

Introduction

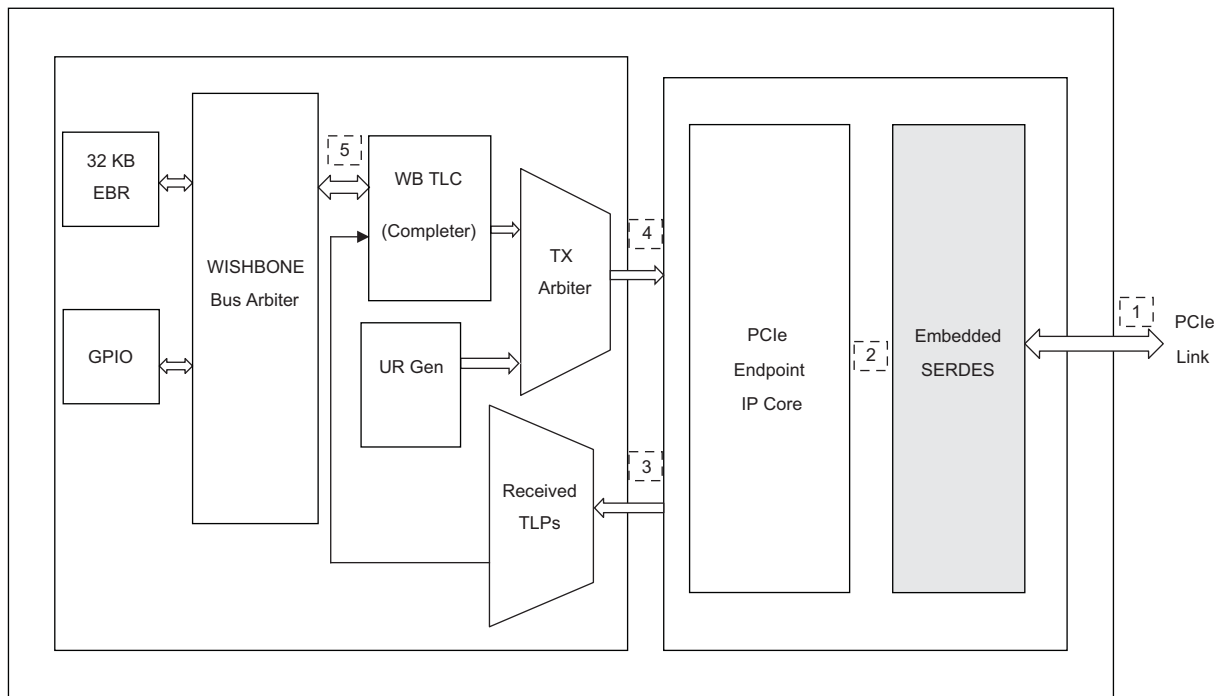
Various techniques are available for observing circuit activity in the Lattice PCI Express (PCIe) IP core in the LatticeECP3 Versa Evaluation Board PCIe Basic Demo. Observations are performed using the Lattice Diamond® Reveal Analyzer and Reveal Inserter tools. A clear understanding of the flow of packets through the different interfaces of the design is required. You also need to identify the types of packets on these interfaces by analyzing the packet header.

You can track packets through different interfaces in the core, select signals for triggering in Reveal Analyzer, and set the correct trigger to capture packets on different interfaces.

The main objective of this technical note is to guide you on using the Power-On-Reset (POR) feature in Reveal and going into the low level details of the core. These techniques can be utilized to help debug any design. Information related to both link training and PCIe packet traffic using the POR feature are also provided. Wherever needed, screen captures of the Reveal Analyzer interface are provided to illustrate how signals and packets are analyzed.

Signal and Packet Analysis Interfaces

Figure 1. Block Diagram



The main interfaces are as follows:

- PCIe Link
- Embedded SERDES
- TRN TX
- TRN RX
- Wishbone

The Reveal Analyzer is used to capture and track a particular packet along each of the interfaces. It does not offer much flexibility in triggering for the correct capture with only one trigger port. If you need to trigger only when a completion packet is presented to the core for a particular incoming read, you need to trigger only after these two conditions are met:

- memory read packet on the receive interface
- corresponding completion packet on the transmit interface

The techniques for capturing packets in similar scenarios are discussed in the following sections.

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. Please refer to the [PCIe Express Base Specification](#) for detailed descriptions of LTSSM ((Link Training Status State Machine) states. The section provides the list of signals to capture for debugging a link training issue using the Reveal tool along with the detailed analysis of the signals. A number of Reveal waveform screen captures are provided for each LTSSM state to illustrate the toggling of signals. If you are capturing signals using Reveal, compare your captures with the screen captures provided in this document to make sure that the signals in your design are toggling correctly. The screen captures are taken from the PCIe Basic Demo of LatticeECP3 Versa Evaluation Board.

LTSSM States

The signal below indicates in which LTSSM state the core is currently. The Core may go to recovery state to achieve bit lock and symbol lock. If the Reveal Analyzer capture shows frequent transition to the recovery state, this normally indicates a noisy link.

- phy_ltssm_state[3:0]
- phy_ltssm_substate[2:0]

Table 1 in “Appendix A: Description of LTSSM States” shows the different states of the link training state machine as indicated by phy_ltssm_state and phy_ltssm_substate.

Inserting Reveal

You can debug a PCIe Demo using the POR-Debug feature in Reveal Inserter. This feature is similar to the current logic analyzer core with the addition of a trigger enable signal. A trigger enable input is added to the current settings (such as TU (Trigger Unit), TE (Trigger Expression), Sample Clock, Sample Enable, and others).

Trigger Enable is the equivalent of the run button. When this input is active, the LA core is armed and looking for a trigger condition as defined by the TU and TE settings. The trigger enable should have the same options as sample enable.

You should select an appropriate signal for trigger enable, just as you would identify the sample enable.

This tutorial also shows techniques on how to trigger and capture LTSSM states which occur during the enumeration process. Once the FPGA is configured with bit stream containing Reveal data, the debug core is enabled for triggering based on the trigger enable signal. If a trigger condition occurs, the data (such as LTSSM States) is captured, saved, and uploaded to Reveal Analyzer when the PC restarts.


Using the POR Feature

Your goal is to test for conditions which occur shortly after the device is powered on. To achieve this, the debug core needs to preset and be armed for triggering conditions before the Reveal Analyzer is started or connected to the hardware. Specifically, this feature is useful in debugging the PCIe IP Core to capture early LTSSM states which occur shortly after the device is configured with the hardware, such as during the device enumeration process.

The following steps show how the early LTSSM states in PCIe Demo are captured using the POR feature when the current system with Windows XP as OS in which ECP3 Versa Kit is interfaced is getting enumerated (i.e. during PC restart). This LTSSM States will be captured and stored so that they can be later uploaded to Reveal Analyzer when the PC has started and Reveal Analyzer is opened and Connected to the hardware.

Note: When using the POR feature in Reveal Analyzer, the Reveal Inserter only supports one TE. You can, however, still use up to 16 TUs depending on targeted device family.

Step 1: Create a new .rvl file

1. Click the **Reveal Inserter** button  to create a new por_debug.rvl file.
2. Enter **clk_125** in Sample Clock to capture the Samples in Reveal Analyzer.
3. Select **2048** in Buffer Depth.
4. Select **Trigger Enable** to enable POR Debug.
5. Select an appropriate trigger enable signal which occurs shortly after power on.

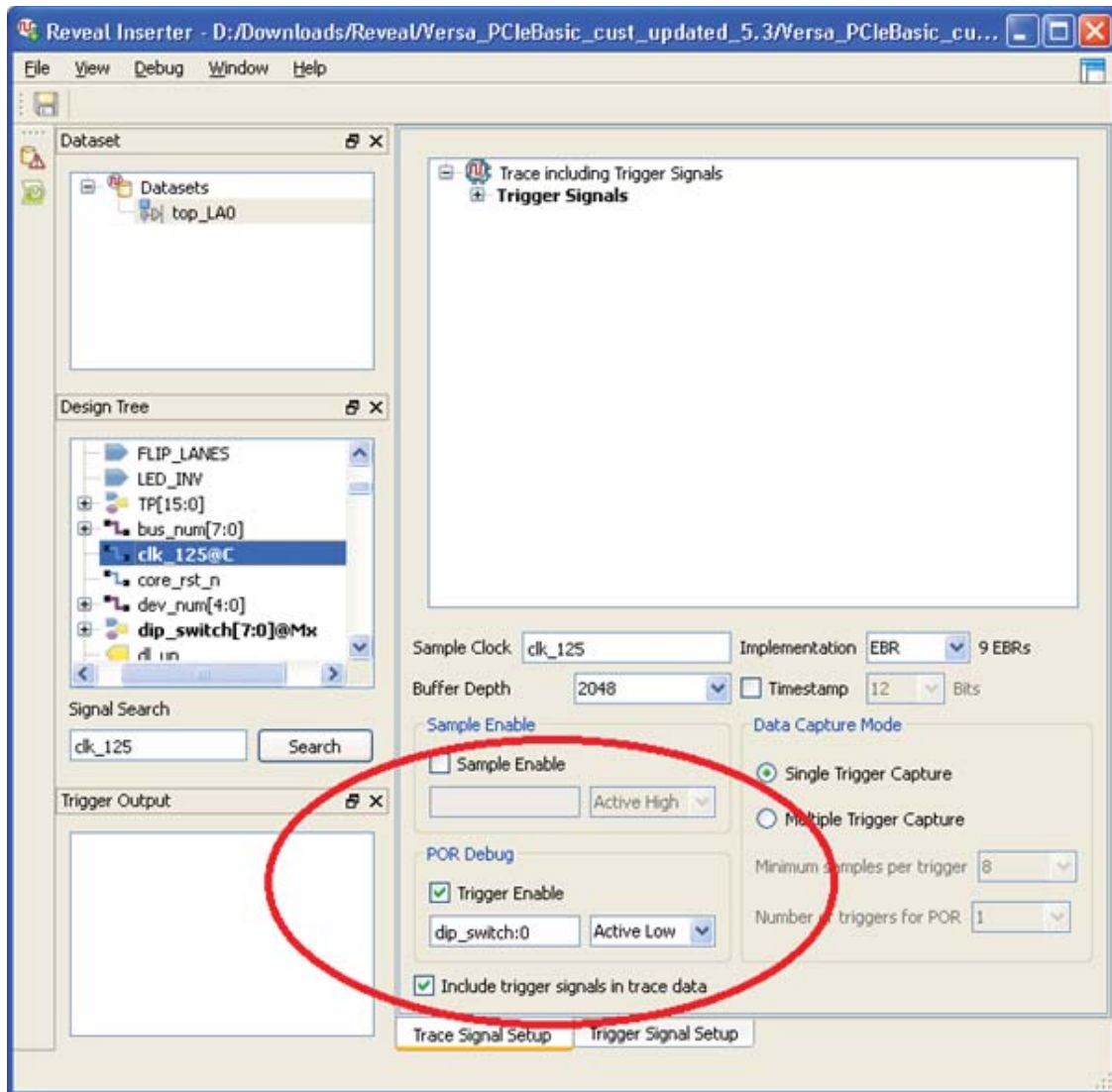
For example, dip_swich:0, an SW4:1 switch on the LatticeECP3 Versa board in ON position is set as Trigger Enable signal.

Once the FPGA is configured, the debug core is enabled for triggering based on the trigger enable signal, which is "dip_swich:0" in this example.

As soon as the trigger condition occurs and the input trigger enable signal is active (i.e. dip_swich:0 -> '1'), the LA Core is armed and looks for trigger condition as defined in the TU and TE settings in Reveal Inserter.

Figure 2 shows the settings in Reveal Inserter for creating a new .rvl file.

Figure 2. Select Clock Signal and Trigger Enable for POR Debug

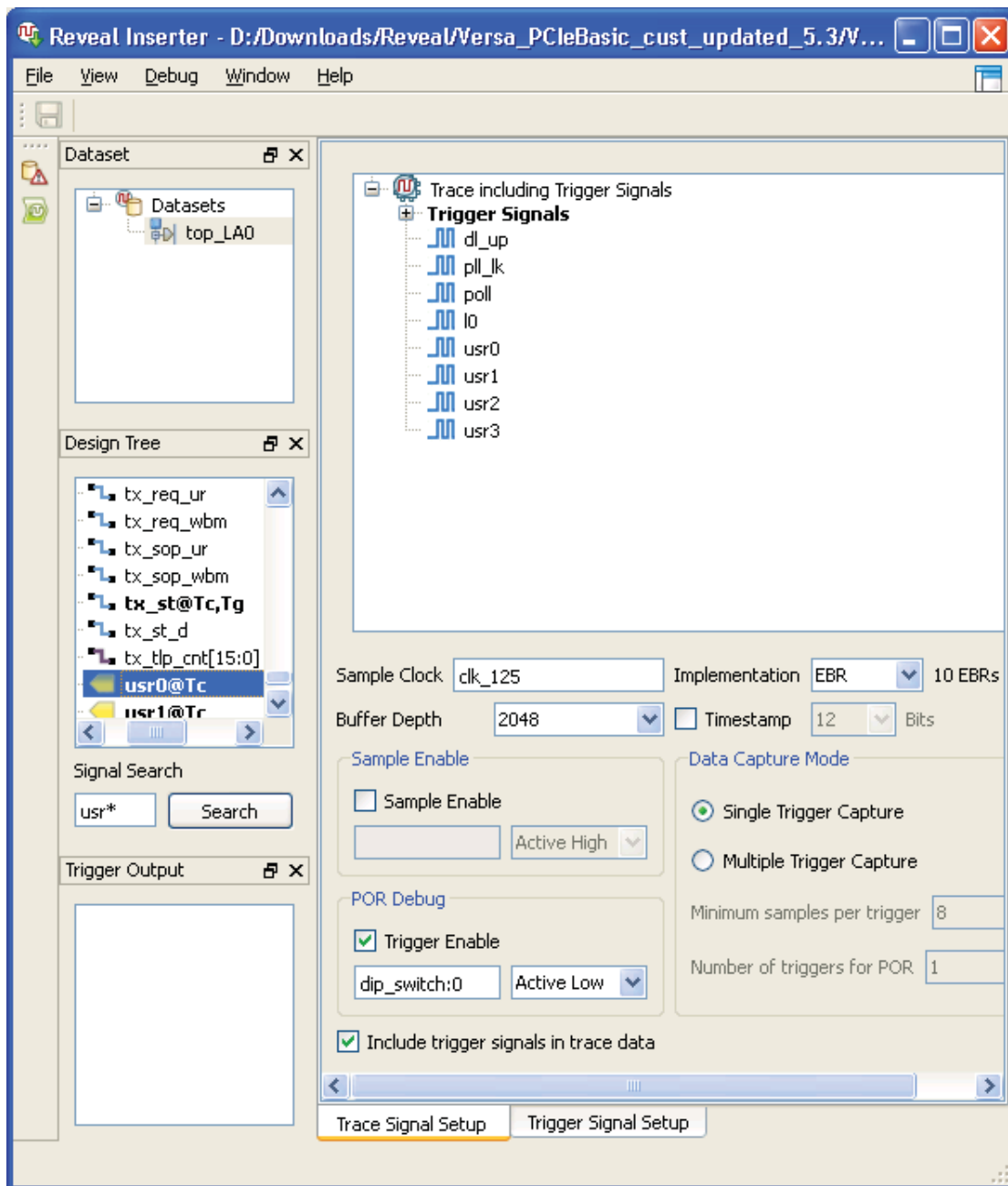


Step 2: Select trace signals

1. In Design Tree, select the signals for tracing in Signal Search.
2. Once the signals are selected, drag and drop the selected signals to the Trace Signal Window.

Figure 3 shows the list of signals added in the Trace Signal Setup section. These signals are related to link training (such as the initial setup). The dl_up is also listed in this section. When this signal is asserted, it indicates that the core has finished link training and is now in L0 state. Whenever you are debugging issues related to link training, it would be useful to check the status of these signals in Reveal Analyzer. These signals should be checked as the first step in debugging link training issues. Please note that all signals should toggle correctly.

Figure 3. Select Trace Signals



Step 3: Select trigger signals

1. Select the **Trigger Signal** tab.
2. Click the **Add** button in TU1 section.
3. Select TU Signals for TU1.

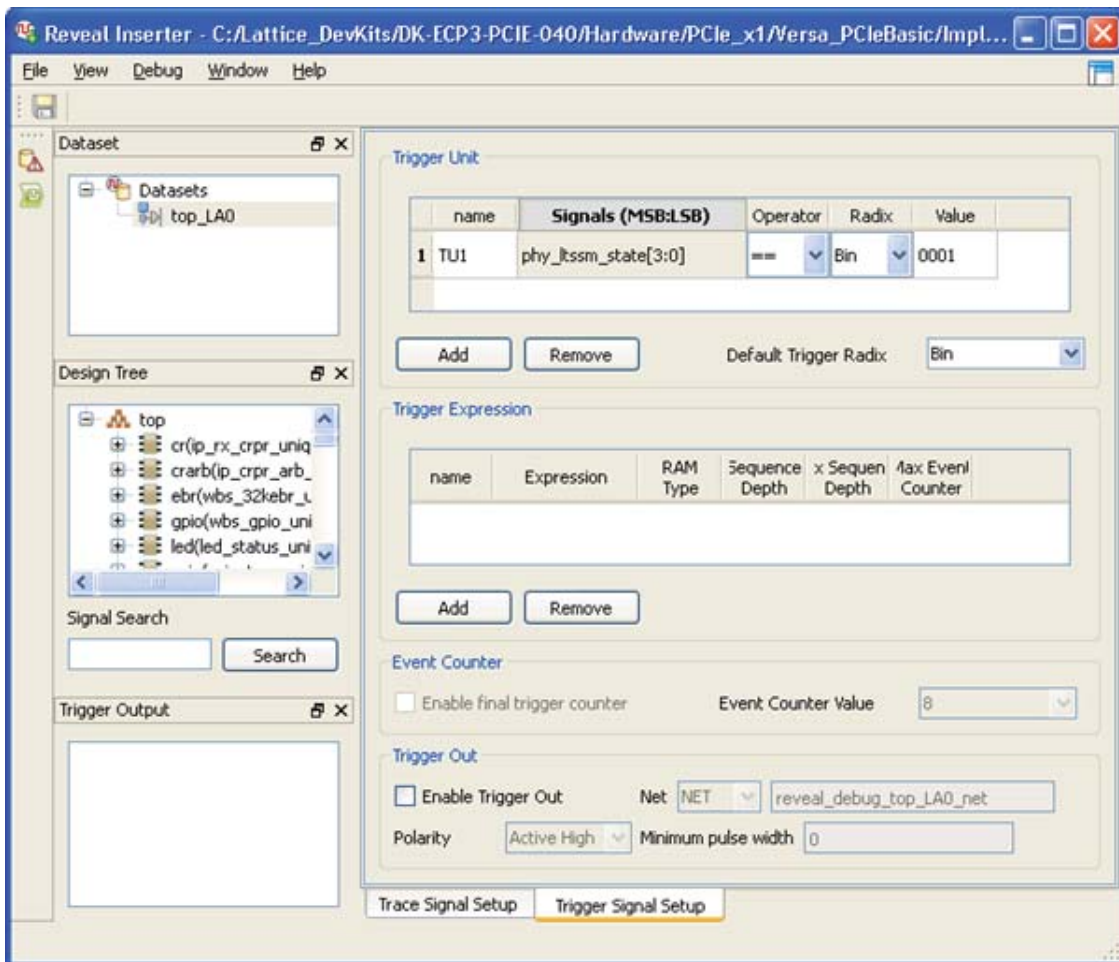
Trigger Unit 1 (TU1)

The TU1 port includes the LTSSM signals, which are the main signals when debugging link training issues. When a link problem occurs, always check the status of these signals in Reveal Analyzer to see the core state. Under normal working conditions, the signal shows the binary value 0011, which indicates that the core is in active L0 state. If there is a problem with the link due to signal integrity issues on the board, the LTSSM frequently goes into recovery state.

If the link is not coming up, you can use the POR feature to trace the early LTSSM states which occur during the enumeration process, such as when the PC with which the Versa kit is interfaced is getting rebooted. When the trigger enable signal for POR is active or the trigger condition occurs, capture and store the early LTSSM states. These occur when the PC is restarted, such as 0001-> Polling state for example. You can see the captured LTSSM signal states when the PC has restarted and the Reveal Analyzer is opened and connected to the hardware.

Figure 4 shows the list of signals selected in Trigger Unit 1 (TU1).

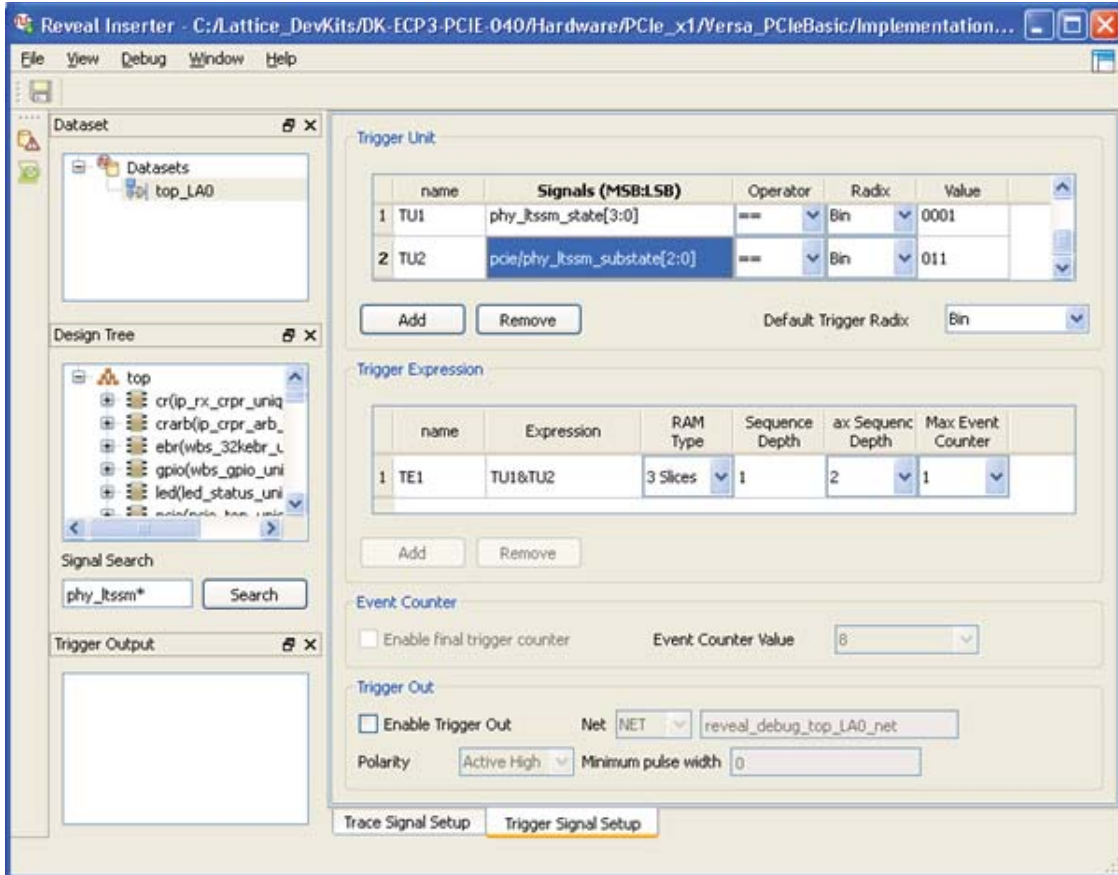
Figure 4. Add Signals in Trigger Unit 1 (TU1)



Trigger Unit 2 (TU2)

The TU2 port includes the phy_ltssm_substate signals which are sub states of phy_ltssm. These states are described in Table 1 in “Appendix A: Description of LTSSM States”.

Figure 5. Add Signals in Trigger Unit 2 (TU2)



Trigger Unit 3 (TU3)

The TU3 port includes the signals coming out of the Serdes interface. Probing the signals on this interface helps determine whether there are any issues at the physical level. If the host is sending packets and no toggling on the rxdata or txdata occurs, this indicates that the Serdes signals are not properly constrained. During Link Training, the physical layer Ordered Sets TS1s and TS2s are exchanged with the link partners. If the link is not coming up, always check this interface to make sure TS1s and TS2s are correctly exchanged.

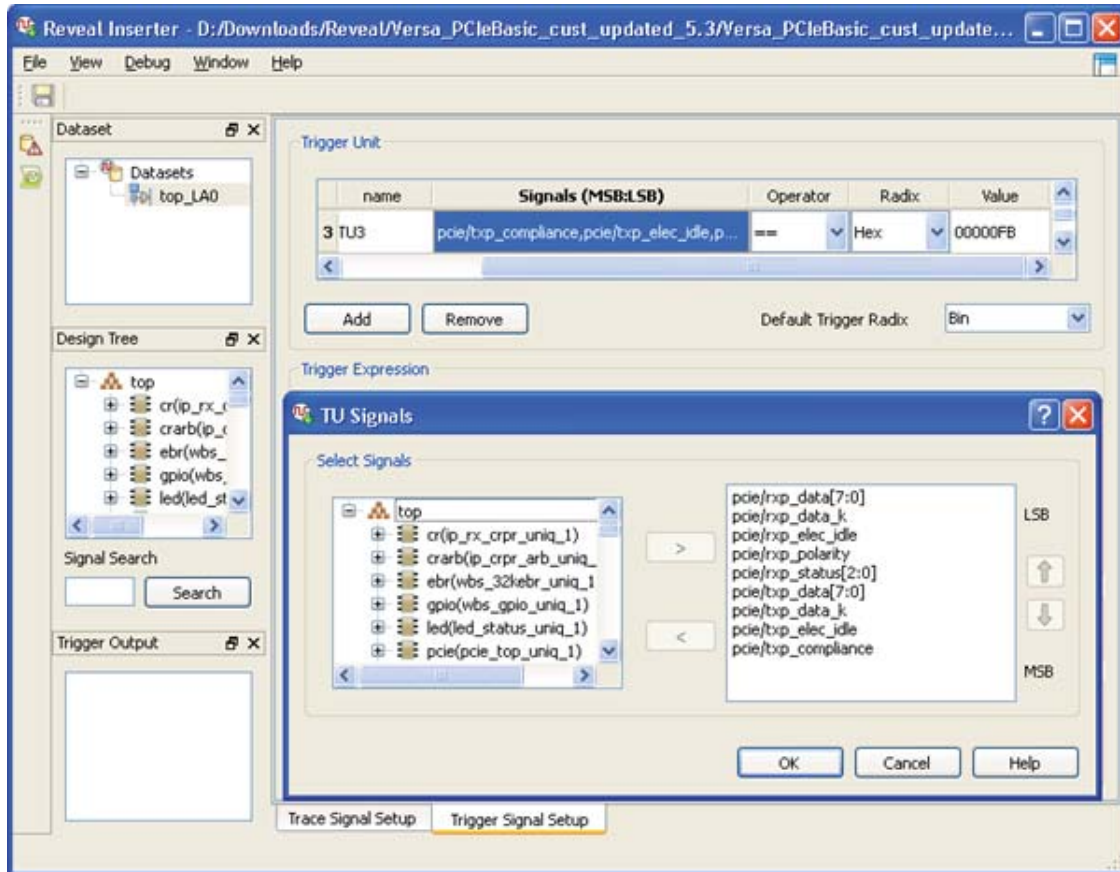
The data on the txdata and rxdata contains the control characters such as FB (Start of TLP), FD (End of TLP) and so on. If you think that a packet is not coming into the core, or is not going out, probe rxp_data and txp_data respectively and look for the FB character, followed by the FD character. The FB character is qualified by rxp_data_k signal indicating the corresponding character on rxp_data is a control character. The presence of FB on rxp_data or txp_data indicates the presence of a TLP.

For example, if you are generating an MSI Interrupt and the MSI packet is not making it to the root complex, the best way to check whether or not the packet has gone out of the core. There is no signal in the core that indicates that a packet has successfully transmitted to the PCIe link. You can verify by checking for FB on the txp_data, which indicates that the packet has gone out of the core.

MSI packet has a definite size. Check for FB on the Serdes Interface on txp_data and then FD after that. Count the number of bytes between these two characters, if it is equal to valid MSI message, this indicates that the MSI packet generated by the core has gone out.

Figure 6 shows the list of signals selected in Trigger Unit 3 (TU 3).

Figure 6. Add Signals in Trigger Unit 3 (TU3)

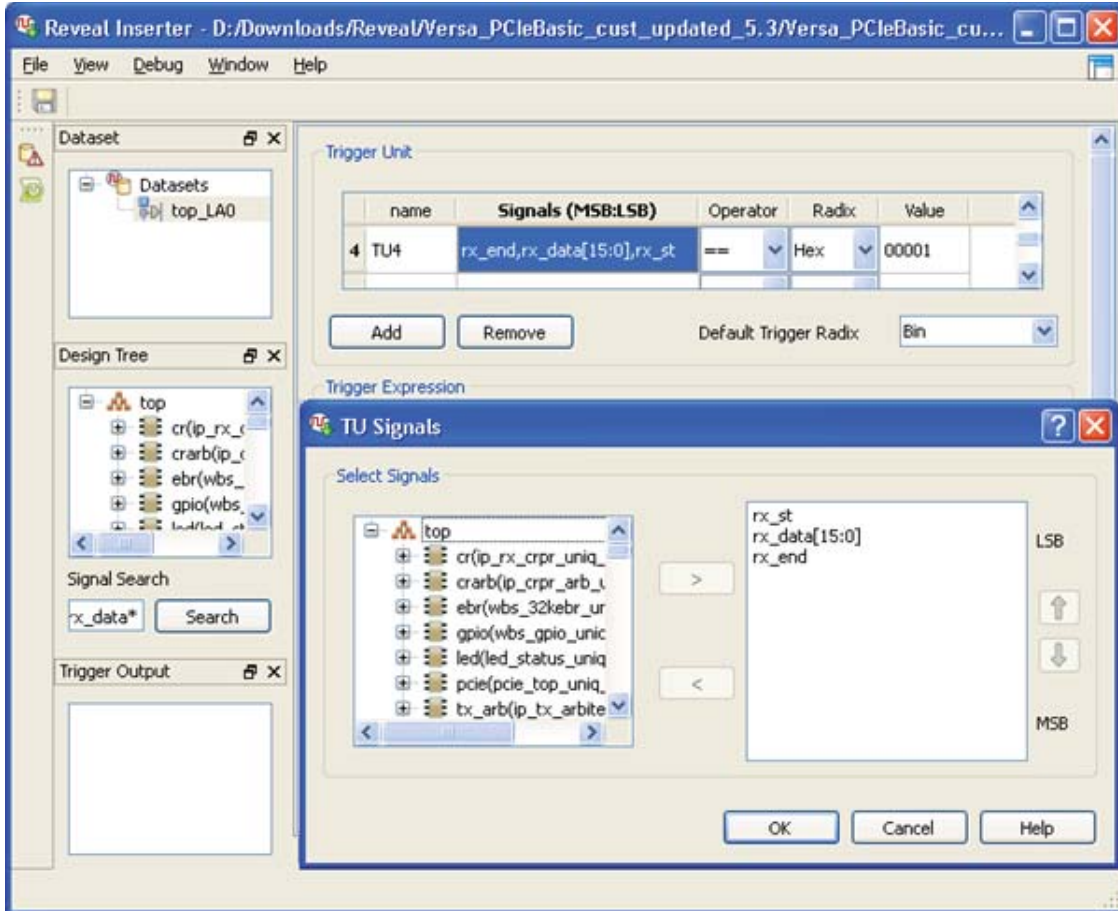


Trigger Unit 4 (TU4)

The TU4 port includes all signals at the receive side of User Interface. All incoming TLP packets from the transaction layer of Lattice PCIe Endpoint IP Core can be tracked here. If a packet is not coming towards the user interface, check this interface triggering on rx_st and rx_end signals and verify if it is a valid packet or not.

For example, in Figure 7, the trigger is set on rx_st to check the validity of the data coming on rx_data.

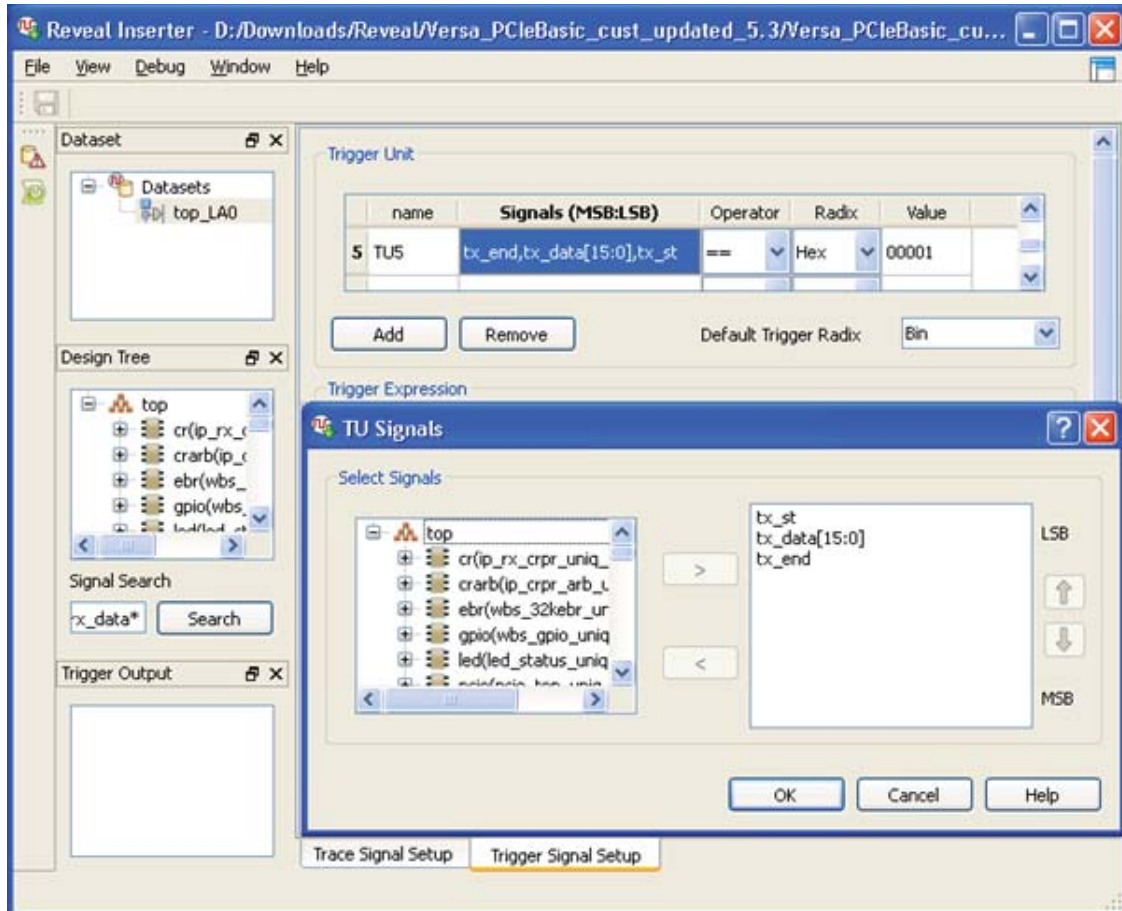
Figure 7. Add Signals in Trigger Unit 4 (TU4)



Trigger Unit 5 (TU5)

Similar to incoming packets the outgoing packets can also be tracked before they are transmitted towards the Serdes interface.

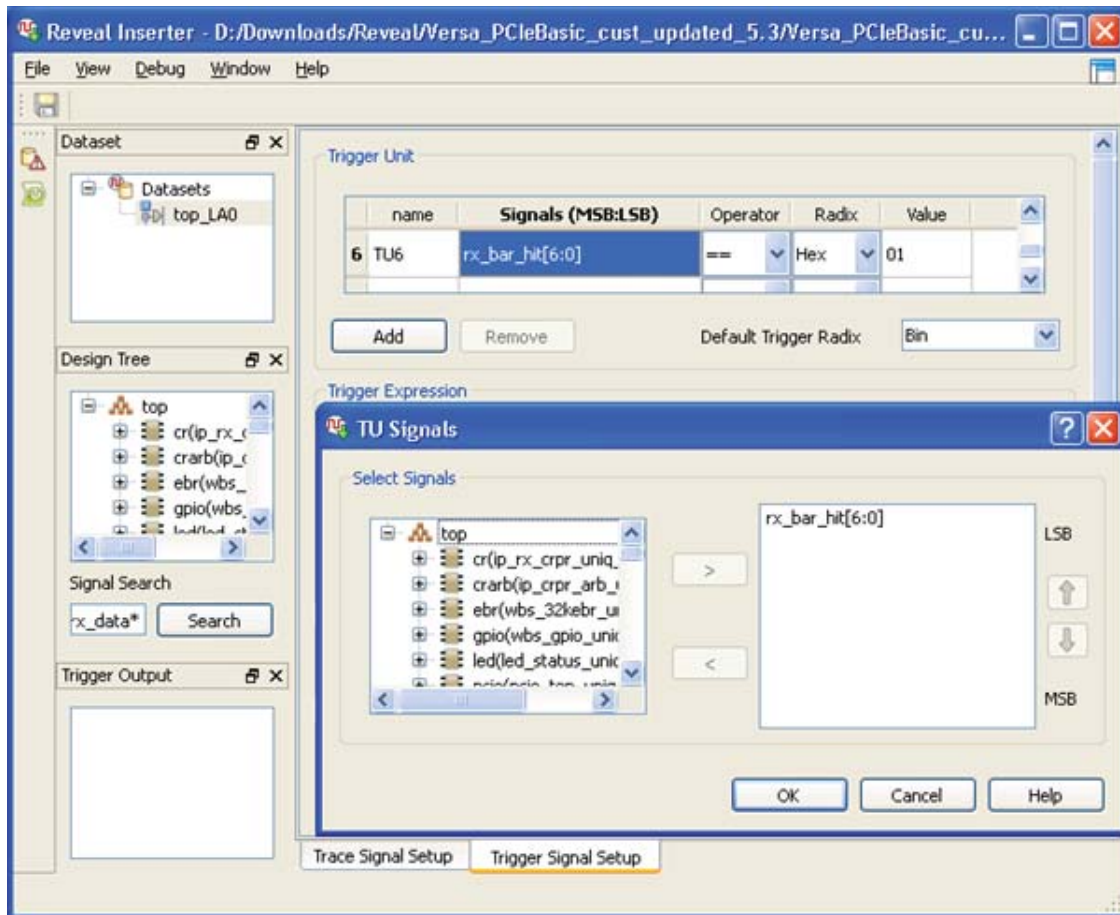
Figure 8. Add Signals in Trigger Unit 5 (TU5)



Trigger Unit 6 (TU6)

The TU6 port includes all receive side signals between the core and the user application. If the host is sending packets downstream and you think your backend application is not receiving those packets, the first thing to do is to check this interface. One probable cause for the packet not coming in to the user interface could be that the incoming packet is not hitting the BAR range. If this is the case, the corresponding signals `rx_bar_hit` is not asserted. In this case, make sure that the host is targeting the correct memory address. To check whether or not any packets are making it to the user application, set your trigger on `rx_st` signal. If the host is performing memory read from the Endpoint user application and you are not getting the corresponding completion back from the Endpoint, verify the memory read TLP on the `trn` receive interface.

Figure 9. Add Signals in Trigger Unit 6 (TU6)

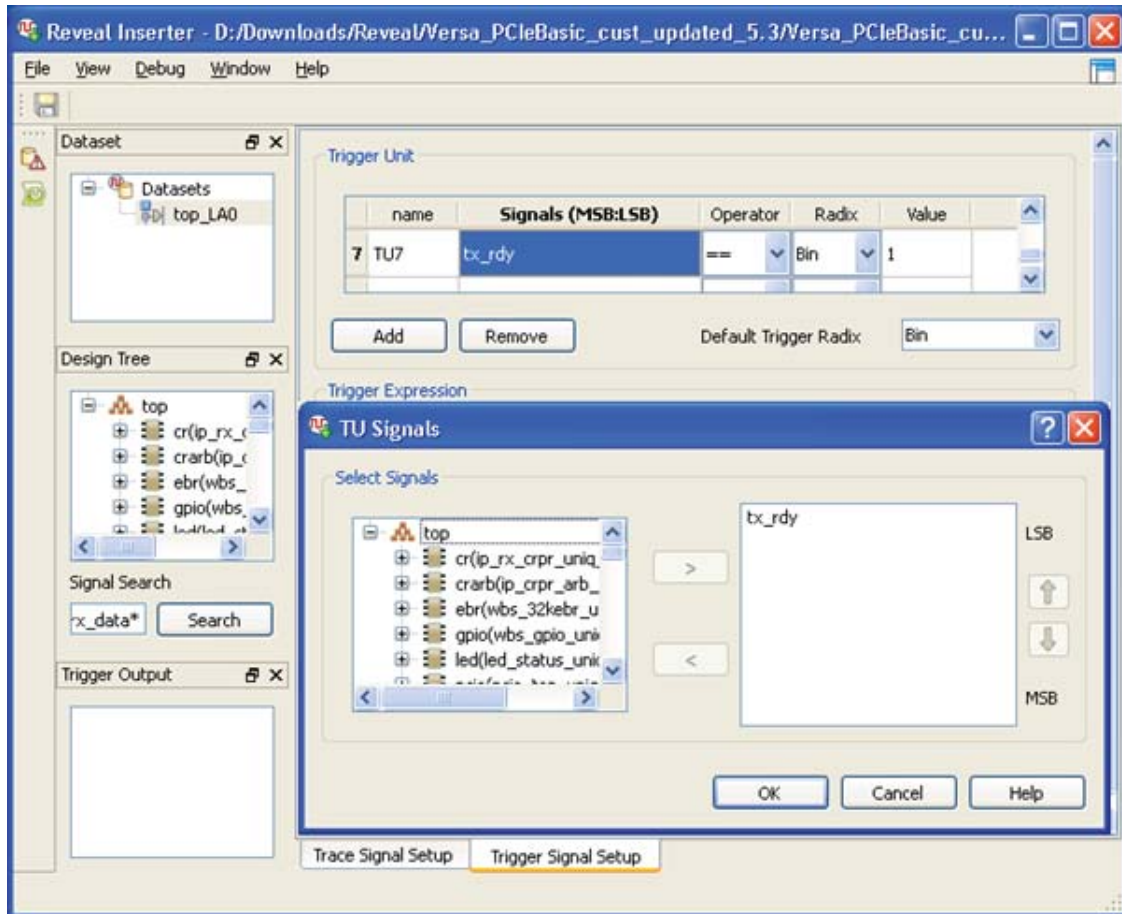


Trigger Unit 7 (TU7)

The TU7 port includes the TX side of the signals between the user application and the core. If the host is not receiving completions for the memory reads, probe this interface and verify whether or not the completion packet has been generated by your user application and presented to the core.

If there is no sufficient credits at the host, then the core will deassert tx_rdy. This signal is also deasserted if the core is not ready to accept packets from the user application. If you do not see any packets coming into the host from the endpoint, set a trigger on this signal in Reveal Inserter to make sure it is not deasserted.

Figure 10. Add Signals in Trigger Unit 7 (TU7)



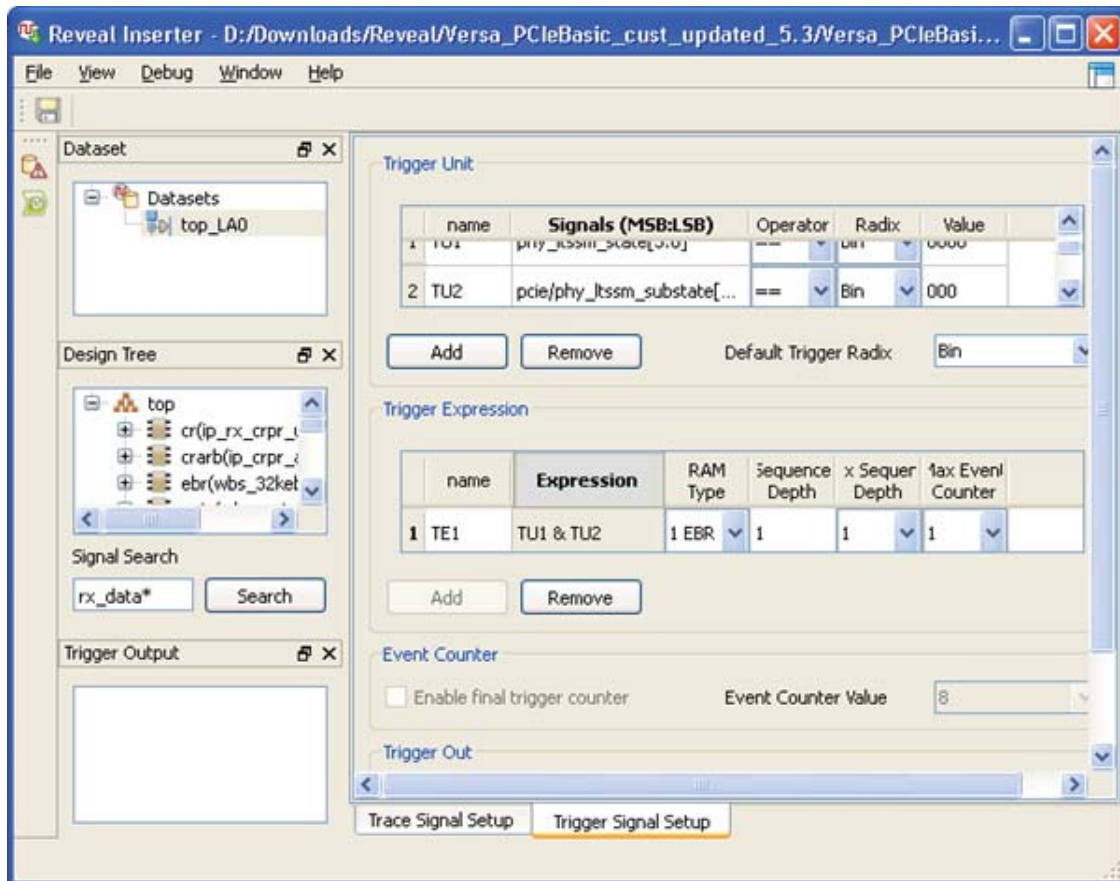
Step 4: Select Trigger Expression (TE1)

After selecting Trigger Units in the TU section, select Trigger Expression (TE1). In the Expression box, enter the names of trigger units and the operators you want to use to connect them.

You can also use the following operators to connect the trigger units in Trigger Expression:

- & (AND) – Combines the trigger units using an & operator in the trigger expression.
- | (OR) – Combines the trigger units using an OR operator in the trigger expression.
- ^ (X-OR) – Combines the trigger units using XOR operator in the trigger expression.
- ! (NOT) – Combines the trigger units using NOT operator in the trigger expression.
- THEN – Creates a sequence of wait conditions. For example, TU1 THEN TU2. This means “Wait for TU1 to be true,” then “Wait for TU2 to be true.”



Figure 11. Setting the Trigger Expression (TE1)



To simplify, we have selected TE1= (TU1 & TU2) as example. You may, however, select the trigger expression according to the requirement such as TE1 = (TU1 THEN TU2) for example.

Note: If you update TE1, rerun the implementation flow.

Step 5: Make the Reveal file active in your design

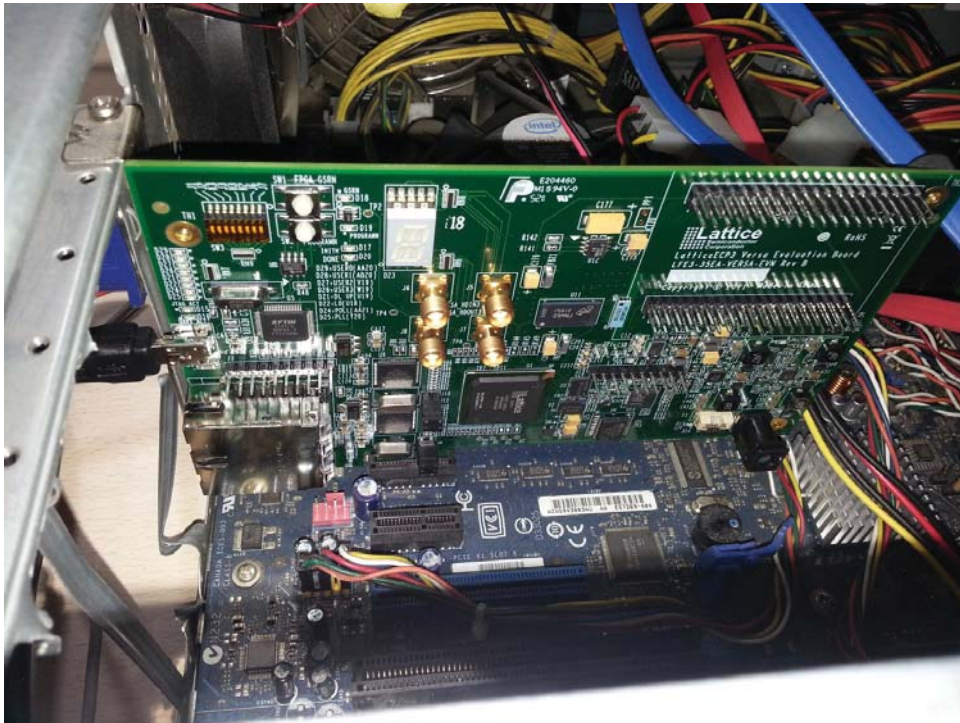
1. Once the trigger units such as TUs and TE are defined in the Reveal file (.rvl), click the **Design Rule Check** button .
2. Click the **Insert Debug** button  to select the core and activate the Reveal Inserter file in your design project.
3. Once the Reveal Inserter file is active in the design project, run the implementation flow to generate the bitstream.

Step 6: Install and set up hardware

1. Insert your LatticeECP3 Versa Evaluation Board into the x1 slot of your PC (as shown in Figure 12).


Do not power the board externally. The LatticeECP3 Versa board acquires power from the PCI Express slot once the board is plugged into the PCIe x1 slot and the PC is ON.

Figure 12. LatticeECP3 Versa Hardware Setup



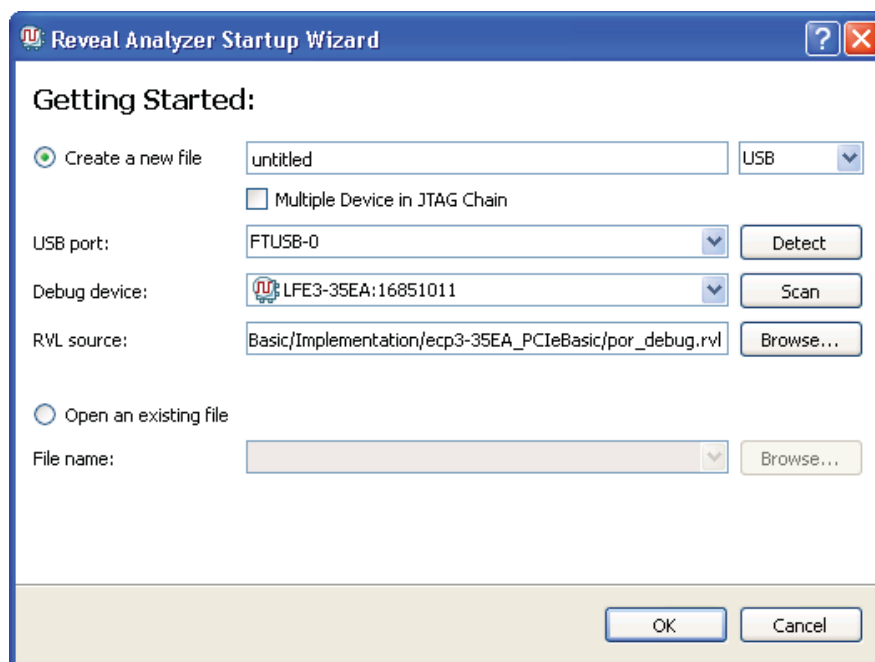
2. Once the ECP3 Versa Kit is interfaced in the PCIe Slot and the PC is ON, you can program the SPI Flash with the PCIe Basic Demo bitstream.
3. After the SPI Flash is programmed successfully, restart the PC.

Step 7: Run Reveal Analyzer

1. Open the Diamond software and click the **Reveal Analyzer** button  to start the Reveal Analyzer tool. Figure 13 shows the interface when you open the Reveal Analyzer tool for the first time.
2. Click **Detect** to detect the USB Port.
3. Click **Scan** to scan the debug device and make sure that the Reveal inserter file is present with the bit-stream configured in the device.
4. Click the **Browse** button and select **por_debug.rvl** as source file.

Note: If the Reveal Inserter source file (such as por_debug.rvl) does not match the file inserted in your design and configured with the FPGA, Reveal Analyzer shows a Signature Mismatch Error and the session is aborted.

Figure 13. Reveal Analyzer Setup



1. Click **OK** to invoke the waveform window.
2. Once the Reveal Analyzer is open, the captured waveforms are automatically displayed as shown in Figure 14.

Detecting Active State

Figure 14 shows the LTSSM state at DETECT_ACTIVE1 on Trigger Unit. You may, however, set a different LTSSM state to trigger according to your requirements to capture the status of LTSSM state.

Figure 14. Triggered LTSSM State - Detect_Active 1

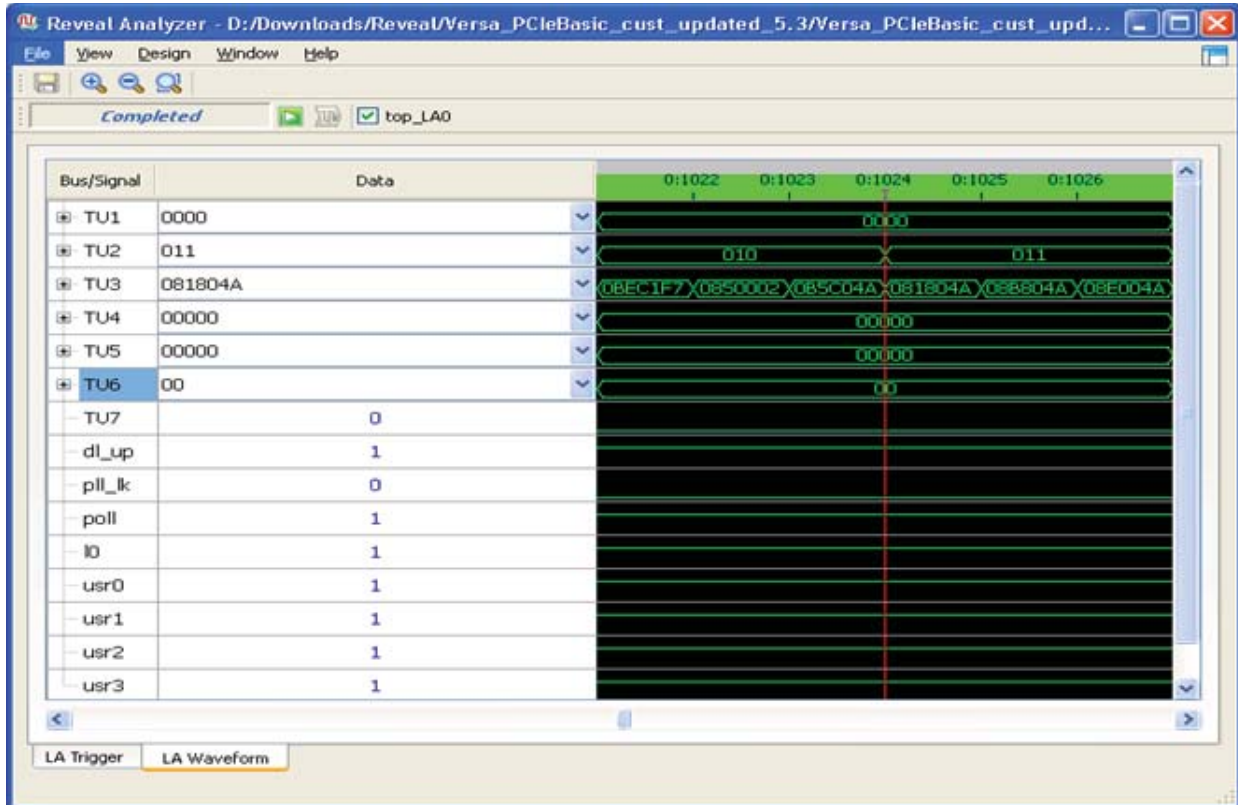


Figure 15 shows the capture of phy_ltssm[3:0] and phy_ltssm_substate[2:0] state when Reveal is triggered on POLL_CONFIG (phy_ltssm_state [3:0] =0001 & phy_ltssm_substate[2:0]=011).

Figure 15. Trigger LTSSM State – POLL_CONFIG

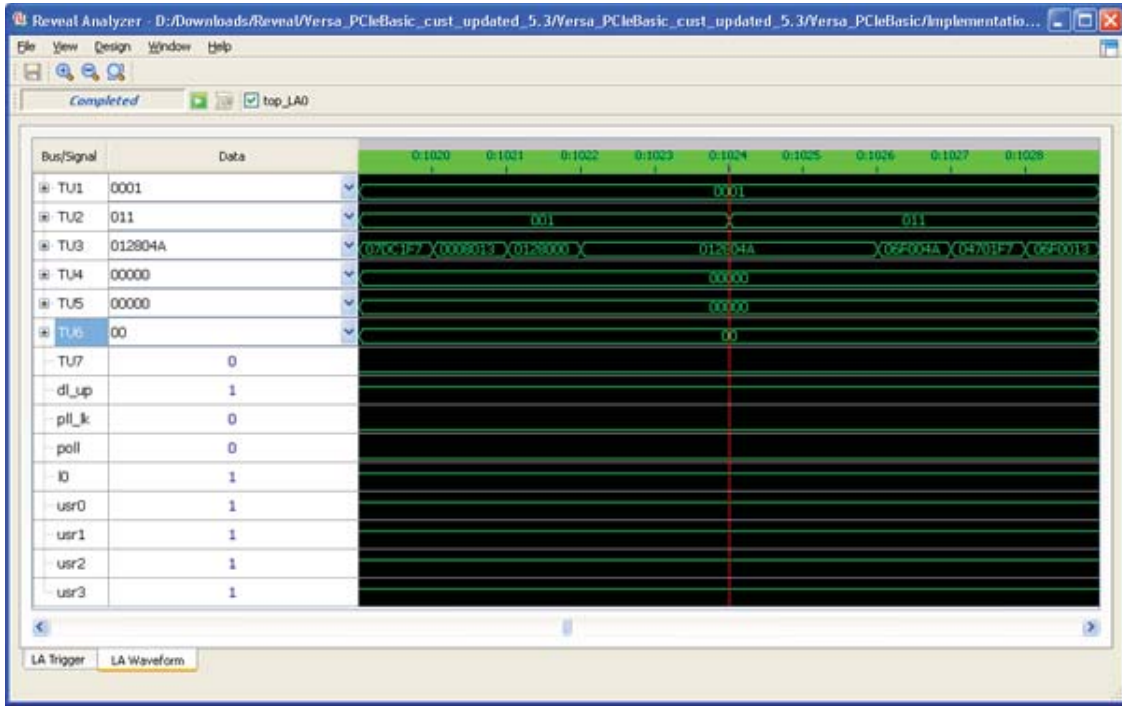


Figure 16 shows the LTSSM State at CONFIG_COMPLETE on Trigger Unit.

Figure 16. Trigger LTSSM State – CONFIG_COMPLETE

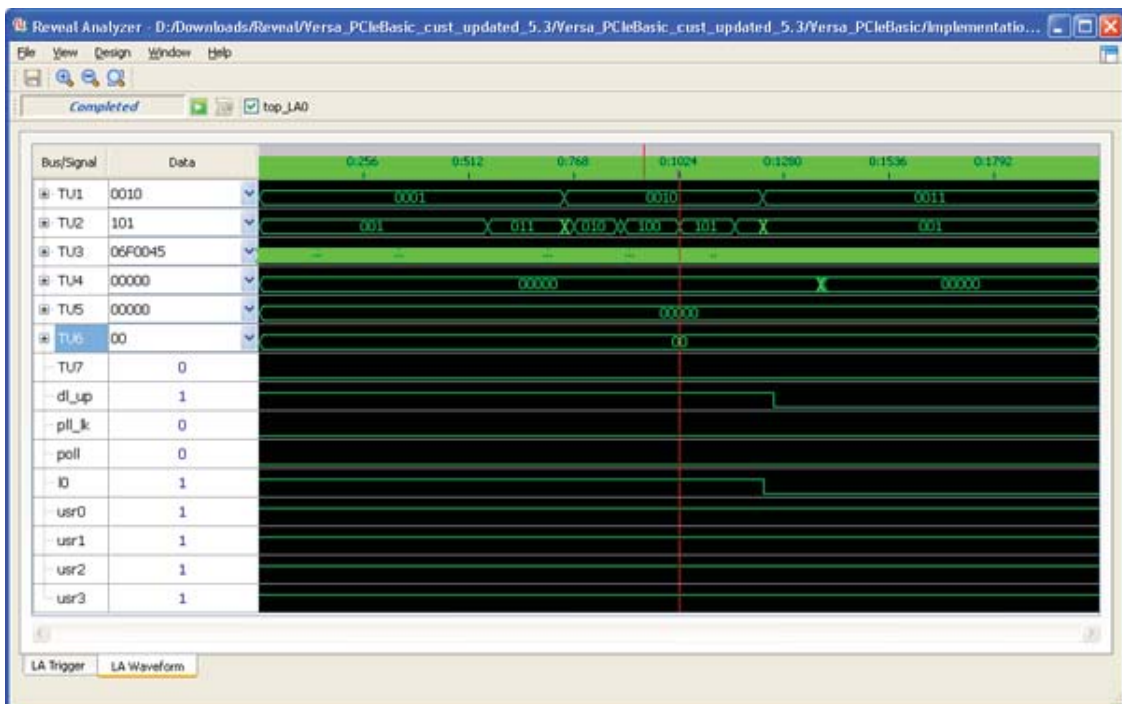
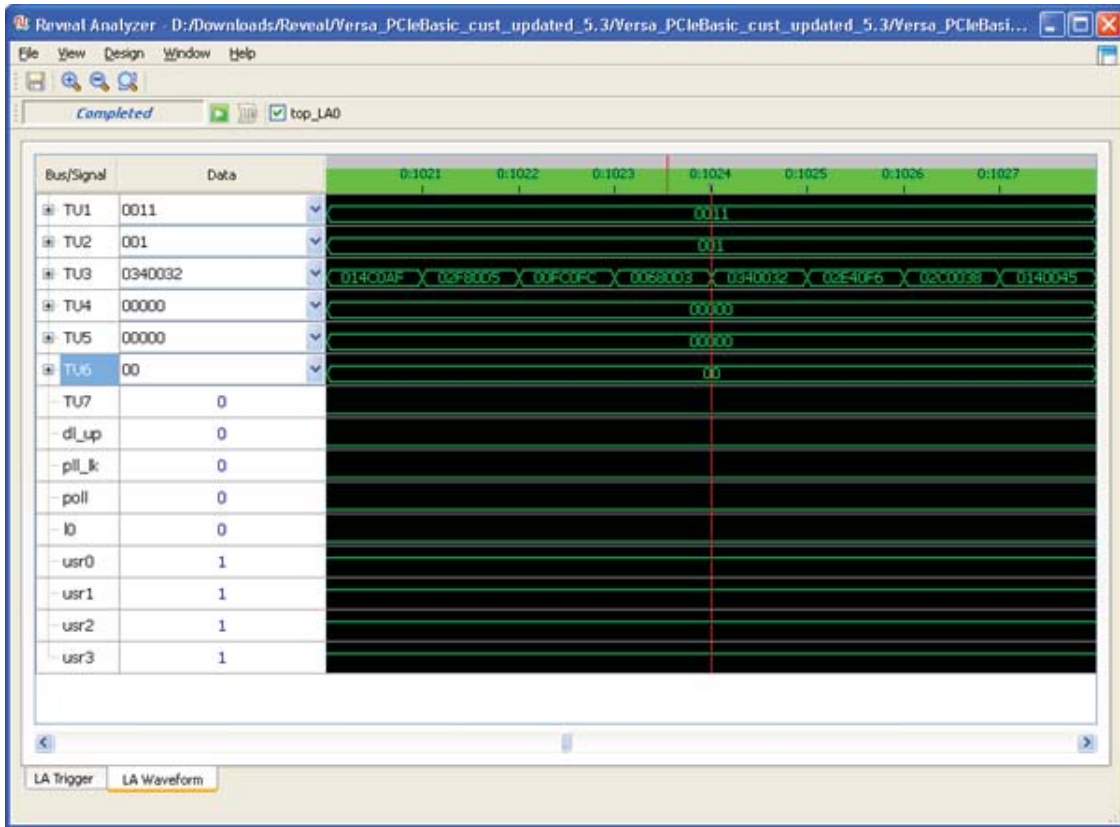


Figure 17 shows the LTSSM State at L0_L0 on Trigger Unit.

Figure 17. Trigger LTSSM State - L0_L0



When the `phy_ltssm_state[3:0] = 0011` and the `phy_ltssm_substate[2:0] = 001`, this indicates that the core has completed link training and is now in L0 state. While you are debugging issues related to link training, it would be useful to first check the status of these signals in Reveal Analyzer.

After the core completes link training, you can trace the incoming and outgoing TLP Packets at the Transaction Layer of Lattice PCIe Endpoint IP Core by using the normal Reveal flow. Disable the POR feature, uncheck the Trigger Enable check box as shown in Figure 2.

Note: Set the Trigger Expression with respective TUs to capture. For example, to capture the status of `rx_bar_hit[6:0]` signal, set `TE1 = TU6` and so forth. Once you update the TE, rerun the Implementation flow.

Figure 18 shows 'FB' is detected on rxp_data which indicates the start of frame.

Figure 18. 'FB' (start of frame) Detected on rxp_data

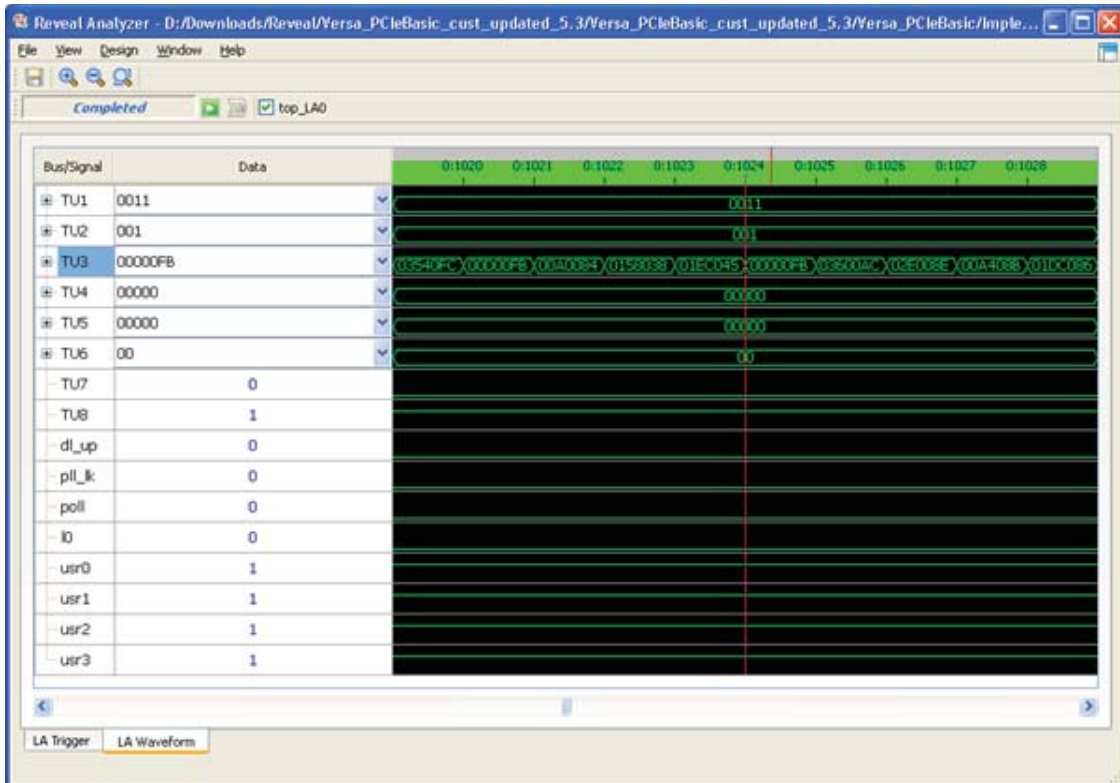


Figure 19 shows the status of rx_bar_hit[6:0] which is active high BAR indicator for the current TLP. If this bit is high, the current TLP on the receive interface is in the address range of the specified BAR.

Figure 19. Status of rx_bar_hit [6:0]

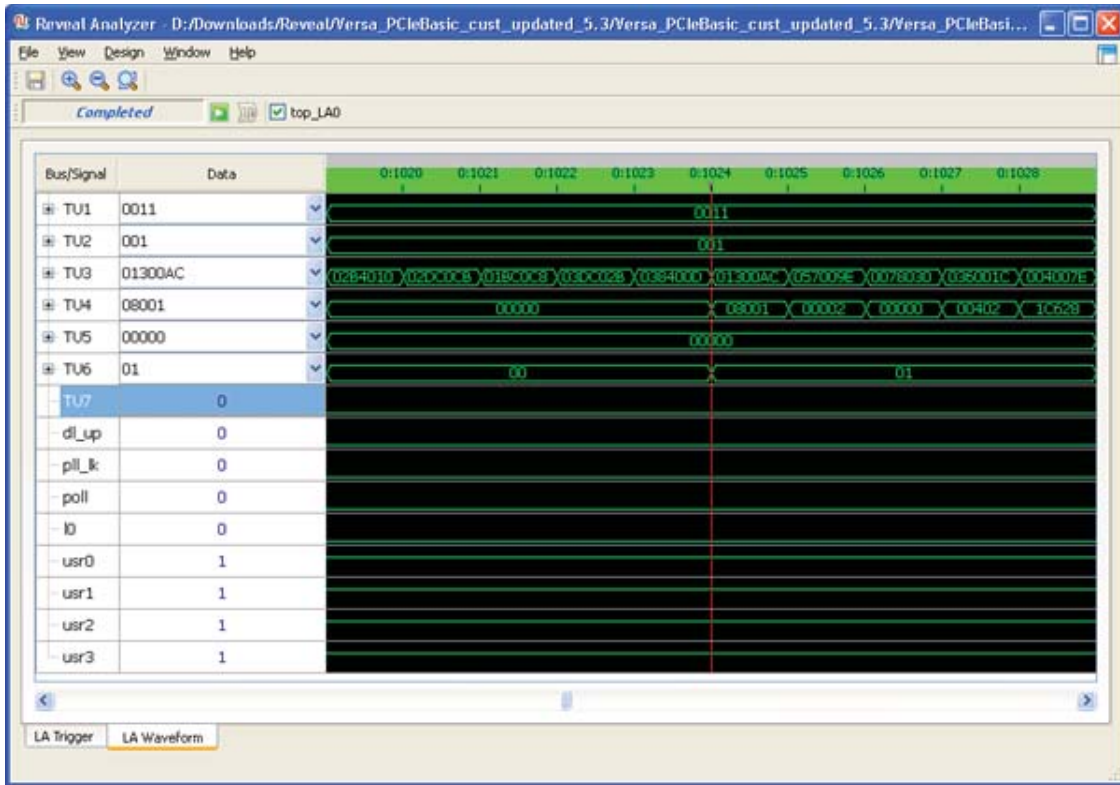


Figure 20 shows the status of tx_rdy signal which is active high signal used to indicate that there are sufficient credits available at the host before Endpoint begin transmitting the data. If there are no sufficient credits at the host, the core de-asserts tx_rdy signal. This signal is also de-asserted if the core is not ready to accept packets from the user application.

If you do not see any packets coming in to the host from the Endpoint, set a trigger on this signal and make sure it is not de-asserted indefinitely.

Figure 20. Status of tx_rdy Signal

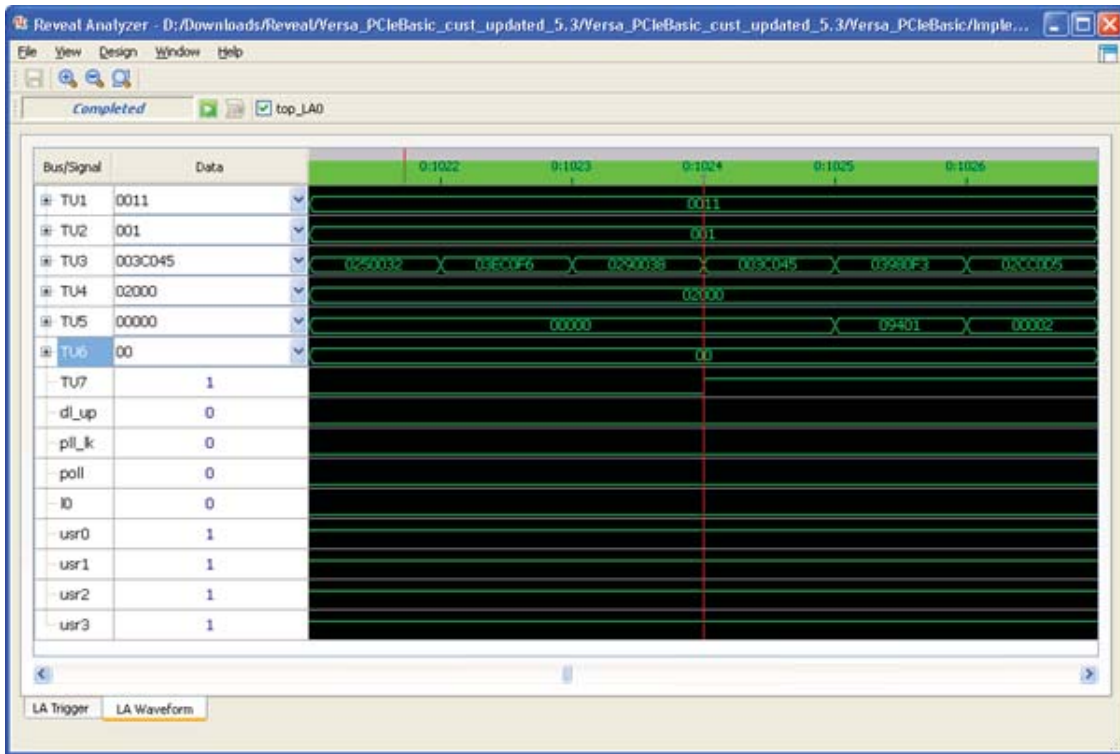
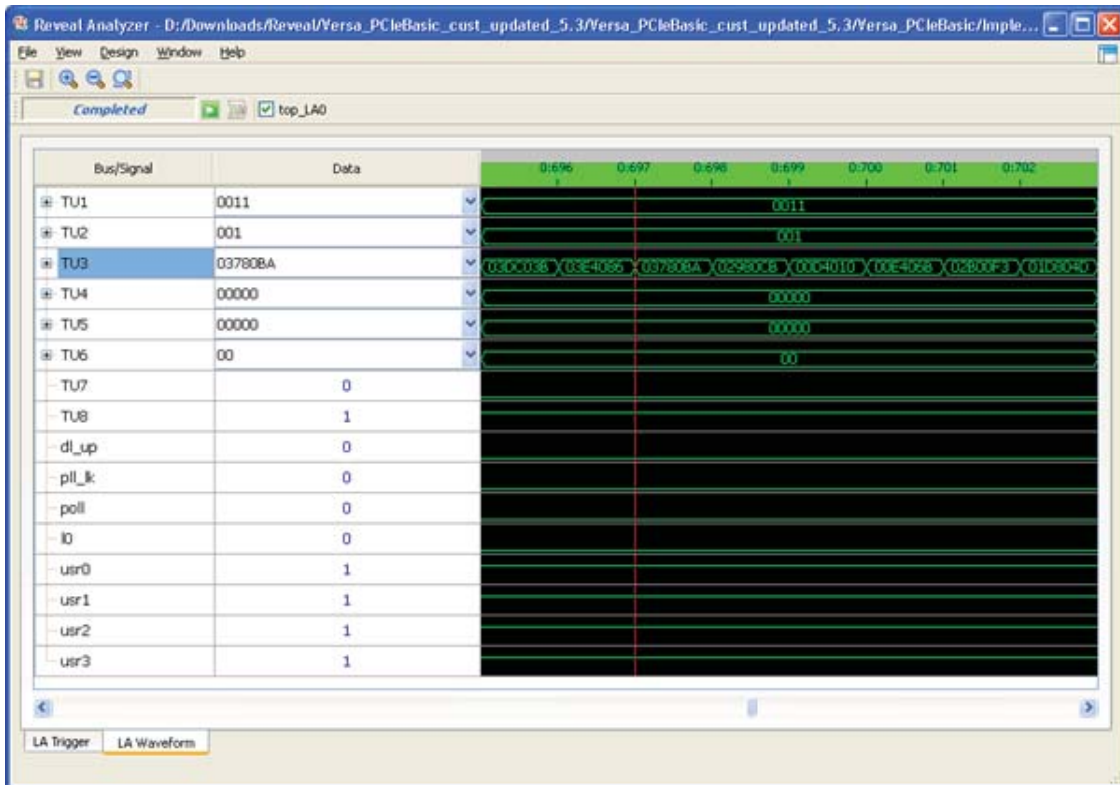


Figure 21 shows the status of rxp_valid signal which is active high signal used to indicate that the valid TLP Packet is coming from Host to Endpoint on rxp_data.

Figure 21. Received Data from Host when rxp_valid is Asserted



Conclusion

Observing circuit activities in the Lattice PCI Express (PCIe) IP core is performed by going through packet and signal analysis. To observe link training or the transmission and receiving of packets, there are recommended procedures to capture the relevant signals. The Reveal Inserter and the Reveal Analyzer tools may then be used to analyze the captured waveforms and perform root cause analysis, if needed.

References

- [Lattice PCIe Endpoint IP Core User's Guide](#)
- [PCIe Express Base Specification](#)
- [Reveal Users Guide](#)
- [Reveal Troubleshooting Guide](#)

Technical Support Assistance

e-mail: techsupport@latticesemi.com

Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
October 2013	01.0	Initial release.

Appendix A: Description of LTSSM States

This section describes various LTSSM states which occur during enumeration process of PCIe.

Table 1. LTSSM State Description

LTSSM State	LTSSM Sub state	Description
0000 - DETECT	000 - DET_WAIT	Wait in reset and set transmitter to electrical idle.
	001 - DET_QUIET	Wait for receive electrical idle to be broken or 12ms to expire.
	010 - DET_GODET1	Send command to SERDES to start receiver detection process, and wait 120ns.
	011 - DET_ACTIVE1	Wait for receiver detection process to be done and check status. If +ve then exit.
	100 - DET_WAIT12MS	Wait for 12ms if few receivers are detected but not all.
	101 - DET_GODET2	Again, send command to SERDES to start receiver detection process, and wait 120ns.
	110 - DET_ACTIVE2	Wait for receiver detection process to be done and check status if same lanes are detected. If +ve then exit
	111 - DET_EXIT	Exit from DETECT State and enter into POLLING State.
0001 - POLLING	000 - POL_WAIT	Wait for Main State to go to POLLING.
	001 - POL_ACTIVE	Send & receive TS1 ordered sets, send 1024 TS1s with PADs.
	010 - POL_COMPLIANCE	Send compliance pattern after 24ms in POL_ACTIVE and no electrical idle exit.
	011 - POL_CONFIG	Send & receive TS2 ordered sets, go to main state Detect if TS2s are not received after 48ms.
	100 - POL_EXIT	Exit from POLLING state enter into DETECT or CONFIGURATION State.
0010 - CONFIGURATION	000 - CFG_WAIT	Wait for Main State to go to CONFIGURATION.
	001 - CFG_LINK_WIDTH_ST	Wait until Link no. not PAD and lane no. PAD, exit to Detect after 24ms timeout.
	010 - CFG_LINK_WIDTH_ACC	Wait until Link no. match and lane no. non PAD, exit to Detect after 2ms timeout.
	011 - CFG_LANE_NUM_WAIT	Wait until Link no. match and lane no. match, exit to Detect after 2ms timeout.
	100 - CFG_LANE_NUM_ACC	Accept Lane numbers with TS2s Link no. match and lane no. match.
	101 - CFG_COMPLETE	Wait for 8 ts2s with Link no. match and lane no. match and 16ts2s sent.exit to Detect after 2ms timeout.
	110 - CFG_IDLE	Wait for 8 idles and 16 idles sent. Exit to Detect after 2ms timeout.
	111 - CFG_EXIT	Exit from CONFIGURATION State and enter into DETECT or L0 State.

LTSSM State	LTSSM Sub state	Description
0011 - L0	000 - L0_WAIT	Wait for Main State to go to L0.
	001 - L0_L0	Normal traffic mode, Exit to Recovery state if TS1/TS2 received or higher layer is instructs.
	010 - L0_L0RX	Lo is in receive mode and L0s_TX is transmitting.
	011 - L0_L0TX	L0 is in transmitting mode and L0s_RX is receiving.
	100 - L0_EIDLE_0	Send an electrical idle.
	101 - L0_EIDLE_1	Send an electrical idle and wait to receive electrical idle before exiting to L1 or L2.
	110 - L0_EXIT	Exit from L0 and enter into L0s/L1/L2.
0100 - L0s	000 - L0s_RX_WAIT	Wait for L0 State to go L0s_RX state.
	001 - L0s_RX_ENTRY	Wait for 20ns.
	010 - L0s_RX_IDLE	Wait for receiver to be out of electrical idle.
	011 - L0s_RX_FTS	Wait for SKP to exit to L0 or send FTS and exit to recover.
	100 - L0s_RX_EXIT	Exit from L0s and enter into L0/Recovery.
0101 - L1	000 - L1_WAIT	Wait for Main State to go to L1.
	001 - L1_ENTRY	Wait for 20ns.
	010 - L1_IDLE	Wait for receiver to be out of electrical idle and exit to Recovery.
	011 - L1_EXIT	Exit from L1 State and enter into Recovery State.
0110 - L2	000 - L2_WAIT	Wait for Main State to go to L2.
	001 - L2_IDLE	Wait for higher layer to instruct or receiver to be out of electrical idle and exit to Detect.
	011 - L2_EXIT	Exit from L2 State and enter into DETECT State.
0111 -RECOVERY	000 - RCVRY_WAIT	Wait for Main State to go to Recovery.
	001 - RCVRY_RCVRLK	Receive 8 TS1/TS2s and send 1024 TS1s, exit to detect after 24ms.
	010 - RCVRY_RCVRCFG	Receive 8TS2s and send 16 TS2s, exit to detect after 48ms.
	011 - RCVRY_IDLE	Receive 8 idles and send 16 idles, Exit to DETECT State after 2ms.
	100 - RCVRY_EXIT	Exit from Recovery State and enter into L0 or DETECT State.