Summary

The Society of Motion Picture and Television Engineers (SMPTE) serial digital interface (SDI) family of standards is widely used in professional broadcast video equipment. These interfaces are used in broadcast studios and video production centers to carry uncompressed digital video, along with embedded ancillary data such as multiple audio channels.

The Xilinx SMPTE SD/HD/3G-SDI LogicCORE™ IP is a generic SDI receive/transmit datapath that does not have any device-specific control functions. This application note provides a module containing control logic to couple the SMPTE SD/HD/3G-SDI LogicCORE IP with the Virtex®-7 GTH transceivers to form a complete SDI interface. This application note also provides several example SDI designs that run on the Xilinx® Virtex-7 FPGA VC709 evaluation board.

Terms in this document are explained in Appendix A: Glossary, page 67. Titles of SMPTE standards are listed in Appendix B: References, page 69, and referred to by SMPTE document number in the text.

Introduction

The Xilinx SMPTE SD/HD/3G-SDI LogicCORE IP (hereinafter called the SDI core) can be connected to a Virtex-7 GTH transceiver to implement an SDI interface capable of supporting the SMPTE SD-SDI, HD-SDI, and 3G-SDI standards. The SDI core and GTH transceiver must be supplemented with some additional logic to connect them together to implement a fully functional SDI interface. This application note describes this additional control and interface logic and provides the necessary control and interface modules as Verilog source code.

The primary functions of the device-specific control logic are:

- Reset logic for the GTH transceiver
- Dynamic switching of the GTH RX and TX serial clock dividers to support the three SDI standards
- Dynamic TX reference clock switching to support the two different bit rates in each of the HD-SDI and 3G-SDI standards:
  - 1.485 Gb/s and 1.485/1.001 Gb/s in HD-SDI mode
  - 2.97 Gb/s and 2.97/1.001 Gb/s in 3G-SDI mode
- Data recovery unit for recovering data in SD-SDI mode
- RX bit rate detection used to determine if the RX is receiving a 1/1 bit rate signal or a 1/1.001 bit rate signal

Also supplied with this application note is a wrapper file that contains an instance of the control module for the GTH transceiver and the SDI core with the necessary connections between them. This simplifies the process of creating an SDI interface.

The terms listed are used in this document. Figure 1 shows a simplified block diagram of how the various pieces fit together to form an SDI interface.

- The SDI core refers to the SMPTE SD/HD/3G-SDI core that is available in the Vivado® IP catalog.
• The **control module** is a module that implements the various device-specific functions required when using the GTH transceiver to implement an SDI interface using the SMPTE SDI core. The control module is supplied in source code form with this application note.

• The **SDI wrapper** is a wrapper module that instances and interconnects the SDI core and the control module. The SDI wrapper is supplied in source code form with this application note.

• The **GTH wrapper** is a Verilog module that includes an instance of a single GTHE2_CHANNEL transceiver. This wrapper is generated by the 7 Series FPGAs transceiver wizard which is available in the Vivado IP catalog.

• The **GTH common wrapper** is a Verilog or VHDL module that includes an instance of a GTHE2_COMMON block containing the QPLL for the GTH Quad. This wrapper is generated by the 7 Series FPGAs transceiver wizard along with the GTH wrapper. If the QPLL is not used in the SDI application, this wrapper is not required.

The SDI wrapper includes one instance of a control module and one instance of an SDI core. The SDI core includes both an SDI RX and an SDI TX datapath. The wrapper module is usually connected to the RX and TX units in the same GTH transceiver, but this does not have to be the case. The RX and TX units of different GTH transceivers can be connected to the same SDI wrapper. If only an SDI RX or only a SDI TX is required, the unused portions of the control module and the SDI core are optimized away during synthesis.

This application note includes two example demonstration applications using the SDI core. These applications run on the VC709 evaluation board. An inrevium SDI FMC mezzanine board is also required to provide the SDI physical interfaces.
Using Virtex-7 GTH Transceivers for SDI Interfaces

The information in this section is intended to supplement, not replace, the information in the 7 Series FPGAs GTX/GTH Transceivers User Guide (UG476) [Ref 1]. This information highlights features and operating requirements of the GTH transceivers that are of particular importance for SDI applications.

In this document, the naming convention used in the 7 Series FPGAs GTX/GTH Transceivers User Guide (UG476) [Ref 1] for the GTH transceiver ports is followed. This convention is to use only the base name of a port. When the 7 Series FPGAs transceiver wizard is used to create a GTH wrapper, all input ports names have a suffix of _in and all outputs have a suffix of _out. For example, when a port named txrate is discussed in this document, the actual name of that port in the GTH wrapper would be txrate_in.

Starting with version 3.0 of the 7 Series FPGAs transceiver wizard supplied with Vivado 2013.3 tools, all port names on the GTH wrapper are all lowercase. The ISE tools version of the wizard still generates GTH wrappers with all uppercase port names. This application note is targeted towards using Vivado tools and version 3.0, or later, of the 7 Series FPGAs transceiver wizard.

There are several clocks required in applications using GTH transceivers. The SDI protocol, which does not allow for clock correction by stuffing and removing extra data in the data stream, requires careful attention to how these clocks are generated and used in the application. GTH transceivers require reference clocks to operate. The reference clocks are used by phase-locked loops (PLLs) in the GTH transceiver Quad to generate serial clocks for the RX and TX paths.
receiver and transmitter sections of each transceiver. As described in more detail in the **GTH Transceiver Reference Clocks** section, the serial bit rate of the GTH transmitter is an integer multiple of the reference clock frequency it is using. Furthermore, the data rate of the video provided to the input of the SDI transmitter datapath must also exactly match (or be a specific multiple of) the frequency of the reference clock used by the GTH transmitter. Consequently, you must determine how to generate the transmitter reference clock so that it is frequency-locked exactly with the data rate of the video stream being transmitted.

The GTH transmitter outputs a clock on its txoutclk port at a frequency that is exactly equal to the word rate of the data that must enter the txdata port of the GTH transmitter. The txoutclk is generated in the GTH transmitter by dividing the serial clock from the PLL down to the word rate. In most applications, the txoutclk from the GTH transmitter is buffered by a global (BUFG) or horizontal (BUFH) clock buffer and then used to clock the SDI transmitter datapath and the txusrclk and txusrclk2 clock inputs of the GTH transmitter. It is possible to use a clock other than one derived directly from txoutclk as the clock source for the SDI transmitter datapath and the txusrclk and txusrclk2 ports of the GTH transmitter. A shallow TX buffer in the GTH transmitter does allow for phase differences between the data entering the txdata port and the internal clock of the GTH transmitter. However, any frequency difference between the incoming data and the internal clock frequency of the GTH transmitter (as represented by txoutclk) quickly causes the TX buffer to underflow or overflow, resulting in errors in the serial bitstream generated by the GTH transmitter. Consequently, the data rate of the data stream entering the txdata port of the GTH transmitter (as represented by the frequency of the txusrclk and txusrclk2 clocks) and the internal data rate of the GTH transmitter (as set by the transmitter reference clock and represented by the frequency of txoutclk) must match exactly.

The GTH receiver reference clock, however, does not need an exact relationship with the bit rate of the incoming SDI signal. This is because the clock and data recovery (CDR) unit in the GTH receiver can receive bit rates that are up to ±1,250 ppm away from the nominal bit rate as set by the reference clock frequency. This allows the receiver reference clock to be generated by a local oscillator that has no exact frequency relationship to the incoming SDI signal. The GTH receiver generates a recovered clock that is frequency-locked to the incoming SDI bit rate. This clock is output on the rxoutclk port of the GTH transceiver. As is described in more detail later in this application note, rxoutclk is a true recovered clock when receiving HD-SDI and 3G-SDI signals, but not when receiving SD-SDI signals. Typically, rxoutclk is buffered by a global or horizontal clock buffer and then applied to the rxusrclk and rxusrclk2 ports of the GTH receiver and used as the clock for the SDI receiver datapath.

One additional clock is required for SDI applications. This is a free-running, fixed-frequency clock that is used as the clock for the dynamic reconfiguration port (DRP) of the GTH transceiver. This same clock is also usually supplied to the control module in the SDI wrapper where it is used for timing purposes. Xilinx recommends that the frequency of this clock be at least 10 MHz. The frequency of this clock does not require any specific relationship relative to other clocks or data rates of the SDI application. This clock must not change frequencies when the SDI mode changes. It must always remain running at the same nominal frequency at all times. It also must never stop while the SDI application is active. This clock can be used for all SDI interfaces in the device.

### GTH Transceiver Reference Clocks

Virtex-7 GTH transceivers are grouped into Quads. Each Quad contains four GTHE2_CHANNEL transceiver primitives and one GTHE2_COMMON primitive containing a Quad PLL (QPLL) as shown in Figure 2. The clock generated by the QPLL is distributed to all four transceivers in the Quad. Each GTHE2_CHANNEL has its own PLL called the Channel PLL (CPLL), which can provide a clock to the RX and TX of that transceiver only. Each RX and TX unit in the Quad can be individually configured to use either the QPLL or the CPLL as its clock source. Furthermore, any RX or TX unit can dynamically switch its clock source between the QPLL and the CPLL. This configuration and the dynamic switching capability are particularly useful for SDI applications.
**Important**: The QPLL in -1 speed grade Virtex-7 GTH transceiver Quads does not support the frequency needed for SDI bit rates. When using -1 speed grade devices, only the CPLL can be used to generate the serial clock for the GTH transceiver for SDI interfaces. The QPLL in -2 or faster speed grade devices does support SDI bit rates, so in these faster devices, both the QPLL and the CPLL can be used to generate serial clocks for GTH transceivers for SDI interfaces. When using -1 speed grade devices, the unavailability of the QPLL means that, in most SDI applications, only the RX unit or only the TX unit in each transceiver can be used, but both the RX and TX cannot be used simultaneously. Again, this is a limitation only of -1 speed grade devices.

**Figure 2**: Virtex-7 GTH Transceiver Quad Configuration

Typical SDI applications require the GTH transceivers to support five different bit rates:
• 270 Mb/s for SD-SDI
• 1.485 Gb/s for HD-SDI
• 1.485/1.001 Gb/s (~1.4835 Gb/s) for HD-SDI
• 2.97 Gb/s for 3G-SDI
• 2.97/1.001 Gb/s (~2.967 Gb/s) for 3G-SDI

The CDR unit in the RX section of the GTH transceiver can receive signals with bit rates that are up to ±1,250 ppm from the reference clock frequency. Because the two bit rates of HD-SDI differ by exactly 1,000 ppm and likewise the two 3G-SDI bit rates differ by exactly 1000 ppm, it is possible to receive all five of the SDI bit rates using a single reference clock frequency while still providing a sufficient amount of ppm offset margin.

The TX section of the GTH transceiver, however, requires two different reference clock frequencies to support all five SDI bit rates. This is because the transmitters, in general, can only transmit at certain exact integer multiples of the supplied reference clock frequency. Therefore, most SDI applications provide two separate reference clocks to the GTH Quad. One of those clocks is used as the RX reference clock and both of them are used as TX reference clocks. Usually, the frequencies of the supplied reference pair are 148.5 MHz and 148.5 MHz/1.001 MHz.

The source of the GTH transceiver reference clocks is very application specific. The receiver reference clock source can be a local oscillator because it does not need to match the incoming SDI bit rate exactly. However, because the GTH transmitter line rate is always an integer multiple of the reference clock frequency, the frequency of the transmitter reference clock must be exactly related to the data rate of the transmitted data. Most often, the transmitter reference clocks are generated by genlock PLLs, thereby deriving the GTH transmitter line rate from the studio video reference signal. In some cases, such as the SDI pass-through demonstration included with this application note, the transmitter line rate is derived from the recovered clock of the GTH receiver that is receiving the SDI signal. In such cases, an external PLL is required to reduce the jitter on the recovered clock before using it as the transmitter reference clock.

In a typical SDI application, one of the two reference clocks is connected to the QPLL and the other is connected to all the CPLLs in the Quad. It does not matter which one is used for the QPLL reference clock and which is used for the CPLL reference clock. The RX units of each transceiver in the Quad are configured to use the clock from the QPLL. The TX units dynamically switch between the QPLL clock and the local CPLL clock, depending on the bit rate that is required at the moment. The GTH txsysclksel port is used to select the TX unit serial clock source between the QPLL and the CPLL. This common configuration for SDI applications is shown in Figure 3. In this figure, multiplexers that are not used dynamically in the implementation have been replaced with wires and the reference clock routing between Quads is not shown. The clocking configuration shown in Figure 3 cannot be used in -1 speed grade devices because the QPLL cannot be used for SDI.

Also, each GTH RX and TX unit has a serial clock divider that divides the selected clock by several selectable integer powers of two. This allows, for example, all of the RX units in the Quad to use the same clock frequency from the QPLL but operate at different lines rates by using different serial clock divider values. This is very useful for SDI interfaces because the 3G-SDI bit rates are exactly twice as fast the HD-SDI bit rates. And, for 270 Mb/s SD-SDI, the GTH transceiver runs at the 3G-SDI line rate using 11X oversampling techniques. Thus, by using two divisors that differ by a value of two locally in each RX unit, reception of all the SDI bit rates is supported by a single RX clock frequency from the QPLL. The ability of the TX units to also locally divide the clock source by two divisors that differ by a factor of two is also important, allowing transmission of all SDI bit rates using just two reference clock frequencies.

The serial clock divider value of each RX and TX unit can be changed dynamically using the rxrate and txrate port of each GTH transceiver. The serial clock dividers can also be changed through the DRP by using the RXOUT_DIV and TXOUT_DIV attributes. The control module provided with this application note controls the TX serial clock divider using the GTH
Using Virtex-7 GTH Transceivers for SDI Interfaces

Using Virtex-7 GTH Transceivers for SDI Interfaces

transceiver txrate port and the RX serial clock divider using the GTH transceiver RXOUT_DIV attribute which it changes through the DRP. This provides the most efficient GTH transmitter and receiver dynamic line rate change sequences for SDI applications.

The configuration shown in Figure 3 is an optimal solution for most SDI applications for several reasons:

- The receivers can receive all SDI bit rates from one fixed reference clock frequency and the QPLL provides the serial clock derived from that reference clock to all receivers in the Quad.
- The transmitters have the flexibility to dynamically switch between the clocks from QPLL and CPLL to get both frequencies they need to transmit all supported SDI bit rates.
- All four receivers and all four transmitters in the Quad are fully independent and can each be running at different SDI bit rates and can dynamically switch between bit rates without disrupting the other RX or TX units.
- For genlocked applications, modern genlock PLLs usually can simultaneously provide both required reference clock frequencies from the synchronization reference input signal.

Again, however, the configuration shown in Figure 3 cannot be used in -1 speed grade devices. SDI applications that must simultaneously use both the RX and TX units in the same GTH transceiver usually require the configuration shown in Figure 3 and cannot be implemented in -1 speed grade devices.

**Figure 3:** Typical GTH Reference Clock Implementation for SDI

In some SDI applications, it might be necessary for different SDI transmitters to be running at slightly different bit rates even though they are transmitting at the same nominal bit rate. This is often the case with SDI routers where the bit rate of each TX must exactly match the bit rate of the SDI signal received by the SDI RX to which the TX is currently connected. In these cases,
two transmitters that are transmitting at the same nominal bit rate, in fact, have bit rates that
differ by a few ppm. Supporting such applications is possible with the Virtex-7 GTH Quad
architecture because each TX unit has exclusive use of its own CPLL. To accomplish this, each
CPLL must be provided with its own individual reference clock frequency, and the number of
GTH reference clock inputs is limited. There are two reference clock inputs per GTH Quad. A
Quad can use reference clocks from the Quad above and the Quad below. Thus, it is possible
to provide some GTH Quads in the device with five different reference clock frequencies (one
for the RX and four for the four TX units), but overall, there are not enough reference clock
inputs to allow every GTH TX in the device to have its own reference clock. The phase
interpolator controlled oscillator (PICXO) technique can be very useful in these cases because
it allows a GTH TX to be pulled by a few hundred ppm away from the frequency of its serial
clock. Thus, applications where the bit rate of each SDI TX needs to be individually locked to
the bit rate of the received SDI signal can be implemented by using common reference clocks
as in Figure 3 and then using the PICXO technique with each GTH TX to set the exact bit rate
of each SDI transmitter individually. This application note does not cover the PICXO technique.
For further information about using PICXO, contact Xilinx technical support.

Resets

The GTH transceiver has very specific reset requirements as described in the 7 Series FPGAs
GTX/GTH Transceivers User Guide (UG476) [Ref 1]. The GTH transceiver requires careful
coordination of resets of the PLLs, GTH transceiver resets (gttxreset and gtrxreset), dynamic
changes of some GTH transceiver ports such as txrate, and dynamic changes of GTH
transceiver attributes through the DRP. Without proper coordination of all of these events, it is
possible for the GTH to fall into a state in which it does not function properly for SDI, a situation
from which the only possible recovery is to reconfigure the FPGA. The control module supplied
with this application note enforces all of these requirements to ensure proper operation of the
GTH transceiver.

The user application should never directly control the GTH inputs gttxreset and gtrxreset. To
ensure proper operation of the GTH transceiver, these GTH transceiver inputs must only be
controlled by the SDI control module. The user application can request GTH resets using the
various reset inputs of the control module. These reset requests are handled at the next
appropriate time by the control module, coordinating the resets with other GTH transceiver
actions so that they do not interfere.

GTH Transceiver Initialization Sequence

Immediately following FPGA configuration, the SDI control module executes initialization
sequences for the QPLL, the CPLLS, and the RX and TX units of the GTH transceiver. The
control module has separate state machines that execute the following initialization sequence
separately for the RX and the TX portions of the GTH transceiver. The sequence described
here is for the RX. The TX initialization sequence is identical except that the transmitter and
CPLL ports replace the receiver and QPLL ports.

1. After waiting at least 500 ns following completion of FPGA configuration, assert the
   qdllreset and gtrxreset signals.
2. Wait until the rx_refclk_stable input is asserted, then negate the qdllreset.
3. Wait until the qdlllock signal is asserted, then negate the gtrxreset signal.
4. Wait until the rxresetdone signal is asserted, then indicate that the initialization sequence is
   complete.

Also, the GTH txuserrdy and rxuserrdy inputs must be properly controlled. The SDI wrapper
generates both of these signals. It asserts txuserrdy five txusclk cycles after gttxreset is
negated. Likewise, it asserts rxuserrdy fives rxusclk cycles after gtrxreset is negated.

In steps 2, 3, and 4 of the initialization sequence where the sequence is waiting on a condition
to be satisfied, a timeout counter is running. If the timeout counter expires before the wait
condition is satisfied, the state machine moves to a timeout state where it increments a retry counter and then cycles back in the initialization sequence and resumes the sequence. If the retry counter reaches its maximum count due to numerous timeouts, the initialization sequence fails and the state machine moves to a fail state, indicating failure of the initialization sequence. The maximum number of retries allowed is controlled by a parameter of the SDI wrapper.

**PLL Resets**

Besides being reset during the initialization sequences that run automatically after FPGA configuration, a QPLL or CPLL must also be reset whenever there is a change in frequency or interruption of the reference clock supplied to that PLL. The reset is required in to force the PLL to relock to the reference clock. The qpllreset input of the GTH common wrapper and the cpllreset input of the GTH wrapper are controlled by the SDI control module to implement the PLL resets. The user application should not assert the PLL resets directly. The SDI control module, alone, should control the PLL resets. However, it is up to the user application to determine when PLL resets are required and request that the affected PLL be reset and that all of the GTH RX and/or TX units using the serial clock from that PLL also be reset.

The SDI wrapper has a rx_pllreset output and a tx_pllreset output. These are used to control the qpllreset input of the GTH common wrapper and the cpllreset input of the GTH wrapper. If a PLL is used by only one RX or one TX unit, then it is a simple matter to connect the correct rx_pllreset or tx_pllreset output of the SDI wrapper to the corresponding PLL reset input port. But, when a PLL provides the serial clock to multiple RX and/or TX units, it is more complicated. See the GTH PLL Usage Models for SDI Applications section for more details.

The SDI wrapper has two inputs that the application should use to request a full reset of the GTH RX (rx_gth_full_reset) and the GTH TX (tx_gth_full_reset). Asserting either of these inputs causes the appropriate reset state machine in the control module to execute the full initialization sequence of the RX or TX section of the GTH, including resetting the associated PLL. The user application must properly control the rx_gth_full_reset and tx_gth_full_reset inputs so that these initialization sequences are done whenever there is an interruption or change in the reference clock used by the PLL.

It is up to the user application to properly control the rx_refclk_stable and tx_refclk_stable inputs to the control module. These must be asserted only when the reference clocks to the PLLs are stable. As previously described, the initialization sequences wait until these inputs are asserted before negating the PLL resets. Negating the rx_refclk_stable or tx_refclk_stable inputs does not initiate a reset of the associated PLL. PLL resets are only initiated by asserting the rx_gth_full_reset and tx_gth_full_reset inputs to the control module. The rx_refclk_stable and tx_refclk_stable are only effective in delaying completion of the reset sequence after the initialization sequence has been started by assertion of rx_gth_full_reset or tx_gth_full_reset.

**GTH TX Resets**

There are three conditions that require the TX portion of the GTH transceiver to be reset.

1. Whenever the PLL that supplies the serial clock to the GTH TX is reset, the gttxreset port must be used to reset the TX section. This is done automatically after FPGA configuration by the SDI control module and whenever the user application asserts the tx_gth_full_reset to the SDI wrapper, causing both the PLL and the GTH TX to be reset.

2. The GTH gttxreset input must be asserted during dynamic changes of the txsysclksel port. The txsysclksel port is used to select between the QPLL or the CPLL as the serial clock source for the GTH TX. Each GTH transceiver has its own txsysclksel port and can independently switch its serial clock source between the two PLLs. The txsysclksel port should not be controlled directly by the application. The SDI control module dynamically changes the txsysclksel port of a GTH transceiver in response to changes on its tx_m input. When the control module detects a change on its tx_m input, it first asserts the gttxreset signal, then changes txsysclksel, and then negates gttxreset. The sequence is complete after the GTH transceiver asserts its txresetdone output. At that point, the SDI
control module indicates completion of the txsysclksel change by asserting its tx_change_done output.

3. The GTH TX is automatically reset by the GTH transceiver itself whenever its txrate input port dynamically changes. The txrate port controls the serial clock divider for the GTH TX. The user application should not change the txrate port directly. The SDI control module changes the txrate port, when appropriate, in response to changes on its tx_mode input port.

The QPLL and CPLL have different operating frequency ranges. For SDI applications, the serial clock from the QPLL is twice the frequency of the serial clock from the CPLL. Thus, when the tx_m input port of the SDI wrapper changes to request a dynamic switch of the GTH TX between the two PLLs, a dynamic change of the serial clock divider through the txrate port must also be done at the same time if the transmitter is staying in the same SDI mode. For example, when switching from an HD-SDI bit rate of 1.485 Gb/s using the QPLL as the serial clock source to an HD-SDI bit rate of 1.485/1.001 Gb/s using the CPLL as the serial clock source, both the txsysclksel and txrate ports must be changed. However, if the SDI mode, as selected by the tx_mode input port of the SDI wrapper, changes at the same time as the tx_m port, the serial clock divider might not need to be changed. For example, if changing from HD-SDI mode using the CPLL to the 3G-SDI mode using the QPLL, the txrate port does not need to change because changing from the CPLL to the QPLL inherently increases the serial clock frequency and the resulting line rate by a factor of two.

Because tx_m and tx_mode are separate input ports to the SDI wrapper, when one of these ports changes, a small settling delay is implemented before the txsysclksel and/or txrate ports are dynamically changed. This settling delay allows a short window of time for the other port to also change before the TX control logic decides whether the txrate port needs to change.

If both txrate and txsysclksel ports need to be changed to implement a requested SDI mode or bit rate change, there is a possibility that the frequency of the clock on the GTH txoutclk port could be 297 MHz for the short period of time between changes on the two ports. Because a clock period constraint of 150 MHz is typically placed on the txoutclk, logic clocked by txoutclk could be negatively impacted should the frequency of txoutclk become 297 MHz even for a short period of time. The TX control logic prevents this from occurring by carefully selecting the order in which txrate and txsysclksel are changed so that txoutclk never exceeds 150 MHz.

The SDI wrapper has three reset inputs for the TX section:

- **tx_rst**: When asserted High, this input resets the SDI TX datapath in the SDI core.
- **tx_gth_full_reset**: When asserted High, this input resets both the PLL associated with the TX and then the TX section of the GTH transceiver (gttxreset). These two resets are sequenced so that the gttxreset does not complete until after the PLL reset is complete and the PLL is locked to its reference clock.
- **tx_gth_reset**: When asserted High, this input resets only the TX section of the GTH transceiver (gttxreset). If the PLL is not locked when the gttxreset sequence begins, the gttxreset sequence does not complete until after the PLL is locked.

**GTH RX Resets**

As with the TX section, the user application should rely on the SDI control module to carefully coordinate all of the RX reset and dynamic change activities described here to prevent them from interfering with each other.

These conditions require resets of the GTH RX section.

- Whenever the PLL that supplies the serial clock to the GTH RX (usually the QPLL) is reset, the gttxreset port must be used to reset the RX section. This is done automatically after FPGA configuration by the SDI control module and whenever the user application asserts the rx_gth_full_reset to the SDI wrapper, causing both the PLL and the GTH RX to be reset. Whenever the gttxreset signal is used to reset the GTH RX for any reason, a very specific sequence must be implemented as described in the 7 Series FPGAs GTX/GTH
Using Virtex-7 GTH Transceivers for SDI Interfaces

Transceiver User Guide (UG476) [Ref 1]. This sequence involves using the DRP port to change bit 11 of DRP address 0x011 during a portion of the sequence and then restore that bit to its previous value. This bit must be 1 for normal SDI operation. A state machine in the GTH wrapper implements this complete sequence whenever GTH transceiver gtrxreset input is asserted.

- Changes in the SDI mode between SD-SDI, HD-SDI, and 3G-SDI require changes to one or more of the following four things: the rxcdrhold port, enabling or disabling the auto adaptation mode of the LPM equalizer, the RXCDR_CFG attribute, and the RXOUT_DIV attribute. The RXCDR_CFG and RXOUT_DIV attributes are changed through the DRP. The rxcdrhold port must be asserted High when the RX SDI mode is SD-SDI and Low in other SDI modes. The LPM equalizer auto adaptation mode must be disabled for SD-SDI and enabled for HD-SDI and 3G-SDI. The RXCDR_CFG attribute is modified when switching into either HD-SDI or 3G-SDI modes to optimize the CDR for the current line rate. The RXOUT_DIV attribute controls the serial clock divider for the GTH RX. The GTH RX must be reset using the gtrxreset port after dynamic changes are made to any of these four things. If more than one of these things changes during the same SDI mode change sequence, only a single gtrxreset is required after all changes have been made.

The SDI wrapper has three reset inputs for the RX section:

- **rx_rst**: When asserted High, this input resets the SDI RX datapath in the SDI core.
- **rx_gth_full_reset**: When asserted High, this input resets both the PLL associated with the RX and then the RX section of the GTH transceiver (gtrxreset). These two resets are sequenced so that the gtrxreset does not complete until after the PLL reset is complete and the PLL is locked to its reference clock.
- **rx_gth_reset**: When asserted High, this input resets only the RX section of the GTH transceiver (gtrxreset). If the PLL is not locked when the gtrxreset sequence begins, the gtrxreset sequence does not complete until after the PLL is locked.

GTH PLL Usage Models for SDI Applications

This section describes several typical configurations of PLLs and transceivers used in SDI applications. Not every possible configuration is described, but the configurations shown here are sufficient to describe the proper connection of the PLL reset and locked signals.

The SDI wrapper has three parameters that specify which serial clock sources come from the QPLL and which come from the CPLL. These attributes do not control the routing of PLL clocks. They are only used to calculate the correct RX and TX serial clock divider values and, for the TX, the value to drive onto the GTH wrapper txsyscksel port based on the current value of tx_m. These three parameters are integers and must be assigned values as described here:

- **The RX_CLK_QPLL parameter** must be set to 1 if the QPLL is the clock source for the GTH RX. This parameter must be set to 0 if the CPLL is the clock source.
- **The TX_CLK0_QPLL parameter** must be set to 1 if the QPLL is the clock source for the GTH TX when the tx_m input to the SDI wrapper is Low. This parameter must be set to 0 if the CPLL is the clock source for the GTH TX when tx_m is Low.
- **The TX_CLK1_QPLL parameter** must be set to 1 if the QPLL is the clock source for the GTH TX when the tx_m input to the SDI wrapper is High. This parameter must be set to 0 if the CPLL is the clock source for the GTH TX when tx_m is High.

All three parameters are static. There are two parameters for the TX clock to support dynamic switching of the TX between the CPLL and the QPLL using the tx_m port of the SDI wrapper. **TX_CLK0_QPLL** is used when tx_m is Low and TX_CLK1_QPLL is used when tx_m is High. In applications where the TX is not dynamically switched between the QPLL and the CPLL, set both TX_CLK0_QPLL and TX_CLK1_QPLL to 1 if the QPLL is always the TX serial clock source and to 0 if the CPLL is always the TX serial clock source.
Usage Model 1: A Single Transceiver Active in the Quad, RX Clocked by QPLL, TX Clocked by both QPLL and CPLL

In this usage model, shown in Figure 4, there is a single transceiver active in the Quad with the RX serial clock provided by the QPLL and the GTH TX dynamically switched between the QPLL and the CPLL. In this case, the RX portion of the SDI wrapper controls the QPLL reset and the TX portion controls the CPLL reset. The TX section must, however, observe the locked status of both the QPLL and the CPLL because both PLLs must be locked before the gttxreset cycle completes.

The following connections must be made:

- The gth_rxpllrreset output of the SDI wrapper must be connected to the qpllrreset port of the GTH common wrapper.
- The gth_txpllrreset output of the SDI wrapper must be connected to the cpllrreset port of the GTH wrapper.
- The gth_rxplllock input of the SDI wrapper must be connected to the qplllock output of the GTH common wrapper.
- The gth_txplllock input of the SDI wrapper must be driven by the logical OR of the GTH common wrapper qplllock output and the GTH wrapper cplllock output.
- The rx_refclk_stable input of the SDI wrapper must be asserted High only when the reference clock source to the QPLL is stable.
- The tx_refclk_stable input of the SDI wrapper must be asserted High only when the reference clock source to the CPLL is stable.
- The RX_CLK_QPLL parameter of the SDI wrapper must be set to 1.
- The tx_m input port of the SDI wrapper controls dynamic switching of the TX serial clock source by controlling the gth_txsyclksel output of the SDI wrapper which must be connected to the txsyclksel port of the GTH wrapper.
- The TX_CLK0_QPLL and TX_CLK1_QPLL parameters of the SDI wrapper must be set appropriately based on how the tx_m port is used to select between the QPLL and the CPLL. Typically, TX_CLK0_QPLL is set to 1 and TX_CLK1_QPLL is set to 0. This configures tx_m to select the QPLL as the TX serial clock source when tx_m is Low and the CPLL when tx_m is High.
- When the QPLL needs to be reset due to a reference clock change or interruption, assert the rx_gth_full_reset input of the SDI wrapper to reset both the QPLL and the GTH RX. Also assert the tx_gth_reset input of the SDI wrapper to reset the GTH TX without resetting the CPLL.
- When the CPLL needs to be reset due to a reference clock change or interruption, assert the tx_gth_full_reset input of the SDI wrapper to reset both the CPLL and the GTH TX.
Usage Model 2: A Single Transceiver Active in the Quad, RX Clocked by the QPLL, TX Clocked by the CPLL

In this usage model, shown in Figure 5, there is a single transceiver active in the Quad with the GTH RX clocked by the QPLL and the GTH TX clocked by the CPLL.

The following connections must be made:

- The gth_rxpilreset output of the SDI wrapper must be connected to the qplireset port of the GTH common wrapper.
- The gth_txpilreset output of the SDI wrapper must be connected to the cplireset port of the GTH wrapper.
- The gth_rxpillock input of the SDI wrapper must be driven by the qpllock output of the GTH common wrapper.
- The gth_txpillock input of the SDI wrapper must be driven by the cpllock output of the GTH wrapper.
- The rx_refclk_stable input of the SDI wrapper must be asserted High only when the reference clock source to the QPLL is stable.
- The tx_refclk_stable input of the SDI wrapper must be asserted High only when the reference clock source to the CPLL is stable.
- The txsysclksel input port of each GTH wrapper must be driven with a value of 2'b00 to permanently select the CPLL as the TX serial clock source. The gth_txsysclksel output port of the SDI wrapper is left unconnected.
- The RX_CLK_QPLL parameter of the SDI wrapper must be set to 1.
The TX_CLK0_QPLL and TX_CLK1_QPLL parameters of the SDI wrapper must be set to 0.

The tx_m input port of the SDI wrapper is not used and should be driven Low.

When the QPLL needs to be reset due to a reference clock change or interruption, assert the rx_gth_full_reset input of the SDI wrapper to reset both the QPLL and the GTH RX.

When the CPLL needs to be reset due to a reference clock change or interruption, assert the tx_gth_full_reset input of the SDI wrapper to reset both the CPLL and the GTH TX.

Usage Model 3: Multiple Transceivers Active in the Quad, all RX Clocked by the QPLL, each TX Dynamically Switched between the QPLL and the CPLL

In this usage model, shown in Figure 6, there are multiple transceivers active in the Quad. All GTH receivers are clocked by the QPLL. All of the GTH transmitters can be independently switched between the QPLL and their own CPLL. All of the CPLLs use the same reference clock. This model conforms to the standard usage model shown in Figure 3.

In this usage model, one SDI wrapper is chosen as the QPLL master and controls the gth_qpllreset port of the GTH common wrapper. The other SDI wrappers do not control the QPLL reset, but they do monitor the qplllock output of the GTH common wrapper so that the reset sequences of the GTH transceiver do not proceed until the QPLL is locked.

The following connections must be made:

- The gth_rxplreset output of the SDI wrapper designated as the QPLL master must be connected to the qpllreset port of the GTH common wrapper. The gth_rxplreset outputs of the other SDI wrappers in the Quad are left unconnected.
- The gth_txplreset output of each SDI wrapper must be connected to the associated GTH wrapper cpllreset port.
• The gth_rxpllock input of every SDI wrapper must be driven by the qplllock output of the GTH common wrapper.

• The gth_txpllock input of each SDI wrapper must be driven by the logical OR of the qplllock output of the GTH common wrapper and the associated GTH wrapper cpllock output.

• The rx_refclk_stable input of the QPLL master SDI wrapper must be asserted High only when the reference clock source to the QPLL is stable. The rx_refclk_stable inputs of the other SDI wrappers must be permanently wired High.

• The tx_refclk_stable input of each SDI wrapper must be asserted High only when the CPLL reference clock source is stable.

• The RX_CLK_QPLL parameter of every SDI wrapper must be set to 1.

• The tx_m input port of each SDI wrapper controls dynamic switching of the TX serial clock source in the associated GTH transceiver by controlling the gth_txsysclksel output of the SDI wrapper which must be connected to the txsysclksel port of the associated GTH wrapper.

• The TX_CLK0_QPLL and TX_CLK1_QPLL parameters of each SDI wrapper must be set appropriately based on how the tx_m port is used to select between the QPLL and the CPLL. Typically, TX_CLK0_QPLL is set to 1 and TX_CLK1_QPLL is set to 0. This configures tx_m to select the QPLL as the TX serial clock source when tx_m is Low and the CPLL when tx_m is High.

• When the QPLL needs to be reset due to a reference clock change or interruption, assert the rx_gth_full_reset input of all SDI wrappers. The QPLL master SDI wrapper resets the QPLL and all GTH RX units are reset. Also, assert the tx_gth_full_reset input of all SDI wrappers to reset the GTH TX units.

• When the CPLL reference clock source changes or is interrupted, all CPLLS using that reference clock must be reset by asserting the tx_gth_full_reset port of all the SDI wrappers.
Usage Model 4: Multiple Transceivers are Active in a Quad, all RX use the QPLL, all TX use their own CPLL

In this usage model, shown in Figure 7, there are multiple transceivers active in the Quad. All of the receivers are clocked by the QPLL. Each transmitter is clocked only by its associated CPLL. Each CPLL has its own reference clock source.

This usage model covers a very common case where multiple transceivers are active in the Quad, all implementing SDI interfaces. All the active GTH RX units in the Quad use the serial clock from the QPLL. All the GTH TX units use the serial clock from their associated CPLLs.

In this usage model, one SDI wrapper is designated as the QPLL master and controls the gth_qpllexset port of the GTH common wrapper. The other SDI wrappers do not control the QPLL reset, but they do monitor the QPLL locked output of the GTH common wrapper.
The following connections must be made:

- The \texttt{gth}\_\texttt{rxpllreset} output of the QPLL master SDI wrapper must be connected to the \texttt{qpllreset} port of the GTH common wrapper. The \texttt{gth}\_\texttt{rxpllreset} outputs of the other SDI wrappers are left unconnected.
- The \texttt{gth}\_\texttt{txpllreset} output of each SDI wrapper must be connected to the \texttt{cpllreset} port of the associated GTH wrapper.
- The \texttt{gth}\_\texttt{rxplllock} input of every SDI wrapper must be driven by the \texttt{qplllock} output of the GTH common wrapper.
- The \texttt{gth}\_\texttt{txplllock} input of each SDI wrapper must be driven by the \texttt{cplllock} output of the associated GTH wrapper.
- The \texttt{rx}\_\texttt{refclk\_stable} input of the QPLL master SDI wrapper must be asserted High only when the reference clock source to the QPLL is stable. The \texttt{rx}\_\texttt{refclk\_stable} inputs of the other SDI wrappers must be wired High.
- The \texttt{tx}\_\texttt{refclk\_stable} input of each SDI wrapper must be asserted High only when the reference clock source to the associated transceiver CPLL is stable.
- The \texttt{txsysclksel} port of each GTH wrapper must be wired to \texttt{2'b00} to permanently select the CPLL as the TX serial clock source. The \texttt{gth}\_\texttt{txsysclksel} output port of the SDI wrapper is left unconnected.
- The \texttt{RX\_CLK\_QPLL} parameter of every SDI wrapper must be set to 1.
- The \texttt{TX\_CLK0\_QPLL} and \texttt{TX\_CLK1\_QPLL} parameters of every SDI wrapper must be set to 0.
- The \texttt{tx}\_\texttt{m} input port of every SDI wrapper is not used and should be wired Low.
- When the QPLL needs to be reset due to a reference clock change or interruption, assert the \texttt{rx}\_\texttt{gth\_full\_reset} input of every SDI wrapper to reset both the QPLL and all of the GTH RX.
- When the CPLL of a particular transceiver needs to be reset due to a reference clock change or interruption, assert the \texttt{tx}\_\texttt{gth\_full\_reset} input of the associated SDI wrapper to reset both the CPLL and the GTH TX.
Usage Model 5: Using only the CPLL

In this usage model, shown in Figure 8, only the CPLLs are used as serial clock sources as would be the case when using a –1 speed grade device. Two GTH transceivers are shown, one used for SDI RX and the other for SDI TX. A single SDI wrapper is used, connected to both GTH transceivers.

The following connections must be made:
• The gth_rxpllreset output of the SDI wrapper must be connected to the cplireset port of the RX GTH wrapper.
• The gth_txpllreset output of the SDI wrapper must be connected to the cplireset port of the TX GTH wrapper.
• The gth_rxplllock input of the SDI wrapper must be driven by the cplilock output of the RX GTH wrapper.
• The gth_txplllock input of the SDI wrapper must be driven by the cplilock output of the TX GTH wrapper.
• The rx_refclk_stable input of the SDI wrapper must be asserted High only when the reference clock source to the CPLL in the RX GTH wrapper is stable.
• The tx_refclk_stable input of the SDI wrapper must be asserted High only when the reference clock source to the CPLL in the TX GTH wrapper is stable.
• The txsysclksel port of the TX GTH wrapper must be wired to 2'b00 to permanently select the CPLL as the TX serial clock source. The gth_txsysclksel output port of the SDI wrapper is left unconnected.
• The RX_CLK_QPLL parameter of the SDI wrapper must be set to 0.
• The TX_CLK0_QPLL and TX_CLK1_QPLL parameters of the SDI wrapper must be set to 0.
• The tx_m input port of every SDI wrapper is not used and should be wired Low. The only way to change the TX between the integer bit rates and the 1/1.001 bit rates is by changing the frequency of the reference clock provided to the CPLL in the TX GTH wrapper. When such reference clock frequency changes occur, the CPLL in the TX GTH wrapper must be reset by asserting the tx_gth_full_reset input of the SDI wrapper.
• When the CPLL in the RX GTH wrapper needs to be reset due to a reference clock change or interruption, assert the rx_gth_full_reset input of the SDI wrapper to reset the CPLL and the GTH RX.
SDI Electrical Interface

External SDI cable equalizers and cable drivers are required to convert the serial signals into and out of the GTH transceivers to SDI electrical standards.

An external SDI cable equalizer must be used to convert the single-ended 75Ω SDI signal to a 50Ω differential signal compatible with the receiver input signal requirements of the GTH transceiver. Appropriate SDI cable equalizers are available from several manufacturers. The differential outputs of these cable equalizers usually must be AC-coupled to the GTH receiver input signals due to common mode voltage differences. An example of interfacing a typical SDI cable equalizer to a GTH receiver is shown in Figure 9.

**Important**: The capacitance values of the AC coupling capacitors between the outputs of the external SDI cable equalizer and the serial inputs of the GTH RX must be large enough to pass the SDI pathological signals without significant signal droop. AC coupling capacitors with values of at least 1.0 μF are required and 4.7 μF capacitors are recommended.

The differential inputs of the GTH RX have built-in differential termination. As described in the *7 Series FPGAs GTX/GTH Transceivers User Guide* (UG476) [Ref 1], RX Termination Use Mode 3 is the recommended termination mode for the GTH RX inputs in SDI applications. The GTH internal programmable termination voltage should be set to 800 mV for SDI applications.
Similarly, the differential serial outputs of the GTH transmitter are connected to the inputs of a SDI cable driver, usually with AC coupling as shown in Figure 10. The cable driver converts the differential signal from the GTH transmitter into a single-ended signal with electrical characteristics meeting the SDI standards. SDI cable drivers typically have a slew rate control input that sets the slew rate of the cable driver. The slew rate requirements for SD-SDI are significantly different than the slew rate requirements for HD-SDI and 3G-SDI. The slew rate control input of the SDI cable driver is typically controlled by the FPGA. The control module supplied with this application note generates a slew rate control signal for use with the external SDI cable driver.

**Important**: The capacitance values of the AC coupling capacitors between the GTH TX serial outputs and the inputs of the external SDI cable driver must be large enough to pass the SDI pathological signals without significant signal droop. AC coupling capacitors with values of at least 1.0 µF are required and 4.7 µF capacitors are recommended.

**SD-SDI Considerations**

**Receiving SD-SDI**

The 270 Mb/s bit rate of SD-SDI is less than the minimum line rate supported by the GTH RX. To receive 270 Mb/s SD-SDI, the GTH RX is used as an asynchronous oversampler to sample the SD-SDI bitstream at 11 times 270 Mb/s (2.97 Giga-samples per second) without regard to where bit transitions occur. The CDR unit in the GTH RX is locked to the reference clock by asserting the GTH transceiver rxcdrhold input port High. This prevents the CDR from trying to lock to the slow SD-SDI signal and results in more uniform oversampling of the SD-SDI signal.

When receiving an SD-SDI signal, the auto adaptation feature of the low power mode (LMP) equalizer must be disabled. The long run lengths at the slow bit rate cause problems for the
LPM equalizer auto adaptation feature. The LPM auto adaptation feature is disabled by asserting the following ports of the GTHE2_CHANNEL primitive High: RXOSOVRDEN, RXLPMLFOVRDEN, and RXLPMLFKLOVRDEN. With version 3.0 of the 7 Series FPGAs transceiver wizard, these three ports are typically not present on the GTH wrapper and are permanently wired Low inside the GTH wrapper, so the GTH wrapper has to be hand edited. The easiest thing to do is to connect the rxcdrhold_in port of the GTH wrapper to these ports of the GTHE2_CHANNEL primitive. The rxcdrhold_in port is driven High by the SDI control logic when the receiver is in SD-SDI mode, so these three ports are driven High in SD-SDI mode if connected in this manner.

A data recovery unit (DRU), implemented in the programmable logic of the FPGA, examines the oversampled SD-SDI data from the GTH RX, determines the most likely value for each bit, and outputs the recovered data. This DRU is not part of the SDI core, but is provided as part of the SDI control module of this application note.

The DRU provided with this application note is a version of the DRU described in the Xilinx application note *Dynamically Programmable DRU for High-Speed Serial I/O* (XAPP875) [Ref 2]. The specific version provided has been optimized for recovering 270 Mb/s SD-SDI bitstreams from 11X oversampled data. The general purpose DRU described in XAPP875 can recover data using many different oversampling factors and, as a result, is larger and uses more FPGA resources than the optimized version provided here for use with the SDI core.

The SD-SDI standard SMPTE ST 259 specifies several other bit rates besides 270 Mb/s. The optimized DRU supplied with this application note only supports 270 Mb/s because the vast majority of SDI interfaces only need to support the 270 Mb/s SD-SDI bit rate. However, if other SD-SDI bit rates need to be supported by the application, the optimized DRU can be replaced with the DRU from XAPP875. Because that DRU supports fractional oversampling factors, it is possible to receive the other SD-SDI bit rates without requiring any additional RX reference clock frequencies. The 540 Mb/s SD-SDI bit rate specified by SMPTE ST 344 is within the supported line rate range of the GTH transceiver and thus the GTH RX does not need to use the DRU to receive it. However, receiving the 540 Mb/s bit rate without using the DRU requires a different reference clock frequency than is used for the other SDI bit rates. Thus, it is usually more convenient to use the XAPP875 DRU to receive the 540 Mb/s ST 344 signal using 5.5X oversampling so that the standard SDI reference clock frequency can be used.

Receiving the additional SD-SDI bit rates also requires modifications to the SDI RX rate detector which controls the locking of the SDI RX by searching sequentially through all SDI bit rates until the receiver locks. The rate detection algorithm is implemented in the triple_sdi_rx_autorate.v file supplied with the SMPTE SDI core. Xilinx does not provide an equivalent module that supports the additional SD-SDI bit rates.

The DRU does not recover a clock and, because the CDR unit in the GTH RX is locked to its reference clock, the rxoutclk is not locked to the incoming bit rate in SD-SDI mode. The DRU does produce a data strobe indicating when a 10-bit data word is ready on its output. This data strobe is used by the SDI core to generate a clock enable that is asserted at a 27 MHz rate, typically with a 5/6/5/6 cadence relative to the rxoutclk clock from the GTH. The rx_ce_sd output of the SDI wrapper is derived from the DRU data strobe and has the same cadence. Occasionally, the cadence of the DRU data strobe and the rx_ce_sd signal vary from the typical 5/6/5/6 cadence. This occurs when the DRU needs to make up for the slight difference between the actual SD-SDI bit rate and the frequency of the local reference clock provided to the PLL used by the GTH RX.

Figure 11 shows a screen capture from an oscilloscope displaying the 27 MHz rx_ce_sd signal. The scope is triggered on the rising edge of rx_ce_sd at the center of the screen. The scope is in infinite persistence mode and the waveform was allowed to accumulate for several minutes. The waveform is temperature-coded from red (indicating the most common position of the signal), to blue (indicating the least common position). The incoming SD-SDI signal that was used to create this screen capture was asynchronous to the local reference clock used by the GTH receiver. The rx_ce_sd pulses on either side of the center pulse are always 5 or 6 clock cycles away from the center pulse because of the 5/6/5/6 cadence of the rx_ce_sd signal.
The two pulses at the far right and far left of the trace are nominally 11 clock cycles from the center pulse because of the 5/6/5/6 cadence. The nominal position is marked by the yellow and red pulse. For the far right pulse, the dashed yellow vertical cursor marks the position that is 11 clock cycles from the rising edge of the center pulse. The nominal location of the central yellow/red pulses are surrounded on either side by blue pulses indicating that, occasionally, the DRU needs to make the period of the rx_ce_sd cycle either 10 clock cycles or 12 clock cycles long to compensate for the frequency differences between the local reference clock and the incoming SD-SDI signal.

The SD-SDI DRU is supplied with this application note as an encrypted, pre-generated file called `dru.ngc`. It is not possible to do any simulation of a design using the `dru.ngc` file because of its encryption. However, the `dru_sim.v` file included with this application note provides a simplified simulation model of the DRU. This file can be used during simulation to replace the `dru.ngc` file. However, this simulation model must not be used in a design intended for use in the actual FPGA because the model does not support any variation in frequency between the GTH RX reference clock and the SD-SDI bitstream.

Transmitting SD-SDI

As with reception of SD-SDI, transmission of the slow 270 Mb/s SD-SDI bit rate is not directly supported by the GTH TX. To transmit the SD-SDI signal, the GTH TX is configured for a line rate of 2.97 Gb/s. The SDI core replicates each bit to be transmitted 11 times so that the data out of the SDI core and into the txdata port of the GTH TX contains 11 consecutive copies of each bit. The resulting signal output by the GTH TX is a valid 270 Mb/s SD-SDI signal.

Generating a SD-SDI Recovered Clock

In SD-SDI mode, the rxoutclk of the GTH RX is not really a recovered clock because the CDR unit is locked to the frequency of the reference clock, not to the SD-SDI bitstream. The only
signal available that actually indicates the data rate of the incoming SD-SDI bitstream is the 27 MHz rx_ce_sd output of the SDI wrapper.

For some video applications, particularly those that do not need to retransmit the recovered video over an SDI interface, the rx_ce_sd signal might be sufficient as a recovered clock. Typically, this signal is used as a clock enable to downstream modules that are clocked with the rxoutclk from the GTH RX. This is how the SDI datapath in the SDI core works – using the rx_ce_sd signal as a clock enable.

If the received video data is to be retransmitted as an SD-SDI signal using a GTH TX, then a low-jitter recovered clock is required. The recovered clock must have low enough jitter that it can be used as a reference clock for the PLL generating the serial clock for the GTH TX. Furthermore, the frequency of the recovered clock must be 148.5 MHz so that the GTH TX can use 11X oversampling to transmit the 270 Mb/s SD-SDI data. This requires the use of an external, low-bandwidth PLL. The bandwidth of the mixed-mode clock manager (MMCM) in the Virtex-7 FPGA is too high to adequately filter out the large amounts of low frequency jitter present on the rx_ce_sd signal from the SDI receiver. The Texas Instruments LMH1983 and the Silicon Labs Si5324 can both perform this function. Both of these devices can take in the rx_ce_sd signal as a 27 MHz reference and multiply it up to 148.5 MHz while also filtering out the jitter. The resulting clock is suitable for use as a reference clock for the GTH TX. The pass-through demonstration included with this application note uses a Si5324 to generate a 148.5 MHz reference clock for the GTH TX from the 27 MHz rx_ce_sd signal in exactly this manner in SD-SDI mode. When retransmitting either HD-SDI or 3G-SDI, the same Si5324 is reprogrammed to filter jitter from the rxoutclk output of the GTH RX, doubling its frequency in the case of HD-SDI, thereby producing a low-jitter 148.5 MHz reference clock for the GTH TX.

Another option is to use an external genlock PLL and lock it to the video sync signals from the recovered video. The output of the genlock PLL will be an SD-SDI recovered clock.

Sometimes a recovered clock is required to drive external video application-specific standard product (ASSP) devices. In SD-SDI mode, such a clock probably needs to have a frequency of 27 MHz and have lower jitter than is present on the rx_ce_sd signal, but does not need to have very low jitter as is the case when producing a GTH TX reference clock. The techniques mentioned previously can be used, but it might be preferable to generate such a recovered clock entirely in the FPGA without requiring external components. Unfortunately, the jitter on the rx_ce_sd signal is too high to allow it to be used directly as a reference clock input to the Virtex-7 device MMCM. But, there is a way to generate a recovered SD-SDI clock using a spare GTH TX as shown in Figure 12.

The control module recclk_txdata port can be connected to the txdata port of a spare GTH TX. The GTH TX must use the same reference clock as the GTH RX that is receiving the SDI input signal. The txusrclk and txusrclk2 ports of the GTH TX must be connected to the same clock that is driving the rxusrclk and rxusrclk2 ports of the GTH TX and the rx_usrclk port of the SDI wrapper. The GTH TX must be configured for a line rate of 2.97 Gb/s with no encoding and with a 20-bit txdata port.

When configured in this manner, the serial output of the GTH TX is a 270 MHz clock that is frequency locked to the incoming SD-SDI signal. In other words, it is a true recovered clock for SD-SDI. The GTH TX serial output pins can be connected to a global or regional clock LVDS input of the Virtex-7 device, with appropriate care to properly terminate the CML outputs and translate them to LVDS. Then, the 270 MHz clock can be used in whatever manner is required in the FPGA. For example, it can be divided by 10 to get a 27 MHz recovered clock to drive internal or external video datapaths. The signal has low enough jitter that it can be used as a reference clock to a MMCM.

The recclk_txdata port of the DRU is not wired from the SDI control module to an output port of the SDI wrapper. However, if an application needs to use this feature, the SDI wrapper can be edited to add this output port.
The GTH TX that is used to generate the recovered SD-SDI clock does not have to be configured for SDI. It only needs to be configured to always run a 2.97 Gb/s with no encoding. The data supplied to the txdata port of the GTH transceiver from the recclk_txdata port of the control module creates a 270 MHz clock on the GTH TX serial output pins. The edges of the generated clock move around by plus or minus one bit time of the 2.97 Gb/s line rate to modify the frequency of the output signal so as to exactly match with the bit rate of input SD-SDI signal. Thus, the cycle-to-cycle jitter on the 270 MHz clock generated by the GTH TX is ±337 ps plus whatever jitter is inherent in the GTH TX output signal (1 bit time at 2.97 Gb/s is 337 ps). This can be seen in Figure 13. The top trace is the 270 MHz clock generated by the GTH TX. The scope was triggered on the rising edge of the recovered clock at the center of the screen. Looking at the rising edges of the cycles on either side of the trigger point, it is easy to see the ±337 ps cycle-to-cycle jitter as these rising edges each have three discrete rising points. The bottom trace in Figure 13 is the SD-SDI that is being retransmitted by another GTH TX.
RX Bit Rate Detection

The SDI core can automatically determine the SDI mode (SD-SDI, HD-SDI, and 3G-SDI) of the SDI signal received by the GTH RX. When it is not locked to the current SDI input signal, the SDI core sequences the GTH RX through the three different SDI modes until it detects recognizably good SDI data on the rxdata output port of the GTH wrapper. At that point, the SDI core indicates that it is locked to the SDI signal by asserting its rx_mode_locked output. It also indicates which SDI mode the RX is locked to on its sdi_mode output port.

However, when the SDI core is in HD-SDI mode, it has no way of determining if the bit rate of the input SDI signal is 1.485 Gb/s or 1.485/1.001 Gb/s. Likewise, in 3G-SDI mode, the SDI core cannot determine whether the bit rate of the input SDI signal is 2.97 Gb/s or 2.97/1.001 Gb/s. The control module supplied with this application note, however, contains a bit rate detector that can distinguish between 1.485 Gb/s and 2.97 Gb/s and between 1.485/1.001 Gb/s and 2.97/1.001 Gb/s. The SDI wrapper output port rx_bit_rate is Low when the input SDI signal bit rate is either 1.485 Gb/s or 2.97 Gb/s. And, rx_bit_rate is High when the input SDI signal bit rate is either 1.485/1.001 Gb/s or 2.97/1.001 Gb/s.

For the bit rate detection feature to work, the SDI wrapper must be supplied with a fixed-frequency clock on its clk input port. It is recommended that the frequency of this clock be at least 10 MHz. If the frequency is over 150 MHz, it might be difficult to meet timing in the bit rate detection logic. The SDI wrapper has a parameter called FXDCLK_FREQ that must be used to specify the frequency of the clock connected to the clk port. The value of FXDCLK_FREQ must be set equal to the frequency of the fixed frequency clock in Hz.

The SDI wrapper uses the fixed-frequency clock for other purposes besides RX bit rate detection. Thus, even if the bit rate detection feature is not used in a particular application, a fixed-frequency clock must be supplied to the clk port of the SDI wrapper and the FXDCLK_FREQ parameter must be set correctly.
Implementing an SDI Interface in Virtex-7

There are several steps required to implement an SDI interface in a Virtex-7 FPGA design. Those steps are:

1. Generate a GTH wrapper and GTH common wrapper using the 7 Series FPGAs transceiver wizard found in the Vivado IP catalog.
2. Generate the SMPTE SD/HD/3G-SDI IP LogicCORE from the Vivado IP catalog.
3. Instance the GTH wrapper and GTH common wrapper generated in step 1 and the SDI wrapper from this application note into the application.
4. Add the dru.ngc file from this application note as a source in the Vivado project (see the readme.txt file in xapp1187.zip for more information).
5. Apply proper timing constraints the SDI interfaces.

Generating the GTH Wrapper

Use the 7 Series FPGAs transceiver wizard to generate a GTH wrapper. Beginning with version 3.0 of the wizard, the GTHE2_COMMON block is not included in the GTH wrapper, but is instanced in a separate wrapper called the GTH common wrapper.

The GTH wrapper generated by the wizard is actually a hierarchy of wrapper levels. Beginning with version 3.0 of the wizard the upper level wrappers contain additional reset logic that is not compatible with SDI operation. So, only the lowest level GTH wrapper file is actually useful for SDI applications. The lowest level GTH wrapper always contains a single GTHE2_CHANNEL instance. The easiest way to generate and use the GTH wrapper is to use the wizard to generate just a single transceiver and then instantiate the lowest level GTH wrapper multiple times in the application, once for each GTH transceiver that is used for SDI. Also, the GTH common wrapper must be instantiated as many times as necessary, once for each GTH Quad containing transceivers implementing SDI interfaces. If only the CPLL is being used for serial clocks to the GTH transceivers, then the GTH common wrapper does not need to be instantiated at all because it only contains the QPLL. The SDI two SDI demonstration applications supplied with this application note provide examples of how to instantiate the GTH wrapper and the GTH common wrapper.

The following information details exactly the steps required to generate the GTH wrapper using the wizard version 3.0 from the Vivado IP catalog.

Important: Version 3.0 of the GTH wrapper generates GTH wrapper and GTH common wrapper files that are not completely compatible with SDI, and those wrappers do require some hand editing. Instructions for editing the wrapper files are located in the Editing the GTH wrapper section of this document.

Because the top level GTH wrapper is not used in the SDI application, it is best not to generate the GTH wrapper in the same Vivado project as the SDI application. Run Vivado tools and create a new project just for the purpose of generating the GTH wrapper for SDI. After the GTH wrapper is generated, only those GTH wrapper files that are needed for SDI can be added to the actual SDI Vivado project. Always specify the same Virtex-7 FPGA device in the GTH wrapper Vivado project and in the SDI Vivado project.

After creating the GTH wrapper Vivado project, open the IP catalog. The 7 Series FPGAs transceiver wizard is found in the I/O Interfaces folder in the top-level FPGA Features and Design folder of the Vivado IP catalog. Find the wizard in the IP catalog and double-click to launch the wizard.

Version 3.0 of the wizard does not contain a protocol template for SDI. The instructions given in this section describe how to create a GTH wrapper with all the proper settings and ports necessary for implementing an SDI interface. In the future, an SDI template will be added to the GTH wrapper.

The wizard launches with the GT Selection tab open as shown in Figure 14. Above the tabs is a text field called Component Name. The name entered here is used as the name for the GTH
wrapper file and the name of the GTH component. In this example, the component name is v7gth_sdi_wrapper.

![Figure 14: 7 Series Transceivers Wizard - GT Selection Tab](image)

Near the top of the GT selection tab, the type of transceiver used must be specified. Depending on the Virtex-7 device selected for the project, GTX and/or GTH transceivers can be selected. The device selected for the Vivado project in this example only has GTH transceivers, so only GTH transceivers can be selected and the GT Type selection menu is grayed out in Figure 14.

In the Shared Logic section, select Include Shared Logic in example directory.

When moving from tab to tab, click the tabs located under the Component Name field. Do not click OK until all tabs have been correctly set up. The OK button closes the wizard.

Select the Line Rate, RefCk Selection tab, shown in Figure 15. If the wizard had an SDI protocol template available, it would be called hd_sdi and could be selected from the Protocol drop-down list. This would select all the required settings for SDI interfaces with the GTH transceiver. However, because version 3.0 of the wizard does not have such a protocol template, leave the Protocol setting set to its default of Start from scratch.

As shown in Figure 15, set the Line Rate for both the TX and RX to 1.485 Gb/s. Set the line rate to this value no matter what SDI bit rates will actually be supported in the SDI application. Set the Reference Clock frequency for both the TX and RX to the desired value, typically 148.5 MHz.

This sets the line rate to 1.485 Gb/s and the reference clock frequency to 148.5 MHz for both RX and TX. Do not change the line rate to 1.485/1.001 Gb/s or the reference clock frequency
to 148.5/1.001 MHz. The SDI control module takes care of switching to the 1/1.001 rates from
the 1/1 rates. The control module also takes care of dynamically switching to the other line rates
of 2.97 Gb/s for 3G-SDI and 270 Mb/s for SD-SDI. The line rate specified on this tab should
always be 1.485 Gb/s. Alternative reference clock frequencies can be chosen on this tab, but
only choose from those that are available in the Reference Clock pull-down lists.

The TX off and RX off check boxes allow the creation of GTH wrappers with only transmitters
(by selecting RX off) or only receivers (by selecting TX off). In this example, neither of these
options is selected.

The Quad column does not matter in this case, so just leave it to its default value.

Use Common DRP is usually not selected for SDI applications.

The bottom section of the Line Rate, RefClk Selection tab allows you to choose which GTH
transceivers and Quads are included in the top level GTH wrapper. It also allows you to choose
the reference clocks used by the PLLs and which PLL supplies the serial clock to each
transceiver. For SDI applications, always generate a GTH wrapper with a single GTH
transceiver. It does matter which transceiver is selected and using the single transceiver
that is selected by default is the easiest.

In this example, the RX unit uses the QPLL which uses REFCLK0 Q1 as its reference clock.
The TX unit uses the CPLL referenced to REFCLK1 Q1. The wizard does not explicitly handle
the case where TX units are dynamically switched between the QPLL and the CPLL. The SDI
control module takes care of the control for this dynamic switching. But, to build a GTH wrapper
with all the PLLs active and connected properly for dynamic switching of the TX between the
QPLL and the CPLL, assign the QPLL as the RX clock source and the CPLL as the TX clock
source, and assign different reference clocks to the QPLL and the CPLL as shown in Figure 15.
In cases where the QPLL is not being used and only the CPLL is used, use the CPLL as the
reference clock source to both the RX and the TX units.

Enable the Advanced Clocking Option.
Select the **Encoding and Clocking** tab, shown in **Figure 16**.

For both the TX and the RX section, set the **External Data Width** to 20 and the **Internal Data Width** to 20. Set the TX **Encoding** and the RX **Decoding** to **None**.

**Use DRP** is always selected and cannot be deselected. Set the DRP frequency to the nominal frequency of the clock connected to the GTH drpclk port.

None of the optional ports in the top section, under the DRP frequency selection, are required for SDI.

It is highly recommended that the RX and TX buffers be used for SDI applications. Thus, you should select **Enable TX Buffer** and **Enable RX Buffer**. The **TXUSRCLK Source** is set to TXOUTCLK and cannot be changed. However, change the **RXUSRCLK Source** from its default value of TXOUTCLK to RXOUTCLK, as shown in **Figure 16**.

In the bottom Optional Ports section, the following ports are required for SDI applications: TXPCSRESET, TXRATE, TXBUFSTATUS, and RXCDRHOHLD. If the application requires that the TX units dynamically switch between the QPLL and the CPLL, then the TXSYSCLKSEL port is also required. It is recommended that the TXSYSCLSEL port always be selected and, if dynamic switching of the TX is not required, the TXSYSCLKSEL port can be hardwired to select either the QPLL or the CPLL as the serial clock source.
Select the Comma Alignment and Equalization tab, shown in Figure 17.

In the RXCOMMA Alignment section of this tab, Use COMMA detection and the RXSLIDE port are selected by default. Deselect Use COMMA detection and this automatically disables the RXSLIDE port. Comma detection and the RXSLIDE features are not used for SDI.

Change the settings in Termination and Equalization to the values shown in Figure 17. The Differential Swing and Emphasis Mode must be set to Custom, RX Equalization Mode must be set to LPM-Auto, the RX Termination Voltage must be set to Programmable, and the Trim Value must be set to 800 mV.

In the Optional Ports section, any of these ports can be enabled or disabled depending on the application requirements. The TXDIFFCTRL port is typically enabled, allowing the application to set the output swing of the TX to match the input voltage requirements of external SDI cable driver. The TXPOSTCURSOR and TXPRECURSOR ports can be selected if these ports are needed to improve the integrity of the signal from the TX to the external SDI cable driver.
Select the PCIE, SATA, PRBS tab, shown in Figure 18. Most of the options on this page are not relevant to SDI and should be left in their default values. There are a few ports in the Optional Ports section that can be useful for SDI applications.

The LOOPBACK port is selected by default. This port allows for dynamic selection of various loopback modes where the data being transmitted by the GTH TX is looped back to the GTH RX in the same transceiver. The loopback modes can be useful for debugging purposes, but generally are not used in production applications.

The TXPOWERDOWN and RXPOWERDOWN ports allow the TX and RX to be dynamically powered down to save power.
Now, all selections necessary for creating a GTH wrapper for SDI applications have been made. The CB and CC Sequence tab is for protocols that use channel bonding and clock correction. SDI uses neither of these. The Summary tab provides a summary of the selections made on the other tabs. To generate the GTH wrapper, click OK and then click Generate when the next menu opens.

The wizard generates several files, some that are required for an SDI application plus several other example files that should not be used for SDI. The files that are used all start with the component name that you assigned to the GTH wrapper in the wizard. The required files are:

- `<component_name>_gt.v/vhd` lowest level GTH wrapper
- `<component_name>_gtrxreset_seq.v/vhd` gtrxreset sequence logic
- `<component_name>_sync_block.v/vhd` synchronization logic
- `<component_name>_common.v/vhd` common wrapper

If the Vivado project name is sdi_wrapper, Verilog is selected as the default language, and the component name given to the GTH wrapper is v7gth_sdi_wrapper, then the paths to the necessary files would be:

```
<component_name>_gt.v/vhd
```

```
<component_name>_gtrxreset_seq.v/vhd
```

```
<component_name>_sync_block.v/vhd
```

```
<component_name>_common.v/vhd
```

If the Vivado project name is sdi_wrapper, Verilog is selected as the default language, and the component name given to the GTH wrapper is v7gth_sdi_wrapper, then the paths to the necessary files would be:

```
sdi_wrapper/sdi_wrapper.srcs/sources_1/ip/v7gth_sdi_wrapper/v7gth_sdi_wrapper_gt.v
```
sdi_wrapper/sdi_wrapper.srcs/sources_1/ip/v7gth_sdi_wrapper/v7gth_sdi_wrapper/example_design/v7gth_sdi_wrapper_gtrxreset_seq.v
sdi_wrapper/sdi_wrapper.srcs/sources_1/ip/v7gth_sdi_wrapper/v7gth_sdi_wrapper/example_design/v7gth_sdi_wrapper_sync_block.v
sdi_wrapper/sdi_wrapper.srcs/sources_1/ip/v7gth_sdi_wrapper/v7gth_sdi_wrapper/example_design/support/v7gth_sdi_wrapper_common.v

The support directory and the GTH common wrapper located in the support directory are not automatically created when you generate the GTH wrapper using the wizard. You must right-click the SDI wrapper item in the Sources panel and choose Open IP Example Design as shown in Figure 19.

![Image](image_url)

**Figure 19:** Generating the support directory

**Editing the GTH wrapper**

Version 3.0 of the wizard generates GTH wrapper and GTH common wrappers that require some hand editing to be compatible with SDI. This section describes the required changes to these wrapper files.
The GTH common wrapper file, named `<component_name>_common.v/vhd`, generated by version 3.0 of the wizard has an incorrect parameter that causes the QPLL to run at the wrong frequency. The parameter is QPLL_FBDIV_TOP. The correct value of this parameter for SDI is 80 when using a reference clock of 148.5 MHz or 148.5/1.001 MHz.

The QPLL_FBDIV_TOP parameter is used to calculate the correct values of the GTHE2_COMMON primitive QPLL_FBDIV and QPLL_FBDIV_RATIO parameters. If a reference clock frequency other than 148.5 MHz or 148.5/1.001 MHz is used, refer to the 7 Series FPGAs GTX/GTH Transceivers User Guide (UG476) [Ref 1] for details of how to determine the correct values for the QPLL_FBDIV and QPLL_FBDIV_RATIO parameters. Version 3.1 of the wizard is scheduled to include a fix for this issue.

In the lowest level GTH wrapper, named `<component_name>_gt.v/.vhd`, several changes need to be made to turn off auto adaptation of the LPM equalizer in SD-SDI mode. This is done by connecting the rxcdrhold_in input signal in this wrapper to the following three ports of the GTHE2_CHANNEL primitive that is instantiated inside this wrapper:

- RXOSOVRDEN
- RXLPMHFOVRDEN
- RXLPMLFKLOVRDEN

By default, these three ports are wired to the net named tied_to_ground_i. Change the signal wired to these ports to rxcdrhold_in. The GTH wrapper provided with the demonstration applications supplied with this application note has these changes made so it can be used as an example.

**Generate the SMPTE SD/HD/3G-SDI IP LogicCORE**

Use the Vivado IP catalog to generate the SMPTE SD/HD/3G-SDI core. The SMPTE SD/HD/3G-SDI core is located in the Video and Image Processing folder of the IP catalog.

The SDI core is a source code core, not a precompiled core. When the SDI core is generated, a folder is created containing the source code files for the SDI core in Verilog.

The only option available when generating the SDI core is whether or not to include the error detection and handling (EDH) processor for the RX section.

**Instantiating the GTH and SDI Wrappers**

The GTH wrapper and the SDI wrapper need to be instantiated and connected in your design. It is possible to implement the SDI interface without the SDI wrapper supplied with this application note, but the wrapper makes things easier because it interconnects the SDI control module and the SDI core. If the wrapper is not used, you must make all of these connections. The standard SDI wrapper file is called `v7gth_sdi_rxtx_wrapper.v`. There is an alternative SDI wrapper file called `v7gth_sdi_rxtx_noedh_wrapper.v` that should be used when the SDI core is generated without the RX EDH processor.

In addition to the SDI core, the SDI wrapper instances the following files:

- `v7gth_sdi_control.v`
- `v7gth_tx_control.v`
- `v7gth_tx_control_sequence.v`
- `v7gth_tx_control_fsm.v`
- `v7gth_sdi_drp_control.v`
- `v7gth_sdi_drp_arbit.v`
- `v7gth_sdi_rx_reset_control.v`
- `sdi_rate_detect.v`
- `sdi_control_sync_block.v`
The `dru.v` file is an empty module which, in Verilog, specifies the ports on the precompiled `dru.ngc` file. When using the SDI wrapper, the `dru.v` must be included in the project. The actual DRU precompiled module, `dru.ngc`, must also be added to the project as a source file, just like adding any of the Verilog files.

**Note:** Do not use the `dru_sim.v` file that is included with this application note in a design intended to be used in the actual FPGA. This file is for simulation purposes only. Using it in an actual hardware implementation results in an SDI receiver that is not able to receive SD-SDI signals correctly. For simulation purposes, the `dru_sim.v` file can be added to the design instead of the `dru.v` file and the `dru.ngc` file.

**Important:** The SDI wrapper contains an instance of the SMPTE SD/HD/3G-SDI core. The SDI wrapper must be edited so that the name given to the SDI core when it is generated is used where the core is instanced in the SDI wrapper. This can be avoided by using the component name `smpte_sdi` when generating the SMPTE SDI core.

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clk</td>
<td>In</td>
<td>1</td>
<td>This input must be connected to a fixed-frequency free running clock. This clock is used by the SDI wrapper for various timing purposes. The frequency of this clock must be specified by the parameter <code>FXDCLK_FREQ</code>. If the clock frequency does not closely match the frequency specified by <code>FXDCLK_FREQ</code>, the timing delays generated by the wrapper will not be correct and the RX bit rate detection circuit might not function. Most SDI implementations use the same global clock to drive the SDI wrapper clk and gth_drpclk ports and the GTH wrapper drpclk port. However, clk and drpclk do not have to be the same clk.</td>
</tr>
</tbody>
</table>

Table 1 describes all of the ports of the SDI Wrapper. This port list is similar to the port list of the SDI core itself, but there are some differences. Also, refer to the example SDI applications provided with this application note for examples of how to interconnect the GTH and SDI wrappers.
Receive Ports

rx_rst In 1

This synchronous reset input resets the receiver section of the SDI core. It can normally be wired Low because a reset is not required. From the time that FPGA configuration is completed until the GTH RX is fully initialized, the SDI wrapper keeps the RX section of the SDI core in reset. After the GTH RX initialization is complete, as indicated by assertion of the rx_change_done output, the SMPTE SDI core is in a fully operational mode and does not require a reset.

This input resets only the receiver section of the SDI core. It does not reset any portion of the GTH transceiver.

Both rx_ce_sd and rx_din_rdy_3g must be High when rx_rst is High to completely reset the receiver.

Asserting rx_rst also resets the state machine that controls the automatic SDI mode lock detector. Do not assert rx_rst just because the SDI RX is not locked, otherwise the SDI RX never locks.

rx_usrclk In 1

This input must be driven by the same clock that drives the rxusrclk input of the GTH, usually the rxoutclk of the GTH buffered by a global clock buffer. The clock frequency must be 148.5 MHz (or 148.5/1.001 MHz) for 3G-SDI and SD-SDI modes. It must be 74.25 MHz (or 74.25/1.001 MHz) for HD-SDI mode. All input and outputs of the wrapper prefixed with rx_ are synchronous with this clock unless otherwise noted.

rx_gth_full_reset In 1

When this input is asserted High, a full GTH RX reset sequence is initiated. First, if the gth_rxplreset output of this module is connected to a GTH PLL reset input, that PLL is reset. After the PLL locks to the reference clock input, the GTH RX is reset using the GTH transceiver gtrxreset. Completion of this reset sequence is indicated by assertion of the rx_change_done output. To reset the GTH RX and the associated PLL, this input should be asserted High and it should remain High until the SDI wrapper asserts gth_gtrxreset High. After gth_gtrxreset goes High, rx_gth_full_reset must be driven Low. The reset sequence does not continue while rx_gth_full_reset is High.

This signal connected to this input must be synchronous with the gth_drpclk clock.

rx_gth_reset In 1

When this input is asserted High, the GTH RX is reset using the GTH transceiver gtrxreset. If the PLL providing the serial clock to the GTH RX is not locked, the gtrxreset sequence does not complete until that PLL is locked. Completion of this reset sequence is indicated by assertion of the rx_change_done output. To reset the GTH RX, this input should be asserted High and it should remain High until the SDI wrapper asserts gth_gtrxreset High. After gth_gtrxreset goes High, rx_gth_reset must be driven Low. The reset sequence does not continue while rx_gth_reset is High.

This signal connected to this input must be synchronous with the gth_drpclk clock.

rx_fabric_reset_out Out 1

Following the completion of FPGA configuration, this output is asserted High and remains High until the GTH RX is fully initialized. During portions of this initialization period, the GTH rxoutclk can be erratic. To prevent problems with any modules clocked by rxoutclk, the rx_fabric_reset_out signal can be used to keep those modules reset during this period initialization period.

rx_refclk_stable In 1

This input is used by the RX initialization logic to keep the PLL that is providing the serial clock to the GTH RX in reset until the PLL reference clock is stable. If this SDI wrapper is controlling the PLL reset, then the rx_refclk_stable input must be kept Low until the PLL reference clock is stable. This input does not initiate a PLL reset. It only delays completion of the PLL reset sequence initiated by the rx_gth_full_reset input until the rx_refclk_stable input is High.

This input is treated as an asynchronous input.
Table 1: SDI Wrapper Port List (Cont’d)

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rx_frame_en</td>
<td>In</td>
<td>1</td>
<td>This input enables the SDI framer function. When this input is High, the framer automatically readjusts the output word alignment to match the alignment of each timing reference signal (TRS): EAV or SAV. Normally, this input should always be High. However, if controlled properly, this input can be used to implement TRS alignment filtering. For example, if the rx_nsp output is connected to the rx_frame_en input, the framer ignores a single misaligned TRS, keeping the existing word alignment until the new word alignment is confirmed by a second matching TRS. If a TRS alignment filtering scheme is employed, it is very important to turn off any TRS filtering during the synchronous switching lines by driving the rx_frame_en input High during the synchronous switching lines.</td>
</tr>
<tr>
<td>rx_mode_en</td>
<td>In</td>
<td>3</td>
<td>This port has unary bits to enable reception of each of the three SDI modes:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Bit 0 enables HD-SDI mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Bit 1 enables SD-SDI mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Bit 2 enables 3G-SDI mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>When a bit is High, the corresponding SDI mode is included in the search for the correct SDI mode when the SDI RX is not locked to the incoming signal. When a bit is Low, the SDI RX does not attempt to detect incoming SDI signals of that mode. Disabling unused SDI modes using these bits decreases the amount of time it takes for the SDI RX to lock to the incoming signal when it changes modes.</td>
</tr>
<tr>
<td>rx_mode</td>
<td>Out</td>
<td>2</td>
<td>When the receiver is not locked, the rx_mode port changes values as the SDI RX searches for the correct SDI mode. During this time, the rx_mode_locked output is Low. When the SDI RX detects the correct SDI mode, the rx_mode_locked output goes High and the mode of the incoming SDI signal is indicated by the rx_mode port.</td>
</tr>
<tr>
<td>rx_mode_hd</td>
<td>Out</td>
<td>1</td>
<td>These three output ports are decoded versions of the rx_mode port. Unlike the rx_mode port, which changes continuously as the SDI RX seeks to identify and lock to the incoming signal, these outputs are all forced Low when the SDI RX is not locked. The output matching the current SDI mode of the SDI RX is High when rx_mode_locked is High.</td>
</tr>
<tr>
<td>rx_mode_sd</td>
<td>Out</td>
<td>1</td>
<td>When this output is Low, the SDI RX is actively searching for the SDI mode that matches the input data stream. During this time, the rx_mode output port changes frequently. When the SDI RX locks to the correct SDI mode, the rx_mode_locked output goes High. This is a low level locked signal that indicates what is going on inside the SDI mode detection logic. When the input SDI signal is disconnect, it is common for this signal to occasionally be asserted High briefly as random noise is interpreted as valid data.</td>
</tr>
<tr>
<td>rx_mode_3g</td>
<td>Out</td>
<td>1</td>
<td>This output port indicates which bit rate is being received in HD-SDI and 3G-SDI modes, as shown below. This output is not valid in SD-SDI mode.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>HD-SDI mode:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• rx_bit_rate = 0: Bit rate = 1.485 Gb/s</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• rx_bit_rate = 1: Bit rate = 1.485/1.001 Gb/s</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3G-SDI mode:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• rx_bit_rate = 0: Bit rate = 2.97 Gb/s</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• rx_bit_rate = 1: Bit rate = 2.97/1.001 Gb/s</td>
</tr>
<tr>
<td>rx_bit_rate</td>
<td>Out</td>
<td>1</td>
<td>This output is High when the transport detection function in the SDI RX has identified the transport format of the SDI signal.</td>
</tr>
<tr>
<td>rx_t_locked</td>
<td>Out</td>
<td>1</td>
<td>This output indicates which family of video signals is being used as the transport signal on the SDI interface. This output is only valid when rx_t_locked is High. This port does not necessarily identify the video format of the picture being transported. It only identifies the transport characteristics. See Table 3 for the encoding of this port.</td>
</tr>
<tr>
<td>rx_t_family</td>
<td>Out</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>
Table 1: SDI Wrapper Port List (Cont’d)

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rx_t_rate</td>
<td>Out</td>
<td>4</td>
<td>This output indicates the frame rate of the SDI transport signal. This is not necessarily the same as the frame rate of the actual picture. See Table 4 for the encoding of this port. This output is only valid when rx_t_locked is High.</td>
</tr>
<tr>
<td>rx_t_scan</td>
<td>Out</td>
<td>1</td>
<td>This output indicates whether the SDI transport signal is interlaced (Low) or progressive (High). This is not necessarily the same as the scan mode of the actual picture. This output is only valid when rx_t_locked is High.</td>
</tr>
<tr>
<td>rx_level_b_3g</td>
<td>Out</td>
<td>1</td>
<td>This output is asserted High when the input 3G-SDI signal is level B and Low when it is 3G-SDI level A. This output is only valid when the SDI RX is locked to a 3G-SDI signal (when rx_mode_3g is High).</td>
</tr>
<tr>
<td>rx_ce_sd</td>
<td>Out</td>
<td>1</td>
<td>This output is a clock enable used in SD-SDI mode. This output is asserted, on average, one cycle of rx_usclk out of every 5.5 cycles in SD-SDI mode. The SD-SDI data stream output on the rx_ds1a port and the rx video timing signals (rx_trs, rx_eav, and rx_sav) are only valid when rx_ce_sd is High in SD-SDI mode. In other SDI modes, rx_ce_sd is always High.</td>
</tr>
<tr>
<td>rx_nsp</td>
<td>Out</td>
<td>1</td>
<td>When this output is High, it indicates that the SDI framer has detected a TRS (EAV or SAV) at a new word alignment. If rx_frame_en is High, this output is only asserted for one video sample period. If rx_frame_en is Low, this output remains High until the framer is allowed to realign to the new TRS alignment (by the assertion of rx_frame_en during the occurrence of a TRS).</td>
</tr>
<tr>
<td>rx_line_a</td>
<td>Out</td>
<td>11</td>
<td>The current line number captured from the LN words of the Y data stream of the SDI input signal is output on this port. This output is valid in HD-SDI and 3G-SDI modes, but not in SD-SDI mode. In 3G-SDI level B-DL mode, the output value is the line number from the Y data stream of link A. In 3G-SDI level B-DS mode, the output value is the line number from HD-SDI signal 1. For any case where the interface line number is not the same as the picture line number, such as for 1080p 60 Hz carried on 3G-SDI level B-DL or dual link HD-SDI, the output value on this port is the interface line number, not the picture line number.</td>
</tr>
<tr>
<td>rx_a_vpid</td>
<td>Out</td>
<td>32</td>
<td>All four user data bytes of the SMPTE ST 352 payload ID packet from data stream 1 are output on this port in this format: MS byte to LS byte: byte4, byte3, byte2, byte1. This output port is valid only when rx_a_vpid_valid is High. This port is potentially valid in any SDI mode, but only if there are ST 352 packets integrated in the SDI signal. In 3G-SDI level A mode, the output data is the ST 352 data bytes captured from data stream 1 (luma). In 3G-SDI level B-DL mode, the output data is the ST 352 data bytes captured from data stream 1 of link A. In 3G-SDI level B-DS mode, the output data is the ST 352 data bytes captured from HD-SDI signal 1.</td>
</tr>
<tr>
<td>rx_a_vpid_valid</td>
<td>Out</td>
<td>1</td>
<td>This output is High when rx_a_vpid is valid. This output should not be considered to be valid when the SDI RX is not locked.</td>
</tr>
<tr>
<td>rx_b_vpid</td>
<td>Out</td>
<td>32</td>
<td>All four user data bytes of the SMPTE ST 352 payload ID packet from data stream 2 are output on this port in this format: MS byte to LS byte: byte4, byte3, byte2, byte1. This output is valid only in 3G-SDI mode and only when rx_b_vpid_valid is High. In 3G-SDI level A mode, the output data is the ST 352 data bytes captured from data stream 2 (chroma). In 3G-SDI level B-DL mode, the output data is the ST 352 data bytes captured from data stream 1 of link B. In 3G-SDI level B-DS mode, the output data is the ST 352 data bytes captured from HD-SDI signal 2.</td>
</tr>
<tr>
<td>rx_b_vpid_valid</td>
<td>Out</td>
<td>1</td>
<td>This output is High when rx_b_vpid is valid. This output should not be considered to be valid when the SDI RX is not locked.</td>
</tr>
</tbody>
</table>
Table 1: SDI Wrapper Port List (Cont'd)

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
</tr>
</thead>
<tbody>
<tr>
<td>rx_crc_err_a</td>
<td>Out</td>
<td>1</td>
</tr>
</tbody>
</table>

This output is asserted High for one video sample period when a CRC error is detected on the previous video line. For 3G-SDI level B-DL mode, this output indicates CRC errors on link A only. For 3G-SDI level B-DS mode, this output indicates CRC errors on HD-SDI stream 1 only. There is a second output called rx_crc_err_b that indicates CRC errors on link B for 3G-SDI level B-DL mode and on HD-SDI stream 2 for 3G-SDI level B-DS mode. This output is not valid in SD-SDI mode.

The CRC error outputs are asserted High for one video line time when a CRC error has been detected on the previous video line. There is a six or seven video sample period latency, depending on the SDI mode, from the video sample in which the rx_eav signal is asserted until the rx_crc_err_a signal changes values.

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
</tr>
</thead>
<tbody>
<tr>
<td>rx_ds1a</td>
<td>Out</td>
<td>10</td>
</tr>
</tbody>
</table>

The recovered SDI data stream 1 is output on this port. The contents of this data stream are dependent on the SDI mode:
- SD-SDI: Multiplexed Y/CB/CR components
- HD-SDI: Y component
- 3G-SDI level A: Data stream 1
- 3G-SDI level B-DL: Data stream 1 of link A
- 3G-SDI level B-DS: Y component of HD-SDI signal 1

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
</tr>
</thead>
<tbody>
<tr>
<td>rx_ds2a</td>
<td>Out</td>
<td>10</td>
</tr>
</tbody>
</table>

The recovered SDI data stream 2 is output on this port. The contents of this data stream are dependent on the SDI mode:
- SD-SDI: Not used
- HD-SDI: Interleaved CB and CR components
- 3G-SDI level A: Data stream 2
- 3G-SDI level B-DL: Data stream 2 of link A
- 3G-SDI level B-DS: Interleaved CB and CR components of HD-SDI signal 1

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
</tr>
</thead>
<tbody>
<tr>
<td>rx_eav</td>
<td>Out</td>
<td>1</td>
</tr>
</tbody>
</table>

This output is asserted High for one video sample period when the XYZ word of an EAV is present on the data stream output ports (rx_ds1a, rx_ds2a, rx_ds1b, and/or rx_ds2b).

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
</tr>
</thead>
<tbody>
<tr>
<td>rx_sav</td>
<td>Out</td>
<td>1</td>
</tr>
</tbody>
</table>

This output is asserted High for one video sample period when the XYZ word of an SAV is present on the data stream output ports.

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
</tr>
</thead>
<tbody>
<tr>
<td>rx_trs</td>
<td>Out</td>
<td>1</td>
</tr>
</tbody>
</table>

This output is asserted High for four consecutive video sample periods as all four words of an EAV or SAV, starting with the 3FF word and continuing through the XYZ word, are output on the data stream ports.

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
</tr>
</thead>
<tbody>
<tr>
<td>rx_line_b</td>
<td>Out</td>
<td>11</td>
</tr>
</tbody>
</table>

This output port is only valid in 3G-SDI level B mode, and outputs the line number for the Y data stream of link B (level B-DL) or HD-SDI signal 2 (level B-DS). For any case where the interface line number is not the same as the picture line number, the line number output on this port is the interface line number, not the picture line number.

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
</tr>
</thead>
<tbody>
<tr>
<td>rx_dout_rdy_3g</td>
<td>Out</td>
<td>1</td>
</tr>
</tbody>
</table>

In 3G-SDI level B mode, the output data rate is 74.25 MHz, but the rx_usuario frequency is 148.5 MHz. The rx_dout_rdy_3G output is asserted every other cycle of rx_usuario in 3G-SDI level B mode. When this output is High, the data stream and video timing outputs are valid. This output is always High in all other SDI modes, allowing it to be used as a clock enable to downstream modules.

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
</tr>
</thead>
<tbody>
<tr>
<td>rx_crc_err_b</td>
<td>Out</td>
<td>1</td>
</tr>
</tbody>
</table>

This is the CRC error indicator valid only in 3G-SDI level B mode. It indicates that a CRC error was detected on link B for 3G-SDI B-DL signals and HD-SDI signal 2 for 3G-SDI level B-DS signals.

This output has the same timing as the rx_crc_err_a signal.

To distinguish between 3G-SDI B-DL and B-DS, the value output on the rx_a_vpid or rx_b_vpid ports must be decoded.
### Table 1: SDI Wrapper Port List (Cont'd)

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rx_ds1b</td>
<td>Out</td>
<td>10</td>
<td>This output is only valid in 3G-SDI level B mode. The data stream output on this port is:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- 3G-SDI level B-DL: Data stream 1 of link B</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- 3G-SDI level B-DS: Y component of HD-SDI signal 2</td>
</tr>
<tr>
<td>rx_ds2b</td>
<td>Out</td>
<td>10</td>
<td>This output is only valid in 3G-SDI level B mode. The data stream output on this port is:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- 3G-SDI level B-DL: Data stream 2 of link B</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- 3G-SDI level B-DS: Interleaved CB and CR components of HD-SDI signal 2</td>
</tr>
<tr>
<td>rx_edh_errcnt_en</td>
<td>In</td>
<td>16</td>
<td>This input controls which EDH error conditions increment the EDH error counter. See Table 5 for more details.</td>
</tr>
<tr>
<td>rx_edh_clr_errcnt</td>
<td>In</td>
<td>1</td>
<td>When High, this input clears the EDH error counter. The EDH error counter is cleared on the rising edge of rx_usrclk only if both rx_edh_clr_errcnt and rx_ce_sd are both High.</td>
</tr>
<tr>
<td>rx_edh_ap</td>
<td>Out</td>
<td>1</td>
<td>This output is asserted High when the active picture (AP) CRC calculated for the previous field does not match the AP CRC value in the EDH packet.</td>
</tr>
<tr>
<td>rx_edh_ff</td>
<td>Out</td>
<td>1</td>
<td>This output is asserted High when the full field (FF) CRC calculated for the previous field does not match the FF CRC value in the EDH packet.</td>
</tr>
<tr>
<td>rx_edh_anc</td>
<td>Out</td>
<td>1</td>
<td>This output is asserted High when an ancillary data packet checksum error is detected.</td>
</tr>
<tr>
<td>rx_edh_ap_flags</td>
<td>Out</td>
<td>5</td>
<td>The active picture error flag bits from the most recently received EDH packet are output on this port. See Table 6 for more information.</td>
</tr>
<tr>
<td>rx_edh_ff_flags</td>
<td>Out</td>
<td>5</td>
<td>The full field error flag bits from the most recently received EDH packet are output on this port. See Table 6 for more information.</td>
</tr>
<tr>
<td>rx_edh_anc_flags</td>
<td>Out</td>
<td>5</td>
<td>The ancillary data error flag bits from the most recently received EDH packet are output on this port. See Table 6 for more information.</td>
</tr>
<tr>
<td>rx_edh_packet_flags</td>
<td>Out</td>
<td>4</td>
<td>This port outputs four error flags related to the most recently received EDH packet. See Table 7 for more information.</td>
</tr>
<tr>
<td>rx_edh_errcnt</td>
<td>Out</td>
<td>16</td>
<td>This is the SD-SDI EDH error counter. It increments each field when any of the error conditions enabled by the rx_edh_err_en port occur. The counter increments at most once per field even if multiple errors occur during that field.</td>
</tr>
<tr>
<td>rx_change_done</td>
<td>Out</td>
<td>1</td>
<td>This output is Low during those periods when the GTH RX is being initialized, reset, or when it is being dynamically switched between SDI modes. If the initialization, reset, or dynamic change sequence completes successfully, the rx_change_done output is asserted High to indicate successful completion. This output is synchronous with the gth_drpclk.</td>
</tr>
<tr>
<td>rx_change_fail</td>
<td>Out</td>
<td>1</td>
<td>Under normal conditions, this output is always Low. It only goes High if the control module is unsuccessful in completing a GTH RX initialization, reset, or SDI mode change sequence. If such a failure occurs, the rx_change_fail port is asserted High and the rx_change_fail_code port indicates the nature of the failure. If a failure occurs, the GTH RX must be reset using the rx_gth_full_reset input. This output is synchronous with the gth_drpclk.</td>
</tr>
<tr>
<td>rx_change_fail_code</td>
<td>Out</td>
<td>3</td>
<td>When the rx_change_fail port is High, this port indicates the nature of the sequence failure. See Table 8 for encoding of this port.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>This output is synchronous with the gth_drpclk.</td>
</tr>
</tbody>
</table>
### Table 1: SDI Wrapper Port List (Cont'd)

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Transmit Ports</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tx_rst</td>
<td>In</td>
<td>1</td>
<td>This synchronous reset input resets the transmitter section of the SDI core. It can normally be hardwired Low because a reset is usually not required. After FPGA configuration is complete, the transmitter in the SDI core is in a fully operational mode and does not require a reset. This input resets only the transmitter section of the SDI core. It does not initiate a reset of the GTH transceiver. Both tx_ce and tx_din_rdy must be High when tx_rst is High to completely reset the transmitter section of the SDI core.</td>
</tr>
<tr>
<td>tx_usrclk</td>
<td>In</td>
<td>1</td>
<td>This input must be driven by the same clock that is driving the txusrclk port of the GTH transceiver: usually txoutclk of the GTH buffered by a global clock buffer. It must have a frequency of 74.25 MHz or 74.25/1.001 MHz for HD-SDI and 148.5 MHz or 148.5/1.001 MHz for 3G-SDI and 148.5 MHz for SD-SDI. The combination of the tx_usrclk frequency and tx_ce must result in a 27 MHz data rate in SD-SDI mode. All input and outputs of the SDI wrapper prefixed with tx_ are synchronous with this clock unless otherwise noted.</td>
</tr>
<tr>
<td>tx_gth_full_reset</td>
<td>In</td>
<td>1</td>
<td>When this input is asserted High, a full GTH TX reset sequence is initiated. First, if the gth_txpllreset output of this module is connected to a GTH PLL reset input, that PLL is reset. After the PLL locks to the reference clock input, the GTH TX is reset using the GTH transceiver gtxreset. Completion of this reset sequence is indicated by assertion of the tx_change_done output. To reset the GTH TX and its associated PLL, this input should be asserted High and it should remain High until the SDI wrapper asserts gth_gtxreset High. Once gth_gtxreset goes High, tx_gth_full_reset must be driven Low. The reset sequence does not continue while tx_gth_full_reset is High. This signal connected to this input must be synchronous with the gth_drpclk clock.</td>
</tr>
<tr>
<td>tx_gth_reset</td>
<td>In</td>
<td>1</td>
<td>When this input is asserted High, the GTH TX is reset using the GTH transceiver gtxreset. If the PLL providing the serial clock to the GTH TX is not locked, the gtxreset sequence will not complete until that PLL is locked. Completion of this reset sequence is indicated by assertion of the tx_change_done output. To reset the GTH TX, this input should be asserted High and it should remain High until the SDI wrapper asserts gth_gtxreset High. Once gth_gtxreset goes High, tx_gth_reset must be driven Low. The reset sequence will not continue while tx_gth_reset is High. This signal connected to this input must be synchronous with the gth_drpclk clock.</td>
</tr>
<tr>
<td>tx_refclk_stable</td>
<td>In</td>
<td>1</td>
<td>This input is used by the TX initialization logic to keep the PLL that is providing the serial clock to the GTH TX in reset until the PLL reference clock is stable. If this SDI wrapper is controlling the PLL reset, then this input must be kept Low until the PLL reference clock is stable. This input does not initiate a PLL reset. It only delays completion of the PLL reset sequence initiated by the tx_gth_full_reset input until the tx_refclk_stable input is High. This input is treated as an asynchronous input.</td>
</tr>
<tr>
<td>tx_ce</td>
<td>In</td>
<td>3</td>
<td>This is the clock enable input for the transmitter section of the SDI core. tx_ce must always be High in HD-SDI and 3G-SDI modes. In SD-SDI mode, tx_ce must be asserted at a 27 MHz rate with a mandatory 5/6/5/6 clock cycle cadence. Three identical copies of the clock enable signal must be provided on the three bits of this port. Three input bits are provided to make it easier to meet timing. If these three inputs are all driven by the same flip-flop, the loading on the single clock enable signal might be too high to meet timing. In those cases, duplicate copies of the clock enable signal can be created by multiple flip-flops each driving a different bit of the tx_ce input port. In SD-SDI, HD-SDI, and 3G-SDI level A modes, this input must be kept High at all times. In 3G-SDI level B mode, this input must be asserted every other clock cycle.</td>
</tr>
<tr>
<td>tx_din_rdy</td>
<td>In</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
Implementing an SDI Interface in Virtex-7

Table 1: SDI Wrapper Port List (Cont’d)

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tx_mode</td>
<td>In</td>
<td>2</td>
<td>This input port is used to select the SDI transmitter mode:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 00 = HD-SDI (including dual link HD-SDI)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 01 = SD-SDI</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 10 = 3G-SDI</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 11 = Invalid</td>
</tr>
<tr>
<td>tx_level_b_3g</td>
<td>In</td>
<td>1</td>
<td>In 3G-SDI mode, this input determines whether the SDI transmitter is configured for level A (Low) or for level B (High).</td>
</tr>
<tr>
<td>tx_m</td>
<td>In</td>
<td>1</td>
<td>This port is used to select which PLL serial clock is used by the GTH TX. This input causes the SDI wrapper gth_txsysclksel output port to change the GTH transceiver TX PLL clock select MUX.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>By convention, when tx_m is Low, the 1/1 bit rates are selected and when tx_m is High, the 1/1.001 bit rates are selected. However, this distinction is entire governed by the frequency of the QPLL and the PLL and by the TX_CLK0_QPLL and TX_CLK1_QPLL parameters. More information is provided in the descriptions of the TX_CLK0_QPLL and TX_CLK1_QPLL parameters in Table 2 and in the discussion of these parameters immediately after Table 2.</td>
</tr>
<tr>
<td>tx_insert_crc</td>
<td>In</td>
<td>1</td>
<td>When this input is High, the SDI TX generates and inserts CRC values on each video line in HD-SDI and 3G-SDI modes. When this input is Low, CRC values are not generated and inserted. This input is ignored in SD-SDI mode. CRC values are required by both the HD-SDI and 3G-SDI standards. If the data streams entering the SDI TX input ports do not have CRC values, this input should be asserted High. If the data streams entering the SDI TX input ports already have CRC values, the existing CRC values are replaced by newly calculated CRC values if tx_insert_crc is High or passed through unchanged if tx_insert_crc is Low.</td>
</tr>
<tr>
<td>tx_insert_ln</td>
<td>In</td>
<td>1</td>
<td>When this input is High, the SDI TX inserts line numbers words after the EAV in each video line. The line number must be supplied on the tx_line_a and tx_line_b input ports. This input is ignored in SD-SDI mode. Line numbers are required by both the HD-SDI and 3G-SDI standards. If the data streams entering the SDI TX input ports do not have line number words already integrated, this input should be asserted High and valid line numbers must be supplied on the tx_line_a and tx_line_b ports. If the data streams entering the SDI TX input ports already have line numbers integrated, those line numbers are overwritten if tx_insert_ln is High or passed through unchanged if tx_insert_ln is Low.</td>
</tr>
<tr>
<td>tx_insert_edh</td>
<td>In</td>
<td>1</td>
<td>When this input is High, the SDI TX generates and inserts EDH packets in every field in SD-SDI mode. When this input is Low, EDH packets are not inserted. This input is ignored in HD-SDI and 3G-SDI modes. EDH packets are optional but commonly used in SD-SDI mode and are never used in HD-SDI and 3G-SDI modes. If the SD-SDI data stream entering the SDI TX already has an EDH packet integrated, it is overwritten with a new packet if tx_insert_edh is High or passed through unchanged if tx_insert_edh is Low.</td>
</tr>
<tr>
<td>tx_insert_vpid</td>
<td>In</td>
<td>1</td>
<td>When this input is High, SMPTE ST 352 packets are inserted into the TX data streams, otherwise the packets are not inserted. ST 352 packets are mandatory in 3G-SDI and dual link HD-SDI modes and optional in HD-SDI and SD-SDI modes.</td>
</tr>
<tr>
<td>tx_overwrite_vpid</td>
<td>In</td>
<td>1</td>
<td>If this input is High and the tx_insert_vpid port is High, SMPTE ST 352 packets already present in the TX data streams are overwritten with new ST 352 packets. If this input is Low, existing ST 352 packets are not overwritten, even if the tx_insert_vpid port is High.</td>
</tr>
</tbody>
</table>
This is the SDI data stream A Y input to the SDI TX. The data on this port depends on the SDI mode:
- SD-SDI: Multiplexed Y/C data stream
- HD-SDI: Y component
- 3G-SDI level A: Data stream 1
- Dual link HD-SDI or 3G-SDI level B-DL: Data stream 1 of link A
- 3G-SDI level B-DS: Y component of HD-SDI signal 1

This is the SDI data stream A C input to the SDI TX. The data on this port depends on the SDI mode:
- SD-SDI: Unused
- HD-SDI: Interleaved CB and CR components
- 3G-SDI level A: Data stream 2
- Dual link HD-SDI or 3G-SDI level B-DL: Data stream 2 of link A
- 3G-SDI level B-DS: Interleaved CB and CR components of HD-SDI signal 1

This is the SDI data stream B Y input to the SDI TX. The data stream on this port depends on the SDI mode:
- Dual link HD-SDI or 3G-SDI level B-DL: Data stream 1 of link B
- 3G-SDI level B-DS: Y component of HD-SDI signal 2
- For other SDI modes, this input port is unused.

This is the SDI data stream B C input to the SDI TX. The data stream on this port depends on the SDI mode:
- Dual link HD-SDI or 3G-SDI level B-DL: Data stream 2 of link B
- 3G-SDI level B-DS: Interleaved CB and CR components of HD-SDI signal 2
- For other SDI modes, this input port is unused.

The current line number must be provided to the module through this port if either ST 352 VPID packet insertion is enabled (tx_insert_vpid = High) or if HD-SDI and 3G-SDI line number insertion is enabled (tx_insert_ln = High).

SD-SDI only uses 10-bit line numbers, so bit 10 of this port must be 0 in SD-SDI mode if ST 352 payload ID packet insertion is enabled in SD-SDI mode. Line number insertion is never done in SD-SDI mode so this input port is only used for ST 352 payload ID packet insertion in SD-SDI mode.

The value on this port must be valid at least one clock cycle before the start of the horizontal ancillary (HANC) space (by the XYZ word of the EAV) and must remain valid during the entire HANC interval.

This input is the only line number input used for SD-SDI, HD-SDI, and 3G-SDI level A modes. For 3G-SDI level B mode, a second line number input port, tx_line_b, is also provided.

For video formats where the picture line number is different than the transport line number, the value supplied on this port must be the transport line number.

This is the second line number input port and is used only for 3G-SDI level B mode. This additional line number port allows the two separate HD-SDI signals to be vertically unsynchronized in level B-DS mode. When using either 3G-SDI level B-DL or B-DS, this port must be given a valid line number input. In 3G-SDI level B-DL mode, the value on this input port must be the same as the value on the tx_line_a port. This input port has the same timing and other requirements described for tx_line_a.

The value on this port is inserted as the first user data word of the ST 352 packet. It must be valid during the entire HANC interval of the lines that are to contain the ST 352 packet if ST 352 packets are being inserted or overwritten.

The value on this port is inserted as the second user data word of the ST 352 packet. It must be valid during the entire HANC interval of the lines that are to contain the ST 352 packet if ST 352 packets are being inserted or overwritten.
The value on this port is inserted as the third user data word of the ST 352 packet. It must be valid during the entire HANC interval of the lines that are to contain the ST 352 packet if ST 352 packets are being inserted or overwritten.

The value on this port is inserted as the fourth user data word of the ST 352 packet. This word is used for the ST 352 packets inserted into SD-SDI, HD-SDI, and 3G-SDI level A data streams. For 3G-SDI level B and dual link HD-SDI modes, this value is used for the ST 352 packet inserted into data stream 1 of link A only. This input must be valid during the entire HANC interval of the lines that are to contain the ST 352 packet if ST 352 packets are being inserted or overwritten. Separate values are allowed for byte 4 for link A and link B because this byte contains the link ID bits which must be different on link A than on link B in 3G-SDI level B-DL mode.

This value on this port is inserted as the fourth user data word of ST 352 packets inserted in the data stream 1 of link B for 3G-SDI level B and dual link HD-SDI modes only. This input value is not used for SD-SDI, HD-SDI, or 3G-SDI level A modes. This input must be valid during the entire HANC interval of the lines that are to contain the ST 352 packet if ST 352 packets are being inserted or overwritten.

The ST 352 packet is inserted in the HANC space of the line number specified by this input port. For an interlaced transport, this input port specifies a line number in field 1. For a progressive transport, this specifies the only line in the frame where the packet is inserted. The input value must be valid during the entire HANC interval. If tx_insert_vpid is low, this input is ignored.

For an interlaced transport, a ST 352 packet is inserted on the line number in field 2 indicated by this value. For a progressive transport, insertion of ST 352 packets on the line specified by this input port must be disabled by holding the tx_vpid_line_f2_en port Low. The input value must be valid during the entire HANC interval. This input is ignored if either tx_insert_vpid or tx_vpid_line_f2_en are Low.

This input controls whether or not ST 352 packets are inserted on the line indicated by tx_vpid_line_f2. For an interlaced transport, this input must be High. For a progressive transport, this input must be Low.

This is the link A data stream 1 output. The data stream output on this port comes from the ST 352 packet insertion module. If the application needs to insert ancillary data packets, the packets should be inserted into the data stream output on this port. This allows the higher priority ST 352 packets to be inserted into the data streams before any other ancillary data. The resulting data stream after ancillary data insertion by the application should then be supplied to the tx_ds1a_in port.

The data on this port depends on the SDI mode:
- SD-SDI: Interleaved Y/C data stream
- HD-SDI: Y component
- 3G-SDI level A: Data stream 1
- Dual link HD-SDI or 3G-SDI level B-DL: Data stream 1 of link A
- 3G-SDI level B-DS: Y component of HD-SDI signal 1

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tx_vpid_byte3</td>
<td>In</td>
<td>8</td>
<td>The value on this port is inserted as the third user data word of the ST 352 packet. It must be valid during the entire HANC interval of the lines that are to contain the ST 352 packet if ST 352 packets are being inserted or overwritten.</td>
</tr>
<tr>
<td>tx_vpid_byte4a</td>
<td>In</td>
<td>8</td>
<td>The value on this port is inserted as the fourth user data word of the ST 352 packet. This word is used for the ST 352 packets inserted into SD-SDI, HD-SDI, and 3G-SDI level A data streams. For 3G-SDI level B and dual link HD-SDI modes, this value is used for the ST 352 packet inserted into data stream 1 of link A only. This input must be valid during the entire HANC interval of the lines that are to contain the ST 352 packet if ST 352 packets are being inserted or overwritten. Separate values are allowed for byte 4 for link A and link B because this byte contains the link ID bits which must be different on link A than on link B in 3G-SDI level B-DL mode.</td>
</tr>
<tr>
<td>tx_vpid_byte4b</td>
<td>In</td>
<td>8</td>
<td>This value on this port is inserted as the fourth user data word of ST 352 packets inserted in the data stream 1 of link B for 3G-SDI level B and dual link HD-SDI modes only. This input value is not used for SD-SDI, HD-SDI, or 3G-SDI level A modes. This input must be valid during the entire HANC interval of the lines that are to contain the ST 352 packet if ST 352 packets are being inserted or overwritten.</td>
</tr>
<tr>
<td>tx_vpid_line_f1</td>
<td>In</td>
<td>11</td>
<td>The ST 352 packet is inserted in the HANC space of the line number specified by this input port. For an interlaced transport, this input port specifies a line number in field 1. For a progressive transport, this specifies the only line in the frame where the packet is inserted. The input value must be valid during the entire HANC interval. If tx_insert_vpid is low, this input is ignored.</td>
</tr>
<tr>
<td>tx_vpid_line_f2</td>
<td>In</td>
<td>11</td>
<td>For an interlaced transport, a ST 352 packet is inserted on the line number in field 2 indicated by this value. For a progressive transport, insertion of ST 352 packets on the line specified by this input port must be disabled by holding the tx_vpid_line_f2_en port Low. The input value must be valid during the entire HANC interval. This input is ignored if either tx_insert_vpid or tx_vpid_line_f2_en are Low.</td>
</tr>
<tr>
<td>tx_vpid_line_f2_en</td>
<td>In</td>
<td>1</td>
<td>This input controls whether or not ST 352 packets are inserted on the line indicated by tx_vpid_line_f2. For an interlaced transport, this input must be High. For a progressive transport, this input must be Low. For progressive video transported on an interlaced transport, such as 1080p 50 Hz transported by either 3G-SDI level B-DL or dual link HD-SDI, ST 352 packets must be inserted into both fields of the interlaced transport, so this input must be High in these cases. This input must be valid during the entire HANC interval. This input is ignored if tx_insert_vpid is Low.</td>
</tr>
<tr>
<td>tx_ds1a_out</td>
<td>Out</td>
<td>10</td>
<td>This is the link A data stream 1 output. The data stream output on this port comes from the ST 352 packet insertion module. If the application needs to insert ancillary data packets, the packets should be inserted into the data stream output on this port. This allows the higher priority ST 352 packets to be inserted into the data streams before any other ancillary data. The resulting data stream after ancillary data insertion by the application should then be supplied to the tx_ds1a_in port. The data on this port depends on the SDI mode:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- SD-SDI: Interleaved Y/C data stream</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- HD-SDI: Y component</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- 3G-SDI level A: Data stream 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Dual link HD-SDI or 3G-SDI level B-DL: Data stream 1 of link A</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- 3G-SDI level B-DS: Y component of HD-SDI signal 1</td>
</tr>
</tbody>
</table>
The data on this port depends on the SDI mode:

- SD-SDI: Unused
- HD-SDI: Interleaved CB/CR component
- Dual link HD-SDI or 3G-SDI level B-DL: Data stream 2 of link A
- 3G-SDI level B-DS: Interleaved CB/CR component data stream of HD-SDI signal 1

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
<th>Description</th>
</tr>
</thead>
</table>
| tx_ds1b_out | Out | 10    | This is the link B data stream 1 output. The data stream output on this port comes from the ST 352 packet insertion module. If the application needs to insert ancillary data packets, the packets should be inserted into the data stream output on this port. This allows the higher priority ST 352 packets to be inserted into the data streams before any other ancillary data. The resulting data stream after ancillary data insertion by the application should then be supplied to the tx_ds1b_in port. The data on this port depends on the SDI mode:  
  - Dual link HD-SDI or 3G-SDI level B-DL: Data stream 1 of link B  
  - 3G-SDI level B-DS: Y component of HD-SDI signal 2  
  - 3G-SDI level B-DS: Y component of HD-SDI signal 2  
  - For other SDI modes, this input port is unused |
| tx_ds2b_out | Out | 10    | This is the link B data stream 2 output. The data stream output on this port comes from the ST 352 packet insertion module. If the application needs to insert ancillary data packets, the packets should be inserted into the data stream output on this port. This allows the higher priority ST 352 packets to be inserted into the data streams before any other ancillary data. The resulting data stream after ancillary data insertion by the application should then be supplied to the tx_ds2b_in port.  
  - Dual link HD-SDI or 3G-SDI level B-DL: Data stream 2 of link A  
  - 3G-SDI level B-DS: Interleaved CB/CR component of HD-SDI signal 2  
  - For other SDI modes, this input port is unused |
| tx_use_dsin | In  | 1     | This input controls the source of the data streams sent by the SDI TX. When this input is High, the sources of the transmitted data streams are the tx_ds1a_in, tx_ds2a_in, tx_ds1b_in, and tx_ds2b_in input ports. When this input is Low, the source of the transmitted data streams are internal to the core, coming directly from the ST 352 packet inserter. When the application needs to do ancillary data insertion, the tx_use_dsin port is set High to allow the application to modify the data streams and provide the modified data streams to the transmitter on the tx_dxx_in ports. When no ancillary data insertion is required, the tx_use_dsin input is set Low and the tx_dxx_in ports are ignored. |
| tx_ds1a_in  | In  | 10    | This is the link A data stream 1 input. This port is ignored if tx_use_dsin is Low. If tx_use_dsin is High, this port must supply a data stream to be transmitted. The data stream supplied input to this port depends on the SDI mode:  
  - SD-SDI: Interleaved Y/C data stream  
  - HD-SDI: Y component  
  - 3G-SDI level A: Data stream 1  
  - Dual link HD-SDI or 3G-SDI level B-DL: Data stream 1 of link A  
  - 3G-SDI level B-DS: Y component of HD-SDI signal 1 |
Implementing an SDI Interface in Virtex-7

Table 1: SDI Wrapper Port List (Cont’d)

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
<th>Description</th>
</tr>
</thead>
</table>
| tx_ds2a_in      | In  | 10    | This is the link A data stream 2 input. This port is ignored if tx_use_dsin is Low. If tx_use_dsin is High, this port must supply a data stream to be transmitted. The data stream input to this port depends on the SDI mode:  
  - SD-SDI: Unused  
  - HD-SDI: Interleaved CB/CR component  
  - Dual link HD-SDI or 3G-SDI level B-DL: Data stream 2 of link A  
  - 3G-SDI level B-DS: Interleaved CB/CR component data stream of HD-SDI signal 1 |
| tx_ds1b_in      | In  | 10    | This is the link B data stream 1 input. This port is ignored if tx_use_dsin is Low. If tx_use_dsin is High, this port must supply a data stream to be transmitted. The data stream input to this port depends on the SDI mode:  
  - Dual link HD-SDI or 3G-SDI level B-DL: Data stream 1 of link B  
  - 3G-SDI level B-DS: Y component of HD-SDI signal 2  
  - For other SDI modes, this input port is unused |
| tx_ds2b_in      | In  | 10    | This is the link B data stream 2 input. This port is ignored if tx_use_dsin is Low. If tx_use_dsin is High, this port must supply a data stream to be transmitted. The data stream input to this port depends on the SDI mode:  
  - Dual link HD-SDI or 3G-SDI level B-DL: Data stream 2 of link B  
  - 3G-SDI level B-DS: Interleaved CB/CR component of HD-SDI signal 2  
  - For other SDI modes, this input port is unused |
| tx_ce_align_err | Out | 1     | This output indicates problems with the 5/6/5/6 clock cycle cadence on the tx_ce clock enable inputs in SD-SDI mode. In SD-SDI mode, the tx_ce signals must follow a regular 5/6/5/6 clock cycle cadence. If they do not, the SD-SDI bitstream forms incorrectly. The tx_ce_align_err will go High if the cadence is incorrect. This port is only valid in SD-SDI mode. |
| tx_slew         | Out | 1     | This output is designed to control the slew rate signal of the external SDI cable equalizer. It is High when the TX mode is SD-SDI, otherwise it is Low. |
| tx_change_done  | Out | 1     | This output is Low during those periods when the GTH TX is being initialized or reset or the GTH txrate or txsyscksel ports are being dynamically changed. If the sequence completes successfully, the tx_change_done output is asserted High to indicate successful completion.  
  This output is synchronous with the gth_drpclk. |
| tx_change_fail  | Out | 1     | Under normal conditions, this output is always Low. It only goes High if the control module is unsuccessful in completing a GTH TX initialization or reset sequence or a dynamic change of the GTH txrate or txsyscksel ports. If such a failure occurs, the tx_change_fail port is asserted High and the tx_change_fail_code port indicates the nature of the failure.  
  If a failure occurs as indicated by tx_change_fail going High, a full reset must be done using the tx_gth_full_reset input.  
  This output is synchronous with the gth_drpclk. |
| tx_change_fail_code | Out | 3     | When the tx_change_fail port is High, this port indicates the nature of the failure. See Table 9 for encoding of this port.  
  This output is synchronous with the gth_drpclk. |

Ports That Connect to the GTH RX

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gth_rxdata</td>
<td>In</td>
<td>20</td>
<td>Connect this port to the rxdata port of the GTH transceiver.</td>
</tr>
<tr>
<td>gtp_rxplreset</td>
<td>In</td>
<td>1</td>
<td>This port is used to reset the PLL supplying the serial clock to the GTH RX. If this SDI wrapper is acting as the PLL master, connect this output to the opplreset input of the GTH common wrapper or the opplreset input of the GTH wrapper depending on which PLL is supplying the RX serial clock. See GTH PLL Usage Models for SDI Applications for more details.</td>
</tr>
</tbody>
</table>
Implementing an SDI Interface in Virtex-7

**Table 1: SDI Wrapper Port List (Cont’d)**

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gth_rxplllock</td>
<td>In</td>
<td>1</td>
<td>Connect this port to the PLL lock signal of the PLL supplying the clock to the GTH RX, either the qplllock output of the GTH common wrapper or the cplllock output of the GTH wrapper. See GTH PLL Usage Models for SDI Applications for more details.</td>
</tr>
<tr>
<td>gth_rxresetdone</td>
<td>In</td>
<td>1</td>
<td>Connect this port to the rxresetdone port of the GTH wrapper.</td>
</tr>
<tr>
<td>gth_gtrxreset</td>
<td>Out</td>
<td>1</td>
<td>Connect this port to the gtrxreset port of the GTH wrapper.</td>
</tr>
<tr>
<td>gth_rxuserrdy</td>
<td>Out</td>
<td>1</td>
<td>Connect this port to the rxuserrdy port of the GTH wrapper.</td>
</tr>
<tr>
<td>gth_rxcdrhod</td>
<td>Out</td>
<td>1</td>
<td>Connect this port to the rxcdrhod port of the GTH wrapper.</td>
</tr>
<tr>
<td>gth_drpclk</td>
<td>In</td>
<td>1</td>
<td>Connect this port to the clock that is driving the drpclk port of the GTH wrapper.</td>
</tr>
<tr>
<td>gth_drprd</td>
<td>In</td>
<td>1</td>
<td>Connect this port to the drpclk port of the GTH wrapper.</td>
</tr>
<tr>
<td>gth_drpbuss</td>
<td>In</td>
<td>1</td>
<td>Connect this port to the drp_busy port of the GTH wrapper.</td>
</tr>
<tr>
<td>gth_drpadr</td>
<td>Out</td>
<td>10</td>
<td>Connect this port to the drpadr port of the GTH wrapper.</td>
</tr>
<tr>
<td>gth_drdi</td>
<td>Out</td>
<td>16</td>
<td>Connect this port to the drdi port of the GTH wrapper.</td>
</tr>
<tr>
<td>gth_drpdo</td>
<td>In</td>
<td>16</td>
<td>Connect this port to the drpdo port of the GTH wrapper.</td>
</tr>
<tr>
<td>gth_drepn</td>
<td>Out</td>
<td>1</td>
<td>Connect this port to the drepn port of the GTH wrapper.</td>
</tr>
<tr>
<td>gth_drpee</td>
<td>Out</td>
<td>1</td>
<td>Connect this port to the drpwe port of the GTH wrapper.</td>
</tr>
</tbody>
</table>

**Ports That Connect to the GTH TX**

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I/O</th>
<th>Width</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gth_txdata</td>
<td>In</td>
<td>20</td>
<td>Connect this port to the txdata port of the GTH wrapper.</td>
</tr>
<tr>
<td>gth_txpllreset</td>
<td>In</td>
<td>1</td>
<td>This port is used to reset the PLL supplying the serial clock to the GTH TX. If this SDI wrapper is acting as the PLL master, connect this output to the opllreset input of the GTH common wrapper or the cplllReset input of the GTH wrapper depending on which PLL is supplying the serial clock to the GTH TX. If the GTH is dynamically switched between the QPLL and the CPLL, then typically the gth_txpllreset is connected to the opllreset input of the GTH common wrapper and gth_txpllreset is connected to the cplllReset input of the GTH wrapper. See GTH PLL Usage Models for SDI Applications for more details.</td>
</tr>
<tr>
<td>gth_txplllock</td>
<td>In</td>
<td>1</td>
<td>Connect this port to the PLL lock signal of the PLL supplying the clock to the GTH TX, either the oplllock output of the GTH common wrapper or the cpllllock output of the GTH wrapper. If the GTH TX is dynamically switched between the QPLL and the CPLL, this input must be driven by the logical OR of the oplllock and cpllllock signals. See GTH PLL Usage Models for SDI Applications for more details.</td>
</tr>
<tr>
<td>gth_gtrxreset</td>
<td>Out</td>
<td>1</td>
<td>Connect this port to the gtrxreset port of the GTH wrapper.</td>
</tr>
<tr>
<td>gth_txresetdone</td>
<td>In</td>
<td>1</td>
<td>Connect this port to the txresetdone port of the GTH wrapper.</td>
</tr>
<tr>
<td>gth_txbratedone</td>
<td>In</td>
<td>1</td>
<td>Connect this port to the txbratedone port of the GTH wrapper.</td>
</tr>
<tr>
<td>gth_txuserrdy</td>
<td>Out</td>
<td>1</td>
<td>Connect this port to the txuserrdy port of the GTH wrapper.</td>
</tr>
<tr>
<td>gth_txrate</td>
<td>Out</td>
<td>3</td>
<td>Connect this port to the txrate port of the GTH wrapper.</td>
</tr>
<tr>
<td>gth_txsysclksel</td>
<td>Out</td>
<td>2</td>
<td>If the clock source of the GTH TX needs to be dynamically switched between the two PLLs, connect this port to the txsysclksel port of the GTH wrapper.</td>
</tr>
</tbody>
</table>

**Notes:**
1. The RX ports related to the EDH processor are not present on the SMPTE core when the core is generated without the RX EDH processor, an option allowed in the SDI core GUI. If the RX EDH processor is not included in the SDI core, the v7gth_sdi_rxtx_wrapper.v SDI wrapper file should not be used as it has all the ports to support the RX EDH processor. Instead, the v7gth_sdi_rxtx_noedh_wrapper.v SDI wrapper file should be used.
Table 2 lists the parameters that can be applied to the SDI wrapper.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FXDCLK_FREQ</td>
<td>Integer</td>
<td>27,000,000</td>
<td>This parameter specifies the frequency, in Hz, of the fixed frequency clock on the clk port of the SDI wrapper. The nominal frequency of this clock must be correctly specified so that the portions of the control module that depend on this clock for timing purposes will function correctly.</td>
</tr>
<tr>
<td>DRPCLK_PERIOD</td>
<td>integer</td>
<td>37</td>
<td>This parameter specifies the period, (in ns) of the clock that is driving the GTH wrapper drpclk port and the SDI wrapper gth_drpclk port. Round all non-integer values down to the nearest integer. The nominal period of this clock must be specified correctly so that the control module can generate delays in the GTH initialization sequences based on the period of this clock.</td>
</tr>
<tr>
<td>PLLLOCK_TIMEOUT_PERIOD</td>
<td>integer</td>
<td>2,000,000</td>
<td>This parameter specifies the duration of the PLL lock timeout period in ns. If, after being reset, a PLL does not assert its lock signal within this period of time, the control module times out and retries the PLL reset sequence. The default value is equivalent to 2 ms.</td>
</tr>
<tr>
<td>RESET_TIMEOUT_PERIOD</td>
<td>Integer</td>
<td>500,000</td>
<td>This parameter specifies the duration of the GTH transceiver reset timeout period in ns. If, after being reset, the GTH transceiver does not assert its rxresetdone or txresetdone within this period of time, the control module times out and retries the GTH transceiver reset sequence. The default value is equivalent to 500 μs.</td>
</tr>
<tr>
<td>TIMEOUT_CNTR_BITWIDTH</td>
<td>Integer</td>
<td>16</td>
<td>This parameter specifies the width in bits of the timeout counter used to during the PLL and GTH transceiver reset sequences. The width of this counter must be sufficient to count up to the maximum timeout periods specified by PLLLOCK_TIMEOUT_PERIOD and RESET_TIMEOUT_PERIOD based on the period specified by DRPCLK_PERIOD. For example, the default value of 16 bits is sufficient for timeout periods of up to about 2.4 ms with the default DRPCLK_PERIOD of 37 which is larger than the default values of both PLLLOCK_TIMEOUT_PERIOD and RESET_TIMEOUT_PERIOD.</td>
</tr>
<tr>
<td>RETRY_CNTR_BITWIDTH</td>
<td>Integer</td>
<td>8</td>
<td>This parameter specifies the width in bits of the retry counter. The retry counter counts the number of retry cycles attempted to complete a GTH RX or TX initialization or reset sequence or a dynamic change of the GTH transceiver txrate or txsysclksel ports. If the retry counter reaches its maximum value of all ones, the sequence is considered to have failed. Thus, this parameter specifies the number of retries that are permitted before the control module gives up on the sequence. The default value of 8 allows for 255 retry cycles.</td>
</tr>
<tr>
<td>TX_CLK0_QPLL</td>
<td>Integer</td>
<td>1</td>
<td>If the QPLL is the serial clock source to the GTH TX when the tx_m port is Low, this parameter must be assigned a value of 1. If the CPLL is the serial clock source to the GTH TX when the tx_m port is Low, this parameter must be assigned a value of 0.</td>
</tr>
<tr>
<td>TX_CLK1_QPLL</td>
<td>Integer</td>
<td>0</td>
<td>If the QPLL is the serial clock source to the GTH TX when the tx_m port is High, this parameter must be assigned a value of 1. If the CPLL is the serial clock source to the GTH TX when the tx_m port is High, this parameter must be assigned a value of 0.</td>
</tr>
<tr>
<td>RX_CLK_QPLL</td>
<td>Integer</td>
<td>1</td>
<td>If the QPLL is the serial clock source to the GTH RX, this parameter must be assigned a value of 1. If the CPLL is the serial clock source to the GTH RX, this parameter must be assigned a value of 0.</td>
</tr>
</tbody>
</table>

The TX_CLK0_QPLL and TX_CLK1_QPLL parameters, along with the tx_m port of the SDI wrapper work together to control the GTH transceiver txsysclksel port when dynamic switching of the GTH TX serial clock source is required. The txsysclksel port must be driven with a value...
of 2'b00 to select the CPLL and 2'b11 to select the QPLL. When tx_m is Low, if TX_CLK0_QPLL is 0, then txsysclksel is driven with a value of 2'b00. But, if TX_CLK0_QPLL is 1, then txsysclksel is driven with a value of 2'b11. Likewise, TX_CLK1_QPLL determines the value driven onto txsysclksel when tx_m is High.

The TX_CLK0_QPLL and TX_CLK1_QPLL parameters also determine the TX serial clock divider value used. Different divider values are used depending on whether the QPLL or the CPLL is the TX serial clock source. The TX serial clock divider also depends on which SDI mode the TX is currently running.

The RX_CLK_QPLL parameter is used to determine proper serial clock divider value based on whether the QPLL or the CPLL is the RX serial clock source. The RX serial clock divider also depends on which SDI mode the RX is currently running.

The TX_CLK0_QPLL and TX_CLK1_QPLL parameters are static and cannot be changed dynamically. Two parameters are provided for the TX so that the TX serial clock source can be switched dynamically. The RX_CLK_QPLL parameter is also static, but only one such parameter is specified because SDI applications do not dynamically switch the RX serial clock source.

### Video Transport Detector Ports

The RX section of the SDI core has a SDI transport format detector. This function examines the timing of the video transport in the SDI data streams and determines which video format is being received. The operation of this function is not dependent on the presence of ST 352 payload ID packets. This function determines the transport format, not the picture format. Usually these are the same, but not always. For example, when 1080p 50 Hz video is transported on 3G-SDI level B-DL, the video transport is actually 1080i 50 Hz – the transport is interlaced, but the picture is progressive.

The rx_t_family output port provides a 4-bit code indicating which video format family of the transport in the SDI signal. The encoding of this output port is shown in Table 3. The transport detection unit also determines whether the SDI transport is interlaced or progressive and reports this on the rx_t_scan output port.

<table>
<thead>
<tr>
<th>rx_t_family</th>
<th>Transport Video Format</th>
<th>Active Pixels</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>SMPTE ST 274</td>
<td>1920 x 1080</td>
</tr>
<tr>
<td>0001</td>
<td>SMPTE ST 296</td>
<td>1280 x 720</td>
</tr>
<tr>
<td>0010</td>
<td>SMPTE 2048-2</td>
<td>2048 x 1080</td>
</tr>
<tr>
<td>0011</td>
<td>SMPTE 295</td>
<td>1920 x 1080</td>
</tr>
<tr>
<td>1000</td>
<td>NTSC</td>
<td>720 x 486</td>
</tr>
<tr>
<td>1001</td>
<td>PAL</td>
<td>720 x 576</td>
</tr>
<tr>
<td>1111</td>
<td>Unknown</td>
<td>-</td>
</tr>
<tr>
<td>Others</td>
<td>Reserved</td>
<td>-</td>
</tr>
</tbody>
</table>

The transport detector also determines the frame rate of the transport in the SDI signal. The rx_t_rate port indicates the frame rate of the transport signal as shown in Table 4. The encoding of the frame rate matches the encoding used is the picture rate field of SMPTE ST 352 video...
payload ID packets. However, the `rx_t_rate` shows the transport frame rate, not the picture rate. The `rx_t_rate` port value is always the frame rate, even for interlaced transports.

**Table 4: rx_t_rate Encoding**

<table>
<thead>
<tr>
<th>rx_t_rate</th>
<th>Frame Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>None</td>
</tr>
<tr>
<td>0010</td>
<td>23.98 Hz</td>
</tr>
<tr>
<td>0011</td>
<td>24 Hz</td>
</tr>
<tr>
<td>0100</td>
<td>47.95 Hz</td>
</tr>
<tr>
<td>0101</td>
<td>25 Hz</td>
</tr>
<tr>
<td>0110</td>
<td>29.97 Hz</td>
</tr>
<tr>
<td>0111</td>
<td>30 Hz</td>
</tr>
<tr>
<td>1000</td>
<td>48 Hz</td>
</tr>
<tr>
<td>1001</td>
<td>50 Hz</td>
</tr>
<tr>
<td>1010</td>
<td>59.94 Hz</td>
</tr>
<tr>
<td>1011</td>
<td>60 Hz</td>
</tr>
<tr>
<td>Others</td>
<td>Reserved</td>
</tr>
</tbody>
</table>

**Note:** It can take the transport format detector up to two video frames to identify the transport format after the SDI RX locks to the SDI signal.

**SD-SDI RX EDH Processor**

Optionally, the SDI receiver can include an EDH processor for detecting receiver errors in SD-SDI mode. The EDH processor does not update EDH packets in the SD-SDI data stream. It reports any errors found and also captures the error flags from each EDH packet.

The EDH processor has a 16-bit counter that counts the number of fields with errors. The current error count is output on the `rx_edh_errcnt` port of the SDI wrapper. The counter is cleared by asserted `rx_edh_clr_errcnt` High. You can specify which types of errors are counted by this counter using the `rx_edh_errcnt_en` port. This port has 16 unary bits that enable and disable 16 different error types. Any bit that is High enables the corresponding error to be counted by the error counter. Any bit that is Low disables the corresponding error. If multiple errors occur in the same field, the EDH error counter only increments by one. **Table 5** shows the encoding of the bits on the `rx_edh_errcnt_en` port.

**Table 5: rx_edh_errcnt_en Bits**

<table>
<thead>
<tr>
<th>Bit #</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>ANC EDH error(^{(1)})</td>
</tr>
<tr>
<td>1</td>
<td>ANC EDA error(^{(1)})</td>
</tr>
<tr>
<td>2</td>
<td>ANC IDH error(^{(1)})</td>
</tr>
<tr>
<td>3</td>
<td>ANC IDA error(^{(1)})</td>
</tr>
<tr>
<td>4</td>
<td>ANC UES error(^{(1)})</td>
</tr>
<tr>
<td>5</td>
<td>FF EDH error(^{(2)})</td>
</tr>
<tr>
<td>6</td>
<td>FF EDA error(^{(2)})</td>
</tr>
<tr>
<td>7</td>
<td>FF IDH error(^{(2)})</td>
</tr>
<tr>
<td>8</td>
<td>FF IDA error(^{(2)})</td>
</tr>
</tbody>
</table>
Each ANC, FF, and AP error condition set has five individual error flags. All flags are asserted High to indicate an error condition. For a complete description of the EDH, EDA, IDH, IDA, and UES error flags in the EDH packet, refer to the SMPTE RP 165 document [Ref 5].

- **EDH error**: This error condition occurs when the EDH processor detects a CRC error (checksum error for ANC packets) in a field. For example, the FF EDH error flag indicates an error was detected by the full field CRC.

- **EDA error**: This error condition occurs when the EDA or EDH flags of the received EDH packet are asserted.

- **IDH error**: This error condition is not supported by the RX EDH processor.

- **IDA error**: This error condition occurs when the IDA or IDH flags of the received EDH packet are asserted.

- **UES error**: This error condition occurs when the UES flag in the received EDH packet is asserted.

Besides being counted, if enabled by the error counter, any detected ANC EDH, AP EDH, and FF EDH errors are also indicated by assertion of the rx_edh_anc, rx_edh_ap, and rx_edh_ff ports, respectively. Thus, the rx_edh_anc port is asserted whenever a checksum error is detected in an ancillary data packet. The rx_edh_ap port is asserted when the calculated active picture CRC does not match the AP CRC in the EDH packet. And, the rx_edh_ff port is asserted when the calculated full field CRC does not match the FF CRC in the EDH packet.

The RX EDH processor also outputs the ANC, AP, and FF error flags from the EDH packet on the rx_edh_anc_flags, rx_edh_ap_flags, and rx_edh_ff_flags ports, respectively. These output ports are exact copies of the flags found in the last received EDH packet. Thus, they differ from the detected errors used to increment the error counter and output on the rx_edh_anc, rx_edh_ap, and rx_edh_ff ports. For example, the EDH flag (bit 0) of the rx_edh_ap_flags port indicates that the AP EDH flag was set in the last received EDH packet. However, the rx_edh_ap port indicates that the active picture CRC calculated locally by the EDH processor does not match the AP CRC value in the EDH packet. The rx_edh_anc_flags, rx_edh_ap_flags, and rx_edh_ff_flags ports are each five bits wide; the encoding of all three

<table>
<thead>
<tr>
<th>Bit #</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>FF UES error(^{(2)})</td>
</tr>
<tr>
<td>10</td>
<td>AP EDH error(^{(3)})</td>
</tr>
<tr>
<td>11</td>
<td>AP EDA error(^{(3)})</td>
</tr>
<tr>
<td>12</td>
<td>AP IDH error(^{(3)})</td>
</tr>
<tr>
<td>13</td>
<td>AP IDA error(^{(3)})</td>
</tr>
<tr>
<td>14</td>
<td>AP UES error(^{(3)})</td>
</tr>
<tr>
<td>15</td>
<td>EDH packet checksum error(^{(4)})</td>
</tr>
</tbody>
</table>

Notes:

1. The ANC error conditions are associated with errors in the ancillary data packets.
2. The FF error conditions are associated with errors detected by the full field CRC.
3. The AP error conditions are associated with errors detected by the active picture CRC.
4. The EDH packet checksum error indicates a checksum error was found within the EDH packet itself.

Each ANC, FF, and AP error condition set has five individual error flags. All flags are asserted High to indicate an error condition. For a complete description of the EDH, EDA, IDH, IDA, and UES error flags in the EDH packet, refer to the SMPTE RP 165 document [Ref 5].

- **EDH error**: This error condition occurs when the EDH processor detects a CRC error (checksum error for ANC packets) in a field. For example, the FF EDH error flag indicates an error was detected by the full field CRC.

- **EDA error**: This error condition occurs when the EDA or EDH flags of the received EDH packet are asserted.

- **IDH error**: This error condition is not supported by the RX EDH processor.

- **IDA error**: This error condition occurs when the IDA or IDH flags of the received EDH packet are asserted.

- **UES error**: This error condition occurs when the UES flag in the received EDH packet is asserted.

Besides being counted, if enabled by the error counter, any detected ANC EDH, AP EDH, and FF EDH errors are also indicated by assertion of the rx_edh_anc, rx_edh_ap, and rx_edh_ff ports, respectively. Thus, the rx_edh_anc port is asserted whenever a checksum error is detected in an ancillary data packet. The rx_edh_ap port is asserted when the calculated active picture CRC does not match the AP CRC in the EDH packet. And, the rx_edh_ff port is asserted when the calculated full field CRC does not match the FF CRC in the EDH packet.

The RX EDH processor also outputs the ANC, AP, and FF error flags from the EDH packet on the rx_edh_anc_flags, rx_edh_ap_flags, and rx_edh_ff_flags ports, respectively. These output ports are exact copies of the flags found in the last received EDH packet. Thus, they differ from the detected errors used to increment the error counter and output on the rx_edh_anc, rx_edh_ap, and rx_edh_ff ports. For example, the EDH flag (bit 0) of the rx_edh_ap_flags port indicates that the AP EDH flag was set in the last received EDH packet. However, the rx_edh_ap port indicates that the active picture CRC calculated locally by the EDH processor does not match the AP CRC value in the EDH packet. The rx_edh_anc_flags, rx_edh_ap_flags, and rx_edh_ff_flags ports are each five bits wide; the encoding of all three
ports are identical and is shown in Table 6.

Table 6: Encoding of rx_edh_anc_flags, rx_edh_ap_flags, and rx_edh_ff_flags Ports

<table>
<thead>
<tr>
<th>Bit #</th>
<th>Flag</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>EDH</td>
</tr>
<tr>
<td>1</td>
<td>EDA</td>
</tr>
<tr>
<td>2</td>
<td>IDH</td>
</tr>
<tr>
<td>3</td>
<td>IDA</td>
</tr>
<tr>
<td>4</td>
<td>UES</td>
</tr>
</tbody>
</table>

The RX EDH processor also produces four error flags related to the format and contents of the EDH packet itself. These error flags are output on the rx_edh_packet_flags port. The encoding of this port is shown in Table 7.

Table 7: Encoding of rx_edh_packet_flags Port

<table>
<thead>
<tr>
<th>Bit #</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>EDH packet is missing</td>
</tr>
<tr>
<td>1</td>
<td>Parity error in user data words of EDH packet</td>
</tr>
<tr>
<td>2</td>
<td>Checksum error in EDH packet</td>
</tr>
<tr>
<td>3</td>
<td>Format error in EDH packet – such as invalid data count</td>
</tr>
</tbody>
</table>

GTH Initialization and Reset and Change Sequence Failure Codes

If a failure occurs during a GTH RX initialization or reset sequence or during a dynamic change of the RX SDI mode, the rx_change_fail port will be asserted high and a failure code will be output on the rx_change_fail_code port. A sequence only ends in failure once it has been retried the maximum number of times allowed by the retry counter. The maximum number of retries is controlled by the width of the retry counter as specified by the RETRY_CNTR_BITWIDTH parameter or generic. The number of retries attempted is:

\[
\text{Retries} = 2^{\text{RETRY\_CNTR\_BITWIDTH}} - 1
\]

The encoding of the rx_change_fail port is shown in Table 8.

Table 8: rx_change_fail_code Port Encoding

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>This code indicates that the PLL failed to lock to its reference clock within the allowed time or the GTH transceiver failed to assert rxresetdone within the allowed period of time following a gtrxreset.</td>
</tr>
<tr>
<td>1</td>
<td>This code indicates that the DRP arbiter was not able to grant control of the DRP to the gtrxreset state machine in the GTH wrapper to do a gtrxreset sequence because the DRP was constantly busy. Such a failure should only occur if there is an issue with the v7gth_sdi_drp_control module which prevents it from giving up the DRP.</td>
</tr>
<tr>
<td>2</td>
<td>When a RX SDI mode change sequence is initiated, the v7gth_sdi_drp_control module begins that sequence by requesting the DRP from the DRP arbiter. If the DRP request is not granted within the allowed amount of time, including retries, the sequence fails with this failure code.</td>
</tr>
</tbody>
</table>
Implementing an SDI Interface in Virtex-7

Any sequence failure that results in the rx_change_fail port going High will cause the GTH RX control logic in the SDI wrapper to stop in a failure condition. The GTH RX might still receive an SDI signal, but will not dynamically switch between SDI modes as it normally would. A full reset of the GTH RX, by asserting rx_gth_full_reset High, is required to attempt to resolve the failed condition. Repeated failures most likely indicate an issue with the design of the application.

If a failure occurs during a GTH TX initialization or reset sequence or during a dynamic change of the GTH transceiver txrate or txsysclksel ports, the tx_change_fail port will be asserted High and a failure code will be output on the rx_change_fail port. As with the RX side, sequences only fail after the maximum number of retries has been attempted. The encoding of the tx_change_fail_code port is shown in Table 9.

Table 9: tx_change_fail_code Port Encoding

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reserved</td>
</tr>
<tr>
<td>1</td>
<td>During a full reset sequence or the GTH TX initialization sequence, this failure code indicates that the PLL providing the serial clock to the GTH TX failed to assert its lock signal within the given amount of time, including retries, after being reset.</td>
</tr>
<tr>
<td>2</td>
<td>During the GTH transceiver initialization sequence, a GTH transceiver full reset sequence, or a gttxreset sequence requested by the application by assertion of the tx_gth_full_reset or tx_gth_reset ports, this failure code indicates that the GTH transceiver failed to negate its txresetdone signal within the given amount of time, including retries, after assertion of gttxreset. This indicates a failure of the GTH transceiver to start the reset sequence.</td>
</tr>
<tr>
<td>3</td>
<td>During the GTH initialization sequence, a GTH full reset sequence, or a gttxreset sequence requested by the application by assertion of the tx_gth_full_reset or tx_gth_reset ports, this failure code indicates that the GTH transceiver failed to assert its txresetdone signal within the given amount of time, including retries, after a gttxreset. This indicates that the GTH transceiver failed to complete the reset sequence.</td>
</tr>
</tbody>
</table>
SDI Timing Constraints

For the SDI wrapper and the SDI core, only the periods of the clocks need to be constrained. These are the clocks applied to the clk, rx_usrclk, tx_usrclk, and gth_drpclk ports of the SDI wrapper.

The rx_usrclk and tx_userclk clocks are usually constrained to 148.5 MHz, sometimes rounded up to 150 MHz.

Vivado tools consider all clocks to be related, unless told otherwise. The various clocks of the SDI wrapper are generally unrelated, so a constraint is required to specify that these clocks are not related.

See the timing constraints files of the example SDI demonstration provided with this application note for examples of setting these constraints.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>This failure code indicates that the GTH transceiver failed to indicate successful completion of a txrate change by asserting its txratedone output within the allowed amount of time, including retries.</td>
</tr>
<tr>
<td>5</td>
<td>When the application requests a dynamic change of txsysclkssel by changing the tx_m input of the SDI wrapper, gttxreset is asserted prior to the change of txsysclkssel. If the GTH fails to negate its txresetdone output in response to the assertion of gttxreset within the allowed amount of time, including retries, the txsysclkssel change sequence fails with this failure code.</td>
</tr>
<tr>
<td>6</td>
<td>During a dynamic change of txsysclkssel, gttxreset is asserted. After txsysclkssel is changed, gttxreset is negated. If the GTH transceiver fails to assert the txresetdone output within the allowed amount of time, including retries, after gttxreset is negated, the txsysclkssel change sequence fails with this failure code.</td>
</tr>
<tr>
<td>7</td>
<td>Reserved</td>
</tr>
</tbody>
</table>

Table 9: \textit{tx\_change\_fail\_code} Port Encoding (Cont’d)

Example SDI Demonstrations

Two example SDI demonstration applications are included with this application note. The source code for these demonstrations is provided in Verilog only. Instructions for building these demonstrations using Vivado tools are included in the \textit{readme.txt} file located in the xapp1187.zip file along with the source code. Pre-generated FPGA configuration files are also provided for both demonstrations that can be loaded onto a Virtex-7 FPGA VC709 evaluation board. These demonstrations require an inrevium TB-FMCH-3GSDI2A FMC, which provides the SDI cable drivers and SDI cable equalizers connected to the FMC connector of the VC709 board. The inrevium FMC also provides SDI-specific clock sources that are used as reference clocks for the GTH transceivers.

Quad SDI Demonstration

This demonstration application includes four SDI RX interfaces and four SDI TX interfaces that are all independent. All four GTH transceivers used by this application are in the same GTH Quad.

Each SDI TX is driven by a video pattern generator. The SDI mode, video format, and video pattern of each SDI TX can independently be selected using VIO windows in the ChipScope™ Pro analyzer. The status of each SDI RX can be monitored using a VIO window in the ChipScope Pro analyzer. And the video data received by each SDI RX can be captured and viewed using an ILA window in the ChipScope Pro analyzer.

It is possible to use the Vivado Logic Analyzer instead of ChipScope Pro analyzer to control and monitor the demonstrations. However, at this time, the ChipScope Pro analyzer provides a better user interface for these SDI demos. It is recommended that ChipScope Pro analyzer be
used and the only instructions provided in this application note are for ChipScope Pro analyzer. The applications can be built using either ChipScope Pro analyzer or Vivado analyzer and instructions for building the demos both ways are included in the readme.txt file. The FPGA configuration files included in the xapp1187.zip file are built for ChipScope Pro analyzer.

The Inrevium SDI FMC board has six connectors for the SDI interfaces. The connectors labeled CH0-RX and CH0-TX are the SDI RX and TX connectors for the first SDI interface and the connectors labeled CH1-RX and CH1-TX are the SDI RX and TX connectors for the second SDI interface. The third and fourth SDI interfaces have only a single connector each, labeled CH2 and CH3. These are bidirectional interfaces. The TX2 and TX3 VIO windows in ChipScope Pro analyzer each have a button that controls the direction of the bidirectional interface.

Figure 20 is a block diagram of the demonstration, showing SDI channel 0 which is connected to the first GTH transceiver in the Quad. All four SDI channels are identical in this demonstration, with the exception of the CH0 SDI wrapper which is the QPLL master responsible for resetting the QPLL, CH2, and CH3 to have bidirectional SDI physical interfaces on the Inrevium FMC.

**Figure 20: Quad SDI Block Diagram**

The Inrevium SDI FMC board has 148.5 MHz and 148.5/1.001 MHz oscillators which this demonstration uses to supply reference clocks to the QPLL and the CPLLLs of each transceiver. The 148.5 MHz reference clock is used by the QPLL and the 148.5/1.001 MHz reference clock...
is used by the CPLL. The GTH transmitters are dynamically switched between the serial clock from the QPLL and the serial clock from its CPLL to support all SDI bit rates.

The LMH1983 device on the inrevium board supplies a 27 MHz clock to the Virtex-7 FPGA that is used for the DRP clock and fixed frequency clock required by the control module.

To make it easier to replicate the SDI interface four times in this demonstration, the SDI wrapper, GTH wrapper, video pattern generators, TX clock enable generator, ChipScope VIO and ILA modules, and other miscellaneous logic for a single SDI interface are contained in a module called v7gth_sdi_rxtx. This module is instantiated four times in the top-level module of the design.

The following are required to run the Quad SDI demonstration:

- Xilinx Virtex-7 FPGA VC709 Evaluation Kit
- inrevium TB-FMCH-3GSDI2A SDI FMC
- DIN 1.0/2.3 to BNC converter cables (supplied with the TB-FMCH-3GSDI2A)
- SDI signal source
- SDI signal sink (waveform monitor or other device to view signal from SDI transmitters)
- PC with ChipScope Pro analyzer installed
The inrevium SDI FMC board must be connected to the FMC connector on the VC709 board as shown in Figure 20.

ChipScope Pro analyzer is required to run this demonstration. It is used to control the SDI transmitters and to look at the status and received data from the SDI receivers. The VC709 board must be connected to a PC with ChipScope Pro analyzer installed by the USB JTAG cable provided with the VC709 board.

The configuration file named `vc709_sdi_demo.bit` provided with this application note must be loaded into the Virtex-7 FPGA on the VC709 board using ChipScope Pro analyzer. After this configure file is loaded into the FPGA, the `vc709_sdi_demo.cpj` ChipScope project file must be opened in ChipScope Pro analyzer. When the project is opened, it looks like Figure 22. There are nine VIO windows, one for each RX and TX, and one showing the locked status of the PLLs. There are also four ILA waveform windows, one for each receiver in the demonstration, that are minimized in Figure 22. The PLL status VIO might not be visible when the ChipScope project is opened. It can be opened by selecting UNIT:0 MyVIO0 in the Project panel in the upper right of the ChipScope Pro analyzer window.

Figure 21: **VC709 Board with TB-FMCH-3GSDI2A Board Connected**
To observe the signal being generated by an SDI transmitter, an SDI waveform monitor or other SDI display device must be connected to the output of the SDI TX. Or, the SDI transmitter output can be connected back to one of the SDI inputs on the inrevium FMC with a cable. The SDI connectors on the inrevium SDI FMC board are not standard BNC connectors, so adapter cables are required to go from these DIN 1.0/2.3 connectors to regular BNC connectors.

**Figure 22: ChipScope Pro Analyzer with Quad SDI Project**

To observe the signal being generated by an SDI transmitter, an SDI waveform monitor or other SDI display device must be connected to the output of the SDI TX. Or, the SDI transmitter output can be connected back to one of the SDI inputs on the inrevium FMC with a cable. The SDI connectors on the inrevium SDI FMC board are not standard BNC connectors, so adapter cables are required to go from these DIN 1.0/2.3 connectors to regular BNC connectors.
Each SDI transmitter has a VIO control window. The VIO control window for TX2 is shown in Figure 23.

![Quad SDI Demonstration TX VIO Control Window](image)

**Figure 23:** Quad SDI Demonstration TX VIO Control Window

The first three items at the top of the TX VIO window indicate the status of the last GTH TX initialization or dynamic change sequence. If the last sequence completed normally, the Change Done indicator is green. If the last sequence failed, the Change Fail indicator is red and the Change Failure Code indicates the cause of the failure as shown in Table 9.

The TXRESETPDONE indicator shows the status of the GTH transceiver TXRESETPDONE signal. During normal operation, this indicator is green.

The TX Bit Rate toggle button, TX Video Format selection field, and the TX SDI Mode selection field work together to select the format of the SDI signal generated by the SDI transmitter as shown in Table 10.

![Table 10: Quad SDI Demonstration TX Video Format Selection](image)
The TX Video Pattern value selects the video test pattern generated by the video pattern generator driving the SDI TX. In HD-SDI and 3G-SDI mode, three test patterns are available:

- 0 = SMPTE RP 219 color bars
- 1 and 3 = SDI pathological checkfield
- 2 = 75% color bars

In SD-SDI mode, two test patterns are available:

- 0 and 2 = SMPTE EG 1 color bars
- 1 and 3 = SDI pathological checkfield

The VIO windows for TX2 and TX3 have a toggle button called TX Enable. These buttons determine whether the bidirectional SDI interface is configured to transmit (TX Enable = 1) or receive (TX Enable = 0). When the TX Enable button is 1 and the interface is configured to transmit, the SDI receiver is not disabled and will receive the signal being sent by the transmitter. So, for example, if TX2 is enabled (TX Enable = 1), RX2 will receive the signal sent by TX2. The SDI signal sent by TX2 is looped back internally in the bidirectional SDI cable driver and equalizer device on the inrevium FMC back to the input of RX2. The VIO windows for TX0 and TX1 do not have TX Enable toggle buttons because these channels have separate TX and RX connectors on the inrevium FMC.

At the bottom of the TX VIO window are two buttons that reset the GTH TX. The TX GTH Full Reset button resets both the CPLLL and the GTH TX unit. The TX GTH Reset button resets just the GTH TX unit and not the CPLL.

Each SDI receiver has a VIO window to monitor the status of the receiver and an ILA window through which the video data received by the SDI RX and can be viewed. Figure 24 shows the VIO window for one of the receivers.

![VIO Console - DEV:0 MyDevice0 (X...](image)

**Figure 24:** Quad SDI Demonstration RX Status Window
The RX Locked Indicator is green when the SDI RX is locked to the incoming SDI signal and gray when it is not locked.

The RX SDI Signal Type shows the type of SDI signal being received: SD-SDI, HD-SDI, 3G-SDI level A or 3G-SDI level B. This field does not distinguish between 3G-SDI levels B-DL and B-DS.

The RX Bit Rate shows the bit rate of the SDI signal being received.

The SDI Transport Video Format provides information about the video transport that has been detected in the SDI signal. The SDI Transport Frame Rate is the frame rate of the video transport that has been detected in the SDI signal. Both of these refer to the transport structure, not necessarily the picture format. For example, if the signal is 1080p 50 Hz carried on a 3G-SDI level B-DL interface, the transport would be detected and reported as 1080i 25 Hz because the transport is interlaced and has a frame rate of 25 Hz.

The ST 352 Payload ID Data Bytes are the four data bytes of the ST 352 payload ID packet. They are shown with byte 1 on the left and byte 4 on the right. They are only valid when the ST 352 Payload Packet Valid indicator is green.

The RX Error Indicator is red if any CRC or EDH error has been detected and gray if no errors have been detected. After an error has been detected, this indicator will stay red until it is manually reset by clicking the RX Error Clear button. The RX Error Count is an integer count of the number of CRC (HD-SDI and 3G-SDI modes) or EDH errors (SD-SDI mode only) received since the counter was last cleared. The error counter is manually cleared by clicking the RX Error Clear button. The error counter is also cleared automatically when the incoming SDI signal changes bit rates and the SDI RX has to re-lock to the signal. However, the error counter is automatically cleared early in the process of locking to the new SDI signal. Thus, after the SDI RX has fully locked to the new SDI signal, the error count will typically not be zero.

Figure 25 illustrates how to use the ChipScope Pro analyzer ILA to view the data being received by an SDI receiver. Each receiver has an ILA connected to its outputs. To use one of these ILAs, its trigger setup and waveform windows must be brought to the foreground in the ChipScope Pro analyzer. One way to do that is to click the Trigger Setup and Waveform items under the appropriate unit in the Project panel in the upper left as shown in Figure 25. UNIT 3 is the ILA for RX0, UNIT 6 is the ILA for RX1, UNIT 9 is the ILA for RX2, and UNIT 12 is the ILA for RX3.

The trigger setup window can be used to change the trigger point and storage qualification. There are two match units. Typically, match unit M0 is used to trigger the ILA capture and match unit M1 is used to qualify the data storage, usually when the clock enable is High so that, in SD-SDI mode, only valid data words are captured. The vc709_sdi_demo.cpj ChipScope project file has M0 configured to trigger on an EAV and M1 configured to capture data only when the clock enable is High.
With either the trigger setup window or the waveform window for the desired receiver selected, click the triangular play button as shown in Figure 25 to initiate a capture by the ILA. The capture buffer is large enough to capture multiple lines of video.

**Figure 25: Using the ChipScope ILA to View RX Data in the Quad SDI Demonstration**

**SDI Pass-Through Demonstration**

The second SDI demonstration has one SDI RX and one SDI TX connected together in a pass-through configuration such that the TX always retransmits the data received by the RX. Figure 26 shows a block diagram of this demonstration.

The QPLL is locked to a 148.5 MHz reference clock and provides the serial clock to the GTH RX unit. The data from the GTH RX goes through the SDI RX datapath and then into an asynchronous FIFO. The FIFO moves the data from RX clock domain (rx_usrclk) to the TX clock domain (tx_usrclk). In HD-SDI and 3G-SDI modes, the recovered clock from the GTH RX (rxoutclk) is sent to a Silicon Labs Si5324 digital PLL for jitter reduction and then used as the reference clock to the CPLL. In SD-SDI mode, rxoutclk is not a recovered clock and cannot be used to generate the TX reference clock. Instead, the 27 MHz SD-SDI RX clock enable (rx_ce_sd) is sent to the Si5324 which multiplies it to 148.5 MHz while also filtering the jitter. The CPLL is locked to the clock from the Si5324 and provides the serial clock to the GTH TX. Data is read from the asynchronous FIFO in the TX clock domain and enters the SDI TX
 datapath. The resulting SDI data from the SDI TX datapath goes into the GTH TX for serialization.

These items are required to run the SDI pass-through demonstration:

- Xilinx Virtex-7 FPGA VC709 Evaluation Kit
- inrevium TB-FMCH-3GSDI2A SDI FMC
- DIN 1.0/2.3 to BNC converter cables
- SDI signal source
- SDI signal sink (waveform monitor or other device to view signal from SDI transmitters)
- (Optional) A PC with ChipScope Pro analyzer installed and connected to the VC709 board

The inrevium SDI FMC must be connected to the FMC connector on the VC709 board as shown in Figure 21. The only active SDI connectors on the inrevium board are CH0-RX and CH0-TX. An SDI signal source must be connected to the CH0-RX connector. The SDI signal is retransmitted on the CH0-TX connector.

The file named `vc709_sdi_pass.bit` provided with this application note must be loaded into the Virtex-7 FPGA on the VC709 board. After this configuration file is loaded into the FPGA, the `vc709_sdi_pass.cpj` ChipScope analyzer project file can be opened in ChipScope Pro.
analyzer to observe the status of the SDI RX and to capture and observe data received by the SDI RX as shown in Figure 27.

![Diagram of ChipScope VIOs and ILA](image)

Figure 27: Pass-Through Demonstration ChipScope Analyzer Window

There are two ChipScope VIOs and one ILA in this design.

The PLL Status and GTH Reset VIO shows the locked status of the RX PLL (QPLL), the TX PLL (CPLL), and the external Si5324 DPLL that generates the reference clock for the CPLL. In normal operation, the two PLL locked indicators are green and the Si5324 Loss of Lock indicator is gray. When a valid SDI input signal is not present on the input of the SDI RX or during a short period of time after the input SDI signal changes bit rates, the Si5324 will lose lock with the recovered clock from the GTH RX as indicated by Si5324 Loss of Lock indicator turning red. When the Si5324 is not locked, the TX PLL will also not be locked and the TX PLL Locked indicator will turn gray. By observing these PLL lock indicators and the RX Locked indicator in the other VIO window, it is clear that the majority of time required for the SDI output to stabilize after a change of the SDI input signal is the lock time of the Si5324. This VIO also has full GTH reset buttons for the RX and the TX. These buttons generate a full reset of the GTH RX or TX including resetting the associated PLL.

The RX and TX Status VIO window shows the status of the SDI RX and TX. The RX status indicators and the RX Clear Errors buttons in this VIO are identical in function to those in the RX VIO window of the quad SDI demonstration as shown in Figure 24. Refer to that section for a description of the RX status indicators. The three TX status indicators at the bottom of the VIO are identical in function to the similarly named TX status indicators in the TX VIO window of the quad SDI demonstration as shown in Figure 23.
There is a single ILA used to capture and observe the data from the SDI RX. It functions the same ways as the SDI RX ILA in the quad SDI demonstration.

The SDI pass-through demonstration can be used without ChipScope Pro. The pass-through SDI interface is fully functional even when Chipscope Pro is not used to monitor the status of the SDI interface.

**FPGA Resource Usage**

Table 11 shows the FPGA resources required an SDI interface with a Virtex-7 GTH transceiver. The resource usage includes all the modules required to implement the interface, including the SDI core and the SDI wrapper. Resource usage is shown for various common configurations.

The results shown were achieved with Vivado Design Suite 2013.3.

The SDI receiver and transmitter interface designs do not use any MMCM clock managers. And, they do not require any block RAMs or DSP blocks.

Typically, one global or regional clock is required for each SDI TX and for each SDI RX. Also, one fixed frequency global clock is required for timing purposes in the SDI wrapper. This fixed frequency clock is usually also used as the GTH DRP clock. Only one such fixed frequency global clock is required no matter how many SDI interfaces are implemented in the FPGA.

### Table 11: Virtex-7 GTH SDI Interface FPGA Resource Usage

<table>
<thead>
<tr>
<th>Reference Design</th>
<th>LUTs</th>
<th>Flip-Flops</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDI RX with EDH processor and TX</td>
<td>3,781</td>
<td>3,105</td>
</tr>
<tr>
<td>SDI RX without EDH processor and TX</td>
<td>3,106</td>
<td>2,645</td>
</tr>
<tr>
<td>SDI RX with EDH processor</td>
<td>2,253</td>
<td>1,948</td>
</tr>
<tr>
<td>SDI RX without EDH processor</td>
<td>1,569</td>
<td>1,488</td>
</tr>
<tr>
<td>SDI TX</td>
<td>1,548</td>
<td>1,157</td>
</tr>
</tbody>
</table>

**Constraints**

Example constraint files are supplied with the reference designs and can be used as examples of the timing and placement constraints required for SDI interfaces. For timing, generally all that is required are period constraints on the rxoutclk and txoutclk clocks from the GTH transceiver and the fixed frequency clock that is used for the DRPCLK and the SDI wrapper clk port. The rxoutclk and txoutclk constraints should specify the period of these clocks as 148.5 MHz (often rounded up to 150 MHz). For placement, all that is required is to constrain the GTH transceivers to their desired locations by constraining the rxp/rxn and txp/txn pins or using the XY coordinates system to constrain the actual location of the GTH transceiver itself.

**Reference Design**

The reference design files for this application note can be downloaded from: https://secure.xilinx.com/webreg/clickthrough.do?cid=355859

The reference design checklist is shown in Table 12.

### Table 12: Reference Design Checklist

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>General</td>
<td></td>
</tr>
<tr>
<td>Developer name</td>
<td>John Snow</td>
</tr>
<tr>
<td>Target devices</td>
<td>Virtex-7 devices with GTH transceivers, -2 speed grade or faster</td>
</tr>
<tr>
<td>Source code provided</td>
<td>Yes</td>
</tr>
</tbody>
</table>
The readme.txt file describes the directory structure of the files provided with the reference design.

**Conclusion**

This document describes how to use the SMPTE SD/HD/3G-SDI core and the Virtex-7 GTH transceivers to implement SDI interfaces compatible with the SMPTE SD-SDI, HD-SDI, and 3G-SDI standards. The Virtex-7 GTH device-specific control logic necessary to use the transceivers in SDI applications is included with this application note. Also included are two example SDI demonstration applications providing detailed examples of SDI implementations in a Virtex-7 FPGA design.

**Appendix A: Glossary**

The terms in Table 13 are used in this application note.

<table>
<thead>
<tr>
<th><strong>Table 13: Glossary</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>3G-SDI</strong></td>
</tr>
<tr>
<td><strong>Ancillary (ANC) data</strong></td>
</tr>
<tr>
<td><strong>Data stream</strong></td>
</tr>
<tr>
<td><strong>Table 13: Glossary</strong></td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td><strong>EDH</strong></td>
</tr>
<tr>
<td><strong>Embedded audio</strong></td>
</tr>
<tr>
<td><strong>End of active video (EAV)</strong></td>
</tr>
<tr>
<td><strong>HD-SDI</strong></td>
</tr>
<tr>
<td><strong>Interlaced</strong></td>
</tr>
<tr>
<td><strong>Link</strong></td>
</tr>
<tr>
<td><strong>Payload ID</strong></td>
</tr>
<tr>
<td><strong>Progressive</strong></td>
</tr>
<tr>
<td><strong>Serial Digital Interface (SDI)</strong></td>
</tr>
<tr>
<td><strong>SD-SDI</strong></td>
</tr>
<tr>
<td><strong>SMPTE</strong></td>
</tr>
<tr>
<td><strong>Start of active video (SAV)</strong></td>
</tr>
<tr>
<td><strong>Synchronous switching (point, interval, line)</strong></td>
</tr>
<tr>
<td><strong>Transport</strong></td>
</tr>
</tbody>
</table>
Appendix B: References

Available from Xilinx (www.xilinx.com):
1. 7 Series FPGAs GTX/GTH Transceivers User Guide (UG476)
2. Dynamically Programmable DRU for High-Speed Serial I/O (XAPP875)
3. Virtex-7 T and XT FPGAs Data Sheet: DC and Switching Characteristics (DS183)
4. SMPTE SD/HD/3G-SDI Product Guide (PG071)

Available from the Society of Motion Picture and Television Engineers (www.smpte.org):
5. RP 165: Error Detection Checkwords and Status Flags for Use in Bit-Serial Digital Interfaces for Television
6. RP 168: Definition of Vertical Switching Point for Synchronous Video Switching
8. ST 291-1: Television – Ancillary Data Packet and Space Formatting
9. ST 292-1: 1.5 Gb/s Signal/Data Serial Interface
10. ST 344: Television – 540 Mb/s Serial Digital Interface
11. ST 352: Payload Identifier Codes for Serial Digital Interfaces
12. ST 372: Dual Link 1.5 Gb/s Digital Interface for 1920 x 1080 and 2048 x 1080 Picture Formats
13. ST 424: Television – 3 Gb/s Signal/Data Serial Interface
14. ST 425-1: Source Image Format and Ancillary Data Mapping for the 3 Gb/s Serial Interface

Table 13: Glossary

<table>
<thead>
<tr>
<th>Timing reference signal (TRS)</th>
<th>A generic term referring to both EAV and SAV sequences.</th>
</tr>
</thead>
<tbody>
<tr>
<td>XYZ</td>
<td>The fourth word of each EAV and SAV is called the XYZ word. This word carries the horizontal (H), vertical (V), and field (F) bits that indicate the video timing. The XYZ word also contains some protection bits that allow detection of errors in the XYZ word.</td>
</tr>
</tbody>
</table>

Revision History

The following table shows the revision history for this document.

<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Description of Revisions</th>
</tr>
</thead>
<tbody>
<tr>
<td>02/21/2014</td>
<td>1.0</td>
<td>Initial Xilinx release.</td>
</tr>
</tbody>
</table>

Notice of Disclaimer

The information disclosed to you hereunder (the “Materials”) is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of the Limited Warranties which
can be viewed at http://www.xilinx.com/warranty.htm; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in Critical Applications: http://www.xilinx.com/warranty.htm#critapps.