

# การเขียนโปรแกรม Python เชื่อมต่ออุปกรณ์ Rigol DS1052E Digital Oscilloscope ผ่านพอร์ต USB

เขียนโดย เรวัต ศิริโกศาภิรมย์

ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

ม.เทคโนโลยีพระจอมเกล้าพระนครเหนือ

เผยแพร่ครั้งแรก: วันที่ 20 พฤศจิกายน พ.ศ. 2560 (November 20, 2017)

ปรับปรุงแก้ไข: วันที่ 26 กันยายน พ.ศ. 2563 (September 26, 2020)

คำสำคัญ / Keywords: Digital Oscilloscope, Python for Automated Instrumentation and Measurement, Rigol DS1052E

## เครื่องมือวัดออสซิลโลสโคปแบบดิจิทัลและการเขียนโปรแกรมควบคุม

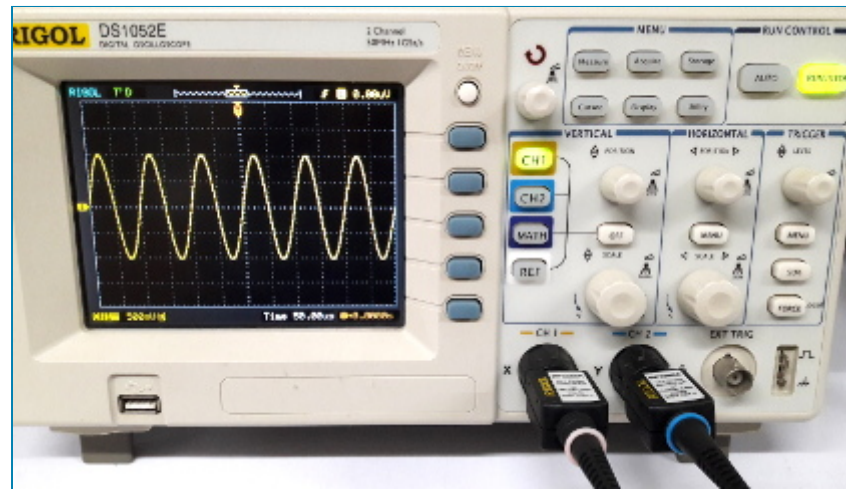
เครื่องมือวัดอย่างเช่น ออสซิลโลสโคป (Oscilloscope) เป็นสิ่งสำคัญในการทำงานด้านระบบสมองกลฝังตัว อิเล็กทรอนิกส์ และงานไฟฟ้าอื่น ๆ ในปัจจุบัน ออสซิลโลสโคปแบบดิจิทัล (Digital Oscilloscope) เริ่มมาแทนที่ออสซิลโลสโคปแบบแอนะล็อก (Analog Oscilloscope)

ด้วยเหตุผลหลาย ๆ ประการ เมื่อพิจารณาความสามารถและคุณลักษณะเฉพาะของเครื่องมือวัดออสซิลโลสโคปแบบดิจิทัล จะเห็นได้ว่าหลาย ๆ รุ่น มีพอร์ตสำหรับเชื่อมต่อกับคอมพิวเตอร์ เช่น RS232, USB และ Ethernet (RJ45) และสามารถใช้ซอฟต์แวร์จากคอมพิวเตอร์ควบคุมการทำงานของอุปกรณ์ดังกล่าวได้ นอกเหนือจากการควบคุมผ่านแผงควบคุม หรือปุ่มกดต่าง ๆ ของตัวเครื่องมือวัด

บทความนี้กล่าวถึง สาธิตการทดลองใช้ภาษาคอมพิวเตอร์ Python ในการเชื่อมต่อกับ Rigol Digital Oscilloscope (DS1000 Series) และได้เลือกทดลองกับเครื่องมือโมเดล DS1052E โดยเชื่อมต่อกับคอมพิวเตอร์ผ่านพอร์ต USB 2.0

การเขียนโปรแกรมสื่อสารกับเครื่องออสซิลโลสโคป จะอาศัยโปรโตคอลสื่อสารที่เรียกว่า VISA (Virtual Instrument Software Architecture) หรือในเชิงซอฟต์แวร์ ก็มองว่าเป็น Programming API สำหรับการเชื่อมต่อกับอุปกรณ์เครื่องมือวัด ซึ่งเป็นที่ยอมรับและใช้งานแพร่หลาย ส่วนคำสั่งที่ใช้ในการสื่อสารกับเครื่องมือวัดเหล่านี้ เรียกว่า SCPI (Standard Commands for Programmable

Instruments) และสามารถศึกษาการใช้งานได้จากเอกสาร Programming Guide ของผู้ผลิตเครื่องมือวัดแต่ละยี่ห้อหรือโมเดลเครื่องที่ใช้



รูปภาพของเครื่องออสซิลโลสโคป Rigol DS1052E



รูปภาพแสดงพอร์ตเชื่อมต่อของเครื่องออสซิลโลสโคป Rigol DS2000 Series  
เช่น พอร์ต USB พอร์ต RJ45 (LAN/LXI)



รูปภาพแสดงพอร์ตเชื่อมต่อของเครื่องออสซิลโลสโคป Rigol DS1052E  
เช่น พอร์ต RS232 และพอร์ต USB

สำหรับการใช้งานร่วมกับภาษา Python ก็มีไลบรารีที่ชื่อว่า PyVISA <https://pyvisa.readthedocs.io/> รองรับการใช้ VISA แต่ในกรณีที่ใช้ Windows จะต้องติดตั้งซอฟต์แวร์อย่างเช่น NI VISA <https://www.ni.com/visa/> ของบริษัท NI (National Instruments) ก่อนใช้งาน แนะนำให้ติดตั้ง NI-VISA Runtime 17.0 หรือเวอร์ชันใหม่กว่า แต่ถ้าใช้ Linux Ubuntu สามารถใช้ pyvisa-py (Pure Python Implementation of VISA) <https://github.com/pyvisa/pyvisa-py> แทนได้

การใช้ภาษา Python นั้นสามารถใช้ได้ทั้งระบบปฏิบัติการ Windows, Linux และ MAC OS X แต่ในบทความนี้จะกล่าวถึงเฉพาะการใช้งานสำหรับระบบปฏิบัติการ Windows 10 และได้ติดตั้ง Python 3.x ไว้แล้ว การทดลองใช้คำสั่งเพื่อรัน Python Script จะกระทำผ่าน Git Bash for Windows <https://git-scm.com/download/win>

### วัตถุประสงค์การเรียนรู้

- เรียนรู้ตัวอย่างการใช้ภาษา Python สำหรับควบคุมการทำงานของเครื่องมือวัดสัญญาณ เช่น ออสซิลโลสโคปแบบดิจิทัล
- ศึกษาขั้นตอนการติดตั้งและใช้งานซอฟต์แวร์สำหรับเชื่อมต่อกับเครื่องมือวัดโดยใช้ภาษา Python
- ทดลองใช้ Python Script ที่ให้ไว้เป็นตัวอย่างในการเชื่อมต่อและความคุมสั่งการเครื่องมือวัดผ่านพอร์ต USB ด้วยภาษา Python
- เรียนรู้คำสั่งต่าง ๆ เบื้องต้น ที่ใช้โปรแกรมเครื่องมือวัด Rigol Digital Oscilloscope Series 1000E

## รายการอุปกรณ์สำหรับการทดลอง

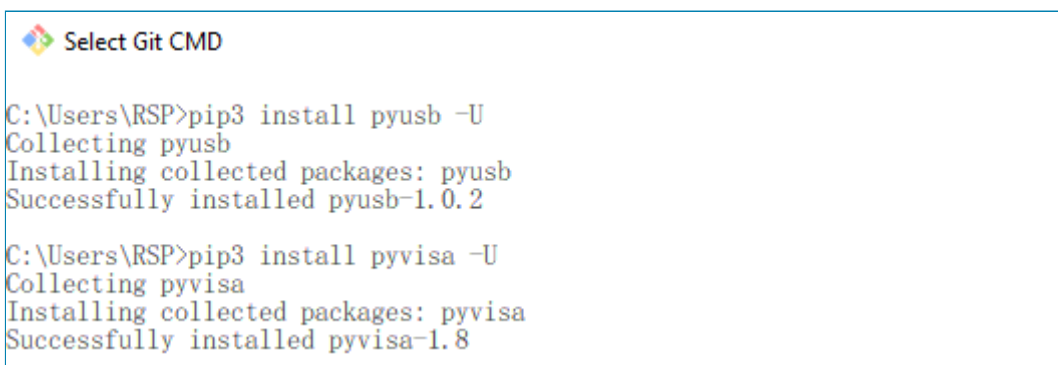
- เครื่องมือวัด Rigol Digital Oscilloscope รุ่น DS1052E พร้อมสาย Probe อย่างน้อย 1 เส้น
- สาย USB type-B สำหรับเชื่อมต่อระหว่างคอมพิวเตอร์และออสซิลโลสโคป
- คอมพิวเตอร์สำหรับรันสคริปต์ Python และติดตั้งซอฟต์แวร์ที่จำเป็นในการทำงานไว้แล้ว
- เครื่องสร้างคลื่นสัญญาณ (Function Generator / Waveform Generator) พร้อมสายสัญญาณ 1 เส้น สำหรับสร้างสัญญาณทดสอบ

## ขั้นตอนการติดตั้งแพ็คเกจสำหรับ Python สำหรับ Windows 10

```
pip3 install pyusb -U
pip3 install pyvisa -U
pip3 install pyvisa-py -U
pip3 install numpy -U
pip3 install matplotlib -U
```

## ขั้นตอนการติดตั้งแพ็คเกจสำหรับ Python สำหรับ Linux Ubuntu

```
sudo -H pip3 install pyusb -U
sudo -H pip3 install pyvisa -U
sudo -H pip3 install pyvisa-py -U
sudo -H pip3 install numpy -U
sudo -H pip3 install matplotlib -U
```

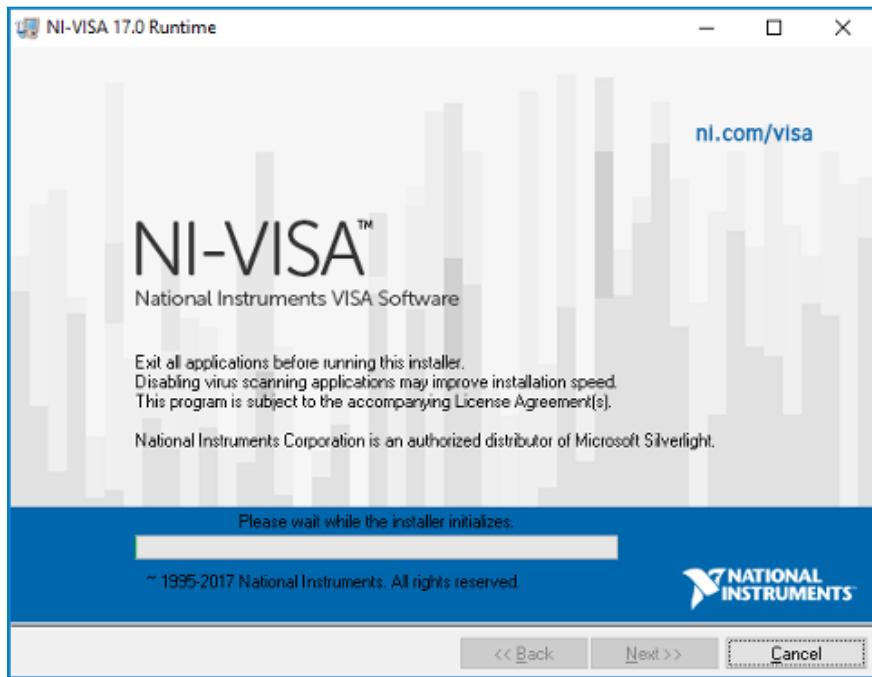


```
Select Git CMD

C:\Users\RSP>pip3 install pyusb -U
Collecting pyusb
Installing collected packages: pyusb
Successfully installed pyusb-1.0.2

C:\Users\RSP>pip3 install pyvisa -U
Collecting pyvisa
Installing collected packages: pyvisa
Successfully installed pyvisa-1.8
```

การติดตั้ง Python Packages เช่น pyusb, pyvisa โดยใช้คำสั่ง PIP สำหรับ Python 3



การติดตั้ง NI-VISA Runtime สำหรับ Windows

## ตัวอย่าง Python Script สำหรับทดสอบการเชื่อมต่อกับอุปกรณ์เครื่องมือวัด

```
#!/usr/bin/env python

#####
# Author: Rawat S.
# (Dept. of Electrical & Computer Engineering, KMUTNB, Bangkok/Thailand)
#####

import visa
import time, sys

visa_driver = '' # use either 'visa64' or 'visa32' or '@py' or left empty.
resources = visa.ResourceManager( visa_driver )
devices = resources.list_resources()
usb_device = None

if len(devices) > 0:
    print ('Found #devices: %d' % len(devices) )
    for device in devices:
        print ('>>', device)
        if str(device).startswith('USB0'):
            usb_device = device
            print ('select:', str(device)) # select the first device found
            break

print (30*'-')
```

```

if usb_device != None:
    instr = resources.open_resource( usb_device ) # use the selected USB device
    instr.write( '*IDN?' )
    time.sleep(0.5)
    ret_str = instr.read()
    fields = ret_str.split(',')
    dev_model = fields[2]
    print( ', '.join(fields[0:3]) )
    print( dev_model )
else:
    print('No device found')
    sys.exit(-1)
print ( '\nDone....' )

```

จากตัวอย่างโค้ด จะเห็นได้ว่า การสื่อสารกับอุปกรณ์เครื่องมือวัดนั้น จะใช้การส่งคำสั่ง (Command) ซึ่งเป็นข้อความ ASCII ไปยังเครื่องมือวัด และหรืออ่านข้อมูลที่ตอบกลับ ลองดูตัวอย่างคำสั่งเช่น  
 "\*IDN?" เป็นคำสั่งที่ใช้สอบถามข้อมูลต่าง ๆ เกี่ยวกับเครื่องมือวัด เพื่อใช้ระบุว่าเป็นเครื่องรุ่นหรือโมเดลอะไร ถ้าอยากทราบว่ามีเครื่องรุ่นไหนรองรับคำสั่งใดบ้าง ให้ศึกษาจากเอกสารของผู้ผลิต ในกรณีของเครื่อง Rigol Series 1000D/E ลองดูเอกสารที่เป็นไฟล์ .PDF ต่อไปนี้ [RIGOL Programming Guide DS1000D/E Series Digital Oscilloscope \(2013\)](#)

ตัวอย่างเอาต์พุตเมื่อทดสอบกับเครื่อง Rigol DS1000 Series

```

Found #devices: 1
>> USB0::6833::1416::DS1ED125318023\x00::0::INSTR
select: USB0::6833::1416::DS1ED125318023\x00::0::INSTR
-----
Rigol Technologies, DS1052E, DS1ED125318023
DS1ED125318023

```

ตัวอย่างเอาต์พุตเมื่อทดสอบกับเครื่อง Rigol DS2000 Series

```

Found #devices: 1
>> USB0::6833::1200::DS2D154700765::0::INSTR
select: USB0::6833::1200::DS2D154700765::0::INSTR
-----
RIGOL TECHNOLOGIES, DS2302A, DS2D154700765
DS2D154700765

```

## ตัวอย่างที่ 2:

โค้ด Python Script สำหรับอ่านข้อมูลจาก Rigol Digital Oscilloscope (ใช้รุ่น DS1052E) แล้วแสดงรูปคลื่นสัญญาณ

```
#!/usr/bin/env python
import visa
import time, sys
import numpy as np
import matplotlib.pyplot as plot

DS1052E_ID = '0x0588'
INSTR_ID = DS1052E_ID
vendor_id = None
device_id = None
instr_model = None
resources = None
instr = None
visa_driver = '' # use either 'visa64' or 'visa32' or '@py' or left empty.
resources = visa.ResourceManager( visa_driver )
devices = resources.list_resources()

if len(devices) > 0:
    print ('Found #devices: %d' % len(devices) )
    for device in devices:
        print ('>>', device)
        device = device.replace(':', ',')
        fields = device.split(',')
        if len(fields) == 5 and fields[3].startswith('DS'):
            vendor_id = fields[1]
            device_id = fields[2]
            instr_model = fields[3]
            print (vendor_id, device_id, instr_model)

#####

def listInstruments():
    global resources
    devices = resources.list_resources()
    print (devices)

def selectInstrument( vendor_id, device_id, instr_model ):
    cmd_str = "USB0::%s::%s::%s::INSTR" % (vendor_id, device_id, instr_model)
    instr = resources.open_resource( cmd_str, timeout=100, chunk_size=1024000 )
    return instr
```

```

def cmdWrite(cmd, dly=0.1):
    global instr
    instr.write( cmd )
    time.sleep( dly )

def cmdRead(cmd, dly=0.1):
    global instr
    instr.write( cmd )
    time.sleep( dly )
    try:
        str = instr.read()
    except Exception as ex:
        print (ex)
        str = None
    return str

def cmdReadRaw(cmd, dly=0.1):
    global instr
    instr.write( cmd )
    time.sleep( dly )
    try:
        data = instr.read_raw()
    except Exception as ex:
        print (ex)
        data = None
    return data

def showInstrumentInfo():
    print ( cmdRead("*IDN?") )

#####

if vendor_id == '0x1AB1' and device_id == INSTR_ID:
    instr = selectInstrument( vendor_id, device_id, instr_model)
else:
    print ( 'No Rigol oscilloscope instrument found !!!' )
    sys.exit(-1)

showInstrumentInfo()

# connect to the remote instrument
cmdWrite( "SYSTem:REMOte" ) # change from LOCAL to REMOTE
cmdWrite( ":RUN" )         # enter run mode

# Oscilloscope Settings
time_per_div = 0.005
cmdWrite(':TIM:SCAL %f' % time_per_div ) # set timescale (seconds)
cmdWrite(':CHAN2:DISP OFF') # CH2: turn off display
cmdWrite(":CHAN1:COUP DC")  # CH1: use DC coupling
cmdWrite(':CHAN1:DISP ON')  # CH1: turn on display

```



```

cmdWrite(':CHAN1:SCAL 1.0') # CH1: set vertical scale to 1.0 V
cmdWrite(':CHAN1:OFFS 0.0') # CH1: set vertical offset to 0.0 V
time.sleep(1.0)

cmdWrite(":STOP")
time.sleep(1.0)

# get the sampling rate
sampling_rate = float(cmdRead(':ACQ:SAMP?',0.5).strip())
print ( 'Sampling rate: {:.3f} MHz'.format( sampling_rate*1e-6 ) )

# Retrieve oscilloscope settings
time_per_div    = float(cmdRead(":TIM:SCAL?",0.5).strip())
time_offset     = float(cmdRead(":TIM:OFFS?").strip())
volt_per_div    = float(cmdRead(":CHAN1:SCAL?").strip())
vertical_offset = float(cmdRead(":CHAN1:OFFS?").strip())

print ( 'Timescale:', time_per_div )
print ( 'Time Offset:', time_offset )
print ( 'Volt/Div Ch1:', volt_per_div )
print ( 'Vertical Offset Ch1:', vertical_offset )

cmdWrite(":WAV:MODE RAW") # set waveform mode to RAW
cmdWrite(":WAV:FORM BYTE")
cmdWrite(":WAV:POIN:MODE NOR")
cmdWrite(":WAV:SOUR CHAN1") # select channel 1 as source
cmdWrite(":WAV:STAR 1")
cmdWrite(":WAV:STOP 600")
cmdWrite(":WAV:RES") # reset waveform reading
cmdWrite(":WAV:BEG") # start waveform reading

# get waveform data for channel 1
rawdata = cmdReadRaw(":WAV:DATA? CHAN1",0.5)
cmdWrite(":RUN")
cmdWrite(":KEY:FORCE",0.5)
instr.close()

bytes_len = int( rawdata[2:10] ) # get the number of data points
print ( 'retrieve %d bytes' % bytes_len )
rawdata = rawdata[10:] # skip the first 10 bytes
data = np.frombuffer(rawdata, 'B' )
data = ((240 - data) * (volt_per_div/25))
data = data - (vertical_offset + volt_per_div * 4.6)

t_left  = time_offset - 6 * time_per_div
t_right = time_offset + 6 * time_per_div
ts = np.linspace( t_left, t_right, num=len(data))

#ts = np.arange( len(data) )
#ts = (ts * (time_per_div/50)) - ((time_per_div*6) - time_offset)

```

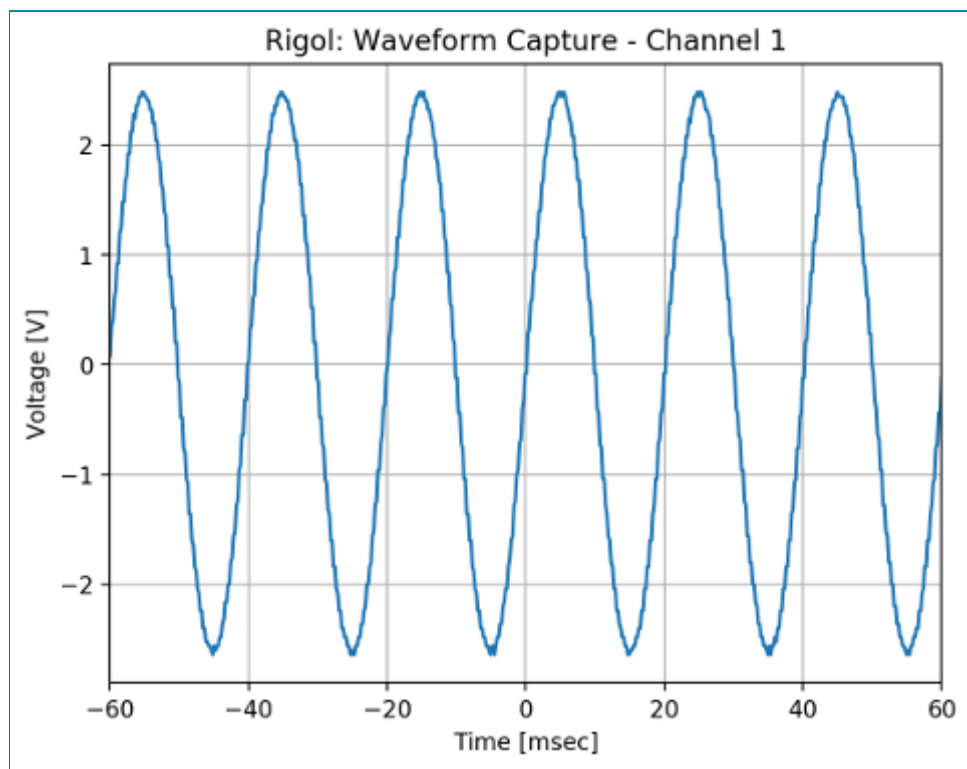
```

if (ts[-1] < 1e-3):
    ts = ts * 1e6
    ts_unit = "usec"
elif (ts[-1] < 1.0):
    ts = ts * 1e3
    ts_unit = "msec"
else:
    ts_unit = "sec"

plot.plot(ts, data)
plot.title( "Rigol: Waveform Capture - Channel 1" )
plot.ylabel( "Voltage [V]" )
plot.xlabel( "Time [%s]" % ts_unit )
plot.xlim( ts[0], ts[-1] )
plot.grid( True )
plot.savefig( 'rigol_plot.png',dpi=200,bbox_inches='tight' )
plot.show()

#####
# Reference:
# - http://www.righto.com/2013/07/rigol-oscilloscope-hacks-with-python.html
#####

```



รูปภาพตัวอย่างที่ได้จากการอ่านข้อมูลคลื่นสัญญาณ (ช่องแรกเท่านั้น) และแสดงผลเป็นรูปกราฟ

## สรุปผลการเรียนรู้ที่คาดหวัง

- ได้เรียนรู้การติดตั้งซอฟต์แวร์ที่จำเป็นสำหรับการเชื่อมต่อกับอุปกรณ์เครื่องมือวัดที่รองรับ VISA Programming API
- สามารถใช้ภาษา Python ในการติดต่อสื่อสารและเชื่อมต่อกับเครื่องมือวัดออสซิลโลสโคป Rigol DS1052E ผ่านพอร์ต USB
- ได้เห็นตัวอย่างการใช้คำสั่งเพื่อตั้งค่าการใช้งานออสซิลโลสโคป อย่างเช่น การกำหนดค่า VOLT/DIV และ TIME/DIV เป็นต้น
- สามารถนำข้อมูลคลื่นสัญญาณจากเครื่องมือวัดที่อ่านข้อมูลได้ด้วยคอมพิวเตอร์ ไปแสดงผลเป็นรูปภาพและบันทึกเป็นรูปภาพได้ โดยใช้ Python NumPy และ Matplotlib
- ได้ฝึกใช้ภาษา Python และประยุกต์ใช้งานด้านการเขียนโปรแกรมควบคุมเครื่องมือวัดทางไฟฟ้า

## แนวทางการเรียนรู้เพิ่มเติม

- ทดลองใช้กับเครื่องออสซิลโลสโคปรุ่นอื่น ๆ (ถ้ามี) เช่น DS1102E, DS1054Z, DS2072A เป็นต้น
- ทดลองใช้คำสั่งอื่น ๆ ที่เกี่ยวข้องกับการอ่านค่าเกี่ยวกับสัญญาณไฟฟ้า หรือตั้งค่าโหมด Trigger ที่นอกเหนือจาก AUTO เป็นต้น
- ทดลองสร้าง WebApp ที่สามารถควบคุมเครื่องออสซิลโลสโคปผ่านระบบเครือข่าย และแสดงข้อมูลผ่านหน้าเว็บได้ (เช่น ใช้ Python Django Framework หรือตัวเลือกอื่น)

**เผยแพร่ภายใต้ลิขสิทธิ์ / This work is licensed under: Attribution-NonCommercial 4.0 International (CC BY-NC-SA 4.0)**