

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_lnk_up is asserted during this state. Note that user_lnk_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.

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.

Table 1 - TS1 Ordered Set

Symbol Number	Encoded Values	Description
Number D2.0, D2.2, D2 4 D2.0, D6.0, D6 D6.4, D6.6 D6.4, D6.6	Encoded Values	Description Data Rate Identifier Bit 0 – Reserved, set to 0b. Bit 1 – When set to 1b, indicates 2.5 GT/s data rate supported. 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). Bit 3:5 – Reserved, must be set to 0b. Bit 6 (Autonomous Change) – 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. In Recovery state, this bit indicates the de-emphasis preference of the Downstream component.
		preference of the Downstream component. 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. 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. This bit is reserved in all other states for a Downstream
		component. 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. This bit is reserved for all other states.
		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. 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.



Symbol Number	Encoded Values	Description
5	D0.0, D1.0, D2.0, D4.0, D8.0, D16.0, D20.0	Training Control $\underline{Bit 0} - Hot Reset$ $Bit 0 = 0b, De-assert$ $Bit 0 = 1b, Assert$ $\underline{Bit 1} - Disable Link$ $Bit 1 = 0b, De-assert$ $Bit 1 = 1b, Assert$ $\underline{Bit 2} - Loopback$ $Bit 2 = 0b, De-assert$ $Bit 2 = 1b, Assert$ $\underline{Bit 3} - Disable Scrambling$ $Bit 3 = 0b, De-assert$ $Bit 3 = 0b, De-assert$ $Bit 3 = 1b, Assert$ $\underline{Bit 4} - Compliance Receive$ $Bit 4 = 0b, De-assert$ $Bit 4 = 1b, Assert$ $Disable Scrambling$ $Dit 4 = 1b, Assert$ $Dit 4 = 0b, De-assert$ $Bit 4 = 1b, Assert$ Components that support 5 GT/s data rate must implementthis bit as specified. Components that support only 2.5 GT/sdata rate may optionally implement this bit as a Receiver. Ifnot implemented for components that support only 2.5 GT/sdata rate, this bit will be reserved and must behave as if thecomponent received a 0b in this bit position. $\underline{Bit 5:7 - Reserved}$ Set to 0b
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 (VDD - VSWING/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 rxelecidle 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 rxelecidle signals as the number of lanes. Trigger on assertion of each rxelecidle signal. If it does trigger on one of the lanes but rxelecidle 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 <u>DS182</u> (Kintex-7 FPGAs Data Sheet: DC and Switching Characteristics) and <u>DS183</u> (*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 <u>DS182</u> (*Kintex-7 FPGAs Data Sheet: DC and Switching Characteristics*) and <u>DS183</u> (*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	 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 <u>DS182</u>, <i>Kintex-7 FPGAs Data Sheet: DC and Switching Characteristics</i> or <u>DS183</u>, <i>Virtex-7 FPGAs Data Sheet: DC and Switching Characteristics</i>. If reference clock input is not used, leave the associated pin pair unconnected. 		
MGTXRXP[3:0]/MGTXRXN[3:0] MGTHRXP[3:0]/MGTHRXN[3:0]	 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. 		
MGTXTXP[3:0]/MGTXTXN[3:0] MGTHTXP[3:0]/MGTHTXN[3:0]	 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	 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	 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]	• The nominal voltage is 1.0 VDC.	
moniveelity	 See <u>DS182</u>, Kintex-7 FPGAs Data Sheet: DC and Switching Characteristics or <u>DS183</u>, Virtex-7 FPGAs Data Sheet: DC and Switching Characteristics 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, 7 Series FPGAs Packaging and Pinout Specifications. 	
	 The following filter capacitor is recommended: 1 of 4.7 vF 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 <u>www.xilinx.com/power</u>. 	
MGTAVTT[N]	The nominal voltage is 1.2 VDC.	
	 See <u>DS182</u>, Kintex-7 FPGAs Data Sheet: DC and Switching Characteristics or <u>DS183</u>, Virtex-7 FPGAs Data Sheet: DC and Switching Characteristics 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 <u>UG475</u> , 7 Series FPGAs Packaging and Pinout Specifications.	
	 The following ceramic filter capacitor is recommended: 1 of 4.7 yE 10% 	
	 For optimal performance, power supply poise must be less than 10 mVpp. 	
	 If all of the Quads in a power supply group are not used, the associated power pips can be left unconnected or tied to ground. 	
	 For power consumption, refer to the XPower Estimator (XPE) for 7 series devices at <u>www.xilinx.com/power</u>. 	
MGTVCCAUX[N]	The nominal voltage is 1.8 VDC.	
	 See <u>DS182</u>, Kintex-7 FPGAs Data Sheet: DC and Switching Characteristics or <u>DS183</u>, Virtex-7 FPGAs Data Sheet: DC and Switching Characteristics 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 <u>UG475</u> , 7 Series FPGAs Packaging and Pinout Specifications.	
	 The following filter capacitor is recommended: 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 pine 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: LO	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



Select Net				
Structure / Nets	Structure / Nets			
•/ [xilinx_pcie_2_1_	ep_7x]			
- app (pcie_app	_7x]			
🗢 ext_clk.pipe_c	lock_i (pcie_7x_v1_9_pip	e_clock]		
pcie_7x_v1_9	_i [pcie_7x_v1_9]			
← pcie top i	[pcie 7x v1 9 pcie top	1		
⊶ at top i[p	cie 7x v1 9 at top]	•		
5-2-4-2-14				-
•				
Net Name	Pattern: *itssm*		► FI	Iter
Net Name	Source Instance	Source Component	Base Type	
pl_ltssm_state<5>	pcie_7x_v1_9_i	pcie_7x_v1_9	PCIE_2_1	
pl_ltssm_state<4>	pcie_7x_v1_9_i	pcie_7x_v1_9	PCIE_2_1	
pl_ltssm_state<3>	pcie_7x_v1_9_i	pcie_7x_v1_9	PCIE_2_1	
pl_ltssm_state<2>	pcie_7x_v1_9_i	pcie_7x_v1_9	PCIE_2_1	
pl_ltssm_state<1>	pcie_7x_v1_9_i	pcie_7x_v1_9	PCIE_2_1	
pl_ltssm_state<0>	pcie_7x_v1_9_i	pcie_7x_v1_9	PCIE_2_1	

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.

Structure / Nets Net Selections	
Structure / Nets • / [xilinx_pcie_2_1_ep_7x] • app [pcie_app_7x] • ext_clk.pipe_clock.i [pcie_7x_v1_9_pipe_clock] • pcie_7x_v1_9_i[pcie_7x_v1_9_pipe_clock] • pcie_17x_v1_9_i[pcie_7x_v1_9_pipe_clock] • pcie_17x_v1_9_pipe_clock_i [pcie_7x_v1_9_pcie_top] • pcie_top_i [pcie_7x_v1_9_pcie_top] • pcie_top_i [pcie_7x_v1_9_pcie_top] • pcie_17x_i [pcie_7x_v1_9_pcie_top] • pcie_17x_i [pcie_7x_v1_9_pcie_7x] • axi_basic_top [pcie_7x_v1_9_axi_basic_top] • gt_top_i [pcie_7x_v1_9_atub] • gt_top_i [pcie_7x_v1_9_gt_top] • pipe_wrapper_i [pcie_7x_v1_9_pipe_wrapper] • dt_tilt_pcie_7x_v1_9_ipcie_top_i/pipe_rx1_status_gt<2> CH:6 (H:6 [pcie_7x_v1_9_ipcie_top_i/pipe_rx1_status_gt<2> CH:6 [pcie_7x_v1_9_ipcie_top_i/pipe_rx1_status_gt<2> CH:6 [pcie_7x_v1_9_ipcie_top_i/pipe_rx1_status_gt<2> CH:6 [pcie_7x_v1_9_ipcie_top_i/pipe_rx1_status_gt<2> CH:6 [pcie_7x_v1_9_ipcie_top_i/pipe_rx1_status_gt<2> CH:6 [pcie_7x_v1_9_ipcie_top_i/pipe_rx1_status_gt<2> CH:6 [pcie_7x_v1_9_ipcie_top_i/pipe_rx1_status_gt<2> [CH:7 [pcie_7x_v1_9_ipcie_top_i/pipe_rx1_status_gt<2> [CH:7 [pcie_7x_v1_9_ipcie_top_i/pipe_rx1_status_gt<2> [CH:7 [pcie_7x_v1_9_ipcie_top_i/pipe_rx1_status_gt<2> [CH:7 [pcie_7x_v1_9_ipcie_top_i/pipe_rx1_status_gt<2> [CH:7 [pcie_7x_v1_9_ipcie_top_i/pipe_rx1_status_gt<2> [CH:7 [pcie_7x_v1_9_ipcie_top_i/pipe_rx2_status_gt<2> [CH:7 [pcie_7x_v1_9_ipcie_top_i/pipe_rx2_status_gt<2> [CH:7 [pcie_7x_v1_9_ipcie_top_i/pipe_rx2_status_gt<2> [CH:7 [pcie_7x	TECTRX
CH:12 //pcie_7x_v1_9_i/pcie_top_i/pipe_rx2_status_gt<1> CH:13 //pcie_7x_v1_9_i/pcie_top_i/pipe_rx2_status_gt<0>	
CH:14 /pcie_/x_v1_9_/pcie_top_/pipe_rx3_status_gt<2> CH:15 /pcie_7x_v1_9_/pcie_top_//pipe_rx3_status_gt<1>	
CH:16 /pcie_7x_v1_9_i/pcie_top_i/pipe_rx3_status_gt<0>	
CH:17 //pcie_rx_v1_9_//pcie_top_//pipe_tx0_elec_lale CH:18 //pcie_7x_v1_9_//pcie_top_i//pipe_tx1_elec_idle	
CH:19 /pcie_7x_v1_9_i/pcie_top_i/pipe_tx2_elec_idle	

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

Structure / Nets		lections
•-/[xilinx_pcie_2_1_ep_7x]	Clock	k Signals Trigger Signals Data Signals
🗠 app [pcie_app_7x]	Chapr	
— ext_clk.pipe_clock_i [pcie_7x_v1_9_pipe_clock]	CH:0	/pcie 7x v1 9 i/pcie top i/user reset
Pcie_7x_v1_9_i [pcie_7x_v1_9]	CH:1	/pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_reset.pipe_reset_i/cpllreset
pcie_top_i [pcie_7x_v1_9_pcie_top]	CH:2	/pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_CPLLLOCK
- ncie nine nineline i Incie 7x v1 9 ncie nine ninelinel	CH:3	/pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/gt_rxresetdone
polo_pipo_pipointo_ (polo_rit_o_polo_pipo_pipointo)	CH:4	/pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/gt_txresetdone
pcie_/x_i [pcie_/x_v1_9_pcie_/x]	CH:5	/pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_RXUSERRDY
axi_basic_top [pcie_7x_v1_9_axi_basic_top]	CH:6	/pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_TXUSERRDY
• at top i [pcie 7x v1 9 at top]	CH:7	/pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_GTRXRESET
at re valid filter[0] CT_RV_VALID_EILTER_7x_inst[ncis_7y		/pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_GTTXRESET
	CH:9	/pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_PHYSTATUS
 pipe_wrapper_i [pcie_/x_v1_9_pipe_wrapper] 	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.







Signal Set-4, Clocking



Figure 19 - PCIe Clocking Related Signals

Signal Set-5, GT RX

🛗 Select Net		
Structure / Nets	Net Selections	
<pre> • / [xilinx_pcie_2_1_ep_7x] </pre>	Clock Signals Trigger Signals Data Signals	
🕶 app [pcie_app_7x]		_
– ext_clk.pipe_clock_i [pcie_7x_v1_9_pipe_clock]		_
• pcie 7x v1 9 i [pcie 7x v1 9]	CH:1 /pcie_7x_v1_9_igt_top_ipipe_wrapper_i/PIPE_RXDATA<0>	
• ncie ton i [ncie 7x v1 9 ncie ton]	CH:2	
a dt ten i [neie Zv. v1 0 dt ten]	CH:3 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/PIPE_RXDATA<31>	
Y gt_top_tpcte_/x_v1_9_gt_topj	CH:4 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/PIPE_RXDATAK<0>	
gt_rx_valid_filter[0].G1_RX_vALID_FILTER_7x_inst [pcie_7x	CH:5 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/PIPE_RXDATAK<1>	
pipe_wrapper_i [pcie_7x_v1_9_pipe_wrapper]	CH:6 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/PIPE_RXDATAK<2>	
pipe_lane[0].pipe_user_i [pcie_7x_v1_9_pipe_user]	CH:7 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/PIPE_RXDATAK<3>	
🔶 pipe_lane[0].gt_wrapper_i [pcie_7x_v1_9_gt_wrapper]	CH:8 /pcie_/X_V1_9_//gt_top_//pipe_wrapper_//PIPE_RXELECIDLE<0>	
pipe lane[0].pipe drp.pipe drp i [pcie 7x v1 9 pipe o	CH:10 /pcie_7x_v1_9_/gl_top_//pipe_w1apper_//PIPE_RAPOLARITISOP	
pipe Jane[0] pipe eq pipe eq i [pcie 7x v1 9 pipe eq	CH:11 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/PIPE_RXSTATUS<0>	
pipe lane[0] pipe guad pipe common gpll drn i [ncie]	CH:12 /pcie 7x v1 9 i/ot top i/pipe wrapper i/PIPE RXSTATUS<2>	
pipe_lane[0].pipe_quad.pipe_common.qpii_uip_r[pole_	CH:13 /pcie 7x v1 9 i/gt top i/pipe wrapper i/pipe lane[0].gt wrapper i/GT RXBUFSTATUS<0>	
pipe_lane[0].pipe_quad.pipe_common.qpii_wrapper_i [CH:14 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_RXBUFSTATUS<1>	
pipe_lane[0].pipe_rate.pipe_rate_i [pcie_7x_v1_9_pipe_	CH:15 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_RXBUFSTATUS<2>	
pipe_lane[0].pipe_sync_i [pcie_7x_v1_9_pipe_sync]	CH:16 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_RXBYTEISALIGNED	
🔶 pipe_reset.pipe_reset_i [pcie_7x_v1_9_pipe_reset]	CH:17 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_RXCDRLOCK	
optimized upped to a set of the set of th	CH:18 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_RXCHANISALIGNED	
	CH:19 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_RXCDRRESET	
	CH:20 /pcie_7x_v1_9_//gt_top_//pipe_wrapper_//pipe_lane[0].gt_wrapper_//GT_RXCHARISCOMMA<0	2
	CH:21 /pcie_7x_v1_9_/gt_top_//pipe_wrapper_//pipe_lane[0].gt_wrapper_//GT_RXCHARISCOMMA<1	2
	CH.22 /pcie_7X_V1_9_/yt_top_//pipe_wiapper_//pipe_lane[0].gt_wiapper_//GT_RXCHARISCOMMA<2	5
	CH:24 /pcie_7x_v1_9_igt_top_ipipe_wrapper_ipipe_lane[0].gt_wrapper_i/oT_txtchAki3cOmmA<5	-
	CH:25 /pcie 7x v1 9 i/ot top i/pipe wrapper i/pipe lane[0].ot wrapper i/GT RXELECIDLE	
	CH:26 /pcie 7x v1 9 i/gt top i/pipe wrapper i/pipe lane[0].gt wrapper i/GT RXPHALIGN	
	CH:27 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_RXPHALIGNDONE	
	CH:28 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_RXVALID	
—		_
	DPO	

Figure 20 – GT Receive Channel Signals

Signal Set-6, GT TX

Structure / Nets	Net Selections
<pre> / [xilinx_pcie_2_1_ep_7x] </pre>	Clock Signals Trigger Signals Data Signals
🕶 app [pcie_app_7x]	Channel
- ext_clk.pipe_clock_i [pcie_7x_v1_9_pipe_clock]	CH:30 /pcie 7x v1 9 i/gt top i/pipe wrapper i/pipe lane[0].gt wrapper i/GT GTTXRESET
Pcie_7x_v1_9_i [pcie_7x_v1_9]	CH:31 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_TXDATA<0>
pcie_top_i [pcie_7x_v1_9_pcie_top]	CH:32
• at top i[pcie 7x v1 9 at top]	CH:33 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_TXDATA<31>
at rx valid filter(0) GT_RX_VALID_FILTER_7x_inst[ncie_7x	CH:34 //pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_TXDATAK<0>
pipe wronner i Incie 7x v1 0 pipe wronner	CH:35 /pcie_/x_v1_9_//gt_top_//pipe_wrapper_//pipe_lane[0].gt_wrapper_//GT_TXDATAK<1>
Pipe_wiappei_i[bcie_rx_vi_9_pipe_wiappei]	CH:30 /pcie_7x_v1_9_/gt_top_/pipe_wrapper_/pipe_lane[0].gt_wrapper_//GT_TXDATAK<22
pipe_lane[0].pipe_user_l [pcie_7x_v1_9_pipe_user]	CH:38 /ncie 7x v1 9 i/ot ton i/nine wranner i/nine lane[0].gt_wranner i/GT_TXDEFMDH
pipe_lane[0].gt_wrapper_i [pcie_7x_v1_9_gt_wrapper]	CH:39 /pcie_7x_v1_9_i/of top i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_TXELEINPIT
🗠 pipe_lane[0].pipe_drp.pipe_drp_i [pcie_7x_v1_9_pipe_d	CH:40 /pcie 7x v1 9 i/gt top i/pipe wrapper i/pipe lane[0].gt wrapper i/GT TXPHALIGN
🕶 pipe_lane[0].pipe_eq.pipe_eq_i [pcie_7x_v1_9_pipe_eq	CH:41 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_TXPHALIGNDONE
🗠 pipe Jane[0].pipe guad.pipe common.gpll drp i [pcie	CH:42 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_TXMARGIN<0>
nine Jane[0] nine guad nine common goll wrapper i l	CH:43 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_TXMARGIN<1>
pipe_tane[0].pipe_quad.pipe_common.qpii_wappet_t	CH:44 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_TXMARGIN<2>
pipe_tane[0].pipe_tate.pipe_tate_t[pcte_tx_v1_9_pipe_	CH:45 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_TXRATEDONE
pipe_lane[0].pipe_sync_i [pcie_7x_v1_9_pipe_sync]	CH:46 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_TXRESETDONE
🔶 pipe reset.pipe reset i [pcie 7x v1 9 pipe reset]	CH:47 /pcie_7x_v1_9_i/gt_top_i/pipe_wrapper_i/pipe_lane[0].gt_wrapper_i/GT_TXUSERRDY

Figure 21 - GT Transmit Channel Signals

Signal Set-7, User

Structure / Nets		Net Sele	ections		
<pre>P- / [xilinx_pcie_2_1_ep_7x]</pre>	_	Clock	Signals	Trigger Signals	Data Signals
🕶 app [pcie_app_7x]		Channe	a		
 ext_clk.pipe_clock_i [pcie_7x_v1_9_pipe_clock] 		CH:49	/user li	nk up	
pcie_7x_v1_9_i [pcie_7x_v1_9]		CH:50	/user_r	eset	
pcie top i [pcie 7x v1 9 pcie top]		CH:51	/sys_rs	t_n_c	

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.

Structure / Nets Net Selections

 app [pcie_app_7X] ext_clk.pipe_clock_i [pcie_7x_v1_9_pipe_clock] channel chino [pcie_7x_v1_9_i/pl_phy_Ink_up_q chino [pcie_7x_v1_9_i/pl_phy_Ink_up_q chino [pcie_7x_v1_9_i/pl_phy_Ink_up_q
ext_clk.pipe_clock_i [pcie_7x_v1_9_pipe_clock] P pcie_7x_v1_9_i [pcie_7x_v1_9] CH:0 [pcie_7x_v1_9_i/pl_phy_Ink_up_q CH:1 [pcie_7x_v1_9_i/pl_received_hot_rst_q
P cie_7x_v1_9_i [pcie_7x_v1_9] CH:1 /pcie_7x_v1_9_i/pl_received_hot_rst_q =
pcie_top_i [pcie_7x_v1_9_pcie_top] CH:2 /pcie_7x_v1_9_i/pl_link_partner_gen2_supported
pcie_pipe_pipeline_i [pcie_7x_v1_9_pcie_pipe_pipeline] CH:3 /pcie_7x_v1_9_jpi_link_geng_cap
• pcie_7x_i [pcie_7x_v1_9_pcie_7x] = $cn_* + pcie_7x_v1_9_pcie_7x_v1_9$
Protection of the second
rx_inst[pcie_7x_v1_9_axi_basic_rx] CH:7 /pcie_7x_v1_9_i/pl_initial_link_width<1>
• tx_inst[pcie_7x_v1_9_axi_basic_tx] CH:8 /pcie_7x_v1_9_i/pi_initial_link_width<0>
• gt_top_i[pcie_7x_v1_9_gt_top]
pipe wrapper i [pcie 7x v1 9 pipe wrapper] CH:12 /pcie_7x_v1_9_i/pl_sel_Ink_width<0>
pipe lane[0].pipe user i [pcie 7x v1 9 pipe user] CH:13 /pcie_7x_v1_9_i/pl_tx_pm_state<2>
CH:14 /pcie 7x v1 9 i/pl tx pm_state<1>
• pipe lane(0)pipe dro pipe dro i focie 7x v1 9 pipe dro)
>> pipe lane(0)pipe eq.pipe eq i[pcie 7x v1 9 pipe eq] ✓ CH:17 /pcie 7x v1 9.j/pl_x.pm_state<0>

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. • 000: Link not trained • 001: 1-Lane link • 011: 2-Lane link • 100: 8-Lane link
pl_phy_lnk_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. 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) 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. 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. 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 La	yer 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.

📧 Wave - Default 🖂 🚽 👘			
\$ 1 -	Msgs		
Get Signal Set-1, LTSSM		(Signal Set-1, LTSSM)	
/EP/pcie_7x_v1_9_i/pl_ltssm_state	1f	00 X02 104 X05 XX0d XX0f XX13X X16 X1c11d X1f	<u> </u>

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.

🔳 Wave - Default 💳							
61 -	Msgs						
■		(Signal Set-'	I, LTSŚM)			
💼 🖕/EP/pcie_7x_v1_9_i/pl_ltssm_state	1f	02			104		
Signal Set-2, Detect State		(Signal Set-2	2, Detect S	Statė)			
	St0						
	St0						
	St0						
9_i/pcie_top_i/pipe_rx2_phy_status	St0						
	St0						
•	000	000			X000		
•	000	000			X000		
•	000	000			1000)	
🔹 🖕 🕂v1_9_i/pcie_top_i/pipe_rx3_status	000	000			1000)	
	St1						
	St1						
1_9_i/pcie_top_i/pipe_tx2_elec_idle	St1						
	St1						

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

😠 Wave - Default 💳		
🖕 🗸	Msgs	5
■→ Signal Set-1, LTSSM		(Signal Set-1, LTSSM)
/EP/pcie_7x_v1_9_i/pl_ltssm_state	1f	02
■→ Signal Set-2, Detect State		(Signal Set-2, Detect State')
-4pe_wrapper_i/PIPE_TXDETECTRX	St0	
	St0	
9_i/pcie_top_i/pipe_rx1_phy_status	St0	
9_i/pcie_top_i/pipe_rx2_phy_status	St0	
	St0	
v1_9_i/pcie_top_i/pipe_rx0_status	000	000 1011 1000
v1_9_i/pcie_top_i/pipe_rx1_status	000	000 X011 X000
v1_9_i/pcie_top_i/pipe_rx2_status	000	000 1011 1000
v1_9_i/pcie_top_i/pipe_rx3_status	000	000 1011 1000
1_9_i/pcie_top_i/pipe_tx0_elec_idle	St1	
1_9_//pcie_top_//pipe_tx1_elec_idle	Sti	
	SU	
	รถ	

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.

💼 Wave - Default 💳			
\$1 +	Msgs		
■ 🔶 Signal Set-1, LTSSM		(Signal Set-1, LTSSM)	
board/EP/pcie_7x_v1_9_i/pl_ltssm_state	16	00 102	X04 X05
		(Signal Set-2, Detect State)	
■ ♦ Signal Set-3, Reset		(Signal Set-3, Reset)	
	St0		
per_i/pipe_reset/pipe_reset_i/cpllreset	0		
lane[0]/gt_wrapper_I/GT_CPLLLOCK	ISt1		
t_top_//pipe_wrapper_//gt_txresetdone	1111		
•	 0+1		
	011 011		
ane[0]/gt_wrapper_i/GT_CTVRESET	1311 19t0		
	St0		
• 4	0000	0000	
→top i/pipe wrapper i/PIPE RESET N	St1		
■ I Signal Set-4, Clocking		(Signal Set-4, Clocking)	
/pcie_7x_v1_9_i/gt_top_i/clock_locked	St1		
	St1		
• • •i/gt_top_i/pipe_wrapper_i/gt_rxcdrlock	1111	1111	
e_7x_v1_9_i/gt_top_i/reg_clock_locked	1		
• 💠i/pipe_wrapper_i/PIPE_CPLL_LOCK	1111	1111	
i/pipe_wrapper_i/PIPE_PCLK_LOCK	St1		
lane[0]/gt_wrapper_I/GT_CPLLLOCK	ISt1		
	Sti		
➡ Signal Set 6 CT TV		(Signal Sets, GT KX)	
Signal Set-0, GT TA		(Signal Set-0, GTTA)	
/hoard/EP/ncie 7x v1 9 i/user lok un	St1		
/board/EP/pcie 7x v1 9 i/user reset int	0		
//board/EP/pcie_7x_v1_9_i/svs_rst_n	st1		

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 Itssm_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.

■		(Signal S	Set-1, LT	SSM)										
	16	04	105											
■		(Signal S	Set-5, GT	ˈÅX)										
• • • …pe_wrapper_i/PIPE_RXCHANISALIGNED	1111	1111												
is determined with the second seco	f5f6	4a4a	bc	licb	2)f7 f.	7,106ff (4)4a4	a		bclf	7f7.)06	6ff /4	/4a4a	
EP/pcie 7x v1∖9 i/pipe rx0 char is k gt (00	00	110	X11		100				10 (1 /00)		
# /board/EP/pcie_7x_v1_9_i/pipe_rx1_data_gt	f5f6	4a4a	bc	liclbo	2	7,106ff (4)4a4	а		bclf	7f7.)06	6ff /4	/4a4a	
EP/pcie_7x_v1_9_i/pipe_rx1_char_is_k_gt (00	00	110	X11		100				10 ľ	1)00)		
💿 👳 🔶 /board/EP/pcie 7x v1_9 i/pipe_rx2_data_gt	f5f6	4a4a	bc	liclbo	2	7,06ff (4)4a4	a		bclf	7f7 (06	iff (4	/4a4a	
EP/pcie 7x v1 9 i/pipe rx2 char is k gt (00	00	10	X11		100				10 ľ	1 /00)		
# /board/EP/pcie_7x_v1_9_i/pipe_rx3_data_gt	f5f6	4a4a	bc	liclbo	2)f7f.	7 (06ff (4)4a4	a		bclf	7f7)06	6ff./4	1/4a4a	
EP/pcie_7x_v1_9_i/pipe_rx3_char_is_k_gt (00	00	110	<u>X11</u>		100				10 (*	1 00)		
🛛 🙀 🔩op i/pipe wrapper i/PIPE RXELECIDLE 🛛	0000	0000												
• • • • …op_i/pipe_wrapper_i/PIPE_RXPOLARITY	0000	0000												
🔹 🤹 🚛 top_i/pipe_wrapper_i/PIPE_RXSTATUS 🛛 🤅	0000000	0000000	00000											
🛛 🙀 🔩ane[0]/gt_wrapper_i/GT_RXBUFSTATUS 🛛	000	000												
40]/gt wrapper i/GT RXBYTEISALIGNED	St1													
4 lane[0]/gt wrapper i/GT RXCDRLOCK	St1													
🚽 🗠]/gt wrapper i/GT RXCHANISALIGNED 🛛	St1													
🥠 "lanef0]/gt wrapper i/GT RXCDRRESET 🛛	St0													
🚽 📥 lane[0]/gt wrapper i/GT_RXCDRLOCK	St1													
🔹 🗄 👍0]/gt_wrapper_i/GT_RXCHARISCOMMA 🔤	0000	0000 X	o 100	<u>Xol Xoo</u>	00				10 X	0000				10
	St0													
- A lane[0]/gt wrapper i/GT RXELECIDLE	St0								Т					
🧈 🌛pe lane[0]/gt wrapper i/GT RXPHALIGN 🛛	St0													
-4[0]/gt wrapper i/GT RXPHALIGNDONE	St0													
	St1													

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

🖙 🔶 Signal Set-1, LTSSM	(ˈSignalˈSet-1, LTSSM)		
istate	04 105		
■ I Signal Set-6, GT TX	(Signal Set-6, GT TX)		
🥠 lane[0]/gt wrapper i/GT GTTXRESET St0			
💀 🧄 /board/EP/pcie 7x v1 9 i/pipe tx0 data gt 06d3	f7(fff7 10)4a4a	XF7 Xfff7 X0 X4545	(f7)(fff7 (0)(4545
EP/pcie 7x v1 9 i/pipe tx0 char is k gt 00	11 (01)(00	X11 X01 X00	(11)(01)(00
• / /beard/EP/pcie_7x_v1_9_i/pipe_tx1_data_gt_06d3	f7	XF7 Xfff7 X0 X4545	(f7)(fff7 (0)(4545
•••EP/pcie_7x_v1_9_i/pipe_tx1_char_is_k_gt 00	11 /01 /00	X <u>11 X01 X00</u>	<u>, 11 101 100 </u>
/board/EP/pcie_7x_v1_9_i/pipe_tx2_data_gt_06d3	f7,)(fff7)(0)(4a4a	XF7XFFF7 X0 X4545	, 17, 1117 10 14545
EP/pcie_7x_v1_9_i/pipe_tx2_char_is_k_gt 00	11 /01 /00	X <u>11 X01 X00</u>	(11)(01)(00
/board/EP/pcie_7x_v1_9_i/pipe_tx3_data_gt_06d3	f7,lfff7 l0 l4a4a	XF7Xfff7 X0 X4545), 17), 1117 (D), 14545
EP/pcie_7x_v1_9_i/pipe_tx3_char_is_k_gt 00	11 /01 /00	X11 X01 X00	(11)(01)(00
pe_lane[0]/gt_wrapper_i/GT_TXDEEMPH St0			
lane[0]/gt_wrapper_i/GT_TXELECIDLE St0			
pe_lane[0]/gt_wrapper_i/GT_TXPHALIGN St0			
•ipe_lane[0]/gt_wrapper_i/GT_TXMARGIN 000	000		
•	000		

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



TS ordered set with lane numbers in each lane instead of the PAD value. If the link trains down (for example, from x8 to x4), rx_data and tx_data at the GT interface should be captured in ChipScope Pro/Vivado ILA to figure out how the change in the link and lane number field in the ordered sets is occurring. If the endpoint device is sending lane numbers on all 8 lanes, but link partner is replying with lane numbers only on the first four lanes and the rest still with PAD value, it would potentially indicate some signal integrity issue on the link. The value that endpoint sent in the link on those lanes are probably not understood by the link partner due to the signal integrity issue.

- B: Configuration Linkwidth, State 0
- C: Configuration Linkwidth, State 1
- D: Configuration Linkwidth, Accept 0
- ${\ensuremath{\mathbb E}}$: Configuration Linkwidth, Accept 1
- F: Configuration Lanenum Wait
- 10: Configuration Lanenum, Accept
- 11: Configuration Complete x1
- 12: Configuration Complete x2
- 13: Configuration Complete x4
- 14: Configuration Complete x8
- 15: Configuration Idle

Figure 31 - Sub-states in LTSSM Configuration state

Figure 33 shows the root complex sending TS1s with link number assigned to 00. The endpoint agrees with this and starts transmitting TS1s on tx_data with link number 00 in the link number field. This is shown in Figure 34. After the link number has been negotiated, the root complex then starts to send TS1 with lane numbers on lane number field of TS1 as shown in Figure 35. In response to transmission of TS1 with lane numbers from the link partner, the endpoint starts sending the same corresponding lane numbers on each lane, thus agreeing with the lane numbers to communicate with. This is shown in Figure 36.

In CONFIGURATION, the N_FTS value is agreed. In the captures shown, the endpoint is sending FF in the N_FTS field in TS1, indicating that the endpoint requires 255 FTS when exiting from L0s to L0 to achieve bit and symbol lock. On the other hand, the RP also sends FF in its N_FTS field in TS1, indicating that it also requires o255 FTS to be transmitted by the endpoint when exiting from L0s to L0.

■		(Signal Set-1, LTSSM)				
i doard/EP/pcie_7x_v1_9_i/pl_Itssm_state	0b	05 ľ0b				
🖃 🔶 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	f7f7 (06ff)(4500)(4545	bc45 (f7f7 (06)	f 4500	4545	/bc45
🔹 🤣pcie 7x v1 9 i/plse rx0 char is k gt	00	11 00	10 X11 X00			10
🔹 💠d/EP/pcie 7x v1 9 i/pipe rx1 data gt	4545	f7f7 (06ff)4500)4545	lbc45 lf7f7 l061	f 4500	4545	/bc45
🔹 💠pcie 7x v1 9 i/pipe rx1 char is k qt	00	11 00	X10 X11 X00			110
💁 💠d/EP/pcie 7x v1 9 i/pipe rx2 data gt	4545	f7f7 (06ff)4500)4545	(bc45 (f7f7 (06)	f 4500	4545	lbc45
💁 💠pcie 7x v1 9 i/pipe rx2 char is k gt	00	11 00	<u>10 (11)00</u>			10
•d/EP/pcie 7x v1 9 i/pipe rx3 data gt	4545	f7f7 106ff 14500 14545	1061 lbc45 lf7f7 l061	f 4500	4545	/bc45
🗤 🚽pcie 7x v1 9 i/pipe rx3 char is k at	00	11 00	10 111 100			110
NX Other Signals		(RX Other Signals)				
■♦ Signal Set-6. GT TX		(Signal Set-6, GT TX)				
		(Signal Set-6, GT TX)				
= 🔶 TX DATA						
• d/FP/pcie 7x v1 9 i/pipe tx0 data ot	4545	4545 IF7bc Ifff7 10006 14545		lf7hc) Ifff7 10006 14a4a	
$\mathbf{r} \rightarrow \mathbf{r}$	00			<u>111</u>	101 100	
$\mathbf{p} \rightarrow \mathbf{q} = \mathbf{q} + \mathbf{p} + \mathbf{q} = \mathbf{q} + \mathbf{q} + \mathbf{q} = \mathbf{q} + \mathbf{q} + \mathbf{q} = \mathbf{q} + $	4545	4545 YF7bc Yfff7 Y0006 Y4545		lf7hc	Yfff7 Y0006 Y4a4a	
\Rightarrow ncie 7x v1 9 i/nine tv1 char is k at	00			Y11		
d/FP/ncie 7x v1.9 i/nine tv2 data at	4545	4545 YEZHO YEET YOOOG YA545		lf7bc	Yfff7 Y0006 Y1343	
$\mathbf{r}_{\mathbf{r}}$ ncie 7x v1 9 i/nine tv2 char is k at	00			Y11		
d/EP/ncie 7x y(1.9 i/nine ty3 data at	A 5A 5	4545 YEZHO YEET YOOOG YA545		lif7bc	Ytter 10006 14-24-2	
$\sim \sim $	00			Y1 1	You You	
TX Other Signals	00	(TX Other Signals)				

Figure 32 – Both RP and EP sending PAD (F7) in link number and lane number fields

Signal Set-1, LTSSM		(Signal S	Set-1, LTSSM)					
is the second	0c	0b		l(l)c					
🖃 🔶 Signal Set-5, GT RX		(Signal S	Set-5, GŤ RX)						
pe_wrapper_i/PIPE_RXCHANISALIGNED	1111	1111							
■ 🔶 RX DATA		(RX DAT	A)						
d/EP/p≱ie_7x_v1_9_i/pipe_rx0_data_gt	f700	4a4a	ĺbc4a	lf700	,06ff	/4a00	/4a4a		
👳 🗇pcie_7x_v1_9_i/pipe_rx0_char_is_k_gt	10	00	(10		100				
•	f700	4a4a	lbc4a	lf700	,06ff	/4a00	/4a4a		
•	10	00	(10		100				
🔹 💠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		100				
•d/EP/pcie_7x_v1_9_i/pipe_rx3_data_gt	f700	4a4a	lbc4a	(f700	,06ff	/4a00	/4a4a		
🔹 🗇pcie_7x_v1_9_i/pipe_rx3_char_is_k_gt	10	00	(10		200				
RX Other Signals		(RX Othe	er Signals)						
■ 🔶 Signal Set-6, GT TX		(Signal S	Set-6, GŤ ŤX)						
		i Signal S	et-6, GT TX í						
A TX DATA		Î TX DAT	Ά) Ι						
d/EP/pcie 7x v1 9 i/pipe tx0 data at	4a4a	4a4a				lf7bc	lfff7	10006	14a4a
■ ↓ …pcie 7x v1 9 i/pipe tx0 char is k gt	00	00				X11	X01	100	
• /d/EP/pcie 7x v1 9 i/pipe tx1 data at	4a4a	4a4a				lf7bc	Xfff7	10006	14a4a
•pcie 7x v1 9 i/pipe tx1 char is k at	00	00				X11	101	100	
•d/EP/pcie 7x v1 9 i/pipe tx2 data at	4a4a	4a4a				lf7bc	lfff7	10006	14a4a
• 🛧	00	00				Ĭ11	101	100	
•d/EP/pcie 7x v1 9 i/pipe tx3 data ot	4a4a	4a4a				lf7bc	Xfff7	10006	14a4a
• • • • • • • • • • • • • • • • • • •	00	ho				¥11	101	100	
TX Other Signals		(TX Othe	er Signals)						
			n enginaio)						

■		/ Signal Set-1	LTSSM)									هسه	
/board/EP/pcie 7x v1 9 i/pl Itssm state	0e	0e	<u>, ,</u>										
■ ♦ Signal Set-5, GT RX		(Signal Set-5	, GT ['] RX)										
	1111	1111										كالت	
🖕 🔶 RX DATA' 👘 👘		(RX DATA)											
🔹 🌪d/EP/pcie_7x_v1_9_i/pipe_rx0_data_gt	bc4a	4a4a	lb¢4a	10000	Ï¢6ff	/4a00)4a4a				lbd4a	0000	lo.
pcie_7x_v1_9_i/pipe_rx0_char_is_k_gt	10	00	(1φ	100							10	00)	
d/EP/pcie_7x_v1_9_i/pipe_rx1_data_gt	bc4a	4a4a	lb¢4a	10100	Ϊ¢6ff	/4a00	/4a4a				lbc4a	0100	10.
pcie_7x_v1_9_i/pipe_rx1_char_is_k_gt	10	00	(1φ	100							110	00	
d/EP/pcie_7x_v1_9_i/pipe_rx2_data_gt	bc4a	4a4a	lb¢4a	10200	X¢6ff	/4a00	/4a4a				lbo <mark>4</mark> a	0200	10.
pcie_7x_v1_9_i/pipe_rx2_char_is_k_gt	10	00	(1φ	100							110	00	
d/EP/pcie_7x_v1_9_i/pipe_rx3_data_gt	bc4a	4a4a	lb¢4a	10300	Ĭ¢6ff	/4a00	/4a4a				lbc <mark>4</mark> a	0300	ĭo
📃 🗄 🔶pcie_7x_v1_9_i/pipe_rx3_char_is_k_gt	10	00	<u> (1φ</u>	100							110	00	
🗄 🔶 RX Other Signals		(RX Other S	ignals)										
🖃 🔶 Signal Set-6, GT TX		(Signal Set-6	i, GT TX)										
		(Signal Set-6	i, GT TX)										
🖕 🔶 TX DATA		(TX DATA)											
d/EP/pcie_7x_v1_9_i/pipe_tx0_data_gt	4a4a	fff7 \0006	/4a4a					100bc),fff7	10006	14a4a	کی	
pcie_7x_v1_9_i/pipe_tx0_char_is_k_gt	00	01 ХФО						101		χοφ		<u>ع مع م</u>	
d/EP/pcie_7x_v1_9_i/pipe_tx1_data_gt	4a4a	fff7 \0006	/4a4a					100bc),fff7	10006	14a4a	کی	
pcie_7x_v1_9_i/pipe_tx1_char_is_k_gt	00	01 ХФО						101		χοφ		<u>ع مع م</u>	
d/EP/pcie_7x_v1_9_i/pipe_tx2_data_gt	4a4a	fff7 X0006	/4a4a					100þc	_\fff7	χοφοε	14a4 <mark>a</mark>		
pcie_7x_v1_9_i/pipe_tx2_char_is_k_gt	00	01 ХФО						101		χοφ		<u>ع مع م</u>	
i.d/EP/pcie_7x_v1_9_i/pipe_tx3_data_gt	4a4a	fff7 \0006	14a4a					100bc	_Xfff7	10006	14a4a	كصع	
pcie_7x_v1_9_i/pipe_tx3_char_is_k_gt	00	01 (00						101		100		کی	
🗄 🔶 TX Other Signals		(TX Other Si	gnals)										

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 Itssm state	10	0f (10											
📼 🔶 Signal Set-5, ĠT 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	lbc45	10000	46ff	4500	4545				(bc45	10000	
pcie 7x v1 9 i/pipe rx0 char is k gt	00	00	10	100							10	100	
•d/EP/pcie 7x v1 9 i/pipe rx1 data at	4500	4545	lbc45	10100	/46ff	4500	4545				lbo45	10100	
💀	00	00	110	100							10	100	
• 4d/EP/pcie 7x v1 9 i/pipe rx2 data at	4500	4545	lbc45	10200	(46ff	4500	4545				lbc45	10200	
•••	00	00	¥10	100							10	100	
• d/EP/pcie 7x v1 9 i/pipe rx3 data gt	4500	4545	lhc45	10300	146ff	14.500	¥4.54.5				1hd4.5	10300	
ndie 7x v1 9 i/nipe rx3 char is k at	00	00	110	100	in em						10	100	
• A BX Other Signals		(BX Other S	Signals 1									100	
Signal Set-6 GT TX		(Signal Set-	6 GT TX 1										
		(Signal Set-	6 GT TX I										
		(TX DATA)											
d/EP/ncie 7x v1 9 i/nine tx0 data at	4a4a	FFOO TOOOS	4949					lonbo	YffOO	10006	4242		
\sim ncie 7 v v1 9 i/nine tv0 char is k at	00	h00 10000	лана					101	100		- muru		
d/EP/ncie 7x y1 0 i/nine tv1 data at	/9/9	FF0.1 10006	14-24-2					100bc	Yff0 1	YAAAA	1/1-/1-2		
$a \rightarrow a$ nois 7 v1 9 i/nine tv1 char is k at	00		/HaHa								Hata		
d(ER)poie_7X_v1_9_/pipe_x1_cital_is_x_gt	4040	600 V0000	Y4 - 4 -					10	100	VADOR	10-0-		
- urer/pole_/x_vi_9_r/pipe_ux2_uala_yl	4d4d	1102 10006	/4848							10000	4343		
			Va. a							10000	- Va - A		
	4a4a	103 10006	/4a4a						103	10006	/4a4a		
•••pcie_/x_v1_9_i/pipe_tx3_char_is_k_gt	00	00											
		(TX Other S	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 Itssm state	13	10 113										
🖘 🔶 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	10000	/46ff	4500	4545			1bc45	(0000	
•pcie_7x_v1_9_i/pipe_rx0_char_is_k_gt	00	00	10)00						1 X10)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						1 X10)00	
•d/EP/pcie_7x_v1_9_i/pipe_rx2_data_gt	4545	4545	lbc45	0200	/46ff	4500	4545			(bc45	0200	
•pcie_7x_v1_9_i/pipe_rx2_char_is_k_gt	00	00	Ľ10	100						10)00	
•d/EP/pcie_7x_v1_9_i/pipe_rx3_data_gt	4545	4545	lbc45	0300	/46ff	4500	4545			(bc45	0300	
pcie_7x_v1_9_i/pipe_rx3_char_is_k_gt	00	00		100						10 X10	<u> 100</u>	
🖦 🔶 RX Other Signals		(RX Other Signals)										
🖃 🔶 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	X4545			
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	ĴffO 1	10046	4545			
pcie_7x_v1_9_i/pipe_tx1_char_is_k_gt	00	00				101	100			ļ		
d/EP/pcie_7x_v1_9_i/pipe_tx2_data_gt	4545	ff02,0006 ,4a4a				/00bc	lff02	10046	4545	ļ		
pcie_7x_v1_9_i/pipe_tx2_char_is_k_gt	00	00				101	100			ļ		
id/EP/pcie_7x_v1_9_i/pipe_tx3_data_gt	4545	ff03/0006 /4a4a				/00bc	lff03	10046	 4545	ļ		
pcie_7x_v1_9_i/pipe_tx3_char_is_k_gt	00	00				101	100					
i → 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_lnk_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.

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

Table 4 – Pipe Wrapper Parameters

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_PHYSTATYUS
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'b1000000000_or PIPE_RATE_FSM_stuck at 24'b0001000000000000000000000000000000000
PIPE_SYNC_FSM_RX	Not supported
PIPE_DRP_FSM	Add to Chipscope if PIPE_RATE_FSM stuck at 24'b000000000000000000000000000000000000
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 ~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 ~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 posttap. 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.


Port	Dir	Clock Domain		Description
TXDIFFCTRL[3:0]	In	TXUSRCLK2	Driver Swing Contrare in mV_{PPD} .	rol. The default is user specified. All listed values
			[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
			Note: These are pre voltage is defined w TXPRECURSOR =	eliminary values. The peak-to-peak differential when TXPOSTCURSOR = 5 'b00000 and 5 'b00000.

Table 7 - TXDIFFCTRL



Port	Dir	Clock Domain		Des	cription	
TXPOSTCURSOR[4:0]	In	Async	Transmitter pos specified. All lis	t-cursor TX pre-e sted values (dB) a	mphasis control. The default is are typical.	user
			[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	
			Note: These are are defined whe	preliminary val en the TXPRECU	ues. The TXPOSTCURSOR valu RSOR =5 ' b00000	ies

Table 8 - TXPOSTCURSOR

Emphasis = $20\log_{10}(V_{high}/V_{low}) = |20\log_{10}(V_{high})|$



Port	Dir	Clock Domain		Descri	ption	
TXPRECURSOR[4:0]	In	Async	Transmitter pre-c specified. All liste	ursor TX pre-emp ed values (dB) are	phasis control. The de typical.	efault is user
			[4:0]	Emphasis (dB)	ICoefficient UnitsI	
			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	
		ļ	5'b10101	6.02	20	ļ
			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	
			Note: These are p defined when the Emphasis = 20log	reliminary values TXPOSTCURSO z10(V _{high} /V _{low}) =	s. The TXPRECURSC R =5 ' b00000 : 20log10 (Vlow/Vl)R values are high)

Table 9 - TXPRECURSOR

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 rxelecidle 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.

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

Skew [ns] = UI * number_of_characters * 10 = 0.4 ns * 7 * 10 = 28 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 deassertion 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.
 - o 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.
 - o 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 <u>DS182</u>.

Standard	Descript	tion	Line Rate (Mb/s)	Min	Max	Units
PCI Express Transmitter Jit	ter Generation					
PCI Express Gen 1	Total transmitter jitter		2500	-	0.25	UI
PCI Express Gen 2	Total transmitter jitter		5000	-	0.25	UI
	Total transmitter jitter uncorrelated		-	31.25	ps	
PCI Express Gen 3(2)	Deterministic transmitter j	itter uncorrelated	8000	-	12	ps
PCI Express Receiver High	Frequency Jitter Toleran	ice				
PCI Express Gen 1	Total receiver jitter toleran	се	2500	0.65	-	UI
	Receiver inherent timing e	error	5000	0.40	-	UI
PCI Express Gen 20	Receiver inherent determi	inistic timing error	5000	- 31.25 - 12 0.65 - 0.40 - 0.30 - 1.00 - Note 4 -	-	UI
		0.03 MHz-1.0 MHz		1.00	-	UI
PCI Express Gen 3 ⁽²⁾	Receiver sinusoidal jitter	1.0 MHz-10 MHz	8000	Note 4	-	UI
		10 MHz-100 MHz		0.10	-	UI

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

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.



GTX Transceive	er en				
V (8)	Analog supply voltage for the GTX transceiver QPLL frequency range \leq 10.3125 GHz ⁽⁹⁾⁽¹⁰⁾	PLL frequency range0.971.0PLL frequency range1.021.05PLL frequency range1.021.05d receiver termination1.171.2sceivers1.751.80	1.08	v	
VMGTAVCC ⁽⁰⁾	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

Table 11 – GTX Transceiver Voltage Requirements (DS182)

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

9. For data rates \leq 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
 - o 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.







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





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.



View by Function View by Name	Version	AXI4 AXI4-Stream	AXI4-Lite	Status	License	Vendor	Library	
+ Z Automotive & Industrial	version.	Poer Poer Steam	Pour cite	Status	cicense	*endor	cibitary	
AXI Infrastructure								
BaseIP								
Basic Elements								
E Communication & Networking								
Debug & Verification								
🚽 🐳 AXI Bus Functional Model	2.1			Pre-Production		xilinx.com	ip	
🖻 📂 Debug								
- 🏹 ATC2 (ChipScope Pro - Agilent Trace Core 2)	1.05.a			Production		xilinx.com	ip	
– 🍕 AXI Chipscope Monitor	3.05.a			Pre-Production		xilinx.com	ip	
– 🦞 IBERT 7 Series GTH (ChipScope Pro - IBERT)	2.00.a					xilinx.com	ip	
– 🦞 IBERT 7 Series GTH (ChipScope Pro - IBERT)	2.01.a					xilinx.com	ip	
— 🍕 IBERT 7 Series GTP (ChipScope Pro - IBERT)	2.00.a					xilinx.com	ip	
– 💐 IBERT 7 Series GTX (ChipScope Pro - IBERT)	2.02.a			Production		xilinx.com	ip	
– 🍕 IBERT Spartan6 GTP (ChipScope Pro - IBERT)	2.02.a					xilinx.com	ip	
– 🌾 IBERT Virtex5 GTX (ChipScope Pro - IBERT)	2.01.a					xilinx.com	ip	
— 🍕 IBERT Virtex6 GTH (ChipScope Pro - IBERT)	2.03.a					xilinx.com	ip	
– 🍕 IBERT Virtex6 GTX (ChipScope Pro - IBERT)	2.06.a					xilinx.com	ip	
- 🍕 ICON (ChipScope Pro - Integrated Controller)	1.06.a			Production		xilinx.com	ip	
– 🍕 ILA (ChipScope Pro - Integrated Logic Analyzer)	1.05.a			Production		xilinx.com	ip	
- VIO (ChipScope Pro - Virtual Input/Output)	1.05.a			Production		xilinx.com	ip	
Digital Signal Processing								
Embedded Processing								
PGA Features and Design								
Search IP Catalog:								Clear
						Contu ID	en non atila la cuitta c	-

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.

Symbol	8×				
	_	IB	ERT 7 Sei	nies GTX	
IBERT_SYSCLOCK		19- · ·	cuibacob	CIIO	
Q115_REFCLK0_IN			IBER'	F)	villey comula chinacona thart Zeoriae, atu 2.02 a
Q115_REFCLK1_IN				.,	xiiiix.com.ip.cnipscope_berc_/series_gox.z.oz.z
MGT0_115_RX_IN	→MGT0_115_TX_OUT	Component Name	ibort?		
	→M0T0_115_RECOLIC_OUT	component name	RPCI CZ.		
MG11_115_KX_IN	→MG11_115_1X_001	Board Configuration S	etting User Defined		
MOT2 115 RV IN-	MOTO 115 TV OUT				
mo12_113_10/_14	MOTZ 115 RECCLK OUT	Converte Ditates an			
MGT3_115_RX_IN-+	→MOT3 115 TX OUT	Generate bitstream			
2.272		When using ISE, en	able 'Generate Bitstr	eam using ISE tools	' checkbox
Q116_REFCLK0_IN		Generate Bits	tream using ISE Too	ols	
Q116_REFCLK1_IN		When using Virado	cource the generate	d v rdi implement (in l
MGT0_116_RX_IN	→MGT0_116_TX_OUT	when using widou,	source the generate	a v_rul_implement.	
		System Design			
MOT1_116_RX_IN	→MGT1_116_TX_OUT		waha		
MOT2 116 RV IN	MOTT 116 TX OUT	Add KXOUTCLK	robe		
mo12_110_10_04	MOT2 116 RECOVE OUT	GTX Naming Style	4GTm n ≠ ex. MGT	0_113 / MGTREFCLK	0_113
MGT3_116_RX_IN-+	→MGT3_116_TX_OUT				
and the second second second		System Clock			
Q117_REFCLK0_IN		V Use External cloc	k source		
Q117_REFCLK1_IN		Enable Diff Term			
MGT0_117_RX_IN	→MGT0_117_TX_OUT	From once	200	MUS	
NOT 417 DV 81	→ MGT0_117_RECCLK_OUT	nequency	200	IMITIZ.	
MOTI_TIT_RX_IN		P Pin Location	C25		
MGT2 117 RV IN-	MOT2 117 TX OUT	N Pin Location	B25		
mon2_111_10/_14	-+ MOT2 117 RECCLK OUT	Pin Input Standard	DIFF SSTL15	•	
MOT3_117_RX_IN	→ M0T3_117_TX_0UT		Version		
		Silicon Version			
Q118_REFCLK8_IN		Silicon Version	General ES		•
Q118_REFCLK1_IN					
MGT0_118_RX_IN	→MGT0_118_TX_OUT				
MOTH HIS DV IN	MOTO_TIB_HECCLIC_OUT				
MOTI_II0_RX_IN	-+MOTI_118_RECOVER OUT				
MGT2 118 RX IN-	→MGT2 118 TX OUT				
MGT3_118_RX_IN-+	→MOT3_118_TX_OUT				

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.

cuments View							
iymbol	e ×		IBERT 7	Series GTX			
		PF	(Ching	Denes Gix			
IDERT SYSCLOCK		Logic	(Cnips	cope Pro -			
OT 5_REFCURD_IN			TP	SERT)			
Q1'5_REFOLK1_IN-+					xiiinx.com:ip:cr	apscope_ident_/serie	s_gtx:2.0.
MOTE_TIS_ROUN		GT clocking	mode selection				
M071_116_F0_01		🕐 Depende	nt TX/RX User Clock	king (Quad Based Protocol	Selection)		
MGT2_115_RX_IN	-+MGT2_15_DC_0UT	Independence	ient TX/RX User Clo	cking (GT based protocol :	election - Upto 4 Qu		
			No. of Quads	4	-		
MOTI_116_RU_N	+W0T3_16_TX_0UT +W0T3_16_RECC_K_0UT		Select Quad	OUAD 115	•		
G116_REFCLAD_IN-+			Select Quad	QUAD 116	*		
G1'8_REFCLM_IN-A			Select Quad	OUAD 117			
MGT0_116_RX_IN			Select Quad	OUAD 118	•		
MGT1_116_RX_IN		Number of Pr	otocok 1				
M0T2_116_RX_IN	-+MGT2_16_TK_OUT	Line rate se	ttings				
		Denter de	al day Data (Cha	whothe wanted Battle Battle	CT count Qued B1		
M0T1_110_R0UN-+		PTODA	u nacionali (oco	Spara visual Reiza (na.s)	Gi count quanter	6	
		Name Proto	col • 5.000	32 • 100.000 •			
OILT_REPOLKO_IN-		Custom_1					
WATE HIT OV IN	NOTO HIT TX OUT						
1.000 (0.000))))))))))	AWGTO -17 DECC & OUT	GTXs Resou	rces				
MOTI_117_FO_F4		GTXs of BUFG (count 4 count 8				
MOTO SIT DV IN	HIGT2 112 TV OUT						
and the second sec	WOT2 17 RECC & OUT						
MATTA 117 RX IN-							
and the second second second							
OT 0_REFCURE_IN							
Q1' E_REFCLICE_IN							
MGTE_119_RX_IN	+ MOTO_118_TX_OUT						
MOT1_110_RV_JN	+ MOT1_112_TX_OUT						
Same and Burnet							
WILL'LIN' KOUN-							
MOTO THE RY MI-	ANGTE TE TX OUT						
	-+WOT3_18_RECC_K_OUT						

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

Xilinx Answer 56616 – 7-Series PCIe Link Training Debug Guide



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

ymbol	Ø×	iller 1	IBERT 7 Series GTX	
IBERT_SVSC_OCK		logic Net	(ChipScope Pro - IBERT)	xilinx.com:p:chipscope_ibert_7series_gtx:2.02.
MOTE_THE ROUNT		tool of the la		
March 1993 March 1997	+ #070_16_RECC_M_0UT	Assign GTAS 10	a protocoi	
MOT1_115_ED_01		GTX	Protocol Selected	
1072_115_RC_N		MGID 115	None	-
state and street		MG11 11)	2 Nooe	
MST7_115_SS_01		MG12_115	Custom 1 / 5.000 Gbps	
Q1'6_REFCUID_N		MG13_115	Injoue	•
Q1 8_REPOLK1_N-+	and the second second second	MGT0_116	None	•
MSTD_118_RX_IN	T00_XT_812_0T0 M	MGT1_116	None	<u> </u>
	-AMOTO_14_RECC_JCOUT	MGT2_116	None	*
MGT1_T10_R5C(N)A	- AMOTI_ 16_TX_OUT	M513_116		
MOTO AND DV DALLAND	ANOTA TO THE OUT	MG10_11/	None	
	WOTZ "18 RECC.N. OUT	MGT1_117	Note	
MOTT_THERE N		MGT3_117	None	
	TUO_K.DOBR_01_ITOM+	MCT0 119	Non	-
OT T_REFCLED_RE		MGT1 118	None	1
OT 7_REFCULT_R	and the second second second second	MGT2 118	None	
WALF 111 TO DI	WOTO ST RECC & OUT	MGT3_118	None	•
MOTI_117_RC.R	-+HOT1_17_TX_OUT		Custom_1	
		Totals (CTs):	0.614	
MGT2_117_R0_01		iotais (o is).	0014	
and the second se	WGT2_17_RECO.X_OUT			
MOTO_HIT_ROUN-	+W0T2_117_TIC_OUT			
ON DESCRIPTION				
MOTO HIS DV PL	ANOTO HA TH OUT			
	HOTO VE RECORD OUT			
MOTCHE RUN-	TUO_NT_ 18 TK_OUT			
And the second second				
MOT2_119_R1(_N	++ #GT2_18_TX_OUT			
	TUO_ALDOBR_BIT_TON +			
MOTO_HIL_ROCIN	-+WGT2_11E_TX_OUT			

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.

BERT 7 Series GTX (ChipSco	ope Pro - IBERT)					- 0 🛋
P Symbol	e x					
r Symbol	0.4	IBER	T 7 Series GTX			
		Indi	inScone Pro -			
IBERT_SYSCLOCK						
0116_REFCLk0_IN			IBERT)	xilinx.com:ip:cf	ipscope ibert 7se	ries abx:2.02.
MOTO 115 EX IN-	WOTO 115 TX OUT		-			_
	MGT0_115_RECCLK_OUT	IBERT Design Summary				
MGT1_116_F0X_IN	→MGT1_116_TX_OUT	Component Name :	ibert2			
	+MGT1_115_RECCLX_OUT	Number of Protocols :	1			
MG12_115_RX_IN	→ MGT2_115_TX_0UT	System Clock Source :	External (P Pin: C25)			
MGT3_115_RX_IN	-+ MOT3_115_TX_OUT	System Clock Frequency :	200 MHz			
	→MGT3_115_RECCLK_OUT	BUEC south :	0			
Q116_REFCLK0_IN		BOFG count :	9			
MOTO THE EV IN-	WOTH HIS TV OUT	GTX count :	4			
300.020.020.000		MMCM count :	1			
MGT1_116_RX_IN	→ MGT1_116_TX_OUT	Refclk sources :	0			
		Board Configuration Setting	as : User Defined			
MOT2_118_RX_IN	→ MGT2_116_TX_OUT					
MGT3 116 RX (N	→ MGT3 116 TX OUT					
0.000	→MGT3_118_RECCLK_OUT					
Q117_REFCLK0_IN+						
Q117_REFCLK1_IN+						
MOTO_117_KX_04	→MGT8_117_RECCLK_DUT					
MGT1_117_F0(_IN	→MGT1_117_TX_OUT					
	-+MGT1_117_RECCLK_OUT					
M0T2_117_RX_IN	→MGT2_117_TX_OUT					
MOTE 117 RX IN						
O118_REFCLK0_IN						
Q118_REFCLK1_IN	and the second second					
m010_118_RX_IN	->MG10_118_1X_0UT					
MGT1_118_RX_IN						
	-+MOT1_118_RECCLK_OUT					
MOTZ_118_RX_IN>	→MGT2_118_TX_OUT					
MOTE 118 222 IN	→MGT2_118_RECCLK_OUT					
1015_115_10_10_10						
2						
		(and the second				
IP Symbol (Core Del	tails	Datasheet	< Back P	age 4 of 4 Next >	Generate	el Help

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)

MGT/BERT Settings	DRP Se	ettings	Port Se	ettings	RXM	largin Analysis		
			GTX_)	(0Y0		GTX	_X0Y1	
ℽ 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 L	ink		5.0	Gbps	
- PLL Status			CPLL LC	OCKED		CPLL	LOCKED	
- Loopback Mode		Far-En	d PMA		-	None		•
- Channel Reset		Reset			R	eset		
- TX/RX Reset		TX F	Reset	RXR	eset	TX Reset	RX Re	eset
- TX Polarity Invert				1				
 TX Error Inject 			Inje	ct		Ir	nject	
- TX Diff Output Swing		850 m ¹	850 mV (1100)		-	750 mV (1010)		-
- TX Pre-Cursor		1.67 dB	1.67 dB (00111)		-	0.00 dB (00000)		-
- TX Post-Cursor		0.68 dB (00011)		-	0.00 dB (00000)		-	
- RX Polarity Invert								
- Termination Voltage		Progra	Programmable		-	Programmable		-
P BERT Settings								
- BERT Reset			Res	et		R	eset	
- TX Data Pattern		PRBS	31-bit		-	PRBS 7-bit		•
- RX Data Pattern		PRBS	31-bit		-	PRBS 7-bit		•
- RX Bit Error Ratio			5.0368	5-001		5.03	3E-014	
- RX Received Bit Cou	int		2.158	E013		1.98	37E013	
RX Bit Error Count			1.087	E013		0.00	00E000	
Clocking Settings								
- TXUSRCLK Freq (MI	Hz)		156.	27		15	i6.27	
- TXUSRCLK2 Freq (N	(Hz)		156.	27		15	i6.27	
- RXUSRCLK Freq (M	Hz)		156.	27		15	i6.27	
RXUSRCLK2 Freq (MHz)		156.	27		15	6.27	

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.
- 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.



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.



Figure 59 - RP in Polling.Active

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



Packet R← 2.5 528731 R← x4	PATN	PATN Symbols K28.5 ***.5 K28.5 D**.* K	28.5 D**.*	Idle 8.000 ns	Time Stamp 0019 . 493 011 978
Packet 2.5 528732 R← X4	PATN	PATN Symbols K28.5 ***.5 K28.5 ***.*	Idle 16.000 ns	Time St 0019 . 493 0	tamp 112 010 s
Packet 2.5 528733 x4	PATN	PATN Symbols K28.5 ***.5 K28.5 D**.*	Idle 0.000 ns	Time St 0019 . 493 0	amp 112 042 s
Packet R← 2.5 528734 x4	 Data 12 byt 	es 20.000 ns 0019.4	n <mark>e Stamp</mark> 93 012 058 s		
Packet R← 2.5 528735 x4	EIOS	COM EIOS Symbols K28.5 K28.3 K28.3 K28.3	Idle 0.000 ns	Time 9 0019 . 493	Stamp 012 090 s
Packet 2.5 528736 R← x4	EIOS	COM EIOS Symbols K28.5 K28.3 K28.3 K28.3	Idle 0.000 ns	Time 3 0019 . 493	Stamp 012 106 s

Figure 60 - EP exits Polling. Compliance

4. EP enters Polling. Active and starts transmitting TS1s, as shown in Figure 61.

Packet R← 2.5 561008 R← x4 TS1	COM Link Lan K28.5 PAD PA	N_FTS Training Control	Data Rate 2.5.CT/s .5.CT/s	TS1 Symbols	Time Delta	Time Stamp
Packet 2.5	COM Link La	😂 LTSSM Flow Graph - [Fusi	ion_IO_SandyBridg	ge_IServerMB]		
561009 x4 151	K28.5 PAD P	Direction		\checkmark	Show Number Of	fTransitions
Packet R← 2.5 561010 R← x4 TS1	COM Link La K28.5 PAD P	 Upstream (R<-) Downstream (R->) 		- Data da		
Packet 2.5	COM Link La	🔘 Both		Detect		
561011 X4	COM Link La	States Status Previous: Detect		Ų ₽	$\langle \rangle$	
561012 R→ 2.5 TS1	K28.5 PAD P	Current: Polling		Polling		Disabled
Packet R→ 2.5 561013 R→ 4 TS1	COM Link La			1	K	
Packet _ 2.5	COM Link La			Configuration	ه ا	Hot Reset
561014 R← x4 TS1	K28.5 PAD P				N I	

Figure 61 - EP in Polling.Active

5. EP achieves bit/symbol lock moves to Polling.Config and starts transmitting TS2s, as shown in Figure 62.

	Packot		2.5		COM	Link	Lano	N ETS	Training Control	Data Pata	TQ1 Symbole	Time Dolta	Time Sta	mn
	563049	R⊷	x4	TS1	K28.5	PAD	PAD	255	00000	2.5 GT/s, 5 GT/s	D10.2	48.000 ns	0019.49518	0 394 s
	Packet	R→	2.5	TS1	COM	Link	Lane	N_FTS	Training Control		Data Rate		TS1 Symbols	Time Delta
	563050		х4		K28.5	PAD	PAD	94	00000	2.5 GT/s, 5 GT/s,	8 GT/s, Autonom	ous Change	D10.2	16.000 ns
Г	Packet	_	2.5	* 	COM	Link	Lane	N_FTS	Training Control	Data Rate	TS2 Symbols	Time Delta	Time Star	mp
	563051	R⊢	x4	152	K28.5	PAD	PAD	255	0000	2.5 GT/s, 5 GT/s	D05.2	48.000 ns	0019 . 495 18	0 458 s
Т	Packet	_	2.5	•	COM	Link	Lane	N_FTS	Training Control		Data Rate	•	TS1 Symbols	Time Delta
	563052	R→	x4	TS1	K28.5	PAD	PAD	94	00000	2.5 GT/s, 5 GT/s,	8 GT/s, Autonom	ous Change	D10.2	16.000 ns
	Packet		2.5		COM	Link	Lone	N ETS	Training Control	Data Rate	TS2 Symbole	Time Delta	Time Sta	mn
┢	562052	R⊢	2.J V/	TS2	K20 5	DAD		255			D05 2	48.000 ps	0010 405 19	0.522 e
	303033		A-4		R20.5	IT ND	TAD	233	0000	2.3 61/3, 3 61/3	003.2	40.000 113	10013.43310	0 322 3
	Packet	D	2.5) то1	COM	Link	Lane	N_FTS	Training Control		Data Rate		TS1 Symbols	Time Delta
	563054	K -	x4	101	K28.5	PAD	PAD	94	00000	2.5 GT/s, 5 GT/s,	8 GT/s, Autonom	ous Change	D10.2	16.000 ns
	Packet		2.5		COM	Link	Lone	N ETS	Training Control	Data Rate	TS2 Symbols	Time Delta	Time Sta	mn
	562055	R⊢	2.0 VA	TS2	K28.5	PAD		255		2.5 CT/s 5 CT/s	D05.2	48.000 ps	0010 405 19	0.596 c
	003000		74		K20.0	FAD	FAD	200	0000	2.0 GHS, 0 GHS	D05.2	40.000 115	0019.495 10	0.000.5

Figure 62 - EP in Polling.Config



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

	Packet 1243593	R-	2.5 x4	TS1	COM K28.5	Link Lane PAD PAD	N_FTS 94	Training Control 00000	2.5 GT/s, 5 GT/s,	Data Rate 8 GT/s, Autonomous Change	TS1 Symbols D10.2	Idle -64.000 ns	Tii 0019 .
	Packet 1243594	R-	2.5 x4	TS1	COM K28.5	Link Lane PAD PAD	N_FTS 94	Training Control 00000	2.5 GT/s, 5 GT/s,	Data Rate 8 GT/s, Autonomous Change	TS1 Symbols D10.2	Idle -64.000 ns	Tii 0019 .
I	Packet 1243595	R-	2.5 x4	EIOS	COM K28.5	EIOS Syr K28.3 K28.	nbols 3 K28.3	Time Stam	PED o Flow Graph - IEusi	ion IO SandyBridge IServerMB	1	_	_
	Packet 1243596	R⊣	2.5 x4	Link Eve Link Do	wn 20	me Delta 0.000 ns 0	Time 019 . 51	Stan 7 077 Direction		lon_to_oundyoingge_locitiento	Show Number	OfTransitions	
	Packet 1243597	R⊢	2.5 x4	TS2	CCM K28.5	Link Lane PAD PAD	N_FTS 255	Trair O Upst	ream (R<-) nstream (R->)	Dete	ect		
	Packet 1243598	R+	2.5 x4	TS2	COM K28.5	Link Lane PAD PAD	N_FTS 255	Trair O Both States S	Status				
	Packet 1243599	R+	2.5 x4	TS2	COM K28.5	Link Lane PAD PAD	N_FTS 255	Trair Current	s: Polling : Detect Polling	Pollin	g	Disabled	
	Packet 1243600	R+	2.5 x4	TS2	COM K28.5	Link Lane PAD PAD	N_FTS 255		< >	Ţ			

Figure 63 - RP Polling. Active timeout to Detect

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

J	Packet	-	2.5		COM	Link	Lane	N_FTS	Training Control	Data Rate		TS1 Symbols	Time Delta	Time Stamp
1	2038037	R→	x4	181	K28.5	PAD	PAD	94	00000	2.5 GT/s, 5 GT/s, 8 GT/s, Autonom	ous Change	D10.2	64.000 ns	0019 . 545 634 298 s
	Packet 2038038	R⊢	2.5 x4	TS2	COM K28.5	Link PAD	Lane PAD	N_FTS 255	Training Control 0 0 0 0	Data Rate TS2 Symbols 2.5 GT/s, 5 GT/s D05.2	Time Delta 0.000 ns	Time Star 0019 . 545 634	mp 4 362 s	
	Packet 2038039	R→	2.5 x4	TS1	COM K28.5	Link PAD	Lane PAD	N_FTS 94	Training Control 0 0 0 0 0	Data Rate 2.5 GT/s, 5 GT/s, 8 GT/s, Autonom	ous Change	TS1 Symbols D10.2	Time Delta 64.000 ns	Time Stamp 0019 . 545 634 362 s
	Packet 2038040	R⊢	2.5 x4	TS2	COM K28.5	Link PAD	Lane PAD	N_FTS 255	Training Control 0 0 0 0	Data Rate TS2 Symbols 2.5 GT/s, 5 GT/s D05.2	Time Delta 0.000 ns	Time Star 0019 . 545 634	np 1 426 s	
	Packet 2038041	R→	2.5 x4	TS1	COM K28.5	Link PAD	Lane PAD	N_FTS 94	Training Control 0 0 0 0 0	Data Rate 2.5 GT/s, 5 GT/s, 8 GT/s, Autonom	ous Change	TS1 Symbols D10.2	Time Delta 64.000 ns	Time Stamp 0019 . 545 634 426 s
	Packet 2038042	R←	2.5 x4	EIOS	COM K28.5	EI0 K28.3	DS Sym 3 K28.3	ibols 3 K28.3	Time Delta 0.000 ns 0	Time Stamp 0019 . 545 634 490 s				
-	Packet 2038043	R→	2.5 x4	TS1	COM K28.5	Link PAD	Lane PAD	N_FTS 94	Training Control 0 0 0 0 0	Data Rate 2.5 GT/s, 5 GT/s, 8 GT/s, Autonom	ous Change	TS1 Symbols D10.2	Idle 0.000 ns	Time Stamp 0019 . 545 634 490 s

Figure 64 - EP Polling.Config timeout to Detect

8. 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.



-						1		
	Packet	₽→ 2.5	, TS1	COM	Link	Lane N_FT	S Training Contr	trol Data Rate TS1 Symbols Time Delt
1	2084730	× x4	101	K28.5	0	PAD 94	00000	2.5 GT/s, 5 GT/s, 8 GT/s, Autonomous Change D10.2 64.000 n
	Packet 2084731	R← 2.5 x4	TS2	COM K28.5	Link PAD	😂 LTSSM F	low Graph - [Fusio	sion_IO_SandyBridge_IServerMB]
						Direction		Show Number Of Transitions
	Packet 2084732	R→ 2.5 x4	TS1	COM K28.5	Link 0	 Upstre Down: 	eam (R<-) stream (R->)	
	Packet 2084733	R← 2.5 x4	TS2	COM K28.5	Link PAD	States Sta	atus	
	Packet 2084734	R→ 2.5 x4	TS1	COM K28.5	Link 0	Previous: Current:	Polling Configuration	Polling Disabled
	Packet 2084735	R← 2.5 x4	TS2	COM K28.5	Link PAD	Next:	Configuration	
	Packet 2084736	$R \rightarrow \frac{2.5}{x4}$	TS1	COM K28.5	Link 0			Configuration Hot Reset
	Packet 2084737	R← 2.5 x4	TS2	COM K28.5	Link PAD			
•	Packet	_ 2.5	·	COM	Link			



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.

Bus/Signal	х	0	- 226	- 221	-216	- 211	- 206	- 201	- 196	- 191	- 186	- 181	- 176	- 171	-1
PIPERXOELEC	1	1													
- PIPERX1ELEC	1	1													
- PIPERX2ELEC	1	1													
- PIPERX3ELEC	1	1													
- PIPERXOPHYS	0	0													
- PIPETXRCVRD	1	1													
- PIPERXOSTATUS	4	4		4		<u>X3</u> X		4		X					
- PLLTSSMSTATE	02	02			02			04	08)	19\0A\04	(05) 2D	X	0	2	

Figure 66 - Chipscope LTSSM transition with Storage Qualification

The cause of this issue is explained in the <u>Intel Errata</u> 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
	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_Ink_rate when pl_link_gen2_cap was asserted, as shown in Figure 68.

±	'h 1F	1 F	(1C	(1D)(1E	(1F	(1C
	0					
⊒ 🜆 ∙pl_sel_lnk_width	'h 3	3				
	1					



By capturing the rxelecidle signal on all four lanes in ChipScope tool, it was seen that this signal was errorneously 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 X4 IS1	K28.5 P	AD PAD	255	00000	2.5 GT/s, 5 GT/s	D10.2	0.000 ns	0002.365157064s
Packet R← 2.5 15924 R← x4 TS1	COM L	ink Lane	N_FTS 255	Training Control	Data Rate	TS1 Symbols	Idle 0.000 ns	Time Stamp
Packet D. 2.5	COM L	ink Lane	N_FTS	Training Control	Data Rate	TS1 Symbols	Idle	Time Stamp
15925 X4	K28.5 P	AD PAD	255	00000	2.5 GT/s, 5 GT/s	D10.2	0.000 ns	0002.365157192s
Packet 2.5	COM L	ink Lane	N_FTS	Training Control	Data Rate		TS1	Symbols
15926 X4	K28.5 P	AD PAD	255	00000	2.5 GT/s, 5 GT/s	D10.2D10.2D1	0.2 D10.2 D10.	2D10.2D10.2D10.2D10.2D10.2
	K28.5 P	AD PAD	255	00000	2.5 GT/s, 5 GT/s	D10.2D10.2D1	0.2 D10.2 D10.	2D10.2D10.2D10.2D10.2D10.2
	K28.5 P	AD PAD	255	00000	2.5 GT/s, 5 GT/s	D10.2D10.2D1	0.2 D10.2 D10.	2D10.2D10.2D10.2D10.2D10.2
	K28.5 P	AD PAD	255	00000	2.5 GT/s, 5 GT/s	D10.2D10.2D1	0.2 D10.2 D10.	2D10.2D10.2D10.2D10.2D10.2
Packet 2.5	COM L	ink Lane	N_FTS	Training Control	Data Rate	TS1 Symbols	Time Delta	Time Stamp
15927 R- x1 IS1	K28.5 P	AD PAD	255	00000	2.5 GT/s, 5 GT/s	D10.2	32.000 ns	0002.365157288s
Packet 2.5	COM L	ink Lane	N_FTS	Training Control	Data Rate		TS1	Symbols
15928 X4	K28.5 P	AD PAD	255	00000	2.5 GT/s, 5 GT/s	D10.2D10.2D1	0.2 D10.2 D10.	2D10.2D10.2D10.2D10.2D10.2
	K28.5 P	AD PAD	255	00000	2.5 GT/s, 5 GT/s	D10.2D10.2D1	0.2 D10.2 D10.	2D10.2D10.2D10.2D10.2D10.2
	K28.5 P	AD PAD	255	00000	2.5 GT/s, 5 GT/s	D10.2D10.2D1	0.2 D10.2 D10.	2D10.2D10.2D10.2D10.2D10.2
	K28.5 P	AD PAD	255	00000	2.5 GT/s, 5 GT/s	D10.2D10.2D1	0.2 D10.2 D10.	2 D10.2 D10.2 D10.2 D10.2 D10.2
Packet 2.5	COM L	ink Lane	N_FTS	Training Control	Data Rate	TS1 Symbols	Time Delta	Time Stamp
15929 x1	K28.5 P	AD PAD	255	00000	2.5 GT/s, 5 GT/s	D10.2	32.000 ns	0002.365157352s

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 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 REFCLCK 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.

		Go 📌	0 @of ### -
Type /	Unstream	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				
🖶 🖂 🏉 🛍 🗰			G	o 🕐 0 🚔 of ### - Pac
🖃 🖺 All link speeds, x4	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
Eller 1 x4	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

Xilinx Answer 56616 – 7-Series PCIe Link Training Debug Guide



Figure 72 - Lecroy Traffic Summary

 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 75 - Rcvry.RcvrLock 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

-									1.4			
	-				_							
	Packet	R+ 2.5 TS1	COM L	ink Lan	IE N_FIS	5 Training Contro	I Data Rate		IS1 Syr	nbols		Time Delta
	847657	x4	K28.5 P	AD PA	D 255	00000	2.5 GT/s, 5 GT/s	D10.2 D10.2 [D10.2 D10.2 D10.2 D	10.2 D10.2 D10.2 D	10.2 D10.2	32.000 ns
			K28.5 P	AD PA	D 255	00000	2.5 GT/s, 5 GT/s	D10.2 D10.2 [D10.2 D10.2 D10.2 D	10.2 D10.2 D10.2 D	10.2 D10.2	
			K28.5 P	AD PA	D 255	00000	2.5 GT/s, 5 GT/s	D10.2 D10.2 [D10.2 D10.2 D10.2 D	10.2 D10.2 D10.2 D	010.2 D10.2	
			K28.5 P		D 255	00000	2.5 GT/s 5 GT/s	D10 2 D10 2 [D10 2 D10 2 D10 2 D	10 2 D10 2 D10 2 D	10 2 D10 2	
							2.0 0.0, 0 0.00	010.2010.20			10.2 2 10.2	
		Time Stamp										
		0051 . 770 259 936 s	s									
					-							
н	Packet	R→ 2.5 TS1	COM L	ink Lan	e N_FTS	S Training Control	Data Rate		-	rS1 Symbols		
IL	847658	x4	K28.5	0 PAI	D 194	00000	2.5 GT/s, 5 GT/s,	8 GT/s D10.2	D10.2 D10.2 D10.2	D10.2 D10.2 D10.2	D10.2 D10.2	D10.2
			K28.5	0 PAI	D 94	00000	2.5 GT/s, 5 GT/s,	8 GT/s D10.2	D10.2 D10.2 D10.2	D10.2 D10.2 D10.2	D10.2 D10.2	D10.2
			K28.5	0 PAI	D 194	00000	2.5 GT/s, 5 GT/s,	8 GT/s D10.2	D10.2 D10.2 D10.2	D10.2 D10.2 D10.2	D10.2 D10.2	D10.2
			K28.5		94	00000	2.5 GT/s .5 GT/s	8 GT/s D10 2	D10 2 D10 2 D10 2	D10 2 D10 2 D10 2	D10 2 D10 2	D10.2
			1020.0	0 174	<u> </u>	00000	2.5 6113, 5 6113,	0 0113 010.2	010.2 010.2 010.2	010.2 010.2 010.2	010.2 010.2	010.2
		Time Delta	Time Stamp)								
		32.000 ns 0051	. 770 259 9	968 s								
	Packet	- R+	COM	ink I an	ne N FTS	S Training Contro	L Data Rate	TS1 Symbol	s Time Delta	Time Stamp		
	847659	LTSSM Flow	Graph - [Su	perMicro	SandyBrid	dge V7 Run 1]						
l r	Daakat	Direction								United		
	047660	- R-	(V 5	how Number Of Transi	ons		Upstream	Downst	ream
	847660	Opstream	(R<-)						_			
l r	Packet	 Downstrea 	am (R->)			Detect			2.5	Polling	2.5	Polling
	847661	R+ 🔘 Both			/				25	Cfo I W Start	1 2 5 CFr	LW Start
	011001	States Status				/ TAK			2.3	CIGLEWISTAL		.EW.Start
	D	States States			/	/ + \					·	
			Figur	e 77 -	- Root	complex a	dvertisina li	nk numbe	er on all four	lanes		
			U			•	0					
Г	Dackat	26	COM	Link 1	ano M	ETS Training C	notrol Data Pa	10		C1 Sumbole		
H	Packet	R+TS1	Kdo c		ane ne	FTS Training Co			040.0.040.0.040.0	DAG O DAG O DAG (0.040.0.040	
L	64/66/	X4	K20.5		PAD Z	55 0000	0 2.5 G1/s, 5	G1/s D10.2 L	D10.2 D10.2 D10.2	D10.2 D10.2 D10.2	2 D10.2 D10	2 010.2
			K28.5	0 6	PAD 2	55 0000	0 2.5 GT/s, 5	GT/s D10.2 [D10.2 D10.2 D10.2	D10.2 D10.2 D10.2	2 D10.2 D10	.2 D10.2
			K28.5	0 8	PAD 2	55 0000	0 2.5 GT/s, 5	GT/s D10.2 [D10.2 D10.2 D10.2	D10.2 D10.2 D10.2	2 D10.2 D10	.2 D10.2
			K28.5	0.0	PAD 2	55 0000	0 2.5 GT/s 5	GT/s D10 2 f	010 2 D10 2 D10 2	D10 2 D10 2 D10 2	2 D10 2 D10	2 D10 2
			1020.0	<u> </u>			2.0 0110,0	0110 010.2 0		010201020104		2010.2
		Time Stamp										
		0051 . 770 260 256	6 s									
	Packet	2.5 Tet	COM	Link T	ane N_	FTS Training Co	ontrol Data	a Rate		TS1 Symb	ols	
Γ	847668	x4	K28.5	0 8	PAD !	94 0000	0 2.5 GT/s, 5	GT/s, 8 GT/s	D10.2 D10.2 D10.3	2 D10.2 D10.2 D10	.2 D10.2 D1	10.2 D10.2 D10
			K28.5	0.0	PAD	94 0.000	0 25 GT/s 5	GT/s 8 GT/s	D10 2 D10 2 D10	2 D10 2 D10 2 D10	2 D10 2 D1	10 2 D10 2 D10
			140.0				0 2.5 OT/5, 5	CT/s, 0 CT/s	D10.2 D10.2 D10.	010.2 010.2 010	2 010 2 0	10.2 D10.2 D10
			K20.5	0 1	PAD	94 0000	0 Z.5 GI/S, 5	GI/S, 8 GI/S	010.2 010.2 010.	2 010.2 010.2 010	2 010.2 01	10.2 010.2 010
			K28.5	0 6	PAD 19	94 0000	0 2.5 GT/s, 5	GT/s, 8 GT/s	D10.2 D10.2 D10.3	2 D10.2 D10.2 D10	1.2 D10.2 D1	10.2 D10.2 D10
		Trees Dalla	Trees Die									
		Time Delta	Time Sta	mp								
		40.000 ns 005	51 . 770 26	0 288 s								
Г	Pack #		-			10 41				-		
H	0470	LISSM Flow Graph -	SuperMicr	o Sandy	Bridge Vi	/ Run 1]					L.	
L	04/6	Disasting								Understand		
Г	Pack	Direction				Show	Number Of Transitions			Upstream	Downs	ueam
H	9470	Upstream (R<-)										
L	04/6	Downstream (R->))		(Datast			2.5	Polling	2.5	Poling
Г	Pack	Both			-	Detect			2.5	roung		
H	8476			/	1	ART			2.5	Cfg.LW.Start	2.5 Cf	g.LW.Start
L	0470	States Status		1						CENTRAL PROPERTY	25 00	1 W Accest
		Draviour: Configuration	0						2.5	Cig.LW.Accept	City City	ALW.ACCEPT

Figure 78 - Endpoint advertising link number on all four lanes

Polling

Previous: Configuration

Current: Configuration

Pac

Cfg.LN.Wait/Accept





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
 - o 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
 - o <u>http://www.xilinx.com/support/answers/50234.htm</u>

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.

- <u>http://www.xilinx.com/support/answers/40469.html</u>
 - 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)
- <u>http://www.xilinx.com/support/answers/54643.html</u>



 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 CPLLLOCK 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.

ChipScope Pro Core Inserter	
<u>F</u> ile <u>E</u> dit <u>I</u> nsert <u>H</u> elp	
	1 (?
DEVICE	ICON Select Integrated Controller Options
ICON	Parameters
Core Utilization BRAM Cou 0	Boundary Scan Chain USER1 💌
	< <u>P</u> revi Ne <u>x</u> t > New ILA Unit New ATC2 Unit
Messages	

Figure 80: Chipscope Pro Inserter - Boundary Scan Chain

• Select trigger width as required (Figure 81).



ChipScope Pro Core Inserter	
<u>F</u> ile <u>E</u> dit <u>I</u> nsert <u>H</u> elp	
다 🖻 🖶 🗢 🍁	?
DEVICE	ILA Select Integrated Logic Analyzer Options
	Trigger Parameters Capture Parameters Net Connections
U.I.LA	Trigger Input and Match Unit Settings
	Number of Input Trigger Ports: 1 💌 Number of Match Units Used: 1
Core Utilization BRAM Count: 1	TRIGO: Trigger Width: 8 Match Type: Basic w/edges Image: Second
	< <u>Previous</u> Ne <u>x</u> t> Remove Unit
Messages	
copy C:Xilinx_Bin_Projects_Project_PCle_ C:Xilinx_Bin_Projects_Project_PCle_	Cle_K/_Link_Debug\ISE_Coregen_Project\pcie_/x_v1_9\implement\xilinx_pcie_2_1_ep_/x.ngc => K7_Link_Debug\ISE_Coregen_Project\pcie_7x_v1_9\implement\xilinx_pcie_2_1_ep_7x_signalbrowser.ngo
•	



• 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.



Scope Pro Core Inserter				
File Edit Insert Help				9
	DEVICE Design Files			Select Device Options
	Input Design Netlist:	bugUSE_Coregen_Projectipcie_7x_v1_91implementwilinx_pcie_2_1_ep_7x.ngc	Browse	
Core Utilization	Output Design Netlist: Output Directory:	ougustCoregen_Projectipale_/X_v1_9umplementituilinx_pale_z_1_ep_/x.ngc Project_PCIe_K7_Link_DebugUSE_Coregen_Projectipale_7x_v1_9Umplement	Browse	
BRAM Count: 1	Device Settings Device Family: Kintex Use SRLs Use RPMs	7		
Messages	Cle K7 Link DebunVSE Co	>		•
C.Wilinx_Bin_Projects_Project_PCle_I	K7_Link_Debug\ISE_Corege			

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

• Select USER1 in 'Boundary Scan Chain'.

ChipScope Pro Core Inserter	
<u>File E</u> dit <u>I</u> nsert <u>H</u> elp	
	1 (? 1
DEVICE	ILA Select Integrated Logic Analyzer Options
	Trigger Parameters Capture Parameters Net Connections
	Capture Settings Data Width: 8 Clock Edge Clock Edge
Core Utilization	Data Depth: 1024 🔻 Samples 🗌 Data Same As Trigger
BRAM Count: 1	Trigger Ports Used As Data
	< Previous Next > Remove Unit
Messages C:\Xilinx_Bin_Projects_Project_P nac =>	Cle_K7_Link_Debug\ISE_Coregen_Project\pcie_7x_v1_9\implement\xilinx_pcie_2_1_ep_7x.
1	

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

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

ChipScope Pro Core Inserter	
<u>F</u> ile <u>E</u> dit <u>I</u> nsert <u>H</u> elp	
	* · · · · · · · · · · · · · · · · · · ·
	ILA Select Integrated Logic Analyzer Options
	Trigger Parameters Capture Parameters Net Connections
Core Utilization BRAM Cou 1	Net Connections
	< Previous Insert > Remove Unit
Messages	
C:\Xilinx_Bin_Projects_Project_ nt\xilinx_pcie_2_1_ep_7x.ngc =>	PCIe_K7_Link_Debug\ISE_Coregen_Project\pcie_7x_v1_9\impleme

Figure 84: Chipscope Pro Inserter - Net Connections

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

Select Net						\boxtimes	
Structure / Nets	Structure / Nets Net Selections						
•/[xilinx_pcie_2_1	_ep_7x]			•	Trigger Signals Data Signals		
- app [pcie_ap	p_7x]				Clock	Signals	
- ext clk.pipe	clock i [pcie 7x v1 9	9 pipe clock]					
• pcie 7x v1 9	i (pcie 7x v1 9)				Channel		
e ncie ton	i Incie 7x v1 9 ncie	tonl			CH-1		
	ripcie_/x_v1_3_pole	Zv v1 0 poio pipo	ninolinal		CH:2		
pcie_	hihe_hiheime_i (hcie	_/x_vi_9_pcie_pipe_	pipeiiiiej	= 3	CH:3		
• pcie_	/x_I [pcie_/x_v1_9_p	cie_/xj			CH:4		
Υ _axi_ba	asic_top [pcie_7x_v1_	_9_axi_basic_top]			CH:5		
•- rx_	_inst [pcie_7x_v1_9_	axi_basic_rx]			CH:6		
⊶ tx_	_inst [pcie_7x_v1_9_a	axi_basic_tx]			CH:7		
q− gt_top_i [pcie_7x_v1_9_gt_top]					
⊶ gt rx	valid_filter[0].GT_RX	VALID FILTER 7x in	st[pcie_7x_v1_9	gt 📥 🗄			
- nino	wranner i Incie 7v v	1 0 nino wrannorl		(
Net Name	Pattern:		▼ Filt	ег			
Net Name	Source Instance	Source Component	Base Type				
s_axis_tx_tdata<63>	axi_basic_top	pcie_7x_v1_9_axi_b	FDRE				
s_axis_tx_tdata<62>	axi_basic_top	pcie_7x_v1_9_axi_b	FDRE				
s_axis_tx_tdata<61>	axi_basic_top	pcie_7x_v1_9_axi_b	FDRE		TP0		
s_axis_tx_tdata<60>	axi_basic_top	pcie_7x_v1_9_axi_b	FDRE	_			
s_axis_tx_tdata<59>	axi_basic_top	pcie_/x_v1_9_axi_b	FURE	-			
s_axis_tx_tdata<50>	axi_basic_top	poie_/x_v1_9_axi_0	FDRE		Make Connections	Move Nets Up	
s axis tv trlata<56>	axi hasic ton	ncie 7x v1 9 axi h	EDRE	▼	Remove Connections	s Move Nets Down	
OK Cancel							

Figure 85: Chipscope Pro Inserter - Selecting Data Signals



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

Select Net						
Structure / Nets	Net Selections					
<pre>P-/[xilinx_pcie_2_1_ep_7x]</pre>		Trigger Signals Data Signals				
🕶 app [pcie_app_7x]	1000	Clock Signals				
← ext_clk.pipe_clock_i [pcie_7x_v1_9_pipe_clock]	1000	Channel				
<pre> pcie_7x_v1_9_i [pcie_7x_v1_9] </pre>	1000	CH:0 /pcie_7x_v1_9_i/pipe_clk				
pcie_top_i [pcie_7x_v1_9_pcie_top]	1000					
pcie_pipe_pipeline_i [pcie_/X_V1_9_pcie_pipe_pipeline]	=					
pole_rx_l[pole_rx_v1_9_pole_rx]	1000					
• rx inst[ncie 7x v1 9 avi hasic_rv]	1000					
• tx inst[pcie 7x v1 9 axi basic tx]	1000					
<pre>- qt top i[pcie 7x v1 9 qt top]</pre>	00000					
gt_rx_valid_filter[0].GT_RX_VALID_FILTER_7x_inst [pcie_7x_v1]						
pino wrappor i Incio 7x u1 0 pino wrappor	•					
Net Name Pattern: *pipe_clk* Filter		СРО				
Net Name Source Instance Source Compon Base Type	1000					
pipe_clk gt_top_i pcie_7x_v1_9_gt_top BUFGCTRL		Make Connections Move Nets Up				
Remove Connections Move Nets Down						
OK Cancel						

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>.

File Edit Insert Help Help Insert Help Insert Help Insert Help Insert Insert Remove Unit	Scope Pro Core Inserter		- • ×
DEVICE ILA Select Integrated Logic Analyzer Options Core Utilization Capture Parameters Net Connections Vet Connections Trigger Parameters Vet Connections Vet Connections Vet Connections Vet Connections <t< td=""><td>Eile Edit Insert Help</td><td>4</td><td>8</td></t<>	Eile Edit Insert Help	4	8
Insert Remove Unit Messages C:\Xilinx_Bin_Projects_Project_PCIe_K7_Link_Debug\ISE_Coregen_Project\pcie_7x_v1_9\imp \ Immediate the design from the Translate (NGDBUILD) step. Immediate the design from the Translate (NGDBUILD) step.	Core Utilization	ILA Select Integrated Logic Ana Capture Parameters Net Connections Trigger Parameters Net Connections P UNIT CH0: /pcie_7x_v1_9_i/pipe_clk P TRIGGER PORTS CH0: /pcie_7x_v1_9_i/pipe_clk	alyzer Options

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.

Rising	•				
1024	•				
🕱 Enable Storage Qualification					
🗌 Data Same As Trigger					
8	Range: 14096				
	Rising 1024 Jalificatio ger 8				


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

Trigger Port Width	8		Range: 1	256
Match Units	4	F		
Counter Width	Disable	•		
Match Type	basic wi	th edges	•	
Bit Values:	0,1,x,r,f,t	o,n		
Functions:	=,<>			

3. There are no changes needed while generating chipscope_icon

Modifications needed in PCIe wrappers

1. Create a storage qualifier (store_ltssm) for cfg_ltssm_state

```
reg [5:0]
            cfg_ltssm_state_reg0 = 6'b0;
           cfg_ltssm_state_reg1 = 6'b0;
reg [5:0]
reg [5:0] cfg_ltssm_state_reg2 = 6'b0;
                store ltssm = 1'b0;
req
always @ (posedge user_clk)
begin
   cfg_ltssm_state_reg0
                            <= cfg_ltssm_state;
                            <= cfg_ltssm_state_reg0;
   cfg_ltssm_state_reg1
   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;</pre>
  else
   store_ltssm
                  <= 1'b0;
end
```

2. 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

9	Trigger Setup - DEV:0 MyDevice0 (XC7VX69	90T) UNIT:0 MyILA0 (ILA)	
ž	Match Unit	Function	Value
atcl	P─	==	XXXX_XXX1
-	- 🗋 TRIG0[7]		X
	 _ Cfg_ltssm_state[5] 		Х
	 _ Cfg_ltssm_state[4] 		Х
	 – Cfg_ltssm_state[3] 		Х
	 – Cfg_ltssm_state[2] 		Х
	 – Cfg_ltssm_state[1] 		Х
	 – Cfg_ltssm_state[0] 		Х
	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)

(1)	Trigger Setup - DEV:0 MyDevice0 (XC7VX69	90T) UNIT:0 MyILAO (ILA)	
ž	Match Unit	Function	Value
atcl	← 📑 M0:TRIG0	==	XXXX_XXX1
-	P — □ M1:TRIG0	==	X000_001X
	- 🗋 TRIG0[7]		Х
	- 🗋 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		Х
	⊷ 📑 M2:TRIG0	==	XXXX_XXXX
	🕶 🚍 M3:TRIG0	==	XXXX_XXXX

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

2	Trigger Setup - DEV:0 MyDevice0 (XC7VX69	90T) UNIT:0 MyILA0 (ILA)	
×	Match Unit	Function	Value
atcl	← 🗂 M0:TRIG0	==	XXXX_XXX1
1	← 🚍 M1:TRIG0	==	X000_001X
	P-	==	X000_010X
	- 🗋 TRIG0[7]		Х
	- Cfg_ltssm_state[5]		0
	 — Cg_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_Itssm		X
	← 🗂 M3:TRIG0	==	XXXX_XXXX

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



9	Trigger Setup - DEV:0 MyDevice0 (XC7VX69	90T) UNIT:0 MyILAO (ILA)	
ž	Match Unit	Function	Value
atc	►	==	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	۲	TriggerCondition0	МО	* *
				-

Trigger Condition: TriggerCondition	0	
mgger condition. mgger condition		
Boolean Sequencer		
Number of Levels: 2 💌		Jse Contiguous Match Events Only
Level	Match Unit	Negate
1	M1	
2	M2	
		_
Trigger Condition Fountion		
Ingger Condition Equation		
	$M1 \rightarrow M2$	
	OK Cancel	

6. 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.

 Caption 	Type: Window	Windows: 1	Depth: 64	-	Position: 8	
alr	Storage Qualification:	All Data	3			



Storage Condition			×
○ All Data	OR Equation	Negate Whole Equation	on
Match Unit	Enable	Negate	
MO	v		•
M1			
M2			
M3			
			•
Storage Condition Equation			
	M0		
	OK Cancel		

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

▼ Trig	Add	Active	Trigger Condition Name	Trigger Condition Equation	
	Del		riggi Orinkovo		
 Captu 	Type:	Window Windows:	1 Depth: 64	Position: 8	
ē	Storag	e Qualification:	МО		

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)

Waveform - DEV:0	MyDe	vice0 ((XC7	VX690T) UNIT:0 MyILAO (ILA)										
Bus/Signal	х	0	-14		4	1	6	11 	16	21	26	31	36	41	46
⊶ cfg_ltssm_s	01	01		01 X 00 X 01 X 00 X 01 X	00 \ 01 \	02 (04	<u> </u>	08 (07 (08	X 09	X 0A X 10 X 0E	3 X OD X OC X	28 X	29 (2A (2B (0E (10
◦ pipe_tx_rat	0	0	11				0				X		2		



Capturing Signals in Vivado ILA

Synthesized Design - xc7k325tffg900-2 (active)					
Netlist	_ 🗆 🖻 ×	∑ Project Summary × ♦ Device ×			
		→			
pcie_7x_v1_8_0			YOYE		
• • • Primitives (1023)		+	X010	XII0	
Ist (pcie_7x_v1_8_top)		<u>→</u>	X0Y5	X1Y5	
		Q+			
		•	X0Y4	X1Y4	
			X0Y3	X1Y3	
🚴 Sources 🕅 Netlist			X0Y2	¥1¥2	
Properties	_ 🗆 🖻 ×		7012		
← → 100 k					
		268 1			
			ΧΟΥΟ	X1Y0	
		*			

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 <u>UG936</u>. This section illustrates how to grab signals for debugging in the PCIe example design. For more information, please refer to <u>UG936</u>.

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.

Find	×
Eind Nets	
Name matches tusertupt	8 +
▶	
Unique nets only	
Note: results applicable to logical nets only.	
	🗌 <u>M</u> atch Case
V Open in a new tag	
	OK Cancel

Figure 89 – Search 'nets' for Probing in Vivado ILA



Synthesized Design * - xc7k325tffg900-2 (active)					
Netlist	_ 🗆 🖻 ×	Σ Project Su	immary ×	🔷 Device 🛛 🗙	
물 개 🗉					
	_	51			
🔄 🗕 🖕 user_reset_out					X0Y6
💁 🧰 Primitives (1023)		4			
φ- inst (pcie_7x_v1_8_top)		\rightarrow			XOXE
🍅 🛅 Nets (901)		Q+			AUTS
💁 🧰 Primitives (2)	-	_			
•- a inst (pcie_7x_v1_8_core_top)					X0Y4
👃 Sources 🛛 🕅 Netlist					
Nice Deservation					V0V3
		<u> </u>			1013
🗲 🔶 🔯 隆					
🌦 user_Ink_up		₩_₩			X0Y2
Tarmstance pri count. 1		1			
Port count: 1		200			X0Y1
Route status: Has unplaced ports or pins	-	1000 100			
Number of podec: 0		_ P			Vovo
Caparal Attributes Instance Pins Aliases Nodes Pi	nNeder 4 N E	飞			XUTU
General Autodies Instance Fins Anases Nodes Fi		*			
Find Results - Nets - Name matches "*user*up*" (3)					
Q Id Name	Instance Pins	Flat Pins	Driver	Route Status	
1 inst/inst/n_467_user_lnk_up_int_reg_i_1	5	5	~	Has unplaced p	orts or pins
2 inst/inst/user_lnk_up	2	8	\checkmark	Has unplaced po	orts or pins
🧇 🏇 3 🛛 user_Ink_up	1	1	\sim	Has unplaced p	orts or pins

Figure 90 – PCle Example Design user_Ink_up Signal

Synthesized Design * -	xc7k325tffg900-2 (active)			
Netlist		$=\Box \mathrel{{\scriptstyle !}{\scriptstyle !}{\scriptstyle !}{\scriptstyle !}} \times$	vice	× 🗈 xilinx_pcie_2_1.
Z 🔄 🖪			B	:onstrs_1/imports/exam
pi_sei_im	<_rate			117 # User Clk Heart
⊢ 🍌 pl_transr	nit_hot_rst			118 #
- 🎠 pl_upstre	am_prefer_deemph		C7	119 set_property PAC
- 🎠 reg_clock	<_locked		a.	120 set_property PAC
- 🎠 rx_np_ok	t.		0	121 set_property PAC
- 🎠 rx_np_re	q			122 set_property PAC
- 🌦 s_axis_tx	_tlast		1	123
- 🌦 s_axis_tx	_tready			124
- 🌦 s_axis_tx	_tvalid			126 # Physical Const
– ♣ sys_clk			//	120 # Thysteat conse
- sys_rst_n				128 #
- 2% tx_cfg_gr	it			129 # SYS clock 100
- 2% tx_cfg_re	q		49	130 # signals are th
tx_err_di	тор		Щ 🔊	131 # Transceiver ar
- 🦉 user_lnk	🚳 Net Properties	Ctrl+E	-	132 # resources (FPG.
- 29 user_res	Unroute		- 🖓	133 # To use these p
	Boport Not Bouto Stati	10	102	134 # instantiated i
O- C Primitives (1	Report Net Route Stat	42		135 # Please refer t
💦 🕺 Sources 🛛 🕅 Netli	👋 Mark Debug			130 # (06) Tor guide
Not Proportion	Unmark Debug			138
Net Properties	🚽 💕 Assign to Debug Port			139 set property PAC
🗲 🔿 🚱 📐	Select Driver Pin			140 set_property PAC
璿 user_Ink_up	😕 Schematic	F4		141
Q CLASS	Show Connectivity	Ctrl+T		142
NAME	The show Hierarchy	F6		144 ##################
MARK_DEBUG	and show there eny		_	145 # End

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



Confirm Debug Net(s)
OK to debug user_Ink_up net?
This will create MARK_DEBUG constraints, which will be added to the target XDC constraint file when you save the design, causing synthesis to go out of date. To avoid having to rerun synthesis you can click Force-up-to-date.
Don't show this dialog again
OK Cancel



Figure 92 – user_Ink_up Net Properties after enabling 'MARK_DEBUG'

XILINX.

Save Constraints
Select a target file to write new unsaved constraints to. Choosing an existing file will update that file with the new constraints.
○ <u>C</u> reate a new file
File name:
Fil <u>e</u> location: 🐻 <local project="" to=""></local>
xilinx_pcie_2_1_ep_7x_1_lane_gen1_xc7k325t-ffg900-2_KC705
OK Cancel

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

Synthesized Design - xc7k325tffg900-2 (active)	
Sources 💷 🖻 🛎 🔀	vice × 🗈 xilinx_pcie_2_1_ep_7x_1_lane_gen1_xc7k325t-ffg900-2_KC705_REVC.xdc × 4
🔍 🛣 🚔 📑 💽	:onstrs_1/imports/example_design/xilinx_pcie_2_1_ep_7x_1_lane_gen1_xc7k325t-ffg900-2
Image: Sources (1) Image: Sources (1) Image: Sources (1) Image: Sources (2) Image: Sources (1) Image: Sources (2)	<pre>117 # User Clk Heartbeat = led_3 118 # 118 # 119 set_property PACKAGE_PIN AB8 [get_ports led_0] 120 set_property PACKAGE_PIN A88 [get_ports led_1] 121 set_property PACKAGE_PIN AC9 [get_ports led_2] 122 set_property PACKAGE_PIN AB9 [get_ports led_3] 123 124 X 125 ###################################</pre>
Hierarchy IP Sources Libraries Compile Order	 132# resources (FPGA input pins) associated with each GT Transceiver. 133# ro use these pins an IBUFDS primitive (refclk_ibuf) is 134# instantiated in user's design. 135# Please refer to the Virtex-7 GT Transceiver User Guide 136# (IG) for quidelines reparting clock resource selection
Sources Netlist	137 #
Source File Properties L × ↓ → ○○ ▷	<pre>138 139 set_property PACKAGE_PIN U8 [get_ports sys_clk_p] 140 set_property PACKAGE_PIN U7 [get_ports sys_clk_n] 141 142</pre>
Location: /home/deepeshm/h/LFAR_Projects/pcie_ Type: XDC Size: 6.2 KB Modified: Today at 20:55:52 PM Consideration: Size: 72 V1.8.0 example sets (construction)	<pre>143 144 ###################################</pre>

Figure 94 – MARK_DEBUG Constraint in PCIe Example Design XDC File



Generating Debug Cores (Set up Debug)

pcie_7x_v1_8	3_0_example - [/home/deepesi	hm/h/LFAR_Proje
<u>F</u> ile <u>E</u> dit F <u>l</u> ow	<u>T</u> ools <u>W</u> indow Layout ⊻iew <u>H</u> el	p
🏂 😂 🖺 🐼	Eloorplanning	• 📀 🍪 🗞 🔼
Flow Navigator	<u>I</u> /O Planning	esign * – xc7k325
0、 🔀 🚔	<u>T</u> iming	▶
	🕍 Sch <u>e</u> matic 🛛 🛛 F4	
Project Manag	Show <u>C</u> onnectivity Ctrl+T	pi_sel_ink_rate
🚳 Project	盂 S <u>h</u> ow Hierarchy F6	pl_transmit_hot_rs
😽 Add Sor	🖩 Report Utili <u>z</u> ation	pl_upstream_pref
🚛 IP Catal	Report Power	* reg_clock_lockec
	✓ Report <u>D</u> RC	rx_np_req
 IP Integrator 	Markeport <u>N</u> oise	s_axis_tx_tlast
👫 Create	Report Clock Networks	s_axis_tx_tready
Doen B	🌲 IP Pac <u>k</u> ager	s_axis_tx_tvalid
	Run Tcl Script	svs rst n
 Simulation 	Set up Debug	tx_cfg_gnt
🄞 Simulati	C <u>u</u> stom Commands	tx_cfg_req
🔍 Run Sim	🚳 Project <u>S</u> ettings	tx_err_drop
.	😽 Options	user_ink_up

Figure 95 – Generating Debug Cores

🗌 Set up Debug 📲 🚬	×	
	Set up Debug	
	This wizard will guide you through the process of choosing nets and connecting them to debug cores.	
	The wizard is automatically populated with any selected nets and with nets from the Unassigned Debug Nets folder.	
VIVADO.	To continue, click Next	
	< <u>Back</u> <u>Next</u> > <u>Finish</u> Cancel	

Figure 96 – Setup Debug GUI

XILINX.

Set up Debug Specify Nets to Debug Specify Nets for debugging		×
Name pcie_7x_v1_8_0_i/inst/reg_clock_locked pcie_7x_v1_8_0_i/inst/sys_rst_n pcie_7x_v1_8_0_i/inst/user_lnk_up	Clock DomainDriver TyTRIGDATAPIPE_OOBCLFDCEImage: Clock DomainImage: Clock DomainImage: Clock DomainPIPE_USERCIBUFImage: Clock DomainImage: Clock DomainImage: Clock DomainImage: Clock DomainPIPE_USERCFDREImage: Clock DomainImage: Clock DomainImage: Clock DomainImage: Clock DomainPIPE_USERCFDREImage: Clock DomainImage: Clock DomainImage: Clock DomainImage: Clock DomainPIPE_USERCFDREImage: Clock DomainImage: Clock DomainImage: Clock DomainImage: Clock DomainImage: Clock DomainFDREImage: Clock DomainImage: Clock DomainImage: Clock DomainImage: Clock DomainImage: Clock DomainFDREImage: Clock DomainImage: Clock DomainImage: Clock DomainImage: Clock DomainImage: Clock DomainFDREImage: Clock DomainImage: Clock DomainImage: Clock DomainImage: Clock DomainImage: Clock DomainFDREImage: Clock DomainImage: Clock DomainImage: Clock DomainImage: Clock DomainImage: Clock DomainFDREImage: Clock DomainImage: Clock DomainImage: Clock DomainImage: Clock DomainImage: Clock DomainFDREImage: Clock DomainImage: Clock DomainImage: Clock DomainImage: Clock DomainImage: Clock DomainFDREImage: Clock DomainImage: Clock DomainImage: Clock DomainImage: Clock DomainImage: Clock DomainFDREImage: Clock Domain <t< td=""><td></td></t<>	
Add/Remove Nets	Nets to debu	g: 3
<	Back Next > Finish Cance	2

Figure 97 – Selected Nets for Probing in Chipscope

Add/Remove Nets		×
Name T matches	◎ +	
 ✓ Unique nets only Note: results applicable to logical nets only. ✓ Eilter out non-debuggable nets 		Match case
	Find	
- Find Results (0) 章 圣	Nets to Debug (3) Image: State	

Figure 98 – Add/Remove nets in Setup Debug





Figure 99 - Set up Debug Summary

Synthesized Design * - xc7k325tffg900-2 (active)
Netlist _ 🗆 🖻 🗶 🗡
🕅 xilinx_pcie_2_1_ep_7x
💁 🧰 Nets (440)
💁 🛅 Primitives (80)
🛛 🕘 app (pcie_app_7x)
- 🛛 dbg_hub (dbg_hub_CV)
ext_clk.pipe_clock_i (pcie_7x_v1_8_0_pipe_clock)
@- 2 pcie_7x_v1_8_0_i (pcie_7x_v1_8_0)
- 🖸 u_ila_0 (u_ila_0_CV)
u_ila_1 (u_ila_1_CV)
👃 Sources 🔋 Netlist 👹 Debug

Figure 100 – Chipscope Debug Cores in Netlist Window

E XILINX.

Synthesized Design * - xc7k325tffg900-2 (active)	
Debug	_ 🗆 🖻 ×
으, 🔀 🖨 🛃 🐳 🔐 📸	
Name	
♀-II dbg_hub (labtools_xsdbmasterlib_v2)	
v-II u_ila_0 (labtools_ilalib_v2)	
∲- 🗃 CLK (1)	
PROBEO (1)	
└ Ch 0 (pcie_7x_v1_8_0_i/inst/sys_rst_n)	
PROBE1 (1)	
Gh 0 (pcie_7x_v1_8_0_i/inst/user_lnk_up)	
Q-II u_ila_1 (labtools_ilalib_v2)	
∲- 🗃 CLK (1)	
└ 🥑 Ch O (PIPE_OOBCLK_IN)	
 	
Ch 0 (pcie_7x_v1_8_0_i/inst/reg_clock_locked)	
— Contraction Contractic Con	
💑 Sources 🛛 🕅 Netlist 👋 Debug	

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.



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	

Table 12 - Special Symbols

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. <u>UG936</u>, Vivado Design Suite Tutorial, Programming and Debugging

Revision History

07/29/2013 - Initial release