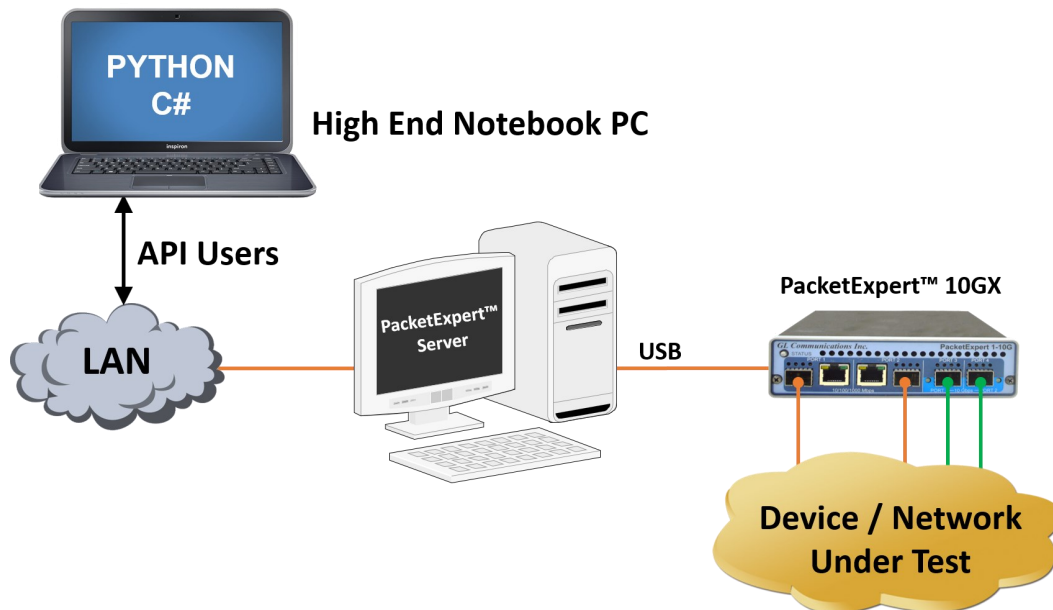


Automated GigE Network Testing using PacketExpert™ 10GX- MAPS™ CLI / API Architecture



Overview

[PacketExpert™](#) supports Command line Interface (CLI) for test automation and remote accessibility of various functionalities such as Bert, Loopback, RFC 2544, Record Playback, IPNetSim™, ExpertSAM™, PacketBroker™, and Multi Stream Traffic Generator and Analyzer using Python, C# client APIs and MAPS™ CLI Client/Server architecture.

Required licenses across PacketExpert™ 1G, 1G/2.5G/10G platforms for CLI support are -

- CXE100 (CLI Server for PXE100 basic and optional software)
- CXN100 (CLI Server for PXN100 basic and optional software)

PacketExpert™ can be configured as server-side application using the GL's MAPS™ Client-Server architecture, to provide the capability of remote operation, automation, and multi-site connectivity, using any client-side scripting tools such as the Python and C#. On the client side, the scripting library enables communication with the MAPS™ CLI Server using TCP/IP socket from a client environment.

The MAPS™ CLI server interfaces with the PacketExpert™ hardware through the USB. The MAPS™ CLI Server developed specifically for PacketExpert™ runs (*.gls) scripts that can control the PacketExpert™ hardware. The advantage of such communication enables user to control PacketExpert™ by sending commands and receiving responses in a scripting language such as Python, C# that is already familiar with many users.

For more information, visit [PacketExpert™ CLI testing](#) webpage.



GL Communications Inc.

818 West Diamond Avenue - Third Floor, Gaithersburg, MD 20878, U.S.A
(Web) www.gl.com - (V) +1-301-670-4784 (F) +1-301-670-9187 - (E-Mail) info@gl.com

Main Features

- Capability of remote operation, automation and multi-site connectivity using Python/C# client and MAPS™ CLI server.
- Supports Bert, Loopback, RFC 2544, Record Playback, ExpertSAM™, PacketBroker™, and Multi-Stream Traffic Generator and Analyzer functionalities.
- PacketExpert™ CLI offers complete Lab Management, Device Provisioning and Test Automation solutions.
- CLI integration with popular framework such as LabVIEW/TestStand and TestShell for test automation.
- Multiple PacketExpert™ can be controlled remotely from single client application via MAPS™ CLI server.
- Support for a wide range of tests setup, interfaces, protocols, and script languages.
- Python, C# client access through MAPS™ CLI Server.
- High Level APIs allows to access PacketExpert™ functionalities.
- Scripts for MAC, VLAN, MPLS, IP and UDP layers testing
- Remote monitoring capability.
- Requires additional licensing for CLI support across various PacketExpert™ platforms.

Working Principle of MAPS™ CLI Client/Server Architecture

MAPS™ CLI Client/Server platform supports various client libraries in different languages, so that users can make use of these different libraries to communicate with the MAPS™ server, and achieve automation using their language of choice. However, these are relatively low-level libraries, which gives users a very fine grain control.

For PacketExpert™ platform, a set of relatively High-Level APIs have been developed on top of the MAPS™ Client library, which greatly reduces the time to develop sample applications and achieve automation. These APIs are developed in the respective languages and are easy-to-use and intuitive. E.g.: C# APIs are provided by means of API classes for each application. Similarly, Python APIs are provided through API scripts that implement API classes for different applications. Also supplied are sample applications, that users can use to work with APIs. Using these high-level APIs and sample applications, users can develop automated tests in a very short period.

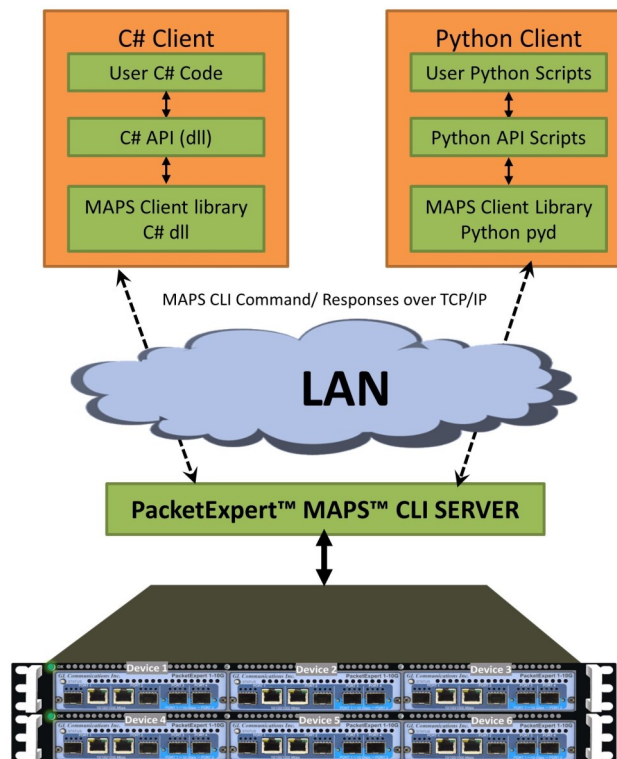


Figure: PacketExpert™ MAPS™ CLI Working Principle

CLI Functional Modules

CLI application consists of 3 functional modules.

- Client Users (Python/C#) - Acts as User Interface which executes Python/C# Scripts instructing the CLI/API server to run the particular script to perform the specific test like BERT/RFC 2544, etc.
- MAPS™ Client Interface (MAPS Client IFC) - acts as an interface between MAPS™ CLI Server and its clients Python/C#. The MAPS™ Python/C# Client application includes a dll file, a packaged library that enables communication with the MAPS™ Server from the client environment.
- MAPS™ CLI Server - is an executable which inherits all features of MAPS™ GUI. MAPS™ CLI/API Server is a scripting based framework which controls the PacketExpert™ hardware using proprietary MAPS™ scripts.

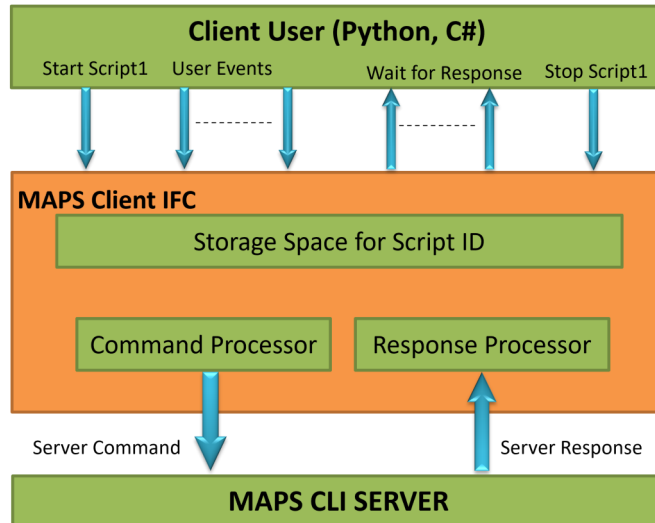


Figure: MAPS™ CLI Functional Modules

C# Client and Scripting

The C# interface developed for PacketExpert™ allows users to control all features of PacketExpert™ through C# APIs. The C# interface is implemented based on a client-server model. The C# client connects to the MAPS™ CLI server using TCP/IP sockets. MAPS™ CLI Server interfaces with PacketExpert™ low level API controlling the hardware. There will be different MAPS™ scripts to implement different applications like BERT, RFC 2544 etc.,

The MAPS C# Interface (MAPS Client IFC) application includes a MAPSCSAPI.dll file, a packaged library that enables communication with the MAPS™ CLI Server from a C# environment.

C# Client invokes APIs which executes the command, that instructs the MAPS™ CLI Server to run the particular script which performs the particular PacketExpert™ tests like BERT, RFC 2544 etc.

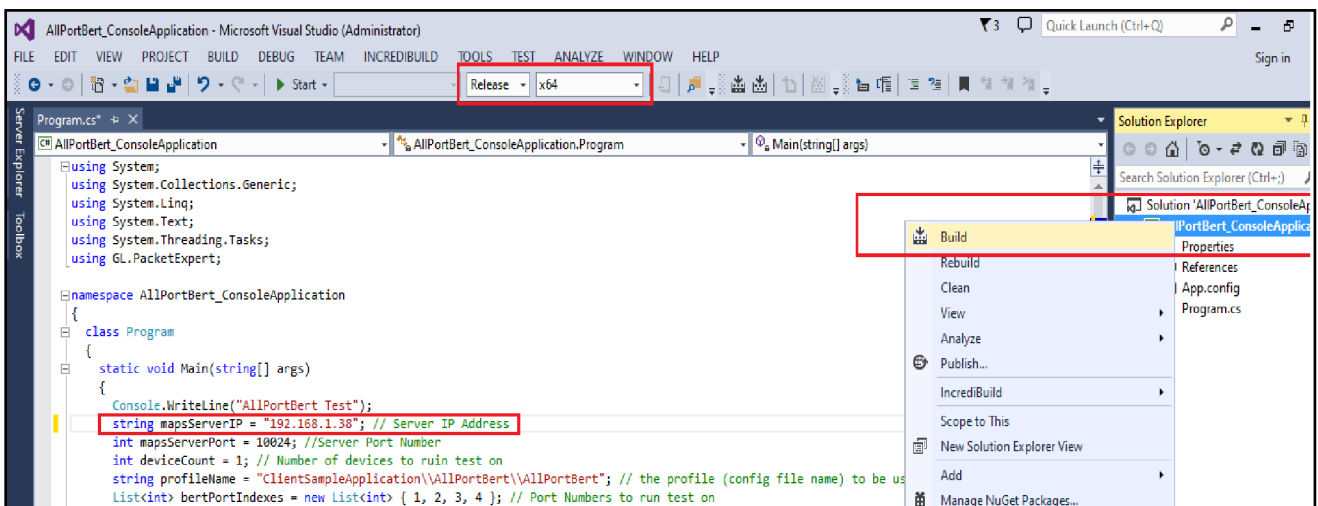


Figure: C# Sample API Client

MAPDS CLI Server (C#)

```

CLI MapsCLI (PACKETEXPERT )
File Edit View
View Latest Command
1:: 2019-1-25 10:40:15.016000 : Start "TestBedDefault.xml";
1:: 2019-1-25 10:40:15.060000 : LoadProfile ""
1:: 2019-1-25 10:40:17.472000 : StartScript 1 "PEX_Init.gls" "" 1;
1:: 2019-1-25 10:40:17.473000 : UserEvent 1 "InitDevice";
1:: 2019-1-25 10:40:17.569000 : UserEvent 1 "LoadModule" # "DeviceId"=1,"ModuleName"="AllPortBert";
1:: 2019-1-25 10:40:21.498000 : StartScript 2 "PEX_BERT_Main.gls" "" 1;
1:: 2019-1-25 10:40:21.499000 : UserEvent 2 "InitBertModule" # "BoardCount"=1;
1:: 2019-1-25 10:40:23.202000 : UserEvent 2 "StartBertModule" # "BoardCount"=1;
1:: 2019-1-25 10:40:43.205000 : UserEvent 2 "LoadInterfaceProfile" # "USProfile"="BERT.pex.AllPortBert.ifc.xml", "USSubProfile"="Port1InterfaceConfig";
1:: 2019-1-25 10:40:43.284000 : UserEvent 2 "LoadBertProfile" # "ProfileName"="BERT.pex.AllPortBert.bert.xml", "USSubProfile"="Port1TxConfig";
1:: 2019-1-25 10:40:43.373000 : UserEvent 2 "LoadBertProfile" # "ProfileName"="BERT.pex.AllPortBert.bert.xml", "USSubProfile"="Port1RxConfig";
1:: 2019-1-25 10:40:43.429000 : UserEvent 2 "ApplyConfiguration" # "PortIndex"=1;
1:: 2019-1-25 10:40:43.473000 : UserEvent 2 "LoadInterfaceProfile" # "USProfile"="BERT.pex.AllPortBert.ifc.xml", "USSubProfile"="Port2InterfaceConfig";
1:: 2019-1-25 10:40:47.488000 : UserEvent 2 "LoadBertProfile" # "ProfileName"="BERT.pex.AllPortBert.bert.xml", "USSubProfile"="Port2TxConfig";
1:: 2019-1-25 10:40:47.556000 : UserEvent 2 "LoadBertProfile" # "ProfileName"="BERT.pex.AllPortBert.bert.xml", "USSubProfile"="Port2RxConfig";
1:: 2019-1-25 10:40:47.634000 : UserEvent 2 "ApplyConfiguration" # "PortIndex"=2;
1:: 2019-1-25 10:40:47.711000 : UserEvent 2 "LoadInterfaceProfile" # "USProfile"="BERT.pex.AllPortBert.ifc.xml", "USSubProfile"="Port3InterfaceConfig";
1:: 2019-1-25 10:40:47.712000 : UserEvent 2 "LoadBertProfile" # "ProfileName"="BERT.pex.AllPortBert.bert.xml", "USSubProfile"="Port3TxConfig";
1:: 2019-1-25 10:40:47.735000 : UserEvent 2 "LoadBertProfile" # "ProfileName"="BERT.pex.AllPortBert.bert.xml", "USSubProfile"="Port3RxConfig";
1:: 2019-1-25 10:40:47.780000 : UserEvent 2 "ApplyConfiguration" # "PortIndex"=3;
1:: 2019-1-25 10:40:47.858000 : UserEvent 2 "LoadInterfaceProfile" # "USProfile"="BERT.pex.AllPortBert.ifc.xml", "USSubProfile"="Port4InterfaceConfig";
1:: 2019-1-25 10:40:47.870000 : UserEvent 2 "LoadBertProfile" # "ProfileName"="BERT.pex.AllPortBert.bert.xml", "USSubProfile"="Port4TxConfig";
1:: 2019-1-25 10:40:47.881000 : UserEvent 2 "LoadBertProfile" # "ProfileName"="BERT.pex.AllPortBert.bert.xml", "USSubProfile"="Port4RxConfig";
1:: 2019-1-25 10:40:47.925000 : UserEvent 2 "ApplyConfiguration" # "PortIndex"=4;
1:: 2019-1-25 10:40:48.002000 : UserEvent 2 "StartRxBert" # "PortIndex"=1;
1:: 2019-1-25 10:40:48.025000 : UserEvent 2 "StartRxBert" # "PortIndex"=2;
1:: 2019-1-25 10:40:48.037000 : UserEvent 2 "StartRxBert" # "PortIndex"=3;
1:: 2019-1-25 10:40:48.059000 : UserEvent 2 "StartRxBert" # "PortIndex"=4;
1:: 2019-1-25 10:40:48.070000 : UserEvent 2 "StartTxBert" # "PortIndex"=1;
1:: 2019-1-25 10:40:48.082000 : UserEvent 2 "StartTxBert" # "PortIndex"=2;

```

Figure: MAPS CLI Server (C#)

Python Client and Scripting

The Python interface developed for PacketExpert™ allows users to control all features of PacketExpert™ through Python APIs. The Python interface is implemented based on a client-server model. The server is the MAPS™ CLI server, which interfaces with the PacketExpert™ hardware through the USB. The client consists of a Python API dll and Python API scripts, which allows user to control the MAPS™ CLI server, issue commands and get back results.

The MAPS™ Python Interface (MAPS Client IFC) application includes a PythonMapsCliIfc.pyd file or PythonMapsCliIfc.so, a packaged library that enables communication with the MAPS™ CLI Server from a Python environment. MAPS Client IFC provides added benefits of a fully capable flow control engine with built commands.

```

Run: AllPortBert_SampleApplication x AllPortBert_SampleApplication x
AllPortBert application Initialised
Press any key to continue , 'q' to quit
Running BERT Test
Loading Configuration
*****Device 1 *****
Load Configuration Done
*****Ports Interface Information*****
*****Device 1 *****
['Up', '00-21-C2-00-09-B4', 'ELECTRICAL', 'Complete', '1000Mbps', 'Full Duplex', 'Enabled']
['Up', '00-21-C2-00-09-B5', 'ELECTRICAL', 'Complete', '1000Mbps', 'Full Duplex', 'Enabled']
['Up', '00-21-C2-00-09-B6', 'ELECTRICAL', 'Complete', '1000Mbps', 'Full Duplex', 'Enabled']
['Up', '00-21-C2-00-09-B7', 'ELECTRICAL', 'Complete', '1000Mbps', 'Full Duplex', 'Enabled']
*****Device 1 *****
Port : 1 Rx Started
Port : 2 Rx Started
Port : 3 Rx Started
Port : 4 Rx Started
Port : 1 Tx Started
Port : 2 Tx Started
Port : 3 Tx Started
Port : 4 Tx Started

```

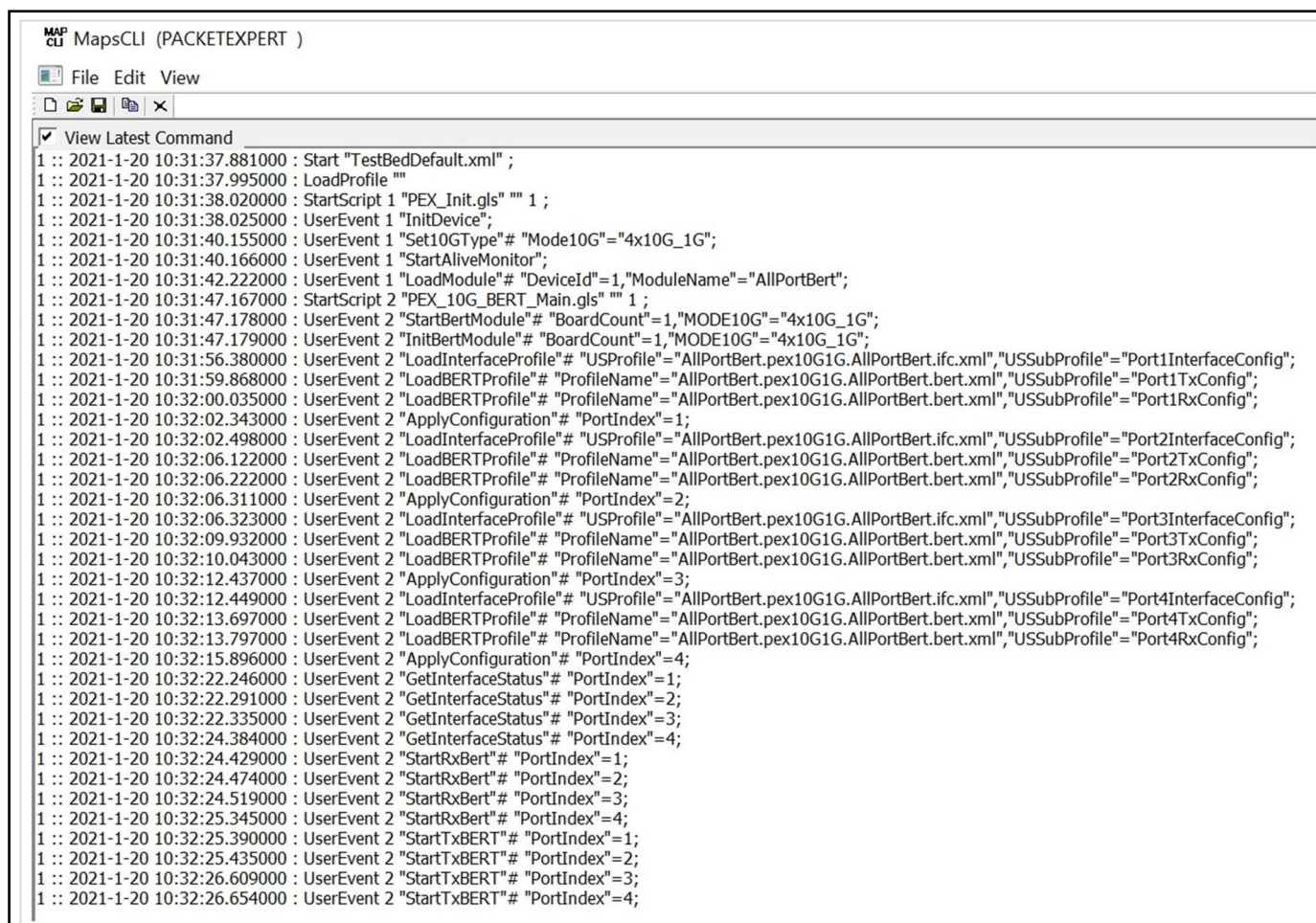
Figure: Executing the Client script in “PyCharm” console

MAPS CLI Server

The client connects to the MAPS™ CLI server via TCP/IP sockets.

PacketExpert™ will run the MAPS™ CLI Server, which can interface with PacketExpert™ low level API to control hardware. There are different MAPS™ scripts to implement various applications like BERT, RFC 2544 etc.

MAPS™ CLI Server consists of GL's proprietary scripts (.gls files) that actually implement various PacketExpert™ functionalities like BERT, RFC 2544 etc. XML files (called as "Profiles") containing the configuration information for the test. PacketExpert™ internal low level APIs are used within MAPS™ scripts to control the PacketExpert™ hardware.



```

MAPS CLI (PACKETEXPERT)
File Edit View
View Latest Command
1 :: 2021-1-20 10:31:37.881000 : Start "TestBedDefault.xml" ;
1 :: 2021-1-20 10:31:37.995000 : LoadProfile ""
1 :: 2021-1-20 10:31:38.020000 : StartScript 1 "PEX_Init.gls" "" 1 ;
1 :: 2021-1-20 10:31:38.025000 : UserEvent 1 "InitDevice";
1 :: 2021-1-20 10:31:40.155000 : UserEvent 1 "Set10GType" # "Mode10G"="4x10G_1G";
1 :: 2021-1-20 10:31:40.166000 : UserEvent 1 "StartAliveMonitor";
1 :: 2021-1-20 10:31:42.222000 : UserEvent 1 "LoadModule" # "DeviceId"=1,"ModuleName"="AllPortBert";
1 :: 2021-1-20 10:31:47.167000 : StartScript 2 "PEX_10G_BERT_Main.gls" "" 1 ;
1 :: 2021-1-20 10:31:47.178000 : UserEvent 2 "StartBertModule" # "BoardCount"=1,"MODE10G"="4x10G_1G";
1 :: 2021-1-20 10:31:47.179000 : UserEvent 2 "InitBertModule" # "BoardCount"=1,"MODE10G"="4x10G_1G";
1 :: 2021-1-20 10:31:56.380000 : UserEvent 2 "LoadInterfaceProfile" # "USProfile"="AllPortBert.pex10G1G.AllPortBert.ifc.xml", "USSubProfile"="Port1InterfaceConfig";
1 :: 2021-1-20 10:31:59.868000 : UserEvent 2 "LoadBERTProfile" # "ProfileName"="AllPortBert.pex10G1G.AllPortBert.bert.xml", "USSubProfile"="Port1TxConfig";
1 :: 2021-1-20 10:32:00.035000 : UserEvent 2 "LoadBERTProfile" # "ProfileName"="AllPortBert.pex10G1G.AllPortBert.bert.xml", "USSubProfile"="Port1RxConfig";
1 :: 2021-1-20 10:32:02.343000 : UserEvent 2 "ApplyConfiguration" # "PortIndex"=1;
1 :: 2021-1-20 10:32:02.498000 : UserEvent 2 "LoadInterfaceProfile" # "USProfile"="AllPortBert.pex10G1G.AllPortBert.ifc.xml", "USSubProfile"="Port2InterfaceConfig";
1 :: 2021-1-20 10:32:06.122000 : UserEvent 2 "LoadBERTProfile" # "ProfileName"="AllPortBert.pex10G1G.AllPortBert.bert.xml", "USSubProfile"="Port2TxConfig";
1 :: 2021-1-20 10:32:06.222000 : UserEvent 2 "LoadBERTProfile" # "ProfileName"="AllPortBert.pex10G1G.AllPortBert.bert.xml", "USSubProfile"="Port2RxConfig";
1 :: 2021-1-20 10:32:06.311000 : UserEvent 2 "ApplyConfiguration" # "PortIndex"=2;
1 :: 2021-1-20 10:32:06.323000 : UserEvent 2 "LoadInterfaceProfile" # "USProfile"="AllPortBert.pex10G1G.AllPortBert.ifc.xml", "USSubProfile"="Port3InterfaceConfig";
1 :: 2021-1-20 10:32:09.932000 : UserEvent 2 "LoadBERTProfile" # "ProfileName"="AllPortBert.pex10G1G.AllPortBert.bert.xml", "USSubProfile"="Port3TxConfig";
1 :: 2021-1-20 10:32:10.043000 : UserEvent 2 "LoadBERTProfile" # "ProfileName"="AllPortBert.pex10G1G.AllPortBert.bert.xml", "USSubProfile"="Port3RxConfig";
1 :: 2021-1-20 10:32:12.437000 : UserEvent 2 "ApplyConfiguration" # "PortIndex"=3;
1 :: 2021-1-20 10:32:12.449000 : UserEvent 2 "LoadInterfaceProfile" # "USProfile"="AllPortBert.pex10G1G.AllPortBert.ifc.xml", "USSubProfile"="Port4InterfaceConfig";
1 :: 2021-1-20 10:32:13.697000 : UserEvent 2 "LoadBERTProfile" # "ProfileName"="AllPortBert.pex10G1G.AllPortBert.bert.xml", "USSubProfile"="Port4TxConfig";
1 :: 2021-1-20 10:32:13.797000 : UserEvent 2 "LoadBERTProfile" # "ProfileName"="AllPortBert.pex10G1G.AllPortBert.bert.xml", "USSubProfile"="Port4RxConfig";
1 :: 2021-1-20 10:32:15.896000 : UserEvent 2 "ApplyConfiguration" # "PortIndex"=4;
1 :: 2021-1-20 10:32:22.246000 : UserEvent 2 "GetInterfaceStatus" # "PortIndex"=1;
1 :: 2021-1-20 10:32:22.291000 : UserEvent 2 "GetInterfaceStatus" # "PortIndex"=2;
1 :: 2021-1-20 10:32:22.335000 : UserEvent 2 "GetInterfaceStatus" # "PortIndex"=3;
1 :: 2021-1-20 10:32:24.384000 : UserEvent 2 "GetInterfaceStatus" # "PortIndex"=4;
1 :: 2021-1-20 10:32:24.429000 : UserEvent 2 "StartRxBert" # "PortIndex"=1;
1 :: 2021-1-20 10:32:24.474000 : UserEvent 2 "StartRxBert" # "PortIndex"=2;
1 :: 2021-1-20 10:32:24.519000 : UserEvent 2 "StartRxBert" # "PortIndex"=3;
1 :: 2021-1-20 10:32:25.345000 : UserEvent 2 "StartRxBert" # "PortIndex"=4;
1 :: 2021-1-20 10:32:25.390000 : UserEvent 2 "StartTxBERT" # "PortIndex"=1;
1 :: 2021-1-20 10:32:25.435000 : UserEvent 2 "StartTxBERT" # "PortIndex"=2;
1 :: 2021-1-20 10:32:26.609000 : UserEvent 2 "StartTxBERT" # "PortIndex"=3;
1 :: 2021-1-20 10:32:26.654000 : UserEvent 2 "StartTxBERT" # "PortIndex"=4;

```

Figure: MAPS CLI Server (Python)

LabVIEW / TestStand Integration

Using PacketExpert™ APIs, it is very easy to integrate PacketExpert™ into LabVIEW. Since LabVIEW supports C# language, the PacketExpert™ C# API dll can be directly imported into LabVIEW and used in the Graphical environment that LabVIEW provides to control PacketExpert™ devices and automate testing.

With LabVIEW, it is easy to create flexible test scripts that control multiple hardware, customize test system with graphical programming, included analysis, drag and drop interface. This makes the system compatible with GL's PacketExpert™ software. Eg: Importing the C# Client API dll into LabVIEW instantly provides ability to run any PacketExpert™ test application like – BERT, RFC2544, Loopback and others.

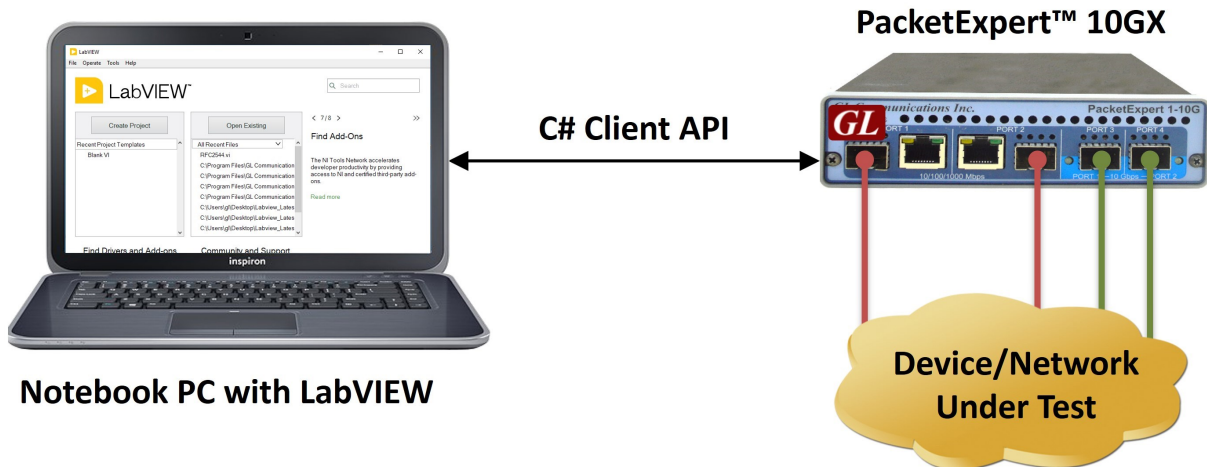


Figure: LabVIEW with PacketExpert™

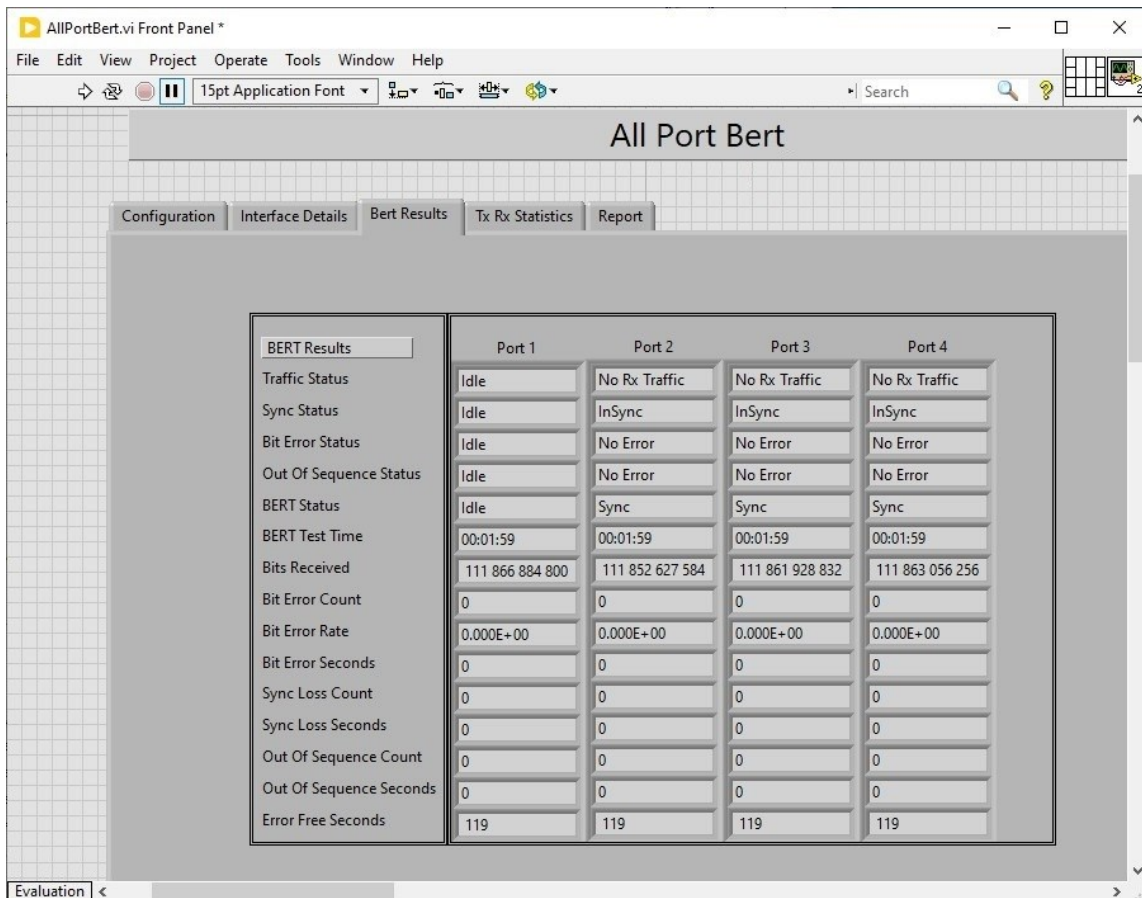


Figure: LabVIEW Test Results

Hardware Specifications

| | | |
|---|--|---|
|  <p>Portable 10GX Hardware Unit</p> |  <p>1U mTOP™ PacketExpert™ 10GX Rack Unit (3 PXN100s)</p>  <p>Stacked 1U mTOP™ PacketExpert™ 10GX Rack Unit</p> |  <p>PacketExpert™ 10GX mTOP™ Probe</p> |
| <p>Physical Specification: Length: 8.45 in. (214.63 mm) Width: 5.55 in. (140.97 mm) Height: 1.60 in (40.64 mm) Weight: 1.713 lbs. (0.75 kg)</p> | <ul style="list-style-type: none"> • Dimension: 1U/2U mTOP™ - 19" W x 16" L • 1U mTOP™ Rackmount Enclosure can support up to 3 PXN100s • 2U mTOP™ Rackmount Enclosure can support up to 6 PXN100s • Optional 4 to 12 Port SMA Jack Trigger Board (TTL Input/Output) • Weight: (not including the rails) 1U with 3x PXN100 : 11 lbs 2U with 6x PXN100 : 22 lbs | <p>Physical Specification: Length: 10.4 in. (264.16 mm) Width: 8.4 in. (213.36 mm) Height: 3.0 in. (76.2 mm)</p> |
| <p>Bus Interface: USB 3.0 External Power Supply:</p> <ul style="list-style-type: none"> • +12 Volts (Medical Grade), 3 Amps (For portable units having serial number ≥ 188400) • +9 Volts, 2 Amps (For portable units having serial number < 188400) • Optional 4-Port SMA Jack Trigger Board (TTL Input/Output) | <p>SBC Specifications:</p> <ul style="list-style-type: none"> • Intel Core i3 or optional i7 NUC Equivalent, • Windows® 11 64-bit Pro Operating System • USB 3.0 and USB 2.0 Ports, ATX Power Supply • USB Type C Ports, Ethernet 2.5GigE port • 256 GB Hard drive, 8G Memory (Min) • Two HDMI ports | <p>SBC Specifications:</p> <ul style="list-style-type: none"> • Intel Core i3 or optional i7 NUC Equivalent, • Windows® 11 64-bit Pro Operating System • USB 3.0 and USB 2.0 Ports • 12/3A Power Supply • USB Type C Ports, Ethernet 2.5GigE port • 256 GB Hard drive, 8G Memory (Min) • Two HDMI ports |
| <p>Temperature: Operating Temperature 0° C to +50° C (only up to operating altitude of 5000 feet, and for Optical SFPs only i.e. Non Electrical SFPs) +5° to +40° C (for operating altitude up to 10,000 feet, and for both Electrical and Optical SFPs) Non-Operating Temperature: -30° to +60° C</p> <p>Humidity: Operating Humidity: 0% to 80% RH Non-Operating Humidity: 0% to 95% RH</p> <p>Altitude: Operating Altitude: up to 10,000 feet Non-Operating Altitude: up to 50,000 feet</p> <p>Interfaces: 4 x 1G Base-X Optical OR 10/100/1000 Base-T Electrical 2 x 100Mbps Base-FX Optical 2 x 2.5 Gbps Electrical/Optical Interface 2 x 10G Base-SR, -LR -ER Electrical/Optical Interface Single Mode or Multi Mode Fiber SFP support with LC connector</p> <p>Protocols: IEEE 802.3ae LAN PHY compliance RFC 2544 compliance</p> <div data-bbox="1086 1563 1489 1995">  <p>Pelican Carry Case</p> </div> | | |

Buyer's Guide

| Item No | Product Description |
|------------------------|------------------------------------|
| PXN100 | PacketExpert™ 10GX |
| PXN101 | 10G and 2.5G option for PXN100 |
| CXN100 | CLI Server for PXN100 |
| PXN112 | PacketExpert™ 10GX – SA (12-Port) |
| PXN124 | PacketExpert™ 10GX – SA (24-Port) |
| MT001 | mTOP 1U Rack Mount Enclosure w/SBC |
| PXE100 | PacketExpert™ 1G |
| CXE100 | CLI Server for PXE100 |

Note: PCs which include GL hardware/software require Intel or AMD processors for compliance.

For more information, visit [IP ethernet testers PacketExpert™ platforms](#) webpage.



GL Communications Inc.

818 West Diamond Avenue - Third Floor, Gaithersburg, MD 20878, U.S.A
 (Web) www.gl.com - (V) +1-301-670-4784 (F) +1-301-670-9187 - (E-Mail) info@gl.com