SERDES Framer Interface Level 5 for Virtex-6 Devices

Author: Vasu Devunuri

XAPP882 (v1.1) May 10, 2010

Summary

This application note describes the implementation of SERDES Framer Interface Level 5 (SFI-5) [Ref 1] in a Virtex-6 XC6VLX240T FPGA. SFI-5 is a standard defined by the Optical Internetworking Forum (OIF). The interface must operate bidirectionally at a payload data rate of 40 Gb/s with 0–25% forward error correction (FEC) overhead, up to a maximum of 50 Gb/s. The interface consists of 17 bidirectional GTX transceivers and logic to compensate skew differences between the transmission paths of the data channels.

Introduction

The SFI-5 system reference model (as defined by the OIF) is shown in Figure 1. SFI-5 is intended to interface between a SERDES component and an FEC processor, between an FEC processor and a framer, or directly between a SERDES component and an FEC processor. The reference model consists of 16 data channels and a 17th channel called the deskew channel (TXDSC/RXDSC), which transmits out-of-band data samples to enable an algorithm in the receiver to deskew the 16 data channels.

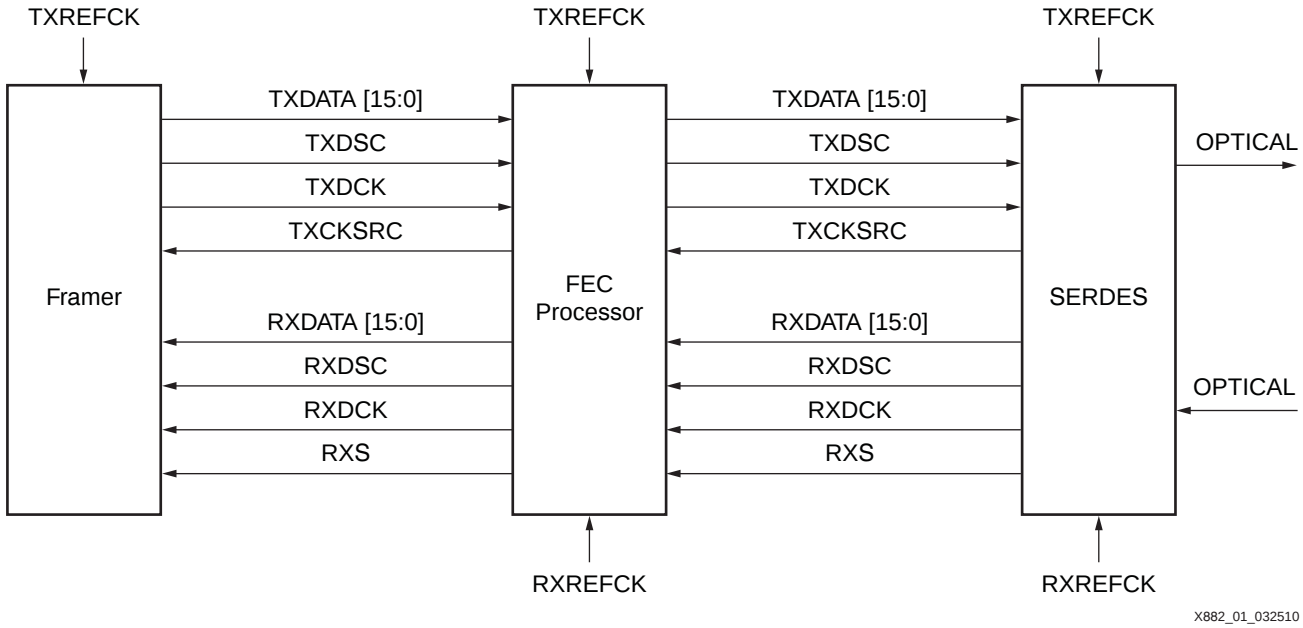

Figure 1: **SFI-5 System Reference Mode**

SFI-5 is a fully synchronous system, meaning that there is only a single reference clock. For example, on the link from the FEC processor to the SERDES in Figure 1, the source reference clock is the same as the sink reference clock. This synchronization can be accomplished using any of these three methods:

Method 1: TXREFCK is physically connected to both the FEC processor and the SERDES.

Method 2: If the FEC processor does not have access to TXREFCK, the SERDES must send the reference clock to the FEC processor via the TXCKSRC signal.

Method 3: If the SERDES does not have access to TXREFCK, the FEC processor must send the reference clock to the SERDES via the TXDCK signal.

In this reference design, TXREFCK is connected directly to both the source and sink devices of the SFI-5 interface (method 1). However, this reference design can easily be modified to support the clocking schemes described in methods 2 and 3 with no change to the logic design.

The same synchronous principles apply to the receive direction (e.g., SERDES to FEC processor) in Figure 1, except that the reference clock is RXREFCK. The OIF specification states that TXREFCK and RXREFCK can be separate clocks, or they can be tied together. Virtex-6 FPGAs support configurations in which TXREFCK and RXREFCK are tied together, as well as configurations in which TXREFCK and RXREFCK are different. This is because the Virtex-6 FPGA contains two separate differential clock inputs in each GTX Quad and two separate PLLs for the transmitter (TX) and receiver (RX) in each GTX transceiver. This reference design assumes that both TXREFCK and RXREFCK are tied together. The reason for using a single reference clock is discussed later in this document.

The signal RXS in Figure 1 is not used in this reference design. There is a port for RXS, but it is tied to `0`. If RXS is desired, this functionality must be added by the user. When set to `1`, the signal indicates that RXDCK and RXDATA are not derived from the optical receive signal.

The SFI-5 specification budgets 2 unit intervals (UI) of skew at the source device outputs. It budgets a further 3 UI of skew for the transmission lines to the sink device. The sink device is then required to deskew a minimum of 5 UI. The GTX transmitter lane-to-lane skew specification (T_{LLSKEW}) for the Virtex-6 FPGA is defined in the *Virtex-6 Data Sheet: DC and Switching Characteristics* [Ref 2] as 1 UI + 100 ps, which is less than the 2 UI requirement. With this, the transmission lines have the flexibility of taking more skew than the defined 3 UI, but the receiver should still see no more than 5 UI of skew.

The reference design supports the SFI-5 performance requirements, which state that the interface must operate between 40–50 Gb/s in all speed grades of the XC6VLX240T device. The maximum specified frequency of the recovered clock (F_RXREC) in the GTX transceiver exceeds the SFI-5 performance, as shown in Table 1.

Table 1: **Maximum Performance Targets of SFI-5 Reference Design in XC6VLX240T**

Speed Grade	Maximum F_RXREC (MHz)	Maximum Interface Performance (Gb/s)
–1	312.50	80
–2	406.25	104
–3	406.25	104

In LXT devices, the recovered clock is one-sixteenth the line rate. This application note focuses specifically on implementing SFI-5 in an XC6VLX240T device.

The hierarchy of the HDL modules composing the SFI-5 reference design is shown in Figure 2. Each of these modules is described in detail in this application note. The 17 GTX transceivers are wrapped in a single module that serves as the interface to the rest of the reference design. The TX is composed of deskew channel frame generation logic, and the RX is composed of barrel shifters and logic to adjust the delay of each channel to compensate skew between data channels. The TX and RX both have initialization sequences in which all circuitry is reset.

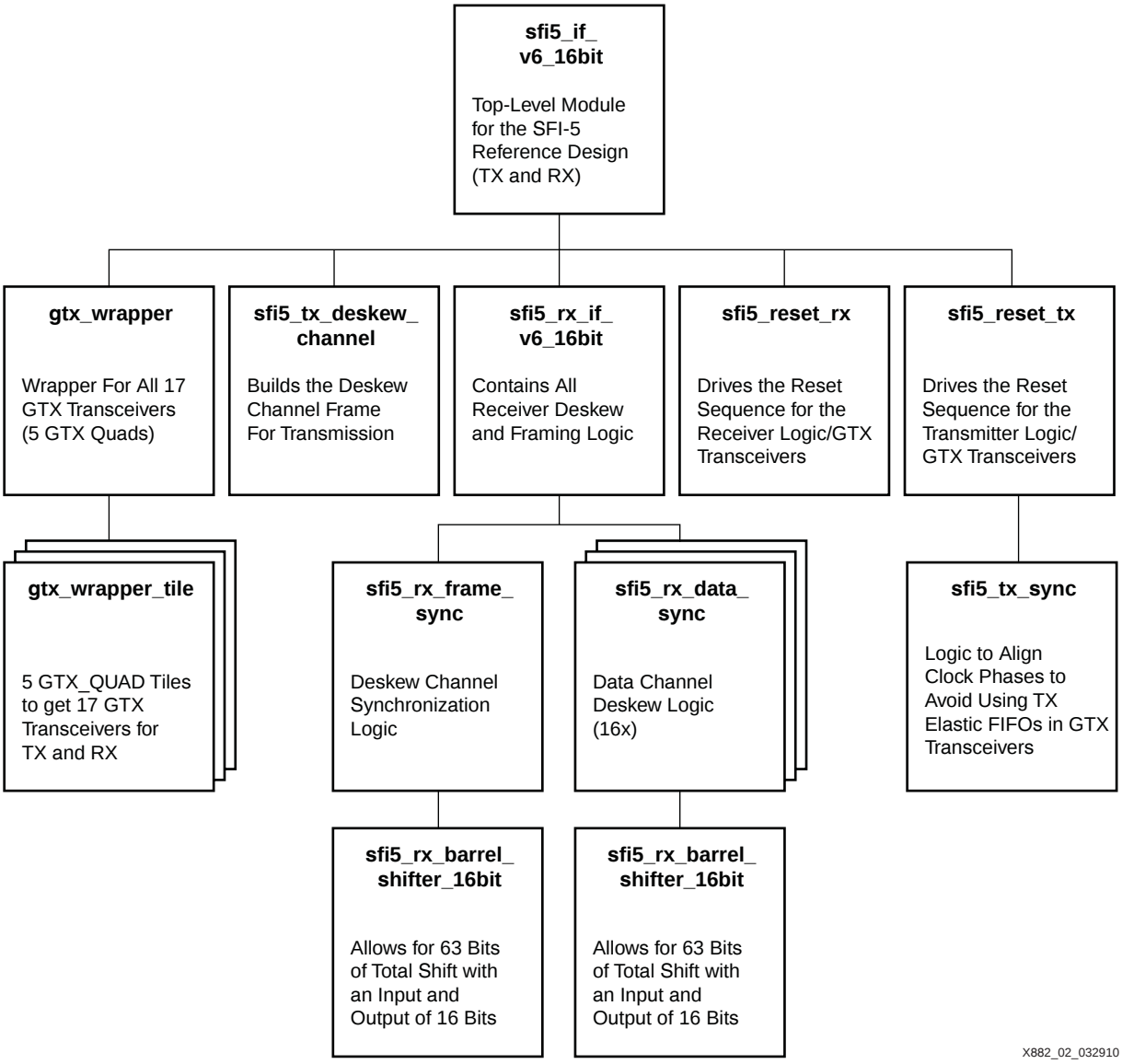

Figure 2: **Hierarchy of HDL Modules Composing SFI-5 Interface**

Port List and Descriptions

Table 2 lists the ports used in the SFI-5 interface. All signals are active-High unless stated otherwise. Optional settings should not be left unconnected. They must be set either to user-specific values or to the default values given in this table.

Table 2: **Port List and Description of SFI-5 Interface**

Port	Type (I/O)	Width	Clock Domain	Description
SFI-5 TX Interface Signals				
TXDATA_P	O	16	Line Rate	SFI-5 TX data channels (P-side).
TXDATA_N	O	16	Line Rate	SFI-5 TX data channels (N-side).
TXDSC_P	O	1	Line Rate	SFI-5 TX deskew channel (P-side).
TXDSC_N	O	1	Line Rate	SFI-5 TX deskew channel (N-side).
TXREFCK	I	2	txusrclk2	Reference clock input to GTX transceivers for TX and RX. Frequency is 1/16th line rate (156.25–195.3125 MHz, or 2.5–3.125 Gb/s).
TXREFCK_2	I	2	txusrclk2	Reference clock input to GTX transceivers for TX and RX. Must be the same frequency as TXREFCK and synchronous to it.
TXDCK	O	2	txusrclk2	Differential clock reference at one-sixteenth line rate forwarded to RX (optional to use in RX).
RXS	O	1	N/A	Receive status (not used, tied to 0).
SFI-5 RX Interface Signals				
RXDATA_P	I	16	Line Rate	SFI-5 RX data channels (P-side).
RXDATA_N	I	16	Line Rate	SFI-5 RX data channels (N-side).
RXDSC_P	I	1	Line Rate	SFI-5 RX deskew channel (P-side).
RXDSC_N	I	1	Line Rate	SFI-5 RX deskew channel (N-side).
Global Signals				
i_RST	I	1	txusrclk2	Global reset to initiate reset sequence of TX/RX.
o_RESETDONE	O	1	txusrclk2	Indicates that all GTX transceivers have completed their reset sequences. Clear by asserting i_CLEAR_MISMATCHES.(1)
o_GTXPLL_LOCK	O	1	txusrclk2	Indicates that all shared PMA PLLs are locked. Clear by asserting i_CLEAR_MISMATCHES.
System-side TX Data/Clock Signals				
iv_TXDATA00_IN [15:0] iv_TXDATA01_IN [15:0] … iv_TXDATA15_IN [15:0]	I	256	txusrclk2	System-side data input to SFI-5 TX. Serialized by GTX transceivers and transmitted on TXDATA_P and TXDATA_N.
o_TXUSRCLK2	O	1	txusrclk2	User-accessible connection to txusrclk2.
System-side TX Diagnostics				
o_TX_INIT_DONE	O	1	txusrclk2	Indicates that TX reset sequence is complete. Clear by asserting i_CLEAR_MISMATCHES.
i_INSERT_FRAME_ERROR	I	1	txusrclk2	Insert one bit error in frame header of deskew channel.
i_INSERT_DATA_ERROR	I	1	txusrclk2	Insert error on data channel 15.

Table 2: **Port List and Description of SFI-5 Interface *(Cont'd)***

Port	Type (I/O)	Width	Clock Domain	Description
i_LOOPBACK	I	3	async	GTX transceiver loopback setting for troubleshooting: 000: Normal (no loopback). 001: Near-end physical coding sublayer (PCS) loopback. 010: Near-end physical medium attachment (PMA) loopback. 100: Far-end PMA loopback. 110: Far-end PCS loopback.
System-side RX Diagnostics				
o_RXOOA	O	1	rxusrclk2	RX out of alignment. One or more data channels is misaligned.
o_RXOOA_HISTORY	O	1	rxusrclk2	RX out of alignment history. One or more data channels is misaligned. Clear by asserting i_CLEAR_MISMATCHES.
o_RXLOF	O	1	rxusrclk2	RX loss of frame. Framer is not locked to deskew channel frame.
o_RXLOF_HISTORY	O	1	rxusrclk2	RX loss of frame. Framer is not locked to deskew channel frame. Clear by asserting i_CLEAR_FRAME_ERRORS.
i_CLEAR_FRAME_ERRORS	I	1	rxusrclk2	Clears frame error count and o_RXLOF_HISTORY.
i_CLEAR_MISMATCHES	I	1	rxusrclk2	Clears mismatch counts and all diagnostic history bits.
ov_FRAME_ERRORS	O	32	rxusrclk2	Wrap-around running count of frame errors.
ov_FRAMES_RECEIVED	O	32	rxusrclk2	Wrap-around running count of frames received.
ov_DATA_MISMATCHES_CH00 ov_DATA_MISMATCHES_CH01 … ov_DATA_MISMATCHES_CH15	O	32	rxusrclk2	Running count of data mismatches when compared to deskew channel.
ov_RXFRAME_SHIFT	O	6	rxusrclk2	Barrel shifter setting of deskew channel.
ov_RXDATA_SHIFT_CH00 ov_RXDATA_SHIFT_CH01 … ov_RXDATA_SHIFT_CH15	O	6	rxusrclk2	Barrel shifter setting of each data channel.
o_RX_INIT_DONE	O	1	txusrclk2	Indicates that RX reset sequence is complete. Clear by asserting i_CLEAR_MISMATCHES.
o_RX_BUFFER_UNDERFLOW	O	1	txusrclk2	GTX RX elastic buffer underflow. Clear by asserting i_CLEAR_MISMATCHES.
o_RX_BUFFER_OVERFLOW	O	1	txusrclk2	GTX RX elastic buffer overflow. Clear by asserting i_CLEAR_MISMATCHES.
Optional Settings				
i_TX_PREEMPHASIS	I	4	async	Driver pre-emphasis (pre-cursor) setting of all GTX transceivers in SFI-5 interface. For details on pre-emphasis settings, see the *Virtex-6 FPGA GTX Transceivers User Guide* [Ref 3].
i_TX_POSTEMPHASIS	I	5	async	Driver post-emphasis (post-cursor) setting of all GTX transceivers in SFI-5 interface.

Table 2: **Port List and Description of SFI-5 Interface *(Cont'd)***

Port	Type (I/O)	Width	Clock Domain	Description
i_TX_INHIBIT	I	1	async	When asserted, GTX TX drivers are disabled.
i_TX_DIFF_CTRL	I	4	async	Driver output swing of all GTX transceivers in SFI-5 interface.
i_RX_EQUALIZATION_MIX	I	3	async	Receiver equalization control.
i_FRAMES2LOCK	I	7	async	User-defined threshold for the number of consecutive matching frames that need to be seen before deasserting o_RXLOF. Valid settings = 0–127 (decimal). Default = 3F (63 decimal).
i_FRAMES2UNLOCK	I	7	async	User-defined threshold for the number of frame mismatches that need to be seen before asserting o_RXLOF. Valid settings = 0–127 (decimal). Default: 3F (63 decimal).
iv_MISMATCHES_2_UNLOCK	I	7	async	User-defined threshold for the number of consecutive data mismatches that need to be seen before asserting o_RXOOA. Valid settings = 0–127 (decimal). Default = 3F (63 decimal).

SFI-5 Clocking

This section describes the various clock domains in the SFI-5 interface. TXREFCK is the clock defined in the OIF specification as the frequency reference for the Framer–FEC–SERDES links in Figure 1. RXREFCK is defined as the frequency reference for the SERDES–FEC–Framer links. As discussed in Introduction, page 1, TXREFCK and RXREFCK in this reference design come from the same clock source. For this reason, the label TXREFCK is used interchangeably to refer to TXREFCK and RXREFCK. TXREFCK must be provided by an oscillator directly to the dedicated reference clock input of a GTX transceiver. Dedicated routing from the reference clock input to the GTX transceivers ensures that the GTX transceivers receive a clock with minimal jitter. TXREFCK can also be provided to the GTX transceivers via a global clock network in the FPGA, but the path is not ideal for maintaining the highest quality signal. This reference design uses clock sources provided directly to the reference clock input pins of the GTX transceiver.

Figure 3 shows the clocking architecture of the SFI-5 transmitter. TXREFCK is provided by an external clock source to the REFCLK0 of Quad 113 which is distributed to transceiver channels 0–11 (Quad 112 to Quad 114.) Each Quad contains four GTX transceivers. TXREFCK_2 is also provided by an external clock source to REFCLK0 of Quad 115, and is distributed to transceiver channels 12–15, as well as the deskew channel (DSC), Quad 115, and Quad 116. Two TXREFCK clock signals are required because the *Virtex-6 FPGA GTX Transceivers User Guide* [Ref 3] states that a reference clock input on a GTX transceiver of a source Quad can only span a maximum of one Quad tile above and below that Quad. Therefore, a reference clock can only span 12 GTX transceivers, which is insufficient to clock the 17 channels of the SFI-5 interface. TXREFCK_2 must be synchronous to TXREFCK and have the same frequency, although their phase relationship is not important.

The TXOUTCLK port of the DSC channel is used to gain access to TXREFCK for the system logic. TXOUTCLK is a forwarded version of CLKIN that is not affected by resetting the GTX transceiver. This is important because the initialization logic for the TX and RX interfaces must run on a clock that is independent of the reset sequence. All logic in the `sfi5_reset_tx` module is driven exclusively by TXREFCK via the TXOUTCLK port of the DSC channel. TXOUTCLK is used to generate TXUSRCLK2 and TXDCK. TXDCK is the same as TXOUTCLK and is forwarded to the receiver as the optional reference clock input from the transmitter. The only logic in the transmitter, other than the initialization logic, is the deskew channel frame generation. This logic is driven by TXUSRCLK2. The user interface of all GTX transceivers is clocked (using TXUSRCLK2) by the same clock source (TXOUTCLK) from the DSC channel to minimize the skew between GTX transceivers used.

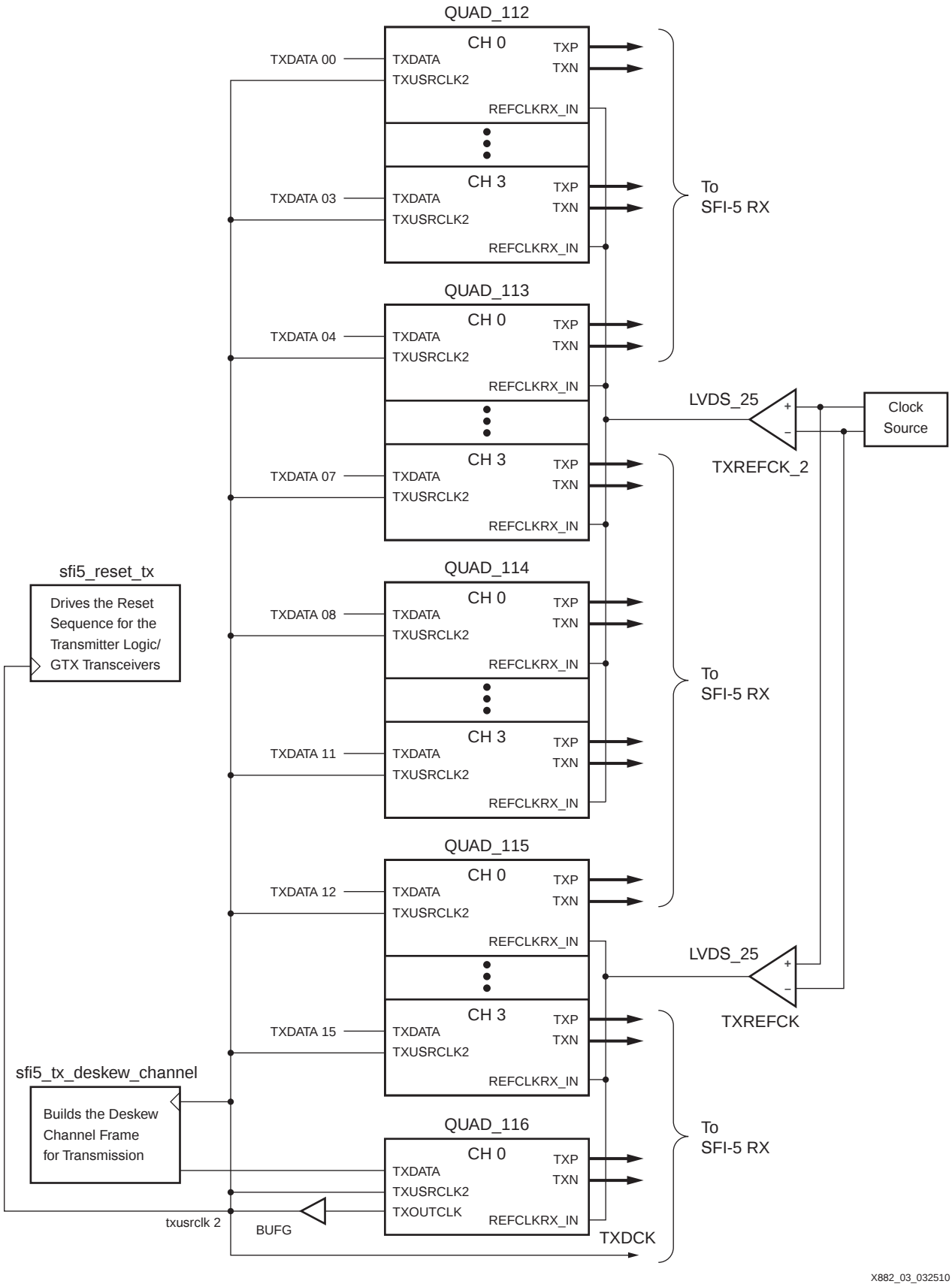

Figure 3: **SFI-5 Transmitter Clocking**

Figure 4 shows the clocking architecture of the SFI-5 receiver. The reference clock inputs TXREFCK and TXREFCK_2 are the same inputs shown in Figure 3 because the TX and RX GTX transceivers share the same reference clock, i.e., it uses only the RX PLL to generate the serial clock for both TX and RX to reduce the power. However, RXUSRCLK2 is not generated from TXREFCK, but rather from RXRECCLK, the recovered clock from the incoming data stream on the deskew channel. RXRECCLK is one-sixteenth the line rate, which makes it the frequency required for RXUSRCLK2. Framing and data deskewing logic is driven by RXUSRCLK2. Although it is not shown in Figure 4, RXDATA15 also feeds into its own `sfi5_rx_data_sync` module. Only the logic in the `sfi5_reset_rx` module is driven by TXREFCK via TXOUTCLK because of the requirement that the clock be independent of the reset sequence.

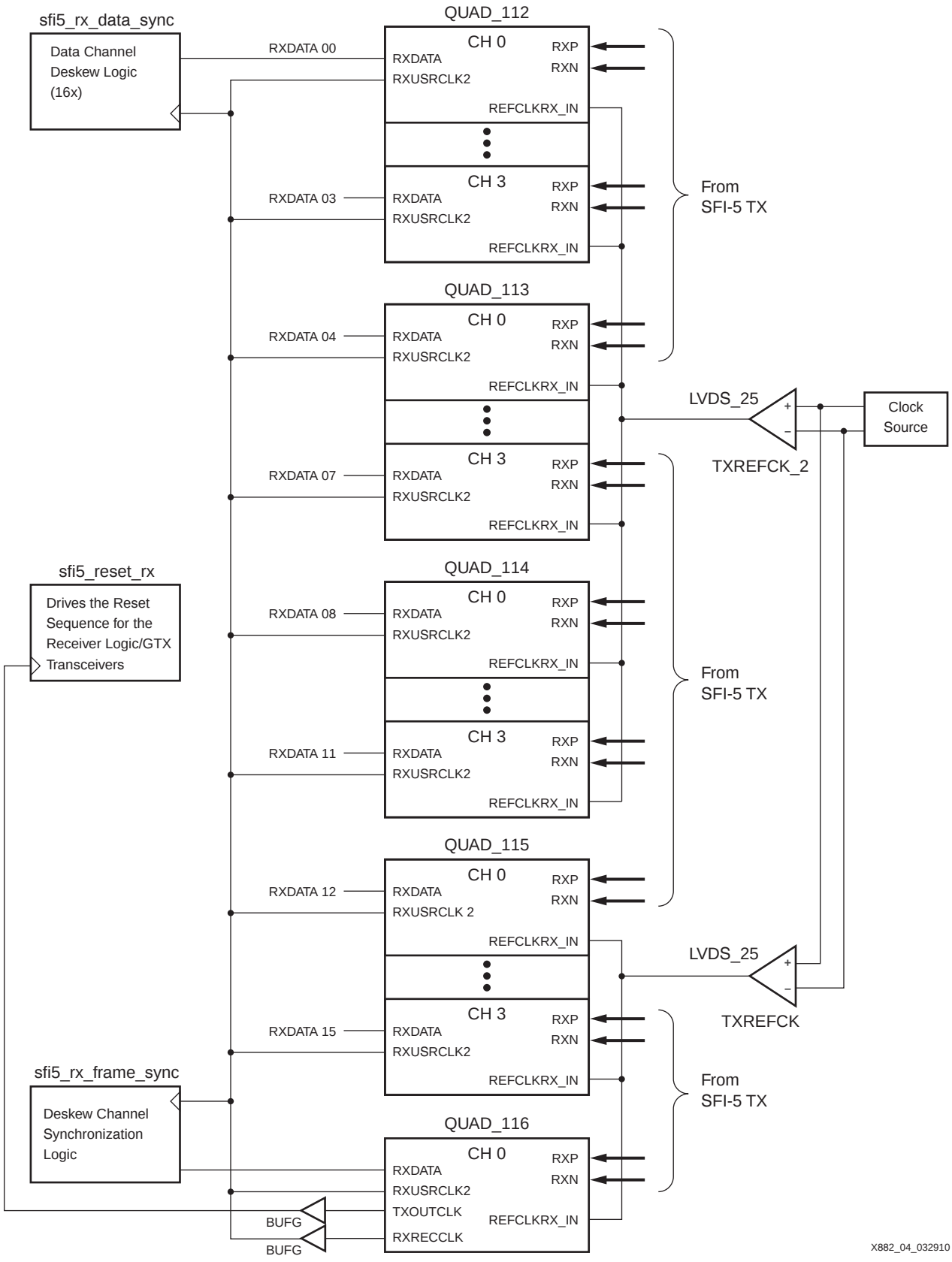

Figure 4: **SFI-5 Receiver Clocking**

SFI-5 Transmitter

The SFI-5 transmitter takes 16 inputs of 16 bits each (256 bits total) and stripes each 16-bit word across all 16 GTX transceivers. Therefore, the 16-bit word that each GTX transceiver transmits has one bit from each of the 16 inputs. The SFI-5 receiver in this reference design "unstripes" the data to return it to the normal order. It is important that the transmitter and receiver are complements of each other. If striping is not desired in either the TX or RX of this reference design, the assignment statements that perform the striping in the `sfi5_if_v6_16bit` module can be replaced by simple one-to-one assignments.

After the data inputs have been striped, they form 16 new words, each of which is transmitted by one of the 16 GTX transceivers. The 16 words arrive at the inputs of the GTX transceivers at the same time, but over the course of the transmission process, some words can arrive at the receiver sooner or later than others. There is no framing information in the data channels themselves. Thus, there must be a way to record the proper order of the data words, such that the receiver can later use that recorded order to realign the data words. This is done by recording fragments of the data words onto the deskew channel before transmission when the order is still known to be correct. The deskew channel content is strictly defined by the OIF specification. The deskew frame is shown in Figure 5.

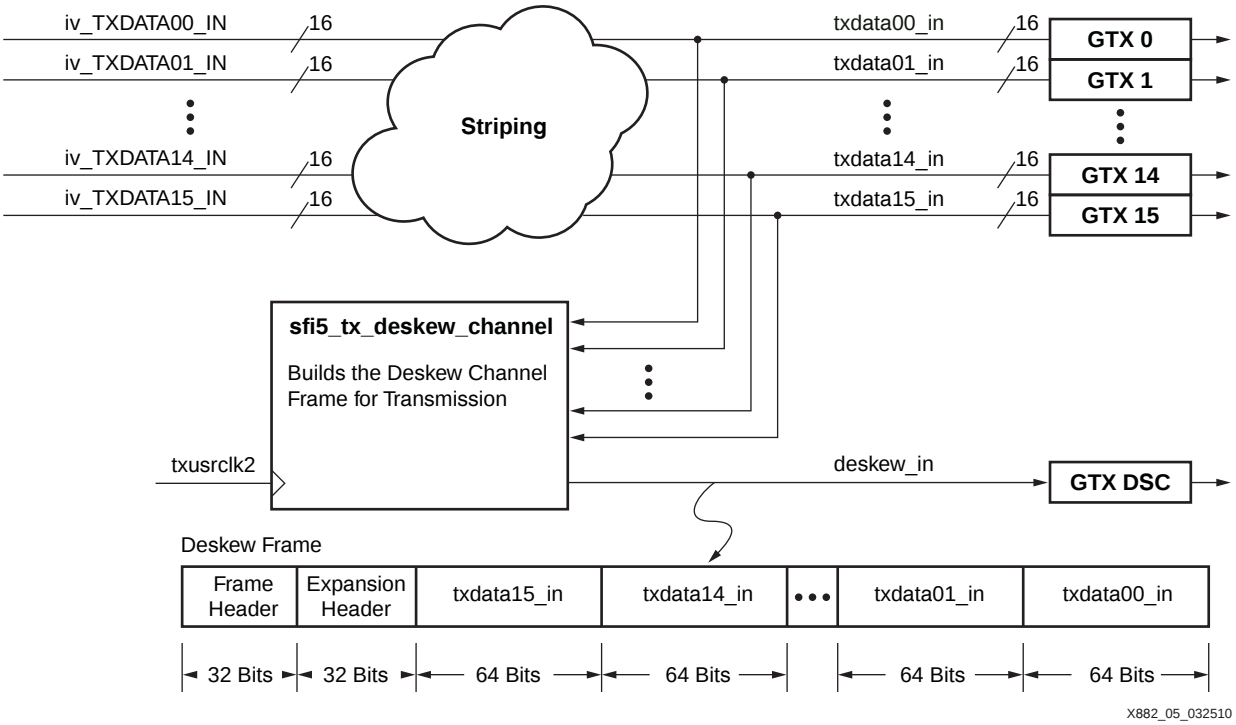

Figure 5: **SFI-5 Transmitter Datapath and Deskew Channel Generation**

The state machine that generates the deskew frame is shown in Figure 6. In normal operation, the state machine has no input dependencies. Each state represents a different setting in a large multiplexer in the datapath that selects from one of the 16 data channels (or static frame bits). The state machine is a loop that has no exit condition except for reset, and each iteration of the loop generates one frame.

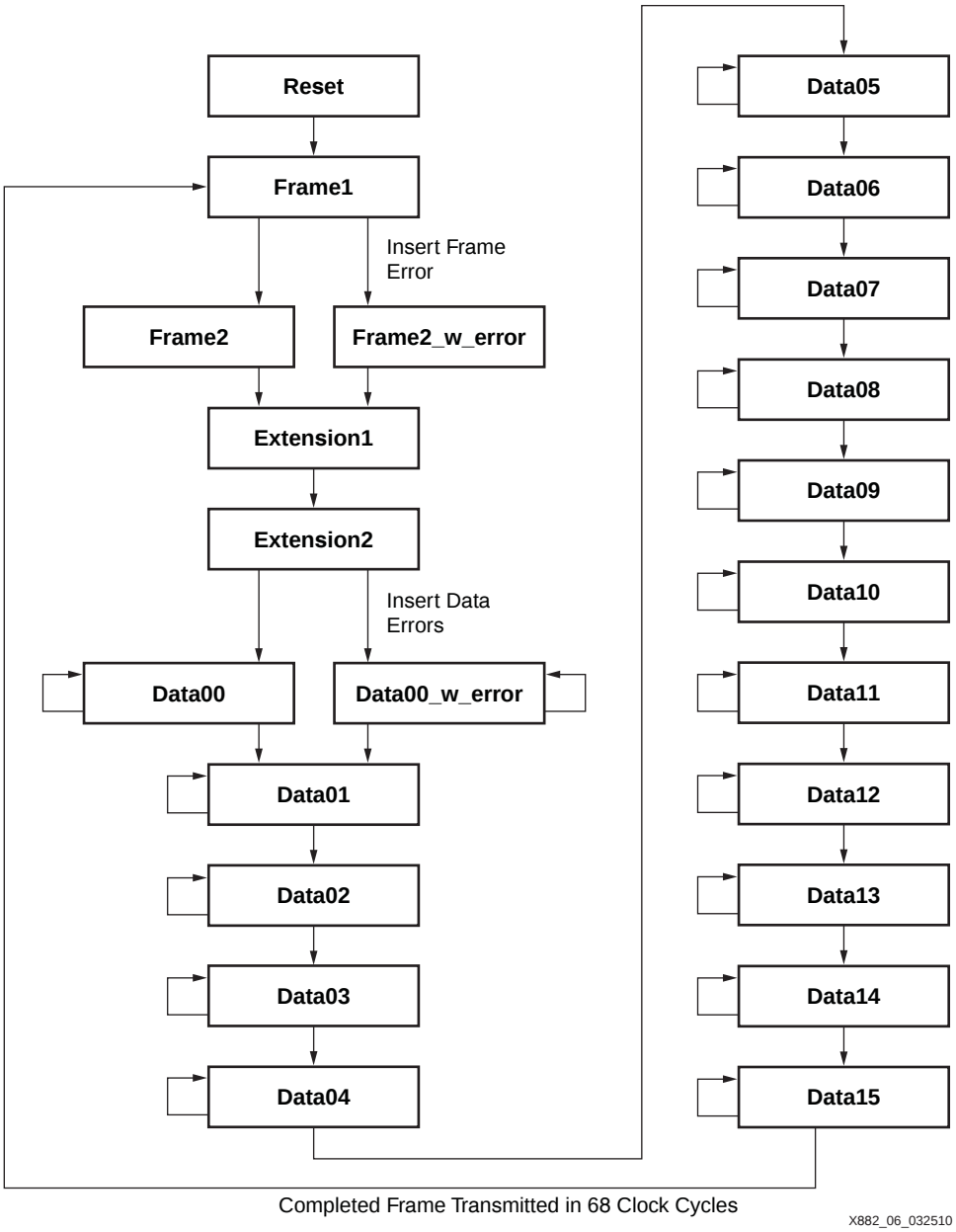

Figure 6: **Deskew Frame Generation State Machine**

Two input conditions are provided for diagnostic purposes: frame error insertion and data error insertion. When i_INSERT_FRAME_ERROR is asserted, the state machine replaces the correct frame (`F6F6 2828`) with an incorrect frame (`F6F6 2928`). When i_INSERT_DATA_ERROR is asserted, the state machine inverts the first 16 bits of the 64-bit fragment of data channel 15. The names of the states in Figure 6 are taken directly from the Verilog module. The states Data00, Data01, etc., correspond to the transmission of txdata15_in, txdata14_in, etc. The names of the states are in the reverse order of the data channels that are copied to the DSC channel in those states.

The initialization sequence of the transmitter is controlled by the state machine shown in Figure 7.

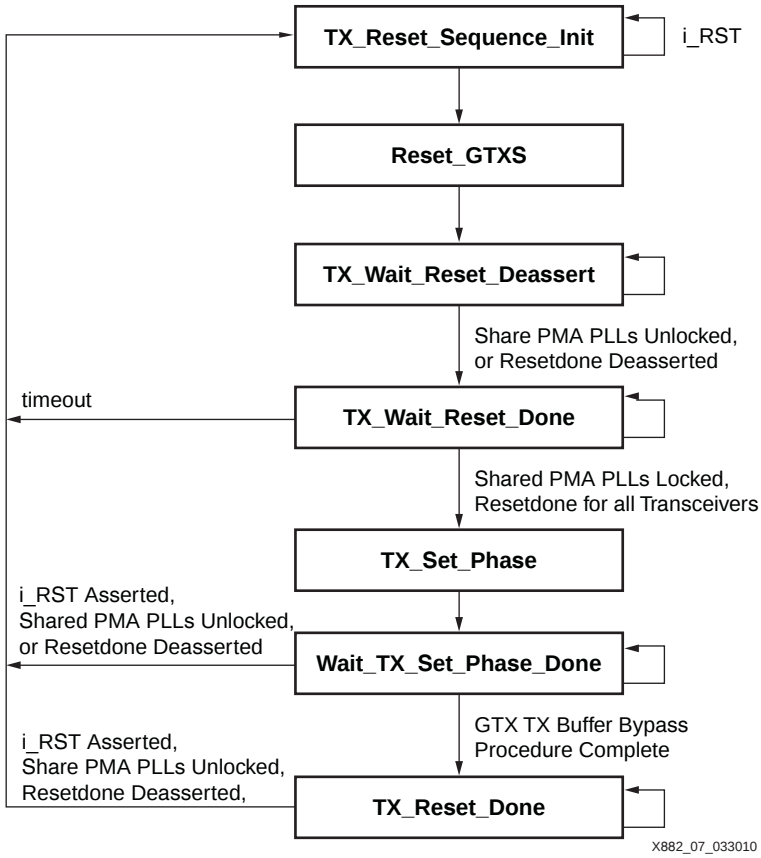

Figure 7: **Initialization Sequence of SFI-5 Transmitter**

The TX initialization sequence consists of two main steps:

1. Reset all 17 GTX transceivers by asserting the GTXRXRESET and GTXTXRESET inputs of all GTX tiles. This initiates the full reset sequence of all circuitry in the GTX tiles. The TX_Wait_Reset_Deassert state is included in the state machine because the RESETDONE ports do not deassert immediately when GTXRXRESET and GTXTXRESET are asserted. Without the TX_Wait_Reset_Deassert state, the state machine prematurely continues to the next step without waiting for the reset sequence to complete.
2. Initialize the TX phase-alignment circuit, which bypasses the TX buffer and guarantees low output skew between GTX transceivers. This procedure follows the instructions for bypassing the buffer given in the *Virtex-6 FPGA GTX Transceivers User Guide* [Ref 3].

After these steps are complete, the state machine rests in the TX_Reset_Done state unless another reset sequence is initiated, the PMA PLLs unlock, or the RESETDONE output of any GTX transceiver is deasserted. If the i_RST input is manually asserted to the SFI-5 interface, the reset sequence starts over regardless of the current state of the state machine.

The TX_Wait_Reset_Done state has an additional exit condition of TIMEOUT. The reset sequence starts over if the RESETDONE outputs of the GTX transceivers are not asserted after one million TXREFCK cycles.

SFI-5 Receiver

The SFI-5 receiver recovers the data from the 16 GTX transceiver inputs and presents 256 bits of data at the RXUSRCLK2 rate to the system side of the interface. The receiver must adjust the delay of each of the incoming datapaths to guarantee alignment between channels. The SFI-5 receiver datapath is shown in Figure 8.

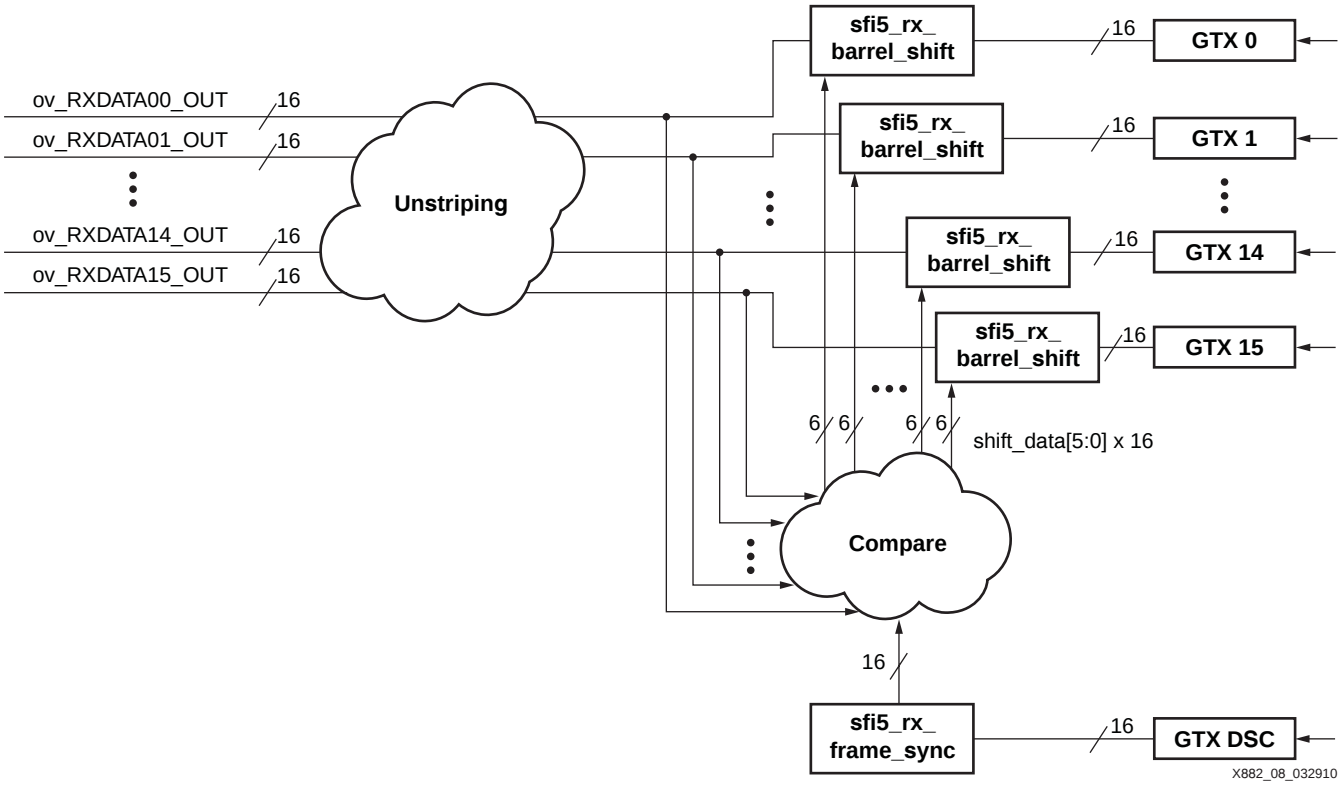

Figure 8: **Receiver Datapath**

Each data channel has a 79-bit barrel shifter in its path that allows a state machine to select between 63 different delayed versions of the 16-bit output (63 + 16 = 79) of the GTX transceiver. A barrel shifter in the deskew channel datapath facilitates the framing process. The 256 data bits are divided into 16 words of 16 bits each.

The receiver assumes that the data has been striped across the 16 GTX transceivers, so the words are unstriped before being presented to the system. It is important that the transmitter and receiver are complements of each other. If striping is not desired in either the TX or RX of this reference design, the assignment statements that perform the striping in the `sfi5_if_v6_16bit` module can be replaced by simple one-to-one assignments.

Barrel Shifter

The barrel shifter is the fundamental building block of the receiver. All datapaths and the deskew channel path pass through a 79-bit barrel shifter like the one shown in Figure 9. With 79 bits, the barrel shifter can add 0 to 63 bit times of delay to a 16-bit word. Sixteen new bits are added to the left of the chain on each RXUSRCLK2 cycle, and all other bits in the chain are rotated by 16 to the right. The 16 bits farthest to the right are lost when new data is added. A shift value setting of `00` selects the most recent data bits for the output of the barrel shifter.

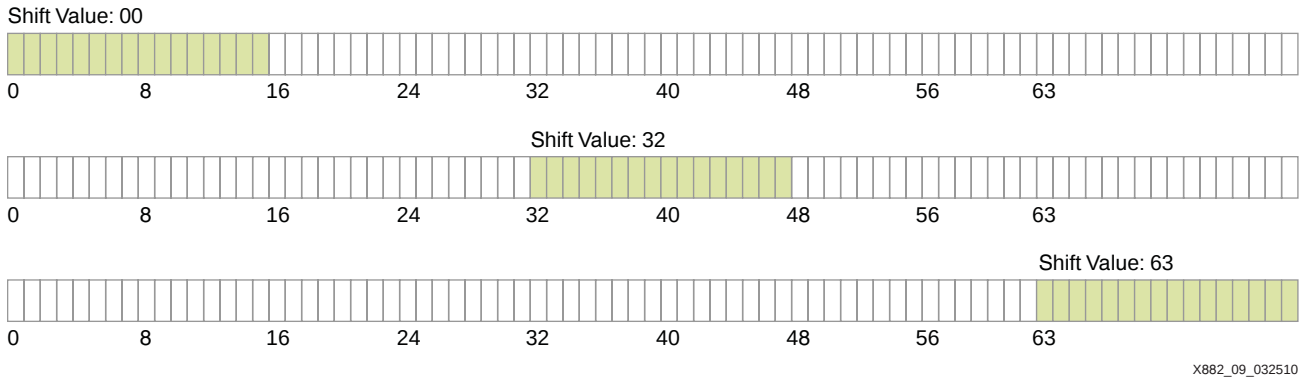

Figure 9: **79-bit Barrel Shifter Chain**

The shift value is a 6-bit control signal ($2^6 = 64$). There are six levels of 2-to-1 MUXs connected in series that reduce the data selection from 79 bits (the full barrel shifter) to 16 bits (the desired output width). Each bit of the shift value is a select line to one level of 2-to-1 MUXs. The most significant shift value bit must reduce the entire chain from 79 bits ($2^6 – 1 + 16$) to 47 bits ($2^5 – 1 + 16$). This first reduction corresponds to MUX level 1, shown in Figure 10, and is accomplished using 47 2-to-1 MUXs. If the most significant shift value bit is 0, the 47 bits selected are [46:0]. If the most significant shift value bit is 1, the 47 bits selected are [78:32].

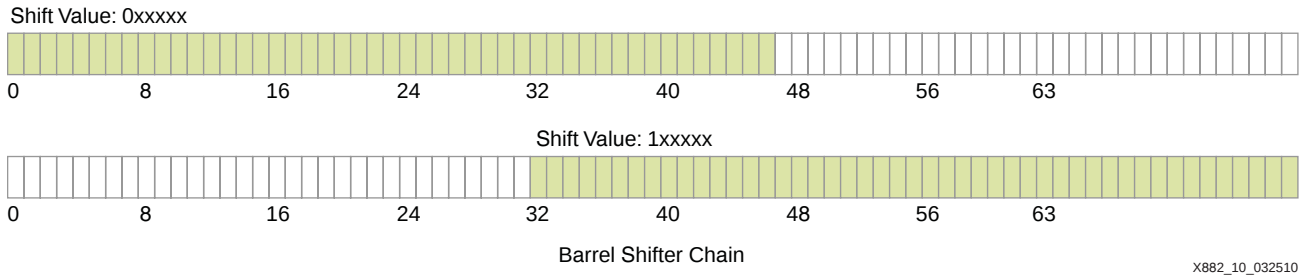

Figure 10: **Barrel Shifter MUX Level 1 Reduction**

All six levels of MUXs and their corresponding reductions are shown in Figure 11. To ease timing closure, registers are added at the outputs of MUX levels 3 and 6.

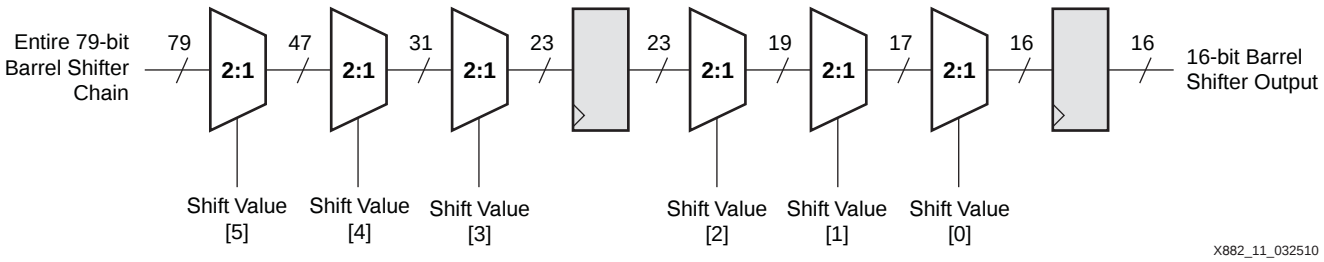

Figure 11: **Barrel Shifter Output Selection Logic**

Deskew Frame Synchronization

Before the receiver can begin deskewing the data channels, it must first synchronize to the incoming deskew channel. To do this, a state machine in the receiver must scan the incoming deskew channel data in search of the frame header (`0xF6F62828`). The state machine must monitor the incoming deskew channel data for at least 68 cycles, which is the size of the deskew frame (actual scanning time is 128 cycles). If the frame is not found, it is possible that the deskew channel is not aligned to the frame boundary. After scanning for 128 cycles without

finding the frame, the state machine adds one bit time of delay by incrementing the shift value of the barrel shifter in the deskew channel path. This process never repeats more than 16 times because the size of the barrel shifter output is 16 bits, and there must be one correct alignment of the frame header (0xF6F6) in any 16-bit window. The deskew channel frame synchronization state machine is shown in Figure 12.

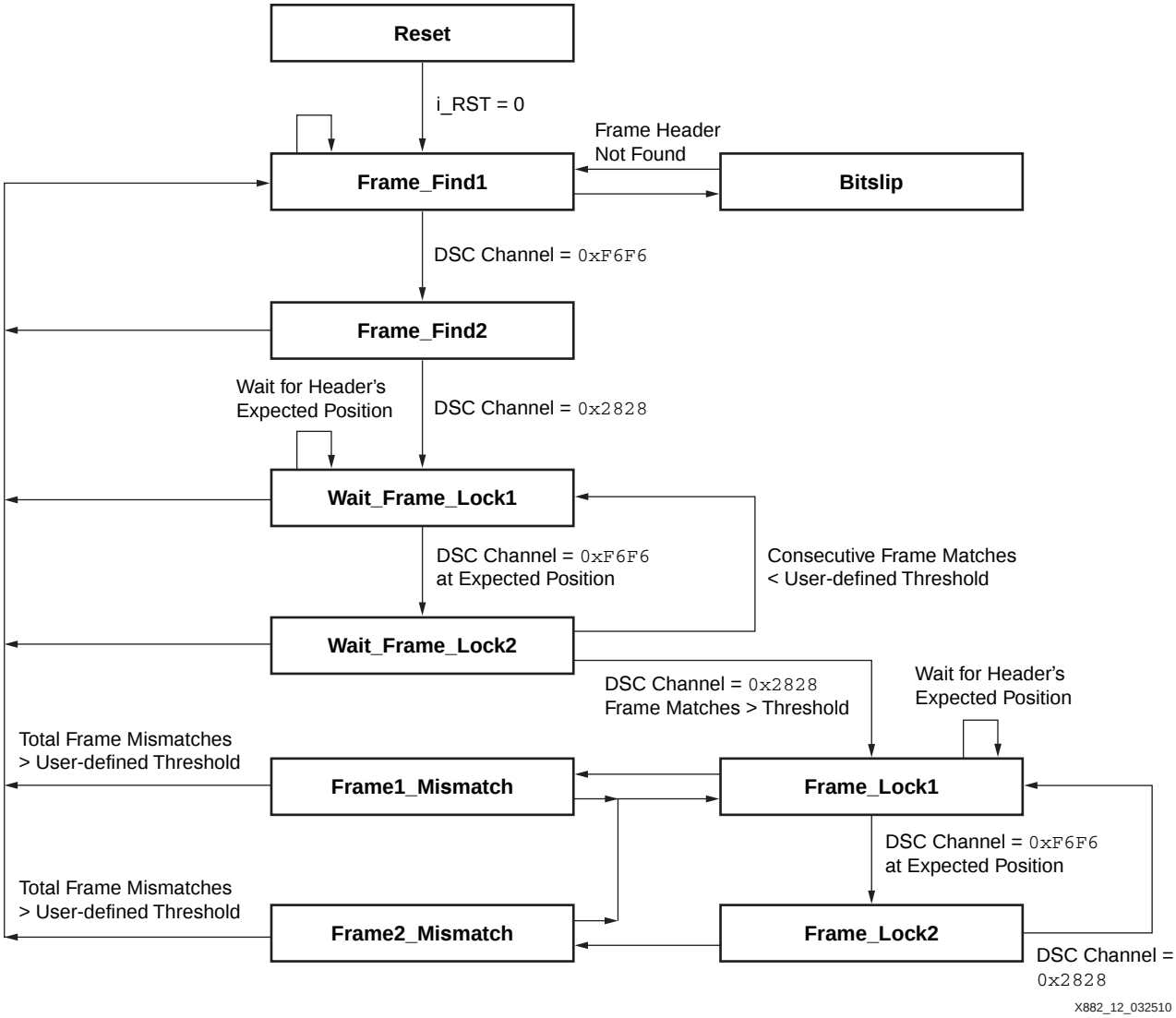

Figure 12: **Deskew Channel Frame Synchronization State Machine**

After the frame synchronization algorithm is complete, the barrel shifter setting of the deskew channel becomes an anchor for all the data channels deskew logic. Each data channel's skew is defined relative to the deskew channel. If the deskew channel barrel shifter is set to 0, the data deskew algorithm can deskew data that is +63 UI later than the deskew channel, but only 0 UI before the deskew channel. This asymmetric deskew capability is not desirable because data channels can be slower or faster than the deskew channel, and both contingencies must be addressed.

Ideally, the barrel shifter setting of the deskew channel is 32 because that setting results in perfectly symmetric deskew capability (±32 UI). Unfortunately, the deskew channel setting cannot be simply set to a specific value. It is determined by the frame synchronization algorithm. The frame synchronization algorithm can find the correct alignment setting in a search field of any 16 barrel shifter settings. For this reason, the deskew channel barrel shifter is initialized to 24 and increments only as high as 39 (search field = 16). The frame

synchronization algorithm finds the setting with the correct alignment somewhere between 24 and 39. Allowing for the worst cases of 24 and 39, there is still a minimum of ±24 UI of skew compensation capability (both *63 – 39* and *24 – 0* equal 24). The synchronization process consists of these steps:

1. Find the frame header in the incoming deskew channel, adjusting the barrel shifter setting as necessary.
2. After the first frame is detected, keep a running count of frames detected. As soon as the user-defined threshold (i_FRAMES2LOCK) for declaring frame lock is reached, RXLOF is driven Low to indicate that the deskew channel is synchronized.
3. When the framer is in the locked state, it continuously checks every frame. If an error is detected in any part of the frame, a mismatch is recorded. As long as the total number of frame mismatches does not exceed the user-defined threshold (i_FRAMES2UNLOCK), the state machine returns to the locked state. When the threshold is exceeded, RXLOF is asserted and the algorithm starts over from the beginning.

Data Channel Deskew

After the deskew channel barrel shifter setting has been established by the frame synchronization algorithm, each data channel can be compared to the deskew channel. The barrel shifter setting of each data channel can be adjusted until the data content of that channel matches the 64-bit data fragment contained in the deskew frame. Each data channel is initialized with a barrel shifter setting of 0, and every setting from 0 to 63 is searched to find the data that matches the deskew channel. After all 16 channels complete this process, each channel finishes with a unique barrel shifter setting that is both a measure of skew relative to the deskew channel and a measure of skew relative to the other data channels. The smallest barrel shifter setting of the 17 channels subtracted from the largest barrel shifter setting yields the peak skew of the entire interface from TX to RX. Figure 13 shows actual measurements of skew compensation collected on a 50 Gb/s interface (3.125 Gb/s x 16 channels).

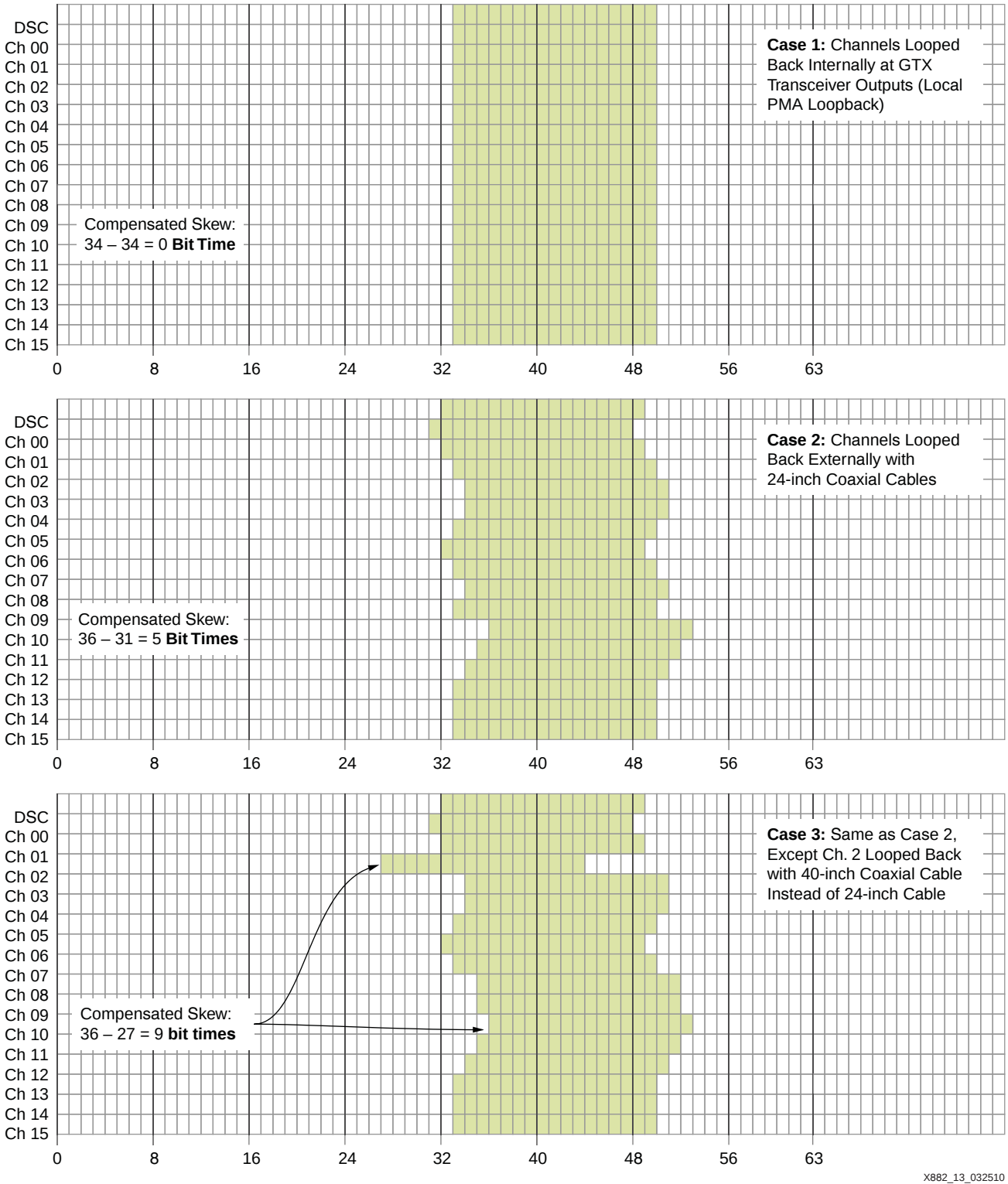

Figure 13: **Barrel Shifter Selections (3.125 Gb/s x 16 Channels)**

Every data channel and the deskew channel have uniquely determined barrel shifter settings, each creating a picture of the skew between the channels. To illustrate this, each of the three cases introduces a different amount of skew on the link. In case 1, the transmitter is looped back to the receiver internally (minimal skew). The results show that every data channel independently chooses the same barrel shifter setting of 34, and the total compensated skew is

0 UI. In case 2, the channels are looped back externally with 24-inch cables. The data channels now have different barrel shifter settings because the datapaths have differences in latency due to package routing, board routing, and cable lengths. The compensated skew is 5 UI in case 2. To accentuate the skew compensation even further, the cable on channel 2 is replaced by a cable that is 40 inches long in case 3. This produces a very visible effect on channel 2, which now has a significantly different barrel shifter setting. From case 2 to case 3, the channel 2 setting changes from 33 to 27. The setting decreases (move towards the left) because the most recent data is added to the barrel shifter on the left. Because channel 2 takes longer to get to the receiver in case 3, the state machine has to choose a more recent barrel shifter position from which to read the data (27 is more recent than 33). As for the actual magnitude of the change from case 2 to case 3, the theoretical calculation matches the measurement well:

Measured skew difference: 33 – 27 = 6 UI.

Added delay: (16 inches of extra cable) × (113 ps/inch signal velocity in coaxial cable) = 1.808 ns.

Theoretical skew difference: (1.808 ns added delay) ÷ (0.32 ns period of 3.125 Gb/s) = 5.65 UI.

Figure 14 shows another set of actual measurements of skew compensation. It is for the same device as in Figure 13, but the data rate is now 40 Gb/s (2.5 Gb/s x 16 channels).

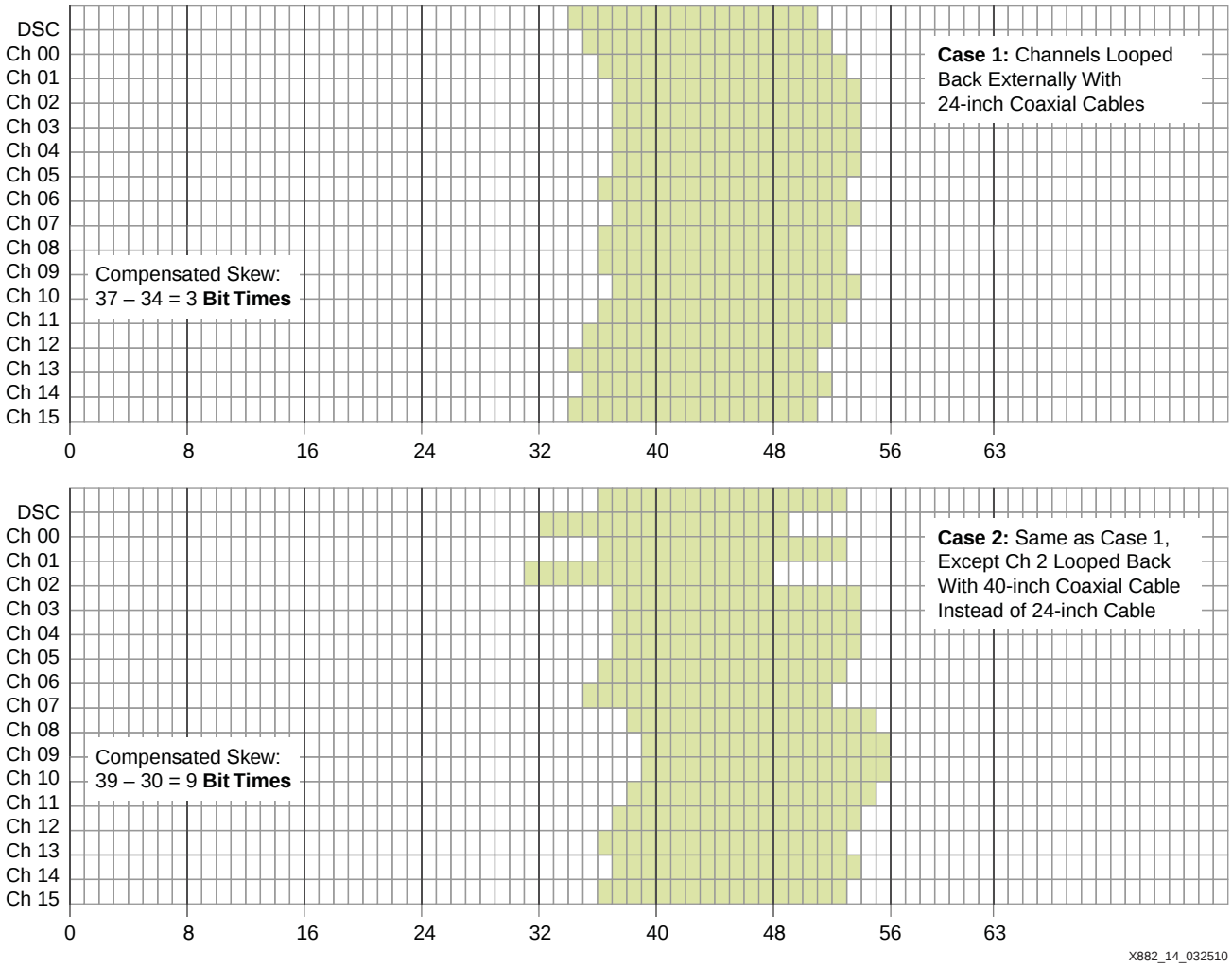

Figure 14: **Barrel Shifter Selections (2.5 Gb/s x 16 Channels)**

In case 1, all channels are looped back externally with 24-inch cables. In case 2, the channel 2 cable is replaced by a 40-inch cable. The result illustrates how the channel 2 barrel shifter works. In case 1, the channel 2 is set to 37. After adding 16 inches of cable, the expectation is that the setting would decrease (move to the left) according to this calculation:

Measured skew difference: 36 – 31 = 5 UI.

Added delay: (16 inches of extra cable) × (113 ps/inch signal velocity in coaxial cable) = 1.808 ns.

Theoretical skew difference: (1.808 ns added delay) ÷ (0.400 ns period of 2.5 Gb/s) = 4.52 UI.

Another source of skew in the receiver is the serial-to-parallel converter. Because the RXRECCLKs of the different GTX transceiver channels have no fixed phase relationship to one another, the serial-to-parallel conversion stage can introduce between 1 to 16 bit-times of skew (for a 16-bit datapath).

The amount of skew that can be compensated by the receiver must be quantified. How much skew is allowed at the input pins of the receiver? With a 63-bit barrel shifter, the absolute maximum skew compensation of the receiver is ±32 UI. However, some of that margin is lost to the uncertainty of the deskew channel setting, which is not always set to the ideal setting of 32. It can be set as low as 24 or as high as 39. The explanation for this is discussed in Deskew Frame Synchronization, page 15. The ideal margin is reduced from ±32 UI to ±24 UI.

The skew budget is further reduced by the receive path, which contributes a maximum of ±16 UI for the serial-to-parallel conversion. That leaves an absolute minimum of ±8 UI left over for the skew budget at the input pins of the receiver, as shown in Figure 15. The OIF SFI-5 specification for the minimum jitter that the receiver must tolerate is ±5 UI of skew.

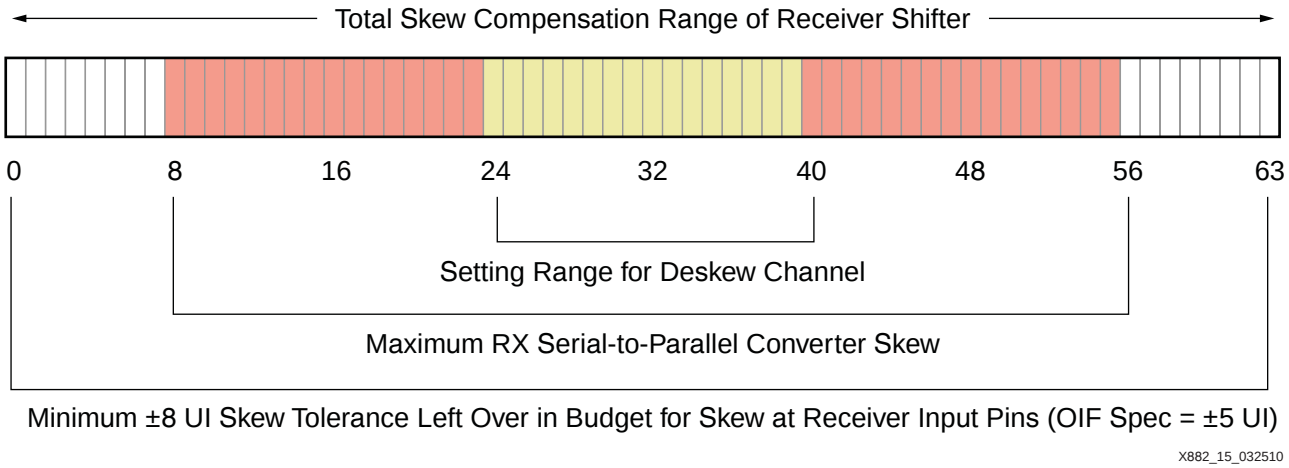

Figure 15: **Receiver Skew Budget**

The algorithm that deskews each data channel by controlling the barrel shifter setting and comparing the data to the deskew channel is shown in Figure 16.

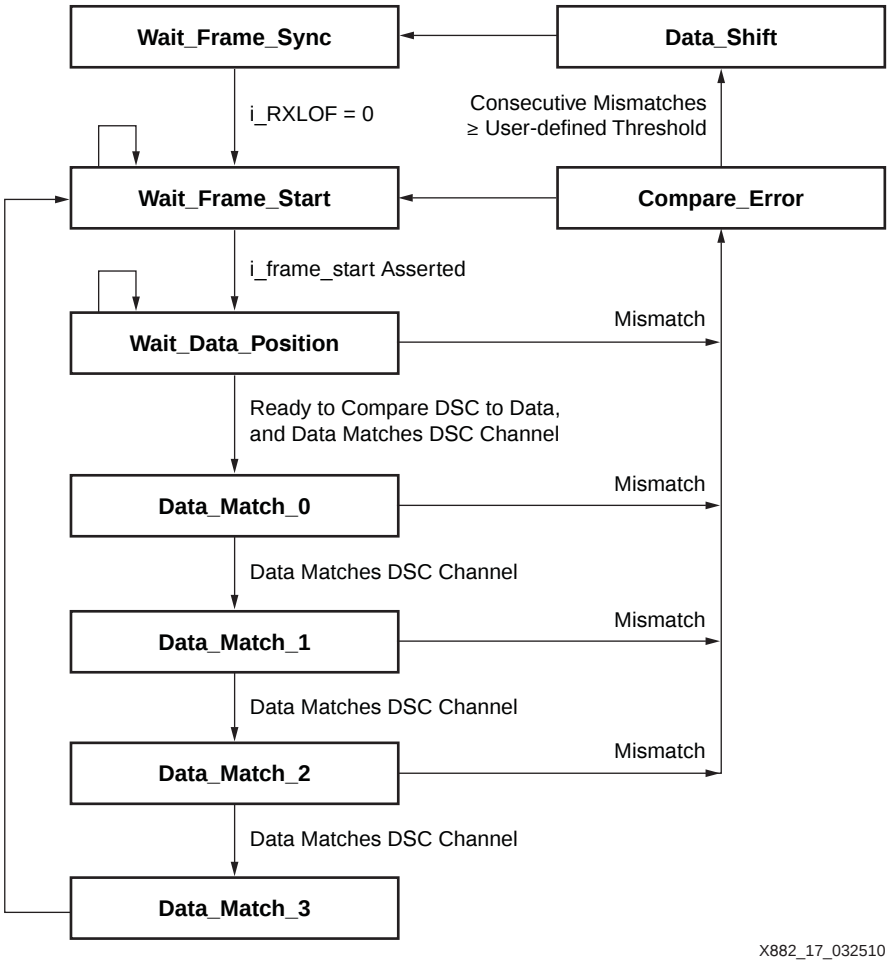

Figure 16: **Data Channel Deskew State Machine**

The data channel deskew state machine is contained in the `sfi5_rx_data_sync` module. The state machine does not attempt alignment until the frame synchronization algorithm reports that it has synchronized to the deskew channel (RXLOF deasserted). After the deskew channel is framed, the state machine waits for the beginning of the frame. The state machine for each channel then waits an additional number of cycles to arrive at the specific data fragment in the deskew frame for that channel. At this point, the data channel and deskew channel content are compared for four consecutive cycles (64-bit data fragment in deskew channel). All four of the comparisons match if the data channel is correctly aligned. If there is a mismatch in one of the four comparisons, a counter records the mismatch. If the number of consecutive mismatches reaches the user-defined threshold iv_MISMATCHES_2_UNLOCK, the barrel shifter setting of that data channel is incremented and the algorithm repeats the comparison process. The state machine starts at barrel shifter setting 0 and increments all the way to 63. If the receiver input skew specification is satisfied, the algorithm finds a match. If the algorithm does not find a match, it searches from 0 to 63 indefinitely. However, this indefinite search is interrupted when the receiver times out and reinitializes the whole link. This time-out functionality is part of the `sfi5_reset_rx` module.

Receiver Initialization

The receiver initialization process consists of resetting the GTX transceivers and waiting to finish the reset sequence, and for the PMA PLLs to lock. The state machine alone does not initiate the reset of the GTX transceiver. Each GTX is shared by the TX and RX interface. Thus, the TX interface initialization is responsible for initiating the GTX reset. The RX state machine waits for the GTX reset sequence to complete.

Time-out counters are provided in case the GTX transceivers never come out of reset, or the PMA PLLs do not lock. After timing-out, the algorithm restarts the RX initialization. The time-out allows the link to recover automatically when the physical link is broken for some period of time due to disconnection, one side of the link in a power-down state, etc. After the link is restored, the receiver restarts the initialization process, and the link recovers without user intervention or manual resets. The receiver initialization process is shown in Figure 17.

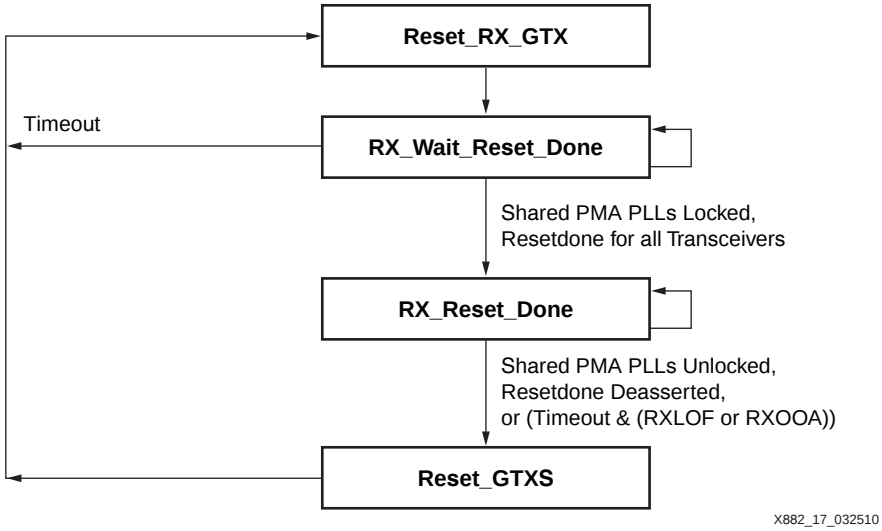

Figure 17: **Initialization Sequence of SFI-5 Receiver**

After the GTX reset sequence has completed, the state machine remains in the RX_Reset_Done state. However, if the PMA PLLs unlock, or if the RESETDONE output of any GTX transceiver is deasserted, the state machine resets all GTX transceivers and the initialization process starts over. The RX_Reset_Done state also has a time-out condition. If the frame synchronization and data deskew state machines cannot achieve frame alignment (RXLOF) or data alignment (RXOOA) after one million RXUSRCLK2 cycles (6 ms), the state machine declares an exceptional condition and restarts the initialization process after resetting the GTX transceivers. If it is not desired to have the TX or RX interfaces reset themselves automatically (for troubleshooting purposes or otherwise), the TX_Reset_Done and RX_Reset_Done next-state logic must be modified to remain in those states unconditionally.

SFI-5 Resource Utilization

The FPGA resources used by the stand-alone SFI-5 interface are shown in Table 3.

Table 3: **Resource Utilization**

Resource	Used	Percentage of Total	Comment
Slice	1,222	3.3	
Slice Register	3,226	1	
Slice LUT	2,717	1.8	
BUFG	2	9	txusrclk2, rxusrclk2
GTX Transceiver	17	85	16 data channels + 1 DSC channel
IOB	7	<1	TXREFCK (differential) TXREFCK_2 (differential) TXDCK (differential) RXS

SFI-5 Hardware Testbench

The reference design is hardware tested on the ML623 development board. The device is an XC6VLX240T in an FF1156 package. To test the interface, a hardware testbench consisting of a ChipScope™ Pro analyzer is wrapped around the interface. The analyzer reads link statistics from the SFI-5 interface, and allows control of the interface's optional settings from a virtual I/O (VIO) interface. The hierarchy of the hardware testbench is shown in Figure 18.

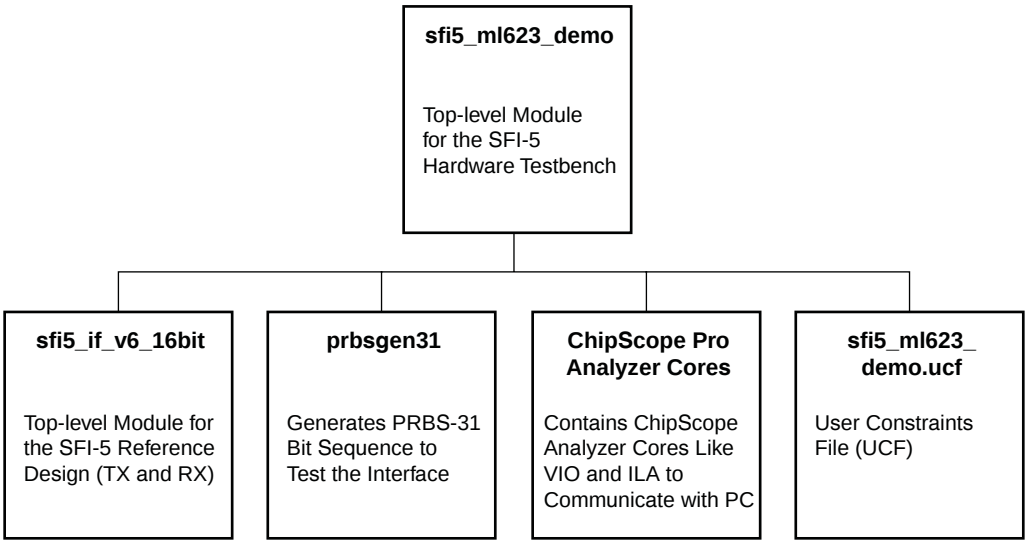

Figure 18: **Hierarchy of Hardware Testbench HDL Modules Comprising SFI-5 Interface**

A pseudo-random bit sequence 31 (PRBS31) is generated by the testbench to simulate the complexity of user data. The PRBS31 pattern is sent across the SFI-5 link. The error checking mechanism on the receive side is a comparison of the data channels with the deskew channel, which is already done by the SFI-5 receiver. The ChipScope analyzer VIO interface shows the mismatch counts of all 16 data channels as counted by the SFI-5 receiver. The VIO GUI is shown in Figure 19.

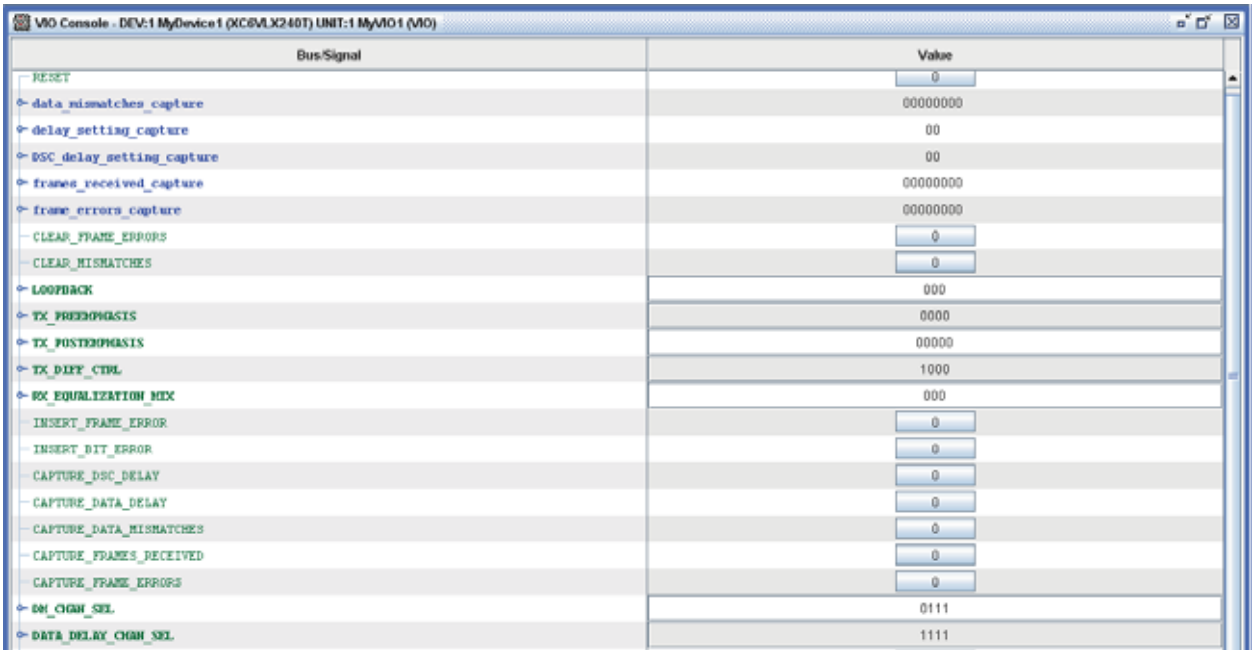

Figure 19: **ChipScope Pro Analyzer VIO GUI**

The user can start the test by pressing the RESET pushbutton followed by the CLEAR_FRAME_ERRORS and CLEAR_MISMATCHES pushbuttons (these two pushbuttons clear the error latch registers). This reset sequence resets the entire system, including the error detector latches.

The ability to insert errors is provided as a method to check the ability of the receiver to detect errors. When the INSERT_FAMRE_ERROR button is pressed, the frame header of the deskew channel is replaced by an incorrect header, resulting in a frame error. When the INSERT_BIT_ERROR button is pressed, the first 16 bits of the 64-bit fragment of data channel 15 are inverted for one clock cycle, resulting in datapath error.

The VIO GUI shows this information:

- Frames received (frames_received_capture)
  - The number of data frames received while the receiver is locked to the incoming data
  - The VIO latches this value based on the latch enable selected, e.g., CAPTURE_FRAMES_RECEIVED
- Frame errors and data mismatch errors (frame_error_capture and data_mismatches_capture)
  - The number of frame errors and data mismatch errors when compared to the deskew channel
  - The VIO latches this value based on the latch enables selected, e.g., CAPTURE_FRAME_ERRORS and CAPTURE_DATA_MISMATCHES, respectively
- The barrel shifter values of all data channels (multiplexed by the DATA_DELAY_CHAN_SEL tab, which selects between `0000` and `1111`) and the deskew channel
  - The VIO latches this value based on the latch enables selected, e.g., CAPTURE_DATA_DELAY and CAPTURE_DSC_DELAY, respectively

The remaining settings such as LOOPBACK, TX_DIFF_CTRL etc., are optional settings and use the default values identified in Table 2, page 4, unless the user selects other values using VIO.

Characterization

In this section, the SFI-5 interface is tested on several devices to verify these parameters of operation:

- Interface meets receiver eye mask requirement as specified in the OIF specification
- Deskew capability meets or exceeds the maximum skew compensation requirement
- Two SFI-5 interfaces on two different FPGAs pass traffic at bit error rate < 10^{-12} for an extended period of time.

Figure 20 shows that the SFI-5 transmitter meets the receiver eye mask requirements with several transmission media at 2.5 Gb/s. The longest medium is 10 inches of FR4 and 72 inches of coaxial cable terminated with SMA connectors. This 10-inch FR4 is considered in all cases.

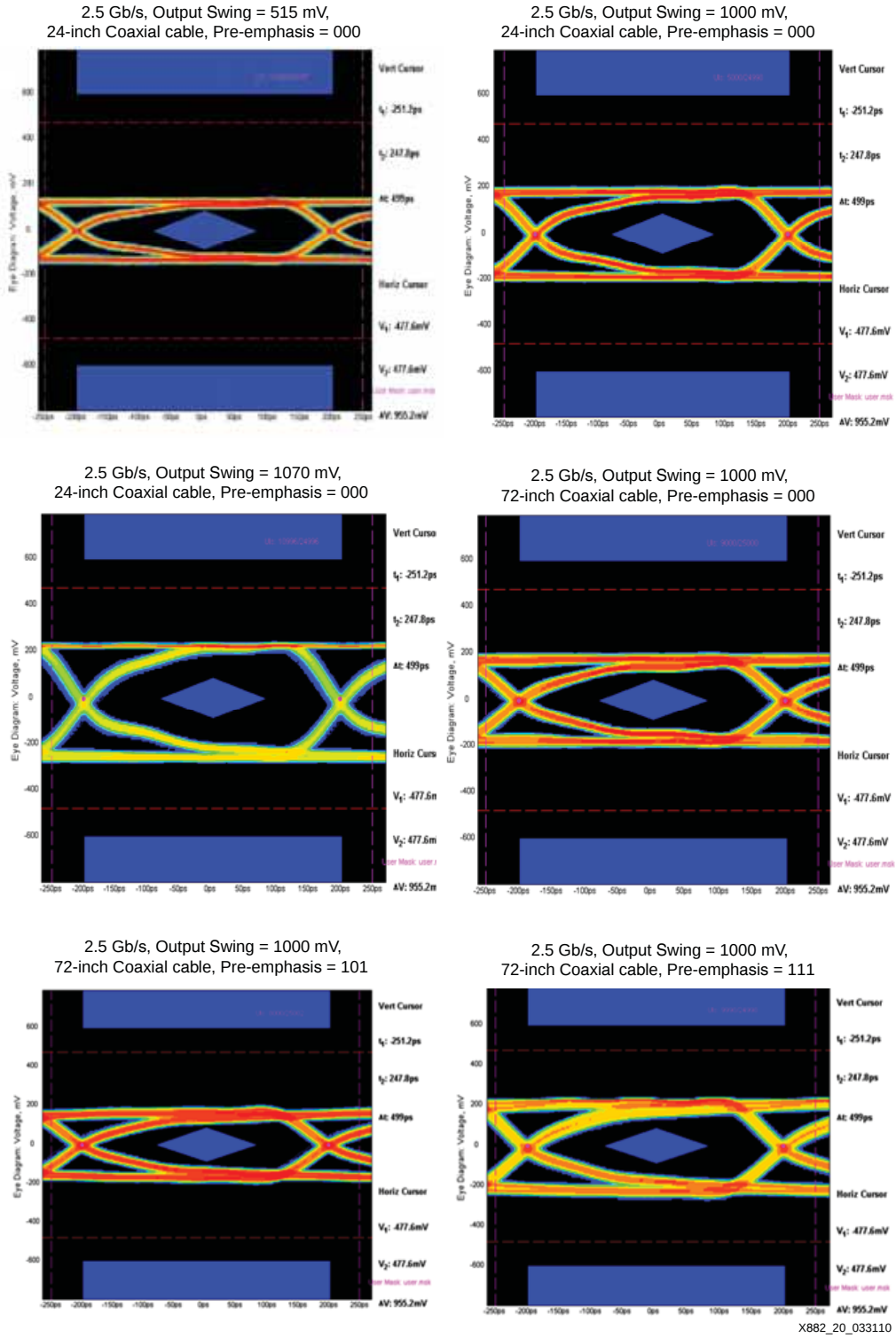

Figure 20: **Eye Masks for Various Media at 2.5 Gb/s**

Figure 21 shows the same cases as Figure 20, except that the data rate is 3.125 Gb/s, corresponding to a total interface speed of 50 Gb/s.

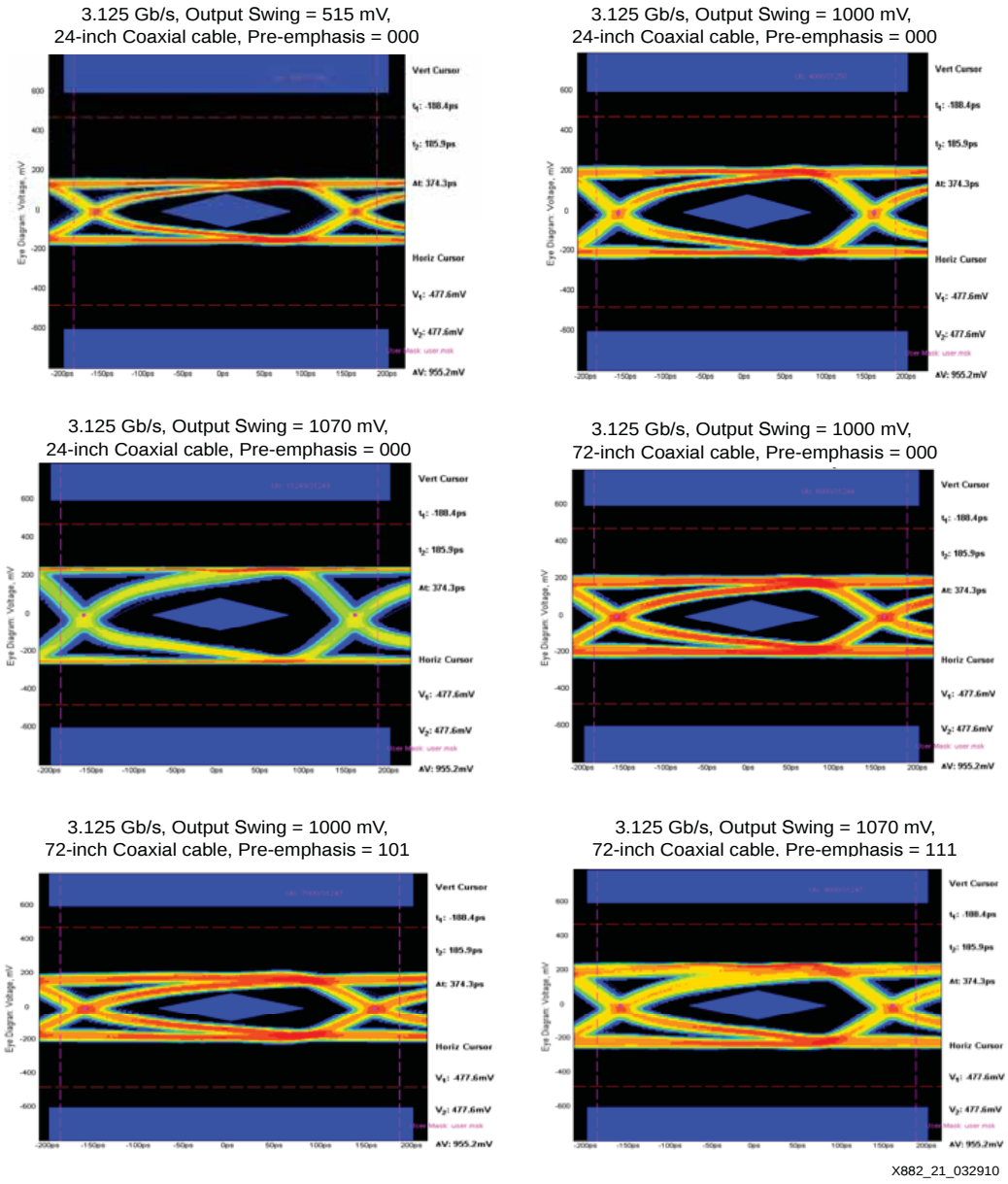

Figure 21: **Eye Masks for Various Media at 3.125 Gb/s**

The reference clock of the SFI-5 interface can come from an oscillator or from another device in the SFI-5 link. Every SFI-5 transmitter is responsible for providing a reference clock, and the receiver can optionally use that clock. This clock is provided on the TXDCK port. For a 40 Gb/s interface, the reference clock is 156.25 MHz. This is shown in Figure 22. For a 50 Gb/s interface, the reference clock is 195.3125 MHz.

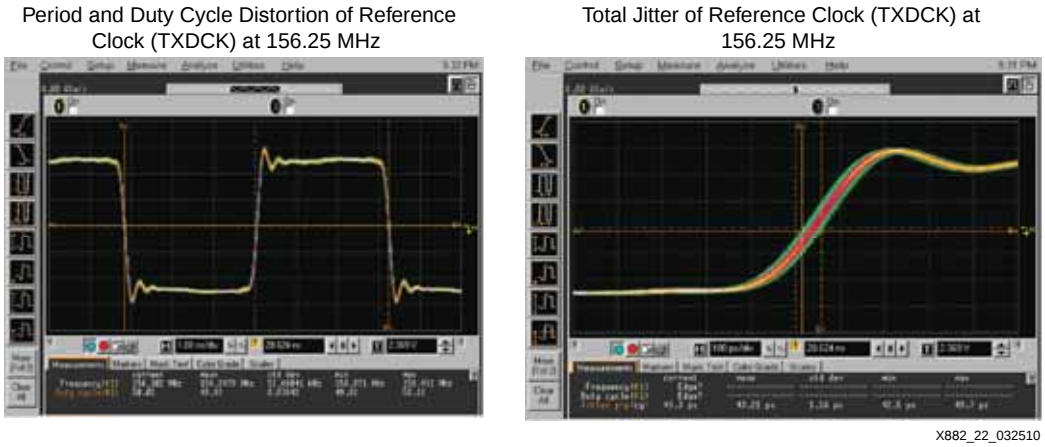

Figure 22: **Reference Clock (TXDCK) Provided by Transmitter for Optional Use by Receiver**

The deskewing capability of the receiver is tested in two XC6VLX240T devices with these conditions:

- 16 inches of extra skew only on data channel 2 (~6 UI of skew at the receiver)
- 2.5 Gb/s and 3.0 Gb/s operation
- Multiple resets in some cases

In all of these conditions, the device is looped back externally to itself. In another method of deskew testing, two independent XC6VLX240T devices exchange data via an SFI-5 interface without any loopback. This test case is detailed in Figure 23, which also shows the setup used for the SFI-5 system test.

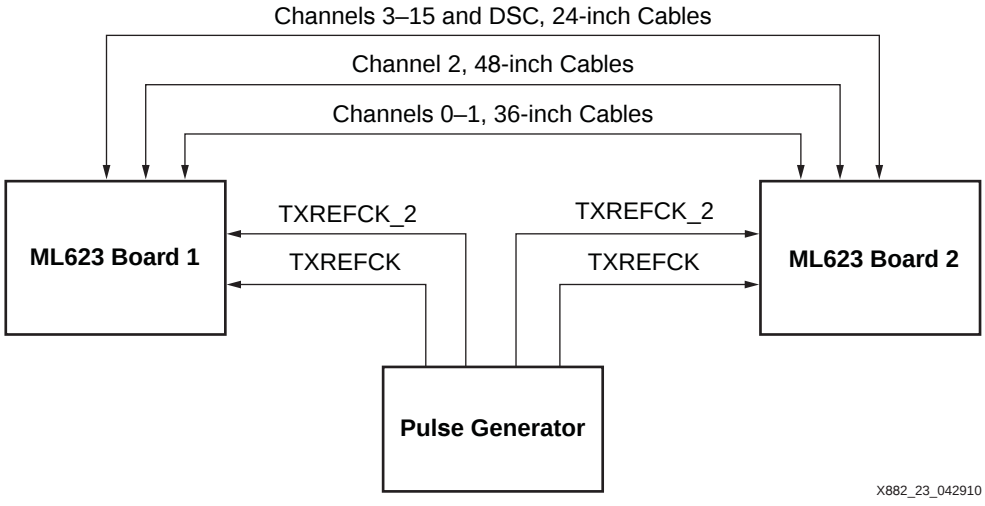

Figure 23: **System Test Setup**

The reference clocks provided to both boards were synchronous. Three different cable lengths were used, creating a maximum skew of 24 inches, which corresponds roughly to 8 UI. The optional interface settings were set to the default values given in Table 2, page 4.

A temperature forcing unit was applied to the FPGA on ML623 board 1. The unit was programmed to perform the temperature ramp shown in Figure 24, which takes approximately eight hours to complete. The system test was performed twice: once with ML623 board 1 as the DUT, and once with ML623 board 2 as the DUT. In both cases, there were zero mismatches, zero frame errors, and no alarms after the temperature ramp completed.

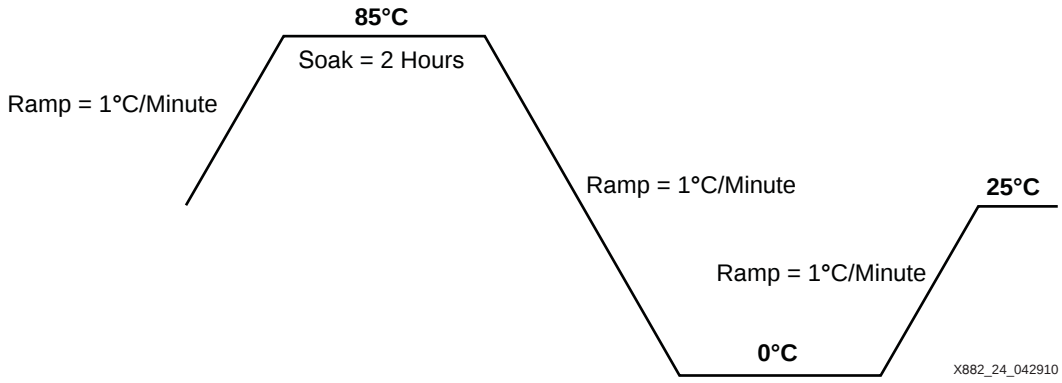

Figure 24: **Temperature Ramp for System Test**

The deskew results for the devices on boards 1 and 2 are shown in Figure 25 and Figure 26, respectively, at a data rate of 40 Gb/s (2.5 Gb/s x 16 channels).

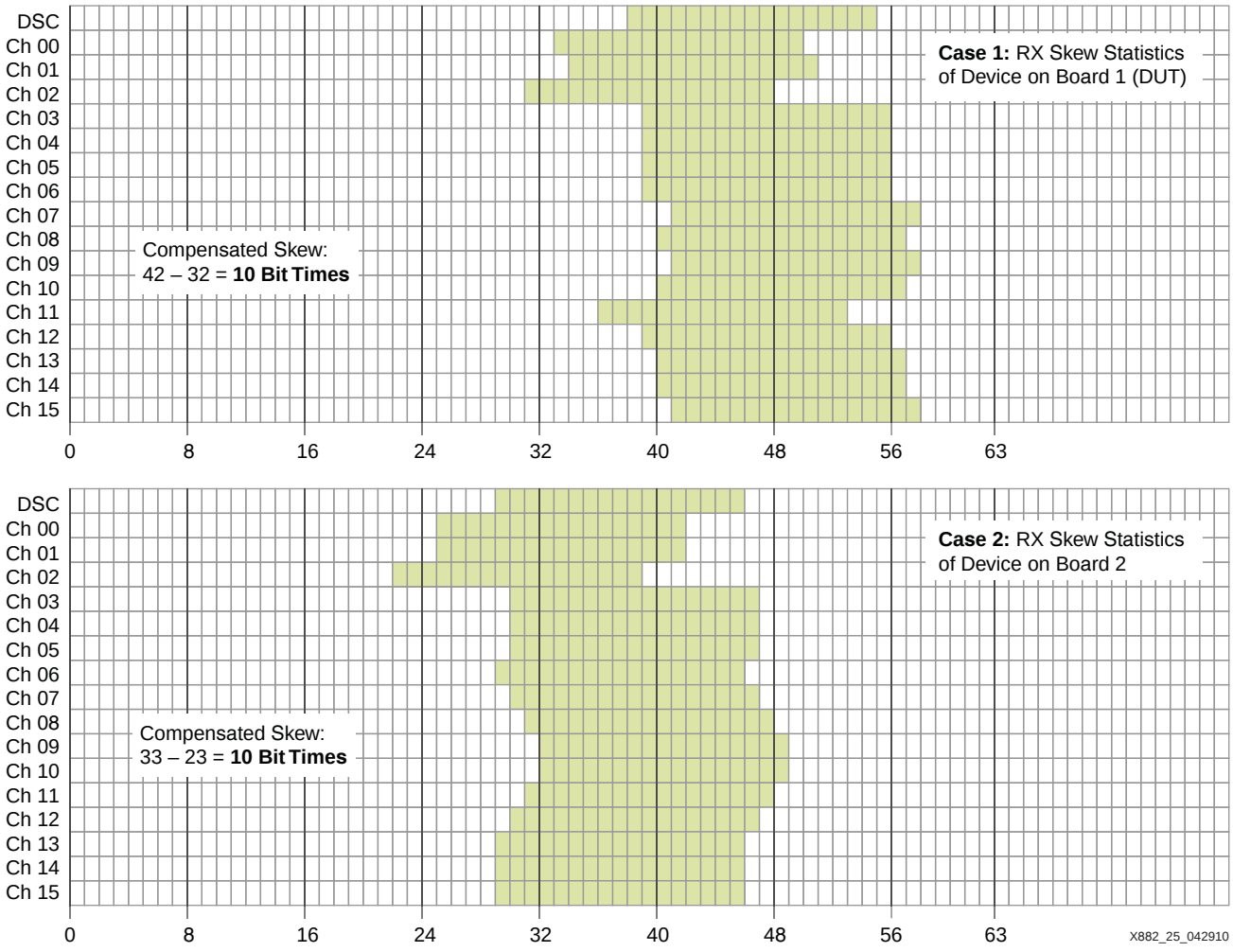

Figure 25: **Barrel Shifter Selections for All Channels Showing Skew Results for System Test**

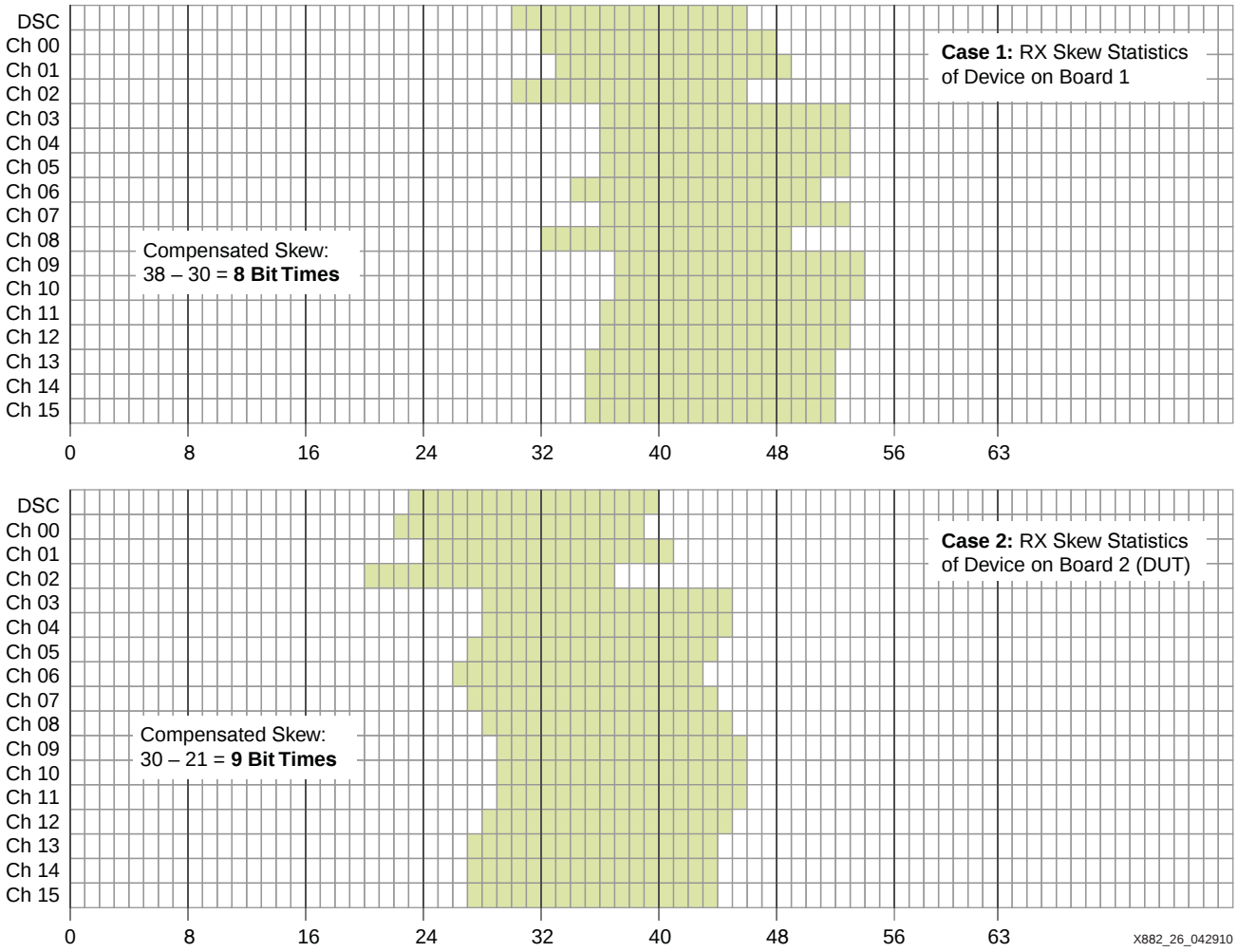

Figure 26: **Barrel Shifter Selections for All Channels Showing Skew Results for System Test**

Reference Design

The reference design for this application note can be found at:

https://secure.xilinx.com/webreg/clickthrough.do?cid=144052

The reference design matrix is shown in Table 4.

Table 4: **Reference Design Matrix**

Parameter	Description
General	
Developer Name	Xilinx
Target Devices (Stepping Level, ES, Production, Speed Grades)	Virtex-6 FPGAs
Source Code Provided	Y
Source Code Format	Verilog
Design Uses Code or IP from Existing Reference Design, Application Note, 3rd party, or CORE Generator™ Software	Y
Simulation	
Functional Simulation Performed	Y
Timing Simulation Performed	N

Table 4: **Reference Design Matrix *(Cont'd)***

Parameter	Description
Testbench Provided for Functional and Timing Simulations	Y
Testbench Format	Verilog
Simulator Software Used/Version (e.g., ISE® software, Mentor, Cadence, other)	Mentor Graphics, version 6.5c
SPICE/IBIS Simulations	N
Implementation	
Synthesis Software Tools and Version	XST (ISE software, version 12.1)
Implementation Software Tools and Version	ISE software, version 12.1
Static Timing Analysis Performed?	Y
Hardware Verification	
Hardware Verified?	Y
Hardware Platform Used for Verification	ML623

References

This application note uses the following references:

1. *SERDES Framer Interface Level 5 (SFI-5): Implementation Agreement for 40 Gb/s Interface for Physical Layer Devices*, Optical Internetworking Forum http://www.oiforum.com/public/documents/OIF-SFI5-01.0.pdf
2. DS152, *Virtex-6 FPGA Data Sheet: DC and Switching Characteristics*.
3. UG366, *Virtex-6 FPGA GTX Transceivers User Guide*.

Revision History

The following table shows the revision history for this document.

Date	Version	Description of Revisions
04/09/10	1.0	Initial Xilinx release.
05/10/10	1.1	Added description of deskewing to Characterization, including Figure 23 through Figure 26.

Notice of Disclaimer