

Xilinx Answer 56616

Debugging Guide for 7-Series Integrated PCI Express Block Link Training Issues

Important Note: This downloadable PDF of an Answer Record is provided to enhance its usability and readability. It is important to note that Answer Records are Web-based content that are frequently updated as new information becomes available. You are reminded to visit the Xilinx Technical Support Website and review ([Xilinx Answer 56616](#)) for the latest version of this Answer.

This answer record has screen shots of tables and figures from other documents. The guidelines provided may have changed in the latest release of those documents. The readers are advised to refer to the latest release of the corresponding documents.

Some GTX/GTP (Gigabit Transceiver) settings can be tuned to correct link training issues. Some guidance on which parameters should be tuned is provided in this document. In general, the default settings should work across all boards and systems. In the case where non-default value works, please contact Xilinx Technical Support before permanently using those parameter values in your design.

Introduction

This document describes techniques to debug link training issues with 7-Series Integrated PCI Express Block. A complete list of signals to capture in ChipScope Pro/Vivado ILA when debugging link training issues has been provided. Screen captures of the signal waveforms illustrate how to analyze those signals and establish theories on potential reasons causing the problem. One of the main reasons behind running into link training issues is due to Signal Integrity (SI) issues on the board. A general guideline of things to check has been provided to debug probable issue due to SI.

Link training issues do not entirely depend on the PCIe Core. They are equally a function of the board and how the system is connected up. Therefore, it is important to make sure that all the factors affecting the signal integrity on the board should be thoroughly checked (e.g. reference clock quality, voltage signal level etc.). There are few transceiver parameters that a user could tune to suit their system. These parameters will be discussed in this document.

Link Training Overview

After FPGA configuration, the two connected devices go through the link training process. The Link Training and Status State Machine (LTSSM) defines this process. Figure 1 shows the different states of the LTSSM. The main states to consider while debugging link training issues are DETECT, POLLING, CONFIGURATION, and L0. Detailed descriptions of the LTSSM states are found in section 4.2.5 of the PCI Express Base Specification v2.1.

In the DETECT state, each lane performs receiver detect to determine if a link partner is present on that lane. Lanes that do not detect a link partner are not used and the FPGA drives an electrical idle on these lanes. The second state entered during link training is the POLLING state. This is the first state where the link partners exchange TS1 and TS2 ordered sets. During this state, bit symbol lock and lane polarity are established.

The CONFIGURATION state follows POLLING. During CONFIGURATION, link and lane numbers are exchanged through TS1 and TS2 ordered sets and the link width are established. Once CONFIGURATION completes, the next state is L0.

The L0 state is the normal working state where data is transferred on the link. The core output signal user_Ink_up is asserted during this state. Note that user_Ink_up does not assert immediately upon entering L0, but asserts after the data link layer achieves the DL.ACTIVE state, meaning the initial flow control credits have been exchanged.

During the link training process, the following are discovered and determined:

- Lane polarity
- Link data rate
- Link and lane numbers
- Link width
- Lane reversal

In overall, link training process does the following:

- Link data rate negotiation
- Bit lock per lane
- Lane polarity
- Symbol lock per lane
- Lane ordering within a link
- Link width negotiation
- Lane-to-Lane de-skew within a multi-lane link

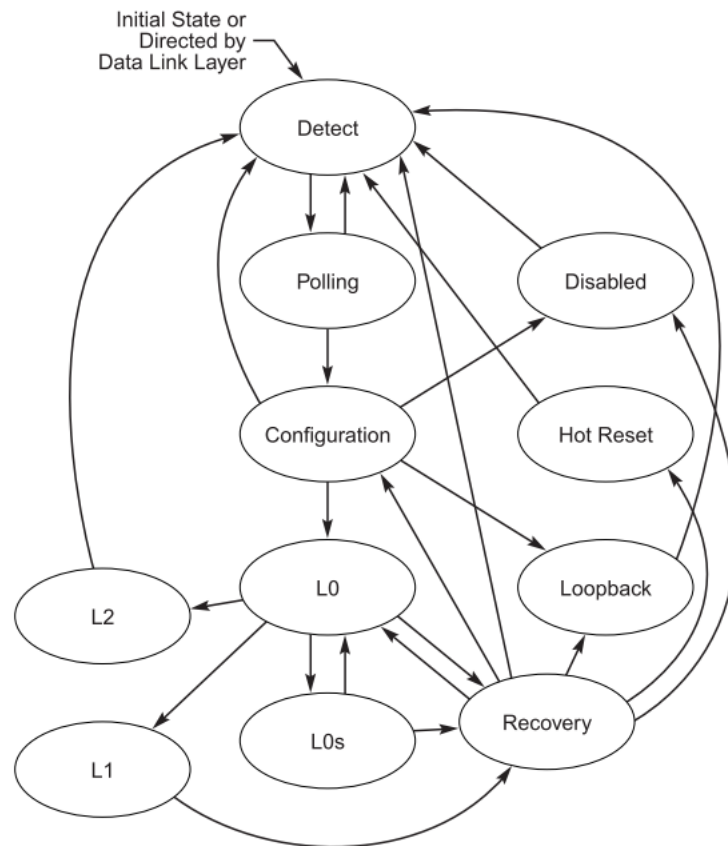


Figure 1 - Link Training Status and State Machine (LTSSM)

Ordered Sets

During the link training process, the physical layer communicates by exchanging TS1 and TS2 ordered sets. Ordered sets are packets that originate and terminate in the physical layer.

There are four different types of ordered sets. Ordered sets are not scrambled, so they are easily viewed using ChipScope Pro/Vivado ILA or in simulation at the GT TX/RX interface. The four different types of ordered sets are Training Sequence ordered sets (TS1s and TS2s), Electrical Idle ordered sets (EIOS), Skip ordered sets (SKP), and

Fast Training Sequence ordered sets (FTS). Link training uses TS1 and TS2 ordered sets to exchange information to establish the link. Occasionally, a SKP ordered set is transmitted during link training, so it is necessary to distinguish the difference.

Training Sequence 1 and 2 (TS1 and Ts2):

- TS1 and TS2 ordered sets are comprised of 16 symbols.
- The first symbol is COM, which is the K28.5 character. The receiver uses this character to achieve Bit Lock and Symbol Lock.
- TS1 and TS2 ordered sets contain information regarding link number, lane number, N_FTS, training control (such as hot reset, disable link, loopback etc.). For detail information on TS1 and TS2 refer to section 4.2.4 of the PCI Express Base Specification v2.1.
- A TS1 is identified by the D10.2 (4Ah) data character or a D21.5 (B5h) on a polarity reversed link.
- A TS2 is identified by the D5.2 (45h) data character or a D26.5 (BAh) on a polarity reversed link.

Table 1 shows the description for each symbol in TS1 ordered set. TS1s and TS2s are mostly the same except for the following:

- Symbols 6-15 which denote the TS2 identifier for a TS2 ordered set.
- TS2 symbol 4, bit 6, can be used to determine Link Upconfigure Capability/Selectable De-emphasis on top of the "Autonomous Change" as in TS1.
- TS1 Symbol 5, bit 4, is required to be implemented for GEN2 speed while it is reserved in TS2 symbol.

For more details on TS1 and TS2, check section 4.2.4.1 of the PCI Express Base Specification Rev2.1.

Table 1 - TS1 Ordered Set

Symbol Number	Encoded Values	Description
0	K28.5	COM for Symbol alignment
1	D0.0 - D31.7, K23.7	Link Number within component
2	D0.0 - D31.0, K23.7	Lane Number within Port
3	D0.0 - D31.7	N_FTS. This is the number of Fast Training Sequences required by the Receiver to obtain reliable bit and Symbol lock.

Symbol Number	Encoded Values	Description
4	D2.0, D2.2, D2.4, D2.6, D6.0, D6.2, D6.4, D6.6	<p>Data Rate Identifier</p> <p>Bit 0 – Reserved, set to 0b.</p> <p>Bit 1 – When set to 1b, indicates 2.5 GT/s data rate supported.</p> <p>Bit 2 – When set to 1b, indicates 5.0 GT/s data rate supported. Devices that advertise the 5 GT/s data rate must also advertise support for the 2.5 GT/s data rate (i.e., set Bit 1 to 1b).</p> <p>Bit 3:5 – Reserved, must be set to 0b.</p> <p>Bit 6 (Autonomous Change) –</p> <p>Downstream component: Autonomous Change/Selectable De-emphasis: When set to 1b in Configuration state and LinkUp = 1b, indicates that the speed or Link width change initiated by the Downstream component is not caused by a Link reliability issue.</p> <p>In Recovery state, this bit indicates the de-emphasis preference of the Downstream component.</p> <p>In Polling.Active substate, this bit specifies the de-emphasis level the Upstream component must operate in if it enters Polling.Compliance and operates in 5.0 GT/s data rate.</p> <p>In Configuration.Linkwidth.Start substate with LinkUp = 0b and in the Loopback.Entry substate, this bit specifies the de-emphasis level the Upstream component must operate in if it enters Loopback state (from Configuration) and operates in 5.0 GT/s data rate. For de-emphasis, a value of 1b indicates -3.5 dB de-emphasis and a value of 0b indicates -6 dB de-emphasis.</p> <p>This bit is reserved in all other states for a Downstream component.</p> <p>Upstream component: In Polling.Active, Configuration.Linkwidth.Start, and Loopback.Entry substates, this bit specifies the de-emphasis level the Downstream component must operate in 5.0 GT/s data rate if it enters Polling.Compliance and Loopback states, respectively. A value of 1b indicates -3.5 dB de-emphasis and a value of 0b indicates -6 dB de-emphasis.</p> <p>This bit is reserved for all other states.</p> <p>Bit 7 (speed_change) – When set to 1b, indicates a request to change the speed of operation. This bit can be set to 1b only during Recovery.RcvrLock state.</p> <p>All Lanes under the control of a common LTSSM must transmit the same value in this Symbol. Transmitters must advertise all supported data rates in Polling.Active and Configuration.LinkWidth.Start substates, including data rates they do not intend to operate on.</p>

Symbol Number	Encoded Values	Description
5	D0.0, D1.0, D2.0, D4.0, D8.0, D16.0, D20.0	<p>Training Control</p> <p><u>Bit 0 – Hot Reset</u> Bit 0 = 0b, De-assert Bit 0 = 1b, Assert</p> <p><u>Bit 1 – Disable Link</u> Bit 1 = 0b, De-assert Bit 1 = 1b, Assert</p> <p><u>Bit 2 – Loopback</u> Bit 2 = 0b, De-assert Bit 2 = 1b, Assert</p> <p><u>Bit 3 – Disable Scrambling</u> Bit 3 = 0b, De-assert Bit 3 = 1b, Assert</p> <p><u>Bit 4 – Compliance Receive</u> Bit 4 = 0b, De-assert Bit 4 = 1b, Assert</p> <p>Components that support 5 GT/s data rate must implement this bit as specified. Components that support only 2.5 GT/s data rate may optionally implement this bit as a Receiver. If not implemented for components that support only 2.5 GT/s data rate, this bit will be reserved and must behave as if the component received a 0b in this bit position.</p> <p><u>Bit 5:7 – Reserved</u> Set to 0b</p>
6 – 15	D10.2	TS1 Identifier

Electrical Idle Ordered Set

- The Electrical Idle Ordered-Set consists of four symbols- COM, IDL, IDL, IDL = BC, 7C, 7C, 7C.
- The transmitter sends out the electrical idle ordered set before driving electrical idle.
- After receiving the electrical idle ordered set, the link partner prepares the link for transition to electrical idle.

SKP Ordered Set

- Consists of four symbols - COM, SKP, SKP, SKP = BC, 1C, 1C, 1C
- SKP ordered set is transmitted at regular intervals from transmitter to the receiver.
- Used for clock tolerance compensation

FTS Ordered Set

- Also consists of four symbols - COM, FTS, FTS, FTS = BC, 3C, 3C, 3C
- A transmitter sends FTS ordered sets
- The number of required ordered sets is agreed during link training and initialization

Link Training Failure Types and Debug Flow

The link training issue could be due to a multitude of things, with some happening at the start of the link training and some during link training (e.g., LTSSM getting stuck in Polling or Configuration states). Some other issues could be a link going into recovery right after it has linked up, a link going into recovery after the link has been working for a while, and so on.

Figure 2 categorizes link training failures into five types. This covers only the most common issues. There could be other issues that are seen in a system during link training. Debugging guidelines and approaches described in this document should still be applicable to debug such issues.

Figure 2 shows the initial debug flow to identify the link training failure type. This can be done by probing rxstatus, pl_ltssm_state, user_lnk_up, pl_initial_link_width, pl_sel_lnk_width and pl_sel_lnk_rate signals in ChipScope/Vivado ILA.

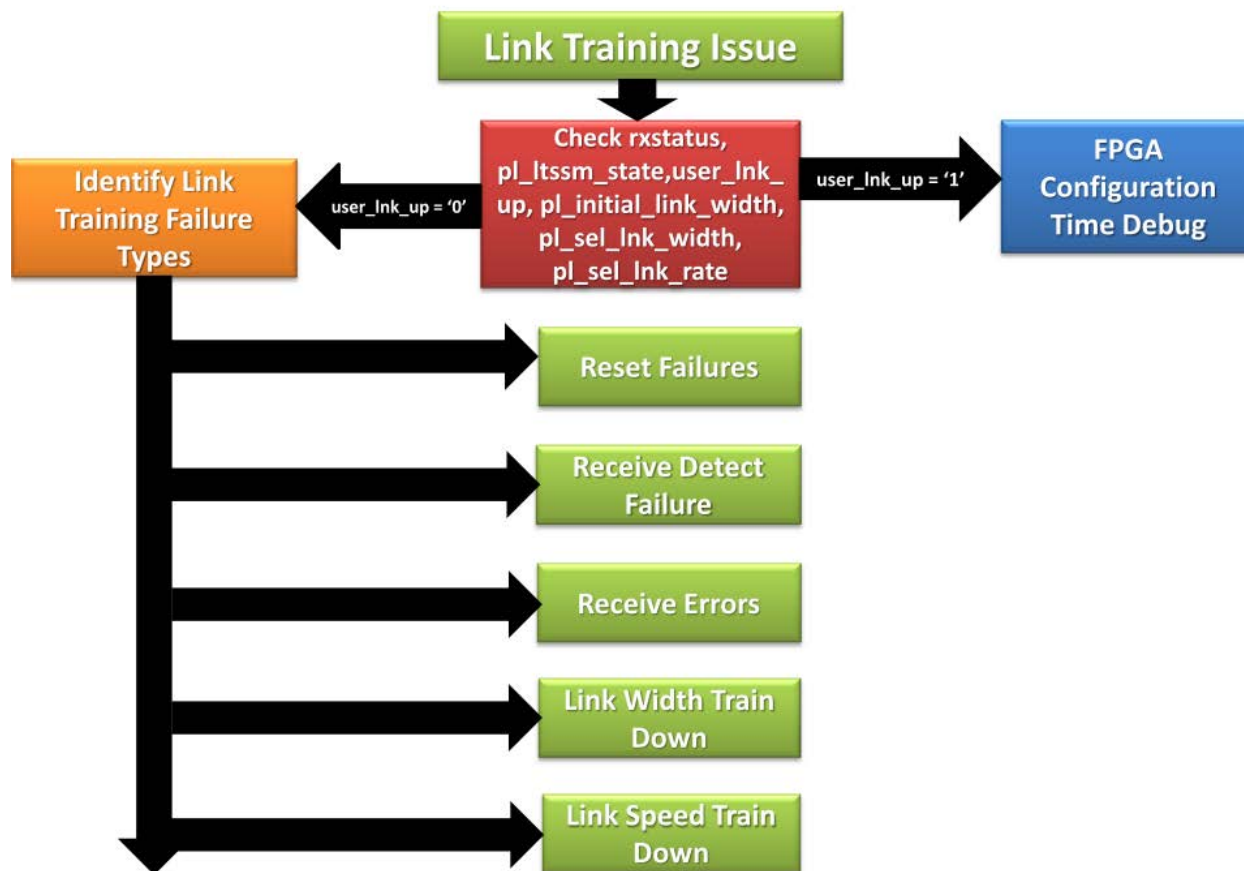


Figure 2 - Identifying Link Training Failure Type

The five major link training failure types are as follows:

1. Reset failure
2. Receiver detect failure
3. Receive errors
4. Link width train down
5. Link speed train down

The first thing to check is whether user_lnk_up is asserted or not. If user_lnk_up is asserted, but the core is not detected by the host, go to the “**FPGA Configuration Time Debug**” section to proceed with further investigation. If user_lnk_up is not asserted, then proceed investigating by identifying the failure type.

Most of the Link training problems are due to board signal integrity problems or incorrect GT usage. The board must meet both the electrical requirements set forth by the GT user guide and also the PCI Express Base Specification. This will be discussed in more detail in the latter part of this document.

Receiver Detect Failures

If “pl_ltssm_state[5:0]” is stuck in 0,1,2 or 3, the link has probably run into a Receiver Detect Failure.

Receiver detect is the first state of the Link Training Status State Machine (LTSSM). The PCI Express specification includes a feature that allows the transmitter on a given link to detect if a receiver is present. The decision if a receiver is

present is based on the rise time of TXP/TXN. Figure 3 shows the circuit model used for receiver detection. The GTX/GTH transceiver must be in the P1 power down state to perform receiver detection. Receiver detection requires a 75 nF to 200 nF external coupling capacitor between the transmitter and receiver, and the receiver must be terminated to GND. The receiver detection sequence starts with the assertion of TXDETECTRX. In response, the receiver detection logic drives TXN and TXP to $(V_{DD} - V_{SWING}/2)$ and then releases them. After a programmable interval, the levels of TXN and TXP are compared with a threshold voltage. At the end of the sequence, the receiver detection status is presented on RXSTATUS when PHYSTATUS is asserted High for one cycle.

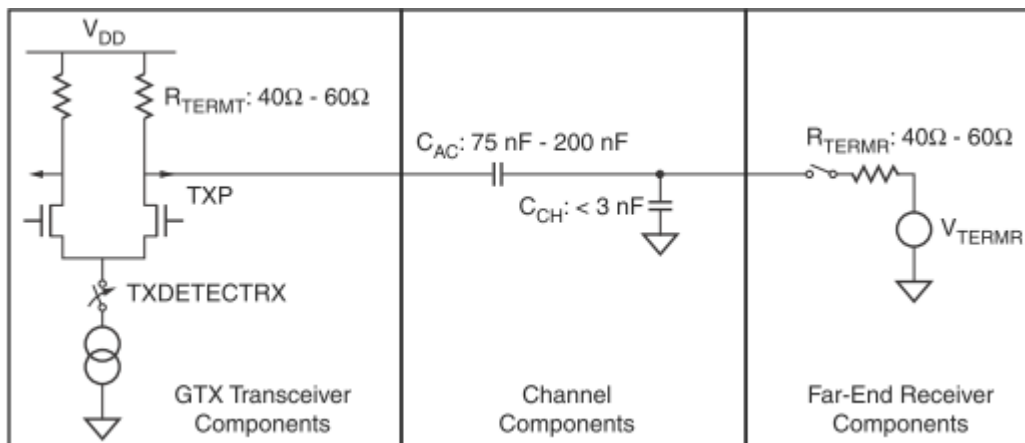


Figure 3 - Receiver Detect Circuit Model

To determine whether the link partner has passed the detect state, check the transceiver's RXELECIDLE output. If this signal is asserted indefinitely, the link partner did not detect the PCI Express core. The transmitter on each side of the link performs receiver detect once the LTSSM moves into the DETECT.ACTIVE state.

Figure 4 shows the general debug flow for issues related to Receiver Detect. After capturing “**Signal Set-2, Detect State**”, “**Signal Set-3, Reset**” and “**Signal Set-4, Clocking**”, make sure the signals are toggling correctly as described in “**Signal Set-1, LTSSM**” section.

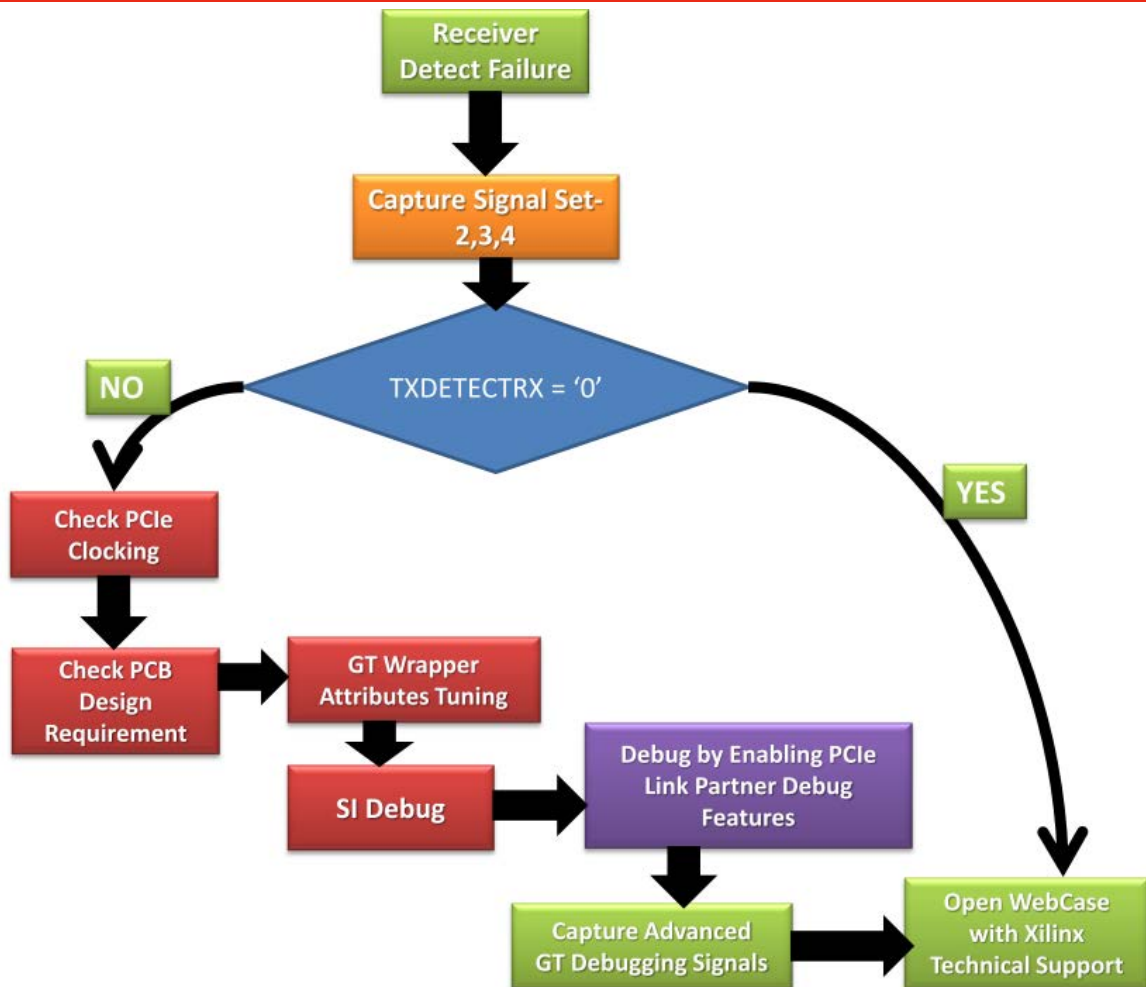


Figure 4 - Receiver Detect Failure Debug Flow Chart

Receive Errors

When the incoming data is corrupted due to crosstalk or other forms of interference on the link, the RXSTATUS signal would normally indicate 8B/10B errors or disparity errors. In such a scenario, follow the debug flow as shown in Figure 5.

000: Data Received OK
001: One Skip Symbol (SKP) added
010: One SKP removed
011: Receiver detected
100: 8B/10B decode error
101: Elastic Buffer overflow
110: Elastic Buffer underflow
111: Receive disparity error

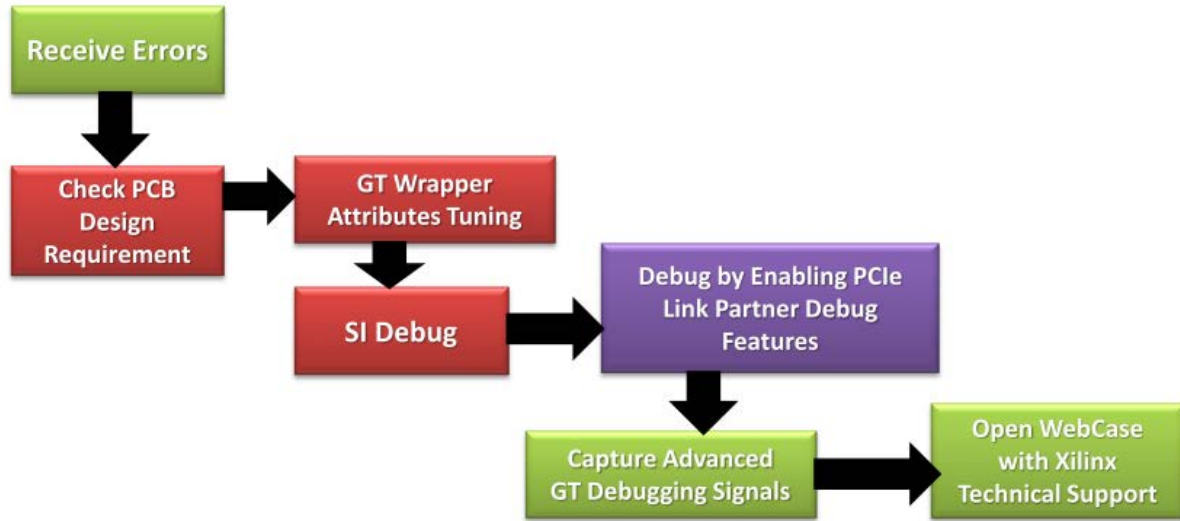


Figure 5 - Receive Errors Debug Flow Chart

Reset Failures

During debug, make sure the reset signals are asserted and de-asserted correctly. Capture “**Signal Set-3, Reset**” signals for further analysis. Users should make sure a device at either end of the link is not stuck in reset.

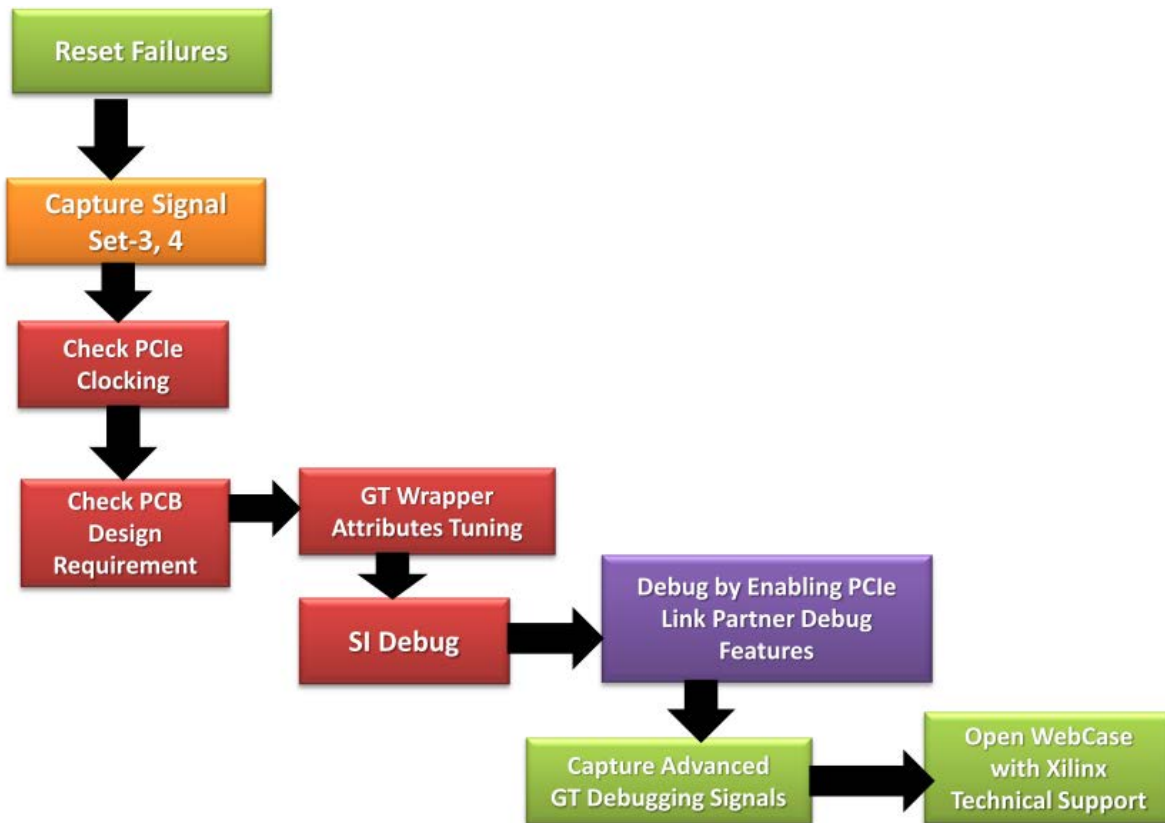


Figure 6 - Reset Failure Debug Flow Chart

Link Width Train Down

Whether the link is training to the correct link width or not can be checked by probing 'pl_initial_link_width' and 'pl_sel_lnk_width' signals in Chipscope/ Vivado ILA.

Multi-lane designs can introduce crosstalk and noise on the serial lanes. When having link training issues where the link is training down to a lower link width, first try isolating the upper lanes and then force the link to attempt to train as an x1. For add-in cards, this can be done by using any interposer or by placing scotch tape on the upper lane pins on the connector, as shown in Figure 7 and Figure 8.



Figure 7 - x8 lanes down to x4 lanes



Figure 8 - x8 lanes down to x1 lane

Figure 9 shows the debug flow chart for debugging the 'Link Width Train Down' issue. TS1 and TS2 analysis for checking whether the link width train down is initiated by the endpoint or the host is discussed in the "**CONFIGURATION State**" section.

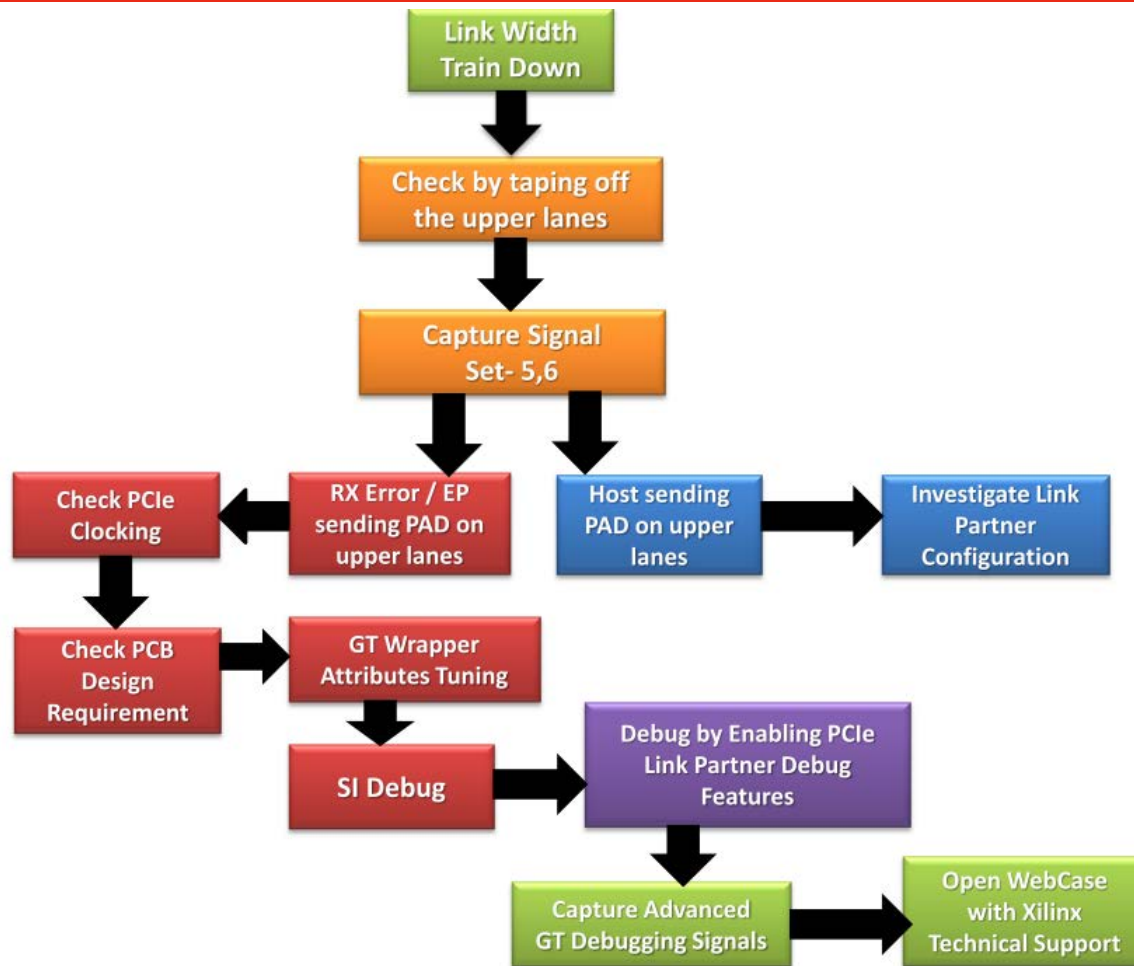


Figure 9 - Link Width Train down Debug Flow Chart

Link Speed Train Down

Whether the link is training down to a lower speed e.g. Gen2 to Gen1, it can be checked by reading the configuration registers or by probing `pl_sel_lnk_rate`. Probe `pl_link_partner_gen2_supported` (Figure 12) signal. If this signal indicates the link partner does not support gen2, investigate the link partner configuration and make sure the host is capable of gen2.

If `pl_link_partner_gen2_supported` is asserted but still the link is training down to a lower speed, capture “**Signal Set-5, GT RX**” and “**Signal Set-6, GT TX**” signals and analyze TS1s and TS2s to investigate whether it is the endpoint or the link partner that is not correctly following the required protocol to link train to Gen2 speed.

The main technique in debugging PCIe link training issue is try to figure out which LTSSM state the core is stuck at. After this, analyze the ordered sets being exchanged in this particular LTSSM state and compare with the specification. This will help narrow down whether it is the host or the endpoint that is not following the correct link training protocol.

In this section, a step-by-step method to narrow down probable causes in debugging ‘Link Speed Train Down’ issue is presented. In the description provided below, first the statement in the protocol is presented and based on that what signals to capture in Chipscope/ Vivado ILA and what to check for are discussed.

Protocol: Suppose a Link connects the two 5.0GT/s capable components, A and B. The Link comes up to L0 state in 2.5 GT/s speed. Component A decides to change the speed to 5.0 GT/s, sets the `directed_speed_change` variable to 1b and enters `Recovery.RcvrLock` from L0. Component A sends TS1 Ordered Sets with `speed_change` bit set to 1b and

advertises the entire data rate it is capable of. Component B sees the first TS1 in L0 state and enters Recovery.RcvrLock state. Initially, component B sends TS1s with speed_change set to 0b. Component B will start sending the speed_change indication in its TS1 after it receives eight consecutive TS1 Ordered Sets from component A and advertises all the data rates it can support.

Debug Action: Trigger on Recovery.RcvrLock. Trigger at the middle of the buffer. Look at the gt_rx_data, there should be 8 TS1 coming in with speed_change bit set to '1'. Also check bit-2, to see if 5GT/s is supported or not. Look at the gt_tx_data to see if the core is sending 8 TS1 with the speed_change bit set to '1'. This might not be the case immediately after the trigger point. TS1 should have speed change bit set to '1' after 8 consecutive TS1 in gt_rx_data with speed_change bit set to '1'. Also, check the corresponding bit-2, to find out the data rate it supports.

Protocol: Component B will enter Recovery.RcvrCfg from where it will enter Recovery.Speed.

Debug Action: Trigger on Recovery.RcvrCfg to check if the speed change process is in progress on or not. Also trigger on Recovery.Speed.

Protocol: Component A will wait for eight consecutive TS1/TS2 with speed_change bit set from component B before moving to Recovery.RcvrCfg and on to Recovery.Speed. Both component A and component B enter Recovery.Speed and record 5.0 GT/s as the maximum speed they can operate with. The directed_speed_change variable will be reset to 0b when in Recovery.Speed. When they enter Recovery.RcvrLock from Recovery.Speed, they will operate in than 5.0 GT/s speed and send TS1s with speed_change set to 0b.

Debug Action: Check if TS1 has speed_change bit set to 0.

Protocol: If both sides work well at 5.0 GT/s, they will continue on to Recovery.RcvrCfg and enter L0 through Recovery.Idle at 5.0 GT/s speed. However, if component B fails to achieve Symbol lock, it will timeout in Recovery.RcvrLock and enters Recovery.Speed.

Debug Action: Trigger in Recovery.Speed. Check if it is going to this state directly from Recovery.RcvrLock state. If this is the case, then Component B has failed to achieve Symbol lock.

From Recovery.RcvrLock, the LTSSM can go to following states:

1. Recovery.RcvrCfg : Good scenario...the link is training to 5GT/s.
2. Recovery.Speed: Link not able to operate at speed greater than 2.5GT/s.
3. Configuration: Core is not receiving the expected TS1 and TS2. Fault at the host.
4. Detect : Link is lost. Start again.

Good Scenario:

Recovery.RcvrLock -> Recovery.RcvrCfg -> Recovery.Speed

Bad Scenario:

Recovery.RcvrLock -> Recovery.Speed
Recovery.RcvrLock-> Configuration
Recovery.RcvrLock -> Detect

Figure 10 – Link Speed Negotiation Bad/Good Scenarios

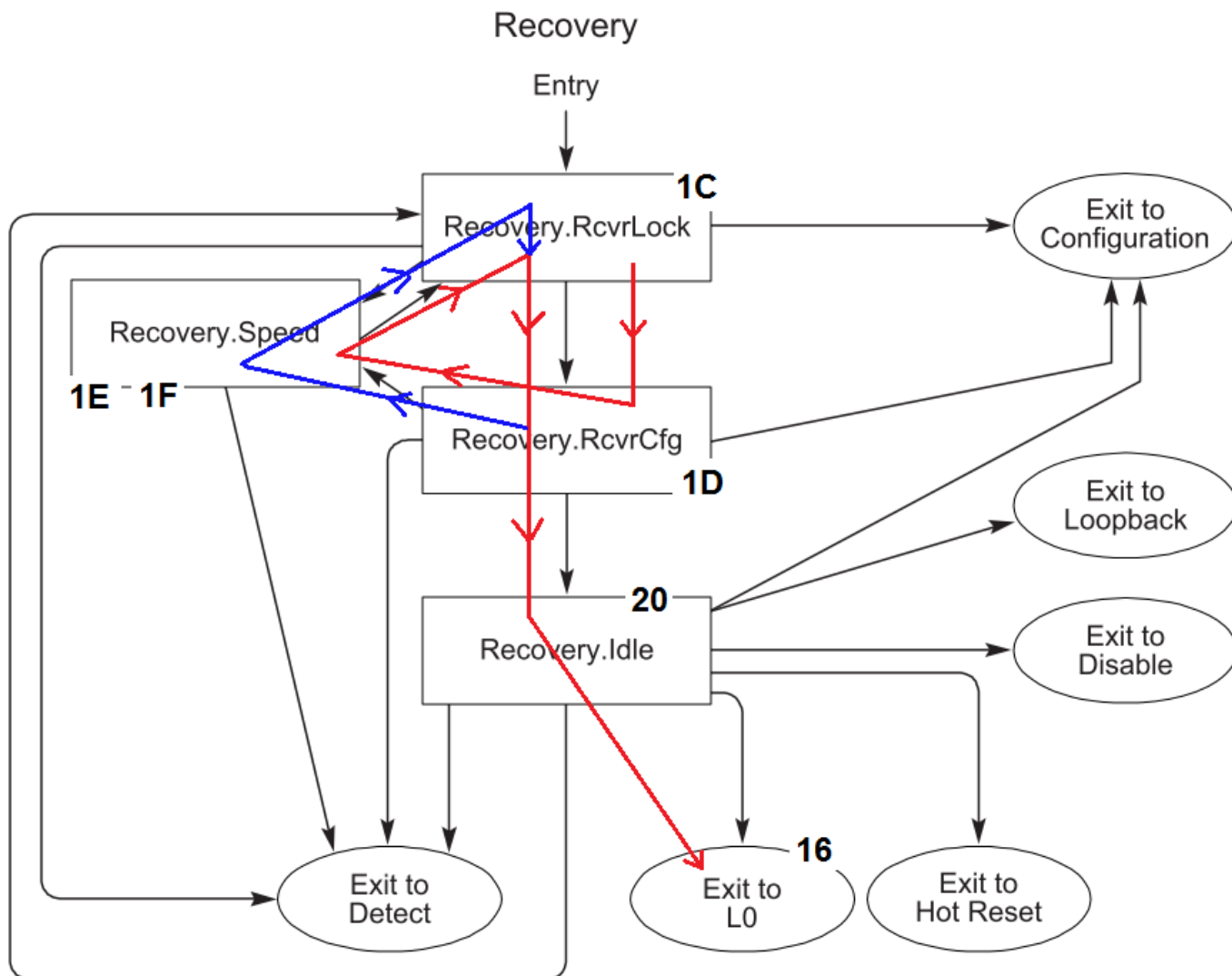


Figure 11 – Recovery State Link Speed Negotiation

Figure 11 shows the Recovery Substate Machine. The normal working scenario is the red line. It starts from 1C -> 1D -> 1E/1F -> 1C -> 1D -> 20 -> 16 (Refer **Figure 13 - LTSSM States**). However, during incorrect transition to Gen1 speed, it may take the route of the blue line after it re-enters 1D after following the red line. In such case, probe rxeleidle signal ("**Signal Set-5, GT RX**") and make sure it is not getting asserted erroneously.

For a multilane core, there will be the same number of rxeleidle signals as the number of lanes. Trigger on assertion of each rxeleidle signal. If it does trigger on one of the lanes but rxeleidle signals for other lanes are still de-asserted, it is an indication of potential issue with electrical idle detect threshold. Try different values of the RXOOB_CFG attribute as described in "**GTX/GTP Wrapper Settings**" section. If the issue is still not resolved, follow the debug flow chart shown in Figure 5.

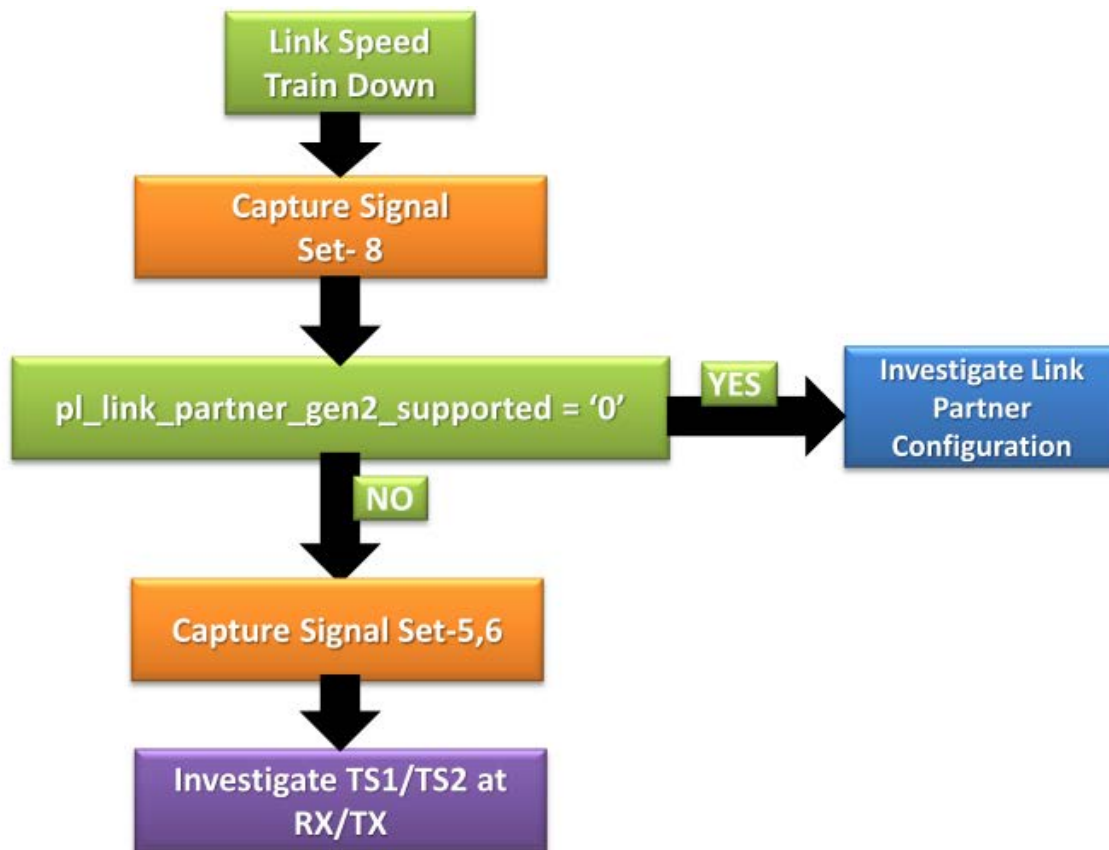


Figure 12 - Link Speed Train down Debug Flow Chart

FPGA Configuration Time Debug

If the user_lnk_up is asserted but the device is not detected by the host, it could be caused by not having the FPGA configured fast enough to enter link training and be recognized by the system. Section 6.6 of PCI Express Base Specification, rev. 2.1 states two rules that might be impacted by FPGA Configuration Time:

- A component must enter the LTSSM Detect state within 20 ms of the end of the Fundamental reset.
- A system must guarantee that all components intended to be software visible at boot time are ready to receive Configuration Requests within 100 ms of the end of Conventional Reset at the Root Complex.

These statements mean the FPGA must be configured within a certain finite time, and not meeting these requirements could cause problems with link training and device recognition. When using JTAG to configure the device, configuration typically occurs after the Chipset has enumerated each peripheral. After configuring the FPGA, a soft reset is required to restart enumeration and configuration of the device. A soft reset on a Windows based PC is performed by going to **Start -> Shut Down** and then selecting **Restart**.

To eliminate FPGA configuration as a root cause, the designer should perform a soft restart of the system. Performing a soft reset on the system keeps power applied and forces re-enumeration of the device. If the device links up and is recognized after a soft reset is performed, then FPGA configuration is most likely the issue. Most typical systems use ATX power supplies which provide some margin on this 100 ms window, as the power supply is normally valid before the 100 ms window starts.

PCIe Clocking

Many link training issues arise from bad clocking; it is advised to make sure the clocking to the core is correct before investigating further. If `user_reset_out` signals are not asserted, it could be that the fabric PLL (MMCM) and Transceiver PLL have not locked to the incoming clock. To verify clocking is correct, make sure the signals in Figure 19 are asserted and deasserted correctly. If the PLLs do not lock as expected, it is necessary to ensure the incoming reference clock meets the requirements in the *7 Series FPGAs GTX/GTH Transceivers User Guide* ([UG476](#)), as listed below:

- Provide AC coupling between the oscillator output pins and the dedicated GTX/GTH transceiver Quad clock input pins.
- Ensure that the differential voltage swing of the reference clock is the range as specified in [DS182](#) (Kintex-7 FPGAs Data Sheet: DC and Switching Characteristics) and [DS183](#) (Virtex-7 FPGAs Data Sheet: DC and Switching Characteristics). The nominal range is 250 mV – 2000 mV, and the nominal value is 1200 mV.
- Meet or exceed the reference clock characteristics as specified in [DS182](#) (Kintex-7 FPGAs Data Sheet: DC and Switching Characteristics) and [DS183](#) (Virtex-7 FPGAs Data Sheet: DC and Switching Characteristics).
- Meet or exceed the reference clock characteristics as specified in the standard for which the GTX/GTH transceiver provides physical layer support.
- Fulfill the oscillator vendor's requirement regarding power supply, board layout, and noise specification.
- Provide a dedicated point-to-point connection between the oscillator and GTX/GTH transceiver Quad clock input pins.
- Keep impedance discontinuities on the differential transmission lines to a minimum (impedance discontinuities generate jitter).

The bit rate clock source for transmitter and receiver must be +/- 300 ppm or better. If Spread Spectrum Clocking is used, both ports must use the same bit rate clock source.

There should be AC coupling between the clock source and the dedicated GTX transceiver Quad clock input pins.

PCB Design Checklist

As defined in the specification, it is required to put AC coupling capacitors at the transmitter lanes differential signal pair. The value of AC coupling capacitor is between 75 nF and 200 nF. The user should make sure that the PCI express card has an AC coupling capacitor placed in the close proximity of the transmitter lane. Check if the correct AC capacitor value has been put in place or not. There might be a possibility for a cracked capacitor.

Make sure the PCB Design Checklist provided in the *7 Series FPGAs GTX/GTH Transceivers* ([UG476](#)) has been followed. Table 2 is from ([UG476 v1.9.1, April 23, 2013](#)). Please visit the Xilinx website ([7-Series documentation](#)) for the latest version of the UG476 where there may be more current guidelines on the PCB Design Checklist provided in Table 2.

Table 2 - PCB Design Checklist

Pins	Recommendations
MGTREFCLK0P MGTREFCLK0N MGTREFCLK1P MGTREFCLK1N	<ul style="list-style-type: none"> • Use AC coupling capacitors for connection to oscillator. • For AC coupling capacitors, see Reference Clock Interface, page 303. • Reference clock traces should be provided enough clearance to eliminate crosstalk from adjacent signals. • Reference clock oscillator output must comply with the minimum and maximum input amplitude requirements for these input pins. See DS182, Kintex-7 FPGAs Data Sheet: DC and Switching Characteristics or DS183, Virtex-7 FPGAs Data Sheet: DC and Switching Characteristics. • If reference clock input is not used, leave the associated pin pair unconnected.
MGTXRX[3:0]/MGTXRXN[3:0] MGTHRX[3:0]/MGTHRXN[3:0]	<ul style="list-style-type: none"> • Use AC coupling capacitors for connection to transmitter. The recommended value for AC coupling capacitors is 100 nF. • Receiver data traces should be provided enough clearance to eliminate crosstalk from adjacent signals. • If a receiver is not used, connect the associated pin pair to ground. • See RX Analog Front End, page 160.
MGTXXP[3:0]/MGTXN[3:0] MGTHXP[3:0]/MGHTXN[3:0]	<ul style="list-style-type: none"> • Transmitter should be AC coupled to the receiver. The recommended value for the AC coupling capacitors is 100 nF. • Transmitter data traces should be provided enough clearance to eliminate crosstalk from adjacent signals. • If a transmitter is not used, leave the associated pin pair unconnected.
Pins	Recommendations
MGTAVTTRCAL	<ul style="list-style-type: none"> • Connect to MGTAVTT and to a 100Ω resistor that is also connected to MGTRREF. Use identical trace geometry for the connection between the resistor and this pin and for the connection from the other pin of the resistor to MGTRREF. • See Termination Resistor Calibration Circuit, page 294.
MGTRREF	<ul style="list-style-type: none"> • Connect to a 100Ω resistor that is also connected to MGTAVTTRCAL. Use identical trace geometry for the connection between the resistor to this pin and for the connection from the other pin of the resistor to MGTAVTTRCAL. • See Termination Resistor Calibration Circuit, page 294.

MGTAVCC[N]	<ul style="list-style-type: none"> • The nominal voltage is 1.0 VDC. • See DS182, <i>Kintex-7 FPGAs Data Sheet: DC and Switching Characteristics</i> or DS183, <i>Virtex-7 FPGAs Data Sheet: DC and Switching Characteristics</i> for power supply voltage tolerances. • The power supply regulator for this voltage should not be shared with non-transceiver loads. • Many packages have multiple groups of power supply connections in the package for MGTAVCC. Refer to Table 5-2, page 295 to identify in which power supply group a specific GTX/GTH transceiver Quad is located. Information on pin locations for each package can be found in UG475, <i>7 Series FPGAs Packaging and Pinout Specifications</i>. • The following filter capacitor is recommended: <ul style="list-style-type: none"> • 1 of 4.7 μF 10% • For optimal performance, power supply noise must be less than 10 mVpp. • If all of the Quads in a power supply group are not used, the associated power pins can be left unconnected or tied to ground. • For power consumption, refer to the XPower Estimator (XPE) for 7 series devices at www.xilinx.com/power.
MGTAVTT[N]	<ul style="list-style-type: none"> • The nominal voltage is 1.2 VDC. • See DS182, <i>Kintex-7 FPGAs Data Sheet: DC and Switching Characteristics</i> or DS183, <i>Virtex-7 FPGAs Data Sheet: DC and Switching Characteristics</i> for power supply voltage tolerances. • The power supply regulator for this voltage should not be shared with non-MGT loads. • Many packages have multiple groups of power supply connections in the package for MGTAVTT. Refer to Table 5-2, page 295 to identify in which power supply group a specific GTX/GTH transceiver Quad is located. Information on pin locations for each package can be found in UG475, <i>7 Series FPGAs Packaging and Pinout Specifications</i>. • The following ceramic filter capacitor is recommended: <ul style="list-style-type: none"> • 1 of 4.7 μF 10% • For optimal performance, power supply noise must be less than 10 mVpp. • If all of the Quads in a power supply group are not used, the associated power pins can be left unconnected or tied to ground. • For power consumption, refer to the XPower Estimator (XPE) for 7 series devices at www.xilinx.com/power.
MGTVCCAUX[N]	<ul style="list-style-type: none"> • The nominal voltage is 1.8 VDC. • See DS182, <i>Kintex-7 FPGAs Data Sheet: DC and Switching Characteristics</i> or DS183, <i>Virtex-7 FPGAs Data Sheet: DC and Switching Characteristics</i> for power supply voltage tolerances. • The power supply regulator for this voltage should not be shared with non-MGT loads. • Many packages have multiple groups of power supply connections in the package for MGTAVTT. Refer to Table 5-2, page 295 to identify in which power supply group a specific GTX/GTH transceiver Quad is located. For information on pin locations for each package, see UG475, <i>7 Series FPGAs Packaging and Pinout Specifications</i>. • The following filter capacitor is recommended: <ul style="list-style-type: none"> • 1 of 4.7 μF 10% • For optimal performance, power supply noise must be less than 10 mVpp. • If all of the QPLLs in this power supply group are not used but the Quads are used, the filter capacitors are not necessary and these pins can be connected to VCCAUX. • If all of the Quads in a power supply group are not used, the associated pins can be left unconnected or tied to ground.

Link Training Debug Signals

As detailed in “*Link Training Failure Types and Debug Flow*” section, link training problem could be due to a range of issues. There are different signals you should capture and analyze depending on the nature of the issue. In this section, the signals are grouped into different sets to make it easier to understand.

Signal Set-1, LTSSM

The `pl_ltssm_state[5:0]` signal as shown in Figure 14 is one of the main signals when debugging link training issues. Whenever there is a problem with the link, it is always advised to check this signal in ChipScope tool and see what LTSSM state the core is in. In the normal working condition, this signal will show the value ‘16’ indicating L0 state.

The core goes into Recovery state to achieve bit lock and symbol lock. If the ChipScope capture shows frequent transition into the recovery state, it normally indicates a noisy link.

Different states of the link training state machine (indicated by `pl_ltssm_state`) are shown in Figure 13.

0, 1: Detect Quiet	1B: L1 Exit
2, 3: Detect Active	1C: Recovery Rcvrlock
4: Polling Active	1D: Recovery Rcvrcfg
5: Polling Configuration	1E: Recovery Speed_0
6: Polling Compliance, Pre_Send_EIOS	1F: Recovery Speed_1
7: Polling Compliance, Pre_Timeout	20: Recovery Idle
8: Polling Compliance, Send_Pattern	21: Hot Reset
9: Polling Compliance, Post_Send_EIOS	22: Disabled Entry 0
A: Polling Compliance, Post_Timeout	23: Disabled Entry 1
B: Configuration Linkwidth, State 0	24: Disabled Entry 2
C: Configuration Linkwidth, State 1	25: Disabled Idle
D: Configuration Linkwidth, Accept 0	26: Root Port, Configuration, Linkwidth State 0
E: Configuration Linkwidth, Accept 1	27: Root Port, Configuration, Linkwidth State 1
F: Configuration Lanenum Wait	28: Root Port, Configuration, Linkwidth State 2
10: Configuration Lanenum, Accept	29: Root Port, Configuration, Link Width Accept 0
11: Configuration Complete x1	2A: Root Port, Configuration, Link Width Accept 1
12: Configuration Complete x2	2B: Root Port, Configuration, Lanenum_Wait
13: Configuration Complete x4	2C: Root Port, Configuration, Lanenum_Accept
14: Configuration Complete x8	2D: Timeout To Detect
15: Configuration Idle	2E: Loopback Entry0
16: L0	2F: Loopback Entry1
17: L1 Entry0	30: Loopback Active0
18: L1 Entry1	31: Loopback Exit0
19: L1 Entry2	32: Loopback Exit1
1A: L1 Idle	33: Loopback Master Entry0

Figure 13 - LTSSM States

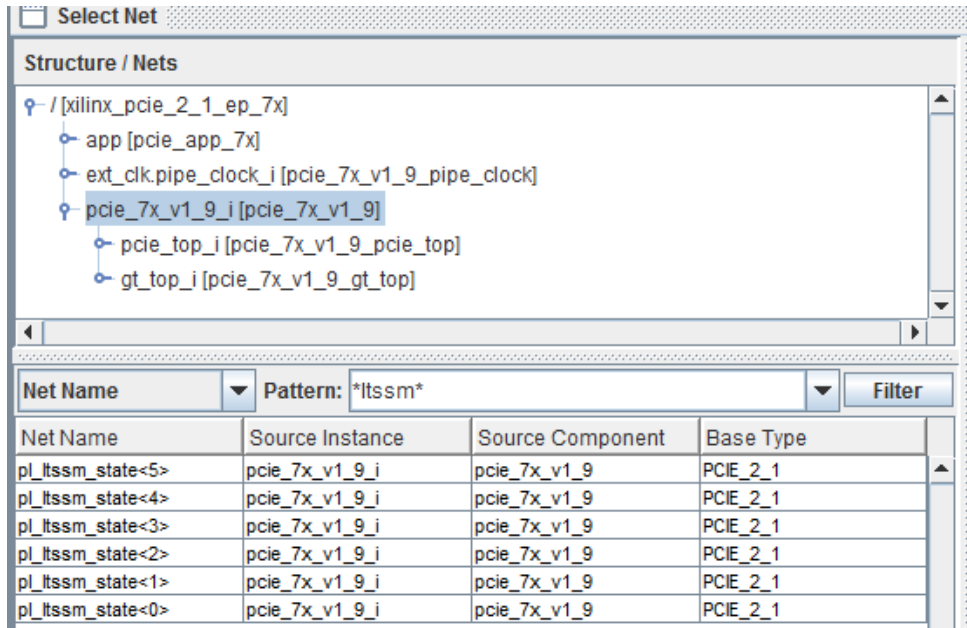


Figure 14 - LTSSM Signal

Signal Set-2, Detect State

Signals shown in Figure 15 should be captured to check whether the receiver was successfully detected or not.

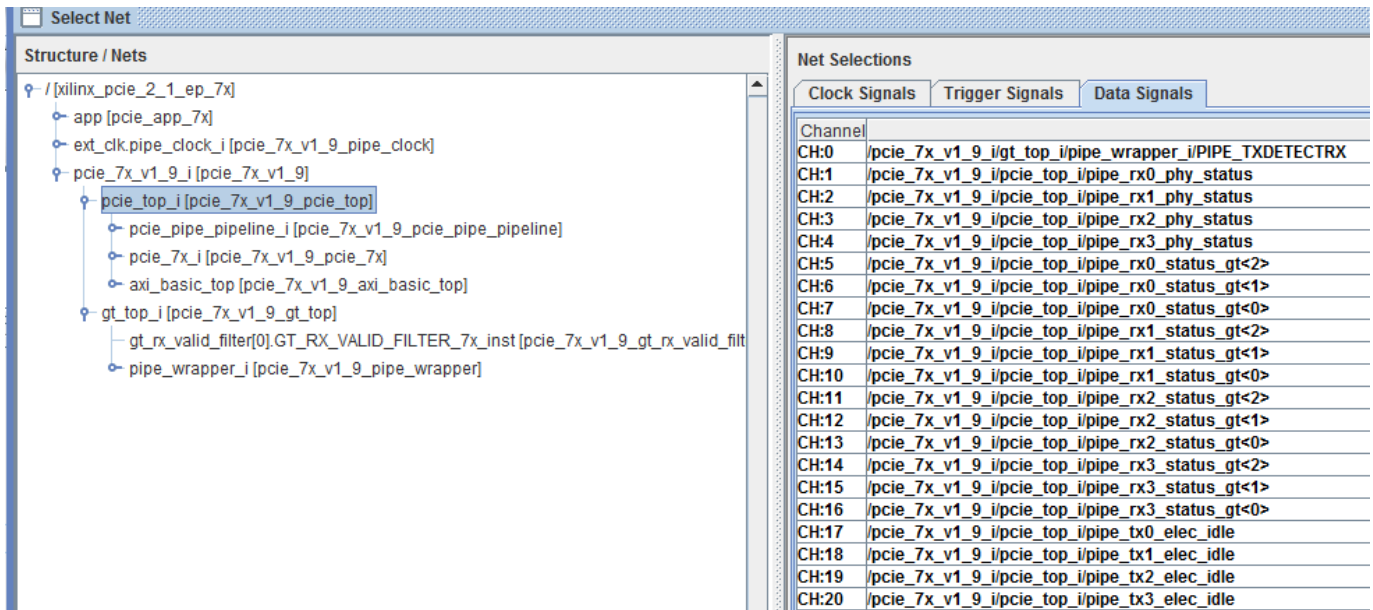


Figure 15 - Detect State Signals

TXDETECTRX: This input activates the receive detection sequence. The sequence ends when PHYSTATUS is asserted to indicate that the results of the test are ready on RXSTATUS.

PHYSTATUS: In PCI Express mode, this signal is used to communicate completion of several GTX transceiver functions, including power management state transitions, rate change, and receiver detection. During receiver detection, this signal is asserted High to indicate receiver detection completion.

During receiver detection, RXSTATUS signal is read when PHYSTATUS is asserted High. Only these encodings are valid during receiver detection:

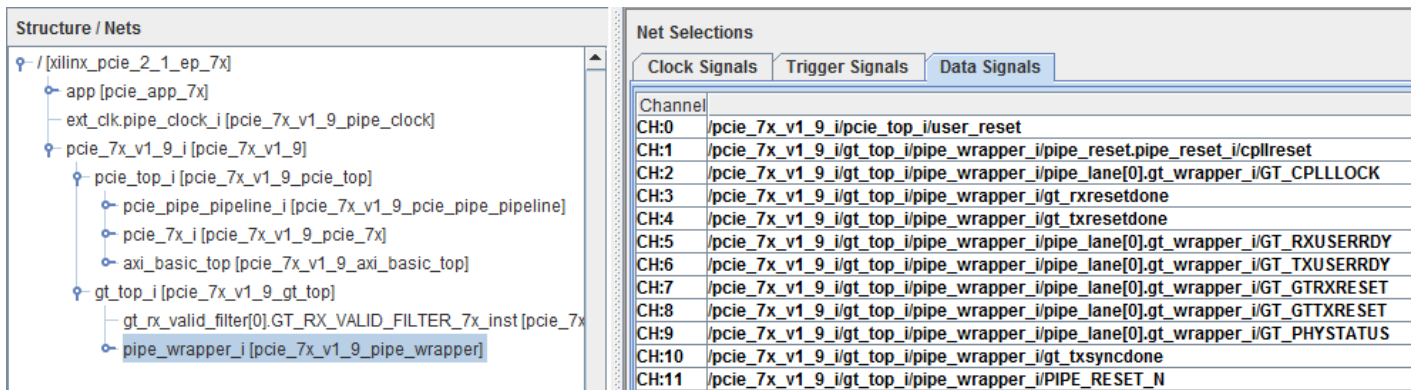
- 000: Receiver not present.
- 011: Receiver present.

RXSTATUS indicates the receiver status and error codes as shown in Figure 16.

000: Data Received OK
001: One Skip Symbol (SKP) added
010: One SKP removed
011: Receiver detected
100: 8B/10B decode error
101: Elastic Buffer overflow
110: Elastic Buffer underflow
111: Receive disparity error

Figure 16 - RXSTATUS Encoding

Signal Set-3, Reset



The screenshot shows a logic analyzer interface. On the left, a tree view under 'Structure / Nets' shows a hierarchy of signals, with 'pipe_wrapper_i [pcie_7x_v1_9_pipe_wrapper]' selected. On the right, the 'Net Selections' panel is active, showing a list of channels (CH:0 to CH:11) with their corresponding signal names.

Channel	Signal Name
CH:0	/pcie_7x_v1_9_i/pcie_top_i/user_reset
CH:1	/pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_reset.pipe_reset_i/cpllreset
CH:2	/pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_CPLLLOCK
CH:3	/pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/gt_rxresetdone
CH:4	/pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/gt_txresetdone
CH:5	/pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_RXUSERDDY
CH:6	/pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_TXUSERDDY
CH:7	/pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_GTRXRESET
CH:8	/pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_GTXRESET
CH:9	/pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_PHYSTATUS
CH:10	/pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/gt_txsyncdone
CH:11	/pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/PIPE_RESET_N

Figure 17 – Reset related signals

GTRXRESET: This port is asserted high and then deasserted to start the full channel RX reset sequence.

RXRESETDONE: This port goes high when GT transceiver RX has finished reset and is ready for use.

TXRESETDONE: This port goes high when the GT transceiver TX has finished reset and is ready for use.

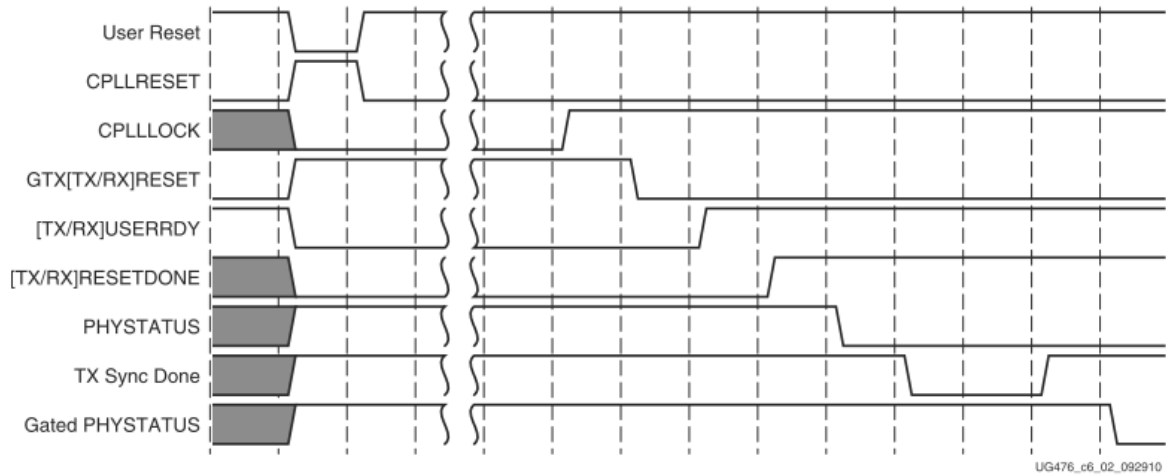


Figure 18 – Correct toggling of reset related signals

Signal Set-4, Clocking

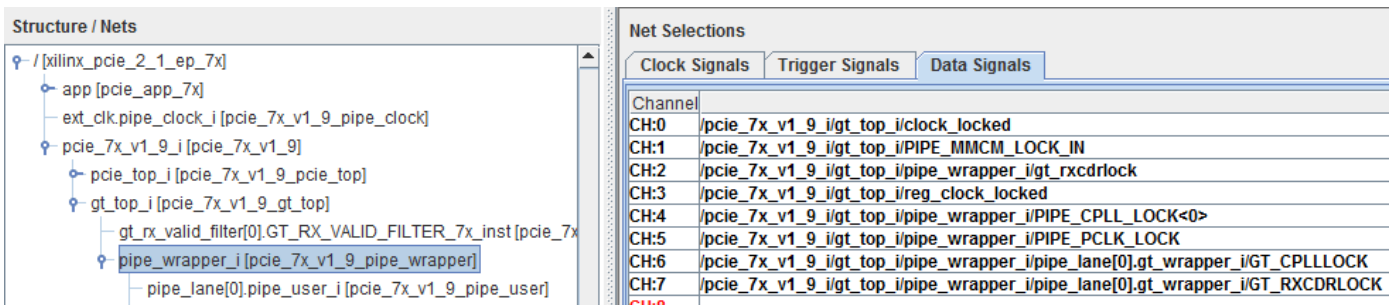


Figure 19 - PCIe Clocking Related Signals

Signal Set-5, GT RX

The screenshot shows the 'Select Net' window with a tree view on the left and a 'Net Selections' table on the right. The tree view shows a hierarchy starting from 'xilinx_pcie_2_1_ep_7x' down to 'pipe_wrapper_i' and 'pipe_lane[0].gt_wrapper_i'. The 'Net Selections' table has three tabs: 'Clock Signals', 'Trigger Signals', and 'Data Signals'. The 'Data Signals' tab is active, showing a list of channels from CH:0 to CH:28. Channel CH:2 is highlighted in red.

Channel	Signal Name
CH:0	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/PIPE_RXCHANISALIGNED<0>
CH:1	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/PIPE_RXDATA<0>
CH:2	
CH:3	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/PIPE_RXDATA<31>
CH:4	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/PIPE_RXDATAK<0>
CH:5	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/PIPE_RXDATAK<1>
CH:6	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/PIPE_RXDATAK<2>
CH:7	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/PIPE_RXDATAK<3>
CH:8	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/PIPE_RXELECIDL<0>
CH:9	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/PIPE_RXPOLARITY<0>
CH:10	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/PIPE_RXSTATUS<0>
CH:11	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/PIPE_RXSTATUS<1>
CH:12	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/PIPE_RXSTATUS<2>
CH:13	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_RXBUFSTATUS<0>
CH:14	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_RXBUFSTATUS<1>
CH:15	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_RXBUFSTATUS<2>
CH:16	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_RXBYTEISALIGNED
CH:17	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_RXCDRLOCK
CH:18	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_RXCHANISALIGNED
CH:19	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_RXCDRRESET
CH:20	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_RXCHARISCOMMA<0>
CH:21	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_RXCHARISCOMMA<1>
CH:22	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_RXCHARISCOMMA<2>
CH:23	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_RXCHARISCOMMA<3>
CH:24	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_RXCHARISCOMMADET
CH:25	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_RXELECIDL
CH:26	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_RXPHALIGN
CH:27	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_RXPHALIGNDONE
CH:28	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_RXVALID

Figure 20 – GT Receive Channel Signals

Signal Set-6, GT TX

The screenshot shows the 'Select Net' window with a tree view on the left and a 'Net Selections' table on the right. The tree view shows a hierarchy starting from 'xilinx_pcie_2_1_ep_7x' down to 'pipe_wrapper_i' and 'pipe_lane[0].gt_wrapper_i'. The 'Net Selections' table has three tabs: 'Clock Signals', 'Trigger Signals', and 'Data Signals'. The 'Data Signals' tab is active, showing a list of channels from CH:30 to CH:47. Channel CH:32 is highlighted in red.

Channel	Signal Name
CH:30	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_GTTXRESET
CH:31	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_TXDATA<0>
CH:32	
CH:33	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_TXDATA<31>
CH:34	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_TXDATAK<0>
CH:35	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_TXDATAK<1>
CH:36	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_TXDATAK<2>
CH:37	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_TXDATAK<3>
CH:38	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_TXDEMPH
CH:39	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_TXELECIDL
CH:40	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_TXPHALIGN
CH:41	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_TXPHALIGNDONE
CH:42	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_TXMARGIN<0>
CH:43	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_TXMARGIN<1>
CH:44	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_TXMARGIN<2>
CH:45	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_TXRATEDONE
CH:46	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_TXRESETDONE
CH:47	/pcie_7x_v1_9_i/igt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/IGT_TXUSERRDY

Figure 21 - GT Transmit Channel Signals

Signal Set-7, User

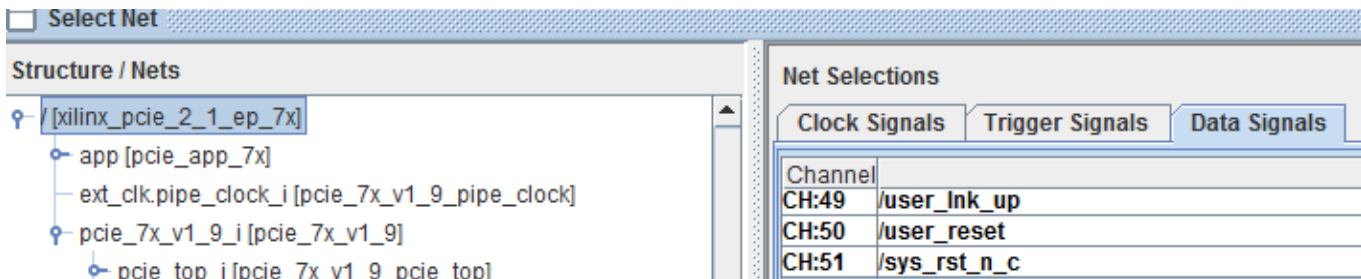


Figure 22 - Reset and Link up signal from the core to the user application

Signal Set-8, Physical Layer

The Physical Layer (PL) interface enables the user design to inspect the status of the Link and Link Partner and control the Link State.

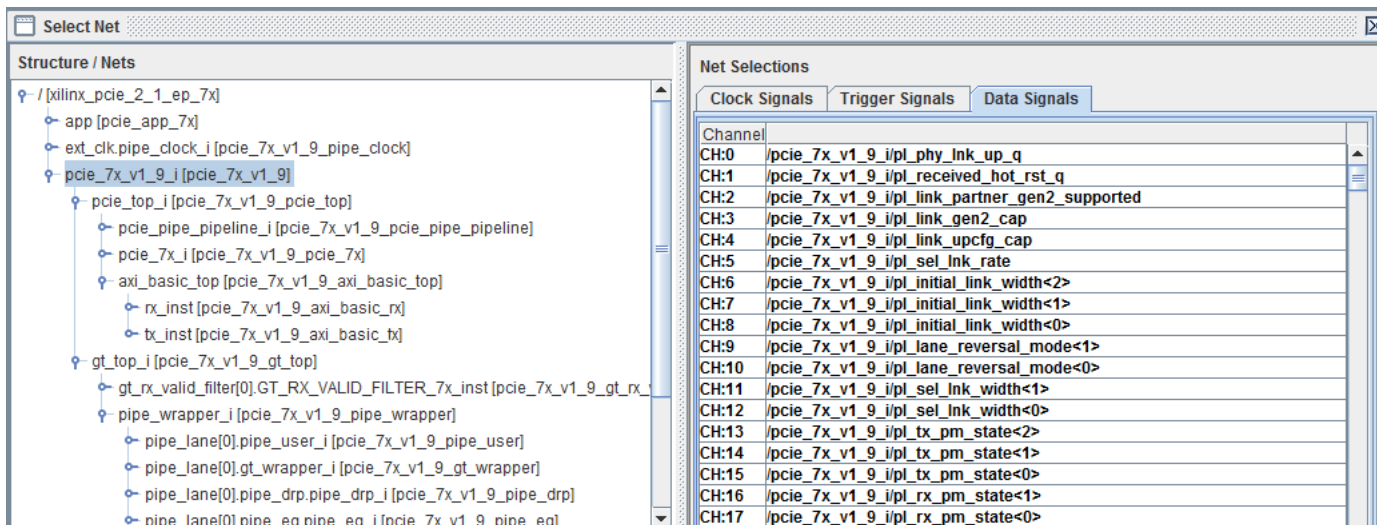


Figure 23 - Physical Layer Interface Signals

Name	Direction	Description
pl_initial_link_width[2:0]	Output	Initial Negotiated Link Width: Indicates the link width after the PCI Express port has achieved the first successful link training. Initial Negotiated Link Width represents the widest link width possible during normal operation of the link, and can be equal to or smaller than the capability link width (smaller of the two) supported by link partners. This value is reset when the core is reset or the LTSSM goes through the Detect state. Otherwise the value remains the same. <ul style="list-style-type: none"> • 000: Link not trained • 001: 1-Lane link • 010: 2-Lane link • 011: 4-Lane link • 100: 8-Lane link
pl_phy_ink_up	Output	Physical Layer Link Up Status: Indicates the physical layer link up status.

pl_rx_pm_state[1:0]	Output	RX Power Management State: Indicates the RX Power Management State: 00: RX Not in L0s 01: RX L0s Entry 10: RX L0s Idle 11: RX L0s FTS
pl_tx_pm_state[2:0]	Output	TX Power Management State: Indicates the TX Power Management State: 000: TX not in L0s 001: TX L0s Entry0 010: TX L0s Entry1 011: TX L0s Entry2 100: TX L0s Idle 101: TX L0s FTS0 110: TX L0s FTS1 111: TX L0s FTS2
pl_lane_reversal_mode[1:0]	Output	Lane Reversal Mode: Indicates the current Lane Reversal mode. <ul style="list-style-type: none"> • 00: No reversal • 01: Lanes 1:0 reversed • 10: Lanes 3:0 reversed • 11: Lanes 7:0 reversed
pl_link_gen2_cap	Output	Link Gen2 Capable: Indicates that the PCI Express link is 5.0 Gb/s (Gen 2) speed capable (both the Link Partner and the Device are Gen 2 capable) <ul style="list-style-type: none"> • 0: Link is not Gen2 Capable • 1: Link is Gen2 Capable
pl_link_partner_gen2_supported	Output	Link Partner Gen2 Capable: Indicates if the PCI Express link partner advertises 5.0 Gb/s (Gen2) capability. Valid only when user_lnk_up is asserted. <ul style="list-style-type: none"> • 0: Link partner not Gen2 capable • 1: Link partner is Gen2 capable
pl_link_upcfg_cap	Output	Link Upconfigure Capable: Indicates the PCI Express link is Upconfigure capable. Valid only when user_lnk_up is asserted. <ul style="list-style-type: none"> • 0: Link is not Upconfigure capable • 1: Link is Upconfigure capable
pl_sel_lnk_rate	Output	Current Link Rate: Reports the current link speed. Valid only when user_lnk_up is asserted. 0: 2.5 Gb/s 1: 5.0 Gb/s
pl_sel_lnk_width[1:0]	Output	Current Link Width: Reports the current link width. Valid only when user_lnk_up is asserted. 00: 1-Lane link 01: 2-Lane link 10: 4-Lane link 11: 8-Lane link

Table 3 – Physical Layer Interface Signals Description (PG023)

LTSSM Signal Analysis

After the FPGA configuration, the two connected devices go through the link training process. The main states to consider while debugging link training issues are DETECT, POLLING, CONFIGURATION and L0. Refer to section 4.2.5 of the PCI Express Base Specification v2.1 for detailed description on these LTSSM states.

This section provides an analysis of signals described in previous section at different LTSSM states for debugging link training issues. A number of waveform screenshots have been provided for each LTSSM state to illustrate the toggling of corresponding signals. If you are capturing signals in ChipScope tool, compare your captures with the screenshots provided below to make sure the signals in your design are toggling as expected.



Figure 24 – Signal Set-1, LTSSM

DETECT State

Figure 25 shows a capture of signals related to Detect state during Detect.Active and Polling.Active states. On successful receiver detection, the pipe wrapper should present '011' on RXSTATUS when PHYSTATUS is asserted as shown in Figure 26.

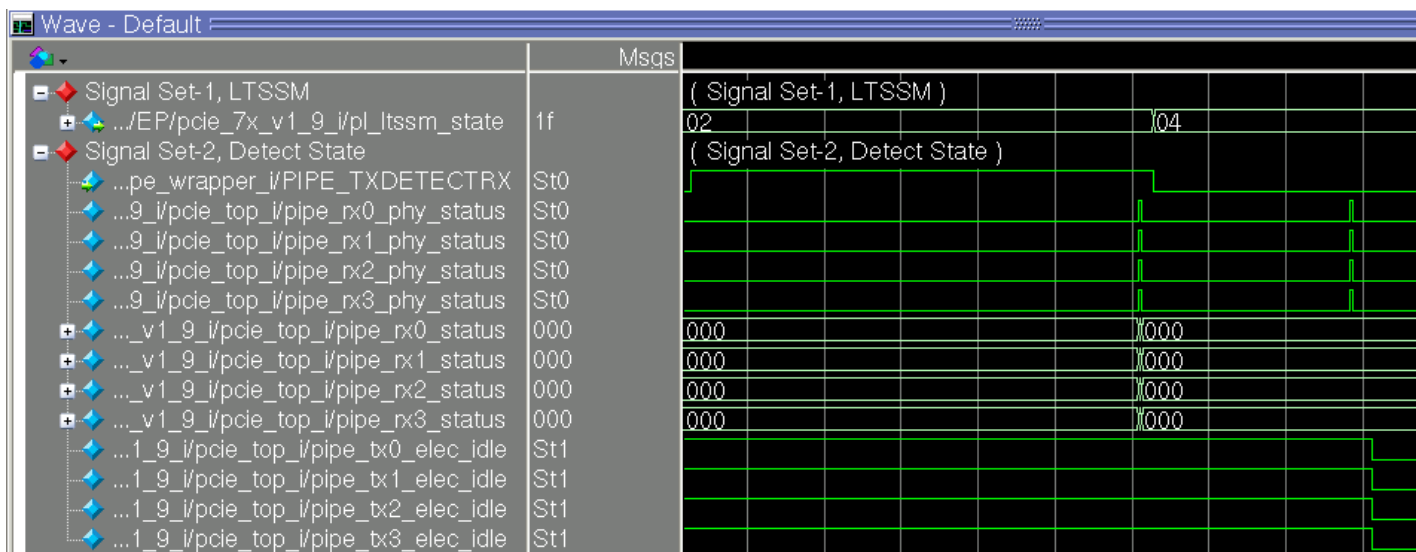


Figure 25 – Signal Set-2 in Detect.Active and Polling.Active LTSSM states

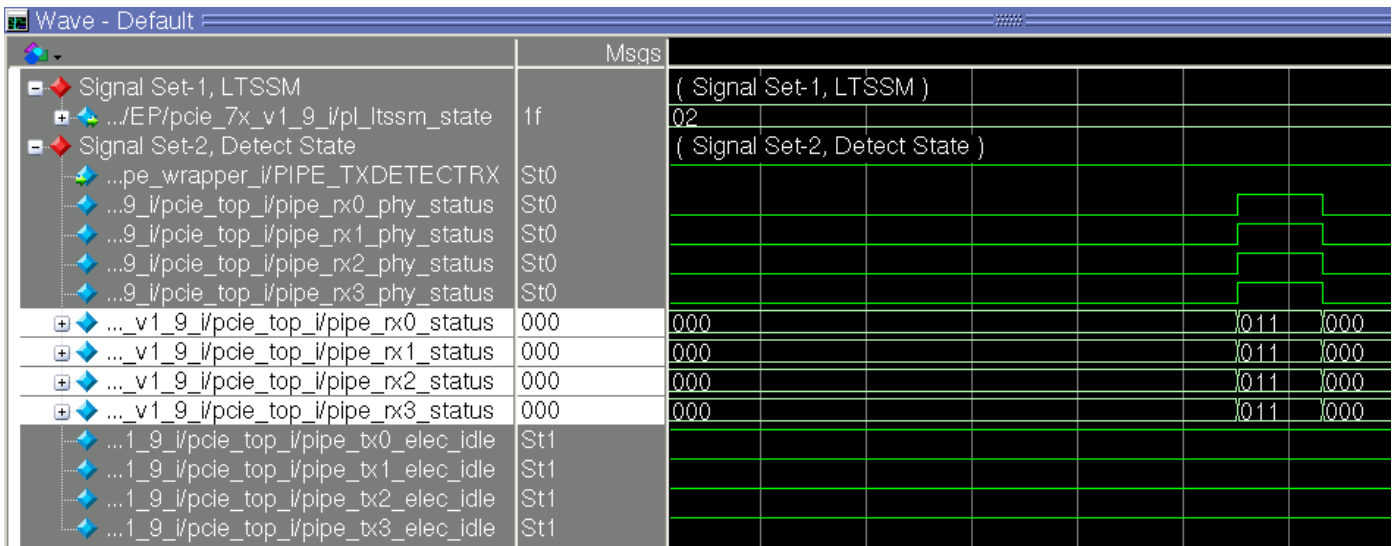


Figure 26 - Successful receiver detection

During the DETECT state, the receiver detection takes place on each lane. If the detection process is done correctly, the following sequence should be observed in the ChipScope capture.

- PCIe Hard Block asserts TxDetectRx.
- GT performs receiver DETECT.
- After the receiver is detected, GTP asserts pipe_rx_phy_status and puts 011 on pipe_rx_status to indicate the receiver is present.
- PCIe Hard Block then de-asserts TxDetectRx and pipe_tx_elec_idle.

If the receiver detect is failing, then make sure the signals in “**Signal Set-3, Rese**” and “**Signal Set-4, Clocking**” as shown in Figure 27 are correctly toggling.

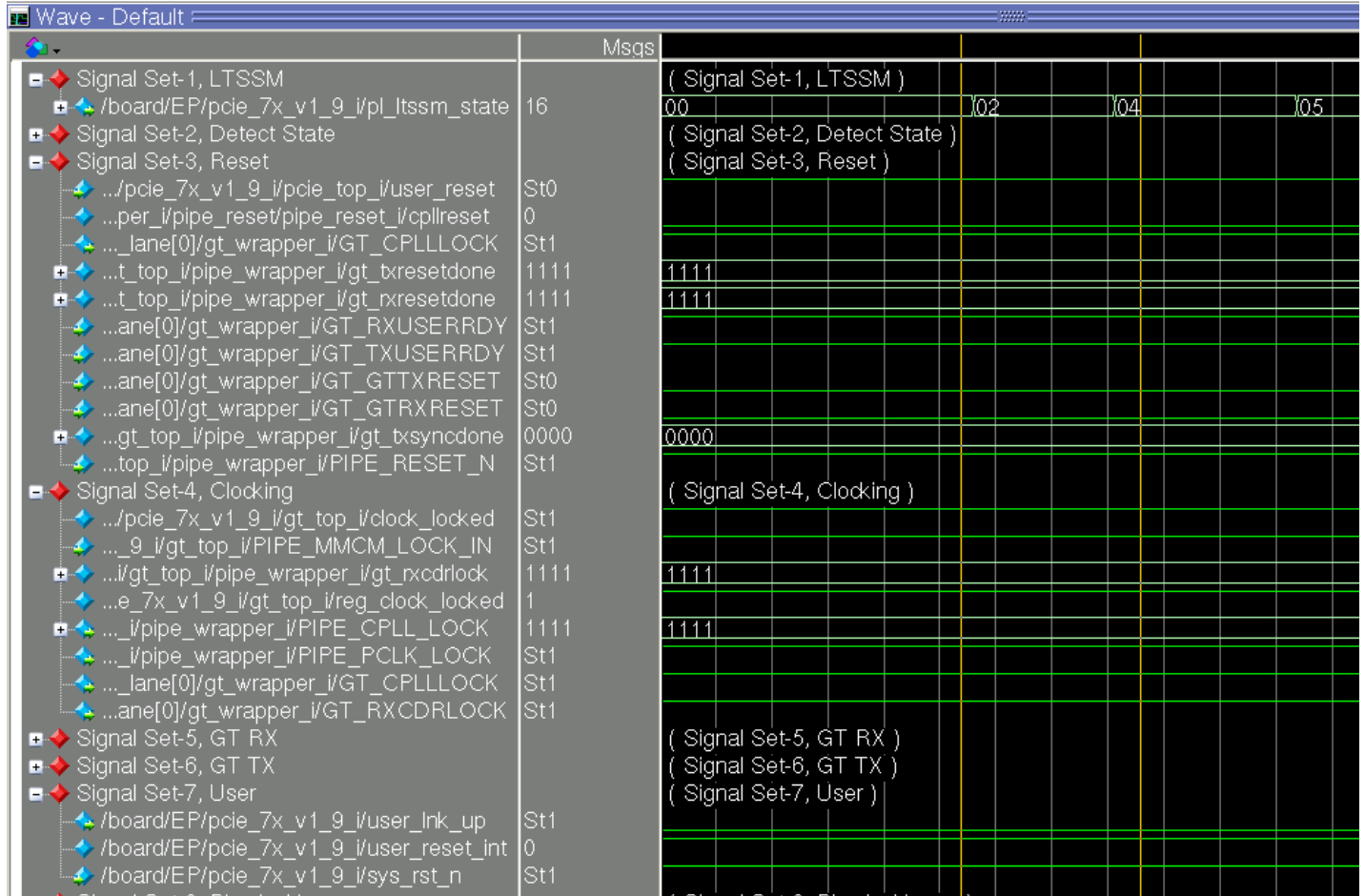


Figure 27 – Reset and Clocking related signals

POLLING State

When each link partner enters into POLLING, it begins transmitting TS1 ordered sets. However, each link partner might not enter polling at the same time, so it is possible that the Xilinx endpoint might be transmitting TS1s on pipe_tx_data while still receiving 00h on the pipe_rx_data pins. Hence, in ChipScope Pro/Vivado ILA tools, when TS1 appears at pipe_tx_data, pipe_rx_data might still be 00.

Figure 28 shows different sub states inside POLLING state.

- 4: Polling Active
- 5: Polling Configuration
- 6: Polling Compliance, Pre_Send_EIOS
- 7: Polling Compliance, Pre_Timeout
- 8: Polling Compliance, Send_Pattern
- 9: Polling Compliance, Post_Send_EIOS
- A: Polling Compliance, Post_Timeout

Figure 28 - Sub-states in LTSSM Polling state

To check whether TS1 transmission has started or not, trigger when ltssm_state enters POLLING. Figure 29 and Figure 30 show GT RX and TX interface signals when the endpoint device enters POLLING. As soon as the device comes out of the electrical idle, the device starts to send TS1s. Note that the link and lane number are set to PAD value which

is F7. TS1 ends with 4A, whereas TS2 ends with 45. According to the PCI Express Base Specification v2.1, both devices should send a minimum of 1024 TS1s, which amounts to 64 μ s, to achieve bit and symbol lock.

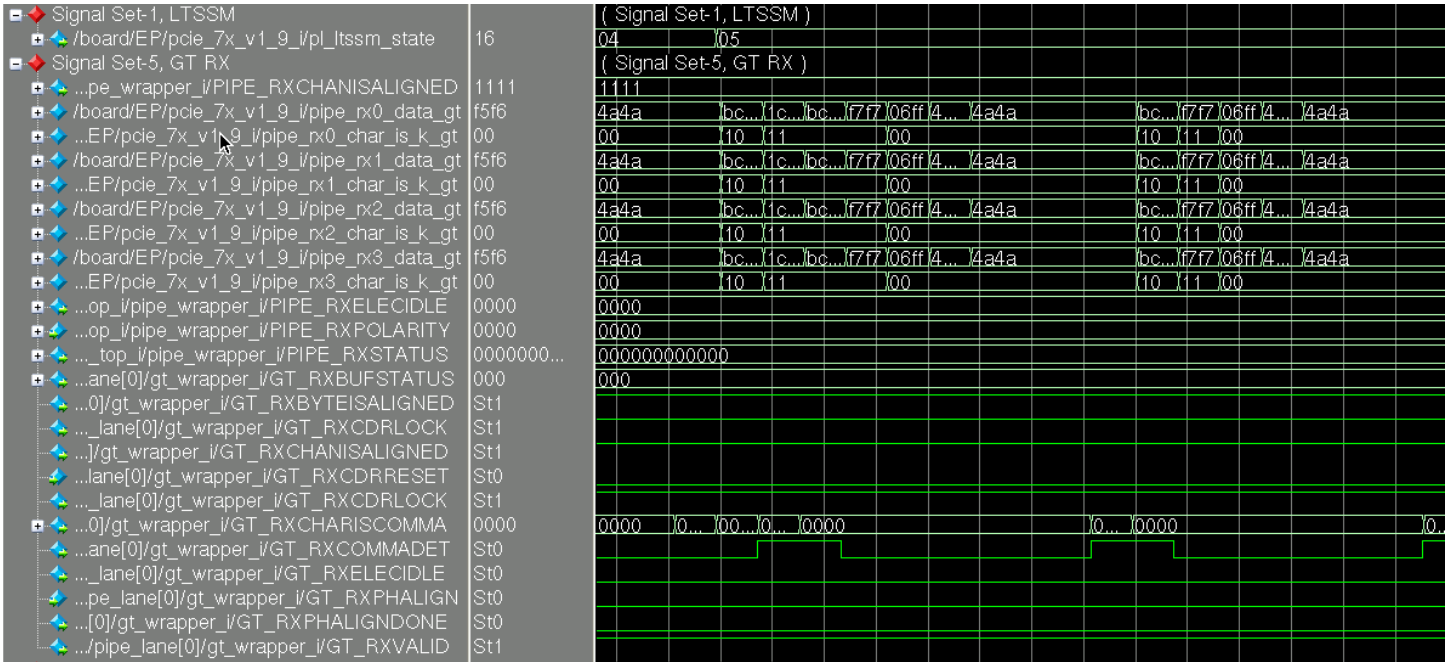


Figure 29 – GT RX channel interface signals during EP Polling.Active and Polling.Configuration

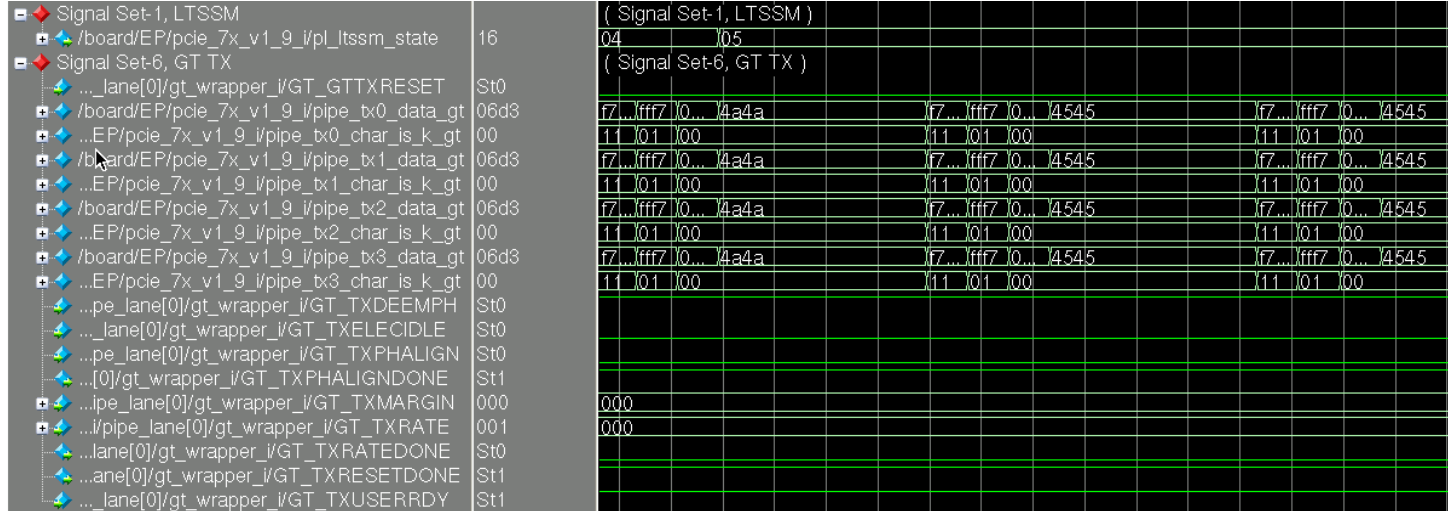


Figure 30 - GT TX channel interface signals during EP Polling.Active and Polling.Configuration

After receiving eight consecutive TS2 ordered sets and transmitting 16 TS2 ordered sets (after receiving one TS2 ordered set), the device exits to the configuration state. Devices at both ends of the link do not exit to CONFIGURATION at the same time.

CONFIGURATION State

In CONFIGURATION, link numbers and lane numbers are negotiated. A downstream port proposes a link number to the link partner. The upstream port accepts the link number and returns TS1 ordered sets with the link number value. Next, the downstream port sends the lane numbers. If the upstream port agrees with the proposed lane numbers, it replies with

Signal Set-1, LTSSM		(Signal Set-1, LTSSM)							
/board/EP/pcie_7x_v1_9_i/pl_tssm_state	0c	0b							0c
Signal Set-5, GT RX		(Signal Set-5, GT RX)							
...pe_wrapper_i/PIPE_RXCHANISALIGNED	1111	1111							
RX DATA		(RX DATA)							
...d/EP/pcie_7x_v1_9_i/pipe_rx0_data_gt	f700	4a4a	bc4a	f700	06ff	4a00	4a4a		
...pcie_7x_v1_9_i/pipe_rx0_char_is_k_gt	10	00	10		00				
...d/EP/pcie_7x_v1_9_i/pipe_rx1_data_gt	f700	4a4a	bc4a	f700	06ff	4a00	4a4a		
...pcie_7x_v1_9_i/pipe_rx1_char_is_k_gt	10	00	10		00				
...d/EP/pcie_7x_v1_9_i/pipe_rx2_data_gt	f700	4a4a	bc4a	f700	06ff	4a00	4a4a		
...pcie_7x_v1_9_i/pipe_rx2_char_is_k_gt	10	00	10		00				
...d/EP/pcie_7x_v1_9_i/pipe_rx3_data_gt	f700	4a4a	bc4a	f700	06ff	4a00	4a4a		
...pcie_7x_v1_9_i/pipe_rx3_char_is_k_gt	10	00	10		00				
RX Other Signals		(RX Other Signals)							
Signal Set-6, GT TX		(Signal Set-6, GT TX)							
Signal Set-6, GT TX		(Signal Set-6, GT TX)							
TX DATA		(TX DATA)							
...d/EP/pcie_7x_v1_9_i/pipe_tx0_data_gt	4a4a	4a4a			f7bc	fff7	0006	4a4a	
...pcie_7x_v1_9_i/pipe_tx0_char_is_k_gt	00	00			11	01	00		
...d/EP/pcie_7x_v1_9_i/pipe_tx1_data_gt	4a4a	4a4a			f7bc	fff7	0006	4a4a	
...pcie_7x_v1_9_i/pipe_tx1_char_is_k_gt	00	00			11	01	00		
...d/EP/pcie_7x_v1_9_i/pipe_tx2_data_gt	4a4a	4a4a			f7bc	fff7	0006	4a4a	
...pcie_7x_v1_9_i/pipe_tx2_char_is_k_gt	00	00			11	01	00		
...d/EP/pcie_7x_v1_9_i/pipe_tx3_data_gt	4a4a	4a4a			f7bc	fff7	0006	4a4a	
...pcie_7x_v1_9_i/pipe_tx3_char_is_k_gt	00	00			11	01	00		
TX Other Signals		(TX Other Signals)							

Figure 33 – RP sending ‘00’ in link number field

Signal Set-1, LTSSM		(Signal Set-1, LTSSM)							
/board/EP/pcie_7x_v1_9_i/pl_tssm_state	0e	0e							
Signal Set-5, GT RX		(Signal Set-5, GT RX)							
...pe_wrapper_i/PIPE_RXCHANISALIGNED	1111	1111							
RX DATA		(RX DATA)							
...d/EP/pcie_7x_v1_9_i/pipe_rx0_data_gt	bc4a	4a4a	bc4a	0000	06ff	4a00	4a4a	bc4a	0000
...pcie_7x_v1_9_i/pipe_rx0_char_is_k_gt	10	00	10	00				10	00
...d/EP/pcie_7x_v1_9_i/pipe_rx1_data_gt	bc4a	4a4a	bc4a	0100	06ff	4a00	4a4a	bc4a	0100
...pcie_7x_v1_9_i/pipe_rx1_char_is_k_gt	10	00	10	00				10	00
...d/EP/pcie_7x_v1_9_i/pipe_rx2_data_gt	bc4a	4a4a	bc4a	0200	06ff	4a00	4a4a	bc4a	0200
...pcie_7x_v1_9_i/pipe_rx2_char_is_k_gt	10	00	10	00				10	00
...d/EP/pcie_7x_v1_9_i/pipe_rx3_data_gt	bc4a	4a4a	bc4a	0300	06ff	4a00	4a4a	bc4a	0300
...pcie_7x_v1_9_i/pipe_rx3_char_is_k_gt	10	00	10	00				10	00
RX Other Signals		(RX Other Signals)							
Signal Set-6, GT TX		(Signal Set-6, GT TX)							
Signal Set-6, GT TX		(Signal Set-6, GT TX)							
TX DATA		(TX DATA)							
...d/EP/pcie_7x_v1_9_i/pipe_tx0_data_gt	4a4a	fff7	0006	4a4a			00bc	fff7	0006
...pcie_7x_v1_9_i/pipe_tx0_char_is_k_gt	00	01	00				01	00	
...d/EP/pcie_7x_v1_9_i/pipe_tx1_data_gt	4a4a	fff7	0006	4a4a			00bc	fff7	0006
...pcie_7x_v1_9_i/pipe_tx1_char_is_k_gt	00	01	00				01	00	
...d/EP/pcie_7x_v1_9_i/pipe_tx2_data_gt	4a4a	fff7	0006	4a4a			00bc	fff7	0006
...pcie_7x_v1_9_i/pipe_tx2_char_is_k_gt	00	01	00				01	00	
...d/EP/pcie_7x_v1_9_i/pipe_tx3_data_gt	4a4a	fff7	0006	4a4a			00bc	fff7	0006
...pcie_7x_v1_9_i/pipe_tx3_char_is_k_gt	00	01	00				01	00	
TX Other Signals		(TX Other Signals)							

Figure 34 – RP sending link number and corresponding lane numbers on all four lanes, EP accepts link number but still sending PAD (F7) in lane number field

Signal Set-1, LTSSM		(Signal Set-1, LTSSM)								
.../board/EP/pcie_7x_v1_9_i/pl_ltssm_state	10	of 10								
Signal Set-5, GT RX		(Signal Set-5, GT RX)								
...pe_wrapper_i/PIPE_RXCHANISALIGNED	1111	1111								
RX DATA		(RX DATA)								
...d/EP/pcie_7x_v1_9_i/pipe_rx0_data_gt	4500	4545	bc45	0000	46ff	4500	4545	bc45	0000	
...pcie_7x_v1_9_i/pipe_rx0_char_is_k_gt	00	00	10	00				10	00	
...d/EP/pcie_7x_v1_9_i/pipe_rx1_data_gt	4500	4545	bc45	0100	46ff	4500	4545	bc45	0100	
...pcie_7x_v1_9_i/pipe_rx1_char_is_k_gt	00	00	10	00				10	00	
...d/EP/pcie_7x_v1_9_i/pipe_rx2_data_gt	4500	4545	bc45	0200	46ff	4500	4545	bc45	0200	
...pcie_7x_v1_9_i/pipe_rx2_char_is_k_gt	00	00	10	00				10	00	
...d/EP/pcie_7x_v1_9_i/pipe_rx3_data_gt	4500	4545	bc45	0300	46ff	4500	4545	bc45	0300	
...pcie_7x_v1_9_i/pipe_rx3_char_is_k_gt	00	00	10	00				10	00	
RX Other Signals		(RX Other Signals)								
Signal Set-6, GT TX		(Signal Set-6, GT TX)								
Signal Set-6, GT TX		(Signal Set-6, GT TX)								
TX DATA		(TX DATA)								
...d/EP/pcie_7x_v1_9_i/pipe_tx0_data_gt	4a4a	ff00	0006	4a4a				00bc	ff00	0006
...pcie_7x_v1_9_i/pipe_tx0_char_is_k_gt	00	00						01	00	
...d/EP/pcie_7x_v1_9_i/pipe_tx1_data_gt	4a4a	ff01	0006	4a4a				00bc	ff01	0006
...pcie_7x_v1_9_i/pipe_tx1_char_is_k_gt	00	00						01	00	
...d/EP/pcie_7x_v1_9_i/pipe_tx2_data_gt	4a4a	ff02	0006	4a4a				00bc	ff02	0006
...pcie_7x_v1_9_i/pipe_tx2_char_is_k_gt	00	00						01	00	
...d/EP/pcie_7x_v1_9_i/pipe_tx3_data_gt	4a4a	ff03	0006	4a4a				00bc	ff03	0006
...pcie_7x_v1_9_i/pipe_tx3_char_is_k_gt	00	00						01	00	
TX Other Signals		(TX Other Signals)								

Figure 35 - RP and EP both sending same link and lane numbers on corresponding lanes

Signal Set-1, LTSSM		(Signal Set-1, LTSSM)								
.../board/EP/pcie_7x_v1_9_i/pl_ltssm_state	13	10 13								
Signal Set-5, GT RX		(Signal Set-5, GT RX)								
...pe_wrapper_i/PIPE_RXCHANISALIGNED	1111	1111								
RX DATA		(RX DATA)								
...d/EP/pcie_7x_v1_9_i/pipe_rx0_data_gt	4545	4545	bc45	0000	46ff	4500	4545	bc45	0000	
...pcie_7x_v1_9_i/pipe_rx0_char_is_k_gt	00	00	10	00				10	00	
...d/EP/pcie_7x_v1_9_i/pipe_rx1_data_gt	4545	4545	bc45	0100	46ff	4500	4545	bc45	0100	
...pcie_7x_v1_9_i/pipe_rx1_char_is_k_gt	00	00	10	00				10	00	
...d/EP/pcie_7x_v1_9_i/pipe_rx2_data_gt	4545	4545	bc45	0200	46ff	4500	4545	bc45	0200	
...pcie_7x_v1_9_i/pipe_rx2_char_is_k_gt	00	00	10	00				10	00	
...d/EP/pcie_7x_v1_9_i/pipe_rx3_data_gt	4545	4545	bc45	0300	46ff	4500	4545	bc45	0300	
...pcie_7x_v1_9_i/pipe_rx3_char_is_k_gt	00	00	10	00				10	00	
RX Other Signals		(RX Other Signals)								
Signal Set-6, GT TX		(Signal Set-6, GT TX)								
Signal Set-6, GT TX		(Signal Set-6, GT TX)								
TX DATA		(TX DATA)								
...d/EP/pcie_7x_v1_9_i/pipe_tx0_data_gt	4545	ff00	0006	4a4a				00bc	ff00	0046
...pcie_7x_v1_9_i/pipe_tx0_char_is_k_gt	00	00						01	00	
...d/EP/pcie_7x_v1_9_i/pipe_tx1_data_gt	4545	ff01	0006	4a4a				00bc	ff01	0046
...pcie_7x_v1_9_i/pipe_tx1_char_is_k_gt	00	00						01	00	
...d/EP/pcie_7x_v1_9_i/pipe_tx2_data_gt	4545	ff02	0006	4a4a				00bc	ff02	0046
...pcie_7x_v1_9_i/pipe_tx2_char_is_k_gt	00	00						01	00	
...d/EP/pcie_7x_v1_9_i/pipe_tx3_data_gt	4545	ff03	0006	4a4a				00bc	ff03	0046
...pcie_7x_v1_9_i/pipe_tx3_char_is_k_gt	00	00						01	00	
TX Other Signals		(TX Other Signals)								

Figure 36 – EP sending '00' in link number field and corresponding lane numbers in lane number field.

After CONFIGURATION state, the next state is the normal working state, which is L0. The initial phase of link training completes after user_ink_up is asserted.

Advanced GT Debugging for PCIe Link Training

This section lists different groups of signals for Advanced GT debugging. If you have come to this point and the issue is still not resolved, capture signals mentioned in this section in Chipscope/Vivado ILA and attach the waveforms to the webcase you create with Xilinx Technical Support.

PIPE Wrapper Parameters

The wrappers that come with the generation of the core should be used as is, without any modification. If you have changed some wrapper parameters during the debug or due to some other reasons, please verify you have the default value for the parameters listed in Table 4.

Table 4 – Pipe Wrapper Parameters

Wrapper Parameters	GTX Default
PCIE_SIM_MODE	"FALSE"
PCIE_SIM-TX_EIDLE_DRIVE_LEVEL	"1"
PCIE_GT_DEVICE	"GTX"
PCIE_USE_MODE	"3.0"
PCIE_PLL_SEL	"CPLL"
PCIE_LPM_DFE	"LPM"
PCIE_EXT_CLK	"FALSE"
PCIE_POWER_SAVING	"TRUE"
PCIE_ASYNC_EN	"FALSE"
PCIE_TXBUF_EN	"FALSE"
PCIE_RXBUF_EN	"TRUE"
PCIE_CHAN_BOND	0
PCIE_CHAN_BOND_EN	"TRUE"
PCIE_LANE	1
PCIE_LINK_SPEED	3
PCIE_REFCLK_FREQ	0
PCIE_USERCLK1_FREQ	2
PCIE_USERCLK2_FREQ	2
PCIE_OOBCLK_MODE	1
PCIE_DEBUG_MODE	0

PIPE Wrapper Idle Indicator

Most of the signals listed in Table 5 have been discussed in previous sections. When capturing Idle Indicator and FSM signals, capture other signals listed in the table as well to make it easier for analysis.

Table 5 - PIPE Wrapper Debug Signals

Command	Data	Status
PIPE_TXDETECTRX	PIPE_RXDATA	PIPE_RXELECIDLE
PIPE_TXELECIDLE	PIPE_RXDATAK	PIPE_RXVALID
PIPE_TXCOMPLIANCE		PIPE_PHYSTATUS
PIPE_RXPOLARITY		PIPE_RXELECIDLE
PIPE_POWERDOWN		PIPE_RXSTATUS[2:0]
PIPE_RATE		PIPE_RXBUFSTATUS[2:0]
		PL_LTSSM_STATE [5:0]-

Lock Indicator	Status Indicator	Idle Indicator
PIPE_CPLL_LOCK	PIPE_TXSYNC_DONE	PIPE_RST_IDLE
PIPE_QPLL_LOCK	PIPE_RXSYNC_DONE	PIPE_QRST_IDLE
PIPE_PCLK_LOCK	PIPE_GEN3_RDY	PIPE_RATE_IDLE
PIPE_RXCDRLOCK	PIPE_RXCHANISALIGNED	
PIPE_RXCDRLOCK	PIP_ACTIVE_LANE	

PIPE Wrapper FSM

The PIPE Wrapper is in idle state when PIPE_RST_IDLE, PIPE_QRST_IDLE, and PIPE_RATE_IDLE are all HIGH. If any idle status is LOW, add the following Wrapper FSM ports to ChipScope/Vivado ILA.

Table 6 - Wrapper FSM Ports

Wrapper FSM	Guideline
PIPE_RST_FSM	Add to Chipscope if PIPE_RST_IDLE stuck at 0
PIPE_QRST_FSM	Add to Chipscope if PIPE_QRST_IDLE stuck at 0
PIPE_RATE_FSM	Add to Chipscope if PIPE_RATE_IDLE stuck at 0
PIPE_SYNC_FSM_TX	Add to Chipscope if PIPE_RST_FSM stuck at 11'b10000000000 or PIPE_RATE_FSM stuck at 24'b000100000000000000000000
PIPE_SYNC_FSM_RX	Not supported
PIPE_DRP_FSM	Add to Chipscope if PIPE_RATE_FSM stuck at 24'b00000000000000000000000100000000
PIPE_TXEQ_FSM	Not used for Gen1/Gen2
PIPE_RXEQ_FSM	Not used for Gen1/Gen2
PIPE_QDRP_FSM	Add to Chipscope if PIPE_QRST_FSM stuck at 12'b000001000000

GTX/GTP Wrapper Settings

The GTP/GTX wrapper is generated with all recommended settings. Although it is not recommended to change the default parameters, it might be necessary to do so during the debug procedure.

TXPOSTCURSOR, TXPRECURSOR, TXDIFFCTRL[3:0]

To reduce the effect of inter symbol interference, PCI express employs the concept of de-emphasis. Pre-emphasis and De-emphasis are basically the same. If five consecutive bits are transmitted with the same polarity, the bits after the first bit are de-emphasized compared to the first bit. In other words, the first bit is pre-emphasized compared to the rest of the four following bits.

In 7 series FPGA transceivers, the tap weights are all programmable to meet different channel conditions. GTX/GTH/GTP transceivers have 32 settings for post-tap de-emphasis (TXPOSTCURSOR), up to 12.96 dB, and 21 settings for pre-tap de-emphasis (TXPRECURSOR), up to 6.02 dB. Both TXPOSTCURSOR and TXPRECURSOR attributes work on the data transitions. To increase the signal strength (amplitude), change TXDIFFCTRL setting.

Figure 37 from WP419[4] shows the data stream without any de-emphasis. The symbols following the transition have a peak-to-peak amplitude of $\sim 0.28V$.

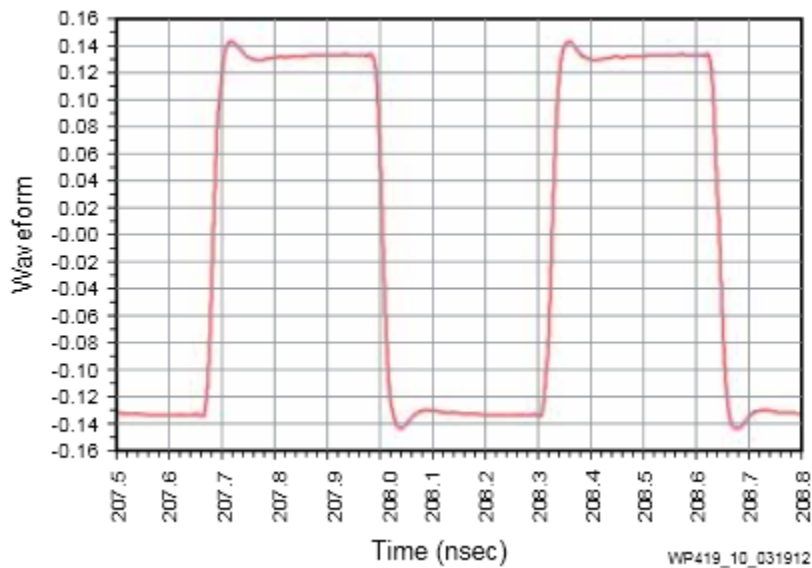


Figure 37 - 11110000 Binary Patter, No De-emphasis

Figure 38 shows the data stream with 6 dB of de-emphasis. The symbols following the transition now have a peak-to-peak amplitude of $\sim 0.14V$. The difference is because the de-emphasis tap weight applied to the data stream is 6 dB.

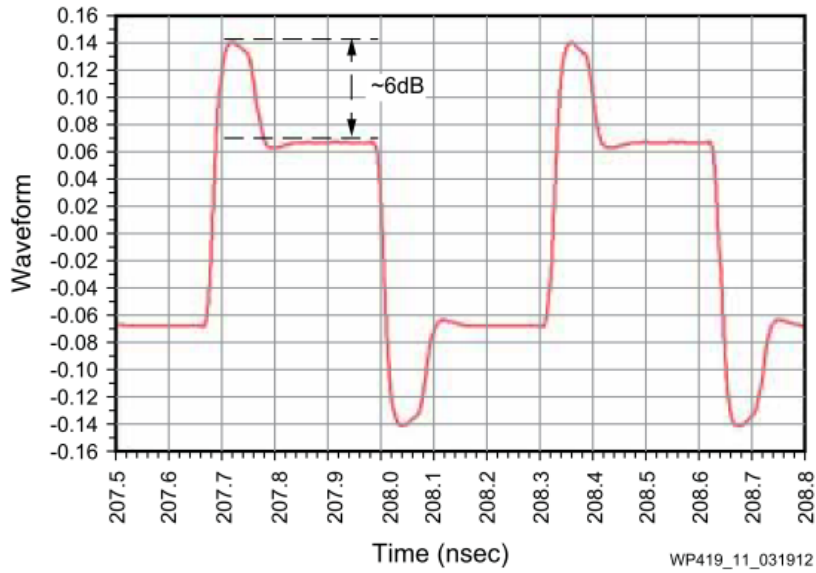


Figure 38 - 11110000 Binary Pattern, 6dB De-emphasis

Figure 37 and Figure 38 show the post-tap de-emphasis impact on the signal. The pre-tap de-emphasis is similar to post-tap. The difference is that the symbol that has the high swing is the one before the transition rather than after the transition. Figure 39 shows a simulation example of GTX transmitter output, how a 6 dB pre-tap de-emphasis can reshape a signal repeating 11110000.

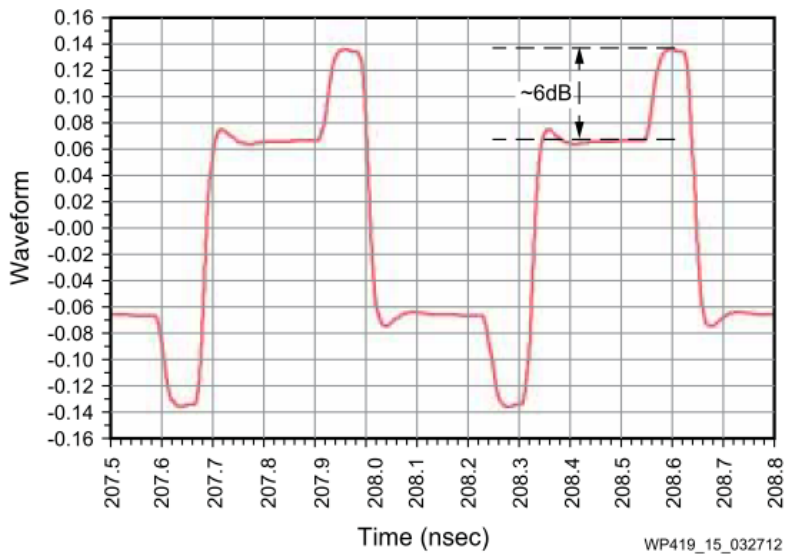


Figure 39 - 11110000 Binary Pattern, 6dB Pre-tap De-emphasis

Figure 40 and Figure 41 show the impact of applying 2 dB post-tap De-emphasis in the GTX Transceiver Eye Diagram.

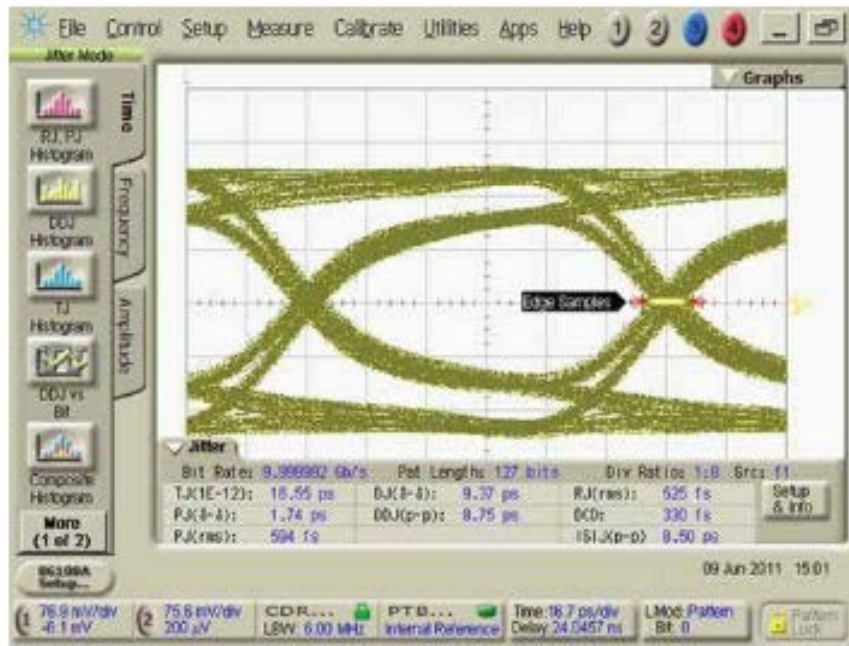


Figure 40 – GTX Transceiver Eye Diagram without De-emphasis

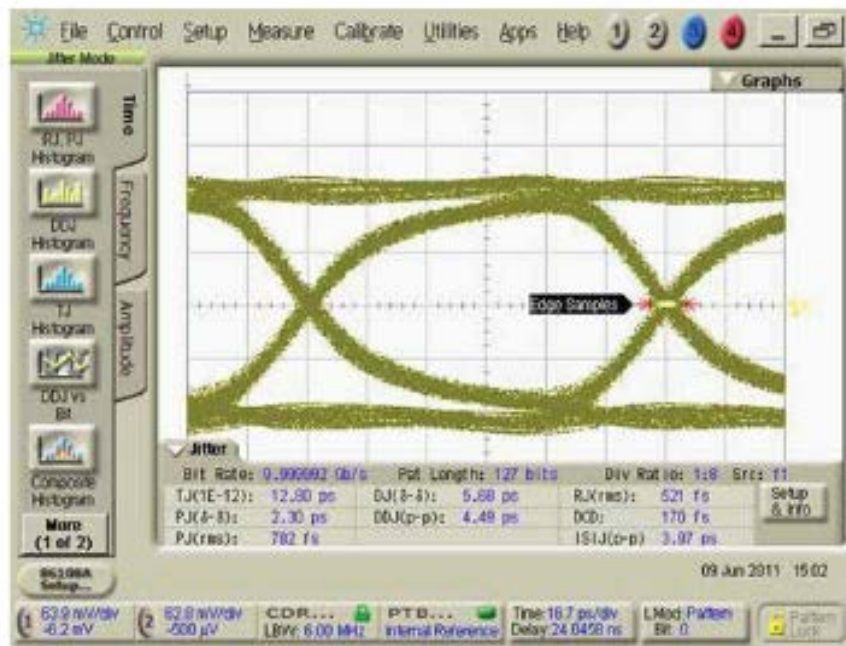


Figure 41 - GTX Transceiver Eye Diagram with 2dB Post-tap De-emphasis

Table 7, Table 8 and Table 9 are from UG476[2] and provide different values for TXDIFFCTRL, TXPOSTCURSOR, and TXPRECURSOR, respectively.

Table 7 - TXDIFFCTRL

Port	Dir	Clock Domain	Description																																		
TXDIFFCTRL[3:0]	In	TXUSRCLK2	<p>Driver Swing Control. The default is user specified. All listed values are in mV_{PPD}.</p> <table border="1"> <thead> <tr> <th>[3:0]</th> <th>mV_{PPD}</th> </tr> </thead> <tbody> <tr><td>4'b0000</td><td>250</td></tr> <tr><td>4'b0001</td><td>300</td></tr> <tr><td>4'b0010</td><td>350</td></tr> <tr><td>4'b0011</td><td>400</td></tr> <tr><td>4'b0100</td><td>450</td></tr> <tr><td>4'b0101</td><td>500</td></tr> <tr><td>4'b0110</td><td>550</td></tr> <tr><td>4'b0111</td><td>600</td></tr> <tr><td>4'b1000</td><td>650</td></tr> <tr><td>4'b1001</td><td>700</td></tr> <tr><td>4'b1010</td><td>750</td></tr> <tr><td>4'b1011</td><td>800</td></tr> <tr><td>4'b1100</td><td>850</td></tr> <tr><td>4'b1101</td><td>900</td></tr> <tr><td>4'b1110</td><td>950</td></tr> <tr><td>4'b1111</td><td>1000</td></tr> </tbody> </table> <p>Note: These are preliminary values. The peak-to-peak differential voltage is defined when TXPOSTCURSOR = 5'b00000 and TXPRECURSOR = 5'b00000.</p>	[3:0]	mV_{PPD}	4'b0000	250	4'b0001	300	4'b0010	350	4'b0011	400	4'b0100	450	4'b0101	500	4'b0110	550	4'b0111	600	4'b1000	650	4'b1001	700	4'b1010	750	4'b1011	800	4'b1100	850	4'b1101	900	4'b1110	950	4'b1111	1000
[3:0]	mV_{PPD}																																				
4'b0000	250																																				
4'b0001	300																																				
4'b0010	350																																				
4'b0011	400																																				
4'b0100	450																																				
4'b0101	500																																				
4'b0110	550																																				
4'b0111	600																																				
4'b1000	650																																				
4'b1001	700																																				
4'b1010	750																																				
4'b1011	800																																				
4'b1100	850																																				
4'b1101	900																																				
4'b1110	950																																				
4'b1111	1000																																				

Table 8 - TXPOSTCURSOR

Port	Dir	Clock Domain	Description																																																																																																			
TXPOSTCURSOR[4:0]	In	Async	Transmitter post-cursor TX pre-emphasis control. The default is user specified. All listed values (dB) are typical.																																																																																																			
			<table border="1"> <thead> <tr> <th>[4:0]</th> <th>Emphasis (dB)</th> <th> Coefficient Units </th> </tr> </thead> <tbody> <tr><td>5'b00000</td><td>0.00</td><td>0</td></tr> <tr><td>5'b00001</td><td>0.22</td><td>1</td></tr> <tr><td>5'b00010</td><td>0.45</td><td>2</td></tr> <tr><td>5'b00011</td><td>0.68</td><td>3</td></tr> <tr><td>5'b00100</td><td>0.92</td><td>4</td></tr> <tr><td>5'b00101</td><td>1.16</td><td>5</td></tr> <tr><td>5'b00110</td><td>1.41</td><td>6</td></tr> <tr><td>5'b00111</td><td>1.67</td><td>7</td></tr> <tr><td>5'b01000</td><td>1.94</td><td>8</td></tr> <tr><td>5'b01001</td><td>2.21</td><td>9</td></tr> <tr><td>5'b01010</td><td>2.50</td><td>10</td></tr> <tr><td>5'b01011</td><td>2.79</td><td>11</td></tr> <tr><td>5'b01100</td><td>3.10</td><td>12</td></tr> <tr><td>5'b01101</td><td>3.41</td><td>13</td></tr> <tr><td>5'b01110</td><td>3.74</td><td>14</td></tr> <tr><td>5'b01111</td><td>4.08</td><td>15</td></tr> <tr><td>5'b10000</td><td>4.44</td><td>16</td></tr> <tr><td>5'b10001</td><td>4.81</td><td>17</td></tr> <tr><td>5'b10010</td><td>5.19</td><td>18</td></tr> <tr><td>5'b10011</td><td>5.60</td><td>19</td></tr> <tr><td>5'b10100</td><td>6.02</td><td>20</td></tr> <tr><td>5'b10101</td><td>6.47</td><td>21</td></tr> <tr><td>5'b10110</td><td>6.94</td><td>22</td></tr> <tr><td>5'b10111</td><td>7.43</td><td>23</td></tr> <tr><td>5'b11000</td><td>7.96</td><td>24</td></tr> <tr><td>5'b11001</td><td>8.52</td><td>25</td></tr> <tr><td>5'b11010</td><td>9.12</td><td>26</td></tr> <tr><td>5'b11011</td><td>9.76</td><td>27</td></tr> <tr><td>5'b11100</td><td>10.46</td><td>28</td></tr> <tr><td>5'b11101</td><td>11.21</td><td>29</td></tr> <tr><td>5'b11110</td><td>12.04</td><td>30</td></tr> <tr><td>5'b11111</td><td>12.96</td><td>31</td></tr> </tbody> </table>	[4:0]	Emphasis (dB)	Coefficient Units	5'b00000	0.00	0	5'b00001	0.22	1	5'b00010	0.45	2	5'b00011	0.68	3	5'b00100	0.92	4	5'b00101	1.16	5	5'b00110	1.41	6	5'b00111	1.67	7	5'b01000	1.94	8	5'b01001	2.21	9	5'b01010	2.50	10	5'b01011	2.79	11	5'b01100	3.10	12	5'b01101	3.41	13	5'b01110	3.74	14	5'b01111	4.08	15	5'b10000	4.44	16	5'b10001	4.81	17	5'b10010	5.19	18	5'b10011	5.60	19	5'b10100	6.02	20	5'b10101	6.47	21	5'b10110	6.94	22	5'b10111	7.43	23	5'b11000	7.96	24	5'b11001	8.52	25	5'b11010	9.12	26	5'b11011	9.76	27	5'b11100	10.46	28	5'b11101	11.21	29	5'b11110	12.04	30	5'b11111	12.96	31
			[4:0]	Emphasis (dB)	Coefficient Units																																																																																																	
			5'b00000	0.00	0																																																																																																	
			5'b00001	0.22	1																																																																																																	
			5'b00010	0.45	2																																																																																																	
			5'b00011	0.68	3																																																																																																	
			5'b00100	0.92	4																																																																																																	
			5'b00101	1.16	5																																																																																																	
			5'b00110	1.41	6																																																																																																	
			5'b00111	1.67	7																																																																																																	
			5'b01000	1.94	8																																																																																																	
			5'b01001	2.21	9																																																																																																	
			5'b01010	2.50	10																																																																																																	
			5'b01011	2.79	11																																																																																																	
			5'b01100	3.10	12																																																																																																	
			5'b01101	3.41	13																																																																																																	
			5'b01110	3.74	14																																																																																																	
			5'b01111	4.08	15																																																																																																	
			5'b10000	4.44	16																																																																																																	
			5'b10001	4.81	17																																																																																																	
			5'b10010	5.19	18																																																																																																	
			5'b10011	5.60	19																																																																																																	
			5'b10100	6.02	20																																																																																																	
			5'b10101	6.47	21																																																																																																	
			5'b10110	6.94	22																																																																																																	
			5'b10111	7.43	23																																																																																																	
			5'b11000	7.96	24																																																																																																	
			5'b11001	8.52	25																																																																																																	
			5'b11010	9.12	26																																																																																																	
			5'b11011	9.76	27																																																																																																	
			5'b11100	10.46	28																																																																																																	
5'b11101	11.21	29																																																																																																				
5'b11110	12.04	30																																																																																																				
5'b11111	12.96	31																																																																																																				
			<p>Note: These are preliminary values. The TXPOSTCURSOR values are defined when the TXPRECURSOR = 5'b00000</p> $\text{Emphasis} = 20\log_{10}(V_{\text{high}}/V_{\text{low}}) = 20\log_{10}(V_{\text{low}}/V_{\text{high}}) $																																																																																																			

Table 9 - TXPRECURSOR

Port	Dir	Clock Domain	Description																																																																																																			
TXPRECURSOR[4:0]	In	Async	Transmitter pre-cursor TX pre-emphasis control. The default is user specified. All listed values (dB) are typical.																																																																																																			
			<table border="1"> <thead> <tr> <th>[4:0]</th> <th>Emphasis (dB)</th> <th>Coefficient Units</th> </tr> </thead> <tbody> <tr><td>5'b00000</td><td>0.00</td><td>0</td></tr> <tr><td>5'b00001</td><td>0.22</td><td>1</td></tr> <tr><td>5'b00010</td><td>0.45</td><td>2</td></tr> <tr><td>5'b00011</td><td>0.68</td><td>3</td></tr> <tr><td>5'b00100</td><td>0.92</td><td>4</td></tr> <tr><td>5'b00101</td><td>1.16</td><td>5</td></tr> <tr><td>5'b00110</td><td>1.41</td><td>6</td></tr> <tr><td>5'b00111</td><td>1.67</td><td>7</td></tr> <tr><td>5'b01000</td><td>1.94</td><td>8</td></tr> <tr><td>5'b01001</td><td>2.21</td><td>9</td></tr> <tr><td>5'b01010</td><td>2.50</td><td>10</td></tr> <tr><td>5'b01011</td><td>2.79</td><td>11</td></tr> <tr><td>5'b01100</td><td>3.10</td><td>12</td></tr> <tr><td>5'b01101</td><td>3.41</td><td>13</td></tr> <tr><td>5'b01110</td><td>3.74</td><td>14</td></tr> <tr><td>5'b01111</td><td>4.08</td><td>15</td></tr> <tr><td>5'b10000</td><td>4.44</td><td>16</td></tr> <tr><td>5'b10001</td><td>4.81</td><td>17</td></tr> <tr><td>5'b10010</td><td>5.19</td><td>18</td></tr> <tr><td>5'b10011</td><td>5.60</td><td>19</td></tr> <tr><td>5'b10100</td><td>6.02</td><td>20</td></tr> <tr><td>5'b10101</td><td>6.02</td><td>20</td></tr> <tr><td>5'b10110</td><td>6.02</td><td>20</td></tr> <tr><td>5'b10111</td><td>6.02</td><td>20</td></tr> <tr><td>5'b11000</td><td>6.02</td><td>20</td></tr> <tr><td>5'b11001</td><td>6.02</td><td>20</td></tr> <tr><td>5'b11010</td><td>6.02</td><td>20</td></tr> <tr><td>5'b11011</td><td>6.02</td><td>20</td></tr> <tr><td>5'b11100</td><td>6.02</td><td>20</td></tr> <tr><td>5'b11101</td><td>6.02</td><td>20</td></tr> <tr><td>5'b11110</td><td>6.02</td><td>20</td></tr> <tr><td>5'b11111</td><td>6.02</td><td>20</td></tr> </tbody> </table>	[4:0]	Emphasis (dB)	Coefficient Units	5'b00000	0.00	0	5'b00001	0.22	1	5'b00010	0.45	2	5'b00011	0.68	3	5'b00100	0.92	4	5'b00101	1.16	5	5'b00110	1.41	6	5'b00111	1.67	7	5'b01000	1.94	8	5'b01001	2.21	9	5'b01010	2.50	10	5'b01011	2.79	11	5'b01100	3.10	12	5'b01101	3.41	13	5'b01110	3.74	14	5'b01111	4.08	15	5'b10000	4.44	16	5'b10001	4.81	17	5'b10010	5.19	18	5'b10011	5.60	19	5'b10100	6.02	20	5'b10101	6.02	20	5'b10110	6.02	20	5'b10111	6.02	20	5'b11000	6.02	20	5'b11001	6.02	20	5'b11010	6.02	20	5'b11011	6.02	20	5'b11100	6.02	20	5'b11101	6.02	20	5'b11110	6.02	20	5'b11111	6.02	20
			[4:0]	Emphasis (dB)	Coefficient Units																																																																																																	
			5'b00000	0.00	0																																																																																																	
			5'b00001	0.22	1																																																																																																	
			5'b00010	0.45	2																																																																																																	
			5'b00011	0.68	3																																																																																																	
			5'b00100	0.92	4																																																																																																	
			5'b00101	1.16	5																																																																																																	
			5'b00110	1.41	6																																																																																																	
			5'b00111	1.67	7																																																																																																	
			5'b01000	1.94	8																																																																																																	
			5'b01001	2.21	9																																																																																																	
			5'b01010	2.50	10																																																																																																	
			5'b01011	2.79	11																																																																																																	
			5'b01100	3.10	12																																																																																																	
			5'b01101	3.41	13																																																																																																	
			5'b01110	3.74	14																																																																																																	
			5'b01111	4.08	15																																																																																																	
			5'b10000	4.44	16																																																																																																	
			5'b10001	4.81	17																																																																																																	
			5'b10010	5.19	18																																																																																																	
			5'b10011	5.60	19																																																																																																	
			5'b10100	6.02	20																																																																																																	
			5'b10101	6.02	20																																																																																																	
			5'b10110	6.02	20																																																																																																	
			5'b10111	6.02	20																																																																																																	
			5'b11000	6.02	20																																																																																																	
			5'b11001	6.02	20																																																																																																	
			5'b11010	6.02	20																																																																																																	
5'b11011	6.02	20																																																																																																				
5'b11100	6.02	20																																																																																																				
5'b11101	6.02	20																																																																																																				
5'b11110	6.02	20																																																																																																				
5'b11111	6.02	20																																																																																																				
<p>Note: These are preliminary values. The TXPRECURSOR values are defined when the TXPOSTCURSOR = 5'b00000 $Emphasis = 20\log_{10}(V_{high}/V_{low}) = 20\log_{10}(V_{low}/V_{high})$</p>																																																																																																						

RXOOB_CFG

During link training, if rxelecidle shows unexpected behavior, tune the RXOOB_CFG parameter in the GT wrapper. In the generated wrapper, it is commented out. Uncomment this parameter setting and tune the parameter to suit your system. In some boards, it has been observed that changing RXOOB_CFG from 7'b0000110 to 7'b0000010 fixed an incorrect

assertion of rxeleidle in one of the lanes, and hence fixes the entire link training issue where it was incorrectly training down after multiple resets.

TX_RXDETECT_REF

The default value of TX_RXDETECT_REF parameter is 011. This value should work without any issue. In cases where the link training is running into receive detect issues, test with different values (e.g. 010, 100). It is not recommended to set a different value for this parameter other than the default value. If other values work and the default value does not, please contact Xilinx Technical Support before using the non-default value in your design.

LPM/DFE

The PCIe wrapper uses LPM mode by default. DFE mode is recommended for medium- to long-reach applications, with channel losses of 8 dB and above at the Nyquist frequency. A DFE has the advantage of equalizing a channel without amplifying noise and crosstalk. In case of severe link training issues, try with DFE mode instead of LPM.

Port	Dir	Clock Domain	Description
RXLPMEN	In	RXUSRCLK2	RX datapath 0: DFE 1: LPM

Figure 42 – LPM/DFE mode selection port

RXBUFSTATUS

Check RXBUFSTATUS[2:0] port from GTs to see if the buffer underflows (3'b101) or overflows (3'b110). During “normal” operation, there should not be any underflows/overflows. If this is seen on RX GT’s, check if the link partner device is sending the clock compensation sequences as it should be and if GTs are actually adjusting the RX Elastic Buffer pointers to correct for bit rate differences. Check the RXCLKCORCNT[2:0] bus from the GTs to see if the GT has performed clock correction. Also, check the RXDATA and RXCHARISK signals from the GT to see if there is clock compensation sequence (SKP ordered set). If RXCLKCORCNT indicates the GT has performed clock correction, it is likely that SKP ordered set will not be received on the RXDATA interface since the GT will have had to add or remove characters as part of the correction.

Debugging Channel Bonding issue

Channel bonding is used by protocols to transmit data over multiple lanes. PCIe uses channel bonding over multiple lanes, so there is a chance that due to variation in PCB trace lengths, or other factors when the data is received, it may no longer be perfectly aligned. Channel bonding realigns the data by adjusting the RX BUFFER FIFO read pointers. On the left in Figure 43, it shows the aligned data RRRR coming out of the transmitter and due to various system level electrical effects (like tracelength, etc.), there can be a skew introduced when the data is captured in the receiver, so the original RRRR data can be received as RSQR as shown in the middle section of Figure 43. The PCS section of the GTs adjusts the read pointers in the RX Buffer FIFOs and realigns the data as RRRR.

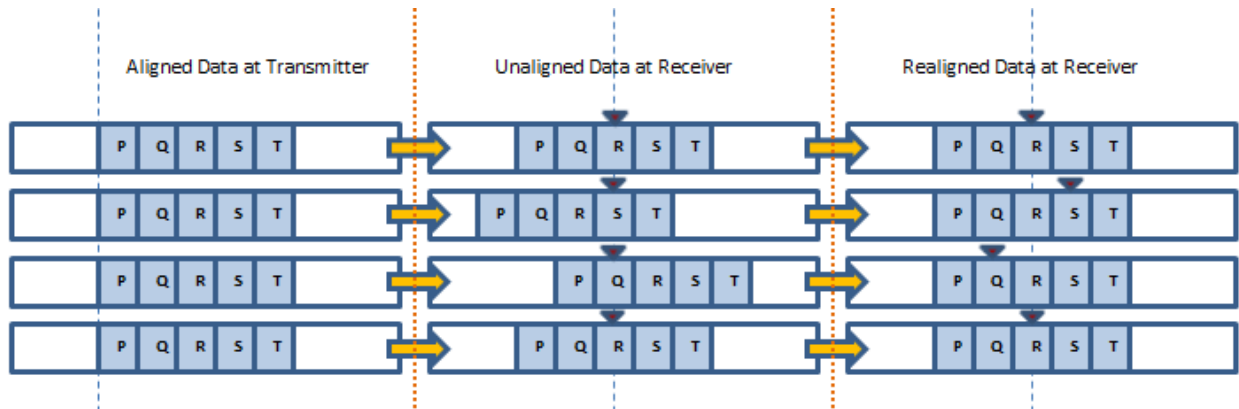


Figure 43 - Channel Bonding

The channel bonding is done using a channel bond sequence, which is identified by the master lane and then triggered through its RXCHBONDO ports to the slaves' RXCHBONDI ports to search for the bonding sequence within the slave transceivers RX FIFO buffers and adjust the read pointers accordingly.

To provide enough time for the slave to collect bytes for bonding sequence `CHAN_BOND_MAX_SKEW` attribute is used. This attribute controls the number of `USRCLK` cycles that the master waits before ordering the slaves to execute channel bonding. This attribute determines the maximum skew that can be handled by channel bonding. It must always be less than one-half the minimum distance (in bytes or 10-bit codes) between channel bonding sequences. Valid values range from 1 to 14. More information on this is available in UG476. Ideally, this parameter should not be changed from the default value. However, based on the board and the interacting system, it might be required to make necessary tuning.

The maximum allowable distance between the channel bonding characters sets maximum skew `CHAN_BOND_x_MAX_SKEW` attribute.

One channel bonding character is 10-bit (8B/10B) and one bit equals one UI (unit interval = 1/line rate). If you run at a PCIe Gen1 line rate of 2.5 Gb/s (UI = 0.4 ns) and set the skew to 7 (default for `CHAN_BOND_x_MAX_SKEW` attribute), the calculated skew will be:

$$\text{Skew [ns]} = \text{UI} * \text{number_of_characters} * 10 = 0.4 \text{ ns} * 7 * 10 = 28 \text{ ns}.$$

So, if you are using PCIe over back plane or extender cards or any other system where there is a possibility of large skew, you may need to adjust `CHAN_BOND_x_MAX_SKEW` accordingly.

`CLK_COR_MIN_LAT` is another parameter you could tune if you run into channel bonding issue indicated by the de-assertion of `RXCHANISALIGNED` signal. When using channel bonding, you add the additional requirement that the buffer needs to have head room to see the skewed channel bonding sequences. By increasing `CLK_COR_MIN_LAT`, you buy a little more room to allow for greater skew between lanes. If the link is heavily skewed, increasing the value of `CLK_COR_MIN_LAT` might help.

PCIe Signal Integrity (SI) Debug

Link training issue is most likely related to Signal Integrity(SI) on the board. SI debug is an extensive field where a SystemIO specialist might be required to be called in to debug SI related issues. In this section, a general SI debug guideline is provided. They are listed as follows:

- Make sure reference clocks are stable in both devices.
 - Check GT PLL lock signal (“**Signal Set-4, Clocking**”). It indicates if the GT is locked to the reference clock.
 - Use a scope to measure the reference clock frequency and jitter.
 - Make sure the reference clock is within the phase noise limits as discussed in [\(Xilinx Answer 44549\)](#).

- Run a general SI check analysis:
 - Run some basic SI simulation.
 - If the channel is chip to chip with only 3-4 inches of trace length, then most likely the SI is fine. The general procedure is to run SI simulation to determine a ball park Emphasis settings.
 - Run IBERT as described “*In System Eye Scan*”.
 - Probe the eye diagram. Adjust the preemphasis settings (in Section – “*TXPOSTCURSOR, TXPRECURSOR, TXDIFFCTRL[3:0]*”) and see the eye change in scope. Basically, you have to figure out the best settings on both ends to get a reliable link.
 - PCI Express Card Electromechanical Specification, Rev2.0 Section 4.7, describes requirements for Eye Diagrams at the add-in Card Interface that must be met for both the add-in card and a system board interfacing with such an add-in card.
 - For Eye capture, solder down diff. probes at the receive VIA. A high sampling scope must be used. Capture the eye and see the quality. Apply PCIe mask to see if it meets PCIe specification requirements. Users should make sure jitter characteristics are met as provided in Table 10, taken from [DS182](#).

Table 10 - GTX Transceiver PCI Express Jitter Characteristics ([DS182](#))

Standard	Description	Line Rate (Mb/s)	Min	Max	Units	
PCI Express Transmitter Jitter Generation						
PCI Express Gen 1	Total transmitter jitter	2500	–	0.25	UI	
PCI Express Gen 2	Total transmitter jitter	5000	–	0.25	UI	
PCI Express Gen 3 ⁽²⁾	Total transmitter jitter uncorrelated	8000	–	31.25	ps	
	Deterministic transmitter jitter uncorrelated		–	12	ps	
PCI Express Receiver High Frequency Jitter Tolerance						
PCI Express Gen 1	Total receiver jitter tolerance	2500	0.65	–	UI	
PCI Express Gen 2 ⁽³⁾	Receiver inherent timing error	5000	0.40	–	UI	
	Receiver inherent deterministic timing error		0.30	–	UI	
PCI Express Gen 3 ⁽²⁾	Receiver sinusoidal jitter tolerance	8000	0.03 MHz–1.0 MHz	–	UI	
			1.0 MHz–10 MHz	Note 4	–	UI
			10 MHz–100 MHz	0.10	–	UI

Notes:

1. Tested per card electromechanical (CEM) methodology.
2. PCI-SIG 3.0 certification and compliance test boards are currently not available.
3. Using common REFCLK.
4. Between 1 MHz and 10 MHz the minimum sinusoidal jitter roll-off with a slope of 20dB/decade.

- Check RXSTATUS (see Section – “*Signal Set-5, GT RX*”) to see if it reports any error.
 - The reported error could be due to number of reasons mentioned below:
 - One of the reference clocks has incorrect frequency in an asynchronous link.
 - Excessive jitter on the reference clock.
 - Excessive power supply noise on GT power supplies.
- It is important to check the Power supply. Check if the the correct voltage has been applied or not as shown in Table 11. Measure the power voltage to make sure there are no periodical spikes of noise that cause intermittent bit errors.

Table 11 – GTX Transceiver Voltage Requirements (DS182)

GTX Transceiver					
V _{MGTAVCC} ⁽⁸⁾	Analog supply voltage for the GTX transceiver QPLL frequency range ≤ 10.3125 GHz ⁽⁹⁾⁽¹⁰⁾	0.97	1.0	1.08	V
	Analog supply voltage for the GTX transceiver QPLL frequency range > 10.3125 GHz	1.02	1.05	1.08	V
V _{MGTAVTT} ⁽⁸⁾	Analog supply voltage for the GTX transmitter and receiver termination circuits	1.17	1.2	1.23	V
V _{MGTVCCAUX} ⁽⁸⁾	Auxiliary analog QPLL voltage supply for the transceivers	1.75	1.80	1.85	V

8. Each voltage listed requires the filter circuit described in [UG476: 7 Series FPGAs GTX/GTH Transceiver User Guide](#).

9. For data rates ≤ 10.3125 Gb/s, V_{MGTAVCC} should be 1.0V ±3% for lower power consumption.

10. For lower power consumption, V_{MGTAVCC} should be 1.0V ±3% over the entire CPLL frequency range.

- GTs need dedicated power supplies and should not be shared with other digital supplies.

In-System Eye Scan

7 series transceivers have ability to perform non-destructive in-system eye scans while the link is up. Acquiring the eye scan data can be performed at all speeds and on multiple lanes simultaneously. The information from an eye scan can lead to critical information regarding the link and will accelerate the debug process.

Below is a list of scenarios where in-system eye scans provide valuable debug information:

- A link analyzer detects replay packets to the FPGA. This typically means the FPGA NAK'd a packet which can mean there was an LCRC error due to a bit flip.
- A marginal link going in and out of recovery under different environmental conditions
- A production system where only a few boards exhibit link failures.
- A system down-trains in speed or lane width occasionally

Below is a list where eye scan data will not provide helpful debug information:

- If there is a suspicion that this is a transmit problem from the FPGA
 - When a link analyzer shows replay packets from the FPGA
- Going in and out of the detect state frequently
 - Eye scan data is statistical, and therefore requires a data stream where re-alignment of the transceiver is not happening

Overview of In-System Eye Scan Example Design

Implementing an eye scan on a PCI express link is very simple with the example design provided in Xilinx Answer Record 56648. This example uses a MicroBlaze processor to control the accesses to the DRP interface of the transceiver. MicroBlaze processor also manages the eye scan data by storing the data to Block RAM. Once the Block RAM fills up, XMD reads the data from Block RAM and stores it locally on the PC.

How to implement the Eye Scan Example design

Download the example designs from Xilinx Answer Record 56648 for the appropriate transceiver family of interest. For example, the KC705 example lends well for any GTX transceiver. Likewise, the VC709 lends well for the GTH transceiver. After downloading the example, the example will build a bitstream by sourcing the Tcl script in the 'pcie_eyescan/proj' directory.

Sourcing the Tcl script will generate a bit file for the evaluation board and it is ready to be programmed to the board. After the board is programmed, an XMD connection is required to extract the data from the FPGA. Connecting via XMD will require an XMD console. To get an XMD window in Linux, source the Vivado or ISE tools and type 'xmd' into the console. In Windows, click on XMD as shown in Figure 44.

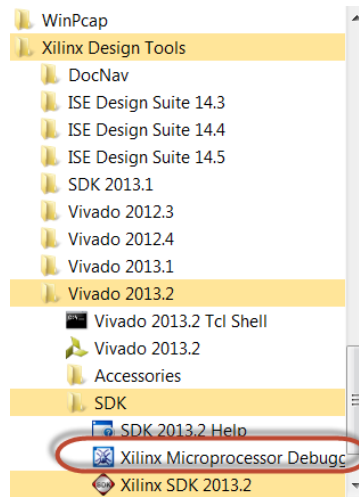


Figure 44 – Starting XMD

After the XMD console is open, change your present working directory to the 'tcl' directory provided in the zip file. Before executing the next Tcl scripts, make sure you have the FPGA programmed with the bit file, and also make sure the PC is turned on and you have an active link. Now execute the following Tcl commands in the XMD console:

- `connect mb mdm`
- `source get_eyescan_data.tcl`
- `run_test`

This will begin the eye scan and you will see the XMD console actively scrolling by as it is extracting the data. After the scans are completed, the eye scan data will be stored in the 'tcl' directory. The data is stored in the CSV files and they are labeled:

CH#_viv.csv; where the # is a value between 0 and 7.

To view the scan data open a new Vivado session. In the Tcl console of Vivado, change directories to the 'tcl' directory. Then source the `load_vivado_scans.tcl` file. This will show the eye scan data as shown in Figure 45.

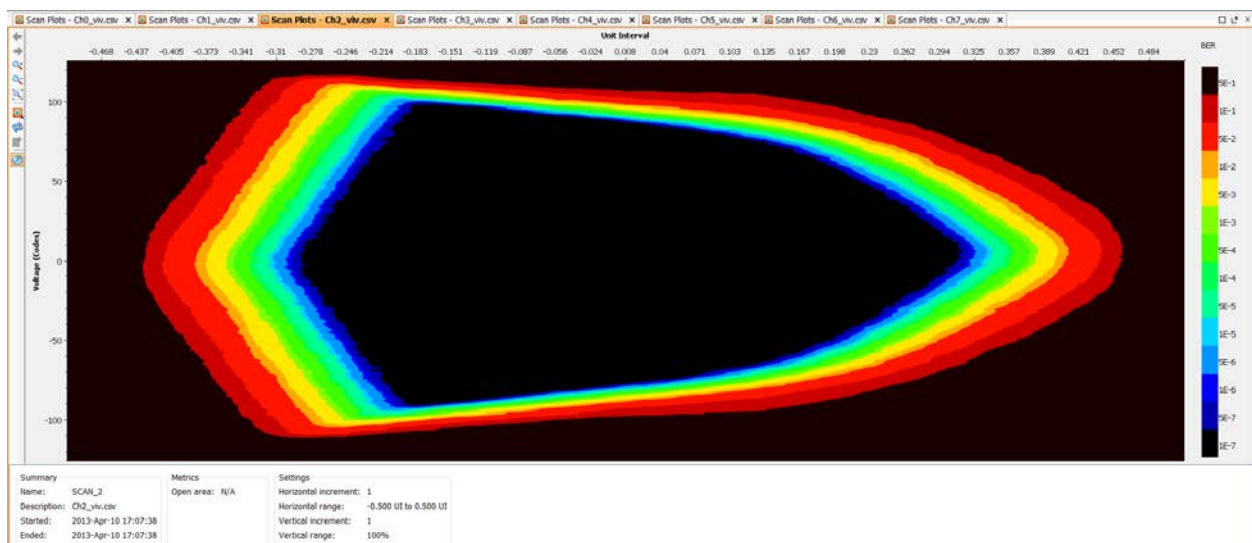


Figure 45 – Eye Scan Data

Using the 7 series eye scan feature with IBERT for PCI Express Debug

This 2D eye scan feature can be used through IBERT tool as well. As discussed in the previous section, Xilinx 7 series Transceivers have an inbuilt piece of hardware in them which is useful for RX margin analysis. This piece of hardware can be used to see the post equalization statistical eye (an external oscilloscope shows the eye before equalization on the transceiver pins) and can operate with any type of traffic without any pre-known pattern as it operates by comparison of the offset sample with the center sample, and counts the number of times it disagrees as an error. More information on hardware architecture and the process can be found in [UG476](#) in RX margin analysis section.

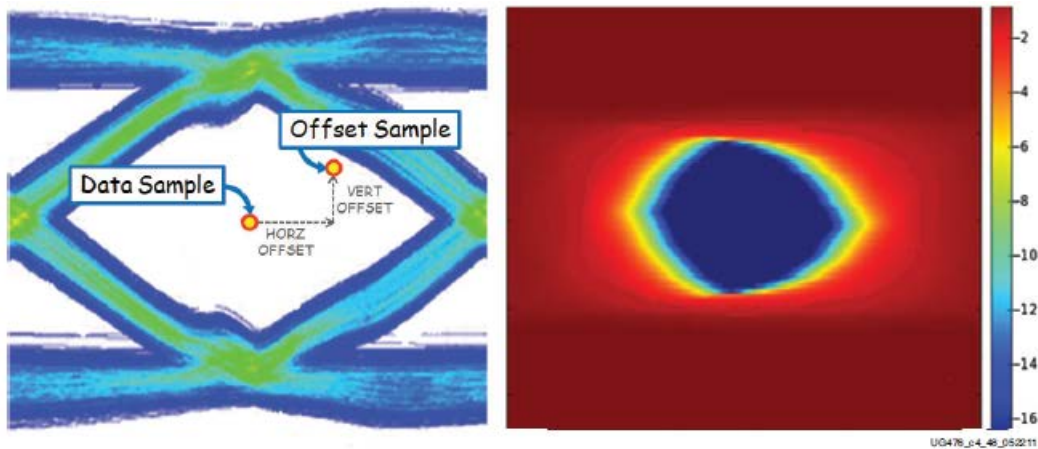


Figure 46 - Offset Sample and Data Sample to Calculate BER as a Function of Offset - Statistical Eye

Xilinx transceivers support the Far End PMA loopback mode which works by accepting the data from the link partner transceivers RX port and then putting it back to the TX port of the Xilinx transceiver

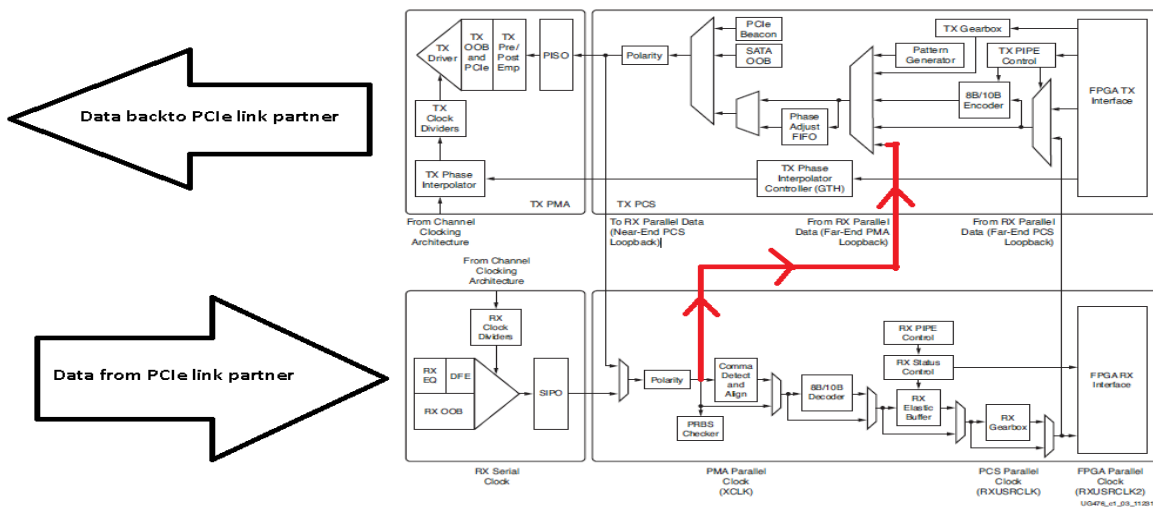


Figure 47 - Far End PMA Loopback of the data from the PCIe Link Partner

This loopback control can be done with the 3 bit loopback control port available on the transceivers. Xilinx IBERT (Integrated Bit Error Rate Tester) is a standalone design available from Xilinx core generator which can be used to control all the parameters of the Xilinx transceiver and can do the 2D eye scan for link debugging.

Figure 48 shows IBERT in the IP catalogue under 'Debug and Verification' tab.

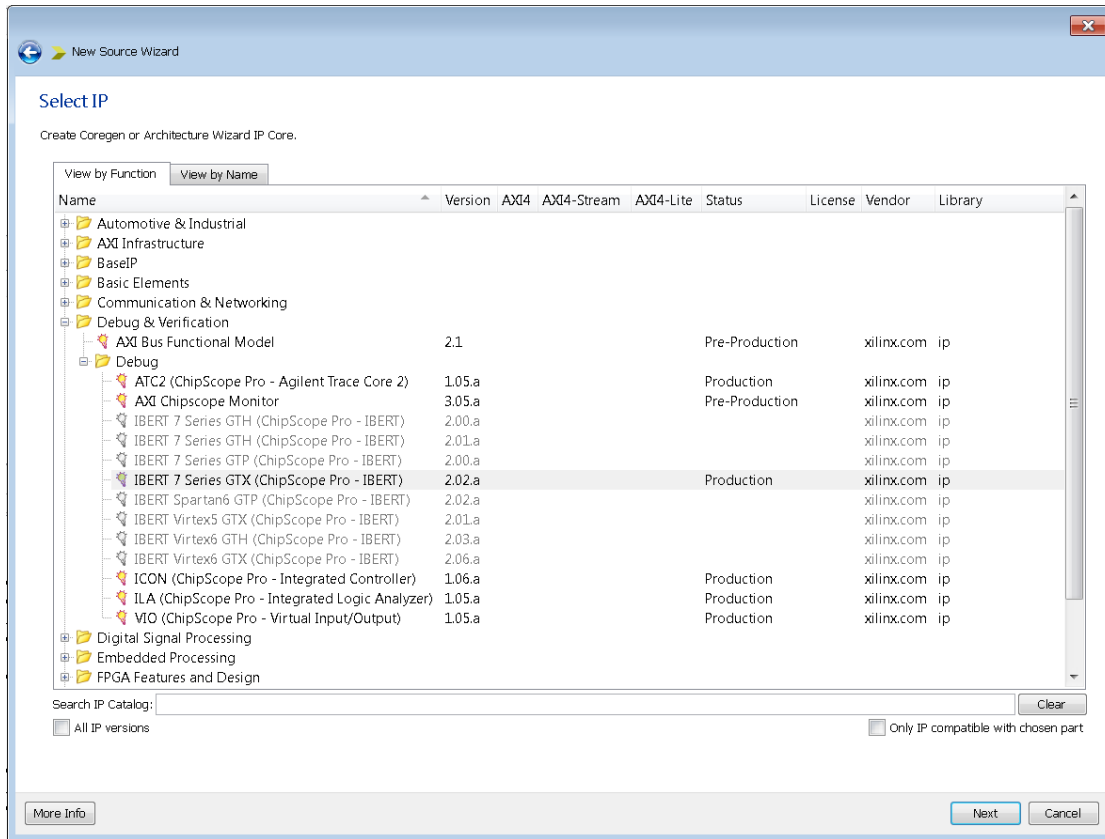


Figure 48 - 7-Series GTX (Chipscope Pro - IBERT)

The first screen of the IBERT IP configurator which allows selection of the naming style and the external clock source is shown in Figure 49. External clock source is optional and instead you can use the transceiver reference clock as the system clock used for running the logic in the standalone design.

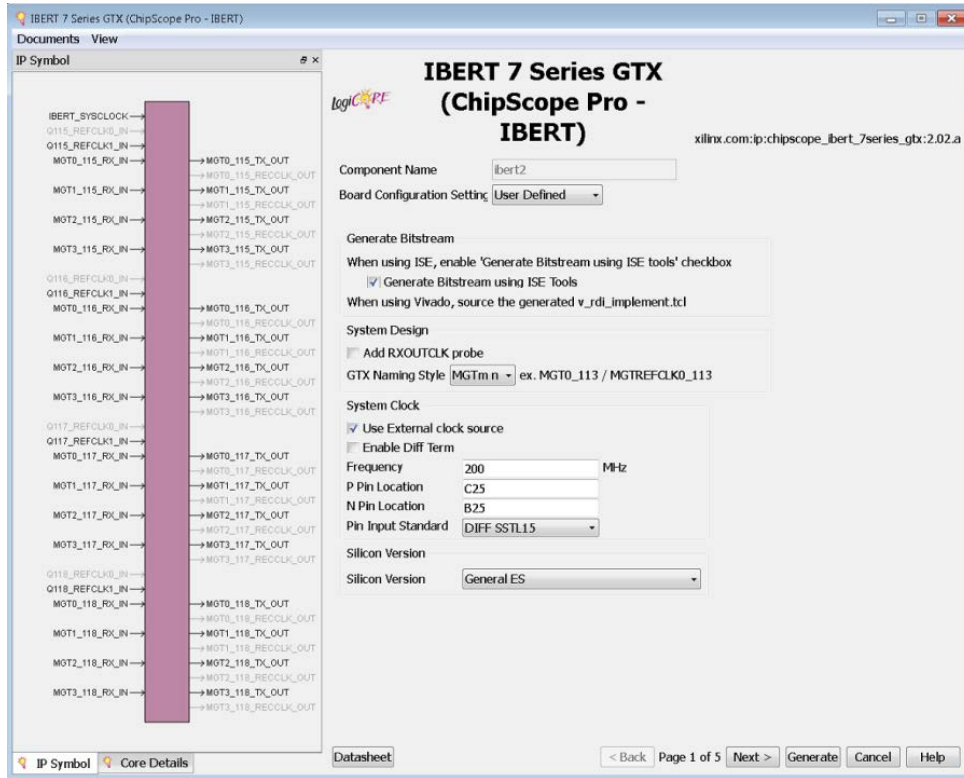


Figure 49 - IBERT IP Configurator - Naming Style and External Clock Source Selection

The second screen in Figure 50 allows choosing the line rate we intend to use, the transceiver at the reference clock frequency, and the quad we intend to test.

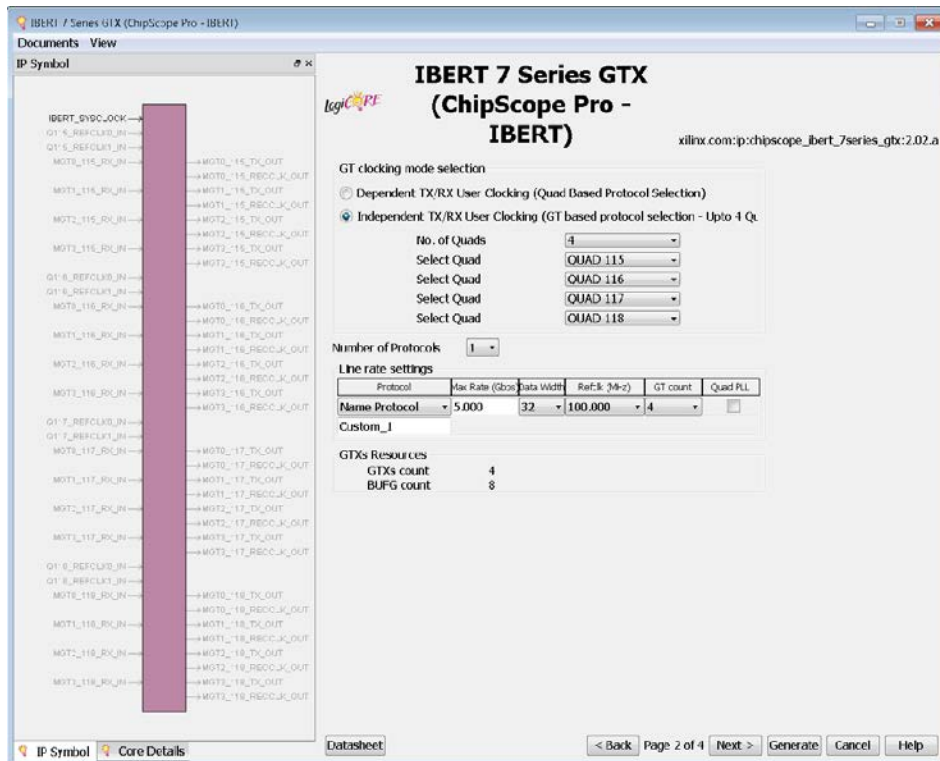


Figure 50 - IBERT IP Configurator - Line Rate and Quad Selection

The third screen, in Figure 51, allows selection of the transceivers that need to be tested in the design.

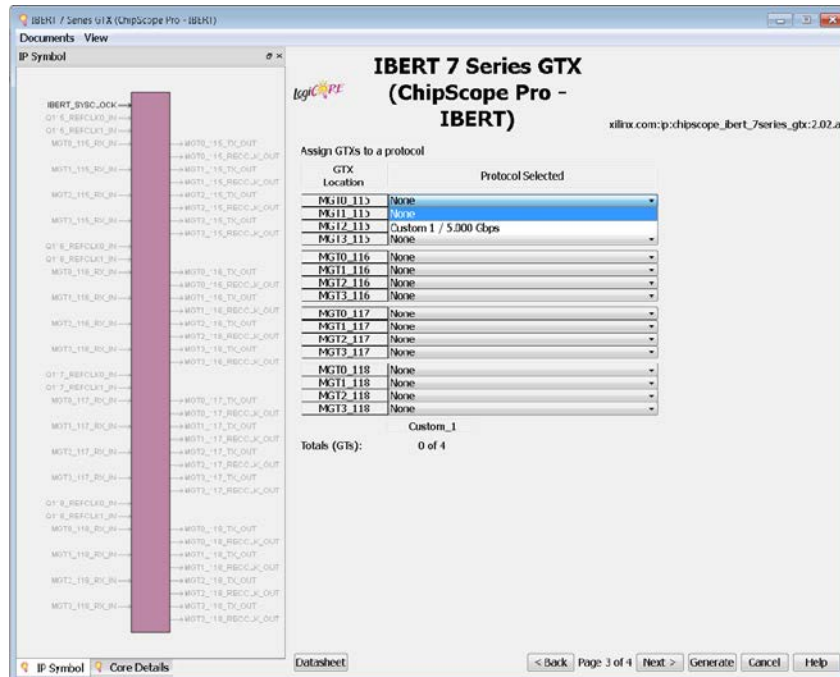


Figure 51 - IBERT IP Configurator - Transceiver Selection

The fourth screen displays the summary for a quick review before you generate the IP for use in your debug.

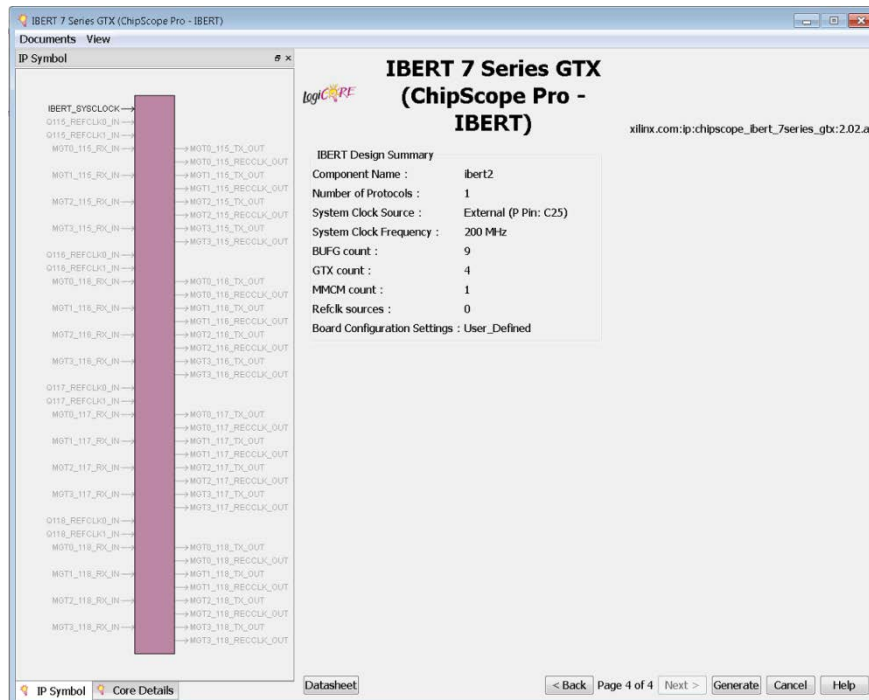


Figure 52 - IBERT IP Configurator - Core Summary

Once the IP bit stream is generated, you can download the design in your test system. Via IBERT console, the transceiver in the FPGA can be put in Far End PMA loopback so the link partner receives the data it is transmitting on the line. (In the Far End PMA loopback mode you will see no link in IBERT, this is an expected behavior)

IBERT Console - DEV:1 MyDevice1 (XC7K325T) UNIT:1_0 MyIBERT K7 GTX1_0 (IBERT K7 GTX)			
MGT/BERT Settings		DRP Settings	Port Settings
		RX Margin Analysis	
	GTX_X0Y0	GTX_X0Y1	
MGT Settings			
RX Common Mode	900 mV	900 mV	
MGT Alias	GTX0_115	GTX1_115	
Tile Location	GTX_X0Y0	GTX_X0Y1	
MGT Link Status	No Link	5.0 Gbps	
PLL Status	CPLL LOCKED	CPLL LOCKED	
Loopback Mode	Far-End FMA	None	
Channel Reset	Reset	Reset	
TX/RX Reset	TX Reset RX Reset	TX Reset RX Reset	
TX Polarity Invert	<input type="checkbox"/>	<input type="checkbox"/>	
TX Error Inject	Inject	Inject	
TX Diff Output Swing	850 mV (1100)	750 mV (1010)	
TX Pre-Cursor	1.67 dB (00111)	0.00 dB (00000)	
TX Post-Cursor	0.68 dB (00011)	0.00 dB (00000)	
RX Polarity Invert	<input type="checkbox"/>	<input type="checkbox"/>	
Termination Voltage	Programmable	Programmable	
BERT Settings			
BERT Reset	Reset	Reset	
TX Data Pattern	PRBS 31-bit	PRBS 7-bit	
RX Data Pattern	PRBS 31-bit	PRBS 7-bit	
RX Bit Error Ratio	5.036E-001	5.033E-014	
RX Received Bit Count	2.158E013	1.987E013	
RX Bit Error Count	1.087E013	0.000E000	
Clocking Settings			
TXUSRCLK Freq (MHz)	156.27	156.27	
TXUSRCLK2 Freq (MHz)	156.27	156.27	
RXUSRCLK Freq (MHz)	156.27	156.27	
RXUSRCLK2 Freq (MHz)	156.27	156.27	

Figure 53 - IBERT Console

The transceiver eyes can be then observed by doing the eye scan on them individually and seeing the results in the RX margin analysis tab. Before starting the tests, set the horizontal and vertical increment range from the drop down box. For faster / coarse eye scans you can select bigger jumps like 4 or 8 and a lower BER rate of 10 exp-6 or 10 exp-7. The amount of time spent in the scan goes higher with finer jumps like increment 1 and lower BER like 10 exp-9.

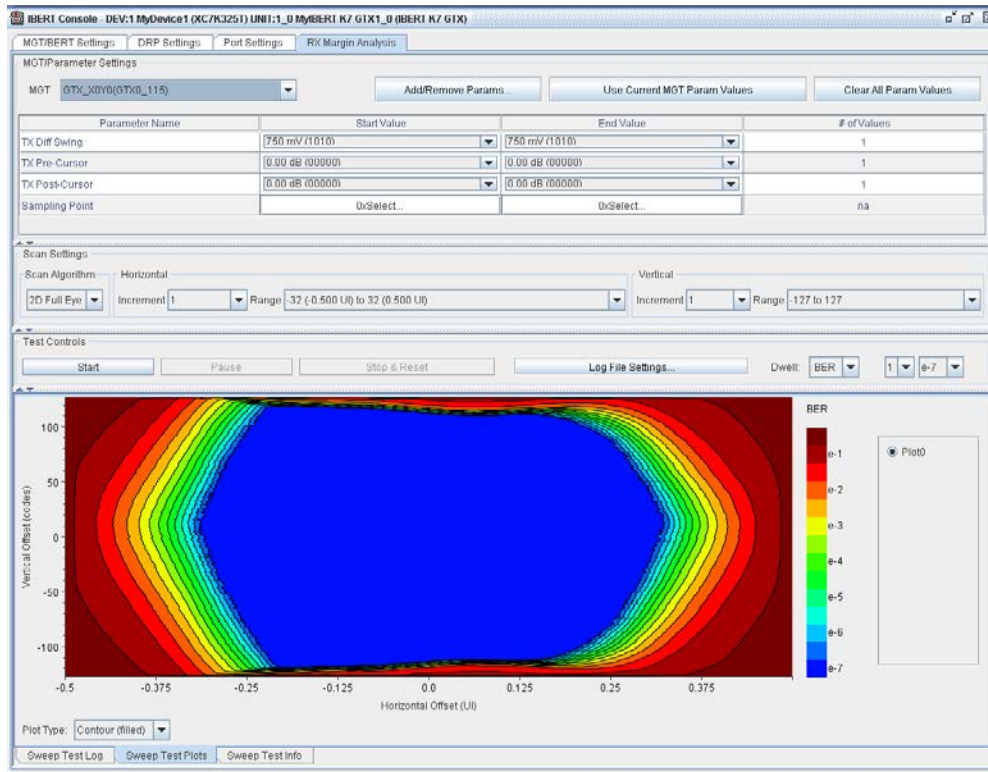


Figure 54 - RX Margin Analysis

This can then be used to observe and tune the eye with adjustment of GT attribute setting as described in “**GTX/GTP Wrapper Settings**” section.

Link Training Debug by Enabling Link Partner Debugging Features

Xilinx Endpoint link partner vendors may have PCIe debug features built in it which could be useful in debugging PCIe link training issues. One such example of a link partner is a PCIe PLX chip.

PLX chips have loopback and PRBS counter features built into their transceivers. They also have PLX *visionpak* debug software, similar to IBERT, which is used for transceiver eye capture without use of an external scope.

The loopback feature can be used by enabling the PRBS counters in the Xilinx transceiver and doing an external TX loopback in the PLX chip. The pattern will be sent back to the Xilinx endpoint transceiver RX pattern checker. This could be used to tune the link parameters as described in section – “**GTX/GTP Wrapper Settings**”.

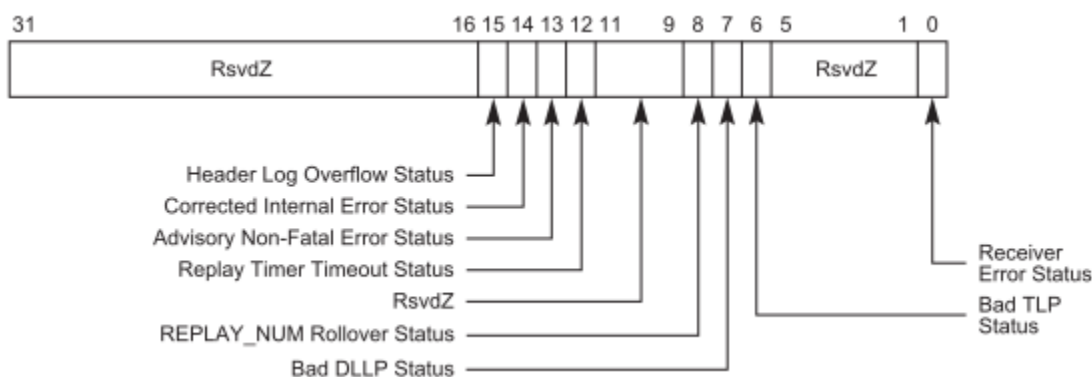
Similarly, the Xilinx Transceiver PMA Far End loopback can be used for testing with PRBS counters and checkers built in SerDes of the PLX transceivers and accessed through register read and write on PLX chips.

While debugging link training issues, users should explore debug capabilities available in the link partner device and how they can be used in conjunction with debug capabilities available in Xilinx transceivers for quicker debug of system level and link training issues.

PCIe Link Quality Indicators

Although the link might be up, it might not remain stable all the time. The following are the key indicators of a bad PCIe link. If any of the following is seen repeatedly in the system, it is advised to go through the signal integrity checks discussed in “**PCIe Signal Integrity (SI) Debug**” section.

1. The PCI Express Base Specification defines the Correctable Error Status Register. If any bit of this register is set, it indicates the corresponding source of the error. Figure 55 shows a snapshot of PCI Express Base Specification, v2.1 correctable error status register definition.
2. When the link frequently goes into Recovery state, it is another indication of a poor link. This could be checked by looking at LTSSM graph in a link analyzer or by doing multiple triggers in ChipScope tool on entry into the Recovery state.
3. If the link analyzer shows numerous NAKs on the link, this is also an indication of a bad link and could affect the bandwidth of the system. NAKs are generated due to reasons such as bad CRC, bad sequence number, and et cetera.



Bit Location	Register Description	Attributes	Default
0	Receiver Error Status ¹⁰¹	RW1CS	0b
6	Bad TLP Status	RW1CS	0b
7	Bad DLLP Status	RW1CS	0b
8	REPLAY_NUM Rollover Status	RW1CS	0b
12	Replay Timer Timeout Status	RW1CS	0b
13	Advisory Non-Fatal Error Status	RW1CS	0b
14	Corrected Internal Error Status (Optional)	RW1CS	0b
15	Header Log Overflow Status (Optional)	RW1CS	0b

Figure 55 - PCI Express Base Specification, v2.1 - Correctable Error Status Register

Case Study - 1 - Virtex-6 board takes longer time to link up

The issue was seen with a board with a Virtex-6 device on it. Boards with Virtex-7 devices were working fine. The issue was that the link training was taking longer with a Virtex-6 board, which was twice the time a Virtex-7 board took. The link partner was a host system with an Intel chipset. The link training was going through Detect and Polling twice before going into configuration state, as shown in Figure 56. Virtex-6 boards took > 60 ms to link train, whereas Virtex-7 boards were linking up within ~25 ms. It did not go through double Detect and Polling (Figure 57) as it was happening with the Virtex-6 boards.

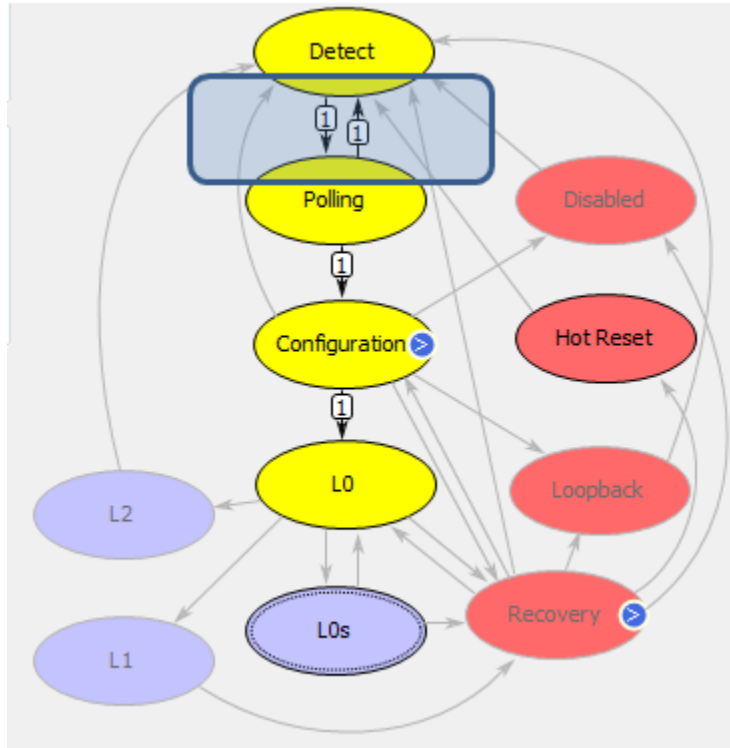


Figure 56 - Virtex-6 Board LTSSM

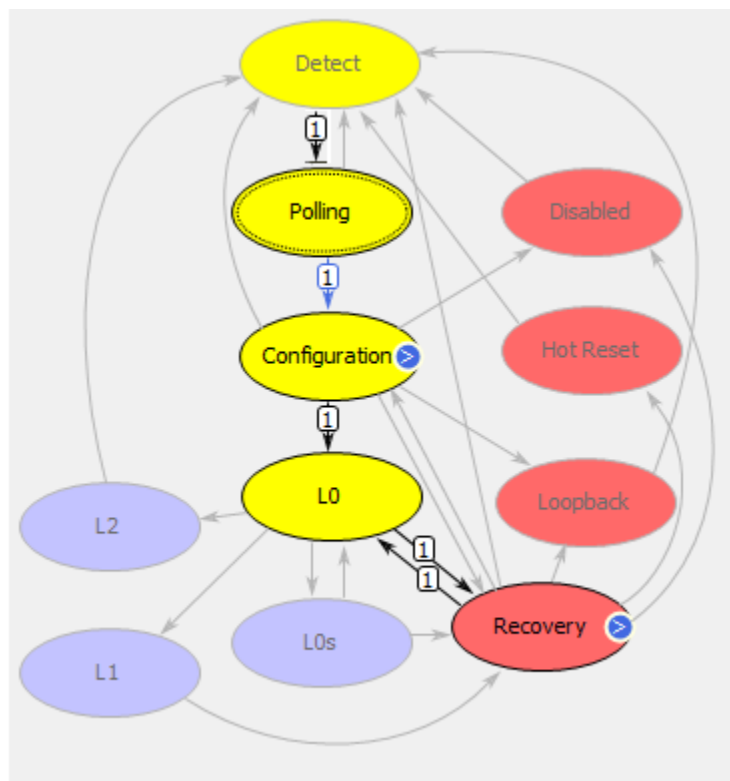


Figure 57 - Virtex-7 Board LTSSM

The following analysis of the Lecroy analyzer capture illustrates how the root cause of the issue was uncovered.

1. EP (Virtex-6 Endpoint) performs receiver detect, enters Polling.Active state and then Polling.Compliance as it receives no TS1s from Intel Server RP (Root Port) at this time. This is shown in Figure 58.

Packet	R	2.5	PATN	PATN Symbols	Idle	Time Stamp
0	R	x4	PATN	K28.5 D21.5 K28.5 D10.2	0.000 ns	0019 . 483 320 290 s
1	R	x4	PATN	K28.5 D21.5 K28.5 D10.2	0.000 ns	0019 . 483 320 306 s
2	R	x4	PATN	K28.5 D21.5 K28.5 D10.2	0.000 ns	0019 . 483 320 322 s
3	R	x4	PATN	K28.5 D21.5 K28.5 D10.2	0.000 ns	0019 . 483 320 338 s
4	R	x4	PATN	K28.5 D21.5 K28.5 D10.2	0.000 ns	0019 . 483 320 354 s
5	R	x4	PATN	K28.5 D21.5 K28.5 D10.2	0.000 ns	0019 . 483 320 370 s
6	R	x4	PATN	K28.5 D21.5 K28.5 D10.2	0.000 ns	0019 . 483 320 386 s

Figure 58 - EP in Polling. Compliance-> Send Pattern

2. After some time, RP enters Polling.Active and starts transmitting TS1s, as shown in Figure 59.

Packet	R	2.5	TS1	COM	Link	Lane	N_FTS	Training Control	Data Rate	TS1 Symbols	Idle	Time Stamp
528744	R	x4	TS1	K28.5	PAD	PAD	94	0 0 0 0 0	2.5 GT/s, 5 GT/s, 8 GT/s, Autonomous Change	D10.2 ...	0.000 ns	0019 . 493 037 84
528745	R	x4	TS1	K28.5	PAD	PAD	94	0				
528746	R	x4	TS1	K28.5	PAD	PAD	94	0				
528747	R	x4	TS1	K28.5	PAD	PAD	94	0				
528748	R	x4	TS1	K28.5	PAD	PAD	94	0				
528749	R	x4	TS1	K28.5	PAD	PAD	94	0				

LTSSM Flow Graph - [Fusion_IO_SandyBridge_ServerMB]

Direction: Upstream (R<-) Downstream (R->) Both

States Status: Previous: --- Current: Polling Next: Detect

Figure 59 - RP in Polling.Active

3. EP exits Polling.Compliance, as shown in Figure 60.

- RP fails to achieve bit/symbol lock and times out to Detect after 24 ms, as shown in Figure 63.

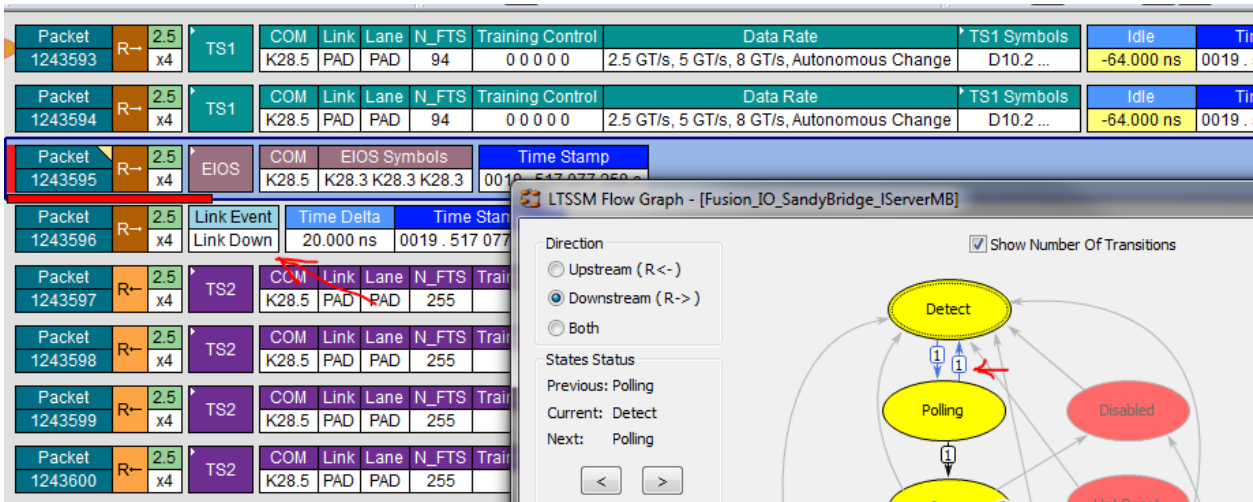


Figure 63 - RP Polling.Active timeout to Detect

- EP in Polling.Config does not receive TS2s and times out to Detect after 48 ms, as shown in Figure 64.

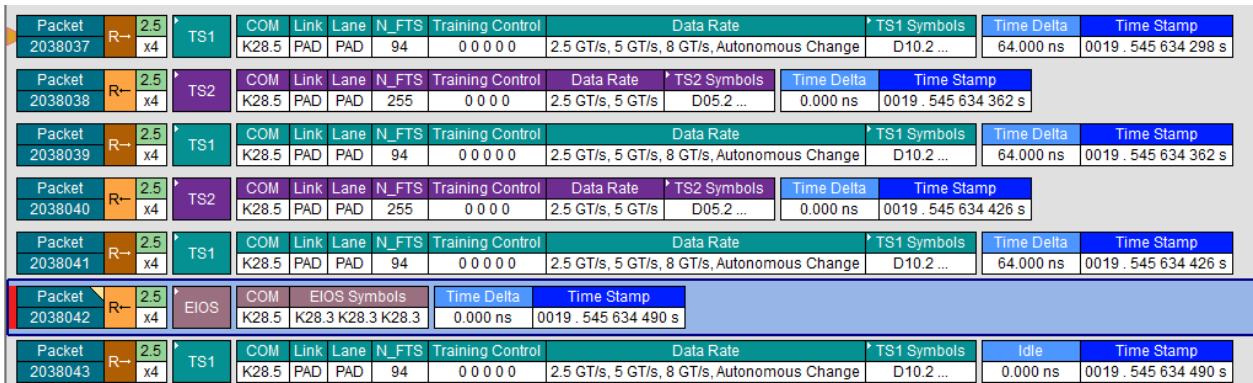


Figure 64 - EP Polling.Config timeout to Detect

- In the next pass of Polling.Active, RP gains bit/symbol lock moves to Polling.Config and link training is successful, as shown in Figure 65.

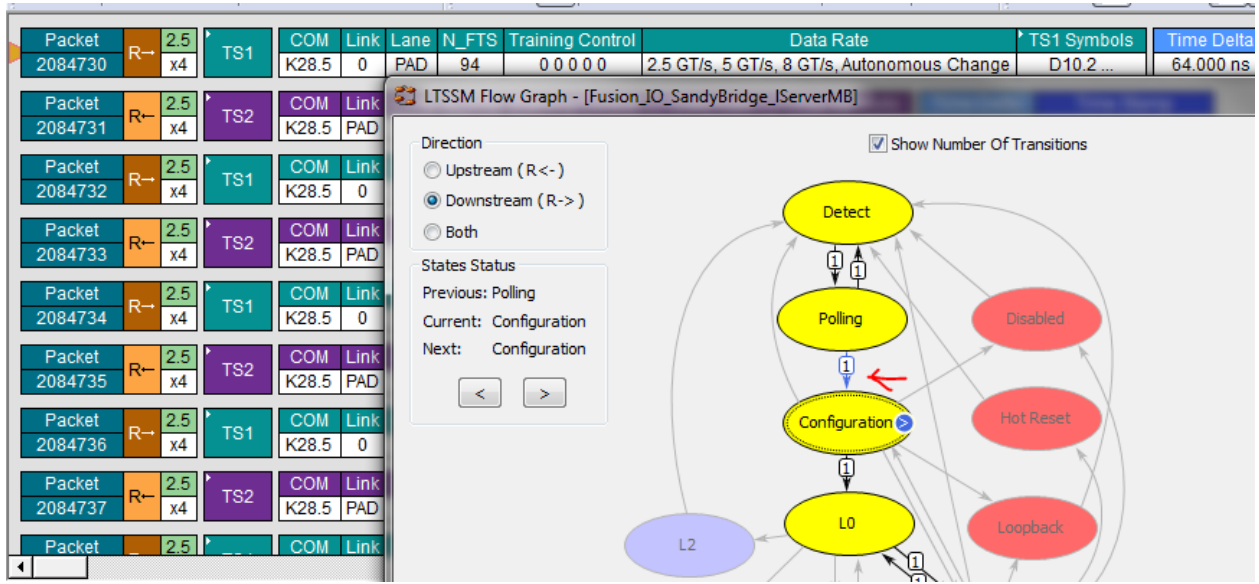


Figure 65 - Transition to Configuration State

The overlap of the 24 ms RP timeout / 48 ms EP timeout was resulting in 60 ms total linkup time. The ChipScope snapshot with storage qualification shown in Figure 66 shows the same sequence of events (05 (Polling.Configuration) -> 2D (Timeout to Detect) -> 02 (Detect)) as the 48 ms timeout.

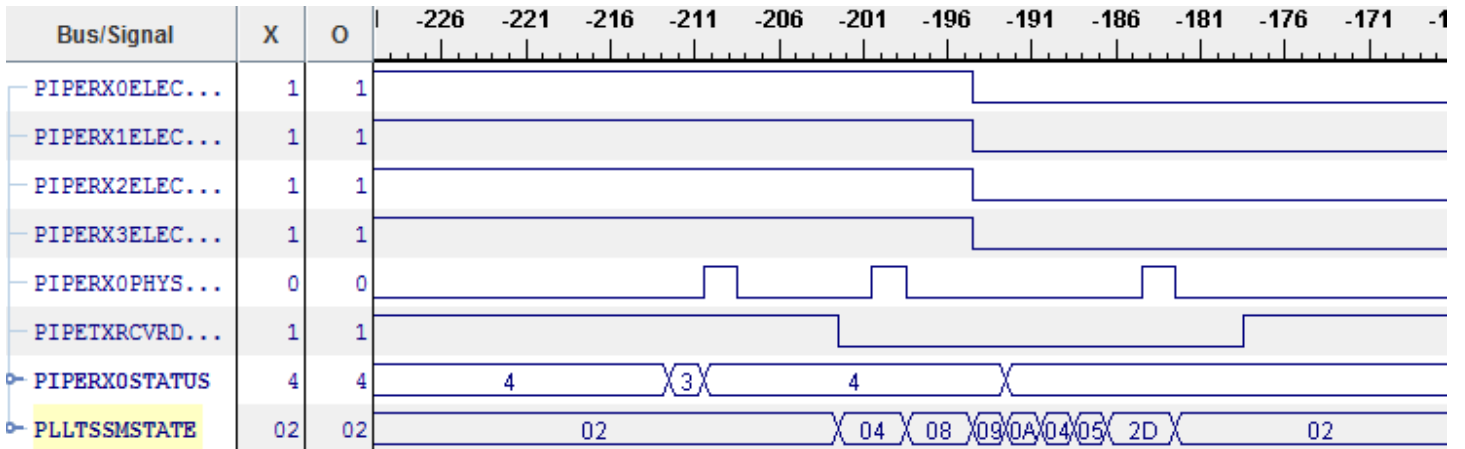


Figure 66 - ChipScope LTSSM transition with Storage Qualification

The cause of this issue is explained in the [Intel Errata](#) shown in Figure 67. The issue that was seen on Virtex-6 board is because the EP enters Polling.Compliance as mentioned in the errata, which in turn causes misalignment between the Polling.Active states of EP/RP, causing bit/symbol lock issues ending up in timeouts that result in the ~60 ms link training.

BT126. The PCIe* Receiver Lanes Surge Protection Circuit May Intermittently Cause a False Receive Detection on Some PCIe Devices

Problem: The processor implements a surge protection circuit on the PCIe receiver lanes. Due to this erratum, during platform power-on some PCIe devices may trigger the surge protection circuit causing a false receive detect. If this unexpected detection occurs before the processor's PCIe lane termination impedances are enabled and the resulting PCIe device link training enters the link training Polling.Active state, the PCIe device may incorrectly transition into the Polling.Compliance state.

Implication: After platform power-on, some PCIe devices may not exit from the compliance state causing the link to fail to train or the link may train to a degraded width.

Workaround: A BIOS change has been identified and may be implemented as a workaround for this erratum.

Figure 67 - Intel Errata

Case Study - 2 - Multiple resets result in link training down to Gen1 from Gen2

This was an issue in a particular system where after multiple resets the link was training to Gen1 from Gen2. This was seen in ChipScope capture by triggering ChipScope on the falling edge of pl_sel_lnk_rate when pl_link_gen2_cap was asserted, as shown in Figure 68.

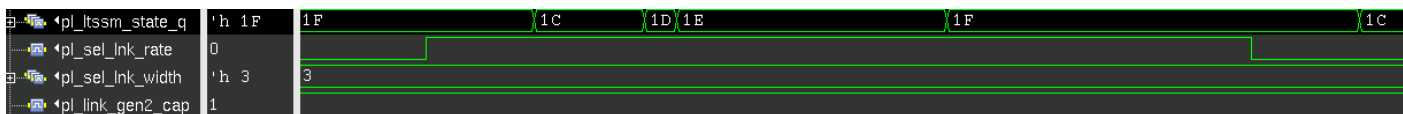


Figure 68 - Link training to Gen1 from Gen2 after multiple resets

By capturing the rxelecidle signal on all four lanes in ChipScope tool, it was seen that this signal was erroneously asserted on the first lane when the endpoint was in the Recovery Rcvrcfg LTSSM state, as shown in Figure 69.

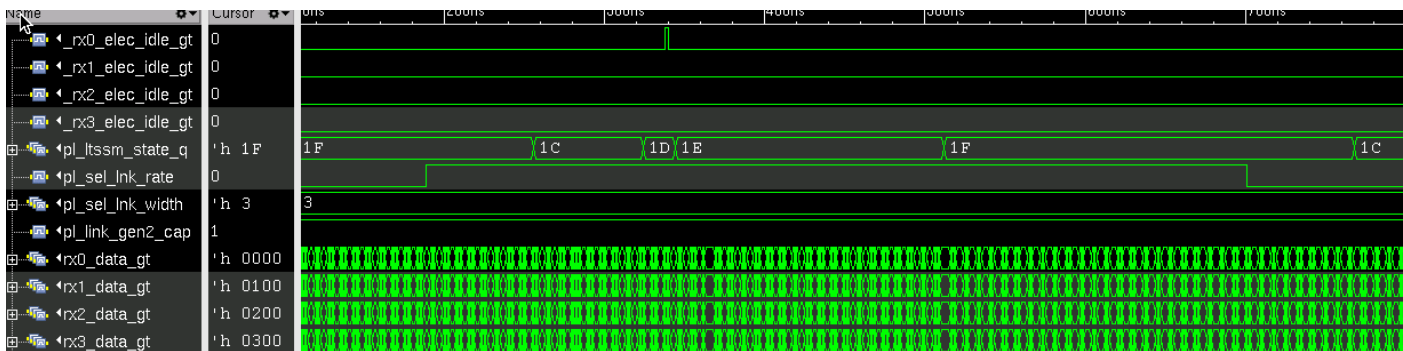


Figure 69 - Erroneous assertion of RXELECIDLE signal on Lane-0

The issue was resolved after changing RXOOB_CFG value from 7'b0000110 to 7'b0000010. RXOOB_CFG is a OOB block configuration attribute that determines the voltage level for detecting electrical idle on the link.

Case Study - 3 - x4 Gen 2 link only training to x1Gen2

This is an issue we have seen in the past where the link was training to x1Gen2 instead of x4Gen2. By looking at the Lecroy analyzer capture (Figure 70), it was found that the link partner was not detecting upper 3 lanes. The downstream component never turns on its transmitter on the upper three lanes. This is different than the problems encountered when you see the link partner initially turn on all four lanes and then down configure during CONFIGURATION.

15923	R	x4	TS1	K28.5	PAD	PAD	255	0 0 0 0	2.5 GT/s, 5 GT/s	D10.2 ...	0.000 ns	0002.365157064 s
Packet	R	2.5	TS1	COM	Link	Lane	N_FTS	Training Control	Data Rate	TS1 Symbols	Idle	Time Stamp
15924	R	x4	TS1	K28.5	PAD	PAD	255	0 0 0 0	2.5 GT/s, 5 GT/s	D10.2 ...	0.000 ns	0002.365157128 s
Packet	R	2.5	TS1	COM	Link	Lane	N_FTS	Training Control	Data Rate	TS1 Symbols	Idle	Time Stamp
15925	R	x4	TS1	K28.5	PAD	PAD	255	0 0 0 0	2.5 GT/s, 5 GT/s	D10.2 ...	0.000 ns	0002.365157192 s
Packet	R	2.5	TS1	COM	Link	Lane	N_FTS	Training Control	Data Rate	TS1 Symbols	Time Delta	Time Stamp
15926	R	x4	TS1	K28.5	PAD	PAD	255	0 0 0 0	2.5 GT/s, 5 GT/s	D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2		
				K28.5	PAD	PAD	255	0 0 0 0	2.5 GT/s, 5 GT/s	D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2		
				K28.5	PAD	PAD	255	0 0 0 0	2.5 GT/s, 5 GT/s	D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2		
				K28.5	PAD	PAD	255	0 0 0 0	2.5 GT/s, 5 GT/s	D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2		
Packet	R	2.5	TS1	COM	Link	Lane	N_FTS	Training Control	Data Rate	TS1 Symbols	Time Delta	Time Stamp
15927	R	x1	TS1	K28.5	PAD	PAD	255	0 0 0 0	2.5 GT/s, 5 GT/s	D10.2 ...	32.000 ns	0002.365157288 s
Packet	R	2.5	TS1	COM	Link	Lane	N_FTS	Training Control	Data Rate	TS1 Symbols	Time Delta	Time Stamp
15928	R	x4	TS1	K28.5	PAD	PAD	255	0 0 0 0	2.5 GT/s, 5 GT/s	D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2		
				K28.5	PAD	PAD	255	0 0 0 0	2.5 GT/s, 5 GT/s	D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2		
				K28.5	PAD	PAD	255	0 0 0 0	2.5 GT/s, 5 GT/s	D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2		
				K28.5	PAD	PAD	255	0 0 0 0	2.5 GT/s, 5 GT/s	D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2D10.2		
Packet	R	2.5	TS1	COM	Link	Lane	N_FTS	Training Control	Data Rate	TS1 Symbols	Time Delta	Time Stamp
15929	R	x1	TS1	K28.5	PAD	PAD	255	0 0 0 0	2.5 GT/s, 5 GT/s	D10.2 ...	32.000 ns	0002.365157352 s

Figure 70 – RP transmitting on lane-0 only while EP transmits on all 4 lanes

This issue was resolved by adding a resistor on the board trace that allowed the RP to correctly detect receiver on all four lanes.

Points to note

Most of the things that need to be taken care of in a design and on the board have been discussed in the previous sections. Below are few more points that a designer must check to ascertain proper working of the link.

- An AC coupling capacitor given by $CTX = 75 \text{ nF}$ to 200 nF (per Differential Transmitter (TX) Output Specifications) must be used on the Transmitter side of each lane of a link.
- REFCLK must meet the electrical specifications listed in REFCLK DC Specification and AC Timing Requirements mentioned in the PCI Express Card Electromechanical Specification.
- REFCLK must meet the jitter specifications listed in Maximum Allowed Phase Jitter When Applied to Fixed Filter Characteristic mentioned in the PCI Express Card Electromechanical Specification.
- A PCI Express add-in card must incorporate AC coupling capacitors on the Transmitter differential pair. The value must comply to the value in the PCI Express Base Specification.
- Add-in cards must meet the Add-in Card Transmitter Path Compliance Eye Requirements specified in Add-in Card Transmitter Path Compliance Eye Requirements of the PCI Express Card Electromechanical Specification, measured when all lanes are active.
- The PCB differential trace impedance for 5.0 GT/s capable add-in cards and motherboards must be between 68 and 105 ohms.

Debugging Link Training Issues with Lecroy Protocol Analyzer

When debugging PCIe link training issues, it would be helpful to have a link protocol analyzer such as Lecroy. It allows you to check the traffic on the link during the link training process. If the link is not stable, LTSSM state diagram feature in the Lecroy Tracer software shows how many times the link is going into Recovery. If the link fails to train to L0 state, the state diagram shows at what point in the state diagram, including a sub-state in the LTSSM main states, the link gets stuck. By clicking on that particular state/sub-state, it shows the traffic on the link at that point. This will give an idea for the designer whether the issue is on the TX side or the RX side and if both, what are the errors that exist in the link.

This section does not cover detailed steps for debugging link training issues using Lecroy Protocol Analyzer. However, few major features that help in debugging link training with a Lecroy analyzer are listed below to illustrate to readers the advantage of having a protocol analyzer for debugging link training issues. A comprehensive detail on how to use the link capture software and how to setup triggers (and etc.) can be found in the Lecroy documentation.

1. The Error Summary dialog shown in Figure 71 displays the number of errors for each event, and the packet containing errors.

Traffic Summary Report

Type /	Upstream	Downstream	Total
Invalid Code	0	0	0
Running Disparity Error	0	0	0
Unexpected K/D Code	0	22	22
Idle Data Error (not D0.0)	6458	1	6459
Skip Late	0	0	0
Skew Error	0	0	0
Bad Packet Length	0	0	0
Ordered Set Format Error	0	31	31
Delimiter Error	0	0	0
Alignment Error	0	0	0
DLLP: Invalid Encoding	0	0	0
DLLP: Bad CRC16	0	0	0
DLLP: Reserved Field not 0	0	0	0
DLLP: FC Initialization Error	0	0	0
TLP: Invalid Encoding	0	0	0
TLP: Bad LCRC	0	0	0
TLP: Bad ECRC	0	0	0
TLP: Reserved Field not 0	0	0	0
TLP: Payload/Length Error	0	0	0
TLP: Length Error (not 1)	0	0	0
TLP: TC Error (not 0)	0	0	0
TLP: Attr Error (not 0)	0	0	0
TLP: AT Error (not 0)	0	0	0
TLP: Byte Enables Violation	0	0	0
Memory TLP: Address/Length Crosses 4K	0	0	0
Mem64 TLP: Used Incorrectly	0	0	0
Cfg TLP: Register Error	0	0	0
Msg TLP: Invalid Routing	0	0	0
Gen3 TLP: Bad Len CRC/Parity	0	0	0
Invalid Packet	83895	0	83895
FC: Invalid Advertisement	0	0	0
FC: Insufficient Credits	0	0	0

Figure 71 Lecroy Error Summary

This summary will help in figuring out where the issue might be and what should be looked at. For example, running disparity error, idle data error, and et cetera are the indications of signal integrity issues on the board. After tuning various aspects in the board and also tuning GT attributes, if this reduces the number of errors reported, it would help to narrow down the issue.

2. Traffic summary, shown in Figure 72, provides a summary of different packet types (e.g. TLP, Physical Ordered Sets such as TS1s/ TS2s, etc.) on the link. If there is no TS1/TS2 reported in the summary, it would indicate a major issue with the link (i.e., due to signal integrity issue). If these ordered sets are not properly captured by the analyzer, it is not expected that the core would be able to recognize these ordered sets.

Traffic Summary Report

Type /	Upstream	Downstream	Total
TLP	0	3	3
DLLP	498697	2089165	2587862
TS1 Ordered Set	72	376297	376369
TS2 Ordered Set	374992	97	375089
Fast Training Sequence	0	0	0
Electrical Idle Ordered Set	1	1	2
SKP Ordered Set	1556967	1580699	3137666
Compliance Pattern	6454	0	6454
Electrical Idle Exit Ordered Set	31	86	117
Link Event	16	2	18
Start Data Stream Ordered Set	0	0	0
End Bad Framing Token	0	0	0
End Data Stream Framing Token	0	0	0
Invalid	83895	0	83895
			6567475

Figure 72 - Lecroy Traffic Summary

3. LTSSM Flow Graph is one of the key features in Lecroy that shows how the link is transitioning through different states of LTSSM. Figure 73 shows downstream LTSSM flow graph where the link trains to L0 and again goes to recovery to link train to Gen2 speed. The process for link training to Gen2 speed has been described in section – **“Link Speed Train Down”**.

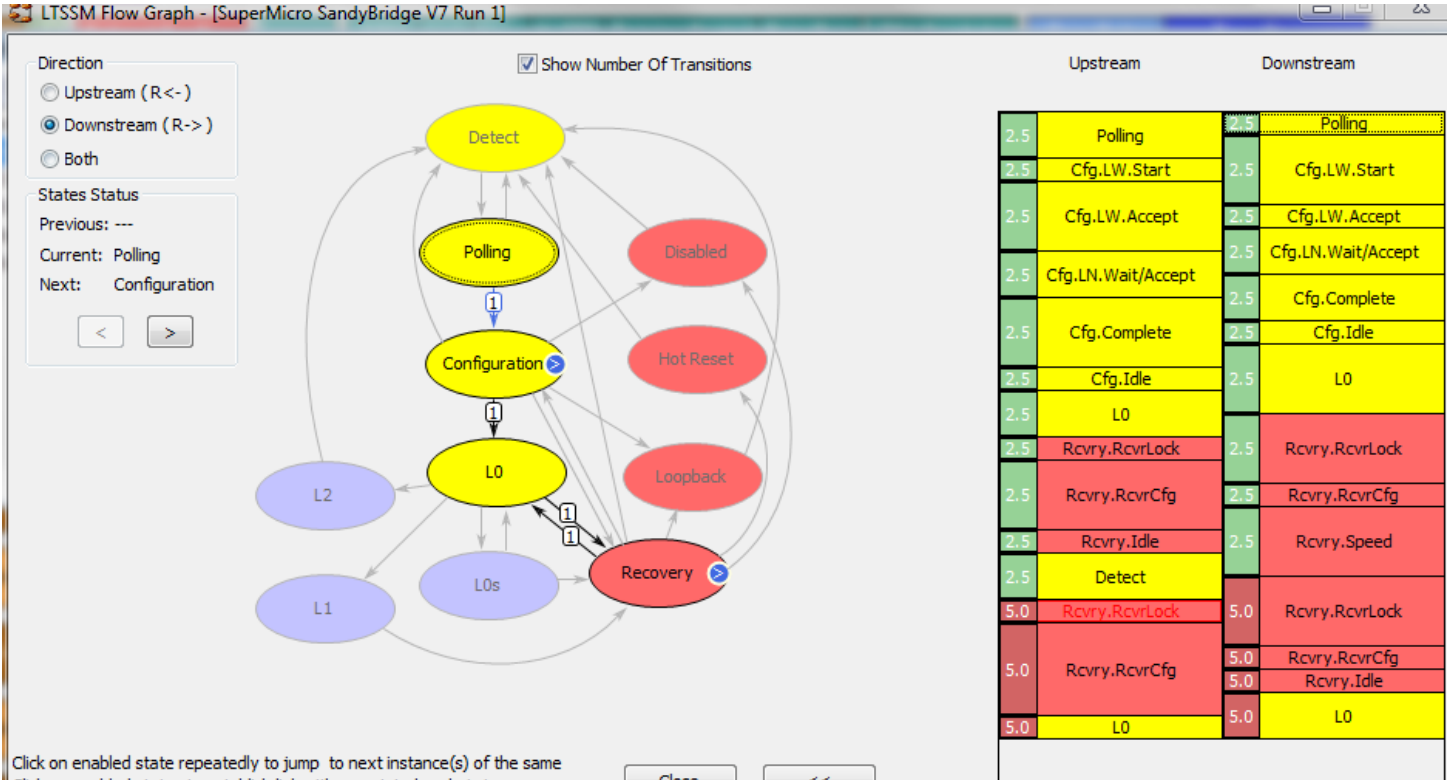


Figure 73 - Downstream LTSSM Flow Graph

Figure 74 shows Recovery sub-states State diagram. From L0 it goes to Rcvry.Rcvr.Lock -> Rcvry.RcvrCfg -> Rcvry.Speed-> Rcvry.RcvrLock -> Rcvry.RcvrCfg -> Rcvry.Idle -> L0. This is exactly the flow that should be followed, as defined in the PCI Express Base Specification, for link training to Gen2 speed. This is also illustrated in Figure 11.

Figure 75 shows a packet on the link when LTSSM goes into Rcvry.RcvrLock substate. The tool jumps into specific packet on the link when clicking a substate either in the LTSSM state diagram shown in Figure 75 or by clicking on the rectangular sub-state box in the vertical state flow diagram shown in the same figure.

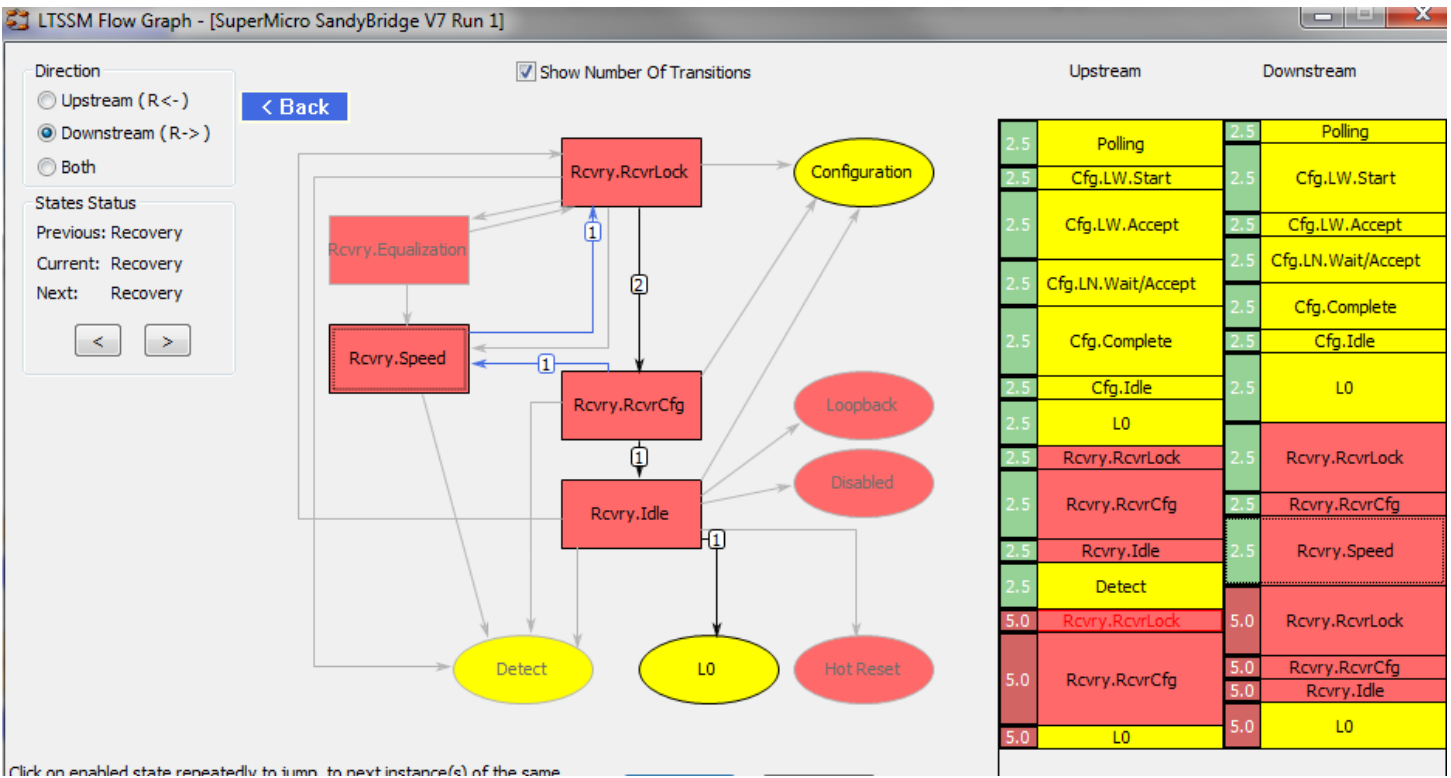


Figure 74 - Downstream Recovery sub-states state diagram

Packet 847932 R- 2.5 x4 TS1 COM K28.5 Link 0 Lane * N_FTS 94 Training Control 0 0 0 0 Data Rate 2.5 GT/s, 5 GT/s, 8 GT/s, Speed Change TS1 Symbols D10.2 ... Time Delta 8.000 ns Time Stamp 0051.770265528 s

Packet 847933 R- 2.5 x4 DLLP InitFC2-Cpl VC ID 0 HdrFC 0 DataFC 0 CRC 16 0xA2ED Idle 0.000 ns Time Stamp 0051.770265536 s

Packet 847934 R+ 2.5 x4 DLLP

Packet 847935 R+ 2.5 x4 DLLP

Packet 847936 R+ 2.5 x4 DLLP

Packet 847937 R+ 2.5 x4 DLLP

Packet 847938 R+ 2.5 x4 DLLP

Packet 847939 R+ 2.5 x4 DLLP

Packet 847940 R+ 2.5 x4 DLLP

Packet 847941 R- 2.5 x4 TS1

Packet 847942 R+ 2.5 x4 DLLP

Packet 847943 R+ 2.5 x4 DLLP

Figure 75 - Rcvry.RcvLock sub-state.

Figure 76 shows Configuration Substate Machine. It would be helpful to check the TS1 and TS2 content during different substates. This would be helpful when the link is training down to lower lane width. Link and Lane width negotiation takes place in Configuration state when each link partner advertises the link number and lane number on each lane that it can communicate with. After exchanging certain number of TS1s and TS2s, both sides of the link agree on the same link number and lane number on respective lanes. On other lanes, it will contain PAD value in both link and lane number field.

Figure 77 shows Root Complex advertising Link Number-0 on all four lanes whereas the lane number on all lanes is set to PAD. This is in Cfg.LW.Start configuration sub-state. After some time, the endpoint also goes into CfgLW.Start configuration substate and starts to advertise the same link number on all of its four lanes.

The next states are Cfg.LW.Accept and Cfg.LN.Wait/Accept states. In these states, both Root Complex and the Endpoint start to send link numbers and lane numbers on respective lanes, as shown in Figure 79. In the case of successful link negotiation (without down training to lower lane width), both sides should be sending the same numbers on both link number and lane number fields. When the link down trains to lower lane width, either one or both the partners would be advertising link and lane number on lane-0 only if it down trains to x1 lane; other lanes would have PAD in the link and lane number fields in TS1. If the Root Complex is advertising the link and lane numbers on all four lanes but the endpoint replies with link and lane numbers on only lane-0, this could be an indication of an issue at the receive side, which causes the endpoint to not be able to understand the incoming TS1s on upper lanes. If it was vice versa, it could be an indication of an issue at the transmit side, causing the data to be garbled on the upper lanes on the link, and hence the Root Complex would not be able to understand the incoming TS1s on upper lanes.

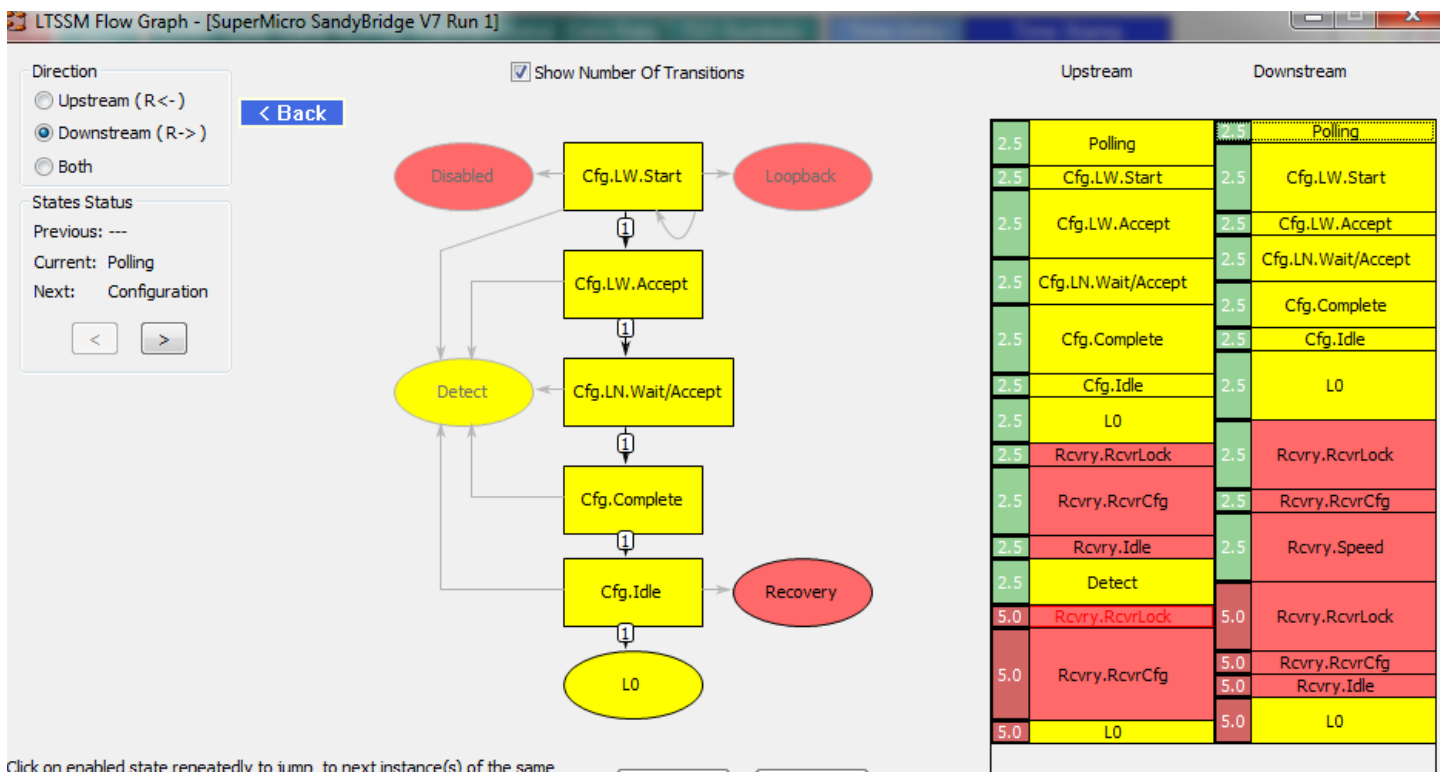


Figure 76 – Downstream Configuration Substate Machine

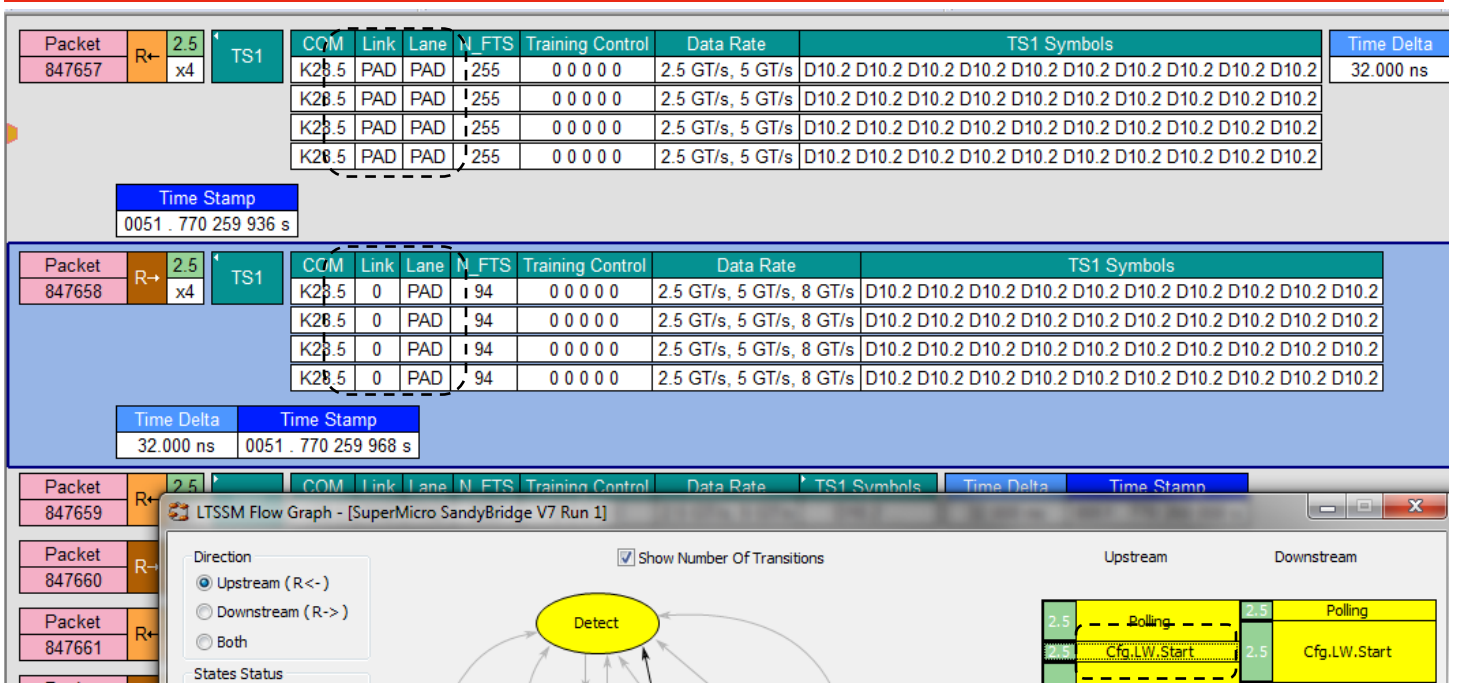


Figure 77 - Root complex advertising link number on all four lanes

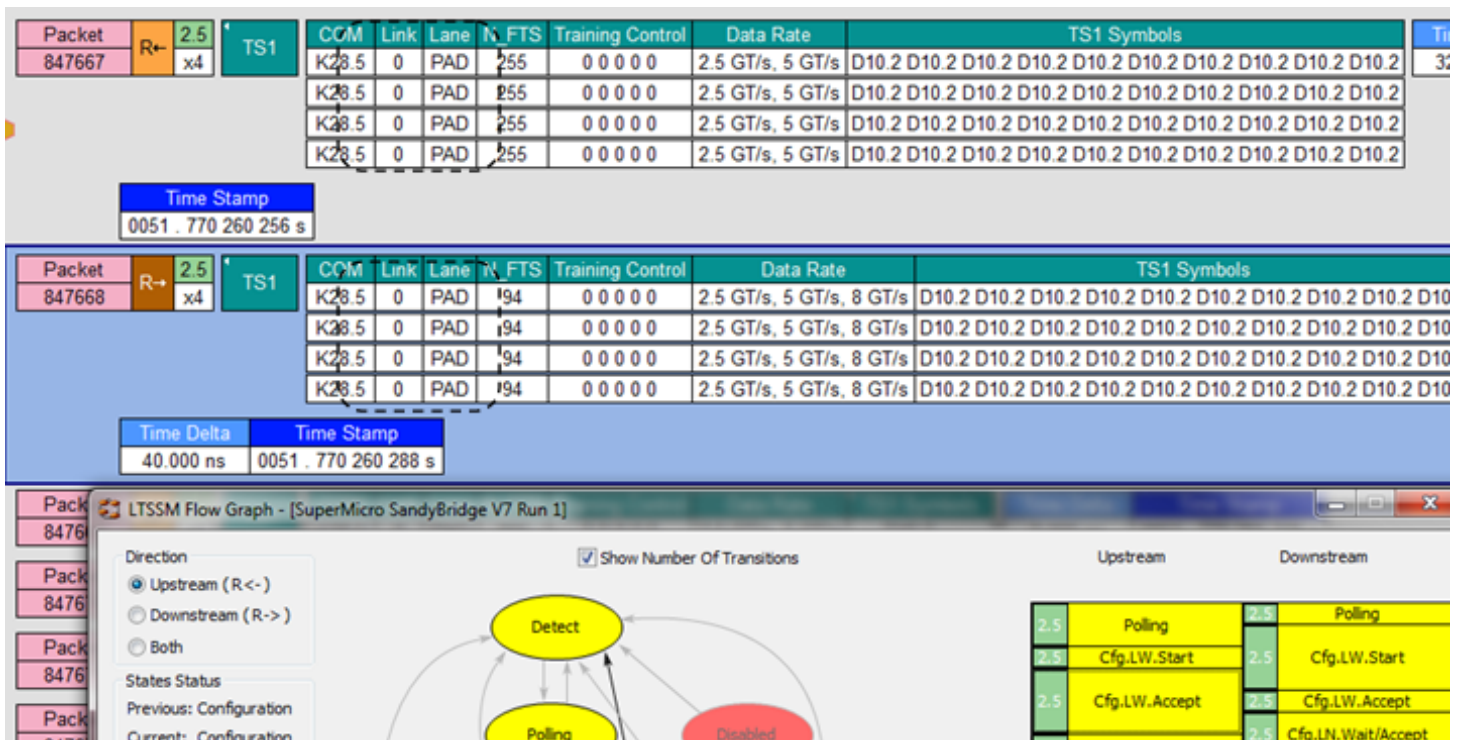


Figure 78 - Endpoint advertising link number on all four lanes

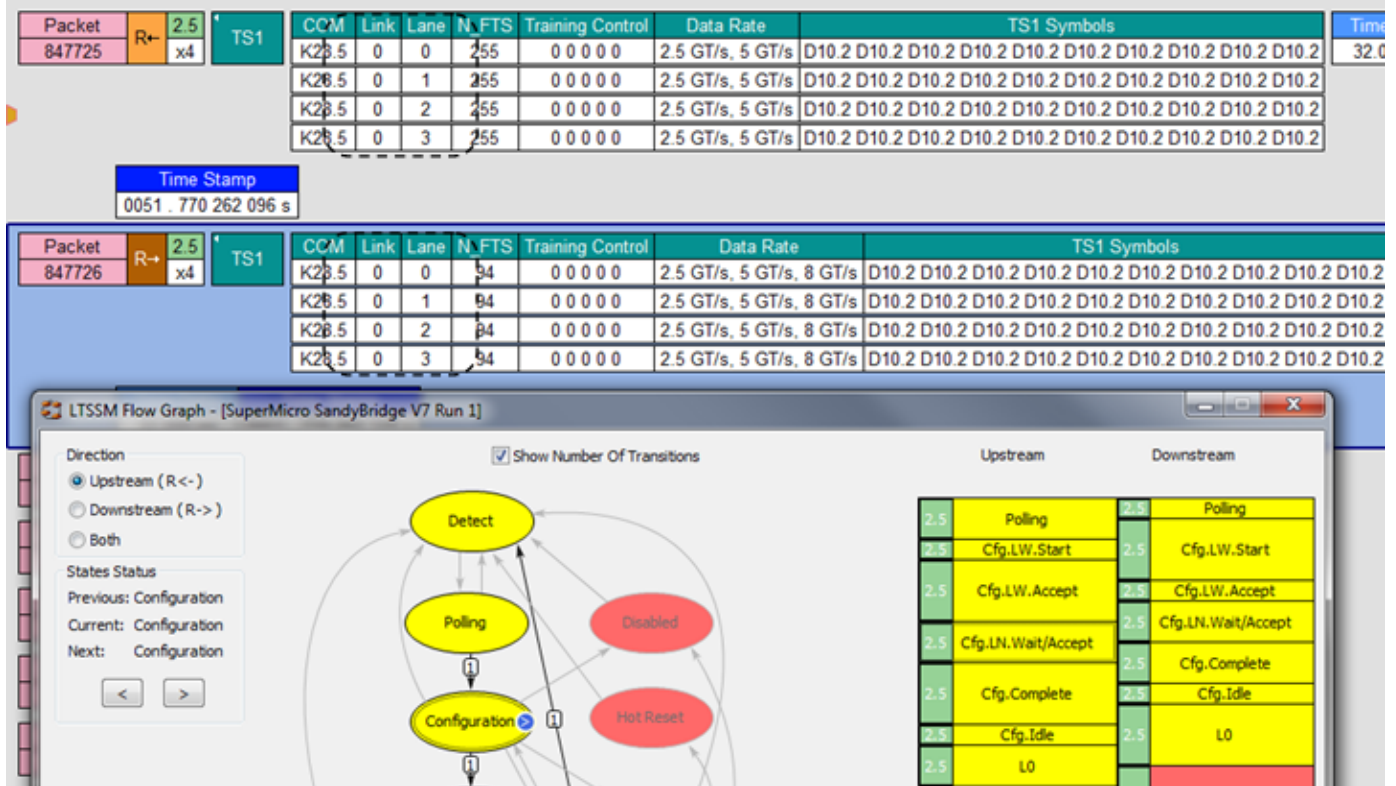


Figure 79 - Successful x4 link negotiation

General Debugging and Packet Analysis Guides

Although the documents in the following answer records are for Virtex-5 and Virtex-6 FPGAs, the principle applies to Kintex-7 as well. Xilinx Answer Record 42368 describes debugging link training issues in Virtex-5 FPGA. The other two documents talk about packet analysis, how to identify packets and how to track PCIe packets along different interfaces of the core.

- 42368: V5 PCIe Link Training Debugging Guide
 - <http://www.xilinx.com/support/answers/42368.htm>
- 46888: V5 PCIe Packet Analysis
 - <http://www.xilinx.com/support/answers/46888.htm>
- 50234: Virtex-6 PCIe Packet Analysis using PIO Example Design
 - <http://www.xilinx.com/support/answers/50234.htm>

Known Issues Answer Record

When debugging issues related to link training or any other issues related to the PCI Express cores, first take a look at the release notes of the corresponding cores. The ARs below lists release notes AR for 7-Series Integrated Block for PCI Express cores.

- <http://www.xilinx.com/support/answers/40469.html>
 - Release notes for versions of the 7-Series Integrated Block for PCI Express which were released in ISE Design Suite and Vivado Design tool (prior to 2013.1)
- <http://www.xilinx.com/support/answers/54643.html>

- Release notes for version of the 7 Series Integrated Block for PCI Express were released in Vivado Design tool in and after 2013.1.

What if the issue is still not resolved?

After going through the debug steps and things to check described in this document, if the issue you are having is still unresolved, create a Webcase with Xilinx Technical Support providing entire details of the debugging you have done. Also, attach ChipScope / Vivado ILA captures with the Webcase. If you have any specific ideas about what might be causing the issue based on the debugging you have done, and you want Xilinx Technical Support to focus investigation on that particular possibility, attach the details with the Webcase. To make it easier to analyze and investigate the issue you are having, below you will find a list of questions for us to understand your system environment and to get the basic understanding of the issue you are having. Please copy this list in the Webcase and provide answer to all applicable questions.

1. Please describe the failure observed in as much detail as possible.
2. Silicon Revision (IES,GES, Production)
3. Silicon Serial Number
4. Silicon Speed Grade
5. PCIe Core Version
6. Did the issue occur in previous PCIe core versions too?
7. Was the failure observed in previous silicon revision?
8. Indicate the part ID for the failing part and passing parts (if any).
9. Indicate what board you are using: is it a Xilinx Development Board or a Customer Board. If it is a Xilinx development board, please provide the board revision ID.
10. Indicate motherboard description.
 - a. "What is the link partner? Is it a switch, a PC?" Who's the manufacturer of that switch or PC?
 - Which Chipset are you using?
11. Was Lecroy used? If so, provide the Lecroy captures with the details of your analysis of the captures.
12. Indicate which ISE/Vivado build was used?
13. Did the failure occur in Gen1, Gen2, and/or Gen3?
14. Provide the physical GT location of each lane.
15. Did the failure occur as RP (Root Port) and/or EP (Endpoint)?
16. Were there any implementation (synthesis, mapping, routing) errors?
17. Were there any timing errors?
18. Which lane(s) failed?
19. Is the failure always on the same lane?
20. Do other link width configurations show similar behavior?
21. What is the frequency of the error? For e.g., does it happen immediately or after 1 hour?
22. Can the error be cleared? If cleared, does the error come back?
23. Is this failure observed on multiple parts?
24. Did failure occur immediately after reset?

25. Did failure occur immediately, after first rate change, after multiple rate changes? How long after successful rate change did it failed?
26. Are PCLK and RXUSRCLK at the correct frequency and locked? May need to bring out clocks to confirm with scope.
27. Did all the lock signals, PCLK_LOCK, RXCDRLOCK, and CPLLOCK remain locked after reset?
28. Does the issue occur with the Example Design as well or only in your design?
29. Have you tried with x1 configuration?
30. Do you have a different board that you could try on? If you do, do you see the same issue on that board?
31. Have you tried on a different machine?
32. What is the clocking architecture? Synchronous or Asynchronous?
33. Is it through a backplane, embedded system, or through PCIe fingers?

Appendix

Capturing Signals in ChipScope Pro

To capture signals in ChipScope Pro, a user may use either ChipScope Pro Inserter flow or ChipScope Pro CORE Generator flow. In the Inserter flow, the user would enter the .ngc file into the tool and the tool then automatically lists the signals for the user to select and capture in ChipScope Pro. In the CORE Generator flow, the user must generate the ChipScope Pro cores in CORE Generator and instantiate them manually in the source file. ChipScope Pro Inserter flow is easier, but the required signals might not be visible. However, in the CORE Generator flow, a user can select to capture any signals in the source file. In this section, ChipScope Pro Inserter flow is discussed.

In some cases, the signals are optimized away during synthesis and hence the signals cannot be found in the ChipScope Pro inserter. In such cases, use the KEEP attribute to stop XST from optimizing a particular signal.

In VHDL, declare the KEEP attribute in the file architecture, before the "begin" keyword:

```
attribute keep: string
```

After KEEP and the signal have been declared, specify the VHDL constraint as follows:

```
attribute keep of signal_name: signals is "true";
```

In Verilog, add following:

```
(* KEEP = "{TRUE}" *)  
wire signal_name;
```

Below are the steps to capture signals with ChipScope Pro inserter flow.

1. After generating the core in CORE Generator, modify the xilinx_pcie_2_1_ep_7x.xst script in the 'implement' directory to set 'KEEP_HIERARCHY' to yes, if it has not already done so.

```
run  
-p xc7k325t-ffg676-2  
-ifn xilinx_pcie_2_1_ep_7x.prj
```



```
-ifmt VERILOG
-ofn xilinx_pcie_2_1_ep_7x.ngc
-use_dsp48 no
-bufg 0
-top xilinx_pcie_2_1_ep_7x
-opt_mode SPEED
-opt_level 2
-max_fanout 100
-keep_hierarchy yes
-rtlview yes
-use_sync_reset yes
-uc xilinx_pcie_2_1_ep_7x.xcf
```

2. Run `implement.bat/implement.sh` depending on the operating system you are using.

3. Once the synthesis is complete, the `.ngc` file called 'xilinx_pcie_2_1_ep_7x.ngc' is generated in the 'results' directory inside the 'implement' directory.

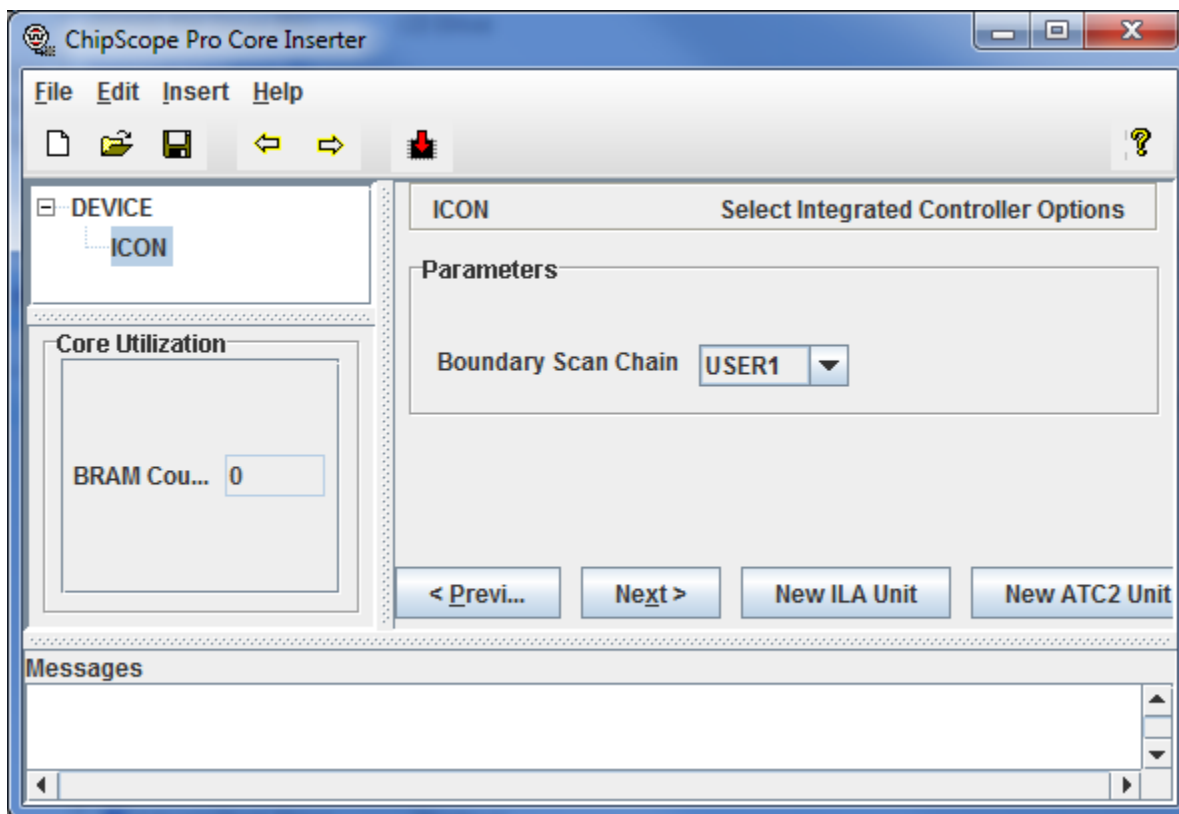


Figure 80: Chipscope Pro Inserter - Boundary Scan Chain

- Select trigger width as required (Figure 81).

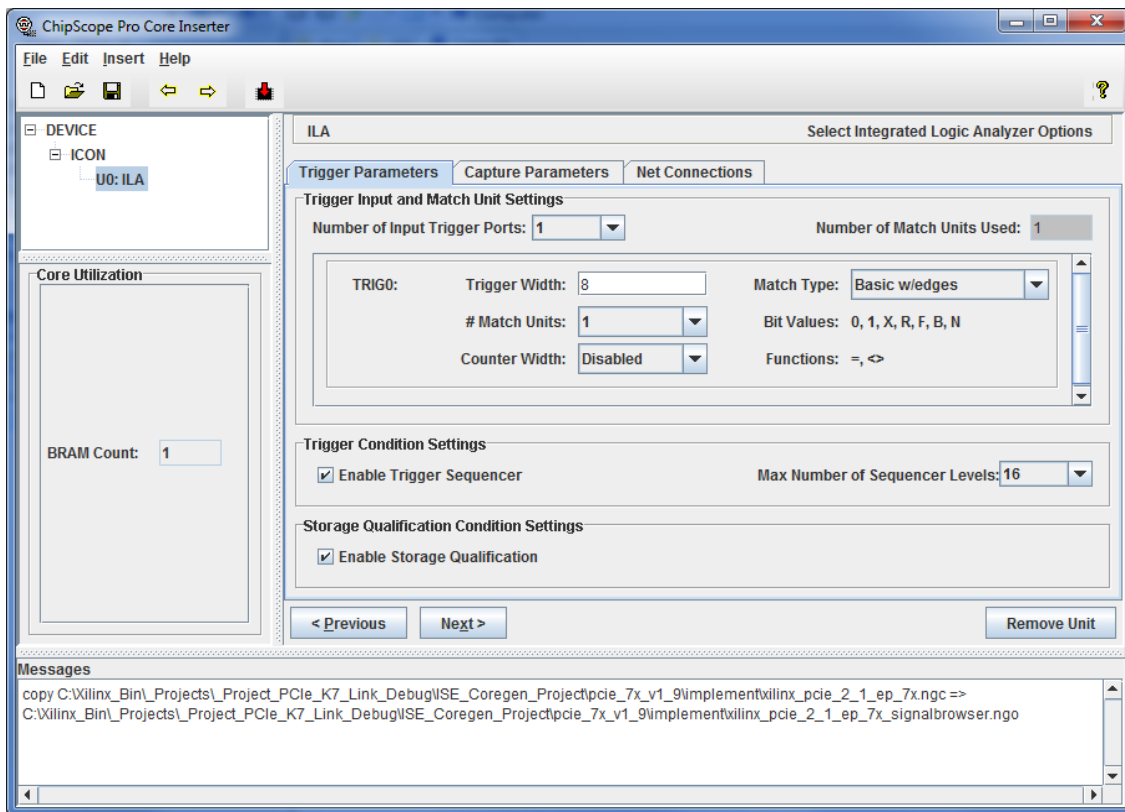


Figure 81: Chipscope Pro Inserter - Integrated Logic Analyzer Options

- Select the 'data width' and the 'data depth' as required. Open ChipScope Pro inserter. Specify the location of the input design netlist. If you are using the same name for the output design netlist and the output directory you specify is where the original input design netlist is located, ChipScope Pro inserter will replace the input design netlist with the output design netlist. If you either rename the output design netlist or specify a different output directory, make sure you replace the input design netlist with the generated output design netlist.

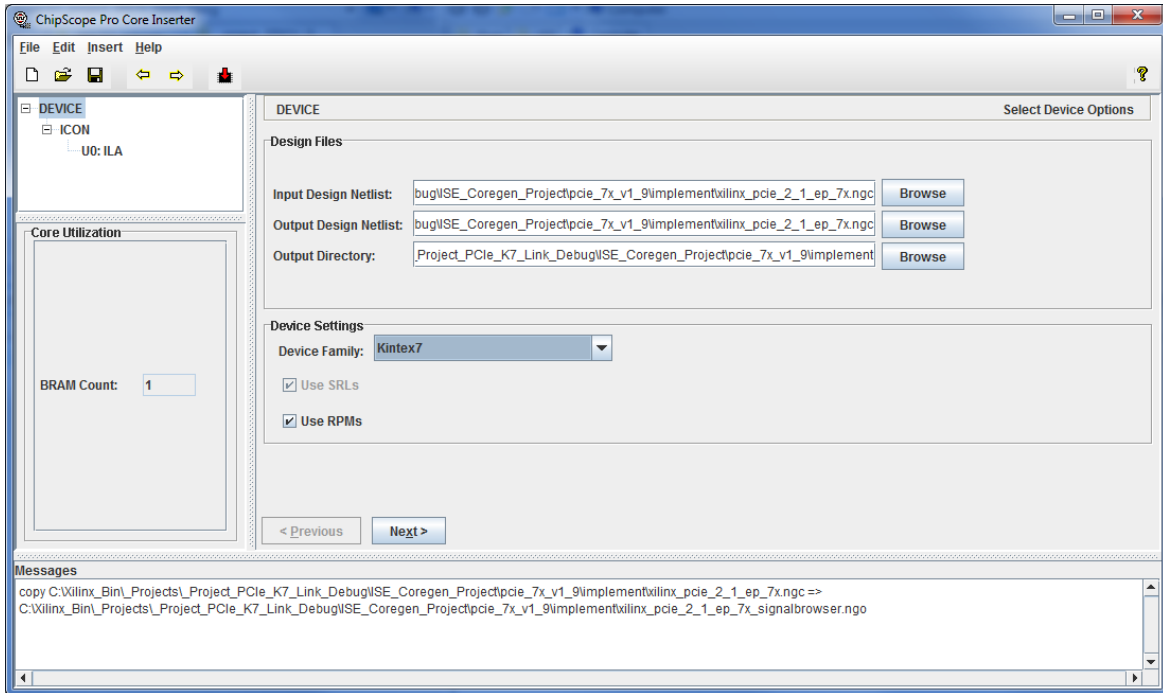


Figure 82: ChipScope Pro Inserter - Device and Design Netlist Entry

- Select USER1 in 'Boundary Scan Chain'.

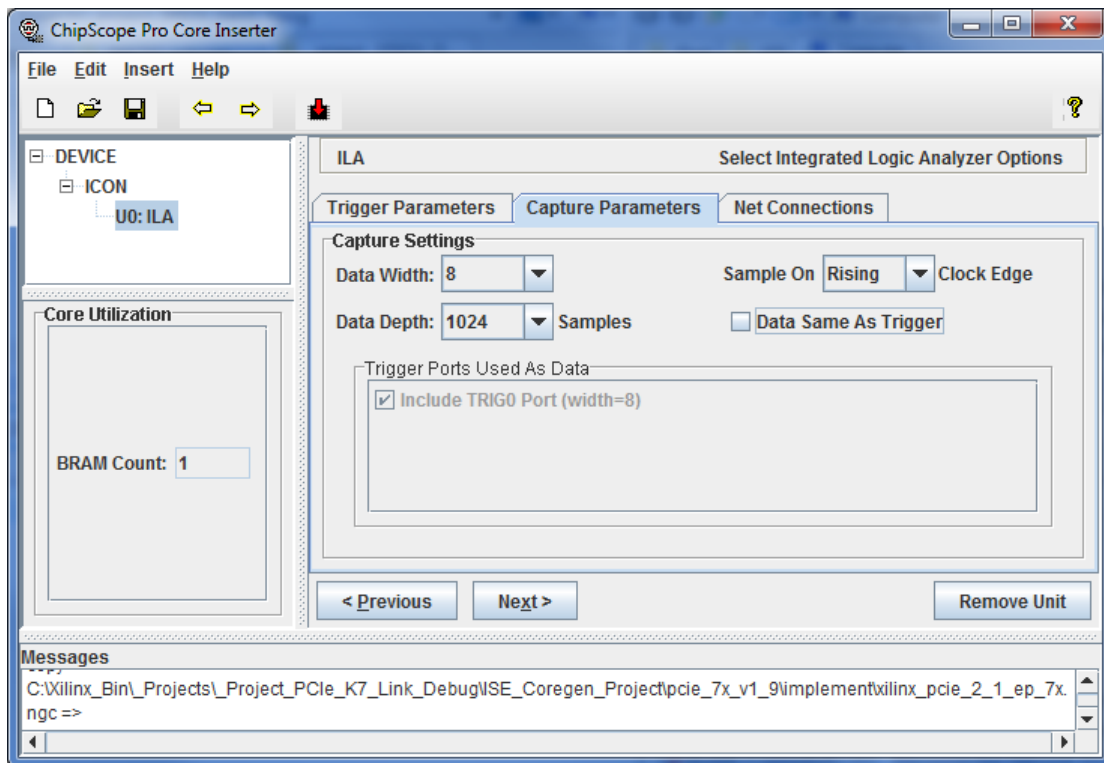


Figure 83: ChipScope Pro Inserter - Data Width and Data Depth Selection

- Double click on any of the ports shown in red below:

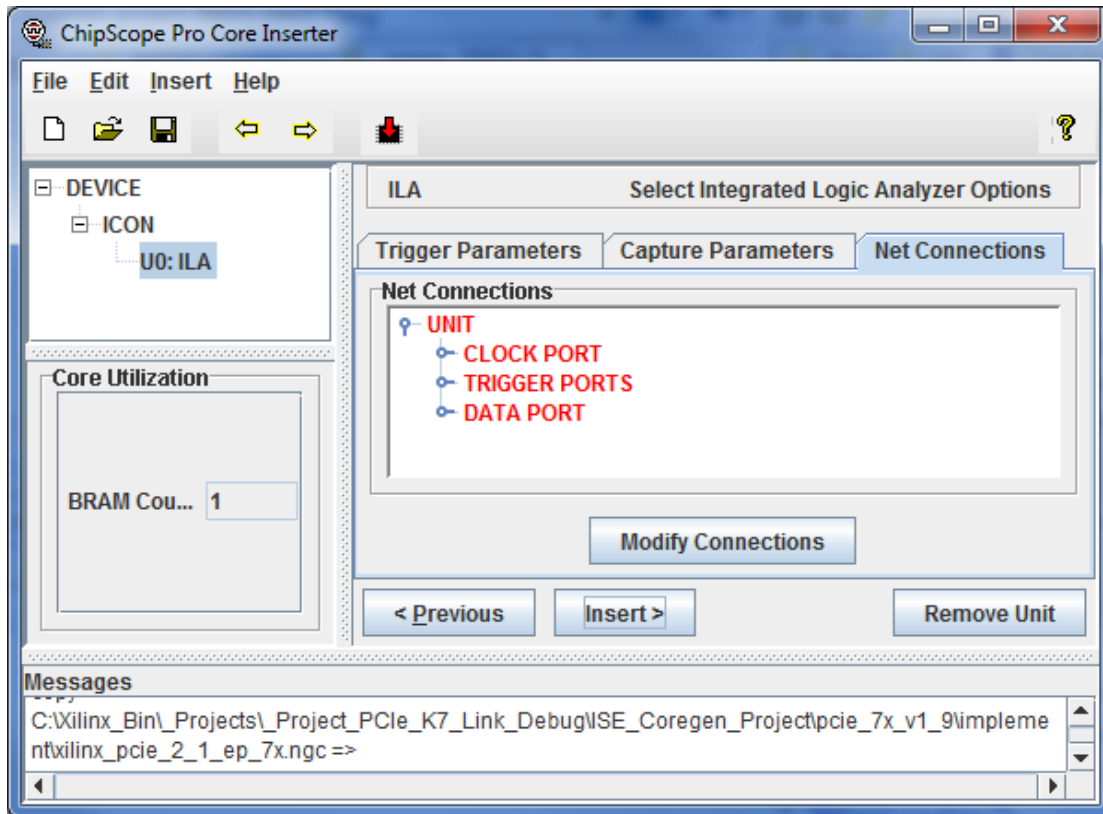


Figure 84: ChipScope Pro Inserter - Net Connections

- Click on the appropriate section of the structure hierarchy to select the signals (Figure 85).

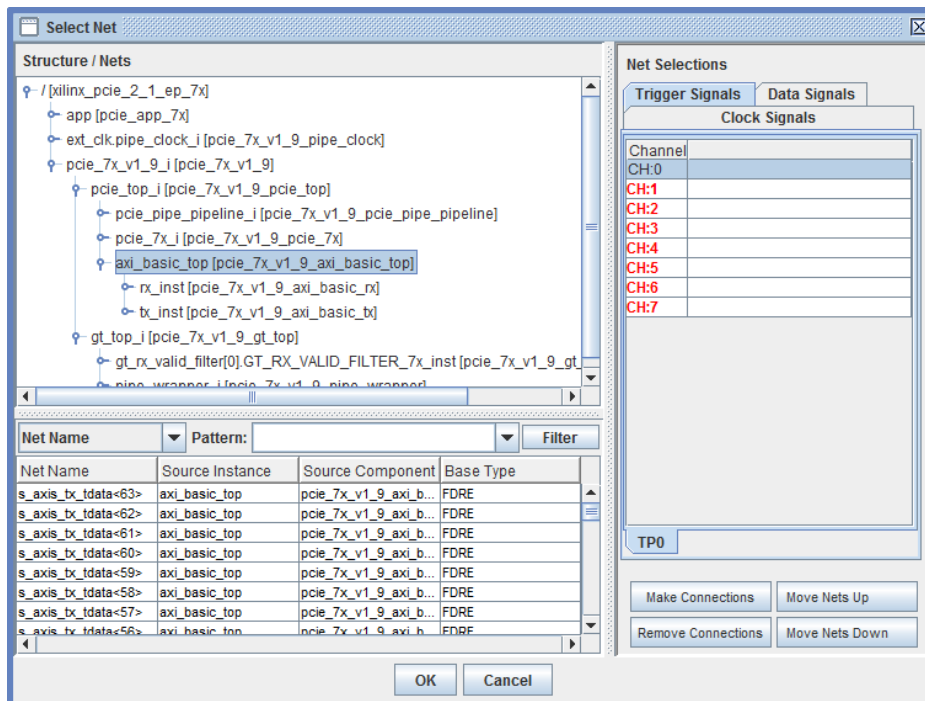


Figure 85: ChipScope Pro Inserter - Selecting Data Signals

- Select pipe_clk for the clock signal (Figure 86).

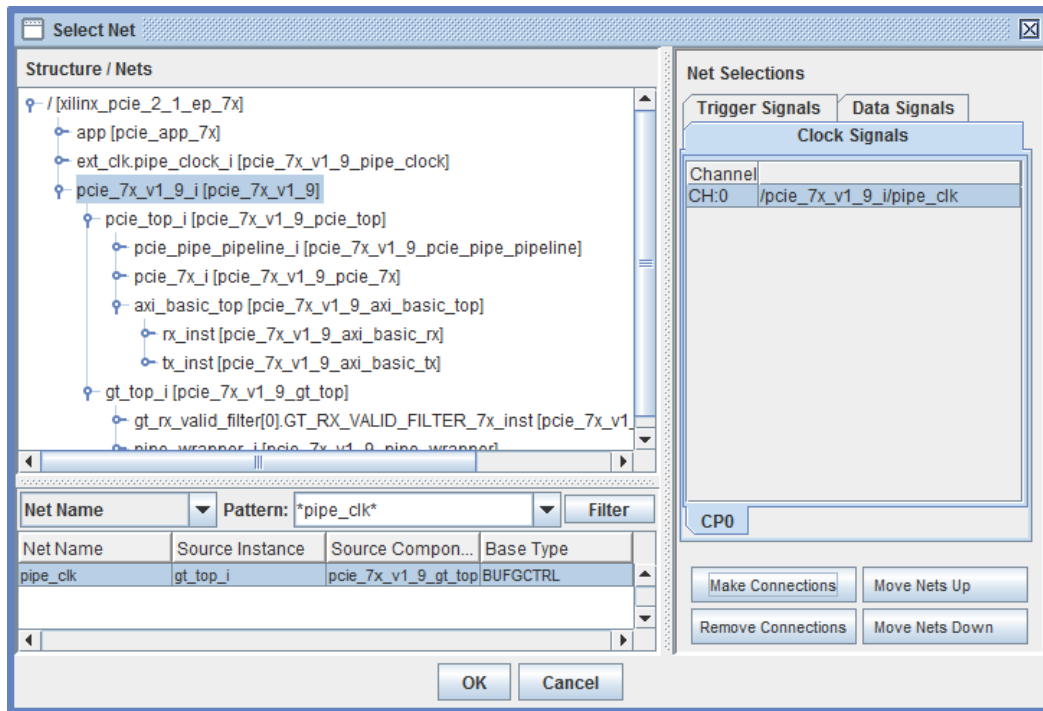


Figure 86: ChipScope Pro Inserter - Selecting Clock Signal

- After the trigger, data, and the clock signals have been selected, click **OK** and then click **Insert>**.

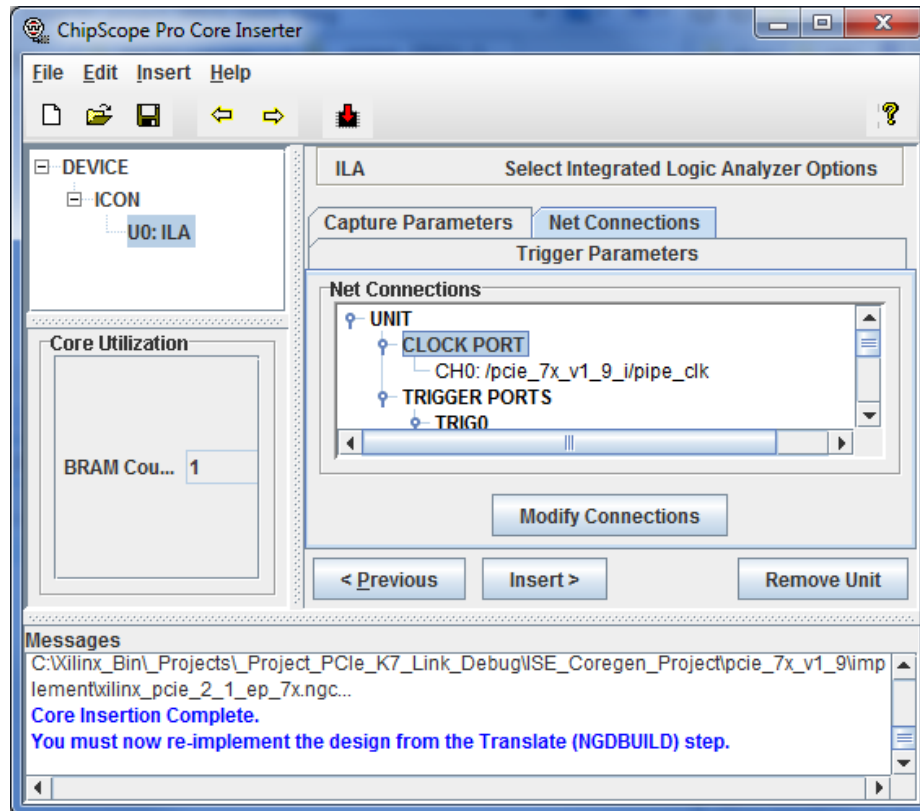


Figure 87: ChipScope Pro Inserter - Final Step, Core Insertion

- A new .ngc file will be generated with the ChipScope Pro core inside the input .ngc file. Before closing the ChipScope Pro inserter, save the project; a CDC file will be generated. This CDC file is required to view the signals in ChipScope Pro Analyzer.
- Re-implement the design by running `implement.bat` or `implement.sh`. Make sure the section of the script with commands to synthesize has been removed. If not, the synthesis will run again and replace the .ngc file that contains the ChipScope Pro core. The implementation script should only contain following:

```

cd results
echo 'Running ngdbuild'
ngdbuild -verbose -uc ../../example_design/xilinx_pcie_2_1_ep_7x_01_lane_gen1_xc7k325t-
ffg676-2-PCIE_X0Y0.ucf xilinx_pcie_2_1_ep_7x.ngc -sd .

echo 'Running map'
map -w -o mapped.ncd xilinx_pcie_2_1_ep_7x.ngd mapped.pcf

echo 'Running par'
par -w mapped.ncd routed.ncd mapped.pcf

echo 'Running trce'
trce -u -e 100 routed.ncd mapped.pcf

echo 'Running design through netgen'
netgen -sim -ofmt verilog -ne -w -tm xilinx_pcie_2_1_ep_7x -sdf_path . routed.ncd

# Uncomment to enable Bitgen. To generate a bitfile, all I/O must be LOC'd to pin.
# Refer to AR 41615 for more information
#echo 'Running design through bitgen'
#bitgen -w routed.ncd

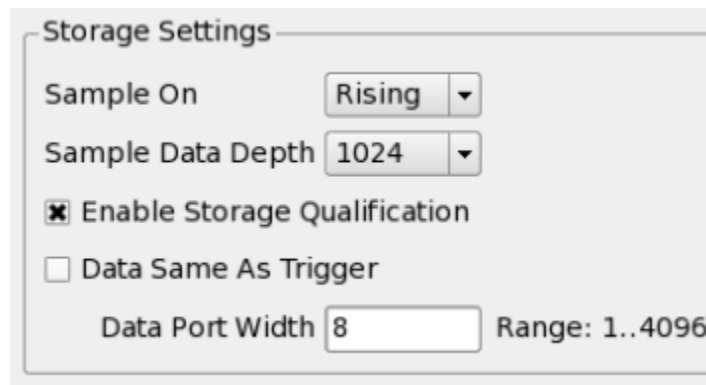
```

Storage Qualification with Chipscope for PCI Express Debug

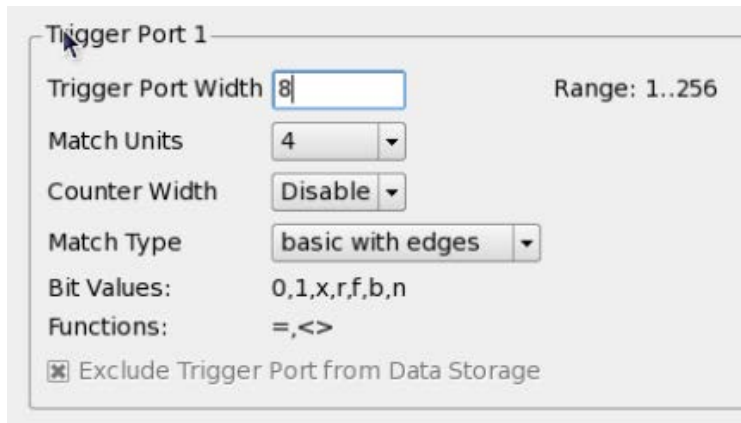
This section describes usage of the storage qualification and sequencer feature of ChipScope for PCIe Debug. This feature is useful in capturing only the LTSSM transitions in ChipScope Pro. During link training, sometimes a Gen2x8 link might come up as Gen1 speed. It would be helpful in debugging to find out whether the link initially trained as Gen1, or it trained to Gen2 first and then to Gen1.

Generating and Customizing the Chipscope ILA / Chipscope Icon

1. While generating the `chipscope_ila` core, on Page 1 - select Enable Storage Qualification under storage settings.



- On Page 2, increase the number of match units (let us use 4 in this example)



- There are no changes needed while generating chipscope_icon

Modifications needed in PCIe wrappers

- Create a storage qualifier (store_ltssm) for cfg_ltssm_state

```
reg [5:0]   cfg_ltssm_state_reg0 = 6'b0;
reg [5:0]   cfg_ltssm_state_reg1 = 6'b0;
reg [5:0]   cfg_ltssm_state_reg2 = 6'b0;
reg        store_ltssm = 1'b0;
```

```
always @ (posedge user_clk)
begin
    cfg_ltssm_state_reg0    <= cfg_ltssm_state;
    cfg_ltssm_state_reg1    <= cfg_ltssm_state_reg0;
    cfg_ltssm_state_reg2    <= cfg_ltssm_state_reg1;
end
```

```
always @ (posedge user_clk)
begin
    if (cfg_ltssm_state_reg0 != cfg_ltssm_state_reg2 )
        store_ltssm    <= 1'b1;
    else
        store_ltssm    <= 1'b0;
end
```

- Instantiate the ChipScope icon and ila in pcie_wrappers

```
assign capture_clock = user_clk;
assign capture_data = {cfg_ltssm_state [5:0], pipe_tx_rate_gt[1:0]};
assign capture_trigger = { cfg_ltssm_state [5:0], store_ltssm};

chipscope_icon icon_0 (.CONTROL0(control0));

chipscope_ila ila_0 (.CONTROL(control0), .DATA(capture_data),
    .TRIG0(capture_trigger), .CLK(capture_clock));
```

Note: pipe_tx_rate_gt[1:0] is not in user_clk domain but is being used for example purposes.

Chipscope captures using Storage Qualification

1. Assign M0 match case to store_ltssm == 1'b1

Trigger Setup - DEV:0 MyDevice0 (XC7VX690T) UNIT:0 MyILA0 (ILA)

Match Unit	Function	Value
M0:TRIG0	==	XXXX_XXX1
TRIG0[7]		X
cfg_ltssm_state[5]		X
cfg_ltssm_state[4]		X
cfg_ltssm_state[3]		X
cfg_ltssm_state[2]		X
cfg_ltssm_state[1]		X
cfg_ltssm_state[0]		X
store_ltssm		1
M1:TRIG0	==	XXXX_XXXX
M2:TRIG0	==	XXXX_XXXX
M3:TRIG0	==	XXXX_XXXX

2. Assign M1 match case to cfg_ltssm == 5'b1 (Detect.Active)

Trigger Setup - DEV:0 MyDevice0 (XC7VX690T) UNIT:0 MyILA0 (ILA)

Match Unit	Function	Value
M0:TRIG0	==	XXXX_XXX1
M1:TRIG0	==	X000_001X
TRIG0[7]		X
cfg_ltssm_state[5]		0
cfg_ltssm_state[4]		0
cfg_ltssm_state[3]		0
cfg_ltssm_state[2]		0
cfg_ltssm_state[1]		0
cfg_ltssm_state[0]		1
store_ltssm		X
M2:TRIG0	==	XXXX_XXXX
M3:TRIG0	==	XXXX_XXXX

3. Assign M2 match case to cfg_ltssm == 5'b2 (Polling. Active)

Trigger Setup - DEV:0 MyDevice0 (XC7VX690T) UNIT:0 MyILA0 (ILA)

Match Unit	Function	Value
M0:TRIG0	==	XXXX_XXX1
M1:TRIG0	==	X000_001X
M2:TRIG0	==	X000_010X
TRIG0[7]		X
cfg_ltssm_state[5]		0
cfg_ltssm_state[4]		0
cfg_ltssm_state[3]		0
cfg_ltssm_state[2]		0
cfg_ltssm_state[1]		1
cfg_ltssm_state[0]		0
store_ltssm		X
M3:TRIG0	==	XXXX_XXXX

4. After all match cases are set up, the window looks as the one in figure below

Trigger Setup - DEV:0 MyDevice0 (XC7VX690T) UNIT:0 MyILA0 (ILA)			
Match Unit	Function	Value	
M0:TRIG0	==		XXXX_XXX1
M1:TRIG0	==		X000_001X
M2:TRIG0	==		X000_010X
M3:TRIG0	==		XXXX_XXXX

- In the trigger window, click on trigger condition equation(M0) and select sequencer and set condition to M1->M2 (Detect.Active->Polling.Active) . This is to ensure that we do not capture redundant Detect.Quiet to Detect.Active transitions during the system power up, before root starts link training.

Add	Active	Trigger Condition Name	Trigger Condition Equation
Del	<input type="radio"/>	TriggerCondition0	M0

Trigger Condition: TriggerCondition0

Boolean **Sequencer**

Number of Levels: Use Contiguous Match Events Only

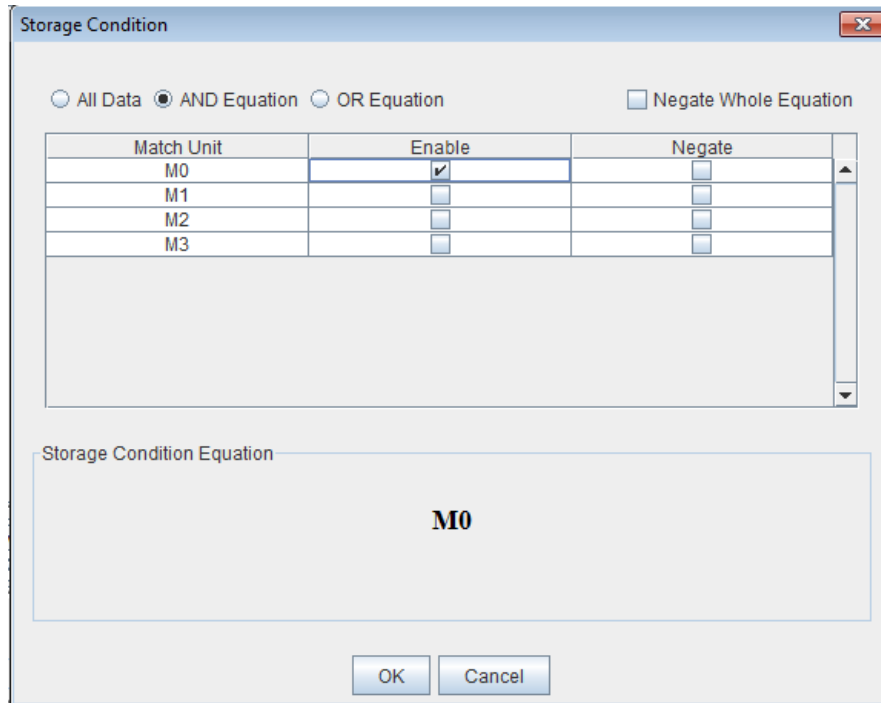
Level	Match Unit	Negate
1	M1	<input type="checkbox"/>
2	M2	<input type="checkbox"/>

Trigger Condition Equation

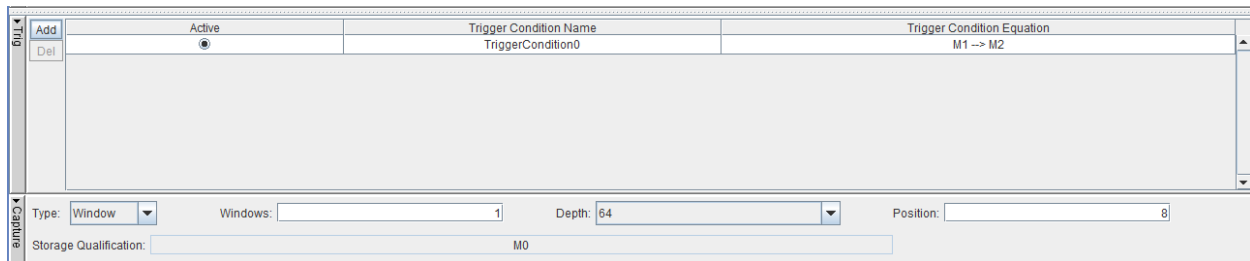
M1 -> M2

- In the capture section, click on **All Data** under storage qualification, select **AND Equation** and enable M0. This will ensure that data will be stored only when LTSSM changes states.

Type: <input type="text" value="Window"/>	Windows: <input type="text" value="1"/>	Depth: <input type="text" value="64"/>	Position: <input type="text" value="0"/>
Storage Qualification: <input type="text" value="All Data"/>			

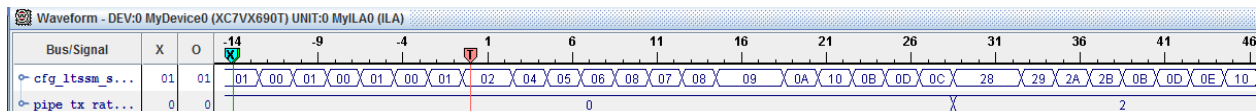


7. The windows should look like the screen shot below once all capture parameters are set correctly.



Note: Select a small depth (ex: 64 in this case) as the buffer will not fill up with large sizes. Adjust the position to ensure that buffer fills up.

8. The captured waveform shows all `cfg_ltssm_state` transitions before training to Gen3 on power up (using storage depth of 64)



Capturing Signals in Vivado ILA

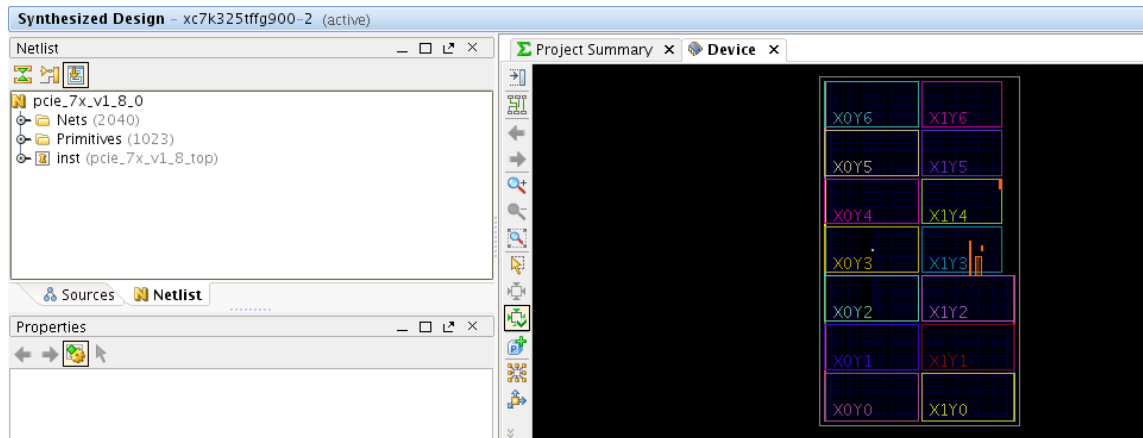


Figure 88 – PCIe Example Design Vivado Project GUI after opening the Synthesized Design

The details on how to debug a design using ChipScope in Vivado Design Suite are provided in [UG936](#). This section illustrates how to grab signals for debugging in the PCIe example design. For more information, please refer to [UG936](#).

Vivado tools allow selecting signals for debugging, same as in ChipScope inserter. There is an additional feature where you could search for specific nets, using wild cards, in the whole design. This is shown in Figure 89. To start grabbing signals for ChipScope, you should first open the synthesized design as shown in Figure 90.



Figure 89 – Search ‘nets’ for Probing in Vivado ILA

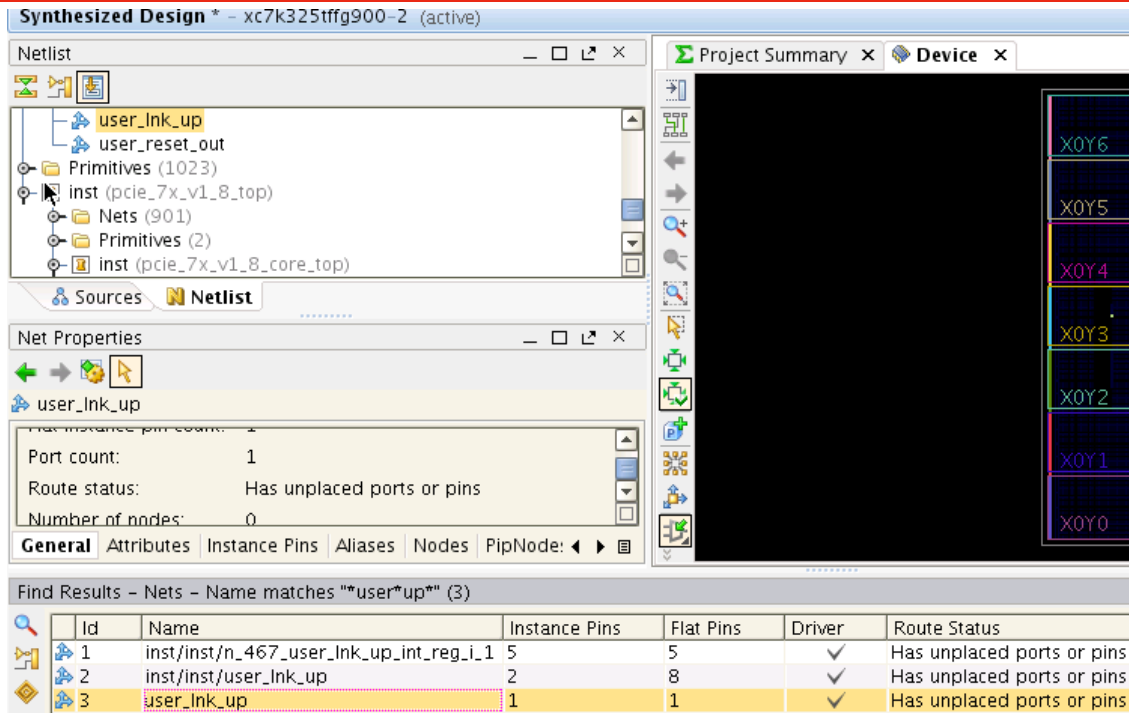


Figure 90 – PCIe Example Design user_ink_up Signal

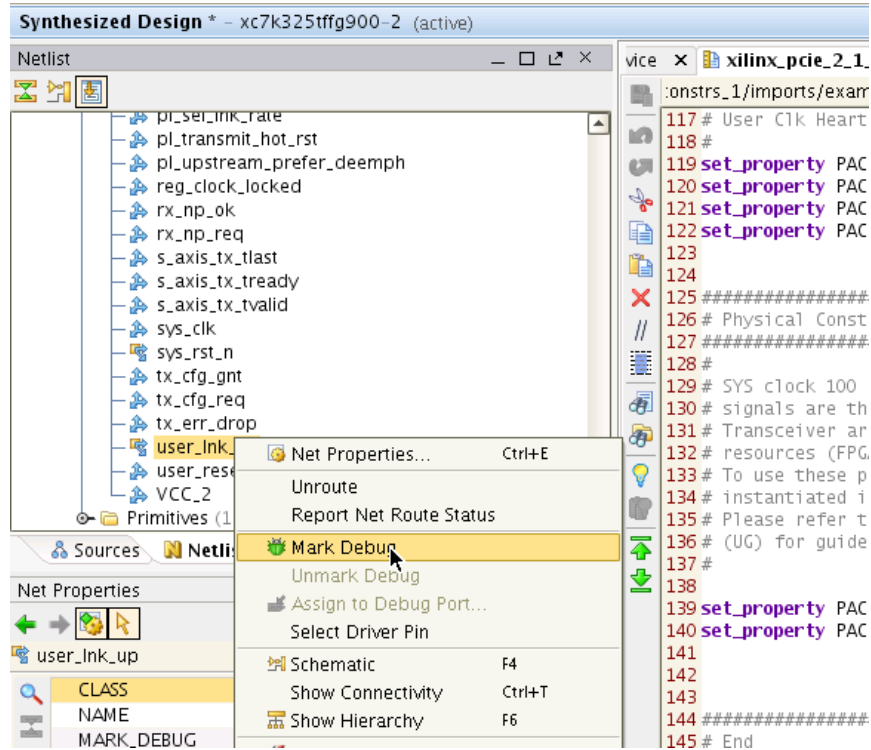


Figure 91 – 'Mark Debug' for Probing user_ink_up in Chipscope

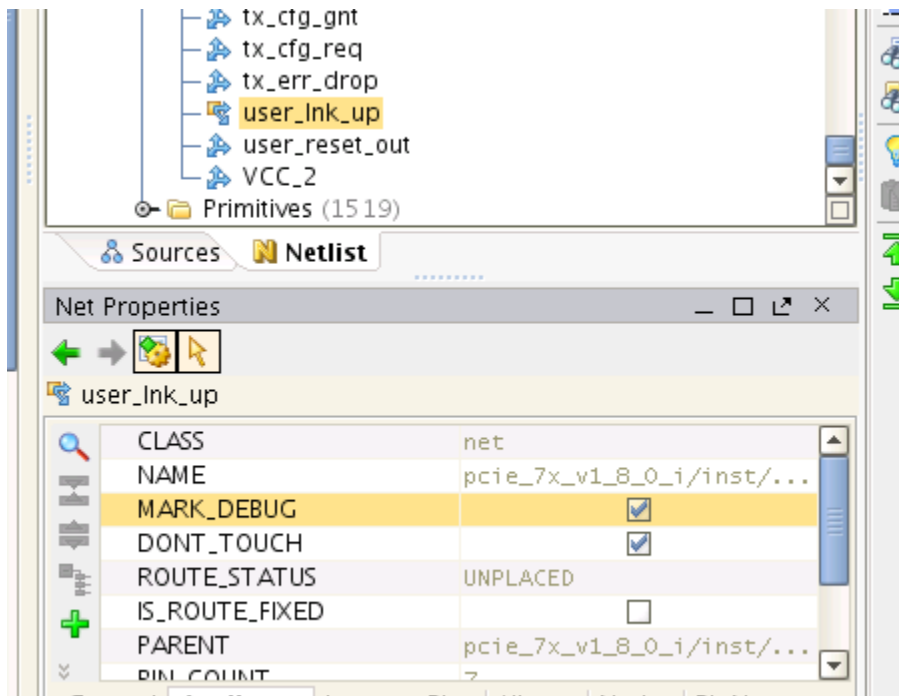
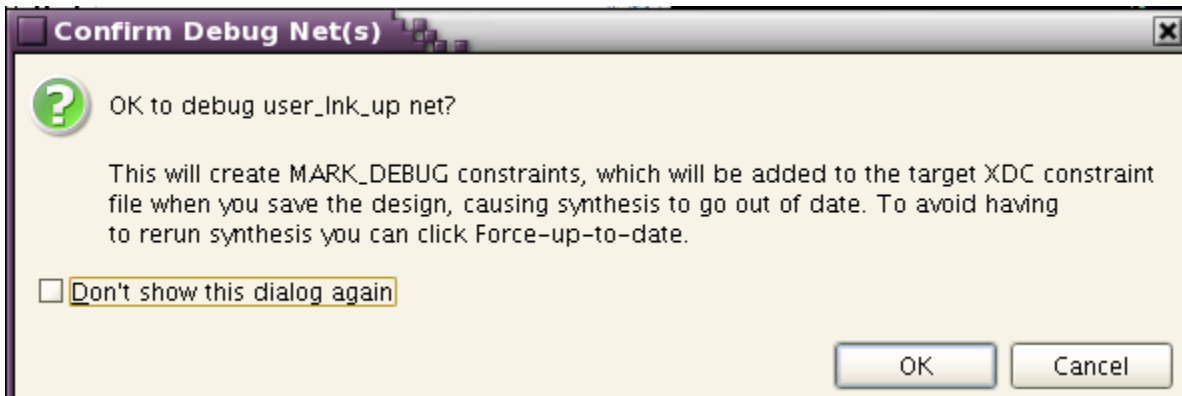


Figure 92 – user_lnk_up Net Properties after enabling 'MARK_DEBUG'

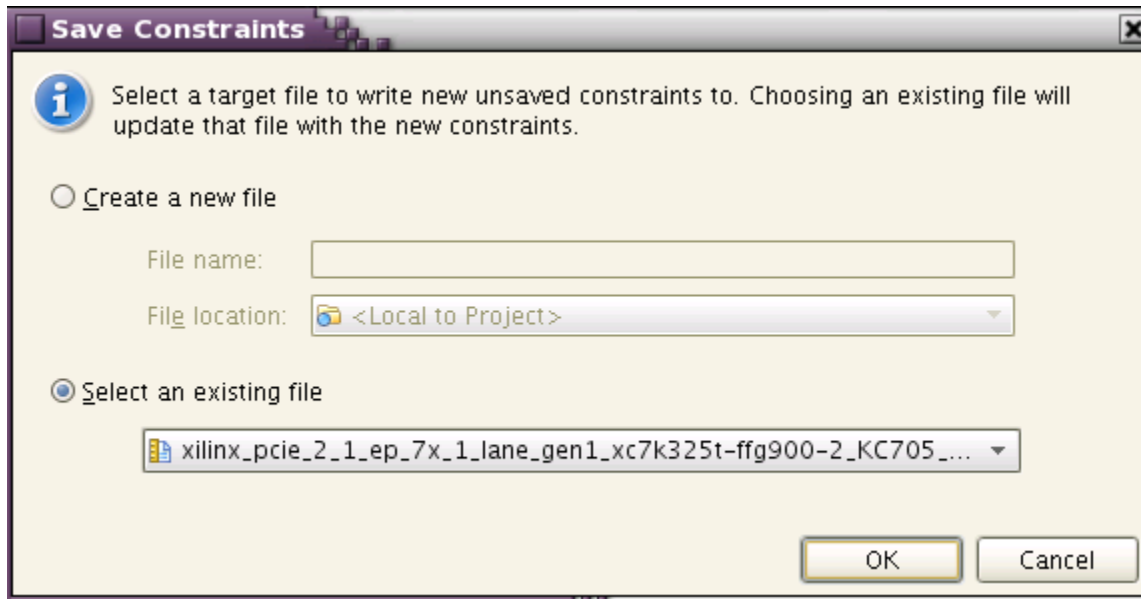


Figure 93 – Saving ‘Mark Debug’ Constraints to the existing XDC file

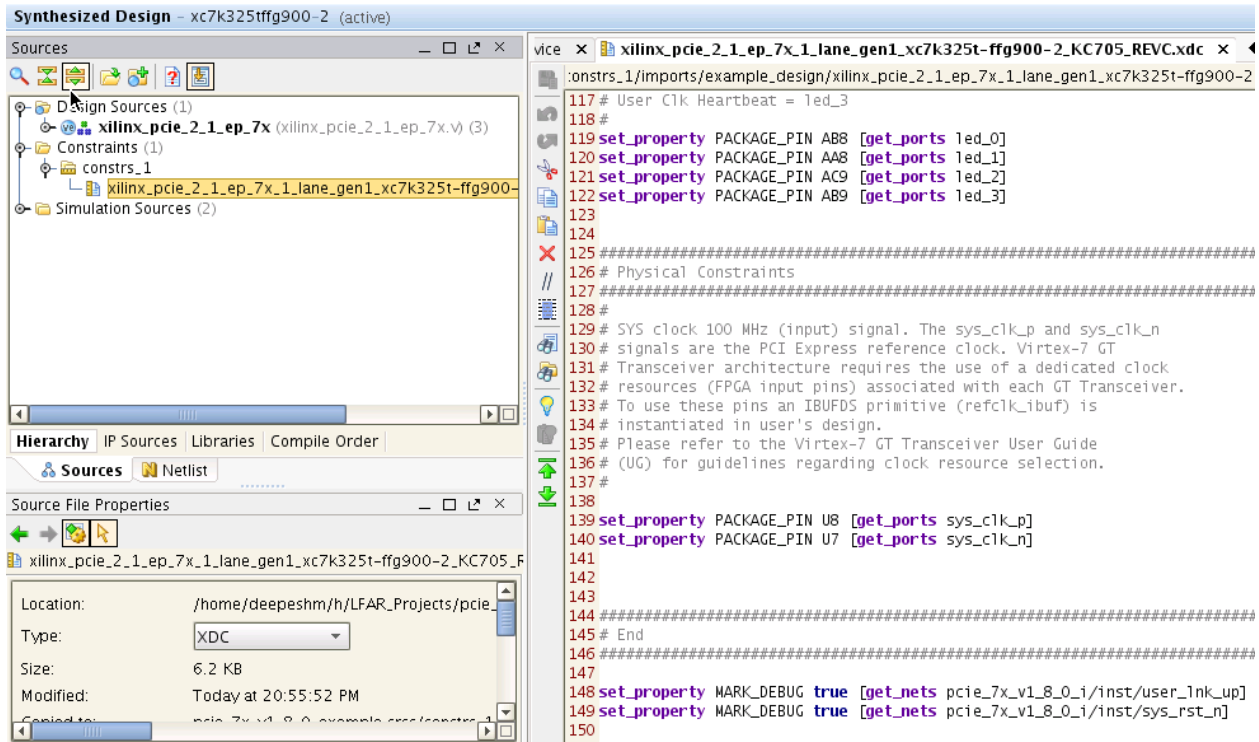


Figure 94 – MARK_DEBUG Constraint in PCIe Example Design XDC File

Generating Debug Cores (Set up Debug)

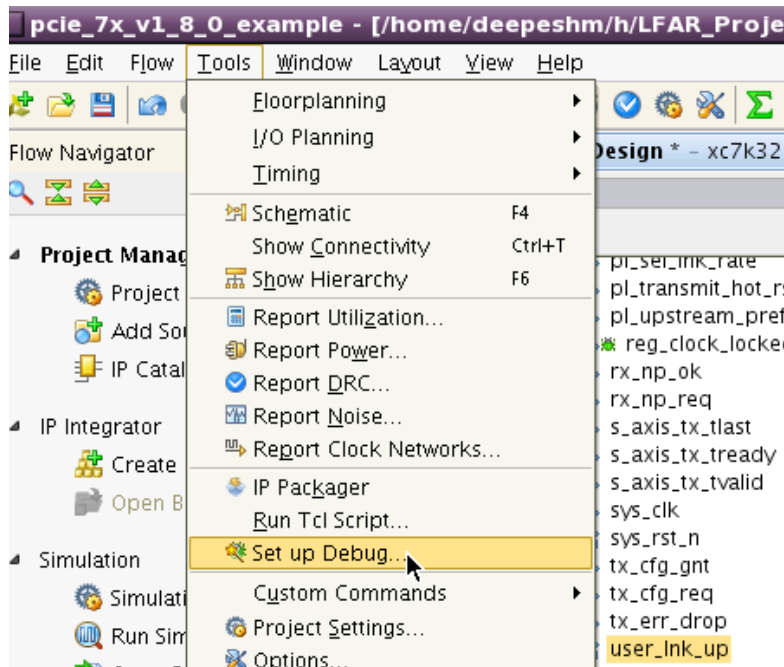


Figure 95 – Generating Debug Cores

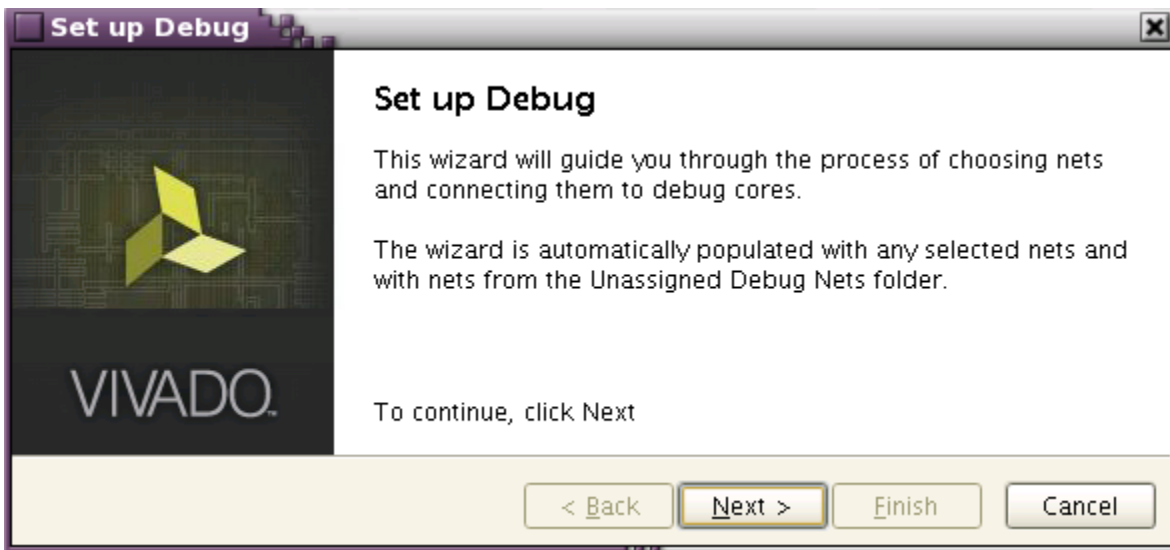


Figure 96 – Setup Debug GUI

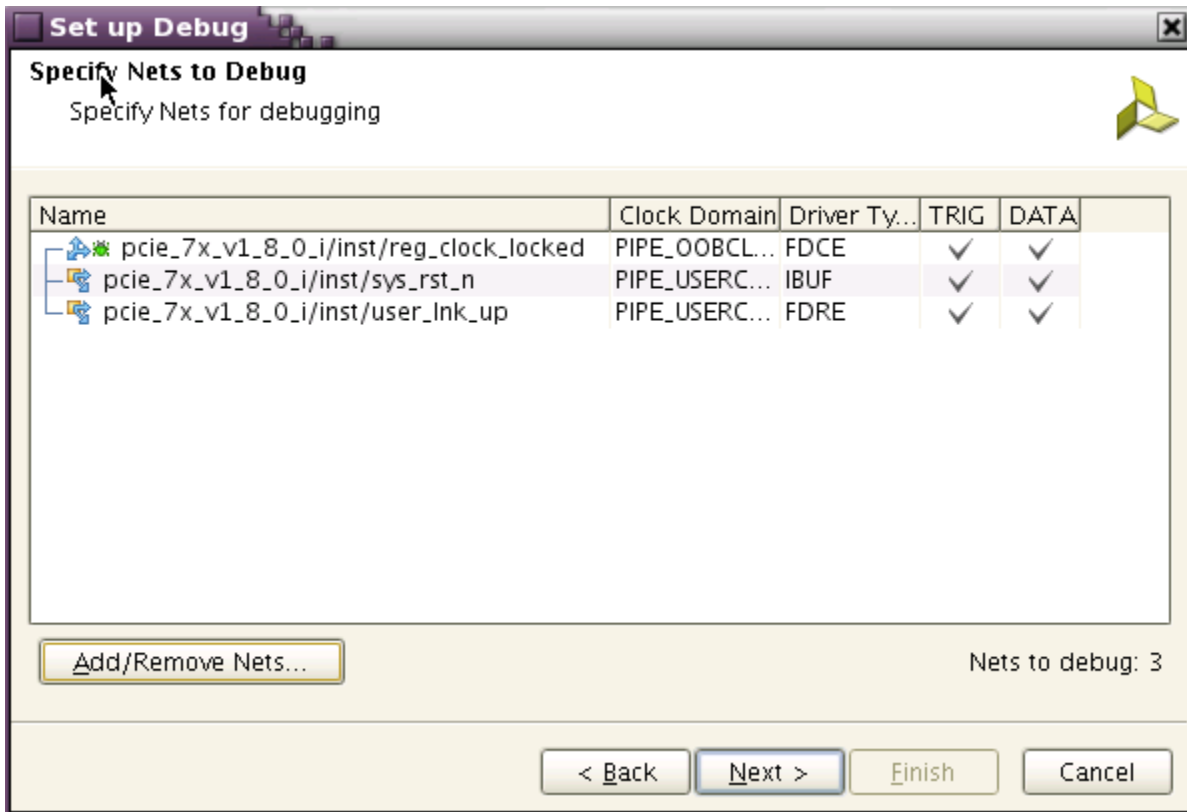


Figure 97 – Selected Nets for Probing in Chipscope

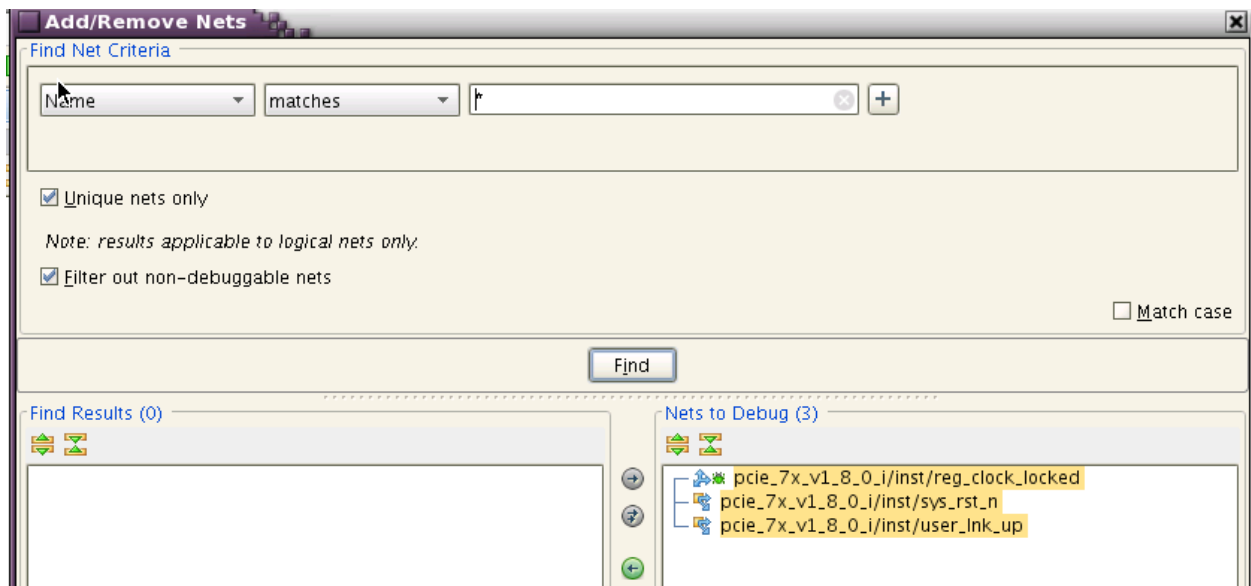


Figure 98 – Add/Remove nets in Setup Debug

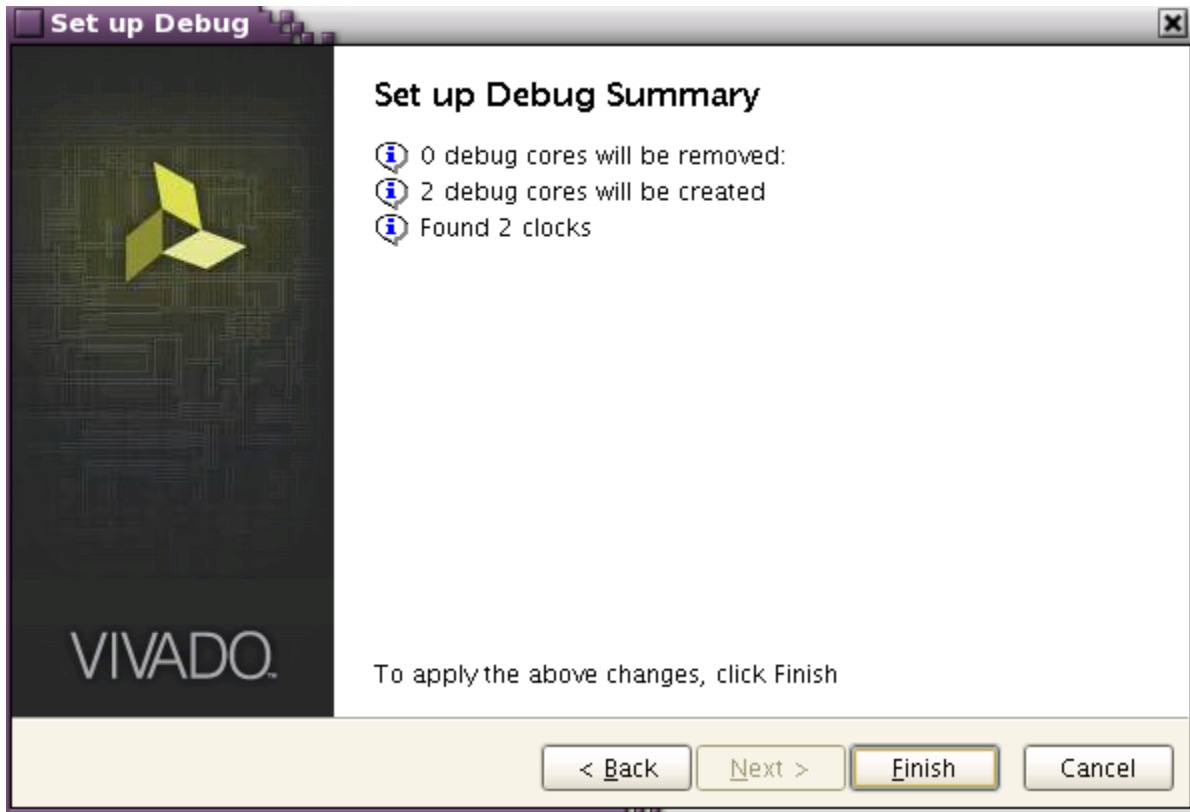


Figure 99 - Set up Debug Summary

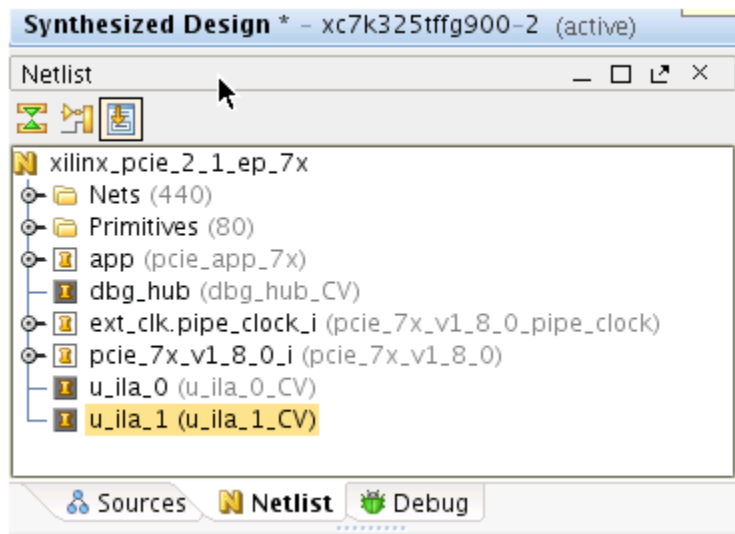


Figure 100 – Chipscope Debug Cores in Netlist Window

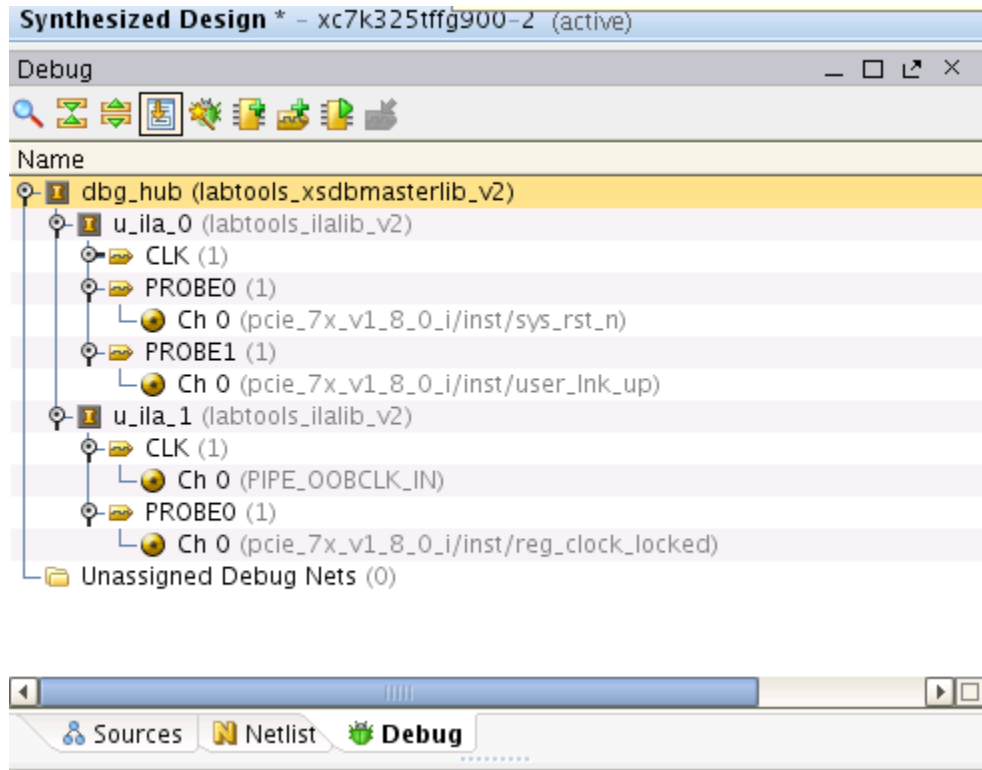


Figure 101 – PCIe Example Design Selected Debug Signals in ‘Debug’ Window

Setting up the ILA Core to Take a Measurement

The ILA core(s) that you add to your design appear in the Hardware window under the target device as shown in Figure 102. If you do not see the ILA core(s) appear, right click on the device and select Refresh Hardware. This re-scans the FPGA device and refreshes the Hardware window.

Setting the ILA Core Trigger Condition

Use the Trigger Cond control in the Hardware window (or the Trigger Condition property in the ILA Core Properties window) to select between “AND” and “OR” settings. The “AND” setting causes a trigger event when all of the ILA probe comparisons are satisfied. The “OR” setting causes a trigger event when any of the ILA probe comparisons are satisfied. You can also use the `set_property` Tcl command to change the ILA core trigger condition:

```
set_property CONTROL.TRIGGER_CONDITION AND [get_hw_ilas hw_ila_1]
```

Reading ILA Probes Information

The ILA probe file is automatically associated with the FPGA hardware device if the probes file is called `debug_nets.ltx` and is found in the same directory as the bitstream programming (`.bit`) file that is associated with the device.

You can also specify the location of the probes file:

1. Select the FPGA device in the Hardware window.
2. Set the Probes file location in the Hardware Device Properties window.
3. Click **Apply** to apply the change.

Running or Arming the ILA Core Trigger

You can run or arm the ILA core trigger in two different modes:

- **Run Trigger:** Selecting the ILA core to be armed, followed by clicking the **Run Trigger** button on the Hardware window toolbar arms the ILA core to detect the trigger event that is defined by the ILA core trigger condition and probe compare values.
- **Run Trigger Immediate:** Selecting the ILA core to be armed, followed by clicking the **Run Trigger Immediate** button on the Hardware window toolbar arms the ILA core to trigger immediately regardless of the settings of the ILA core trigger condition and probe compare values. This command is useful for capturing any values that present at the probe inputs of the ILA core.

You can also arm the trigger by selecting and right clicking the ILA core and selecting **Run Trigger** or **Run Trigger Immediate** from the popup menu as shown in Figure 102.

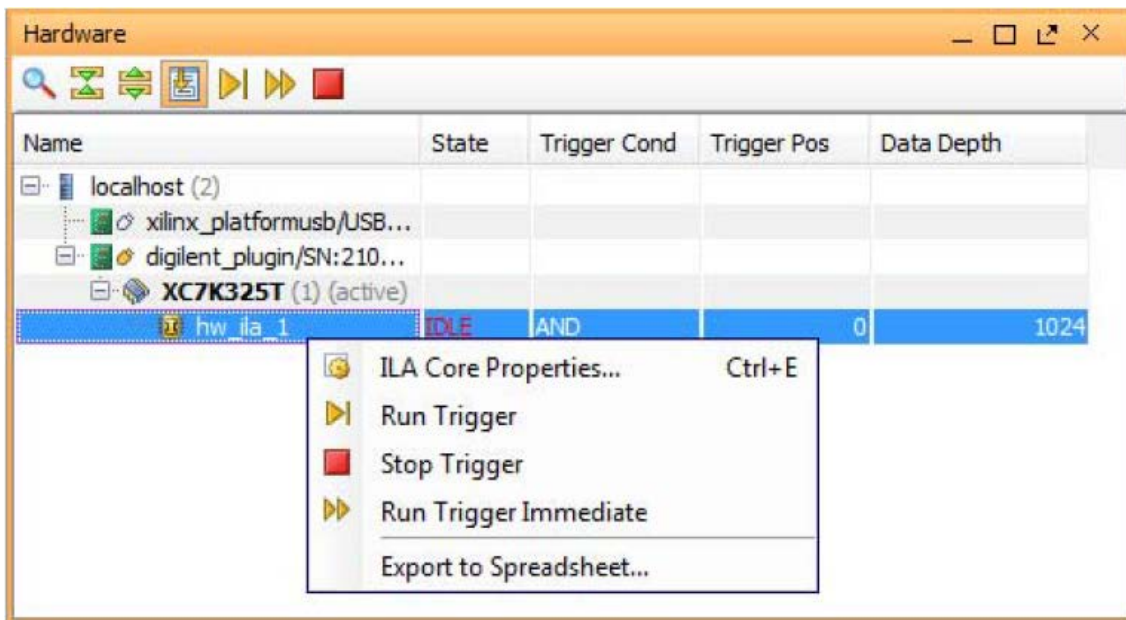


Figure 102 - Vivado Integrated Design Environment Hardware Window

Stopping the ILA Core Trigger

You can stop the ILA core trigger by selecting the appropriate ILA core, followed by clicking the **Stop Trigger** button on the **Hardware** window toolbar. You can also stop the trigger by selecting and right clicking the appropriate ILA core and selecting **Stop Trigger** from the popup menu.

Viewing Captured Data from the ILA Core in the Waveform Viewer

Once the ILA core captured data has been uploaded to the Vivado Integrated Design Environment, it is displayed in the Waveform Viewer. See *Viewing ILA Probe Data Using Waveform Viewer (ug908)* for details on using the Waveform Viewer to view captured data from the ILA core.

Saving and Restoring Captured Data from the ILA Core

In addition to displaying the captured data that is directly uploaded from the ILA core, you can also write the captured data to a file then read the data from a file and display it in the waveform viewer.

Saving Captured ILA Data to a File

Currently, the only way to upload captured data from an ILA core and save it to a file is to use the following Tcl command:

```
write_hw_ila_data my_hw_ila_data_file [upload_hw_ila_data hw_ila_1]
```

Restoring Captured ILA Data from a File

Currently, the only way to restore captured data from a file and display it in the waveform viewer is to use the following Tcl command:

```
display_hw_ila_data [read_hw_ila_data my_hw_ila_data_file]
```

Special Symbols for Framing and Link Management

Special Symbols are distinct from the Data Symbols. This is part of 8b/10b encoding scheme that is used to represent control characters. These symbols are not scrambled, and hence readable on RX/TX GT interface. These Special Symbols are used for various Link Management purposes.

Table 12 - Special Symbols

Encoding	Symbol	Name	Description
K28.5	COM	Comma	Used for Lane and Link initialization and management
K27.7	STP	Start TLP	Marks the start of a Transaction Layer Packet
K28.2	SDP	Start DLLP	Marks the start of a Data Link Layer Packet
K29.7	END	End	Marks the end of a Transaction Layer Packet or a Data Link Layer Packet
K30.7	EDB	EnD Bad	Marks the end of a nullified TLP
K23.7	PAD	Pad	Used in Framing and Link Width and Lane ordering negotiations
K28.0	SKP	Skip	Used for compensating for different bit rates for two communicating Ports
K28.1	FTS	Fast Training Sequence	Used within an Ordered Set to exit from L0s to L0
K28.3	IDL	Idle	Used in the Electrical Idle Ordered Set (EIOS)
K28.4			Reserved
K28.6			Reserved
K28.7	EIE	Electrical Idle Exit	Reserved in 2.5 GT/s Used in the Electrical Idle Exit Ordered Set (EIEOS) and sent prior to sending FTS at speeds other than 2.5 GT/s

Data Byte Name	Data Byte Value	Bits HGF EDCBA	Current RD - abcdei fghj	Current RD + abcdei fghj
K28.0	1C	000 11100	001111 0100	110000 1011
K28.1	3C	001 11100	001111 1001	110000 0110
K28.2	5C	010 11100	001111 0101	110000 1010
K28.3	7C	011 11100	001111 0011	110000 1100
K28.4	9C	100 11100	001111 0010	110000 1101
K28.5	BC	101 11100	001111 1010	110000 0101
K28.6	DC	110 11100	001111 0110	110000 1001
K28.7	FC	111 11100	001111 1000	110000 0111
K23.7	F7	111 10111	111010 1000	000101 0111
K27.7	FB	111 11011	110110 1000	001001 0111
K29.7	FD	111 11101	101110 1000	010001 0111
K30.7	FE	111 11110	011110 1000	100001 0111

Conclusion

This document provides description about debugging PCIe link training issues in reference to 7 Series Integrated PCI Express block core. It is expected that after going through this document, a user will be able to debug link training issues on their own. If the issue is still unresolved, please create a WebCase with Xilinx Technical Support. Attach all of the captured ChipScope Pro/Vivado ILA waveforms, Lecroy Captures (if any), and the details of your investigation and analysis along with answer to the questions in section “***What if the issue is still not resolved?***”.

References

1. [DS182](#), Kintex-7 FPGAs Data Sheet: DC and AC Switching Characteristics
2. [UG476](#), 7 Series FPGAs GTX/GTH Transceivers User Guide
3. [PG054](#), 7-Series Integrated PCI Express Block Product Guide
4. [WP419](#), Equalization for High-Speed Serial Interfaces in Xilinx 7-Series FPGA Transceivers
5. [UG936](#), Vivado Design Suite Tutorial, Programming and Debugging

Revision History

07/29/2013 - Initial release