PROTOCOL SOLUTIONS GROUP
3385 SCOTT BLVD
SANTA CLARA, CA 95054

PE*Tracer* ML™ and PE*Tracer* EML™

PCI Express Multi-Lane Protocol Analyzer

PE*Trainer* ML™ and PE*Trainer* EML™

PCI Express Multi-Lane Exerciser

User Manual

Version 5.0

For Software Version 5.0

June 2006

Document Disclaimer

The information in this document has been carefully checked and is believed to be reliable. However, no responsibility can be assumed for inaccuracies that may not have been detected.

LeCroy reserves the right to revise the information in this document without notice or penalty.

Trademarks and Servicemarks

Le*Croy, CATC Trace, PCI Express, PETracer EML, PETracer ML, PETracer, PETrainer EML, PETrainer ML, Universal Protocol Analyzer System, UPAS*, and *BusEngine* are trademarks of LeCroy.

Microsoft and *Windows* are registered trademarks of Microsoft Inc.

All other trademarks are property of their respective companies.

Copyright

Part Number: 730-0070-00

TABLE OF CONTENTS

Section 1. General

Chapter 1: Overview

Designed for developers, the LeCroy PE*Trac*er ML™ and PE*Tracer* Edge are PCI Express™ Multi-Lane advanced verification systems. The LeCroy PE*Tracer* EML™ is a PCI Express 16-lane advanced verification system.

By leveraging years of experience in protocol analysis tools for emerging markets, PE*Tracer* Edge, PE*Tracer* ML, and PE*Tracer* EML blend sophisticated functionality with practical features to speed the development of PCI Express IP cores, semiconductors, bridges, switches and systems.

1.1 PE*Tracer* Analyzer Hardware and Software

Features and Benefits

PE*Tracer* Edge, PE*Tracer* ML, and PE*Tracer* EML Analyzers have these features and benefits:

Features	**Benefits**
Powerful and Intuitive CATC Trace Expert Software	Faster interpretation and debug of PCI Express traffic with color-coded, clearly labeled protocol elements. Reduces coding and deciphering.
Extensive Decoding	Complete, accurate and reliable decoding of TLPs (Transaction Layer Packets), DLLPs (Data Link Layer Packets), and all PCI Express Primitives
Monitoring and Link Utilization	View link utilization and other performance measurements changing in real time. Save this data to view as post-session statistics.
Advanced Triggering	Isolate important traffic, specific errors, or data patterns. Trigger condition setup is made simple with drag and drop tools to link events to trigger or filter actions.
Hardware Filtering	Analyze faster and understand transactions more clearly by removing non-essential fields from the trace.
Intelligent Reporting	Quickly identify and track error rates and abnormal link or timing conditions. Display configuration space and protocol specification details.
Sophisticated Viewing	View Packet, Link and Split Transaction levels of the PCI Express protocol. Collapsible/expandable headers with Tool tip explanations make it easy to navigate and interpret packet contents, headers, and commands. View packets, transactions of TLPs and DLLPs in classic CATC Trace™ format or in raw 10b code for deeper analysis.
2 GB Recording Capacity (PE*Tracer* Edge and PE*Tracer* ML) 8 GB Recording Capacity (PE*Tracer E*ML)	Capture long recording sessions for analysis and problem solving
Hi-Speed USB Port	Access PCI Express recordings up to 40x faster than USB 1.1. No complicated setup required.
Downloadable Trace Viewer	Share and annotate trace recordings within a development team. Freely distributable PE*Tracer* software enables collaborative analysis across sites and time zones.

Taps non-intrusively into a single x1, 2.5 gigabit/second PCI Express port using MicroGigaCN receptacle and one standard x4 cable	Accurate and non-intrusive collection of PCI Express trace data
Field-upgradeable firmware and recording engine	Upgrades available for download direct from LeCroy
PE*Tracer* Slot Interposer Probe Card	Modular form factor
External interface for probing and monitoring auxiliary digital signals	Enables cross triggering between other test instruments.
One-year Hardware Warranty	Protect your investment.
Dword to Transaction Level Viewer	View Dwords, Packet, Link and Split Transaction levels of the PCI Express protocol.
CRC checking	Know that info being displayed is accurate.

1.2 PE*Trainer* Exerciser Hardware

Features and Benefits

PE*Trainer* ML and PE*Trainer* EML Exercisers have these features and benefits

Features	**Benefits**
Script level traffic generation	Programmability to test PCI Express components with more precision and control
Convert Trace files into generation scripts	Recreate failure scenarios by replaying recorded traffic
Manual Error injection	Verify fault handling and identify error recovery
Host/End-Point Emulation Support	Host emulation platform/end-point emulation Interposer allows for design and stress testing.
Programmable Physical Layer	Flexibility to program lane skews, link control bits, skip intervals and link states for more robust verification
Programmable Data Link Layer	Ability to modify flow control, ACK/NAK, and retry behaviors
Flexible/programmable Transaction Layer	User ability to define arbitrary sequence of transactions, payload generation, and conditional repeat of transactions provide users with maximum flexibility
Raw Generation mode	Allows complete control of each symbol transmitted
Programmable reply timers	Allows testing of ACK latency timeouts and retry mechanisms
Point and Click Script Editor	Complex scripts can be created quickly and easily
Three Year Hardware Warranty	Protect your investment with industry leading warranty
Error Insertion	CRC Errors Running Disparity Errors Insertion of Invalid 10 bit symbols Lane to Lane Skew Insertion Lane Reversal Polarity Inversion
Programmable ACK/NAK behavior	ACK all TLP packets NAK all TLP packets Automatic ACK/NAK behavior ACK/NAK delay timer

Programmable Flow control behavior	Set Credit values Disable Credit checking Set Update InitFC intervals
Automatic CRC calculation and Sequence number assignment	DLLP CRC calculation TLP LCRC calculation
Replay buffer	Programmable Transaction timeout timer Auto Recovery behavior after 4 Replays
Programmable configuration space	Full 4 KB configuration space configurable by user Accessible through Configuration Reads and Writes over PCI Express
Programmable address spaces (PE*Tracer* EML only)	
Automatic transaction completer (PE*Tracer* EML only)	

1.3 CATC Trace Software

PE*Tracer* Edge, PE*Tracer* ML, and PE*Tracer* EML utilize the CATC Trace™ to assist users in analyzing how PCI Express components work together, diagnose problems, and test for interoperability and standards compliance.

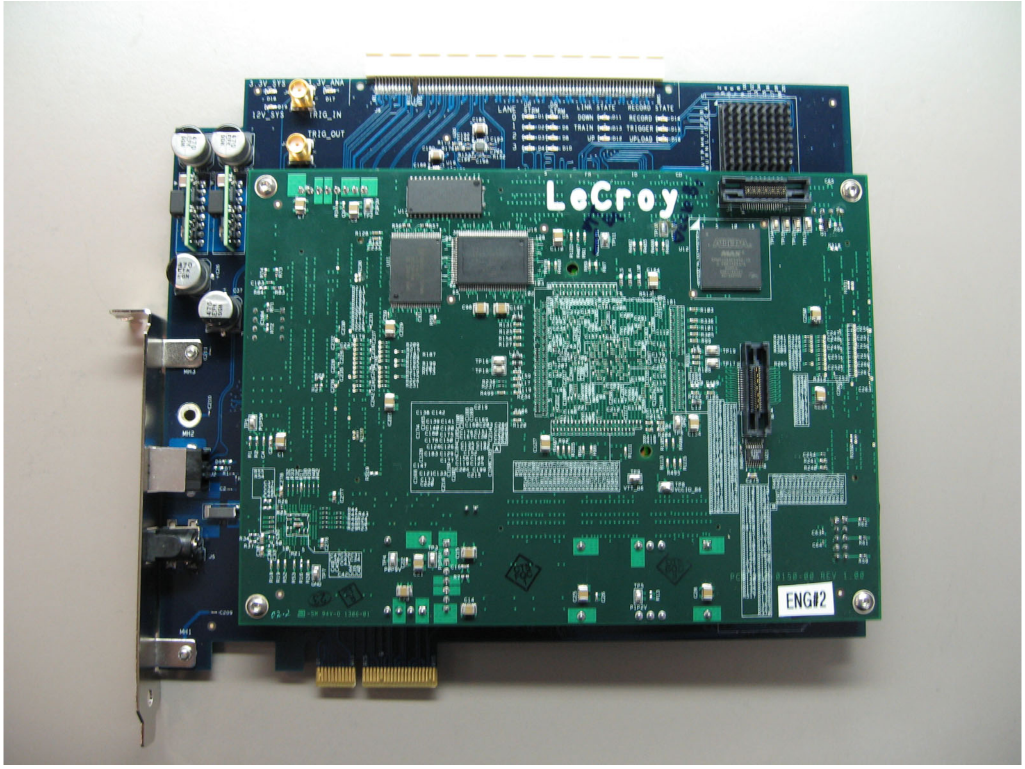
The CATC Trace is a powerful and intuitive expert software system embedding detailed knowledge of the protocol hierarchy and intricacies, as defined in the protocol specification. The software allows the user to control the Analyzer and set specific real-time triggering and filtering conditions. The CATC Trace utilizes a Windows-based graphical display that has been optimized for fast and easy navigation through a captured traffic session. Users are alerted as violations are detected at all levels of the protocol layering and can easily drill down on areas of interest or collapse and hide fields that are not relevant.

For additional information on the CATC Trace, please download the White Paper from the LeCroy website: **http://www.lecroy.com/support/white_papers/index.html.**

1.4 PE*Tracer* Edge

The PE*Tracer* Edge Analyzer uses a high-impedance, non-intrusive probing technology, allowing full, unaltered data pass-through. The analyzer supports bidirectional x1, x2, and x4 PCI Express link data capture and analysis. It supports PCI Express specification version 2.0 at 2.5 GB/s per lane. This product is ideal for anyone doing protocol development for add-in boards, servers, desktops, and workstations.

The PE*Tracer* Edge Analyzer installs directly in the PCI Express downstream slot, between the Root Complex and the card under test.



The PCI Express PE*Tracer* Edge is offered in two classes: PRO and EXPERT.
PRO users will appreciate a low cost but highly featured class of analyzer.
EXPERT users will recognize advanced features that enable deep analysis for error troubleshooting of bus management and protocol operation issues. Both classes of the PCI Express PE*Tracer* Edge have many innovative features that dramatically reduce debugging time, accelerating time-to-market for PCI Express solutions.

The PCI Express PE*Tracer* Edge utilizes the CATC Trace to assist users in analyzing how PCI Express components work together in diagnosing problems. The CATC Trace is a powerful and intuitive expert software system, embedding detailed knowledge of the protocol hierarchy and intricacies, as defined in the protocol specification.

The CATC Trace utilizes a Windows-based graphical display that has been optimized for fast and easy navigation through a captured traffic session. Users are alerted as violations are detected at all levels of the protocol layering and can easily drill down to areas of interest or can collapse and hide fields that are not relevant. Protocol data can be viewed in several ways from logical to chronological, as well as by events unique to PCI Express.

You can find errors quickly by using powerful triggering, filtering, and error reporting. You can view meaningful reports about performance and protocol behavior in real time and as post-captured traffic. You can know that your data is accurate through CRC rechecking, which produces reliable and complete decodes of Transaction Layer Packets (TLPs), Data Link Layer Packets (DLLPs), and all PCI Express primitives. For deeper analysis, you can display packet contents as raw 10-bit codes.

Powerful display views allow for easy analysis of protocol traffic.

PCI Express PE*Tracer* Edge users who are budget conscious today can get started with the PRO and move to EXPERT functionality later.

Features	**PRO**	**EXPERT**
Protocol Hierarchical Display	**x**	**x**
DWORD View	**x**	**x**
Link Tracker	**x**	**x**
Trace Navigator	**x**	**x**
TLP Packet Script Decoding	**x**	**x**
Trigger/Filter Control	**x**	**x**
Power Search	**x**	**x**
Header Field Viewer	**x**	**x**
Config Space Viewer	**x**	**x**
Advanced Hide	**x**	**x**
Timing Calculator	**x**	**x**
x4 / x2 / x1 lane width support	**x**	**x**
Performance Metrics	**x**	**x**
Flow Control Credit Tracking	**x**	**x**
Automation API	**x**	**x**
Expert Recording Buffer Size	256 MB	2 GB
Expert Real-time Bus Monitoring		**x**
Expert Triggering		**x**
Expert Traffic Summary View		**x**
Expert Graphical Bus Utilization View		**x**
Verification Script Engine		**x**
Compliance Ready		**x**

1.5 PE*Tracer* ML Analyzer

The LeCroy PE*Tracer* ML hardware taps transparently into the fabric to capture, decode, and display x1, x2, x4 and x8 bidirectional transfers of PCI Express traffic. Developers use the CATC Trace™ expert software system to define recording, triggering and filtering conditions and to view, analyze and print the trace.

PE*Tracer* ML uses a high impedance, non-intrusive probing technology, allowing full, unaltered data pass-through. The Analyzer offers a dual approach in tapping into the PCI Express fabric. Whether the design utilizes a card-edged solution, or implements the PCI Express link between circuit board components, PETracer ML provides support for both a Slot Interposer card and a mid-bus probe (for more information about mid-bus probes, refer to the *PETracer Mid-Bus Probe™ Installation Guide*).

A single PE*Tracer* ML Analyzer supports bidirectional x1, x2 and x4 links, as well as unidirectional x8 PCI Express link data capture and analysis. Combining two PE*Tracer* ML Analyzer systems allows for full bidirectional decode and capture of a x8 PCI Express link.

PE*Tracer* ML has many innovative features that dramatically reduce time-to-market for PCI Express solutions, including a comprehensive set of triggering and filtering options that allow the user to capture traffic that is of most interest and relevance.

Users are provided with a list of protocol events from link conditions to error violations, which enables them to set commands to the Analyzer to search, isolate and capture the essential data.

PE*Tracer* ML and PE*Tracer* EML provide accurate, reliable and complete decodes of Transaction Layer Packets (TLPs), Data Link Layer Packets (DLLPs) and all PCI Express primitives. For deeper analysis, you can display packet contents as raw 10-bit codes.

The PE*Tracer* ML Multi-Lane PCI Express Protocol Analyzer is a hardware module that installs into the LeCroy Universal Protocol Analyzer System™ (UPAS™) 10000. The UPAS 10000 is a base system designed to accommodate different protocol modules.

Together, PE*Tracer* ML and the UPAS connect to a PC that runs the PE*Tracer* user interface. This interface administers the Analyzer, displays traces, and analyzes the performance of the devices under test.

PE*Tracer* ML supports the pairing of two UPAS/PE*Tracer* ML. Pairing doubles the width of the PCI Express bidirectional link that can be monitored from x4 to x8.

PE*Tracer* ML uses hardware triggering to capture real-time events, and hardware filtering to filter in or out different types of packets from the recording. Filtering allows users to focus recordings on events of interest and to preserve recording memory so that the recording time can be extended. All traffic is recorded non-intrusively, meaning that PE*Tracer* ML does not adversely affect the link it is recording.

Recorded data is presented in colored graphics in a trace viewer application. This application has advanced search and viewing capabilities that allow the user to quickly locate specific data, errors, and other desired conditions, thereby focusing the user's attention on events of interest.

The PE*Tracer* ML Protocol Analyzer functions with any personal computer using the Microsoft® Windows® 2000 or Windows XP operating system and equipped with a functional USB interface.

PE*Tracer* ML provides on-the-fly detection of and triggering on PCI Express events. Whether recording manually or with a specified trigger condition, PE*Tracer* ML continuously records the link data in a wrap-around fashion until manually stopped or until the Trigger Event is detected and a specified amount of post-Trigger link data is recorded.

Upon detection of a triggering event, the Analyzer continues to record data up to a point specified by the user. Real-time detection of events can be individually enabled or disabled to allow triggering on events as they happen. This includes predefined exception or error conditions and a user-defined set of trigger events. The unit can also be triggered by an externally supplied signal. An external DB-25 connector provides a path for externally supplied data or timing data to be recorded along with traffic.

This DB-25 connector also provides a path for PE*Tracer* ML to transmit signals externally for purposes of probing and use by other circuitry.

The PE*Tracer* ML software provides powerful search functions that enable investigation of particular events and allow the software to identify and highlight specific events. In addition to immediate analysis, you can print any part of the data. Use the **Save As** feature to save the data on disk for later viewing. The program also provides a variety of timing information and data analysis reports.

1.6 PE*Trainer* ML Exerciser

The LeCroy PE*Trainer*™ ML™ is a Multi-lane PCI Express (x1, x4) Exerciser designed to assist engineers in improving reliability of their solutions and providing advanced capabilities for stress and compliance testing. Together with the LeCroy PE*Tracer* ML expert PCI Express protocol analysis solutions, PE*Trainer* ML reduces time to market by enabling users to quickly identify logic and design flaws.

PE*Trainer* ML is designed as a stand alone Multi-lane PCI Express Exerciser that is capable of generating and responding to all types of PCI Express transactions while allowing users to create protocol variations and anomalies. The users may also create corner case and stress test scenarios to evaluate the robustness of their solutions. By utilizing the PE*Trainer* ML error injection feature, engineers can create worst-case PCI Express traffic scenarios allowing them to validate the error handling capabilities of their solutions.

When PE*Trainer* ML is used in conjunction with the LeCroy PE*Tracer* ML or LeCroy PE*Tracer* Analyzer, a complete expert test and analysis system is created. This integrated solution delivers traffic generation and expert protocol analysis to assist developers with early validation of designs along with error injection and stress testing in preparation for compliance testing.

1.7 PE*Tracer* EML Analyzer

The PE*Tracer* EML allows semiconductor, motherboard and add-in card manufacturers to capture, analyze and view PCI Express traffic.

There are two ways to connect the PE*Tracer* EML to the point-to-point PCI Express bus, using either a card-edge slot Interposer card or a mid-bus probe (for more information about mid-bus probes, refer to the *PETracer Mid-Bus Probe™ Installation Guide*). PCI Express traffic can then be non-intrusively recorded. Live data is sent from the PE*Tracer* EML to a PC in order display real-time statistics. Sophisticated trigger options are used to capture and buffer only data of interest. Real-time hardware filters in the system filter out unwanted traffic in order to maximize memory buffer utilization. Captured data is uploaded to the PC for further processing and displayed by the powerful CATC Trace^{TM} software.

PE*Tracer* EML provides accurate, reliable and complete decodes of Transaction Layer Packets (TLPs), Data Link Layer Packets (DLLPs), and all PCI Express ordered sets. The buffer size has been increased in the *PETracer* EML to 4GB per direction. PCI Express Specification 1.1, 1.0a, and 1.0 are supported at the full-speed of 2.5GHz per lane. Automatic disparity checking, lane deskewing and data descrambling are all supported. The FPGA based design is fully field upgradeable. The PE*Tracer* EML can be controlled remotely via the user's LAN.

Sophisticated controls allow the user to decide what is captured and how it is viewed in order to analyze design problems and perform root cause analysis. Powerful triggering conditions allow the user to trigger on Error, Link, TLP, DLLP or any user-defined data pattern in the traffic. Each type of trigger condition can be cascaded to create complex multi-layer triggers. Trigger points can be set to any location within the buffer to allow the user to capture data before and after the trigger. Full-featured filtering capabilities allow engineers to isolate areas of interest. Captured packets can be expanded or collapsed. Columns are user configurable to allow hiding fields that are not relevant. Powerful search capabilities further enhance the users abilities to solve engineering and production problems.

The PE*Tracer* EML includes a powerful expert system that incorporates detailed knowledge of the hierarchy and intricacies of the PCI Express protocol specification to create multiple graphical views of PCI Express data. These Windows®-based color-coded graphical views have been optimized for fast and easy navigation through a captured traffic session. Each packet is decoded to display packet direction, sequence number, timestamp, packet type and other packet specific fields. Errors are detected at all levels of the protocol hierarchy and color highlighted for the user. Error conditions include: DLLP CRC, LCRC, ECRC, delimiter, disparity and symbol errors. Real-time statistics and post processing traffic summaries give the big picture of the data captured. Statistics can be viewed in tabular form or viewed in graphical form. Traffic summary views are hyperlinked back to data traces to quickly jump to the source of bugs.

The PCI Express protocol data can be viewed in several different hierarchical views. Data can be displayed at the physical layer with Link Tracker to display either 10B/8B codes, Symbols or hex in either raw or scrambled formats. Traffic can also be displayed at the packet or Link layer or in the application layer (split view). Placing the mouse pointer over packet data causes a tooltip with an explanation of the field. Tooltips explain annotations and often display substantial additional information about selected fields.

The PE*Tracer* EML is the first add-in module to utilize LeCroy's most advanced Universal Protocol Analysis System (UPAS), the UPAS 100K platform. The expanded memory buffer of 4GB per direction allows the user to capture deeper traces over a longer time period. The platform also provides scalability for the future with expanded hardware capabilities and the capacity for two independent plug-in modules in the same chassis. The UPAS 100K was specifically designed to support high-speed serial protocols like PCI Express.

PE*Tracer* EML has full x16 lane analysis capabilities as well as many innovative triggering and filtering options that allows the user to easily diagnose problems and test for interoperability and standards compliance in order to quickly create marketable PCI Express products.

1.8 PE*Trainer* EML Exerciser

LeCroy PE*Trainer*™ EML™, a PCI Express 16-lane advanced Exerciser system, is a critical test and verification tool intended to assist engineers in improving the reliability of their systems. PCI Express root complexes or device endpoints can be emulated. Built upon LeCroy's most advanced Universal Protocol Analysis System (UPAS) platform, the UPAS 100k, the PE*Trainer* EML is LeCroy's second generation PCI Express Exerciser. By leveraging years of experience in protocol analysis tools for emerging markets, the PE*Trainer* EML blends sophisticated functionality with intuitive controls to speed the development and testing of PCI Express IP cores, semiconductors, bridges, switches and systems.

The PE*Trainer* EML is a stand-alone PCI Express Exerciser that is capable of generating and responding to all types of PCI Express transactions. The powerful scripting language allows for the creation of Transaction Layer Packets (TLPs), Data Link Layer Packets (DLLPs) and Ordered Sets. ACK's or even NAK's can be automatically generated under user control. Any packet field not explicitly specified is created automatically. For example CRC's can be automatically calculated. Configuration ROM space can be emulated for any device including endpoints, bridges and switches.

A Configuration Space can be easily created by exporting a Configuration Space file created by the PE*Tracer*.

The scripting language is powerful yet easy to use. One way to create test scripts is to export traffic from a trace file captured with a LeCroy PE*Tracer*. The exported script can then be modified to generate different test cases, insert errors or create loop tests. The “point and click” capability of the script editor makes modifying or creating scripts from scratch simple. The powerful scripting language allows for a link training script to be created with just three simple commands. Macro's can also be created for frequently used sequences.

1.9 Other PE*Tracer and* PE*Trainer* Documents

For more information about PE*Tracer* and PE*Trainer*, refer to the following documents:

- *PCIe Compliance Tests Software User Manual v1.3*
- *LeCroy Analyzers File-based Decoding Manual v1.2*
- *PETracerTrainer Automation Manual v1.70*
- *PETracer VSE Manual v1.30*
- *PETracer Mid-Bus Probe™ Installation Guide v1.50*
- *PETrainer Scripting Language Reference Manual v1.70*
- PE*Tracer* Online Help

Chapter 2: Hardware Description

The PCI Express™ system features the Universal Protocol Analyzer System™ (UPAS™), PE*Tracer*™ Analyzers, and PE*Trainer*™ Exercisers.

2.1 System Components

- **PE*Tracer* Edge system:**
  - PE*Tracer* Edge card
  - One USB Cable
  - 12 V external power supply
  - PE*Tracer* Software program CD-ROM
  - User Manual
- **PE*Tracer* ML system (based on the UPAS 10K platform):**
  - Factory-installed PE*Tracer* ML Analyzer module(s)
  - One USB cable
  - Probe data cable
  - Breakout Board with cable
  - PE*Tracer* Software program CD-ROM
  - User Manual
  - PCI Express Slot Interposer probe
- **PE*Trainer* ML system (based on the UPAS 10K platform):**
  - Factory-installed PE*Trainer* ML Exerciser module
  - One USB cable
  - Probe data cable
  - Breakout Board with cable
  - PE*Tracer* Software program CD-ROM
  - User Manual
  - Device Emulation Interposer or Host Emulation Test Fixture
- **PE*Tracer* EML system (based on the UPAS 100K platform):**
  - Two factory-installed PE*Tracer* EML Analyzer module(s)
  - One USB cable
  - Two probe data cables
  - Breakout Board with cable
  - x16 to x1 Edge Adapter
  - x16 to x4 Edge Adapter
  - x16 to x8 Edge Adapter
  - PE*Tracer* Software program CD-ROM
  - User Manual
  - PCI Express Slot Interposer probe

- **PE*Trainer* EML system (based on the UPAS 100K platform):**
  - Two factory-installed PE*Trainer* EML Exerciser modules
  - One USB cable
  - Two probe data cables
  - Breakout Board with cable
  - x16 to x1 Edge Adapter
  - x16 to x4 Edge Adapter
  - x16 to x8 Edge Adapter
  - PE*Tracer* Software program CD-ROM
  - User Manual
  - Device Emulator Card or Host Emulation Test Fixture

2.2 Host PC Requirements

PE*Tracer* Edge, PE*Tracer* ML, PE*Tracer* EML, PE*Trainer* ML, and PE*Trainer* EML connect to a Host PC. Please consult the readme file on the installation CD for the latest PC requirements.

2.3 Universal Protocol Analyzer System (UPAS)

The Universal Protocol Analyzer System (UPAS) is a base unit for LeCroy protocol Analyzer modules. The UPAS contains the recording memory and other resources used by all LeCroy Analyzer modules. At the back of the UPAS is a hi-speed USB port for transferring trace data to a PC.

There are two models of UPAS:

- **UPAS 100K**: The newest and fastest of the LeCroy base systems. This unit supports the PE*Tracer* EML™.

- **UPAS 10K**: Supports the PE*Tracer* ML™.

2.4 Protocol Analyzer Modules

PE*Tracer* EML and PE*Tracer* ML are hardware modules that plug into LeCroy’s Universal Protocol Analyzer System (UPAS).

- **PE*Tracer* EML**: x16 version of PE*Tracer*. PE*Tracer* EML works with the UPAS 100K.

- **PE*Tracer* ML**: Multi-lane version of PE*Tracer*. It works with the UPAS 10K and occupies both slots of the UPAS.

The two Analyzer modules use hardware triggering to capture real-time events and hardware filtering to exclude unwanted data. Data is recorded into the Analyzer's buffer and then transferred over a USB line to a PC for display and analysis. Search and display capabilities in the Analyzer software allow you to quickly locate specific data, errors and other desired conditions.

An external DB-25 (PE*Tracer* ML) or DB-44 (PE*Tracer* EML) connector provides a path for externally supplied data or timing information to be recorded along with link traffic.

2.5 PE*Trainer* Exerciser

PE*Trainer*™ is a Multi-lane PCI Express Exerciser designed to assist engineers in improving reliability of their solutions and providing advanced capabilities for stress and compliance testing.

PE*Trainer* Models

LeCroy makes two models of PE*Trainer*:

- PE*Trainer* ML™: 4-lane system

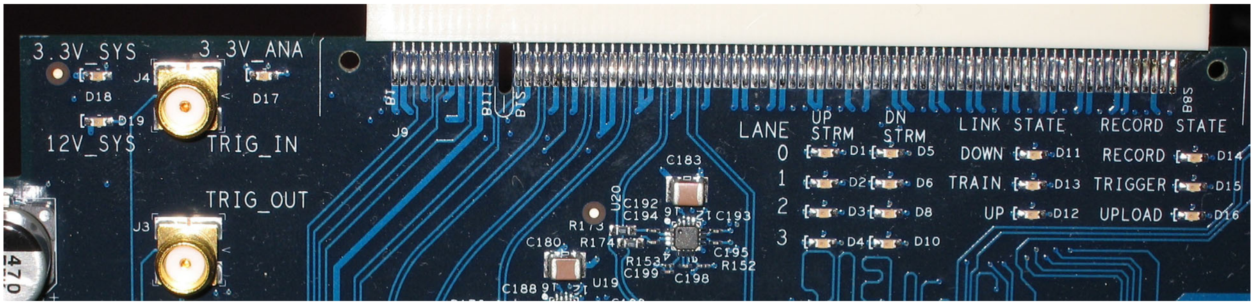
- PE*Trainer* EML™: 16-lane system

Both models of PE*Trainer* operate as stand alone Exercisers capable of generating and responding to all types of PCI Express transactions. Both Exercisers also have the ability to create protocol variations and anomalies. Users may also create corner case and stress test scenarios to evaluate the robustness of their solutions. By utilizing the PE*Trainer* error injection feature, engineers can create worst-case PCI Express traffic scenarios allowing them to validate the error handling capabilities of their solutions.

When PE*Trainer* is used in conjunction with a PE*Tracer* Analyzer, a complete expert test and analysis system is created. This integrated solution delivers traffic generation and expert protocol analysis to assist developers with early validation of designs along with error injection and stress testing in preparation for compliance testing.

2.6 PE*Tracer* Edge Analyzer LED Descriptions

When powered on, the PE*Tracer* Edge activates the user-accessible controls and LEDs on the top of the Edge card:



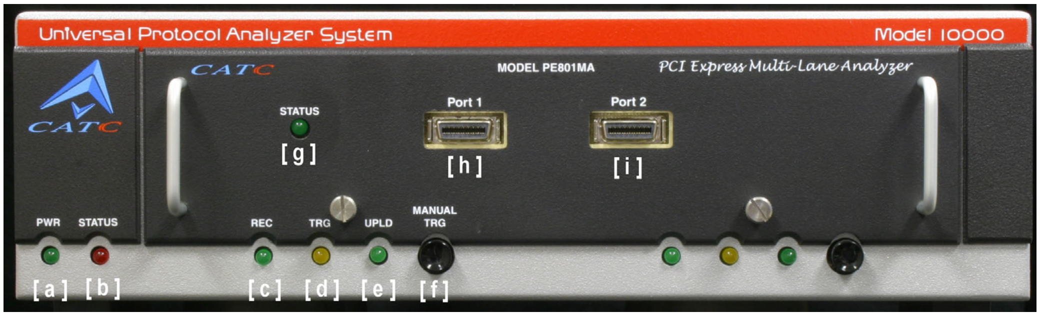
- **3.3V_SYS (D18)**: Lights when 3.3 Volt power is provided by the slot under test.
- **12V_SYS (D19)**: Lights when 12 Volt power is provided by the slot under test.
- **3.3V_ANA (D17)**: Lights when external power is provided to the Edge card.
- **LANE 0 UP STRM (D1)**: Lights when there is activity on the Upstream physical lane number 0.
- **LANE 0 DN STRM (D5)**: Lights when there is activity on the Downstream physical lane number 0.
- **LANE 1 UP STRM (D2)**: Lights when there is activity on the Upstream physical lane number 1.
- **LANE 1 DN STRM (D6)**: Lights when there is activity on the Downstream physical lane number 1.
- **LANE 2 UP STRM (D3)**: Lights when there is activity on the Upstream physical lane number 2.
- **LANE 2 DN STRM (D8)**: Lights when there is activity on the Downstream physical lane number 2.
- **LANE 3 UP STRM (D4)**: Lights when there is activity on the Upstream physical lane number 3.
- **LANE 3 DN STRM (D10)**: Lights when there is activity on the Downstream physical lane number 3.
- **LINK STATE DOWN (D11)**: Indicates state of the physical PCI Express link. Lights when the link is down.
- **LINK STATE TRAIN (D13)**: Indicates state of the physical PCI Express link. Lights when the link is in training.
- **LINK STATE UP (D12)**: Indicates state of the physical PCI Express link. Lights when the link is up.
- **RECORD STATE RECORD (D14)**: Lights when the Edge is recording.
- **RECORD STATE TRIGGER (D15)**: Lights when the Edge triggers on an event.
- **RECORD STATE UPLOAD (D16)**: Lights when the Edge is uploading data to the Host PC.

2.7 PE*Tracer* ML Analyzer LED Descriptions

When powered on, the PE*Tracer* ML activates the user-accessible controls and LEDs on the front and rear panels of the UPAS.

Warning! Do not open the UPAS enclosure. There are no operator serviceable parts inside. Refer servicing to LeCroy.

UPAS LEDs (from left to right)



(a) Green **PWR** (power) indicator LED for UPAS: lights when the unit power is switched on.

(b) Red **Status** indicator LED for UPAS: lights during initialization/power up of UPAS base unit. Blinks if a self-test fails.

The following LEDs and the **Manual Trigger** button on the UPAS function in conjunction with the PE*Tracer* ML module inserted above it:

(c) Green **REC** (recording) LED: lights when the unit is recording.

(d) Orange **TRG** (triggered) LED: lights when the unit triggers on an event.

(e) Green **UPLD** (Upload): lights when unit is uploading data to PC.

(f) **Manual Trigger** push-button: allows a manual Trace capture.

LEDs and Connectors on the PE*Tracer* ML Module

(g) Green **Status** LED: Illuminates when there is traffic on the link.

(h) **Probe Data Connector**

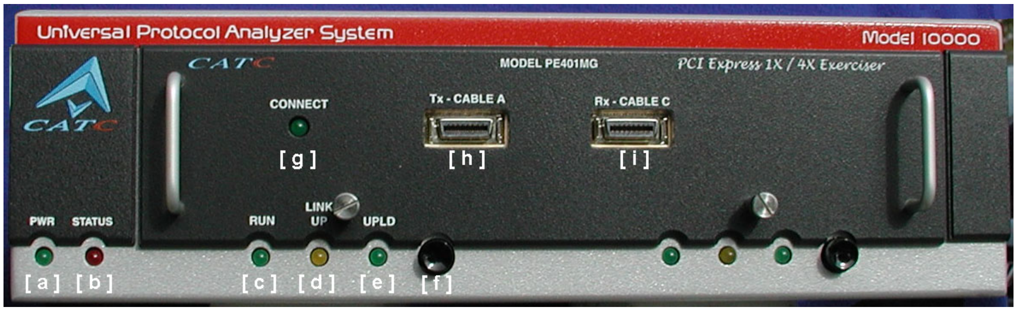
(i) **Probe Data Connector**

2.8 PE*Trainer* ML Exerciser LED Descriptions

When powered on, the PE*Trainer* ML activates the user-accessible controls and LEDs on the front and rear panels of the UPAS.

Warning! Do not open the UPAS enclosure. There are no operator serviceable parts inside. Refer servicing to LeCroy.

UPAS LEDs (from left to right)



(a) Green **PWR** (power) indicator LED for UPAS: Lights when the unit power is switched on.

(b) Red **Status** indicator LED for UPAS: Lights during initialization/power up of UPAS base unit. Blinks if a self-test fails.

The following LEDs on the UPAS function in conjunction with the PE*Trainer* ML module inserted above it:

(c) Green **Run** LED: Lights when script execution is in process.

(d) Orange **Link Up** LED: Lights when the PCI Express link is trained.

(e) Green **UPLD** (Upload): Lights when script is transferred from PC to PE*Trainer*.

(f) **Manual Trigger** push-button: Disabled.

LEDs and Connectors on the PE*Trainer* ML Module

(g) Green **Connect** LED: Indicates that cabling is correct.

(h) **Tx Cable A**: Corresponds to connector A on probe data cable

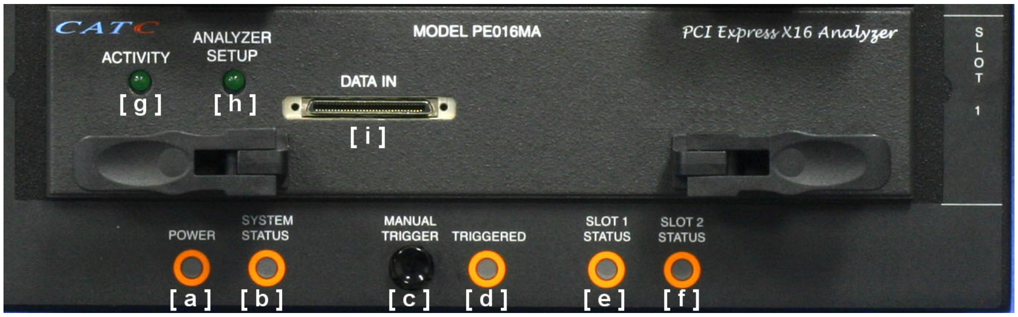
(i) **Rx Cable C**: Corresponds to connector C on probe data cable

2.9 PE*Tracer* EML Analyzer LED Descriptions

When powered on, the PE*Tracer* EML activates the user-accessible controls and LEDs on the front and rear panels of the UPAS.

Warning! Do not open the UPAS enclosure. There are no operator serviceable parts inside. Refer servicing to LeCroy.

UPAS LEDs (from left to right)



(a) **Power** indicator LED for UPAS: Lights when the unit power is switched on.

(b) **System Status** indicator LED for UPAS: Lights during initialization/power up of UPAS base unit. Blinks if a self-test fails.

(c) **Manual Trigger** push-button: Forces a trace trigger.

(d) **Triggered LED**: Lights when the unit triggers on an event.

(e) **Slot 1 Status**: Illuminates when the Analyzer in Slot 1 is recording

(f) **Slot 2 Status**: Illuminates when the Analyzer in Slot 2 is recording

PE*Tracer* EML Module

(g) **Activity LED**: Illuminates when there is traffic on the link.

(h) **Analyzer Setup LED**: Illuminates when cable is correctly connected between front of Analyzer and Interposer.

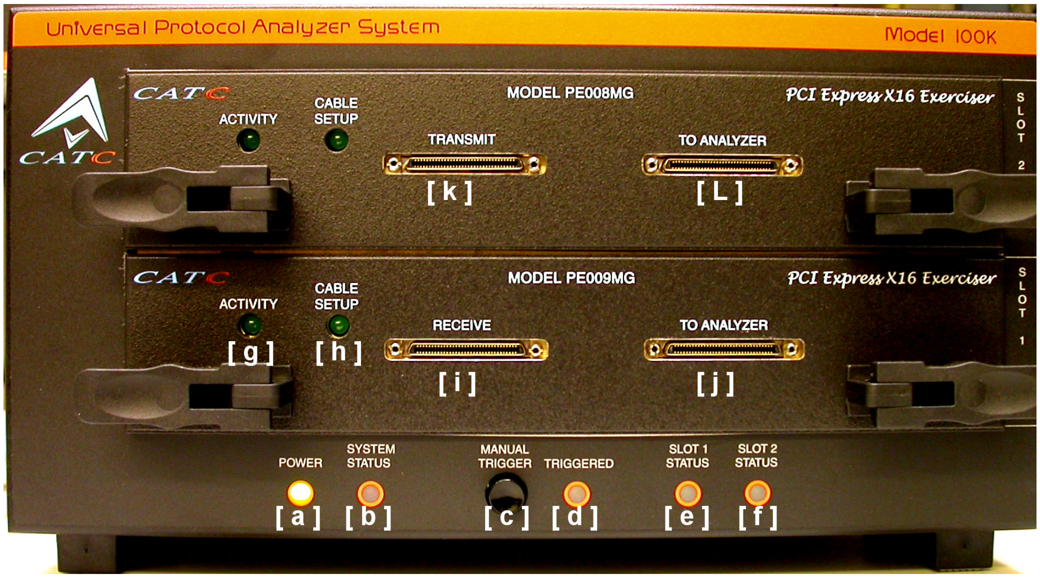
(i) **Probe Data Connector**

2.10 PE*Trainer* EML Exerciser LED Descriptions

When powered on, the PE*Trainer* EML activates the user-accessible controls and LEDs on the front and rear panels of the UPAS.

Warning! Do not open the UPAS enclosure. There are no operator serviceable parts inside. Refer servicing to LeCroy.

UPAS LEDs (from left to right)



(a) **Power** indicator LED for UPAS: Lights when the unit power is switched on.

(b) **System Status** indicator LED for UPAS: Lights during initialization/power up of UPAS base unit. Blinks if a self-test fails.

(c) **Manual Trigger** push-button: Disabled. On PE*Tracer* EML, this button forces an event trigger.

(d) **Triggered** LED: Disabled. On PE*Tracer* EML, this LED lights when the Analyzer triggers on an event.

(e) **Slot 1 Status**: Disabled. On PE*Tracer* EML, this LED illuminates when the Analyzer in Slot 1 is recording.

(f) **Slot 2 Status**: Disabled. On PE*Tracer* EML, this LED illuminates when the Analyzer in Slot 2 is recording.

PE*Trainer* EML Modules

(g) **Activity** LED: On the **Transmit** module, this LED illuminates when linkup is established between the Exerciser and the DUT. On the **Receive** module, this LED illuminates when there is traffic on the link.

(h) **Cable Setup** LED: On the **Transmit** module, this LED illuminates when an connection is established between the Exerciser and host emulator or device emulator.

(i) **Receive** Probe Data Connector: Receives traffic from DUT.

(j) **To Analyzer**: Transmits generated traffic from Exerciser to the **Receive** channel on a PE*Tracer* EML Analyzer. This is an optional connection for users adding a PE*Tracer* EML Analyzer to their setup.

(k) **Transmit** Probe Data Connector: Sends generated traffic from Exerciser to DUT.

(l) **To Analyzer**: Sends traffic received from DUT on the **Transmit** channel on a PE*Tracer* EML Analyzer. This is an optional connection for users adding a PE*Tracer* EML Analyzer to their setup.

2.11 Rear Panel Description

From left to right, the UPAS rear panel contains the following components:

Figure 2.1 PE*Tracer* ML Rear Panel

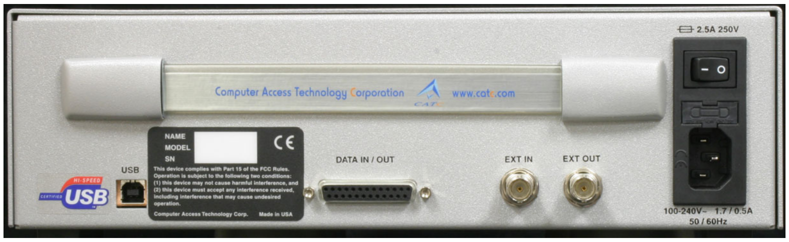
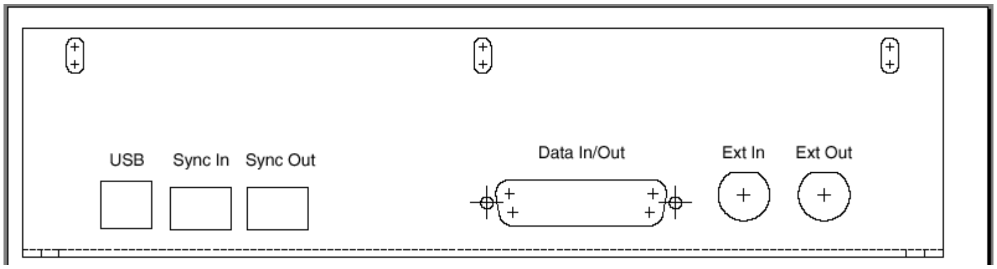

2.5A 250V
Computer Access Technology Corporation
www.catc.com
NAME
MODEL
SN
This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (1) this device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.
Computer Access Technology Corp. Made in USA
USB
HI-SPEED
CERTIFIED
USB
DATA IN / OUT
EXT IN
EXT OUT
100-240V~ 1.7 / 0.5A
50 / 60Hz

Figure 2.2 PE*Tracer* EML Rear Panel



USB Type B Host Computer Connector

This connector links the Analyzer to the Host PC for the purpose of transmitting commands from the PC to the Analyzer and uploading traces from the Analyzer's recording memory to the PE*Tracer* software for viewing and analysis or links the PE*Trainer* Exerciser to the Host PC for the purpose of downloading scripts and controlling the behavior of the Exerciser.

SFP Sync In and Sync Out Connectors

These connectors allow multiple UPAS 100K chassis to send synchronization and control messages to one another.

RS-232 25-pin Data Output Connector

This connector links a 25 pin RS-232 cable to an external Breakout Board. The Breakout Board allows signals to be sent from the Exerciser or Analyzer to an external device such as an oscilloscope or from an external device to the Exerciser or Analyzer for the purpose of triggering on an external input. You configure input/output signalling through the Recording Options dialog box. The Breakout Board use is described at the end of this chapter.

RS-232 40-pin Data Input/Output Connector

This connector links a 40 pin RS-232 cable to an external Breakout Board. The Breakout Board allows signals to be sent from the Analyzer to an external device such as an oscilloscope or from an external device to the Analyzer for the purpose of triggering on an external input. You configure input/output signalling through the Recording Options dialog box. Breakout Board use is described at the end of this chapter. This connector is not used by PE*Trainer* EML.

BNC Connectors Ext. In and Ext. Out

These BNC connectors allow the Analyzer to transmit or receive external signals. External signaling, in turn, can serve two functions:

- **Linking of Analyzers**: On dual Analyzer systems, the BNC connectors are used to create a loop for transmitting clocking information and commands between the Analyzers. This linking allows the two Analyzers to function as a single, logical Analyzer.
- **Triggering**: The **Ext. In** can receive a signal from another device and use that signal to trigger the end of the recording. Conversely, the **Ext. Out** connector can be used to send an output signal from the Analyzer to another device.

The **Ext. In** and **Ext. Out** connectors have the same function as the 25 pin RS-232 connector - i.e., they channel input and output signals but do not support the use of a Breakout Board.

The BNC connectors are not used for the PE*Trainer* Exerciser.

Wide-range AC Connector Module

- Power socket
- Power on/off switch
- Enclosed 5x20 mm 2.0A 250 V fast acting glass fuse

Warning! For continued protection against fire, replace fuse only with the type and rating specified above.

2.12 Specifications for PE*Tracer* Edge

The following specifications describe the standalone PE*Tracer* Edge card.

Package

Dimensions	8.0 x 7.4 x 0.6 inches (20.3 x 13.8 x 1.5 cm)
Connectors	2 Probe Data (HSSDC2 receptacle) UPAS AC power connection External trigger connection (TRIG IN/OUT, SMA) PC connection (USB2.0)
Weight	15 oz. (0.42 kg)

Power Requirements

Adapter	100 to 240 VAC, 47 to 63 Hz (universal input), 60 W maximum
Card	12 V DC, 50 W

Environmental Conditions

Operating Range	0 to 40 °C (32 to 104°F)
Storage Range	-20 to 80 °C (-4 to 176°F)
Humidity	10 to 90%, non-condensing

Probing Characteristics

Connection	Slot interposer card

Recording Memory Size

1 GB for trace capture, timing, and control information.

Host PC Software Requirements

Operating System	Windows® 2000 and XP

Trigger Events

Basic Trigger Events	Error, Link Up/Down, TX1, Ts2, FTS, Any TLP, Config Rd, Config Wr, IO Rd, IO Wr, Mem Rd, Mem Wr, Message, Completion, InitFC1, InitFC2, ACK, NAK, PM
Link Conditions	Link Alive, Electrical Idle Link Condition, TS1 Ordered Set, TS2 Ordered Set, Skip Ordered Set, Electrical Idle Ordered Set, FTS Ordered Set, DLLP, TLP
Payload	Customizable Payload, Mask, Match
Errors	Delimiter, EDB (End-of-Bad Packet), Disparity, Symbol, Idle data, Ordered Set Format

Breakout Board (BOB)

Filter in/out capabilities

Reporting and Statistics

Transport Layer Packet (TLP)	Memory Read 32-bit, Memory Read 64-bit, Memory Write 32-bit, Memory Write 64-bit, I/O Read Request, I/O Write Request, Configuration Read Type 0, Configuration Write Type 0, Configuration Read Type 1, Configuration Write Type 1, Message, Message with Data, Completion, Completion with Data, Completion for Locked Memory Read, Completion for Locked Memory Read with Data, Requesters, Completers, Traffic Class Number, Virtual Channel ID Number
Data Link Layer Packet (DLLP)	Ack, Nak, PM, Vendor, InitFC1-P, InitFC1-NP, Init-FC1-Cpl, Update FC-P, Update FC-NP, Update FC-Cpl, InitFC2-P, InitFC2-NP, Init-FC2-Cpl, Flow Control
Link Transactions	Implicit, Explicit, Incomplete
Split Transaction	Successful Completion, Unsupported Request, Cfg Request Retry, Completer Abort, Incomplete
Error Reports	Invalid 10b code, Running Disparity Error, End of Bad Packet, Delimiter Error, Ordered Set Format Error, Idle Error

2.13 Specifications for PE*Tracer* ML/PE*Trainer* ML

The following specifications describe a combined PE*Tracer* ML *or* PE*Trainer* ML and Universal Protocol Analyzer System (UPAS).

Package

Dimensions	**UPAS 10000**: 12.2 x 12.2 x 3.5 inches (31.1 x 31.1 x 8.9 cm) **PE*Tracer* ML Plug-in:** 9.3 x 6.7 x 1.3 inches (23.6 x 17.0 x 3.2 cm) **PE*Trainer* ML Plug-in**: 9.3 x 6.7 x 1.3 inches (23.6 x 17.0 x 3.2 cm)
Connectors	**PE*Tracer* ML:** 2 Probe Data (HSSDC2 receptacle) **PE*Traine*r ML:** 2 Probe Data (SFF-8470 Multi-lane Copper Connector) UPAS AC power connection External trigger connection (DB-25, TRIG IN/OUT, BNC) PC connection (USB2.0) Breakout Board (type D)
Weight	**UPAS 10000**: 9.5 lbs (4.3 kg) **PE*Tracer* ML Plug-in**: 1.7 lb (0.77 kg) **PE*Trainer* ML Plug-in**: 1.7 lb (0.77 kg)

Power Requirements

90 to 254 VAC, 47 to 63 Hz (universal input), 100 W maximum

Environmental Conditions

Operating Range	0 to 40 °C (32 to 104°F)
Storage Range	-20 to 80 °C (-4 to 176°F)
Humidity	10 to 90%, non-condensing

Switches

Power	On/Off
Manual Trigger	When pressed forces a trigger event

Indicators (LEDs)

UPAS:

Power (PWR)	Illuminated when Analyzer is powered on
Status (STATUS)	Illuminated during Power-On Self Test (POST) and when the Analyzer detects valid on the PCI Express link
Recording (REC)	Illuminated when the Analyzer is actively recording traffic data
Triggered (TRG)	Illuminated when the Analyzer has a valid trigger condition
Uploading (UPLD)	Illuminated when the Analyzer is uploading its recording memory to the Host PC for displaying the CATC Trace and during the memory-testing step of the POST

PE*Tracer* ML Plug-in:

Activity	Illuminated when Analyzer detects valid traffic for the two dual-simplex connections of the PCI Express Link

Probing Characteristics

Connection	**PE*Tracer* ML:** PCI Express Slot Interposer card Mid-Bus probe (for more information about mid-bus probes, refer to the *PETracer Mid-Bus Probe™ Installation Guide*) **PE*Trainer* ML:** Slot Interposer card (Device emulation Interposer) Host Emulation Test fixture (Host emulation)

PE*Tracer* Recording Memory Size

2 GB for trace capture, timing, and control information.

PE*Tracer* Basic Trigger Events

Link Conditions	Link Alive, Electrical Idle Link Condition, TS1 Ordered Set, TS2 Ordered Set, Skip Ordered Set, Electrical Idle Ordered Set, FTS Ordered Set, Compliance Pattern Ordered Set, Comma, DLLP, TLP
TLP Headers	**TLP Type**: Mem, I/O, Cfg, Msg, Cpl, Any Traffic Class number
DLLP Messages	Ack/Nak, InitFC1, InitFC2, UpdateFC, PM, Vendor
Payload	Customizable Payload, Mask, Match
Errors	DLLP CRC, TLP LCRC, TLP ECRC, Delimiter, EDB (End-of-Bad Packet), Disparity, Symbol, Idle data, Skip late, Ordered Set Format

Breakout Board (BOB)

Filter in/out capabilities

PE*Tracer* Reporting and Statistics

Transport Layer Packet (TLP)	Memory Read 32-bit, Memory Read 64-bit, Memory Write 32-bit, Memory Write 64-bit, I/O Read Request, I/O Write Request, Configuration Read Type 0, Configuration Write Type 0, Configuration Read Type 1, Configuration Write Type 1, Message, Message with Data, Completion, Completion with Data, Completion for Locked Memory Read, Completion for Locked Memory Read with Data, Requesters, Completers, Traffic Class Number, Virtual Channel ID Number
Data Link Layer Packet (DLLP)	Ack, Nak, PM, Vendor, InitFC1-P, InitFC1-NP, Init-FC1-Cpl, Update FC-P, Update FC-NP, Update FC-Cpl, InitFC2-P, InitFC2-NP, Init-FC2-Cpl, Flow Control
Link Transactions	Implicit, Explicit, Incomplete
Split Transaction	Successful Completion, Unsupported Request, Cfg Request Retry, Completer Abort, Incomplete
Error Reports	Bad ECRCs, Bad LCRCs, Invalid 10b code, Running Disparity Error, End of Bad Packet, Delimiter Error, TS Data Error, Ordered Set Format Error, Idle Error

Host PC Software Requirements

Operating System	Windows® 2000 and XP

PE*Trainer* Script Memory Size

2 GB

PE*Trainer* Wait Conditions

Time-Based Wait

Wait for User Input

2.14 Specifications for PE*Tracer* EML/PE*Trainer* EML

The following specifications describe a combined PE*Tracer* EML or PE*Trainer* EML and Universal Protocol Analyzer System (UPAS).

Package

Dimensions	**UPAS 100K:** 11.5 x 5.75 x 16.25 inches (29.21x 14.6x 41.275cm) **PE*Tracer* EML Plug-in:** 9.16x 6x 1.25inches (23.26x 15.24x 3.175cm) **PE*Trainer* EML Plug-in module** (2 per platform) 9.16 x 6x 1.25 inches (23.26 x 15.24 x 3.175 cm)
Connectors	**PE*Tracer* EML:** 2 Probe Data **PE*Trainer* EML:** Two Data Probe Connectors (Molex VHDCI) UPAS AC power connection External trigger connection (TRIG IN/OUT, BNC) 2 High Speed Expansion Ports (SFP) (Sync In, Sync Out) PC connection (USB 2.0) Breakout Board (type D)
Weight	**UPAS 100000**: 9.5 lbs (4.3 kg) **PE*Tracer* EML Plug-in**: 1.7 lb (0.77 kg) **PE*Trainer* EML Plug-in**: 1.7 lb (0.77 kg)

Power Requirements

90 to 254 VAC, 47 to 63 Hz (universal input), 100 W maximum

Environmental Conditions

Operating Range	0 to 40 °C (32 to 104°F)
Storage Range	-20 to 80 °C (-4 to 176°F)
Humidity	10 to 90%, non-condensing

Switches

Power	On/Off
Manual Trigger	When pressed forces a trigger event

Indicators (LEDs)

UPAS:	
Power	Illuminated when Analyzer is powered on
System Status	Illuminates during initialization/power up of UPAS. Blinks if Power On Self Test (POST) fails.
Triggered	Illuminates when a trigger event occurs.
Slot 1 Status	Illuminates when the Analyzer in Slot 1 is recording
Slot 2 Status	Illuminates when the Analyzer in Slot 2 is recording
PE*Tracer* EML Plug-in LEDs:	
Activity	Illuminated when the Analyzer is actively recording traffic data
Cable Setup	Illuminates when cable is correctly connected between front of Analyzer and Interposer.

Probing Characteristics

Connection	**PE*Tracer* EML:** PCI Express Slot Interposer **PE*Trainer* EML:** Slot Interposer card/Device Emulation Interposer (end-point emulation) Test Fixture (Root Complex emulation)

PE*Tracer* Recording Memory Size

4 GB per direction for trace capture, timing, and control information.

PE*Tracer* Basic Trigger Events

TLP Headers	**TLP Type**: Mem, I/O, Cfg, Msg, Cpl, Any Traffic Class number
DLLP Messages	Ack/Nak, InitFC1, InitFC2, UpdateFC, PM, Vendor
Link Conditions	Link Alive, Electrical Idle Link Condition, TS1 Ordered Set, TS2 Ordered Set, Skip Ordered Set, Electrical Idle Ordered Set, FTS Ordered Set, DLLP, TLP, Training Control Bits
Payload	Customizable Payload, Mask, Match
Errors	DLLP CRC, TLP CRC, Delimiter, EDB (End-of-Bad Packet), Disparity, Symbol, Idle Data Skip Late DLLP Ordered Set Format

Breakout Board (BOB)

Filter in/out capabilities

PE*Tracer* Reporting and Statistics

Transport Layer Packet	Memory Read 32-bit, Memory Read 64-bit, Memory Write 32-bit, Memory Write 64-bit, I/O Read Request, I/O Write Request, Configuration Read Type 0, Configuration Write Type 0, Configuration Read Type 1, Configuration Write Type 1, Message, Message with Data, Completion, Completion with D for Locked Memory Read with Data, Requesters, Completers, Traffic Class Number, Virtual Channel ID Number
Data Link Layer Packet (DLLP)	Ack, Nak, PM, Vendor, InitFC1-P, InitFC1-NP, Init-FC1-Cpl, Update FC-P, Update FC-NP, Update FC-Cpl, InitFC2-P, InitFC2-NP, Init-FC2-Cpl, Flow Control
Link Transactions	Implicit, Explicit, Incomplete
Split Transaction	Successful Completion, Unsupported Request, Cfg Request Retry, Completer Abort, Incomplete
Error Reports	Bad ECRCs, Bad LCRCs, Invalid 10b code, Running Disparity Error, End of Bad Packet, Delimiter Error, TS Data Error, Ordered Set Format Error, Idle Error

PE*Trainer* Script Memory Size

4 GB for trace generation, timing and control information

Host PC Software Requirements

Operating System	Windows® 2000, ME, XP

PE*Trainer* Wait Conditions

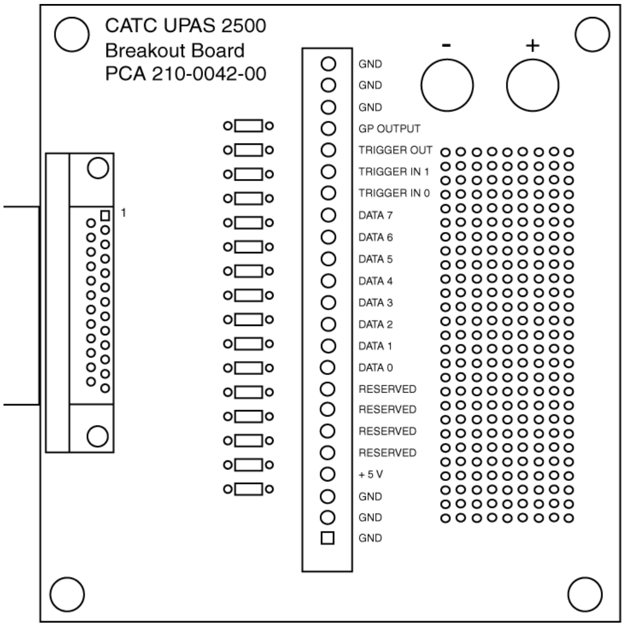
Time-Based Wait

Wait for User Input

Wait for DLLP, TLP, or Ordered Set

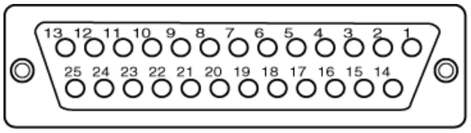
2.15 External Interface Breakout Board

With each Analyzer, LeCroy includes an External Interface Breakout Board for accessing several potentially useful standard, LV TTL output and input signals. The Breakout Board also offers a simple way to connect logic Analyzers or other tools to the PE*Tracer* ML or PE*Tracer* EML Analyzer unit. Six ground pins and one 5-volt pin are provided.



The Breakout Board connects via a cable to the **Data In/Out** connector located on the rear of the Analyzer unit. Each signaling pin is isolated by a 100 Ω series resistor and a buffer inside the Analyzer unit.

Data In/Out Connector (on cable)



Pin-Outs for the Data In/Out Connector

The following table lists the pin-out and signal descriptions for the **Data In/Out** connector on a cable that connects to the Breakout Board.

Data In/Out Connector – Pin-Out

Pin	Signal Name	Signal Description
1	RSV	Reserved
2	GND	Ground
3	GP OUT	General Purpose Output
4	TRG IN 1	Trigger In 1
5	GND	Ground
6	DATA 6	Data 6
7	DATA 4	Data 4
8	DATA 3	Data 3
9	DATA 1	Data 1
10	GND	Ground
11	RSV	Reserved
12	RSV	Reserved
13	+5V	+5 Volts, 250 mA DC Source
14	RSV	Reserved
15	GND	Ground
16	TRG OUT	Trigger Out
17	TRG IN 0	Trigger In 0
18	DATA 7	Data 7
19	DATA 5	Data 5
20	GND	Ground
21	DATA 2	Data 2
22	DATA 0	Data 0
23	GND	Ground
24	RSV	Reserved
25	RSV	Reserved

Note: (*) Pins 4 and 17 have the same function: they allow external signals to be used to cause triggering or recording. Pins 3 and 16 are used to transmit output signals. Pins 6, 7, 8, 9, 18, 19, 21, and 22 (data pins) are used to define data patterns for external input signals.

Note: All models of PE*Tracer* only support Data 0 - Data 3.

Prototype Rework Area

The Breakout Board contains a prototype rework area for making custom circuits for rapid development. The area consists of plated-through holes, 20 columns wide by 27 rows long. The top row of holes is connected to GND and the bottom row is connected to +5V. The remaining holes are not connected. Use the rework area to insert custom components and wire-wrap their respective signal, power, and ground pins.

Breakout Board Input and Output Signals

A Breakout Board can be used to channel input signals into the Analyzer in order to cause triggering. A Breakout Board can also be used to channel signals from the Analyzer to an external source.

Drive strength for all outputs is about 30 mA high (@2 V) and 60 mA low (@0.5 V). Inputs can handle 0 V to 5.5 V. Inputs above 2 V are detected as logic high; inputs below 0.8 V are detected as logic low.

The Breakout Board connects via a cable to the Data In/Out connector on the rear of the UPAS.

External Recordable Signals

Breakout Board Data 0-3: These pins let you define a 4-bit Data Pattern that can be recorded in a trace file.

External Input Triggers

You can use either an external input signal or the Trigger button on the front of the UPAS to cause triggering. The following descriptions show what pins or buttons to use:

Breakout Board Data 3 - Data 0: Triggers on a 4-bit input pattern.

Breakout Board Trigger In 0: Selectable Edge triggered inputs. Triggers on any edge it detects.

Breakout Board Trigger In 1: Selectable Edge triggered inputs. Triggers on any edge it detects.

BNC Trigger (EXT IN): Selectable Edge triggered inputs. Triggers on any edge it detects. Located on the back of the chassis.

Push Button Trigger: The Trigger button on the front of the UPAS can be pressed to manually cause a trigger.

External Outputs

The Analyzer can be configured to send an external signal anytime a trigger and/or event occurs. The following descriptions show the behavior of these output signals:

Breakout Board Trigger Out: Changes from low to high when Analyzer triggers
(one time per recording only)

Breakout Board G.P. Output: Programmable waveform (low or high pulse, toggle). Each event can be programmed to enable this external signal.

BNC Output (EXT OUT): Same as Breakout Board G.P. Output. Located on the back of the chassis.

Setting Recording Options to Support External Input/Output Signaling

After a Breakout Board has been connected to a UPAS, the Analyzer must be configured for external or internal input signaling.

Chapter 3: Installation

PE*Tracer* Edge is a stand-alone card with an external 12-volt power supply.

PE*Tracer*™ ML™ or PE*Trainer*™ ML™ is a factory-installed hardware module that is sold as part of the Universal Protocol Analyzer System™ 10000 (UPAS™ 10K).

PE*Tracer* EML™ is a factory-installed hardware module that is sold as part of the Universal Protocol Analyzer System 100K (UPAS 100K).

PE*Trainer* EML™ consists of two factory-installed hardware modules that are sold as part of the Universal Protocol Analyzer System 100K (UPAS 100K).

The UPAS 10K, UPAS 100K, modules, and associated software are easily installed. You can begin making PCI Express™ recordings or begin traffic generation after following the steps in this chapter.

3.1 Setting Up the PE*Tracer* Edge Card

Step 1 Remove the card from its shipping container.

Step 2 Connect the USB cable between the USB port on the card and a USB port on the PC.

Step 3 Connect the external 12-volt power supply to the Analyzer unit.

Step 4 Connect the external 12-volt power supply to a 100-volt to 240-volt, 50 Hz to 60 Hz, 60 W power outlet. This turns on the Analyzer, which has no power switch.

Step 5 Follow on-screen Plug-and-Play instructions for the automatic installation of the Analyzer as a USB device on the Host PC (the required USB files are included on the installation CD). Step through the Windows® hardware wizard. If the wizard prompts you for driver information, point it to the CD which should be in your disk drive.

The host operating system detects the Analyzer and begins to install the USB driver.

3.1 Setting Up the PE*Tracer* Analyzer

Step 1 Remove the Analyzer/UPAS from its shipping container. The module is already installed in the UPAS.

Step 2 Connect the Analyzer unit to a 100-volt to 240-volt, 50 Hz to 60 Hz, 120 W power outlet using the provided power cord.

Note: The Analyzer is capable of supporting supply voltages between 100 volts and 240 volts, 50 Hz or 60 Hz, thus supporting all known supply voltages around the world.

Step 3 Connect the USB cable between the USB port on the back of the Analyzer and a USB port on the PC.

Step 4 Turn on the power switch on the rear of the UPAS.

Note: At power-on, the Analyzer initializes itself in approximately five seconds and performs an exhaustive self-diagnostic that lasts about fifteen seconds. The Status LED of the UPAS base unit turns red on power up/initialization. The LED remains on approximately 20 seconds while the Analyzer performs self-diagnostic testing. If the diagnostics fail, the Status LED blinks red, indicating a hardware failure. If this occurs, call LeCroy Customer Support for assistance.

Step 5 Follow on-screen Plug-and-Play instructions for the automatic installation of the Analyzer as a USB device on the Host PC (the required USB files are included on the CD). Step through the Windows® hardware wizard. If the wizard prompts you for driver information, point it to the CD which should be in your disk drive.

The host operating system detects the Analyzer and begins to install the USB driver.

3.2 Setting Up the PE*Trainer* Exerciser

Step 1 Remove the Exerciser and UPAS from its shipping container. The PE*Tracer* module is already installed in the UPAS.

Step 2 Connect the Exerciser to a 100-volt to 240-volt, 50 Hz to 60 Hz, 120 W power outlet using the provided power cord.

Note: The Exerciser is capable of supporting supply voltages between 100-volt and 240-volt, 50 Hz or 60 Hz, thus supporting all known supply voltages around the world.

Step 3 Connect the USB cable between the USB port on the back of the UPAS and a USB port on the PC.

Note: All hardware and cables should be setup prior to power-up. Install Device Emulator or Host Emulator as described in the previous sections. Also install any cables between the Emulator and PE*Trainer* EML Exerciser as described in the previous sections.

Step 4 Turn on the power switch on the rear of the UPAS.

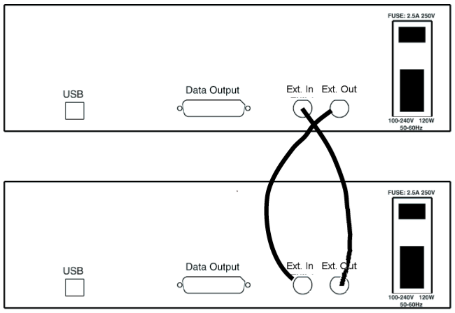
Note: At power-on, the Exerciser initializes itself in approximately five seconds and performs an exhaustive self-diagnostic that lasts about fifteen seconds. The Status LED of the UPAS base unit turns red on power up/initialization. The LED remains on approximately 20 seconds while the Exerciser performs self-diagnostic testing. If the diagnostics fail, the Status LED blinks red, indicating a hardware failure. If this occurs, call LeCroy Customer Support for assistance.

Step 5 Follow on-screen Plug-and-Play instructions for the automatic installation of the Exerciser as a USB device on the Host PC (the required USB files are included on the CD). Step through the Windows® hardware wizard. If the wizard prompts you for driver information, point it to the CD which should be in your disk drive.

The host operating system detects the Exerciser and begins to install the USB driver.

3.3 Installing a Dual Analyzer System

If you have purchased a dual Analyzer system, you must physically link the Analyzers together. You link the Analyzers via their BNC connectors on the back of the two UPASs. You connect the **Ext. Out** connector of one Analyzer to the **Ext. In** connector on the other Analyzer, and *vice versa*, thereby creating a loop. The Analyzers use this loop to transmit clocking information and commands from one Analyzer to the other. From a user point of view, the interaction between the two Analyzers is transparent and requires no user intervention other than the physical linking of the Analyzers.

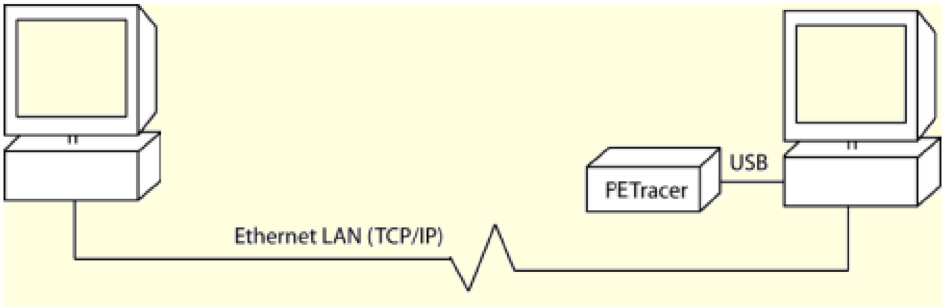


3.4 Networking PE*Tracer*

PE*Tracer*™ has a networking capability that allows users to run an Analyzer remotely over an IP-based LAN. Using the Network browse dialog, you can locate and connect to other PC hosts on the LAN, which, in turn, are connected to Analyzers. Through this connection, you can remotely control an Analyzer.

Setup for IP LAN Use

In order to connect to a remote Analyzer, the Analyzer must be attached to a PC that is on the LAN. This PC must have PE*Tracer* installed.



3.5 Installing the PE*Tracer* Software

PE*Tracer* software operates all of LeCroy's PCI Express protocol Analyzer and Exerciser products:

Exercisers

- PE*Trainer* ML
- PE*Trainer* EML

Analyzers

- PE*Tracer* Edge
- PE*Tracer* ML
- PE*Tracer* EML

The PE*Tracer* software is installed on a Microsoft® Windows®-based PC and serves as the interface for the Exerciser and/or Analyzer. When an Exerciser is attached, traffic generation features are enabled.

After the Analyzer or Exerciser has been recognized as a USB device, install the PE*Tracer* software on the Host PC:

Step 1 Insert the CD into the CD ROM drive of the PC that controls the Analyzer or Exerciser. The installation window opens and displays links to the PE*Tracer* software, user manuals, a readme file, and Acrobat Reader.

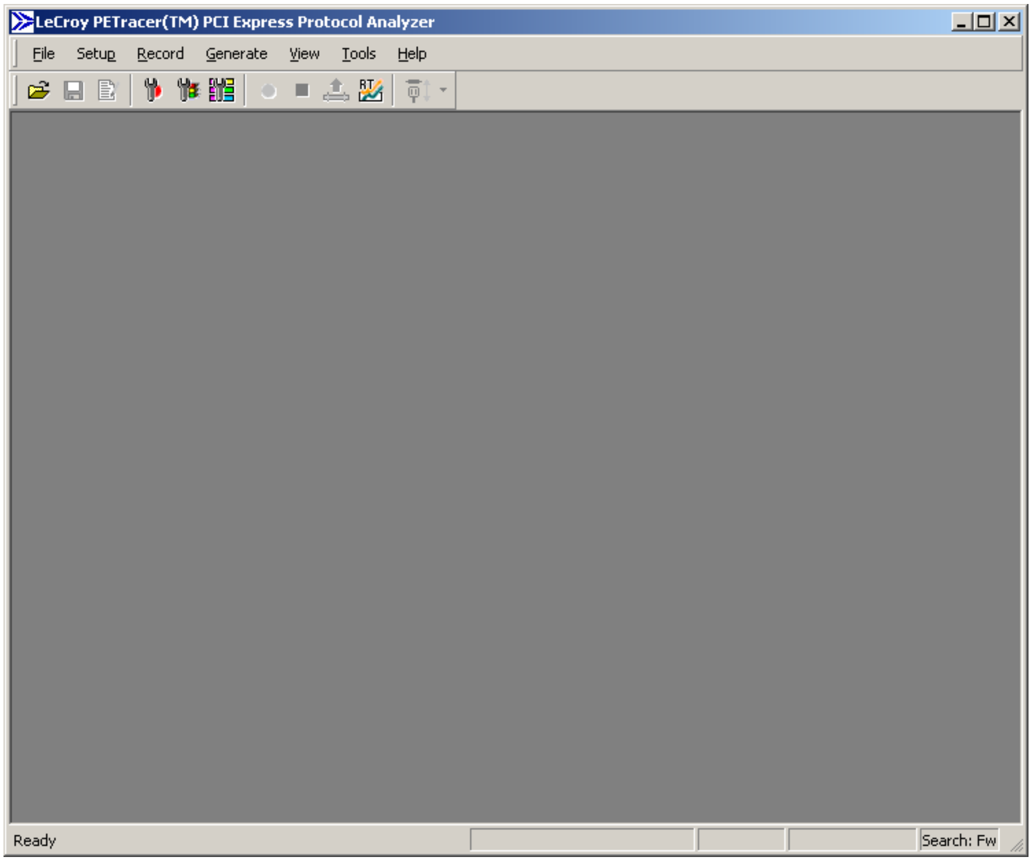
Step 2 Select **Install Software** and follow the onscreen instructions.

The PE*Tracer* software installs automatically on the PC hard disk. During installation, all necessary USB drivers are loaded automatically. Drivers included in the installation are:

- **ctcupa10.sys**: UPAS 10K driver
- **upas100k.sys**: UPAS 100K driver, PE*Tracer* EML and PE*Trainer* EML driver
- **petrcedg.sys**: PE*Tracer* Edge driver
- **petrcrml.sys**: PE*Tracer* ML driver
- **petranx4.sys**: PE*Trainer* ML driver

Step 3 To start the application, launch the PE*Tracer* program from the Start menu:
Start > Programs > LeCroy > PETracer > LeCroy PETracer

The PE*Tracer* program opens:


LeCroy PETracer(TM) PCI Express Protocol Analyzer
File Setup Record Generate View Tools Help
Ready
Search: Fw

Note: The software may be used with or without the Exerciser or Analyzer. When used without an Exerciser or Analyzer attached to the computer, the program functions as a Trace Viewer to view, analyze, and print captured traffic.

3.6 Connecting the PE*Tracer* Edge Analyzer to the Device Under Test

Installing the Edge Card between the Devices Under Test (DUT)

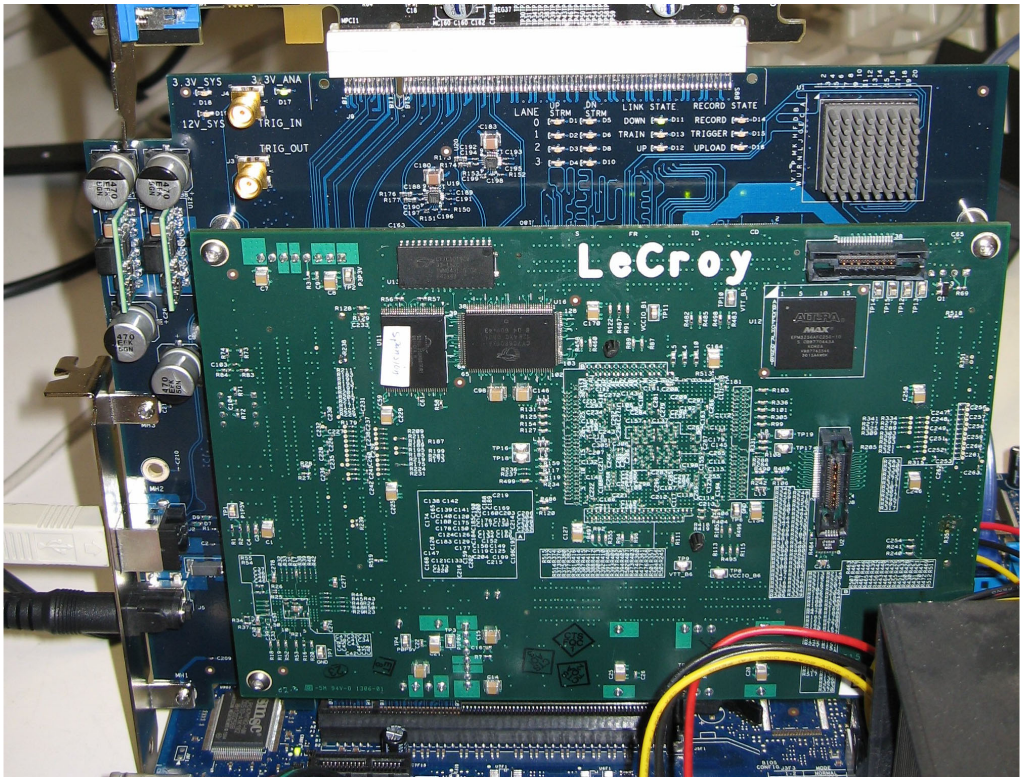
To install the Edge card between the Devices Under Test (DUT), perform the following steps:

Step 1 If not already powered off, unplug the external 12-volt power supply from the power outlet and power off the DUTs.

Step 2 Insert the gold male connector on the Edge card into a PCI Express slot in the motherboard (or whatever board you are testing).

Step 3 Insert the PCI Express DUT into the white female device connector on the top edge of the Edge card. The Edge card's female device connector can accommodate an x1, x2, or x4 PCI Express device.

Step 4 The Edge card is shipped from LeCroy with a metal face plate for attachment to a PC case. If you are working with a motherboard that is not in a PC case, you may prefer to remove the metal face plate so the Edge card can sit flat with the motherboard. To remove the face plate, unscrew the two screws that hold it onto the Edge card.



Connect USB Cable and Power on the Analyzer and DUTs

Step 1 If not already connected, connect the provided USB cable between the USB port on the Edge card and a USB port on the Host PC that runs the PE*Tracer* software.

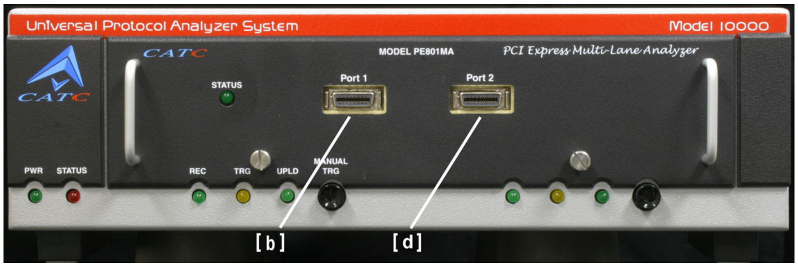
Step 2 If not already connected, connect the external 12-volt power supply to the Edge card.

Step 3 Connect the external 12-volt power supply to a 100-volt to 240-volt, 50 Hz to 60 Hz, 60 W power outlet. This turns on the Analyzer, which has no power switch.

Step 4 After 20 seconds, power on the PCI Express device under test.

Step 5 Open the PE*Tracer* software on the PC host system. The Edge card is now ready for PCI Express traffic recording.

3.7 Connecting the PE*Tracer* ML Analyzer to the Device Under Test

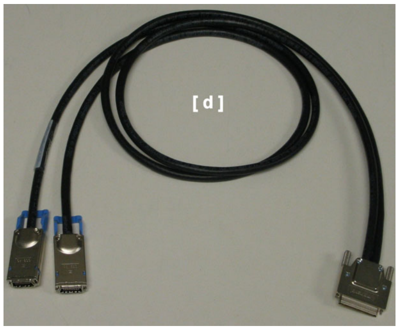


PE*Tracer* ML Components

(a) PE*Tracer* ML Probe Data connector

(b) PE*Tracer* ML Probe Data connector

(c) Probe Data cable: Cable with four connectors. The four-connector cable supports x8, x4, x2, and x1 links. The small connectors connect to the front of the Analyzer(s). The large connector at the other end connects to the Interposer.

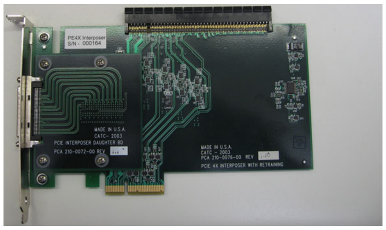


(d) Probe Data cable: Cable with two connectors. The two-connector cable supports x4, x2, and x1 links. The small connectors connect to the front of the Analyzer(s). The large connector at the other end connects to the Interposer.

(e) PE*Tracer* ML Slot Interposer Probe: The probe has two connectors: a gold male connector that fits into a standard PCI Express slot, and a black female connector that accommodates a PCI Express device. There are two versions of the probe, an x8 probe and a x4 probe. The gold male adapter on the x4 probe is half the length of the x8.

Black Female Connector for a Device

PE*Tracer* Cable Connector To Analyzer



Edge Connector (to Motherboard)

Installing the Interposer between the Devices Under Test (DUTs)

To install the Interposer between the Devices Under Test (DUT), perform the following step:

Step 1 If not already powered off, power off the UPAS 10000 Analyzer and the DUTs.

Step 2 Insert the gold male connector on the Interposer probe [e] into a PCI Express slot in the motherboard (or whatever board you are testing). The x4 Interposer is designed to fit tightly into any standard x4, x8, or x16 PCI Express slot. The x8 Interposer is designed to fit into any x8 or x16 PCI Express slot.

Step 3 Insert the PCI Express DUT into the black female device connector on the top edge of the Interposer. The Interposer's female device connector can accommodate any standard PCI Express device (x1, x4, x8, x16).

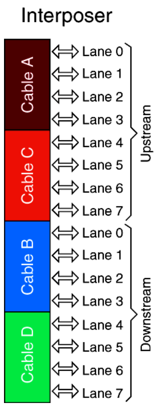
Step 4 The slot Interposer is shipped from LeCroy with a metal face plate for attachment to a PC case. If you are working with a motherboard that is not in a PC case, you may prefer to remove the metal face plate so the Interposer can sit flat with the motherboard. To remove the face plate, unscrew the two screws that hold it onto the Interposer.

Connecting the Probe Data Cable

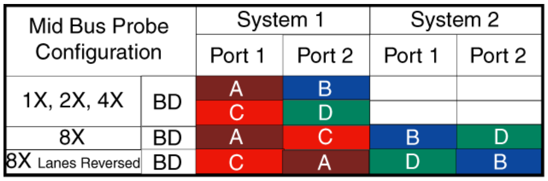
The next step is to connect the Probe Data cable. How you connect the cable depends on the width of your connection.

Probe Data Cable Description

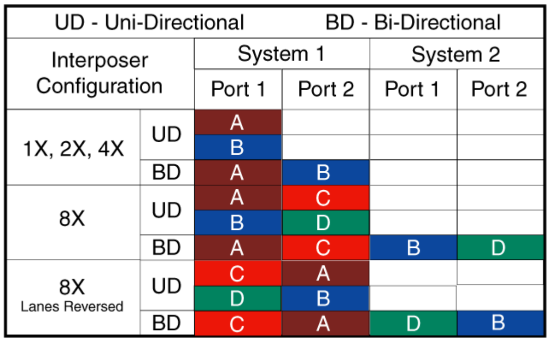
The Probe Data cable consists of a single, large connector on one end and two or four smaller, 16 pin connectors on the other end. The large connector attaches to the Interposer, the smaller connectors attach to the front of the Analyzer.



The smaller connectors are labeled A, B, C, and D. Each transmits four physical lanes of traffic in one direction (shown in the illustration on the right). This means that to record bidirectional traffic you must use at least two of the small connectors: one to capture the transmit traffic and one to capture the receive traffic (with respect to one of the devices). The various configurations of connectors and link widths are shown in the table below.



Mid Bus Probe Configuration		System 1		System 2	
		Port 1	Port 2	Port 1	Port 2
1X, 2X, 4X	BD	A	B		
		C	D		
8X	BD	A	C	B	D
8X Lanes Reversed	BD	C	A	D	B



UD - Uni-Directional BD - Bi-Directional

Interposer Configuration		System 1		System 2	
		Port 1	Port 2	Port 1	Port 2
1X, 2X, 4X	UD	A			
		B			
	BD	A	B		
8X	UD	A	C		
		B	D		
	BD	A	C	B	D
8X Lanes Reversed	UD	C	A		
		D	B		
	BD	C	A	D	B

Connecting the Interposer Data Cable
To connect the Interposer data cable:

Step 1 Insert the large connector of the Interposer data cable [c] or [d] into the Interposer’s data connector located on the metal face plate of the Interposer [e].

Step 2 Connect the small connectors of the Interposer data cable [c] or [d] to the Analyzer probe data connectors [a] and, if needed, [b] on the front of the Analyzer. The number of connectors you use depends on the width and direction of the link you are attempting to monitor. See foregoing table to determine the appropriate number of connectors for your test.

Connect USB Cable and Power on the Analyzer and DUTs

Step 1 Connect the provided USB cable between the UPAS 10000 Analyzer and the PC host system that runs the PE*Tracer* ML software.

Step 2 Power on the UPAS10000 Analyzer. The Analyzer's green power LED lights, and the red status LED turns on for approximately 20 seconds while the Analyzer performs self-diagnostics.

Step 3 After the Analyzer's red status LED turns off, power on the PCI Express system under test.

Step 4 Open the PE*Tracer* ML software on the PC host system. The Analyzer is now ready for PCI Express traffic recording.

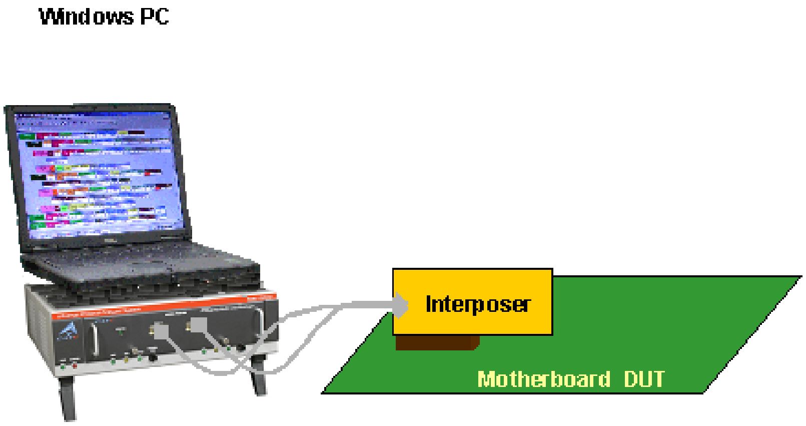
3.8 Connecting PE*Trainer* ML to the Device Under Test

PE*Trainer* ML can test both the host and device sides of a PCI Express link through the use of two types of adapter:

- **Motherboards and host controllers**: Using a **device emulation Interposer**. A device emulation Interposer is a PCI Express adapter card that fits into slotted PCI Express devices.
- **PCI Express add-on cards**: Using a **host emulation test fixture**. A host emulation test fixture is a box-like adapter with a slot for testing PCI Express cards.

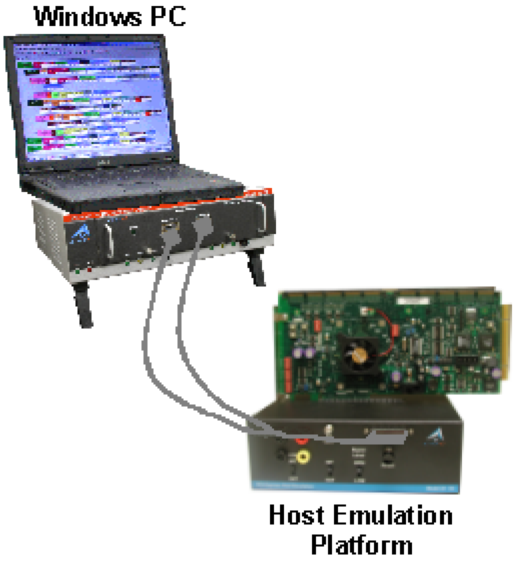
Both of these test devices can be purchased from LeCroy.

Device Emulator Interposer - Description



The LeCroy Device emulation Interposer is an adapter that provides a way of connecting a PE*Trainer* ML Exerciser or PE*Tracer* ML Analyzer to a PCI Express motherboard. The Interposer has two connectors: a connector for a data cable and an edge connector for inserting the Interposer into a slotted DUT.

Host Emulation Test Fixture - Description



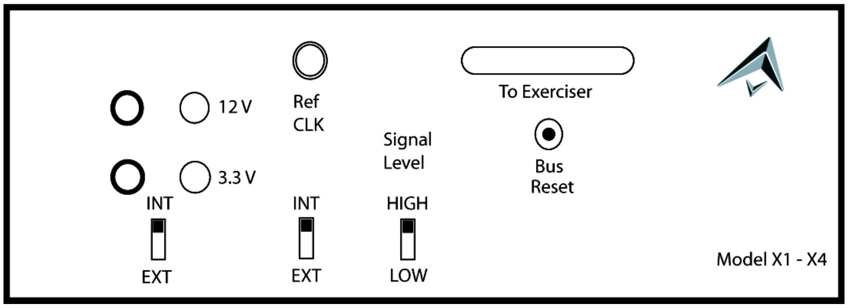


Figure 3.3 Host Emulation Test Fixture

LeCroy Host Emulation Test Fixture: Hardware enclosure with a slot on top for accommodating a PCI Express card.

Data Cable Connector (**To Exerciser**): To connect to the PE*Trainer* ML and/or PE*Tracer* ML Analyzer.

Bus Reset button: Reset asserts PERST # for > 250 ms

Signal Level switch:

- Hi = PCI Express compliant (0.9V - 1.2V)
- Low = Testing (0.45V - 0.65V)

Power Switch: Selects between the internal power supply and the banana jacks on the front panel of the unit. When **External** power is selected, both 3.3V and 12V supplies must be provided by the user.

Reference Clock: Selects between the internal supplied 100 MHz reference or the user supplied reference via the **RefCLK** SMA connector.

Installing the Device Emulation Interposer

To install the Interposer into the DUT, perform the following steps.

Step 1 If not already powered off, power off the PE*Trainer*/UPAS 10K system and the DUT.

Step 2 Insert the edge connector on the Device Emulation Interposer into a PCI Express slot in the DUT. The Interposer is designed to fit tightly into any standard x4, x8, or x16 PCI Express slot.

Step 3 The slot Interposer is shipped from LeCroy with a metal face plate for attachment to a PC case. If you are working with a motherboard that is not in a PC case, you may prefer to remove the metal face plate so the Interposer can sit flat with the motherboard. To remove the face plate, unscrew the two screws that hold it onto the Interposer.

Step 4 Attach probe data cable as described below in “Probe Data Cable Description” on page 52.

Installing the Host Emulation Test Fixture

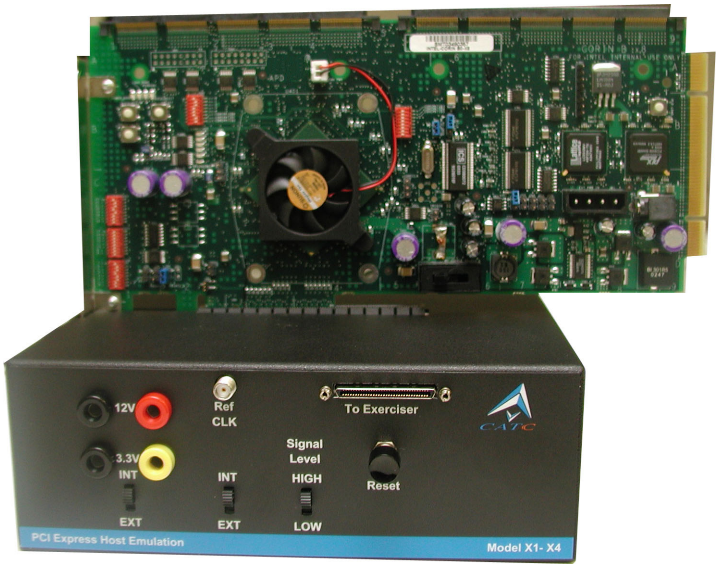


Figure 3.4 Host Emulation Test Fixture with DUT Card on Top

Step 1 If not already powered off, power off the PE*Trainer*/UPAS 10K system and the DUT.

Step 2 Insert the male edge connector of the DUT into the PCI Express slot on top of the Host Emulation Test Fixture.

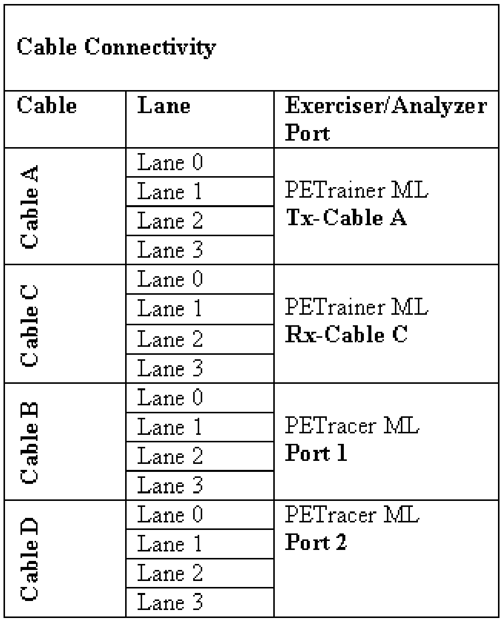
Step 3 Attach probe data cable as described in the next section.

Probe Data Cable

The probe data cable is used to transfer x1 and x4 link PCI Express data to and from the Exerciser. The cable has a single wide connector on one end for connecting to the Interposer or the Host Emulation Test Fixture. At the cable's other end are four smaller connectors labeled A, B, C, and D. These connectors attach to ports on the front of a PE*Trainer* ML or PE*Tracer* ML system.

Each of the small connectors can only transmit in one direction. This means that at a minimum, two connectors are needed: one to transmit and one to receive.

Each connector transmits four physical lanes of unidirectional traffic. The lanes and port assignments are shown in the illustration below.



Cable Connectivity

Cable	**Lane**	**Exerciser/Analyzer Port**
Cable A	Lane 0	PETrainer ML **Tx-Cable A**
	Lane 1	
	Lane 2	
	Lane 3	
Cable C	Lane 0	PETrainer ML **Rx-Cable C**
	Lane 1	
	Lane 2	
	Lane 3	
Cable B	Lane 0	PETracer ML **Port 1**
	Lane 1	
	Lane 2	
	Lane 3	
Cable D	Lane 0	PETracer ML **Port 2**
	Lane 1	
	Lane 2	
	Lane 3	

Cable A, for example, transmits Lanes 0, 1, 2, and 3.

Attaching the Probe Data Cable to PE*Trainer* ML

Connector **A** on the probe data cable goes to port **Tx Cable A** and connector **C** goes to port **Rx Cable C** on the Exerciser.

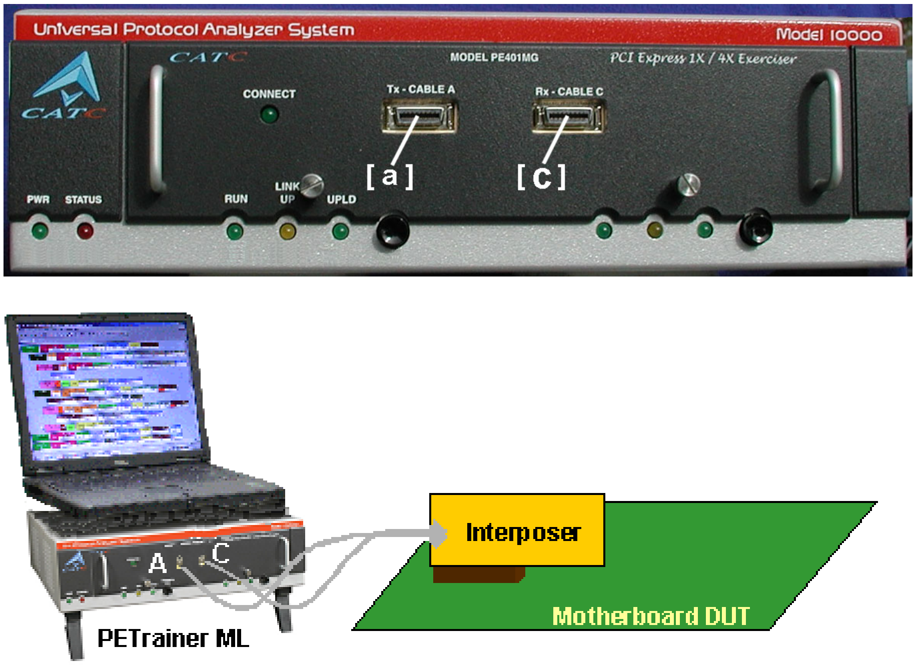


Figure 3.5 PE*Trainer* ML Front Panel showing where Connectors A and C Attach

Add a PE*Tracer* ML to Device Emulation Test Setup

To add a PE*Tracer* ML to the above setup:

Step 1 Attach connectors A and C to the Exerciser as described above.

Step 2 Attach connectors B to **Port 1** and connector D to **Port 2** on the PE*Tracer* ML Analyzer.

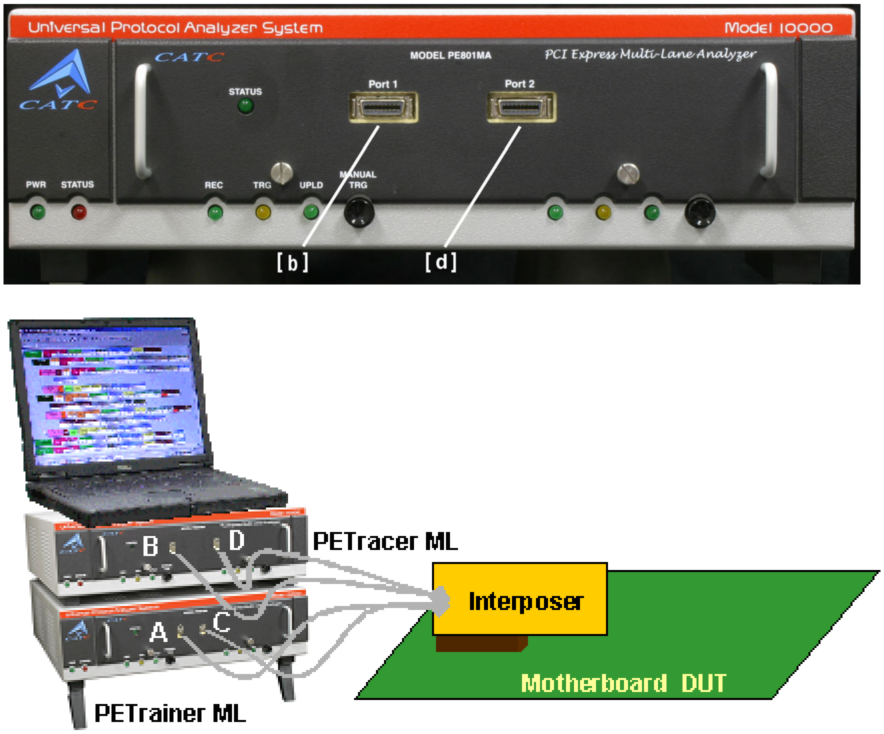


Figure 3.6 PE*Tracer* ML Front Panel showing where Connectors B and D Attach

Add a PE*Tracer* ML to Host Emulation Test Setup

To add a PE*Tracer* ML to a Host Emulation setup:

Step 1 Attach connectors A and C to the Exerciser as described above.

Step 2 Attach connectors D to **Port 1** and connector **D** to **Port 2** on the PE*Tracer* ML Analyzer.

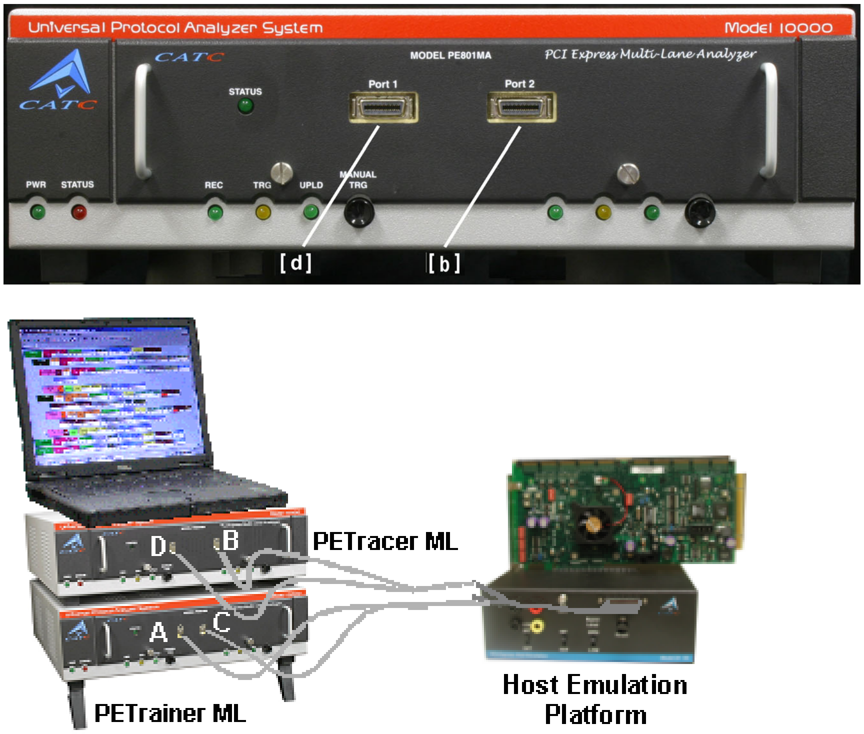


Figure 3.7 PE*Tracer* ML Front Panel showing where Connectors B and D Attach

Final Steps: Connect USB Cable and Power On the Exerciser and DUT

Step 1 Connect the provided USB cable between the UPAS 10K/PE*Trainer* ML and the PC host system that runs the PE*Tracer* software.

Step 2 Power on the PCI Express system under test.

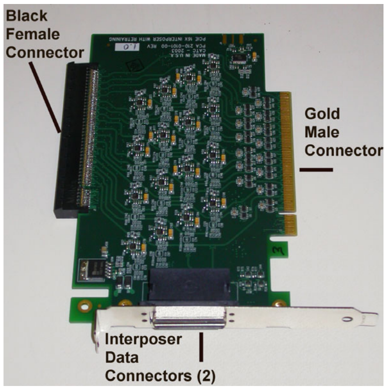
Step 3 Power on the UPAS 10K/Exerciser. The Exerciser's green power LED lights, and the red status LED turns on for approximately 20 seconds while the Exerciser performs self-diagnostics.

3.9 Connecting the PE*Tracer* EML Analyzer to the Device Under Test

PE*Tracer* EML Components

PE*Tracer* EML components used in the installation are:

- **Probe Data Cable (2)**: Each cable supports x16 data from one direction.
- **PE*Tracer* EML x16 Slot Interposer**: The slot Interposer provides the point of attachment for the Analyzer to the Device Under Test (DUT). The Interposer is designed to fit between a motherboard and one of its cards - for example, a LAN card. The Interposer has three sets of connectors: a gold male connector that fits into a standard x16 PCI Express slot on a motherboard, a black female connector that accepts a x16 PCI Express device such as a graphics card, and two Interposer cable connectors that connect the Interposer to the Analyzer.



- **USB Cable**: Connects the UPAS 100K to the host PC running the PE*Tracer* software.

Installing the Interposer

To install the Interposer, perform the following steps.

Step 1 If not already powered off, power off the UPAS 100K Analyzer and the DUTs.

Step 2 Insert the gold male connector on the Interposer probe into a x16 PCI Express slot in the motherboard.

Step 3 Insert the PCI Express DUT into the Interposer’s black female device connector.

Step 4 The slot Interposer is shipped from LeCroy with a metal face plate for attachment to a PC case. If you are working with a motherboard that is not in a PC case, you may prefer to remove the metal face plate so the Interposer can sit flat with the motherboard. To remove the face plate, unscrew the two screws that hold it onto the Interposer.

Connecting the Probe Data Cable and USB Cable

To connect the Interposer data cable:

Step 1 Connect the two probe data cables to the Interposer card and to the Analyzer. It does not matter which slot you plug the cables into on the Analyzer as the direction of the data is determined automatically by the software.

Step 2 Connect the provided USB cable between the UPAS 100K Analyzer and the PC host system that runs the PE*Tracer* software.

Power On Analyzer and DUT

Step 1 Power on the UPAS 100K. The Analyzer's green Power LED lights, and the red Status LED turns on for approximately 20 seconds while the Analyzer performs self-diagnostics.

Step 2 After the Analyzer's red status LED turns off, power on the PCI Express system under test.

Step 3 Open the PE*Tracer* EML software on the PC host system. The Analyzer is now ready for PCI Express traffic recording.

3.10 Connecting PE*Trainer* EML to the Device Under Test

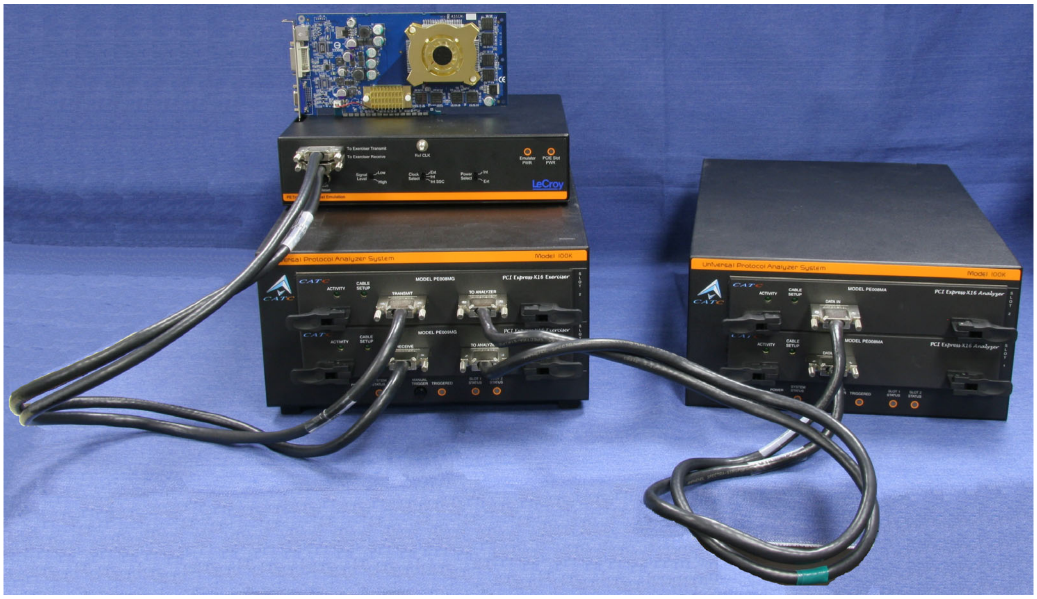


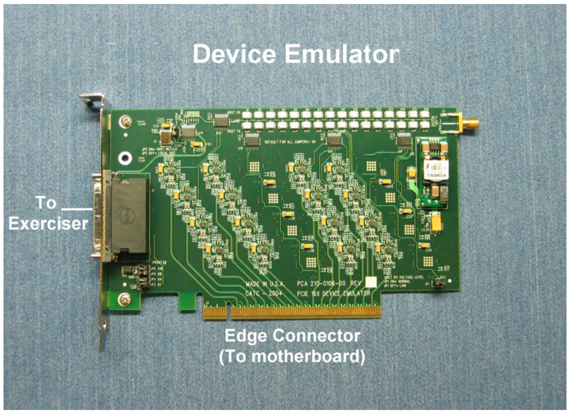
Figure 3.8 PE*Trainer* EML (bottom left) connected to a Host Emulator adapter (top) and a PE*Tracer* EML Analyzer (right). The Host Emulator has a PCI Express graphics card inserted.

PE*Trainer* EML can test both the host and device sides of a PCI Express link through the use of two types of adapter:

- **Motherboards and host controllers**: Using a **device emulator.** A device emulator is an adapter card that fits into motherboards and other slotted PCI Express devices.
- **PCI Express add-on cards**: Using a **host emulation test fixture**. A host emulation test fixture is a box-like adapter with a slot for testing PCI Express cards.

Both of these test devices can be purchased from LeCroy.

Device Emulator - Description



The LeCroy Device Emulator is an adapter that provides a way of connecting a PE*Trainer* EML Exerciser to a PCI Express motherboard. The emulator has two connectors: a connector for a data cable and an edge connector for inserting the Device Emulator into a slotted DUT.

Installing the Device Emulator

To install the Device Emulator into the DUT, perform the following steps:

Step 1 If not already powered off, power off the PE*Trainer*/UPAS 100K system and the DUT. The PE*Trainer* power switch is located on the back of the UPAS.

Step 2 Check the Jumper settings. The default is to have all jumpers installed on the device emulator. The jumpers can be used to modify the following:

- **JP1 (Host RX Voltage Level):** This jumper can be removed to reduce the differential voltage level that the DUT receives from the PE*Trainer* to (0.45V - 0.65V). When installed (default) the differential voltage level is (.9V - 1.2V).
- **JP2 (Refclk):** This jumper can be removed for systems that do not provide a 100MHz RefClk to the PCI Express slot. When installed (default), the PE*Trainer* uses the RefClk that is provided by the DUT.
  Note: This jumper MUST be installed if the DUT is providing a Spread Spectrum enabled RefClk.
- **JP3-JP6 (PRSNT2#):** These jumpers can be used to open circuit the Presence detect signals driven to the PCI Express connector. When removed, the PRSNT2# signal is floating. When installed (default), the PRSNT2# signal is tied to GND. There are four PRSNT2# signals on the PCI Express connector, one for each lane width (x1, x4, x8, and x16).

Step 3 Insert the edge connector on the Device Emulator into a PCI Express slot in the DUT. The Device Emulator is designed to fit tightly into any standard x16 PCI Express slot. To connect to a x1, x4, or x8 slot, you must install a slot reducer, available from LeCroy.

Step 4 The Device Emulator is shipped from LeCroy with a metal face plate for attachment to a PC case. If you are working with a motherboard that is not in a PC case, you may prefer to remove the metal face plate so the emulator can sit flat with the motherboard. To remove the face plate, unscrew the two screws that hold it onto the emulator.

Step 5 Attach probe data cables between the two connectors on the emulator and the **Transmit** and **Receive** ports on the Exerciser.

Step 6 At this point the emulator is ready for use. Skip ahead to Section , “Final Steps: Power On the Exerciser and DUT” on page 67.

Host Emulation Test Fixture - Description

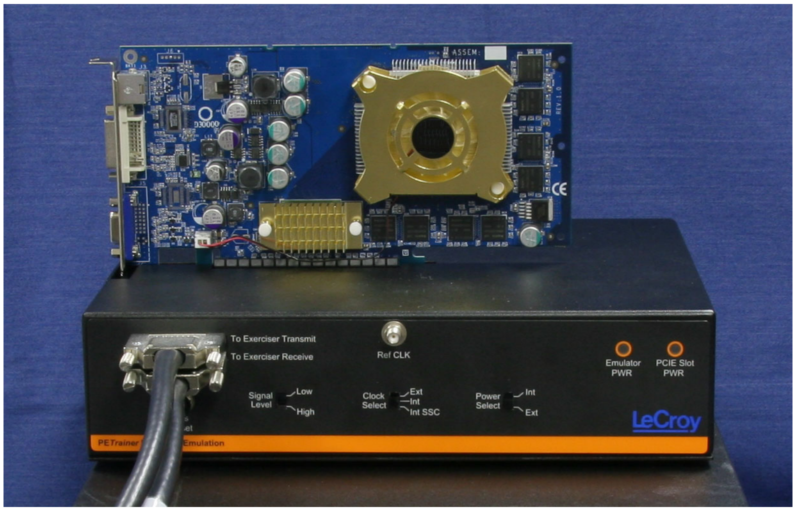


Figure 3.9 Host Emulation Test Fixture with a PCI-E Graphics Card on Top

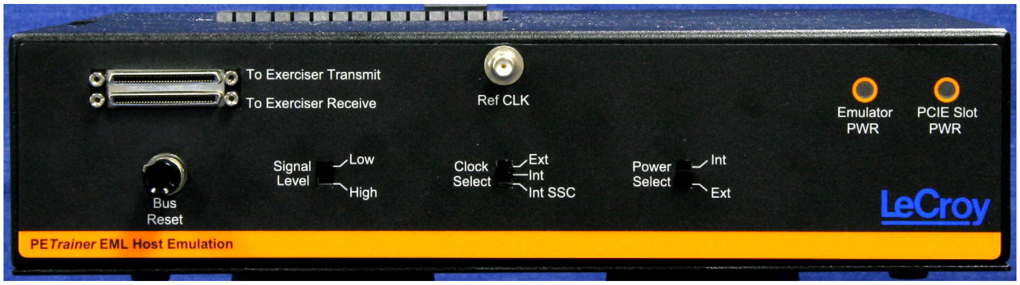


Figure 3.10 Host Emulation Test Fixture

LeCroy Host Emulation Test Fixture: Hardware enclosure with a slot on top for accommodating a PCI Express card.

Data Cable Connector (**To Exerciser Transmit**): Connects to the Transmit port on PE*Trainer* EML.

Data Cable Connector (**To Exerciser Receive**): Connects to the Receive port on PE*Trainer* EML.

Bus Reset button: Reset asserts PERST # for > 250 ms

Signal Level switch:

- Hi = PCI Express compliant (0.9V - 1.2V)
- Low = Testing (0.45V - 0.65V)

Clock Select: Selects from the following clocking options:

- **Ext:** a user supplied reference via the **RefCLK** SMA connector.
- **Int**: an internal supplied 100 MHz reference clock.
- **Int SSC**: an internal supplied Spread Spectrum Clock.

Power Select switch: Selects between the internal power supply and the external power jacks on the back of the emulator unit. When **External** power is selected, both 3.3V and 12V supplies must be provided by the user via the banana jacks on the back of the emulator.

Reference Clk connector: Provides a point of attachment for an external reference clock.

Emulator Power LED: Lights when emulator is powered on.

PCIE Slot Power LED: Lights when the PCI Express connector on the Host Emulator is supplying power. Power can be provided either via the internal power supply or by an external power supply via the banana jacks on the back of the emulator. If an external power source is provided, the Power Select switch should be set to **Ext**.

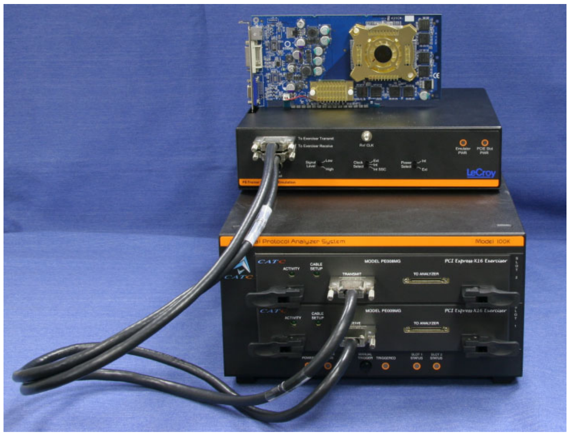
Installing the Host Emulation Test Fixture

Step 1 If not already powered off, power off the PE*Trainer*/UPAS 100K system and the DUT.

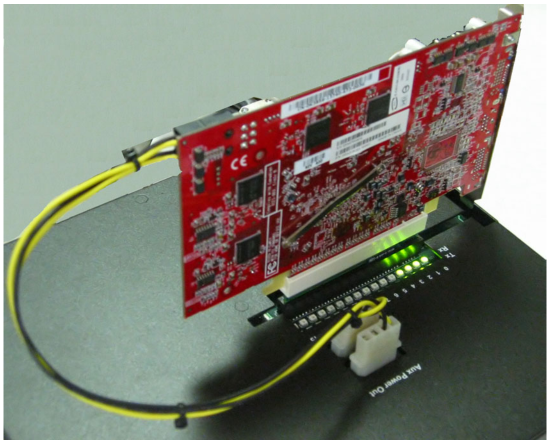
Step 2 Insert the male edge connector of the DUT into the PCI Express slot on top of the Host Emulation Test Fixture. If you are testing a x1, x4, or x8 device, install a slot reducer in the Host Emulator before installing the DUT.

Step 3 Attach a probe data cable to the **To Exerciser Transmit** port on the Host Emulator and the **Transmit** port on the PE*Trainer*.

Step 4 Attach a second probe data cable between the **To Exerciser Receive** port on the Host Emulator the **Receive** port on the PE*Trainer*.



Step 5 If the DUT is 150 watt device such as a graphics card, connect the card's external power cables to the two external power ports on top of the Host Emulator.



Final Steps: Power On the Exerciser and DUT

Step 1 If testing a motherboard, power on the motherboard. If testing a PCI-Express device such as a graphics card, power on the Host Emulator.

Step 2 Power on the PE*Trainer* EML Exerciser. The Exerciser's green power LED lights, and the red status LED turns on for approximately 20 seconds while the Exerciser performs self-diagnostics.

Step 3 Open the PE*Tracer* application on the PC host system. The Exerciser is now ready for traffic generation.

Step 4 Check the **Cable Setup** LEDs on the front of the Exerciser for indication of proper connectivity.

Optional: Adding a PE*Tracer* EML Analyzer

A PE*Tracer* EML can be added to the above setup to allow the recording of traffic between the PE*Trainer* EML Exerciser and the device under test. A photo of such a setup is shown in Section 3.10, “Connecting PETrainer EML to the Device Under Test” on page 63. When an Analyzer is added to the setup, a single PC administers both the Exerciser and Analyzer.

Step 1 Connect a USB cable between the PE*Tracer* EML Analyzer and the host PC for PE*Trainer* EML. The same PC administers both devices.

Step 2 Connect data cables between the two **To Analyzer** ports on the PE*Trainer* EML and the **Data In** ports on the PE*Tracer* EML Analyzer.

Step 3 Connect a power cable to the Analyzer.

Step 4 Power on the Analyzer.

Step 5 Start the PE*Tracer* application. When both PE*Tracer* EML and PE*Trainer* EML are attached and running, the PE*Tracer* application detects both platforms. At this point, you are ready to set the Recording Options in PE*Tracer* EML and start both traffic generation and trace recording.

Section 2. PE*Tracer* Analyzer Software

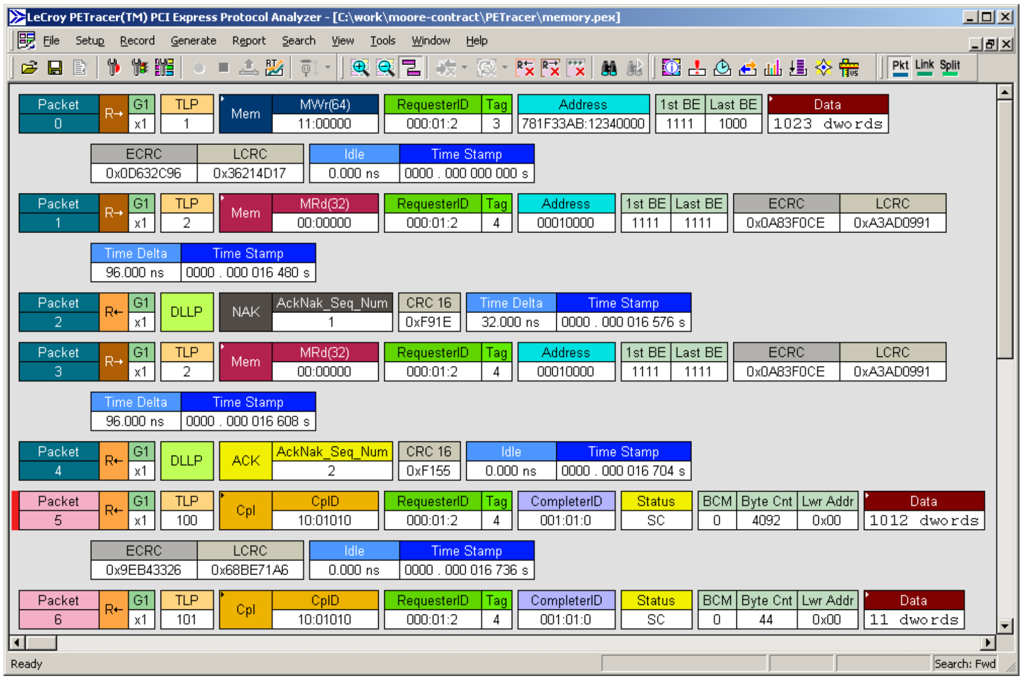
Chapter 4: Software Overview

4.1 The PE*Tracer* Software

The PE*Tracer*™ software administers Analyzer platforms and handles all trace analysis and display. When a PE*Trainer*™ Exerciser is present, the software generates PCI Express™ traffic. The core software is thus the same for all of LeCroy's PCI Express products.

The software runs on a Windows®-based PC that is attached to the Analyzer via a USB 2.0 connection (USB 1.0 is also supported). PE*Tracer* software can be used without the Analyzer as a trace viewer for viewing, analyzing and printing traces.

The PE*Tracer* software operates in Microsoft® Windows® 2000 and XP environments.


LeCroy PETracer(TM) PCI Express Protocol Analyzer - [C:\work\moore-contract\PETracer\memory.pex]
File
Setup
Record
Generate
Report
Search
View
Tools
Window
Help
Pkt
Link
Split

Packet		G1	TLP		MWr(64)	RequesterID	Tag	Address	1st BE	Last BE	Data
0	R→	x1	1	Mem	11:00000	000:01:2	3	781F33AB:12340000	1111	1000	1023 dwords

ECRC	LCRC	Idle	Time Stamp
0x0D632C96	0x36214D17	0.000 ns	0000 . 000 000 000 s

Packet		G1	TLP		MRd(32)	RequesterID	Tag	Address	1st BE	Last BE	ECRC	LCRC
1	R→	x1	2	Mem	00:00000	000:01:2	4	00010000	1111	1111	0x0A83F0CE	0xA3AD0991

Time Delta	Time Stamp
96.000 ns	0000 . 000 016 480 s

Packet		G1			AckNak_Seq_Num	CRC 16	Time Delta	Time Stamp
2	R←	x1	DLLP	NAK	1	0xF91E	32.000 ns	0000 . 000 016 576 s

Packet		G1	TLP		MRd(32)	RequesterID	Tag	Address	1st BE	Last BE	ECRC	LCRC
3	R→	x1	2	Mem	00:00000	000:01:2	4	00010000	1111	1111	0x0A83F0CE	0xA3AD0991

Time Delta	Time Stamp
96.000 ns	0000 . 000 016 608 s

Packet		G1			AckNak_Seq_Num	CRC 16	Idle	Time Stamp
4	R←	x1	DLLP	ACK	2	0xF155	0.000 ns	0000 . 000 016 704 s

Packet		G1	TLP		CplD	RequesterID	Tag	CompleterID	Status	BCM	Byte Cnt	Lwr Addr	Data
5	R←	x1	100	Cpl	10:01010	000:01:2	4	001:01:0	SC	0	4092	0x00	1012 dwords

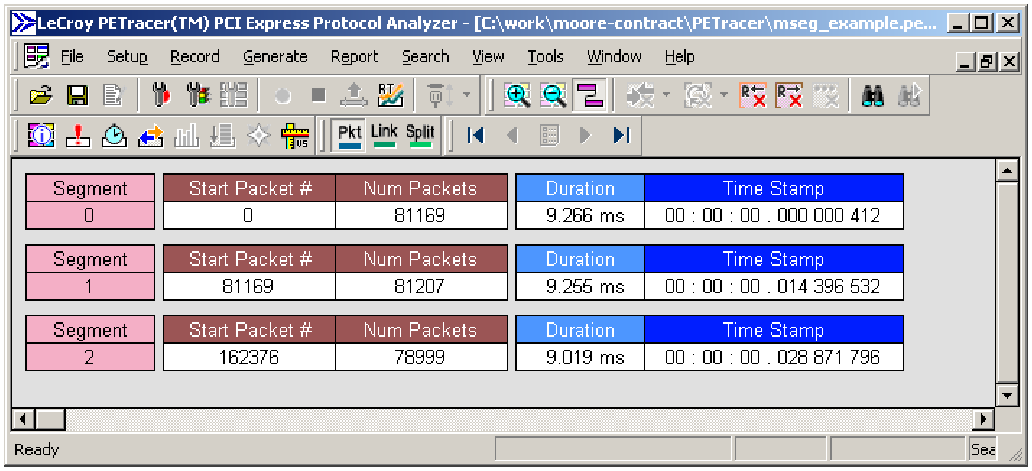
ECRC	LCRC	Idle	Time Stamp
0x9EB43326	0x68BE71A6	0.000 ns	0000 . 000 016 736 s

Packet		G1	TLP		CplD	RequesterID	Tag	CompleterID	Status	BCM	Byte Cnt	Lwr Addr	Data
6	R←	x1	101	Cpl	10:01010	000:01:2	4	001:01:0	SC	0	44	0x00	11 dwords

Ready Search: Fwd

4.2 Application Layout

The PE*Tracer* application contains the following components:


LeCroy PETracer(TM) PCI Express Protocol Analyzer - [C:\work\moore-contract\PETracer\mseg_example.pe...
File
Setup
Record
Generate
Report
Search
View
Tools
Window
Help
Pkt
Link
Split
Ready
Sea

Segment	Start Packet #	Num Packets	Duration	Time Stamp
0	0	81169	9.266 ms	00 : 00 : 00 . 000 000 412
1	81169	81207	9.255 ms	00 : 00 : 00 . 014 396 532
2	162376	78999	9.019 ms	00 : 00 : 00 . 028 871 796

Title bar: The title bar is at the top of the application window and displays the trace file name or generation script name.

Menu bar: The menu bar is located below the title bar, by default. The menu bar can be moved by clicking a blank area of the bar and then dragging the menu to a new position. It can be docked in another part of the application window or moved outside of the window to become a floating menu.

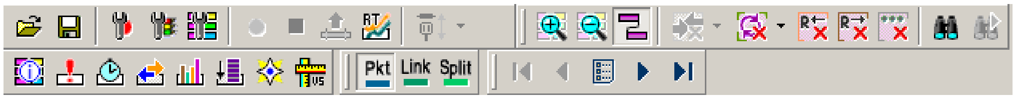
Toolbar: The toolbar is composed of buttons that represent the commonly used commands. The toolbar divides into smaller toolbars that can be moved and docked in a new position or made to float outside of the window.

Display area: The display area is the main part of the application window in which traces are displayed.

Status bar: The status bar is located at the bottom of the application window. The left end of the status bar displays descriptions of buttons and menu items when the mouse is positioned over them. The right end of the bar shows the search direction.

4.3 Using the Toolbar

You can use the PE*Tracer* application Toolbar as a shortcut to most of the operations supplied by the menus.



Opens a trace file

Save As. Allows the open file to be saved with a new name.

Preview. Shows what the open file looks like when it is printed.

Print … Prints the open trace of traffic generation file.

Script Editor. Opens a text editor for editing PE*Trainer* traffic generation files (***.peg)**

Setup Recording Options. Opens Recording Options dialog box.

Setup Generation Options

Setup Display Options. Opens Display Options dialog box.

Start Recording

Stop Recording

Hide Upstream

Hide Downstream

Advanced Hiding Options

Find

Find Next. Repeats last Find or Search action.

File Information

Error Report

Timing and Bus Usage Calculations

Traffic Summary

Bus Utilization

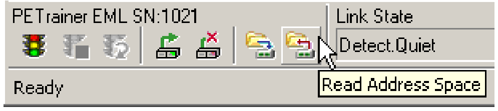
	Repeat Upload. Causes the Analyzer to re-send the trace from the Analyzer buffer to the host PC.		Link Tracker
	Disconnect/Connect Link. Click once to break and momentarily later restore link. Open menu to select either Connect or Disconnect.	RT	Real Time Statistics monitor
	Zoom in		Navigator. Shows/hides the Navigator bar - a utility that lets you easily navigate the trace.
	Zoom out	Pkt	View Packet Level
	Wraps the display	Link	View Link Transaction Level
	Hide Training	Split	View Split Transaction Level
	Hide SKIP and Update FC	VS	Running Verification Scripts
			Multi-Segment Toolbar

Multi-Segment Toolbar

⏮	Open first segment in the multisegment trace.		Open index file. This button becomes active if a multisegment trace file is open.
◀	Open previous segment in the multisegment trace.	▶	Open next segment in multisegment trace.
		⏭	Open last segment in multisegment trace.

Traffic Generation Toolbar

These buttons appear on the status bar at the bottom of the screen if a PE*Trainer*™ Exerciser is attached. The buttons are explained below.



Start traffic generation

Resume traffic generation.

Stop traffic generation

Trainer Connect. Attempts to establish a connection between PE*Trainer* and the DUT.

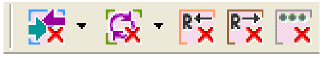
Trainer Disconnect. Breaks the connection between PE*Trainer* and the DUT.

Read Address Space

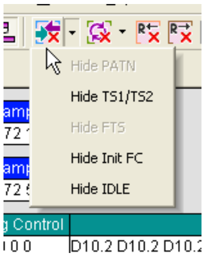
Write Address Space

Toolbar Hide Buttons

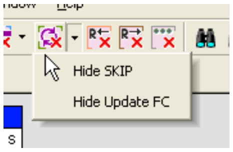
The PE*Tracer* application toolbar has five buttons related to show/hide of trace file contents. The buttons allow you to quickly adjust the trace display to your needs.



Hide Training: Brings up a drop-down menu that lets you to hide all or a portion of the packets in the training portion of the trace.



Hide SKIP and Update FC: Brings up a drop-down menu that lets you hide SKIP or Update FC packets in a trace.

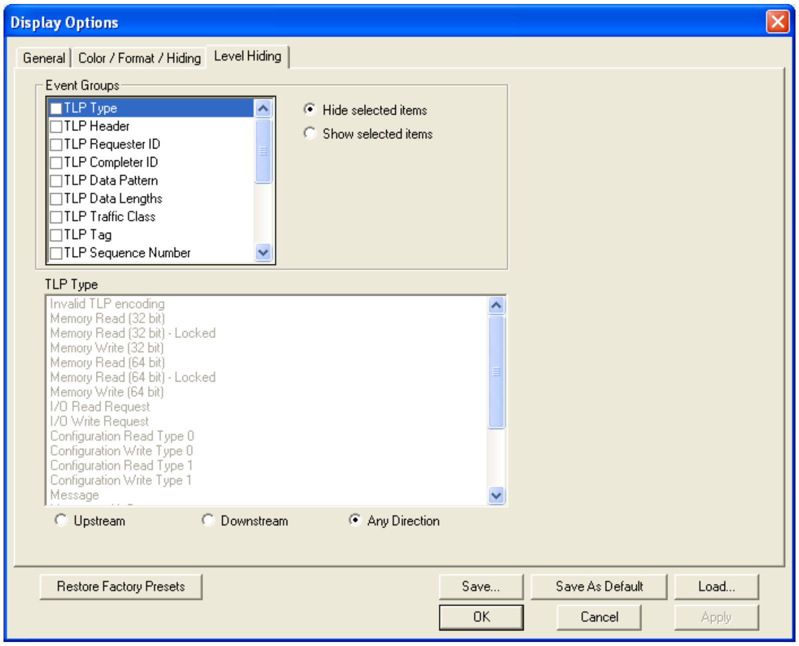


Hide Upstream: Hide all traffic from endpoint devices to the root complex.

{

Hide Downstream: Hide all traffic from the root complex to endpoint devices.

Advanced Hiding Options: Brings up the Level Hiding pane of the Display Options window. Use this pane to tune the show/hide options within the trace display.



Display Options

General | Color / Format / Hiding | Level Hiding

Event Groups

- □ TLP Type
- □ TLP Header
- □ TLP Requester ID
- □ TLP Completer ID
- □ TLP Data Pattern
- □ TLP Data Lengths
- □ TLP Traffic Class
- □ TLP Tag
- □ TLP Sequence Number

● Hide selected items
○ Show selected items

TLP Type

- Invalid TLP encoding
- Memory Read (32 bit)
- Memory Read (32 bit) - Locked
- Memory Write (32 bit)
- Memory Read (64 bit)
- Memory Read (64 bit) - Locked
- Memory Write (64 bit)
- I/O Read Request
- I/O Write Request
- Configuration Read Type 0
- Configuration Write Type 0
- Configuration Read Type 1
- Configuration Write Type 1
- Message

○ Upstream ○ Downstream ● Any Direction

4.4 Using the Menus

The PE*Tracer* application main display includes the following set of pull-down menus:

File Menu

New PE*Trainer* Script	Creates a new Traffic Generation file.
Open	Opens a file.
Close	Closes the current file.
Save as	Saves all or a specified range of packets to a uniquely named file.
Convert Old Files	Convert trace files made in previous versions of PE*Tracer* to the new format for PE*Tracer* version 5.0 and higher.
Print	Prints part or all of the current traffic data file.
Print Preview	Produces an on-screen preview before printing.
Print Setup	Setup your current or new printer.
Edit Comment	Create or edit the comment field in a trace file.
Export >> to Text >> Packets to CSV Text >> to Generator File Format	Saves all or part of a trace to a text file or traffic generation file. This text file summarizes the traffic in the trace. Saves the trace to a text file in Comma Separated Value (CSV) format. Saves the trace to a generation file.
Exit	Exit the PE*Tracer* application.

Setup Menu

Display Options

Allows you to customize display options such as field colors, field formats and level hiding.

Recording Options

Allows you to customize control and setup features associated with recording, triggering, and filtering recorded events.

Generation Options

The Generation Options dialog box is used to set configuration settings in a traffic generation script (*.peg). This dialog provides a convenient alternative means of editing a traffic generation file.

Generation Macros

Opens a dialog for creating buttons on the status bar that can be used to run traffic generation macros on a PE*Trainer* ML or PE*Trainer* EML.

Update Device

Opens a dialog box that lets you update the BusEngine™ and Firmware.

Analyzer Network

Displays the list of PCs with connected analyzers or exercisers that you want to use for recording or traffic generation.

All Connected Devices

Opens a dialog box with a list of locally or remotely connected devices. Allows you to update the BusEngine, Firmware, and your licensing information.

Record Menu

Start

Causes the Analyzer to begin recording traffic.

Stop

Causes the Analyzer to stop recording traffic.

Reupload

Causes the Analyzer to re-send the trace from the Analyzer buffer to the host PC.

Disconnect/Connect

Causes the Analyzer to momentarily break, and then establish the PCI Express link connection in both link directions.

Reset Link Configuration

Causes the Analyzer to reset the series and thereby reset its the current link configuration. This command is needed when lane width or other lane settings are changed because otherwise the old link configuration is used and errors are generated in the trace.

Generate Menu

Connect
Initializes the link between PE*Trainer* and the device under test.

Disconnect
Disconnects the link between PE*Trainer* and the device under test.

Write Address Space
Reads PE*Trainer* internal memory used for address space mapping.

Read Address Space
Loads PE*Trainer* internal memory used for address space mapping.

Report Menu

Report Menu operations are only available when you are working with a trace file.

File Information
Displays the brief information about the file contents and describes the conditions of the file’s recording (Recording Options, hardware information, and so on).

Error Summary
Displays an error summary of the current trace file and allows you to go to a specific packet, and save the error file to a uniquely named file.

Timing Calculations
Starts the mode-less calculator dialog for calculating various timing and bandwidth parameters in the recording file.

Traffic Summary
Details the number and types of packets that were transferred during the recording.

Bus Utilization
Opens a window with graphs of bus usage for the open trace.

Link Tracker
Opens a window for displaying a detailed chronological view of traffic. The window provides view and navigation options.

TC to VC Mapping
This command allows Traffic Classes to be mapped to Virtual Channels for purposes of simplifying navigation (for example, Search > Go to > TLP Virtual Channel) and changing the way the trace is displayed (for example, in Split Transactions).

Configuration Space
Presents a list of Configuration Spaces. Clicking an item displays the selected Configuration Space.

Run Verification Scripts
Presents a list of verification scripts, from which you can run a verification script.

Search Menu

Search Menu operations are only available when you are working with a trace file.

Go to Trigger	Repositions the display to show the first packet following the trigger event.
Go to Packet	Repositions the display to a specific packet.
Go to Time	Repositions the display to specific time.
Go to Marker	Repositions the display to a previously marked packet.
Go to	Allows searching for specific link events.
Find	Allows searching for various events within a trace. You can search by Event Group and by Display Level. Find allows you to create complex searches based on numerous criteria.
Find Next	Repeats the previous Find or Search operation.
Search Direction	Allows you to specify a forward or backward search of a trace file.

View Menu

Toolbars>>	Displays list of available toolbars.
Script Editor	Displays the Script Editor (only appears if a **.peg** file is open).
Analyzer Network Chat Bar	Opens up a chat window for communicating with persons working with networked Analyzers. This command requires that your host PC be attached to a LAN. The Chat window broadcasts messages to whatever hosts have been connected to via the Analyzer Network command (under Setup in the menu).
Status Bar	Switches display of the Status Bar on or off.
Real Time Statistics	Opens the Real Time Statistics monitor dialog and displays a real-time graph of link activity.
Zoom In	Zoom in increases the size of the displayed elements.
Zoom Out	Zoom out decreases the size of the displayed elements.
Wrap	Allows the display to wrap.
FC Credits	Toggles a display for tracking Flow Control Credit update and consumption on a PCI Express link.
FC Credits Setup	Allows you to customize the display for tracking Flow Control Credits.
Packet Level	View Packet Level.
Link Transaction Level	View Link Transaction Level.
Split Transaction Level	View Split Transaction Level.

Tools Menu

Configuration Space Editor	Opens the Configuration Space Editor
Memory/IO Space Editor	Opens the Memory/IO Space Editor

Window Menu

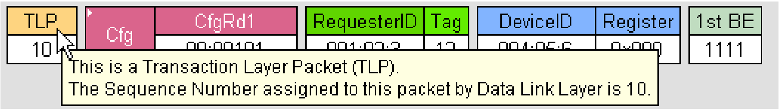
New Window	Opens a copy of the current trace window.
Cascade	Displays all open trace windows in an overlapping arrangement.
Tile	Displays all open trace windows as a series of strips across the display.
Arrange Icons	Arranges minimized trace windows at the bottom of the display.

Help Menu

Help Topics	Accesses the PE*Tracer* application's Online Help.
Update License	Displays a dialog box for entering updated license information.
Display License Information	Opens an information box describing the current license information.
About	Displays version information about the attached Analyzer and its Firmware and BusEngine™.

4.5 Tool Tips

Tool tips provide details about fields within the trace. To see a tool tip, position your mouse pointer over a field within the trace.



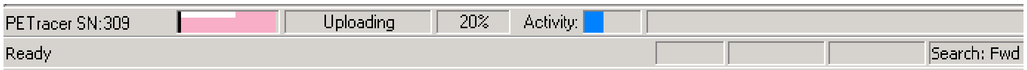
This is a Transaction Layer Packet (TLP).
The Sequence Number assigned to this packet by Data Link Layer is 10.

4.6 Keyboard Shortcuts

Keystrokes	**Operation**
Ctrl+O	Open file
Ctrl+P	Print
Ctrl+S	Save file
Ctrl+Home	Jump to first packet
Ctrl+End	Jump to last packet
F3	Find next
F6	Next pane
Shift+F6	Previous pane
Ctrl+B	Search backward
Ctrl+F	Search forward

4.7 Status Bar

The Status Bar is a gray bar that runs along the bottom of the application window.



PETracer SN:309 | Uploading | 20% | Activity:

Ready | Search: Fwd

From left to right:

PE*Tracer* SN:309: Analyzer Serial Number

Ready: Analyzer Status

Recording Progress Bar: The colored bar to the right of the serial number represents how much traffic has been recorded. The trigger point is indicated by the black line at the left side of the bar. In this example, the trigger occurred at the very beginning of the trace. If the trigger is set in the middle of the trace, the line is positioned in the middle of the bar. Additionally, the color of the bar is different on each side of the trigger point. For examples, see “Recording Progress Bar” on page 86. The white strip along the top edge of the color bar indicates how much traffic has been uploaded from the Analyzer buffer to the host PC.

Uploading and 20%: Indicates the Analyzer's recording status, what part of the recording process the Analyzer is now in. In this example, the Analyzer is in the upload stage and has completed 20% of the upload from the Analyzer to the PC. See “Recording Status” on page 86 for details on other status messages.

Activity: The colored bar moves to indicate that the Analyzer is currently recording.

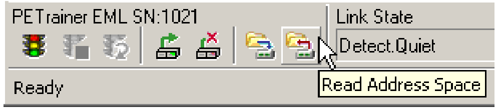
Search: Fwd: Indicates search direction. The direction can be toggled to **Search: Bwd** by double-clicking the search direction or by selecting **Search Direction** from the Search menu.

PE*Trainer* Status Bar

When a PE*Trainer* is attached, a Generation Status bar is displayed at the bottom of the screen. This bar has buttons to start and stop generation and to break and re-establish connection with the DUT. In addition, status info is displayed. See “Using the Toolbar” on page 73 for an explanation of these buttons.

From left to right:

Buttons: Start Generation, Stop Generation, Resume Generation, Break Connection, Resume Connection, Write Address Space, Read Address Space

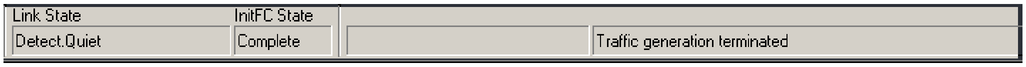


PETrainer EML SN:1021

Link State: Detect.Quiet

Ready

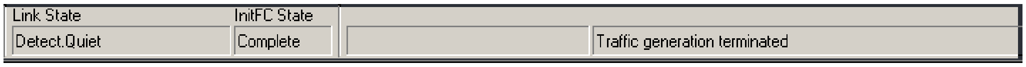
Status: Link State, InitFC State, Trainer Status



Link State	InitFC State		
Detect.Quiet	Complete		Traffic generation terminated

Link State Messages on PE*Trainer* Status Bar

When running PE*Trainer*, the status bar displays PE*Trainer* buttons and status information. The buttons allow you to start and stop traffic generation and to create and break links between PE*Trainer* and the DUT.



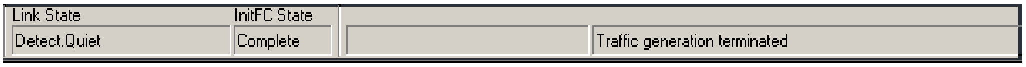
Link State: Detect.Quiet
InitFC State: Complete
Traffic generation terminated

The Status bar has a section labeled **Link State** that displays messages about the Trainer link. The following list shows what types of messages can be displayed:

- Detect.Quiet
- Detect.Active
- Polling.Active
- Polling.Compliance
- Polling.Configuration
- Polling.Speed
- Configuration.Linkwidth.Start
- Configuration.Linkwidth.Accept
- Configuration.Lanenum.Wait
- Configuration.Lanenum.Accept
- Configuration.Complete
- Configuration.Idle
- L0
- L0s.Idle
- L0s.FTS
- L1
- L2
- Recovery.RcvrLock
- Recovery.RcvrCfg
- Recovery.Idle
- Loopback
- Hot Reset
- Disabled

InitFC State Messages on PE*Trainer* Status Bar

The Status bar has a section labeled **InitFC State** that displays messages about the Trainer InitFC State.



Link State: Detect.Quiet
InitFC State: Complete
Traffic generation terminated

The following list shows what types of messages can be displayed:

- Not initialized
- Pending
- Complete

Recording Progress Bar

This indicator bar changes color to reflect the recording progress.

- Black vertical line is at the location of Trigger position.
- Black vertical line wiggles when Trigger Position is nearly reached.
- Field to right of Trigger Position changes color to indicate post-trigger activity.
- Upper half of progress indicator turns white when recording is complete.

Recording Status

The second segment from the left in the Status Bar indicates recording status.

During recording, this status flashes one of the following messages:

- Trigger?
- Triggered!
- Uploading

After recording stops, the message changes to

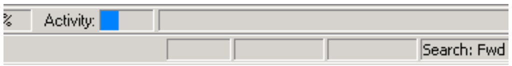
- Uploading data - x% done (where x equals the percent that has been uploaded.)

As uploading progresses, the percent increases to 100. You can abort this upload if you wish by pressing the **<escape>** button on your keyboard or

clicking ■ in the Tool Bar.

Recording Activity

The third segment displays recorded activity. Activity is indicated in blue:



Search Direction Indicator

The fourth segment in the status bar indicates search direction. The direction can be changed by selecting Search Direction from the Search menu.

4.8 Recording PCI Express Data

To record PCI Express traffic:

Step 1 Load or define a recording options file.

Step 2 Start a recording session by selecting **Start** from the Record menu or clicking the **REC** icon in the toolbar.

- Your recording session runs according to the specified recording options.
- Recording is tracked and reported on an activity meter in the status bar.

Step 3 You can interrupt a session by pressing the **Escape** key (**Esc**). If Recording is finished or Upload has started but has not finished, a message box appears. If Upload has started but has not finished, you can:

- Continue uploading.
- Abort the upload and flush the data.
- Abort the upload and preserve all the previously saved data.

Step 4 You can manually stop a recording session by selecting **Stop** from the Record menu or clicking the **Stop** button on the toolbar.

When the recording session finishes, the PCI Express traffic is saved to the hard drive as the file specified in the recording options and this file is opened in the application for you to view the traffic.

4.9 Making a PCI Express Recording

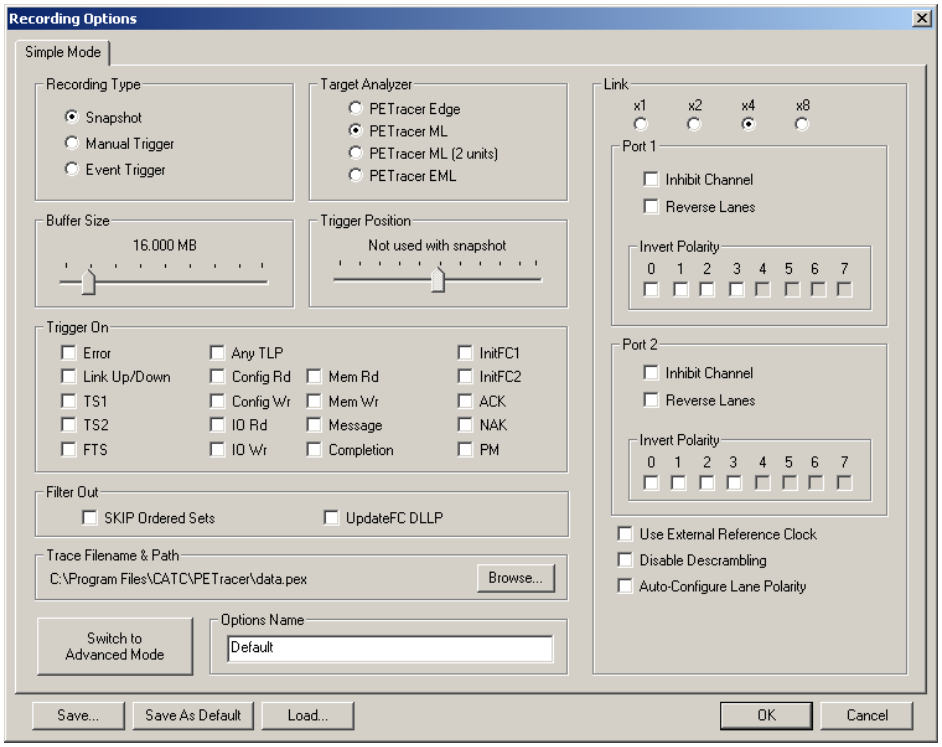
After connecting the Analyzer to the device(s), you must configure the Recording Options. Then you can test the Analyzer by creating a 16-Mbyte snapshot recording.

To make this recording, follow these steps:

Step 1 From the **Setup** menu, select **Recording Options**.

Step 2 Select the **General** tab.

The following window displays the factory **PE*Tracer* ML** default settings in **Simple Mode**, such as Snapshot and 16 MB buffer size. (The window is the same for PE*Tracer* Edge in Simple Mode, except that there is no x8 Link Width.) For your first recording, you can leave these settings unchanged.



Recording Type
● Snapshot
○ Manual Trigger
○ Event Trigger

Target Analyzer
○ PETracer Edge
● PETracer ML
○ PETracer ML (2 units)
○ PETracer EML

Buffer Size
16.000 MB

Trigger Position
Not used with snapshot

Trigger On
□ Error □ Link Up/Down □ TS1 □ TS2 □ FTS
□ Any TLP □ Config Rd □ Config Wr □ IO Rd □ IO Wr
□ Mem Rd □ Mem Wr □ Message □ Completion
□ InitFC1 □ InitFC2 □ ACK □ NAK □ PM

Filter Out
□ SKIP Ordered Sets □ UpdateFC DLLP

Trace Filename & Path
C:\Program Files\CATC\PETracer\data.pex

Options Name
Default

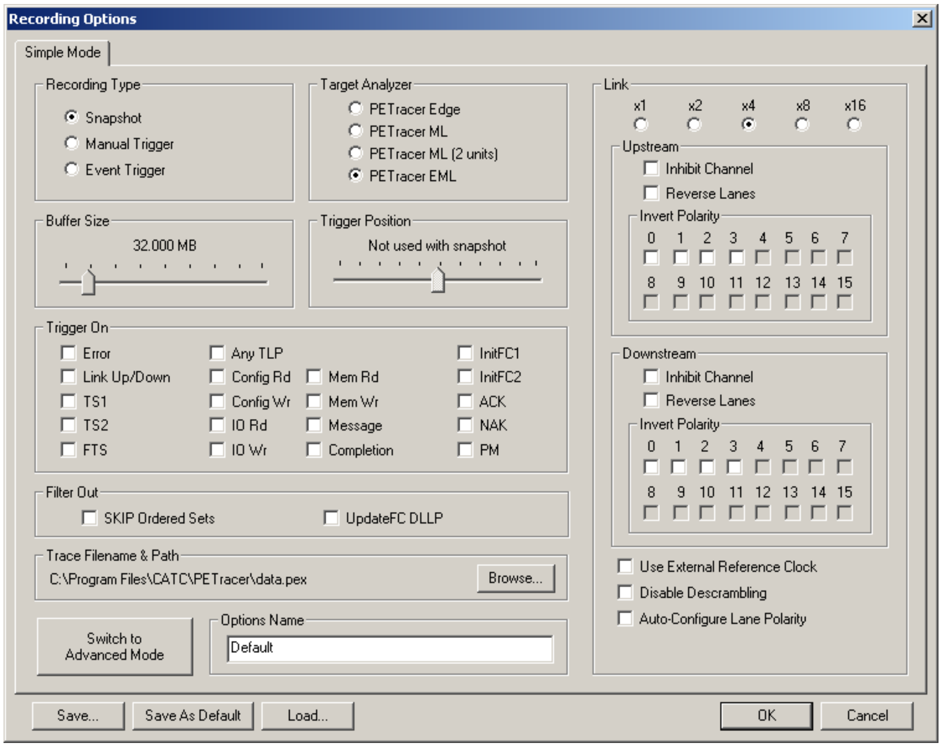
Link
x1 ○ x2 ○ x4 ● x8 ○

Port 1
□ Inhibit Channel
□ Reverse Lanes
Invert Polarity
0 □ 1 □ 2 □ 3 □ 4 □ 5 □ 6 □ 7 □

Port 2
□ Inhibit Channel
□ Reverse Lanes
Invert Polarity
0 □ 1 □ 2 □ 3 □ 4 □ 5 □ 6 □ 7 □

□ Use External Reference Clock
□ Disable Descrambling
□ Auto-Configure Lane Polarity

The following window displays the factory **PE*Tracer* EML** default settings in **Simple Mode**.



Recording Options

Simple Mode

Recording Type
● Snapshot
○ Manual Trigger
○ Event Trigger

Target Analyzer
○ PETracer Edge
○ PETracer ML
○ PETracer ML (2 units)
● PETracer EML

Link
○ x1 ○ x2 ● x4 ○ x8 ○ x16

Upstream
□ Inhibit Channel
□ Reverse Lanes
Invert Polarity
0 □ 1 □ 2 □ 3 □ 4 □ 5 □ 6 □ 7 □
8 □ 9 □ 10 □ 11 □ 12 □ 13 □ 14 □ 15 □

Buffer Size
32.000 MB

Trigger Position
Not used with snapshot

Trigger On
□ Error □ Any TLP □ InitFC1
□ Link Up/Down □ Config Rd □ Mem Rd □ InitFC2
□ TS1 □ Config Wr □ Mem Wr □ ACK
□ TS2 □ IO Rd □ Message □ NAK
□ FTS □ IO Wr □ Completion □ PM

Filter Out
□ SKIP Ordered Sets □ UpdateFC DLLP

Trace Filename & Path
C:\Program Files\CATC\PETracer\data.pex Browse...

Switch to Advanced Mode

Options Name
Default

Downstream
□ Inhibit Channel
□ Reverse Lanes
Invert Polarity
0 □ 1 □ 2 □ 3 □ 4 □ 5 □ 6 □ 7 □
8 □ 9 □ 10 □ 11 □ 12 □ 13 □ 14 □ 15 □

□ Use External Reference Clock
□ Disable Descrambling
□ Auto-Configure Lane Polarity

Save... Save As Default Load... OK Cancel

Step 3 In the **Link** section, specify the lane width of the PCI Express link to be analyzed. The rest of the settings in this section can be left at the factory defaults for most PCI Express systems.

Step 4 For multi-lane PCI Express links, the Analyzer needs to observe link training in order to record link traffic correctly. If link training (or re-training) is not easily controllable for the devices under test, the Analyzer includes the capability to force link training by disconnecting and reconnecting the PCI Express link.

Clicking the **Connect/Disconnect** button disconnects the PCI Express link for one second in both directions, then re-establishes the link.

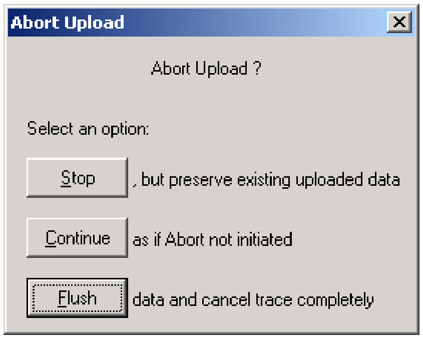
Note: For x1 PCI Express links, it is not necessary for the Analyzer to observe link training in order to record link traffic; this step may be skipped.

Step 5 Click **OK** at the bottom of the Recording Options dialog box to apply the Analyzer recording settings specified.

Step 6 Click the **Record** ● button.

Stopping a Recording

You can stop the recording process at any time by pressing the **Stop** button ■ . This causes the Analyzer to stop the recording and upload the trace to the host PC. If you press the **Stop** button again, the Analyzer temporarily halts the upload process and opens the following dialog box:



Abort Upload

Abort Upload ?

Select an option:

Stop , but preserve existing uploaded data

Continue as if Abort not initiated

Flush data and cancel trace completely

This dialog presents options for stopping, continuing, or aborting the upload:

- **Stop**: Aborts further trace upload and displays whatever trace data that has already been uploaded.
- **Continue**: Resumes the upload. This command tells the Analyzer to finish uploading whatever trace data is still in its buffer.
- **Flush**: Flushes the trace without saving or displaying it.

If you allow the traffic data to be uploaded, it is automatically saved on the PC’s hard drive as a file named **data.pex** or the name you assign as the default filename in the recording options.

Saving a Recording

Step 1 To save a current recording for future reference, select **Save As** from the File menu.

Step 2 Give the recording a unique name, then save it to the appropriate directory.

4.10 Recording Multi-Segmented Traces

If **Save as Multisegment trace** is enabled, PE*Tracer* divides the trace as it records it into segments and stores them on the host's hard drive.

This option is useful for very large recordings and for host PCs with limited memory. In the latter case, multi-segmenting gives a PC with limited memory a way to open recordings that are otherwise too large to open.

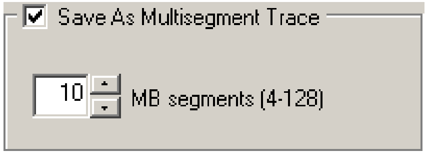
The only downside to multi-segmenting is that limits the scope of reports such as Traffic Summary, Bus Utilization, and Error Summary to each of the segments. You cannot perform summary statistics on the full recording.

The default segment size is 10 MB. Before attempting large recordings, it is recommended that you play with this number to see what value best suits your needs.

How to Create a Multisegment Recording

To create a Multisegment recording:

Step 1 Select **Setup > Recording Options** to open the Recording Options dialog at the General tab. Click **Switch to Advanced Mode**..



☑ Save As Multisegment Trace

10 MB segments (4-128)

Step 2 Check the box marked **Save as Multisegment trace**.

Step 3 In the box marked MB Segments, enter a value of 4 to 128 MB.

What Gets Created

Multisegmenting produces an index file and segmented trace files. The default name of the index file is **data.pem**.

The index file and the segmented trace files are stored in a directory named after the index file. The directory is named **data_pem_files**. Below this directory additional, sequentially numbered sub-directories (up to 100,000) hold the segmented trace files. These sub-directories bear simple numerical names: 00000 to 00999. Each of these subdirectories can hold up to 100 sequentially numbered segment files. Collectively, the entire directory structure can hold up to 10 million files.

Example

If you create a 1010 MB recording using 10 MB segments, the following files and sub-directories are created:

`data.pem` (This is the index file.)

```
data_pem_files\00000\segment_00000.pex

data_pem_files\00000\segment_00001.pex

...

data_pem_files\00000\segment_00099.pex

data_pem_files\00001\segment_00100.pex

data_pem_files\00001\segment_00101.pex
```

Note: An additional index file is created called **segments.col**. This is an internal system file that users should not modify.

4.11 PE*Tracer* Files

The PE*Tracer* software creates and uses different kinds of files:

- **Trace Files**: Recorded traffic
- **Recording Options Files**: Configuration file that contains the various options selected in the Recording Options dialog box to configure the recording
- **Display Options Files**: Configuration file that contains the options selected in the Display Options dialog box to configure how traffic is displayed

Trace Files

PE*Tracer* records PCI Express traffic into a trace file with the default name **data.pex** or any other that you specified in the Recording Options. This file is overwritten with new data each time PCI Express traffic is recorded. If recording an x8 link using two PE*Tracer* ML™ units, the trace files from the separate Analyzer units are merged into a single file with the default name **data.pex.**

When Multisegment traces are created, a special index file is also created called **<filename>.pem**. This index file provides a high level view of the trace segments created in the recording session.

If you want to save a trace, use the **File > Save As** function. This option allows you to save the current trace to a unique file name, thereby ensuring that it is not overwritten. This option also allows you to save a range of packets in a trace file.

You can pre-define the name of the recorded trace file using the Trace Filename and Path option in Recording Options.

Recording Options Files

Recording Options files are created when you set recording options. These files use the **.rec** extension and contain recording option information.

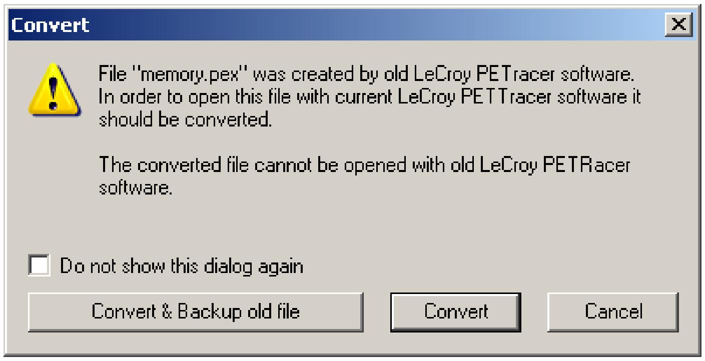
Display Options Files

Display Options files are created when you set display options.

4.12 Opening Trace Files

To open an existing trace file, click **File > Open** or 📂 .

If the file was made in a previous version of PE*Tracer*, the application presents the Convert dialog box:



File "memory.pex" was created by old LeCroy PETracer software. In order to open this file with current LeCroy PETTracer software it should be converted.

The converted file cannot be opened with old LeCroy PETRacer software.

☐ Do not show this dialog again

You must convert trace files made in previous versions of PE*Tracer* for them to open in PE*Tracer* version 5.0 or higher. The converted file has all the information that was in the original file.

You can:

- **Convert & Backup old file**: Convert the file, open it in PE*Tracer* version 5.0 or higher, and save it with the original name. Save the original file with the same name plus the extension **.bak**.
- **Convert**: Convert the file, open it in PE*Tracer* version 5.0 or higher, and save it with the original name.

Note: After you convert a trace file, you cannot open the converted file in a previous PE*Tracer* version.

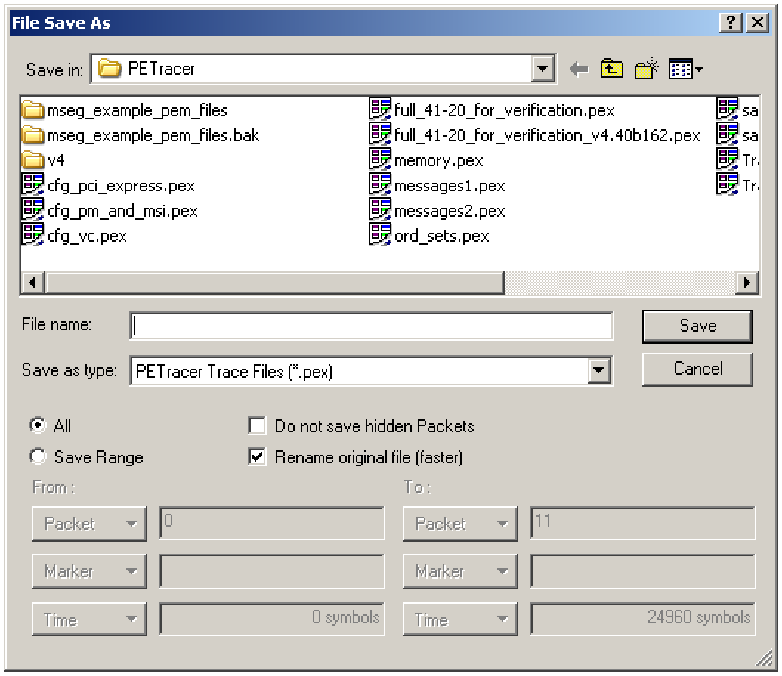
4.13 Saving Trace Files

Using the Save As Function

When you record a trace file, the Analyzer software provides a pre-defined name to the trace file (**data.pex** or any other that you specified in the Recording Options). If you do consecutive recordings, each time the previous recording is overwritten. If you see a recording you want to analyze later, you need to give a unique name to the trace file, so it is not overwritten with the next recording.

Also, when you analyze a recorded trace file, you might be interested in preserving just a part of the PCI Express traffic that was recorded. If you save a portion of a trace file, it can get significantly smaller, allowing you to attach it to an e-mail.

To save a portion of a trace file or the whole file to a unique name, select **Save As** from the File menu.



Save in: PETracer

mseg_example_pem_files
mseg_example_pem_files.bak
v4
cfg_pci_express.pex
cfg_pm_and_msi.pex
cfg_vc.pex
full_41-20_for_verification.pex
full_41-20_for_verification_v4.40b162.pex
memory.pex
messages1.pex
messages2.pex
ord_sets.pex
sa
sa
Tr
Tr

File name:

Save as type: PETracer Trace Files (*.pex)

● All
○ Save Range
☐ Do not save hidden Packets
☑ Rename original file (faster)

From : Packet 0; Marker; Time 0 symbols
To : Packet 11; Marker; Time 24960 symbols

If you want to give a unique name to the trace file, select the **All** option and keep **Rename original file** checked. This is the default setting for the dialog.

Saving a Portion of a Trace

If you want to save a portion of a trace file, select **Save packet range**. Enter starting and ending packet numbers in the **From Packet** and **To Packet** fields. By default, it has the numbers of the first and the last packets in the file. The software is going to save all the packets in specified range to the new file, unless you have **Do not save hidden packets** checked. In this case it is going to save all packets in the range, EXCLUDING the currently hidden packets.

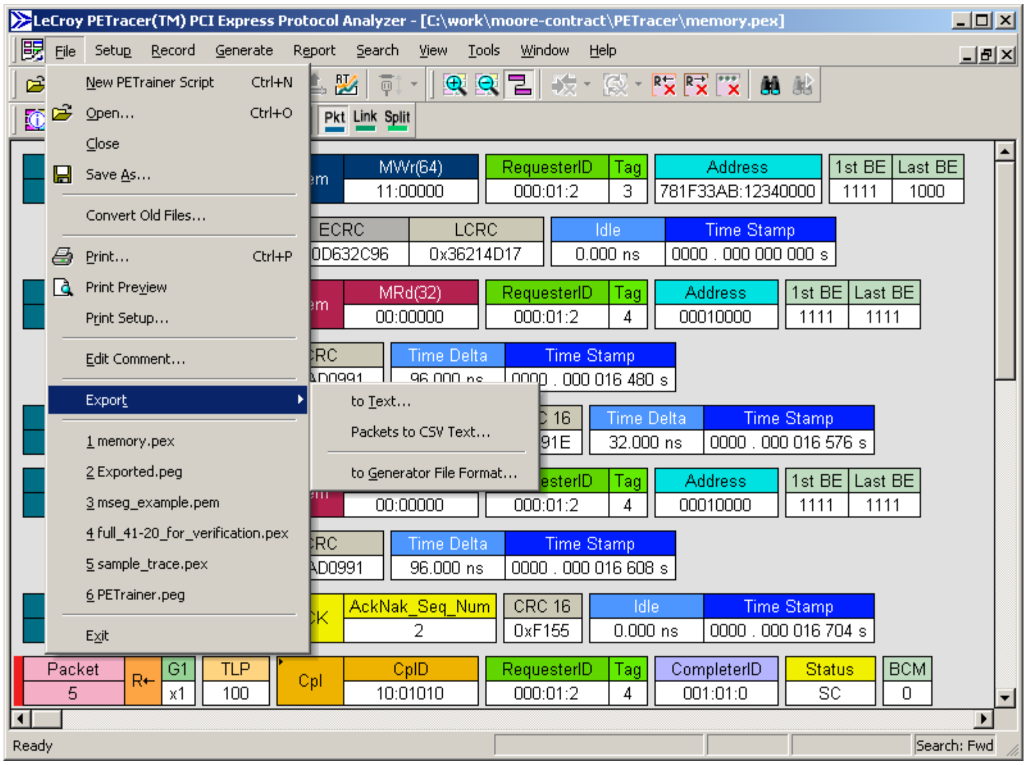
4.14 Exporting a Trace File

By default, PE*Tracer* saves trace files in the **.pex** format. However, you can export a trace to a file in any of the following other formats:

- Text
- Comma Separated Value (CSV)
- Generator file format

Exporting to Generator file format is a simple way to create a script file from your trace.

To export a trace file, select **File > Export**:



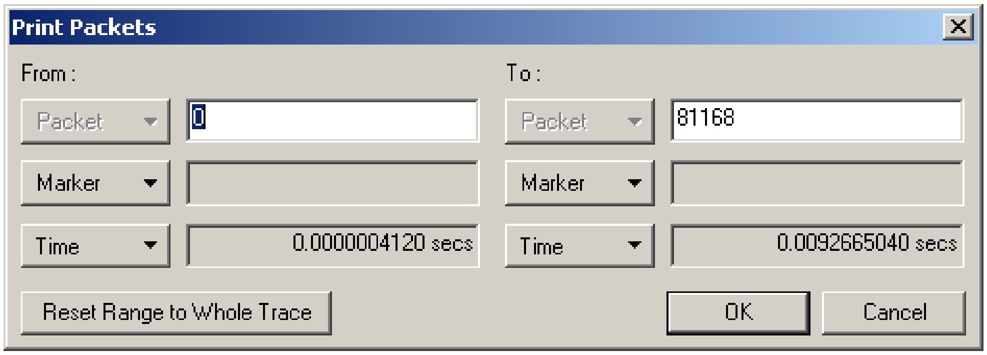
4.15 Printing Data Files

To print all or part of the open trace:

Step 1 Select **File > Print** from the menu or

click the **Printer** 🖨 button on the toolbar.

The Print Packets dialog opens:



Print Packets

From :

Packet ▾ 0

Marker ▾

Time ▾ 0.0000004120 secs

To :

Packet ▾ 81168

Marker ▾

Time ▾ 0.0092665040 secs

Step 2 To select a range of packets, enter values in the **From packet #** and **To packet #** fields and click **Print**.

Step 3 To print an entire file, leave the From and To fields empty and click **Print**.

The specified traffic information is printed as currently displayed, in color or gray scale as supported by your printer. Any Trace File comments you entered are printed following the current document name at the top of each page.

Note: Trace File comments can be created by using **Edit Comment** on the File menu.

4.16 Set Marker

A marker is a unique label for a packet that allows you to go to that packet and also serves as a **comment string** for a specific packet. When you select a marker, the identified packet appears at the top of the screen. Packets that have been **marked** have a red bar on the left edge.

To set a marker, perform the following steps:

Step 1 Left-click the **Packet #** field of the packet you want to mark.

Step 2 Click **Set Marker**.

Step 3 When the Mark Packet # pop-up appears, enter a unique identifier for the packet in the Comment field.

Later, you can go directly to this packet using the Go To Marker operation in the Search Menu.

4.17 Edit or Clear Marker

To change a markers identifier, or clear (delete) the marker:

Step 1 Click the mouse pointer in the **Packet #** field of the desired packet. A pop-up menu appears.

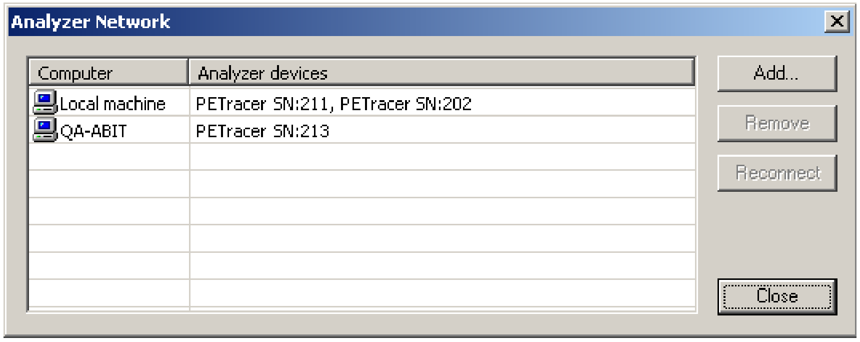
Step 2 Choose **Edit Marker Comment** and enter a new identifier into the Edit Marker pop-up, or
Choose **Clear Marker**. When you choose Clear Marker, the marker is removed and the red line disappears.

You can also view the raw bits that make up the data field by left-clicking the field and selecting View Data Block from the pop-up menu.

4.18 Browsing to a Networked Analyzer

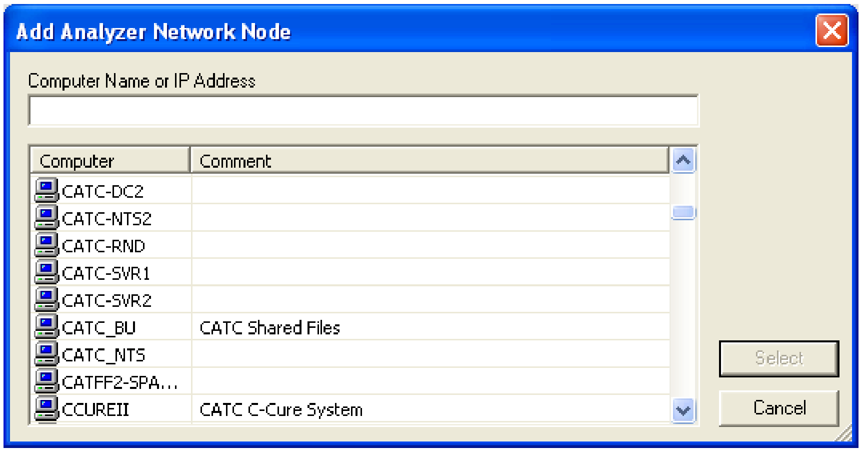
To browse to a networked Analyzer:

Step 1 From the menu, select **Setup > Analyzer Network**. The Analyzer Network dialog box opens. Locally connected Analyzers (connected via USB) display in the list. The list may also show PC hosts and Analyzers identified in previous sessions.



Computer	Analyzer devices
Local machine	PETracer SN:211, PETracer SN:202
QA-ABIT	PETracer SN:213

Step 2 If the desired host and Analyzer are not present in the list, click **Add**. The Add Analyzer dialog box opens.



Computer	Comment
CATC-DC2	
CATC-NTS2	
CATC-RND	
CATC-SVR1	
CATC-SVR2	
CATC_BU	CATC Shared Files
CATC_NTS	
CATFF2-SPA...	
CCUREII	CATC C-Cure System

Step 3 Browse to the PC host that has the Analyzer(s), then click **Select**. The selected host and its Analyzer appear in the Analyzer Network browse dialog.

Note: The Analyzer must be turned on before it appears in the list.

Step 4 Remove hosts and Analyzers you do not intend to use in the current recording session by selecting them from the list and clicking Remove.

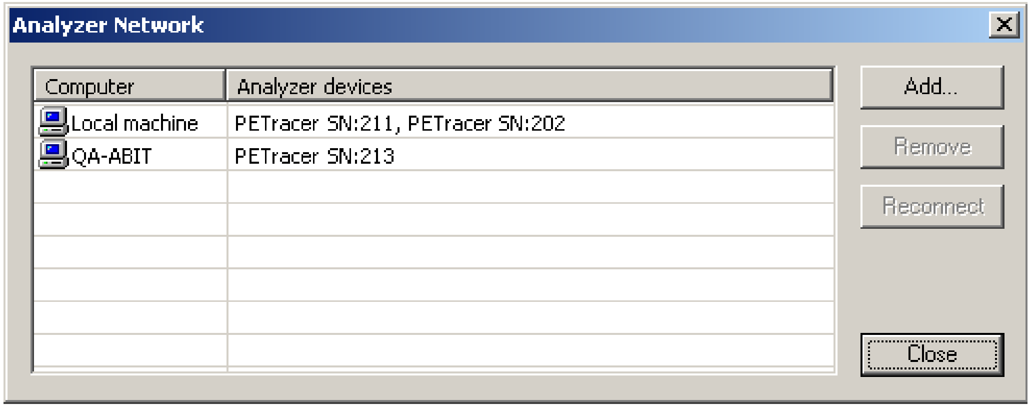
The PE*Tracer* software establishes a connection to the selected Analyzer.

4.19 Analyzer Chat Window

PE*Tracer* has a Chat window that allows you to communicate with users on remote PCs. For Chat to work, two conditions must be met:

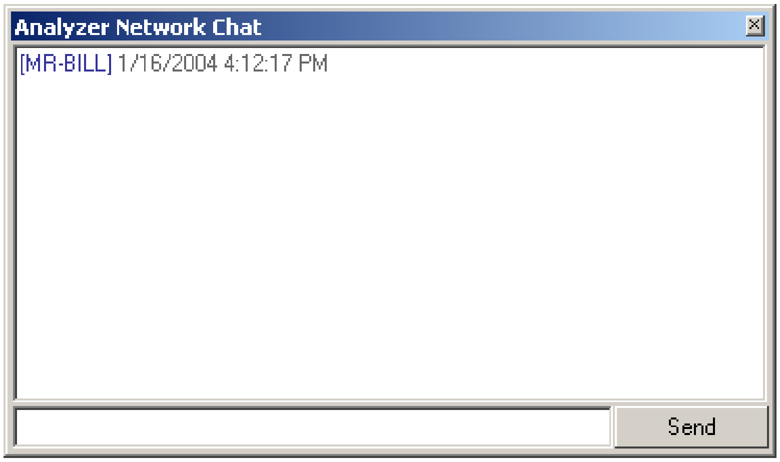
1. The PC hosts must have PE*Tracer* software installed and running.

2. The hosts must be listed in each other's Network Browse list. This means that your PC must have the remote PC listed in its Network Browse window and the remote PC must have your PC listed in its Network Browse window.



Computer	Analyzer devices
Local machine	PETracer SN:211, PETracer SN:202
QA-ABIT	PETracer SN:213

If the above conditions are met, a Chat session is initiated by running the command: **View > Analyzer Network Chat Bar**. The following dialog opens.



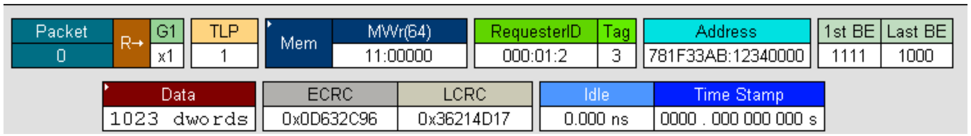
[MR-BILL] 1/16/2004 4:12:17 PM

Enter some text, then press **Send**. The message is then broadcast to all host PCs listed in your network browse window. If a target PC also lists your PC in its Network Browse window, then it can receive your message. When the message arrives, the Chat window automatically opens.

Chapter 5: Reading Traces

5.1 Viewing PCI Express Traces

PE*Tracer*™ displays traffic as labeled, color-coded, and time-stamped rows.



```
Packet  R→  G1  TLP   Mem  MWr(64)   RequesterID  Tag  Address            1st BE  Last BE
0           x1  1          11:00000  000:01:2     3    781F33AB:12340000  1111    1000

Data          ECRC        LCRC        Idle      Time Stamp
1023 dwords   0x0D632C96  0x36214D17  0.000 ns  0000 . 000 000 000 s
```

Tool tips provide details about fields within the trace. Hold the mouse cursor over a field to see a tool tip.

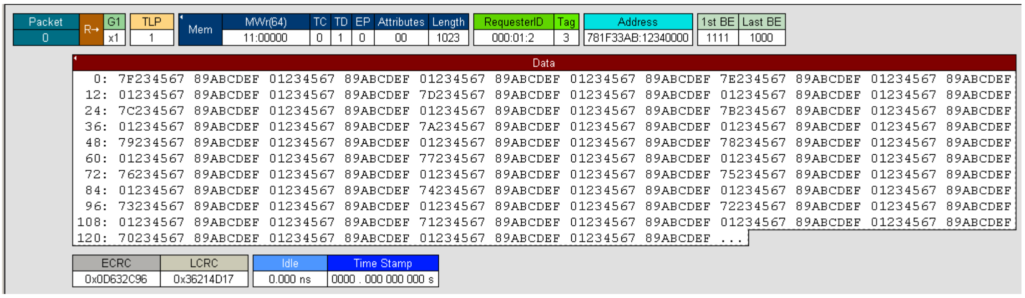
Additional information is available from pop-up menus. For example, if you click the left mouse button on the first cell in a packet a menu appears with an option to view Raw 10b Codes.

5.2 Expand and Collapse Data Fields

Packet data fields are displayed in a short format by default.

You can view a data field’s long format by performing one of the following three actions:

- Click the **small triangle** in the left corner.
- Double-click anywhere in the data field.
- Click once in the Data Field with the left mouse button, then choose **Expand Data** from the pop-up menu.



```
Packet  R→  G1  TLP   Mem  MWr(64)   TC  TD  EP  Attributes  Length  RequesterID  Tag  Address            1st BE  Last BE
0           x1  1          11:00000  0   1   0   00          1023    000:01:2     3    781F33AB:12340000  1111    1000

                                              Data
  0: 7F234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF 7E234567 89ABCDEF 01234567 89ABCDEF
 12: 01234567 89ABCDEF 01234567 89ABCDEF 7D234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF
 24: 7C234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF 7B234567 89ABCDEF 01234567 89ABCDEF
 36: 01234567 89ABCDEF 01234567 89ABCDEF 7A234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF
 48: 79234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF 78234567 89ABCDEF 01234567 89ABCDEF
 60: 01234567 89ABCDEF 01234567 89ABCDEF 77234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF
 72: 76234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF 75234567 89ABCDEF 01234567 89ABCDEF
 84: 01234567 89ABCDEF 01234567 89ABCDEF 74234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF
 96: 73234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF 72234567 89ABCDEF 01234567 89ABCDEF
108: 01234567 89ABCDEF 01234567 89ABCDEF 71234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF
120: 70234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF 01234567 89ABCDEF ...

ECRC        LCRC        Idle      Time Stamp
0x0D632C96  0x36214D17  0.000 ns  0000 . 000 000 000 s
```

A repeat of any above methods causes the display to return to a Short Data format.

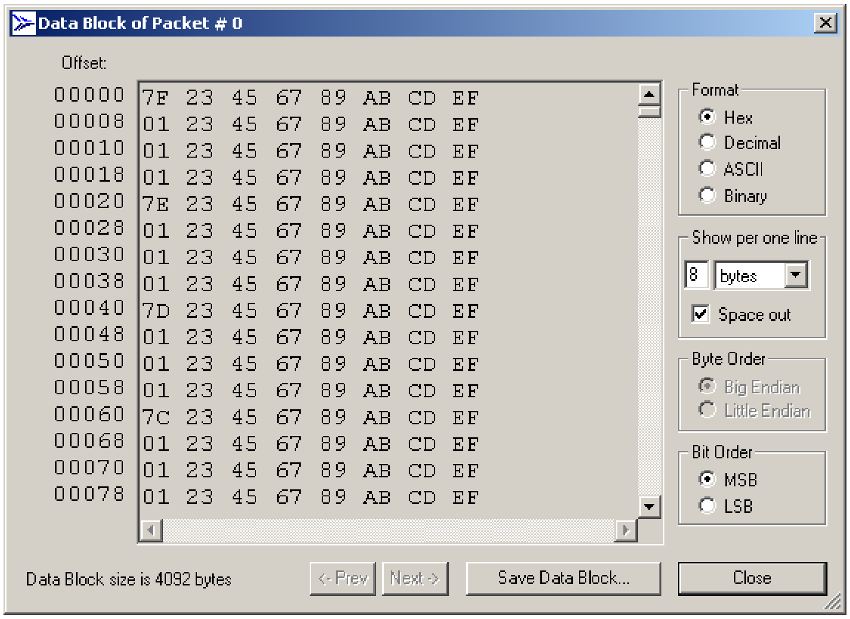
5.3 Resizing Cells

Data cells can be resized by pointing the mouse pointer on the edge of a data cell, depressing and holding the left mouse button, and then repositioning the mouse pointer while keeping the mouse button depressed.

5.4 View Data Block

To view the raw bits that make up the data in a data field:

- Left-click a data field and select **View Data Block** from the pop-up menu to display the Data Block window.



Data Block of Packet # 0

Offset:

```
00000 7F 23 45 67 89 AB CD EF
00008 01 23 45 67 89 AB CD EF
00010 01 23 45 67 89 AB CD EF
00018 01 23 45 67 89 AB CD EF
00020 7E 23 45 67 89 AB CD EF
00028 01 23 45 67 89 AB CD EF
00030 01 23 45 67 89 AB CD EF
00038 01 23 45 67 89 AB CD EF
00040 7D 23 45 67 89 AB CD EF
00048 01 23 45 67 89 AB CD EF
00050 01 23 45 67 89 AB CD EF
00058 01 23 45 67 89 AB CD EF
00060 7C 23 45 67 89 AB CD EF
00068 01 23 45 67 89 AB CD EF
00070 01 23 45 67 89 AB CD EF
00078 01 23 45 67 89 AB CD EF
```

Format
● Hex
○ Decimal
○ ASCII
○ Binary

Show per one line
8 bytes
☑ Space out

Byte Order
● Big Endian
○ Little Endian

Bit Order
● MSB
○ LSB

Data Block size is 4092 bytes

The window provides options for changing the format of the displayed data.

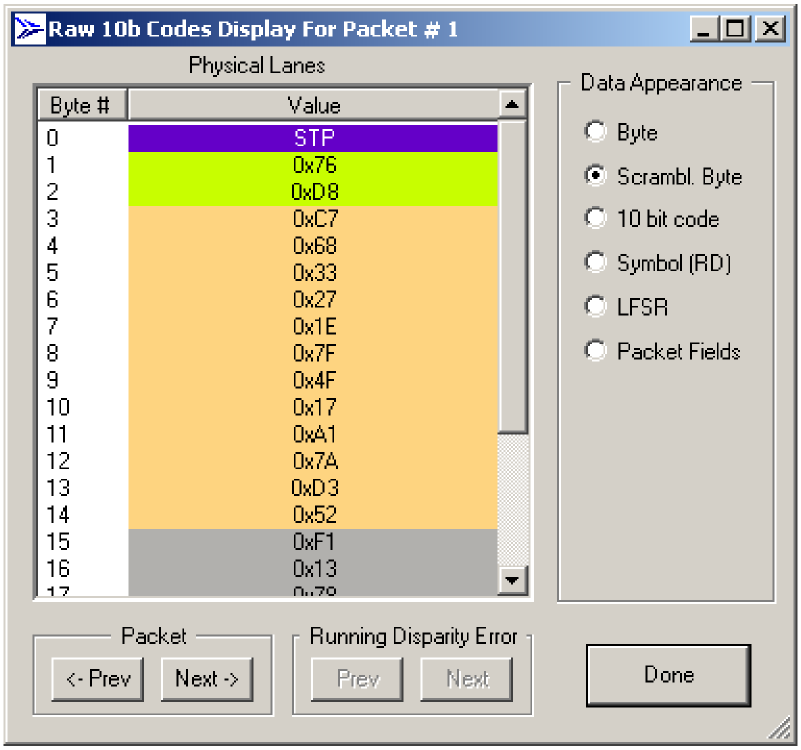
5.5 Show Raw 10b Codes

You can view the raw bits that make up the data field by left-clicking the field and selecting **View Data Block** from the pop-up menu.

To view Raw 10b Codes:

Step 1 Left-click the **first cell** in a packet.

Step 2 Select **Show Raw 10b Codes** from the pop-up menu to display the Raw 10b Codes window.



Byte #	Value
0	STP
1	0x76
2	0xD8
3	0xC7
4	0x68
5	0x33
6	0x27
7	0x1E
8	0x7F
9	0x4F
10	0x17
11	0xA1
12	0x7A
13	0xD3
14	0x52
15	0xF1
16	0x13

Data Appearance

○ Byte
● Scrambl. Byte
○ 10 bit code
○ Symbol (RD)
○ LFSR
○ Packet Fields

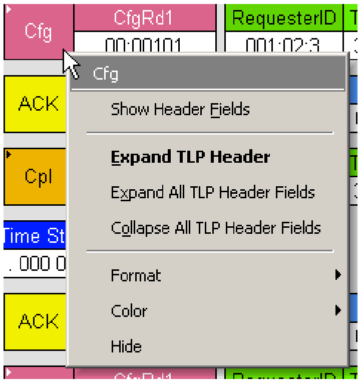
Step 3 To change the format of the data, use the options along the right side of the dialog.

Step 4 To navigate the trace, use the **Prev** and **Next** buttons.

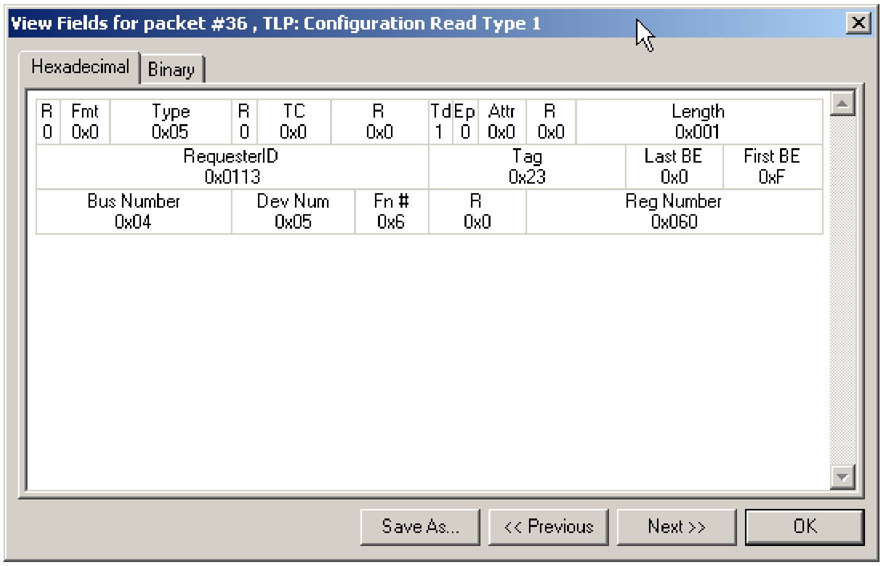
5.6 Show Header Fields

You can view details about header fields by opening the Show Header Fields dialog box.

Step 1 Click a **header.** A pop-up menu appears.



Step 2 Select **Show Header Fields**. The following dialog box opens.



View Fields for packet #36 , TLP: Configuration Read Type 1

R 0	Fmt 0x0	Type 0x05	R 0	TC 0x0	R 0x0	Td 1	Ep 0	Attr 0x0	R 0x0	Length 0x001
RequesterID 0x0113						Tag 0x23			Last BE 0x0	First BE 0xF
Bus Number 0x04			Dev Num 0x05		Fn # 0x6	R 0x0		Reg Number 0x060		

Step 3 Use the **Prev** and **Next** buttons to navigate to other headers.

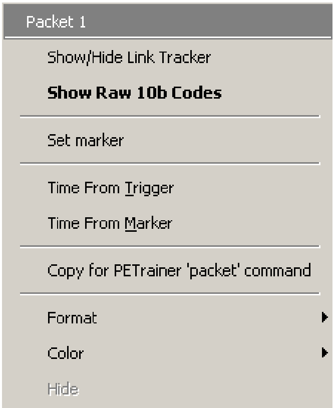
5.7 Pop-up Menus

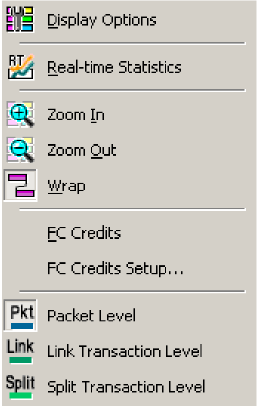
The Analyzer software makes extensive use of pop-up menus. In some instances, pop-up menus provide the only means of accessing dialog boxes that contain detailed information about cells within the trace, for example, the Show Configuration Space dialog box.

To see a pop-up menu, left-click or right-click a **cell** within the trace. Right-click or left-click behavior depends on the Display Options setting. For default left-click, the right-click menu is not cell-dependent. For other left-click behavior, the type of menu that opens varies depending on the type of cell that is selected. Take some time to explore traces and the various pop-up menus.

Packet Heading Cell (first cell in the packet) Popup Menu

The left-click pop-up menu includes Show Raw 10b Codes.





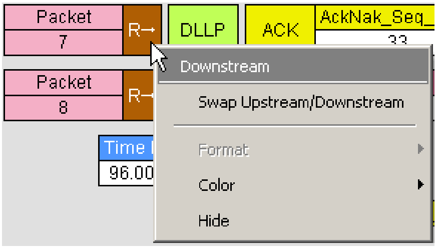
The right-click pop-up menu from same cell is also shown.

Packet Header Cell Popup Menu

Packet Header cell display pop-up menu has an option called Show Header Fields (see "Show Header Fields" on page 104) that exposes a detailed view of the selected Header field.

Packet Directionality Cell (second cell in the packet) Popup Menu

The Swap Upstream/Downstream command changes the directionality of the packets in the trace.

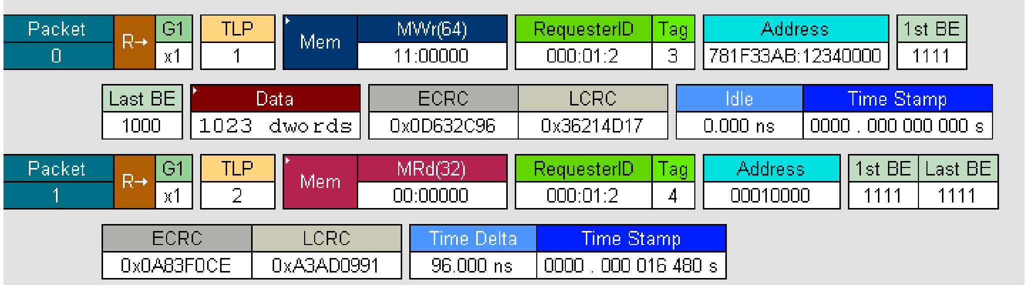


5.8 Decoding Traffic

The PE*Tracer* software has three decode levels:

Packet

Packet level decode Pkt includes all TLP and DLLP packets and all ordered sets.



Packet	R→	G1	TLP	Mem	MWr(64)	RequesterID	Tag	Address	1st BE
0		x1	1		11:00000	000:01:2	3	781F33AB:12340000	1111

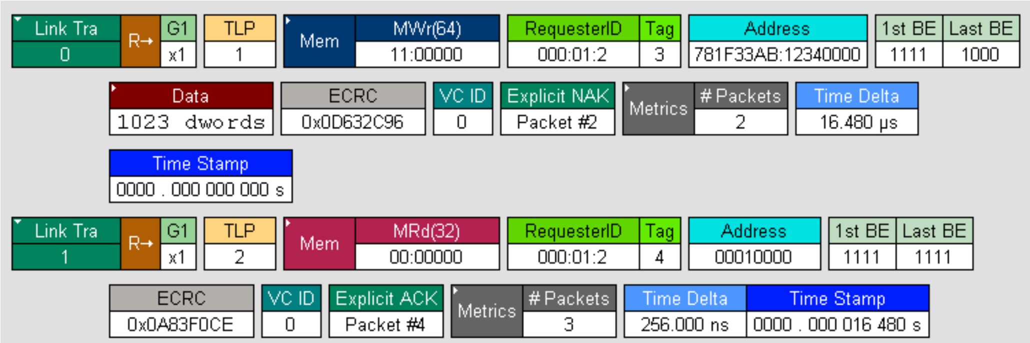
Last BE	Data	ECRC	LCRC	Idle	Time Stamp
1000	1023 dwords	0x0D632C96	0x36214D17	0.000 ns	0000 . 000 000 000 s

Packet	R→	G1	TLP	Mem	MRd(32)	RequesterID	Tag	Address	1st BE	Last BE
1		x1	2		00:00000	000:01:2	4	00010000	1111	1111

ECRC	LCRC	Time Delta	Time Stamp
0x0A83F0CE	0xA3AD0991	96.000 ns	0000 . 000 016 480 s

Link

Link level decode Link is composed of TLP packet matched with a corresponding ACK or NAK coming from the opposite direction.



Link Tra	R→	G1	TLP	Mem	MWr(64)	RequesterID	Tag	Address	1st BE	Last BE
0		x1	1		11:00000	000:01:2	3	781F33AB:12340000	1111	1000

Data	ECRC	VC ID	Explicit NAK	Metrics	# Packets	Time Delta
1023 dwords	0x0D632C96	0	Packet #2		2	16.480 µs

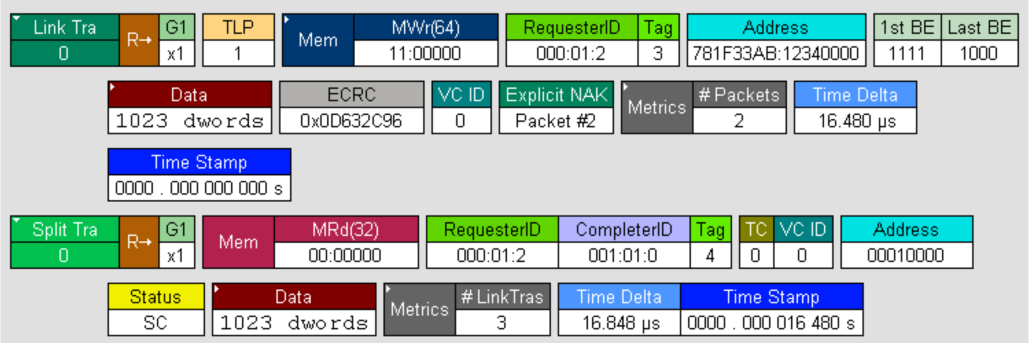
Time Stamp
0000 . 000 000 000 s

Link Tra	R→	G1	TLP	Mem	MRd(32)	RequesterID	Tag	Address	1st BE	Last BE
1		x1	2		00:00000	000:01:2	4	00010000	1111	1111

ECRC	VC ID	Explicit ACK	Metrics	# Packets	Time Delta	Time Stamp
0x0A83F0CE	0	Packet #4		3	256.000 ns	0000 . 000 016 480 s

Split

Split level decode Split is composed of two Link transactions, the Request TLP and the Completion TLP from the other direction.



Link Tra	R→	G1	TLP	Mem	MWr(64)	RequesterID	Tag	Address	1st BE	Last BE
0		x1	1		11:00000	000:01:2	3	781F33AB:12340000	1111	1000

Data	ECRC	VC ID	Explicit NAK	Metrics	# Packets	Time Delta
1023 dwords	0x0D632C96	0	Packet #2		2	16.480 µs

Time Stamp
0000 . 000 000 000 s

Split Tra	R→	G1	Mem	MRd(32)	RequesterID	CompleterID	Tag	TC	VC ID	Address
0		x1		00:00000	000:01:2	001:01:0	4	0	0	00010000

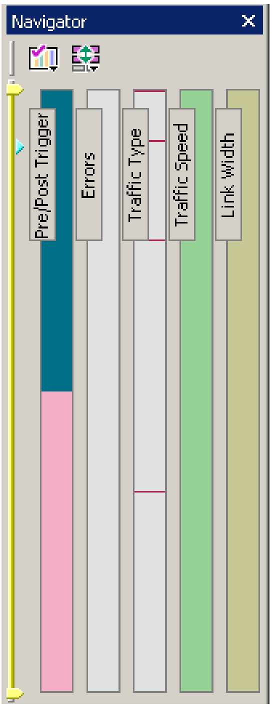
Status	Data	Metrics	# LinkTras	Time Delta	Time Stamp
SC	1023 dwords		3	16.848 µs	0000 . 000 016 480 s

5.9 Using the Navigator

The trace Navigator is a tool for navigating within the trace. It allows you to view the location of errors and triggers in a trace and to narrow the range of traffic on display. It also allows you to quickly jump to any point in the trace.

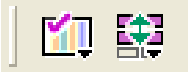
Displaying the Navigator

Click in the toolbar to display the Navigator. The Navigator appears on the right side of the Main window. It has a two-button toolbar and a vertical slider bar. It also has colored panes for navigating the trace in different ways. You set which panes are displayed through Navigator pop-up menus.



Navigator Toolbar

The Navigator toolbar lets you quickly set Navigator features. The toolbar has two buttons.



Navigator Ranges: This button brings up a pop-up menu that lets you reset the Navigator range. The range determines what packets are viewable in the trace display.

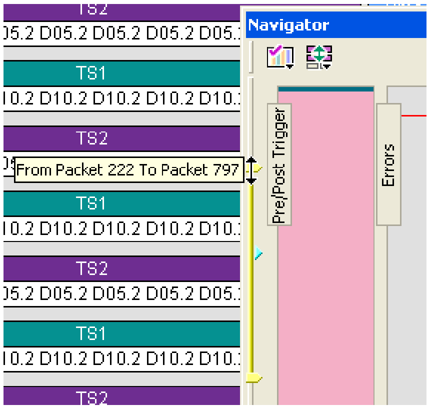
Navigator Panes: This button has two purposes: To select which Navigator panes appear and to bring up the Navigator legend. The legend determines how information is shown in the panes.

Navigator Ranges

You set the viewing range by dragging the **yellow range delimiters** along the slider.

To set the lowest packet viewable, drag the **top delimiter up**. As you do so, a tool tip appears to indicate the current range. Stop dragging when you reach the desired lowest packet.

To set the highest packet viewable, drag the **bottom delimiter down**. Stop when the tool tip indicates you are at the desired highest packet.


TS2
)5.2 D05.2 D05.2 D05.2 D05.
TS1
I0.2 D10.2 D10.2 D10.2 D10.
TS2
From Packet 222 To Packet 797
TS1
I0.2 D10.2 D10.2 D10.2 D10.
TS2
)5.2 D05.2 D05.2 D05.2 D05.
TS1
I0.2 D10.2 D10.2 D10.2 D10.
TS2
Navigator
Pre/Post Trigger
Errors

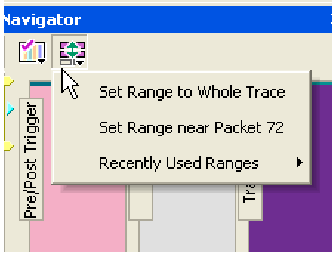
To Determine Current Position

In addition to the two range delimiters, the slider has a **blue current-position** indicator (see above). The current-position indicator shows where you are in the trace display with respect to the possible viewing range.

For example, suppose you set viewing range to packet 0 through packet 500 (the top range delimiter is at packet 0, and the bottom range delimiter is at packet 500). If you then move the current-position indicator on the slider to midway between the top and bottom delimiters, then packet 250 appears in the middle of the trace display.

To Reset Navigator Range

You can reset the Navigator range using the toolbar **Navigator Range** button. Press the button to bring up the Navigator Range drop-down menu.

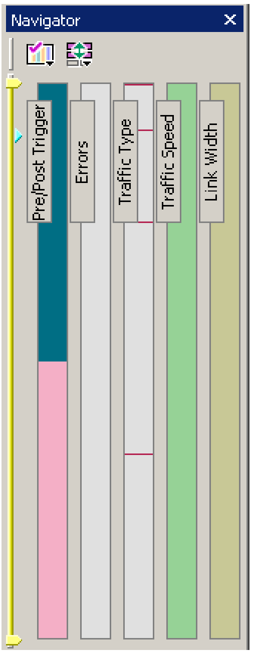


The menu has the following options:

- **Set Range to Whole Trace**: Allows you to reset the range to include the entire trace file contents. The top range delimiter is placed at the lowest packet number in the trace. The bottom range delimiter is placed at the highest packet number in the trace.
- **Set Range Near Packet xxx**: Allows you to collapse the range so that only the packets immediately above and below the xxx packet are displayed. The xxx packet is whatever packet is currently at the top in the trace display.
- **Recently Used Ranges**: Allows you to reset the range to any of a number of recently used (previously set) ranges.

Navigator Panes

You can display any combination of trace Navigator panes.



From left to right, the panes are: Pre/Post Trigger, Errors, Traffic Type, Traffic Speed, and Link Width. Each pane represents the entire trace with respect to different types of information. The top of each pane represents the start of the trace file, and the bottom represents the end of the trace file.

- **Pre/Post Trigger:** To view the trigger event in the trace and the relative size of pre-trigger and post-trigger portions of the trace. The two portions are set apart as different colors. The trigger event occurs at the point the two colors meet.
- **Errors:** To view any errors in the trace. A thin red line represents each error in the pane.
- **Traffic Type:** To view the types of packets that occur in the trace. A different color represents each packet type in the pane. The relative size of colored portions in the pane corresponds to the amounts of the various packet types in the trace. As described below, you can use the Navigator legend to change the types of packets that take precedence in the display.
- **Traffic Speed:** To view the speed that occurs in the trace.
- **Link Width:** To view the link width that occurs in the trace.

To Show/Hide Navigator Panes

You can show/hide any of the panes using pop-up menus accessible through left-click the **Navigator Panes** button or by right-click anywhere in any Trace Navigator pane.

Navigator Slider

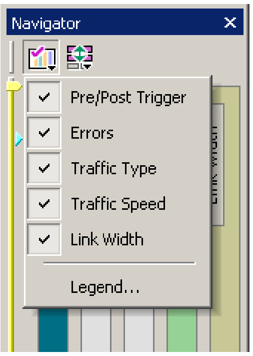
The Navigator slider appears at the left of Navigator panes. The slider has **yellow upper and lower range delimiters** and a **blue current-position** indicator.

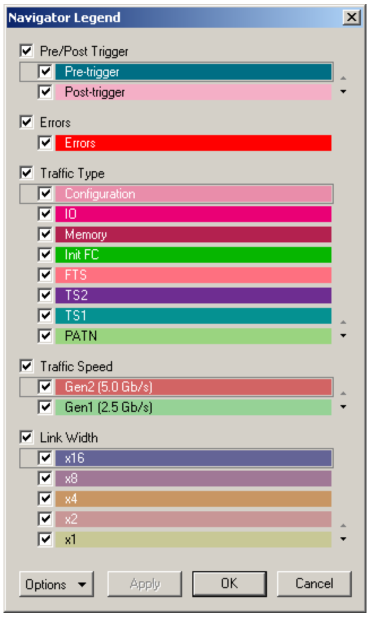
The Navigator slider lets you to set the range of packets viewable in the trace display. In other words, it sets scrolling range of the display. You can scroll the display up to the lowest packet number in the viewing range. You can scroll the display down to the highest packet number in the viewing range.

Trace Navigator Legend

The Navigator legend lets you control the display of content in Navigator panes.

You bring up the legend through the Navigator Panes drop-down menu. Press the toolbar **Navigator Panes** button to access the menu. Select the **Legend** option to bring up the Navigator Legend dialog box.





☑ Pre/Post Trigger
- ☑ Pre-trigger
- ☑ Post-trigger

☑ Errors
- ☑ Errors

☑ Traffic Type
- ☑ Configuration
- ☑ IO
- ☑ Memory
- ☑ Init FC
- ☑ FTS
- ☑ TS2
- ☑ TS1
- ☑ PATN

☑ Traffic Speed
- ☑ Gen2 (5.0 Gb/s)
- ☑ Gen1 (2.5 Gb/s)

☑ Link Width
- ☑ x16
- ☑ x8
- ☑ x4
- ☑ x2
- ☑ x1

The Navigator Legend dialog box has areas corresponding to each of the panes. Each area has check boxes that allow you to hide/display information in the pane. You can set the priority of information displayed in the panes using the up and down triangles on the right.

Using the Legend to Show/Hide Navigator Panes

To use the legend to show/hide an entire pane, use the **checkbox** next to the name of each pane in the legend.

In the case of the Pre/Post Trigger and Errors areas, the action of show/hide in the legend is identical to that provided by Trace Navigator pop-up menus.

In the case of the Traffic Types pane, there is no equivalent show/hide available through the pop-up menus.

Using the Legend to Set the Priority of Information Display

You can use the legend to set the priority of information displayed in the Pre/Post Trigger Traffic Type panes. This is a two-step process.

Step 1 For a particular item in a pane, click the **column next to the checkbox** for the item. That labels the item as currently active.

Step 2 Next, use the **up-down** at the lower-right of the area to move the item higher or lower in priority.

In the case of the Traffic Type pane, priority determines display priority of each packet type. For portions of the trace that are dominated by a particular packet type, this setting no effect: only the color corresponding to that packet type is displayed in that portion of the pane. Suppose, however, that part of the trace includes equal or near equal numbers of several types of packets. In that case, you can use the legend to select which among those types is represented in that portion of the Traffic Types pane. This allows you to view only packets of interest in crowded portions of the trace display.

5.10 Configuration Space Dialog

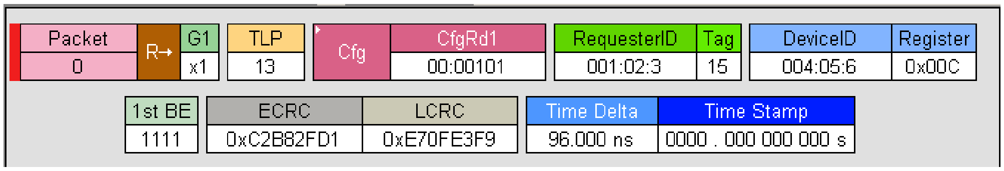
To view the Configuration Space, select **Report > Configuration Space > <device>".**

The Configuration Space dialog box displays information about the Configuration Space state as of the current packet of the currently selected device.

To access the Show Configuration Space dialog box:

Step 1 Open a trace, such as the sample file **cfg_pci_express.pex**.

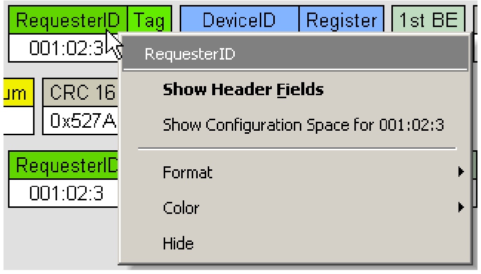
Step 2 Scroll to a packet with a Configuration header.



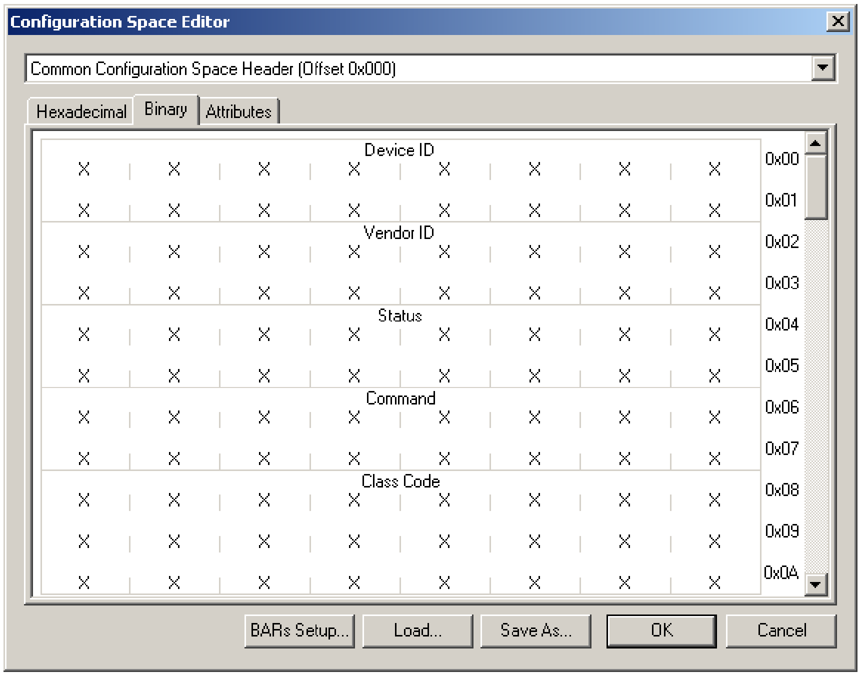
Packet	R→	G1	TLP	Cfg	CfgRd1	RequesterID	Tag	DeviceID	Register
0		x1	13		00:00101	001:02:3	15	004:05:6	0x00C

1st BE	ECRC	LCRC	Time Delta	Time Stamp
1111	0xC2B82FD1	0xE70FE3F9	96.000 ns	0000 . 000 000 000 s

Step 3 Click a **RequesterID, CompeterID**, or **DeviceID** field. A pop-up menu opens.



Step 4 Select **Show Configuration Space for xxxx** from the menu, where **xxxx** is the device number. The Show Configuration Space dialog box opens.



Common Configuration Space Header (Offset 0x000)

Device ID								
X	X	X	X	X	X	X	X	0x00
X	X	X	X	X	X	X	X	0x01
Vendor ID								
X	X	X	X	X	X	X	X	0x02
X	X	X	X	X	X	X	X	0x03
Status								
X	X	X	X	X	X	X	X	0x04
X	X	X	X	X	X	X	X	0x05
Command								
X	X	X	X	X	X	X	X	0x06
X	X	X	X	X	X	X	X	0x07
Class Code								
X	X	X	X	X	X	X	X	0x08
X	X	X	X	X	X	X	X	0x09
X	X	X	X	X	X	X	X	0x0A

Step 5 The dialog box gives you several options:

- **Hexadecimal vs. Binary tabs**: Presents the data in different formats.
- **Combo box**: At top of dialog with menu of Read and Write Configuration Spaces present in the current trace.
- **First, Prev, Next, and Last buttons**: For navigating to other Read and Write Configuration States in the trace.

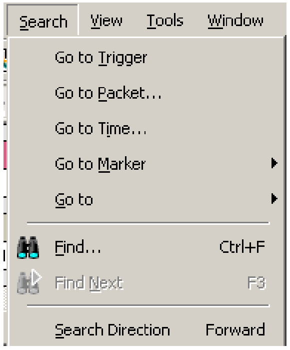
Chapter 6: Searching Traces

This chapter describes how to search for trace events.

6.1 Trace Search Overview

Several search commands let you navigate a trace view to search for key events, such as errors and triggers.

To view the search options, click **Search** in the Menu bar.



6.2 Go to Trigger

To position a trigger packet at the top of the screen:

- Select **Search > Go to Trigger**.

6.3 Go to Packet

To position a packet at the top of the screen:

Step 1 Select **Go to Packet** from the Search menu. A pop-up menu prompts you for the packet number, marker, or time.

Step 2 Enter the packet number, marker, or time.

Step 3 Click **OK**.

6.4 Go to Time

To position a specific time at the top of the screen:

Step 1 Select **Go to Time**from the Search menu. A pop-up menu prompts you for the time in Seconds or Clocks.

Step 2 Enter the time.

Step 3 Click **OK**.

6.5 Go To Marker

Use this operation to go directly to a specific packet that has been marked with a unique marker by the Set Marker operation.

To go to a marker:

Step 1 Select **Go To Marker** from the Search Menu.

Step 2 Select the marker you want from the fly-out menu.

The packet you want appears at the top of the screen. Marked packets have a red bar on the left edge.

6.6 Go To Menu

The **Go To menu** in the Search menu provides a quick way to search for a packet based on a simple condition. You can search for the following types of events:

- TLP Type
- DLLP Type
- Ordered Set
- Link Event
- Traffic Class
- DLLP Virtual Channel
- TLP Virtual Channel
- Direction
- Speed
- Link Width
- Requester ID
- Completer ID
- Data Lengths
- Errors

6.7 Search Direction

Search direction can be toggled back and forth by using the command under the Search menu. Search Direction controls the direction of the search. Each time Search Direction is selected, the search order is reversed. For example, if the previous search was **forward**, choosing **Search Direction** toggles the current search to **backward**.

To verify the direction of a search look at the lower right corner of the screen. **Search: Bwd** or **Search: Fwd** should appear. If a direction is not indicated, it means that the status bar is turned off.

To turn on the Status bar, select from the menu **View > Status Bar**.

Keyboard shortcuts can also be used to control search direction:

- **Control-f** means Search Forward.
- **Control-b** means Search Backward.

6.8 Find

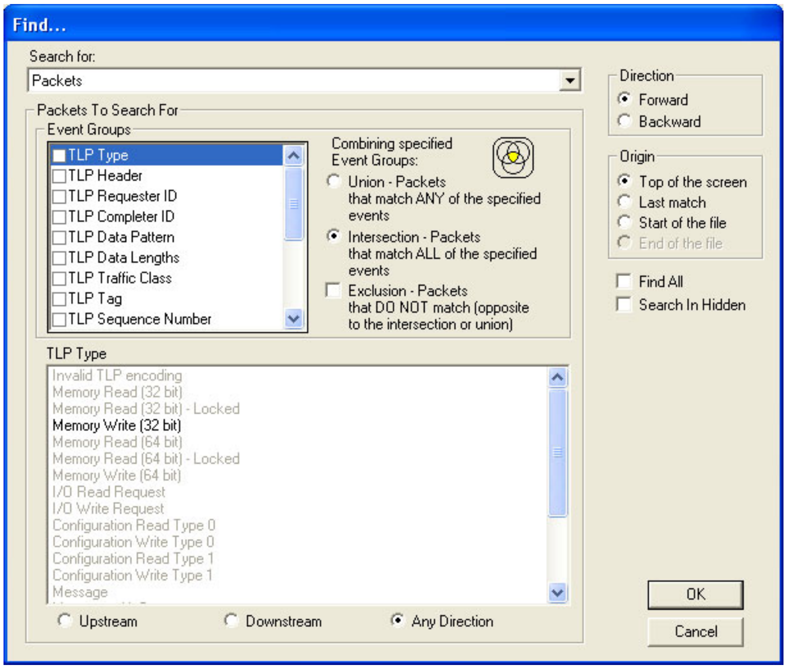
Find allows you to conduct complex searches in a trace. You can search by protocol level and by Event Groups, such as Errors and DLLP Types.

The options Union, Intersection, and Exclusion allow you to create complex searches such as "Find x OR y" or "Exclude all x or y."

To find a item:

Step 1 Open a trace.

Step 2 Open Find by selecting **Search > Find** from the menu or clicking .



Find...

Search for: Packets

Packets To Search For

Event Groups

- ☐ TLP Type
- ☐ TLP Header
- ☐ TLP Requester ID
- ☐ TLP Completer ID
- ☐ TLP Data Pattern
- ☐ TLP Data Lengths
- ☐ TLP Traffic Class
- ☐ TLP Tag
- ☐ TLP Sequence Number

Combining specified Event Groups:

○ Union - Packets that match ANY of the specified events

● Intersection - Packets that match ALL of the specified events

☐ Exclusion - Packets that DO NOT match (opposite to the intersection or union)

TLP Type

- Invalid TLP encoding
- Memory Read (32 bit)
- Memory Read (32 bit) - Locked
- Memory Write (32 bit)
- Memory Read (64 bit)
- Memory Read (64 bit) - Locked
- Memory Write (64 bit)
- I/O Read Request
- I/O Write Request
- Configuration Read Type 0
- Configuration Write Type 0
- Configuration Read Type 1
- Configuration Write Type 1
- Message

○ Upstream ○ Downstream ● Any Direction

Direction

● Forward

○ Backward

Origin

● Top of the screen

○ Last match

○ Start of the file

○ End of the file

☐ Find All

☐ Search In Hidden

Step 3 From the Search For menu, select a display level such as **Packets**. To search multiple levels, finish steps 3 to 9, and then repeat 3 to 9 for each additional display level.

Step 4 From the Event Groups menu, select an **Event Group**. The menu in the far right of the dialog box is context sensitive and changes to reflect the options for that group.

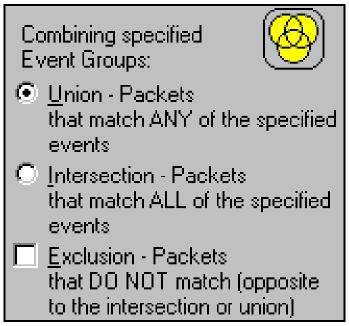
Step 5 In the menu in the far right, select one or more items.

Step 6 Repeat Step 5 for each Event Group of interest for the selected display level.

Step 7 Under Direction, select **Forward** or **Backward** to select a search direction.

Step 8 Under Origin, select a starting point for the search.

Step 9 Under **Combining Specified Event Groups**, select



Combining specified Event Groups:

● Union - Packets that match ANY of the specified events

○ Intersection - Packets that match ALL of the specified events

□ Exclusion - Packets that DO NOT match (opposite to the intersection or union)

The options Union, Intersection, and Exclusion let you set conditions on your searches:

- **Union**: To search for **any** of the selected criteria.
  Example: "Find packets with ANY of the following characteristics …"
- **Intersection**: To search for **all** of the selected criteria.
  Example: "Find packets with ALL of the following characteristics …"
- **Exclude**: To **exclude** items from a search. This option works in conjunction with Union and Intersection. You select Union and Exclude to exclude any of the specified traffic. You select Intersection and Exclude to exclude all of the specified traffic.
  Example: "Exclude packets with ANY of the following …" or
  "Exclude packets with ALL of the following …"

Step 10 Repeat Steps 3 through 9 for additional display levels.

Step 11 Click **OK**.

Event Groups

The Event Groups for Packets are:

- TLP Type
- TLP Header
- TLP Requester ID
- TLP Completer ID
- TLP Data Pattern
- TLP Data Lengths
- TLP Traffic Class
- TLP Tag
- TLP Sequence Number
- DLLP Type
- DLLP Header
- DLLP Virtual Channel
- ACK/NAK Seq Number
- Ordered Sets
- Link Event
- Direction
- Errors

The Event Groups for Link Transactions and Split Transactions are:

- TLP Type
- Traffic Class
- Virtual Channel
- Direction
- RequesterID
- CompleterID
- Ordered Sets
- Staus
- Tag

6.9 Search for the Next Packet Type

Use **Find Next** to search for the next packet meeting the search criteria.

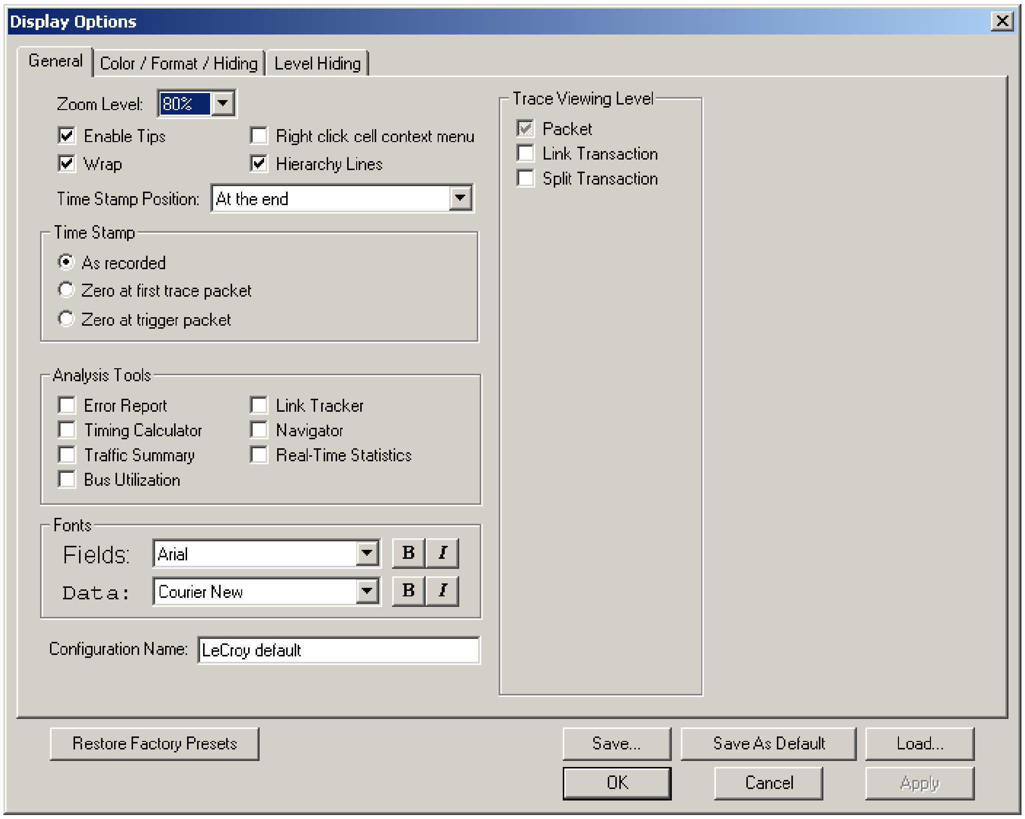
Chapter 7: Display Options

7.1 Setting Display Options

Display Options allow you to customize the colors and formats of displayed traffic.

To open the Display Options dialog at the General tab, Select **Setup > Display Options** from the menu or

click the **Display Options** button on the toolbar.



Display Options

General | Color / Format / Hiding | Level Hiding

Zoom Level: 80%

☑ Enable Tips ☐ Right click cell context menu

☑ Wrap ☑ Hierarchy Lines

Time Stamp Position: At the end

Time Stamp
- ● As recorded
- ○ Zero at first trace packet
- ○ Zero at trigger packet

Analysis Tools
- ☐ Error Report ☐ Link Tracker
- ☐ Timing Calculator ☐ Navigator
- ☐ Traffic Summary ☐ Real-Time Statistics
- ☐ Bus Utilization

Fonts
- Fields: Arial
- Data: Courier New

Configuration Name: LeCroy default

Trace Viewing Level
- ☑ Packet
- ☐ Link Transaction
- ☐ Split Transaction

7.2 Specifying General Display Options

The Display Options General tab allows you to specify:

Zoom Level: Defines the size of packet fields in the packet view. Zoom level is adjustable as 10, 20, 40, 60, 80, 100, 120, 140, 160, 180 and 200 percent.

Enable Tool tips: Allows information to be displayed on a packet by resting your mouse pointer over it.

Wrap: Allows packets, Link Transactions, and Split Transactions to wrap within the display.

Right click cell context menu: Swaps mouse functions.

Timestamp at the beginning: Swaps the location of the timestamp from the end of the packet to the beginning.

Trace Viewing Level: Allows the trace to be displayed in different hierarchical levels: Packets, Link Transactions, and Split Transactions.

Time Stamp: Gives you options for setting the timestamp to zero for either the first trace packet or the trigger packet or for leaving the stamp unchanged as it was originally recorded.

Analysis Tools Checkboxes

Error Report: Opens Traffic Summary window and displays list of errors that occurred in the trace.

Timing Calculator: Opens Timing Calculator dialog for calculating timing between events.

Traffic Summary: Displays a table of events that occurred in the trace.

Bus Utilization: Opens a graph of bus usage in the open trace.

Link Tracker: Opens window that displays traffic in a strictly chronological view.

Navigator: Displays/Hides Navigation bar. The Navigation bar lets you see in a glance the position of errors and triggers in a trace and narrow the range of traffic displayed in the trace window.

Real-Time Statistics: Opens a dialog that displays a graph of bus activity on the DUT in real-time.

Fonts

Fonts: Allows the appearance of field text and/or data text to be defined.

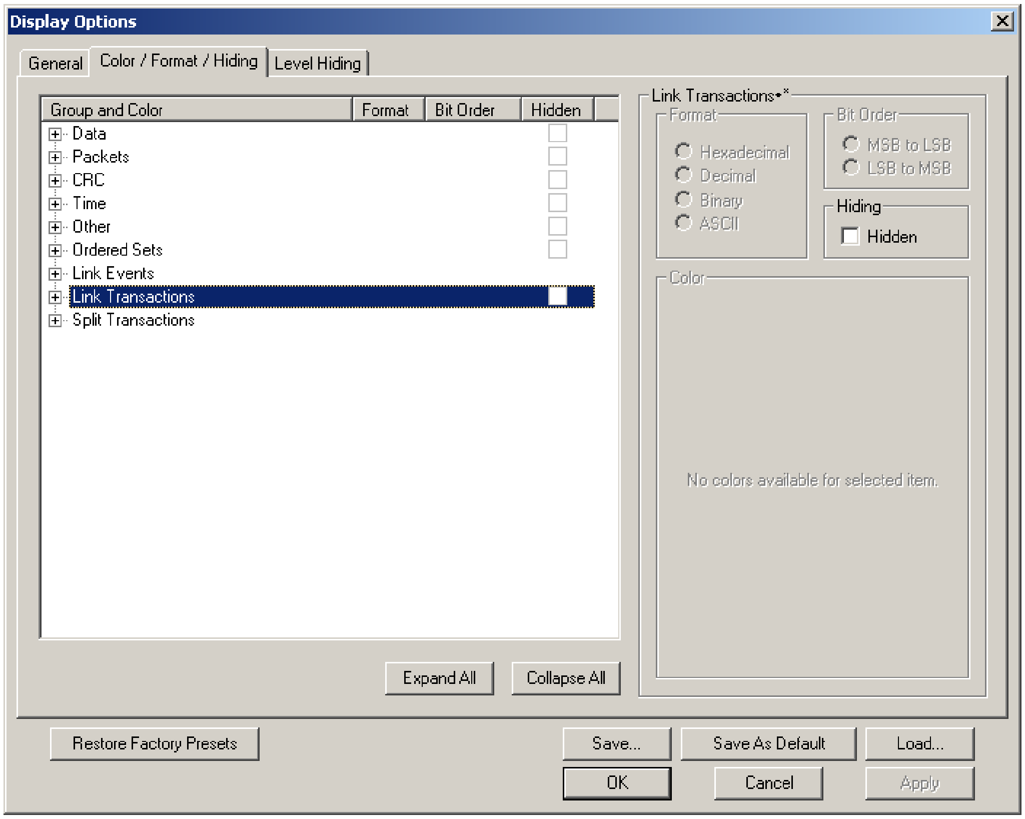
Configuration Name

Display Configuration Name: This field is used to enter a text string to be associated with the current set of Display Options. This name is saved with the Display Options file, and appears as part of the title of the packet view window that uses this set of Display Options.

7.3 Color, Format, and Hiding Options

The Color/Format/Hiding tab allows you to customize the colors and formats associated with each field in the trace and to selectively hide fields or packets. You access these display options by selecting **Setup > Display Options > Color/Format/Hiding** tab.

The Color/Format/Hiding property page lets you set how fields display in a trace. This property page lets you set field color and data format (binary, hex, decimal, ASCII), and lets you hide selected fields from the display.



Setting Field Colors

The Field Colors tab allows you to customize the colors associated with each field used in the packet view.

You may experiment with this option to achieve a color combination that suits you.

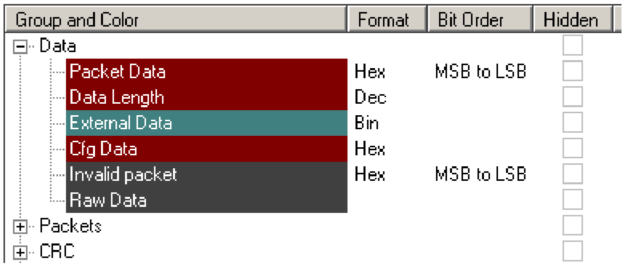
Select or change the trigger color using the color buttons labeled **- Packet #** and **+ Packet #** (before and after trigger) found under the Packet# section of the Field Colors window.

You select or change a color by clicking the appropriate color button. This action causes a color palette to pop up. Select the desired color and press OK.

Step 1 Click **View > Set Display Options** to open the Display Options dialog box.

Step 2 Select the **Color/Format/Hiding** property page.

Step 3 Under the Group and Colors column, click the **plus** symbol (**+**) next to the group you want to reformat. The group expands to show the individual fields within the group. Each field has a color, as shown below:



Group and Color	Format	Bit Order	Hidden
⊟ Data			□
Packet Data	Hex	MSB to LSB	□
Data Length	Dec		□
External Data	Bin		□
Cfg Data	Hex		□
Invalid packet	Hex	MSB to LSB	□
Raw Data			□
⊞ Packets			□
⊞ CRC			□

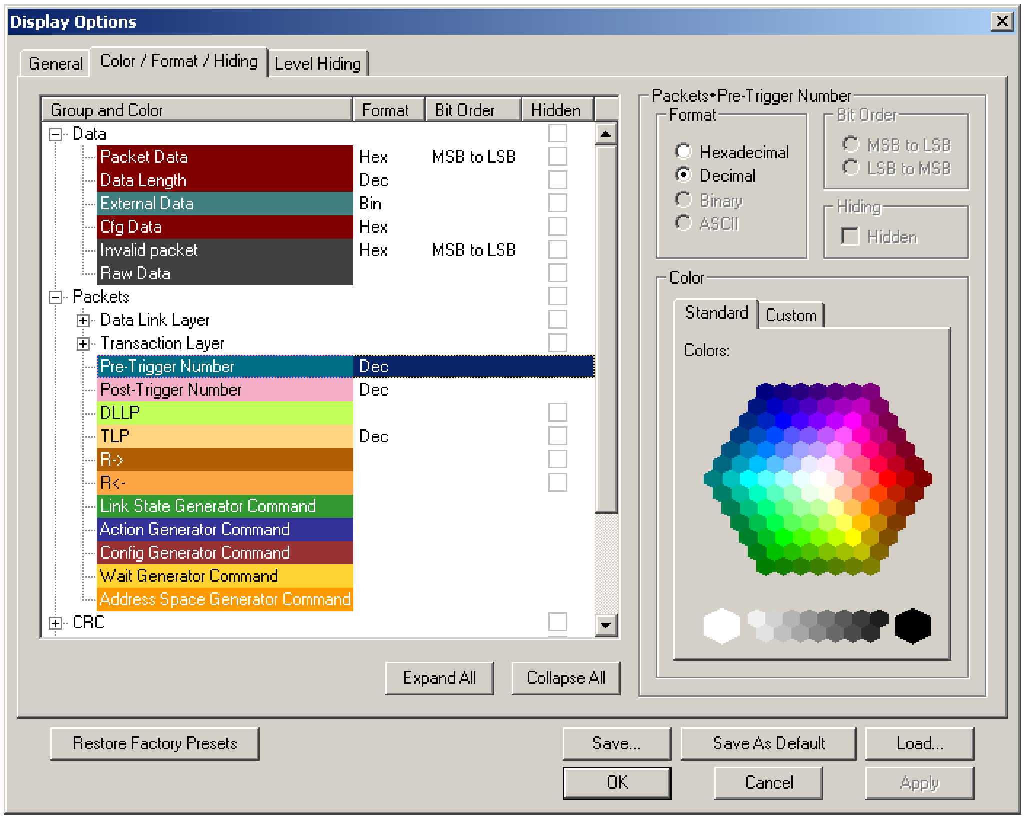
Step 4 Click the colored cell that you want to change. A color palette appears.

Step 5 Click a color in the palette, then click **Apply** or **OK**.

Note: The colors of the following Frame types cannot be changed:

- Invalid Data (frame error) field (red)
- Softbit Errors (yellow)

You can also change color by left-clicking a field in the trace and selecting Color from the pop-up menu.



Group and Color	Format	Bit Order	Hidden
⊟ Data			□
Packet Data	Hex	MSB to LSB	□
Data Length	Dec		□
External Data	Bin		□
Cfg Data	Hex		□
Invalid packet	Hex	MSB to LSB	□
Raw Data			□
⊟ Packets			□
⊞ Data Link Layer			□
⊞ Transaction Layer			□
Pre-Trigger Number	Dec		
Post-Trigger Number	Dec		
DLLP			□
TLP	Dec		□
R->			□
R<-			□
Link State Generator Command			
Action Generator Command			
Config Generator Command			
Wait Generator Command			
Address Space Generator Command			
⊞ CRC			□

Packets•Pre-Trigger Number

Format: ○ Hexadecimal ● Decimal ○ Binary ○ ASCII

Bit Order: ○ MSB to LSB ○ LSB to MSB

Hiding: □ Hidden

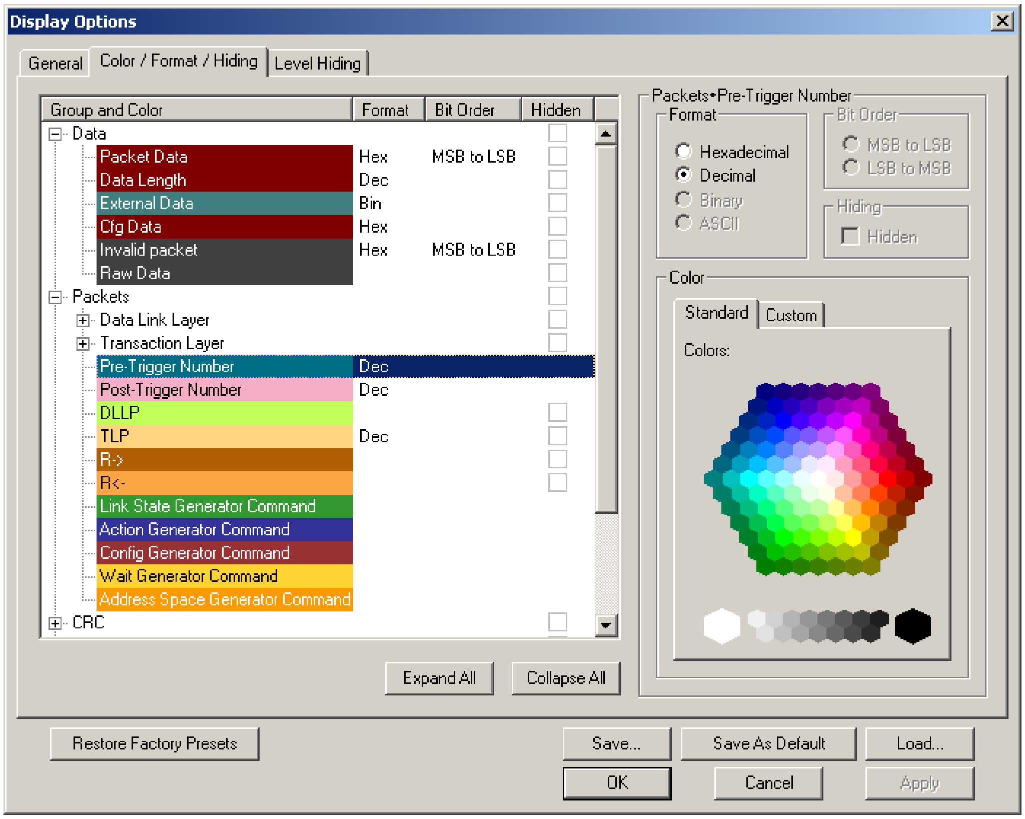
Color: Standard | Custom

Colors:

Changing Field Formats

The Field Formats tab allows you to define the way various numeric fields are shown in the packet display. You can select Hexadecimal, Decimal, Binary, or ASCII for certain fields or groups of fields.

To change a field's format, click the plus sign (+) next to a field in the list. This causes the selected item to expand so you can see its constituent sub-fields. Select a sub-field, and then choose the format from the formatting choices that appear at the bottom of the window.



Group and Color	Format	Bit Order	Hidden
Data			□
Packet Data	Hex	MSB to LSB	□
Data Length	Dec		□
External Data	Bin		□
Cfg Data	Hex		□
Invalid packet	Hex	MSB to LSB	□
Raw Data			□
Packets			□
Data Link Layer			□
Transaction Layer			□
Pre-Trigger Number	Dec		
Post-Trigger Number	Dec		
DLLP			□
TLP	Dec		□
R->			□
R<-			□
Link State Generator Command			
Action Generator Command			
Config Generator Command			
Wait Generator Command			
Address Space Generator Command			
CRC			□

Packets•Pre-Trigger Number

Format: ○ Hexadecimal ● Decimal ○ Binary ○ ASCII

Bit Order: ○ MSB to LSB ○ LSB to MSB

Hiding: □ Hidden

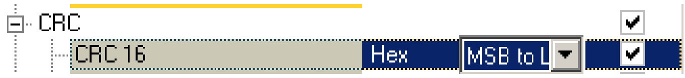
Color: Standard | Custom

Colors:

To change the format of alphanumeric characters in a field:

Step 1 Under the Group and Colors column, click the **plus** symbol (**+**) next to the group you want to reformat. The group expands to show the individual fields within the group (as shown above).

Step 2 Click the **row** representing the field that you want to reformat. If the field can be reformatted, the format options at the top of the dialog box become active, as shown below:



CRC			☑
CRC 16	Hex	MSB to L	☑

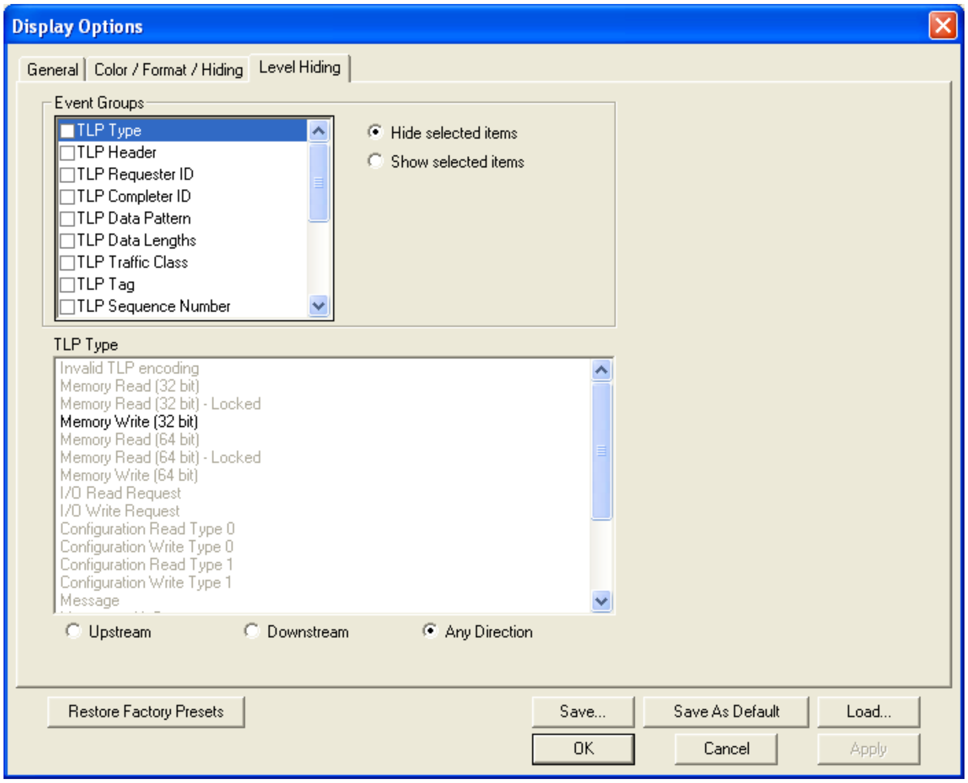
Step 3 Select a **format**.

Step 4 Specify the bit order in the displayed fields by checking/unchecking the **MSB > LSB** checkboxes.

Step 5 Click **Apply** or **OK**.

Hiding Fields

The Level Hiding page lets you hide traffic by Event Group. If you select one or more event types from the Event Group list, the selected types are hidden from the trace.



Event Groups

- □ TLP Type
- □ TLP Header
- □ TLP Requester ID
- □ TLP Completer ID
- □ TLP Data Pattern
- □ TLP Data Lengths
- □ TLP Traffic Class
- □ TLP Tag
- □ TLP Sequence Number

● Hide selected items
○ Show selected items

TLP Type

Invalid TLP encoding
Memory Read (32 bit)
Memory Read (32 bit) - Locked
Memory Write (32 bit)
Memory Read (64 bit)
Memory Read (64 bit) - Locked
Memory Write (64 bit)
I/O Read Request
I/O Write Request
Configuration Read Type 0
Configuration Write Type 0
Configuration Read Type 1
Configuration Write Type 1
Message

○ Upstream ○ Downstream ● Any Direction

To hide a field:

Step 1 Under the Group and Colors column, click the **plus** symbol (**+**) next to the group that has the field(s) you want to hide. The group expands to show the individual fields within the group (as shown above).

Step 2 Click the **checkbox** in the row representing the field that you want to hide.

Step 3 Click **Apply** or **OK**.

The Event Groups for Packets are:

- TLP Type
- TLP Header
- TLP Requester ID
- TLP Completer ID
- TLP Data Pattern
- TLP Data Lengths
- TLP Traffic Class
- TLP Tag
- TLP Sequence Number
- DLLP Type
- DLLP Header
- DLLP Virtual Channel
- ACK/NAK Seq Number
- Ordered Sets
- Link Event
- Direction
- Errors

The Event Groups for Link Transactions and Split Transactions are:

- TLP Type
- Traffic Class
- Virtual Channel
- Direction
- RequesterID
- CompleterID
- Ordered Sets
- Staus
- Tag

7.4 Load a Previously Saved Display Options File

If you have previously saved Display Options, you can load them by opening the Display Options dialog and clicking the **Load** button. A dialog box opens to let you load a previously saved display options file.

Step 1 Click **Load** to use a previously defined display options file.

Step 2 When you see the Open File pop-up window, enter the name of the file you want to load and click **Open**.

Step 3 When the PE*Tracer*™ software returns you to the Recording Options menu, click **OK** to activate the display options you selected.

7.5 Saving Display Options

If you have customized the Display Options and wish to save them, you can do so by clicking the **Save** button, then entering a unique file name. The **.opt** extension is added by default.

Setting the Defaults: Save the currently specified Display Options to the file name: **default.opt** by clicking **Save As Default.** When the Analyzer software begins execution, it automatically loads the **default.opt** file, if one exists.

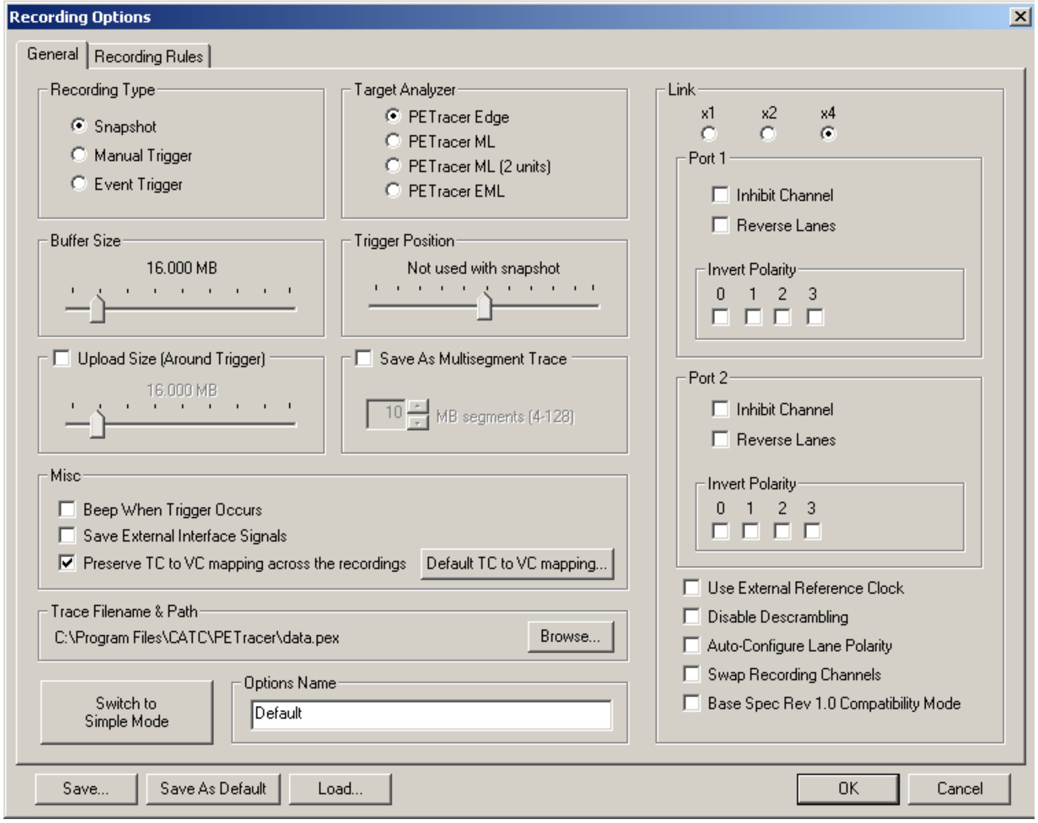
Chapter 8: Recording Options

8.1 Setting Recording Options

The Recording Options dialog is used to configure a recording.

To open the Recording Options window, click or select the command **Setup > Recording Options**.

PE*Tracer* Edge in Advanced Mode:



Recording Options

General | Recording Rules

Recording Type
- ● Snapshot
- ○ Manual Trigger
- ○ Event Trigger

Target Analyzer
- ● PETracer Edge
- ○ PETracer ML
- ○ PETracer ML (2 units)
- ○ PETracer EML

Link
- ○ x1 ○ x2 ● x4

Port 1
- □ Inhibit Channel
- □ Reverse Lanes

Invert Polarity
- 0 □ 1 □ 2 □ 3 □

Buffer Size: 16.000 MB

Trigger Position: Not used with snapshot

□ Upload Size (Around Trigger): 16.000 MB

□ Save As Multisegment Trace: 10 MB segments (4-128)

Port 2
- □ Inhibit Channel
- □ Reverse Lanes

Invert Polarity
- 0 □ 1 □ 2 □ 3 □

Misc
- □ Beep When Trigger Occurs
- □ Save External Interface Signals
- ☑ Preserve TC to VC mapping across the recordings

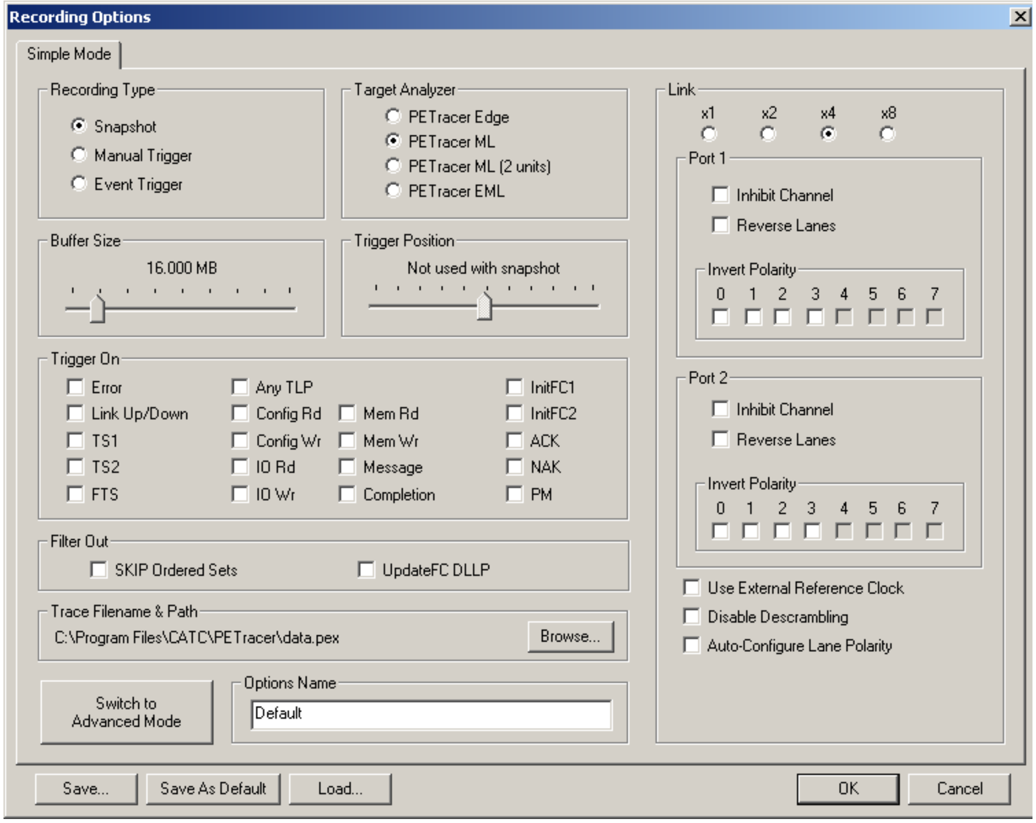
Trace Filename & Path: C:\Program Files\CATC\PETracer\data.pex

Options Name: Default

- □ Use External Reference Clock
- □ Disable Descrambling
- □ Auto-Configure Lane Polarity
- □ Swap Recording Channels
- □ Base Spec Rev 1.0 Compatibility Mode

Note: There are separate sets of Recording Options for each Analyzer type. To set the Analyzer type, select the appropriate platform from the Target Analyzer menu in the General page of the Recording Options.

PE*Tracer* ML in Simple Mode:



Recording Options

Simple Mode

Recording Type
- ● Snapshot
- ○ Manual Trigger
- ○ Event Trigger

Target Analyzer
- ○ PETracer Edge
- ● PETracer ML
- ○ PETracer ML (2 units)
- ○ PETracer EML

Buffer Size: 16.000 MB

Trigger Position: Not used with snapshot

Trigger On

□ Error	□ Any TLP		□ InitFC1
□ Link Up/Down	□ Config Rd	□ Mem Rd	□ InitFC2
□ TS1	□ Config Wr	□ Mem Wr	□ ACK
□ TS2	□ IO Rd	□ Message	□ NAK
□ FTS	□ IO Wr	□ Completion	□ PM

Filter Out
- □ SKIP Ordered Sets
- □ UpdateFC DLLP

Trace Filename & Path: C:\Program Files\CATC\PETracer\data.pex Browse...

Switch to Advanced Mode

Options Name: Default

Link
- x1 ○ x2 ○ x4 ● x8 ○

Port 1
- □ Inhibit Channel
- □ Reverse Lanes

Invert Polarity

0	1	2	3	4	5	6	7
□	□	□	□	□	□	□	□

Port 2
- □ Inhibit Channel
- □ Reverse Lanes

Invert Polarity

0	1	2	3	4	5	6	7
□	□	□	□	□	□	□	□

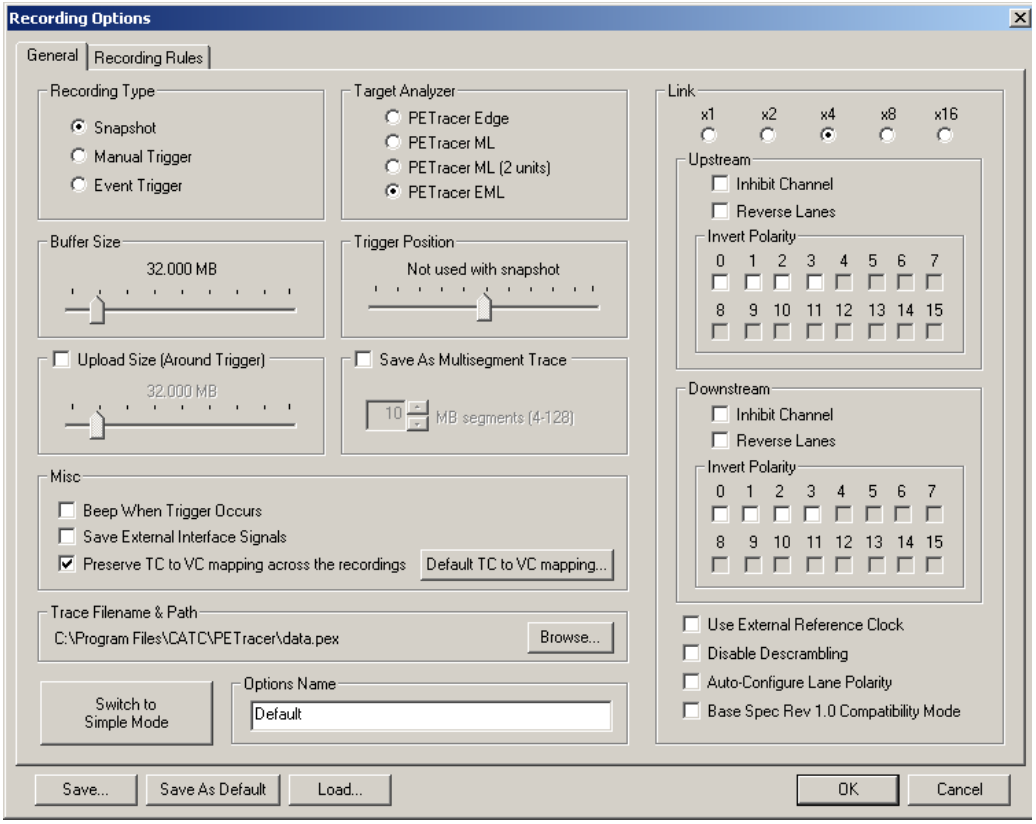
- □ Use External Reference Clock
- □ Disable Descrambling
- □ Auto-Configure Lane Polarity

Save... Save As Default Load... OK Cancel

8.2 General Tab

The General Page presents options that are affect all recordings:

PE*Tracer* EML in Advanced Mode:



Recording Options

General | Recording Rules

Recording Type
● Snapshot
○ Manual Trigger
○ Event Trigger

Target Analyzer
○ PETracer Edge
○ PETracer ML
○ PETracer ML (2 units)
● PETracer EML

Link
x1 ○ x2 ○ x4 ● x8 ○ x16 ○

Upstream
□ Inhibit Channel
□ Reverse Lanes
Invert Polarity
0 □ 1 □ 2 □ 3 □ 4 □ 5 □ 6 □ 7 □
8 □ 9 □ 10 □ 11 □ 12 □ 13 □ 14 □ 15 □

Buffer Size
32.000 MB

Trigger Position
Not used with snapshot

□ Upload Size (Around Trigger)
32.000 MB

□ Save As Multisegment Trace
10 MB segments (4-128)

Downstream
□ Inhibit Channel
□ Reverse Lanes
Invert Polarity
0 □ 1 □ 2 □ 3 □ 4 □ 5 □ 6 □ 7 □
8 □ 9 □ 10 □ 11 □ 12 □ 13 □ 14 □ 15 □

Misc
□ Beep When Trigger Occurs
□ Save External Interface Signals
☑ Preserve TC to VC mapping across the recordings
Default TC to VC mapping...

Trace Filename & Path
C:\Program Files\CATC\PETracer\data.pex
Browse...

□ Use External Reference Clock
□ Disable Descrambling
□ Auto-Configure Lane Polarity
□ Base Spec Rev 1.0 Compatibility Mode

Switch to Simple Mode

Options Name
Default

Save... Save As Default Load... OK Cancel

Recording Type: Sets the trigger mechanism for the recording.

Buffer Size: Causes the Analyzer to record traffic to its buffer and then upload the trace to the host PC. Recordings are limited in size to the size of the Analyzer's buffer (4 GB).

Target Analyzer: Presents a menu with options for selecting an Analyzer platform:

- **PE*Tracer* Edge**
- **PE*Tracer* ML**
- **PE*Tracer* ML** (2 Units)
- **PE*Tracer* EML**

Trigger Position: Controls the percentage of buffer allocated for pre- and post-buffer recording.

Trigger On: Error, Link Up/Link Down. TS1, TS2, FTS, Any TLP, Config Rd, Config Wr, IO Rd, IO Wr, Mem Rd, Mem Wr, Message, Completion, InitFC1, InitFC2, ACK, NAK, and/or PM.

Filter Out: SKIP Ordered Sets or UpdateFC DLLP

Trace Filename & Path: Sets the path and trace name for the recording.

Options Name: Sets a descriptive label for the Recording Options so you can more easily recall what settings are in the Recording Options file.

Link: Settings: For link width, inhibiting recording, polarity, external reference clock, swapping recording channels, inverting link polarity, and descrambling.

Upload Size (Advanced Mode): Causes the Analyzer to upload a portion of the Analyzer's buffer. This option lets you look at part of the trace. Half of the uploaded trace is pre-buffer and half post-trigger.

Save As MultiSegment Trace (Advanced Mode): Divides the recording into segments so as to make very large recordings more easily uploaded and viewed. If you are planning to create a large recording, you might want to test this option to see how it affects performance. Large trace files are easier to navigate but slower to open.

Misc (Advanced Mode): Turns on trigger beep, sets external clocking (EML only), allows external interface signals to be saved into the trace, and tells the Analyzer to use whatever TC to VC mapping was used in the last recording (to re-use previously discovered Configuration Space data) in all future recordings. Also presents a button for manually mapping the Traffic Classes to Virtual Channels.

Recording Type

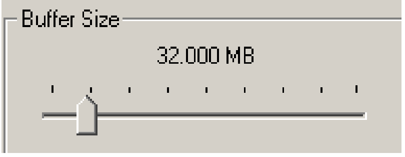
Recording Type lets you to specify the type of recording you want to make:

- **Snapshot**: A recording of a pre-determined length. You set the recording length in the Buffer Size box. Recording begins when you click the **Rec** button on the toolbar and ends when the selected buffer size is filled or when you press the **Stop** button.
- **Manual Trigger**: A recording that ends when you push the **Trigger** button on the front of the Analyzer. Recording begins when you select **Start** in the application. Pressing the **Trigger** button causes the Analyzer to begin to finish recording. Recording continues until the post-trigger buffer has been filled. You can also end the recording by pressing the **Stop** button in the application.
- **Event Trigger**: A recording whose end is triggered by an event in the trace. An Event Trigger begins when you select **Start** in the application and ends when the specified triggering event occurs in the trace or you press **Stop**. If an event triggers the end of the recording, the Analyzer records a predefined amount of post-trigger data (specified by Trigger Position and Buffer Size.)

  Note: You can also terminate an Event Trigger recording by pressing the Manual Trigger button on the front of the Analyzer. When the Manual Trigger button is pressed, the Analyzer continues to record until the specified post-trigger buffer has been filled.

Buffer Size

The Buffer Size slide-bar allows you to set the size of the recording buffer.



For PE*Tracer* ML, the size of this buffer is adjustable from 1.6 MB to 2 GB.

After you have set the Buffer Size, you must set the Recording type and Trigger position options. These options determine how the buffer is used.

Note: The Buffer Size slide-bar does not precisely portray the buffer size because of the way the packets are stored in the Analyzer's memory.

Target Analyzer

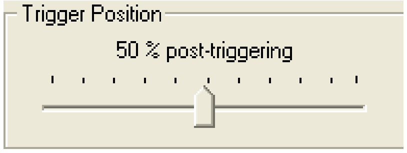
Target Analyzer presents a menu with these choices:

- **PE*Tracer* Edge**: Displays the options for the PE*Tracer* Edge Analyzer.
- **PE*Tracer* ML**: Displays the options for the PE*Tracer* ML Analyzer.
- **PE*Tracer* ML** (2 Units): Displays the same options as PE*Tracer* ML but lists **Unit 1** and **Unit 2** in the Link Section on the right. In a two-unit setup, **Unit 1** and **Unit 2** are two separate Analyzers linked together by BNC on the back of the units to form a single, logical Analyzer. The cabling for a two-unit setup is described in "Connecting the PETracer ML Analyzer to the Device Under Test" on page 51.
- **PE*Tracer* EML**: Displays the options for the PE*Tracer* EML Analyzer.

Selecting an Analyzer platform changes the options presented in the Link and Events pages within the Recording Options dialog.

Trigger Position

This Trigger Position slide-bar lets you to adjust the amount of recording buffer allocated to recording pre-trigger and post-trigger traffic.



For example, if you set the Trigger Position to 90% Post-Triggering Traffic, the Analyzer records 10% pre-trigger traffic and 90% post-trigger traffic.

Trigger Position is only available when Manual Trigger or Event Trigger is selected.

Trace Filename and Path

The Trace Filename and Path button on the Recording Options General panel allows you to change the default file name and path for the recorded trace file. The pre-defined name is **data.pem**.

Step 1 Select the **Recording File Name** button.

Step 2 When you see the **Save As** menu, navigate to the directory you want.

Step 3 Enter the new file name in the File name field.

Step 4 Click the **Save** button.

This action does not do any immediate save operation. It just changes the default name and uses it in subsequent recordings.

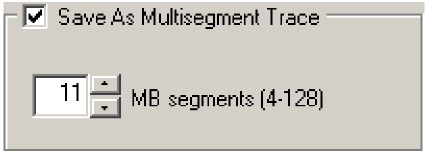
Options Name

The Options Name is a descriptive, supplemental label that you can assign to a Recording Options file.

For example, if your Recording Options file were named **StandardSettings.rec**, your Options Name could be a long descriptive label such as **Standard Record Options used for all normal Recordings**.

Save As MultiSegment Trace

This option causes the Analyzer to segment the trace into 4 MB to 128 MB files and to create an index file that summarizes the starting and finishing frame for each segment.



☑ Save As Multisegment Trace

11 MB segments (4-128)

The index file has a default name of **data.pem**.

The index file and segmented trace files are stored in a directory named **data_pem_files**. The directory is named after the index file. Below this directory additional, sequentially numbered sub-directories (up to 10,000) hold the segmented trace files. These sub-directories bear simple numerical names: 00000 to 00999. Each of these subdirectories can hold up to 100 sequentially numbered segment files.

Example
A 1010 MB recording using the default file names creates the following sub-directories and files:

`data.pem` (This is the index file.)

```
data_pem_files\00000\segment_00000.pex

data_pem_files\00000\segment_00001.pex

...

data_pem_files\00000\segment_00099.pex

data_pem_files\00001\segment_00100.pex

data_pem_files\00001\segment_00101.pex
```

The index file looks something like a trace file but contains packet-like entities that summarize each segment.

Viewing Multisegmented Files

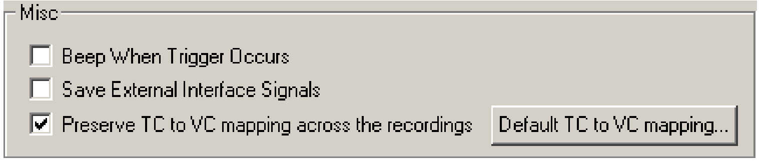
If Save As MultiSegment Trace is enabled in the General page of the Recording Options, PE*Tracer* divides the trace into segments and stores them on the host's hard drive. Segment size is set in the MB Segments box. Enter a value from 4 MB to 128 MB.

PE*Tracer* also creates an index file that provides a brief summary of each segment and hyperlinks to each of the segments. Double-clicking a segment summary in the index file causes PE*Tracer* to open that segment.

You can also navigate through the segments by clicking the buttons on the Multisegment toolbar.

Misc

Allows you to specify the following parameters for recording and uploading traffic:



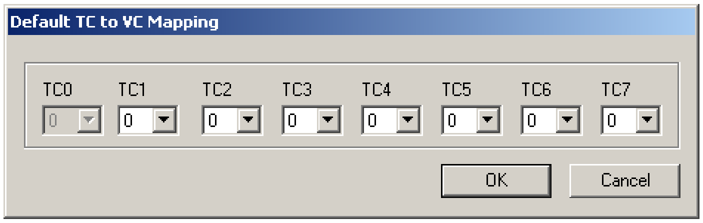
Misc

☐ Beep When Trigger Occurs

☐ Save External Interface Signals

☑ Preserve TC to VC mapping across the recordings

- **Beep When Trigger Occurs**: Causes the Analyzer to beep when a trigger event is detected.
- **Save External Interface Signals**: If selected, causes the Analyzer to save signals from a Breakout Board as fields in the trace.
- **Preserve TC to VC mapping across the channels**: Causes the Analyzer to use whatever TC to VC mapping it established in the last recording.
- **Default TC to VC mapping button**: Lets users manually configure the default mapping.



Default TC to VC Mapping

TC0	TC1	TC2	TC3	TC4	TC5	TC6	TC7
0	0	0	0	0	0	0	0

Link

The Link section allows you to manage links:

Link Width: Sets the physical width of the link.

Port 1 (Unit1) and Port 2 (Unit 2) (PE*Tracer* Edge and PE*Tracer* ML): Gives you low-level control over each link direction. The heading for these boxes indicates the port and/or unit to which the following options are applied based on the current Analyzer configuration:

- **Inhibit Channel**: Prevents recording on the selected link direction.
- **Reverse Lanes**: Causes the Analyzer to reverse the lane order for this link direction. For example, if you are working with a PCI Express™ device that uses a 3 2 1 0 lane order instead of a 0 1 2 3 order or uses a 0 to 15 lane order instead of a 15 to 0 order,, you select this option to enable the Analyzer to match the order of the device. **Note:** If reversing lanes for a PE*Tracer* ML x8 link, the Port 1 and Port 2 cable connections for the link must also be swapped.
- **Invert Polarity**: Inverts the d+ and d- polarity for each specified lane. If for some reason the d+ on one side of a link is wired to the d- on the other side (or vice versa), then selecting Invert Link Polarity corrects for that condition.

Upstream and Downstream (PE*Tracer* EML): Gives you low-level control over each link direction. The heading for these boxes indicates the port and/or unit to which the following options are applied based on the current Analyzer configuration:

- **Inhibit Channel**: See above.
- **Reverse Lanes**: See above.
- **Invert Polarity**: See above.

Use External Reference Clock: If the PCI Express link under analysis uses spread-spectrum clocking, then the Analyzer must use the external reference clock from the host system. If host does not supply a reference clock, the internal reference clock in the Analyzer module is used instead for link analysis.

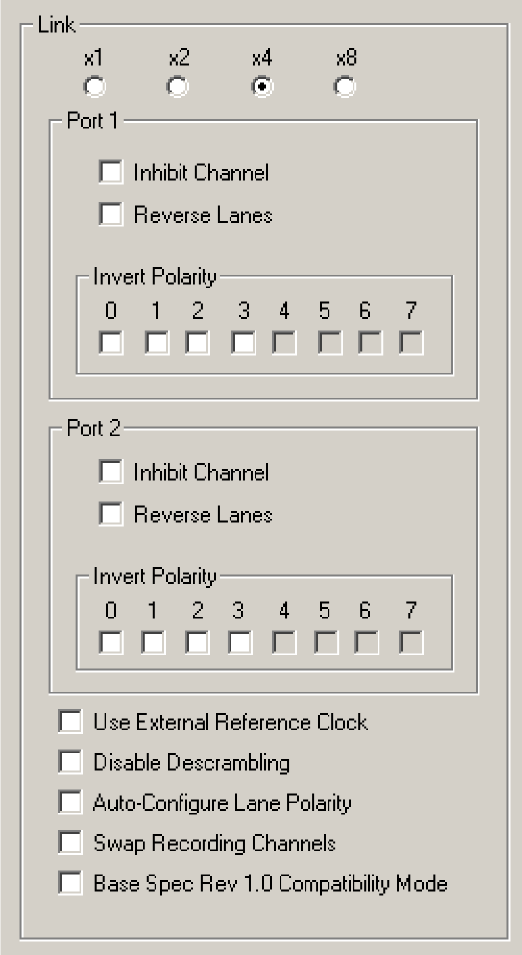
Disable Descrambling: If checked, causes the Analyzer to assume that none of the PCI Express traffic is scrambled. By default, the Analyzer determines the scrambling state of the devices under test.

Auto-Configure Lane Polarity: Lets the Analyzer determine lane polarity.

Base Spec 1.0 Rev Compatibility Mode (Advanced Mode): This option causes the Analyzer to conform to the PCI Express 1.0 Specification.

Swap Recording Channels (Advanced Mode) (PE*Tracer* ML and PE*Tracer* Edge): Swaps upstream/downstream assignments. Normally, the software makes upstream and downstream assignments based on Configuration Requests within the trace. Sometimes these requests are not captured. You can manually re-assign channels using this checkbox.

PE*Tracer* ML (PE*Tracer* Edge looks the same but does not have an x8 Link Width):



Link

x1 ○ x2 ○ x4 ● x8 ○

Port 1

□ Inhibit Channel

□ Reverse Lanes

Invert Polarity

0 □ 1 □ 2 □ 3 □ 4 □ 5 □ 6 □ 7 □

Port 2

□ Inhibit Channel

□ Reverse Lanes

Invert Polarity

0 □ 1 □ 2 □ 3 □ 4 □ 5 □ 6 □ 7 □

□ Use External Reference Clock

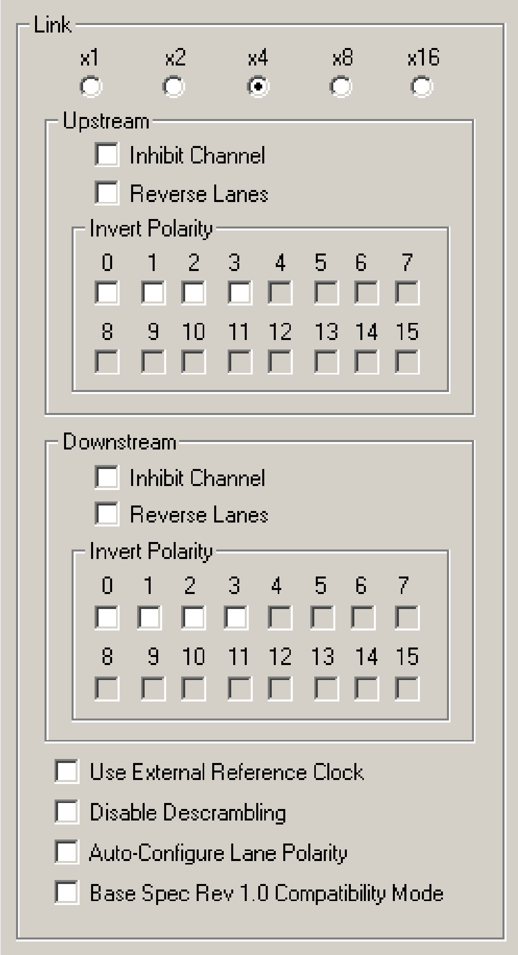
□ Disable Descrambling

□ Auto-Configure Lane Polarity

□ Swap Recording Channels

□ Base Spec Rev 1.0 Compatibility Mode

PE*Tracer* EML:



Link

○ x1 ○ x2 ● x4 ○ x8 ○ x16

Upstream

□ Inhibit Channel

□ Reverse Lanes

Invert Polarity

0 □ 1 □ 2 □ 3 □ 4 □ 5 □ 6 □ 7 □

8 □ 9 □ 10 □ 11 □ 12 □ 13 □ 14 □ 15 □

Downstream

□ Inhibit Channel

□ Reverse Lanes

Invert Polarity

0 □ 1 □ 2 □ 3 □ 4 □ 5 □ 6 □ 7 □

8 □ 9 □ 10 □ 11 □ 12 □ 13 □ 14 □ 15 □

□ Use External Reference Clock

□ Disable Descrambling

□ Auto-Configure Lane Polarity

□ Base Spec Rev 1.0 Compatibility Mode

Saving and Loading Previously Saved Recording Options

The options are:

Save: Saves the current options to whatever file name you provide.

Save As Default: Saves the current options into the default options file. This file is called **default.rec** or whatever other name you have assigned to the default options file. Whenever the LeCroy PE*Tracer* software begins execution, it automatically loads the default file, if one exists.

Load: Loads a previously saved set of recording options.

OK: Applies changes and closes the Recording Options dialog box.

Cancel: Cancels changes and closes the Recording Options dialog box.

Loading Recording Options

In the Recording Options menu, you can load a previously saved recording options file.

To load Recording Options:

Step 1 Select **Setup > Recording Options** from the menu.

Step 2 Click the **Load** button from the Recording Options dialog box. The Load dialog opens and lists previously saved options files (***.rec**).

Step 3 Select a file and click **OK**. The options file loads.

Saving Recording Options

Recording Options settings can be saved and later reused. Recording options settings are stored in ***.rec** files.

Step 1 Open the Recording Options dialog by selecting **Setup > Recording Options.**

Step 2 Set your options, then click **Save**.

Step 3 Enter a unique file name. The **.rec** extension is added by default.

Step 4 (optional) To add a descriptive label to this file to help you remember what options were set, use the **Options Name** box.

Setting Default Recording Options

To save the current recording options into the default Recording Options file:

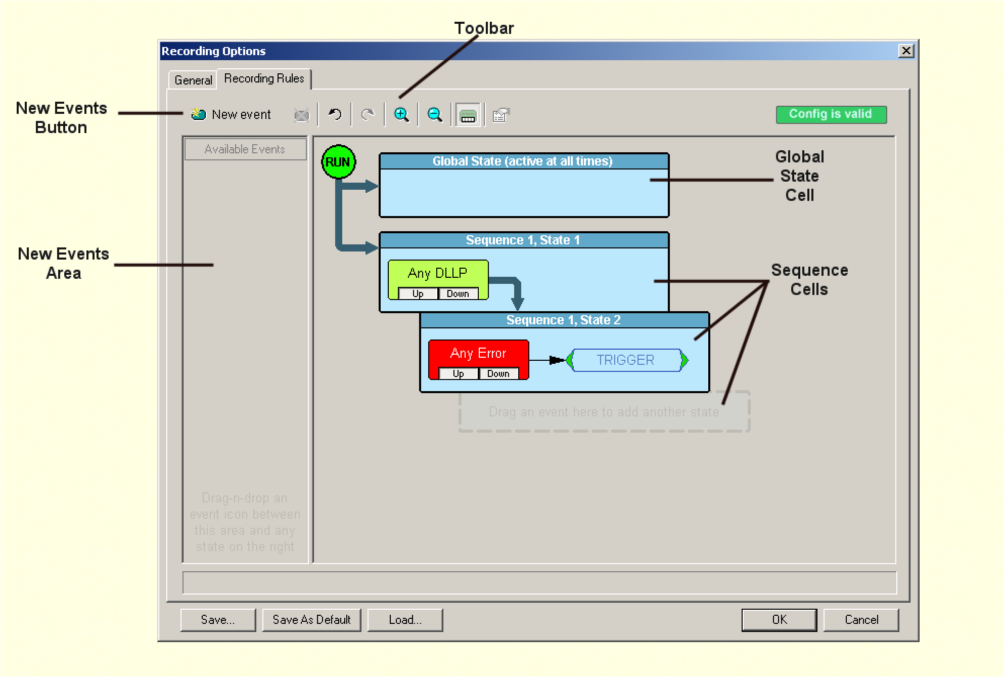
- Click **Save As Default**.

  The default file for the options is **default.rec**.

  When the PE*Tracer* software begins execution, it automatically loads the **default.rec** file, if one exists.

8.3 Recording Rules Overview

The Recording Rules page lets you to set triggers and filters.



The page divides into three areas:

Toolbar: Contains buttons such as the New Events button for issuing commands.

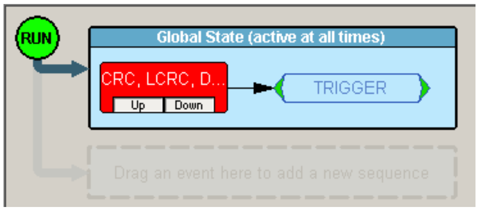
Available Events area: A part of the screen where you can park buttons that you intend to use in the Main display area.

Main display area: The part of the screen where you create trigger and filter conditions. You create conditions by dragging buttons onto the Main display area from the Available Events area. You then create additional conditions by right-clicking a button and selecting options from a pop-up menu. See Creating Recording Rules.

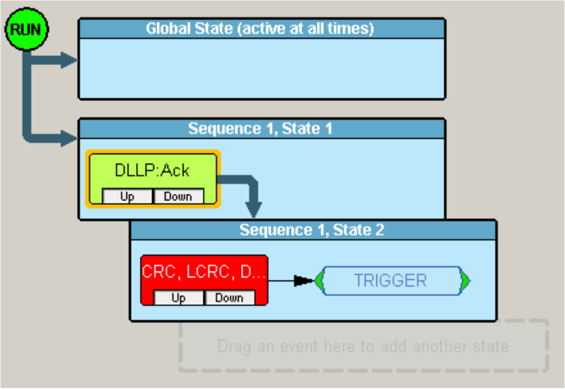
Global State vs. Sequences

The Main Display area in the center of the Recording Rules page has two cells that affect events differently.

Global State: Events dragged into the Global State cell are searched for throughout the recording. For example, if you place an Error in the Global State cell and assign a Trigger to it, the Analyzer searches for errors from the start of the recording until the end.

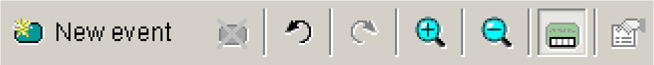

RUN
Global State (active at all times)
CRC, LCRC, D...
Up
Down
TRIGGER
Drag an event here to add a new sequence

Sequence State: The cell marked **Drag an event here to create a new sequence** is a Sequence Cell. Sequence cells are used to event sequences, which are chains of events culminating in a trigger or other action. One sequence (i.e., a separate chains of events) can be created with up to 32 states. A state is an event condition plus some action within a sequence.



8.4 Recording Rules Buttons

The Recording Rules toolbar allows you to create and edit recording rules:



New event	New Event. Opens a drop-down menu with a list of events.	⊕	Zoom in
☒	Delete. Delete selected event.	⊖	Zoom out
↶	Undo. Undoes last action.	▤	Show/Hide Channels. Shows/hides the channel buttons.
↷	Redo. Undoes last Undo command.	▦	Show properties. Opens the Properties dialog box for the selected item.

8.5 Creating Recording Rules

The Recording Rules page is used to set triggers and filters. To access this page, select **Setup > Recording Options > Recording Rules**.

There are three steps to creating a recording rule:

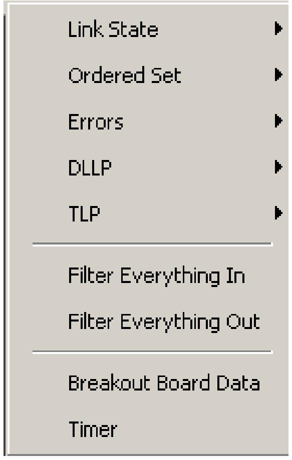
1) Select events.

2) Place the events in the Global State or Sequence cells.

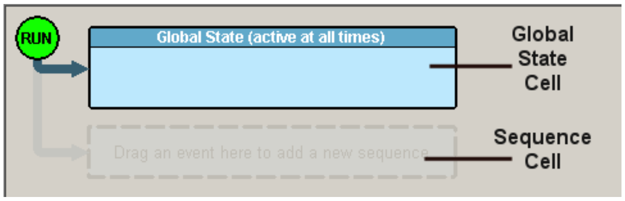
3) Assign actions to the events.

Note: There are limits to the types of rules that can be created. See “Recording Rule Limits” on page 146 for details.

Step 1 Click New event and select one or more events from the menu. Selecting an event automatically places it in the Available Events area. This area serves as a parking lot where you can place event buttons without them having any effect on the Analyzer.



Step 2 Drag the selected events from the Available Events area into one of either the Global State cell or the Sequence cell (see “Global State vs. Sequences” on page 142):

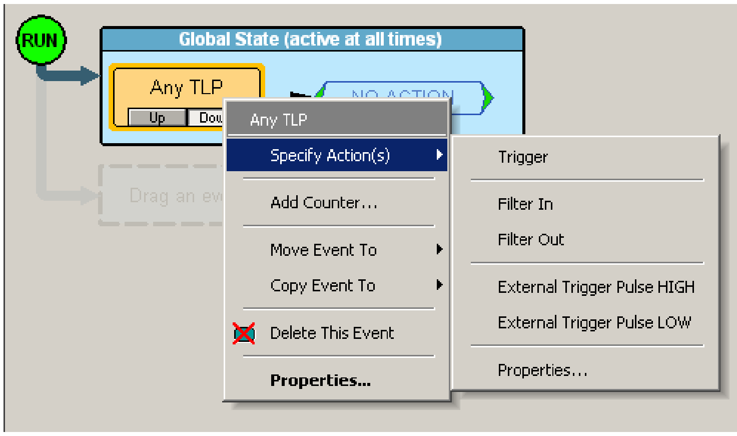

RUN
Global State (active at all times)
Drag an event here to add a new sequence
Global State Cell
Sequence Cell

Global State Cell: If you want the Analyzer to always search for the event, place it in the cell marked **Global State**.

Sequence Cell: To create an event sequence, place two or more events in the faintly marked cell that reads **Drag an event here to create a sequence**. At this point, the selected events have no effect because an action has not been assigned.

Step 3 To select a sub-set of your selected event, right-click it and choose **Properties**. A Properties dialog box opens that presents additional options. For example, if you open the Properties dialog for Errors, you can set the specific types of errors the Analyzer should look for.

Step 4 Assign an action to the selected events by right-clicking each of the events, selecting **Specify Action** from the pop-up menu, and assigning an action such as **Trigger, Filter**, or **Count**. Be sure to click the event itself and not the **State** cell that it is sitting in (which produces a different pop-up menu.)



Note: You can also assign actions to events by double-clicking the event and selecting the Actions page when the Properties dialog box opens.

Step 5 Click **OK** to close the dialog box. At this point, assuming that the other options in the Recording Options dialog box have been set (such as the General page), you can begin the recording by pressing the

Start **Recording** ● button.

8.6 Recording Rule Limits

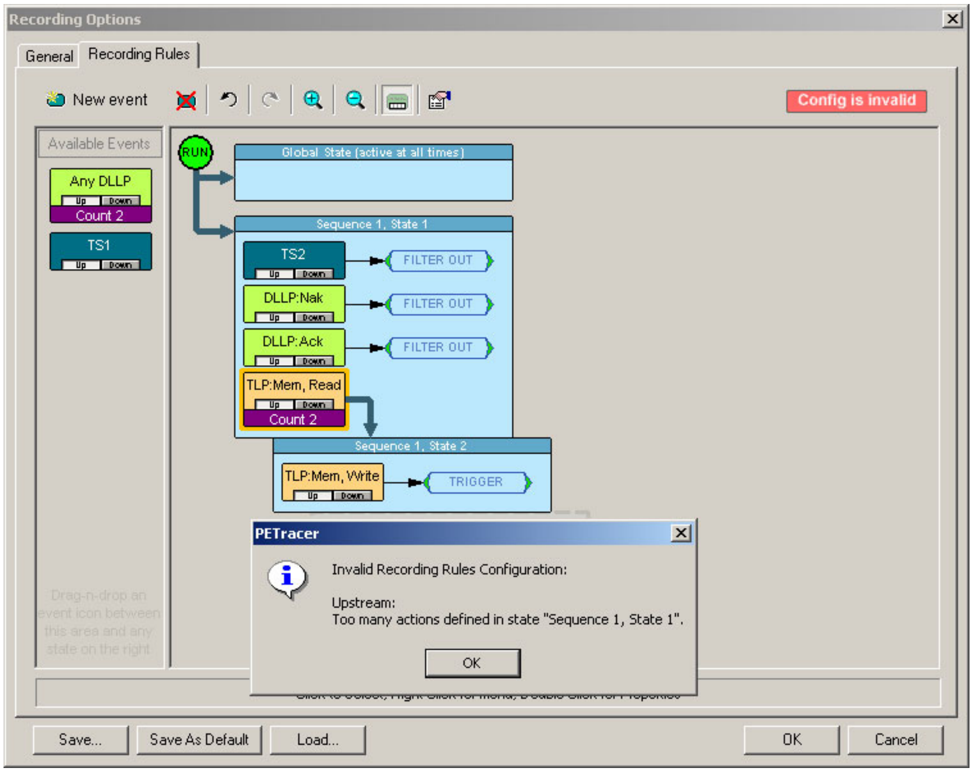
PE*Tracer* ML and PE*Tracer* EML have finite buffer resources that limit the number of actions and events that can be assigned for each recording channel. When creating complex event sequences, it is possible to hit the limits of these resources and get an error message like the ones shown in the screenshots below.

In addition to the following limitations, you should also be aware of how PE*Tracer* handles user-created contradictory rules. See “Recording Rules Logic: How Contradictory Rules are Resolved” on page 148 for an explanation.

In order to maximize the buffer allocation for recordings, LeCroy has imposed the following configuration limitations into PE*Tracer* ML and PE*Tracer* EML.

Configuration Limitations in PE*Tracer* ML

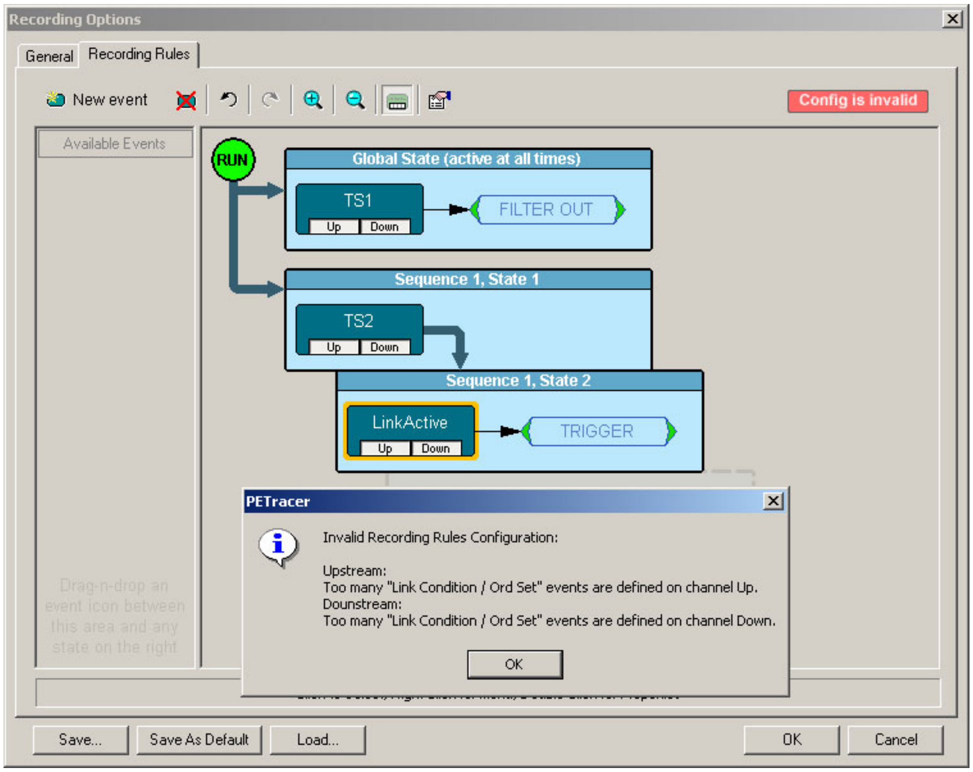
- **Up to four actions per state**: No more than four actions can be assigned per state or the error message shown below appears and the Status Indicator in the top right of the dialog generates an **Invalid Configuration** status.
  Example: Too many Actions per State. In this example, the actions are: three Filter Outs, an Advance the Sequence, and a Count.



Invalid Recording Rules Configuration:

Upstream:
Too many actions defined in state "Sequence 1, State 1".

- **Up to two sets of Link Conditions or Ordered Sets per channel**: If you place a combination of three Link Conditions and Ordered Sets into three separate state cells and then assign three different actions to each, an error results.
  Example: Place TS1 in the Global State cell and select **Filter Out**. Then place TS2 in **Sequence 1, State 1** and select **Advance the Sequence**. Then add Link Active to **Sequence 1, State 2** and select **Trigger**. When you attempt to apply **Trigger** to **Link Active**, an error message appears.
  Example: Too many Link Conditions and Ordered Sets. In this example, the user attempted to create a rule with three Link Conditions and Ordered Sets. The maximum allowable Link Conditions and Ordered Sets is two.



Invalid Recording Rules Configuration:

Upstream:
Too many "Link Condition / Ord Set" events are defined on channel Up.
Dounstream:
Too many "Link Condition / Ord Set" events are defined on channel Down.

- **In two-unit setups, no cross-box sequencing is allowed**: For example, you cannot configure the first Analyzer to advance sequence on TLP and then configure the second to trigger on a DLLP-ACK for this TLP. Actions apply to both Analyzers.
- **Four different DLLPs per channel**
- **Three different TLPs per channel**
- **Four payload DWORD-checkers per channel**
- **One set of errors per channel**
- **One Timer event per channel**
- **Two Counters per channel**

Configuration Limitations in PE*Tracer* EML

- **Four actions per state per channel**: See example above.
- **Two Link Condition/ Ordered Set resources per channel**: See example above.
- **Up to four different DLLPs per channel**
- **Up to three different TLPs per channel**
- **One error set per channel**
- **Four payload DWORD-checkers per channel**
- **Two Timers per channel**
- **Two counters per channel**: No cross channel counting.

8.7 Recording Rules Logic: How Contradictory Rules are Resolved

When creating rules in the Recording Rules page, it is possible to create contradictory instructions such **Filter Anything Out** and **Filter Anything In**. To resolve such conflicts, the Recording Rules page implements three internal rules that are described here:

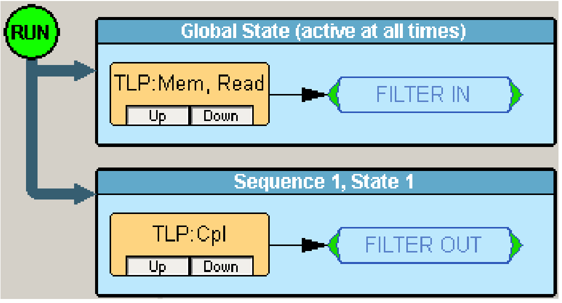
- Rules created in Sequence cells have a higher priority than rules created in the Global State cell: If a rule is placed in a Sequence cell, and a contradictory rule is placed in Global State cell, the rule in the Sequence cell applies.
- Filter-In has a higher priority than Filter-Out, so when a Filter-In rule is placed inside the same state as Filter-Out rules, only the Filter-In rule applies.
- Filter-In Anything and Filter-Out Anything have the highest priority and override any other filtering rules, so when Filter-In Anything or Filter-Out Anything are placed in the same state cell as other Filter rules, only the Filter-In Anything or Filter-Out Anything rules apply.

Recording Rules Examples

Read through the following examples to better understand how the three rules apply.

Note: In addition to these three rules, the Recording Rule Limitations define the **upper limits** of rule creation. These limitations should not affect you, but you should be aware of them.

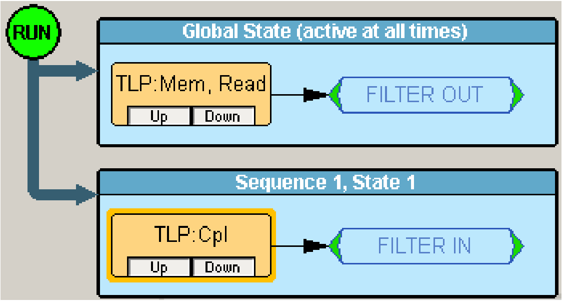
Example 1



Result: Only TLP:Mem,Read is in the trace.

Reason: The Sequence rule Filter out TLP:Completion is not contradicting the global state rule to filter in TLP:Memory. Both rules are applied.

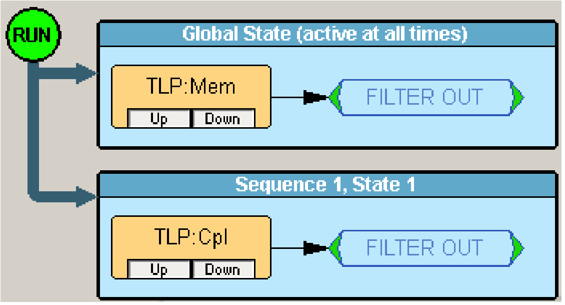
Example 2



Result: Only TLP:Completion is in the trace.

Reason: The sequence rule to filter in TLP:Completion is not contradicting the global state rule to filter out TLP:Memory. Both rules are applied.

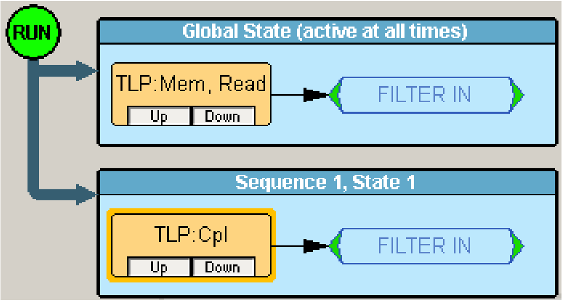
Example 3



Result: TLP:Completion and TLP:Memory are filtered out.

Reason: The Sequence rule to filter out TLP:Completion is not contradicting the global state rule to filter out TLP:Memory. Both rules are applied.

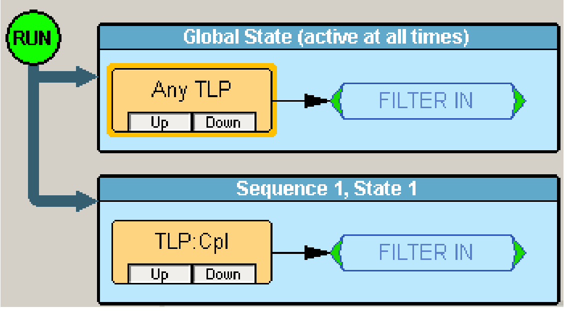
Example 4



Result: Only TLP:Completion and TLP:Memory are in the trace.

Reason: The sequence rule to filter in TLP:Completion is not contradicting the global state rule to filter in TLP:Memory. Both rules are applied.

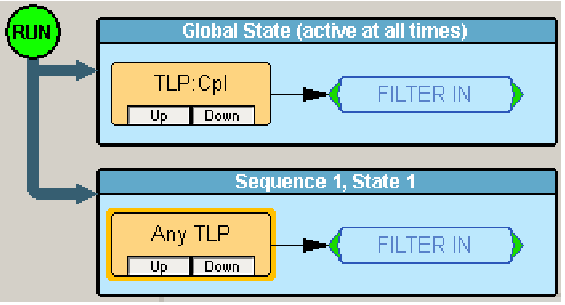
Example 5



Result: Only TLP (any type) is in the trace.

Reason: The sequence rule to filter in TLP:Completion is not contradicting the global state rule to filter in Any TLP. Both rules are applied.

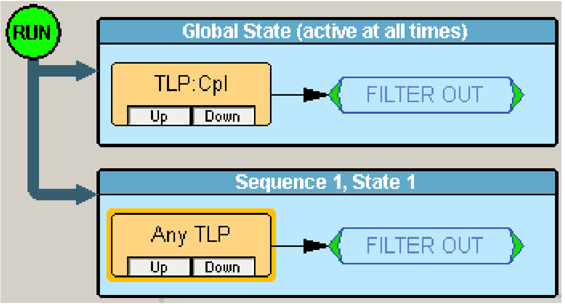
Example 6



Result: Only TLP (any type) is in the trace.

Reason: The sequence rule to filter in Any TLP is not contradicting the global state rule to filter in TLP:Completion. Both rules are applied.

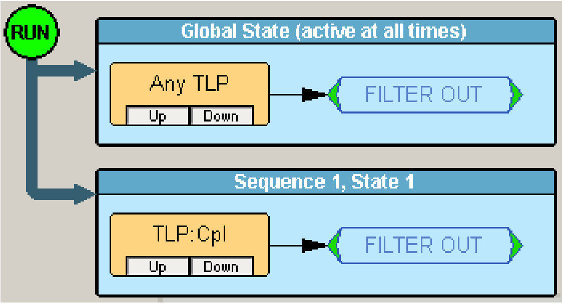
Example 7



Result: All TLP are filtered out.

Reason: The sequence rule to filter out Any TLP is not contradicting the global state rule to filter out TLP:Completion. Both rules are applied.

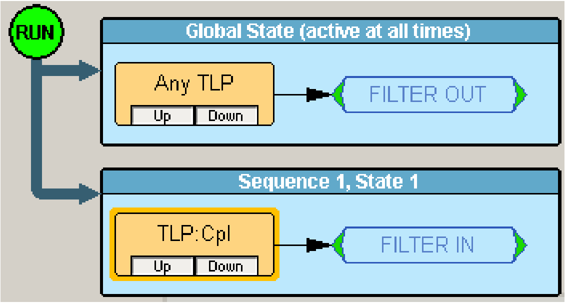
Example 8



Result: All TLP are filtered out.

Reason: The sequence rule to filter out TLP:Completion is not contradicting the global state rule to filter out Any TLP. Both rules are applied.

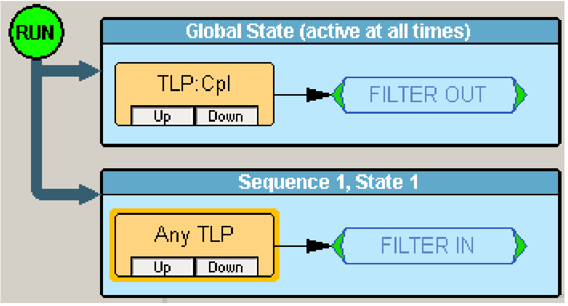
Example 9



Result: Only TLP:Completion is in the trace.

Reason: The sequence rule to filter in TLP:Completion is overwriting the global state rule to filter out Any TLP.

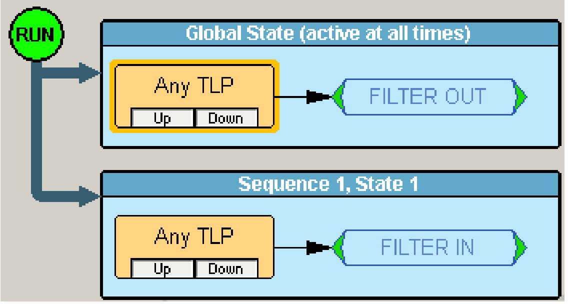
Example 10



Result: Only TLP (any type) is in the trace.

Reason: The sequence rule to filter in Any TLP is overwriting the global state rule to filter out TLP:Completion.

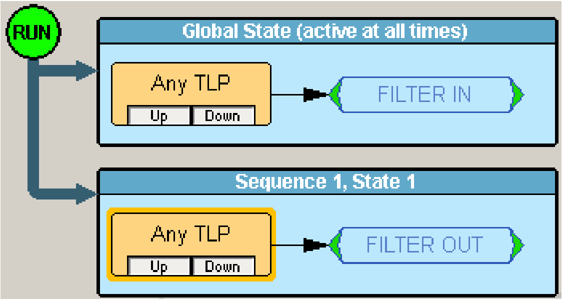
Example 11



Result: Only TLP (any type) is in the trace.

Reason: The sequence rule to filter in Any TLP is overwriting the global state rule to filter out Any TLP.

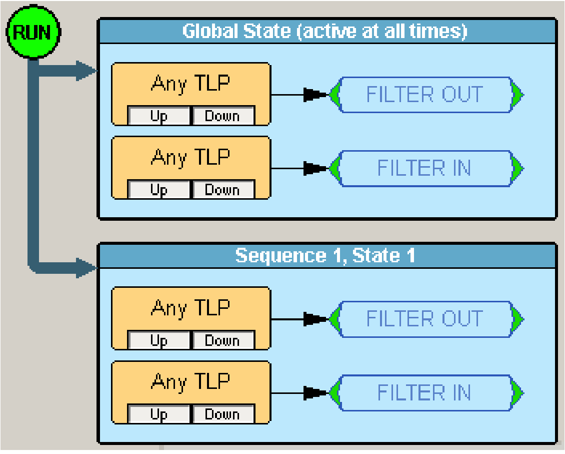
Example 12



Result: Empty trace.

Reason: The sequence rule to filter out Any TLP is not contradicting the global state rule to filter in Any TLP. Both rules are applied. (The global rule filters out DLLPs and Ordered Sets, and the sequence rule filters out TLPs.)

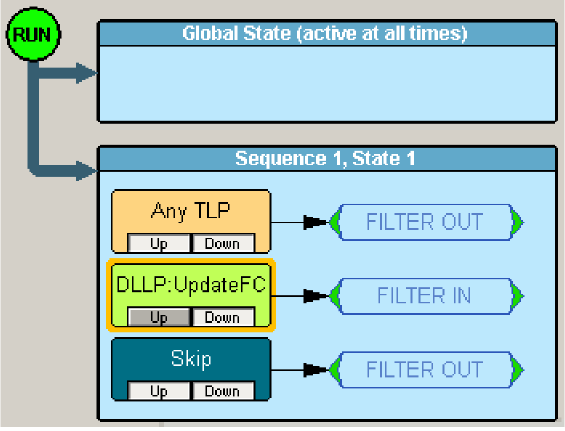
Example 13



Result: Only TLP (any type) is in the trace.

Reason: Filter out rules in the global state cell and in the sequence state cell are ignored. The Filter-In Any TLP rule is used.

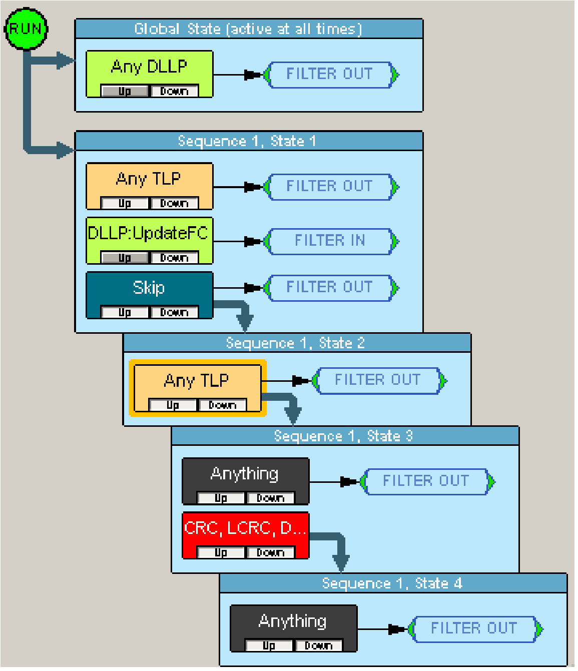
Example 14



Result: Only DLLP:UpdateFC is in the trace.

Reason: The Filter out rules are ignored. Filter in DLLP:updateFC rule is used.

Example 15



While in state 1: Only DLLP:UpdateFC is in the trace.

Reason: The Filter out rules are ignored. Filter in DLLP:updateFC rule is used.

While in state 2: Only Ordered Sets are in the trace.

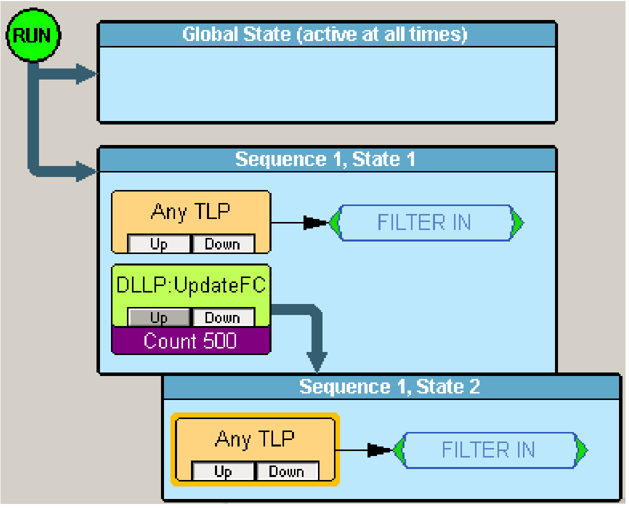
Reason: Both the sequence state and the global state rules are used, and all TLPs and DLLPs are filtered out.

While in state 3: Nothing is recorded.

While in state 4: Everything is recorded.

Reason: The sequence state rule to filter in anything overwrites global state rule.

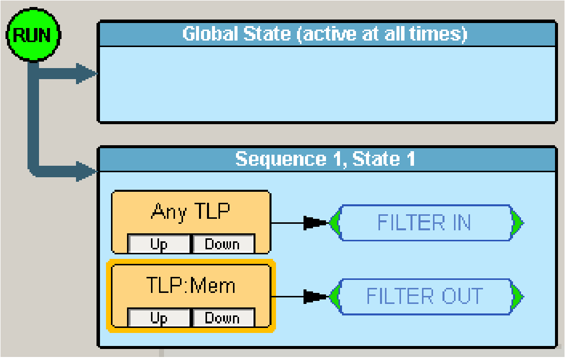
Example 16



Result: Only TLP (any type) is in the trace.

Reason: In both states, the rule is to keep only TLP.

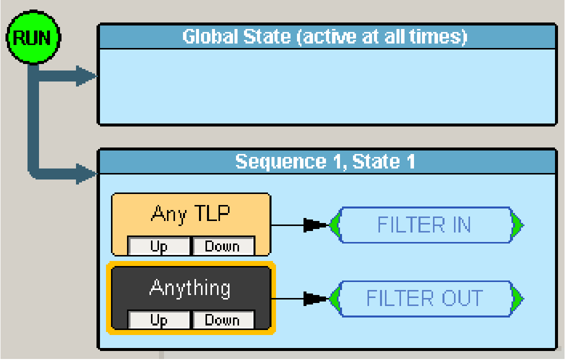
Example 17



Result: Only TLP (any type) is in the trace.

Reason: The Filter-out rule is ignored because there is a filter-in rule with a higher priority.

Example 18



Result: Empty trace.

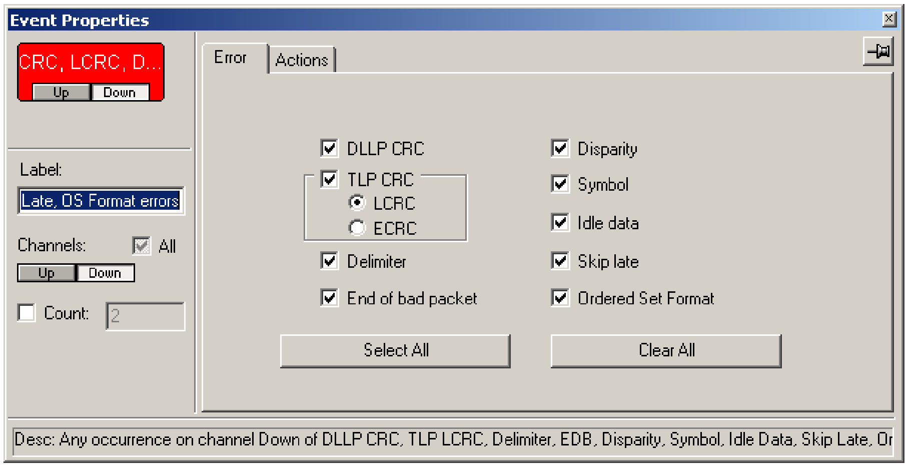
Reason: Filter Anything has the highest priority and overwrites filter-in rule.

8.8 Properties Dialog Boxes

Properties dialog boxes provide additional settings for Events, States, Actions, and other objects in the Recording Rules page. You can access a Properties dialog by double-clicking an **Event, State, Action** or other object.

8.9 Event Properties Dialog

The Event Properties dialog presents options for refining triggers and filters. For example, to set a trigger on a specific type of error, you open the Properties dialog



Label: Late, OS Format errors

Channels: ☑ All

☐ Count: 2

☑ DLLP CRC

☑ TLP CRC

● LCRC

○ ECRC

☑ Delimiter

☑ End of bad packet

☑ Disparity

☑ Symbol

☑ Idle data

☑ Skip late

☑ Ordered Set Format

Desc: Any occurrence on channel Down of DLLP CRC, TLP LCRC, Delimiter, EDB, Disparity, Symbol, Idle Data, Skip Late, Or

Accessing the Properties Dialog

To see the Properties dialog, first create an event button by clicking the **New Events** button and choosing an event from the menu. Afterwards, open the Event Properties dialog by doing one of the following:

- Double-click an event.
- Right-click an event and select **Properties** from the pop-up menu.
- Click the **Properties** button on the toolbar.

Dialog Settings and Features

Event Icon Preview: This icon shows you which event properties you are editing. The Icon Preview looks exactly like the icon in the Main Display area.

Icon Label: A text box for labeling the button. Whatever you type here appears on the button.

Channels: These controls allow you to select the channel(s) that the Analyzer should search when it is looking for the event.

Count: A counter tells the Analyzer to search for *x* instances of the selected event. For example, if you enter **10**, the Analyzer counts 10 instances of the selected event before it performs whatever action you assign. There are only two counts available in the hardware so if you try to assign more than two, you get a warning. Counters cannot be applied to events with Filter Actions. The maximum counter value is 65,535.

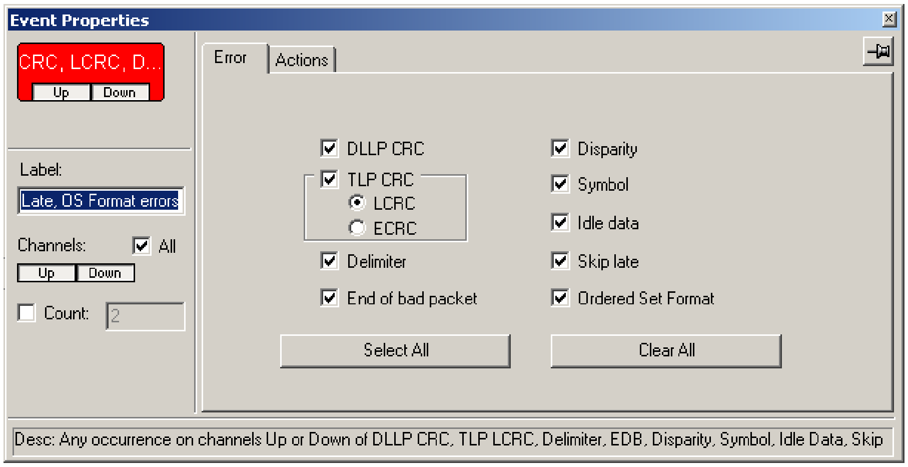
Pin Button: Allows you to **pin** the Properties dialog box to the application so that it does not go away when another object appears such as an event, state or action.

Description String: This area contains a textual description of the event.

Event-specific Settings: The largest part of the Event Properties dialog box. The settings in this area vary for different events. Some events do not have any additional settings (for example, Basic Link Services, Extended Link Services). The Events that do have settings are: Error Event, Data Frame Event, Primitive Event, Advanced Primitive Event, SCSI Command Event, Breakout Board Event, and Timer Event.

Error Properties Dialog

The Error Properties dialog box lets you select specific error types for performing an action. There are two sets of Error types: Packet Errors and Idle Errors.



Event Properties

CRC, LCRC, D...

Label: Late, OS Format errors

Channels: ☑ All

□ Count: 2

Error | Actions

☑ DLLP CRC
☑ TLP CRC
● LCRC
○ ECRC
☑ Delimiter
☑ End of bad packet

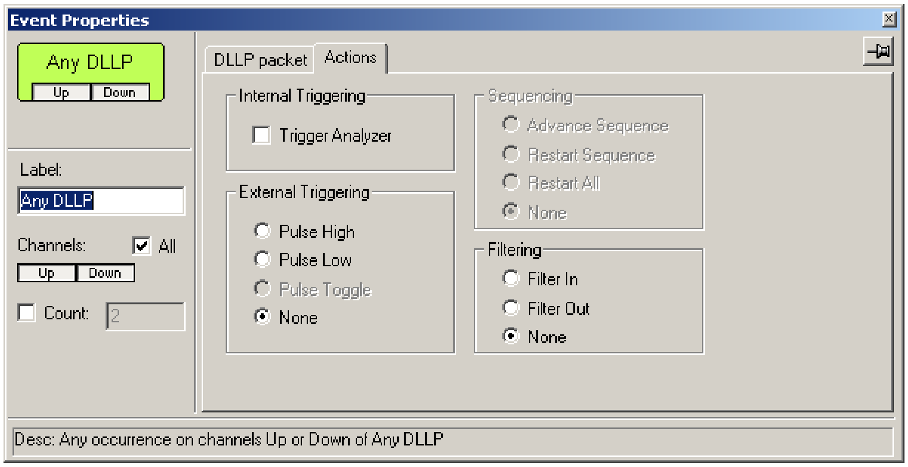
☑ Disparity
☑ Symbol
☑ Idle data
☑ Skip late
☑ Ordered Set Format

Select All | Clear All

Desc: Any occurrence on channels Up or Down of DLLP CRC, TLP LCRC, Delimiter, EDB, Disparity, Symbol, Idle Data, Skip

Actions Properties Dialog

The Actions Properties dialog box.



Event Properties

Any DLLP

Label: Any DLLP

Channels: ☑ All

□ Count: 2

DLLP packet | Actions

Internal Triggering
□ Trigger Analyzer

External Triggering
○ Pulse High
○ Pulse Low
○ Pulse Toggle
● None

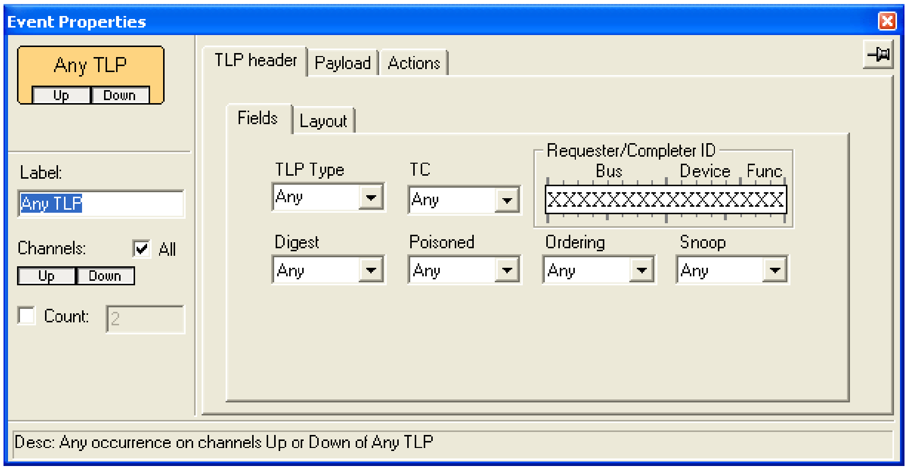
Sequencing
○ Advance Sequence
○ Restart Sequence
○ Restart All
● None

Filtering
○ Filter In
○ Filter Out
● None

Desc: Any occurrence on channels Up or Down of Any DLLP

TLP Header Properties Dialog

The TLP Header Properties dialog box.



Label: Any TLP

Channels: ☑ All

Up Down

□ Count: 2

TLP Type: Any

TC: Any

Requester/Completer ID: XXXXXXXXXXXXXXXX

Digest: Any

Poisoned: Any

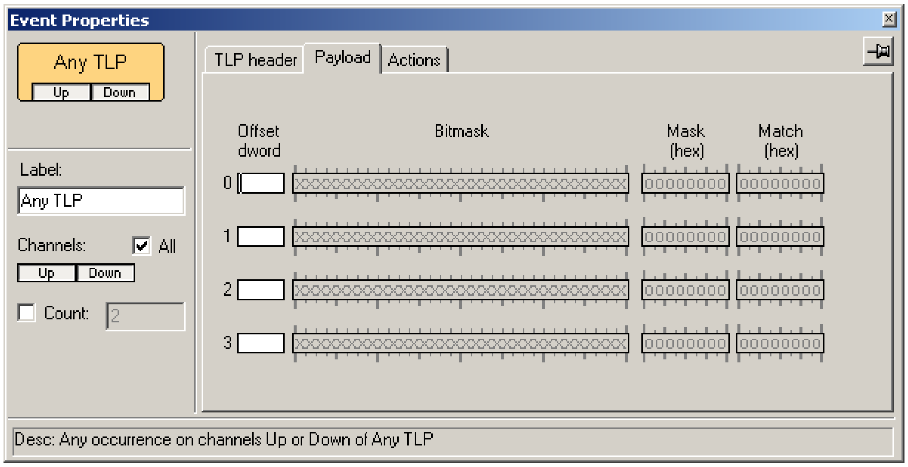
Ordering: Any

Snoop: Any

Desc: Any occurrence on channels Up or Down of Any TLP

Payload Properties Dialog

The Payload Properties dialog box.



Label: Any TLP

Channels: ☑ All

Up Down

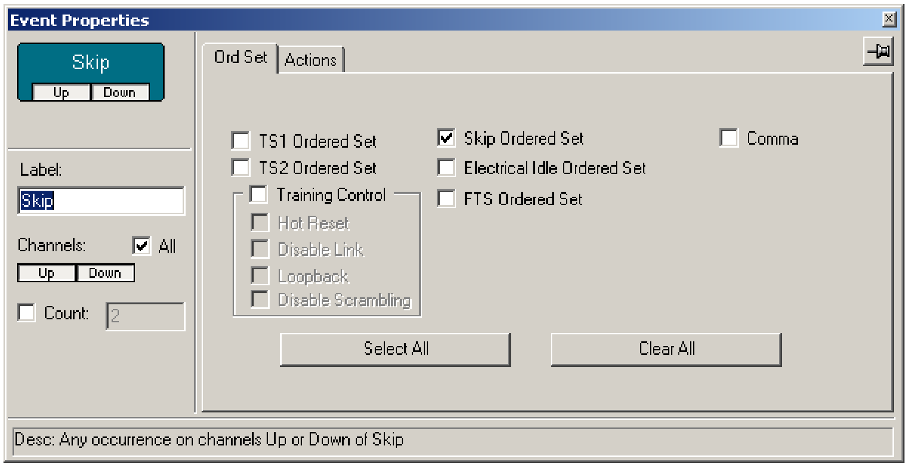
□ Count: 2

Offset dword		Bitmask	Mask (hex)	Match (hex)
0		XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	00000000	00000000
1		XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	00000000	00000000
2		XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	00000000	00000000
3		XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	00000000	00000000

Desc: Any occurrence on channels Up or Down of Any TLP

Ordered Set Properties Dialog

The Ordered Set Properties dialog box.



Event Properties

Skip

Up Down

Label: Skip

Channels: ☑ All

Up Down

□ Count: 2

Ord Set | Actions

□ TS1 Ordered Set
□ TS2 Ordered Set
□ Training Control
□ Hot Reset
□ Disable Link
□ Loopback
□ Disable Scrambling
☑ Skip Ordered Set
□ Electrical Idle Ordered Set
□ FTS Ordered Set
□ Comma

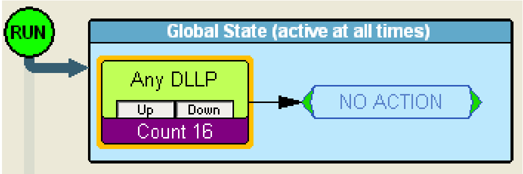
Select All Clear All

Desc: Any occurrence on channels Up or Down of Skip

8.10 Counting Events

Triggers can be set on multiple instances of an event. For example, you can set a trigger to occur following five instances of any DLP. To configure the Analyzer to look for multiple events, you enable **Counters**. Counters tell the Analyzer how many occurrences of an event it should wait for before triggering.
For example, **Trigger following the 16th occurrence of an error**.

Counters enable triggers to be set that are based on a count of events. For example, you could use a counter to **Trigger following the 16th occurrence of a DLLP message**



To use a counter, follow these steps:

Step 1 Click an event. This causes an arrow to appear.

Step 2 Click one of the two counters (it does not matter which you use). This causes the counter to attach itself to the bottom of the event (shown above). An arrow automatically connects the counter to the Trigger button.

To change the counter value:

Step 1 Click the small blue dot in the upper-left corner of the counter button. A menu appears.

Step 2 Select **Change Counter Value**.

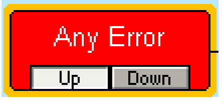
Step 3 Enter a new value in the pop-up dialog box. This causes the new value to appear in the counter button.

How to Set a Counter

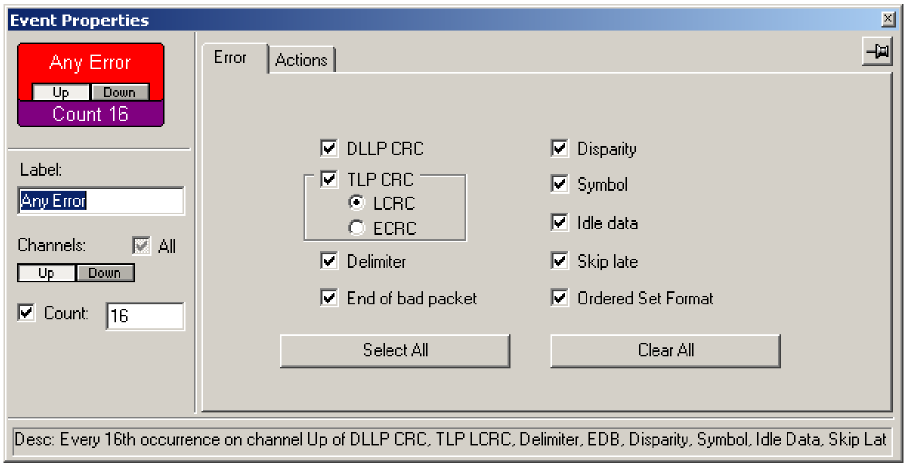
To set a counter:

Step 1 Open the Recording Rules page, select an event, and drag it to the **Global State** or **Sequence** cells. For details on these steps, see “Creating Recording Rules” on page 143.

Step 2 Counts can only be set on a per channel basis, so press the **Up** or **Down** channel buttons to select the channel on which the count is performed.



Step 3 Right-click the selected event and select **Add Counter** from the menu to open the Properties dialog.



Label: Any Error

Channels: ☑ All

Up Down

☑ Count: 16

☑ DLLP CRC

☑ TLP CRC

● LCRC

○ ECRC

☑ Delimiter

☑ End of bad packet

☑ Disparity

☑ Symbol

☑ Idle data

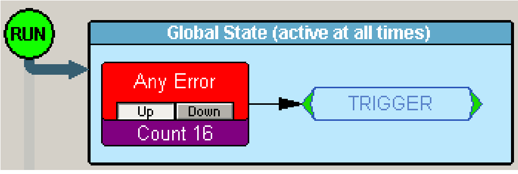
☑ Skip late

☑ Ordered Set Format

Desc: Every 16th occurrence on channel Up of DLLP CRC, TLP LCRC, Delimiter, EDB, Disparity, Symbol, Idle Data, Skip Lat

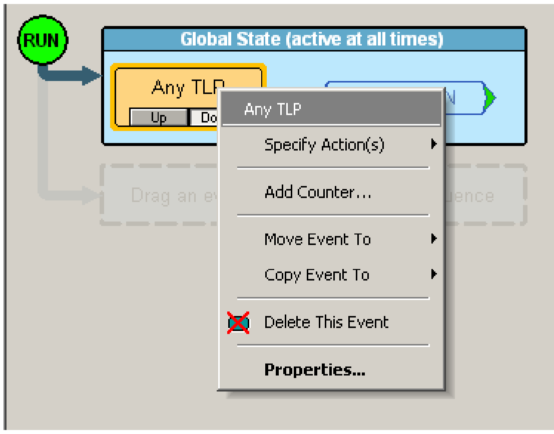
Step 4 In the text box to the right of the label Count enter a value. Make sure the checkbox to the left of the word Count is checked.

Step 5 Click the **X** in the top right corner of the dialog box to close the dialog. A counter button should appear just below your selected event.

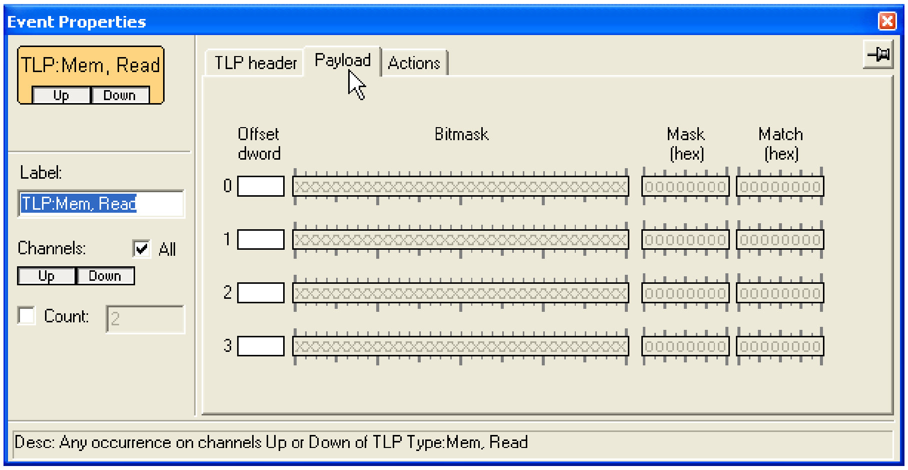


8.11 Creating a Payload Match

You can configure PE*Tracer* to trigger, count, and restart on a specific payload pattern within a Transaction Layer Packet (TLP). In Recording Rules, create a **New** event of the desired TLP type. An icon appears that represents that event type in the Available Events area. Right-click the icon and select **Properties** to bring up the Properties dialog box for the event.



Select the **Payload** tab to bring up the pane that lets you configure payload pattern. You can match up to four patterns, labeled 0 through 3.



Label: TLP:Mem, Read

Channels: ☑ All

Up Down

□ Count: 2

Desc: Any occurrence on channels Up or Down of TLP Type:Mem, Read

For each pattern, you specify the following:

Offset dword: Indicates the DWORD you want to match within the payload (range is 0 through 1024). For example, if you want the Analyzer match a pattern in the first DWORD of the payload, enter 0. If you want the Analyzer to match a pattern in the thirty-fifth DWORD, enter 35.

Bitmask: Indicates which bits you want to match for the DWORD indicated. You can enter any of the following in the bitmask area:

- X (bit not relevant)
- 1 (present)
- 0 (bit not present)

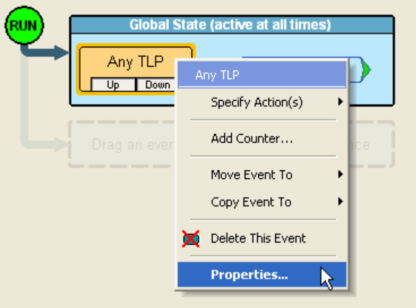
Note: The Mask (hex) and Match (hex) entries represent bitmask in hex format. As you type entries in the bitmask, corresponding values appear in fields in the hex Mask and Match. Alternately, you can type values in the hex Mask and Match, and corresponding values appear in the bitmask.

Example 1

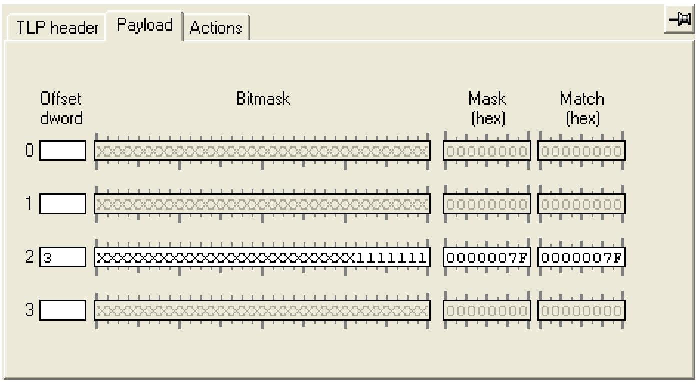
To tell the Analyzer to trigger on any payload with the binary pattern 1111111 in the third DWORD:

Step 1 Create an **Any TLP** event in the Available Events area of Recording Rules. Drag the event to the **Global State** cell.

Step 2 Open the **Properties** dialog box for the event and bring up the **Payload** pane.



Step 3 Enter **3** in the Offset dword area of any of the four patterns (0 to 3). For purposes of this example, we choose pattern 2.



	Offset dword	Bitmask	Mask (hex)	Match (hex)
0		XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	00000000	00000000
1		XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	00000000	00000000
2	3	XXXXXXXXXXXXXXXXXXXXXXXXX1111111	0000007F	0000007F
3		XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	00000000	00000000

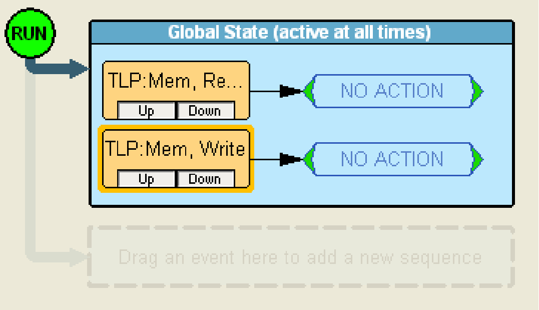
Step 4 Enter the following in pattern 2: **1111111**.

Example 2

To tell the Analyzer to trigger on any payload with the binary pattern 1111111 in the third DWORD, OR to trigger on any occurrence of Memory Write:

Step 1 To set up the first condition, do steps 1 through 4 in Example 1.

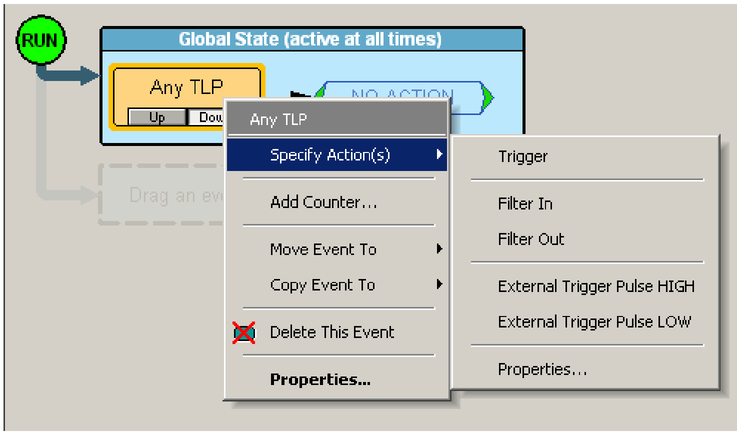
Step 2 Create a **Memory Write** event in the Available Events area of Recording Rules. Drag the **Memory Write** icon to the same cell as the **Any Event** icon from Example 1. The Analyzer now is set to trigger on either of the two conditions.


RUN
Global State (active at all times)
TLP:Mem, Re...
Up
Down
NO ACTION
TLP:Mem, Write
Up
Down
NO ACTION
Drag an event here to add a new sequence

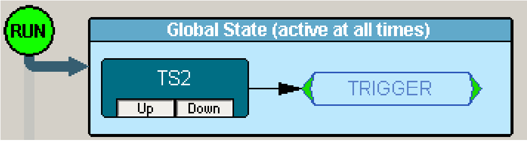
8.12 Trigger

A trigger is one of five actions that can be assigned to an event. (The other four actions are Filter In, Filter Out, Advance the Sequence, and Restart the Sequence.)

To enable a trigger, place an event button into either the **Global State** cell or the **Sequence** cell, then right-click the button and choose an **Action**.



From the sub-menu, select **Trigger**.



The exact end of the recording depends on how you have set the Trigger Position slider in the General page of the Recording Options dialog. This setting determines whether the recording terminates immediately following a trigger, or some time afterwards. See “General Tab” on page 131 for further explanation.

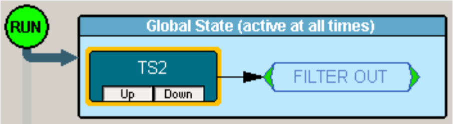
8.13 Filter In and Filter Out

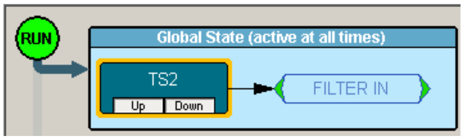
A filter causes the Analyzer to filter in or out specified events from the recording. If events are filtered out of the recording, they are excluded from the Analyzer's buffer and not simply hidden from the trace. The purpose of filtering is to preserve recording memory so you conduct longer recording sessions and exclude events that do not interest you.

A filter causes the Analyzer to filter in or out specified events from the recording so you can preserve recording memory and thereby increase the duration of your recording. Filtering also lets you exclude unwanted data so your trace displays only the traffic that interests you.

To enable or disable filtering, place an event button into either the **Global State** cell or the **Sequence** cell, then right-click the button and choose an **Action** (see previous topic).

From the sub-menu, select **Filter In** or **Filter Out**.





Chapter 9: Reports

Reports assist you in analyzing traffic recorded by the Analyzer. The available reports are:

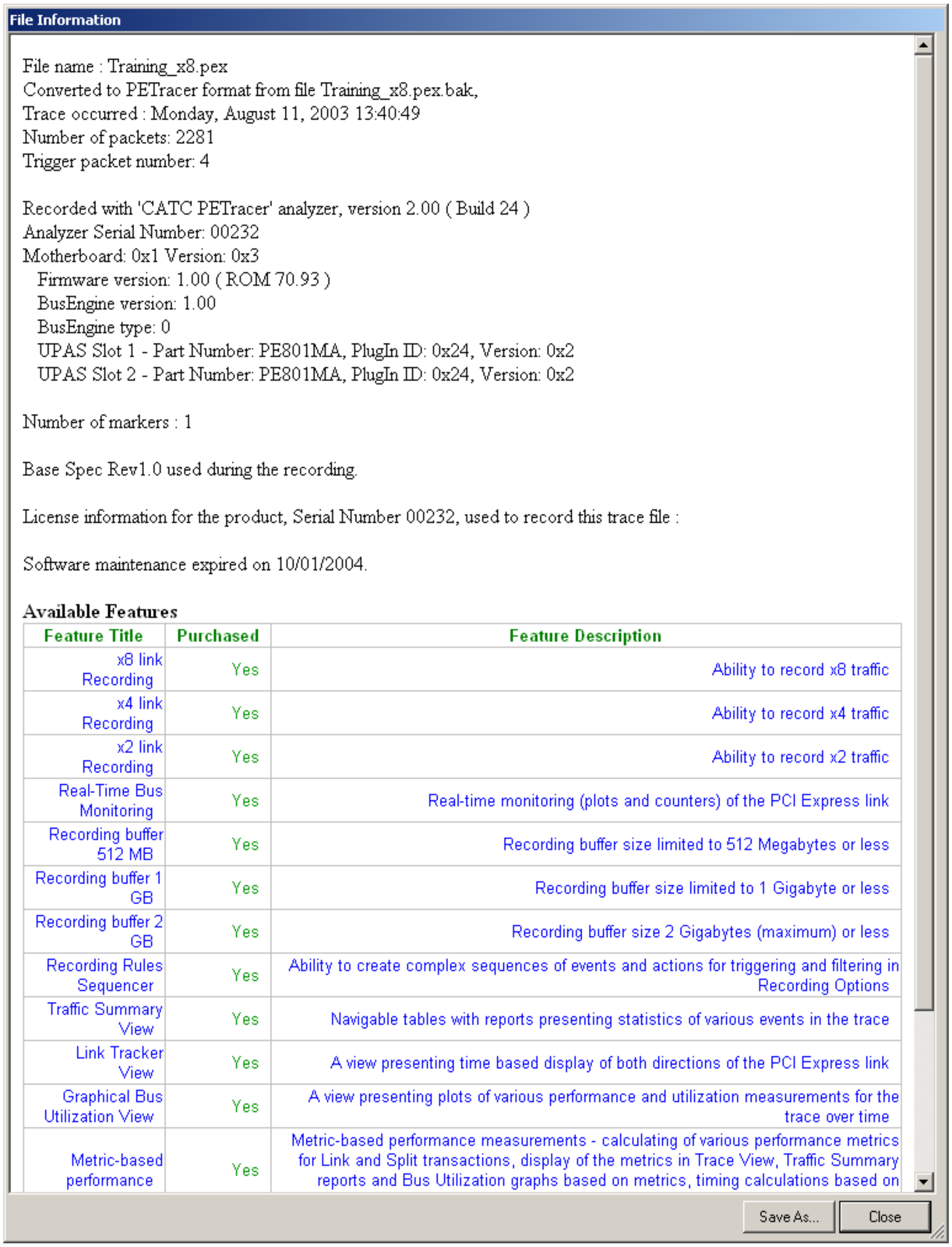
- **File Information**: To view general information about the trace file.
- **Error Summary**: To view a count of errors in a trace file.
- **Timing Calculations**: To view timing measured between two events set within the trace file.
- **Link Tracker**: Displays a detailed chronological view of events.
- **Traffic Summary**: To view a summary of protocol-related information in the trace file summary information about a selected group of items in the trace file (such as a count of particular frame or packet types).
- **Bus Utilization**: To display information on bandwidth usage for the transmit and receive channels.
- **Real Time Statistics**: To display statistical information for the channels.

Reports are available from the Report menu and from buttons on the Tool bar.

9.1 File Information

The File Information window provides a summary on the currently displayed file.

Select **Reports > File Information** to obtain the File Information window.



File name : Training_x8.pex
Converted to PETracer format from file Training_x8.pex.bak,
Trace occurred : Monday, August 11, 2003 13:40:49
Number of packets: 2281
Trigger packet number: 4

Recorded with 'CATC PETracer' analyzer, version 2.00 (Build 24)
Analyzer Serial Number: 00232
Motherboard: 0x1 Version: 0x3
Firmware version: 1.00 (ROM 70.93)
BusEngine version: 1.00
BusEngine type: 0
UPAS Slot 1 - Part Number: PE801MA, PlugIn ID: 0x24, Version: 0x2
UPAS Slot 2 - Part Number: PE801MA, PlugIn ID: 0x24, Version: 0x2

Number of markers : 1

Base Spec Rev1.0 used during the recording.

License information for the product, Serial Number 00232, used to record this trace file :

Software maintenance expired on 10/01/2004.

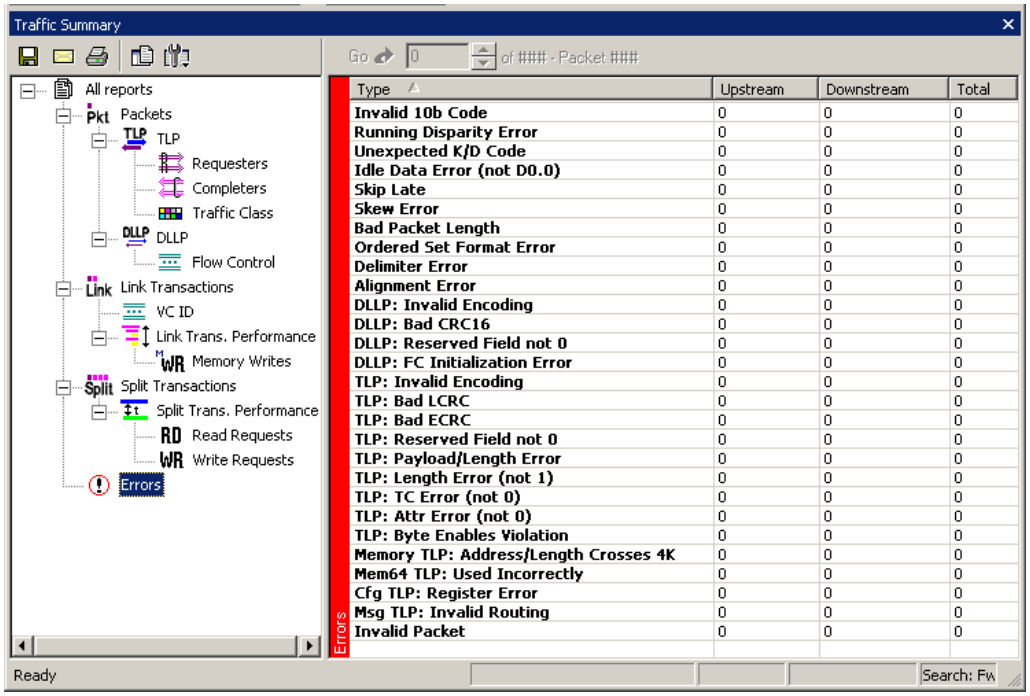
Available Features

Feature Title	Purchased	Feature Description
x8 link Recording	Yes	Ability to record x8 traffic
x4 link Recording	Yes	Ability to record x4 traffic
x2 link Recording	Yes	Ability to record x2 traffic
Real-Time Bus Monitoring	Yes	Real-time monitoring (plots and counters) of the PCI Express link
Recording buffer 512 MB	Yes	Recording buffer size limited to 512 Megabytes or less
Recording buffer 1 GB	Yes	Recording buffer size limited to 1 Gigabyte or less
Recording buffer 2 GB	Yes	Recording buffer size 2 Gigabytes (maximum) or less
Recording Rules Sequencer	Yes	Ability to create complex sequences of events and actions for triggering and filtering in Recording Options
Traffic Summary View	Yes	Navigable tables with reports presenting statistics of various events in the trace
Link Tracker View	Yes	A view presenting time based display of both directions of the PCI Express link
Graphical Bus Utilization View	Yes	A view presenting plots of various performance and utilization measurements for the trace over time
Metric-based performance	Yes	Metric-based performance measurements - calculating of various performance metrics for Link and Split transactions, display of the metrics in Trace View, Traffic Summary reports and Bus Utilization graphs based on metrics, timing calculations based on

9.2 Error Summary

The Error Summary dialog box displays the number of errors for each event and the packet containing the errors.

Select **Reports > Error Summary** to obtain the Error Summary dialog box.

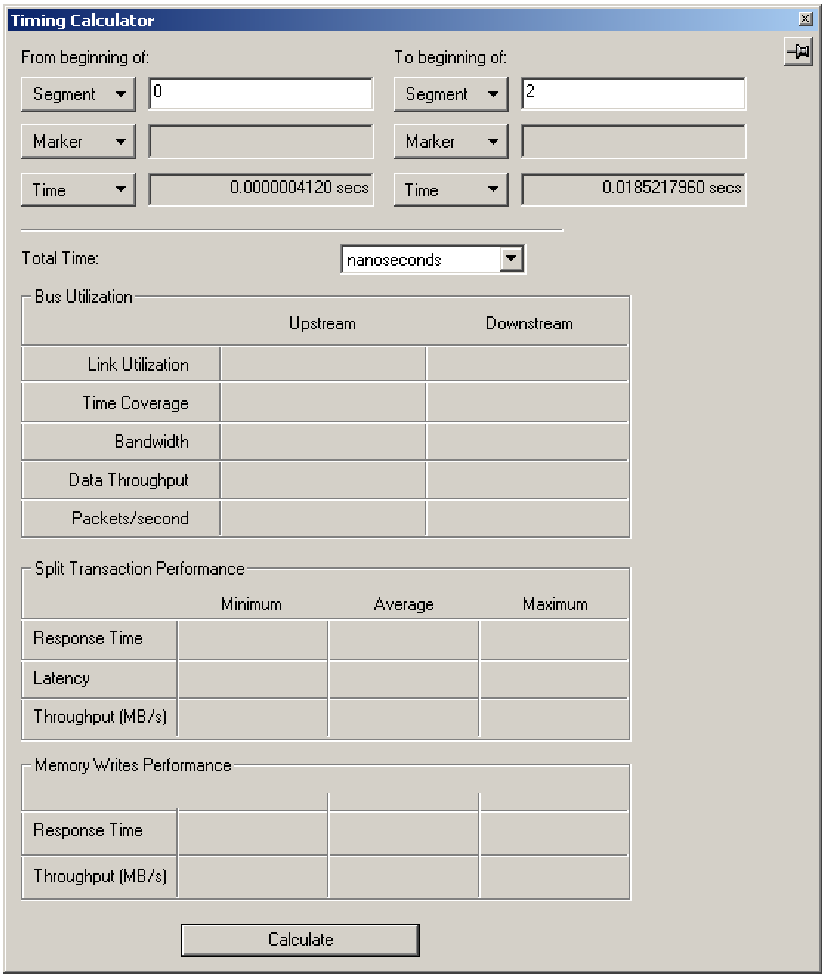


Type	Upstream	Downstream	Total
Invalid 10b Code	0	0	0
Running Disparity Error	0	0	0
Unexpected K/D Code	0	0	0
Idle Data Error (not D0.0)	0	0	0
Skip Late	0	0	0
Skew Error	0	0	0
Bad Packet Length	0	0	0
Ordered Set Format Error	0	0	0
Delimiter Error	0	0	0
Alignment Error	0	0	0
DLLP: Invalid Encoding	0	0	0
DLLP: Bad CRC16	0	0	0
DLLP: Reserved Field not 0	0	0	0
DLLP: FC Initialization Error	0	0	0
TLP: Invalid Encoding	0	0	0
TLP: Bad LCRC	0	0	0
TLP: Bad ECRC	0	0	0
TLP: Reserved Field not 0	0	0	0
TLP: Payload/Length Error	0	0	0
TLP: Length Error (not 1)	0	0	0
TLP: TC Error (not 0)	0	0	0
TLP: Attr Error (not 0)	0	0	0
TLP: Byte Enables Violation	0	0	0
Memory TLP: Address/Length Crosses 4K	0	0	0
Mem64 TLP: Used Incorrectly	0	0	0
Cfg TLP: Register Error	0	0	0
Msg TLP: Invalid Routing	0	0	0
Invalid Packet	0	0	0

9.3 Timing and Bus Usage Calculations

The Timing and Bus Usage Calculator allows you to calculate time between packets.

Select **Reports > Timing Calculations** to obtain the Timing Calculator dialog box.



Timing Calculator

From beginning of:		To beginning of:	
Segment	0	Segment	2
Marker		Marker	
Time	0.0000004120 secs	Time	0.0185217960 secs

Total Time: nanoseconds

Bus Utilization

	Upstream	Downstream
Link Utilization		
Time Coverage		
Bandwidth		
Data Throughput		
Packets/second		

Split Transaction Performance

	Minimum	Average	Maximum
Response Time			
Latency			
Throughput (MB/s)			

Memory Writes Performance

Response Time			
Throughput (MB/s)			

Total Time: Total time from beginning of the first unit to beginning of the second unit.

Bus Utilization

This portion of the Timing Calculator window gives values that are cumulative for all packets during the timing period. For example, throughput is combined throughput of all packets during the timing period.

Upstream is from endpoint devices to the root complex. Downstream is from the root complex to endpoint devices.

Link Utilization: Percentage of non-idle symbols in total number of symbols transferred.

Time Coverage: Percentage of non-idle symbol times in total number of symbol times. (Non-idle symbol time occurs when at least on one of the lanes there were non-idle symbols transferred.)

Bandwidth: Number of non-idle symbol bits transferred per second.

Data Throughput: Number of TLP payload bytes transferred per second.

Packets/second: Number of packets transferred per second.

Split Transaction Performance

This portion of the Timing Calculator window gives minimum, maximum, and average values for all Split transactions during the timing period. For example, minimum throughput is throughput of the Split transaction that passes the least amount of data. Maximum throughput is throughput of the Split transaction that passes the most amount of data. Average throughput is the average calculated for all Split transactions during the timing period.

Response Time: The time it took to transmit this Split transaction on the PE link, from the beginning of the first packet in the Split transaction to the end of the last packet in the Split transaction.

Latency: The time measured from the end of the request transaction to the first completion transmitted in response to the request within this Split transaction.

Throughput: The transaction payload divided by response time, expressed in megabytes per second.

Memory Writes Performance

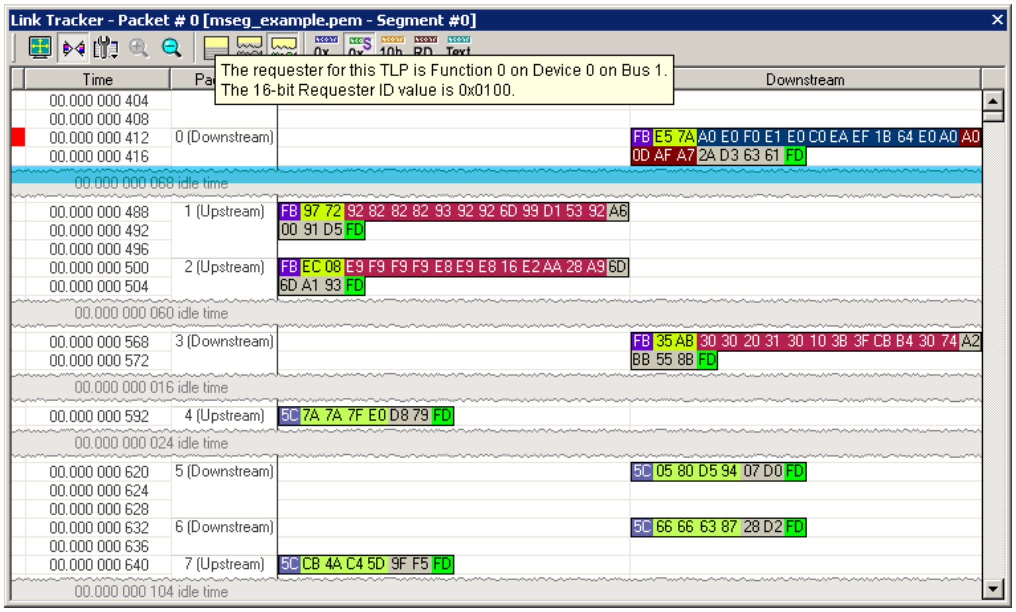
This portion of the Timing Calculator window gives minimum, maximum, and average values for all Memory Write transactions during the timing period. For example, minimum throughput is throughput of the Memory Write transaction that passes the least amount of data. Maximum throughput is throughput of the Memory Write transaction that passes the most amount of data. Average throughput is the average calculated for all Memory Write transactions during the timing period.

Response Time: The time it took to transmit this Memory Write on the PE link, from the beginning of the first packet in the Memory Write to the end of the last packet in the Memory Write.

Throughput: The Memory Write payload divided by response time, expressed in megabytes per second.

9.4 Link Tracker

The Link Tracker window displays a detailed chronological view of events. Events are shown on a channel-by-channel basis in columns within the window.



Link Tracker - Packet # 0 [mseg_example.pem - Segment #0]

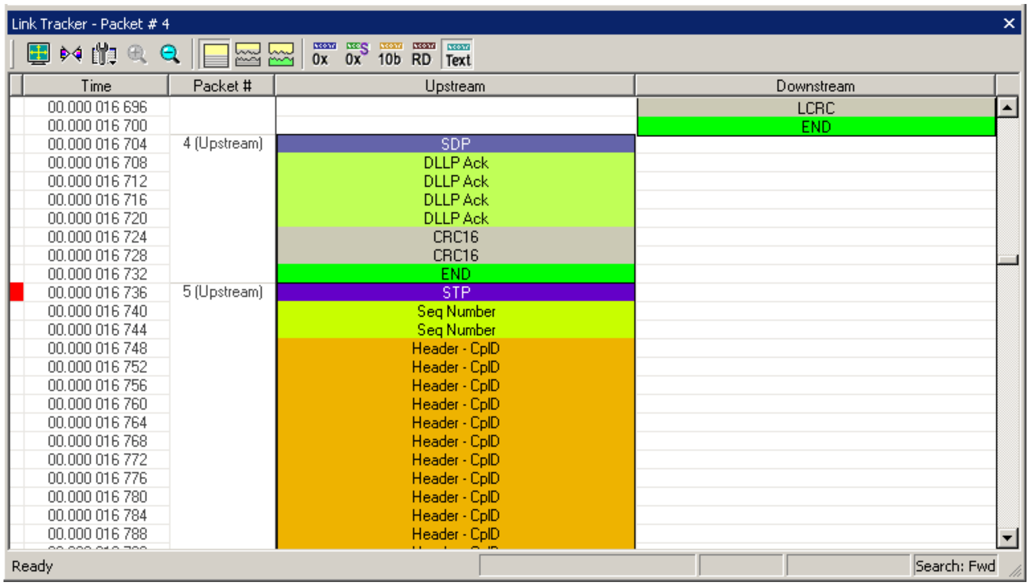
The requester for this TLP is Function 0 on Device 0 on Bus 1.
The 16-bit Requester ID value is 0x0100.

Time	Pa		Downstream
00.000 000 404			
00.000 000 408			
00.000 000 412	0 (Downstream)		FB E5 7A A0 E0 F0 E1 E0 C0 EA EF 1B 64 E0 A0 A0
00.000 000 416			0D AF A7 2A D3 63 61 FD
00.000 000 068 idle time			
00.000 000 488	1 (Upstream)	FB 97 72 92 82 82 82 93 92 92 6D 99 D1 53 92 A6	
00.000 000 492		00 91 D5 FD	
00.000 000 496			
00.000 000 500	2 (Upstream)	FB EC 08 E9 F9 F9 F9 E8 E9 E8 16 E2 AA 28 A9 6D	
00.000 000 504		6D A1 93 FD	
00.000 000 060 idle time			
00.000 000 568	3 (Downstream)		FB 35 AB 30 30 20 31 30 10 3B 3F CB B4 30 74 A2
00.000 000 572			BB 55 8B FD
00.000 000 016 idle time			
00.000 000 592	4 (Upstream)	5C 7A 7A 7F E0 D8 79 FD	
00.000 000 024 idle time			
00.000 000 620	5 (Downstream)		5C 05 80 D5 94 07 D0 FD
00.000 000 624			
00.000 000 628			
00.000 000 632	6 (Downstream)		5C 66 66 63 87 28 D2 FD
00.000 000 636			
00.000 000 640	7 (Upstream)	5C CB 4A C4 5D 9F F5 FD	
00.000 000 104 idle time			

Each time slot in the vertical axis represents the minimum time that a DWORD requires to traverse the bus.

Toolbar: Presents buttons for changing the format of the Link Tracker window.

Main Display Area: Displays traffic chronologically as it occurred in the recording. The window divides into columns: the first column shows time and traffic is shown on a channel-by-channel basis in the columns on the right.



Link Tracker - Packet # 4

Time	Packet #	Upstream	Downstream
00.000 016 696			LCRC
00.000 016 700			END
00.000 016 704	4 (Upstream)	SDP	
00.000 016 708		DLLP Ack	
00.000 016 712		DLLP Ack	
00.000 016 716		DLLP Ack	
00.000 016 720		DLLP Ack	
00.000 016 724		CRC16	
00.000 016 728		CRC16	
00.000 016 732		END	
00.000 016 736	5 (Upstream)	STP	
00.000 016 740		Seq Number	
00.000 016 744		Seq Number	
00.000 016 748		Header - CplD	
00.000 016 752		Header - CplD	
00.000 016 756		Header - CplD	
00.000 016 760		Header - CplD	
00.000 016 764		Header - CplD	
00.000 016 768		Header - CplD	
00.000 016 772		Header - CplD	
00.000 016 776		Header - CplD	
00.000 016 780		Header - CplD	
00.000 016 784		Header - CplD	
00.000 016 788		Header - CplD	

Ready Search: Fwd

Using the Link Tracker Window

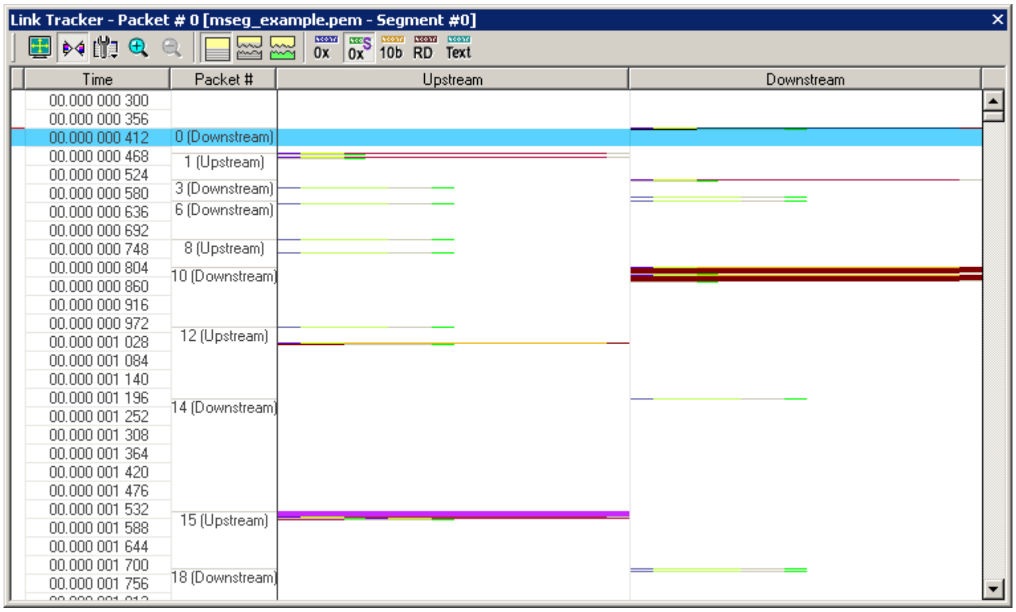
The Link Tracker window can be reformatted in several ways.

Zooming In and Out

Zooming out can give you a quick, high-level view of a trace. A fully zoomed out trace only shows columns and colored lines. Using the colors, you can see what types of traffic run through the trace.

Further information can be obtained on any point of interest in the trace by positioning your mouse pointer over it. Tool tips provide detailed description of events.

Note: When fully zoomed out, the smallest graphical unit is the DWORD, represented by a single line. Zooming out makes the trace appear smaller and increases the time scale in the first column.



Link Tracker - Packet # 0 [mseg_example.pem - Segment #0]

Time	Packet #	Upstream	Downstream
00.000 000 300			
00.000 000 356			
00.000 000 412	0 (Downstream)		
00.000 000 468	1 (Upstream)		
00.000 000 524			
00.000 000 580	3 (Downstream)		
00.000 000 636	6 (Downstream)		
00.000 000 692			
00.000 000 748	8 (Upstream)		
00.000 000 804	10 (Downstream)		
00.000 000 860			
00.000 000 916			
00.000 000 972			
00.000 001 028	12 (Upstream)		
00.000 001 084			
00.000 001 140			
00.000 001 196			
00.000 001 252	14 (Downstream)		
00.000 001 308			
00.000 001 364			
00.000 001 420			
00.000 001 476			
00.000 001 532			
00.000 001 588	15 (Upstream)		
00.000 001 644			
00.000 001 700			
00.000 001 756	18 (Downstream)		

Collapsing Idle Time, Enabling Tool tips, and Resetting Column Widths

Click the **View Options** button to open a menu with options for formatting the display. Three options are presented:

Collapsible Idle Time: Opens a dialog box for setting the Idle time value. Setting a value tells the Analyzer when to collapse Idle times and display them as grayed out strips within the Bus View window.

Tooltip Display: Opens a menu with options for adding content to Tooltips. Tooltips display when you position the mouse pointer over an item in the Bus View window. The options are:

- Tooltips Display Values
- Tooltips Display Scrambled Values
- Tooltips Display 10 bit Codes
- Tooltips Display Symbols

Time Format: Seconds or Clock

Reset Column Widths: This option resets column widths to their defaults and enables columns to resize themselves automatically any time the application window is resized. Normally, columns automatically resize themselves if the application window is made larger or smaller. However, if you manually resize any columns in the Bus View window, column widths become static. Thereafter, if you resize the application window, the Bus View columns do not adjust automatically. Reset Column Widths re-enables the automatic resizing capability.

Reset Columns Order: Return to default column sequence.

Docking and Undocking the Window

You can undock the Link Tracker window by double-clicking the blue title bar along the left side of the window. Once undocked, the window can be dragged anywhere in the application. To redock, double-click again on the title bar.

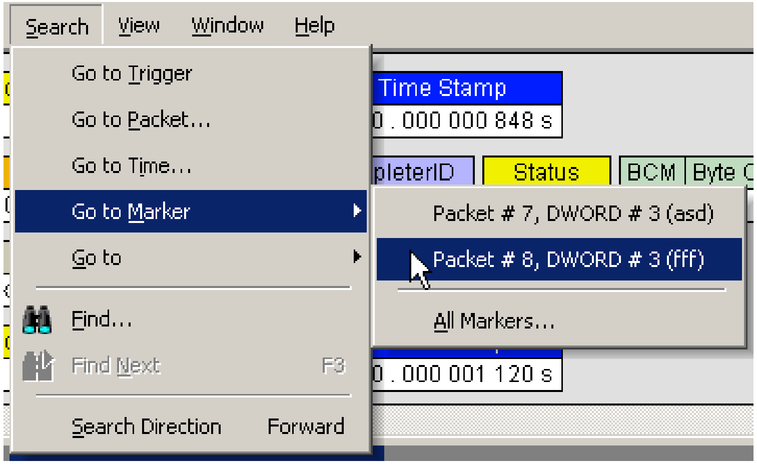
Setting Markers

Markers can be set on any event within the Link Tracker window.

To set a marker, right-click an event, then select **Set Marker** from the pop-up menu.

Once marked, you can navigate to events with the **Go to Marker** command in the Search menu.

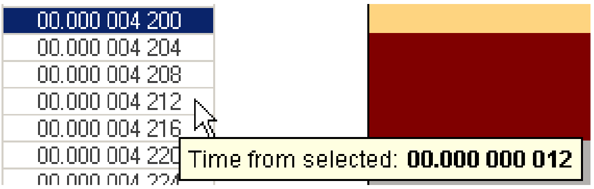
Markers set in the Link Tracker window display the packet number and DWORD number. In contrast, markers set in the Trace window just show the packet number.



Calculating Time between DWORDs

You can calculate time between DWORDs by clicking an event and then positioning your mouse pointer over a second event and reading the ensuing Tool tip.

Click the **time value** for the first event. Scroll down through the trace to the second event and position the mouse pointer above its time value. A Tool tip appears showing the time interval between the first and second events.



Time from selected: **00.000 000 012**

Hiding Traffic

You can hide Idles and other data from the Link Tracker window by clicking the **Hide** buttons on the toolbar.

Link Tracker Buttons

The Link Tracker window has a row of buttons for changing the format of the displayed data and for exporting data: The buttons have the following functions:

Full Screen. Expands the Link Tracker window to fill the entire screen.

View Options. Opens a menu with three options:

- **Collapsible Idle Time** (Collapse Idle Bigger Than **n** nanoseconds. **Note:** Does not affect Collapse Idle Plus.)
- **Tooltip Display** (Values, Scrambled Values, 10-bit Codes, Symbols)
- **Time Format** (Seconds, Clock)
- **Reset Columns Widths** (return to default widths)
- **Reset Columns Order** (return to default column sequence)

See “Using the Link Tracker Window” on page 171 for further details.

Synchronize Trace View. Synchronizes the Trace View and Link Tracker windows so that a move in one window repositions the other.

Because of the differences in scale and logic between the Link Tracker and Trace view window, scrolling produces different effects depending on which window is being scrolled.

Scrolling in the trace window causes the Link Tracker window to rapidly jump from event to event. Long periods of idle time are thus skipped.

Scrolling in the Link Tracker window, in contrast, produces modest movements within the trace window.

Scrolling in the Link Tracker window causes the trace window to pause until the beginning of a packet is displayed. At that point, the trace window repositions itself. While scrolling long Idle periods or through the contents of a packet, the trace window does not move.

Zoom In

Zoom Out

Continuous Time Scale. No collapsing.

Collapse Idle. Do not show some periods of Link being idle.

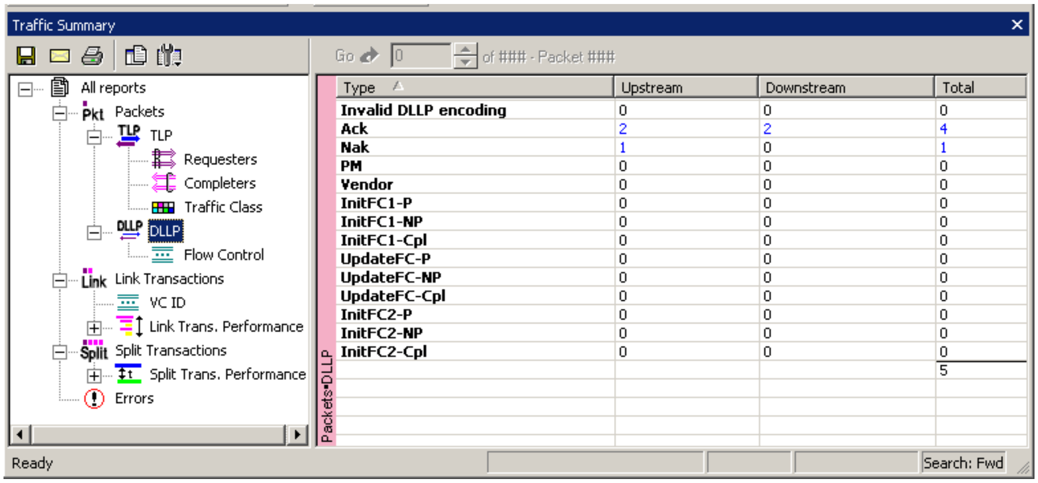
Collapse Idle Plus. Do not show periods of Link being idle.

0x	Show Values
0x S	Show Scrambled Values
10b	Show 10b Codes
RD	Show Symbols
Text	Show Text

9.5 Traffic Summary

The Traffic Summary window summarizes the traffic in the current trace. The left side of the window displays a tree of protocol levels. The right side displays a summary of traffic for the displayed levels.

Select **Reports > Traffic Summary** or click to display the Traffic Summary window.



Type	Upstream	Downstream	Total
Invalid DLLP encoding	0	0	0
Ack	2	2	4
Nak	1	0	1
PM	0	0	0
Vendor	0	0	0
InitFC1-P	0	0	0
InitFC1-NP	0	0	0
InitFC1-Cpl	0	0	0
UpdateFC-P	0	0	0
UpdateFC-NP	0	0	0
UpdateFC-Cpl	0	0	0
InitFC2-P	0	0	0
InitFC2-NP	0	0	0
InitFC2-Cpl	0	0	0
			5

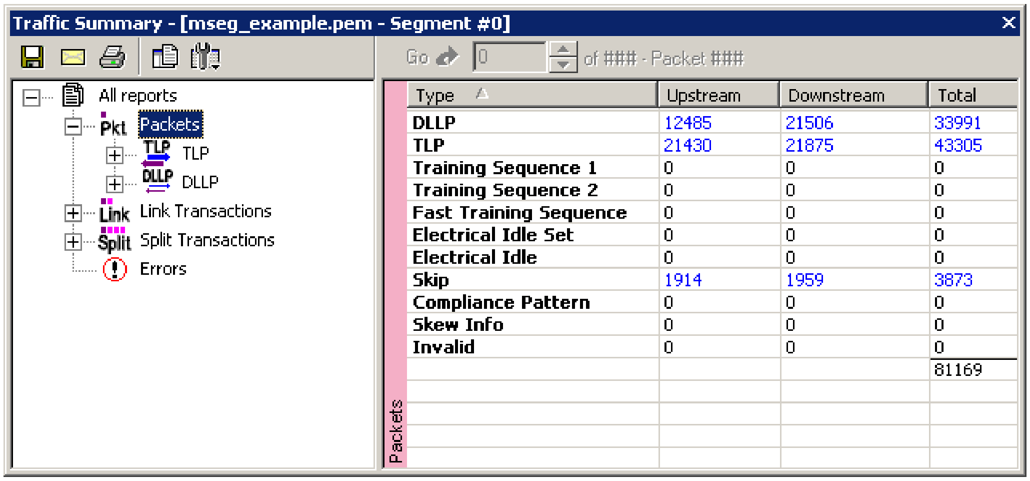
Buttons at the top of the Traffic Summary window change the display format and enable data to be exported to email, file, or the printer.

Using the Traffic Summary Window to Search the Trace

You can use the Traffic Summary window to move the trace to packets of interest:

Step 1 Click one of the numbers in the right side of the Traffic Summary window.

The trace jumps to the first instance of the selected protocol.



Type	Upstream	Downstream	Total
DLLP	12485	21506	33991
TLP	21430	21875	43305
Training Sequence 1	0	0	0
Training Sequence 2	0	0	0
Fast Training Sequence	0	0	0
Electrical Idle Set	0	0	0
Electrical Idle	0	0	0
Skip	1914	1959	3873
Compliance Pattern	0	0	0
Skew Info	0	0	0
Invalid	0	0	0
			81169

Step 2 Click the **up** or **down** arrows ▲▼ of. The trace jumps forward or backward through the display to the next instance of selected protocol level.

Traffic Summary Buttons

Buttons at the top of the Traffic Summary dialog box provide options for exporting the data or formatting its appearance:

💾 Save. Saves Traffic Summary results into an HTML format.

✉ Email. Attaches an HTML file of the results to a new email.

🖨 Print. Prints results.

📄 Text. Displays results in HTML format.

🛠 Options. Opens a drop-down menu with the following options:

- **Grid Lines**: Displays/Hides grid lines
- **Row Selection**: Allows entire rows to be selected
- **Tight Columns**: Reformats column widths to match data
- **Event Navigation**:

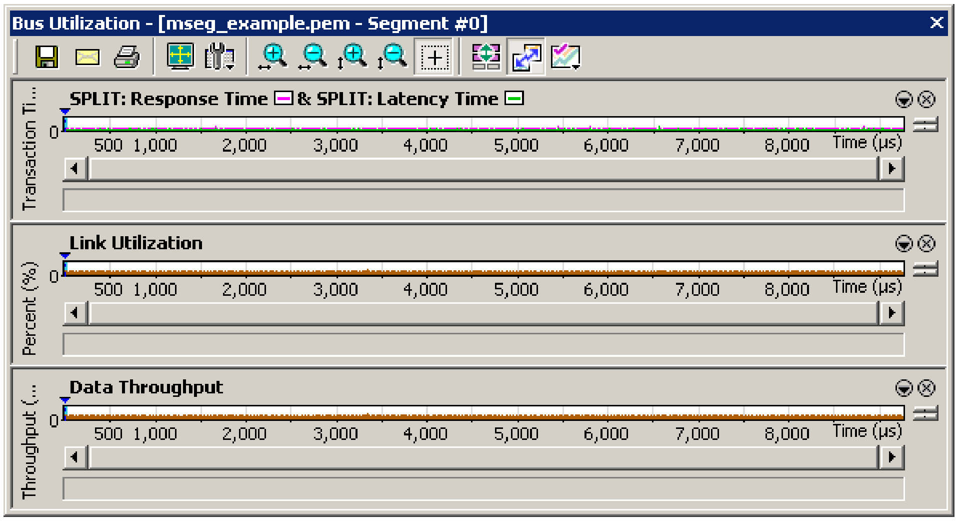
  Skip Hidden Items

  Show Hidden Items

  Prompt each time

9.6 Bus Utilization

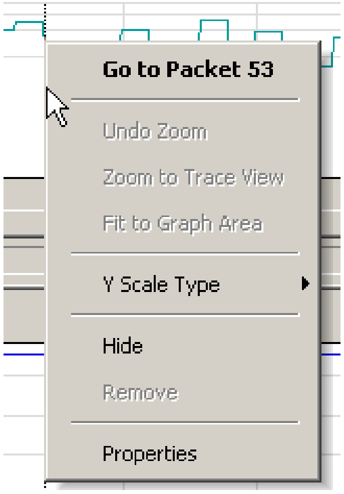
Select **Report > Bus Utilization** from the menu or press 📊 to open the Bus Utilization window.



The Bus Utilization window displays information on bandwidth use for the transmit and receive channels.

Bus Utilization Pop-up Menu

You can reformat the display by right-clicking a graph and making a selection from the Bus Utilization pop-up menu.



Go to Packet #: Relocates the trace to the selected packet number.

Undo Zoom: If you have zoomed in, this command undoes the zoom.

Zoom to Trace View: Zooms in on graph to show traffic currently displayed in the trace screen.

Fit to Graph Area: Redisplays graph so that the entire trace fits inside graph area.

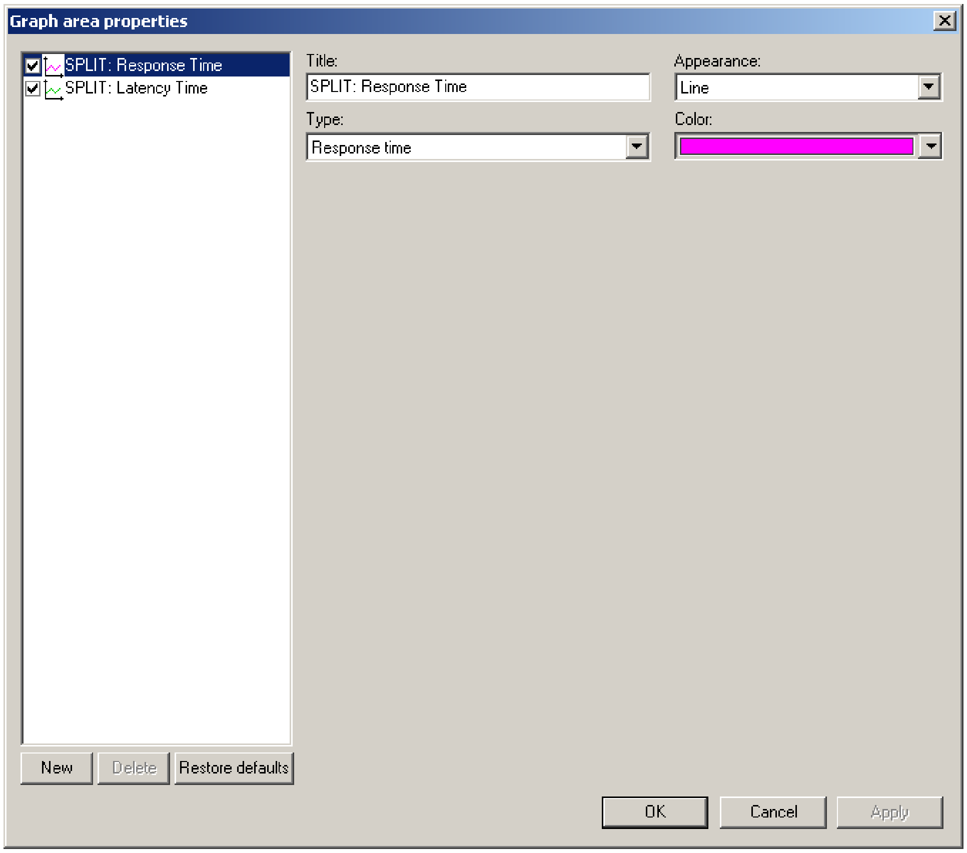
Y Scale Type:

- **Linear**: Converts display to linear format
- **Logarithmic**: Converts display to logarithmic format

Hide: Hides the selected graph

Remove: Allows you to remove any graph that you created via the **New** command

Properties: Opens a dialog box with options for changing the Title, Type, Appearance, and Color of the graphs.



☑ SPLIT: Response Time

☑ SPLIT: Latency Time

Title: SPLIT: Response Time

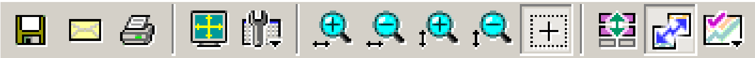
Appearance: Line

Type: Response time

Color: 🟪

Bus Utilization Buttons

The Bus Utilization window buttons allow you to reformat the display and export data.



Button	Function
💾	Save. Saves Bus Utilization data to a bitmap file (***.bmp**).
✉	Email. Opens an email and attaches a bitmap file of the Bus Utilization data.
🖨	Print. Prints the Bus Utilization data.
🖥	Full Screen. Maximizes the Bus Utilization window.
🛠	View Settings. Opens a sub-menu with the following choices: • Orient Horizontally • Tile Vertically • Show Markers • Show Plumbline • Status >> Bar Tool tips None • Grid Lines >> Both Axes X Axis Y Axis No Grid • Grid on Top • Fonts & Colors
🔍	Horizontal zoom in
🔍	Horizontal zoom out

Vertical zoom in

Vertical zoom out

Click and Drag zoom. Click and drag to zoom in on a part of the graph.

Select Range. Displays a dialog box for selecting a packet range.

Synchronize Graph Areas. If two or more graphs are displayed, this button synchronizes the graphs to one another. Once synchronized, the positioning slider of one graph moves the other graphs.

Graph Areas. Provides options for creating and displaying additional graphs of data lengths, packet lengths, and percentage of bus used.

9.7 Using Unit Metrics

For every protocol unit at the Link or Split transaction level, PE*Tracer*™ calculates and displays a set of metrics. Metrics are measurements of key operating parameters. You can use metrics to evaluate performance of traffic in the trace stream.

You can view metrics information in the trace display, the Traffic Summary window, and the Bus Utilization window.

PE*Tracer* defines different metrics for Link and Split transactions.

Metrics Defined for Link Transactions

Number of Packets: The total number of packets that compose this Link transaction.

Payload: The number of data payload bytes this Link transaction transferred.

Response Time: The time it took to transmit this Link transaction on the PE link, from the beginning of the first packet in the transaction to the end of the last packet in the transaction.

Data Throughput: The payload divided by response time, expressed in megabytes per second.

Note: Usually, the Number of Packets metric for a link transaction is two (in case of explicit acknowledge) or one (in case of implicit acknowledge). However if Naks/link level retries are involved, this metric might be higher. As a result, the Number of Packets metric is useful in highlighting unusual link transactions.

Metrics Defined for Split Transactions

The following types of metrics currently are defined for Split transactions:

Number of Link Transactions: The total number of Link transactions that compose this Split transaction.

Payload: The number of payload bytes this Split transaction transferred.

Response Time: The time it took to transmit this Split transaction on the PE link, from the beginning of the first packet in the Split transaction to the end of the last packet in the Split transaction;

Data Throughput: The transaction payload divided by response time, expressed in megabytes per second.

Latency Time: The time measured from the end of the request transaction to the first completion transmitted in response to the request within this Split transaction.

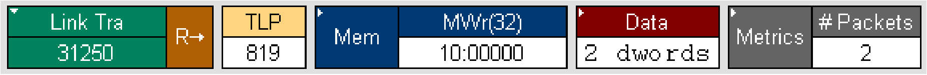
Note: The Number Of Link Transactions metric for a Split Transaction usually is two for a Configuration or IO request. It can be bigger then two for a Memory Read request.

9.8 Show Metrics in the Trace Display

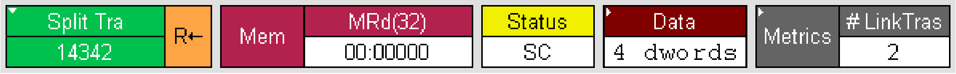
In the trace view display of Link Transaction or Split Transaction levels, all metrics information applicable to a specific protocol unit is displayed in a Metrics header. The header is located close to the end of the unit, in front of the Time cell. You can expand and collapse the header to show or hide the metrics information.

Collapsed Metrics Header Display

Following is a collapsed metrics display for a unit of Link Transaction. The collapsed header display shows only the metrics cell representing the Number of Packets.

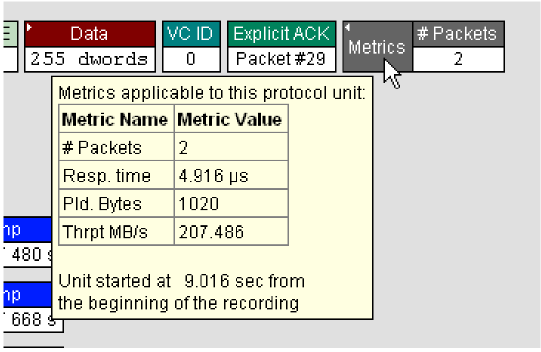


Following is a collapsed metrics display for a unit of Split Transaction. The collapsed header display shows only the metrics cell representing Number of Link Transactions.



Metrics Tool Tip Display

In both trace level views, you can view the summary of all the unit metrics in the Metrics header tool tip. To view the tool tip, simply place the mouse cursor over the Metrics header. Following is the tool tip for a unit in the Link Transaction view.



Metrics applicable to this protocol unit:

Metric Name	Metric Value
# Packets	2
Resp. time	4.916 µs
Pld. Bytes	1020
Thrpt MB/s	207.486

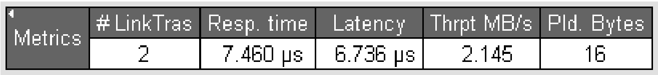
Unit started at 9.016 sec from the beginning of the recording

Note: The tool tip also presents information for time passed from the start of the recording till the beginning of this Link or Split Transaction.

Expanded Metrics Header Display

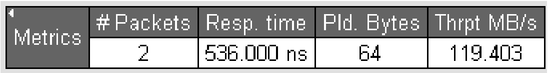
When you expand the Metrics header, the display creates a separate cell for each applicable metric:

Following is the expanded Metric header for a unit in the Split Transaction view.



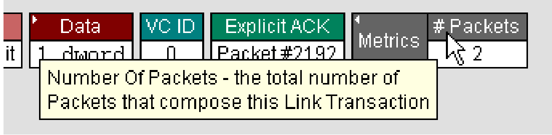
Metrics	# LinkTras	Resp. time	Latency	Thrpt MB/s	Pld. Bytes
	2	7.460 µs	6.736 µs	2.145	16

Following is the expanded Metric header for a unit in the Link Transaction view.



Metrics	# Packets	Resp. time	Pld. Bytes	Thrpt MB/s
	2	536.000 ns	64	119.403

Note: Each of the metric cells pops up a tool tip window with the explanation of what the metric means.



	Data	VC ID	Explicit ACK	Metrics	# Packets
it	1 dword	0	Packet #2192		2

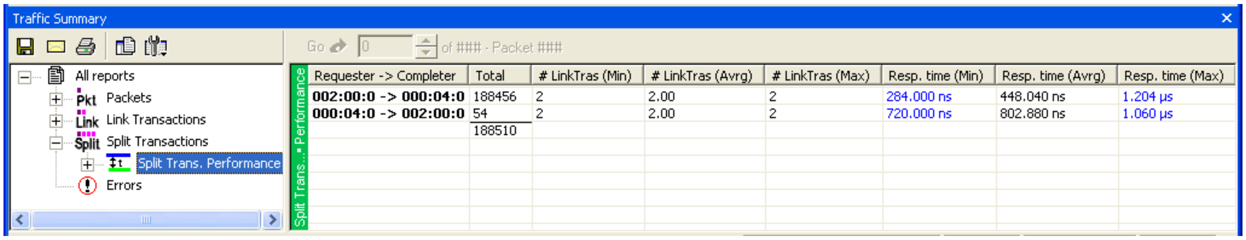
Number Of Packets - the total number of Packets that compose this Link Transaction

9.9 Show Metrics in the Traffic Summary Window

Some of the Traffic Summary reports at the Link and Split Transaction levels are based on metrics collected for the corresponding protocol units in the trace.

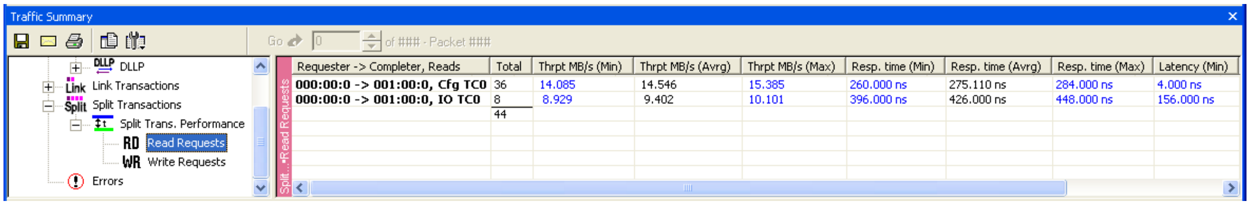
Reports at Split Transaction Level

Split Transaction Performance: This report table groups the Split Transactions by Requester-Completer pair and displays Minimum/Average/Maximum data for Number Of Link Transactions and Response Time metrics.



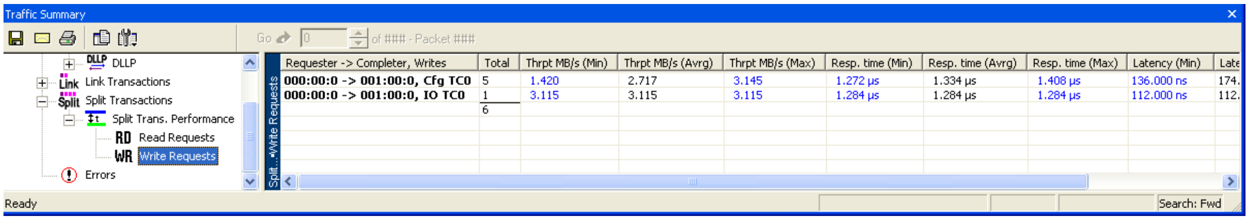
Requester -> Completer	Total	# LinkTras (Min)	# LinkTras (Avrg)	# LinkTras (Max)	Resp. time (Min)	Resp. time (Avrg)	Resp. time (Max)
002:00:0 -> 000:04:0	188456	2	2.00	2	284.000 ns	448.040 ns	1.204 µs
000:04:0 -> 002:00:0	54	2	2.00	2	720.000 ns	802.880 ns	1.060 µs
	188510						

Read Requests Performance: This report table includes only the Split Transactions that present Read Requests (Configuration, IO and Memory). It groups them by the combination of Requester-Completer pair, request type, and Traffic Class and displays Minimum/Average/Maximum data for Throughput, Response Time, and Latency metrics.



Requester -> Completer, Reads	Total	Thrpt MB/s (Min)	Thrpt MB/s (Avrg)	Thrpt MB/s (Max)	Resp. time (Min)	Resp. time (Avrg)	Resp. time (Max)	Latency (Min)
000:00:0 -> 001:00:0, Cfg TC0	36	14.085	14.546	15.385	260.000 ns	275.110 ns	284.000 ns	4.000 ns
000:00:0 -> 001:00:0, IO TC0	8	8.929	9.402	10.101	396.000 ns	426.000 ns	448.000 ns	156.000 ns
	44							

Write Requests Performance: This report table includes only the Split Transactions that present Write Requests (Configuration and IO). It groups them by the combination of Requester-Completer pair, request type, and Traffic Class and displays Minimum/Average/Maximum data for Throughput, Response Time, and Latency metrics.

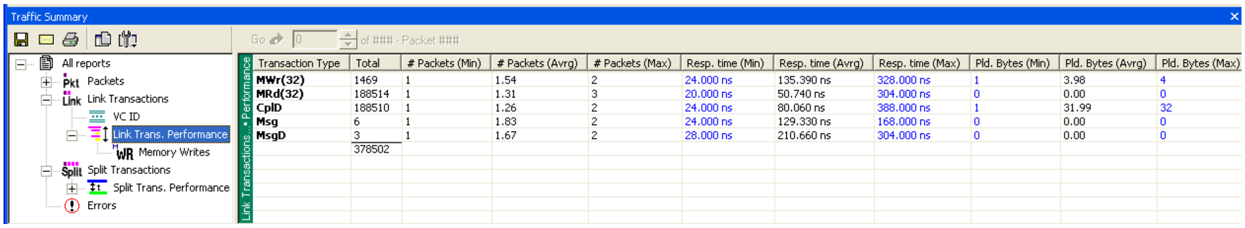


Requester -> Completer, Writes	Total	Thrpt MB/s (Min)	Thrpt MB/s (Avrg)	Thrpt MB/s (Max)	Resp. time (Min)	Resp. time (Avrg)	Resp. time (Max)	Latency (Min)	Late
000:00:0 -> 001:00:0, Cfg TC0	5	1.420	2.717	3.145	1.272 µs	1.334 µs	1.408 µs	136.000 ns	174.
000:00:0 -> 001:00:0, IO TC0	1	3.115	3.115	3.115	1.284 µs	1.284 µs	1.284 µs	112.000 ns	112.
	6								

Reports at the Link Transaction Level

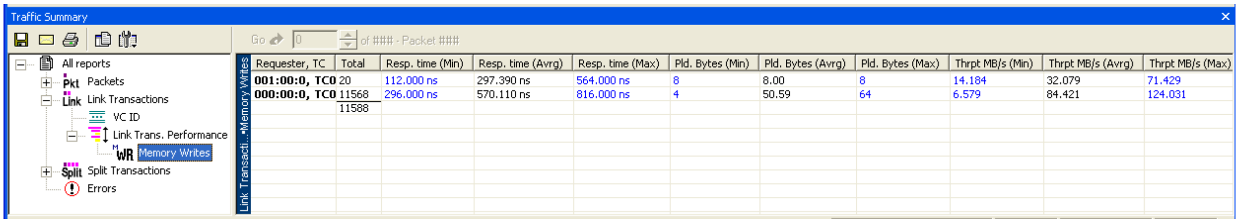
The following metric-based reports are displayed at the Link Transaction level:

Link Transaction Performance: This report table groups the Link Transactions by TLP Type and displays Minimum/Average/Maximum data for Number Of Packets, Response Time and Payload Bytes metrics.



Transaction Type	Total	# Packets (Min)	# Packets (Avrg)	# Packets (Max)	Resp. time (Min)	Resp. time (Avrg)	Resp. time (Max)	Pld. Bytes (Min)	Pld. Bytes (Avrg)	Pld. Bytes (Max)
MWr(32)	1469	1	1.54	2	24.000 ns	135.390 ns	328.000 ns	1	3.98	4
MRd(32)	188514	1	1.31	3	20.000 ns	50.740 ns	304.000 ns	0	0.00	0
CplD	188510	1	1.26	2	24.000 ns	80.060 ns	388.000 ns	1	31.99	32
Msg	6	1	1.83	2	24.000 ns	129.330 ns	168.000 ns	0	0.00	0
MsgD	3	1	1.67	2	28.000 ns	210.660 ns	304.000 ns	0	0.00	0
	378502									

Memory Writes: This report table includes only the Link Transactions that present Memory Write Requests. Memory Writes are the only (posted) requests that don't get promoted to the Split transaction level, therefore their performance should be viewed at the Link transaction level. The table groups Memory Writes by the combination of Requester ID and Traffic Class and displays Minimum/Average/Maximum data for Response Time, Payload and Throughput metrics.



Requester, TC	Total	Resp. time (Min)	Resp. time (Avrg)	Resp. time (Max)	Pld. Bytes (Min)	Pld. Bytes (Avrg)	Pld. Bytes (Max)	Thrpt MB/s (Min)	Thrpt MB/s (Avrg)	Thrpt MB/s (Max)
001:00:0, TC0	20	112.000 ns	297.390 ns	564.000 ns	8	8.00	8	14.184	32.079	71.429
000:00:0, TC0	11568	296.000 ns	570.110 ns	816.000 ns	4	50.59	64	6.579	84.421	124.031
	11588									

Following features apply to all of the report tables described above for the Split Transaction and Link Transaction levels:

- Each report row for the defined tables contains the total number of units in this group for the trace (total number of units for Split Transaction performance, Read Requests performance, Memory Writes, and so on).
- In many cases, the Maximum and Minimum values in the report tables are navigable. By clicking table cells, you reposition the corresponding trace view to the Split or Link Transaction that has yielded this maximum or minimum value. This can help you to find specific units in the trace, such as transactions that produced spikes in Response Time or Throughput.

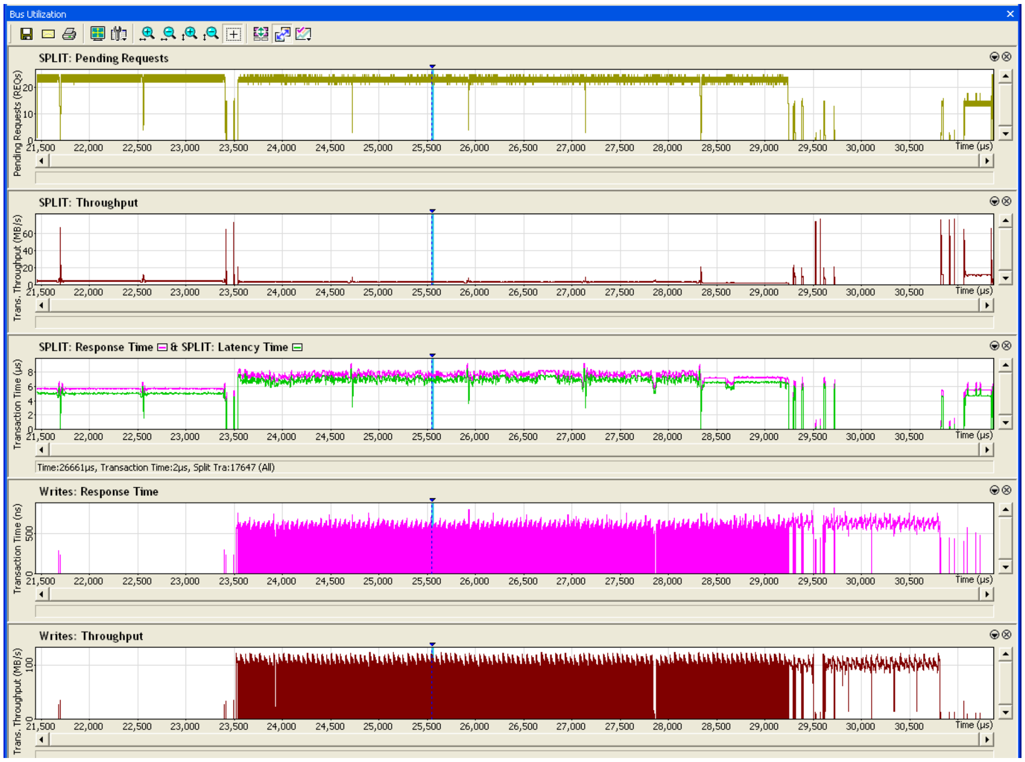
9.10 Show Metrics in the Bus Utilization Window

The Bus Utilization window provides graphs for packet-level information in the trace. The window also provides graphs for information on Split and Link Transaction levels, plotted over time. The graphs for Split and Link Transaction levels are based on metrics collected for the transactions throughout the trace.

The seven graphs related to Split and Transaction levels are:

1. Pending Requests at Split Transaction level.
2. Response Time at Split Transaction level.
3. Latency Time at Split Transaction level.
4. Throughput Per Transaction at Split Transaction level.
5. Response Time and Latency Time at Split Transaction level (combined graph).
6. Response Time at Link Transaction level, for Memory Writes only.
7. Throughput at Link Transaction level, for Memory Writes only.

The following is an example of Bus Utilization window display of graph numbers 1, 4, 5, 6 and 7:



Time:26661µs, Transaction Time:2µs, Split Tra:17647 (All)

Unit-Based Averaging

The Analyzer builds metric graphs using unit-based averaging (as opposed to time-based averaging). For the total duration of a certain request (or Memory Write transaction), the graph value is assumed equal to the corresponding metric for this request (transaction). If there are overlapping operations for a certain time period, then the value is calculated as an average of metric values for all the overlapped requests (transactions).

It is important to remember that the Analyzer uses unit-based averaging rather than time-based averaging. Time-based averaging can be misleading in some situations. For example, consider the Throughput Per Transaction graph. Sometimes, while many outstanding requests are in progress, latency (and response time) grows for each of the transactions, resulting in a lower throughput per transaction over time (which is reflected in the graph). This happens even though aggregated throughput across all the transactions is constant.

Bus Utilization Window Features

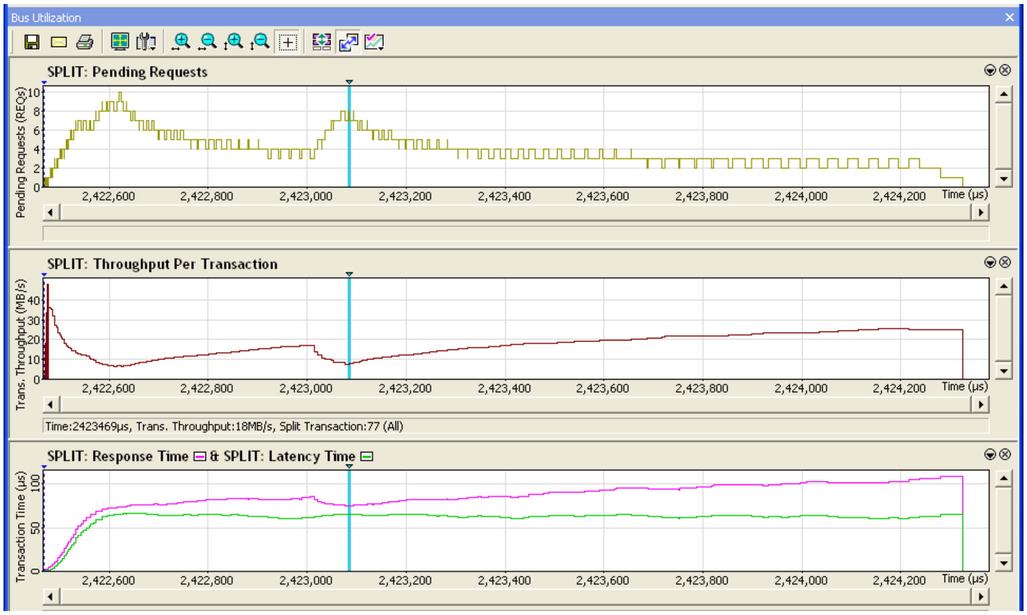
For the seven Split- and Transaction-level graphs listed, all Bus Utilization window features are available, such as zooming in/out, changing scale type, scrolling, context-sensitive status, and graph synchronization. See Bus Utilization and Bus Utilization Buttons for more on these features.

Note: Clicking a certain place within a graph area repositions the trace display at the Link or Split transaction level to the transaction that was in progress at that time.

Split Transaction Level Graphs

Transactions at the Split level combine all the non-posted requests with corresponding completions. This includes Configuration and IO Read and Write requests, as well as Memory Read requests.

The following shows the graphs for the Split level:

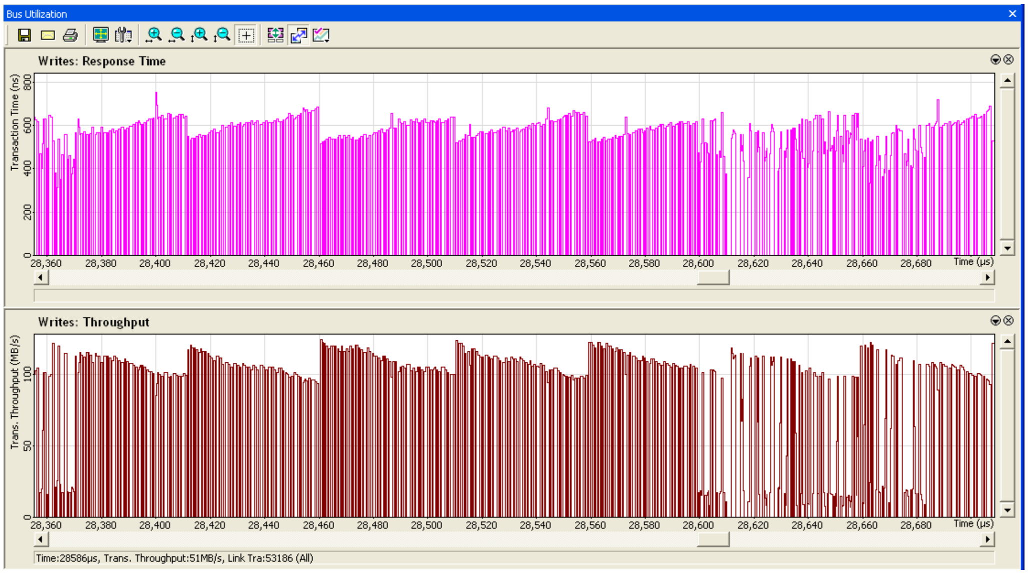


Time:2423469µs, Trans. Throughput:18MB/s, Split Transaction:77 (All)

Note: The Pending Requests graph is not directly defined by metrics, but it is useful when considering metrics. The Pending Requests graph presents the unit-averaged number of requests that were pending (in progress) at any moment of time. It allows you to correlate the number of requests posted to a completer with other performance metrics.

Link Transaction Level Graphs

Memory Writes are the only (posted) requests that do not get promoted to the Split transaction level. Therefore, Memory Write performance should be viewed at the Link Transaction level. That is the reason graphs at the Link level only present the Memory Write-related metrics and are titled **Writes: Response Time:** and **Writes: Throughput**.

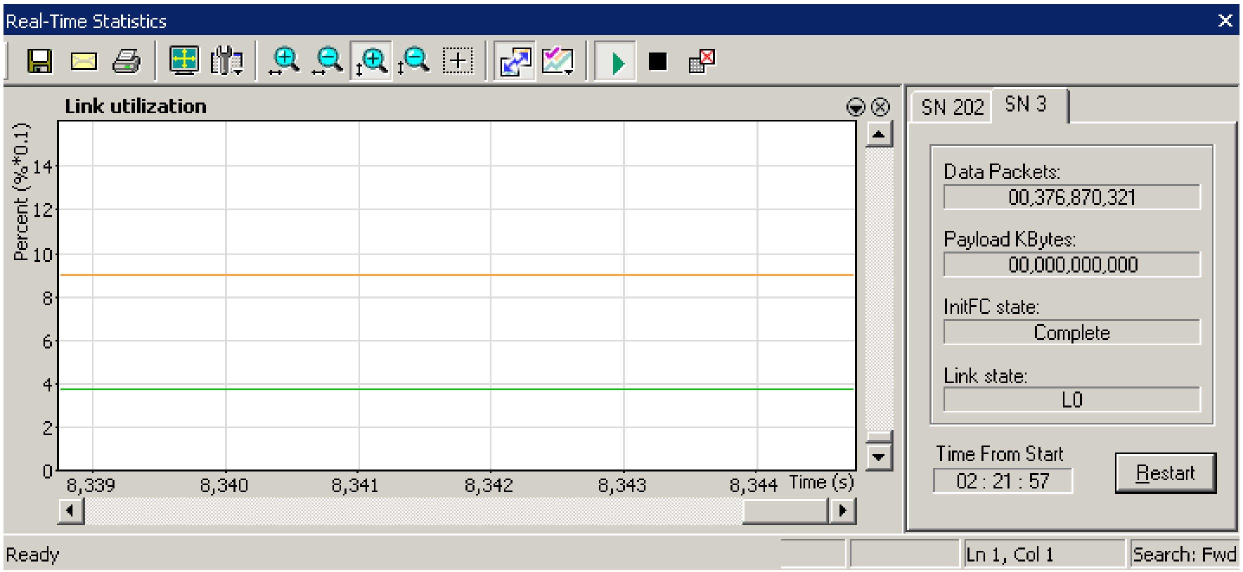


Time:28586µs, Trans. Throughput:51MB/s, Link Tra:53186 (All)

9.11 Real-Time Statistics Window

The Real-Time Statistics Window displays Link Utilization, performance measurements, and statistical values for a PCI Express link plotted in real time.

Click RT to open the Real-Time Statistics window.



Data Packets: 00,376,870,321

Payload KBytes: 00,000,000,000

InitFC state: Complete

Link state: L0

Time From Start 02 : 21 : 57

Start **PCI Express**™ link activity.

To start the monitor, press ▶ .

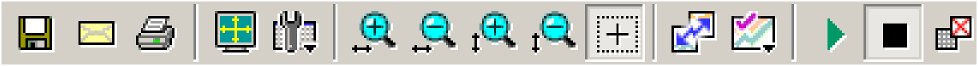
To stop the monitor, press ■ .

The remaining Real-Time Statistics buttons provide options for changing the format of the display.

Additional formatting options are available through the Real-Time Statistics pop-up menu and the Real-Time Statistics toolbar.

Real-Time Statistics Buttons

Additional formatting options are available through the Real-Time Statistics toolbar.



Button	Function
💾	Save. Saves Real-Time Statistics data to bitmap file (*.bmp).
✉	Email. Opens an email and attaches a bitmap file of the Real-Time Statistics data.
🖨	Print. Prints the Real-Time Statistics data.
🖥	Full Screen. Maximizes the Real-Time Statistics window.
🛠	View Settings. Opens a sub-menu with the following choices: • Orient Horizontally • Tile Vertically • Show Markers • Show Plumbline • Status >> Bar Tool tips None • Grid Lines >> Both Axes X Axis Y Axis No Grid • Grid on Top • Fonts & Colors
🔍	Horizontal zoom in
🔍	Horizontal zoom out

Vertical zoom in

Vertical zoom out

Click and Drag Zoom. Click and drag to zoom in on a part of the graph.

Synchronize Graph Areas. If two or more graphs are displayed, this button synchronizes the graphs to one another. Once synchronized, the positioning slider of one graph moves the other graphs.

Graph Areas. Allows you to hide or display the graphs and the counters. You can toggle Statistics Accumulation, Link Utilization, Data Payload Throughput, and Data Packet Count.

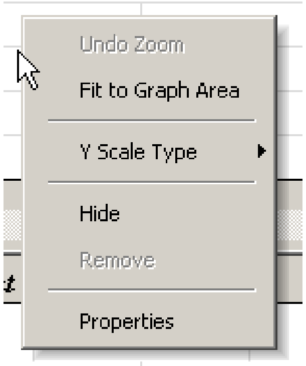
Start real-time statistics. Starts the real-time statistical monitor.

Stop real-time statistics. Stops the real-time statistical monitor.

Reset Graphs. Resets the graphs.

9.12 Real-Time Statistical Monitor Pop-up Menu

Additional formatting options are available through the Real-Time Statistics pop-up menu. Right-click a graph in the Real-Time Statistical Monitor window to display a pop-up menu with options for changing the format of the display.



Undo Zoom: If you have zoomed in, this command undoes the zoom.

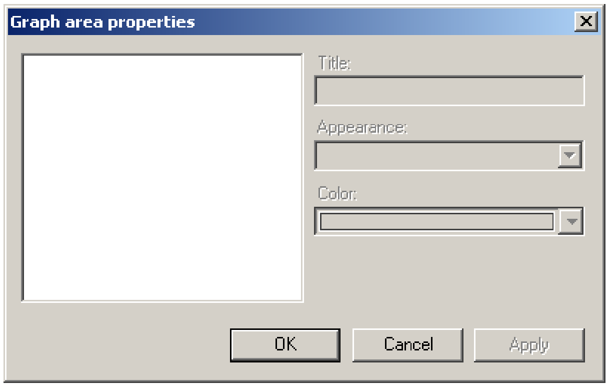
Fit to Graph Area: Redisplays graph so that the entire trace fits inside graph area.

Y Scale Type:

- **Linear**: Converts display to linear format.
- **Logarithmic**: Converts display to logarithmic format.

Hide: Hides the selected graph

Properties: Opens a dialog box with options for changing the Title, Appearance, and Color of the graph.



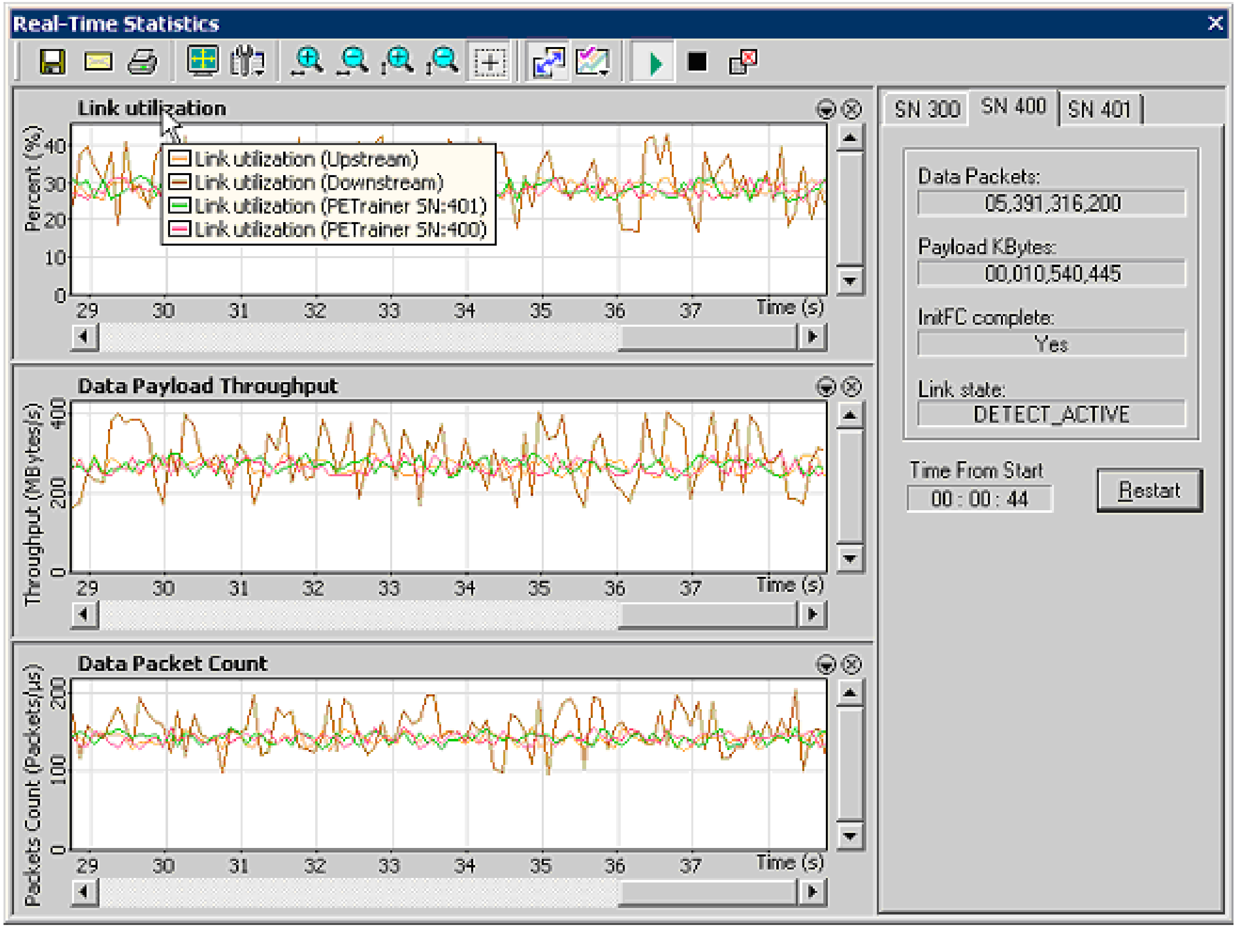
9.13 Real-Time Statistics Graph Areas

The Real-time Statistics window has three graph areas:

- Link Utilization
- Data Payload Throughput
- Data Packet Count

Each graph area has Upstream and Downstream graphs for a PE*Tracer* device (if connected to a PC), plus as many graphs as PE*Trainer*™ devices connected to the PC.

The Statistics Accumulation area shows the PE*Tracer* statistics tab, plus as many tabs as PE*Trainer* devices connected:



Data Packets: 05,391,316,200

Payload KBytes: 00,010,540,445

InitFC complete: Yes

Link state: DETECT_ACTIVE

Time From Start 00 : 00 : 44

For each PE*Trainer* device the following information displayed:

- **Number of data packets**
- **Payload size**
- **InitFC complete status**: Yes/No
- **Link State**: Possible states are:
    - DETECT_QUIET
    - DETECT_ACTIVE
    - POLLING_ACTIVE
    - POLLING_CONFIG
    - CONFIG_LINKWIDTH_START
    - CONFIG_LINKWIDTH_ACCEPT
    - CONFIG_LANENUM_WAIT
    - CONFIG_LANENUM_ACCEPT
    - CONFIG_COMPLETE
    - CONFIG_IDLE
    - L0
    - L0S_IDLE
    - L0S_FTS
    - L1
    - RECOVERY_RCVRLOCK
    - RECOVERY_RCVRCFG
    - RECOVERY_IDLE
    - HOT_RESET
    - DISABLED

Section 3. PE*Trainer* Exerciser Traffic Generation

Chapter 10: Traffic Generation Overview

After the PE*Trainer*™ ML™ or PE*Trainer* EML™ and DUT have been cabled and powered on, you can test the setup by generating some traffic. The following steps show how to configure the Exerciser to generate a Link Training sequence.

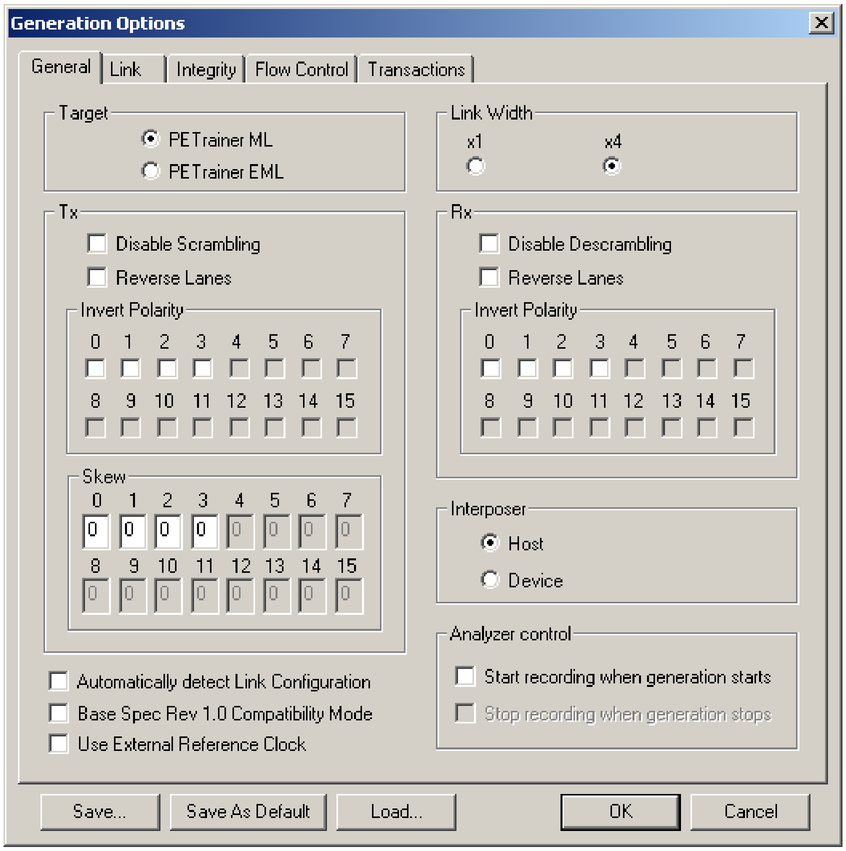
Note: This chapter describes the default values for the important generation options, to introduce you to traffic generation. The chapters following this chapter provide detailed descriptions.

10.1 Set the Generation Options

Before beginning generation, set options in the Generation Options dialog box:

Step 1 Open the Generation Options dialog box by selecting **Setup** > **Generation Options** from the menu. The Generation Options dialog opens.

For **PE*Trainer* ML**, by default the General page displays.



Target: ● PETrainer ML ○ PETrainer EML

Link Width: x1 ○ x4 ●

Tx

□ Disable Scrambling

□ Reverse Lanes

Invert Polarity

0	1	2	3	4	5	6	7
□	□	□	□	□	□	□	□
8	9	10	11	12	13	14	15
□	□	□	□	□	□	□	□

Skew

0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0
8	9	10	11	12	13	14	15
0	0	0	0	0	0	0	0

Rx

□ Disable Descrambling

□ Reverse Lanes

Invert Polarity

0	1	2	3	4	5	6	7
□	□	□	□	□	□	□	□
8	9	10	11	12	13	14	15
□	□	□	□	□	□	□	□

Interposer: ● Host ○ Device

Analyzer control

□ Start recording when generation starts

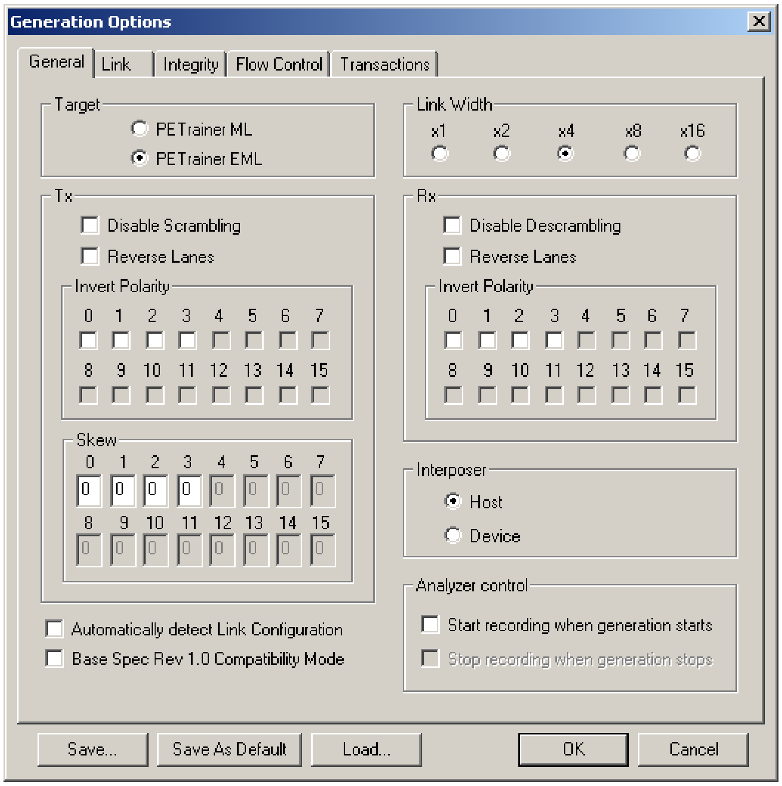
□ Stop recording when generation stops

□ Automatically detect Link Configuration

□ Base Spec Rev 1.0 Compatibility Mode

□ Use External Reference Clock

For **PE*Trainer* EML**, by default the General page displays.



Generation Options

General | Link | Integrity | Flow Control | Transactions

Target
○ PETrainer ML
● PETrainer EML

Link Width
x1 ○ x2 ○ x4 ● x8 ○ x16 ○

Tx
□ Disable Scrambling
□ Reverse Lanes

Invert Polarity

0	1	2	3	4	5	6	7
□	□	□	□	□	□	□	□
8	9	10	11	12	13	14	15
□	□	□	□	□	□	□	□

Skew

0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0
8	9	10	11	12	13	14	15
0	0	0	0	0	0	0	0

Rx
□ Disable Descrambling
□ Reverse Lanes

Invert Polarity

0	1	2	3	4	5	6	7
□	□	□	□	□	□	□	□
8	9	10	11	12	13	14	15
□	□	□	□	□	□	□	□

Interposer
● Host
○ Device

Analyzer control
□ Start recording when generation starts
□ Stop recording when generation stops

□ Automatically detect Link Configuration
□ Base Spec Rev 1.0 Compatibility Mode

Step 2 To test 1.0 devices, check the box next to **Base Spec Rev. 1.0 Compatibility Mode** to select 1.0 compatibility mode.

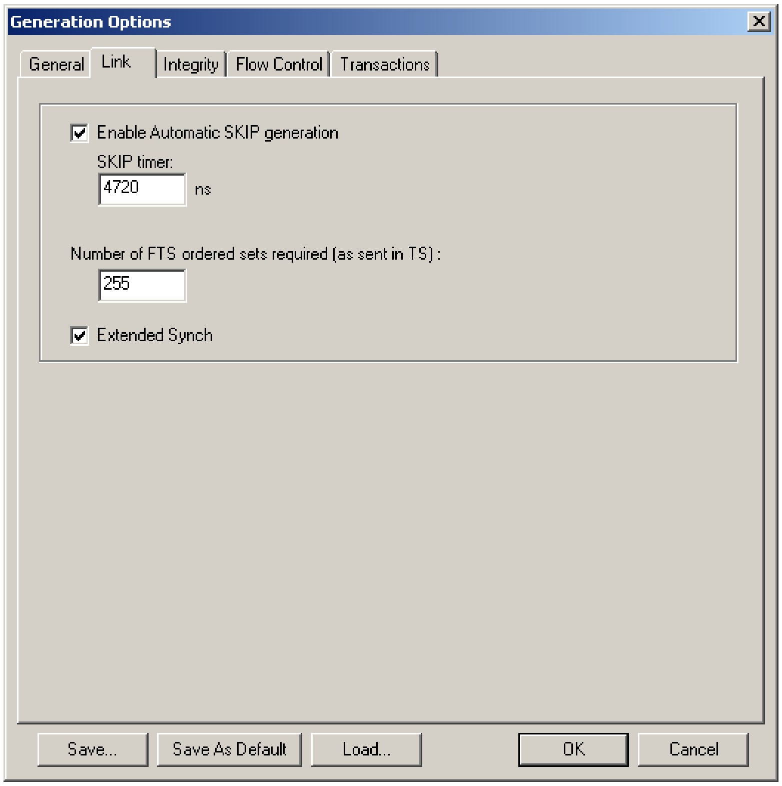
To test 1.0A devices, leave this box unchecked.

Step 3 Set the generation direction based on type of device you are emulating.

Step 4 Select one of the two **Interposer** options:

- **Host Emulation**: Select **Host**
- **Device Emulation**: Select **Device**

Step 5 Select the **Link** tab. The Link page opens.



☑ Enable Automatic SKIP generation

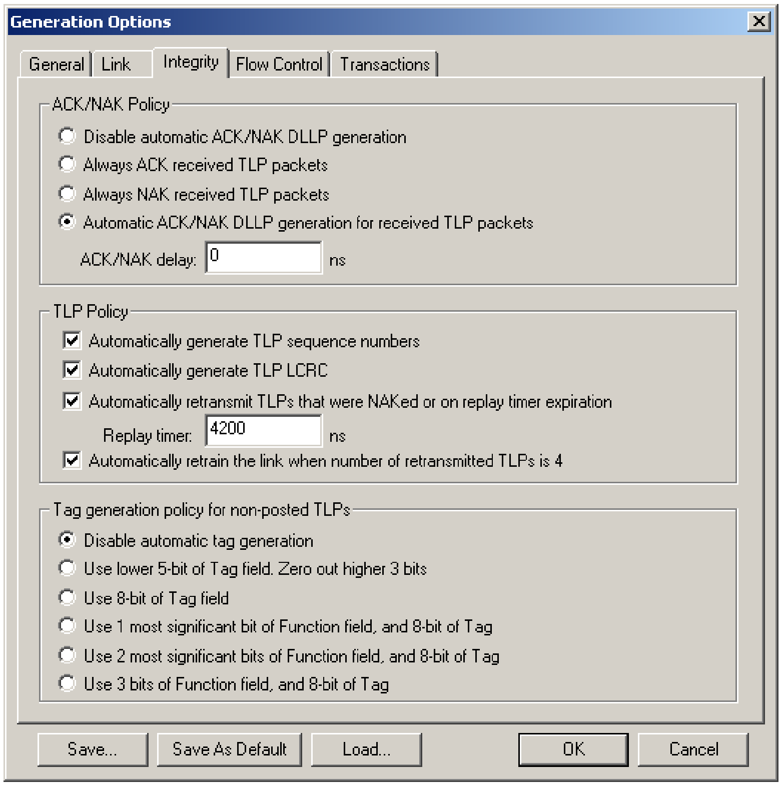
SKIP timer: 4720 ns

Number of FTS ordered sets required (as sent in TS) : 255

☑ Extended Synch

Step 6 Select **Enable Automatic Skip Generation,** but leave the setting at the default value: 4720 ns.

Step 7 Select the **Integrity** tab**.** The Integrity page opens.



ACK/NAK Policy

○ Disable automatic ACK/NAK DLLP generation
○ Always ACK received TLP packets
○ Always NAK received TLP packets
● Automatic ACK/NAK DLLP generation for received TLP packets

ACK/NAK delay: 0 ns

TLP Policy

☑ Automatically generate TLP sequence numbers
☑ Automatically generate TLP LCRC
☑ Automatically retransmit TLPs that were NAKed or on replay timer expiration

Replay timer: 4200 ns

☑ Automatically retrain the link when number of retransmitted TLPs is 4

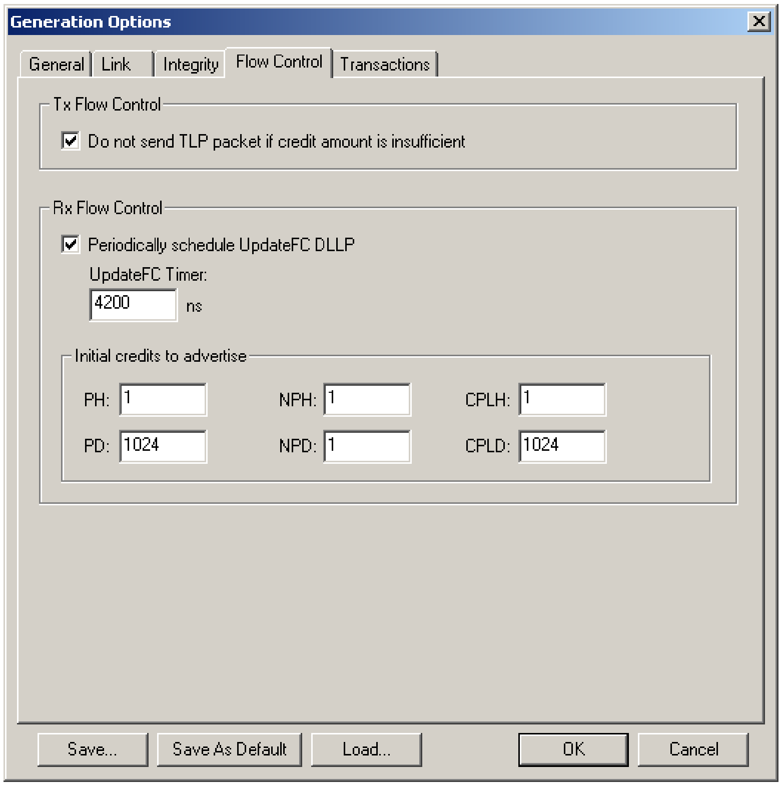
Tag generation policy for non-posted TLPs

● Disable automatic tag generation
○ Use lower 5-bit of Tag field. Zero out higher 3 bits
○ Use 8-bit of Tag field
○ Use 1 most significant bit of Function field, and 8-bit of Tag
○ Use 2 most significant bits of Function field, and 8-bit of Tag
○ Use 3 bits of Function field, and 8-bit of Tag

Step 8 Check to enable **Automatic ACK/NAK DLLP generation for received TLP packets**.

Step 9 Enable all four TLP policies and set **ACK/NAK Delay** = **0 ns**.

Step 10 Select the **Flow Control** tab. The Flow Control page opens**.**



Tx Flow Control

☑ Do not send TLP packet if credit amount is insufficient

Rx Flow Control

☑ Periodically schedule UpdateFC DLLP

UpdateFC Timer:

4200 ns

Initial credits to advertise

PH:	1	NPH:	1	CPLH:	1
PD:	1024	NPD:	1	CPLD:	1024

Step 11 Check the checkbox to enable **Do not send TLP packet if credit amount is insufficient.**

Step 12 Check the checkbox to enable **Periodically schedule UpdateFC DLLP**.

Step 13 Use the defaults for all other boxes.

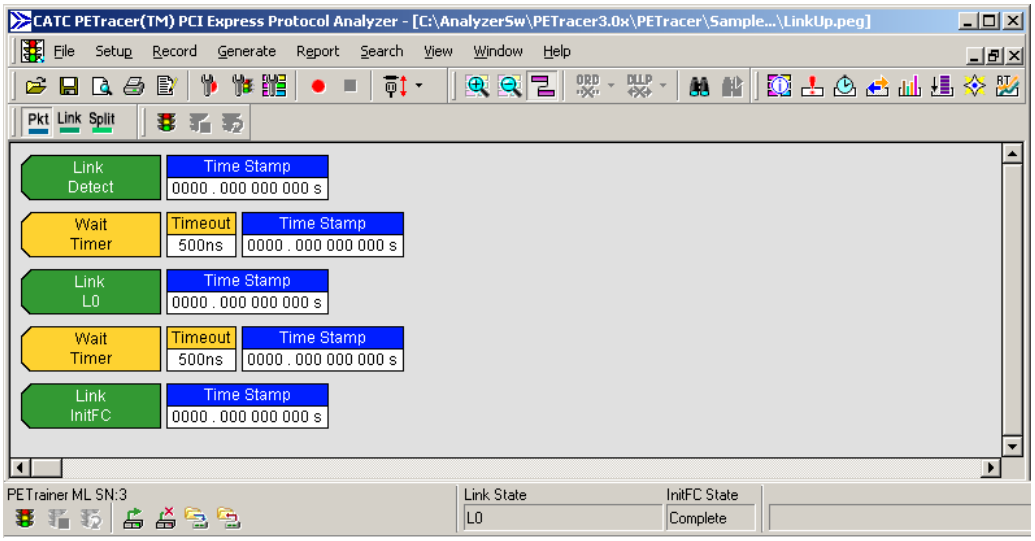
Step 14 Click **OK** to apply all changes and close the Generation Options dialog.

10.2 Preparing Traffic Generation

To prepare for traffic generation, follow these steps:

Step 1 If it is not running, start the PE*Tracer*™ software.

Step 2 Open the traffic generation file **Linkup.peg** by selecting **File** > **Open** from the menu. The following packets display in the main window.


CATC PETracer(TM) PCI Express Protocol Analyzer - [C:\AnalyzerSw\PETracer3.0x\PETracer\Sample...\LinkUp.peg]
File Setup Record Generate Report Search View Window Help
Pkt Link Split
PETrainer ML SN:3

Link Detect	Time Stamp 0000 . 000 000 000 s	
Wait Timer	Timeout 500ns	Time Stamp 0000 . 000 000 000 s
Link L0	Time Stamp 0000 . 000 000 000 s	
Wait Timer	Timeout 500ns	Time Stamp 0000 . 000 000 000 s
Link InitFC	Time Stamp 0000 . 000 000 000 s	

Link State: L0
InitFC State: Complete

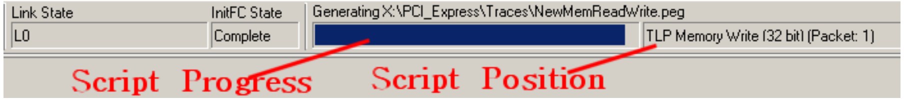
Step 3 The trace window shows the type of traffic that is to be generated in the current generation session.

10.3 Begin Traffic Generation

To begin traffic generation:

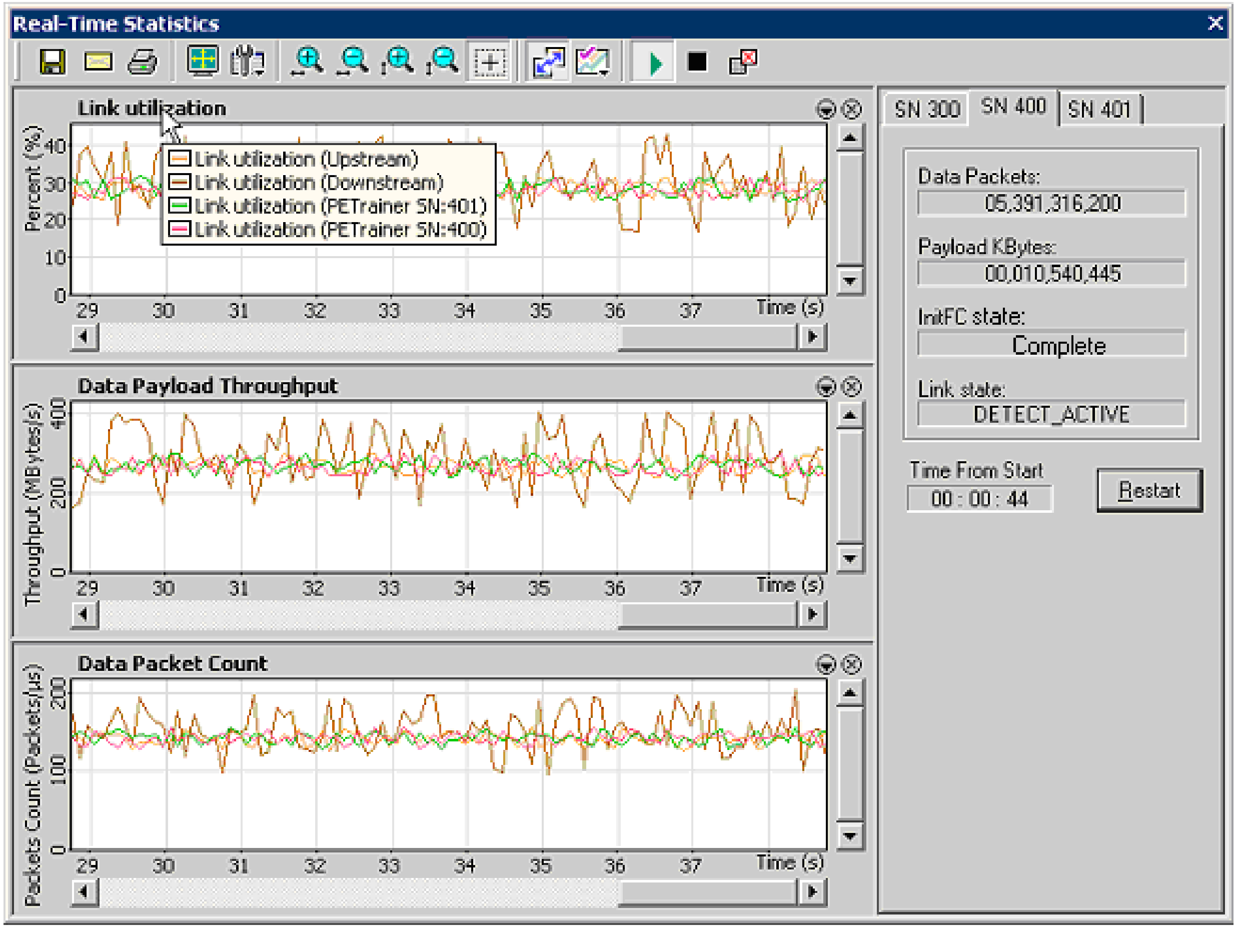
Step 4 Click 🚦 .

The PE*Tracer* software downloads the script to the device. After the download completes, the device starts executing the script. As the script executes, the Status bar displays the script's progress. The **Current script position** field displays the command description and trace packet number currently being generated.


Link State: L0
InitFC State: Complete
Generating X:\PCI_Express\Traces\NewMemReadWrite.peg
TLP Memory Write (32 bit) (Packet: 1)

Script Progress Script Position

Step 5 Open the Real-Time Statistics window by clicking .



Real-Time Statistics

SN 300 SN 400 SN 401

Data Packets: 05,391,316,200

Payload KBytes: 00,010,540,445

InitFC state: Complete

Link state: DETECT_ACTIVE

Time From Start 00 : 00 : 44

Restart

Link utilization
Percent (%)
40
30
20
10
0
Link utilization (Upstream)
Link utilization (Downstream)
Link utilization (PETrainer SN:401)
Link utilization (PETrainer SN:400)
29
30
31
32
33
34
35
36
37
Time (s)
Data Payload Throughput
Throughput (MBytes/s)
400
200
0
Data Packet Count
Packets Count (Packets/μs)
200
100
0

To the right is a **Statistics Accumulation** area. In this area look for two messages:

- **InitFC State:** Complete (shown in the example above)
- **Link State**: LO

If the messages appear, then it means that the Exerciser successfully completed the Link training.

Note: Link training can also be performed by clicking the **Trainer Connect** button on the Status bar.

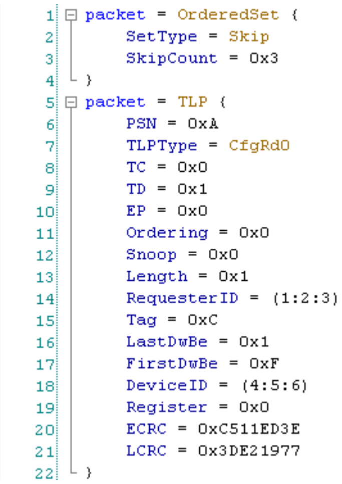
Chapter 11: Traffic Generation

11.1 Traffic Generation Overview

PE*Trainer*™ is a traffic generator that can emulate PCI Express™ root complexes and endpoint devices. Traffic generation can be used to transmit known errors, allowing you to observe how your device handles faulty link conditions.

Traffic Generation (*.peg) Files

The **.peg** traffic generation files are text files consisting of a series of commands from a scripting language (see Chapter 12), and optionally, one or more **Include** statements linking other generation files into the current file.

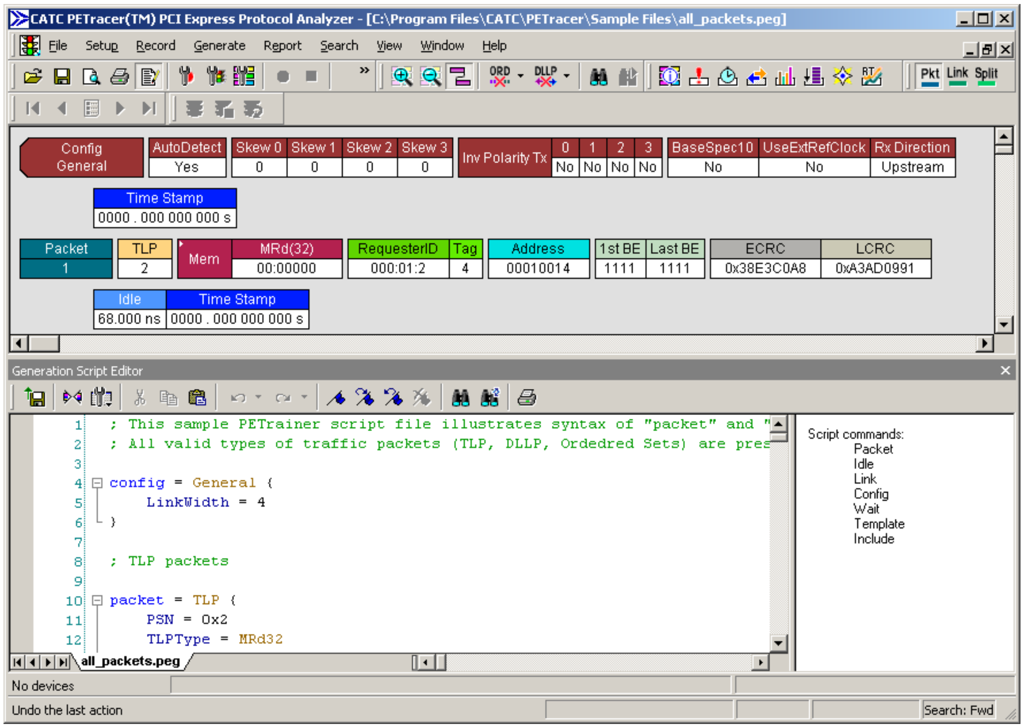


```
packet = OrderedSet {
    SetType = Skip
    SkipCount = 0x3
}
packet = TLP {
    PSN = 0xA
    TLPType = CfgRd0
    TC = 0x0
    TD = 0x1
    EP = 0x0
    Ordering = 0x0
    Snoop = 0x0
    Length = 0x1
    RequesterID = (1:2:3)
    Tag = 0xC
    LastDwBe = 0x1
    FirstDwBe = 0xF
    DeviceID = (4:5:6)
    Register = 0x0
    ECRC = 0xC511ED3E
    LCRC = 0x3DE21977
}
```

Note: To edit a generation file, use the Script Editor, a specially designed text editor tool. See "Editing Generation Files with the Script Editor" on page 215.

Generating Traffic

To generate traffic, open a traffic generation file (***.peg**) and then run it.



Config General	AutoDetect	Skew 0	Skew 1	Skew 2	Skew 3	Inv Polarity Tx 0	1	2	3	BaseSpec10	UseExtRefClock	Rx Direction
	Yes	0	0	0	0	No	No	No	No	No	No	Upstream

Time Stamp
0000 . 000 000 000 s

Packet	TLP	Mem	MRd(32)	RequesterID	Tag	Address	1st BE	Last BE	ECRC	LCRC
1	2		00:00000	000:01:2	4	00010014	1111	1111	0x38E3C0A8	0xA3AD0991

Idle	Time Stamp
68.000 ns	0000 . 000 000 000 s

```
; This sample PETrainer script file illustrates syntax of "packet" and "
; All valid types of traffic packets (TLP, DLLP, Ordedred Sets) are pres

config = General {
    LinkWidth = 4
}

; TLP packets

packet = TLP {
    PSN = 0x2
    TLPType = MRd32
```

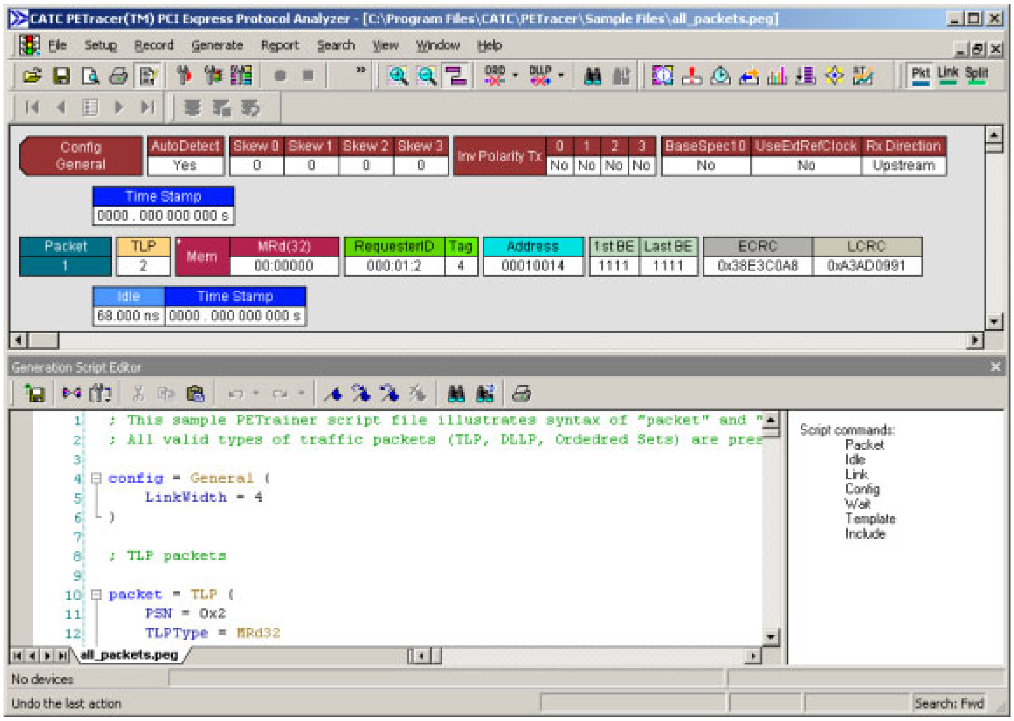
Script commands:
- Packet
- Idle
- Link
- Config
- Wait
- Template
- Include

When the file is opened, it appears in the trace window looking like a trace file.

To run the file, press .

Creating/Editing a Traffic Generator File

To create or edit a **.peg** file, use the Script Editor.



Config General	AutoDetect	Skew 0	Skew 1	Skew 2	Skew 3	Inv Polarity Tx 0	1	2	3	BaseSpec10	UseExtRefClock	Rx Direction
	Yes	0	0	0	0	No	No	No	No	No	No	Upstream

Time Stamp
0000 . 000 000 000 s

Packet	TLP	Mem	MRd(32)	RequesterID	Tag	Address	1st BE	Last BE	ECRC	LCRC
1	2		00:00000	000:01:2	4	00010014	1111	1111	0x38E3C0A8	0xA3AD0991

Idle	Time Stamp
68.000 ns	0000 . 000 000 000 s

```
; This sample PETrainer script file illustrates syntax of "packet" and "
; All valid types of traffic packets (TLP, DLLP, Ordedred Sets) are pres

config = General (
      LinkWidth = 4
)

; TLP packets

packet = TLP (
      PSN = 0x2
      TLPType = MRd32
```

Script commands:
Packet
Idle
Link
Config
Wait
Template
Include

The Script Editor is a text-editing tool that can be opened by clicking 📝 .

Alternatively, the traffic generation file can be created by exporting the data from a trace into a traffic generator file.

11.2 Traffic Generation Theory of Operation

Overview

PE*Trainer* offers two mechanisms for implementing traffic generation: scripts, in which any type of traffic can be defined and executed, and the Generation Options dialog box, which offers a collection of PCI Express specific **behaviors** that can be enabled for automatic generation of traffic.

Using scripts, packets can be transmitted one after another or with certain timing or event-based pauses between them. This allows the PE*Trainer* to act as a pattern generator with PCI Express-specific formatting and transmission rates. However, creating traffic that emulates real devices with relatively complex protocol behaviors using a simple pattern generator is quite complicated. Certain behaviors such as ACK policies, and flow control require concurrent processing. This is where the automated features of the PE*Trainer* ML™ and PE*Trainer* EML™ products become useful.

The PE*Trainer* includes a collection of automated traffic generation circuits that commonly exist in other PCI Express devices. These circuits include ACK/NAK generation, flow control management, a Link Training and Status State machine (LTSSM), replay buffers, and transaction timers. What makes the PE*Trainer* unique and so useful is that each of these behaviors can be individually modified or disabled. This allows the user to perform operations that might not otherwise be possible using an off the shelf PCI Express device. This can be particularly useful when doing compliance or fault recovery testing.

Starting Point

When the PE*Trainer* first powers up, it is at electrical idle on all lanes. The link is not trained, but the Link Training and Status State Machine (LTSSM) is enabled and waiting for a command to train the link. To begin communication with a PCI Express device, the Link training must occur, but first the generation settings must be set. This involves setting the Link parameters such as link width, polarity inversion, and lane reversal, through the Generation Options dialog.

Note: Setting the options in the Generation Options dialog has no effect on the PE*Trainer* behavior until the first script is executed. In fact, each time a script is executed, these behaviors are reprogrammed to the PE*Trainer* and the behaviors are modified accordingly.

The **Link Connect** toolbar button can then be pressed to signal to the LTSSM to initiate Link training. Alternatively, a script can executed with the **Link=L0** command.

The PE*Trainer* Status bar at the bottom of the screen shows the current state of the PCI Express Link. When the Link is down, it shows **Detect**. When it is up, it shows **L0**.

All of the intermediate link states, such as **Polling** and **Configuration**, are handled automatically. In fact, after the **Link=L0** command is executed and the LTSSM is alive, the PE*Trainer* attempts to keep the Link active just as any other PCI Express device does. This includes handling of Recovery states and subsequent retraining events.

The next step is to enable flow control. This is done by executing the **Link=InitFC** command. This causes PE*Trainer* to perform flow control initialization and, if enabled, begin periodic transmission of **Update_FC DLLP**'s. If the Link connect toolbar button was used, the flow control initialization happens automatically.

By using these basic commands, the complicated process of link training is managed automatically. The Link is now trained and the script execution can now focus on sending TLP packets.

Script Execution

As mentioned above, each time a script is executed, the configuration settings for the PE*Trainer* are reprogrammed. When the **Start generation** button is pressed, the script is uploaded to the PE*Trainer* hardware and executed immediately. Progress of the script can be tracked in the status bar at the bottom of the screen. Subsequent executions of the same script do not require upload of the script to the PE*Trainer*, however, as mentioned above, the generation options are still reprogrammed.

Script execution can be throttled using **Wait** commands inserted directly into the script. Waits can be time based, require receipt of certain packet types, or can even require **User** input from the GUI.

After the script is complete, any enabled PCI Express **behaviors,** such as SKIP insertion, Flow control, and ACK generation, continue. This allows you to run multiple scripts, one after another, without interrupting the Link state.

11.3 Creating a Traffic Generation File

There are four ways to create PE*Trainer*™ traffic generation script file:

- **Export an existing PE*Tracer*™ trace** to a PE*Trainer* script file
- **Save an existing PE*Trainer* script** to a new file
- **Select File > New** to create a new and empty script file that contains no text.
- **Create an empty file** using an OS shell (with **.peg** extension) and open it with PE*Tracer* software.

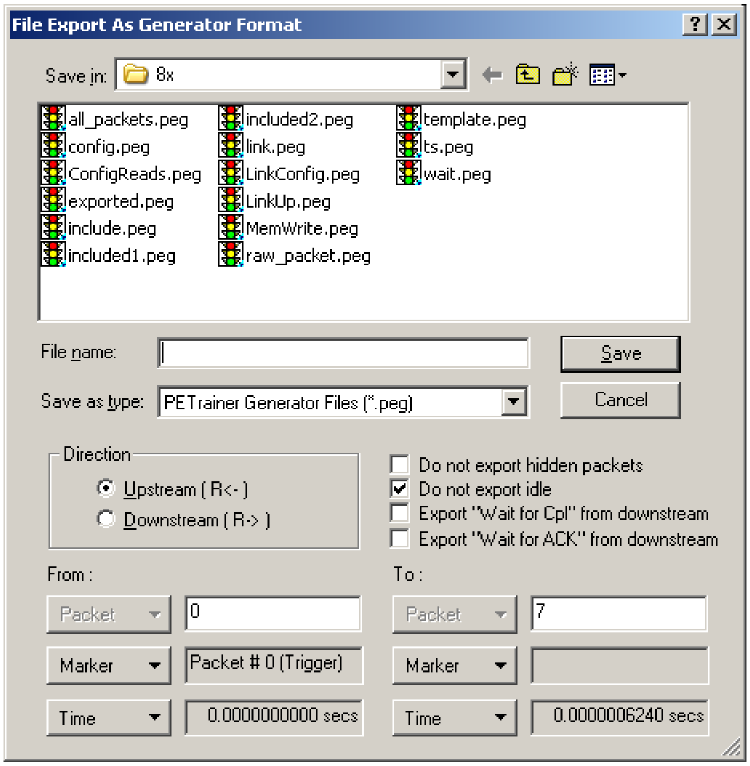
Exporting a Trace to a Traffic Generation File

A simple way to create a script file is to open a trace and then to export the trace data to a generation file:

Step 1 Open a trace file.

Step 2 Select **File > Export > to Generator File Format**.

Step 3 Select the desired options from the File Export dialog box:



File Export As Generator Format

Save in: 8x

all_packets.peg
config.peg
ConfigReads.peg
exported.peg
include.peg
included1.peg
included2.peg
link.peg
LinkConfig.peg
LinkUp.peg
MemWrite.peg
raw_packet.peg
template.peg
ts.peg
wait.peg

File name:

Save as type: PETrainer Generator Files (*.peg)

Direction
● Upstream (R<-)
○ Downstream (R->)

□ Do not export hidden packets
☑ Do not export idle
□ Export "Wait for Cpl" from downstream
□ Export "Wait for ACK" from downstream

From :
Packet 0
Marker Packet # 0 (Trigger)
Time 0.0000000000 secs

To :
Packet 7
Marker
Time 0.0000006240 secs

Direction: Selects the direction of the traffic to be exported.

From and To: Selects a range for exporting. You can export all or part of the trace.

Do not export hidden packets: Ignores any packets hidden through the various hide options.

Do not export Idle packets: Excludes Idles from the export.

Export "Wait for Cpl" from upstream: Exports all **Wait for Completions** from the opposite direction. **Wait = TLP** is a PE*Trainer* script command (see “Wait Command” on page 273 for further details).

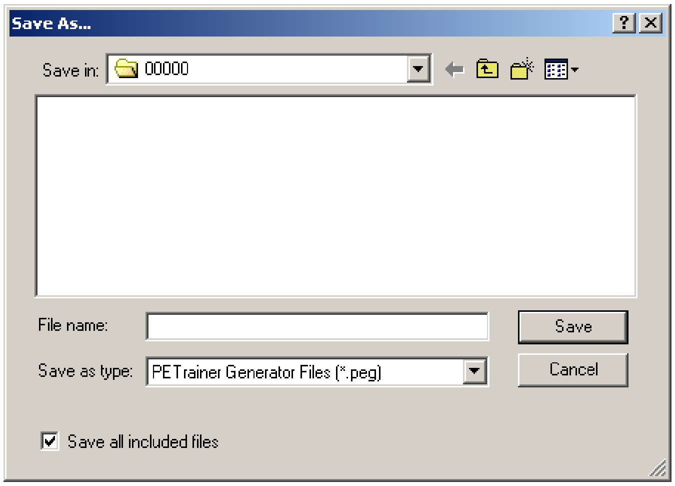
Export "Wait for ACK" from upstream: Exports all **Wait for ACK DLLPs** from the opposite direction.

Saving a PE*Trainer* Script to a New File

To save a script file as a generation file:

Step 1 Open an existing PE*Trainer* script file.

Step 2 Select **File > Save As**… or click 💾 on the toolbar.


Save As...

Save in: 00000

File name:

Save as type: PETrainer Generator Files (*.peg)

☑ Save all included files

Step 3 Navigate to the desired folder and type a new file name where you want to save the current script, then click the **Save** button:

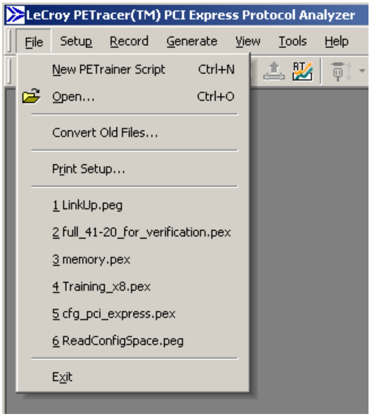
When the Save As command completes, the newly created script is displayed.

Note: The Save As dialog box includes the option **Save all included files**. This option causes the software to save any **Include** files that have included into the script. The Included files are saved to the same directory as the traffic generation file. For details, see “Include Command” on page 288.

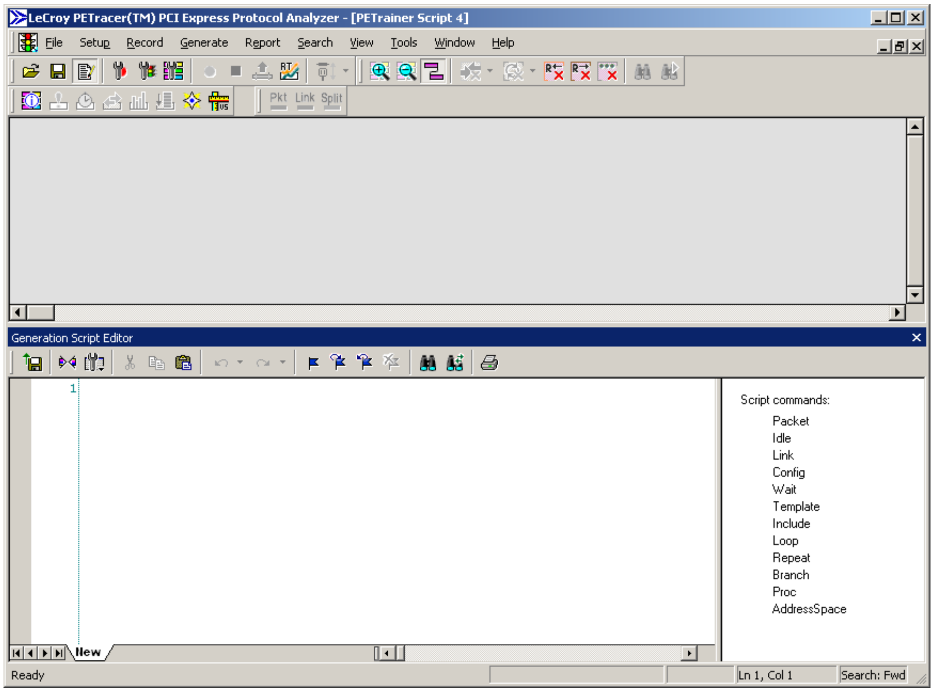
Creating a New Empty Generation File

To create a generation file from scratch:

Step 1 Select **File > New PE*Trainer* Script** or use the shortcut **Ctrl+N**.



An empty PE*Trainer* traffic generation file appears. You can also view the Script Editor..

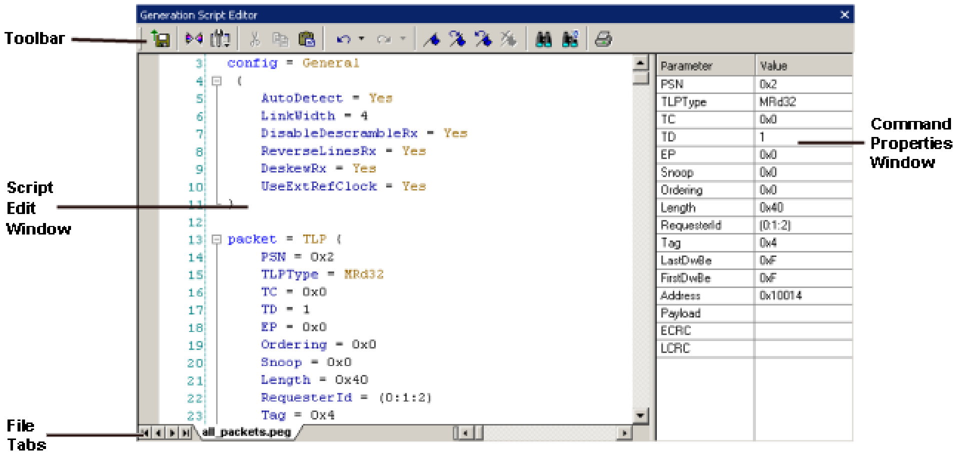


11.4 Editing Generation Files with the Script Editor

The Script Editor is an editing tool for PE*Tracer* traffic generation files (**<filename>.peg**). The generation script is presented in the Script Edit window. Parameters are presented in menus and text boxes in the Command Properties window in the right-hand portion of the Script Editor.

To launch the Script Editor: click the **Script Editor** button on the toolbar or right-click the trace window and choose **Edit as Text** from the pop-up menu.

The Script Editor window displays in the lower portion of the trace window.



```
config = General
 (
     AutoDetect = Yes
     LinkWidth = 4
     DisableDescrambleRx = Yes
     ReverseLinesRx = Yes
     DeskewRx = Yes
     UseExtRefClock = Yes
 )

packet = TLP (
     PSN = 0x2
     TLPType = MRd32
     TC = 0x0
     TD = 1
     EP = 0x0
     Ordering = 0x0
     Snoop = 0x0
     Length = 0x40
     RequesterId = (0:1:2)
     Tag = 0x4
```

Parameter	Value
PSN	0x2
TLPType	MRd32
TC	0x0
TD	1
EP	0x0
Snoop	0x0
Ordering	0x0
Length	0x40
RequesterId	(0:1:2)
Tag	0x4
LastDwBe	0xF
FirstDwBe	0xF
Address	0x10014
Payload	
ECRC	
LCRC	

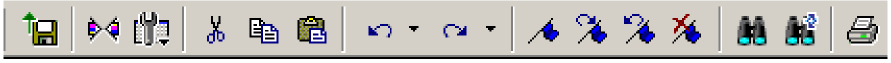
Layout: The Script Editor divides into four areas:

- **Script Editor toolbar**: Presents options for printing, saving, bookmarking, and other options.
- **Script Edit window**: Main window where the script is displayed and edited. Text in this window behaves as in most text editors. Text can be copied, pasted, and searched.
- **Command Properties window**: Presents editable parameters. Many parameters have menus. Click the parameter in the Command Properties window to see if a down-arrow appears.
- **File tabs**: Lists the name of the traffic generation file and any open **Include** files that are associated with the generation file. If the generation file has **Include** statements, the **Include** files automatically open and display as tabs in the File tabs section of the Editor window.

Error Log: An error log opens automatically at the bottom of the window any time a script error occurs. The window closes automatically whenever the error is corrected, after you save the script.

Script Editor Toolbar

The toolbar contains buttons for saving your edits, navigating, searching and other functions.



Go to Trace View

Save. Saves your edits and immediately updates the setting bars and Frames shown in the trace window.

View Options. Opens a menu with three options:

- Enable Outlining
- Toggle Outlining
- Line Numbers.

See “View Options Menu” on page 221 for descriptions.

Cut.

Copy.

Paste.

Undo.

Redo.

Toggle Bookmark. Allows markers to be set or removed to aid in navigation.

Next Bookmark.

Previous Bookmark.

Clear All Bookmarks.

Find.

Replace. Find and replace.

Print.

Script Edit Window

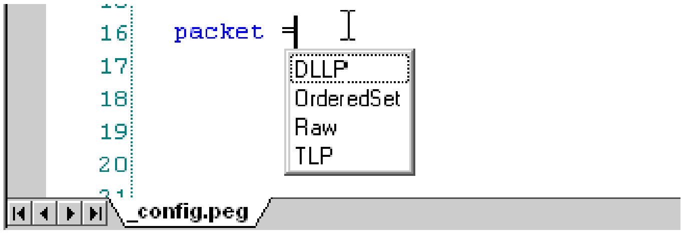
The Script Edit Window offers several features to simplify the process of editing.

Syntax Highlighting

All known commands and parameters are highlighted in **blue**.

All predefined values and command modifiers are highlighted in **brown**.

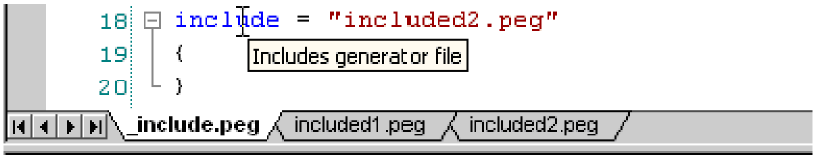
Intellisense prompts for known predefined values/literals. This functionality is invoked when you type **=** after a known key or select **List values** from the Context menu.



```
packet =
```

Tooltips

The Tooltips look like the following:

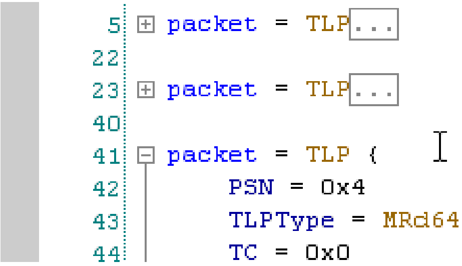


```
include = "included2.peg"
{
}
```

Includes generator file

Outlining

When outlining is enabled user have the option to **collapse/expand** code blocks:



```
packet = TLP...

packet = TLP...

packet = TLP {
        PSN = 0x4
        TLPType = MRd64
        TC = 0x0
```

Text Editing Commands

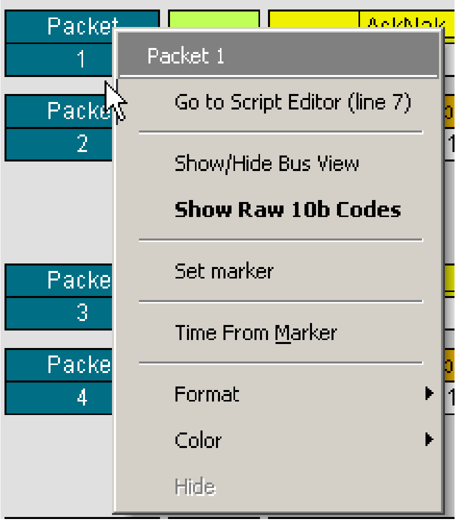
The Script Editor supports standard editor commands:

- Copy/Paste
- Undo/Redo
- Find/Replace
- Bookmarks

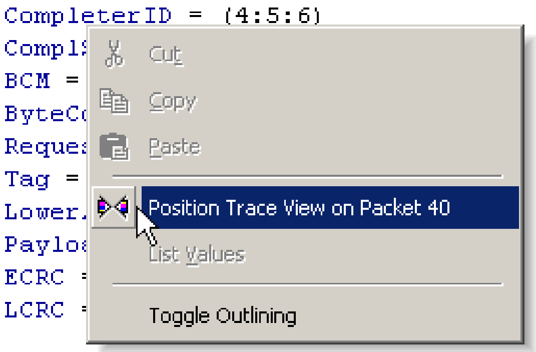
Synchronized Scrolling with the Trace Window

You can navigate from the generation code window to the corresponding place in the trace representation, and visa versa.

From the trace window, right-click the **first cell in a packet** in the trace window and select **Go to Script Editor** from the pop-up menu. The Script Editor window then repositions to the corresponding code.



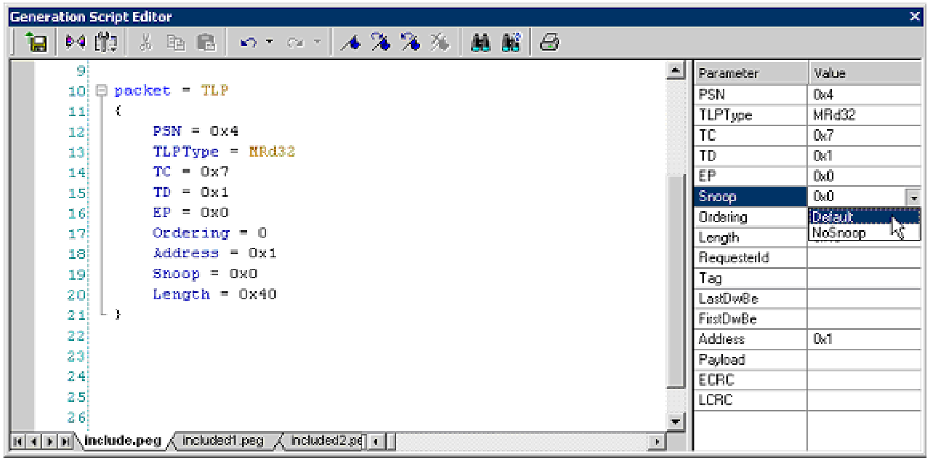
From the script editor, right-click some **code** within the Script Editor window and select **Position Trace view on packet x** from the pop-up menu (where **x** is a packet number). The trace window repositions to the corresponding packet number.



```
CompleterID = (4:5:6)
Compl$
BCM =
ByteC(
Reques
Tag =
Lower.
Payloa
ECRC =
LCRC =
```

Command Properties Window

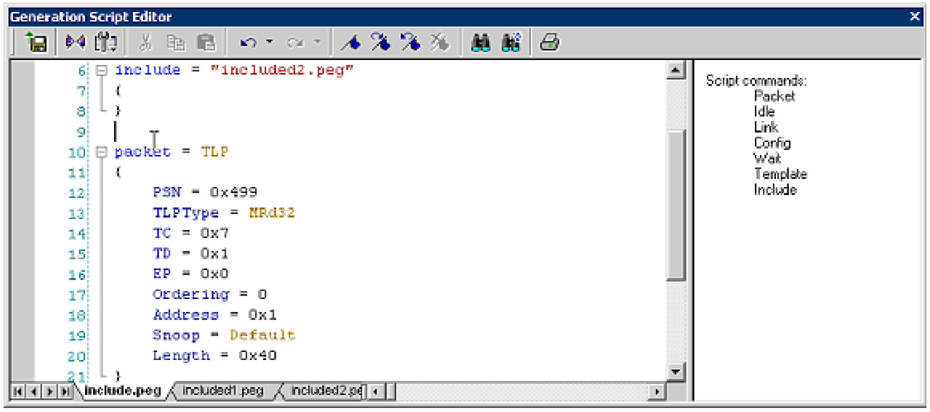
The Command Properties window lists all possible parameters for the current script command and all values for the parameters currently defined in this command. Entering parameters/values within the Edit window causes the Command Properties window to automatically update. Parameters/values can be changed by entering text into the text boxes or by selecting items from pull-down menus as shown in the example below.


Generation Script Editor

```
packet = TLP
{
    PSN = 0x4
    TLPType = MRd32
    TC = 0x7
    TD = 0x1
    EP = 0x0
    Ordering = 0
    Address = 0x1
    Snoop = 0x0
    Length = 0x40
}
```

Parameter	Value
PSN	0x4
TLPType	MRd32
TC	0x7
TD	0x1
EP	0x0
Snoop	0x0
Ordering	Default NoSnoop
Length	
RequesterId	
Tag	
LastDwBe	
FirstDwBe	
Address	0x1
Payload	
ECRC	
LCRC	

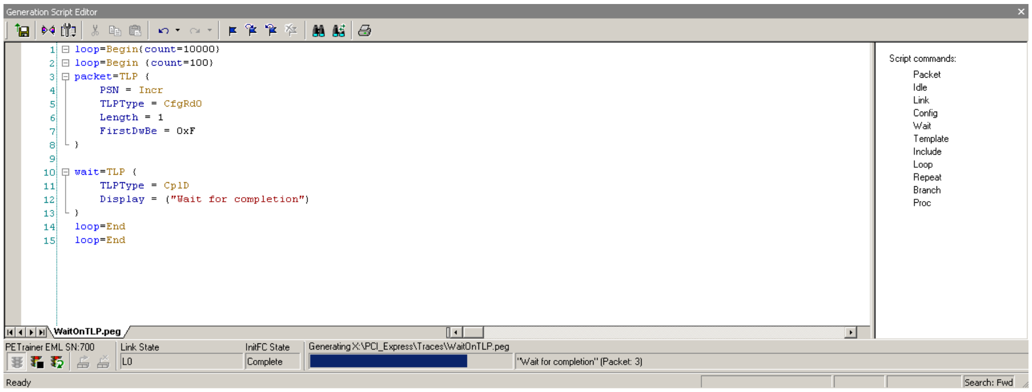
In this case, the edit window is updated automatically. If the current script command does not have parameters or the current cursor position is outside of any script command, then a generic prompt is displayed in the window:


Generation Script Editor

```
include = "included2.peg"
{
}

packet = TLP
{
    PSN = 0x499
    TLPType = MRd32
    TC = 0x7
    TD = 0x1
    EP = 0x0
    Ordering = 0
    Address = 0x1
    Snoop = Default
    Length = 0x40
}
```

Script commands:
Packet
Idle
Link
Config
Wait
Template
Include


Generation Script Editor

```
loop=Begin(count=10000)
loop=Begin (count=100)
packet=TLP {
    PSN = Incr
    TLPType = CfgRd0
    Length = 1
    FirstDwBe = 0xF
}

wait=TLP {
    TLPType = CplD
    Display = ("Wait for completion")
}
loop=End
loop=End
```

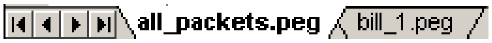
Script commands:
Packet
Idle
Link
Config
Wait
Template
Include
Loop
Repeat
Branch
Proc

PETrainer EML SN:700
Link State: L0
InitFC State: Complete
Generating X:\PCI_Express\Traces\WaitOnTLP.peg
"Wait for completion" (Packet: 3)

Script Editor File Tabs

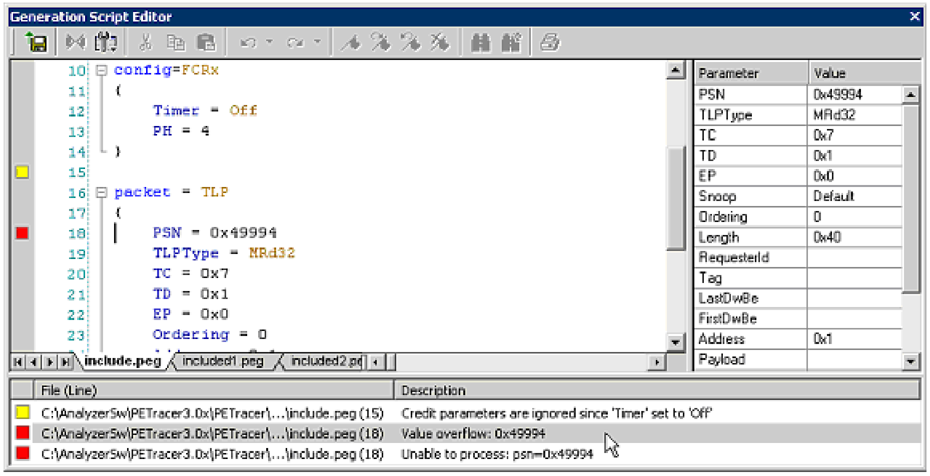
At the bottom of the Script Editor window is the name of the open generation file.

If there are **Include** statements in the generation file that link it to other generation files, these files automatically open and display as tabs at the bottom of the window. You can click the tabs to toggle between the open generation files.



Script Editor Error Log

Whenever you create a scripting error, a log opens at the bottom of the application window. When the error is corrected, the window automatically closes.



```
10  config=FCRx
11  {
12      Timer = Off
13      PH = 4
14  }
15
16  packet = TLP
17  {
18      PSN = 0x49994
19      TLPType = MRd32
20      TC = 0x7
21      TD = 0x1
22      EP = 0x0
23      Ordering = 0
```

Parameter	Value
PSN	0x49994
TLPType	MRd32
TC	0x7
TD	0x1
EP	0x0
Snoop	Default
Ordering	0
Length	0x40
RequesterId	
Tag	
LastDwBe	
FirstDwBe	
Address	0x1
Payload	

	File (Line)	Description
🟨	C:\AnalyzerSw\PETracer3.0x\PETracer\...\include.peg (15)	Credit parameters are ignored since 'Timer' set to 'Off'
🟥	C:\AnalyzerSw\PETracer3.0x\PETracer\...\include.peg (18)	Value overflow: 0x49994
🟥	C:\AnalyzerSw\PETracer3.0x\PETracer\...\include.peg (18)	Unable to process: psn=0x49994

Errors: Marked by red squares.

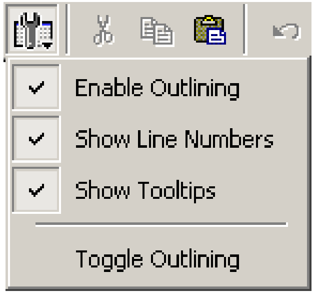
Warnings: Marked by yellow squares.

Double-clicking an error in the error log causes the cursor in the edit window to move to where the error was detected.

Note: You cannot run the script if it has syntax errors.

View Options Menu

The **View Options** button displays a menu with these options:



Enable Outlining: Adds an expandable/collapsible tree structure to the left side of the Script Editor showing the hierarchical relationships of the script lines.

Show Line Numbers: Adds line numbers to the left side of the Script Editor window.

Show Tooltips: Adds line numbers to the left side of the Script Editor window.

Toggle Outlining: Toggles the outline tree between collapsed and expanded states.

11.5 Generation Options Dialog Box Overview

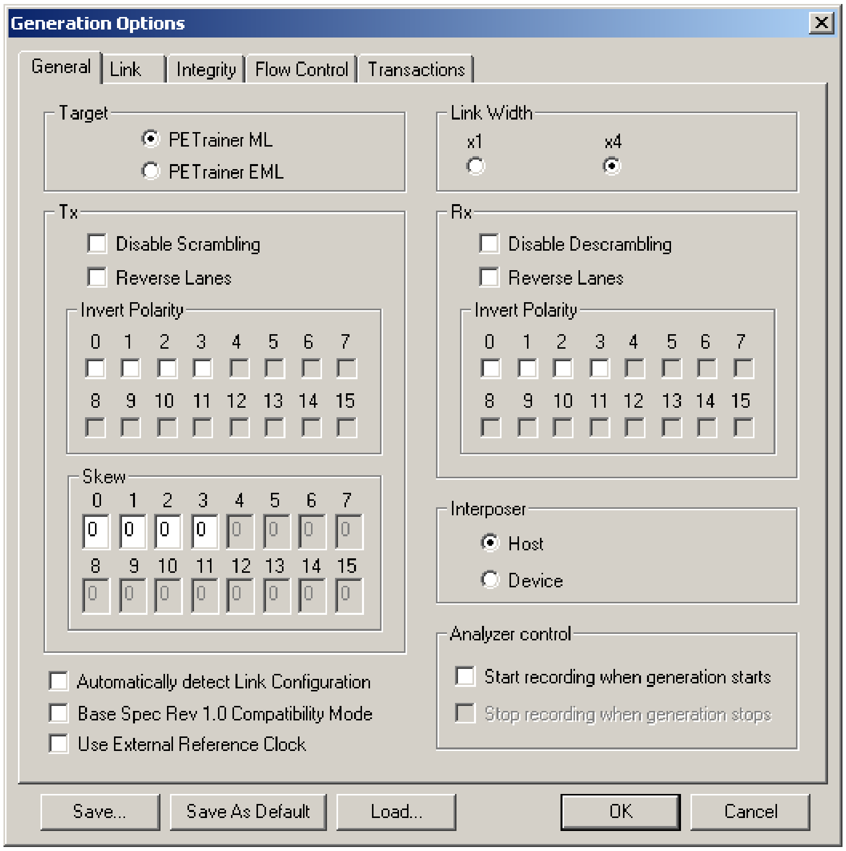
The Generation Options dialog box is used to set Config settings in a traffic generation script (for example, **Config = General** or **Config = Link**). This dialog duplicates the **Config** script command and is provided as a convenient alternative means of setting this command in the script.

Note: The Generation Options dialog is subordinate to the script itself, so script commands override options selected in this dialog box.

Opening the Dialog

To open the Generation Options dialog, select **Setup > Generation Options** or click the **Generation Options** button .

PE*Trainer* ML



Target
● PETrainer ML
○ PETrainer EML

Link Width
x1 ○
x4 ●

Tx
□ Disable Scrambling
□ Reverse Lanes

Invert Polarity

0	1	2	3	4	5	6	7
□	□	□	□	□	□	□	□
8	9	10	11	12	13	14	15
□	□	□	□	□	□	□	□

Skew

0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0
8	9	10	11	12	13	14	15
0	0	0	0	0	0	0	0

Rx
□ Disable Descrambling
□ Reverse Lanes

Invert Polarity

0	1	2	3	4	5	6	7
□	□	□	□	□	□	□	□
8	9	10	11	12	13	14	15
□	□	□	□	□	□	□	□

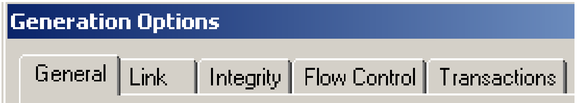
Interposer
● Host
○ Device

Analyzer control
□ Start recording when generation starts
□ Stop recording when generation stops

□ Automatically detect Link Configuration
□ Base Spec Rev 1.0 Compatibility Mode
□ Use External Reference Clock

Dialog Layout

The Generation Options dialog is organized into five pages: General, Link, Integrity, Flow Control, and Transactions.



The options in these pages correlate to those of the **Config** script command:

Dialog Page	**Script Command**
General	Config = General
Link	Config = Link

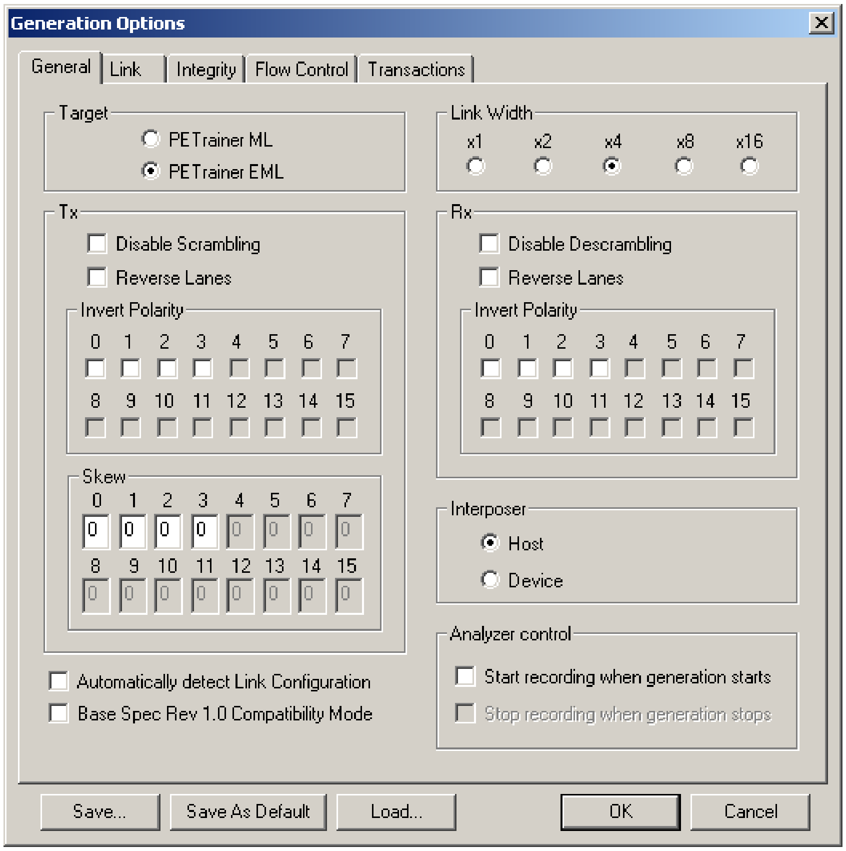
Integrity	Config = TLP
	Config = AckNak

Flow Control	Config = FCTx
	Config = FCRx

Transactions	Config = Transactions Also see “Configuration Space Editor” on page 306.

Generation Options - General

PE*Trainer* EML



Generation Options

General | Link | Integrity | Flow Control | Transactions

Target
○ PETrainer ML
● PETrainer EML

Link Width
x1 ○ x2 ○ x4 ● x8 ○ x16 ○

Tx
□ Disable Scrambling
□ Reverse Lanes

Invert Polarity

0	1	2	3	4	5	6	7
□	□	□	□	□	□	□	□
8	9	10	11	12	13	14	15
□	□	□	□	□	□	□	□

Skew

0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0
8	9	10	11	12	13	14	15
0	0	0	0	0	0	0	0

Rx
□ Disable Descrambling
□ Reverse Lanes

Invert Polarity

0	1	2	3	4	5	6	7
□	□	□	□	□	□	□	□
8	9	10	11	12	13	14	15
□	□	□	□	□	□	□	□

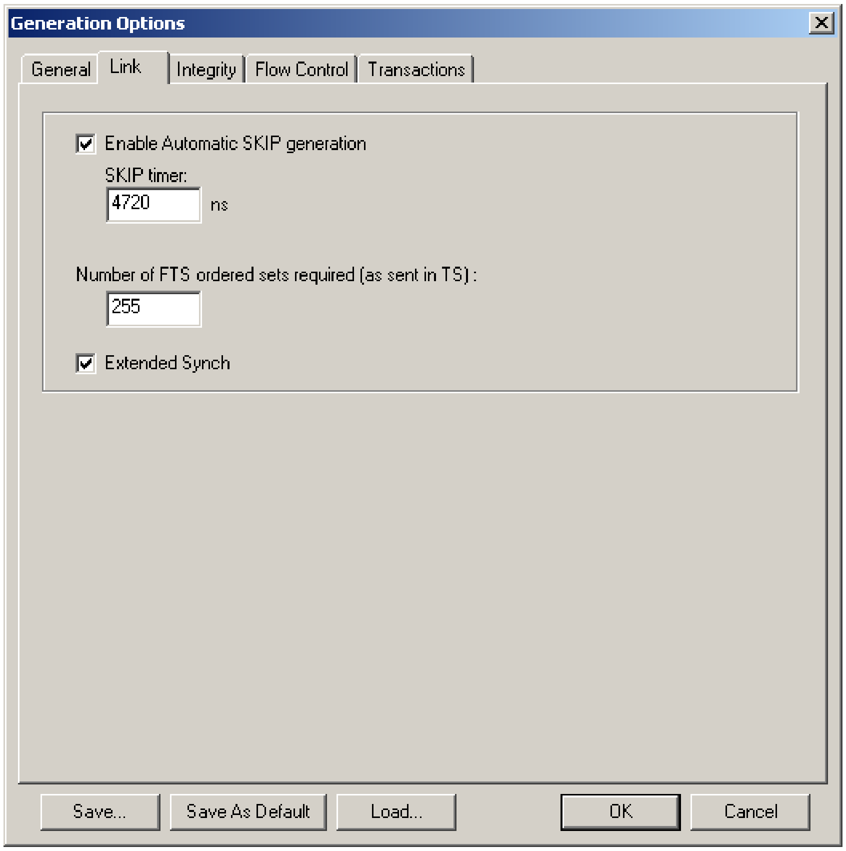
Interposer
● Host
○ Device

□ Automatically detect Link Configuration
□ Base Spec Rev 1.0 Compatibility Mode

Analyzer control
□ Start recording when generation starts
□ Stop recording when generation stops

Parameter	Values	Default	Comment
Target			Specifies the generation platform. The choice affects some of the options presented in the Generation Options dialog.
LinkWidth	1 2 4 8 16	1	Ignored if AutoDetect is set.
DisableScrambleTx	Yes No	No	Ignored if AutoDetect is set.
DisableDescrambleRx	Yes No	No	Ignored if AutoDetect is set.
ReverseLinesTx	Yes No	No	Ignored if AutoDetect is set.
ReverseLinesRx	Yes No	No	Ignored if AutoDetect is set.
InvertPolarityTx	(X,X,X,X)		Ignored if AutoDetect is set.
InvertPolarityRx	(X,X,X,X)		Ignored if AutoDetect is set.
SkewTx	(X,X,X,X)		
Interposer	Host Device	Host	Host = Host Emulation
BaseSpec10	Yes No	No	
UseExtRefClock	Yes No	Yes	Use external reference clock. (PE*Trainer* ML only)
AutoDetect	Yes No	Yes	Automatically detects Link Width (PE*Trainer* ML only), Polarity Inversion, Lane Reversal, and Scrambling.

Generation Options - Link



☑ Enable Automatic SKIP generation

SKIP timer: 4720 ns

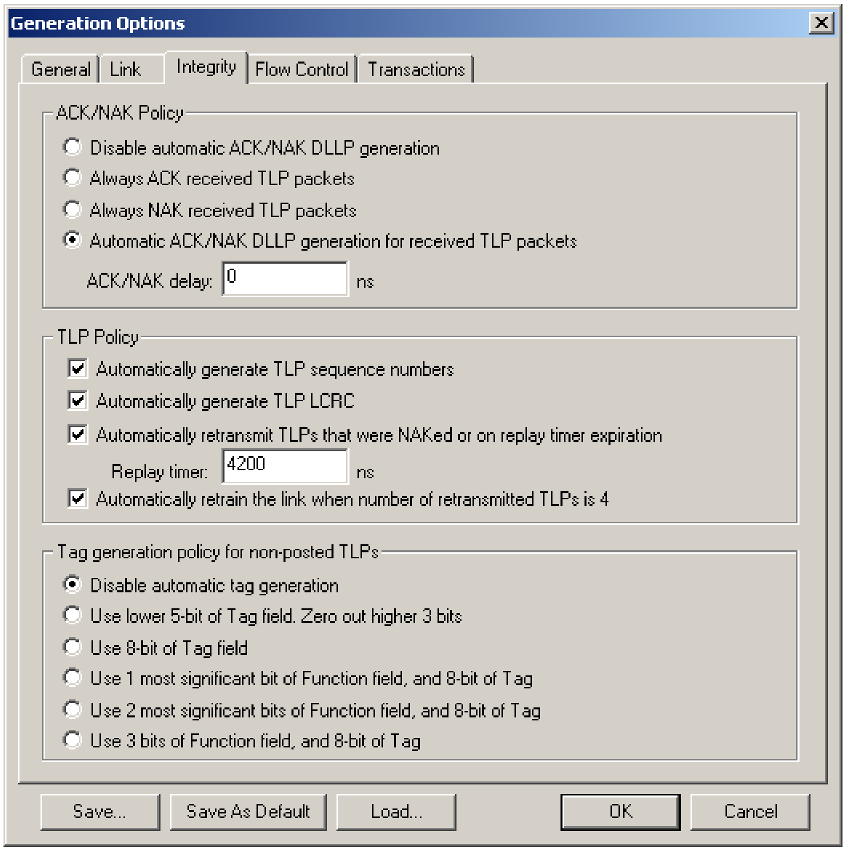
Number of FTS ordered sets required (as sent in TS) : 255

☑ Extended Synch

Parameter	Values	Default	Comment
SkipTimer	In ns (rounded to nearest 8) Off	4720	Periodic timer that controls sending of SKIP ordered sets at specific intervals. Timer’s value is measured in 1us units. The SKIP timer should be 4720 ns <-> 6152 ns per the spec.
FTSCount	0 to 255	255	Number of FTS ordered sets required (as sent in TS)
ExtendedSynch	Yes No	Yes	Forces LTSSM to send 4096 Fast Training Sequences when leaving LOs state. The value entered in the text box sets the NUM_FTS field in training patterns TS1, TS2 as generated by LTSSM.

Generation Options - Integrity

The Integrity page sets the parameters for two Config commands: **Config = TLP** and **Config = AckNak**



ACK/NAK Policy

○ Disable automatic ACK/NAK DLLP generation
○ Always ACK received TLP packets
○ Always NAK received TLP packets
● Automatic ACK/NAK DLLP generation for received TLP packets

ACK/NAK delay: 0 ns

TLP Policy

☑ Automatically generate TLP sequence numbers
☑ Automatically generate TLP LCRC
☑ Automatically retransmit TLPs that were NAKed or on replay timer expiration

Replay timer: 4200 ns

☑ Automatically retrain the link when number of retransmitted TLPs is 4

Tag generation policy for non-posted TLPs

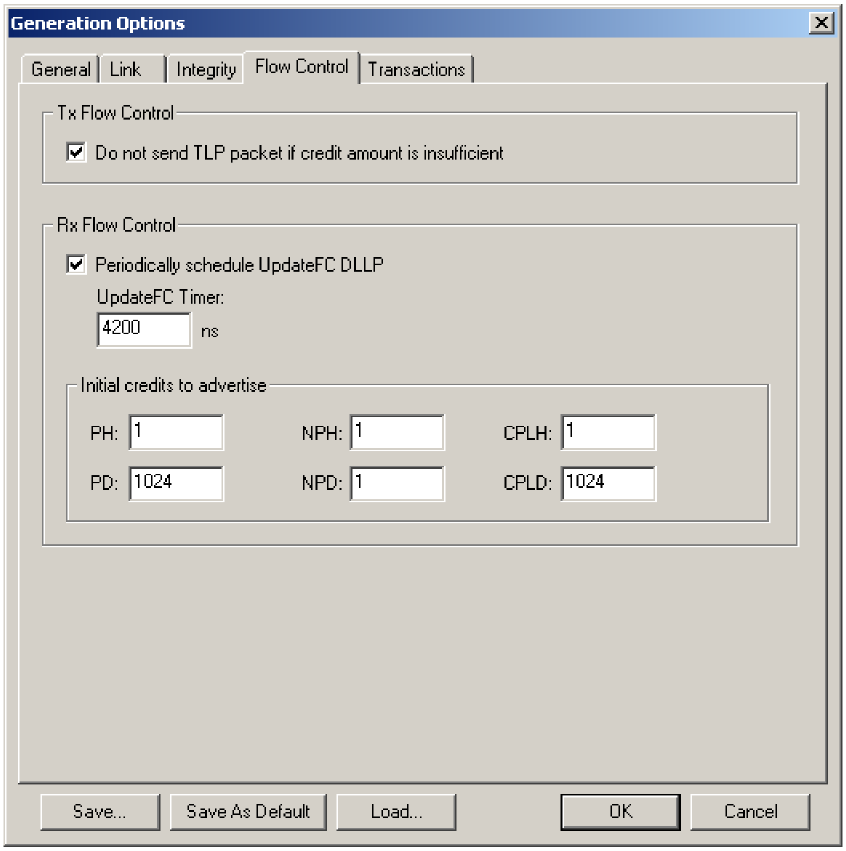
● Disable automatic tag generation
○ Use lower 5-bit of Tag field. Zero out higher 3 bits
○ Use 8-bit of Tag field
○ Use 1 most significant bit of Function field, and 8-bit of Tag
○ Use 2 most significant bits of Function field, and 8-bit of Tag
○ Use 3 bits of Function field, and 8-bit of Tag

AckNak Parameter	**Values**	**Default**	**Comment**
AckNak	Auto Ack Nak Disable	Auto	**Auto**: Automatic ACK/NAK (default) **Ack**: Always ACK **Nak**: Always NAK **Disable**: Disable automatic ACK/NAK DLLP generation.
Delay	In ns (rounded to nearest 8)	0	Timer that controls how much delay is added to AckNak DLLP response after TLP reception. Valid if AckNak is Auto, Ack, or Nak. Used to delay AckNak, thereby allowing customers to test their replay mechanisms and replay timer.

TLP Parameter	Values	Default	Comment
AutoSeqNumber	Yes No	Yes	If set to 0, overrides automatic generation of the TLP sequence number and uses user-defined value of the field in the **Packet=TLP** commands. This option overrides any sequence numbers specified in the script and LCRC specified in the script.
ReplayTimer	In ns (rounded to nearest 8) Off	4200	Timeout in TLP transmitter path that counts time since last Ack or Nak DLLP is received. If set, automatically retransmit TLPs that were NAKed or on replay timer expiration.
AutoRetrain	Yes No	Yes	If set, enable automatic retraining of the link in case the number of retransmitted TLP is 4. Valid only when AutoRetransmission is set.
Tag Generation (PE*Trainer* EML only)			Disable automatic tag generation. Prevents PE*Trainer* from automatically inserting a tag. Tags are a sub-field of the transaction ID field. When auto tag insertion is enabled, the tag field is only modified for non-posted transactions such as **CfgRd, CfgWr**, and **MemRd**.

Generation Options - Flow Control

The Flow Control page sets parameters for **Config = FCTx** and **Config = FCRx**.



Tx Flow Control

☑ Do not send TLP packet if credit amount is insufficient

Rx Flow Control

☑ Periodically schedule UpdateFC DLLP

UpdateFC Timer: 4200 ns

Initial credits to advertise

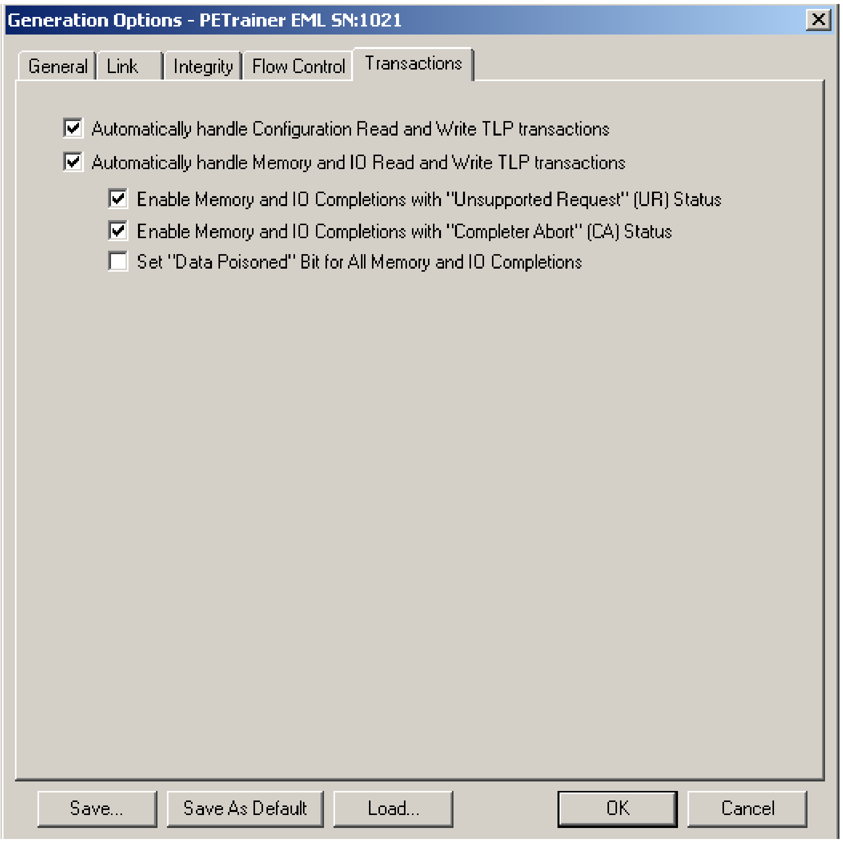
PH:	1	NPH:	1	CPLH:	1
PD:	1024	NPD:	1	CPLD:	1024

FCRx Parameter	Values	Default	Comment
Enable Tx Flow Control	Yes No	Yes	When not set, the TLPs are being sent without the regard of how many credits are available. This option prevents TLP transmission if insufficient credits are available.
Enable Rx Flow Control	In ns (rounded to nearest 8) Off	4200	When enabled, allows automatic updating of these DLLPs. This option enables a periodic timer that controls sending of UpdateFC DLLPs. You should leave timer and credit values to defaults for correct behavior.
PH	0 to 255	1	Posted Request Headers
NPH	0 to 255	1	Non-Posted Request Headers
CplH	0 to 255	1	Completion Headers
PD	0 to 4095	1024	Posted Request Data Payload
NPD	0 to 4095	1	Non-Posted Request Data Payload
CplD	0 to 4095	1024	Completion Data Payload

11.6 Generation Options - Transactions

The Transaction page displays different options depending on whether you are running PE*Trainer* ML or PE*Trainer* EML (selected at the General tab).

PE*Trainer* EML



Generation Options - PETrainer EML SN:1021

☑ Automatically handle Configuration Read and Write TLP transactions

☑ Automatically handle Memory and IO Read and Write TLP transactions

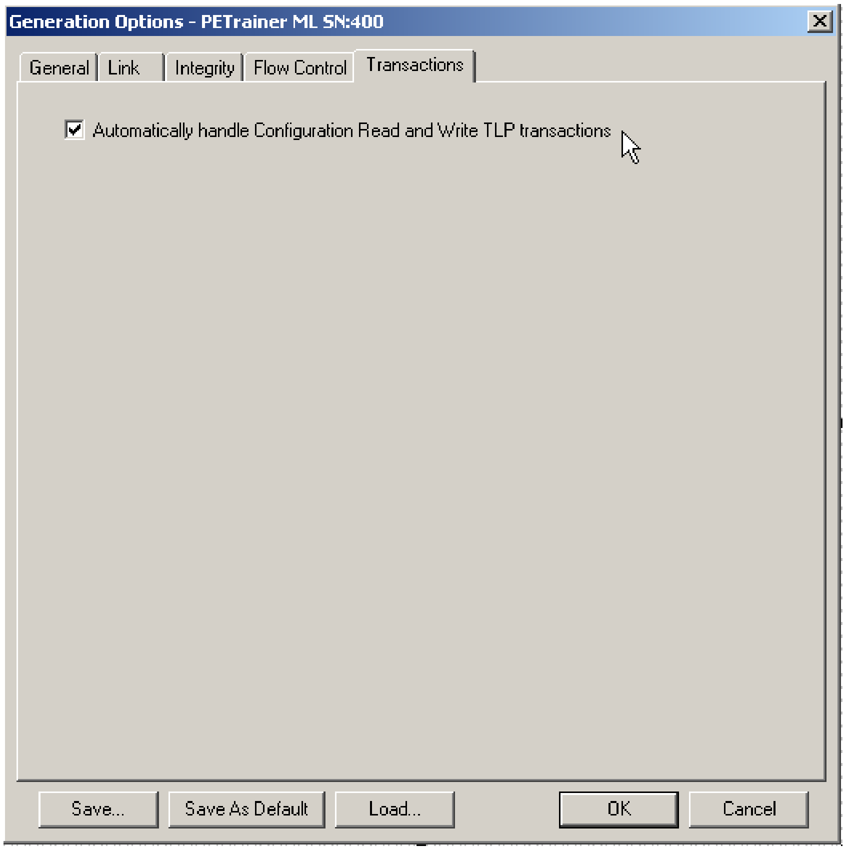
☑ Enable Memory and IO Completions with "Unsupported Request" (UR) Status

☑ Enable Memory and IO Completions with "Completer Abort" (CA) Status

□ Set "Data Poisoned" Bit for All Memory and IO Completions

Parameter	Values	Default	Comment
Automatically handle Configuration Read and Write TLP transactions	Yes No	No	If set, automatically handles Configuration Read and Write TLP transactions. For Configuration Read transaction, Completion TLP contains the data read from the internal Configuration Space according to specified register address. For Configuration Write transaction, internal Configuration Space is updated at the address with the data taken from Configuration Write TLP, and Configuration Write Completion is returned. This option enables Read and Write access to 4 KB configuration space.
Automatically handle Memory and IO Read and Write TLP transactions	Yes No	No	If set, automatically handles Memory and IO Read and Write TLP transactions. For Memory and IO Read transactions, Completion TLP contains the data read from the internal Memory/IO Address Space according to specified address. For Memory and IO Write transactions, internal Memory/IO Address Space is updated at the address with the data taken from TLP.
Enable Memory and IO Completions with Unsupported Request (UR) Status	Yes No	No	If set, enables Unsupported Request (UR) status for Memory/IO completions. AutoMemIoCompletion must be set to enable UR completions.
Enable Memory and IO Completions with Completer Abort (CA) Status	Yes No	No	If set, enables Completer Abort (CA) status for Memory/IO completions. AutoMemIoCompletion must be set to enable CA completions.
Set **Data Poisoned** Bit for All Memory and IO Completions	Yes No	No	If set, all Memory/IO completions have **Poisoned** bit set.

PE*Trainer* ML



☑ Automatically handle Configuration Read and Write TLP transactions

Parameter	Values	Default	Comment
Automatically handle Configuration Read and Write TLP transactions	Yes No	No	If set, automatically handles Configuration Read and Write TLP transactions. For Configuration Read transaction, Completion TLP contains the data read from the internal Configuration Space according to specified register address. For Configuration Write transaction, internal Configuration Space is updated at the address with the data taken from Configuration Write TLP, and Configuration Write Completion is returned. This option enables Read and Write access to 4 KB configuration space.

11.7 Running a PE*Trainer* Generation File

To run PE*Trainer* generation script:

Step 1 Open an existing script file or create new script.

Step 2 If needed, make changes to the script file,

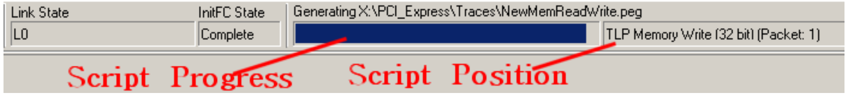
then save the file by clicking the **Save** button .

Step 3 Start generation by clicking the **Start Traffic Generation** button on the Status bar. This action causes the software to download the script to the PE*Trainer* device. After the download completes, the PE*Trainer* device starts executing the script.

Note: Script execution can be terminated at any time by clicking the

Stop Generation button on the toolbar.

As the script executes, the Status bar displays the script's progress. The **Current script position** field displays the command description and trace packet number currently being generated.



Link State: L0
InitFC State: Complete
Generating X:\PCI_Express\Traces\NewMemReadWrite.peg
TLP Memory Write (32 bit) (Packet: 1)

If a Wait command is executed where the Display parameter is specified, the user-defined text is displayed.

If a **wait=user** script command is executed, the script pauses until you click the

Resume Generation button on the toolbar.

For further details on Wait command syntax and usage, see “Wait Command” on page 273.

Chapter 12: Traffic Generation Language

12.1 Language Syntax

Syntax for the PE*Trainer*™ Generator Script files is:

```
COMMAND = MODIFIER
{
  PARAM1 = VALUE1
  …
  PARAMn = VALUEn
}
```

For some commands, the list of the parameters is optional.

All literals are not case sensitive.

All default values are zeros unless otherwise noted.

Integer literals represent numeric values with no fractions or decimal points.

Hexadecimal, decimal, and binary notation are supported:

- Hexadecimal numbers must be preceded by **0x**: 0x2A, 0x54, 0xFFFFFF01
- Decimal numbers are written as usual: 24, 1256, 2
- Binary numbers are denoted with **0b**: 0b01101100, 0b01, 0b100000

It is possible to use expressions, for example, **(i - 239)**.

String literals are surrounded by double quotes.

Array data types are represented by integer or string literals surrounded by “**(**“ and “**)**” characters, and separated by comma “**,**”, for example, **(2,23,4)**.

Single-line comments are supported and should be preceded semicolon “**;**”.

Multi-line comments are also supported. Multi-line comments begin with a “**/***” combination, and end with the reverse “***/**” combination.

12.2 Command List

The following list summarizes the traffic generation commands supported by the PE*Trainer*.

Command	Modifiers	Comment
Packet	TLP DLLP OrderedSet Raw <TemplateName>	Sends a packet.
Idle	<# of ns>	Sends idle symbols (D0.0).
Link	L0 L1 L0s HotReset Disabled Recovery Detect LTSSMOff InitFC	Sets a link condition.
Config	General FCTx FCRx TLP AckNak Transactions Link Definitions	Configures the PE*Trainer*.
Wait	TLP DLLP Error LinkCondition BOB Payload User	Waits for the condition specified.
Include	<Include file path>	Includes a PE*Trainer* script file.

Branch	TLP DLLP Error Link BOB Payload User	Enables/disables an interrupt for the specified condition.
Proc	Begin End	Declares the procedure to be used in a branch statement.
Loop	Begin End	Creates a PE*Trainer* loop.
Repeat	Begin End	Repeats traffic some number of times.
Template	TLP DLLP OrderedSet Raw <TemplateName>	Creates a template for a packet that can be used in the Packet command.
AddressSpace	Read Write	Reads/Writes address space.

12.3 Packet Command

This command initiates transmission of a specified packet on the bus.

Command	**Modifiers**	**Comment**
Packet	TLP DLLP OrderedSet Raw <TemplateName>	Sends packet.

Packet = TLP

Function: Initiates transmission of a TLP packet on the bus. The parameters of the **Packet = TLP** command cover all the fields in the TLP header: TLP Payload, PSN (Packet Sequence Number), ECRC, and LCRC. Reserved fields can be set with the RawData parameter.

Parameter	Values	Default Value	Comment
PSN	0:4095 Incr	0	When **Incr** is specified, the PSN for the current TLP is assigned as the PSN of the previously sent TLP incremented by 1. When the PSN is generated automatically (see the **AutoSeqNumber** parameter), this parameter has no effect.
TLPType	MRd32 MRdLk32 MWr32 MRd64 MRdLk64 MWr64 IoRd IoWr CfgRd0 CfgWr0 CfgRd1 CfgWr1 Msg MsgD Cpl CplLk CplD CplDLk	0	Sets the **Fmt** (bits 6:5 of byte 0 in the TLP header) and **Type** (bits 4:0 of byte 0 in the TLP header) fields in the TLP header. Also, this field can be specified as a direct numeric value that specifies bits 6:0 of byte 0 in the TLP header.
TC	0:7	0	**Traffic Class.** Bits 6:4 of byte 1 in the TLP header
TD	0:1	0	Bit 7 of byte 2 in the TLP header: 1 indicates presence of TLP digest in the form of a single DW at the end of the TLP.
EP	0:1	0	Bit 6 of byte 2 in the TLP header. Indicates the TLP is poisoned.
Snoop	0:1	0	Bit 4 of byte 2 in the TLP header: 0 indicates that hardware enforced cache coherency is expected. 1 indicates that hardware enforced cache coherency is not expected.

Ordering	0:1	0	Bit 5 of byte 2 of TLP header 0 indicates PCI Strongly Ordered Model. 1 indicates PCI-X Relaxed Ordering Model.
Length	0:1023	0	Length of data payload in DWORDs. If not specified, this field is 1 for all read requests and is calculated according to the actual payload for write requests.
Tag	0:255	0	Byte 6 of the TLP Header for Memory, IO and Configuration TLP packets Byte 10 for Completion TLP packets When Tags are generated automatically (see the **TagGeneration** parameter), this parameter has no effect for Memory, IO, and Configuration TLP packets.
RequesterID	(XX:XX:X) direct value	0	Bytes 4-5 of the TLP Header for Memory, IO, and Configuration TLP packets Bytes 8-9 for Completion TLP packets This parameter can be set in the following format: **(BusNumber : DeviceNumber : FunctionNumber)**
ECRC	0x00000000: 0xFFFFFFFF	Calculated automatically	When not specified, the PE*Tracer* software automatically calculates the ECRC. (TD field has to be specified.)
LCRC	0x00000000: 0xFFFFFFFF	Calculated automatically	When not specified, the PE*Tracer* software automatically calculates the LCRC. When LCRC is generated automatically by the PE*Trainer* hardware (see the **AutoLCRC** parameter), this parameter has no effect.
Payload	(XXXX,XXXX,…) Incr Random Zeros Ones		Specified as the array of DWORDs in hexadecimal format (Big Endian). The **Payload** parameter applies only to TLP packets with data. **Incr**: Specifies a payload as the sequence (0, 1, …'Length'). **Random**: Specifies a random payload. **Zeros**: Specifies a payload of all zeros. **Ones**: Specifies a payload of all ones. Payload can be specified for Memory, IO, Configuration writes, and Completion with Data TLP packets. **Note**: When Incr, Random, Zeros, and Ones are used, the Length parameter must be specified before the payload.

Field[<start>:<end>] Field[<pos>]			The arbitrary TLP Header field could be specified by using **Field** parameter. **Start, end**, and **pos** are bit positions from the beginning of TLP Header. Position 0 corresponds to the Most Significant Bit of the first byte of TLP Header. Position 95 for 3 DWORD header (and position 127 for 4 DWORD header) correspond to the Least Significant Bit of the last byte of TLP. Header. Fields are limited by 32 bit values. Use **Field[<start>:<end>]** syntax to specify a multi-bit field. Use **Field[<pos>]** to specify a single-bit field.
RawData@<start>			Inserts raw data symbols at **<start>** byte position from the beginning of the TLP. See the **Packet = Raw** description for possible raw data formats.
Count	1:65535	1	Repeats this packet by the number of times specified.

Example 1

Read one DWORD of data from address 0x1000.

Length parameter is not specified, so the default value of 1 is used.

TC, TD, EP, Ordering, Snoop, and **Tag** parameters are not specified, so the default value of 0 is used.

LCRC is not specified, so the **LCRC** is calculated by software.

```
Packet = TLP {
PSN = 0
TLPType = MRd32
Address = 0x1000
}
```

Example 2

Read 32 DWORDs of data starting from address 0x1000.

PSN accepts values 0 for first TLP and 1 for second TLP.

TC, **EP**, **Ordering**, and **Snoop** parameters are not specified, so the default value of 0 is used.

LCRC is not specified, so the **LCRC** is calculated by software.

ECRC is not specified, so the **ECRC** is calculated by software.

```
Packet = TLP {
PSN = Incr
TLPType = MRd32
Tag = 0
Address = 0x1000
TD = 1
FirstDwBe = 0xF
Length = 16
}

Packet = TLP {
PSN = Incr
TLPType = MRd32
Tag = 1
Address = 0x1010
TD = 1
FirstDwBe = 0xF
Length = 16
}
```

Example 3

This example does not specify **PSN, Tag,** and **LCRC**. Those values are calculated automatically by the PE*Trainer* hardware (see more on **Config = TLP** command).

```
Config = TLP {
AutoSeqNumber = Yes
AutoLCRC = Yes
TagGeneration = Default
}

Packet = TLP {
TLPType = MRd32
Address = 0x1010
TD = 1
Length = 1
}
```

Example 4

This example shows how to specify a reserved field in the TLP header using the **RawData** parameter (see more on the **RawData** parameter).

```
Packet = TLP {
TLPType = MRd32
Address = 0x1010
RawData@4 = ( D1 )
}
```

Example 5

This example shows how to specify reserved fields in the TLP header using the **Field** parameter.

```
Packet=TLP {
TLPType=CfgRd0
Register = 0x34
Length = 1
FirstDwBe = 0xF
Field[0] = 0x1
Field[8] = 0x1
Field[12:15] = 0xF
Field[20:21] = 0x3
Field[80:83] = 0xF
}
```

Example 6

This example shows how to specify the TLP type directly. Any invalid TLP type can be generated with this method.

```
Packet = TLP {
TLPType = 0x4F
}
```

Example 7

Repeat this TLP packet 64 times.

```
Packet = TLP {
TLPType = MRd32
Address = 0x1000
Count = 64
}
```

TLPType Equals Mrd32, Mrdlk32, Mwr32

Parameter	Value	Default	Comment
LastDwBe	0:15	0	Byte 7 in the TLP header. See rules for **Last DW BE** in the PCI Express Specification.
FirstDwBe	0:15	0	Byte 7 in the TLP header. See rules for **1st DW BE** in the PCI Express Specification.
Address	0x00000000: 0xFFFFFFFF	0	Bytes 8-11 in the TLP header.

Example 1

This example shows how to send a 32-bit Memory Write TLP.

The **Length** field is not specified, so it is calculated by software. (**Length = 4** is used.)

TC, TD, EP, Ordering, Snoop, and **Tag** parameters are not specified, so the default value of 0 is used.

LCRC is not specified, so the **LCRC** is calculated by software.

```
Packet = TLP {
TLPType = MWr32
LastDwBe = 0xF
FirstDwBe = 0xF
Address = 0x1000
Payload = ( 0x2, 0x4, 0x6, 0x8 )
}
```

Example 2

This example shows how to send a 32-bit Memory Write TLP. This command generates a random payload of 1024 DWORDs

```
Packet = TLP {
TLPType = MWr32
LastDwBe = 0xF
FirstDwBe = 0xF
Address = 0x1000
Length = 0; 0 means 1024 DWORDs of payload
Payload = Random
}
```

Type Equals Mrd64, Mrdlk64, Mwr64

Parameter	Value	Default	Comment
LastDwBe	0:15	0	Byte 7 in the TLP header. See rules for **Last DW BE** in the PCI Express Specification.
FirstDwBe	0:15	0	Byte 7 in the TLP header. See rules for **1st DW BE** in the PCI Express Specification.
AddressLo	0x00000000: 0xFFFFFFFF	0	Bytes 8-11 in the TLP header.
AddressHi	0x00000000: 0xFFFFFFFF	0	Bytes 12-15 in the TLP header.

Example 1
This example shows how to send a 64-bit Memory Write TLP.

Length parameter is set to 3 intentionally in order to generate a TLP with incorrect length.

TC, TD, EP, Ordering, Snoop, and **Tag** parameters are not specified, so the default value of 0 is used.

LCRC is not specified, so the **LCRC** is calculated by software.

```
Packet = TLP {
TLPType = MWr64
LastDwBe = 0xF
FirstDwBe = 0xF
AddressLo = 0x1000
AddressHi = 0x60000000
Payload = ( 0x2, 0x4, 0x6, 0x8, 0x2, 0x4, 0x6, 0x8 )
Length = 3
}
```

Type Equals Iord, Iowr

Parameter	Value	Default	Comment
LastDwBe	0:15	0	Byte 7 in the TLP header. See rules for **Last DW BE** in the PCI Express Specification.
FirstDwBe	0:15	0	Byte 7 in the TLP header. See rules for **1st DW BE** in the PCI Express Specification.
Address	0x00000000: 0xFFFFFFFF	0	Bytes 8-11 in the TLP header.

Example 1
Read one DWORD of data from address 0x1000 of the IO address space.

Length parameter is not specified; so the default value of 1 is used.

TC, TD, EP, Ordering, Snoop, and **Tag** parameters are not specified, so the default value of 0 is used.

LCRC is not specified, so the **LCRC** is calculated by software

```
Packet = TLP {
TLPType = IoRd
Address = 0x1000
}
```

Type Equals Cfgrd0, Cfgwr0, Cfgrd1, Cfgwr1

Parameter	Value	Default	Comment
LastDwBe	0:15	0	Byte 7 in the TLP header. See rules for **Last DW BE** in the PCI Express Specification.
FirstDwBe	0:15	0	Byte 7 in the TLP header. See rules for **1st DW BE** in the PCI Express Specification.
DeviceID	(XX:XX:X) direct value	0	Bytes 8-9 in the TLP header. This parameter can be set in the following format: **(BusNumber : DeviceNumber : FunctionNumber)**
Register		0	Bytes 10-11 in the TLP header.

Example 1
This example reads the Capability Pointer from the device’s configuration space (Bus Number 0, Device Number 2, Function Number 4).

```
Packet = TLP {
TLPType = CfgRd0
DeviceId = (0:2:4)
Register = 0x34
Length = 1
FirstDwBe = 0x1
}
```

Example 2
This example writes to the Command Register of the device’s configuration space (Bus Number 0, Device Number 0, Function Number 1).

```
Packet = TLP {
TLPType = CfgWr0
DeviceId = 1
Register = 0x04
Length = 1
FirstDwBe = 0x3
Payload = ( 0x03000000 )
}
```

Type Equals Msg, Msgd

Parameter	Value	Default	Comment
MessageRoute	ToRootComplex ByAddress ByID FromRootComplex Local Gather	ToRootComplex	**MessageRoute** affects the **Type** field of TLP header. (Bits 2:0)
MessageCode	Assert_INTA Assert_INTB Assert_INTC Assert_INTD Deassert_INTA Deassert_INTB Deassert_INTC Deassert_INTD PM_Active_State_Nak PM_PME PME_Turn_Off PME_TO_Ack ERR_COR ERR_NONFATAL ERR_FATAL Unlock Set_Slot_Power_Limit Vendor_Defined_Type0 Vendor_Defined_Type1 Attention_Indicator_On Attention_Indicator_Blink Attention_Indicator_Off Power_Indicator_On Power_Indicator_Blink Power_Indicator_Off Attention_Button_Pressed Direct numeric values can also be used	0	Byte 7 in the TLP Header

AddressHi	0x00000000: 0xFFFFFFFF	0	Used only if MessageRoute=ByAddress
AddressLo	0x00000000: 0xFFFFFFFF	0	Used only if MessageRoute=ByAddress
DeviceID	(XX:XX:X) direct value	0	Used only if **MessageRoute=ById**. This parameter can be set in the following format: **(BusNumber : DeviceNumber : FunctionNumber)**

Example 1
This example shows how to send a **PME_Turn_Off** Power Management Message while emulating the Root Complex.

```
Packet = TLP {
TLPType = Msg
MessageCode = PME_Turn_Off
MessageRoute = FromRootComplex
}
```

Example 2
This example shows how to send a **Vendor_Defined_Type0** Vendor Defined Message to the function 1 of device 1 on bus 0.

```
Packet = TLP {
TLPType = Msg
MessageCode = Vendor_Defined_Type0
MessageRoute = ByID
DeviceID = (0:1:1)
}
```

TLPType = Cpl, CplLk, CplD, CplDLk

Parameter	Value	Default	Comment
CompleterId	(XX:XX:X) direct value	0	Identifies the Completer. This parameter can be set in the following format: (**BusNumber : DeviceNumber : FunctionNumber**)
ComplStatus	SC UR CRS CA	SC	Indicates the completion status.
BCM	0:1	0	**Byte Count Modified**: Must not be set by PCI Express Completers and may only be set by PCI-X completers. Indicates that the Byte Count field reports the size of just the first packet instead of the entire remaining byte count.
ByteCount	0:4095	0	Remaining byte count for the request
LowerAddr	0:63	0	Lower byte address for the starting byte of the completion

Example 1
This example shows how to send a **Completion TLP**. This **Completion TLP** returns **Unsupported Request (UR)** status.

Requester is Function 0 of Device 0 on Bus 0.

Completer is Function 0 of Device 1 on Bus 0.

This completes the **TLP** request with Tag Number 4.

```
Packet = TLP {
TLPType = Cpl
RequesterId = (0:0:0)
CompleterId = (0:1:0)
Tag=4
ComplStatus = UR
}
```

Example 2
This example shows how to send a **Completion with Data TLP**. This **Completion TLP** returns **Successful Completion (SC)** status.

Requester is Function 0 of Device 0 on Bus 0.

Completer is Function 0 of Device 1 on Bus 0.

This completes the **TLP** request with Tag Number 4.

This is the last **Completion of the Split Transaction since ByteCount** field is equal to the number of bytes transferred and BCM is not set.

```
Packet = TLP {
TLPType = CplD
RequesterId = (0:0:0)
CompleterId = (0:1:0)
Tag=4
ComplStatus = SC
ByteCount = 32
Payload = ( 0x00000001, 0x00000002, 0x00000003, 0x00000004,
          0x00000005, 0x00000006, 0x00000007, 0x00000008 )
}
```

Packet = DLLP

Function: Initiates transmission of a DLLP packet on the bus.

Parameters for the **Packet = DLP** command cover all the fields in a DLLP.

Reserved fields can be set using the **RawData** parameter.

Parameter	Values	Default	Comment
DLLPType	Ack Nak InitFC1_P InitFC1_NP InitFC1_Cpl InitFC2_P InitFC2_NP InitFC2_Cpl UpdateFC_P UpdateFC_NP UpdateFC_Cpl PM_Enter_L1 PM_Enter_L23 PM_Active_State_Request_L1 PM_Request_Ack Vendor		First byte in the DLLP
CRC	0: 65535	Automatically calculated	Bytes 4-5 in the DLLP. When not specified, calculated automatically.
Field[<start>:<end>] Field[<pos>]			The arbitrary DLLP field could be specified by using **Field** parameter. **Start, end**, and **pos** are bit positions from the beginning of DLLP. Position 0 corresponds to the Most Significant Bit of the first byte of DLLP. Position 31 corresponds to the Least Significant Bit of the last byte of DLLP. Use **Field[<start>:<end>]** syntax to specify a multi-bit field. Use **Field[<pos>]** to specify a single-bit field.
RawData@<start>			Inserts raw data symbols at **<start>** byte position from the beginning of the DLLP. See **Packet = Raw** description for possible raw data formats.

Count	1: 65535	1	Repeats this packet by the number of times specified.

Example 1

This example shows how to send a **PM_Active_State_Request_L1** power management DLLP. This DLLP is sent 132 times.

The DLLP's **CRC** is calculated automatically since **CRC** is not specified.

```
Packet = DLLP {
DLLPType = PM_Active_State_Request_L1
Count = 132
}
```

Example 2

This example shows how to send a DLLP with an incorrect CRC.

```
Packet = DLLP {
DLLPType = PM_Enter_L1
CRC = 0x1234
}
```

Example 3

This example shows how to modify reserved fields in a DLLP using the **RawData** parameter. (See more on the **RawData** parameter.)

```
Packet = DLLP {
DLLPType = PM_Active_State_Request_L1
RawData@3 = ( D11.1, D11.2 )
}
```

Example 4

This example shows how to specify reserved fields in a DLLP using the **Field** parameter.

```
Packet = DLLP {
DLLPType = Ack
Field[8:19] = 0b101001000111
}
```

DLLPType Equals Ack, Nak

Parameter	Values	Default	Comment
AckNak_SeqNum	0:4095	0	Bytes 2-3 in the DLLP

Example 1
This example acknowledges all TLP packets with a sequence number less than or equal to 120 and initiates retransmission of TLP packets with a sequence number more than 120. The DLLP's **CRC** is calculated automatically since **CRC** is not specified.

```
Packet = DLLP {
DLLPType = Ack
AckNak_SeqNum = 120
}
```

DLLPType Equals InitFC1_p, InitFC1_np, InitFC1_cpl, InitFC2_p, InitFC2_np, InitFC2_cpl, UpdateFC_p, UpdateFC_np, UpdateFC_cpl

Parameter	Values	Default	Comment
VC_ID	0:7	0	Virtual Channel, bits 2:0 in the first byte of the DLLP
HdrFC	0:255	0	Contains the credit value for headers of the indicated type (P, NP, or Cpl)
DataFC	0:4095	0	Contains the credit value for payload Data of the indicated type (P, NP, or Cpl)

Example 1
The following example initializes credits for VC 0 for posted TLP requests.

Credit value for headers is 0. Credit value for data payload is infinite.

The DLLP's **CRC** is calculated automatically since **CRC** is not specified.

```
Packet = DLLP {
DLLPType = InitFc1_P
VC_ID = 0
HdrFC = 2
DataFC = 0
}
```

DLLPType Equals Vendor

Parameter	Values	Default	Comment
Data	0x000000:0xFFFFFF	0	Vendor specific data, bytes 1-3 in the DLLP

Example 1

```
Packet = DLLP {
DLLPType = Vendor
VendorSpecific = 0x010203
}
```

Packet = OrderedSet

Function: Initiates transmission of ordered set on the bus.

Parameter	**Values**	**Default**	**Comment**
SetType	TS1 TS2 FTS Pattern Idle Skip		
RawData@<start>			Inserts raw data symbols at **<start>** byte position from the beginning of the ordered set. See **Packet = Raw** description for possible raw data formats.
Count	1: 65535	1	Repeats this packet by the number of times specified.

Example

The following example sends 255 Fast Training Sequences:

```
Packet = OrderedSet {
SetType = FTS
Count = 255
}
```

SetType equals TS1, TS2

Parameter	**Values**	**Default**	**Comment**
LinkNumber	0:255 PAD	PAD	Link Number within component
LaneNumber	0:31 PAD	PAD	Lane Number within Port
N_FTS	0:255	0	The number of fast training ordered sets required by the Receiver to obtain reliable bit and Symbol lock.
TrainingControl	(X,X,X,X)	(0,0,0,0)	Training control bits. The order of the bits is: **(HotReset, DisableLink, Loopback, DisableScrambling)**
Identifier	(X,X,X…)	D10.2 for TS1 and D5.2 for TS2	Use the same format as in **Packet = Raw,** with exception of 10-bit codes

In x4, x8 or x16 configurations, the keys listed above apply to all lanes. When you want to specify parameters for a particular lane, use the following format:

```
<key>@<lane_number> = <value>
```

Example 1
The following example sends a TS1 ordered set.

N_FTS is equal to 255 for all lanes.

LinkNumber and **LaneNumber** are PADs (the default value) for all lanes.

TrainingControl bits are zeroes for all lanes.

Identifier symbols are
(D10.2 D10.2 D10.2 D10.2 D10.2 D10.2 D10.2 D10.2 D10.2 D10.2) for all lanes.

```
Packet = OrderedSet {
SetType = TS1
N_FTS = 255
}
```

Example 2
The following example sends a TS1 ordered set.

N_FTS is equal to 255 for all lanes.

LinkNumber is 0 for all lanes.

LaneNumber are 3, 2, 1, 0 for lanes 0, 1, 2, 3, and PADs for all other lanes.

TrainingControl bits are zeroes for all lanes.

Identifier symbols are
(D10.2 D10.2 D10.2 D10.2 D10.2 D10.2 D10.2 D10.2 D10.2 D10.2) for all lanes.

```
Packet = OrderedSet {
SetType = TS1
LinkNumber = 0
LaneNumber@0 = 3
LaneNumber@1 = 2
LaneNumber@2 = 1
LaneNumber@3 = 0
N_FTS = 255
}
```

Example 3
The following example sends a TS2 ordered set.

N_FTS is equal to 255 for all lanes.

LinkNumber and **LaneNumber** are PADs (the default value) for all lanes.

TrainingControl's Disable Scrambling bit is asserted on all lanes. All other **TrainingControl** bits are de-asserted.

Identifier symbols are
(D10.2 D10.2 D10.2 D10.2 D10.2 D10.2 D10.2 D10.2 D10.2 D10.2) for all lanes.

```
Packet = OrderedSet {
SetType = TS1
N_FTS = 255
TrainingControl = (0,0,0,1)
}
```

Example 4
The following example sends a TS2 ordered set.

N_FTS is equal to 255 for all lanes.

LinkNumber and **LaneNumber** are PADs (the default value) for all lanes.

All **TrainingControl** bits are de-asserted.

Identifier symbols are
(D5.2, D5.2, D5.2, D5.2, D5.2, D5.2, D5.2, D5.2, D5.1, D5.2) for lane 2.

Identifier symbols are
(D5.2, D5.2, D5.2, D5.2, D5.2, D5.2, D5.2, D5.2, D5.2, D5.2) for all other lanes.

This sends a corrupted TS2 ordered set, since the **Identifier** is incorrect for lane 2.

```
Packet = OrderedSet {
SetType = TS2
N_FTS = 255
Identifier@2 = (D5.2, D5.2, D5.2, D5.2, D5.2, D5.2, D5.2,
D5.2, D5.1, D5.2)
}
```

SetType Equals Skip

Parameter	Values	Default	Comment
SkipCount	0:5	3	Number of SKIP symbols to send after COMMA

Example 1

This example sends a **Skip** ordered set. Comma followed by 3 **SKIP** symbols is sent on each lane.

```
Packet = OrderedSet {
SetType = Skip
}
```

Example 2

This example sends a **Skip** ordered set. Comma followed by 2 **SKIP** symbols is sent on each lane.

```
Packet = OrderedSet {
SetType = Skip
SkipCount = 2
}
```

Packet = Raw

Function: Initiates transmission of raw data on the bus.

Parameter	Values	Default	Comment
RawData	(X,X,X…)		Specifies the array of bytes or 10-bit symbols to send.
Count	1: 65535	1	Repeats packet specified number of times.

The elements of data can be specified in the following formats:

Symbols:

```
Packet = Raw
{
RawData = ( K28.5, D21.5, K28.5, D10.2 )
}
```

Bytes in hexadecimal format with preceding K/D modifier:

```
Packet = Raw
{
RawData = ( KBC, DB5, KBC, D4A )
}
```

In addition to generate fully qualified 10 bit symbols, you can specify running disparity sign for each symbol:

```
Packet = Raw
{
RawData = ( K28.5+, D21.5-, K28.5-, D10.2- )
}
```

Specify 10-bit symbols in binary, hex, or decimal format:

```
Packet = Raw
{
RawData = ( 0b0011111010, 0b1100111001, 0b0011111010,
0b1110000110 )
}
```

Packet = <TemplateName>

This command initiates transmission of the packet specified by the **Template** command.

User can override packet fields according to the template.

Example 1

This sequence issues three 32-bit Memory read requests. The address field of TLP header accepts the values 0, 64, and 128. Every other field in the TLP header accepts the value from the packet template:

```
Template = TLP {
Name = “TestPacket”
Type = MRd32
RequesterID = (1:0:0)
Length = 64
Address = 0
}
Packet = “TestPacket”
{
}
Packet = “TestPacket”
{
Address = 64
}
Packet = “TestPacket”
{
Address = 128
}
```

12.4 Idle Command

This command sends idle symbols (D0.0) for the time specified.

Example

The following example sends two TLP packets separated by D0.0 symbols. The idle time between those TLP packets is 64 ns. Eight D0.0 symbols are sent between TLP packets on each lane.

```
Packet = TLP {
TLPType = MRd32
Address = 0x1000
}
Idle = 64
Packet = TLP {
TLPType = MRd32
Address = 0x1000
}
```

12.5 Link Command

All of these commands, with the exception of **Link = InitFC**, are controls to the Link Training and Status State Machine (also known as the **LTSSM**). These commands are issued to the LTSSM to steer it to a particular state.

This is not a means to force the Link state to a particular value. For instance, if the Script contains the **Link = L0** command, it is a request to bring the link to the L0 state. The LTSSM is responsible for managing all of the link training and all of the intermediate link states to accomplish this.

Command	Description
Link = L0	Transitions link to L0 state.
Link = L1	Transitions the link into the L1 (low power) state. Applies only in L0 state.
Link = L0s	Transitions the link into the L0s (low power) state. Applies only in L0 state.
Link = Disabled	Tells the LTSSM to move into the Disabled State. To get to this state, the LTSSM must either be in the Configuration State or the Recovery State. If the link is currently in the Detect state, and the **Link=Disabled** command is issued, it goes to Configuration first and then goes directly to Disabled. Once in the Disabled state, the LTSSM sends 16 TS1's with the Disable Link modifier bit set, followed by an electrical Idle ordered set, followed by electrical idle. To exit the Disabled state, simply set **Link=Detect** or **Link=L0**.
Link = HotReset	Tells the LTSSM to move into the HotReset State. To get to this state, the LTSSM must first be in the Recovery state. Once in the HotReset State, the LTSSM sends TS1 ordered sets with the HotReset modifier bit set. The LTSSM then goes to the Detect state automatically after 2 ms.
Link = Recovery	Transitions the link into the Recovery state. Applies only in L0, L0S, or L1 States.
Link = Detect	Tells the PE*Trainer* to immediately bring the Link down. In this state, the LTSSM drives all of the PCI Express lanes to electrical idle. Before the lanes go electrical idle, a single electrical idle ordered set is transmitted. Applies while in any state.
Link = LTSSMOff	Disables the LTSSM. This essentially means that the PE*Trainer* is not responsible for managing the link state. Instead, the user is free to transmit ordered sets, DLLP's and RAW packets blindly.
Link = InitFC	Starts the flow control initialization state machine.
Link = PERST	Sends a PERST# signal for the period specified.

Link = PERST

Parameter	Values	Default	Comment
Duration	In ns (rounded to nearest 8)	1000	Duration of the PERST# signal

12.6 Config Command

This command configures the PE*Trainer*.

Config = General

This command should precede any statement in a PE*Trainer* script file. There should be only one **Config = General** command in a PE*Trainer* script file. All **Config = General** commands from included files are ignored.

Parameter	**Values**	**Default**	**Comment**
AutoDetect	Yes No	No	Automatically detect link parameters
LinkWidth	1 4 8 16	4	Ignored in PE*Trainer* ML if AutoDetect is set.
DirectionRx	Upstream Downstream	U	
DisableScrambleTx	Yes No	No	Ignored if AutoDetect is set
DisableDescrambleRx	Yes No	No	Ignored if AutoDetect is set
ReverseLanesTx	Yes No	No	Ignored if AutoDetect is set
ReverseLanesRx	Yes No	No	Ignored if AutoDetect is set
InvertPolarityTx	(X,X,X,X,…)		The array of 1/0 elements. The size of the array should match the link width.
InvertPolarityRx	(X,X,X,X,…)		The array of 1/0 elements. The size of the array should match the link width. Ignored if AutoDetect is set.
BaseSpec10	Yes No	No	

SkewTx	(X,X,X,X,…)		The array of integer elements. The size of the array should match the link width. Measured in symbols, valid values are from 0 to 7
UseExtRefClock	Yes No	No	Use external reference clock. Applicable for PE*Trainer* ML only.
TrainerReset	Yes No	No	When set, resets PE*Trainer* before script execution.

Example 1

The following example configures PE*Trainer* to generate traffic on an x4 link **(LinkWidth = 4)** as a host emulator **(DirectionRx = Upstream)** and invert polarity on the first two lanes on incoming traffic **(InvertPolarityRx = (1,1,0,0))**.

The PE*Trainer* is reset before script execution **(TrainerReset = Yes)**.

All options that are not specified **(DisableScrambleTx, DisableDescrambleRx, ReverseLanesTx, ReverseLanesRx, InvertPolarityTx, BaseSpec10, SkewTx**, and **UseExtRefClock)** are taken from the Generation Options dialog.

```
Config = General {
LinkWidth = 4
DirectionRx = Upstream
InvertPolarityRx = (1,1,0,0)
TrainerReset = Yes
}
```

Example 2

The following example configures PE*Trainer* to generate traffic on x8 link **(LinkWidth = 8)** as a device emulator **(DirectionRx = Downstream)**.

Outgoing lanes are reversed **(ReverseLanesTx = Yes)**.

Polarity on the last four outgoing lanes on outgoing traffic is inverted **(InvertPolarityTx = (0,0,0,0,1,1,1,1))**.

Lanes 0 and 4 have a skew value of 1 symbol time.

PE*Trainer* is reset before script execution **(TrainerReset = Yes)**.

```
Config = General
{
LinkWidth = 8
DirectionRx = Downstream
SkewTx = (1,0,0,0,1,0,0,0)
InvertPolarityTx = ( 0,0,0,0,1,1,1,1 )
ReverseLanesTx = Yes
TrainerReset = Yes
}
```

Config = FCTx

This command allows user to specify the policy for TLP transmission in regard of received Flow Control DLLP packets.

Parameter	Values	Default	Comment
CareForFC	Yes No	Yes	When not set, the TLP packets are being sent without regard of how many credits are available

Example

In this example we are turning off Flow Control checking for outgoing TLP packets. The TLP packets that are declared after this **Config = FCTx** command are sent without checking for available FC credits.

```
Config = FCTx {
CareForFC = No
}

Packet = TLP {
TLPType = CfgRd0
Length = 1
Register = 0
Count = 10000
}
```

Config = FCRx

This command configures automatic **UpdateFC** DLLP generation.

Parameter	Values	Default	Comment
Timer	In ns (rounded to nearest 8) Off	4200	Periodic timer that controls the sending of **UpdateFC** DLLP packets
PH	0:255	1	Posted Request Headers
NPH	0:255	1	Non-Posted Request Headers
CplH	0:255	1	Completion Headers
PD	0:4095	1024	Posted Request Data Payload
NPD	0:4095	1	Non-Posted Request Data Payload
CplD	0:4095	1024	Completion Data Payload

Example

In this example, we specify the timer for sending **Update FC** DLLP packets. Also we specify the initial number of FC credits for headers to advertise. The default values are used for data credits.

```
Config = FCRx {
Timer = 4000; send UpdateFC DLLP packets every 4000ns
PH = 1; 1 credit for Posted Request Headers
NPH = 2; 2 credits for Non-Posted Request Headers
CplH = 0; Infinite number of credits for Completion Headers
}
```

Config = TLP

This command facilitates data integrity control.

Parameter	Values	Default	Comment
AutoSeqNumber	Yes No	Yes	If set to 0, overrides automatic generation of the TLP sequence number and uses the user-defined value as set in the **Packet = TLP** command.
AutoLCRC	Yes No	Yes	If set to 0, overrides automatic generation of the TLP LCRC and uses the user-defined value as set in the **Packet = TLP** command.
ReplayTimer	In ns (rounded to nearest 8) Off	4200	Timeout in the TLP transmitter path that counts time since last **Ack** or **Nak** DLLP is received. If set, automatically retransmit TLP packets that were **Nak'**ed or on replay timer expiration.
AutoRetrain	Yes No	Yes	If set, enable automatic retraining of the link in case the number of retransmitted TLPs is 4. Applicable only when the **ReplayTimer** is not turned off.
TagGeneration	Manual Default Extended Phantom1 Phantom2 Phantom3	Manual	Tag generation policy for posted TLP packets: **Manual**: Tags are taken from the script. **Default**: Use lower 5-bits of Tag field. Zero out higher 3 bits. **Extended**: Use 8-bits of Tag field. **Phantom1**: Use 1 most significant bit of the Function field and 8-bits of Tag. **Phantom2**: Use 2 most significant bits of the Function field and 8-bits of Tag. **Phantom3**: Use 3 bits of Function field and 8-bits of Tag.

Example

This example shows how to turn off automatic PSN and LCRC generation for outgoing TLP packets. The **ReplayTimer, AutoRetrain,** and **TagGeneration** parameters are omitted so the default values are used.

```
Config = TLP {
AutoSeqNumber = No
AutoLCRC = No
}
```

Config = AckNak

Parameter	Values	Default	Comment
AckNak	Auto Ack Nak Disable	Auto	**Auto**: Automatic Ack/Nak **Ack**: Always Ack **Nak**: Always Nak **Disable**: Disable automatic Ack/Nak DLLP generation.
Delay	In ns (rounded to nearest 8)	0	Timer that controls how much delay is added to Ack/Nak DLLP response after TLP reception. Valid if **AckNak** is set to **Auto, Ack,** or **Nak.**

Example

This example shows how to configure the PE*Trainer* so it **Nak**'s each incoming TLP packet.

```
Config = AckNak {
AckNak = Nak
}
```

Config = Transactions

This command determines the behavior of PE*Trainer* as it responds to Memory, Configuration, and IO TLP requests. So that it properly responds to Memory and IO TLP requests, Configuration Address Space must be defined.

Parameter	Values	Default	Comment
AutoCfgCompletion	Yes No	No	If set, automatically handles Configuration Read and Write TLP transactions. For a Configuration Read transaction, Completion TLP contains the data read from the internal Configuration Space according to specified register address. For a Configuration Write transaction, internal Configuration Space is updated at the address with the data taken from Configuration Write TLP, and a Configuration Write Completion is returned.
AutoMemIoCompletion	Yes No	No	If set, automatically handles Memory and IO Read/Write TLP transactions. For Memory and IO Read transactions, a Completion TLP contains the data read from the internal Memory/IO Address Space according to specified address. For Memory and IO Write transactions, internal Memory/IO Address Space is updated at the address with the data taken from the TLP. (PE*Trainer* EML only)
EnableUR	Yes No	No	If set, enables Unsupported Request (UR) status for Memory/IO completions. **AutoMemIoCompletion** must be set to enable UR completions. (PE*Trainer* EML only)
EnableCA	Yes No	No	If set, enables Completer Abort (CA) status for Memory/IO completions. **AutoMemIoCompletion** must be set to enable CA completions. (PE*Trainer* EML only)
Poisoned	Yes No	No	If set, all Memory/IO completions have the **Poisoned** bit set. (PE*Trainer* EML only)

Example

This example enables automatic completion for Configuration TLP requests. In order to automatically complete Configuration TLP requests, the Configuration Space must be configured first.

```
Config = Transactions {
AutoCfgCompletion = Yes; Automatically complete
Configuration TLP requests
}
```

Note: After this command, automatic completion for Memory and I/O TLP requests are turned off, since the default value **(No)** is used for **AutoMemIoCompletion** parameter.

Config = Link

Parameter	Values	Default	Comment
FTSCount	0:255	255	Number of FTS ordered sets required (as sent in TS)
ExtendedSynch	Yes No	Yes	When set, forces the transmission of 4096 FTS ordered sets
SkipTimer	In ns (rounded to nearest 8) Off	4720	Periodic timer that controls sending of SKIP ordered sets at specific intervals. Timer's value is measured in 1 us units.

Example

This example configures the number of Fast Training Sequences to send when transitioning from L0s state. This number is also be advertised during Link Training. This command also configures the periodic timer for SKIP Ordered Sets – sent every 4700 ns.

```
Config = Link
{
SkipTimer = 4700
FTSCount = 255
}
```

Config = Definitions

Parameter	Values	Default	Comment
Any literal	Any integer, string, array or predefined value		The defined values can be used anywhere in the script where as a parameter value.

Example 1

```
Config = Definitions {
my_register = 0x24
my_tlptype = CfgWr0
my_payload = ( 0x12345678 )
my_wait_message = “my wait”
}

Packet = TLP {
PSN = Incr
TlpType = my_tlptype
Register = my_register
Payload = my_payload
}

Config = Definitions {
my_register = 0x20
my_tlptype = CfgWr1
}

Packet = TLP {
PSN = Incr
TlpType = my_tlptype
Register = my_register
Payload = my_payload
}

wait = my_wait_message
```

Example 2

This example shows how to use definitions in the expressions and how to redefine the values.

```
Config = Definitions {
READ_START = 0x10
}

; Repeat 10 times
Repeat = Begin {
Count=10
Counter = i
}; send TLP using repeat counter (i) and READ_START to
specify the address
Packet = TLP {
TLPType = CfgRd0
Register = ( READ_START + ( 4 << i ) )
}

Repeat=End

; redefine READ_START, now READ_START is 0x40
Config = Definitions
{
READ_START = ( READ_START + 0x30 )
}

; send TLP using READ_START to specify the address
Packet = TLP {
TLPType = CfgRd0
Register = READ_START
}
```

12.7 Wait Command

This command yields script execution until condition specified is true or timeout expires.

Parameter	Values	Default	Comment
Timeout		0	Timeout in nanoseconds. **0** means infinite timeout.
Display	Any string literal		Message that is displayed during the waiting in status bar
Count	1: 65535	1	Repeats wait specified number of times

Wait = TLP

This command waits for a TLP that matches the defined condition. Only TLP Header fields can be specified. All parameters from **Packet = TLP** command are valid except **PSN, ECRC, LCRC,** and **Payload** parameters.

TLP Header fields can be masked using the following format:

- `0x0XAXX`: For hexadecimal values
- `0b0001XX`: For binary values

Example

This command waits infinitely for a Configuration Write request to registers from 0x1000 to 0x1FFF.

```
Wait = TLP {
TLPType = CfgWr
Register = “0x1XXX”
Timeout = 0
}
```

Wait = DLLP

This command waits for a DLLP that matches the defined condition. All parameters from **Packet = DLLP** command are valid except the **CRC** field.

DLLP fields can be masked using the following format:

- `0x0XAXX`: For hexadecimal values
- `0b0001XX`: For binary values

Example 1

This command waits for Ack DLLP. The execution continues when Ack DLLP is received or after 256 ns timeout expires.

```
Wait = DLLP {
DLLPType = Ack
Timeout = 256
}
```

Example 2
This command waits for a Vendor DLLP with the Least Significant Bit of the vendor specific data set. The execution continues when such DLLP is received or after the 256 ns timeout expires.

```
Wait = DLLP {
DLLPType = Vendor
VendorSpecific = "0bXXXXXXXXXXXXXXXXXXXXXXX1"
Timeout = 256
}
```

Wait = Error

Parameter	Values	Default	Comment
Errors	DLLPCRC TLPLCRC Delimiter Disparity Symbol IdleData SkipLate OrdSetFormat EndBadPacket		The list of errors to wait for. If not specified, this waits for any error

Example
This command waits for a **Delimiter, Disparity**, or **Symbol** error to occur in incoming traffic. The script continues running when any of the specified errors occur or after the 1024 ns timeout expires.

```
Wait = Error {
Errors = (Delimiter, Disparity, Symbol)
Timeout = 1024
}
```

Wait = LinkCondition

Parameter	Values	Default	Comment
Conditions	SKIP IDLE TS1 TS2 FTS PATN DLLP TLP COMMA		List of conditions to wait for.
TrainingControl	(X,X,X,X)		(PE*Trainer* EML only). Training control bits. The order of the bits is as follows: **(HotReset, DisableLink, Loopback, DisableScrambling)**

Example 1
This command waits for the COMMA symbol in incoming traffic. The script execution continues when the COMMA symbol is received or after the 1024 ns timeout expires.

```
Wait = LinkCondition {
Conditions = ( COMMA )
Timeout = 1024
}
```

Example 2
This command waits for a Training Sequence Ordered Set (TS1 or TS2) in incoming traffic with the **HotReset** bit asserted in the **TrainingControl** bits. The script execution also continues after the 1024 ns timeout expires.

```
Wait = LinkCondition {
Conditions = ( TS1, TS2 )
TrainingControl = ( 1, 0, 0, 0 )
Timeout = 1024
}
```

Wait = BOB

This command waits for Breakout Board data match.

Parameter	Values	Default	Comment
Data			Mask and Match four bits of Breakout Board data

Example

This command waits for Breakout Board data with the Least Significant Bit and Most Significant Bit set.

```
Wait = BOB {
Data = “0b1XX1”
}
```

Wait = Payload

This command waits for TLP payload match.

Parameter	Values	Default	Comment
Data			Mask and Match up to four DWORDs of TLP payload PE*Trainer* ML: Any offset from the beginning of payload PE*Trainer* EML: Zero offset from the beginning of payload
Data@<offset>			Mask and Match up to four DWORDs of TLP payload starting from **<offset>** offset from the beginning of payload (PE*Trainer* EML only)

Up to four DWORDs of the payload can be specified.

Example 1
This command waits for a TLP with data payload **0x12345678**.

Notes:
When this command is executed on PE*Trainer* EML, it matches only the first DWORD of the TLP payload.
When this command is executed on PE*Trainer* ML, it matches any DWORD from the TLP payload.
Script execution continues when a TLP with the specified payload is received or after the 1024 ns timeout expires.

```
Wait = Payload {
Payload = ( 0x12345678 )
Timeout = 1024
}
```

Example 2
This command waits for a TLP with a data payload that matches the following criteria:

- 1st DWORD's upper-most word must have **0xABCD**.
- 4th DWORD's lowest word must have **0x1234**.
- 2nd and 3rd DWORDs are insignificant.

Only the first four DWORDs of a TLP payload are checked when this command is executed on PE*Trainer* EML.

Any four subsequent DWORDs of a TLP payload are checked when this command is executed on PE*Trainer* ML.

Script execution continues when a TLP with specified payload is received or after the 1024 ns timeout expires.

```
Wait = Payload {
Payload = ( 0xABCDXXXX, 0xXXXXXXXX, 0xXXXXXXXX, 0xXXXX1234 )
Timeout = 1024
}
```

Example 3
The following example can be executed only on PE*Trainer* EML. This command waits for a TLP with a data payload that matches the following criteria:

- 3rd DWORD's upper-most word must have **0xABCD**.
- 9th DWORD's lowest word must have **0x1234**.
- 10th DWORD's upper-most byte must have **0x56**.

Script execution continues when a TLP with specified payload is received or after the 1024 ns timeout expires.

```
Wait = Payload {
Payload@2 = ( 0xABCDXXXX )
Payload@8 = ( 0xXXXX1234, 0x56XXXXXX )
Timeout = 1024
}
```

Wait = User

This command waits for user input. The script execution continues when user resumes the script from PE*Tracer* software UI.

Example

This example pauses the script execution and displays the message to the user.

```
Wait = User {
Display = “Now you can continue”
}
```

Wait = <number>

Unconditionally yields script execution for specified number of nanoseconds. Example:

```
Wait = 500
```

Wait = <Text>

Equivalent to:

```
Wait = User {
Display = <Text>
}
```

Example:

```
Wait = “Press the button to continue script execution”
```

A count parameter can be applied to this command, which causes it to wait for that number of clicks on the user input button.

12.8 Branch Command

This command enables/disables interrupt for the condition specified.

Branch = <condition>

This command enables the interrupt for the condition specified.

The conditions are the same as in **Wait** command, except for **User.**

The parameter list is the same as the **Wait** command, except for the **Timeout, Display**, and **Count** parameters.

Here is a list of additional parameters for **Branch = <Condition>** command.

Parameter	Values	Default	Comment
BranchName	Any string literal		Name of the branch. Must be specified if this branch is to be disabled later.
ProcName	Any string literal		Name of the procedure to execute when branch conditions are met

ProcName parameter is mandatory.

BranchName parameter could be omitted if you do not plan to disable the branch later in the script.

The procedure that handles the branch condition must be defined before the **Branch = <Condition>** command.

Example:

```
...

Proc = Begin {
  ProcName = “Procedure1”
}

...

Proc = End

; the following statement specifies that if
; Delimiter, Disparity or Symbol error occurs, then the
; code declared in “Procedure1” should be executed

Branch = Error {
BranchName = “SomeErrorBranch”
ProcName = “Procedure1”
Errors = (Delimiter, Disparity, Symbol)
}

...

; disable the branch “SomeErrorBranch” specified above

Branch = Disable {
  BranchName = “SomeErrorBranch”
}

...
```

Branch = Disable

This command disables the interrupt that was previously enabled.

Parameter	**Values**	**Default**	**Comment**
BranchName	Any string literal		Name of the branch

Branch with the name specified in **BranchName** parameter must be defined.

12.9 Proc Command

This command declares the procedure to be executed for the **Branch** command. Procedure declaration must precede its usage in the **Branch** statement.

Proc=Begin

This command declares the start point of the procedure.

Parameter	Values	Default	Comment
ProcName	Any string literal		Name of the procedure

Proc=End

This command declares the end point of the procedure.

12.10 Loop Command

This command causes the PE*Trainer* BusEngine™ to re-execute a block of commands a predefined number of times.

Note: Loops require up to 1 us to branch to the beginning of the loop. During this time, script execution is paused. Internally generated packets, such as SKIP ordered sets, Ack/Nak DLLP packets, and flow control updates, still occur as programmed.

Loops can be nested up to four deep.

Loop=Begin

This command marks the beginning of the loop.

Parameter	Values	Default	Comment
Count	0:65535 Infinite		Specifies how many times to repeat the loop. Setting Count to 0 causes an infinite loop.

Loop=End

This command marks the end of the loop.

Example

```
Loop = Begin { count = 10 }
  Packet = TLP { TLPType = CfgRd0 Length = 1 Register = 0 }
Loop = End
```

12.11 Repeat Command

This command causes one or more commands to be repeated. This is not implemented as a branch instruction in the BusEngine™, but is a replication of commands during script compilation in the software. This allows back-to-back execution of these commands with as little as 0 symbol times of IDLE traffic between them.

This command increases the size of the script object that is downloaded to the PE*Trainer* and increases download time accordingly.

Repeat=Begin

This command marks the beginning of the code being repeated.

Parameter	Values	Default	Comment
Count	1:65535		Values of Infinite and 0 are not supported
Counter			

Counter Parameter

Any string literal can be used for the **Counter** parameter.

The value of the **Counter** parameter can be used within the **Repeat** statement (for example, between **Repeat=Begin** and **Repeat=End**) in arithmetic expressions for any parameter, except the parameters that require array data type (such as ***Payload*** for TLP packet).

The value of the **Counter** parameter changes from 0 to the value of the **Count** parameter minus one.

Arithmetic expressions must be included in round brackets (parentheses).

The operators are: +, -, *, /, <<, >>, &, |, ~.

Example 1

Within this repeat, **ppp** can be used in arithmetic expressions for any packet field. The value of **ppp** changes from 0 to 3 in the example. The **Tag** parameter accepts the values **0x10, 0x11, 0x12**, and **0x13**. The **AddressHi** parameter accepts the values **0x00400000, 0x00400001, 0x00400001**, and **0x00400002**.

```
Repeat = Begin { Count = 4 Counter = ppp }

Packet = TLP {
TLPType = MRd64
Tag = ( ppp + 0x10 )
AddressHi = ( 0x400000 + 4 / ( 5 - ppp ) )
}

Repeat = End
```

Example 2
The following example shows the usage of the counters in nested repeats. The counter **qqq** is used for the outer repeat. The counter **www** is used for the inner repeat.

Packet = TLP in the inner repeat uses both counters to construct the **AddressHi** parameter.

```
Repeat = Begin { Count = 3 Counter = qqq }

Packet = DLLP {
DLLPType = Ack
AckNak_SeqNum = ( qqq + 1 )
}

Packet = DLLP {
DLLPType = Ack
AckNak_SeqNum = ( 0xf & ~qqq )
}

Repeat = Begin { Count=4 Counter = www }

Packet = TLP {
TLPType = MRd64
AddressHi = ( 0x400000 + www * 4 + qqq ) )
}

Repeat = End

Repeat = End
```

Repeat=End

This command marks the end of the code being repeated.

Example

```
Repeat = Begin { count = 10 }
  Packet = TLP { TLPType = CfgRd0 length = 1 register = 0 }
Repeat = End
```

12.12 Template Command

This command creates a template for a packet that can be used in the **Packet** command. The fields specified in the **Template** command may be overridden in the **Packet** command.

Example 1

The following example issues three Memory Read requests.

```
Template = TLP {
Name = “TestPacket”
Type = MRd32
TC = 0
Tag = 0
RequesterID = (1:0:0)
Length = 64
Address = 0
}

Packet = “TestPacket” {
}

Packet = “TestPacket” {
Address = 64
}

Packet = “TestPacket” {
Address = 128
}
```

Example 2

The following example shows nested templates (when one template is based on another template).

```
; First define the template "SomeTlp3" for TLP packet
Template = TLP {
Name = "SomeTlp3"
TLPType = MRd32
RequesterID = (0:1:2)
Length = 0x40
LastDwBe = 0xF
FirstDwBe = 0xF
Address = 0x10000
}

; The template "SomeTlp4" is based on the template
"SomeTlp3"
; with Address overridden
Template = "SomeTlp3" {
Name = "SomeTlp4"
Address = 0x10040
}

; This TLP packet has Address parameter equal to 0x10000
Packet = "SomeTlp3" {
Length = 0x80
}

; This TLP packet has Address parameter equal to 0x10040
Packet = "SomeTlp4" {
Length = 0x80
}
```

12.13 Include Command

This command includes the PE*Trainer* script file inline. All commands in the included file are executed, with the exception of the **Config = General** command.

The format of this command is:

```
Include = <file_path>
```

where **file_path** is a path to the file to be included. If **file_path** is not a fully qualified path, then the relative path to the current script file is used.

Example 1

In this example, all commands from the **included1.peg** file are executed first, then all commands from the **included2.peg** file are executed, and then the 32-bit Memory Read TLP is sent.

```
Include = "included1.peg"
; All packets from included1.peg file are inserted here.
Include = "included2.peg"
; All packets from included2.peg file are inserted here.

Packet = TLP    ; Sending 32-bit Memory Read TLP request
{
TLPType = MRd32 ; Memory Read request (32 bit)
TC = 0x7        ; Traffic class is 7.
TD = 0x1        ; TLP digest is present.
EP = 0x0        ; TLP is not poisoned.
Address = 0x1000 ; Reading from address 1000h of memory space
Length = 0x40   ; Reading 40h DWORDs
}
```

Example 2

The first command of this example includes all commands from the file **c:/Testing/included1.peg**. If we assume that the current script is located in the folder **c:/Testing/TLP**, then the second command of this example includes all commands from the file **c:/Testing/TLP/included2.peg.** If we assume that the current script is located in the folder **c:/Testing/TLP**, then the third command of this example includes all commands from the file **c:/Testing/included3.peg**.

```
Include = "c:/Testing/included1.peg"
; All packets from included1.peg file are inserted here.
Include = "included2.peg"
; All packets from included2.peg file are inserted here.
Include = "../included3.peg"
; All packets from included3.peg file are inserted here.
```

12.14 AddressSpace Command

This command reads/writes the PE*Trainer* memory region.

PE*Trainer* maps Memory and IO address spaces to its internal memory region according to Base Address Registers (BAR) specified in the Configuration Address Space.

PE*Trainer* uses its memory regions when processing Memory, IO, and Configuration TLP requests.

PE*Trainer* maps Configuration address space to its internal memory region (**Cfg**).

PE*Trainer* supports one 64-bit Memory region, two 32-bit Memory regions, and two IO Memory regions.

Maximum address space sizes supported by PE*Trainer* are:

Address Space	**Size**
Configuration	4 KB
32-bit memory	128 MB
64-bit memory	512 MB
IO	256 MB

Mapping of BARs to PE*Trainer* memory regions:

Memory Region	**BAR**
Mem64	First BAR that defines 64-bit Memory Address Space
Mem32A	First BAR that defines 32-bit Memory Address Space
Mem32B	Second BAR that defines 64-bit Memory Address Space
IOA	First BAR that defines IO Address Space
IOB	Second BAR that defines IO Address Space

To properly respond to Memory and IO TLP requests, the Configuration space must be written to the PE*Trainer* first.

Mem64, Mem32A, Mem32B, IOA, and IOB memory regions are not implemented in PE*Trainer* ML.

AddressSpace = Read

This command reads specified memory region from PE*Trainer* and stores it in specified file.

Parameter	Values	Default	Comment
Location	Cfg Mem64 Mem32A Mem32B IOA IOB		Specifies the memory region to read from. The memory region is mapped to address space according to the rules described above. **Mem64, Mem32A, Mem32B, IOA**, and **IOB** are applicable to PE*Trainer* EML only.
Offset	Any number from 0 to the maximum allowed address determined by the memory region specified in the **Location** parameter	0	Specifies offset in bytes from the beginning of memory region specified in **Location** parameter.
Size	Any number from 0. The combination of **Offset** and **Size** parameters is limited by the maximum allowed address. (The maximum allowed address is determined by memory region specified in the **Location** parameter.)	Maximum allowed size for memory region specified in the **Location** parameter	Specifies number of bytes to read starting from address specified in the **Offset** parameter.
SaveTo	Any file path		File path to store the memory read.

Example 1

This command reads the whole Mem32A memory region and stores it in the **c:/mem.bin** file. The offset is 0. Read size is 128 MB.

```
AddressSpace = Read {
Location = Mem32A
SaveTo = “c:/mem.bin”
}
```

Example 2

This command reads 16 bytes from address **0x1000** of **Mem64** memory region and stores it in the **c:/mem.bin** file.

```
AddressSpace = Read {
Location = Mem64
Offset = 0x1000
Size = 0x10
SaveTo = “c:/mem.bin”
}
```

AddressSpace = Write

Writes specified memory region into PE*Trainer* from specified data source.

During write operations into the Mem64, Mem32A, Mem32B, IOA, and IOB regions, the automatic completions of Memory and IO TLP requests are disabled.

During write operations into the Cfg region, the automatic completions of Configuration TLP requests are disabled.

Parameter	Values	Default	Comment
Location	Cfg Mem64 Mem32A Mem32B IOA IOB		Specifies the memory region to write into. The memory region is mapped to address space according to the rules described above. **Mem64, Mem32A, Mem32B, IOA**, and **IOB** are applicable to PE*Trainer* EML only.
Offset	Any number from 0 to the maximum allowed address determined by the memory region specified in the **Location** parameter	0	Specifies offset in bytes from the beginning of memory region specified in **Location** parameter.
Size	Any number from 0. The combination of **Offset** and **Size** parameters is limited by the maximum allowed address. (The maximum allowed address is determined by memory region specified in **Location** parameter.)	If **Zeros, Ones, Random**, or **Incr** specified for the **LoadFrom** parameter, then the default value is the maximum allowed size for the memory region specified in the **Location** parameter. Otherwise, the default size is the size of data specified in the **LoadFrom** parameter.	Specifies number of bytes to write starting from address specified in the **Offset** parameter
LoadFrom	Any file path Any array of bytes Zeros Ones Random Incr	Zeros	

Example 1

This command clears the whole **Mem32A** memory region.

```
AddressSpace = Write {
Location = Mem32B
LoadFrom = Zeros
}
```

Example 2
This command writes 16 bytes, starting from address **0x1000**, into the **Mem64** memory region from file **c:/mem.bin**.

```
AddressSpace = Write {
Location = Mem64
Offset = 0x1000
Size = 0x10
LoadFrom = “c:/mem.bin”
}
```

Example 3
This command writes 7 bytes, starting from address **0x1000**, into the **Mem64** memory region from data specified.

```
AddressSpace = Write {
Location = Mem64
Offset = 0x1000
LoadFrom = ( 0x02, 0x08, 0x01, 0x03, 0x06, 0x07, 0x07 )
}
```

Example 4
This command writes 48 bytes of random data, starting from address **0x10**, into the **IOA** memory region.

```
AddressSpace = Write {
Location = IOA
Offset = 0x10
Size = 0x30
LoadFrom = Random
}
```

Chapter 13: Traffic Generation Macros

13.1 Macros

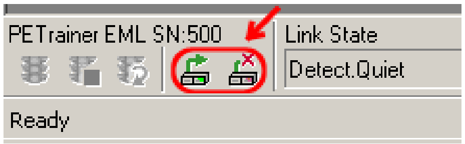
PE*Trainer*™ EML™ and PE*Trainer*™ ML allow users to add buttons to the Status bar at the bottom of the window (and add commands to the Generate menu) to run traffic generation macros on the Exerciser.

After a macro script has been defined and assigned to a button the on the Status bar, the macro can be run by clicking the macro button with the mouse or selecting the macro name from the Generate menu.

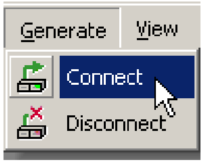
Default Macros: Connect and Disconnect

By default, the PE*Tracer*™ software includes two macros, **Connect** and **Disconnect**. These buttons execute macros for creating and breaking a connection between the Exerciser and a DUT.

The buttons are on the Status bar when Exerciser hardware is present.



The commands are also on the Generate menu:



Connect Macro

The default code for this macro is the following:

```
Config = General { TrainerReset = 1 }
Link = Detect
Wait = 500
Link = L0
Wait = 500
Link = InitFC
```

Disconnect Macro

The default code for this macro is the following:

```
Config = General { TrainerReset = 1 }
Link = Detect
```

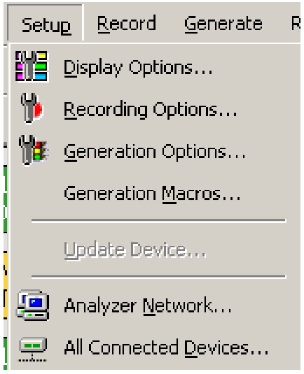
13.2 Adding New Script Macros

There are two ways to add script macros:

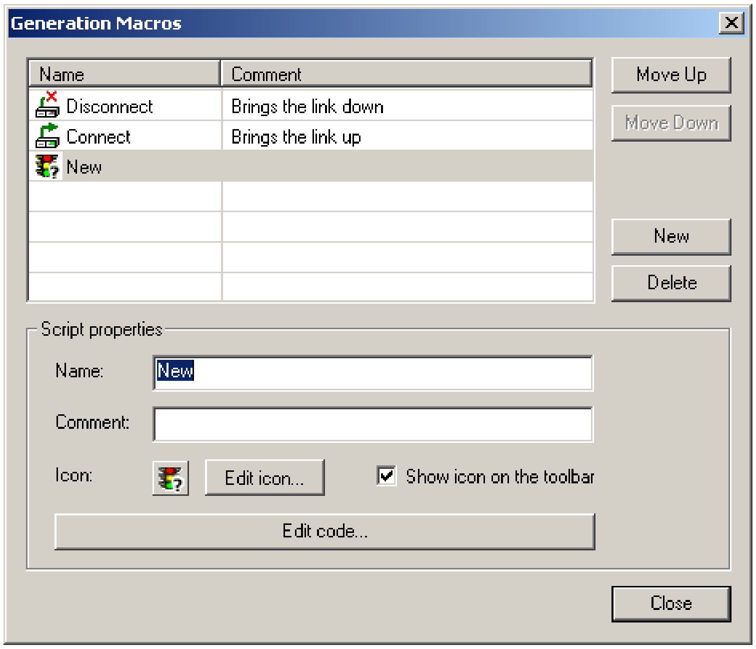
- Using the Generation Macros dialog
- Adding script files to the GenScriptMacros directory.

Using the Generation Macros Dialog

Step 1 Open the Generation Macros dialog by selecting **Setup > Generation Macros** from the menu.



Step 2 In the Generation Macros dialog box, click the **New** button.



Name	Comment
Disconnect	Brings the link down
Connect	Brings the link up
New	

Name: New

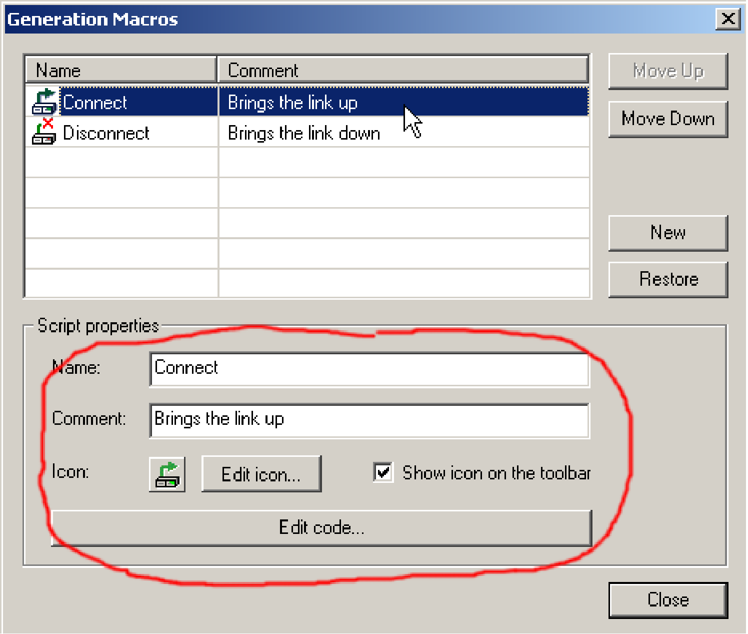
Comment:

Icon:

☑ Show icon on the toolbar

Step 3 Within **Script properties** area specify **script name, script comment, script icon,** and **whether or not to show script icon** on a toolbar.

Step 4 Create a new icon for the script by clicking **Edit Icon**... button. The Edit Generation Macro Icon dialog appears.



Name	Comment
Connect	Brings the link up
Disconnect	Brings the link down

Script properties

Name: Connect

Comment: Brings the link up

☑ Show icon on the toolbar

Step 5 Using the tools provided, paint the icon for new script macro and then press **OK**.

Step 6 Edit the script code by clicking the **Edit code**... button. The dialog closes and an empty script editing window appears (see Section 11.4, “Editing Generation Files with the Script Editor” on page 215).

Step 7 Type the script code and press the **Save** button. The script macro and icon are saved in the **GenScriptMacros** directory located under the PE*Tracer* directory (for example, **Program Files\CATC\PETracer\GenScriptMacros**).

Step 8 Close the script window. The new macro script button automatically appears on the Status bar at the bottom of the window. Clicking this button causes PE*Trainer* to execute the script.

Adding Script Files to the GenScriptMacros Directory

The second way to add a new PE*Trainer* script macro is to copy an existing PE*Trainer* script file in the **GenScriptMacros** directory.

Step 1 Copy a script file into the **GenScriptsMacros** directory located under the PE*Tracer* directory (for example, **Program Files\CATC\PETracer\GenScriptMacros**).

Step 2 Switch to the PE*Tracer* application. You see that a new icon has been automatically added for the script file to the Status bar at the bottom of the window. The default icon is assigned to the new script macro and the file name is used as a script name.

13.3 Modifying Script Macros

To modify a macro assigned to a button:

Step 1 Select **Setup > Generation Macro** from the menu.

The Generation Macros dialog opens for modifying, creating, and deleting macros:

Name: Name of Macro

Comment: Descriptive comment so you can remember what the macro does

Icon: Currently assigned button for the macro.

Show icon on the toolbar: If checked, places the icon on the Status bar.

Edit Code: Opens a dialog for editing the macro script.

Step 2 Select the macro to be modified.

Step 3 Within the **Script properties** area, modify the **script name, script comment, script icon,** and whether or not to show script icon

Step 4 To edit script code, press the **Edit code**... button. The Generation Macros dialog closes and the Script Editing window appears, showing current code for the selected script macro.

Note: If the Generation Script Editor pane does not appear, click 📝 .

Step 5 Modify the script code and press the **Save** button 💾 .

Step 6 Close the script window. The macro has now been modified.

13.4 Changing the Order of Macro Icons on the Status Bar

To change the order of script macro icons on the Status bar:

Step 1 Open Generation Macro Scripts dialog by selecting **Setup > Generation Macros**.

Step 2 Click the **Up** and **Down** buttons to change the order of the script macros.

Step 3 Close the dialog. The button order is changed.

13.5 Deleting User-Defined Script Macros

To delete a script macro:

Step 1 Open the Generation Macro Scripts dialog by selecting **Setup > Generation Macros**.

Step 2 Select the macro you want to delete and press the **Delete** button:

All deleted scripts and icons are removed from **GenScriptMacros** directory. A backup copy is stored in the **GenScriptMacros\Deleted** directory.

Note: You cannot delete the Connect and Disconnect script macros.

13.6 Restoring the Default Appearance of the Connect and Disconnect Icons

You can restore the default appearance of the Connect and Disconnect icons by clicking the **Restore** button.

Step 1 Open the Generation Macro Scripts dialog by selecting **Setup > Generation Macros**.

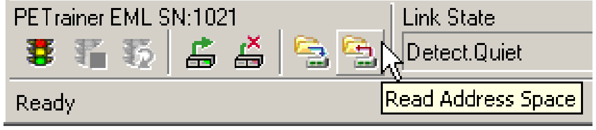
Step 2 Select the macro you want to restore and press the **Restore** button.

Chapter 14: Traffic Generation Address Space

14.1 Address Space Toolbar Buttons and Dialogs

On the PE*Trainer*™ toolbar at the bottom of the application window are two buttons:

- Write Address Space
- Read Address Space



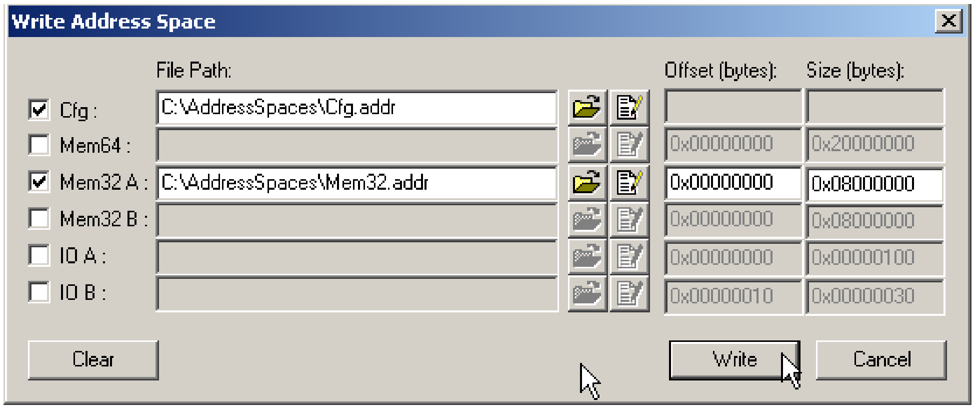
Clicking these buttons opens a dialog for selecting, creating, or editing Address Space settings used by PE*Trainer*. The dialog can also be accessed from the menu: **Generate > Write Address Space** and **Generate > Read Address Space**.

Write Address Space Button

The Write Address Space Button brings up one of two dialogs depending on which PE*Trainer* platform is running (PE*Trainer* ML™ or PE*Trainer* EML™).

Write Address Space Dialog for PE*Trainer* EML

On the PE*Trainer* EML platform, clicking the **Write Address Space** button opens the following dialog:



Write Address Space

	File Path:	Offset (bytes):	Size (bytes):
☑ Cfg :	C:\AddressSpaces\Cfg.addr		
□ Mem64 :		0x00000000	0x20000000
☑ Mem32 A :	C:\AddressSpaces\Mem32.addr	0x00000000	0x08000000
□ Mem32 B :		0x00000000	0x08000000
□ IO A :		0x00000000	0x00000100
□ IO B :		0x00000010	0x00000030

The dialog shows a list of address spaces. Check the checkboxes to select Address Spaces.

Note: PE*Trainer* memory regions are assigned to the specific BARs according to configuration space. See “AddressSpace Command” on page 289.

To select Address Space file(s), click the **Browse** button 📂 .

To edit a selected file, click the **Edit** button 📝 . This action opens either the Configuration Space Editor or the Memory/IO Address Space Editor depending on which option you have selected.

To specify the offset from the beginning of the PE*Trainer* memory region, enter a value into the **Offset (bytes)** box. The default offset is 0.

To set the size of the data to write, enter a value in the box marked **Size (bytes)**. The default size is the maximum allowed memory size (limited by PE*Trainer* hardware). The **offset+size** should not exceed the maximum allowed memory size (Mem64 = 512 MB, Mem32 = 128 MB, IO = 256 MB).

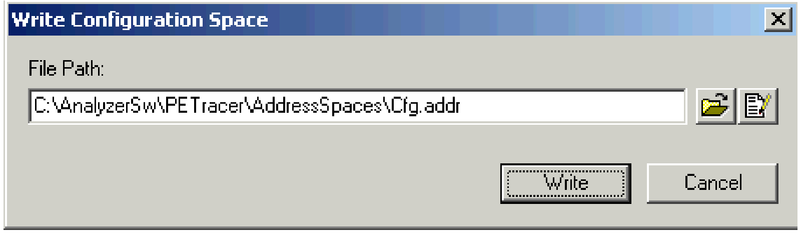
Note: You cannot specify Offset and Size for Configuration Address Space.

To load all selected memory regions into PE*Trainer* from the selected files, click the **Write** button.

To set all selected memory regions to 0, click the **Clear** button.

Write Address Space Dialog for PE*Trainer* ML

PE*Trainer* ML supports only the Configuration Address Space.



C:\AnalyzerSw\PETracer\AddressSpaces\Cfg.addr

You can open a browse dialog for selecting a configuration space file.

You can open the Configuration Space Editor for editing or creating configuration space files.

Note: The Configuration Address Space cannot be written to PE*Trainer* ML when a generation script is running.

Read Address Space Button

This Read Address Space Button brings up one of two dialogs depending on whether you are running (PE*Trainer* ML or PE*Trainer* EML).

PE*Trainer* EML Read Address Space Dialog

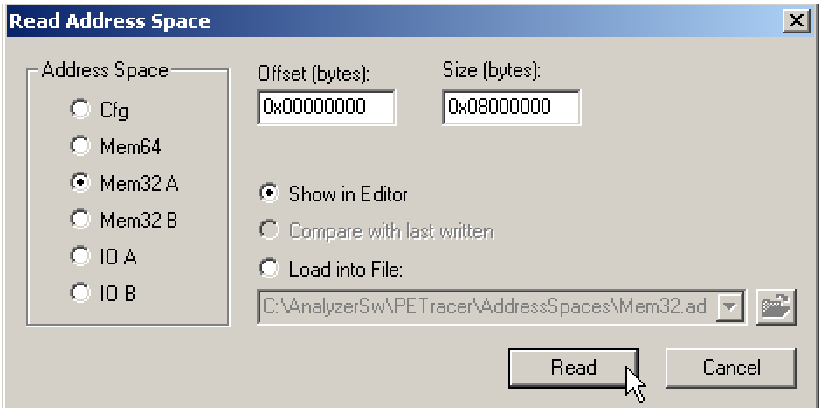
This dialog lets you specify the:

- Address space to be read
- Offset from the beginning of PE*Trainer* memory region
- Size of the data

You cannot specify the Offset and Size for the Configuration Address Space.

The data read can be processed in three ways:

- Displayed in the Configuration Space Editor or the Address Space Editor. To view the data in these editors, select the **Show in Editor** radio button option.



Read Address Space

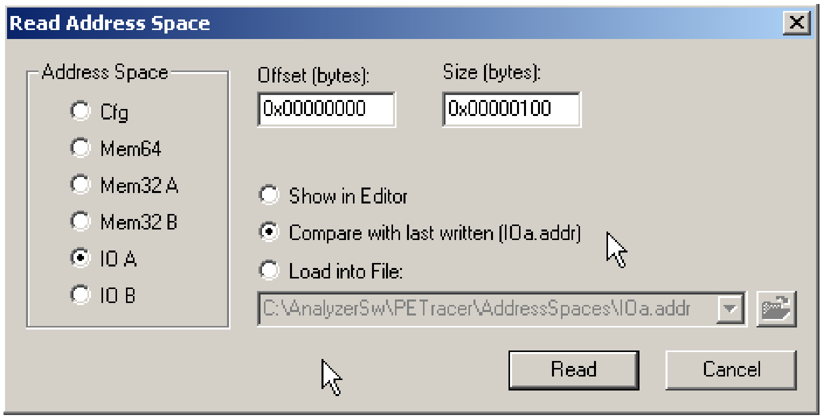
Address Space
○ Cfg
○ Mem64
● Mem32 A
○ Mem32 B
○ IO A
○ IO B

Offset (bytes): 0x00000000

Size (bytes): 0x08000000

● Show in Editor
○ Compare with last written
○ Load into File:
C:\AnalyzerSw\PETracer\AddressSpaces\Mem32.ad

- Stored in a file. To store the data into a file, select the **Load into File** option.
- Compared with previously written file. If you have written memory into the PE*Trainer* hardware as described above (Write Memory Address Space Button), then **Compare with last written** option is enabled.
Note: This option is not available for Configuration Address Space)



Read Address Space

Address Space
○ Cfg
○ Mem64
○ Mem32 A
○ Mem32 B
● IO A
○ IO B

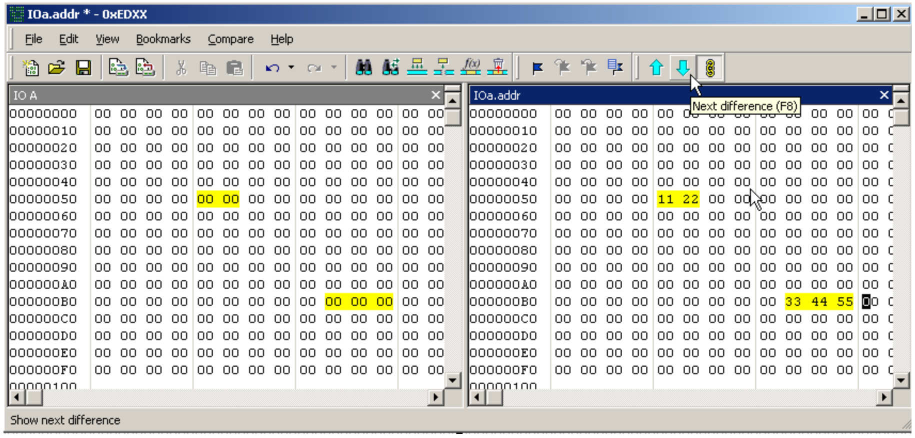
Offset (bytes): 0x00000000

Size (bytes): 0x00000100

○ Show in Editor
● Compare with last written (IOa.addr)
○ Load into File:
C:\AnalyzerSw\PETracer\AddressSpaces\IOa.addr

In this screenshot Compare with last written option is enabled because the **IO A** memory region was previously loaded from the **IOa.addr** file using Write Address Space functionality.

When you click the **Read** button, the memory is read from the address space, and both memory dumps (**IOa.addr** file and the currently read one) are displayed in the Memory/IO Space Editor:



IO A

```
00000000  00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000010  00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000020  00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000030  00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000040  00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000050  00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000060  00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000070  00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000080  00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000090  00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000100
```

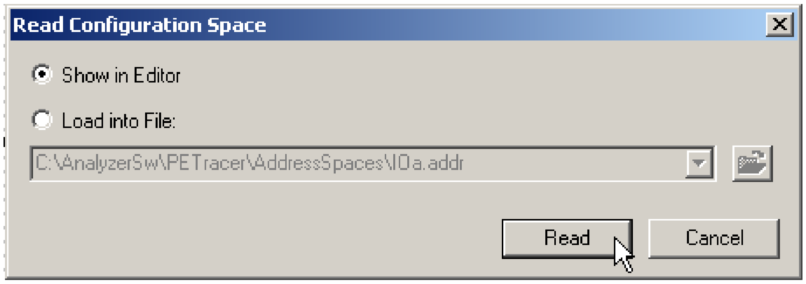
IOa.addr

```
00000000  00 00 00 00 00 0                00 00 00 C
00000010  00 00 00 00 00 00 00 00 00 00 00 00 00 C
00000020  00 00 00 00 00 00 00 00 00 00 00 00 00 C
00000030  00 00 00 00 00 00 00 00 00 00 00 00 00 C
00000040  00 00 00 00 00 00 00 00 00 00 00 00 00 C
00000050  00 00 00 00 11 22 00 00 00 00 00 00 00 C
00000060  00 00 00 00 00 00 00 00 00 00 00 00 00 C
00000070  00 00 00 00 00 00 00 00 00 00 00 00 00 C
00000080  00 00 00 00 00 00 00 00 00 00 00 00 00 C
00000090  00 00 00 00 00 00 00 00 00 00 00 00 00 C
000000A0  00 00 00 00 00 00 00 00 00 00 00 00 00 C
000000B0  00 00 00 00 00 00 00 00 00 33 44 55 00 C
000000C0  00 00 00 00 00 00 00 00 00 00 00 00 00 C
000000D0  00 00 00 00 00 00 00 00 00 00 00 00 00 C
000000E0  00 00 00 00 00 00 00 00 00 00 00 00 00 C
000000F0  00 00 00 00 00 00 00 00 00 00 00 00 00 C
00000100
```

All the differences are marked in **yellow**.

PE*Trainer* ML Read Address Space Dialog

PE*Trainer* ML supports only Configuration Address Space.



● Show in Editor

○ Load into File:

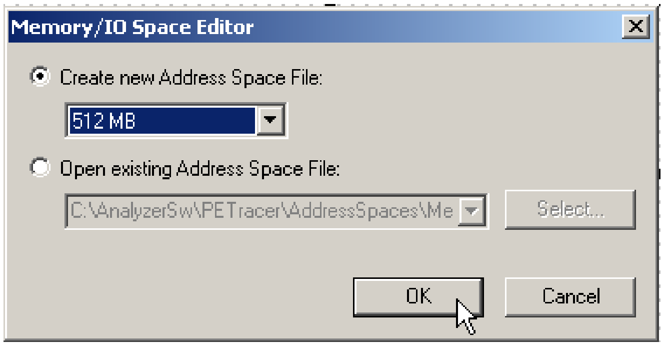
C:\AnalyzerSw\PETracer\AddressSpaces\IOa.addr

Note: The Configuration Address Space cannot be read from PE*Trainer* ML when a generation script is running.

14.2 Memory/IO Space Editor

The Memory/IO Space Editor lets users write address space memory into PE*Trainer*™ and read address space memory from PE*Trainer*.

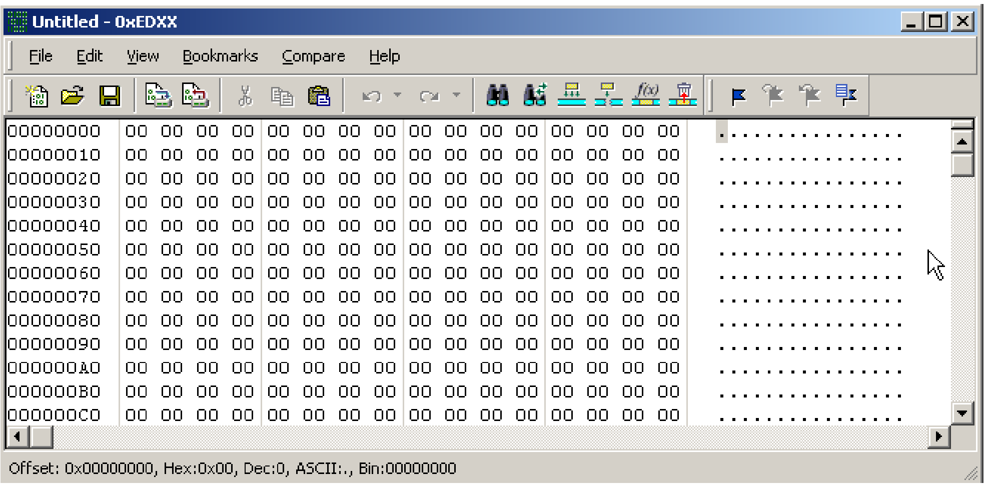
To access the Memory/IO Space Editor, select **Tools > Memory/IO Space Editor**. The following dialog opens:



● Create new Address Space File: 512 MB

○ Open existing Address Space File: C:\AnalyzerSw\PETracer\AddressSpaces\Me

- **Create new Address Space File**: Opens the Memory/IO Space Editor and displays an empty address space with the size specified.
- **Open existing Address Space File**: Activates the list of recently used Memory/IO Address Space files and enables the Select... button. After you have selected a file and then clicked **Select,** the Memory/IO Space Editor opens.



```
00000000  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000080  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00000090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
000000A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
000000B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
000000C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
```

Offset: 0x00000000, Hex:0x00, Dec:0, ASCII:., Bin:00000000

The Memory/IO Address Space editor is implemented as a standard hex editor.

The address space memory must be saved to a file so that it can be downloaded to the PE*Trainer* hardware using PE*Tracer*™ application.

If you prefer, you can write address space memory into PE*Trainer* and read address space memory from PE*Trainer* directly using the

and toolbar buttons or using the menu **File > Write Address Space** or **File > Read Address Space**.

Chapter 15: Traffic Generation Configuration Space

15.1 Configuration Read and Write

PE*Trainer*™ can automatically handle Configuration Read and Write TLP transactions:

- For a Configuration Read transaction, the Completion TLP contains the data read from the internal Configuration Space according to specified register address.
- For a Configuration Write transaction, the address for the internal Configuration Space is updated with the data taken from Configuration Write TLP, and Configuration Write Completion is returned.

You can specify the initial Configuration Space by editing the **AddressSpace=Write** command directly in the script or by using the UI provided in the Generator toolbar.

The PE*Tracer*™ software provides a special editor for editing this file called the Configuration Space Editor. The editor can be launched from a trace file, the Tools menu, or the Script Editor window (see “Launching the Configuration Space Editor” on page 308).

15.2 Configuration Space Editor

The Configuration Space Editor allows the Configuration Space to be modified on a field-by-field basis using hexadecimal of binary format.

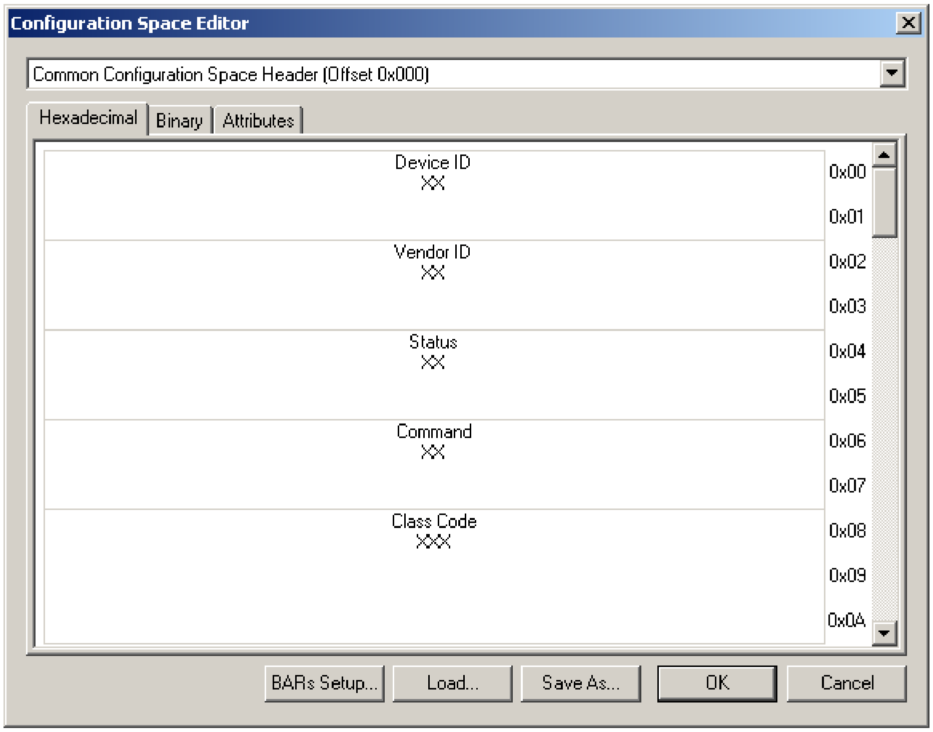
The editor supports PCI-compatible configuration space and PCI Express enhanced Configuration Space.

Export

The editor allows Configuration Space files to be exported from a PE*Tracer* trace.

Editing

When a new file is created, only the **PCI Configuration Space Header** and the first **Enhanced Capability Structure** are available for editing. When you define **Capabilities Pointers** (for PCI compatible configuration space) and/or **Next Offset** (for PCI Express enhanced configuration space,) additional Capability Structures appear in the list:



Common Configuration Space Header (Offset 0x000)

Device ID XX

Vendor ID XX

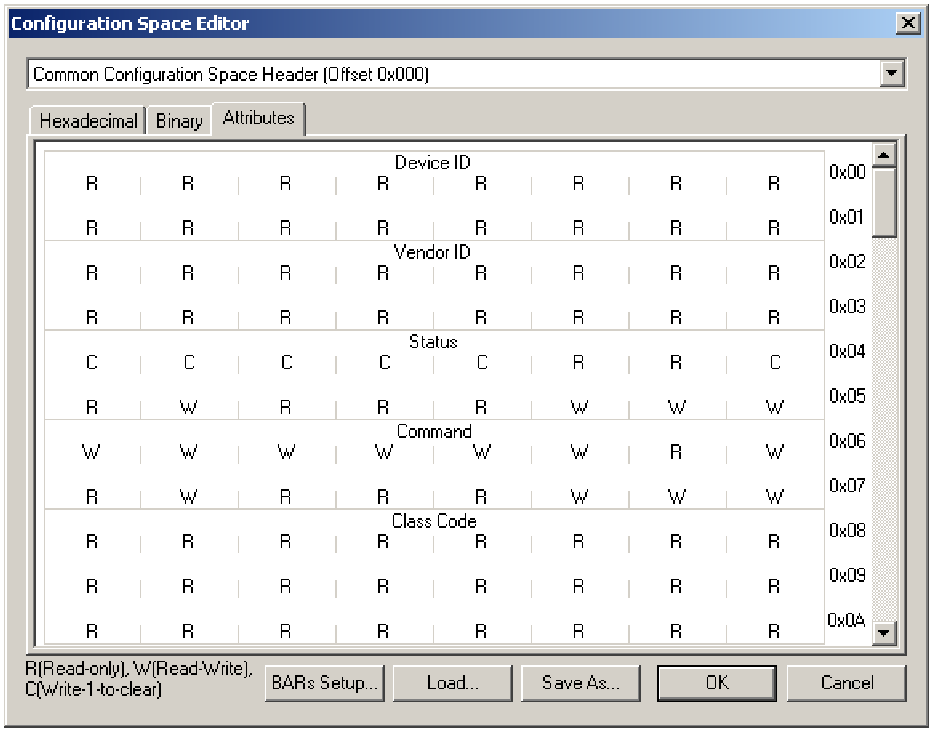
Status XX

Command XX

Class Code XXX

Attributes Page

The Attributes page lets users override the attributes for each bit of configuration space.



Common Configuration Space Header (Offset 0x000)

Device ID	R	R	R	R	R	R	R	R	0x00
	R	R	R	R	R	R	R	R	0x01
Vendor ID	R	R	R	R	R	R	R	R	0x02
	R	R	R	R	R	R	R	R	0x03
Status	C	C	C	C	C	R	R	C	0x04
	R	W	R	R	R	W	W	W	0x05
Command	W	W	W	W	W	W	R	W	0x06
	R	W	R	R	R	W	W	W	0x07
Class Code	R	R	R	R	R	R	R	R	0x08
	R	R	R	R	R	R	R	R	0x09
	R	R	R	R	R	R	R	R	0x0A

R(Read-only), W(Read-Write), C(Write-1-to-clear)

The options are:

- **R**: Read-only
- **C**: Write-1-to-clear
- **W**: Read-Write

By default, all attributes are set according to the PCI Express specification:

- **R**: RO, ROS
- **C**: RW1C, RW1CS
- **W**: all other register types

15.3 Launching the Configuration Space Editor

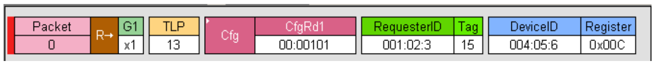
The Configuration Space dialog box displays read and write statuses for the Configuration Space of the currently selected device.

Launching from a Trace File Field

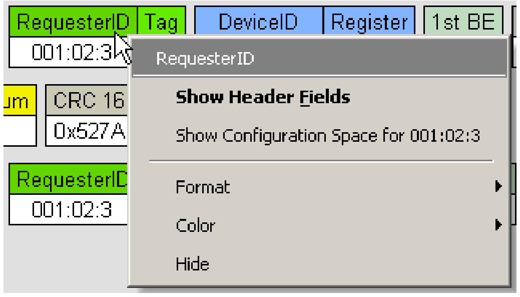
To access the Show Configuration Space dialog box from a trace file field:

Step 1 Open a trace, such as the sample file **cfg_pci_express.pex**.

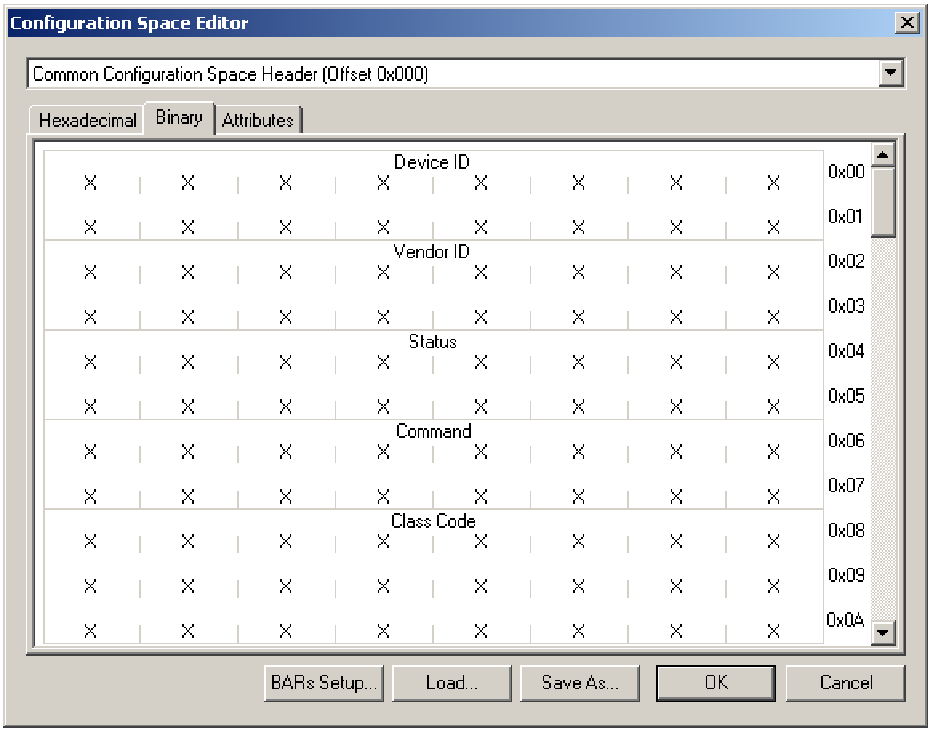
Step 2 Scroll to a **packet with a Configuration header**.



Step 3 Click a **RequesterID, CompeterID,** or **DeviceID** field. A pop-up menu opens.



Step 4 Select **Show Configuration Space for xxxx** from the menu, where **xxxx** is the device number. The Show Configuration Space dialog box opens.



Common Configuration Space Header (Offset 0x000)

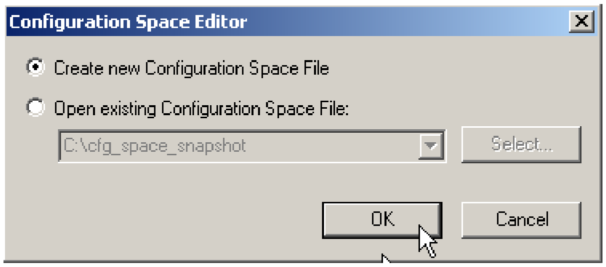
Device ID	X	X	X	X	X	X	X	X	0x00
	X	X	X	X	X	X	X	X	0x01
Vendor ID	X	X	X	X	X	X	X	X	0x02
	X	X	X	X	X	X	X	X	0x03
Status	X	X	X	X	X	X	X	X	0x04
	X	X	X	X	X	X	X	X	0x05
Command	X	X	X	X	X	X	X	X	0x06
	X	X	X	X	X	X	X	X	0x07
Class Code	X	X	X	X	X	X	X	X	0x08
	X	X	X	X	X	X	X	X	0x09
	X	X	X	X	X	X	X	X	0x0A

Step 5 The dialog box gives you several options:

- **Hexadecimal vs. Binary tabs**: Presents the data in different formats.
- **Combo box**: At top of dialog with a menu of Read and Write Configuration Spaces present in the current trace.
- **First, Prev, Next, and Last buttons**: For navigating to other Read and Write Configuration States in the trace.

Launching from the Tools Menu

Selecting **Tools > Configuration Space Editor** displays the following dialog:



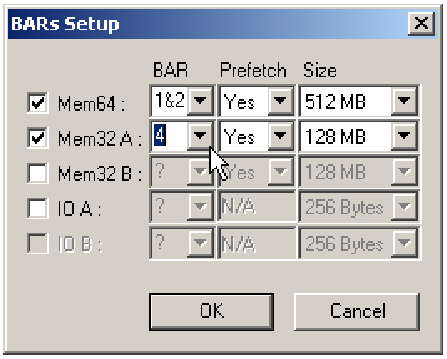
Configuration Space Editor

● Create new Configuration Space File

○ Open existing Configuration Space File:

C:\cfg_space_snapshot

- **Create new Configuration Space File**: Opens the Configuration Space Editor with the default (empty) configuration space.
- **Open existing Configuration Space File**: Activates the list of recently used configuration space files and enables the Select... button. After you have selected a file and then clicked **Select**, the Configuration Space Editor opens.
  - **BARs Setup... button opens the Bars Setup dialog**: The BARs Setup dialog helps setting up Base Address Registers. It provides more convenient way of defining BARs than setting up the BARs manually. In addition, it provides the mapping of BARs to PE*Trainer* memory regions.



BARs Setup

	BAR	Prefetch	Size
☑ Mem64 :	1&2	Yes	512 MB
☑ Mem32 A :	4	Yes	128 MB
□ Mem32 B :	?	Yes	128 MB
□ IO A :	?	N/A	256 Bytes
□ IO B :	?	N/A	256 Bytes

In the screenshot above, BAR1 and BAR2 are assigned to 64-bit memory address space and mapped to Mem64 PE*Trainer* memory region. BAR4 is assigned to 32-bit memory address space and mapped to Mem32A PE*Trainer* memory region.

In addition, Prefetch and Address Space Size can be specified in the dialog. The sizes for each address space are limited to the values supported by PE*Trainer* hardware.

BAR and Prefetch fields affect Hexadecimal/Binary page of Configuration Space Editor.

Size field affects Attributes page of Configuration Space Editor.

The mapping of BARs to PE*Trainer* memory region is discussed in AddressSpace Command in the Help section on the PE*Trainer* Script Language.

The BAR settings are used for Memory and IO completer.

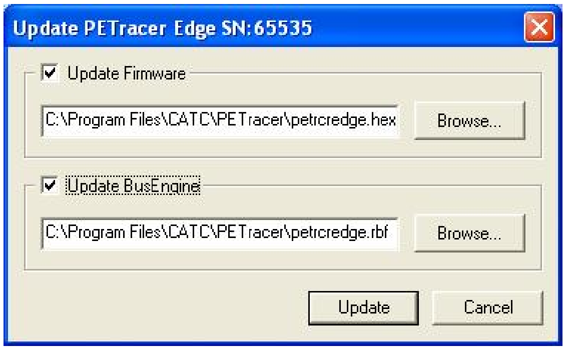
Chapter 16: Updates and Licensing

16.1 Updating the Analyzer

BusEngine™ and/or Firmware updates are now automatic processes that run anytime a new version of the PE*Tracer*™ software is installed that is incompatible with the currently installed BusEngine or Firmware. The update process generates onscreen instructions.

If, however, you need to manually perform BusEngine or Firmware updates, follow these steps:

Step 1 Select **Setup > Update Device** … from the menu to display the Update PETracer Edge dialog box for **PE*Tracer* Edge:**



Update PETracer Edge SN:65535

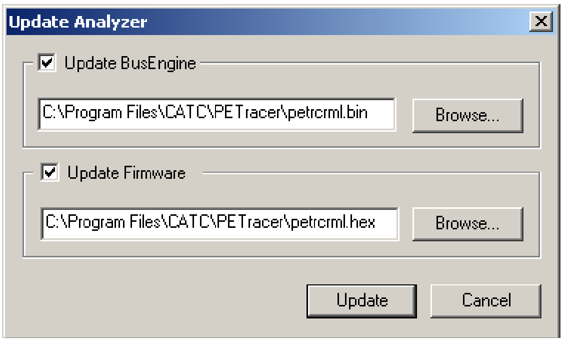
☑ Update Firmware

C:\Program Files\CATC\PETracer\petrcredge.hex

☑ Update BusEngine

C:\Program Files\CATC\PETracer\petrcredge.rbf

or the Update Analyzer dialog box for **PE*Tracer* ML™**:



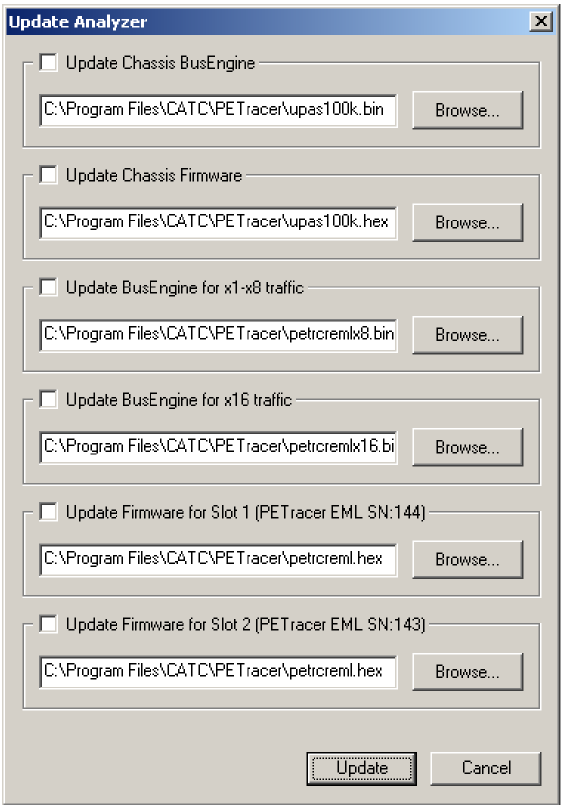
Update Analyzer

☑ Update BusEngine

C:\Program Files\CATC\PETracer\petrcrml.bin

☑ Update Firmware

C:\Program Files\CATC\PETracer\petrcrml.hex or the Update Analyzer dialog box for **PE*Tracer* EML™**:



Update Analyzer

☐ Update Chassis BusEngine
C:\Program Files\CATC\PETracer\upas100k.bin

☐ Update Chassis Firmware
C:\Program Files\CATC\PETracer\upas100k.hex

☐ Update BusEngine for x1-x8 traffic
C:\Program Files\CATC\PETracer\petrcremlx8.bin

☐ Update BusEngine for x16 traffic
C:\Program Files\CATC\PETracer\petrcremlx16.bi

☐ Update Firmware for Slot 1 (PETracer EML SN:144)
C:\Program Files\CATC\PETracer\petrcreml.hex

☐ Update Firmware for Slot 2 (PETracer EML SN:143)
C:\Program Files\CATC\PETracer\petrcreml.hex

Step 2 Select the appropriate file locations for the Firmware and/or BusEngine, using Browse, if necessary.

Step 3 Check the appropriate options (if in doubt, check all options).

Step 4 Click **Update** to initiate the updating of the Analyzer or Edge card.

Updating the Exerciser

To manually update the BusEngine and/or Firmware for PE*Trainer* ML or PE*Trainer* EML, follow the same steps as above.

16.2 License Keys

A License Key is necessary to enable software maintenance. If you attempt to record with the Analyzer without an installed License Key, a message appears to indicate that a License Key is necessary in order to record traffic.

A License Key must be obtained from LeCroy for each Analyzer.

After you obtain the License Key, follow these steps to install it:

Step 1 Select **Help > Update License**… from the menu bar. to display the Update License dialog box.

Step 2 Enter the **path** and **filename** for the License Key or use the Browse button to navigate to the directory that contains the License Key.

Step 3 Select the **.lic** file, and then click **Update Device**.

You can also update your licensing information by selecting **Setup > All Connected Devices** …, then clicking **Update License**.

16.3 License Information

You can view Licensing information for your Analyzer by selecting **Help > Display License Information**... from the menu bar. The License Information window displays data about the maintenance expiration and purchased features.

Appendix A: PE*Tracer* ML Configuration for Lane-Reversed PCI Express Links

The PE*Tracer*™ ML™ and PE*Tracer* Edge protocol Analyzers, when configured appropriately, support data capture and analysis for lane-reversed PCI Express™ links.

Lane reversal is an optional PCI Express link configuration function in which packet data is stripped from physical lane N-1 down to physical lane 0 for a given link of width N (rather than from physical lane 0 up to N-1 as in standard links).

Lane reversal can be used to simplify the physical routing of lane signals from component to component (or component to connector) on a PCB.

A.1 Lane Reversal: Software Configuration

To configure the PE*Tracer* ML or and PE*Tracer* Edge Analyzer for lane reversal, before link recording is initiated, check the **Reverse Lanes** checkbox in the Recording Options > Link dialog box (see “Link” on page 137). Normally, both link directions are reversed if lane reversal is implemented.

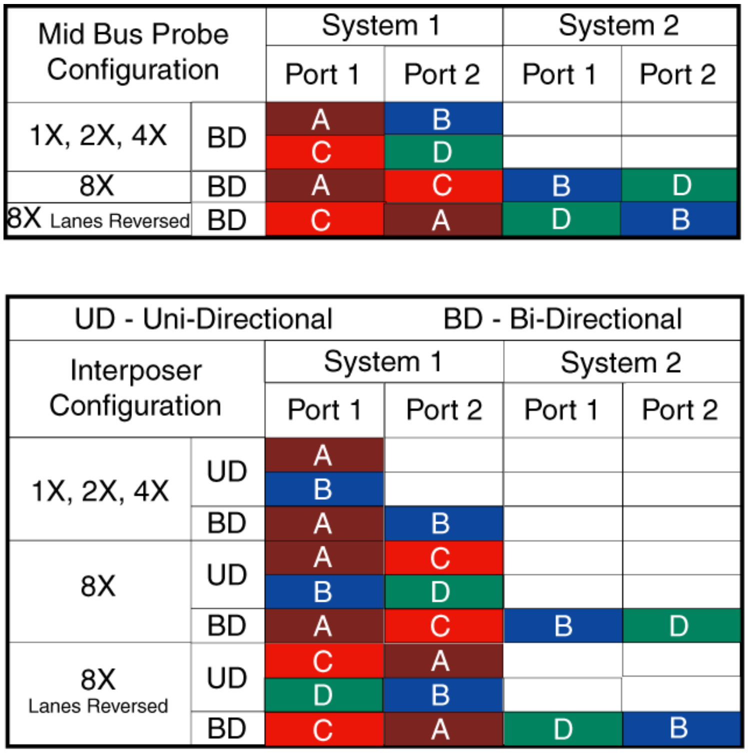
A.2 Lane Reversal for x4 and x2 PCI Express Links

PE*Tracer* ML and PE*Tracer* Edge can be configured to follow lane-reversed x4 and x2 links that include physical lane 0. This covers lane-reversed configurations for native links of x4 and x2 width but may not cover all lane-reversed situations in which the link has been negotiated down to less than its native width. For example, lane reversal of a x4 link on lanes 7-4 of a native x8 connection is not supported, because the resulting link does not include physical lane 0 of the connection.

Note: Lane reversal support does not require any changes to the physical configuration of the PE*Tracer* ML or and PE*Tracer* Edge Analyzer for x4 and x2 links.

A.3 Lane Reversal for x8 PCI Express Links

PEt can be swapped for the Analyzer unit(s) recording a lane-reversed x8 link. The following diagram updates the pull-out card attached to the PE*Tracer* ML unit, showing the special configuration required for x8 lane-reversed links.
(**Note:** The information in this section does not apply to PE*Tracer* Edge.)



Mid Bus Probe Configuration		System 1		System 2	
		Port 1	Port 2	Port 1	Port 2
1X, 2X, 4X	BD	A	B		
		C	D		
8X	BD	A	C	B	D
8X Lanes Reversed	BD	C	A	D	B

UD - Uni-Directional BD - Bi-Directional

Interposer Configuration		System 1		System 2	
		Port 1	Port 2	Port 1	Port 2
1X, 2X, 4X	UD	A			
		B			
	BD	A	B		
8X	UD	A	C		
		B	D		
	BD	A	C	B	D
8X Lanes Reversed	UD	C	A		
		D	B		
	BD	C	A	D	B

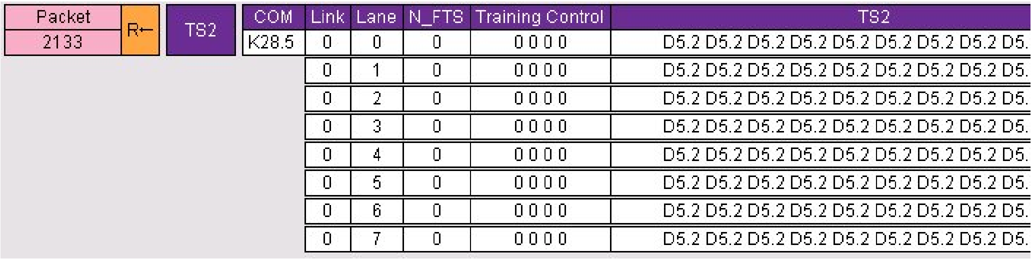
Table 1: Mid-Bus Probe and Interposer Configuration Table

Note: PE*Tracer* ML software revisions prior to version 2.1 have a software problem regarding the configuration of lane reversal for x8 links that prevents correct bidirectional recording of these links. Recording of a single link direction of a lane-reversed x8 link (using a single Analyzer unit) can be performed on software revisions prior to version 2.1 by checking both **Reverse Lanes** checkboxes in the Recording Options > Link dialog box before recording begins, even though the software may indicate that the setting in the right-hand checkbox is not applicable to the current configuration.

A.4 Determining Lane Reversal

The above instructions for configuring the Analyzer for recording lane-reversed PCI Express links assumes that the user knows in advance the configuration of the link. If the configuration of the link (normal or lane-reversed) is unknown, the configuration may be determined by clearing both **Reverse Lanes** checkboxes and recording link training with the Analyzer.

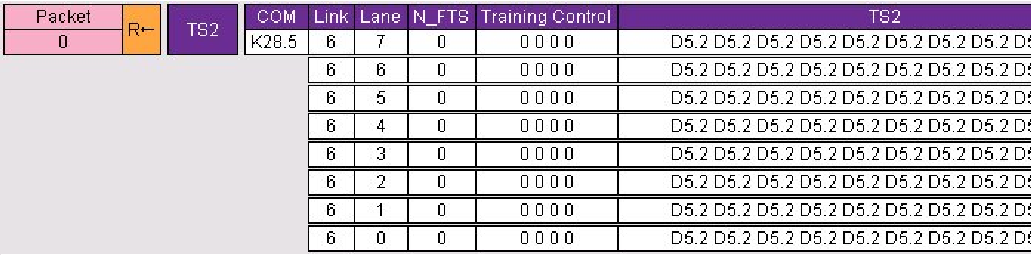
If the link is not lane-reversed, the lane numbers in TS2 ordered sets during link training appear in order from 0 up to N as follows:



Packet	R←	TS2	COM	Link	Lane	N_FTS	Training Control	TS2
2133			K28.5	0	0	0	0 0 0 0	D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.
				0	1	0	0 0 0 0	D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.
				0	2	0	0 0 0 0	D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.
				0	3	0	0 0 0 0	D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.
				0	4	0	0 0 0 0	D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.
				0	5	0	0 0 0 0	D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.
				0	6	0	0 0 0 0	D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.
				0	7	0	0 0 0 0	D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.

Figure A.1 Standard x8 Link

However, if the link is lane-reversed, the lane numbers in the TS2 ordered sets appear in order from N down to 0 as follows:



Packet	R←	TS2	COM	Link	Lane	N_FTS	Training Control	TS2
0			K28.5	6	7	0	0 0 0 0	D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D
				6	6	0	0 0 0 0	D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D
				6	5	0	0 0 0 0	D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D
				6	4	0	0 0 0 0	D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D
				6	3	0	0 0 0 0	D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D
				6	2	0	0 0 0 0	D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D
				6	1	0	0 0 0 0	D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D
				6	0	0	0 0 0 0	D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D5.2 D

Figure A.2 Lane-Reversed x8 Link

How to Contact LeCroy

Type of Service	Contact	
Call for technical support…	US and Canada:	1 (800) 909-2282
	Worldwide:	1 (408) 727-6600
Fax your questions…	Worldwide:	1 (408) 727-6622
Write a letter…	LeCroy Protocol Solutions Group Customer Support 3385 Scott Blvd. Santa Clara, CA 95054 USA	
Send e-mail…	support@CATC.com	
Visit LeCroy's web site…	http://www.lecroy.com/	

Limited Hardware Warranty

So long as you or your authorized representative ("you" or "your"), fully complete and return the registration card provided with the applicable hardware product or peripheral hardware products (each a "Product") within fifteen days of the date of receipt from LeCroy or one of its authorized representatives, LeCroy warrants that the Product will be free from defects in materials and workmanship for a period of three years (the "Warranty Period"). You may also complete your registration form via the internet by visiting http://www.lecroy.com/support/register/. The Warranty Period commences on the earlier of the date of delivery by LeCroy of a Product to a common carrier for shipment to you or to LeCroy's authorized representative from whom you purchase the Product.

What this Warranty Does Not Cover

This warranty does not cover damage due to external causes including accident, damage during shipment after delivery to a common carrier by LeCroy, abuse, misuse, problems with electrical power, including power surges and outages, servicing not authorized by LeCroy, usage or operation not in accordance with Product instructions, failure to perform required preventive maintenance, software related problems (whether or not provided by LeCroy), problems caused by use of accessories, parts or components not supplied by LeCroy, Products that have been modified or altered by someone other than LeCroy, Products with missing or altered service tags or serial numbers, and Products for which LeCroy has not received payment in full.

Coverage During Warranty Period

During the Warranty Period, LeCroy or its authorized representatives will repair or replace Products, at LeCroy's sole discretion, covered under this limited warranty that are returned directly to LeCroy's facility or through LeCroy's authorized representatives.

How to Obtain Warranty Service

To request warranty service, you must complete and return the registration card or register via the internet within the fifteen day period described above and report your covered warranty claim by contacting LeCroy Technical Support or its authorized representative.

LeCroy Technical Support can be reached at 800-909-7112 or via email at support@catc.com. You may also refer to LeCroy's website at **http://www.lecroy.com** for more information on how to contact an authorized representative in your region. If warranty service is required, LeCroy or its authorized representative will issue a Return Material Authorization Number. You must ship the Product back to LeCroy or its authorized representative, in its original or equivalent packaging, prepay shipping charges, and insure the shipment or accept the risk of loss or damage during shipment. LeCroy must receive the Product prior to expiration of the Warranty Period for the repair(s) to be covered. LeCroy or its authorized representative will thereafter ship the repaired or replacement Product to you freight prepaid by LeCroy if you are located in the continental United States. Shipments made outside the continental United States will be sent freight collect.

Please remove any peripheral accessories or parts before you ship the Product. LeCroy does not accept liability for lost or damaged peripheral accessories, data or software.

LeCroy owns all parts removed from Products it repairs. LeCroy may use new and/or reconditioned parts, at its sole discretion, made by various manufacturers in performing warranty repairs. If LeCroy repairs or replaces a Product, the Warranty Period for the Product is not extended.

If LeCroy evaluates and determines there is "no trouble found" in any Product returned or that the returned Product is not eligible for warranty coverage, LeCroy will inform you of its determination. If you thereafter request LeCroy to repair the Product, such labor and service shall be performed under the terms and conditions of LeCroy's then current repair policy. If you chose not to have the Product repaired by LeCroy, you agree to pay LeCroy for the cost to return the Product to you and that LeCroy may require payment in advance of shipment.

General Provisions

THIS LIMITED WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS. YOU MAY HAVE ADDITIONAL RIGHTS THAT VARY BY JURISDICTION. LECROY'S RESPONSIBILITY FOR DEFECTS IN MATERIALS AND WORKMANSHIP IS LIMITED TO REPAIR AND REPLACEMENT AS SET FORTH IN THIS LIMITED WARRANTY STATEMENT. EXCEPT AS EXPRESSLY STATED IN THIS WARRANTY STATEMENT, LECROY DISCLAIMS ALL EXPRESS AND IMPLIED WARRANTIES FOR ANY PRODUCT INCLUDING, BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF AND CONDITIONS OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, AND ANY WARRANTIES THAT MAY ARISE FROM ANY COURSE OF DEALING, COURSE OF PERFORMANCE OR TRADE USAGE. SOME JURISDICTIONS MAY NOT ALLOW LIMITATIONS ON HOW LONG AN IMPLIED WARRANTY LASTS, SO THE PRECEDING LIMITATION MAY NOT APPLY TO YOU.

LECROY DOES NOT ACCEPT LIABILITY BEYOND THE REMEDIES SET FORTH IN THIS LIMITED WARRANTY STATEMENT OR FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES INCLUDING, WITHOUT LIMITATION, ANY LIABILITY FOR THIRD PARTY CLAIMS AGAINST YOU FOR DAMAGES, PRODUCTS NOT BEING AVAILABLE FOR USE, OR FOR LOST DATA OR SOFTWARE. LECROY'S LIABILITY TO YOU MAY NOT EXCEED THE AMOUNT YOU PAID FOR THE PRODUCT THAT IS THE SUBJECT OF A CLAIM. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THE PRECEDING EXCLUSION OR LIMITATION MAY NOT APPLY TO YOU.

The limited warranty on a Product may be transferred for the remaining term if the then current owner transfers ownership of the Product and notifies LeCroy of the transfer. You may notify LeCroy of the transfer by writing to Technical Support at LeCroy, 3385 Scott Blvd., Santa Clara, CA 95054 USA or by email at: support@catc.com. Please include the transferring owner's name and address, the name and address of the new owner, the date of transfer, and the Product serial number.

Index