XAPP1097 (v1.0.1)
November 10, 2015

Implementing SMPTE SDI Interfaces with Artix-7 FPGA GTP Transceivers

Author: John Snow

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 LogiCORE™ 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 LogiCORE IP with the Artix®-7 FPGA GTP transceivers to form a complete SDI interface. This application note also provides several example SDI designs that run on the Xilinx Artix-7 FPGA AC701 evaluation board.

Terms in this document are explained in the Glossary. Titles of SMPTE standards are listed in References, and referred to by SMPTE document number in the text.

Introduction

The Xilinx SMPTE SD/HD/3G-SDI LogiCORE IP (hereinafter called the *SDI core*) can be connected to an Artix-7 FPGA GTP transceiver to implement an SDI interface capable of supporting the SMPTE SD-SDI, HD-SDI, and 3G-SDI standards. The SDI core and GTP 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 in both Verilog and VHDL source code.

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

- Reset logic for the GTP transceiver
- Dynamic switching of the GTP 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 GTP transceiver and the SDI core with the necessary connections between them. This simplifies the process of creating an SDI interface.

The following terms are used in this document. Figure 1 is 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 generated by the CORE Generator™ tool or the Vivado® IP catalog.

- The *control module* is a module that implements the various device-specific functions required when using the GTP 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 *GTP wrapper* is a wrapper file for GTP transceivers generated by the 7 Series FPGAs Transceivers Wizard, which is available in the CORE Generator and Vivado IP catalog tools.
- The *GTP common wrapper* is a wrapper file for the GTP transceiver common block generated by the 7 Series FPGAs Transceivers Wizard. It contains the two PLLs that provide serial clocks for the GTP transceivers in the Quad.

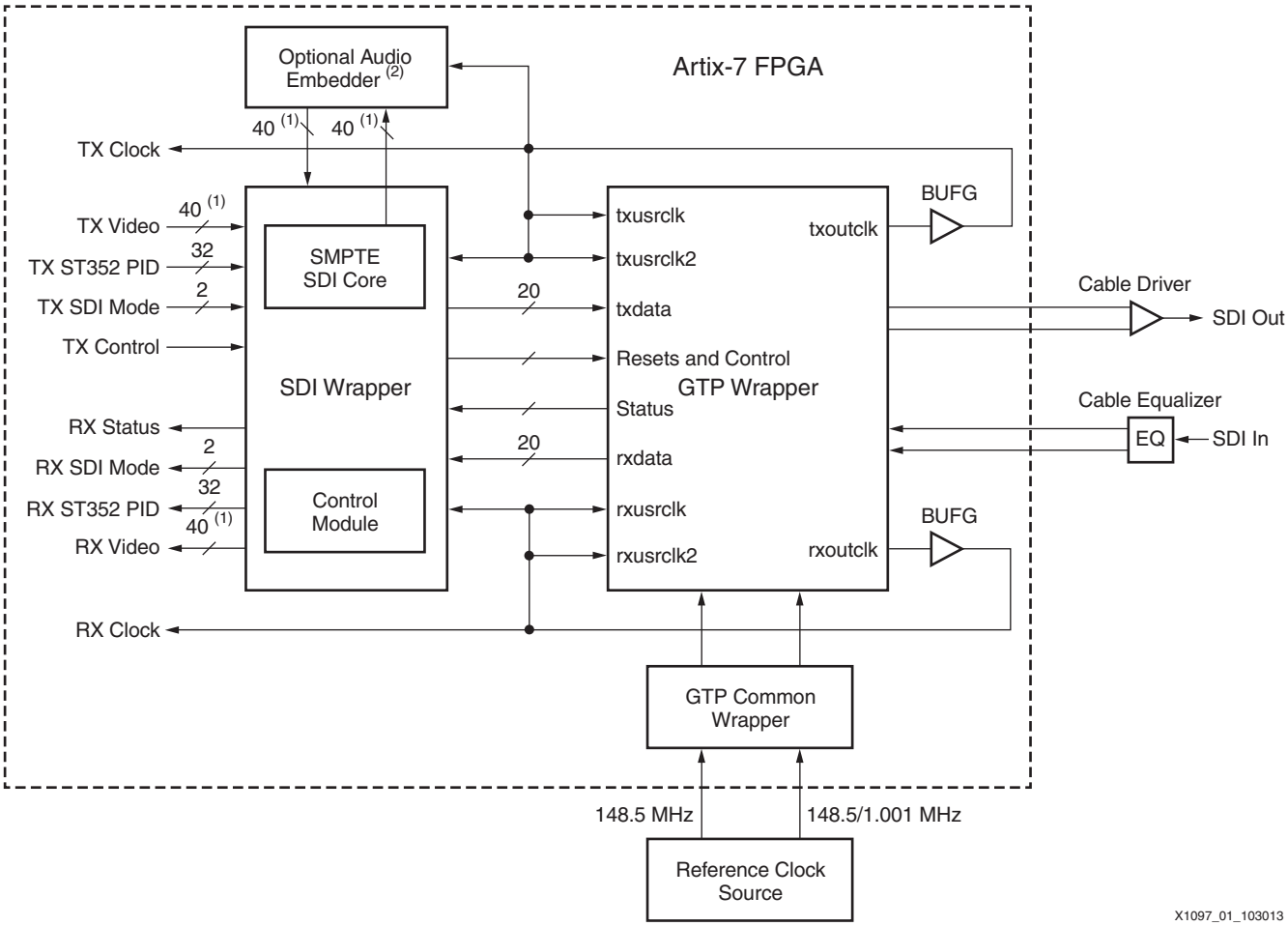


Figure 1: **Block Diagram of Complete SDI RX/TX Interface**

Notes relevant to Figure 1:

1. These 40-bit buses are actually four buses, each of which is 10 bits wide, and each carries a different SDI data stream. The number of active data streams, and therefore buses, varies depending on the SDI mode. For example, in SD-SDI mode, only one 10-bit data stream is active and in HD-SDI mode, two 10-bit data streams are active.
2. The optional audio embedder is a separate core and is not included with the SDI core or with this application note.

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 GTP RX and TX units in the same GTP transceiver, but this does not have to be the case. The RX and TX units of different GTP transceivers can be connected to the same SDI wrapper. If only an SDI RX or only an 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 AC701 evaluation board. An inrevium SDI FPGA mezzanine card (FMC) is also required to provide the SDI physical interfaces.

Using Artix-7 GTP Transceivers for SDI Interfaces

The information in this section is intended to supplement, not replace, the information in *7 Series FPGAs GTP Transceivers User Guide* (UG482) [Ref 1]. This information highlights features and operating requirements of the GTP transceivers that are of particular importance for SDI applications.

In this document, the naming convention used in the *7 Series FPGAs GTP Transceivers User Guide* for the GTP transceiver ports is followed. This convention is to use only the base name of a port. When the 7 Series FPGAs Transceivers Wizard (hereinafter called the Wizard) is used to create a GTP wrapper, all input ports 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 GTP wrapper would be txrate_in for the txrate port of the GTP transceiver.

Starting with version 3.0 of the Wizard, all GTP port names on the top-level GTP wrapper are all lower case, but only in the Vivado tools. The ISE® tools version of the Wizard still produces all upper case port names. All GTP port names used in this application note are lower case. The demonstration source code files are provided in versions that are compatible with both the Vivado and ISE tools and use the appropriate case for the GTP port names.

Also starting with version 3.0 of the Wizard, the GTP common wrapper that contains the two PLLs for the GTP Quad is a separate wrapper and not included in the main GTP wrapper. Again, this only applies to the Vivado tools and not to the ISE tools.

There are a variety of clocks in applications using GTP 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. GTP transceivers require reference clocks to operate. The reference clocks are used by phase-locked loops (PLLs) in the GTP Quad to generate serial clocks for the receiver and transmitter sections of each transceiver. As described in more detail in GTP Reference Clocks, page 4, the serial bit rate of the GTP 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 GTP transmitter. Consequently, the designer 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 GTP 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 GTP transmitter. The txoutclk is generated in the GTP transmitter by dividing the serial clock from the PLL down to the word rate. In most applications, the txoutclk from the GTP transmitter is buffered by a global (BUFG) or regional (BUFR) clock buffer and then used to clock the SDI transmitter datapath and the txusrclk and txusrclk2 clock inputs of the GTP 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 GTP transmitter. A shallow TX buffer in the GTP transmitter does allow for phase differences between the data entering the txdata port and the internal clock of the GTP transmitter. However, any frequency difference between the incoming data and the internal clock frequency of the GTP transmitter (as represented by txoutclk) quickly causes the TX buffer to under- or overflow, resulting in errors in the serial bit stream generated by the GTP transmitter. Consequently, the data rate of the stream entering the txdata port of the GTP transmitter (as represented by the frequency of the txusrclk and txusrclk2 clocks) and the

internal data rate of the GTP transmitter (as set by the transmitter reference clock and represented by the frequency of txoutclk) must match exactly.

The GTP 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 GTP receiver can receive bit rates that are up to ±1250 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 GTP 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 GTP 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 regional clock buffer and then applied to the rxusrclk and rxusrclk2 ports of the GTP 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 GTP 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.

GTP Reference Clocks

Artix-7 FPGA GTP transceivers are grouped into Quads. Each Quad contains four GTPE2_CHANNEL transceiver primitives and one GTPE2_COMMON primitive containing two PLLs called PLL0 and PLL1, as shown in Figure 2. Only the clocks from these two PLLs can be used as the serial clocks for all four receivers and transmitters in the Quad. This does present some limitations for SDI applications as will be described later.

Each receiver and transmitter unit in the Quad can be individually configured to use the clock from either PLL0 or PLL1. Furthermore, any receiver or transmitter unit can dynamically switch its serial clock source between PLL0 and PLL1. This dynamic switching capability is particularly useful for SDI applications.

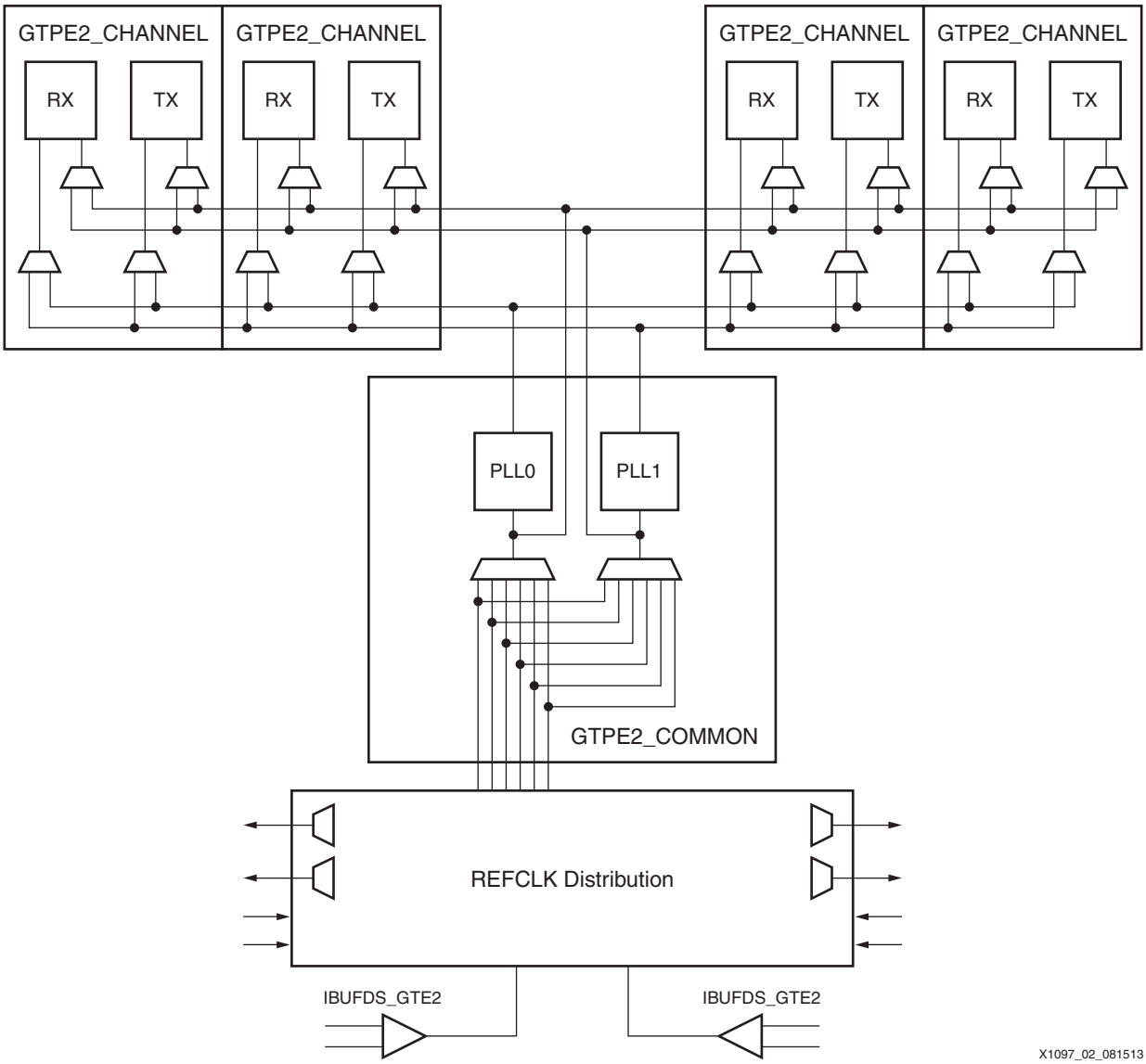


Figure 2: **GTP Transceiver Quad Configuration**

Typical SDI applications require the GTP 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 GTP transceiver can support receiving bit rates that are up to ±1250 ppm from the reference frequency. Because the two bit rates of HD-SDI are exactly 1000 ppm different and likewise the two 3G-SDI bit rates are exactly 1000 ppm different, it is possible to receive all five of the SDI bit rates using a single reference clock frequency.

The TX section of the GTP transceiver, however, requires two different reference frequencies to support all five SDI bit rates. This is because the transmitters, in general, can only transmit at an exact integer multiple of the supplied reference clock frequency. Therefore, most SDI applications provide two separate reference clocks to the GTP Quad. One of those clocks is

used as the RX reference clock and both of them are used as TX reference clocks. Usually, the supplied reference frequency pair are 148.5 MHz and 148.5/1.001 MHz.

The source of the GTP 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 GTP transmitter's 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 GTP transmitter line rate from the studio video reference signal. In some cases, such as the SDI pass-through demonstration included with the application note, the transmitter line rate is derived from the recovered clock of the GTP 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 PLL0 and the other is connected to PLL1 in each Quad that is implementing SDI interfaces. The RX units of each transceiver in the Quad can be configured to use the clock from either PLL. The TX units can dynamically switch between the clocks from PLL0 and PLL1, depending on the bit rate that is required at the moment. The GTP txsysclksel port is used to select the TX unit's clock source between the two PLLs. 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.

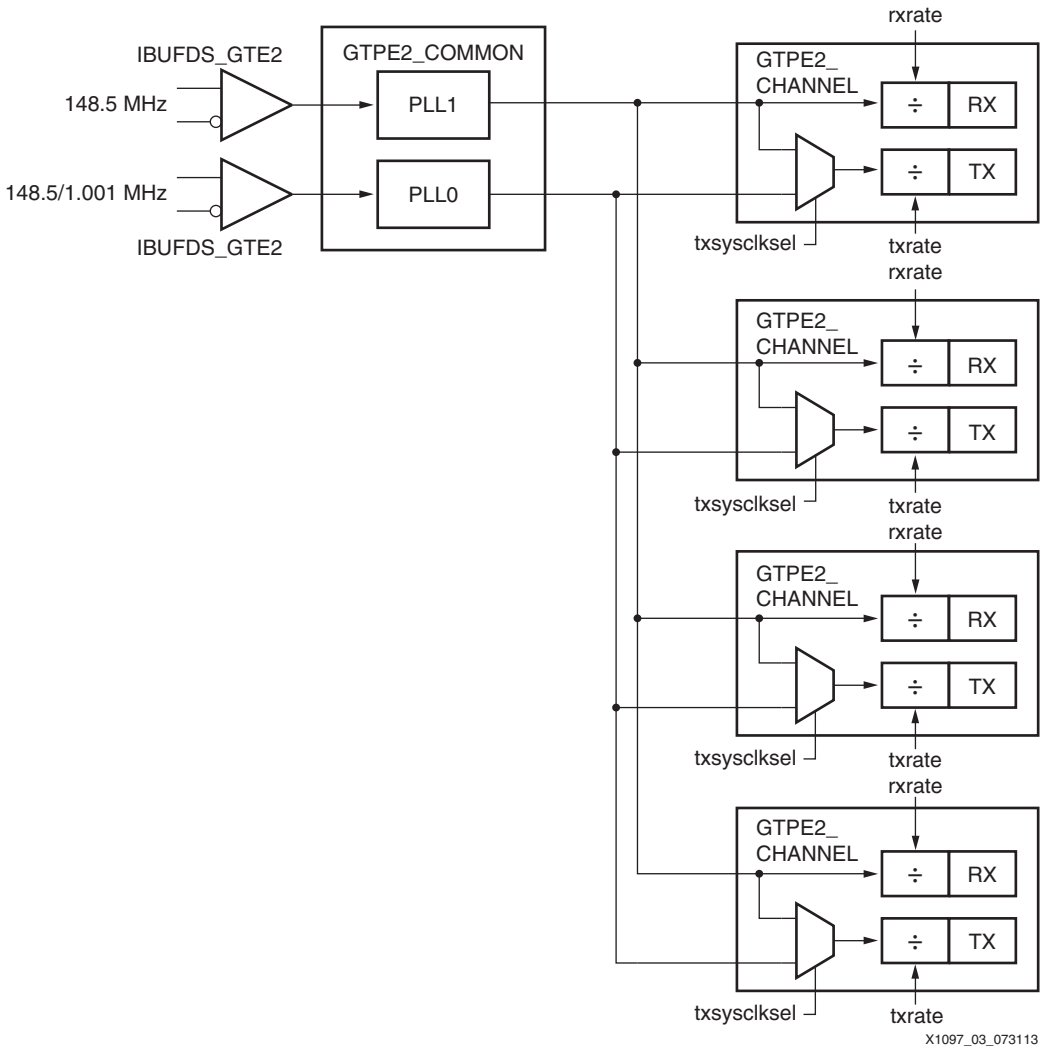


Figure 3: **Typical GTP Reference Clock Implementation for SDI**

Additionally, each GTP 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 one PLL but operate at different line 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. For 270 Mb/s SD-SDI, the GTP 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 one PLL. 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 ports of each GTP transceiver.

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 one PLL 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 the two PLLs to get both serial clocks needed to transmit all supported SDI bit rates.
- All four receivers and all four transmitters in the Quad are fully independent, can each run 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 can simultaneously provide both required reference clock frequencies from the synchronization reference input signal.

In some SDI applications, it might be necessary for 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 actually have bit rates that differ by a few ppm. Supporting such applications with the Artix-7 FPGA GTP Quad structure is difficult, and such applications might be better suited to 7 series devices with GTX or GTH transceivers that have more PLLs available in each Quad and/or have the phase interpolation control oscillator (PICXO) ported to them.

Because the receivers all require an uninterrupted serial clock from one PLL, only the second PLL in the Quad is available to provide a TX serial clock in applications like SDI routers. Thus, for these applications, the GTP Quad can only support two possible configurations where each TX unit has its own serial clock:

- Four receivers using the serial clock from one PLL and a single transmitter using the serial clock from the second PLL (Figure 4)
- Two transmitters and no receivers active in the Quad with each transmitter using a serial clock from a different PLL (Figure 5).

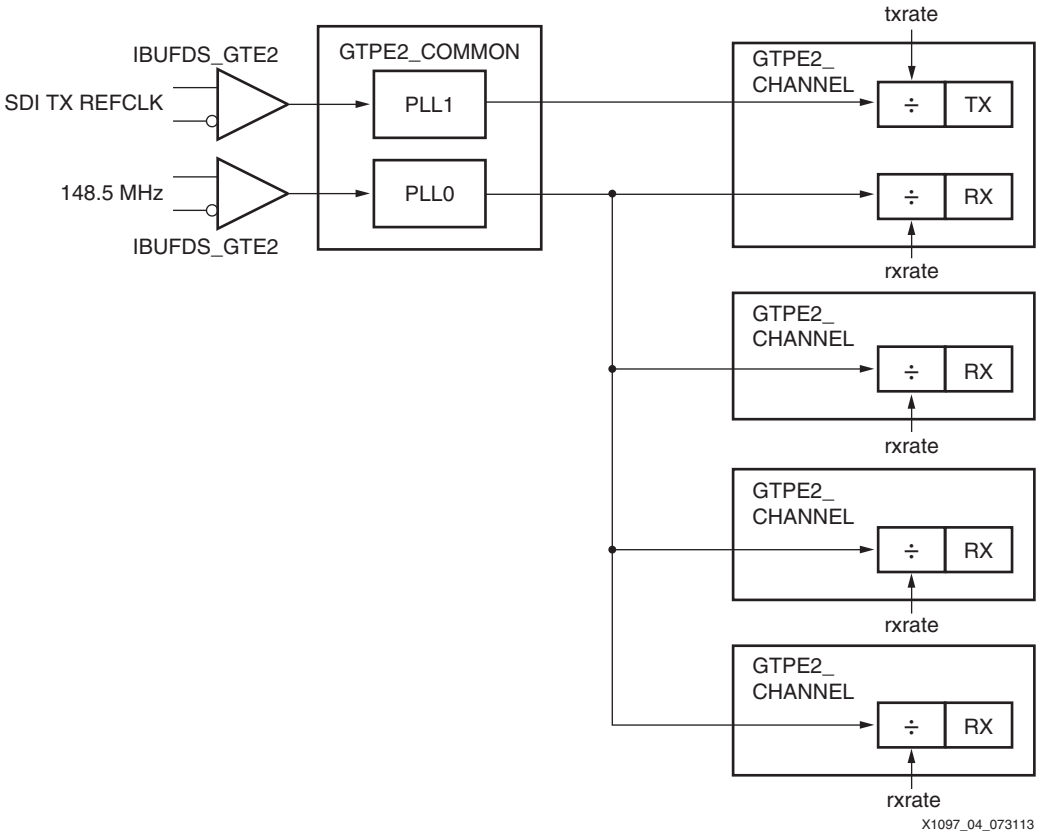


Figure 4: **Four RX and One TX in a GTP Quad**

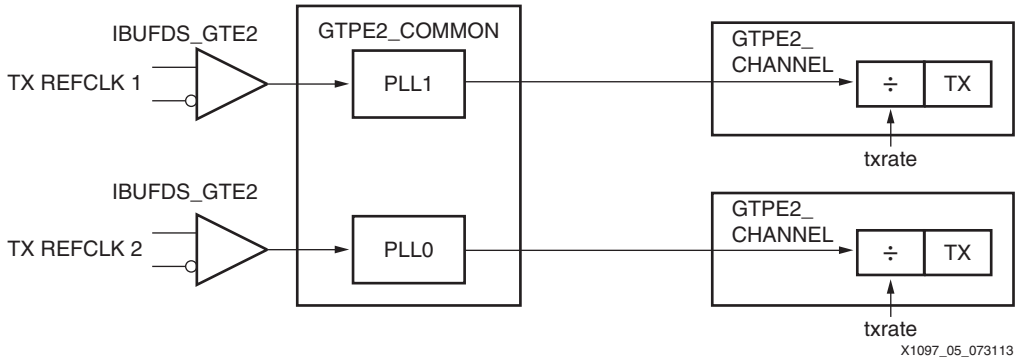


Figure 5: **Two Independent TX in a GTP Quad**

Resets

The GTP transceiver has very specific reset requirements as described in the *7 Series FPGAs GTP Transceivers User Guide* (UG482) [Ref 1]. The GTP transceiver requires careful sequencing between resets of the two PLLs, gttxreset, gtrxreset, and dynamic changes of some GTP ports such as rxrate. Without proper coordination of all of these events, it is possible for the GTP transceiver 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 GTP transceiver.

GTP Initialization Sequence

Immediately following FPGA configuration, the SDI control module executes initialization sequences for the GTPE2_COMMON PLLs and the RX and TX portions of the GTP transceiver. The initialization sequence for both the RX and the TX are the same. The control module has separate state machines that execute an initialization sequence separately for the RX and TX portions of the GTP transceiver. The following procedure describes the sequence for the RX. The TX initialization sequence is identical except that the gttxreset, tx_refclk_stable, and txresetdone signals replace the gtrxreset, rx_refclk_stable, and rxresetdone signals.

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

In addition, the GTP txuserrdy and rxuserrdy inputs must be properly controlled. The SDI wrapper generates both of these signals. It asserts txuserrdy five txusrclk cycles after gttxreset is negated. Likewise, it asserts rxuserrdy five rxusrclk cycles after gtrxreset is negated.

In step 2, step 3, and step 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/generic on the SDI wrapper.

PLL Resets

In addition to being reset during the initialization sequences that run automatically after FPGA configuration, a PLL in the GTPE2_COMMON block 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 to force the PLL to re-lock to the reference clock. The pll0reset and pll1reset inputs of the GTP wrapper are controlled by the SDI control module to implement the PLL resets. The user application should not assert pll0reset or pll1reset directly. The SDI control module, alone, should control the pll0reset and pll1reset signals. However, the user application determines when PLL resets are required and requests that the affected PLL be reset and that all of the GTP RX and/or TX units using the serial clock from that PLL also be reset.

The SDI control module has two inputs that the application should use to request a full reset of the GTP RX (rx_gtp_full_reset) and the GTP TX (tx_gtp_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 GTP transceiver, including resetting the associated PLL. The user application must properly control the rx_gtp_full_reset and tx_gtp_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_gtp_full_reset and tx_gtp_full_reset inputs to the control module. The rx_refclk_stable and tx_refclk_stable are only used after the initialization sequence has been started by assertion of rx_gtp_full_reset or tx_gtp_full_reset.

GTP TX Resets

There are three conditions that require the TX portion of the GTP to be reset:

- Whenever the PLL that supplies the serial clock to the GTP 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_gtp_full_reset to the SDI control module, causing both the PLL and the GTP TX to be reset.
- The GTP gttxreset input must be asserted during dynamic changes of the txsysclksel port. The txsysclksel port is used to select which of the two PLLs in the GTPE2_COMMON block is used as the serial clock source for the GTP TX. Each GTP transceiver in the Quad 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 GTP 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 GTP 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.
- The GTP TX is automatically reset by the GTP transceiver itself whenever its txrate input port dynamically changes. The txrate controls the serial clock divider for the GTP TX. The user application should not change txrate directly. The SDI control module changes txrate, when appropriate, in response to changes on its tx_mode input port.

In addition, the user application can request a reset of the GTP TX by asserting the tx_gtp_reset input port of the SDI control module. This initiates a gttxreset sequence without resetting the PLL used by the GTP TX.

All of the actions, GTP resets, and dynamic changes of txsysclksel and txrate are coordinated by the TX control state machine in the SDI control module in order to prevent them from interfering with each other. It is important that such interference be avoided. Thus, the user application should not control any of these things directly, but should rely on the SDI control module to do so.

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_gtp_full_reset: When asserted High, this input resets both the PLL associated with the TX and then the TX section of the GTP (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_gtp_reset: When asserted High, this input resets the TX section of the GTP transceiver only (gttxreset). If the PLL is not locked when the gttxreset sequence begins, the gttxreset sequence does not complete until after the PLL is locked.

GTP RX Resets

The RX resets are more complicated than the TX resets and must be even more carefully controlled. As with the TX section, the user application should rely on the SDI control module to carefully coordinate all of the activities described in this section to prevent them from interfering with each other.

The following conditions require resets of the GTP RX section:

- Whenever the PLL that supplies the serial clock to the GTP RX is reset, the gtrxreset 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_gtp_full_reset to the SDI control module, causing both the PLL and the GTP RX to be reset. Whenever the gtrxreset signal is used to reset the GTP RX for any reason, a very specific sequence must be implemented as described in the *7 Series FPGAs GTP Transceivers User*

Guide (UG482) [Ref 1]. This sequence involves using the DRP port to clear bit 11 of DRP address `0x011` during a portion of the sequence and then restoring it to its previous value. This bit must be 1 for normal SDI operation. A state machine in the GTP wrapper implements this complete sequence whenever gtrxreset is asserted.

- Whenever the rxrate port is changed dynamically, a very specific sequence is required as described in the rxrate change use model in the *7 Series FPGAs GTP Transceivers User Guide*. This sequence also involves clearing bit 11 of DRP address `0x011` during a portion of the sequence. The PMA section of the GTP transceiver is reset during this rxrate change sequence. A state machine in the GTP wrapper implements this complete sequence whenever rxrate changes.
- Whenever the CDR configuration of the GTP RX is changed, the gtrxreset port must be used to reset the RX section. There are two different things that the SDI control module uses to configure the GTP CDR correctly based on the current SDI mode (SD-SDI, HD-SDI, or 3G-SDI). The rxcdrhold port of the GTP transceiver is asserted when the current RX mode is SD-SDI. When switching to HD-SDI mode from either SD-SDI or 3G-SDI mode, or when switching from HD-SDI mode to SD-SDI or 3G-SDI modes, the RXCDR_CFG attribute of the GTP transceiver is dynamically changed through the DRP. Changes to either rxcdrhold or the RXCDR_CFG attribute must be followed by a gtrxreset.

In addition, the user application can request a reset of the GTP RX by asserting the rx_gtp_reset input port of the SDI control module. This initiates a gtrxreset sequence without resetting the PLL used by the GTP RX.

There are three different sequences that require the use of the GTP DRP for SDI applications: gtrxreset, rxrate changes, and RXCDR_CFG attribute changes. Each of these is controlled by a different state machine. Changes to gtrxreset and rxrate are controlled by separate state machines inside the GTP wrapper. The SDI control module handles RXCDR_CFG changes. The two state machines inside the GTP wrapper do not coordinate their DRP bus activity with each other and can interfere with each other if gtrxreset sequences and rxrate change sequences are allowed to overlap. There is nothing inherent in the GTP wrapper that prevents these sequences from overlapping or interfering with DRP cycles being done by the SDI control state machine to change RXCDR_CFG. Assertion of the gtrxreset input of the GTP wrapper causes the gtrxreset state machine in the GTP wrapper to asynchronously assume complete control of the GTP DRP signals, even if some other state machine such as the rxrate change state machine is in the middle of a DRP read or write cycle. As a result, assertions of gtrxreset, rxrate changes, and RXCDR_CFG attribute changes must be carefully coordinated. The SDI control module handles this coordination. Thus, it is important that the user application never directly assert gtrxreset or change the rxrate port. The user application must request all such actions through the SDI control module. Failure to do so can result in cases where bit 11 of DRP address `0x011` is changed to zero and stays that way until the FPGA is reconfigured. This renders the GTP RX incapable of receiving SDI signals until the FPGA is reconfigured.

The design of the rxrate change state machine in the GTP wrapper also imposes another requirement on the rxrate port during the initialization sequence following FPGA configuration. The rxrate port into the GTP wrapper must be held at a value of all zeroes until after the GTP RX initialization sequence is completed. The rxrate change state machine's rxrate change detection logic emerges from FPGA configuration with a value of `000` in the comparison register. If the value on rxrate is anything other than `000` immediately after FPGA configuration is completed, an rxrate change sequence is initiated even though the GTP RX initialization sequence has not completed. The rxrate change sequence and the gtrxreset sequence can interfere with each other and can leave the GTP RX in a state where it cannot receive SDI. The GTP RX remains in this state until the FPGA is reconfigured.

To avoid this situation, the SDI control module places a value of all zeroes on the GTP rxrate port until after the GTP RX initialization sequence is complete. There is, however, a side effect of doing so. The rxrate controls the GTP RX serial clock divider, which dictates the frequency of the rxoutclk produced by the GTP transceiver. The rxoutclk is used to clock all of the SDI core's RX logic. When rxrate is set to all zeroes, a serial clock divider value of one is selected. This

divider value is never used during normal SDI operation. The rxrate port is always used to select serial clock dividers of two or four during normal SDI operation. Therefore, during the GTP RX initialization sequence that occurs immediately after FPGA configuration, the frequency of rxoutclk is 297 MHz. During normal operation, the frequency of rxoutclk never exceeds 148.5 MHz. Typically, a clock period constraint of 148.5 MHz is applied to rxoutclk and the FPGA design is implemented with the expectation that rxoutclk will never run any faster than that. Using a clock period constraint of 297 MHz on rxoutclk is not an option because the Artix-7 FPGA is not fast enough to support running the SDI core at that frequency. To solve this problem, the SDI control module keeps the RX section of the SDI core in reset from the end of FPGA configuration until rxrate has been successfully changed to a legal value and rxoutclk is running at 148.5 MHz or slower.

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_gtp_full_reset: When asserted High, this input resets both the PLL associated with the RX and then the RX section of the GTP 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_gtp_reset: When asserted High, this input resets only the RX section of the GTP 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.

SDI Wrapper Reset Connections

The two PLLs in each GTP Quad each have their own set of reset and status ports on the GTP wrapper that must be correctly connected to the SDI wrapper. As previously described, the SDI wrapper executes an initialization sequence of both the RX and TX sections of the GTP transceiver after FPGA configuration completes. The PLLs are reset during this sequence. Furthermore, a PLL must be reset following interruptions of or frequency changes to its reference clock.

A separate SDI control module is used for each transceiver in a Quad that is used to implement an SDI interface. Each of these control modules provides input and output ports for resetting and monitoring the PLLs. When multiple transceivers in the same Quad are used for SDI interfaces, the PLL ports of the GTP wrapper and the SDI wrapper must be correctly connected, but those connection requirements vary depending on various aspects of the application. This section describes several different PLL usage models for SDI applications. The following signals on the SDI wrapper are discussed in these use models:

- gtp_rxpllreset: Output port from the SDI wrapper used to reset the PLL used by the GTP RX.
- gtp_txpllreset: Output port from the SDI wrapper used to reset the PLL used by the GTP TX.
- gtp_rxpllock: Input port to the SDI wrapper driven by the appropriate pll0lock or pll1lock GTP common wrapper output.
- gtp_common_wrapper: Input port to the SDI wrapper driven by the appropriate pll0lock or pll1lock GTP transceiver output.
- rx_refclk_stable: Input port to the SDI wrapper that must be asserted High only when the reference clock is stable to the PLL that is providing the GTP RX serial clock.
- tx_refclk_stable: Input port to the SDI wrapper that must be asserted High only when the reference clock is stable to the PLL or PLLs that are providing the GTP TX serial clock.
- rx_gtp_full_reset and rx_gtp_reset: RX section reset request inputs to the SDI wrapper from the user application.
- tx_gtp_full_reset and tx_gtp_reset: TX section reset request inputs to the SDI wrapper from the user application.

GTP PLL Usage Models for SDI Applications

Usage Model 1: A Single Transceiver Active in the Quad with the TX Using Both PLL0 and PLL1

When there is a single transceiver active in the Quad and the TX unit is dynamically switched between PLL0 and PLL1, the RX section of the SDI control module must control the PLL used by the GTP RX. The TX section of the SDI control module controls the reset of the other PLL but observes the locked status of both PLLs. The connections are shown in Figure 6. This figure shows PLL0 as the common PLL used by both the GTP RX and the GTP TX while PLL1 is the second PLL used only by the GTP TX. However, it is equally possible to use PLL1 as the common PLL and PLL0 as the second PLL used by the GTP TX only.

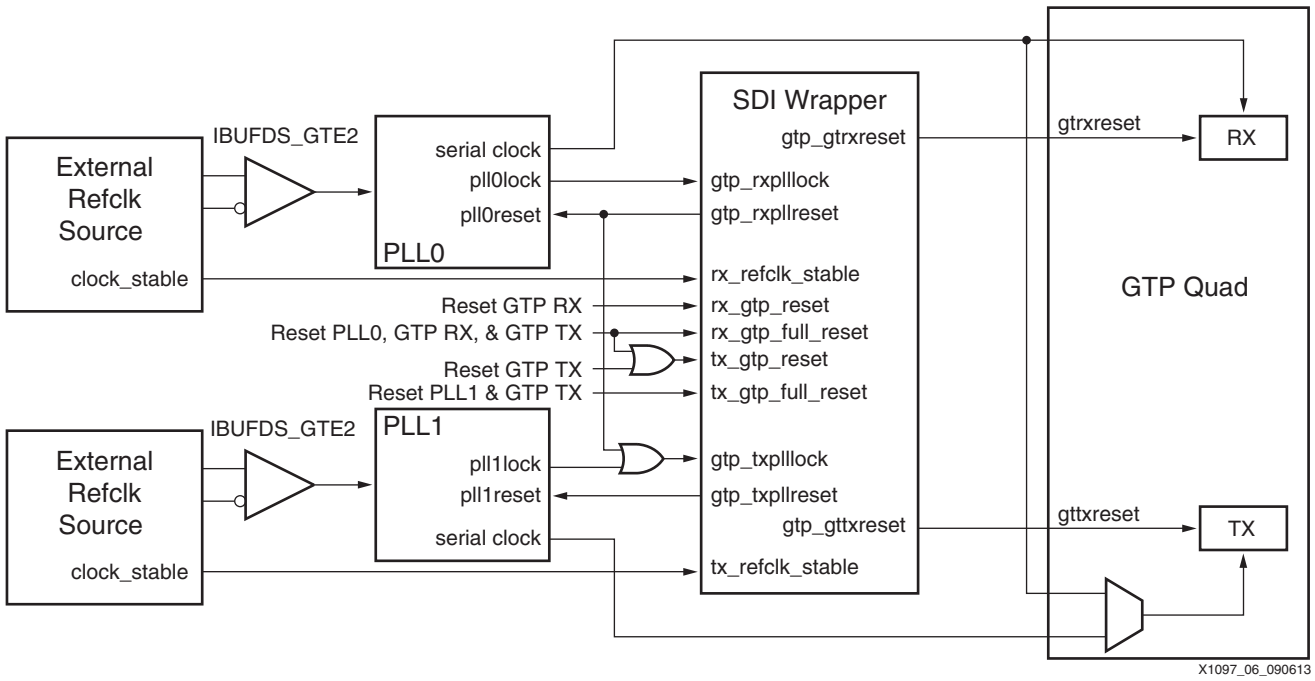


Figure 6: **PLL Usage Model 1**

The following connections must be made:

- The gtp_rxpllreset output of the SDI wrapper must be connected to the pllreset of the PLL used by the RX.
- The gtp_txpllreset output of the SDI wrapper must be connected to the pllreset of the other PLL.
- The gtp_rxplllock input of the SDI wrapper must be connected to the plllock of the PLL used by the RX.
- The gtp_txplllock input of the SDI wrapper must be driven by the logical OR of both plllock signals.
- The rx_refclk_stable input of the SDI wrapper must be asserted High only when the reference clock source to the PLL used by the RX is stable.
- The tx_refclk_stable input of the SDI wrapper must be asserted High only when the reference clock source to the other PLL (the one not used by the RX) is stable.
- When the common PLL needs to be reset due to a reference clock change or interruption, assert the rx_gtp_full_reset input of the SDI wrapper to reset both the common PLL and the GTP RX. Also assert the tx_gtp_reset input of the SDI wrapper to reset the GTP TX without resetting the other PLL.

- When the PLL used only by the TX needs to be reset due to a reference clock change or interruption, assert the tx_gtp_full_reset input of the SDI wrapper to reset both the PLL and the GTP TX.

Usage Model 2: A Single Transceiver Active in the Quad with the RX and TX Using Different PLLs

When there is a single transceiver active in the Quad, and the GTP RX and TX each use a different PLL, the connections are as shown in Figure 7. This figure shows PLL0 as the serial clock source for the GTP RX and PLL1 as the serial clock source for the GTP TX. However, it is simple to use PLL1 for the GTP RX and PLL0 for the GTP TX.

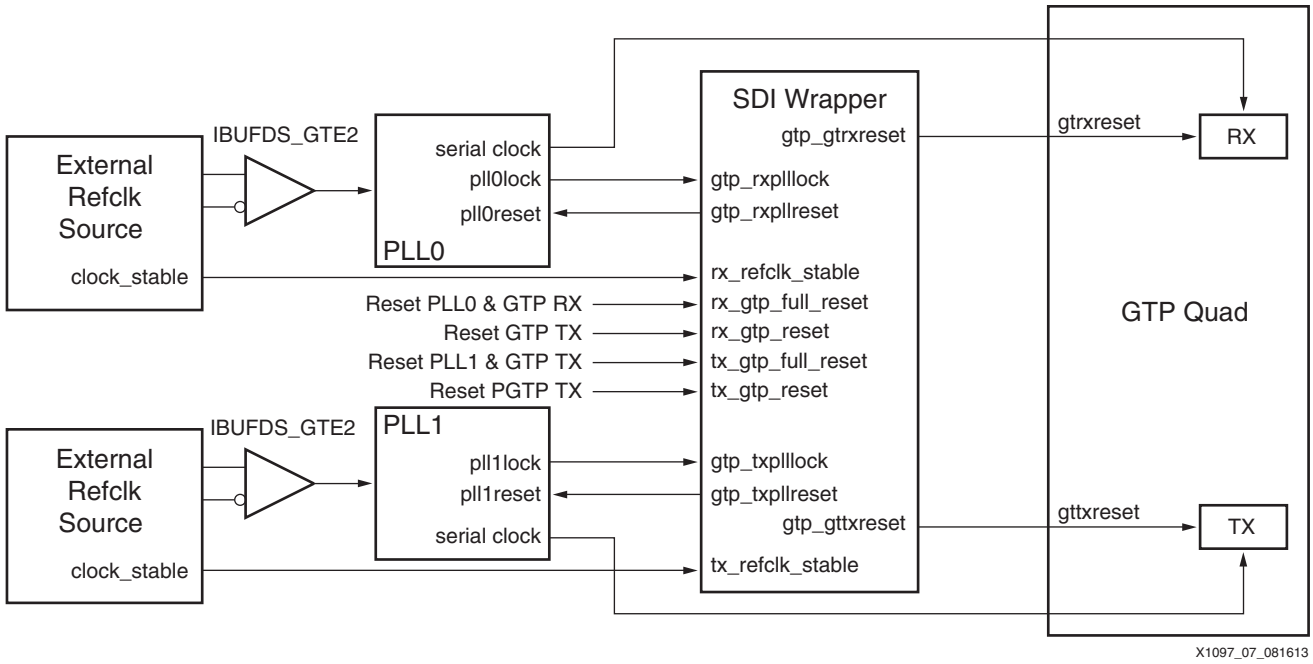


Figure 7: **PLL Usage Model 2**

The following connections must be made:

- The gtp_rxpllreset output of the SDI wrapper must be connected to the pllreset of the PLL used by the RX.
- The gtp_txpllreset output of the SDI wrapper must be connected to the pllreset of the PLL used by the TX.
- The gtp_rxplllock input of the SDI wrapper must be driven by the plllock of the PLL used by the RX.
- The gtp_txplllock input of the SDI wrapper must be driven by the plllock of the PLL used by the TX.
- The rx_refclk_stable input of the SDI wrapper must be asserted High only when the reference clock source to the PLL used by the RX is stable.
- The tx_refclk_stable input of the SDI wrapper must be asserted High only when the reference clock source to the PLL used by the TX is stable.
- When the PLL used by the RX needs to be reset due to a reference clock change or interruption, assert the rx_gtp_full_reset input of the SDI wrapper to reset both the PLL and the GTP RX.
- When the PLL used by the TX needs to be reset due to a reference clock change or interruption, assert the tx_gtp_full_reset input of the SDI wrapper to reset both the PLL and the GTP TX.

Usage Model 3: A Single Transceiver Active in the Quad with the RX and TX Using the Same PLL

When there is a single transceiver active in the Quad and the GTP RX and TX use the same PLL, the RX section of the SDI control module is used to reset that PLL. However, both the GTP RX and GTP TX must monitor the locked status of that PLL as shown in Figure 8. PLL0 is used in the figure, but PLL1 could have been used instead.

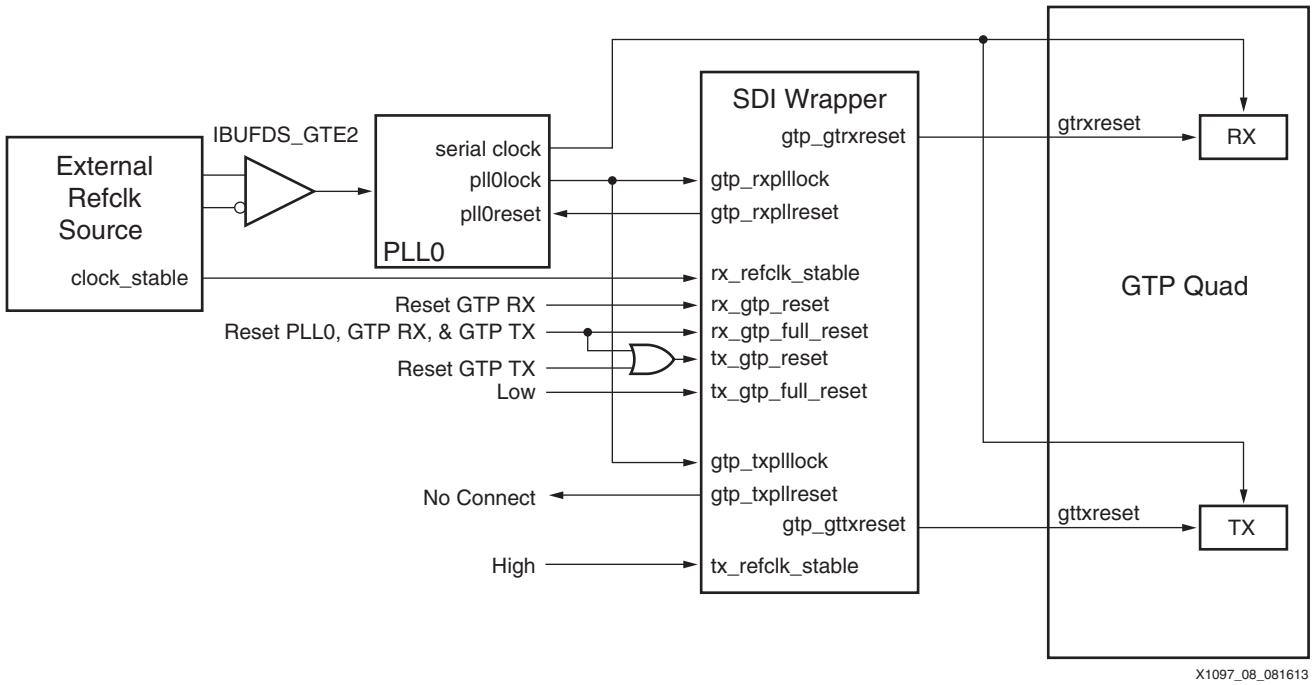


Figure 8: **PLL Usage Model 3**

The following connections must be made:

- The gtp_rxpllreset output of the SDI wrapper must connect to the pllreset of the PLL.
- The gtp_txpllreset output of the SDI wrapper should be left unconnected.
- The gtp_rxpllock input of the SDI wrapper must be driven by the plllock of the PLL.
- The gtp_txplllock input of the SDI wrapper must be driven by the plllock of the PLL.
- The rx_refclk_stable input of the SDI wrapper must be asserted High only when the reference clock source to the PLL is stable.
- The tx_refclk_stable input of the SDI wrapper must be tied High.
- When the PLL needs to be reset due to a reference clock change or interruption, assert the rx_gtp_full_reset input of the SDI wrapper to reset both the PLL and the GTP RX. Also assert the tx_gtp_reset input of the SDI wrapper to reset the GTP TX.

Usage Model 4: Multiple Transceivers Active in a Quad, All RX Use the Same PLL, All TX Use Both PLLs

This usage model covers a very common case where multiple transceivers are active in the Quad, all implementing SDI interfaces. All the active GTP RX units in the Quad use the serial clock from the same PLL. All the active GTP TX units in the are dynamically switched between

both PLLs. This usage module is shown in Figure 9. One of the SDI wrappers is designated as the PLL master and controls the PLL resets.

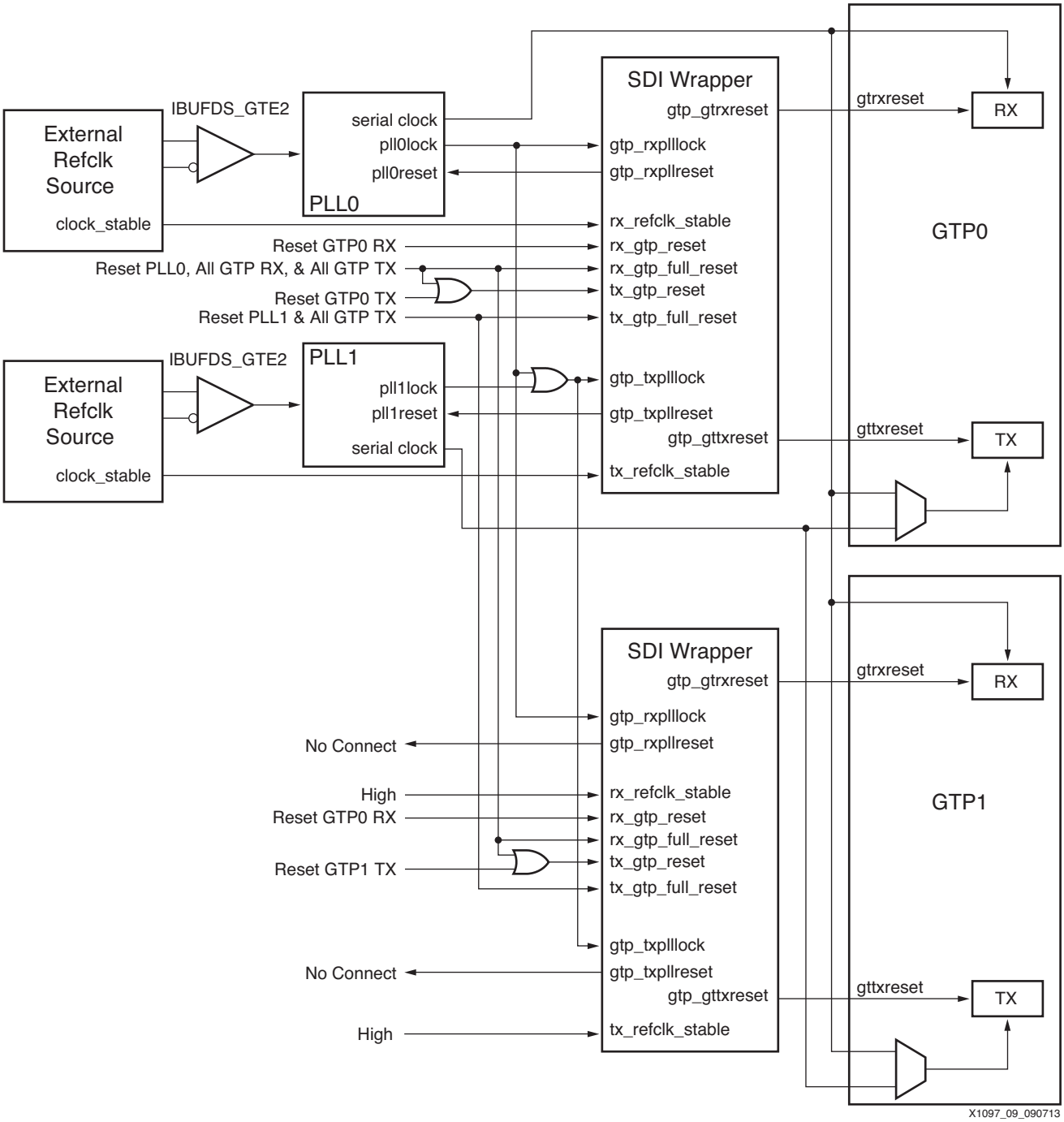


Figure 9: **PLL Usage Model 4**

The following connections must be made:

- The gtp_rxpllreset output of the PLL master SDI wrapper must be connected to the pllreset of the PLL used by the RX units. The gtp_rxpllreset outputs of the other SDI wrappers are left unconnected.
- The gtp_txpllreset output of the PLL master SDI wrapper must be connected to the pllreset of the other PLL. The gtp_txpllreset outputs of the other SDI wrappers are left unconnected.
- The gtp_rxplllock input of ***all*** SDI wrappers must be driven by the plllock of the PLL used by the RX units.
- The gtp_txplllock input of ***all*** SDI wrapper must be driven by the logical OR of both plllock signals.
- The rx_refclk_stable input of the PLL master SDI wrapper must be asserted High only when the reference clock source to the PLL used by the RX is stable. The rx_refclk_stable inputs of all other SDI wrappers should be tied High.
- The tx_refclk_stable input of the PLL master SDI wrapper must be asserted High only when the reference clock source to the other PLL (the one not used by the RX) is stable. The tx_refclk_stable inputs of the other SDI wrappers should be tied High.
- When the common PLL (used by all RX and TX units) needs to be reset due to a reference clock change or interruption, assert the rx_gtp_full_reset input of the PLL master SDI wrapper. The other GTP RX units must also be reset, and this can be done either by asserting their rx_gtp_full_reset or their rx_gtp_reset inputs. Also assert the tx_gtp_reset input of all SDI wrappers to reset the GTP TX units without resetting the second PLL.
- When the PLL used only by the TX units needs to be reset due to a reference clock change or interruption, assert the tx_gtp_full_reset input of the PLL master SDI wrapper. The other GTP TX units must also be reset, and this can be done either by asserting their tx_gtp_full_reset or their tx_gtp_reset inputs.

Usage Model 5: Multiple Transceivers Active in a Quad, All RX Use One PLL, All TX Use the Other PLL

This usage model covers the case where there are multiple transceivers active in a GTP Quad. The serial clocks for all GTP RX units in the Quad come from one PLL while the serial clocks for all GTP TX units in the Quad come from the other PLL, as shown in Figure 10. One SDI wrapper is designated as the PLL master and controls the PLL resets.

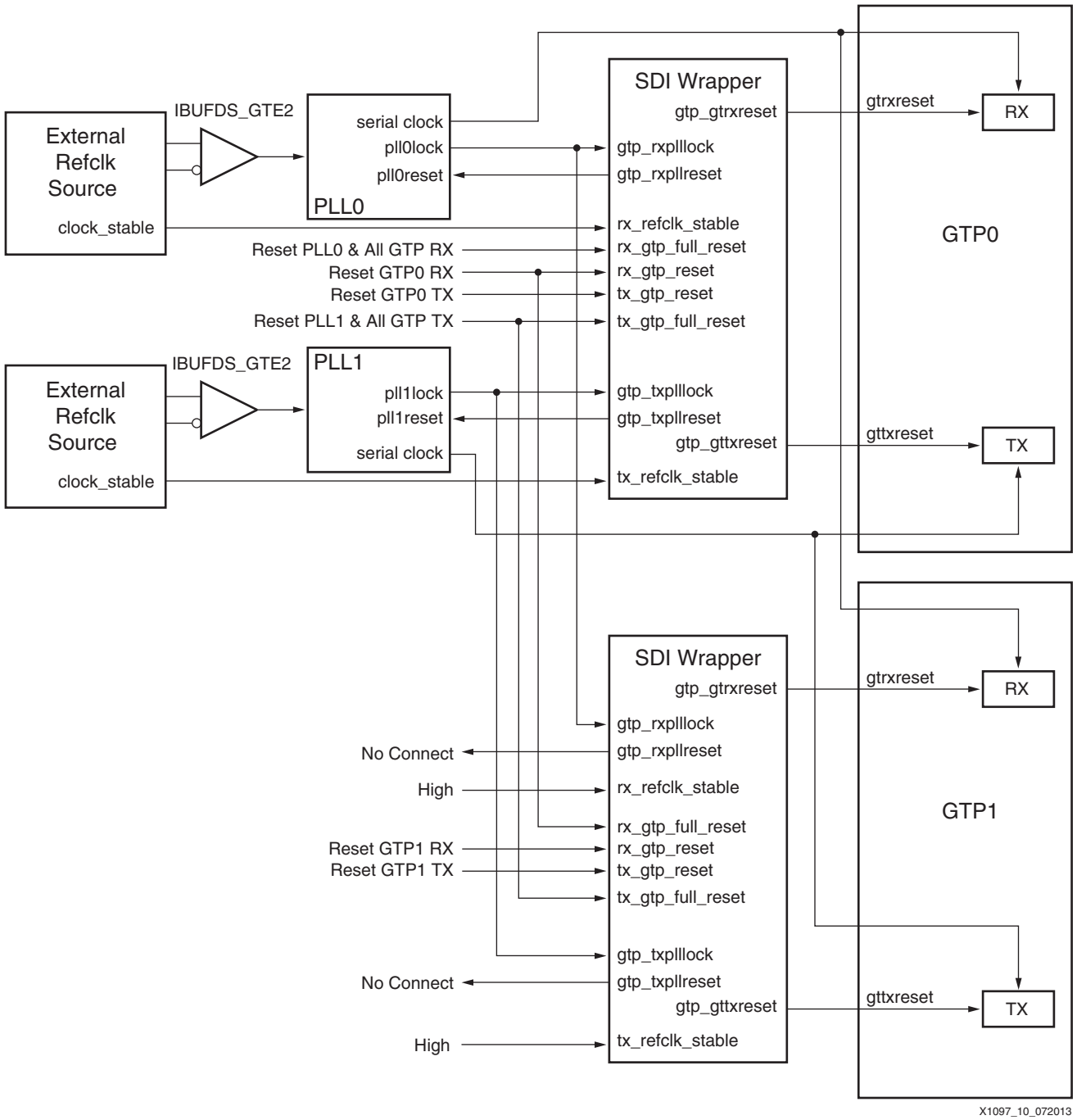


Figure 10: **PLL Usage Model 5**

The following connections must be made:

- The gtp_rxpllreset output of the PLL master SDI wrapper must be connected to the pllreset of the PLL used by all GTP RX units in the Quad.
- The gtp_txpllreset output of the PLL master SDI wrapper must be connected to the pllreset of the PLL used by all GTP TX units in the Quad.
- The gtp_rxplllock input of ***all*** SDI wrappers must be driven by the plllock of the PLL used by the RX units.
- The gtp_txplllock input of ***all*** SDI wrappers must be driven by the plllock of the PLL used by the TX units.
- The rx_refclk_stable input of the PLL master SDI wrapper must be asserted High only when the reference clock source to the PLL used by the RX units is stable. The rx_reflk_stable inputs of all other SDI wrappers should be tied High.
- The tx_refclk_stable input of the PLL master SDI wrapper must be asserted High only when the reference clock source to the PLL used by the TX units is stable. The tx_refclk_stable inputs of all other SDI wrappers should be tied High.
- When the PLL used by the RX units needs to be reset, assert the rx_gtp_full_reset input of the PLL master SDI wrapper. The other GTP RX units must also be reset, and this can be done either by asserting their rx_gtp_full_reset or their rx_gtp_reset inputs.
- When the PLL used by the TX units needs to be reset, assert the tx_gtp_full_reset input of the PLL master SDI wrapper. The other GTP TX units must also be reset, and this can be done either by asserting their tx_gtp_full_reset or their tx_gtp_reset inputs.

Usage Model 6: Multiple Transceivers Active in a Quad, All RX and TX Units Use One Common PLL

This usage model covers the case where there are multiple transceivers active in a GTP Quad and all RX and TX units in the Quad are using a serial clock from one single PLL, as shown in Figure 11. One SDI wrapper is designated as the PLL master, and the RX section of the SDI wrapper controls the PLL reset.

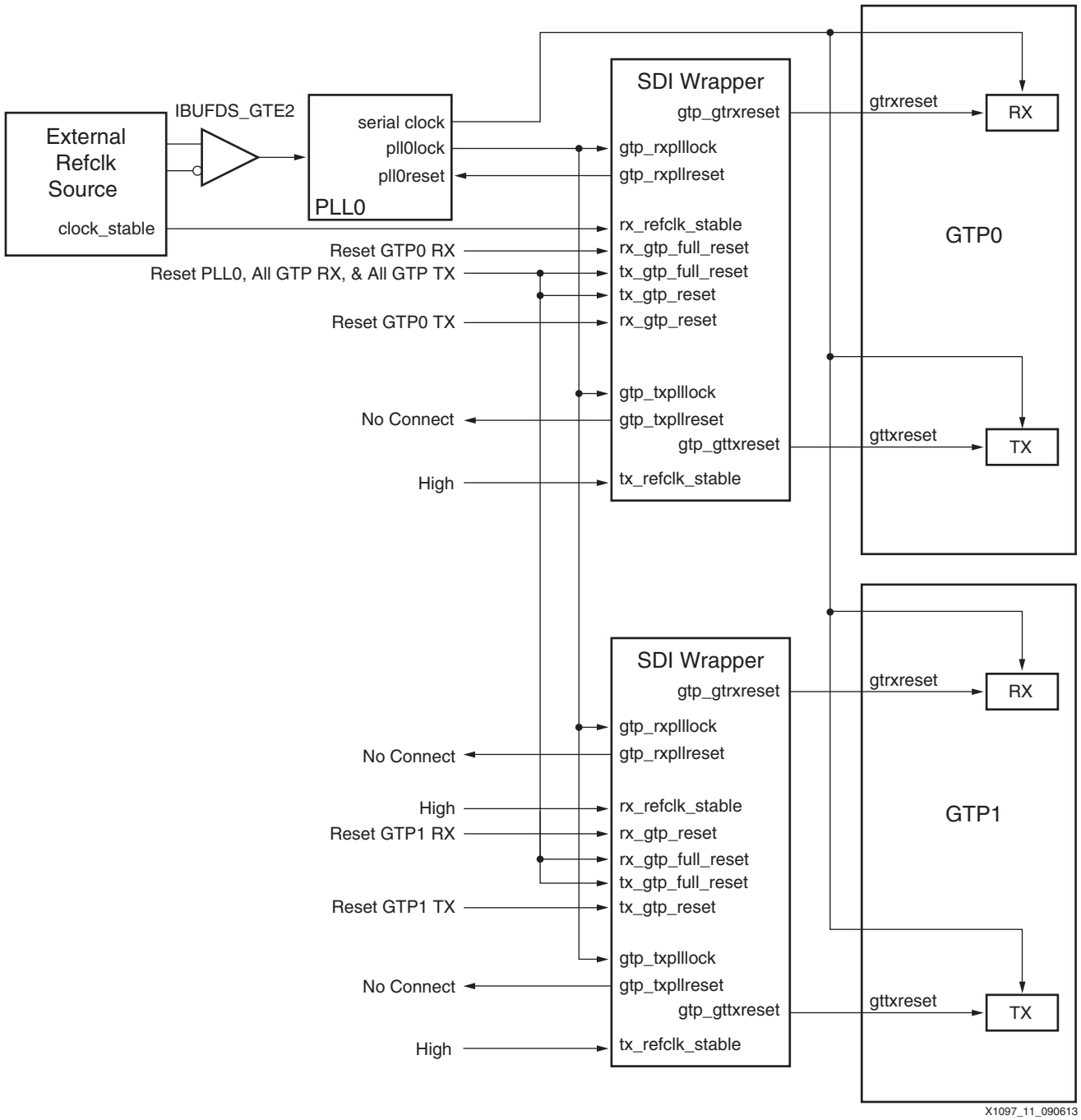


Figure 11: **PLL Usage Model 6**

The following connections must be made:

- The gtp_rxpllreset output of the PLL master SDI wrapper must be connected to the pllreset of the PLL. The gtp_rxpllreset outputs of all other SDI wrappers should be left unconnected.
- The gtp_txpllreset output of ***all*** SDI wrappers should be left unconnected.
- The rx_pllock input of ***all*** SDI wrappers must be driven by the plllock of the PLL.
- The gtp_txplllock input of ***all*** SDI wrappers must be driven by the plllock of the PLL.
- The rx_refclk_stable input of the PLL master SDI wrapper must be asserted High only when the reference clock source to the PLL is stable. The rx_reflk_stable inputs of all other SDI wrappers should be tied High.
- The tx_refclk_stable input of all SDI wrappers must be tied High.
- When the PLL needs to be reset, assert the rx_gtp_full_reset input of the PLL master SDI wrapper. Also assert the rx_gtp_full_reset or rx_gtp_reset of the other SDI wrappers to reset their associated GTP RX units. Assert the gtp_tx_full_reset or gtp_tx_reset inputs of all SDI wrapper to reset their associated GTP TX units.

Usage Model 7: Two Transceivers Active in the Quad Both Implementing Only SDI TX and Each Using a Different PLL

This usage model covers the case where there are exactly two transceivers active in a GTP Quad and each transceiver is only implementing an SDI TX. There are no SDI RX units active. Each active GTP TX uses a different PLL. This case is used when each TX unit must have its own PLL such as in SDI routers. This case is shown in Figure 12.

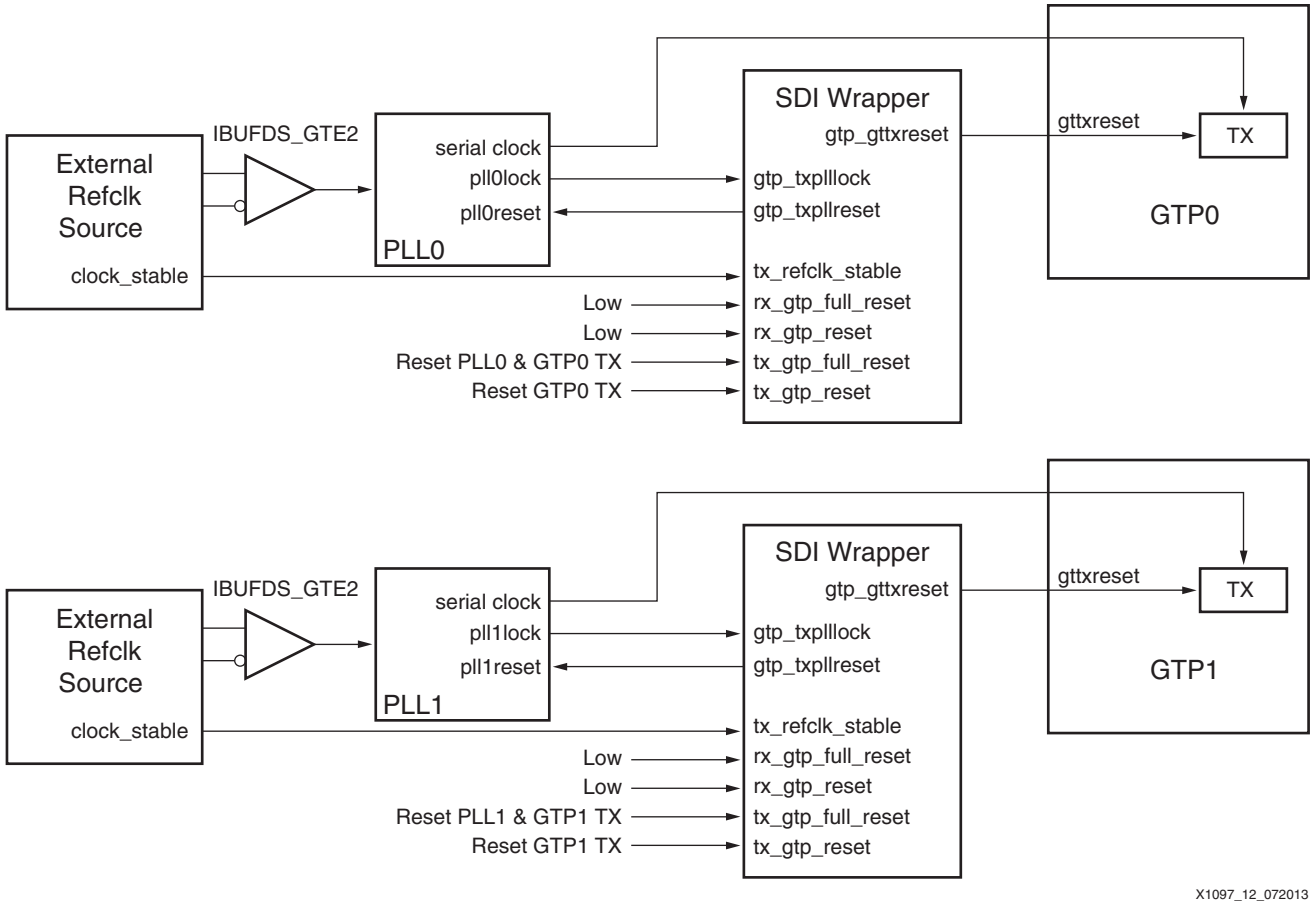


Figure 12: **PLL Usage Model 7**

The following connections must be made:

- The gtp_txpllreset outputs of each SDI wrapper must each be connected to the pllreset input of the associated PLL.
- The gtp_rxpllreset outputs of the SDI wrappers are left unconnected.
- The gtp_txplllock input of each SDI wrapper must be connected to the plllock output of the associated PLL.
- The gtp_rxplllock inputs of the SDI wrappers should be tied High.
- The tx_refclk_stable input of each SDI wrapper must be asserted High only when the reference clock source to the associated PLL is stable.
- The rx_refclk_stable inputs of each SDI wrapper should be tied High.
- When a PLL needs to be reset, assert the tx_gtp_full_reset input of the SDI wrapper associated with that PLL.

SDI Electrical Interface

External SDI cable equalizers and cable drivers are required to convert the serial signals into and out of the GTP 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 GTP transceiver. Appropriate SDI cable equalizers are available from several manufacturers. The differential outputs of these cable equalizers usually must be AC-coupled to the GTP receiver input signals. An example of interfacing a typical SDI cable equalizer to a GTP receiver is shown in Figure 13.

Note: Capacitance values of the coupling capacitors must be large enough to pass the SDI pathological signals without significant signal droop. Typical values used are between 1 µF and 4.7 µF.

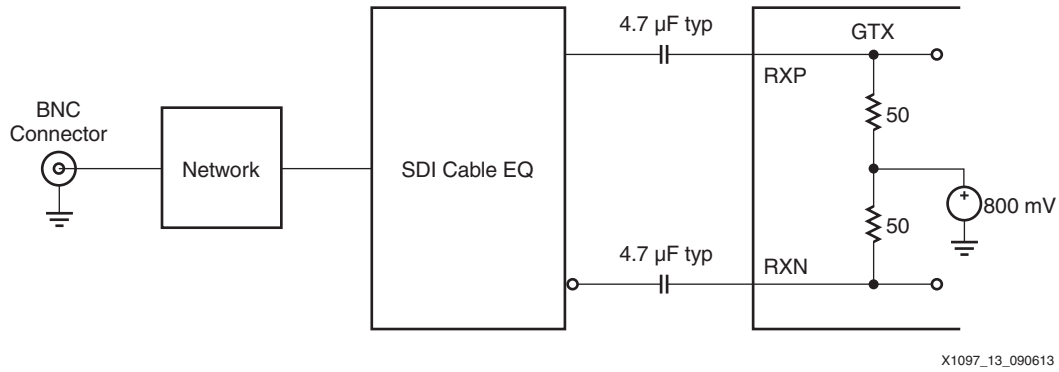


Figure 13: **Interfacing a SDI Cable Equalizer to the GTP Receiver Inputs**

Note: Consult the SDI cable EQ manufacturer's information for the network between the SDI cable EQ and the BNC connector.

The differential inputs of the GTP RX have built-in differential termination. As described in *7 Series FPGAs GTP Transceivers User Guide* (UG482) [Ref 1], RX Termination Use Mode 3 is the recommended termination mode for the GTP RX inputs in SDI applications. The GTP internal programmable termination voltage should be set to 800 mV for SDI applications.

Similarly, the differential serial outputs of the GTP transmitter are connected to the inputs of an SDI cable driver, usually with AC coupling as shown in Figure 14. The cable driver converts the differential signal from the GTP 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.

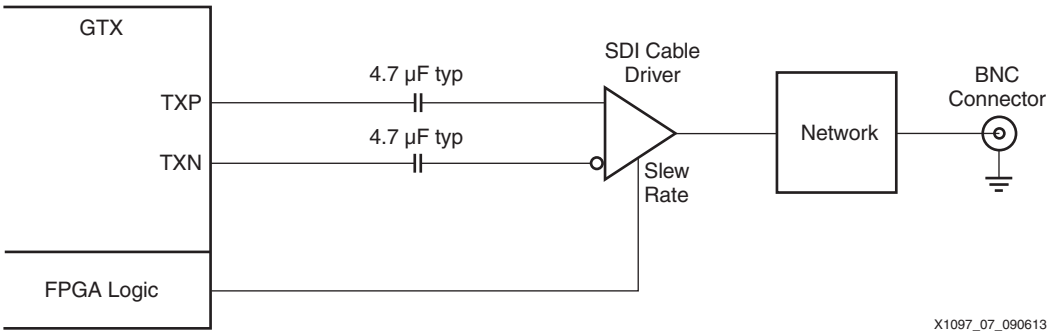


Figure 14: **Interfacing a SDI Cable Driver to the GTP Transmitter Outputs**

Note: Consult the SDI cable EQ manufacturer's information for the network between the SDI cable EQ and the BNC connector.

SD-SDI Considerations

Receiving SD-SDI

The 270 Mb/s bit rate of SD-SDI is below the minimum line rate supported by the GTP RX. In order to receive 270 Mb/s SD-SDI, the GTP RX is used as an asynchronous oversampler to sample the SD-SDI bit stream at 11 times 270 Mb/s (2.97 gigasamples per second) without regard to where bit transitions occur. The CDR unit in the GTP RX is locked to the reference clock by asserting the GTP 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.

A data recovery unit (DRU), implemented in the programmable logic of the FPGA, examines the oversampled SD-SDI data from the GTP RX, determines the best sample to use for each bit, and outputs the recovered data. This DRU is not part of the SDI core, but is provided as part of this application note's SDI control module.

The DRU provided with this application note is a version of the DRU described in *Dynamically Programmable DRU for High-Sped Serial I/O* (XAPP875) [Ref 2] that has been optimized for recovering 270 Mb/s SD-SDI bit streams from 11X oversampled data. The general-purpose DRU described in *Dynamically Programmable DRU for High-Sped Serial I/O* 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 [Ref 5] 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 *Dynamically Programmable DRU for High-Sped Serial I/O*. 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 [Ref 6] is within the supported line rate range of

the GTP transceiver and thus the GTP RX does not need to use the DRU to receive it. However, receiving the 540 Mb/s bit rate without the DRU requires a different reference cock frequency than is used for the other SDI bit rates. Thus, it is usually more convenient to use the DRU from *Dynamically Programmable DRU for High-Sped Serial I/O* 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 GTP 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 GTP transceiver. 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 GTP RX.

Figure 15 is a screen capture from an oscilloscope showing 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 GTP receiver. The rx_ce_sd pulses on either side of the center pulse are always five or six clock cycles away from the center pulse because of the 5/6/5/6 cadence of the rx_ce_sd signal.

Figure 15: **Oscilloscope Capture of SD-SDI Clock Enable**

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 ten 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 file `dru_sim.v` that is included with this application note provides a simplified simulation model of the DRU. This file can be used during simulation to replace `dru.ngc`. However, this simulation model should 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 GTP RX reference clock and the SD-SDI bit stream.

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 GTP TX. To transmit the SD-SDI signal, the GTP 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 GTP TX contains 11 consecutive copies of each bit. The resulting signal output by the GTP TX is a valid 270 Mb/s SD-SDI signal.

Generating an SD-SDI Recovered Clock

In SD-SDI mode, the rxoutclk of the GTP 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 bit stream. The only signal available that actually indicates the data rate of the incoming SD-SDI bit stream 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 GTP RX. The SDI datapath in the SDI core works by 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 GTP TX, 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 GTP TX. Furthermore, the frequency of the recovered clock must be 74.25 MHz or 148.5 MHz so that the GTP 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 Artix-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 National Semiconductor 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 either 74.25 MHz or 148.5 MHz while also filtering out the jitter. The resulting clock is suitable for use as a reference clock for the GTP TX. The pass-through demonstration included with this application note uses an Si5324 to generate a 148.5 MHz reference clock for the GTP 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 GTP RX, doubling its frequency in the case of HD-SDI, thereby producing a low-jitter 148.5 MHz reference clock for the GTP TX.

Another alternative 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 is 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 GTP 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

Artix-7 FPGA MMCM. But, there is a way to generate a recovered SD-SDI clock using a spare GTP TX as shown in Figure 16.

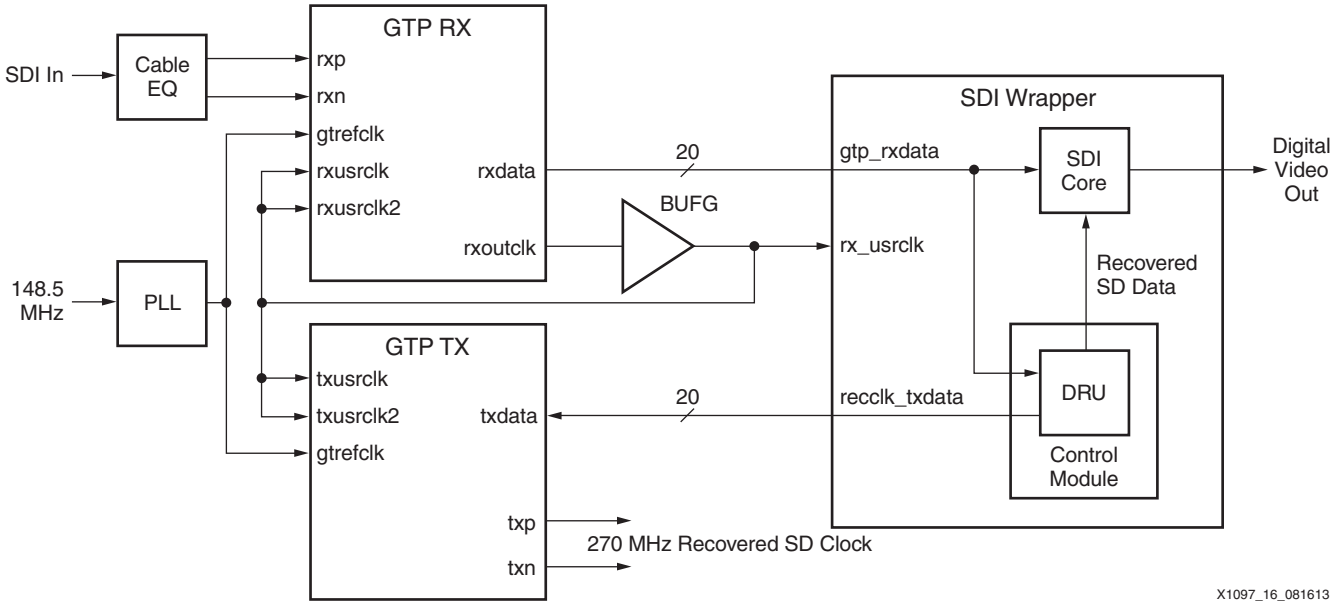


Figure 16: **Using a GTP TX to Generate a SD-SDI Recovered Clock**

The control module's recclk_txdata port can be connected to the txdata port of a spare GTP TX. The GTP TX must use the same reference clock as the GTP RX that is receiving the SDI input signal. The txusrclk and txusrclk2 ports of the GTP TX must be connected to the same clock that is driving the rxusrclk and rxusrclk2 ports of the GTP TX and the rx_usrclk port of the SDI wrapper. The GTP 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 GTP 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 GTP TX serial output pins can be connected to a global or regional clock LVDS input of the Artix-7 FPGA, with appropriate care to properly terminate the current mode logic (CML) outputs and translate them to LVDS. The 270 MHz clock can then 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 an MMCM.

The recclk_txdata port of the DRU is not wired from the SDI control module to an output port in the SDI wrapper supplied with this application note. However, if an application needs to use this feature, the SDI wrapper can easily be edited to add this output port.

The GTP 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 at 2.97 Gb/s with no encoding. The data supplied to the txdata port of the GTP transceiver from the recclk_txdata port of the control module creates a 270 MHz clock on the GTP 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 the input SD-SDI signal. Thus, the cycle-to-cycle jitter on the 270 MHz clock generated by the GTP TX is ±337 ps plus whatever jitter is inherent in the GTP TX output signal (1 bit time at 2.97 Gb/s is 337 ps). This can be seen in Figure 17. The top trace is the 270 MHz clock generated by the GTP 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 17 is the SD-SDI that is being retransmitted by another GTP TX.

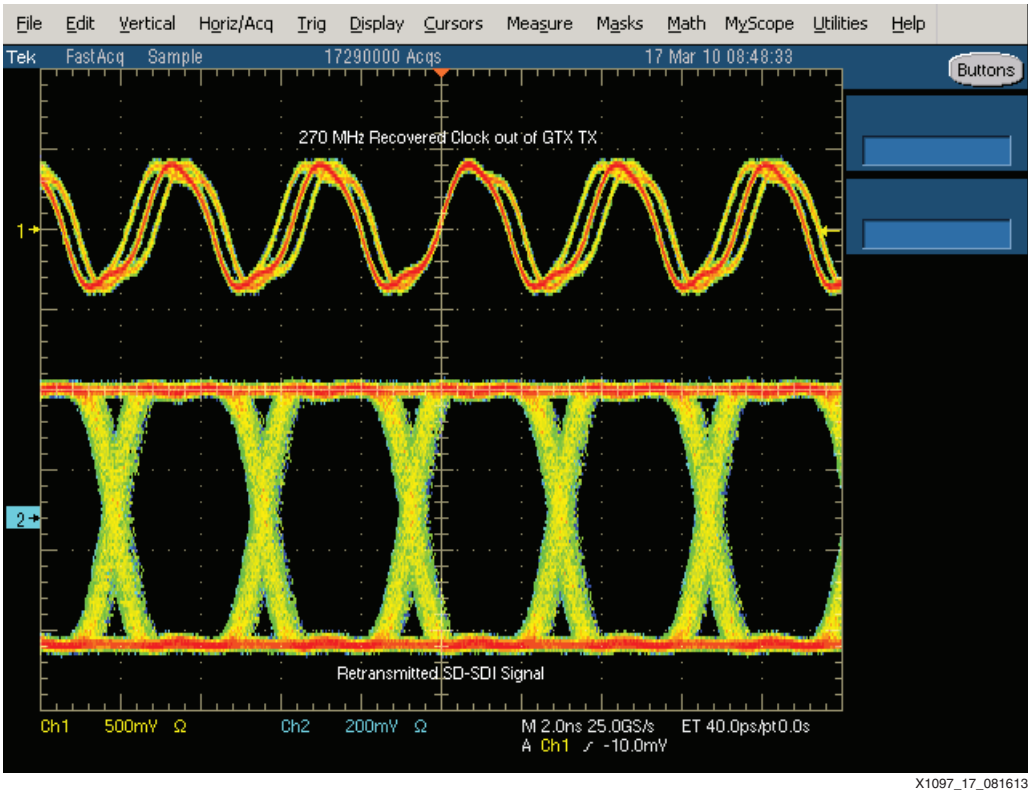


Figure 17: **Recovered SD-SDI Clock from GTP Transceiver**

The recclk_txdata port is not output from the SDI wrapper. This is because this port is not used by most SDI applications. If needed, the SDI wrapper can be edited to add a new port and connect it to the recclk_txdata port of the control module.

RX Bit Rate Detection

The SDI core can automatically determine the SDI mode (SD-SDI, HD-SDI, and 3G-SDI) of the SDI signal coming into the GTP RX. When it is not locked to the current SDI input signal, the SDI core sequences the GTP RX through the three different SDI modes until it detects recognizably good SDI data on the rxdata output port of the GTP transceiver. At that point, the SDI core indicates that it is locked to the SDI signal by asserting its rx_mode_locked output. The SDI core 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 1.485/1.001 Gb/s, and between 2.97 Gb/s and 2.97/1.001 Gb/s. The SDI wrapper output port rx_bit_rate is Low when the input SDI signal's bit rate is either 1.485 Gb/s or 2.97 Gb/s. The rx_bit_rate is High when the input SDI signal's 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/generic 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 Hertz.

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.

Implementing an SDI Interface in Artix-7 FPGAs

Several steps are required to implement an SDI interface in an Artix-7 FPGA design:

1. Generate a GTP wrapper using the 7 Series FPGAs Transceivers Wizard.
2. Generate the SMPTE SD/HD/3G-SDI LogiCORE IP using the CORE Generator tool or from the Vivado IP catalog.
3. Instance the GTP wrapper and the SDI wrapper from this application note into the application.
4. Put the `dru.ngc` file from this application note into the ISE tools project directory or add it as a source in a Vivado tools project (see the `readme.txt` file in `xapp1097.zip` for more information).
5. Apply proper timing constraints for the SDI wrapper.

Generating the GTP Wrapper with Wizard Version 3.0

Use the 7 Series FPGAs Transceivers Wizard to generate a GTP wrapper. The instructions in this section are for the Wizard version 3.0 in the Vivado IP catalog.

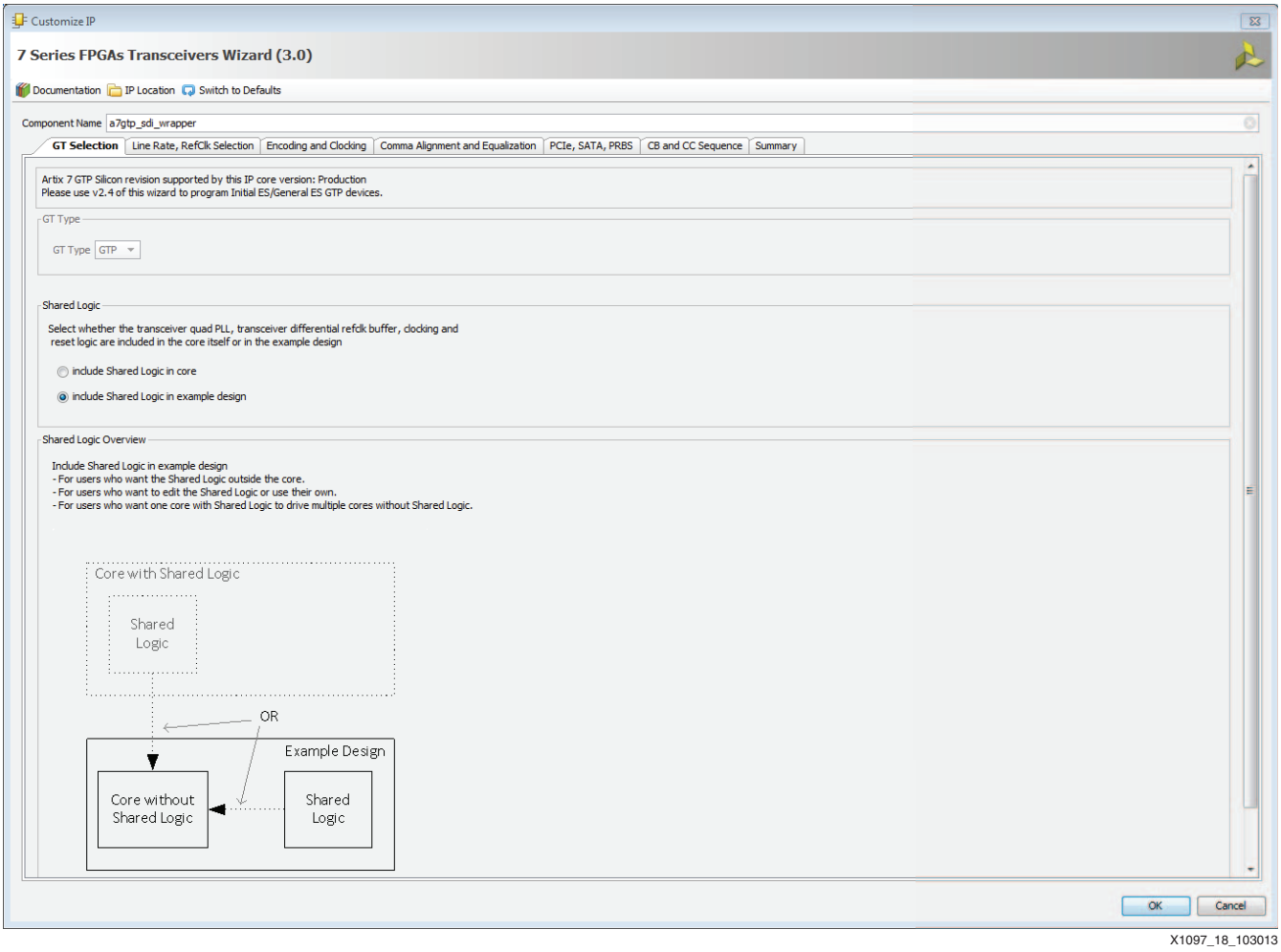
The GTP wrapper generated by version 3.0 or later of the Wizard generates a hierarchy of files collectively called the GTP wrapper. The entire set of files generated by the Wizard are not used for SDI application. The top several levels of the GTP wrapper hierarchy contain initialization logic that is incompatible with SDI applications. The control logic provided with this application note replaces the initialization logic provided with the GTP wrapper. Only the lowest level GTP wrapper generated by the Wizard and its associated files along with the GTP common wrapper are used for SDI applications. Because of this, the GTP wrapper should be generated in a separate Vivado project and then the appropriate GTP wrapper files should be included in the actual Vivado SDI project.

The lowest level GTP wrapper is a wrapper around a single GTP transceiver. This GTP wrapper can be instanced as many times as is needed in the application to implement multiple GTP transceivers for SDI interfaces. The Wizard allows you to create a top-level wrapper with multiple transceivers in the same wrapper, but because the top-level wrapper will not be used, there is no reason to use the Wizard to create a multiple transceiver wrapper. For SDI applications, always create a GTP wrapper with a single transceiver in it.

When generating a GTP wrapper containing transceivers used to implement SDI interfaces, choose the **hd sdi** protocol template in the Wizard. This selects the most common SDI settings for the transceivers. It is recommended that the hd sdi protocol template be used and not the 3g sdi protocol template, even for transceivers that will implement only 3G-SDI.

The following information details the steps required to generate the GTP wrapper using the Wizard version 3.0 from the Vivado IP catalog. The Wizard is found in the `IO Interfaces` folder in the top-level `FPGA Features and Design` folder of the CORE Generator tool or the Vivado IP catalog.

The Wizard launches with the **GT Selection** tab open as shown in Figure 18. Above the tabs is a text field called **Component Name**. The name entered here is used as the name for the GTP wrapper file and the module name of the GTP transceiver.



X1097_18_103013

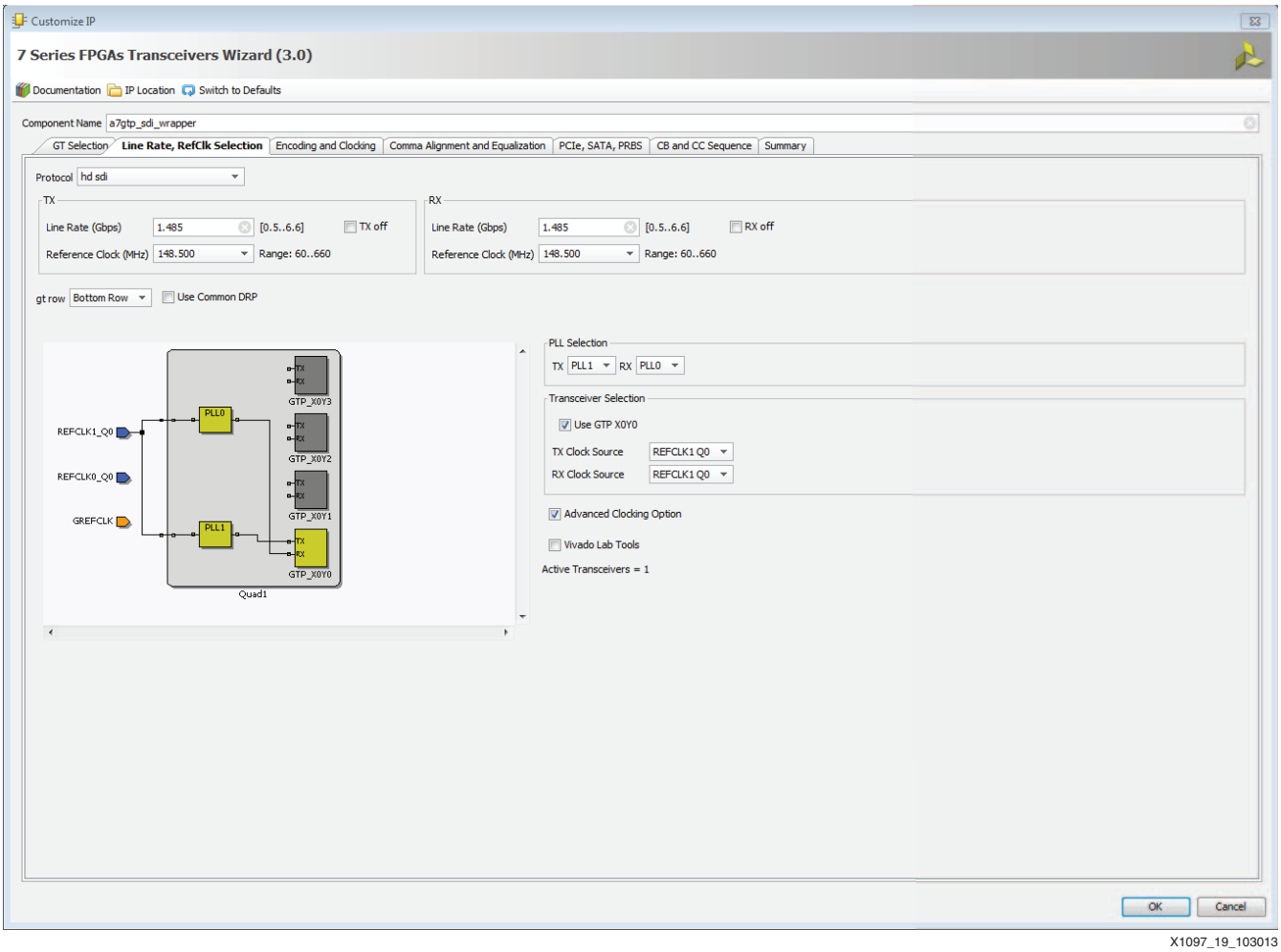
Figure 18: **7 Series FPGAs Transceivers Wizard – GT Selection Tab**

The GT selection tab allows you to specify the type of transceiver. For Artix-7 FPGAs, the only transceiver type allowed is GTP.

In the Shared Logic section, select **include Shared Logic in example design**.

When moving from tab to tab, click on the tabs located under the **Component Name** field. Do not click the **OK** button until all tabs have been correctly setup. The **OK** button closes the Wizard and generates the GT wrapper.

Go to the **Line Rate, RefClk Selection** tab, shown in Figure 19. On this tab, select **hd sdi** from the **Protocol** drop-down list. 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.



X1097_19_103013

Figure 19: **7 Series FPGAs Transceivers Wizard – Line Rate, RefClk Selection Tab**

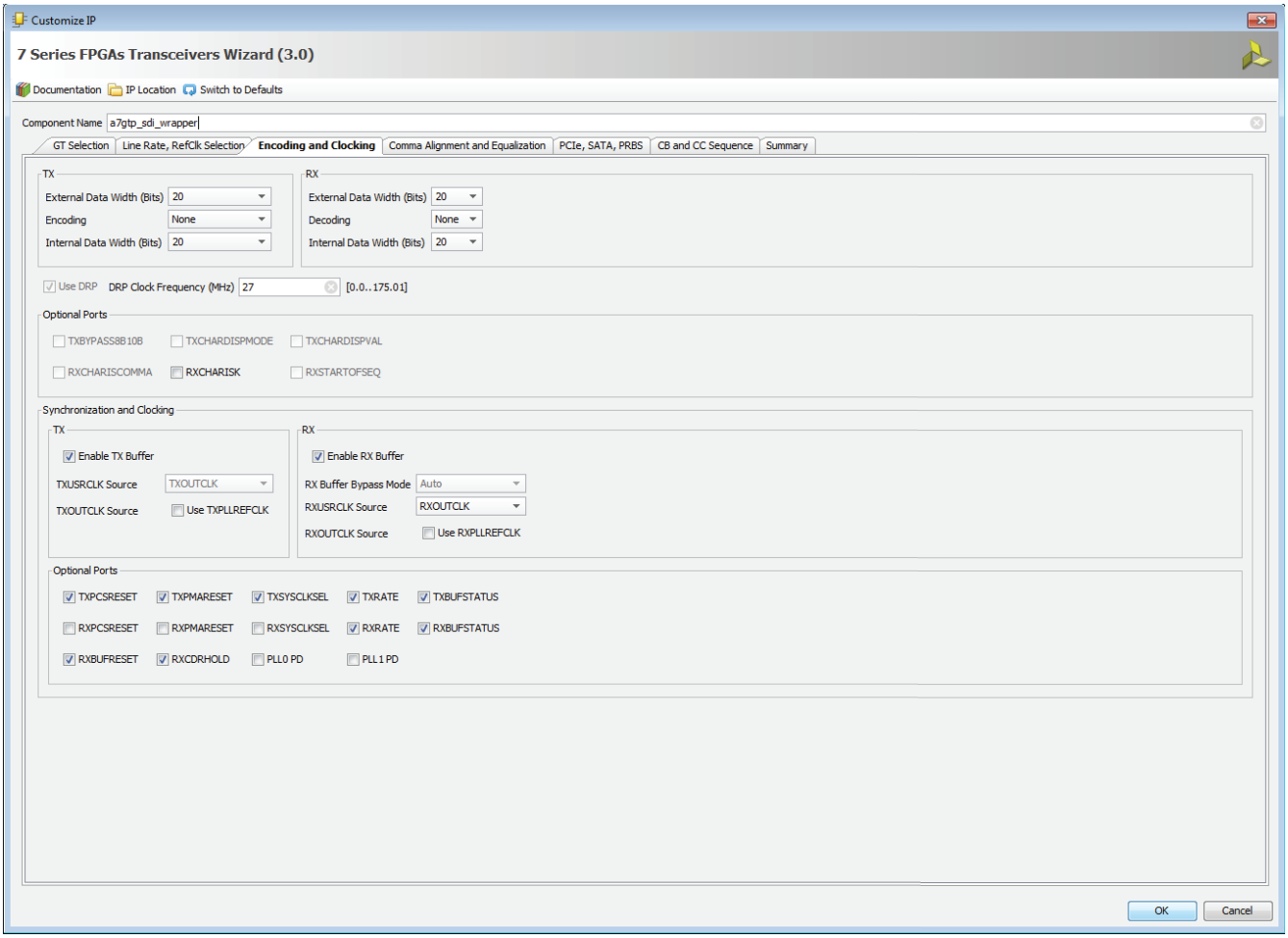
The **TX off** and **RX off** check boxes allow the creation of GTP wrappers with only transmitters (by selecting **RX off**) or only receivers (by selecting **TX off**).

The bottom section of the **Line Rate, RefClk Selection** tab allows the user to choose which GTP transceivers and Quads are included in the GTP wrapper. It also allows the user to choose the reference clocks used by the PLLs and which PLL supplies the serial clock to each transceiver. Applications requiring dynamic switching of the TX between the two PLLs must specify the TX PLL to be the opposite PLL from the one that is assigned to the RX so that both PLLs are enabled in the GTP wrapper. The control module takes care of dynamically switching each TX between the two PLLs.

By default, the same reference clock (REFCLK1) is assigned to both PLLs when this tab opens. In Figure 19, this assignment has been changed and the PLLs are assigned to different reference clocks, as is typically required for most SDI applications. It does not matter which

GTP transceiver is selected. However, it is important to select only one transceiver. It is okay to use the default transceiver that is selected when this tab opens.

Go to the **Encoding and Clocking** tab, shown in Figure 20. The contents of this tab are automatically set up correctly for SDI applications when the HD-SDI protocol was selected. Most of the selections on this tab should not be changed, but there are a few that can be changed.



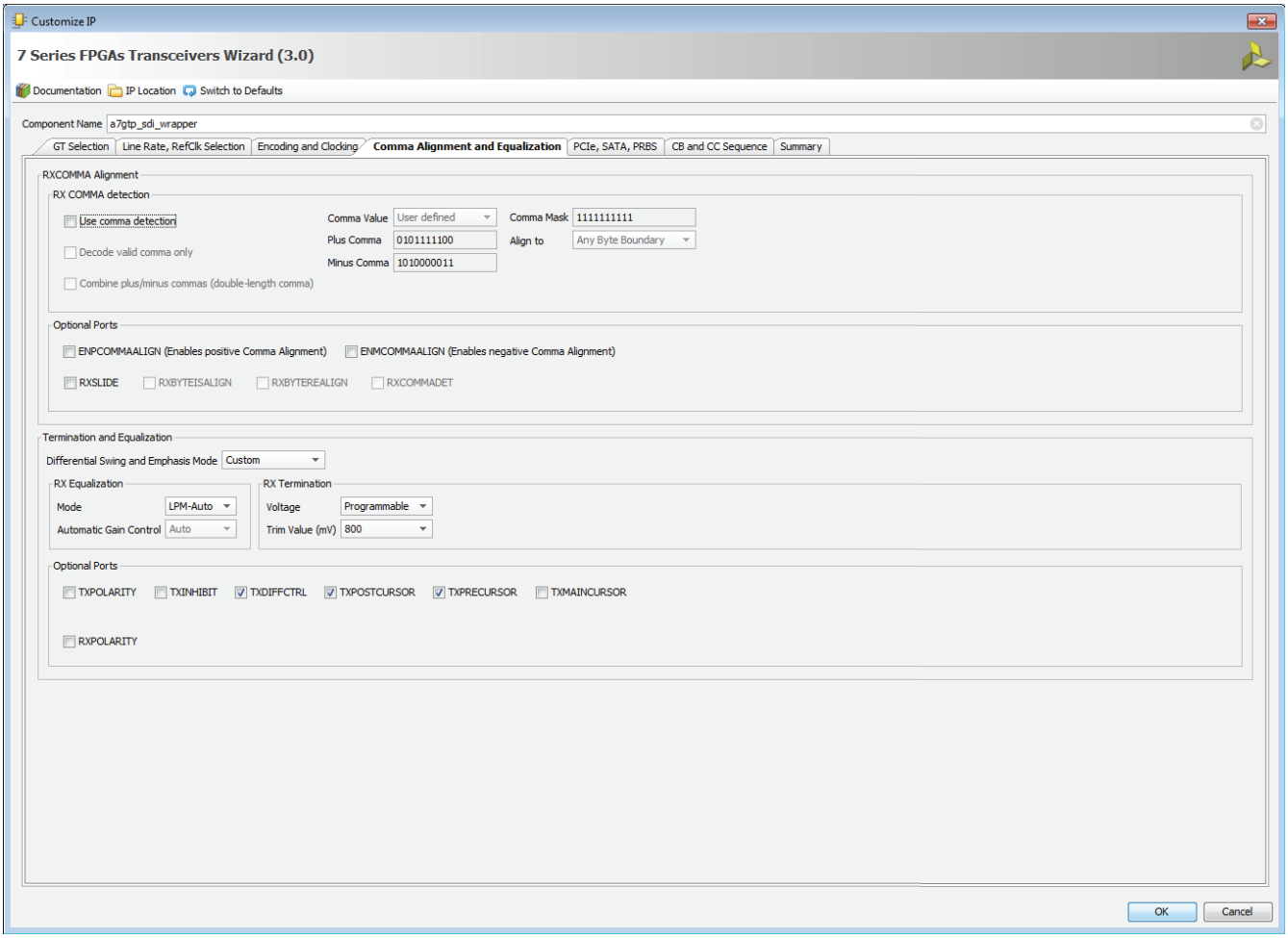
X1097_20_103013

Figure 20: **7 Series FPGAs Transceivers Wizard – Encoding and Clocking Tab**

Use DRP is selected and cannot be changed. However, the frequency of the DRP bus must be set correctly. The GTP wrapper uses the DRPCLK to time the delays of certain sequences. Thus the nominal frequency of the DRPCLK must be specified correctly in the box next to **Use DRP**. In the example shown in Figure 20, the DRPCLK frequency has be modified to 27 MHz from the default of 100 MHz.

If desired, the **PLL0 PD** and **PLL1 PD** optional ports can be selected. These ports allow their respective PLLs to be powered down. In most SDI applications, both PLLs are always used, so these ports typically are not selected for inclusion in the wrapper.

Move to the **Comma Alignment and Equalization** tab, shown in Figure 21. The settings in the **RXCOMMA Alignment** section of this tab should not be changed from the defaults, as shown in Figure 21. In particular, **Use comma detection** and the **RXSLIDE** port must not be enabled.



X1097_21_103013

Figure 21: **7 Series FPGAs Transceivers Wizard – Comma Alignment and Equalization Tab**

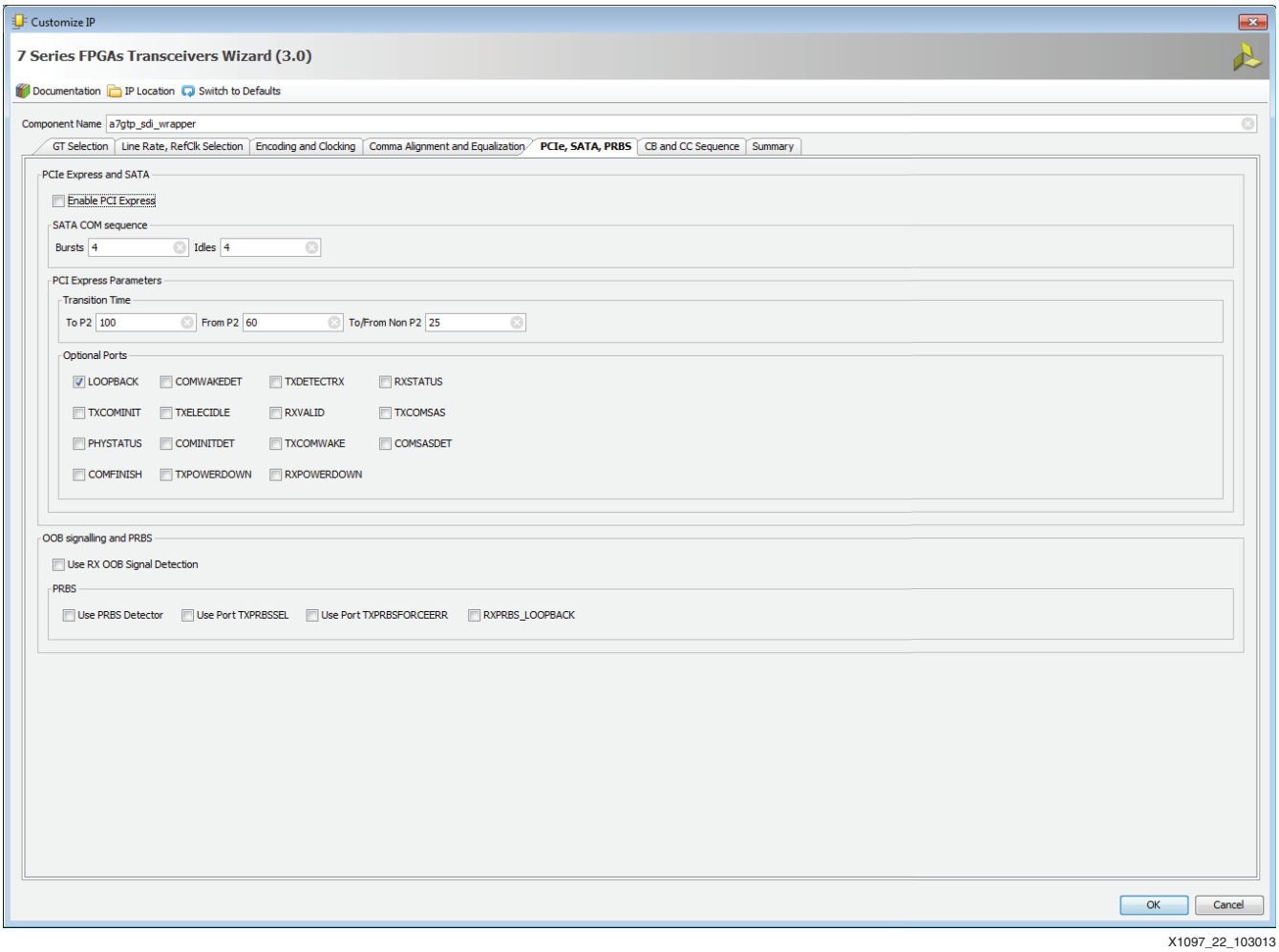
The settings in the **Termination and Equalization** section should not be changed from the defaults shown in Figure 21. In particular, the **RX Equalization Mode** must be set to **LPM-Auto**, the **RX Termination Voltage** must be set to **Programmable**, and the **Trim Value** must be set to **800 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**, **TXPRECURSOR**, and **TXMAINCURSOR** 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.

Go to the **PCIE, SATA, PRBS** tab, shown in Figure 22. Most of the options on this page are not relevant to SDI and should be left at 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 GTP TX is looped back to the GTP 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.



X1097_22_103013

Figure 22: **7 Series FPGAs Transceivers Wizard – PCIE, SATA, PRBS Tab**

At this point, all selections necessary for creating a GTP 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. When the user is satisfied with all selections made in the various tabs, the GTP wrapper can be generated by clicking the **OK** button.

Version 3.0 of the Wizard generates a number of files in a hierarchy of folders. The files that will be used are shown below. The file names are all prefixed by the component name used to generate the GTP wrapper. In this example, that component name is a7gtp_sdi_wrapper.

All file names are shown as Verilog file names with `.v` extensions, but if VHDL is the language selected for the project, the file names would have `.vhd` extensions instead.

In folder <vivado_project>/<vivado_project>.srcs/sources_1/ip/a7gtp_sdi_wrapper:

- a7gtp_sdi_wrapper_gt.v

In folder <vivado_project>/<vivado_project>.srcs/sources_1/ip/a7gtp_sdi_wrapper/a7gtp_sdi_wrapper/example_design:

- a7gtp_sdi_wrapper_gtrxreset_seq.v
- a7gtp_sdi_wrapper_rxrate_seq.v
- a7gtp_sdi_wrapper_sync_block.v

In folder <vivado_project>/<vivado_project>.srcs/sources_1/ip/a7gtp_sdi_wrapper/a7gtp_sdi_wrapper/example_design/support:

- a7gtp_sdi_wrapper_common.v

The last folder, the support folder, might not be present after the Wizard finishes generating the GTP wrapper. If so, in the Vivado tools, right click on the GTP wrapper in the Sources window and select the menu item **Open IP Example Design**. This generates the additional support folder and its contents.

The a7gtp_sdi_wrapper_gt.v file is the wrapper around a single GTP transceiver. This wrapper must be instantiated one or more times in the SDI application, one instance for each GTP transceiver that is used as an SDI interface.

The a7gtp_sdi_wrapper_common.v file is a wrapper around the GTPE2_COMMON primitive that contains the two PLLs for the GTP Quad. This wrapper must be instantiated at least once in the application, and multiple times if more than one GTP Quad is used for SDI interfaces.

When using version 3.0 of the Wizard, the common wrapper might not be entirely correct. This version of the Wizard sets the PLL0REFCLKSEL and PLL1REFCLKSEL ports of the GTPE2_COMMON primitive in the common wrapper to 3'b001. Thus, both PLLs always use the same reference clock, even if different reference clocks were selected for the two PLLs in the Wizard GUI. If different reference clocks are to be used for the two PLLs, the common wrapper must be edited to set the PLL0REFCLKSEL and PLL1REFCLKSEL ports appropriately. The common wrapper used in the SDI demonstrations supplied with this application note has been modified to bring PLL0REFCLKSEL and PLL1REFCLKSEL out as ports of the common wrapper. This allows each application to set these ports as required to select the correct reference clock sources for each PLL.

Generate the SMPTE SD/HD/3G-SDI LogiCORE IP

Use the CORE Generator tool or the Vivado IP catalog to generate the SMPTE SD/HD/3G-SDI core. Do not use the older Triple-Rate SDI core, which is only for Virtex-6 FPGAs. The SMPTE SD/HD/3G-SDI core is the generic SDI core that works with 7 series FPGAs.

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 either Verilog or VHDL, depending on project's preferred language setting. When generated from the Vivado IP catalog, only Verilog files are generated for the SDI core.

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. Even if the RX EDH processor is not included, the SDI core has all RX EDH ports, but they are inactive.

Instancing the GTP and SDI Wrappers

The GTP and SDI wrappers need to be instantiated and interconnected in the user design. It is possible to implement the SDI interface without the SDI wrapper supplied with this application note, but the wrapper simplifies the design because it interconnects the SDI control module and the SDI core. If the wrapper is not used, the user must make all of these connections. The SDI

wrapper file is called a7gtp_sdi_rxtx_wrapper.v (Verilog version) or a7gtp_sdi_rxtx_wrapper.vhd (VHDL version). In addition to the SDI core, it also instances these files:

- a7gtp_sdi_control.v/vhd
- a7gtp_tx_control.v/vhd
- a7gtp_sdi_drp_control.v/.vhd
- a7gtp_sdi_drp_arbit.v/.vhd
- a7gtp_sdi_rx_reset_control.v/.vhd
- sdi_rate_detect.v/.vhd
- dru_bshift10to10.v/.vhd
- dru_maskencoder.v/.vhd
- dru_control.v/.vhd
- dru_rot20.v/.vhd
- dru.v (Verilog only)

The dru.v file is an empty module which, in Verilog, specifies the ports on the precompiled dru.ngc file. When using a7gtp_sdi_rxtx_wrapper.v, dru.v must be included in the project. When using the a7gtp_sdi_rxtx_wrapper.vhd VHDL file, the component definition serves the same purpose as dru.v. Thus, dru.v is not required.

When using the ISE tools, the dru.ngc file included with this application note must be moved or copied into the ISE tools project directory where the tools can find and include it in the design. When using the Vivado tools, the dru.ngc file must be added to the project as a source file just like adding any of the Verilog or VHDL files. The dru.ngc file is a pre-generated and encrypted DRU module.

> ***Caution!*** Do not use the dru_sim.v or dru_sim.vhd files that are included with this application note in a design intended to be used in the actual FPGA. These files are for simulation purposes only. Using them in an actual hardware implementation will result in the SDI receiver not being able to correctly receive SD-SDI signals. For simulation purposes, the dru_sim.v or VHD files can be added to the design instead of the dru.v and dru.ngc files.

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 using the IP catalog 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 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 GTP and SDI wrappers.

Some signals are described as being asserted for some number of video sample periods. A video sample period lasts for differing numbers of cycles of the appropriate clock (either tx_usrclk or rx_usrclk) depending on the SDI mode. In HD-SDI and 3G-SDI level A modes, a sample period lasts one clock cycle. In SD-SDI mode, a sample period is either 5 or 6 clock cycles long and begins and ends with the rising edge of the clock when the clock enable (either tx_ce or rx_ce_sd) is asserted. In 3G-SDI level B mode, a sample period is two clock cycles long as controlled by the assertion of the 3G-SDI data ready signal (either tx_din_rdy or rx_dout_rdy_3g).

Most of the RX and TX ports in this list are wired directly to the ports of the same name on the SDI core that is instantiated inside the SDI wrapper. Timing diagrams of the video and video timing signals can be found in the *SMPTE SD/HD/3G-SDI Product Guide* (PG071) [Ref 4].

Table 1: **SDI Wrapper Port List**

Port Name	I/O	Width	Description
clk	In	1	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/generic FXDCLK_FREQ. If the clock frequency does not closely match the frequency specified by FXDCLK_FREQ, the timing delays generated by the wrapper will not be correct and the RX bit rate detection circuit might not function.
Receive Ports			
rx_rst	In	1	This synchronous reset input resets the receiver section of the SDI core. It can normally be hardwired Low because a reset is not required. Immediately after FPGA configuration until the GTP RX is fully initialized, the SDI wrapper keeps the RX section of the SDI core in reset. After the GTP 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 initiate a reset of the GTP 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 GTP transceiver, usually the rxoutclk of the GTP 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 inputs and outputs of the wrapper prefixed with **rx**_ are synchronous with this clock, unless otherwise noted.
rx_gtp_full_reset	In	1	When this input is asserted High, a full GTP RX reset sequence is initiated. First, if the gtp_rxpllreset output of this module is connected to the PLL reset input, the PLL is reset. After the PLL locks to the reference clock input, the GTP RX is reset using the GTP's gtrxreset. Completion of this reset sequence is indicated by assertion of the rx_change_done output. The signal connected to the rx_gtp_full_reset input must be synchronous with the gtp_drpclk clock.
rx_gtp_reset	In	1	When this input is asserted High, the GTP RX is reset using the GTP's gtrxreset. If the PLL providing the serial clock to the GTP RX is not locked, the gtrxreset sequence will not complete until that PLL is locked. Completion of this reset sequence is indicated by assertion of the rx_change_done output. The signal connected to the rx_gtp_reset input must be synchronous with the gtp_drpclk clock.
rx_fabric_reset_out	Out	1	Following the completion of FPGA configuration, this output is asserted High and remains High until the GTP RX is fully initialized. During portions of this time, the frequency of the GTP rxoutclk is 297 MHz. To prevent problems with any modules that are designed for a maximum rxoutclk frequency of 148.5 MHz, the rx_fabric_reset_out signal can be used to keep those modules reset during this 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 GTP RX in reset until the PLL's reference clock is stable. If this SDI wrapper is controlling the PLL reset, the rx_refclk_stable input must be kept Low until the PLL's 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_gtp_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)***

Port Name	I/O	Width	Description
rx_frame_en	In	1	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): end of active video (EAV), or start of active video (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.
rx_mode_en	In	3	This port has unary bits to enable reception of each of the three SDI modes: • Bit 0 enables HD-SDI mode • Bit 1 enables SD-SDI mode • Bit 2 enables 3G-SDI mode 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.
rx_mode	Out	2	This output port indicates the current SDI mode of the SDI RX: • `00` = HD-SDI • `01` = SD-SDI • `10` = 3G-SDI 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.
rx_mode_hd rx_mode_sd rx_mode_3g	Out	1	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.
rx_mode_locked	Out	1	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.
rx_bit_rate	Out	1	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. HD-SDI mode: • rx_bit_rate = 0: Bit rate = 1.485 Gb/s • rx_bit_rate = 1: Bit rate = 1.485/1.001 Gb/s 3G-SDI mode: • rx_bit_rate = 0: Bit rate = 2.97 Gb/s • rx_bit_rate = 1: Bit rate = 2.97/1.001 Gb/s
rx_t_locked	Out	1	This output is High when the transport detection function in the SDI RX has identified the transport format of the SDI signal.
rx_t_family	Out	4	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 4 for the encoding of this port.

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

Port Name	I/O	Width	Description
rx_t_rate	Out	4	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 5 for the encoding of this port. This output is only valid when rx_t_locked is High.
rx_t_scan	Out	1	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.
rx_level_b_3g	Out	1	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).
rx_ce_sd	Out	1	This output is a clock enable for 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.
rx_nsp	Out	1	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).
rx_line_a	Out	11	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 mode, the output value is the line number from the Y data stream of link A or 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 or dual-link HD-SDI, the output value on this port is the interface line number, not the picture line number.
rx_a_vpid	Out	32	All four user data bytes of the SMPTE ST 352 [Ref 7] 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 embedded 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 mode, the output data is the ST 352 data bytes captured from data stream 1 of link A (dual-link streams) or HD-SDI signal 1 (dual HD-SDI signals).
rx_a_vpid_valid	Out	1	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.
rx_b_vpid	Out	32	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 mode, the output data is the ST 352 data bytes captured from data stream 1 of link B (dual-link streams) or HD-SDI signal 2 (dual HD-SDI signals).
rx_b_vpid_valid	Out	1	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.
rx_crc_err_a	Out	1	This output is asserted High for one video sample period when a cyclic redundancy check (CRC) error is detected on the previous video line. For 3G-SDI level B mode, this output indicates CRC errors on data stream 1 only. There is a second output called rx_crc_err_b that indicates CRC errors on data stream 2 for 3G-SDI level B 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 1: **SDI Wrapper Port List *(Cont'd)***

Port Name	I/O	Width	Description
rx_ds1a	Out	10	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/C_B/C_R 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
rx_ds2a	Out	10	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 C_B and C_R 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 C_B and C_R components of HD-SDI signal 1
rx_eav	Out	1	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).
rx_sav	Out	1	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.
rx_trs	Out	1	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.
rx_line_b	Out	11	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 or HD-SDI signal 2. 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.
rx_dout_rdy_3g	Out	1	In 3G-SDI level B mode, the output data rate is 74.25 MHz, but the rx_usrclk frequency is 148.5 MHz. The rx_dout_rdy_3g output is asserted every other cycle of rx_usrclk 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.
rx_crc_err_b	Out	1	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.
rx_ds1b	Out	10	This output is only valid in 3G-SDI level B mode. The data stream output on this port is: • 3G-SDI level B-DL: Data stream 1 of link B • 3G-SDI level B-DS: Y component of HD-SDI signal 2
rx_ds2b	Out	10	This output is only valid in 3G-SDI level B mode. The data stream output on this port is: • 3G-SDI level B-DL: Data stream 2 of link B • 3G-SDI level B-DS: Interleaved C_B and C_R components of HD-SDI signal 2
rx_edh_errcnt_en	In	16	This input controls which EDH error conditions increment the EDH error counter. See Table 6 for more details.
rx_edh_clr_errcnt	In	1	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.

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

Port Name	I/O	Width	Description
rx_edh_ap	Out	1	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.
rx_edh_ff	Out	1	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.
rx_edh_anc	Out	1	This output is asserted High when an ancillary data packet checksum error is detected.
rx_edh_ap_flags	Out	5	The active picture error flag bits from the most recently received EDH packet are output on this port. See Table 7 for more information.
rx_edh_ff_flags	Out	5	The full field error flag bits from the most recently received EDH packet are output on this port. See Table 7 for more information.
rx_edh_anc_flags	Out	5	The ancillary data error flag bits from the most recently received EDH packet are output on this port. See Table 7 for more information.
rx_edh_packet_flags	Out	4	This port outputs four error flags related to the most recently received EDH packet. See Table 8 for more information.
rx_edh_errcnt	Out	16	This is the SD-SDI EDH error counter. It increments once for each field when any of the error conditions enabled by the rx_edh_err_en port occur.
rx_change_done	Out	1	This output is Low during those periods when the SDI RX is being initialized, reset, or when the rxrate port of the GTP transceiver is being changed. If the sequence completes successfully, the rx_change_done output is asserted High to indicate successful completion. This output is synchronous with the gtp_drpclk.
rx_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 GTP RX initialization or reset sequence or an rxrate 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. This output is synchronous with the gtp_drpclk.
rx_change_fail_code	Out	3	When the rx_change_fail port is High, this port indicates the nature of the sequence failure. See Table 9 for encoding of this port. This output is synchronous with the gtp_drpclk.
Transmit Ports			
tx_rst	In	1	This synchronous reset input resets the transmitter section of the SDI core. It can normally be hardwired Low because a reset is 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 GTP 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.
tx_usrclk	In	1	This input must be driven by the same clock that is driving the txusrclk port of the GTP transceiver: usually txoutclk of the GTP 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 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 inputs and outputs of the SDI wrapper prefixed with **tx**_ are synchronous with this clock unless otherwise noted.

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

Port Name	I/O	Width	Description
tx_gtp_full_reset	In	1	When this input is asserted High, a full GTP TX reset sequence is initiated. First, if the gtp_txpllreset output of this module is connected to the PLL reset input, the PLL is reset. After the PLL locks to the reference clock input, the GTP TX is reset using the GTP transceiver's gttxreset. Completion of this reset sequence is indicated by assertion of the tx_change_done output. The signal connected to this input must be synchronous with the gtp_drpclk clock.
tx_gtp_reset	In	1	When this input is asserted High, the GTP TX is reset using the GTP transceiver's gttxreset. If the PLL providing the serial clock to the GTP TX is not locked, the gttxreset sequence does not complete until that PLL is locked. Completion of this reset sequence is indicated by assertion of the tx_change_done output. The signal connected to this input must be synchronous with the gtp_drpclk clock.
tx_refclk_stable	In	1	This input is used by the TX initialization logic to keep the PLL that is providing the serial clock to the GTP TX in reset until the PLL's reference clock is stable. If this SDI wrapper is controlling the PLL reset, then the tx_refclk_stable input must be kept Low until the PLL's 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_gtp_full_reset input until the tx_refclk_stable input is High. This input is treated as an asynchronous input.
tx_ce	In	3	This is the clock enable input for the transmitter section of the SDI core. The tx_ce input 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.
tx_din_rdy	In	1	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.
tx_mode	In	2	This input port is used to select the SDI transmitter's mode: • `00` = HD-SDI (including dual-link HD-SDI) • `01` = SD-SDI • `10` = 3G-SDI • `11` = Invalid
tx_level_b_3g	In	1	In 3G-SDI mode, this input determines whether the SDI transmitter is configured for level A (Low) or for level B (High).
tx_m	In	1	This port is used to select which PLL clock is used by the GTP TX. This input causes the SDI wrapper's gtp_txsysclksel output port to change in order to change the GTP TX PLL clock select multiplexer. 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 entirely governed by the frequency of the two serial clocks provided to the GTP TX and the values of the txsysclksel_M_0 and txsysclksel_M_1 parameters.
tx_insert_crc	In	1	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 asserted High or passed through unchanged if tx_insert_crc is Low.

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

Port Name	I/O	Width	Description
tx_insert_ln	In	1	When this input is High, the SDI TX inserts line number 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 embedded, this input should be asserted High, and valid line numbers must be supplied on the tx_line_a and tx_line_b port. If the data streams entering the SDI TX input port already have line numbers embedded, those line numbers are overwritten if tx_insert_ln is High or passed through unchanged if tx_insert_ln is Low.
tx_insert_edh	In	1	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 embedded, it is overwritten with a new packet if tx_insert_edh is High or passed through unchanged if tx_insert_edh is Low.
tx_insert_vpid	In	1	When this input is High, SMPTE ST 352 [Ref 7] packets are inserted into the 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.
tx_overwrite_vpid	In	1	If this input is High and the tx_insert_vpid port is High, SMPTE ST 352 packets already present in the 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.
tx_video_a_y_in	In	10	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
tx_video_a_c_in	In	10	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 C_B and C_R 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 C_B and C_R components of HD-SDI signal 1
tx_video_ b_y_in	In	10	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.
tx_video_b_c_in	In	10	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 C_B and C_R components of HD-SDI signal 2 • For other SDI modes, this input port is unused.

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

Port Name	I/O	Width	Description
tx_line_a	In	11	The current line number must be provided to the module through this port if either ST 352 [Ref 7] 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.
tx_line_b	In	11	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.
tx_vpid_byte1	In	8	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.
tx_vpid_byte2	In	8	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.
tx_vpid_byte3	In	8	The value on this port is inserted as the third user data word of the ST 352 [Ref 7] 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.
tx_vpid_byte4a	In	8	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.
tx_vpid_byte4b	In	8	The 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.
tx_vpid_line_f1	In	11	The ST 352 packet is inserted in the HANC space of the line number specified by this input port. For interlaced video, this input port specifies a line number in Field 1. For progressive video, 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.

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

Port Name	I/O	Width	Description
tx_vpid_line_f2	In	11	For interlaced video, an ST 352 packet is inserted on the line number in field 2 indicated by this value. For progressive video, 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.
tx_vpid_line_f2_en	In	1	This input controls whether or not ST 352 packets are inserted on the line indicated by tx_vpid_line_f2. For interlaced video, this input must be High. For progressive video, this input must be Low. For progressive video transported on an interlaced transport, such as 1080p 60 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.
tx_ds1a_out	Out	10	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, they should be inserted into the data stream output on this port so that ST 352 packets have already been inserted into the data streams. 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
tx_ds2a_out	Out	10	This is the link A data stream 2 output. The data stream output on this port comes from the ST 352 [Ref 7] packet insertion module. If the application needs to insert ancillary data packets, they should be inserted into the data stream output on this port so that ST 352 packets have already been inserted into the data streams. The resulting data stream after ancillary data insertion by the application should then be supplied to the tx_ds2a_in port. The data on this port depends on the SDI mode: • HD-SDI: Interleaved C_B/C_R component • Dual-link HD-SDI or 3G-SDI level B-DL: Data stream 2 of link A • 3G-SDI level B-DS: Interleaved C_B/C_R component data stream of HD-SDI signal 1
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, they should be inserted into the data stream output on this port so that ST 352 packets have already been inserted into the data streams. 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 • For other SDI modes, this input port is unused

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

Port Name	I/O	Width	Description
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, they should be inserted into the data stream output on this port so that ST 352 packets have already been inserted into the data streams. 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 carrying dual-link HD-SDI: Data stream 2 of link B • 3G-SDI level B carrying dual HD-SDI signals: Interleaved C_B/C_R 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_dsxx_in ports. When no ancillary data insertion is required, the tx_use_dsin input is set Low and the tx_dsxx_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
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: • HD-SDI: Interleaved C_B/C_R component • Dual-link HD-SDI or 3G-SDI level B-DL: Data stream 2 of link A • 3G-SDI level B-DS: Interleaved C_B/C_R 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 carrying dual-link HD-SDI: Data stream 2 of link B • 3G-SDI level B carrying dual HD-SDI signals: Interleaved C_B/C_R component of HD-SDI signal 2 • For other SDI modes, this input port is unused

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

Port Name	I/O	Width	Description
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 bit stream is formed incorrectly. The tx_ce_align_err goes 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 SDI TX is being initialized or reset or the GTP txrate or txsysclksel 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 gtp_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 GTP TX initialization or reset sequence or a dynamic change of the GTP txrate or txsysclksel 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. This output is synchronous with the gtp_drpclk.
tx_change_fail_code	Out	3	When the tx_change_fail port is High, this port indicates the nature of the sequence failure. See Table 10 for encoding of this port. This output is synchronous with the gtp_drpclk.
Ports That Connect to the GTP RX			
gtp_rxdata	In	20	Connect this port to the rxdata port of the GTP transceiver.
gtp_rxpllreset	In	1	This port is used to reset the PLL supplying the serial clock to the GTP RX. If this SDI wrapper is acting as the PLL master, connect this output to the appropriate PLL reset input of the GTP wrapper. See GTP PLL Usage Models for SDI Applications, page 13 for more details.
gtp_rxplllock	In	1	Connect this port to the PLL lock signal of the PLL supplying the clock to the GTP RX. See GTP PLL Usage Models for SDI Applications, page 13 for more details.
gtp_rxresetdone	In	1	Connect this port to the rxresetdone port of the GTP transceiver.
gtp_gtrxreset	Out	1	Connect this port to the gtrxreset port of the GTP transceiver.
gtp_rxuserrdy	Out	1	Connect this port to the rxuserrdy port of the GTP transceiver.
gtp_rxrate	Out	3	Connect this port to the rxrate port of the GTP transceiver.
gtp_rxratedone	In	1	Connect this port to the rxratedone port of the GTP transceiver.
gtp_rxcdrhold	Out	1	Connect this port to the rxcdrhold port of the GTP transceiver.
gtp_drpclk	In	1	Connect this port to the clock that is driving the drpclk port of the GTP transceiver.
gtp_drprdy	In	1	Connect this port to the drprdy port of the GTP transceiver.
gtp_drpbusy	In	1	Connect this port to the drp_busy port of the GTP transceiver.
gtp_drpaddr	Out	10	Connect this port to the drpaddr port of the GTP transceiver.
gtp_drpdi	Out	16	Connect this port to the drpdi port of the GTP transceiver.
gtp_drpen	Out	1	Connect this port to the drpen port of the GTP transceiver.
gtp_drpwe	Out	1	Connect this port to the drpwe port of the GTP transceiver.
Ports That Connect to the GTP TX			
gtp_txdata	Out	20	Connect this port to the txdata port of the GTP transceiver.

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

Port Name	I/O	Width	Description
gtp_txpllreset	In	1	This port is used to reset the PLL supplying the serial clock to the GTP TX. If this SDI wrapper is acting as the PLL master, connect this output to the appropriate PLL reset input of the GTP wrapper. See GTP PLL Usage Models for SDI Applications, page 13 for more details.
gtp_txplllock	In	1	Connect this port to the PLL lock signal of the PLL supplying the clock to the GTP TX. See GTP PLL Usage Models for SDI Applications, page 13 for more details.
gtp_gttxreset	Out	1	Connect this port to the gttxreset port of the GTP transceiver.
gtp_txresetdone	In	1	Connect this port to the txresetdone port of the GTP transceiver.
gtp_txratedone	In	1	Connect this port to the txratedone port of the GTP transceiver.
gtp_txuserrdy	Out	1	Connect this port to the txuserrdy port of the GTP transceiver.
gtp_txrate	Out	3	Connect this port to the txrate port of the GTP transceiver.
gtp_txsysclksel	Out	2	If the clock source of the GTP TX needs to be dynamically switched between the two PLLs, connect this port to the txsysclksel port of the GTP transceiver.

Table 2 lists the parameters that can be applied to the Verilog version of the SDI wrapper.

Table 2: **SDI Wrapper Verilog Parameter List**

Name	Type	Default	Description
FXDCLK_FREQ	Integer	27000000	This parameter specifies the frequency, in Hz, of the fixed-frequency clock on the clk port of the GTP 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 function correctly.
DRPCLK_PERIOD	Integer	37	This parameter specifies the period, in ns, of the clock that is driving the GTP drpclk port and the SDI wrapper gtp_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 GTP initialization sequences based on the period of this clock.
PLLLOCK_TIMEOUT_PERIOD	Integer	2000000	This parameter specifies the duration of the PLL lock timeout period in ns. If, after being reset, the PLL does not assert its plllock signal within this period of time, the control module will timeout and retry the PLL reset sequence. The default value is equivalent to 2 ms.
RESET_TIMEOUT_PERIOD	Integer	500000	This parameter specifies the duration of the GTP transceiver reset timeout period in ns. If, after being reset, the GTP transceiver does not assert its rxresetdone or txresetdone within this period of time, the control module will timeout and retry the GTP transceiver reset sequence. The default value is equivalent to 500 µs.
TIMEOUT_CNTR_BITWIDTH	Integer	16	This parameter specifies the width in bits of the timeout counter used to generate both PLL lock and reset timeouts. 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 PLLOCK_TIMEOUT_PERIOD and RESET_TIMEOUT_PERIOD.

Table 2: **SDI Wrapper Verilog Parameter List *(Cont'd)***

Name	Type	Default	Description
RETRY_CNTR_BITWIDTH	Integer	8	This parameter specifies the width in bits of the retry counter. The retry counter counts the number of retry cycles used to attempt to complete a GTP RX or TX initialization or reset sequence or a dynamic change of the GTP transceiver's rxrate, txrate, or txsysclksel ports. If the retry counter reaches its maximum value of all ones, the sequence is considered to have failed. Thus, RETRY_CNTR_BITWIDTH specifies the number of retries that are permitted before the control module abandons the sequence. The default value of 8 allows for 255 retry cycles.
TXSYSCLKSEL_M_0	2-bit value	`2'b11`	Specifies the value output on the gtp_txsysclksel port when the tx_m is Low.
TXSYSCLKSEL_M_1	2-bit value	`2'b00`	Specifies the value output on the gtp_txsysclksel port when the tx_m is High.

Table 3 lists the generics that can be applied to the VHDL version of the SDI wrapper.

Table 3: **SDI Wrapper VHDL Generic List**

Name	Type	Default	Description
FXDCLK_FREQ	Integer	27000000	This generic specifies the frequency, in Hz, of the fixed-frequency clock on the clk port of the GTP 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 function correctly.
DRPCLK_PERIOD	Integer	37	This generic specifies the period, in ns, of the clock that is driving the GTP drpclk port and the SDI wrapper gtp_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 GTP initialization sequences based on the period of this clock.
PLLLOCK_TIMEOUT_PERIOD	Integer	2000000	This generic specifies the duration of the PLL lock timeout period in ns. If, after being reset, the PLL does not assert its plllock signal within this period of time, the control module will timeout and retry the PLL reset sequence. The default value is equivalent to 2 ms.
RESET_TIMEOUT_PERIOD	Integer	500000	This generic specifies the duration of the GTP transceiver reset timeout period in ns. If, after being reset, the GTP transceiver does not assert its rxresetdone or txresetdone within this period of time, the control module will timeout and retry the GTP transceiver reset sequence. The default value is equivalent to 500 µs.
TIMEOUT_CNTR_BITWIDTH	Integer	16	This generic specifies the width in bits of the timeout counter used to generate both PLL lock and reset timeouts. The width of this counter must be sufficient to count up to the maximum timeout periods specified by PLLOCK_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.

Table 3: **SDI Wrapper VHDL Generic List *(Cont'd)***

Name	Type	Default	Description
RETRY_CNTR_BITWIDTH	Integer	8	This generic specifies the width in bits of the retry counter. The retry counter counts the number of retry cycles used to attempt to complete a GTP RX or TX initialization or reset sequence or a dynamic change of the GTP transceiver's rxrate, 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 abandons the sequence. The default value of 8 allows for 255 retry cycles.
TXSYSCLKSEL_M_0	std_logic_vector (1 downto 0)	"11"	Specifies the value output on the gtp_txsysclksel port when the tx_m is Low.
TXSYSCLKSEL_M_1	std_logic_vector (1 downto 0)	"00"	Specifies the value output on the gtp_txsysclksel port when the tx_m is High.

Video Transport Detector Ports

The RX section of the SDI core has an 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 [Ref 7] 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 the video format family of the transport in the SDI signal. The encoding of this output port is shown in Table 4. 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 4: **rx_t_family Encoding**

rx_t_family	Transport Video Format	Active Pixels
0000	SMPTE ST 274 [Ref 8]	1920 x 1080
0001	SMPTE ST 296 [Ref 9]	1280 x 720
0010	SMPTE 2048-2 [Ref 10]	2048 x 1080
0011	SMPTE 295 [Ref 11]	1920 x 1080
1000	NTSC	720 x 486
1001	PAL	720 x 576
1111	Unknown	
Others	Reserved	

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 5. The encoding of the frame rate matches the encoding used in the picture rate field of SMPTE ST 352 [Ref 7] video payload ID packets. However, the rx_t_rate shows the transport frame rate, not the picture rate. Also, the rx_t_rate port value is always the frame rate, even for interlaced transports.

Table 5: **rx_t_rate Encoding**

rx_t_rate	Frame Rate (Hz)
0000	None
0010	23.98
0011	24
0100	47.95
0101	25
0110	29.97
0111	30
1000	48
1001	50
1010	59.94
1011	60
Others	Reserved

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

The SDI receiver can, optionally, 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 only 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 asserting rx_edh_clr_errcnt High. The user 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 6 shows the encoding of the bits on the rx_edh_errcnt_en port.

Table 6: **rx_edh_errcnt_en Bits**

Bit #	Error
0	ANC EDH error
1	ANC EDA error
2	ANC IDH error
3	ANC IDA error
4	ANC UES error
5	FF EDH error
6	FF EDA error

Table 6: **rx_edh_errcnt_en Bits**

Bit #	Error
7	FF IDH error
8	FF IDA error
9	FF UES error
10	AP EDH error
11	AP EDA error
12	AP IDH error
13	AP IDA error
14	AP UES error
15	EDH packet checksum error

The ANC error conditions are associated with errors in the ancillary data packets. The FF error conditions are associated with errors detected by the full field CRC. The AP error conditions are associated with errors detected by the active picture CRC. 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 [Ref 12] document.

- 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.

In addition to 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 7.

Table 7: **Encoding of rx_edh_anc_flags, rx_edh_ap_flags, and rx_edh_ff_flags Ports**

Bit #	Flag
0	EDH
1	EDA
2	IDH
3	IDA
4	UES

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 8.

Table 8: **Encoding of rx_edh_packet_flags Port**

Bit #	Error
0	EDH packet is missing
1	Parity error in user data words of EDH packet
2	Checksum error in EDH packet
3	Format error in EDH packet, such as invalid data count

GTP Initialization and Reset and Change Sequence Failure Codes

If a failure occurs during a GTP RX initialization or reset sequence or during a dynamic change of the GTP transceiver's rxrate port, the rx_change_fail port is asserted High and a failure code is output on the rx_change_fail_code port. A sequence only ends in failure after 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 9.

Table 9: **rx_change_fail_code Port Encoding**

Code	Description
0	This code indicates that the PLL failed to lock to its reference clock within the allowed time or the GTP transceiver failed to assert rxresetdone within the allowed period of time following a gtrxreset.
1	This code indicates that the DRP arbiter was not able to grant control of the DRP to the gtrxreset state machine in the GTP wrapper in order to do a gtrxreset sequence because the DRP was constantly busy. Such a failure should only occur if there is a problem with the a7gtp_sdi_drp_control module preventing it from giving up the DRP.
2	This code indicates that a sequence involving a change to rxrate failed because the GTP transceiver didn't assert the rxratedone signal within the allowed time period, including retries.
3	When an RX change sequence is initiated, the a7gtp_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.
4	When the RX SDI mode changes from HD-SDI to SD-SDI mode, rxrate is changed and rxcdrhold is asserted. The GTP CDR must be reset, and this reset is initiated by a state machine in the GTP wrapper in response to a dynamic change on the rxrate port. The a7gtp_sdi_drp_control module watches to make sure the GTP wrapper initiates the required reset. If the required reset is not initiated by the GTP wrapper within the allowed amount of time, including retries, this failure code is asserted.
5	When a change of the RX SDI mode occurs that requires changing the RXCDR_CFG attribute in the GTP transceiver, the a7gtp_sdi_drp_control module attempts to do a series of DRP write cycles to change that attribute. If any of these write cycles is not acknowledged by the GTP transceiver by asserting the drprdy port within the given amount of time, the entire sequence is aborted and retried up to the maximum allowed number of retries. If the RXCDR_CFG attribute cannot be modified correctly after the maximum number of retires, this failure code is asserted.
6	When the RX SDI mode changes to 3G-SDI or HD-SDI modes, but not SD-SDI mode, and an rxrate change is required, the GTP CDR must be reset. This reset is initiated by a state machine in the GTP wrapper in response to a dynamic on the rxrate port. The a7gtp_sdi_drp_control module watches to make sure the GTP wrapper initiates the required reset. If the required reset is not initiated by the GTP wrapper within the allowed amount of time, including retries, this failure code is asserted. This failure code is asserted for the same reason as failure code 4. The difference is that failure code 4 is asserted only when a failure occurs while transitioning into SD-SDI mode, and failure code 6 is asserted only when a failure occurs while transitioning into a mode other than SD-SDI mode.
7	When the RX SDI mode changes to a mode other than 3G-SDI but rxrate does not need to change (e.g., a change from SD-SDI mode to 3G-SDI mode), the a7gtp_sdi_drp_control module requests a gtrxreset to reset the CDR. If the GTP wrapper does not respond to this gtrxreset request within the allowed amount of time, including retries, this failure code is asserted.

If a failure occurs during a GTP TX initialization or reset sequence or during a dynamic change of the GTP transceiver's txrate or txsysclksel ports, the tx_change_fail port is asserted High and a failure code is output on the rx_change_fail port. As with the RX side, the sequence only fails after the maximum number of retries has been attempted. The encoding of the tx_change_fail_code port is shown in Table 10.

Table 10: **tx_change_fail_code Port Encoding**

Value	Description
0	This failure code is reserved.
1	During a full reset sequence or the GTP initialization sequence, this failure code indicates that the PLL providing the serial clock to the GTP TX failed to assert its plllock signal within the given amount of time, including retries, after being reset.
2	During the GTP initialization sequence, a GTP full reset sequence, or a gttxreset sequence requested by the application, this failure code indicates that the GTP 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 GTP transceiver to respond to the assertion of gttxreset.
3	During the GTP initialization sequence, a GTP full reset sequence, or a gttxreset sequence requested by the application, this failure code indicates that the GTP transceiver failed to assert its txresetdone signal within the given amount of time, including retries, after a gttxreset.
4	This failure code indicates that the GTP transceiver failed to indicate successful completion of a txrate change by asserting its txratedone output within the allowed amount of time, including retries.
5	When the application requests a dynamic change of txsysclksel by changing the tx_m input of the SDI wrapper, gttxreset is asserted prior to the change of txsysclksel. If the GTP transceiver fails to negate its txresetdone output in response to the assertion of gttxreset within the allowed amount of time, including retries, the txsysclksel change sequence fails with this failure code.
6	During a dynamic change of txsysclksel, gttxreset is asserted. At the end of the sequence, gttxreset is negated. If the GTP transceiver fails to assert the txresetdone output within the allowed amount of time, including retries, after gttxreset is negated, the txsysclksel change sequence fails with this failure code.
7	This failure code is reserved.

SDI Timing Constraints

For the SDI wrapper and 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 gtp_drpclk ports of the SDI wrapper. See the constraints files of the example SDI applications provided with this application note for examples of setting these constraints.

Example SDI Demonstrations

Two example SDI demonstration applications are include with this application note. The source code for these demonstrations is provided in Verilog only. Instructions for building these demonstrations using either the ISE or Vivado tools are included in the `readme.txt` file located in the `xapp1097.zip` file along with the source code. Pre-generated FPGA configuration files are also provided for both demonstrations that can be loaded onto an Artix-7 FPGA AC701 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 AC701 board. The inrevium FMC also provides SDI-specific clock sources that are used as reference clocks for the GTP transceivers.

Dual SDI Demonstration

This demonstration application includes two SDI RX interfaces and two SDI TX interfaces that are all independent. The demonstration is limited to two SDI RX and two SDI TX interfaces because the AC701 board only connects two GTP transceivers to the FMC connector. This is not a limitation of the Artix-7 FPGA or the SDI core. It is solely a limitation of this particular board.

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 virtual I/O (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. The video data received by each SDI RX can be captured and viewed using an integrated logic analyzer (ILA) window in the ChipScope Pro analyzer.

It is possible to use the Vivado logic analyzer instead of ChipScope Pro analyzer. However, the ChipScope Pro analyzer provides a better user interface for these SDI demonstrations. It is recommended that ChipScope Pro analyzer be used, and the only instructions provided in this application note are 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 GTP transceivers, and the connectors labeled CH1-RX and CH1-TX are the SDI RX and TX connectors for the second GTP transceiver.

Figure 23 is a block diagram of the demonstration, showing one SDI channel that is connected to the first GTP transceiver. Both SDI channels are identical in this demonstration with the exception that the first GTP transceiver is the PLL master, responsible for resetting the GTP PLLs.

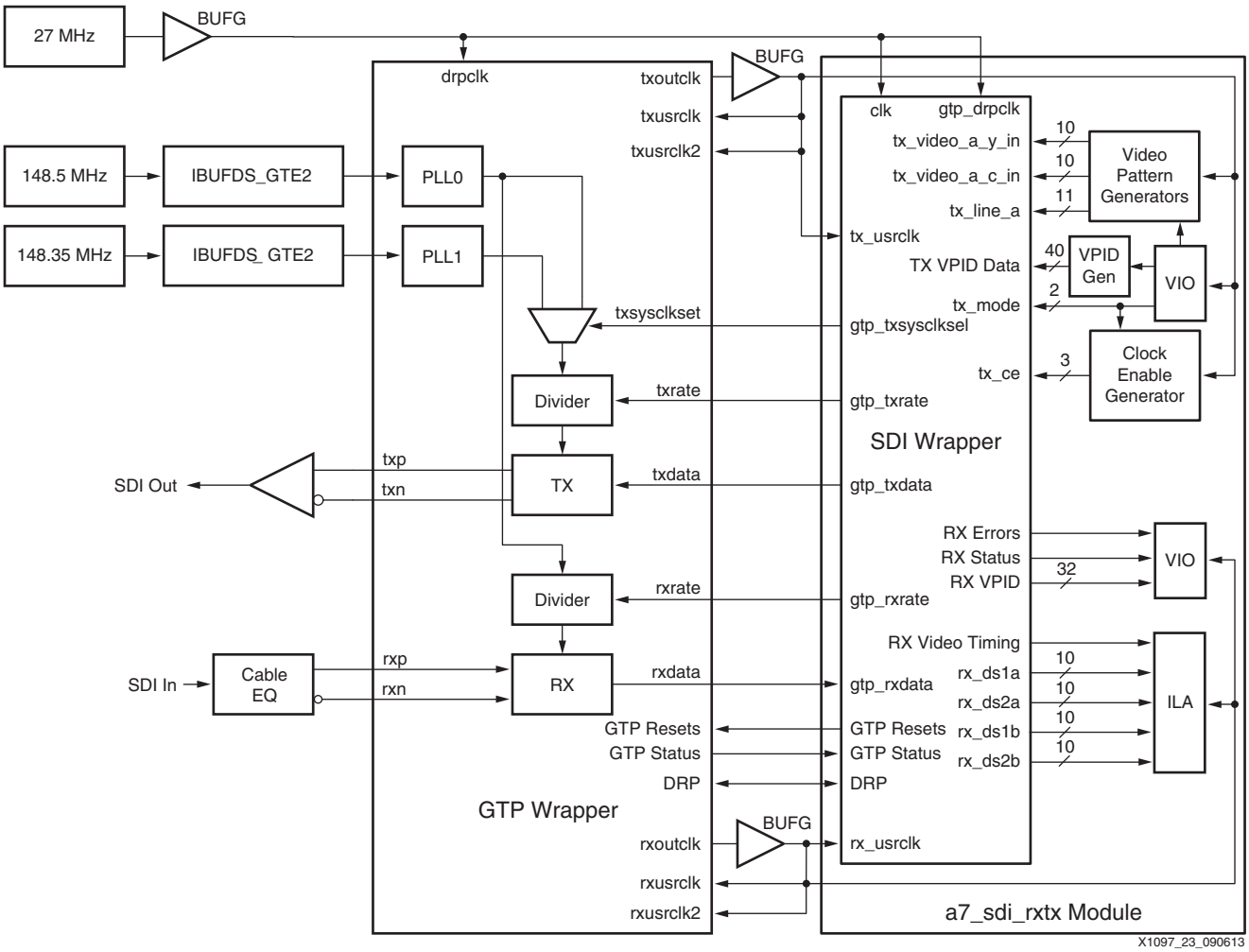


Figure 23: **Dual SDI Block Diagram**

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 PLL0 and PLL1, respectively, in the GTP Quad. The GTP transmitters are dynamically switched between the serial clocks from the two PLLs to support all SDI bit rates.

The LMH1983 device on the inrevium board supplies a 27 MHz clock to the Artix-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 twice in this demonstration, the SDI wrapper, video pattern generators, TX clock enable generator, ChipScope VIO and ILA modules, and other miscellaneous logic are contained in a module called a7_sdi_rxtx. This module is instanced twice in the top-level module of the design.

The following are required to run the Dual SDI demonstration:

- Xilinx Artix-7 FPGA AC701 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 AC701 board as shown in Figure 24.

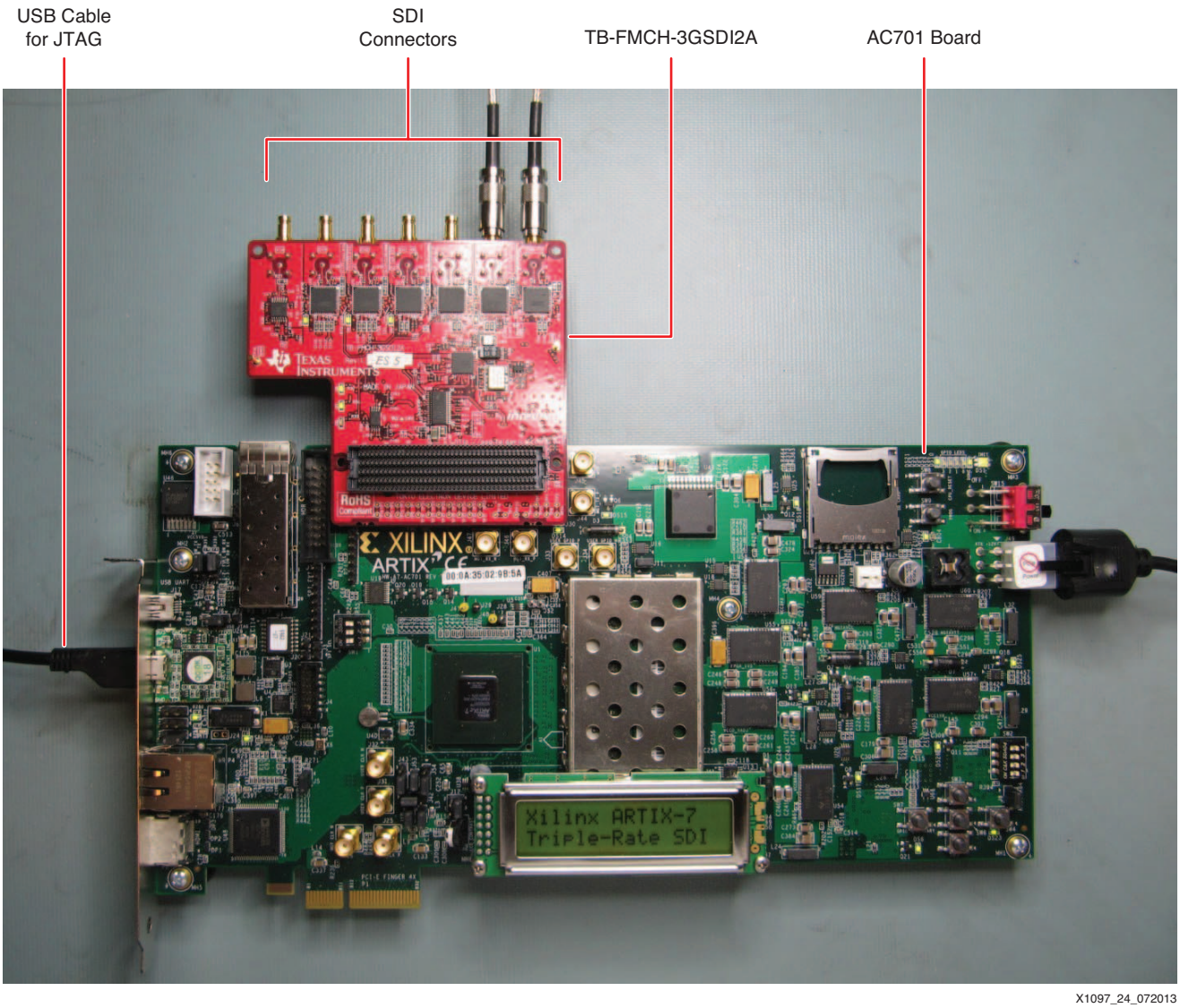


Figure 24: **AC701 Board with TB-FMCH-3GSDI2A Board Connected**

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 AC701 board must be connected to a PC with ChipScope Pro analyzer installed by the USB JTAG cable provided with the AC701 board.

The file called `ac701_sdi_demo.bit` provided with this application note must be loaded into the Artix-7 FPGA on the AC701 board using ChipScope Pro analyzer. After the BIT file is loaded into the FPGA, the `ac701_sdi_demo.cpj` ChipScope project file must be opened in ChipScope Pro analyzer. When the project is opened, it looks like Figure 25. There are five VIO windows, one for each RX and TX and one showing the locked status of the two GTP PLLs. There are also two ILA waveform windows, one for each receiver in the demonstration, that are minimized in Figure 25.

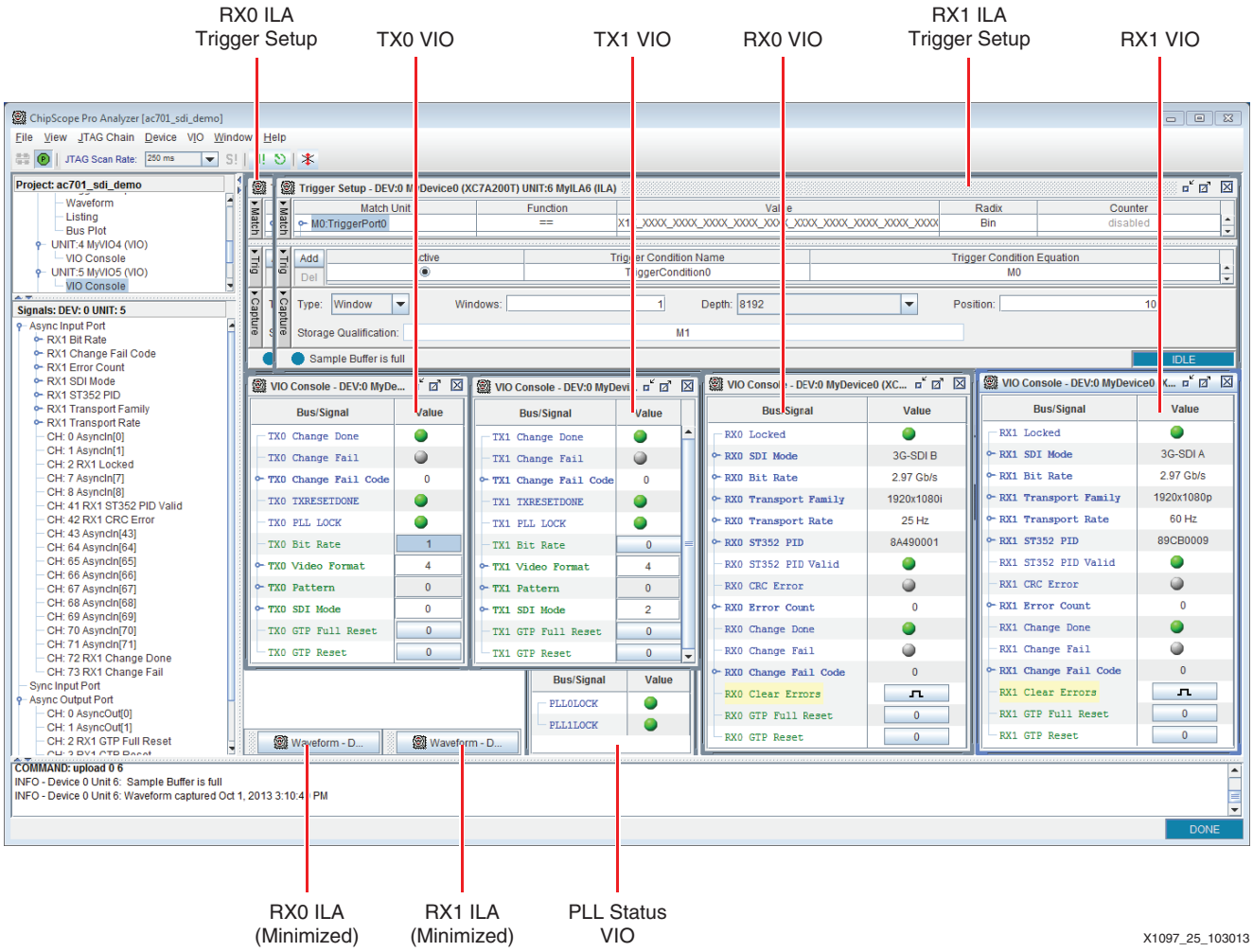


Figure 25: **ChipScope Pro Analyzer with Dual SDI Project Opened**

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. 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 TX1 is shown in Figure 26. Each transmitter is controlled by its VIO window. The VIO window for TX1 is shown in Figure 26.

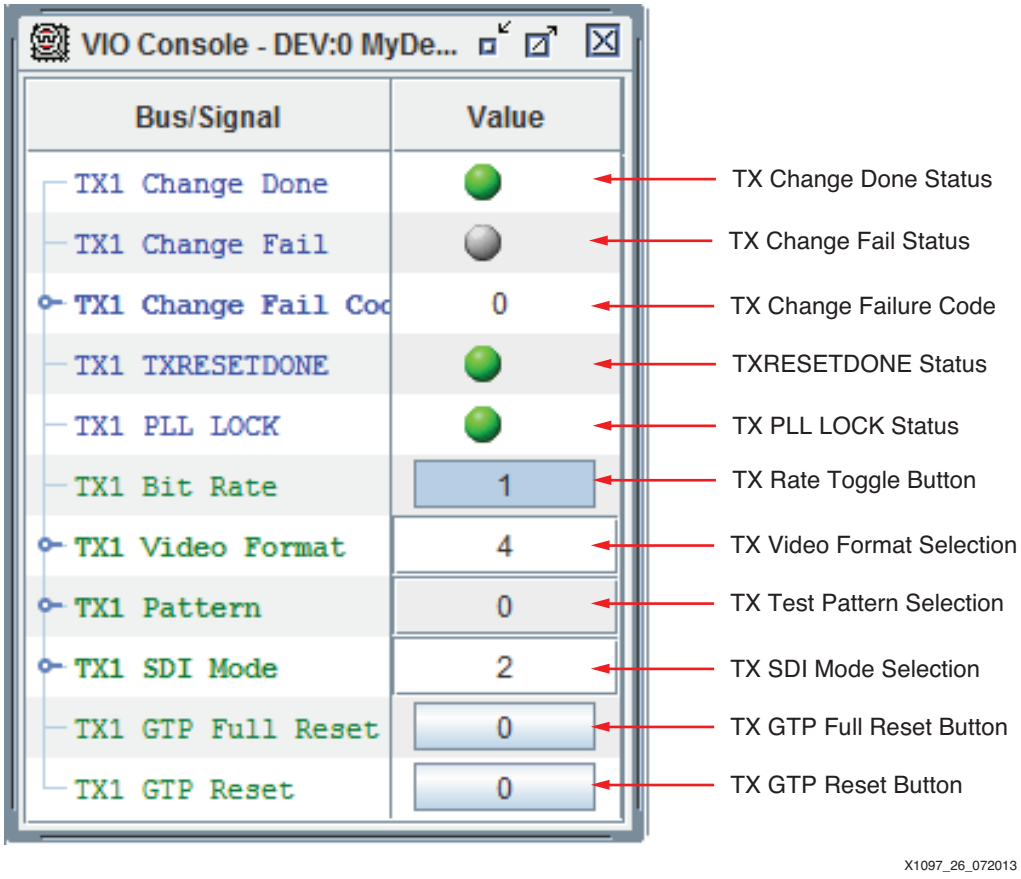


Figure 26: **Dual SDI Demonstration TX VIO Control Window**

The first three items at the top of the TX VIO window indicate the status of the last GTP 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 10.

The TXRESETDONE and PLL LOCK indicators indicate the status of these two signals from the GTP transceiver. During normal operation, both of these indicators are green.

The TX Bit Rate toggle button, TX Video Format selection field, and the TX SDI Mode selection field work together to set the format of the SDI signal generated by the SDI transmitter as shown in Table 11.

Table 11: **Quad SDI Demonstration TX Video Format Selection**

TX Video Format	HD-SDI (SDI Mode = 0)		3G-SDI (SDI Mode = 2)		SD-SDI (SDI Mode = 1)
	TX Bit Rate = 0	TX Bit Rate = 1	TX Bit Rate = 0	TX Bit Rate = 1	
0	720p 50 Hz	Not Valid	Not Valid	Not Valid	NTSC
1	1080pSF 24 Hz	1080pSF 23.98 Hz	Not Valid	Not Valid	PAL
2	1080i 60 Hz	1080i 59.94 Hz	Not Valid	Not Valid	NTSC
3	1080i 50 Hz	Not Valid	Not Valid	Not Valid	PAL
4	1080p 30 Hz	1080p 29.97 Hz	1080p 60 Hz	1080p 59.97 Hz	NTSC
5	1080p 25 Hz	Not Valid	1080p 50 Hz	Not Valid	PAL
6	1080p 24 Hz	1080p 23.98 Hz	Not Valid	Not Valid	NTSC
7	720p 60 Hz	720p 59.94 Hz	Not Valid	Not Valid	PAL

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 modes, 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

At the bottom of the TX VIO window are two buttons that reset the GTP TX. The TX GTP Full Reset button resets both the PLL (TX0 only) and the GTP TX unit. Because TX1 is not the PLL master, the full reset button in the TX1 VIO window does not reset the PLL. The TX GTP Reset button resets just the GTP TX unit and not the PLL.

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 27 shows the VIO window for one of the receivers.

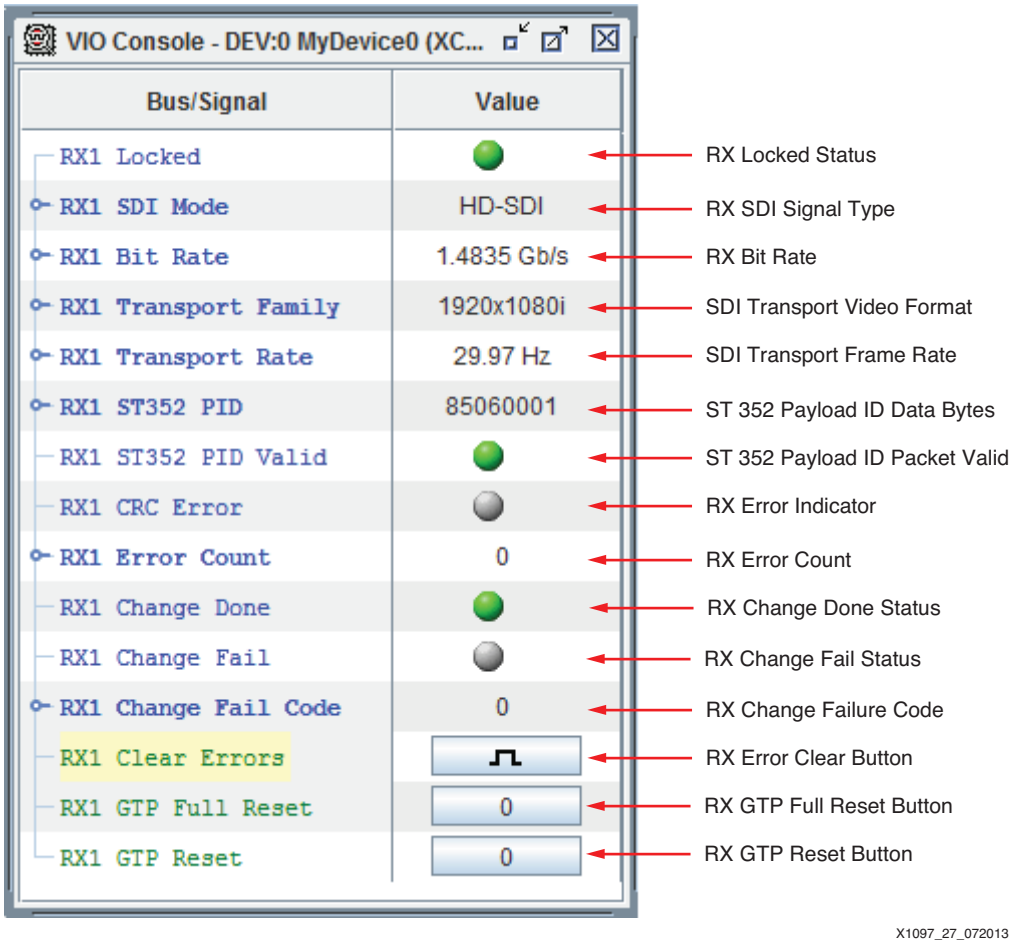


Figure 27: **Dual SDI Demonstration RX Status Window**

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 (frame rate).

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 3 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 grey if no errors have been detected. After an error has been detected, this indicator stays 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.

At the bottom of the RX VIO window are two buttons that reset the GTP RX. The RX GTP Full Reset button resets both the PLL (RX0 only) and the GTP RX unit. Because RX1 is not the PLL master, the full reset button in the RX1 VIO window does not reset the PLL. The RX GTP Reset button resets just the GTP RX unit and not the PLL.

Figure 28 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 on the Trigger Setup and Waveform items under the appropriate UNIT in the Project panel in the upper-left as shown in Figure 28. UNIT 3 is the ILA for RX0 and UNIT 6 is the ILA for RX1.

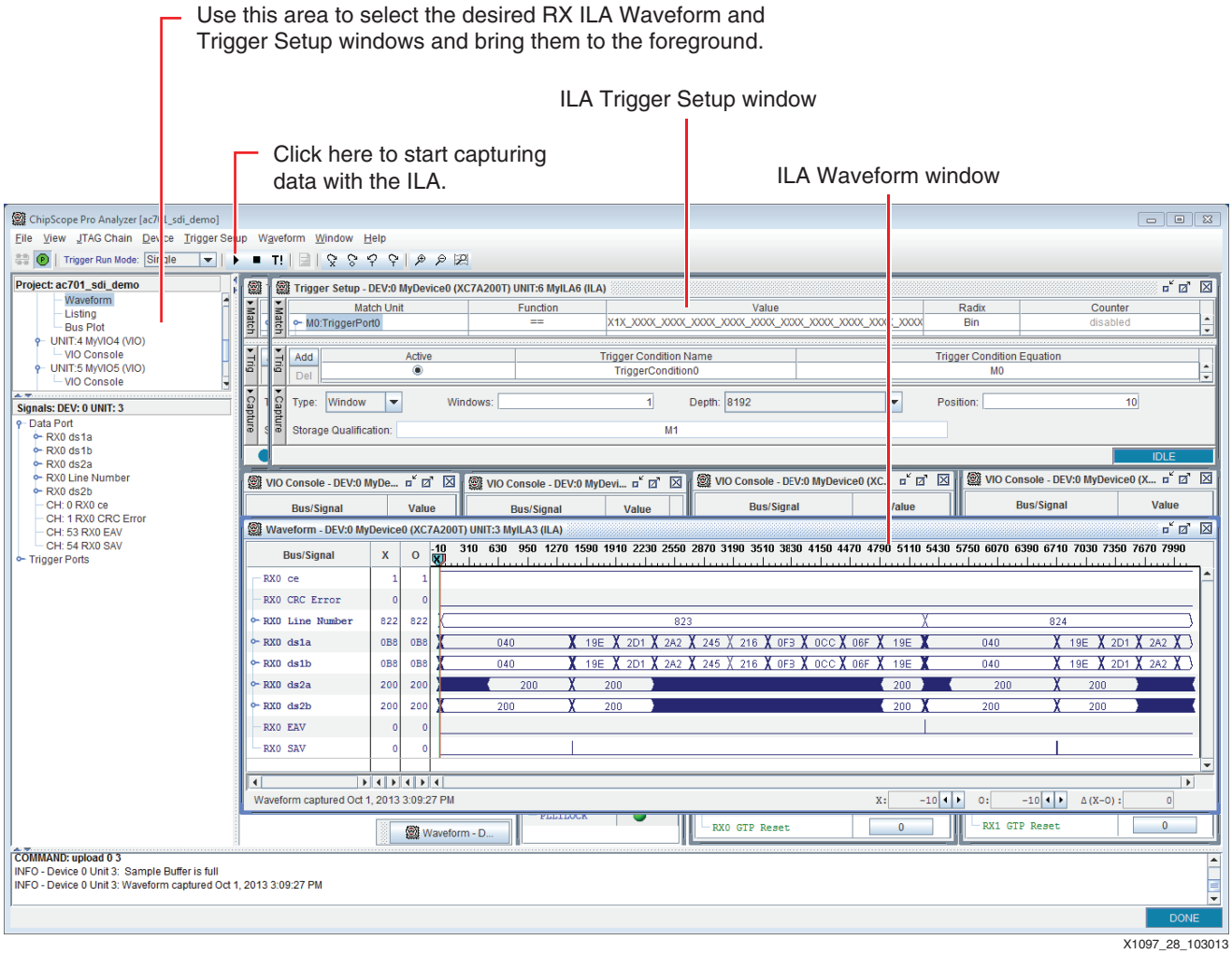


Figure 28: **Using the ChipScope ILA to View RX Data in the Dual SDI Demonstration**

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 `ac701_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 28 to initiate a capture by the ILA. The capture buffer is large enough to capture multiple lines of video.

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 29 is a block diagram of this demonstration.

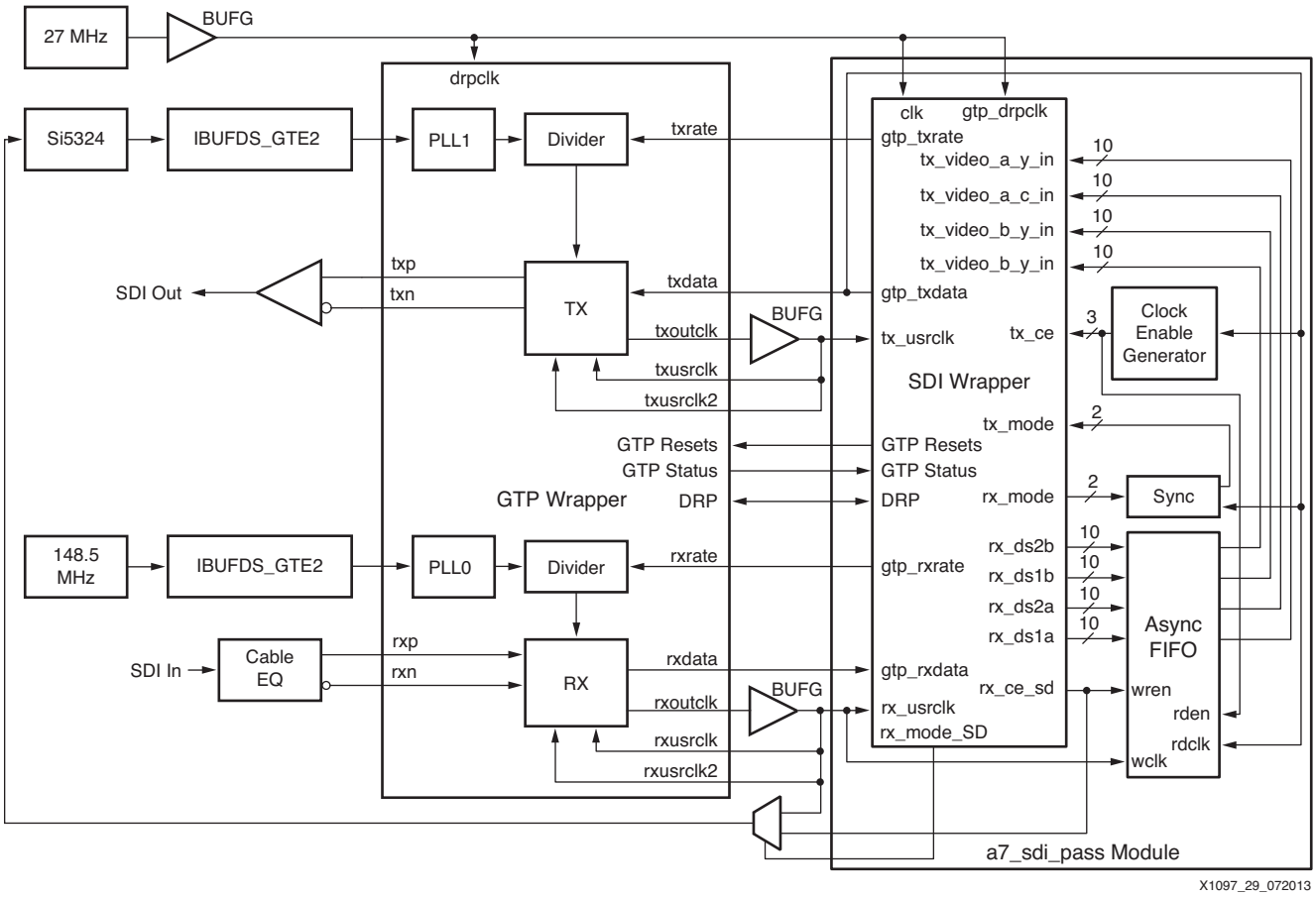


Figure 29: **SDI Pass-Through Demonstration**

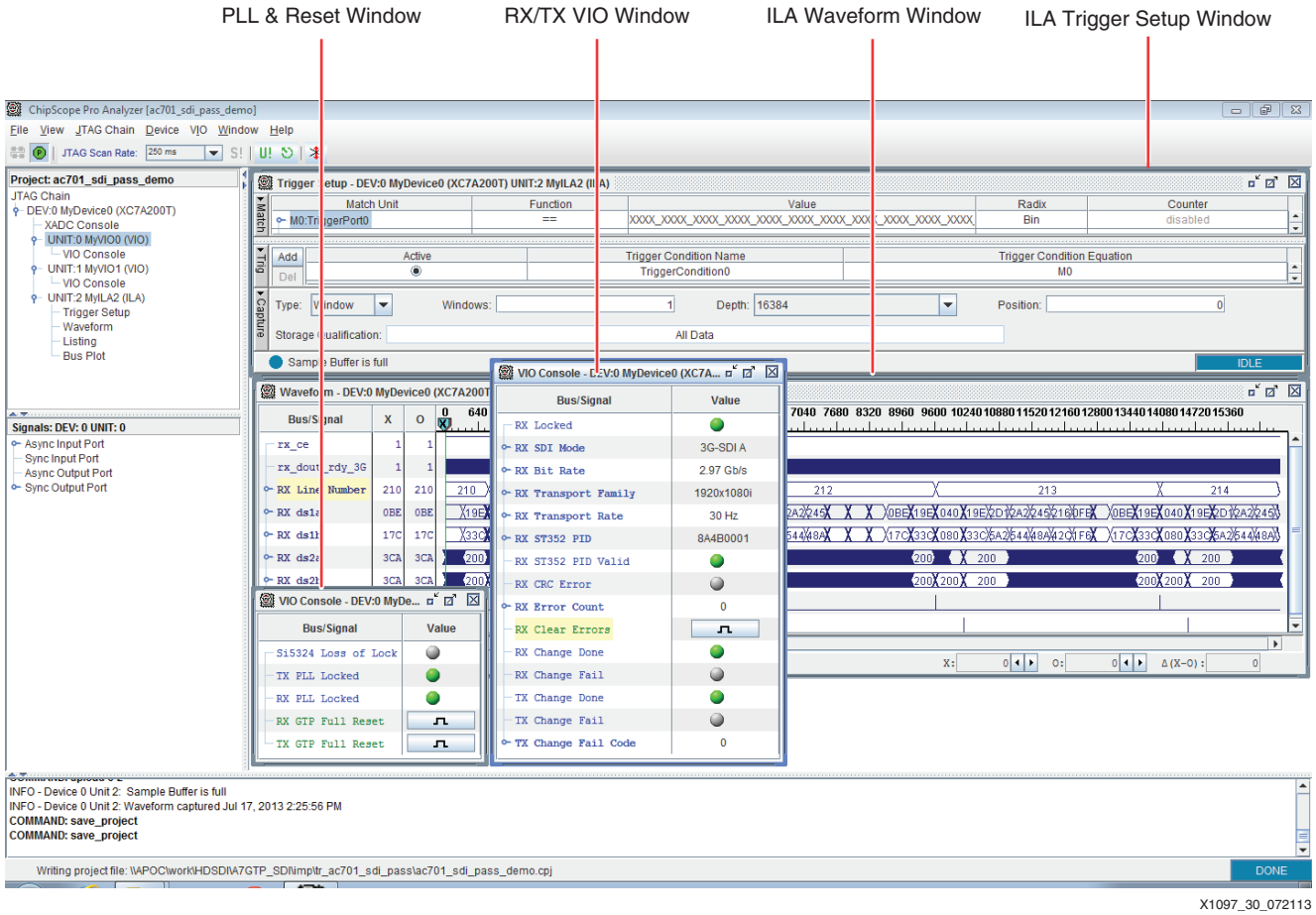
PLL0 is locked to a 148.5 MHz reference clock and provides the serial clock to the GTP RX unit. The data from the GTP RX goes through the SDI RX datapath and then into an asynchronous FIFO. The FIFO moves the data from the RX clock domain (rx_usrclk) to the TX clock domain (tx_usrclk). In HD-SDI and 3G-SDI modes, the recovered clock from the GTP RX (rxoutclk) is sent to an Si5324 digital PLL for jitter reduction and then used as the reference clock to PLL1. 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. PLL1 is locked to the clock from the Si5324 and provides the serial clock to the GTP 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 GTP TX for serialization.

The following are required to run the SDI pass-through demonstration:

- Xilinx Artix-7 FPGA AC701 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 AC701 board's JTAG USB connector

The inrevium SDI FMC must be connected to the FMC connector on the AC701 as shown in Figure 24. 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 `ac701_sdi_pass_demo.bit` provided with this application note must be loaded into the Artix-7 FPGA on the AC701 board. After the BIT file is loaded into the FPGA, the ac`701_sdi_pass_demo.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 30.



X1097_30_072113

Figure 30: **Pass-Through Demonstration ChipScope Analyzer Window**

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

One VIO window shows the status of GTP PLLs and the Si5324 digital PLL. During normal operation, the TX PLL Locked and RX PLL Locked indicators are green and the Si5324 Loss of

Lock indicator is grey. When there is no valid SDI input signal at the input of the SDI RX or during a short period of time after the input SDI signal changes bit rates, the Si5324 loses lock with the recovered clock from the GTP RX as indicated by the Si5324 Loss of Lock indicator turning red. When the Si5324 is not locked, the TX PLL is also not locked and the TX PLL Locked indicator turns grey. 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 GTP reset buttons for the RX and the TX. These buttons generate a full reset of the GTP RX or TX including resetting the associated PLL.

The other VIO window shows the status of the SDI RX and TX. The RX status indicators and the RX Clear Errors button in this VIO are identical in function to those in the RX VIO window of the dual SDI demonstration, as shown in Figure 27. 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 dual SDI demonstration, as shown in Figure 26.

A single ILA is used to capture and observe the data from the SDI RX. It functions just like the SDI RX ILA in the dual SDI demonstration.

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

FPGA Resource Usage

Table 12 shows the FPGA resources required in an SDI interface with an Artix-7 FPGA GTP 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 the Vivado tools 2013.3.

Table 12: **Artix-7 FPGA GTP SDI Interface FPGA Resource Usage**

Reference Design	LUTs	FFs
SDI RX with EDH processor & TX	3337	2855
SDI RX without EDH processor & TX	2818	2456
SDI RX with EDH processor	2080	1861
SDI RX without EDH processor	1581	1462
SDI TX	1247	994

The SDI receiver and transmitter interface designs do not use any MMCM clock managers. They also 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. In addition, one fixed-frequency global clock is required for timing purposes in the SDI wrapper. This fixed-frequency clock is usually also used as the GTP DRP clock. Only one such fixed-frequency global clock is required no matter how many SDI interfaces are implemented in the FPGA.

Constraints

Because of the use of 20-bit RX and TX datapaths to and from the GTP transceiver, the maximum clock frequency used in these designs is 148.5 MHz. It is very difficult to meet timing in -1 speed grade Artix-7 FPGAs, so Xilinx does not support the use of -1 speed grade devices with the SDI core. For SDI applications, -2 or faster speed grade devices are required.

As previously described, from the end of FPGA configuration until the GTP transceiver is fully initialized, the RXOUTCLK from the GTP transceiver will have a frequency of 297 MHz. The SDI wrapper keeps the RX section of the SDI core in reset until the RXOUTCLK frequency drops to 148.5 MHz or slower.

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 GTP transceiver and the fixed-frequency clock that is used for the DRPCLK and the SDI wrapper's clk port. The rxoutclk and txoutclk constraints should specify the period of these clocks as 148.5 MHz. For placement, all that is required is to constrain the GTP transceivers to their desired locations by constraining the rxp/rxn and txp/txn pins or using the XY coordinate system to constrain the actual location of the GTP transceiver itself. All GTP transceivers that are instanced in the same GTP wrapper must be constrained to be in the same GTP Quad tile.

Glossary

Table 13 lists a glossary of terms used in this application note.

Table 13: **Glossary**

Term	Definition
3G-SDI	Common name for SMPTE ST 424, the 3 Gb/s serial digital interface [Ref 13]. 3G-SDI supports three mapping modes defined in ST 425-1 called 3G-SDI level A, level B-DL, and level B-DS. Refer to *Source Image Format and Ancillary Data Mapping for the 3 Gb/s Serial Interface* (ST 425-1) [Ref 14] for details about these mapping modes.
Ancillary (ANC) data	Non-video data embedded in portions of the SDI data stream not used for active picture data. One very common type of ANC data is embedded audio. ANC data must be formatted into ancillary data packets, as specified by SMPTE *Television – Ancillary Data Packet and Space Formatting* (ST 291-1) [Ref 15].
Data stream	The actual data into and out of the SDI interface. The data stream must be formatted according to the transport data structure as it enters and exits the SDI interface.
EDH	The error detection and handling protocol for SD-SDI as defined by SMPTE *Error Detection Checkwords and Status Flags for Use in Bit-Serial Digital Interfaces for Television* (RP 165) [Ref 12].
Embedded audio	Generally refers to digital audio that is carried as ancillary data in an SDI signal.
End of active video (EAV)	In SDI compatible data streams, the EAV is a sequence of four words, unique in the data steam, marking the end of the active portion of the line and start of the horizontal blanking interval. Each video line is considered to begin with the first word of the EAV.
HD-SDI	Common name for the SMPTE *1.5 Gb/s Signal/Data Serial Interface* (ST 292-1) [Ref 16].
Interlaced	A video scanning system in which the video frame is divided into two sequential fields. Field one consists of the odd lines and field two consist of the even lines that are displayed between the odd lines of field one. The two fields represent different pictures displaced in time by one half of the frame time.

Table 13: **Glossary *(Cont'd)***

Term	Definition
Link	If the picture's bandwidth exceeds the capacity of the serial digital interface, two or more serial digital interfaces can be ganged together to increase the bandwidth to transport the picture. Each separate serial digital interface of a multi-link set is called a link. SMPTE *Dual Link 1.5 Gb/s Digital Interface for 1920 x 080 and 2048 x 1080 Picture Formats* (ST 372) [Ref 17] defines how to transport some higher bandwidth video formats on two HD-SDI links. Multi-link 3G-SDI standards in the ST 425-x family are currently under development by SMPTE [Ref 14]. The 3G-SDI level B-DL transport carries both links of a dual-link HD-SDI (ST 372) pair on one 3G-SDI interface. Each of the two HD-SDI signals carried by 3G-SDI level B-DL is still called a link.
Payload ID	Sometimes called the Video Payload ID (VPID), the payload ID is an ancillary data packet defined by SMPTE *Payload Identifier Codes for Serial Digital Interfaces* (ST 352) [Ref 7]. The four data words of the ST 352 payload ID packet identify both the nature of the video picture (video format, frame rate, scanning structure, and color space) and the type of SDI interface used to transport that payload. In multi-link interfaces, the payload ID also contains bits that distinguish between the individual links.
Progressive	A non-interlaced video scanning system. All lines of the progressive frame belong to the same picture.
SD-SDI	Common name for SMTPE *Television – SDTV Digital Signal/Data – Serial Digital Interface* (ST 259) [Ref 5], the standard-definition serial digital interface.
Serial Digital Interface (SDI)	Originally referred to as SMPTE *Television – SDTV Digital Signal/Data – Serial Digital Interface* (ST 259), the standard-definition serial digital interface. With the advent of HD-SDI and 3G-SDI, ST 259 is now often called SD-SDI to avoid confusion. This document uses the term *SDI* to generically refer to SD-SDI, HD-SDI and 3G-SDI. When referring specifically to ST 259, this document always uses the term *SD-SDI*.
SMPTE	Society of Motion Picture and Television Engineers.
Start of active video (SAV)	In SDI-compatible data streams, the SAV is a sequence of four words, unique in the data stream, marking the end of the horizontal blanking interval and the start of the active video portion of the line. The first active video sample of a line, usually called sample 0, occurs immediately after the SAV.
Synchronous switching (point, interval, line)	SMPTE *Definition of Vertical Switching Point for Synchronous Video Switching* (RP 168) [Ref 18] defines the point(s) in a video frame where it is permissible to switch between synchronous video sources. This is often called the synchronous switching point but is actually defined as an interval, a portion of a line, rather than an exact point on a line. The line that contains the synchronous switching interval is sometimes called the *synchronous switching line*.
Timing reference signal (TRS)	A generic term referring to both EAV and SAV sequences.

Table 13: **Glossary *(Cont'd)***

Term	Definition
Transport	The data structure of an interface data stream or streams. The transport data structure defines the EAV and SAV sequences used to carry video timing information.
XYZ	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.

Reference Design

The reference design files for this application note can be download from:

https://secure.xilinx.com/webreg/clickthrough.do?cid=344558

The reference design checklist is shown in Table 14.

Table 14: **Reference Design Checklist**

Parameter	Description
General	
Developer name	John Snow
Target devices	Artix-7 FPGAs with GTP transceivers, -2 speed grade or faster
Source code provided	Yes
Source code format	Verilog
Design uses code/IP from existing Xilinx application note/reference design, IP catalog, or third party	Yes, IP cores for IP catalog
Simulation	
Functional simulation performed	No
Timing simulation performed	No
Test bench used for functional and timing simulations	None
Test bench format	N/A
Simulator software/version used	N/A
SPICE/IBIS simulations	N/A
Implementation	
Synthesis software tools/version used	Vivado tools 2013.3 and XST in ISE Design Suite 14.7
Implementation software tools/version used	Vivado tools 2013.3 and ISE Design Suite 14.7
Static timing analysis performed	Yes
Hardware Verification	
Hardware verified	Yes
Hardware platform used for verification	AC701 and TB-FMCH-3GSDI2A boards

The `readme.txt` file describes the directory structure of the files provided with the reference design.

Conclusion

This application note describes how to use the SMPTE SD/HD/3G-SDI core and the Artix-7 FPGA GTP transceivers to implement SDI interfaces compatible with the SMPTE SD-SDI, HD-SDI, and 3G-SDI standards. The device-specific control logic necessary to use the GTP 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 an Artix-7 FPGA design.

References

This section lists the references used in this document.

1. *7 Series FPGAs GTP Transceivers User Guide* (UG482)
2. *Dynamically Programmable DRU for High-Speed Serial I/O* (XAPP875)
3. *Artix-7 FPGAs Data Sheet: DC and Switching Characteristics* (DS181)
4. *Society of Motion Picture and Television Engineers (SMPTE) SD/HD/3G-SDI Product Guide* (PG071)

The following references are available from the Society of Motion Picture and Television Engineers (www.smpte.org):

5. ST 259, *Television – SDTV Digital Signal/Data – Serial Digital Interface*
6. ST 344, *Television – 540 Mb/s Serial Digital Interface*
7. ST 352, *Payload Identification Codes for Serial Digital Interfaces*
8. ST 274: *Television – 1920 x 1080 Image Sample Structure, Digital Representation and Digital Timing Reference Sequences for Multiple Picture Rates*
9. ST 296: *1280 x 270 Progressive Image 4:2:2 and 4:4:4 Sample Structure — Analog and Digital Representations and Analog Interface*
10. ST 2048-2: *2048 x 1080 Digital Cinematography Production Image FS/709 Formatting for Serial Digital Interface*
11. ST 295: *Television – 1920 x 1080 50-Hz - Scanning and Interface*
12. RP 165, *Error Detection Checkwords and Status Flags for Use in Bit-Serial Digital Interfaces for Television*
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*
15. ST 291-1, *Television – Ancillary Data Packet and Space Formatting*
16. ST 292-1, *1.5 Gb/s Signal/Data Serial Interface*
17. ST 372, *Dual Link 1.5 Gb/s Digital Interface for 1920 x 1080 and 2048 x 1080 Picture Formats*
18. RP 168, *Definition of Vertical Interval Switching Point for Synchronous Video Switching*

Revision History

The following table shows the revision history for this document.

Date	Version	Description of Revisions
12/05/2013	1.0	Initial Xilinx release.
11/10/2015	1.0.1	Updated the `xapp1097.zip` file to address the RX reset sequence getting stuck when changing SDI standards.

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 Xilinx’s limited warranty, please refer to Xilinx’s Terms of Sale which can be viewed at www.xilinx.com/legal.htm#tos; 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 such critical applications, please refer to Xilinx’s Terms of Sale which can be viewed at www.xilinx.com/legal.htm#tos.

Automotive Applications Disclaimer

XILINX PRODUCTS ARE NOT DESIGNED OR INTENDED TO BE FAIL-SAFE, OR FOR USE IN ANY APPLICATION REQUIRING FAIL-SAFE PERFORMANCE, SUCH AS APPLICATIONS RELATED TO: (I) THE DEPLOYMENT OF AIRBAGS, (II) CONTROL OF A VEHICLE, UNLESS THERE IS A FAIL-SAFE OR REDUNDANCY FEATURE (WHICH DOES NOT INCLUDE USE OF SOFTWARE IN THE XILINX DEVICE TO IMPLEMENT THE REDUNDANCY) AND A WARNING SIGNAL UPON FAILURE TO THE OPERATOR, OR (III) USES THAT COULD LEAD TO DEATH OR PERSONAL INJURY. CUSTOMER ASSUMES THE SOLE RISK AND LIABILITY OF ANY USE OF XILINX PRODUCTS IN SUCH APPLICATIONS.