Net Protocol Suite Python API Reference Manual

Net Protocol Suite Software Version 6.21

Generated: March 7, 2024, 09:11

Document disclaimer

The information contained 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.

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

Trademarks and Service Marks

Teledyne LeCroy is a trademark of Teledyne LeCroy.

Microsoft and Windows are registered trademarks of Microsoft Inc.

All other trademarks are property of their respective companies.

Copyright

Contents

List of Tables

Chapter 1

Introduction / Setup / Definitions

1.1 Introduction

Net Protocol Suite contains a Python module library, TLNetAPI, that provides automation features for the Python language.

1.2 Setup Requirements

1.2.1 Windows

Python version: 2.7.x or 3.6.x

Python module location: The "TLNetAPI.py" module must be located in the \API\SDK\Bin directory

e.g., C:\Users\Public\Documents\LeCroy\Net Protocol Suite\API\SDK\Bin\TLNetAPI.py

Note: In order to use " TLNetAPI.py " module in your python script, use (for example):

sys.path.append(' C:\Users\Public\Documents\LeCroy\Net Protocol Suite\API\SDK\Bin')

import TLNetAPI

1.2.2 Linux

Linux versions: Ubuntu 16 64-bit, Fedora 18 32-bit, Fedora 18 64-bit Python version:

- Ubuntu 16: 2.7.x and 3.6.x
- Fedora 18: 2.7.x and 3.3.x

Python module location: The "libTLNetAPI_python.py" module must be located in the "/API/SDK/Bin" directory

e.g., /usr/local/LeCroy/NetProtocolSuite/API/SDK/Bin/libTLNetAPI_python.py

In order to use " libTLNetAPI_python.py" module in your python script use below statement:

sys.path.append('/usr/local/LeCroy/NetProtocolSuite/API/SDK/Bin')

import libTLNetAPI_python

Below rules shall be added to the firewall:

-A OUTPUT -p tcp --match multiport --dport 4000:4003 -j ACCEPT

-A INPUT -p tcp --match multiport --dport 4000:4003 -j ACCEPT

-A OUTPUT -p udp --match multiport --dport 4033:4035 -j ACCEPT

-A INPUT -p udp --match multiport --dport 4033:4035 -j ACCEPT

Note: All required environment variables will be set by the “install.sh” script.

1.2.3 Python Libraries

The enum34 library is required, which might not be installed by default in Python 2.7.x or 3.3.x installations. To install it manually, run 'python -m pip install enum34'.

1.2.4 Connecting to Analyzer over the Network

By default, the SierraNet products use DHCP to obtain an IP address from the network. If needed, static IP settings can be configured through the product’s front panel controls.

The SierraNet products use the following ports to communicate with the TLNetAPI:

- TCP Ports: 3999 – 4003
- UDP Ports: 4033 – 4035

If the TLNetAPI is running on a different network subnet than the SierraNet analyzer, then it is recommended to configure the TLNetAPI to automatically discover analyzers on other subnets using the APIDeviceManager.Discovery_AddSubnet function.

1.3 Definitions

The following is a list of some terms frequently used in this document. Refer to the Net Protocol Suite User Manual for descriptions of other terms not listed here.

Trace - This is the file that contains the data captured by the analyzer.

Project - This is a configuration file that defines the following parameters of your test setup:

- One or more analyzer hardware chains (models and serial numbers)
- Ports configuration
- Analyzer hardware settings
- Analyzer recording, triggering, and filtering settings
- Jammer scenarios

Chain - This is a set of analyzer hardware units. A chain of 1 unit is valid and most common. A chain of multiple units would be linked together with a CATC Sync cable. Refer to the Net Protocol Suite User Manual for details on the hardware's expansion capabilities. Each chain in a Project operates independently from other chains in the Project.

Analyzer - This is the function of the analyzer hardware that captures traffic at line rate and stores it in a dedicated capture buffer.

Jammer - This is the function of the analyzer hardware that injects errors into traffic at line rate.

Scenario - This is the user-defined program that specifies the Jammer behavior.

Chapter 2

Net Protocol Suite API Reference

2.1 API Return Values

On success, the return value of a function call will be either the data requested by the call, if any, or the value TLNetAPI.**EAPIErrorCode**.API_OK.

On failure, a Python runtime exception will be raised, and the error details can be retrieved from the exception object.

2.2 TLNetAPI

The package name is as follows:

TABLE 2.1: Package Name

Name	Description
TLNetAPI.py	For Windows
libTLNetAPI_python.py	For Linux

This documentation assumes that you have imported the one you need as follows:

import libTLNetAPI_python as TLNetAPI or import TLNetAPI

enum TLNetAPI.EDataType

This enumeration describes the type of packet.

TABLE 2.2: Type of Packet

Value	Description
DATA_TYPE_NET_INVALID	An invalid type of packet that is not recognized.
DATA_TYPE_CONNECT_DISCONNECT	A connect/disconnect event.
DATA_TYPE_ETHERNET_FRAME	An Ethernet frame.
DATA_TYPE_FC_FRAME	A Fibre Channel frame.
DATA_TYPE_ETHERNET_ORDERSET	An Ethernet ordered set.
DATA_TYPE_FC_ORDERSET	A Fibre Channel ordered set.
DATA_TYPE_AUTO_NEGOTIATION	An auto-negotiation packet.
DATA_TYPE_TRAINING_SEQUENCE	A link training packet.

enum TLNetAPI.EAPIErrorCode

This enumeration describes the returned error code from API execution.

TABLE 2.3: Returned Error Code

Value	Description
API_OK	Success.
API_EXCEPTION	An exception occurred.
API_ERROR_INVALID_OBJECT	An invalid object was used or referenced.
API_ERROR_INSUFFICIENT_MEMORY	Not enough memory to complete the task.
API_ERROR_INVALID_CHAIN_INDEX	The specified chain index is invalid.
API_ERROR_INVALID_DEVICE_INDEX	The specified device index is invalid.
API_ERROR_INVALID_PAIR_PORT_INDEX	The specified port index is invalid.
API_ERROR_INVALID_TRIGGER_NAME	The specified trigger settings name is invalid.
API_ERROR_INVALID_TRIGGER_POSITION	The specified trigger position is invalid.
API_ERROR_INVALID_SEGMENT_SIZE	The specified buffer segment size is invalid.

API_ERROR_INVALID_NUMBER_OF_SEGMENT	The specified buffer segment count is invalid.
API_ERROR_INVALID_TRIG_MODE	The specified trigger mode is invalid.
API_ERROR_CAN_NOT_STOP_JAMMER	Unable to stop the jammer.
API_ERROR_CAN_NOT_START_JAMMER	Unable to start the jammer.
API_ERROR_INVALID_TRACE	The specified trace is invalid.
API_ERROR_INVALID_INDEX_OF_PACKET	The specified packet index is invalid.
API_ERROR_RECORDING	An exception occurred during recording.

Value	Description
API_ERROR_UPLOADING	An exception occurred during uploading.
API_ERROR_JAMMER	An exception occurred during jamming.
API_ERROR_CAN_NOT_OPEN_TRACE	Unable to open the specified trace.
API_ERROR_CAN_NOT_UPDATE_TRACE	Unable to update the specified trace.
API_ERROR_CAN_NOT_SAVE_TRACE	Unable to save the specified trace.
API_ERROR_CAN_NOT_CLOSE_TRACE	Unable to close the specified trace.
API_ERROR_CAN_NOT_CLOSE_PROJECT	Unable to close the specified project.
API_ERROR_CAN_NOT_SAVE_PROJECT	Unable to save the specified project.
API_ERROR_CAN_NOT_OPEN_PROJECT	Unable to open the specified project.
API_ERROR_CAN_NOT_DISCONNECT_FROM_DEVICE	Unable to disconnect from the specified device.
API_ERROR_SELECTED_DEVICE_IS_NOT_UPDATED	The specified device needs a FW/BE update.
API_ERROR_INVALID_SCENARIO_NAME	The specified jammer scenario name is invalid.
API_ERROR_CAN_NOT_STOP_RECORDING	Unable to stop recording.
API_ERROR_CAN_NOT_START_RECORDING	Unable to start recording.
API_ERROR_CAN_NOT_ASSIGN_TO_DEVICE	Unable to assign the specified device to a project.
API_ERROR_TLNETAPI_IS_NOT_INITIALIZED	The module is not initialized.
API_ERROR_INVALID_ARGUMENT	A specified argument is invalid.
API_ERROR_PROJECT_IS_NOT_OPEN	The specified project is not open.
API_ERROR_PROJECT_ALREADY_OPEN	The specified project is already open.

API_ERROR_CAN_NOT_STOP_RTS	Could not stop RTS.
API_ERROR_CAN_NOT_START_RTS	Could not start RTS.
API_ERROR_CAN_NOT_OPEN_LOG_RTS	Could not open RTS log.
API_ERROR_NO_STARTED_RTS_TO_STOP	No running RTS instances to be stopped.
API_ERROR_INVALID_SCRIPT	Invalid script specified.
API_ERROR_FAILED_TO_SET_DEVICE_ALIASNAME	Unable to set device alias name.
API_ERROR_FAILED_TO_GET_DEVICE_IP	Unable to get device IP address.
API_ERROR_FAILED_TO_SET_DEVICE_IP	Unable to set device IP address.
API_ERROR_FAILED_TO_UPDATE_DEVICE	Unable to update device FW/BE.
API_ERROR_FAILED_TO_UPDATE_LICENSE	Unable to update device license.
API_UPDATED_SUCCESSFULLY_BUT_POWER_CYCLE_NEEDED	Successful device update, but must power cycle device.
API_ERROR_FAILED_TO_GET_REGISTER_VALUE	Unable to get device register.
API_ERROR_FAILED_TO_SET_REGISTER_VALUE	Unable to set device register.
API_ERROR_REPEATED_SUBNET_ADDRESS	Duplicated subnet address specified.
API_ERROR_SUBNET_DOES_NOT_EXIST	Invalid subnet name specified.
API_ERROR_FAILED_TO_ADD_SUBNET	Unable to add subnet.
API_ERROR_ADMIN_MODE_LOGIN_INVALID_PASSWORD	Invalid password specified for admin mode.
API_ERROR_ADMIN_MODE_UNABLE_LOGIN	Unable to login to admin mode.
Value	**Description**
API_ERROR_ADMIN_MODE_DISCONNECT_FAILED	Unable to force device disconnect.
API_ERROR_FAIL_TO_ADD_DEVICE_IN_PROJECT	Unable to add device to project.
API_ERROR_FAIL_TO_ADD_JAMMER_OBJECT	Unable to add jammer item in scenario.
API_ERROR_NO_JAMMER_FEATURE_AVAILABLE	The jammer function is not available.
API_ERROR_NO_ANALYZER_FEATURE_AVAILABLE	The analyzer function is not available.
API_ERROR_FAIL_TO_ADD_TRIGGER_SETTING	Unable to add trigger.
API_ERROR_FAIL_TO_ADD_FILTER_SETTING	Unable to add filter.

enum TLNetAPI.ELinkConfiguration

This enumeration describes the analyzer port configuration.

In general, the form of an enumeration value is:

`LINK_CONFIG__[P1/P2 Config]__[P3/P4 Config]__[P5/P6 Config]__[P7/P8 Config]`

where each `[Px/Py Config]` is made up of a Function Code and a Protocol/Speed Code.

The Function Codes are:

TABLE 2.4: Function Codes

Function Code	Description
A	Analyzer
J	Jammer
AJ	Before-Jam-Analyzer + Jammer
JA	Jammer + After-Jam-Analyzer
AJA	Before-Jam-Analyzer + Jammer + After-Jam-Analyzer
0	unconfigured

The Protocol/Speed Codes are:

TABLE 2.5: Protocol / Speed Codes

Protocol/Speed Code	Description	Supported Devices
GE10	10GbE or 25GbE	10GbE: M168/M408/T328/M328/M328Q 25GbE: T328/M328/M328Q
GE50	50GbE	T328/M328/M328Q
GE100	40GbE or 100GbE	40GbE: M408/T328/M328/M328Q 100GbE: T328/M328/M328Q
FC	Fibre Channel	All

Note: Not all combinations are valid.

TABLE 2.6: ELinkConfiguration

Link configuration	Product/Supported speed			
	M408 M168	T328	M328 M328Q	M648
LINK_CONFIG__A_GE10__A_GE10__A_GE10__A_GE10 LINK_CONFIG__A_GE10__A_GE10__0__0 LINK_CONFIG__A_GE10__0__0__0 LINK_CONFIG__0__0__A_GE10__A_GE10 LINK_CONFIG__0__0__0__A_GE10 LINK_CONFIG__0__0__A_GE10__0 LINK_CONFIG__0__A_GE10__0__0 LINK_CONFIG__0__0__0__A_GE10 LINK_CONFIG__0__0__A_GE10__0 LINK_CONFIG__0__A_GE10__0__0 LINK_CONFIG__A_GE10__0__A_GE10__0 LINK_CONFIG__0__A_GE10__0__A_GE10	10G	×	×	×
LINK_CONFIG__A_FC__A_FC__A_FC__A_FC LINK_CONFIG__A_FC__A_FC__0__0 LINK_CONFIG__A_FC__0__0__0 LINK_CONFIG__0__0__A_FC__A_FC LINK_CONFIG__0__0__0__A_FC LINK_CONFIG__0__0__A_FC__0 LINK_CONFIG__0__A_FC__0__0 LINK_CONFIG__A_FC__0__A_FC__0	1G,4G, 8G, 16G	8G,16G, 32G	8G,16G, 32G	×
LINK_CONFIG__A_FC__0__A_GE10__0 LINK_CONFIG__A_GE10__0__A_FC__0 LINK_CONFIG__A_FC__A_FC__A_GE10__A_GE10 LINK_CONFIG__A_GE10__A_GE10__A_FC__A_FC LINK_CONFIG__A_GE10__A_FC__A_GE10__A_FC LINK_CONFIG__A_FC__A_GE10__A_FC__A_GE10 LINK_CONFIG__0__A_FC__0__A_FC	GE [10G] FC [1G,4G,8G, 16G]	×	×	×
LINK_CONFIG__J_GE10__0__0__0 LINK_CONFIG__0__0__0__J_GE10 LINK_CONFIG__0__J_GE10__0__0 LINK_CONFIG__0__0__J_GE10__J_GE10 LINK_CONFIG__J_GE10__J_GE10__0__0 LINK_CONFIG__J_GE10__J_GE10__J_GE10__J_GE10	10G	×	×	×
LINK_CONFIG__J_GE10__0__J_GE10__0 LINK_CONFIG__J_GE10__0__0__0 LINK_CONFIG__0__0__J_GE10__0	10G	×	10,25G	×
LINK_CONFIG__0__0__0__J_FC LINK_CONFIG__0__J_FC__0__0 LINK_CONFIG__0__0__J_FC__J_FC	1G,4G,8G, 16G	×	8G,16G, 32G	×

LINK_CONFIG__J_FC__J_FC__0__0 LINK_CONFIG__J_FC__J_FC__J_FC__J_FC				
LINK_CONFIG__J_FC__0__0__0 LINK_CONFIG__0__0__J_FC__0	1,4,8, 16G	×	8G,16G, 32G	×
LINK_CONFIG__J_FC__00__J_GE10__0 LINK_CONFIG__J_GE10__0__J_FC__0 LINK_CONFIG__J_FC__J_FC__J_GE10__J_GE10 LINK_CONFIG__J_GE10__J_GE10__J_FC__J_FC LINK_CONFIG__AJAFC__0__AJAGE10__0 LINK_CONFIG__AJAGE10__0__AJAFC__0 LINK_CONFIG__J_GE10__J_FC__J_GE10__J_FC LINK_CONFIG__J_FC__J_GE10__J_FC__J_GE10 LINK_CONFIG__0__AJAGE10__0__AJAGE10 LINK_CONFIG__0__AJAFC__0__AJAFC LINK_CONFIG__0__AJAFC__0__0 LINK_CONFIG__0__AJAFC__0__AJAGE10 LINK_CONFIG__0__AJAGE10__0__0 LINK_CONFIG__0__AJAGE10__0__AJAFC	GE [10G] FC [1,4,8, 16G]	×	×	×
LINK_CONFIG__AJAGE10__0__AJAGE10__0 LINK_CONFIG__AJAGE10__0__0__0	10G	×	10,25G	×
LINK_CONFIG__AJAFC__0__AJAFC__0 LINK_CONFIG__AJAFC__0__0__0	1G,4G,8G, 16G	×	8G,16G, 32G	×
LINK_CONFIG__AJAGE10__0__AJAGE10__0 LINK_CONFIG__AJAGE10__0__0__0	10G	×	10,25G	×
LINK_CONFIG__AJAFC__0__AJAFC__0 LINK_CONFIG__AJAFC__0__0__0	1G,4G,8G, 16G	×	8G,16G, 32G	×
LINK_CONFIG__A_FC64__0__0__0	×	64G	64G	×
LINK_CONFIG__A_GE40__0__0__0 LINK_CONFIG__JA_GE40__0__0__0 LINK_CONFIG__J_GE40__0__0__0 LINK_CONFIG__AJ_GE40__0__0__0	40G	×	×	×
LINK_CONFIG__A_GE25__A_GE25__A_GE25__A_GE25 LINK_CONFIG__A_GE25__A_GE25__0__0 LINK_CONFIG__0__0__A_GE25__A_GE25 LINK_CONFIG__0__A_GE25__0__0 LINK_CONFIG__0__0__0__A_GE25	×	10G,25G	10,25G	×
LINK_CONFIG__A_GE50PAM4__0__0__0	×	50GPAM4	50GPAM4	×

LINK_CONFIG__A_GE50__0__0__0	×	50G	50G	×
LINK_CONFIG__J_GE50__0__0__0	×	×	50G	×
LINK_CONFIG__A_GE100__0__0__0	×	40G,100G	40G,100G	×
LINK_CONFIG__J_GE100__0__0__0	×	×	40G,100G	×
LINK_CONFIG__QSFP__AJ_GE50__0 LINK_CONFIG__QSFP__AJAGE50__0 LINK_CONFIG__QSFP__A_50G__0	×	×	×	50G
LINK_CONFIG__QSFP__A_50GPAM4__A_50GPAM4 LINK_CONFIG__0__A_50GPAM4__0__A_50GPAM4 LINK_CONFIG__A_50GPAM4__0__A_50GPAM4__0 LINK_CONFIG__AJ_50GPAM4__0__AJ_50GPAM4__0 LINK_CONFIG__JA_50GPAM4__0__JA_50GPAM4__0	×	×	×	50PAM4 , 10G,25G
LINK_CONFIG__A_64GPAM4__0__A_64GPAM4__0 LINK_CONFIG__0__A_64GPAM4__0__A_64GPAM4 LINK_CONFIG__JA_64GPAM4__0__JA_64GPAM4__0 LINK_CONFIG__AJ_64GPAM4__0__AJ_64GPAM4__0	×	×	×	16, 32, 64G

TABLE 2.7: ELinkConfiguration Special Values

Value	Description
LINK_CONFIG__INVALID	No valid link configuration.
LINK_CONFIG_0_0_0_0	All ports are unconfigured.

enum TLNetAPI.EConnectionType

This enumeration describes the analyzer connection type.

TABLE 2.8: Analyzer Connection Type

Value	Description
CONNECTION_TYPE_NONE	No valid connection to device.
CONNECTION_TYPE_USB	USB connection to device.
CONNECTION_TYPE_TCP	Ethernet connection to device.

enum TLNetAPI.ERecordingStatus

This enumeration describes the analyzer's recording status which returned by status report event.

TABLE 2.9: Analyzer's Recording Status

Value	Description
RECORDING_STATUS_NONE	No valid recording status.
RECORDING_STATUS_IDLE	Recording is idle, not in progress.
RECORDING_STATUS_PROGRAMMING	Recording is in the settings programming phase.
RECORDING_STATUS_UPLOADING	Recording is in the uploading phase.
RECORDING_STATUS_WAITING_TRIGGER	Recording is waiting for a trigger event.
RECORDING_STATUS_RECORDING_TRIGGERED	Recording is triggered.
RECORDING_STATUS_CASCADING_PENDING	Recording start is waiting for other connected analyzers in the chain.

enum TLNetAPI.EJammerSessionStatus

This enumeration describes the jammer's status which return by status report event.

TABLE 2.10: Jammer Session Status

Value	Description
JAMMER_SESSION_STATUS_NONE	No valid jammer status.
JAMMER_SESSION_STATUS_IDLE	Jammer is idle, not in progress.
JAMMER_SESSION_STATUS_RUNNING	Jammer is running.
JAMMER_SESSION_STATUS_PROGRAMMING	Jammer is in the scenario programming phase.

enum TLNetAPI.ETriggerMode

This enumeration describes the trigger mode.

TABLE 2.11: Trigger Mode

Value	Description
TRIGGER_MODE_NONE	No valid trigger mode.
TRIGGER_MODE_SNAPSHOT	Snapshot trigger mode.
TRIGGER_MODE_PATTERN	Event pattern trigger mode.

enum TLNetAPI.EAnalyzerSpeed

This enumeration describes the analyzer link speed.

TABLE 2.12: Analyzer Link Speed

Value	Description
ANALYZER_SPEED_AUTO	Automatically sets the speed
ANALYZER_SPEED_INVALID	No valid speed
ANALYZER_SPEED_UNKNOWN	Unrecognized speed
ANALYZER_SPEED_10	10G Ethernet
ANALYZER_SPEED_25	25G Ethernet
ANALYZER_SPEED_40	40G Ethernet
ANALYZER_SPEED_50	50G Ethernet
ANALYZER_SPEED_50_PAM4	50G Ethernet with PAM4 speeds
ANALYZER_SPEED_100	100G Ethernet
ANALYZER_SPEED_100_PAM4	100G Ethernet with PAM4 speeds
ANALYZER_SPEED_100_PAM4_1LANE	100G Ethernet 1 Lane
ANALYZER_SPEED_200	200G Ethernet
ANALYZER_SPEED_200_PAM4_2LANE	200G Ethernet 2 Lanes
ANALYZER_SPEED_400_PAM4_4LANE	400G Ethernet 4 Lanes
ANALYZER_SPEED_800_PAM4	800G Ethernet with PAM4 speeds
ANALYZER_SPEED_1	1G FC
ANALYZER_SPEED_1_PLUS_400	For FC and a set speed 1G, use this constant. For GE, it is used for 400G L1.
ANALYZER_SPEED_2	2G FC
ANALYZER_SPEED_4	4G FC
ANALYZER_SPEED_8	8G FC
ANALYZER_SPEED_AUTO_8	8G FC Automatically selected
ANALYZER_SPEED_16	16G FC
ANALYZER_SPEED_32	32G FC

Value	Description
ANALYZER_SPEED_64_PAM4	64G FC with PAM4 speeds
ANALYZER_SPEED_128	128G FC

enum TLNetAPI.EICActionType

This enumeration describes the types of jammer actions.

TABLE 2.13: Types of Jammer Actions

Value	Description
IC_ACTION_GE_FRAME_REMOVE_JAM	Frame removal for Ethernet
IC_ACTION_GE_FRAME_TRUNCATE_JAM	Frame truncation for Ethernet
IC_ACTION_GE_CAPTURE_DWORD	Capture DWORD for Ethernet
IC_ACTION_GE_CAPTURE_FRAME	Capture frame for Ethernet
IC_ACTION_GE_INSERT_PRECAPTURE_FRAME	Insert pre-captured frame for Ethernet
IC_ACTION_GE_INSERT_FRAME	Insert frame for Ethernet
IC_ACTION_GE_INSERT_SYMBOL_66_BITS	Insert 66-bit symbol for Ethernet
IC_ACTION_GE_INSERT_BYTES	Insert raw bytes for Ethernet
IC_ACTION_GE_INSERT_BYTES_INSIDE_FRAME	Insert bytes within frame for Ethernet
IC_ACTION_GE_FRAME_MODIFY_JAM	Modify frame for Ethernet
IC_ACTION_GE_FRAME_REPLACE_JAM	Replace frame for Ethernet
IC_ACTION_GE_INJECT_ERROR_TYPE_FEC_PARITY	Inject FEC Parity Error for Ethernet. (applicable on M408) Parameter for this action shall be set to zero.
IC_ACTION_GE_INJECT_ERROR_TYPE_SYNC_HEADER	Insert sync header error for Ethernet (applicable on M408) Parameter for this action shall be set to zero.
IC_ACTION_GE_SUBS_SYMBOL_16G_JAM	Replace 66-bit symbol for Ethernet
IC_ACTION_GE_DELETE_SYMBOL_16G_JAM	Delete 66-bit symbol for Ethernet
IC_ACTION_AUTO_NEGOTIATION	Auto negotiation packet jam

Value	Description
IC_ACTION_SUBS_WITH_PRECAPTURED_SYMBOL_16G_JAM	Replace with precaptured 66-bit symbol for Ethernet
IC_ACTION_SUBS_SUBS_IDLE_SYMBOL_16G_JAM	Replace 66-bit symbol with Idle for Ethernet
IC_ACTION_GE_INJECT_ERROR_TYPE_SYMBOL_VIOLATION	Inject symbol error for Ethernet. (applicable on M408) Parameter for this action shall be set to zero.
IC_ACTION_GE_INJECT_ERROR_TYPE_BLOCK	Inject block type error for Ethernet. (applicable on M408) Parameter for this action shall be set to zero.
IC_ACTION_GE_INJECT_ERROR_TYPE_ALIGNMENT	Inject alignment Marker error for Ethernet. applicable on M328/M328Q/M648) Use APIActionInjectErrorAlignment as parameter for this action.
IC_ACTION_GE_INJECT_ERROR_TYPE_ORDERSET	Inject invalid ordered set for Ethernet. (applicable on M408) Parameter for this action shall be set to zero.
IC_ACTION_BEEP	Beep
IC_ACTION_MONITOR_COUNT	Monitor/count
IC_ACTION_RECONNECT	Reconnect link
IC_ACTION_DISCONNECT	Disconnect link
IC_ACTION_STOP	Stop jammer
IC_ACTION_RESTART_ALL	Restart all sequencers
IC_ACTION_RESTART_CURRENT_SEQ	Restart current sequencers
IC_ACTION_BRANCH	Branch sequencer to another state
IC_ACTION_ANALYZER_TRIGGER	Trigger analyzer
IC_ACTION_TRIGGER_OUT	Trigger external output
IC_ACTION_FC_INSERT_FRAME	Insert frame for RC
IC_ACTION_FC_INSERT_SYMBOL_66_BITS	Insert 66-bit symbol for FC
IC_ACTION_FC_INSERT_BYTES	Insert raw bytes for FC
IC_ACTION_FC_INSERT_BYTES_INSIDE_FRAME	Insert bytes within frame for FC

Value	Description
IC_ACTION_FC_FRAME_MODIFY_JAM	Modify frame for FC
IC_ACTION_FC_FRAME_REPLACE_JAM	Replace frame for FC
IC_ACTION_FC_INJECT_ERROR_TYPE_FEC_PARITY	Inject FEC parity error for FC. (applicable on M408) Parameter for this action shall be set to zero.
IC_ACTION_FC_INJECT_ERROR_TYPE_SYNC_HEADER	Inject sync header error for FC. (applicable on M408/M328/M328Q/M648) Use APIActionInjectErrorSymbol as parameter for this action.
IC_ACTION_FC_SUBS_SYMBOL_16G_JAM	Replace 66-bit symbol for FC
IC_ACTION_FC_DELETE_SYMBOL_16G_JAM	Delete 66-bit symbol for FC
IC_ACTION_FC_FRAME_REMOVE_JAM	Remove frame for FC
IC_ACTION_FC_FRAME_TRUNCATE_JAM	Truncate frame for FC
IC_ACTION_FC_INJECT_ERROR_TYPE_10b_CODE	Inject invalid 10-bit code for FC. (applicable on M408/M328/M328Q) Parameter for this action shall be set to zero.
IC_ACTION_FC_INJECT_ERROR_TYPE_RD	Inject RD error for FC. (applicable on M408/M328/M328Q) Parameter for this action shall be set to zero.
IC_ACTION_DELETE_ORDERSET_JAM	Delete ordered set
IC_ACTION_REMOVE_ORDERSET_JAM	Remove ordered set
IC_ACTION_SUBS_ORDERSET_JAM	Replace ordered set
IC_ACTION_CAPTURE_DATA_DWORD	Capture DWORD
IC_ACTION_REPLACE_DWORD_GENERAL	Replaces a DWORD in event with value and mask
IC_ACTION_GE_INJECT_ERROR_TYPE_DELIMITER	Inject frame delimiter error for Ethernet (applicable on M408)
IC_ACTION_GE40_INSERT_FRAME	Insert frame for 40G Ethernet
IC_ACTION_GE40_INSERT_SYMBOL_66_BITS	Insert 66-bit symbol for 40G Ethernet
IC_ACTION_GE40_INSERT_PRECAPTURE_FRAME	Insert precaptured frame for 40G Ethernet

Value	Description
IC_ACTION_GE40_SUBS_SYMBOL_16G_JAM	Replace 66-bit symbol for 40G Ethernet
IC_ACTION_GE40_DELETE_SYMBOL_16G_JAM	Delete 66-bit symbol for 40G Ethernet
IC_ACTION_GE40_SUBS_SUBS_IDLE_SYMBOL_16G_JAM	Replace 66-bit symbol with Idle for 40G Ethernet
IC_ACTION_GE40_INJECT_ERROR_TYPE_SYNC_HEADER	Inject sync header error for 40G Ethernet. (applicable on M408) Parameter for this action shall be set to zero.
IC_ACTION_GE40_INJECT_ERROR_TYPE_FEC_PARITY	Inject FEC parity error for 40G Ethernet. (applicable on M408) Parameter for this action shall be set to zero.
IC_ACTION_GE40_INJECT_ERROR_TYPE_BLOCK	Inject block type error for 40G Ethernet. (applicable on M408) Parameter for this action shall be set to zero.
IC_ACTION_GE40_INJECT_ERROR_TYPE_DELIMITER	Inject frame delimiter error for 40G Ethernet. (applicable on M408) Parameter for this action shall be set to zero.
IC_ACTION_FC_CAPTURE_FRAME	Capture frame for FC
IC_ACTION_FC_INSERT_PRECAPTURE_FRAME	Insert pre-captured frame for FC
IC_ACTION_ST_ANALYZER_MARKER_TRIGGER	Insert analyzer marker
IC_ACTION_GE25_INJECT_ERROR_TYPE_SYNC_HEADER	Inject sync header error for Ethernet (applicable on M328/M328Q/M648). Use APIActionInjectErrorSymbol as parameter for this action.
IC_ACTION_GE25_INJECT_ERROR_TYPE_DELIMITER	Inject frame delimiter error for Ethernet (applicable on M328/M328Q/M648). Parameter for this action shall be set to zero.
IC_ACTION_GE25_INJECT_ERROR_TYPE_BLOCK	Inject block type error for Ethernet (applicable on M328/M328Q/M648). Use APIActionInjectErrorSymbol as parameter for this action.
IC_ACTION_INJECT_ERROR_TYPE_BASERFEC_PARITY	Inject Base R FEC error for FC/Ethernet. (applicable on M328/M328Q/M648) Use APIActionInjectErrorFEC as parameter for this action.

Value	Description
IC_ACTION_INJECT_ERROR_TYPE_RSFEC_PARITY	Inject Base R FEC error for FC/Ethernet. (applicable on M328/M328Q/M648) Use APIActionInjectErrorFEC as parameter for this action.
IC_ACTION_GE_INJECT_ERROR_TYPE_ALIGNMENT_BIP	Inject alignment BIP error for Ethernet. applicable on M648) Use APIActionInjectErrorAlignment as parameter for this action.

enum TLNetAPI.**VSEStatus**

This enumeration describes the types of script status.

TABLE 2.14: Script Status

Value	Description
Idle	The script engine is idle.
Enqueued	The script is enqueued for execution.
Running	The script is running.
Success	The script execution completed normally, and the result is Success.
Failed	The script execution completed normally, and the result is Failed.
Error	The script execution completed abnormally with an error.

enum TLNetAPI.**ECaptureStrategy**

This enumeration has the capture/filtering types

TABLE 2.15: Capture/Filtering types

Value	Description
CAPTURE_EVERYTHING	Capture everything.
CAPTURE_PATTERN_INCLUDE	Capture only specified patterns.
CAPTURE_PATTERN_EXCLUDE	Capture all except specified patterns.

enum TLNetAPI.**EJammerSequencer**

This enumeration has the jammer sequencer ids

TABLE 2.16: Jammer Sequencer ids

Value	Description
JAMMER_GLOBAL_SEQUENCER	The global jammer sequencer with a single state.
JAMMER_SEQUENCER_0	The first jammer sequencer with 32 states.
JAMMER_SEQUENCER_1	The second jammer sequencer with 32 states.

enum TLNetAPI.**EDeviceType**

This enumeration has the device types.

TABLE 2.17: Devices Types

Value	Description
DEVICE_TYPE_M328Q	SierraNet M328Q
DEVICE_TYPE_M328	SierraNet M328
DEVICE_TYPE_T328	SierraNet T328
DEVICE_TYPE_M408	SierraNet M408
DEVICE_TYPE_M168	SierraNet M168
DEVICE_TYPE_M164	SierraFC M164

enum TLNetAPI.**PatternType**

This enumeration has the event types for triggering, filtering, and jammer.

TABLE 2.18: Event Types for Triggering, Filtering and Jammer

Value	Analyzer/ Jammer	Supported Devices	Description
ID_PATTERN_TYPE_FC_TIMER	All	All	Timer event type for FC scenarios.
ID_PATTERN_TYPE_GE_TIMER	All	M168/M408/ T328/M328/ M328Q	Timer event type for Ethernet.
ID_PATTERN_TYPE_IPG	Analyzer only	All	IPG event type.
ID_PATTERN_TYPE_FC_SYMBOL_66_BITS	All	All	66-bit symbol event type for FC.
ID_PATTERN_TYPE_FC_SYMBOL_8G	Jammer only	M164/M168/ M408/M328/ M328Q	10-bit symbol event type for FC.
ID_PATTERN_TYPE_FC_ORDERSET	All	All	Ordered set event type for FC.
ID_PATTERN_TYPE_GE_SYMBOL_66_BITS	All	M168/M408/ T328/M328/ M328Q	66-bit symbol event type for Ethernet.
ID_PATTERN_TYPE_FC_CONNECT_DISCONNECT	Jammer only	M164/M168/ M408/M328/ M328Q	Connect/Disconnect link event type for FC.
ID_PATTERN_TYPE_CONNECT_DISCONNECT	Jammer only	M168/M408/ M328/M328Q	Connect/Disconnect link event type for Ethernet.
ID_PATTERN_TYPE_GE_40G_CONNECT_DISCONNECT	Jammer only	M408/M328/ M328Q	Connect/Disconnect link event type for 40GbE.
ID_PATTERN_TYPE_FC16_PROTOCOL_ERRORS	All	All	Protocol error type for FC 16G.
ID_PATTERN_TYPE_GE10_PROTOCOL_ERRORS	All	M168/M408/ T328/M328/ M328Q	Protocol error type for 10GbE.
ID_PATTERN_TYPE_GE25_PROTOCOL_ERRORS	All	T328/M328/ M328Q	Protocol error type for 25GbE.

Value	Analyzer/ Jammer	Supported Devices	Description
ID_PATTERN_TYPE_GE40_PROTOCOL_ERRORS	All	M408/T328/ M328/M328Q	Protocol error type for 40GbE.
ID_PATTERN_TYPE_GE50_PROTOCOL_ERRORS	All	T328/M328/ M328Q	Protocol error type for 50GbE.
ID_PATTERN_TYPE_GE100_PROTOCOL_ERRORS	All	T328/M328/ M328Q	Protocol error type for 100GbE.
ID_PATTERN_TYPE_EXTERNAL_TRIGS	All	All	External trigger event type.
ID_PATTERN_TYPE_OTHER_TRIGS	Jammer only	M164/M168/ M408/M328/ M328Q	Other triggers event type.
ID_PATTERN_TYPE_GE_AUTO_NEGOTIATION_IEEE_802	All	M168/M408/ T328/M328/ M328Q	Auto-negotiation event type for Ethernet.
ID_PATTERN_TYPE_FC_PHYSICAL_TRAINING_SEQUENCE	Analyzer only	All	Training frame event type for FC.
ID_PATTERN_TYPE_GE_PHYSICAL_TRAINING_SEQUENCE	Analyzer only	All	Training frame event type for Ethernet.
ID_PATTERN_TYPE_GE_PHYSICAL_TRAINING_SEQUENCE_ PAM4	Analyzer only	T328	Training frame event type for PAM4 Ethernet.
ID_PATTERN_TYPE_BOTH_LINKS_UP	Jammer only	M164/M168/ M408/M328/ M328Q	Link up event type.
ID_PATTERN_TYPE_FC8_LINK_SPEED	Jammer only	M164/M168/ M408/M328/ M328Q	Link speed event type for FC 8G.
ID_PATTERN_TYPE_FC16_LINK_SPEED	Jammer only	M164/M168/ M408/M328/ M328Q	Link speed event type for FC 16G.
ID_PATTERN_TYPE_GE10_LINK_SPEED	Jammer only	M168/M408/ M328/M328Q	Link speed event type for 10GbE.
ID_PATTERN_TYPE_GE25_LINK_SPEED	Jammer only	M328/M328Q	Link speed event type for 25GbE.
ID_PATTERN_TYPE_GE40_LINK_SPEED	Jammer only	M408/M328/ M328Q	Link speed event type for 40GbE.

Value	Analyzer/ Jammer	Supported Devices	Description
ID_PATTERN_TYPE_FC_ANY_FC_FRAME	All	All	General frame event type for FC.
ID_PATTERN_ANY_ETHERNET_FRAME	All	M168/M408/ T328/M328/ M328Q	General frame event type for Ethernet.
ID_PATTERN_TYPE_FC_DWORD_MATCHER	Jammer only	All	Match Dword for any frame/symbol

enum TLNetAPI.**EGEProtocolError**

This enumeration has the Ethernet protocol error types.

TABLE 2.19: Ethernet Protocol Error Types

Value	Description
ID_PE_GE_DELIMITER_ERROR	Frame delimiter error.
ID_PE_GE_FRAME_LENGTH_ERROR	Frame length error.
ID_PE_GE_ETHERNET_CRC_ERROR	FCS error.
ID_PE_GE_FC_CRC_ERROR	FCoE CRC error.
ID_PE_GE_BLOCK_TYPE_ERROR	Invalid 66-bit block type.
ID_PE_GE_ORDER_SET_ERROR	Invalid ordered set.
ID_PE_GE_ALIGNMENT_ERROR	Alignment error.
ID_PE_GE_SYNC_HEADER_ERROR	Invalid 66-bit block sync header.
ID_PE_GE_FEC_ERROR	Uncorrectable FEC error.
ID_PE_GE_AUTO_NEG_FRAME_MARKER_ERROR	Auto-negotiation packet marker error.
ID_PE_GE_AUTO_NEG_MANCHESTER_ERROR	Auto-negotiation packet Manchester encoding error.
ID_PE_GE_MARKER_INTERVAL_ERROR	Marker interval error.
ID_PE_GE_TRAINING_FRAME_MARKER_ERROR	Training frame marker error.
ID_PE_GE_TRAINING_MANCHESTER_ERROR	Training frame Manchester encoding error.

enum TLNetAPI.**EFCProtocolError**

This enumeration has the FC protocol error types.

TABLE 2.20: FC protocol error types

Value	Description
ID_PE_FC_SYMBOL_VIOLATION	10-bit symbol violation.
ID_PE_FC_DISPARITY_ERROR	10-bit symbol disparity error.
ID_PE_FC_SPACING_ERROR	Spacing error.
ID_PE_FC_ALIGNMENT_ERROR	Alignment error.
ID_PE_FC_DELIMITER_ERROR	Frame delimiter error.
ID_PE_FC_EOF_ERROR	Invalid EOF.
ID_PE_FC_PRIMITIVE_ERROR	Invalid Primitive.
ID_PE_FC_FRAME_LENGTH_ERROR	Frame length error.
ID_PE_FC_CRC_ERROR	CRC error.
ID_PE_FC_SYNC_HEADER_ERROR	Invalid 66-bit block sync header.
ID_PE_FC_FEC_PARITY_ERROR	Uncorrectable FEC error.
ID_PE_FC_ILLEGAL_PRIMITIVE_ERROR	Illegal Primitive.
ID_PE_FC_TRAINING_FRAME_MARKER_ERROR	Training frame marker error.
ID_PE_FC_TRAINING_MANCHESTER_ERROR	Training frame Manchester encoding error.

Enum TLNetAPI.**IPGControl**

This enumeration has the comparison operations for IPG events.

TABLE 2.21: Comparison Operations for IPG Events

Value	Description
ID_IPG_CONTROL_EQUAL_TO	Measured IPG Equal to settings value.

Value	Description
ID_IPG_GREATER_THAN	Measured IPG Greater than settings value.
ID_IPG_LESS_THAN	Measured IPG Less than settings value.
ID_IPG_GREATER_THAN_EQUAL_TO	Measured IPG Greater than equal to settings value.
ID_IPG_LESS_THAN_EQUAL_TO	Measured IPG Less than equal to settings value.

Enum TLNetAPI.**EapiEthernetHeader**

This enumeration has the optional header types for Ethernet frame detection.

TABLE 2.22: Header Types for Ethernet Frame Detection

Value	Description
DEFAULT	No extra headers.
VN_HDR	VN header.
CN_HDR	CN header.
VLAN_HDR	VLAN header.
ISL_HDR	ISL header.
VXLAN_HDR	VXLAN header.
NVGRE_HDR	NVGRE header.
VSAN_FC	VSAN FC header.

Enum TLNetAPI.**EtimerUnits**

This enumeration has the value units for timer events.

TABLE 2.23: Value Units for Timer Events

Value	Description
TIMER_MILISEC	Millisecond units.
TIMER_MICROSEC	Microsecond units.

Value	Description
TIMER_SEC	Second units.
EXTERNAL	External trigger.
ANALYZER	Analyzer internal trigger.
TIMER_DWORD_OR_SYMBOL	Number of dwords or symbol.
TIMER_NANOSEC	Nanosecond units.

Enum TLNetAPI.**EotherTrigger**

This enumeration has the other trigger action types.

TABLE 2.24: Other Trigger Action

enum TLNetAPI.**EbeepValues**

This enumeration has the beep action durations.

TABLE 2.25: Beep Action Duration

Value	Description
Duration_1ms	1 ms
Duration_2ms	2 ms
Duration_5ms	5 ms
Duration_10ms	10 ms
Duration_25ms	25 ms
Duration_50ms	50 ms
Duration_100ms	100 ms
Duration_250ms	250 ms
Duration_500ms	500 ms
Duration_1s	1 s
Duration_2s	2 s
Duration_4s	4 s

enum TLNetAPI.**EdwordCaptureRegisterIndex**

This enumeration has the jammer dword capture registers.

TABLE 2.26: Jammer Dword Capture Registers

Value	Description
DwordRegister_0	Dword register 0
DwordRegister_1	Dword register 1
DwordRegister_2	Dword register 2
DwordRegister_3	Dword register 3

enum TLNetAPI.**EframeCaptureRegisterIndex**

This enumeration has the jammer frame capture registers.

TABLE 2.27: Jammer Frame Capture Registers

Value	Description
FrameRegister_0	Frame register 0
FrameRegister_1	Frame register 1
FrameRegister_2	Frame register 2
FrameRegister_3	Frame register 3
FrameRegister_4	Frame register 4
FrameRegister_5	Frame register 5
FrameRegister_6	Frame register 6

enum TLNetAPI.**EIPMode**

This enumeration has the device IP modes.

TABLE 2.28: Device IP Models

Value	Description
STATIC_IP_MODE	Static IP mode
DHCP_IP_MODE	DHCP IP mode

enum TLNetAPI.**EApiSOFOrdersets**

This enumeration has the FC SOF ordered sets.

TABLE 2.29: FC SOF Ordered Sets

Value	Description
FC_ORDERSET_SOF_ANY	Any SOF
FC_ORDERSET_SOFc1	SOFc1
FC_ORDERSET_SOFi1	SOFi1
FC_ORDERSET_SOFn1	SOFn1
FC_ORDERSET_SOFi2	SOFi2
FC_ORDERSET_SOFn2	SOFn2
FC_ORDERSET_SOFi3	SOFi3
FC_ORDERSET_SOFn3	SOFn3
FC_ORDERSET_SOFc4	SOFc4
FC_ORDERSET_SOFi4	SOFi4
FC_ORDERSET_SOFn4	SOFn4
FC_ORDERSET_SOFf	SOFf

enum TLNetAPI.EApiEOFOrdersets

This enumeration has the FC EOF ordered sets.

TABLE 2.30: FC EOF Ordered Sets

Value	Description
FC_ORDERSET_EOF_ANY	Any EOF
FC_ORDERSET_EOFt	EOFt
FC_ORDERSET_EOFdt	EOFdt
FC_ORDERSET_EOFa	EOFa
FC_ORDERSET_EOFn	EOFn
FC_ORDERSET_EOFni	EOFni
FC_ORDERSET_EOFdti	EOFdti
FC_ORDERSET_EOFrt	EOFrt
FC_ORDERSET_EOFrti	EOFrti

enum TLNetAPI.EExerciserSessionStatus

This enumeration describes the exerciser status which return by status report event.

TABLE 2.31: Exerciser Session Status

Value	Description
EXERCISER_SESSION_STATUS_NONE	No valid exerciser status.
EXERCISER_SESSION_STATUS_IDLE	exerciser is idle, not in progress.
XERCISER_SESSION_STATUS_RUNNING	Exerciser is running.
EXERCISER_SESSION_STATUS_PROGRAMMING	Exerciser is in the programming phase.
EXERCISER_SESSION_STATUS_PAUSE	Exerciser is paused.

Enum TLNetAPI. ETrainingSignalPackMode

This enumeration has the value units for the training signal's packing mode.

TABLE 2.31: Value Units for Training Pack Mode

Value	Description
UnPacked	Training signal is unpacked.
Packed	Training signal is packed.

Enum TLNetAPI. **EApiOrderedSetGroup**

This enumeration has the value for the training signal's packing mode.

TABLE 2.32: Value Units for Ordered Set Group

Value	Description
AnySOF	Any type of SOF
AnyEOF	Any type of EOF
AnyIDLE	Any type of IDLE
AnyARB	Any type of ARB
AnySYN	Any type of SYN
AnyOPN	Any type of OPN
AnyLBP	Any type of LBP
AnyFillWordGroup	Any type of FillWordGroup
AnyLIP	Any type of LIP
AnyVS_BB_SCs	Any type of VS_BB_SCs
AnyVS_BB_SCr	Any type of VS_BB_SCr
AnyER_RDY	Any type of ER_RDY
Undefined	Undefined

Enum TLNetAPI. EInsertPosition

This enumeration has the value for the Insert position.

TABLE 2.33: Value Units for EInsertPosition

Value	Description
AfterCurrentDowrd	After current dword.
OffsetFromFrameStart	Offset from the frame start.

Enum TLNetAPI. ELinkSpeed

This enumeration has the value for the Link Speed, to be used in the link speed event.

TABLE 2.34: Value Units for ELinkSpeed

Value	Description
FC_LINK_1_GBPS	FC Link of 1Gbps
FC_LINK_2_GBPS	FC Link of 2Gbps
FC_LINK_4_GBPS	FC Link of 4Gbps
FC_LINK_8_GBPS	FC Link of 8Gbps
FC_LINK_16_GBP	FC Link of 16Gbp
FC_LINK_16_GBPS_WITH_FEC	FC Link of 16Gbps with fec
FC_LINK_16_GBPS_WITHOUT_FEC	FC Link of 16Gbps without fec
GE_LINK_10_GIG	GE Link of 10Gig
GE_LINK_10_GIG_WITH_FEC	GE Link of 10Gig with fec
GE_LINK_10_GIG_WITHOUT_FEC	GE Link of 10Gig without fec
GE_LINK_40_GIG	GE Link of 40Gig
GE_LINK_40_GIG_WITH_FEC	GE Link of 40Gig with fec

TABLE 2.35t 2: In Frame Dword Replace/Inject error sub actions

EApiReplaceDWORDAction.REPLACE_WITH_CRC	Replace Dword with CRC. Applicable for both FC and Ethernet frames. Supported jam action: IC_ACTION_GE_FRAME_MODIFY_JAM IC_ACTION_GE_FRAME_REPLACE_JAM IC_ACTION_FC_FRAME_MODIFY_JAM IC_ACTION_FC_FRAME_REPLACE_JAM
EApiReplaceDWORDAction.REPLACE_WITH_FCS	Replace Dword with FCS. Applicable for Ethernet frames. Supported GE jam action: IC_ACTION_GE_FRAME_MODIFY_JAM IC_ACTION_GE_FRAME_REPLACE_JAM
EApiReplaceDWORDAction.REPLACE_WITH_IDLE	Replace Dword with IDLE. Applicable for FC frames. Supported FC jam action: IC_ACTION_FC_FRAME_MODIFY_JAM IC_ACTION_FC_FRAME_REPLACE_JAM
EApiReplaceInjectErrorAction.REPLACE_WITH_ERROR_CODE_VIOLATION	Inject byte error type code violation for FC frames. Supported FC jam action (M328 and M408 only): IC_ACTION_FC_FRAME_MODIFY_JAM IC_ACTION_FC_FRAME_REPLACE_JAM
EApiReplaceInjectErrorAction.REPLACE_WITH_ERROR_DISPARITY	Inject byte error type Disparity for FC frames. Supported FC jam action (M328 and M408 only): IC_ACTION_FC_FRAME_MODIFY_JAM IC_ACTION_FC_FRAME_REPLACE_JAM

TLNetAPI.CreateProject()

Creates a new project.

Return value: APIProject object for the new project.

TLNetAPI.GetDeviceManager()

Gets the Device Manager.

Return value: APIDeviceManager object.

TLNetAPI.GetHardwareInfo(serial_number)

Returns an integer that is the hardware model ID for the connected device with the specified serial_number.

TLNetAPI.GetVersion()

Returns a string containing the software version and build number in "x.y.z" format, where x is major version number, y is minor version number, and z is build number.

TLNetAPI.SetTempFolder(temp_path, saved)

Sets temp folder path.

Parameter:

Temp_path:	string, specifies temp folder path.
Saved:	boolean, if it is true, given temp path saved in software preferences, otherwise just considered as a temporary change of temp folder for current running session.

TLNetAPI.OpenFile(filename, OnUpdateTraceProgress = None)

Opens the trace file 'filename'. Returns an **APITrace** object that represents that trace.

'OnUpdateTraceProgress ' is callback for progress reporting. You can use None if you don't need progress events. Otherwise, you must pass your own function as specified below.

Callback function details:

OnUpdateTraceProgress (progress)

This callback function is called to provide progress updates on the progress of updating trace(if is needed). 'progress' is an integer from 0 to 100 that indicates the percentage of completion of the updating process.

TLNetAPI.OpenProject(filename)

Opens the project file 'filename'. Returns an **APIProject** object that represents that project.

TLNetAPI.shutdown()

Finalize and shutdown TLNetAPI. If you call Python "exit" command in the script explicitly, you must call this function before calling "exit".

2.3 APITrace

This class represents an opened trace file.

Class TLNetAPI.APITrace

The **APITrace** class has the following methods:

Close()

Close the opened trace file. Note: This will not release the memory.

GetStartDateTimestamp()

Returns a time_t integer that is the start date timestamp for the trace.

GetEndDateTimestamp()

Returns a time_t integer that is the end date timestamp for the trace.

GetTriggerDateTimestamp()

Returns a time_t integer that is the trigger date timestamp for the trace.

Retruned timestamp represents an integer value which is number of seconds elapsed since 00:00 hours, Jan 1, 1970 UTC.

GetCount()

Returns an integer that is the number of packets in the trace.

GetPacket(index)

Return APITracePacket object by a given index. The index is 1-based.

ExportToCSV(file_name, OnExportStatus, viewer_type, split_to_multiple_files, column_list)

Parameters:

file_name:	string, specifies the full path of the output csv file.
OnExportStatus:	callback function to status reporting, set to None if you don't need status report.
viewer_type:	Set to TLNetAPI.EViewerType.SPREAD_SHEET

split_to_multiple_files: bool, if True, it generates multiple csv file, each one contains 1000 records.

column_list: string or integer list, specifies list of column indexes or names to be exported. Set to None means all columns. List of columns can be either integer, which specifies the column index, or string, which is the column name. When using string, it requires putting the exact same string as in Spreadsheet View. For example, “No” and “Time Delta” are not going to be exported but they need to be “No.” and “Time Delta (ns)”.

Callback function details:

OnExportStatus(progress)

This callback function is called to provide status updates on the progress of an export operation. ‘progress’ is an integer from 0 to 100 that indicates the percentage of completion of the export.

ExportToWireshark(file_name, protocol, OnExportStatus)

Export the opened trace file to Wireshark format. The second parameter specifies the protocol, and can be one of the values from table 32.

The third parameter is the callback for status reporting. You can use None if you don’t need status events. Otherwise, you must pass your own function as specified below.

Table 32: enum TLNetAPI. EAPILinkProtocol

Value	Description
API_LINK_PROTOCOL_ETHERNET	Selects Ethernet traffic for export to Wireshark.
API_LINK_PROTOCOL_FC	Selects FC traffic for export to Wireshark.

RunVScript(script_file_path, OnVScriptStatus, OnVScriptProgress)

Run the verification script 'script_file_path' on the opened trace file. The function returns when the script is finished running.

'OnVScriptStatus' and 'OnVScriptProgress' are callbacks for status and progress reporting. You can use None if you don't need status/progress events. Otherwise, you must pass your own function as specified below.

Callback function details:

OnVScriptStatus(status)

This callback function is called to provide status updates on state of the script engine. 'status' is TLNetAPI.VSEStatus enumeration value.

OnVScriptProgress(progress)

This callback function is called to provide progress updates on the progress of an executing script. 'progress' is an integer from 0 to 100 that indicates the percentage of completion of the script.

RemoveAllBookmarks()

Removes all bookmarks from trace.

GetAllBookmarks()

Returns an array of strings. Each string is a bookmark name in the trace.

2.4 APITracePacket

This class represents a packet from opened trace.

Class TLNetAPI.APITracePacket

The APITracePacket class has the following methods:

GetTimeStamp()

Returns a 64-bit integer that is time stamp of the packet in picoseconds resolution.

GetChannel()

Returns an integer that is the analyzer channel on which the packet was captured.

GetChannel_2()

Returns a list contains board index and port. Board index is 1 based integer that specifies index of board and port is same as port label on the board.

GetSpeed()

Returns an EanalyzerSpeed value that is the protocol data rate or speed.

GetType()

Returns an EdataType that is the type of the packet.

GetData()

It returns a memoryview object which contains raw packet data between SOF to EOF. The buffer is managed by packet object, it is valid as long as the packet object is alive and the trace is open.

GetRawData()

It returns a memoryview object which contains raw packet data same what data that is shown in raw data view. The buffer is managed by packet object, it is valid as long as the packet object is alive and the trace is open.

GetBookmark()

Returns a string that is the packet bookmark title.

SetBookmark(new_bookmark)

Set the packet bookmark to the string specified in ‘new_bookmark’. If the packet already has an existing bookmark, it is replaced with the new one.

RemoveBookmark()

Remove bookmark from packet. If there is no bookmark on packet, it does nothing.

GetPEMask ()

Returns an integer that shows protocol error mask. The return value can be combination of protocol errors. See TABLE 2.20 and Table TABLE 2.21.

Example: See 4.3

2.5 APIProject

This class represents an opened project file.

Note: The chain_index, device_index and pair_port_index parameters mentioned in the descriptions below are all 0-based indices.

---

class TLNetAPI.APIProject

The **APIProject** class has the following methods:

Close()

Close the opened project file. Note: This will not release the memory.

Save()

Save the project file.

SaveAs(file_name)

Save the project file to the given 'file_name'.

GetChainCount()

Returns an integer that is the number of included chains in the project.

GetDeviceCount(chain_index)

Returns an integer that is the number of devices in the specified chain of a project.

GetPairPortCount(chain_index, device_index)

Returns an integer that is the number of pair ports supported by the specified device in the specified chain of project.

GetAnalyzerSettings(chain_index)

Returns **APIAnalyzerSettings** object for the specified chain in the opened project.

GetPortConfiguration(chain_index, device_index)

Returns an ElinkConfiguration value that is the port configuration of the specified device in the specified chain.

Assign(serial_number, connection_type, chain_index, device_index, onupdate_device_status, onupdate_device_error)

Connects to a device and assigns it to the specified device in the specified chain in the project. If the device needs any update, it returns error code. You must make sure to Assign an appropriate board to the project before calling StartRecording or StartJammer.

Parameters :

serial_number:	integer, serial number of device to connect to.
connection_type:	EConnectionType, specifies connection type device to connect to.
chain_index:	integer, the index of chain. It is a zero-based index
device_index:	integer, zero-based index of device in chain
onupdate_device_status:	call back function. If you set this callback function, it sends progress of update process. Default value is null.
onupdate_device_error:	call back function. If you set this callback function, it sends error that may happened during device update process. Default value is null.

Return value: EAPIErrorCode

Callback function details:

onupdate_device_status (status)

This callback function is called to provide status of device update.

Parameters :

Status:	string, with below values: “Device Update Started” “Waiting for device reset” “Device Is Resetting...” “Device Is Restarting...” “Device Update Finished.” “Connecting To Device...”

onupdate_device_error(error_code, busengine_id, error_message)

This callback function is called if error occurred during device update process.

Parameters:

error_code:	integer. It specifies error code.
busengine_id:	integer, it specifies bus engine identifier that error occurred during updating it.
error_message:	string. It contains error message correspondent with error_code.

AssignByIP(ip, chain_index, device_index, onupdate_device_status, onupdate_device_error)

Adds an online device with the specified link configuration to a chain in the project.

Parameters:

ip:	string, Ipv4-formatted address (e.g. “192.168.100.10”) of connectable device on network
chain_index:	integer, the index of chain. It is a zero-based index
device_index:	integer, zero-based index of device in chain
onupdate_device_status:	call back function. If you set this callback function, it sends progress of update process. Default value is null.
onupdate_device_error	call back function. If you set this callback function, it sends error that may happened during device update process. Default value is null.

Return value: EAPIErrorCode

Callback function details:

onupdate_device_status (status)

This callback function is called to provide status of device update.

Parameters:

Status:	string, with below values: “Device Update Started” “Waiting for device reset” “Device Is Resetting...” “Device Is Restarting...” “Device Update Finished.” “Connecting To Device...”

onupdate_device_error(error_code, busengine_id, error_message)

This callback function is called if error occurred during device update process.

Parameters:

error_code:	integer. It specifies error code.
busengine_id:	integer, it specifies bus engine identifier that error occurred during updating it.
error_message:	string. It contains error message correspondent with error_code.

StartRecording(chain_index, reserved, trigger_settings_names, OnTraceCreated, OnReportRecordingStatus, OnAnalyzerError, time_out)

It starts capturing process.

Chain_index is the index of chain. It is a zero-based index.

Reserved is not used, set it to 0.

Trigger_setting_names is the trigger/filter setting in current opened project. To use the first/default trigger settings in the project, specify “”.

APIProject_StopRecording will be called after ‘timeout’ is elapsed. If it is zero, it never calls APIProject_StopRecording(). It is supposed that this timer just stops recording process. It cannot stop uploading process.

OnTraceCreated, OnReportRecordingStatus, and OnAnalyzerError are callback functions for event handling. If you pass None for any of them, then no callback will occur for that event. Otherwise, you must pass your own function as specified below.

Callback function details:

OnTraceCreated(chain_index, file_name)

This callback function will be called when the recording session on the specified chain has finished creating a trace file. The file path to the trace file is specified in 'file_name'.

OnReportRecordingStatus(chain_index, status, value)

This callback function will be called when there is a change to the recording status on the specified chain. The 'status' is an ErecordingStatus enumeration value.

OnAnalyzerError(chain_index, api_error_code)

This callback function will be called if an error occurs during the recording session. The 'api_error_code' is an EAPIErrorCode enumeration value.

StartRecordingWait(chain_index, reserved, trigger_settings_names, OnTraceCreated, OnReportRecordingStatus, OnAnalyzerError, time_out)

It starts capturing process and will wait until trace has been created before returning.

Parameters:

Chain_index is the index of chain. It is zero base indexes. Reserved is not used, set it to 0.

Trigger_setting_names is the trigger/filter setting in current opened project. To use the first/default trigger settings in the project, specify "".

APIProject_StopRecording will be called after 'timeout' seconds have elapsed. If it is zero, it never calls APIProject_StopRecording(). It is supposed that this timer just stops recording process. It cannot stop uploading process.

OnTraceCreated, OnReportRecordingStatus, and OnAnalyzerError are callback functions for event handling. If you pass None for any of them, then no callback will occur for that event. Otherwise, you must pass your own function as specified below.

Callback function details:

OnTraceCreated(chain_index, file_name)

This callback function will be called when the recording session on the specified chain has finished creating a trace file. The file path to the trace file is specified in 'file_name'.

OnReportRecordingStatus(chain_index, status, value)

This callback function will be called when there is a change to the recording status on the specified chain. The 'status' is an ErecordingStatus enumeration value.

OnAnalyzerError(chain_index, api_error_code)

This callback function will be called if an error occurs during the recording session. The 'api_error_code' is an EAPIErrorCode enumeration value.

StopRecording(chain_index, do_not_upload)

Stops recording or uploading process on the specified chain.

Set 'do_not_upload' to true if you want to skip uploading. Otherwise, set it to false.

WaitForTraces(chain_index)

This function blocks until the running recording operation finishes.

Chain_index parameter should contain which chain index analyzer is running on and the user wants to wait for it.

In an environment without event-loop, like console applications for example, you must call WaitForTraces after start recording, otherwise the callbacks won't get invoked.

After you call WaitForTraces, then client will received OnTraceCreated notification.

StartJammer(chain_index, device_index, pair_port_index, scenario_name, OnReportMonitoringStatus, OnReportJammerStatus, OnJammerError)

Starts the jammer process using the specified chain, device, port pair, and scenario.

Parameters:

chain_index:	integer, zero-based index of chain in project
device_index:	integer, zero-based index of device in chain
pair_port_index:	integer, zero-based index of a pair port. (i.e. P1/P2 pair pot index is 0)
scenario_name:	string, specifies scenario name to be run on give pair port.

OnReportMonitoringStatus, OnReportJammerStatus, and OnJammerError are callback functions for event handling. If you pass None for any of them, then no callback will occur for that event. Otherwise, you must pass your own function as specified below.

Callback function details:

OnReportMonitoringStatus(chain_index, device_index, pair_port_index, action, count)

This callback function will be called when there are updates to monitoring counters that have been configured in the scenario that is running on the specified chain, device, and port pair. The 'action' is a EICActionType enumeration value specifying which action in the scenario has caused the monitoring count to update. The 'count' in an integer

containing the current monitoring count value. The function call comes from a worker thread.

OnReportJammerStatus (chain_index, device_index, pair_port_index, status)

This callback function will be called when there is a change to the session status on the specified chain, device, and port pair. The 'status' is an EjammerSessionStatus enumeration value. The function call comes from the main thread.

OnJammerError (chain_index, device_index, pair_port_index, api_error_code)

This callback function will be called when there is an error during the session running on the specified chain, device, and port pair. The 'api_error_code' is an EAPIErrorCode enumeration value.

StopJammer(chain_index, device_index, pair_port_index)

Stops the jammer process on the specified chain, device, and port pair.

Parameters:

chain_index:	integer, zero-based index of chain in project
device_index:	integer, zero-based index of device in chain
pair_port_index:	integer, zero-based index of a pair port. (i.e. P1/P2 pair pot index is 0).

StartExerciser(chain_index, device_index, pair_port_index, script_name, OnExerciserExitCode, OnReportExerciserStatus, OnExerciserError)

Starts the FC exerciser process using the specified chain, device, port pair index, and given script name

Parameters:

chain_index:	integer, zero-based index of chain in project
device_index:	integer, zero-based index of device in chain
pair_port_index:	integer, zero-based index of a pair port. (i.e. P1/P2 pair pot index is 0)
script_name:	string, specifies exerciser script to be run on given pair port.

OnExerciserExitCode, OnReportExerciserStatus, and **OnExerciserError** are callback functions for event handling. If you pass None for any of them, then no callback will occur for that event. Otherwise, you must pass your own functions as specified below.

Callback function details:

OnExerciserExitCode (chain_index, device_index, pair_port_index, exit_code)

This callback function will be called when the execution of the running script ends and returns an exit code. The 'exit_code' is an integer. The function call comes from a worker thread.

OnReportExerciserStatus (chain_index, device_index, pair_port_index, status)

This callback function will be called when there is a change to the session status on the specified chain, device, and port pair. The 'status' is an ExerciserSessionStatus enumeration value. The function call comes from the main thread.

OnExerciserError (chain_index, device_index, pair_port_index, api_error_code)

This callback function will be called when there is an error during the session running on the specified chain, device, and port pair. The 'api_error_code' is an EAPIErrorCode
enumeration value.

StartExerciser(chain_index, device_index, port_index, OnExerciserExitCode, OnReportExerciserStatus, OnExerciserError)

Starts Ethernet exerciser on given chain, device, port index. This function activates exerciser before starting exerciser if it is not activated.

Parameters:

chain_index:	integer, zero-based index of chain in project
device_index:	integer, zero-based index of device in chain
port_index:	integer, zero-based index of a port. (i.e. P1 index is 0 and port 2 index is 1)

OnExerciserExitCode, OnReportExerciserStatus, and **OnExerciserError** are callback functions for event handling. If you pass None for any of them, then no callback will occur for that event. Otherwise, you must pass your own functions as specified below.

Callback function details:

OnExerciserExitCode (chain_index, device_index, pair_port_index, exit_code)

This callback function will be called when the execution of the running script ends and returns an exit code. The 'exit_code' is an integer. The function call comes from a worker thread.

OnReportExerciserStatus (chain_index, device_index, pair_port_index, status)

This callback function will be called when there is a change to the session status on the specified chain, device, and port pair. The 'status' is an ExerciserSessionStatus enumeration value. The function call comes from the main thread.

OnExerciserError (chain_index, device_index, pair_port_index, api_error_code)

This callback function will be called when there is an error during the session running on the specified chain, device, and port pair. The 'api_error_code' is an EAPIErrorCode
enumeration value.

StopExerciser(chain_index, device_index, pair_port_index)

Stops the exerciser process on the specified chain, device, and port pair.

Parameters:

chain_index:	integer, zero-based index of chain in project
device_index:	integer, zero-based index of device in chain
pair_port_index:	integer, zero-based index of a pair port. (i.e. P1/P2 pair pot index is 0)

ExerciserActivate(chain_index, device_index, port_index,enable = True , timeout = 2)

Activates/Deactivates Ethernet exerciser.

Parameters:

chain_index:	integer, zero-based index of chain in project
device_index:	integer, zero-based index of device in chain
port_index:	integer, zero-based index of a port. (i.e.P1 port index is 0)
enable:	Boolean, Set to True to activate and set to False to deactivate the ethernet exerciser.
timeout:	integer, timeout in second for waiting for activation and link up. Default value is 2 seconds.

ExerciserResetLink(chain_index, device_index, port_index , speed , auto_negotiation , training_sequence , fec_type)

Resets Ethernet exerciser link using given speed and Auto negotiation and training sequence settings on given port.

Parameters:

chain_index:	integer, zero-based index of chain in project
device_index:	integer, zero-based index of device in chain
port_index:	integer, zero-based index of a port. (i.e.P1 port index is 0)
speed:	can be ANALYZER_SPEED_10 or ANALYZER_SPEED_25 or ANALYZER_SPEED_50_PAM4
auto_negotiation:	bool, to enable/disable auto negotiation
training_sequence:	bool,to enable/disable training sequence
fec_type:	EFecType, **enum TLNetAPI.EFecType** RS_Fec BR_Fec Fec_Disabled

InjectErrorNow(chain_index, device_index, port_index ,error_type,correctable = False)

Injects given error on given port.

Parameters:

chain_index:	integer, zero-based index of chain in project
device_index:	integer, zero-based index of device in chain
port_index:	integer, zero-based index of a port. (i.e.P1 port index is 0)
error_type:	EInjectedErrorType
correctable:	bool,applicable only if error_type= ET_FecError

enum TLNetAPI.EInjectedErrorType

ET_None
ET_FCSError
ET_FrameLengthError
ET_FecError

WaitForStatus(time_out)

This function blocks until all running jamming or recording or exerciser operations are finished.
time_out is the maximum time that this function waits for running operations to finish. This function will not finish running operations when the timeout is reached. So, you must stop running tasks when timeout is reached. time_out less than or equal to zero is considered no timeout at all.

Note1: In an environment without event loop, like console applications, you must call WaitForStatus after start jamming or start recording, otherwise the callbacks won't get invoked.
Note2: You can replace WaitForTraces with WaitforStatus, if you need all the analyzer, jammer, and exerciser callbacks at the same time.

AddDevice(device_type, link_config, chain)

Adds an offline device with the specified link configuration to a chain in the project.

Parameters :

device_type:	EdeviceType, product hardware type
link_config:	ElinkConfiguration, port configuration type

chain:	integer, zero-based index of chain in project

Return value: EAPIErrorCode

AddDeviceByIP(ip, link_config , chain, onupdate_device_status, onupdate_device_error)

Adds an online device with the specified link configuration to a chain in the project.

Parameters:

ip:	string, Ipv4-formatted address (e.g. “192.168.100.10”) of connectable device on network
link_config:	ElinkConfiguration, port configuration type
chain:	integer, zero-based index of chain in project
onupdate_device_status:	call back function. If you set this callback function, it sends progress of update process. Default value is null.
onupdate_device_error	call back function. If you set this callback function, it sends error that may happened during update process. Default value is null.

Return value: EAPIErrorCode

Callback function details:

onupdate_device_status (status)

This callback function is called to provide status of device update.

Parameters:

Status:	string, with below values: “Device Update Started” “Waiting for device reset” “Device Is Resetting...” “Device Is Restarting...” “Device Update Finished.” “Connecting To Device...”

onupdate_device_error(error_code, busengine_id, error_message)

This callback function is called if error occurred during device update process.

Parameters:

error_code:	integer. It specifies error code.
busengine_id:	integer, it specifies bus engine identifier that error occurred during updating it.
error_message:	string. It contains error message correspondent with error_code.

AddDeviceBySerialNumber(serial_num, link_config , chain, onupdate_device_status, onupdate_device_error)

Adds an online device with the specified link configuration to a chain in the project.

Parameters :

serial_num:	integer, serial_number of a connectable device
link_config:	ElinkConfiguration, port configuration type
chain:	integer, zero-based index of chain in project
onupdate_device_status:	call back function. If you set this callback function, it sends progress of update process. Default value is null.
onupdate_device_error	call back function. If you set this callback function, it sends error that may happened during update process. Default value is null.

Return value: EAPIErrorCode

Callback function details:

onupdate_device_status (status)

This callback function is called to provide status of device update.

Parameters :

Status:	string, with below values: “Device Update Started” “Waiting for device reset” “Device Is Resetting...” “Device Is Restarting...” “Device Update Finished.” “Connecting To Device...”

onupdate_device_error(error_code, busengine_id, error_message)

This callback function is called if error occurred during device update process.

Parameters:

error_code:	integer. It specifies error code.
busengine_id:	integer, it specifies bus engine identifier that error occurred during updating it.
error_message:	string. It contains error message correspondent with error_code.

GetTriggerSettings(chain, device, link_index=0)

Gets trigger settings object of a device in the project. Actually, when You call this function Net API deletes previous TriggerSetting object (if exist) and create a new one and returns it to the client. It means you cannot keep TriggerSettings objects in client.

Parameters :

chain: integer, zero-based index of chain in project

device: integer, zero-based index of device in chain

link_index: integer, zero-based index of link in device

Return value: APITriggerSettings object for the specified link.

GetJammerManager(chain, device)

Gets the jammer manager of a device in project chain

Parameters :

chain: integer, zero-based index of chain in project

device: integer, zero-based index of device in chain

Return value: APIJammerManager object for the specified device.

StartRTS(chain, device, filename)

Starts the RTS process using the specified chain, device and generate RTS log into given csv filename.

Parameters :

chain: integer, zero-based index of chain in project

device: integer, zero-based index of device in chain

filename: file name of output csv file contains full path.

Return value: EAPIErrorCode

StopRTS(chain, device)

Stops the RTS process on the specified chain and device.

Parameters :

chain: integer, zero-based index of chain in project

device: integer, zero-based index of device in chain

Return value: EAPIErrorCode

StartAutoCalibration (chain_index, device_index, link_index, setting, OnReportAutoCalibrationResult, timeout=-2)

Starts the Auto Calibration process using the specified chain, device, link index and its parameter setting. OnReportAutoCalibrationResult is callback function for result and notification. This function works on M648.

chain_index:	integer, zero-based index of chain in project
device_index:	integer, zero-based index of device in chain
link_index:	integer, zero-based index of link.
setting:	is APIAutoCalibrationSetting data structure to control the operation of Auto Calibration. Defined as below:

DwellTimeSecond: default(15s), duration of Auto calibration monitors for link stability for each setting.

TargetValue: default(5×10^{-10}), Auto Calibration will auto stop once Pre FEC BER reach this limit.

CalibrateOnlywithPreset: default(False), Enable Auto Calibration with existing preset only.

LogEnabled : default(True), enable/disable Log.

ForceFullSweep: Force continue testing for more candidates even if candidate with TargetValue is found.

FilterParameter: Specifoes lower and upper parameter as filter in range setting. If no filter is specified, then all possible setting combinations are checked during the auto calibration run. This setting is represented by APIAutoCalibrationFilterParameter.

APIAutoCalibrationFilterParameter:
This class represents filter settings for auto calibration.
It has the following properties:

Name	Type	Description
ParameterIDs	list	A list of parameter setting of EApiAutoCalibrationParameterID enum type that is used for Auto calibration.
ValuesLowerLimit	list	List of lower limit value for each ParameterID
ValuesUpperLimit	list	List of upper limit value for each ParameterID

Note: All of these properties must be set with equal length

enum TLNetAPI. EApiAutoCalibrationParameterID
This enumeration describes parameters that is being tuned during auto calibration.

Value	Description
AC_CTLE_BOOST_DUT_ID	CTLE Gain DUT PHY setting. Values supported by this parameter is from 0 to 72. Used in M648 product
AC_AMP_GAIN_DUT_ID	AMP Gain DUT PHY setting. Values supported by this parameter is from 0 to 7. Used in M648 product
AC_CTLE_BOOST_ANA_ID	CTLE Gain ANA PHY setting. Values supported by this parameter is from 0 to 72. Used in M648 product
AC_AMP_GAIN_ANA_ID	AMP Gain ANA PHY setting. Value supported by this parameter is from 0 to 7. Used in M648 product

timeout: integer in second,

-2 : (default value) : works as non-blocking

0 : auto calibration process continue until a candidate found.

Greater than 0: auto calibration process continue until a candidate found or given timeout elapsed.

-1 : is an invalid value, function returns error.

OnReportAutoCalibrationResult chain_index, device_index, link_index, result, message)

This callback function will be called when there are candidates found that enables link up with target device on the specified chain, device, and link index.

result :

Result is a `EAutoCalibrationResult` enumeration value that indicates the output result of calibration run.

`EAutoCalibrationResult :`

AC_RESULT_CANDIDATE_NOT_FOUND = 0
AC_RESULT_CANDIDATE_FOUND = 1
AC_RESULT_TARGET_FOUND = 2
AC_RESULT_NOTIFY_APPLY = 3
AC_RESULT_NOTIFY_COMPLETE = 4

message:

message provides additional information about the current calibration result. Mainly reports the TI setting label that is currently being tested.

StopAutoCalibration(chain_index, device_index, link_index)

Stops the Auto Calibration process on the specified chain, device, and link index. Best Candidate setting will be applied if any candidate was found, otherwise the previous setting before the Auto Calibration run will be applied.

chain_index: integer, zero-based index of chain in project

device_index: integer, zero-based index of device in chain

link_index: integer, zero-based index of link.

WriteDeviceProbeCalibrationSettings (chain_index, device_index, port_index, lane_index, setting)

Writes probe calibration setting to the specified device, by port and lane index. This function is available only for M1288.

chain, device, and link index :
Specifies the chain index, device index of the device in project chain.
port_index and lane_index :
port index refers to either (DD/QSFP/SFP) port 0 odd port or port 1 even port. Lane index starts from index 0 up to 7

setting :
Setting is of APIDeviceProbeCalibrationSetting data structure to that specifies the setting item to update with its value.

APIDeviceProbeCalibrationSetting
This class represents settings for device probe calibration.
It has the following properties:

Name	Type	Description
ParameterIDs	list	A list of setting of EApiAutoCalibrationParameterID enum type that is used for device probe setting.
Values	list	List of value for each ParameterID

Note: All of these properties must be set with equal length

enum TLNetAPI. EApiAutoCalibrationParameterID
This enumeration describes parameter that is being set for probe calibration.

Value	Description
AC_DC_GAIN_ANA_ID	DC Gain Analyzer setting

	Values supported by this parameter is from 0 to 2. Used in M1288 product
AC_EQ_GAIN_ANA_ID	EQ Gain Analyzer setting Values supported by this parameter is from 0 to 9. Used in M1288 product
AC_DC_GAIN_DUT_ID	DC Gain DUT setting Values supported by this parameter is from 0 to 2. Used in M1288 product
AC_EQ_GAIN_DUT_ID	EQ Gain DUT setting Values supported by this parameter is from 0 to 9. Used in M1288 product

ExportDeviceSettings(chain_index, device_index, probe_calibration_setting_file_name, phy_setting_file_name =None)

It exports the active device settings to the specified files, which includes probe calibration settings.

chain_index: integer, zero-based index of chain in project

device_index: integer, zero-based index of device in chain

probe_calibration_setting_file_name:string, Full file path of the output csv file, where the probe calibration settings must be exported.

phy_setting_file_name: string, Full file path of the output csv file, where the phy settings must be exported. It is applicable only for M648 and default value is None.

Return value: EAPIErrorCode

ImportDeviceSettings(chain_index, device_index, probe_calibration_setting_file_name phy_setting_file_name=None)

It imports the device settings from the specified files, which includes probe calibration settings.

chain_index: integer, zero-based index of chain in project

device_index: integer, zero-based index of device in chain

probe_calibration_setting_file_name: Full file path of the output csv file,

where the probe calibration settings must be imported.

phy_setting_file_name: string, Full file path of the output csv file, where the phy settings must be exported. It is applicable only for M648 and default value is None.

Return value: EAPIErrorCode

SwitchDeviceSettingsPreset(chain_index, device_index, preset_name)

It changes the current Preset to given preset.

chain_index: integer, zero-based index of chain in project

device_index: integer, zero-based index of device in chain

preset_name: string, name of preset.

2.6 APIJammerManager

This class is the interface for creating jammer scenarios. It has the following methods:

AddAction(scenario_id, sequencer_id, state_index, condition_index, action_id, general_setting, parameter)

Parameters :

scenario_id: integer, the id of the scenario into which to add the action

sequencer_id: EjammerSequencer, the id of the sequencer to which to add the action.

State_index: integer, the 0-based state index into which to add the action

condition_index: integer, the 0-based condition index into which to add the action

action_id: EICActionType, the base type of action to add.

General_setting: APIActionGeneralSetting, general settings for the action.

Parameter: Additional detailed settings for the specified action type. The type used for 'parameter' depends on the specified 'action_id', as shown below:

TABLE 2.33: Add Action

Action_id	Parameter Type
IC_ACTION_BEEP	ActionBeepData
IC_ACTION_GE_FRAME_TRUNCATE_JAM	ActionFrameTruncate
IC_ACTION_FC_FRAME_TRUNCATE_JAM	ActionFrameTruncate
IC_ACTION_AUTO_NEGOTIATION	ActionANJam
IC_ACTION_RECONNECT	ActionConnectDisconnect
IC_ACTION_DISCONNECT	ActionConnectDisconnect
IC_ACTION_BRANCH	ActionBranch
IC_ACTION_ST_ANALYZER_MARKER_TRIGGER	ActionMarker
IC_ACTION_GE_INSERT_BYTES_INSIDE_FRAME	ActionInsertByteinFrame
IC_ACTION_FC_INSERT_BYTES_INSIDE_FRAME	ActionInsertByteinFrame
IC_ACTION_GE_CAPTURE_DWORD	APIActionDWordCapture
IC_ACTION_CAPTURE_DATA_DWORD (is applicable for FC)	APIActionDWordCapture
IC_ACTION_GE_CAPTURE_FRAME	APIActionFrameCapture
IC_ACTION_FC_CAPTURE_FRAME	APIActionFrameCapture
IC_ACTION_GE_INSERT_PRECAPTURE_FRAME	APIActionFrameCapture
IC_ACTION_GE40_INSERT_PRECAPTURE_FRAME	APIActionFrameCapture
IC_ACTION_GE25_INSERT_PRECAPTURE_FRAME	APIActionFrameCapture
IC_ACTION_FC_INSERT_PRECAPTURE_FRAME	APIActionFrameCapture
IC_ACTION_FC_FRAME_MODIFY_JAM	ActionFrameJam
IC_ACTION_GE_FRAME_MODIFY_JAM	ActionFrameJam
IC_ACTION_FC_FRAME_REPLACE_JAM	ActionFrameJam
IC_ACTION_GE_FRAME_REPLACE_JAM	ActionFrameJam
IC_ACTION_GE_INSERT_FRAME	ActionFrameJam
IC_ACTION_FC_INSERT_FRAME	ActionFrameJam
IC_ACTION_GE40_INSERT_FRAME	ActionFrameJam

IC_ACTION_GE25_INSERT_FRAME	ActionFrameJam
IC_ACTION_GE25_SUBS_SYMBOL_16G_JAM	APISymbolData
IC_ACTION_GE40_SUBS_SYMBOL_16G_JAM	APISymbolData
IC_ACTION_GE_SUBS_SYMBOL_16G_JAM	APISymbolData
IC_ACTION_FC_SUBS_SYMBOL_16G_JAM	APISymbolData

Return value: (EAPIErrorCode,item_id), where item_id is a unique id for the newly added action.

AddEvent(scenario_id, sequencer_id, state_index, condition_index, event_id, general_setting, parameter)

Parameters :

scenario_id: integer, the id of the scenario into which to add the event

sequencer_id: EjammerSequencer, the id of the sequencer to which to add the event.

State_index: integer, the 0-based state index into which to add the event

condition_index: integer, the 0-based condition index into which to add the event

event_id: PatternType, the base type of event to add.

General_setting: APIEventGeneralSetting, general settings for the event.

Parameter: Additional detailed settings for the specified pattern type. The type used for 'parameter' depends on the specified 'event_id', as shown below:

TABLE 2.34: AddEvent

Event_id	Parameter Type
ID_PATTERN_TYPE_GE_SYMBOL_66_BITS	APISymbolData
ID_PATTERN_TYPE_FC_SYMBOL_66_BITS	APISymbolData
ID_PATTERN_TYPE_GE10_PROTOCOL_ERRORS	APIProtocolError
ID_PATTERN_TYPE_GE25_PROTOCOL_ERRORS	APIProtocolError
ID_PATTERN_TYPE_GE40_PROTOCOL_ERRORS	APIProtocolError
ID_PATTERN_TYPE_GE50_PROTOCOL_ERRORS	APIProtocolError
ID_PATTERN_TYPE_GE100_PROTOCOL_ERRORS	APIProtocolError
ID_PATTERN_TYPE_GE10_PROTOCOL_ERRORS	APIProtocolError
ID_PATTERN_TYPE_FC_PROTOCOL_ERRORS	APIProtocolError
ID_PATTERN_TYPE_FC_ORDERSET	APIOrderedSet
ID_PATTERN_TYPE_GE_TIMER	APITimerData
ID_PATTERN_TYPE_FC_TIMER	APITimerData
ID_PATTERN_TYPE_OTHER_TRIGS	APIOtherTrigger
ID_PATTERN_TYPE_GE10_LINK_SPEED	None
ID_PATTERN_TYPE_GE40_LINK_SPEED	None
ID_PATTERN_TYPE_GE25_LINK_SPEED	None
ID_PATTERN_TYPE_FC8_LINK_SPEED	None
ID_PATTERN_TYPE_FC16_LINK_SPEED	None
ID_PATTERN_TYPE_BOTH_LINKS_UP	None
ID_PATTERN_TYPE_GE_OTHER_FPGA_TRIGGER	None
ID_PATTERN_TYPE_FC_OTHER_FPGA_TRIGGER	None
ID_PATTERN_TYPE_GE_AUTO_NEGOTIATION_IEEE_802	APIFrameData
ID_PATTERN_TYPE_GE_AUTO_NEGOTIATION_OUI_FORMATTED	APIFrameData
ID_PATTERN_TYPE_GE_AUTO_NEGOTIATION_OUI_UNFORMATTED	APIFrameData
ID_PATTERN_TYPE_GE_AUTO_NEGOTIATION_MESSAGE_CODE_1_NULL	APIFrameData

ID_PATTERN_TYPE_GE_AUTO_NEGOTIATION_MESSAGE_CODE_5_OUI_TAG_CODE	APIFrameData
ID_PATTERN_TYPE_GE_AUTO_NEGOTIATION_MESSAGE_CODE_6_PHY_ID_TAG	APIFrameData
ID_PATTERN_TYPE_GE_AUTO_NEGOTIATION_MESSAGE_CODE_10_EEE_TECH	APIFrameData
ID_PATTERN_TYPE_GE_AUTO_NEGOTIATION_MESSAGE_CODE_ANY	APIFrameData
All Other Types	APIFrameData

Return value: (EAPIErrorCode,item_id), where item_id is a unique id for the newly added event.

AddState(scenario_id, sequencer_id)

Parameters :

scenario_id: integer, the id of the scenario into which to add the state.

sequencer_id: EjammerSequencer, the id of the sequencer to which to add the state. JAMMER_GLOBAL_SEQUENCER is not allowed.

Return value: (EAPIErrorCode,item_id), where item_id is a unique id for the newly added state.

CreateScenario(port_index)

Parameters :

port_index: integer, the 0-based port index on which to create the scenario. Valid values are 0-7.

Return value: (EAPIErrorCode,scenario_name,item_id), where scenario_name is a string and item_id is a unique id for the newly created scenario.

RenameScenario (scenario_id, new_name)

Parameters :

scenario_id: integer, scenario id of the scenario, whose name is intended to rename.

new_name: string, The new name of the scenario.

Return value: EAPIErrorCode

SetJamDirection(scenario_id, direction)

This is only applicable to M164, M168, M408 and M648. It is not applicable to M328 nor M328Q.

Set the traffic direction in which the jammer scenario will perform traffic modification actions. Actions that do not modify traffic are not affected by this.

Parameters :

scenario_id:	integer,	The id of the scenario to change.
direction:	boolean,	If True, the scenario will perform traffic modification actions on traffic coming in to even-numbered ports. Otherwise, the traffic modification actions will be

performed on traffic coming in to odd-numbered ports.

Return value: EAPIErrorCode

GetLibraryTree ()

It returns the Jammer library in the form of a dictionary.

Return value:

It returns a dictionary; consist of all nodes of the library tree.

The key represents the name of the node, and the value can be either a

- dict : Representing the key as a group, and dict represent its children.
- None : Representing the key as a Scenario.
- Empty list: Representing the key as a group with no children.

Example : Below is a simple example to iterate through all the scenario, and run one of your choice

```
def RunSpecificScenerio(self, lib_tree):

    for key, value in lib_tree.items():

        if isinstance(value, dict):

            self.RunSpecificScenerio (value)

        else:

            if value is None:

                if key == " MyScenerioPort1": # This is a scenario
                    Project.StartJammer(0, 0, 0, key, None, None, None)

                    #..More logic here
```

```
def RunMySenerios(self):

    lib_tree = self.jammer_manager.GetLibraryTree()

    self.RunSpecificScenerio(lib_tree)
```

ImportScenarioLibrary (file_name)

Imports compatible scenario items from existing jammer scenario libraries (*.gelib).

Parameters :

file_name: string, full string path including the file name of the jammer library(*.gelib).

Return value:

EAPIErrorCode

2.7 APIAnalyzerSettings

This class represents the AnalyzerSettings from an opened project.

Class TLNetAPI.**APIAnalyzerSettings**

The **APIAnalyzerSettings** class has the following methods:

GetSegmentBufferSize(device_index)

Returns an integer that is the size of the segment buffer of the specified device.

SetSegmentBufferSize(device_index, segment_size)

Sets the size of segment buffer of the specified device to 'segment_size'.

SetTraceFileName(file_name)

Sets the file name of the analyzer settings.

GetTraceFileName()

Returns a string that is the trace file name of analyzer settings object.

SetNumberOfSegment(segment_number)

Sets number of segments for the recording to 'segment_number'. If the segment buffer size is not valid according to the device property and new segment number, it will change to the valid max segment buffer size according to the current segment number.

GetNumberOfSegment()

Returns an integer that is the number of recording buffer segments.

SetTrigMode(trig_mode)

Sets trigger mode to ‘trig_mode’, which must be an EtriggerMode value.

GetTrigMode()

Returns an EtriggerMode value that is the trigger mode.

SetTrigFilterSetting(device_index, trig_setting_name)

Sets the trigger filter setting of the specified device to the specified ‘trig_setting_name’.

SetTrigPosition(device_index, trig_position)

Sets the trigger position of the specified device to ‘trig_position’, which is the percentage post-trigger.

GetTrigPosition(device_index)

Returns an integer that is the trigger position, as percentage post-trigger, of the specified device.

SetSpeed(device_index, link_index, speed)

It sets the Analyser speed of the specified link index. The speed is of the type EAnalyzerSpeed**.**

GetSpeed(device_index, link_index)

Returns the Analyser speed of the devices on the specified link index.

SetTrainingSignalPackedMode(device, link_index, mode)

It sets the Training signal’s packing mode of the specified link index. The mode is of the type ETrainingSignalPackMode.

GetTrainingSignalPackedMode(device, link_index)

Returns the Training signal’s packing mode of the specified link index**.**

SetTrainingSignalLaneNoMask(device, link_index, lane_mask)

It sets the Lane number(s) of the Training signal of the specified link index, to be used for capturing. It is valid only when the pack-mode is Unpacked. Note that the lane_mask is bit masking, where now only single lane is allowed to be captured.

GetTrainingSignalLaneNoMask(device, link_index)

Returns the Lane number(s) of the Training signal of the specified link index to capture, as a bit mask. It is valid when the pack mode is Unpacked.

SetScrambling(device, link_index, is_enable)

It sets the scrambling mode of the specified link index.

IsScrambling(device, link_index)

Returns if scrambling is done on the specified link index.

SetProtocolErrorDetection(device, link_index, protocol_error)

It sets the protocol detection of the specified link index. The protocol_error is of the type APIProtocolError

2.8 APIDevice

This class represents a connected online device.

Update(onupdate_device_status, onupdate_device_error)

Automatically update the Firmware and Bus Engines to the latest required versions.

Parameters :

onupdate_device_status: call back function. If you set this callback function, it sends progress of update process. Default value is null.

onupdate_device_error call back function. If you set this callback function, it sends error that may happened during update process. Default value is null.

Return value: EAPIErrorCode

Callback function details:

onupdate_device_status (status)

This callback function is called to provide status of device update.

Parameters :

Status: string, with below values:
"Device Update Started"
"Waiting for device reset"
"Device Is Resetting..."
"Device Is Restarting..."
"Device Update Finished."
"Connecting To Device..."

onupdate_device_error(error_code, busengine_id, error_message)

This callback function is called if error occurred during device update process.

Parameters:

error_code: integer. It specifies error code.

busengine_id: integer, it specifies bus engine identifier that error occurred during updating it.

error_message:	string. It contains error message correspondent with error_code.

UpdateLicense(license_file)

Updates the license on a device.

Parameters :

license_file:	string, path to license file to update the device with.

Return value: EAPIErrorCode.

SetIP(ip_mode, ip, subnet_mask, default_gateway)

Sets the IP configuration for a device.

Parameters :

ip_mode:	EIPMode, IP mode (static/dhcp) to use for the device.
Ip:	string, if setting to static, the IP address to set for the device.
Subnet_mask:	string, if setting to static, the subnet mask to set for the device.
Default_gateway:	string, if setting to static, the default gateway to set for the device.

Return value: EAPIErrorCode.

GetIP()

Gets the IP configuration of a device.

Return value: (ip_mode, ip, subnet_mask, default_gateway)

ip_mode:	EIPMode, IP mode (static/dhcp) of the device.
Ip:	string, the IP address of the device.
Subnet_mask:	string, the subnet mask of the device.
Default_gateway:	string, the default gateway of the device.

GetSerialNumber()

Gets the serial number of a device.

Return value: integer, the serial number of the device.

GetAliasName()

Gets the alias name of a device.

Return value: string, the alias name of the device.

SetAliasName (alias_name)

Sets the alias name of a device.

Parameters :

alias_name: string, the alias name to set for the device.

Return value: EAPIErrorCode.

GetType()

Gets the type of a device.

Return value: EdeviceType, the type the device.

2.9 APIDeviceManager

This class represents the Device Manager that provides interfaces for connecting to devices and for certain administrative functions.

ConnectByIP(ip, onupdate_device_status, onupdate_device_error)

Connects to an online device at specified IP address.

Parameters :

ip:	string, IP address of online device to connect to.
onupdate_device_status:	call back function. If you set this callback function, it sends progress of update process. Default value is null.
onupdate_device_error	call back function. If you set this callback function, it sends error that may happened during update process. Default value is null.

Return value: APIDevice object for connected device.

Callback function details:

onupdate_device_status (status)

This callback function is called to provide status of device update.

Parameters :

Status: string, with below values:

"Device Update Started"

"Waiting for device reset"

"Device Is Resetting..."

"Device Is Restarting..."

"Device Update Finished."

"Connecting To Device..."

onupdate_device_error(error_code, busengine_id, error_message)

This callback function is called if error occurred during device update process.

Parameters:

error_code: integer. It specifies error code.

busengine_id:	integer, it specifies bus engine identifier that error occurred during updating it.
error_message:	string. It contains error message correspondent with error_code.

ConnectBySerialNumber(serial_number, onupdate_device_status, onupdate_device_error)

Connects to an online device with specified serial number.

Parameters :

serial_number:	integer, serial number of online device to connect to.
onupdate_device_status:	call back function. If you set this callback function, it sends progress of update process. Default value is null.
onupdate_device_error	call back function. If you set this callback function, it sends error that may happened during update process. Default value is null.

Return value: APIDevice object for connected device.

Callback function details:

onupdate_device_status (status)

This callback function is called to provide status of device update.

Parameters :

Status:	string, with below values: “Device Update Started” “Waiting for device reset” “Device Is Resetting...” “Device Is Restarting...” “Device Update Finished.” “Connecting To Device...”

onupdate_device_error(error_code, busengine_id, error_message)

This callback function is called if error occurred during device update process.

Parameters:

error_code:	integer. It specifies error code.
busengine_id:	integer, it specifies bus engine identifier that error occurred during updating it.
error_message:	string. It contains error message correspondent with error_code.

DisconnectByIP(ip)

Disconnects from the connected device at specified IP address.

Parameters :

ip: string, IP address of online device to disconnect from.

Return value: EAPIErrorCode.

DisconnectBySerialNumber(serial_number)

Disconnects from the connected device with specified serial number.

Parameters :

serial_number: integer, serial number of online device to disconnect from.

Return value: EAPIErrorCode.

Discovery_SelectAdapter(ip)

Select the network adapter to use to for automatic device discovery. The effects of this function are saved to a configuration file on disk.

Parameters :

ip: string, IP address of the adapter to use.

Return value: EAPIErrorCode.

Discovery_AddSubnet(name, ip, mask)

Add a subnet to include in automatic device discovery. The effects of this function are saved to a configuration file on disk. This function only needs to be called once on a machine. You may implement a separate script for this purpose and run it on your machine one time.

Parameters:

name: string, name/description to use for the subnet.

Ip: string, IP address of the subnet.

Mask: string, subnet mask for the subnet

Return value: EAPIErrorCode.

Discovery_RemoveSubnet(ip)

Remove a subnet from automatic device discovery. The effects of this function are saved to a configuration file on disk.

Parameters:

ip: string, IP address of the subnet.

Return value: EAPIErrorCode.

Admin_ForceDisconnectByIP(ip)

If an online device is locked/in-use by someone else, this method will unlock it and make it available.

Parameters :

ip: string, IP address of the device.

Return value: EAPIErrorCode.

Admin_ForceDisconnectBySerialNumber(serial_number)

If an online device is locked/in-use by someone else, this method will unlock it and make it available.

Parameters :

serial_number: integer, serial number of the device.

Return value: EAPIErrorCode.

Admin_ResetDeviceByIP(ip)

Reboot/Reset a device.

Parameters :

ip: string, IP address of the device.

Return value: EAPIErrorCode.

Admin_ResetDeviceBySerialNumber(serial_number)

Reboot/Reset a device.

Parameters :

serial_number: integer, serial number of the device.

Return value: EAPIErrorCode.

GetDeviceStatusByIP(ip)

Get the current connection status of a device.

Parameters :

ip: string, IP address of the device.

Return value: string, an informative description of the current connection status of the device

GetDeviceStatusBySerialNumber(serial_number)

Get the current connection status of a device.

Parameters :

serial_number: integer, serial number of the device.

Return value: string, an informative description of the current connection status of the device

GetDeviceIndexByIP(ip)

Returns device index of given device.

Parameters :

ip: string, IP address of the device.

Return value: integer, it is zero if device is not in cascading chain, otherwise it is index of device in the cascading chain as zero based.

GetDeviceIndexBySerialNumber(serial_number)

Returns device index of given device.

Parameters :

serial_number: integer, serial number of the device.

Return value: integer, it is zero if device is not in cascading chain, otherwise it is index of device in the cascading chain as zero based.

AddStaticDevice(ip, devicetype,ForceConnect)

Adds a device with given ip to the device list.

Parameters :

ip:	string, IP address of the device.
Devicetype:	Can be one of below values: EDeviceType.DEVICE_TYPE_M648 EDeviceType.DEVICE_TYPE_M328Q EDeviceType.DEVICE_TYPE_M328 EDeviceType.DEVICE_TYPE_T328 EDeviceType.DEVICE_TYPE_M408 EDeviceType.DEVICE_TYPE_M168
ForceConnect:	bool. It can be True or False. True means ForceConnect is enabled

2.10 APITriggerSettings

This class represents the analyzer triggering and filtering settings for a device in an opened project. Use it to add new triggering and filtering conditions to the project.

AddTrigger(general_setting, pattern_type, parameter)

Adds the specified trigger condition to the project.

Parameters:

general_setting: 2.13, general settings for the trigger condition

pattern_type: PatternType, the base pattern type to be used for the condition

parameter: Additional detailed settings for the specified pattern type. The type used for 'parameter' depends on the specified 'pattern_type', as shown below:

Pattern Type	Parameter Type
ID_PATTERN_TYPE_GE_SYMBOL_66_BITS:	APISymbolData
ID_PATTERN_TYPE_FC_SYMBOL_66_BITS:	APISymbolData
ID_PATTERN_TYPE_GE10_PROTOCOL_ERRORS:	APIProtocolError
ID_PATTERN_TYPE_GE25_PROTOCOL_ERRORS:	APIProtocolError
ID_PATTERN_TYPE_GE40_PROTOCOL_ERRORS:	APIProtocolError
ID_PATTERN_TYPE_GE50_PROTOCOL_ERRORS:	APIProtocolError
ID_PATTERN_TYPE_GE100_PROTOCOL_ERRORS:	APIProtocolError
ID_PATTERN_TYPE_GE10_PROTOCOL_ERRORS:	APIProtocolError
ID_PATTERN_TYPE_FC_PROTOCOL_ERRORS:	APIProtocolError
ID_PATTERN_TYPE_FC_ORDERSET:	APIOrderedSet
ID_PATTERN_TYPE_IPG:	APIIPG
ID_PATTERN_TYPE_GE_TIMER:	APITimerData
ID_PATTERN_TYPE_FC_TIMER:	APITimerData
ID_PATTERN_TYPE_EXTERNAL_TRIGS	None
All Other Types:	APIFrameData

Return value: (EAPIErrorCode, item_id), where item_id is the integer id of the added trigger item.

AddFilter(general_setting , pattern_type , parameter)

Adds the specified filter condition to the project.

Parameters :

general_setting: APITrigGeneralSetting, general settings for the trigger condition

pattern_type: PatternType, the base pattern type to be used for the condition

parameter: Additional detailed settings for the specified pattern type

The type used for 'parameter' depends on the specified 'pattern_type', as shown below:

Pattern Type	Parameter Type
ID_PATTERN_TYPE_GE_SYMBOL_66_BITS:	APISymbolData
ID_PATTERN_TYPE_FC_SYMBOL_66_BITS:	APISymbolData
ID_PATTERN_TYPE_GE10_PROTOCOL_ERRORS:	APIProtocolError
ID_PATTERN_TYPE_GE25_PROTOCOL_ERRORS:	APIProtocolError
ID_PATTERN_TYPE_GE40_PROTOCOL_ERRORS:	APIProtocolError
ID_PATTERN_TYPE_GE50_PROTOCOL_ERRORS:	APIProtocolError
ID_PATTERN_TYPE_GE100_PROTOCOL_ERRORS:	APIProtocolError
ID_PATTERN_TYPE_GE10_PROTOCOL_ERRORS:	APIProtocolError
ID_PATTERN_TYPE_FC_PROTOCOL_ERRORS:	APIProtocolError
ID_PATTERN_TYPE_FC_ORDERSET:	APIOrderedSet
All Other Types:	APIFrameData

Return value: (EAPIErrorCode, item_id), where item_id is the integer id of the added filter item.

SetCaptureStrategy(strategy)

Sets the overall capturing strategy: Everything except selected patterns (by AddFilter), Nothing, or selected patterns only (by AddFilter).

Parameters :

strategy: EcaptureStrategy

Return value: EAPIErrorCode

SetFilterOutOptions(filter_out_options)

Sets some basic/common conditions to filter out from a recording.

Parameters :

filter_out_options: APIFilterOutOption

Return value: EAPIErrorCode

2.11 APIEventGeneralSetting

This class represents general settings for a jammer event. It has the following properties:

Name	Type	Description
CounterValue	integer	The counter value for the event. See CountRandomly below for details. Default is 1.
SOFType	EApiSOFOrdersets	For FC frames, the type of SOF to match on. Default is FC_ORDERSET_SOF_ANY.
CountRandomly	boolean	If False, then CounterValue is the count of event occurrences that must be seen before event detection is asserted. Otherwise, if True, then the random count will be between 1 and CounterValue. Default is False.
Direction	boolean	False: Coming in an odd-numbered port; True: Coming in an even-numbered port. Default is False.

2.12 APIActionGeneralSetting

This class represents general settings for a jammer action. It has the following properties:

Name	Type	Description
EnableVariableHeaderLength	boolean	If True, then variable header length detection will be used. Default is True.
Random	boolean	If False, then NthOccurrence is the count of event occurrences that must be seen before the action is executed. Otherwise, if True, then the random count will be between 1 and Nth Occurrence. Default is False.
NthOccurrence	integer	The counter value for the action. See Random for details. Default is 1.
NthOccuranceToStopScenario	integer	Stops the Jammer action after the number of actions specified in this counter (from 1 to 500,000,000). 0 means disabled.
MonitorCount	boolean	If True, the action will be counted/monitored; if a jammer callback function has been registered, it will be invoked for this action. Otherwise, if False, then the action will not be counted/monitored, and no callbacks will be invoked for it. Default is False.

2.13 APITrigGeneralSetting

This class represents general settings for an analyzer triggering/filtering condition. It has the following properties:

Name	Type	Description
PortSelection	integer	This is a bit mask to qualify the ports on which the trigger condition should be detected. Bit 0 is the least-significant bit and corresponds to Port 1. Bit 7 is the most-significant bit and corresponds to Port 8. To include a port in the trigger condition, set its bit to 1; otherwise set its bit to 0. Default is 0xFF (all ports). Note: you may not deselect a port that is not active in the project's link configuration. For example, if your link configuration only uses P1/P2, you may not deselect P3/P4; leave the bits for any nonconfigured ports set to 1.
Occurrence	integer	The counter value for the condition. It is the count of event occurrences that must be seen before event detection is asserted. Default is 1.
EnableVariableHeaderLength	boolean	For frame events, set to True to enable variable header detection; otherwise, set to False. This is ignored for non-frame events. Default is True.
LaneSelection	integer	This is a bit mask to qualify the lane on which the trigger condition should be detected in training sequence pattern. Note that this will be ignored if set for other patterns.

2.14 APITimerData

This class represents timer settings for a timer condition. It has the following properties:

Name	Type	Description
Value	integer	The timer value. Default is 1.
Unit	EtimerUnits	The units for the timer value. Default is EtimerUnits. TIMER_MILISEC.

2.15 APIOtherTrigger

This class represents settings for other trigger types. It has the following property:

Name	Type	Description
Value	EotherTrigger	The type of other trigger to detect. Default is EotherTrigger.EXTERNAL.

2.16 APIIPG

This class represents settings for the IPG trigger. It has the following properties:

Name	Type	Description
OptionID	IPGControl	The IPG comparison operator for the trigger. Default is IPGControl.ID_IPG_GREATER_THAN.
Length	integer	The IPG value to compare against.

2.17 APISymbolData

This class represents settings for the Symbol Data pattern type. It has the following properties:

Name	Type	Description
SyncHeader	list	A 2-item list representing bits of the Sync Header. Valid values for each item are: 'X' (don't care), '0', and '1'. Default is all 'X'.
Symbol	list	A 16-item list representing the bits of the Symbol. Each item represents 4 bits and is encoded as a single hexadecimal digit. Valid values for each item are 'X' (don't care), '0', '1', '2', …, 'E', 'F'. Default is all 'X'.

2.18 APIProtocolError

This class represents settings for the Protocol Error pattern type. It has the following properties:

Name	Type	Description
ProtocolErrorIDs	list	A list of EGEProtocolError or EFCProtocolError values. Default is empty list. Note: when used for analyzer triggering/filtering, the list may contain multiple items; however, when used for jammer, the list may only contain one item.

2.19 APIOrderedSet

This class represents settings for the Ordered Set pattern type. It has the following properties:

Name	Type	Description
Value	list	A 8-item list representing the bits of the Ordered Set. Each item represents 4 bits and is encoded as a single hexadecimal digit. Valid values for each item are 'X' (don't care), '0', '1', '2', …, 'E', 'F'. Default is all 'X'.

GroupId	EApiOrderedSetGroup	This will indicate the order set group id for the FC order sets. To indicate a specific order set, GroupdId can be left as default (i.e., the value of Undefined).
IsNot	Boolean	This will indicate any order set other than the specified one.

2.20 APIFrameData

This class represents settings for Ethernet/FC Frame pattern types. It has the following properties:

Name	Type	Description
DataIndexes	list	A list of integer byte indexes at which to set data custom data values/masks. Any non-specified indexes will be treated as 'Don't Care'. Default is empty list.
DataValues	list	A list of data byte values corresponding to the byte indexes in DataIndexes. The item positions must match with those in DataIndexes. Default is empty list.
DataMasks	list	A list of data byte masks corresponding to the byte indexes in DataIndexes. A mask bit of '0' means the corresponding value bit will be used; otherwise, it is ignored. The item positions must match with those in DataIndexes. Default is empty list.
DwordCaptureDataList	list	A list of APIDWORDCaptureData items that user wants to insert into FrameData. A mask bit of '0' means the corresponding value bit will be used; otherwise, it is ignored. The item positions must match with those in DataIndexes. Default is empty list.

Note: You must set all of these properties with equal length.

2.21 APIDWORDCaptureData

Name	Type	Description
CaptureIndex	integer	Index of captured data
StartByte	integer	Byte location to insert this captured Dword data in target FrameData
Mask	List	A 4 item byte list for masking the captured Dword data pointed by CaptureIndex. A mask bit of 'F' means the corresponding value byte will be used; otherwise, it is ignored. Default is enable all captured bytes.

2.22 APIFilterOutOption

This class represents global pre-capture filtering settings for recording. It has the following properties:

Name	Type	Description
PortSelection	integer	This is a bit mask to specify the ports on which to filter out all traffic. Bit 0 is the least-significant bit and corresponds to Port 1. Bit 7 is the most-significant bit and corresponds to Port 8. To filter out a port's traffic entirely, set its bit to 1; otherwise set its bit to 0. Default is 0 (no ports filtered out).
Idle	boolean	Filter out Idle ordered sets. Default is False.
RRDY	boolean	Filter out R_RDY ordered sets. Default is False.
ARBff	boolean	Filter out ARBff ordered sets. Default is False.
NOS	boolean	Filter out NOS ordered sets. Default is False.
VC_RDY	boolean	Filter out FC_RDY ordered sets. Default is False.
BeforePortsAreUp	boolean	Filter out traffic before ports are linked up. Default is False.
AlignmentMarker	boolean	Filter out alignment markers. Default is False.
AutoNegotiation	boolean	Filter out auto-negotiation packets. Default is False.
TrainingPattern	boolean	Filter out link training frames. Default is False.
TruncatePayload	boolean	Enable payload truncation. Default is False.
TruncateLength	integer	If payload truncation is enabled, then truncate frames after the specified length in bytes.

2.23 ActionBeepData

This class represents settings for the Beep action. It has the following properties:

Name	Type	Description
Duration	EbeepValues	Duration of the beep action. Default is EbeepValues.Duration_50ms.

2.24 ActionMarker

Supported on M164, M168, and M408. Not supported on M328/M328Q.

This class represents settings for the marker insertion action. It has the following properties:

Name	Type	Description
MarkerID	integer	MarkerID for the Insert Marker jammer action. Valid values are 0-7. Default is 0.
MarkerLabel	String	This will set the marker label to any text, if intended. The default marker label will be considered if it is left unchanged.

2.25 ActionFrameTruncate

This class represents settings for the frame truncation jammer action. It has the following properties:

Name	Type	Description
TruncateLength	integer	Truncate frame after the specified length. Default is 0.
ReplaceCRC	list	A 8-item list representing the bits of the CRC replacement value. Each item represents 4 bits and is encoded as a single hexadecimal digit. Valid values for each item are 'X' (don't change), '0', '1', '2', …, 'E', 'F'. Default is all 'X'. This property is ignored is AutoCalculateCRC is True.
ReplaceFCS	list	A 8-item list representing the bits of the FCS replacement value. Each item represents 4 bits and is encoded as a single hexadecimal digit. Valid values for each item are 'X' (don't change), '0', '1', '2', …, 'E', 'F'. Default is all 'X'. This property is ignored if AutoCalculateFCS is True.
EOFType	EApiEOFOrderedSets	The EOF type to use on the truncated frame.
AutoCalculateCRC	boolean	Set to True to automatically recalculate the CRC on the truncated FC frame. Set to False to use the ReplaceCRC values. Default is True.
AutoCalculateFCS	boolean	Set to True to automatically recalculate the FCS on the truncated Ethernet frame. Set to False to use the ReplaceFCS values. Default is True.

2.26 ActionBranch

This class represents settings for the branching jammer action. It has the following properties:

Name	Type	Description
DestinationStateIndex	integer	The 0-based index of the destination state. Default is 0.

2.27 ActionInsertByteinFrame

This class represents settings for the insert-bytes-in-frame jammer action. It has the following properties:

Name	Type	Description
DataValues	list	A list representing the bits of data to be inserted into the target frame, starting at the byte offset specified by Offset. Each item represents 4 bits and is encoded as a single hexadecimal digit. Valid values for each item are '0', '1', '2',..., 'E', 'F'. Default is empty list.
Offset	integer	The starting byte offset at which to insert the bytes specified in DataValues. Default is 0.
RecalculateFCS	boolean	Recalculate the FCS value after inserting the data. Default is False.
RecalculateCRC	boolean	Recalculate the CRC value after inserting the data. Default is False.
InsertPosition	EInsertPosition	This indicates the insert position of the bytes

2.28 ActionFrameJam

This class represents settings for the frame modification jammer action. It has the following properties:

Name	Type	Description
ReplaceFrameID	PatternType	The base frame pattern to use as replacement. Default is 0.
ReplaceCRC	list	A 8-item list representing the bits of the CRC replacement value. Each item represents 4 bits and is encoded as a single hexadecimal digit. Valid values for each item are 'X' (don't change), '0', '1', '2', …, 'E', 'F'. Default is all 'X', which means to automatically recalculate it.
ReplaceSOF	list	A 8-item list representing the bits of the SOF replacement value. Each item represents 4 bits and is encoded as a single hexadecimal digit. Valid values for each item are 'X' (don't change),'0', '1', '2', …, 'E', 'F'. Default is all 'X'.
ReplaceEOF	list	A 8-item list representing the bits of the EOF replacement value. Each item represents 4 bits and is encoded as a single hexadecimal digit. Valid values for each item are 'X' (don't change),'0', '1', '2', …, 'E', 'F'. Default is all 'X'.
ReplaceFCS	list	A 8-item list representing the bits of the FCS replacement value. Each item represents 4 bits and is encoded as a single hexadecimal digit. Valid values for each item are 'X' (don't change), '0', '1', '2', …, 'E', 'F'. Default is all 'X', which means to leave it unchanged.
ReplaceInvertAction	list	list of APIReplaceInvertAction items that user wants to apply into Frame Jam. Mask bit 'FF' means the corresponding value bit will be inverted; otherwise, it is ignored. StartByte specifies the positions to jam in frame data. Default is empty list Supported jam action : IC_ACTION_GE_FRAME_MODIFY_JAM IC_ACTION_GE_FRAME_REPLACE_JAM IC_ACTION_FC_FRAME_MODIFY_JAM IC_ACTION_FC_FRAME_REPLACE_JAM
ReplaceWithErrorAction	list	list of APIReplaceWithErrorAction items that user wants to apply into Frame Jam. Error Type is enumerated item of EApiReplaceInjectErrorAction. StartByte specifies the byte positions to inject error in frame data. Default is empty list

ReplaceWithDWORDAction	list	list of APIReplaceWithDWORDAction items that user wants to apply into Frame Jam. DwordType is enumerated item of EApiReplaceDWORDAction. StartByte specifies the start byte to replace dword data in frame. Default is empty list
ReplaceWithOrdersetAction	list	list of APIReplaceWithOrdersetAction items that user wants to apply into Frame Jam. UserData holds the user defined orderset Dword data of SAPIOrderedSet data type. StartByte specifies the start byte to replace dword data in frame. Default is empty list Supported FC jam action: IC_ACTION_FC_FRAME_MODIFY_JAM IC_ACTION_FC_FRAME_REPLACE_JAM
FrameData	APIFrameData	The frame data structure specifying the contents of the replacement/modified frame.
FrameLength	integer	User defined Packet length. Default is 0: means packet length will follow software default.
AutoCalculateFCS	boolean	Set to True to automatically recalculate the FCS on the Ethernet frame. Set to False to use the ReplaceFCS values. Default is True.
AutoCalculateCRC	boolean	Set to True to automatically recalculate the CRC on the FC frame. Set to False to use the ReplaceCRC values. Default is True.
AutoCalculateIPCheckSum	boolean	Set to True to automatically recalculate the IP header checksum, if any. Default is False.
AutoCalculateTCPCheckSum	boolean	Set to True to automatically recalculate the TCP header checksum, if any. Default is False. If this is True, then AutoCalculateUDPCheckSum must be set to False; otherwise, the jammer might malfunction.
AutoCalculateUDPCheckSum	boolean	Set to True to automatically recalculate the UDP header checksum, if any. Default is False. If this is True, then AutoCalculateTCPCheckSum must be set to False; otherwise, the jammer might malfunction.

2.29 APIActionFrameCapture

This class represents settings for the frame capture jammer action. It has the following properties:

Name	Type	Description
CaptureIndex	EframeCaptureRegisterIndex	The register into which to store the captured frame. Default is EframeCaptureRegisterIndex.FrameRegister_1.

2.30 APIActionDWordCapture

This class represents settings for the dword capture jammer action. It has the following properties:

Name	Type	Description
FrameDetect	boolean	If True, then dword at specified dword time of event detection will be captured. Default is False.
Offset	integer	If FrameDetect is True, then this is the dword offset into the frame at which to capture the dword. Otherwise, this is ignored. Default is 0.
CaptureIndex	EdwordCaptureRegisterIndex	The register into which to store the captured dword. Default is EdwordCaptureRegisterIndex.DwordRegister_1.

2.31 ActionANJam

This class represents extra settings for the auto-negotiation jam action. It has the following properties:

TABLE 2.35: Auto-Negotiation Jammer Action Settings

Name	Type	Description
CodeWord	list	A 12-item list representing the bits of the auto-negotiation packet code word. Each item represents 4 bits and is encoded as a single hexadecimal digit. Valid values for each item are 'X' (don't change), '0', '1', '2', …, 'E', 'F'. Default is all 'X'.
ManchesterCode	list	A 12-item list representing a bitmask for injecting 107anchester code violations of the CRC replacement value. Each item represents 4 bits and is encoded as a single hexadecimal digit. A bit set to 1 means that an error will be injected at that bit. A bit set to 0 means that an error will no be injected at that bit. Valid values for each item are '0', '1', '2', …, 'E', 'F'. Default is all '0'.
PageDelimiter	boolean	Inject a page delimiter error. Default is False.
PseudoRandom	boolean	Inject error by pseudo-random bit inversion. Default is False.
Recode	boolean	Inject error by recoding. Default is False.

2.32 ActionConnectDisconnect

This class represents extra settings for the Connect/Disconnect link action. It has the following properties:

TABLE 2.36: Connect/Disconnect Link Action Settings

Name	Type	Description
LinkPort1	boolean	If True, the action (Connect or Disconnect) will be applied on the odd-numbered port of the link. Default is False.
LinkPort2	boolean	If True, the action (Connect or Disconnect) will be applied on the even-numbered port of the link. Default is False.

2.33 APIActionInjectErrorSymbol

This class represents extra settings for the Block and Sync Header Inject Error action. It has the following properties:

Name	Type	Description
EnableFrameDetect	boolean	Set true for Frame based event for jamming,
Offset	Integer	Offset in symbols, available for Sync Header Inject Error
NumberofCorruptedSymbols	Integer	Number of symbols to be corrupted, available for Sync Header Inject Error
CorruptRemainingData	boolean	Corrupt Remaining Data, Available for Sync Header Inject Error
CorruptSOF	boolean	Corrupt SOF, Available for Block Type Inject Error
CorruptEOF	boolean	Corrupt EOF

2.34 APIActionInjectErrorAlignment

This class represents extra settings for the Alignment Marker and Bip Inject Error action. It has the following properties:

Name	Type	Description
Position	Integer	Index value for Error, can accept random number

2.35 APIActionInjectErrorFEC

This class represents extra settings for the RS FEC and Base RS FEC Parity Inject Error action. It has the following properties:

Name	Type	Description
InjectErrorID	Integer	Index value for Error, can accept random number
Description of available InjectErrorID (Base R FEC)		
0 1	1 bit error at Data[0] bit	

2	1 bit error at Data[0] bit and 1 bit error at Data[10] (Error Distance 9 bit)
3	11 consecutive bit errors at Data[10:0]
4	11 consecutive bit errors at Data[27:17]
5	1 bit error at Parity[8] bit and 3 bit error at Parity[18:16] (Error Distance 7 bit) 1 bit error at Parity[31] bit and 1 bit error at Parity[21] (Error Distance 9 bit)
6	1 bit error at Data[0] bit and 1 bit error at Data[11] (Error Distance 10 bit)
7	1 bit error at Data[0] bit and 1 bit error at Data[28] (Error Distance 27 bit)
8	1 bit error at Data[31] and 1 bit error at Parity[0] (Error Distance 2049 bit)
9	1 bit error at Data[27] and 1 bit error at Parity[27] (Error Distance 2080 bit)
10	1 bit error at Data[0] and 1 bit error at Parity[27] (Error Distance 2108 bit)

Description of available InjectErrorID (RS FEC)

0	1 symbol at Data
1	1 symbol at Parity
2	4 symbols at Data and 3 symbols at Parity
3	1 symbol at Data and 6 symbols at Parity
4	7 symbols at Data
5	7 symbols at Parity
6	4 symbols at Data and 4 symbols at Parity
7	8 symbols at Data
8	8 symbols at Data (only one bit in each symbol)
9	8 symbols at Parity
10	1 symbol at Data and 7 symbols at Parity (only one bit in each symbol)
11	1 symbol at Data (only one bit in each symbol) and 7 Parity symbols
12	7 symbols at Data and 1 symbol at Parity
13	All Parity symbols

2.36 ActionConnectDisconnect

This class represents extra settings for the connect disconnect action. It has the following properties:

Name	Type	Description
LinkPort1	boolean	Effect the port1 of the link
LinkPort2	boolean	Effect the port2 of the link

2.37 APILinkSpeed

This class represents extra settings for the Link Speed. It has the following properties:

Name	Type	Description
Speed	ELinkSpeed	Indicates the link speed of the event.

2.38 APIPatternConnectDisconnect

This class represents extra settings for the connect disconnect pattern. It has the following properties:

Name	Type	Description
IsConnect	boolean	Sets if a connection needs to be detected
IsDisconnect	boolean	Sets if a disconnection needs to be detected

2.38 APIOtherFPGATrigger

This class represents extra settings for the other FPGA pattern trigger. It has the following properties:

Name	Type	Description
InternalTriggerIndex	integer	Indicates the trigger index

Chapter 3

Memory Management and Garbage Collection

Python's memory allocation and de-allocation method is automatic. The TLNetAPI implements memory management into Python garbage collection; users should not need to worry about memory management.

3.1 API Initialization and Un-initialization

The TLNetAPI will get initialized when the module is imported into Python. The un-initialization process is registered as module cleanup function. Usually the cleanup function is invoked when the module is out of scope or during the Python interpreter termination.

3.2 API Object Garbage Collection

Memory management for all API objects is integrated into Python garbage collection. The API object such as APITrace and APIPacket will be deleted and memory will be released when the object is out of scope. The cleanup function is registered as class __del__ function. User can also use **del** function to explicitly delete and release an object. To avoid exceptions in API object destructors on exiting a script, make sure to explicitly delete API objects before exiting.

Chapter 4

Examples

4.1 Open a project, record something, wait for trace

```
import sys
sys.path.append("C:\Users\Public\Documents\LeCroy\Net Protocol Suite\API\SDK\Bin")
import TLNetAPI

# OnTraceCreated callback for StartRecording
def OnTraceCreated(chain_index, file_name):

    # ====
    # Do your own callback logic here…
    # ====

project = TLNetAPI.OpenProject("myproject.gep")

# Assign board with serial number 9 and connection USB to the project
project.Assign(9, TLNetAPI.EconnectionType.CONNECTION_TYPE_USB, 0, 0)


# Start chain 0, first/default trigger settings in project,
# OnTraceCreated callback, and no timeout
project.StartRecording(0,0,"", OnTraceCreated, None,None,0)

# ====
# Do your own program logic here…
# ====

# Stop chain 0 and start the trace upload
project.StopRecording(0,     False)


# Wait until trace uploading is done. Note that this method requires
# Net 1.75 or later.
project.WaitForTraces(0)

del project
```

4.2 Open a trace file and retrieve packet count

```
import sys

sys.path.append("C:\Users\Public\Documents\LeCroy\Net Protocol Suite\API\SDK\Bin")
import TLNetAPI


trace = TLNetAPI.OpenFile("mytrace.get")

trace.GetCount()

del trace
```

4.3 Retrieve protocol error from a packet

```
pe_mask = packet.GetPEMask()

crc_mask = 1 << libTLNetAPI_python.EGEProtocolError.ID_PE_GE_FC_CRC_ERROR.value

has_crc_error = pe_mask & crc_mask

if(has_crc_error!=0):
   print('has CRC Error')
```

4.4 Auto negotiation Jamming

```
   # Frame Data
    frame_data_event = APIFrameData()

   # AN JAM Data
  AN_jam = ActionANJam()
  AN_jam.CodeWord = ['1', '0', '1', '0', '1', '0', '1', '0', '1', '0', '1', '0']
  AN_jam.ManchesterCode = ['0', 'F', '0', 'F', '0', 'F', '0', 'F', '0', 'F', '0',
'F']
  AN_jam.PageDelimiter = True
  AN_jam.PseudoRandom = True

 # Adding Event
     (result, item_id) = jammer_manager.AddEvent(scenario_id, seq_0, 0, 0,
                                                 PatternType.ID_PATTERN_TYPE_GE_AUTO
_NEGOTIATION_IEEE_802, event_general_setting,
                                                 frame_data_event)
```

```
    # Adding Action
    (result, action_id_1) = jammer_manager.AddAction(scenario_id, seq_0, 0, 0,
                                                     EICActionType.IC_ACTION_AUTO_N
EGOTIATION, action_general_setting,
                                                     AN_jam)
```

4.5 Jammer : Inserting Captured Dword Data in Frame Event

```
# Frame Data Definition
frame_data1 = APIFrameData()
frame_data1.DataValues = [0xD, 0xE, 0xA, 0xD, 0xC, 0xC, 0xC,0xC]
frame_data1.DataIndexes = [20, 21, 22, 23, 24, 25, 26, 27]
frame_data1.DataMasks = [0, 0, 0, 0, 0, 0, 0, 0]
dwordInsert_data = APIDWORDCaptureData()
dwordInsert_data.CaptureIndex = EdwordCaptureRegisterIndex.DwordRegister_3
dwordInsert_data.StartByte = 0x2C
dwordInsert_data.Mask = ['F','0','F','F']
frame_data1.DwordCaptureDataList.append(dwordInsert_data)
```

4.5 Jammer : DWord Matcher Event / Replace DWord Action

```
# Event Data
   dword_match_event = APIFCDWordMatcher()
   dword_match_event.Mode = EFCDWordMatcherMode.CustomDword_0
   dword_match_event.Value = ['4', 'B', '0', '0', '0', '0', '0', '0']
   dword_match_event.Mask = ['F', 'F', '0', '0', '0', '0', '0', '0']
   print("Add Match Event")
   # Adding the Event
   (result, item_id) = jammer_manager.AddEvent(scenario_id, seq_global, 0,
0,PatternType.ID_PATTERN_TYPE_FC_DWORD_MATCHER,
            event_general_setting, dword_match_event)

   print("Action General Setting")
   action_general_setting = APIActionGeneralSetting()
   action_general_setting.MonitorCount = True

   print("Action Data")
   # Action Data
   dword_replace_action = APIFCDWordMatcher()
```

```
    dword_replace_action.Value = ['B', 'E', 'E', 'F', '0', '0', '0', '0']
    dword_replace_action.Mask = ['F', 'F', 'F', 'F', '0', '0', '0', '0']
    # Adding Action
    (result, action_id_1) = jammer_manager.AddAction(scenario_id, seq_global, 0,
0,EICActionType.IC_ACTION_REPLACE_DWORD_GENERAL,
    action_general_setting, dword_replace_action)
```

4.6 Jammer : Inject error action

```
event_general_setting = APIEventGeneralSetting()
    event_general_setting.CounterValue = 1
    event_general_setting.CountRandomly = False
    #event_general_setting.Direction = True

    print("Symbol")
    Symbol_ev = ['4', 'B', '0', '0', '0', '0', '0', '1', '0', '0', '0', '0', '0', '0', '0', '0']
    SyncHeader_ev = ['1', '0']
    Symbol_ac = ['4', 'B', '0', '0', '0', '0', '0', '1', '0', '0', '0', '0', '0', '0', '1', '0']
    SyncHeader_ac = ['1', '0']
# Event Data
    symbol_data_event = APISymbolData()
    symbol_data_event.Symbol = Symbol_ev
    symbol_data_event.SyncHeader = SyncHeader_ev
    print("Add Symbol Event")
    # Adding the Event
    (result, item_id) = jammer_manager.AddEvent(scenario_id, seq_global, 0,
0,PatternType.ID_PATTERN_TYPE_GE_SYMBOL_66_BITS,
event_general_setting, symbol_data_event)

    print("Action General Setting")
    action_general_setting = APIActionGeneralSetting()
    action_general_setting.MonitorCount = True
    action_general_setting.NthOccuranceToStopScenario = 1     #0 to disable
    print("Action Symbol")
    # Action Data
    symbol_data_action = APISymbolData()
```

```
    symbol_data_action.Symbol = Symbol_ac
    symbol_data_action.SyncHeader = SyncHeader_ac
    # Adding Action
    (result, action_id_1) = jammer_manager.AddAction(scenario_id, seq_global, 0,
0,EICActionType.IC_ACTION_GE25_SUBS_SYMBOL_16G_JAM,
    action_general_setting, symbol_data_action)

    inject_error_symbol = APIActionInjectErrorSymbol()
    print("Inject Error IC_ACTION_GE25_INJECT_ERROR_TYPE_SYNC_HEADER")
    inject_error_symbol.EnableFrameDetect = False
    (result, action_id_1) = jammer_manager.AddAction(scenario_id, seq_global, 0, 0,

EICActionType.IC_ACTION_GE25_INJECT_ERROR_TYPE_SYNC_HEADER,
                                    action_general_setting, inject_error_symbol)
    print("Inject Error IC_ACTION_GE25_INJECT_ERROR_TYPE_BLOCK")
    (result, action_id_1) = jammer_manager.AddAction(scenario_id, seq_global, 0, 0,

EICActionType.IC_ACTION_GE25_INJECT_ERROR_TYPE_BLOCK,
                                    action_general_setting, inject_error_symbol)
    print("Inject Error IC_ACTION_INJECT_ERROR_TYPE_BASERFEC_PARITY")
    inject_error_fec = APIActionInjectErrorSymbol()
    inject_error_fec.InjectErrorID = 1
    (result, action_id_1) = jammer_manager.AddAction(scenario_id, seq_global, 0, 0,

EICActionType.IC_ACTION_INJECT_ERROR_TYPE_BASERFEC_PARITY,
                                    action_general_setting, inject_error_fec)
    print("Inject Error IC_ACTION_INJECT_ERROR_TYPE_RSFEC_PARITY")
    inject_error_fec.InjectErrorID = 8
    (result, action_id_1) = jammer_manager.AddAction(scenario_id, seq_global, 0, 0,

EICActionType.IC_ACTION_INJECT_ERROR_TYPE_RSFEC_PARITY,
                                    action_general_setting, inject_error_fec)
    print("Inject Error IC_ACTION_GE_INJECT_ERROR_TYPE_ALIGNMENT")
    inject_error_alignment = APIActionInjectErrorAlignment()
    inject_error_alignment.Position = 20
    (result, action_id_1) = jammer_manager.AddAction(scenario_id, seq_global, 0, 0,

EICActionType.IC_ACTION_GE_INJECT_ERROR_TYPE_ALIGNMENT,
```

```
                                        action_general_setting, inject_error_alignment)
    print("Inject Error IC_ACTION_GE_INJECT_ERROR_TYPE_ALIGNMENT_BIP")
    inject_error_alignment.Position = 2
(result, action_id_1) = jammer_manager.AddAction(scenario_id, seq_global, 0, 0,

  EICActionType.IC_ACTION_GE_INJECT_ERROR_TYPE_ALIGNMENT_BIP,
                                        action_general_setting, inject_error_alignment)
    frame_data = APIFrameData()
    frame_data.DataIndexes=[116,117,118,119]
    frame_data.DataValues=[0x12,0x34,0x56,0x78]
    frame_data.DataMasks=[0,0,0,0]

    frame_event=APIFrameData()

    event_general_setting.CounterValue = 1
    print("Frame Event")
    (result,item_id) = jammer_manager.AddEvent(scenario_id,
EJammerSequencer.JAMMER_SEQUENCER_0,  0, 0,
PatternType.ID_PATTERN_ROCE_NVME_FCTYPE_CONNECT_COMMAND,
event_general_setting, frame_event)
    (result,item_id) = jammer_manager.AddEvent(scenario_id,
EJammerSequencer.JAMMER_SEQUENCER_0,  0, 1,
PatternType.ID_PATTERN_ROCE_NVME_FCTYPE_CONNECT_COMMAND,
event_general_setting, frame_event)

    action_frame_general_setting=APIActionGeneralSetting()
    ReplacementFrame=ActionFrameJam()
    ReplacementFrame.FrameData=frame_data
    ReplacementFrame.AutoCalculateCRC = True
    #ReplacementFrame.AutoCalculateFCS = True
    #ReplacementFrame.ReplaceCRC =['B', 'C', '5', 'F', '0', '1', '2', '3']
    action_frame_general_setting.Direction=True
    print("Frame Jam Action")
    (result,action_id_1) = jammer_manager.AddAction(scenario_id,
EJammerSequencer.JAMMER_SEQUENCER_0, 0, 0,
EICActionType.IC_ACTION_GE_FRAME_MODIFY_JAM, action_frame_general_setting,
ReplacementFrame)
```

```
   action_frame_general_setting.Direction=True#for P2/P4/P6/P8 default is False for
P1/P3/P5/P7
   (result,action_id_1) = jammer_manager.AddAction(scenario_id,
EJammerSequencer.JAMMER_SEQUENCER_0, 0, 1,
EICActionType.IC_ACTION_GE25_INSERT_FRAME, action_frame_general_setting,
ReplacementFrame)
   action_frame_general_setting.Direction=False
   (result) = jammer_manager.SetJamDirection(scenario_id,False)

   print("Frame Based Error Injection")
   event_general_setting.Direction = True
   (result,item_id) = jammer_manager.AddEvent(scenario_id,
EJammerSequencer.JAMMER_SEQUENCER_1,  0, 0,
PatternType.ID_PATTERN_ROCE_NVME_FCTYPE_CONNECT_COMMAND,
event_general_setting, frame_event)
   event_general_setting.Direction = False
   (result,item_id) = jammer_manager.AddEvent(scenario_id,
EJammerSequencer.JAMMER_SEQUENCER_1,  0, 1,
PatternType.ID_PATTERN_ROCE_NVME_FCTYPE_CONNECT_COMMAND,
event_general_setting, frame_event)
   print("Inject Error IC_ACTION_GE25_INJECT_ERROR_TYPE_SYNC_HEADER")
   action_frame_general_setting.Direction=True
   inject_error_symbol.EnableFrameDetect = True
   inject_error_symbol.Offset = 10
   inject_error_symbol.NumberofCorruptedSymbols = 8
   inject_error_symbol.CorruptRemainingData = True
   inject_error_symbol.CorruptSOF = True
   inject_error_symbol.CorruptEOF = True
   (result, action_id_1) = jammer_manager.AddAction(scenario_id,
EJammerSequencer.JAMMER_SEQUENCER_1, 0, 0,

EICActionType.IC_ACTION_GE25_INJECT_ERROR_TYPE_SYNC_HEADER,
                                    action_frame_general_setting, inject_error_symbol)
   action_frame_general_setting.Direction=False
   inject_error_symbol.EnableFrameDetect = True
   print("Inject Error IC_ACTION_GE25_INJECT_ERROR_TYPE_BLOCK")
   (result, action_id_1) = jammer_manager.AddAction(scenario_id,
EJammerSequencer.JAMMER_SEQUENCER_1, 0, 1,
```

```
EICActionType.IC_ACTION_GE25_INJECT_ERROR_TYPE_BLOCK,
                                    action_frame_general_setting, inject_error_symbol)
```

4.7 Exerciser: Start Exerciser

```
import sys
import os
import time

script_dir = os.path.dirname(os.path.realpath(__file__))
if(sys.platform == 'win32'):
sys.path.append(r'C:/Users/Public/Documents/LeCroy/Net Protocol Suite/API/SDK/Bin')
else:
sys.path.append(r'/usr/local/LeCroy/NetProtocolSuite/API/SDK/Bin')

import libTLNetAPI_python



def on_exerciser_exit_code(chain_index, device_index,pair_port_index,exit_code):
        print("exerciser_exit_code : Chain Index:{} ,
device_index:{},exit_code:{}".format(chain_index, device_index,exit_code))

def report_exerciser_status(chain_index, device_index,pair_port_index,status):
        print("exerciser_status : Chain Index:{} ,
device_index:{},status:{}".format(chain_index, device_index,status))

def report_exerciser_error(chain_index, device_index,pair_port_index,error):
        print("exerciser_error : Chain Index:{} ,
device_index:{},error:{}".format(chain_index, device_index,error))
```

```
api_version = libTLNetAPI_python.GetVersion()
print('API version: ' + str(api_version))
fileName = script_dir + "/Exerciser-script.gep"
project = libTLNetAPI_python.OpenProject(fileName)
print("File Name : " + fileName)

project.Assign(20313, libTLNetAPI_python.EConnectionType.CONNECTION_TYPE_TCP,
0, 0)
project.StartExerciser(0, 0, 2 , "scriptname", on_exerciser_exit_code,
report_exerciser_status, report_exerciser_error)
#time.sleep(2)
project.WaitForStatus(10)
del project

input("press any key to continue")
```

4.8 Auto Calibration: Start Auto calibration and stop when candidate is found.

```
import sys, time
import platform
import os

script_dir = os.path.dirname(os.path.realpath(__file__))
if platform.system() == 'Windows':
   sys.path.append(r'C:/Users/Public/Documents/LeCroy/Net Protocol Suite/API/SDK/Bin')
   import TLNetAPI
else:
   sys.path.append(r'/usr/local/LeCroy/NetProtocolSuite/API/SDK/Bin')
   import libTLNetAPI_python as TLNetAPI

import libTLNetAPI_python
from libTLNetAPI_python import *

def GetAutoCalibrationResultString(result):
   return{
      0 : "No Candidate Found",
      1 : "Candidate Found",
      2 : "Target Found",
      3 : "Applying",
      4 : "Completed",
      }[result]

# AutoCalibration callback
def OnReportAutoCalibrationResult(chain_index,device_index,link_index, result, message):
   print("Info : "+ message + " Result : " + GetAutoCalibrationResultString(result))
   return

deviceManager = TLNetAPI.GetDeviceManager()
serial = 20969

dev_Stat = deviceManager.GetDeviceStatusBySerialNumber(serial)
print("Board serial Number:{} is {}".format(serial,dev_Stat))
```

```
project = TLNetAPI.CreateProject()
project.AddDeviceBySerialNumber(serial,
TLNetAPI.ELinkConfiguration.LINK_CONFIG__QSFP__A_100GPAM4__0, 0)

setting = TLNetAPI.APIAutoCalibrationSetting()
setting.LogEnabled = True
setting.ForceFullSweep = True
#setting.TargetValue = 0.0000000005

ACFilterParameter = TLNetAPI.APIAutoCalibrationFilterParameter()
ACFilterParameter.ParameterIDs =
[TLNetAPI.EApiAutoCalibrationParameterID.AC_CTLE_BOOST_DUT_ID,
TLNetAPI.EApiAutoCalibrationParameterID.AC_AMP_GAIN_DUT_ID,
TLNetAPI.EApiAutoCalibrationParameterID.AC_CTLE_BOOST_ANA_ID,
TLNetAPI.EApiAutoCalibrationParameterID.AC_AMP_GAIN_ANA_ID]
ACFilterParameter.ValuesLowerLimit = [1,1,1,2]
ACFilterParameter.ValuesUpperLimit = [5,4,5,5]
setting.FilterParameter = ACFilterParameter
project.StartAutoCalibration(0, 0, 0,setting, OnReportAutoCalibrationResult)
print("Auto Calibration has started")
time.sleep(600)
print("Force Stopping Auto Calibration")
project.StopAutoCalibration(0, 0, 0)
del project
```

4.9 Jammer: In Frame sub frame jam actions

```
 print("Action Frame Jam")
action_modify_frame_data = ActionFrameJam()
#Invert sub action
invert_action = APIReplaceInvertAction()
invert_action.StartByte = 0x05
invert_action.Mask = ['5','6']
action_modify_frame_data.ReplaceInvertAction.append(invert_action)
invert_action = APIReplaceInvertAction()
```

```
    invert_action.StartByte = 0x10
    invert_action.Mask = ['0','F']
    action_modify_frame_data.ReplaceInvertAction.append(invert_action)

    #Inject Error sub action
    error_action = APIReplaceWithErrorAction()
    error_action.StartByte = 0x15
    error_action.ErrorType =
EApiReplaceInjectErrorAction.REPLACE_WITH_ERROR_CODE_VIOLATION
    action_modify_frame_data.ReplaceWithErrorAction.append(error_action)
    error_action = APIReplaceWithErrorAction()
    error_action.StartByte = 0x25
    error_action.ErrorType =
EApiReplaceInjectErrorAction.REPLACE_WITH_ERROR_DISPARITY
    action_modify_frame_data.ReplaceWithErrorAction.append(error_action)

    #Replace with orderset sub action
    orderset_action = APIReplaceWithOrdersetAction()
    orderset_action.StartByte = 0x30
    orderset_action.UserData = ['B', 'C', '9', '5', '7', '5', '7', '5']
    action_modify_frame_data.ReplaceWithOrdersetAction.append(orderset_action)
    orderset_action = APIReplaceWithOrdersetAction()
    orderset_action.StartByte = 0x40
    orderset_action.UserData = ['X', 'C', '9', '5', '4', 'A', 'X', 'A']
    action_modify_frame_data.ReplaceWithOrdersetAction.append(orderset_action)

    #Replace Dword sub action
    replace_dword_action = APIReplaceWithDWORDAction()
    replace_dword_action.StartByte = 0x50
    replace_dword_action.DwordType = EApiReplaceDWORDAction.REPLACE_WITH_CRC

action_modify_frame_data.ReplaceWithDWORDAction.append(replace_dword_action)
    replace_dword_action = APIReplaceWithDWORDAction()
    replace_dword_action.StartByte = 0x54
    replace_dword_action.DwordType =
EApiReplaceDWORDAction.REPLACE_WITH_IDLE
```

```
action_modify_frame_data.ReplaceWithDWORDAction.append(replace_dword_action)
   replace_dword_action = APIReplaceWithDWORDAction()
   replace_dword_action.StartByte = 0x58
   replace_dword_action.DwordType =
EApiReplaceDWORDAction.REPLACE_WITH_IDLE

action_modify_frame_data.ReplaceWithDWORDAction.append(replace_dword_action)
   (result, action_id_1) = jammer_manager.AddAction(scenario_id, seq_global, 0, 0,
  EICActionType.IC_ACTION_FC_FRAME_MODIFY_JAM, action_general_setting,
action_modify_frame_data)
```

4.10 Write Device Probe Calibration for M1288:

```
deviceManager = TLNetAPI.GetDeviceManager()
serial = 24522   #replace with your board serial number

dev_Stat = deviceManager.GetDeviceStatusBySerialNumber(serial)
print("Board serial Number:{} is {}".format(serial,dev_Stat))

project = TLNetAPI.CreateProject()
project.AddDeviceBySerialNumber(serial,
TLNetAPI.ELinkConfiguration.LINK_CONFIG__QSFPDD__A_400GPAM4__0__0__0, 0)

setting = TLNetAPI.APIDeviceProbeCalibrationSetting()
setting.ParameterIDs =
[TLNetAPI.EApiAutoCalibrationParameterID.AC_DC_GAIN_ANA_ID,TLNetAPI.EApiAutoCali
brationParameterID.AC_EQ_GAIN_ANA_ID,
                TLNetAPI.EApiAutoCalibrationParameterID.AC_DC_GAIN_DUT_ID,
TLNetAPI.EApiAutoCalibrationParameterID.AC_EQ_GAIN_DUT_ID]
setting.Values = [1,2,1,6]

project.WriteDeviceProbeCalibrationSettings(0, 0, 0, 1, setting)#odd port lane 1
project.WriteDeviceProbeCalibrationSettings(0, 0, 1, 1, setting)
project.WriteDeviceProbeCalibrationSettings(0, 0, 0, 6, setting)
project.WriteDeviceProbeCalibrationSettings(0, 0, 1, 6, setting)#even port lane 6

del project
```

Appendix A

How to Contact Teledyne LeCroy

Send e-mail to Support	psgsupport@teledyne.com
Contact support	teledynelecroy.com/support/contact
Visit Teledyne LeCroy's web site	teledynelecroy.com
Tell Teledyne LeCroy	Report a problem to Teledyne LeCroy Support via e-mail by selecting **Help > Tell Teledyne LeCroy** from the application toolbar. This requires that an e-mail client be installed and configured on the host machine.