## Contents

### SDI II MegaCore Function Quick Reference

1-1

### SDI II MegaCore Function Overview

2-1
- General Description
- SMPTE372 Dual Link Support
- SMPTE RP168 Switching Support
- SD 20-Bits Interface for Dual/Triple Standard
- Tx PLL Dynamic Switching
- Resource Utilization

### Getting Started

3-1
- Installation and Licensing
- OpenCore Plus Evaluation
- OpenCore Plus Time-Out Behavior
- Design Walkthrough
  - Creating a New Quartus II Project
  - Launching MegaWizard Plug-In Manager
  - Parameterizing the MegaCore Function
  - Generating a Design Example and Simulation Testbench
  - Simulating the SDI II MegaCore Function Design
- Compiling the SDI II MegaCore Function Design
- Programming an FPGA Device
- Design Reference
  - SDI II MegaCore Function Parameters
  - SDI II MegaCore Function Component Files
  - Video Pattern Generator Signals
  - Transceiver Reconfiguration Controller Signals
  - Reconfiguration Router Signals
  - Design Examples

### Functional Description

4-1
- Protocol
The Altera® Serial Digital Interface (SDI) II MegaCore® function is the next generation SDI intellectual property (IP).

The SDI II MegaCore function is part of the MegaCore IP Library, which is distributed with the Quartus® II software and downloadable from the Altera website at www.altera.com.

**Note:** For system requirements and installation instructions, refer to *Altera Software Installation & Licensing.*

### Table 1-1: Brief Information About the SDI II MegaCore Function

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Release Information</strong></td>
<td></td>
</tr>
<tr>
<td>Version</td>
<td>13.0SP1</td>
</tr>
<tr>
<td>Release Date</td>
<td>July 2013</td>
</tr>
<tr>
<td>Ordering Code</td>
<td>IP-SDI-II</td>
</tr>
<tr>
<td>Product ID(s)</td>
<td>0111</td>
</tr>
<tr>
<td>Vendor ID</td>
<td>6AF7</td>
</tr>
<tr>
<td><strong>IP Core Information</strong></td>
<td></td>
</tr>
<tr>
<td>SDI Data Rate Support</td>
<td>• 270-Mbps SD-SDI, as defined by SMPTE259M specification</td>
</tr>
<tr>
<td></td>
<td>• 1.485-Gbps or 1.4835-Gbps HD-SDI, as defined by SMPTE292M specification</td>
</tr>
<tr>
<td></td>
<td>• 2.97-Gbps or 2.967-Gbps 3G-SDI, as defined by SMPTE424M specification</td>
</tr>
<tr>
<td></td>
<td>• Dual link HD-SDI, as defined by SMPTE372M specification</td>
</tr>
<tr>
<td></td>
<td>• Dual standard support for SD-SDI and HD-SDI</td>
</tr>
<tr>
<td></td>
<td>• Triple standard support for SD-SDI, HD-SDI, and 3G-SDI</td>
</tr>
<tr>
<td></td>
<td>• 20-bit interface support for SD-SDI</td>
</tr>
<tr>
<td></td>
<td>• SMPTE425M level A support (direct source image formatting)</td>
</tr>
<tr>
<td></td>
<td>• SMPTE425M level B support (dual link mapping)</td>
</tr>
<tr>
<td></td>
<td>• SMPTE RP168 switching support</td>
</tr>
</tbody>
</table>
### IP Core Information

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Features</strong></td>
<td>• Multiple SDI standards and video formats</td>
</tr>
<tr>
<td></td>
<td>• Video payload identification (VPID) packet insertion and extraction</td>
</tr>
<tr>
<td></td>
<td>• Clock enable generator</td>
</tr>
<tr>
<td></td>
<td>• Video rate detection</td>
</tr>
<tr>
<td></td>
<td>• Cyclical redundancy check (CRC) encoding and decoding (HD only)</td>
</tr>
<tr>
<td></td>
<td>• Line number (LN) insertion and extraction (HD only)</td>
</tr>
<tr>
<td></td>
<td>• Word scrambling and descrambling</td>
</tr>
<tr>
<td></td>
<td>• Word alignment</td>
</tr>
<tr>
<td></td>
<td>• Transmitter clock multiplexer (optional)</td>
</tr>
<tr>
<td></td>
<td>• Framing and extraction of video timing signals</td>
</tr>
<tr>
<td></td>
<td>• Dual link data stream synchronization (HD only)</td>
</tr>
<tr>
<td></td>
<td>• IPUTF compliance core (hw.tcl)</td>
</tr>
<tr>
<td></td>
<td>• IEEE encryption for functional simulation</td>
</tr>
<tr>
<td></td>
<td>• Transceiver dynamic reconfiguration</td>
</tr>
<tr>
<td></td>
<td>• Dynamic generation of simulation testbench</td>
</tr>
<tr>
<td></td>
<td>• Dynamic generation of design example entity</td>
</tr>
<tr>
<td></td>
<td>• RP168 support for synchronous video switching</td>
</tr>
<tr>
<td></td>
<td>• Tx PLL dynamic switching</td>
</tr>
<tr>
<td></td>
<td>• OpenCore Plus evaluation</td>
</tr>
<tr>
<td><strong>Application</strong></td>
<td>• Digital video equipment</td>
</tr>
<tr>
<td></td>
<td>• Mixing and recording equipment</td>
</tr>
<tr>
<td><strong>Design Tools</strong></td>
<td>• MegaWizard™ Plug-In Manager in the Quartus II software for design creation and compilation</td>
</tr>
<tr>
<td></td>
<td>• ModelSim®-Altera, Riviera-Pro, and VCS/VCS MX software for design simulation or synthesis</td>
</tr>
</tbody>
</table>

### Related Information

- [Altera Software Installation and Licensing](#)
- [Device Family Support](#)
- [What’s New in Altera IP](#)
The SDI II MegaCore function implements a transmitter, receiver, or full-duplex SDI at standard definition (SD), high definition (HD), or 3 gigabits per second (3G) rate as defined by the Society of Motion Picture and Television Engineers (SMPTE). The SDI II MegaCore function supports dual standard (SD-SDI and HD-SDI) and triple standard (SD-SDI, HD-SDI, and 3G-SDI). These modes provide automatic receiver rate detection and transceiver dynamic reconfiguration.

The SDI II MegaCore function highlights the following new features:

- Supports 28 nm devices and beyond.
- Arria V and Stratix V from Quartus II version 12.1 onwards
- Arria V GZ and Cyclone V from Quartus II version 13.0 onwards
- IPUTF compliance core (hw.tcl).
- IEEE encryption for functional simulation.
- Dynamic generation of user simulation testbench that matches the IP configuration.
- Dynamic generation of design example that serves as common entity for simulation and hardware verification.

General Description

The SMPTE defines a SDI standard that is widely used as an interconnect between equipment in video production facilities. The SDI II MegaCore function can handle the following SDI data rates:

- 270 megabits per second (Mbps) SD-SDI, as defined by SMPTE259M-1997 10-Bit 4:2:2 Component Serial Digital Interface
- 1.485 gigabits per second (Gbps) or 1.4835-Gbps HD-SDI, as defined by SMPTE292M-1998 Bit-Serial Digital Interface for High Definition Television Systems
- 2.97-Gbps or 2.967-Gbps 3G SDI, as defined by SMPTE424M
- Dual link HD-SDI, as defined by SMPTE372M-Dual Link 1.5Gb/s Digital Interface for 1920×1080 and 2048×1080 Picture Formats
- Dual standard support for SD-SDI and HD-SDI
- Triple standard support for SD-SDI, HD-SDI, and 3G-SDI
- SMPTE425M level A support (direct source image formatting)
- SMPTE425M level B support (dual link mapping)
- SMPTE RP168 switching support
### Table 2-1: SDI Standard Support

Table below lists the SDI standard support for various FPGA devices.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Arria V</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Arria V GZ</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Stratix V</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Cyclone V</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

### SMPTE372 Dual Link Support

Recording studios support HD 1080p format by using a dual-link connection (SMPTE372) from cameras to the mixing and recording equipment. The SMPTE 372 specification defines a way of interconnecting digital video equipment with a dual link HD-SDI (link A and link B), based upon the SMPTE292 specification data structure. The total data rate of the dual link connection is 2.97 Gbps or 2.97/1.001 Gbps. Level A is defined as a direct source image mapping while level B is defined as 2 x SMPTE292 HD-SDI mapping (including SMPTE372 dual link mapping).

### Level A to Level B Conversion

To interface between a HD dual link receiver and 3G-SDI single link transmitter equipment, perform a level A to level B conversion. This conversion takes either two 1.485 Gbps dual link signals or two separate co-timed HD signals and combines them into a single 3G-SDI stream.

### Figure 2-1: Example of Level A to Level B Conversion

Conversion of Two HD-SDI Data Streams to 3G-SDI Level B Data Streams
Figure 2-2: Implementation of Level A to Level B Conversion

Block diagram of level A to level B conversion.

Figure 2-3: Example of Level B to Level A Conversion

Conversion of 3G-SDI Level B Data To Two HD-SDI Data Streams.

Level B to Level A Conversion

To interface between 3-Gbps single link receiver and HD dual link transmitter equipment, perform a level B to level A conversion. This conversion takes a single 3G-SDI signal and separates the signal into two 1.485 Gbps signals, which can either be a dual link 1080p signal or two separate co-timed HD data streams.
**SMPTE RP168 Switching Support**

The SMPTE RP168 standard defines the requirements for synchronous switching between two video sources to take place with minimal interference to the receiver. The RP168 standard has restrictions for which lines the source switching can occur. The SDI II MegaCore function has flexibility and does not restrict you to switch at only a particular line defined in the RP168 standard. You can perform switching at any time between different video sources as long as the source has similar standard and format. After switching, all the status output signals, including the `rx_trs_locked`, `rx_frame_locked`, and `rx_align_locked` signals, remain unchanged. You should see a zero interrupt at the downstream.

**SD 20-Bits Interface for Dual/Triple Standard**

For a common SD interface, the serial data format is 10-bits wide, whereas for HD or 3G, the data format is 20-bits wide, divided into two parallel 10-bits datastreams (known as Y and C).

To make the interface bit width common for all standards in the dual standard or triple standard SDI mode, the receiver can extract the data and align them in 20-bits width, whereas transmitter can accept SD data in 20-bits width and retransmit them successfully. The timing diagrams below show a comparison of data arrangement between 10-bits and 20-bits interface.
The upper 10-bits of \textit{rx\_dataout} are insignificant data. The lower 10-bits of \textit{rx\_dataout} are Luma (Y) and chroma (Cb, Cr) channels (interleaved). The 1H 4L 1H 5L cadence of \textit{rx\_dataout\_valid} repeats indefinitely (ideal).

**Figure 2-6: SD 20-Bits Interface**

- The upper 10-bits of \textit{rx\_dataout} are Luma (Y) channel and the lower 10-bits are Chroma (Cb, Cr) channel.
- The 1H 4L 1H 5L cadence of \textit{rx\_dataout\_valid} repeats indefinitely (ideal).

**Tx PLL Dynamic Switching**

This feature (available from Quartus II version 13.0SP1 onwards) allows you to dynamically switch between NTSC and PAL transceiver data rates for all video standards that include HD and 3G video standards. This feature allows the SDI video equipment to function regardless of whether the video system is operating on NTSC or PAL. This feature is not required for SD video standard.

To implement this feature, you are required to provide two reference clocks (\textit{xcvr\_refclk} and \textit{xcvr\_refclk\_alt}) to the SDI II MegaCore Function. The frequency of the reference clocks must be assigned to 148.5 MHz and 148.35 MHz in any assignment order.

The Tx PLL select signal (\textit{ch1\_{tx/du}\_tx_pll\_sel}) is an input control signal that you provide to the core and the transceiver reconfiguration controller to select the desired clock input for the hard transceiver.

- Set \textit{ch1\_{tx/du}\_tx_pll\_sel} to 0 to select \textit{xcvr\_refclk}
- Set \textit{ch1\_{tx/du}\_tx_pll\_sel} to 1 to select \textit{xcvr\_refclk\_alt}

To dynamically switch between the two reference clocks, you need to implement a simple handshaking mechanism. The Tx PLL dynamic switching handshake is initiated when the reconfiguration request signal (\textit{ch1\_{tx/du}\_tx\_start\_reconfig}) is asserted high. This signal must remain asserted until the reconfiguration process completes. The reconfiguration process completes when the reconfiguration done signal (\textit{ch1\_{tx/du}\_tx\_reconfig\_done}) is asserted high. The Tx PLL select signal (\textit{ch1\_{tx/du}\_tx_pll\_sel}) needs to be stable throughout the reconfiguration process.

To complete the handshaking process, you must deassert the reconfiguration request signal (\textit{ch1\_{tx/du}\_tx\_start\_reconfig}) upon assertion of the reconfiguration done signal (\textit{ch1\_{tx/du}\_tx\_reconfig\_done}). The Tx PLL dynamic switching only takes effect after the \textit{tx\_rst} is asserted high and deasserted low accordingly.
Figure 2-7: Hardware Implementation of the Dynamic Switching Feature

The table below describes the behavior of the dynamic switching feature when you initiate a handshaking process (with reference to the timing diagram).

**Table 2-3: Dynamic Switching Behavior During a Handshaking Process**

<table>
<thead>
<tr>
<th>Case</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The handshaking process attempts to switch to select $xcvr_{refclk_alt}$. The $tx_{clkout}$ successfully locks to $xcvr_{refclk_alt}$ (148.35 MHz)</td>
</tr>
<tr>
<td>2</td>
<td>The handshaking process attempts to switch to select $xcvr_{refclk}$. The $tx_{clkout}$ successfully locks to $xcvr_{refclk}$ (148.5 MHz)</td>
</tr>
</tbody>
</table>
The handshaking process attempts to switch to select `xcvr_refclk_alt`. The switching fails because the `ch1_{tx/du}_tx_pll_sel` changed from 1 to 0 prior to the assertion of `ch1_{tx/du}_tx_start_reconfig`. Therefore `tx_clkout` is still locked to `xcvr_refclk` (148.5MHz).

### Resource Utilization

The table below lists the typical resource utilization for each functional submodule of the SDI II MegaCore function with the Quartus II software, version 13.0.

**Note:** The resource utilization of the MegaCore function is based on the bidirectional interface settings unless otherwise specified.

#### Table 2-4: Resource Utilization For Each Submodule

<table>
<thead>
<tr>
<th>Submodule</th>
<th>Combinational ALUTs</th>
<th>Logic Registers</th>
<th>Block Memory Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insert Line</td>
<td>11</td>
<td>11</td>
<td>0</td>
</tr>
<tr>
<td>Insert CRC</td>
<td>48</td>
<td>29</td>
<td>0</td>
</tr>
<tr>
<td>Insert VPID</td>
<td>99</td>
<td>78</td>
<td>0</td>
</tr>
<tr>
<td>Tx Match TRS</td>
<td>25</td>
<td>16</td>
<td>0</td>
</tr>
<tr>
<td>Scrambler</td>
<td>55</td>
<td>31</td>
<td>0</td>
</tr>
<tr>
<td>Tx Sample</td>
<td>121</td>
<td>24</td>
<td>0</td>
</tr>
<tr>
<td>Generate Clock Enable</td>
<td>7</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>Rx Sample</td>
<td>259</td>
<td>156</td>
<td>0</td>
</tr>
<tr>
<td>Rx Sample (when <strong>SD Interface Bit Width</strong> = 20)</td>
<td>293</td>
<td>177</td>
<td>0</td>
</tr>
<tr>
<td>Detect Video Standard</td>
<td>85</td>
<td>74</td>
<td>0</td>
</tr>
<tr>
<td>Detect 1 and 1/1.001 Rates</td>
<td>76</td>
<td>71</td>
<td>0</td>
</tr>
<tr>
<td>Transceiver Controller (Custom PHY IP)</td>
<td>222</td>
<td>140</td>
<td>0</td>
</tr>
<tr>
<td>Transceiver Controller (Native PHY IP)</td>
<td>166</td>
<td>82</td>
<td>0</td>
</tr>
<tr>
<td>Descrambler</td>
<td>40</td>
<td>31</td>
<td>0</td>
</tr>
<tr>
<td>TRS Aligner</td>
<td>442</td>
<td>598</td>
<td>0</td>
</tr>
<tr>
<td>TRS Aligner (when <strong>SD Interface Bit Width</strong> = 20)</td>
<td>521</td>
<td>648</td>
<td>0</td>
</tr>
<tr>
<td>Rx Match TRS</td>
<td>20</td>
<td>16</td>
<td>0</td>
</tr>
<tr>
<td>Submodule</td>
<td>Combinational ALUTs</td>
<td>Logic Registers</td>
<td>Block Memory Bits</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>---------------------</td>
<td>-----------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Rx Match TRS (when <strong>SD Interface Bit Width</strong> = 20)</td>
<td>21</td>
<td>16</td>
<td>0</td>
</tr>
<tr>
<td>3Gb Demux</td>
<td>29</td>
<td>94</td>
<td>0</td>
</tr>
<tr>
<td>3Gb Demux (level B to level A conversion)</td>
<td>156</td>
<td>318</td>
<td>10,752</td>
</tr>
<tr>
<td>Extract Line</td>
<td>3</td>
<td>18</td>
<td>0</td>
</tr>
<tr>
<td>Check CRC</td>
<td>48</td>
<td>19</td>
<td>0</td>
</tr>
<tr>
<td>Extract VPID</td>
<td>70</td>
<td>108</td>
<td>0</td>
</tr>
<tr>
<td>Extract VPID (when <strong>SD Interface Bit Width</strong> = 20)</td>
<td>119</td>
<td>118</td>
<td>0</td>
</tr>
<tr>
<td>Detect Format</td>
<td>260</td>
<td>169</td>
<td>0</td>
</tr>
<tr>
<td>Sync Streams</td>
<td>140</td>
<td>186</td>
<td>1,312</td>
</tr>
<tr>
<td>Sync Streams (level A to level B conversion)</td>
<td>212</td>
<td>374</td>
<td>2,336</td>
</tr>
<tr>
<td>Convert SD Bits</td>
<td>8</td>
<td>18</td>
<td>0</td>
</tr>
</tbody>
</table>
Installation and Licensing

To evaluate the SDI II MegaCore function using the OpenCore Plus feature, follow these steps in your design flow:

1. Install the SDI II MegaCore function.
2. Create a custom variation of the SDI II MegaCore function.
3. Implement the rest of your design using the design entry method of your choice.
4. Use the IP functional simulation model to verify the operation of your design.
5. Compile your design in the Quartus II software. You can also generate an OpenCore Plus time-limited programming file, which you can use to verify the operation of your design in hardware.

Note: For more information on IP functional simulation models, refer to the Simulating Altera Designs chapter in volume 3 of the Quartus II Handbook.

The default installation directory for the SDI II MegaCore function on Windows is `c:\altera\<version>`; on Linux, it is `/opt/altera<version>`.

You can obtain a license for the MegaCore function only when you are completely satisfied with its functionality and performance, and want to take your design to production. After you purchase a license for the SDI II MegaCore function, follow these steps:

1. Set up licensing.
2. Generate a programming file for the Altera device or devices on your board.
3. Program the Altera device or devices with the completed design.

Related Information
Simulating Altera Designs

OpenCore Plus Evaluation

The Altera OpenCore evaluation feature allows you to generate RTL files and simulation models to simulate, compile, and validate timing. However, you must acquire a license to generate a programming file to configure an FPGA. Without a license for the variation in your design, you cannot create an SRAM Object File (.sof) or a Programmer Object File (.pof) for programming a device with your design.
With the free OpenCore Plus evaluation feature, you can perform the following actions:

- Simulate the behavior of a megafuntion (Altera MegaCore function or AMPPSM megafuntion) within your system.
- Verify the design functionality and quickly evaluate its size and speed.
- Generate time-limited device programming files for designs that include MegaCore functions.
- Program a device and verify your design in hardware.

**OpenCore Plus Time-Out Behavior**

The OpenCore Plus evaluation can support the following modes of operation:

- **Untethered**—the design runs for a limited time.
- **Tethered**—requires a connection between your board and the host computer. If all megafuntions in a design supports the tethered mode, the device can operate for a longer time or indefinitely.

All megafuntions in a device time out simultaneously when reaching the most restrictive evaluation time. If there is more than one megafuntion in a design, a specific megafuntion’s time-out behavior might be masked by the time-out behavior of the other megafuntions.

**Note:** For MegaCore functions, the untethered time-out is 1 hour; the tethered time-out value is indefinite.

Your design stops working after the hardware evaluation time expires and the \texttt{rst} signal goes high.

**Related Information**

- Altera Software Installation and Licensing
- AN 320: OpenCorePlus Evaluation of Megafuntions

**Design Walkthrough**

This walkthrough explains how to create an SDI II MegaCore function design using the MegaWizard Plug-In Manager and the Quartus II software. After you generate a custom variation of the SDI II MegaCore function, you can incorporate it into your overall project.

This walkthrough includes the following steps:

1. Creating a New Quartus II Project on page 3-2
2. Launching MegaWizard Plug-In Manager on page 3-3
3. Parameterizing the MegaCore Function on page 3-3
4. Generating a Design Example and Simulation Testbench on page 3-3
5. Simulating the SDI II MegaCore Function Design on page 3-4

**Creating a New Quartus II Project**

You need to create a new Quartus II project with the New Project Wizard, which specifies the working directory for the project, assigns the project name, and designates the name of the top-level design entity.

To create a new project, perform the following steps:

1. From the Windows Start menu, select **Programs > Altera > Quartus II <version>** to run the Quartus II software. Alternatively, you can use the Quartus II Web Edition software.
3. In the New Project Wizard: Directory, Name, Top-Level Entity page, specify the working directory, project name, and top-level design entity name. Click Next.
4. In the New Project Wizard: Add Files page, select the existing design files (if any) you want to include in the project. Click Next.
5. In the New Project Wizard: Family & Device Settings page, select the device family and specific device you want to target for compilation. Click Next.
6. In the EDA Tool Settings page, select the EDA tools you want to use with the Quartus II software to develop your project.
7. The last page in the New Project Wizard window shows the summary of your chosen settings. Click Finish to complete the Quartus II project creation.

Launching MegaWizard Plug-In Manager

To launch the MegaWizard Plug-In Manager in the Quartus II software, follow these steps:
1. On the Tools menu, click MegaWizard Plug-In Manager.
2. Select Create a new custom megafonction variation and click Next.
3. Select your target device family from the list.
4. Expand the Interfaces > SDI folder and click SDI II <version>.
5. Select the output file type for your design; the wizard supports VHDL and Verilog HDL.
6. Affix a variation name for the MegaCore function output files <project path>\<variation name>. The MegaWizard Plug-In Manager shows the project path that you specified in the New Project Wizard.
7. Click Next to display the SDI II MegaCore function parameter editor.

Parameterizing the MegaCore Function

To parameterize your MegaCore function, follow these steps:
1. Select the video standard.
2. Select Bidirectional, Transmitter, or Receiver interface direction.
3. Select Combined, Transceiver, or Protocol.
4. Turn on the necessary transceiver options. Some options are grayed out if it is not supported in a selected configuration.
5. Turn on the necessary receiver options.
6. Turn on the necessary transmitter options.
7. Click Finish.

Related Information
SDI II MegaCore Function Parameters on page 3-6

Generating a Design Example and Simulation Testbench

After you have parameterized the MegaCore function, you have the option to generate a design example, in addition to generating the MegaCore component files.

---

2 To include existing files, you must specify the directory path to where you installed the SDI II MegaCore function. You must also add the user libraries if you installed the MegaCore IP Library in a different directory from where you installed the Quartus II software.
To generate a design example, check the **Generate Example Design** option to create the following entities:

- **design example**—serves as a common entity for simulation and hardware verification.
- **simulation testbench**—consists of the design example entity and other non-synthesizable components.

The example testbench and the automated script are located in the 
<variation name>_example/sdi_ii/simulation/verilog or 
<variation name>_example/sdi_ii/simulation/vhdl directory.

**Note:** Generating a design example can increase processing time.

You can now integrate your custom MegaCore function variation into your design, simulate, and compile.

### Simulating the SDI II MegaCore Function Design

After design generation, the files located in the <variation name>_example/sdi_ii/simulation/verilog or <variation name>_example/sdi_ii/simulation/vhdl directory are available for you to simulate your design.

The SDI II MegaCore function supports the following EDA simulators listed in the table below.

#### Table 3-2: Supported EDA Simulators

<table>
<thead>
<tr>
<th>Simulator</th>
<th>Supported Platform</th>
<th>Supported Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>ModelSim-Altera SE/AE</td>
<td>Windows/Linux</td>
<td>VHDL and Verilog HDL</td>
</tr>
<tr>
<td>VCS/VCS MX</td>
<td>Windows/Linux</td>
<td>Verilog</td>
</tr>
<tr>
<td>Aldec Riviera-PRO</td>
<td>Linux</td>
<td>Verilog</td>
</tr>
</tbody>
</table>

To simulate the design using the ModelSim®-Altera simulator, follow these steps:

1. Start the ModelSim-Altera simulator.
2. On the File menu, click **Change Directory > Select**
   <variation name>_example_design/sdi_ii/simulation/verilog (for Verilog HDL language) or
   _example_design/sdi_ii/simulation/vhdl (for VHDL language).
3. On the File menu, click **Load > Macro file**. Select **run_sim.tcl**. This file compiles the design and runs the simulation automatically, providing a pass/fail indication on completion.

To simulate the design using the VCS/VCS MX simulator (in Linux), follow these steps:

1. Start the VCS/VCS MX simulator.
2. On the File menu, click **Change Directory > Select**
   <variation name>_example_design/sdi_ii/simulation/verilog.
3. Run **source run_vcs.sh** (for VCS) or **source run_vcsmx.sh** (for VCSMX). This file compiles the design and runs the simulation automatically, providing a pass/fail indication on completion.

To simulate the design using the Aldec Riviera-PRO simulator, follow these steps:

1. Start the Aldec Riviera-PRO simulator.
2. On the File menu, click **Change Directory > Select**
   <variation name>_example_design/sdi_ii/simulation/verilog.
3. On the Tools menu, click **Execute Macro**. Select **run_riviera.tcl**. This file compiles the design and runs the simulation automatically, providing a pass/fail indication on completion.
Compiling the SDI II MegaCore Function Design

To compile your design, click **Start Compilation** on the Processing menu in the Quartus II software. You can use the generated .qip file to include relevant files into your project.

You can find the design examples of the SDI II MegaCore function in the `<variation name>_example_design/sdi_ii/example_design/sdi_ii_0001_example_design` directory. For the design example illustrations, refer to the **Design Examples** section.

**Note:** To create a new project using the generated design example, follow the steps in the *Creating a New Quartus II Project* section and add the design example .qip file in **step 4**.

**Related Information**
- **Quartus II Help**
  More information about compilation in Quartus II software.
- **Creating a New Quartus II Project** on page 3-2
- **Design Examples** on page 3-10

Programming an FPGA Device

After successfully compiling your design, program the targeted Altera device with the Quartus II Programmer and verify the design in hardware.

For instructions on programming the FPGA device, refer to the **Device Programming** section in volume 3 of the Quartus II Handbook.

**Related Information**
- **Device Programming**

Design Reference

This section describes the SDI II MegaCore function parameters, signals, and files to help you configure your design. This section includes detailed description about the SDI II MegaCore function design examples.
## SDI II MegaCore Function Parameters

### Table 3-3: SDI II MegaCore Function Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Video Standard</strong></td>
<td>SD SDI, HD SDI, 3G SDI, HD SDI Dual Link, Dual Standard, Triple Standard</td>
<td>Sets the video standard.</td>
</tr>
</tbody>
</table>
|                            |                                                                       | - SD-SDI—disables option for line insertion and extraction, and CRC generation and extraction  
                          |                            |                                                                       | - HD-SDI—enables option for in line insertion and extraction and CRC generation and extraction  
                          |                            |                                                                       | - Dual or triple standard SDI—includes the processing blocks for both SD-SDI and HD-SDI standards. Logics for bypass paths and to automatically switch between the input standards are included. |
| **Configuration Options**  |                                                                       |                                                                                                                                                                                                          |
| SD Interface Bit Width     | 10, 20                                                                | Selects the SD interface bit width. Only applicable for dual standard and triple standard.                                                                                                                 |
| **Direction**              | Bidirectional, Receiver, Transmitter                                   | Sets the port direction. The selection enables or disables the receiver and transmitter supporting logic appropriately.                                                                                       |
|                            |                                                                       | - Bidirectional—instantiates the transceiver in duplex mode.                                                                                                                                             |
|                            |                                                                       | - Receiver and transmitter—instantiates the transceiver in a separate receiver and transmitter mode respectively.                                                                                       |
| Transceiver and/or Protocol| Combined, Transceiver, Protocol                                       | Selects the components.                                                                                                                                                                                     |
|                            |                                                                       | - Transceiver—includes tx/rx_phy_mgmt/phy_adapter and hard transceiver. This option is useful if you want to use the same transceiver component to support both SDI and ASI.  |
|                            |                                                                       | - Protocol                                                                                                                                                                                                |
| **Transceiver Option**     | Tx PLL Dynamic Switching                                              |                                                                                                                                                                                                          |
|                            | On, Off                                                               | - On: Enable dynamic switching  
                          |                            |                                                                       | - Off: Disable dynamic switching                                                                                                     |
|                            |                                                                       | Turn on this option to instantiate two Tx PLLs within the transceiver and allow dynamic switching between 1 and 1/1.001 data rates.                                                                         |
### Receiver Options

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
</table>
| Increase error tolerance level                      | On, Off   | • On: Error tolerance level = 15  
• Off: Error tolerance level = 4  
Turn on this option to increase the error tolerance level for consecutive missed end of active videos (EAVs), start of active videos (SAVs), or erroneous frames. |
| CRC error output                                    | On, Off   | • On: CRC monitoring (Not applicable for SD-SDI mode)  
• Off: No CRC monitoring (saves logic) |
| Extract Video Payload ID (SMPTE 352M)               | On, Off   | • On: Extract VPID  
• Off: No VPID extraction (saves logic)  
It is compulsory to turn on this option for 3G-SDI, HD SDI dual link, and triple standard modes. The extracted VPID is required for consistent detection of the 1080p format.  
It is compulsory to turn on this option for the design example demonstration when you enable either the Convert level A to level B (SMPTE 372M) or Convert level B to level A (SMPTE 372M) option. |
| Convert level A to level B (SMPTE 372M)             | On, Off   | • On: Convert level A (direct image format mapping) to level B (2 x SMPTE 292M HD-SDI mapping, including SMPTE 372M dual link mapping) for HD dual link receiver output.  
• Off: No conversion  
This option is only available for HD-SDI dual link receiver. |
| Convert level B to level A (SMPTE 372M)             | On, Off   | • On: Convert level B (2 x SMPTE 292M HD-SDI mapping including SMPTE 372M dual link mapping) to level A (direct image format mapping) for 3G-SDI or triple standard SDI receiver output.  
• Off: No conversion  
This option is only available for 3G-SDI and triple standard SDI receiver. |
| Two times oversample mode for HD-SDI                | On, Off   | • On: Run the hard transceiver at twice the rate with improved jitter performance. This requires a 148.5-MHz or 148.35-MHz transceiver reference clock (xcvr_refclk).  
• Off: No two times oversampling  
This option is only available for HD-SDI standard. |
| Insert Video Payload ID (SMPTE 352M)                | On, Off   | • On: Insert VPID  
• Off: No VPID insertion (saves logic) |

### Transmitter Options

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
</table>
| Insert Video Payload ID (SMPTE 352M)                | On, Off   | • On: Insert VPID  
• Off: No VPID insertion (saves logic) |
SDI II MegaCore Function Component Files

Table 3-4: Generated Files

Table below describes the generated files and other files that might be in your project directory. The names and types of files in the MegaWizard Plug-In Manager report vary depending on whether you create your design with VHDL or Verilog HDL.

<table>
<thead>
<tr>
<th>Extension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;variation name&gt;.v or .sv</td>
<td>A MegaCore function variation file, which defines a Verilog HDL description of the custom MegaCore function. Instantiate the entity defined by this file inside your design. Include this file when compiling your design in the Quartus II software.</td>
</tr>
<tr>
<td>&lt;variation name&gt;.sdc</td>
<td>Contains timing constraints for your SDI variation.</td>
</tr>
<tr>
<td>&lt;variation name&gt;.qip</td>
<td>Contains Quartus II project information for your MegaCore function variations.</td>
</tr>
<tr>
<td>&lt;variation name&gt;.tcl</td>
<td>Tcl script file to run in Quartus II software.</td>
</tr>
</tbody>
</table>

Video Pattern Generator Signals

Table 3-5: Video Pattern Generator Top Level Signals

Table below lists the input signals for the video pattern generator. The listed signals are exported at the top level of the design example. Other signals—that are not exported—connects within the design example entity.

<table>
<thead>
<tr>
<th>Signal</th>
<th>Width</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pattgen_tx_std</td>
<td>2</td>
<td>Input</td>
<td>Transmit video standard.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 00: SD-SDI</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 01: HD-SDI or HD-SDI dual link</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 10: 3G-SDI level B</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 11: 3G-SDI level A</td>
</tr>
<tr>
<td>pattgen_tx_format</td>
<td>4</td>
<td>Input</td>
<td>Transmit video format.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 0000: SMPTE259M 525i</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 0001: SMPTE259M 625i</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 0100: SMPTE274M 1080i60</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 0101: SMPTE274M 1080i50</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 0110: SMPTE274M 1080p24</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 0111: SMPTE296M 720p60</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 1000: SMPTE296M 720p50</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 1001: SMPTE296M 720p30</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 1010: SMPTE296M 720p25</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 1011: SMPTE296M 720p24</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 1100: SMPTE274M 1080p30</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 1101: SMPTE274M 1080p25</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Others: Reserved for future use</td>
</tr>
</tbody>
</table>
### Transceiver Reconfiguration Controller Signals

**Table 3-6: Transceiver Reconfiguration Controller Top Level Signals**

Table below lists the input signals for the transceiver reconfiguration controller. The listed signals are exported at the top level of the design example. Other signals—that are not exported—connects within the design example entity.

<table>
<thead>
<tr>
<th>Signal</th>
<th>Width</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pattgen_dl_mapping</td>
<td>1</td>
<td>Input</td>
<td>Dual link mapping. Set to 1 for HD-SDI dual link and 3Gb transmit video standard only.</td>
</tr>
</tbody>
</table>
| pattgen_ntsc_paln     | 1     | Input     | Transmit rate.  
  - 0: PAL (1) rate. For example, 1080p30  
  - 1: NTSC (1/1.001) rate. For example, 1080p29.97.  
  This input ignores all SD video formats (525i, 625i) and certain HD video formats that do not support NTSC rate (1080i50, 720p50, 720p25, 1080p25). |
| pattgen_bar_100_75n   | 1     | Input     | Generate color bars.  
  - 0: 75% color bars  
  - 1: 100% color bars |
| pattgen_patho         | 1     | Input     | Set to 1 to generate pathological pattern.                                                                                                  |
| pattgen_blank         | 1     | Input     | Set to 1 to generate black signal.                                                                                                         |
| pattgen_no_color      | 1     | Input     | Set to 1 to generate bars with no color.                                                                                                   |

**Related Information**

*Altera Transceiver PHY IP Core User Guide*

More information about the transceiver reconfiguration controller frequency range.
Reconfiguration Router Signals

Table 3-7: Reconfiguration Router Top Level Signals

Table below lists the signals for the reconfiguration router. The listed signals are exported at the top level of the design example. Other signals—that are not exported—connects within the design example entity.

<table>
<thead>
<tr>
<th>Signal</th>
<th>Width</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ch1_&lt;direction&gt;_tx_start_reconfig</td>
<td>1</td>
<td>Input</td>
<td>Dynamic reconfiguration request signal for Tx PLL dynamic switching at transmitter or duplex instance at channel 1. This signal is only available if you turn on the Tx PLL Dynamic Switching option. Refer to the Tx PLL Dynamic Switching for the usage requirement.</td>
</tr>
<tr>
<td>ch1_&lt;direction&gt;_tx_pll_sel</td>
<td>1</td>
<td>Input</td>
<td>Tx PLL select signal for Tx PLL dynamic switching at transmitter or duplex instance at channel 1. This signal is also connected to xcvr_refclk_sel signal of the SDI instance. This signal is only available if you turn on the Tx PLL Dynamic Switching option. Refer to the Tx PLL Dynamic Switching for the usage requirement.</td>
</tr>
<tr>
<td>ch1_&lt;direction&gt;_tx_reconfig_done</td>
<td>1</td>
<td>Output</td>
<td>Dynamic reconfiguration acknowledge signal for Tx PLL dynamic switching at transmitter or duplex instance at channel 1. This signal is only available if you turn on the Tx PLL Dynamic Switching option.</td>
</tr>
</tbody>
</table>

Design Examples

Figure below illustrates the generated design example entity and simulation testbench. The design example entity is synthesizable and consists of a video pattern generator, transceiver reconfiguration controller, reconfiguration management, loopback path, and various SDI blocks occupying two transceiver channels. One of the SDI block is the device under test (DUT), which is configured according to your parameterization. The rest of the SDI blocks are test instances for a complete serial loopback simulation. For example, if you choose to generate SDI RX, the software instantiates an SDI TX block to serve as a video source. The loopback block (SDI duplex) is always instantiated in the design example for parallel loopback demonstration.
Figure 3-1: Design Example Entity and Simulation Testbench

Testbench

Design Example

Loopback Path

Ch0 Loopback (SDI Duplex)

Transceiver Reconfiguration Controller

Reconfiguration Management/Router

Ch1 Test (SDI RX)

Video Pattern Generator

Ch1 DUT (SDI TX)

TX Checker

RX Checker

Test Control

Data

Control

SDI II MegaCore Function
Figure 3-2: Design Example Entity and Simulation Testbench for Level A to Level B Conversion

Figure below illustrates the generated design example entity and simulation testbench if you choose to generate HD SDI dual link receiver with level A to level B conversion option enabled.
Design Example Components

The design examples for the SDI II MegaCore function consist of the following components:

- Video pattern generator
- Transceiver reconfiguration controller
- Reconfiguration management
- Reconfiguration router

Video Pattern Generator

The video pattern generator generates a colorbar or pathological pattern. The colorbar is preferable for image generation while the pathological pattern can stress the PLL and cable equalizer of the attached video equipment. You can configure the video pattern generator to generate various video formats.
Table 3-8: Configuring the Video Pattern Generator to Generate Different Video Formats

Table below lists the examples of how to configure the video pattern generator signals to generate a video format that you desire.

<table>
<thead>
<tr>
<th>Example</th>
<th>Video Format</th>
<th>Interface</th>
<th>Signal</th>
<th>pattgen_tx_std</th>
<th>pattgen_tx_format</th>
<th>pattgen_dl_mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example 1: Generate 1080i video format</td>
<td>1080i60</td>
<td>HD-SDI</td>
<td>2'b01</td>
<td>4'b0100</td>
<td>1'b0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1080i60x2</td>
<td>HD-SDI dual link</td>
<td>2'b01</td>
<td>4'b0100</td>
<td>1'b0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3Gb</td>
<td>2'b10</td>
<td>4'b0100</td>
<td>1'b0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Example 2: Generate 1080p video format</td>
<td>1080p30</td>
<td>HD-SDI</td>
<td>2'b01</td>
<td>4'b1100</td>
<td>1'b0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1080p30x2</td>
<td>HD-SDI dual link</td>
<td>2'b01</td>
<td>4'b1100</td>
<td>1'b0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1080p60</td>
<td>HD-SDI dual link</td>
<td>2'b01</td>
<td>4'b1100</td>
<td>1'b1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3Ga</td>
<td>2'b11</td>
<td>4'b1100</td>
<td>1'b0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3Gb</td>
<td>2'b10</td>
<td>4'b1100</td>
<td>1'b1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Related Information

**Video Pattern Generator Signals** on page 3-8

**Transceiver Reconfiguration Controller**

The transceiver reconfiguration controller allows you to change the device transceiver settings at any time. Any portion of the transceiver can be selectively reconfigured. Each portion of the reconfiguration requires a read-modify-write operation (read first, then write), in such a way by modifying only the appropriate bits in a register and not changing other bits. Prior to this operation, you must define the logical channel number and the streamer module mode.

You can perform a transceiver dynamic reconfiguration in these two modes:

- streamer module mode 1 (manual mode)—execute a series of Avalon® Memory-Mapped (Avalon-MM) write operation to change the transceiver settings. In this mode, you can execute a write operation directly from the reconfiguration management/router interface to the device transceiver registers.
- streamer module mode 0—use the .miif files to change the transceiver settings.

For read operation, after defining the logical channel number and the streamer module mode, the following sequence of events occur:

1. Define the transceiver register offset in the offset register.
2. Read the data register. Toggle the read process by setting bit 1 of the control and status register (CSR) to logic 1.
3. Once the busy bit in the CSR is cleared to logic 0, it indicates that the read operation is complete and the required data should be available for reading.
For write operation, after setting the logical channel number and the streamer module mode, the following sequence of events occur:

1. Define the transceiver register offset (in which the data will be written to) in the offset register.
2. Write the data to the data register. Toggle the write process by setting bit 0 of the CSR to logic 1.
3. Once the busy bit in the CSR is cleared to logic 0, it indicates that the transceiver register offset modification is successful.

Related Information

- **Altera Transceiver PHY IP Core User Guide**
  More information about the transceiver reconfiguration controller streamer module.

- **Transceiver Reconfiguration Controller Signals** on page 3-9

**Reconfiguration Management**

The reconfiguration management block contains the reconfiguration user logic (a finite state machine) to determine the bits that need to be modified and selects the correct data to be written to the appropriate transceiver register. It also provides handshaking between the SDI receiver and the transceiver reconfiguration controller.

During the reconfiguration process, the logic first reads the data from the transceiver register that needs to be reconfigured and stores the data temporarily in a local register. Then, the logic overwrites only the appropriate bits of the data with predefined values and write the modified data to the transceiver register. Since only one transceiver register can be accessed at a time, the whole process repeats when reconfiguring other registers.

For multiple SDI channels reconfiguration, the logical channel number needs to be set appropriately for each channel and reconfiguration interface. For example, in the design example and simulation testbench figure, there are one SDI duplex, one SDI RX, and one SDI TX block. The number of reconfiguration interface for SDI duplex is 2 (one for channel and one for Tx PLL), for SDI RX is 1 (for channel), for SDI TX is 2 (one for channel and one for Tx PLL). The total number of reconfiguration interface required in the transceiver reconfiguration controller is 5. The table below lists the channel and transceiver reconfiguration controller interface numbers.

<table>
<thead>
<tr>
<th>SDI Block</th>
<th>SDI Channel Number</th>
<th>Transceiver Reconfiguration Controller Interface Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDI Duplex</td>
<td>0</td>
<td>0 and 1</td>
</tr>
<tr>
<td>SDI RX</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>SDI TX</td>
<td>1</td>
<td>3 and 4</td>
</tr>
</tbody>
</table>

**Reconfiguration Router**

The reconfiguration router connects multiple SDI instances to the reconfiguration management and transceiver reconfiguration controller blocks. The reconfiguration router receives all the interface signals...
between the transceiver reconfiguration controller and reconfiguration management, as well as SDI instances, and transmits the signals to their respective destinations.

The reconfiguration router converts reconfiguration related interface signals of multiple SDI instances and user interface to a single-wide data bus for the reconfiguration management and transceiver reconfiguration controller blocks. You can bypass this component if you want to implement designs that expands to more channels. The details are described in the *Expanding to Multiple Channels* section.

**Related Information**

- [Reconfiguration Router Signals](#) on page 3-10
- [Expanding to Multiple Channels](#) on page 3-17

### Avalon-MM Translators

The Avalon-MM Master Translator and Avalon-MM Slave Translator are Avalon-MM interface blocks that access the Transceiver Reconfiguration Controller registers. The translators are not SDI-specific and are automatically instantiated when the core interfaces with an Avalon-MM master or slave component.

For more information about the Avalon-MM Translator functions, refer to the Qsys Interconnect chapter in volume 1 of the *Quartus II Handbook*.

**Related Information**

- [Qsys Interconnect](#) More information about the Avalon-MM Translator functions.
- [Qsys System Design Components](#) More information about the Avalon interface.
- [Avalon Interface Specifications](#)

### Transceiver Dynamic Reconfiguration

The dual and triple standard SDI receivers (or receivers of duplex SDIs) require the transceiver dynamic reconfiguration feature to perform auto detection and locking to different SDI video rates. Transceiver dynamic reconfiguration reconfigures the transceivers to support the three SDI video standards (SD, HD and 3G).

**Note:** You need to perform transceiver dynamic reconfiguration on the SDI transmitter (or transmitters of duplex SDIs) if you want to use the Tx PLL dynamic switching feature.

The dual and triple standard SDI use 11 times oversampling for receiving SD-SDI standard. As the rates for 3G-SDI and SD-SDI are the same, transceiver reconfiguration is not required when the video standard changes from 3G-SDI to SD-SDI and vice versa. Transceiver reconfiguration is only required when the video standard changes from 3G-SDI or SD-SDI to HD-SDI, and vice versa.

To perform transceiver dynamic reconfiguration, you need the transceiver reconfiguration controller and reconfiguration management/router blocks.
The following sequence of events occur when there is a change in the SDI receiver video standard:

1. The SDI receiver locks to 3G-SDI standard and detects the incoming video standard has changed from 3G-SDI to HD-SDI. The transceiver controller requests a reconfiguration.
2. The transceiver reconfiguration controller determines the appropriate settings to write based on the information from the transceiver controller. Then, it performs the read-modify-write operation to reconfigure the device transceiver.
3. Once the reconfiguration completes, the transceiver reconfiguration controller indicates to the SDI receiver that reconfiguration is complete.
4. The SDI receiver begins the process of locking to the incoming data.

Expanding to Multiple Channels

The generated design example consists of two SDI channels, where the SDI duplex instance always occupy Channel 0 (Ch0), while the SDI instance at Channel 1 (Ch1) depends on your selection from the parameter editor. To expand and accommodate more channels, you must perform some modifications to the source files.

For example, when Ch0 is duplex, Ch1 is RX and TX, if you want to instantiate an additional SDI duplex instance at Channel 2 (Ch2), you need to make some modifications to the following components.

Modifying the Transceiver Reconfiguration Controller

Perform the following changes to modify the transceiver reconfiguration controller:

- Edit the `Number_of_reconfig_interfaces` parameter. This parameter specifies the total number of reconfiguration interfaces that connect to this block.
- Each channel or Tx PLL needs one interface. Therefore, a SDI duplex or SDI TX mode requires two interfaces while a SDI RX mode requires only one interface. If the `Tx PLL Dynamic Switching` option is enabled, the SDI duplex or SDI TX mode requires three interfaces. The additional interface is for the additional Tx PLL. For simplicity, assume this option is disabled.
- Determine the total number of reconfiguration interfaces required in your design and assign the parameter value accordingly. In this design example, the total number of reconfiguration interfaces is 7 (Ch0=2, Ch1=3 and Ch2=2).
- Link the `reconfig_to_xcvr` and `reconfig_from_xcvr` signals from the additional SDI duplex instance at Ch2. You must link the signals in the order of the logical channel number (`rx_log_ch_num` and `tx_log_ch_num`) in the reconfiguration logic source file (`sdi_ii_reconfig_logic.v`).
- In the design example that instantiates the transceiver reconfiguration controller, add the wire connection between the additional SDI duplex instance at Ch2 and the transceiver reconfiguration controller as shown below:

```vhdl
wire [139:0] reconfig_to_xcvr_du_ch2;
wire [ 91:0] reconfig_from_xcvr_du_ch2;
wire [139:0] reconfig_to_xcvr_tx_ch1;
wire [ 69:0] reconfig_to_xcvr_rx_ch1;
wire [ 91:0] reconfig_from_xcvr_tx_ch1;
wire [ 45:0] reconfig_from_xcvr_rx_ch1;
wire [139:0] reconfig_to_xcvr_du_ch0;
wire [ 91:0] reconfig_from_xcvr_du_ch0;
```

```vhdl
alt_xcvr_reconfig #(
  .number_of_reconfig_interfaces (7),
  ...
```
Modifying the Reconfiguration Management

Perform the following changes to modify the reconfiguration management:

- Edit the **Number_of_channels** parameter. This parameter value should be the total number of the SDI RX channels declared in the design. In this example, the NUM_CHS is 3.
- Link the interface signals—in particular, `sdi_rx_start_reconfig`, `sdi_rx_reconfig_done`, and `sdi_rx_std`—between multiple SDI instances and reconfiguration management block. Link the interface signals—in particular, `sdi_tx_start_reconfig`, `sdi_tx_reconfig_done`, and `sdi tx_pll_sel`—between user and reconfiguration management block. You must link the signals in the order of the logical channel number (`rx_log_ch_num` and `tx_log_ch_num`) in the reconfiguration logic source file (`sdi_ii_reconfig_logic.v`). For example:

```vhdl
wire tx_start_reconfig_ch2, tx_start_reconfig_ch1, tx_start_reconfig_ch0;
wire tx_pll_sel_ch2, tx_pll_sel_ch1, tx_pll_sel_ch0;
wire tx_reconfig_done_ch2, tx_reconfig_done_ch1, tx_reconfig_done_ch0;
wire rx_start_reconfig_ch2, rx_start_reconfig_ch1, rx_start_reconfig_ch0;
wire [1:0] rx_std_ch2, rx_std_ch1, rx_std_ch0;
wire rx_reconfig_done_ch2, rx_reconfig_done_ch1, rx_reconfig_done_ch0;

sdi_ii_ed_reconfig_mgmt #( .NUM_CHS (3),
) u_reconfig_mgmt (.
  .sdi tx_start_reconfig (tx_start_reconfig_ch2,
    tx_start_reconfig_ch1, tx_start_reconfig_ch0),
  .sdi tx_pll_sel (tx_pll_sel_ch2, tx_pll_sel_ch1, tx_pll_sel_ch0),
  .sdi tx_reconfig_done (tx_reconfig_done_ch2,
    tx_reconfig_done_ch1, tx_reconfig_done_ch0),
  .sdi rx_start_reconfig (rx_start_reconfig_ch2,
    rx_start_reconfig_ch1, rx_start_reconfig_ch0),
  .sdi rx_std (rx_std_ch2, rx_std_ch1, rx_std_ch0),
  .sdi rx_reconfig_done (rx_reconfig_done_ch2,
    rx_reconfig_done_ch1, rx_reconfig_done_ch0)
);
```

- In the reconfiguration logic source file, the default setting for the wire `rx_log_ch_num` is 0 and 2 for channel 0 and channel 1, respectively. The default setting for the wire `tx_log_ch_num` is 0 and 2 (duplex) or 3 (Tx) for channel 0 and channel 1, respectively. These numbers are referring to the **Number_of_channels** parameter value that was set in transceiver reconfiguration controller. Hence, the logical channel number for each SDI channel is as listed in the table below.
<table>
<thead>
<tr>
<th>SDI Channel</th>
<th>Direction</th>
<th>Number of Reconfiguration Interfaces</th>
<th>Logical Channel Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Duplex</td>
<td>2</td>
<td>0: Rx/Tx channel</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1: Tx PLL</td>
</tr>
<tr>
<td>1</td>
<td>Rx and Tx</td>
<td>3 (1 for Rx and 2 for Tx)</td>
<td>2: Rx channel</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3: Tx channel</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4: Tx PLL</td>
</tr>
<tr>
<td>2</td>
<td>Duplex</td>
<td>2</td>
<td>5: Rx/Tx channel</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>6: Tx PLL</td>
</tr>
</tbody>
</table>

- Edit the reconfiguration logic source file to assign the logical channel number for the additional SDI duplex instance, which occupies the SDI Ch2. The logical channel number specified in the source file is the reconfiguration interface that is intended for dynamic reconfiguration. For example, if Tx channel is intended for dynamic reconfiguration, the tx_log_ch_num[2] should be 5.

```vhd
wire [7:0] rx_log_ch_num [0:NUM_CHS-1];
assign rx_log_ch_num[0] = 8'd0; // Duplex Rx channel share same logical channel number with Tx
assign rx_log_ch_num[1] = 8'd2; // Rx channel
assign rx_log_ch_num[2] = 8'd5; // Duplex Rx channel

wire [7:0] tx_log_ch_num [0:NUM_CHS-1];
assign tx_log_ch_num[0] = 8'd0; // Duplex Tx channel share same logical channel number with Rx
assign tx_log_ch_num[1] = 8'd3; // Tx channel
assign tx_log_ch_num[2] = 8'd5; // Duplex Tx channel
```

**Related Information**

*Altera Transceiver PHY IP Core User Guide*

More information about the transceiver reconfiguration controller logical channel numbering.

**Modifying the Reconfiguration Router**

For ease of implementation, you can bypass this block by connecting the interface signals—reconfig_to_xcvr, reconfig_from_xcvr, sdi_rx_start_reconfig, sdi_rx_reconfig_done, sdi_rx_std, sdi_tx_start_reconfig, sdi_tx_reconfig_done, and sdi_tx_pll_sel—directly between the SDI instance and the transceiver reconfiguration controller or the reconfiguration management.
The SDI II MegaCore function implements a transmitter, receiver, or full-duplex interface. The SDI II MegaCore function consists of the following components:

- Protocol block—transmitter or receiver
- Transceiver blocks—PHY management & adapter and hard transceiver

In the MegaWizard Plug-In Manager, you can specify either protocol, transceiver, or combined blocks for your design. For example, if you have multiple protocol blocks in a design, you can multiplex them into one transceiver.

The modular hierarchy design allows you to remove or reuse each submodule within the components across different video standards. The transmitter and receiver data paths are independent from each other.

**Figure 4-1: SDI II MegaCore Function Block Diagram**

---

**Protocol**

The protocol block handles the SDI-specific parts of the core and generally operates on a parallel domain data.
Transmitter

The transmitter performs the following functions:

- HD-SDI LN insertion
- HD-SDI CRC generation and insertion
- VPID (video payload identification) insertion
- Matching timing reference signal (TRS) word
- Clock enable signal generation
- Scrambling and non-return-zero inverted (NRZI) coding

The block diagrams below illustrate the SDI II MegaCore function transmitter (simplex) data path for each supported video standard.

For more information about the function of each submodules, refer to the Submodules section.

Figure 4-2: SD-SDI Transmitter Data Path Block Diagram

Figure 4-3: HD/3G-SDI Transmitter Data Path Block Diagram
Figure 4-4: Dual Rate SDI Transmitter Data Path Block Diagram

Figure 4-5: Triple Rate SDI Transmitter Data Path Block Diagram
## Receiver

The receiver performs the following functions:

- Video standard detection
- Video rate detection
- NRZI decoding and descrambling
- Word alignment
- Demultiplex data links
- Video timing flags extraction
- HD-SDI LN extraction
- HD-SDI CRC
- VPID extraction
- Synchronizing data streams
- Accessing transceiver
- Identifying and tracking of ancillary data

The block diagrams below illustrate the SDI II MegaCore function receiver (simplex) data path for each supported video standard.
Figure 4-7: SD-SDI Receiver Data Path Block Diagram

Figure 4-8: HD-SDI Receiver Data Path Block Diagram
Figure 4-9: 3G-SDI Receiver Data Path Block Diagram

Figure 4-10: Dual Rate SDI Receiver Data Path Block Diagram
Figure 4-11: Triple Rate SDI Receiver Data Path Block Diagram

Figure 4-12: Dual Link HD-SDI Receiver Data Path Block Diagram
For bidirectional or duplex mode, the protocol and PHY management & adapter blocks remain the same for each direction, except the hard transceiver, which is configured in duplex mode. The figure below illustrates the data path of a SD-SDI duplex mode.

**Figure 4-13: SD-SDI Duplex Mode Block Diagram**

---

**Transceiver**

The transceiver block consists of two components—PHY management & adapter and hard transceiver. These two components handle the serial transport aspects of the SDI.

From Quartus II version 13.0SP1 onwards, the hard transceiver utilizes the Altera Native PHY IP Core for all the supported devices:

- Arria V ([altera_xcvr_native_av_hw.tcl](#))
- Stratix V ([altera_xcvr_native_sv_hw.tcl](#))
- Cyclone V ([altera_xcvr_native_cv_hw.tcl](#))

The SDI II MegaCore function instantiates the PHY IP core using the Tcl file associated with each device.

**Note:** In Quartus II 13.0, the hard transceiver utilizes the Altera Custom PHY IP Core ([altera_xcvr_custom_hw.tcl](#)).

The block diagram below illustrate the Native PHY IP core setup in the SDI II MegaCore function (duplex) data path.
The Altera Native PHY IP Core does not include an embedded reset controller and an Avalon-MM interface. This PHY IP core exposes all signals directly as ports. To implement reset functionality for a new IP core, the transceiver reset controller is required to handle all the transceiver reset sequencing. The transceiver reset controller controls the embedded reset controller and also manages additional control options such as automatic or manual reset recovery mode.

Related Information

Altera Transceiver PHY IP Core User Guide
More information about the Altera Native PHY IP Core.

Submodules

You can reuse the submodules in the protocol and transceiver components across different video standard. The SDI II MegaCore function consists of the following submodules:

- Insert Line
- Insert/Check CRC
- Insert VPID
- Match TRS
- Scrambler
- Tx Oversample
- Rx Oversample
Insert Line

The HD-SDI has the option to include line numbers along with the video data. This information is in the end of active video (EAV) extension words of the data stream as defined in the SMPTE292M specification. The line number is 11 bits wide and spreads over two SDI words to utilize the SDI legal data space.

This submodule takes the 11 bits line number data value, correctly encodes them, and inserts them into the 10-bit stream. The line number value is user-defined. The top level port signal is $tx_{ln}[10:0]$ and $tx_{ln\_b}[10:0]$ for link B in 3G and HD dual link modes. You also have the option to enable or disable this feature using the $tx_{enable\_ln}$ signal on the top level port. The same line number value is inserted into both video channels. Two of these submodules are required for Y and C channels.

The Match TRS submodule indicates to this submodule when to insert the values into the stream.

Figure 4-16: Line Number Insertion and Signal Requirements

Figure below illustrates the line number insertion and signal requirements. For a correct line insertion, assert the $tx_{trs}$ signal for the first word of both EAV and start of active video (SAV) TRS.

Insert/Check CRC

The HD-SDI can optionally include a line-based CRC code, which makes up two of the EAV extension words as defined in the SMPTE292M specification.

This submodule calculates the CRC based on the LFSR approach in the SMPTE specification. Note that you can configure this submodule to either insert or check the CRC.

For the transmitter, the core formats and inserts the CRC into two CRC EAV extension words—CRC0 and CRC1. The Match TRS submodule indicates to this submodule when to calculate, reset, and insert the CRC into the stream. For correct CRC generation and insertion, assert the $tx_{trs}$ signal for the first word of
both EAV and SAV TRS as shown in the Insert Line timing diagram. Perform CRC insertion only when the top level port, tx_enable_crc, is set to logic 1.

For the receiver, the core checks the CRC against the value of CRC0 and CRC1 that appear in the incoming stream. If there is a mismatch between the locally calculated value and the value in the stream, this submodule indicates an error. The Match TRS submodule indicates when the CRC0 and CRC1 words are present in the incoming data stream.

**Insert VPID**

The SMPTE352M specification defines an ancillary packet type that provides specific information about the video payload carried by a digital interface. These VPID packets carry information such as the interface type, sampling structure, component bit depth, and picture update rate. Recent SMPTE interfaces such as dual link HD-SDI and 3G-SDI require the VPID packets because it is very difficult to properly interpret the video data without the packet information from the VPID packets.

The VPID packet must be on specific video lines locations at the beginning of the horizontal ancillary (HANC) space, which is right after the EAV, or CRC words that follow the EAV on the interfaces using CRC words.

**Table 4-1: VPID Packet Location**

<table>
<thead>
<tr>
<th>Video Format</th>
<th>Field</th>
<th>Line Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>525i</td>
<td>1</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>276</td>
</tr>
<tr>
<td>625i</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>322</td>
</tr>
<tr>
<td>1080i</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>572</td>
</tr>
<tr>
<td>525p</td>
<td>—</td>
<td>13</td>
</tr>
<tr>
<td>625p</td>
<td>—</td>
<td>9</td>
</tr>
<tr>
<td>720p</td>
<td>—</td>
<td>10</td>
</tr>
<tr>
<td>1080p</td>
<td>—</td>
<td>10</td>
</tr>
</tbody>
</table>

For dual link HD-SDI interface, the VPID packets are placed only in the Y data stream of both links. This submodule in the transmitter data path modifies the Y data stream that passes through.

**Note:** This submodule introduces a latency of a few clock cycles. The C data stream is delayed by a few clock cycles to keep it synchronized with the Y data stream.
The following rules apply for inserting and overwriting VPID packets:

- Rule 1: If there is no ancillary packet at the beginning of the HANC space on a line where the VPID packet is supposed to occur, the submodule inserts the VPID packet at the beginning of the HANC space.

- Rule 2: If there is an existing VPID packet at the beginning of the HANC space on a line where the VPID packet is supposed to occur, the submodule overwrites the packet with the new VPID information if the `tx_vpid_overwrite` signal is high. If the `tx_vpid_overwrite` signal is low, no overwrite takes place.

- Rule 3: If there is a different type of ancillary packet or multiple ancillary packets at the beginning of the HANC space on a line where the VPID packet is supposed to occur, the submodule does not overwrite the existing ancillary packet(s). Instead, it looks for empty space in the HANC space to insert the VPID packet after the existing ancillary packet(s). If it finds a VPID packet later in the HANC space before if finds an empty space, it overwrites the existing VPID packet with the new data if the `tx_vpid_overwrite` signal is high. If the `tx_vpid_overwrite` signal is low, no overwrite takes place.

For correct VPID insertion, assert the `tx_trs` signal for the first word of both EAV and SAV TRS as shown in the Insert Line timing diagram.

**Match TRS**

This submodule indicates that the current word is a particular TRS word in both the transmitter and receiver. This submodule has the following features:

- Decodes synchronous pulse information (generate F, V, H and AP signals).
- Indicates availability of new synchronous information (a strobe when new synchronous values are seen)
- Indicates the line number EAV extension words (for HD only)
- Indicates the CRC EAV extension words (for HD only)
- Indicates when the CRC should run and when the CRC should reset (for HD only)
- Indicates when the horizontal blanking region starts
### Scrambler

The SMPTE259M and SMPTE292M specifications define a common channel coding for both SD-SDI and HD-SDI. This channel coding consists of a scrambling function \( G_1(X) = X^9 + X^4 + 1 \), followed by NRZI encoding \( G_2(X) = X + 1 \).

The scrambling submodule implements the channel coding by iteratively applying the scrambling and NRZI encoding algorithm to each bit of the output data, processing the LSB first. The code handles both the SD transmit data (10 bits wide) and HD data (20 bits wide).

### Tx Sample

This submodule transmits oversampling block by repeating each bit of the input word a given number of times and constructs the output words. This submodule relies on the fact that the input data is only valid on \( 1/x \) of the clock cycles, where \( x \) is the oversampling factor. Both the input and output words are clocked from the same clock domain.

This submodule is required for oversampling SD-SDI by 11 times in single SD-SDI, dual and triple rate SDI, as well as oversampling HD-SDI by 2 times in single HD-SDI (with 2x oversampling mode enabled) and dual and triple rate SDI.

### Clock Enable Generator

The clock enable generator is a simple logic that generates a clock enable signal, which serves as a data valid signal, \( tx\_data\_in\_valid \) for the incoming video data signal, \( tx\_data\_in \). The video data signal is based on the incoming video standard signal, \( tx\_std \). The transmit parallel clock, \( tx\_pclk \), can be a single frequency of either 148.5 MHz or 148.35 MHz.

---

**Figure 4-17: Match TRS Output Signals Relative to the Video Data Stream**

<table>
<thead>
<tr>
<th>SAV</th>
<th>ACTIVE VIDEO</th>
<th>EAV</th>
<th>BLANKING</th>
</tr>
</thead>
<tbody>
<tr>
<td>reset CRC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>run CRC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>match LN 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>match LN 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>match CRC 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>match CRC 1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Scrambler**

The SMPTE259M and SMPTE292M specifications define a common channel coding for both SD-SDI and HD-SDI. This channel coding consists of a scrambling function \( G_1(X) = X^9 + X^4 + 1 \), followed by NRZI encoding \( G_2(X) = X + 1 \).

The scrambling submodule implements the channel coding by iteratively applying the scrambling and NRZI encoding algorithm to each bit of the output data, processing the LSB first. The code handles both the SD transmit data (10 bits wide) and HD data (20 bits wide).

**Tx Sample**

This submodule transmits oversampling block by repeating each bit of the input word a given number of times and constructs the output words. This submodule relies on the fact that the input data is only valid on \( 1/x \) of the clock cycles, where \( x \) is the oversampling factor. Both the input and output words are clocked from the same clock domain.

This submodule is required for oversampling SD-SDI by 11 times in single SD-SDI, dual and triple rate SDI, as well as oversampling HD-SDI by 2 times in single HD-SDI (with 2x oversampling mode enabled) and dual and triple rate SDI.

**Clock Enable Generator**

The clock enable generator is a simple logic that generates a clock enable signal, which serves as a data valid signal, \( tx\_data\_in\_valid \) for the incoming video data signal, \( tx\_data\_in \). The video data signal is based on the incoming video standard signal, \( tx\_std \). The transmit parallel clock, \( tx\_pclk \), can be a single frequency of either 148.5 MHz or 148.35 MHz.
The clock enable generator generates a clock signal in the following conditions:

- If the `tx_datain` signal is SD—generate a `tx_datain_valid` pulse every 5th and 11th clock cycle of the `tx_pclk` domain.
- If the `tx_datain` signal is HD—generate a `tx_datain_valid` pulse every other clock cycle of the `tx_pclk` domain.
- If the `tx_datain` signal is neither SD nor HD—the `tx_datain_valid` pulse remains high for 3G.

**Figure 4-18: Triple Rate Transmit Clocking Scheme**

Figure below illustrates the behavior of the `tx_datain_valid` pulse in each video standard.

---

### Rx Sample

This submodule extracts data from the oversampled incoming data stream. In oversampling schemes, each bit is repeated many times. For example, a stream of 0 1 0 1 may look like 000111000111 at the oversample clock or data rate.

This submodule examines the incoming data stream for logic transitions. These transitions mark the boundaries of the incoming data bit. It is desirable to extract a sample at a point between these two boundaries. This submodule identifies a transition, then extracts the sample X clocks after this transition. When no transitions are present in the incoming data stream, the submodule uses knowledge of oversampling rate to continuously sample the incoming data at a particular rate. When the next transition is present, the scheme resets and the sampling aligns to this new transition.
Table 4-2: Sampling Process

The submodule executes the sampling process in the following manner.

<table>
<thead>
<tr>
<th>Step</th>
<th>Process</th>
<th>Description</th>
</tr>
</thead>
</table>
| 1    | Detect transitions in the incoming data. | The first part of the code detects the transitions in the incoming data. This detection is done by comparing adjacent bits in the incoming data word. This process generates a word that indicates the position of the transitions in the data.
|      |         | Example: Data in = 1100000111 Transition = 0100001000 |
| 2    | Generate a sample_now data by using the transition data. | The submodule generates a sample_now bus from the transition bus and the sample_mask. Since the data is not sampled at the transition point, the submodule applies an offset to which bit it selects using the sample mask. The sample mask gives the position of the desired sample relative to the transition point.
|      |         | Example: Data in = 1100000111 Transition = 0100001000 Sample now = 0000100001 |
| 3    | Select and place the data bits in the shift register. | If a bit in the sample_now bus is set, the corresponding bit in the data input word is selected and placed into a shift register. For instance, if sample_now[4] is set, the submodule places the value of din[4] into the shift register.
|      |         | Example: Data in = 1100000111 Transition = 0100001000 Sample now = 0000100001 Data to shift reg = xxxx0xxxx1 |
| 4    | Remove the data bits from the shift register and construct a parallel word. | The submodule removes samples from the shift register one at a time and places them into consecutive bits in the output word. |
| 5    | Assert dout_valid bit. | The submodule asserts this bit to indicate that the word is complete and ready for the downstream system to process. |

Detect Video Standard

This submodule performs coarse rate detection on the incoming video stream for dual or triple rate SDI. This scheme is required so the SDI II MegaCore function can reprogram the transceivers to the correct settings for the video standard that is present at the input.
The submodule executes the detection process in the following manner:

1. Look for transitions in the incoming data words by comparing each bit with the adjacent bit in the incoming word. Then, generate a bus with one bit set for each transition or edge seen.
2. Count the number of bit sets in the bus and return a value, which represents the number of edges present in a particular input data word.
3. Count the total number of edges seen over a given number of input words using an accumulator. Then, add the number of transitions seen in the current input word to a running total of transitions seen since the accumulator was reset.
4. Compare the total number of edges with a fixed set of values determined by experimentation. The actual thresholds are relative to the data rates of the three standards.

This submodule asserts the `rate_detect_done` flag to indicate to the transceiver controller submodule that rate detection has been performed. This approach is further described in the *Transceiver Controller* section.

**Related Information**

*Transceiver Controller* on page 4-16

### Detect 1 and 1/1.001 Rates

This submodule indicates if the incoming video stream is running at PAL (1) or NTSC (1/1.001) rate. The output port signal, `rx_clkout_is_ntsc_paln` is set to 0 if the submodule detects the incoming stream as PAL (148.5 MHz or 74.25 MHz recovered clock) and set to 1 if the incoming stream is detected as NTSC (148.35 MHz or 74.175 MHz recovered clock).

For correct video rate detection, you must set the top level port signal, `rx_coreclk_is_ntsc_paln`, to the following bit:

- 0 if the `rx_coreclk` signal is 148.5 MHz or 74.25 MHz
- 1 if the `rx_coreclk` signal is 148.35 MHz or 74.175 MHz

### Transceiver Controller

The transceiver controller controls the transceiver to achieve the desired receiver functionality for the SDI.

When the interface receives SD-SDI, the transceiver receiver PLL locks to the receiver reference clock.

When the interface receives HD-SDI, the transceiver receiver PLL is first trained by locking to the receiver reference clock. When the PLL is locked, it can then track the actual receiver data rate. If a period of time passes without a valid SDI signal, the PLL is retrained with the reference clock and the process repeats.

The transceiver controller uses a different approach to detect the incoming video standard. Instead of setting the core to each of the standards and waiting to see if lock is achieved, the core directly analyses the incoming stream and try to determine the rate. This is done by looking at the edge density, or by number of transitions in the incoming stream as described in the *Detect Video Standard* section.

Firstly, the core is set into 3G mode (transceiver running at 2.97 Gbps) and in lock-to-refclk (LTR) mode. It is essentially running in a fixed frequency sampling mode. The core examines for transitions in the data...
stream. The number of transitions in the incoming stream is counted over a fixed period of time and is determined in the following manner:

- If the incoming stream is 3G, you would see X number of transitions.
- If the incoming stream is HD (1.485 Gbps), you would see X/2 transitions.
- If the incoming stream is SD, you would see X/11 transitions.

Compare the edge count value with a number of fixed values that correspond to the thresholds of the video standards. This approach works because the scrambling algorithm in the SDI guarantees a maximum and a minimum number of transitions in the SDI stream.

The output of this circuit determines if the transceiver requires dynamic reconfiguration to a new mode. The dual and triple rate SDI core uses 11x oversampling for the reception of SD-SDI. This means that you require only two transceiver setups since the rates for 3G-SDI and 11x SD-SDI are the same.

The transceiver controller uses the presence (or absence) of TRSs on the stream to determine if the SDI signal is correctly received. The detect format submodule indicates to the transceiver controller that the receiver is acquiring some valid SDI samples when it detects a single and valid TRS. The transceiver controller only deasserts this flag when it does not detect any EAV sequences within the number of consecutive lines specified. At this point, the transceiver controller state machine resets and performs the relock algorithm.

The receive transceivers can be set into one of two modes, manual or automatic. In automatic mode, a state machine internal to the transceiver controls the training. In manual mode, the external logic must take care of the transceiver training by using either lock-to-refclk (LTR) or lock-to-data (LTD) mode.

In LTR mode, a state machine internal to the receive transceiver uses the applied reference clock for operation. In this mode, the core samples the incoming data using the refclk signal and does not perform clock recovery. The sampling clock is not locked to the incoming data stream. This mode is used for transceiver training and in the oversampling modes of SD-SDI. In this mode, the rx_clkout signal of the transceiver is a mirror of the reference clock.

In LTD mode, a state machine internal to the receive transceiver uses the clock generated by the CDR circuitry. The CDR extracts a clock from the incoming data stream and uses the clock to sample the incoming data. The sampling clock locks to the incoming data. This mode can only be used once the transceiver has been trained. Use this mode to recover data for HD and 3G streams. In this mode, the rx_clkout signal of the transceiver locks to the data.

**Related Information**

*Detect Video Standard* on page 4-15

**Descrambler**

This submodule implements data descrambling as defined in the SMPTE259 and SMPTE292 specifications. This submodule is similar to the scrambler submodule, where it implements the reverse of the scrambling applied to the data. This submodule uses an LFSR and also implements NRZI.

**TRS Aligner**

The TRS aligner word aligns the descrambled receiver data until the bit order of the output data and the original video data are the same. The EAV and SAV sequences determine the correct word alignment.
The TRS aligner matches the selected pattern in the descrambled receiver data. If the aligner detects a pattern at any of the possible word alignments, it raises a flag and indicates the matched alignment. This process applies continuously to the receiver data.

In the second stage, the TRS aligner determines the correct word alignment for the data. The aligner looks for three consecutive TRSs with the same alignment and then stores that alignment. If the aligner subsequently detects two consecutive TRSs with a different alignment, then it stores this new alignment.

In the final stage, the TRS aligner applies a barrel shift function to the received data to generate the correctly aligned parallel word output. The barrel shifter allows the design to instantly switch from one alignment to another.

3Gb Demux

This submodule demultiplex the Y link A, C link A, Y link B, and C link B from the received 20-bit data for further processing. This submodule is mainly for 3G-SDI level B operation and it is required in 3G-SDI and triple rate SDI modes.

If you enable the option for level B to level A conversion, the FIFO buffer within this submodule is instantiated to transfer the received data across asynchronous clocks. This process is described in the Level B to Level A conversion section.

Related Information
• Level B to Level A Conversion on page 2-3

Extract Line

The HD-SDI includes the current video line number as part of the EAV extension words. The insert line submodule encodes the 11-bit line number in two of these extension words as defined in the SMPTE292M specification.

This submodule decodes the data words and registers them when the Match TRS submodule indicates that the current words are LN0 and LN1 extension words.
**Extract VPID**

This submodule detects one 10-bit Y data stream from an interface and extracts the VPID packet present in that data stream. For dual link HD-SDI interfaces, you need two of these modules to look for VPID packets on the Y data streams of both links. For 3G-SDI level B operation, the insert VPID submodule inserts the VPID packets as they would be in a dual link HD SDI (SMPTE372) interface. The 3G-SDI data streams must be unpacked into two HD SDI data streams, and you need two of these modules to extract the VPID packets from the Y data streams of both links.

This submodule outputs a valid signal which indicates that a valid VPID packet data is present on the submodule’s payload output port. The submodule updates this payload each time it detects an error-free SMPTE352 packet. The submodule discards erroneous packets like checksum error and the payload port retains the information from the last good packet. The valid output signal goes high immediately upon receiving a good packet. If the submodule detects erroneous packets or the packets are no longer present, the valid output signal remains high for a number of frames or fields after the last good packet is received. This submodule provides all four bytes of the VPID data on its payload output port.

**Detect Format**

This submodule monitors the line and frame timing of an incoming SDI stream. It generates various flags to indicate whether the receive stream is locked.

This submodule implements a pixel counter and a line counter. These counters are driven from the pixel clock and synchronous pulses. The basic approach is to measure a synchronous pulse over time and detect if it is consistent over a number of lines or frames. In this manner, the core can track whether the incoming SDI is stable and locked.

The word counter monitors the EAV and SAV positions in the incoming video. This is done by incrementing a counter on each valid word and storing the count value when an EAV or SAV is seen. If the count values are the same as a predefined value, the incoming video is determined to be TRS locked. The predefined value is set to 6, therefore after six consecutive lines of the same EAV and SAV timing, the `rx_trs_locked` signal is active. An enhancement allows a predefined value of consecutive missed EAV or SAV to be tolerated without deasserting the `rx_trs_locked` signal. For example, if the predefined value is 4, this means four consecutive missed EAVs does not deassert the `rx_trs_locked` signal but five consecutive missed EAVs will deassert the signal.

The line counter increments at the start of each video line. When the first active line of a field or frame is found, the line counter starts incrementing until the last active line of the same field or frame. To determine the video format, a comparison logic compares the word and line count values in the video stream against the known values predefined for various video formats. The search is done sequentially from one known value to another. If the logic finds a match, the core is determined to be frame locked and the `rx_frame_locked` signal is active. Then, the matched known values is used to compare with the word and line counts found in the subsequent video fields or frames. The core allows a predefined value of
consecutive mismatch fields or frames to be tolerated without the `rx_frame_locked` signal. For example, if the predefined value is 4, this means four consecutive mismatch fields or frames does not deassert the `rx_frame_locked` signal but five consecutive mismatch fields or frames will deassert the signal.

**Sync Streams**

This submodule is required in the HD-SDI dual link receiver as it synchronizes and deskews both data streams received by two separate transceivers of link A and link B.

This submodule contains two FIFO buffers, where each buffer holds and transfers received data and miscellaneous signals like line number and CRC error for each link. The read operation on both FIFO buffers begin when the control state machine detects that both links are alignment locked (`rx_align_locked` is active). If a TRS is first seen on link A but not link B, the control state machine halts reading from FIFO buffer link A until TRS is seen on link B. This is also similar to the case when a TRS is first seen on link B but not link A. Then, the core is considered locked and `rx_dl_locked` signal is active. The SMPTE 372 specification defines that the timing difference between link A and link B must not exceed 40 ns.

When the core is locked, the control state machine continuously sees TRS from both FIFO buffers at the same time. If not, both links might have unaligned but it does not necessarily become TRS or frame unlocked. The control state machine aligns both links at the next TRS without deasserting the `rx_dl_locked` signal. The control state machine only deasserts the `rx_dl_locked` signal when the `rx_trs_locked` signal is deasserted.

**Convert SD Bits**

This submodule is enabled when you set the **SD Interface Bit Width** parameter option to 20. This submodule converts the SD parallel data in 20 bits back to 10 bits format required for further processing.

This submodule contains a clock enable generator to generate two data valid pulses at every 11th clock cycle of the `tx_pclk` domain. Each time the data valid signal is asserted, this block will alternately output the lower 10 bits and upper 10 bits of the SD 20-bit interface data to the downstream logic.

**Clocking Scheme**

To reduce design complexity and logic resource utilization, the SDI II MegaCore function implements a clock enable generator submodule to replace both the PLL at the parallel clock domain and the FIFO buffer in the transceiver interface block.

For the oversampling rate, the SDI II MegaCore function clocking scheme is standardized by applying an oversampling rate of 11 times for SD-SDI, dual, and triple rate modes.

**Related Information**

[Clock Enable Generator](#) on page 4-13

**SDI II MegaCore Function Signals**

The following tables list the SDI II MegaCore function signals by components.

- Protocol blocks—transmitter, receiver
- Transceiver blocks—PHY management, PHY adapter, hard transceiver
### Table 4-4: Transmitter Protocol Signals

<table>
<thead>
<tr>
<th>Signal</th>
<th>Width</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tx_rst</td>
<td>1</td>
<td>Input</td>
<td>Transmitter reset signal. This signal is active high and level sensitive. For Arria V and Stratix V devices, this reset signal must be</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>synchronous to tx_pclk clock domain. For Cyclone V device, this reset signal must be synchronous to xcvr_refclk clock domain.</td>
</tr>
<tr>
<td>tx_pclk</td>
<td>1</td>
<td>Input</td>
<td>Transmitter parallel clock input. Driven by the tx_clkout signal.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SD-SDI = 148.5 MHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• HD-SDI = 74.25 MHz or 74.175 MHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• HD-SDI (2x) = 148.5 MHz or 148.35 MHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 3G-SDI = 148.5 MHz or 148.35 MHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• HD-SDI Dual Link = 74.25 MHz or 74.175 MHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Dual Standard = 148.5 MHz or 148.35 MHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Triple Standard = 148.5 MHz or 148.35 MHz</td>
</tr>
<tr>
<td>tx_enable_crc</td>
<td>1</td>
<td>Input</td>
<td>Enables CRC insertion for all modes except SD-SDI.</td>
</tr>
<tr>
<td>tx_enable_ln</td>
<td>1</td>
<td>Input</td>
<td>Enables LN insertion for all modes except SD-SDI.</td>
</tr>
<tr>
<td>tx_ln</td>
<td>11</td>
<td>Input</td>
<td>Transmitter line number. Not applicable when you disable the <strong>Insert Video Payload ID (SMPTE 352M)</strong> option in SD-SDI.</td>
</tr>
<tr>
<td>tx_std</td>
<td>2</td>
<td>Input</td>
<td>Transmitter video standard. Applicable for 3G-SDI, dual standard and triple standard only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SD-SDI = 00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• HD-SDI = 01</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 3G-SDI Level B = 10</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 3G-SDI Level A = 11</td>
</tr>
<tr>
<td>tx_datain</td>
<td>20</td>
<td>Input</td>
<td>User-supplied transmitter parallel data.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SD-SDI = bits 19:10 unused; bits 9:0 C, Y, Cr, Y multiplex</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• HD-SDI = bits 19:10 Y; bits 9:0 C</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• HD-SDI dual link = bits 19:10 Y link A, bits 9:0 C link A</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 3G-SDI Level A = bits 19:10 Y; bits 9:0 C</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 3G-SDI Level B = bits 19:10 C, Y multiplex (link A); bits 9:0 C, Y multiplex (link B)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Dual Standard = bits 19:10 Y; bits 9:0 C</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Triple Standard = bits 19:10 Y; bits 9:0 C.</td>
</tr>
</tbody>
</table>
### Functional Description

#### Signals

<table>
<thead>
<tr>
<th>Signal</th>
<th>Width</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
</table>
| tx_datain_valid      | 1     | Input     | Transmitter parallel data valid. The timing (H: High, L: Low) must be synchronous to tx_pclk clock domain and have the following settings:  
|                      |       |           | - SD-SDI = 1H 4L 1H 5L  
|                      |       |           | - HD-SDI = H  
|                      |       |           | - HD-SDI (2x) = 1H 1L  
|                      |       |           | - 3G-SDI = H  
|                      |       |           | - HD-SDI Dual Link = H  
|                      |       |           | - Dual standard = SD (1H 4L 1H 5L); HD (1H 1L)  
|                      |       |           | - Triple standard = SD (1H 4L 1H 5L); HD (1H 1L); 3G (H)  
|                      |       |           | Otherwise, this signal can be driven by the tx_dataout_valid signal for SD-SDI, HD-SDI (2x), dual standard and triple standard.  |
| tx_trs               | 1     | Input     | Transmitter TRS input. For use in LN, CRC, or VPID insertion. Assert this signal on first word of both EAV and SAV TRSs.  |
| tx_ln_b              | 11    | Input     | Transmitter line number for link B. For use in 3G-SDI, HD-SDI dual link, and triple standard line number insertion.  |
| tx_datain_b          | 20    | Input     | User-supplied transmitter parallel data for link B. Applicable for HD-SDI dual link mode only.  
|                      |       |           | - HD-SDI dual link = bits 19:10 Y link B, bits 9:0 C link B  |
| tx_datain_valid_b    | 1     | Input     | Transmitter parallel data valid for link B. Applicable for HD-SDI dual link mode only.  
|                      |       |           | - HD-SDI dual link = H  |
| tx_trs_b             | 1     | Input     | Transmitter TRS input for link B. For use in HD-SDI dual link mode LN, CRC, or VPID insertion. Assert this signal on first word of both EAV and SAV TRSs.  |
| tx_dataout           | 20    | Output    | Transmitter parallel data out.  |
| tx_dataout_valid     | 1     | Output    | Data valid generated by the core. This signal can be used to drive tx_datain_valid. The timing (H: High, L: Low) must be synchronous to tx_pclk clock domain and have the following settings:  
|                      |       |           | - SD-SDI = 1H 4L 1H 5L  
|                      |       |           | - HD-SDI = H  
|                      |       |           | - HD-SDI (2x) = 1H 1L  
|                      |       |           | - 3G-SDI = H  
|                      |       |           | - HD-SDI Dual Link = H  
|                      |       |           | - Dual standard = SD (1H 4L 1H 5L); HD (1H 1L)  
<p>|                      |       |           | - Triple standard = SD (1H 4L 1H 5L); HD (1H 1L); 3G (H)  |
| tx_dataout_b         | 20    | Output    | Transmitter parallel data out for link B. Applicable for HD-SDI dual link transmitter protocol configuration only.  |</p>
<table>
<thead>
<tr>
<th>Signal</th>
<th>Width</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tx_dataout_valid_b</td>
<td>1</td>
<td>Output</td>
<td>Data valid generated by the core for link B. Applicable for HD-SDI dual link mode only. The timing (H: High, L: Low) is identical to the tx_dataout_valid signal and is synchronous to tx_pclk clock domain.</td>
</tr>
<tr>
<td>tx_std_out</td>
<td>2</td>
<td>Output</td>
<td>Indicates the transmitted video standard. Applicable for 3G-SDI, dual standard, and triple standard modes only.</td>
</tr>
<tr>
<td>tx_vpid_overwrite</td>
<td>1</td>
<td>Input</td>
<td>When a VPID is embedded in the video stream, the core enables this signal to overwrite the existing VPID. No effect when disabled. Applicable only when you enable the Insert Video Payload ID (SMPTE 352M) option.</td>
</tr>
<tr>
<td>tx_vpid_byte1</td>
<td>8</td>
<td>Input</td>
<td>The core inserts VPID byte 1. Applicable only when you enable the Insert Video Payload ID (SMPTE 352M) option.</td>
</tr>
<tr>
<td>tx_vpid_byte2</td>
<td>8</td>
<td>Input</td>
<td>The core inserts VPID byte 2. Applicable only when you enable the Insert Video Payload ID (SMPTE 352M) option.</td>
</tr>
<tr>
<td>tx_vpid_byte3</td>
<td>8</td>
<td>Input</td>
<td>The core inserts VPID byte 3. Applicable only when you enable the Insert Video Payload ID (SMPTE 352M) option.</td>
</tr>
<tr>
<td>tx_vpid_byte4</td>
<td>8</td>
<td>Input</td>
<td>The core inserts VPID byte 4. Applicable only when you enable the Insert Video Payload ID (SMPTE 352M) option.</td>
</tr>
<tr>
<td>tx_vpid_byte1_b</td>
<td>8</td>
<td>Input</td>
<td>The core inserts VPID byte 1 for link B. For 3G-SDI, HD-SDI dual link, and triple standard modes only. Applicable only when you enable the Insert Video Payload ID (SMPTE 352M) option.</td>
</tr>
<tr>
<td>tx_vpid_byte2_b</td>
<td>8</td>
<td>Input</td>
<td>The core inserts VPID byte 2 for link B. For 3G-SDI, HD-SDI dual link, and triple standard modes only. Applicable only when you enable the Insert Video Payload ID (SMPTE 352M) option.</td>
</tr>
<tr>
<td>tx_vpid_byte3_b</td>
<td>8</td>
<td>Input</td>
<td>The core inserts VPID byte 3 for link B. For 3G-SDI, HD-SDI dual link, and triple standard modes only. Applicable only when you enable the Insert Video Payload ID (SMPTE 352M) option.</td>
</tr>
</tbody>
</table>
### Table 4-5: Receiver Protocol Signals

<table>
<thead>
<tr>
<th>Signal</th>
<th>Width</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tx_vpid_byte4_b</td>
<td>8</td>
<td>Input</td>
<td>The core inserts VPID byte 4 for link B. For 3G-SDI, HD-SDI dual link, and triple standard modes only. Applicable only when you enable the <strong>Insert Video Payload ID (SMPTE 352M)</strong> option.</td>
</tr>
<tr>
<td>tx_line_f0</td>
<td>11</td>
<td>Input</td>
<td>Line number of field 0 (F0) of inserted VPID. Applicable only when you enable the <strong>Insert Video Payload ID (SMPTE 352M)</strong> option.</td>
</tr>
<tr>
<td>tx_line_f1</td>
<td>11</td>
<td>Input</td>
<td>Line number of field 1 (F1) of inserted VPID. Applicable only when you enable the <strong>Insert Video Payload ID (SMPTE 352M)</strong> option.</td>
</tr>
<tr>
<td>rx_std</td>
<td>2</td>
<td>Output</td>
<td>Receiver video standard. Applicable for 3G-SDI, dual standard, and triple standard modes only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- SD-SDI = 00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- HD-SDI = 01</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- 3G-SDI Level B = 10</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- 3G-SDI Level A = 11</td>
</tr>
<tr>
<td>rx_std_in</td>
<td>2</td>
<td>Input</td>
<td>Indicates the received video standard. Applicable for 3G-SDI, dual standard, and triple standard modes only.</td>
</tr>
<tr>
<td>rx_clkin</td>
<td>1</td>
<td>Input</td>
<td>Receiver protocol clock input. This signal must be driven by the <strong>rx_clkout</strong> clock signal from the transceiver block.</td>
</tr>
<tr>
<td>rx_rst_proto_in</td>
<td>1</td>
<td>Input</td>
<td>Receiver protocol reset signal. This signal must be driven by the <strong>rx_rst_proto_out</strong> reset signal from the transceiver block.</td>
</tr>
<tr>
<td>rx_clkin_b</td>
<td>1</td>
<td>Input</td>
<td>Receiver protocol clock input for link B. This signal must be driven by the <strong>rx_clkout_b</strong> clock signal from the transceiver block. For HD-SDI dual link configuration only.</td>
</tr>
<tr>
<td>rx_rst_proto_in_b</td>
<td>1</td>
<td>Input</td>
<td>Receiver protocol reset signal for link B. This signal must be driven by the <strong>rx_rst_proto_out_b</strong> reset signal from the transceiver block. For HD-SDI dual link receiver protocol configuration only.</td>
</tr>
<tr>
<td>rx_dataout</td>
<td>20</td>
<td>Output</td>
<td>Receiver parallel data out.</td>
</tr>
<tr>
<td>Signal</td>
<td>Width</td>
<td>Direction</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>rx_dataout_valid</td>
<td>1</td>
<td>Output</td>
<td>Data valid from the oversampling logic. The receiver asserts this signal to</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>indicate current data on rx_dataout is valid. The timing (H: High, L: Low)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>for each video standard has the following settings:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SD-SDI = 1H 4L 1H 5L</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• HD-SDI = H</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 3G-SDI = H</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• HD-SDI Dual Link = H</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Dual standard = SD (1H 4L 1H 5L); HD (H)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Triple standard = SD (1H 4L 1H 5L); HD (H); 3G (H)</td>
</tr>
<tr>
<td>rx_f</td>
<td>1</td>
<td>Output</td>
<td>Field bit timing signal. This signal indicates which video field is</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>currently active. For interlaced frame, 0 means first field (F0) while</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 means second field (F1). For progressive frame, the value is always</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.</td>
</tr>
<tr>
<td>rx_v</td>
<td>1</td>
<td>Output</td>
<td>Vertical blanking interval timing signal. The receiver asserts this signal</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>when the vertical blanking interval is active.</td>
</tr>
<tr>
<td>rx_h</td>
<td>1</td>
<td>Output</td>
<td>Horizontal blanking interval timing signal. The receiver asserts this</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>signal when the horizontal blanking interval is active.</td>
</tr>
<tr>
<td>rx_ap</td>
<td>1</td>
<td>Output</td>
<td>Active picture interval timing signal. The receiver asserts this signal</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>when the active picture interval is active.</td>
</tr>
<tr>
<td>Signal</td>
<td>Width</td>
<td>Direction</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>rx_format</td>
<td>4</td>
<td>Output</td>
<td>Indicates the format for the received video.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SMPTE259M 525i = 0000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SMPTE259M 625i = 0001</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SMPTE274M 1080i60/59.94 = 0100</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SMPTE274M 1080i50 = 0101</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SMPTE274M 1080p24/23.98 = 0110</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SMPTE296M 720p60/59.94 = 0111</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SMPTE296M 720p50 = 1000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SMPTE296M 720p30/29.97 = 1001</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SMPTE296M 720p25 = 1010</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SMPTE296M 720p24/23.98 = 1011</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SMPTE274M 1080p30/29.97/60/59.94 = 1100</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SMPTE274M 1080p25 = 1101</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Others = Reserved</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>To differentiate between video format with 1 and 1/1.001 rate, you have to</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>also refer to the rx_clkout_is_ntsc_paln output signal. For example, if</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>rx_format = 0100, rx_clkout_is_ntsc_paln = 1, then the format for the</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>received video is 1080i59.94. Otherwise, it is 1080i60.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>To differentiate between video format across HD and 3G interfaces, you</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>have to also refer to the rx_std output signal. For example, if rx_format</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>= 1100 and rx_clkout_is_ntsc_paln = 0, rx_std = 01, then the received video</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>format is 1080p30. If the rx_std = 11 or 10, then the received video</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>format is 1080p60.</td>
</tr>
</tbody>
</table>

**Note:** For 3G-SDI, HD-SDI dual link, and triple standard modes, you may get an inconsistent format detection if the incoming data does not include a VPID packet.

<table>
<thead>
<tr>
<th>Output</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rx_eav</td>
<td>Receiver output that indicates current TRS is EAV. This signal is asserted at the fourth word of TRS, which is the XYZ word.</td>
</tr>
<tr>
<td>rx_trs</td>
<td>Receiver output that indicates current word is TRS. This signal is asserted at the first word of 3FF 000 000 TRS.</td>
</tr>
<tr>
<td>rx_align_locked</td>
<td>Alignment locked, indicating that a TRS has been spotted and word alignment is performed.</td>
</tr>
<tr>
<td>rx_trs_locked</td>
<td>TRS locked, indicating that six consecutive TRSs with same timing has been spotted.</td>
</tr>
<tr>
<td>rx_frame_locked</td>
<td>Frame locked, indicating that multiple frames with same timing has been spotted.</td>
</tr>
<tr>
<td>rx_ln</td>
<td>Receiver line number output. Applicable for all modes except SD-SDI.</td>
</tr>
<tr>
<td>rx_ln_b</td>
<td>Receiver line number output for link B. Applicable for 3G-SDI, HD-SDI dual link, and triple standard modes only.</td>
</tr>
<tr>
<td>Signal</td>
<td>Width</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>rx_clkout</td>
<td>1</td>
</tr>
<tr>
<td>rx_clkout_b</td>
<td>1</td>
</tr>
<tr>
<td>rx_dataout_b</td>
<td>20</td>
</tr>
<tr>
<td>rx_dataout_valid_b</td>
<td>1</td>
</tr>
<tr>
<td>rx_align_locked_b</td>
<td>1</td>
</tr>
<tr>
<td>rx_trs_locked_b</td>
<td>1</td>
</tr>
<tr>
<td>rx_frame_locked_b</td>
<td>1</td>
</tr>
<tr>
<td>rx_dl_locked</td>
<td>1</td>
</tr>
<tr>
<td>rx_trs_loose_lock_out</td>
<td>1</td>
</tr>
<tr>
<td>rx_trs_loose_lock_out_b</td>
<td>1</td>
</tr>
<tr>
<td>rx_line_f0</td>
<td>11</td>
</tr>
<tr>
<td>rx_line_f1</td>
<td>11</td>
</tr>
<tr>
<td>rx_crc_error_c</td>
<td>1</td>
</tr>
<tr>
<td>rx_crc_error_y</td>
<td>1</td>
</tr>
<tr>
<td>Signal</td>
<td>Width</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------</td>
</tr>
<tr>
<td>rx_crc_error_c_b</td>
<td>1</td>
</tr>
<tr>
<td>rx_crc_error_y_b</td>
<td>1</td>
</tr>
<tr>
<td>rx_vpid_byte1</td>
<td>8</td>
</tr>
<tr>
<td>rx_vpid_byte2</td>
<td>8</td>
</tr>
<tr>
<td>rx_vpid_byte3</td>
<td>8</td>
</tr>
<tr>
<td>rx_vpid_byte4</td>
<td>8</td>
</tr>
<tr>
<td>rx_vpid_valid</td>
<td>1</td>
</tr>
<tr>
<td>rx_vpid_checksum_error</td>
<td>1</td>
</tr>
<tr>
<td>rx_vpid_byte1_b</td>
<td>8</td>
</tr>
<tr>
<td>rx_vpid_byte2_b</td>
<td>8</td>
</tr>
<tr>
<td>rx_vpid_byte3_b</td>
<td>8</td>
</tr>
<tr>
<td>Signal</td>
<td>Width</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>rx_vpid_byte4_b</td>
<td>8</td>
</tr>
<tr>
<td>rx_vpid_valid_b</td>
<td>1</td>
</tr>
<tr>
<td>rx_vpid_checksum_error_b</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 4-6: Transceiver (PHY Management, PHY Adapter, and Hard Transceiver) Signals

<table>
<thead>
<tr>
<th>Signal</th>
<th>Width</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phy_mgmt_clk</td>
<td>1</td>
<td>Input</td>
<td>Clock input for the Avalon-MM PHY management interface within the Custom PHY IP core. The frequency range is 100–125 MHz. This signal is only valid for Arria V and Stratix V devices. Altera recommends that you use the same clock for the PHY management interface and transceiver reconfiguration, reconfig_clk.</td>
</tr>
<tr>
<td>phy_mgmt_clk_rst</td>
<td>1</td>
<td>Input</td>
<td>Global reset signal for the Avalon-MM PHY management interface within the Custom PHY IP core. This signal is active high and level sensitive. This signal is only valid for Arria V and Stratix V devices.</td>
</tr>
<tr>
<td>rx_rst</td>
<td>1</td>
<td>Input</td>
<td>Reset signal for the receiver. This signal is active high and level sensitive. This reset signal must be synchronous to the rx_coreclk or rx_coreclk_hd clock domain.</td>
</tr>
<tr>
<td>rx_coreclk</td>
<td>1</td>
<td>Input</td>
<td>148.5-MHz or 148.35-MHz receiver controller clock input. The frequency of this signal must be similar to the frequency of the xcvr_refclk signal. Not applicable for HD-SDI and HD-SDI dual link modes.</td>
</tr>
<tr>
<td>rx_coreclk_hd</td>
<td>1</td>
<td>Input</td>
<td>74.25-MHz or 74.175-MHz receiver controller clock input. The frequency of this signal must be similar to the frequency of the xcvr_refclk signal. Applicable for HD-SDI and HD-SDI dual link modes only.</td>
</tr>
<tr>
<td>tx_coreclk</td>
<td>1</td>
<td>Input</td>
<td>148.5-MHz or 148.35-MHz transmitter clock input signal. Not applicable for HD-SDI and HD-SDI dual link modes.</td>
</tr>
</tbody>
</table>

3 This signal is only applicable in Quartus II version 13.0.
<table>
<thead>
<tr>
<th>Signal</th>
<th>Width</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tx_coreclk_hd</td>
<td>1</td>
<td>Input</td>
<td>74.25-MHz or 74.175-MHz transmitter clock input signal. Applicable for HD-SDI and HD-SDI dual link modes only.</td>
</tr>
<tr>
<td>xcvr_refclk</td>
<td>1</td>
<td>Input</td>
<td>Clock input for the hard transceiver. Only a single reference clock frequency is required for the receiver to support both 1 and 1/1.001 rate. For example, a clock frequency of 148.5 MHz for the triple standard receiver can receive both 2.97 Gbps and 2.967 Gbps video stream. But a clock frequency of 148.5 MHz and 148.35 MHz are required to transmit 2.97 Gbps and 2.967 Gbps video stream, respectively. • SD-SDI = 148.5 MHz or 148.35 MHz • HD-SDI = 74.25 MHz or 74.175 MHz • HD-SDI (2x) = 148.5 MHz or 148.35 MHz (transmitter only) • 3G-SDI = 148.5 MHz or 148.35 MHz • HD-SDI Dual Link: 74.25 MHz or 74.175 MHz • Dual Standard: 148.5 MHz or 148.35 MHz • Triple Standard: 148.5 MHz or 148.35 MHz Applicable only when you enable the Tx PLL Dynamic Switching option.</td>
</tr>
<tr>
<td>xcvr_refclk_alt</td>
<td>1</td>
<td>Input</td>
<td>Alternative clock input for the hard transceiver. The frequency of this signal must be the alternate frequency value of the xcvr_refclk signal. • HD-SDI = 74.25 MHz or 74.175 MHz • HD-SDI (2x) = 148.5 MHz or 148.35 MHz (transmitter only) • 3G-SDI = 148.5 MHz or 148.35 MHz • HD-SDI Dual Link: 74.25 MHz or 74.175 MHz • Dual Standard: 148.5 MHz or 148.35 MHz • Triple Standard: 148.5 MHz or 148.35 MHz Applicable only when you enable the Tx PLL Dynamic Switching option.</td>
</tr>
<tr>
<td>xcvr_refclk_sel</td>
<td>1</td>
<td>Input</td>
<td>Tx PLL select signal for the transceiver reset controller. • 0 = Lock Tx PLL using Tx PLL0 • 1 = Lock Tx PLL using Tx PLL1 Applicable only when you enable the Tx PLL Dynamic Switching option.</td>
</tr>
<tr>
<td>rx_trs_loose_lock_in</td>
<td>1</td>
<td>Input</td>
<td>Indicates that the receiver protocol block detects a single and valid TRS locking signal. This signal must be driven by rx_trs_loose_lock_out of the receiver protocol block.</td>
</tr>
<tr>
<td>rx_trs_loose_lock_in_b</td>
<td>1</td>
<td>Input</td>
<td>Indicates that the receiver protocol block for link B detects a single and valid TRS locking signal. This signal must be driven by rx_trs_loose_lock_out_b of the receiver protocol block. Applicable for HD-SDI dual link receiver transceiver configuration only.</td>
</tr>
<tr>
<td>sdi_rx</td>
<td>1</td>
<td>Input</td>
<td>Serial input signal for the hard transceiver.</td>
</tr>
</tbody>
</table>

**Functional Description**

Altera Corporation
<table>
<thead>
<tr>
<th>Signal</th>
<th>Width</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sdi_rx_b</td>
<td>1</td>
<td>Input</td>
<td>Serial input signal for the hard transceiver (link B). Applicable for HD-SDI dual link configuration only.</td>
</tr>
<tr>
<td>reconfig_to_xcvr</td>
<td>70N</td>
<td>Input</td>
<td>Dynamic reconfiguration input for the hard transceiver, where N is the reconfiguration interface.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• N = 1 for Rx</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• N = 2 for Tx and bidirectional</td>
</tr>
<tr>
<td>reconfig_from_xcvr</td>
<td>46N</td>
<td>Output</td>
<td>Dynamic reconfiguration output for the hard transceiver, where N is the reconfiguration interface.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• N = 1 for Rx</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• N = 2 for Tx and bidirectional</td>
</tr>
<tr>
<td>reconfig_to_xcvr_b</td>
<td>70N</td>
<td>Input</td>
<td>Dynamic reconfiguration input for the hard transceiver (link B), where N is the reconfiguration interface. For HD-SDI dual link configuration only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• N = 1 for Rx</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• N = 2 for Tx and bidirectional</td>
</tr>
<tr>
<td>reconfig_from_xcvr_b</td>
<td>46N</td>
<td>Output</td>
<td>Dynamic reconfiguration output for the hard transceiver (link B), where N is the reconfiguration interface. For HD-SDI dual link configuration only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• N = 1 for Rx</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• N = 2 for Tx and bidirectional</td>
</tr>
<tr>
<td>rx_coreclk_is_ntsc_paln</td>
<td>1</td>
<td>Input</td>
<td>Indicates the incoming video rate. For use in all modes except SD-SDI.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 0 = PAL rate (when rx_coreclk = 148.5 MHz or rx_coreclk_hd = 74.25 MHz)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 1 = NTSC rate (when rx_coreclk = 148.35 MHz or rx_coreclk_hd = 74.175 MHz)</td>
</tr>
<tr>
<td>rx_sdi_start_reconfig</td>
<td>1</td>
<td>Output</td>
<td>Request to start dynamic reconfiguration. Applicable for dual standard and triple standard modes only.</td>
</tr>
<tr>
<td>rx_sdi_reconfig_done</td>
<td>1</td>
<td>Input</td>
<td>Indicates that dynamic reconfiguration has completed. Applicable for dual standard and triple standard modes only.</td>
</tr>
<tr>
<td>rx_clkin_smpete372</td>
<td>1</td>
<td>Input</td>
<td>Clock input for level A to level B and level B to level A operations.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Level A to level B = 148.5 MHz or 148.35 MHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Level B to level A = 74.25 MHz or 74.175 MHz</td>
</tr>
<tr>
<td>tx_rst</td>
<td>1</td>
<td>Input</td>
<td>Reset signal for the transmitter. This signal is active high and level sensitive. This reset signal must be synchronous to tx_pclk clock domain.</td>
</tr>
<tr>
<td>Signal</td>
<td>Width</td>
<td>Direction</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>tx_pclk</td>
<td>1</td>
<td>Input</td>
<td>Parallel clock input signal for the transmitter. Driven by the tx_clkout signal.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SD-SDI = 148.5 MHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• HD-SDI = 74.25 MHz or 74.175 MHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• HD-SDI (2x) = 148.5 MHz or 148.35 MHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 3G-SDI = 148.5 MHz or 148.35 MHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• HD-SDI Dual Link = 74.25 MHz or 74.175 MHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Dual Standard = 148.5 MHz or 148.35 MHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Triple Standard = 148.5 MHz or 148.35 MHz</td>
</tr>
<tr>
<td>tx_datain</td>
<td>20</td>
<td>Input</td>
<td>User-supplied parallel data signal for the transmitter.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SD-SDI = bits 19:10 unused; bits 9:0 C, Y, Cr, Y multiplex</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• HD-SDI = bits 19:10 Y; bits 9:0 C</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• HD-SDI dual link = bits 19:10 Y link A, bits 9:0 C link A</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 3G-SDI Level A = bits 19:10 Y; bits 9:0 C</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 3G-SDI Level B = bits 19:10 C, Y multiplex (link A); bits 9:0 C, Y multiplex (link B)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Dual Standard = bits 19:10 Y; bits 9:0 C</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Triple Standard = bits 19:10 Y; bits 9:0 C</td>
</tr>
<tr>
<td>tx_datain_valid</td>
<td>1</td>
<td>Input</td>
<td>Parallel data valid signal for the transmitter. The timing (H: High, L: Low) must be synchronous to tx_pclk clock domain and have the following settings:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SD-SDI = 1H 4L 1H 5L</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• HD-SDI = H</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• HD-SDI (2x) = 1H 1L</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 3G-SDI = H</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• HD-SDI Dual Link = H</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Dual Standard = SD (1H 4L 1H 5L); HD (1H 1L)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Triple standard = SD (1H 4L 1H 5L); HD (1H 1L); 3G (H)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Otherwise, this signal can be driven by tx_dataout_valid for SD-SDI, HD-SDI (2x), dual standard and triple standard.</td>
</tr>
<tr>
<td>rx_rst_proto_out</td>
<td>1</td>
<td>Output</td>
<td>Reset the receiver protocol downstream logic. This generated signal is synchronous to rx_clkout clock domain and must be used to drive the rx_rst_proto_in signal of the receiver protocol block.</td>
</tr>
<tr>
<td>rx_clkout_is_ntsc_paln</td>
<td>1</td>
<td>Input</td>
<td>Indicates the video rate received. Applicable for all modes except SD-SDI.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 0 = PAL rate</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 1 = NTSC rate</td>
</tr>
<tr>
<td>rx_dataout</td>
<td>20</td>
<td>Output</td>
<td>Parallel data out signal for the receiver.</td>
</tr>
</tbody>
</table>
### SDI II MegaCore Function Signals

<table>
<thead>
<tr>
<th>Signal</th>
<th>Width</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rx_dataout_valid</td>
<td>1</td>
<td>Output</td>
<td>Data valid from the oversampling logic. The receiver asserts this signal to indicate current data on rx_dataout is valid. The timing (H: High, L: Low) for each video standard must have the following settings:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SD-SDI = 1H 4L 1H 5L</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• HD-SDI = H</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 3G-SDI = H</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• HD-SDI Dual Link = H</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Dual standard = SD (1H 4L 1H 5L); HD (H)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Triple standard = SD (1H 4L 1H 5L); HD (H); 3G (H)</td>
</tr>
<tr>
<td>rx_clkout</td>
<td>1</td>
<td>Output</td>
<td>CDR clock output signal for the hard transceiver.</td>
</tr>
<tr>
<td>rx_pll_locked</td>
<td>1</td>
<td>Output</td>
<td>CDR PLL locked signal for the hard transceiver.</td>
</tr>
<tr>
<td>rx_rst_proto_out_b</td>
<td>1</td>
<td>Output</td>
<td>Reset the receiver protocol downstream logic. Applicable for HD-SDI dual link configuration only.</td>
</tr>
<tr>
<td>rx_dataout_b</td>
<td>20</td>
<td>Output</td>
<td>Parallel data out signal for the receiver (link B). Applicable for HD-SDI dual link configuration only.</td>
</tr>
<tr>
<td>rx_dataout_valid_b</td>
<td>1</td>
<td>Output</td>
<td>Data valid from the oversampling logic. The receiver asserts this signal to indicate current data on rx_dataout_b is valid. The timing (H: High, L: Low) for each video standard is identical to the rx_dataout_valid signal.</td>
</tr>
<tr>
<td>rx_clkout_b</td>
<td>1</td>
<td>Output</td>
<td>CDR clock output signal for the hard transceiver (link B).</td>
</tr>
<tr>
<td>rx_pll_locked_b</td>
<td>1</td>
<td>Output</td>
<td>CDR PLL locked signal for the hard transceiver (link B). Applicable for HD-SDI dual link configuration only.</td>
</tr>
<tr>
<td>tx_dataout_valid</td>
<td>1</td>
<td>Output</td>
<td>Data valid generated by the core. This signal can be used to drive tx_datain_valid. The timing (H: High, L: Low) must be synchronous to tx_pclk clock domain and have the following settings:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SD-SDI = 1H 4L 1H 5L</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• HD-SDI = H</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• HD-SDI (2x) = 1H 1L</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 3G-SDI = H</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• HD-SDI Dual Link = H</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Dual Standard = SD (1H 4L 1H 5L); HD (1H 1L)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Triple Standard = SD (1H 4L 1H 5L); HD (1H 1L); 3G (H)</td>
</tr>
<tr>
<td>tx_dataout_valid_b</td>
<td>1</td>
<td>Output</td>
<td>Data valid generated by the core for link B. For HD-SDI dual link mode only. The timing (H: High, L: Low) is identical to the tx_dataout_valid signal and is synchronous to tx_pclk clock domain.</td>
</tr>
<tr>
<td>sdi_tx</td>
<td>1</td>
<td>Output</td>
<td>Serial output signal for the hard transceiver.</td>
</tr>
<tr>
<td>Signal</td>
<td>Width</td>
<td>Direction</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------</td>
<td>-----------</td>
<td>------------------------------------------------------------------</td>
</tr>
<tr>
<td>sdi_tx_b</td>
<td>1</td>
<td>Output</td>
<td>Serial output signal for the hard transceiver (link B). Applicable for HD-SDI dual link configuration only.</td>
</tr>
<tr>
<td>tx_pll_locked</td>
<td>1</td>
<td>Output</td>
<td>PLL locked signal (Tx PLL0) for the hard transceiver.</td>
</tr>
<tr>
<td>tx_pll_locked_alt</td>
<td>1</td>
<td>Output</td>
<td>PLL locked signal (Tx PLL1) for the hard transceiver.</td>
</tr>
<tr>
<td>tx_clkout</td>
<td>1</td>
<td>Output</td>
<td>PLL clock output signal for the hard transceiver.</td>
</tr>
</tbody>
</table>
Additional Information about the document and Altera.

Document Revision History

<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>July 2013</td>
<td>2013.06.28</td>
<td>• Added a section for each new feature:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Tx PLL Dynamic Switching</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SMPTE RP168 Switching</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SD Optional 20-bit Interface for Dual/Triple Standard</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Added information about a new submodule, Convert SD Bits.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Added information about a new parameter, SD Interface Bit Width.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Added more information about the design example components—Reconfiguration Management, Reconfiguration Router, Avalon-MM Translators.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Added more information about the design example operation:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Transceiver Dynamic Reconfiguration</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Expanding to Multiple Channels</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Updated the protocol and transceiver signals table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Updated the resource utilization table.</td>
</tr>
<tr>
<td>November 2012</td>
<td>2012.11.15</td>
<td>Initial release.</td>
</tr>
</tbody>
</table>

Related Information

- Design Examples on page 3-10
- SDI II MegaCore Function Signals on page 4-20
- Resource Utilization on page 2-7
# How to Contact Altera

## Table 5-1: Altera Contact Information

<table>
<thead>
<tr>
<th>Contact</th>
<th>Contact Method</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technical support</td>
<td>Website</td>
<td><a href="http://www.altera.com/support">www.altera.com/support</a></td>
</tr>
<tr>
<td>Technical training</td>
<td>Website</td>
<td><a href="http://www.altera.com/training">www.altera.com/training</a></td>
</tr>
<tr>
<td></td>
<td>Email</td>
<td><a href="mailto:custrain@altera.com">custrain@altera.com</a></td>
</tr>
<tr>
<td>Product literature</td>
<td>Website</td>
<td><a href="http://www.altera.com/literature">www.altera.com/literature</a></td>
</tr>
<tr>
<td>Nontechnical support</td>
<td>General</td>
<td><a href="mailto:nacom@altera.com">nacom@altera.com</a></td>
</tr>
<tr>
<td></td>
<td>Software licensing</td>
<td><a href="mailto:apgcs@altera.com">apgcs@altera.com</a></td>
</tr>
</tbody>
</table>

**Related Information**

- [www.altera.com/support](http://www.altera.com/support)
- [www.altera.com/training](http://www.altera.com/training)
- [www.altera.com/literature](http://www.altera.com/literature)

---

4 You can also contact your local Altera sales office or sales representative.