Keysight Protocol Exerciser for PCI Express

User Guide

KEYSIGHT TECHNOLOGIES

Notices

Installation Guide

You can find the installation guides for different components of the product on the product CD. Keysight 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.keysight.com/find/spt

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

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

Edition

10th Edition, November 2014

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, KEYSIGHT 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. KEYSIGHT SHALL NOT BE LIABLE FOR ERRORS OR FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, USE, OR PERFORMANCE OF THIS DOCUMENT OR ANY INFORMATION CONTAINED HEREIN. SHOULD KEYSIGHT 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 WILL 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 Keysight 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 Summary

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 or operating instructions in the product manuals violates safety standards of design, manufacture, and intended use of the instrument. Keysight Technologies assumes no liability for the customer's failure to comply with these requirements. Product manuals are provided with your instrument on CD-ROM and/or in printed form. Printed manuals are an option for many products. Manuals may also be available on the Web. Go to www.keysight.com and type in your product number in the Search field at the top of the page.

General

Do not use this product in any manner not specified by the manufacturer. The protective features of this product may be impaired if it is used in a manner not specified in the operation instructions.

Before Applying Power

Verify that all safety precautions are taken. Make all connections to the unit before applying power. Note the instrument's external markings described in “Safety Symbols”.

Ground the Instrument

If your product is provided with a grounding type power plug, the instrument chassis and cover must be connected to an electrical ground to minimize shock hazard. The ground pin must be firmly connected to an electrical ground (safety ground) terminal 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.

Fuses

See the user's guide or operator's manual for information about line-fuse replacement. Some instruments contain an internal fuse, which is not user accessible.

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

Only qualified, service-trained personnel who are aware of the hazards involved should remove instrument covers. Always disconnect the power cable and any external circuits before removing the instrument cover.

Cleaning

Clean the outside of the instrument with a soft, lint-free, slightly dampened cloth. Do not use detergent or chemical solvents.

Do Not Modify the Instrument

Do not install substitute parts or perform any unauthorized modification to the product. Return the product to an Keysight Sales and Service Office for service and repair to ensure that safety features are maintained.

In Case of Damage

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

CAUTION

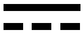
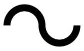
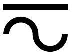
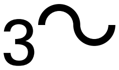
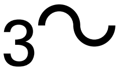
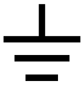
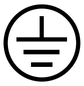
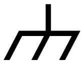
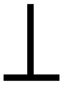
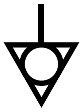
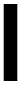
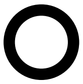
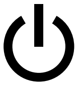
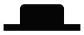
A CAUTION notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in damage to the product or loss of important data. Do not proceed beyond a CAUTION notice until the indicated conditions are fully understood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

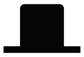
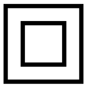
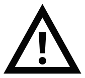
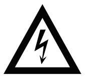
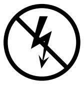
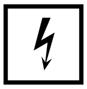
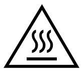
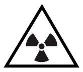
WARNING

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

Table 1 Safety Symbol

Symbol	Description
	Direct current
	Alternating current
	Both direct and alternating current
	Three phase alternating current
	Three phase alternating current
	Earth ground terminal
	Protective earth ground terminal
	Frame or chassis ground terminal
	Terminal is at earth potential
	Equipotentiality
N	Neutral conductor on permanently installed equipment
L	Line conductor on permanently installed equipment
	On (mains supply)
	Off (mains supply)
	Standby (mains supply). The instrument is not completely disconnected from the mains supply when the power switch is in the standby position
	In position of a bi-stable push switch

Symbol	Description
	Out position of a bi-stable push switch
	Equipment protected throughout by DOUBLE INSULATION or REINFORCED INSULATION
	Caution, refer to accompanying documentation
	Caution, risk of electric shock
	Do not apply around or remove from HAZARDOUS LIVE conductors
	Application around and removal from HAZARDOUS LIVE conductors is permitted
	Caution, hot surface
	Ionizing radiation
CAT I	IEC Measurement Category I
CAT II	Measurement Category II
CAT III	Measurement Category III
CAT IV	Measurement Category IV

Compliance and Environmental Information

Table 2 Compliance and Environmental Information

Safety Symbol	Description
SP ©	CSA is the Canadian certification mark to demonstrate compliance with the Safety requirements.
✓	The C-tick mark is a registered trademark of the Spectrum Management Agency of Australia. This signifies compliance with the Australia EMC Framework regulations under the terms of the Radio Communication Act of 1992.
CE	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.

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 Keysight office, or see www.keysight.com/environment/product for more information.

Document History

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

Manual Name: Keysight Protocol Exerciser for PCI Express - User's Guide

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

Published	Applicable Release
October 2014	8.74

Contents

What’s New in this Release

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

- Exerciser can now function as an NVMe endpoint. You can configure its controller registers as an NVMe endpoint and then start NVMe traffic on it. You can also view and edit the values to be used with the Get and Set Features commands. While emulating an NVMe endpoint, it supports the mandatory Admin as well as NVMe commands execution. For more details on how to configure and use Exerciser as an NVMe endpoint, refer to Chapter 12, “Emulating an NVMe EndPoint”.
- The API calls initiated through the Exerciser GUI are now logged. You can view the logged API calls using the API log window. To know more, refer to the “Log Window Dialog Box" on page 256.
- Exerciser now supports creation and edition of SGL lists when emulating an NVMe Root Complex. You can add SGL segments and SGL description to an SGL list. To know more, refer to “Creating and Using a SGL List" on page 198.
- Exerciser can now support x2 linkwidth which was not available in the earlier releases.
- Exerciser now provides two initialization scripts to initialize NVMe devices with 64bit and 32bit BARs. To know more, refer to “Running the NVMe Conformance Test Cases" on page 240.

1 Protocol Exerciser for PCI Express - Introduction

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

Keysight Protocol Exerciser for PCI Express (PCIe) provides a test and debug solution for PCI Express to test the next generation of PCI Express technology. Keysight 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.

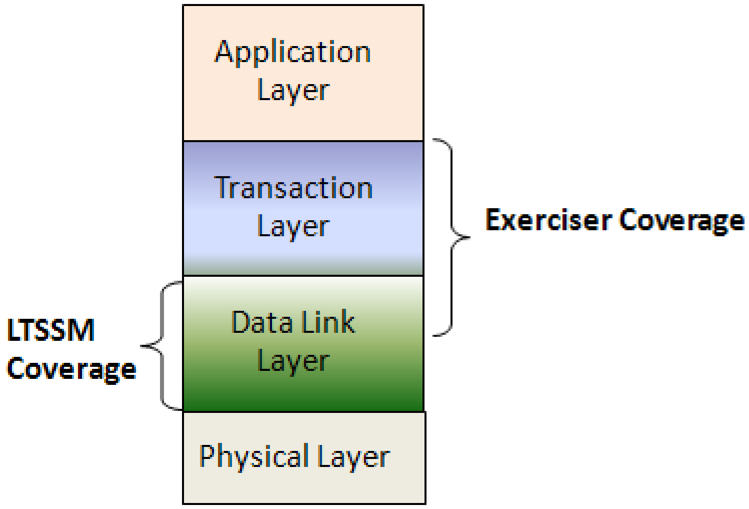


Figure 1 Protocol Exerciser coverage

Refer to the topic “Protocol Exerciser for PCI Express Modes (As Exerciser and as LTSSM)" on page 24 to know more about its functions as an Exerciser and an LTSSM.

Emulation Modes

The Keysight 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 Keysight 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 Keysight 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.

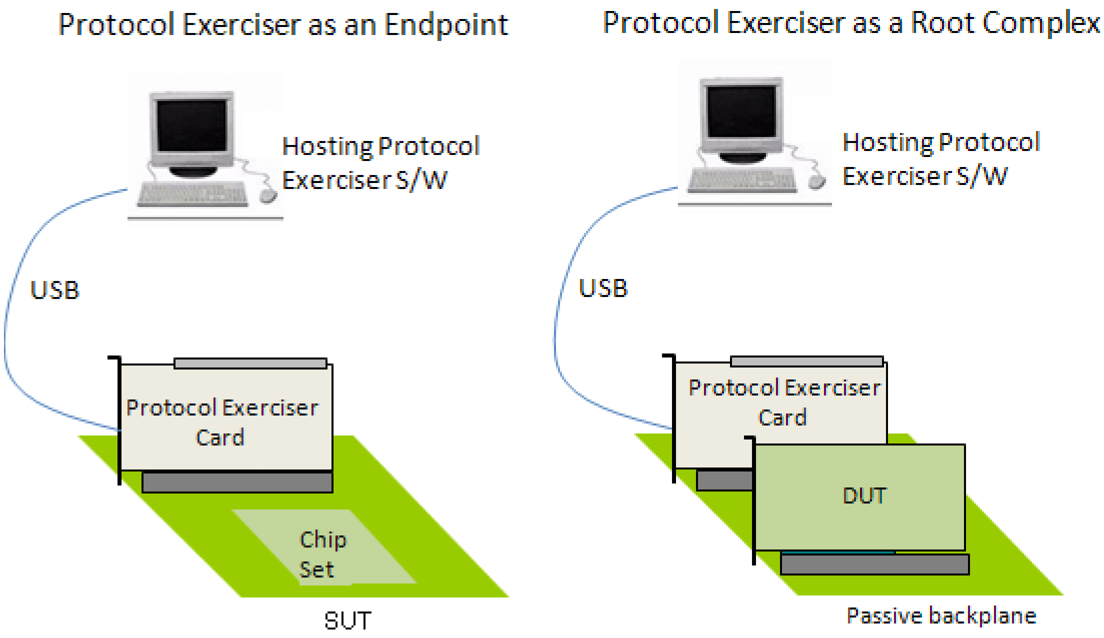

Protocol Exerciser as an Endpoint
Hosting Protocol Exerciser S/W
USB
Protocol Exerciser Card
Chip Set
SUT
Protocol Exerciser as a Root Complex
Hosting Protocol Exerciser S/W
USB
Protocol Exerciser Card
DUT
Passive backplane

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 110.

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 138.

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 104.

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	Function A (function number 0) Function B (function number 1) Function C (function number 2)	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/AN/A	N/A	

As a SRIOV capable component

	Base and Physical Functions	Virtual functions	Hardware Channels
Virtual Channels	Completion Queues	Virtual Link	Virtual hierarchies
Default	3	Function A (function number 0)	Function B (function number 1)
Function C (function number 2)	4	7	2
2 (Queue0 and Queue1)	1	N/A	With additional 2 physical functions license
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)	8	13
2	2 (Queue0 and Queue1)	1	N/A

	Base and Physical Functions	Virtual functions	Hardware Channels
Notes	Function A is non IOV.	Function B to E are physical functions and have the SRIOV extended capabilities.	Two virtual functions (VF1 and VF2) associated with each physical function
One hardware channel associated with each base, physical and virtual function	Virtual Resource 0 - VC0	Virtual Resource 1 - VC [x]	(where x is the number that you assign to this second VC resource)
Queue0 mapped to VC resource 0	Queue1 mapped to VC resource 1		

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	All virtual hierarchies are mapped to VL0.

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.

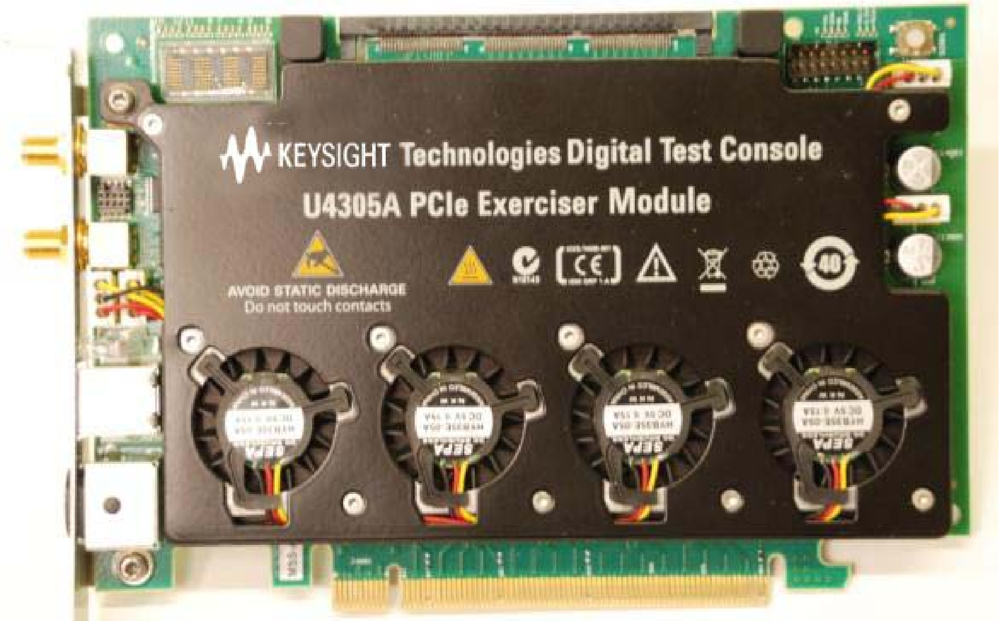

KEYSIGHT Technologies Digital Test Console
U4305A PCIe Exerciser Module
AVOID STATIC DISCHARGE
Do not touch contacts

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 37.
- **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 308.

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

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.

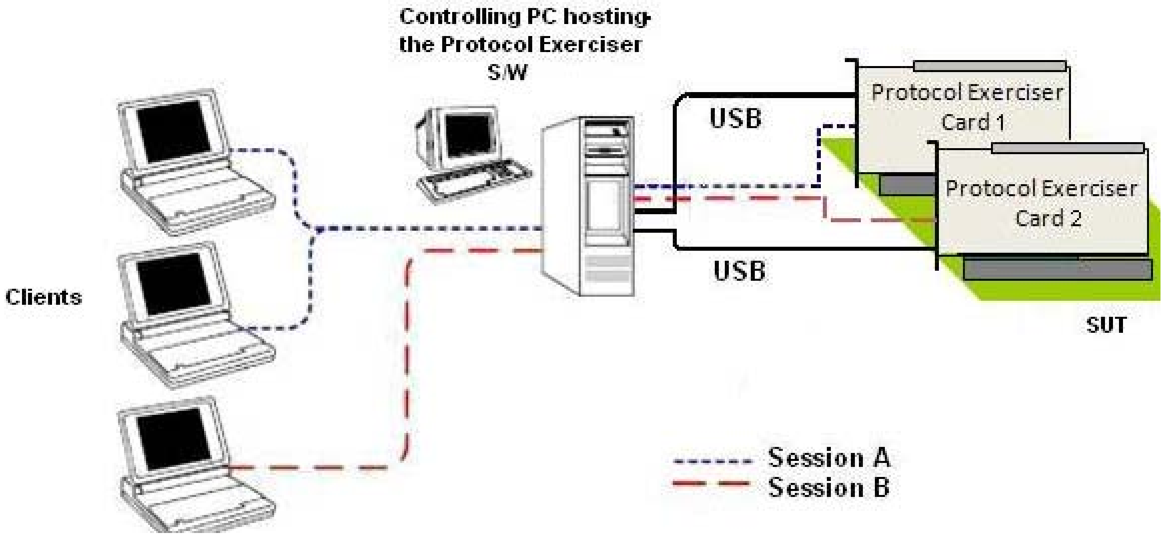

Controlling PC hosting the Protocol Exerciser S/W
USB
Protocol Exerciser Card 1
Protocol Exerciser Card 2
USB
Clients
SUT
Session A
Session B

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.

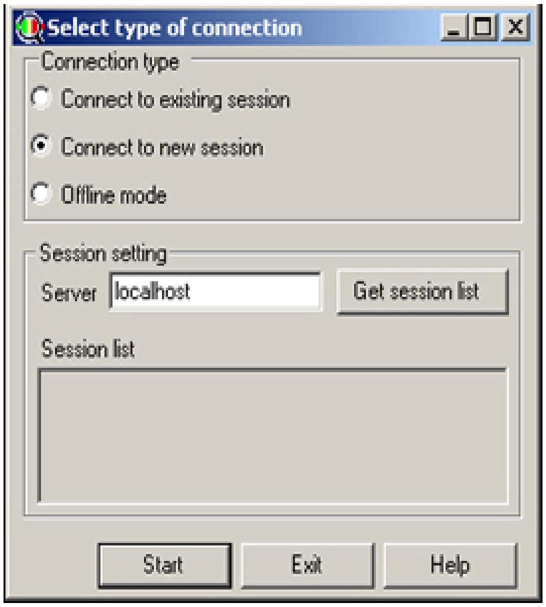
Protocol Exerciser does not enable you to lock a session (online or offline). Therefore, it is possible for multiple Protocol Exerciser instances to concurrently access a single session.

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

Session list

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

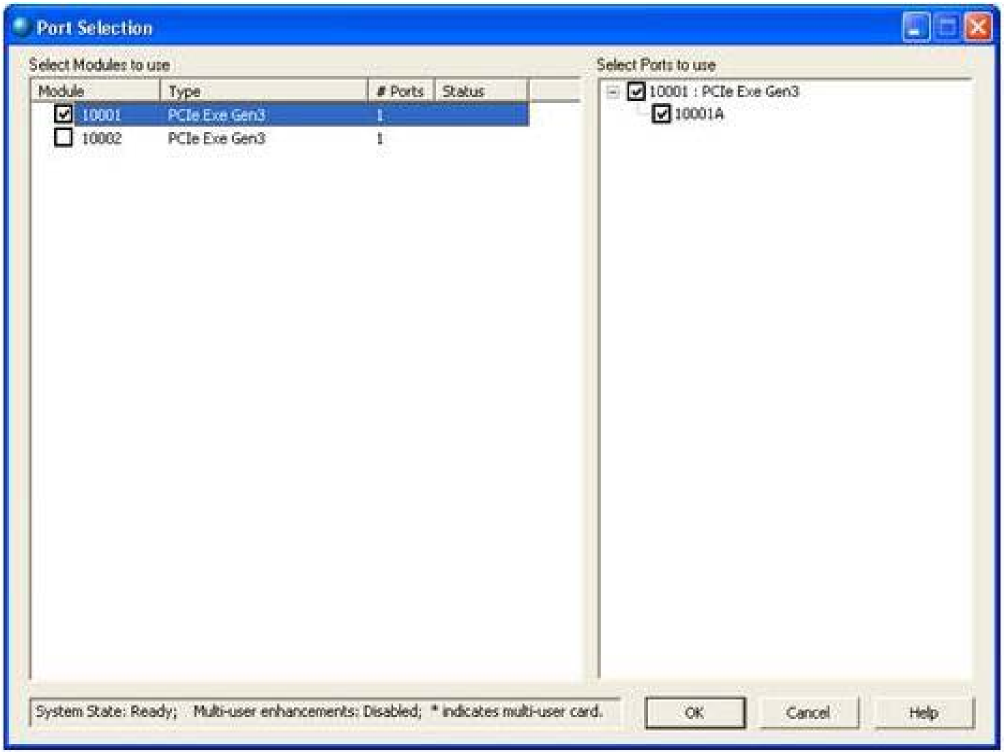
1 Select the **Connect to new session** radio button.

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 Exe Gen3	1	
☐ 10002	PCIe Exe Gen3	1	

☑ 10001 : PCIe Exe Gen3
☑ 10001A

System State: Ready;   Multi-user enhancements: Disabled;   * 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.

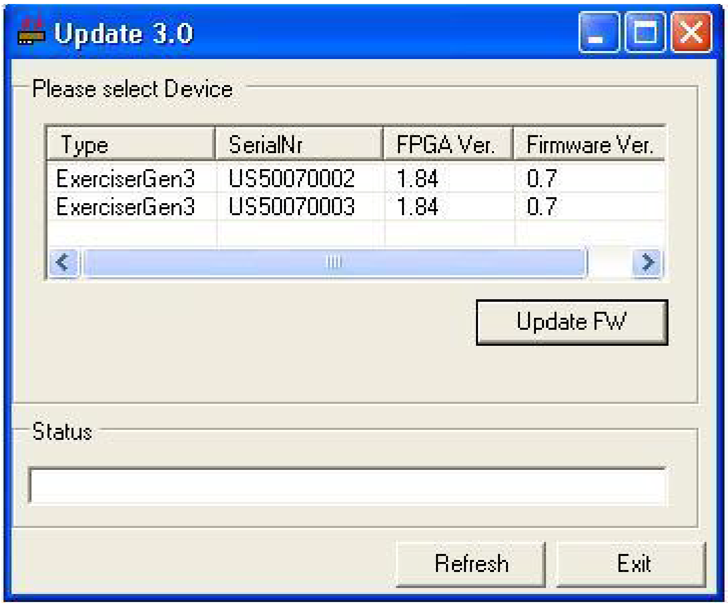
5 Click **OK**.

At any time, you can access the session information by clicking the **Help -> Session Information** 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 308 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 > Keysight SPT > PCIe Exerciser 8.74 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 32 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.

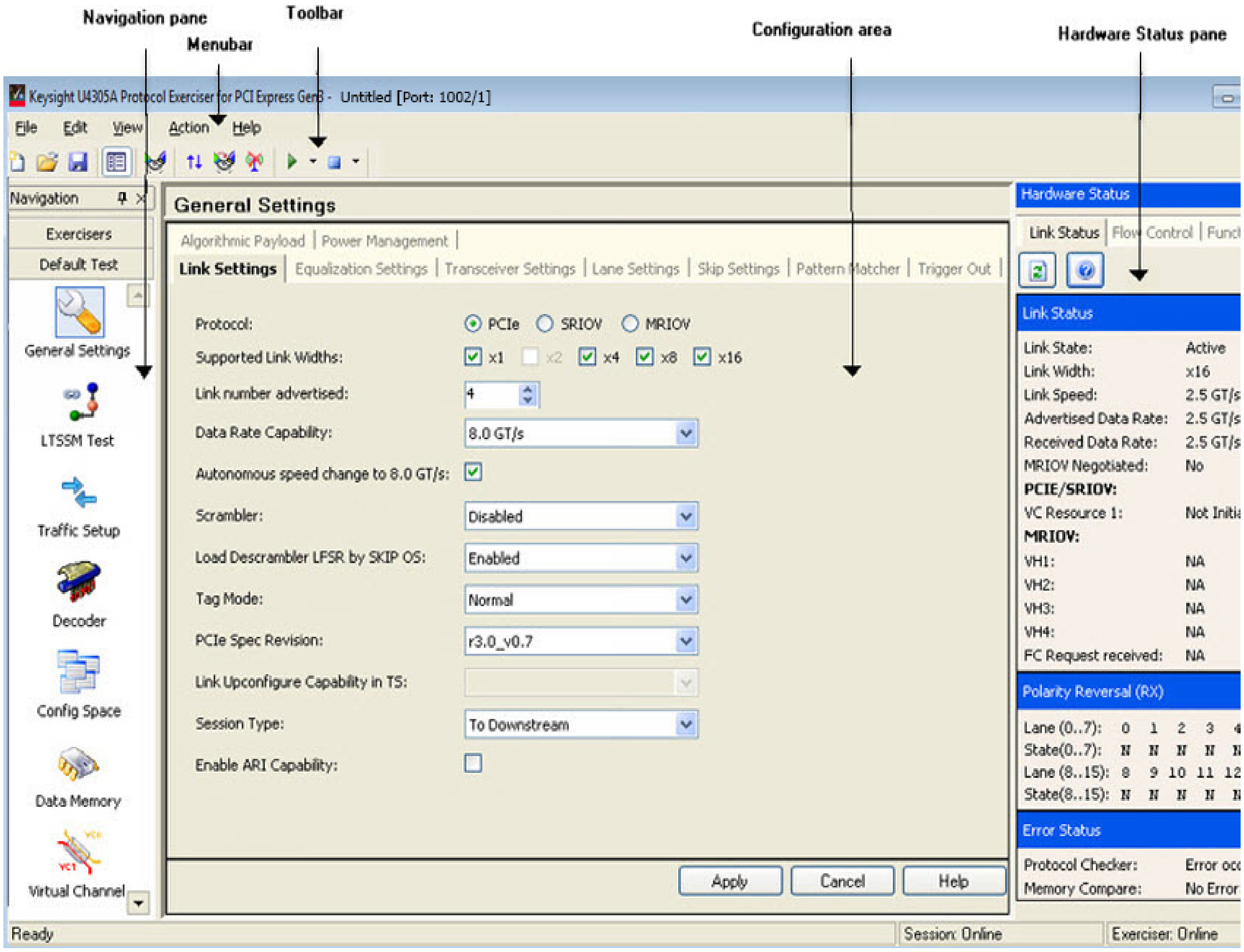
   Completing this message box displays the Closing Session dialog box.
4. Do one of the following:
   - Click **Yes** to remove the associated session before closing the Protocol Exerciser window.
   - Click **No** to close the Protocol Exerciser window without removing its associated session.

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 Status

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

Polarity Reversal (RX)

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

Error Status

Protocol Checker:	Error occ
Memory Compare:	No Error

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).

- **Hard ware 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.

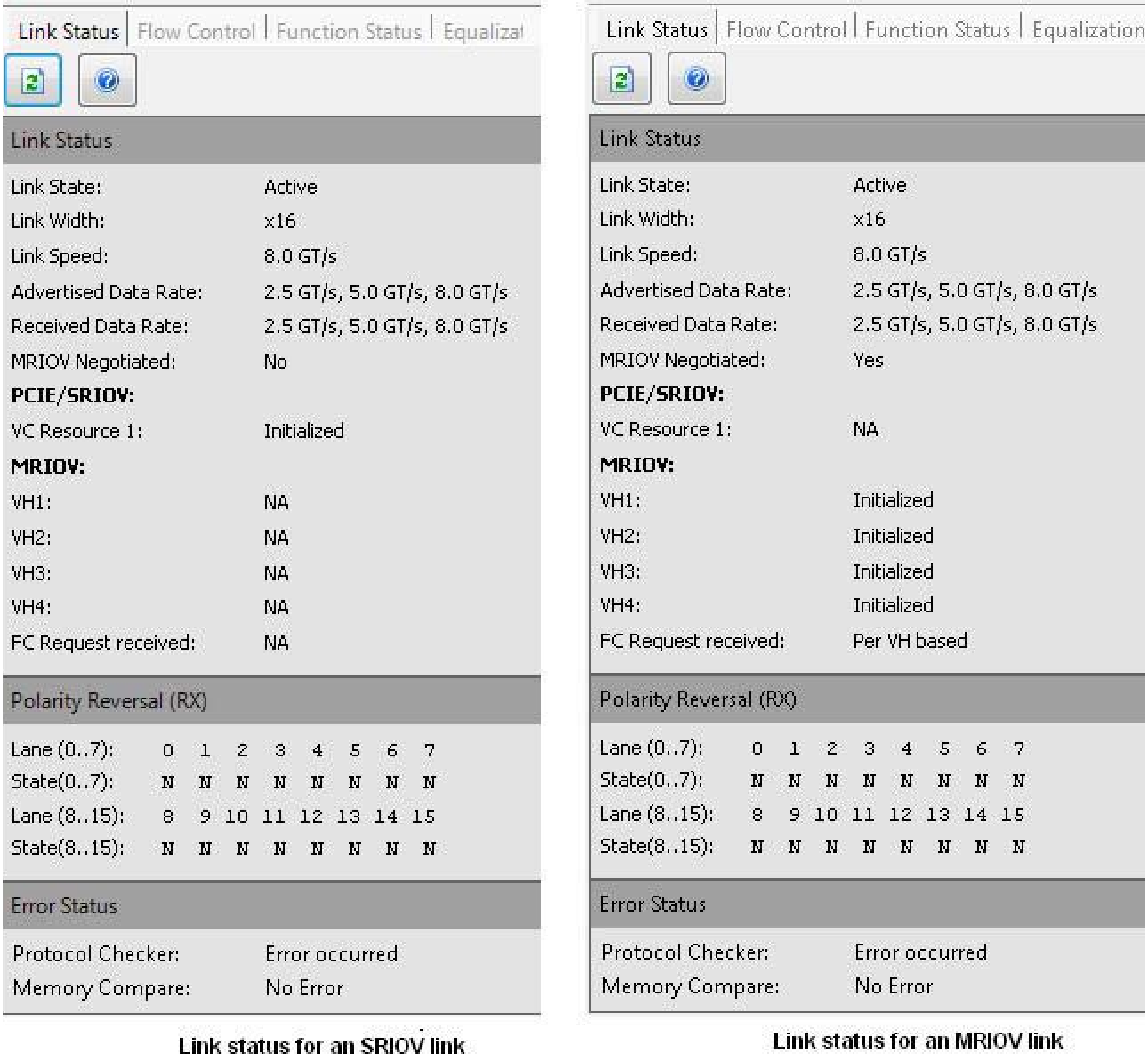
The information in this pane is updated when you perform any action such as sending stimulus or running a test that impacts the link partners and link status. For instance, when the link between the DUT and Protocol Exerciser is not up, the link status is displayed as No Link. When you perform the steps 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 > Hard ware Status**.

  The H**ard ware Status** pane appears. A sample screen is displayed.



Link status for an SRIOV link

Link Status	
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
MRIOV Negotiated:	No
PCIE/SRIOV:	
VC Resource 1:	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

Link status for an MRIOV link

Link Status	
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
MRIOV Negotiated:	Yes
PCIE/SRIOV:	
VC Resource 1:	NA
MRIOV:	
VH1:	Initialized
VH2:	Initialized
VH3:	Initialized
VH4:	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

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:

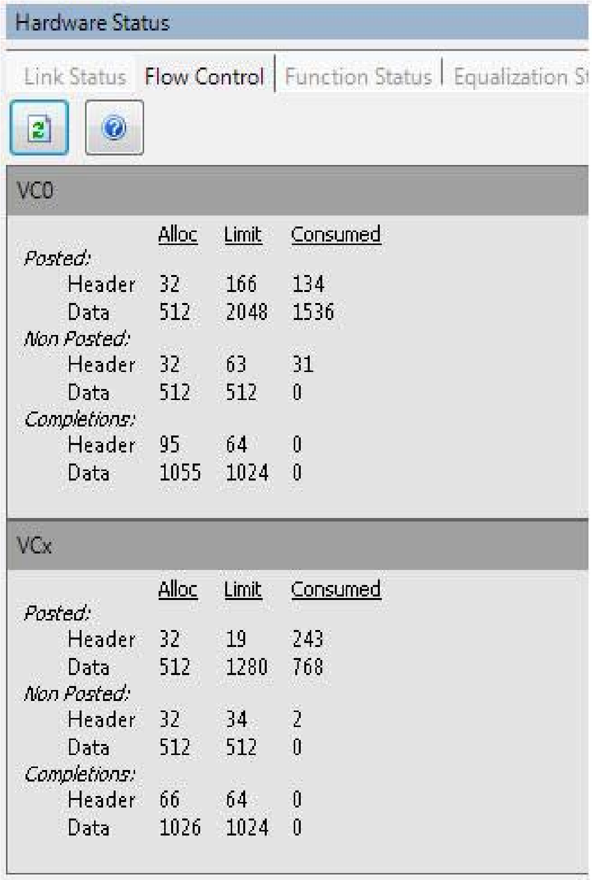
Link Status

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 **VL only** flow control is applicable and displayed. Refer to the topic "Initializing Flow Control" on page 81 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 166 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 131 to know more.

Flow Control



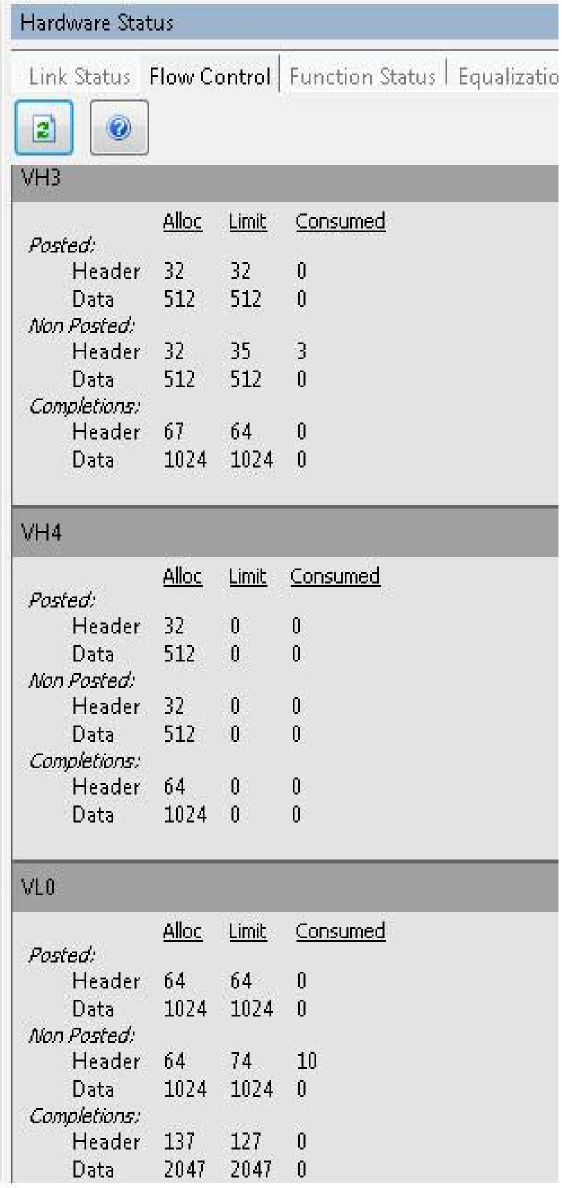
VC0

	Alloc	Limit	Consumed
Posted:			
Header	32	166	134
Data	512	2048	1536
Non Posted:			
Header	32	63	31
Data	512	512	0
Completions:			
Header	95	64	0
Data	1055	1024	0

VCx

	Alloc	Limit	Consumed
Posted:			
Header	32	19	243
Data	512	1280	768
Non Posted:			
Header	32	34	2
Data	512	512	0
Completions:			
Header	66	64	0
Data	1026	1024	0

Flow control status for SRIOV capable Exerciser



VH3

	Alloc	Limit	Consumed
Posted:			
Header	32	32	0
Data	512	512	0
Non Posted:			
Header	32	35	3
Data	512	512	0
Completions:			
Header	67	64	0
Data	1024	1024	0

VH4

	Alloc	Limit	Consumed
Posted:			
Header	32	0	0
Data	512	0	0
Non Posted:			
Header	32	0	0
Data	512	0	0
Completions:			
Header	64	0	0
Data	1024	0	0

VL0

	Alloc	Limit	Consumed
Posted:			
Header	64	64	0
Data	1024	1024	0
Non Posted:			
Header	64	74	10
Data	1024	1024	0
Completions:			
Header	137	127	0
Data	2047	2047	0

Flow control status for MRIOV capable Exerciser

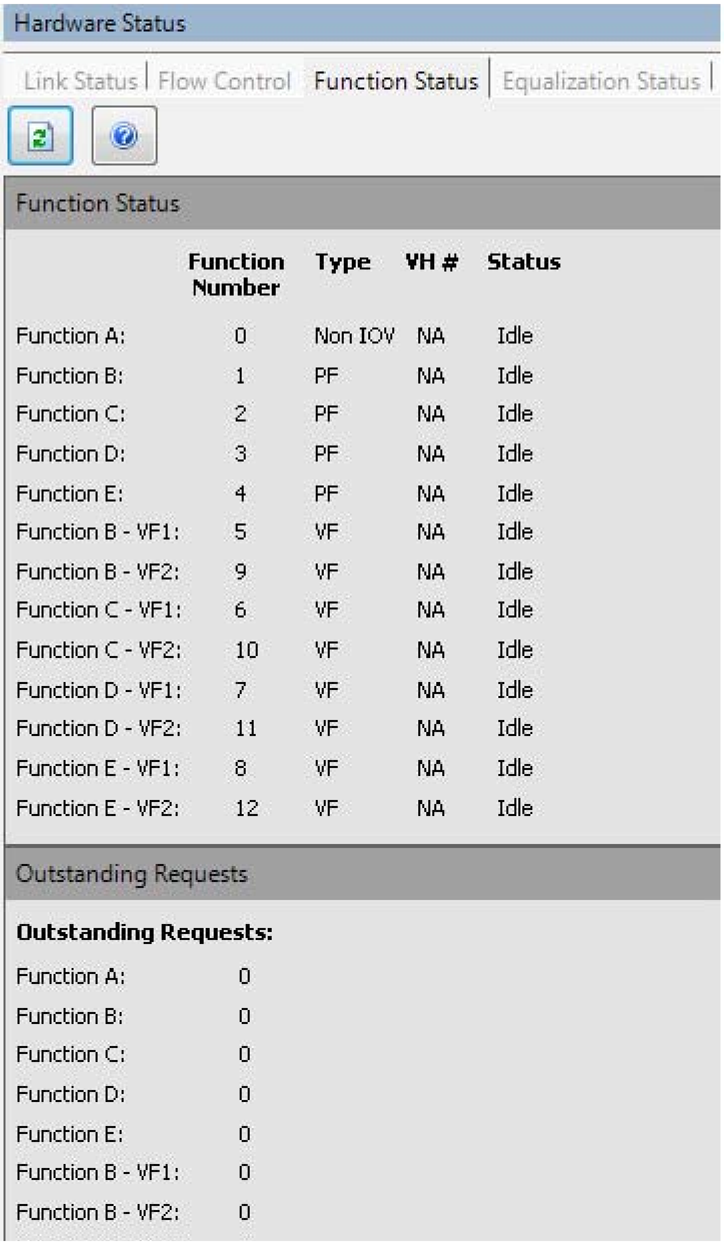
This tab provides the flow control credit information of the underlying Exerciser's Receiver Buffer and the credit limit available to the Transmitter. It displays the allocated, used, and available credit limit of virtual channels of Exerciser as per the credit limits that you assigned to these virtual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

For a MRIOV capable Exerciser, the flow control status for virtual hierarchies and virtual link VL0 is displayed. For a non IOV / SRIOV capable Exerciser, the flow control status for virtual channels (VC0 and VCx) are displayed.

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

Function Status



Function Status

	Function Number	Type	VH #	Status
Function A:	0	Non IOV	NA	Idle
Function B:	1	PF	NA	Idle
Function C:	2	PF	NA	Idle
Function D:	3	PF	NA	Idle
Function E:	4	PF	NA	Idle
Function B - VF1:	5	VF	NA	Idle
Function B - VF2:	9	VF	NA	Idle
Function C - VF1:	6	VF	NA	Idle
Function C - VF2:	10	VF	NA	Idle
Function D - VF1:	7	VF	NA	Idle
Function D - VF2:	11	VF	NA	Idle
Function E - VF1:	8	VF	NA	Idle
Function E - VF2:	12	VF	NA	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

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

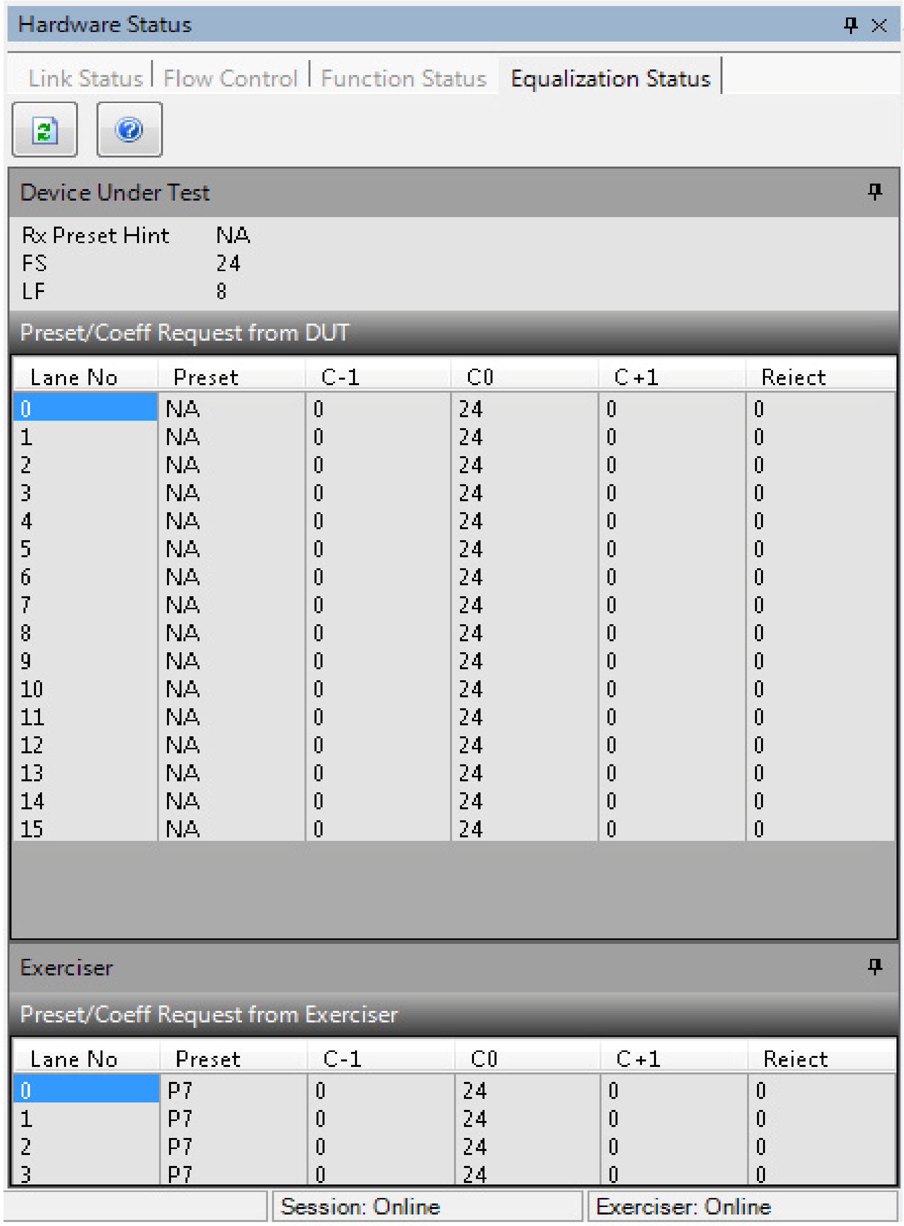
C. If you select the MRIOV or SRIOV protocol, then this tab displays information related to the virtual functions as well. If you have the additional 2 physical function license, then this tab displays the status of five functions (Function A to E). Refer to the topic "Functions, Hardware Channels, Virtual Hierarchies, and Virtual Channels" on page 26 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.

Equalization Status



Device Under Test

Rx Preset Hint	NA
FS	24
LF	8

Preset/Coeff Request from DUT

Lane No	Preset	C-1	C0	C+1	Reject
0	NA	0	24	0	0
1	NA	0	24	0	0
2	NA	0	24	0	0
3	NA	0	24	0	0
4	NA	0	24	0	0
5	NA	0	24	0	0
6	NA	0	24	0	0
7	NA	0	24	0	0
8	NA	0	24	0	0
9	NA	0	24	0	0
10	NA	0	24	0	0
11	NA	0	24	0	0
12	NA	0	24	0	0
13	NA	0	24	0	0
14	NA	0	24	0	0
15	NA	0	24	0	0

Exerciser

Preset/Coeff Request from Exerciser

Lane No	Preset	C-1	C0	C+1	Reject
0	P7	0	24	0	0
1	P7	0	24	0	0
2	P7	0	24	0	0
3	P7	0	24	0	0

Session: Online

Exerciser: Online

This tab provides the status information for the equalization settings after the link up is achieved between Exerciser and DUT at 8 GT/s. If the link up between Exerciser and DUT is at 2.5 or 5.0 GT/s, the Equalization Status tab displays "NA" as the status of equalization settings. The equalization status is categorized primarily in the following two sections:

- **Device Under Test** - This section displays the equalization settings advertised/requested by DUT. It displays:
    - the Rx preset hint that the DUT sent to Exerciser for its Rx AC gain adjustment. This is displayed as NA if DUT is a DSC.
    - the FS and LF values that DUT advertised in phase 1 of equalization

  - the final preset and coefficient request received from DUT for each lane. It also displays whether or not Exerciser rejected the requested preset and coefficient values for a lane. The Reject column displays a 1 if rejected and 0 if accepted.
- **Exerciser** - This section displays the final preset and coefficient values requested by Exerciser for each lane. It also displays whether or not DUT rejected the requested preset and coefficient values for a lane. The Reject column displays a 1 if rejected and 0 if accepted.

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 > Hard ware 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.

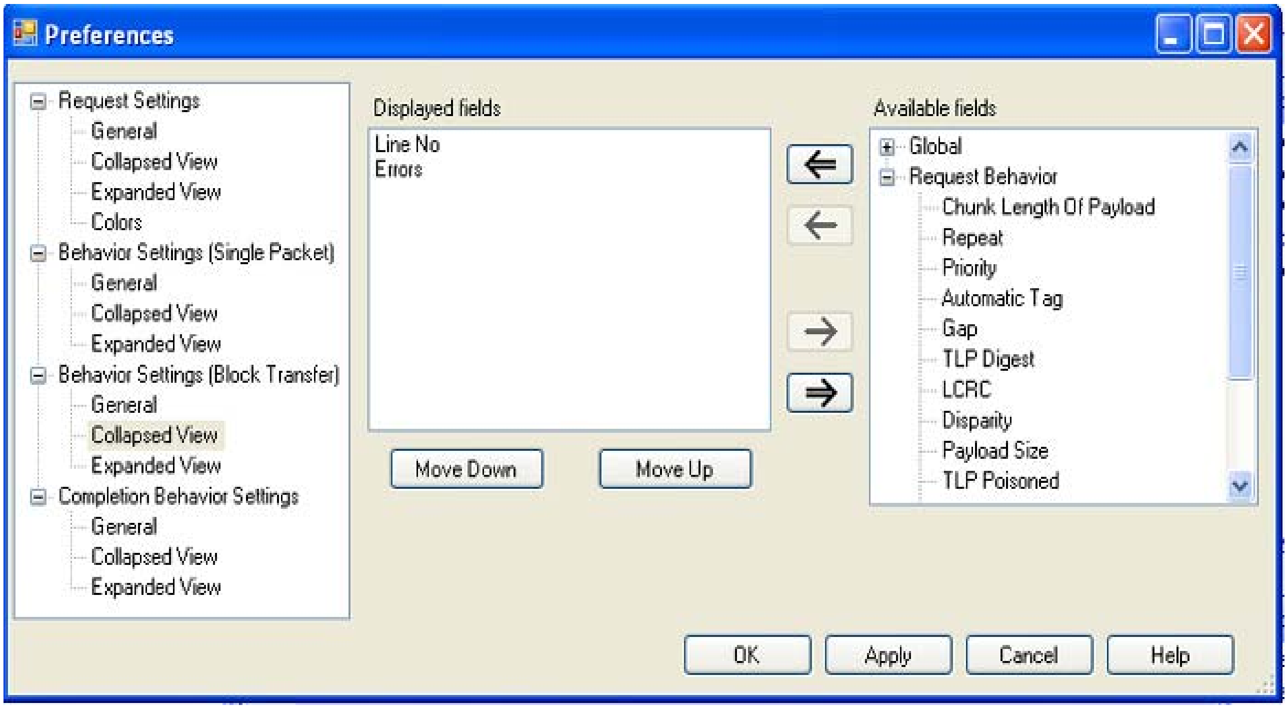
Configuring Data Display Preferences Settings

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 259.

3 Configuring Protocol Exerciser

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

When you save Exerciser settings in an .exs setup file, all the settings are saved in the file except the settings you made to the:

- DUT Config Space page

- NVMe Express page displayed when you configure Exerciser in NVMe Root Complex mode.

The settings made to the NVMe Express page displayed when you configure Exerciser in NVMe Endpoint mode are saved to the .exs 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 PCIe Link Settings

Before you can create/initialize a PCIe link between Protocol Exerciser and DUT for 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 non IOV PCIe link or an SRIOV/MRIOV capable 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 test 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 the appropriate SRIOV software license without which the SRIOV option is disabled in the Link settings tab. The U4305A-026 software license provides SRIOV capabilities for Exerciser at Gen1, Gen2, and Gen3 speed.
- **MR Enabled PCIe link** - If you want to test 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.

  To enable the MRIOV capabilities for Exerciser, you need the appropriate MRIOV software license without which the MRIOV option is disabled in the Link settings tab. The U4305A-025 software license provides MRIOV capabilities for Exerciser at Gen1, Gen2, and Gen3 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.

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.

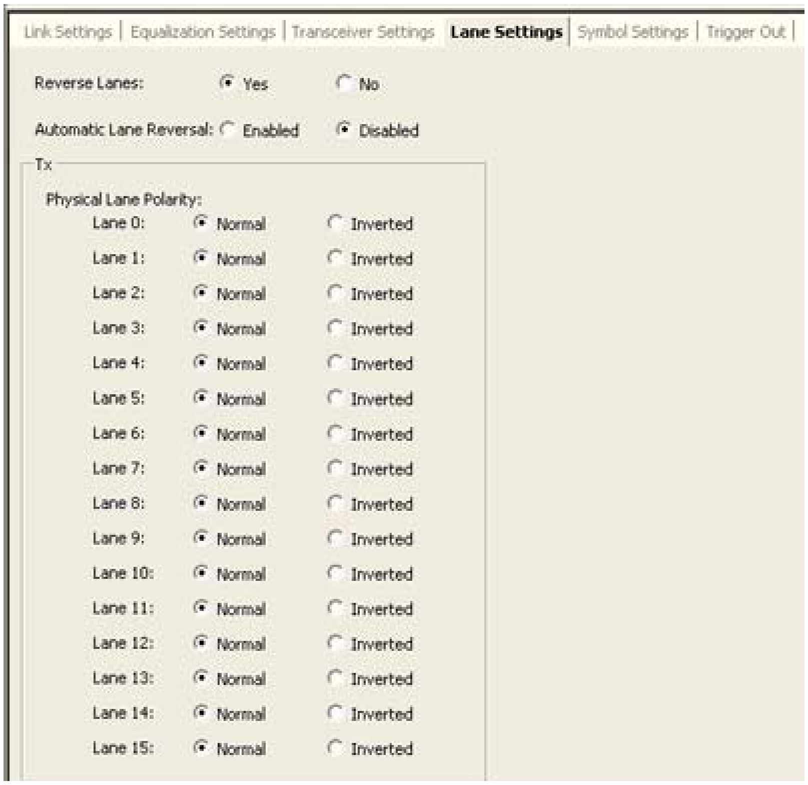
- You select the session type based on whether you want the Protocol Exerciser to act as a USC or a DSC for the session. When Protocol Exerciser is emulating a root complex and acting as a USC, the session type should be To Downstream. The DUT is at the end point in this case. When Protocol Exerciser 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

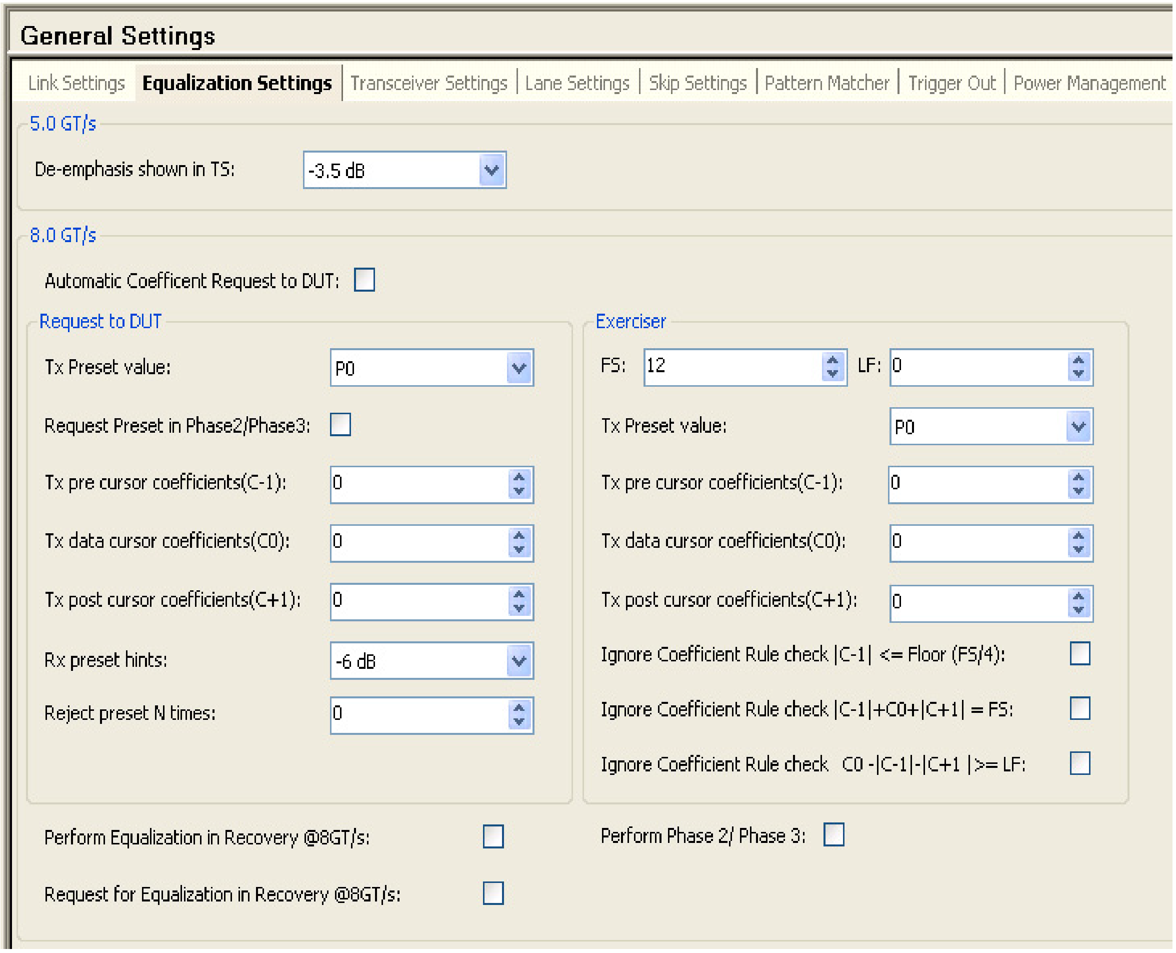
You can enable or disable the lane reversal feature for the Transmitter. Protocol Exerciser can reverse its lanes before starting the link negotiation. Enabling the lane reversal can help you test the DUT's capability to perform lane ordering during link training.

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 configure the equalization settings for Protocol Exerciser separately for 5.0 and 8.0 link speeds. This topic describes how to enable/disable and configure the equalization settings.


General Settings

Link Settings | **Equalization Settings** | Transceiver Settings | Lane Settings | Skip Settings | Pattern Matcher | Trigger Out | Power Management

5.0 GT/s

De-emphasis shown in TS: -3.5 dB

8.0 GT/s

Automatic Coefficent Request to DUT: □

Request to DUT

Tx Preset value: P0

Request Preset in Phase2/Phase3: □

Tx pre cursor coefficients(C-1): 0

Tx data cursor coefficients(C0): 0

Tx post cursor coefficients(C+1): 0

Rx preset hints: -6 dB

Reject preset N times: 0

Exerciser

FS: 12 LF: 0

Tx Preset value: P0

Tx pre cursor coefficients(C-1): 0

Tx data cursor coefficients(C0): 0

Tx post cursor coefficients(C+1): 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, Exerciser and DUT have to go through the phases of the Equalization procedure. The Exerciser and DUT 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.0

You can choose to enable or disable the Phase 2 and 3 of equalization when Exerciser is acting as a USC.

Types of Equalization

Protocol Exerciser supports the following two types of equalization when moving to 8.0 GT/s:

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 use the **Equalization Settings** and **Transceiver Settings** tabs in the General Settings navigation pane to configure various setting that impact equalization at Exerciser and DUT ends.

Broadly, you can configure the following equalization settings:

- Exerciser's Tx settings that it advertises during equalization phases: Settings such as FS, LF, Tx preset value (if Exerciser is a USC), and Tx coefficients. You use the Exerciser groupbox in the Equalization Settings tab to configure these settings.
- Preset and coefficients requests to DUT: You use the Request to DUT groupbox in the Equalization Settings tab to configure these settings.
- Exerciser's transceiver settings: Settings such as Exerciser's Tx De-emphasis, preshoot, boost levels, and Rx AC and DC gains. You use the Transceiver Settings tab to configure these settings.

For configuring these equalization settings, you can:

- either manually specify the values that Exerciser should use while performing or responding to equalization phases
- or instruct Exerciser to automatically compute or adjust its applicable equalization settings

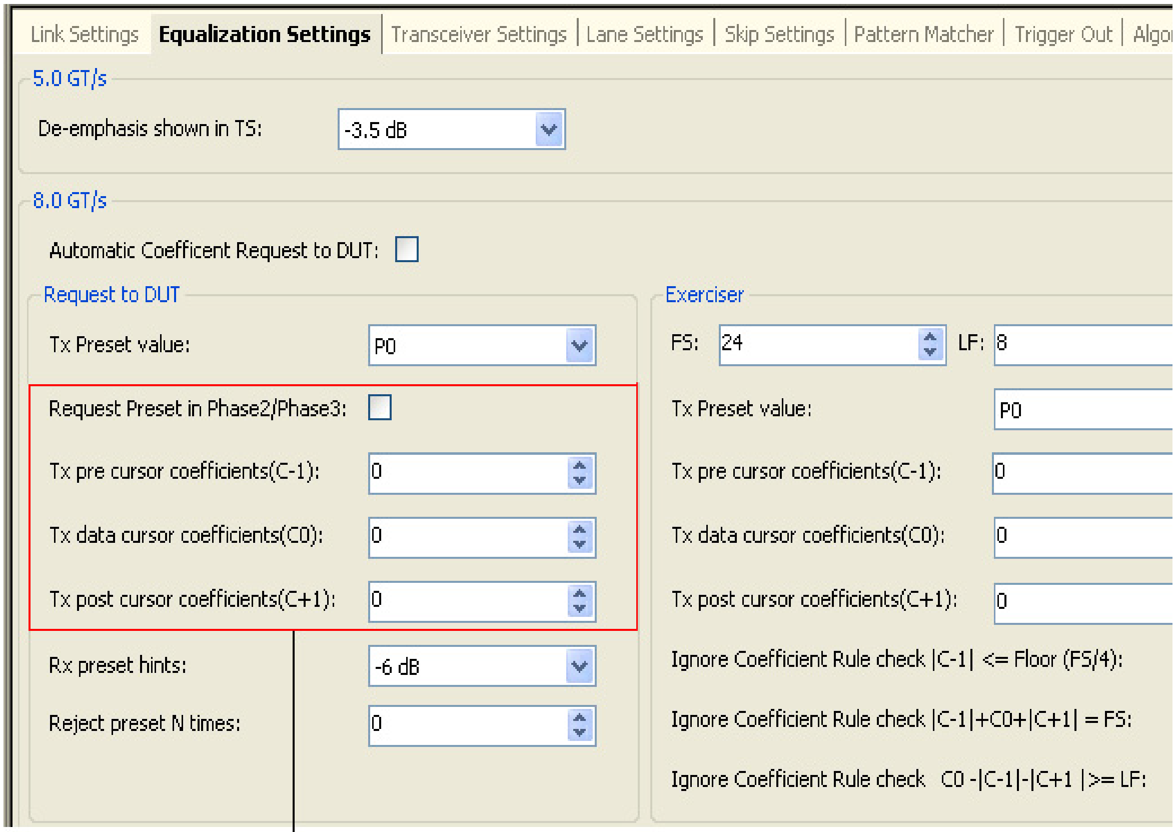
The following sections describe how you can manually set the equalization settings or allow Exerciser to auto compute these as an active participant in different phases of equalization.

Preset and Coefficient request to DUT

For preset request to be sent to DUT, you set the:

- Tx preset value using the Tx preset value listbox in the Request to DUT groupbox of the Equalization Settings tab.
- Rx present hint to be sent to DUT using the Rx preset hints listbox in the Request to DUT groupbox of the Equalization Settings tab.

For coefficient request to be sent to DUT, you can set the coefficient values using manual or automatic methods available in the Equalization Settings tab as displayed in the following screens.



5.0 GT/s

De-emphasis shown in TS: -3.5 dB

8.0 GT/s

Automatic Coefficent Request to DUT: □

Request to DUT

Tx Preset value: P0

Request Preset in Phase2/Phase3: □

Tx pre cursor coefficients(C-1): 0

Tx data cursor coefficients(C0): 0

Tx post cursor coefficients(C+1): 0

Rx preset hints: -6 dB

Reject preset N times: 0

Exerciser

FS: 24 LF: 8

Tx Preset value: P0

Tx pre cursor coefficients(C-1): 0

Tx data cursor coefficients(C0): 0

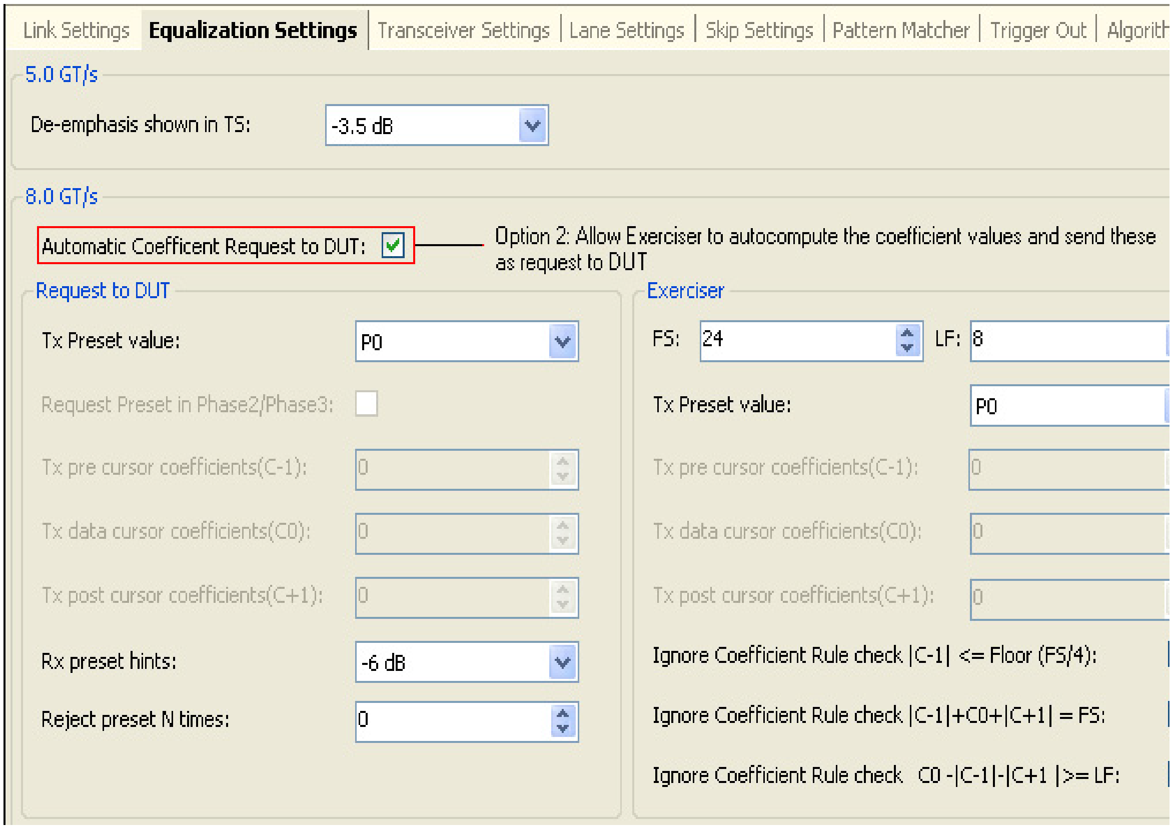
Tx post cursor coefficients(C+1): 0

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:

Option 1 : Manually set the coefficient values to be sent as request to DUT



5.0 GT/s

De-emphasis shown in TS: -3.5 dB

8.0 GT/s

Automatic Coefficent Request to DUT: ☑

Option 2: Allow Exerciser to autocompute the coefficient values and send these as request to DUT

Request to DUT

Tx Preset value: P0

Request Preset in Phase2/Phase3: □

Tx pre cursor coefficients(C-1): 0

Tx data cursor coefficients(C0): 0

Tx post cursor coefficients(C+1): 0

Rx preset hints: -6 dB

Reject preset N times: 0

Exerciser

FS: 24 LF: 8

Tx Preset value: P0

Tx pre cursor coefficients(C-1): 0

Tx data cursor coefficients(C0): 0

Tx post cursor coefficients(C+1): 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:

If you choose the automatic method of computing the coefficient values, the following are automatically computed by Exerciser and therefore disabled in the GUI to prevent their manual setting.

- Tx coefficient values that the Protocol Exerciser advertises during the link training as its Tx capabilities. These fields are present in the Exerciser group box of Equalization Settings tab.
- Tx coefficient values that the Protocol Exerciser will send to DUT in the training sequences to request DUT to use the specified coefficient values. These fields are present in the Request to DUT group box of the Equalization Settings tab.

Exerciser's Transmitter Settings

You can either set the Transmitter settings of Exerciser manually or configure Exerciser to do Tx equalization automatically based on preset or coefficient request made by DUT. In case of manual settings, Exerciser applies the specified De-emphasis, Pre-shoot & Boost levels on its transmitted signals. In case you configure Exerciser for automatic computation, Exerciser applies the De-emphasis, Pre-shoot & Boost levels requested by DUT on its transmitted signals.

To set Exerciser's Transmitter settings manually

1 In the **Transceiver Settings** tab, ensure that the **Automatic** (As per DUT request) checkbox is not selected.

2 Specify the **Tx Output Voltage, Tx Pre-emphasis Pre Tap, Tx Pre-emphasis First Post Tap, Tx Pre-emphasis Second Post Tap** for Exerciser in the **Transceiver Settings** tab.

3 Click **Apply**.

4 In the **Exerciser** group box of the **Equalization Settings** tab, specify the **FS** and **LF** values that Exerciser advertises during the link training as its Tx capabilities.

5 Click **Apply**.

To configure Exerciser to automatically set its Transmitter settings using preset/coefficient requests from DUT

1 In the **Transceiver Settings** tab, ensure that the **Automatic** (As per DUT request) checkbox is selected.

   On selecting this checkbox, the **Tx Output Voltage, Tx Pre-emphasis Pre Tap, Tx Pre-emphasis First Post Tap, Tx Pre-emphasis Second Post Tap** for Exerciser in the **Transceiver Settings** tab will be disabled.

2 Click **Apply**.

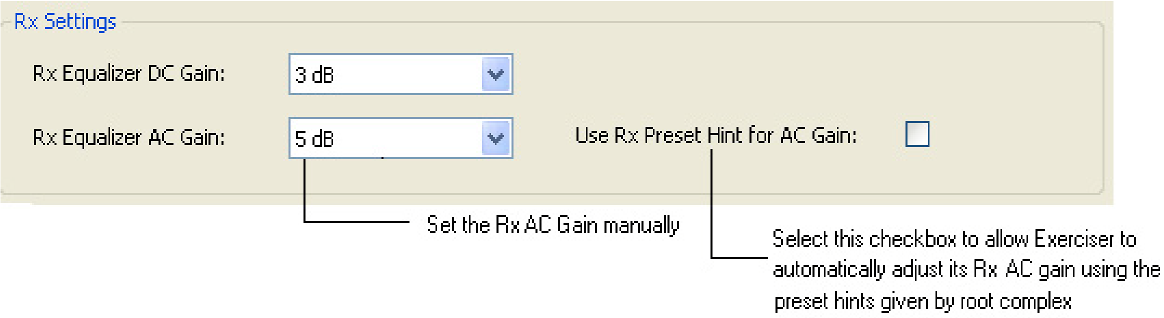
   In the Exerciser group box of the Equalization Settings tab, the **FS** and **LF** values that Exerciser advertises during the link training as its Tx capabilities will also be disabled.

   All the disabled values are auto-computed as per DUT request.

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.

Rx AC gain setting for Exerciser's receiver

If Exerciser is configured as an endpoint (DSC), you can set its Rx AC Gain using one of the two options in the Transceiver Settings tab as displayed in the following screen:


Rx Settings

Rx Equalizer DC Gain: 3 dB

Rx Equalizer AC Gain: 5 dB

Use Rx Preset Hint for AC Gain: □

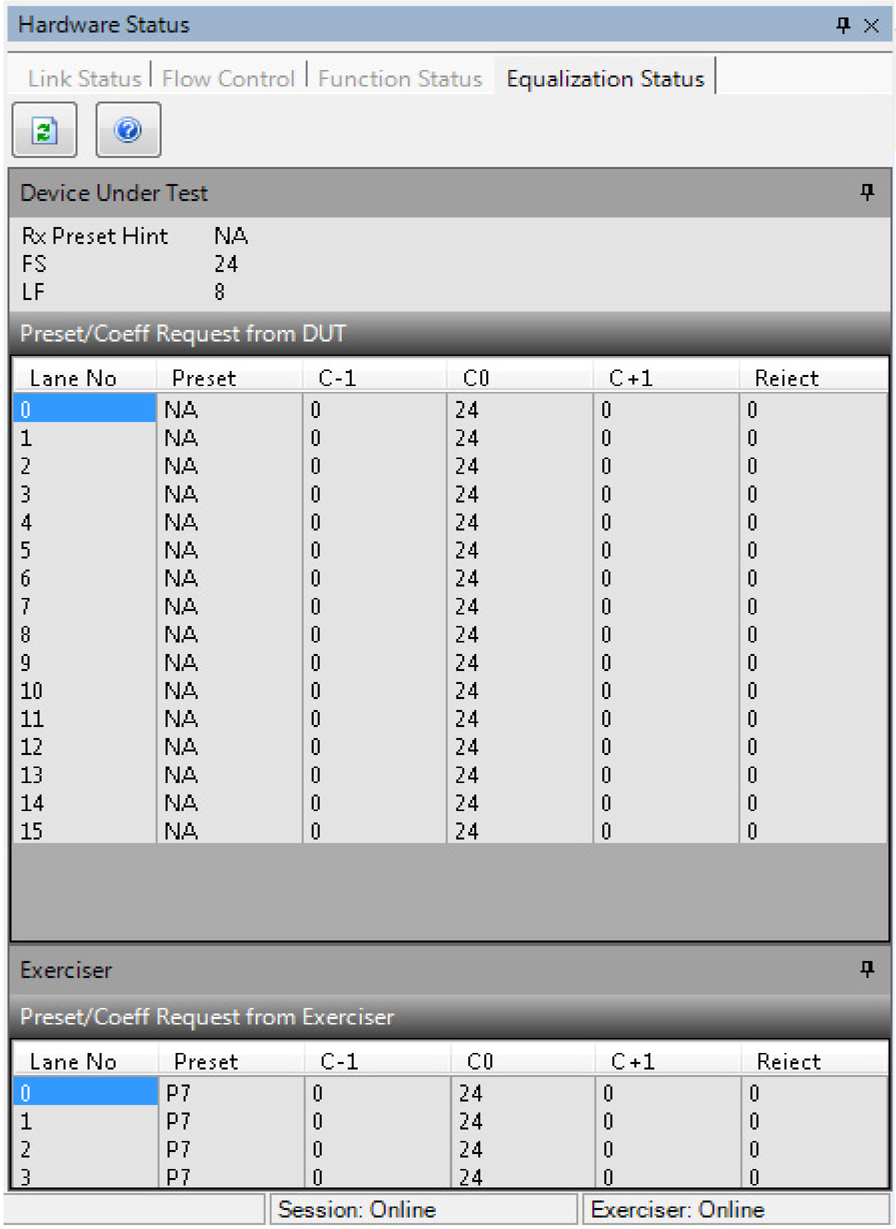
Set the Rx AC Gain manually

Select this checkbox to allow Exerciser to automatically adjust its Rx AC gain using the preset hints given by root complex

If Exerciser is configured as a root complex (USC), then you have to set its Rx AC gain manually using the Rx Equalizer AC Gain listbox. The option to configure Exerciser to automatically adjust its Rx AC gain based on Rx preset hints for AC gain is not available in this case as Rx preset hints are provided only by USC (root complex) to DSC.

Viewing the Status of Equalization Settings of Exerciser and DUT

When a link up is achieved between Exerciser and DUT at 8 GT/s, you can view the status of various equalization related settings that you configured in the Equalization Settings and Transceiver Settings tabs of the Exerciser GUI. This status is displayed in the **Equalization Status** tab of the **Hardware Status** pane.



Device Under Test

Rx Preset Hint	NA
FS	24
LF	8

Preset/Coeff Request from DUT

Lane No	Preset	C-1	C0	C+1	Reiect
0	NA	0	24	0	0
1	NA	0	24	0	0
2	NA	0	24	0	0
3	NA	0	24	0	0
4	NA	0	24	0	0
5	NA	0	24	0	0
6	NA	0	24	0	0
7	NA	0	24	0	0
8	NA	0	24	0	0
9	NA	0	24	0	0
10	NA	0	24	0	0
11	NA	0	24	0	0
12	NA	0	24	0	0
13	NA	0	24	0	0
14	NA	0	24	0	0
15	NA	0	24	0	0

Exerciser

Preset/Coeff Request from Exerciser

Lane No	Preset	C-1	C0	C+1	Reiect
0	P7	0	24	0	0
1	P7	0	24	0	0
2	P7	0	24	0	0
3	P7	0	24	0	0

Session: Online

Exerciser: Online

In the Equalization Status tab, you can view the following equalization settings:

- FS and LF values advertised by DUT
- Rx Preset Hint - The Rx preset hint received from DUT. It displays NA when Exerciser is configured as a USC.
- The final preset/coefficient request made by DUT, along with whether or not Exerciser rejected the request from DUT.

- The final preset/coefficient request made by Exerciser, along with whether or not DUT rejected the request from Exerciser. These requests are as per the automatic co-efficient calculation or manual settings that you configured in the Equalization Settings tab.

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.

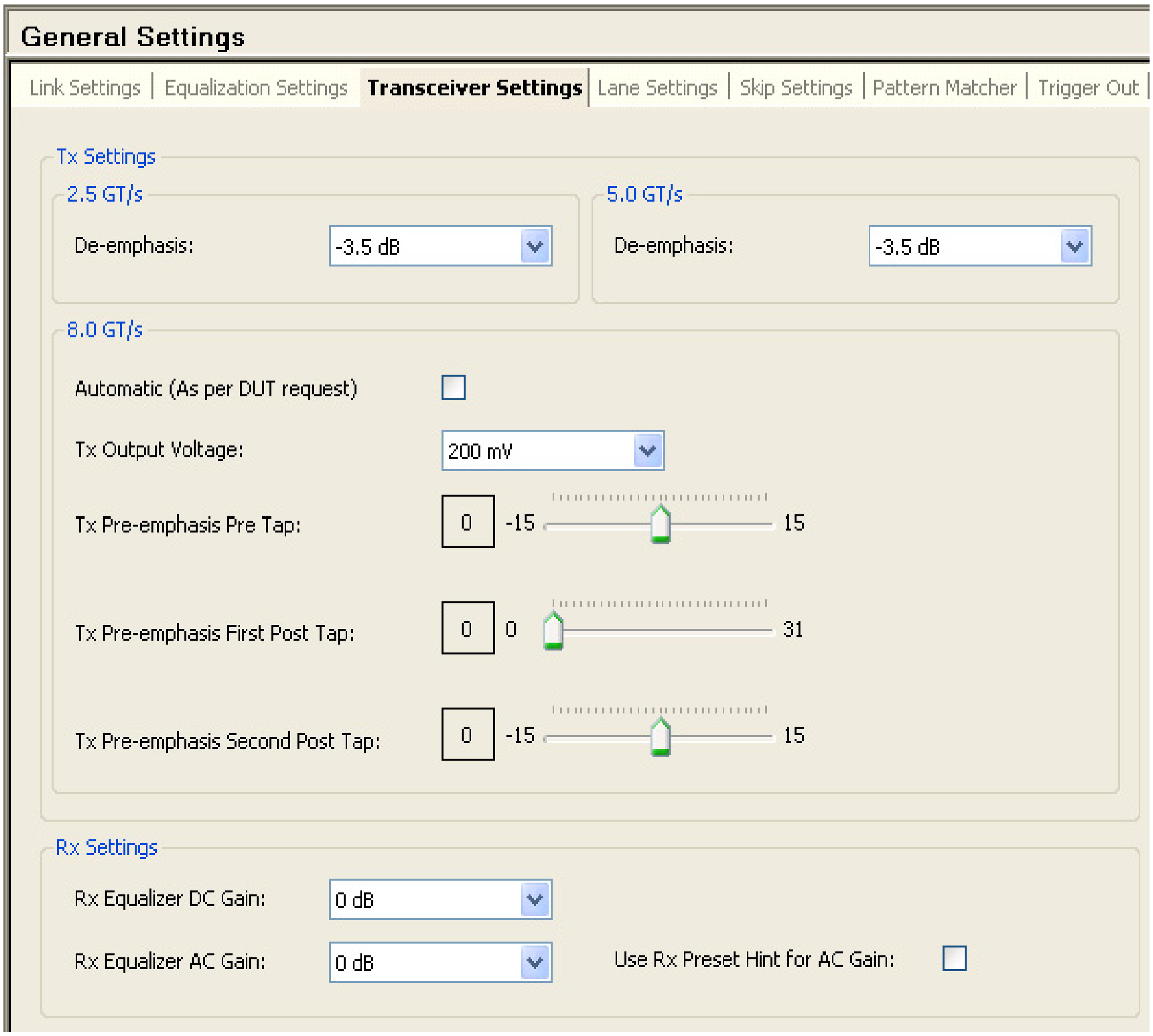
Using the Transceiver Settings tab, you can set the de-emphasis, output voltage, pre tap, first post tap, and second post tap for the Protocol Exerciser transmitter and the AC and DC Gain settings for the Protocol Exerciser Receiver.

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

For 8.0 GT/s, you can either configure the transmitter settings of Exerciser manually or configure Exerciser to automatically set these settings based on the preset/coefficient request received from DUT. The 8.0 GT/s group box in the Transceiver Settings tab provides options for automatic or manual configuration of these settings.

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). You can either configure the Rx Equalizer AC Gain setting for Exerciser manually or configure Exerciser to automatically adjust its Rx AC gain based on the Rx Preset Hint given by DUT (root complex).

The following figure displays the Transceiver settings.



General 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

Automatic (As per DUT request) □

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

Rx Settings

Rx Equalizer DC Gain: 0 dB

Rx Equalizer AC Gain: 0 dB

Use Rx Preset Hint for AC Gain: □

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.

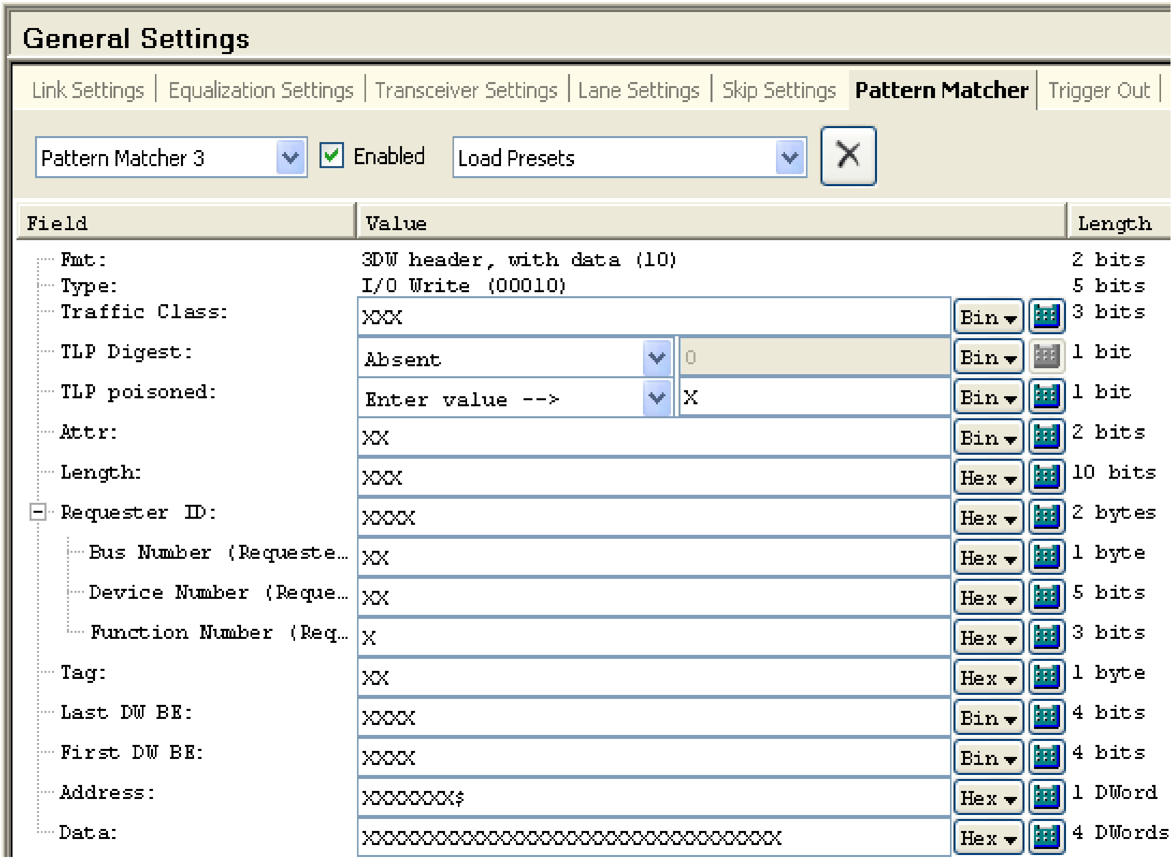
   The fields of the selected TLP template are displayed with the default values.

   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 166 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 66 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 131 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 Algorithmic Payload Generation and Verification Settings

Protocol Exerciser allows you to:

- either use the internal memory of Exerciser to read and write the payload for packets exchanged with DUT.
- or use the Payload Generator and Checker feature to generate and check the payload of these packets algorithmically.

This topic describes how you can use this feature to generate and check payloads algorithmically.

How Payload Generator and Checker work

Payload generator and checker use the same algorithm, which generates the payload data of a packet based on the PCI DWord address of the packet and data generation prefix. You can either set this data generation prefix separately for each physical function in the Algorithmic Payload tab (see page 273) or use the default data generation prefix set for each function.

In each of the generated payload, the upper 11 bits come from this prefix and the rest of the (lower) 21 bits come from the Address field of the packet.

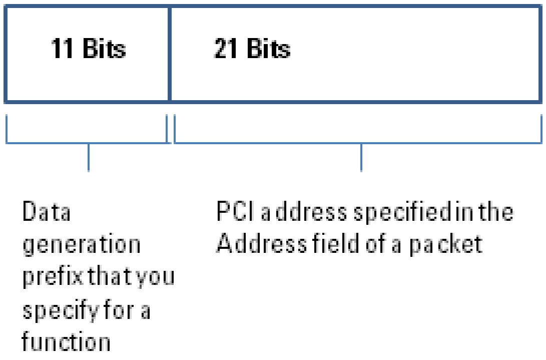
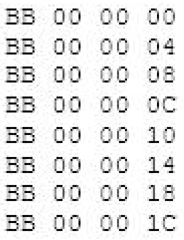

11 Bits
21 Bits
Data generation prefix that you specify for a function
PCI address specified in the Address field of a packet

Figure 13 Figure - Algorithmically generated payload

Following is an example of the subsequent payloads generated algorithmically using a Data Generation prefix and incremental PCI address of packets.



```
BB 00 00 00
BB 00 00 04
BB 00 00 08
BB 00 00 0C
BB 00 00 10
BB 00 00 14
BB 00 00 18
BB 00 00 1C
```

The Payload checker uses the data generation prefix value specified for a function and the PCI address of incoming TLPs to generate the expected payloads for verification of the incoming TLPs for that function. The Payload checker then compares this expected payload with the actual payload of incoming TLPs. If a mismatch is found, it reports it as an error in the Algorithmic Payload tab along with the expected and actual DW and the header of the packet in which the mismatch is found. If no mismatch is found, the status is displayed as No Errors in the Algorithmic Payload tab.

Some packets, such as Config Read/Write packets, do not have the Address field. For such packets, the last 21 bits of the last DW of the packet are used along with the Data Generation prefix to generate the payload.

On choosing the algorithmic payload feature:

- the payloads of the Write request packets sent to DUT are generated algorithmically using the Payload generator feature
- the payload of Read request completions received from DUT are verified using the Payload checker feature.
- the payload of Read request completions sent to DUT are generated algorithmically using the Payload generator feature.
- the payloads of the Write request packets received from DUT are verified using the Payload checker feature.

NOTE The algorithmic payload generation and checker feature is supported only for physical functions of Exerciser. For virtual functions, this feature is not available.

Generating and Verifying Payloads Algorithmically

To generate and verify payloads algorithmically, you need to perform the following tasks:

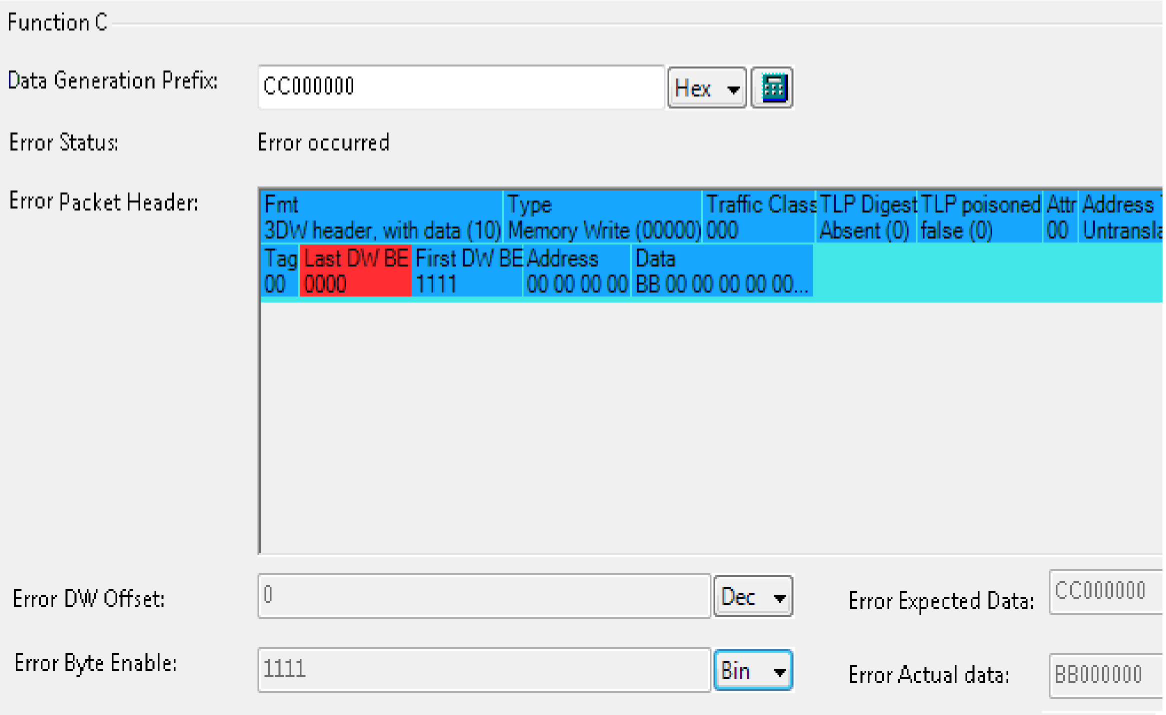
Task	How to...
Configure the Data generation prefix using which the Payload Generator and Checker generates and checks the payloads. This prefix is set separately for each physical function of Exerciser for which you want to generate and verify payloads algorithmically.	Use the **Algorithmic Payload tab** (see page 273) in the General Settings page.
Configure the decoders of the required functions to use payload generator and checker feature to: ▪ generate the completion data for the Read packets received from DUT. ▪ verify the payload of the Write packets received from DUT. These settings are done separately for the decoder of each BAR and Expansion ROM of a function for which you want to generate and verify payloads algorithmically.	Select the **Data Generator** option from the **Resource** listbox in the Decoder Settings page. (see page 267)
Edit the settings of the Write request packets to be sent to DUT so that the payloads of these packets are generated by Payload generator. Edit the settings of the Read request packets to be sent to DUT so that the completions for these requests are verified by the Payload checker. (***Note*** - These settings are applicable only when you are sending PCIe packets as a block transfer.)	For Write requests to be sent to DUT: ▪ Select the **Data Generator** option from the **Read Data From...** section of the "Edit Packet dialog box. For Read requests to be sent to DUT: ▪ Select the **Data Comparator** option from the **Write Data To...** section of the "Edit Packet dialog box.

Viewing the Results of Algorithmic Payload Generation and Verification

After configuring the algorithmic payload settings and starting packet exchange with DUT, you can use the Algorithmic Payload tab to view the results of algorithmic payload verification of packets.

This tab displays whether or not a payload data mismatch occurred while verifying the payload data. It also displays the header of the packet in which payload data mismatch has occurred.

The following screen displays one such Memory Write request packet received from DUT. The expected payload for this packet as generated by Payload checker is CC000000. The actual payload received for this packet is BB00000 resulting in a mismatch. The mismatch was found in the first DW of the actual payload. Therefore, the Error DW Offset is displayed as 0 and the Byte Enable status of this DW is displayed as 1111.



Function C

Data Generation Prefix: CC000000 Hex

Error Status: Error occurred

Error Packet Header:

Fmt	Type	Traffic Class	TLP Digest	TLP poisoned	Attr	Address
3DW header, with data (10)	Memory Write (00000)	000	Absent (0)	false (0)	00	Untransla

Tag	Last DW BE	First DW BE	Address	Data
00	0000	1111	00 00 00 00	BB 00 00 00 00 00...

Error DW Offset: 0 Dec

Error Expected Data: CC000000

Error Byte Enable: 1111 Bin

Error Actual data: BB000000

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**.

By default, Exerciser automatically initiates link recovery if it does not receive any FC update packets for 200us when in L0 or L0s state. However, in certain L0s testing scenarios, you may not want Exerciser to do this. In such situations, you can deselect the Initiate link recovery if no FC update is received for 200us in L0, L0s state checkbox in the Virtual Channel pane to ensure that the link recovery is initiated as per the exit timer value that you specified in the Power Management tab.

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.

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.

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

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

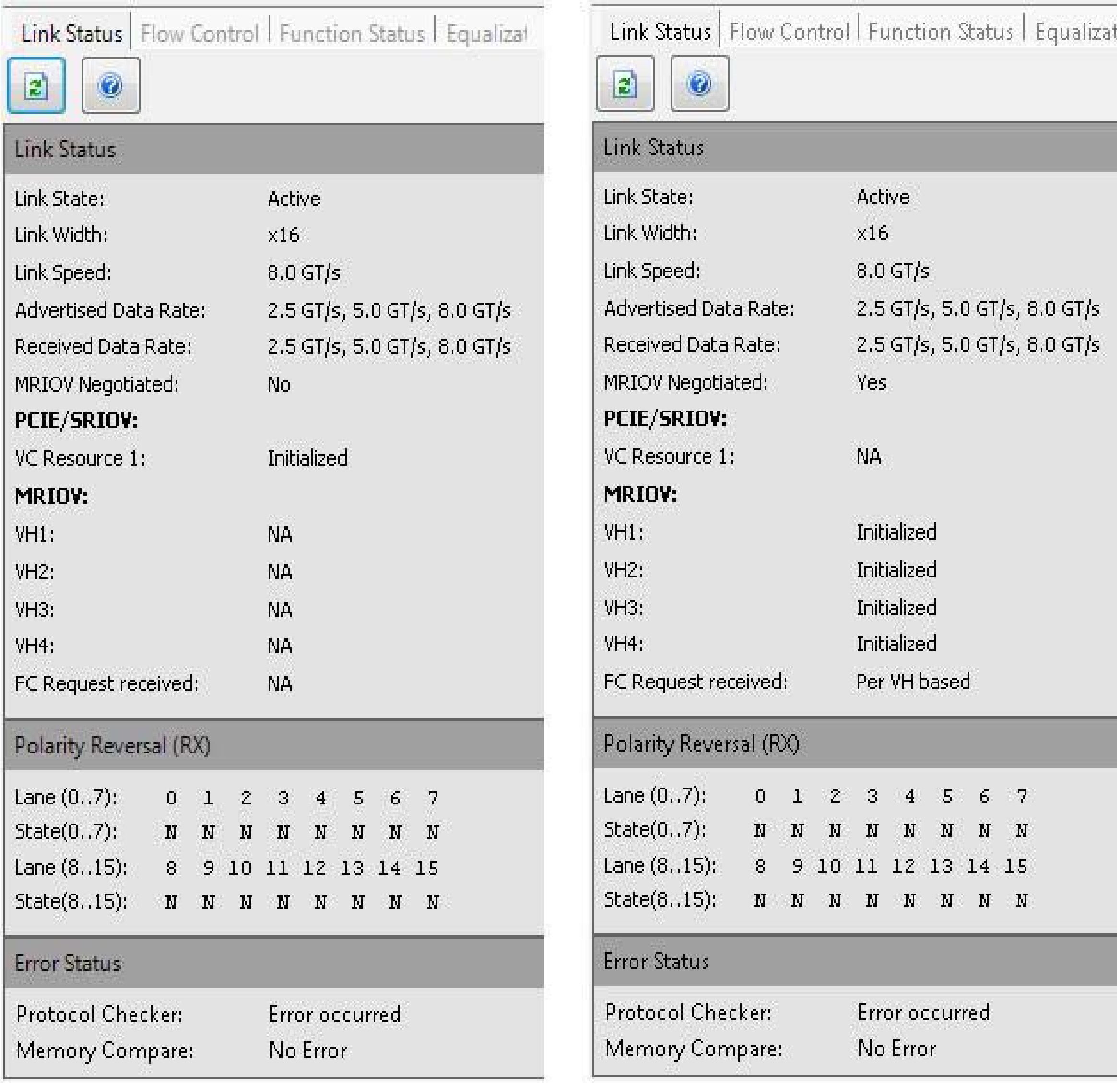
- Make sure you have configured the link, lane, and other settings relevant for link training in a Protocol Exerciser Setup file. You use the General navigation window of the Protocol Exerciser GUI to configure these settings. Refer to the chapter Configuring Protocol Exerciser to know more.
- If you want the link to be MR (Multi-root) enabled, ensure that you select the MRIOV protocol in the **General settings -> Link settings** page. You need appropriate license (see page 261) to get MRIOV capabilities for Exerciser.
- 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 need appropriate license (see page 261) to get 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 81 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 Status

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
MRIOV Negotiated: No
PCIE/SRIOV:
VC Resource 1: 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

Link status for an SRIOV link

Link Status

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
MRIOV Negotiated: Yes
PCIE/SRIOV:
VC Resource 1: NA
MRIOV:
VH1: Initialized
VH2: Initialized
VH3: Initialized
VH4: 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

Error Status

Protocol Checker: Error occurred
Memory Compare: No Error

Link status for an MRIOV link

Figure 14 Hardware Status After Link Up (For SRIOV 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 credits allocated to each VC should be divided for header and payload of Posted and Not Posted requests and Completions. Protocol Exerciser can advertise defined total credits during the flow control initialization process.

If Exerciser emulates a non IOV/SRIOV component, then you can specify advertised credit limits for VC0 and VCx virtual channels. If Exerciser emulates a MRIOV component, then you can specify advertised credit limits for virtual hierarchies and virtual link VL0.

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 295 GUI reference topic.

Initializing flow control for an MRIOV capable component

As an MRIOV capable component, Protocol Exerciser supports five virtual hierarchies (VH0 to VH4) and one virtual link (VL0). 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 and virtual link VL0.

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 along with VL flow control.

This type of flow control is applicable if you have enabled ‘per VH flow control 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 only 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)

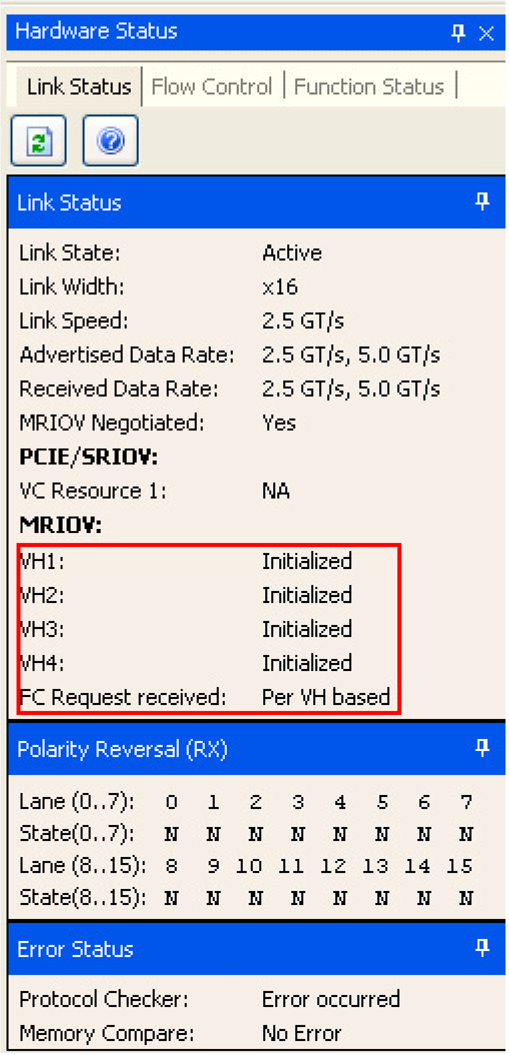
In this type of flow control, all the completions that you added to completion queues (Queue0 to Queue4) are passed through VC0.

To configure the flow control initialization settings for virtual channels of an MRIOV capable Exerciser, refer to the “Virtual Channel function" on page 295 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 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

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

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. For a MRIOV capable Exerciser, the flow control status for virtual hierarchies and virtual link VL0 is displayed. For a non IOV / SRIOV capable Exerciser, the flow control status for virtual channels VC0 and VCx are displayed. Refer to the topic "Starting the Stimulus and Viewing Results" on page 144 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.

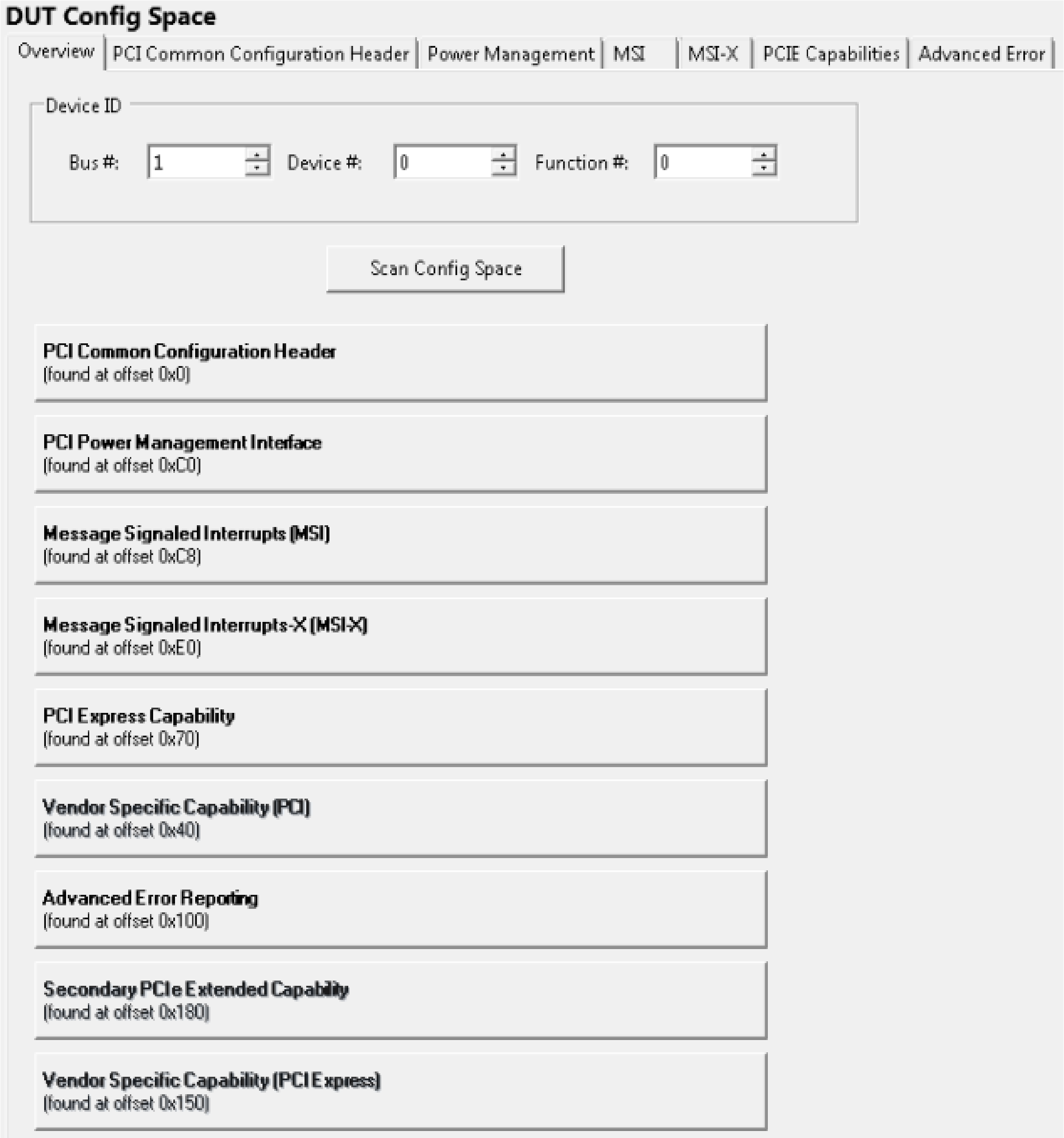
If the link is already at the highest data rate supported by both the link partners and you want to downgrade the link speed, then you first need to downgrade the data rate capabilities of a link partner before starting speed change. In such a situation, the link speed is downgraded based on the changed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

When Exerciser emulates a root complex, you may want to or at times need to examine the DUT capabilities and also edit some of these capabilities to set up the DUT appropriately. To accomplish this, Exerciser provides you the option to view and edit the configuration space of the DUT using the **DUT Config Space** page of the Protocol Exerciser GUI.

For viewing the configuration space of the DUT, you can initiate the DUT configuration space scan. On doing so, Exerciser reads the DUT capabilities from various registers and updates the DUT Config Space page with the scanned results. After the configuration space scan, the offset values at which DUT capabilities are found are displayed.



Bus #: 1 Device #: 0 Function #: 0

PCI Common Configuration Header
(found at offset 0x0)

PCI Power Management Interface
(found at offset 0xC0)

Message Signaled Interrupts (MSI)
(found at offset 0xC8)

Message Signaled Interrupts-X (MSI-X)
(found at offset 0xE0)

PCI Express Capability
(found at offset 0x70)

Vendor Specific Capability (PCI)
(found at offset 0x40)

Advanced Error Reporting
(found at offset 0x100)

Secondary PCIe Extended Capability
(found at offset 0x180)

Vendor Specific Capability (PCI Express)
(found at offset 0x150)

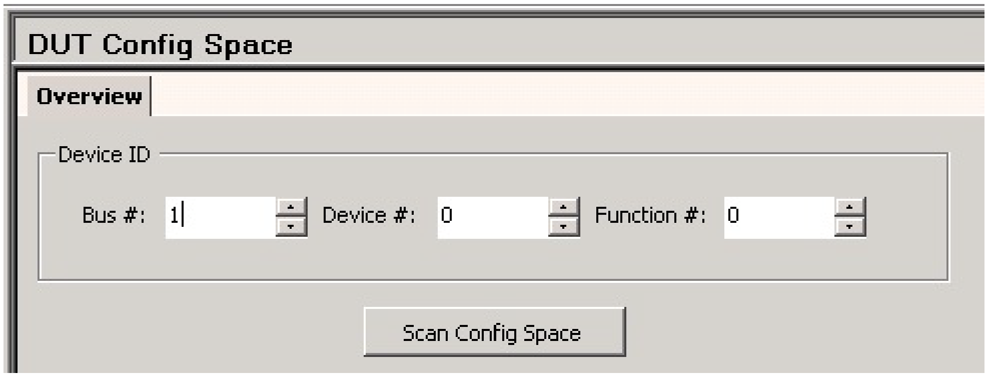
In the previous releases of Exerciser, only for few capabilities such as MSI and MSI-X, register values were retrieved and displayed and were available for editing. For other capabilities, only the offset value was displayed. With the PCIe Exerciser 8.72 release, few additional capabilities, such as Power Management, PCIe Capability, and Advance Error Reporting Capability have also been enabled for viewing and editing. The capabilities that you edit in the DUT Config Space page are written to the appropriate registers in the DUT configuration space.

To scan and edit DUT configuration space

1 Ensure that Exerciser is connected to DUT and the PCIe link between the Exerciser and DUT is active to allow Exerciser to read DUT's configuration space.

2 Click **DUT Config Space** from the left navigation pane.

The **DUT Config Space** page is displayed.

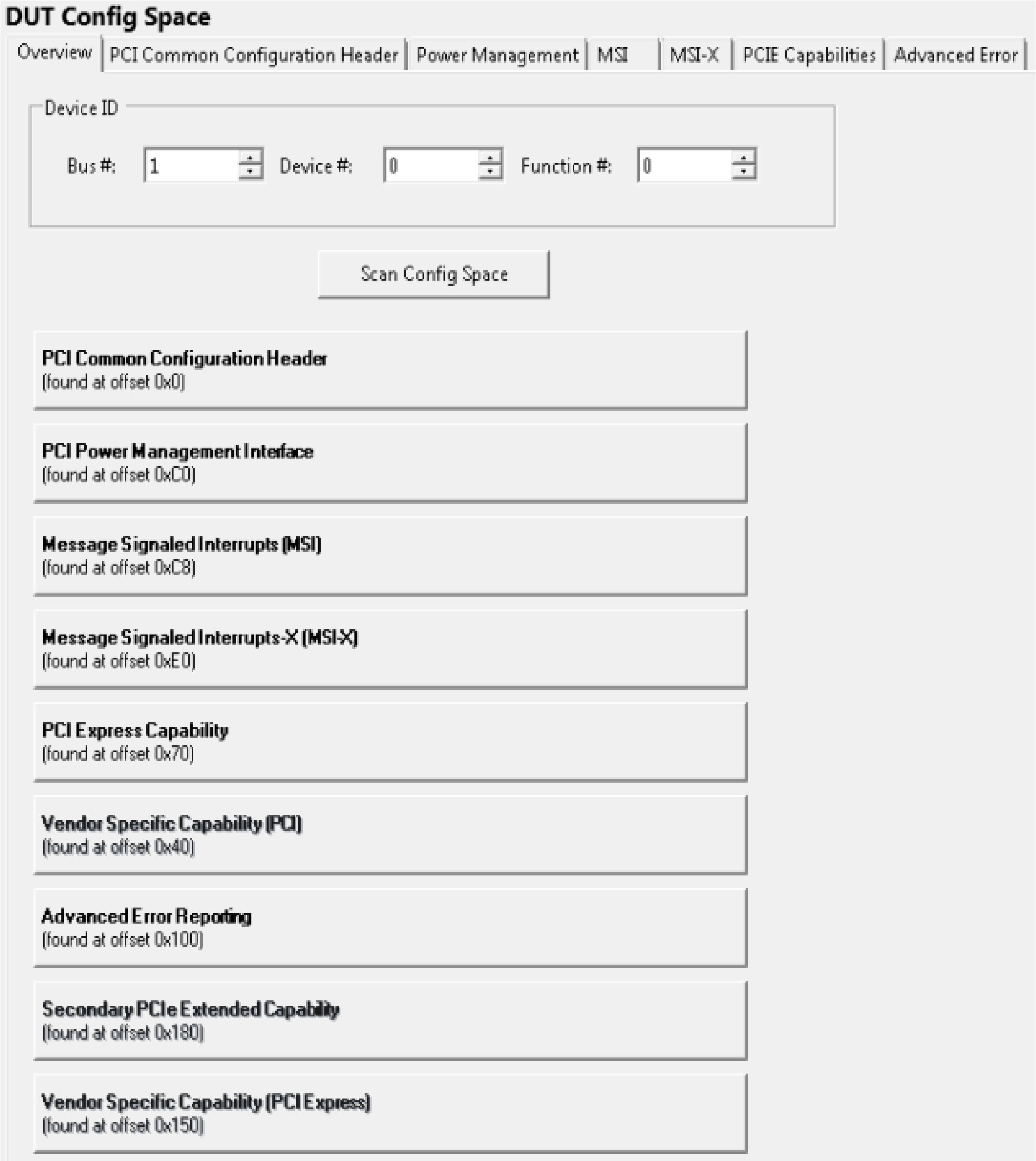


Bus #: 1 Device #: 0 Function #: 0

3 In the **Overview** tab, specify/verify the DUT's Bus number, Device number and Function number to be used.

4 Click the **Scan Config Space** button.

The offset values at which DUT capabilities are found are displayed.



Bus #: 1 Device #: 0 Function #: 0

PCI Common Configuration Header
(found at offset 0x0)

PCI Power Management Interface
(found at offset 0xC0)

Message Signaled Interrupts (MSI)
(found at offset 0xC8)

Message Signaled Interrupts-X (MSI-X)
(found at offset 0xE0)

PCI Express Capability
(found at offset 0x70)

Vendor Specific Capability (PCI)
(found at offset 0x40)

Advanced Error Reporting
(found at offset 0x100)

Secondary PCIe Extended Capability
(found at offset 0x180)

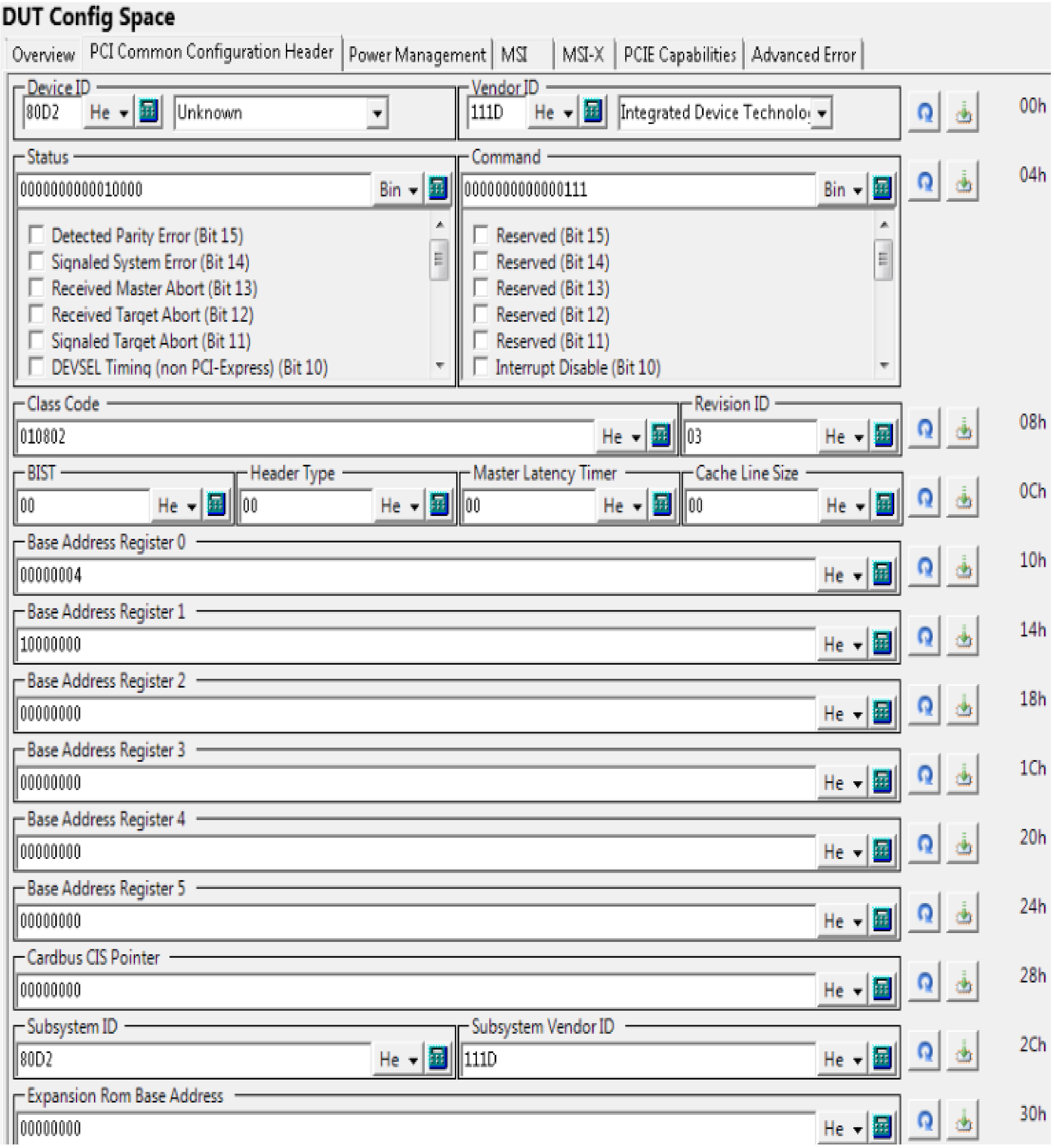
Vendor Specific Capability (PCI Express)
(found at offset 0x150)

5 From the displayed list of capabilities, click a capability that you want to edit. Capabilities that are displayed enabled are editable.

The tab page for the selected capability is displayed with values retrieved from the applicable capability registers. Let's have a glance at the capabilities available in 8.74 release of PCIe Exerciser.

PCI Common Configuration Header

If you click PCI Common Configuration Header Capability to edit its values, following screen appears:



DUT Config Space

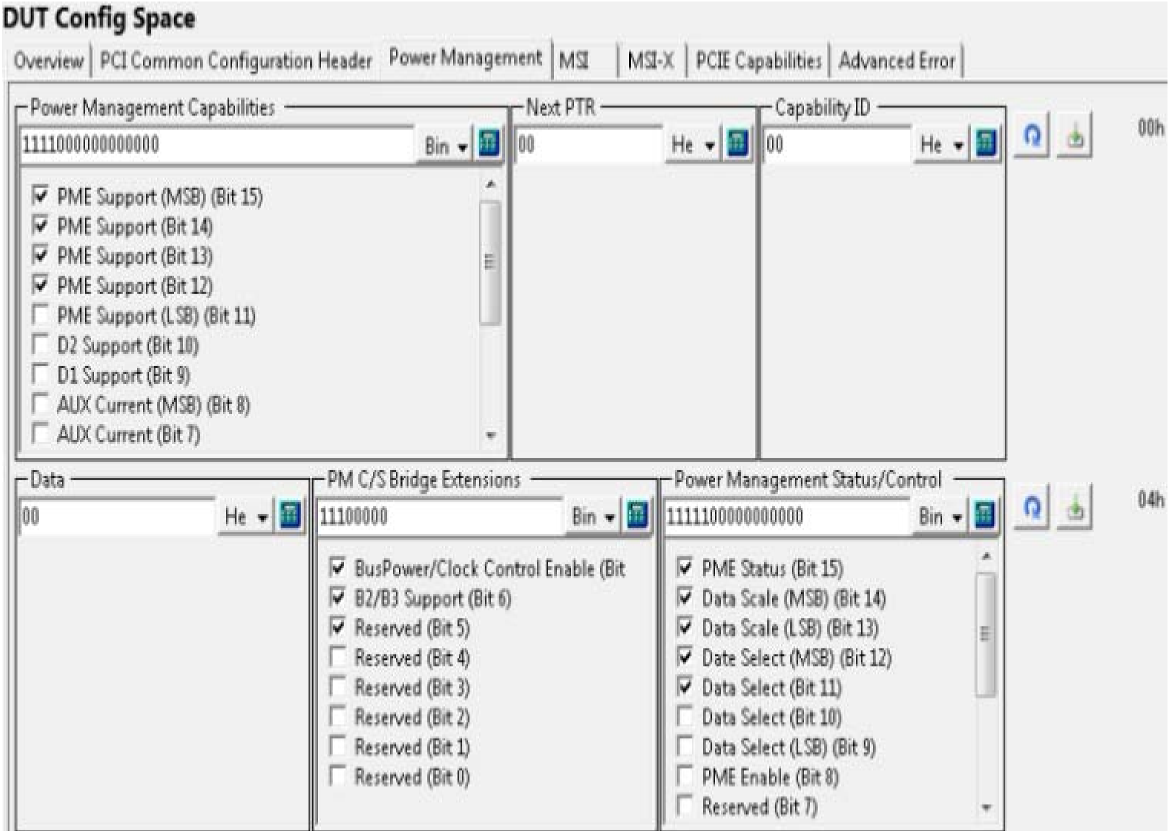
Device ID	80D2 He	Unknown	00h
Vendor ID	111D He	Integrated Device Technolo	
Status	0000000000010000 Bin		04h
Command	0000000000000111 Bin		
Class Code	010802 He		08h
Revision ID	03 He		
BIST	00 He		0Ch
Header Type	00 He		
Master Latency Timer	00 He		
Cache Line Size	00 He		
Base Address Register 0	00000004 He		10h
Base Address Register 1	10000000 He		14h
Base Address Register 2	00000000 He		18h
Base Address Register 3	00000000 He		1Ch
Base Address Register 4	00000000 He		20h
Base Address Register 5	00000000 He		24h
Cardbus CIS Pointer	00000000 He		28h
Subsystem ID	80D2 He		2Ch
Subsystem Vendor ID	111D He		
Expansion Rom Base Address	00000000 He		30h

Status:
- □ Detected Parity Error (Bit 15)
- □ Signaled System Error (Bit 14)
- □ Received Master Abort (Bit 13)
- □ Received Target Abort (Bit 12)
- □ Signaled Target Abort (Bit 11)
- □ DEVSEL Timing (non PCI-Express) (Bit 10)

Command:
- □ Reserved (Bit 15)
- □ Reserved (Bit 14)
- □ Reserved (Bit 13)
- □ Reserved (Bit 12)
- □ Reserved (Bit 11)
- □ Interrupt Disable (Bit 10)

Power Management

If you click Power Management Capability to edit its values, following screen appears:



Power Management Capabilities: 1111000000000000 Bin

- ☑ PME Support (MSB) (Bit 15)
- ☑ PME Support (Bit 14)
- ☑ PME Support (Bit 13)
- ☑ PME Support (Bit 12)
- ☐ PME Support (LSB) (Bit 11)
- ☐ D2 Support (Bit 10)
- ☐ D1 Support (Bit 9)
- ☐ AUX Current (MSB) (Bit 8)
- ☐ AUX Current (Bit 7)

Next PTR: 00 He

Capability ID: 00 He

Data: 00 He

PM C/S Bridge Extensions: 11100000 Bin

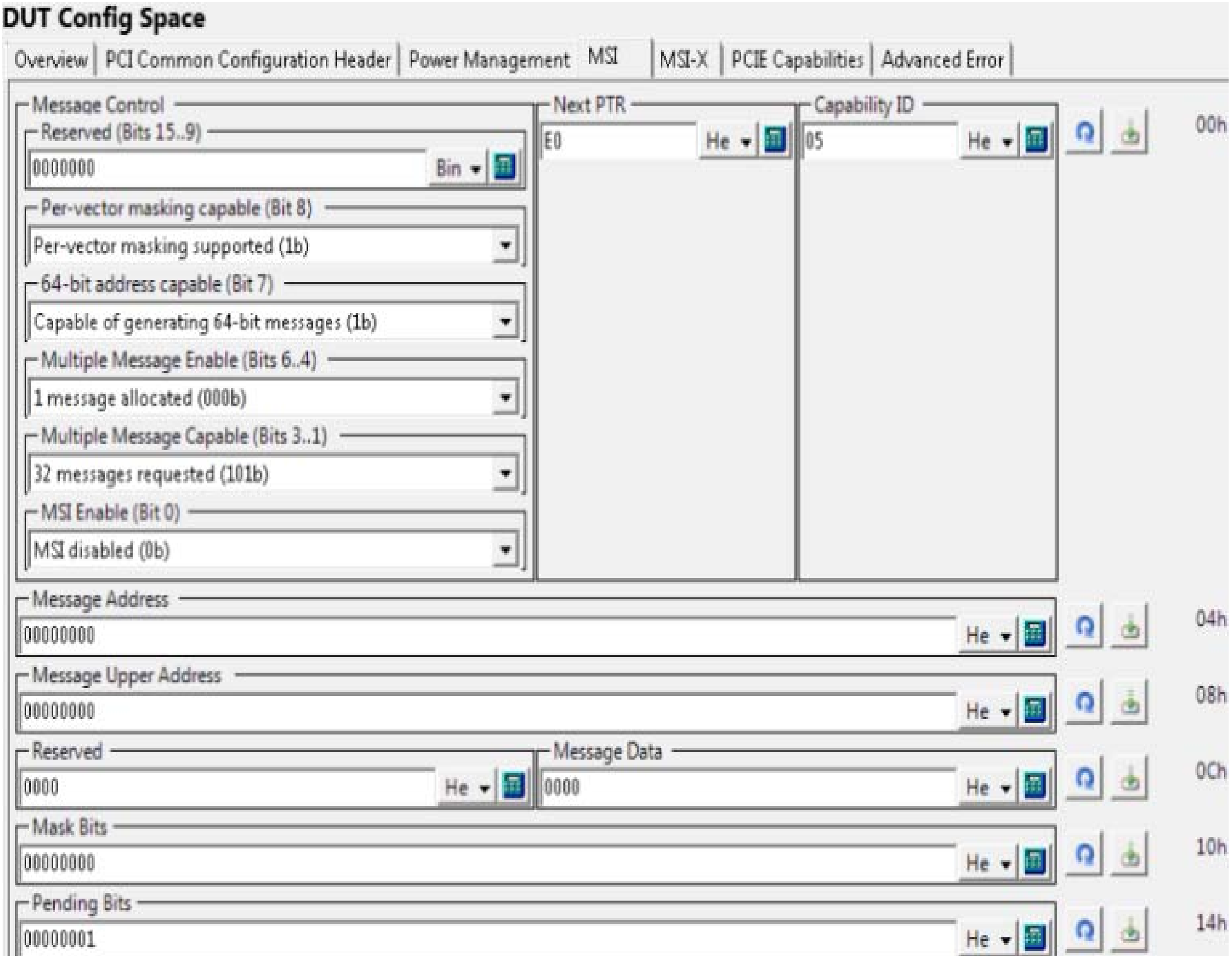
- ☑ BusPower/Clock Control Enable (Bit
- ☑ B2/B3 Support (Bit 6)
- ☑ Reserved (Bit 5)
- ☐ Reserved (Bit 4)
- ☐ Reserved (Bit 3)
- ☐ Reserved (Bit 2)
- ☐ Reserved (Bit 1)
- ☐ Reserved (Bit 0)

Power Management Status/Control: 1111100000000000 Bin

- ☑ PME Status (Bit 15)
- ☑ Data Scale (MSB) (Bit 14)
- ☑ Data Scale (LSB) (Bit 13)
- ☑ Date Select (MSB) (Bit 12)
- ☑ Data Select (Bit 11)
- ☐ Data Select (Bit 10)
- ☐ Data Select (LSB) (Bit 9)
- ☐ PME Enable (Bit 8)
- ☐ Reserved (Bit 7)

MSI

If you click MSI Capability to edit its values, following screen appears:



DUT Config Space

Message Control

Reserved (Bits 15..9): 0000000

Per-vector masking capable (Bit 8): Per-vector masking supported (1b)

64-bit address capable (Bit 7): Capable of generating 64-bit messages (1b)

Multiple Message Enable (Bits 6..4): 1 message allocated (000b)

Multiple Message Capable (Bits 3..1): 32 messages requested (101b)

MSI Enable (Bit 0): MSI disabled (0b)

Next PTR: E0

Capability ID: 05

Message Address: 00000000

Message Upper Address: 00000000

Reserved: 0000

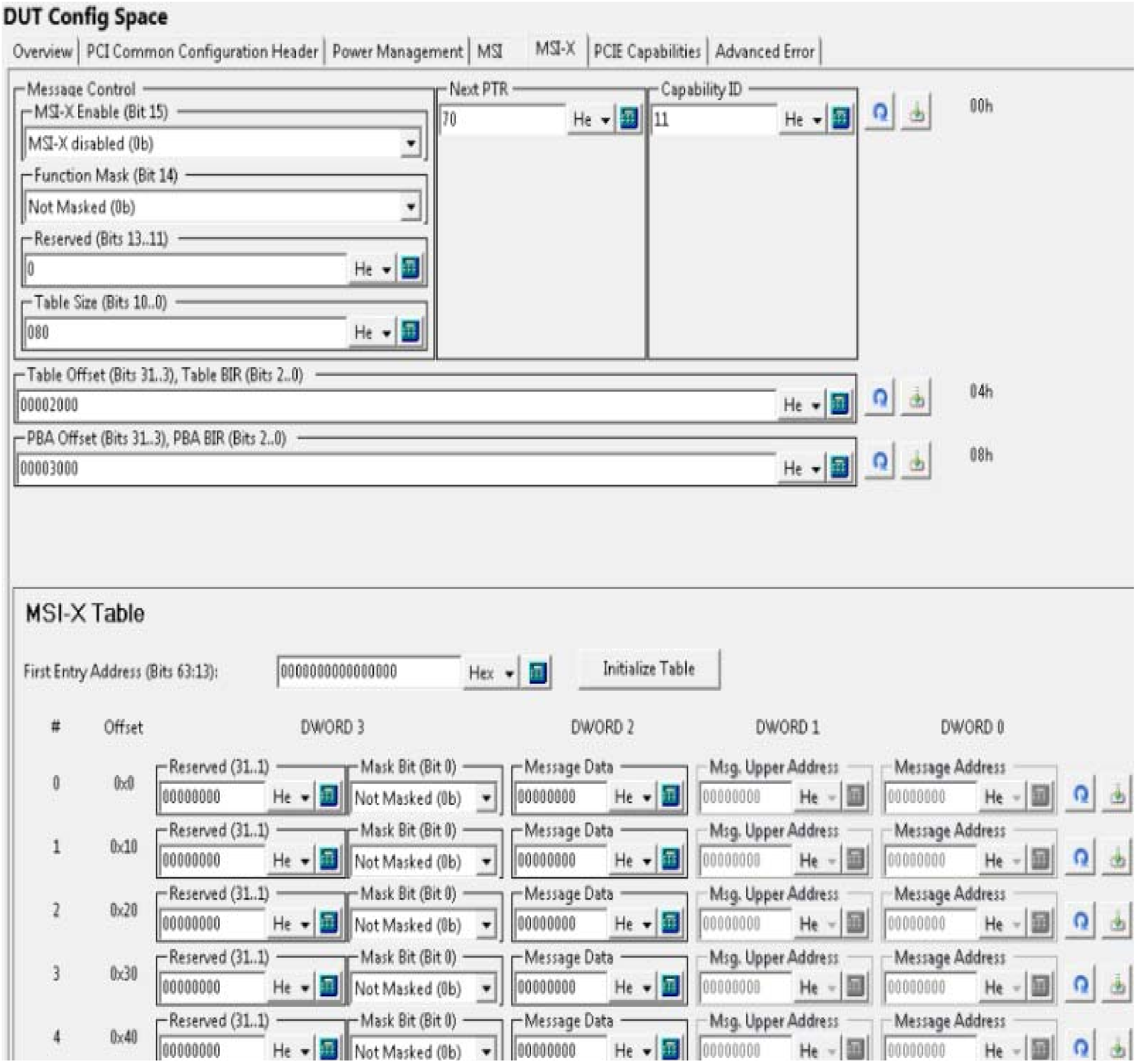
Message Data: 0000

Mask Bits: 00000000

Pending Bits: 00000001

MSI-X

If you click MSI-X Capability to edit its values, following screen appears:



DUT Config Space

Message Control

MSI-X Enable (Bit 15): MSI-X disabled (0b)

Function Mask (Bit 14): Not Masked (0b)

Reserved (Bits 13..11): 0 He

Table Size (Bits 10..0): 080 He

Next PTR: 70 He

Capability ID: 11 He

00h

Table Offset (Bits 31..3), Table BIR (Bits 2..0): 00002000 He

04h

PBA Offset (Bits 31..3), PBA BIR (Bits 2..0): 00003000 He

08h

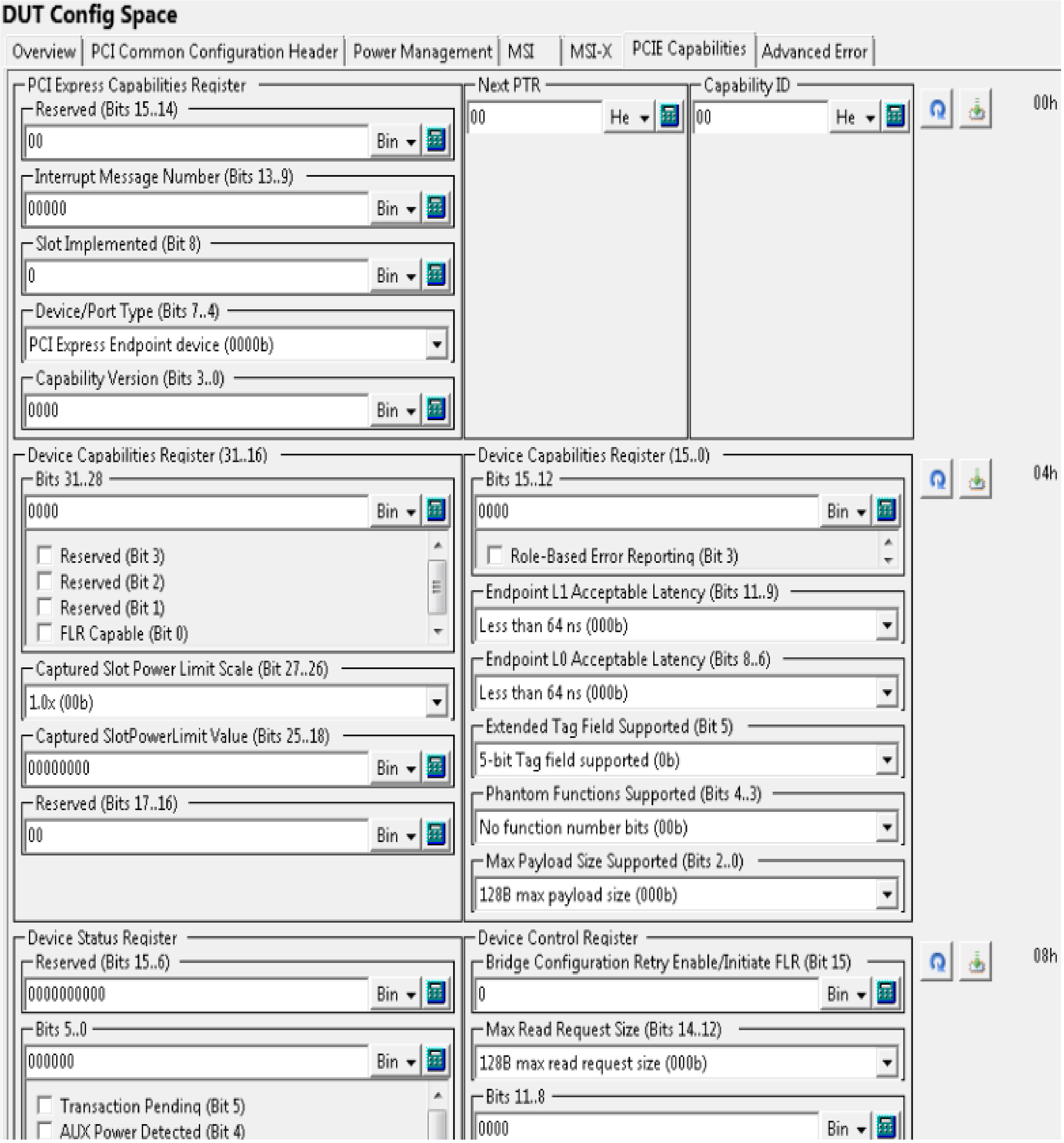
MSI-X Table

First Entry Address (Bits 63:13): 0000000000000000 Hex

#	Offset	DWORD 3 Reserved (31..1)	DWORD 3 Mask Bit (Bit 0)	DWORD 2 Message Data	DWORD 1 Msg. Upper Address	DWORD 0 Message Address
0	0x0	00000000 He	Not Masked (0b)	00000000 He	00000000 He	00000000 He
1	0x10	00000000 He	Not Masked (0b)	00000000 He	00000000 He	00000000 He
2	0x20	00000000 He	Not Masked (0b)	00000000 He	00000000 He	00000000 He
3	0x30	00000000 He	Not Masked (0b)	00000000 He	00000000 He	00000000 He
4	0x40	00000000 He	Not Masked (0b)	00000000 He	00000000 He	00000000 He

PCIe Capabilities

If you click PCIe Capability, the PCIe capabilities page is displayed as shown below:



DUT Config Space

PCI Express Capabilities Register
- Reserved (Bits 15..14): 00 Bin
- Interrupt Message Number (Bits 13..9): 00000 Bin
- Slot Implemented (Bit 8): 0 Bin
- Device/Port Type (Bits 7..4): PCI Express Endpoint device (0000b)
- Capability Version (Bits 3..0): 0000 Bin

Next PTR: 00 He

Capability ID: 00 He

00h

Device Capabilities Register (31..16)
- Bits 31..28: 0000 Bin
  - □ Reserved (Bit 3)
  - □ Reserved (Bit 2)
  - □ Reserved (Bit 1)
  - □ FLR Capable (Bit 0)
- Captured Slot Power Limit Scale (Bit 27..26): 1.0x (00b)
- Captured SlotPowerLimit Value (Bits 25..18): 00000000 Bin
- Reserved (Bits 17..16): 00 Bin

Device Capabilities Register (15..0)
- Bits 15..12: 0000 Bin
  - □ Role-Based Error Reporting (Bit 3)
- Endpoint L1 Acceptable Latency (Bits 11..9): Less than 64 ns (000b)
- Endpoint L0 Acceptable Latency (Bits 8..6): Less than 64 ns (000b)
- Extended Tag Field Supported (Bit 5): 5-bit Tag field supported (0b)
- Phantom Functions Supported (Bits 4..3): No function number bits (00b)
- Max Payload Size Supported (Bits 2..0): 128B max payload size (000b)

04h

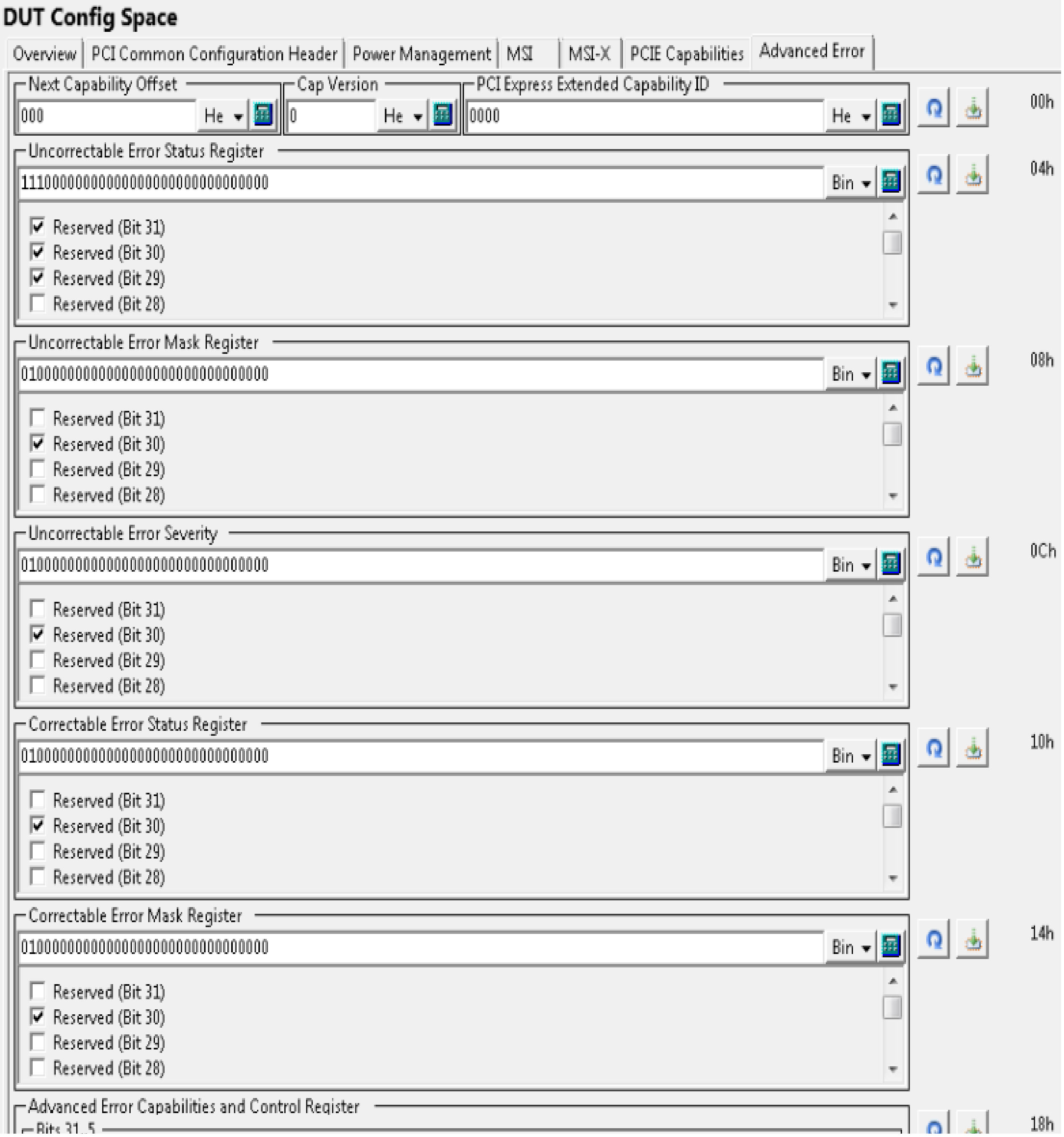
Device Status Register
- Reserved (Bits 15..6): 0000000000 Bin
- Bits 5..0: 000000 Bin
  - □ Transaction Pending (Bit 5)
  - □ AUX Power Detected (Bit 4)

Device Control Register
- Bridge Configuration Retry Enable/Initiate FLR (Bit 15): 0 Bin
- Max Read Request Size (Bits 14..12): 128B max read request size (000b)
- Bits 11..8: 0000 Bin

08h

Ad vance Error Reporting

If you click Advance Error Reporting Capability, the Advance Error capabilities page is displayed:



DUT Config Space

Overview | PCI Common Configuration Header | Power Management | MSI | MSI-X | PCIE Capabilities | Advanced Error

00h
- Next Capability Offset: 000 (He)
- Cap Version: 0 (He)
- PCI Express Extended Capability ID: 0000 (He)

04h Uncorrectable Error Status Register: 11100000000000000000000000000000 (Bin)
- ☑ Reserved (Bit 31)
- ☑ Reserved (Bit 30)
- ☑ Reserved (Bit 29)
- ☐ Reserved (Bit 28)

08h Uncorrectable Error Mask Register: 01000000000000000000000000000000 (Bin)
- ☐ Reserved (Bit 31)
- ☑ Reserved (Bit 30)
- ☐ Reserved (Bit 29)
- ☐ Reserved (Bit 28)

0Ch Uncorrectable Error Severity: 01000000000000000000000000000000 (Bin)
- ☐ Reserved (Bit 31)
- ☑ Reserved (Bit 30)
- ☐ Reserved (Bit 29)
- ☐ Reserved (Bit 28)

10h Correctable Error Status Register: 01000000000000000000000000000000 (Bin)
- ☐ Reserved (Bit 31)
- ☑ Reserved (Bit 30)
- ☐ Reserved (Bit 29)
- ☐ Reserved (Bit 28)

14h Correctable Error Mask Register: 01000000000000000000000000000000 (Bin)
- ☐ Reserved (Bit 31)
- ☑ Reserved (Bit 30)
- ☐ Reserved (Bit 29)
- ☐ Reserved (Bit 28)

18h Advanced Error Capabilities and Control Register
- Bits 31..5

6 To read the current values of a register from DUT, click the ↻ button displayed with that register.

7 You can click the **Reload** ↻ toolbar button to read the current values of all the registers displayed in a tab of DUT Config Space.

8 Make the required edits to the displayed capability registers.

9 To write the changes to an individual DUT register, click the button displayed with that register.

10 To write all the changes that you made to various DUT registers displayed in a tab of DUT Config Space, click **Apply**.

Keysight Protocol Exerciser for PCI Express

User Guide

6 Testing DUT’s LTSSM Functions

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 Keysight 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.

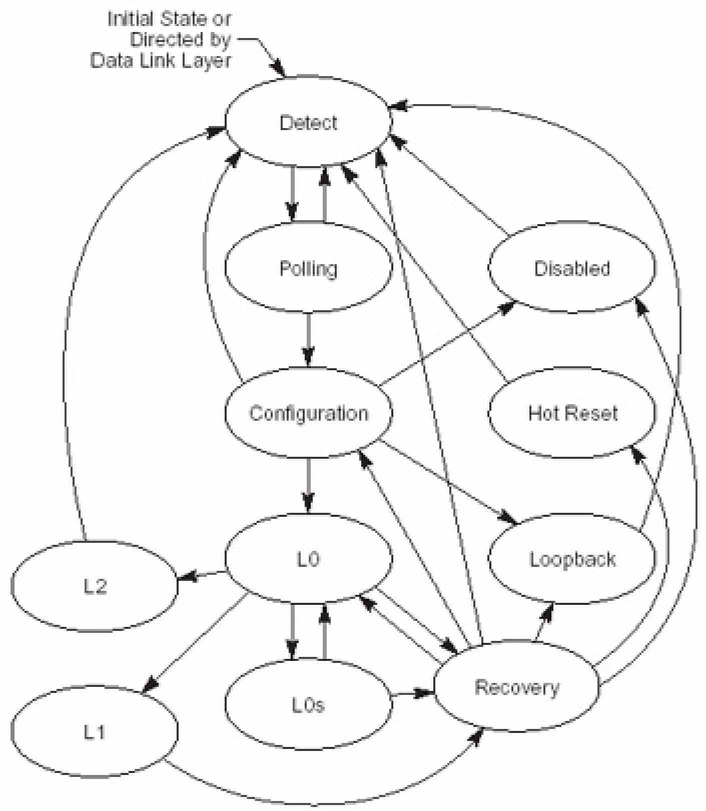


Figure 15 LTSSM States

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.

Keysight 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 Keysight'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, x2, x4, x8 to x16 (any combinations allowed).
- 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.

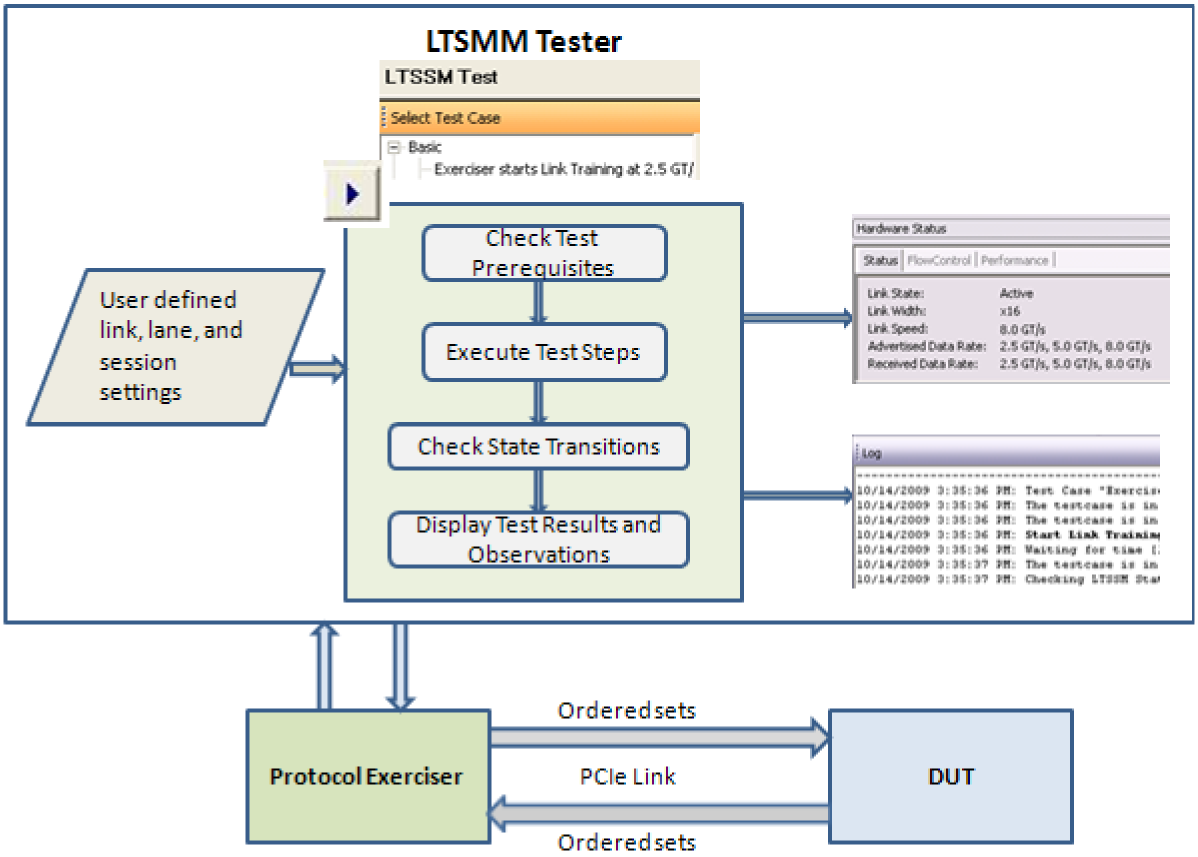
All the LTSSM tests are available in the LTSSM Tests tab of the Protocol Exerciser GUI. When a user runs an LTSSM test, the LTSSM Tester tool executes that test's steps taking all the required link, lane, equalization, and other settings as inputs. The test execution and observations depend on whether 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.



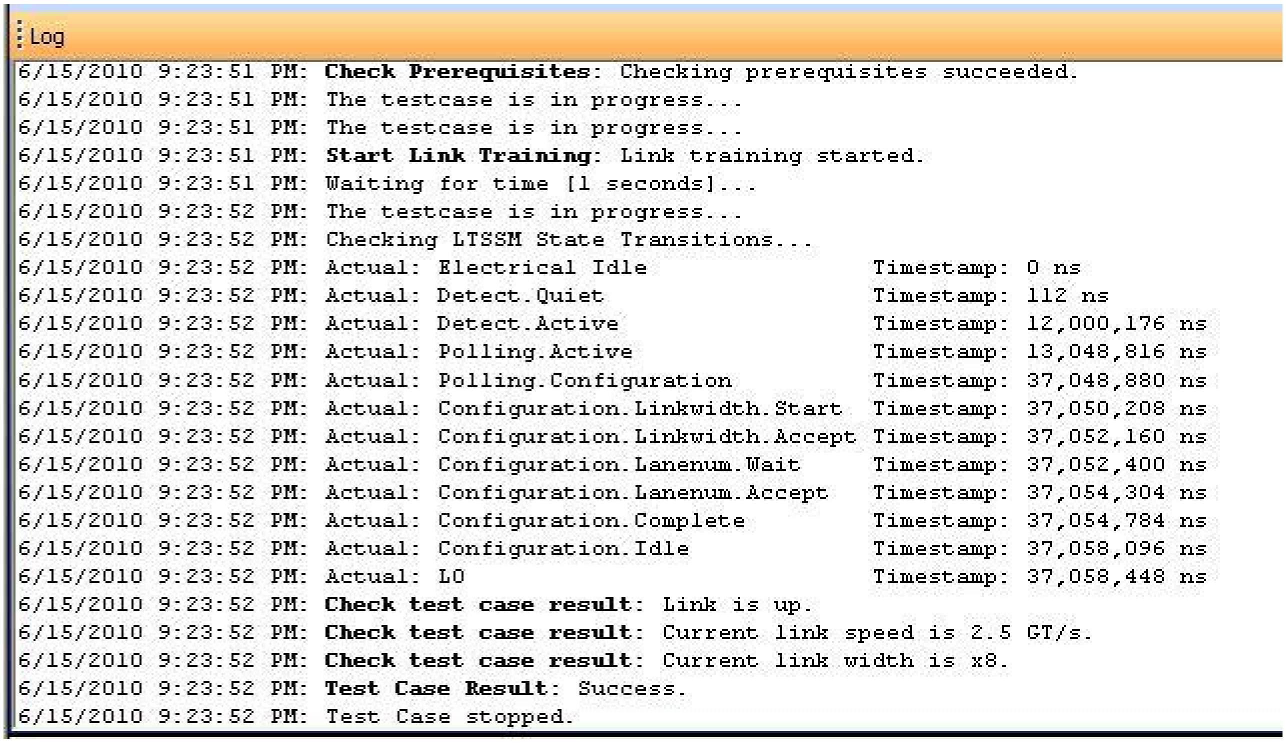
Link State: Active
Link Width: x16
Link Speed: 8.0 GT/s
Advertised Data Rate: 2.5 GT/s, 5.0 GT/s, 8.0 GT/s
Received Data Rate: 2.5 GT/s, 5.0 GT/s, 8.0 GT/s

```
-----------------------------------------------------------
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 Trainin
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 16 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 17 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

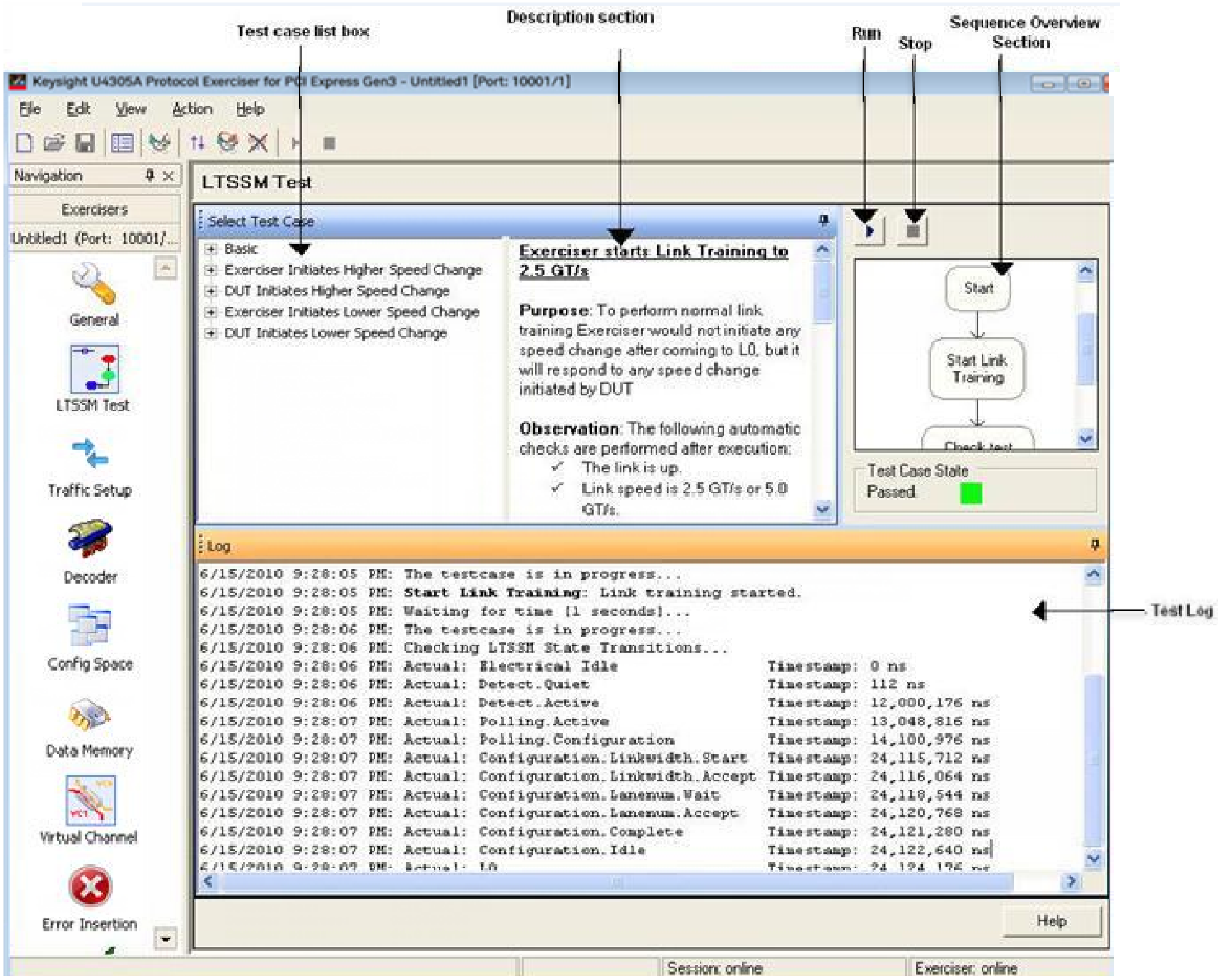
You can access LTSSM Tester from the Navigation window of the Protocol Exerciser GUI.

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.



Test case list box
Description section
Run
Stop
Sequence Overview Section
Test Log
Keysight U4305A Protocol Exerciser for PCI Express Gen3 - Untitled1 [Port: 10001/1]
File Edit View Action Help
Navigation
Exercisers
Untitled1 (Port: 10001/...
General
LTSSM Test
Traffic Setup
Decoder
Config Space
Data Memory
Virtual Channel
Error Insertion
LTSSM Test
Select Test Case
Basic
Exerciser Initiates Higher Speed Change
DUT Initiates Higher Speed Change
Exerciser Initiates Lower Speed Change
DUT Initiates Lower Speed Change
Start
Start Link Training
Check test
Test Case State
Passed 🟩
Log
Help
Session: online
Exerciser: online

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
6/15/2010 9:28:07 PM: Actual: Configuration.Idle                Timestamp: 24,122,640 ns
6/15/2010 9:28:07 PM: Actual: L0                                Timestamp: 24,124,176 ns
```

Figure 18 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 111 ▪ "Link training to 8.0 GT/s with all the phases of Equalization test" on page 112 ▪ "Link training to 8.0 GT/s with the first phase of Equalization test" on page 114 ▪ "Exerciser Initiates Transition to Recovery Test" on page 119 ▪ "Capture State Transitions Test" on page 120
Link Speed Upgrade tests	▪ "Link Speed Change With Equalization Tests" on page 118 ▪ "Link Speed Change Without Equalization Tests" on page 115
Link Speed Downgrade Tests	▪ "Link Speed Change With Equalization Tests" on page 118 ▪ "Link Speed Change Without Equalization Tests" on page 115

Test	Description
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.
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.

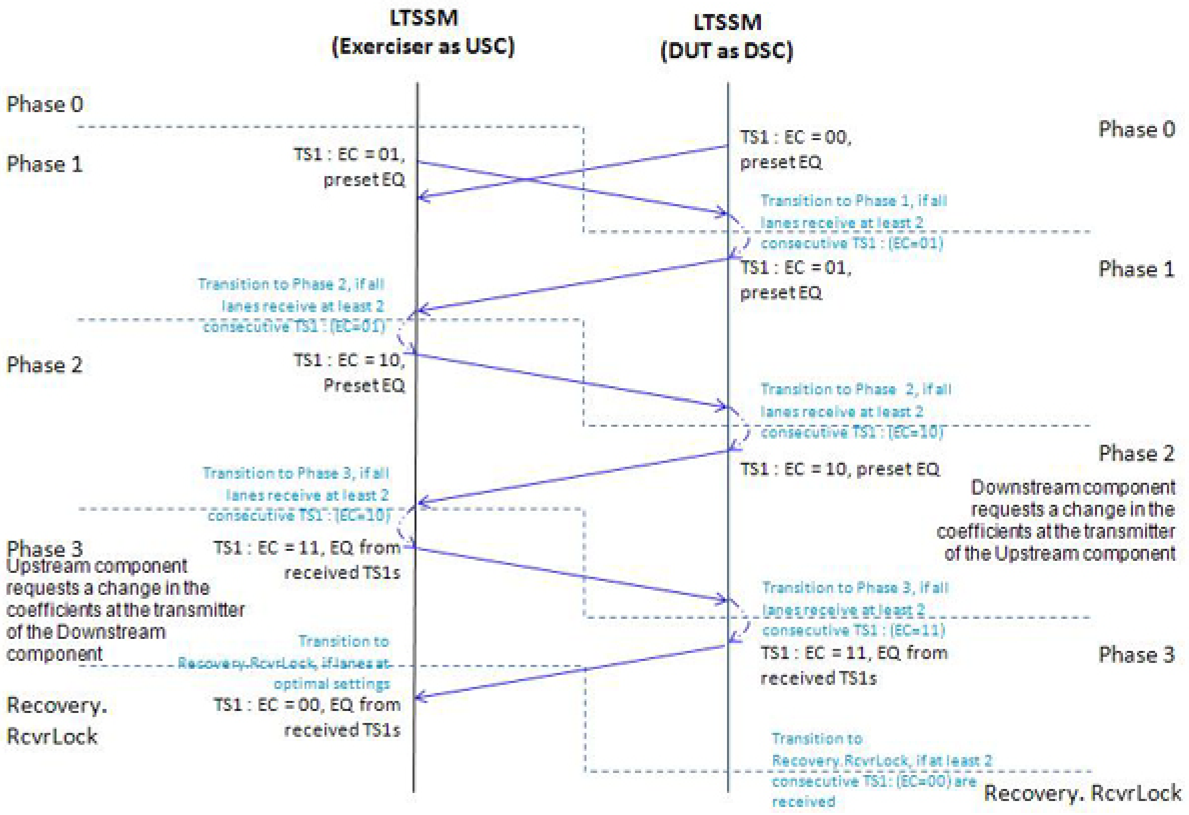
Usage

- 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)
TS1 : EC = 10, preset EQ
Phase 2
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

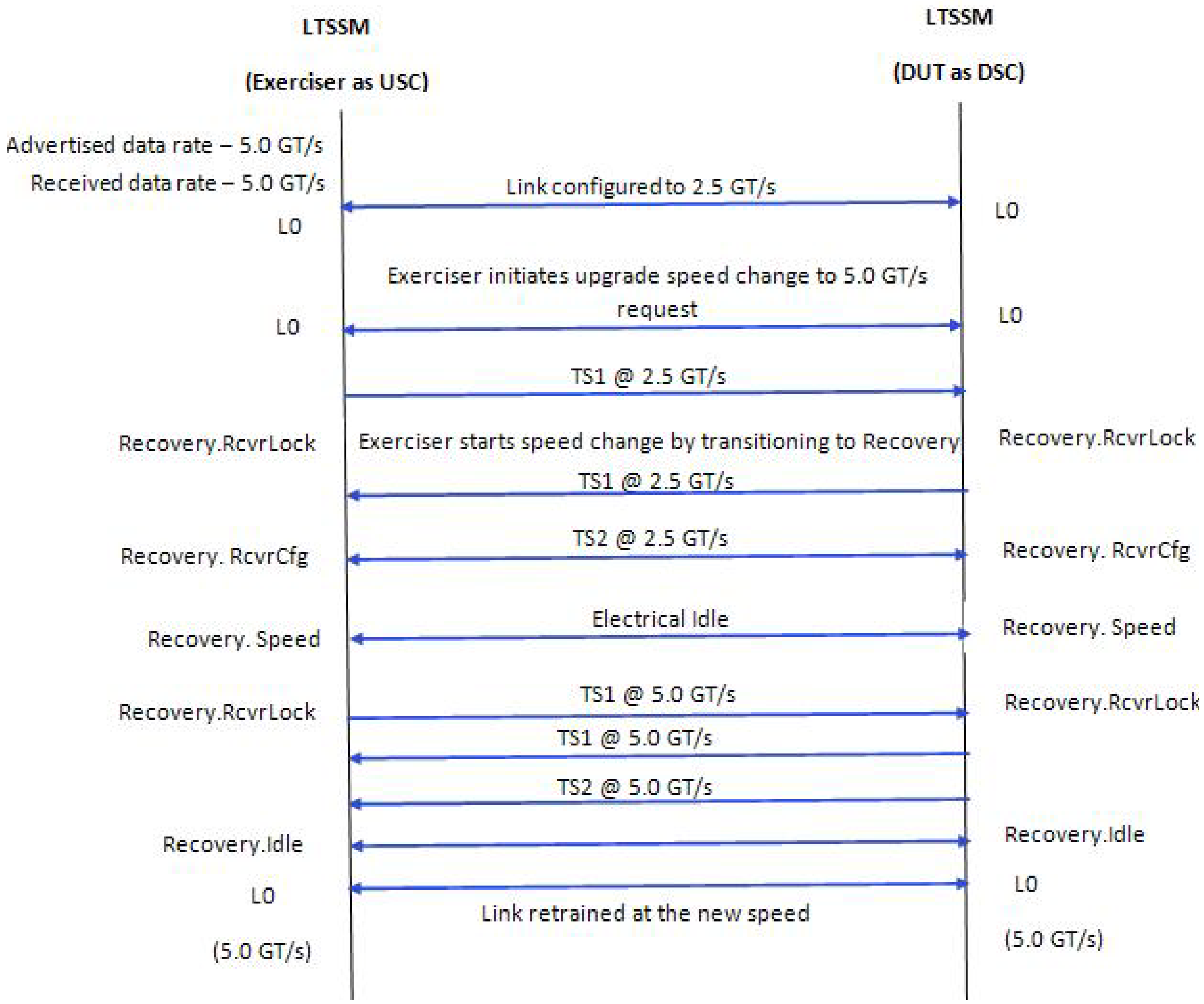
You can use the link speed change tests to assess:

- 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, x2, 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.

7 Defining Configuration Space of Exerciser

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.

8 Configuring and Mapping Decoder and Data Memory

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

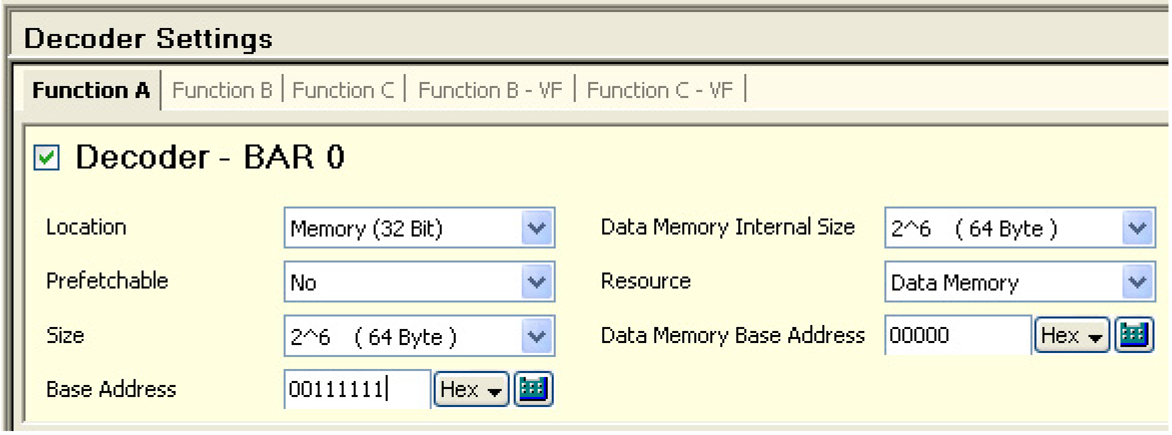
Exerciser provides six Base Address Registers (BARs) - BAR 0 to BAR 5 and an Expansion ROM for each function supported in its configuration. For each of these six BARs and an Expansion ROM of a function, an address decoder is available. You use the Decoder page to configure the address decoder settings 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. Alternatively, you can configure the decoder settings to verify the incoming TLP's payload algorithmically using the Payload Checker feature (see page 70).

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. Alternatively, you can configure the decoder settings to generate the payload of read completions algorithmically using the Payload generator feature (see page 70).

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 19 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, compare these with the contents at the specified internal memory base address, or use the payload generator feature to generate or check the payload of packets algorithmically.

VF BARs provided for an MRIOV capable Exerciser

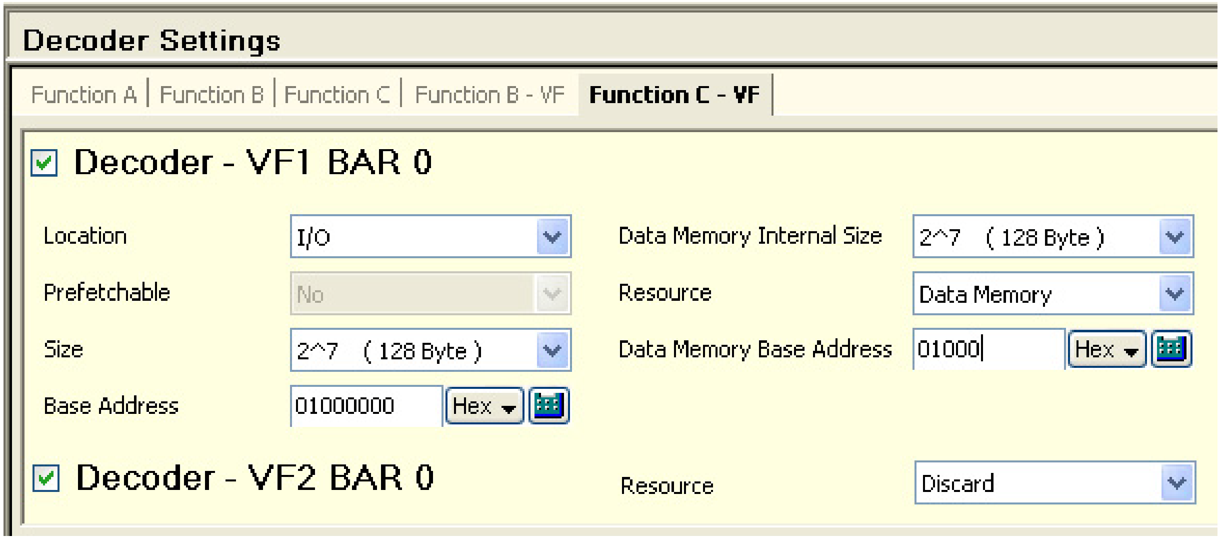
When Exerciser emulates a non IOV PCIe component, it provides six BARs for each of the three functions - Function A, B, and C.

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

Function	BARs
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

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

Resource: Data Memory

Data Memory Base Address: 01000

☑ Decoder - VF2 BAR 0

Resource: Discard

Figure 20 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.

NOTE **Alternatively, you can use the Payload generation and checker feature (see page 70) of Exerciser to generate and verify the payloads of packets algorithmically.**

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.

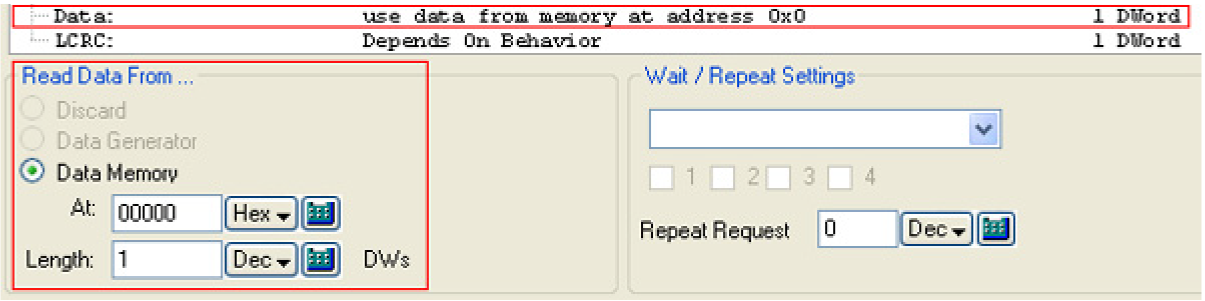
Mapping data memory addresses to BAR addresses

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 124 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 251 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

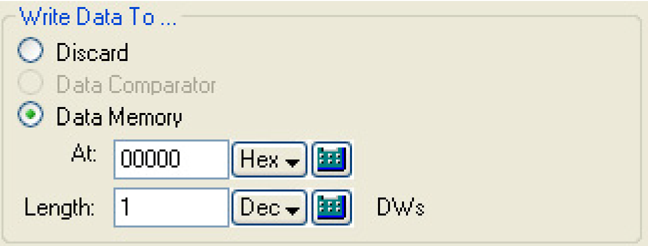
At: 00000 Hex

Length: 1 Dec DWs

Wait / Repeat Settings

□ 1 □ 2 □ 3 □ 4

Repeat Request 0 Dec



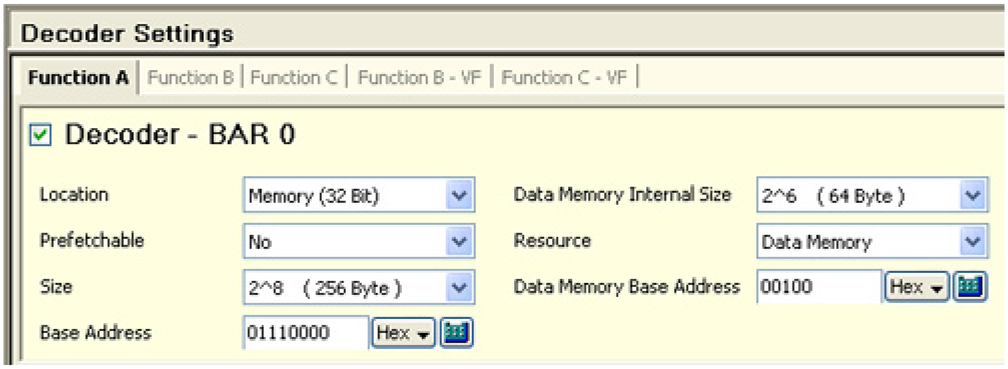
Write Data To ...

○ 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 124 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)	Data Memory Internal Size	2^6 (64 Byte)
Prefetchable	No	Resource	Data Memory
Size	2^8 (256 Byte)	Data Memory Base Address	00100 Hex
Base Address	01110000 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 124 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 I**mport Memory from File** dialog box appears.

2 Refer to the “Import Memory from File dialog box" on page 254 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 253 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 255. 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 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.

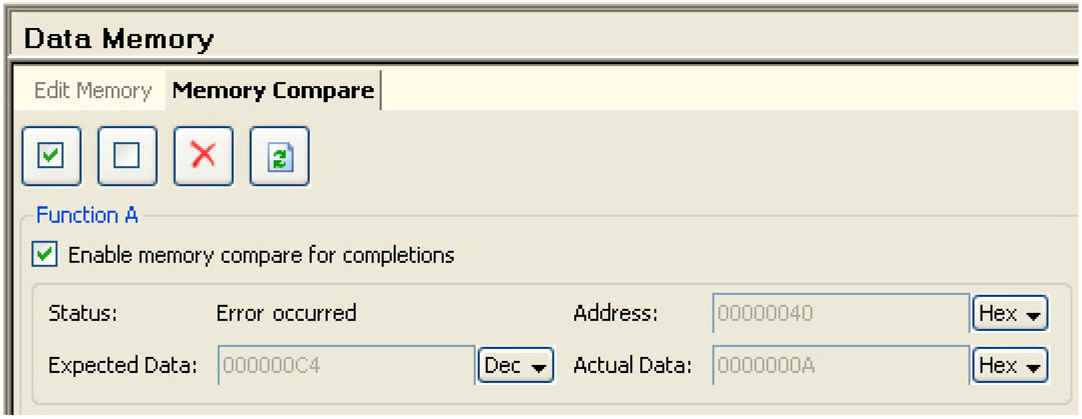
3 Write the expected contents to the target memory location that you specified in the decoder settings. You can either manually edit the memory to write the contents or import the contents from a file at the required address. The expected contents are compared with the actual data of incoming TLPs.

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.



☑ 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.

9 Providing Stimulus to DUT

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 26 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 **Hard ware 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 81 to know more about flow controls.

Defining Stimulus Traffic

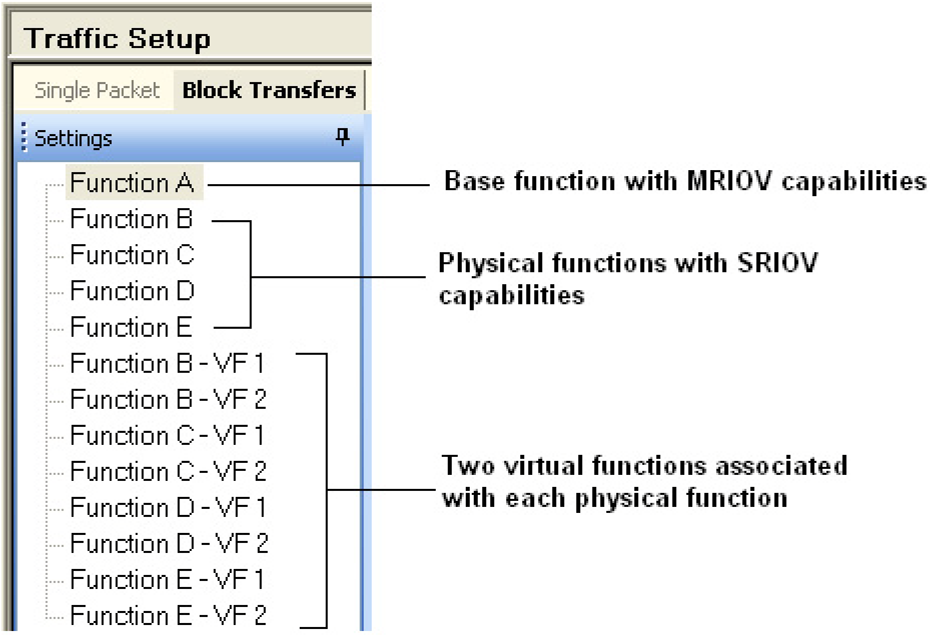
This topic describes how to define the stimulus traffic that Protocol Exerciser can send over a PCIe link to a DUT.

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 26 and "Stimulus Traffic - Overview" on page 134.

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

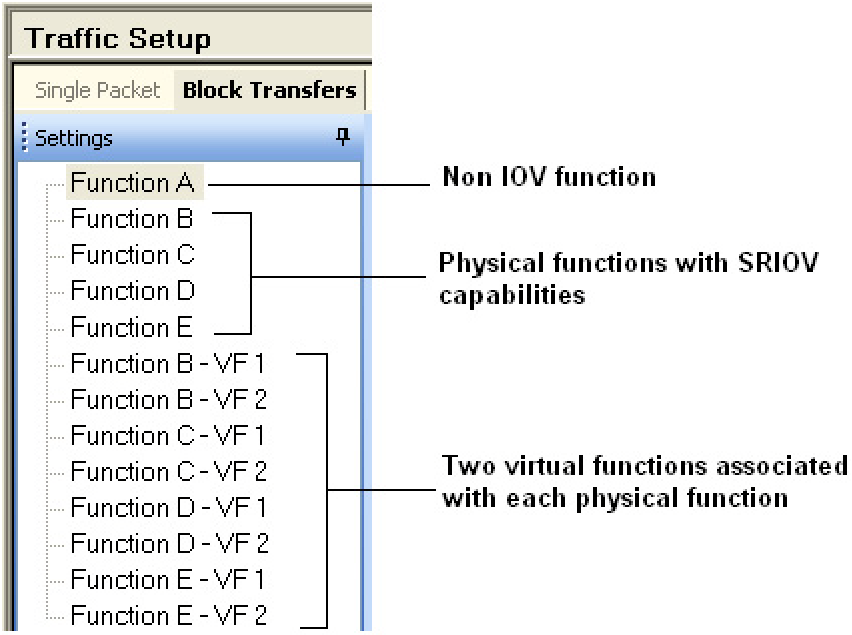


Base function with MRIOV capabilities

Physical functions with SRIOV capabilities

Two virtual functions associated with each physical function

Figure 21 Functions for an MRIOV capable Exerciser

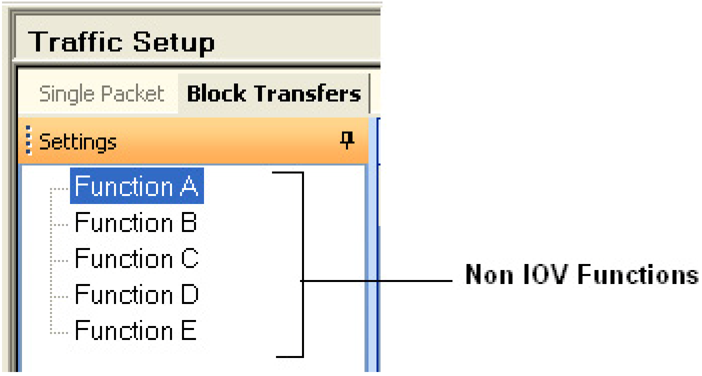


Non IOV function

Physical functions with SRIOV capabilities

Two virtual functions associated with each physical function

Figure 22 Functions for a SRIOV Exerciser



Non IOV Functions

Figure 23 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.

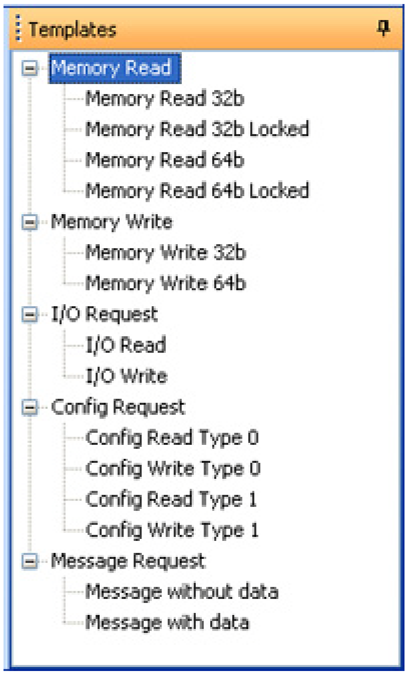


Figure 24 PCIe traffic templates

The templates are categorized as:

Templates	Description
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 81 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

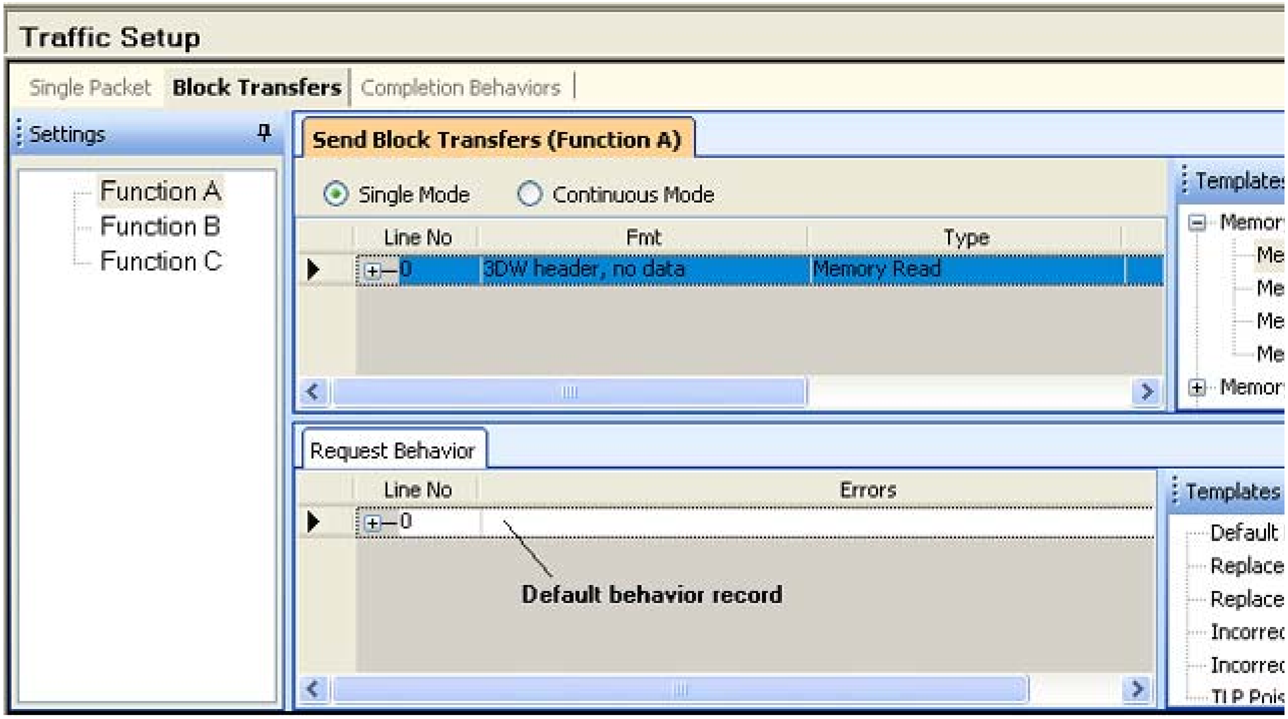
You can save the stimulus traffic setup in a Protocol Exerciser setup file. To do this, click File -> Save As and save the setup in a .exs file.

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 25 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 Packets	Request Behavior Records
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 249 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. Alternatively, you can use the Payload generator feature (see page 70) to generate the payload of such a packet algorithmically. You use the **Data Generator** option in the **Read Data From** group box to do this.

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. Alternatively, you can use the Payload checker feature (see page 70) to algorithmically verify the payload of such a completion. You use the **Data Comparator** option in the **Write Data To**... group box to do this.

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.

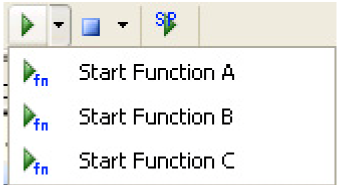


Figure 26 Start stimulus for a function

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.

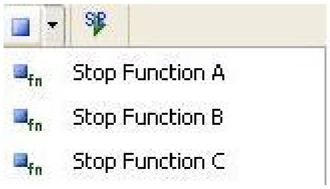
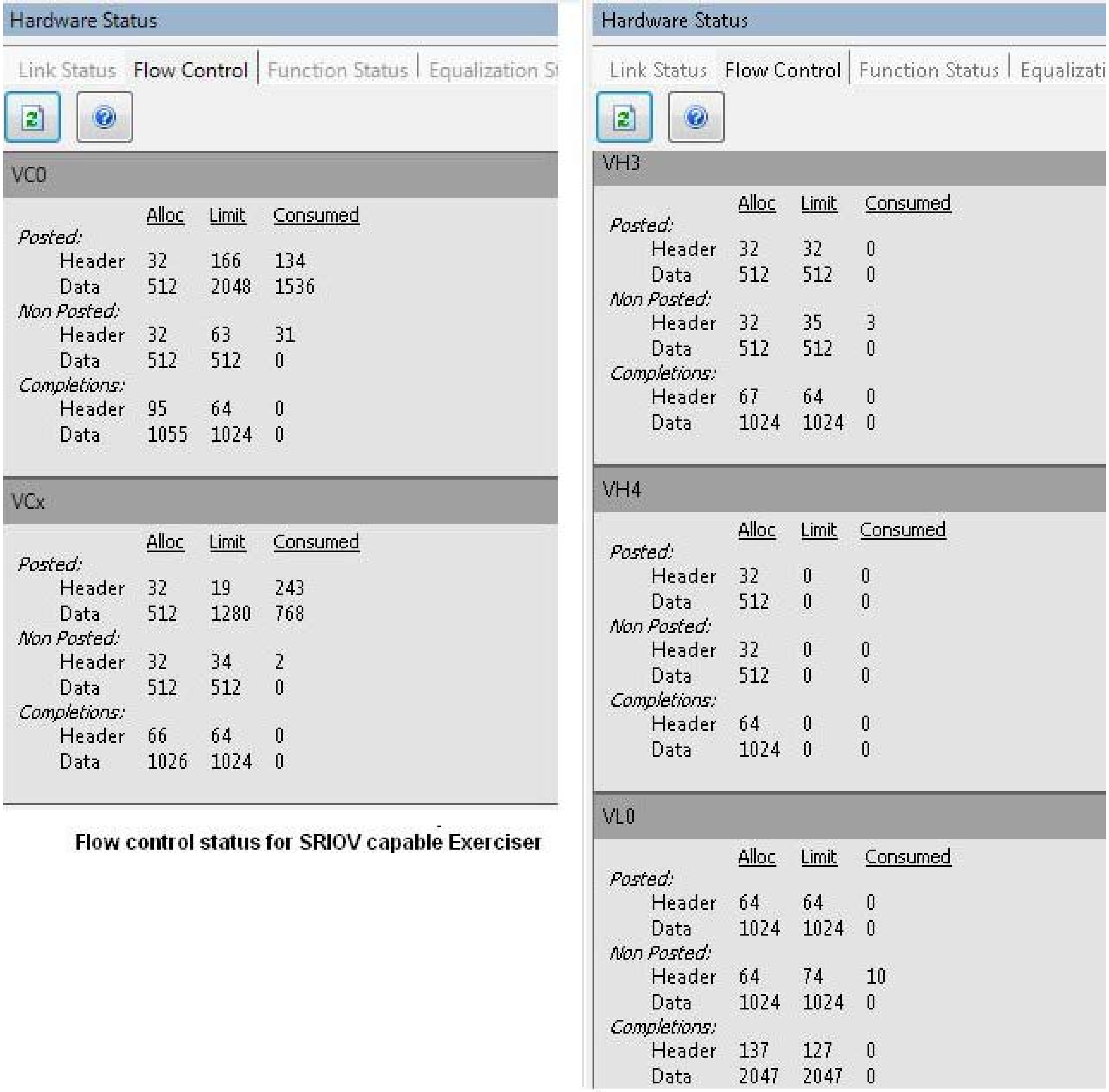


Figure 27 Stop stimulus for a function

Viewing the stimulus results

You can view the stimulus results in the **Hard ware status** pane.



Hardware Status

Link Status | Flow Control | Function Status | Equalization S

VC0

	Alloc	Limit	Consumed
Posted:			
Header	32	166	134
Data	512	2048	1536
Non Posted:			
Header	32	63	31
Data	512	512	0
Completions:			
Header	95	64	0
Data	1055	1024	0

VCx

	Alloc	Limit	Consumed
Posted:			
Header	32	19	243
Data	512	1280	768
Non Posted:			
Header	32	34	2
Data	512	512	0
Completions:			
Header	66	64	0
Data	1026	1024	0

Flow control status for SRIOV capable Exerciser

Hardware Status

Link Status | Flow Control | Function Status | Equalizati

VH3

	Alloc	Limit	Consumed
Posted:			
Header	32	32	0
Data	512	512	0
Non Posted:			
Header	32	35	3
Data	512	512	0
Completions:			
Header	67	64	0
Data	1024	1024	0

VH4

	Alloc	Limit	Consumed
Posted:			
Header	32	0	0
Data	512	0	0
Non Posted:			
Header	32	0	0
Data	512	0	0
Completions:			
Header	64	0	0
Data	1024	0	0

VL0

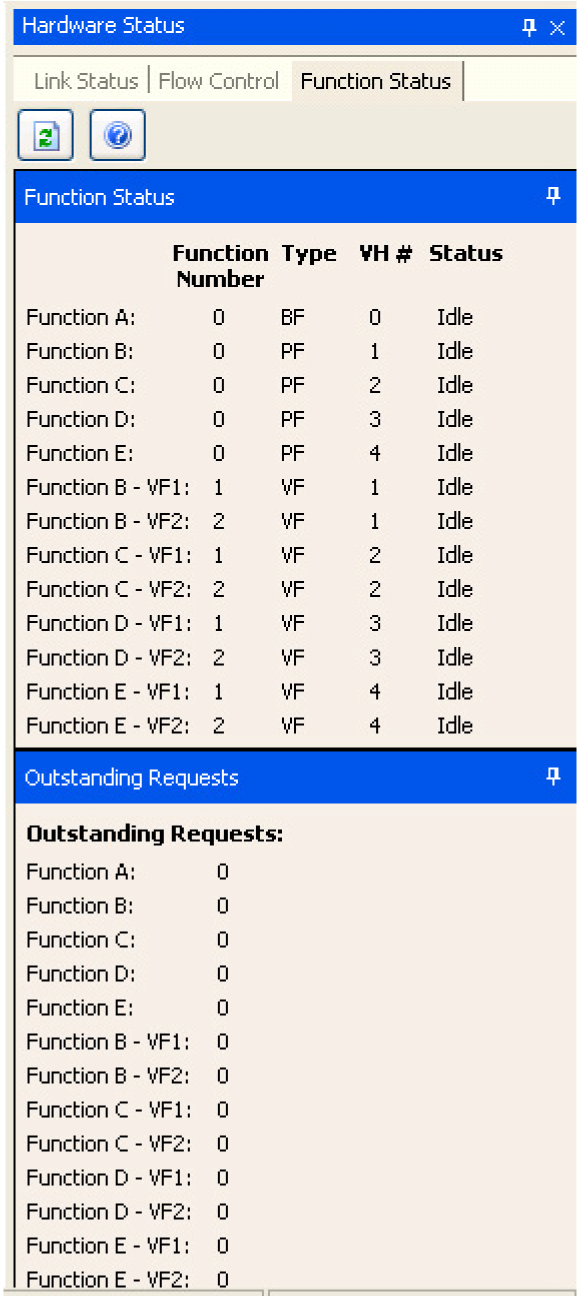
	Alloc	Limit	Consumed
Posted:			
Header	64	64	0
Data	1024	1024	0
Non Posted:			
Header	64	74	10
Data	1024	1024	0
Completions:			
Header	137	127	0
Data	2047	2047	0

Figure 28 Flow control status after stimulus for SRIOV and MRIOV capable Exerciser

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 if Exerciser emulates a non IOV/SRIOV component and for each virtual hierarchy and virtual link VL0 if Exerciser emulates an MRIOV component. These credit values 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 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 29 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 150 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

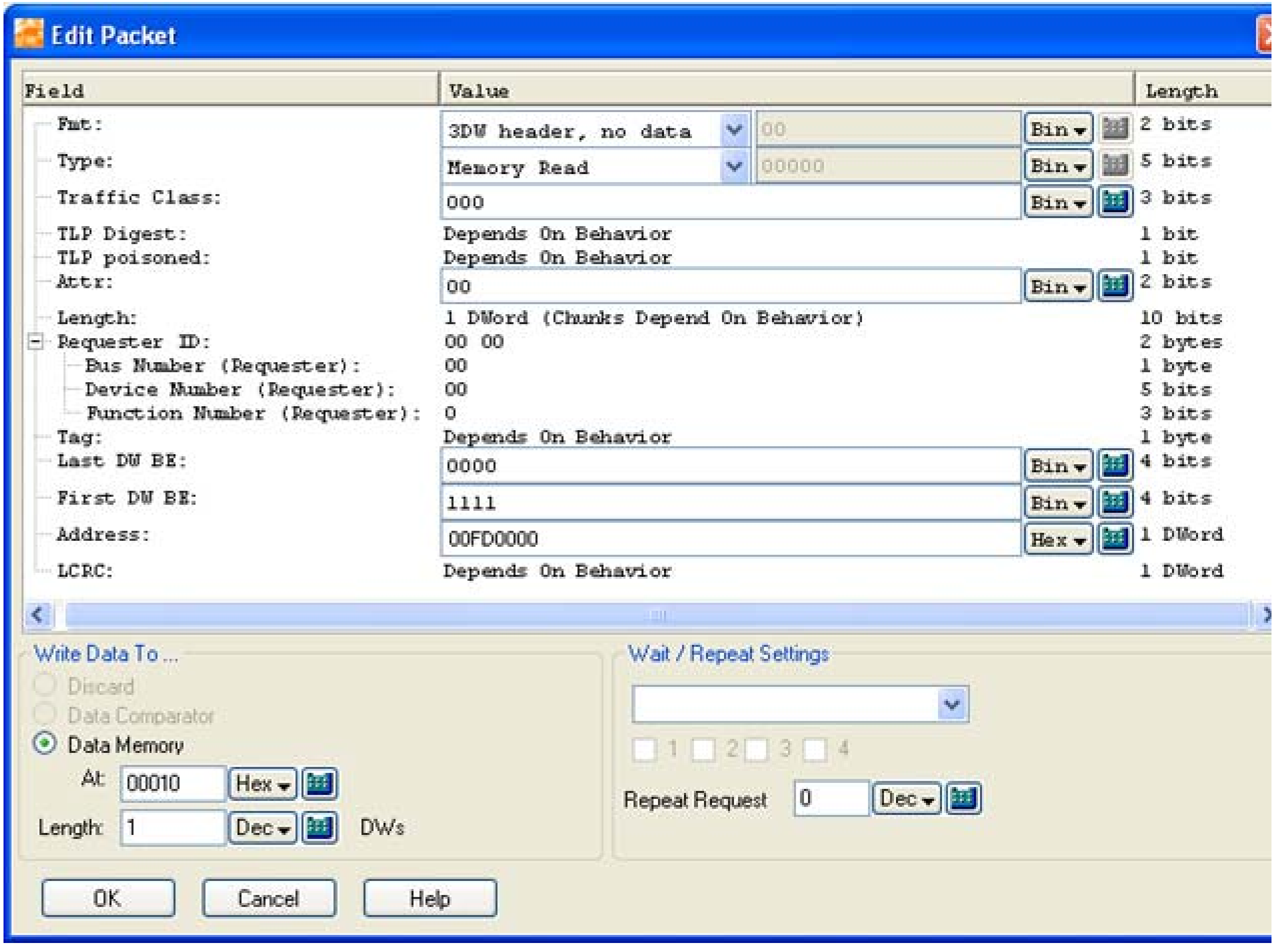
- Number and type of errors (belonging to different categories) received from DUT

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.



Edit Packet

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

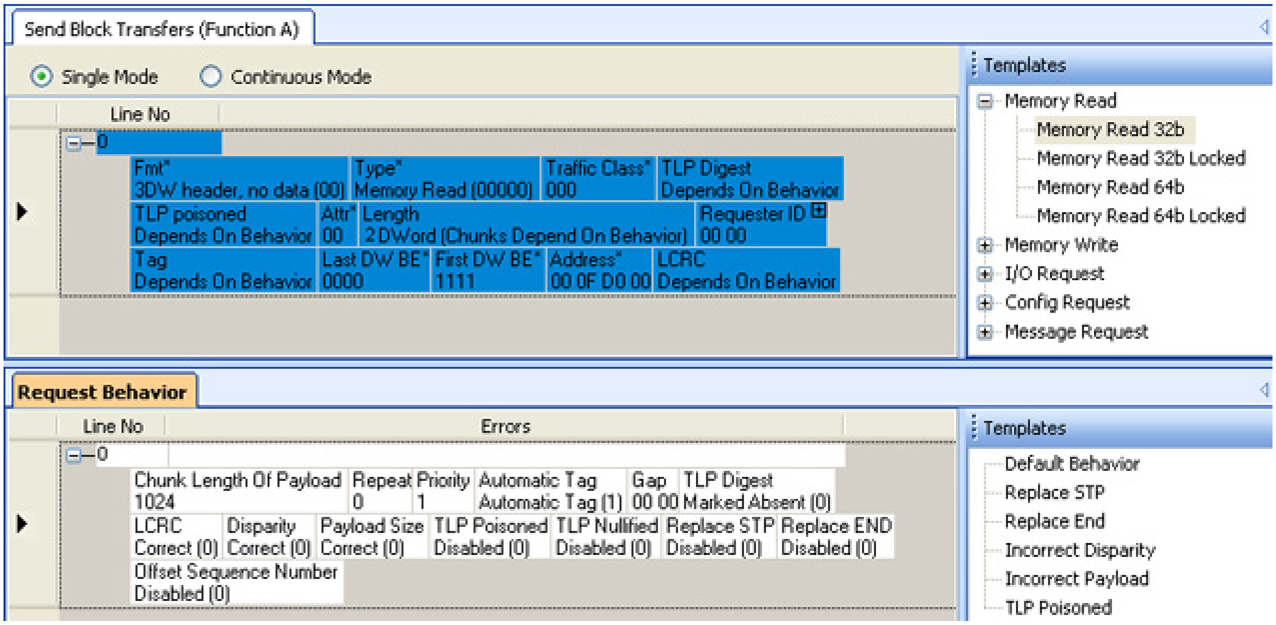
Length: 1 Dec DWs

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

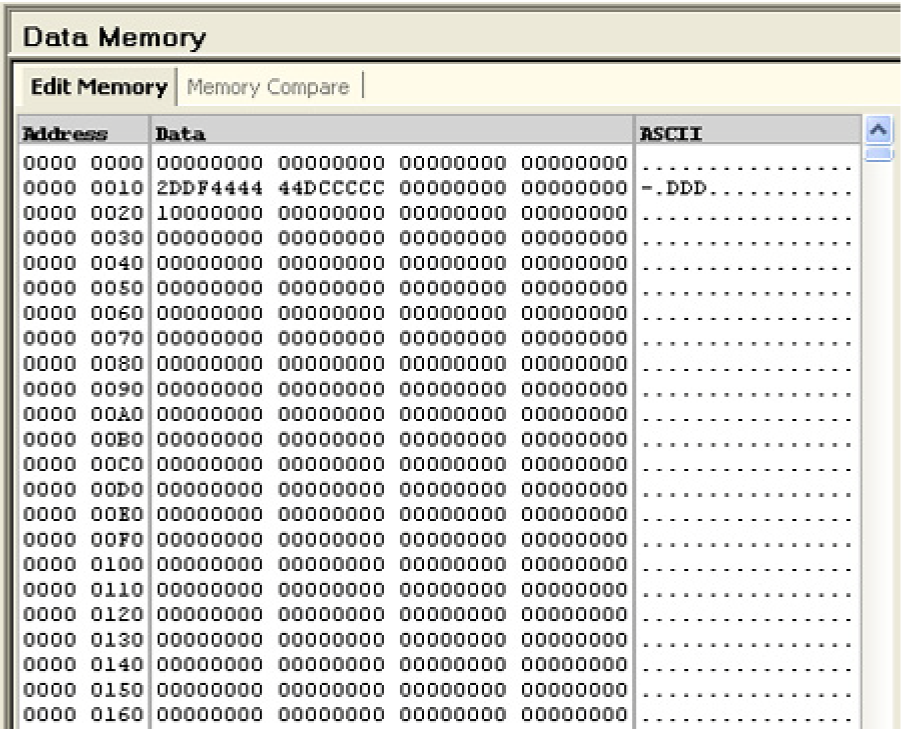
Chunk Length Of Payload	Repeat	Priority	Automatic Tag	Gap	TLP Digest
1024	0	1	Automatic Tag (1)	00 00	Marked Absent (0)

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.



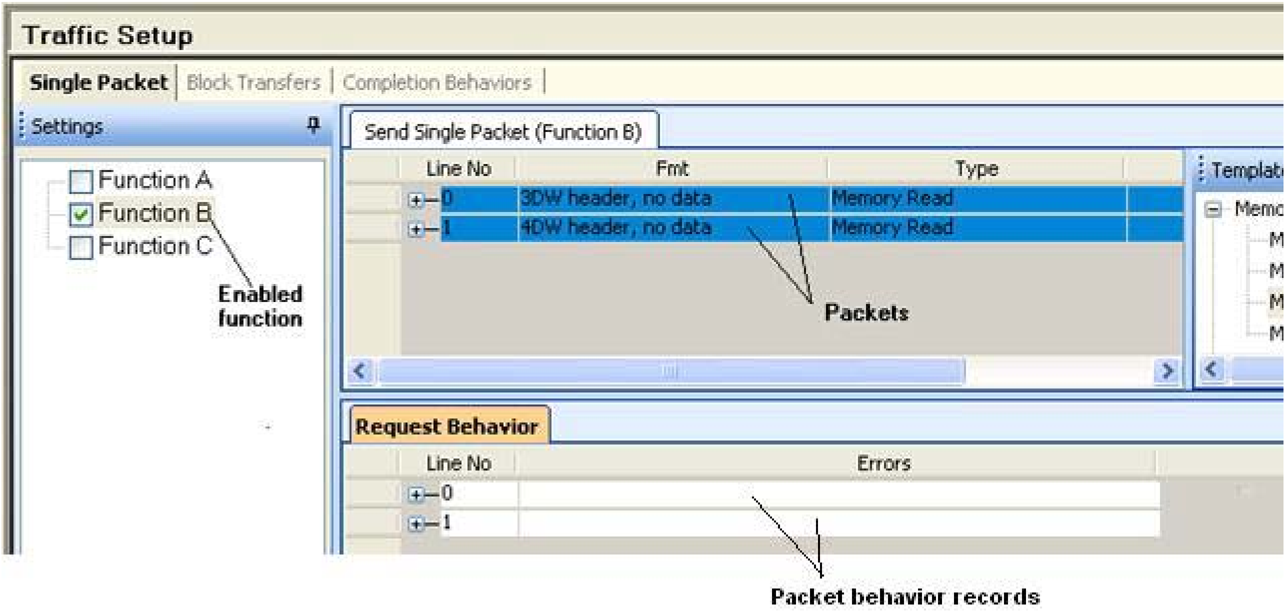
Data Memory

```
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

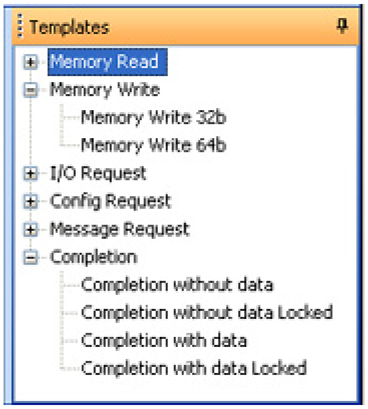
Line No	Errors
0	
1	

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 SP 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:

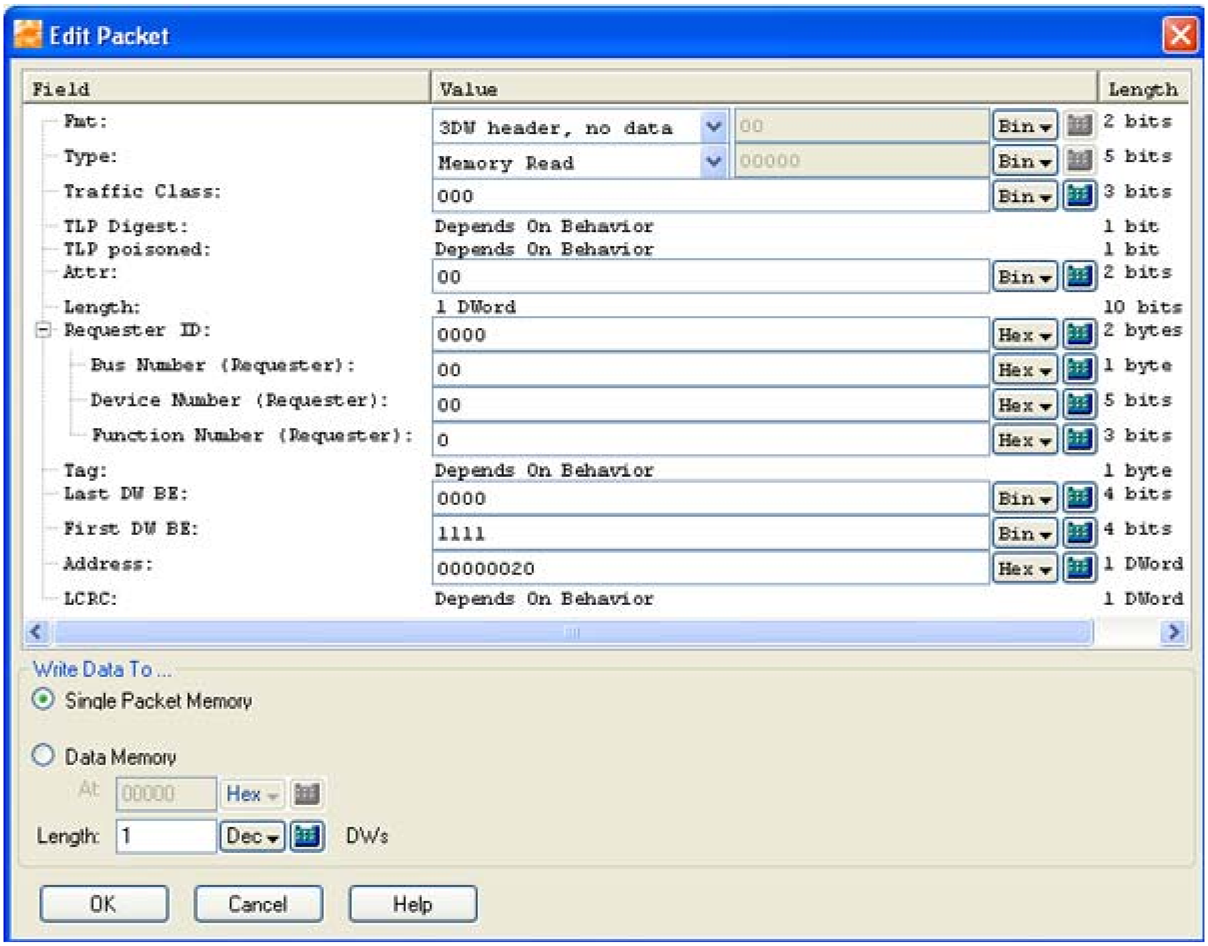
1 Click the Single Packet tab in the Traffic Setup page.

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.



Edit Packet

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 ...

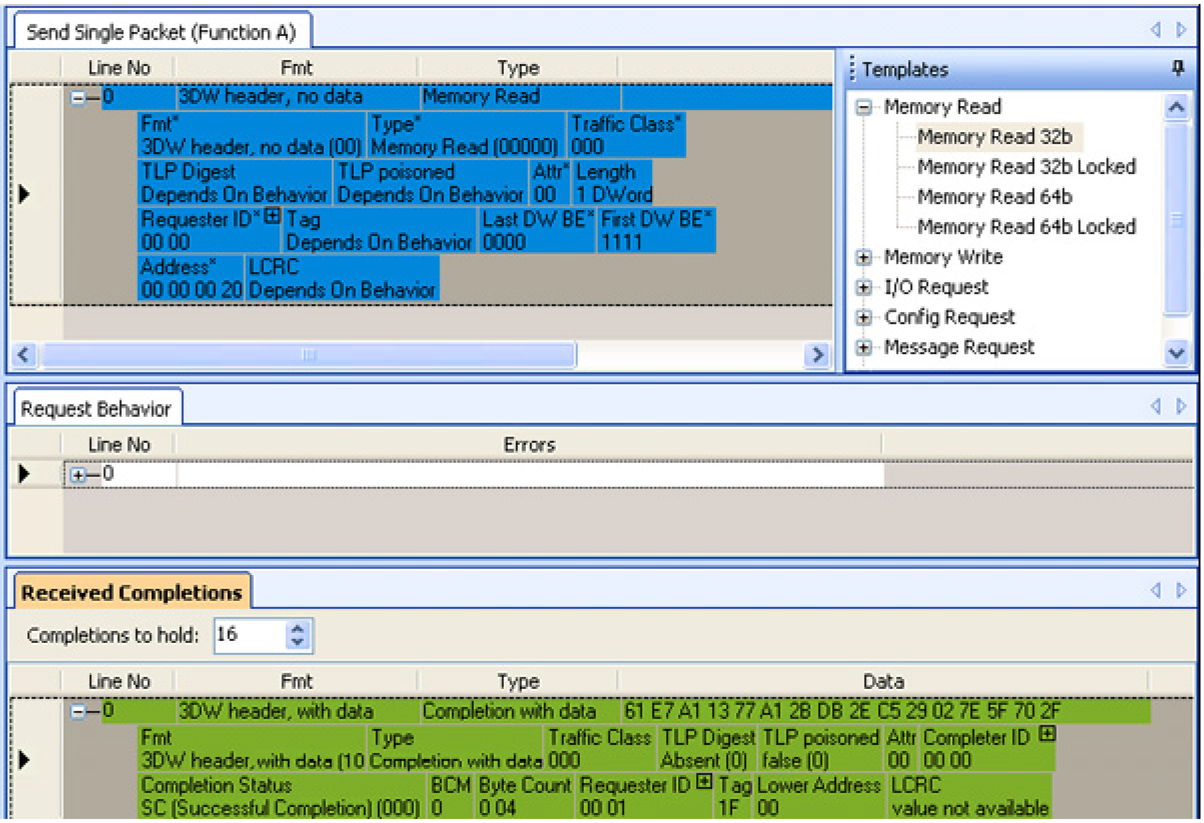
● Single Packet Memory

○ 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.



Send Single Packet (Function A)

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

Request Behavior

Line No	Errors
0	

Received Completions

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:

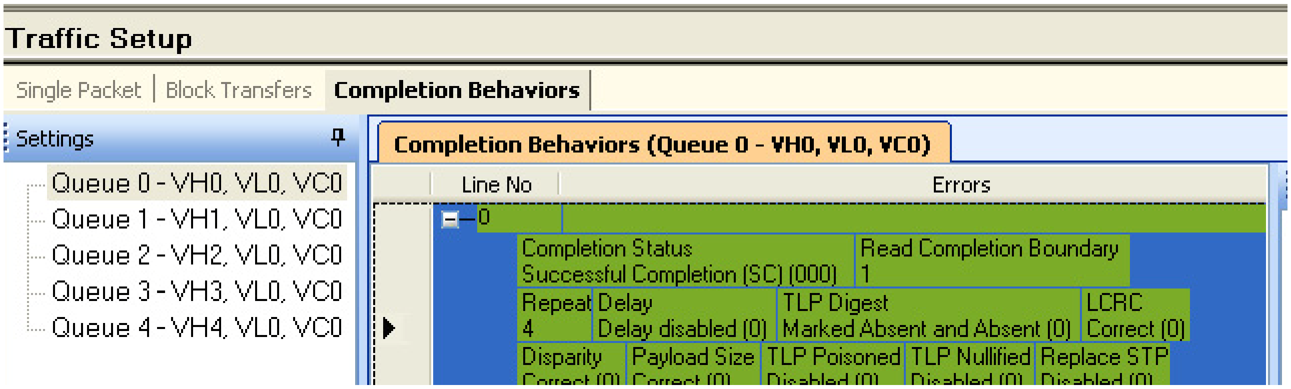
1 Click the Traffic Setup icon from the navigation bar of the Protocol Exerciser GUI.

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 Requests	Completion Behavior Records
TLP request 1	Completion packet as per completion behavior record 1
TLP request 2	Completion packet as per completion behavior record 2

TLP Requests	Completion Behavior Records
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

10 Performing Protocol Checks and Inserting Errors

This chapter provides information on the Protocol Checker function of the Protocol Exerciser application. It also describes how you can use Protocol Exerciser to send an arbitrary DLLP or insert a wrong CRC to the DLLP and 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.

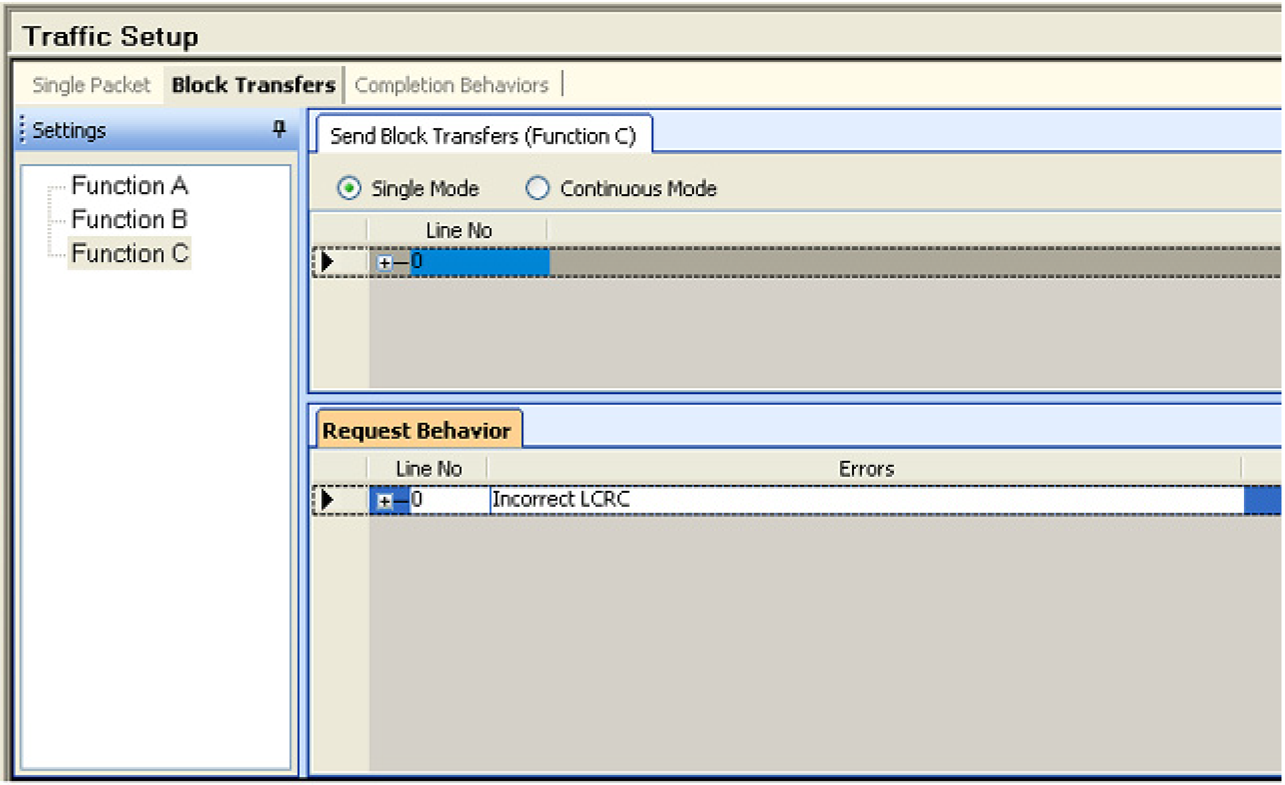
The Error Insertions page provides the error insertion settings categorized in three different tabs as follows:

- **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.

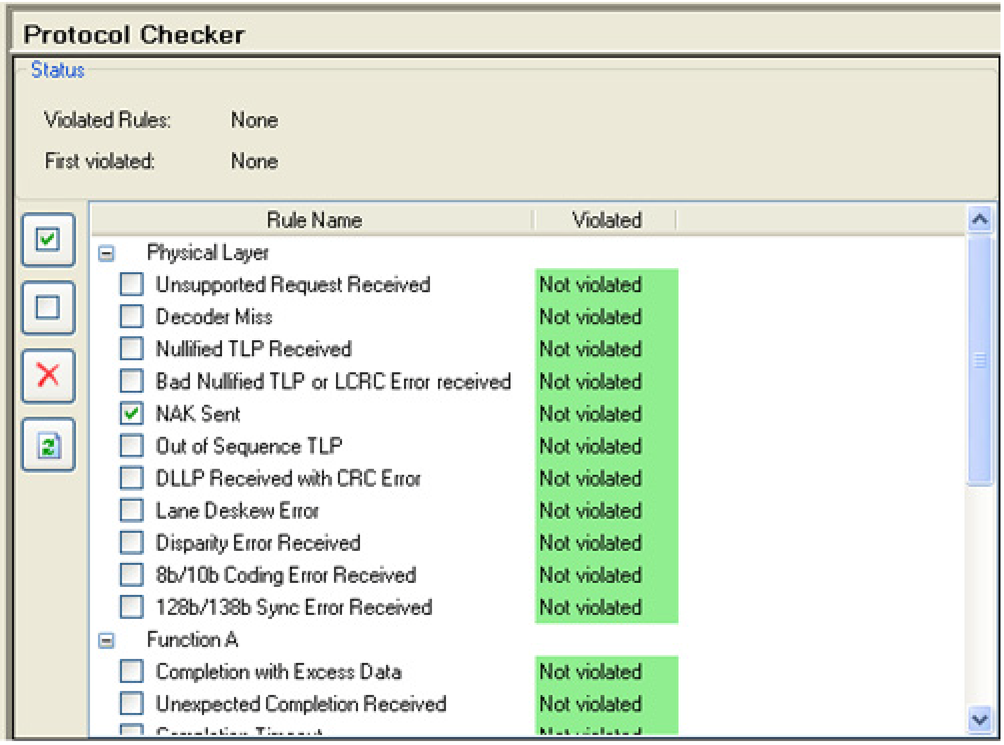


● Single Mode ○ Continuous Mode

Line No
0

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

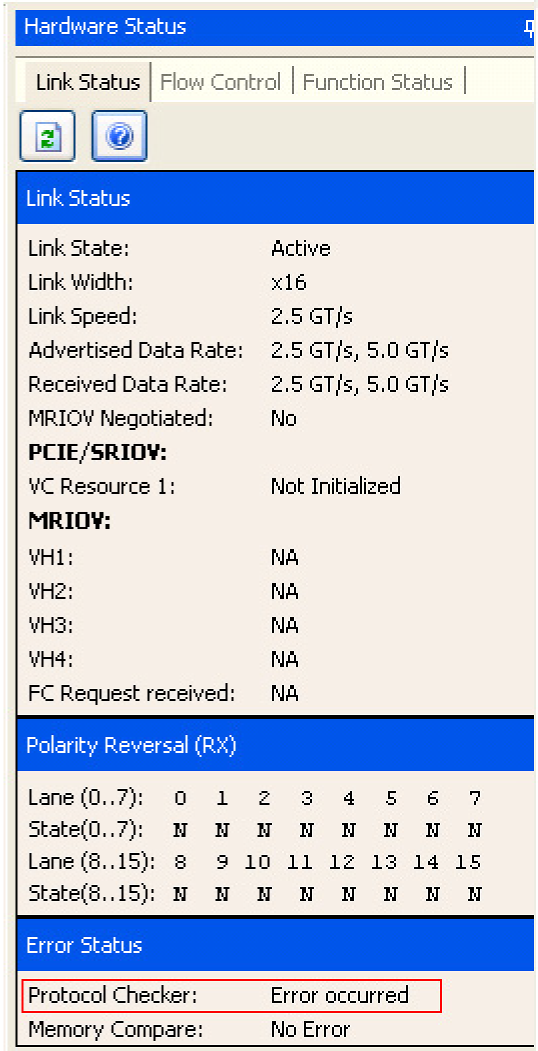
Status

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

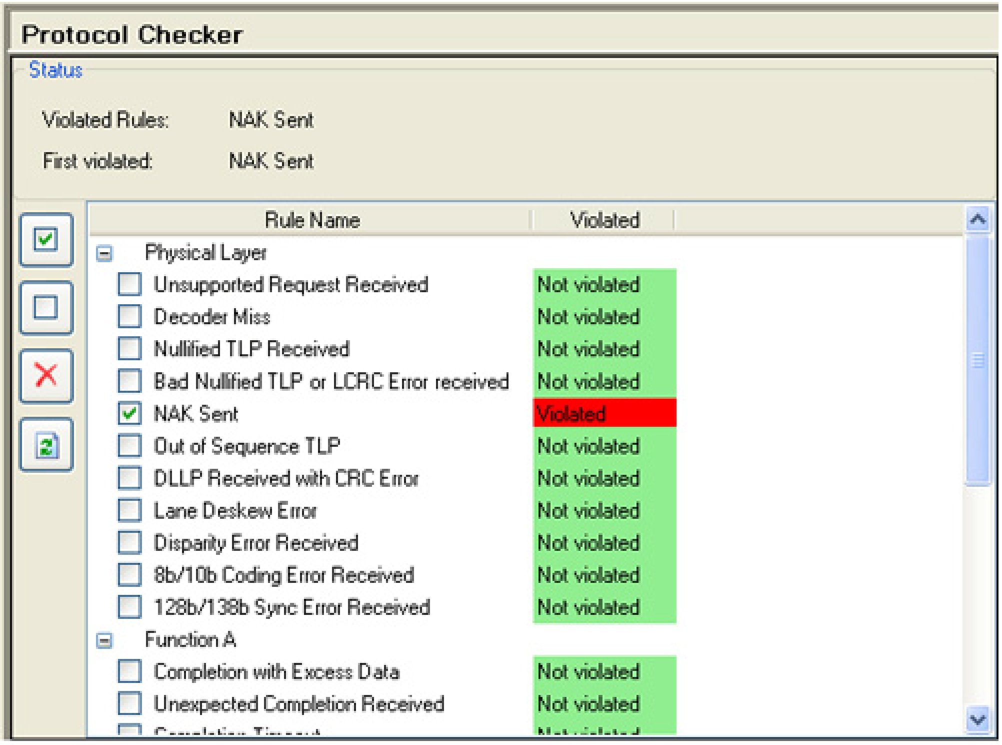
Polarity Reversal (RX)

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:

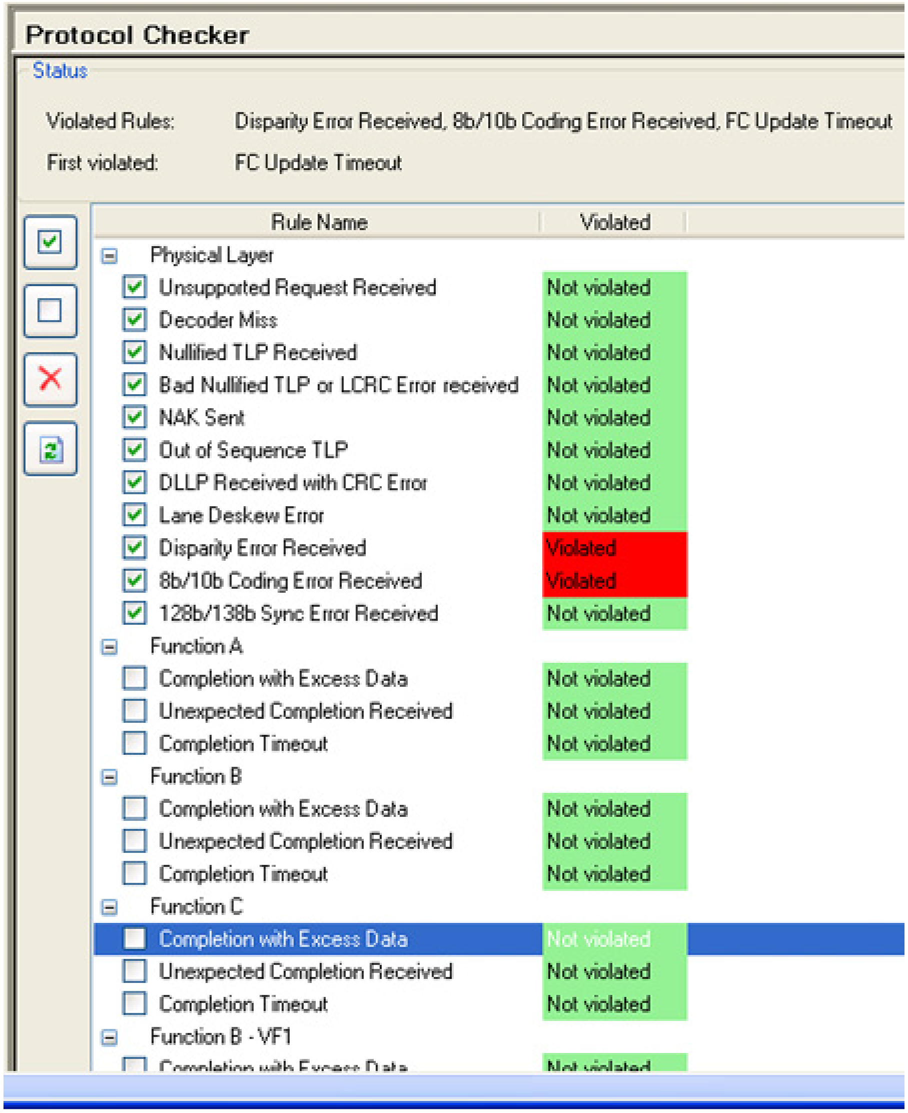
1 Click the **Protocol Checker** icon in the Navigation pane.

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 **Hard ware 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
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.
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.

Protocol Rule	Description
For each virtual hierarchy supported by Exerciser (applicable when protocol is set to MRIOV)	
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.

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.

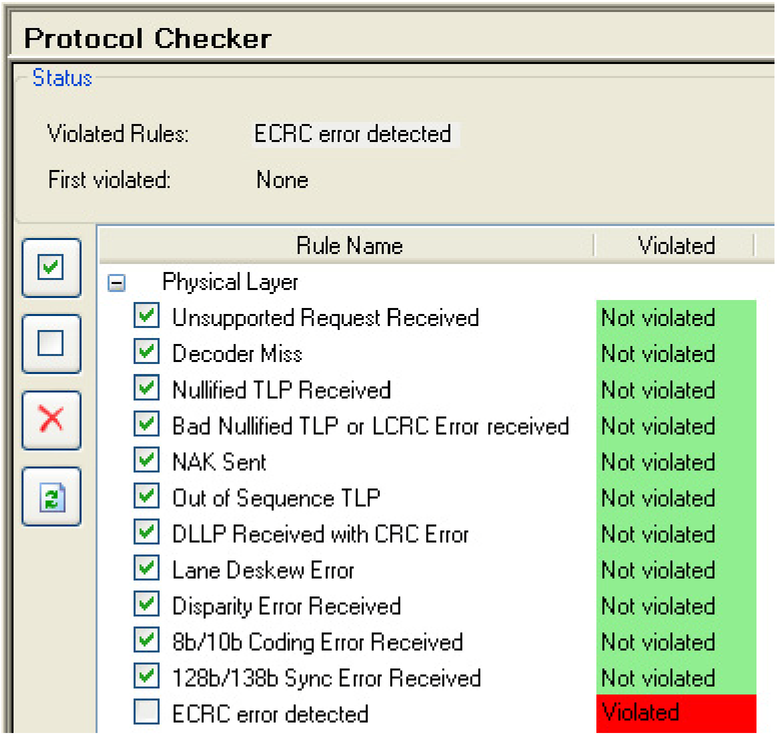
As a receiver, Exerciser has the ECRC checking supported and enabled.

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.

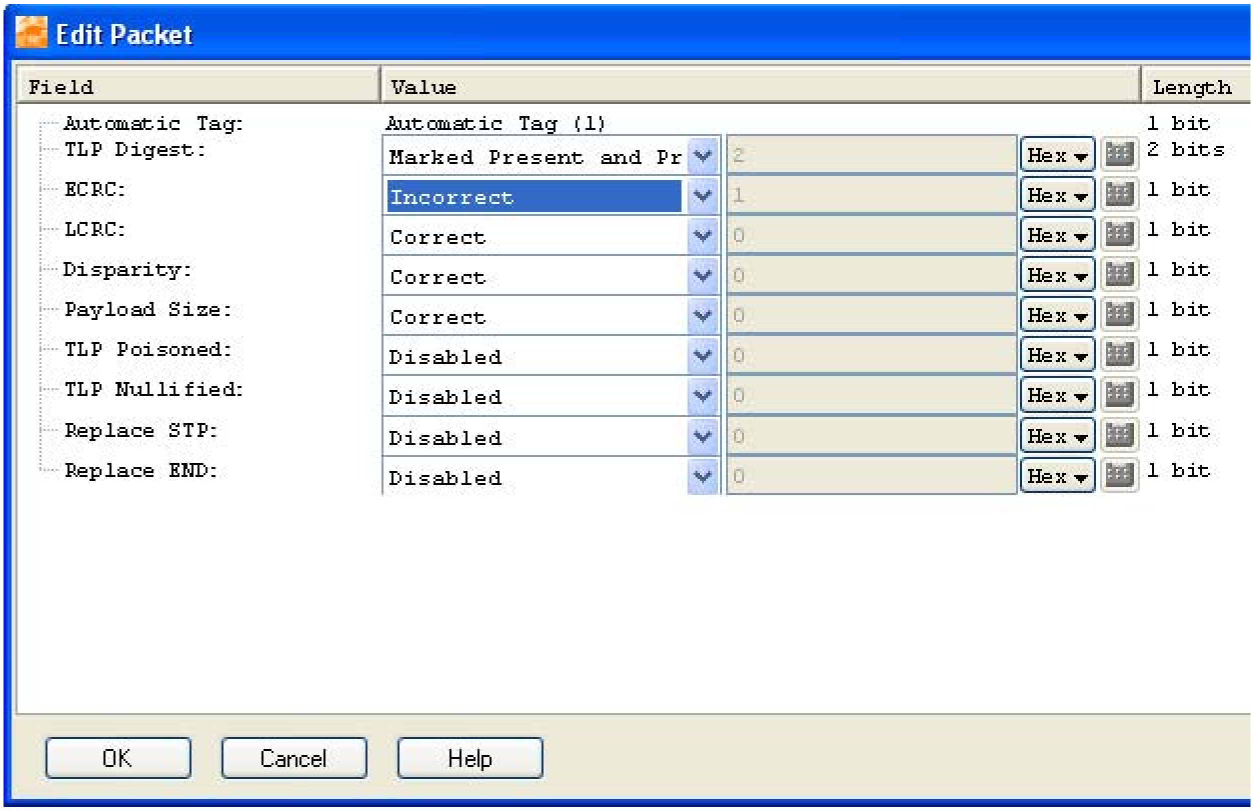
2 Double-click the TLP request behavior record to configure the behavior of the request.

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.

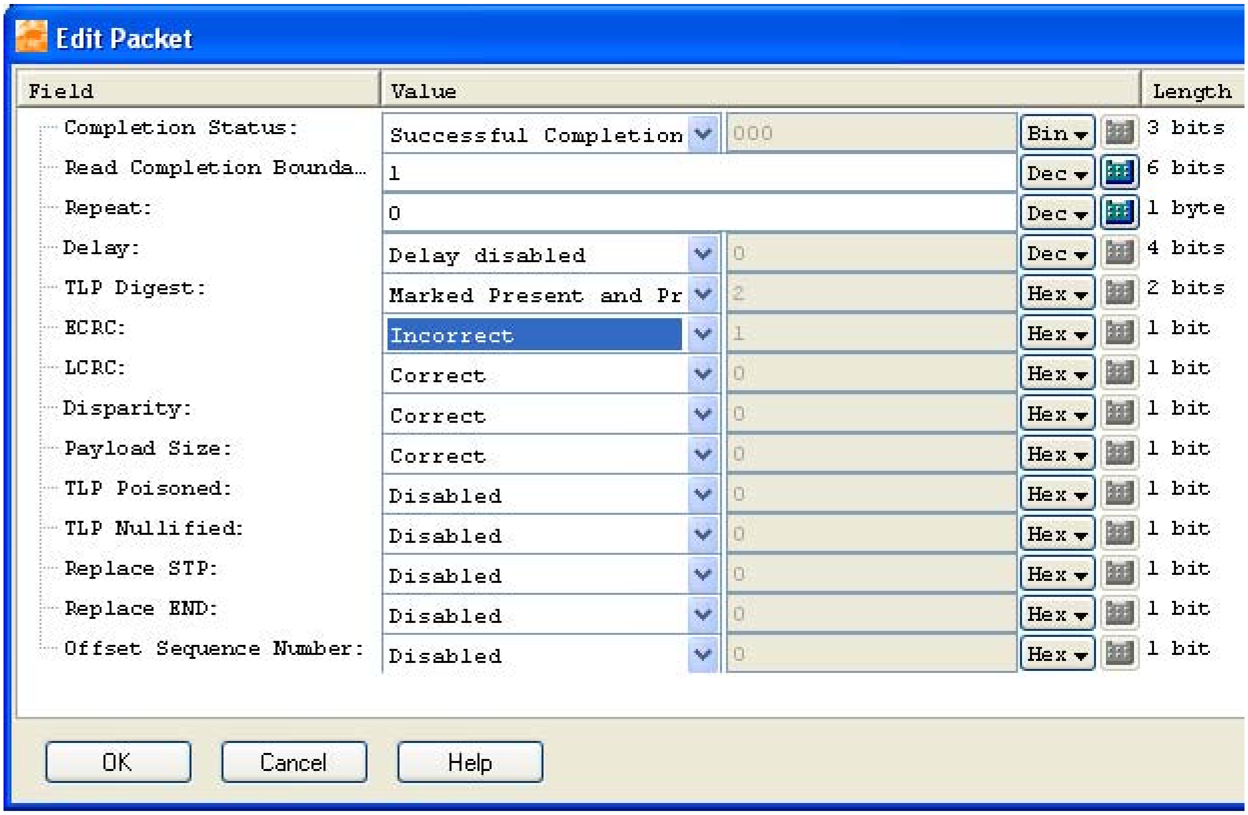
5 Click OK.



Edit Packet

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.


Edit Packet

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:

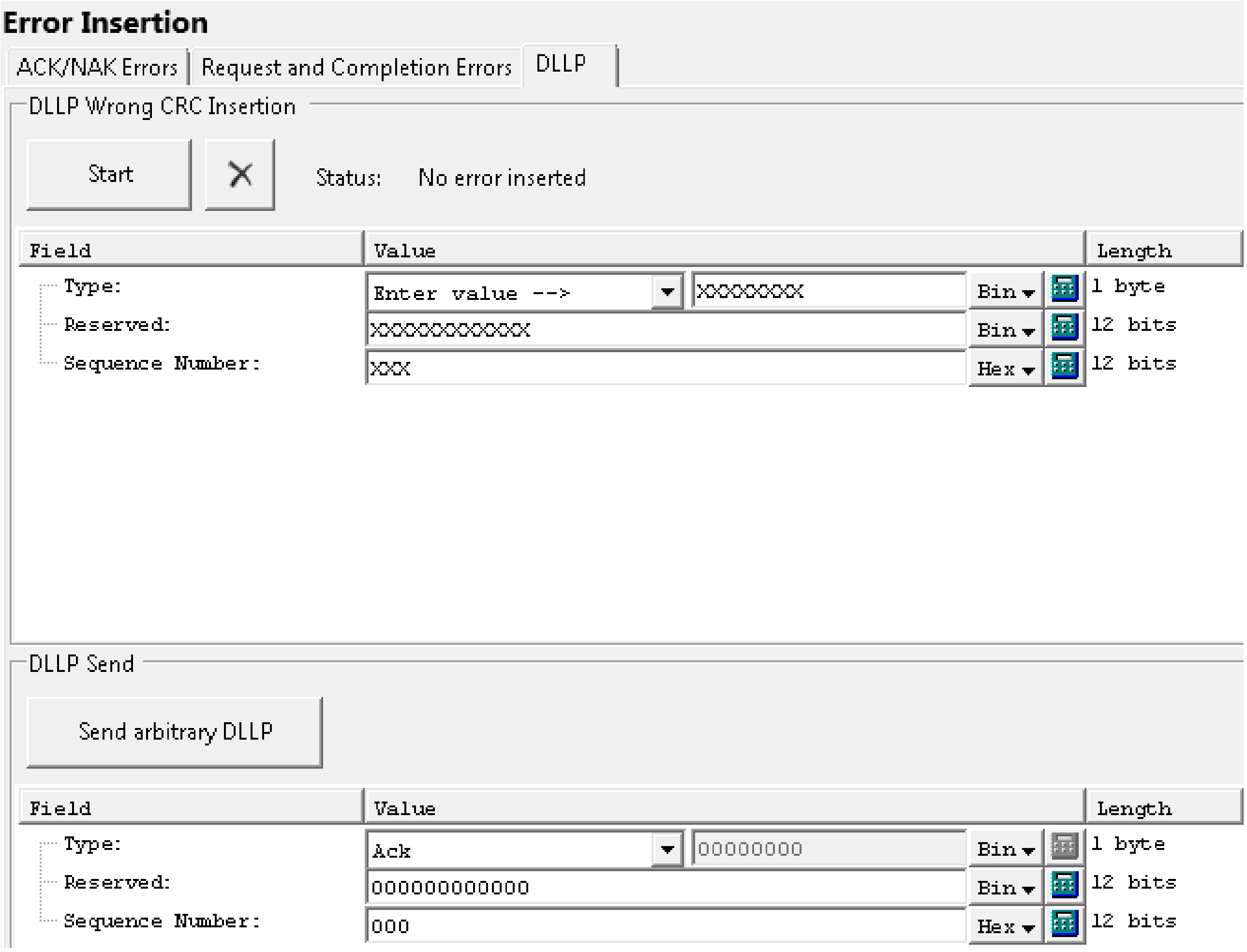
- Marked Absent and Absent
- Marked Present but Missing

Sending an Arbitrary DLLP Or Inserting a Wrong CRC in the DLLP

You can configure Exerciser to purposely send arbitrary DLLPs, including unexpected and incorrect DLLPs with an incorrect checksum. You can send an arbitrary DLLP or insert a wrong CRC in DLLP packets to test how a DUT responds to the error conditions. You can select which type of DLLPs (i.e. ACK, NAK, FC, PM, or any other) you want to get affected by enabling error insertion.

NOTE **The error insertion occurs only once. Thus, the next DLLP of that type will be transmitted with the correct checksum.**

For sending an arbitrary DLLP or inserting a wrong CRC to the DLLP, click the **Error Insertion** icon from the Navigation window of the Protocol Exerciser GUI and then click the **DLLP** tab in the Error Insertion window. Following screen appears:



Status: No error inserted

Field	Value			Length
Type:	Enter value -->	XXXXXXXX	Bin	1 byte
Reserved:	XXXXXXXXXXXX		Bin	12 bits
Sequence Number:	XXX		Hex	12 bits

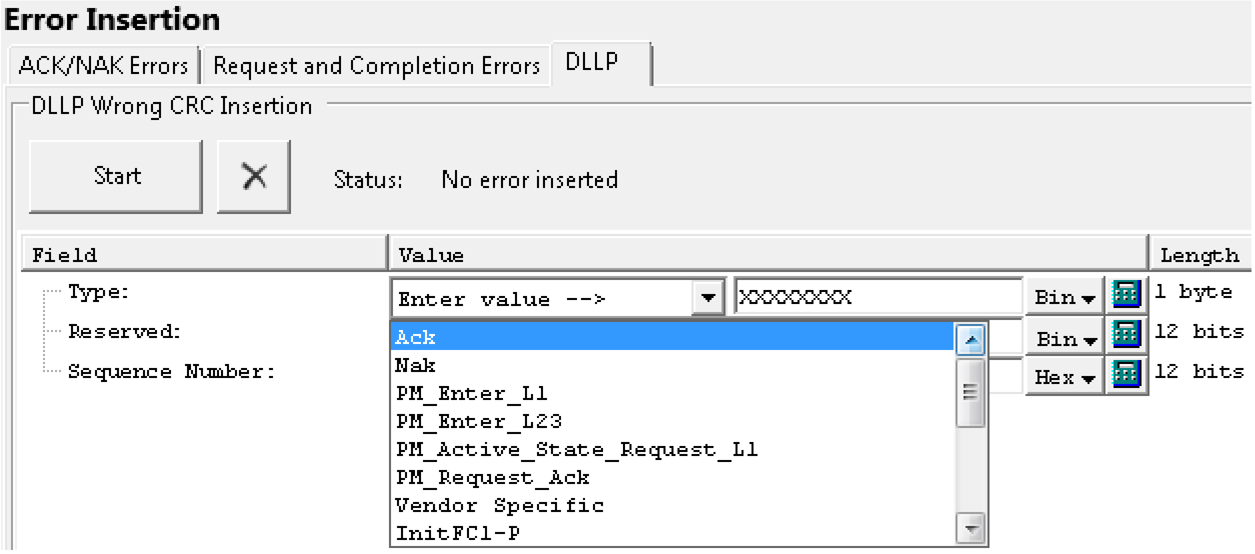
Field	Value			Length
Type:	Ack	00000000	Bin	1 byte
Reserved:	000000000000		Bin	12 bits
Sequence Number:	000		Hex	12 bits

Inserting a Wrong CRC in the DLLP

You can test how a DUT responds to a DLLP packet with a wrong CRC value received from the Exerciser.

To insert a wrong CRC

Select the DLLP type (ACK, NAK, FC, PM, or any other) for the DLLP Field type by clicking drop-down in the Value column as displayed in the following screen.



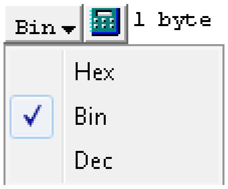
Error Insertion

Status: No error inserted

Field	Value			Length
Type:	Enter value -->	XXXXXXXX	Bin	1 byte
Reserved:			Bin	12 bits
Sequence Number:			Hex	12 bits

6 Specify the appropriate value for the other associated fields (such as Reserved, Sequence Number etc.) displayed on the basis of input component selected in the above step.

7 You may specify your inputs in hexadecimal, binary, or decimal format by clicking the drop-down button of calculator provided for input assistance.

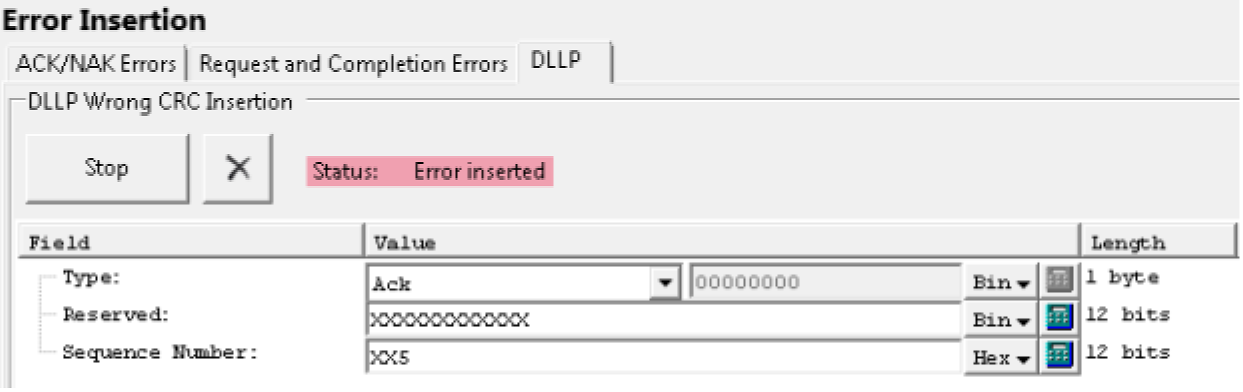


8 Click the Apply button to save the changes made.

9 Click the Start button to schedule the wrong CRC insertion into a DLLP that matches the pattern specified in the DLLP Wrong CRC insertion section. Clicking the Start button replaces it with the Stop button, until the scheduled wrong CRC insertion is done.

10 When a wrong CRC is inserted in a DLLP, status is displayed as 'Error Inserted' which is otherwise displayed as 'No error inserted' message.

11 Click the ✕ button to clear the status and insert the error again



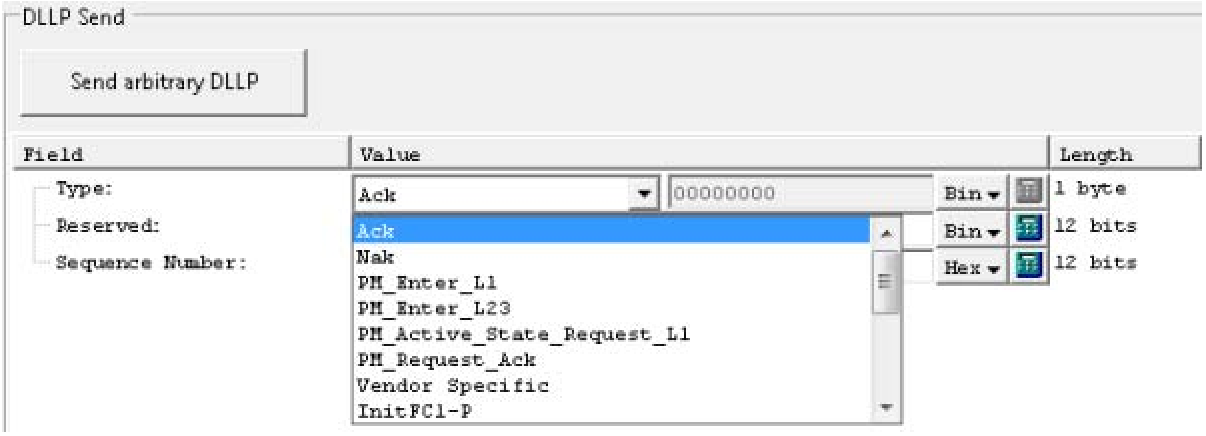
Field	Value			Length
Type:	Ack	00000000	Bin	1 byte
Reserved:	XXXXXXXXXXXX		Bin	12 bits
Sequence Number:	XX5		Hex	12 bits

Sending an Arbitrary DLLP to the DUT

You can test how a DUT responds to arbitrary DLLP packets sent by exerciser. You can check whether the DUT takes those packets as normal packets or notifies the exerciser in a different manner after receiving those packets.

To send an Arbitrary DLLP to the DUT

1 Select the **DLLP type** (ACK, NAK, FC, PM, or any other) for the **DLLP** Field type by clicking drop-down in the **Value** column as displayed in the following screen.

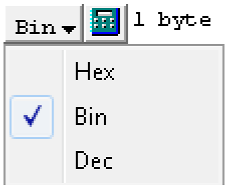


Field	Value			Length
Type:	Ack	00000000	Bin	1 byte
Reserved:			Bin	12 bits
Sequence Number:			Hex	12 bits

NOTE Number of fields available depends on the input value you selected in the Value column.

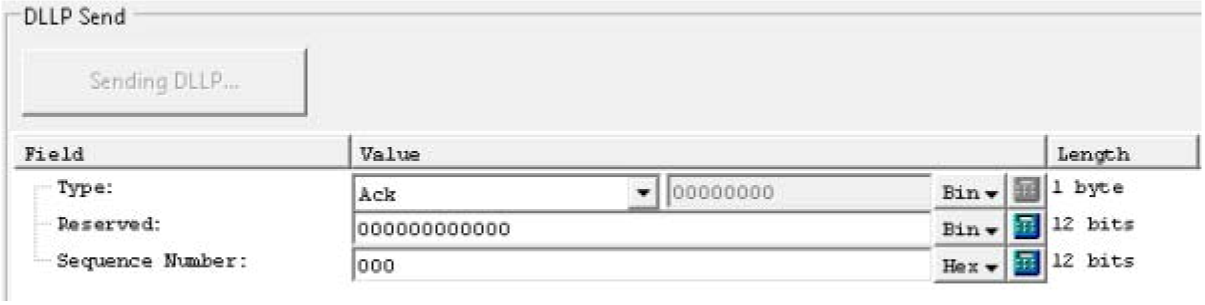
2 Specify the appropriate value for the other associated fields (such as Reserved, Sequence Number etc.) displayed on the basis of input component selected in the above step.

3 You may specify your inputs in hexadecimal, binary, or decimal format by clicking the drop-down button of calculator provided for input assistance.



4 Click the **Apply** button to save the changes made.

5 Click the **Send Arbitrary DLLP** button to send the DLLP packets to the DUT that matches the pattern specified in the DLLP Send section. Following screen appears:



Field	Value			Length
Type:	Ack	00000000	Bin	1 byte
Reserved:	000000000000		Bin	12 bits
Sequence Number:	000		Hex	12 bits

11 Emulating an NVMe Root Complex

This chapter provides information on how you can configure and use U4305A Protocol Exerciser as an NVMe root complex to provide stimulus to an NVMe DUT.

Exerciser’s Capabilities and Role as an NVMe Root Complex

This topic briefly describes the capabilities of Exerciser when emulating the role of an NVMe root complex.

In NVMe root complex emulation, you can use Exerciser to:

- view and configure NVMe controller registers of DUT.
- configure the Admin submission and completion queue attributes.
- initialize and configure the interrupt mechanism.
- initialize, view, and edit the MSI-X table of the DUT.
- create multiple I/O submission and completion queues.
- add NVMe commands to submission queues and increase the doorbell accordingly. The commands are available as predefined templates.
- view the commands and their subsequent completions in the completion queues.
- create PRP lists and PRP entries that can be used in the submitted NVMe commands for data transfer.
- create SGL lists and use these lists for data transfers in read/write commands that Exerciser submits to an NVMe DUT. You can also define the number of SGL segment entries for a SGL list.

As an NVMe root complex, Exerciser submits various requests (NVMe commands) to an NVMe DUT for completion. These commands include Admin commands submitted to the Admin queue as well as the I/O commands submitted to the I/O submission queue(s). By sending NVMe command requests to DUT, you can check how the NVMe controller responds to and completes these requests. You can also verify how the NVMe controller handles admin requests such as queue management requests or controller initialization requests.

Preparing Exerciser and DUT for the NVMe Mode

List of Steps Involved in Preparation

Prerequisites

The following are the prerequisites for Exerciser to emulate an NVMe root complex.

- You must have purchased and installed the NVMe license to get the NVMe functionality.
- Exerciser must be connected to an NVMe DUT.
- The PCIe link between the NVMe DUT and Exerciser should be active.

Steps Involved in NVMe Emulation

You must perform these steps in the specified sequence to set up Exerciser and DUT to function in the NVMe mode.

All these steps are performed using the Protocol Exerciser GUI or APIs.

- "Step 1 - Configure the PCIe Link for the NVMe Mode
- "Step 2 - Scan and Edit NVMe DUT's Configuration Space
- "Step 3 - Enable and configure the Interrupt (MSI and MSI-X) Capabilities of DUT
- "Step 4 - Configure the Required Decoders of Exerciser
- "Step 5 - Initialize the Admin Submission and Completion Queues

Click the above-mentioned steps to get detailed information on how to perform each step.

Once you complete these steps, you can start creating I/O submission and completion queues and adding NVMe commands to these queues for execution.

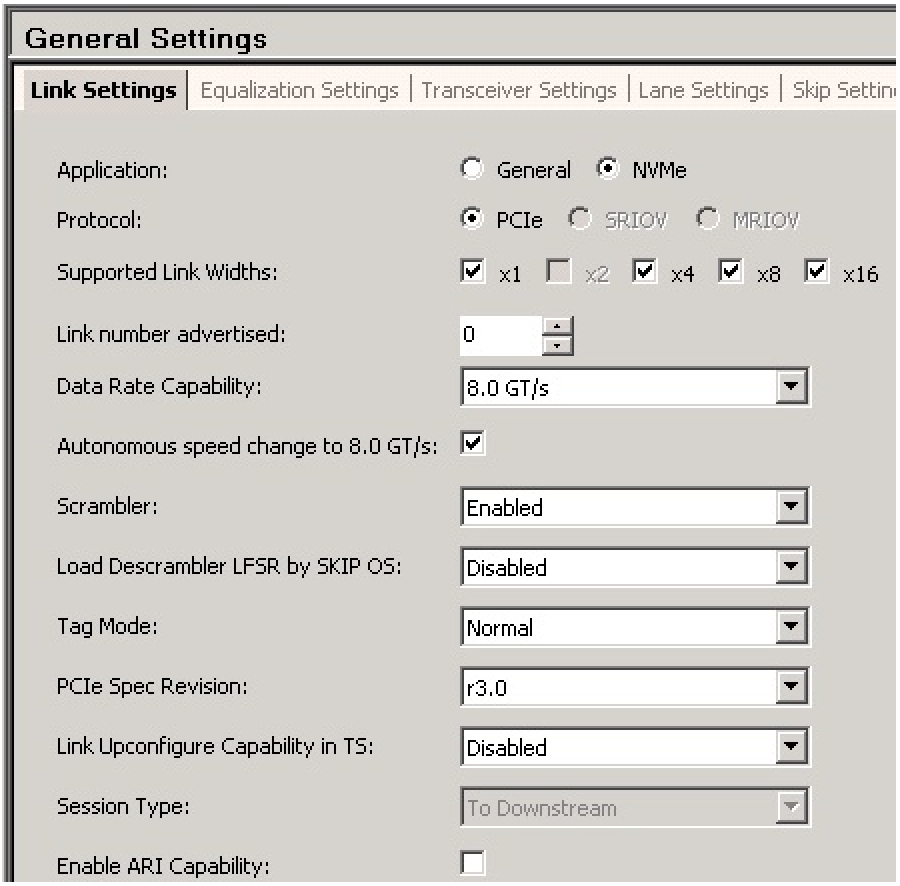
Step 1 - Configure the PCIe Link for the NVMe Mode

By default, Exerciser is not set to work in the NVMe mode. If you want Exerciser to emulate an NVMe root complex, then the first step is to set the link settings of Exerciser to the NVMe mode.

To configure the PCIe link to work in the NVMe mode

1. Click the **General Settings** icon from the left navigation pane in the GUI.
2. Click the **Link Settings** tab.
3. Click **NVMe** in the **Application** field.

On selecting NVMe, the **MRIOV** and **SRIOV** protocol support is disabled. Only **PCIe** is supported and displayed enabled and selected in the **Protocol** field. Also, the **Session Type** is automatically set to **"To Downstream"** as Exerciser can only act as a root complex in the NVMe mode.



General Settings

Link Settings | Equalization Settings | Transceiver Settings | Lane Settings | Skip Settin

Application: ○ General ● NVMe

Protocol: ● PCIe ○ SRIOV ○ MRIOV

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

Link number advertised: 0

Data Rate Capability: 8.0 GT/s

Autonomous speed change to 8.0 GT/s: ☑

Scrambler: Enabled

Load Descrambler LFSR by SKIP OS: Disabled

Tag Mode: Normal

PCIe Spec Revision: r3.0

Link Upconfigure Capability in TS: Disabled

Session Type: To Downstream

Enable ARI Capability: □

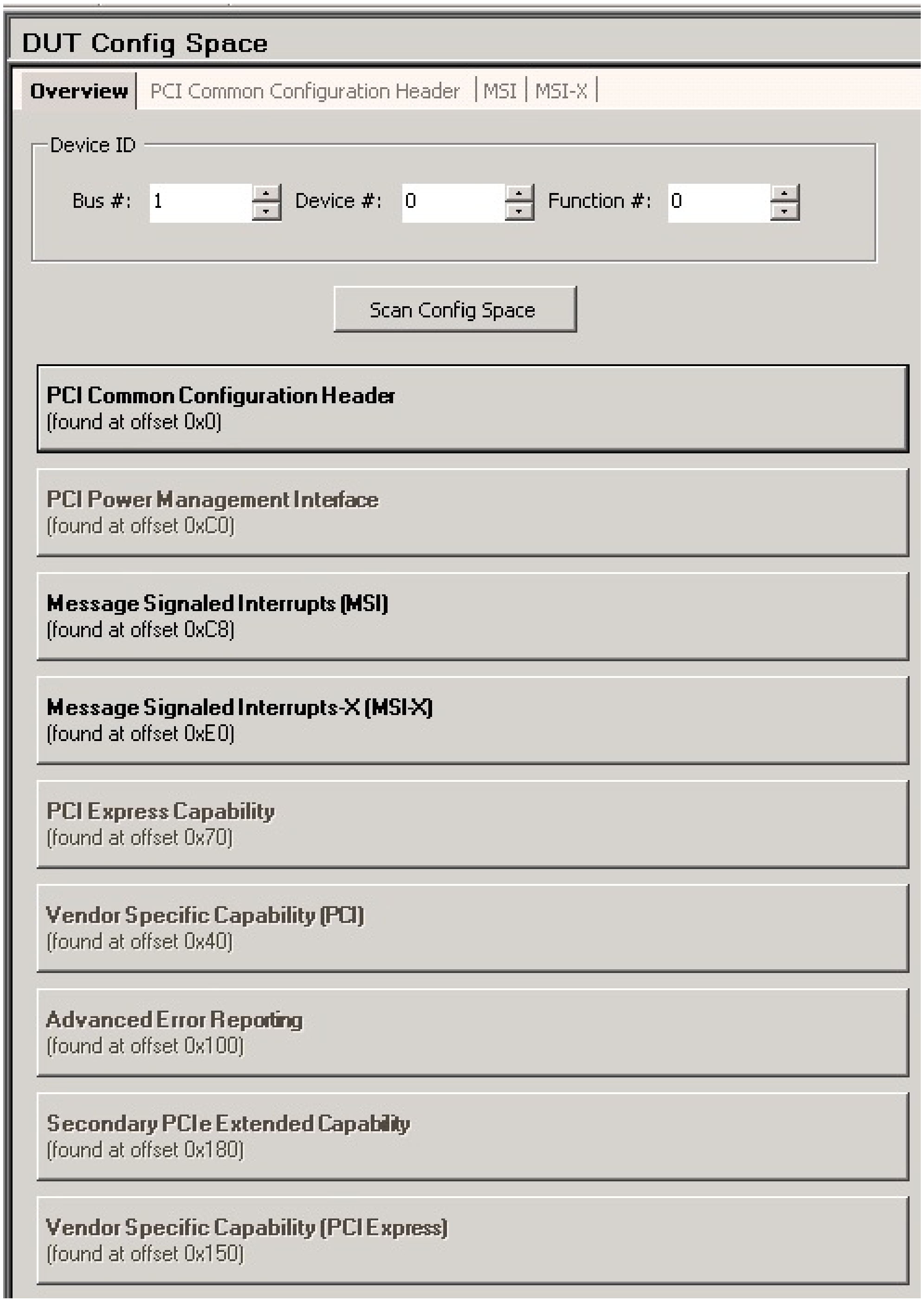
4 Click **Apply**.

Step 2 - Scan and Edit NVMe DUT's Configuration Space

Scan the DUT configuration space

After configuring the PCIe link to work in the NVMe mode, the next step is to scan and edit the NVMe DUT's configuration space. To know how to scan the configuration space of a DUT, refer to the topic “Viewing and Editing the Configuration Space of a DUT.

On scanning the configuration space of a DUT, the DUT capabilities from various registers are displayed in the DUT Config Space page of the GUI.



DUT Config Space

Device ID

Bus #: 1 Device #: 0 Function #: 0

PCI Common Configuration Header
(found at offset 0x0)

PCI Power Management Interface
(found at offset 0xC0)

Message Signaled Interrupts (MSI)
(found at offset 0xC8)

Message Signaled Interrupts-X (MSI-X)
(found at offset 0xE0)

PCI Express Capability
(found at offset 0x70)

Vendor Specific Capability (PCI)
(found at offset 0x40)

Advanced Error Reporting
(found at offset 0x100)

Secondary PCIe Extended Capability
(found at offset 0x180)

Vendor Specific Capability (PCI Express)
(found at offset 0x150)

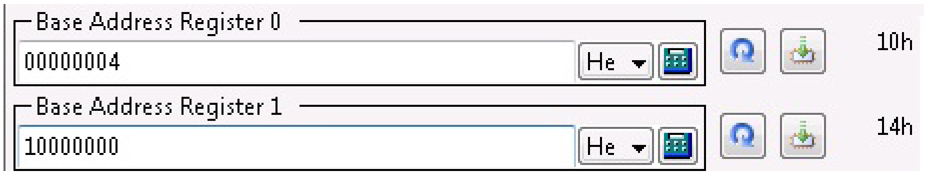
Out of these capabilities, ensure that you must set up / edit the following register values specifically for the NVMe setup.

Configure BAR 0 and 1 of DUT

BAR 0 points to NVMe controller registers. Therefore, you must initialize BAR 0 by mapping it to a memory address for the NVMe controller registers. If BAR 0 of DUT is 64-bit BAR, then configure BAR 0 and 1 of DUT. If BAR 0 is 32-bit BAR, then configure BAR 0 of DUT.

1 In the **DUT Config Space** page, click the **PCI Common Configuration Header** tab.

2 Edit the **Base Address Register 0** and **1** values.



Base Address Register 0: 00000004 He 10h

Base Address Register 1: 10000000 He 14h

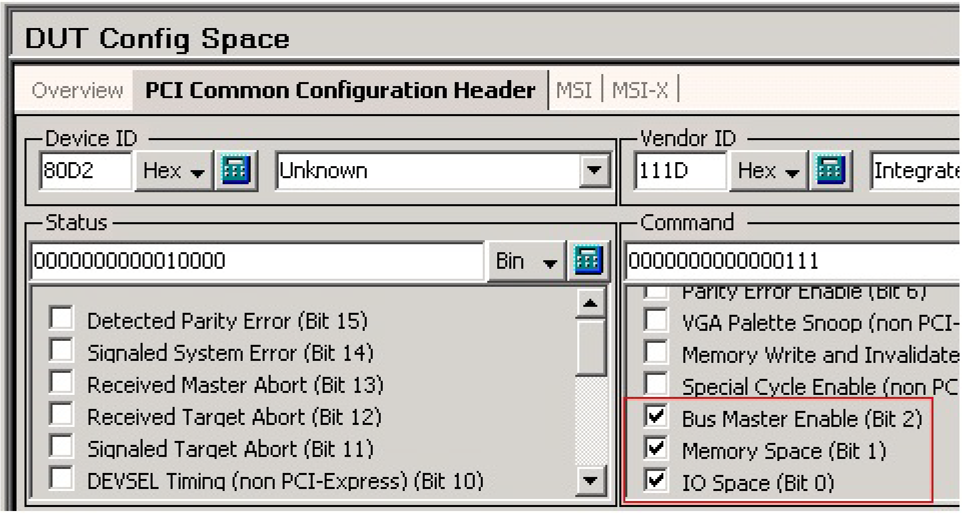
3 Click the button displayed with **Base Address Register 0** and **Base Address Register 1** to write the edited BAR values to the DUT registers.

Enable the Required Bits

1 In the **DUT Config Space** page, click the **PCI Common Configuration Header** tab.

2 Select the **Bus Master Enable**, **Memory Space Bit**, and **IO Space Bit** checkboxes in the **Command** section.

3 Click the button displayed with **Command** to write the enabled bit values to the DUT register.



DUT Config Space

Overview | PCI Common Configuration Header | MSI | MSI-X |

Device ID: 80D2 Hex Unknown

Vendor ID: 111D Hex Integrate

Status: 0000000000010000 Bin

□ Detected Parity Error (Bit 15)
□ Signaled System Error (Bit 14)
□ Received Master Abort (Bit 13)
□ Received Target Abort (Bit 12)
□ Signaled Target Abort (Bit 11)
□ DEVSEL Timing (non PCI-Express) (Bit 10)

Command: 0000000000000111

□ Parity Error Enable (Bit 6)
□ VGA Palette Snoop (non PCI-
□ Memory Write and Invalidate
□ Special Cycle Enable (non PC
☑ Bus Master Enable (Bit 2)
☑ Memory Space (Bit 1)
☑ IO Space (Bit 0)

Step 3 - Enable and configure the Interrupt (MSI and MSI-X) Capabilities of DUT

Based on the interrupt mechanism option that you want to use, you need to configure the DUT's capabilities for that interrupt mechanism. For instance, if you want to use MSI-X, then you need to enable the MSI-X capabilities of DUT and configure/edit the MSI-X table. Exerciser can then recognize and handle MSI-X interrupts received from DUT.

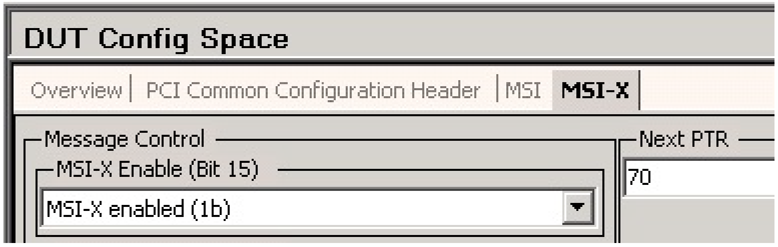
To configure the interrupt capabilities of DUT

1 Click the **DUT Config Space** icon from the left navigation pane of the GUI.

2 Click the **MSI** or **MSI-X** tab based on the interrupt mechanism capabilities that you want to set/edit.

3 To configure the MSI-X capability:

a Select the **MSI-X Enabled (1b)** option from the **MSI-X Enable** field.



Message Control

MSI-X Enable (Bit 15): MSI-X enabled (1b)

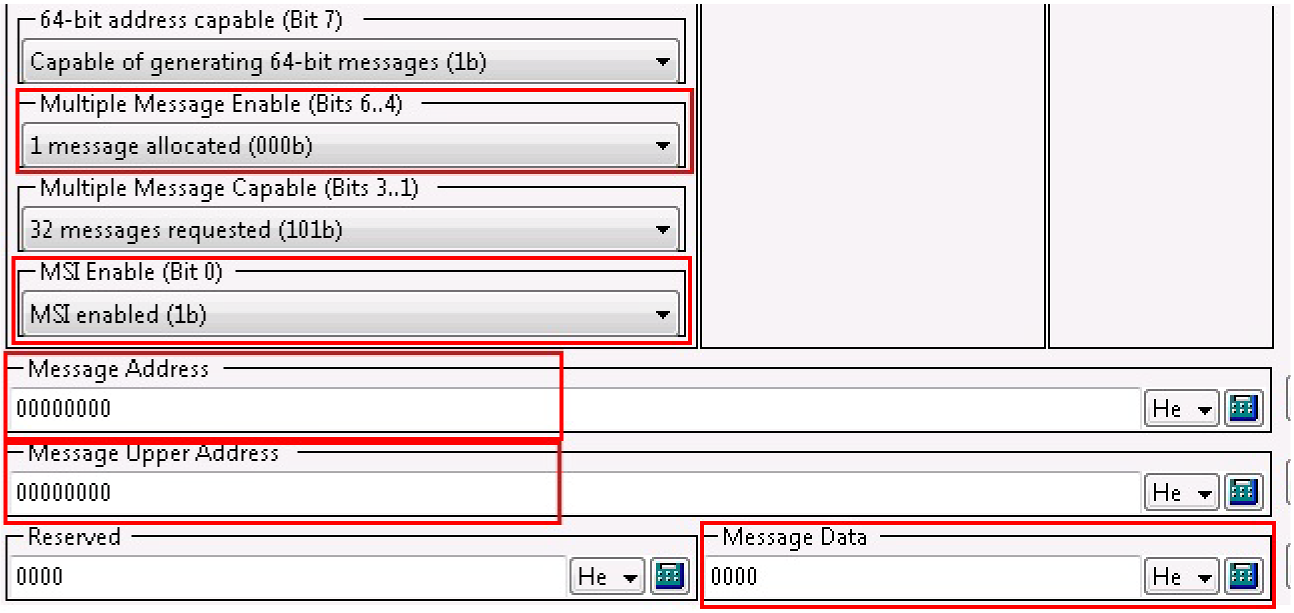
Next PTR: 70

b Click **Apply** to write changes to the DUT register.

c Configure the MSI-X table as described on page 183.

4 To configure the MSI capability:

a Configure the MSI fields highlighted in the following figure. Select the **MSI Enabled (1b)** option from the **MSI Enable** field. If the DUT supports multiple messages MSI, select the number of messages to be allocated from the **Multiple Message Enable** bit. Specify the **Message Address** and **Message Data**. If the DUT is 64-bit capable, then specify the **Message Upper Address**.



64-bit address capable (Bit 7): Capable of generating 64-bit messages (1b)

Multiple Message Enable (Bits 6..4): 1 message allocated (000b)

Multiple Message Capable (Bits 3..1): 32 messages requested (101b)

MSI Enable (Bit 0): MSI enabled (1b)

Message Address: 00000000

Message Upper Address: 00000000

Reserved: 0000

Message Data: 0000

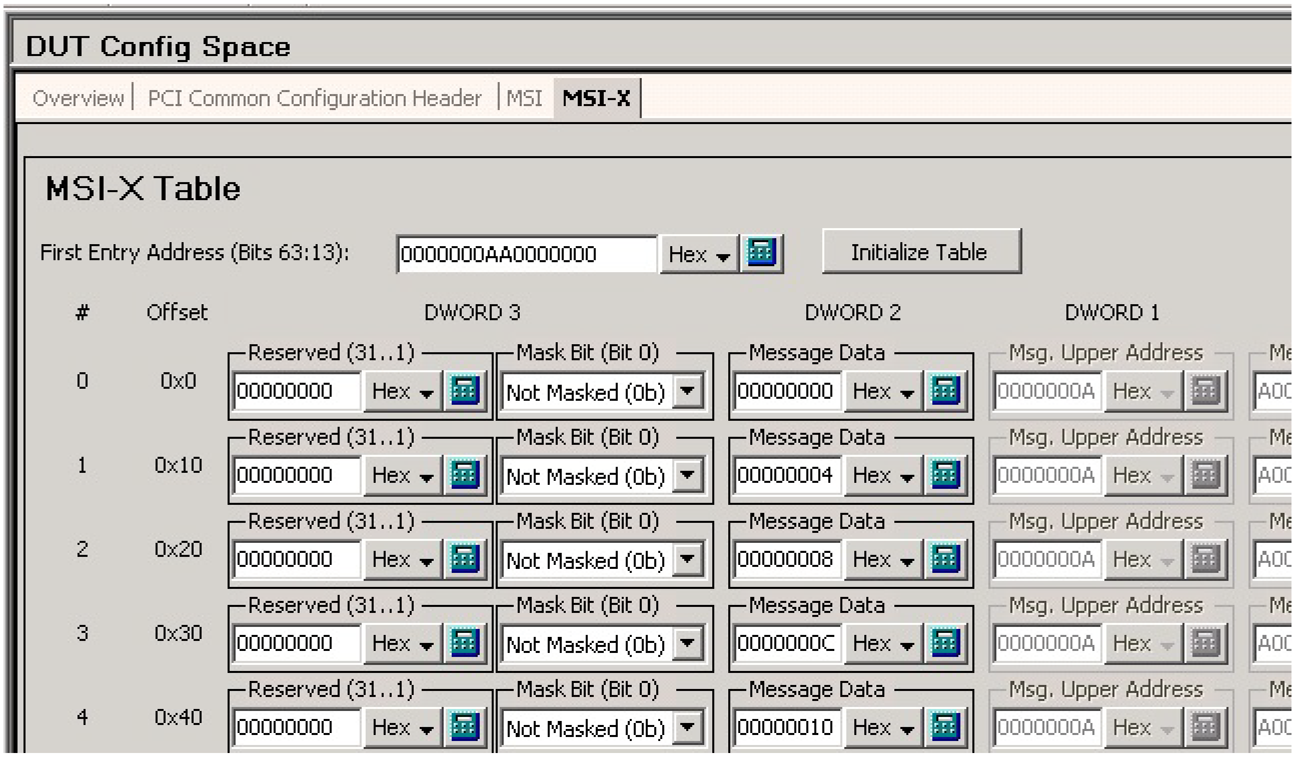
5 Click **Apply** to write changes to the DUT registers.

Configure the MSI-X table if MSI-X interrupt mechanism is used

1 Click the **MSI-X** tab of the **DUT Config Space** page.

2 In the **First Entry Address** field of the MSI-X table section, specify the address for the first MSI-X table entry and click **Initialize Table**.

3 Based on the first entry address specified by you, Exerciser automatically generates the subsequent table entries in an incremental pattern. The initialized MSI-X table entries are displayed in the tab page.



DUT Config Space

MSI-X Table

First Entry Address (Bits 63:13): 0000000AA0000000 Hex

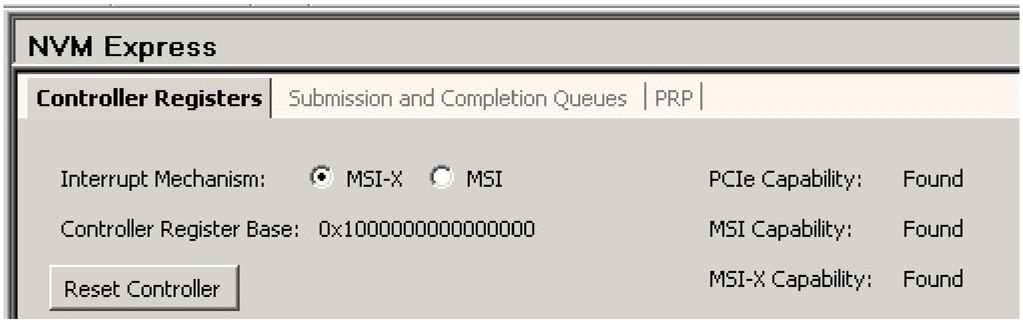
#	Offset	DWORD 3		DWORD 2	DWORD 1	
		Reserved (31..1)	Mask Bit (Bit 0)	Message Data	Msg. Upper Address	Me
0	0x0	00000000 Hex	Not Masked (0b)	00000000 Hex	0000000A Hex	A0C
1	0x10	00000000 Hex	Not Masked (0b)	00000004 Hex	0000000A Hex	A0C
2	0x20	00000000 Hex	Not Masked (0b)	00000008 Hex	0000000A Hex	A0C
3	0x30	00000000 Hex	Not Masked (0b)	0000000C Hex	0000000A Hex	A0C
4	0x40	00000000 Hex	Not Masked (0b)	00000010 Hex	0000000A Hex	A0C

Set the interrupt mechanism of DUT

Once you initialize the DUT for interrupts, you need to instruct Exerciser which interrupt mechanism to use. Exerciser can handle the following interrupt mechanism options.

- MSI
- MSI-X

During DUT config space scan, the DUT's interrupt mechanism capabilities are read and displayed in the **Controller Registers** tab of the **NVM Express** page. Based on the DUT's interrupt mechanism capabilities found, the interrupt mechanism options are enabled or disabled in this tab.



NVM Express

Controller Registers | Submission and Completion Queues | PRP

Interrupt Mechanism: ● MSI-X ○ MSI

Controller Register Base: 0x1000000000000000

PCIe Capability: Found

MSI Capability: Found

MSI-X Capability: Found

1 Click the **NVM Express** icon from the left navigation pane of the GUI.

2 In the **Controller Registers** tab, select an appropriate option from the **Interrupt Mechanism** field.

3 Click **Apply**.

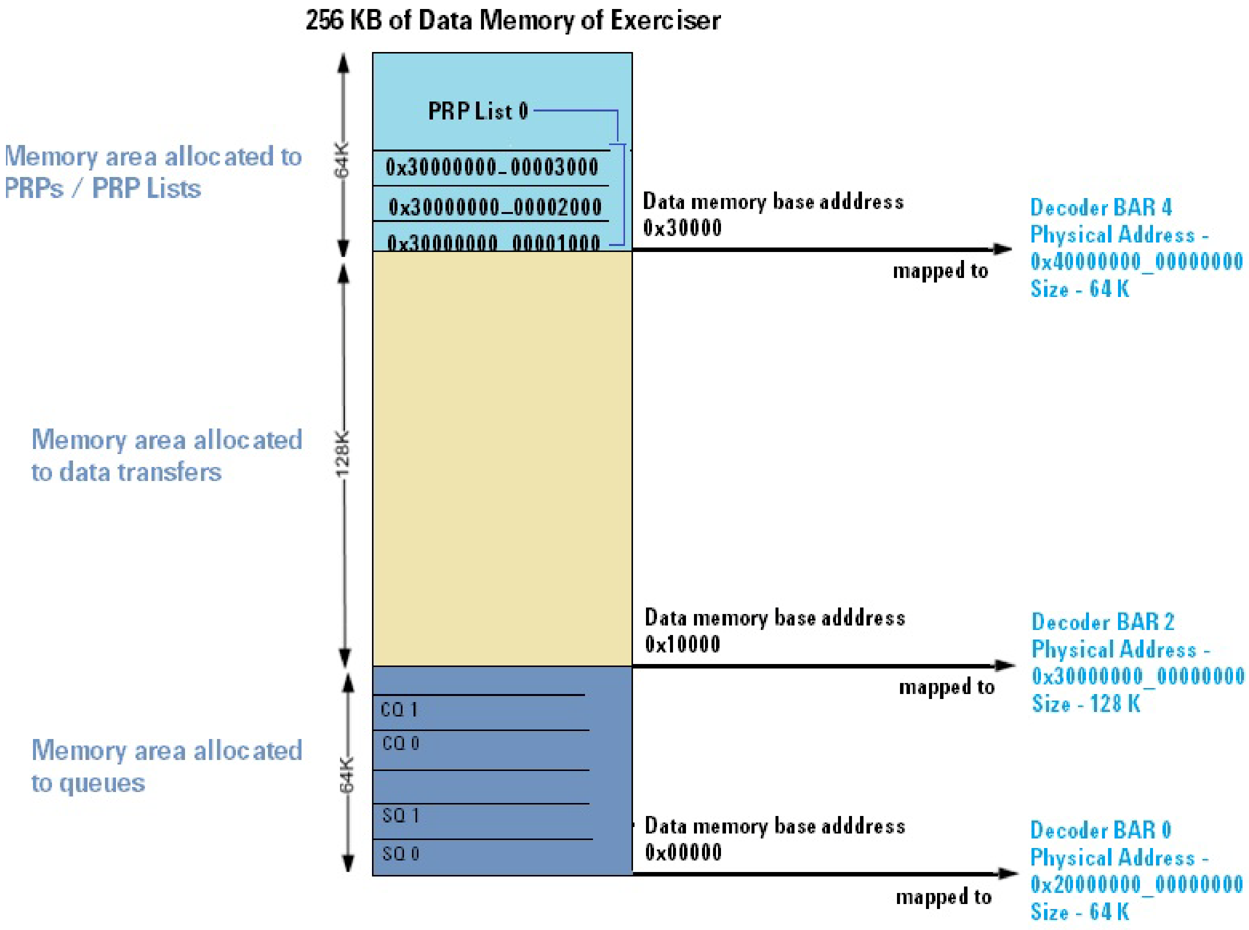
Step 4 - Configure the Required Decoders of Exerciser

Exerciser uses data memory of size 256 KB. For NVMe components such as Admin queues, I/O queues, PRP lists etc, Exerciser requires a mapping between the physical memory address and internal data memory address to be used for each of these components. Decoders are provided to define this mapping.

Organizing the data memory of Exerciser

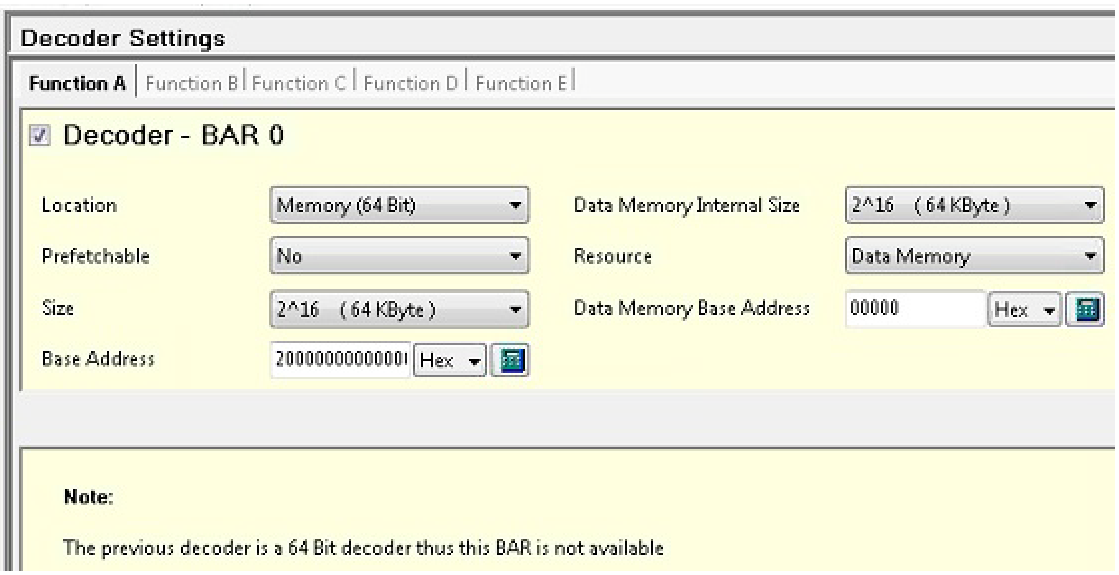
Before you start configuring decoders to define mapping, you should plan how you want to organize and use the 256 KB of data memory of Exerciser. It is recommended that you use separate memory areas and thereby separate decoders for queues, data transfer, and PRPs. This way you can ensure that data transfers do not overlap/overwrite the queues area.

The following diagram illustrates a sample organization of data memory of Exerciser and the decoder mappings for these areas.


256 KB of Data Memory of Exerciser
Memory area allocated to PRPs / PRP Lists
64K
PRP List 0
0x30000000_00003000
0x30000000_00002000
0x30000000_00001000
Data memory base adddress 0x30000
mapped to
Decoder BAR 4
Physical Address - 0x40000000_00000000
Size - 64 K
Memory area allocated to data transfers
128K
Data memory base adddress 0x10000
mapped to
Decoder BAR 2
Physical Address - 0x30000000_00000000
Size - 128 K
Memory area allocated to queues
64K
CQ 1
CQ 0
SQ 1
SQ 0
Data memory base adddress 0x00000
mapped to
Decoder BAR 0
Physical Address - 0x20000000_00000000
Size - 64 K

In the above example, the data memory area with the base address 0x00000 is allocated to queues. This memory area is of 64 K size and mapped to the physical address 0x20000000_00000000 using the Decoder BAR 0. Decoder BAR 2 and 4 are used to define mapping for the data memory areas for data buffer and PRP lists.

Once you have planned how you want to organize the data memory, you use the Decoder Settings page to configure decoders as per this planning. In the following screen, the Decoder BAR 0 is enabled and configured to map the data memory base address 0x00000 to physical address 0x20000000_00000000 as per the memory mapping planned in the above diagram.



☑ Decoder - BAR 0

Location: Memory (64 Bit)

Prefetchable: No

Size: 2^16 (64 KByte)

Base Address: 2000000000000 Hex

Data Memory Internal Size: 2^16 (64 KByte)

Resource: Data Memory

Data Memory Base Address: 00000 Hex

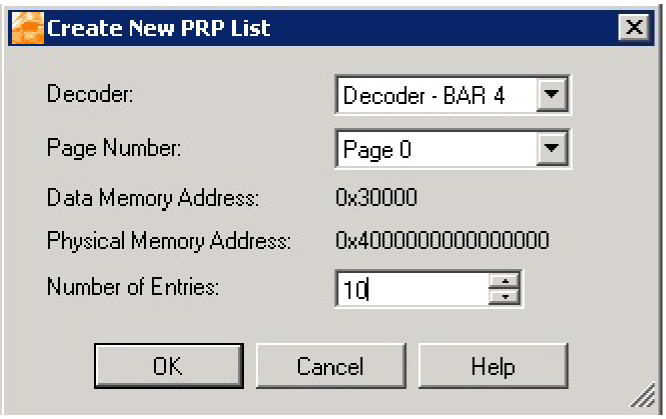
Note:

The previous decoder is a 64 Bit decoder thus this BAR is not available

To know about configuring decoders in detail, refer to the topic "Configuring Decoder Settings" on page 124.

Using the configured decoders while creating NVMe components

After you have configured decoders, you can later allocate these decoders to NVMe components such as an I/O queue or a PRP list while creating these components. The mapping that you defined for the decoder is then used and displayed for the NVMe component.



Decoder: Decoder - BAR 4

Page Number: Page 0

Data Memory Address: 0x30000

Physical Memory Address: 0x4000000000000000

Number of Entries: 10

Recording Decoder Mappings

You can record the decoder mapping used for various NVMe components in the Decoder Mapping sheet (see page 210) provided in this Online help. This can help you in avoiding overlapping or overwriting memory areas.

Step 5 - Initialize the Admin Submission and Completion Queues

After configuring decoders, the next step is to initialize the Admin submission and completion queues.

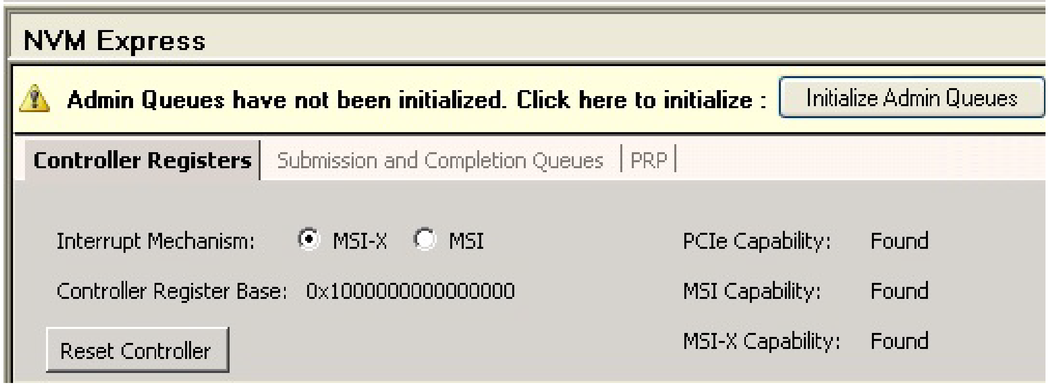
When you scan the configuration space of DUT, NVMe controller capabilities are also read and displayed in the **Controller Registers** tab of the **NVM Express** page. View and edit these NVMe controller capabilities, as needed. For instance, you can set the arbitration mechanism, initialize the memory page size, and select the I/O command set to be used.

Out of these controller capabilities, you must initialize the Admin submission and completion queues as described below.

Initialize the Admin queues

1 Click the **NVM Express** icon from the left navigation pane of the GUI.

2 In the **Controller Registers** tab, click the **Initialize Admin Queues** button.


NVM Express

Admin Queues have not been initialized. Click here to initialize :

Interrupt Mechanism: ● MSI-X ○ MSI

Controller Register Base: 0x1000000000000000

PCIe Capability: Found

MSI Capability: Found

MSI-X Capability: Found

If you do not see the **Initialize Admin Queues** button, then it

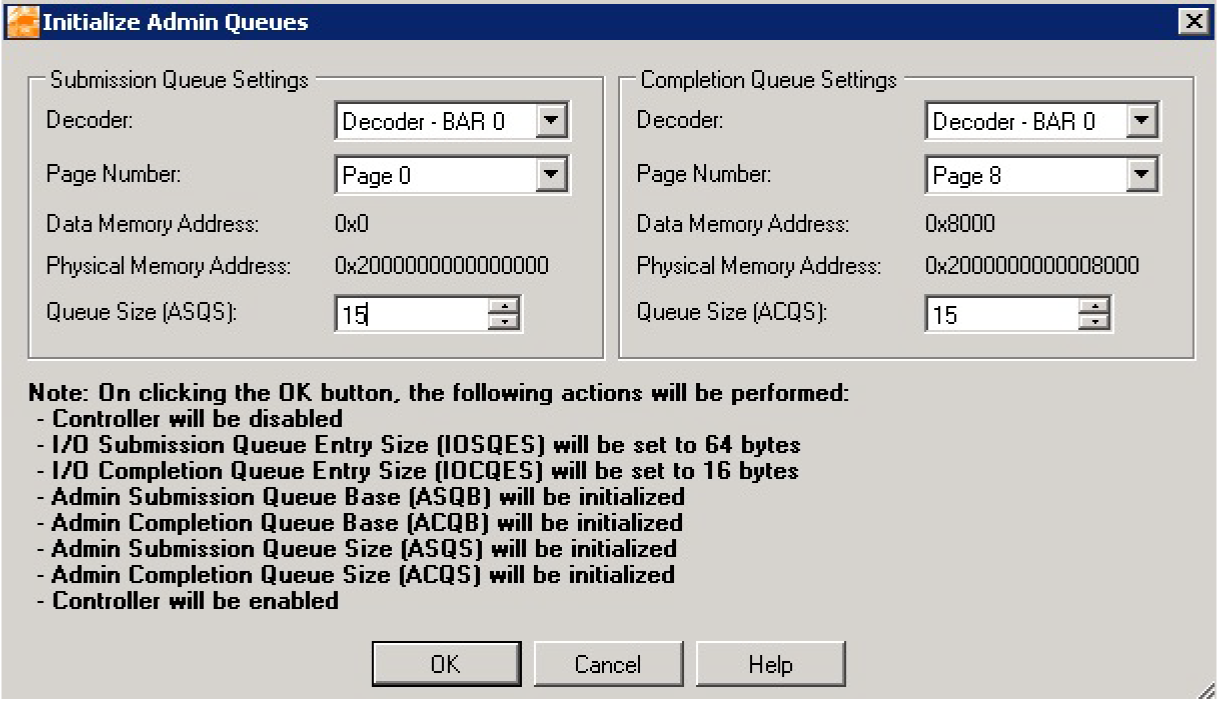
- either indicates that the Admin queues have already been initialized.
- or there may be a situation when a prerequisite such as the link being up or DUT config space being configured properly is not met resulting in the display of an appropriate warning instead of the I**nitialize Admin Queues** button. In such a situation, first ensure that the prerequisite is met.

3 In the **Initialize Admin Queues** dialog box, select a decoder for Admin submission queue and Admin completion queue. It is recommended that you note down the decoders used for these queues in the Decoder mapping sheet (see page 210) to avoid overlapping or overwriting to the data memory areas.

4 Select a page offset (page number) within the selected decoder that you want to allocate to the Admin submission queue and completion queue. It is recommended that you note down the decoder's pages used for the queues in the Decoder mapping sheet (see page 210) to avoid overlapping or overwriting to the data memory areas.

On selecting decoders and pages for queues, the mapping that you defined for the selected decoders is displayed as **Data Memory Address** and **Physical Memory Address**.

5 Specify the size for the Admin submission and completion queues. The queue is created with the N+1 size where N is the specified size of queue. For instance, if you specified 15 as the size of the queue, then the queue can take commands from 0 to 15 making the actual size of the queue as 16.



Initialize Admin Queues

Submission Queue Settings

Decoder: Decoder - BAR 0

Page Number: Page 0

Data Memory Address: 0x0

Physical Memory Address: 0x2000000000000000

Queue Size (ASQS): 15

Completion Queue Settings

Decoder: Decoder - BAR 0

Page Number: Page 8

Data Memory Address: 0x8000

Physical Memory Address: 0x2000000000008000

Queue Size (ACQS): 15

Note: On clicking the OK button, the following actions will be performed:
- Controller will be disabled
- I/O Submission Queue Entry Size (IOSQES) will be set to 64 bytes
- I/O Completion Queue Entry Size (IOCQES) will be set to 16 bytes
- Admin Submission Queue Base (ASQB) will be initialized
- Admin Completion Queue Base (ACQB) will be initialized
- Admin Submission Queue Size (ASQS) will be initialized
- Admin Completion Queue Size (ACQS) will be initialized
- Controller will be enabled

6 Click **OK**.

On clicking OK, Exerciser proceeds with disabling the controller to initialize the admin queues with the specified attributes. Exerciser then initializes the admin queues and brings the controller back to the enabled state.

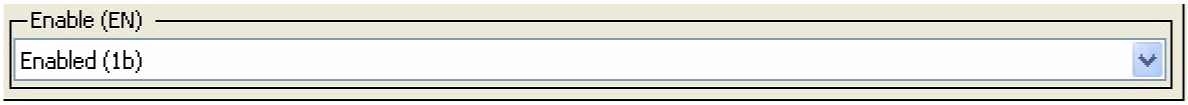
Check the Enable bit

After initializing the Admin queues, Exerciser automatically enables the controller. However, it is recommended that you check the Enable bit of DUT and enable it, if needed, before you proceed further. By setting the **Enable (EN)** bit of the **Controller Configuration** register to 1, you enable the NVMe controller to process NVMe commands and post completions.

1 Click the **NVM Express** icon from the left navigation pane of the GUI.

2 In the **Controller Registers** tab, scroll down to the **CC - Controller Configuration** register.

3 In the **CC - Controller Configuration** register, select **Enabled (1b)** from the **Enable** field.



Enable (EN)

Enabled (1b)

4 Click the button displayed with **CC - Controller Configuration** register to write the Enable bit value to the DUT register.

Check the Ready bit

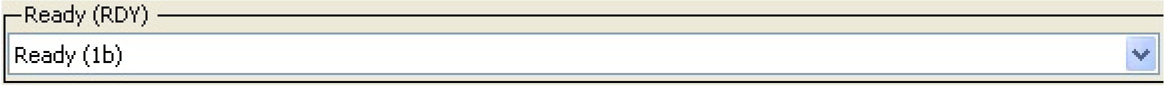
When you set the Enable bit of controller to 1, the controller sets the Ready bit to 1 when it is ready to process NVMe commands.

After setting the Enable bit to 1, verify that the controller is ready to process commands by checking the Ready bit.

1 Click the **NVM Express** icon from the left navigation pane of the GUI.

2 In the **Controller Registers** tab, scroll down to the **CSTS - Controller Status** register.

3 In the **CSTS - Controller Status** register, verify that the value of the **Ready (RDY)** field is set to **Ready (1b)**.



Ready (RDY)

Ready (1b)

4 To read the current value of this bit from the DUT register, click the ↻ button displayed with **CSTS - Controller Status** register.

Editing Admin Queue Attributes

If required, you can set/change the Admin Queue attributes by directly editing the relevant registers such as AQA, ASQ, and ACQ registers displayed in the **Controller Registers** tab of the **NVM Express** page. Before doing so, make sure that you disable the controller by selecting **Disabled (0b)** from the **Enable** field in the **CC - Controller Configuration** register.

Creating, Deleting, and Viewing Submission and Completion Queues

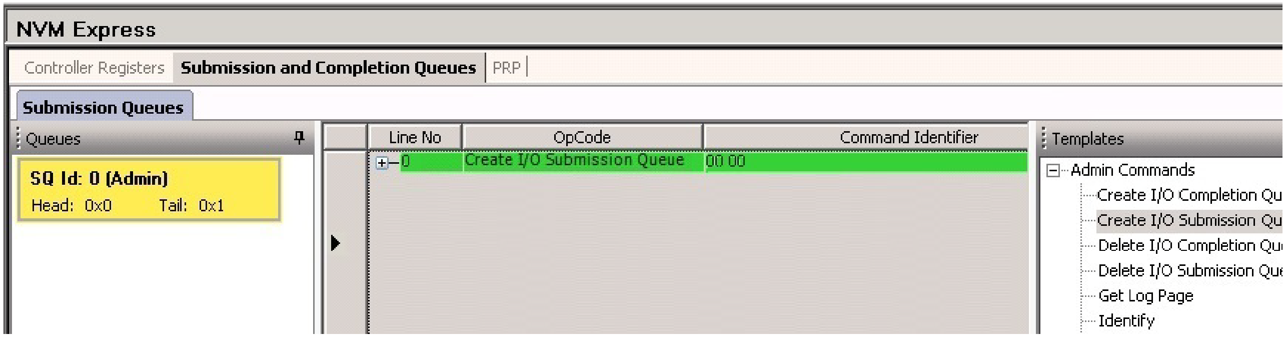
You can use Exerciser to submit commands for the creation of I/O submission and completion queues. You can create multiple I/O submission and completion queues. Each submission queue needs to be mapped to a completion queue. Multiple submission queues can share the same completion queue.

NOTE You can create a maximum of 64 I/O submission queues and 64 I/O completion queues.

To create an I/O submission or completion queue

1. Click the **NVM Express** icon from the left navigation pane of the GUI.
2. Click the **Submission and Completion Queues** tab.
3. To add the queue creation command to the Admin queue, click the **SQ Id : 0 (Admin)** option displayed under **Queues** section on the left.
4. Drag the **Create I/O Submission Queue** or **Create I/O Completion Queue** template from the **Templates** pane on the right and drop it to the pane displayed in the center pane.

   An instance of the queue creation template is added to the list of commands to be submitted.
5. Click the + icon displayed with the newly created command instance to view the command details.

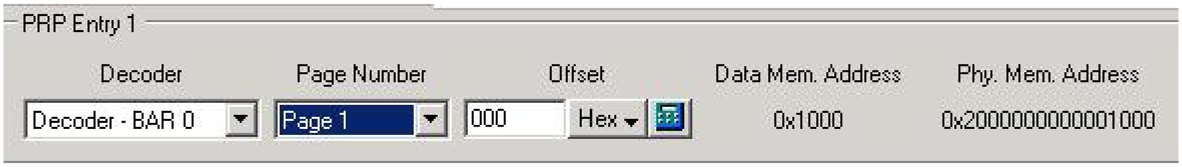


SQ Id: 0 (Admin)
Head: 0x0 Tail: 0x1

Line No	OpCode	Command Identifier
0	Create I/O Submission Queue	00 00

6. Double-click the command instance to edit the queue details, as needed in the **Edit Packet** dialog box. For instance, to associate the submission queue to a completion queue or to configure the PRP entries to be used for the queue.
7. In the **PRP Entry** section, select a Decoder that you want Exerciser to use for the PRP entry of the queue. The physical memory address and internal data memory address mapping of the selected decoder is then used for the queue. It is recommended that you note down the decoder used for the queue in the Decoder mapping sheet (see page 210) to avoid overlapping or overwriting to the data memory areas.
8. Select a page offset (**Page number**) within the selected decoder that you want to allocate to the PRP entry of the queue. It is recommended that you note down the decoder's page used for the queue in the Decoder mapping sheet (see page 210) to avoid overlapping or overwriting to the data memory areas.

   The internal **Data Memory Address** and **Physical Memory Address** applicable for the queue are displayed based on the decoder and page offset you selected for the queue.



PRP Entry 1

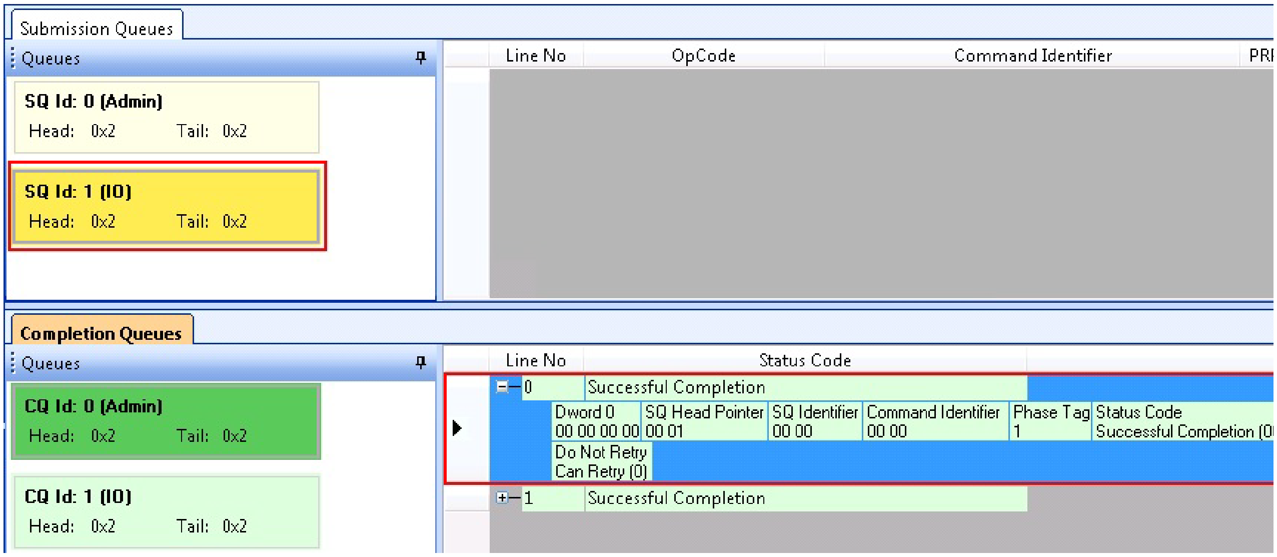
Decoder	Page Number	Offset	Data Mem. Address	Phy. Mem. Address
Decoder - BAR 0	Page 1	000 Hex	0x1000	0x2000000000001000

9 The queue creation command is now ready for submission. Click **Apply** to confirm and submit the command for processing to DUT.

Once the command is submitted, the creation of the queue is displayed in the Exerciser GUI only when the NVMe controller executes the command successfully. The controller posts the completion status of the queue creation command in the associated Admin completion queue.

In the following example, notice that a successful completion status is displayed for the submission queue creation command in the **Completion Queues** section. As a result, a new submission queue SQ Id 1 is added to the Submission Queues section.

This queue is now available for adding NVMe commands for submission.



Submission Queues

Queues

SQ Id: 0 (Admin)
Head: 0x2 Tail: 0x2

SQ Id: 1 (IO)
Head: 0x2 Tail: 0x2

Line No	OpCode	Command Identifier	PRI

Completion Queues

Queues

CQ Id: 0 (Admin)
Head: 0x2 Tail: 0x2

CQ Id: 1 (IO)
Head: 0x2 Tail: 0x2

Line No	Status Code
0	Successful Completion
1	Successful Completion

Dword 0	SQ Head Pointer	SQ Identifier	Command Identifier	Phase Tag	Status Code
00 00 00 00	00 01	00 00	00 00	1	Successful Completion (0
Do Not Retry Can Retry (0)					

To delete an I/O submission queue

1 Click the **NVM Express** icon from the left navigation pane of the GUI.

2 Click the **Submission and Completion Queues** tab.

3 To add a queue deletion command to the Admin queue, click the **SQ Id : 0 (Admin)** option displayed under **Queues** section on the left.

4 Drag the **Delete I/O Submission Queue** or **Delete I/O Completion Queue** template from the **Templates** pane on the right and drop it to the pane displayed in the middle of the **Submission Queues** section.

An instance of the template is added to the list of commands to be submitted.

5 Click the + icon displayed with the newly created command instance to view the command details.

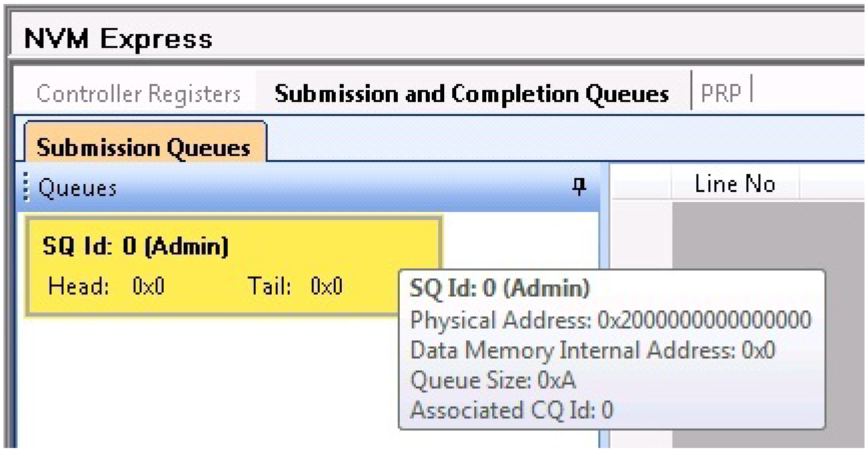
6 Double-click the command instance to edit the command details, as needed.

7 The queue deletion command is now ready for submission. Click **Apply** to confirm and submit the command to DUT for processing.

The queue is deleted from the **Submission Queues** section only when the successful completion status is posted for the queue deletion command in the Admin completion queue.

To view information about a queue

You can get useful information about a queue by hovering the mouse over the queue's entry in the **Submission Queues** or **Completion Queues** list. A tooltip is provided for each queue with information such as its Physical memory address, data memory address, and size.

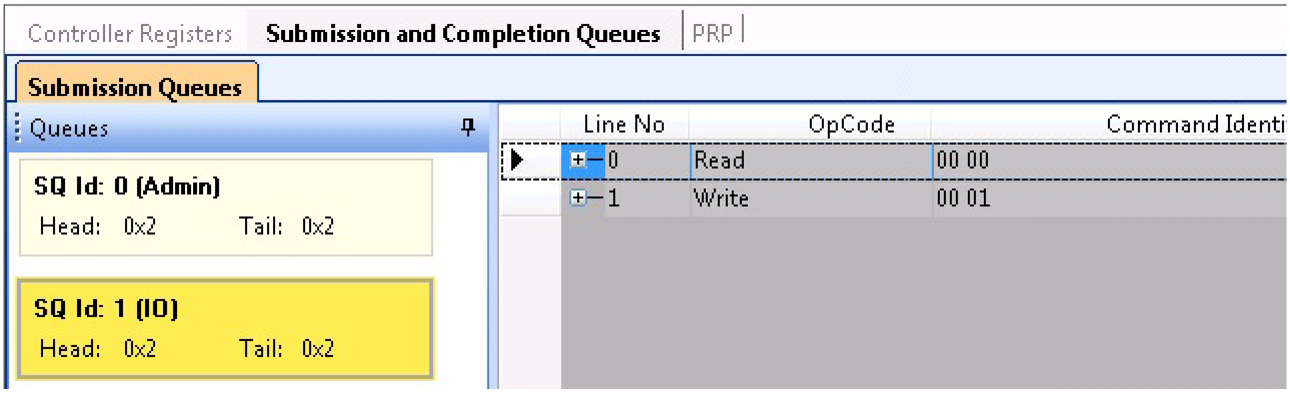


SQ Id: 0 (Admin)
Head: 0x0 Tail: 0x0

SQ Id: 0 (Admin)
Physical Address: 0x2000000000000000
Data Memory Internal Address: 0x0
Queue Size: 0xA
Associated CQ Id: 0

To view commands submitted to a submission queue

You can click a submission queue from the **Queues** section to view the NVMe commands that you submitted to that queue.



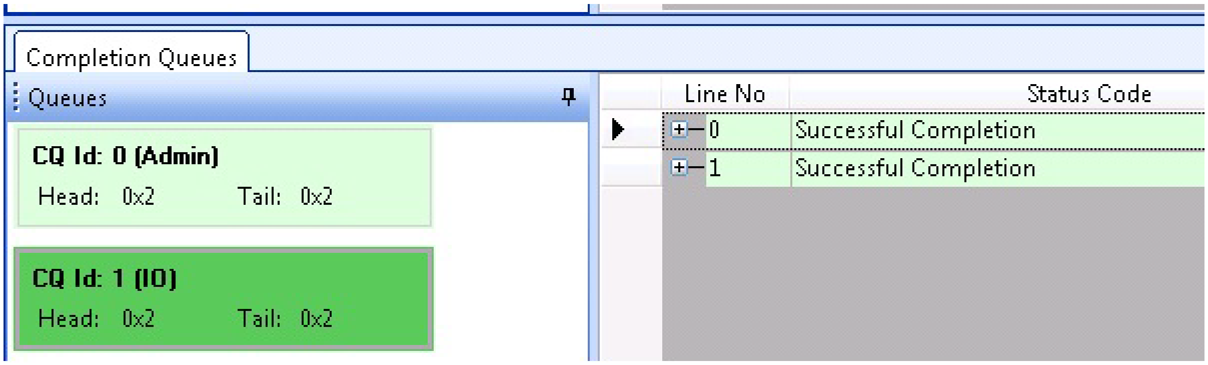
SQ Id: 0 (Admin)
Head: 0x2 Tail: 0x2

SQ Id: 1 (IO)
Head: 0x2 Tail: 0x2

Line No	OpCode	Command Identi
0	Read	00 00
1	Write	00 01

To view completions received in a completion queue

You can click a completion queue from the **Completion Queues** section to view the command completion status posted to that queue.



Completion Queues

Queues

CQ Id: 0 (Admin)
Head: 0x2 Tail: 0x2

CQ Id: 1 (IO)
Head: 0x2 Tail: 0x2

Line No	Status Code
0	Successful Completion
1	Successful Completion

Creating and Using a PRP List

Physical Region Page (PRP) entries are used in NVMe commands to indicate the physical memory locations in system memory. These memory locations are used by controller for data transfers to and from system memory. For a Read request, the PRP entry points to the data buffer where the controller should transfer the data read from the NVMe device. For a Write request, the PRP entry points to the data buffer from where the controller has to read the data to be written to the NVMe device.

These PRP entries can be:

- either directly a memory location.
- or a pointer to a memory location that provides a set of memory addresses to perform large data transfer operations. Such a memory location is referred to as a PRP list that defines a set of PRP entries.

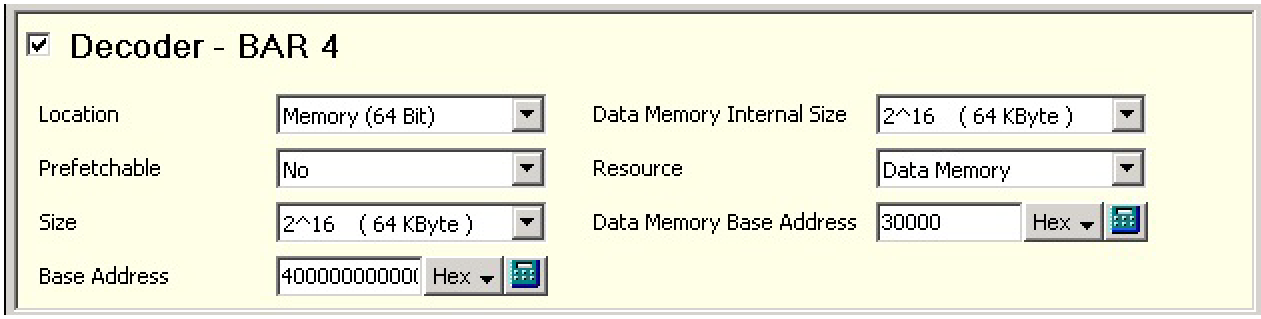
Exerciser allows you to create PRP lists and use these lists for data transfers in read/write commands that Exerciser submits to an NVMe DUT. You can also define the number of PRP entries for a PRP list.

Using decoders for PRP lists

For creating a PRP list, the Data memory of Exerciser is used. Exerciser requires a mapping between the physical memory address and internal data memory address to be used for each of these lists and for each PRP entry within a PRP list. To define this mapping, you use the **Decoder** feature of Exerciser. In the **Decoder Settings** page, you:

- enable the required decoders from the available decoders.
- configure the settings for the enabled decoder(s) and define the mapping for these decoder(s).

For instance, in the following screen, Decoder BAR 4 is enabled for PRP lists.



☑ Decoder - BAR 4

Location: Memory (64 Bit)
Prefetchable: No
Size: 2^16 (64 KByte)
Base Address: 40000000000(Hex
Data Memory Internal Size: 2^16 (64 KByte)
Resource: Data Memory
Data Memory Base Address: 30000 Hex

To know about configuring decoders in detail, refer to the topic “Step 4 - Configure the Required Decoders of Exerciser.

Once configured, you can select a decoder and associate it to a PRP list while creating this list. The mapping that you defined for this decoder is used and displayed for the PRP list.

To create a PRP list

1 Click the **NVM Express** icon from the left navigation pane of the GUI.

2 Click the **PRP** tab.

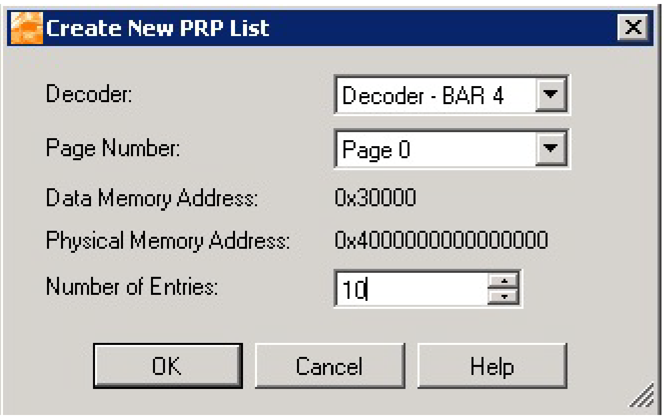
3 Click the ✚ button displayed in the **PRP Lists** pane on the left.

The **Create New PRP List** dialog box is displayed.

4 Select a **Decoder** that you want Exerciser to use for the PRP list. The physical memory address and internal data memory address mapping of the selected decoder is then used for the PRP list. It is recommended that you note down the decoder used for the PRP list in the Decoder mapping sheet (see page 210) to avoid overlapping or overwriting to the data memory areas.

5 Select a page offset (**Page number**) within the selected decoder that you want to allocate to the PRP list. It is recommended that you note down the decoder's page used for the PRP list in the Decoder mapping sheet (see page 210) to avoid overlapping or overwriting to the data memory areas.

The internal **Data Memory Address** and **Physical Memory Address** applicable for the PRP list are displayed based on the decoder and page offset you selected for the PRP list.



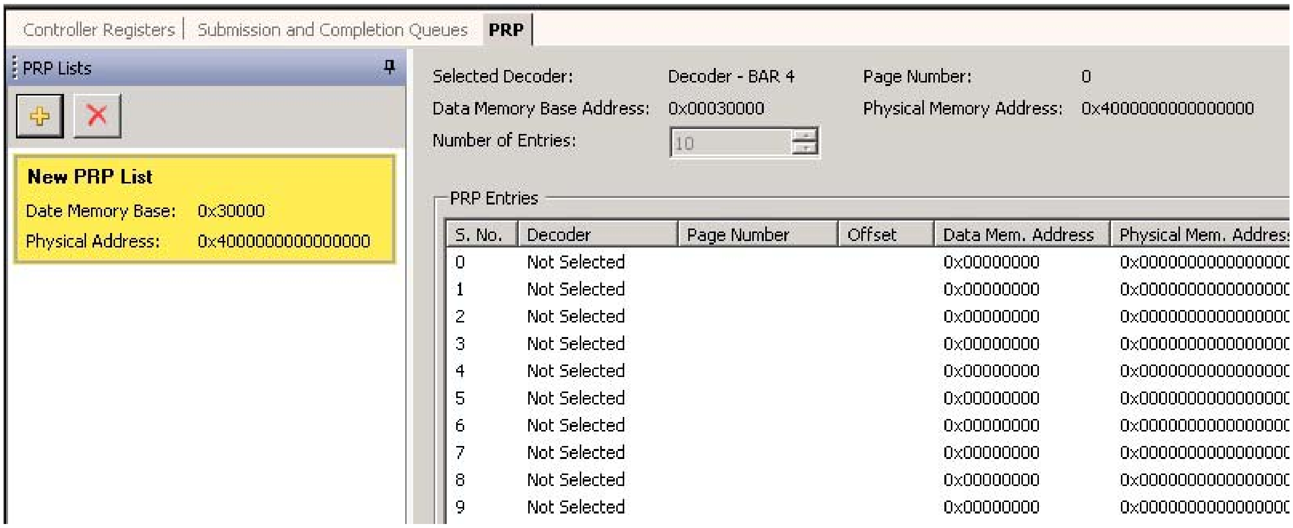
Create New PRP List

Decoder:	Decoder - BAR 4
Page Number:	Page 0
Data Memory Address:	0x30000
Physical Memory Address:	0x4000000000000000
Number of Entries:	10

6 In the **Number of Entries** field, select the number of PRP entries that the list should contain.

7 Click **OK**.

8 A PRP list is created and displayed in the **PRP Lists** pane on the left. Select this list to display its PRP entries in the right pane.



New PRP List
Date Memory Base: 0x30000
Physical Address: 0x4000000000000000

Selected Decoder: Decoder - BAR 4
Page Number: 0
Data Memory Base Address: 0x00030000
Physical Memory Address: 0x4000000000000000
Number of Entries: 10

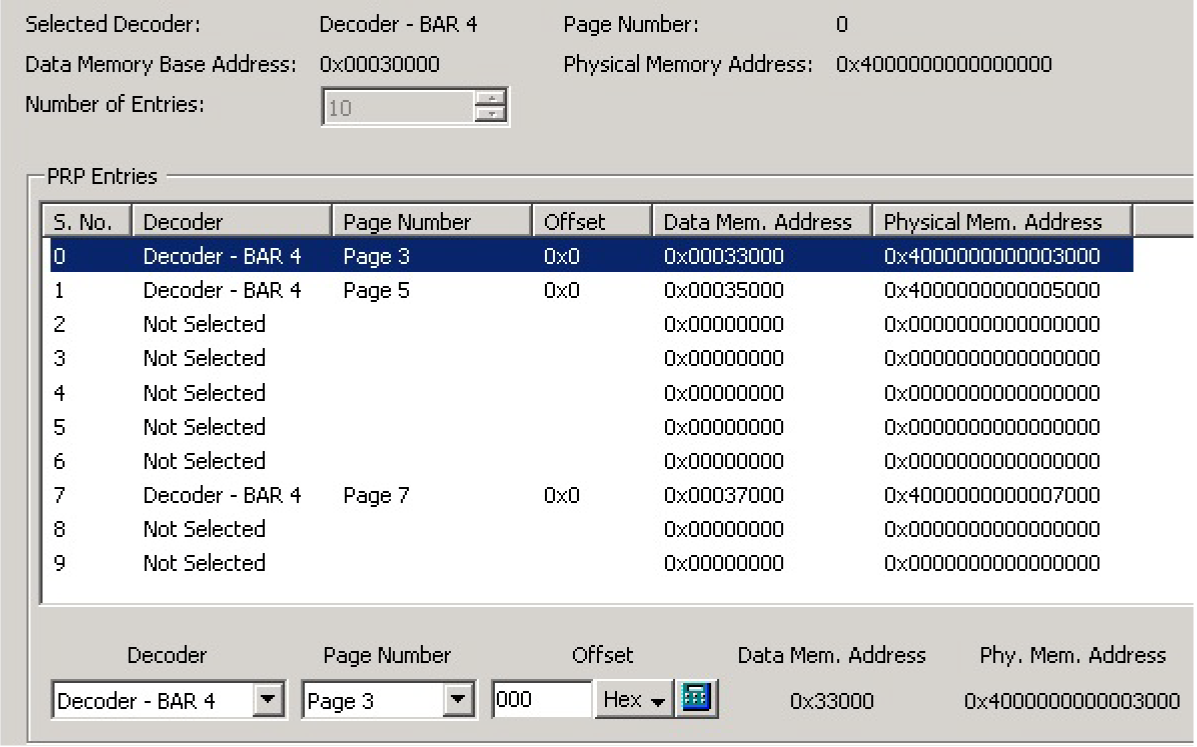
PRP Entries

S. No.	Decoder	Page Number	Offset	Data Mem. Address	Physical Mem. Addres
0	Not Selected			0x00000000	0x0000000000000000
1	Not Selected			0x00000000	0x0000000000000000
2	Not Selected			0x00000000	0x0000000000000000
3	Not Selected			0x00000000	0x0000000000000000
4	Not Selected			0x00000000	0x0000000000000000
5	Not Selected			0x00000000	0x0000000000000000
6	Not Selected			0x00000000	0x0000000000000000
7	Not Selected			0x00000000	0x0000000000000000
8	Not Selected			0x00000000	0x0000000000000000
9	Not Selected			0x00000000	0x0000000000000000

9 From the right pane, click each PRP entry and select the following for each entry:

- a decoder
- page number (page offset within the selected decoder)
- offset within the selected page

On the basis of the above selections, the base address in the data memory of Exerciser and the physical memory base address applicable for the PRP entry are displayed.



Selected Decoder: Decoder - BAR 4
Page Number: 0
Data Memory Base Address: 0x00030000
Physical Memory Address: 0x4000000000000000
Number of Entries: 10

PRP Entries

S. No.	Decoder	Page Number	Offset	Data Mem. Address	Physical Mem. Address
0	Decoder - BAR 4	Page 3	0x0	0x00033000	0x4000000000003000
1	Decoder - BAR 4	Page 5	0x0	0x00035000	0x4000000000005000
2	Not Selected			0x00000000	0x0000000000000000
3	Not Selected			0x00000000	0x0000000000000000
4	Not Selected			0x00000000	0x0000000000000000
5	Not Selected			0x00000000	0x0000000000000000
6	Not Selected			0x00000000	0x0000000000000000
7	Decoder - BAR 4	Page 7	0x0	0x00037000	0x4000000000007000
8	Not Selected			0x00000000	0x0000000000000000
9	Not Selected			0x00000000	0x0000000000000000

Decoder	Page Number	Offset		Data Mem. Address	Phy. Mem. Address
Decoder - BAR 4	Page 3	000	Hex	0x33000	0x4000000000003000

10 Click **Apply** when all PRP entries of the list are mapped to a decoder and page number.

To use a PRP list in an NVMe command

Once a PRP list is created, you can use it in a command sent from Exerciser to an NVMe DUT.

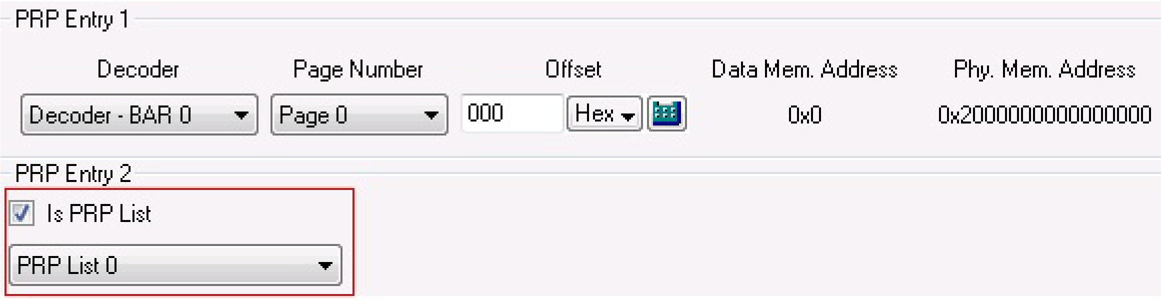
1 Click the **NVM Express** icon from the left navigation pane of the GUI.

2 Click the **Submission and Completion Queues** tab.

3 From the **Submission Queues** tab, drag and drop an appropriate command template from the Templates section to the left.

4 Double-click the NVMe command that you created in the previous step.

5 In the **Edit Packet** dialog box, select the **Is PRP List** checkbox from the **PRP Entry** section and then select the PRP list that you want to use for data transfer.



PRP Entry 1

Decoder	Page Number	Offset		Data Mem. Address	Phy. Mem. Address
Decoder - BAR 0	Page 0	000	Hex	0x0	0x2000000000000000

PRP Entry 2

☑ Is PRP List

PRP List 0

6 Click **OK**.

To delete a PRP list

1 Select the PRP list from the **PRP Lists** section.

2 Click the ❌ button.

The PRP List and its PRP entries are deleted.

Creating and Using a SGL List

Scatter Gather List (SGL) are used in NVMe commands to indicate the physical memory locations in system memory. These memory locations are used by controller for data transfers to and from system memory. A SGL contains one or more SGL segments. For a Read request, the SGL entry points to the data buffer where the controller should transfer the data read from the NVMe device. For a Write request, the SGL entry points to the data buffer from where the controller has to read the data to be written to the NVMe device.

These SGL can be:

- either directly a memory location.
- or a pointer to a memory location that provides a set of memory addresses to perform large data transfer operations. Such a memory location is referred to as a PRP list that defines a set of PRP entries.

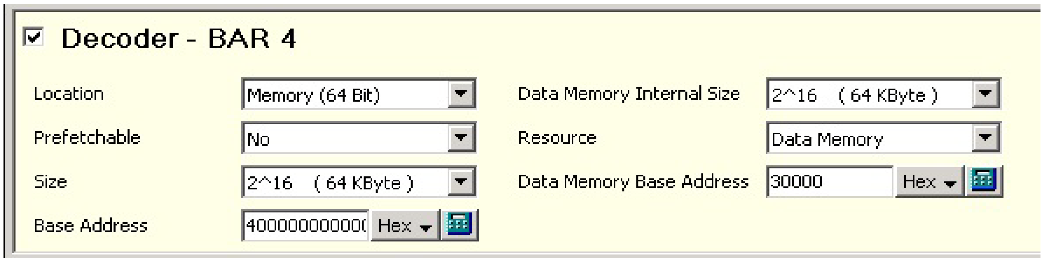
Exerciser allows you to create SGL lists and use these lists for data transfers in read/write commands that Exerciser submits to an NVMe DUT. You can also define the number of SGL segment entries for a SGL list.

Using decoders for SGL lists

For creating a SGL list, the Data memory of Exerciser is used. Exerciser requires a mapping between the physical memory address and internal data memory address to be used for each of these lists and for each SGL segment entry within a SGL list. To define this mapping, you use the **Decoder** feature of Exerciser. In the **Decoder Settings** page, you:

- enable the required decoders from the available decoders.
- configure the settings for the enabled decoder(s) and define the mapping for these decoder(s).

For instance, in the following screen, Decoder BAR 4 is enabled for SGL lists.



☑ Decoder - BAR 4

Location: Memory (64 Bit)
Prefetchable: No
Size: 2^16 (64 KByte)
Base Address: 400000000000 Hex
Data Memory Internal Size: 2^16 (64 KByte)
Resource: Data Memory
Data Memory Base Address: 30000 Hex

To know about configuring decoders in detail, refer to the topic "Step 4 - Configure the Required Decoders of Exerciser.

Once configured, you can select a decoder and associate it to a SGL list while creating this list. The mapping that you defined for this decoder is used and displayed for the SGL list.

To create a SGL list

1 Click the **NVM Express** icon from the left navigation pane of the GUI.

2 Click the **SGL** tab.

3 Click the button displayed in the **SGL Lists** pane on the left.

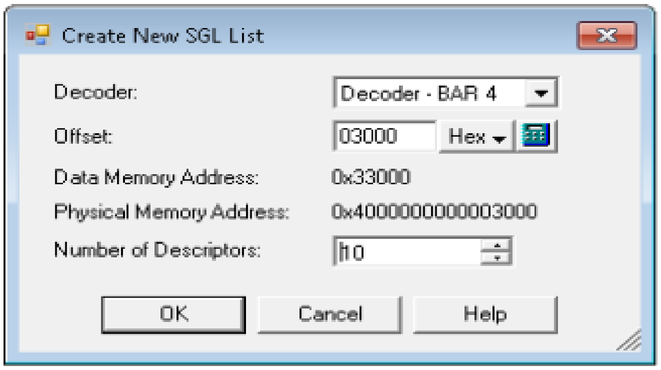
The **Create New SGL List** dialog box is displayed.

4 Select a **Decoder** that you want Exerciser to use for the SGL list. The physical memory address and internal data memory address mapping of the selected decoder is then used for the SGL list. It is

recommended that you note down the decoder used for the SGL list in the Decoder mapping sheet (see page 210) to avoid overlapping or overwriting to the data memory areas.

5 Select an **offset** within the selected decoder that you want to allocate to the SGL list. It is recommended that you note down the decoder's page used for the SGL list in the Decoder mapping sheet (see page 210) to avoid overlapping or overwriting to the data memory areas.

The internal **Data Memory Address** and **Physical Memory Address** applicable for the SGL list are displayed based on the decoder and page offset you selected for the SGL list.



Create New SGL List

Decoder: Decoder - BAR 4

Offset: 03000 Hex

Data Memory Address: 0x33000

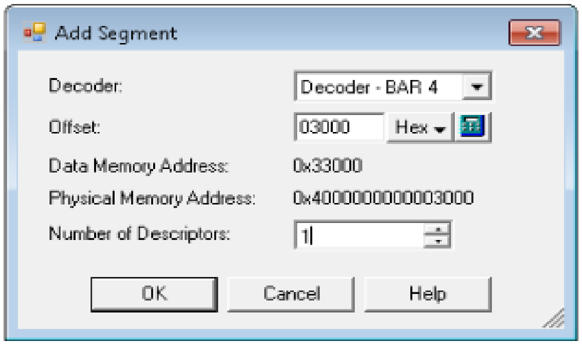
Physical Memory Address: 0x4000000000003000

Number of Descriptors: 10

6 In the **Number of Descriptors** field, select the number of descriptors that the first segment of the list should contain.

7 Click **OK**.

8 A SGL list is created with SGL segments section on the left. You can create one or more SGL segment under SGL list. Select SGL segment list to display its SGL entries in the right pane.



Add Segment

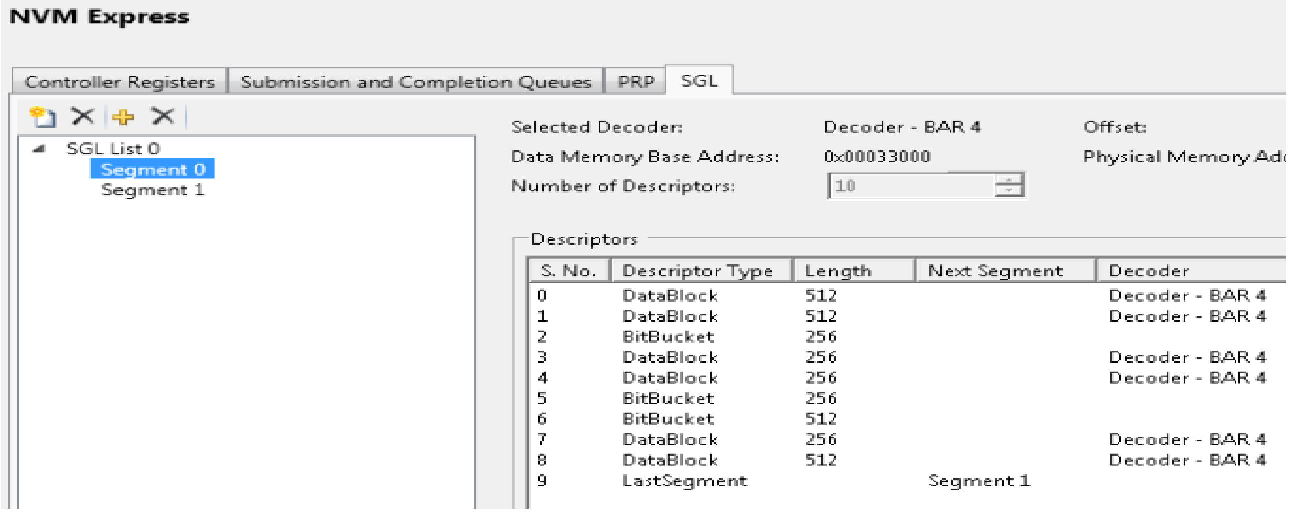
Decoder: Decoder - BAR 4

Offset: 03000 Hex

Data Memory Address: 0x33000

Physical Memory Address: 0x4000000000003000

Number of Descriptors: 1



Selected Decoder: Decoder - BAR 4

Data Memory Base Address: 0x00033000

Number of Descriptors: 10

Offset:

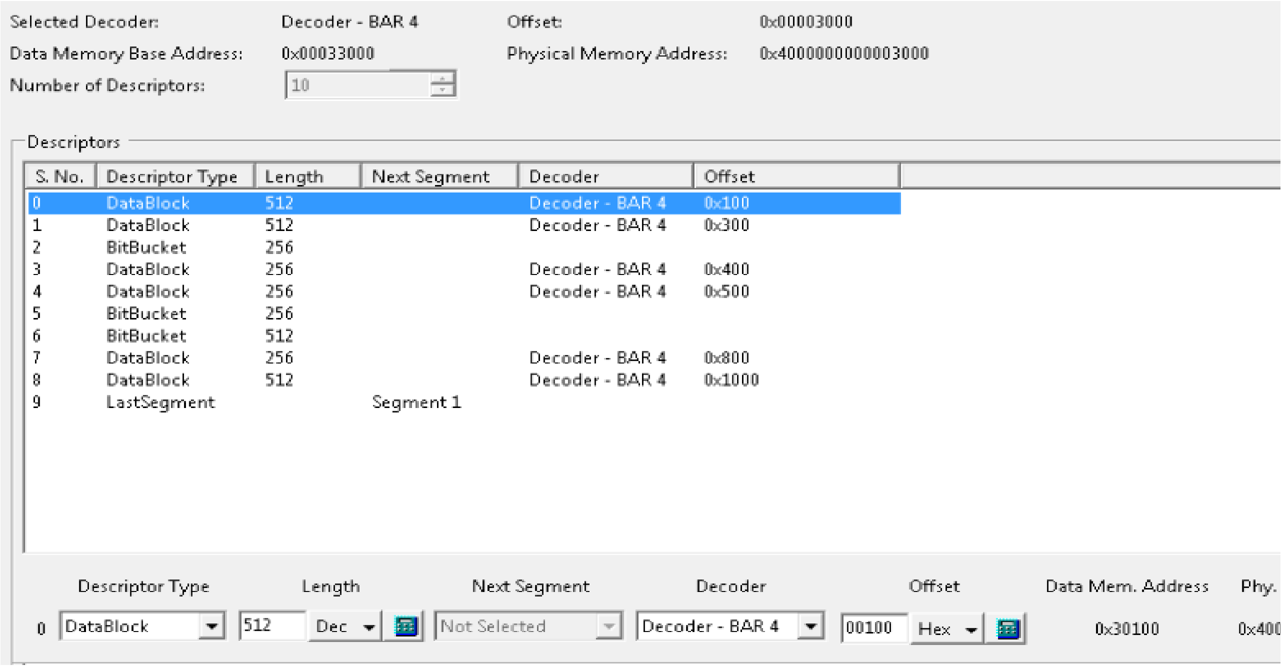
Physical Memory Ad

S. No.	Descriptor Type	Length	Next Segment	Decoder
0	DataBlock	512		Decoder - BAR 4
1	DataBlock	512		Decoder - BAR 4
2	BitBucket	256		
3	DataBlock	256		Decoder - BAR 4
4	DataBlock	256		Decoder - BAR 4
5	BitBucket	256		
6	BitBucket	512		
7	DataBlock	256		Decoder - BAR 4
8	DataBlock	512		Decoder - BAR 4
9	LastSegment		Segment 1	

9 From the right pane, click each descriptor list and select the following for each entry:

- **Descriptor Type** - There are 4 types of descriptors available in each list of SGL segments. The Data Block descriptor is used to transfer specific memory of the data block. The Bit Bucket descriptor will not transfer specific number of bites of the source data of the logical memory. The Segment descriptor is used to jump one segment to another SGL segment. The Last Segment points to the last memory location of the SGL.
- **Length** - You can specify the length in bytes
- **Next Segment** - You can select segment types based on descriptor type.
- **Decoder** - You can select which decoder to be used for the SGL segment list.
- **Offset - Select an offset within the selected decoder that you want to use with the descriptor.**

On the basis of the above selections, the base address in the data memory of Exerciser and the physical memory base address applicable for the SGL entry are displayed.



Selected Decoder: Decoder - BAR 4 Offset: 0x00003000

Data Memory Base Address: 0x00033000 Physical Memory Address: 0x4000000000003000

Number of Descriptors: 10

Descriptors

S. No.	Descriptor Type	Length	Next Segment	Decoder	Offset
0	DataBlock	512		Decoder - BAR 4	0x100
1	DataBlock	512		Decoder - BAR 4	0x300
2	BitBucket	256			
3	DataBlock	256		Decoder - BAR 4	0x400
4	DataBlock	256		Decoder - BAR 4	0x500
5	BitBucket	256			
6	BitBucket	512			
7	DataBlock	256		Decoder - BAR 4	0x800
8	DataBlock	512		Decoder - BAR 4	0x1000
9	LastSegment		Segment 1		

	Descriptor Type	Length		Next Segment	Decoder	Offset		Data Mem. Address	Phy.
0	DataBlock	512	Dec	Not Selected	Decoder - BAR 4	00100	Hex	0x30100	0x400

10 Click **Apply** when all SGL entries of the list are mapped to a decoder and length.

To use a SGL list in an NVMe command

Once a SGL list is created, you can use it in a command sent from Exerciser to an NVMe DUT.

1 Click the **NVM Express** icon from the left navigation pane of the GUI.

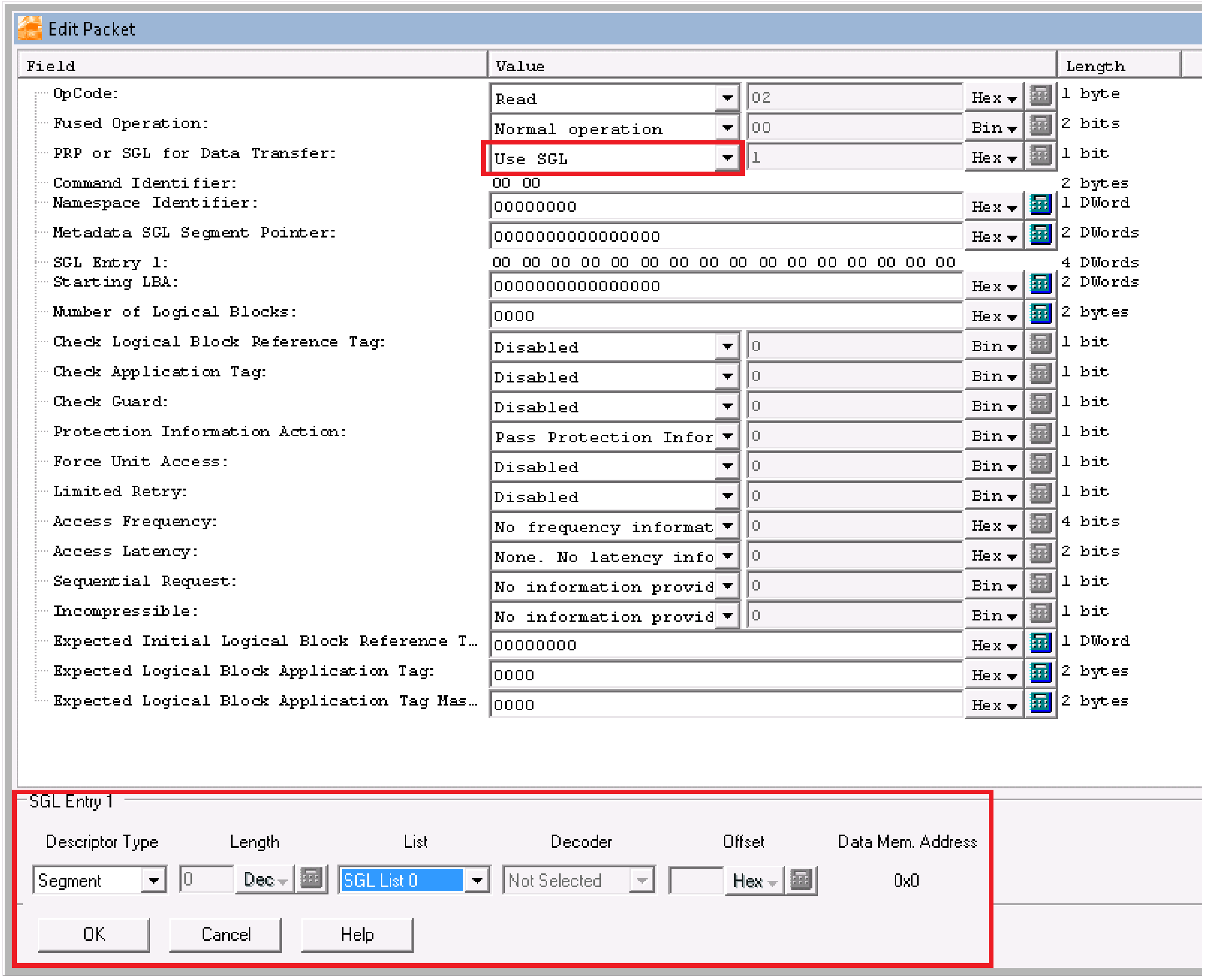
2 Click the **Submission and Completion Queues** tab.

3 From the **Submission Queues** tab, drag and drop an appropriate command template from the Templates section to the left.

4 Double-click the NVMe command that you created in the previous step.

5 In the **Edit Packet** dialog box, select **Use SGL** in the **PRP or SGL for Data Transfer**. Set the SGL parameters in the SGL Entry 1 section as follows.

- To use an SGL Data Block, set **Descriptor Type** to **Data Block** and select the desired values in **Length**, **Decoder** and **Offset**.
- To use an SGL list, set **Descriptor Type** to **Segment** for SGL lists with more than one segment and **Last Segment** for SGL lists with only one segment. Then select the SGL list from the **List** dropdown.

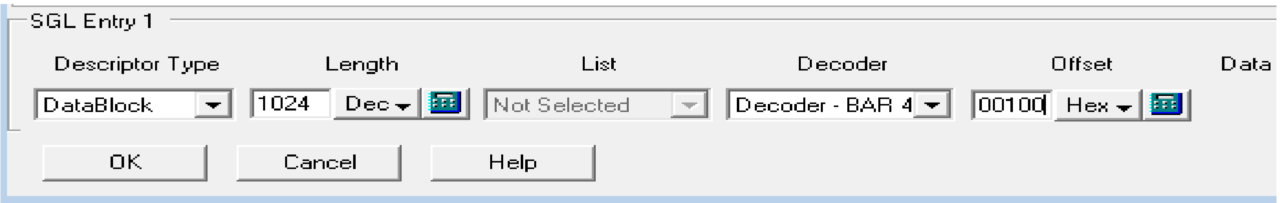


Edit Packet

Field	Value			Length
OpCode:	Read	02	Hex	1 byte
Fused Operation:	Normal operation	00	Bin	2 bits
PRP or SGL for Data Transfer:	Use SGL	1	Hex	1 bit
Command Identifier:	00 00			2 bytes
Namespace Identifier:	00000000		Hex	1 DWord
Metadata SGL Segment Pointer:	0000000000000000		Hex	2 DWords
SGL Entry 1:	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00			4 DWords
Starting LBA:	0000000000000000		Hex	2 DWords
Number of Logical Blocks:	0000		Hex	2 bytes
Check Logical Block Reference Tag:	Disabled	0	Bin	1 bit
Check Application Tag:	Disabled	0	Bin	1 bit
Check Guard:	Disabled	0	Bin	1 bit
Protection Information Action:	Pass Protection Infor	0	Bin	1 bit
Force Unit Access:	Disabled	0	Bin	1 bit
Limited Retry:	Disabled	0	Bin	1 bit
Access Frequency:	No frequency informat	0	Hex	4 bits
Access Latency:	None. No latency info	0	Hex	2 bits
Sequential Request:	No information provid	0	Bin	1 bit
Incompressible:	No information provid	0	Bin	1 bit
Expected Initial Logical Block Reference T…	00000000		Hex	1 DWord
Expected Logical Block Application Tag:	0000		Hex	2 bytes
Expected Logical Block Application Tag Mas…	0000		Hex	2 bytes

SGL Entry 1

Descriptor Type	Length	List	Decoder	Offset	Data Mem. Address
Segment	0 Dec	SGL List 0	Not Selected	Hex	0x0



SGL Entry 1

Descriptor Type	Length	List	Decoder	Offset	Data
DataBlock	1024 Dec	Not Selected	Decoder - BAR 4	00100 Hex	

6 Click **OK**.

To delete a SGL list

1 Select the SGL list from the **SGL Lists** section.

2 Click the ✕ button.

The SGL List and its SGL segment entries are deleted.

To delete a SGL Segment

1 Select the SGL list from the **SGL Lists** section.

2 Click the ❌ button.

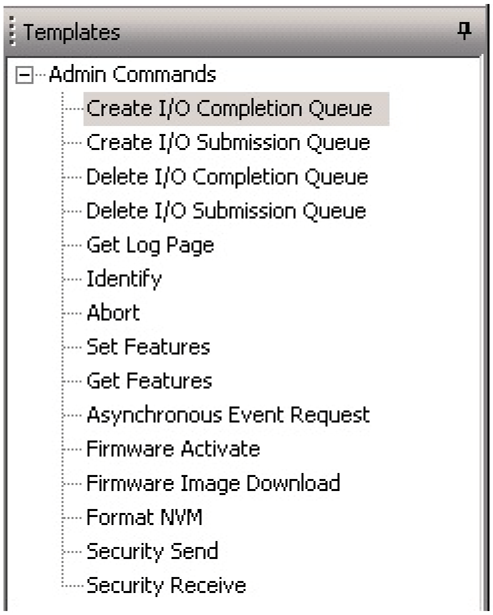
The SGL segment entries are deleted.

Submitting an NVMe Command and Viewing its Completion Status

Supported Commands

You can submit the following NVMe commands from Exerciser emulating the role of an NVMe root complex.

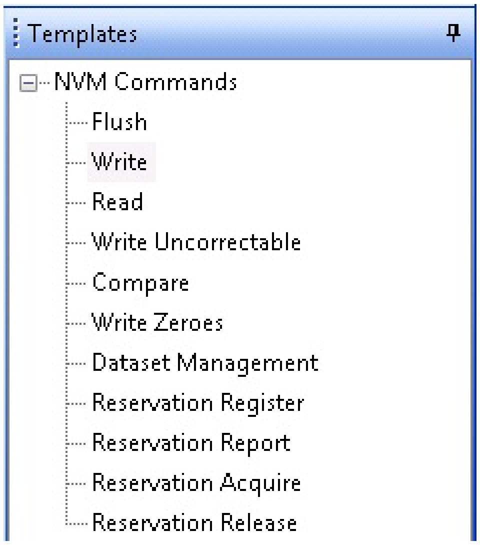
NVMe Admin Commands



Templates

- Admin Commands
  - Create I/O Completion Queue
  - Create I/O Submission Queue
  - Delete I/O Completion Queue
  - Delete I/O Submission Queue
  - Get Log Page
  - Identify
  - Abort
  - Set Features
  - Get Features
  - Asynchronous Event Request
  - Firmware Activate
  - Firmware Image Download
  - Format NVM
  - Security Send
  - Security Receive

NVMe I/O Commands



Templates

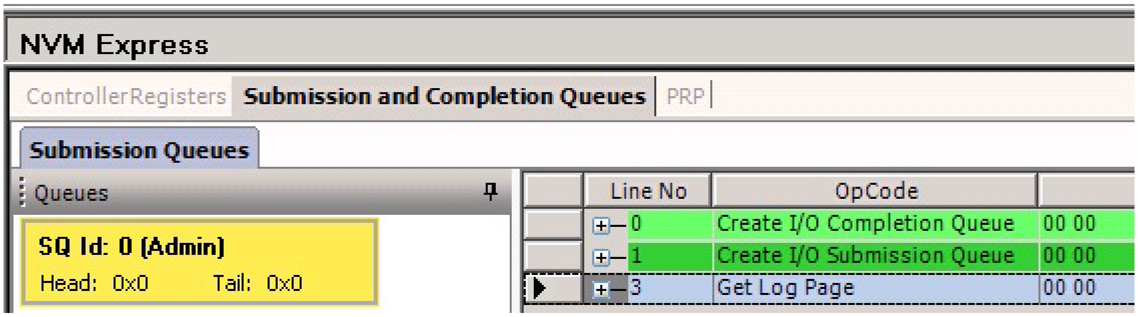
- NVM Commands
  - Flush
  - Write
  - Read
  - Write Uncorrectable
  - Compare
  - Write Zeroes
  - Dataset Management
  - Reservation Register
  - Reservation Report
  - Reservation Acquire
  - Reservation Release

To submit a command

1 Click the **NVM Express** icon from the left navigation pane of the GUI.
2 Click the **Submission and Completion Queues** tab.
3 To submit an Admin command, click the **SQ Id : 0 (Admin)** option displayed under **Queues** section on the left. All the Admin commands are available for submission to this queue.

   To submit an NVMe I/O command, click an I/O submission queue displayed under **Queues** section on the left. All the I/O commands are available for submission to an I/O submission queue.
4 Drag the command template from the **Templates** pane on the right and drop it to the pane displayed in the center pane.

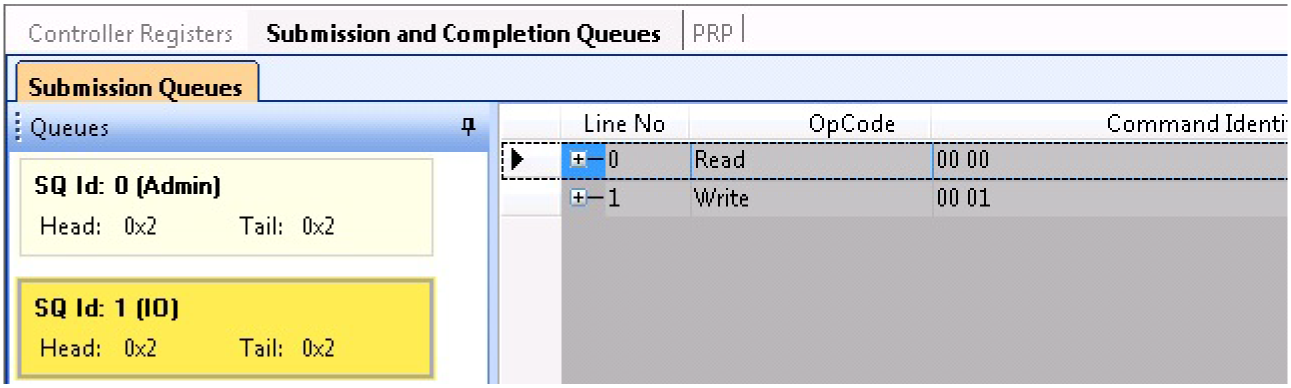
   An instance of the template is added to the list of commands to be submitted.



SQ Id: 0 (Admin)
Head: 0x0 Tail: 0x0

Line No	OpCode	
0	Create I/O Completion Queue	00 00
1	Create I/O Submission Queue	00 00
3	Get Log Page	00 00

Admin commands added to the Admin submission queue



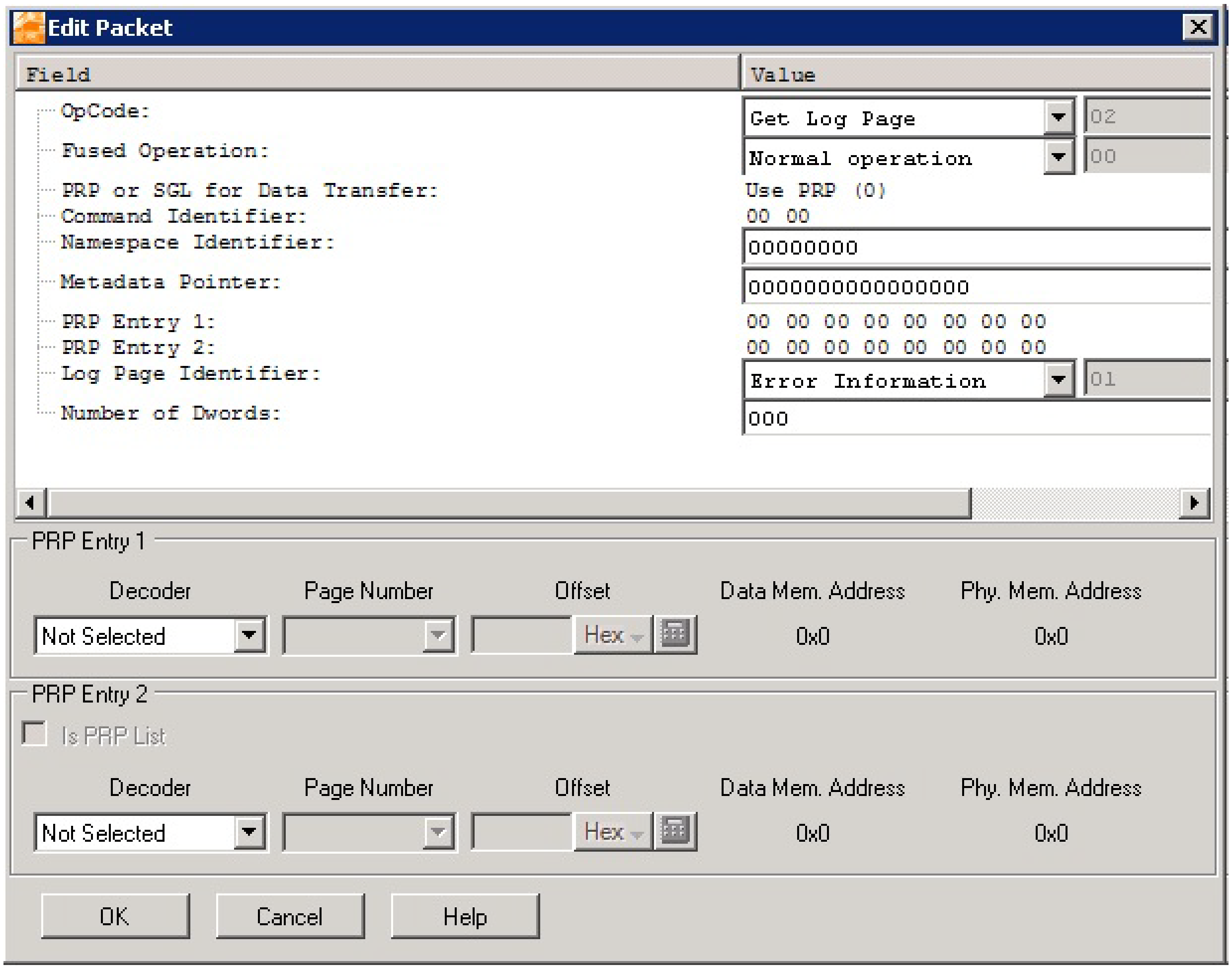
SQ Id: 0 (Admin)
Head: 0x2 Tail: 0x2

SQ Id: 1 (IO)
Head: 0x2 Tail: 0x2

Line No	OpCode	Command Identi
0	Read	00 00
1	Write	00 01

I/O commands added to an I/O submission queue

5 Click the + icon displayed with the newly created command instance to view the command details.
6 Double-click the command instance to edit the command details, as needed in the **Edit Packet** dialog box. Click **OK** to confirm the command settings.



Edit Packet

Field	Value	
OpCode:	Get Log Page	02
Fused Operation:	Normal operation	00
PRP or SGL for Data Transfer:	Use PRP (0)	
Command Identifier:	00 00	
Namespace Identifier:	00000000	
Metadata Pointer:	0000000000000000	
PRP Entry 1:	00 00 00 00 00 00 00 00	
PRP Entry 2:	00 00 00 00 00 00 00 00	
Log Page Identifier:	Error Information	01
Number of Dwords:	000	

PRP Entry 1

Decoder	Page Number	Offset	Data Mem. Address	Phy. Mem. Address
Not Selected		Hex	0x0	0x0

PRP Entry 2

□ Is PRP List

Decoder	Page Number	Offset	Data Mem. Address	Phy. Mem. Address
Not Selected		Hex	0x0	0x0

7 The command is now ready for submission. Click **Apply** to confirm and submit the command for processing to DUT.

To view the completion status for a command

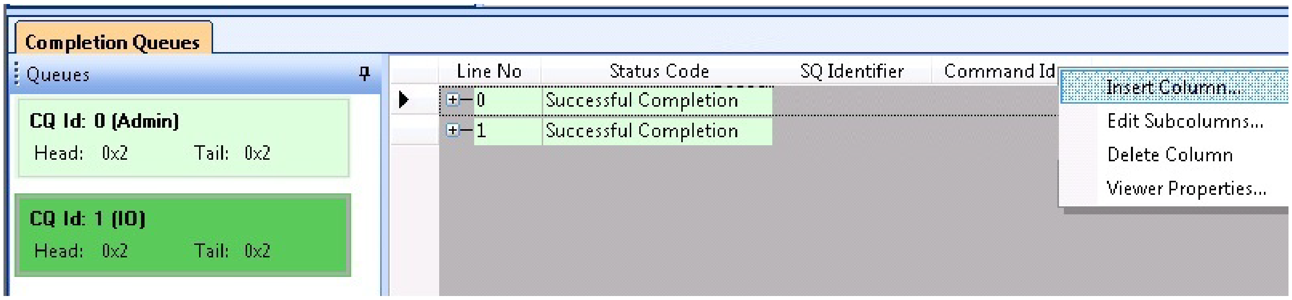
Completion status for a command is posted in the mapped completion queue. These completion status are displayed in the **Completion Queues** section of the **Submission and Completion Queues** tab.

To quickly identify the completion status for a particular command from the list of completion status posted in a queue, you can add the **SQ Identifier** and **Command Identifier** fields to the collapsed view of the completion queue:

1 Click the **NVM Express** icon from the left navigation pane of the GUI.

2 Click the **Submission and Completion Queues** tab.

3 Right-click the header of the pane in which completion status are displayed and then select **Insert Column** from the context-menu.



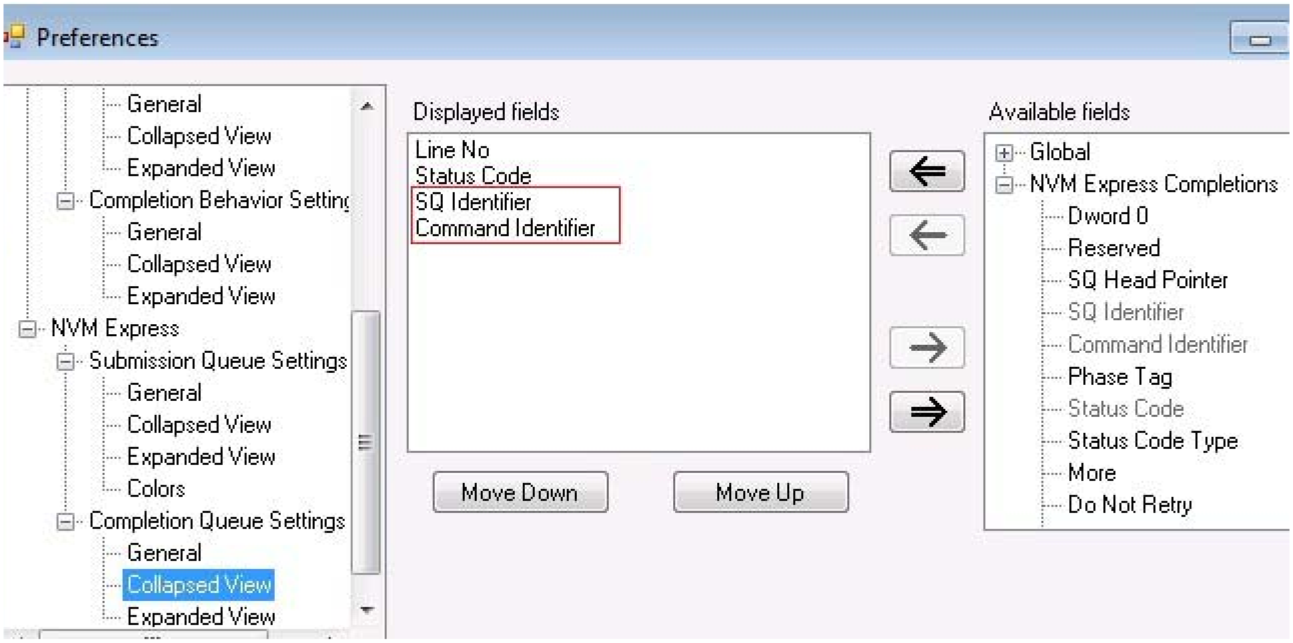
CQ Id: 0 (Admin)
Head: 0x2 Tail: 0x2

CQ Id: 1 (IO)
Head: 0x2 Tail: 0x2

Line No	Status Code	SQ Identifier	Command Id
0	Successful Completion		
1	Successful Completion		

4 In the **Preferences** dialog box, expand the **NVM Express** > **Completion Queue Settings** > **Collapsed View** from the left pane.

5 Add the **SQ Identifier** and **Command Identifier** fields to the **Collapsed** view of the completion queue by dragging and dropping these fields from the **Available fields** section to the **Displayed fields** section.

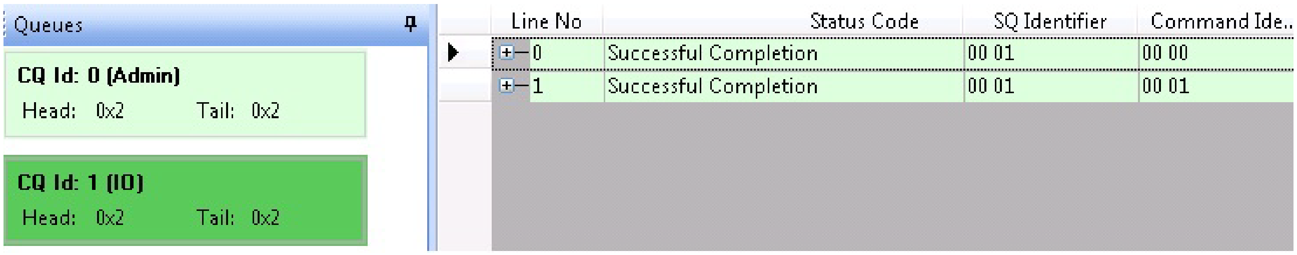


Displayed fields
Line No
Status Code
SQ Identifier
Command Identifier

Available fields
Global
NVM Express Completions
Dword 0
Reserved
SQ Head Pointer
SQ Identifier
Command Identifier
Phase Tag
Status Code
Status Code Type
More
Do Not Retry

6 Click **OK**.

The completion status posted in a completion queue will now display the identifier for the relevant submission queue and command in the collapsed view. This can help you quickly identify the completion status posted for a command.



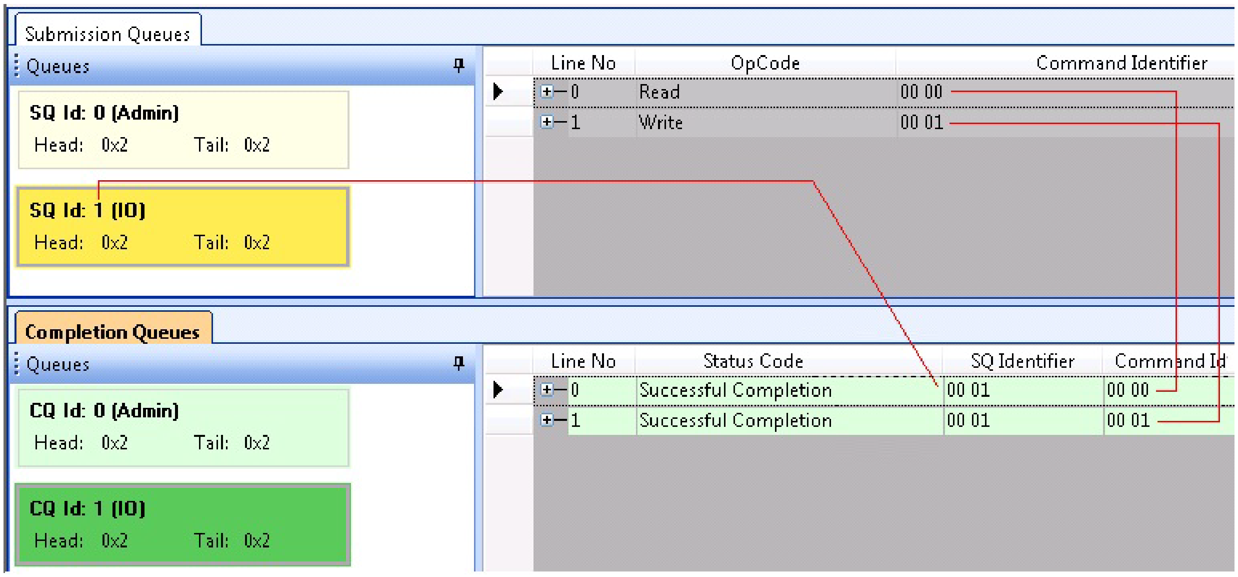
CQ Id: 0 (Admin)
Head: 0x2 Tail: 0x2

CQ Id: 1 (IO)
Head: 0x2 Tail: 0x2

Line No	Status Code	SQ Identifier	Command Ide...
0	Successful Completion	00 01	00 00
1	Successful Completion	00 01	00 01

To view the completion status for a command

1. Click the **NVM Express** icon from the left navigation pane of the GUI.
2. Click the **Submission and Completion Queues** tab.
3. Click the submission queue to which you posted the command. Clicking the submission queue selects its mapped completion queue in the bottom pane of the tab.
4. From the completion status posted in the highlighted completion queue, use the **SQ Identifier** and **Command Identifier** fields to identify the completion status of the command.



SQ Id: 0 (Admin)
Head: 0x2 Tail: 0x2

SQ Id: 1 (IO)
Head: 0x2 Tail: 0x2

Line No	OpCode	Command Identifier
0	Read	00 00
1	Write	00 01

CQ Id: 0 (Admin)
Head: 0x2 Tail: 0x2

CQ Id: 1 (IO)
Head: 0x2 Tail: 0x2

Line No	Status Code	SQ Identifier	Command Id
0	Successful Completion	00 01	00 00
1	Successful Completion	00 01	00 01

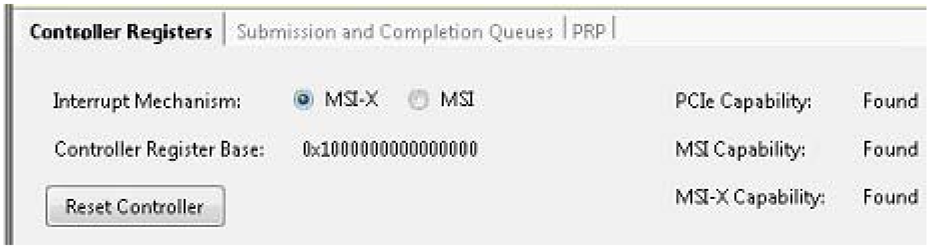
Resetting the NVMe Controller Capabilities

You can reset the controller to reset the queues and registers to their default values and state. On resetting the controller:

- all I/O submission and completion queues are deleted.
- the Admin submission and completion queues are reset.
- all controller registers except the Admin queue registers are reset to their default values.

There are two ways in which you can reset the NVMe controller from the **Controller Registers** tab of the **NVM Express** page.

- Either use the **Reset Controller** button. On doing so, the **Enable** bit of the **Controller Configuration** register transitions from 1 to 0 and then restores back to 1.



Interrupt Mechanism: ● MSI-X ○ MSI

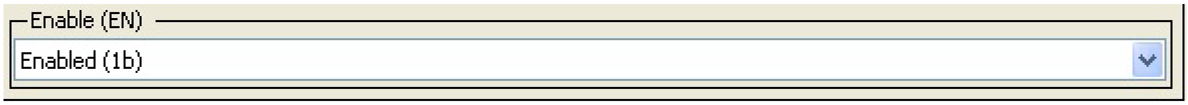
Controller Register Base: 0x1000000000000000

PCIe Capability: Found

MSI Capability: Found

MSI-X Capability: Found

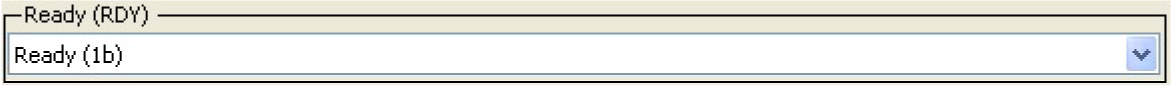
- Or set the **Enable** field in the **CC - Controller Configuration** register to **Disabled (0b)**, and then set it back to **Enabled (1b)**. Make sure that you write this transition from 1 to 0 and then back to 1 to DUT registers by clicking **Apply** or clicking the button displayed with the **CC - Controller Configuration** register.



Enable (EN)

Enabled (1b)

After resetting the controller, verify if the controller is ready to process commands by checking the **Ready** bit in the **CSTS - Controller Status** register on the **Controller Registers** tab.



Ready (RDY)

Ready (1b)

Decoder Mapping Sheet for NVMe Emulation

You can use the following sheet to record the mapping that you define between physical memory addresses and internal memory addresses of Exerciser using Decoders. This sheet can help you organize the 256 KB of internal memory of Exerciser effectively and avoid overlaps and overwrites to the memory areas.

NVMe Component for which mapping is defined (Examples, Queue, PRP list, SGL list, data transfer)	Decoder BAR used	Page offset within selected decoder (Page number)	Offset within selected page	Physical memory address	Internal data memory Address	Size allocated
Admin submission queue						
Admin completion queue						
I/O submission queue 1						
I/O submission queue 2						
I/O submission queue 3						
I/O submission queue 4						
I/O completion queue 1						
I/O completion queue 2						
I/O completion queue 3						
I/O completion queue 4						
PRP List 0						
PRP List 1						
PRP List 2						
PRP List 3						
SGL List 0						
SGL List 1						
SGL List 2						
SGL List 3						
Data Buffer 0						
Data Buffer 1						
Data Buffer 2						

12 Emulating an NVMe EndPoint

This chapter provides information on how you can configure and use U4305A Protocol Exerciser as an NVMe EndPoint.

Exerciser’s Capabilities and Role as an NVMe EndPoint

This topic briefly describes the following capabilities of Exerciser when emulating the role of an NVMe EndPoint:

- Exerciser in its NVMe endpoint emulation acts as a mass storage device. After you perform its initial setup and configuration as an NVMe endpoint, it displays itself as a two-drive DUT on an SUT.
- It supports two namespaces. You can define the size of each of these namespaces. The maximum size allowed for each namespace is 1 GB.
- Exerciser allows the RC device to configure its controller registers as an NVMe endpoint and to start NVMe traffic on the Exerciser.
- As an NVMe endpoint, Exerciser can execute mandatory Admin commands for creating and deleting submission and completion queues, Get Features, Set Features and Identify command. It also supports the following three mandatory NVMe commands:
  - Flush
  - Read
  - Write
- Exerciser supports MSI and MSI-X interrupt mechanisms so that the RC device can use these mechanisms. Exerciser can respond to interrupts sent via MSI or MSI-X and do the needful as expected from an NVMe device, that is fetch commands from the submission queues, execute them and write back completions in the completion queues.
- Exerciser allows you to set up the values in the controller registers and also to set values for data structures such as, identify structures and log pages. It also lets you set the values for device features to be used with the Get / Set features command.

Preparing Exerciser and SUT for the NVMe EndPoint Mode

Prerequisites and List of Steps Involved in Preparation

Prerequisites

The following are the prerequisites for the Exerciser to emulate an NVMe EndPoint:

- You must have purchased and installed the NVMe license to get the NVMe functionality.
- You should have plugged Exerciser into the PCIe slot of SUT.

You must perform the below-mentioned steps in the specified sequence to set up Exerciser to function in the NVMe EndPoint mode.

Step 1 - Install the NVMe Endpoint Driver on SUT

When you install the Exerciser software, the NVMe Driver is available at the following location on the computer hosting the Exerciser software.

‘C:\Program Files (x86)\Keysight\SPT\PCIEExerciserGen3\8.74 Release\Drivers’ in Windows 7 64 bit

‘C:\ProgramFiles\Keysight\SPT\PCIEExerciserGen3\8.74 Release\Drivers’ in Windows XP/7 32 bit.

To install this Driver:

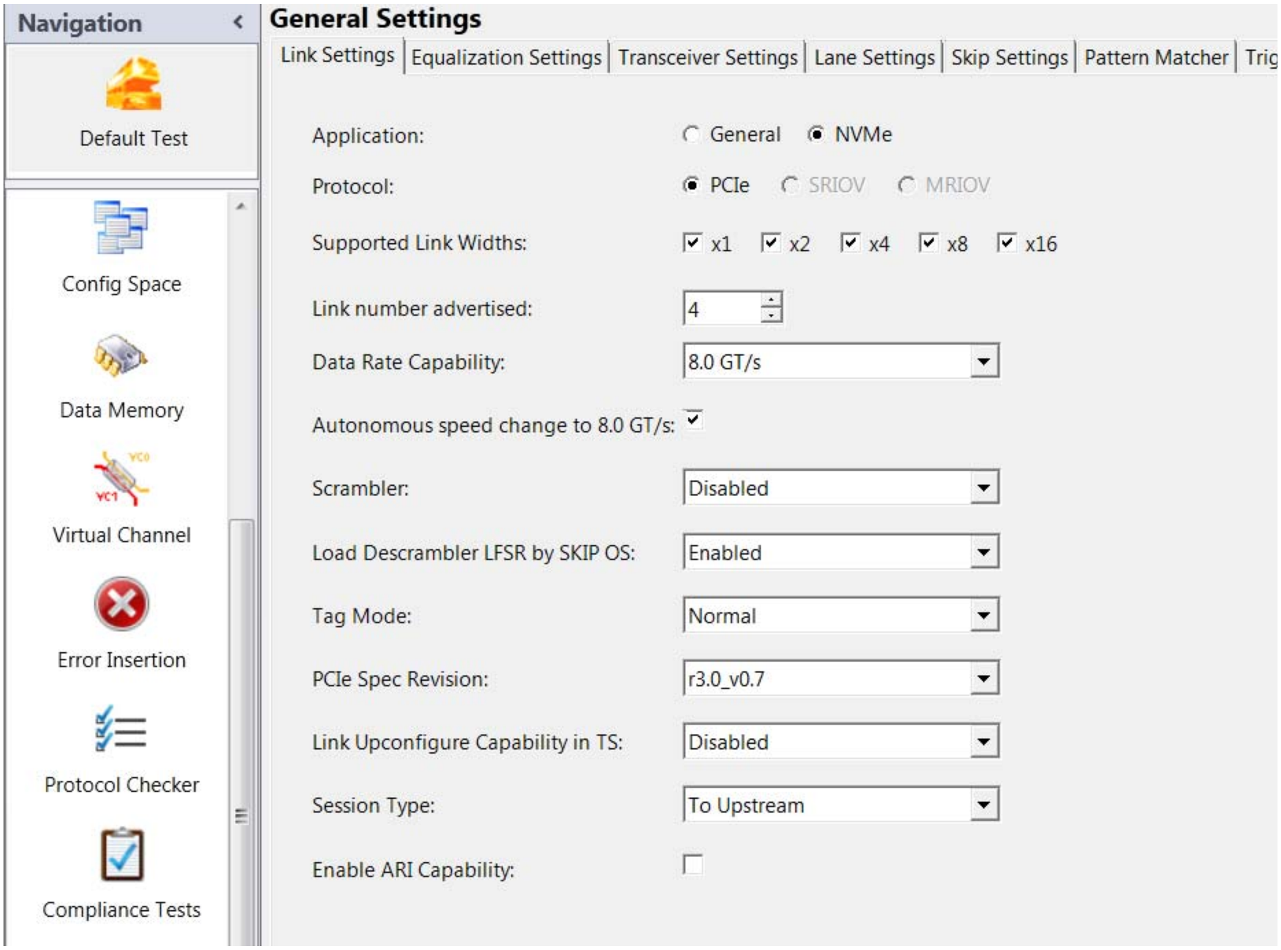
1. Copy the appropriate driver (depending on the operating system you use) from the above-mentioned location, to any folder on the SUT.
2. Start ‘dpinst.exe’ from the copied folder on SUT.
3. Follow the on screen instructions to install the driver.

This Driver is based on Open Source NVM Express driver from http://www.openfabrics.com. For further information please refer to: https://www.openfabrics.org/index.php/developer-tools/nvme-windows-development.html.

Step 2 - Configuring the PCIe link for NVMe EndPoint

By default, Exerciser is not set to work in the NVMe mode. If you want Exerciser to emulate an NVMe EndPoint, then the first step is to set the link settings of Exerciser to the NVMe mode.

1. Click the **General Settings** icon from the left navigation pane in the GUI.
2. Click the **Link Settings** tab.
3. Click **NVMe** in the Application field. On selecting NVMe, the MRIOV and SRIOV protocol support is disabled. Only PCIe is supported and displayed enabled and selected in the Protocol field.
4. In the Session Type, select **To Upstream** for the Exerciser to act as an NVMe EndPoint.



General Settings

Application: ○ General ● NVMe

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: Disabled

Session Type: To Upstream

Enable ARI Capability: □

5 Click **Apply.**

6 Click the **Start Link Training** button in the menu bar.

The PCIe link is now configured in the NVMe Endpoint mode.

Step 3 - Configure the NVMe Endpoint Related Settings and Registers (if needed)

You can either continue with the default NVMe Endpoint settings and controller register values set for Exerciser or configure these settings to suit your requirements in the **NVM Express Endpoint** page. Refer to the “Configuring the NVM Express EndPoint Settings and Controller Registers" on page 219 to know about customizing/changing NVMe Endpoint settings as required.

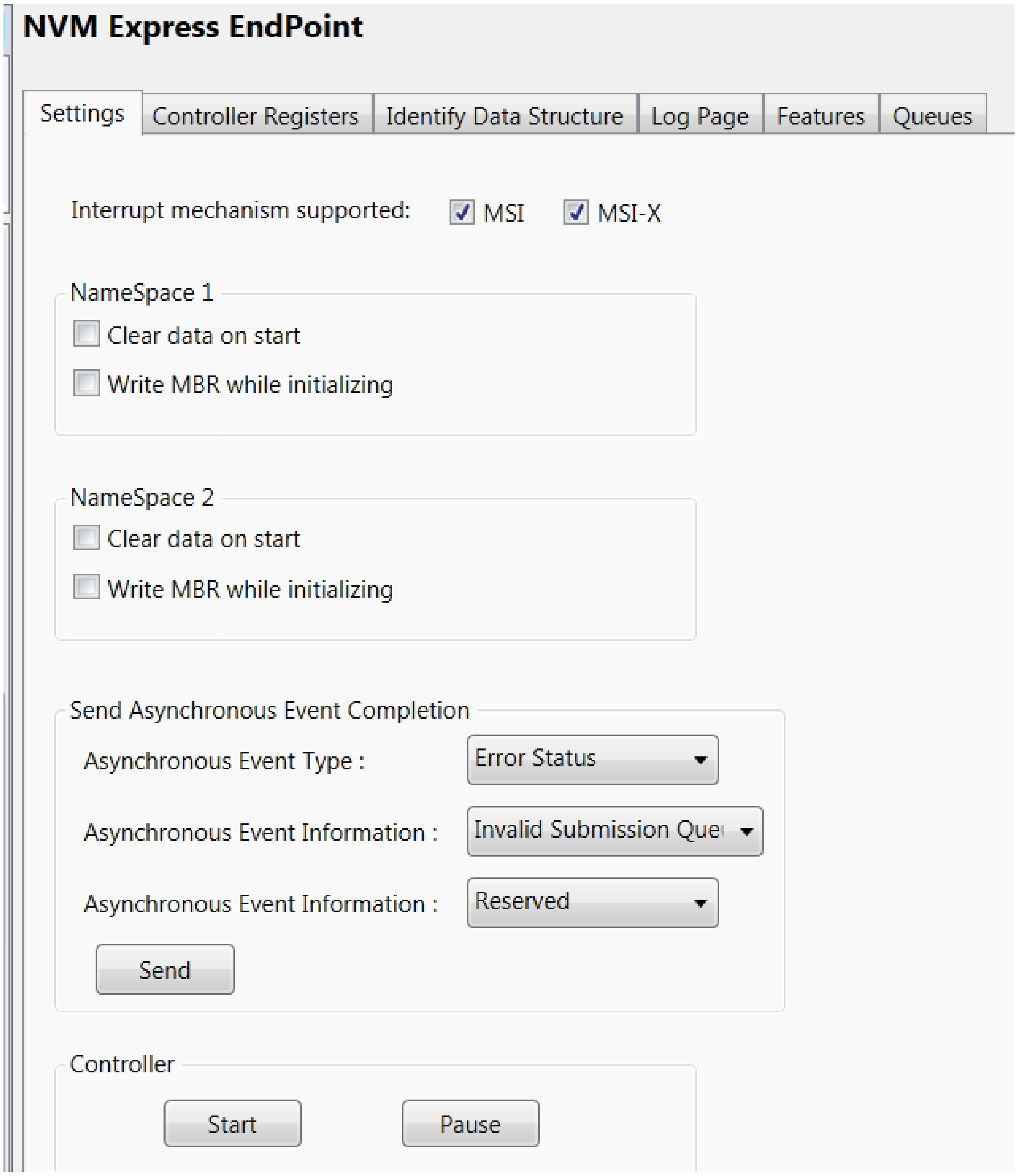
NOTE

The **NVM Express Endpoint** page is displayed only if you have selected **NVMe** in the Application field and **To Upstream** in the Session type field of the General Settings page.

Step 4 - Start the Exerciser’s NVMe Controller

1 Click **NVM Express** in the Navigation pane.

The **NVM Express EndPoint** screen displays as follows.



NVM Express EndPoint

Interrupt mechanism supported: ☑ MSI ☑ MSI-X

NameSpace 1

□ Clear data on start

□ Write MBR while initializing

NameSpace 2

□ Clear data on start

□ Write MBR while initializing

Send Asynchronous Event Completion

Asynchronous Event Type : Error Status

Asynchronous Event Information : Invalid Submission Que

Asynchronous Event Information : Reserved

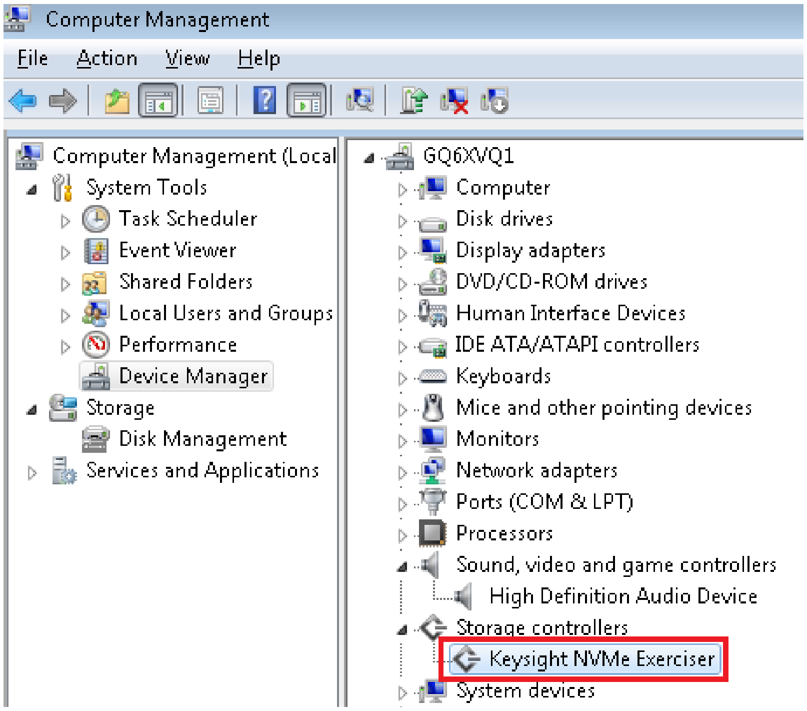
Controller

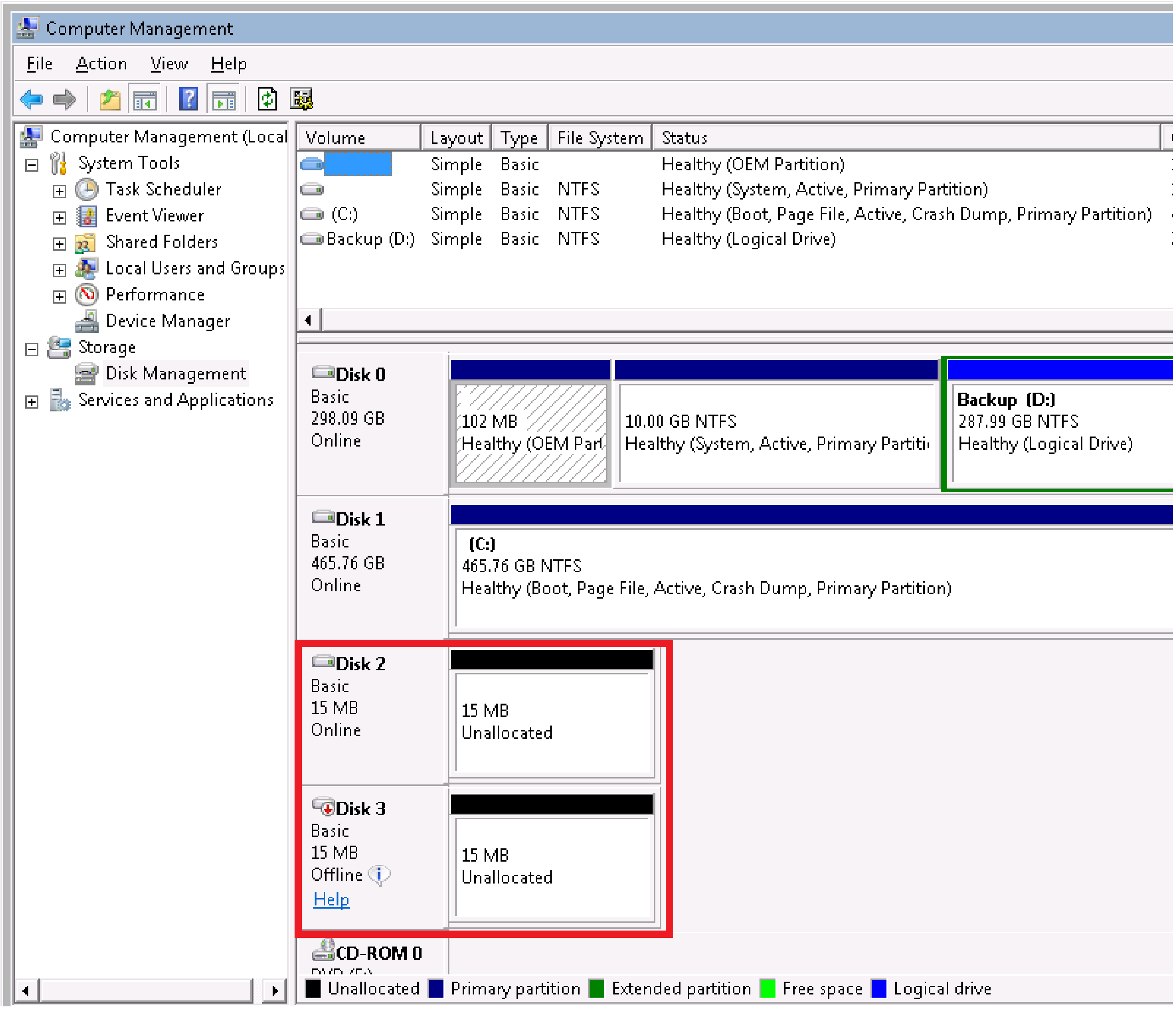
2 In the **Settings** tab, click the **Start** button in the **Controller** section to start the controller.

Step 5 - Restart the System Under Test (SUT)

As a final step in preparation, restart the SUT to which you plugged the Exerciser

Exerciser then shows up in the Device Manger of SUT as ‘Keysight NVMe Exerciser’ and the two disks of Exerciser appear in the disk management console of SUT.


Computer Management
File
Action
View
Help
Computer Management (Local
System Tools
Task Scheduler
Event Viewer
Shared Folders
Local Users and Groups
Performance
Device Manager
Storage
Disk Management
Services and Applications
GQ6XVQ1
Computer
Disk drives
Display adapters
DVD/CD-ROM drives
Human Interface Devices
IDE ATA/ATAPI controllers
Keyboards
Mice and other pointing devices
Monitors
Network adapters
Ports (COM & LPT)
Processors
Sound, video and game controllers
High Definition Audio Device
Storage controllers
Keysight NVMe Exerciser
System devices



Volume	Layout	Type	File System	Status
	Simple	Basic		Healthy (OEM Partition)
	Simple	Basic	NTFS	Healthy (System, Active, Primary Partition)
(C:)	Simple	Basic	NTFS	Healthy (Boot, Page File, Active, Crash Dump, Primary Partition)
Backup (D:)	Simple	Basic	NTFS	Healthy (Logical Drive)

Disk 0
Basic
298.09 GB
Online

102 MB
Healthy (OEM Part

10.00 GB NTFS
Healthy (System, Active, Primary Partiti

Backup (D:)
287.99 GB NTFS
Healthy (Logical Drive)

Disk 1
Basic
465.76 GB
Online

(C:)
465.76 GB NTFS
Healthy (Boot, Page File, Active, Crash Dump, Primary Partition)

Disk 2
Basic
15 MB
Online

15 MB
Unallocated

Disk 3
Basic
15 MB
Offline
Help

15 MB
Unallocated

It is recommended that you disable write caching for the NVMe drives emulated by the Exerciser. IF write catching is enabled, the progress information of various file operations may not show correctly.

To disable write caching for the NVMe drives emulated by the Exerciser:

1 Open **Computer Management** Console by right clicking the **My Computer** icon and selecting **Manage** from the context menu.

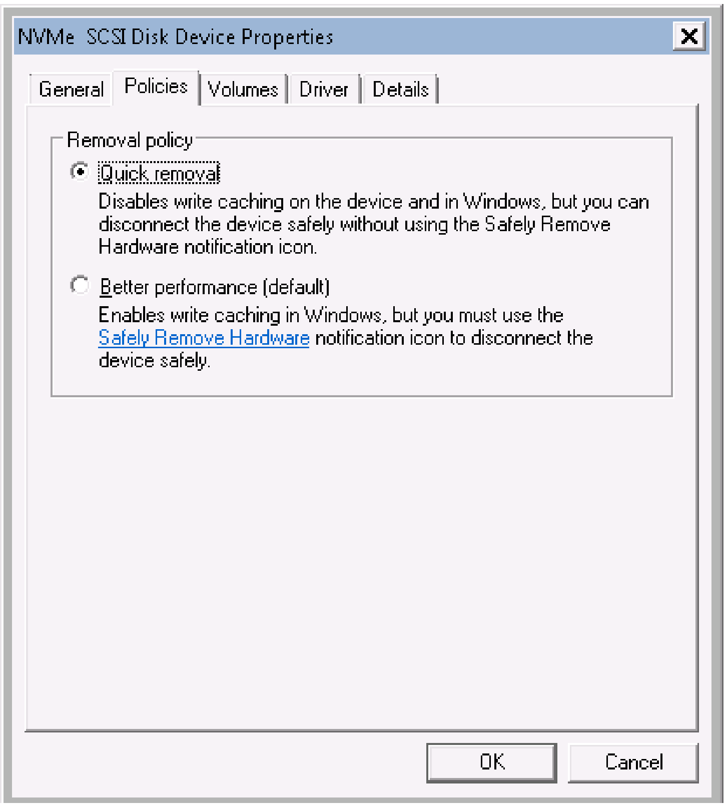
2 Go to **Device Manager**.

3 Expand the node **Disk drives**. You will see 2 disc drives here named ‘NVMe SCSI Disk Device’. Select each of these and perform the below mentioned steps.

   a Right click the NVMe SCSI Disk Device node and select **Properties**.

   b Click the **Policies** tab in the Properties dialog box.

   c Select **Quick Removal** in Removal Policy and click **OK**.



Removal policy

● Quick removal

Disables write caching on the device and in Windows, but you can disconnect the device safely without using the Safely Remove Hardware notification icon.

○ Better performance (default)

Enables write caching in Windows, but you must use the Safely Remove Hardware notification icon to disconnect the device safely.

Configuring the NVM Express EndPoint Settings and Controller Registers

NOTE

While changing controller register settings, remember that some settings may require you to reboot the SUT for these changes to take effect.

Also, for certain register settings, such as namespace and memory size, Exerciser can accept the value only as per its supported range. If you set a value outside this range, Exerciser will not implement such a value. The following list displays the supported maximum value for the specific registry settings.

- Maximum Namespace Size supported: 1 GB

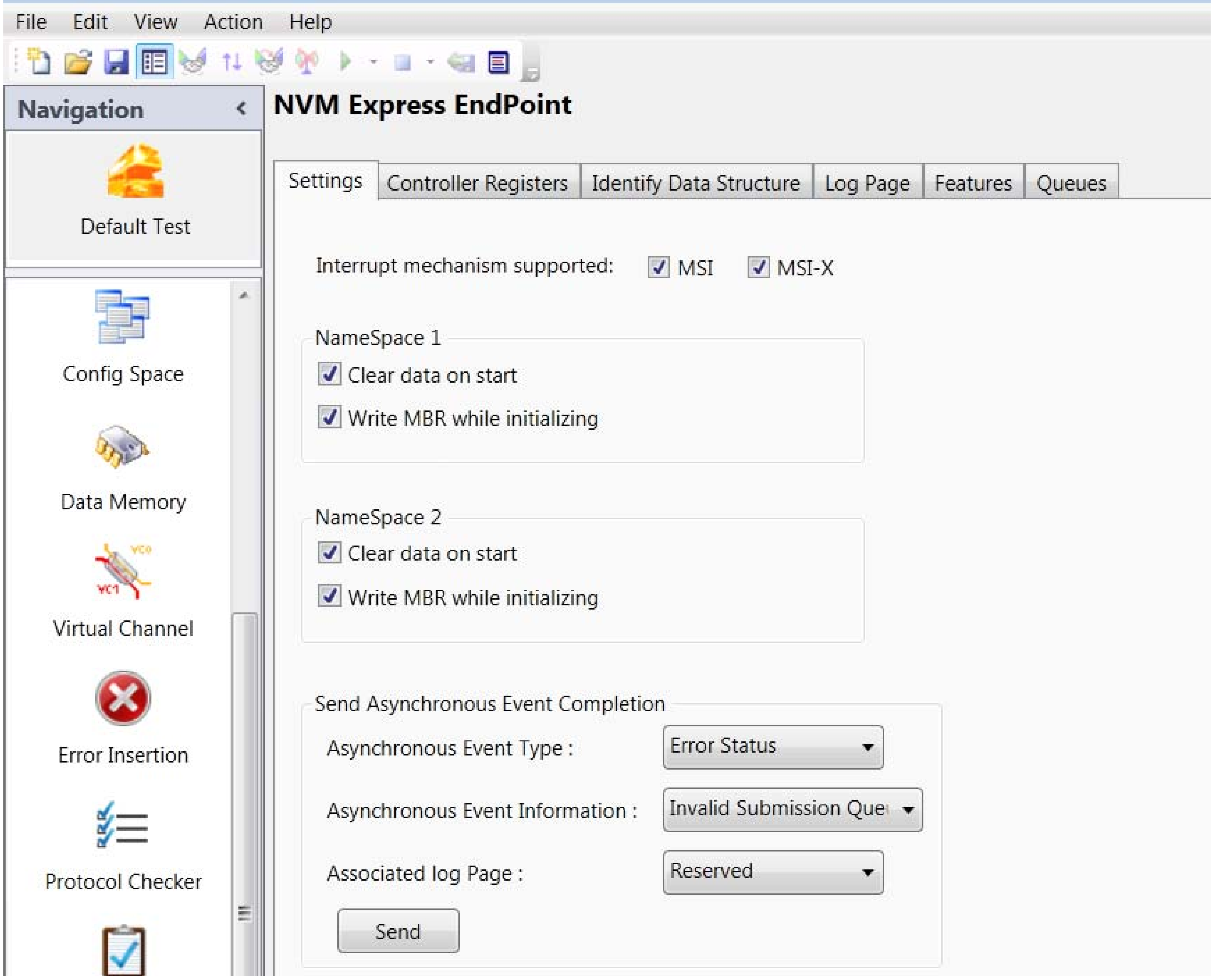
- Maximum Memory Page Size supported: 128 KB

- Maximum Data Transfer Size supported: 1 MB

To configure NVMe Endpoint Settings:

You use the Settings tab in the NVM Express Endpoint page to configure these settings.

- Select MSI or MSI-X field based on the interrupt mechanism capabilities that you want to set.
- Select the **Namespace** attributes you want to set for each of the two namespaces supported by Exerciser.
  - Select the **Clear data on start** checkbox to clear the namespace data on start. This will erase all data written to the namespace and drive will show as a RAW unformatted drive in the SUT.
  - Select the **Write MBR while initializing** checkbox to write the Master Boot Record (MBR) to the namespace during start.



NVM Express EndPoint

Interrupt mechanism supported: ☑ MSI ☑ MSI-X

NameSpace 1
☑ Clear data on start
☑ Write MBR while initializing

NameSpace 2
☑ Clear data on start
☑ Write MBR while initializing

Send Asynchronous Event Completion
Asynchronous Event Type : Error Status
Asynchronous Event Information : Invalid Submission Que
Associated log Page : Reserved

- Send Asynchronous Event Completion

  You can use this feature to send an Asynchronous Event Completion to the RC device. Select the values of the Asynchronous Event Type, Asynchronous Event Information and Associated Log Page fields and then click the **Send** button.

NOTE

Asynchronous Event Completion will be sent only for Asynchronous packets that are already sent or will be sent.

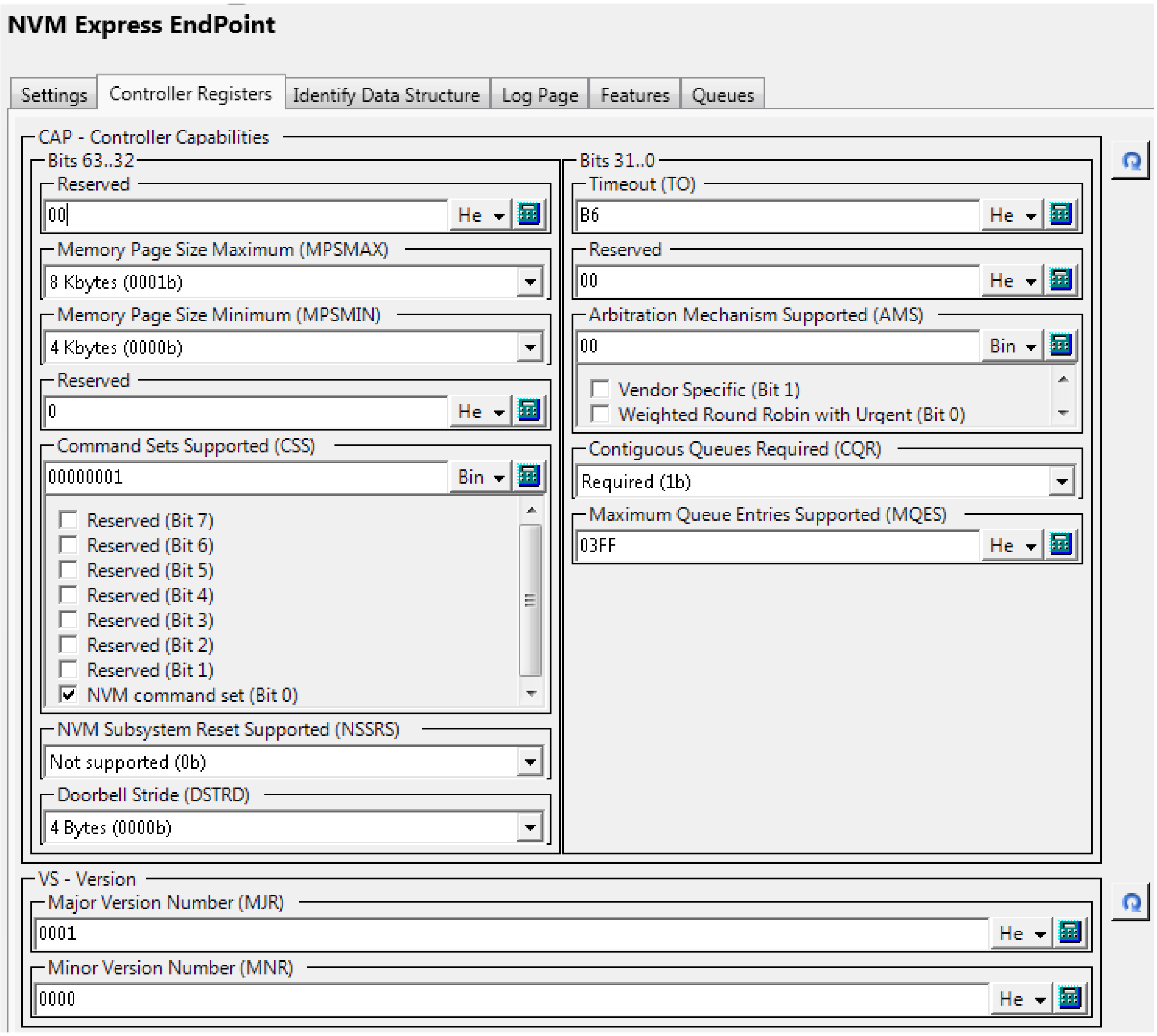
- Click **Apply** to save changes made to the NVMe EndPoint settings.

To get a description of each field in this tab, refer to "NVMe EndPoint -Settings" on page 287.

To Update the Controller Registers

You use the **Controller Registers** tab in the NVM Express Endpoint page to update Exerciser's controller registers for NVMe endpoint emulation.

You can view and edit the controller registers of Exerciser using this tab.



CAP - Controller Capabilities

Bits 63..32

- Reserved: 00 (He)
- Memory Page Size Maximum (MPSMAX): 8 Kbytes (0001b)
- Memory Page Size Minimum (MPSMIN): 4 Kbytes (0000b)
- Reserved: 0 (He)
- Command Sets Supported (CSS): 00000001 (Bin)
  - □ Reserved (Bit 7)
  - □ Reserved (Bit 6)
  - □ Reserved (Bit 5)
  - □ Reserved (Bit 4)
  - □ Reserved (Bit 3)
  - □ Reserved (Bit 2)
  - □ Reserved (Bit 1)
  - ☑ NVM command set (Bit 0)
- NVM Subsystem Reset Supported (NSSRS): Not supported (0b)
- Doorbell Stride (DSTRD): 4 Bytes (0000b)

Bits 31..0

- Timeout (TO): B6 (He)
- Reserved: 00 (He)
- Arbitration Mechanism Supported (AMS): 00 (Bin)
  - □ Vendor Specific (Bit 1)
  - □ Weighted Round Robin with Urgent (Bit 0)
- Contiguous Queues Required (CQR): Required (1b)
- Maximum Queue Entries Supported (MQES): 03FF (He)

VS - Version

- Major Version Number (MJR): 0001 (He)
- Minor Version Number (MNR): 0000 (He)

A ⤓ and ↻ button is displayed with each controller register.

The ⤓ button allows you to write the changes that you made to a particular controller register with which the button is displayed.

The ↻ button allows you to read the current values for a register.

Clicking the **Apply** button saves all the changes that you made to various controller registers displayed in the Controller Registers tab.

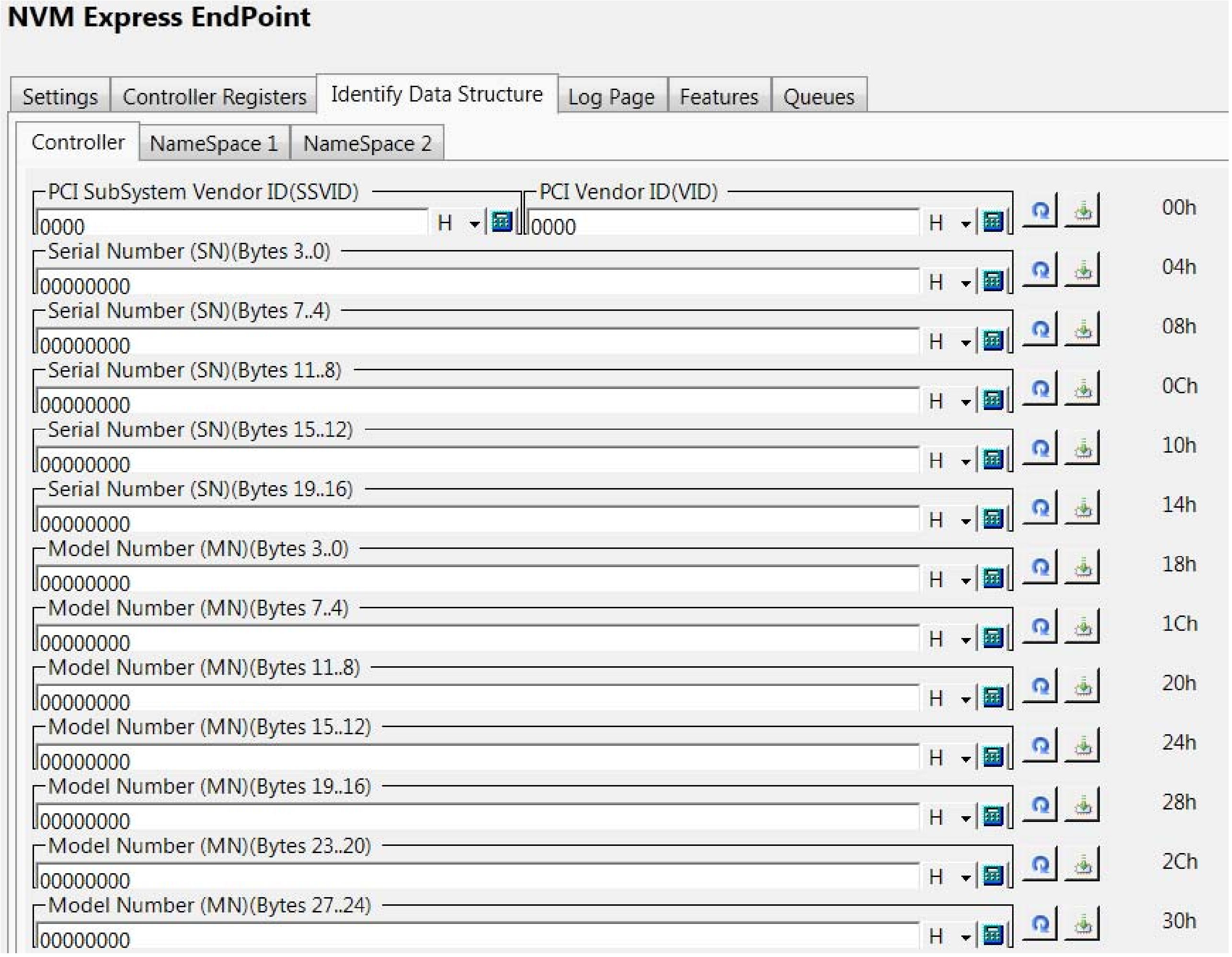
Editing Identify Data Structure

You can edit the Identify Data Structure for the controller as well as the supported NameSpaces using the **Identify Data Structure** tab.

You can view and set the register settings for each of the NameSpaces supported by Exerciser in the Namespaces tab.

NOTE: The maximum size allowed for a namespace is 1 GB. You can use the fields Namespace Size and Namespace Capacity to set the namespace size.

You can also import the Identify Data Structure and namespace settings from an xml file that has the data captured using the Keysight PCIe Analyzer. You use the **Import from analyzer** toolbar button to do this import.



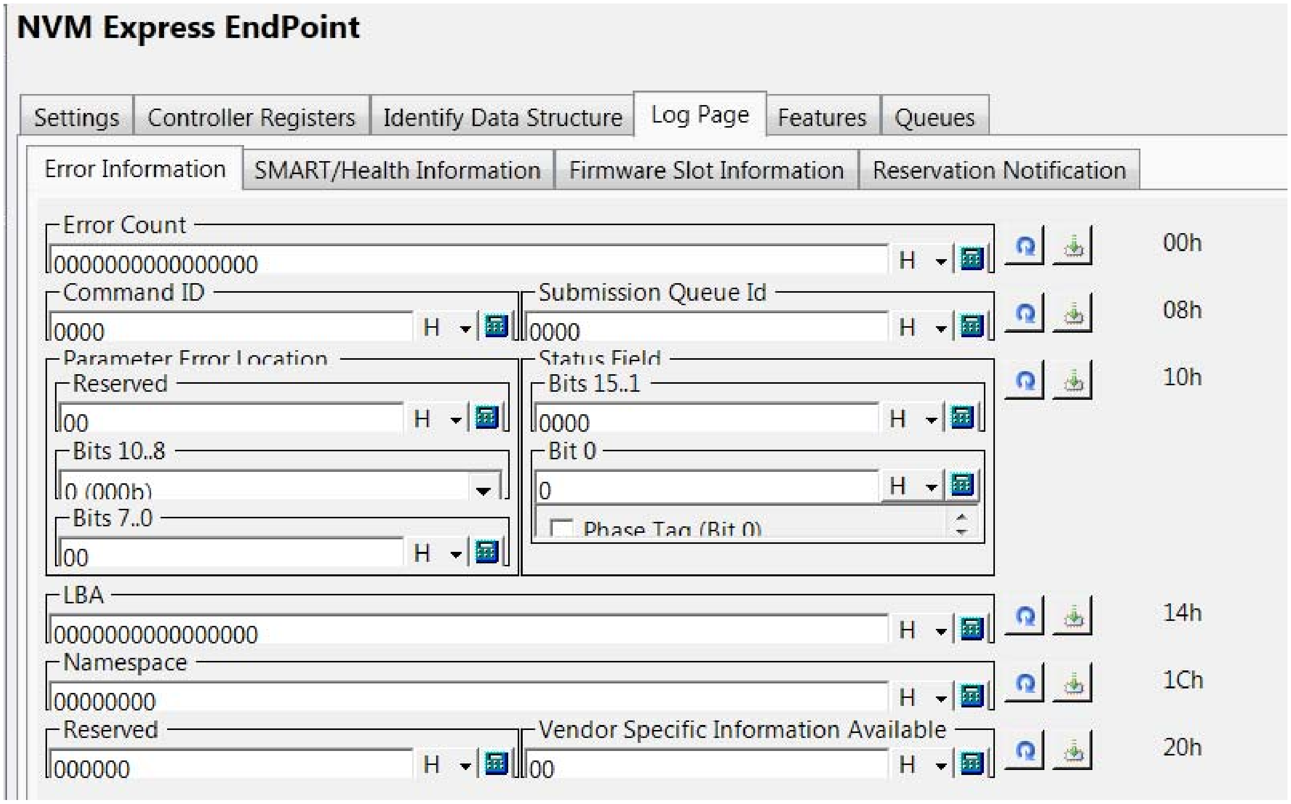
NVM Express EndPoint

PCI SubSystem Vendor ID(SSVID): 0000 H — PCI Vendor ID(VID): 0000 H — 00h
Serial Number (SN)(Bytes 3..0): 00000000 H — 04h
Serial Number (SN)(Bytes 7..4): 00000000 H — 08h
Serial Number (SN)(Bytes 11..8): 00000000 H — 0Ch
Serial Number (SN)(Bytes 15..12): 00000000 H — 10h
Serial Number (SN)(Bytes 19..16): 00000000 H — 14h
Model Number (MN)(Bytes 3..0): 00000000 H — 18h
Model Number (MN)(Bytes 7..4): 00000000 H — 1Ch
Model Number (MN)(Bytes 11..8): 00000000 H — 20h
Model Number (MN)(Bytes 15..12): 00000000 H — 24h
Model Number (MN)(Bytes 19..16): 00000000 H — 28h
Model Number (MN)(Bytes 23..20): 00000000 H — 2Ch
Model Number (MN)(Bytes 27..24): 00000000 H — 30h

Editing Log Page

You can edit the log page structures using the **Log Page** tab.

You can also import the log page structures from an xml file that has the data captured using the Keysight PCIe Analyzer. You use the **Import from analyzer** toolbar button to do this import.



NVM Express EndPoint

Error Count: 0000000000000000 H — 00h

Command ID: 0000 H; Submission Queue Id: 0000 H — 08h

Parameter Error Location: Reserved: 00 H; Bits 10..8: 0 (000b); Bits 7..0: 00 H

Status Field: Bits 15..1: 0000 H; Bit 0: 0 H; □ Phase Tag (Bit 0) — 10h

LBA: 0000000000000000 H — 14h

Namespace: 00000000 H — 1Ch

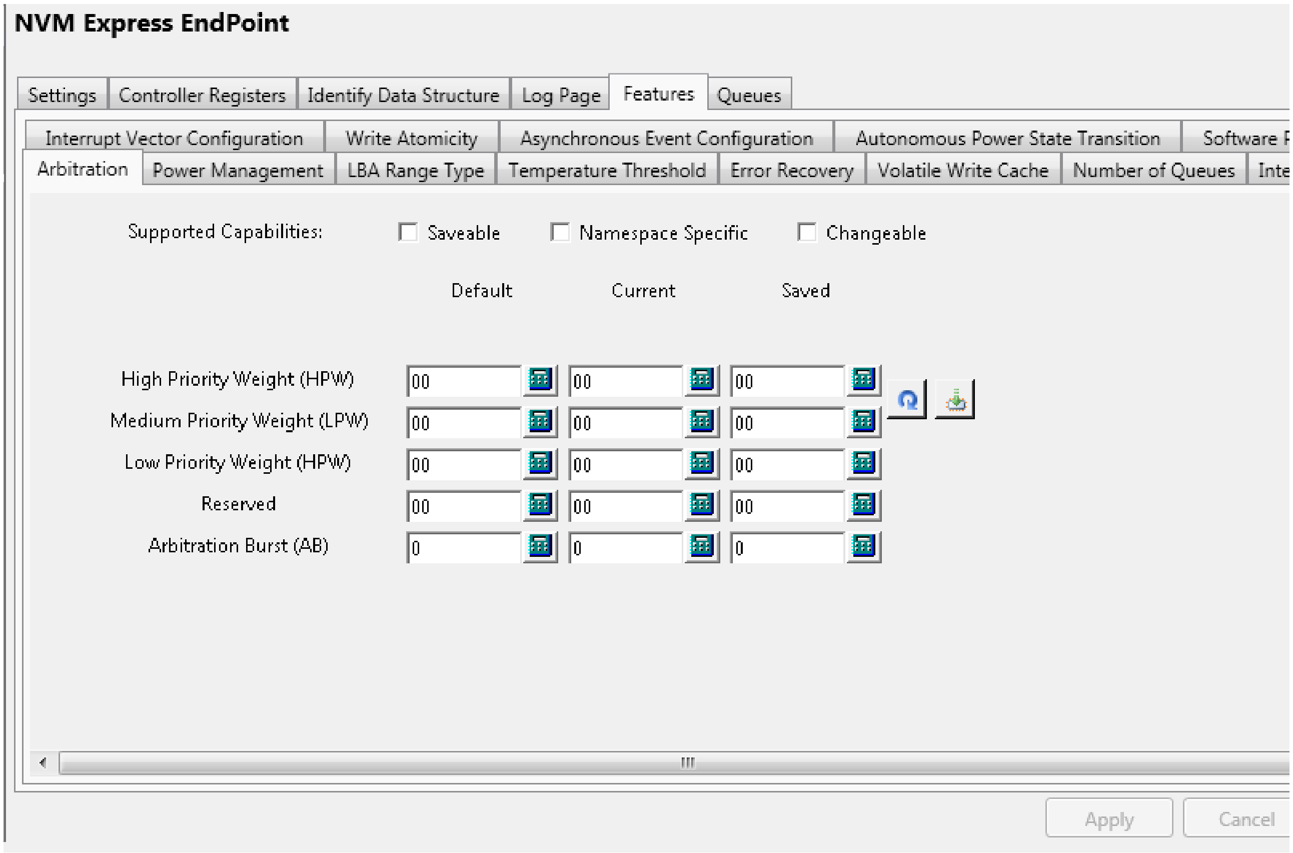
Reserved: 000000 H; Vendor Specific Information Available: 00 H — 20h

Log page has the following tabs:

- Error Information
- Smart/Health Information
- Firmware Slot Information
- Reservation Notification

Editing Values for Feature Commands the Commands

You can edit the values to be used with the Get and Set Features commands. You can view and edit the default, current, and saved values for the different features using the **Features** tab.

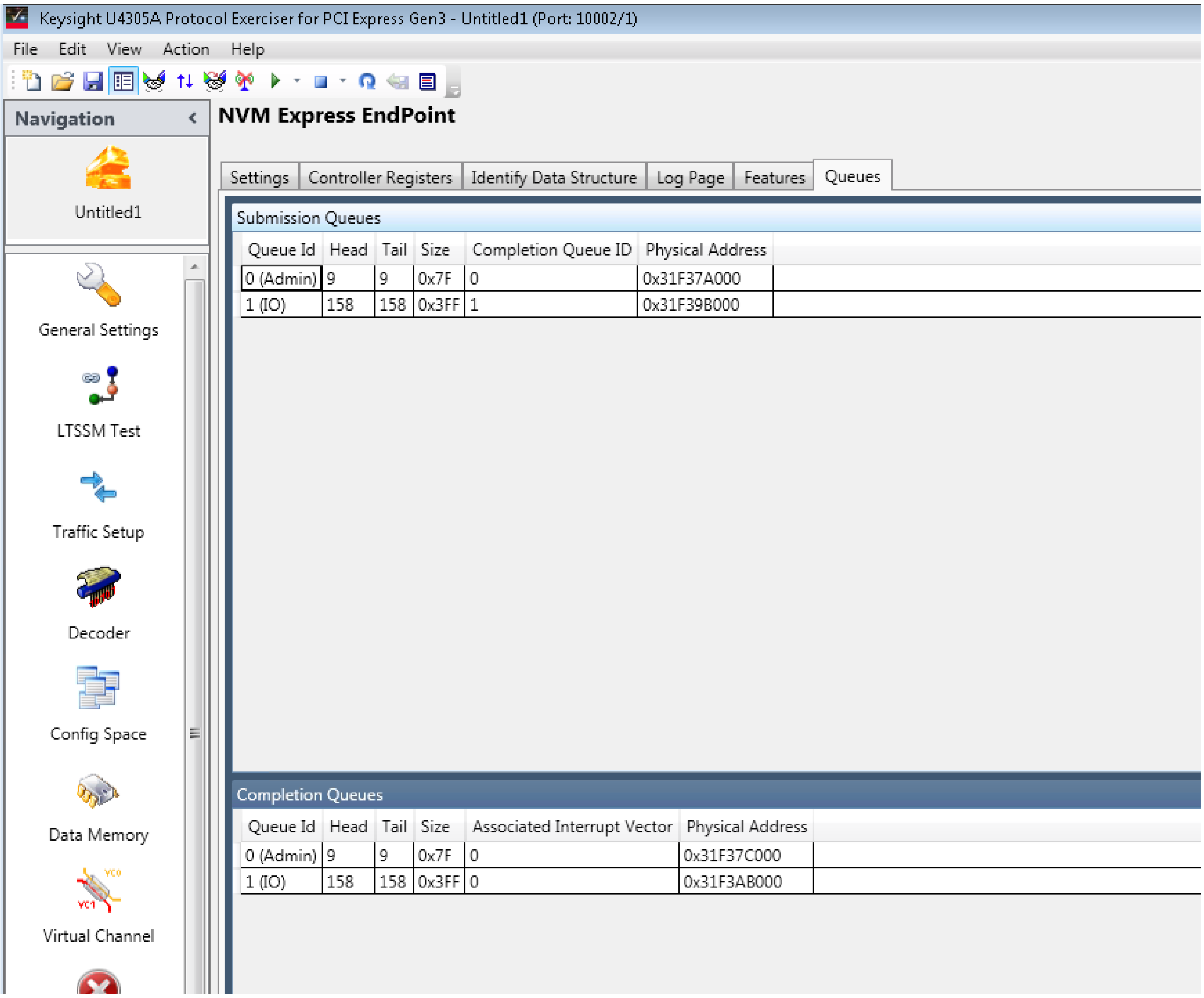


Supported Capabilities: ☐ Saveable ☐ Namespace Specific ☐ Changeable

	Default	Current	Saved
High Priority Weight (HPW)	00	00	00
Medium Priority Weight (LPW)	00	00	00
Low Priority Weight (HPW)	00	00	00
Reserved	00	00	00
Arbitration Burst (AB)	0	0	0

Viewing Submission and Completion Queues

The **Queues** tab shows submission and completion queues that have been created for the NVMe traffic and queue information such as size, tail, head, Completion Queue ID, Associated Interrupt Vector and Physical Address.



Submission Queues

Queue Id	Head	Tail	Size	Completion Queue ID	Physical Address
0 (Admin)	9	9	0x7F	0	0x31F37A000
1 (IO)	158	158	0x3FF	1	0x31F39B000

Completion Queues

Queue Id	Head	Tail	Size	Associated Interrupt Vector	Physical Address
0 (Admin)	9	9	0x7F	0	0x31F37C000
1 (IO)	158	158	0x3FF	0	0x31F3AB000

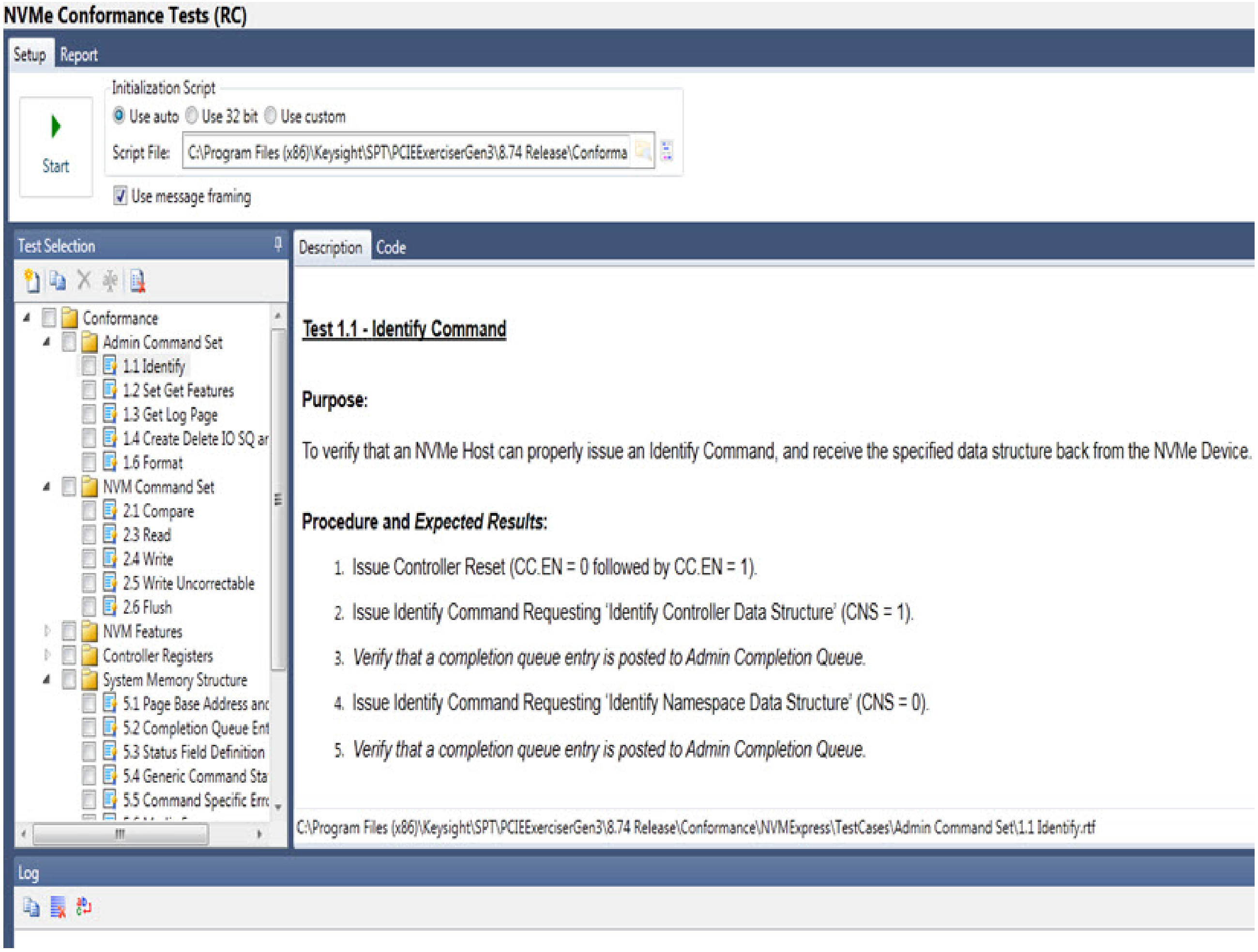
13 Testing DUT’s NVMe Compliance using NVMe Conformance Suite

This chapter provides detailed information on Protocol Exerciser's use as an NVMe Conformance Test Suite to test a DUT's NVMe compliance. This chapter describes Protocol Exerciser's NVMe Conformance, which is integrated with the exerciser, and how to use it to run NVMe Conformance tests.

NVMe Conformance Test Suite-Introduction

NVMe Conformance Test Suite is a tool that helps you to test the NVMe conformance of a DUT. This tool, which is integrated in the PCIe Exerciser GUI, provides you a set of predefined NVMe Conformance test cases to help you assess if the DUT’s NVMe capabilities are as per the NVMe specifications.

Before 8.74 release of Exerciser, you had to configure registers, decoders, and submission and completion queue settings of Exerciser manually for its emulation as an NVMe root complex. With the introduction of NVMe Conformance Test Suite, the process of value entries to the controller register, and submission and completion queues has been automated to simplify the NVMe root complex emulation. Now, you can use Exerciser to test DUT's NVMe conformance when emulating the root complex using NVMe Conformance Test Suite. The screen below displays the NVMe Conformance Test screen:



● Use auto ○ Use 32 bit ○ Use custom

Script File: C:\Program Files (x86)\Keysight\SPT\PCIEExerciserGen3\8.74 Release\Conforma

☑ Use message framing

Test 1.1 - Identify Command

Purpose:

To verify that an NVMe Host can properly issue an Identify Command, and receive the specified data structure back from the NVMe Device.

Procedure and *Expected Results*:

1. Issue Controller Reset (CC.EN = 0 followed by CC.EN = 1).
2. Issue Identify Command Requesting 'Identify Controller Data Structure' (CNS = 1).
3. *Verify that a completion queue entry is posted to Admin Completion Queue.*
4. Issue Identify Command Requesting 'Identify Namespace Data Structure' (CNS = 0).
5. *Verify that a completion queue entry is posted to Admin Completion Queue.*

C:\Program Files (x86)\Keysight\SPT\PCIEExerciserGen3\8.74 Release\Conformance\NVMExpress\TestCases\Admin Command Set\1.1 Identify.rtf

NVMe Conformance Test Suite is a set of test cases. Each time when you press the Start button after selecting the test cases in the NVMe Conformance Test window, a controller reset occurs, an initialization script executes and test cases start executing automatically.

In the context of NVMe Conformance Test Suite, you can use Exerciser to:

- run NVMe commands to test the NVMe compliance of the DUT. The commands are available in the form of test cases.
- create your own sets of test cases based on your requirements.
- view the description (purpose, procedure and expected results) of the selected NVMe conformance test.
- view the log details of the steps being performed during the execution of test cases and save these for further use.
- view the detailed report summary of the executed NVMe Conformance tests and save these for later use.

As an NVMe Conformance test suite, Exerciser submits various requests (NVMe commands) to an NVMe DUT for completion. By sending NVMe command requests to the DUT, you can check how the NVMe host controller handles user requests, responds to and completes these requests, and verify the NVMe conformance capabilities of the DUT.

NOTE

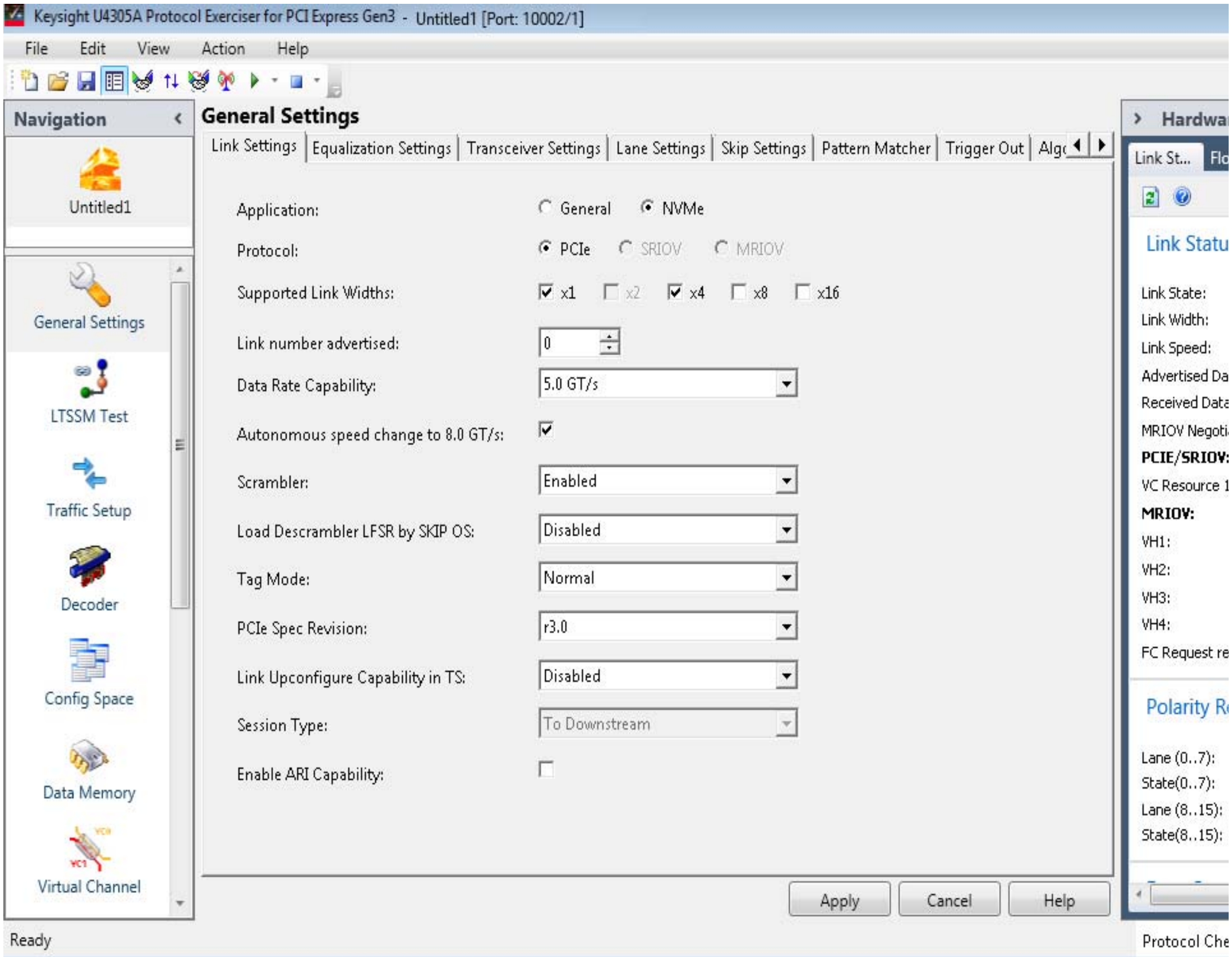
In this release, Exerciser can function as a root complex only when you run its NVMe Conformance test suite. It does not currently support the endpoint emulation.

Getting Started with NVMe Conformance

Prerequisites for NVMe Conformance

- You must have purchased and installed the NVMe license to get the NVMe conformance functionality.
- Ensure that all the test specific prerequisites are met before executing the test.
- Ensure that NVMe is enabled in the General Setting Tab of the Protocol Exerciser GUI.
- Ensure that the DUT is connected to the Protocol Exerciser and the link between the two is up.

To start with the NVMe Conformance, first you need to do settings in the Link Settings tab of the General Setting page of the Protocol Exerciser GUI. For this, click the '**General Settings**' icon from the left navigation pane of the GUI and enable **NVMe** by clicking the **NVMe** radio button as displayed in the following screen.



Application: ○ General ● NVMe

Protocol: ● PCIe ○ SRIOV ○ MRIOV

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

Link number advertised: 0

Data Rate Capability: 5.0 GT/s

Autonomous speed change to 8.0 GT/s: ☑

Scrambler: Enabled

Load Descrambler LFSR by SKIP OS: Disabled

Tag Mode: Normal

PCIe Spec Revision: r3.0

Link Upconfigure Capability in TS: Disabled

Session Type: To Downstream

Enable ARI Capability: □

Clicking the '**Apply**' button saves all the changes that you made to the Settings tab of the protocol exerciser.

NOTE

You only need to enable **NVMe** in the **General Settings** tab; remaining settings are done by the initialization script.

Accessing NVMe Conformance

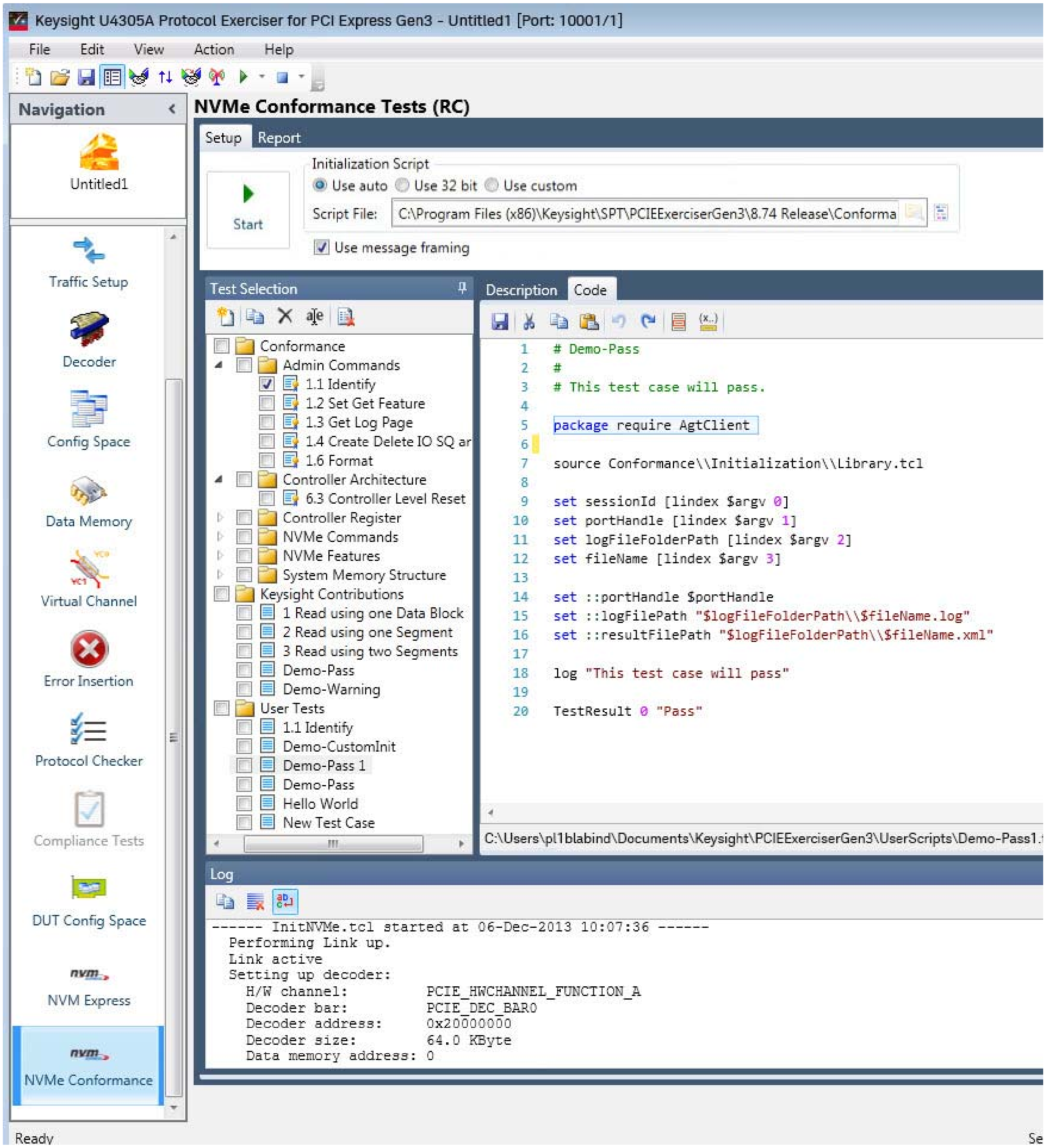
To access NVMe Conformance

1. Start the Protocol Exerciser GUI.

   For information on starting the Protocol Exerciser GUI, refer to “Starting and Exiting the Protocol Exerciser GUI.

2. Click the **NVMe Conformance** icon in the Navigation window.

   The NVMe Conformance Test screen is displayed in the main Protocol Exerciser window.



Initialization Script
● Use auto ○ Use 32 bit ○ Use custom
Script File: C:\Program Files (x86)\Keysight\SPT\PCIEExerciserGen3\8.74 Release\Conforma
☑ Use message framing

☑ 1.1 Identify

```
# Demo-Pass
#
# This test case will pass.

package require AgtClient

source Conformance\\Initialization\\Library.tcl

set sessionId [lindex $argv 0]
set portHandle [lindex $argv 1]
set logFileFolderPath [lindex $argv 2]
set fileName [lindex $argv 3]

set ::portHandle $portHandle
set ::logFilePath "$logFileFolderPath\\$fileName.log"
set ::resultFilePath "$logFileFolderPath\\$fileName.xml"

log "This test case will pass"

TestResult 0 "Pass"
```

C:\Users\pl1blabind\Documents\Keysight\PCIEExerciserGen3\UserScripts\Demo-Pass1.

```
------ InitNVMe.tcl started at 06-Dec-2013 10:07:36 ------
  Performing Link up.
  Link active
  Setting up decoder:
    H/W channel:          PCIE_HWCHANNEL_FUNCTION_A
    Decoder bar:          PCIE_DEC_BAR0
    Decoder address:      0x20000000
    Decoder size:         64.0 KByte
    Data memory address:  0
```

NVMe Conformance Tests are grouped under multiple headings in the **Test Selection** pane of the **NVMe Conformance Tests (RC)** window of the Protocol Exerciser GUI. On clicking a test, its description and sequence of execution steps are displayed in the Description pane. The Description pane displays the purpose, procedure and expected result of the selected test case, whereas the code tab displays the code snippets of the same.

When a test case is in execution, the Log pane keeps updating the details of the steps that the NVMe Conformance is performing during the test execution. It displays the prerequisites that are checked, the steps that are performed (such as performing link-up, setting up decoder, scanning DUT Config space, initializing submission and completion queue, executing Initialization script successfully etc.) and the steps that are being performed during the test execution. Based on whether all the test prerequisites are met and how the step sequence of test case execution proceeds, the Log pane displays test result as Passed, Failed or Passed but with Warning.

NOTE

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

How NVMe Conformance Works

This topic describes the overall working of the NVMe Conformance Test Suite.

NVMe Conformance Test Suite is fully integrated with the PCIe Exerciser which lets you to check the NVMe Compliance of the DUT. All the conformance tests are available in the Test Selection Pane of the NVMe Conformance Test window. When you run a conformance test, the NVMe conformance suite check the prerequisites and if the test prerequisites are met, it executes that test's steps taking all the required link, lane, decoder, and other settings as inputs.

While a test is in execution, NVMe Conformance Suite keeps updating the log status of the link, decoders, and other relevant settings, that are impacted by the test execution in the Hardware Status Window. The test results are displayed in the test log pane of the GUI. NVMe Conformance also displays the timestamp when the initialization script started executing and when '.tcl' file (a regression script) of the respective command set called to execute the test case.

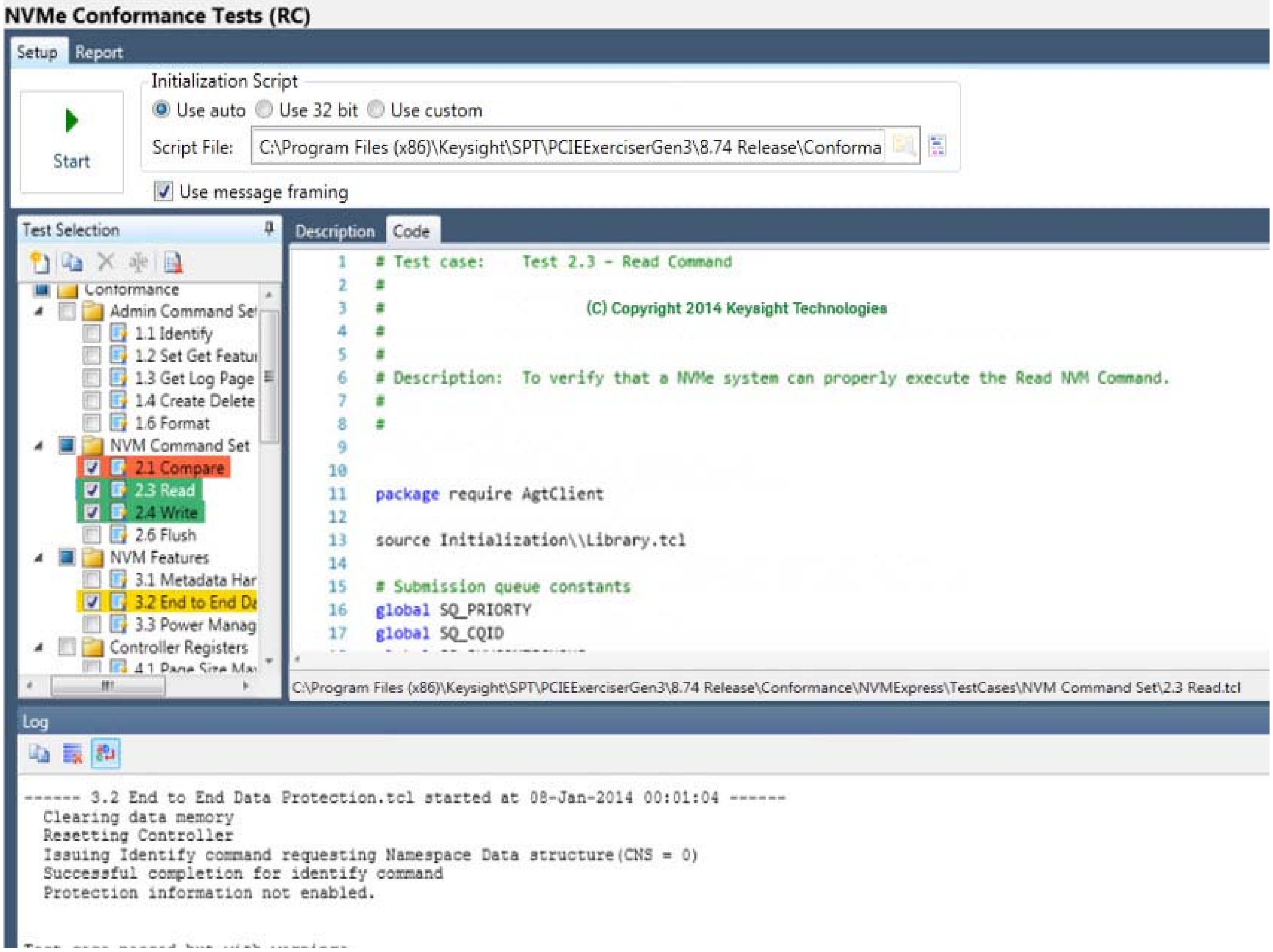
Flow of Test Case Execution:

When you select a test case and each time you press the '**Start**' button (to run the test cases) in the NVMe conformance test window, Exerciser performs the following steps:

1. Check for the pre-requisites. If the pre-requisites are met, Exerciser scans DUT's config space, notes its capabilities and configures the DUT.
2. Configure **MSI-X** and decoder settings, and assert controller reset.
3. Initialize Admin queues (submission and completion queue) of DUT.
4. After successful completion of 'Initialization Script', Conformance Suite performs the test case execution.
5. Run specific code of the test case.
6. Run the test and display the result in the log pane as 'passed', 'failed', and 'passed with warning message'.
7. Once all the selected test cases are executed, a final consolidated report is compiled stating how many test cases are executed, how many passed and how many failed.

Consider an example of an NVMe 'End to End Data Protection' test case to verify whether an NVMe Conformance system can properly execute this command.

When you execute the 'End to End Data Protection' command from the 'NVMe Features' test category of Conformance Test group, NVMe Conformance Suite calls InitNVMe.tcl and starts the execution of initialization script. During this process, it performs link up, sets up decoder, scans DUT config space, enables MSI-X, initializes admin completion and submission queue and keeps updating the status in the log pane. Once the Initialization script executes successfully, it starts executing the test case (here 'End to End Data Protection' test case). It starts executing the 'End to End Data Protection.tcl'. When the test execution completes, it displays the result appearing in yellow color. In this example, the test case passed with a warning because protection information were not enabled. You can view the warning message in the summary section of the Report tab of the NVMe Conformance Tests window.



● Use auto ○ Use 32 bit ○ Use custom

Script File: C:\Program Files (x86)\Keysight\SPT\PCIEExerciserGen3\8.74 Release\Conforma

☑ Use message framing

```
# Test case:    Test 2.3 - Read Command
#
#                       (C) Copyright 2014 Keysight Technologies
#
#
# Description:  To verify that a NVMe system can properly execute the Read NVM Command.
#
#


package require AgtClient

source Initialization\\Library.tcl

# Submission queue constants
global SQ_PRIORTY
global SQ_CQID
```

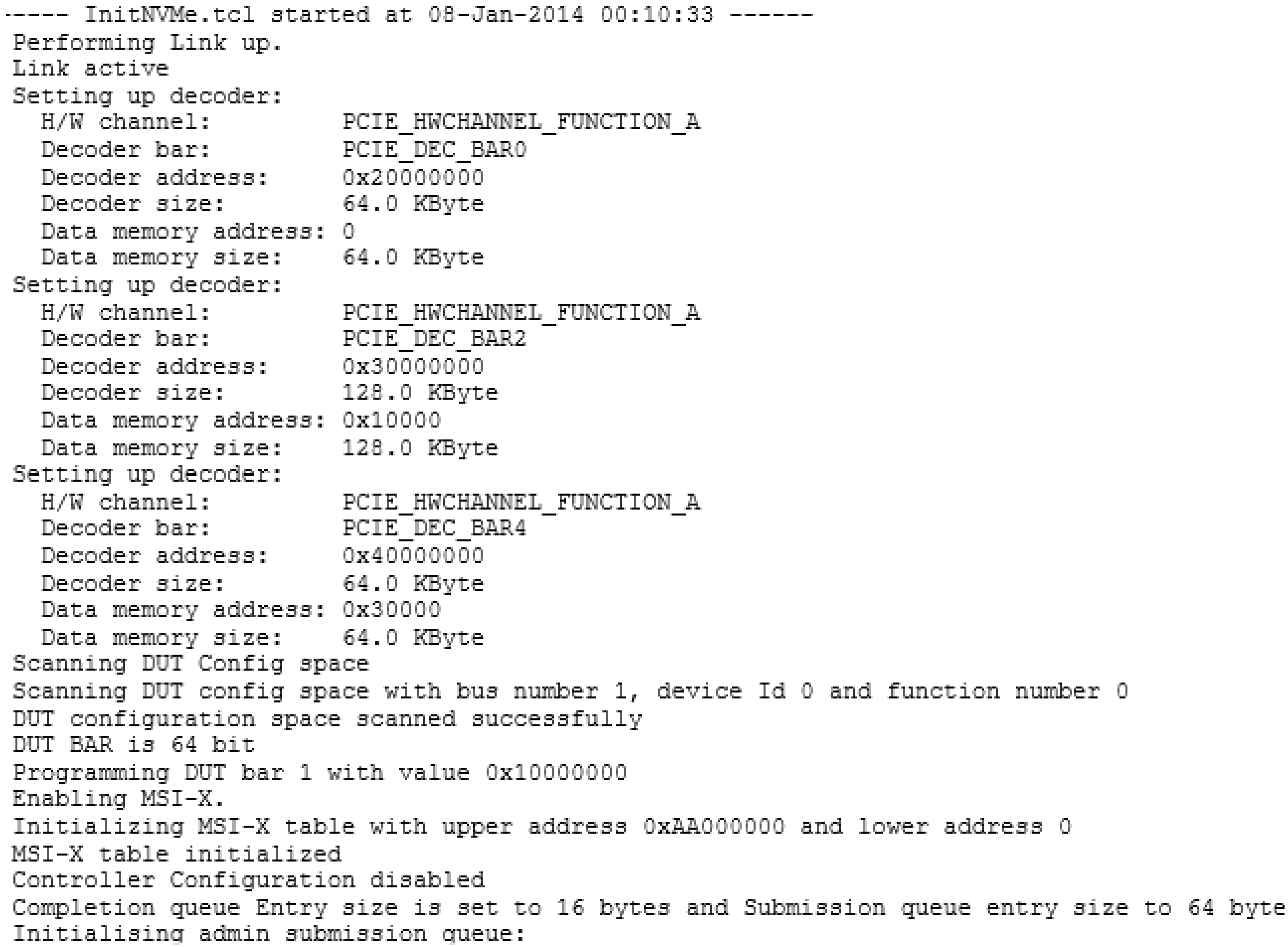
C:\Program Files (x86)\Keysight\SPT\PCIEExerciserGen3\8.74 Release\Conformance\NVMExpress\TestCases\NVM Command Set\2.3 Read.tcl

```
------ 3.2 End to End Data Protection.tcl started at 08-Jan-2014 00:01:04 ------
  Clearing data memory
  Resetting Controller
  Issuing Identify command requesting Namespace Data structure(CNS = 0)
  Successful completion for identify command
  Protection information not enabled.
```

NOTE

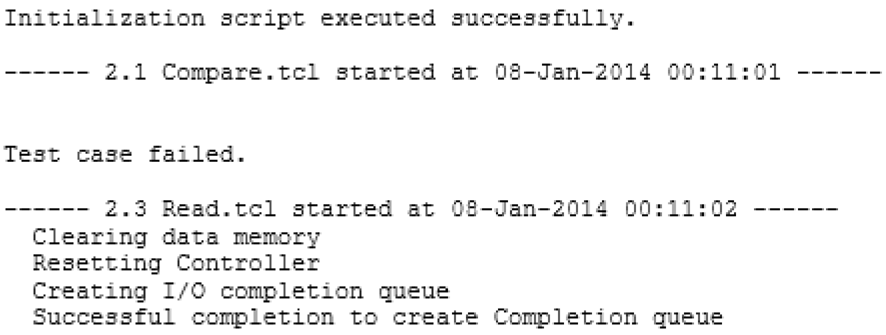
Test cases 'Passed', 'Failed' or 'Passed but with warning' are displayed in specific colors. Green color symbolizes a passed test case, red color symbolizes a failed test case, and yellow color symbolizes a test cases passed but with warning. You can view warning message in the summary section of the test report displayed in the Report tab of the NVMe Conformance Test window.

The following screen displays a sample test log generated on execution of initialization script before the test case execution:



```
----- InitNVMe.tcl started at 08-Jan-2014 00:10:33 ------
 Performing Link up.
 Link active
 Setting up decoder:
   H/W channel:         PCIE_HWCHANNEL_FUNCTION_A
   Decoder bar:         PCIE_DEC_BAR0
   Decoder address:     0x20000000
   Decoder size:        64.0 KByte
   Data memory address: 0
   Data memory size:    64.0 KByte
 Setting up decoder:
   H/W channel:         PCIE_HWCHANNEL_FUNCTION_A
   Decoder bar:         PCIE_DEC_BAR2
   Decoder address:     0x30000000
   Decoder size:        128.0 KByte
   Data memory address: 0x10000
   Data memory size:    128.0 KByte
 Setting up decoder:
   H/W channel:         PCIE_HWCHANNEL_FUNCTION_A
   Decoder bar:         PCIE_DEC_BAR4
   Decoder address:     0x40000000
   Decoder size:        64.0 KByte
   Data memory address: 0x30000
   Data memory size:    64.0 KByte
 Scanning DUT Config space
 Scanning DUT config space with bus number 1, device Id 0 and function number 0
 DUT configuration space scanned successfully
 DUT BAR is 64 bit
 Programming DUT bar 1 with value 0x10000000
 Enabling MSI-X.
 Initializing MSI-X table with upper address 0xAA000000 and lower address 0
 MSI-X table initialized
 Controller Configuration disabled
 Completion queue Entry size is set to 16 bytes and Submission queue entry size to 64 byte
 Initialising admin submission queue:
```

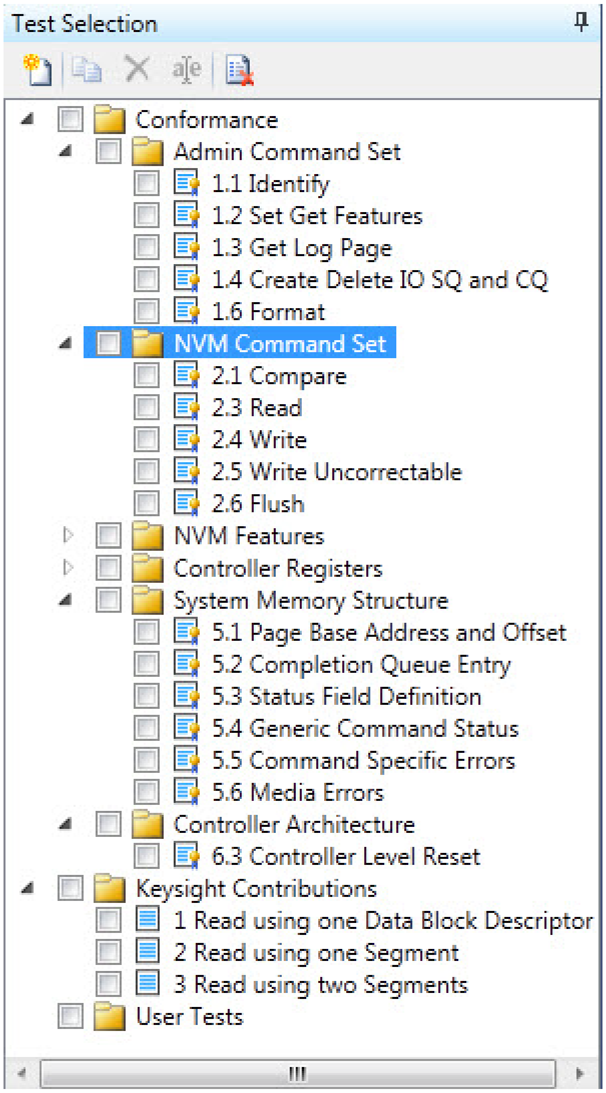
The following screen displays a sample test log generated on execution of a test case after the successful execution of the initialization script:



```
Initialization script executed successfully.

------ 2.1 Compare.tcl started at 08-Jan-2014 00:11:01 ------


Test case failed.

------ 2.3 Read.tcl started at 08-Jan-2014 00:11:02 ------
  Clearing data memory
  Resetting Controller
  Creating I/O completion queue
  Successful completion to create Completion queue
```

NVMe Conformance Test Cases- An Overview

In the NVMe Conformance Test Suite, NVMe Conformance Tests are categorized into the following different sets each having further subsets. These test categories are explained below. Following is the list of test cases organized under three headings as shown in the below screen:



Test Selection

- □ Conformance
  - □ Admin Command Set
    - □ 1.1 Identify
    - □ 1.2 Set Get Features
    - □ 1.3 Get Log Page
    - □ 1.4 Create Delete IO SQ and CQ
    - □ 1.6 Format
  - □ NVM Command Set
    - □ 2.1 Compare
    - □ 2.3 Read
    - □ 2.4 Write
    - □ 2.5 Write Uncorrectable
    - □ 2.6 Flush
  - □ NVM Features
  - □ Controller Registers
  - □ System Memory Structure
    - □ 5.1 Page Base Address and Offset
    - □ 5.2 Completion Queue Entry
    - □ 5.3 Status Field Definition
    - □ 5.4 Generic Command Status
    - □ 5.5 Command Specific Errors
    - □ 5.6 Media Errors
  - □ Controller Architecture
    - □ 6.3 Controller Level Reset
- □ Keysight Contributions
  - □ 1 Read using one Data Block Descriptor
  - □ 2 Read using one Segment
  - □ 3 Read using two Segments
- □ User Tests

Double-clicking the individual test category will expand and collapse the list of test cases displaying the subset of commands listed under these categories.

Conformance Tests

Conformance tests are test cases that are written on the basis of PCIe and NVMe specifications. This category comprises of various sets of commands each having their own subsets.

- **Admin Commands**- The command falling under this category is defined to be submitted to the Admin Submission Queue. The commands belonging to this category are listed below:
  - Identify
  - Set Get Features

  - Get Log Page
  - Create Delete IO, SQ, and CQ
  - Format
- **NVM Command Set**- These commands are submitted by the host only when I/O submission queue(s) and I/O completion queue(s) are created and the controller is ready to execute command. The set of commands grouped under this category are listed below:
  - Compare
  - Read
  - Write
  - Flush
- **NVMe Features**- This test category is introduced to perform the conformance verification for NVMe products by implementing NVM Features command set.
  - Metadata Handling
  - End to End Data Protection
  - Power Management
- **Controller Registers**- The set of commands grouped under this category are meant for register mapping for the controller. Controller Register contains following set of commands:
  - Page Size Max
  - Page Size Min
  - Command Sets Supported
  - Doorbell Stride
  - Timeout
  - Arbitration Mechanism
  - Contiguous Queues Required
  - Max Queue Entries
  - Interrupt Mask Set
  - IO CQ Entry Size
  - IO SQ Entry Size
  - Shutdown Notification
  - Arbitration Mechanism Selected
  - Command Sets Selected
  - Enable
  - Shutdown Status
  - Controller Fatal Status
- **System Memory Structure**- The System Memory Structure category enlists the commands that address the system memory structures used by NVMe Conformance Suite. The commands under this category are used to verify whether an NVMe host can properly return the status value of the issued command. The following command list falls under this category:
  - Page Base Address and Offset
  - Completion Queue Entry
  - Status Field Definition
  - Generic Command Status
  - Command Specific Errors
- **Controller Architecture**- This category lets you perform the test that involves controller level reset mechanism. The following command is enlisted under this category:
  - Controller Level Reset

User Tests

This section lets you write your own test cases depending upon your requirements.

NOTE You can view the description (purpose, procedure and expected results) of each of the above listed commands in the 'Description Pane' of the NVMe Conformance Test screen.

Out of the above-mentioned categories, the test cases under the 'User Tests' heading are editable whereas Conformance Tests are non-editable. You can create your own test case, make a duplicate copy of the existing test case and make some changes according to your requirements, delete any test case which you think of no more use, rename any test case etc under the 'User Tests' heading.

To Create a New Test Case:

Clicking button in the Test Selection Pane enables you to create a new user test case as per your specific requirements.

To Create a Duplicate copy of the existing Test Case

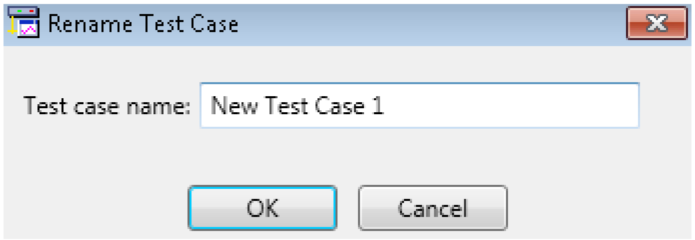
Clicking button enables you to create a duplicate copy of the existing test case and modify it as per your requirement.

To Delete a Test Case

Select a test case you want to delete and click ✕ button available in the Test Selection Pane.

To Rename a Test Case

Clicking button lets you rename an existing test case. Once you click this button after selecting a test case, a pop-up appears which asks you to enter the relevant test case name.



Rename Test Case

Test case name: New Test Case 1

Enter test case name and click OK. Test Case will be renamed.

Running the NVMe Conformance Test Cases

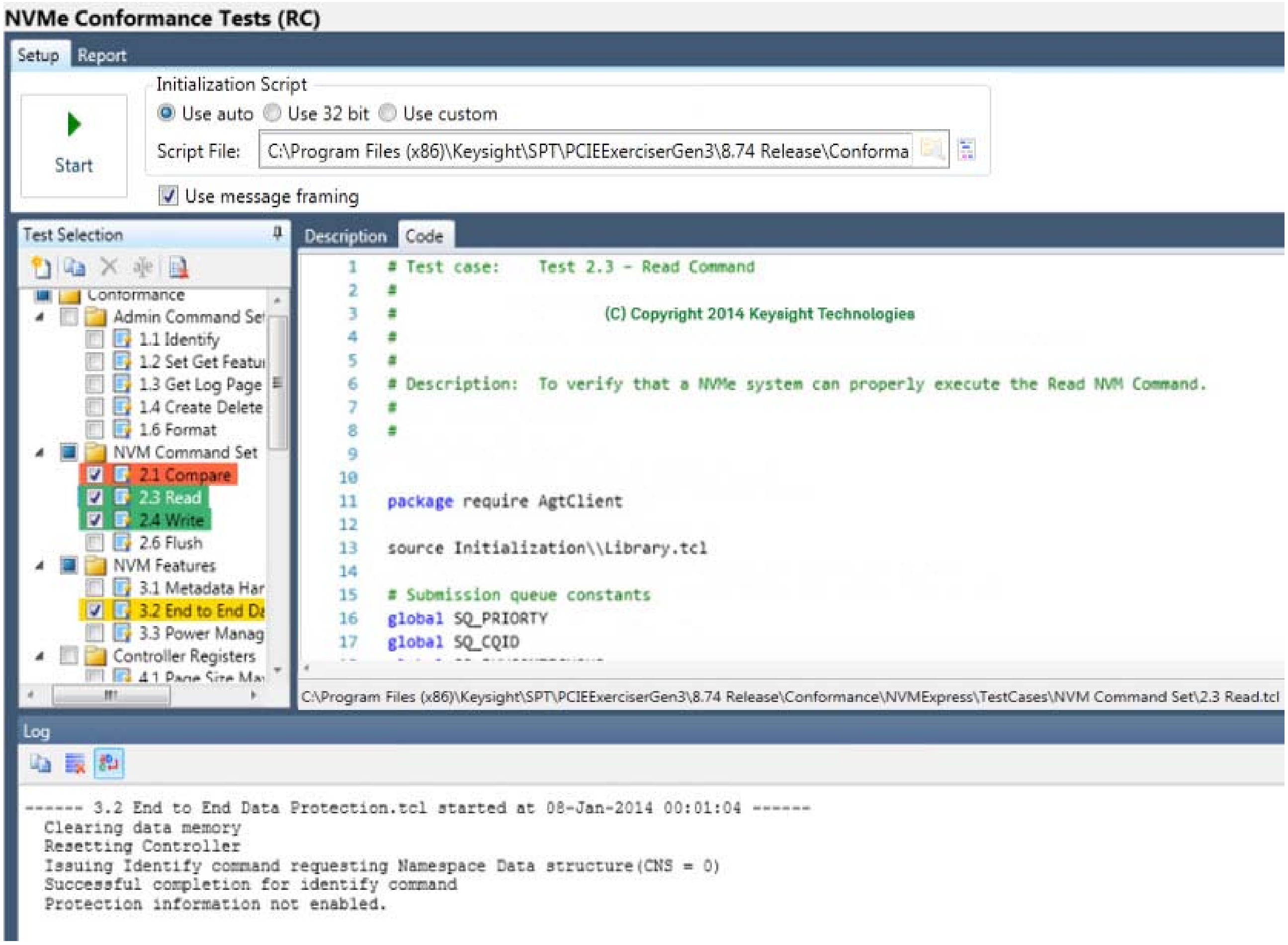
To run test cases

1 Select an initialization script. Exerciser provides two initialization scripts to initialize NVMe devices with 64bit and 32bit BARs. If you have a 64bit device, select the **Use auto** option. If you have e a 32bit device, select either **Use auto** or **Use 32 bit**. If you want to use your own customized initialization script, select **Use custom**.

2 Select the test case from the Test Selection pane of NVMe Conformance RC window.

3 Press the **Start** button.

The following screen displays the test execution result.



Initialization Script

● Use auto ○ Use 32 bit ○ Use custom

Script File: C:\Program Files (x86)\Keysight\SPT\PCIEExerciserGen3\8.74 Release\Conforma

☑ Use message framing

```
# Test case:    Test 2.3 - Read Command
#
#                        (C) Copyright 2014 Keysight Technologies
#
#
# Description:  To verify that a NVMe system can properly execute the Read NVM Command.
#
#


package require AgtClient

source Initialization\\Library.tcl

# Submission queue constants
global SQ_PRIORTY
global SQ_CQID
```

C:\Program Files (x86)\Keysight\SPT\PCIEExerciserGen3\8.74 Release\Conformance\NVMExpress\TestCases\NVM Command Set\2.3 Read.tcl

```
------ 3.2 End to End Data Protection.tcl started at 08-Jan-2014 00:01:04 ------
  Clearing data memory
  Resetting Controller
  Issuing Identify command requesting Namespace Data structure(CNS = 0)
  Successful completion for identify command
  Protection information not enabled.
```

The color code in the above screen have their significant meaning. Green color symbolizes a passed test case, red color symbolizes a failed test case, and yellow color symbolizes a test cases passed but with warning.

Understanding the Test Results

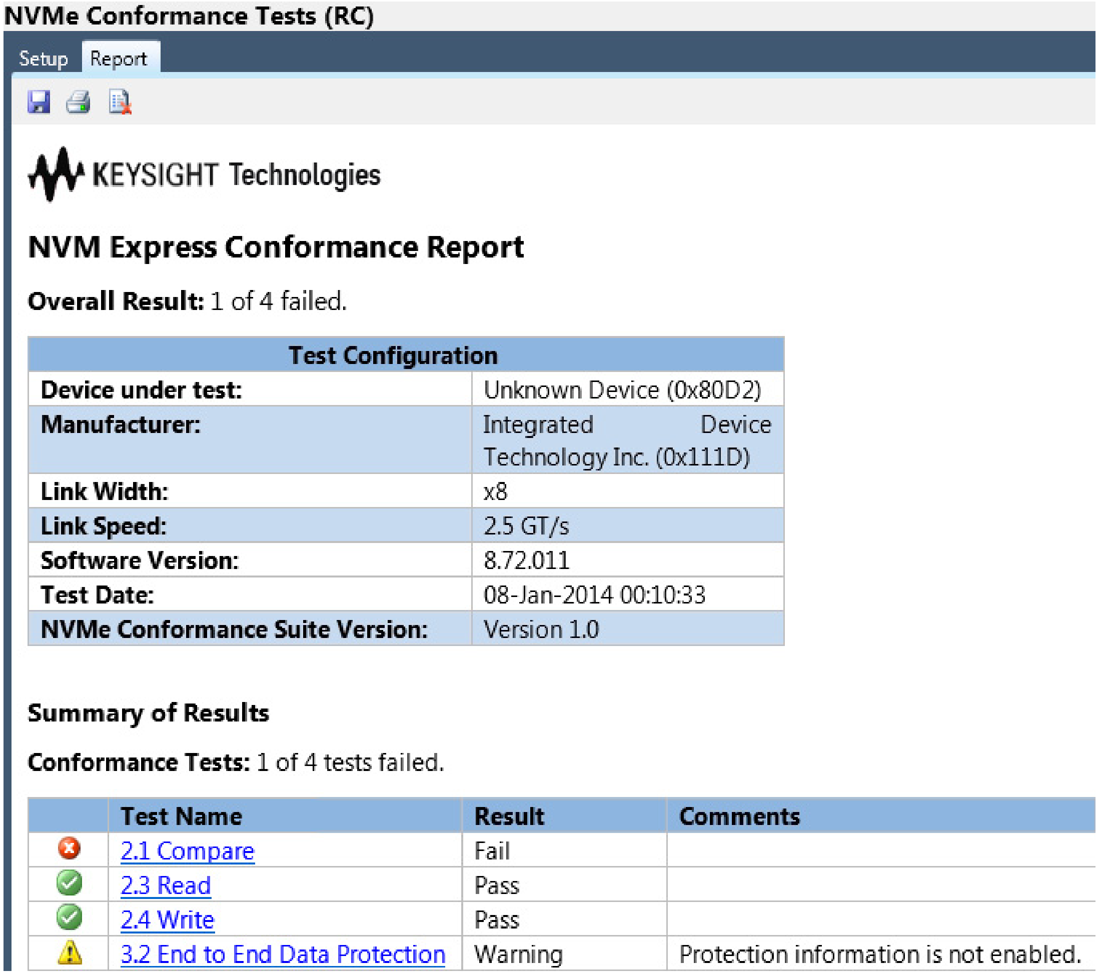
The test results are displayed in the Log pane of the NVMe Conformance Tests window. A test log helps to identify the root cause of a problem. For instance, the test log shows The test results are displayed in the Log pane of the NVMe Conformance Tests window. A test log helps to identify the root cause of a problem. For instance, the test log shows warning message, 'Protection information not enabled' in case of End to End Data Protection test case. The Log pane displays the following information with the date and time stamp. The Log pane displays the following information with the date and time stamp.

- Sequence of steps that are being performed during the initialization script execution. It displays the message 'initialization script executed successfully' if the script is executed without any obstruction, and reports the error if failed.
- Name of the executed test.
- The test case result based on whether or not all prerequisites are met and all the test case sequences are completed as per PCIe and NVMe specifications. It displays Passed, Failed, or Passed but with warnings.

You can save the test results as an RTF or a TXT file by clicking the Save button inside the Log pane and clicking Save as... To view the consolidated conformance test case report, click the Report tab of the NVMe Conformance Tests window. For more information on the NVMe Conformance Test Case report, refer to the topic “Viewing, Printing, and Understanding the NVMe Conformance Test Report.

Viewing, Printing, and Understanding the NVMe Conformance Test Report

To view the test case report, click the **Report** tab in the **NVMe Conformance Test** window of the protocol exerciser GUI.



NVM Express Conformance Report

Overall Result: 1 of 4 failed.

Test Configuration	
Device under test:	Unknown Device (0x80D2)
Manufacturer:	Integrated Device Technology Inc. (0x111D)
Link Width:	x8
Link Speed:	2.5 GT/s
Software Version:	8.72.011
Test Date:	08-Jan-2014 00:10:33
NVMe Conformance Suite Version:	Version 1.0

Summary of Results

Conformance Tests: 1 of 4 tests failed.

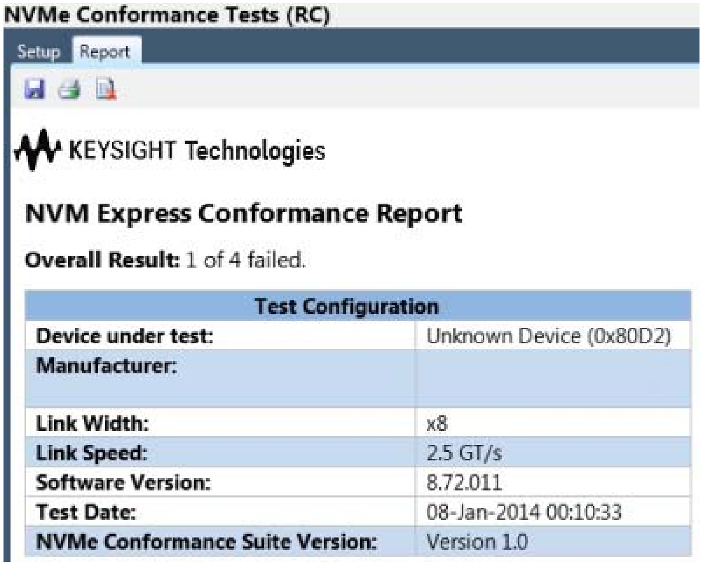
	Test Name	Result	Comments
✖	2.1 Compare	Fail	
✓	2.3 Read	Pass	
✓	2.4 Write	Pass	
⚠	3.2 End to End Data Protection	Warning	Protection information is not enabled.

You can save, print, and clear the conformance test report, by clicking 💾, 🖨, and 📄 button respectively in the Report tab of the NVMe Conformance Tests screen.

Understanding the NVMe Conformance Test Report

The information in this topic will help you understand the different sections of the NVMe Conformance Test Report.

The top section of the conformance test report contains the report header displaying the overall result. The screen that follows displays information about the device under test, its manufacturer, link width and link speed at which test execution occurred, the software version of the exerciser being used, the date and time when test was executed, and the NVMe Conformance Suite version taken into use.



NVMe Conformance Tests (RC)

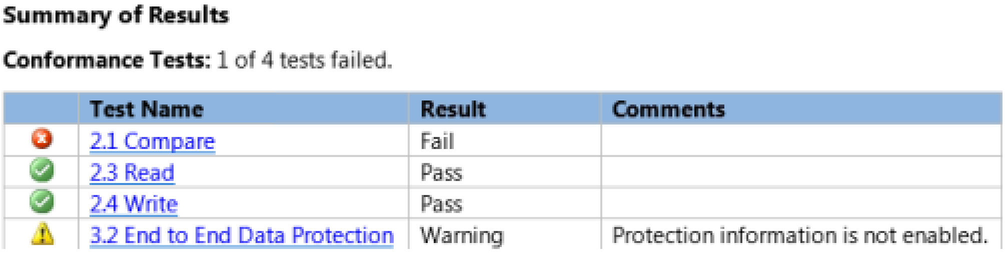
NVM Express Conformance Report

Overall Result: 1 of 4 failed.

Test Configuration	
Device under test:	Unknown Device (0x80D2)
Manufacturer:	
Link Width:	x8
Link Speed:	2.5 GT/s
Software Version:	8.72.011
Test Date:	08-Jan-2014 00:10:33
NVMe Conformance Suite Version:	Version 1.0

Summary Section

The Summary section provides a brief overview of each of the test cases that were run. The **Test Name** column shows which test executed, and **Result** column shows if the test execution passed or failed. **Comment** section displays the information why a test case failed, and also states the warning reasons if a test case passed with a warning.



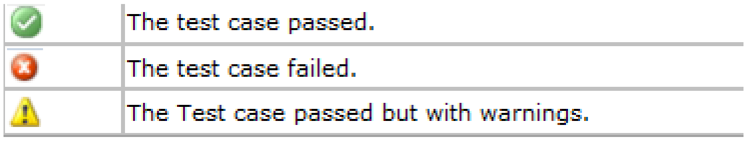
Summary of Results

Conformance Tests: 1 of 4 tests failed.

	Test Name	Result	Comments
✗	2.1 Compare	Fail	
✓	2.3 Read	Pass	
✓	2.4 Write	Pass	
⚠	3.2 End to End Data Protection	Warning	Protection information is not enabled.

Clicking the test case name redirects you to the respective test logs generated in the **Tests Logs** section of the **Report** tab.

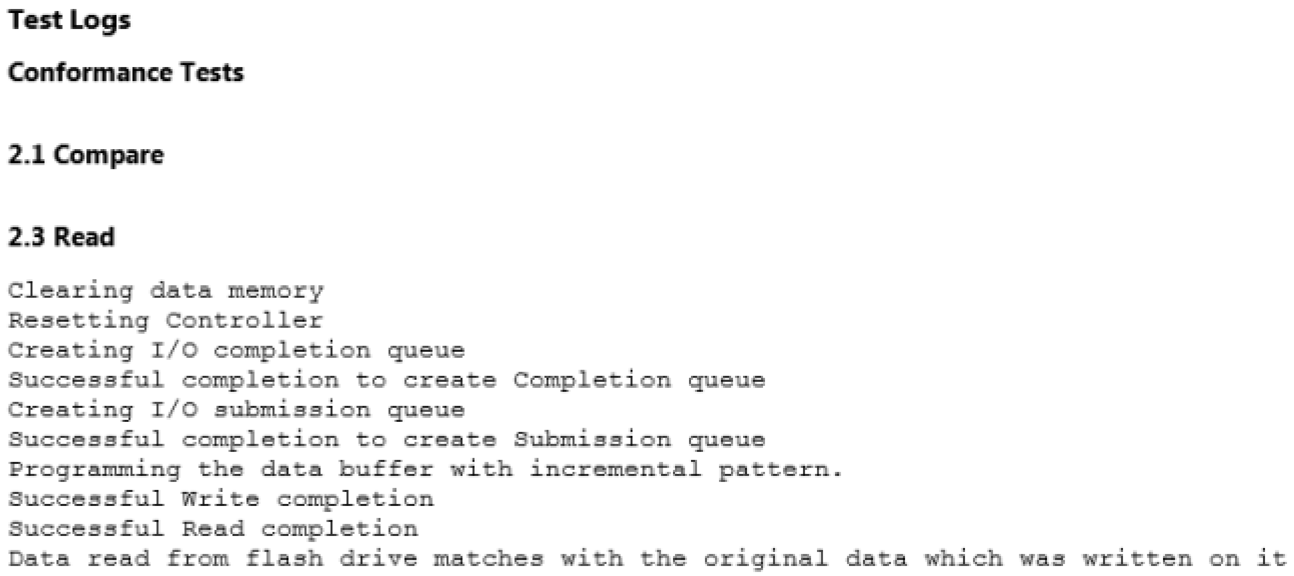
The icons in the above screen have the following meanings:



✓	The test case passed.
✗	The test case failed.
⚠	The Test case passed but with warnings.

Tests Logs

This section displays the log details of the conformance test executed.



Test Logs

Conformance Tests

2.1 Compare

2.3 Read

```
Clearing data memory
Resetting Controller
Creating I/O completion queue
Successful completion to create Completion queue
Creating I/O submission queue
Successful completion to create Submission queue
Programming the data buffer with incremental pattern.
Successful Write completion
Successful Read completion
Data read from flash drive matches with the original data which was written on it
```

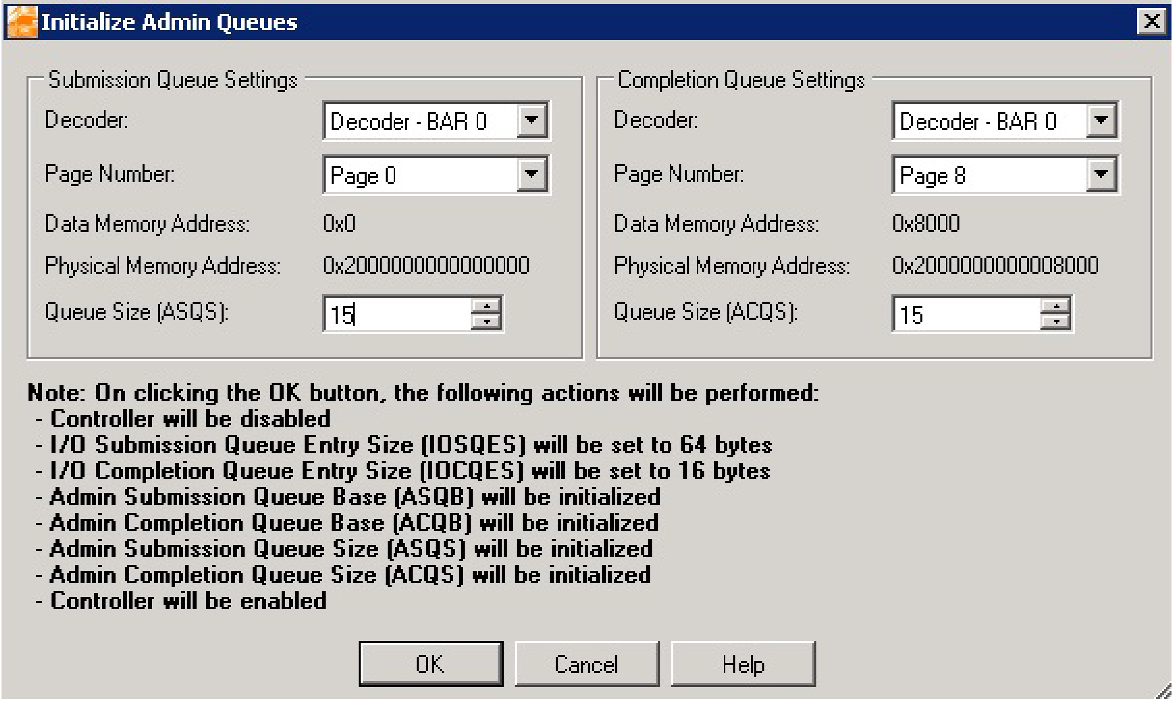
14 Protocol Exerciser GUI Reference

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

Initialize Admin Queues dialog box

To access this dialog: Select **NVM Express > Initialize Admin Queues**. If you do not see the **Initialize Admin Queues** button, then it indicates that the Admin queues have been initialized or a prerequisite is not met due to which some other warning is displayed in this tab.



Initialize Admin Queues

Submission Queue Settings

Decoder: Decoder - BAR 0

Page Number: Page 0

Data Memory Address: 0x0

Physical Memory Address: 0x2000000000000000

Queue Size (ASQS): 15

Completion Queue Settings

Decoder: Decoder - BAR 0

Page Number: Page 8

Data Memory Address: 0x8000

Physical Memory Address: 0x2000000000008000

Queue Size (ACQS): 15

Note: On clicking the OK button, the following actions will be performed:
- **Controller will be disabled**
- **I/O Submission Queue Entry Size (IOSQES) will be set to 64 bytes**
- **I/O Completion Queue Entry Size (IOCQES) will be set to 16 bytes**
- **Admin Submission Queue Base (ASQB) will be initialized**
- **Admin Completion Queue Base (ACQB) will be initialized**
- **Admin Submission Queue Size (ASQS) will be initialized**
- **Admin Completion Queue Size (ACQS) will be initialized**
- **Controller will be enabled**

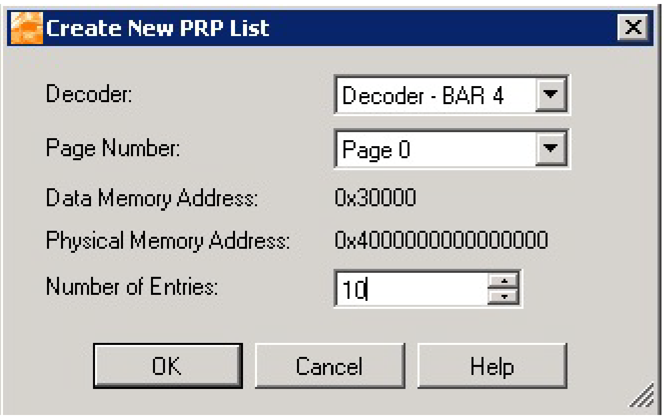
To know more about initializing admin queues, refer to the topic "Step 5 - Initialize the Admin Submission and Completion Queues" on page 187.

Field	Description
Submission Queue Settings - This group box provides options to configure settings for the initialization of the Admin submission queue.	
Decoder	Displays the list of Exerciser decoders that you configured for NVMe using the Decoder page of the GUI. From this list, select a decoder that you want to use for the Admin submission queue. It is recommended that you note down the decoder used for the queue in the Decoder mapping sheet to avoid overlapping or overwriting to the data memory areas.
Page Number	Displays the page offsets within the selected decoder. Exerciser automatically detects the number of pages supported by the selected decoder. From these pages, select the page that you want to allocate to the Admin submission queue. It is recommended that you note down the decoder's page used for the queue in the Decoder mapping sheet to avoid overlapping or overwriting to the data memory areas.
Data Memory Address	Based on the decoder and page number that you selected for the queue, Exerciser automatically displays the Data Memory address applicable for the queue. To display this address, it uses the mapping between the data memory address and physical memory address that you specified while configuring the decoder.
Physical Memory Address	Based on the decoder and page number that you selected for the queue, Exerciser automatically displays the Physical Memory address applicable for the queue. To display this address, it uses the mapping between the data memory address and physical memory address that you specified while configuring the decoder.

Field	Description
Queue Size	Select the size for the Admin submission queue. The queue is created with the N+1 size where N is the specified size of queue. For instance, if you specified 15 as the size of the queue, then the queue can take commands from 0 to 15 making the actual size of the queue as 16.
Completion Queue Settings - This group box provides options to configure settings for the initialization of the Admin completion queue.	
Decoder	Displays the list of Exerciser decoders that you configured for NVMe using the Decoder page of the GUI. From this list, select a decoder that you want to use for the Admin completion queue. It is recommended that you note down the decoder used for the queue in the Decoder mapping sheet to avoid overlapping or overwriting to the data memory areas.
Page Number	Displays the number of pages for the selected decoder. Exerciser automatically detects the number of pages supported by the selected decoder. From these pages, select the page that you want to allocate to the Admin submission queue. It is recommended that you note down the decoder's page used for the queue in the Decoder mapping sheet to avoid overlapping or overwriting to the data memory areas.
Data Memory Address	Based on the decoder and page number that you selected for the queue, Exerciser automatically displays the Data Memory address applicable for the queue. To display this address, it uses the mapping between the data memory address and physical memory address that you specified while configuring the decoder.
Physical Memory Address	Based on the decoder and page number that you selected for the queue, Exerciser automatically displays the Physical Memory address applicable for the queue. To display this address, it uses the mapping between the data memory address and physical memory address that you specified while configuring the decoder.
Queue Size	Select the size for the Admin completion queue. The queue is created with the N+1 size where N is the specified size of queue. For instance, if you specified 15 as the size of the queue, then the queue can take completions from 0 to 15 making the actual size of the queue as 16.

Create New PRP List dialog box

You use this dialog box to create a new PRP list that can be used for data transfers to and from system memory in commands that Exerciser submits to an NVMe DUT. This dialog box is accessible only when Exerciser is emulating an NVMe root complex.



Create New PRP List

Decoder: Decoder - BAR 4

Page Number: Page 0

Data Memory Address: 0x30000

Physical Memory Address: 0x4000000000000000

Number of Entries: 10

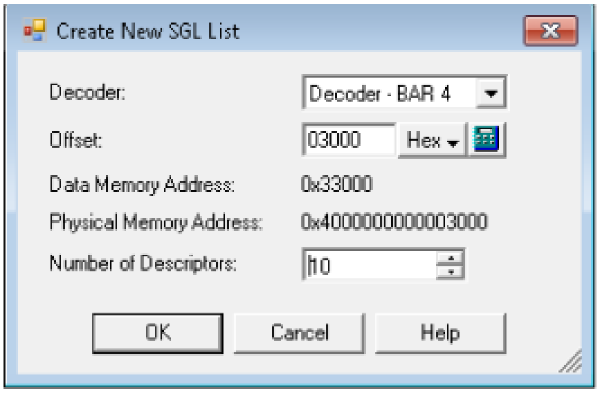
To access this dialog: Click the ➕ button in the **PRP** tab of the **NVM Express** page.

To know more about PRP lists and PRP entries and how to use these in NVMe commands sent to DUT, refer to the topic "Creating and Using a PRP List" on page 194.

Field	Description
Decoder	Displays the list of decoders that you configured using the Decoder Settings page. Select a Decoder that you want Exerciser to use for the PRP list. The physical memory address and internal data memory address mapping of the selected decoder is then used for the PRP list. It is recommended that you note down the decoder used for the PRP list in the Decoder mapping sheet to avoid overlapping or overwriting to the data memory areas.
Page Number	Displays the page offsets within the selected decoder. Exerciser automatically detects the number of pages supported by the selected decoder. From these pages, select the page offset for the PRP list. It is recommended that you note down the decoder's page used for the PRP list in the Decoder mapping sheet to avoid overlapping or overwriting to the data memory areas.
Data Memory Address	Based on the decoder and page number that you selected for the PRP list, Exerciser automatically displays the Data Memory address applicable for the PRP list. To display this address, it uses the mapping between the data memory address and physical memory address that you specified while configuring the decoder.
Physical Memory Address	Based on the decoder and page number that you selected for the PRP list, Exerciser automatically displays the Physical Memory address applicable for the PRP list. To display this address, it uses the mapping between the data memory address and physical memory address that you specified while configuring the decoder.
Number of Entries	Specify the number of PRP entries that the list should contain.

Create New SGL List dialog box

You use this dialog box to create a new SGL list that can be used for data transfers to and from system memory in commands that Exerciser submits to an NVMe DUT. This dialog box is accessible only when Exerciser is emulating an NVMe root complex.



Create New SGL List

Decoder: Decoder - BAR 4

Offset: 03000 Hex

Data Memory Address: 0x33000

Physical Memory Address: 0x4000000000003000

Number of Descriptors: 10

To access this dialog: Click the button in the **SGL** tab of the **NVM Express** page.

To know more about SGL lists, SGL Segment and Descriptor and how to use these in NVMe commands sent to DUT, refer to the topic "Creating and Using a SGL List" on page 198.

Field	Description
Decoder	Displays the list of decoders that you configured using the Decoder Settings page. Select a Decoder that you want Exerciser to use for the SGL list. The physical memory address and internal data memory address mapping of the selected decoder is then used for the SGL list. It is recommended that you note down the decoder used for the SGL list in the Decoder mapping sheet to avoid overlapping or overwriting to the data memory areas.
Offset	Select an offset within the selected decoder that you want to use with the descriptor. It is recommended that you note down the decoder's page used for the SGL list in the Decoder mapping sheet to avoid overlapping or overwriting to the data memory areas.
Data Memory Address	Offset that you selected for the SGL list, Exerciser automatically displays the Data Memory address applicable for the SGL list. To display this address, it uses the mapping between the data memory address and physical memory address that you specified while configuring the decoder.
Physical Memory Address	Offset that you selected for the SGL list, Exerciser automatically displays the Physical Memory address applicable for the SGL list. To display this address, it uses the mapping between the data memory address and physical memory address that you specified while configuring the decoder.
Number of Descriptors	Select the number of descriptors that the first segment of the list should contain.

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.
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.

Component	Description
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**.
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.

Component	Description
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.
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.

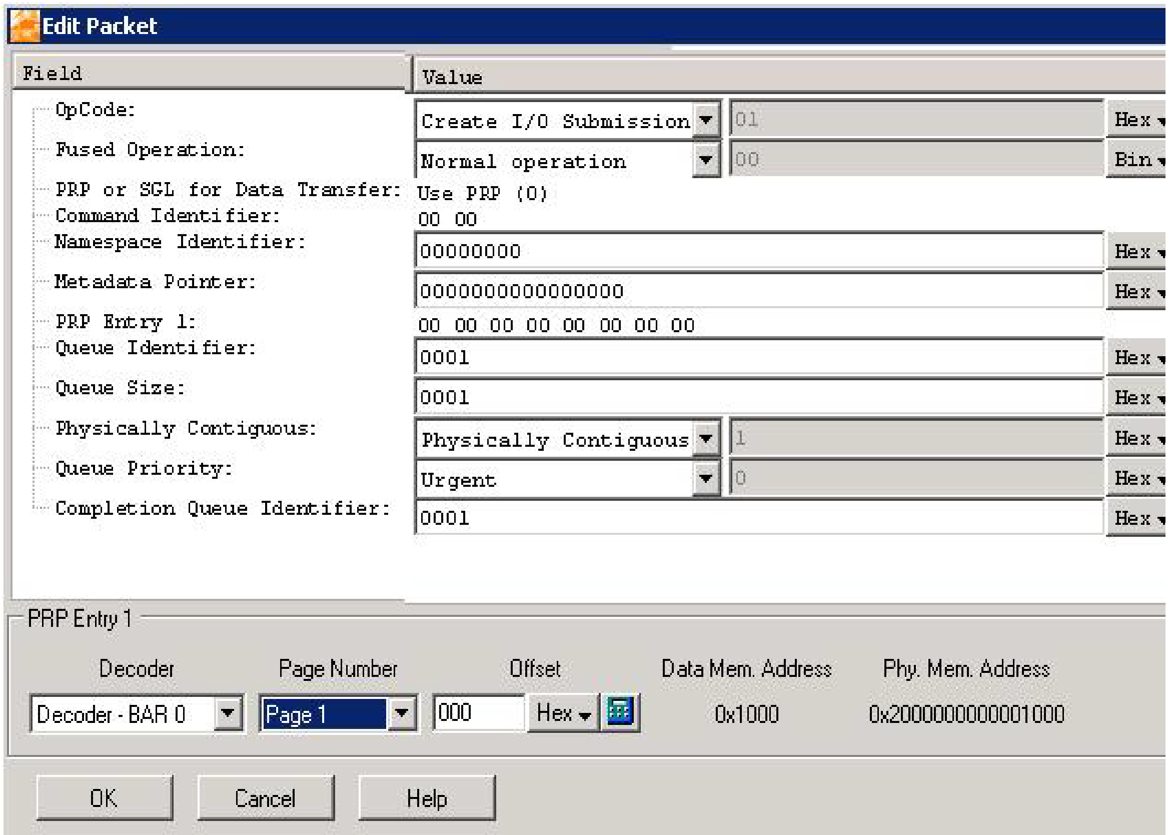
Component	Description
Field	The Field column contains the names of the fields applicable for a PCIe packet. The number of fields displayed in this column may differ for different types of packets. The fields that are displayed also depend on whether the PCIe packet is to be sent on an MR enabled or a 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.
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.

Component	Description
Write Data To	This section is only applicable for a Read request packet to enable you to specify what needs to be done to the completion data received for such a packet. This section has the following options for the completion data of a Read request packet: ▪ **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** - Select this option if you want the completion data received for the Read request packet to be checked algorithmically by the Payload Checker feature of Protocol Exerciser. This option is available only when you are sending packets as a block transfer. Refer to the topic Configuring Algorithmic Payload Generation and Verification Settings to know more about checking a payload algorithmically. ▪ **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 if the **Data Memory** option is selected. If the **Data Comparator** option is selected, it enables you to specify the length of the expected payload to be generated algorithmically for verification of the completion data. The maximum value permissible in this text box is 1024 DWs for sending a single packet and 2097151 DWs for sending block transfer.
Read Data From	This section is only applicable for a Write request packet to enable you to specify how the payload of such a packet should be generated. This section has the following options for generating the payload for a Write request: ▪ **Single Packet Memory**: Select this option if you want the payload data for the single packet to be read 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**: Select this option if you want the payload data for the Write request packet to be generated algorithmically by the Payload generator feature of Protocol Exerciser. This option is available only when you are sending packets as a block transfer. Refer to the topic Configuring Algorithmic Payload Generation and Verification Settings to know more about generating a payload algorithmically. ▪ **Data Memory**: Select this option if you want the payload data for the Write request packet to be read from the 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 if the **Data Memory** option is selected. It also enables you to specify the length of the payload to be generated algorithmically for the Write request packet if the **Data Generator** option is selected.
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.

Edit Packet dialog box (for NVMe Commands)

You use this dialog box to configure the settings for an NVMe command to be sent as a PCIe packet to DUT. This dialog box is accessible only when Exerciser is emulating an NVMe root complex.

To access this dialog: Double-click the NVMe command (packet) to be edited from the **Submission and Completion Queues** tab of the **NVM Express** page.



Field	Value		
OpCode:	Create I/O Submission	01	Hex
Fused Operation:	Normal operation	00	Bin
PRP or SGL for Data Transfer:	Use PRP (0)		
Command Identifier:	00 00		
Namespace Identifier:	00000000		Hex
Metadata Pointer:	0000000000000000		Hex
PRP Entry 1:	00 00 00 00 00 00 00 00		
Queue Identifier:	0001		Hex
Queue Size:	0001		Hex
Physically Contiguous:	Physically Contiguous	1	Hex
Queue Priority:	Urgent	0	Hex
Completion Queue Identifier:	0001		Hex

PRP Entry 1

Decoder	Page Number	Offset	Data Mem. Address	Phy. Mem. Address
Decoder - BAR 0	Page 1	000 Hex	0x1000	0x2000000000001000

Field	Description
Field	The Field column contains the names of the fields applicable for an NVMe command. Fields displayed for a command are as per the NVMe specifications. The number of fields displayed for a command may differ for different types of commands. For instance, for a Create I/O Submission Queue command, the Priority field is displayed.
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.
Length	The Length column displays the maximum length of a field in bits, bytes, or DWords.
PRP Entry - This group box displays fields to allow you to set decoder and the page offset within the selected decoder to be used for the PRP entry associated with the command. It also provides fields to allow you to use PRP lists in a command.	

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.
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.
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.
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.

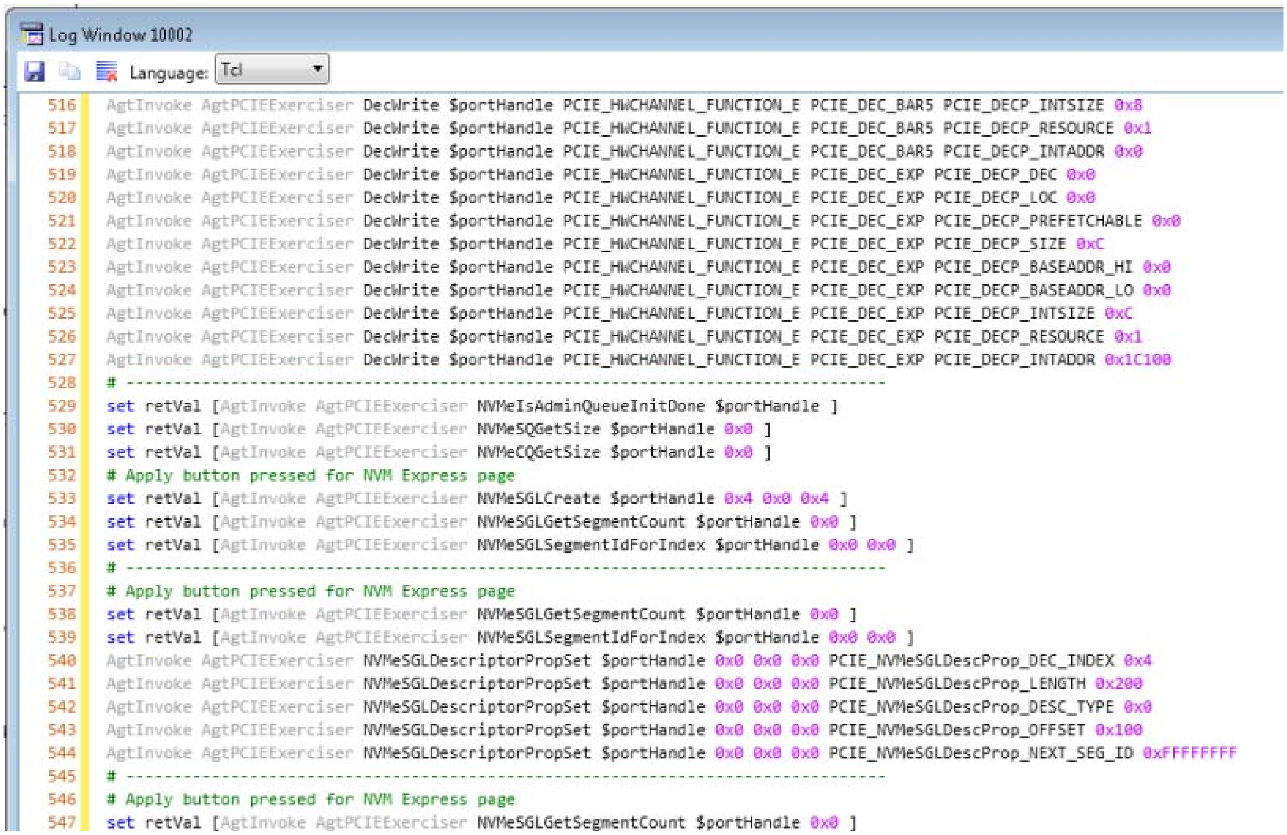
Log Window Dialog Box

This dialog box logs all API calls initiated through the Exerciser GUI. Note that the following API calls are NOT logged:

- Calls initiated through PTC, LTSSM, or NVMe Conformance test suite.
- Calls initiated through GUI updates due to events from embedded/host.

Click the **Show Log Window** option from the **Action** menu to access this dialog box. You can keep this dialog box open or close it while working on the Exerciser GUI. Closing the dialog box does not impact API logging.

The screen below shows API calls logged in the Log Window dialog box.



Language: Tcl

```
AgtInvoke AgtPCIEExerciser DecWrite $portHandle PCIE_HWCHANNEL_FUNCTION_E PCIE_DEC_BAR5 PCIE_DECP_INTSIZE 0x8
AgtInvoke AgtPCIEExerciser DecWrite $portHandle PCIE_HWCHANNEL_FUNCTION_E PCIE_DEC_BAR5 PCIE_DECP_RESOURCE 0x1
AgtInvoke AgtPCIEExerciser DecWrite $portHandle PCIE_HWCHANNEL_FUNCTION_E PCIE_DEC_BAR5 PCIE_DECP_INTADDR 0x0
AgtInvoke AgtPCIEExerciser DecWrite $portHandle PCIE_HWCHANNEL_FUNCTION_E PCIE_DEC_EXP PCIE_DECP_DEC 0x0
AgtInvoke AgtPCIEExerciser DecWrite $portHandle PCIE_HWCHANNEL_FUNCTION_E PCIE_DEC_EXP PCIE_DECP_LOC 0x0
AgtInvoke AgtPCIEExerciser DecWrite $portHandle PCIE_HWCHANNEL_FUNCTION_E PCIE_DEC_EXP PCIE_DECP_PREFETCHABLE 0x0
AgtInvoke AgtPCIEExerciser DecWrite $portHandle PCIE_HWCHANNEL_FUNCTION_E PCIE_DEC_EXP PCIE_DECP_SIZE 0xC
AgtInvoke AgtPCIEExerciser DecWrite $portHandle PCIE_HWCHANNEL_FUNCTION_E PCIE_DEC_EXP PCIE_DECP_BASEADDR_HI 0x0
AgtInvoke AgtPCIEExerciser DecWrite $portHandle PCIE_HWCHANNEL_FUNCTION_E PCIE_DEC_EXP PCIE_DECP_BASEADDR_LO 0x0
AgtInvoke AgtPCIEExerciser DecWrite $portHandle PCIE_HWCHANNEL_FUNCTION_E PCIE_DEC_EXP PCIE_DECP_INTSIZE 0xC
AgtInvoke AgtPCIEExerciser DecWrite $portHandle PCIE_HWCHANNEL_FUNCTION_E PCIE_DEC_EXP PCIE_DECP_RESOURCE 0x1
AgtInvoke AgtPCIEExerciser DecWrite $portHandle PCIE_HWCHANNEL_FUNCTION_E PCIE_DEC_EXP PCIE_DECP_INTADDR 0x1C100
# --------------------------------------------------------------------------------
set retVal [AgtInvoke AgtPCIEExerciser NVMeIsAdminQueueInitDone $portHandle ]
set retVal [AgtInvoke AgtPCIEExerciser NVMeSQGetSize $portHandle 0x0 ]
set retVal [AgtInvoke AgtPCIEExerciser NVMeCQGetSize $portHandle 0x0 ]
# Apply button pressed for NVM Express page
set retVal [AgtInvoke AgtPCIEExerciser NVMeSGLCreate $portHandle 0x4 0x0 0x4 ]
set retVal [AgtInvoke AgtPCIEExerciser NVMeSGLGetSegmentCount $portHandle 0x0 ]
set retVal [AgtInvoke AgtPCIEExerciser NVMeSGLSegmentIdForIndex $portHandle 0x0 0x0 ]
# --------------------------------------------------------------------------------
# Apply button pressed for NVM Express page
set retVal [AgtInvoke AgtPCIEExerciser NVMeSGLGetSegmentCount $portHandle 0x0 ]
set retVal [AgtInvoke AgtPCIEExerciser NVMeSGLSegmentIdForIndex $portHandle 0x0 0x0 ]
AgtInvoke AgtPCIEExerciser NVMeSGLDescriptorPropSet $portHandle 0x0 0x0 0x0 PCIE_NVMeSGLDescProp_DEC_INDEX 0x4
AgtInvoke AgtPCIEExerciser NVMeSGLDescriptorPropSet $portHandle 0x0 0x0 0x0 PCIE_NVMeSGLDescProp_LENGTH 0x200
AgtInvoke AgtPCIEExerciser NVMeSGLDescriptorPropSet $portHandle 0x0 0x0 0x0 PCIE_NVMeSGLDescProp_DESC_TYPE 0x0
AgtInvoke AgtPCIEExerciser NVMeSGLDescriptorPropSet $portHandle 0x0 0x0 0x0 PCIE_NVMeSGLDescProp_OFFSET 0x100
AgtInvoke AgtPCIEExerciser NVMeSGLDescriptorPropSet $portHandle 0x0 0x0 0x0 PCIE_NVMeSGLDescProp_NEXT_SEG_ID 0xFFFFFFFF
# --------------------------------------------------------------------------------
# Apply button pressed for NVM Express page
set retVal [AgtInvoke AgtPCIEExerciser NVMeSGLGetSegmentCount $portHandle 0x0 ]
```

You can view and save these API logs in the following formats:

- Tcl
- C#
- C++
- Python
- Plain Text

You can switch between these formats using the options available in the dialog box.

If you saved an API log in TCL, C#, C++, or Python, then such files can be executed later to execute the API calls initiated by the GUI. If you want to introduce delays or other checks in such API calls execution, then you need to edit the saved script file.

To set-up Python API for Exerciser, refer to the ReadMe document located in C:\ProgramFiles (x86)\Keysight\SPT\PCIEExerciserGen3\8.74 Release\Samples\Python

To run an API log file saved in C# format

1 Create a new C# project in Visual Studio.

2 Add the saved API log file to the created project.

3 Copy the files **ModuleManager.cs** and **TestSession.cs** from *"C:\Program Files (x86)\Keysight\SPT\PCIEExerciserGen3\8.74 Release\Samples\CSharp\SendTLP"* and add to the project.

4 Add Reference to Interop.AgtPCIE, Interop.AgtResourceManagerLib, Interop.AgtSessionPlatform and Interop.AgtSptControl files in the project. These files are available at:

"C:\Program Files (x86)\Keysight\SPT\PCIEExerciserGen3\8.74 Release\bin" location.

5 Compile the solution to verify that it does not contain any errors.

6 Press F5 to run the project.

To run an API log file saved in the C++ format

1 Create a new C++ project in Visual Studio.

2 Add the saved API log file to the created project.

3 Copy the files **PCIeExerciser.h** and **PCIeExerciser.cpp** from "*C:\Program Files (x86)\Keysight\SPT\PCIEExerciserGen3\8.74 Release\Samples\dcom\SendTlp"* and add to the project.

4 Go to project Properties > Configuration Properties > C/C++ > General and modify "Additional Include Directories" property to include "C:\Program Files (x86)\Keysight\SPT\PCIEExerciserGen3\8.74 Release\bin".

5 Compile the solution to verify that it does not contain any errors.

6 Press F5 to run the project.

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.
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.

Component	Description
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.
NVM Express	These settings are applicable for the NVMe submission and completion queue records displayed in the Submission and Completion Queues tab. These settings are relevant when Exerciser works in the NVMe mode. 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 submission / completion is in a collapsed form. The Available fields list displays all the fields that you can add for submission/completion records. The Displayed fields list displays all the fields that are displayed in the collapsed view in the Submission and Completion Queues tab. 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 submission / completion is in an expanded form. The Available fields list displays all the fields that are displayed for submission/completion records. The Hidden fields list displays all the fields that are not displayed in the expanded form. 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 types of submission/completion records displayed in the Submission and Completion Queues tab. This helps you to easily recognize different types of submission/completion records.

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.

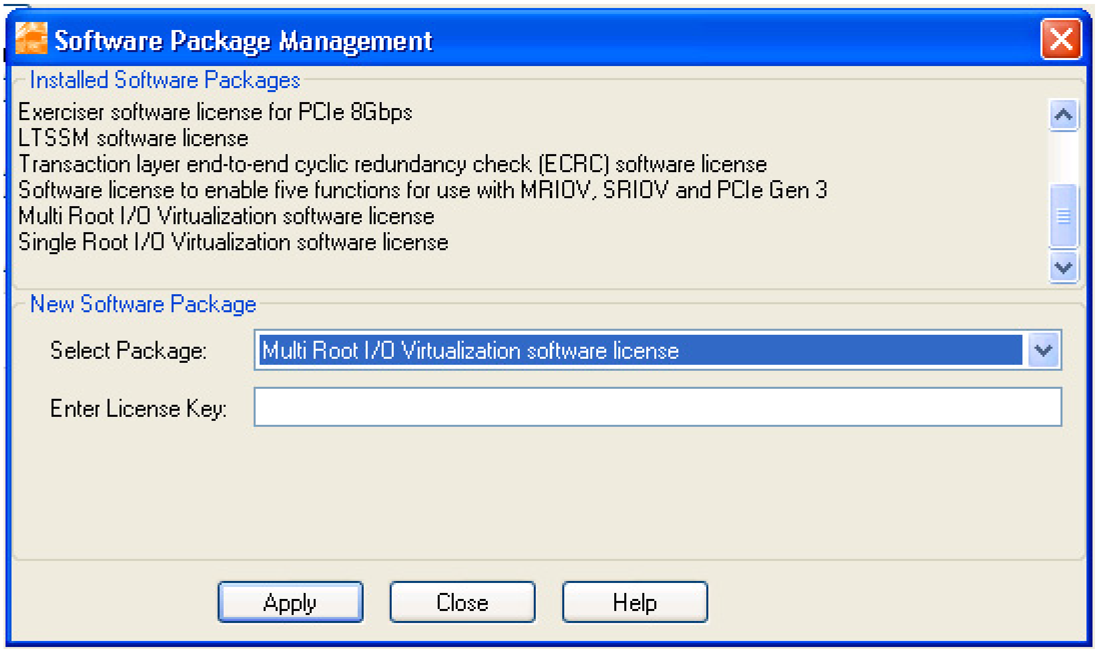
Select type of connection dialog box

To access this dialog: Click **Start > Programs > Keysight 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**.



Software Package Management

Installed Software Packages

Exerciser software license for PCIe 8Gbps
LTSSM software license
Transaction layer end-to-end cyclic redundancy check (ECRC) software license
Software license to enable five functions for use with MRIOV, SRIOV and PCIe Gen 3
Multi Root I/O Virtualization software license
Single Root I/O Virtualization software license

New Software Package

Select Package: Multi Root I/O Virtualization software license

Enter License Key:

Component	Description
Installed Software Packages	The Installed Software Packages section displays a list of currently installed software packages.
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 x2 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 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. ■ Transaction layer end-to-end cyclic redundancy check (ECRC) software license - With this license, you get ECRC support. Using this license, you can also perform ECRC error checking in the received TLPs and ECRC error insertion in the transmitted TLPs. ■ Protocol Test Card 3.0 software license ■ Multi Root I/O Virtualization software license (U4305A-025) - This license adds MRIOV capabilities to Exerciser at Gen1, Gen2, and Gen3 speed. Requires Exerciser software license for PCIe 8Gbps. ■ Single Root I/O Virtualization software license (U4305A-026) - This license adds SRIOV capabilities to Exerciser at Gen1, Gen2, and Gen3 speed. Requires Exerciser software license for PCIe 8Gbps. ■ Software license to enable five functions for use with MRIOV, SRIOV, and PCIe Gen3 - 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.
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 Keysight 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.
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 Generator**: Select this option if you want Exerciser to use the payload generator feature to generate or check the payload of packets algorithmically. On selecting this option, the completion data for the Read packets received from DUT is generated algorithmically using the payload generator. Also, the Write packets received from DUT are compared with the expected payload generated by the payload checker. This option therefore eliminates the need for configuring or accessing the data memory of Exerciser for payloads of the Read/Write requests. Refer to the topic Configuring Algorithmic Payload Generation and Verification Settings to know more about how to configure Exerciser to generate or check payloads algorithmically.
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.

DUT Config Space Function

To access this function: Click **DUT Config Space** in the Navigation pane. You use the DUT Config Space page to:

- initiate the DUT configuration space scan from the Overview tab.
- read the DUT capabilities results retrieved by the scan in various tabs of this page.
- edit the DUT capabilities for which the edit operation is allowed.

To know more, refer to the topic Viewing and Editing the Configuration Space of a DUT.

NOTE

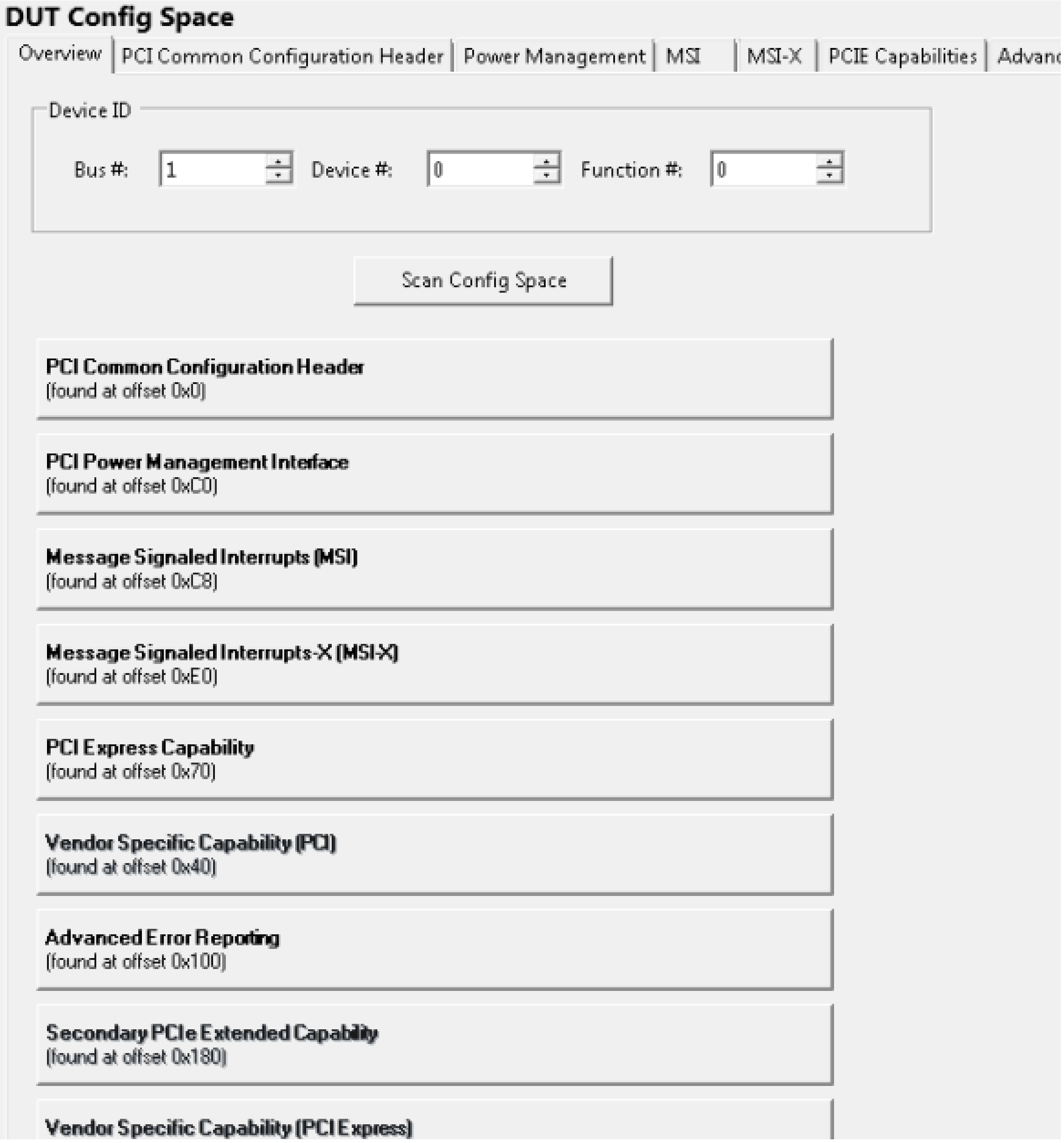
In the left navigation pane of the GUI, the DUT Config Space icon is disabled if the Exerciser is configured to act as an endpoint, that is the Session type is configured as "To upstream".

The fields in the DUT Config Space page are disabled and a warning message is displayed at the top of the page if:

- the Exerciser is not connected to DUT

- the PCIe link between the Exerciser and DUT is not active.

To enable fields and initiate scan, ensure that a PCIe link is active between the Exerciser and DUT.



Bus #: 1 Device #: 0 Function #: 0

PCI Common Configuration Header
(found at offset 0x0)

PCI Power Management Interface
(found at offset 0xC0)

Message Signaled Interrupts (MSI)
(found at offset 0xC8)

Message Signaled Interrupts-X (MSI-X)
(found at offset 0xE0)

PCI Express Capability
(found at offset 0x70)

Vendor Specific Capability (PCI)
(found at offset 0x40)

Advanced Error Reporting
(found at offset 0x100)

Secondary PCIe Extended Capability
(found at offset 0x180)

Vendor Specific Capability (PCI Express)

Field	Description
Device ID - This group box allows you to configure the DUT settings for which you want to perform configuration space scan.	
Bus #	Bus number of the DUT.
Device #	Device number of the DUT.
Function #	DUT's function number for which configuration scan is to be performed.
Scan Config Space	Click this button to initiate the configuration space scan of the DUT for which details are specified in the tab.

Field	Description
PCI Common Configuration Header	After the configuration space scan, the offset value at which the PCI Common Configuration Header capabilities are found is displayed. Click this button to display the PCI Common Configuration Header tab in which all the applicable capability registers of the DUT are displayed. In this tab, you can edit the PCI Common Configuration Header capabilities of DUT.
Power Management	After the configuration space scan, the offset value at which the DUT's power management capabilities are found is displayed. Click this button to display the Power Management tab in which all the applicable power management capability registers of the DUT are displayed. In this tab, you can edit the Power Management capabilities of DUT.
MSI	After the configuration space scan, the offset value at which the DUT's MSI capabilities are found is displayed. Click this button to display the MSI tab in which all the applicable MSI capability registers of the DUT are displayed. In this tab, you can edit the MSI capabilities of DUT.
PCIe Capabilities	After the configuration space scan, the offset value at which the PCIe capabilities of DUT are found is displayed. Click this button to display the PCIe Capabilities tab in which all the applicable PCIe capability registers of the DUT are displayed. In this tab, you can edit the PCIe capabilities of the DUT.
Virtual Channel	After the configuration space scan, the offset value at which the DUT's virtual channel capabilities are found is displayed.
Advanced Error	After the configuration space scan, the offset value at which the DUT's Advanced Error capabilities are found is displayed. Click this button to display the Advance Error Capabilities tab in which all the applicable capability registers of the DUT are displayed. In this tab, you can edit the Advance Error capabilities of the DUT
MSI-X	After the configuration space scan, the offset value at which the DUT's MSI-X capabilities are found is displayed. Click this button to display the MSI-X tab in which all the applicable MSI-X capability registers of the DUT are displayed. In this tab, you can edit the MSI-X capabilities of DUT.
⟳ and ⤓ buttons	In the PCI Common Configuration Header, MSI, and MSI-X tabs, a ⟳ and ⤓ button is displayed with each capability register. The ⤓ button allows you to write the changes you made to a particular DUT register with which the button is displayed. The ⟳ button allows you to read the current values for a register from DUT.
Apply	Clicking the **Apply** button writes all the changes that you made to various DUT registers displayed in a tab of DUT Config Space.

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.

To know more, refer to the topic "Sending an Arbitrary DLLP Or Inserting a Wrong CRC in the DLLP.

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.
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.
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.
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 - Algorithmic Payload tab

To access this function: Click **General Settings** in the Navigation pane. Then, click the **Algorithmic Payload** tab. You use this tab to configure the settings for algorithmically generating and checking payload for packets using the Payload Generator and Checker feature of Exerciser. This eliminates the need for accessing the data memory of Exerciser for reading/writing the payloads for packets.

Refer to the topic Configuring Algorithmic Payload Generation and Verification Settings to know how to generate and check the payloads of packets algorithmically.

NOTE

The algorithmic payload generation and checker feature is supported only for physical functions of Exerciser. For virtual functions, this feature is not available.

Field	Description
Clear Status	Click **Clear Status** to reset the status of the various fields displayed on the page.
Refresh	Click **Refresh** to refresh the data displayed in various fields on the Algorithmic Payload tab page.
Function <Number>	You set the algorithmic payload generation and verification settings separately for each physical function supported by Exerciser. The fields are therefore grouped on the basis of functions on this page.
Data Generation Prefix	The prefix that the Payload Generator and Checker use to algorithmically generate and verify the payload of packets. This prefix becomes the first 11 bits of the generated payload of a packet that Exerciser sends to DUT. It is also used to generate the expected payload with which the actual payload of an incoming packet is compared for verification. By default, the prefix is specified as follows for the physical functions of Exerciser. However, if needed, you can change the prefix value. ▪ Function A: AA000000 ▪ Function B: BB000000 ▪ Function C: CC000000 ▪ Function D: DD000000 ▪ Function E: EE000000
Error Status	Displays whether or not an error occurred while algorithmically verifying the payloads of the packets belonging to the function. The field displays: ▪ **No Error i**f the payload of an incoming packet matches the expected payload that the Payload checker generated algorithmically. ▪ **Error occurred** if the payload of an incoming packet does not match the expected payload that the Payload checker generated algorithmically. To generate the expected payload for verification of the payloads of incoming packets, the Payload checker uses the Data generation prefix that you specified for the function and the PCI address of the incoming TLPs.
Error Packet Header	Displays the header of the packet for which the Payload checker found a mismatch when compared to the algorithmically generated payload of the packet. If no mismatch is found, this field does not display any data.
Error DW Offset	Displays the offset of the DW in which the payload data mismatch was found. For example, if the mismatch occurred due to the second DW, then Error DW Offset is displayed as 1. If the mismatch was found in the first DW, then Error DW Offset is displayed as 0. If there are multiple DWs where the mismatch is found, then the offset of only the first DW in which the mismatch is found is displayed.
Error Byte Enable	Displays the Byte Enable status of the DW responsible for the payload data mismatch. If there are multiple DWs where the mismatch is found, then the Byte Enable status of only the first DW in which the mismatch is found is displayed.
Error Expected Data	Displays the expected payload of the incoming packet for which the Payload checker found a mismatch when comparing this expected payload with the actual payload of packet. The expected payload is generated algorithmically.
Error Actual Data	Displays the payload of the packet for which the Payload checker found a mismatch when compared to the expected payload of the packet. The expected payload is generated algorithmically.

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 settings" on page 58 to know more about the equalization implementation in Protocol Exerciser.

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	
Automatic Coefficient Request to DUT	Select this checkbox to ensure that Exerciser automatically computes the preset/coefficient values and makes coefficient requests to DUT using these values in Phase 2/3 of equalization. When you select this checkbox, the fields provided in the Request to DUT and Exerciser group boxes for manually specifying the coefficient values are disabled. Exerciser then auto computes the coefficient values that Exerciser advertises during the link training and sends as request to DUT. If you do not select this checkbox, then Exerciser uses the coefficient values that you specify in the Request to DUT group box of the Equalization Settings tab. The final preset/coefficient values requested from Exerciser to DUT are displayed in the Preset/coefficient request from Exerciser section of the Equalization Status tab in the Hardware Status pane.
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.
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. This field is disabled if you configure auto computation and automatic request of co-efficient values to DUT by selecting the Automatic Coefficient Request to DUT checkbox.
Tx pre cursor coefficients	Options to set the Tx coefficient –C1 value from the Tx Equalization Finite Impulse Response Filter. This field is disabled if you configure auto computation of co-efficient values by selecting the Automatic Coefficient Request to DUT checkbox.
Tx data cursor coefficients	Options to set the Tx coefficient C0 value from the Tx Equalization Finite Impulse Response Filter. This field is disabled if you configure auto computation of co-efficient values by selecting the Automatic Coefficient Request to DUT checkbox.
Tx post cursor coefficients	Options to set the Tx coefficient +C1 value from the Tx Equalization Finite Impulse Response Filter. This field is disabled if you configure auto computation of co-efficient values by selecting the Automatic Coefficient Request to DUT checkbox.
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.

Field	Description
FS	Options to set the value for Full Swing (FS) resolution for de-emphasis. Protocol Exerciser advertises this value in Phase 1 of Equalization. This field is disabled if you select the Automatic (As per DUT request) checkbox in the Transceiver Settings tab. On selecting this checkbox, the FS value is automatically set to 24 and manual editing of the LF value is disabled.
LF	Options to set the value for Low Frequency (LF) resolution for de-emphasis. This field is disabled if you select the Automatic (As per DUT request) checkbox in the Transceiver Settings tab. On selecting this checkbox, the LF value is automatically set to 8 and manual editing of the LF value is disabled.
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. This field is disabled if Exerciser is configured as a DSC (session type set to ToUpstream in the Link Settings tab).
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. This field is disabled if you configure auto computation of co-efficient values by selecting the Automatic Coefficient Request to DUT checkbox.
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. This field is disabled if you configure auto computation of co-efficient values by selecting the Automatic Coefficient Request to DUT checkbox.
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. This field is disabled if you configure auto computation of co-efficient values by selecting the Automatic Coefficient Request to DUT checkbox.
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.
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
Application	Select the mode in which you want to use Exerciser. ▪ **General** - Use this option to set Exerciser in the general PCIe mode (non NVMe usage). ▪ **NVMe** - Use this option to set Exerciser to work in the NVMe mode and emulate an NVMe root complex. On selecting NVMe, the MRIOV and SRIOV protocol support is disabled. Only PCIe is supported and displayed enabled and selected in the Protocol field. Also, the Session Type is automatically set to "To Downstream" for Exerciser to act as a root complex in the NVMe mode.
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 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 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. ■ To enable the SRIOV capabilities for Exerciser, you need the appropriate SRIOV software license without which the SRIOV option is disabled in the Link settings tab. The U4305A-026 software license provides SRIOV capabilities for Exerciser at Gen1, Gen2, and Gen3 speed. ▪ **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. ■ To enable the MRIOV capabilities for Exerciser, you need the appropriate MRIOV software license without which the MRIOV option is disabled in the Link settings tab. The U4305A-025 software license provides MRIOV capabilities for Exerciser at Gen1, Gen2, and Gen3 speed. 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.
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.
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 66 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.
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 73.

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. ■ ***Note***: By default, Exerciser automatically initiates link recovery if it does not receive any FC update packets for 200us when in L0 or L0s state. However, in certain L0s testing scenarios, you may not want Exerciser to do this. In such situations, you can deselect the **Initiate link recovery if no FC update is received for 200us in L0, L0s state** checkbox in the Virtual Channel function to ensure that the link recovery is initiated as per the exit timer value that you specified in the Power Management tab. ▪ **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.
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.

Field	Description
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.

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.
Automatic (As per DUT request)	Select this checkbox to ensure that Exerciser automatically sets its transmitter settings based on the preset/coefficient requests received from DUT. On selecting this checkbox, ▪ the fields provided for manually specifying the values for adjusting Exerciser's Transmitter settings are disabled in this tab. ▪ the FS and LF fields are disabled in the Equalization Settings tab and the FS and LF values for Exerciser are automatically set to 24 and 8 respectively. ▪ Exerciser applies the De-emphasis, Pre-shoot & Boost levels requested by DUT on its transmitted signals. If you do not select this checkbox, Exerciser applies the user-specified De-emphasis, Pre-shoot & Boost levels on its transmitted signals.
Tx output voltage	This setting controls the maximum Vout before any pre-emphasis or de-emphasis (Vd). The default value is 900 mV. This field is disabled if you configure automatic setting of transmitter settings by selecting the **Automatic (As per DUT request)** checkbox in this tab.
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. This field is disabled if you configure automatic setting of transmitter settings by selecting the **Automatic (As per DUT request)** checkbox in this tab.
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. This field is disabled if you configure automatic setting of transmitter settings by selecting the **Automatic (As per DUT request)** checkbox in this tab.
Tx Pre-emphasis Second Post Tap	Second post-tap De-emphasis. This field is disabled if you configure automatic setting of transmitter settings by selecting the **Automatic (As per DUT request)** checkbox in this tab.

Field	Description
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	The Rx Equalizer AC and DC Gain settings impact the Receiver equalization of the Protocol Exerciser. You can use these two listboxes to select the AC and DC gain settings to tune the Exerciser's Rx equalization. Typically Rx DC gain would be 3dB. For setting the Rx AC gain, you can use the following recommendations to set it as per the channel length (in inches): Channel Length Recommended Rx AC Gain 3-7 5dB 8 - 12 6dB 13-16 8dB 17-20 10dB
Use Rx Preset Hint for AC Gain	Select this checkbox to instruct Exerciser to automatically adjust its Rx AC gain using the Rx Preset hint given by DUT. Selecting this checkbox eliminates the need for manually specifying an Rx AC gain value for Exerciser to tune its Rx AC gain and therefore disables the Rx Equalizer AC Gain listbox in this tab. The Rx preset hints are provided by USC (root complex) to DSC (endpoint). Therefore, this checkbox is enabled only when Exerciser is configured as a DSC. If Exerciser is configured as a USC, this checkbox is disabled and you have to set its Rx AC gain manually using the Rx Equalizer AC Gain listbox. The Rx preset hint received from DUT is displayed in the Device Under Test section of the Equalization Status tab of the Hardware Status pane. It is displayed as NA when Exerciser is configured as USC.

Refer to the topic “Configuring Transceiver Settings" on page 64 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.
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.
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.

NVM Express Function - Submission and Completion Queues

To access this function: Click **NVMe** in the Navigation pane. Then, select the **Submission and Completion Queues** tab.

You use this tab to define and add NVMe Admin and I/O commands to submission queues for processing by NVMe DUT. You can also view the completion status of these commands in this tab.

To know more, refer to the topics Creating, Deleting, and Viewing Submission and Completion Queues and Submitting an NVMe Command and Viewing its Completion Status.

Field	Description
Submission Queues - This section consists of three panes described below.	
Queues	By default, this pane displays only the Admin submission queue. When you create I/O submission queues, the Queues section displays these newly created I/O submission queues as well. Clicking a submission queue selects its mapped completion queue in the bottom pane of the tab.
NVMe commands	This pane displays the NVMe commands (PCIe packets) that you added to the currently selected submission queue for processing by DUT. These commands are added as records and assigned a sequenced Line number. This Line number is used to map the command to its completion in the lower pane of the tab. Clicking the + icon displayed with a command displays its details. Double-clicking the command displays the Edit Packet dialog box to configure the command settings.
Templates	This pane provides a list of templates for NVMe Admin and I/O commands that you can add to the submission queues for processing. If the currently selected queue is the Admin submission queue, then Admin command set is displayed. IF the currently selected queue is an I/O queue, then the NVMe I/O command set is displayed.

Field	Description
Completion Queues - This section consists of two panes as described below.	
Queues	By default, this pane displays only the Admin completion queue. When you create I/O completion queues, the Queues section displays these newly created I/O completion queues as well.
NVMe Completions	This pane displays the NVMe command completions posted by DUT in the selected completion queue. These completions are added as records and assigned a sequenced Line number. Clicking the + icon displayed with a completion displays its details. You can identify a completion for a command using the SQ Identifier and Command Identifier fields in that completion.

NVM Express Function - Controller Registers

To access this function: Click **NVMe** in the Navigation pane. Then, select the **Controller Registers** tab.

You use this tab to view and initialize/edit the NVMe controller capabilities of the DUT.

To know more, refer to the topic Step 5 - Initialize the Admin Submission and Completion Queues.

NOTE

To ensure that NVMe controller capabilities are read and displayed in the **Controller Registers** tab, you first need to scan the DUT configuration space using the **DUT Config Space** page. If you do not scan the DUT configuration space, controller register values are not displayed and a warning message is displayed in this tab.

Field	Description
Interrupt Mechanism	Allows you to set the interrupt mechanism that the NVMe controller should use to generate interrupts. Exerciser can handle the following two interrupt mechanism options. ▪ MSI ▪ MSI-X During DUT config space scan, the DUT's interrupt mechanism capabilities are read and displayed next to the Interrupt Mechanism field. Based on the DUT's interrupt mechanism capabilities found, the interrupt mechanism options are enabled or disabled in this field.
Controller Register Base	Displays the memory address to which you mapped the Base Address Register 0 (BAR 0) using the **PCI Common Configuration Header** tab of the **DUT Config Space** page. This is the location for the NVMe controller registers.
Reset Controller	Resets the queues and registers to their default values and state. On resetting the controller: ▪ all I/O submission and completion queues are deleted. ▪ the Admin submission and completion queue are reset. ▪ all controller registers except the Admin queue registers are reset to their default values. To know more, refer to Resetting the NVMe Controller Capabilities.
Controller Registers	Various controller registers are displayed in this tab. On scanning the DUT config space, the values for these registers are read from DUT and displayed in this tab. You can edit the values of the registers.
⤓ and ↻ buttons	A ⤓ and ↻ button is displayed with each controller register. The ⤓ button allows you to write the changes that you made to a particular controller register with which the button is displayed. The ↻ button allows you to read the current values for a register from DUT.
Apply	Clicking the **Apply** button saves all the changes that you made to various controller registers displayed in the Controller Registers tab.

NVMe Conformance Suite function

To access NVMe Conformance: Click **NVMe Conformance** icon in the Navigation pane of the protocol exerciser GUI.

You use this tab to test the NVMe conformance capabilities of the DUT.

To know more, refer "Testing DUT's NVMe Compliance using NVMe Conformance Suite.

Component	Description
When you select NVMe Conformance icon in the navigation pane of the Exerciser GUI, NVMe Conformance Tests screen appears displaying two tabs. ▪ Setup Tab ▪ Report Tab The descriptions of the above two is given below:	
Setup Tab	Setup tab provides you a set-up platform where you can select initialization script, select tests to see their descriptions, run test cases, see the log details of steps that are being performed during the execution process, and save the log results etc.
Report Tab	Report tab displays the detailed report of the NVMe Conformance tests that you executed. It displays the overall results that include: ▪ test configuration (such as which device was under test, its manufacturer, link width and link speed on which test was performed, software version, date and time of execution of the test cases, etc.) ▪ summary of the test completion (whether passed or failed), and ▪ the test logs (i.e. which test was executed).
Initialization Script Pane: This pane provides you options to choose whether you want to use a default script or a custom script to run a conformance test(s). ▪ Use Default: Selecting this radio button allows you to run the conformance test on the basis of default values predefined by the initialization script. ▪ Use Custom: Selecting this radio button allows you to choose your own custom script defined by you on the basis of your requirement. Once, you select the **Use Custom** radio button, the **Script File** field allows you to click 📂 button and enable you to browse and select the script file. When you use the custom initialization script for running test cases, it will be reflected in the Report tab. It displays a warning message in the report tab stating "Warning: Custom Initialization Script used".	
Message Framing	Clicking this checkbox enables the message framing option that sends an acknowledgment message packet along with test cases packets. This helps you to identify the exact packets with which a test case started executing and where ended.
📂	This button allows you to open the file browser and choose the custom script file you want to use while running conformance test(s).
📄	Clicking this button allows you to view code of the initialization script being used for running conformance test(s).
▶	This button enables you to run the selected test cases. Clicking Start replaces the Start button with the Stop button. This Stop button is visible until the selected test case execution is complete or you click it to replace it with the Start button.
Test Selection Pane: This pane provides a list of conformance tests and NVMe commands that you can select to run to test the DUT's NVMe functionality.	
🗋	Clicking this button enables you to create a new user test case as per your specific requirements.
📋	The Copy button enables you to create a duplicate copy of the existing test case and modify it as per your requirements.
✕	This button lets you to delete the selected test case.
aIe	Clicking this button enables you to rename the existing test case. Once you click this button after selecting a test case, a pop-up appears which asks you to enter the relevant test case name.
📄	When test cases are run, the color of the selected tests changes into green and red after their execution. Green and red color signify the test case execution state as passed or failed respectively. Clicking the **Clear Results** button clears the test results and sets them to the normal state as they were set before execution.

Component	Description
Description Pane: This pane displays the description and sequence of the execution steps. In this pane, there are following two tabs: ▪ Description Tab ▪ Code Tab Description tab displays the description of the selected test case. It displays purpose, procedure and expected results of the conformance test being executed; whereas, the code tab displays the code snippets of the same. The Description tab has the following heads of information for a test: ▪ Purpose: This head states what the test case intends to do. ▪ Procedure: This head states what steps are being performed during test execution. ▪ Expected Results: This head states the results which is expected to appear after the execution of a test case. Few buttons are commonly used in both Description as well as Code tab, whereas few fields differ in both the tabs, as listed below:	
💾	Clicking this button displays the Save dialog box which allows you to save the data displayed in the 'Code' tab of the Description pane in RTF or TXT format.
✂	This button enables you to cut the selected part of the test cases descriptions and code snippets displayed in the 'Descriptions' and 'Code' tab respectively.
⧉	This button lets you to copy the selected part of the test cases descriptions and code snippets displayed in the 'Descriptions' and 'Code' tab respectively.
📋	Clicking this button enables you to paste the cut or copied part of the test cases descriptions and code snippets to the destined location.
↶	This button lets you to undo the last action.
↷	Once you undo any action, the Redo button is highlighted. Clicking this button lets you to bring back to the same action level where you were before performing the undo action.
Code Tab ☰	This button facilitates you with 'auto-complete list' of functions while writing code for self defined test case.
(x..)	Clicking this button displays the set of parameters to be used with a function while writing code for any self defined test case.
Description Tab Format Tool Bar	Format Tool Bar allows you to format the test cases description with the help of buttons available. The Formatting Bar contains font family, font size, text color, bulleted and numbered list, and set alignment buttons. You can change the font style, font size and font color by clicking their respective drop down button and selecting your preferred choice. You can align text to the right, center, left or keep it in justify style with the help of alignment buttons. You can also insert bullets and numbering with the help of Bulleted list and Numbered list buttons respectively.
Log Pane: The Log pane provides you details of the steps that are performed during the execution of the conformance test(s).	
⧉	This button allows you to copy the data from the Log pane to the clipboard and paste it to the destination folder whenever required.
☒	Clicking this button clears the log results being displayed in the log pane.
ab c↵	Clicking this button toggles the log details displayed in the log pane. It serves as a line breaker and proves useful if you are viewing on a small screen device.
Report Tab: This tab enables you to view the consolidated report summary of the NVMe Conformance tests executed.	
💾	This button allows you to save the conformance test report displayed in the Report tab for further use.
🖨	This button allows you to print the conformance test report.
☒	Clicking this button lets you to clear the consolidated test report summary.

NOTE

Buttons lying under Description and Code tabs appear in highlighted mode only when you are performing any task in '**User Tests**' section of the **Test Selection Pane**.

NVMe EndPoint -Settings

To access this function: click **NVMe** in the navigation pane and click the **Settings** tab of the NVM Express Endpoint page.

The NVM Express Endpoint page is displayed only when you select **NVMe** in the Application field and **To Upstream** in the Session type field of the General Settings page.

To know more about Exerciser's NVMe endpoint emulation, refer to the chapter "Emulating an NVMe EndPoint" on page 211.

Field	Description
Interrupt Mechanism	Allows you to set the interrupt mechanism that the NVMe controller should use to generate interrupts. Exerciser can handle the following two interrupt mechanism options: ▪ MSI ▪ MSI-X
NameSpace	You can select the following options for each of the Namespaces using this section. ▪ Select the **Clear data on start** checkbox to clear the namespace data on start. This will erase all data written to the namespace and drive will show as a RAW unformatted drive in the SUT. ▪ Select the **Write MBR while initializing** checkbox to write the Master Boot Record (MBR) to the namespace during start.
Asynchronous Event Completion	You can use the fields available in this section to send an Asynchronous Event Completion to the RC device. Select the values of the Asynchronous Event Type, Asynchronous Event Information and Associated Log Page fields and then click the **Send** button. Asynchronous event completion allows you to retrieve health information about the device that may indicate errors and failures. This capability yields better error handling and preventive measures can be taken to minimize the risk of data loss.Asynchronous event completion allows you to retrieve health information about the device that may indicate errors and failures. This capability yields better error handling and preventive measures can be taken to minimize the risk of data loss.
Controller	This section has two buttons - **Start/Stop** and **Pause/Resume**. ▪ You can use the Start and Stop buttons to start and stop the controller. Stopping the controller resets and sets the controller to the disabled state. ▪ You can use the Pause and Resume buttons to temporarily pause and then resume the controller. The controller will not respond to any commands during the paused state. This is an error insertion feature.You can pause the controller in order to cause errors. When the Pause option is selected, Exerciser stops polling the controller registers and doorbell registers for changes and does not respond to any command issued by the root complex.

NVMe EndPoint - Controller Registers

To access this function: click **NVMe** in the navigation pane and click the **Controller Registers** tab of the NVM Express Endpoint page.

The NVM Express Endpoint page is displayed only when you select **NVMe** in the Application field and **To Upstream** in the Session type field of the General Settings page.

To know more about Exerciser's NVMe endpoint emulation, refer to the chapter "Emulating an NVMe EndPoint" on page 211.

Field	Description
Controller Registers	The set of commands grouped under this category are meant for register mapping for the controller. Controller Register contains following set of commands: ▪ Page Size Max ▪ Page Size Min ▪ Doorbell Stride ▪ Command Sets Supported ▪ Reserved ▪ Timeout ▪ Arbitration Mechanism Supported ▪ Contiguous Queues Required ▪ Max Queue Entries ▪ Interrupt Mask Set ▪ IO CQ Entry Size ▪ IO SQ Entry Size ▪ NVM Subsystem Reset Supported ▪ Command Sets Supported ▪ Enable ▪ Shutdown Status ▪ Controller Fatal Status

NVMe EndPoint - Identify Data Structure

To access this function: Click the **General Settings** icon from the left navigation pane in the GUI. Click the **Link Settings** tab. Select **NVMe** in the Application field. In the Session Type select "**To Upstream**" for the Exerciser to act as NVMe EndPoint Mode.Click **Identify Data Structure** tab.

You can also import the Identify Data Structure and namespace settings from an xml file that has the data captured using the Keysight PCIe Analyzer. You use the **Import from analyzer** toolbar button to do this import.

Field	Description
Controller	Various controller registers are displayed in this tab.
NameSpace	You can view and set the register settings for each of the NameSpaces supported by Exerciser in the Namespaces tab. NOTE: The maximum size allowed for a namespace is 1 GB. You can use the fields Namespace Size and Namespace Capacity to set the namespace size.

NVMe EndPoint - Log Page

To access this function: click **NVMe** in the navigation pane and click the **Log page** tab of the NVM Express Endpoint page.

The NVM Express Endpoint page is displayed only when you select **NVMe** in the Application field and **To Upstream** in the Session type field of the General Settings page.

To know more about Exerciser’s NVMe endpoint emulation, refer to the chapter “Emulating an NVMe EndPoint" on page 211.

Field	Description
Log Page	You can edit the log page structures using this tab. You can also import the Log Page Structures from an xml file that has the data captured using the Keysight PCIe Analyzer. You use the Import from analyzer toolbar button to do this import. Log page has the following tabs: ▪ Error Information ▪ Smart/Health Information ▪ Firmware Slot Information ▪ Reservation Notification

NVMe EndPoint - Features

To access this function: click **NVMe** in the navigation pane and click the **Features** tab of the NVM Express Endpoint page.

The NVM Express Endpoint page is displayed only when you select **NVMe** in the Application field and **To Upstream** in the Session type field of the General Settings page.

To know more about Exerciser's NVMe endpoint emulation, refer to the chapter "Emulating an NVMe EndPoint" on page 211.

Field	Description
Features	You can edit the values to be used with the Get and Set Features commands. Allows you to view and edit the default, current, and saved values for the different features using this tab. Features tab has the following tabs: ▪ Interrupt Coalescing ▪ Power Management ▪ Write Atomicity ▪ Temperature Threshold ▪ Asynchronous Event Configuration ▪ Error Recovery ▪ Autonomous Power State Transition ▪ Volatile Write Catch ▪ Software Progress Marker ▪ Arbitration ▪ LBA Range Type ▪ Interrupt Vector Configuration ▪ Number of Queues
Supported Capabilities	▪ **Saveable** - Select/deselect this checkbox to make the feature support / not support a Saveable value. ▪ **Namespace Specific** - Select/deselect this checkbox to make the feature Namespace specific or applicable across all Namespaces. ▪ **Changeable** - Select/deselect this checkbox to make the feature changeable / not changeable.

NVMe EndPoint - Queues

To access this function: click **NVMe** in the navigation pane and click the **Queues** tab of the NVM Express Endpoint page.

The NVM Express Endpoint page is displayed only when you select **NVMe** in the Application field and **To Upstream** in the Session type field of the General Settings page.

To know more about Exerciser's NVMe endpoint emulation, refer to the chapter "Emulating an NVMe EndPoint" on page 211.

Field	Description
Submission Queues - This section displays the following attributes of the available submission queues for Exerciser.	
Queue id	Is a unique id allocated to a Submission Queue.
Head	Is the Head pointer of the Submission Queue.
Tail	Is the Tail pointer of the Submission Queue.
Size	Is the Size of the Submission Queue.
Completion Queue ID	Is the Completion Queue ID associated to a Submission Queue.
Physical Address	Is the Physical Address of the Submission Queue.

Field	Description
Completion Queues - This section displays the following attributes of the available completion queues for Exerciser.	
Queue id	Is a unique id allocated to a Completion Queue.
Head	Is the Head pointer of the Completion Queue.
Tail	Is the Tail pointer of the Completion Queue.
Size	Is the Size of the Completion Queue.
Associated Interrupt Vector	Is the Associated Interrupt Vector of the Completion Queue.
Physical Address	Is the Physical Address of the Completion Queue.

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.
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.
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.
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), virtual hierarchies and virtual link (VL0) supported by Protocol Exerciser. If Exerciser emulates a non IOV/SRIOV component, then you can specify advertised credit limits for VC0 and VCx virtual channels. If Exerciser emulates a MRIOV component, then you can specify advertised credit limits for virtual hierarchies and virtual link VL0. The fields to specify the credit limits for virtual hierarchies and virtual link are enabled only when you select the MRIOV protocol while creating a link between Exerciser and DUT.
Header	For each virtual channel/virtual hierarchy/virtual link, 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.
Data	For each virtual channel/virtual hierarchy/virtual link, 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.
Initiate link recovery if no FC update is received for 200us in L0, L0s state	This checkbox is selected by default indicating that Exerciser automatically initiates link recovery if it does not receive any FC update packets for 200us when in L0 or L0s state. However, in certain L0s testing scenarios, you may not want Exerciser to do this. In such situations, you can deselect this checkbox to ensure that the link recovery is initiated as per the exit timer value that you specified in the General Settings function - Power Management tab. ***NOTE***: If the DUT advertised infinite credits for all the applicable credit types, then even if this checkbox is selected, Exerciser will not initiate link recovery after 200us of not receiving any FC update packets in L0 or L0s states.
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.

Field	Description
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.
Enable per VH Flow Control	As an MRIOV capable component, Protocol Exerciser supports two types of flow controls, per VH flow control and VL only 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 VL only 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.

15 Protocol Test Card

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, Keysight 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 B**, “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 30).

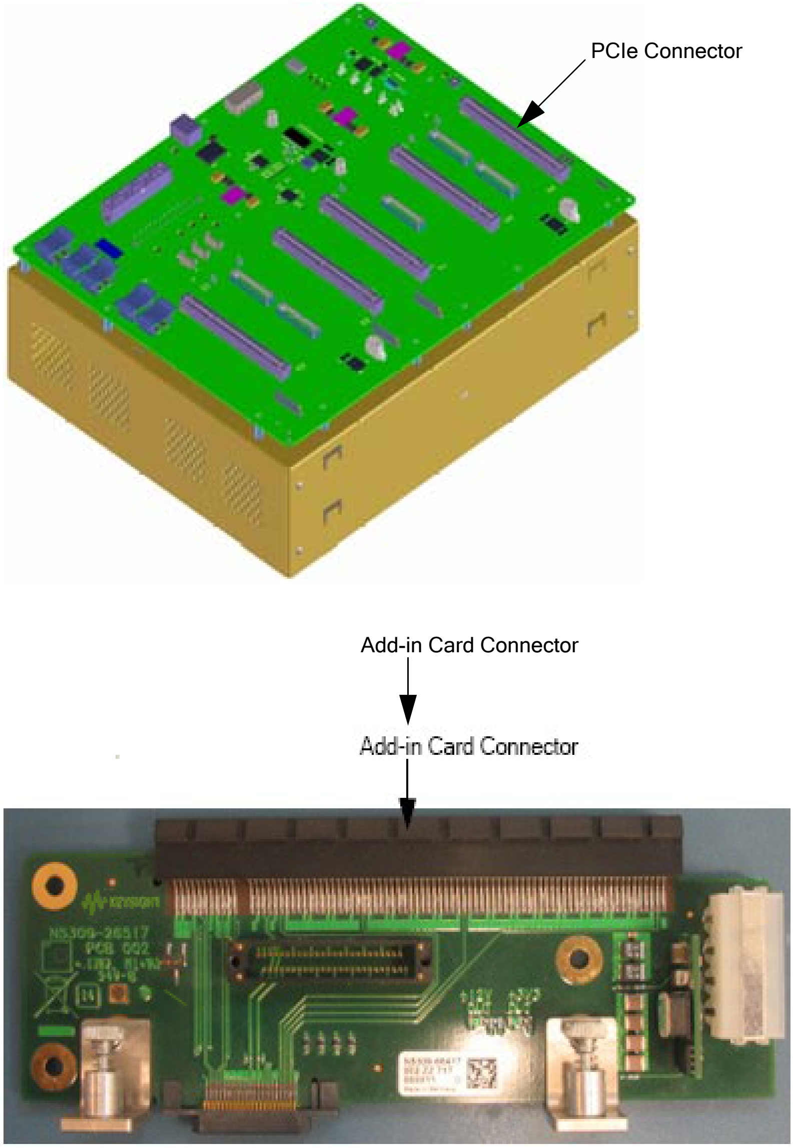


Figure 30 N5316A Passive Backplane and N5309-66417 Extension Card

NOTE

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

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

Accessing Protocol Test Card

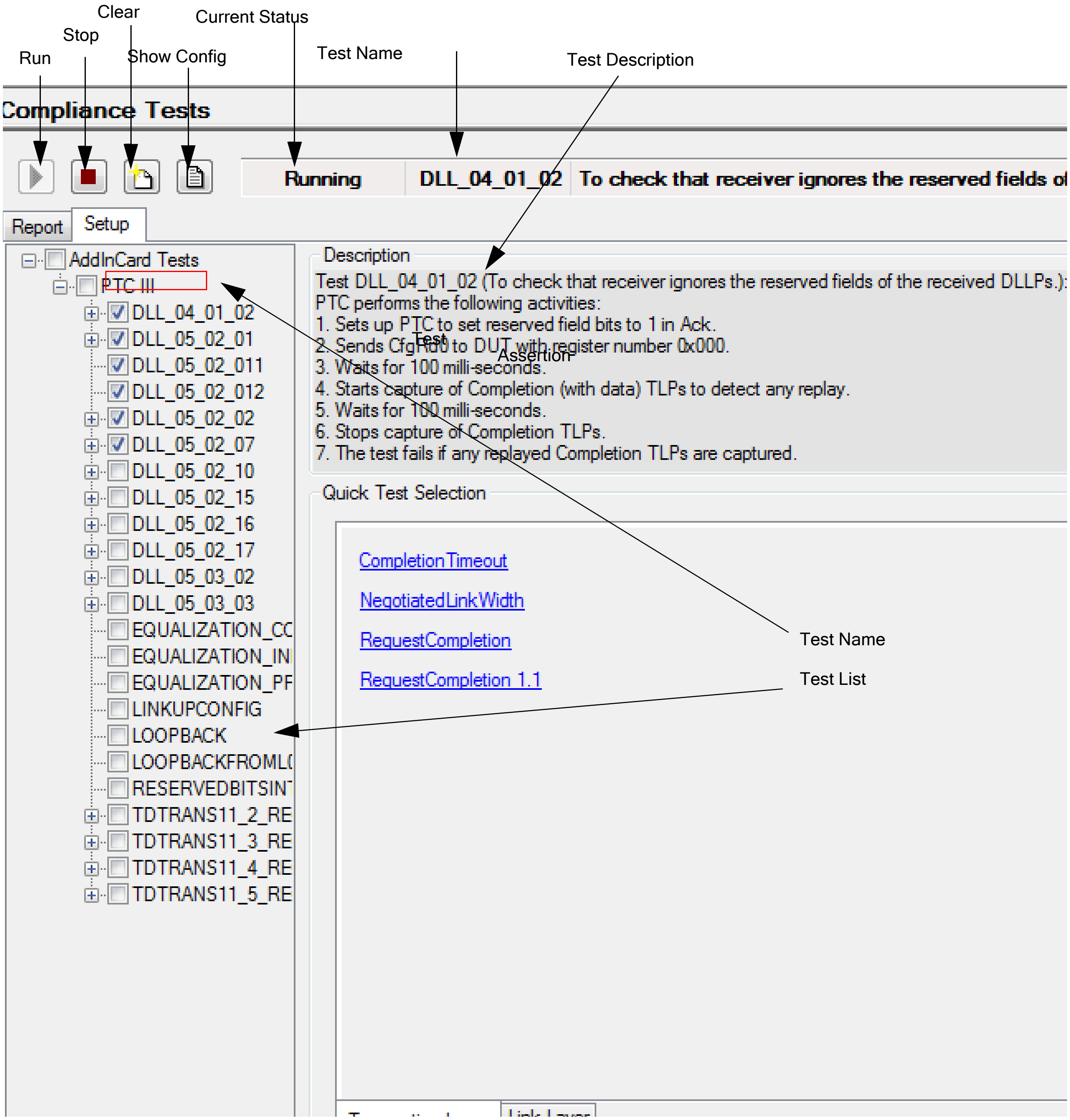
To access PTC:

- Start the Protocol Exerciser GUI.

- Select *PCIe Gen3 PTC* module type while adding port.

  The Compliance Tests screen appears (Figure 31).

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



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

- ☐ 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
    - ☐ RESERVEDBITSIN
    - ☐ 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

Figure 31 Compliance Tests screen

Table 3 briefly describes the components of the Compliance Tests screen.

Table 3 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.
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 31 shows the Setup tab, and Table 4 briefly describes the components of the Setup tab.

Table 4 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.*
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 5 briefly describes the components of the Report tab.

Table 5 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

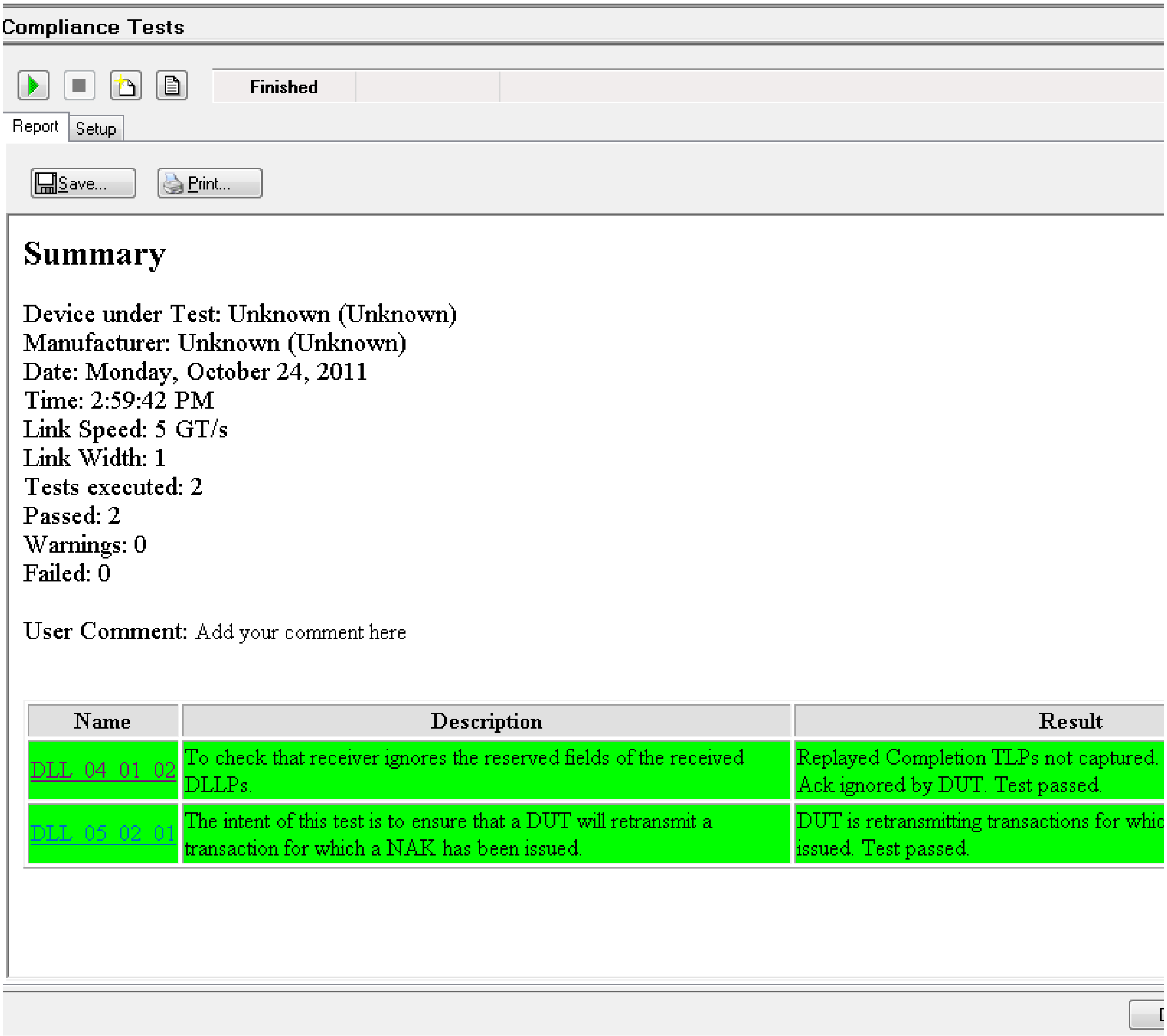
Consider that you want to run a couple of tests for your DUT to check:

- 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. 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 whic issued. Test passed.

Figure 32 Compliance Report

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

A Updating the Firmware

This appendix provides information on the Firmware Update tool.

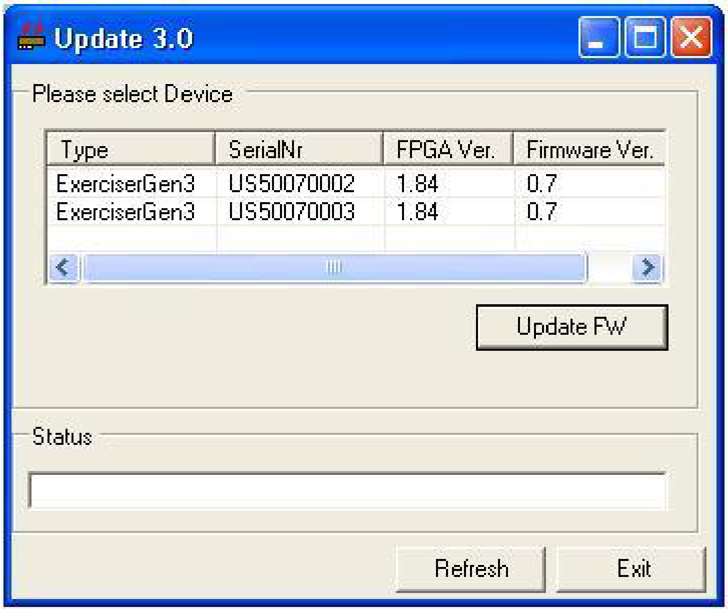
Using the Firmware Update Tool

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 > Keysight SPT > PCIe Exerciser 8.74 Release -> Update GUI**.

The **Firmware Update** window appears.



Update 3.0

Please select Device

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

Figure 33 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**.

B PTC Tests and Assertions

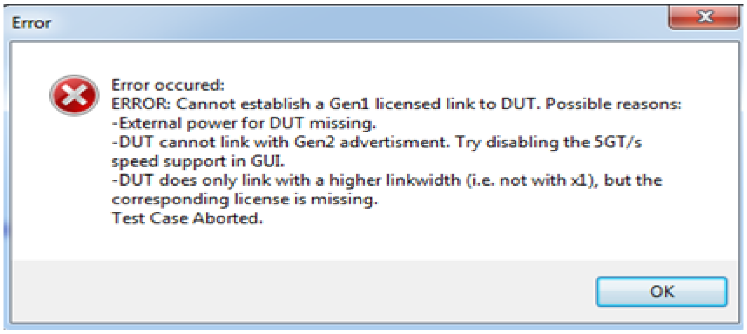
This appendix provides a very brief information on the tests available in PTC and their brief description.

Tests and Assertions

Table 6 lists and briefly describes the available PTC tests and assertions:

NOTE

When running PTC test(s), you may receive an error message similar to the message displayed below. PTC testing is designed to utilize a x1 link. If your device is more than x1 and your U4305A is more than x1 capable, this may cause difficulty in establishing a x1 link, which may be the result of such an error.



Error occured:
ERROR: Cannot establish a Gen1 licensed link to DUT. Possible reasons:
-External power for DUT missing.
-DUT cannot link with Gen2 advertisment. Try disabling the 5GT/s speed support in GUI.
-DUT does only link with a higher linkwidth (i.e. not with x1), but the corresponding license is missing.
Test Case Aborted.

It is recommended that you use a lane reducer or tape off the upper lanes of the U4305A.

Table 6 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.

Test	Intent	Assertion	Description
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.
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
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.

Test	Intent	Assertion	Description
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.

Index

Acknowledgements

Protocol Exerciser GUI uses Outlook Bar, a Graphical library created by Tim Dawson (Refer http://www.divil.co.uk/net/).

This information is subject to
change without notice.

Edition 10, November 2014

KEYSIGHT
TECHNOLOGIES

www.keysight.com