# Granite River Labs
# Python Programming API's User Guide
# for Using the GRL 5-Port Switch Board Test Fixture (GRL-USB-PD-MULT)

All product names are trademarks, registered trademarks, or service marks of their respective owners.

The GRL disclaims all warranties and liability for the use of this document and the information contained herein and assumes no responsibility for any errors that may appear in this document, nor does the GRL make a commitment to update the information contained herein.

Contact the GRL office to obtain the latest revision of this document.

Questions regarding this document may be forwarded to:

Granite River Labs 3500 Thomas Road, Suite A, Santa Clara, CA 95054

Phone: 408-627-7608 Fax: 408-912-1810

E-mail: info@graniteriverlabs.com

http://www.graniteriverlabs.com

DISCLAIMER

This document is provided "as is" with no warranties whatsoever, including any warranty of merchantability, no infringement, fitness for any particular purpose, or any warranty otherwise arising out of any proposal, specification, or sample. The GRL disclaims all liability for infringement of proprietary rights, relating to use of information in this specification. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted herein.

All product names are trademarks, registered trademarks, or service marks of their respective owners.

# TABLE OF CONTENTS

# Revision Record

| Version | Date | Description of Changes | Author(s) |
|---|---|---|---|
| 1.0 | 01/2020 | New User Guide Creation | Pooja SM (GRL)<br>pooja@graniteriverlabs.in<br>Ong Gaik Pheng (GRL)<br>gpong@graniteriverlabs.com |

# 1 Scope of this User Guide

This User Guide helps you get started with Python programming API's for the GRL 5-Port Switch Board (GRL-USB-PD-MULT option).

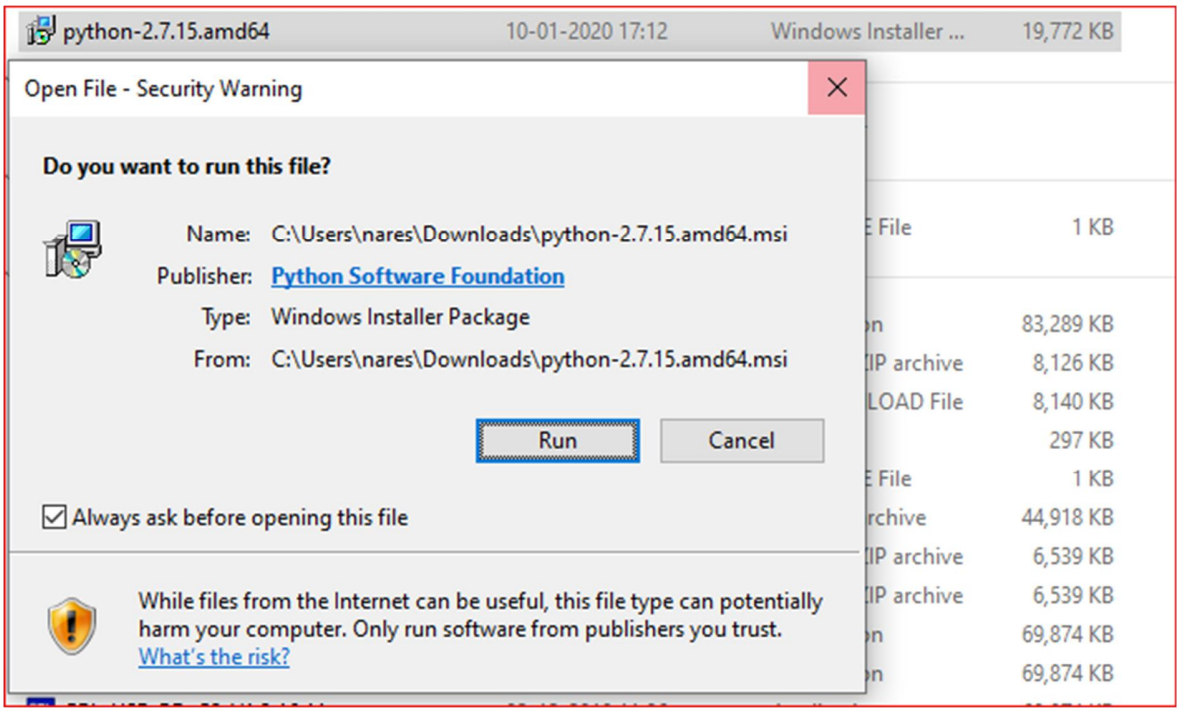
The GRL 5-Port Switch Board is provided as a separate test fixture accessory for the GRL-USB-PD-C2 controller. The GRL 5-Port Switch Board consists of a five ports switch extension fixture that is plugged in to the GRL-USB-PD-C2 controller and used to connect up to five Unit Under Tests (UUT's) to perform switching during tests. For more information on the GRL 5-Port Switch Board, please contact support@graniteriverlabs.com.

# 2 Getting Started with GRL-USB-PD-MULT Python API's

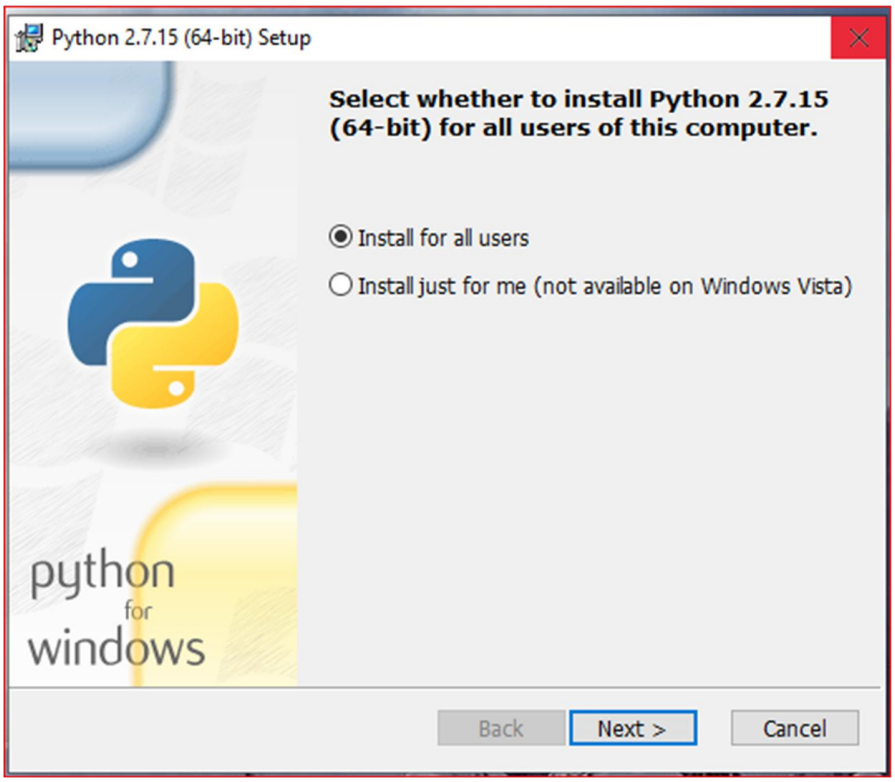
This section explains how to install and use the following Python API's to create test cases to meet more customized test requirements.

## 2.1 Python Software Installation

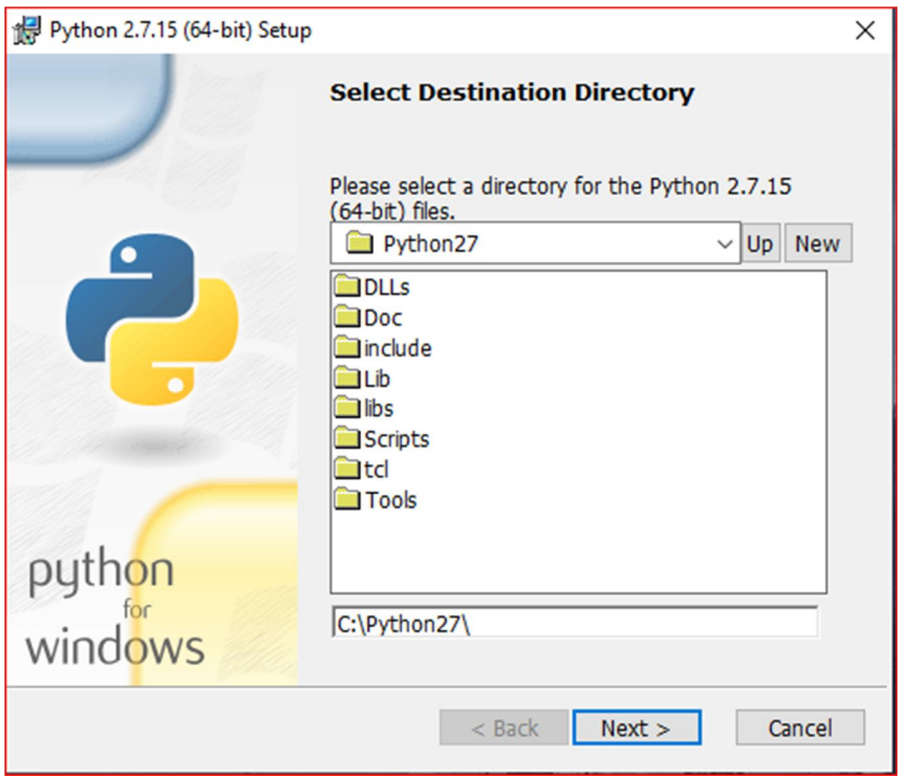
1. Download the Python software version 2.7.15 and install to your PC.



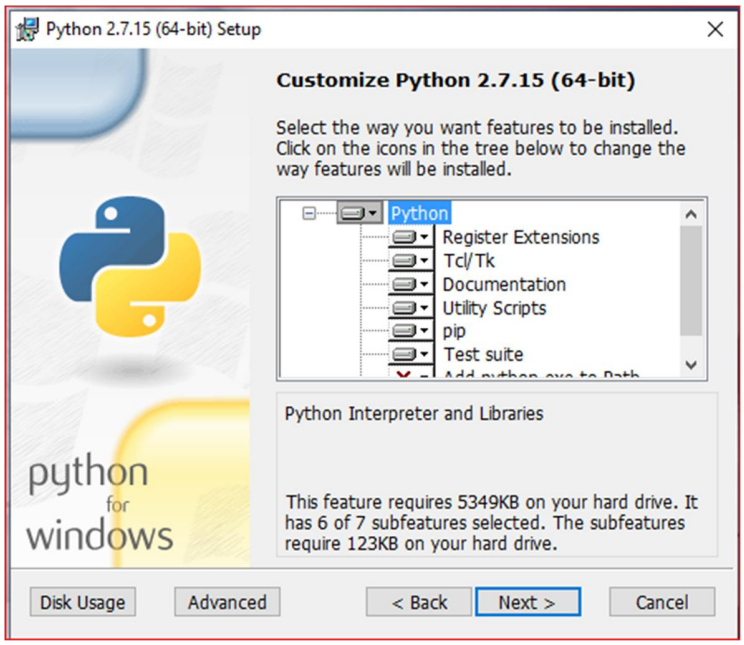

2. Click **Next** to install the Python software.



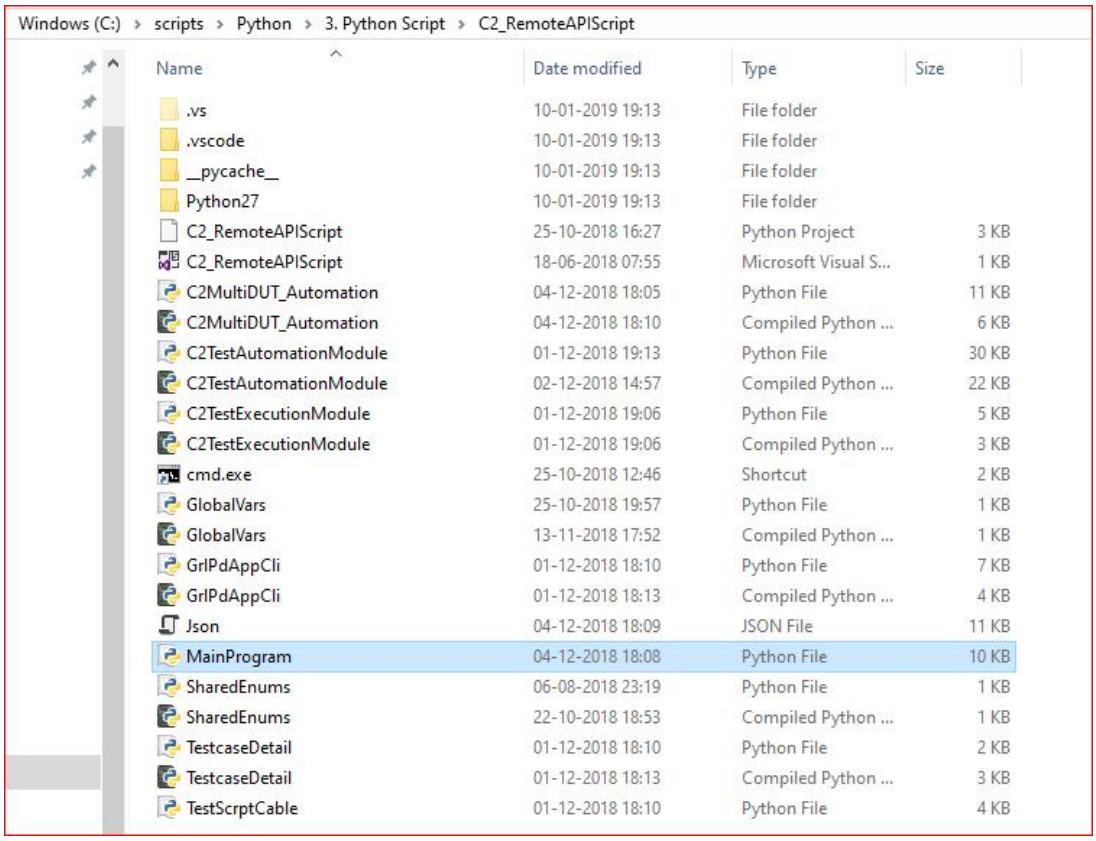

3. Select the directory for the Python files and then click **Next**.



4. Set up the libraries for the Python features and then click **Next**.

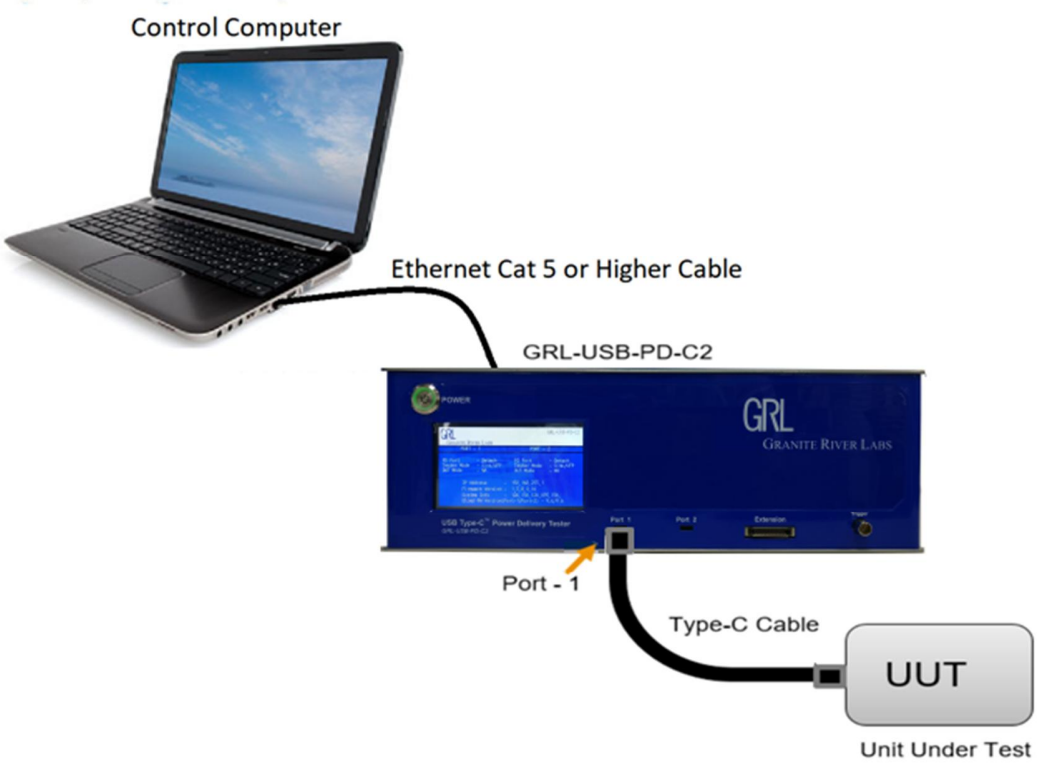



5. Follow the remaining instructions to complete the Python installation.
6. Once installation has completed, open the GRL-USB-PD-C2 controller remote API Python scripts from *C:\scripts\Python\3. Python Script\C2_RemoteAPIScript*.



## 2.2 Hardware Connection Setup

Figure below shows an example hardware setup for testing a USB Power Delivery Provider, Consumer or Dual Role Powered device, or USB Type-C hub or cable using the Python API's running on a control computer and connected via Ethernet to the GRL-USB-PD-C2 controller that is attached to the USB device to be tested through one of the two controller USB Type-C test ports.



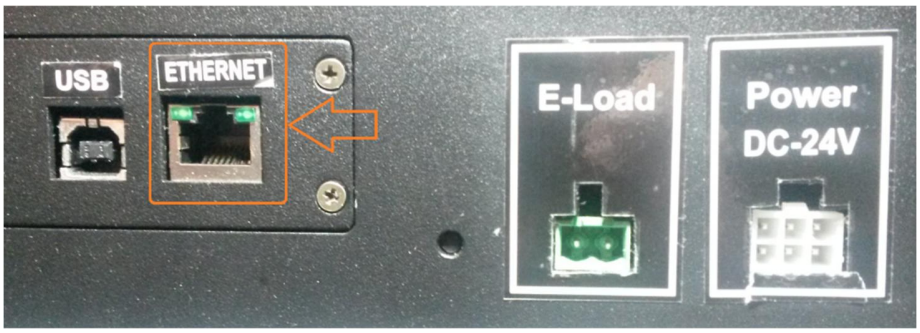

Below is a procedure for connecting the hardware and verifying proper hardware connections.

1. Connect a power supply to the GRL-USB-PD-C2 controller power interface using the 24V, 280W power brick included with the controller.
2. Connect the GRL-USB-PD-C2 controller to the control computer using a physical Ethernet connection.

   *Note: Automation of the UUT power supply switching in the GRL-USB-PD-C2 is handled internally to the controller. Thus, there is no Ethernet, USB or GPIB connection attached to the power supply.*

### 2.2.1 Connect Ethernet Cable and Turn On Controller

1. Connect the Ethernet (RJ-45) connector to one of the control computer's Ethernet ports. A USB to Ethernet adapter can be used if there are no native Ethernet ports on the control computer.

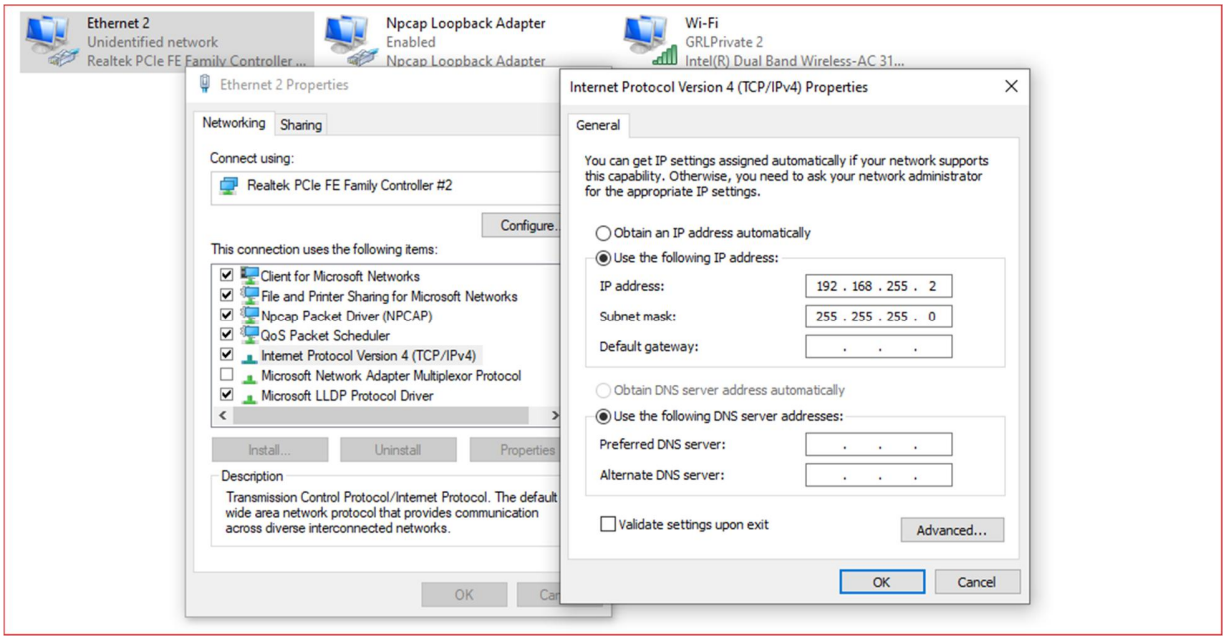



2. Turn on the GRL-USB-PD-C2 controller using the Power button on the front of the instrument.

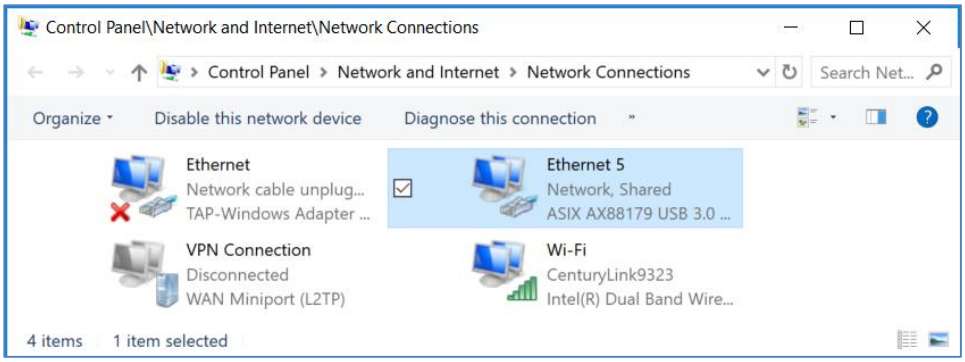

### 2.2.2 Verify Controller Ethernet Installation

The Ethernet port on the control computer needs to be configured correctly for the GRL-USB-PD-C2 controller to recognize the control computer and vice versa.

To make sure the network connection is set up correctly, open the Network Connections panel from the control panel and set up the Ethernet properties as shown below:



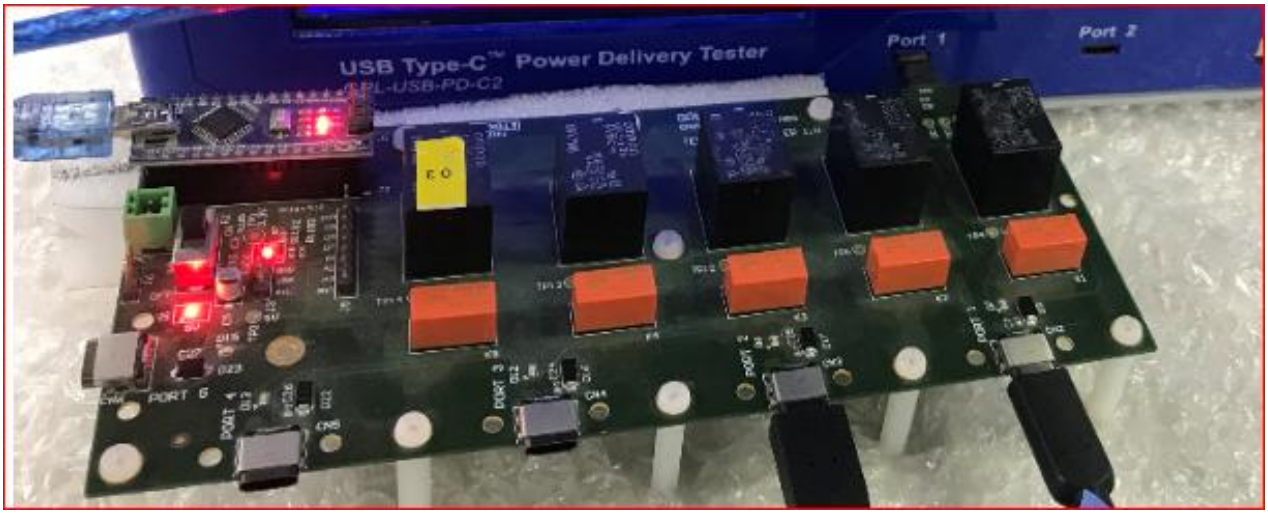
Click the 'OK' button on the Internet Protocol Properties and close the Ethernet Properties. Make sure the GRL-USB-PD-C2 controller is powered on and completely booted up (front panel display showing firmware version number), then connect the Ethernet cable from the GRL-USB-PD-C2 controller to the control computer's Ethernet port that was just set up. The network connections panel should now look as shown below:




The GRL-USB-PD-C2 controller is now set up and ready for use.

### 2.2.3 Connect GRL-USB-PD-MULT Board to Controller and Control Computer

The GRL 5-Port Switch Board test fixture (GRL-USB-PD-MULT board) contains five USB Type-C ports that can be used to connect and test five UUT's at one time (one after another) or a single port UUT.




1. Connect the GRL-USB-PD-MULT board to one of the Test Ports of the GRL-USB-PD-C2 controller as shown above.
2. Connect a Mini-B USB cable from the GRL-USB-PD-MULT board to the control computer.
3. Make sure the Ethernet cable has been connected from the GRL-USB-PD-C2 controller to the control computer's Ethernet port (as described in previous sections) before running the five-port/single port switch function.

## 2.3 Using GRL-USB-PD-MULT Python Automation Control Scripts

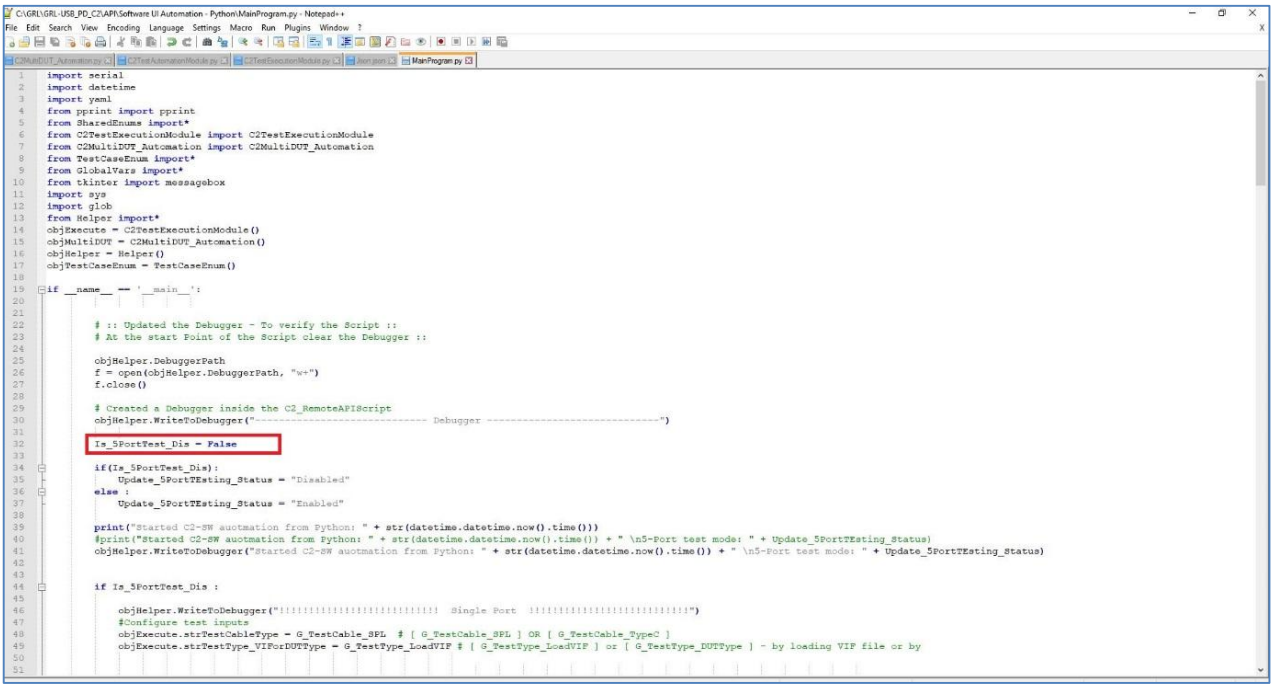
The GRL-USB-PD-MULT Python test scripts allow automation control for single port and five-port testing with the GRL 5-Port Switch Board.

### 2.3.1 Running MainProgram.py Script

The MainProgram.py script is used to run scripts with the requirements given in the respective fields.

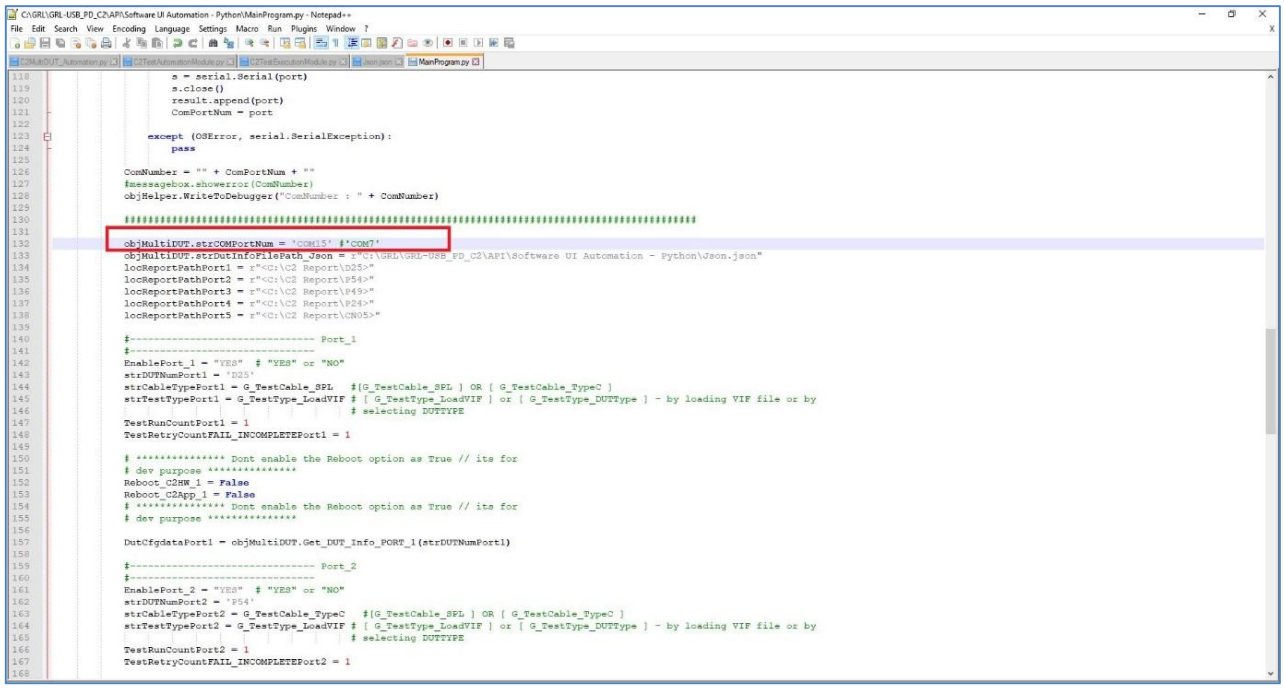
- Go to *C:\scripts\Python\3. Python Script\C2_RemoteAPIScript\MainProgram.py* and open **MainProgram.py** in Notepad++.

Note: Single port testing is enabled by default in the MainProgram.py script; to enable five-port testing, set the `Is_5PortTest_Dis` field code as **False** as shown below:



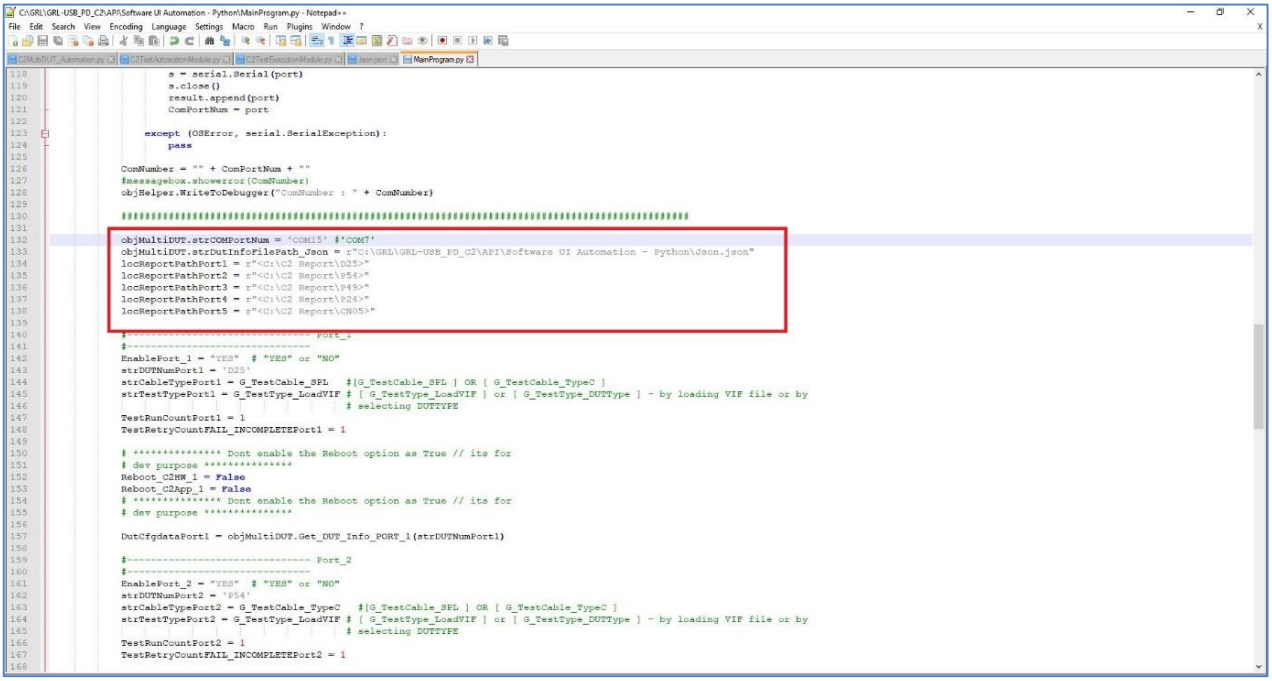
```
import serial
import datetime
import yaml
from pprint import pprint
from SharedEnums import*
from C2TestExecutionModule import C2TestExecutionModule
from C2MultiDUT_Automation import C2MultiDUT_Automation
from TestCaseEnum import*
from GlobalVars import*
from tkinter import messagebox
import sys
import glob
from Helper import*
objExecute = C2TestExecutionModule()
objMultiDUT = C2MultiDUT_Automation()
objHelper = Helper()
objTestCaseEnum = TestCaseEnum()

if __name__ == '__main__':

    # :: Updated the Debugger - To verify the Script ::
    # At the start Point of the Script clear the Debugger ::

    objHelper.DebuggerPath
    f = open(objHelper.DebuggerPath, "w+")
    f.close()

    # Created a Debugger inside the C2_RemoteAPIScript
    objHelper.WriteToDebugger("------------------------------ Debugger ------------------------------")

    Is_5PortTest_Dis = False

    if(Is_5PortTest_Dis):
        Update_5PortTEsting_Status = "Disabled"
    else :
        Update_5PortTEsting_Status = "Enabled"

    print("Started C2-SW auotmation from Python: " + str(datetime.datetime.now().time()))
    #print("Started C2-SW auotmation from Python: " + str(datetime.datetime.now().time()) + " \n5-Port test mode: " + Update_5PortTEsting_Status)
    objHelper.WriteToDebugger("Started C2-SW auotmation from Python: " + str(datetime.datetime.now().time()) + " \n5-Port test mode: " + Update_5PortTEsting_Status)


    if Is_5PortTest_Dis :

        objHelper.WriteToDebugger("!!!!!!!!!!!!!!!!!!!!!!!!!!!!  Single Port  !!!!!!!!!!!!!!!!!!!!!!!!!!!!")
        #Configure test inputs
        objExecute.strTestCableType = G_TestCable_SPL  # [ G_TestCable_SPL ] OR [ G_TestCable_TypeC ]
        objExecute.strTestType_VIForDUTType = G_TestType_LoadVIF # [ G_TestType_LoadVIF ] or [ G_TestType_DUTType ] - by loading VIF file or by
```

- Change to the correct COM Port Number according to the Cable connected to the control computer's port from the GRL 5-Port Switch Board fixture.



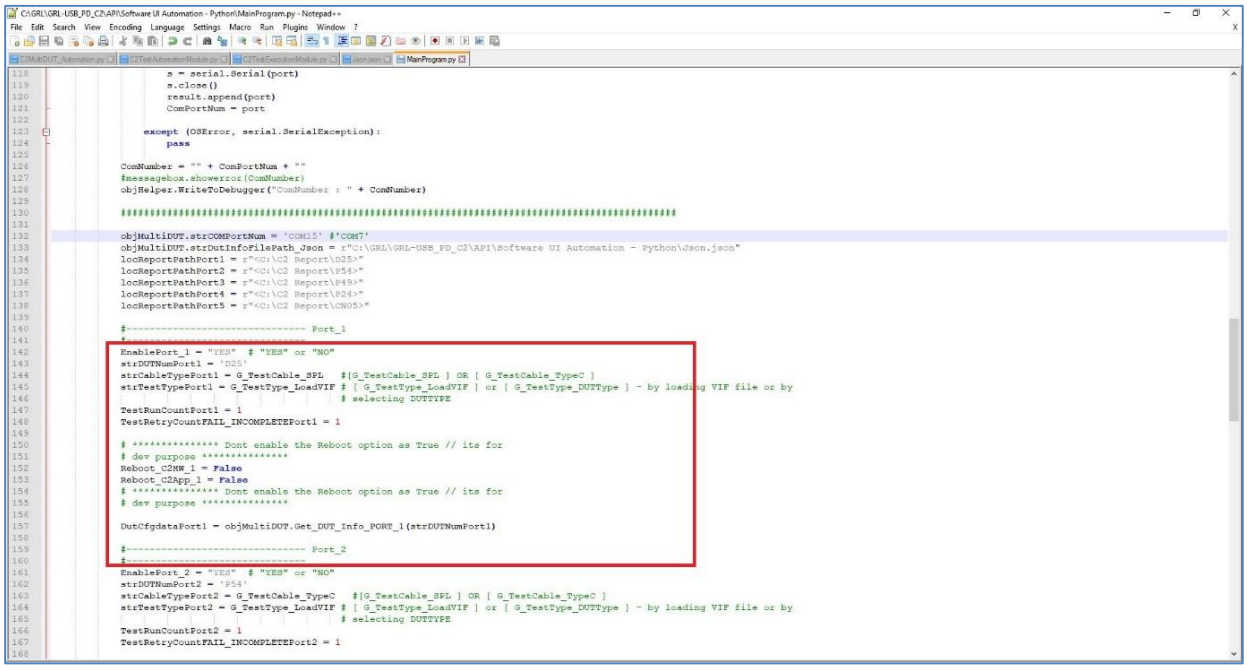
```
                s = serial.Serial(port)
                s.close()
                result.append(port)
                ComPortNum = port

            except (OSError, serial.SerialException):
                pass

        ComNumber = "" + ComPortNum + ""
        #messagebox.showerror(ComNumber)
        objHelper.WriteToDebugger("ComNumber : " + ComNumber)

        ##########################################################################################

        objMultiDUT.strCOMPortNum = 'COM15' #'COM7'
        objMultiDUT.strDutInfoFilePath_Json = r"C:\GRL\GRL-USB_PD_C2\API\Software UI Automation - Python\Json.json"
        locReportPathPort1 = r"<C:\C2 Report\D25>"
        locReportPathPort2 = r"<C:\C2 Report\P54>"
        locReportPathPort3 = r"<C:\C2 Report\P49>"
        locReportPathPort4 = r"<C:\C2 Report\P24>"
        locReportPathPort5 = r"<C:\C2 Report\CN05>"

        #------------------------------ Port_1
        #-----------------------------
        EnablePort_1 = "YES"  # "YES" or "NO"
        strDUTNumPort1 = 'D25'
        strCableTypePort1 = G_TestCable_SPL   #[G_TestCable_SPL ] OR [ G_TestCable_TypeC ]
        strTestTypePort1 = G_TestType_LoadVIF # [ G_TestType_LoadVIF ] or [ G_TestType_DUTType ] - by loading VIF file or by
                                              # selecting DUTTYPE
        TestRunCountPort1 = 1
        TestRetryCountFAIL_INCOMPLETEPort1 = 1

        # *************** Dont enable the Reboot option as True // its for
        # dev purpose ***************
        Reboot_C2HW_1 = False
        Reboot_C2App_1 = False
        # *************** Dont enable the Reboot option as True // its for
        # dev purpose ***************

        DutCfgdataPort1 = objMultiDUT.Get_DUT_Info_PORT_1(strDUTNumPort1)

        #------------------------------ Port_2
        #-----------------------------
        EnablePort_2 = "YES"  # "YES" or "NO"
        strDUTNumPort2 = 'P54'
        strCableTypePort2 = G_TestCable_TypeC   #[G_TestCable_SPL ] OR [ G_TestCable_TypeC ]
        strTestTypePort2 = G_TestType_LoadVIF # [ G_TestType_LoadVIF ] or [ G_TestType_DUTType ] - by loading VIF file or by
                                              # selecting DUTTYPE
        TestRunCountPort2 = 1
        TestRetryCountFAIL_INCOMPLETEPort2 = 1
```

- Define the test report file directory as shown in below example based on the location that you want to save the reports. The Json file directory should also refer to where the Json file is located.



```
                s = serial.Serial(port)
                s.close()
                result.append(port)
                ComPortNum = port

            except (OSError, serial.SerialException):
                pass

        ComNumber = "" + ComPortNum + ""
        #messagebox.showerror(ComNumber)
        objHelper.WriteToDebugger("ComNumber : " + ComNumber)

        ##########################################################################################

        objMultiDUT.strCOMPortNum = 'COM15' #'COM7'
        objMultiDUT.strDutInfoFilePath_Json = r"C:\GRL\GRL-USB_PD_C2\API\Software UI Automation - Python\Json.json"
        locReportPathPort1 = r"<C:\C2 Report\D25>"
        locReportPathPort2 = r"<C:\C2 Report\P54>"
        locReportPathPort3 = r"<C:\C2 Report\P49>"
        locReportPathPort4 = r"<C:\C2 Report\P24>"
        locReportPathPort5 = r"<C:\C2 Report\CN05>"

        #------------------------------ Port_1
        #-----------------------------
        EnablePort_1 = "YES"  # "YES" or "NO"
        strDUTNumPort1 = 'D25'
        strCableTypePort1 = G_TestCable_SPL   #[G_TestCable_SPL ] OR [ G_TestCable_TypeC ]
        strTestTypePort1 = G_TestType_LoadVIF # [ G_TestType_LoadVIF ] or [ G_TestType_DUTType ] - by loading VIF file or by
                                              # selecting DUTTYPE
        TestRunCountPort1 = 1
        TestRetryCountFAIL_INCOMPLETEPort1 = 1

        # *************** Dont enable the Reboot option as True // its for
        # dev purpose ***************
        Reboot_C2HW_1 = False
        Reboot_C2App_1 = False
        # *************** Dont enable the Reboot option as True // its for
        # dev purpose ***************

        DutCfgdataPort1 = objMultiDUT.Get_DUT_Info_PORT_1(strDUTNumPort1)

        #------------------------------ Port_2
        #-----------------------------
        EnablePort_2 = "YES"  # "YES" or "NO"
        strDUTNumPort2 = 'P54'
        strCableTypePort2 = G_TestCable_TypeC   #[G_TestCable_SPL ] OR [ G_TestCable_TypeC ]
        strTestTypePort2 = G_TestType_LoadVIF # [ G_TestType_LoadVIF ] or [ G_TestType_DUTType ] - by loading VIF file or by
                                              # selecting DUTTYPE
        TestRunCountPort2 = 1
        TestRetryCountFAIL_INCOMPLETEPort2 = 1
```

- Based on the DUT Type and Cables used, the following fields need to be modified for each Port (using Port 1 as example):
    - Set the `EnablePort_1` field code to "**YES**" in order to run the UUT connected to Port 1.
    - Modify the UUT's TID name in the `strDUTNumPort1` field code according to the respective Json file.
    - If the UUT is connected using the GRL-SPL cable, use the `strCableTypePort1 = G_TestCable_SPL` field code. If using the Captive cable, use the `strCableTypePort1 = G_TestCable_TypeC` field code.
    - If a VIF file is available for the UUT, use the `strTestTypePort1 = G_TestType_LoadVIF` field code to load the VIF file. If a VIF file is not available, use the `strTestTypePort1 = G_TestType_DUTType` field code to define the device type.
    - The `TestRunCountPort1` field code allows you to define the number of times to repeat running tests while the `TestRetryCountFAIL_INCOMPLETEPort1` field code allows you to set the number of times to repeat running the Fail and Incomplete test cases.
    - If you want to reboot the GRL-USB-PD-C2 controller and application for each Port, set the `Reboot_C2HW_1` and `Reboot_C2APP_1` to "**True**" or otherwise set to "**False**".
- Similarly follow the procedure as above for other Ports.



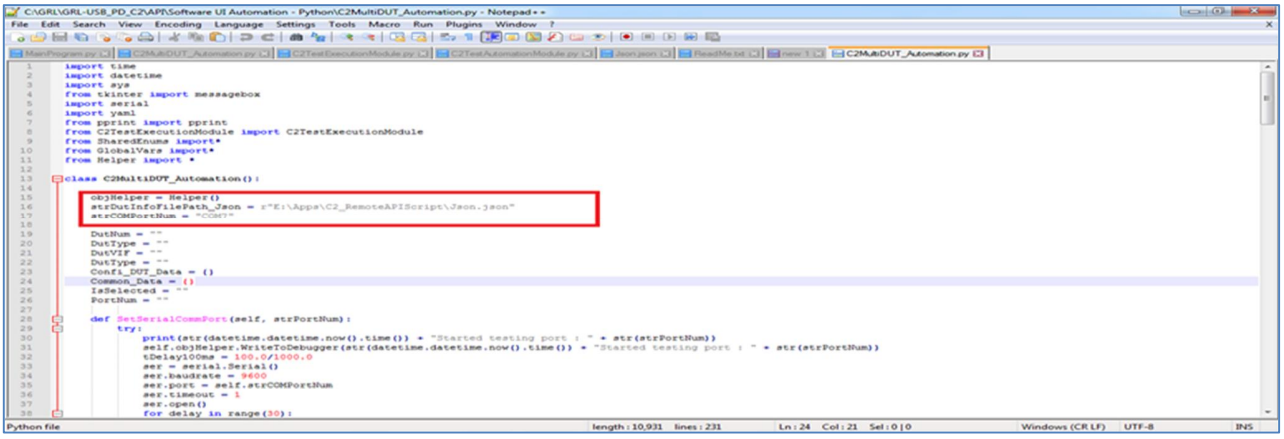
### 2.3.2 Running C2MultiDUT_Automation.py Script

The C2MultiDUT_Automation.py script is used to select test MOI's based on UUT's connected to specific test Ports.

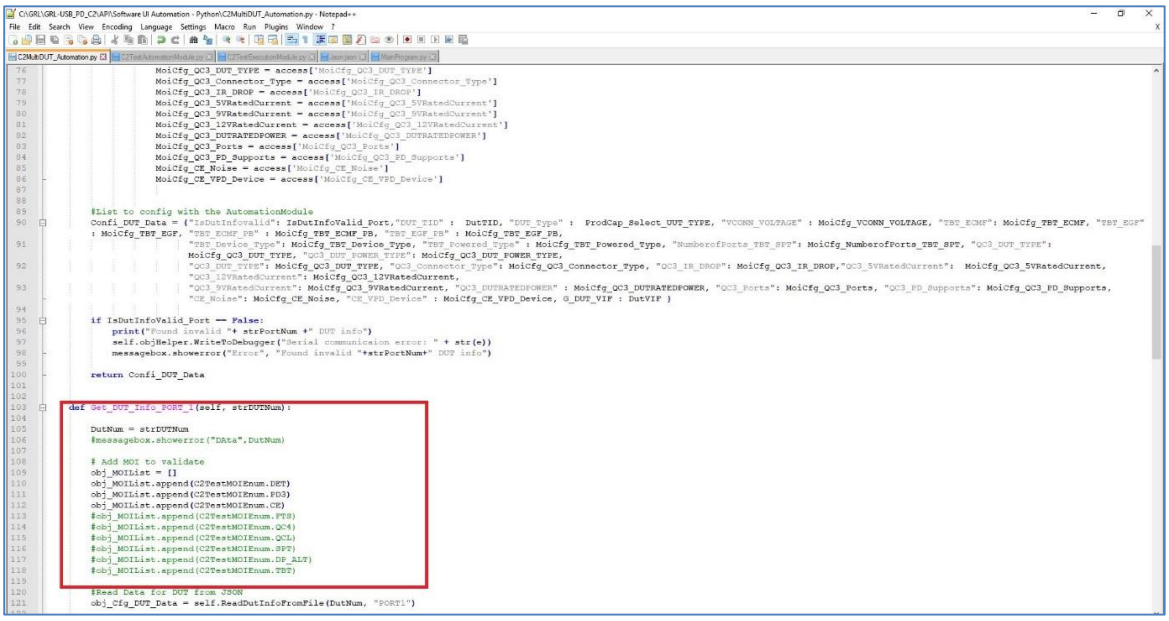
- Go to *C:\scripts\Python\3. Python Script\C2_RemoteAPIScript\C2MultiDUT_Automation.py* and open **C2MultiDUT_Automation.py** in Notepad++.

Note: The MainProgram.py script will run according to the selected MOI's.

1. Similar to the MainProgram.py script as described in the previous section, make sure to also change the Json file path and COM Port Number in the C2MultiDUT_Automation.py script file.



2. Select the MOI's to be tested for each port as shown in below example.
   - Specify "#" for those MOI's that you are not running for a specific Port.
   - Select MOI's based on the UUT type for each Port.

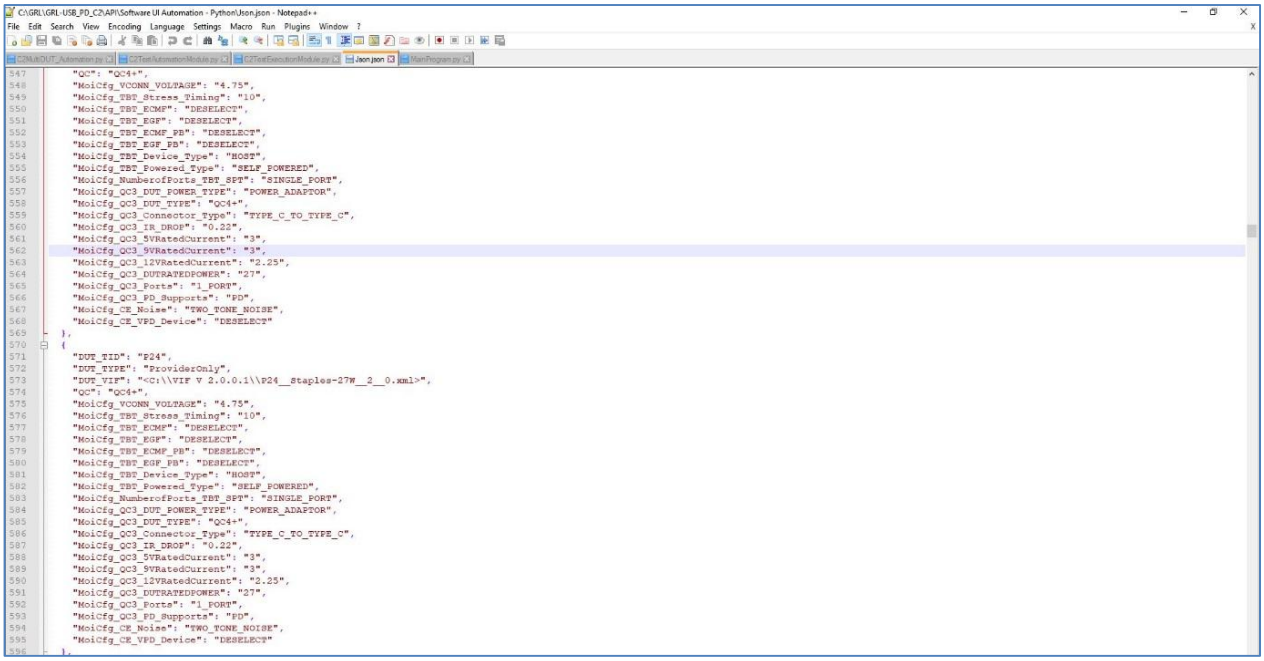


### 2.3.3 Running Json Script

The Json script contains all VIF file information and is used to call the VIF file from the given directory to the MainProgram.py script.

- Go to the *C:\scripts\Python\3. Python Script\C2_RemoteAPIScript\Json.json* and open **Json.json** in Notepad++.
- You will need to modify the VIF information based on the UUT before running tests.
- Modify required fields in the Json file, for example `DUT_VIF`, `DUT_TID` and `DUT_TYPE`.
- Make sure to specifically define the `DUT_TID` and `DUT_TYPE` field codes based on the UUT.
- Specify the VIF file directory where the VIF files are stored using the `DUT_VIF` field code. Make sure the Json script has VIF file directories specified for all UUT's.

  Note: Use "\\" instead of "\" and "_" instead of spacing when entering the VIF file directory.




**END_OF_DOCUMENT**