Agilent
Protocol Exerciser for PCI Express

User's Guide

Agilent Technologies

Notices

Trademarks

Windows 2000®, Windows XP®, and Microsoft .NET Framework 1.1® are U.S. registered trademarks of Microsoft Corporation.

Installation Guides

You can find the installation guides for different components of the product on the product CD. Agilent recommends you to do not switch on the instrument before you have understood all the applicable installation instructions and have met all the installation prerequisites.

Where to find more information

You can find more information about Protocol Analyzer from the following link:

http://www.agilent.com/find/spt

For further assistance, you can search for a local contact on the following link:

http://www.agilent.com/find/assist

Edition

Fifth edition, January 2012

Warranty

The material contained in this document is provided "as is," and is subject to being changed, without notice, in future editions. Further, to the maximum extent permitted by applicable law, Agilent disclaims all warranties, either express or implied, with regard to this manual and any information contained herein, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. Agilent shall not be liable for errors or for incidental or consequential damages in connection with the furnishing, use, or performance of this document or of any information contained herein. Should Agilent and the user have a separate written agreement with warranty terms covering the material in this document that conflict with these terms, the warranty terms in the separate agreement shall control.

Technology Licenses

The hardware and/or software described in this document are furnished under a license and may be used or copied only in accordance with the terms of such license.

Restricted Rights Legend

If software is for use in the performance of a U.S. Government prime contract or subcontract, Software is delivered and licensed as "Commercial computer software" as defined in DFAR 252.227-7014 (June 1995), or as a "commercial item" as defined in FAR 2.101(a) or as "Restricted computer software" as defined in FAR 52.227-19 (June 1987) or any equivalent agency regulation or contract clause. Use, duplication or disclosure of Software is subject to Agilent Technologies' standard commercial license terms, and non-DOD Departments and Agencies of the U.S. Government will receive no greater than Restricted Rights as defined in FAR 52.227-19(c)(1-2) (June 1987). U.S. Government users will receive no greater than Limited Rights as defined in FAR 52.227-14 (June 1987) or DFAR 252.227-7015 (b)(2) (November 1995), as applicable in any technical data.

Safety Notices

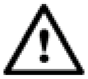
CAUTION

A **CAUTION** notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in damage to the product or loss of important data. Do not proceed beyond a **CAUTION** notice until the indicated conditions are fully understood and met.

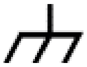
WARNING

A WARNING notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in personal injury or death. Do not proceed beyond a WARNING notice until the indicated conditions are fully understood and met.

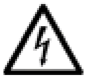
Safety Symbols on Instruments



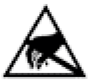
Indicates warning or caution. If you see this symbol on a product, you must refer to the manuals for specific Warning or Caution information to avoid personal injury or damage to the product.



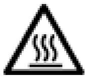
Frame or chassis ground terminal. Typically connects to the equipment's metal frame.



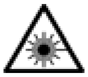
Indicates hazardous voltages and potential for electrical shock.



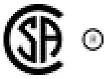
Indicates that antistatic precautions should be taken.



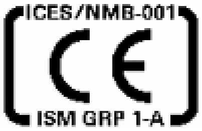
Indicates hot surface. Please do not touch.



Indicates laser radiation turned on.



CSA is the Canadian certification mark to demonstrate compliance with the Safety requirements.



CE compliance marking to the EU Safety and EMC Directives.

ISM GRP-1A classification according to the international EMC standard.

ICES/NMB-001 compliance marking to the Canadian EMC standard.

Safety Summary

General Safety Precautions

The following general safety precautions must be observed during all phases of operation of this instrument. Failure to comply with these precautions or with specific warnings elsewhere in this manual violates safety standards of design, manufacture, and intended use of the instrument.

Agilent Technologies Inc. assumes no liability for the customer's failure to comply with these requirements.

Before operation, review the instrument and manual for safety markings and instructions. You must follow these to ensure safe operation and to maintain the instrument in safe condition.

General

This product is a Safety Class 1 instrument (provided with a protective earth terminal). The protective features of this product may be impaired if it is used in a manner not specified in the operation instructions.

All Light Emitting Diodes (LEDs) used in this product are Class 1 LEDs as per IEC 60825-1.

Environment Conditions

This instrument is intended for indoor use in an installation category II, pollution degree 2 environment. It is designed to operate at a maximum relative humidity of 95% and at altitudes of up to 2000 meters.

Refer to the specifications tables for the ac mains voltage requirements and ambient operating temperature range.

Before Applying Power

Verify that all safety precautions are taken. The power cable inlet of the instrument serves as a device to disconnect from the mains in case of hazard. The instrument must be positioned so that the operator can easily access the power cable inlet. When the instrument is rack mounted the rack must be provided with an easily accessible mains switch.

Ground the Instrument

To minimize shock hazard, the instrument chassis and cover must be connected to an electrical protective earth ground. The instrument must be connected to the ac power mains through a grounded power cable, with the ground wire firmly connected to an electrical ground (safety ground) at the power outlet. Any interruption of the protective (grounding) conductor or disconnection of the protective earth terminal will cause a potential shock hazard that could result in personal injury.

Do Not Operate in an Explosive Atmosphere

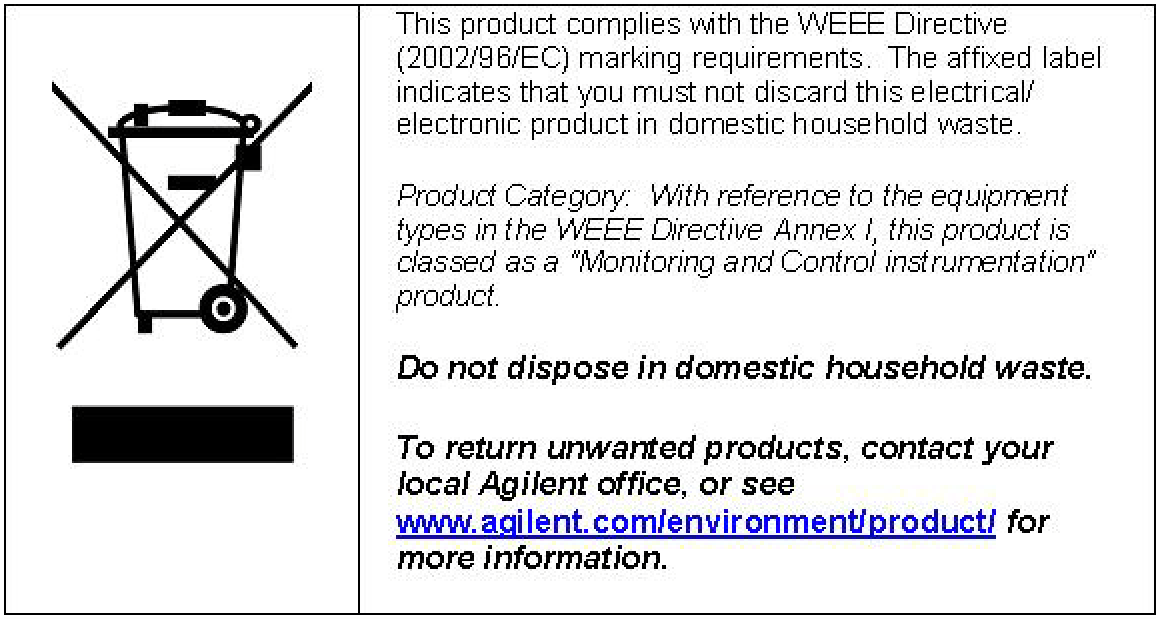
Do not operate the instrument in the presence of flammable gases or fumes.

Do Not Remove the Instrument Cover

Operating personnel must not remove instrument covers. Component replacement and internal adjustments must be made only by qualified personnel.

Instruments that appear damaged or defective should be made inoperative and secured against unintended operation until they can be repaired by qualified service personnel.

Environmental Information



This product complies with the WEEE Directive (2002/96/EC) marking requirements. The affixed label indicates that you must not discard this electrical/ electronic product in domestic household waste.

Product Category: With reference to the equipment types in the WEEE Directive Annex I, this product is classed as a "Monitoring and Control instrumentation" product.

Do not dispose in domestic household waste.

To return unwanted products, contact your local Agilent office, or see www.agilent.com/environment/product/ for more information.

Document History

Agilent Technologies, Inc. can issue revisions between the product releases to reflect the latest and correct information in the guide. Agilent Technologies, Inc. also reserves its right to not issue a new edition of the guide for every system release.

Manual Name: Agilent Protocol Exerciser for PCI Express - User’s Guide

The edition number of the guide, publishing time of the guide, and applicable release number of the product are given in the following table.

Edition	Published	Applicable Release
1.4	January, 2012	8.5

Contents

What's New in this Release

1 Agilent Protocol Exerciser for PCI Express - Introduction

What is Protocol Exerciser for PCI Express 16

Emulation Modes 16

Features of Protocol Exerciser 19

Protocol Exerciser for PCI Express Modes (As Exerciser and as LTSSM) 20

When to use Protocol Exerciser for PCI Express 21

Functions, Hardware Channels, Virtual Hierarchies, and Virtual Channels 22

As a Non IOV PCIe component 22

As a SRIOV capable component 23

As a MRIOV capable component 23

Protocol Exerciser for PCI Express Components 25

Hardware Components 25

Software Components 26

2 Getting Started with Protocol Exerciser

Creating a Protocol Exerciser Session 30

Protocol Exerciser Online mode 30

Protocol Exerciser Offline mode 31

Creating/Using a Protocol Exerciser Session 31

Programming Protocol Exerciser Card as Gen3 Exerciser 35

Starting and Exiting the Protocol Exerciser GUI 36

Protocol Exerciser GUI Components 38

Viewing and Updating the Protocol Exerciser Hardware Status 40

Link Status 41

Flow Control 43

Function Status 44

Managing Floating Windows 46

Configuring Data Display Preferences Settings 48

3 Configuring Protocol Exerciser

Creating and Managing Protocol Exerciser Setup Files 50

Configuring a PCIe Link Settings 51

Types of link 51

Configuring Lane Settings 54

Configuring Equalization Procedure Settings 56

Transmitter Based Equalization (De-emphasis) for 5.0 GT/s speed 56

Equalization for 8.0 GT/s speed 57

Autonomous equalization 57

Software based equalization 58

Configuring Equalization settings 59

Configuring Transceiver Settings 61

Configuring Pattern Matchers 64

Creating a pattern matcher 64

Configuring an action when a pattern match is found 66

Configuring External Triggers Settings 69

Configuring Power Management Settings 71

Configuring L0s state 71

Configuring L1 state 72

4 Initializing and Managing a PCIe Link

Initializing and training a PCIe Link between Protocol Exerciser and DUT 76

Training a PCIe Link 77

Verifying PCIe Link Training 77

Initializing Flow Control 79

Defining advertised header and data credit limits 79

Initializing flow Control for a non IOV / SRIOV capable PCIe component 79

Initializing flow control for an MRIOV capable component 80

Viewing the Flow Control Initialization status 81

Retraining a PCIe Link between Protocol Exerciser and DUT 84

Initiating Link Speed Change on a Configured Link 85

5 Testing DUT’s LTSSM Functions

LTSSM 88

LTSSM Functions 88

LTSSM States 88

LTSSM Tester - Introduction 90

Protocol Exerciser as a USC and a DSC in context of LTSSM testing 90

LTSSM Tester Features 91

When to Use LTSSM Tester 92

LTSSM Supported States 92

How LTSSM Tester Works 94

Getting Started with LTSSM Tester 97

Accessing LTSSM Tester 97

Prerequisites for LTSSM Testing 99

Testing Link Initialization, Training, and Management 100

LTSSM Tests 100

Testing Link Training 102

Testing Link Speed Changes 107

Testing Link Retraining and Recovery 113

Testing LTSSM State Transitions 114

6 Defining Configuration Space of Exerciser

Configuration Space - Overview 116

7 Configuring and Mapping Decoder and Data Memory

Configuring Decoder Settings 118

VF BARs provided for an MRIOV capable Exerciser 119

Configuring and Using Data Memory of Exerciser 122

Writing to and reading from data memory of Exerciser 123

Importing and Exporting Data Memory contents 124

Initializing and reloading data memory 125

Comparing Actual Data with Expected Data in Memory 126

8 Providing Stimulus to DUT

Stimulus Traffic - Overview 130

Stimulus traffic flow as a non IOV PCIe component 130

Stimulus traffic flow as a SRIOV capable PCIe component 131

Stimulus traffic flow as an MRIOV capable PCIe component 133

Defining Stimulus Traffic 136

Functions 136

PCIe Traffic Templates 138

Defining the PCIe packets to be sent as stimulus 138

Defining the behavior of the PCIe packets to be sent as stimulus 139

Changing the default settings of PCIe packets 141

Starting the Stimulus and Viewing Results 143

Viewing the stimulus results 144

Stimulus - Example 149

Sending a Single Packet as Stimulus and Viewing Response 152

Sending Completions as Response to DUT 157

Completion Queues 157

Adding Completion behavior records 158

Changing the default behavior for completions 158

Sending Completions 159

9 Performing Protocol Checks and Inserting Errors

Testing DUT Under Error Conditions 162

Supported Error Insertions 162

Configuring Error Insertion Settings for Exerciser 163

Testing DUT Under Error Conditions - Example 164

Performing Protocol Checks 169

Generating a Trigger out on encountering a protocol error 169

Enabling a protocol rule 169

Supported Protocol Checks 171

Testing End to End Cyclic Redundancy Checks 174

Checking ECRC errors in Received TLPs 174

Inserting ECRC Errors in Transmitted TLPs 175

10 Protocol Test Card

Protocol Test Card 180

Accessing Protocol Test Card 181

Important Points about Protocol Test Card 186

Protocol Test Card Example 187

To Check that the receiver ignores the reserved fields of the received DLLPs, and To check that DUT retransmits a transaction for which NAK has been issued 187

11 Protocol Exerciser GUI Reference

Dialog Boxes 190

Edit Packet Behavior dialog box 190

Edit Packet dialog box 193

Export Memory to File dialog box 196

Find in Memory dialog box 196

Goto Address dialog box 197

Import Memory from File dialog box 197

Initialize Memory dialog box 198

Open dialog box 199

Port Selection dialog box 200

Preferences dialog box 200

Save dialog box 202

- Select type of connection dialog box 203
- Software Package Management dialog box 204

Menus 206

- Action Menu 206
- Edit Menu 207
- File Menu 207
- Help Menu 208
- View Menu 208

Windows 209

- Config Space function 209
- Data Memory function - Edit Memory tab 210
- Data Memory function - Memory Compare tab 211
- Decoder function 212
- Error Insertion function - ACK/NAK Errors tab 214
- Error Insertion function - DLLP tab 215
- Error Insertion function - Request and Completion Errors tab 216
- General Settings Function Equalization Settings tab 219
- General Settings Function Lane Settings tab 221
- General Settings Function Link Settings tab 222
- General Settings function - Pattern Matcher tab 224
- General Settings function - Power Management tab 225
- General Settings Function Symbol Settings tab 227
- General Settings Function Transceiver Settings tab 227
- General Settings Function - Trigger Out tab 229
- LTSSM Tests 230
- Protocol Checker function 231
- Traffic Setup Function - Completion Behaviors 232
- Traffic Setup Function - Block Transfers 233
- Traffic Setup Function - Single Packet 235
- Virtual Channel function 237

A Updating the Firmware

Using the Firmware Update Tool 242

Index

B PTC Tests and Assertions

Tests and Assertions 249

What's New in this Release

The following are the new/key features of this release of the Agilent Protocol Exerciser for PCI Express Gen3 product.

- New LTSSM tests have been added. Refer to the topic LTSSM Tests on page 100 to know more.
- You can now enable/disable the data memory compare setting for comparison of payload of incoming completions separately for each physical and virtual function of Exerciser. Refer to the topic Comparing Actual Data with Expected Data in Memory to know more.Refer to the topic Comparing Actual Data with Expected Data in Memory on page 126 to know more.
- The Pattern Matcher feature is provided that you can use to compare the TLP header and the first 4 DWs of the TLP payload with the contents of the incoming TLPs. Protocol Exerciser provides four pattern matchers. Refer to the topic Configuring Pattern Matchers on page 64 to know more.
- New options have been added in the Equalization Settings tab to allow you to:
  - control the number of times the DUT can reject preset/coefficient values (requested by Protocol Exerciser).
  - instruct the DUT to use only the presets from the incoming TS OS in Phase2/Phase3 of Equalization.

  Refer to the topic General Settings Function Equalization Settings tab on page 219 tab to know about these options.
- You can now set the transceiver settings separately for Gen1, Gen2, and Gen3 speeds. These settings are for the Transmitter (Tx) of Protocol Exerciser. Refer to the topic General Settings Function Transceiver Settings tab on page 227 to know more.

1
Agilent Protocol Exerciser for PCI Express - Introduction

What is Protocol Exerciser for PCI Express 16

Protocol Exerciser for PCI Express Modes (As Exerciser and as LTSSM) 20

When to use Protocol Exerciser for PCI Express 21

Functions, Hardware Channels, Virtual Hierarchies, and Virtual Channels 22

Protocol Exerciser for PCI Express Components 25

This chapter introduces you to Protocol Exerciser for PCI Express and provides information on its emulation modes, features, uses, and its hardware and software components.

What is Protocol Exerciser for PCI Express

Agilent Protocol Exerciser for PCI Express (PCIe) provides a test and debug solution for PCI Express to test the next generation of PCI Express technology. Agilent Protocol Exerciser for PCIe is an advanced traffic generator that you can use to send and respond to the TLP and DLLP packets to stimulate PCIe devices and systems.

It can emulate a PCIe device or topology with or without MRIOV / SRIOV capabilities to test MRIOV / SRIOV capable/non IOV PCIe components.

Protocol Exerciser can function as an Exerciser and a Link Training and Status State Machine (LTSSM) with a PCIe system or a PCIe device under test. This release of Protocol Exerciser supports both the LTSSM and Exerciser modes.

The following figure illustrates its coverage in terms of the OSI standard model.

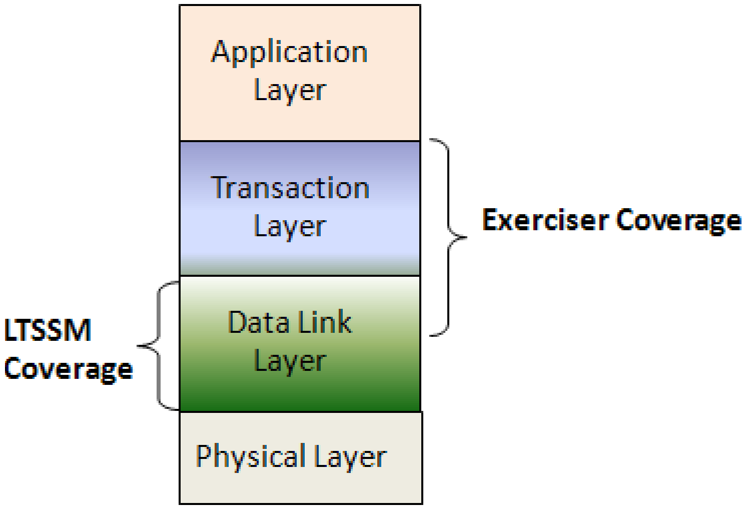


Figure 1 Protocol Exerciser coverage

Refer to the topic Protocol Exerciser for PCI Express Modes (As Exerciser and as LTSSM) on page 20 to know more about its functions as an Exerciser and an LTSSM.

Emulation Modes

The Agilent Protocol Exerciser for PCI-Express supports the following emulation modes:

- **a non IOV PCIe component**
    - end point (DSC) to exercise the PCIe System Under Test (SUT) acting as a root complex.
    - root complex (USC) to exercise a PCIe end point.
- **an MRA (Multi Root Aware) PCIe component**
    - end point to exercise an MRA PCIe root complex. As an MRA PCIe end point, Protocol Exerciser can be used by three virtual hierarchy domains (five virtual hierarchies if the additional physical functions license is obtained).
    - root complex to exercise an MRA PCIe end point. As an MRA PCIe root complex, Protocol Exerciser can support three virtual hierarchy domains (five virtual hierarchies if the additional physical functions license is obtained).
- **a SRIOV capable PCIe component**
    - end point to exercise a SRIOV PCIe root complex.
    - root complex to exercise a SRIOV PCIe end point.

As an MRA PCIe component, Protocol Exerciser supports the multi root capabilities as defined in the MRIOV specifications. As a SRIOV capable component, Protocol Exerciser is compliant with the SRIOV specifications Rev 1.1.

For a non IOV testing scenario, you can create a PCIe link between the Protocol Exerciser and DUT/SUT to send and respond to PCIe traffic. For an MRIOV capable testing scenario, you can create an MR enabled link between the Protocol Exerciser and MRIOV capable DUT/SUT as per the MRIOV specifications.

In all these modes, a controlling system is needed that hosts the Protocol Exerciser software to manage and control the Exerciser card.

As an endpoint

When emulating an endpoint, the Agilent Gen3 exerciser card is plugged into the motherboard, as a normal PCIe device. You can use multiple exerciser cards to load and stress a server (system under test). You can connect up to four Exerciser cards to one system controller through USB.

As a root complex

When emulating a root complex, the Agilent Gen3 exerciser card communicates to the DUT through the backplane board. When you use the backplane, Protocol Exerciser communicates to the DUT through the bottom connectors.

Another way is to connect the DUT through the top connector daughter board. In this mode, the link width would be limited. However, this would be the preferred test method of compliance test, in order to limit the link width.

The following figure demonstrates a sample setup of a Protocol Exerciser card emulating an endpoint and a root complex.

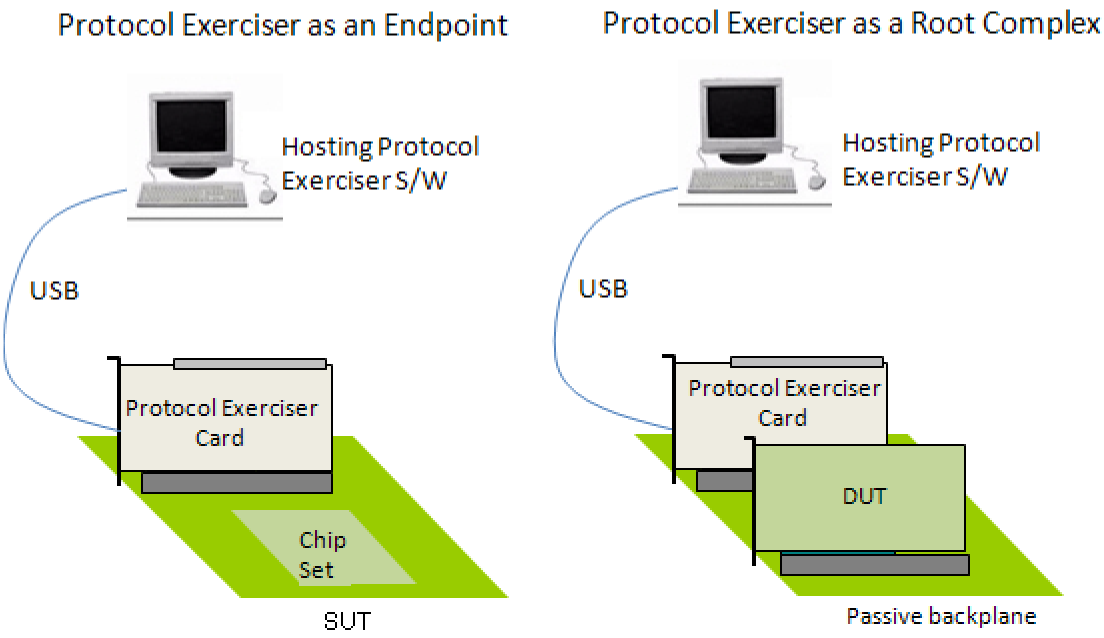

Protocol Exerciser as an Endpoint
Hosting Protocol Exerciser S/W
USB
Protocol Exerciser Card
Chip Set
SUT
Protocol Exerciser as a Root Complex
Hosting Protocol Exerciser S/W
USB
Protocol Exerciser Card
DUT
Passive backplane

Figure 2 Protocol Exerciser Card Sample Setups

NOTE

Refer to the Installation guide for the Protocol Exerciser for PCIe Express to know how to set up the Protocol Exerciser hardware to emulate as a root complex or as a PCIe endpoint.

Features of Protocol Exerciser

A brief list of the features of Protocol Exerciser is given below:

- Performs link negotiation, initialization and training, data link layer functions, and handles incoming requests and completions as per the:
  - PCI Express 3.0 Base specification for testing a non IOV PCIe component.
  - MRIOV specifications Revision 1.0 for testing an MRIOV capable PCIe component.
  - SRIOV specifications Revision 1.1 for testing an SRIOV capable PCIe component.
- Addresses the need for effective link negotiation testing by providing the LTSSM Tester tool:
  - Thorough link testing with LTSSM Tester.
  - Exercising DUT with the fully featured x16 Protocol Exerciser.
- Generates training sequences at 8.0 GT/s, 5 GT/s, and 2.5 GT/s speed on all lanes for link width up to x16, which allows effective link negotiation testing.
- Offers a fully featured and freely programmable stimulus solution for link width up to x16 that enables root complex emulation, endpoint emulation, and add-in card testing at link widths x1 to x16.
- Enables you to insert errors in request and completion packets to be sent as stimulus to DUT to test the behavior of DUT in response to these errors.
- Enables you to perform protocol checks by selecting the PCIe protocol rules that you want to observe for violation.
- Enables you to set voltage settings specific to hardware to get the best equalization at Gen3 speed.
- Enables you to set the TC to VC mapping for each hardware channel to check for effective stimulus transmission.

Protocol Exerciser for PCI Express Modes (As Exerciser and as LTSSM)

Protocol Exerciser can function as an Exerciser as well as an LTSSM.

As an LTSSM, you can use it for thorough link testing with DUT and exchanging training sequences to test and validate LTSSM state transitions. To know more about its LTSSM mode and how to use it for LTSSM testing, refer to the chapter Testing Link Initialization, Training, and Management on page 100.

As an Exerciser, it exercises DUT with the configured stimulus traffic. The traffic sent to DUT depends on the protocol that you selected while creating the link between the Exerciser and DUT. You can select PCIe, SRIOV, or MRIOV protocol while creating the link.

Exerciser can send TLP requests as well as completions to a DUT. It allows you to configure a block of TLP packets for stimulus and define the behavior of these packets and also the behavior of the completions. To know more about its Exerciser mode and how to use it to stimulate and test DUT by generating appropriate stimulus traffic, refer to the topic Defining Stimulus Traffic on page 136.

When to use Protocol Exerciser for PCI Express

Usage as a Protocol Exerciser

By emulating a PCIe component (with or without MRIOV capabilities), Protocol Exerciser acts as an ideal link partner by sending appropriate I/O traffic to stimulate the DUT. This way, it can help you test DUT under various conditions and scenarios without influencing the performance parameters of DUT. You can use it to send a block of TLP requests of 32 or 64 bit Memory, I/O, Configuration, or Message types as stimulus to DUT. You can also use it to send completion packets in response to DUT's requests.

To know more about its usage as an Exerciser, refer to the chapter **Providing Stimulus To DUT**.

Usage as an LTSSM

One of the key challenges with PCI Express is the validation testing for the LTSSM functions and thorough link testing supporting the Gen3 high speed. Protocol Exerciser in its mode as an LTSSM provides you LTSSM functions to test DUT's LTSSM. You can use various LTSSM tests that it provides to force DUT into various LTSSM states and verify the state transitions and timeout implementations.

To know more about its usage as an LTSSM, refer to the topic When to Use LTSSM Tester on page 92.

Functions, Hardware Channels, Virtual Hierarchies, and Virtual Channels

This topic provides information on the physical and virtual functions, hardware channels, virtual hierarchies, and virtual channels that Protocol Exerciser supports in its three different emulations as a non IOV PCIe component, MRIOV capable component, and SRIOV capable component.

For information on how to send stimulus using these functions, hierarchies, and channels of Protocol Exerciser, refer to Providing Stimulus To DUT.

As a Non IOV PCIe component

	Default	With additional 2 physical functions license	Notes
Functions	3 Function A (function number 0) Function B (function number 1) Function C (function number 2)	5 Function A (function number 0) Function B (function number 1) Function C (function number 2) Function D (function number 3) Function E (function number 4)	All functions are non IOV functions.
Hardware Channels	3	5	One hardware channel associated with each function
Virtual Channels	2	2	Virtual Resource 0 - VC0 Virtual Resource 1 - VC [x] (where x is the number that you assign to this second VC resource)
Completion Queues	2 (Queue0 and Queue1)	2 (Queue0 and Queue1)	Queue0 mapped to VC resource 0 Queue1 mapped to VC resource 1
Virtual Link	1	1	
Virtual Functions/ hierarchies	N/A	N/A	

As a SRIOV capable component

	Default	With additional 2 physical functions license	Notes
Base and Physical Functions	3 Function A (function number 0) Function B (function number 1) Function C (function number 2)	5 Function A (function number 0) Function B (function number 1) Function C (function number 2) Function D (function number 3) Function E (function number 4)	Function A is non IOV. Function B to E are physical functions and have the SRIOV extended capabilities.
Virtual functions	4	8	Two virtual functions (VF1 and VF2) associated with each physical function
Hardware Channels	7	13	One hardware channel associated with each base, physical and virtual function
Virtual Channels	2	2	Virtual Resource 0 - VC0 Virtual Resource 1 - VC [x] (where x is the number that you assign to this second VC resource)
Completion Queues	2 (Queue0 and Queue1)	2 (Queue0 and Queue1)	Queue0 mapped to VC resource 0 Queue1 mapped to VC resource 1
Virtual Link	1	1	
Virtual hierarchies	N/A	N/A	

As a MRIOV capable component

	Default	With additional 2 physical functions license	Notes
Base and Physical Functions	3 Function A (function number 0) Function B (function number 0) Function C (function number 0)	5 Function A (function number 0) Function B (function number 0) Function C (function number 0) Function D (function number 0) Function E (function number 0)	Function A is the base function with MRIOV extended capabilities. Function B to E are physical functions and have the SRIOV extended capabilities.
Virtual functions	4	8	Two virtual functions (VF1 and VF2) associated with each physical function - B to E. VF1 and VF2 are assigned the function numbers 1 and 2.
Hardware Channels	7	13	One hardware channel associated with each base, physical and virtual function
Virtual Hierarchies	3 (VH0 to VH2)	5 (VH0 to VH4)	Each base and physical function is mapped to a virtual hierarchy. Function A is mapped to VH0 and so on. The virtual function of a physical function are mapped to the same hierarchy as their physical function. For instance, VF1 and VF2 of Function A are mapped to VH0.
Virtual Channels	1 (Virtual Resource 0 - VC0)	1 (Virtual Resource 0 - VC0)	All virtual hierarchies are mapped to VC0.
Completion Queues	3 (Queue0 to Queue2)	5 (Queue0 to Queue4)	Each virtual hierarchy is mapped to a completion queue. For instance, VH0 is mapped to queue 0 and so on. All queues are mapped to VC resource 0
Virtual Link	1	1	

Protocol Exerciser for PCI Express Components

This topic describes the hardware and software components of Protocol Exerciser.

Hardware Components

U4305A Exerciser Card

The U4305A Exerciser card acts as both a Protocol Exerciser as well as an LTSSM for upto x16 link widths. This card supports simultaneously use of LTSSM and Protocol Exerciser functions, without requiring any configuration.

The following figure shows the U4305A exerciser card.

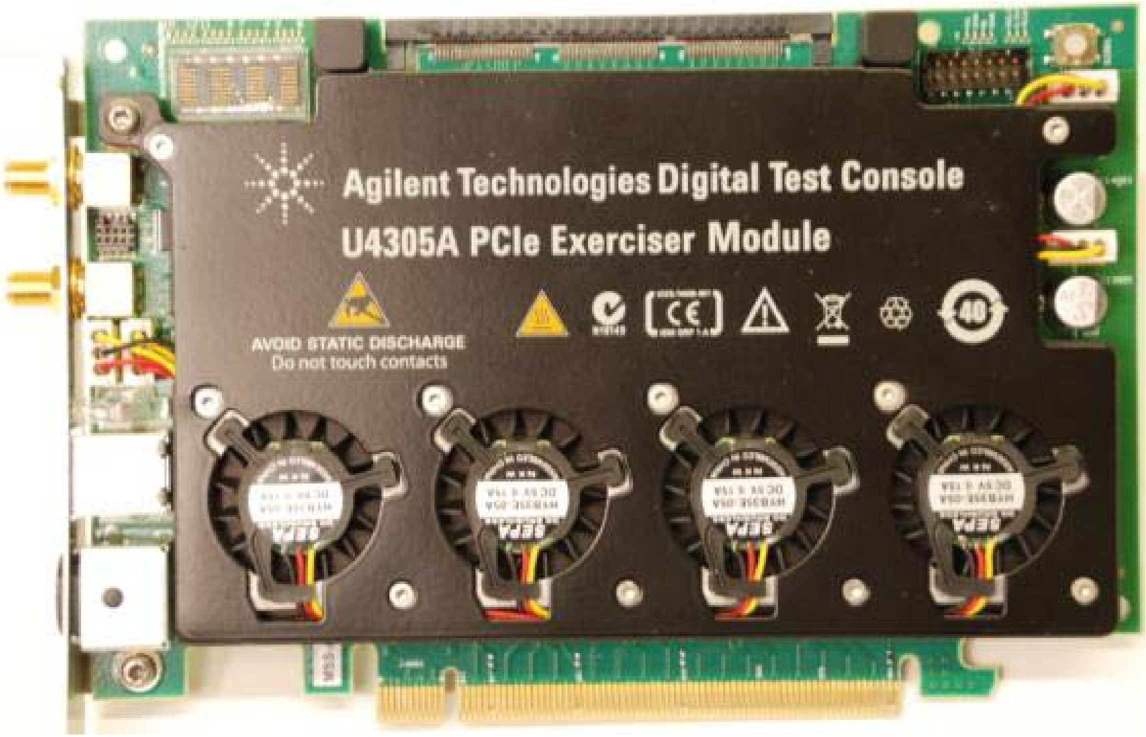

Agilent Technologies Digital Test Console
U4305A PCIe Exerciser Module
AVOID STATIC DISCHARGE
Do not touch contacts
SEPA

Figure 3 U4305A Exerciser Card

NOTE

Refer to the Hardware guide for the Protocol Exerciser for PCIe Express to know more about the U4305A Exerciser card.

While setting up the hardware for Protocol Exerciser, you can mount this card on the System under Test (SUT) to stimulate SUT with PCIe testing scenarios. You can also set up this card as a root complex by mounting it on the backplane board and communicating with the DUT through this passive backplane.

NOTE

Refer to the Installation guide for the Protocol Exerciser for PCIe Express to know how to set up the Protocol Exerciser hardware to emulate as a root complex or as a PCIe endpoint.

Software Components

When you install Protocol Exerciser, the following software components are installed.

- **Protocol Exerciser GUI** - You can use the Protocol Exerciser GUI to control, manage, and use the Protocol Exerciser card. Following are the primary functions of the Protocol Exerciser GUI.
  - Create and manage setup files for Protocol Exerciser
  - Create a link between DUT and Protocol Exerciser
  - Run LTSSM tests provided to test DUT's LTSSM functions
  - View and update the link and Exerciser hardware status
  - Define stimulus traffic
  - Start the Stimulus and view results

  To know about the Protocol Exerciser GUI components, refer to the topic Protocol Exerciser GUI Components on page 38.
- **Protocol Exerciser API** - Protocol Exerciser provides a set of APIs to allow you to control, manage, and use Protocol Exerciser programmatically through these APIs. Refer to the API Reference guide to know about the usage of these APIs.
- **Firmware Update tool** - You use this tool to update the Firmware version for an Exerciser card. To know more, refer to the topic, Using the Firmware Update Tool on page 242.

NOTE

This guide covers only the GUI aspect of Protocol Exerciser. For information on API, please refer to the API Reference guide.

2
Getting Started with Protocol Exerciser

Creating a Protocol Exerciser Session 30
Programming Protocol Exerciser Card as Gen3 Exerciser 35
Starting and Exiting the Protocol Exerciser GUI 36
Protocol Exerciser GUI Components 38
Viewing and Updating the Protocol Exerciser Hardware Status 40
Managing Floating Windows 46
Configuring Data Display Preferences Settings 48

This chapter describes the tasks that you should perform before you get started with using Protocol Exerciser to test a DUT. This chapter also introduces you to the main screen components of the Protocol Exerciser application and how to access it.

Creating a Protocol Exerciser Session

You can access Protocol Exerciser in either Offline or Online mode. This topic describes the Offline and Online modes of using the Protocol Exerciser GUI.

Protocol Exerciser Online mode

When you want to use Protocol Exerciser in an online mode, you either create a new Protocol Exerciser session with the controller PC or use an existing session on the controller PC. A controller PC hosts the Protocol Exerciser software and hardware support services and connects to the Exerciser card(s) through a USB. A Protocol Exerciser session forms the basis of communication between the controller PC and exerciser cards, and also initializes the necessary firmware. Once a session is created, you can control and manage the installed exerciser card(s) using the Protocol Exerciser GUI.

Multiple clients can remotely connect to a single Protocol Exerciser session on the controller PC.

The following figure displays a Protocol Exerciser session scenario in which Session A and Session B have been created on the controller PC with two Protocol Exerciser cards connected to the controller PC through USB. Two clients are accessing Session A and one client is accessing Session B.

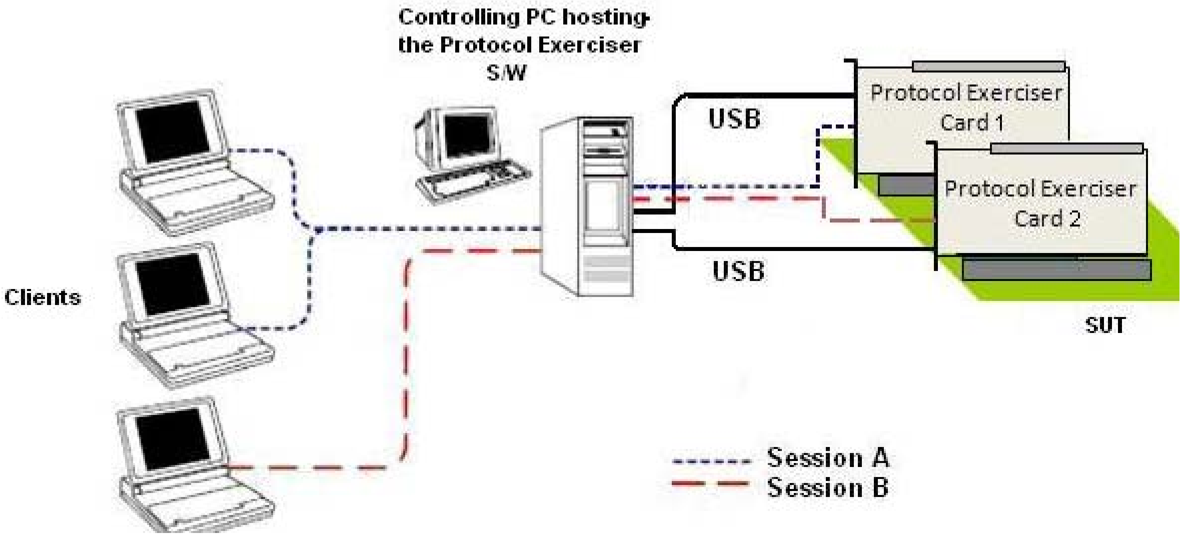

Controlling PC hosting the Protocol Exerciser S/W
USB
Protocol Exerciser Card 1
Protocol Exerciser Card 2
USB
Clients
SUT
Session A
Session B

Figure 4 Protocol Exerciser Sessions

In an online session, you generate real-time PCIe traffic to exercise a DUT. Therefore, before you start an online session, ensure that at least one exerciser card is connected with the Controller PC. Ending an online session releases the hardware resources of Protocol Exerciser, thereby enabling these to be used for a new session.

Protocol Exerciser Offline mode

When you want to use Protocol Exerciser in an offline mode, you do not need to create a session between the Controller PC and an Exerciser card. You generally use an offline mode when you do not have an established connection with any external hardware device. However, you want to create configuration specifications to be used later when an online session is established.

Some options in offline mode are disabled due to their dependency on the connected hardware. For instance, the Hardware status is not displayed in the offline mode.

Protocol Exerciser does not enable you to lock a session (online or offline). Therefore, it is possible for multiple Protocol Exerciser instances to concurrently access a single session.

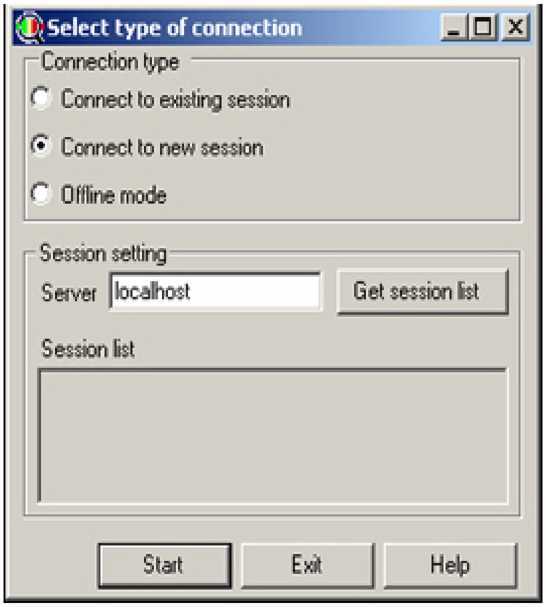
NOTE

The controller PC does not protect against meaningless, or even conflicting requests. Therefore, it is recommended that only one user should own a particular session at a time.

Creating/Using a Protocol Exerciser Session

While accessing the Protocol Exerciser GUI, you need to select whether you want to connect to a new or an existing Protocol Exerciser session or you want to use the GUI in an offline mode.

The **Select type of connection** dialog box helps you specify the mode of accessing Protocol Exerciser and the session related information if an online mode is needed. This dialog box is displayed when you try to access the Protocol Exerciser GUI.



Select type of connection

Connection type

○ Connect to existing session

● Connect to new session

○ Offline mode

Session setting

Server localhost

Figure 5 Select type of connection dialog box

Working in an offline mode

In the Select type of connection dialog box, select the **Offline mode** radio button and click **Start**.

Using an existing Protocol Exerciser session

1 Select the **Connect to existing session** radio button.

2 In the **Server** text box, specify the name or IP address of the controlling server that hosts the Protocol Exerciser software and on which you want to use an existing session.

The **Session list** listbox displays the list of Protocol Exerciser sessions existing on the specified server.

3 Select a session from the **Session list** listbox.

4 Click **Start**.

Creating a new Protocol Exerciser session

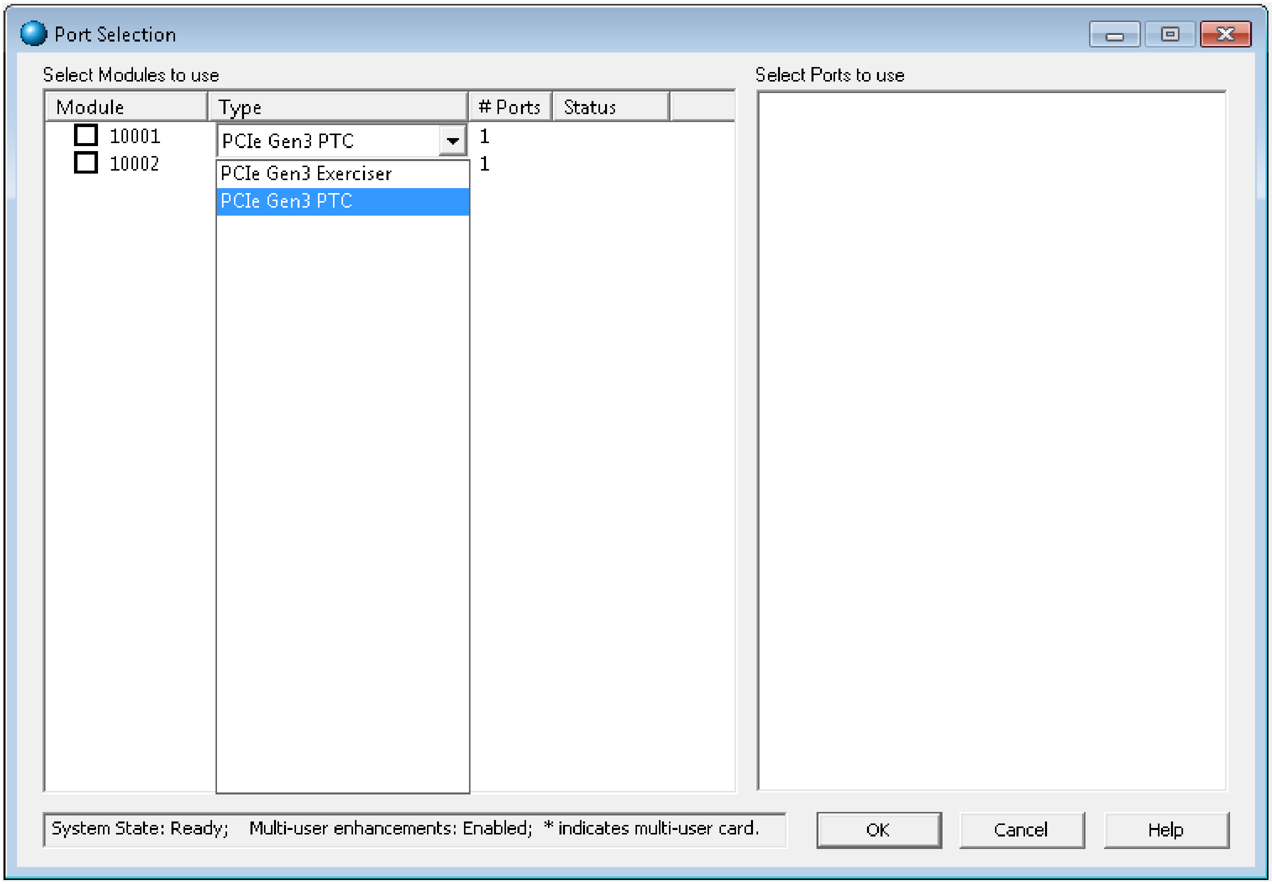
1 Select the **Connect to new session** radio button.

2 In the **Server** text box, specify the name or IP address of the controlling server that hosts the Protocol Exerciser software and on which you want to create a new session.

3 Click **Start**.

The **Port Selection** dialog box is displayed. It displays a list of ready to use as well as already in use serial I/O exerciser modules along with their type. There are currently two module types supported: *PCIe Gen3 Exerciser* for exerciser mode and *PCIe Gen3 PTC* for PTC mode. Here, you can only select an exerciser module, which is not in use.

In *PCIe Gen3 Exerciser* mode, the Compliance Test page is disabled while in *PCIe Gen3 PTC* mode, only Compliance Test and General Settings page are enabled. However access to PTC requires a separate license other than that of exerciser.



Module	Type	# Ports	Status
☐ 10001	PCIe Gen3 PTC	1	
☐ 10002		1	

System State: Ready; Multi-user enhancements: Enabled; * indicates multi-user card.

Figure 6 Port Selection dialog box

4 Select the appropriate Exerciser module and then select the available port of the selected Exerciser module from the **Select Port to use** listbox.

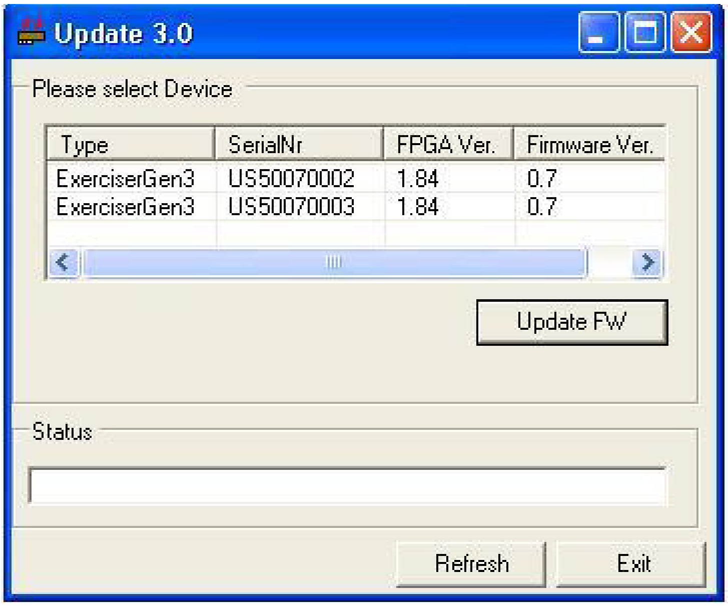
5 Click **OK**.

At any time, you can access the session information by clicking the **Help -> Session I**nformation menu option in the Protocol Exerciser GUI.

Programming Protocol Exerciser Card as Gen3 Exerciser

Before you start using Protocol Exerciser, check if the installed Protocol Exerciser card(s) are programmed as Gen3 Exerciser to ensure that the card contains the appropriate FPGA version. When you add ports while creating a Protocol Exerciser session, the appropriate FPGA is automatically downloaded on the controller PC. However, if the FPGA type is not appropriate and updated or the Firmware version of the card is not updated, you can use the Firmware Update tool to accomplish this. Refer to the topic Using the Firmware Update Tool on page 242 to know more.

When you program the Protocol Exerciser card for Gen3 Exerciser, the card type is set as Gen3 Exerciser in the FirmWare Update tool.



Type	SerialNr	FPGA Ver.	Firmware Ver.
ExerciserGen3	US50070002	1.84	0.7
ExerciserGen3	US50070003	1.84	0.7

Figure 7 FirmWare Update tool

Starting and Exiting the Protocol Exerciser GUI

To access Protocol Exerciser GUI

1 On the Windows task bar, click **Start** > **Programs** > **Agilent SPT** > **PCIe Exerciser 8.3 Release** > **Exerciser GUI**.

   The **Select type of connection** dialog box appears.

2 Select the Protocol Exerciser mode in which you want to use the GUI (Online or offline). If you want to use the GUI in an online mode, then either create a new session or join an existing session on the controller PC that hosts the Protocol Exerciser software and is connected the exerciser card hardware. Refer to the topic Creating a Protocol Exerciser Session on page 30 to know more about sessions.

The main **Protocol Exerciser** window is displayed.

Depending on the selected mode of usage (online or offline), some GUI features are enabled/disabled.

Exiting Protocol Exerciser GUI

To exit from the Protocol Exerciser GUI:

1 Click **File** > **Exit**.

   The Changes not applied message appears if you had not applied all the changes made to Protocol Exerciser.

2 Do one of the following:

   - Click **Yes** to apply all the unapplied changes before closing the Protocol Exerciser window.
   - Click **No** to close the Protocol Exerciser window without applying the unapplied changes.
   - Click **Cancel** to quit the message box and return to Protocol Exerciser.

   Clicking **Yes** or **No** displays a message box that prompts to save the settings of the current session in a file.

3 Do one of the following:

   - Click **Yes** to save the underlying Protocol Exerciser settings before exiting.
   - Click **No** to close the Protocol Exerciser window without saving the underlying Protocol Exerciser settings.

- Click **Cancel** to quit the message box and return to Protocol Exerciser.

Completing this message box displays the Closing Session dialog box.

4 Do one of the following:

- Click **Yes** to remove the associated session before closing the Protocol Exerciser window.
- Click **No** to close the Protocol Exerciser window without removing its associated session.

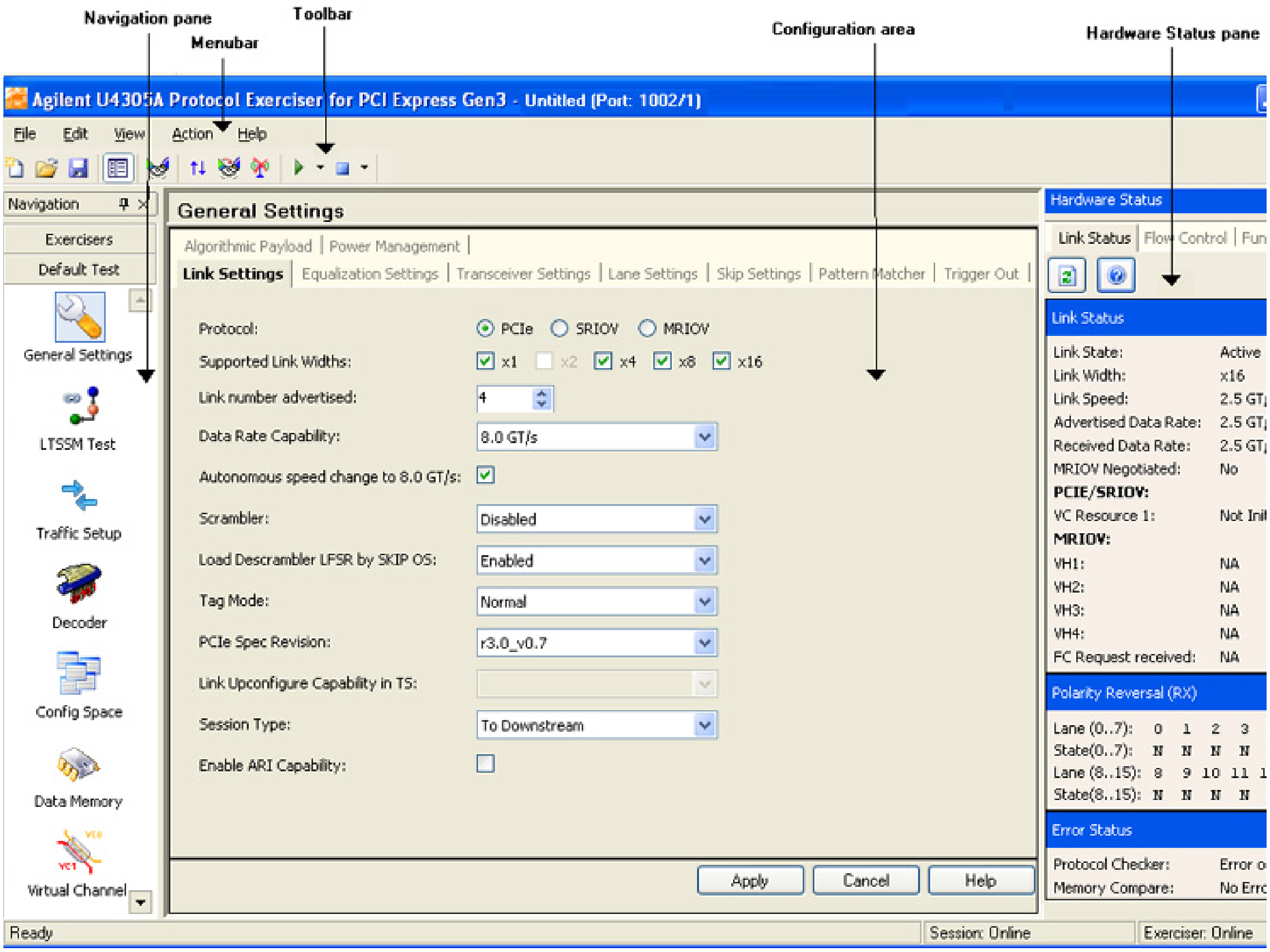
NOTE

Closing a session automatically makes all the GUI interfaces that are connected to it, inaccessible.

Protocol Exerciser GUI Components

This topic describes various components of the Protocol Exerciser GUI to help you understand how it is organized in various pane, the purpose of each pane, and how to access these panes.

The following figure displays the Protocol Exerciser application main window with its main components.



Protocol: ● PCIe ○ SRIOV ○ MRIOV

Supported Link Widths: ☑ x1 □ x2 ☑ x4 ☑ x8 ☑ x16

Link number advertised: 4

Data Rate Capability: 8.0 GT/s

Autonomous speed change to 8.0 GT/s: ☑

Scrambler: Disabled

Load Descrambler LFSR by SKIP OS: Enabled

Tag Mode: Normal

PCIe Spec Revision: r3.0_v0.7

Link Upconfigure Capability in TS:

Session Type: To Downstream

Enable ARI Capability: □

Link State:	Active
Link Width:	x16
Link Speed:	2.5 GT/
Advertised Data Rate:	2.5 GT/
Received Data Rate:	2.5 GT/
MRIOV Negotiated:	No
PCIE/SRIOV:	
VC Resource 1:	Not Ini
MRIOV:	
VH1:	NA
VH2:	NA
VH3:	NA
VH4:	NA
FC Request received:	NA

```
Lane (0..7):   0  1  2  3
State(0..7):   N  N  N  N
Lane (8..15):  8  9 10 11 1
State(8..15):  N  N  N  N
```

Protocol Checker:	Error o
Memory Compare:	No Erro

Ready | Session: Online | Exerciser: Online

Figure 8 Protocol Exerciser window

The main components of the Protocol Exerciser window are briefly described below:

- **Menu Bar and Toolbar**: These provide easy and quick access to the features of the Protocol Exerciser application.

- **Navigation Pane**: Displays the features of Protocol Exerciser as various icons. Using this pane, you can access and navigate through various features of Protocol Exerciser.
- **Configuration Area**: Displays the existing settings and allows you to specify the new settings for the Protocol Exerciser feature currently selected in the Navigation pane.
- **Status Bar**: Displays the current status of the various Protocol Exerciser activities, such as user level (User or Expert), session status (offline or online), and exerciser status (offline or online).
- **Hardware Status pane**: Displays the Protocol Exerciser hardware status and the link status. The information in this pane is updated automatically when you perform any function or run a test that impacts the hardware and link status.

You can view/hide a pane using the **View** menu.

Viewing and Updating the Protocol Exerciser Hardware Status

You can get information about the Protocol Exerciser hardware status and the link status using the Hardware Status pane of the Protocol Exerciser GUI.

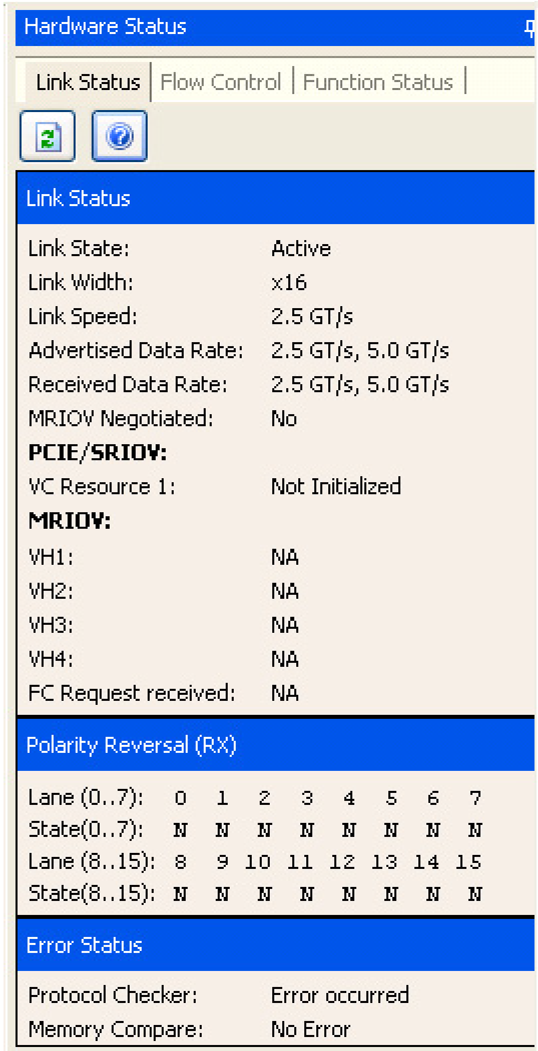
The information in this pane is updated when you perform any action such as sending stimulus or running a test that impacts the link partners and link status. For instance, when the link between the DUT and Protocol Exerciser is not up, the link status is displayed as No Link. When you perform the steps to initialize this link, the link status is updated to Active if the procedure is completed successfully. Consider a situation when you selected MRIOV as the protocol for the link up. You can check whether or not the PCIe link is MR enabled by viewing the status of MRIOV negotiations using the Hardware status window.

If needed, you can also manually update the status on this pane using the ⟳ button after performing a specific action and to see its impact on the status. To view the status of flow control credits and outstanding requests for each function, you need to click the ⟳ button.

To access the Hardware Status pane:

- Click **View** > **Hardware Status**.

The **Hardware Status** pane appears. A sample screen is displayed.



Link Status

Link State: Active
Link Width: x16
Link Speed: 2.5 GT/s
Advertised Data Rate: 2.5 GT/s, 5.0 GT/s
Received Data Rate: 2.5 GT/s, 5.0 GT/s
MRIOV Negotiated: No
PCIE/SRIOV:
VC Resource 1: Not Initialized
MRIOV:
VH1: NA
VH2: NA
VH3: NA
VH4: NA
FC Request received: NA

Polarity Reversal (RX)

```
Lane (0..7):   0  1  2  3  4  5  6  7
State(0..7):   N  N  N  N  N  N  N  N
Lane (8..15):  8  9 10 11 12 13 14 15
State(8..15):  N  N  N  N  N  N  N  N
```

Error Status

Protocol Checker: Error occurred
Memory Compare: No Error

Figure 9 Hardware Status pane

This pane has the status information grouped under the following three tabs:

Link Status

The **Link Status** tab provides you the following information:

- The current state of the PCIe link between DUT and Exerciser.
- The width at which the link is currently active.
- The current speed of the underlying link.
- The advertised data rate capabilities of the Protocol Exerciser.

- The received data rate, that is the data rate capabilities of the DUT.
- Whether or not MRIOV has been negotiated on the PCIe link. If you select the **MRIOV** protocol for Exerciser in the **General Settings** -> **Link Settings** page and MRIOV is also applicable at the DUT's end, then **MRIOV Negotiated** is displayed as **Yes** after link up. This indicates that the PCIe link is MR enabled.
- Whether or not the second virtual channel (VC Resource 1) of Exerciser has got initialized. VC0 gets initialized by default. However, VC Resource 1 gets initialized only when you select PCIe or **SRIOV** protocol in the **Link Settings** page and **Enable VC Resource 1** checkbox in the **Virtual Channels** page.
- Whether or not the virtual hierarchy 1 to 4 (VH1 - VH4) of Exerciser have got initialized for an MRIOV link. In case of an MRIOV link, the virtual hierarchy **VH0** gets initialized by default after linkup. However, the other four hierarchies (VH1 to VH4) get initialized if you enable these hierarchies by selecting the **Enable Virtual Hierarchy** checkbox in the **Virtual Channels** page. When disabled, the virtual hierarchy's status is displayed as **Not Initialized**.
- The type of flow control that has been requested in case of an MR Enabled link. For a non IOV/SRIOV capable link, this field is not applicable. If you select the **Enable per VH Flow control** checkbox in the **Virtual Channels** page and both link partners support this type of flow control, then Per VH Based flow control is displayed. If per VH flow control is not enabled in the Virtual Channels page or the link partner does not support it, then **Per VL based** flow control is applicable and displayed. Refer to the topic Initializing Flow Control on page 79 to know more about these two types of flow controls.
- The received state of polarity for each lane.
- Protocol Checker - Displays the **Error occurred** message if a rule that you enabled for checking in the **Protocol Checker** page has been violated. Refer to the topic Performing Protocol Checks on page 169 to know more.

- Memory Compare: Displays the **Error occurred** message if there is a compare error between the payload data of an incoming TLP and the data at a specific location in the data memory. Displays the No Errors message if there is no compare error between the data of an incoming TLP and the data in the targeted data memory location of Exerciser. Refer to the topic Comparing Actual Data with Expected Data in Memory on page 126 to know more.

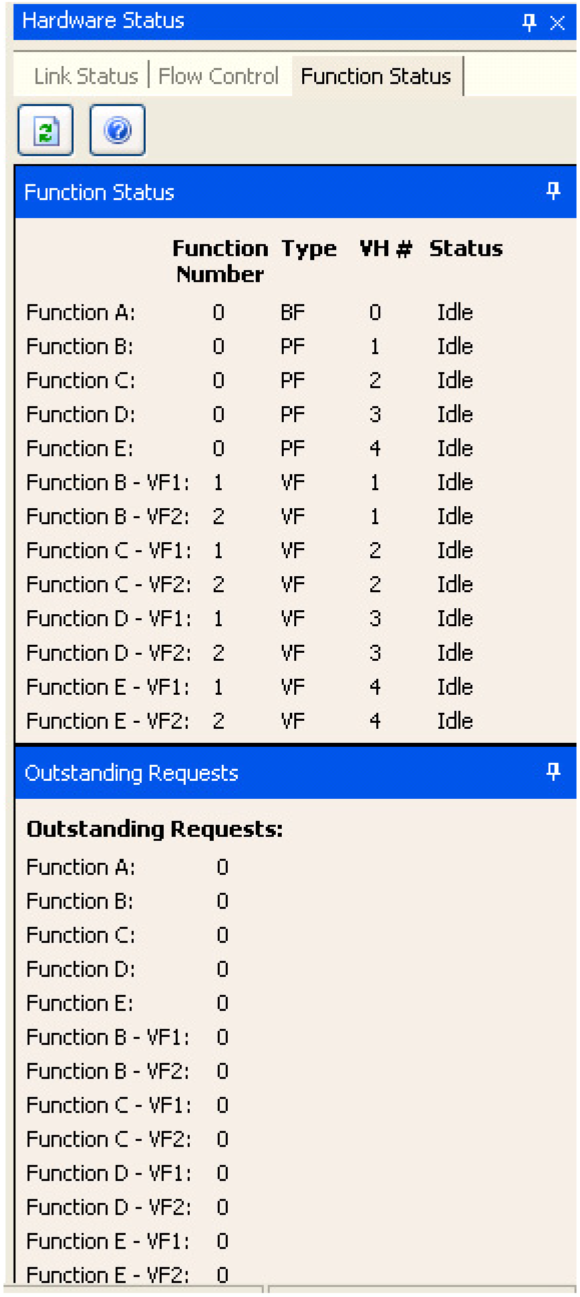
Flow Control

This tab provides the flow control credit information of the underlying Exerciser's Receiver Buffer and the credit limit available to the Transmitter. It displays the allocated, used, and available credit limit of virtual channels of Exerciser as per the credit limits that you assigned to these virtual channels in the **Virtual Channels** page.

The flow control credit of Exerciser decreases when Exerciser sends a packet and it increases when the DUT frees buffer by sending a flow control update packet.

To view the current status of flow control credits, click the 🔄 button.

Function Status



	Function Number	Type	VH #	Status
Function A:	0	BF	0	Idle
Function B:	0	PF	1	Idle
Function C:	0	PF	2	Idle
Function D:	0	PF	3	Idle
Function E:	0	PF	4	Idle
Function B - VF1:	1	VF	1	Idle
Function B - VF2:	2	VF	1	Idle
Function C - VF1:	1	VF	2	Idle
Function C - VF2:	2	VF	2	Idle
Function D - VF1:	1	VF	3	Idle
Function D - VF2:	2	VF	3	Idle
Function E - VF1:	1	VF	4	Idle
Function E - VF2:	2	VF	4	Idle

Outstanding Requests:

Function A:	0
Function B:	0
Function C:	0
Function D:	0
Function E:	0
Function B - VF1:	0
Function B - VF2:	0
Function C - VF1:	0
Function C - VF2:	0
Function D - VF1:	0
Function D - VF2:	0
Function E - VF1:	0
Function E - VF2:	0

Figure 10 Function status tab

This tab provides the status information related to each physical and virtual function that Exerciser supports based on its configuration. If you select the PCIe protocol and Exerciser emulates a non IOV PCIe component, then this tab displays information related to the three functions, Function

A, B, and C. If you select the MRIOV or SRIOV protocol, then this tab displays information related to the virtual functions as well. If you have the additional 2 physical function license, then this tab displays the status of five functions (Function A to E). Refer to the topic Functions, Hardware Channels, Virtual Hierarchies, and Virtual Channels on page 22 to get a description of each function of Exerciser.

The function number assigned to each function of Exerciser is fixed and displayed in this tab. If there are no pending requests for a function, the status for that function is displayed as Idle.

The **Outstanding Requests** section displays the number of TLP requests for a function which are not yet completed. The tags assigned to these outstanding requests are not freed up until these requests are completed. If there are any outstanding requests for a function, then this section displays a value representing the number of tags that have not yet been released.

To view the current status of outstanding requests for each function, click the 🔄 button.

Managing Floating Windows

The main Protocol Exerciser window contains many floating windows. These windows can float anywhere on the desktop, or you can dock them together with other windows on the main Protocol Exerciser window. In the default window arrangement, all windows are properly docked on the main Protocol Exerciser window.

In this section, you will learn:

- To Display a Floating Window
- To Hide a Floating Window
- To Auto-Hide a Floating Window
- To Change a Docked Window to a Floating Window
- To Dock a Floating Window

To Display a Floating Window

- Click **View** > ***menu_command_for_floating_window***.

  Here, *menu_command_for_floating_window* represents the name of the menu command to access the floating window. For example, Hardware Status is a floating window. To display it, click the **View** > **Hardware Status** menu command.

To Hide a Floating Window

- Click the ☒ icon on the title bar of the floating window.

  You can also hide a floating window by clicking its menu command in the View menu.

To Auto-Hide a Floating Window

- Click the 📌 icon on the title bar of the floating window.

  This automatically hides the floating window when you are not using it. This also displays a new 📌 icon on the title bar of the floating window.

  To display the window again, place the cursor over the tab displayed for the hidden window, or click its menu command in the View menu.

To Change a Docked Window to a Floating Window

- Double-click the title bar of the docked window.

To Dock a Floating Window

- Double-click the title bar of the floating window.

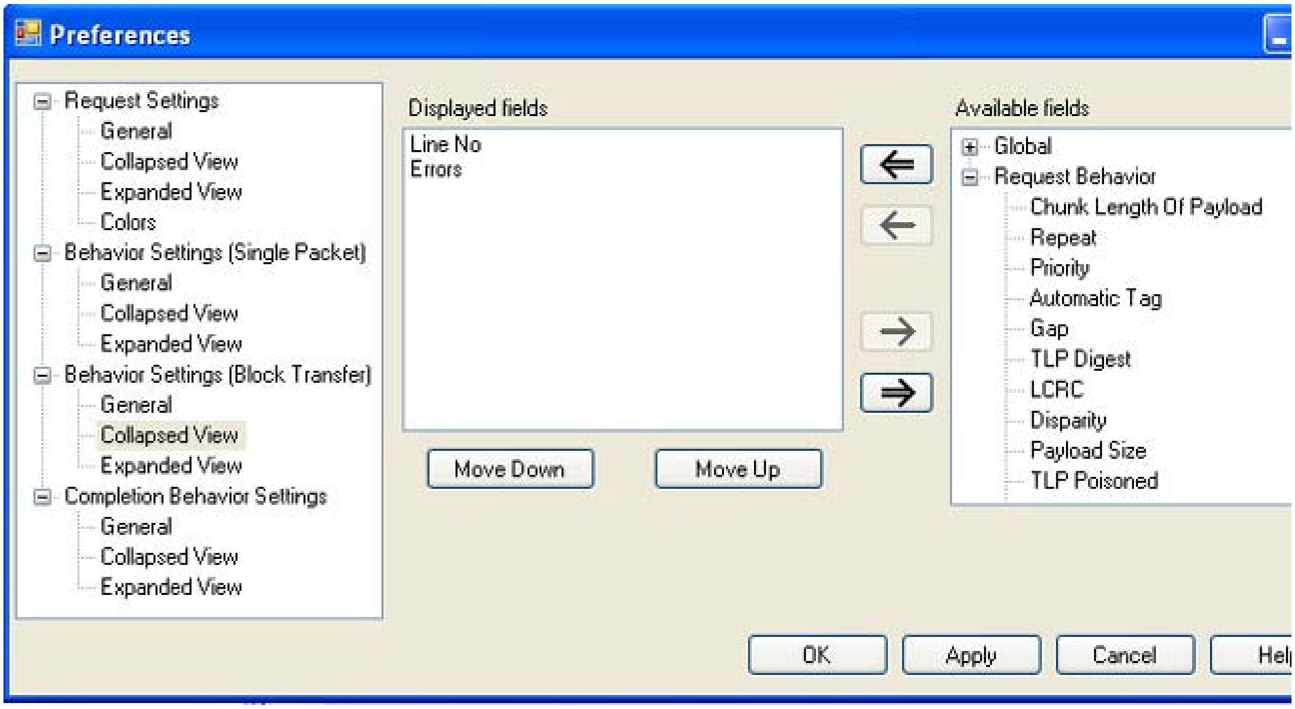
Configuring Data Display Preferences Settings

By default, Protocol Exerciser displays the configured stimulus data in the Configuration area of the GUI using predefined columns, fields, time format, and color settings. There may be a situation when you want to make some changes to these predefined settings to match your data display needs. To make such changes, Protocol Exerciser provides the Preferences dialog box. This dialog box allows you to make changes to the way the stimulus data (TLP requests and behaviors) that you added is displayed in the Configuration area of the Traffic Setup page.

To access the Preferences dialog box:

Select **View** > **Preferences**.

The **Preferences** dialog box appears.



You can configure the preferences for the displays of the TLP requests, the applicable request behavior records, and the behavior records for the completions sent from Exerciser to DUT.

To get a description of each field in this dialog box, refer to the GUI Reference help topic for Preferences dialog box on page 200.

3
Configuring Protocol Exerciser

Creating and Managing Protocol Exerciser Setup Files 50
Configuring a PCIe Link Settings 51
Configuring Lane Settings 54
Configuring Equalization Procedure Settings 56
Configuring Transceiver Settings 61
Configuring Pattern Matchers 64
Configuring External Triggers Settings 69
Configuring Power Management Settings 71

This chapter provides information on the various settings that you can configure in the General tab of the Protocol Exerciser GUI to customize its functions according to your specific requirements. The help book also describes how to create and manage a Protocol Exerciser setup file.

Creating and Managing Protocol Exerciser Setup Files

Protocol Exerciser performs its functions as an Exerciser and an LTSSM based on the settings that you configure in the Protocol Exerciser GUI. These settings are used by Protocol Exerciser to configure a session with DUT for functions such as link and lane negotiations, exchanging training sequences, performing equalization, and link speed changes.

Protocol Exerciser saves these settings in the Protocol Exerciser setup file. You can create multiple versions of this file for multiple Exerciser cards with different link, lane, and other settings to use these versions in different testing scenarios with DUTs. For instance, you can create different setup files for an Upstream and a Downstream session with DUT and use these files based on whether Protocol Exerciser is emulating a root complex (as USC) or a PCIe endpoint (as DSC). You may also want to have different setup files for a session type with different link speed or lane width settings to test link negotiations with different link settings scenarios.

Ensure that you have created the Protocol Exerciser setup file with the required settings before starting link up between the DUT and Protocol Exerciser.

To create a Protocol Exerciser setup file, click the **General** icon in the Navigation window of the Protocol Exerciser GUI and then click the relevant tabs on the screen to configure various settings. Save the file with an appropriate name. The Protocol Exerciser setup files are saved with an exs extension file.

Using the File menu options, you can open, close, or save a Protocol Exerciser setup file.

NOTE

Some fields in the General tab pages are disabled if you are using Protocol Exerciser in an offline mode. Also, if the Protocol Exerciser is used as a DSC (based on the configured session type), some fields which are not applicable for Protocol Exerciser as a DSC are disabled.

Refer to the Protocol Exerciser GUI Reference help topics in this online help to get a detailed description of each field in the tab pages. Alternatively, click the Help button displayed in the tab page to get a context-sensitive help page.

Configuring a PCIe Link Settings

Before you can create/initialize a PCIe link between Protocol Exerciser and DUT for normal exchange of data packets, you need to configure the settings for this link. This ensures that Protocol Exerciser initializes the link as per your specific link settings. To configure the link settings, click the **General Settings** icon in the Navigation window of the Protocol Exerciser GUI and then click the **Link settings** tab.

NOTE

Protocol Exerciser saves the link settings in an Exerciser setup file. You can create multiple versions of this file with different link, lane, and other settings to use these versions in different testing scenarios with DUTs. To know more, refer to Creating and Managing Protocol Exerciser Setup Files on page 50.

Types of link

You can create either a PCIe link or an MR (Multi root) enabled PCIe link using the Protocol Exerciser GUI.

- **PCIe link** - If the PCIe component that you want to test is not IOV capable, you can choose the **PCIe** option in the **Protocol** radio button in the **Link settings** tab. Selecting this option ensures that a non IOV PCIe link is created between the Protocol Exerciser and DUT as per the PCI Express 3.0 Base Specifications. Exerciser does not have any IOV capabilities in this case and can support only one domain hierarchy at any time.
- **PCIe link with SRIOV capabilities** - If you want to stimulate an SRIOV capable PCIe component, then Protocol Exerciser needs to emulate a PCIe device with SRIOV capabilities. To make the Exerciser SRIOV capable, you need to select the **SRIOV** option in the **Protocol** radio button in the **Link settings** tab. Selecting this option ensures that a PCIe link is created as per the SRIOV protocol between the Exerciser and DUT. Protocol Exerciser acts as a SRIOV capable component in this case. To enable the SRIOV capabilities for Exerciser, you need to have the **U4305A-023** or **U4305U-023** SRIOV software license. In the absence of this license, the **SRIOV** option is disabled in the **Link settings** tab. Exerciser can function at Gen1 and Gen 2 speed when emulating a SRIOV component.

- **MR Enabled PCIe link** - If you want to stimulate an MRIOV capable PCIe component, then Protocol Exerciser needs to emulate a PCIe device with MRIOV capabilities. To make the Exerciser MRIOV capable, you need to select the **MRIOV** option in the **Protocol** radio button in the **Link settings** tab. Selecting this option ensures that an MR (Multi root) enabled PCIe link is created as per the MR Link protocol between the Exerciser and DUT. Protocol Exerciser acts as a MRIOV capable component in this case and can support three virtual hierarchies at a time. Exerciser can function at Gen1 and Gen 2 speed when emulating a MRIOV component.

NOTE

Refer to the Protocol Exerciser GUI Reference help topics in this online help to get a detailed description of each field in this tab page. Alternatively, click the Help button displayed in the tab page to get a context-sensitive help page.

Some fields in this tab page are disabled if you are using Protocol Exerciser in an offline mode. Also, if the Protocol Exerciser is used as a DSC (based on the configured session type), some fields which are not applicable for Protocol Exerciser as a DSC are disabled.

The following are some of the points to be considered when configuring link settings.

- You select the session type based on whether you want the Protocol Exerciser to act as a USC or a DSC for the session. When Protocol Exerciser is emulating a root complex and acting as a USC, the session type should be **To Downstream**. The DUT is at the end point in this case. When Protocol Exerciser is emulating a PCIe end point and acting as a DSC, the session type should be **To Upstream**. The DUT is at root-complex in the hierarchy in this case.
- Select the data rate capabilities of the Protocol Exerciser based on the link speed scenarios that you want to test. The data rate capabilities options are disabled in the offline mode of Protocol Exerciser.

- Protocol Exerciser performs data scrambling as per the PCIe specifications. At Gen1 and Gen2 speeds, you can enable or disable scrambling using the Link settings page. If you disable scrambling, Protocol Exerciser generates and responds to training sequences with the “disable scrambling” bit set, which causes the link to operate without scrambling. The disabling takes effect the next time Protocol Exerciser exchanges Training sequences after you select the Disable Scrambling option. At the Gen3 speed, scrambling is always enabled to follow the new Gen3 Scrambling Rules as listed in the PCIe specifications. Therefore, the Enable/Disable Scrambling option is not provided for the Gen3 speed.
- If the data rate capabilities of both the link partners is 8.0 GT/s and you want to initialize the link to 8.0 GT/s, then ensure that the **Autonomous speed change to 8 GT/s** option in the **Link Settings** tab is selected. Selecting this check box autonomously moves the link speed to 8.0 GT/s during link training.

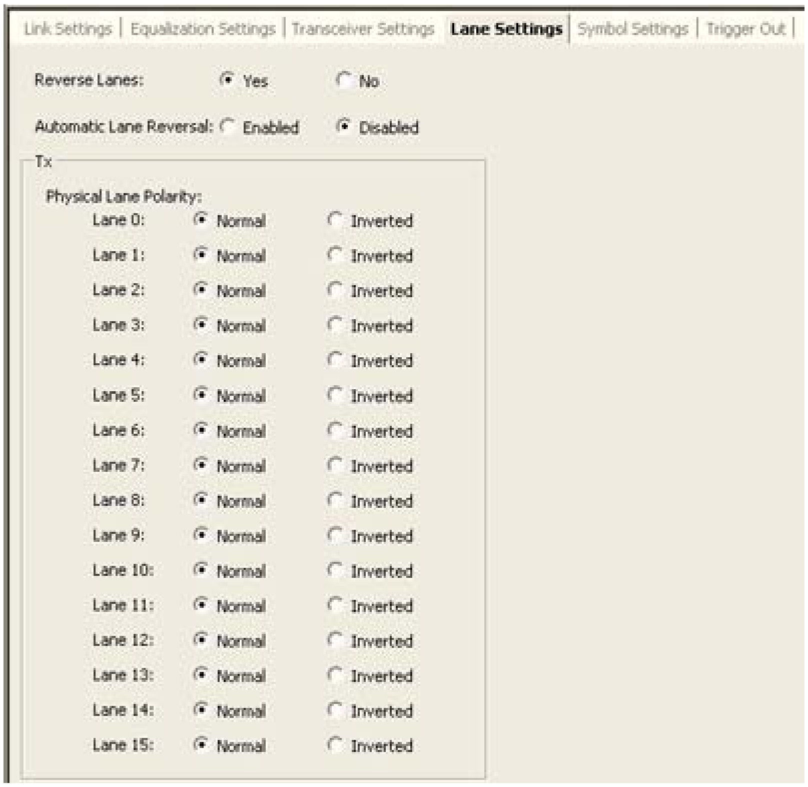
Configuring Lane Settings

Before you can create/initialize a PCIe link between Protocol Exerciser and DUT for normal exchange of data packets, you need to configure the lane settings for this link. This ensures that the lanes of a port are associated and configured into a link through link width and lane negotiation as per the specified lane settings.

NOTE

Protocol Exerciser saves the lane settings in an Exerciser setup file. You can create multiple versions of this file with different link, lane, and other settings to use these versions in different testing scenarios with DUTs. To know more, refer to Creating and Managing Protocol Exerciser Setup Files on page 50.

To configure the lane settings, click the **General Settings** icon in the Navigation window of the Protocol Exerciser GUI and then click the **Lane settings** tab.



Reverse Lanes: ● Yes ○ No

Automatic Lane Reversal: ○ Enabled ● Disabled

Tx

Physical Lane Polarity:

Lane 0:	● Normal	○ Inverted
Lane 1:	● Normal	○ Inverted
Lane 2:	● Normal	○ Inverted
Lane 3:	● Normal	○ Inverted
Lane 4:	● Normal	○ Inverted
Lane 5:	● Normal	○ Inverted
Lane 6:	● Normal	○ Inverted
Lane 7:	● Normal	○ Inverted
Lane 8:	● Normal	○ Inverted
Lane 9:	● Normal	○ Inverted
Lane 10:	● Normal	○ Inverted
Lane 11:	● Normal	○ Inverted
Lane 12:	● Normal	○ Inverted
Lane 13:	● Normal	○ Inverted
Lane 14:	● Normal	○ Inverted
Lane 15:	● Normal	○ Inverted

NOTE

Refer to the Protocol Exerciser GUI Reference help topics in this online help to get a detailed description of each field in this tab page. Alternatively, click the Help button displayed in the tab page to get a context-sensitive help page.

The number of lanes displayed in the **Lane settings** page depend on the Link Width that you selected in the **Link settings** page.

Lane Reversal

You can enable or disable the lane reversal feature for the Transmitter. Protocol Exerciser can reverse its lanes before starting the link negotiation. Enabling the lane reversal can help you test the DUT’s capability to perform lane ordering during link training.

Lane Polarity Inversion

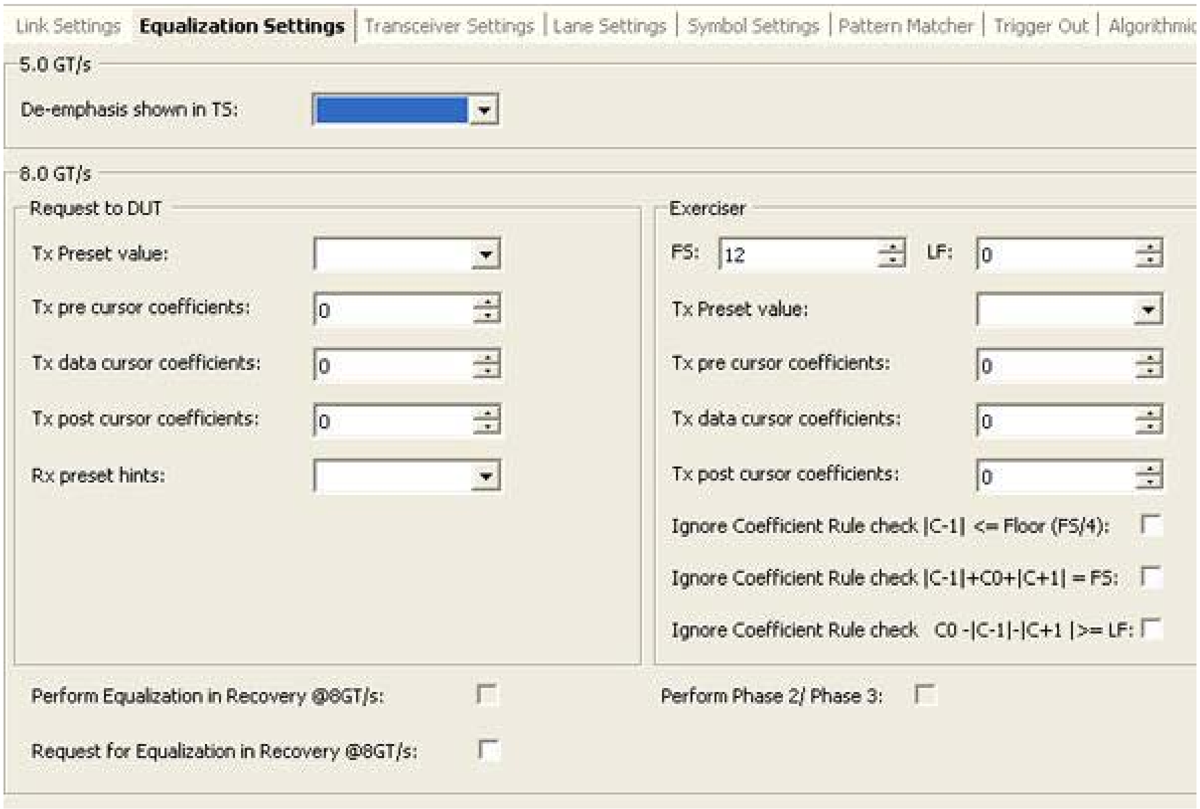
You can set the Lane Polarity inversion for each lane of the link independently. If you select the Polarity inversion for a lane, then at the time of link training, the receiver gets an indication of the lane polarity inversion setting through the transmitted TS1 and TS2 ordered sets. Subsequently, it indicates that the receiver must invert the received data. Setting the Tx lane polarity inversion can help you assess DUT's capabilities to understand and implement polarity inversion during the link training.

Configuring Equalization Procedure Settings

You use the Equalization Settings tab in the General pane to configure the Equalization settings for the Protocol Exerciser Transmitter (Tx). The settings are configured separately for 5.0 and 8.0 link speeds. This topic describes how to enable/disable and configure the equalization settings.

NOTE

If you select the MRIOV protocol for creating a link between Protocol Exerciser and DUT, then the Equalization settings that you specify in this tab do not have any impact.



5.0 GT/s

De-emphasis shown in TS:

8.0 GT/s

Request to DUT

Tx Preset value:

Tx pre cursor coefficients: 0

Tx data cursor coefficients: 0

Tx post cursor coefficients: 0

Rx preset hints:

Exerciser

FS: 12 LF: 0

Tx Preset value:

Tx pre cursor coefficients: 0

Tx data cursor coefficients: 0

Tx post cursor coefficients: 0

Ignore Coefficient Rule check |C-1| <= Floor (FS/4): □

Ignore Coefficient Rule check |C-1|+C0+|C+1| = FS: □

Ignore Coefficient Rule check C0 -|C-1|-|C+1 |>= LF: □

Perform Equalization in Recovery @8GT/s: □

Perform Phase 2/ Phase 3: □

Request for Equalization in Recovery @8GT/s: □

Figure 11 Equalization settings tab

Transmitter Based Equalization (De-emphasis) for 5.0 GT/s speed

For 5.0 GT/s link speed, you can select the Tx de-emphasis option on a per-link basis. You can enable and set the Tx de-emphasis level that the Protocol Exerciser will use for transmission using the De-emphasis shown in TS listbox in the 5.0 groupbox. Typically, the -3.5dB option is used when Protocol Exerciser is acting as a DSC and the -6dB option is used when Protocol Exerciser is acting as a USC.

Equalization for 8.0 GT/s speed

For 8.0 GT/s, all the operational and associated lanes of a link need to participate in the equalization procedure to adjust their transmitter and receiver setups to improve signal quality when moving to 8.0 GT/s. Each combination of Tx, channel, and Rx will have a unique set of coefficients yielding an optimum signal-to-noise ratio. The training sequences exchanged during the link training to 8.0 GT/s consists of negotiating the equalization coefficients to be used, to ensure reliable transfer of data.

For any link training or link speed upgrade to 8.0 GT/s, the link partners have to go through the phases of the Equalization procedure as per the PCIe specifications. The USC and DSC use the following phases to adjust the Transmitter and Receiver setup of each lane to improve the signal quality.

- Phase 1 - Both USC and DSC transmit with the specified preset values. The specified FS and LF values are advertised in phase 1.
- Phase 2 - DSC tunes the USC by requesting a change in the coefficients at the transmitter of the USC.
- Phase 3 - USC tunes the DSC by requesting a change in the coefficients at the transmitter of the DSC.

Protocol Exerciser supports two types of equalization when moving to 8.0 GT/s. These are described in the following sections.

Autonomous equalization

The Autonomous equalization feature is applicable if the link partners are 8.0 GT/s capable during the initial link negotiations and they are moving to 8.0 GT/s from the Detect state. To perform autonomous equalization, Protocol Exerciser acting as a USC autonomously transitions from L0 to Recovery before transmitting any DLLPs. To accomplish the autonomous equalization, the **Autonomous speed change**

to 8 GT/s option in the **Link Settings** tab should be selected. If this option is not selected, the link is not moved to 8.0 GT/s after it gets trained to 2.5 GT/s.

If **Autonomous speed change to 8 GT/s** option is selected, Protocol Exerciser acting as a USC will always perform equalization at the first entry to 8GT/s after exit from the Detect State. If Protocol Exerciser is acting as a DSC, it will respond to the Equalization state transitions of USC.

Software based equalization

The software based equalization feature is applicable if the link is configured at 2.5 or 5.0 GT/s and a link partner requests to move to 8.0 GT/s with both link partners capable of 8.0 GT/s data rate. You can configure Protocol Exerciser to enable/disable software based equalization while moving to 8.0 GT/s from 2.5 or 5.0 GT/s.

Enabling Equalization at 8.0 GT/s

If Protocol Exerciser is acting as USC and you want Protocol Exerciser to perform software based equalization while moving to 8.0 GT/s, ensure that:

- the **Perform Equalization in Recovery @8 GT/s** option is selected in the **Equalization settings** tab. Protocol Exerciser (acting as a USC) then must perform Equalization when upgrading the link speed to 8.0 GT/s. If this option is not selected, then the link speed upgrade to 8.0 GT/s happens without performing equalization.

If Protocol Exerciser acting as a USC is configured to perform equalization and you want to ensure that it performs all the three phases of equalization, then select the **Perform Phase 2/Phase 3** option in the **Equalization settings** tab. Protocol Exerciser (acting as a USC) then must perform Phase 2 and 3 of Equalization when upgrading the link speed to 8.0 GT/s. If this option is not selected, then Protocol Exerciser exits the Recovery.Equalization sub-state after phase 1 and transitions to the Recovery.RcvrLock substate to proceed towards making the link operational at the 8.0 GT/s speed.

If you want to ensure that Protocol Exerciser (acting as a DSC) should request the USC to redo the equalization, then select the **Request Equalization in Recovery @8GT/s** option in the **Equalization settings** tab.

Disabling Equalization at 8.0 GT/s

If you do not want Protocol Exerciser to perform software based equalization while moving to 8.0 GT/s, ensure that:

- the **Perform Equalization in Recovery @8 GT/s** option is deselected in the **Equalization settings** tab. This is applicable if Exerciser is used as a USC.
- the **Perform Phase 2/Phase 3** option is deselected in the **Equalization settings** tab. This is applicable if Exerciser is used as a USC.

If you do not want Protocol Exerciser acting as a DSC to request USC to redo equalization while moving to 8.0 GT/s, then deselect the **Request Equalization in Recovery @8 GT/s** option in the **Equalization settings** tab.

NOTE

Protocol Exerciser needs to perform or respond to the Equalization phases while running some LTSSM tests used to test DUT's LTSSM functions. For these tests, the Equalization settings that you configured in this tab are over-ridden by the equalization settings relevant for the test when the test is running. The configured equalization settings are reverted back when the test execution is over.

For instance, you have not enabled software based equalization and executed the LTSSM test to upgrade the link speed to 8.0 GT/s with equalization. In this situation, the test overrides the disabled equalization setting and performs equalization while upgrading to 8.0 GT/s.

Configuring Equalization settings

You configure the equalization co-efficients, presets, and FS and LF voltage swings for the Protocol Exerciser transmitter using the **Equalization Settings** tab in the **General Settings** navigation pane of the Protocol Exerciser GUI. Protocol Exerciser uses these settings to perform or respond to the Equalization steps while training/retraining, or upgrading the link to the 8.0 GT/s link speed.

NOTE

Refer to the Protocol Exerciser GUI Reference help topics in this online help to get a detailed description of each field in this tab page. Alternatively, click the Help button displayed in the tab page to get a context-sensitive help page.

Setting Equalization values

You can set the:

- FS and LF values that the Protocol Exerciser advertises during the link training for its Tx capabilities using the Exerciser group box.
- Tx De-emphasis preset and co-efficient values that the Protocol Exerciser advertises during the link training for its Tx capabilities using the Exerciser group box.
- Tx De-emphasis preset and co-efficient values that the Protocol Exerciser will send to DUT in the training sequences to request DUT to use the specified presets and co-efficient values using the Request to DUT group box.

NOTE

Protocol Exerciser ignores all the Tx De-emphasis value requests from DUT. To yield an optimum combination of Tx/Rx equalization, it uses the Protocol Exerciser Tx and Rx equalization settings that you set in the Transceiver settings tab.

NOTE

In comparison to a standard PCIe device, Protocol Exerciser can also control the electrical de-emphasis settings that are independent of the settings in the training sequences. This allows you to test the de-emphasis level negotiation protocol without changing the electrical behavior of Protocol Exerciser.

Configuring Transceiver Settings

You can set the Protocol Exerciser Transmitter and Receiver equalization settings using the Transceiver Settings tab. These settings impact the overall signal quality. You should change the default settings in this page if you are facing problems specific to signal quality or you want to test how the changes in these settings impact the overall signal quality.

You can adjust the transceiver settings to yield an optimum combination of Tx/Rx equalization and for better tuning based on your hardware setup.

Using the Transceiver Settings tab, you can adjust the output voltage, pre tap, first post tap, and second post tap for the Protocol Exerciser transmitter and the AC and DC Gain settings for the Protocol Exerciser Receiver.

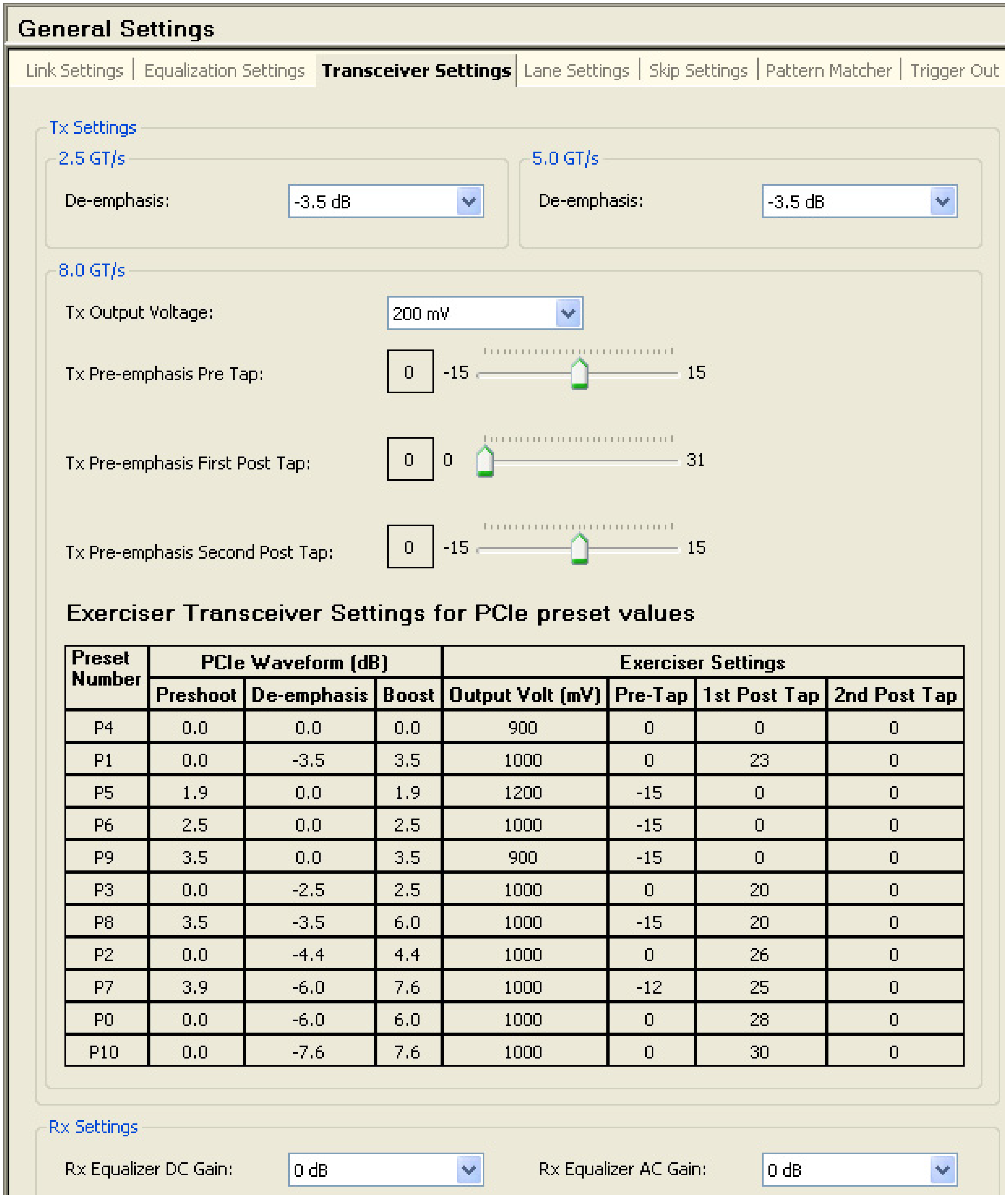
For the Protocol Exerciser Transmitter, you can configure the Transceiver settings for Gen1, Gen2, and Gen3 speeds separately.

The AC and DC Gain settings that you configure for the Protocol Exerciser Receiver are applicable for all the three speeds (Gen1, Gen2, and Gen3).

NOTE

If you select the MRIOV protocol for creating a link between Protocol Exerciser and DUT, then the Transceiver settings that you specify in this tab do not have any impact.

The following figure displays the default Transceiver settings.



Tx Settings

2.5 GT/s

De-emphasis: -3.5 dB

5.0 GT/s

De-emphasis: -3.5 dB

8.0 GT/s

Tx Output Voltage: 200 mV

Tx Pre-emphasis Pre Tap: 0 -15 15

Tx Pre-emphasis First Post Tap: 0 0 31

Tx Pre-emphasis Second Post Tap: 0 -15 15

Exerciser Transceiver Settings for PCIe preset values

Preset Number	PCIe Waveform (dB)			Exerciser Settings			
	Preshoot	De-emphasis	Boost	Output Volt (mV)	Pre-Tap	1st Post Tap	2nd Post Tap
P4	0.0	0.0	0.0	900	0	0	0
P1	0.0	-3.5	3.5	1000	0	23	0
P5	1.9	0.0	1.9	1200	-15	0	0
P6	2.5	0.0	2.5	1000	-15	0	0
P9	3.5	0.0	3.5	900	-15	0	0
P3	0.0	-2.5	2.5	1000	0	20	0
P8	3.5	-3.5	6.0	1000	-15	20	0
P2	0.0	-4.4	4.4	1000	0	26	0
P7	3.9	-6.0	7.6	1000	-12	25	0
P0	0.0	-6.0	6.0	1000	0	28	0
P10	0.0	-7.6	7.6	1000	0	30	0

Rx Settings

Rx Equalizer DC Gain: 0 dB

Rx Equalizer AC Gain: 0 dB

Figure 12 Transceiver Settings tab

NOTE

Refer to the Protocol Exerciser GUI Reference help topics in this online help to get a detailed description of each field in this tab page. Alternatively, click the Help button displayed in the tab page to get a context-sensitive help page.

Configuring Pattern Matchers

Protocol Exerciser allows you to create four pattern matchers. Using a pattern matcher, you can specify a pattern for a TLP header, prefix, and payload data. The specified pattern is compared with the header and payload of incoming TLPs from DUT and if a pattern match is found in a TLP, the configured action is taken. You can configure the following actions to occur in response to a pattern match:

- Configure a trigger out setting which results in generating an event and a trigger out pulse to other test equipments such as logic analyzers or protocol analyzers when the specified pattern match is found in an incoming TLP.
- Configure a packet to wait until the specified pattern match is found in incoming TLPs. Such a packet is then sent as stimulus to DUT only when the pattern match is found.
- Configure a packet to be sent repeatedly until the specified pattern match is found in incoming TLPs. Such a packet is then sent repeatedly as stimulus to DUT until the pattern match is found.

This topic describes how you can configure a pattern match and how you can configure the above-mentioned actions to be performed when a pattern match is found.

Creating a pattern matcher

Components of a pattern matcher

A pattern matcher can contain:

- 4DWs long data string for TLP header matching
- 1DW prefix for TLP prefix matching. This is applicable only for the MRIOV protocol.
- 4DWs to match the first 4DWs in the payload of incoming TLPs.

By default, all the four pattern matchers are disabled. You can enable these pattern matchers individually and then set their components.

You can choose to enable/disable some or all of these components individually in a pattern matcher to define how you want to perform pattern matching on incoming TLPs.

For instance, you can enable only the 4DW long data string for header in a pattern matcher to match just the TLP header.

To create a pattern matcher:

1 Click the **General Settings** icon in the Navigation window of the Protocol Exerciser GUI.

2 Click the **Pattern Matcher** tab.

3 Select the Pattern Matcher from the drop-down list of pattern matchers. Protocol Exerciser allows you to create four pattern matchers (Pattern Matcher 1 to 4).

4 Select the **Enabled** checkbox displayed next to the selected pattern matcher to enable that pattern matcher. By default, all pattern matchers are disabled. On selecting **Enabled**, the fields of the pattern matcher becomes available for editing.

5 Configure the components of the selected pattern matcher to define the matching criteria.

   a Load the required TLP template by selecting a template from the available list displayed next to the Enabled checkbox.

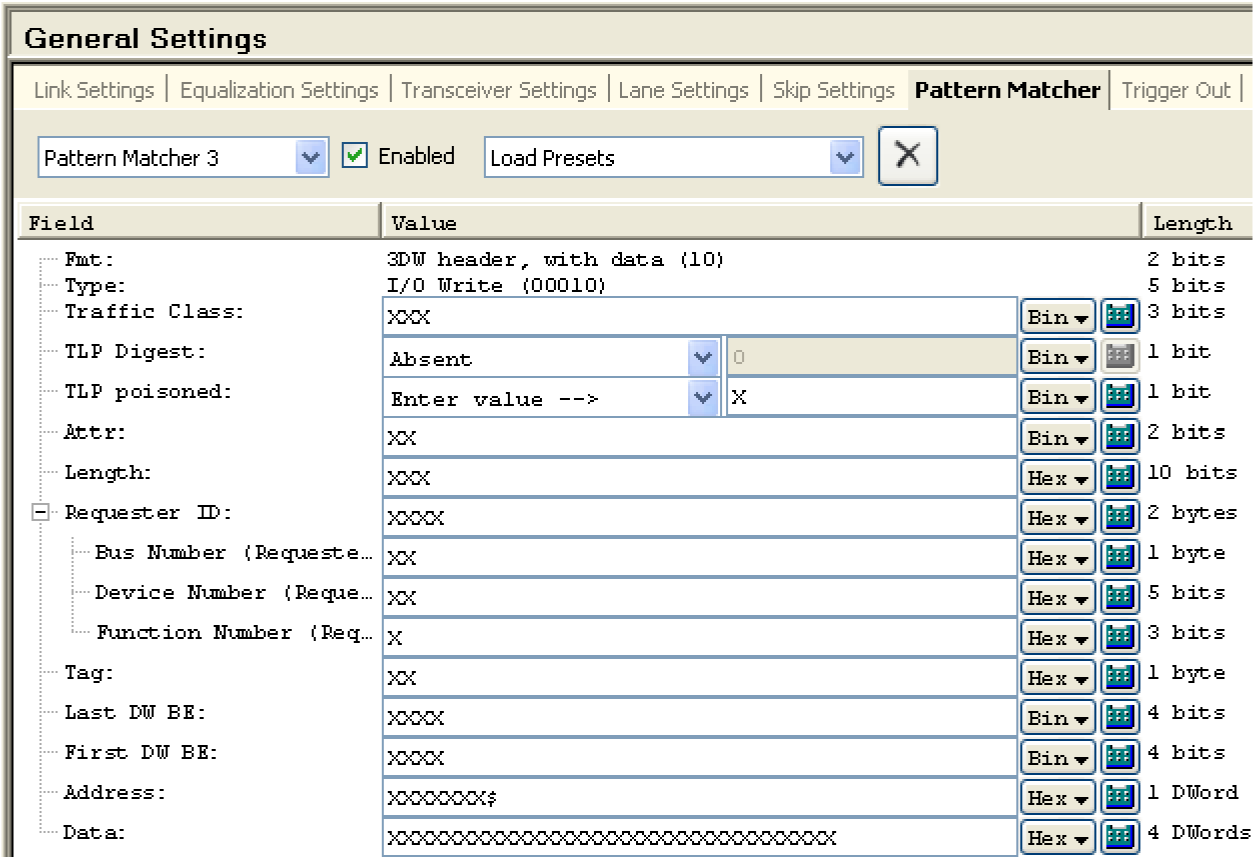
   The fields of the selected TLP template are displayed with the default values.

   b Specify the required values for the fields with which you want to match the fields in the incoming TLPs.

6 Click **Apply**.

Example of a pattern matcher

The following screen displays a pattern matcher. The Pattern Matcher 3 has been enabled. In this pattern matcher, the 3 DWs are specified for TLP header matching. Any incoming I/O Write request with the TLP Digest field marked as Absent in the header will match this pattern matcher.



Pattern Matcher 3 ☑ Enabled Load Presets

Field	Value		Length
Fmt:	3DW header, with data (10)		2 bits
Type:	I/O Write (00010)		5 bits
Traffic Class:	XXX	Bin	3 bits
TLP Digest:	Absent 0	Bin	1 bit
TLP poisoned:	Enter value --> X	Bin	1 bit
Attr:	XX	Bin	2 bits
Length:	XXX	Hex	10 bits
Requester ID:	XXXX	Hex	2 bytes
Bus Number (Requeste...	XX	Hex	1 byte
Device Number (Reque...	XX	Hex	5 bits
Function Number (Req...	X	Hex	3 bits
Tag:	XX	Hex	1 byte
Last DW BE:	XXXX	Bin	4 bits
First DW BE:	XXXX	Bin	4 bits
Address:	XXXXXXX$	Hex	1 DWord
Data:	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	Hex	4 DWords

Configuring an action when a pattern match is found

Configuring a Trigger out

You can configure a trigger out to generate events and trigger other test equipments such as logic analyzers or protocol analyzers when the specified pattern match is found in an incoming TLP.

To set a trigger out on a pattern match:

1 Click the **General Settings** icon in the left navigation pane.

2 Click the **Trigger Out** tab.

3 Select the **Pattern Matcher** checkbox from the **Rule Name** list.

4 Click **Apply**.

Once you enabled a trigger out on Pattern Matcher, Protocol Exerciser displays an event and generates a trigger out pulse when the specified pattern match is found in an incoming TLP. The events are displayed in the **Accumulated Events** list in the **Trigger Out** tab.

Delaying a packet until a pattern match is found

You can delay the transmission of a packet from Protocol Exerciser to DUT until the specified pattern match is found in the incoming TLPs from DUT. To do this:

1. Click the **Traffic Setup** icon in the left navigation pane.
2. Click the **Block Transfers** tab.
3. Double-click the packet that you want to delay until the pattern match is found.

   The **Edit Packet** dialog box is displayed.
4. In the **Wait/Repeat Settings** section, select the **Wait for Pattern Matched** option from the drop-down list.
5. Select from the list of check boxes (1, 2, 3, and 4). These check boxes represent the four pattern matchers available in the General Settings - > Pattern Matcher tab. If you do not select any of these four pattern matcher checkboxes, then the packet will not be delayed.

Selecting these options ensure that the sending of the underlying packet is delayed as long as all the enabled pattern matchers (1, 2, 3, and 4) are not found in the incoming TLPs.

Repeating a packet transmission until a pattern match is found

You can repeat the transmission of a packet from Protocol Exerciser to DUT until the specified pattern match is found in the incoming TLPs from DUT. To do this:

1. Click the **Traffic Setup** icon in the left navigation pane.
2. Click the **Block Transfers** tab.
3. Double-click the packet that you want to repeat until the pattern match is found.

   The **Edit Packet** dialog box is displayed.
4. In the **Wait/Repeat Settings** section, select the **Repeat until Pattern Matched** option from the drop-down list.
5. Select from the list of check boxes (1, 2, 3, and 4). These check boxes represent the four pattern matchers available in the General Settings - > Pattern Matcher tab. If you do

not select any of these four pattern matcher checkboxes, then the packet will not be repeated.

6 In the **Repeat Request** text box, type the number of times you want to repeat the request to send the underlying packet.

Selecting these options ensure that the sending of the underlying packet is repeated as long as all the enabled pattern matchers (1, 2, 3, and 4) are not found in the incoming TLPs.

Configuring External Triggers Settings

Protocol Exerciser can generate events or trigger other test equipments such as logic analyzers or protocol analyzers when a specified trigger condition is met. You can set the trigger conditions using the Trigger Out tab. Once you enable the trigger out conditions, Protocol Exerciser records and displays a corresponding event generating a trigger out pulse when any of these conditions are met. The external trigger out pulse is 125 ns wide. The list of accumulated events is displayed in the **Trigger Out** tab.

NOTE

You can view whether or not an enabled event has occurred and refresh the list of accumulated events in the Trigger out tab by clicking the **Refresh Status** button in this tab.

The following trigger out conditions are available:

- **Protocol Error** - An event is generated on encountering any protocol related error. Refer to Performing Protocol Checks on page 169 to know more about how to enable protocol errors checking.
- **Pattern Matcher** - An event is generated and a trigger out is sent when a specified pattern match is found in an incoming TLP.. Refer to Configuring Pattern Matchers on page 64 to know more about how to create a pattern matcher and use it to generate a trigger.
- **Data Compare Error** - An event is generated on encountering a data compare error. Such an error is generated when an actual byte in the data of a packet does not match the expected byte at a specific address in the data memory of Exerciser during data comparison. Refer to the topic Comparing Actual Data with Expected Data in Memory on page 126 to know more about data comparison.
- **Going Out of L0** - When the LTSSM state transitions out of L0 indicating that the link is not operational, Protocol Exerciser alerts you by generating an event.

The U4305A exerciser card has a component, namely *Trigger Out Connector*, located on the front bracket of the card. This component is used to generate a trigger out pulse from Protocol Exerciser to other test equipment.

NOTE

Refer to the Protocol Exerciser GUI Reference help topics in this online help to get a detailed description of each field in this tab page. Alternatively, click the Help button displayed in the tab page to get a context-sensitive help page.

Configuring Power Management Settings

You can use the Protocol Exerciser GUI to configure the power state transitions for the PCIe link between Exerciser and DUT.

Exerciser supports the L0s and L1 (ASPM and PCI-PM) power management states. You can configure various settings to control when Exerciser should enter and exit from these states and how it behaves in these power management states.

The following section describes how you can configure these two power management states for Exerciser.

Configuring L0s state

If no DLLP or TLP is transmitted for a specified time, then Exerciser can initiate the transition to L0s power state on the transmit side. To ensure that Exerciser can do this initiation to L0s on Tx side, you need to configure the following in the Protocol Exerciser GUI:

- Enable the L0s support for Exerciser. To do this,
  1. Click the **Config space** icon in the navigation pane.
  2. Click the **Function A** tab.
  3. Click the **PCIe Capabilities** tab.
  4. In the **Link Capabilities Register** section, select the **L0s entry supported** option for the **Active State Link PM Support** field.
  5. In the **Link Control Register** section, select the **L0s entry enabled** option for the **Active State PM Control** field.
  6. Click **Apply**.
- Disable the Send FC Update for Exerciser. If this option is not disabled, then Exerciser will continue transmitting flow control update DLLPs at regular intervals and the transition to L0s will not happen. However, you need not do the disabling if the time that you set for sending the FC updates is greater than the time that you set for the link to remain idle for Exerciser to enter the L0s state. To do this,
  1. Click the **Virtual Channel** icon in the navigation pane.
  2. In the **Resend/Update Periods** section, clear the **Send FC Update each (us)** checkbox.

3. Click **Apply**.

- Specify the time interval for which the link should be idle for Exerciser to enter the L0s state. There should be no transmission of TLPs/DLLPs during this time interval. To do this,
  1. Click the **General Settings** icon in the navigation pane.
  2. Click the **Power Management** tab.
  3. In the **L0s** section, specify the time period (in ns) in the **Enter L0s after having seen no TLP/DLLP for (ns)** field.
  4. Click **Apply**.

Exit from L0s

You can specify the time period after which Exerciser should exit from the L0s state. To do this:

1. Click the **General Settings** icon in the navigation pane.
2. Click the **Power Management** tab.
3. In the **L0s** section, select the **Exit L0s when the following timeout (ns) expires** checkbox and specify the time period in the adjacent field.
4. Click **Apply**.

Exerciser can also exit from L0 if it has to transmit a TLP / DLLP or if you change the link speed or retrain the link.

If you do not specify a time period after which Exerciser should exit from the L0s state, then you need to transmit a TLP from Exerciser to make it exit the L0s state.

Configuring L1 state

ASPM L1 state

Exerciser as a downstream component, can initiate the transition to ASPM L1 power state when the link is idle for the specified time. Both the link partners should support the ASPM L1 state. To ensure that Exerciser can do this initiation to ASPM L1, you need to configure the following in the Protocol Exerciser GUI:

- Enable the ASPM L1 support for Exerciser. To do this,
  1. Click the **Config space** icon in the navigation pane.
  2. Click the **Function A** tab.

3. Click the **PCIe Capabilities** tab.
4. In the **Link Capabilities Register** section, select the **L1 supported** option for the **Active State Link PM Support** field. You can also select the **L0s and L1 supported** option to enable the support for both L0s and L1.
5. In the **Link Control Register** section, select the **L1 entry enabled** option for the **Active State PM Control** field. You can also select the **L0s and L1 entry enabled** option to enable entry to both L0s and L1.
6. Click **Apply**.

- Disable the Send FC Update for Exerciser. If this option is not disabled, then Exerciser will continue transmitting flow control update DLLPs at regular intervals and the transition to L1 will not happen. However, you need not do the disabling if the time that you set for sending the FC updates is greater than the time that you set for the link to remain idle for Exerciser to enter the L1 state. To do this,

  1. Click the **Virtual Channel** icon in the navigation pane.
  2. In the **Resend/Update Periods** section, clear the **Send FC Update each (us)** checkbox.
  3. Click **Apply**.

- Specify the time interval for which the link should be idle for Exerciser to enter the L1 state. There should be no transmission of TLPs/DLLPs during this time interval. To do this,

  1. Click the **General Settings** icon in the navigation pane.
  2. Click the **Power Management** tab.
  3. In the **L1** section, specify the time period (in ns) in the **Enter ASPM L1 after having seen no TLP/DLLP for (ns)** field.
  4. Click **Apply**.

PCI-PM L1 state

Exerciser can also initiate the transition to PCI-PM L1 power state in response to the device state transitions (when the device state is changed to non D0). If Exerciser is acting as an upstream component, it can initiate the entry to PCI-PM L1 by sending a config write request to DUT.

If you enabled both ASPM and PCI-PM support, then Exerciser will enter these states as per the configured entry criteria for these states. If Exerciser is in ASPM L1 state and it receive a Config Write request to change the device state, then Exerciser exits out of ASPM L1 and enters PCI-PM L1.

Exit from L1

You can specify the time period after which Exerciser should exit from the L1 (ASPM or PCI-PM) state. To do this:

1. Click the **General Settings** icon in the navigation pane.
2. Click the **Power Management** tab.
3. In the **L1** section, select the **Exit L1 (ASPM or PCI-PM) when the following timeout (ns)** expires checkbox and specify the time period in the adjacent field.
4. Click **Apply**.

If you do not specify a time period after which Exerciser should exit from the L1 state, then

- in case of ASPM L1, you need to transmit a TLP from Exerciser to make it exit the L1 state.
- in case of PCI-PM L1, you need to transmit a Config Write packet from USC to exit from the PCI-PM L1 state.

On exit from the L1 state, Exerciser transitions to Recovery.

NOTE

Refer to the General Settings function - Power Management tab help topic in this online help to get a detailed description of each field in the Power Management tab page. Alternatively, click the Help button displayed in the tab page to get a context-sensitive help page.

4
Initializing and Managing a PCIe Link

Initializing and training a PCIe Link between Protocol Exerciser and DUT 76

Initializing Flow Control 79

Retraining a PCIe Link between Protocol Exerciser and DUT 84

Initiating Link Speed Change on a Configured Link 85

This chapter describes how to initialize and train a PCIe link and how to manage it in terms of retraining it and upgrading or downgrading the link speed.

Initializing and training a PCIe Link between Protocol Exerciser and DUT

Before you can start using Protocol Exerciser to stimulate DUT with PCIe traffic or to test DUT's LTSSM functions like speed changes, you need to initialize the PCIe link between Protocol Exerciser and DUT.

The Hardware Status pane in the Protocol Exerciser GUI displays the current status of the link. Before initializing the link, the status is reflected as **No Link** indicating that the link is not operational.

Before you start

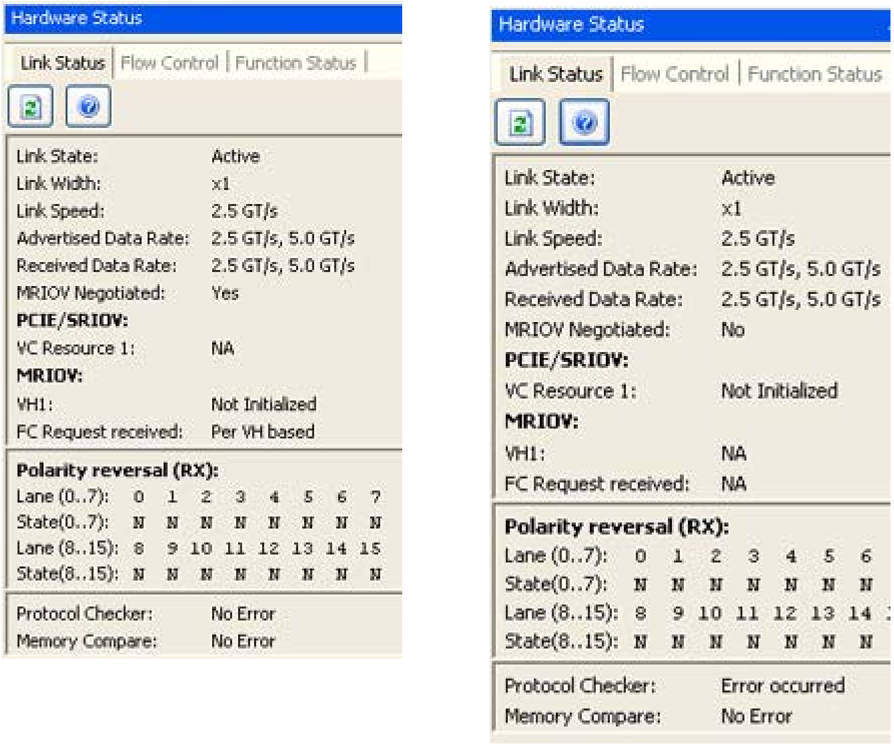
- Make sure you have configured the link, lane, and other settings relevant for link training in a Protocol Exerciser Setup file. You use the General navigation window of the Protocol Exerciser GUI to configure these settings. Refer to the chapter Configuring Protocol Exerciser to know more.
- If you want the link to be MR (Multi-root) enabled, ensure that you select the **MRIOV** protocol in the **General settings -> Link settings** page.
- If you want the link to be as per the SRIOV protocol, ensure that you select the **SRIOV** protocol in the **General settings -> Link settings** page. You should have the **U4305A-023** or **U4305U-023** SRIOV software license to enable SRIOV capabilities for Exerciser.
- If you want Protocol Exerciser to initialize and train the link to 8.0 GT/s autonomously, then ensure that the data rate capability of both the link partners is 8.0 GT/s and you have selected the **Autonomous speed change to 8 GT/s** check box in the Link Settings page. If this check box is not selected, Protocol Exerciser does not autonomously move the link to the 8 GT/s speed and trains it to 2.5 GT/s. You can then later upgrade the speed to 8.0 GT/s through the **Initiate Speed Change** menu option.
- Ensure that you have appropriately configured the flow control settings for Protocol Exerciser. You use the Virtual Channels page to configure these settings. Refer to the topic Initializing Flow Control on page 79 to know more.

Training a PCIe Link

To create/initialize a PCIe link, click the icon on the toolbar. Alternatively, you can also select the **Action** > **Link Up** menu command. This creates a link between the current Protocol Exerciser session and DUT. The link is trained to 2.5 GT/s even if the link partners advertise a higher data rate capability during link training. However, if the data rate capability of both the link partners is 8.0 GT/s and you have selected the **Autonomous Speed change to 8 GT/s** check box in the **Link Settings** page, the link is trained to 8 GT/s.

Verifying PCIe Link Training

You can check the status of the link training action using the Link Status tab of the Hardware Status pane of the GUI. The following screens display the status of the link after the link up.



Link State: Active
Link Width: x1
Link Speed: 2.5 GT/s
Advertised Data Rate: 2.5 GT/s, 5.0 GT/s
Received Data Rate: 2.5 GT/s, 5.0 GT/s
MRIOV Negotiated: Yes
PCIE/SRIOV:
VC Resource 1: NA
MRIOV:
VH1: Not Initialized
FC Request received: Per VH based

Polarity reversal (RX):

Lane (0..7):	0	1	2	3	4	5	6	7
State(0..7):	N	N	N	N	N	N	N	N
Lane (8..15):	8	9	10	11	12	13	14	15
State(8..15):	N	N	N	N	N	N	N	N

Protocol Checker: No Error
Memory Compare: No Error

Link State: Active
Link Width: x1
Link Speed: 2.5 GT/s
Advertised Data Rate: 2.5 GT/s, 5.0 GT/s
Received Data Rate: 2.5 GT/s, 5.0 GT/s
MRIOV Negotiated: No
PCIE/SRIOV:
VC Resource 1: Not Initialized
MRIOV:
VH1: NA
FC Request received: NA

Polarity reversal (RX):

Lane (0..7):	0	1	2	3	4	5	6
State(0..7):	N	N	N	N	N	N	N
Lane (8..15):	8	9	10	11	12	13	14
State(8..15):	N	N	N	N	N	N	N

Protocol Checker: Error occurred
Memory Compare: No Error

Figure 13 Hardware Status After Link Up (For PCIe and MRIOV links)

If you selected MRIOV as the protocol while configuring the link settings, then the MRIOV negotiations are done to make the link MRIOV enabled and the **MRIOV Negotiated** field displays **Yes**. For a PCIe link, the **MRIOV Negotiated** field displays **No**.

If the Exerciser emulates an MRIOV capable component, the virtual hierarchy VH0 of Protocol Exerciser is enabled and initialized by default. The Link status tab therefore displays the status of initialization for the virtual hierarchy VH1 of Protocol Exerciser. Depending on whether or not you have enabled this virtual hierarchy in the Virtual Channels page, the status is displayed.

If the Exerciser emulates a non IOV component, the virtual channel VC0 of Protocol Exerciser is enabled and initialized by default. The Hardware status pane therefore displays the status of initialization for the virtual channel VC1 of Protocol Exerciser. Depending on whether or not you have enabled this virtual channel in the Virtual Channels page, the status is displayed.

NOTE

If you are trying to configure an exerciser card of higher link width (such as x8) to a lower link width (such as x1), then you may face the following issue:

As the receivers of the exerciser card have the 50 OHm termination resistance, the receiver detect logic of the system detects 8 receivers. If during linkup, the system does not get a valid signal on all the lanes where it detected a receiver, the system according to the specification goes into the compliance mode.

This issue can only be worked around by taping off unused lanes, or by using a passive slot-interposer with a smaller edge connector according to the desired link width.

Initializing Flow Control

During the link training between the Protocol Exerciser and DUT, the flow control initialization and negotiation process is completed as per the configured flow control settings of Exerciser. In this process, the default virtual channel/virtual hierarchy of Exerciser gets initialized along with the additional virtual channel /hierarchy that you enabled in flow control settings. After getting initialized, these can be used for the TLP traffic flow. You use the **Virtual Channel** page of the Protocol Exerciser GUI to configure the flow control initialization settings for Protocol Exerciser.

Defining advertised header and data credit limits

You assign the flow control credits to the virtual channels that Protocol Exerciser supports. Protocol Exerciser checks the flow control credits available for each virtual channel before it passes TLPs through that virtual channel. In the absence of the available credit limit, no packets can be sent through the virtual channel. You can define how you want to distribute the flow control credit limit of Exerciser between its two virtual channels (VCs). You can define how the credit limit allocated to each VC should be divided for header and payload of Posted and Not Posted requests and Completions. Protocol Exerciser advertises the defined credit limits during the flow control initialization process.

Initializing flow Control for a non IOV / SRIOV capable PCIe component

As a non IOV / SRIOV capable component, Protocol Exerciser supports two virtual channels for which you need to define the flow control settings. These virtual channels have the following naming and numbering convention.

Virtual channel name	Virtual channel number
VC Resource 0	VC0 This number is fixed and cannot be changed.
VC Resource 1	VC (x) where x is the number that you assign to this second VC resource of Exerciser using the Virtual Channel page in the Protocol Exerciser GUI.

VC0 is enabled by default. If required, you can enable or disable the second VC resource 1 - VC (x).

Out of these two supported VCs, Protocol Exerciser determines which VC should be used to pass a particular stimulus request based on the following settings:

- The traffic class (TC) of the stimulus request that you specified using the Traffic Setup page.
- The function (A, B, C, D, or E) to which you added this request using the Traffic Setup page.
- The TC VC mapping that you defined for the function to which you added the request (using the Virtual Channel page).

For instance, if you added a stimulus request TLP to Function A, assigned traffic class 6 to this request, and mapped traffic class 6 to VC0 for Function A, then this request is passed through VC0.

To configure the flow control initialization settings for virtual channels of a non IOV Exerciser, refer to the Virtual Channel function on page 237 GUI reference topic.

Initializing flow control for an MRIOV capable component

As an MRIOV capable component, Protocol Exerciser supports five virtual hierarchies and one virtual link. The default virtual hierarchy VH0 is enabled by default. If required, you can enable or disable the virtual hierarchy VH1 to VH4. Each of these virtual hierarchies supports the VC resource 0 (VC0) and virtual link VL0 of Protocol Exerciser. You need to define the flow control settings for all these hierarchies.

Types of flow controls

For an MRIOV capable Exerciser, you can decide which type of flow control should be used from the following two options.

per VH flow control

In this type of flow control, the flow control is defined at the VH level. If there are for instance, five virtual hierarchies, then five flow controls are implemented.

The traffic of all the hierarchies pass through VC0 in this type of flow control and all the completions that you added to completion queues (Queue0 to Queue4) also pass through the mapped VC0. This type of flow control is applicable if you have enabled it at the Exerciser's end and the DUT also supports it. If it is not supported/enabled at either the Exerciser or DUT's end, then the VL based flow control is used.

VL based flow control

If per VH flow control is disabled or not supported at either end (Exerciser or DUT), then the VL based flow control is applicable. In this type of flow control, the flow control is defined at the Virtual Link (VL) level. All the virtual hierarchies have a common flow control. As an MRIOV capable component, Exerciser supports only one VL and this VL is mapped to VC0 for all the virtual hierarchies. This means that only one VC resource (VC0) is used when the VL based flow control is applicable. All the traffic of all the hierarchies then pass through VC0 ignoring the second VC resource VC (x). The following mapping is implemented between the virtual channels and virtual hierarchies to define this type of flow control.

- Virtual Channel Resource 0 (VC0) mapped to (VL0, VH0, VC0)
- Virtual Channel Resource 0 (VC0) mapped to (VL0, VH1, VC0)
- Virtual Channel Resource 0 (VC0) mapped to (VL0, VH2, VC0)

In this type of flow control, all the completions that you added to completion queues (Queue0 to Queue4) are passed through VC0.

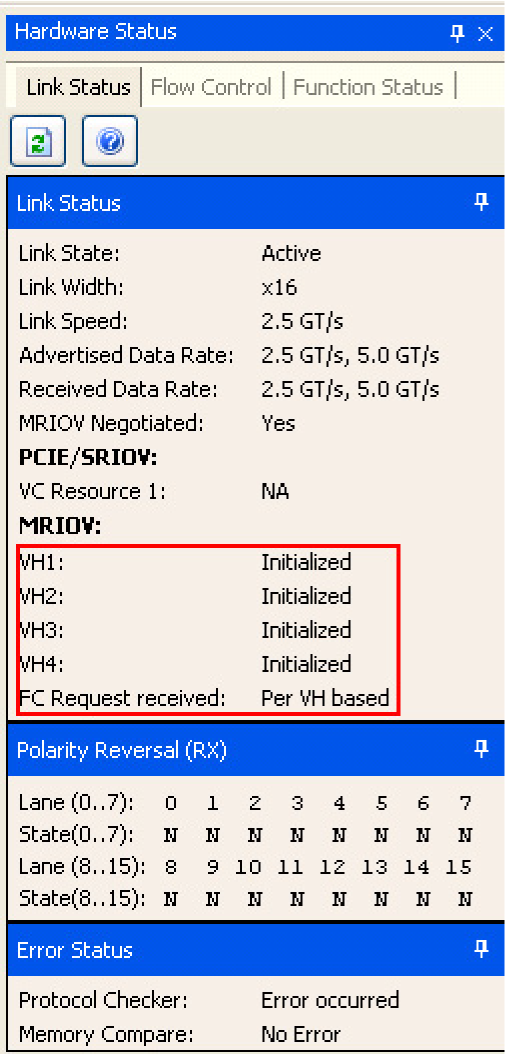
To configure the flow control initialization settings for virtual channels of an MRIOV capable Exerciser, refer to the Virtual Channel function on page 237 GUI reference topic.

Viewing the Flow Control Initialization status

You can view the status of the flow control initialization process for Protocol Exerciser using the Link Status tab of the Hardware Status pane. This pane displays:

- whether or not the VC_Resource 1 (VCx) has initialized. VCx gets initialized only if Exerciser emulates a non IOV PCIe component and you have enabled VCx in the Virtual Channel flow control settings. If you have not enabled VCx in the flow control settings, then the status of this resource is displayed as Not Initialized. VC0 gets initialized by default.

- whether or not the virtual hierarchies (VH1 to VH4) that you enabled have initialized. These hierarchies get initialized if Exerciser emulates an MRIOV capable PCIe component and you have enabled these hierarchies in the flow control settings. If you have not enabled these in the flow control settings, then their status is displayed as Not Initialized. VH0 gets initialized by default.
- the type of flow control requested during the initialization process in case of an MRIOV capable Exerciser. Based on the type of flow control you selected in the flow control settings, VL based or per VH based flow control is requested and displayed in the status.



Link State: Active
Link Width: x16
Link Speed: 2.5 GT/s
Advertised Data Rate: 2.5 GT/s, 5.0 GT/s
Received Data Rate: 2.5 GT/s, 5.0 GT/s
MRIOV Negotiated: Yes

PCIE/SRIOV:

VC Resource 1: NA

MRIOV:

VH1: Initialized
VH2: Initialized
VH3: Initialized
VH4: Initialized
FC Request received: Per VH based

```
Lane (0..7):   0  1  2  3  4  5  6  7
State(0..7):   N  N  N  N  N  N  N  N
Lane (8..15):  8  9 10 11 12 13 14 15
State(8..15):  N  N  N  N  N  N  N  N
```

Protocol Checker: Error occurred
Memory Compare: No Error

NOTE

You can also view the current status of the flow control counters (allocated, used, and remaining limits) of Protocol Exerciser using the **Flow Control** tab in the **Hardware status** pane. Refer to the topic Starting the Stimulus and Viewing Results on page 143 to know more.

The flow control credits of Exerciser decreases when Exerciser sends a packet and it increases when the DUT frees buffer by sending a flow control update packet.

Retraining a PCIe Link between Protocol Exerciser and DUT

Once you have initialized the PCIe link between Protocol Exerciser and DUT, you can start using Protocol Exerciser. As you progress with the DUT testing, you may need to retrain this link between DUT and Protocol Exerciser, for instance, in case of link errors or if during an LTSSM test execution, the advertised or received data rate do not match the data rate capabilities.

Consider a situation when you have changed the speed of a configured link from 5.0 GT/s to 2.5 GT/s. Consequently, the advertised data rate is also reflected as 2.5 GT/s. You can then retrain the link to ensure that the advertised and received data rate are restored back to the set 2.5 and 5.0 GT/s capabilities.

To retrain a PCIe link, click the icon on the toolbar. Alternatively, you can also select the **Action** > **Initiate Retrain Link** menu command.

Protocol Exerciser then retrains the link by moving it to recovery and then back to the Operational state. The link is retrained/recovered as per the link and lane settings negotiated during the initial link training sequence. For example, the link is trained to 2.5 GT/s and the data rate capabilities of DUT and Protocol Exerciser are 5.0 GT/s. By initiating the link retraining, the initial data rate – 2.5 GT/s is used to retrain the link. The Advertised and Received data rate capabilities are updated based on the current data rate capabilities of the link partners.

You can verify if the link is retrained successfully by viewing the status of the link in the Hardware Status pane of the Protocol Exerciser GUI. On the successful retraining, the link status should be Active and the link speed, link width, advertised and received data rate and lane polarity reversal are as per the initial link and lane negotiations.

Initiating Link Speed Change on a Configured Link

As you progress with the DUT testing on a configured link, you may need to change the link speed to test the DUT's capabilities in a higher/lower speed scenarios.

To change the link speed on a configured link, click the ↑↓ icon on the toolbar. Alternatively, you can also select the **Action** > **Initiate Speed Change** menu command.

Protocol Exerciser then initiates link speed change to the highest data rate advertised by both the link partners during link training.

If the link is already at the highest data rate supported by both the link partners and you want to downgrade the link speed, then you first need to downgrade the data rate capabilities of a link partner before starting speed change. In such a situation, the link speed is downgraded based on the changed data rate capability.

You can verify if the link speed has been changed successfully by viewing the status of the link speed in the Hardware Status pane of the Protocol Exerciser GUI.

5
Testing DUT's LTSSM Functions

LTSSM 88

LTSSM Tester - Introduction 90

When to Use LTSSM Tester 92

How LTSSM Tester Works 94

Getting Started with LTSSM Tester 97

Testing Link Initialization, Training, and Management 100

This chapter provides detailed information on Protocol Exerciser's use as an LTSSM to test a DUT's LTSSM functions. This chapter describes Protocol Exerciser's test tool - LTSSM Tester and how to use it to perform LTSSM testing.

LTSSM

Link Training and Status State Machine (LTSSM) is the sub-block that drives and controls the link initialization, and training process for a PCIe device to enable the normal data exchange between the two devices over the link. LTSSM operates at the physical layer level and exchanges physical layer packets (Ordered sets such as TS1 and TS2) to initialize, train, and manage the link.

LTSSM Functions

The following are some of the key functions of LTSSM that are referred to later in this chapter in the context of LTSSM testing.

- Link initialization and configuration to bring the link to an operational state when TLP and DLLPs can be transmitted over it.
- Link recovery or retraining in situations such as speed changes, power management, or recovering from a link error.
- Performing link bandwidth management by downgrading or upgrading the link speed in response to a link speed change request.
- For PCIe Gen3 devices, performing the Equalization procedure before reaching the Gen3 (8.0 GT/s) speed during the link training or retraining.

LTSSM States

LTSSM transits through various states and substates during link initialization, training, and management. The entry and exit of each of these states and the exchange of packets (Ordered sets) between the two devices during each state is as per the PCIe specifications.

As per the PCIe specifications, LTSSM has 11 states and further substates. These states are referred to frequently in this chapter to describe LTSSM testing using the Agilent LTSSM Tester tool. Following is a list of these states followed by a diagram to illustrate the sequence of these states as per the PCIe specifications.

- Detect
- Quiet

- Polling
- Configuration
- L0
- Recovery
- L0s
- L1
- L2
- Hot Reset
- Disabled
- Loopback

The following figure displays the top-level states of LTSSM.

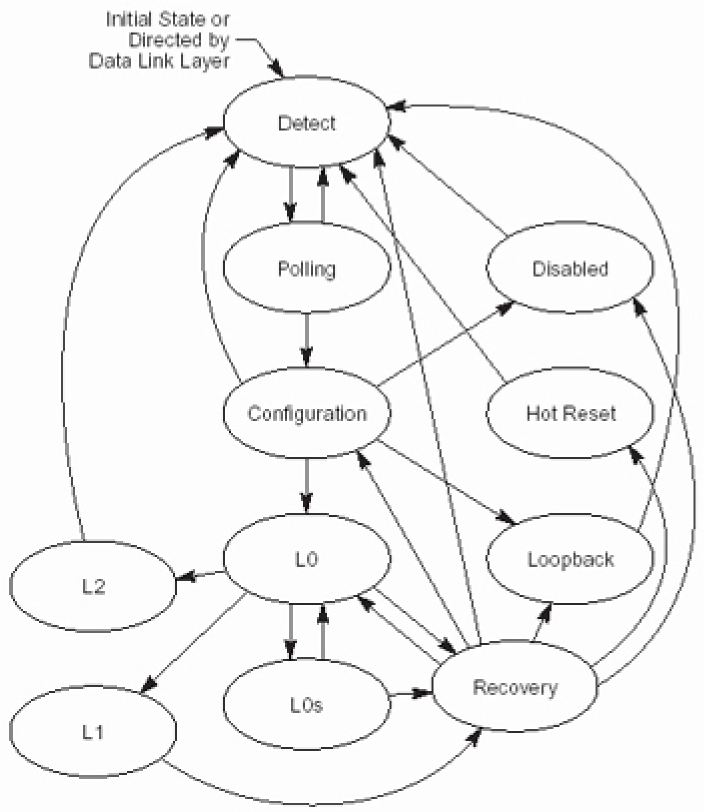


Figure 14 LTSSM States

NOTE For information on different LTSSM states and sub-states, refer to *PCI Express specification*.

LTSSM Tester - Introduction

LTSSM is one of the key challenges when moving from PCI Express Gen2 to Gen3 with link testing for Gen3 as one of the biggest validation challenges.

Agilent LTSSM Tester is a tool that helps you perform link negotiations testing effectively and thoroughly to test a DUT’s LTSSM functions. Using this tool, you can verify all the LTSSM state transitions of DUT and check if all the state timeouts are correctly implemented. This tool provides you a set of predefined LTSSM tests to help you assess if the DUT’s LTSSM capabilities are as per the LTSSM rules specified in the PCIe specifications.

The LTSSM Tester tool is integrated with Protocol Exerciser and is available as a tab in the Protocol Exerciser GUI. When you install and configure the Agilent’s Protocol Exerciser card, the card emulates a PCIe device and provide the LTSSM functionality as well to stimulate the DUT’s LTSSM functions.

Protocol Exerciser exchanges appropriate physical layer packets (ordered sets) with the DUT as per the PCIe specifications to test DUT’s LTSSM capabilities. You can access the LTSSM functions provided by Protocol Exerciser by running the appropriate LTSSM tests using the LTSSM Tester tool. The test results help you assess the DUT’s capabilities in the context of a particular LTSSM function.

Protocol Exerciser as a USC and a DSC in context of LTSSM testing

Protocol Exerciser can act as an Upstream Component (USC) or a Downstream Component (DSC) with DUT for LTSSM testing. Protocol Exerciser functions as a USC or a DSC depending on the session type that you selected based on the Exerciser card setup. You use the Link Settings page of the General tab of the Protocol Exerciser GUI to select the session type.

As a USC, Protocol Exerciser emulates a root complex and controls the link initialization, training, and management sequences for self-initiated or DUT initiated requests. DUT, in this case, acts as a DSC and responds to the sequences.

As a DSC, Protocol Exerciser emulates an end point and only responds to the LTSSM function sequences performed by the DUT acting as a USC in this case.

LTSSM Tester Features

The LTSSM Tester tool provides various features that make it a useful tool to perform the DUT’s LTSSM testing. Some of the key features of LTSSM Tester are listed below.

- Can explicitly test or force LTSSM state transitions.
- Can monitor state transitions and follow timeout controlled process to ensure protocol compliance.
- Supports scalable speed transitions varying from 2.5 GT/s to 8.0 GT/s.
- Supports LTSSM testing for Gen1, Gen2, and Gen3 PCIe devices.
- Supports lane polarity inversion setting for all the lanes independently.
- Supports link widths ranging from x1, x4, x8 to x16 (any combinations allowed). X2 is not supported.
- Supports automatic or manual enabling /disabling of the lane reversal feature to help you test the DUT’s lane reversal feature.
- Supports data scrambling settings.
- Can test link initialization and retraining with or without performing the Equalization procedure.
- Captures and displays the LTSSM state transitions along with the timestamps for:
  - fast and easy debug and
  - verification to ensure that DUT's LTSSM state transitions and timeouts are as per the PCIe specifications.
- Can initiate link transition to recovery.
- Can create corner conditions to test DUT's error responses.
- Supports Equalization Settings to improve signal quality for DUT and Exerciser.
- Supports Transceiver Settings for hardware connected for variable voltages.
- Supports displaying the protocol errors violated while linking, retraining, recovering or speed change.

When to Use LTSSM Tester

LTSSM Tester provides a set of LTSSM tests to help you test various aspects of DUT's LTSSM. The following list provides you pointers to the broad areas/situations when these tests can help you in LTSSM testing.

To test and assess if:

- The DUT is able to perform or respond to the link initialization and training process at the negotiated link speed and width.
- The DUT's LTSSM state transitions are as per the LTSSM rules specified in the PCIe specifications.
- The DUT's LTSSM transits to recovery and back to the operational state for a link retraining request (for instance, link errors or link speed changes).
- The DUT can successfully transit from Gen2 to Gen3 in terms of higher link speeds, new training sequences for the equalization procedure, and other PCIe specifications which are new for Gen3.
- The DUT autonomously changes the link speed to 8.0 GT/s after initiating the link to 2.5 GT/s while performing the Equalization procedure.
- The DUT can perform a downgrade or an upgrade link speed in response to a self initiated or a link partner initiated speed change request on a configured link. This can help you assess how well the DUT handles link bandwidth management.
- The DUT can handle link and lane settings negotiated during the link training process and can successfully train the link with these negotiated or discovered settings. For instance, lane polarity inversion is selected and advertised during the link training. You can test if the DUT‘s LTSSM initializes the link with lane polarity inversion.

LTSSM Supported States

LTSSM has 11 states with further substates. LTSSM transits through these states while performing the link initialization, training, and management.

In this release, LTSSM Tester supports the LTSSM states transitions except the following states or a state feature:

- Multi-link configuration

- Polling compliance testing
- Link power management states
- Direct to Hot Reset or Disabled state
- Loopback

How LTSSM Tester Works

This topic describes the overall working of the LTSSM Tester.

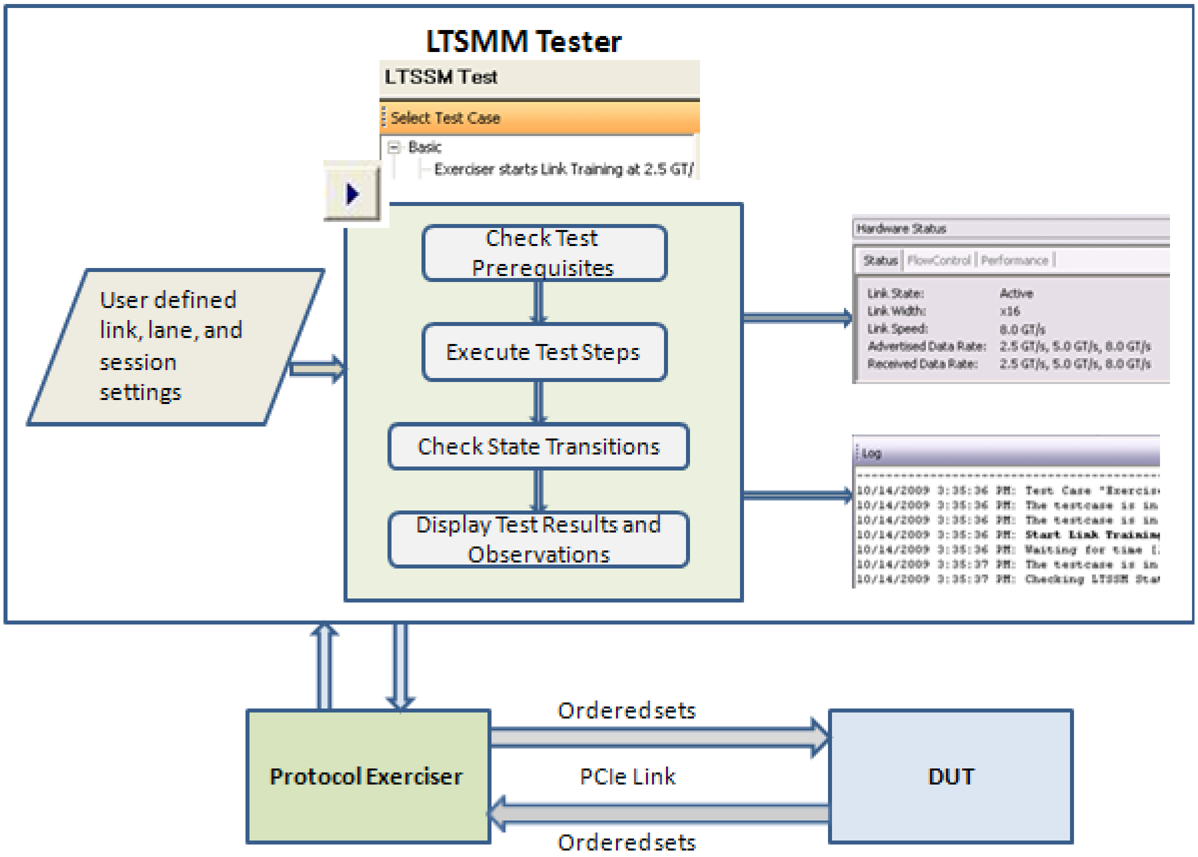
All the LTSSM tests are available in the LTSSM Tests tab of the Protocol Exerciser GUI. When a user runs an LTSSM test, the LTSSM Tester tool executes that test’s steps taking all the required link, lane, equalization, and other settings as inputs. The test execution and observations depend on whether the Protocol Exerciser is acting as a USC or a DSC (based on the user-configured session type). Most of the tests have some prerequisites that need to be fulfilled. LTSSM Tester checks if the test prerequisites are met without which the test execution fails.

When LTSSM Tester starts executing the test, appropriate ordered sets are exchanged between the Protocol Exerciser and DUT as per the PCIe specifications. During the test execution, Protocol Exerciser transits through various applicable LTSSM states and forces the DUT to make the changes to its LTSSM states so that you can verify the state transitions. By looking at the responses from the DUT, Protocol Exerciser checks if the DUT has taken the correct arcs in the state-machine and made the correct transitions or not. LTSSM Tester captures and displays the state transitions in a test log to help you identify the source of problem (if any) and debug when the test stops.

As a USC, Protocol Exerciser performs all the necessary steps of the LTSSM function for which the test is executed. As a DSC, it responds to the steps of the LTSSM functions that DUT performs for that test.

When the test execution completes, LTSSM Tester updates the status of the link, lane, and other relevant settings that are impacted by the test execution. These updates are displayed in the Hardware Status window of the Protocol Exerciser GUI. The test result and the applicable LTSSM state transitions (expected and actual) are displayed in the test log pane of the GUI. If an actual state transition is different from the expected state transition, the test fails notifying the user about the state at which it failed. LTSSM Tester also displays the timestamp for each state transition to indicate the time taken to transit from one state to another.

The following diagram illustrates the working of LTSSM Tester.



Link State: Active
Link Width: x16
Link Speed: 8.0 GT/s
Advertised Data Rate: 2.5 GT/s, 5.0 GT/s, 8.0 GT/s
Received Data Rate: 2.5 GT/s, 5.0 GT/s, 8.0 GT/s

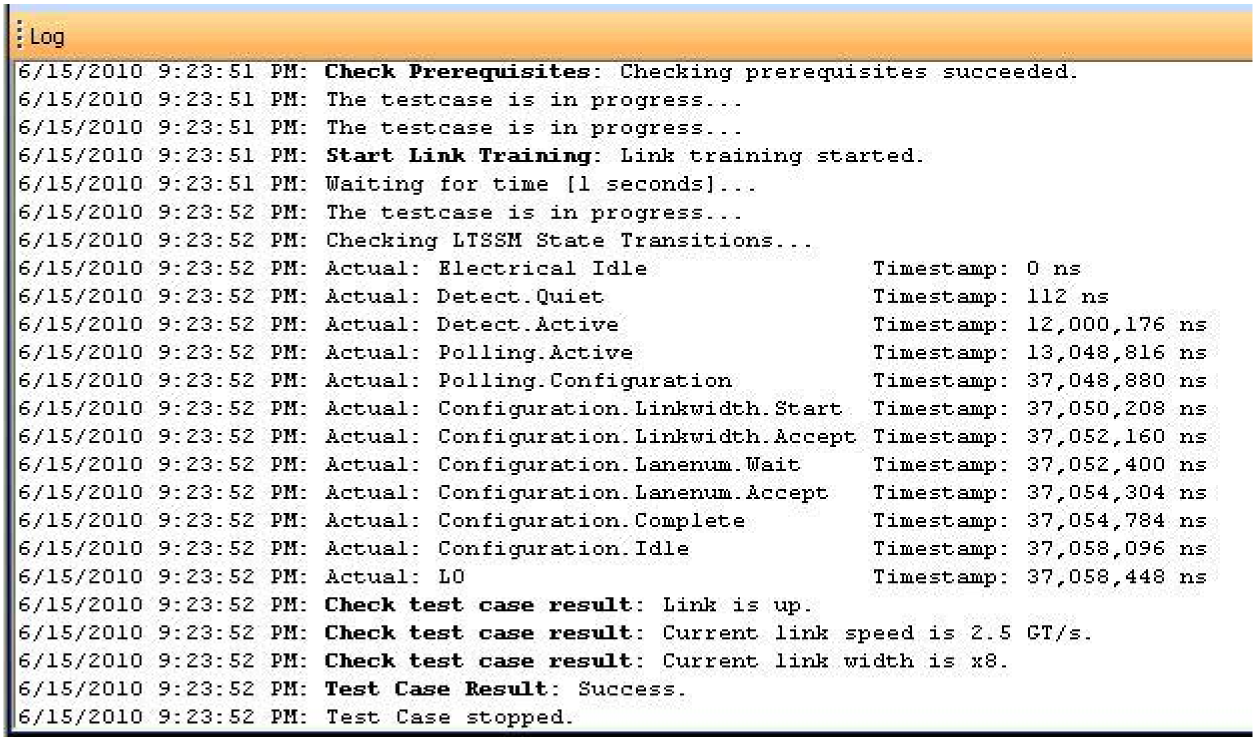
```
------------------------------------------------------------
10/14/2009 3:35:36 PM: Test Case "Exercis
10/14/2009 3:35:36 PM: The testcase is in
10/14/2009 3:35:36 PM: The testcase is in
10/14/2009 3:35:36 PM: Start Link Training
10/14/2009 3:35:36 PM: Waiting for time [
10/14/2009 3:35:37 PM: The testcase is in
10/14/2009 3:35:37 PM: Checking LTSSM Sta
```

Figure 15 How LTSSM Tester Works

Consider an example of an LTSSM test to initialize and train the link to 2.5 GT/s with Protocol Exerciser acting as a USC and DUT as a DSC.

When you execute the Link training test, LTSSM Tester checks the data rate capability of the Protocol Exerciser, selected link width, lane settings, and other relevant settings. Protocol Exerciser then starts performing the link training sequence. It exchanges training sequences with DUT to negotiate link parameters such as link speed, link width, lane polarity, link number, and scrambler enabled or disabled. Protocol Exerciser trains the link initially to 2.5 GT/s even if the link partners (Protocol Exerciser and DUT) are capable of transmitting at higher data rates. When the test execution completes, the link status (No link or Active), negotiated link speed and advertised and received data rate capabilities of Protocol Exerciser and DUT respectively are displayed in the Hardware Status window of the Protocol Exerciser GUI. Once the link is trained to 2.5 GT/s, the

LTSSM state transitions to L0 indicating that the link is operational. After this, Protocol Exerciser handles any upgrade to speed by retraining the link as a response to a self-initiated or DUT initiated speed change request. The following screen displays a sample test log generated for such a test with all the LTSSM state transitions while initiating the link to 2.5 GT/s.



```
6/15/2010 9:23:51 PM: Check Prerequisites: Checking prerequisites succeeded.
6/15/2010 9:23:51 PM: The testcase is in progress...
6/15/2010 9:23:51 PM: The testcase is in progress...
6/15/2010 9:23:51 PM: Start Link Training: Link training started.
6/15/2010 9:23:51 PM: Waiting for time [1 seconds]...
6/15/2010 9:23:52 PM: The testcase is in progress...
6/15/2010 9:23:52 PM: Checking LTSSM State Transitions...
6/15/2010 9:23:52 PM: Actual: Electrical Idle               Timestamp: 0 ns
6/15/2010 9:23:52 PM: Actual: Detect.Quiet                  Timestamp: 112 ns
6/15/2010 9:23:52 PM: Actual: Detect.Active                 Timestamp: 12,000,176 ns
6/15/2010 9:23:52 PM: Actual: Polling.Active                Timestamp: 13,048,816 ns
6/15/2010 9:23:52 PM: Actual: Polling.Configuration         Timestamp: 37,048,880 ns
6/15/2010 9:23:52 PM: Actual: Configuration.Linkwidth.Start  Timestamp: 37,050,208 ns
6/15/2010 9:23:52 PM: Actual: Configuration.Linkwidth.Accept Timestamp: 37,052,160 ns
6/15/2010 9:23:52 PM: Actual: Configuration.Lanenum.Wait     Timestamp: 37,052,400 ns
6/15/2010 9:23:52 PM: Actual: Configuration.Lanenum.Accept   Timestamp: 37,054,304 ns
6/15/2010 9:23:52 PM: Actual: Configuration.Complete        Timestamp: 37,054,784 ns
6/15/2010 9:23:52 PM: Actual: Configuration.Idle            Timestamp: 37,058,096 ns
6/15/2010 9:23:52 PM: Actual: L0                            Timestamp: 37,058,448 ns
6/15/2010 9:23:52 PM: Check test case result: Link is up.
6/15/2010 9:23:52 PM: Check test case result: Current link speed is 2.5 GT/s.
6/15/2010 9:23:52 PM: Check test case result: Current link width is x8.
6/15/2010 9:23:52 PM: Test Case Result: Success.
6/15/2010 9:23:52 PM: Test Case stopped.
```

Figure 16 Sample LTSSM test log

Getting Started with LTSSM Tester

LTSSM Tester is integrated with Protocol Exerciser and all LTSSM functions are available as a tab in the Protocol Exerciser GUI.

Accessing LTSSM Tester

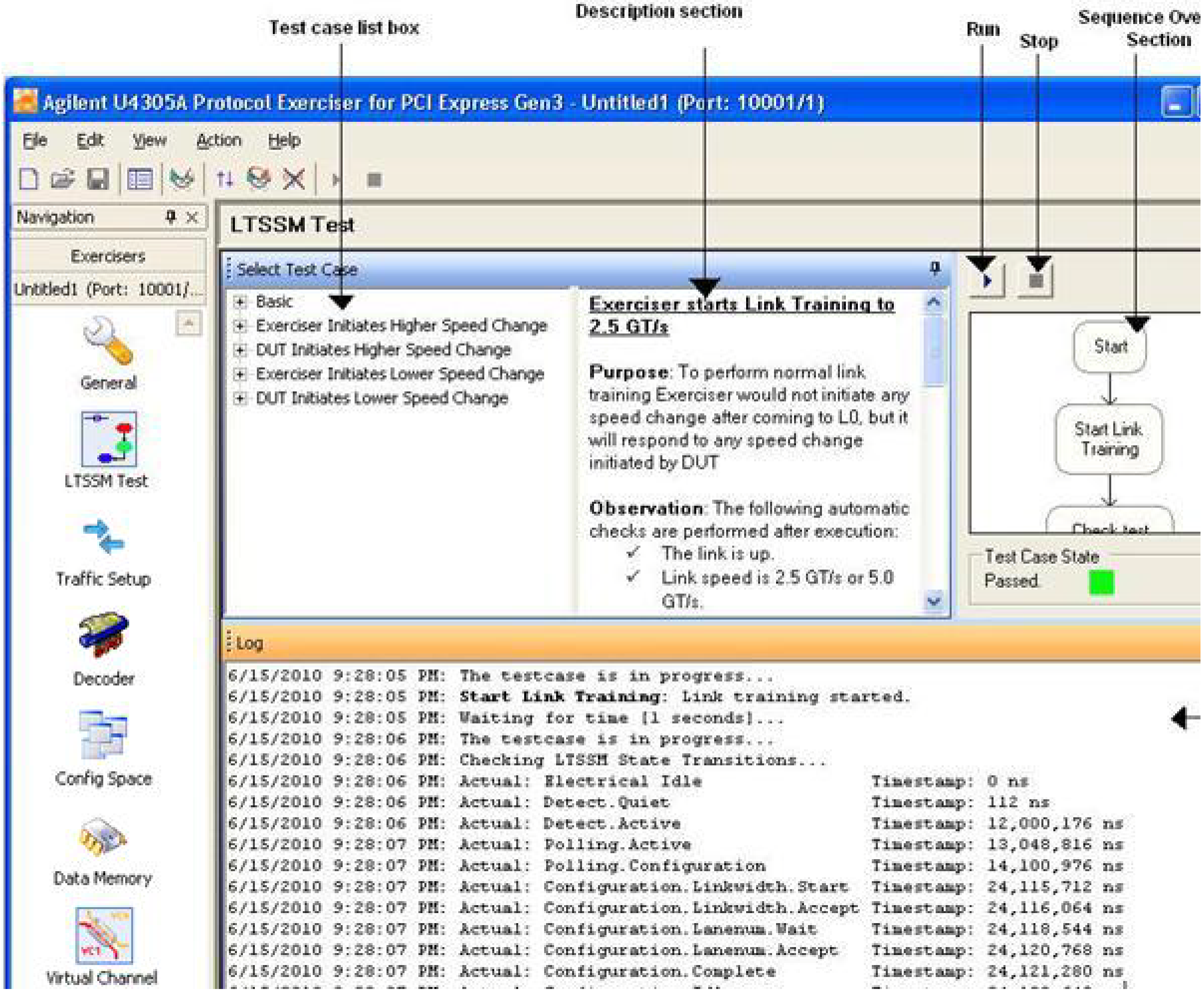
You can access LTSSM Tester from the Navigation window of the Protocol Exerciser GUI.

To access LTSSM Tester

1 Start the Protocol Exerciser GUI. For information on starting the Protocol Exerciser GUI, refer to Starting and Exiting the Protocol Exerciser GUI on page 36.

2 Click the **LTSSM Test** icon in the Navigation window.

The **LTSSM Test** screen is displayed in the main Protocol Exerciser window.



Exerciser starts Link Training to 2.5 GT/s

Purpose: To perform normal link training Exerciser would not initiate any speed change after coming to L0, but it will respond to any speed change initiated by DUT

Observation: The following automatic checks are performed after execution:

✓ The link is up.

✓ Link speed is 2.5 GT/s or 5.0 GT/s.

```
6/15/2010 9:28:05 PM: The testcase is in progress...
6/15/2010 9:28:05 PM: Start Link Training: Link training started.
6/15/2010 9:28:05 PM: Waiting for time [1 seconds]...
6/15/2010 9:28:06 PM: The testcase is in progress...
6/15/2010 9:28:06 PM: Checking LTSSM State Transitions...
6/15/2010 9:28:06 PM: Actual: Electrical Idle              Timestamp: 0 ns
6/15/2010 9:28:06 PM: Actual: Detect.Quiet                 Timestamp: 112 ns
6/15/2010 9:28:06 PM: Actual: Detect.Active                Timestamp: 12,000,176 ns
6/15/2010 9:28:07 PM: Actual: Polling.Active               Timestamp: 13,048,816 ns
6/15/2010 9:28:07 PM: Actual: Polling.Configuration        Timestamp: 14,100,976 ns
6/15/2010 9:28:07 PM: Actual: Configuration.Linkwidth.Start  Timestamp: 24,115,712 ns
6/15/2010 9:28:07 PM: Actual: Configuration.Linkwidth.Accept Timestamp: 24,116,064 ns
6/15/2010 9:28:07 PM: Actual: Configuration.Lanenum.Wait     Timestamp: 24,118,544 ns
6/15/2010 9:28:07 PM: Actual: Configuration.Lanenum.Accept   Timestamp: 24,120,768 ns
6/15/2010 9:28:07 PM: Actual: Configuration.Complete         Timestamp: 24,121,280 ns
```

Figure 17 LTSSM Test screen

All the supported LTSSM tests are grouped under the LTSSM Tests listbox. On clicking a test, its description and sequence of execution steps are displayed in the Description and Sequence panes respectively.

The Log pane provides you details of the steps that the LTSSM Tester is performing during the test execution. It displays the prerequisites that are checked, the steps that are performed, and the LTSSM states that the LTSSM Exerciser transits during the test execution. Based on whether all the test prerequisites are met and whether all the LTSSM state transitions are as expected, the Log pane displays test result as Success or Failure.

NOTE

Refer to the Protocol Exerciser GUI Reference help topics in this user guide to get a detailed description of each field in the LTSSM Test page. Alternatively, click the Help button displayed in the tab page to get a context-sensitive help page.

Prerequisites for LTSSM Testing

- Ensure that all the test specific prerequisites are met before executing the test. Each test has some prerequisites which are listed in the LTSSM Tests pane.
- Ensure that all the link and lane settings are configured using the General tab of the Protocol Exerciser GUI. Protocol Exerciser uses these settings for link negotiations during the link training sequence.
- Ensure that the Equalization and Transceiver settings (coefficients and presets) are appropriately configured for tests that perform or respond to the equalization procedure.
- Ensure that the DUT is connected to the Protocol Exerciser.

Testing Link Initialization, Training, and Management

LTSSM Tests

LTSSM Tester provides a set of predefined LTSSM tests that you can run from the **LTSSM Test** tab of the Protocol Exerciser GUI.

To run a test, click the ▶ icon. Before running a test, check if all the test prerequisites stated in the Prerequisites section of the test are met.

Sequence of Steps in LTSSM Tests

1. For an LTSSM test, LTSSM Tester first checks the test prerequisites.
2. If all prerequisites are met, the training sequence is started and LTSSM state transitions are captured, checked and displayed.
3. If an LTSSM state transition is not as per the PCIe specifications, the test stops and fails.
4. The test result is displayed along with the reason for the test failure if the test failed.

Understanding the Test Results

The test results are displayed in the Log pane of the LTSSM Tests window. A test Log helps to identify the root cause of a problem. For instance, the test log shows test failure due to an unexpected LTSSM state change (RecRvcfg was expected but RecSpeed transition happened). This pane displays the following information with the date and time stamp:

- Name of the executed test.
- The list of prerequisites that are being checked for the test execution.
- The status of the "Check prerequisites" step. It displays Success if all the prerequisites are met and displays Failed if one or more prerequisites are not met causing the test to fail.
- The LTSSM states through which Protocol Exerciser transits during the test execution. If any of the LTSSM state transitions is missing or is not the expected state transition specified in the PCIe specifications, the test fails stating the LTSSM state which caused the test to fail.

- The test case result based on whether or not all prerequisites are met and all the LTSSM state transitions are as per the expected state transitions in PCIe specifications. It displays Success, Failure, or Cancelled by user. It also displays the outcome of the test in terms of the current link speed, link width and link status.

You can save the test results as an RTF or a TXT file by right-clicking anywhere inside the Log pane and clicking Save as...

In this release, the following LTSSM tests are provided.

Test	Description
Basic Tests	• Exerciser starts Link Training to 2.5 GT/s on page 102 • Link training to 8.0 GT/s with all the phases of Equalization test on page 104 • Link training to 8.0 GT/s with the first phase of Equalization test on page 106 • Exerciser Initiates Transition to Recovery Test on page 113 • Capture State Transitions Test on page 114
Link Speed Upgrade tests	• Link Speed Change With Equalization Tests on page 111 • Link Speed Change Without Equalization Tests on page 107
Link Speed Downgrade Tests	• Link Speed Change With Equalization Tests on page 111 • Link Speed Change Without Equalization Tests on page 107
Current Data Rate fails in Recovery.Rcvrlock	There are two tests available in this test group to test how the DUT responds when the current data rate (Gen2 / Gen3) doesn't work reliably. In these tests, Exerciser causes the current data rate failure to force the DUT to transition from Recovery.RcvrLock to Recovery.Speed because of this failure.
Exerciser is Directed to Disabled from L0	This test checks how the DUT responds when Exerciser is directed to move to the Disabled state from the L0 state.
Exerciser is Directed to Hot Reset from L0	This test checks how the DUT responds when Exerciser is directed to move to the Hot Reset state from the L0 state.
Equalization Failure tests	There are four tests available in this group to test how DUT responds to the failure in the Equalization procedure in Phase 2/ 3 while moving from Gen1/Gen2 to Gen3. In these tests, Exerciser causes the Equalization procedure to fail in Phase 2/Phase 3 while tranisitioning to Gen3 speed. The tests check whether or not the DUT transitions from: • Recovery.Equalization.Phase2 to Recovery.Speed due to failure of the equalization procedure in Phase2. • Recovery.Equalization.Phase3 to Recovery.Speed due to failure of the equalization procedure in Phase3.
Equalization Behavior tests	There are two tests available in this test group to test the equalization behavior of the DUT specifically when: • Exerciser rejects 8 different values of coefficients or presets in Phase 3 of equalization before accepting these values. • Exerciser requests the DUT to use only the presets (not co-efficients) in Phase 2 of equalization.

Test	Description
Exerciser initiates DownConfiguration followed by UpConfiguration	In this test, Exerciser first downconfigures the link width and then forces the DUT to upconfigure it to the initial higher link width to test DUT's capability to up configure. Both Exerciser and DUT should be capable of upconfiguration to run this test.
Exerciser initiates DownConfiguration followed by UpConfiguration without being Capable	In this test, Exerciser is not capable of upconfiguration. Exerciser first downconfigures the link width and then forces the DUT to upconfigure it to the initial higher link width to test whether or not the DUT upconfigures the link when Exerciser does not have the upconfiguration capability. Exerciser should not have the upconfigure capability and DUT should be capable of upconfiguration to run this test.
Negotiated and Current data rate fails in Recovery.Rcvrlock	This test checks how the DUT responds when the negotiated as well as the current data rate fails while moving from Gen2 to Gen3 speed. The current speed is Gen2, the negotiated speed is Gen3 and there's a failure in the speed change to Gen3 as well as the current Gen2 speed. In this scenario, the test checks if the DUT moves to Gen1 speed responding to these data rate failures.
Negotiated Data Rate fails in Recover.RcvrLock	There are six tests available in this group to test the failure of the negotiated data rate while transitioning from one speed to another (higher or lower) speed. In these tests, Exerciser negotiates a speed change but does not move to the negotiated speed change. This test checks the DUT's capability to revert back to the original speed when the negotiated speed change fails and enter into Recover.RcvrLock.
Exerciser Request Redo Equalization	There are three tests available in this test group to test how the DUT responds to the Exerciser's request to redo equalization in Recover.RcvrCfg when: • Moving from 2.5 GT/s to 8.0 GT/s speed. • Moving from 5.0 GT/s to 8.0 GT/s speed. • While at the 8.0 GT/s speed.
Send Invalid Sync bits continuously at Gen3	In this test, Exerciser sends framing errors to DUT at 8.0 GT/s causing the DUT to enter into the Recovery state. DUT is expected to transition from L0 to Recovery state as it detects Gen3 framing errors.
Exerciser Initiates Speed Change on Any Lane	There are two tests available in this group to test how the DUT responds when Exerciser initiates speed change from Gen1/Gen2 to Gen3 while sending correct TS1s only on Lane 0 (not on all the configured lanes). The test checks whether or not the DUT transitions to the Configuration state in response to correct TS1s sent only on Lane 0.

Testing Link Training

Exerciser starts Link Training to 2.5 GT/s

Description

This test initializes and trains the PCIe link between the Protocol Exerciser and DUT to 2.5 GT/s. Even if the data rate capabilities of both the link partners is higher, the test trains the link to 2.5 GT/s. However, the link speed can later be upgraded by running a link speed change LTSSM test.

In this test, Protocol Exerciser performs link training and exchanges training sequences with DUT to negotiate link parameters like:

- lane polarity
- link number
- set of lanes that belong to the link
- lane numbers
- scrambler enabled or disabled
- link speed
- number of fast training sequences required

Usage

- You can use this test to check if the DUT's LTSSM can initialize the link to 2.5 GT/s. This is the basic test to test the linkup between Protocol Exerciser and DUT. You can use this test as the initial test to test other scenarios such as link transition to recovery and autonomous or manual speed changes after the link initializes to 2.5 GT/s.
- This test also lets you check whether DUT initializes the link with the user-configured link and lane settings such as lane polarity inversion or lane reversal negotiated during the training sequence. You can test if the DUT can understand and implement polarity inversion for lanes if you selected this option for lanes.

Test Results

If the test passes, the LTSSM state is **L0** indicating that the link is up and operational. The link status is displayed as **Active** in the Hardware Status window to indicate the successful link up.

If the test fails, it indicates:

- Link is not up.
- Link or lane settings incompatibility on either of the two link partners side. For instance, Polarity inversion set for the Transmitter of the Protocol Exerciser but not implemented on DUT or data scrambling enabled at USC but not at DSC.
- Missing or unexpected LTSSM state transitions. For instance, the test log shows test failure when the LTSSM state does not transit to Configuration.Linkwidth.Start after Polling.Configuration while running the link training test.

Exerciser starts Link Training to 8.0 GT/s with Equalization Tests

LTSSM Tester provides the following two tests to train the link to the 8.0 GT/s speed with the execution of the Equalization procedure.

- Link training to 8.0 GT/s with the execution of all the phases of Equalization
- Link training to 8.0 GT/s with only the first phase of Equalization

Both these tests are described below with a focus on the difference in their Equalization execution.

Link training to 8.0 GT/s with all the phases of Equalization test

Description

This test initializes and trains the PCIe link between the LTSSM Exerciser and DUT to 2.5 GT/s initially. If the data rate capabilities of both the link partners is advertised as 8.0 GT/s, the test initiates the transition to the Recovery.Equalization sub state to perform all the phases of the Equalization procedure before moving the link to 8.0 GT/s. While performing equalization, Protocol Exerciser advertises the Equalization settings (presets, coefficients, and voltage swings) that you specified in the Equalization Settings page as its Tx capabilities for the transmission of training sequences. In phase 2 and 3, DUT and Exerciser tune the equalization settings by requesting a change in the coefficients at the transmitter of the link partner. Protocol Exerciser uses the equalization settings specified in the Equalization Settings page to perform or respond to these phases. If the Equalization phases are successfully done, then both Protocol Exerciser and DUT transition to the Recovery.RcvrLock sub state to proceed towards making the link operational at the 8.0 GT/s speed.

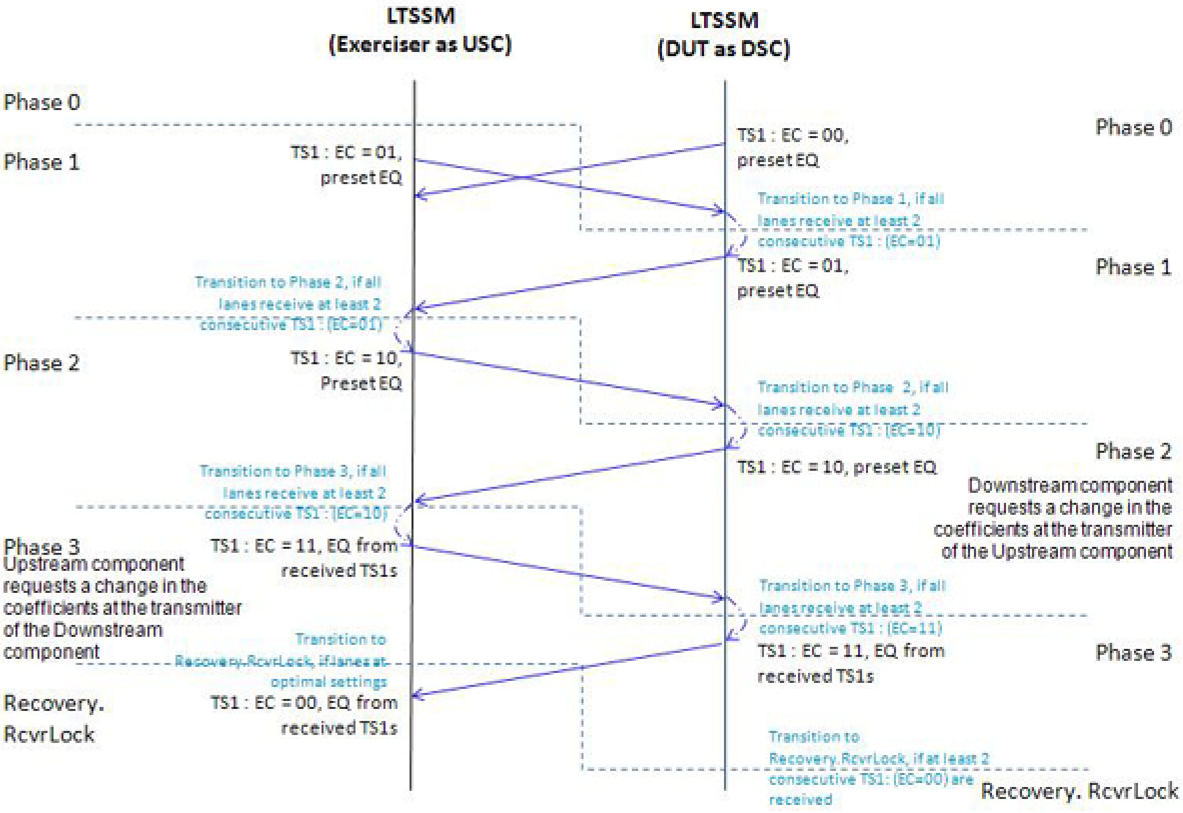
Usage

- You can use this test to check the full execution of the equalization procedure with DUT acting as a USC or a DSC.
- You can verify the Equalization substates entered after reaching to 2.5 GT/s and before achieving 8.0 GT/s.
- You can use this test to troubleshoot at which phase of equalization, the problem occurs.

Test Flow

This section displays the sequence of test execution in terms of the LTSSM state transitions and the steps performed by DUT and Protocol Exerciser during the test execution.

The following figure displays the test flow during the Equalization phases when Exerciser acting as a USC has trained the link to 2.5 GT/s and moved to the Recovery.Equalization substate to perform all the phases of equalization before moving the link to 8 GT/s.


LTSSM
(Exerciser as USC)
LTSSM
(DUT as DSC)
Phase 0
Phase 1
TS1 : EC = 01, preset EQ
TS1 : EC = 00, preset EQ
Phase 0
Transition to Phase 1, if all lanes receive at least 2 consecutive TS1 : (EC=01)
TS1 : EC = 01, preset EQ
Phase 1
Transition to Phase 2, if all lanes receive at least 2 consecutive TS1 : (EC=01)
Phase 2
TS1 : EC = 10, Preset EQ
Transition to Phase 2, if all lanes receive at least 2 consecutive TS1 : (EC=10)
Phase 2
TS1 : EC = 10, preset EQ
Downstream component requests a change in the coefficients at the transmitter of the Upstream component
Transition to Phase 3, if all lanes receive at least 2 consecutive TS1 : (EC=10)
Phase 3
Upstream component requests a change in the coefficients at the transmitter of the Downstream component
TS1 : EC = 11, EQ from received TS1s
Transition to Phase 3, if all lanes receive at least 2 consecutive TS1 : (EC=11)
TS1 : EC = 11, EQ from received TS1s
Phase 3
Transition to Recovery.RcvrLock, if lanes at optimal settings
Recovery.
RcvrLock
TS1 : EC = 00, EQ from received TS1s
Transition to Recovery.RcvrLock, if at least 2 consecutive TS1: (EC=00) are received
Recovery. RcvrLock

Test Results

If the test passes, the LTSSM state is L0 indicating that the link is up and operational at 8.0 GT/s. The link status is displayed as Active in the Hardware Status window to indicate successful link up.

If the test fails, it indicates:

- One or more of the test prerequisites are not met.
- Link or lane settings incompatibility on either of the two link partners side. For instance, Polarity inversion set for the Transmitter of the Protocol Exerciser but not implemented on DUT.

- Failure of transition to correct Equalization substates while performing Equalization.
- Inappropriate Equalization settings.

You can try fine tuning the Tx De-emphasis values for the Protocol Exerciser and the DUT using the Equalization Settings tab.

Link training to 8.0 GT/s with the first phase of Equalization test

Description

This test initializes and trains the PCIe link between the Protocol Exerciser and DUT to 2.5 GT/s initially. If the data rate capabilities of both the link partners is advertised as 8.0 GT/s, the test initiates the transition to Recovery to perform the first phase of the Equalization procedure before achieving 8.0 GT/s. During link training, the first phase requires both USC and DSC to advertise the preset values. Protocol Exerciser advertises the Equalization settings (presets, co-efficients, and voltage swings) that you specified in the Equalization Settings page as its Tx capabilities for the transmission of training sequences. USC exits the Recovery.Equalization sub-state after phase 1 and transitions to the Recovery.RcvrLock substate to proceed towards making the link operational at the 8.0 GT/s speed.

Usage

- You can use this test to check if the DUT’s LTSSM can initialize the link to 8.0 GT/s with the execution of phase 1 of equalization.
- You can verify the Equalization substates entered after reaching to 2.5 GT/s and before achieving 8.0 GT/s.
- You can use this test to troubleshoot whether or not the device is capable of undergoing the first phase of equalization while achieving 8.0 GT/s.

Test Results

If the test passes, the LTSSM state is L0 and link speed is 8.0 GT/s indicating that the link is up and operational at 8.0 GT/s. The link status is displayed as Active in the Hardware Status window to indicate successful link up.

If the test fails, it indicates:

- One or more of the test prerequisites are not met.

- Link or lane settings incompatibility on either of the two link partners side. For instance, Polarity inversion set for the Transmitter of the Protocol Exerciser but not implemented on DUT.
- Failure of transition to correct LTSSM states while performing Equalization.
- Inappropriate Equalization settings.

You can try fine tuning the Tx De-emphasis values for the Protocol Exerciser and DUT using the Equalization Settings tab.

Testing Link Speed Changes

Link Speed Change Without Equalization Tests

This topic describes the LTSSM tests that you can use to test the DUT’s LTSSM function of performing or responding to link speed changes on a configured link. In these tests, Equalization is not performed when moving to the 8.0 GT/s speed from 2.5 GT/s or 5.0 GT/s. The tests transitions the LTSSM of link partners to Recovery and then back to the operation state at the new speed.

Before executing a link speed change test:

- ensure that the Protocol Exerciser's data rate capabilities are appropriately set to support the desired link speed. You set data rate capabilities using the Link Settings page of the General icon in the Protocol Exerciser GUI. If one of the link partners does not support the desired link speed, then the test fails stating that the prerequisites is not met.

NOTE

For link speed change to 8.0 GT/s tests, the Equalization options that you enabled/disabled in the **Equalization Settings** tab are over-ridden by the equalization options relevant for the test when the test is running. However, the equalization options that you set are reverted back when the test execution is over.

For instance, you have not deselected the **Perform Equalization in recovery @ 8 GT/s** option in the **Equalization Settings** tab and executed the LTSSM test to upgrade the link speed to 8.0 GT/s without equalization. In this situation, the test overrides the enabled equalization setting and skips equalization while upgrading to 8.0 GT/s.

Description

Once the link is up between the Protocol Exerciser and DUT, you can use various link speed change tests available in the LTSSM Tester to upgrade the link speed or later downgrade it for link bandwidth management testing.

These tests are categorized into:

- **Link upgrade speed tests** – The tests support the following speed upgrades:
  - 2.5 to 5.0
  - 5.0 to 8.0 without Equalization
  - 2.5 to 8.0 without Equalization
- **Link downgrade speed tests** – The tests support the following speed downgrades:
  - 8.0 to 5.0
  - 5.0 to 2.5
  - 8.0 to 2.5

These tests are further categorized on the basis of the link partner (Protocol Exerciser or DUT) from where you want to initiate the link speed change request. Once the request is initiated, based on whether the Protocol Exerciser is acting as a USC or a DSC for the session, it performs or responds to these link speed change requests during the test execution.

NOTE

If you want to test a link speed change request initiated by DUT, then you manually need to change the link speed at the DUT end on receiving instructions during the test execution.

Using these tests, you can also test the implementation of the autonomous speed change while changing the link speed to 8.0 GT/s on a configured link.

Usage

You can use the link speed change tests to assess:

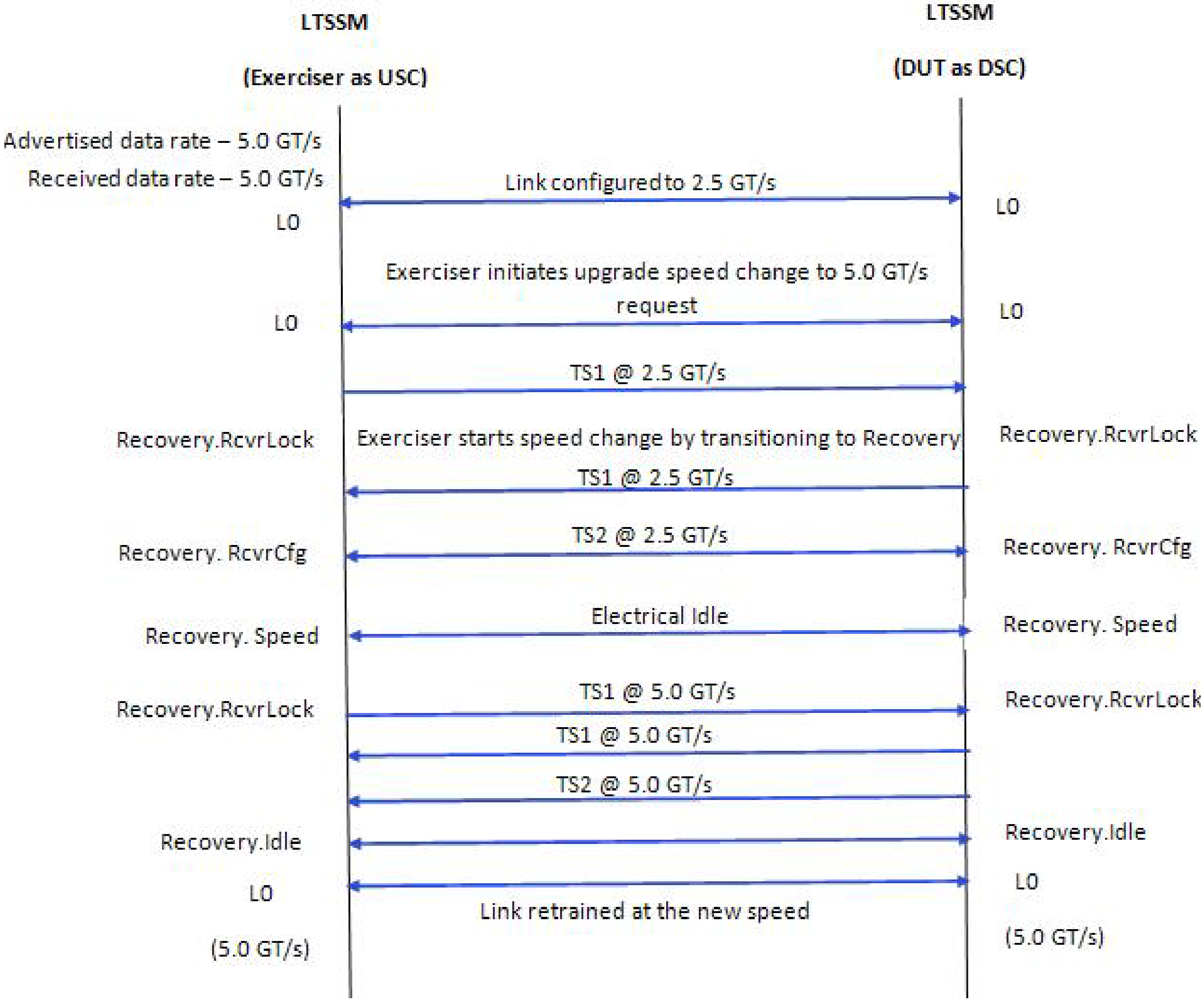
- DUT's capabilities to perform or respond to link bandwidth management requests.
- DUT's capabilities to upgrade to 8.0 GT/s without the equalization procedure on a configured link.
- The interconnect data rate as a result of the link speed changes and data rate capabilities of link partners.

- The LTSSM state transitions are as per the PCIe specifications. For instance, LTSSM state transitions to all the appropriate substates of Recovery before making the link operational at the desired speed change.
- Whether or not DUT enters the Equalization substates before moving the link speed to 8.0 GT/s.

Test Flow

This section displays the sequence of the test execution in terms of the LTSSM state transitions and the steps performed by DUT and LTSSM Exerciser during the test execution.

The following figure displays the test flow when Exerciser initiates the request to upgrade the link speed from 2.5 to 5.0 GT/s with LTSSM Exerciser acting as a USC.



Test Result

If the link speed change test passes, the link is operational and trained to the new (desired speed).

If the test fails, it indicates:

- One or more of the listed test prerequisites failed.
- Data rate incompatibility. Either of the two link partners does not support the desired link speed.
- Missing or unexpected LTSSM state transitions. For instance, the test log shows test failure for an upgrade speed test when the LTSSM state transits to Rec.Speed instead of the expected Rec.RcvrCfg after Rec.RcvrLock.
- Transition to Equalization substate(s) by a link partner for a speed upgrade to 8.0 GT/s.

Link Speed Change With Equalization Tests

This topic describes the LTSSM tests that you can use to test the DUT’s LTSSM function of performing or responding to link speed change to 8.0 GT/s on a configured link. In these tests, equalization is performed before moving to the 8.0 GT/s speed. You can choose a test based on whether you want the test to perform only phase 1 or all the phases of equalization.

In phase 1 of equalization, both DUT and Protocol Exerciser advertise their equalization preset values. In phase 2 and 3, DUT and Exerciser tune the equalization settings by requesting a change in the coefficients at the transmitter of the link partner. Protocol Exerciser uses the equalization settings specified in the Equalization Settings page to perform or respond to these phases.

Either DUT or Exerciser can initiate these tests for a link configured to 2.5 GT/s or 5.0 GT/s. Based on whether the Protocol Exerciser is acting as a USC or a DSC for the session, it performs or responds to these link speed change requests during the test execution.

Before executing a link speed change test:

- ensure that the Protocol Exerciser's data rate capabilities are set to support the desired link speed, 8.0 GT/s. You set data rate capabilities using the Link Settings page of the General icon in the Protocol Exerciser GUI. If one of the link partners does not support the 8.0 GT/s link speed, then the test fails stating that the prerequisites are not met.
- ensure that the preset, co-efficient, and minimum/maximum voltage swing values are set for the Protocol Exerciser transmitter part. Also, the equalization setting values that the Protocol exerciser sends to DUT as a request to use these settings to tune the equalization should be set. You use the Equalization settings page to configure these settings.

NOTE

For these tests, the Equalization options that you enabled/disabled in the Equalization tab are over-ridden by the equalization options relevant for the test when the test is running. However, the equalization options that you set are reverted back when the test execution is over.

For instance, you have not enabled the **Perform Equalization in recovery @ 8 GT/s** option in the **Equalization Settings** tab and executed the LTSSM test to upgrade the link speed to 8.0 GT/s with equalization. In this situation, the test overrides the disabled equalization setting and performs equalization while upgrading to 8.0 GT/s.

Description

Once the link is up between the Protocol Exerciser and DUT, you can use various link speed change tests available in the LTSSM Tester to upgrade the link speed to 8.0 GT/s to specifically test the Gen3 speed bandwidth management.

The tests support the following speed upgrades:

- 5.0 to 8.0 with phase 1 or all phases of equalization
- 2.5 to 8.0 with phase 1 or all phases of equalization

These tests are further categorized on the basis of the link partner (Protocol Exerciser or DUT) from where you want to initiate the link speed change request. Once the request is initiated, based on whether the Protocol Exerciser is acting as a USC or a DSC for the session, it performs or responds to these link speed change requests during the test execution.

NOTE

If you want to test a link speed change request initiated by DUT, then you manually need to change the link speed at the DUT end on receiving instructions during the test execution.

Using these tests, you can test the implementation of the Equalization procedure while changing the link speed to 8.0 GT/s on a configured link.

Usage

You can use the link speed change tests to assess:

- DUT’s capabilities to perform or respond to link bandwidth management requests.
- DUT’s capabilities to implement the Equalization procedure while upgrading to 8.0 GT/s.

- The interconnect data rate as a result of the link speed changes and data rate capabilities of link partners.
- The equalization related LTSSM state transitions are as per the PCIe specifications.

Test Result

If the link speed change test passes, the link is operational and trained to the new (desired speed).

If the test fails, it indicates:

- One or more of the listed test prerequisites failed
- Either of the two link partners does not support the 8.0 GT/s link speed.
- Missing or unexpected LTSSM state transitions. For instance, absence of transition to Equalization substates while moving to the 8.0 GT/s speed.

Testing Link Retraining and Recovery

Exerciser Initiates Transition to Recovery Test

Description

This test allows you to direct the LTSSM Tester to initiate the link retraining or recovery by transiting to the Recovery LTSSM state and then back to the Operational state.

When the test initiates link retraining/recovery, the link is retrained/recovered as per the link and lane settings negotiated during the initial link training sequence. For example, the link is trained to 2.5 GT/s and the data rate capabilities of DUT and Protocol Exerciser are 5.0 GT/s. By initiating the transition to Recovery, the negotiated data rate – 2.5 GT/s is used to recover and retrain the link. The Advertised and Received data rate capabilities are updated based on the current data rate capabilities of the link partners.

Usage

You can use this test to check:

- DUT’s capabilities to respond to the transition to the Recovery state, in situations such as link errors.
- If the link is retrained/recovered as per the link and lane negotiations in the initial training sequence in response to an initiation of the transition to recovery.

Test Results

The test passes if the link transitions to recovery and then back to the Operational state and the data rate at which the link is retrained is the initial data rate at which the link was initially trained. On the successful execution of the test, the link status is displayed as Active in the Hardware Status window.

If the test fails, it indicates:

- Link is not up.
- Link width is not x1, x4, x8 or x16.
- Failure of transition to the correct Recovery substates.

Testing LTSSM State Transitions

Capture State Transitions Test

Description

Protocol Exerciser and DUT transits through various LTSSM states while performing LTSSM functions. This test can capture and display a maximum of 512 consecutive LTSSM state transitions of the Protocol Exerciser to check the sequence of LTSSM transitions and identify the state at which a problem occurred.

Usage

When the link between the Protocol Exerciser and DUT is configured and operational, the link status is displayed in the Hardware Status window of the Protocol Exerciser GUI. The current state of link is reflected through this status (either No Link or Active). There may be situations when you want to troubleshoot the No Link status of the link by checking the LTSSM state transitions. In such situations, this test can provide you a trail of LTSSM state transitions to help you identify the source of problem in the link between DUT and Protocol Exerciser.

You can also use this test to check if the sequence of DUT's LTSSM state transitions for an LTSSM function is as per the PCIe specifications.

Test Results

If the test passes, the LTSSM state transitions are displayed in the test log along with the current link width and link status.

6
Defining Configuration Space of Exerciser

Configuration Space - Overview 116

This chapter provides an overview of the configuration space of Exerciser.

Configuration Space - Overview

The **Configuration Space** page allows you to define the PCI compatible configuration space for all the functions that Exerciser supports as a multi-function device. If you select the MRIOV protocol for creating a link between Exerciser and DUT, then Exerciser emulates an MRIOV capable component. As a result, the following additional blocks are added in its configuration space to define additional capabilities for its role:

- **MRIOV Capability** block for Function A (To initialize and configure the MRIOV capabilities of Base function of Exerciser)
- **SRIOV Capability** block for Function B and Function C (To initialize and configure the SRIOV capabilities of Physical functions of Exerciser)

In the **Configuration Space** page, different tabs are aligned according to the PCI, PCI-X and PCI Express specification and provide settings for various registers which are well documented in the related specifications.

NOTE

Configuration register fields have different access types associated – e.g. read/ write or read-only. By default, the access type for registers is set as per the protocol specifications. However, if required, you can override the default access type for a register bit and take full read/write access for that bit. To do this, right-click the field displayed for that bit in the Configuration Space page of the GUI and select **force r/w access to field**.

On selecting full read/write access for a bit, the color for that bit's field changes to Blue in the GUI indicating its current state as full read/write access. A color code yellow is displayed for fields in cases where mixed access values are applicable based on your selections for other related fields.

7
Configuring and Mapping Decoder and Data Memory

Configuring Decoder Settings 118
Configuring and Using Data Memory of Exerciser 122
Comparing Actual Data with Expected Data in Memory 126

This chapter provides information on the how you can configure the decoders provided by Exerciser and map these decoders to specific locations in the data memory of Exerciser. It also describes how the data memory of Exerciser can be accessed for storing request and completion data and comparing the actual data with the expected data at the targeted memory location.

Configuring Decoder Settings

Exerciser provides six Base Address Registers (BARs) - BAR 0 to BAR 5 and an Expansion ROM for each function supported in its configuration. For each of these six BARs and an Expansion ROM of a function, an address decoder is available. You use the Decoder page to configure the address decoder settings for each BAR provided for a function.

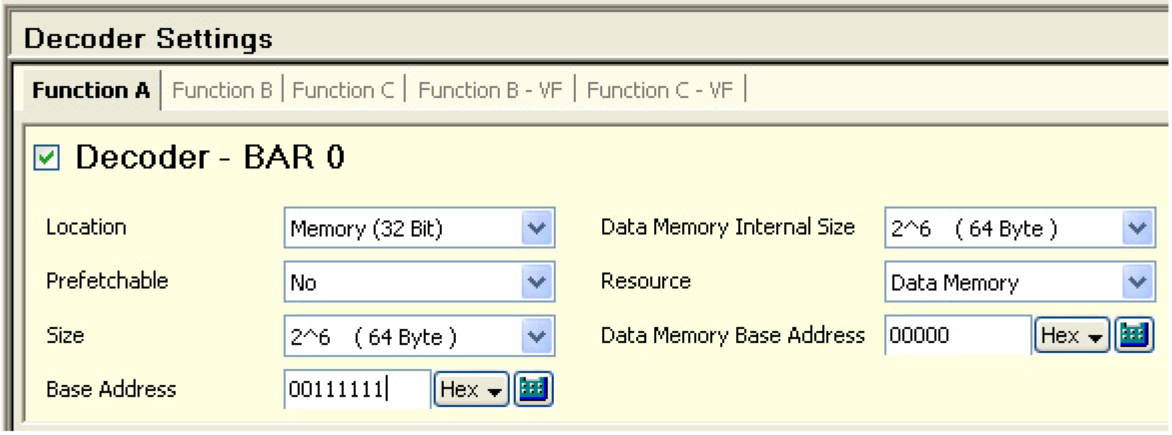
When an incoming TLP (requesting memory access through a BAR) is received, Exerciser assigns it to the decoder of the appropriate bar based on the TLP's address field, the type of TLP, and the decoder base address that you specified in the decoder settings.

The decoder decodes and translates the PCI address used in the incoming TLP to an address in the internal memory of Exerciser. This translation is done based on the mapping that you specified between the decoder base address and the internal memory address of Exerciser in the decoder settings of the BAR. The mapped internal memory address is then used either for the TLP payload storage or for the actual data comparison with the targeted data memory contents based on the specified decoder settings.

When Exerciser sends a completion with data such as a memory read completion, it gets the payload of such a completion from the relevant data memory address. The decoder of the applicable BAR ascertains this data memory address using the mapping that you specified between the PCI address and the internal memory address in the decoder settings.

Enabling or disabling decoders

In the Decoder page, you can choose to enable or disable the decoder for a BAR of a function. If you disable a decoder, then the corresponding BAR of the function is not available for TLPs requesting memory space. To enable the decoder for a BAR, select the checkbox displayed with the decoder in the relevant function's tab in the Decoder page. The following screen displays the decoder enabled for BAR 0 of Function A.



☑ Decoder - BAR 0

Location: Memory (32 Bit)

Prefetchable: No

Size: 2^6 (64 Byte)

Base Address: 00111111 Hex

Data Memory Internal Size: 2^6 (64 Byte)

Resource: Data Memory

Data Memory Base Address: 00000 Hex

Figure 18 Enabled decoder for Function A

Decoder settings

You can specify the following settings for the decoder of a BAR of a function:

- Specify the type of TLPs that you want the decoder to handle. For instance, you can configure a decoder to handle I/O requests for the function.
- Enable or disable the Prefetchable bit for the applicable BAR to indicate whether or not the BAR can request prefetchable memory resources in case of memory addressing.
- Specify the base address and size for the decoder to determine which packets should hit the decoder.
- Specify the internal memory base address and size that you want to map to the specified base address and size for decoder. These attributes are used to either store the payload of incoming TLPs or to compare the payload with the expected contents at this address.
- Select how you want the decoder to handle the TLP's payload. You can choose to discard the payloads, store these at the specified internal memory base address, or compare these with the contents at the specified internal memory base address.

VF BARs provided for an MRIOV capable Exerciser

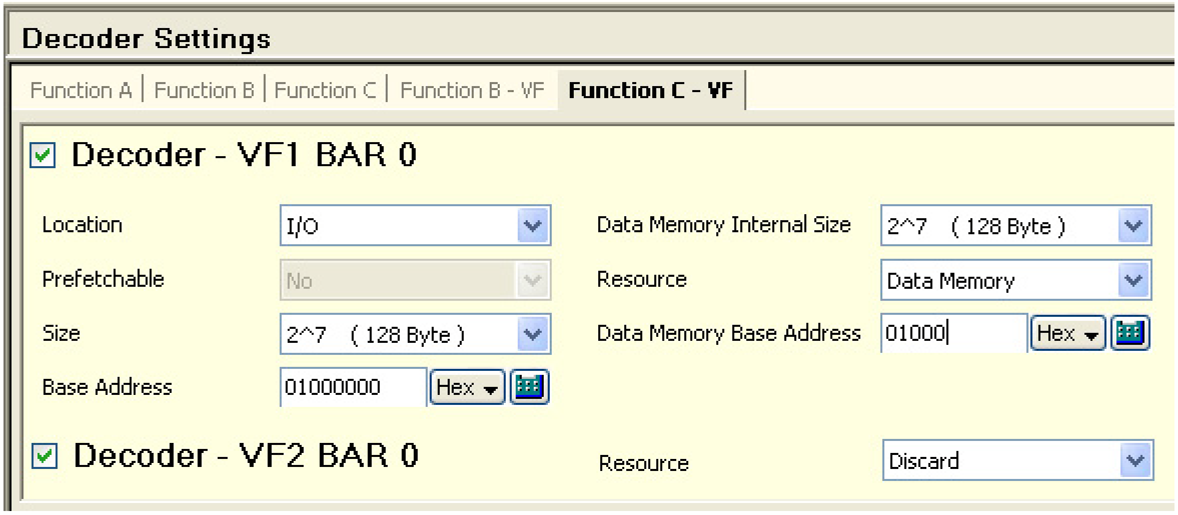
When Exerciser emulates a non IOV PCIe component, it provides six BARs for each of the three functions - Function A, B, and C.

When Exerciser emulates an MRIOV/SRIOV capable component, it provides the BARs as follows for each of its virtual and physical function.

Function A (With MRIOV capabilities)	Six BARs
Function B (Physical function with SRIOV capabilities)	Six BARs
Function C (Physical function with SRIOV capabilities)	Six BARs
Virtual function VF1 associated with Function B	Six VF BARs
Virtual function VF2 associated with Function B	Six VF BARs
Virtual function VF1 associated with Function C	Six VF BARs
Virtual function VF2 associated with Function C	Six VF BARs

In case of an MRIOV/SRIOV capable Exerciser, you specify the decoder settings for the VF BARs for virtual function (VF1) of a physical function. Based on the VF1 settings specified for a BAR, the settings for VF2 associated with that BAR are calculated automatically. For instance, you specify the **Data Memory Internal Size** for BAR 0 for VF1 of Function C. The same size is applicable for the BAR 0 of VF2 of that function. This size is added to the specified **Data Memory Base Address** of VF1 to derive the Data Memory Base Address of VF2 for that BAR. Therefore, the only decoder setting that you specify for VF BARs of VF2 is how you want the payload of TLPs to be handled.

The following screen displays a sample decoder configuration for the VF BAR 0 of the virtual functions VF1 and VF2 provided by Function C.



☑ Decoder - VF1 BAR 0

Location: I/O

Prefetchable: No

Size: 2^7 (128 Byte)

Base Address: 01000000 Hex

Data Memory Internal Size: 2^7 (128 Byte)

Resource: Data Memory

Data Memory Base Address: 01000 Hex

☑ Decoder - VF2 BAR 0

Resource: Discard

Figure 19 Decoder configuration for a VF BAR

Configuring and Using Data Memory of Exerciser

Data memory is a memory block in the U4305A Protocol Exerciser card used to store the payload for the TLP requests and completions transmitted from Exerciser and the payload for the TLP requests and completions received from DUT. A DUT can access the data memory of Exerciser by sending requests such as memory read and write requests.

You can use the data memory to:

- get the payload of a request packet such as a memory write request to be sent as stimulus from Exerciser to DUT.
- get the payload of a completion packet to be sent from Exerciser for a request that requires a completion with data such as a memory read request from DUT.
- store the payload of a request packet such as a memory write request received from DUT.
- store the payload of a completion packet received from DUT for a request that requires a completion with data such as a read request from Exerciser.

The **Data Memory** page in the Protocol Exerciser GUI provides you access to the data memory block of the Exerciser card to perform tasks such as:

- View and edit memory contents
- Go to a specific address in the memory
- Reload data memory
- Initialize data memory
- Compare actual data with the expected data in the memory
- Import memory contents from a file or export memory contents to a file

In the Data Memory page, the memory is represented in the form of Address, Data and the ASCII representation of the Data. The size of the data memory is 256 KByte.

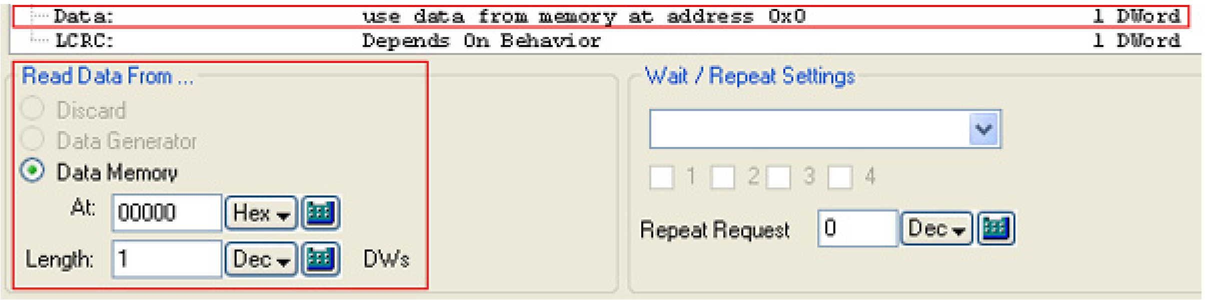
Mapping data memory addresses to BAR addresses

You can map an address of data memory of Exerciser to a Base Address Register (BAR) location for a function supported by Exerciser. The mapping ensures the decoding and translation of PCI addresses into data memory addresses for storing the data of incoming TLP requests requesting memory access through a BAR. Refer to the topic Configuring Decoder Settings on page 118 to know more.

Writing to and reading from data memory of Exerciser

When a request is sent with data from Exerciser, the data memory of Exerciser card is accessed to get the payload of this request from the specified start address. Similarly, when a completion with data is received from DUT, the data memory of Exerciser card is accessed to store the payload of this completion at the specified address.

The following screens display the relevant group boxes of the Edit Packet dialog box on page 193 that you use to read payload from and write payload to data memory. In the first screen, the group box in which you specify the start address of the data memory from where the payload of a request should be constructed is displayed. The second screen displays the group box in which you specify the start address of the data memory where the payload of the completion received for the request should be stored.



```
Data:      use data from memory at address 0x0      1 DWord
LCRC:      Depends On Behavior                      1 DWord
```

Read Data From ...

○ Discard

○ Data Generator

● Data Memory

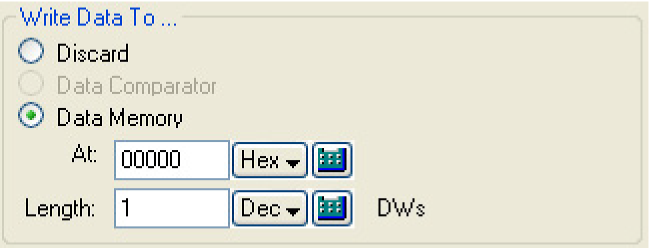
At: 00000 Hex

Length: 1 Dec DWs

Wait / Repeat Settings

□ 1 □ 2 □ 3 □ 4

Repeat Request 0 Dec



Write Data To ...

○ Discard

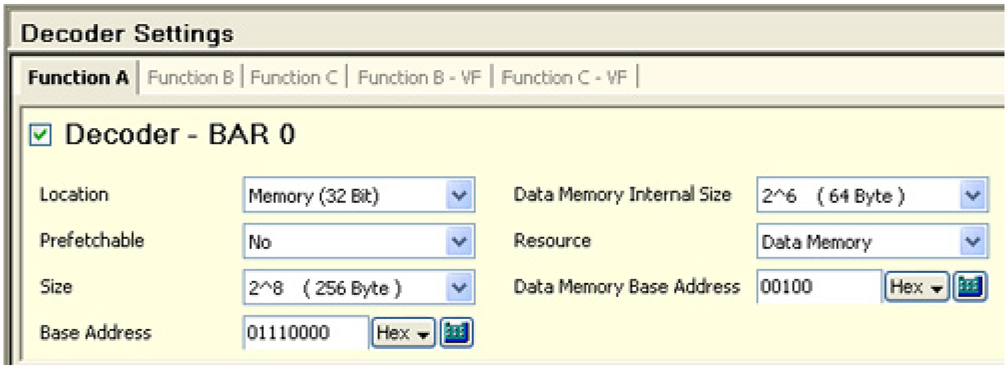
○ Data Comparator

● Data Memory

At: 00000 Hex

Length: 1 Dec DWs

When a TLP request (with data) is received from DUT, Exerciser can store the payload of such a request at a specified data memory address. This address is determined using the mapping that you defined between the data memory address and the decoder address for the decoder applicable for the request. In addition, to allow the storage of incoming payloads in data memory, you should have selected **Data Memory** in the **Resource** listbox of the applicable decoder settings. Refer to the topic Configuring Decoder Settings on page 118 to know more. The following screen displays a decoder to data memory address mapping for Function A in the Decoder Settings page.



Decoder Settings

Function A | Function B | Function C | Function B - VF | Function C - VF |

☑ Decoder - BAR 0

Location: Memory (32 Bit)
Prefetchable: No
Size: 2^8 (256 Byte)
Base Address: 01110000 Hex

Data Memory Internal Size: 2^6 (64 Byte)
Resource: Data Memory
Data Memory Base Address: 00100 Hex

When Exerciser sends a completion with data such as a memory read completion, it can read the payload of such a completion from a specified data memory address. This address is determined using the mapping that you defined between the data memory address and the decoder address for the applicable decoder for the read request. Refer to the topic Configuring Decoder Settings on page 118 to know more.

NOTE

You can view the contents written at a particular data memory address or write contents to a particular data memory address using the **Edit memory** tab in the **Data Memory** page. In this tab, you can directly jump to a desired data memory address by selecting **Edit** > **Goto Data Memory Address** or clicking the ⇥ toolbar button. You can manually edit the contents of the data memory in the **Edit memory** tab.

Importing and Exporting Data Memory contents

You can import the contents of a specified file as payloads to the data memory of Exerciser. This allows you to configure the data memory with the data of your choice that you can use to insert payloads with the stimulus packets sent from Exerciser.

To import data memory contents from a file

1 Select **File** > **Import** > **Import Payload Memory from File**.

Or, click the toolbar button in the Data Memory page.

The **Import Memory from File** dialog box appears.

2 Refer to the Import Memory from File dialog box on page 197 GUI reference help topic to get a description of each field in this dialog box.

NOTE You can also edit the contents of the data memory manually using the **Edit Memory** tab in the **Data Memory** page.

You can export the contents of the data memory of the Exerciser card to a specified file for a later use.

To export data memory contents to a file

1 Select **File** > **Export** > **Export Payload Memory to File**.

Or, click the toolbar button in the Data Memory page.

The **Export Memory to File** dialog box appears.

2 Refer to the Export Memory to File dialog box on page 196 GUI reference help topic to get a description of each field in this dialog box.

Initializing and reloading data memory

Reloading the data memory contents refreshes the **Data** column in the **Edit Memory** tab with the most recent values from the data memory.

To reload data memory contents

1 Select **Action** > **Reload Data Memory**.

Alternatively, click the icon on the toolbar or press F5 on the keyboard.

Directly typing the data is a quick method of updating the memory contents of a particular memory address. However, this method might be time consuming and difficult if you need to initialize the entire data memory or a big range of memory addresses. To help you in this situation, Exerciser provides the Initialize Memory dialog box on page 198. This dialog box initializes the entire data memory or a specific range of data memory addresses with the value specified in this dialog box.

To access the Initialize Memory dialog box:

1 Select **Action** > **Initialize Data Memory**.

   Or, click the 00 icon on the toolbar.

Comparing Actual Data with Expected Data in Memory

In addition to writing to and reading from data memory, Protocol Exerciser also allows you to compare the payload data of the incoming TLPs with the data stored in the data memory. You can choose to discard the data of the incoming TLPs, compare the data with the data at a specific location in data memory, or store this data at a specific location in data memory. You can make this choice for each decoder for a BAR of a function. If you configure a decoder settings to perform data comparison, then all the incoming TLP requests applicable for that decoder are compared with the data at the specified target memory location.

Besides the data comparison for incoming requests, you can also configure Exerciser to perform data comparison for incoming completions. You can set the option for data comparison for completions separately for each physical and virtual function supported by Exerciser. You select the **Enable memory compare for completions** checkbox for the required function(s) in the **Data Memory** > **Memory Compare** page. On selecting this checkbox, the incoming completions for that function are compared with the data at the data memory location that you specified in the applicable decoder settings.

You can configure Exerciser to perform data comparison for the incoming TLP requests and completions by performing the following steps:

1 Configure the applicable decoder bar settings for data comparison:

   a Click **Decoder** in the navigation pane to access the **Decoder Settings** page.

   b Enable the decoder for the BAR of the function for which you want Exerciser to perform data comparison on incoming TLPs.

   c Set the required decoder settings for the enabled decoder. Specifically, ensure that you:

   - select **Compare** in the **Resource** listbox.
   - specify a target memory address in the **Data Memory Base Address** textbox at which the expected data is to be stored for comparison with actual data.

   d Save the decoder settings.

2 Enable the data comparison for the incoming completions received by Exerciser separately for each function:

   a Click **Data Memory** in the navigation pane to access the Data Memory page.

   b Click the **Memory Compare** tab.

   c Select the **Enable memory compare for completions** checkbox for the required function.

   d Save the changes.

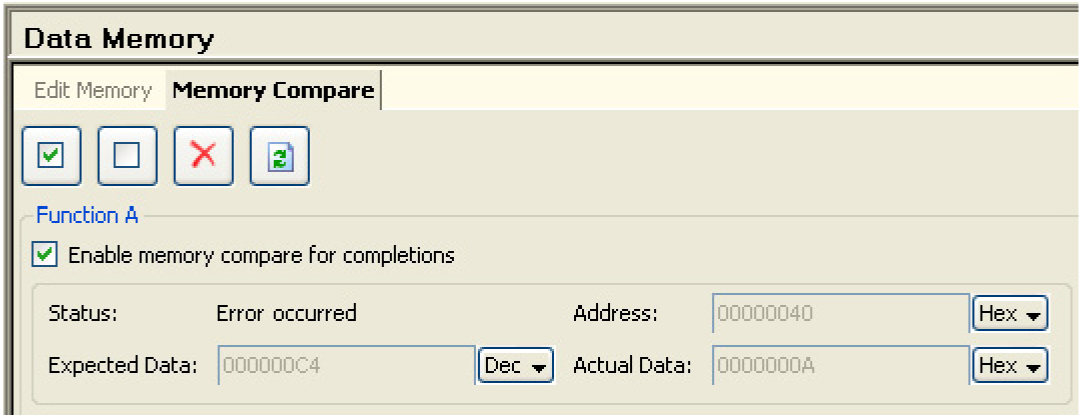
3 Write the expected contents to the target memory location that you specified in the decoder settings. You can either manually edit the memory to write the contents or import the contents from a file at the required address. The expected contents are compared with the actual data of incoming TLPs.

Generating a Trigger out on encountering a data compare error

Exerciser can also generate an event and a Trigger Out pulse on encountering a data compare error. Such an event is displayed in the **General Settings -> Trigger Out** tab. The trigger out pulse is sent from Exerciser to another test equipment connected to the Trigger out Connector component of the Exerciser card. To ensure that an event and a trigger out pulse is generated for a data comparison error, you need to select **Data Compare error** as the trigger out condition in the **General Settings -> Trigger Out** tab. Refer to the topic Configuring External Triggers Settings on page 69 to know more.

Viewing the actual and expected data

Once you have configured Exerciser to perform data comparison for specific functions, Exerciser starts comparing the incoming completions data with the expected data at the specified address in the memory. The results of this comparison are displayed separately for each function for which comparison is enabled in the **Memory Compare** tab of the **Data Memory** page. Exerciser takes enabled bytes into account while doing data comparison. If an error is encountered in the data comparison, it is displayed on the Memory Compare page along with the Expected and Actual data at the targeted data memory address. The following screen displays an example in which the actual data of a completion for Function A does not match the expected data at the displayed memory address.



Data Memory

Function A

☑ Enable memory compare for completions

Status: Error occurred

Address: 00000040 Hex

Expected Data: 000000C4 Dec

Actual Data: 0000000A Hex

NOTE In the Link Status tab of the Hardware Status pane, the Memory Compare section shows the **Error occurred** message to indicate that a memory compare error has occurred.

8
Providing Stimulus to DUT

Stimulus Traffic - Overview 130

Defining Stimulus Traffic 136

Starting the Stimulus and Viewing Results 143

Stimulus - Example 149

Sending a Single Packet as Stimulus and Viewing Response 152

Sending Completions as Response to DUT 157

This chapter describes how you can define the stimulus traffic for the DUT and send this traffic to DUT over the configured PCIe link. It also includes how you can edit PCIe packets and their behavior before sending these to the DUT and how you can configure Protocol Exerciser to respond to DUT by sending completions.

Stimulus Traffic - Overview

Protocol Exerciser can send a single TLP or a block of TLPs as stimulus requests to DUT and can also respond to a DUT's request by sending completion packets.

This topic provides an overview of the stimulus traffic flow of Protocol Exerciser as a non IOV PICe component, SRIOV capable component, and as an MRIOV capable PCIe component.

Refer to the topic Functions, Hardware Channels, Virtual Hierarchies, and Virtual Channels on page 22 to get an overview of the functions, channels, and hierarchies that Exerciser provides in its PCIe, SRIOV, and MRIOV configurations.

Stimulus traffic flow as a non IOV PCIe component

When you create a non IOV PCIe link between Protocol Exerciser and DUT, Protocol Exerciser emulates a PCIe component that does not have IOV capabilities.

When you create a SRIOV link between Protocol Exerciser and DUT, Protocol Exerciser emulates a PCIe component that has SRIOV extended capabilities.

Hardware channels for stimulus traffic

In the non IOV and SRIOV PCIe emulation, Protocol Exerciser provides three hardware channels for sending stimulus traffic. Each of these hardware channels is associated with a function. As a non IOV PCIe component, Exerciser supports the following three functions:

- Function A
- Function B
- Function C

If you have the U4305A-024 or U4305U-024 software license, then two additional physical functions (Function D and E) are provided making it a total of five non IOV functions and five hardware channels.

You can define a single stimulus packet or a block of stimulus packets separately for each of these three/five functions. On starting the stimulus, Protocol Exerciser sends the packets that you added for a function using the hardware channel that corresponds to that function.

Protocol Exerciser sends the block of packets defined for a function in the sequence in which you add these packets to the function. However, if you defined block of packets for multiple functions, then the sequence in which the blocks from different functions are sent is decided by Protocol Exerciser based on factors such as the credit limit available for the applicable virtual channel.

By default, all functions are enabled. However, you can choose to send stimulus either from one of these functions or from all the functions in which you added stimulus traffic.

Each function is assigned a unique number which is displayed in the Function status tab of the Hardware status pane.

Virtual channels for stimulus traffic

Protocol Exerciser supports two Virtual Channels (VC0 and VCx) as a non IOV PCIe component. It determines the VC to be used to pass a particular stimulus packet added to a function based on the following settings:

- The traffic class (TC) of the stimulus request that you specified using the **Traffic Setup** page.
- The function (A, B, or C) to which you added this request using the **Traffic Setup** page.
- The TC VC mapping that you defined for the function to which you added the request. You use the **Virtual Channel** page to define this mapping.

For instance, if you added a stimulus request TLP to Function A, assigned traffic class 6 to this request, and mapped traffic class 6 to VC0 for Function A, then this request is passed through VC0.

Stimulus traffic flow as a SRIOV capable PCIe component

When you create a SRIOV link between Protocol Exerciser and DUT, Protocol Exerciser emulates a PCIe component that has SRIOV extended capabilities.

Hardware channels for stimulus traffic

In the SRIOV emulation, Protocol Exerciser provides seven hardware channels for sending stimulus traffic. Each of these hardware channels is associated with a physical or a virtual function. As a SRIOV capable component, Exerciser supports the following three physical functions:

- Function A
- Function B
- Function C

The function B and C are SRIOV capable and have two virtual functions VF1 and VF2 associated with each of them.

If you have the U4305A-024 or U4305U-024 software license, then two additional physical functions (Function D and E) are provided making it a total of 13 physical and virtual functions and 13 hardware channels.

You can define a single stimulus packet or a block of stimulus packets separately for each of these seven/thirteen functions. On starting the stimulus, Protocol Exerciser sends the packets that you added for a function using the hardware channel that corresponds to that function.

Protocol Exerciser sends the block of packets defined for a function in the sequence in which you add these packets to the function. However, if you defined block of packets for multiple functions, then the sequence in which the blocks from different functions are sent is decided by Protocol Exerciser based on factors such as the credit limit available for the applicable virtual channel.

By default, all functions are enabled. However, you can choose to send stimulus either from one of these functions or from all the functions in which you added stimulus traffic.

Each function is assigned a unique number which is displayed in the Function status tab of the Hardware status pane.

Virtual channels for stimulus traffic

Protocol Exerciser supports two Virtual Channels (VC0 and VCx) as a SRIOV component. It determines the VC to be used to pass a particular stimulus packet added to a function based on the following settings:

- The traffic class (TC) of the stimulus request that you specified using the Traffic Setup page.
- The function (A, B, C, D, or E) to which you added this request using the Traffic Setup page.
- The TC VC mapping that you defined for the physical function to which you added the request. If you added the request to a virtual function, then the TC VC mapping of its physical function is used. You use the Virtual Channel page to define this mapping.

For instance, if you added a stimulus request TLP to Function A, assigned traffic class 6 to this request, and mapped traffic class 6 to VC0 for Function A, then this request is passed through VC0.

Stimulus traffic flow as an MRIOV capable PCIe component

When you create an MRIOV enabled link between Protocol Exerciser and an MRIOV capable DUT, Protocol Exerciser emulates a PCIe component with MRIOV capabilities.

Hardware channels for stimulus traffic

As a MRIOV capable component, Exerciser supports the following three physical functions:

- Function A
- Function B
- Function C

The function A is MRIOV capable. Function B and C are SRIOV capable and have two virtual functions VF1 and VF2 associated with each of them making it a total of seven functions.

If you have the U4305A-024 or U4305U-024 software license, then two additional physical functions (Function D and E) are provided making it a total of 13 physical and virtual functions and 13 hardware channels.

You can define a single stimulus packet or a block of stimulus packets separately for each of these physical and virtual functions. On starting the stimulus, Protocol Exerciser sends the packets that you added for a function using the hardware channel that corresponds to that function.

Protocol Exerciser sends the block of packets defined for a function in the sequence in which you add these packets to the function. However, if you defined block of packets for multiple functions, then the sequence in which the blocks from different functions are sent is decided by Protocol Exerciser based on factors such as the credit limit available for the applicable virtual hierarchy.

By default, all the functions are enabled. However, you can choose to send stimulus either from one of these functions or from all the functions in which you added stimulus traffic.

Virtual Hierarchies and Functions

Protocol Exerciser in its emulation as an MRIOV capable PCIe component supports three virtual hierarchies VH0 to VH2 for the three physical functions. If you have the U4305A-024 or U4305U-024 software license, then two additional virtual hierarchies are provided for physical functions (Function D and E) making it a total of 5 virtual hierarchies.

The following table displays the mapping between the virtual hierarchies and functions..

Hierarchy	Functions
Virtual Hierarchy 0 (VH0)	• Function A as the base function (BF 0) with MRIOV capabilities.
Virtual Hierarchy 1 (VH1)	• Function B as the Physical Function with SRIOV capabilities. • Two virtual functions VF1 and VF2 associated with Function B.
Virtual Hierarchy 2 (VH2)	• Function C as the Physical Function with SRIOV capabilities. • Two virtual functions VF1 and VF2 associated with Function C.
Virtual Hierarchy 3 (VH3)	• Function D as the Physical Function with SRIOV capabilities. • Two virtual functions VF1 and VF2 associated with Function D.
Virtual Hierarchy 4 (VH4)	• Function E as the Physical Function with SRIOV capabilities. • Two virtual functions VF1 and VF2 associated with Function E.

Exerciser supports one virtual link VL0 and one virtual channel VC0. All the hierarchies are mapped to this virtual link and virtual channel. Therefore, all the packets added to any of these functions are associated with VC0 and VL0.

The function to which you add a stimulus packet defines the virtual hierarchy to which the packet belongs. For instance, if you added a packet to Function B, then this packet belongs to virtual hierarchy VH1.

Each function is assigned a unique number which is displayed in the **Function status** tab of the **Hardware status** pane.

Virtual channels for stimulus traffic

Protocol Exerciser supports and uses one virtual channel (VC0) for each virtual hierarchy.

Depending on the type of flow control implemented, Exerciser decides the virtual channel to be used for sending a stimulus packet. If VL based flow control is implemented, then all virtual hierarchies use the same flow control for sending packets. If per VH flow control is implemented, then each virtual hierarchy has its own flow control. However, packets are sent through VC0 for all hierarchies. Refer to Initializing Flow Control on page 79 to know more about flow controls.

Defining Stimulus Traffic

This topic describes how to define the stimulus traffic that Protocol Exerciser can send over a PCIe link to a DUT.

You use the **Traffic Setup** page in the Protocol Exerciser GUI to define the stimulus traffic. Using this page, you can add a single packet or a block of packets to be sent as stimulus.

Functions

For an SRIOV / MRIOV capable Exerciser, you can add stimulus traffic to any of the physical and virtual functions supported by Exerciser. For a non IOV Exerciser, you can add stimulus traffic to any of the non IOV functions supported by Exerciser. To know about the virtual hierarchies, functions, corresponding hardware channels, and virtual channels of Protocol Exerciser used for stimulus, refer to the topics Functions, Hardware Channels, Virtual Hierarchies, and Virtual Channels on page 22 and Stimulus Traffic - Overview on page 130.

The following screens display how the functions are listed in the Protocol Exerciser GUI for defining the stimulus traffic for each of these functions.

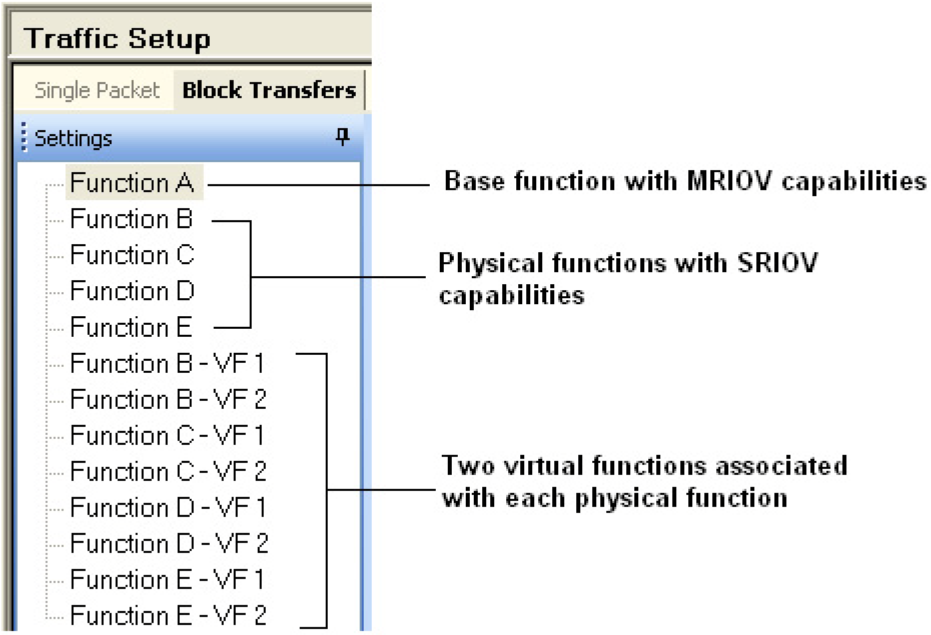


Figure 20 Functions for an MRIOV capable Exerciser

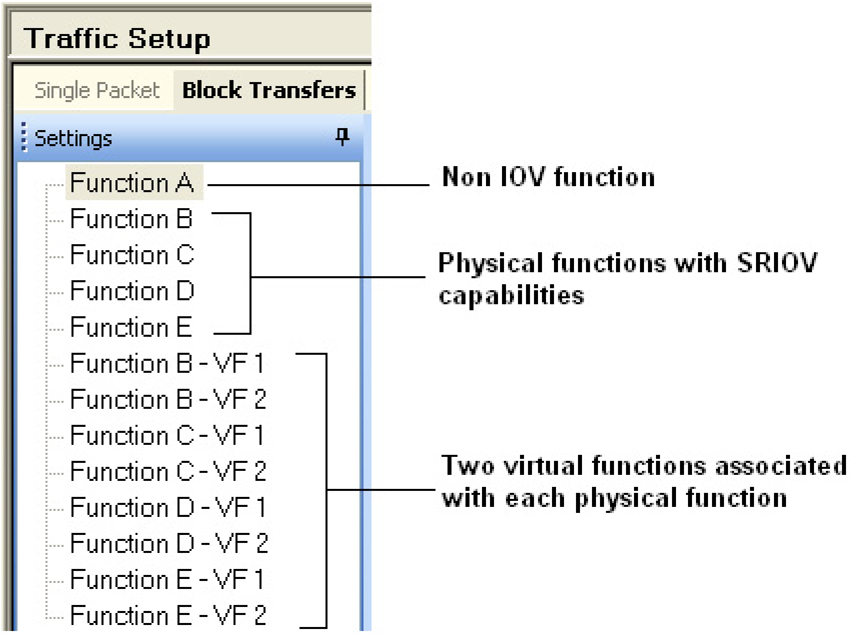


Figure 21 Functions for a SRIOV Exerciser

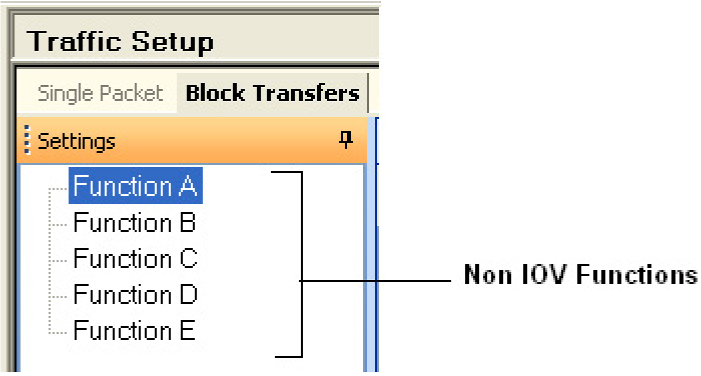


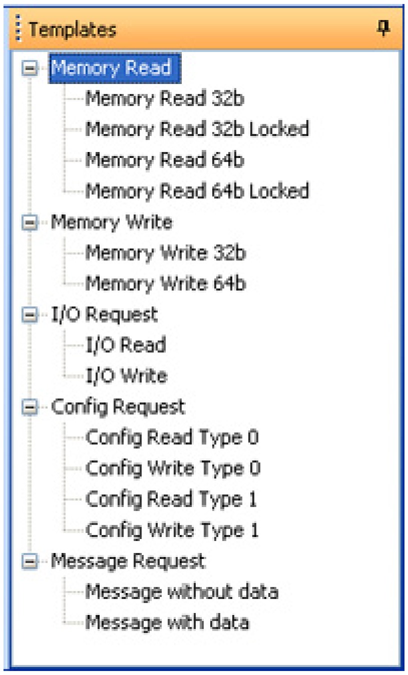
Figure 22 Functions for a non IOV Exerciser

NOTE

The functions D and E and their corresponding virtual functions (when applicable) are displayed only when you have the additional 2 physical functions license of Protocol Exerciser. Otherwise, only three functions and their corresponding virtual functions are displayed.

PCIe Traffic Templates

Protocol Exerciser provides a set of PCIe packet templates in the Protocol Exerciser GUI. You can add instances of these templates in the Protocol Exerciser GUI to send these as PCIe packets over the configured link. The following screen displays these templates.



Templates

- Memory Read
  - Memory Read 32b
  - Memory Read 32b Locked
  - Memory Read 64b
  - Memory Read 64b Locked
- Memory Write
  - Memory Write 32b
  - Memory Write 64b
- I/O Request
  - I/O Read
  - I/O Write
- Config Request
  - Config Read Type 0
  - Config Write Type 0
  - Config Read Type 1
  - Config Write Type 1
- Message Request
  - Message without data
  - Message with data

Figure 23 PCIe traffic templates

The templates are categorized as:

Memory	To transfer data to or from a memory mapped location
IO	To transfer data to or from an I/O mapped location
Configuration	To perform device configuration/setup
Message	Event signal mechanism to general purpose message

Defining the PCIe packets to be sent as stimulus

For each of the functions, you can define TLP requests to be sent as stimulus to DUT. These packets are added as records in the memory allocated for the hardware channel corresponding to that function. The block of packets added to a function is sent in the defined sequence to DUT using this hardware channel.

NOTE

For an MRIOV capable Exerciser:

If you want to add stimulus traffic to a function which is a part of a virtual hierarchy 1 to 4 (VH1 - VH4), then you need to ensure that you enable that hierarchy in the Virtual Channels page. If you do not enable that hierarchy, then packets added to functions that are a part of that hierarchy will not be transmitted. Refer to the topic Initializing Flow Control on page 79_know more.

You use the **Block Transfers** tab in the **Traffic Setup** page to add a block of PCIe packets in a sequence.

To add a block of PCIe packets for stimulus:

1 Select the function from the **Settings** pane for which you want to define the stimulus traffic.

2 Select the mode in which you want the block of packets to be sent. If you want to send the block of packets once, select the **Single** mode. If you want to send the block of packets repeatedly in a sequence until you manually stop the stimulus, click the **Continuous** mode.

3 Drag and drop a PCIe packet from the **Templates** section to the **Send Block Transfers** tab. Each packet in the block is added as a row with an incremental Line number.

You can add blocks of PICe packets in a similar manner to other functions supported by Exerciser. The sequence in which these blocks added to different functions are sent as stimulus by Exerciser depends on the applicable decoder for the requests.

NOTE

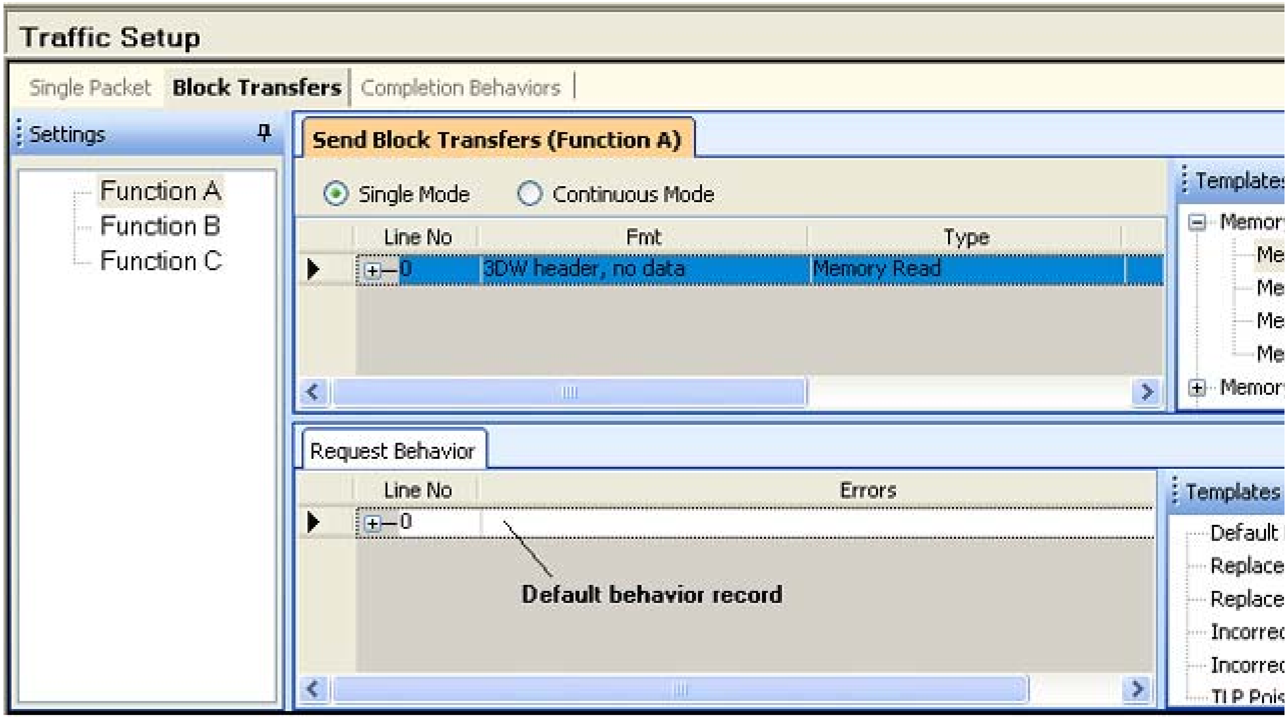
You can save the stimulus traffic setup in a Protocol Exerciser setup file. To do this, click **File** -> **Save As** and save the setup in a .exs file.

Defining the behavior of the PCIe packets to be sent as stimulus

For each packet added to a function, you can also define the instructions on how to send that packet as stimulus. This intended behavior of PCIe packets is stored as behavior records in the memory of Protocol Exerciser.

You use the **Request Behavior** tab in the **Traffic Setup** page to add request behavior records for PCIe packets added to a function. In this tab, Protocol Exerciser provides a list of templates using which you can add different behavior records for different packets.

When you add the first PCIe packet to a function, a request behavior record is automatically added for that packet as per the Default Behavior template available for behavior records. The following screen displays this default request behavior record.



Line No	Fmt	Type
0	3DW header, no data	Memory Read

Line No	Errors
0	

Default behavior record

Figure 24 Default request behavior record

This behavior record is stored as the first record in the memory and is assigned the Line number 0. If you do not add any more request behavior records, then this behavior record is used with all the packets added for that function. You can manually add more behavior records by dragging and dropping a behavior template from the **Templates** pane to the **Request Behavior** tab. Each behavior record is assigned an incremental line number. If the behavior records that you add are equal to the packets that you added to the function, then there is a one to one mapping between the behavior records and packets. Line numbers of packets and behavior records are mapped in this case. However, if the behavior records are less than the number of packets, then the behavior records are applied as follows:

Total number of request behavior records: 3

TLP packets: 5

TLP packet 1	As per request behavior record 1
TLP packet 2	As per request behavior record 2
TLP packet 3	As per request behavior record 3
TLP packet 4	As per request behavior record 1
TLP packet 5	As per request behavior record 2

If you have configured a request packet's settings to repeat its transmission for a specified number of times, then the same behavior record is applicable for all its repetitions.

When you add a behavior record, the settings in this record are as per the behavior template that you used. If needed, you can change these default settings of a behavior record. To do this, double-click the behavior record and specify the new settings in the Edit Packet Behavior dialog box. Refer to the topic Edit Packet Behavior dialog box on page 190 to get a description of each of the field available in this dialog box.

Changing the default settings of PCIe packets

You can change the default settings of the PCIe packets that you added in the Send Block Transfers tab. To do this:

1 Double-click a PCIe packet.

The Edit Packet dialog box is displayed with the current settings for the packet. To get a description of each field in this dialog box, refer to the topic Edit Packet dialog box. In this dialog box, the following are some of the settings that control how the packet is sent as stimulus:

Repeating the transmission of a request packet

You can repeat the transmission of a packet until the pattern match that you specified using the General Settings->Pattern Matcher tab is found in the incoming TLPs from DUT. You can also specify how many times you want to repeat the transmission of a request packet while sending a block of packets as stimulus from Exerciser. You use the **Repeat Request** field in the **Edit Packet** dialog box to specify this repetition for a packet.

Delaying the transmission of a request packet

You can delay the transmission of a request packet as stimulus until one of the following conditions is met:

- The pattern match that you specified using the General Settings->Pattern Matcher tab is found in the incoming TLPs from DUT.
- An external trigger in signal is received from another test equipment on the Trigger In Connector component of the Exerciser card.
- Exerciser has sent all the outstanding completions.

Payload of a request packet

For a request packet with data (such as a memory write request), you can specify the start address in the data memory of Exerciser from where the payload of such a request should be taken. You use the **Read Data From** group box in the **Edit Packet** dialog box of the request packet to specify this start address.

Payload of completion from DUT

For a request packet that requires a completion packet with data from DUT such as a memory read request, you can specify the start address in the data memory of Exerciser where this completion data for such a request should be stored. You use the **Write Data To** group box in the **Edit Packet** dialog box of the request packet to specify this start address. When the completions is received, Exerciser stores the completion data at the specified memory address.

Starting the Stimulus and Viewing Results

Once a PCIe link is created between the Protocol Exerciser and DUT, you can start the stimulus as per the defined stimulus requests that you added in the **Traffic Setup** -> **Block transfers** tab.

Before you start

Before you start sending stimulus packets, ensure that:

- The link state is displayed as **Active** in the Hardware status pane before starting the stimulus.
- You have enabled the relevant virtual hierarchy (VH1-VH4) of Protocol Exerciser if you have added stimulus traffic to a function that is not a part of VH0. For instance, if VH2 is disabled and you add packets to a function that is a part of VH2, then such a packet will not be transmitted, To enable VH2, you use the **Virtual Channel** page. This is applicable if Exerciser emulates an MRIOV capable component.
- You have enabled the VC Resource 1 (VCx) of Protocol Exerciser if you have added stimulus traffic to a function mapped to VCx in TC VC mapping. If VCx is disabled, then VC0 will be used to transmit even those packets for which VCx is applicable as per the TC VC mapping. To enable VCx, you use the **Virtual Channel** page. This is applicable if Exerciser emulates a non IOV/SRIOV component.

To start the stimulus for the block(s) of packets

You can send the packets added to any of the supported functions of Exerciser by clicking the ▶ **Run Exerciser** toolbar button. For instance, if you added blocks of requests to Function A and C, then clicking this button sends all the packets that you added to both the functions.

NOTE

The stimulus requests are not necessarily sent in the same order in which you have added these requests across the functions. However, within a function, packets are sent in the same sequence in which you added these as a block to the function.

You can also send only those packets that have been added to a specific function. For instance, if you want to send only those packets that you added to Function A, then click the drop down button displayed with the ▶ **Run Exerciser** toolbar button and then select the **Start Function A** option. Packets added to other functions are not transmitted in this case. The following screen displays the options available to start stimulus for a specific function.

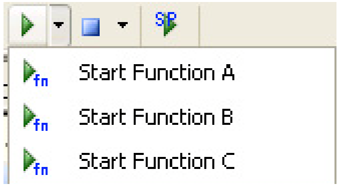


Figure 25 Start stimulus for a function

If you have configured the stimulus traffic block to be repeatedly sent in a Continuous mode, you need to manually stop the stimulus using the 🟥 **Stop Exerciser** toolbar button. You can also stop the stimulus for a specific function only by clicking the drop down button displayed with the 🟥 toolbar button and then select the **Stop Function <No>** option. Stimulus added to other functions is not stopped in this case. The following screen displays the options available to stop stimulus for a specific function.

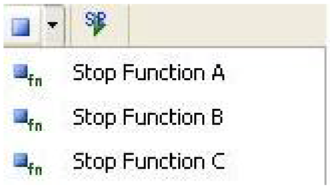
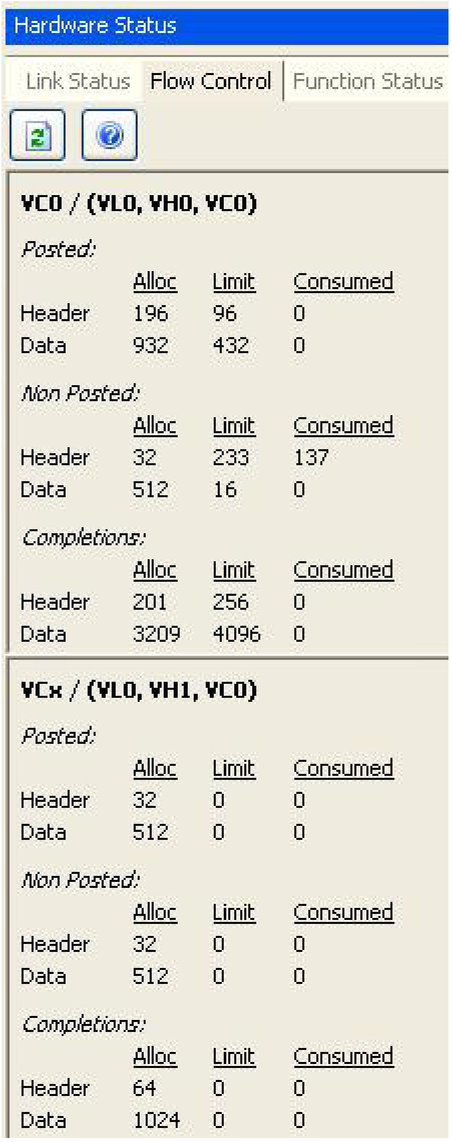


Figure 26 Stop stimulus for a function

Viewing the stimulus results

You can view the stimulus results in the **Hardware status** pane.



VC0 / (VL0, VH0, VC0)

Posted:

	Alloc	Limit	Consumed
Header	196	96	0
Data	932	432	0

Non Posted:

	Alloc	Limit	Consumed
Header	32	233	137
Data	512	16	0

Completions:

	Alloc	Limit	Consumed
Header	201	256	0
Data	3209	4096	0

VCx / (VL0, VH1, VC0)

Posted:

	Alloc	Limit	Consumed
Header	32	0	0
Data	512	0	0

Non Posted:

	Alloc	Limit	Consumed
Header	32	0	0
Data	512	0	0

Completions:

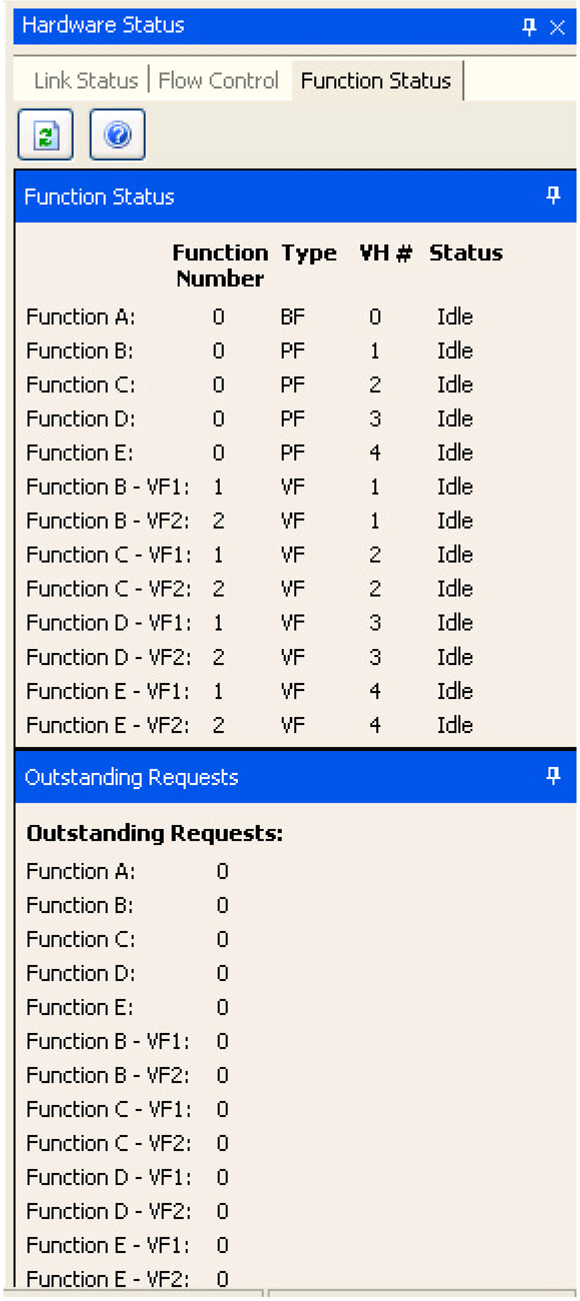
	Alloc	Limit	Consumed
Header	64	0	0
Data	1024	0	0

Figure 27 Flow control status after stimulus

On starting the stimulus, the available credits for the virtual channels of Protocol Exerciser get updated in the **Flow Control** tab of the Hardware status pane. In this pane, the following values are displayed for the flow control credits of Exerciser:

- **Alloc** - Displays the flow control credit values that you allocated for the transmission of requests and completions from Exerciser. The allocated credits are for each virtual channel of Exerciser and are categorized into posted and non posted request headers, posted and non posted request data, and completion headers and data transmitted by Exerciser. You allocate these credits using the **Virtual Channel** page of the Exerciser GUI.
- **Limit** - Displays the flow control credit limit currently available from the allocated credit. This limit is updated based on the transmission from Exerciser.
- **Consumed** - Displays the flow control credit consumed from the allocated credit by Exerciser. This value is updated based on the transmission from Exerciser.

To view the current status of flow control credits, click the ⟳ button.



Function Status

	Function Number	Type	VH #	Status
Function A:	0	BF	0	Idle
Function B:	0	PF	1	Idle
Function C:	0	PF	2	Idle
Function D:	0	PF	3	Idle
Function E:	0	PF	4	Idle
Function B - VF1:	1	VF	1	Idle
Function B - VF2:	2	VF	1	Idle
Function C - VF1:	1	VF	2	Idle
Function C - VF2:	2	VF	2	Idle
Function D - VF1:	1	VF	3	Idle
Function D - VF2:	2	VF	3	Idle
Function E - VF1:	1	VF	4	Idle
Function E - VF2:	2	VF	4	Idle

Outstanding Requests

Outstanding Requests:

Function A:	0
Function B:	0
Function C:	0
Function D:	0
Function E:	0
Function B - VF1:	0
Function B - VF2:	0
Function C - VF1:	0
Function C - VF2:	0
Function D - VF1:	0
Function D - VF2:	0
Function E - VF1:	0
Function E - VF2:	0

Figure 28 Function status after stimulus

You can check the **Outstanding Requests** section of the **Function status** tab to see if there are any TLP requests sent from Exerciser which have not yet completed. If there are any requests transmitted from a function which have not yet completed, the tags assigned to such requests are not released until these requests are completed. The **Outstanding**

Requests section then displays a value for each function indicating the number of these tags which have not been released due to outstanding requests. If all the requests sent from a function are completed, the **Outstanding Requests** value is displayed as **0**.

To view the current status of outstanding requests for each function, click the button.

You can also view the contents of the data memory of Exerciser to check the results of requests requiring memory access. For instance, you can view the completion data received for a memory read request sent from Exerciser or the results of a memory write request sent from DUT. You use the **Data Memory** page to view the memory contents.

Refer to the topic Stimulus - Example on page 149 to get an example of a stimulus sent to DUT and results displayed in the Protocol Exerciser GUI.

NOTE

In this release, you can use the Gen3 Exerciser APIs to get the real-time stimulus statistics such as:

- Number of TLPs transmitted and received by Exerciser

- ACKs and NACKs received from DUT

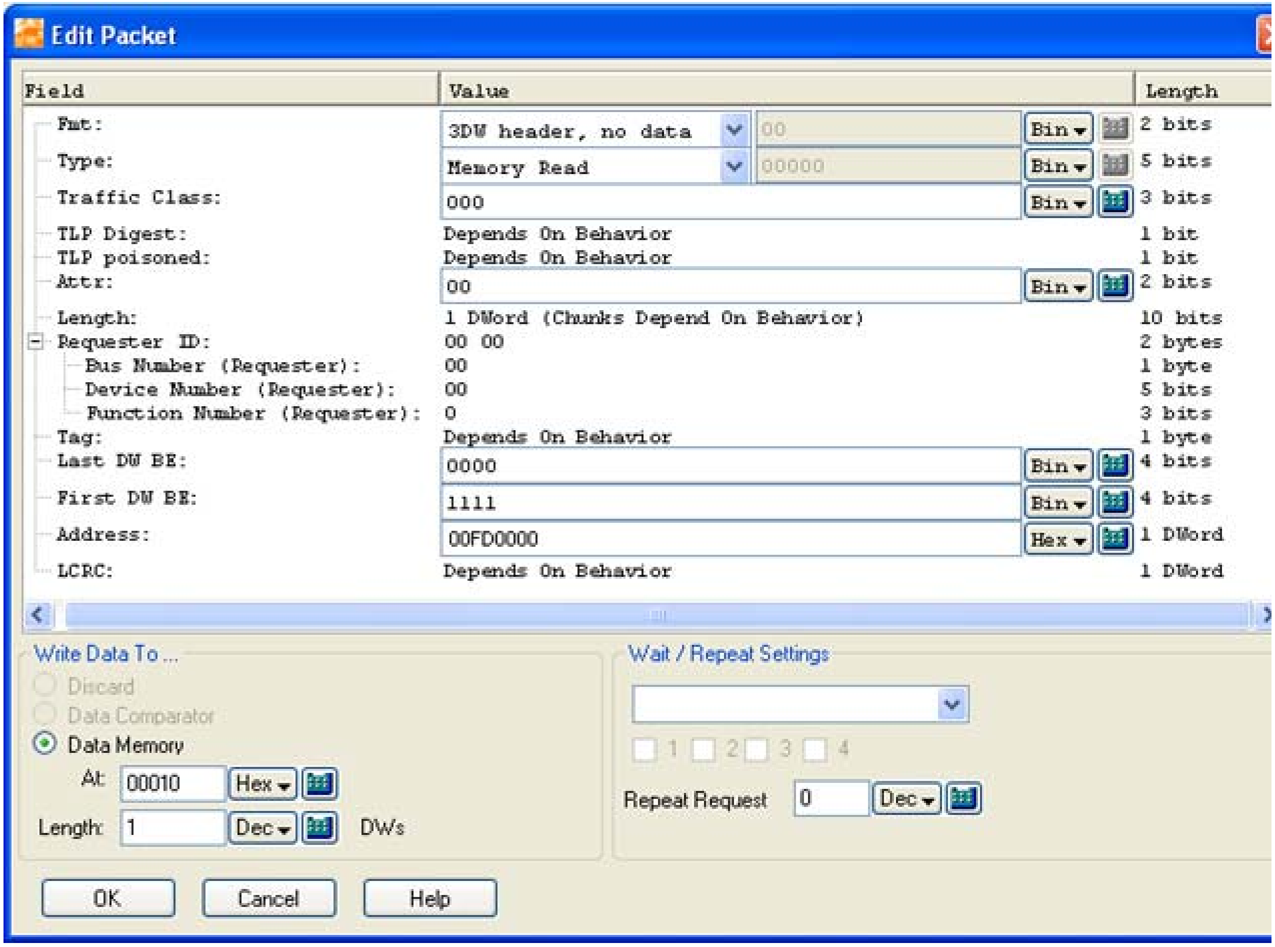
- Number and type of errors (belonging to different categories) received from DUT

You can read the EPCIEPerformanceCounterStatus, EPCIEPerformanceCounterFunctionStatus, and EPCIEPerformanceCounterVCStatus property values to get the stimulus statistics. Refer to the API help to get further details on these APIs.

Stimulus - Example

This topic provides an example of how to send a stimulus request to DUT and viewing its completion received from DUT.

In this example, Exerciser emulates a non IOV PCIe component. A 32 bit Memory Read request has to be sent from Exerciser to DUT. The following screen displays the packet settings done for this memory read request to read from the memory address 00FD0000. The completion data for this read request is configured to be stored at the memory address 000010 in the data memory of Exerciser.



Field	Value			Length
Fmt:	3DW header, no data	00	Bin	2 bits
Type:	Memory Read	00000	Bin	5 bits
Traffic Class:	000		Bin	3 bits
TLP Digest:	Depends On Behavior			1 bit
TLP poisoned:	Depends On Behavior			1 bit
Attr:	00		Bin	2 bits
Length:	1 DWord (Chunks Depend On Behavior)			10 bits
Requester ID:	00 00			2 bytes
Bus Number (Requester):	00			1 byte
Device Number (Requester):	00			5 bits
Function Number (Requester):	0			3 bits
Tag:	Depends On Behavior			1 byte
Last DW BE:	0000		Bin	4 bits
First DW BE:	1111		Bin	4 bits
Address:	00FD0000		Hex	1 DWord
LCRC:	Depends On Behavior			1 DWord

Write Data To ...

○ Discard
○ Data Comparator
● Data Memory

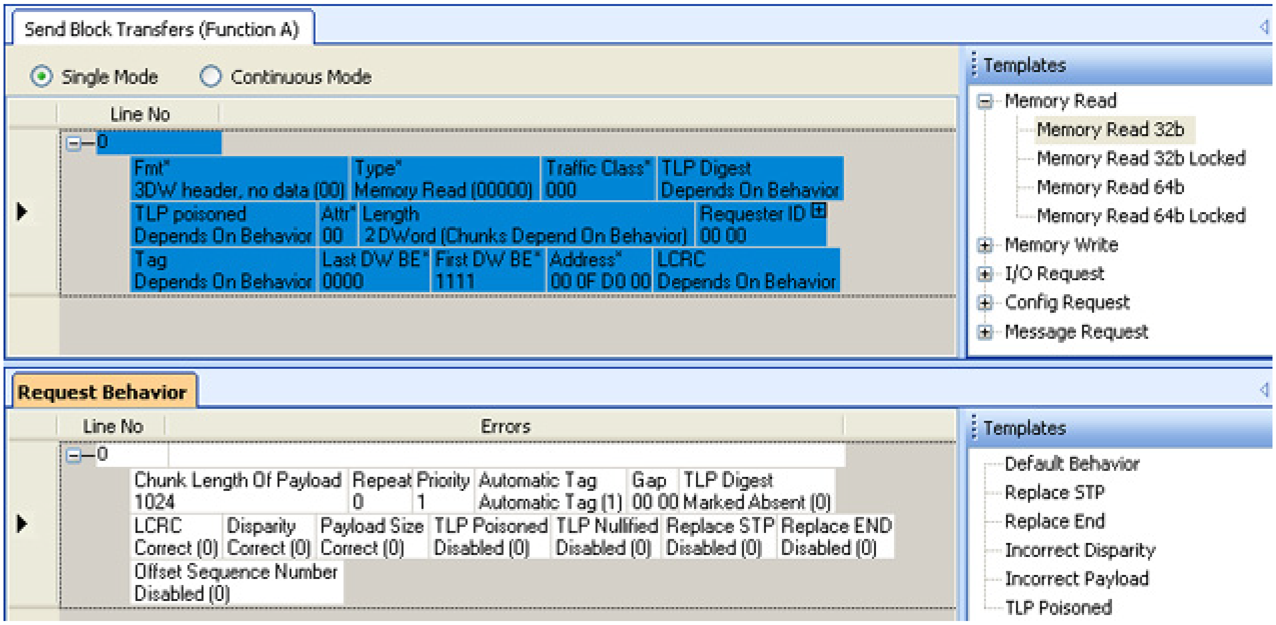
At: 00010 Hex
Length: 1 Dec DWs

Wait / Repeat Settings

□ 1 □ 2 □ 3 □ 4

Repeat Request 0 Dec

The default behavior has been used for the memory read request packet. The following screen displays this default request behavior record added for the request.



Send Block Transfers (Function A)

● Single Mode ○ Continuous Mode

Line No

0

Fmt*	Type*	Traffic Class*	TLP Digest
3DW header, no data (00)	Memory Read (00000)	000	Depends On Behavior

TLP poisoned	Attr*	Length	Requester ID
Depends On Behavior	00	2 DWord (Chunks Depend On Behavior)	00 00

Tag	Last DW BE*	First DW BE*	Address*	LCRC
Depends On Behavior	0000	1111	00 0F D0 00	Depends On Behavior

Request Behavior

Line No Errors

0

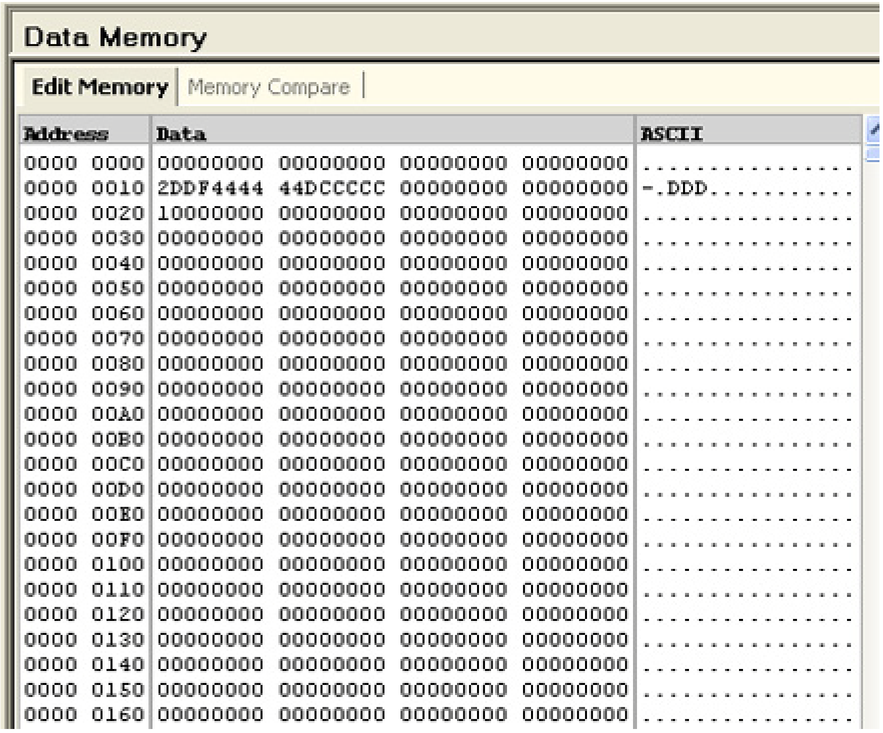
Chunk Length Of Payload	Repeat	Priority	Automatic Tag	Gap	TLP Digest
1024	0	1	Automatic Tag (1)	00 00	Marked Absent (0)

LCRC	Disparity	Payload Size	TLP Poisoned	TLP Nullified	Replace STP	Replace END
Correct (0)	Correct (0)	Correct (0)	Disabled (0)	Disabled (0)	Disabled (0)	Disabled (0)

Offset Sequence Number
Disabled (0)

Once the stimulus traffic setup is ready, the memory read request is sent as stimulus to DUT. The flow control credits of Exerciser are updated accordingly and displayed in the Flow control tab of the Hardware Status pane.

In the following screen, the contents of the data memory of DUT (another Exerciser in this example) are displayed after the transaction is completed. The completion data received from DUT for the memory read request is stored at the address specified in the request packet.

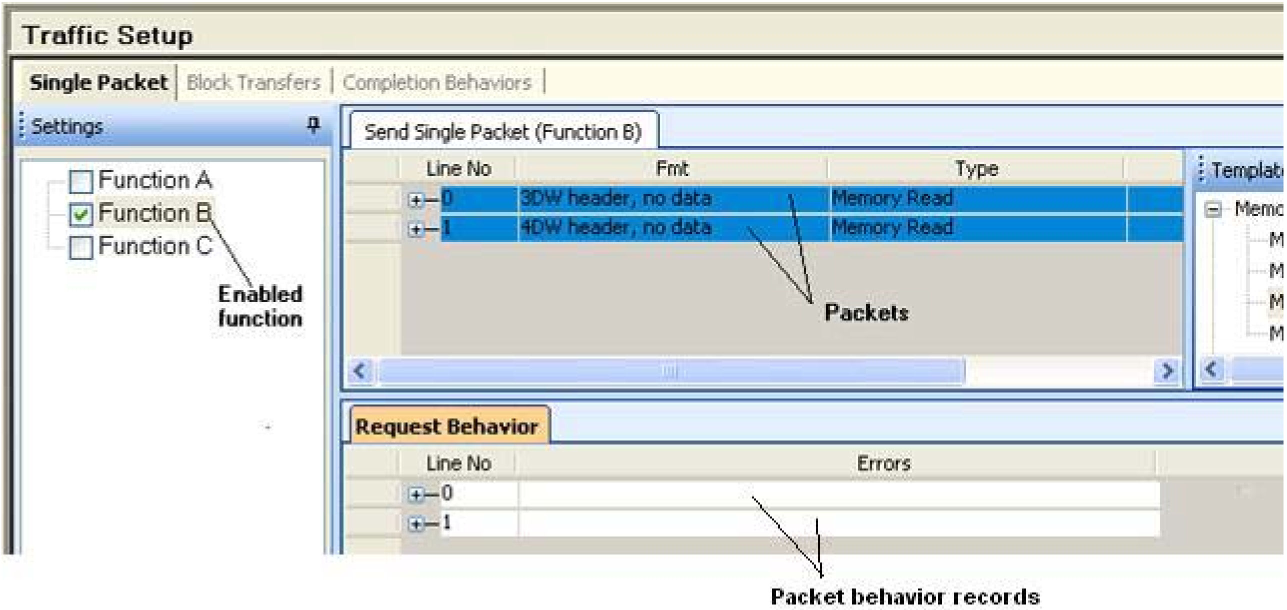


```
Address    Data                                  ASCII
0000 0000 00000000 00000000 00000000 00000000 ................
0000 0010 2DDF4444 44DCCCCC 00000000 00000000 -.DDD...........
0000 0020 10000000 00000000 00000000 00000000 ................
0000 0030 00000000 00000000 00000000 00000000 ................
0000 0040 00000000 00000000 00000000 00000000 ................
0000 0050 00000000 00000000 00000000 00000000 ................
0000 0060 00000000 00000000 00000000 00000000 ................
0000 0070 00000000 00000000 00000000 00000000 ................
0000 0080 00000000 00000000 00000000 00000000 ................
0000 0090 00000000 00000000 00000000 00000000 ................
0000 00A0 00000000 00000000 00000000 00000000 ................
0000 00B0 00000000 00000000 00000000 00000000 ................
0000 00C0 00000000 00000000 00000000 00000000 ................
0000 00D0 00000000 00000000 00000000 00000000 ................
0000 00E0 00000000 00000000 00000000 00000000 ................
0000 00F0 00000000 00000000 00000000 00000000 ................
0000 0100 00000000 00000000 00000000 00000000 ................
0000 0110 00000000 00000000 00000000 00000000 ................
0000 0120 00000000 00000000 00000000 00000000 ................
0000 0130 00000000 00000000 00000000 00000000 ................
0000 0140 00000000 00000000 00000000 00000000 ................
0000 0150 00000000 00000000 00000000 00000000 ................
0000 0160 00000000 00000000 00000000 00000000 ................
```

Sending a Single Packet as Stimulus and Viewing Response

Besides sending a sequence of PCIe packets as stimulus, you can also send a single packet as stimulus to DUT and view the completion received from DUT in response to this stimulus.

You use the **Single Packet** tab in the **Traffic Setup** page to send a single packet and view its response. In this tab, the functions of Protocol Exerciser (that correspond to hardware channels) are displayed. Out of the three functions (3 hardware channels available for a non IOV Exerciser and seven functions available for an MRIOV capable Exerciser, you can activate/enable one function at a time for sending single packets. The hardware channel corresponding to this function is then used for sending the packet added to this function. The packet that you add to other disabled functions are not sent as stimulus.



Line No	Fmt	Type
0	3DW header, no data	Memory Read
1	4DW header, no data	Memory Read

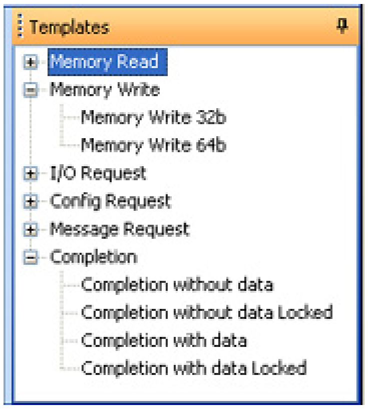
Line No	Errors
0	
1	

In the **Single Packet** tab, when you add a PCIe packet to send as stimulus, a corresponding record is automatically added to define the behavior of that PCIe packet. This behavior definition controls how Protocol Exerciser sends the added packet as stimulus. You can change this default behavior of the added packet, if required.

The PCIe packet that you add as stimulus is assigned a sequenced Line number. The intended behavior record automatically added for this packet is also assigned the same Line number to map it to the packet for which it is added. The screen displayed above has two PCIe packets added as stimulus and a corresponding behavior record for each of these two packets with the same Line numbers as the packets (0 and 1).

The single packet requests and behaviors that you add in the Single Packet tab are written to the Single Packet memory. For a single packet read request, you can specify whether the completion data of this request should be written to Single packet memory or a specific location in the data memory of Protocol Exerciser. For a single packet write request, you can specify whether the payload for this write request should be read from the single packet memory or a specific location in the data memory of Protocol Exerciser.

You can use the Single Packet tab to send a request as well as a completion as a single packet. The templates for these packets are available in the **Templates** pane as displayed in the following screen.



To send a single packet as stimulus:

1 Click the **Traffic Setup** icon displayed in the navigation pane of the Protocol Exerciser GUI.

2 Click the **Single Packet** tab in the **Traffic Setup** page.

3 From the **Settings** pane, select the function of the Protocol Exerciser for which you want to add the single packet as stimulus. The functions that are displayed depends on whether you have configured the Protocol Exerciser as a non IOV or MRIOV capable PCIe

component. The hardware channel corresponding to this function is used for sending the packet.

4 Enable/activate the selected function by selecting the checkbox displayed with the function. Protocol Exerciser does not send the packet added to a disabled function as stimulus.

5 From the **Templates** pane, drag and drop a PCIe packet to the **Send single packet** tab. You can also send a completion to DUT by adding an instance of a completion template.

   You can view the details of the added packet by clicking the + icon displayed with that packet. If required, you can change the value for the fields in the packet by double-clicking the packet.

6 A behavior record is automatically added for the packet in the **Request Behavior** tab. This record has the same line number as assigned to the packet. You can view the details of the behavior record added for the packet by clicking the + icon displayed with that behavior record. If required, you can change the default behavior of the packet by double-clicking its corresponding behavior record.

7 If you have added multiple packets in the **Send single packet** tab, select the packet that you want to send from the list of packets added. Even if you have added multiple packets, only the selected packet will be sent as stimulus.

8 Click the **Send Single Packet** toolbar button.

NOTE You can send multiple PCIe packets as stimulus using the Block Transfers tab in the Traffic Setup page.

To view the completion received from DUT in response to the single packet sent as stimulus:

1 Click the **Single Packet** tab in the **Traffic Setup** page.

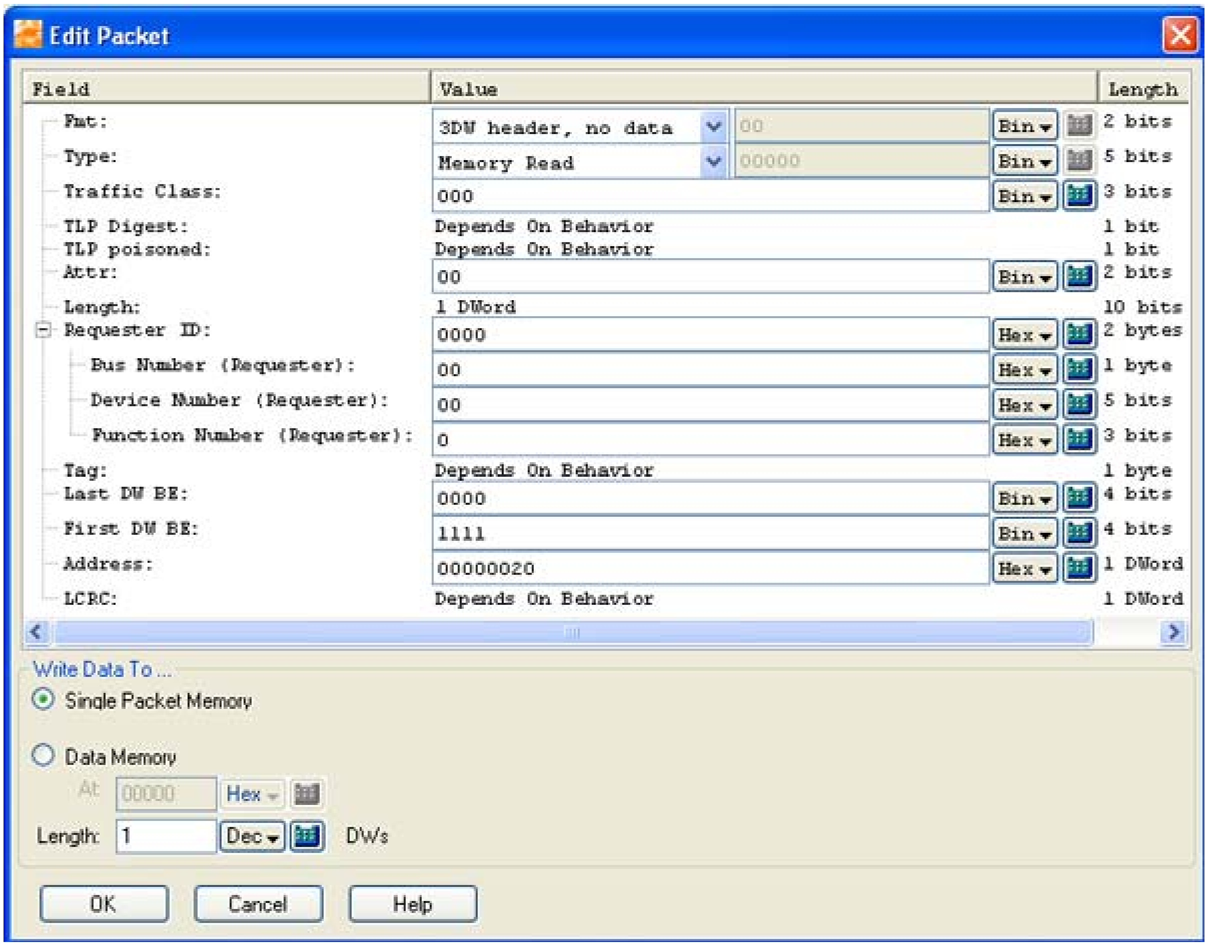
2 Select the function for which you added the PCIe packet sent as a single packet.

The completions received from DUT in response to the single packet stimulus are displayed in the **Received Completions** tab. In this tab, each completion record is assigned the same Line number as assigned to its corresponding packet in the

Send single packet tab. The line number maps the completion record received from DUT with the stimulus packet for which the completion is received.

Sending a single packet - Example

In the following example, Exerciser sends a read request as a single packet to read from the address 00000020. The completion data for this read request has been configured to be saved in the single packet memory of Exerciser instead of the data memory of Exerciser.



Field	Value			Length
Fmt:	3DW header, no data	00	Bin	2 bits
Type:	Memory Read	00000	Bin	5 bits
Traffic Class:	000		Bin	3 bits
TLP Digest:	Depends On Behavior			1 bit
TLP poisoned:	Depends On Behavior			1 bit
Attr:	00		Bin	2 bits
Length:	1 DWord			10 bits
Requester ID:	0000		Hex	2 bytes
Bus Number (Requester):	00		Hex	1 byte
Device Number (Requester):	00		Hex	5 bits
Function Number (Requester):	0		Hex	3 bits
Tag:	Depends On Behavior			1 byte
Last DW BE:	0000		Bin	4 bits
First DW BE:	1111		Bin	4 bits
Address:	00000020		Hex	1 DWord
LCRC:	Depends On Behavior			1 DWord

Write Data To ...

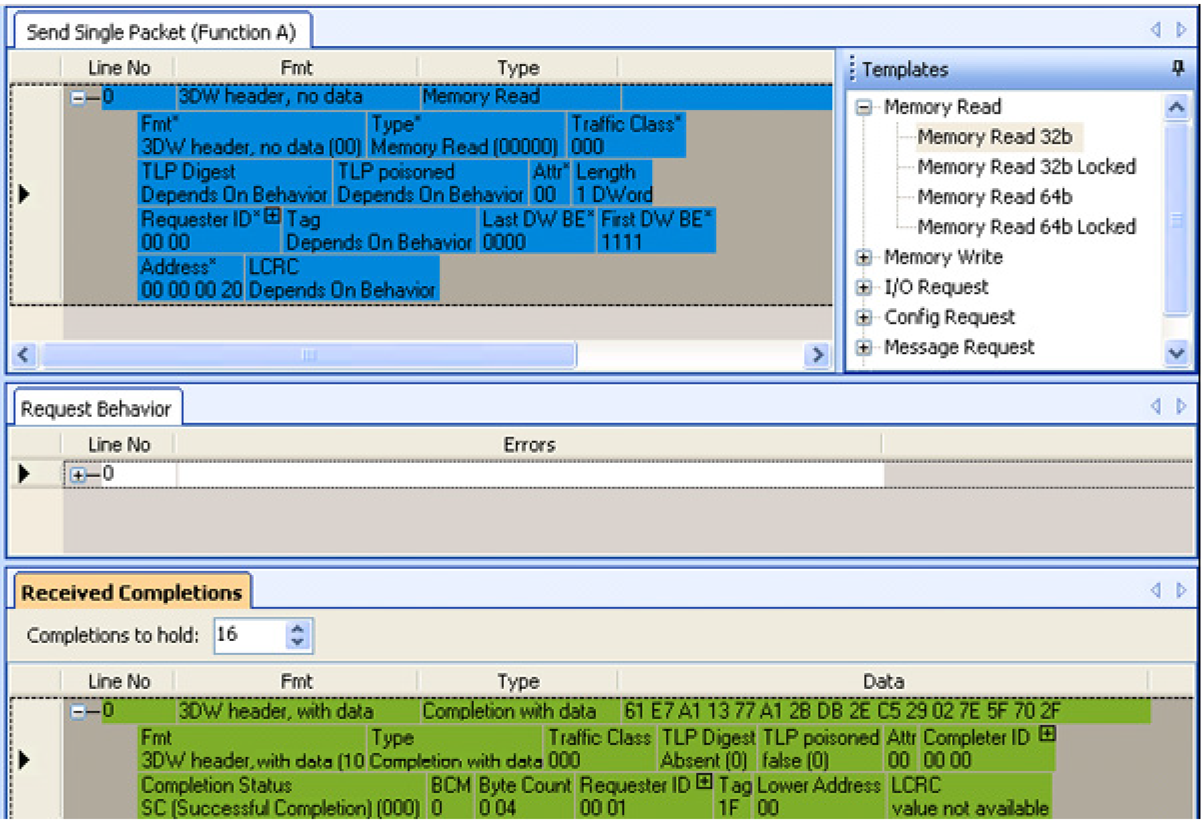
● Single Packet Memory

○ Data Memory

At: 00000 Hex

Length: 1 Dec DWs

On sending the read request, the completion with data received from DUT is displayed in the following screen.



Line No	Fmt	Type
0	3DW header, no data	Memory Read

Fmt*	Type*	Traffic Class*
3DW header, no data (00)	Memory Read (00000)	000

TLP Digest	TLP poisoned	Attr*	Length
Depends On Behavior	Depends On Behavior	00	1 DWord

Requester ID*	Tag	Last DW BE*	First DW BE*
00 00	Depends On Behavior	0000	1111

Address*	LCRC
00 00 00 20	Depends On Behavior

Line No	Errors
0	

Completions to hold: 16

Line No	Fmt	Type	Data
0	3DW header, with data	Completion with data	61 E7 A1 13 77 A1 2B DB 2E C5 29 02 7E 5F 70 2F

Fmt	Type	Traffic Class	TLP Digest	TLP poisoned	Attr	Completer ID
3DW header, with data (10	Completion with data	000	Absent (0)	false (0)	00	00 00

Completion Status	BCM	Byte Count	Requester ID	Tag	Lower Address	LCRC
SC (Successful Completion) (000)	0	0 04	00 01	1F	00	value not available

You can control the number of completion records received from DUT that Exerciser should retain and display in the **Received Completions** tab. You can specify this number in the **Completions to hold** text box in the **Received Completions** tab.

Sending Completions as Response to DUT

Protocol Exerciser sends completions to DUT in response to requests sent by DUT over a configured link. You can define the behavior of these completions and then based on these behaviors, Exerciser generates and sends completions automatically.

You use the **Completion Behaviors** tab in the **Traffic Setup** page of the Protocol Exerciser GUI to define the behavior for these completion packets sent by Exerciser.

NOTE

For testing purposes, you can also send a single completion packet as stimulus to DUT. You can do this using the Single Packet tab in the Traffic Setup page. This tab has a set of completion templates that you can use to send a completion packet.

Completion Queues

Exerciser supports completion queues to send completion packets as responses to DUT.

If Exerciser emulates a non IOV or a SRIOV component, it supports two completion queues, **Queue 0** and **Queue 1**. These queues are mapped respectively to the two virtual channels (VC0 and VCx) that Exerciser supports. Therefore, Exerciser adds the completion packets to be sent through VC0 to Queue 0 and the completion packets to be sent through VCx to Queue1. Exerciser decides the VC for a completion based on the traffic class of the request for which completion is sent, the function applicable for this request, and the TC VC mapping that you defined for this function.

If Exerciser emulates an MRIOV capable component, it supports three completion queues, Queue 0 to Queue 2. A completion queue is mapped to a virtual hierarchy that Exerciser supports. For instance, Queue0 is mapped to VH0 and so on. All the completion queues use VL0 as the virtual link and VC0 as the virtual channel. If you have installed the additional 2 physical function license of Exerciser, then Exerciser provides five completion queue, Queu0 to Queue4, mapped to five functions.

In the **Completion Behaviors** tab of the **Traffic Setup** page, you can add completion behaviors to any of the supported completion queues. The completion behaviors that you add to a queue are added as records to the memory allocated for that queue. Exerciser maps these behavior records added to a queue to the completion packets added to that queue for setting the behavior of these completion packets.

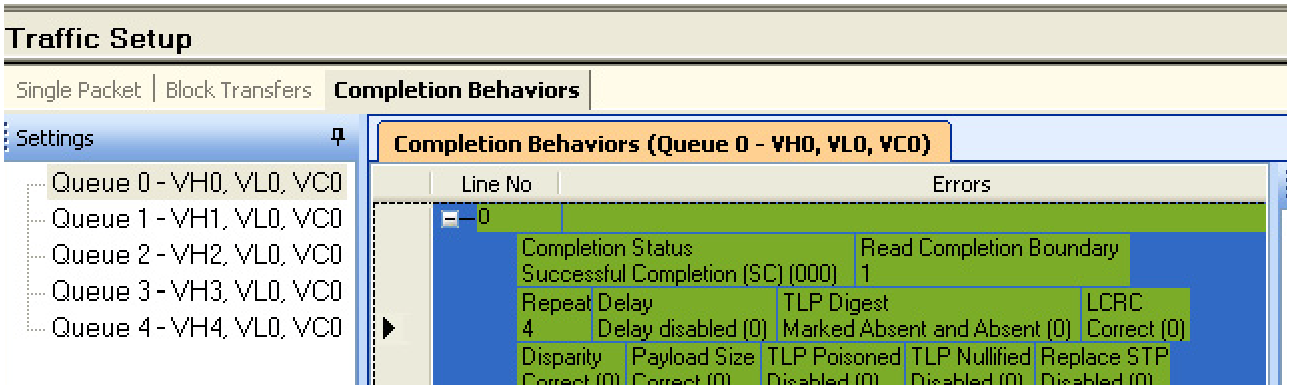
Adding Completion behavior records

When you add a completion behavior, a record with a sequenced number is added to the memory allocated for the selected completion queue.

To add a completion behavior record:

1. Click the **Traffic Setup** icon from the navigation bar of the Protocol Exerciser GUI.
2. Click the **Completion Behaviors** tab.
3. From the **Settings** pane, select the **Completion queue** (Queue 0 or Queue 1) to which you want to add the completion behavior record.
4. Drag and drop the **Default completion** template from the **Templates** pane to the **Completion Behaviors** section.

This adds a completion behavior with the default settings. The Line number assigned to this completion behavior record in the allocated memory is also displayed.



0

Completion Status: Successful Completion (SC) (000)
Read Completion Boundary: 1
Repeat: 4
Delay: Delay disabled (0)
TLP Digest: Marked Absent and Absent (0)
LCRC: Correct (0)
Disparity: Correct (0)
Payload Size: Correct (0)
TLP Poisoned: Disabled (0)
TLP Nullified: Disabled (0)
Replace STP: Disabled (0)

Changing the default behavior for completions

While adding behavior records for the completions, you can change the default settings for these records to change the default behavior of completions. For instance, you can change the default behavior to add a disparity error in the corresponding completion packet to test DUT under error conditions.

To change the default behavior of a completion packet:

1 Click the **Traffic Setup** icon from the navigation bar of the Protocol Exerciser GUI.

2 Click the **Completion Behaviors** tab.

3 From the **Settings** pane, select the **Completion queue** (Queue 0 or Queue 1) to which you added the completion behavior.

4 Double-click the completion behavior record from the list.

The **Edit Packet Behavior** dialog box is displayed. Refer to the Edit Packet Behavior dialog box GUI reference help topic to get a description of each field in this dialog box.

Sending Completions

Protocol Exerciser automatically sends completions as per the completion behavior records that you added. You can add a maximum of 512 completion behavior records in a completion queue.

There is a one to one mapping between the completion packets and the completion behavior records available in a queue. A TLP request received from DUT is sent a completion packet as per the mapped completion behavior record that you added in the sequence. Subsequent TLP requests are sent completion packets as per the subsequent completion behavior records that you added until the end of records is reached. If the number of TLP requests for which completions are to be sent match the added completion behavior records, then there is a one to one mapping between the two. However, if the completion behavior records are less than the number of TLP requests received from DUT, then Exerciser uses a Round robin algorithm to send completion packets. For instance, if five TLP requests are received and you have added two completion behavior records, then the completion packets are sent as follows for these five requests:

TLP request 1	Completion packet as per completion behavior record 1
TLP request 2	Completion packet as per completion behavior record 2
TLP request 3	Completion packet as per completion behavior record 1
TLP request 4	Completion packet as per completion behavior record 2
TLP request 5	Completion packet as per completion behavior record 1

9
Performing Protocol Checks and Inserting Errors

Testing DUT Under Error Conditions 162
Performing Protocol Checks 169
Testing End to End Cyclic Redundancy Checks 174

This chapter provides information on the Protocol Checker function of the Protocol Exerciser application. It also describes how you can use Protocol Exerciser to test DUT under various error conditions.

Testing DUT Under Error Conditions

You can test how a DUT responds to error conditions by sending requests and completions with selected errors as stimulus to the DUT. Using these errors, you can assess the error detection, TLP sequence number and LCRC validation, data integrity, and ACK/NACK mechanisms of a DUT. For instance, you can assess the behavior of the DUT to a wrong payload size packet or a packet with an incorrect LCRC.

Supported Error Insertions

You can use the Edit Packet Behavior dialog box to insert protocol errors in the behavior settings for a request or a completion packet. This dialog box is displayed when you double-click a behavior record for a request or a completion packet added in the Traffic Setup page. If you insert an error that forces a NAK from DUT, for example, wrong LCRC, Exerciser sends the packet once with the error. When the DUT returns a NAK (negative acknowledge), Exerciser automatically sends that packet without the inserted error. Other errors, such as nullified TLP are not checked in the DLL and therefore do not result in an automatic (correct) resend from Exerciser.

Exerciser supports the following errors to be inserted in the behavior settings of a request or a completion packet:

- **TLP Digest** - You can set the TLP Digest bit in the header of a request or a completion packet to indicate its presence. However, the format of the TLP is not changed for TLP Digest thereby introducing an error. You can also set the TLP Digest bit and make it present to introduce an ECRC error.
- **ECRC** - You can insert an incorrect ECRC into a request or a completion packet to test how a DUT responds to an ECRC value. To insert such an error, the TLP Digest bit should be set and present.
- **LCRC** - You can insert a wrong LCRC into a request or a completion packet to test the packet validation mechanism of the DUT. Such a packet is sent with the incorrect LCRC once and then with the correct LCRC again if the DUT responds with NAK.
- **Disparity** - You can send a request or a completion packet with incorrect running disparity.

- **Payload size** - You can send a request or a completion packet with an incorrect payload size. Exerciser can increment or decrement the LEN field of the packet by 1 based on your selection to generate a packet with an incorrect payload size.
- **Poisoned TLP** - You can send a request or a completion packet as a poisoned TLP.
- **Nullified TLP** - You can send a request or a completion packet as a nullified TLP. Exerciser inverts the LCRC and uses the EDB symbol to mark the end of such a TLP.
- **Replace STP** - You can replace the Start TLP special symbol (STP) of a request or a completion packet with a symbol that you specify to generate an error. The encoding K27.7 used for STP symbol is replaced with an encoding that you specify.
- **Replace END** - You can replace the End special symbol (END) of a request or a completion packet with a symbol that you specify to generate an error. The encoding K29.7 used for END symbol is replaced with an encoding that you specify.
- **Offset Sequence Number** - You can add a specified offset to a packet’s sequence number to test the DUT’s sequence number validation algorithm. Exerciser sends such a packet with the incorrect sequence number generated after adding the offset. It then repeats the packet transmission with the correct sequence number.
- **Discard completion** - You can enable the Discard completion setting in a completion behavior record. This setting ensures that Exerciser does not send those completions to DUT for which the completion behavior record with Enabled Discard completion setting is applicable.

Besides adding these errors, you can also insert a delay in sending a completion packet by selecting **Delay Enabled** as the value of the **Delay** field in its behavior record. This can help you test the completion timeout mechanism of the DUT.

Configuring Error Insertion Settings for Exerciser

The errors that you insert in a request or a completion packet's behavior depend on how you configure the relevant error insertion settings in the **Error Insertion** page. For instance, if you inserted a Replace STP error in a request packet's behavior, then the STP symbol for this packet is replaced with the symbol that you specified in the **Replace**

STP with field of the **Error Insertion** -> **Request and Completion Errors** tab. These error insertion settings are globally applicable for all the packets that are sent from Exerciser with relevant errors.

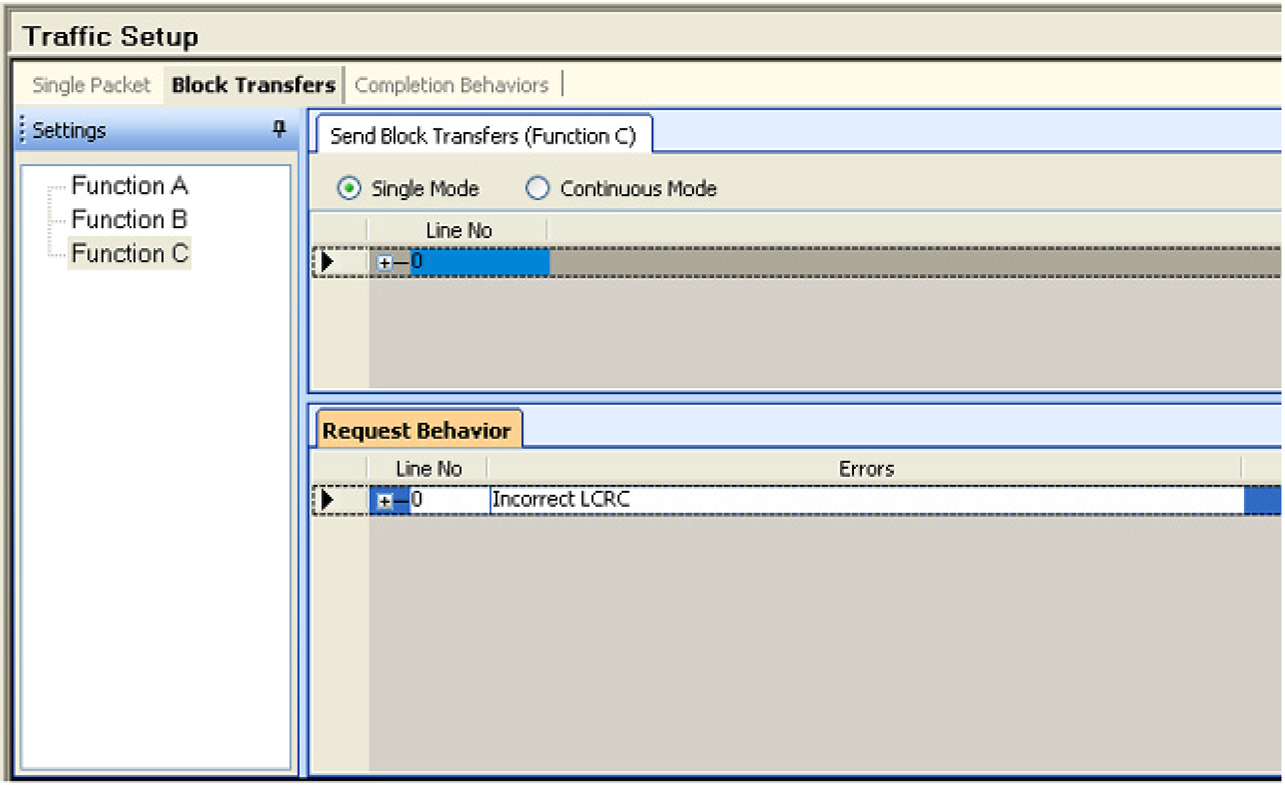
The Error Insertions page provides the error insertion settings categorized in three different tabs as follows:

- **ACK/NAK - Errors tab** - Allows you to test the retry buffer of DUT by enabling you to generate a series of NAKs on a particular sequence number.
- **Request and Completion Errors tab** - Allows you to specify the error settings applicable for the errors that you insert in request and completions packets. In this tab, you can specify settings such as the:
  - symbols that should replace the STP and END symbols of a packet.
  - offset to be added to the sequence number of a packet to generate an incorrect sequence number.
  - time period for which the completions marked for delay should be delayed.
- **DLLP tab** - Allows you to send a DLLP or attach a wrong CRC to the DLLP that Protocol Exerciser sends.

Refer to the topics Error Insertion function - ACK/NAK Errors tab, Error Insertion function - Request and Completion Errors tab, and Error Insertion function - DLLP tab to get a description of each Error Insertion setting.

Testing DUT Under Error Conditions - Example

In the following example, an **incorrect LCRC error** has been inserted in a TLP request to be sent to DUT.



Traffic Setup

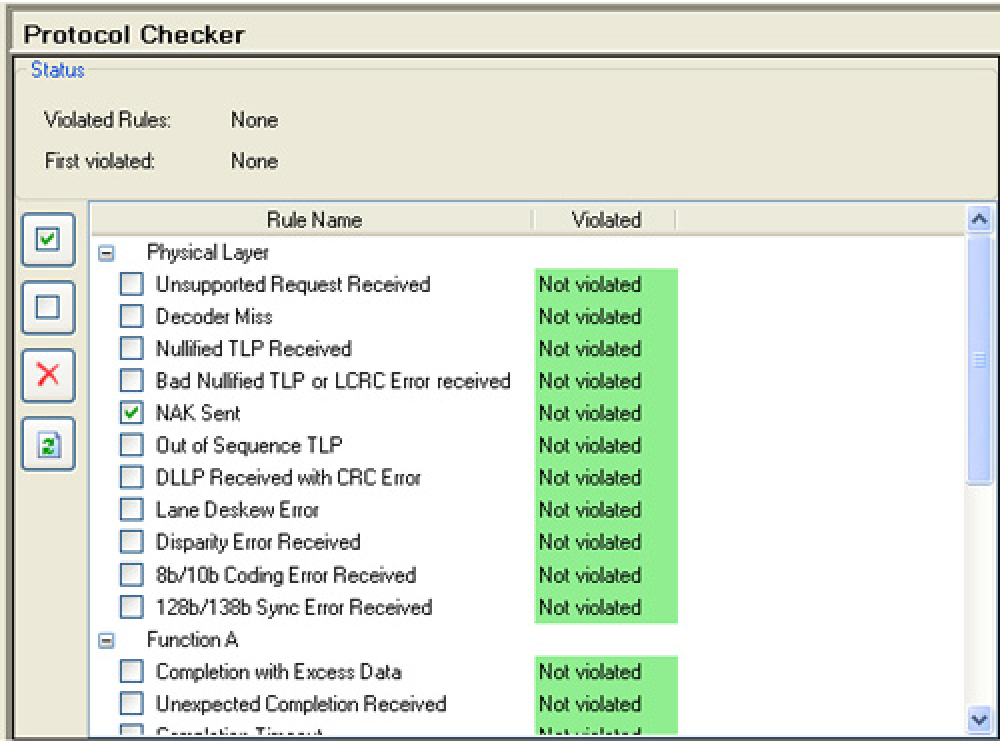
● Single Mode ○ Continuous Mode

Line No
0

Request Behavior

Line No	Errors
0	Incorrect LCRC

To check whether or not DUT responds to such an incorrect LCRC packet with a NAK, the **NAK sent** protocol rule has been enabled in the Protocol Checker page.



Protocol Checker

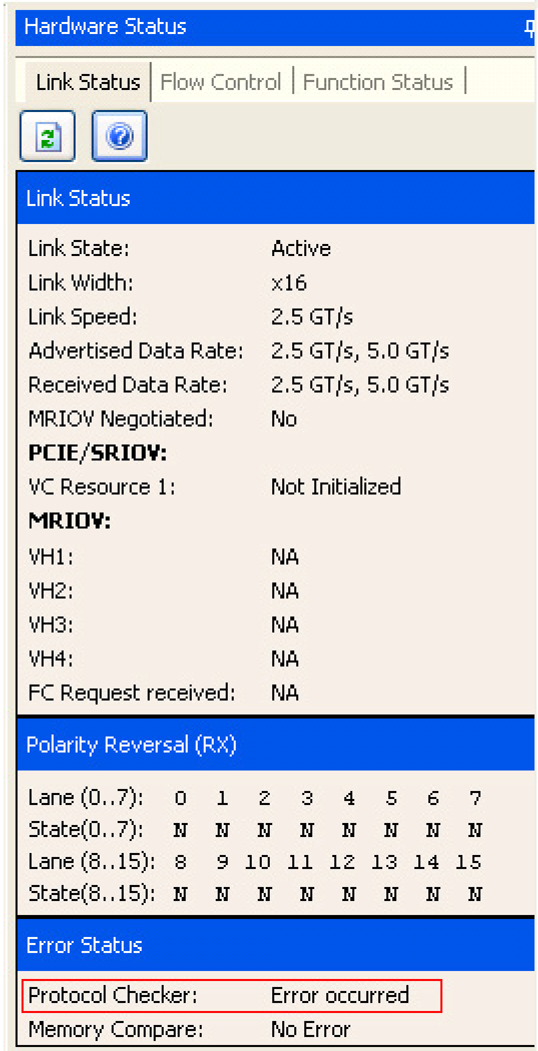
Status

Violated Rules: None

First violated: None

Rule Name	Violated
⊟ Physical Layer	
□ Unsupported Request Received	Not violated
□ Decoder Miss	Not violated
□ Nullified TLP Received	Not violated
□ Bad Nullified TLP or LCRC Error received	Not violated
☑ NAK Sent	Not violated
□ Out of Sequence TLP	Not violated
□ DLLP Received with CRC Error	Not violated
□ Lane Deskew Error	Not violated
□ Disparity Error Received	Not violated
□ 8b/10b Coding Error Received	Not violated
□ 128b/138b Sync Error Received	Not violated
⊟ Function A	
□ Completion with Excess Data	Not violated
□ Unexpected Completion Received	Not violated
□ Completion Timeout	Not violated

On sending the packet with an incorrect LCRC, the NAK received from DUT generates a protocol rule violation which is displayed in the Hardware Status pane.



Link Status

Link State:	Active
Link Width:	x16
Link Speed:	2.5 GT/s
Advertised Data Rate:	2.5 GT/s, 5.0 GT/s
Received Data Rate:	2.5 GT/s, 5.0 GT/s
MRIOV Negotiated:	No
PCIE/SRIOV:	
VC Resource 1:	Not Initialized
MRIOV:	
VH1:	NA
VH2:	NA
VH3:	NA
VH4:	NA
FC Request received:	NA

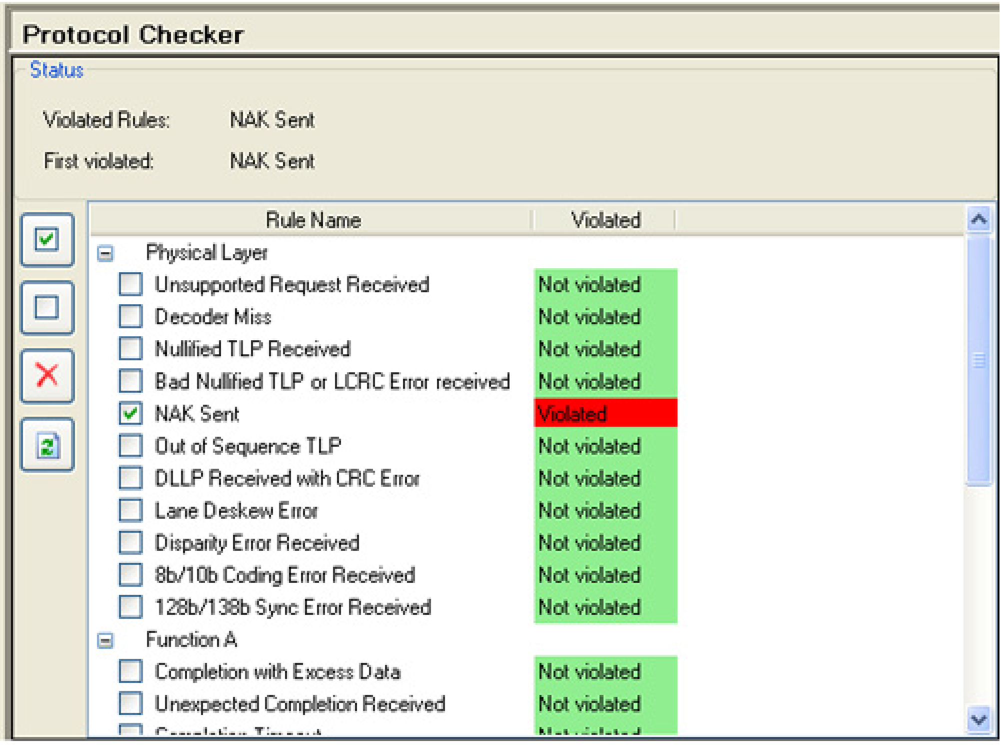
Polarity Reversal (RX)

Lane (0..7):	0	1	2	3	4	5	6	7
State(0..7):	N	N	N	N	N	N	N	N
Lane (8..15):	8	9	10	11	12	13	14	15
State(8..15):	N	N	N	N	N	N	N	N

Error Status

Protocol Checker:	Error occurred
Memory Compare:	No Error

The Status pane of the Protocol Checker page displays **NAK sent** as the violated rule indicating that DUT responded to the erroneous packet with a NAK.



Protocol Checker

Status

Violated Rules: NAK Sent

First violated: NAK Sent

Rule Name	Violated
⊟ Physical Layer	
☐ Unsupported Request Received	Not violated
☐ Decoder Miss	Not violated
☐ Nullified TLP Received	Not violated
☐ Bad Nullified TLP or LCRC Error received	Not violated
☑ NAK Sent	Violated
☐ Out of Sequence TLP	Not violated
☐ DLLP Received with CRC Error	Not violated
☐ Lane Deskew Error	Not violated
☐ Disparity Error Received	Not violated
☐ 8b/10b Coding Error Received	Not violated
☐ 128b/138b Sync Error Received	Not violated
⊟ Function A	
☐ Completion with Excess Data	Not violated
☐ Unexpected Completion Received	Not violated
☐ Completion Timeout	Not violated

Performing Protocol Checks

Exerciser provides a set of protocol checks that you can use to check whether or not there are any protocol rules violations in the transmission between Exerciser and DUT. This helps you assess if the two devices are adhering to the rules of the PCIe protocol. If you want to test DUT by introducing a PCIe rule violation in the transmission, then you can use the protocol checks to see if that rule is actually violated.

The protocol checks are available in the **Protocol Checker** page of the GUI which you can access by clicking the **Protocol Checker** icon in the Navigation pane. In this page, you can enable the required protocol checks. On encountering a violation of an enabled protocol rule in a packet, Exerciser records this violation as a violated rule in the Protocol Checker page.

NOTE

In the Link Status tab of the Hardware Status pane, the Protocol Checker section shows the Error occurred message to indicate that a violation of an enabled protocol rule has occurred.

Generating a Trigger out on encountering a protocol error

Exerciser can also generate an event and a Trigger Out pulse on encountering a violation of an enabled protocol rule. Such an event is displayed in the **General Settings -> Trigger Out** tab. The trigger out pulse is sent from Exerciser to another test equipment connected to the Trigger out Connector component of the Exerciser card. To ensure that an event and a trigger out pulse is generated for a violation of an enabled protocol rule, you need to select **Protocol error** as the trigger out condition in the **General Settings -> Trigger Out** tab. Refer to the topic Configuring External Triggers Settings on page 69 to know more.

Enabling a protocol rule

To enable a protocol rule:

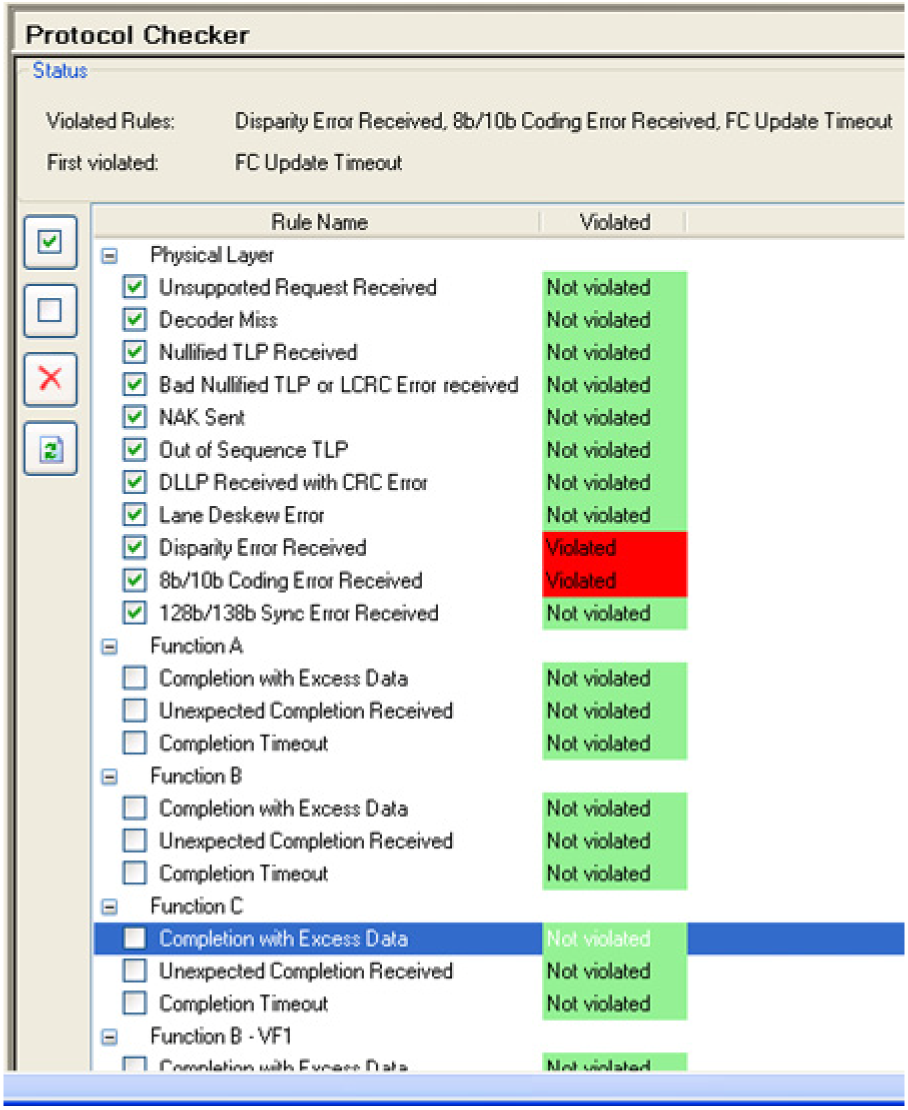
1 Click the **Protocol Checker** icon in the Navigation pane.

2 Select the checkbox displayed with a protocol rule to enable that rule.

Irrespective of whether or not you have enabled a rule, its status will be shown as Violated in case of a violation. However, if you do not enable a protocol rule, then its violation cannot:

- generate a trigger or an event
- be displayed as the first violated rule
- generate a message in the Protocol Checker section of the Link Status tab of the Hardware status pane.

The following screen displays the Protocol Checker pane with three protocol rules displayed as Violated for an MRIOV capable Exerciser.



Protocol Checker

Status

Violated Rules: Disparity Error Received, 8b/10b Coding Error Received, FC Update Timeout

First violated: FC Update Timeout

Rule Name	Violated
⊟ Physical Layer	
☑ Unsupported Request Received	Not violated
☑ Decoder Miss	Not violated
☑ Nullified TLP Received	Not violated
☑ Bad Nullified TLP or LCRC Error received	Not violated
☑ NAK Sent	Not violated
☑ Out of Sequence TLP	Not violated
☑ DLLP Received with CRC Error	Not violated
☑ Lane Deskew Error	Not violated
☑ Disparity Error Received	Violated
☑ 8b/10b Coding Error Received	Violated
☑ 128b/138b Sync Error Received	Not violated
⊟ Function A	
□ Completion with Excess Data	Not violated
□ Unexpected Completion Received	Not violated
□ Completion Timeout	Not violated
⊟ Function B	
□ Completion with Excess Data	Not violated
□ Unexpected Completion Received	Not violated
□ Completion Timeout	Not violated
⊟ Function C	
□ Completion with Excess Data	Not violated
□ Unexpected Completion Received	Not violated
□ Completion Timeout	Not violated
⊟ Function B - VF1	
□ Completion with Excess Data	Not violated

Supported Protocol Checks

Protocol checks are supported for each function of Exerciser. The checks are displayed for:

- each of the five functions (A, B, C, D, and E) and virtual channels supported by Exerciser as a non IOV PCIe component.
- each of the physical as well as virtual functions (VF1 and VF2) and virtual channels supported by Exerciser as a SRIOV capable component.
- each of the physical as well as virtual functions (VF1 and VF2) and virtual hierarchies supported by Exerciser as a MRIOV capable component.

The following table briefly describes each of the protocol checks supported in this release.

Protocol Rule	Description
Physical Layer	
Unsupported Request Received	This protocol rule is violated if Exerciser receives an incoming request which is not supported.
Decoder Miss	This protocol rule is violated if Exerciser receives a request packet that cannot be handled by any of its decoders.
Nullified TLP Received	This protocol rule is violated if Exerciser receives a packet with nullified LCRC.
Bad Nullified TLP or LCRC Error Received	This protocol rule is violated if Exerciser receives a packet with bad end EDB or bad (not poisoned) LCRC.
NAK Sent	This protocol rule is violated if Exerciser receives a NAK for a packet. For instance, if Exerciser sends a packet with an incorrect LCRC error inserted, then you can check if DUT responds with a NAK for such a packet by enabling the NAK sent protocol rule.
Out of Sequence TLP	This protocol rule is violated if Exerciser receives an out-of-sequence TLP. as per the following two scenarios: • Exerciser receives an ACK or NAK with sequence number A and the TLP with sequence number A has not yet been sent or has been sent far in the past. • Exerciser receives an ACK or NAK with a sequence number A. After that, it receives an ACK or NAK with a predecessor of A, say X. The sequence number X is a predecessor of A if [(X-A) MOD 2^12 > 2^11].
DLLP Received with CRC Error	This protocol rule is violated if Exerciser receives a DLLP with checksum error.
Lane Deskew Error	Indicates the receipt of a malformed TLP.
Disparity Error Received	Indicates the receipt of a running disparity error at Gen1/Gen2 data rate.

Protocol Rule	Description
8b/10b Coding Error Received	Indicates the receipt of a coding (symbol) error at Gen1/Gen2 data rate.
128b/138b Sync Error Received	Indicates the receipt of a 128b/138b encoding error at Gen3 data rate.
ECRC Error Detected	This protocol rule's violation is reported globally at the physical layer level if Exerciser emulates a non IOV PCIe component or a SRIOV capable component (protocol set to **PCIe** or **SRIOV** in the **Link Settings** page). This protocol rule is displayed as violated if the ECRC check fails for a received TLP in which the TLP Digest field is used for an ECRC value. When Exerciser receives a TLP from a DUT that has the ECRC generation supported and enabled, it compares the calculated ECRC value with the received value in the TLP Digest field of the TLP. On comparison, if the ECRC check fails, it is reported as a violation. By default, Exerciser has the ECRC checking supported and enabled for the received TLPs. Refer to the topic Testing End to End Cyclic Redundancy Checks to know more about injecting and checking ECRC errors in TLPs.
For each physical and virtual function of Exerciser	
Completion with Excess Data	This protocol rule is violated for a function if Exerciser receives the completion for a request containing more data than expected.
Unexpected Completion Received	This protocol rule is violated for a function if Exerciser receives an unexpected completion packet that is, a completion packet that it cannot associate with any outstanding request.
Completion Timeout	This protocol rule is violated for a function if Exerciser does not receive a completion outstanding for a request within the allowed completion timeout period.
For each virtual channel of Exerciser (applicable when protocol is set to PCIe or SRIOV)	
FC Update Timeout	If Exerciser does not receive a Flow Control update packet during the specified time interval. the Flow Control Update Timeout is triggered and the FC Update Timeout rule is displayed as Violated. You can specify the time interval after which Exerciser should trigger a Flow Control Update timeout in the FC Update Timeout section of the Virtual Channels page.
For each virtual hierarchy supported by Exerciser (applicable when protocol is set to MRIOV)	

Protocol Rule	Description
FC Update Timeout	If Exerciser does not receive a Flow Control update packet during the specified time interval. the Flow Control Update Timeout is triggered and the FC Update Timeout rule is displayed as Violated. You can specify the time interval after which Exerciser should trigger a Flow Control Update timeout in the FC Update Timeout section of the Virtual Channels page.
ECRC Error Detected	This protocol rule's violation is reported separately for each virtual hierarchy if Exerciser emulates a MRIOV PCIe component (protocol set to MRIOV in the Link Settings page). This protocol rule is displayed as violated if the ECRC check fails for a received TLP in which the TLP Digest field is used for an ECRC value. When Exerciser receives a TLP from a DUT that has the ECRC generation supported and enabled, it compares the calculated ECRC value with the received value in the TLP Digest field of the TLP. On comparison, if the ECRC check fails, it is reported as a violation. By default, Exerciser has the ECRC checking supported and enabled for the received TLPs. Refer to the topic Testing End to End Cyclic Redundancy Checks to know more about injecting and checking ECRC errors in TLPs.

Testing End to End Cyclic Redundancy Checks

You can use Exerciser to:

- detect End to End Cyclic Redundancy Check (ECRC) errors in the TLPs received from DUT.
- send TLPs with correct/incorrect ECRC to DUT to test how DUT responds to a correct or an incorrect ECRC in the TLPs.

NOTE

To detect or insert ECRC errors in TLPs, you need the U4305A-022 software license.

As a transmitter, Exerciser has the ECRC generation supported and enabled.

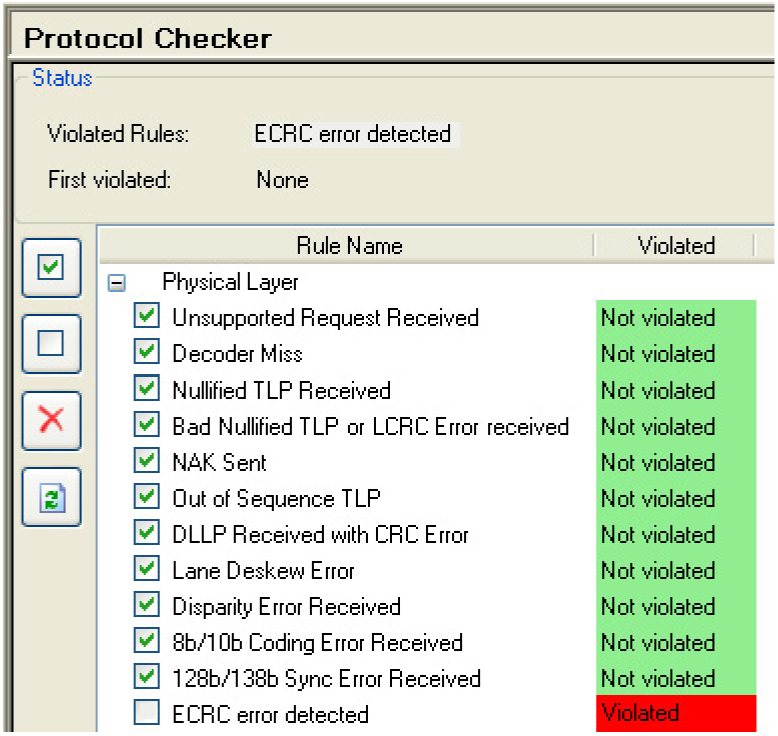
As a receiver, Exerciser has the ECRC checking supported and enabled.

The topic describes both these aspects of ECRC.

Checking ECRC errors in Received TLPs

Exerciser checks the ECRC value in incoming TLPs to help you assess if the DUT is calculating and applying the ECRC value correctly to the TLPs. The DUT should have the ECRC generation supported and enabled.

When Exerciser receives a TLP with an ECRC value at the end of the TLP, it checks this ECRC value by comparing the calculated ECRC value with the value in the TLP Digest field. If the ECRC check fails, Exerciser reports it as the violation of the **ECRC error detected** protocol rule in the **Protocol Checker** page. Exerciser does not discard a TLP for which the ECRC check failed and also sends a completion (if applicable) for such a TLP request.



Protocol Checker

Status

Violated Rules: ECRC error detected

First violated: None

Rule Name	Violated
⊟ Physical Layer	
☑ Unsupported Request Received	Not violated
☑ Decoder Miss	Not violated
☑ Nullified TLP Received	Not violated
☑ Bad Nullified TLP or LCRC Error received	Not violated
☑ NAK Sent	Not violated
☑ Out of Sequence TLP	Not violated
☑ DLLP Received with CRC Error	Not violated
☑ Lane Deskew Error	Not violated
☑ Disparity Error Received	Not violated
☑ 8b/10b Coding Error Received	Not violated
☑ 128b/138b Sync Error Received	Not violated
☐ ECRC error detected	Violated

The violation of the ECRC error detected protocol rule is reported separately for each virtual hierarchy if Exerciser emulates a MRIOV PCIe component and globally at the physical layer level if Exerciser emulates a non IOV/SRIOV capable component.

You can also enable this ECRC error detected protocol rule in the Protocol Checker page to ensure that a trigger out pulse is generated when Exerciser encounters an ECRC error in a received TLP. To know more, refer to Configuring External Triggers Settings.

Inserting ECRC Errors in Transmitted TLPs

You can test how a DUT responds to an ECRC value by transmitting TLPs with a correct or incorrect ECRC value from Exerciser.

To insert an ECRC error

1 Add an instance of the desired TLP request in the **Traffic Setup** page.

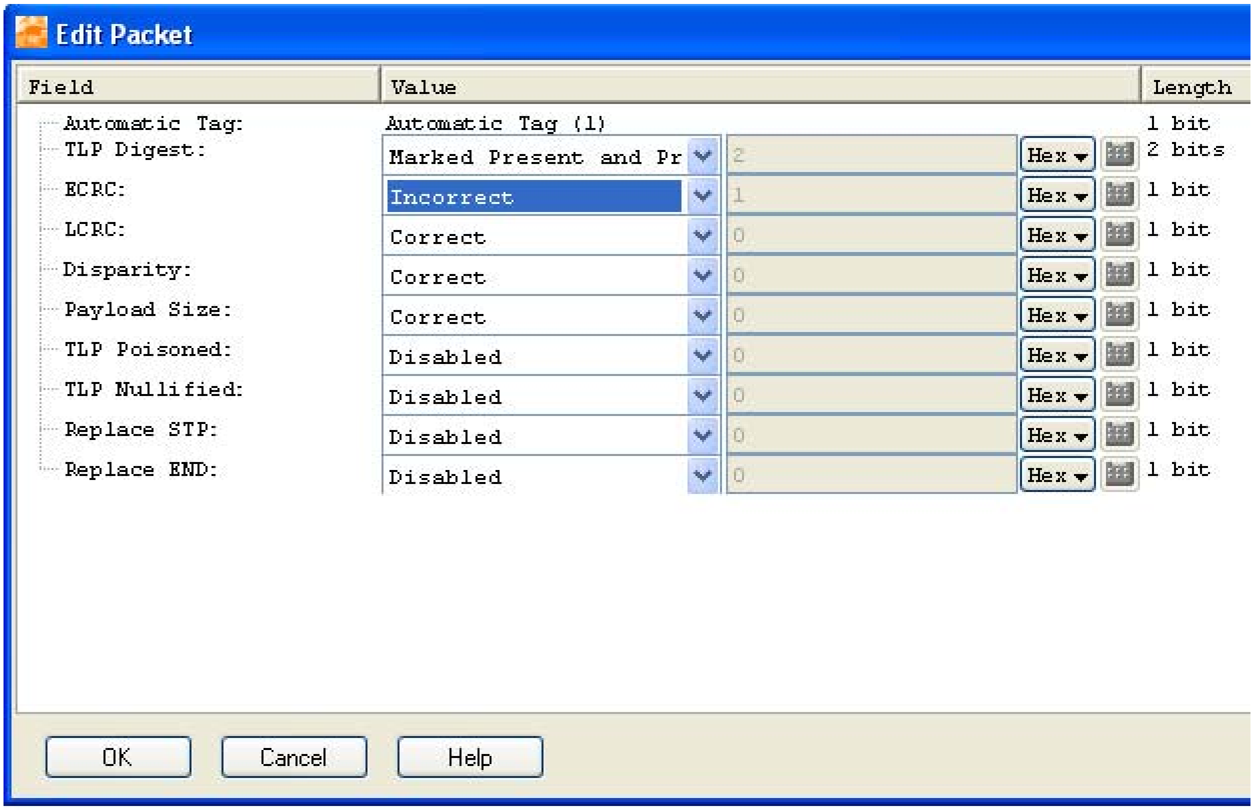
2 Double-click the TLP request behavior record to configure the behavior of the request.

3 In the **Edit Packet** dialog box, select **Marked Present and Present** option for the **TLP Digest** field. This sets the TLP Digest bit in the header of the packet indicating the presence of the TLP Digest field and accordingly changes the format of the packet for the TLP Digest.

The **ECRC** field is now displayed in the **Edit Packet** dialog box.

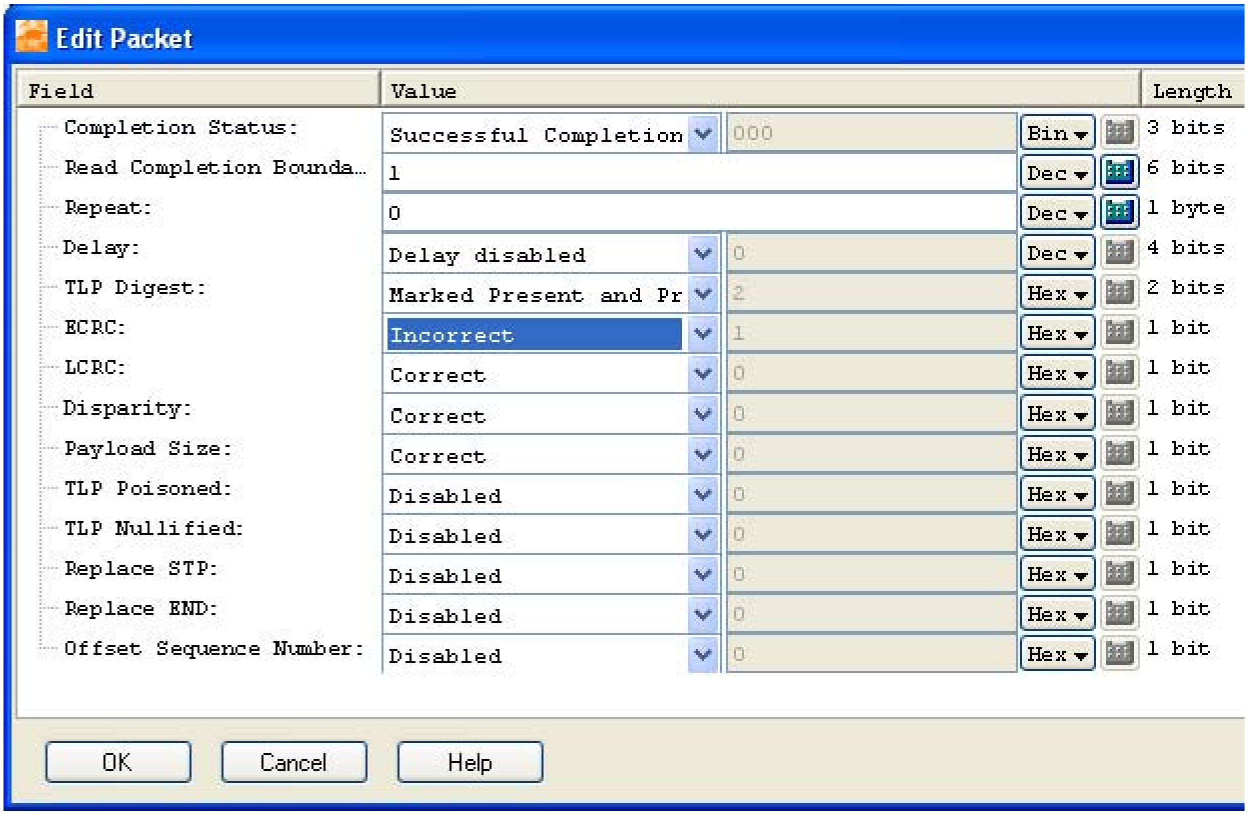
4 Select **Incorrect** for the **ECRC** field to send the TLP request with an incorrect ECRC value.

5 Click **OK**.



Field	Value			Length
Automatic Tag:	Automatic Tag (1)			1 bit
TLP Digest:	Marked Present and Pr	2	Hex	2 bits
ECRC:	Incorrect	1	Hex	1 bit
LCRC:	Correct	0	Hex	1 bit
Disparity:	Correct	0	Hex	1 bit
Payload Size:	Correct	0	Hex	1 bit
TLP Poisoned:	Disabled	0	Hex	1 bit
TLP Nullified:	Disabled	0	Hex	1 bit
Replace STP:	Disabled	0	Hex	1 bit
Replace END:	Disabled	0	Hex	1 bit

You can also insert an ECRC error in a completion behavior record to send the completion with an incorrect ECRC.



Field	Value			Length
Completion Status:	Successful Completion	000	Bin	3 bits
Read Completion Bounda…	1		Dec	6 bits
Repeat:	0		Dec	1 byte
Delay:	Delay disabled	0	Dec	4 bits
TLP Digest:	Marked Present and Pr	2	Hex	2 bits
ECRC:	Incorrect	1	Hex	1 bit
LCRC:	Correct	0	Hex	1 bit
Disparity:	Correct	0	Hex	1 bit
Payload Size:	Correct	0	Hex	1 bit
TLP Poisoned:	Disabled	0	Hex	1 bit
TLP Nullified:	Disabled	0	Hex	1 bit
Replace STP:	Disabled	0	Hex	1 bit
Replace END:	Disabled	0	Hex	1 bit
Offset Sequence Number:	Disabled	0	Hex	1 bit

You can also send a TLP with no ECRC by setting the value of the **TLP Digest** field in the packet's behavior record as one of the following:

- Marked Absent and Absent
- Marked Present but Missing

10
Protocol Test Card

Protocol Test Card 180
Important Points about Protocol Test Card 186
Protocol Test Card Example 187

This chapter provides information on the Protocol Test Card (PTC) that allows you to verify the PCIe compliance of an add-in card.

Protocol Test Card

There may be a situation when you want to verify if your add-in card is *in compliance* with the PCIe protocol. For such a situation, Agilent provides *Protocol Test Card* (PTC). PTC is a tests suite that provides various tests to verify the PCIe compliance of your add-in card. These tests are based on *assertions* that are set of rules written down by PCI-SIG. An add-in card must adhere to these assertions to be in compliance with PCIe.

The following list provides more information about PTC:

- The PTC tests suite comes packaged within the N5309A exerciser card, and the PTC GUI is available inside the Protocol Exerciser GUI.
- The license scheme for both PTC and Protocol Exerciser is different. You need two separate licenses in order to use Protocol Exerciser as well as PTC.

  If you have a license only for PTC, then you will be able to use PTC, but not any other Protocol Exerciser function.
- There are separate modules for Protocol Exerciser(*PCIe Gen3 Exerciser*) and PTC(*PCIe Gen3 PTC*). To use PTC, you have to select *PCIe Gen3 PTC* module type from a dropdown menu next to module number while selecting port. While you are running PTC, only *General Settings* page and *Compliance Test* page will be visible.

NOTE

For a quick look at all the PTC test cases and their brief description, refer to Appendix 2, “PTC Tests and Assertions.

While using PTC, you can mount the add-in card on the *PCIe Connector* of the N5316A passive backplane board or on the *Add-in Card Connector* of the N5309-66417 Extension Card (Figure 29).

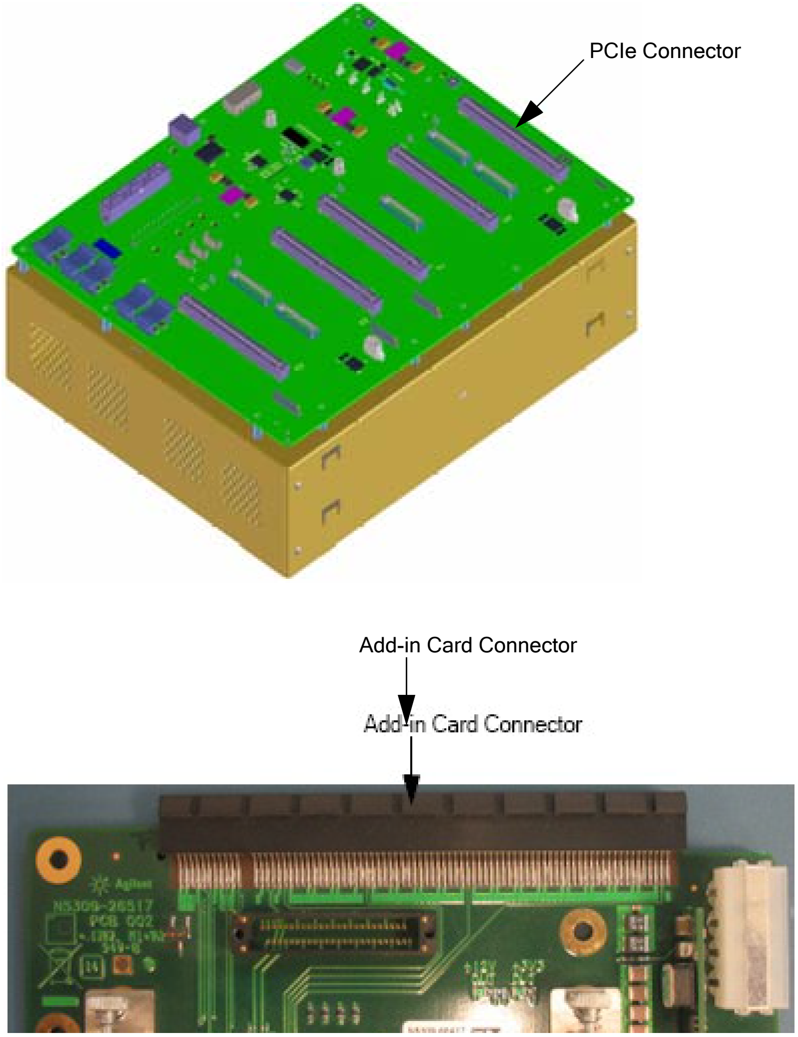


Figure 29 N5316A Passive Backplane and N5309-66417 Extension Card

NOTE

For information on N5309-66417, refer to *Agilent System Protocol Tester, Hardware and Probing Guide*.

For information on N5309A and N5316A, refer to *Agilent System Protocol Tester, Hardware and Probing Guide* and *Agilent System Protocol Tester, Installation Guide*.

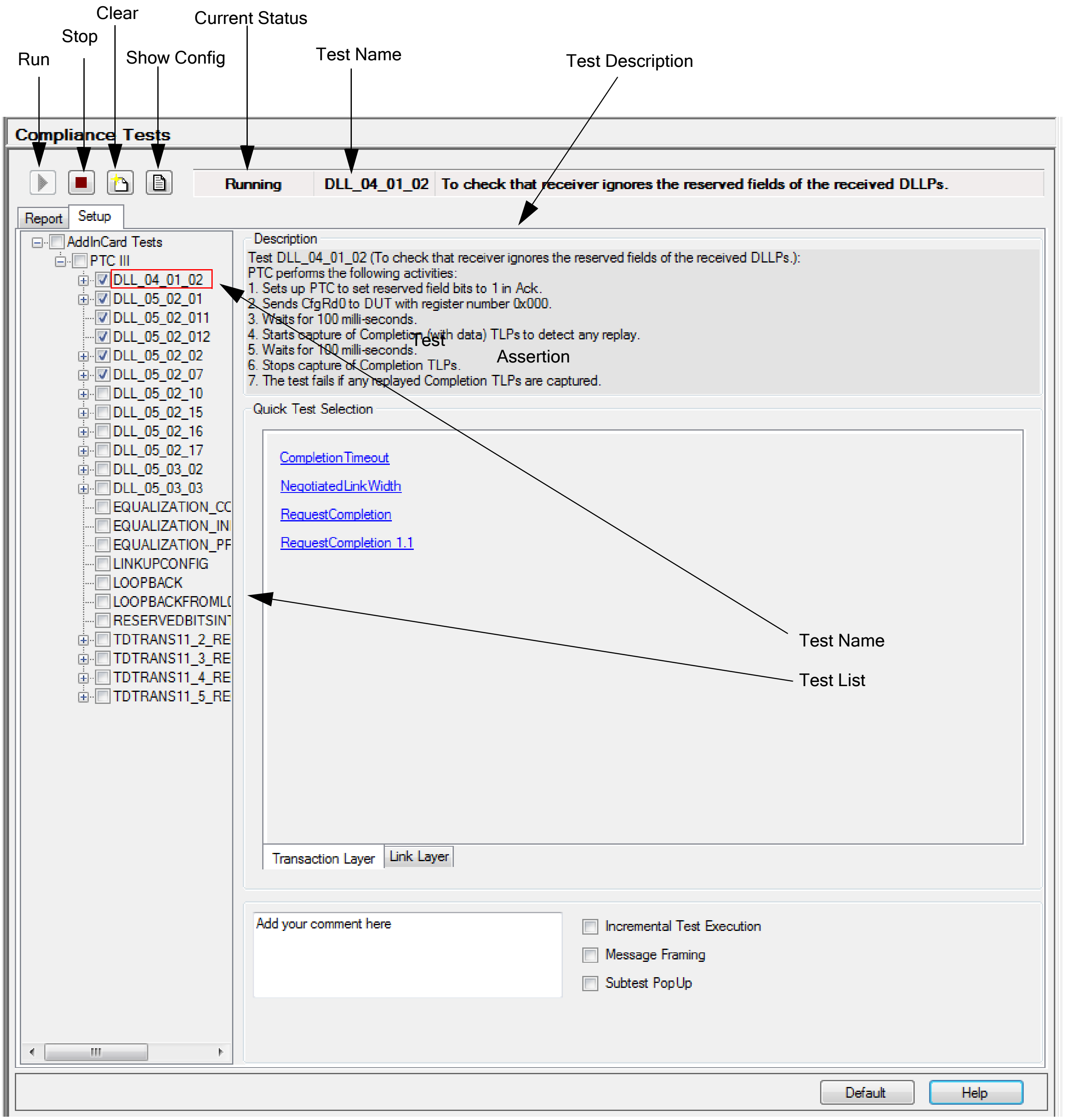
Accessing Protocol Test Card

To access PTC:

- Start the Protocol Exerciser GUI.
- Select *PCIe Gen3 PTC* module type while adding port.

The Compliance Tests screen appears (Figure 30).

For information on starting the Protocol Exerciser GUI, refer to Accessing Protocol Exerciser on page 19.



Compliance Tests

Running | DLL_04_01_02 | To check that receiver ignores the reserved fields of the received DLLPs.

□ AddInCard Tests
- □ PTC III
  - ☑ DLL_04_01_02
  - ☑ DLL_05_02_01
  - ☑ DLL_05_02_011
  - ☑ DLL_05_02_012
  - ☑ DLL_05_02_02
  - ☑ DLL_05_02_07
  - □ DLL_05_02_10
  - □ DLL_05_02_15
  - □ DLL_05_02_16
  - □ DLL_05_02_17
  - □ DLL_05_03_02
  - □ DLL_05_03_03
  - □ EQUALIZATION_CC
  - □ EQUALIZATION_IN
  - □ EQUALIZATION_PF
  - □ LINKUPCONFIG
  - □ LOOPBACK
  - □ LOOPBACKFROML(
  - □ RESERVEDBITSINT
  - □ TDTRANS11_2_RE
  - □ TDTRANS11_3_RE
  - □ TDTRANS11_4_RE
  - □ TDTRANS11_5_RE

Description

Test DLL_04_01_02 (To check that receiver ignores the reserved fields of the received DLLPs.):
PTC performs the following activities:
1. Sets up PTC to set reserved field bits to 1 in Ack.
2. Sends CfgRd0 to DUT with register number 0x000.
3. Waits for 100 milli-seconds.
4. Starts capture of Completion (with data) TLPs to detect any replay.
5. Waits for 100 milli-seconds.
6. Stops capture of Completion TLPs.
7. The test fails if any replayed Completion TLPs are captured.

Quick Test Selection

CompletionTimeout
NegotiatedLinkWidth
RequestCompletion
RequestCompletion 1.1

Add your comment here

□ Incremental Test Execution
□ Message Framing
□ Subtest PopUp

Figure 30 Compliance Tests screen

Table 1 briefly describes the components of the Compliance Tests screen.

Table 1 Components of the Compliance Tests screen

Component	Description
Run	Click **Run** to execute the selected tests. You can also execute a test by right-clicking it, and then clicking **Run Test** from the shortcut menu.
Stop	Click **Stop** to stop the test execution. Stopping is possible only between tests, i.e. only when you are executing multiple tests. You cannot stop a single test.
Clear	Click **Clear** to clear all existing test results from the compliance report. Remember the following points on using the Clear command button: • Running new tests without clicking Clear appends new tests in the existing compliance report. • Running an already executed test without clicking Clear overrides its previous execution result with the new execution result.
Show Config	Click **Show Config** to display the *Configuration Space of DUT* message box, which shows the current configuration space of DUT. Remember to execute a test before using this command button, otherwise the message box appears *empty*.
Current Status	The **Current Status** label shows the following status messages: • **Stopped:** Appears when you access the PTC GUI. This status message stays as long as you do not execute any test. • **Running**: Appears when tests are running. • **Finished**: Appears after the compliance report is prepared and is displayed in the Report tab. • **Cancelling**: Appears when you click the Stop button, while a test is under execution. This message stays as long as the current test is executing. • **Cancelled**: Appears when the tests execution halts after you have clicked the Stop button.
Test Name	The **Test Name** label shows the name of the currently executing test.
Test Description	The **Test Description** label shows the brief description of the currently executing test.

Table 1 Components of the Compliance Tests screen

Component	Description
Setup	The **Setup** tab allows you to specify the test cases to be executed.
Report	The **Report** tab displays the generated compliance report, and also allows you to save and print it for future references.
Default	Click **Default** to revert back to the default settings of the Compliance Tests screen.
Help	Click **Help** to display online help.

Figure 30 shows the Setup tab, and Table 2 briefly describes the components of the Setup tab.

Table 2 Components of the Setup tab

Component	Description
AddInCard Tests	The **AddInCard Tests** list box provides a check box list of PTC tests. Here, select the check boxes of the tests that you want to run. Selecting the check box of a test category selects all tests under it. That is, selecting *PTC III* selects all tests under it. You can also select or clear the check box of a test by right-clicking it , and then clicking **Select** or **Unselect** from the shortcut menu.
Description	The **Description** section displays the description of the test case or assertion you selected in the Tests list box. In this section, the description is displayed only when you select the title of the test case.
Comment	The **Comment** text area allows to add user comments to the tests execution. These user comments are copied to the compliance report.
Incremental Test Execution	Select **Incremental Test Execution** to automatically clear the check boxes of the selected tests after their execution.
Message Framing	Select **Message Framing** to send a vendor defined Type-1 message before and after each test execution.
Subtest PopUp	Select **Subtest PopUp** to display a pop up after each sub-test execution in case of equalization tests. *This is purely for debugging purposes and is not recommended for normal operation.*

Table 2 Components of the Setup tab

Component	Description
Testing Mode	Select **Testing Mode** to display a pop up for initial configuration of each sub-test execution. *This is purely for debugging purposes and is not recommended for normal operation.*

Table 3 briefly describes the components of the Report tab.

Table 3 Components of the Report tab

Component	Description
Report	The **Report** area displays the report for the tests that you executed to verify if the underlying DUT is in compliance with PCIe. This compliance report has the following sections: • **Overview**: This section provides hyperlinks to directly jump to the *Summary* and *Details* sections of the report. • **Summary**: This section provides details about the test cases that were executed, their brief description, and their result of execution. This section also provides details about: • Device under test. • Manufacturer. • Date and time of report generation. • Link speed and link width. • Number of executed test cases, and how many of them passed or failed. • Number of warnings generated by the executed test cases. Warnings are generated when you run test cases without specifying any *safe range*. • **Details**: This section provides the description of the executed test cases and their associated assertions, result of execution, and a log describing the each step that was executed by the test case. • **Configuration Space**: This section provides the description of the DUT's configuration space after each execution. • **Versions**: This section lists the software and FPGA versions used for test execution.
Save	Click **Save** when you want save the compliance report as an HTML document. Clicking Save displays the Save HTML Document dialog box, where you can specify the name and location of the report to be saved.
Print	Click **Print** to print the compliance report.

Important Points about Protocol Test Card

The following are some important points about PTC:

- Test names are derived from their assertion names if provided in specs, otherwise they have brief names describing their intent. Consider the following examples of test names and assertion names:

  Test name: DLL_04_01_02

  Assertion name: DLL.04.01#02

  Test name: EQUALIZATION_COEFFICIENTS

  Assertion name: No Assertion
- PTC allows you to execute tests only. You cannot execute assertions.
- In the Tests list box, the tests and assertions are organized in a tree-view format, where leaves are assertions and their parent nodes are tests.
- Each test, before its execution, resets the underlying DUT by first asserting and then de-asserting the PCIRST# pin.

Protocol Test Card Example

Consider that you want to run a couple of tests for your DUT to check:

- For reserved fields in DLLPs, and
- For retransmission of transaction in case of NAK being issued.

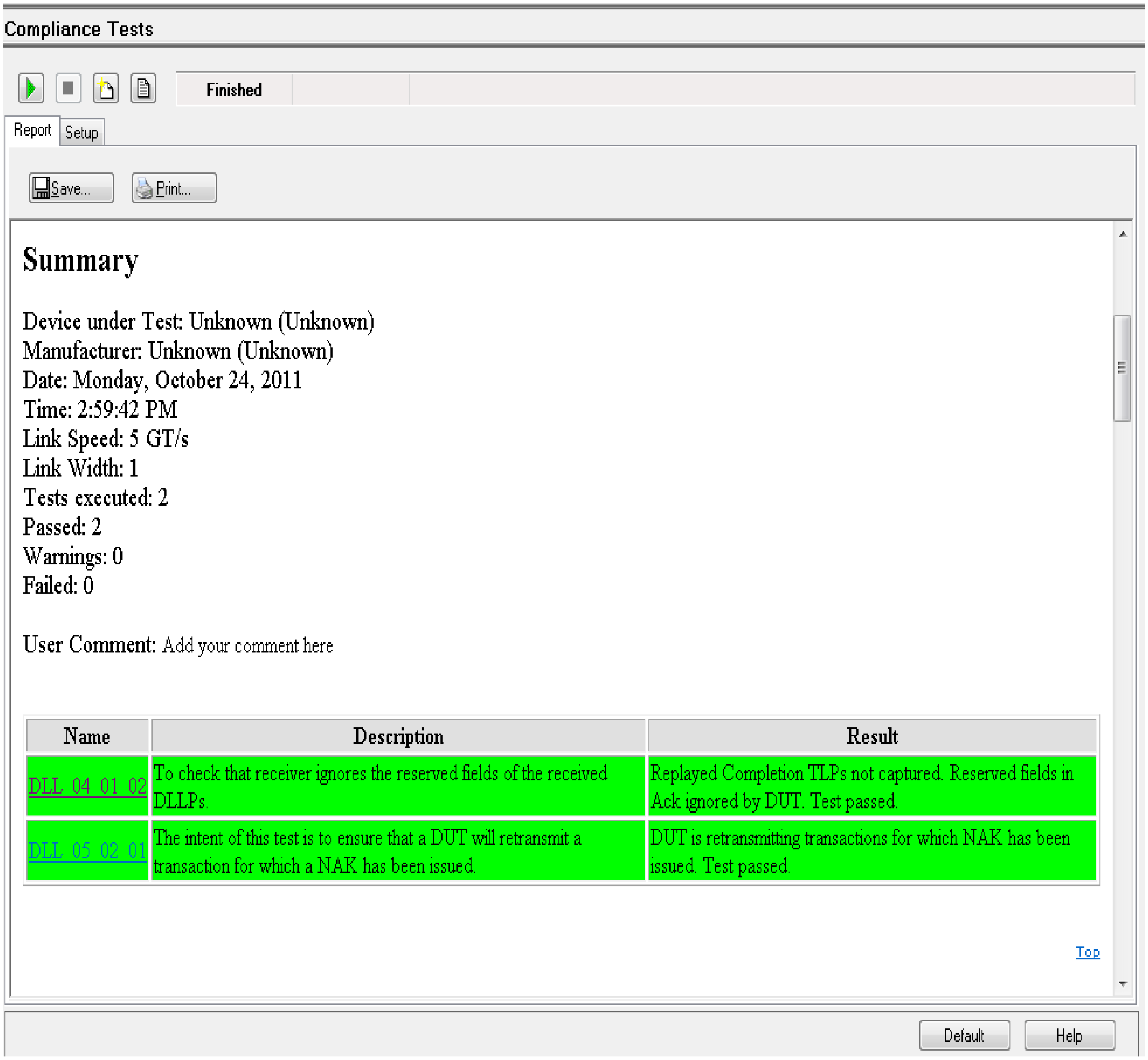
To Check that the receiver ignores the reserved fields of the received DLLPs, and To check that DUT retransmits a transaction for which NAK has been issued

1 Select the check box of the **DLL_04_01_02** test from the Tests list box. This test checks the reserved fields of received DLLPs.

2 Select the check box of the **DLL_05_02_01** test from the Tests list box. This will check for retransmission of transaction.

3 Click **Run**.

This executes the selected test(s), and displays the following compliance report.



Summary

Device under Test: Unknown (Unknown)
Manufacturer: Unknown (Unknown)
Date: Monday, October 24, 2011
Time: 2:59:42 PM
Link Speed: 5 GT/s
Link Width: 1
Tests executed: 2
Passed: 2
Warnings: 0
Failed: 0

User Comment: Add your comment here

Name	Description	Result
DLL_04_01_02	To check that receiver ignores the reserved fields of the received DLLPs.	Replayed Completion TLPs not captured. Reserved fields in Ack ignored by DUT. Test passed.
DLL_05_02_01	The intent of this test is to ensure that a DUT will retransmit a transaction for which a NAK has been issued.	DUT is retransmitting transactions for which NAK has been issued. Test passed.

Figure 31 Compliance Report

You can use this same procedure to execute other PCIe compliance tests.

10
Protocol Exerciser GUI Reference

Dialog Boxes 190
Menus 206
Windows 209

This chapter provides description of each field, menu option, and toolbar option available in various windows and dialog boxes of the Protocol Exerciser GUI.

Dialog Boxes

Edit Packet Behavior dialog box

To access this dialog: Double-click a completion behavior record in the **Completion Behavior** tab or double-click a request behavior record in the **Request Behavior** tab of the **Traffic Setup** page.

This dialog box is used for editing a completion behavior or a request behavior record that you added in the Completion Behavior tab and Request Behavior tab respectively. You can change the default behavior of a completion or a request using this dialog box.

The dialog box displays the applicable fields and the respective value and length of these fields of the behavior record.

The Value column also contains a drop-down list and a calculator for input assistance. The drop-down list enables you to specify the value of fields in hexadecimal, binary, or decimal format. Clicking the calculator for input assistance displays the Input Assistant dialog box. Here, you can perform boolean operations over the hexadecimal, binary, and decimal values, and use the resulting value as the new value for the corresponding packet field. The **Length** column displays the maximum length of a field in bits, bytes, or DWords.

In this dialog box, there are some fields that you use to insert errors in the underlying packet such as Payload size field. You need to set such fields in conjunction to the settings done in the Error Insertion page so that the error insertion work effectively.

Component	Description
Chunk Length of Payload	Length of payload data of the packet in DWORDs. The maximum value that you can specify is 1024. This field is applicable for a request behavior record only.
Completion Status	Select the status that will be used to specify the completion status in the response. This is applicable only for a completion behavior record only.

Component	Description
Read Completion Boundary	Specify how many RCBs will be crossed by the completion packet that is, specify the length of the completion packet. Valid range is 0 to 63. This field is applicable for a completion behavior record only.
Repeat	Repeat field means the number of times: • the completion behavior record will be used repeatedly to send completion packets for TLP requests. • the request behavior record will be used repeatedly to send TLP requests. For instance, if the Repeat property is set to 3 for a completion behavior, then this behavior record will be used with 4 completion packets in a sequence before using the next completion behavior record.
Delay	If you select **Delay Enabled**, then the completion is delayed by the delay time period that you specified for the applicable queue in the Error Insertion page. Packets waiting behind the delayed completion in the same queue cannot pass the delayed completion. Packets can only pass the delayed completion using the other queue. If you select **Delay Disabled**, then the completion is sent within the allowed time gap as per the specifications without causing any manual delay. Default is **Delay Disabled**.
Gap	Specify the count for the gap that you want to insert between the headers of subsequent packets. The gap of the current packet will be taken until the next packet is started.
TLP Digest	• Select **Marked Absent and Absent** to clear the TLP Digest bit in the header of the packet indicating the absence of the TLP Digest field and accordingly change the format of the packet to exclude TLP Digest. • Select **Marked Present but Missing** to set the TLP Digest bit in the header of the packet indicating the presence of the TLP Digest field. However, the format of the packet is kept unchanged for the TLP Digest to introduce an error. • Select **Marked Present and Present** to set the TLP Digest bit in the header of the packet indicating the presence of the TLP Digest field and accordingly change the format of the packet for the TLP Digest. If you select **Marked Absent and Absent** or **Marked Present but Missing**, then the packet is sent without an ECRC. Default is **Marked Absent and Absent**.
ECRC	This field is displayed when you select the **Marked Present and Present** option for the **TLP Digest** field. Select **Incorrect** to insert an incorrect ECRC value in the TLP Digest field at the end of the packet to introduce an ECRC error. Default is **Correct**.

Component	Description
LCRC	Select **Incorrect** to insert an incorrect LCRC in the packet. Default is **Correct**.
Disparity	Select **Incorrect** to send a packet with an incorrect running disparity. Default is **Correct**. NOTE: If the link is operational at Gen3 speed, then selecting Incorrect for the Disparity field does not insert disparity error in the packet.
Payload size	Select whether the payload size of the packet should be correct or incorrect. Exerciser calculates the incorrect payload size by adding or subtracting 1 from the correct payload size in the LEN field of the packet. You can instruct Exerciser on whether to add or subtract 1 by specifying +1 or -1 in the Wrong Payload size behavior section of the Error Insertions -> Request and Completion Errors tab. Default is **Correct**.
TLP Poisoned	Select **Enabled** to indicate that the TLP is poisoned. Default is **Disabled**.
TLP Nullified	Select **Enabled** to indicate that the TLP is nullified. Default is **Disabled**.
Replace STP	Select **Enabled** to ensure that the Start TLP special symbol in the packet is replaced with a specified symbol to generate an error. You can specify this symbol with which the STP symbol should be replaced in the Replace STP with field of the **Error Insertions -> Request and Completion Errors** tab. Default is **Disabled** that means the STP symbol in the packet will not be replaced with any other symbol. NOTE: If the link is operational at Gen3 speed and you enabled the Replace STP for a packet, then the STP symbol in the packet is not replaced. Only one FCRC bit in the packet is reversed.
Replace END	Select **Enabled** to ensure that the End special symbol in the packet is replaced with a specified symbol to generate an error. You can specify this symbol with which the END symbol should be replaced in the Replace END with field of the Error Insertions -> Request and Completion Errors tab. Default is **Disabled** that means the END symbol in the packet will not be replaced with any other symbol. NOTE: If the link is operational at Gen3 speed, then enabling the Replace END for a packet does not replaces the END symbol in the packet.

Component	Description
Offset Sequence Number	Select **Enabled** to ensure that a specified offset is added to the sequence number of the packet to generate an incorrect sequence number for that packet. You can specify this offset in the Sequence Number Offset field of the Error Insertions -> Request and Completion Errors tab. Default is **Disabled** that means no offset is added to the sequence number ensuring correct sequence number for the packet.
Discard Completion	Select Enabled to ensure that Exerciser does not send those completions to DUT for which the completion behavior record with Enabled Discard completion setting is applicable. 1 means **Enabled** and 0 means **Disabled**. Default is **Disabled** that means completions are sent to DUT.

When you save the changes made, the behavior record displays the error bits that you have enabled in the behavior of the packet.

Edit Packet dialog box

You use this dialog box to change the default settings for a PCIe packet to be sent as stimulus to DUT.

To access this dialog: Double-click the PCIe packet to be edited from the **Single Packet** or **Block Transfers** tabs of the **Traffic Setup** page.

Component	Description
Field	The Field column contains the names of the fields applicable for a PCIe packet. The number of fields displayed in this column may differ for different types of packets. The fields that are displayed also depend on whether the PCIe packet is to be sent on an MR enabled or a non IOV PCIe link. For instance, for a PCIe packet to be sent over an MR enabled PCIe link, the fields such as Prefix tag, Global key, VL Number, and VH Number are also displayed in this column.

Component	Description
Value	The Value column allows you to change the default value of a field. This column also contains a drop-down list and a calculator for input assistance. The drop-down list enables you to specify your inputs in hexadecimal, binary, or decimal format. Clicking the calculator for input assistance displays the Input Assistant dialog box. Here, you can perform boolean operations over the hexadecimal, binary, and decimal values, and use the resulting value as the new value for the corresponding packet field. Some fields in the packet get their value depending on the value you set in the packet's behavior record. Such fields display the value "Depends on behavior". To know how to set the behavior of a packet, refer to the Edit Packet Behavior dialog box.
Length	The Length column displays the maximum length of a field in bits, bytes, or DWords.
Write Data To	This section is only applicable for a Read request packet to enable you to specify where the data for the completion received for such a packet should be saved. This section has the following components: • **Single Packet Memory -** Select this option if you want to save the completion data received for the single packet stimulus in Single Packet Memory of the Exerciser card. This option is only available when you are sending a single packet as stimulus using the Single Packet tab. • **Discard** - This option is not available in this release. • **Data Comparator -** This option is not available in this release. • **Data Memory** - Select this option if you want to save the completion received for the request packet into data memory of the Exerciser card. Selecting this option also enables the **At** text box, which enables you to specify the offset of the location where the completion data should be stored in memory. • **Length** - The **Length** text box enables you to specify the length (in DWords) of the completion data to be written at the specified memory address location. Maximum value permissible in this text box is 1024 DWs for sending a single packet and 2097151 DWs for sending block transfer.

Component	Description
Read Data From	This section is only applicable for a Write request packet to enable you to specify the address in the internal memory of the Exerciser card, from where the data for such a packet should be taken. This section has the following components: • **Single Packet Memory**: Select this option if you want to read the data for the single packet from Single Packet Memory of the Exerciser card. This option is only available when you are sending a single packet as stimulus using the Single Packet tab. • **Data Generator:** This option is not available in this release. • **Data Memory**: Select this option if you want to read the data for the packet from data memory of the Exerciser card. Selecting this option also enables the At text box, which enables you to specify the offset of the location from where the data should be read. • **Length**: The Length text box enables you to specify the length (in DWords) of the data that should be read from the specified memory address location.
Wait/Repeat Settings	The Wait/Repeat Settings section is available only for TLPs sent as block transfers (using the Block Transfers tab). This section provides the following components: • **Condition**: The list provides the following options to specify a criterion for delaying or repeating the sending of the underlying packet. • **Repeat until Pattern Matched**: Select this option when you want to repeat the transmission of the underlying packet until the enabled pattern matchers (1, 2, 3, and 4) are found in the incoming TLPs. If all pattern matchers are disabled, then the packet will not be repeated. • **Wait for Pattern Matched**: Select this option when you want to delay sending the underlying packet until all the enabled pattern matchers (1, 2, 3, and 4) are found in the incoming TLPs. If all pattern matchers are disabled, then the packet will not be delayed. • **Wait for Completions finished:** Select this option when you want to delay the sending of the underlying packet until all the previous outstanding requests have been completed. • **Wait for External Trigger In**: Select this option when you want to delay the sending of the underlying packet until an external trigger in signal is received. This signal is received from another test equipment on the Trigger In Connector component of the Exerciser card. • **Pattern**: Provides a list of check boxes (1, 2, 3, and 4) for the four available pattern matchers. These checkboxes are available for selection only when you select **Repeat Until Pattern Matched** or **Wait for Pattern Matched** options from the Wait/Repeat Settings drop-down list. • **Repeat Request**: This text box allows you to specify how many times you want Exerciser to repeat the transmission of the packet while sending a block of packets as stimulus.

Export Memory to File dialog box

To access this dialog: Select **File** > **Export** > **Export Data Memory to File**.

Component	Description
File Name	The **File Name** text box allows you to specify the complete path and name of the file, to which you want to export the data memory contents. You can also browse to the location where you want to save the data using the **Browse** command button, which is right-adjacent to the text box.
Start Address	The Start Address text box allows you to specify the start address of the range of the memory location addresses, whose data memory contents you want to export. There is a drop-down list and a calculator for input assistance in specifying the start address. The drop-down list allows you to specify the start address in hexadecimal, binary, or decimal format. Clicking the calculator for input assistance displays the Input Assistant dialog box. Here, you can perform boolean operations over the hexadecimal, binary, and decimal values, and use the resulting value as the new value for the start address.
Stop Address	The Stop Address text box allows you to specify the stop address of the range of the memory location addresses, whose data memory contents you want to export. There is a drop-down list and a calculator for input assistance in specifying the stop address. The drop-down list allows you to specify the stop address in hexadecimal, binary, or decimal format. Clicking the calculator for input assistance displays the Input Assistant dialog box. Here, you can perform boolean operations over the hexadecimal, binary, and decimal values, and use the resulting value as the new value for the stop address.

Find in Memory dialog box

To access this dialog: Select **Edit** > **Find in Data Memory**.

Component	Description
Number of Bytes to find	The **Number of Bytes to find** drop-down list provides the options to specify the number of bytes you want to find.

Component	Description
Data to find	The **Data to find** section provides the following components: • **Data**: The Data column allows you to type the data memory content that you want to find. • **ASCII**: The ASCII column displays the ASCII representation of the data in the Data column.
Find	Click **Find** to jump to the first occurrence of the data you are trying to find in the data memory.
Find Next	Click **Find Next** to the next occurrence of the data you are trying to find in the data memory.

Goto Address dialog box

To access this dialog: Select **Edit** > **Goto Data Memory Address**.

Component	Description
Goto Address	The Goto Address text box allows you to specify the data memory address, to which you want to jump. There is a drop-down list and a calculator for input assistance in specifying the desired memory address. The drop-down list allows you to specify the memory address in hexadecimal, binary, or decimal format. Clicking the calculator for input assistance displays the Input Assistant dialog box. Here, you can perform boolean operations over the hexadecimal, binary, and decimal values, and use the resulting value as the new value for the desired data memory address.

Import Memory from File dialog box

To access this dialog: Select **File** > **Import** > **Import Data Memory from File**.

Component	Description
File Name	The **File Name** text box allows you to specify the complete path and name of the file, from which you want to import the data memory contents. You can also browse to the location of the file, and select it to import using the **Browse** command button.

Component	Description
Start Address	The **Start Address** text box allows you to specify the start address of the range of the memory location addresses, whose data memory contents you want to import. There is a drop-down list and a calculator for input assistance in specifying the start address. The drop-down list allows you to specify the start address in hexadecimal, binary, or decimal format. Clicking the calculator for input assistance displays the Input Assistant dialog box. Here, you can perform boolean operations over the hexadecimal, binary, and decimal values, and use the resulting value as the new value for the start address.
Stop Address	The **Stop Address** text box allows you to specify the stop address of the range of the memory location addresses, whose data memory contents you want to import. There is a drop-down list and a calculator for input assistance in specifying the stop address. The drop-down list allows you to specify the stop address in hexadecimal, binary, or decimal format. Clicking the calculator for input assistance displays the Input Assistant dialog box. Here, you can perform boolean operations over the hexadecimal, binary, and decimal values, and use the resulting value as the new value for the stop address.

Initialize Memory dialog box

To access this dialog: Select **Action** > **Initialize Data Memory**.

Component	Description
Fill Type	The **Fill Type** drop-down list provides the following options to fill the data memory contents: • **Initialize**: Select this option when you want to initialize the memory contents of a specified range of memory addresses with a particular value. • **Increment**: Select this option when you want to initialize the memory contents of a specified range of memory addresses with incremental values. In this case, you initialize the starting address with a start value, and then increments the value for every next memory location till it reaches to the end of the range. • **Pattern**: Select this option when you want to initialize the memory contents of a specified range of memory addresses in a particular data pattern.

Component	Description
Start Address	The **Start Address** text box is used to specify the starting address of the range of data memory locations that you want to initialize.
Stop Address	The **Stop Address** text box is used to specify the end address of the range of data memory locations that you want to initialize.
Value	The **Value** text box is used specify the value to initialize the range of data memory locations that you specified in the Start Address and Stop Address text boxes. The Value text box is available only when you select the Initialize option in the Fill Type drop-down list.
Start Value	The **Start Value** text box is used to specify the value to initialize the data memory address location that you specified in the Start Address text box. The Start Value text box is available only when you select the Increment option in the Fill Type drop-down list.
Increment by	The **Increment By** text box is used to specify the incremental value, which increments the value specified in the Start Value text box, for every memory location greater than the one specified in Start Address, but smaller than or equal to the one specified in Stop Address. The Increment By text box is available only when you select the Increment option in the Fill Type drop-down list.
Pattern 1	The **Pattern 1** text box is used to specify the value to initialize all the odd numbered DWords, such as 1, 3, 5, 7, 9, 11, and so on. This text box is displayed on when you select the Pattern option in the Fill Type drop-down list.
Pattern 2	The **Pattern 2** text box is used to specify the value to initialize all the even numbered DWords, such as 2, 4, 6, 8, 10, 12, and so on. This text box is displayed on when you select the Pattern option in the Fill Type drop-down list.

Open dialog box

To access this dialog: Select **File** > **Open**. Select the file to open. Then, click **Open**.

Component	Description
Load following items	The **Load following items** list box provides you an expandable/collapsible check box list of exerciser settings. Here, select only those settings that you want to load from the *.exs file.

Component	Description
Description of file	The **Description of file** text box displays a brief description of the setup file, if exists.

Port Selection dialog box

Component	Description
Module	The Module column provides a list of modules in the form of check boxes. These check boxes are organized under the expandable list of chassis to which they belong. Here, select a module on which you want to start a new session. A module, which is already associated with a session, appears disabled.
Type	The Type column shows the link width along with the speed of the port, such as 5.0Gb/s or 2.5Gb/s, at which it operates.
# Ports	The # Ports column displays the number of ports available for each module.
License	The License column displays the license details of each module.
Status	The Status column displays the status of each module. This column displays the following messages: • **Rebooting**: This message indicates that the serial I/O module is restarting. • **Ready**: This message indicates that you can now add port to the session. • **Session_Name**: Session_Name is the name of the session. This message indicates that the Session_Name session is now connected with the port.
Select Ports to use	Select the port to use from the **Select Ports to use** list box.

Preferences dialog box

To access this dialog: Select **View** > **Preferences**.

Component	Description
Request Settings	These settings are applicable for the TLP requests that you add to the Block Transfers and Single Packet tabs of the Traffic Setup page. These settings are grouped under the following three categories: • **General** - You can select the **Enable line wrap** checkbox to display the packet contents in multiple lines in the configuration area without extending the display to the right. • **Collapsed View** - The Collapsed View enables you to specify the fields and the order in which these fields should be displayed when a TLP request is in a collapsed form in the configuration area. The Available fields list displays all the fields that you can add for a TLP request. The Displayed fields list displays all the fields that are displayed in the configuration area for a TLP request. You can move a field from the Available fields list to Displayed Fields list using the displayed Arrow key buttons. • **Expanded View** - The Expanded View enables you to specify the fields and the order in which these fields should be displayed when a TLP request is in an expanded form in the configuration area. The Available fields list displays all the fields that are displayed for a TLP request. The Hidden fields list displays all the fields that are not displayed in the configuration area. You can move a field from the Available fields list to Hidden Fields list using the displayed Arrow key buttons. • **Colors** - Enables you to specify color schemes for different packet types displayed in the Configuration area. This helps you to set separate colors for each packet type to easily recognize these in the Configuration area.
Behavior Settings (Single Packet)	These settings are applicable for the behavior records that are added to the Traffic Setup -> Single Packet tab for sending a single packet as stimulus. These settings are grouped under the following three categories: • **General** - You can select the **Enable line wrap** checkbox to display the packet contents in multiple lines in the configuration area without extending the display to the right. • **Collapsed View** - The Collapsed View enables you to specify the fields and the order in which these fields should be displayed when the request behavior is in a collapsed form in the configuration area. The Available fields list displays all the fields that you can add for request behaviors. The Displayed fields list displays all the fields that are displayed in the configuration area for a request behavior. You can move a field from the Available fields list to Displayed Fields list using the displayed Arrow key buttons. • **Expanded View** - The Expanded View enables you to specify the fields and the order in which these fields should be displayed when the request behavior is in an expanded form in the configuration area. The Available fields list displays all the fields that are displayed for request behaviors. The Hidden fields list displays all the fields that are not displayed in the configuration area. You can move a field from the Available fields list to Hidden Fields list using the displayed Arrow key buttons.

Component	Description
Behavior Settings (Block Transfer)	These settings are applicable for the behavior records that you add to the Traffic Setup -> Block Transfers tab for the TLP requests. These settings are grouped under the following three categories: • **General** - You can select the **Enable line wrap** checkbox to display the packet contents in multiple lines in the configuration area without extending the display to the right. • **Collapsed View** - The Collapsed View enables you to specify the fields and the order in which these fields should be displayed when the request behavior is in a collapsed form in the configuration area. The Available fields list displays all the fields that you can add for request behaviors. The Displayed fields list displays all the fields that are displayed in the configuration area for a request behavior. You can move a field from the Available fields list to Displayed Fields list using the displayed Arrow key buttons. • **Expanded View** - The Expanded View enables you to specify the fields and the order in which these fields should be displayed when the request behavior is in an expanded form in the configuration area. The Available fields list displays all the fields that are displayed for request behaviors. The Hidden fields list displays all the fields that are not displayed in the configuration area. You can move a field from the Available fields list to Hidden Fields list using the displayed Arrow key buttons.
Completion Behavior Settings	These settings are applicable for the completion behavior records that you add to the Traffic Setup -> Completion Behavior tab. These settings are grouped under the following three categories: • **General** - You can select the **Enable line wrap** checkbox to display the packet contents in multiple lines in the configuration area without extending the display to the right. • **Collapsed View** - The Collapsed View enables you to specify the fields and the order in which these fields should be displayed when the completion behavior is in a collapsed form in the configuration area. The Available fields list displays all the fields that you can add for completion behaviors. The Displayed fields list displays all the fields that are displayed in the configuration area. You can move a field from the Available fields list to Displayed Fields list using the displayed Arrow key buttons. • **Expanded View** - The Expanded View enables you to specify the fields and the order in which these fields should be displayed when the completion behavior is in an expanded form in the configuration area. The Available fields list displays all the fields that are displayed for completion behaviors. The Hidden fields list displays all the fields that are not displayed in the configuration area. You can move a field from the Available fields list to Hidden Fields list using the displayed Arrow key buttons.

Save dialog box

To access this dialog: Select **File** > **Save**.

Component	Description
Save following items	The **Save following items** list box provides you an expandable/collapsible check box list of exerciser settings. Here, select only those settings that you want to save in the *.exs file for later use.
Description for Setup file	The **Description for Setup file** text box allows you to specify a brief description for the setup file.

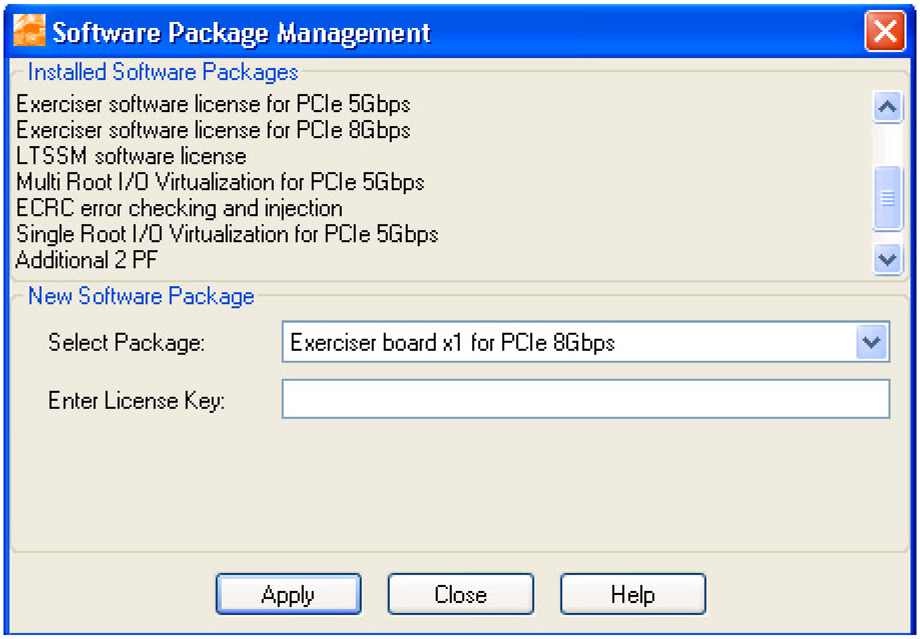
Select type of connection dialog box

To access this dialog: Click **Start** > **Programs** > **Agilent SPT** > **PCIe Exerciser 8.2 Release** > **Exerciser GUI**.

Component	Description
Connect to existing session	Select **Connect to existing session** if you want to use an existing session of Protocol Exerciser on a known server. The existing session can be either initiated by the API or by another GUI since multiple GUIs can share the same session.
Connect to new session	Select **Connect to new session** if you want to create a new Protocol Exerciser session on a known server.
Offline mode	Select **Offline** mode if you want to access the Protocol Exerciser GUI without connecting to the hardware.
Server	Specifies the name or IP address of the server, where you want to start a new or join an existing Protocol Exerciser session. Its default value is **localhost**.
Get session list	Click **Get session list** to retrieve a list of existing Protocol Exerciser sessions from the server you specified in the **Server** text box. This button works only if you have selected the **Connect to existing session** option.
Session list	Displays a list of sessions existing on the server you specified in the **Server** text field.
Start	Click **Start** to start a new Protocol Exerciser instance.
Exit	Click **Exit** to close the dialog box without starting the Protocol Exerciser application.

Software Package Management dialog box

To access this dialog: Click **File** > **Software Packages**.



Exerciser software license for PCIe 5Gbps
Exerciser software license for PCIe 8Gbps
LTSSM software license
Multi Root I/O Virtualization for PCIe 5Gbps
ECRC error checking and injection
Single Root I/O Virtualization for PCIe 5Gbps
Additional 2 PF

Select Package: Exerciser board x1 for PCIe 8Gbps

Enter License Key:

Component	Description
Installed Software Packages	The Installed Software Packages section displays a list of currently installed software packages.

Component	Description
New Software Packages	The New Software Packages section provides the following components: • **Select Package**: Click the **Select Package** drop-down list to select a new software to be installed. In this release, the following licensed software options are available: • Exerciser board x1 for PCIe 8Gbps • Exerciser board x4 for PCIe 8Gbps • Exerciser board x8 for PCIe 8Gbps • Exerciser board x16 for PCIe 8Gbps • Exerciser software license for PCIe 5Gbps • Exerciser software license for PCIe 8Gbps • LTSSM software license - This license adds LTSSM functionality to Exerciser. With this license, Exerciser can function as an LTSSM for thorough link testing and exchanging training sequences with DUT. • Multi Root I/O Virtualization for PCIe 5Gbps - This license adds MRIOV capabilities to Exerciser at Gen2 speed. With this license, Exerciser can emulate a MRIOV root complex and endpoint. • ECRC error checking and injection - This license adds the support for the ECRC error checking in the TLPs received from DUT and the ECRC error insertion in the TLPs that Exerciser transmits to DUT. • Single Root I/O Virtualization for PCIe 5Gbps - This license adds SRIOV capabilities to Exerciser at Gen2 speed. With this license, Exerciser can emulate a SRIOV root complex and endpoint. • Additional 2 PF - This license provides the additional two physical functions for Exerciser making it a total of five physical functions. The additional functions are supported across all the three configurations (non IOV, SRIOV, and MRIOV) of Exerciser. • **Enter License Key**: Type the license key of the software that you selected in the **Select Package** drop-down list.

Menus

Action Menu

The following is a brief description of the Action menu commands.

Menu command	Description
Link Up	Performs a Link-Up between Protocol Exerciser and DUT as per the configured link and lane settings.
Initiate Speed change	Initiates speed change to the highest data rate advertised by both the link partners during link training.
Initiate Retrain Link	Retrains the Link.
Disable Exerciser Transmitter	Disables the transmission from Exerciser.
Reload Data Memory	Reloads the contents of the Data Memory of Exerciser. This option refreshes the Data column in the **Data Memory** -> **Edit Memory** tab with the most recent values from the data memory.
Start all Functions	Starts the transmission of stimulus from Exerciser to DUT as per the configured block of packets added across the supported functions.
Start Individual Functions	Sends the block of packets that you added to the selected function as stimulus to DUT. Packets that you add to other functions are not transmitted in this case.
Stop all Functions	Stops the transmission of stimulus from Exerciser to DUT for all the supported functions.
Stop Individual Functions	Stops the transmission of stimulus from Exerciser to DUT for the packets added to the selected function.
Initialize Data Memory	Initializes the entire data memory or a specific range of data memory addresses with the value specified in the Initialize Memory dialog box.
Send Single Packet	Sends a single TLP packet as stimulus to DUT as per the packet and its request behavior configured in the **Traffic Setup** -> **Send Single Packet** tab.

Edit Menu

The following is a brief description of the Edit menu commands.

Menu command	Description
Find in Data Memory	Allows finding the occurrences of a specified pattern in the Exerciser's Data Memory. The option is enabled only in the **Data Memory** page of the GUI. This option displays the occurrences of the specified pattern in the Data column of the **Edit Memory** tab of the Data Memory page.
Goto Data Memory Address	Allows you to go to a specified memory address of the Exerciser Data Memory. This option is enabled only in the **Data Memory** page of the GUI. The **Edit Memory** tab in this page displays the Address, Data, and ASCII representations of the contents of Data Memory. The Goto Data Memory Address option moves the cursor to the specified memory address in the Edit Memory tab.

File Menu

The following is a brief description of the File menu commands.

Menu command	Description
New	Opens the port selection dialog box.
Open	Displays the Open dialog box that enables you to open the saved session information.
Save	Displays the Save dialog box that enables you to save the session information. You can enter the location and a file name for the data to be saved. After confirming these entries an additional dialog pops up as seen i the figure below. You can define if all or a subset of the Exerciser setting should be saved. As default all possible data will be saved.
Save As	Displays the Save As dialog box that enables you to specify the desired location to save the session information.
Close	Closes the selected session.

Menu command	Description
Import Data Memory from File	Imports the data from the specified file as the contents for the Data memory of Exerciser. The imported data is displayed in the **Data Memory** -> **Edit Memory** tab. This option is enabled only in the **Data Memory** page of the GUI.
Export Data Memory to File	Exports the contents of the **Data memory** of Exerciser to a specified file. This option is enabled only in the **Data Memory** page of the GUI.
Exit	Closes the Protocol Exerciser application.

Help Menu

The Help menu provides access to:

Menu command	Description
Help Topics	Opens the main help menu.
Session Information	Provides an information dialog containing the IP address of the server the module is connected to, the session ID and the used module.
Version Information	Provides an information dialog containing the software version number of the Exerciser solution, outlining the different elements of the solution.
About	Displays the about dialog.

Please have the complete version information available in case you need to contact Agilent support.

View Menu

The following is a brief description of the View menu commands.

Menu command	Description
Navigation Window	Displays the Navigation Window.
Hardware Status Window	Display the Hardware Status Window.

Windows

Config Space function

To access this function: Click **Config Space** in the Navigation pane.

The **Config Space** function allows you to set up the configuration space for each function supported by Exerciser. It provides the following areas:

- PCI Common Configuration Header
- Power Management
- MSI
- PCIE Capabilities
- PCIE Virtual Channel
- Secondary PCIe Extended Capabilities
- ARI

If Exerciser is emulating an MRIOV capable PCIe component, then the following two additional tabs are displayed in the configuration space.

- **MRIOV Capability** block for Function A (To initialize and configure the MRIOV capabilities of the Base function - Function A of Exerciser)
- **SRIOV Capability** block for Function B to E (To initialize and configure the SRIOV capabilities of physical functions (B to E) of Exerciser)

If Exerciser is emulating an SRIOV capable PCIe component, then the following additional tab is displayed in the configuration space.

- **SRIOV Capability** block for Function B to E (To initialize and configure the SRIOV capabilities of physical functions (B to E) of Exerciser)

The different tabs for the configuration space are aligned according to the PCI, PCI-X and PCI Express specification and provide a wealth of settings, which are well documented in the related specifications.

Data entry is provided either via check boxes or via numeric entry. A calculator allows entering the values in a convenient way using the desired number format.

Each register in the Configuration Space has different register types associated – e.g. read/ writable or read-only. To override changing of parameters, they can be set to read/ writable at the GUI by using the right mouse button.

The color code indicates the current state that each register bit is in – spec. compliant or read/ writable (blue). For registers, a summary color code (yellow) is also provided in cases where mixed values are used.

Data Memory function - Edit Memory tab

To access this function: Click **Data Memory** in the Navigation pane. Then click the **Edit Memory** tab.

This tab displays the contents of the Data memory of Exerciser in the form of Address, Data, and the ASCII representation of the Data. The size of the data memory is 256 KByte. You use this tab to view or edit the contents of the data memory.

Refer to the topic Configuring and Using Data Memory of Exerciser to know more about the data memory of Exerciser.

Menu command	Description
Address	This column displays the addresses that represent the data memory locations.
Data	This column contains the actual data written in the data memory against the corresponding address (memory location). In this column, you can also write new or edit any existing data against any data memory location by simply placing the cursor at the desired location and then typing the new data. On applying the changes, the changes are saved to the data memory. Besides editing the contents of the Data column manually, you can also use the Initialize Memory dialog box to initialize the entire memory or the specified range of memory locations with the specified new data.
ASCII	This column displays the ASCII representation of the data written in the Data column.
Data Display Format	Select an option to specify the format of the data written in the Data column.
Bytes per line	Select an option to specify the number of bytes to be shown in the **Data** column for each row of the Data Memory contents displayed in the Edit memory tab.

Data Memory function - Memory Compare tab

To access this function: Click **Data Memory** in the Navigation pane. Then click the **Memory Compare** tab.

You use this tab to:

- enable or disable the comparison of the payload of incoming completions with the expected data at a memory address. You can do this setting separately for each supported physical and virtual function of Exerciser.
- view the status of the comparison done for the payload of requests and completions with the memory contents at a targeted data memory address.

Refer to the topics Configuring and Using Data Memory of Exerciser and Comparing Actual Data with Expected Data in Memory in Memory to know more about data memory of Exerciser.

Menu command	Description
Function	You can set and view the following information for each physical and virtual function of Exerciser. • **Enable Memory Compare for completions** check box. If you select this checkbox, Exerciser compares the payload of the incoming completions for that function with the memory contents at a specified data memory location. You specify the data memory location for the comparison of these completion payloads in the **Decoder** page of the GUI. Refer to the topic Configuring Decoder Settings to know more about this page. If you want to compare the payload of the incoming TLP requests with memory contents, then you need to set this option for the relevant decoder(s) in the Decoder page. • The next section displays information on the memory compare operation for the payload of the incoming requests and completions. The following information is displayed. • **Status**: Displays the Error message when there is any compare error and the No Errors message when there is no compare error between the incoming payload and the data in the targeted memory location. • **Address**: Displays the address of the data memory used to perform the comparison of payloads with the targeted data at that address. • **Expected Data**: Displays the data at the targeted memory location with which the payload is compared. • **Actual Data**: Displays the actual data present in the TLP payload which was compared with the expected data in the memory. **Note**: The data comparison information of incoming TLP requests is displayed only when you have selected Compare and specified a targeted memory location for comparison for the relevant decoders in the **Decoder** page. For the data comparison for completions, you also need to select the **Enable Memory Compare for completions** check box besides setting the data comparison settings for the decoder.
Check all	Click **Check all** to enable memory compare for completions of all the displayed virtual and physical functions of Exerciser.
Uncheck all	Click **Uncheck all** to disable memory compare for completions of all the displayed virtual and physical functions of Exerciser.
Clear Status	Click **Clear Status** to clear all the information displayed for memory comparison.
Refresh	Click **Refresh** to refresh the status of all the information displayed for memory comparison.

Decoder function

To access this function: Click **Decoder** in the Navigation pane.

You use this page to configure the settings for the decoder available for each of the six BARs and Expansion ROM of a function of Exerciser.

Refer to the topic Configuring Decoder Settings to know more about decoder BARs of Exerciser.

Menu command	Description
Decoder	You can use the checkbox displayed with a decoder to enable or disable that decoder. If you disable a decoder, then the corresponding BAR of the function is not available for TLPs requesting memory space.
Location	The **Location** drop-down list provides the following options: • **I/O**: Select this option when you want only I/O packets to hit the underlying decoder. • **Memory (32 bit):** Select this option when you want only 32 bit Memory or Config packets to hit the underlying decoder. • **Memory (64 bit):** Select this option when you want only 64 bit Memory or Config packets to hit the underlying decoder.
Prefetchable	Select **Yes** or **No** to enable or disable the Prefetchable attribute bit of the data memory block used by the underlying decoder. The option is available only for a memory decoder.
Base Address	The **Base Address** text box allows you to specify the base address criteria for the packets that should hit the underlying decoder. This base address, when matches with the Address field of an incoming packet, permits the packet to hit the underlying decoder. There is a drop-down list and a calculator for input assistance in specifying the pattern of the base address. The drop-down list allows you to specify the pattern in hexadecimal, binary, or decimal format. Clicking the calculator for input assistance displays the Input Assistant dialog box. Here, you can perform boolean operations over the hexadecimal, binary, and decimal values, and use the resulting value as the new value for the base address pattern.
Size	The **Size** drop-down list provides the following options to specify the offset criteria for the packets that you want to hit the underlying decoder: • For I/O, it provides options ranging from 4 bytes to 2GB. • For Memory 32 bit, it provides options ranging from 128 bytes to 2GB. • For Memory 64 bit, it provides options ranging from 128 bytes to 8EB.

Menu command	Description
Data Memory Internal Size	The **Data Memory Internal Size** drop-down list provides options ranging from 32 bytes to 256 KB. You can select one of these options to specify the internal memory size for the decoder. Protocol Exerciser uses a wrap around method if the decoder size is larger than the available data memory. Also, the contents of the data memory gets overwritten if you set up multiple decoders without properly aligning the data memory base address and size.
Resource	The **Resource** drop-down list provides the following options: • **Discard**: Select this option if you want to discard the data of the incoming TLPs without performing any operation on the data. • **Compare**: Select this option if you want to compare the incoming TLP payload with the contents of the targeted data memory location. Selecting this option will only compare the payload of incoming TLPs and will not write the payloads to the targeted memory location. • **Data Memory:** Select this option if you want to let the incoming Read packets read the contents from the targeted memory location, and let the incoming Write packets write the contents to the targeted memory location.
Data Memory Base Address	The **Data Memory Base Address** text box is used to specify the base address in the internal data memory of Exerciser. This memory address is used to store the payload of incoming TLP requests and to get the payload of the outgoing completions if **Data Memory** is selected in the **Resource** listbox. This memory address is compared with the payload of incoming TLPs if **Compare** is selected in the **Resource** listbox. There is a drop-down list and a calculator for input assistance in specifying the base address. The drop-down list allows you to specify the base address in hexadecimal, binary, or decimal format. Clicking the calculator for input assistance displays the Input Assistant dialog box. Here, you can perform boolean operations over the hexadecimal, binary, and decimal values, and use the resulting value as the new value for the base address.

Error Insertion function - ACK/NAK Errors tab

To access this function: Click **Error Insertion** in the Navigation pane. Then, click the **ACK/NAK Errors** tab.

Menu command	Description
Sequence Number to trigger	The **Sequence Number to trigger** text box enables you to specify the sequence number for the packets, for which you want Exerciser to respond with NAK instead of ACK. Here, you can specify the sequence number in hexadecimal, binary, or decimal format by selecting the desired format from the drop-down list adjacent to the text box.
Trigger n times	The **Trigger n times** text box enables you to specify the number of times Exerciser responds with NAK for the sequence number specified in the Sequence number to trigger text box.
Start Mode	The **Start Mode** section has the following components: • **Start automatically when 'Run' action is selected:** Select this option to automatically insert ACK/NAK errors every time you run Protocol Exerciser. You can run Protocol Exerciser by selecting the **Action > Run** menu command, or by clicking the Run icon on toolbar. Selecting this option disables the Start Now command button. • **Start manually (Click on 'Start Now' button)**: Select this option to manually operate the error insertion activity. Selecting this option activates the Start Now command button. • **Start Now**: Click **Start Now** to activate the error insertion function for ACK/NAK, and apply the specified error criteria to the next packet with the sequence number shown in the Sequence number to trigger text box.
Start Now	The **Start Now** button activates the error insertion function for ACK/NAK, and applies to the next packet with the sequence number shown in the Sequence number to trigger text box.
Stop Now	The **Stop Now** button stops the function irrespective of how it was started.

Error Insertion function - DLLP tab

To access this function: Click **Error Insertion** in the Navigation pane. Then, click the **DLLP** tab.

Menu command	Description
Send arbitrary DLLP	Click **Send arbitrary DLLP** to display the DLLP Send dialog box, which you can use to send a DLLP.

Menu command	Description
DLLP Wrong CRC insertion	The **DLLP Wrong CRC insertion** section has the following components: • **Field**: This column contains the names of the fields of a DLLP. The number of fields displayed in this column may differ for different types of DLLPs. • **Value**: This column contains an input component for each field. You can specify the desired inputs in these components to edit the selected DLLP. This column also contains a drop-down list and a calculator for input assistance. The drop-down list enables you to specify your inputs in hexadecimal, binary, or decimal format. Clicking the calculator for input assistance displays the Input Assistant dialog box. Here, you can perform boolean operations over the hexadecimal, binary, and decimal values, and use the resulting value as the new value for the corresponding field. • **Length**: This column displays the maximum length of a field in bits, bytes, or DWords.
Status	The **Status** section has the following components: • **Clear Status:** Clicking Clear Status, after you have clicked Start, schedules a wrong CRC insertion into a DLLP that matches the pattern specified in the DLLP Wrong CRC insertion section. Clicking Clear Status, after you have clicked Stop, has no effect. • **Status**: Displays the Error is inserted message when a wrong CRC is inserted in a DLLP, and the No error inserted message otherwise.
Start	Click **Start** to schedule the wrong CRC insertion into a DLLP that matches the pattern specified in the Dllp Wrong CRC insertion section. Clicking Start replaces the Start command button with the Stop command button. This Stop command button is visible until the scheduled wrong CRC insertion is done or you click it to replace it with the Start command button.
Stop	Click **Stop** to cancel the scheduled wrong CRC insertion into a DLLP. The Stop command button is displayed when you click the Start command button.

Error Insertion function - Request and Completion Errors tab

To access this function: Click **Error Insertion** in the Navigation pane. Then, click the **Request and Completion Errors** tab.

You use this tab to specify the error settings to be used for the request or completion packets in which you have inserted error(s) using the Edit Packet Behavior dialog box.

Refer to the topic Testing DUT Under Error Conditions to know more.

Menu command	Description
Replace STP/END	The **Replace STP/END** section provides the following fields: • **Replace STP with**: Select a symbol that should replace the STP symbol of a packet to generate an error. The symbol selected here is used only for the packets for which you have set the Replace STP behavior property to Enabled. • **Replace END with**: Select a symbol that should replace the END symbol of a packet to generate an error. The symbol selected here is used only for the packets for which you have set the Replace END behavior property to Enabled. In case you want to specify the symbols that do not exist in these drop-down lists, select the **Enter value** option from the desired drop-down list. This enables the text box adjacent to the drop-down list. Here, type in the replacement symbols in hexadecimal, binary, or decimal format. You can also use the Input Assistance dialog box.
Offset Sequence Number	Allows you to specify the offset number that gets added to the sequence number of a packet to generate an incorrect sequence number. This offset number is applicable only for the packets for which you have inserted Offset Sequence Number error by setting the Offset Sequence Number behavior property to Enabled.

Menu command	Description
Delay Completion	The **Delay Completion** section provides the following fields: • **VC0 / (VH0, VL0, VC0) Completion Delay (ns)**: Specify the time in nanoseconds to delay the completions belonging to Queue0. This queue maps to: • virtual channel (VC0) in case Exerciser emulates a non IOV/SRIOV component • virtual hierarchy VH0 in case Exerciser emulates a MRIOV component. All virtual hierarchies map to VC0. • **VCx / (VH1, VL0, VC0) Completion Delay (ns):** Specify the time in nanoseconds to delay the completions belonging to Queue1. This queue maps to: • virtual channel (VCx) in case Exerciser emulates a non IOV/SRIOV component • virtual hierarchy VH1 in case Exerciser emulates a MRIOV component. All virtual hierarchies map to VC0. • **VH2, VL0, VC0 Completion Delay (ns)**: Specify the time in nanoseconds to delay the completions belonging to Queue2. This queue maps to virtual hierarchy VH2 in case Exerciser emulates a MRIOV component. All virtual hierarchies map to VC0. This field is disabled if Exerciser emulates a non IOV/SRIOV component as Exerciser supports only two queues (0 and 1) in these configurations. • **VH3, VL0, VC0 Completion Delay (ns)**: Specify the time in nanoseconds to delay the completions belonging to Queue3. This queue maps to virtual hierarchy VH3 in case Exerciser emulates a MRIOV component. All virtual hierarchies map to VC0. This field is disabled if Exerciser emulates a non IOV/SRIOV component as Exerciser supports only two queues (0 and 1) in these configurations. • **VH4, VL0, VC0 Completion Delay (ns):** Specify the time in nanoseconds to delay the completions belonging to Queue4. This queue maps to virtual hierarchy VH4 in case Exerciser emulates a MRIOV component. All virtual hierarchies map to VC0. This field is disabled if Exerciser emulates a non IOV/SRIOV component as Exerciser supports only two queues (0 and 1) in these configurations. The completion delay specified for the queues is applicable for delaying only those completion packets for which you have set the Delay property to Delay Enabled.
Replay Timer	Allows you to set the timer to resend the packet if there is no response (ACK/NAK) for it in the specified time units. Here, 1 time unit = 2 symbol times. Note that when large payload size packets are being transmitted, specifying a small replay timer value may result into unexpected Protocol Exerciser behavior.

Menu command	Description
Wrong Payload Size Behavior	Allows you to control how Exerciser should modify the value of the LEN field of a packet to generate an incorrect payload size for that packet. You can either select +1 to indicate that the value of LEN field has to be incremented by 1 or select -1 to indicate that the value of LEN field has to be decrement by 1. This calculation for the LEN field is applicable only for packets for which you have set the Payload size property to Incorrect.

General Settings Function Equalization Settings tab

To access this function: Click **General Settings** in the Navigation pane. Then, click the **Equalization Settings** tab.

Refer to the topic Configuring Equalization Procedure Settings in the main online help of Protocol Exerciser to know more about the Equalization implementation in Protocol Exerciser.

If you select the MRIOV or SRIOV protocol for creating a link between Protocol Exerciser and DUT, then the Equalization settings that you specify in this tab do not have any impact.

The **Equalization Settings** tab is broadly divided into two groups - **5.0 GT/s** and **8 GT/s** allowing you to configure the Equalization settings separately for 5.0 GT/s and 8 GT/s speeds. The **8 GT/s** group is further organized into two groups - **Request to DUT** and **Exerciser**. The following table describes fields in all these groups and subgroups.

Field	Description
5.0 GT/s	
De-emphasis shown in TS	Provides options to specify the Tx De-emphasis settings in training sequences (TS1 and TS2). You can set the de-emphasis level that the Protocol Exerciser will use for the transmission of training sequences at 5.0 GT/s.
8 GT/s	
Request to DUT	This group box provides options to configure the Equalization settings that the Protocol Exerciser will send to DUT in the training sequences to request DUT to use the specified presets and co-efficient values.
Tx Preset Value	Options to set the Tx De-emphasis preset value that the Protocol Exerciser will send to DUT in the training sequences to request DUT to use the specified preset.

Field	Description
Request Preset in Phase 2/Phase3	Select this checkbox to ensure that Protocol Exerciser instructs the DUT to use only the presets from the incoming TS OS. If you do not select this checkbox, then DUT has to use both presets as well as coefficients from the incoming TS OS. Based on whether or not the checkbox is selected, Protocol Exerciser instructs the DUT by sending TS1 OS in Phase2/Phase3 of equalization.
Tx pre cursor coefficients	Options to set the Tx coefficient –C1 value from the Tx Equalization Finite Impulse Response Filter.
Tx data cursor coefficients	Options to set the Tx coefficient C0 value from the Tx Equalization Finite Impulse Response Filter.
Tx post cursor coefficients	Options to set the Tx coefficient +C1 value from the Tx Equalization Finite Impulse Response Filter.
Rx preset hints	Options to set the preset value that the Protocol Exerciser will send to the DUT as a Hint for the Rx Equalization value based on the trace length of the DUT.
Reject preset N times	Specify the number of times the DUT can reject preset/coefficient values (requested by Protocol Exerciser) before accepting these values in Phase 3 of equalization.
Exerciser	This group box provides options to configure Equalization settings that the Protocol Exerciser will use and advertise as its Tx capabilities during the link training.
FS	Options to set the value for Full Swing (FS) resolution for de-emphasis. Protocol Exerciser advertises this value in Phase 1 of Equalization.
LF	Options to set the value for Low Frequency (LF) resolution for de-emphasis.
Tx Preset Value	Options to set the Tx De-emphasis preset value that the Protocol Exerciser advertises as its Tx capabilities during the link training at 8.0 GT/s.
Tx pre cursor coefficients	Options to set the Tx coefficient -C1 value from the Tx Equalization Finite Impulse Response Filter. The three Tx co-efficient values set for the Exerciser should add up to the specified FS value.
Tx data cursor coefficients	Options to set the Tx coefficient C0 value from the Tx Equalization Finite Impulse Response Filter. The three Tx co-efficient values set for the Exerciser should add up to the specified FS value.
Tx post cursor coefficients	Options to set the Tx coefficient +C1 value from the Tx Equalization Finite Impulse Response Filter. The three Tx co-efficient values set for the Exerciser should add up to the specified FS value.
Ignore coefficient rule checks	Selecting these check boxes allow you to break the specified co-efficient rule(s) so that you can test how the DUT responds when a co-efficient rule is ignored.

Field	Description
Perform Equalization in Recovery @8 GT/s	The **Perform Equalization in Recovery @8 GT/s** option ensures that Protocol Exerciser (acting as a USC) must perform Equalization when upgrading the link speed to 8.0 GT/s. If this option is not selected, then the link speed is upgraded without performing equalization. A USC performs the phases of equalization. Therefore, this option is applicable only for a USC and is disabled when Protocol Exerciser acts as a DSC, that is when the Session type is configured as ToUpstream.
Request Equalization in Recovery @8 GT/s	The **Request Equalization in Recovery @8 GT/s** option ensures that Protocol Exerciser (acting as a DSC) should request the USC to redo equalization. This option is applicable only for a DSC. Therefore, this option is disabled when Protocol Exerciser acts as a USC, that is the Session type is configured as ToDownstream.
Perform Phase 2/Phase 3	Selecting the **Perform Phase 2/Phase 3** option instructs Protocol Exerciser to perform the phase 2 and phase 3 of the Equalization procedure before moving the link to 8 GT/s. If this option is not selected, then Protocol Exerciser exits the Recovery.Equalization sub-state after phase 1 and transitions to the Recovery.RcvrLock substate to proceed towards achieving the 8.0 GT/s speed. A USC performs the phases of equalization. Therefore, this option is applicable only for a USC and is disabled when Protocol Exerciser acts as a DSC, that is when the Session type is configured as ToUpstream.

General Settings Function Lane Settings tab

To access this function: Click **General Settings** in the Navigation pane. Then, click the **Lane Settings** tab.

Field	Description
Reverse Lanes	Provides the following option buttons for transmitter (Tx): • **Yes**: Select to enforce the lanes reversal manually on physical lanes by reversing the logical lane numbers. • **No**: Select to not enforce lane reversal.
Lane Reversal Capability	Provides the following option buttons for transmitter (Tx): • **Disabled**: Automatically deactivates the lane reversal capability. • **Enabled**: Automatically activates lane reversal capability when needed.
Lane Polarity	Provides options to set the polarity of the individual lanes to **Normal** or **Inverted** for transmitter (Tx). Lane numbers available for lane polarity setting vary from 0 to 15 based on the advertised link width.

General Settings Function Link Settings tab

To access this function: Click **General Settings** in the Navigation pane. Then, click the **Link Settings** tab. Refer to the topic Initializing and training a PCIe Link between Protocol Exerciser and DUT in the Online help to know more about how to create a PCIe link.

Field	Description
Protocol	Select the protocol based on which you want to create the link between the Protocol Exerciser and DUT. You can select from the following two options: • **PCIe** - Use this option to create a non IOV (Input Output Virtualization) PCIe link between the Protocol Exerciser and DUT. The PCIe link is created as per the PCI Express 3.0 Base Specifications. Select this option when you want to stimulate and test a non IOV PCIe component. When you select the PCIe protocol for link creation, the Protocol Exerciser acts as a PCIe component (PCIe endpoint or Root complex) without the IOV capabilities. • SRIOV - Use this option to create a SRIOV (Single Root IO Virtualization) capable PCIe link between the Protocol Exerciser and DUT. The link is created as per the SRIOV specifications. Select this option when you want to stimulate and test a SRIOV capable PCIe component. When you select the SRIOV protocol for link creation, the Protocol Exerciser acts as a PCIe component (PCIe endpoint or Root complex) with SRIOV capabilities. You need to have the U4305A-023 or U4305U-023 SRIOV software license without which the SRIOV option is disabled. • **MRIOV** - Use this option to create an MR (Multi root) enabled PCIe link between the Protocol Exerciser and DUT. The MR enabled PCIe link is created as per the MRIOV Revision 1.0 Specifications. Select this option when you want to stimulate and test an MRIOV capable PCIe component. When you select the MRIOV protocol for link creation, the Protocol Exerciser acts as a PCIe component (PCIe endpoint or Root complex) with MRIOV capabilities. Based on the protocol that you select here, the settings and options are changed appropriately in other pages and tabs to support the selected protocol. For instance, when you select MRIOV, MRIOV capabilities are added for the base function Function A in the Configuration Space.
Supported Link Widths	Enables you to specify the desired link width to use for data transfer. Here, only those link widths, which are supported by Protocol Exerciser, are available for selection. The link widths, which are not supported by Protocol Exerciser, are disabled. The x1 check box is by default selected. This is because the support for x1 is mandatory as per PCI Express specification.
Link number advertised	Enables you to specify the link number that you want to advertise in training sequences. The link number specified here should be in the range of zero to 255.

Field	Description
Data Rate Capability	Enables you to the select the data rate capability of Protocol Exerciser. This data rate is advertised during the link training sequence to perform link speed negotiations. You can select a data rate capability from the following options: • 2.5 GT/s • 5.0 GT/s - Advertised data rate is then 2.5 and 5.0 GT/s • 8.0 GT/s - Advertised data rate is then 2.5, 5.0, and 8.0 GT/s This speed is not available if you choose to configure Exerciser to be MRIOV / SRIOV capable.
Autonomous speed change to 8 GT/s	Select this check box to enable autonomous speed change to the 8 GT/s speed. This ensures that Protocol Exerciser autonomously moves the link to 8.0 GT/s during the link training if the data rate capability of both the link partners is 8.0 GT/s. If this option is selected, Protocol Exerciser must perform equalization at the first entry to 8GT/s after exit from the Detect State. This option is applicable only when you are using Protocol Exerciser as a USC that is, the Session Type is configured as ToDownstream.
Scrambler	The **Scrambler** drop-down list provides the following options: • **Enabled**: Selecting this option enables or disables scrambling of data depending on the state of the Disable Scrambling bit of the link partner during link training. • **Disabled**: Selecting this option disables the scrambling of data and also sets the Disable Scrambling bit in the link control field of training sequences during link training.
Load Descrambler LFSR by SKIP OS	Enables or disables the loading of descrambler LFSR by incoming SKIP ordered sets.
Tag Mode	As a requester, Exerciser generates a tag (unique number) for each outstanding request that requires a completion to distinguish its completion from other requests. You can set the tag mode of Exerciser using the **Tag Mode** field. If you set the tag mode to **Normal**, the maximum number of outstanding requests per function are limited to 32, and a function can use only the lower 5 bits of the Tag field. If you set the tag mode to **Extended**, the maximum number of outstanding requests per function is increased to 255, and the entire Tag field (8 bits) is used.
PCIe Spec Revision	Select the PCIe specification revision with which you want the Exerciser hardware to be compliant. In this release, if you select the PCIe Spec Revision as r3.0_v0.7, then only the TS packets are 0.9 compliant.

Field	Description
Session Type	The **Session Type** drop-down list provides the following options: • **To Downstream**: Select this option when Protocol Exerciser emulates a root complex and you are testing a PCIe device at end point. The Protocol Exerciser then acts as an Upstream Component (USC) and performs the training sequences during its use as an LTSSM. • **To Upstream**: Select this option when Protocol Exerciser emulates an end point and you are testing a PCIe device at root-complex. The Protocol Exerciser then acts as a Downstream Component (DSC) and responds to the training sequences during its use as an LTSSM.
Enable ARI Capability	Select this checkbox to enable the Alternative Routing-ID Interpretation (ARI) capability for Protocol Exerciser. On enabling this capability, Protocol Exerciser acts as an ARI device and can support more than eight functions (with function number greater than seven). In situations where Exerciser provides more than eight functions and uses a function number greater than 7, you must select this checkbox. For instance, when Exerciser emulates a SRIOV capable component and has additional two physical functions license, then Exerciser provides 13 functions with function number 0 to 12. In such a situation, the checkbox must be selected to support these functions. If Protocol Exerciser is acting as a root complex and you enable this capability, the ARI forwarding is enabled for Protocol Exerciser to access the extended functions in an ARI device.

General Settings function - Pattern Matcher tab

To access this function: Click **General Settings** in the Navigation pane. Then, click the **Pattern Matcher** tab.

Refer to the topic Configuring Pattern Matchers on page 64 to know about how to configure pattern matchers and use these while sending stimulus and generating triggers.

Field	Description
Pattern Matcher	The Pattern Matcher drop-down list provides the following four pattern matchers: • Pattern Matcher 1 • Pattern Matcher 2 • Pattern Matcher 3 • Pattern Matcher 4
Enabled	By default, all the four pattern matchers are disabled. You can choose a pattern matcher and then select the Enabled checkbox to enable that particular pattern matcher. When you enable a pattern matcher, its fields (displayed in the Pattern Matcher tab) become available for editing.

Field	Description
Load Preset	The **Load Preset** listbox contains a list of available TLP templates from which you can select and load a TLP template based on which the pattern matcher is set. Clicking Load Preset displays a drop-down listbox, with a list of predefined TLP templates. Select a desired TLP template from the list to load it in the Pattern Matcher tab.
Reset	Click **Reset** to reset the field values of the pattern matcher to default values.
Field	The **Field** column contains the names of the fields of a packet. The number of fields displayed in this column may differ for different types of packets.
Value	The **Value** column contains an input component for each field. You can specify the desired inputs in these components to edit the selected packet. This column also contains a drop-down list and a calculator for input assistance. The drop-down list enables you to specify your inputs in hexadecimal, binary, or decimal format. Clicking the calculator for input assistance displays the Input Assistant dialog box. Here, you can perform boolean operations over the hexadecimal, binary, and decimal values, and use the resulting value as the new value for the corresponding packet field.
Length	The **Length** column displays the maximum length of a field in bits, bytes, or DWords.

General Settings function - Power Management tab

To access this function: Click **General Settings** in the Navigation pane. Then, click the **Power Management** tab.

To know how to configure the L0s and L1 states for Exerciser, refer to the topic Configuring Power Management Settings on page 71.

Field	Description
L0s	The L0s section provides the following components: • **Enter L0s after having seen no TLP/DLLP for (ns)**: This text box allows Exerciser (Tx) to enter L0s after the specified nanoseconds, during which there were no TLP/DLLP packets. This text box accepts values in multiple of 16, such as 16, 32, 48, and so on. If you type a value, which is not a multiple of 16, then that value is rounded to the closest multiple of 16 when you click Apply. For example, 20 is rounded to 16. Similarly, 30 is rounded to 32. This maximum value of this field is 268435440. • **Exit L0s when the following timeout (ns) expires**: Select this check box to instruct Exerciser (Tx) to exit from L0s after the specified nanoseconds. The text box adjacent to it accepts values in multiple of 16, such as 16, 32, 48 and so on. If you type a value, which is not a multiple of 16, then that value is rounded to the closest multiple of 16 on clicking Apply. For example, 20 is rounded to 16. Similarly, 30 is rounded to 32. This maximum value of this field is 268435440. • **On exit send the following number of FTS**: Select this check box to send the specified number of FTS Ordered Sets when Exerciser (Tx) exits from L0s. This maximum value of this field is 16383. • **On exit send SKP Ordered Sets**: Select this check box to send SKP Ordered Sets, after FTS Ordered Sets, when Exerciser (Tx) exits from L0s.
L1	The L1 section provides the following components: • **Enter ASPM L1 after having seen no TLP/DLLP for (ns)**: This text box allows Exerciser (Tx) to enter ASPM L1 after the specified nanoseconds, during which there were no TLP/DLLP packets. This text box accepts values in multiple of 16, such as 16, 32, 48, and so on. If you type a value, which is not a multiple of 16, then that value is rounded to the closest multiple of 16 when you click Apply. For example, 20 is rounded to 16. Similarly, 30 is rounded to 32. This maximum value of this field is 268435440. • **Exit L1 (ASPM or PCI-PM) when the following timeout (ns) expires**: Select this check box to instruct Exerciser (Tx) to exit from L1 after the specified nanoseconds.The text box adjacent to it accepts values in multiple of 16, such as 16, 32, 48, and so on. If you type a value, which is not a multiple of 16, then that value is rounded to the closest multiple of 16 on clicking Apply. For example, 20 is rounded to 16. Similarly, 30 is rounded to 32. This maximum value of this field is 268435440. • **Enable aggressive mode for PCI-PM L1**: Select this check box to enable aggressive PCI-PM L1 power management mode. In this mode, when Exerciser is in L1 and receives a packet from the USC, it responds to the received packet and then falls back into L1. In passive mode, if the upstream component sends a packet, Exerciser responds to it and if the device state is non D0, will not go to L1 again.

Field	Description
TS Settings	The TS Settings section provides the following components: • **N_FTS shown in TS at Gen1 speed**: This text box allows you to modify the N_FTS field of Training Sequences that Protocol Exerciser sends at Gen1 speed. The maximum value of this field is 255. • **N_FTS shown in TS at Gen2 speed:** This text box allows you to modify the N_FTS field of Training Sequences that Protocol Exerciser sends at Gen2 speed. The maximum value of this field is 255. • **N_FTS shown in TS at Gen3 speed:** This text box allows you to modify the N_FTS field of Training Sequences that Protocol Exerciser sends at Gen3 speed. The maximum value of this field is 255.

General Settings Function Symbol Settings tab

To access this function: Click **General Settings** in the Navigation pane. Then, click the **Symbol Settings** tab.

Field	Description
2.5 GT/s and 5.0 GT/s	
SKP Interval (in number of symbols)	Enables you to specify the symbol time gap between two SKP ordered sets.
Number of SKP symbols in SKP ordered sets	Provides options to specify the number of SKP symbols, ranging between 1 to 5, to be included in an ordered set.
8.0 GT/s	
SKP Interval (in number of blocks)	Enables you to specify the block time gap between two SKP ordered sets.
Number of SKP symbols in SKP ordered sets	Provides options to specify the number of SKP symbols, ranging between 8 to 24, to be included in an ordered set.

General Settings Function Transceiver Settings tab

To access this function: Click **General Settings** in the Navigation pane. Then, click the **Transceiver Settings** tab.

These settings control the Protocol Exerciser Tx and Rx equalization settings.

If you select the MRIOV or SRIOV protocol for creating a link between Protocol Exerciser and DUT, then the Transceiver settings that you specify in this tab do not have any impact.

Field	Description
Tx Settings	In the TX Settings group box, you can configure the Transceiver settings for the Protocol Exerciser's transmitter separately for Gen1, Gen2, and Gen3 speeds.
2.5 GT/s	This group box is used to set the Transceiver settings that are applicable at the Gen1 speed. In this group box, you can: • either enable and set the de-emphasis level at -3.5 db (applicable when Exerciser transmits at Gen1 speed). • or disable the de-emphasis at Gen1 speed.
5.0 GT/s	This group box is used to set the Transceiver settings that are applicable at the Gen2 speed. In this group box, you can: • either enable and set the de-emphasis level at -3.5 db or -6 db (applicable when Exerciser transmits at Gen2 speed). • or disable the de-emphasis at Gen2 speed.
8.0 GT/s	This group box is used to set the Transceiver settings that are applicable at the Gen3 speed. The following fields are available in this group box.
Tx output voltage	This setting controls the maximum Vout before any pre-emphasis or de-emphasis (Vd). The default value is 900 mV.
Tx Pre-emphasis Pre Tap	This setting controls the amount of pre-emphasis (Vc). As you move towards a more negative value, you get more Pre-emphasis.
Tx Pre-emphasis First Post Tap	Main De-emphasis value, which controls the amount of de-emphasis (Va). The higher the number, the higher is the de-emphasis.
Tx Pre-emphasis Second Post Tap	Second post-tap De-emphasis.
Rx Settings	In the RX Settings group box, you can configure the Transceiver settings for the Protocol Exerciser's Receiver. These settings are applicable for all the three speeds, Gen1, Gen2, and Gen3.
Rx Equalizer AC and DC Gain	These settings impact the Receiver equalization of the Protocol Exerciser. You can tune the Rx equalization by adjusting the AC and DC gain settings. Typically DC gain would be 0 or 3db gain.

A table displayed at the bottom of the page provides the recommended Transceiver settings for Exerciser.

Refer to the topic Configuring Transceiver Settings on page 61 to know more.

General Settings Function - Trigger Out tab

To access this function: Click **General Settings** in the Navigation pane. Then, click the **Trigger Out** tab.

Field	Description
Status	The **Status** section shows the following messages: • **Accumulated events**: Shows a list of events that resulted in generating the trigger-out pulse since the last time you clicked Clear Status. • **First event**: Shows which first events, which occurred in the same timestamp, resulted in generating the trigger-out pulse since the last time you clicked Clear Status.
Source Name	The **Source Name** column displays a list of events, such as Protocol Error and Going Out of L0, which you can use as a source to generate a trigger-out pulse.
Enabled	The **Enabled** column provides a check box for each event in the Source Name column. Here, select a the check box corresponding to a event that you want to use as a source to generate a trigger-out pulse.
Fired	The **Fired** column shows the following messages for each event listed under the Source Name column: • **Fired**: This message appears when the corresponding event has occurred. • **Not fired**: This message appears when the corresponding event has not occurred.
Mask All	Click **Mask All** to select all check boxes in the **Enabled** column. This, as a consequence, instructs Protocol Exerciser to trigger-out whenever one or more of the events listed under the **Source Name** column occur.
Unmask All	Click **Unmask All** to clear all check boxes in the Enabled column.

Field	Description
Clear Status	Click **Clear Status** to set the status of all events to Not fired.
Refresh Status	Click **Refresh Status** to refresh the status of all the events. You can view whether or not an enabled rule has fired and display the list of accumulated events in this tab by clicking the Refresh Status button.

LTSSM Tests

To access this page: Click the **LTSSM** Test icon in the Navigation window.

Field	Description
Select Test Case	The **Select Test Case** pane has the following components: • **Test Case:** The Test Case list box provides a list of LTSSM test cases that you can select to run. • **Description:** The Description list box displays the description of the selected test case. A test case description is divided into the following heads: • **Purpose**: This head states what the test case intends to do. • **Prerequisites**: This head states what should already be in place before you run the test case. • **Observation**: This head states the checks that are performed during the execution of a test case.
Sequence Overview	The **Sequence Overview** section displays a flow-chart diagram to show the execution sequence of the selected test case.
Run	Click **Run** to start executing the selected test case.
Stop	Click **Stop** to halt the execution of the underlying test case.

Field	Description
Test Case State	The Test Case State section displays the following states that an executing test case goes through: • **Running**: Reflects that the test case is executing. • **Passed**: Reflects that the test case is successfully executed. • **Failed**: Reflects that executing test case is either cancelled by user or has met an unwarranted condition. You can cancel an executing test case by clicking the Stop button.
Log	The **Log** pane displays the data describing the various stages of the underlying test case. The Log pane has its own short-cut menu, which is displayed when you right-click anywhere inside it. This short-cut menu has the following commands: • **Clear Log**: Clears the Log pane. • **Copy to Clipboard:** Copies the data in the Log pane to clipboard. • **Save**: Displays the Save log file dialog box, which you can use to save the data shown in the Log pane to a file in RTF or TXT format.

Protocol Checker function

To access this function: Click **Protocol Checker** in the Navigation pane.

Refer to the topic Performing Protocol Checks to know more about protocol checks supported by Exerciser.

Field	Description
Status	The Status section shows the following components: • **Violated Rules**: Shows a list of PCIe protocol rules that were violated since the last time you clicked **Clear Status**. • **First Violated**: Shows which first rules of the PCIe protocol were violated in the same timestamp since the last time you clicked **Clear Status**.
Rule Name	The **Rule Name** column displays a list of PCIe protocol rules, which you can select to observe for violation.

Field	Description
Enabled	The **Enabled** column provides a check box for each rule in the Rule Name column. Here, select a the check box corresponding to a rule that you want to observe for violation.
Violation	The **Violation** column shows the following messages for each rule listed under the Rule Name column: • **Violated**: This message appears when the corresponding rule has been violated. • **Not Violated**: This message appears when the corresponding rule has not been violated.
Mask All	Click **Mask All** to select all check boxes in the Enabled column. Use this command button when you want to enable all the rules listed under the Rule Name column.
Unmask All	Click **Unmask All** to clear all check boxes in the Enabled column.
Clear Status	Click **Clear Status** to set the status of all events to Not violated.
Refresh Status	Click **Refresh Status** to refresh the status of all the rules listed under the Rule Name column.

Traffic Setup Function - Completion Behaviors

To access this function: Click **Traffic Setup** in the Navigation pane. Then select the **Completion Behaviors** tab.

You use this tab to add records to define the behavior of completion packets to be sent as response to DUT. You can add these records in one of the completion queues provided by Protocol Exerciser.

The completion behavior records are added to the memory allocated to the queue selected for adding the record.

Refer to the topic Sending Completions as Response to DUT_ to know more.

Field	Description
Settings (for a PCIe/SRIOV link)	If you selected PCIe or SRIOV protocol for link creation in the Link Settings page, then the Settings pane displays two completion queues, Queue 0 and Queue 1. Protocol Exerciser provides one completion queue for each virtual channel (VC0 and VC1). You can add completion behavior records to each of these queues for these virtual channels. Refer to the topic Completion Queues to know more.
Settings (for an MRIOV enabled PCIe link)	If you selected the MRIOV protocol for link creation in the Link Settings page, then the Settings pane displays three completion queues, Queue 0 to Queue 2. Protocol Exerciser provides one completion queue for each virtual hierarchy (VH0 to VH2) that it supports as an MRIOV capable component. You can add completion behavior records to each of these queues for these virtual hierarchies. Refer to the topic Completion Queues to know more. If you have the additional 2 physical functions license, then the Settings pane displays five completion queues, Queue 0 to Queue 4. Protocol Exerciser provides one completion queue for each virtual hierarchy (VH0 to VH4) that it supports with this license. All virtual hierarchies are mapped to virtual channel (VC0) and virtual link (VL0).
Completion Behaviors	
Line no.	Displays a sequenced number for a completion behavior record that you added to the memory allocated for the selected completion queue. You can add a maximum of 512 completion behavior records in a completion queue.
Error	Displays the details of the completion behavior record that you added to the memory allocated for the selected completion queue. To add a completion behavior record, the Templates pane provides the Default Completion template. To add this template to the Completion Behavior tab, double-click it or drag-and-drop it in the Completion Behavior tab. You can double-click the details of the added behavior record to change the behavior of the completions to be sent. While changing the behavior of a completion packet, you need to set some fields in conjunction with the settings done in the Error Insertion function. This is because, the new settings that you specify in the Error Insertion function become effective only when you have enabled their corresponding behavior properties in the completion behavior record.

Traffic Setup Function - Block Transfers

To access this function: Click **Traffic Setup** in the Navigation pane. Then, select the **Send Block Transfers** tab.

You use this tab to define a block of PCIe packets to be sent as stimulus in the defined sequence. You add packets to a function of Exerciser and also define the behavior for these packets. To know more, refer to the topic Defining Stimulus Traffic.

Field	Description
Settings	This pane displays the base, physical and virtual functions of the Protocol Exerciser emulating a multi-function PCIe component. Each function corresponds to a hardware channel of Protocol Exerciser. You can define stimulus traffic for each of these functions. The number and type of functions displayed depend on the emulation mode of Protocol Exerciser as an MRIOV capable, SRIOV capable, or a non IOV PCIe component. • If you select MRIOV as the protocol while creating the PCIe link, then Protocol Exerciser has MRIOV capabilities and it displays the following functions within the two virtual hierarchies that it supports. • Function A - Base function (BF0) with MRIOV capabilities • Function B and Function C - Physical functions (PF0 and PF1) with SRIOV capabilities • Virtual functions (VF1 and VF2) with each physical function. • If you select SRIOV as the protocol while creating the PCIe link, then Protocol Exerciser has SRIOV capabilities and it displays the following functions. • Function A - Non IOV function • Function B and Function C - Physical functions (PF0 and PF1) with SRIOV capabilities • Virtual functions (VF1 and VF2) with each physical function. • If you select PCIe as the protocol while creating the PCIe link, then Protocol Exerciser does not have MRIOV capabilities and it displays the following three functions with no IOV capabilities. • Function A • Function B • Function C If you have the additional 2 physical function license, then five functions (Function A to E) are displayed in each emulation mode along with the virtual functions in SRIOV and MRIOV modes. Refer to the topic Functions, Hardware Channels, Virtual Hierarchies, and Virtual Channels to know more about the virtual hierarchies, functions, corresponding hardware channels and virtual channels of Exerciser.
Send Block Transfers	This tab contains the PCIe packets to be sent as a block to DUT. These PCIe packets are added as records in the memory of Exerciser and assigned a sequenced Line number.

Field	Description
Run Mode	The Run Mode category has the following options: • **Single**: Select this option if you want to send a block transfer of all packets in the Send Block Transfers tab for each time you select **Action** > **Run**. In this case, block transfer automatically stops after all packets are sent. • **Continuous**: Select this option if you want to send the block of packets added to the function repeatedly until you manually stop the stimulus. In this case, you have to select **Action** > **Stop** to stop the continuous stimulus.
Templates	Provides a list of templates for PCIe packets that you can add to the Send Block Transfers tab as stimulus.
Request Behavior	This tab is used to define the behavior of the PCIe packets that you added in the Send Block Transfers tab. The behavior settings are added as behavior records in the memory.
Line No	Displays the sequenced number for the request behavior records that you added for the PCIe packets in the Send Block Transfers tab.
Errors	Displays the details of the request behavior record that you added for the PCIe packets in the Send Block Transfers tab. These details define the behavior of the PCIe packets to be sent as stimulus. Here, only the first behavior record is added automatically at the time of adding the first PCIe packet in the Send Block Transfers tab. Beyond that, you can manually add more records using the available behavior templates. If there is only one behavior record, then all packets added to a function are sent matching the behavior specified in this record. If there are the same number of packets and behavior records, for example 2 packets and 2 records, then there is one-to-one mapping between the packets and records: If there are unequal numbers of packets and records, for example 5 packets and 3 records, then mapping between these takes place as follows: • Packet 1 is as per behavior record 1 • Packet 2 is as per behavior record 2 • Packet 3 is as per behavior record 3 • Packet 4 is as per behavior record 1 • Packet 5 is as per behavior record 2
Templates	The Templates pane provides templates for defining the behavior of the PCIe packets that you added in the Send Block Transfer tab. To add a request behavior record using one of these templates, double-click or drag-and-drop the desired template.

Traffic Setup Function - Single Packet

To access this function: Click **Traffic Setup** in the Navigation pane. Then, select the **Single Packet** tab. You use this tab to send a single PCIe packet as stimulus to DUT and view the completion received from DUT for this packet.

To know more about sending single packets as stimulus, refer to the topic Sending a Single Packet as Stimulus and Viewing Response.

Field	Description
Settings	This pane displays the base, physical and virtual functions of the Protocol Exerciser emulating a multi-function PCIe component. Each function corresponds to a hardware channel of Protocol Exerciser. You can define stimulus traffic for each of these functions. The number and type of functions displayed depend on the emulation mode of Protocol Exerciser as an MRIOV capable, SRIOV capable, or a non IOV PCIe component. • If you select MRIOV as the protocol while creating the PCIe link, then Protocol Exerciser has MRIOV capabilities and it displays the following functions within the two virtual hierarchies that it supports. • Function A - Base function (BF0) with MRIOV capabilities • Function B and Function C - Physical functions (PF0 and PF1) with SRIOV capabilities • Virtual functions (VF1 and VF2) with each physical function. • If you select SRIOV as the protocol while creating the PCIe link, then Protocol Exerciser has SRIOV capabilities and it displays the following functions. • Function A - Non IOV function • Function B and Function C - Physical functions (PF0 and PF1) with SRIOV capabilities • Virtual functions (VF1 and VF2) with each physical function. • If you select PCIe as the protocol while creating the PCIe link, then Protocol Exerciser does not have MRIOV capabilities and it displays the following three plain functions with no IOV capabilities. • Function A • Function B • Function C If you have the additional 2 physical function license, then five functions (Function A to E) are displayed in each emulation mode along with the virtual functions in SRIOV and MRIOV modes. Out of these functions, you can enable/activate one function at a time by selecting the checkbox displayed with the function. The hardware channel corresponding to this function is used to send the packet added to this enabled function. Protocol Exerciser does not send the packets that you add to a disabled function as stimulus.
Send Single Packet	The Send Single Packet tab displays the PCIe packets that you added from the PCIe templates list to send these packets as stimulus to DUT. You can select a packet from this list and send it as a single packet to DUT.
Templates	Displays a list of PCIe packet templates that you can use to define the single packet stimulus traffic. You can drag and drop a template to the Send Single Packet tab to create an instance of that template. The added instance is sent as a PCIe packet to DUT on running the stimulus.

Field	Description
Request Behavior	The Request Behavior tab contains a behavior record automatically added for each PCIe packet in the Send Single Packet tab. You can change the default behavior of a PCIe packet by editing its behavior record displayed in this tab. Each behavior record is mapped to its packet using the Line number assigned to it. You cannot add or delete these records. They are automatically added/removed at the time of adding/removing the corresponding PCIe packet.
Received Completions	The Received Completions tab displays the completion packets received from DUT in response to the Read packets, such as Memory Read, I/O Read, and Config Read, which you sent as single packet stimulus using Protocol Exerciser.
Completions to hold	The **Completions to hold** text box enables you to specify the number of completion packets to be displayed in the Received Completions tab. In this text box, you can specify a value ranging from 1 to 128.

Virtual Channel function

To access this function: Click **Virtual Channel** in the Navigation pane.

You use this page to configure the flow control initialization settings for the virtual channels supported by Protocol Exerciser. You can configure the settings such as flow control credit limits, enabling a virtual resource or a virtual hierarchy, and mapping traffic classes to a virtual resource for a function.

To know more about virtual channels of Protocol Exerciser and flow control initialization, refer to the topic Initializing Flow Control.

Field	Description
Advertised Credits	You use this group box to define the flow control credit limits for the virtual channels VC0 and VCx and virtual hierarchies supported by Protocol Exerciser. If Exerciser emulates a non IOV/SRIOV component, then you can specify credit limits for VC0 and VCx virtual channels. If Exerciser emulates a MRIOV component, then you can specify credit limits for virtual hierarchies. All these hierarchies map to VC0 and VL0.
Header	For each virtual channel/virtual hierarchy, select the credit limit that you want to allocate to the TLP headers sent by Exerciser. This credit limit for headers is categorized into posted request headers, non posted request headers, and completion headers. You can allocate 'Unlimited' credit limit to each of these header types. However, at x16 Gen 3 link speed, you can allocate a total of 256 credit limit to these header types for both the virtual channels of Protocol Exerciser.

Field	Description
Data	For each virtual channel/virtual hierarchy, select the credit limit that you want to allocate to the TLP data payloads sent by Exerciser. This credit limit for payloads is categorized into posted request payloads, non posted request payloads, and completion payloads. You can allocate 'Unlimited' credit limit to each of these payload types. However, at x16 Gen 3 link speed, you can allocate a total of 4096 credit limit to these payload types for both the virtual channels of Protocol Exerciser.
Resend / Update periods	
InitFC Resend Period (ns)	Specify the time period (in ns) for which you want Protocol Exerciser to wait before resending the InitFC DLLPs to initialize: • VC_Resource 1 (VCx) if Exerciser emulates a non IOV /SRIOV component. • additional virtual hierarchies if Exerciser emulates an MRIOV capable component.
Send FC Update each (us)	Specify the time period (in us) after which Exerciser should start sending UpdateFC packets for the virtual channels to DUT.
Trigger 'FC Update timeout' after (us)	Specify the time interval after which Exerciser should trigger a Flow Control Update timeout if it does not receive a FC update packet during that time period. When the Flow Control Update Timeout is triggered, a PCIe protocol violation flag is raised and the FC Update Timeout rule in the Protocol Checker function shows Violated.
PCIe/SRIOV	Displays the flow control settings applicable for a non IOV/SRIOV capable Exerciser. These settings are enabled when you select PCIe or SRIOV as the protocol in the **General Settings** -> **Link settings** page.
VC Resource 1 VC id (x)	Select a number to assign an ID to the second virtual channel VC_Resource_1 of Protocol Exerciser. This number is then used by Protocol Exerciser for this second virtual channel. The number of the virtual channel VC_Resource_0 of Protocol Exerciser is fixed (VC0) and cannot be changed.
Enable VC Resource 1	Protocol Exerciser provides two virtual channels VC Resource 0 (VC0) and VC Resource 1 (VCx). VC0 is the default virtual channel and is enabled by default. However, VC Resource 1 is not enabled by default. If required, you can choose to enable this virtual channel by selecting this checkbox. This is applicable only when Protocol Exerciser is emulating a non IOV or SRIOV capable PCIe component.
MRIOV	Displays the flow control settings applicable for an MRIOV capable Exerciser. These settings are enabled when you select MRIOV as the protocol in the General Settings -> Link settings page.
Enable Virtual Hierarchy VH1 - VH4	Protocol Exerciser supports five virtual hierarchies VH0 to VH4 in its emulation as an MRIOV capable component. VH0 is enabled by default. If required, you can choose to enable VH1 to VH4 hierarchies as well by selecting the checkboxes for these hierarchies. If you do not enable a hierarchy, then a packet added to a function which is a part of that hierarchy is not transmitted. This checkboxes are enabled only when Protocol Exerciser is emulating an MRIOV capable PCIe component.

Field	Description
Enable per VH Flow Control	As an MRIOV capable component, Protocol Exerciser supports two types of flow controls, per VH flow control and per VL flow control. If you select this checkbox, then the per VH flow control is enabled at the Exerciser end and requested during the flow control initialization process. If you do not select this checkbox, then the per VL flow control is applicable and requested during the flow control initialization process. To know more about the two flow controls and how Exerciser uses these, refer to the topics per VH flow control and VL based flow control. This checkbox is enabled only when Protocol Exerciser is emulating an MRIOV capable PCIe component.
TC to VC Map	This group box allows you to map the Traffic classes (from 0 to 7) with the virtual channels VC0 and VCx for the three functions supported by Exerciser as a non IOV/SRIOV component. The TC to VC mapping is done separately for each of the three functions of Exerciser. The TC to VC Map group boxes are enabled only when you select the PCIe or SRIOV protocol in the Link Settings page. Exerciser decides the virtual channel (VC0 or VCx) to be used to pass a stimulus request added to a function based on the TC to VC Map settings that you specify for that function. For instance, if you mapped traffic class 2 with VC0 for Function A, then a stimulus packet added to Function A and with traffic class 2 will be sent through VC0. If you have the additional 2 physical function license, then the TC to VC mapping is displayed for the five functions supported by Exerciser.

A
Updating the Firmware

Using the Firmware Update Tool 242

This appendix provides information on the Firmware Update tool.

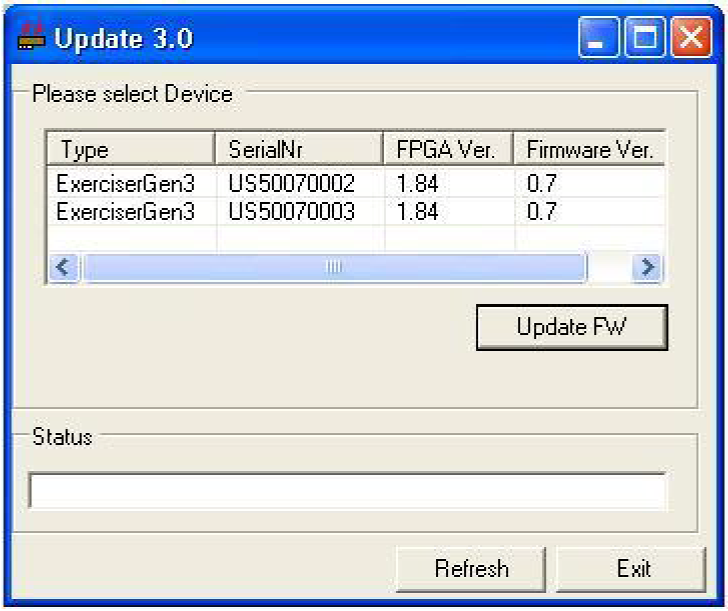
Using the Firmware Update Tool

The Firmware Update tool is automatically installed during the Protocol Exerciser installation. You use this tool to update the Controller PC with the new firmware for the selected Exerciser card.

To access the Firmware Update tool:

Select **Start** > **Programs** > **Agilent SPT** > **PCIe Exerciser 8.2 Release** -> **Update GUI**.

The **Firmware Update** window appears.



Type	SerialNr	FPGA Ver.	Firmware Ver.
ExerciserGen3	US50070002	1.84	0.7
ExerciserGen3	US50070003	1.84	0.7

Figure 32 *Firmware Update window*

NOTE The cards shown with a grey background indicate that these are currently in use and cannot be updated. The cards displayed in red indicate that these cards need an update.

To Update the Firmware Version of the Exerciser Card

1 Select the Exerciser card, for which you want to update the Firmware, in the **Please Select Device** section.

2 Click **Update FW**.

This instructs the Controller PC to check for the new Firmware version for the selected exerciser card. If there is any new version, the Controller PC uses it to update its existing Firmware version.

Once the Controller PC is updated with the new firmware, the Status field in the Firmware Update tool shows Done.

Index

A

AC and DC Gain, 61
address decoder, 118
aggressive PCI-PM L1 power management mode, 226
ARI Capability, 224
ARI device, 224
ASPM L1, 226
Auto-Hide a Floating Window, 46
Autonomous equalization, 57
Autonomous speed change to 8 GT/s, 53, 223

B

backplane, 26
Bad Nullified TLP or LCRC Error Received, 171
BAR, 118
Base Address Register, 122
Base Address Registers, 118
behavior records, 139

C

Capture State Transitions Test, 114
Change a Docked Window to a Floating Window, 46
change the link speed, 85
co-efficient values, 60
completion, 152
Completion Behaviors, 157
completion timeout, 163
completions, 122, 157
components of the Protocol Exerciser GUI, 38
Configuration Space, 116
Configuring Equalization settings, 59
controller PC, 30
Creating a new Protocol Exerciser session, 32

D

data comparison, 126
Data memory, 122
data memory, 123
data scrambling, 53
decoder, 118
Decoder Miss, 171
delay completion, 163
delay packet, 142
Disabling Equalization, 59
Discard completion, 163
Disparity, 162
Disparity Error Received, 171
Display a Floating Window, 46
DLLP Received with CRC Error, 171
Dock a Floating Window, 47
DSC, 17, 90

E

ECRC, 162, 172
ECRC error detected, 174
Enabling Equalization, 58
endpoint, 17
equalization procedure, 57
Equalization Settings, 56
Equalization Settings tab, 219
error conditions, 162
exerciser emulation modes, 16
Exerciser starts Link Training to 2.5 GT/s, 102
Exerciser starts Link Training to 8.0 GT/s with Equalization Tests, 104
Expansion ROM, 118

F

FC Update Timeout, 172
flow control credit, 43
flow control credits, 79
flow control initialization, 79
FS, 60

H

Hardware, 25
Hardware channels, 130
Hide a Floating Window, 46

I

Ignore coefficient rule checks, 220
incorrect ECRC, 175
initialize a PCIe link, 51
internal memory base address, 119

L

L0s, 226
L1, 226
Lane Deskew Error, 171
Lane Polarity Inversion, 55
Lane Reversal, 55
Lane Settings tab, 221
LF, 60
Link downgrade speed tests, 108
Link settings tab, 51
Link Speed Change With Equalization Tests, 111
Link Speed Change Without Equalization Tests, 107
link status, 40
Link Training and Status State Machine, 88
Link Up, 77
Link upgrade speed tests, 108
LTSSM, 88
LTSSM States, 88, 89
LTSSM states, 94
LTSSM test prerequisites, 94
LTSSM Test Results, 100
LTSSM Tester, 90
LTSSM tests, 94
LTSSM Tests Sequence, 100

M

Memory Compare, 43
MR enabled link, 17
MR Enabled PCIe link, 52
MRA, 17
MRA PCIe component, 17
MRIOV, 222
MRIOV capabilities, 52
MRIOV Capability block, 116
MRIOV functions, 23
Multi Root Aware, 17

N

NAK Sent, 171
Non IOV functions, 22
non IOV PCIe end point, 17
non IOV PCIe root complex, 17
Nullified TLP, 163
Nullified TLP Received, 171

O

Offline, 30
Offset Sequence Number, 163
Online, 30
Out of Sequence TLP, 171

P

Payload, 142
payload, 122
PCIe link, 51
PCIe link with SRIOV capabilities, 51
PCIe packet templates, 138
PCI-PM L1, 226
per VH flow control, 80
Perform Equalization in Recovery @8 GT/s, 221
Perform Phase 2/Phase 3 option, 58
phases of equalization, 57
Poisoned TLP, 163
Port Selection, 33, 200
Protocol Checker, 169
protocol checks, 169, 171
protocol errors, 162
Protocol Exerciser API, 26
Protocol Exerciser GUI, 26
Protocol Exerciser hardware status, 40

Protocol Exerciser session, 30
Protocol Exerciser setup file, 50
Protocol Test Card, 180
PTC, 180

R

Recovery.Equalization sub state, 104
Reload data memory, 125
repeat packet, 141
Replace END, 163
Replace STP, 163
request behavior record, 140
Request Equalization in Recovery @8 GT/s, 221
retrain link, 84
root complex, 17

S

Scrambler (2.5 GT/s and 5.0 GT/s), 223
Select type of connection, 203
Select type of connection dialog box, 31
Session list, 32
session type, 52
Setting Equalization values, 59
setup files, 26
Single Packet memory, 153
single packet requests, 153
single packet stimulus, 152
Software based equalization, 58
SRIOV, 51, 222
SRIOV Capability block, 116
SRIOV capable PCIe end point, 17
SRIOV functions, 23
SRIOV software license, 51

T

Table 42, 185
Table 43, 185
TC VC mapping, 80
test log, 94
Testing Link Retraining and Recovery, 113
Testing Link Training, 102
Testing LTSSM State Transitions, 114
TLP requests, 138
traffic class, 80
Training a PCIe Link, 77
transceiver settings, 61
Transceiver Settings tab, 227
trigger, 69
trigger out conditions, 69
Trigger Out Connector, 69
Trigger out Connector component, 169
Trigger Out pulse, 169
Tx de-emphasis, 56
Tx De-emphasis preset, 60
types of equalization, 57

U

U4305A Exerciser Card, 25
Unsupported Request Received, 171
USC, 17, 90
Using an existing Protocol Exerciser session, 32

V

VC0, 135
Verifying PCIe Link Training, 77
VF BARs, 119
violated rule, 169
virtual channel, 79, 135
Virtual channels, 131, 135
virtual channels, 79
Virtual Hierarchies, 134
virtual hierarchies, 80, 134
virtual hierarchy, 79
virtual link, 80, 135
VL based flow control, 81
VL0, 135

W

working of LTSSM Tester, 94
wrong LCRC, 162

B
PTC Tests and Assertions

Tests and Assertions 249

This appendix provides a very brief information on the tests available in PTC and their brief description.

Tests and Assertions

Table 4 lists and briefly describes the available PTC tests and assertions:

Table 4 PTC Tests and Assertions

Test	Intent	Assertion	Description
ASPM_L1	To verify that the DUT correctly requests ASPM L1 entry when it wants to enter ASPM L1 state.	ASPM_L1	DUT must correctly request ASPM L1 entry when it wants to enter ASPM L1 state for all supported data rates.
BADECRC	To verify correct ECRC generation and ECRC checking behavior of DUT.	BADECRC	DUT must correctly generate and check ECRC and report an error in case of incorrect ECRC.
DLL_04_01_02	To check that receiver ignores the reserved fields of the received DLLPs. (ReservedFieldsDLLPReceive).	DLL.04.01#02	Upon receiving a DLLP, the Data Link Layer receiver state machine must ignore reserved fields.
DLL_05_02_01	To check that DUT will retransmit a transaction for which a NAK has been issued. (ReTransmitOnNak).	DLL.05.02#010	The link transmitter must replay a transaction on receiving a NAK.
DLL_05_02_011	To check that the DUT’s REPLAY_TIMER is working properly by not sending either an ACK or a NAK (ReplayTimerTest).	DLL.05.02#011	The link transmitter must time out with a value defined in REPLAY_TIMER when it does not receive an ACK or a NAK, and the transaction is retransmitted.
DLL_05_02_012	To ensure that the DUT will keep retransmitting a transaction for which a NAK has been issued on purpose until the number of times its REPLAY_NUM supports.(ReplayNumTest).	DLL.05.02#012	The link transmitter must retransmit a transaction REPLAY_NUM of times if it keeps on receiving NAK.
DLL_05_02_02	To check that if REPLAY_NUM overflows, link retraining is triggered (LinkRetrainOnRetryFail).	DLL.05.02#02	If repeated retries fail and REPLAY_NUM overflows, the link transmitter must ask the Physical Layer to retrain the link. There must be a reported error to correspond to this as per Section 6.2.
DLL_05_02_07	To check that the retry buffer does not changes in link retraining.	No assertion	The link transmitter must retransmit the same transaction after link retraining after REPLAY_NUM overflow.

Table 4 PTC Tests and Assertions

Test	Intent	Assertion	Description
DLL_05_02_10	To check that the oldest unacknowledged TLP is sent first. (ReplayTLPOrder)	DLL.05.02#10	Oldest unacknowledged TLP must be sent first in replay and followed by the rest of the unacknowledged TLPs in the original transmission order.
DLL_05_02_15	To check that corrupt DLLPs are discarded (Corrupted DLLPs).	DLL.05.02#15	All corrupt DLLPs must be discarded and be reported as an error associated with the port.
DLL_05_02_16	To check that a DLLP with undefined encodings is dropped silently (UndefinedDLLPEncoding).	DLL.05.02#16	A DLLP with undefined encodings shall be dropped silently by the receiver with no error associated.
DLL_05_02_17	To check that an Ack with unknown sequence number is reported as FATAL_ERROR (WrongSeqNumInAckDLLP).	DLL.05.02#17	If an Ack DLLP doesn't have a sequence number of an unacknowledged TLP, or of the most recently acknowledged TLP, it must be reported as a DL Layer protocol error associated with the port.
DLL_05_03_02	To check for wrong LCRC detection (BadLCRC).	DLL.05.03#02	If a normal TLP (one with END framing symbol) is received and its LCRC doesn't match calculated CRC, discard the TLP, free any storage associated with it, schedule a Nak DLLP for transmission if one is not already scheduled and report an error associated with the Port.
DLL_05_03_03	To check that a TLP with wrong sequence number is discarded, and any associated storage is freed (DuplicateTLPSeqNum).	DLL.05.03#03	If the sequence number doesn't match with expected value, discard the TLP and free any storage associated with it.
EQUALIZATION_COEFFICIENTS	To check that the DUT correctly responds to link equalization requests to adjust coefficients.	EQUALIZATION_COEFFICIENTS	The DUT must correctly respond to link equalization requests to adjust TX EQ coefficients for any legal requests following legal timings.
EQUALIZATION_INITIAL_PRESET	To check that the DUT correctly responds to link equalization requests to adjust initial presets.	EQUALIZATION_INITIAL_PRESET	The DUT must correctly respond to link equalization requests to adjust TX EQ initial presets for any legal requests following legal timings
EQUALIZATION_PRESETS	To check that the DUT correctly responds to link equalization requests to adjust presets.	EQUALIZATION_PRESETS	The DUT must correctly respond to link equalization requests to adjust TX EQ presets for any legal requests following legal timings

Table 4 PTC Tests and Assertions

Test	Intent	Assertion	Description
FLR_CHECK_RESET	To verify that FLR does not impact the DUT's link states.	FLR_CHECK_RESET	FLR must not impact the DUT's link states.
FLR_DUT_BEHAVIOR	To verify that DUT correctly performs FLR for all supported data rates.	FLR_DUT_BEHAVIOR	DUT must correctly perform FLR for all supported data rates.
L1_FOR_D3_STATE	To verify that the DUT correctly requests L1 entry when software sets the DUT state to D3.	L1_FOR_D3_STATE	DUT must correctly request L1 entry when software sets the DUT state to D3 for all supported data rates.
LINKUPCONFIG	To check for correct linkup behavior.	LINKUPCONFIG	The DUT must ignore the reserved bits of the TS Ordered sets, and treat reserved bits as zero.
LOOPBACK	To verify that the DUT correctly enters Loopback through Configuration.	LOOPBACK	The DUT must correctly enter loopback state through configuration state.
LOOPBACKFROML0	To verify that the DUT correctly enters Loopback from L0.	LOOPBACKFROML0	The DUT must correctly enter loopback state from L0.
LTR_ENABLE_CLEARED	To verify that DUT correctly transmits an LTR message after the LTR enable bit has been cleared.	LTR_ENABLE_CLEARED	DUT must correctly transmits LTR message for all supported data rates.
LTR_NON_D0_STATE	To verify that DUT sends an LTR message after it has been directed to a non-D0 active state.	LTR_NON_D0_STATE	DUT must send LTR message after it has been directed to a non-D0 active state.
RESERVEDBITSINTS	To verify that the reserved bits in received TS are not set.	RESERVEDBITSINTS	The reserved fields in TS Ordered sets must be ignored by DUT.
TXN_BFT_REQUEST_COMPLETION_UR	To verify that DUT issues completion with Unsupported Request completion status for configuration requests to function number that it does not support.	TXN_BFT_REQUEST_COMPLETION_UR	DUT must issue completion with Unsupported Request completion status for configuration requests to function number that it does not support.

Acknowledgements

Protocol Exerciser GUI uses Outlook Bar, a Graphical library created by Tim Dawson (Refer http://www.divil.co.uk/net/).