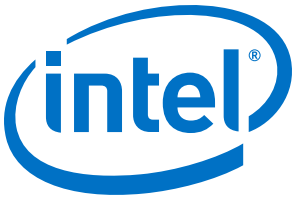


JESD204B Intel Stratix 10 FPGA IP Design Example User Guide

Updated for Intel® Quartus® Prime Design Suite: **18.0**


Subscribe


Send Feedback

UG-DEX-S10-JESD204B | 2018.08.10
Latest document on the web: **PDF** | **HTML**

Contents

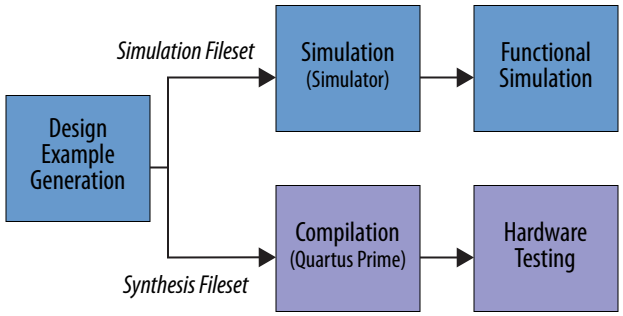
1. JESD204B Intel® Stratix® 10 FPGA IP Design Example User Guide

Intel provides a design example of the JESD204B Intel® FPGA IP targeting Intel Stratix® 10 devices. Generate the JESD204B design example through the IP catalog in the Intel Quartus® Prime Pro Edition software.

1.1. JESD204B Intel Stratix 10 FPGA IP Design Example Quick Start Guide

The JESD204B IP core provides the capability of generating design examples for selected configurations.

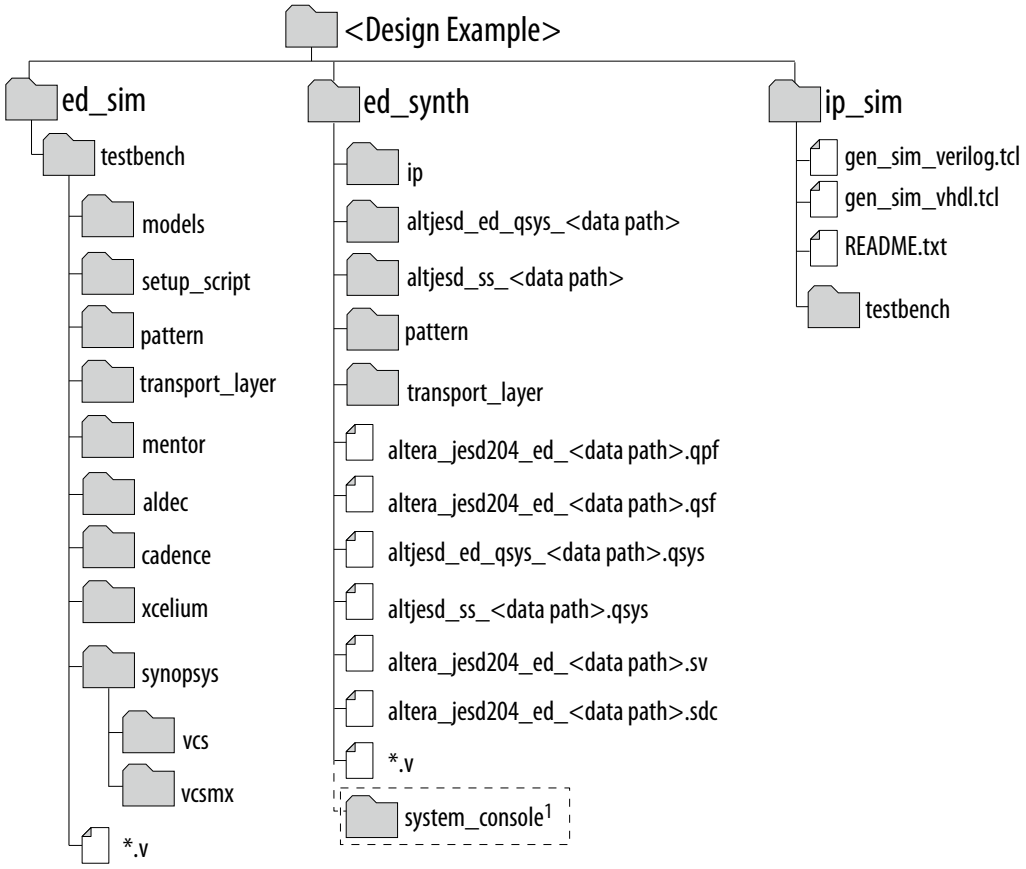
Figure 1. Development Stages for the Design Example



1.1.1. Directory Structure

The JESD204B design example directories contain generated files for the design examples.

Figure 2. Directory Structure for the JESD204B Design Example



Note:

1. *Directory 'system_console' only generated when 'Data Path Only' design example is generated.*

Table 1. Directory and File Description

Directory/File	Description
`ed_sim`	The folder that contains simulation testbench files
`ed_sim/testbench/models`	The folder that contains the testbench and source files
`ed_sim/testbench/setup_scripts`	The folder that contains the test flow setup scripts
`ed_sim/testbench/pattern`	The folder that contains the source files for the pattern generator/checker
`ed_sim/testbench/transport_layer`	The folder that contains the source files for the transport layer
`ed_sim/testbench/aldec`	The folder that contains the test flow run scripts for Riviera-PRO* simulator. Also serves as the working directory for the simulator.
`ed_sim/testbench/cadence`	The folder that contains the test flow run scripts for NCSim simulator. Also serves as the working directory for the simulator.
`ed_sim/testbench/xcelium`	The folder that contains the test flow run scripts for Xcelium* Parallel simulator. Also serves as the working directory for the simulator.

continued...

Directory/File	Description
ed_sim/testbench/mentor	The folder that contains the test flow run scripts for ModelSim* simulator. Also serves as the working directory for the simulator.
ed_sim/testbench/synopsys/vcs	The folder that contains the test flow run scripts for VCS* simulator. Also serves as the working directory for the simulator.
ed_sim/testbench/synopsys/vcsmx	The folder that contains the test flow run scripts for VCS MX simulator. Also serves as the working directory for the simulator.
ed_synth	The folder that contains design example synthesizable components
ed_synth/ip	The folder that contains Platform Designer-instantiated IP modules
ed_synth/altjesd_ed_qsys_<data path>	The folder that contains Platform Designer-generated modules from the altjesd_ed_qsys_<data path>.qsys system
ed_synth/altjesd_ss_<data path>	The folder that contains Platform Designer-generated modules from the altjesd_ss_<data path>.qsys system
ed_synth/pattern	The folder that contains the source files for the pattern generator/checker
ed_synth/transport_layer	The folder that contains the source files for the transport layer
ed_synth/altera_jesd204_ed_<data path>.qpf ed_synth/altera_jesd204_ed_<data path>.qsf	Intel Quartus Prime project and settings files
ed_synth/altjesd_ed_qsys_<data path>.qsys	Platform Designer top level system
ed_synth/altjesd_ss_<data path>.qsys	Platform Designer subsystem
ed_synth/altera_jesd204_ed_<data path>.sv	Top level HDL source file
ed_synth/altera_jesd204_ed_<data path>.sdc	Top level design constraints file
ed_synth/system_console	The folder that contains all files necessary to run scripts in System Console (See Design Example Files for more details on folder content.)
*.v	Miscellaneous source files
ip_sim	The folder that contains the simulation script to generate the JESD204B IP core Verilog/VHDL simulation model.

1.1.2. Generating the Design

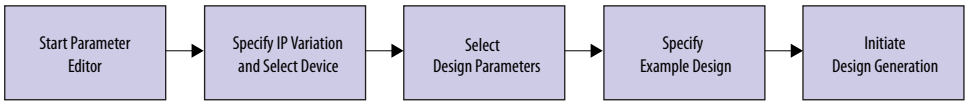
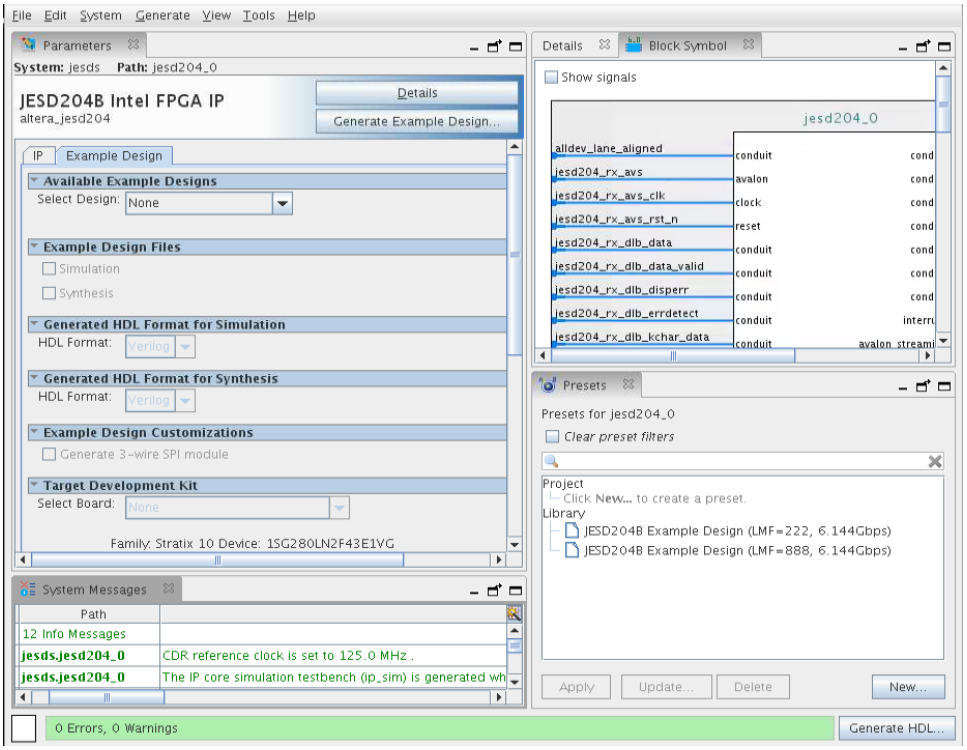


Figure 3. Example Design Tab



To generate the design example from the IP parameter editor:

1. Create a project targeting device family and select the desired device.
2. In the IP Catalog, locate and double-click **Interface Protocols ➤ JESD ➤ JESD204B Intel FPGA IP**. The IP parameter editor appears.
3. Specify a top-level name and the folder for your custom IP variation.. Click **OK**.
4. Select a design from the **Presets** library by double-clicking the desired preset. When you select a design, the system automatically populates the IP parameters for the design.

   Note: If you select another design, the settings of the IP parameters change accordingly.
5. You can customize the preset parameter values according to your specifications. Under the **IP** tab, specify the JESD204B IP core parameters for your design.

   Note: The JESD204B IP core supports a limited range of parameter combinations. Refer to the Supported Configurations on page 13 section for more details. If you specify an unsupported combination of parameters, the **Available Example Designs** automatically selects **None** as the default.
6. Under the **Example Design** tab, specify the design example parameters as described in *Design Example Parameters*.

   Note: To generate the design example for hardware testing on selected Intel development kits, select the appropriate target development kit from the **Target Development Kit** drop down box.
7. Click **Generate Example Design**.

The software generates all design files in the sub-directories. These files are required to run simulation, compilation, and hardware testing.

Related Information

- Presets on page 15
- Supported Configurations on page 13

1.1.2.1. Design Example Parameters

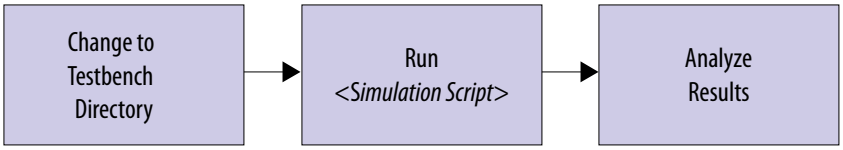
The JESD204B IP parameter editor includes a **Example Design** tab for you to specify certain parameters before generating the design example.

Table 2. Parameters in the Example Design Tab

Parameter	Options	Description
Available Example Designs	None (Default)	No design examples selected.
	System Console Control	Design example with System Console control.
Example Design Files	Simulation	Generate simulation fileset.
	Synthesis	Generate synthesis fileset.
Generated HDL Format for Simulation	Verilog (Default)	Verilog HDL format for entire simulation fileset.
	VHDL	VHDL format for Platform Designer generated top-level wrapper file set.
Generated HDL Format for Synthesis	Verilog (Default)	Verilog HDL format for synthesis fileset.
Example Design Customizations	Generate 3-wire SPI module	Check to enable 3-wire SPI interface instead of 4-wire SPI interface.

1.1.3. Simulating the Design

These general steps describe how to run the design example simulation. For specific commands for each design example variant, refer to its respective section.



To simulate the design, perform the following steps:

1. Change the working directory to `<example_design_directory>`**/ed_sim/testbench**/`<Simulator>`.
2. In the command line, run the simulation script. The table below shows the commands to run the supported simulators.

Simulator	Command
Riviera-PRO	`do run_tb_top.tcl`
NCSim	`sh run_tb_top.sh`

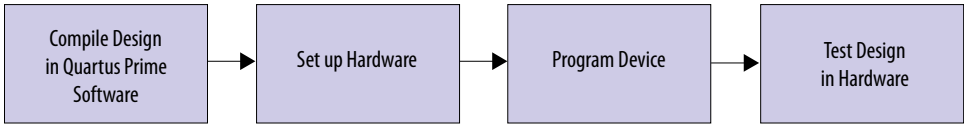
continued...

Simulator	Command
ModelSim	`do run_tb_top.tcl`
VCS/VCS MX	`sh run_tb_top.sh`
Xcelium Parallel	`sh run_tb_top.sh`

The simulation ends with messages that indicate whether the run was successful or not. Refer to *Simulation Message and Description* table in Testbench on page 27 for more information on messages reported by the simulation flow.

1.1.4. Compiling and Testing the Design

The JESD204B parameter editor allows you to run the design example on a target development kit.



Perform the following steps to compile the design and program the development board:

1. Launch the Intel Quartus Prime software and compile the design (**Processing ➤ Start Compilation**).

   The timing constraints and pin assignments for the design example and the design components are automatically loaded during design example compilation.
2. Connect the development board to the host computer either by connecting a USB cable to the on-board Intel FPGA Download Cable II component or using an external Intel FPGA Download Cable II module to connect to the external JTAG connector.
3. Launch the **Clock Control** application that is included with the development board, and set the clock settings according to the selected data rate.

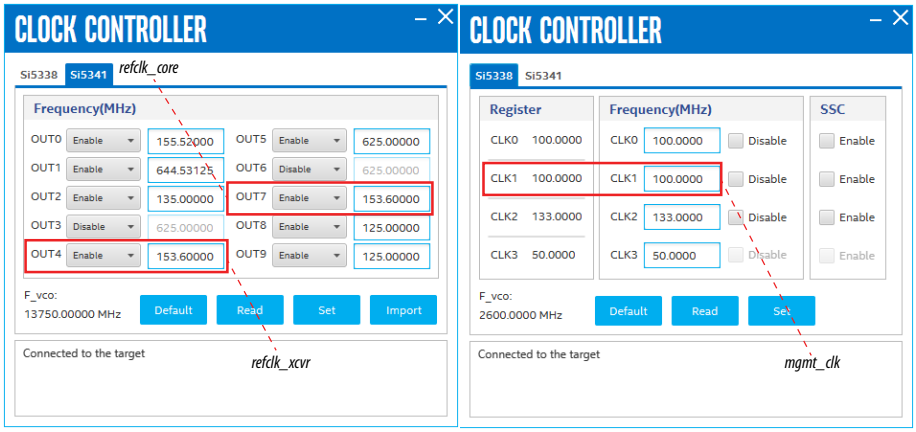
   Note: Refer to the Intel Stratix 10 GX FPGA Development Kit documentation for more information on using the **Clock Control** application.

Table 3. Clock Setting

Clock Name	Clock Frequency
`refclk_xcvr`	Select the frequencies in the **PLL/CDR Reference Clock Frequency** drop down menu of the IP parameter editor.
`refclk_core`	
`mgmt_clk`	100 MHz

Figure 4. Clock Control GUI Setting

This example shows the clock control GUI setting for 6.144 Gbps data rate.



4. If you are performing external FMC loopback test, affix the FMC loopback card to the FMC port A connector.
5. Configure the FPGA on the development board with the generated programming file (`.sof` file) using the Intel Quartus Prime **Programmer**.

Related Information

- Intel Stratix 10 GX FPGA Development Kit User Guide
- JESD204B Intel FPGA IP User Guide
- Intel FPGA JESD204B RX Address Map and Register Definitions
- Intel FPGA JESD204B TX Address Map and Register Definitions

1.1.4.1. Board Connectivity

If you are performing hardware testing on the selected Intel development kits, generate the design example with the appropriate target development kit selected.

Refer to the instructions in Generating the Design on page 5.

Note: Running the hardware test with the design generated as-is is only possible when the JESD204B IP core is configured in duplex data path mode (i.e. with both TX and RX data paths present). Make your own modifications to the design to run the hardware test if generating a simplex data path design.

Table 4. Intel Stratix 10 FPGA Development Kit Board Connectivity

The generated design has pre-assigned pins that target the relevant boards. The table describes the board connectivity of key design ports for all supported target development kits.

Port Name	Port Description	Board Component	Component Description
`global_rst_n`	Global reset	S5	User PB0 push-button
`refclk_xcvr`	Transceiver reference clock input	U7	Si5341 clock generator (OUT4)
`refclk_core`	Core PLL reference clock input	U7	Si5341 clock generator (OUT7)

continued...

Port Name	Port Description	Board Component	Component Description
mgmt_clk	Control clock	U9	Si5338 clock generator (CLK1)
tx_serial_data	TX serial data	J13	FMC port A connector
rx_serial_data	RX serial data	J13	FMC port A connector

1.1.4.2. Hardware Test for System Console Control Design Example

Perform the following instructions to run the hardware test for the design example.

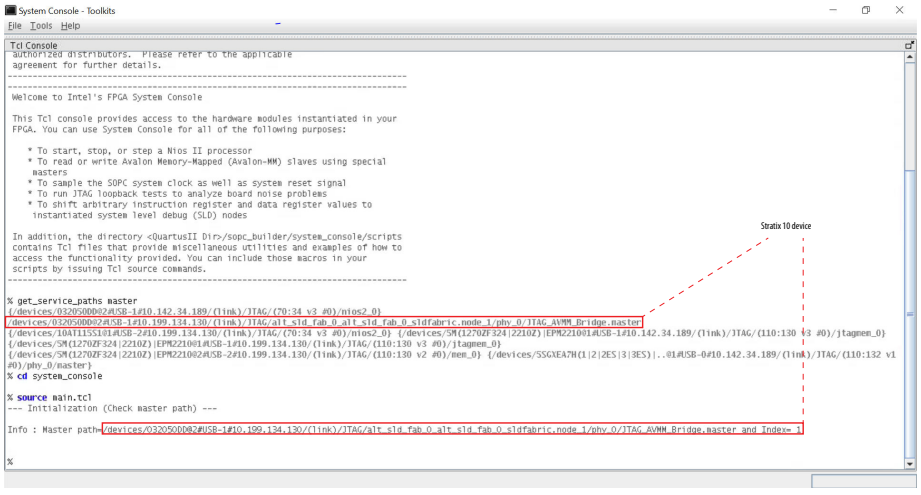
Note: This hardware test assumes that the System Console Control design is configured in duplex mode. Make your own modifications if using simplex mode design.

1. Launch the System Console tool from Intel Quartus Prime (**Tools ➤ System Debugging Tools ➤ System Console**).
2. In the **TCL Console** command prompt, type `get_service_paths master` to print a list of devices connected to your JTAG chain.
3. Open the `main.tcl` Tcl script located in the `System Console` directory in any text editor of your choice and locate the following line.

```
set master_index [expr {$master_list_length - <your offset>}]
```

4. Adjust the `master_index` offset as necessary to reflect your JTAG chain configuration such that the `master_index` always points to the Intel Stratix 10 device and save the file.
5. In the **TCL Console** command prompt, navigate to the `system_console` directory (`cd system_console`) and execute the `main.tcl` script (`source main.tcl`). Your **TCL Console** window should resemble the following figure.

Figure 5. Source main.tcl

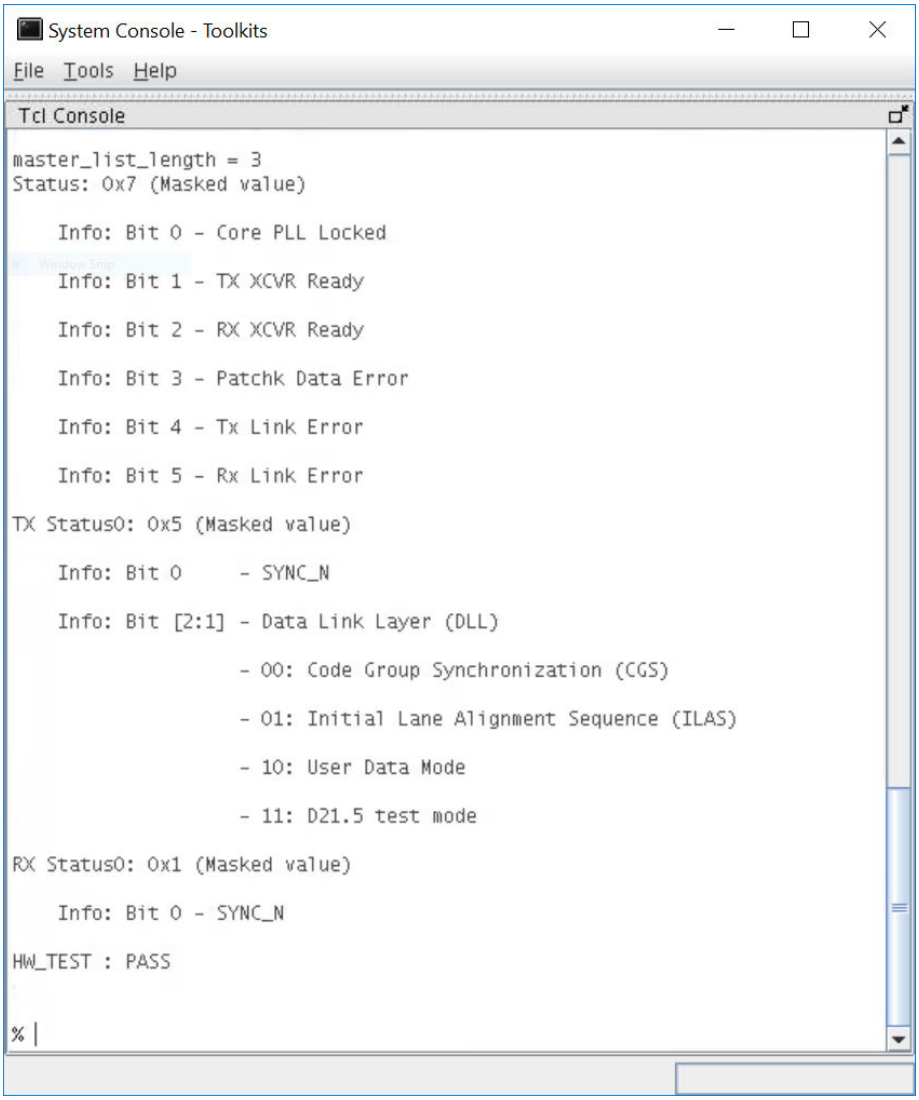


6. Type `start_basic_test` at the command prompt to execute the link setup and test procedure.

This procedure executes a set of instructions to set up the pattern generator and checker to transmit and check PRBS pattern, configure the JESD204B IP PHY internal serial loopback mode and report link status.

The following figure illustrates the expected result from a successful link setup and test.

Figure 6. Successful Test in the System Console



7. In the event that the test fails due to a lane deskew error, use the **rbd_offset** procedure (described in the following table) to offset the default RBD setting. Refer to the *JESD204B Intel FPGA IP User Guide* for more details on using the RBD offset.

Table 5. Procedures in the main.tcl System Console Script

The table describes useful procedures in the **main.tcl** that may be helpful in debugging.

Procedure	Values	Description
get_service_paths	{master}	Reports all devices that are connected to the JTAG chain. Use this information to set the master index to point to the Intel Stratix 10 device
get_master_index	N/A	Set the targeted device master index. Use get_service_paths master to determine the offset of the Intel Stratix 10 device in the JTAG chain, and edit the offset in this procedure accordingly.
start_basic_test	N/A	Main procedure that sets up link serial loopback mode, pattern generator and checker test mode, pulses sysref and reports link status
reset	N/A	Global reset
force_link_frame_reset	{0,1}	0: Deassert link and frame resets 1: Assert and hold link and frame resets *Note:* Link and frame clock domains should be held in reset while writing to JESD204B IP CSR
sloopback	{0,1}	0: Disable internal serial loopback 1: Enable internal serial loopback
set_testmode	{alt, ramp, prbs}	alt: Set pattern generator and checker to alternate pattern ramp: Set pattern generator and checker to ramp pattern prbs: Set pattern generator and checker to PRBS pattern
rbd_offset	{integer}	Adjust RBD offset value to eliminate RX lane deskew error.
sysref	N/A	Single pulse sysref
read_status_pio	N/A	Read status PIO registers. PIO status configuration: Bit 0 — Core PLL locked Bit 1 — TX transceiver ready Bit 2 — RX transceiver ready Bit 3 — Pattern checker mismatch error Bit 4 — TX link error (use read_err_status procedure to report error description) Bit 5 — RX link error (use read_err_status procedure to report error description)
read_err_status	N/A	Read JESD204B IP error status registers. Refer to the JESD204B IP register maps for detailed description of status registers.
clear_err_status	N/A	Clear JESD204B IP error status registers
read_rx_status0	N/A	Read JESD204B IP rx_status0 register. Refer to the JESD204B IP register maps for detailed description of status registers
read_tx_status0	N/A	Read JESD204B IP tx_status0 register. Refer to the JESD204B IP register maps for detailed description of status registers.
read_rx_syncn_sysref_ctrl	N/A	Read JESD204B IP syncn_sysref_ctrl register. Refer to the JESD204B IP register maps for detailed description of status registers
wait_seconds	{integer}	Wait for {integer} seconds
wait_minutes	{integer}	Wait for {integer} minutes

Related Information

- JESD204B Intel FPGA IP User Guide
- Intel FPGA JESD204B RX Address Map and Register Definitions
- Intel FPGA JESD204B TX Address Map and Register Definitions

1.2. Design Example Detailed Description

1.2.1. Features

This design example has the following key features:

- System Console using Tcl script control mechanism
- Synthesis and simulation flows
- Configurable transport layer and pattern generator and checker modules
- Power-on self test with the following configurable test patterns:
  - Alternating
  - Ramp
  - PRBS
- Supports simplex (RX only, TX only) and duplex (both RX and TX) data path modes
- Supports option for 3-wire SPI

1.2.2. Hardware and Software Requirements

Intel uses the following hardware and software to test the example designs:

- Intel Quartus Prime Pro Edition software
- Intel Stratix 10 GX FPGA Development Kit

1.2.3. Supported Configurations

The design examples only support a limited set of JESD204B IP core parameter configurations. The IP parameter editor allows you to generate a design example only if the parameter configurations matches the following table.

Note: If you are not able to generate a design example that fully matches your desired parameter settings, choose the closest allowable parameter values for generation. Modify the post-generated design parameters manually in the Intel Quartus Prime software to match your desire parameter settings. Refer to the *JESD204B IP Core User Guide* for more details on the rules and ranges that govern each IP core and transport layer parameter. Refer to *Customizing the Design Example* for more information about customizing the design example.

Table 6. Supported JESD204B IP Core Parameter Configurations

Table lists the parameters for the JESD204B IP core. The JESD204B IP core parameters are governed by various rules and ranges that are described in the *JESD204B Intel FPGA IP User Guide*. Please refer to the *JESD204B Intel FPGA IP User Guide* for more details on the legal parameter values. The value ranges given below should be considered as a subset of the allowable values described in the *JESD204B IP Core User Guide*.

JESD204B IP Parameters	Values
Wrapper Options	Both Base and Phy
Data Path	• Receiver • Transmitter • Duplex
JESD204B Subclass	1

continued...

JESD204B IP Parameters	Values
Data Rate	Any valid value(1)
PCS Option	• Enabled Hard PCS • Enabled Soft PCS
Bonding Mode	• Bonded • Non-bonded
PLL/CDR Reference Clock Frequency	Any valid value
Enable Bit Reversal and Byte Reversal	Any valid value
Enable Transceiver Dynamic Reconfiguration	Not supported
L	• 1 • 2 • 4 • 6(2) • 8
M	• 1 • 2 • 3(3) • 4 • 8 • 16 • 32
Enable manual F configuration	• No • Yes only for the following configuration: L=8, M=8, F=8, S=5, N'=12, N=12
F	• Auto calculated • Manual F configuration only allowed for the following configuration: L=8, M=8, F=8, S=5, N'=12, N=12
N	Integer, range 12 – 16
N'	• 16 • 12 only for the following configuration: L=8, M=8, F=8, S=5, N=12
S	Any valid value
K	Any valid value
Enable Scramble (SCR)	Any valid value
CS	Integer, range 0 – 3

continued...

(1) Refer to *JESD204B Intel FPGA IP User Guide* for more details on maximum and minimum data rates for your target device.

(2) L=6 is only allowed when F=1

(3) M=3 is only allowed for L=6

JESD204B IP Parameters	Values
CF	0
High Density User Data Format (HD)	• 0 • 1 only for F=1
Enable Error Code Correction (ECC_EN)	Any valid value

Related Information

- JESD204B Intel FPGA IP User Guide
- Customizing the Design Example on page 32

1.2.4. Presets

Standard presets allow instant entry of pre-selected parameter values in the **IP** and **Example Design** tabs. Select the presets at the lower right window in the parameter editor shown in Figure 3 on page 6.

The presets are applicable for JESD204B IP configurations that generate design examples. You can select one of the presets available for your target device to quickly generate a design example without having to set each parameter in the IP tab and verify that the specified parameters match the supported configurations. You can manually change any of the IP and example design parameters in the Platform Designer user interface after selecting a preset. However, you must ensure that your parameter selection falls within the supported configuration ranges detailed in Supported Configurations on page 13 for design example to generate successfully.

Note: Selecting a preset overwrites any pre-existing parameter selections for the IP core under the IP tab.

Table 7. Preset Settings

JESD204B IP Parameters	Preset 1 JESD204B Example Design (LMF = 222, 6.144 Gbps)	Preset 2 JESD204B Example Design (LMF = 888, 6.144 Gbps)
Wrapper Options	Both Base and Phy	Both Base and Phy
Data Path	Duplex	Duplex
JESD204B Subclass	1	1
Data Rate	6144 Mbps	6144 Mbps
PCS Option	Enabled Hard PCS	Enabled Hard PCS
Bonding Mode	Non-bonded	Non-bonded
PLL/CDR Reference Clock Frequency	153.6 MHz	153.6 MHz
Enable Bit Reversal and Byte Reversal	No	No
Enable Transceiver Dynamic Reconfiguration	No	No
L	2	8
M	2	8
Enable manual F configuration	No	Yes
F	2	8

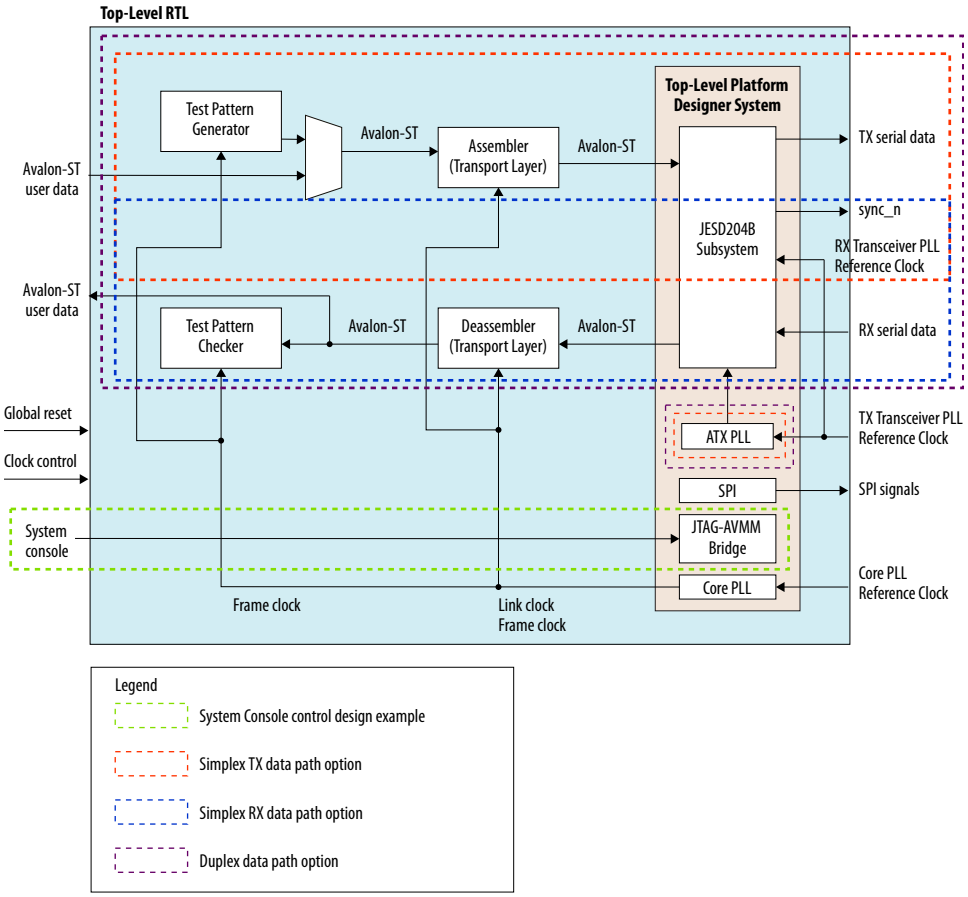
continued...

JESD204B IP Parameters	Preset 1 JESD204B Example Design (LMF = 222, 6.144 Gbps)	Preset 2 JESD204B Example Design (LMF = 888, 6.144 Gbps)
N	16	12
N'	16	12
S	1	5
K	16	32
Enable Scramble (SCR)	No	No
CS	0	0
CF	0	0
High Density User Data Format (HD)	0	0
Enable Error Code Correction (ECC_EN)	No	No

1.2.5. Functional Description

The design examples consist of various components. The following block diagrams show the design components and the top-level signals of the design examples.

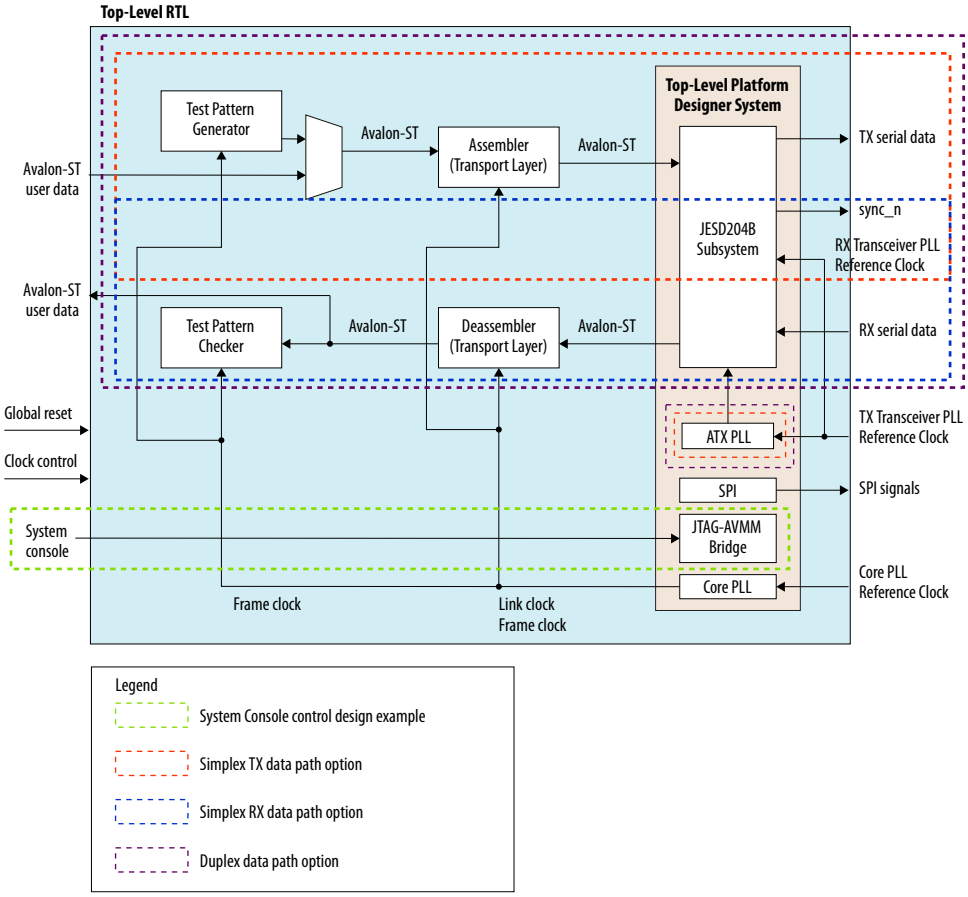
Figure 7. JESD204B Design Example Block Diagram



1.2.5.1. Design Components

The design example consists of various components. The following block diagram shows the design components and the top-level signals of the design example.

Figure 8. JESD204B Design Example Block Diagram

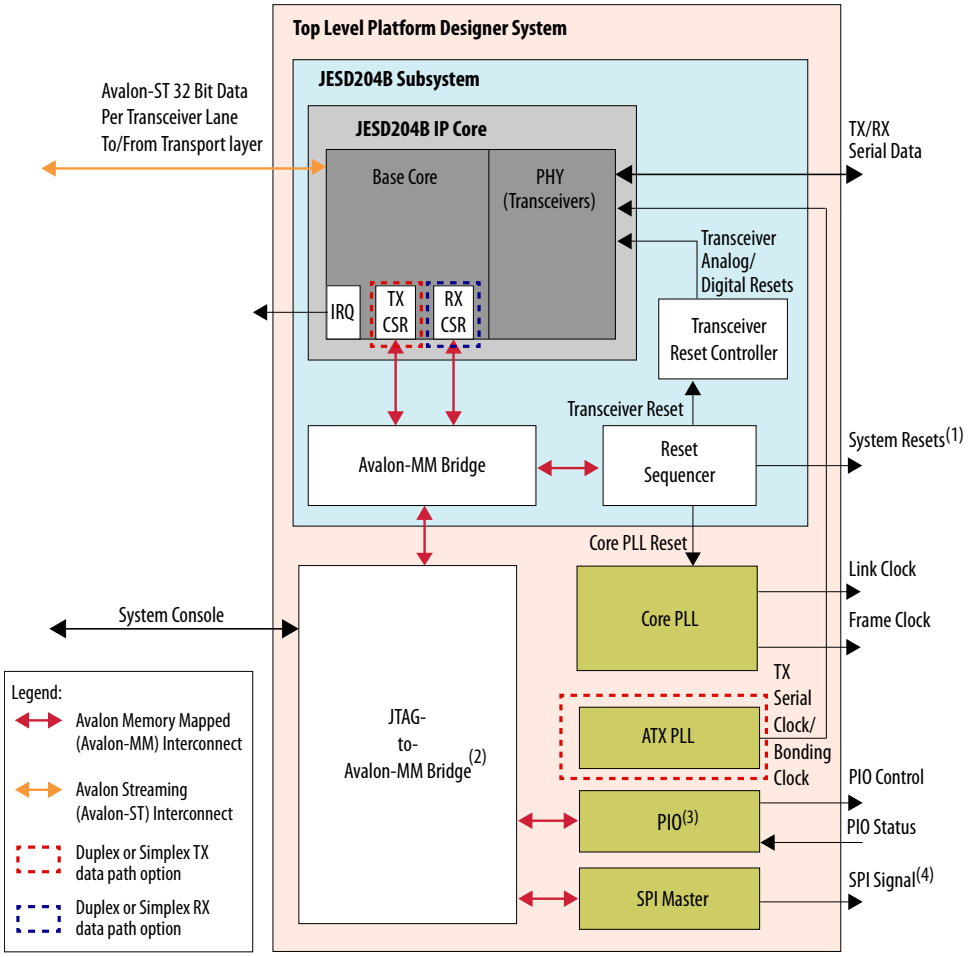


- Platform Designer system
  - JESD204B subsystem
  - JTAG to Avalon master bridge—For System Console Control design example only
  - Parallel I/O (PIO)
  - ATX PLL
  - Core PLL
  - Serial Port Interface (SPI)—master module
- Test pattern generator (For duplex and simplex TX data path only)
- Test pattern checker (For duplex and simplex RX data path only)
- Assembler—TX transport layer (For duplex and simplex TX data path only)
- Deassembler—RX transport layer (For duplex and simplex RX data path only)

1.2.5.1.1. Platform Designer System Component

The Platform Designer system instantiates the JESD204B IP core data path and supporting peripherals.

Figure 9. Platform Designer System for System Console Control Design Example



Notes:

1. System resets comprise the following resets: TX/RX JESD204B IP core CSR resets, TX/RX link resets, TX/RX frame resets.
2. This module is replaced by Avalon-MM Bus Functional Module (BFM) in the simulation flow.
3. Parallel input/output modules. Parallel 32-bit output for control signals from JTAG to -Avalon master bridge to HDL components. Parallel 32-bit input for status signals from HDL components to JTAG to Avalon master.
4. If Generate 3-Wire SPI Module option is not selected, 4-wire SPI signal to external converter SPI interface. If Generate 3-Wire SPI Module option is selected, 3-wire SPI signal to external converter SPI interface.

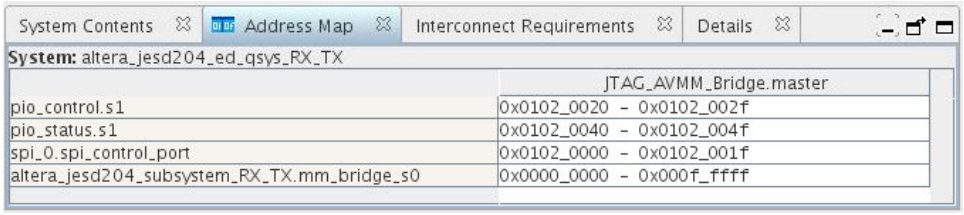
The top level Platform Designer system instantiates the following modules:

- Platform Designer system
  - JESD204B subsystem
  - JTAG to Avalon master bridge
  - Parallel I/O (PIO)
  - ATX PLL
  - Core PLL
  - Serial Port Interface (SPI)—master module

The following are the key features of the top level Platform Designer system:

- Supports System Console control design example
- Supports 3 data path types:
  - Duplex—Both TX and RX data paths present
  - Simplex TX—Only TX data path present
  - Simplex RX—Only RX data path present
- The JESD204B subsystem, parallel I/O and SPI master modules are connected to the JTAG to Avalon master bridge module via the Avalon Memory-Mapped (Avalon-MM) interface.
- JTAG to Avalon master bridge provides a link to the user via System Console. You can control the behavior of the design example via Tcl scripts executed in the System Console interface.
- TX data path flow:
  - Input: 32-bit per transceiver lane Avalon Streaming (Avalon-ST) input from assembler (TX transport layer)
  - Output: TX serial data
- RX data path flow:
  - Input: RX serial data from either external converter source or internal serial loopback
  - Output: 32-bit per transceiver lane Avalon Streaming (Avalon-ST) output to deassembler (RX transport layer)
- SPI master module links out to the SPI configuration interface of external converters via a 3- or 4-wire SPI interconnect (depending on Generate 3-Wire SPI Module setting).
- SPI master module handles the serial transfer of configuration data to the SPI interface on the converter end
- The ATX PLL generates the serial clock for clocking the TX serial data
  - ATX PLL module generated for duplex and simplex TX data path only
- The core PLL generates the following clocks for the system:
  - Link clock
  - Frame clock

Figure 10. Top Level Platform Designer Address Map



JESD204B Subsystem in Platform Designer

The JESD204B subsystem instantiates the following modules:

- JESD204B Intel FPGA IP core
- Reset sequencer
- Transceiver PHY reset controller
- Avalon-MM bridge

JESD204B IP Core

The generated design example is a self-contained system with its own JESD204B IP core instantiation that is separate from the IP core that is generated from the **IP** tab. The JESD204B IP base core and PHY layer connect to System Console through the Avalon-MM interconnect. The JESD204B IP core uses three separate Avalon-MM ports:

- Base core TX data path—For accessing the TX CSR
- Base core RX data path—For accessing the RX CSR
- PHY layer—For accessing the transceiver PHY CSR

The structure of the design example varies depending on the values of these JESD204B IP core parameters:

- Data path:
  - Duplex—Both TX and RX data paths and CSR interfaces present
  - TX only—Only TX data path and CSR interface present
  - RX only—Only RX data path and CSR interface present

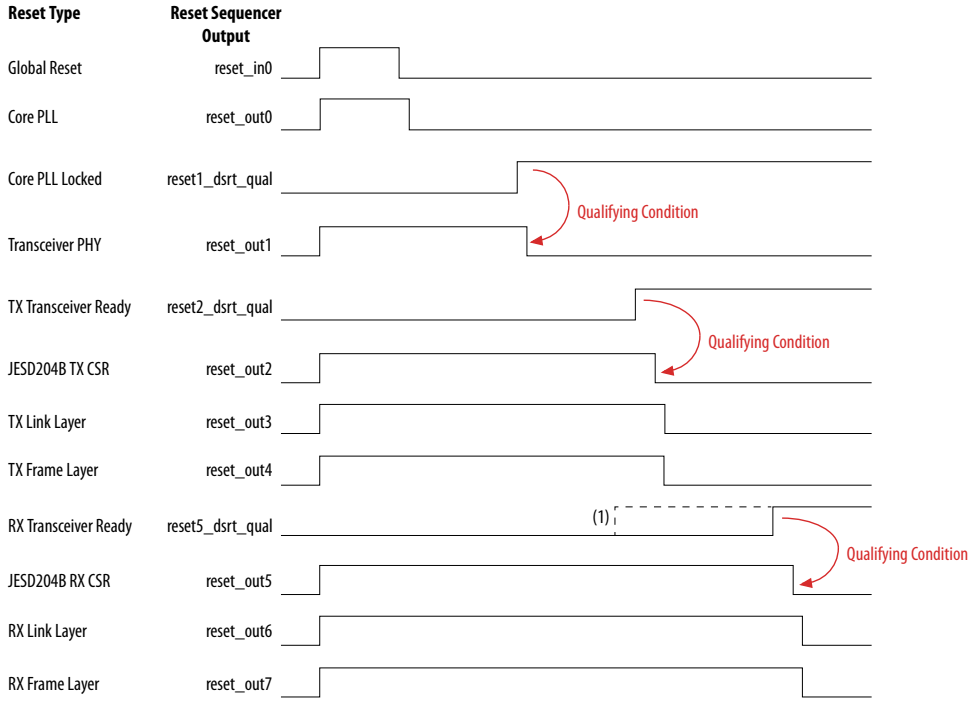
Reset Sequencer

The reset sequencer is a standard Platform Designer component in the **IP Catalog** standard library. The reset sequencer generates the following system resets to reset various modules in the system:

1. Core PLL reset—resets the core PLL
2. Transceiver reset—resets the JESD204B IP core PHY module
3. TX/RX JESD204B IP core CSR reset—resets the TX/RX JESD204B IP core CSRs
4. TX/RX link reset—resets the TX/RX JESD204B IP core base module and transport layer
5. TX/RX frame reset—resets the TX/RX transport layer, upstream and downstream modules

The reset sequencer has hard and soft reset options. The hard reset port connects to the global reset input pin in the top level design. The soft reset is activated via Avalon-MM interface by TCL scripts (System Console control). When you assert a hard or soft reset, the reset sequencer cycles through all the various module resets based on a pre-set sequence. The figure below illustrates the sequence and also shows how the reset sequencer output ports correspond to the modules that are being reset.

Figure 11. Reset Sequence



Note:

(1) In the event that the RX transceiver ready (reset5_dsrt_qual) asserts before the TX transceiver ready (reset2_drst_qual), the RX CSR, RX link layer, and RX frame layer will remain in reset until TX CSR, TX link layer, and TX frame layer are out of reset.

Note: For Intel Stratix 10 devices, reset deassertion staggering of TX/RX analog and digital reset happens before the assertion of TX/RX ready. The reset staggering may incur long simulation time. You may observe the staggering of TX and RX reset through `tx_analogreset_stat`, `tx_digitalreset_stat`, `rx_analogreset_stat`, and `rx_digitalreset_stat` respectively.

Transceiver PHY Reset Controller

The transceiver PHY reset controller is a standard Platform Designer component in the **IP Catalog** standard library. This module takes the transceiver PHY reset output from the reset sequencer and generates the proper analog and digital reset sequencing for the transceiver PHY module.

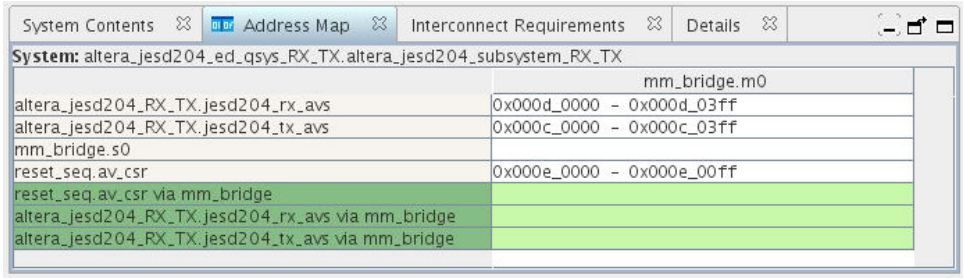
Avalon-MM Bridge

All the Avalon-MM submodules in the JESD204B subsystem are connected via Avalon-MM interconnect to a single Avalon-MM bridge. This bridge is the single interface for Avalon-MM communications into and out of the subsystem.

JESD204B Subsystem Address Map

Access the address map of the submodules in the JESD204B subsystem by clicking on the **Address Map** tab in the Platform Designer window.

Figure 12. JESD204B Subsystem Address Map



JTAG to Avalon Master Bridge

The JTAG to Avalon master bridge is a standard Platform Designer component in the IP Catalog standard library. This module provides a connection between a host system and the Platform Designer system via the respective physical interfaces; JTAG on the host system end and Avalon-MM on the Platform Designer system end. Host systems can initiate Avalon Memory-Mapped (Avalon-MM) transactions by sending encoded streams of bytes via JTAG interface. The module supports reads and writes, but not burst transactions.

Related Information

Platform Designer System Component on page 18

Parallel I/O

Parallel I/O (PIO) modules provide general input/output (I/O) access from the Avalon master (JTAG to Avalon master bridge). There are two sets of 32-bit PIO registers:

- Status registers—input from the HDL components to the Avalon master
- Control registers—output from the Avalon master to the HDL components

The registers are assigned in the top level HDL file (`io_status` for status registers, `io_control` for control registers). The tables below describe the signal connectivity for the status and control registers.

Table 8. Signal Connectivity for Status Registers

Bit	Signal
0	Core PLL locked
1	TX transceiver ready (for duplex and simplex TX data path only)
2	RX transceiver ready (for duplex and simplex RX data path only)
3	Test pattern checker data error (for duplex and simplex RX data path only)
4	TX link error (for duplex and simplex TX data path only)
5	RX link error (for duplex and simplex RX data path only)

Table 9. Signal Connectivity for Control Registers

Bit	Signal
0	RX serial loopback enable (for duplex data path only)
30	Global reset
31	Sysref

ATX PLL

Note: This module is only available in the design example when the duplex or simplex TX data path option is selected.

The ATX PLL is a standard Platform Designer component in the **IP Catalog** standard library. This module supplies a low-jitter serial clock to the transceiver PHY module. The reference clock input to the ATX PLL comes from an external source.

For simplex TX variant, the frequency selection in the **PLL/CDR Reference Clock Frequency** drop-down list in the JESD204B IP parameter editor is disabled. The design example generates the ATX PLL with the reference clock frequency of either:

- Hard PCS: `data_rate/20`
- Soft PCS: `data_rate/40`

Refer to Changing the Data Rate or Reference Clock Frequency on page 33 for more information about modifying the ATX PLL reference clock frequency to suit your application.

For duplex variant, the ATX PLL reference clock frequency shares the frequency with the CDR reference clock. You must select the frequency from the **PLL/CDR Reference Clock Frequency** drop-down list in the IP parameter editor.

For the ATX PLL reference clock frequencies supported range, refer to the *Intel Stratix 10 Device Datasheet*.

Core PLL

The core PLL module generates the clocks for the FPGA core fabric. An IOPLL module is instantiated as core PLL.

The core PLL uses an external clock input as its reference clock to generate two derivative clocks from a single VCO:

- Link clock
- Frame clock

Table 10. Core PLL Ouputs

Clock	Formula	Description
Link Clock	Serial data rate/40	The link clock clocks the JESD204B IP core link layer and the link interface of the transport layer.
Frame Clock	Serial data rate/(10 × F)	The frame clock clocks the transport layer, test pattern generators and checkers, and any downstream modules in the FPGA core fabric.

For the frame clock, when the **F** parameter is 1 or 2, the resulting frame clock frequency can easily exceed the capability of the core PLL to generate and close timing. The top level RTL file, (`altera_jesd204_ed_<data path>.sv`), defines the frame clock division factor parameters, F1_FRAMECLK_DIV (for cases with F = 1) and F2_FRAMECLK_DIV (for cases with F = 2). This factor enables the transport layer and test pattern generator to operate at a divided factor of the required frame clock rate by widening the data width accordingly.

For this design example, F1_FRAMECLK_DIV is set to 4 and F2_FRAMECLK_DIV is set to 2. As an example, the actual frame clock for a serial data rate of 10 Gbps and F = 1 is:

```
(10000/(10 × 1)) / F1_FRAMECLK_DIV = 1000 / 4 = 250 MHz
```

Frame Clock and Link Clock Relationship

The frame clock and link clock are synchronous. For the derived F mode, the ratio of `link_clk` period to `frame_clk` period is given by this formula:

```
link_clk period to frame_clk period ratio = 32xL/(MxSxN')
```

Table 11. $f_{TXframe}$ and $_{RXframe}$ for Different F Parameter Settings

- f_{TXlink} is the TX link clock frequency
- f_{RXlink} is the RX link clock frequency

F Parameter	$f_{TXframe}$(txframe_clk frequency)	$f_{RXframe}$(rxframe_clk frequency)
1	f_{TXlink}x(4/F1_FRAMECLK_DIV)	f_{RXlink}x(4/F1_FRAMECLK_DIV)
2	f_{TXlink}x(2/F2_FRAMECLK_DIV)	f_{RXlink}x(2/F2_FRAMECLK_DIV)
4	f_{TXlink}	f_{RXlink}
8	f_{TXlink}/2	f_{RXlink}/2

Note: The IOPLL is generated with the **Use Nondedicated Feedback Path** option being disabled (default setting). You can turn on the **Use Nondedicated Feedback Path** option in the IP parameter editor to utilize the clock resources efficiently after the design example is successfully generated. Refer to the *Clock Feedback Modes* section of *Intel Stratix 10 Clocking and PLL User Guide* for more information about this option.

Related Information

Clock Feedback Modes section of the Intel Stratix 10 Clocking and PLL User Guide

SPI Master

The SPI master module is a standard Platform Designer component in the **IP Catalog** standard library. This module uses the SPI protocol to facilitate the configuration of external converters (for example, ADC, DAC, external clock modules) via a structured register space inside the converter device. The SPI master has an Avalon-MM interface that connects to the Avalon master (JTAG to Avalon master bridge) via the Avalon-MM interconnect and can receive configuration instructions from the Avalon master.

This module is configured to a 4-wire, 24-bit width interface. If the **Generate 3-Wire SPI Module** option is selected, an additional module is instantiated to convert the 4-wire output of the SPI master to 3-wire.

For more details on the SPI master module, refer to the *JESD204B Intel FPGA IP User Guide*.

Related Information

JESD204B Intel FPGA IP User Guide

1.2.5.1.2. Transport Layer

The transport layer in the design example consists of an assembler at the TX path and a deassembler at the RX path. The transport layer for both the TX and RX path is instantiated in the top level RTL file, not in the Platform Designer project.

Note: When the simplex TX data path option is selected, only the assembler is instantiated in the design example. When the simplex RX data path option is selected, only the deassembler is instantiated in the design example. When the duplex data path option is selected, both assembler and deassembler is instantiated in the design example.

The transport layer provides the following services to the application layer (AL) and the data link layer (DLL):

- Assembler at the TX path:
  - Maps the conversion samples from the AL (through the Avalon-ST interface) to a specific format of non-scrambled octets, before streaming them to the DLL.
  - Reports AL error to the DLL if it encounters a specific error condition on the Avalon-ST interface during TX data streaming.
- Deassembler at the RX path:
  - Maps the descrambled octets from the DLL to a specific conversion sample format before streaming them to the AL (through the Avalon-ST interface).
  - Reports AL error to the DLL if it encounters a specific error condition on the Avalon-ST interface during RX data streaming.

The transport layer has many customization options and you can modify the transport layer RTL to customize it to your specifications. Furthermore, for certain parameters like L, F, and N, the transport layer shares the CSR values with the JESD204B IP core.

For more details on the implementation of the transport layer in RTL and customization options, refer to the *JESD204B Intel FPGA IP User Guide*.

Related Information

JESD204B Intel FPGA IP User Guide

1.2.5.1.3. Test Pattern Generator

Note: This module is only available in the design example when the duplex or simplex TX data path option is selected.

The test pattern generator generates either a parallel PRBS, alternate checkerboard, or ramp wave, and sends it to the transport layer during test mode. The test pattern generator is implemented in the top level RTL file, not in the Platform Designer project.

Related Information

JESD204B Intel FPGA IP User Guide

1.2.5.1.4. Test Pattern Checker

Note: This module is only available in the design example when the duplex or simplex RX data path option is selected.

The test pattern checker checks either a parallel PRBS, alternate checkerboard, or ramp wave from the transport layer during test mode and outputs an error flag if there are any data mismatches. The test pattern checker is implemented in the top level RTL file, not in the Platform Designer project.

Related Information

JESD204B Intel FPGA IP User Guide

1.2.5.2. Clocking Scheme

The main reference clocks for the design example are `refclk_core` and `refclk_xcvr`. These clocks must be supplied from a single external source (i.e `refclk_core` and `refclk_xcvr` must be synchronous to one another). The `refclk_core` is the reference clock for the core PLL and the `refclk_xcvr` is the reference clock for the TX/RX transceiver PHY. The core PLL generates the `link_clk` and `frame_clk` from `refclk_core`.

The `link_clk` clocks the JESD204B IP core link layer and link interface of the transport layer. The `frame_clk` clocks the transport layer, test pattern generator and checker modules, and any downstream modules. An external source supplies a clock called the `mgmt_clk` to clock the Avalon-MM interfaces of Platform Designer components.

Table 12. System Clocking for the Design Example

Clock	Description	Source	Modules Clocked
`refclk_core`	Reference clock for the core PLL	External	Core PLL
`refclk_xcvr`	Reference clock for the ATX PLL, RX transceiver PHY	External	ATX PLL, RX transceiver PHY
`link_clk`	Link layer clock	refclk_core	JESD204B IP core link layer, transport layer link interface
`frame_clk`	Frame layer clock	refclk_core	Transport layer, test pattern generator and checker, downstream modules
`mgmt_clk`	Control plane clock	External	Avalon-MM interfaces

1.2.6. Simulation

Execute the simulation by running the relevant simulation run scripts in the supported simulator environment. The following table shows the simulators supported along with the relevant run scripts.

Table 13. Supported Simulators

Simulators	Simulation Directory	Run Script
Riviera-PRO	`/testbench/aldec/`	`run_tb_top.tcl`
NCSim	`/testbench/cadence/`	`run_tb_top.sh`

continued...

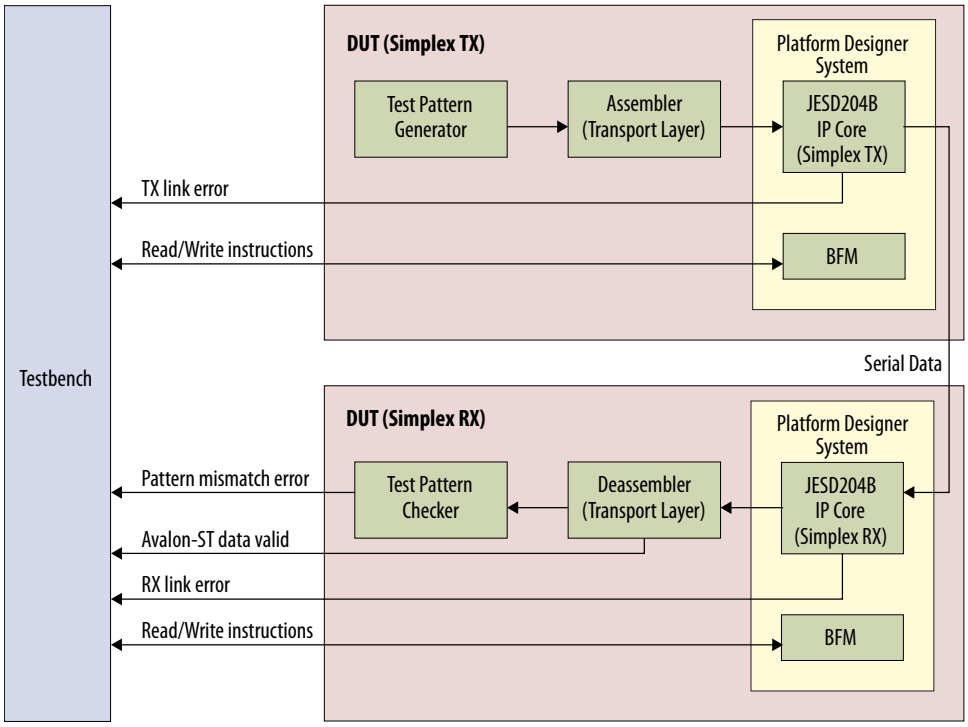
Simulators	Simulation Directory	Run Script
ModelSim	`/testbench/mentor/`	`run_tb_top.tcl`
VCS	`/testbench/synopsys/vcs/`	`run_tb_top.sh`
VCS MX	`/testbench/synopsys/vcsmx/`	`run_tb_top.sh`
Xcelium Parallel	`/testbench/xcelium/`	`run_tb_top.sh`

The design generates the simulation results which include the transcript or log files in the relevant simulation directory.

1.2.6.1. Testbench

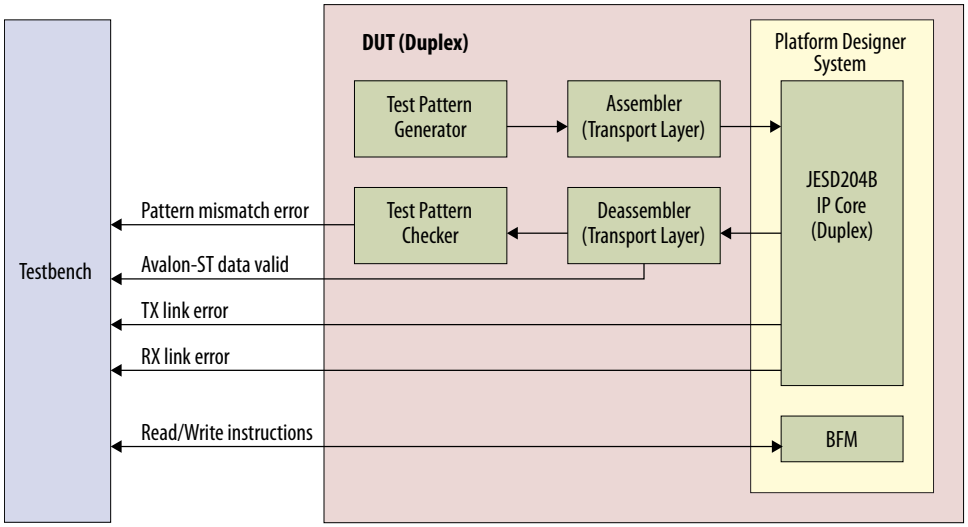
The simulation design-under-test (DUT) is the generated design example which includes a synthesizable pattern generator and checker. The figures below show the testbench block diagram for simplex and duplex options.

Figure 13. Simulation Testbench Block Diagram (Simplex TX or RX)



Note: Both simplex TX and simplex RX design examples generate the same testbench. The testbench instantiates two DUTs: one simplex TX DUT, one simplex RX DUT. The TX serial data output of the simplex TX DUT is connected to the RX serial data input of the simplex RX DUT. The testbench issues separate Avalon-MM read/write instructions to the simplex TX and simplex RX DUTs respectively.

Figure 14. Simulation Testbench Block Diagram (Duplex)



The simulation flow replaces the JTAG to Avalon master bridge module in the Platform Designer system of the System Console Control design example with the Avalon-MM master bus functional model (BFM). This BFM enables a testbench to send Avalon-MM read/write commands to the design example registers to mimic the functionality of System Console.

The testbench provided in the simulation flow (`/testbench/models/tb_top.sv`) executes the following steps:

1. Reset DUT.
2. Initialize BFM.
3. Execute Avalon-MM commands to initialize the DUT in the following mode:
   - Internal serial loopback mode (for duplex option only)
   - Pattern generator/checker set to PRBS pattern
4. Wait for DUT to initialize to user mode.
5. Report JESD204B link status.

When simulation ends, the following messages are shown at end.

Table 14. Simulation Messages and Description

Message	Description
`Pattern Checker(s): Data error(s) found!`	Pattern mismatch errors found on the pattern checker
`Pattern Checker(s): OK!`	No errors found on the pattern checker
`Pattern Checker(s): No valid data found!`	No valid data received by pattern checker
`JESD204B Tx Core(s): Tx link error(s) found!`	Link errors reported by JESD204B IP TX
`JESD204B Tx Core(s): OK!`	No link errors reported by JESD204B IP TX
`JESD204B Rx Core(s): Rx link error(s) found!`	Link errors reported by JESD204B IP RX

continued...

Message	Description
`JESD204B Rx Core(s): OK!`	No link errors reported by JESD204B IP RX
`TESTBENCH_PASSED: SIM PASSED!`	Overall simulation passed
`TESTBENCH_FAILED: SIM FAILED!`	Overall simulation failed

1.2.7. Design Example Files

There are two flows for the design example: simulation and synthesis.

Table 15. Design Example Flows and Directory

Design Example Flow	Directory
Simulation	`<your project>/ed_sim`
Synthesis	`<your project>/ed_synth`

The following tables list the important folders and files for simulation and synthesis.

Table 16. Design Example Files for Simulation

File Type	File/Folder	Description
Run script files	`/testbench/aldec/run_tb_top.tcl`	TCL run script for Riviera-PRO simulator
	`/testbench/cadence/run_tb_top.sh`	Shell run script for NCSim simulator
	`/testbench/mentor/run_tb_top.tcl`	TCL run script for ModelSim simulator
	`/testbench/synopsys/vcs/run_tb_top.sh`	Shell run script for VCS simulator
	`/testbench/synopsys/vcsmx/run_tb_top.sh`	Shell run script for VCS MX simulator
	`/testbench/xcelium/run_tb_top.sh`	Shell run script for Xcelium simulator
Source files	`/testbench/models/altjesd_ed_qsys_<data path>.qsys`	Top level Platform Designer system project
	`/testbench/models/altjesd_ss_<data path>.qsys`	JESD204B subsystem Platform Designer system project
	`/testbench/models/ip/`	IP folder containing instantiated IP modules
	`/testbench/models/altera_jesd204_ed_<data path>.sv`	Top level HDL
	`/testbench/models/tb_top.sv`	Top level testbench
	`/testbench/spi_mosi_oe.v`	Output buffer HDL
	`/testbench/switch_debouncer.v`	Switch debouncer HDL
	`/testbench/pattern/`	Folder containing the test pattern generator and checker HDL
	`/testbench/transport_layer`	Folder containing assembler and de-assembler HDL.

Table 17. Design Example Files for Synthesis

File Type	File/Folder	Description
Intel Quartus Prime project files	altera_jesd204_ed_<data path>.qpf	Intel Quartus Prime project file
	altera_jesd204_ed_<data path>.qsf	Intel Quartus Prime settings file
Source files	altera_jesd204_ed_<data path>.sv	Top level HDL
	altera_jesd204_ed_<data path>.sdc	Synopsys* Design Constraints (SDC) file containing all timing/placement constraints
	transport_layer/	Folder containing assembler and de-assembler HDL
	pattern/	Folder containing the test pattern generator and checker HDL
	spi_mosi_oe.v	Output buffer HDL
	switch_debouncer.v	Switch debouncer HDL
	altjesd_ed_qsys_<data path>.qsys	Top level Platform Designer system project
	altjesd_ss_<data path>.qsys	JESD204B subsystem Platform Designer system project

1.2.8. Registers

Refer to the *JESD204B RX Address Map and Register Definitions* and *JESD204B TX Address Map and Register Definitions* for the list of registers.

Note: The following status bits are not applicable to Intel Stratix 10 devices:

- csr_pcfifo_full_err
- csr_pcfifo_empty_err

Related Information

- Intel FPGA JESD204B RX Address Map and Register Definitions
- Intel FPGA JESD204B TX Address Map and Register Definitions

1.2.9. Signals

Table 18. System Interface Signals

Signal	Clock Domain	Direction	Description
Clocks and Resets			
refclk_core	—	Input	Reference clock for FPGA core modules.
refclk_xcvr	—	Input	Reference clock for transceiver PHY.
mgmt_clk	—	Input	Reference clock for all peripherals connected via Avalon-MM interconnect.
global_rst_n	mgmt_clk	Input	Global reset signal from the push button. This reset is an active low signal and the deassertion of this signal is synchronous to the rising-edge of mgmt_clk.

Signal	Clock Domain	Direction	Description
Serial Data			
rx_serial_data[LINK*L-1:0]	refclk_xcvr	Input	Differential high speed serial input data. The clock is recovered from the serial data stream.
tx_serial_data[LINK*L-1:0]	refclk_xcvr	Output	Differential high speed serial output data. The clock is embedded in the serial data stream.

Signal	Clock Domain	Direction	Description
JESD204B			
sysref_out	mgmt_clk	Output	*SYSREF* signal for JESD204B Subclass 1 implementation.
sync_n_out	link_clk	Output	Indicates a SYNC_N from the receiver. This is an active low signal and is asserted 0 to indicate a synchronization request or error reporting.
tx_link_error	link_clk	Output	Error interrupt from JESD204B IP core indicating TX link error
rx_link_error	link_clk	Output	Error interrupt from JESD204B IP core indicating RX link error

Signal	Clock Domain	Direction	Description
Avalon- ST User Data			
avst_usr_din[LINK*TL_DATA_BUS_WIDTH-1:0]	frame_clk	Input	TX data from the Avalon-ST source interface. The TL_DATA_BUS_WIDTH is determined by the following formulas: • If F = 1, TL_DATA_BUS_WIDTH = F1_FRAMECLK_DIV*8*1*L*N/N_PRIME • If F = 2, TL_DATA_BUS_WIDTH = F2_FRAMECLK_DIV*8*2*L*N/N_PRIME • If F = 4, TL_DATA_BUS_WIDTH = 8*4*L*N/N_PRIME • If F = 8, TL_DATA_BUS_WIDTH = 8*8*L*N/N_PRIME
avst_usr_din_valid[LINK-1:0]	frame_clk	Input	Indicates whether the data from the Avalon-ST source interface to the transport layer is valid or invalid. • 0—data is invalid • 1—data is valid
avst_usr_din_ready[LINK-1:0]	frame_clk	Output	Indicates that the transport layer is ready to accept data from the Avalon-ST source interface. • 0—transport layer is not ready to receive data • 1—transport layer is ready to receive data
avst_usr_dout[LINK*TL_DATA_BUS_WIDTH-1:0]	frame_clk	Output	RX data to the Avalon-ST sink interface. The TL_DATA_BUS_WIDTH is determined by the following formulas: • If F = 1, TL_DATA_BUS_WIDTH = F1_FRAMECLK_DIV*8*1*L*N/N_PRIME • If F = 2, TL_DATA_BUS_WIDTH = F2_FRAMECLK_DIV*8*2*L*N/N_PRIME • If F = 4, TL_DATA_BUS_WIDTH = 8*4*L*N/N_PRIME • If F = 8, TL_DATA_BUS_WIDTH = 8*8*L*N/N_PRIME

continued...

Signal	Clock Domain	Direction	Description
avst_usr_dout_valid[LINK-1:0]	frame_clk	Output	Indicates whether the data from the transport layer to the Avalon-ST sink interface is valid or invalid. • 0—data is invalid • 1—data is valid
avst_usr_dout_ready[LINK-1:0]	frame_clk	Input	Indicates that the Avalon-ST sink interface is ready to accept data from the transport layer. • 0—Avalon-ST sink interface is not ready to receive data • 1—Avalon-ST sink interface is ready to receive data
avst_patchk_data_error [LINK-1:0]	frame_clk	Output	Output signal from pattern checker indicating a pattern check error.

Signal	Clock Domain	Direction	Description
SPI			
spi_MISO [4]	spi_SCLK	Input	Input data from external slave to the master.
spi_MOSI[4]	spi_SCLK	Output	Output data from the master to the external slaves.
spi_SDIO[5]	spi_SCLK	Input/ Output	Output data from the master to external slave. Input data from external slave to master
spi_SCLK	mgmt_clk	Output	Clock driven by the master to slaves, to synchronize the data bits.
spi_SS_n[2:0]	spi_SCLK	Output	Active low select signal driven by the master to individual slaves, to select the target slave. Defaults to 3 bits.

1.2.10. Customizing the Design Example

Use the following guidelines to customize the design example post-generation.

Related Information

AN804: Implementing ADC-Stratix 10 Multi-Link Design with JESD204B RX IP Core

1.2.10.1. Modifying the JESD204B IP Core Parameters

The Platform Designer tool allows only a limited set of design examples to be generated based on the JESD204B IP core parameters selected.

(4) When **Generate 3-Wire SPI Module** option is not enabled.

(5) When **Generate 3-Wire SPI Module** option enabled.

Perform the following instructions to modify the JESD204B IP core parameters post-generation:

1. Open the generated design example project in the Intel Quartus Prime software.
2. Open the `altjesd_ss_<data path>.qsys` system in Platform Designer.
3. In the **System Contents** tab, double-click the **altjesd_<data path>** module. This brings up the parameter editor that shows the current parameter settings of the JESD204B IP core.
4. Modify the parameters of the JESD204B IP core module as per your system specifications. When you are done, save the Platform Designer system (**File ➤ Save**).

   Note: The JESD204B IP core and transport layer imposes certain limits on the values that can be entered as parameters. Refer to the *JESD204B Intel FPGA IP User Guide* for a complete listing of the legal parameter values.
5. Click the **Generate HDL** to generate the HDL files needed for Intel Quartus Prime compilation.
6. After the HDL generation is completed, click the **Finish** to save your settings and exit Platform Designer.
7. You have to manually change the system parameters in the top level RTL file to match the parameters that you set in the Platform Designer project, if applicable. Open the top level RTL file (`altera_jesd204_ed_<data path>.sv`) in any text editor of your choice.
8. Modify the system parameters at the top of the file to match the new JESD204B IP core settings in the Platform Designer project, if applicable.
9. Save the file and compile the design in Intel Quartus Prime software as per the instructions in the Compiling and Testing the Design on page 8.

Related Information

JESD204B Intel FPGA IP User Guide

1.2.10.2. Changing the Data Rate or Reference Clock Frequency

When changing the data rate or reference clock frequency, you must consider the following:

- The relationships between the serial data rate, link clock, and frame clock as described in the *JESD204B Intel FPGA IP User Guide*.
- Change the PLL output clock settings according to Table 11 on page 24.
- Take note when changing the **F1_FRAMECLK_DIV** and **F2_FRAMECLK_DIV** frame clock division factor parameters in the top level RTL file `altera_jesd204_ed_<data path>.sv` for cases when F=1 or F=2. These parameters further divide-down the frame clock frequency requirement so the resulting clock frequency is within bounds of timing closure for the FPGA core fabric.

The frame clock and the link clock for the following cases share the same frequency:

- F=1—the default parameter value for F1_FRAMECLK_DIV=4
- F=2—the default parameter value for F2_FRAMECLK_DIV=2
- F=4

Perform the following instructions to modify the JESD204B IP core parameters post-generation:

1. Open the generated design example project in the Intel Quartus Prime software.
2. Open the top level `altjesd_ed_qsys_<data path>.qsys` in the Platform Designer.
3. In the **System Contents** tab, right-click the **altjesd_ss_<data path>** module and select **Drill into Subsystem**. This opens the `altjesd_ss_<data path>.qsys` Platform Designer subsystem.
4. Double-click the **altjesd_<data path>** module. This brings up the parameter editor that shows the current parameter settings of the JESD204B IP core.
5. Change the **Data rate** and **PLL/CDR Reference Clock Frequency** values to meet your system requirements.
6. Modify the clock frequency values of the **refclk_xcvr**, **link_clk**, **frame_clk** and **mgmt_clk** clock source modules as necessary to meet your system requirements. Double-click the clock source module to bring up the parameters editor and change the **Clock frequency** value as necessary. Ensure that the values match the clock frequency values that you have entered for the other modules above.
7. Navigate back to the top level **altjesd_ed_qsys_<data path>.qsys** hierarchy.
8. Double-click the **xcvr_atx_pll_0** module to bring up the parameters editor for the ATX PLL module.

   This is the module that generates the serial clock for the TX transceiver PHY.
9. Under the **PLL** subtab, locate the **Output Frequency** group and change the **PLL output frequency** and **PLL integer reference clock frequency** values to meet your system requirements.

   The PLL output frequency is half of the PLL output data rate. Ensure that the data rate and PLL reference clock values match the parameters that you entered into the JESD204B IP core module.
10. Double-click the **core_pll** module to bring up the parameters editor for the core PLL module.

    This is the module that generates the `link_clk` and `frame_clk` clocks that clock the core components.
11. Under the **PLL** subtab, change the **Reference Clock Frequency** value in the **General** group to meet your system requirements.

    Ensure that the reference clock frequency value matches the ones set for the JESD204B IP core and ATX PLL modules.
12. Change the `outclk0` group settings (which correspond to the `link_clk`) and `outclk1` group settings (which correspond to the `frame_clk`) where necessary.

    Ensure that the `link_clk` and `frame_clk` values satisfy the frequency requirements as described in the JESD204B IP Core User Guide.
13. Modify the clock frequency values of the **refclk_xcvr**, **refclk_core**, **link_clk**, **frame_clk** and **mgmt_clk** clock source modules as necessary to meet your system requirements. Double-click the clock source module to bring up the parameters editor and change the **Clock frequency** value as necessary. Ensure that the values match the clock frequency values that you have entered for the other modules in earlier steps.

14. Click the **Generate HDL** button to generate the HDL files needed for Intel Quartus Prime compilation.
15. After the HDL generation is completed, click the **Finish** to save your Platform Designer settings and exit the Platform Designer window.
16. If the frame_clk settings (outclk1 of the core_pll module) are such that **F1_FRAMECLK_DIV** or **F2_FRAMECLK_DIV** values are changed, change the parameters in the top level design file, `altera_jesd204_ed_<data path>.sv`.
17. Modify the clock constraints in the SDC constraints file (`altera_jesd204_ed_<data path>.sdc`) to reflect your new clock frequency values, if applicable. The following constraints should be modified:

```
 create_clock -name refclk_xcvr -period <clock period value in ns>
[get_nodes refclk_xcvr]
create_clock -name {u_altjesd_ed_qsys_<data_path>|core_pll|
core_pll_refclk} -period <clock period value in ns> [get_nodes refclk_core]
create_clock -name mgmt_clk -period <clock period value in ns> [get_nodes
mgmt_clk]
```

18. Save the file and compile the design in Intel Quartus Prime software as per the instructions in the Compiling and Testing the Design on page 8.

Related Information

JESD204B Intel FPGA IP User Guide

1.3. JESD204B Intel Stratix 10 FPGA IP Design Example User Guide Archives

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

IP Core Version	User Guide
17.1	Intel FPGA JESD204B Design Example User Guide for Intel Stratix 10 Devices

1.4. Document Revision History for JESD204B Intel Stratix 10 FPGA IP Design Example User Guide

Document Version	Intel Quartus Prime Version	Changes
2018.08.10	18.0	• Added a note in the *Reset Sequencer* section that for Intel Stratix 10 devices, reset deassertion staggering of TX/RX analog and digital reset happens before the assertion of TX/RX ready. • Changed the target development board for the design example from Signal Integrity (SI) development board to FPGA development board. • Updated the *Intel Stratix 10 FPGA Development Kit Board Connectivity* table to include the FPGA development board information.

Document Version	Intel Quartus Prime Version	Changes
		• Updated the Clock Control GUI Setting figure. • Shortened the following Platform Designer file names due to Windows limitation: — altera_jesd204_ed_qsys_<data path>.qsys to altjesd_ed_qsys_<data path>.qsys — altera_jesd204_subsystem_<data path>.qsys to altjesd_ss_<data path>.qsys • Edited the Platform Designer System for System Console Control Design Example figure. — Removed signal connections that indicated support for dynamic transceiver reconfiguration. The design example does not support dynamic transceiver reconfiguration. — Added that the output from ATX PLL could either be TX serial clock or TX bonding clock. — Removed "Core PLL reset" and "JESD204B IP core SerDes PHY reset" from Note 1. These resets are connected internally.

Date	Version	Changes
November 2017	2017.11.06	• Added information about simplex and duplex ATX reference clock frequencies. • Defined (`altera_jesd204_ed_<data path>.sv`) as the top level RTL file in *Core PLL*. • Added *Frame Clock and Link Clock Relationship* subsection. • Defined top level RTL file in *Changing the Data Rate or Reference Clock Frequency*. • Updated SDC constraint to be modified in *Changing the Data Rate or Reference Clock Frequency*. • Added *get_master_index* procedure in *Procedures in the main.tcl System Console Script* table. • Updated document title. • Updated instances of Qsys to Platform Designer.
May 2017	2017.05.08	• Added new directories and descriptions in *Directory Structure*. • Updated steps in *Generating the Design*. • Updated design example parameters and descriptions *Design Example Parameters*. • Added new simulators in *Simulating the Design*. • Updated steps in *Compiling and Testing the Design*. • Added *Hardware Test for System Console Control Design Example*. • Updated the supported configuration in *Supported Configurations*. • Updated preset settings. • Updated *JESD204B Design Example Block Diagram*. • Updated descriptions and figures in *Platform Designer System Components*. • Updated System Clocking for the Design Example. • Added `tx_link_error`, `rx_link_error`, and `spi_SDIO` signals in *System Interface Signals*. • Updated *Testbench*.
December 2016	2016.12.09	Initial release.