OptiInstrument

OptiInstrument 1.0 Release Notes

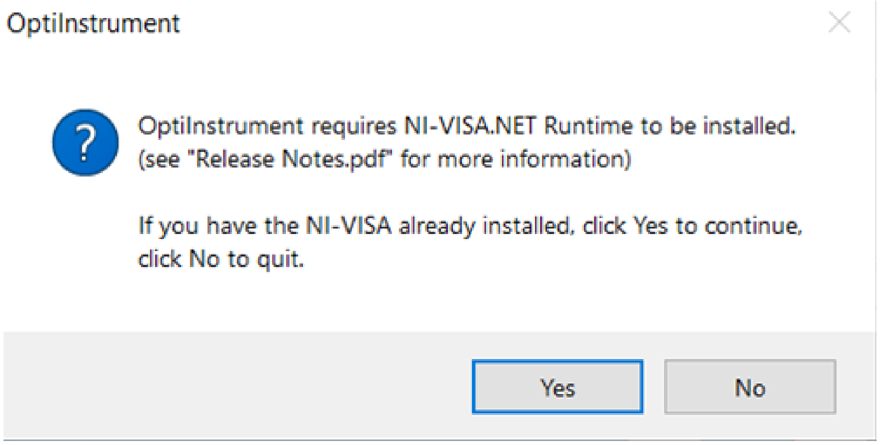
IMPORTANT

- Before installing OptiInstrument, make sure that NI-VISA.NET Runtime is installed on the system. When installing the NI-VISA, select the NI-VISA.NET Runtime.
- Users can install NI-VISA from the following link https://www.ni.com/en-ca/support/downloads/drivers/download.ni-visa.html#346210.

  Note: The NI-VISA is a large package, it may take longer time to download/install.

Installation Notes:

- When starting the installation process of OptiInstrument, the popup message shown in Fig.1 offers the user a choice to quit the installation process if the NI-VISA.NET Runtime package is not installed on the same computer or continue the installation if the package is installed. However, the error message shown in Fig. 2 appears when the user initiate OptiInstrument application and the NI-VISA package is not installed.
- OptiInstrument 1.0 includes the option to install OptiInstrument samples during (or any time after) installation. The installation location for the samples folder can be defined (by default the samples folder will be installed in the user's **Documents**, under "OptiInstrument 1.0 Samples" folder.

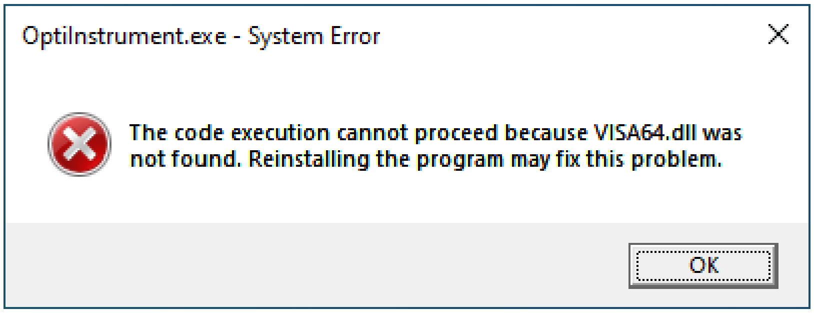


OptiInstrument

OptiInstrument requires NI-VISA.NET Runtime to be installed.
(see "Release Notes.pdf" for more information)

If you have the NI-VISA already installed, click Yes to continue, click No to quit.

Fig. 1 OptiInstrument installation popup message


OptiInstrument.exe - System Error

The code execution cannot proceed because VISA64.dll was not found. Reinstalling the program may fix this problem.

Fig. 2 OptiInstrument installation error message when NI-VISA Runtime is not preinstalled

Minimum Hardware and Software Requirements

- OptiInstrument requires the following minimum/recommended system configuration:
- Minimum PC configuration: PC with Pentium processor (E6, G Series) or equivalent.
- 8GB RAM.
- OptiInstrument requires the following third-party software packages to be installed:
- NI-VISA (NI-VISA.NET Runtime).
- Latest version of Python programming language framework
- Recommended PC configuration: PC with a clock speed > 2 GHz with 2-4 cores (e.g. Intel i3, i5, i7) and 16GB RAM or more.
- Operating Systems: Microsoft Windows 8.1/10 (**64-bit only!**)
- **Microsoft is shelving Windows 7;** we will not support Windows 7 starting this release. However, the software might run under Windows 7, but we do not guarantee it and we will not be able to provide technical support for bugs/crashes.
- 2 GB free hard disk space.
- 1280 x 1024 graphic resolution

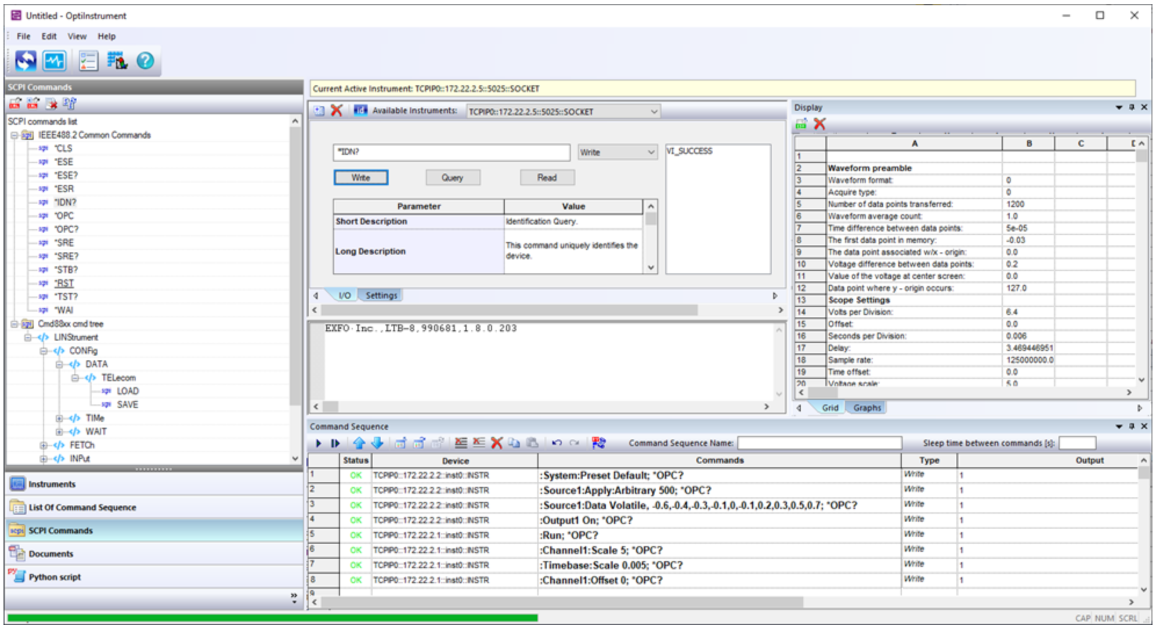
Application Execution

- **Administrators:** when installing OptiInstrument for users with Restricted User Profile, install the sample files in a folder where these users have Read/Write access. By default, the sample files are installed in the current user’s Document folder. OptiInstrument requires the read/write file access and will not work with read-only files.
- For the OptiInstrument Help feature to function properly, Adobe Acrobat Reader must be installed. To get the latest version please visit the Adobe website at http://www.adobe.com/.

- Some computers are configured in power saving mode to go to Hibernation or Sleep mode when they are not in use. It is recommended to disable this feature, especially when running unattended lengthy simulations. Typically, after the simulation is complete, the computer idles and eventually goes to Hibernation. This causes the licensing platform drivers to invalidate the license. When the computer wakes up and resume its execution, OptiInstrument software will issue a message that the license is not available and terminate, losing the simulation results in the process. Please disable the computer hibernation feature to avoid this problem.

OptiInstrument Software Overview

The user-friendly graphical user interface (GUI) of **OptiInstrument 1.0** Software is shown in Fig. 3. It is a standalone tool that can be used to communicate and control different kinds of instruments. OptiInstrument uses the standard commands for programmable instruments (**SCPI**) to communicate **physically** or **remotely** with instruments. The tool uses standard communication interfaces such as **TCP/IP**, **USB**, **GPIB**, or a serial port (**RS232/RS485**). Users can load lists of SCPI commands from **XML files** or write individual commands to control the instrument(s). The commands appear in a **tree configuration**. A single command or a sequence of commands can be executed by OptiInstrument. A **Python script** can be generated for the SCPI commands, saved, loaded and executed by OptiInstrument or in a Python environment. OptiInstrument GUI has a built-in viewer and CSV file analysis window. The GUI supports dockable windows that can be split of the main GUI or placed anywhere in the GUI. OptiInstrument is ideal for automated testing and characterization.



Current Active Instrument: TCPIP0::172.22.2.5::5025::SOCKET

Available Instruments: TCPIP0::172.22.2.5::5025::SOCKET

*IDN? Write

VI_SUCCESS

Parameter	Value
Short Description	Identification Query.
Long Description	This command uniquely identifies the device.

```
EXFO Inc.,LTB-8,990681,1.8.0.203
```

	A	B	C
1			
2	**Waveform preamble**		
3	Waveform format:	0	
4	Acquire type:	0	
5	Number of data points transferred:	1200	
6	Waveform average count:	1.0	
7	Time difference between data points:	5e-05	
8	The first data point in memory:	-0.03	
9	The data point associated w/x - origin:	0.0	
10	Voltage difference between data points:	0.2	
11	Value of the voltage at center screen:	0.0	
12	Data point where y - origin occurs:	127.0	
13	**Scope Settings**		
14	Volts per Division:	6.4	
15	Offset:	0.0	
16	Seconds per Division:	0.006	
17	Delay:	3.469446951	
18	Sample rate:	125000000.0	
19	Time offset:	0.0	
20	Voltage scale:	5.0	

Command Sequence Name:

Sleep time between commands [s]:

	Status	Device	Commands	Type	Output
1	OK	TCPIP0::172.22.2.2::inst0::INSTR	:System:Preset Default; *OPC?	Write	1
2	OK	TCPIP0::172.22.2.2::inst0::INSTR	:Source1:Apply:Arbitrary 500; *OPC?	Write	1
3	OK	TCPIP0::172.22.2.2::inst0::INSTR	:Source1:Data Volatile, -0.6,-0.4,-0.3,-0.1,0,-0.1,0.2,0.3,0.5,0.7; *OPC?	Write	1
4	OK	TCPIP0::172.22.2.2::inst0::INSTR	:Output1 On; *OPC?	Write	1
5	OK	TCPIP0::172.22.2.1::inst0::INSTR	:Run; *OPC?	Write	1
6	OK	TCPIP0::172.22.2.1::inst0::INSTR	:Channel1:Scale 5; *OPC?	Write	1
7	OK	TCPIP0::172.22.2.1::inst0::INSTR	:Timebase:Scale 0.005; *OPC?	Write	1
8	OK	TCPIP0::172.22.2.1::inst0::INSTR	:Channel1:Offset 0; *OPC?	Write	1
9					

Fig. 3 OptiInstrument GUI

OptiInstrument

Key Features of OptiInstrument 1.0

- User friendly GUI
- Execute single or sequence of SCPI commands
- Load XML files and all other file formats into GUI panels
- Drag and drop commands with flexible sequence ordering
- Generate Python script for sequence of commands
- Built-in signal viewer and CSV file analysis page
- Built-in full Python script editor
- Remote operation and control of instruments
- Supports dockable windows

OptiInstrument Software Applications

- Remotely communicate with instruments
- Setup parameters of equipment
- Automate testing and characterization
- View generated signals
- Extract & save the data of generated signals for post processing
- Integrate instruments with photonics and systems simulation tools

Display and Data Acquisition Window

OptiInstrument GUI allows users to display waveforms produced by instruments that support signal generation. Once the sequence of SCPI commands for an instrument is executed, the generated waveform is displayed in the **Display Window** in OptiInstrument GUI as illustrated in Fig. 4. The user is able to acquire the displayed waveform of the instrument as an image and save the waveform data in a .CSV spreadsheet.

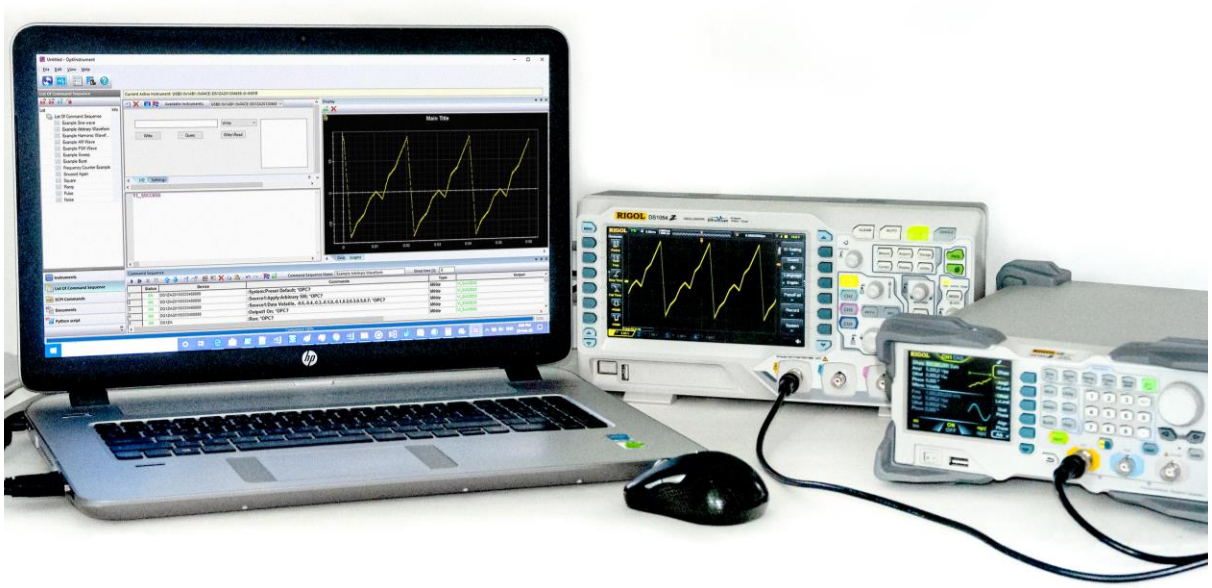
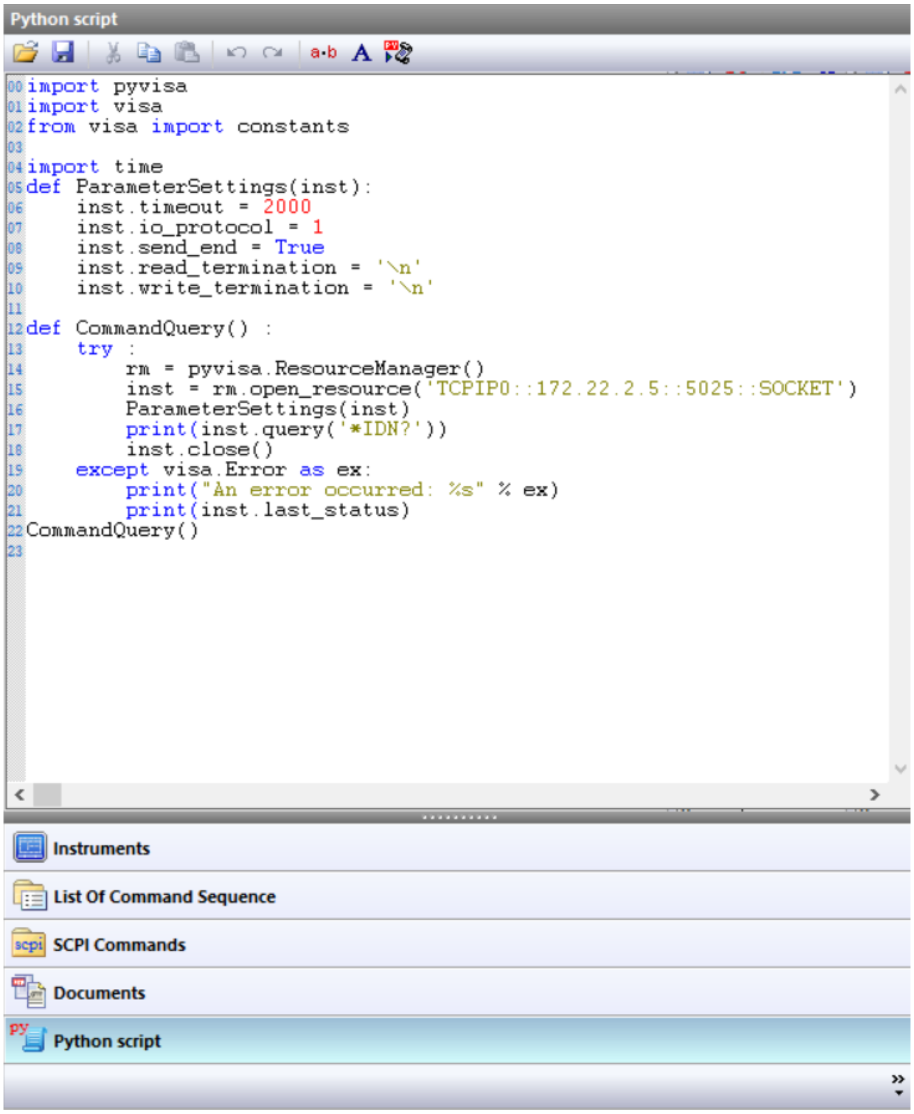


Fig. 4 OptiInstrument software communication and control a function generator and an oscilloscope

OptiInstrument & Python Script

OptiInstrument software supports Python scripting. A Python script is generated for a single SCPI command or a list of commands using the tool. The generated script can be saved into a file. The generated script can be executed from OptiInstrument GUI or in **command prompt** or **Windows PowerShell**. A Python script can be loaded into OptiInstrument GUI and executed by the GUI. Fig. 5 shows a Python script generated for a sequence of SCPI commands and displayed in the GUI Python script pane. This capability allows users to execute features that are not supported by OptiInstrument GUI such as logic control and looping options.



```
import pyvisa
import visa
from visa import constants

import time
def ParameterSettings(inst):
    inst.timeout = 2000
    inst.io_protocol = 1
    inst.send_end = True
    inst.read_termination = '\n'
    inst.write_termination = '\n'

def CommandQuery() :
    try :
        rm = pyvisa.ResourceManager()
        inst = rm.open_resource('TCPIP0::172.22.2.5::5025::SOCKET')
        ParameterSettings(inst)
        print(inst.query('*IDN?'))
        inst.close()
    except visa.Error as ex:
        print("An error occurred: %s" % ex)
        print(inst.last_status)
CommandQuery()
```

Fig. 5 Generated Python script for a sequence of SCPI command displayed in the Python script pane of OptiInstrument GUI

OptiInstrument 1.0 Example Library

OptiInstrument 1.0 Software has many examples that are created using commercial instruments from Rigol and EXFO. The examples are organized in directories for each vendor. Each example has a readme file that describes the setup and the instruments/cards used in each example as well as the result files.

1. EXFO Samples

a. Power Balzer_CFP4_EBERT
b. Power Balzer_QSFP_EBERT
c. Power Meter

d. Switch
e. Switch with CW Sources
f. VOA with OSA
g. VOA with Power Meter
h. Looping Execution of SCPI Commands
i. Data Parsing for Results of Looping Execution of SCPI Commands Example

2. RIGOL Samples

a. AM waveform
b. Arbitrary waveform
c. Burt waveform
d. Harmonic waveform
e. PSK waveform
f. Pulse waveform
g. Ramp waveform
h. Sinewave
i. Square waveform