c.ip
	test_pattern_gen.v
	tpg.v
	tpg_data.v
gxb	gxb_rx.ip
	gxb_rx_reset.ip
	gxb_tx.ip
	gxb_tx_fpll.ip
	gxb_tx_reset.ip
hdmi_rx	hdmi_rx.ip

continued...

Folders	Files/Subfolders
	hdmi_rx_top.v
	mr_hdmi_rx_core_top.v
	mr_rx_oversample.v
hdmi_tx	hdmi_tx.ip
	hdmi_tx_top.v
	mr_ce.v
	mr_hdmi_tx_core_top.v
	mr_tx_oversample.v
i2c_slave	edid_ram.ip
	i2c_avl_mst_intf_gen.v
	i2c_clk_cnt.v
	i2c_condt_det.v
	i2c_databuffer.v
	i2c_rxshifter.v
	i2c_slvfsm.v
	i2c_spksupp.v
	i2c_txout.v
	i2c_txshifter.v
	i2cslave_to_avlmm_bridge.v
	Panasonic.hex
pll	pll_hdmi.ip
	pll_hdmi_reconfig.ip
	pll_reconfig_ctrl.v
	pll_tmds.ip
	pll_vid_frl.ip
	quartus.ini
hdr	altera_hdmi_aux_hdr.v
	altera_hdmi_aux_snk.v
	altera_hdmi_aux_src.v
	altera_hdmi_hdr_infoframe.v
	avalon_st_mutiplexer.ip
reconfig	mr_rx_iopll_ls
	mr_rx_iopll_tmds
	mr_rx_iopll_vid_frl

continued...

Folders	Files/Subfolders
	mr_rxphy
	mr_tx_fpll
	mr_tx_iopll_ls
	mr_tx_iopll_vid_frl
	altera_xcvr_functions.sv
	mr_clock_sync.v
	mr_compare.sv
	mr_rate_detect.v
	mr_rx_rate_detect_top.v
	mr_rx_rcfg_ctrl.v
	mr_rx_reconfig.v
	mr_tx_rate_detect_top.v
	mr_tx_rcfg_ctrl.v
	mr_tx_reconfig.v
	rcfg_array_streamer_iopll.sv
	rcfg_array_streamer_rxphy.sv
	rcfg_array_streamer_rxphy_xn.sv
	rcfg_array_streamer_txphy.sv
	rcfg_array_streamer_txphy_xn.sv
	rcfg_array_streamer_txpll.sv
sdc	a10_hdmi2.sdc
	mr_clock_sync.sdc
	jtag.sdc

Table 6. Generated Software Files

Folders	Files
tx_control_src *Note:* The tx_control folder also contains duplicates of these files.	global.h
	hdmi_rx.c
	hdmi_rx.h
	hdmi_tx.c
	hdmi_tx.h
	hdmi_tx_read_edid.c
	hdmi_tx_read_edid.h
	intel_fpga_i2c.c
	intel_fpga_i2c.h

continued...

Folders	Files
	main.c
	pio_read_write.c
	pio_read_write.h

2.4. Hardware and Software Requirements

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

Hardware

- Intel Arria 10 GX FPGA Development Kit
- HDMI 2.1 Source (Quantum Data 980 48G Generator or Astro Design VA-1847 Generator)
- HDMI 2.1 Sink (Quantum Data 980 48G Analyzer or Astro Design VA-1847 Analyzer)
- Bitec HDMI FMC 2.1 daughter card (Revision 4.0)
- HDMI 2.1 Category 3 cables (tested with Belkin 48Gbps HDMI 2.1 Cable)

Software

- Intel Quartus Prime version 19.4

2.5. Design Components

The HDMI Intel FPGA IP design example consists of the common top-level components and HDMI TX and RX top components.

2.5.1. HDMI TX Components

The HDMI TX top components include the TX core top-level components, and the I^2C master, IOPLL, transceiver PHY reset controller, transceiver native PHY, TX PLL, TX reconfiguration management, IOPLL reconfiguration, and PIO blocks.

Figure 7. HDMI TX Top Components

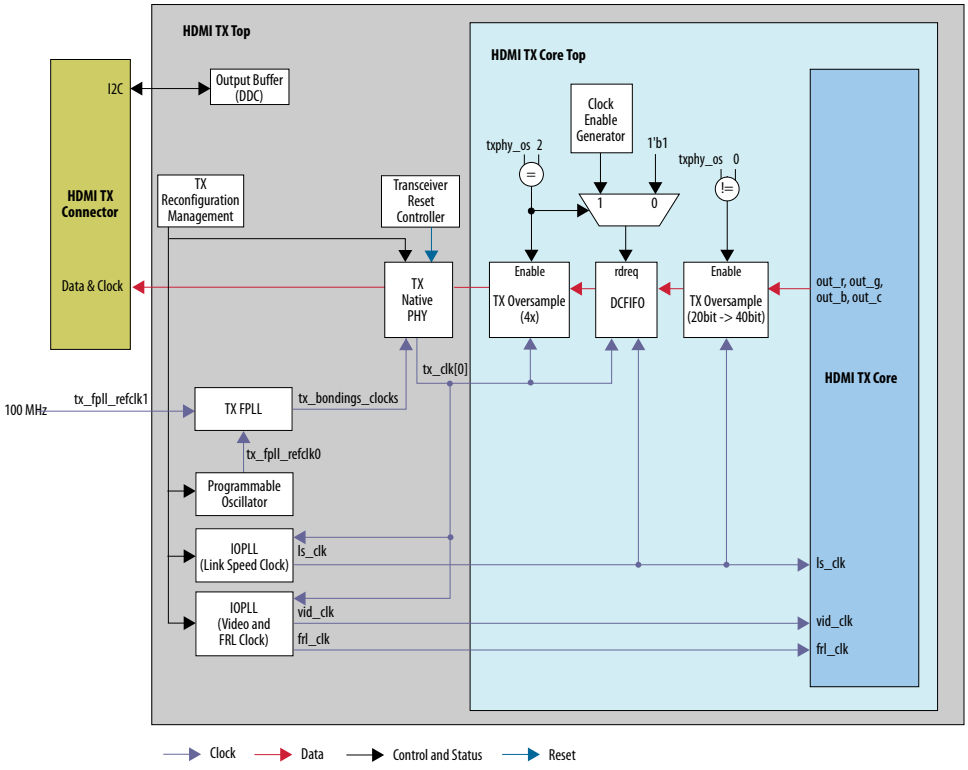


Table 7. HDMI TX Top Components

Module	Description
TX Core Top	The TX Core top level consists of:

continued...

Module	Description
	• HDMI TX Core—The IP receives video data from the top level and performs auxiliary data encoding, audio data encoding, video data encoding, scrambling, TMDS encoding or packetization . • TX Oversampler—The TX Oversampler module transmits data by repeating each bit of the input word a given number of times and constructs the output words. . There are two TX oversampler modules. — The first oversample module perform 2 times oversampling by extending the data width from 20 bits to 40 bits while maintaining the data rate. — The second TX oversampler module perform 4 times oversampling by maintaining the same data width while repeats each data bit for 4 times. So the overall oversampling factor could be either 2 or 8 depending on the TMDS clock frequency for TMDS mode only. The TX oversampler assumes that the input word is only valid every 2 or 8 clock cycles according to the oversampling factor. This block is enabled when the outgoing data stream is below the minimum link rate of the TX transceiver. When the TX transceiver runs on Enhanced PCS, the minimum link rate is 2,000 Mbps. (Refer to Table 8 on page 17 .) • DCFIFO —The DCFIFO transfers data from the TX link speed clock domain to the transceiver parallel clock out domain. When the Nios II processor determines the outgoing data stream is below the TX transceiver minimum data rate, the TX transceiver accepts the data from the TX oversampler. Otherwise, the TX transceiver reads the data directly from the DCFIFO with a read request asserted at all times. • Clock Enable Generator—A logic that generates a clock enable pulse. This clock enable pulse asserts every four clock cycles and serves as a read request signal to clock the data out from the DCFIFO.
IOPLL	The HDMI TX uses two IOPLLs . • The first IOPLL (`iopll_ls`) generates the link speed clock for the TX core. This reference clock receives the TX FPLL output clock. The output clock frequency: Link speed clock = Data rate per lanes/Transceiver width • The second IOPLL (`iopll_vid_frl`) generates the video clock and the FRL clock. Video clock frequency = Data rate per lanes x Number of lanes/(Pixels per clock x 24) FRL clock frequency = Data rate per lanes x 4 / (FRL characters per clock x 18)
Transceiver PHY Reset Controller	The Transceiver PHY reset controller ensures a reliable initialization of the TX transceivers. The reset input of this controller is triggered from the top level, and it generates the corresponding analog and digital reset signal to the Transceiver Native PHY block according to the reset sequencing inside the block. The `tx_ready` output signal from this block also functions as a reset signal to the HDMI Intel FPGA IP to indicate the transceiver is up and running, and ready to receive data from the core.
Transceiver Native PHY	Hard transceiver block that receives the parallel data from the HDMI TX core and serializes the data from transmitting it. *Note:* To meet the HDMI TX inter-channel skew requirement, set the TX channel bonding mode option in the Intel Arria 10 Transceiver Native PHY parameter editor to **PMA and PCS bonding**. You also need to add the maximum skew (`set_max_skew`) constraint requirement to the digital reset signal from the transceiver reset controller (`tx_digitalreset`) as recommended in the *Intel Arria 10 Transceiver PHY User Guide.*
TX PLL	The transmitter PLL block provides the serial fast clock to the Transceiver Native PHY block. For this HDMI Intel FPGA IP design example, fPLL is used as TX PLL.

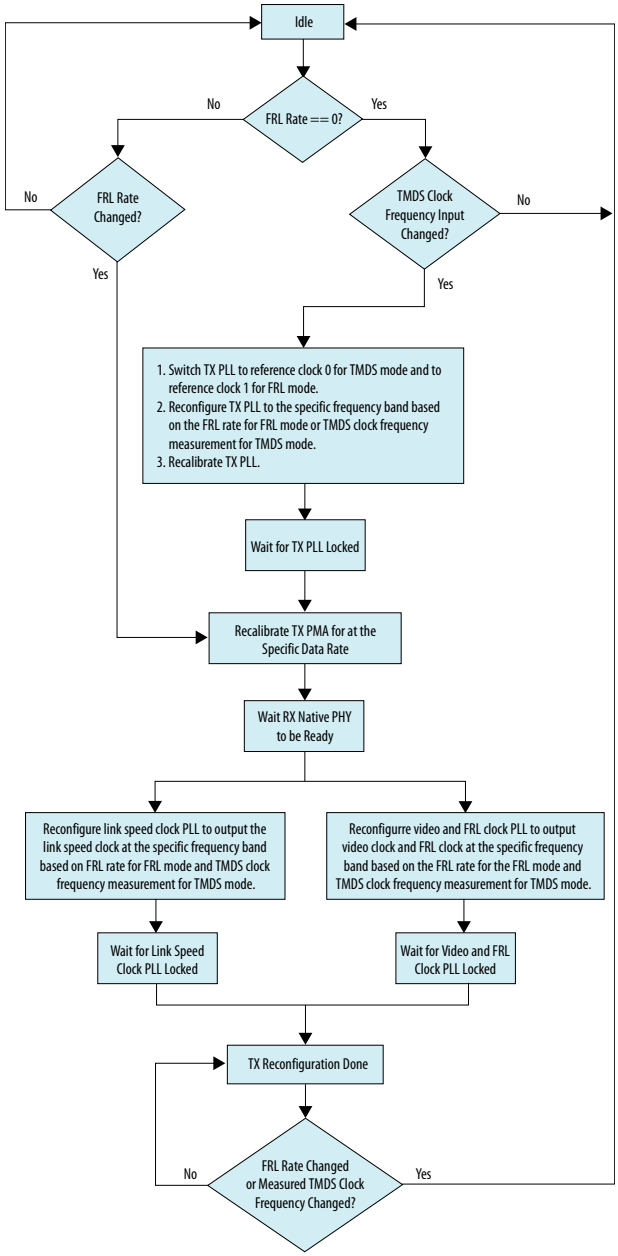
continued...

Module	Description
	TX PLL has two reference clocks. • Reference clock 0 is connected to the programmable oscillator (with TMDS clock frequency) for TMDS mode. In this design example, RX TMDS clock is used to connect to reference clock 0 for TMDS mode. Intel recommends you to use programmable oscillator with TMDS clock frequency for reference clock 0. • Reference clock 1 is connected to a fixed 100 MHz clock for FRL mode.
TX Reconfiguration Management	• In TMDS mode, the TX reconfiguration management block reconfigures the TX PLL for different output clock frequency according to the TMDS clock frequency of the specific video. • In FRL mode, the TX reconfiguration management block reconfigures the TX PLL to supply the serial fast clock for 3 Gbps, 6 Gbps, 8 Gbps, 10 Gbps and 12 Gbps according to `FRL_Rate` field in the 0x31 SCDC register. • TX reconfiguration management block also switches the TX PLL reference clock between reference clock 0 for TMDS mode and reference clock 1 for FRL mode.
Output buffer	This buffer acts as an interface to interact the I^2C interface of the HDMI DDC and redriver components.

Table 8. Transceiver Data Rate and Oversampling Factor Each Clock Frequency Range

Mode	Data Rate	Oversampler 1 (2x oversample)	Oversampler 2 (4x oversample)	Oversample Factor	Oversampled Data Rate (Mbps)
TMDS	250–1000	On	On	8	2000–8000
TMDS	1000–6000	On	Off	2	2000–12000
FRL	3000	Off	Off	1	3000
FRL	6000	Off	Off	1	6000
FRL	8000	Off	Off	1	8000
FRL	10000	Off	Off	1	10000
FRL	12000	Off	Off	1	12000

Figure 8. TX Reconfiguration Sequence Flow



2.5.2. HDMI RX Components

The HDMI RX top components include the RX core top-level components, and the I^2C slave, EDID RAM, IOPLL, transceiver PHY reset controller, RX native PHY, RX reconfiguration management, IOPLL reconfiguration, and PIO blocks.

Figure 9. HDMI RX Top Components

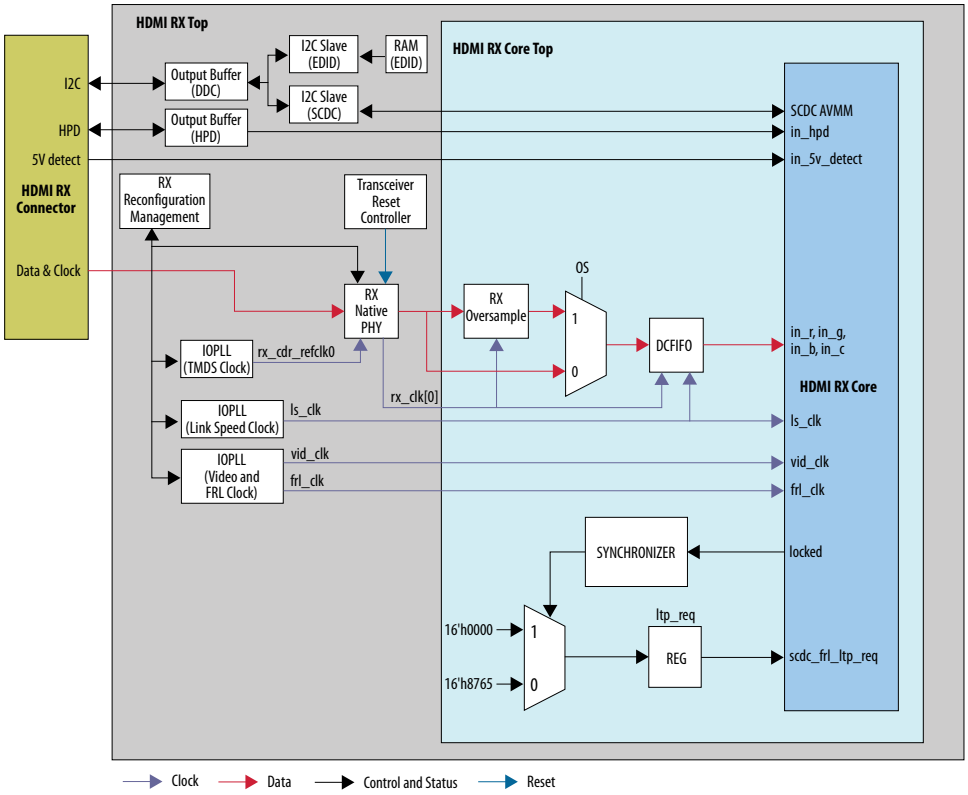


Table 9. HDMI RX Top Components

Module	Description
RX Core Top	The RX Core top level consists of: • HDMI RX Core—The IP receives the serial data from the Transceiver Native PHY and performs data alignment, channel deskew, TMDS decoding, auxiliary data decoding, video data decoding, audio data decoding, and descrambling. • RX Oversampler—The RX Oversampler module extracts data from the oversampled incoming data stream when the detected clock frequency band is below the transceiver minimum link rate. The oversampling factor is fixed at 5 and you can program the data width to support different number of symbols. The extracted bit is accompanied by a data valid pulse asserted every 5 clock cycles. • DCFIFO —The DCFIFO transfers data from the RX transceiver recovered clock domain to the RX link speed clock domain. The DCFIFO receives the RX transceiver parallel data either from Standard PCS (`scdc_frl_rate` = 0) or from Enhanced PCS mode (`scdc_frl_rate` = >0).
I^2C Slave	I^2C is the interface used for Sink Display Data Channel (DDC) and Status and Data Channel (SCDC). The HDMI source uses the DDC to determine the capabilities and characteristics of the sink by reading the Enhanced Extended Display Identification Data (E-EDID) data structure.

continued...

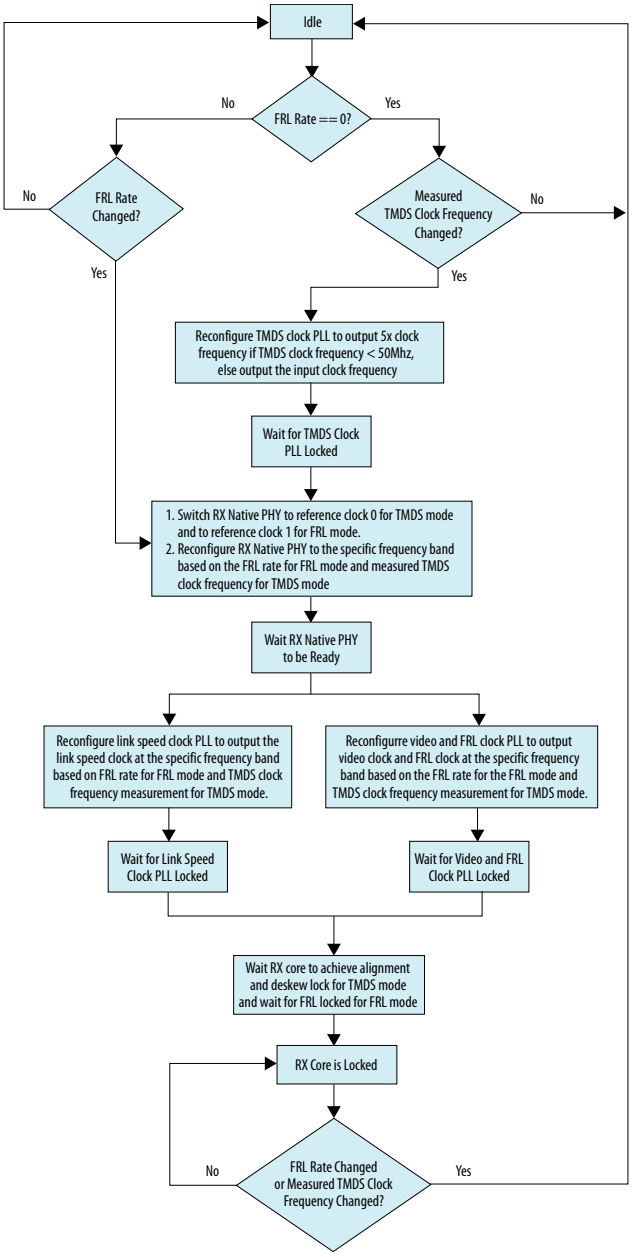
Module	Description
	• The 8-bit I²C slave addresses for E-EDID are 0xA0 and 0xA1. The LSB indicates the access type: 1 for read and 0 for write. When an HPD event occurs, the I²C slave responds to E-EDID data by reading from the on-chip RAM. • The I²C slave-only controller also supports SCDC for HDMI 2.0 and 2.1 operations. The 9-bit I²C slave address for the SCDC are 0xA8 and 0xA9. When an HPD event occurs, the I²C slave performs write or read transaction to or from SCDC interface of the HDMI RX core. • Link training process for Fixed Rate Link (FRL) also happens through I²C interface. During an HPD event or when the source writes a different FRL rate to the FRL `Rate` register (SCDC registers 0x31 bit[3:0]), the link training process starts. *Note:* This I²C slave-only controller for SCDC is not required if HDMI 2.0 or HDMI 2.1 is not intended.
EDID RAM	The design stores the EDID information using the RAM 1-port IP. A standard two-wire (clock and data) serial bus protocol (I²C slave-only controller) transfers the CEA-861-D Compliant E-EDID data structure. This EDID RAM stores the E-EDID information. • When in TMDS mode, the design supports EDID passthrough from TX to RX. During EDID passthrough, when the TX is connected to the external sink, the Nios II processor reads the EDID from the external sink and writes to the EDID RAM. • When in FRL mode, the Nios II processor writes the pre-configured EDID for each link rate based on the `HDMI_RX_MAX_FRL_RATE` parameter in the `global.h` script. Use the following `HDMI_RX_MAX_FRL_RATE` inputs for the supported FRL rate: • 1: 3G 3 Lanes • 2: 6G 3 Lanes • 3: 6G 4 Lanes • 4: 8G 4 Lanes • 5: 10G 4 Lanes (default) • 6: 12G 4 Lanes
IOPLL	The HDMI RX uses three IOPLLs . • The first IOPLL (`pll_tmds`) generates the RX CDR reference clock. This IOPLL is only used in TMDS mode. The reference clock of this IOPLL receives the TMDS clock. The TMDS mode uses this IOPLL because the CDR cannot receive reference clocks below 50 MHz and the TMDS clock frequency ranges from 25 MHz to 340 MHz. This IOPLL provides clock frequency that is 5 times of the input reference clock for frequency range between 25 MHz to 50 MHz and provides the same clock frequency as input reference clock for frequency range between 50 MHz to 340 MHz. • The second IOPLL (`iopll_ls`) generates the link speed clock for the RX core. This reference clock receives the CDR recovered clock. The output clock frequency: Link speed clock = Data rate per lanes/Transceiver width • The third IOPLL (`iopll_vid_frl`) generates the video clock and the FRL clock. Video clock frequency = Data rate per lanes x Number of lanes/(Pixels per clock x 24) FRL clock frequency = Data rate per lanes x 4 / (FRL characters per clock x 18) *Note:* The default IOPLL configuration is not valid for any HDMI resolution. The IOPLL is reconfigured to the appropriate settings upon power up.
Transceiver PHY Reset Controller	The Transceiver PHY reset controller ensures a reliable initialization of the RX transceivers. The reset input of this controller is triggered by the RX reconfiguration, and it generates the corresponding analog and digital reset signal to the Transceiver Native PHY block according to the reset sequencing inside the block.

continued...

Module	Description
RX Native PHY	Hard transceiver block that receives the serial data from an external video source. It deserializes the serial data to parallel data before passing the data to the HDMI RX core. This block runs on Enhanced PCS for FRL mode. RX CDR has two reference clocks. • Reference clock 0 is connected to output clock of IOPLL TMDS (`pll_tmds`), which is derived from the TMDS clock. • Reference clock 1 is connected to a fixed 100 MHz clock. In TMDS mode, RX CDR is reconfigured to select reference clock 0, and in FRL mode, RX CDR is reconfigured to select reference clock 1.
RX Reconfiguration Management	In TMDS mode, the RX reconfiguration management block implements rate detection circuitry with the HDMI PLL to drive the RX transceiver to operate at any arbitrary link rates ranging from 250 Mbps to 6,000 Mbps In FRL mode, the RX reconfiguration management block reconfigures the RX transceiver to operate at 3 Gbps, 6 Gbps, 8 Gbps, 10 Gbps or 12 Gbps depending on the FRL rate in the SCDC_FRL_RATE register field (0x31[3:0]). The RX reconfiguration management block switches between Standard PCS/RX for TMDS mode and Enhanced PCS for FRL mode. Refer to Figure 10 on page 22.
IOPLL Reconfiguration	IOPLL reconfiguration block facilitates dynamic real-time reconfiguration of PLLs in Intel FPGAs. This block updates the output clock frequency and PLL bandwidth in real time, without reconfiguring the entire FPGA. This block runs at 100 MHz in Intel Arria 10 devices. Due to IOPLL reconfiguration limitation, apply the `Quartus INI permit_nf_pll_reconfig_out_of_lock=on` during the IOPLL reconfiguration IP generation. To apply the `Quartus INI`, include “`permit_nf_pll_reconfig_out_of_lock=on`” in the `quartus.ini` file and place in the file the Intel Quartus Prime project directory. You should see a warning message when you edit the IOPLL reconfiguration block (`pll_hdmi_reconfig`) in the Intel Quartus Prime software with the `INI` file. *Note:* Without this `Quartus INI` file, IOPLL reconfiguration cannot be completed if the IOPLL loses lock during reconfiguration.
PIO	The parallel input/output (PIO) block functions as control, status and reset interfaces to or from the CPU sub-system.

Figure 10. RX Reconfiguration Sequence Flow

The figure illustrates the multi-rate reconfiguration sequence flow of the controller when it receives input data stream and reference clock frequency, or when the transceiver is unlocked.



2.5.2.1. HDMI RX Top Link Training Process

This design example demonstrates the HDMI RX core request for LTP5, LTP6,, LTP7, and LTP8 link training patterns and qualifies the received data stream by the locked signal from the HDMI RX core.

These link training patterns start with 4 Scrambler Reset (SR) characters followed by 4096 encoded and scrambled data. After receiving the SR characters, the HDMI RX core achieves alignment and lane deskew lock to qualify the received link training pattern.

For unscrambled and unencoded link training patterns, such as LTP3, check the data output from the RX transceiver.

2.5.3. Top-Level Common Blocks

The top-level common blocks include the transceiver arbiter, the RX-TX link components, and the CPU subsystem.

Table 10. 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 memory-mapped 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 interface connection between the transceiver arbiter and TX/RX Native PHY/PHY Reset Controller blocks in this design example demonstrates a generic mode that applies for any IP combination using the transceiver arbiter. 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. For HDMI applications, only RX initiates reconfiguration. By channeling the Avalon memory-mapped reconfiguration request through the arbiter, the arbiter identifies that the reconfiguration request originates from the RX, which then gates `tx_reconfig_cal_busy` from asserting and allows `rx_reconfig_cal_busy` to assert. The gating prevents the TX transceiver from being moved to calibration mode unintentionally.

continued...

Module	Description
	Note: Because HDMI only requires RX reconfiguration, the `tx_reconfig_mgmt_*` signals are tied off. Also, the Avalon memory-mapped interface is not required between the arbiter and the TX Native PHY block. The blocks are assigned to the interface in the design example to demonstrate generic transceiver arbiter connection to TX/RX Native PHY/PHY Reset Controller.
RX-TX Link	• The video data output and synchronization signals from HDMI RX core loop through a DCFIFO across the RX and TX video clock domains. • The General Control Packet (GCP), InfoFrames (AVI, VSI and AI), auxiliary data, and audio data loop through DCFIFOs across the RX and TX link speed clock domains. • The auxiliary data port of the HDMI TX core controls the auxiliary data that flow through the DCFIFO through backpressure. The backpressure ensures there is no incomplete auxiliary packet on the auxiliary data port. • This block also performs external filtering: — Filters the audio data and audio clock regeneration packet from the auxiliary data stream before transmitting to the HDMI TX core auxiliary data port. — Filters the High Dynamic Range (HDR) InfoFrame from the HDMI RX auxiliary data and inserts an example HDR InfoFrame to the auxiliary data of the HDMI TX through the Avalon streaming multiplexer.
CPU Subsystem	The CPU subsystem functions as SCDC and DDC controllers, and source reconfiguration controller. • The source SCDC controller contains the I^2C master controller. The I^2C master controller transfers the SCDC data structure from the FPGA source to the external sink for HDMI 2.0 operation. For example, if the outgoing data stream is 6,000 Mbps, the Nios II processor commands the I^2C master controller to update the `TMDS_BIT_CLOCK_RATIO` and `SCRAMBLER_ENABLE` bits of the sink TMDS configuration register to 1. • The same I^2C master also transfers the DDC data structure (E-EDID) between the HDMI source and external sink. • The Nios II CPU acts as the reconfiguration controller for the HDMI source. The CPU relies on the periodic rate detection from the RX Reconfiguration Management module to determine if the TX requires reconfiguration. The Avalon memory-mapped slave translator provides the interface between the Nios II processor Avalon memory-mapped master interface and the Avalon memory-mapped slave interfaces of the externally instantiated HDMI source's IOPLL and TX Native PHY. • Perform link training through I^2C master interface with external sink

2.6. Dynamic Range and Mastering (HDR) InfoFrame Insertion and Filtering

The HDMI Intel FPGA IP design example includes a demonstration of HDR InfoFrame insertion in a RX-TX loopback system.

HDMI Specification version 2.0b allows Dynamic Range and Mastering InfoFrame to be transmitted through HDMI auxiliary stream. In the demonstration, the Auxiliary Data Insertion block supports the HDR insertion. You need only to format the intended HDR InfoFrame packet as specified in the module's signal list table and use the provided AUX Insertion Control module to schedule the insertion of the HDR InfoFrame once every video frame.

In this example configuration, in instances where the incoming auxiliary stream already includes HDR InfoFrame, the streamed HDR content is filtered. The filtering avoids conflicting HDR InfoFrames to be transmitted and ensures that only the values specified in the HDR Sample Data module are used.

Figure 11. RX-TX Link with Dynamic Range and Mastering InfoFrame Insertion

The figure shows the block diagram of RX-TX link including Dynamic Range and Mastering InfoFrame insertion into the HDMI TX core auxiliary stream.

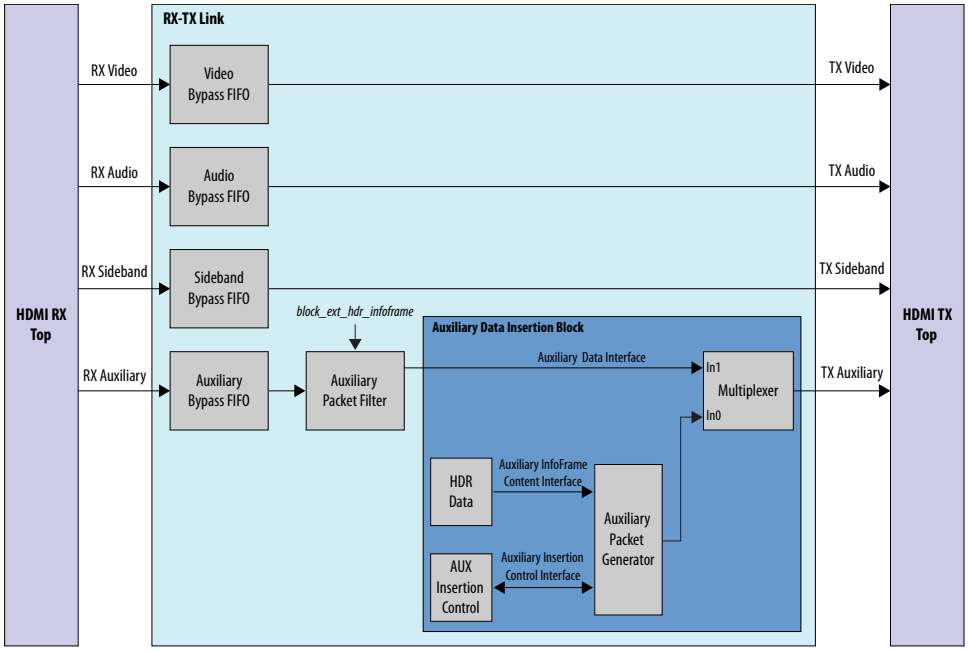


Table 11. Auxiliary Data Insertion Block (altera_hdmi_aux_hdr) Signals

Signal	Direction	Width	Description
Clock and Reset			
clk_clk	Input	1	Clock input. This clock should be connected to the link speed clock.
reset_reset_n	Input	1	Reset input.
Auxiliary Packet Generator and Multiplexer Signals			
multiplexer_out_data	Output	72	Avalon streaming output from the multiplexer.
multiplexer_out_valid	Output	1	
multiplexer_out_ready	Output	1	
multiplexer_out_startofpacket	Output	1	
multiplexer_out_endofpacket	Output	1	
multiplexer_out_channel	Output	11	
multiplexer_in_data	Input	72	Avalon streaming input to the In1 port of the multiplexer.
multiplexer_in_valid	Input	1	
multiplexer_in_ready	Input	1	
multiplexer_in_startofpacket	Input	1	
multiplexer_in_endofpacket	Input	1	

Control Signal			
hdmi_tx_vsync	Input	1	HDMI TX Video Vsync. This signal should be synchronized to the link speed clock domain. The core inserts the HDR InfoFrame to the auxiliary stream at the rising edge of this signal.

Table 12. HDR Data Module (altera_hdmi_hdr_infoframe) Signals

Signal	Direction	Width	Description
hb0	Output	8	Header byte 0 of the Dynamic Range and Mastering InfoFrame: InfoFrame type code.
hb1	Output	8	Header byte 1 of the Dynamic Range and Mastering InfoFrame: InfoFrame version number.
hb2	Output	8	Header byte 2 of the Dynamic Range and Mastering InfoFrame: Length of InfoFrame.
pb	Input	224	Data byte of the Dynamic Range and Mastering InfoFrame.

Table 13. Dynamic Range and Mastering InfoFrame Data Byte Bundle Bit-Fields

Bit-Field	Definition	Static Metadata Type 1
7:0	Data Byte 1: {5'h0, EOTF[2:0]}	
15:8	Data Byte 2: {5'h0, Static_Metadata_Descriptor_ID[2:0]}	
23:16	Data Byte 3: Static_Metadata_Descriptor	display_primaries_x[0], LSB
31:24	Data Byte 4: Static_Metadata_Descriptor	display_primaries_x[0], MSB
39:32	Data Byte 5: Static_Metadata_Descriptor	display_primaries_y[0], LSB
47:40	Data Byte 6: Static_Metadata_Descriptor	display_primaries_y[0], MSB
55:48	Data Byte 7: Static_Metadata_Descriptor	display_primaries_x[1], LSB
63:56	Data Byte 8: Static_Metadata_Descriptor	display_primaries_x[1], MSB
71:64	Data Byte 9: Static_Metadata_Descriptor	display_primaries_y[1], LSB
79:72	Data Byte 10: Static_Metadata_Descriptor	display_primaries_y[1], MSB
87:80	Data Byte 11: Static_Metadata_Descriptor	display_primaries_x[2], LSB
95:88	Data Byte 12: Static_Metadata_Descriptor	display_primaries_x[2], MSB
103:96	Data Byte 13: Static_Metadata_Descriptor	display_primaries_y[2], LSB
111:104	Data Byte 14: Static_Metadata_Descriptor	display_primaries_y[2], MSB
119:112	Data Byte 15: Static_Metadata_Descriptor	white_point_x, LSB
127:120	Data Byte 16: Static_Metadata_Descriptor	white_point_x, MSB
135:128	Data Byte 17: Static_Metadata_Descriptor	white_point_y, LSB

continued...

Bit-Field	Definition	Static Metadata Type 1
143:136	Data Byte 18: Static_Metadata_Descriptor	white_point_y, MSB
151:144	Data Byte 19: Static_Metadata_Descriptor	max_display_mastering_luminance, LSB
159:152	Data Byte 20: Static_Metadata_Descriptor	max_display_mastering_luminance, MSB
167:160	Data Byte 21: Static_Metadata_Descriptor	min_display_mastering_luminance, LSB
175:168	Data Byte 22: Static_Metadata_Descriptor	min_display_mastering_luminance, MSB
183:176	Data Byte 23: Static_Metadata_Descriptor	Maximum Content Light Level, LSB
191:184	Data Byte 24: Static_Metadata_Descriptor	Maximum Content Light Level, MSB
199:192	Data Byte 25: Static_Metadata_Descriptor	Maximum Frame-average Light Level, LSB
207:200	Data Byte 26: Static_Metadata_Descriptor	Maximum Frame-average Light Level, MSB
215:208	Reserved	
223:216	Reserved	

Disabling HDR Insertion and Filtering

Disabling HDR insertion and filter enables you to verify the retransmission of HDR content already available in the source auxiliary stream without any modification in the RX-TX Retransmit design example.

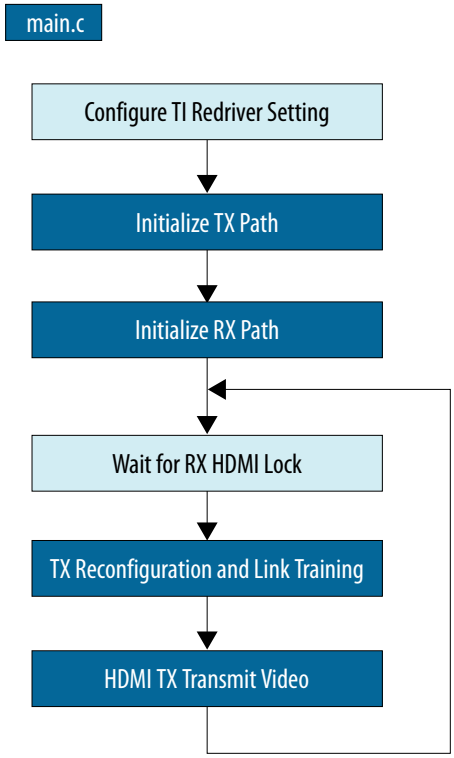
To disable HDR InfoFrame insertion and filtering:

1. Set `block_ext_hdr_infoframe` to `1'b0` in the `rxtx_link.v` file to prevent the filtering of the HDR InfoFrame from the Auxiliary stream.
2. Set `multiplexer_in0_valid` of the avalon_st_multiplexer instance in the `altera_hdmi_aux_hdr.v` file to `1'b0` to prevent the Auxiliary Packet Generator from forming and inserting additional HDR InfoFrame into the TX Auxiliary stream.

2.7. Design Software Flow

In the design main software flow, the Nios II processor configures the TI redriver setting and initializes the TX and RX paths upon power-up.

Figure 12. Software Flow in `main.c` Script



The software executes a while loop to monitor sink and source changes, and to react to the changes. The software may trigger TX reconfiguration, TX link training and start transmitting video.

Figure 13. TX Path Initialization Flowchart

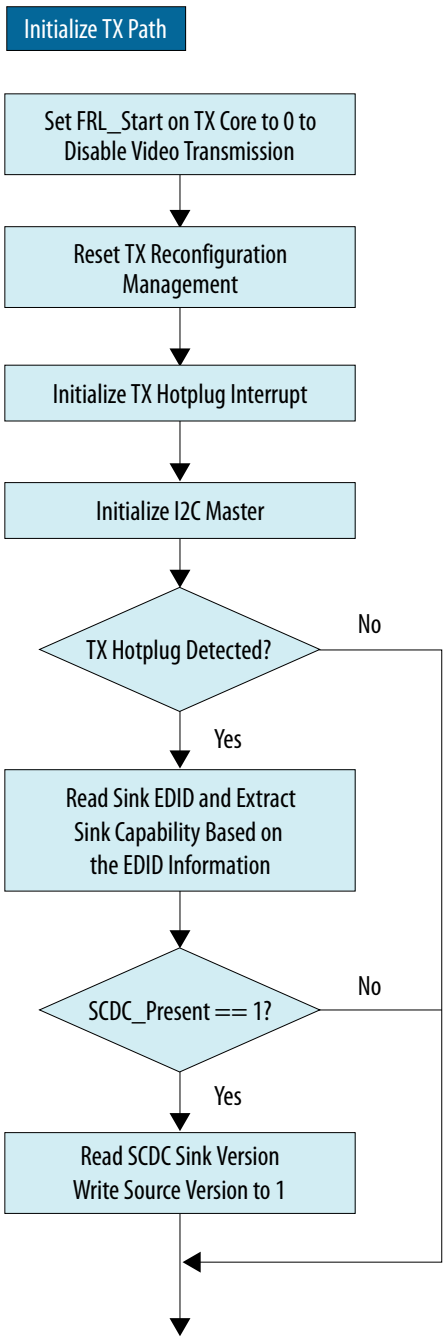


Figure 14. RX Path Initialization Flowchart

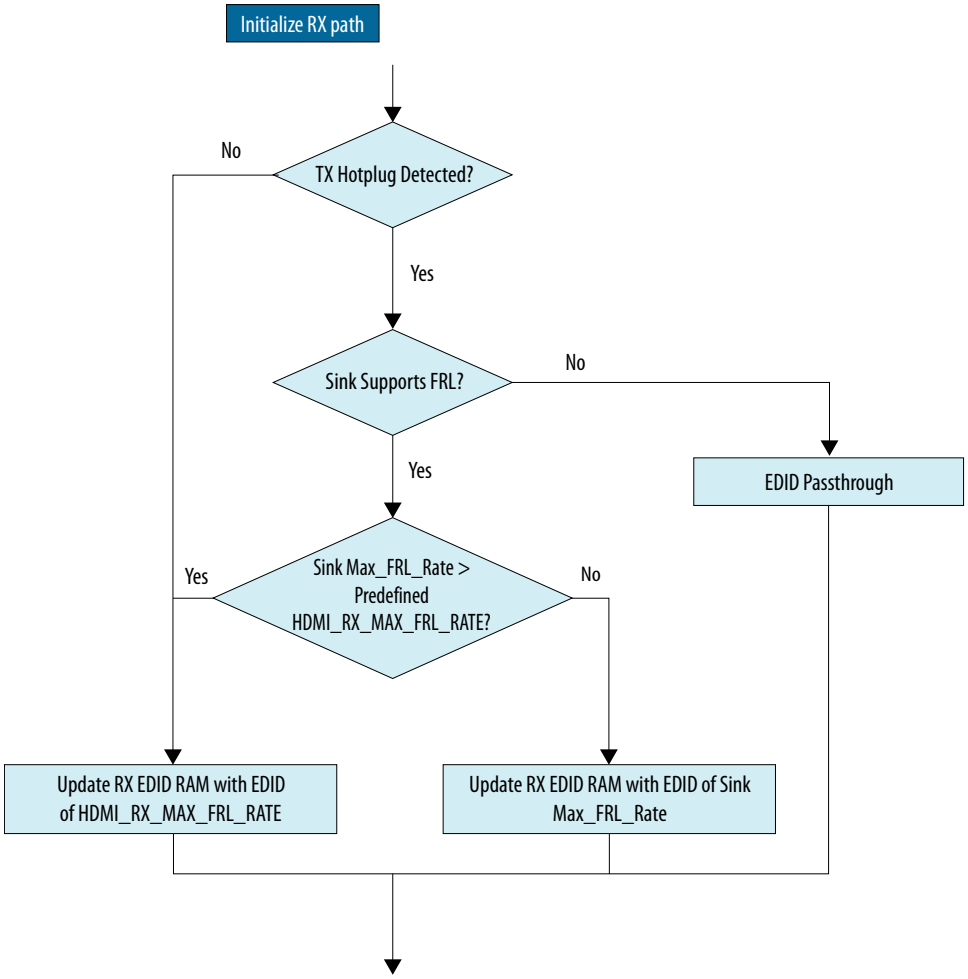


Figure 15. TX Reconfiguration and Link Training Flowchart

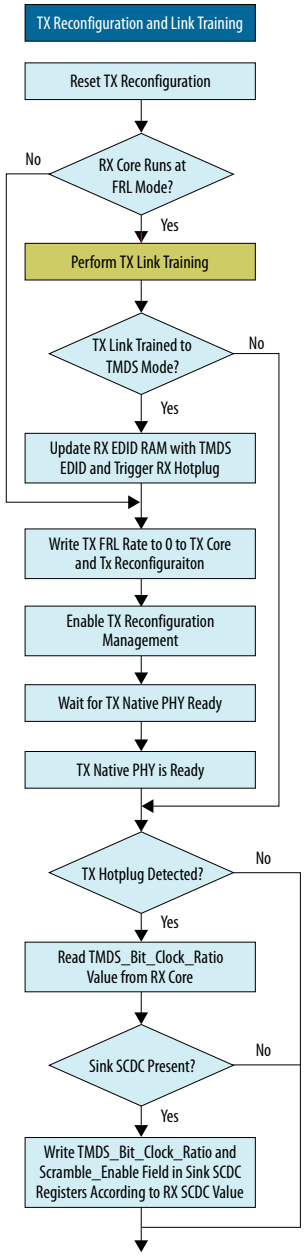


Figure 16. Link Training LTS:3 Process at Specific FRL Rate Flowchart

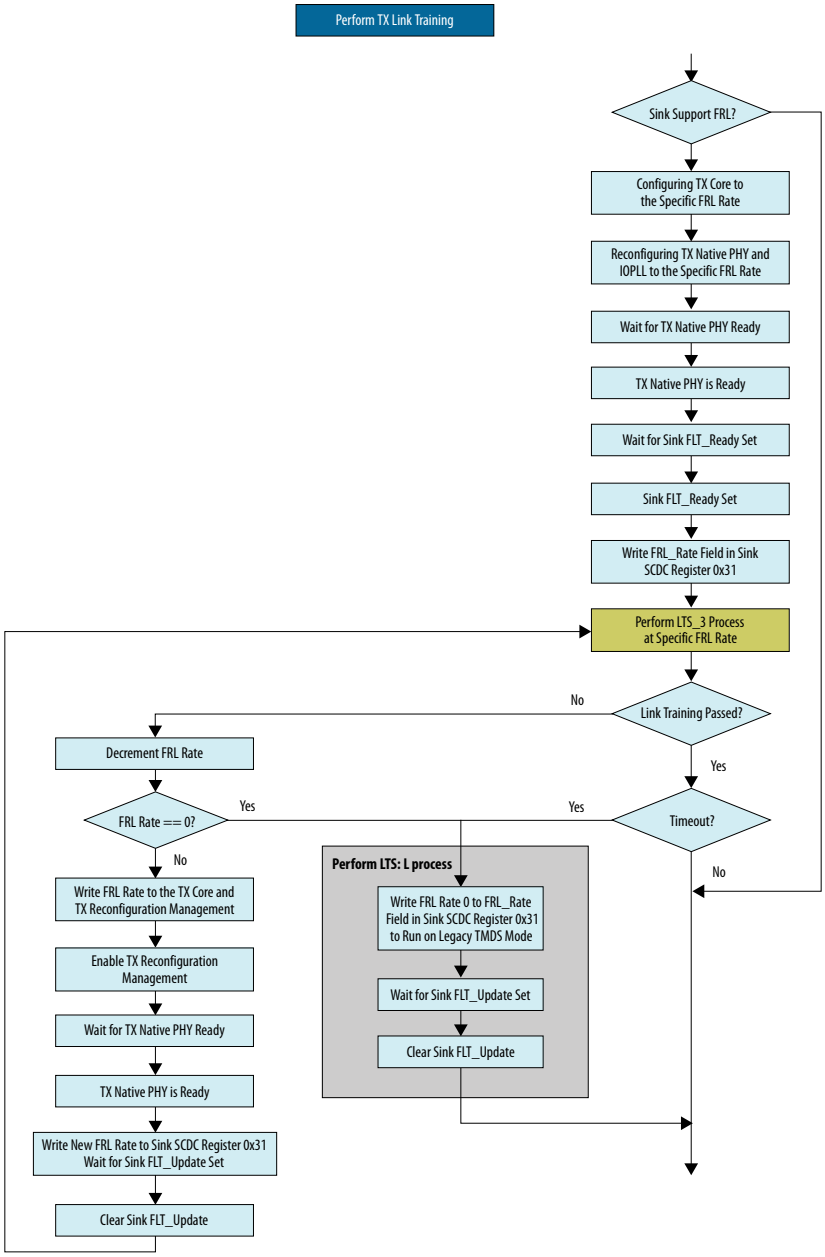
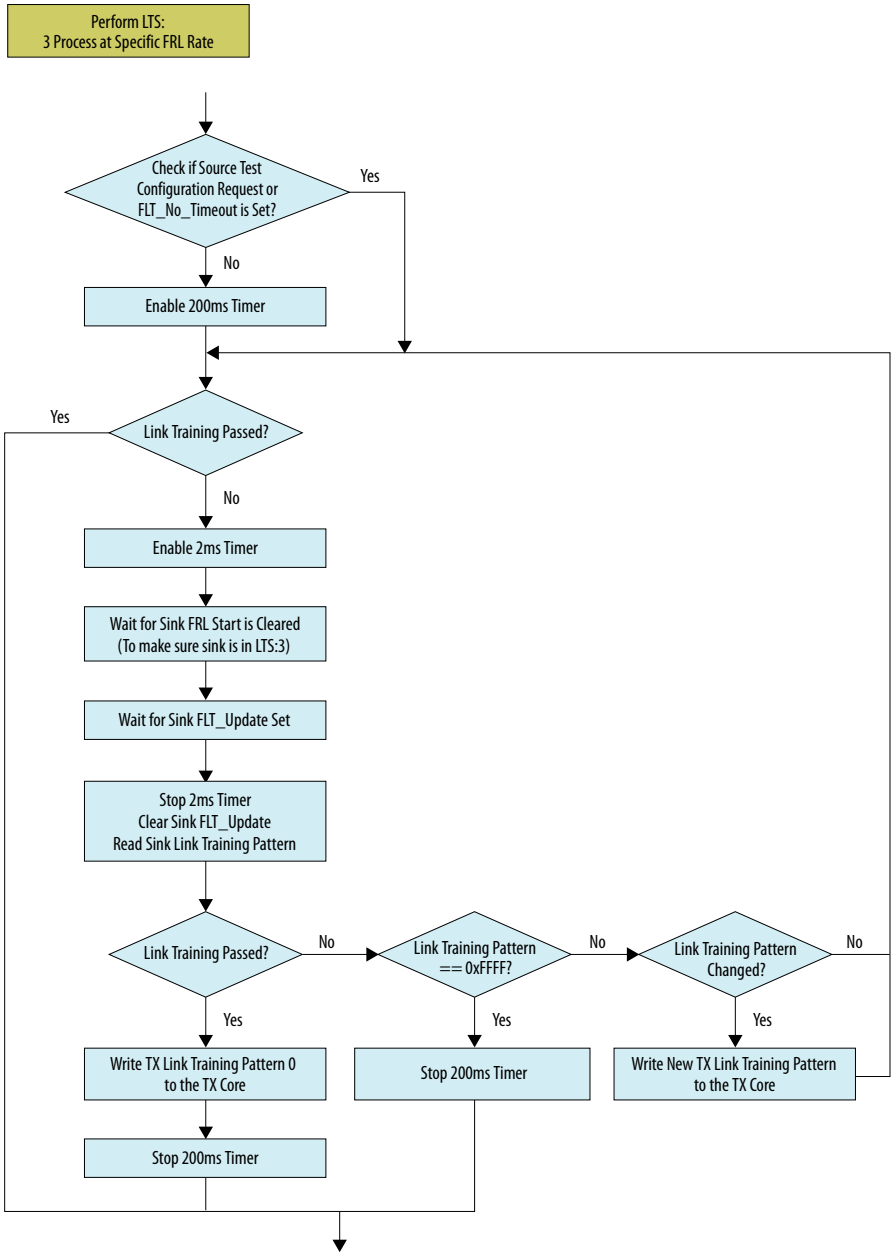


Figure 17. HDMI TX Video Transmission Flowchart



2.8. Modifying the Design to Support Different FRL Rates

In FRL mode, you modify the TX and RX designs to support different FRL rates,

Modifying the TX Design

In the TX design, you can modify the highest FRL rate for the TX link training to start with.

1. Change `bit[21:18]` of the `pio_in0_external_connection_export` script in the Platform Designer to the desired FRL rate.
2. Recompile the design in the Intel Quartus Prime Pro Edition software.
3. Program the SOF file onto the hardware.

Modifying the RX Design

In the RX design, you can modify the software to pre-configure the EDID RAM with the EDID of different FRL rates.

1. Change HDMI_RX_MAX_FRL_RATE in the `global.h` script in the Platform Designer to the desired FRL rate based on the values in the table below..

Table 14. FRL Rate Value

FRL Rate Value	Data Rate (Gbps)	Number of Lanes
6	12	4
5 (Default)	10	4
4	8	4
3	6	4
2	6	3
1	3	3
0	0.25–6	3

2. Run the command below to rebuild the software.

```
script/build_sw.h
```

3. Run the command below to program the software file.

```
nios2-download -r -g software/tx_control/tx_control.elf
```

4. Press CPU `resetn` to reset the design.

2.9. Clocking Scheme

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

Figure 18. HDMI 2.1 Design Example Clocking Scheme

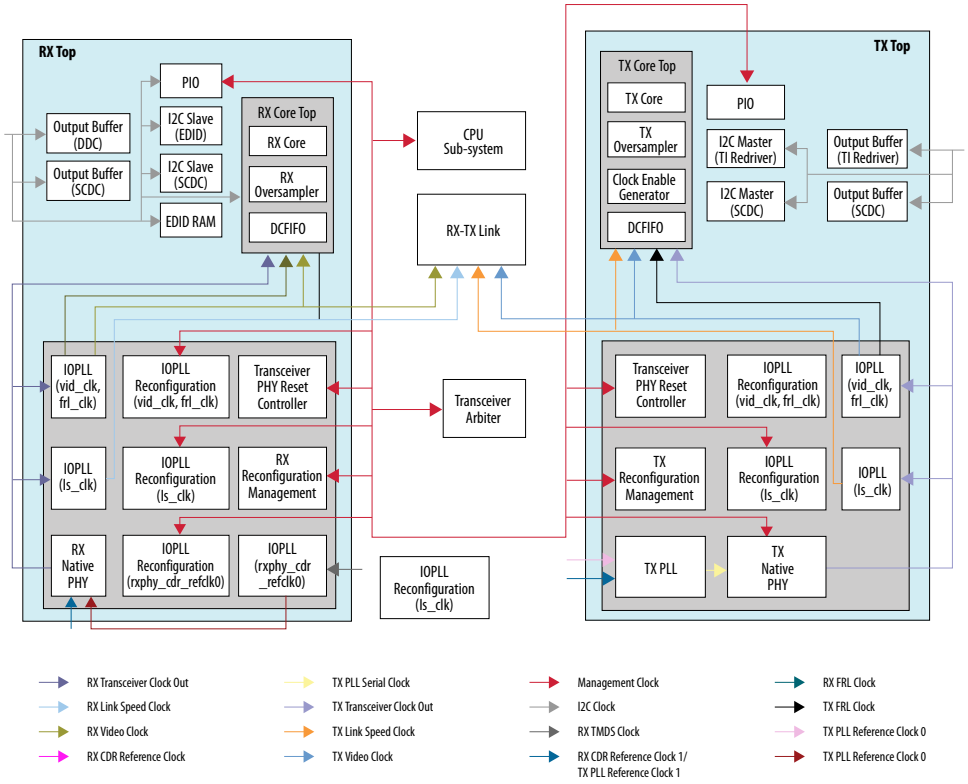


Table 15. Clocking Scheme Signals

Clock	Signal Name in Design	Description
Management Clock	`mgmt_clk`	A free running 100 MHz clock for these components: • Avalon-MM interfaces for reconfiguration — The frequency range requirement is between 100–125 MHz. • PHY reset controller for transceiver reset sequence — The frequency range requirement is between 1–500 MHz. • IOPLL Reconfiguration — The maximum clock frequency is 100 MHz. • RX Reconfiguration Management • TX Reconfiguration Management • CPU • I^2C Master
I^2C Clock	`i2c_clk`	A 100 MHz clock input that clocks I^2C slave, output buffers, SCDC registers, and link training process in the HDMI RX core, and EDID RAM.
TX PLL Reference Clock 0	`tx_tmds_clk`	Reference clock 0 to the TX PLL. The clock frequency is the same as the expected TMDS clock frequency from the HDMI TX TMDS clock channel. This reference clock is used in TMDS mode.

continued...

Clock	Signal Name in Design	Description
		For this HDMI design example, this clock is connected to the RX TMDS clock for demonstration purpose. In your application, you need to supply a dedicated clock with TMDS clock frequency from a programmable oscillator for better jitter performance. *Note:* Do not use a transceiver RX pin as a TX PLL reference clock. Your design will fail to fit if you place the HDMI TX refclk on an RX pin.
TX PLL Reference Clock 1	`txfpll_refclk1/ rxphy_cdr_refclk1`	Reference clock to the TX PLL and RX CDR. The clock frequency is 100 MHz. This reference clock is used in FRL mode.
TX PLL Serial Clock	`tx_bonding_clocks`	Serial fast clock generated by TX PLL. The clock frequency is set based on the data rate.
TX Transceiver Clock Out	`tx_clk`	Clock out recovered from the transceiver, and the frequency varies depending on the data rate and symbols per clock. TX transceiver clock out frequency = Transceiver data rate/ Transceiver width For this HDMI design example, the TX transceiver clock out from channel 0 clocks the TX transceiver core input (`tx_coreclkin`), link speed IOPLL (`pll_hdmi`) reference clock, and the video and FRL IOPLL (`pll_vid_frl`) reference clock.
TX/RX Video Clock	`tx_vid_clk/rx_vid_clk`	Video clock to TX and RX core.
TX/RX Link Speed Clock	`tx_ls_clk/rx_ls_clk`	Link speed clock to for TX and RX core.
TX/RX FRL Clock	`tx_frl_clk/rx_frl_clk`	FRL clock to for TX and RX core.
RX TMDS Clock	`rx_tmds_clk`	TMDS clock channel from the HDMI RX connector and connects to an IOPLL to generate the reference clock for CDR reference clock 0. The core uses this clock when it is in TMDS mode.
RX CDR Reference Clock 0	`rxphy_cdr_refclk0`	Reference clock 0 to RX CDR. This clock is derived from the RX TMDS clock. The RX TMDS clock frequency ranges from 25 MHz to 340 MHz while the RX CDR minimum reference clock frequency is 50 MHz. An IOPLL is used to generate a 5 clock frequency for the TMDS clock between 25 MHz to 50 MHz and generate the same clock frequency for the TMDS clock between 50 MHz - 340 MHz.
RX Transceiver Clock Out	`rx_clk`	Clock out recovered from the transceiver, and the frequency varies depending on the data rate and transceiver width. RX transceiver clock out frequency = Transceiver data rate/ Transceiver width For this HDMI design example, the RX transceiver clock out from channel 1 clocks the RX transceiver core input (`rx_coreclkin`), link speed IOPLL (`pll_hdmi`) reference clock, and the video and FRL IOPLL (`pll_vid_frl`) reference clock.

2.10. Interface Signals

The tables list the signals for the HDMI design example with FRL enabled.

Table 16. Top-Level Signals

Signal	Direction	Width	Description
On-board Oscillator Signal			
clk_fpga_b3_p	Input	1	100 MHz free running clock for core reference clock.
refclk4_p	Input	1	100 MHz free running clock for transceiver reference clock.
User Push Buttons and LEDs			
user_pb	Input	1	Push button to control the HDMI Intel FPGA IP design functionality
cpu_resetn	Input	1	Global reset
user_led_g	Output	8	Green LED display Refer to Hardware Setup on page 83 for more information about the LED functions.
HDMI FMC Daughter Card Pins on FMC Port B			
fmcb_gbtclk_m2c_p_0	Input	1	HDMI RX TMDS clock
fmcb_dp_m2c_p	Input	4	HDMI RX red, green, and blue data channels
fmcb_dp_c2m_p	Output	4	HDMI TX clock, red, green, and blue data channels
fmcb_la_rx_p_9	Input	1	HDMI RX +5V power detect
fmcb_la_rx_p_8	Inout	1	HDMI RX hot plug detect
fmcb_la_rx_n_8	Inout	1	HDMI RX I^2C SDA for DDC and SCDC
fmcb_la_tx_p_10	Input	1	HDMI RX I^2C SCL for DDC and SCDC
fmcb_la_tx_p_12	Input	1	HDMI TX hot plug detect
fmcb_la_tx_n_12	Inout	1	HDMI I^2C SDA for DDC and SCDC
fmcb_la_rx_p_10	Inout	1	HDMI I^2C SCL for DDC and SCDC
fmcb_la_tx_n_9	Inout	1	HDMI I^2C SDA for redriver control
fmcb_la_rx_p_11	Inout	1	HDMI I^2C SCL for redriver control

Table 17. HDMI RX Top-Level Signals

Signal	Direction	Width	Description
Clock and Reset Signals			
mgmt_clk	Input	1	System clock input (100 MHz)
reset	Input	1	System reset input
rx_tmds_clk	Input	1	HDMI RX TMDS clock
i2c_clk	Input	1	Clock input for DDC and SCDC interface
rxphy_cdr_refclk1	Input	1	Clock input for RX CDR reference clock 1. The clock frequency is 100 MHz.

continued...

Signal	Direction	Width	Description
Clock and Reset Signals			
rx_vid_clk	Output	1	Video clock output
rx_ls_clk	Output	1	Link speed clock output
sys_init	Output	1	System initialization to reset the system upon power-up
RX Transceiver and IOPLL Signals			
rxpll_ls_locked	Output	1	Indicates the link speed clock IOPLL is locked
rxpll_tmds_locked	Output	1	Indicates the TMDS clock IOPLL is locked
rxpll_vid_frl_locked	Output	1	Indicates the FRL clock IOPLL is locked
rxphy_serial_data	Input	3	HDMI serial data to the RX Native PHY
rxphy_ready	Output	1	Indicates the RX Native PHY is ready
rxphy_cal_busy_raw	Output	3	RX Native PHY calibration busy to the transceiver arbiter
rxphy_cal_busy_gated	Input	3	Calibration busy signal from the transceiver arbiter to the RX Native PHY
rxphy_rcfg_slave_write	Input	4	Transceiver reconfiguration Avalon memory-mapped interface from the RX Native PHY to the transceiver arbiter
rxphy_rcfg_slave_read	Input	4	
rxphy_rcfg_slave_address	Input	40	
rxphy_rcfg_slave_writedata	Input	128	
rxphy_rcfg_slave_readdata	Output	128	
rxphy_rcfg_slave_waitrequest	Output	4	
RX Reconfiguration Management			
rxphy_rcfg_busy	Output	1	RX Reconfiguration busy signal
rx_tmds_freq	Output	24	HDMI RX TMDS clock frequency measurement (in 10 ms)
rx_tmds_freq_valid	Output	1	Indicates the RX TMDS clock frequency measurement is valid
rxphy_os	Output	1	Oversampling factor: • 0: 1x oversampling • 1: 5× oversampling
rxphy_rcfg_master_write	Output	1	RX reconfiguration management Avalon memory-mapped interface to transceiver arbiter
rxphy_rcfg_master_read	Output	1	
rxphy_rcfg_master_address	Output	12	
rxphy_rcfg_master_writedata	Output	32	
rxphy_rcfg_master_readdata	Input	32	
rxphy_rcfg_master_waitrequest	Input	1	

HDMI RX Core Signals			
rxcore_frl_rate	Output	4	Indicates the FRL rate that the RX core is running. • 0: Legacy Mode (TMDS) • 1: 3 Gbps 3 lanes • 2: 6 Gbps 4 lanes • 3: 6 Gbps 4 lanes • 4: 8 Gbps 4 lanes • 5: 10 Gbps 4 lanes • 6: 12 Gbps 4 lanes • 7-15: Reserved
rxcore_frl_locked	Output	4	Each bit indicates the specific lane that has achieved FRL lock. FRL is locked when the RX core successfully performs alignment, deskew, and achieves lane lock. • For 3-lane mode, lane lock is achieved when the RX core receives Scrambler Reset (SR) or Start-Super-Block (SSB) for every 680 FRL character periods for at least 3 times. • For 4-lane mode, lane lock is achieved when the RX core receives Scrambler Reset (SR) or Start-Super-Block (SSB) for every 510 FRL character periods for at least 3 times.
rxcore_frl_ffe_levels	Output	4	Corresponds to the FFE_level bit in the SCDC 0x31 register bit [7:4] in the RX core.
rxcore_frl_flt_ready	Input	1	Asserts to indicate the RX is ready for the link training process to start. When asserted, the FLT_ready bit in the SCDC register 0x40 bit 6 is asserted as well.
rxcore_frl_src_test_config	Input	8	Specifies the source test configurations. The value is written into the SCDC Test Configuration register in the SCDC register 0x35.
rxcore_tbcr	Output	1	Indicates the TMDS bit to clock ratio; corresponds to the TMDS_Bit_Clock_Ratio register in the SCDC register 0x20 bit 1. • When running in HDMI 2.0 mode, this bit is asserted. Indicates the TMDS bit to clock ratio of 40:1 • When running in HDMI 1.4b, this bit is not asserted. Indicates the TMDS bit to clock ratio of 10:1 • This bit is unused for FRL mode.
rxcore_scrambler_enable	Output	1	Indicates if the received data is scrambled; corresponds to the Scrambling_Enable field in the SCDC register 0x20 bit 0.
rxcore_audio_de	Output	1	HDMI RX core audio interfaces Refer to the *Sink Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
rxcore_audio_data	Output	256	
rxcore_audio_info_ai	Output	48	
rxcore_audio_N	Output	20	

continued...

HDMI RX Core Signals			
rxcore_audio_CTS	Output	20	
rxcore_audio_metadata	Output	165	
rxcore_audio_format	Output	5	
rxcore_aux_pkt_data	Output	72	HDMI RX core auxiliary interfaces Refer to the *Sink Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
rxcore_aux_pkt_addr	Output	6	
rxcore_aux_pkt_wr	Output	1	
rxcore_aux_data	Output	72	
rxcore_aux_sop	Output	1	
rxcore_aux_eop	Output	1	
rxcore_aux_valid	Output	1	
rxcore_aux_error	Output	1	
rxcore_gcp	Output	6	HDMI RX core sideband signals Refer to the *Sink Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
rxcore_info_avi	Output	112	
rxcore_info_vsi	Output	61	
rxcore_locked	Output	1	HDMI RX core video ports *Note: N* = pixels per clock Refer to the *Sink Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
rxcore_vid_data	Output	*N**48	
rxcore_vid_vsync	Output	*N*	
rxcore_vid_hsync	Output	*N*	
rxcore_vid_de	Output	*N*	
rxcore_vid_valid	Output	1	
rxcore_vid_lock	Output	1	
rxcore_mode	Output	1	HDMI RX core control and status ports *Note: N* = symbols per clock Refer to the *Sink Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
rxcore_ctrl	Output	*N**6	
rxcore_color_depth_sync	Output	2	
hdmi_5v_detect	Input	1	HDMI RX 5V detect and hotplug detect Refer to the *Sink Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
hdmi_rx_hpd_n	Inout	1	
rx_hpd_trigger	Input	1	

I^2C Signals			
hdmi_rx_i2c_sda	Inout	1	HDMI RX DDC and SCDC interface
hdmi_rx_i2c_scl	Inout	1	

RX EDID RAM Signals			
edid_ram_access	Input	1	HDMI RX EDID RAM access interface.
edid_ram_address	Input	8	

continued...

RX EDID RAM Signals			
edid_ram_write	Input	1	Assert edid_ram_access when you want to write or read from the EDID RAM, else this signal should be kept low. When you assert edid_ram_access, the hotplug signal deasserts to allow write or read to the EDID RAM. When EDID RAM access is completed, you should deassert edid_ram_assess and the hotplug signal asserts. The source will read the new EDID due to the hotplug signal toggling.
edid_ram_read	Input	1	
edid_ram_readdata	Output	8	
edid_ram_writedata	Input	8	
edid_ram_waitrequest	Output	1	

Table 18. HDMI TX Top-Level Signals

Signal	Direction	Width	Description
Clock and Reset Signals			
mgmt_clk	Input	1	System clock input (100 MHz)
reset	Input	1	System reset input
tx_tmds_clk	Input	1	HDMI RX TMDS clock
txfpll_refclk1	Input	1	Clock input for RX CDR reference clock 1. The clock frequency is 100 MHz.
tx_vid_clk	Output	1	Video clock output
tx_ls_clk	Output	1	Link speed clock output
tx_frl_clk	Output	1	FRL clock output
sys_init	Output	1	System initialization to reset the system upon power-up
tx_init_done	Input	1	TX initialization to reset the TX reconfiguration management block and transceiver reconfiguration interface
TX Transceiver and IOPLL Signals			
txpll_ls_locked	Output	1	Indicates the link speed clock IOPLL is locked
txpll_vid_frl_locked	Output	1	Indicates the video and FRL clock IOPLL is locked
txfpll_locked			Indicates the TX PLL is locked
txphy_serial_data	Output	4	HDMI serial data from the TX Native PHY
txphy_ready	Output	1	Indicates the TX Native PHY is ready
txphy_cal_busy	Output	1	TX Native PHY calibration busy signalr
txphy_cal_busy_raw	Output	4	Calibration busy signal to the transceiver arbiter
txphy_cal_busy_gated	Input	4	Calibration busy signal from the transceiver arbiter to the TX Native PHY
txphy_rcfg_busy	Output	1	Indicates the TX PHY reconfiguration is in progress
txphy_rcfg_slave_write	Input	4	Transceiver reconfiguration Avalon memory-mapped interface from the TX

continued...

TX Transceiver and IOPLL Signals			
txphy_rcfg_slave_read	Input	4	Native PHY to the transceiver arbiter
txphy_rcfg_slave_address	Input	40	
txphy_rcfg_slave_writedata	Input	128	
txphy_rcfg_slave_readdata	Output	128	
txphy_rcfg_slave_waitrequest	Output	4	

TX Reconfiguration Management			
tx_tmds_freq	Input	24	HDMI TX TMDS clock frequency value (in 10 ms)
tx_os	Output	2	Oversampling factor: • 0: 1x oversampling • 1: 2× oversampling ... • 8: 8x oversampling
txphy_rcfg_master_write	Output	1	TX reconfiguration management Avalon memory-mapped interface to transceiver arbiter
txphy_rcfg_master_read	Output	1	
txphy_rcfg_master_address	Output	12	
txphy_rcfg_master_writedata	Output	32	
txphy_rcfg_master_readdata	Input	32	
txphy_rcfg_master_waitrequest	Input	1	

TX IOPLL and TX PLL Reconfiguration Signals			
pll_reconfig_write/ tx_pll_reconfig_write	Input	1	TX IOPLL/TX PLL reconfiguration Avalon-MM interfaces
pll_reconfig_read/ tx_pll_reconfig_read	Input	1	
pll_reconfig_address/ tx_pll_reconfig_address	Input	10	
pll_reconfig_writedata/ tx_pll_reconfig_writedata	Input	32	
pll_reconfig_readdata/ tx_pll_reconfig_readdata	Output	32	
pll_reconfig_waitrequest/ tx_pll_reconfig_waitrequest	Output	1	
os	Input	2	Oversampling factor: • 0: No oversampling • 1: 3× oversampling • 2: 4× oversampling • 3: 5× oversampling
measure	Input	24	Indicates the TMDS clock frequency of the transmitting video resolution.

HDMI TX Core Signals			
txcore_ctrl	Input	*N**6	HDMI TX core control interfaces *Note: N* = pixels per clock Refer to the *Source Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
txcore_mode	Input	1	
txcore_audio_de	Input	1	HDMI TX core audio interfaces Refer to the *Source Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
txcore_audio_mute	Input	1	
txcore_audio_data	Input	256	
txcore_audio_info_ai	Input	49	
txcore_audio_N	Input	20	
txcore_audio_CTS	Input	20	
txcore_audio_metadata	Input	166	
txcore_audio_format	Input	5	
txcore_aux_ready	Output	1	HDMI TX core auxiliary interfaces Refer to the *Source Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
txcore_aux_data	Input	72	
txcore_aux_sop	Input	1	
txcore_aux_eop	Input	1	
txcore_aux_valid	Input	1	
txcore_gcp	Input	6	HDMI TX core sideband signals Refer to the *Source Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
txcore_info_avi	Input	122	
txcore_info_vsi	Input	62	
txcore_vid_data	Input	*N**48	HDMI TX core video ports *Note: N* = pixels per clock Refer to the *Source Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
txcore_vid_vsync	Input	*N*	
txcore_vid_hsync	Input	*N*	
txcore_vid_de	Input	*N*	
txcore_vid_ready	Output	1	
txcore_vid_overflow	Output	1	
txcore_vid_valid	Input	1	
txcore_frl_rate	Input	4	SCDC register interfaces
txcore_frl_pattern	Input	16	
txcore_frl_start	Input	1	
txcore_scrambler_enable	Input	1	
txcore_tbhr	Input	1	

I²C Signals			
nios_tx_i2c_sda_in	Output	1	TX I²C Master interface for SCDC and DDC from the Nios II processor to the output buffer
nios_tx_i2c_scl_in	Output	1	
nios_tx_i2c_sda_oe	Input	1	
nios_tx_i2c_scl_oe	Input	1	
nios_ti_i2c_sda_in	Output	1	TX I²C Master interface from the Nios II processor to the output buffer to control TI redriver on the Bitec HDMI 2.1 FMC daughter card
nios_ti_i2c_scl_in	Output	1	
nios_ti_i2c_sda_oe	Input	1	
nios_ti_i2c_scl_oe	Input	1	
hdmi_tx_i2c_sda	Inout	1	TX I²C interfaces for SCDC and DDC interfaces from the output buffer to the HDMI TX connector
hdmi_tx_i2c_scl	Inout	1	
hdmi_tx_ti_i2c_sda	Inout	1	TX I²C interfaces from the output buffer to the TI redriver on the Bitec HDMI 2.1 FMC daughter card
hdmi_tx_ti_i2c_scl	Inout	1	

Hotplug Detect Signals			
tx_hpd_ack	Input	1	HDMI TX hotplug detect interfaces
tx_hpd_req	Output	1	
hdmi_tx_hpd_n	Input	1	

Table 19. 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 memory-mapped 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	

continued...

Signal	Direction	Width	Description
rx_reconfig_mgmt_waitrequest	Output	1	
tx_reconfig_mgmt_write	Input	1	Reconfiguration Avalon memory-mapped interfaces from the TX reconfiguration management
tx_reconfig_mgmt_read	Input	1	
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 memory-mapped 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 20. RX-TX Link Signals

Signal	Direction	Width	Description
reset	Input	1	Reset to the video/audio/auxiliary/ sidebands FIFO buffer.
mgmt_clk	Input	1	100 MHz clock
i2c_clk	Input	1	I^2C clock
tx_ls_clk	Input	1	HDMI TX link speed clock
rx_ls_clk	Input	1	HDMI RX link speed clock
tx_vid_clk	Input	1	HDMI TX video clock
rx_vid_clk	Input	1	HDMI RX video clock
sys_init	Input	1	System initialization to reset the system upon power-up
rx_hdmi_locked	Input	3	Indicates HDMI RX locked status
tx_src_sel	Input	2	Reserved

continued...

Signal	Direction	Width	Description
rx_vid_valid	Input	1	HDMI RX video interfaces
rx_vid_de	Input	*N*	
rx_vid_hsync	Input	*N*	
rx_vid_vsync	Input	*N*	
rx_vid_data	Input	*N**48	
rx_audio_format	Input	5	HDMI RX audio interfaces
rx_audio_metadata	Input	165	
rx_audio_info_ai	Input	48	
rx_audio_CTS	Input	20	
rx_audio_N	Input	20	
rx_audio_de	Input	1	
rx_audio_data	Input	256	
rx_gcp	Input	6	HDMI RX sideband interfaces
rx_info_avi	Input	112	
rx_info_vsi	Input	61	
rx_aux_eop	Input	1	HDMI RX auxiliary interfaces
rx_aux_sop	Input	1	
rx_aux_valid	Input	1	
rx_aux_data	Input	72	
rx_tbcr	Input	1	HDMI RX SCDC registers
rx_scrambler_enable	Input	1	
tx_vid_ready	Input	1	HDMI TX video interfaces *Note: N* = pixels per clock
tx_vid_de	Output	*N*	
tx_vid_hsync	Output	*N*	
tx_vid_vsync	Output	*N*	
tx_vid_data	Output	*N*48*	
tx_audio_format	Output	5	HDMI TX audio interfaces
tx_audio_metadata	Output	165	
tx_audio_info_ai	Output	48	
tx_audio_CTS	Output	20	
tx_audio_N	Output	20	
tx_audio_de	Output	1	
tx_audio_data	Output	256	
tx_gcp	Output	6	HDMI TX sideband interfaces

continued...

Signal	Direction	Width	Description
tx_info_avi	Output	112	
tx_info_vsi	Output	61	
tx_aux_eop	Output	1	HDMI TX auxiliary interfaces
tx_aux_sop	Output	1	
tx_aux_valid	Output	1	
tx_aux_data	Output	72	
tx_aux_ready	Input	1	
tx_tbcr	Output	1	HDMI TX SCDC registers
tx_scrambler_enable	Output	1	

Table 21. Platform Designer System Signals

Signal	Direction	Width	Description
cpu_clk_in_clk_clk	Input	1	CPU clock
cpu_rst_in_reset_reset	Input	1	CPU reset
edid_ram_slave_translator_avalon_anti_slave_0_address	Output	8	EDID RAM access interfaces.
edid_ram_slave_translator_avalon_anti_slave_0_write	Output	1	
edid_ram_slave_translator_avalon_anti_slave_0_read	Output	1	
edid_ram_slave_translator_avalon_anti_slave_0_readdata	Input	8	
edid_ram_slave_translator_avalon_anti_slave_0_writedata	Output	8	
edid_ram_slave_translator_avalon_anti_slave_0_waitrequest	Input	1	
hdmi_i2c_master_i2c_serial_sda_in	Input	1	I^2C Master interfaces from the Nios II processor to the output buffer for DDC and SCDC control
hdmi_i2c_master_i2c_serial_scl_in	Input	1	
hdmi_i2c_master_i2c_serial_sda_oe	Output	1	
hdmi_i2c_master_i2c_serial_scl_oe	Output	1	
redriver_i2c_master_i2c_serial_sda_in	Input	1	I^2C Master interfaces from the Nios II processor to the output buffer for TI redriver setting configuration
redriver_i2c_master_i2c_serial_scl_in	Input	1	
redriver_i2c_master_i2c_serial_sda_oe	Output	1	
redriver_i2c_master_i2c_serial_scl_oe	Output	1	

continued...

Signal	Direction	Width	Description
pio_in0_external_connection_export	Input	32	Parallel input output interfaces • Bit 0 : Reserved • Bit 1: TX HPD request • Bit 2: TX transceiver ready • Bits 3–7: Reserved • Bits 8–11: RX FRL rate • Bit 12: RX TMDS bit clock ratio • Bits 13–16: RX FRL locked • Bits 17–20: RX FFE levels • Bit 21: RX alignment locked • Bit 22: RX video lock • Bit 23: User push button 2 to read SCDC registers from external sink • Bits 24–31: Reserved
pio_out0_external_connection_export	Output	32	Parallel input output interfaces • Bit 0: TX HPD acknowledgment • Bit 1: TX initialization is done • Bits 2–7: Reserved • Bits 8–11: TX FRL rate • Bits 12–27: TX FRL link training pattern • Bit 28: TX FRL start • Bits 29–31: Reserved
pio_out1_external_connection_export	Output	32	Parallel input output interfaces • Bit 0 : RX EDID RAM access • Bit 1: RX FLT ready • Bits 2–7: Reserved • Bits 8–15: RX FRL source test configuration • Bits 16–31: Reserved

2.11. Design RTL Parameters

Use the HDMI TX and RX Top RTL parameters to customize the design example.

Most of the design parameters are available in the **Design Example** tab of the HDMI Intel FPGA IP parameter editor. You can still change the design example settings you made in the parameter editor through the RTL parameters.

Table 22. HDMI RX Top Parameters

Parameter	Value	Description
SUPPORT_DEEP_COLOR	• 0: No deep color • 1: Deep color	Determines if the core can encode deep color formats. *Note:* This feature is not supported in FRL mode.
SUPPORT_AUXILIARY	• 0: No AUX • 1: AUX	Determines if the auxiliary channel encoding is included.
SYMBOLS_PER_CLOCK	8	Supports 8 symbols per clock for Intel Arria 10 devices.
SUPPORT_AUDIO	• 0: No audio • 1: Audio	Determines if the core can encode audio.

Table 23. HDMI TX Top Parameters

Parameter	Value	Description
SUPPORT_DEEP_COLOR	• 0: No deep color • 1: Deep color	Determines if the core can encode deep color formats. *Note:* This feature is not supported in FRL mode.
SUPPORT_AUXILIARY	• 0: No AUX • 1: AUX	Determines if the auxiliary channel encoding is included.
SYMBOLS_PER_CLOCK	8	Supports 8 symbols per clock for Intel Arria 10 devices.
SUPPORT_AUDIO	• 0: No audio • 1: Audio	Determines if the core can encode audio.

2.12. Hardware Setup

The HDMI FRL-enabled design example is HDMI 2.1 capable and performs a loop-through demonstration for a standard HDMI video stream.

To run the hardware test, connect an HDMI-enabled device—such as a graphics card with HDMI interface—to the Transceiver Native PHY RX block, and the HDMI sink input. The design supports both HDMI 2.1 or HDMI 2.0/1.4b source and sink.

1. The HDMI sink decodes the port into a standard video stream and sends it to the clock recovery core.
2. The HDMI RX core decodes the video, auxiliary, and audio data to be looped back in parallel to the HDMI TX core through the DCFIFO.
3. The HDMI source port of the FMC daughter card transmits the image to a monitor.

Note: If you want to use another Intel FPGA development board, you must change the device assignments and the pin assignments. The transceiver analog setting is tested for the Intel Arria 10 FPGA development kit and Bitec HDMI 2.0 daughter card. You may modify the settings for your own board.

Table 24. On-board Push Button and User LED Functions

Push Button/LED	Function
cpu_resetn	Press once to perform system reset.
user_pb[0]	Press once to toggle the HPD signal to the standard HDMI source.
user_pb[1]	Reserved.
user_pb[2]	Press once to read the SCDC registers from the sink connected to the TX of the Bitec HDMI 2.1 FMC daughter card. *Note:* To enable read, you must set DEBUG_MODE to 1 in the software.
USER_LED[0]	RX TMDS clock PLL lock status. • 0 = Unlocked • 1 = Locked
USER_LED[1]	RX transceiver ready status. • 0 = Not ready • 1 = Ready
USER_LED[2]	RX link speed clock PLL, and RX video and FRL clock PLL lock status. • 0 = Either one of the RX clock PLL is unlocked • 1 = Both RX clock PLLs are locked
USER_LED[3]	RX HDMI core alignment and deskew lock status. • 0 = At least 1 channel is unlocked • 1 = All channels are locked
USER_LED[4]	RX HDMI video lock status. • 0 = Unlocked • 1 = Locked
USER_LED[5]	TX link speed clock PLL, and TX video and FRL clock PLL lock status. • 0 = Either one of the TX clock PLL is unlocked • 1 = Both TX clock PLLs are locked
USER_LED[6]	TX transceiver ready status. • 0 = Not ready • 1 = Ready
USER_LED[7]	TX link training status. • 0 = Failed • 1 = Passed

2.13. Design Limitations

You need to consider some limitations when instantiating the HDMI 2.1 design example.

- Ensure that both the RX and TX instances run at the same FRL rate.
- The Nios II processor must serve the TX link training to completion without any interruption from other processes.
- For this release, the design does not support deep color mode when running on FRL mode.

2.14. Debugging Features

This design example provides certain debugging features to assist you.

2.14.1. Software Debugging Message

You can turn on the debugging message in the software to provide you run-time assistance.

To turn on the debugging message in the software, follow these steps:

1. Change the DEBUG_MODE to 1 in the `global.h` script.
2. Run `script/build_sw.sh` on the Nios II Command Shell.
3. Reprogram the generated `software/tx_control/tx_control.elf` file by running the command on the Nios II Command Shell:

```
nios2-download -r -g software/tx_control/tx_control.elf
```

4. Run the Nios II terminal command on the Nios II Command Shell:

```
nios2-terminal
```

When you turn on the debugging message, the following information print out:

- TI redriver settings on both TX and RX are read and displayed once after programming ELF file.
- Status message for RX EDID configuration and hotplug process
- Resolution with or without FRL support information extracted from EDID on the sink connected to the TX. This information is displayed for every TX hotplug.
- Status message for the TX link training process during TX link training.

2.14.2. SCDC Information from the Sink Connected to TX

You can use this feature to obtain SCDC information.

1. Run the Nios II terminal command on the Nios II Command Shell:

```
nios2-terminal
```

2. Press `user_pb[2]` on the Intel Arria 10 FPGA development kit.

The software reads and displays the SCDC information on the sink connected to TX on the Nios II terminal.

2.14.3. Clock Frequency Measurement

Use this feature to check the frequency for the different clocks.

1. In the `hdmi_rx_top` and `hdmi_tx_top` files, uncomment "`//`define DEBUG_EN 1".
2. Add the `refclock_measure` signal from each `mr_rate_detect` instance to the Signal Tap Logic Analyzer to get the clock frequency of each clock (in 10 ms duration).
3. Compile the design with Signal Tap Logic Analyzer.
4. Program the SOF file and run the Signal Tap Logic Analyzer.

Table 25. Clocks

Module	mr_rate_detect Instance	Clock to be Measured
hdmi_rx_top	rx_pll_tmds	RX CDR reference clock 0
	rx_clk0_freq	RX transceiver clock out from channel 0
	rx_ls_clk_freq	RX link speed clock
	rx_vid_clk_freq	RX video clock
	rx_frl_clk_freq	RX FRL clock
	rx_hsync_freq	Hsync frequency of the received video frame
hdmi_tx_top	tx_clk0_freq	TX transceiver clock out from channel 0
	ls_clk_freq	TX link speed clock
	vid_clk_freq	TX video clock
	frl_clk_freq	TX FRL clock
	tx_hsync_freq	Hsync frequency of the video frame to be transmitted

3. Detailed Description for HDMI 2.0 Design Example

The HDMI Intel FPGA IP design example demonstrates one HDMI instance parallel loopback comprising three RX channels and four TX channels.

Table 26. HDMI Intel FPGA IP Design Example for Intel Arria 10 Devices

Design Example	Data Rate	Channel Mode	Loopback Type
Arria 10 HDMI RX-TX Retransmit	< 6,000 Mbps	Simplex	Parallel with FIFO buffer

Features

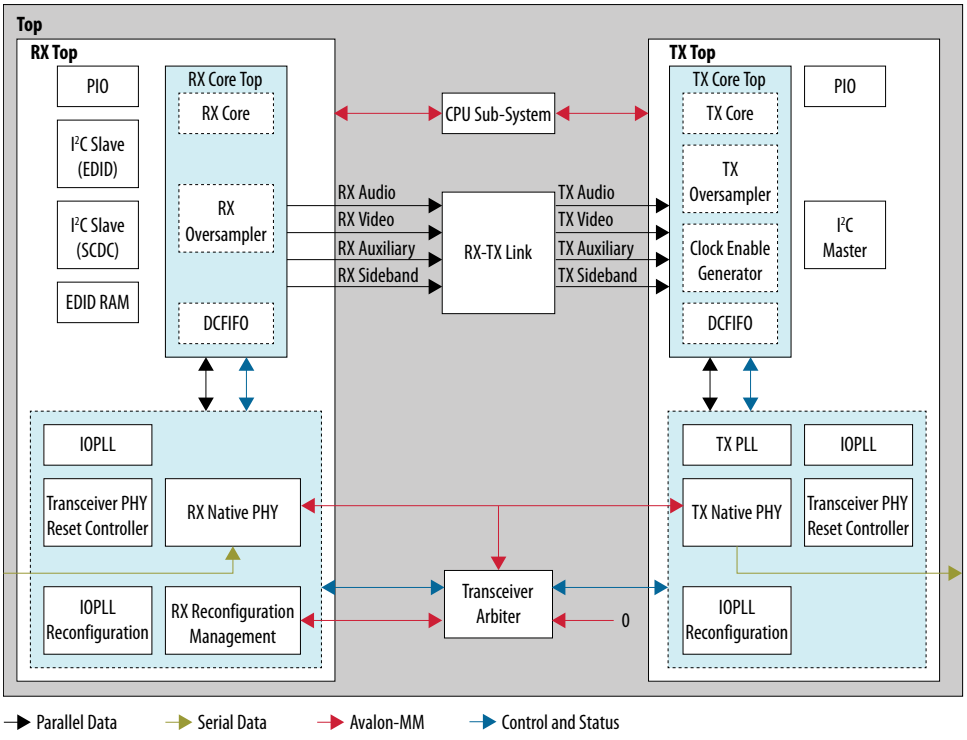
- The design instantiates FIFO buffers to perform a direct HDMI video stream passthrough between the HDMI sink and source.
- The design uses LED status for early debugging stage.
- The design comes with RX and TX only options.
- The design demonstrates the insertion and filtering of Dynamic Range and Mastering (HDR) InfoFrame in RX-TX link module.
- The design demonstrates the management of EDID passthrough from an external HDMI sink to an external HDMI source when triggered by a TX hot-plug event.
- The design uses push-button controlled HDMI TX core signals:
  - `mode` signal to select DVI or HDMI encoded video frame
  - `info_avi[47]`, `info_vsi[61]`, and `audio_info_ai[48]` signals to select auxiliary packet transmission through sidebands or auxiliary data ports

The RX instance receives a video source from the external video generator, and the data then goes through a loopback FIFO before it is transmitted to the TX instance. You need to connect an external video analyzer, monitor, or a television with HDMI connection to the TX core to verify the functionality.

3.1. HDMI 2.0 RX-TX Retransmit Design Block Diagram

The HDMI 2.0 RX-TX retransmit design example demonstrates parallel loopback on simplex channel mode for HDMI Intel FPGA IP.

Figure 19. HDMI RX-TX Retransmit Block Diagram



Related Information

Jitter of PLL Cascading or Non-Dedicated Clock Path for Arria 10 PLL Reference Clock
Refer to this solution for workaround if your design clocks experience additional jitter.

3.2. Hardware and Software Requirements

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

Hardware

- Intel Arria 10 GX FPGA Development Kit
- HDMI Source (Graphics Processor Unit (GPU))
- HDMI Sink (Monitor)
- Bitec HDMI FMC 2.0 daughter card (Revision 11)
- HDMI cables

Note: You can select the revision of the Bitec HDMI daughter card by setting the local parameter `BITEC_DAUGHTER_CARD_REV` to 4, 6, or 11 in the top-level file (`a10_hdmi2_demo.v`). When you change the revision, the design may swap the transceiver channels and invert the polarity according to the Bitec HDMI daughter card requirements. If you set the `BITEC_DAUGHTER_CARD_REV` parameter to 0, the design does not make any changes to the transceiver channels and the polarity.

Software

- Intel Quartus Prime version 18.1 (for hardware testing)
- ModelSim - Intel FPGA Edition, ModelSim - Intel FPGA Starter Edition, NCSim, Riviera-PRO, VCS (Verilog HDL only)/VCS MX, or Xcelium Parallel simulator

3.3. Directory Structure

The directories contain the generated files for the HDMI Intel FPGA IP design example.

Figure 20. Directory Structure for the Design Example

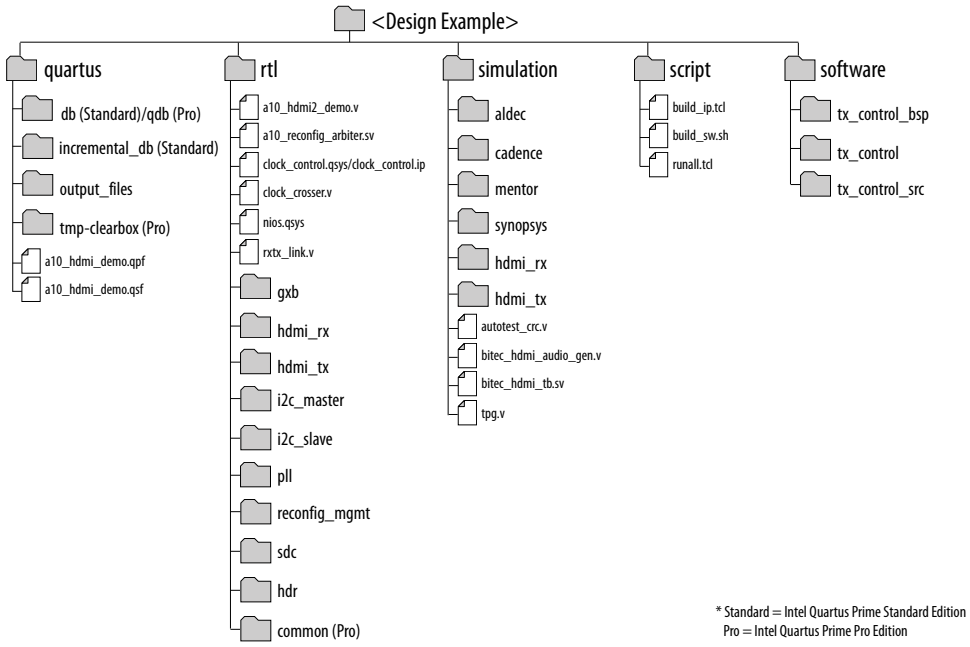


Table 27. Generated RTL Files

Folders	Files
gxb	• /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)
	• /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)
	• /gxb_tx_reset.qsys (Intel Quartus Prime Standard Edition) • /gxb_tx_reset.ip (Intel Quartus Prime Pro Edition)
hdmi_rx	• /hdmi_rx.qsys (Intel Quartus Prime Standard Edition) • /hdmi_rx.ip (Intel Quartus Prime Pro Edition)
	/hdmi_rx_top.v

continued...

Folders	Files
	/mr_clock_sync.v
	/mr_hdmi_rx_core_top.v
	/mr_rx_oversample.v
	/symbol_aligner.v
hdmi_tx	• /hdmi_tx.qsys (Intel Quartus Prime Standard Edition) • /hdmi_tx.ip (Intel Quartus Prime Pro Edition)
	/hdmi_tx_top.v
	/mr_ce.v
	/mr_hdmi_tx_core_top.v
	/mr_tx_oversample.v
i2c_master	/i2c_master_bit_ctrl.v
	/i2c_master_byte_ctrl.v
	/i2c_master_defines.v
	/i2c_master_top.v
	/oc_i2c_master.v
	/oc_i2c_master_hw.tcl
	/timescale.v
i2c_slave	• /edid_ram.qsys (Intel Quartus Prime Standard Edition) • /edid_ram.ip (Intel Quartus Prime Pro Edition)
	/I2Cslave.v
	• /output_buf_i2c.qsys (Intel Quartus Prime Standard Edition) • /output_buf_i2c.ip (Intel Quartus Prime Pro Edition)
	/Panasonic.hex
	/i2c_avl_mst_intf_gen.v
	/i2c_clk_cnt.v
	/i2c_condt_det.v
	/i2c_databuffer.v
	/i2c_rxshifter.v
	/i2c_slvfsm.v
	/i2c_spksupp.v
	/i2c_txout.v
	/i2c_txshifter.v
	/i2cslave_to_avlmm_bridge.v
pll	• /pll_hdmi.qsys (Intel Quartus Prime Standard Edition) • /pll_hdmi.ip (Intel Quartus Prime Pro Edition)

continued...

Folders	Files
	• /pll_hdmi_reconfig.qsys (Intel Quartus Prime Standard Edition) • /pll_hdmi_reconfig.ip (Intel Quartus Prime Pro Edition)
common	/reset_controller (Intel Quartus Prime Pro Edition)
hdr	/altera_hdmi_aux_hdr.v
	/altera_hdmi_aux_snk.v
	/altera_hdmi_aux_src.v
	/altera_hdmi_hdr_infoframe.v
	/avalon_st_mutiplexer.v
reconfig_mgmt	/mr_compare_pll.v
	/mr_compare_rx.v
	/mr_rate_detect.v
	/mr_reconfig_master_pll.v
	/mr_reconfig_master_rx.v
	/mr_reconfig_mgmt.v
	/mr_rom_pll_dprioaddr.v
	/mr_rom_pll_valuemask_8bpc.v
	/mr_rom_pll_valuemask_10bpc.v
	/mr_rom_pll_valuemask_12bpc.v
	/mr_rom_pll_valuemask_16bpc.v
	/mr_rom_rx_dprioaddr_bitmask.v
	/mr_rom_rx_valuemask.v
	/mr_state_machine.v
sdc	/a10_hdmi2.sdc
	/mr.sdc
	/jtag.sdc

Table 28. Generated Simulation Files

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

continued...

Folders	Files
mentor	/mentor.do
	/msim_setup.tcl
synopsys	/vcs/filelist.f
	/vcs/vcs_setup.sh
	/vcs/vcs_sim.sh
	/vcsmx/vcsmx_setup.sh
	/vcsmx/vcsmx_sim.sh
	/vcsmx/synopsys_sim_setup
hdmi_rx	• /hdmi_rx.qsys (Intel Quartus Prime Standard Edition) • /hdmi_rx.ip (Intel Quartus Prime Pro Edition)
	/hdmi_rx.sopcinfo (Intel Quartus Prime Standard Edition)
hdmi_tx	• /hdmi_tx.qsys (Intel Quartus Prime Standard Edition) • /hdmi_tx.ip (Intel Quartus Prime Pro Edition)
	/hdmi_tx.sopcinfo (Intel Quartus Prime Standard Edition)

Table 29. Generated Software Files

Folders	Files
tx_control_src *Note:* The tx_control folder also contains duplicates of these files.	/i2c.c
	/i2c.h
	/main.c
	/xcvr_gpll_rcfg.c
	/xcvr_gpll_rcfg.h
	/ti_i2c.c
	/ti_i2c.h

3.4. Design Components

The HDMI Intel FPGA IP design example requires these components.

Table 30. HDMI RX Top Components

Module	Description
RX Core Top	The RX Core top level consists of:

continued...

Module	Description
	• HDMI RX Core—The IP receives the serial data from the Transceiver Native PHY and performs data alignment, channel deskew, TMDS decoding, auxiliary data decoding, video data decoding, audio data decoding, and descrambling. • RX Oversampler—The RX Oversampler module extracts data from the oversampled incoming data stream when the detected clock frequency band is below the transceiver minimum link rate. The oversampling factor is fixed at 5 and you can program the data width to support different number of symbols. For Intel Arria 10 devices, the supported data width is 20 bits for 2 symbols per clock. The extracted bit is accompanied by a data valid pulse asserted every 5 clock cycles. • DCFIFO —The DCFIFO transfers data from the RX transceiver recovered clock domain to the RX link speed clock domain.
I^2C Slave	I^2C is the interface used for Sink Display Data Channel (DDC) and Status and Data Channel (SCDC). The HDMI source uses the DDC to determine the capabilities and characteristics of the sink by reading the Enhanced Extended Display Identification Data (E-EDID) data structure. • The 8-bit I^2C slave addresses for E-EDID are 0xA0 and 0xA1. The LSB indicates the access type: 1 for read and 0 for write. When an HPD event occurs, the I^2C slave responds to E-EDID data by reading from the on-chip RAM. • The I^2C slave-only controller also supports SCDC for HDMI 2.0 operations. The 8-bit I^2C slave address for the SCDC are 0xA8 and 0xA9. When an HPD event occurs, the I^2C slave performs write or read transaction to or from SCDC interface of the HDMI RX core. *Note:* This I^2C slave-only controller for SCDC is not required if HDMI 2.0b is not intended.
EDID RAM	The design stores the EDID information using the RAM 1-port IP core. A standard two-wire (clock and data) serial bus protocol (I^2C slave-only controller) transfers the CEA-861-D Compliant E-EDID data structure. This EDID RAM stores the E-EDID information.
IOPLL	The IOPLL generates the RX CDR reference clock, link speed clock, and video clock for the incoming TMDS clock. • Output clock 0 (CDR reference clock) • Output clock 1 (Link speed clock) • Output clock 2 (Video clock) *Note:* The default IOPLL configuration is not valid for any HDMI resolution. The IOPLL is reconfigured to the appropriate settings upon power up.
Transceiver PHY Reset Controller	The Transceiver PHY reset controller ensures a reliable initialization of the RX transceivers. The reset input of this controller is triggered by the RX reconfiguration, and it generates the corresponding analog and digital reset signal to the Transceiver Native PHY block according to the reset sequencing inside the block.
RX Native PHY	Hard transceiver block that receives the serial data from an external video source. It deserializes the serial data to parallel data before passing the data to the HDMI RX core.
RX Reconfiguration Management	RX reconfiguration management that implements rate detection circuitry with the HDMI PLL to drive the RX transceiver to operate at any arbitrary link rates ranging from 250 Mbps to 6,000 Mbps. Refer to Figure 21 on page 61 below.
IOPLL Reconfiguration	IOPLL reconfiguration block facilitates dynamic real-time reconfiguration of PLLs in Intel FPGAs. This block updates the output clock frequency and PLL bandwidth in real time, without reconfiguring the entire FPGA. This block runs at 100 MHz in Intel Arria 10 devices. Due to IOPLL reconfiguration limitation, apply the `Quartus INI permit_nf_pll_reconfig_out_of_lock=on` during the IOPLL reconfiguration IP generation.

continued...

Module	Description
	To apply the Quartus INI, include "permit_nf_pll_reconfig_out_of_lock=on" in the quartus.ini file and place in the file the Intel Quartus Prime project directory. You should see a warning message when you edit the IOPLL reconfiguration block (pll_hdmi_reconfig) in the Quartus Prime software with the INI. *Note:* Without this Quartus INI, IOPLL reconfiguration cannot be completed if the IOPLL loses lock during reconfiguration.
PIO	The parallel input/output (PIO) block functions as control, status and reset interfaces to or from the CPU sub-system.

Figure 21. Multi-Rate Reconfiguration Sequence Flow

The figure illustrates the multi-rate reconfiguration sequence flow of the controller when it receives input data stream and reference clock frequency, or when the transceiver is unlocked.

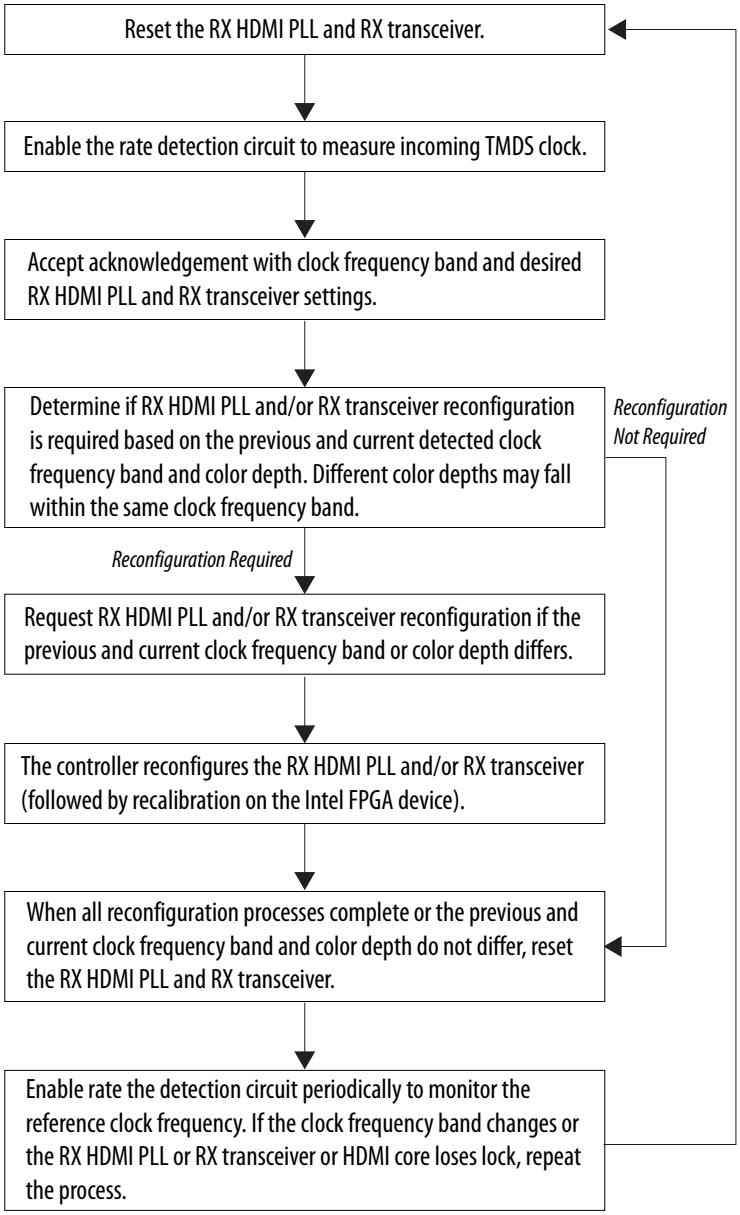


Table 31. HDMI TX Top Components

Module	Description
TX Core Top	The TX Core top level consists of:

continued...

Module	Description
	• HDMI TX Core—The IP core receives video data from the top level and performs TMDS encoding, auxiliary data encoding, audio data encoding, video data encoding, and scrambling. • TX Oversampler—The TX Oversampler module transmits data by repeating each bit of the input word a given number of times and constructs the output words. The oversampling factor can be 3, 4, or 5 depending on the TMDS clock frequency. The TX oversampler assumes that the input word is only valid every 3, 4, or 5 clock cycles according to the oversampling factor. This block is enabled when the outgoing data stream is below the minimum link rate of the TX transceiver. When FPLL input reference clock frequency is below 50 MHz, the allowable data rate must be below 1,500 Mbps. (Refer to Table 32 on page 63.) • DCFIFO —The DCFIFO transfers data from the TX link speed clock domain to the transceiver parallel clock out domain. When the Nios II processor determines the outgoing data stream is below the TX transceiver minimum data rate, the TX transceiver accepts the data from the TX oversampler. Otherwise, the TX transceiver reads the data directly from the DCFIFO with a read request asserted at all times. • Clock Enable Generator—A logic that generates a clock enable pulse. This clock enable pulse asserts every 5 clock cycles and serves as a read request signal to clock the data out from the DCFIFO.
I^2C Master	I^2C is the interface used for Sink Display Data Channel (DDC) and Status and Data Channel (SCDC). The HDMI source uses the DDC to determine the capabilities and characteristics of the sink by reading the Enhanced Extended Display Identification Data (E-EDID) data structure. • As DDC, I^2C Master reads the EDID from the external sink to configure the EDID information EDID RAM in the HDMI RX Top or for video processing. • As SCDC, I^2C master transfers the SCDC data structure from the FPGA source to the external sink for HDMI 2.0b operation. For example, if the outgoing data stream is above 3,400 Mbps, the Nios II processor commands the I^2C master to update the `TMDS_BIT_CLOCK_RATIO` and `SCRAMBLER_ENABLE` bits of the sink SCDC configuration register to 1.
IOPLL	The IOPLL supplies the link speed clock and video clock from the incoming TMDS clock. • Output clock 1 (Link speed clock) • Output clock 2 (Video clock) *Note:* The default IOPLL configuration is not valid for any HDMI resolution. The IOPLL is reconfigured to the appropriate settings upon power up.
Transceiver PHY Reset Controller	The Transceiver PHY reset controller ensures a reliable initialization of the TX transceivers. The reset input of this controller is triggered from the top level, and it generates the corresponding analog and digital reset signal to the Transceiver Native PHY block according to the reset sequencing inside the block. The `tx_ready` output signal from this block also functions as a reset signal to the HDMI Intel FPGA IP to indicate the transceiver is up and running, and ready to receive data from the core.
Transceiver Native PHY	Hard transceiver block that receives the parallel data from the HDMI TX core and serializes the data from transmitting it. Reconfiguration interface is enabled in the TX Native PHY block to demonstrate the connection between TX Native PHY and transceiver arbiter. No reconfiguration is performed for TX Native PHY. *Note:* To meet the HDMI TX inter-channel skew requirement, set the TX channel bonding mode option in the Intel Arria 10 Transceiver Native PHY parameter editor to **PMA and PCS bonding**. You also need to add the maximum skew (`set_max_skew`) constraint requirement to the digital reset signal from the transceiver reset controller (`tx_digitalreset`) as recommended in the *Intel Arria 10 Transceiver PHY User Guide.*

continued...

Module	Description
TX PLL	The transmitter PLL block provides the serial fast clock to the Transceiver Native PHY block. For this HDMI Intel FPGA IP design example, fPLL is used as TX PLL.
IOPLL Reconfiguration	IOPLL reconfiguration block facilitates dynamic real-time reconfiguration of PLLs in Intel FPGAs. This block updates the output clock frequency and PLL bandwidth in real time, without reconfiguring the entire FPGA. This block runs at 100 MHz in Intel Arria 10 devices. Due to IOPLL reconfiguration limitation, apply the `Quartus INI permit_nf_pll_reconfig_out_of_lock=on` during the IOPLL reconfiguration IP generation. To apply the `Quartus INI`, include "`permit_nf_pll_reconfig_out_of_lock=on`" in the `quartus.ini` file and place in the file the Intel Quartus Prime project directory. You should see a warning message when you edit the IOPLL reconfiguration block (pll_hdmi_reconfig) in the Intel Quartus Prime software with the `INI`. *Note:* Without this `Quartus INI`, IOPLL reconfiguration cannot be completed if the IOPLL loses lock during reconfiguration.
PIO	The parallel input/output (PIO) block functions as control, status and reset interfaces to or from the CPU sub-system.

Table 32. Transceiver Data Rate and Oversampling Factor for Each TMDS Clock Frequency Range

TMDS Clock Frequency (MHz)	TMDS Bit clock Ratio	Oversampling Factor	Transceiver Data Rate (Mbps)
85–150	1	Not applicable	3400–6000
100–340	0	Not applicable	1000–3400
50–100	0	5	2500–5000
35–50	0	3	1050–1500
30–35	0	4	1200–1400
25–30	0	5	1250–1500

Table 33. Top-Level Common Blocks

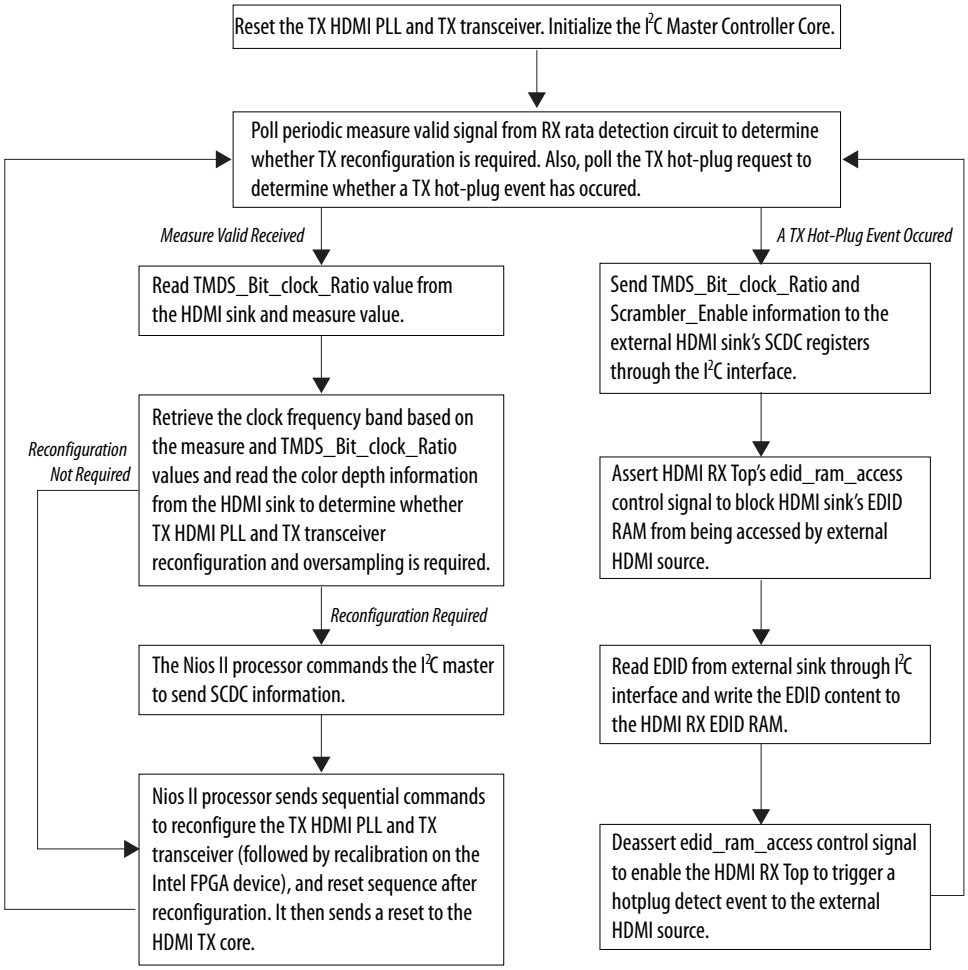
Module	Description
Transceiver Arbiter	This generic functional block prevents transceivers from recalibrating simultaneously when either RX or TX transceivers within the same physical channel require reconfiguration. The simultaneous recalibration impacts applications where RX and TX transceivers within 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 interface connection between the transceiver arbiter and TX/RX Native PHY/PHY Reset Controller blocks in this design example demonstrates a generic mode that apply for any IP combination using the transceiver arbiter. 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.

continued...

Module	Description
	For HDMI application, only RX initiates reconfiguration. By channeling the Avalon-MM reconfiguration request through the arbiter, the arbiter identifies that the reconfiguration request originates from the RX, which then gates `tx_reconfig_cal_busy` from asserting and allows `rx_reconfig_cal_busy` to assert. The gating prevents the TX transceiver from being moved to calibration mode unintentionally. *Note:* Because HDMI only requires RX reconfiguration, the `tx_reconfig_mgmt_*` signals are tied off. Also, the Avalon-MM interface is not required between the arbiter and the TX Native PHY block. The blocks are assigned to the interface in the design example to demonstrate generic transceiver arbiter connection to TX/RX Native PHY/PHY Reset Controller.
RX-TX Link	• The video data output and synchronization signals from HDMI RX core loop through a DCFIFO across the RX and TX video clock domains. • The General Control Packet (GCP), InfoFrames (AVI, VSI and AI), auxiliary data, and audio data loop through DCFIFOs across the RX and TX link speed clock domains. • The auxiliary data port of the HDMI TX core controls the auxiliary data that flow through the DCFIFO through backpressure. The backpressure ensures there is no incomplete auxiliary packet on the auxiliary data port. • This block also performs external filtering: — Filters the audio data and audio clock regeneration packet from the auxiliary data stream before transmitting to the HDMI TX core auxiliary data port. *Note:* To disable this filtering, press `user_pb[2]`. Enable this filtering to ensure there is no duplication of audio data and audio clock regeneration packet in the retransmitted auxiliary data stream. — Filters the High Dynamic Range (HDR) InfoFrame from the HDMI RX auxiliary data and inserts an example HDR InfoFrame to the auxiliary data of the HDMI TX through the Avalon ST multiplexer.
CPU Sub-System	The CPU sub-system functions as SCDC and DDC controllers, and source reconfiguration controller. • The source SCDC controller contains the I^2C master controller. The I^2C master controller transfers the SCDC data structure from the FPGA source to the external sink for HDMI 2.0b operation. For example, if the outgoing data stream is 6,000 Mbps, the Nios II processor commands the I^2C master controller to update the `TMDS_BIT_CLOCK_RATIO` and `SCRAMBLER_ENABLE` bits of the sink TMDS configuration register to 1. • The same I^2C master also transfers the DDC data structure (E-EDID) between the HDMI source and external sink. • The Nios II CPU acts as the reconfiguration controller for the HDMI source. The CPU relies on the periodic rate detection from the RX Reconfiguration Management module to determine if the TX requires reconfiguration. The Avalon-MM slave translator provides the interface between the Nios II processor Avalon-MM master interface and the Avalon-MM slave interfaces of the externally instantiated HDMI source's IOPLL and TX Native PHY. • The reconfiguration sequence flow for TX is same as RX, except that the PLL and transceiver reconfiguration and the reset sequence is performed sequentially. Refer to Figure 22 on page 65.

Figure 22. Reconfiguration Sequence Flow

The figure illustrates the Nios II software flow that involves the controls for I²C master and HDMI source.



3.5. Dynamic Range and Mastering (HDR) InfoFrame Insertion and Filtering

The HDMI Intel FPGA IP design example includes a demonstration of HDR InfoFrame insertion in a RX-TX loopback system.

HDMI Specification version 2.0b allows Dynamic Range and Mastering InfoFrame to be transmitted through HDMI auxiliary stream. In the demonstration, the Auxiliary Data Insertion block supports the HDR insertion. You need only to format the intended HDR InfoFrame packet as specified in the module's signal list table and use the provided AUX Insertion Control module to schedule the insertion of the HDR InfoFrame once every video frame.

In this example configuration, in instances where the incoming auxiliary stream already includes HDR InfoFrame, the streamed HDR content is filtered. The filtering avoids conflicting HDR InfoFrames to be transmitted and ensures that only the values specified in the HDR Sample Data module are used.

Figure 23. RX-TX Link with Dynamic Range and Mastering InfoFrame Insertion

The figure shows the block diagram of RX-TX link including Dynamic Range and Mastering InfoFrame insertion into the HDMI TX core auxiliary stream.

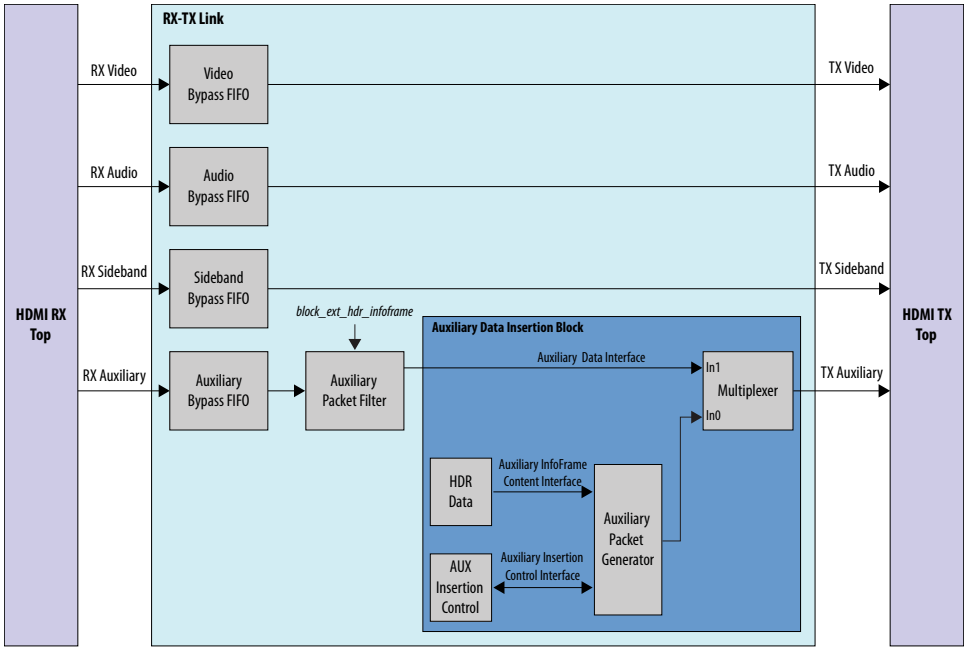


Table 34. Auxiliary Data Insertion Block (altera_hdmi_aux_hdr) Signals

Signal	Direction	Width	Description
Clock and Reset			
clk_clk	Input	1	Clock input. This clock should be connected to the link speed clock.
reset_reset_n	Input	1	Reset input.

Auxiliary Packet Generator and Multiplexer Signals			
multiplexer_out_data	Output	72	Avalon streaming output from the multiplexer.
multiplexer_out_valid	Output	1	
multiplexer_out_ready	Output	1	
multiplexer_out_startofpacket	Output	1	
multiplexer_out_endofpacket	Output	1	
multiplexer_out_channel	Output	11	
multiplexer_in_data	Input	72	Avalon streaming input to the In1 port of the multiplexer.
multiplexer_in_valid	Input	1	
multiplexer_in_ready	Input	1	
multiplexer_in_startofpacket	Input	1	
multiplexer_in_endofpacket	Input	1	

Control Signal			
hdmi_tx_vsync	Input	1	HDMI TX Video Vsync. This signal should be synchronized to the link speed clock domain. The core inserts the HDR InfoFrame to the auxiliary stream at the rising edge of this signal.

Table 35. HDR Data Module (altera_hdmi_hdr_infoframe) Signals

Signal	Direction	Width	Description
hb0	Output	8	Header byte 0 of the Dynamic Range and Mastering InfoFrame: InfoFrame type code.
hb1	Output	8	Header byte 1 of the Dynamic Range and Mastering InfoFrame: InfoFrame version number.
hb2	Output	8	Header byte 2 of the Dynamic Range and Mastering InfoFrame: Length of InfoFrame.
pb	Input	224	Data byte of the Dynamic Range and Mastering InfoFrame.

Table 36. Dynamic Range and Mastering InfoFrame Data Byte Bundle Bit-Fields

Bit-Field	Definition	Static Metadata Type 1
7:0	Data Byte 1: {5'h0, EOTF[2:0]}	
15:8	Data Byte 2: {5'h0, Static_Metadata_Descriptor_ID[2:0]}	
23:16	Data Byte 3: Static_Metadata_Descriptor	display_primaries_x[0], LSB
31:24	Data Byte 4: Static_Metadata_Descriptor	display_primaries_x[0], MSB
39:32	Data Byte 5: Static_Metadata_Descriptor	display_primaries_y[0], LSB
47:40	Data Byte 6: Static_Metadata_Descriptor	display_primaries_y[0], MSB
55:48	Data Byte 7: Static_Metadata_Descriptor	display_primaries_x[1], LSB
63:56	Data Byte 8: Static_Metadata_Descriptor	display_primaries_x[1], MSB
71:64	Data Byte 9: Static_Metadata_Descriptor	display_primaries_y[1], LSB
79:72	Data Byte 10: Static_Metadata_Descriptor	display_primaries_y[1], MSB
87:80	Data Byte 11: Static_Metadata_Descriptor	display_primaries_x[2], LSB
95:88	Data Byte 12: Static_Metadata_Descriptor	display_primaries_x[2], MSB
103:96	Data Byte 13: Static_Metadata_Descriptor	display_primaries_y[2], LSB
111:104	Data Byte 14: Static_Metadata_Descriptor	display_primaries_y[2], MSB
119:112	Data Byte 15: Static_Metadata_Descriptor	white_point_x, LSB
127:120	Data Byte 16: Static_Metadata_Descriptor	white_point_x, MSB
135:128	Data Byte 17: Static_Metadata_Descriptor	white_point_y, LSB

continued...

Bit-Field	Definition	Static Metadata Type 1
143:136	Data Byte 18: Static_Metadata_Descriptor	white_point_y, MSB
151:144	Data Byte 19: Static_Metadata_Descriptor	max_display_mastering_luminance, LSB
159:152	Data Byte 20: Static_Metadata_Descriptor	max_display_mastering_luminance, MSB
167:160	Data Byte 21: Static_Metadata_Descriptor	min_display_mastering_luminance, LSB
175:168	Data Byte 22: Static_Metadata_Descriptor	min_display_mastering_luminance, MSB
183:176	Data Byte 23: Static_Metadata_Descriptor	Maximum Content Light Level, LSB
191:184	Data Byte 24: Static_Metadata_Descriptor	Maximum Content Light Level, MSB
199:192	Data Byte 25: Static_Metadata_Descriptor	Maximum Frame-average Light Level, LSB
207:200	Data Byte 26: Static_Metadata_Descriptor	Maximum Frame-average Light Level, MSB
215:208	Reserved	
223:216	Reserved	

Disabling HDR Insertion and Filtering

Disabling HDR insertion and filter enables you to verify the retransmission of HDR content already available in the source auxiliary stream without any modification in the RX-TX Retransmit design example.

To disable HDR InfoFrame insertion and filtering:

1. Set `block_ext_hdr_infoframe` to `1'b0` in the `rxtx_link.v` file to prevent the filtering of the HDR InfoFrame from the Auxiliary stream.
2. Set `multiplexer_in0_valid` of the avalon_st_multiplexer instance in the `altera_hdmi_aux_hdr.v` file to `1'b0` to prevent the Auxiliary Packet Generator from forming and inserting additional HDR InfoFrame into the TX Auxiliary stream.

3.6. Clocking Scheme

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

Figure 24. HDMI Intel FPGA IP Design Example Clocking Scheme

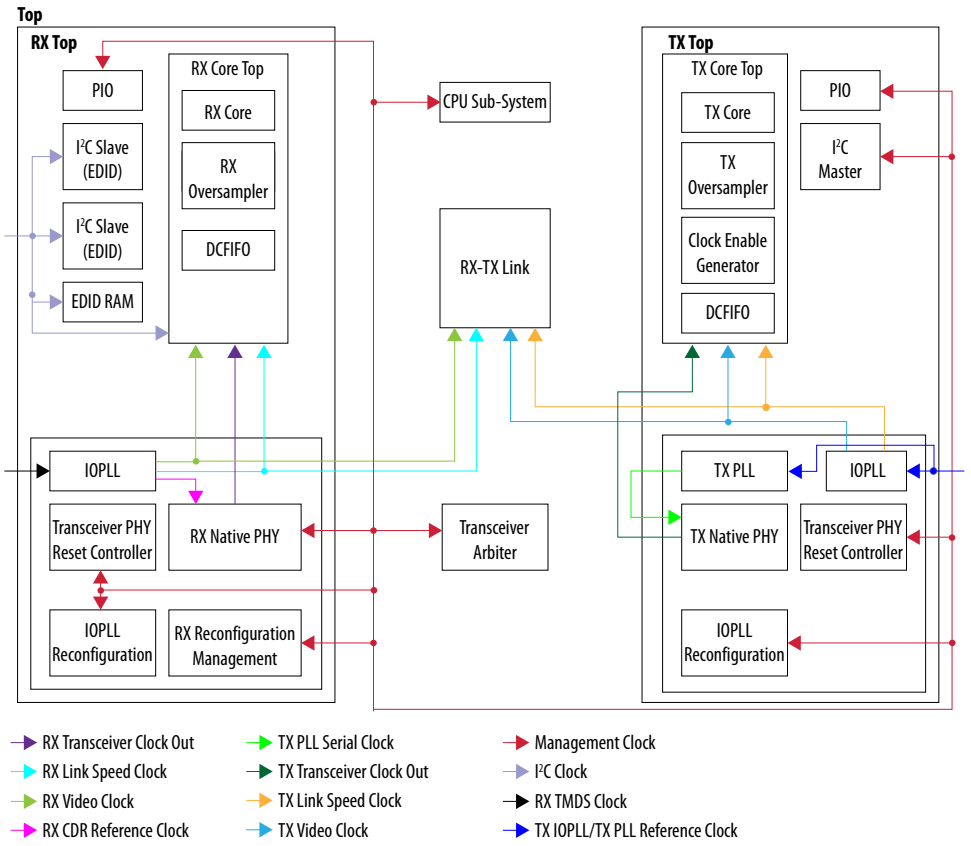


Table 37. Clocking Scheme Signals

Clock	Signal Name in Design	Description
TX IOPLL/ TX PLL Reference Clock	`hdmi_clk_in`	Reference clock to the TX IOPLL and TX PLL. The clock frequency is the same as the expected TMDS clock frequency from the HDMI TX TMDS clock channel. For this HDMI Intel FPGA IP design example, this clock is connected to the RX TMDS clock for demonstration purpose. In your application, you need to supply a dedicated clock with TMDS clock frequency from a programmable oscillator for better jitter performance. *Note:* Do not use a transceiver RX pin as a TX PLL reference clock. Your design will fail to fit if you place the HDMI TX refclk on an RX pin.
TX Transceiver Clock Out	`tx_clk`	Clock out recovered from the transceiver, and the frequency varies depending on the data rate and symbols per clock. TX transceiver clock out frequency = Transceiver data rate/ (Symbol per clock*10)
TX PLL Serial Clock	`tx_bonding_clocks`	Serial fast clock generated by TX PLL. The clock frequency is set based on the data rate.
TX/RX Link Speed Clock	`ls_clk`	Link speed clock. The link speed clock frequency depends on the expected TMDS clock frequency, oversampling factor, symbols per clock, and TMDS bit clock ratio.

continued...

<table>
<tr><th>Clock</th><th>Signal Name in Design</th><th>Description</th></tr>
<tr><td></td><td></td><td>
<table>
<tr><th>TMDS Bit Clock Ratio</th><th>Link Speed Clock Frequency</th></tr>
<tr><td>0</td><td>TMDS clock frequency/ Symbol per clock</td></tr>
<tr><td>1</td><td>TMDS clock frequency *4 / Symbol per clock</td></tr>
</table>
</td></tr>
<tr><td>TX/RX Video Clock</td><td>vid_clk</td><td>Video data clock. The video data clock frequency is derived from the TX link speed clock based on the color depth.
<table>
<tr><th>TMDS Bit Clock Ratio</th><th>Video Data Clock Frequency</th></tr>
<tr><td>0</td><td>TMDS clock/ Symbol per clock/ Color depth factor</td></tr>
<tr><td>1</td><td>TMDS clock *4 / Symbol per clock/ Color depth factor</td></tr>
</table>
<table>
<tr><th>Bits per Color</th><th>Color Depth Factor</th></tr>
<tr><td>8</td><td>1</td></tr>
<tr><td>10</td><td>1.25</td></tr>
<tr><td>12</td><td>1.5</td></tr>
<tr><td>16</td><td>2.0</td></tr>
</table>
</td></tr>
<tr><td>RX TMDS Clock</td><td>tmds_clk_in</td><td>TMDS clock channel from the HDMI RX and connects to the reference clock to the IOPLL.</td></tr>
<tr><td>RX CDR Reference Clock</td><td>iopll_outclk0</td><td>Reference clock to the RX CDR of RX transceiver.
<table>
<tr><th>Data Rate</th><th>RX Reference Clock Frequency</th></tr>
<tr><td>Data rate <1 Gbps</td><td>5× TMDS clock frequency</td></tr>
<tr><td>1 Gbps< Data rate <3.4 Gbps</td><td>TMDS clock frequency</td></tr>
<tr><td>Data rate >3.4 Gbps</td><td>4× TMDS clock frequency</td></tr>
</table>

Data Rate <1 Gbps: For oversampling to meet transceiver minimum data rate requirement.
Data Rate >3.4 Gbps: To compensate for the TMDS bit rate to clock ratio of 1/40 to maintain the transceiver data rate to clock ratio at 1/10.

Note: Do not use a transceiver RX pin as a CDR reference clock. Your design will fail to fit if you place the HDMI RX refclk on an RX pin.</td></tr>
<tr><td>RX Transceiver Clock Out</td><td>rx_clk</td><td>Clock out recovered from the transceiver, and the frequency varies depending on the data rate and symbols per clock.
RX transceiver clock out frequency = Transceiver data rate/ (Symbol per clock*10)</td></tr>
<tr><td>Management Clock</td><td>mgmt_clk</td><td>A free running 100 MHz clock for these components:</td></tr>
</table>

continued...

Clock	Signal Name in Design	Description
		• Avalon-MM interfaces for reconfiguration — The frequency range requirement is between 100–125 MHz. • PHY reset controller for transceiver reset sequence — The frequency range requirement is between 1–500 MHz. • IOPLL Reconfiguration — The maximum clock frequency is 100 MHz. • RX Reconfiguration for management • CPU • I^2C Master
I^2C Clock	`i2c_clk`	A 50 MHz clock input that clocks I^2C slave, SCDC registers in the HDMI RX core, and EDID RAM.

Related Information

- Using Transceiver RX Pin as CDR Reference Clock
- Using Transceiver RX Pin as TX PLL Reference Clock

3.7. Interface Signals

The tables list the signals for the HDMI Intel FPGA IP design example.

Table 38. Top-Level Signals

Signal	Direction	Width	Description
On-board Oscillator Signal			
`clk_fpga_b3_p`	Input	1	100 MHz free running clock
`clk_50`	Input	1	50 MHz free running clock
User Push Buttons and LEDs			
`user_pb`	Input	1	Push button to control the HDMI Intel FPGA IP design functionality
`cpu_resetn`	Input	1	Global reset
`user_led_g`	Output	4	Green LED display Refer to Hardware Setup on page 83 for more information about the LED functions.
`user_led_r`	Output	4	Red LED display Refer to Hardware Setup on page 83 for more information about the LED functions.

HDMI FMC Daughter Card Pins on FMC Port B			
fmcb_gbtclk_m2c_p_0	Input	1	HDMI RX TMDS clock
fmcb_dp_m2c_p	Input	3	HDMI RX red, green, and blue data channels • Bitec daughter card revision 11 — [0]: RX TMDS Channel 1 (Green) — [1]: RX TMDS Channel 2 (Red) — [2]: RX TMDS Channel 0 (Blue) • Bitec daughter card revision 4 or 6 — [0]: RX TMDS Channel 1 (Green)—polarity inverted — [1]: RX TMDS Channel 0 (Blue)—polarity inverted — [2]: RX TMDS Channel 2 (Red)—polarity inverted
fmcb_dp_c2m_p	Output	4	HDMI TX clock, red, green, and blue data channels • Bitec daughter card revision 11 — [0]: TX TMDS Channel 2 (Red) — [1]: TX TMDS Channel 1 (Green) — [2]: TX TMDS Channel 0 (Blue) — [3]: TX TMDS Clock Channel • Bitec daughter card revision 4 or 6 — [0]: TX TMDS Clock Channel — [1]: TX TMDS Channel 0 (Blue) — [2]: TX TMDS Channel 1 (Green) — [3]: TX TMDS Channel 2 (Red)
fmcb_la_rx_p_9	Input	1	HDMI RX +5V power detect
fmcb_la_rx_p_8	Inout	1	HDMI RX hot plug detect
fmcb_la_rx_n_8	Inout	1	HDMI RX I^2C SDA
fmcb_la_tx_p_10	Input	1	HDMI RX I^2C SCL
fmcb_la_tx_p_12	Input	1	HDMI TX hot plug detect
fmcb_la_tx_n_12	Inout	1	HDMI I^2C SDA
fmcb_la_rx_p_10	Inout	1	HDMI I^2C SCL

Table 39. HDMI RX Top-Level Signals

Signal	Direction	Width	Description
Clock and Reset Signals			
mgmt_clk	Input	1	System clock input (100 MHz)
reset	Input	1	System reset input
tmds_clk_in	Input	1	HDMI RX TMDS clock
i2c_clk	Input	1	Clock input for DDC and SCDC interface
vid_clk_out	Output	1	Video clock output
ls_clk_out	Output	8	Link speed clock output
sys_init	Output	1	System initialization to reset the system upon power-up

RX Transceiver and IOPLL Signals			
rx_serial_data	Input	3	HDMI serial data to the RX Native PHY
gxb_rx_ready	Output	1	Indicates RX Native PHY is ready
gxb_rx_cal_busy_out	Output	3	RX Native PHY calibration busy to the transceiver arbiter
gxb_rx_cal_busy_in	Input	3	Calibration busy signal from the transceiver arbiter to the RX Native PHY
iopll_locked	Output	1	Indicate IOPLL is locked
gxb_reconfig_write	Input	3	Transceiver reconfiguration Avalon-MM interface from the RX Native PHY to the transceiver arbiter
gxb_reconfig_read	Input	3	
gxb_reconfig_address	Input	30	
gxb_reconfig_writedata	Input	96	
gxb_reconfig_readdata	Output	96	
gxb_reconfig_waitrequest	Output	3	

RX Reconfiguration Management			
rx_reconfig_en	Output	1	RX Reconfiguration enables signal
measure	Output	24	HDMI RX TMDS clock frequency measurement (in 10 ms)
measure_valid	Output	1	Indicates the measure signal is valid
os	Output	1	Oversampling factor: • 0: No oversampling • 1: 5× oversampling
reconfig_mgmt_write	Output	1	RX reconfiguration management Avalon-MM interface to transceiver arbiter
reconfig_mgmt_read	Output	1	
reconfig_mgmt_address	Output	12	
reconfig_mgmt_writedata	Output	32	
reconfig_mgmt_readdata	Input	32	
reconfig_mgmt_waitrequest	Input	1	

HDMI RX Core Signals			
TMDS_Bit_clock_Ratio	Output	1	SCDC register interfaces
audio_de	Output	1	HDMI RX core audio interfaces Refer to the *Sink Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
audio_data	Output	256	
audio_info_ai	Output	48	
audio_N	Output	20	
audio_CTS	Output	20	
audio_metadata	Output	165	
audio_format	Output	5	

continued...

HDMI RX Core Signals			
aux_pkt_data	Output	72	HDMI RX core auxiliary interfaces Refer to the *Sink Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
aux_pkt_addr	Output	6	
aux_pkt_wr	Output	1	
aux_data	Output	72	
aux_sop	Output	1	
aux_eop	Output	1	
aux_valid	Output	1	
aux_error	Output	1	
gcp	Output	6	HDMI RX core sideband signals Refer to the *Sink Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
info_avi	Output	112	
info_vsi	Output	61	
colordepth_mgmt_sync	Output	2	
vid_data	Output	*N**48	HDMI RX core video ports *Note: N* = symbols per clock Refer to the *Sink Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
vid_vsync	Output	*N*	
vid_hsync	Output	*N*	
vid_de	Output	*N*	
mode	Output	1	HDMI RX core control and status ports *Note: N* = symbols per clock Refer to the *Sink Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
ctrl	Output	*N**6	
locked	Output	3	
vid_lock	Output	1	
in_5v_power	Input	1	HDMI RX 5V detect and hotplug detect Refer to the *Sink Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
hdmi_rx_hpd_n	Inout	1	

I²C Signals			
hdmi_rx_i2c_sda	Inout	1	HDMI RX DDC and SCDC interface
hdmi_rx_i2c_scl	Inout	1	

RX EDID RAM Signals			
edid_ram_access	Input	1	HDMI RX EDID RAM access interface. Assert `edid_ram_access` when you want to write or read from the EDID RAM, else this signal should be kept low.
edid_ram_address	Input	8	
edid_ram_write	Input	1	
edid_ram_read	Input	1	
edid_ram_readdata	Output	8	
edid_ram_writedata	Input	8	
edid_ram_waitrequest	Output	1	

Table 40. HDMI TX Top-Level Signals

Signal	Direction	Width	Description
Clock and Reset Signals			
mgmt_clk	Input	1	System clock input (100 MHz)
reset	Input	1	System reset input
hdmi_clk_in	Input	1	Reference clock to TX IOPLL and TX PLL. The clock frequency is the same as the TMDS clock frequency.
vid_clk_out	Output	1	Video clock output
ls_clk_out	Output	8	Link speed clock output
sys_init	Output	1	System initialization to reset the system upon power-up
reset_xcvr	Input	1	Reset to TX transceiver
reset_pll	Input	1	Reset to IOPLL and TX PLL
reset_pll_reconfig	Output	1	Reset to PLL reconfiguration
TX Transceiver and IOPLL Signals			
tx_serial_data	Output	4	HDMI serial data from the TX Native PHY
gxb_tx_ready	Output	1	Indicates TX Native PHY is ready
gxb_tx_cal_busy_out	Output	4	TX Native PHY calibration busy signal to the transceiver arbiter
gxb_tx_cal_busy_in	Input	4	Calibration busy signal from the transceiver arbiter to the TX Native PHY
iopll_locked	Output	1	Indicate IOPLL is locked
txpll_locked	Output	1	Indicate TX PLL is locked
gxb_reconfig_write	Input	4	Transceiver reconfiguration Avalon-MM interface from the TX Native PHY to the transceiver arbiter
gxb_reconfig_read	Input	4	
gxb_reconfig_address	Input	40	
gxb_reconfig_writedata	Input	128	
gxb_reconfig_readdata	Output	128	
gxb_reconfig_waitrequest	Output	4	
TX IOPLL and TX PLL Reconfiguration Signals			
pll_reconfig_write/ tx_pll_reconfig_write	Input	1	TX IOPLL/TX PLL reconfiguration Avalon-MM interfaces
pll_reconfig_read/ tx_pll_reconfig_read	Input	1	
pll_reconfig_address/ tx_pll_reconfig_address	Input	10	
pll_reconfig_writedata/ tx_pll_reconfig_writedata	Input	32	

continued...

TX IOPLL and TX PLL Reconfiguration Signals			
pll_reconfig_readdata/ tx_pll_reconfig_readdata	Output	32	
pll_reconfig_waitrequest/ tx_pll_reconfig_waitrequest	Output	1	
os	Input	2	Oversampling factor: • 0: No oversampling • 1: 3× oversampling • 2: 4× oversampling • 3: 5× oversampling
measure	Input	24	Indicates the TMDS clock frequency of the transmitting video resolution.

HDMI TX Core Signals			
ctrl	Input	6*N	HDMI TX core control interfaces *Note: N* = Symbols per clock Refer to the *Source Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
mode	Input	1	
TMDS_Bit_clock_Ratio	Input	1	SCDC register interfaces Refer to the *Source Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
Scrambler_Enable	Input	1	
audio_de	Input	1	HDMI TX core audio interfaces Refer to the *Source Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
audio_mute	Input	1	
audio_data	Input	256	
audio_info_ai	Input	49	
audio_N	Input	22	
audio_CTS	Input	22	
audio_metadata	Input	166	
audio_format	Input	5	
aux_ready	Output	1	HDMI TX core auxiliary interfaces Refer to the *Source Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
aux_data	Input	72	
aux_sop	Input	1	
aux_eop	Input	1	
aux_valid	Input	1	
gcp	Input	6	HDMI TX core sideband signals Refer to the *Source Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.
info_avi	Input	113	
info_vsi	Input	62	
vid_data	Input	*N*48	HDMI TX core video ports *Note: N* = symbols per clock
vid_vsync	Input	*N*	
vid_hsync	Input	*N*	

continued...

HDMI TX Core Signals			
vid_de	Input	*N*	Refer to the *Source Interfaces* section in the *HDMI Intel FPGA IP User Guide* for more information.

I^2C and Hot Plug Detect Signals			
nios_tx_i2c_sda_in	Output	1	I^2C Master Avalon-MM interfaces
nios_tx_i2c_scl_in	Output	1	
nios_tx_i2c_sda_oe	Input	1	
nios_tx_i2c_scl_oe	Input	1	
nios_ti_i2c_sda_in	Output	1	
nios_ti_i2c_scl_in	Output	1	
nios_ti_i2c_sda_oe	Input	1	
nios_ti_i2c_scl_oe	Input	1	
hdmi_tx_i2c_sda	Inout	1	HDMI TX DDC and SCDC interfaces
hdmi_tx_i2c_scl	Inout	1	
hdmi_ti_i2c_sda	Inout	1	I^2C interface for Bitec Daughter Card Revision 11 TI181 Control
hdmi_ti_i2c_scl	Inout	1	
hdmi_tx_hpd_n	Input	1	HDMI TX hotplug detect interfaces
tx_hpd_ack	Input	1	
tx_hpd_req	Output	1	

Table 41. 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	

continued...

Signal	Direction	Width	Description
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	
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 42. RX-TX Link Signals

Signal	Direction	Width	Description
reset	Input	1	Reset to the video/audio/auxiliary/ sidebands FIFO buffer.
mgmt_clk	Input	1	100 MHz clock
i2c_clk	Input	1	I^2C clock
hdmi_tx_ls_clk	Input	1	HDMI TX link speed clock
hdmi_rx_ls_clk	Input	1	HDMI RX link speed clock
hdmi_tx_vid_clk	Input	1	HDMI TX video clock
hdmi_rx_vid_clk	Input	1	HDMI RX video clock
sys_init	Input	1	System initialization to reset the system upon power-up

continued...

Signal	Direction	Width	Description
wd_reset	Input	1	Watchdog timer reset
hdmi_rx_locked	Input	3	Indicates HDMI RX locked status
hdmi_rx_de	Input	*N*	HDMI RX video interfaces *Note: N* = symbols per clock
hdmi_rx_hsync	Input	*N*	
hdmi_rx_vsync	Input	*N*	
hdmi_rx_data	Input	*N*48*	
rx_audio_format	Input	5	HDMI RX audio interfaces
rx_audio_metadata	Input	165	
rx_audio_info_ai	Input	48	
rx_audio_CTS	Input	20	
rx_audio_N	Input	20	
rx_audio_de	Input	1	
rx_audio_data	Input	256	
rx_gcp	Input	6	HDMI RX sideband interfaces
rx_info_avi	Input	112	
rx_info_vsi	Input	61	
rx_aux_eop	Input	1	HDMI RX auxiliary interfaces
rx_aux_sop	Input	1	
rx_aux_valid	Input	1	
rx_aux_data	Input	72	
hdmi_tx_de	Output	*N*	HDMI TX video interfaces *Note: N* = symbols per clock
hdmi_tx_hsync	Output	*N*	
hdmi_tx_vsync	Output	*N*	
hdmi_tx_data	Output	*N*48*	
tx_audio_format	Output	5	HDMI TX audio interfaces
tx_audio_metadata	Output	165	
tx_audio_info_ai	Output	48	
tx_audio_CTS	Output	20	
tx_audio_N	Output	20	
tx_audio_de	Output	1	
tx_audio_data	Output	256	
tx_gcp	Output	6	HDMI TX sideband interfaces
tx_info_avi	Output	112	
tx_info_vsi	Output	61	

continued...

Signal	Direction	Width	Description
tx_aux_eop	Output	1	HDMI TX auxiliary interfaces
tx_aux_sop	Output	1	
tx_aux_valid	Output	1	
tx_aux_data	Output	72	
tx_aux_ready	Output	1	

Table 43. Platform Designer System Signals

Signal	Direction	Width	Description
cpu_clk	Input	1	CPU clock
cpu_clk_reset_n	Input	1	CPU reset
tmds_bit_clock_ratio_pio_external_connection_export	Input	1	TMDS bit clock ratio
measure_pio_external_connection_export	Input	24	Expected TMDS clock frequency
	Input	1	Indicates measure PIO is valid
measure_valid_pio_external_connection_export	Input	Output	
i2c_master_i2c_serial_sda_in	Input	1	I^2C Master interfaces
i2c_master_i2c_serial_scl_in	Input	1	
i2c_master_i2c_serial_sda_oe	Output	1	
i2c_master_i2c_serial_scl_oe	Output	1	
i2c_master_ti_i2c_serial_sda_in	Input	1	
i2c_master_ti_i2c_serial_scl_in	Input	1	
i2c_master_ti_i2c_serial_sda_oe	Output	1	
i2c_master_ti_i2c_serial_scl_oe	Output	1	
oc_i2c_master_ti_avalon_anti_slave_address	Output	3	I^2C Master Avalon-MM interfaces for Bitec daughter card revision 11, T1181 control
oc_i2c_master_ti_avalon_anti_slave_write	Output	1	
oc_i2c_master_ti_avalon_anti_slave_readdata	Input	32	
oc_i2c_master_ti_avalon_anti_slave_writedata	Output	32	
oc_i2c_master_ti_avalon_anti_slave_waitrequest	Input	1	
oc_i2c_master_ti_avalon_anti_slave_chipselect	Output	1	
edid_ram_access_pio_external_connection_export	Output	1	EDID RAM access interfaces. Assert edid_ram_access_pio_external_connection_export when you want to
edid_ram_slave_translator_address	Output	8	
edid_ram_slave_translator_write	Output	1	

continued...

Signal	Direction	Width	Description
edid_ram_slave_translator_read	Output	1	write to or read from the EDID RAM on the RX top. Connect EDID RAM access Avalon-MM slave in Platform Designer to the EDID RAM interface on the top-level RX modules.
edid_ram_slave_translator_readdata	Input	8	
edid_ram_slave_translator_writedata	Output	8	
edid_ram_slave_translator_waitrequest	Input	1	
tx_pll_rcfg_mgmt_translator_avalon_anti_slave_waitrequest	Input	1	TX PLL Reconfiguration Avalon-MM interfaces
tx_pll_rcfg_mgmt_translator_avalon_anti_slave_writedata	Output	32	
tx_pll_rcfg_mgmt_translator_avalon_anti_slave_address	Output	10	
tx_pll_rcfg_mgmt_translator_avalon_anti_slave_write	Output	1	
tx_pll_rcfg_mgmt_translator_avalon_anti_slave_read	Output	1	
tx_pll_rcfg_mgmt_translator_avalon_anti_slave_readdata	Input	32	
tx_pll_waitrequest_pio_external_connection_export	Input	1	TX PLL waitrequest
tx_pma_rcfg_mgmt_translator_avalon_anti_slave_address	Output	12	TX PMA Reconfiguration Avalon-MM interfaces
tx_pma_rcfg_mgmt_translator_avalon_anti_slave_write	Output	1	
tx_pma_rcfg_mgmt_translator_avalon_anti_slave_read	Output	1	
tx_pma_rcfg_mgmt_translator_avalon_anti_slave_readdata	Input	32	
tx_pma_rcfg_mgmt_translator_avalon_anti_slave_writedata	Output	32	
tx_pma_rcfg_mgmt_translator_avalon_anti_slave_waitrequest	Input	1	
tx_pma_waitrequest_pio_external_connection_export	Input	1	TX PMA waitrequest
tx_pma_cal_busy_pio_external_connection_export	Input	1	TX PMA Recalibration Busy
tx_pma_ch_export	Output	2	TX PMA Channels
tx_rcfg_en_pio_external_connection_export			TX PMA Reconfiguration Enable
tx_iopll_rcfg_mgmt_translator_avalon_anti_slave_writedata	Output	32	TX IOPLL Reconfiguration Avalon-MM interfaces
tx_iopll_rcfg_mgmt_translator_avalon_anti_slave_address	Output	9	
tx_iopll_rcfg_mgmt_translator_avalon_anti_slave_write	Output	1	

continued...

Signal	Direction	Width	Description
tx_iopll_rcfg_mgmt_translator_avalon_anti_slave_read	Output	1	
tx_iopll_rcfg_mgmt_translator_avalon_anti_slave_readdata	Input	32	
tx_os_pio_external_connection_export	Output	2	Oversampling factor: • 0: No oversampling • 1: 3× oversampling • 2: 4× oversampling • 3: 5× oversampling
tx_rst_pll_pio_external_connection_export	Output	1	Reset to IOPLL and TX PLL
tx_rst_xcvr_pio_external_connection_export	Output	1	Reset to TX Native PHY
wd_timer_resetrequest_reset	Output	1	Watchdog timer reset
color_depth_pio_external_connection_export	Input	2	Color depth
tx_hpd_ack_pio_external_connection_export	Output	1	For TX hotplug detect handshaking
tx_hpd_req_pio_external_connection_export	Input	1	

3.8. Design RTL Parameters

Use the HDMI TX and RX Top RTL parameters to customize the design example.

Most of the design parameters are available in the **Design Example** tab of the HDMI Intel FPGA IP parameter editor. You can still change the design example settings you made in the parameter editor through the RTL parameters.

Table 44. HDMI RX Top Parameters

Parameter	Value	Description
SUPPORT_DEEP_COLOR	• 0: No deep color • 1: Deep color	Determines if the core can encode deep color formats. *Note:* This feature is not supported in FRL mode.
SUPPORT_AUXILIARY	• 0: No AUX • 1: AUX	Determines if the auxiliary channel encoding is included.
SYMBOLS_PER_CLOCK	8	Supports 8 symbols per clock for Intel Arria 10 devices.
SUPPORT_AUDIO	• 0: No audio • 1: Audio	Determines if the core can encode audio.

Table 45. HDMI TX Top Parameters

Parameter	Value	Description
SUPPORT_DEEP_COLOR	• 0: No deep color • 1: Deep color	Determines if the core can encode deep color formats.

continued...

Parameter	Value	Description
		Note: This feature is not supported in FRL mode.
SUPPORT_AUXILIARY	• 0: No AUX • 1: AUX	Determines if the auxiliary channel encoding is included.
SYMBOLS_PER_CLOCK	8	Supports 8 symbols per clock for Intel Arria 10 devices.
SUPPORT_AUDIO	• 0: No audio • 1: Audio	Determines if the core can encode audio.

3.9. Hardware Setup

The HDMI Intel FPGA IP design example is HDMI 2.0b capable and performs a loop-through demonstration for a standard HDMI video stream.

To run the hardware test, connect an HDMI-enabled device—such as a graphics card with HDMI interface—to the Transceiver Native PHY RX block, and the HDMI sink input.

1. The HDMI sink decodes the port into a standard video stream and sends it to the clock recovery core.
2. The HDMI RX core decodes the video, auxiliary, and audio data to be looped back in parallel to the HDMI TX core through the DCFIFO.
3. The HDMI source port of the FMC daughter card transmits the image to a monitor.

Note: If you want to use another Intel FPGA development board, you must change the device assignments and the pin assignments. The transceiver analog setting is tested for the Intel Arria 10 FPGA development kit and Bitec HDMI 2.0 daughter card. You may modify the settings for your own board.

Table 46. On-board Push Button and User LED Functions

Push Button/LED	Function
cpu_resetn	Press once to perform system reset.
user_pb[0]	Press once to toggle the HPD signal to the standard HDMI source.
user_pb[1]	• Press and hold to instruct the TX core to send the DVI encoded signal. • Release to send the HDMI encoded signal.
user_pb[2]	• Press and hold to instruct the TX core to stop sending the InfoFrames from the sideband signals. • Release to resume sending the InfoFrames from the sideband signals.
USER_LED[0]	RX HDMI PLL lock status. • 0 = Unlocked • 1 = Locked
USER_LED[1]	RX transceiver ready status. • 0 = Not ready • 1 = Ready
USER_LED[2]	RX HDMI core lock status. • 0 = At least 1 channel unlocked • 1 = All 3 channels locked

continued...

Push Button/LED	Function
USER_LED[3]	RX oversampling status. • 0 = Non-oversampled (data rate > 1,000 Mbps in Intel Arria 10 device) • 1 = Oversampled (data rate < 100 Mbps in Intel Arria 10 device)
USER_LED[4]	TX HDMI PLL lock status. • 0 = Unlocked • 1 = Locked
USER_LED[5]	TX transceiver ready status. • 0 = Not ready • 1 = Ready
USER_LED[6]	TX transceiver PLL lock status. • 0 = Unlocked • 1 = Locked
USER_LED[7]	TX oversampling status. • 0 = Non-oversampled (data rate > 1,000 Mbps in Intel Arria 10 device) • 1 = Oversampled (data rate < 1,000 Mbps in Intel Arria 10 device)

3.11. Simulation Testbench

The simulation testbench simulates the HDMI TX serial loopback to the RX core.

Figure 25. HDMI Intel FPGA IP Simulation Testbench Block Diagram

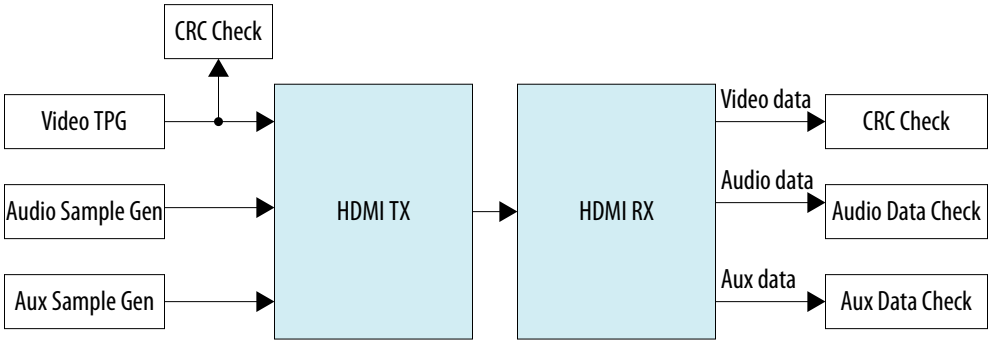


Table 47. Testbench Components

Component	Description
Video TPG	The video test pattern generator (TPG) provides the video stimulus.
Audio Sample Gen	The audio sample generator provides audio sample stimulus. The generator generates an incrementing test data pattern to be transmitted through the audio channel.
Aux Sample Gen	The aux sample generator provides the auxiliary sample stimulus. The generator generates a fixed data to be transmitted from the transmitter.
CRC Check	This checker verifies if the TX transceiver recovered clock frequency matches the desired data rate.
Audio Data Check	The audio data check compares whether the incrementing test data pattern is received and decoded correctly.
Aux Data Check	The aux data check compares whether the expected aux data is received and decoded correctly on the receiver side.

The HDMI simulation testbench does the following verification tests:

HDMI Feature	Verification
Video data	• The testbench implements CRC checking on the input and output video. • It checks the CRC value of the transmitted data against the CRC calculated in the received video data. • The testbench then performs the checking after detecting 4 stable V-SYNC signals from the receiver.
Auxiliary data	• The aux sample generator generates a fixed data to be transmitted from the transmitter. • On the receiver side, the generator compares whether the expected auxiliary data is received and decoded correctly.
Audio data	• The audio sample generator generates an incrementing test data pattern to be transmitted through the audio channel. • On the receiver side, the audio data checker checks and compares whether the incrementing test data pattern is received and decoded correctly.

A successful simulation ends with the following message:

```
# SYMBOLS_PER_CLOCK = 2
# VIC               = 0
# AUDIO_CLK_DIVIDE  = 800
# TEST_HDMI_6G      = 1
# Simulation pass
```

Table 48. HDMI Intel FPGA IP Design Example Supported Simulators

Simulator	Verilog HDL	VHDL
ModelSim - Intel FPGA Edition/ ModelSim - Intel FPGA Starter Edition	Yes	Yes
VCS/VCS MX	Yes	Yes
Riviera-PRO	Yes	Yes
NCSim	Yes	No
Xcelium Parallel	Yes	No

3.12. Upgrading Your Design

When you upgrade your designs to a later version, you may have to add, remove, or edit some of the generated files.

Upgrading from Version 18.0 Update 1 to 18.1

1. Generate a new design example in version 18.1 using the same configurations of your existing design.
2. Compare the whole design example directory; replace the files that have changes with the new files and copy over the new files to your existing design.
3. Click **IP Upgrade** to upgrade all the IP and Platform Designer files.
4. Move the following assignments to the bottom of the QSF file.

```
set_global_assignment -name SDC_ENTITY_FILE ../rtl/sdc/mr_clock_sync.sdc -
entity mr_clock_sync
set_global_assignment -name SDC_ENTITY_FILE ../rtl/sdc/mr_reconfig_mgmt.sdc
-entity mr_reconfig_mgmt
set_global_assignment -name SDC_ENTITY_FILE ../rtl/sdc/rxtx_link.sdc -
entity rxtx_link
```

5. Recompile the design.

Upgrading from Version 18.0 to 18.1

1. Generate a new design example in version 18.1 using the same configurations of your existing design.
2. Compare the whole design example directory; replace the files that have changes with the new files and copy over the new files to your existing design.
3. In the `/quartus/a10_hdmi2_demo.qsf` file:
   - Add the following new file:

```
set_global_assignment -name IP_FILE ../rtl/ip/nios/
nios_oc_i2c_master_ti.ip
```

   - Add new pin assignments for `fmcb_la_tx_p_11` and `fmcb_la_rx_n_9` pins:

```
set_location_assignment PIN_H13 -to fmcb_la_tx_p_11
set_location_assignment PIN_H18 -to fmcb_la_rx_n_9
```

```
set_instance_assignment -name IO_STANDARD "1.8 V" -to fmcb_la_tx_p_11
set_instance_assignment -name IO_STANDARD "1.8 V"-to fmcb_la_rx_n_9
```

```
set_instance_assignment -name AUTO_OPEN_DRAIN_PINS ON -to
fmcb_la_tx_p_11
set_instance_assignment -name WEAK_PULL_UP_RESISTOR ON -to
fmcb_la_tx_p_11
set_instance_assignment -name AUTO_OPEN_DRAIN_PINS ON -to fmcb_la_rx_n_9
set_instance_assignment -name WEAK_PULL_UP_RESISTOR ON -to
fmcb_la_rx_n_9
```

```
set_instance_assignment -name CURRENT_STRENGTH_NEW DEFAULT -to
fmcb_la_tx_p_11
set_instance_assignment -name CURRENT_STRENGTH_NEW DEFAULT -to
fmcb_la_rx_n_9
```

```
set_instance_assignment -name SLEW_RATE 1 -to fmcb_la_tx_p_11
set_instance_assignment -name SLEW_RATE 1 -to fmcb_la_rx_n_9
```

4. Click **IP Upgrade** to upgrade all the IP and Platform Designer files.
5. Move the following assignments to the bottom of the QSF file.

```
set_global_assignment -name SDC_ENTITY_FILE ../rtl/sdc/mr_clock_sync.sdc -
entity mr_clock_sync
set_global_assignment -name SDC_ENTITY_FILE ../rtl/sdc/mr_reconfig_mgmt.sdc
-entity mr_reconfig_mgmt
set_global_assignment -name SDC_ENTITY_FILE ../rtl/sdc/rxtx_link.sdc -
entity rxtx_link
```

6. Copy over the `software/tx_control_src` folder and locate the script directory. Run `sh script/build_sw.sh` script to rebuild the software.
7. Recompile the design.

Upgrading from Version 17.1 Update 1 to 18.1

1. Generate a new design example in version 18.1 using the same configurations of your existing design.
2. Compare the whole design example directory; replace the files that have changes with the new files and copy over the new files to your existing design.
3. In the `/quartus/a10_hdmi2_demo.qsf` file:

- Locate and place the `clock_crosser.v` file into `/rtl/common` folder.

```
set_global_assignment -name VERILOG_FILE ../rtl/common/clock_crosser.v
```

- Add the following new files:

```
set_global_assignment -name VERILOG_FILE ../rtl/common/dcfifo_inst.v
```

```
set_global_assignment -name IP_FILE ../rtl/common/fifo.ip
```

```
set_global_assignment -name IP_FILE ../rtl/ip/nios/
nios_oc_i2c_master_ti.ip
```

- Rename the `a10_reconfig_arbiter.sv` to `xcvr_reconfig_arbiter.sv`.

```
set_global_assignment -name SYSTEMVERILOG_FILE ../rtl/
xcvr_reconfig_arbiter.sv
```

- Remove `mr.sdc` and add the `mr_clock_sync.sdc`, `mr_reconfig_mgmt.sdc`, and `rxtx_link.sdc` files at the bottom of the QSF file.

```
set_global_assignment -name SDC_ENTITY_FILE ../rtl/sdc/
mr_clock_sync.sdc -entity mr_clock_sync
set_global_assignment -name SDC_ENTITY_FILE ../rtl/sdc/
mr_reconfig_mgmt.sdc -entity mr_reconfig_mgmt
set_global_assignment -name SDC_ENTITY_FILE ../rtl/sdc/rxtx_link.sdc -
entity rxtx_link
```

- Add new pin assignments for `fmcb_la_tx_p_11` and `fmcb_la_rx_n_9` pins:

```
set_location_assignment PIN_H13 -to fmcb_la_tx_p_11
set_location_assignment PIN_H18 -to fmcb_la_rx_n_9
```

```
set_instance_assignment -name IO_STANDARD "1.8 V" -to fmcb_la_tx_p_11
set_instance_assignment -name IO_STANDARD "1.8 V"-to fmcb_la_rx_n_9
```

```
set_instance_assignment -name AUTO_OPEN_DRAIN_PINS ON -to
fmcb_la_tx_p_11
set_instance_assignment -name WEAK_PULL_UP_RESISTOR ON -to
fmcb_la_tx_p_11
set_instance_assignment -name AUTO_OPEN_DRAIN_PINS ON -to fmcb_la_rx_n_9
set_instance_assignment -name WEAK_PULL_UP_RESISTOR ON -to
fmcb_la_rx_n_9
```

```
set_instance_assignment -name CURRENT_STRENGTH_NEW DEFAULT -to
fmcb_la_tx_p_11
set_instance_assignment -name CURRENT_STRENGTH_NEW DEFAULT -to
fmcb_la_rx_n_9
```

```
set_instance_assignment -name SLEW_RATE 1 -to fmcb_la_tx_p_11
set_instance_assignment -name SLEW_RATE 1 -to fmcb_la_rx_n_9
```

4. Click **IP Upgrade** to upgrade all the IP and Platform Designer files.
5. Move the following assignments to the bottom of the QSF file.

```
set_global_assignment -name SDC_ENTITY_FILE ../rtl/sdc/mr_clock_sync.sdc -
entity mr_clock_sync
set_global_assignment -name SDC_ENTITY_FILE ../rtl/sdc/mr_reconfig_mgmt.sdc
-entity mr_reconfig_mgmt
set_global_assignment -name SDC_ENTITY_FILE ../rtl/sdc/rxtx_link.sdc -
entity rxtx_link
```

6. Copy over the `software/tx_control_src` folder and locate the script directory. Run `sh script/build_sw.sh` script to rebuild the software.
7. Recompile the design.

4. HDMI Intel Arria 10 FPGA IP Design Example User Guide Archives

IP versions are the same as the Intel Quartus Prime Design Suite software versions up to 19.1. From Intel Quartus Prime Design Suite software version 19.2 or later, IP cores have a new IP versioning scheme.

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	HDMI Intel Arria 10 FPGA IP Design Example User Guide
17.1	Intel HDMI IP Design Example User Guide for Intel Arria 10 Devices
17.0	Intel Arria 10 HDMI IP Core Design Example User Guide
16.1	HDMI IP Core Design Example User Guide

5. Revision History for HDMI Intel Arria 10 FPGA IP Design Example User Guide

Document Version	Intel Quartus Prime Version	IP Version	Changes
2020.01.16	19.4	19.3.0	• Updated the *HDMI Intel FPGA IP Design Example Quick Start Guide for Intel Arria 10 Devices* section with information about the newly added HDMI 2.1 design example with FRL mode. • Added a new chapter, *Detailed Description for HDMI 2.1 Design Example (Support FRL Enabled)* that contains all the relevant information about the newly added design example. • Renamed the *HDMI Intel FPGA IP Design Example Detailed Description* to *Detailed Description for HDMI 2.0 Design Example* for better clarity.
2019.10.31	18.1	18.1	• Added generated files in the `tx_control_src` folder: `ti_i2c.c` and `ti_i2c.h`. • Added support for FMC daughter card revision 11 in the *Hardware and Software Requirements* and *Compiling and Testing the Design* sections. • Removed the *Design Limitation* section. The limitation regarding the timing violation on the maximum skew constraints was resolved in version 18.1 of the HDMI Intel FPGA IP. • Added a new RTL parameter, `BITEC_DAUGHTER_CARD_REV`, to enable you to select the revision of the Bitec HDMI daughter card. • Updated the description for `fmcb_dp_m2c_p` and `fmcb_dp_c2m_p` signals to include information about the FMC daughter card revisions 11, 6, and 4. • Added the following new signals for Bitec daughter card revision 11: — `hdmi_tx_ti_i2c_sda` — `hdmi_tx_ti_i2c_scl` — `oc_i2c_master_ti_avalon_anti_slave_address` — `oc_i2c_master_ti_avalon_anti_slave_write` — `oc_i2c_master_ti_avalon_anti_slave_readdata` — `oc_i2c_master_ti_avalon_anti_slave_writedata` — `oc_i2c_master_ti_avalon_anti_slave_waitrequest` • Added a section about *Upgrading Your Design*.

continued...

Document Version	Intel Quartus Prime Version	IP Version	Changes
2017.11.06	17.1	17.1	• Renamed HDMI IP core to HDMI Intel FPGA IP as per Intel rebranding. • Changed the term Qsys to Platform Designer. • Added information about Dynamic Range and Mastering InfoFrame (HDR) insertion and filtering feature. • Updated the directory structure: — Added script and software folders and files. — Updated common and hdr files. — Removed atx files. — Differentiated files for Intel Quartus Prime Standard Edition and Intel Quartus Prime Pro Edition. • Updated the *Generating the Design* section to add the device used as 10AX115S2F4I1SG. • Edited the transceiver data rate for 50-100 MHz TMDS clock frequency to 2550-5000 Mbps. • Updated the RX-TX link information that you can release the `user_pb[2]` button to disable external filtering. • Updated the Nios II software flow diagram that involves the controls for I^2C master and HDMI source. • Added information about the **Design Example** GUI parameters. • Added HDMI RX and TX Top design parameters. • Added these HDMI RX and TX top-level signals: — `mgmt_clk` — `reset` — `i2c_clk` — `hdmi_clk_in` — Removed these HDMI RX and TX top-level signals: • `version` • `i2c_clk` • Added a note that the transceiver analog setting is tested for the Intel Arria 10 FPGA Development Kit and Bitec HDMI 2.0 Daughter card. You may modify the analog setting for your board. • Added a link for workaround to avoid jitter of PLL cascading or non-dedicated clock paths for Intel Arria 10 PLL reference clock. • Added a note that you cannot use a transceiver RX pin as a CDR refclk for HDMI RX or as a TX PLL refclk for HDMI TX. • Added a note about how to add `set_max_skew` constraint for designs that use TX PMA and PCS bonding.
2017.05.08	17.0	17.0	• Rebranded as Intel. • Changed part number. • Updated the directory structure: — Added hdr files. — Changed `qsys_vip_passthrough.qsys` to `nios.qsys`. — Added files designated for Intel Quartus Prime Pro Edition.

continued...

Document Version	Intel Quartus Prime Version	IP Version	Changes
			• Updated information that the RX-TX Link block also performs external filtering on the High Dynamic Range (HDR) Infoframe from the HDMI RX auxiliary data and inserts an example HDR Infoframe to the auxiliary data of the HDMI TX through Avalon ST multiplexer. • Added a note for the Transceiver Native PHY description that to meet the HDMI TX inter-channel skew requirement, you need to set the TX channel bonding mode option in the Arria 10 Transceiver Native PHY parameter editor to **PMA and PCS bonding**. • Updated description for `os` and `measure` signals. • Modified the oversampling factor for different transceiver data rate at each TMDS clock frequency range to support TX FPLL direct clock scheme. • Changed TX IOPLL to TX FPLL cascade clocking scheme to TX FPLL direct scheme. • Added TX PMA reconfiguration signals. • Edited USER_LED[7] oversampling status. 1 indicates oversampled (data rate < 1,000 Mbps in Arria 10 device). • Updated *HDMI Design Example Supported Simulators* table. VHDL not supported for NCSim. • Added link to archived version of the *Arria 10 HDMI IP Core Design Example User Guide*.
2016.10.31	16.1	16.1	Initial release.