การเขียนโปรแกรม Python อ่านข้อมูลจาก Rigol DS2000A ผ่านพอร์ต USB

เขียนโดย เรวัต ศิริโภคาภิรมย์
ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์ คณะวิศวกรรมศาสตร์
ม.เทคโนโลยีพระจอมเกล้าพระนครเหนือ (KMUTNB)

เผยแพร่ครั้งแรก: วันที่ 26 ธันวาคม พ.ศ. 2560 (December 26, 2017)

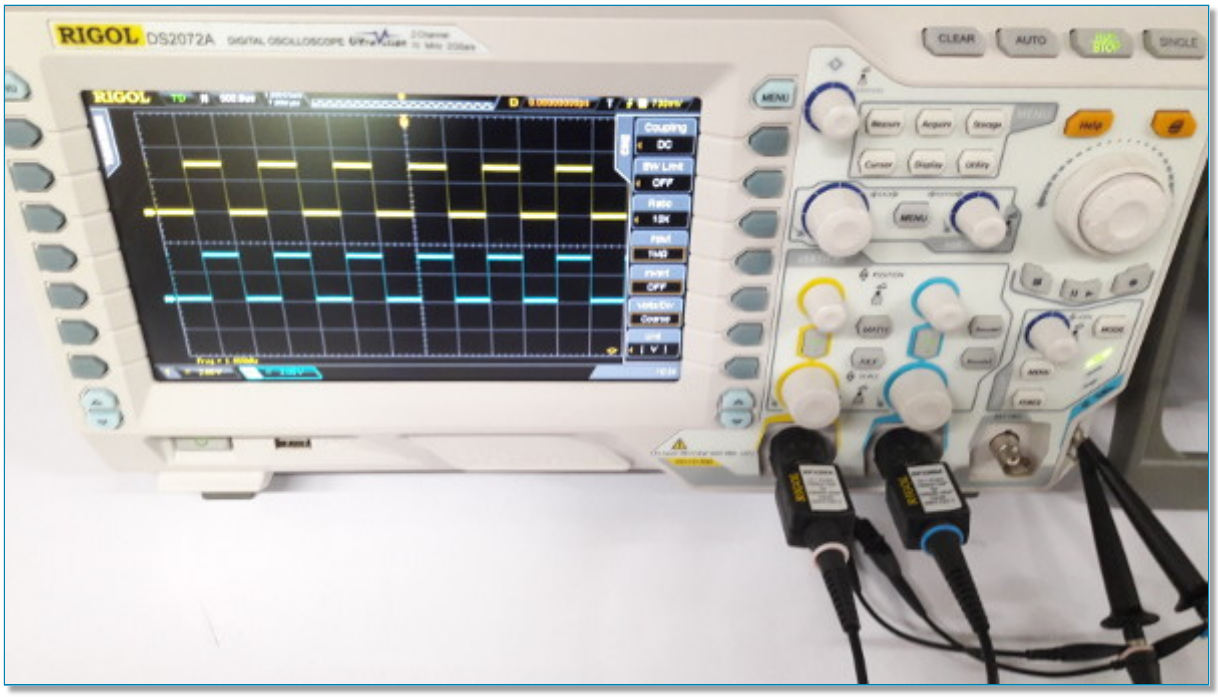
ปรับปรุงแก้ไข: วันที่ 26 กันยายน พ.ศ. 2563 (September 26, 2020)

คำสำคัญ / Keywords: *Digital Oscilloscope, Python for Automated Instrumentation and Measurement, Rigol DS2000A Series*

เครื่องมือวัดออสซิลโลสโคปแบบดิจิทัลและการเขียนโปรแกรมควบคุม

บทความนี้กล่าวถึง การสาธิตการทดลองใช้ภาษาคอมพิวเตอร์ Python ในการเชื่อมต่อกับ Rigol Digital Oscilloscope DS2000A Series และได้เลือกทดลองกับเครื่องโมเดล DS2072A โดยเชื่อมต่อกับคอมพิวเตอร์ผ่านพอร์ต USB 2.0

การเขียนโปรแกรมสื่อสารกับเครื่องมือวัดอย่างเช่น ออสซิลโลโคปแบบดิจิทัล จะอาศัยโปรโตคอลสื่อสารที่เรียกว่า VISA (Virtual Instrument Software Architecture) หรือในเชิงซอฟต์แวร์ ก็มองว่าเป็น Programming API สำหรับการเชื่อมต่อกับอุปกรณ์เครื่องมือวัด ซึ่งเป็นที่ยอมรับและใช้งานแพร่หลาย ส่วนคำสั่งที่ใช้ในการสื่อสารกับเครื่องมือวัดเหล่านี้ เรียกว่า SCPI (Standard Commands for Programmable Instruments) ผู้ที่สนใจสามารถศึกษาการใช้งานได้จากเอกสาร Programming Guide ของผู้ผลิตเครื่องมือวัดแต่ละยี่ห้อหรือโมเดลเครื่องที่ใช้



รูปภาพของเครื่องออสซิลโลสโคป Rigol DS2072A

วัตถุประสงค์การเรียนรู้

- เรียนรู้ตัวอย่างการเขียนโค้ดภาษา Python และไลบรารีที่เกี่ยวข้องสำหรับการนำมาใช้งานกับเครื่องมือวัด (Instrument) อย่างเช่น เครื่องออสซิลโลสโคปแบบดิจิทัล Digital Oscilloscope
- ทดลองใช้ Python Script ที่ให้ไว้เป็นตัวอย่าง ในการเชื่อมต่อและควบคุมสั่งการเครื่องมือวัดผ่านพอร์ต USB ด้วยภาษา Python
- เรียนรู้คำสั่งต่าง ๆ เบื้องต้น ที่ใช้โปรแกรมเครื่องมือวัด Rigol Series 2000A เพื่อวัดค่าที่เกี่ยวข้องกับคุณสมบัติของสัญญาณรูปไซน์

บทความนี้จะไม่กล่าวถึง การติดตั้งซอฟต์แวร์ที่เกี่ยวข้องกับการใช้งาน Python และ VISA Driver เพราะได้เขียนอธิบายไว้ในบทความอื่นที่เกี่ยวข้อง

รายการอุปกรณ์สำหรับการทดลอง

- เครื่องมือวัด Rigol Digital Oscilloscope (ทดลองใช้รุ่น **DS2072A**) พร้อมสาย Probe 2 เส้น
- สาย USB Type-B สำหรับเชื่อมต่อระหว่างคอมพิวเตอร์และออสซิลโลสโคป
- คอมพิวเตอร์สำหรับรันสคริปต์ Python และติดตั้งซอฟต์แวร์ที่จำเป็นในการใช้งานไว้แล้ว
- เครื่องสร้างคลื่นสัญญาณ (Function Generator / Waveform Generator) พร้อมสายสัญญาณ 2 เส้น สำหรับสร้างสัญญาณทดสอบ

ตัวอย่างโค้ด: Python Script

โค้ดตัวอย่างต่อไปนี้ ใช้สำหรับอ่านข้อมูลจาก Rigol Digital Oscilloscope **(DS2000A Series)** เพื่อวัดค่าพารามิเตอร์ต่าง ๆ ที่เกี่ยวกับสัญญาณรูปไซน์

```
#!/usr/bin/env python3
import visa
import time, sys
#############################################################################
# Author: Rawat S.
#   (Dept. of Electrical & Computer Engineering, KMUTNB, Bangkok/Thailand)
# Description:
#   Use Rigol DS2000A Series digital scope to measure signal properties
#   such as frequency, voltage (peak-to-peak) and phase difference.
#   For the input signals, a digital function generator is used to generate
#   two sinusiodal waveforms of the same frequency (50Hz) but with different
#   phase offsets.
#############################################################################
# select Rigol DS2072A : 0x04B0
vendor_id   = None
device_id   = None
instr_model = None
resources   = None
instr       = None
USE_PROBE_10X   = True
TOO_LARGE_VALUE = (9e+37)  # use to detect erroneous values
#############################################################################
visa_driver = '@py'  # use 'visa64', 'visa32' (Windows) or '@py' (Linux)
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]
            if not vendor_id.startswith('0x'):
                vendor_id = '0x{:04X}'.format( int(fields[1]) )
            if not device_id.startswith('0x'):
                device_id = '0x{:04X}'.format( int(fields[2]) )
            instr_model = fields[3]
            print (vendor_id, device_id, instr_model)
```

```
print (50*'-')

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

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

def selectInstrument( vendor_id, device_id, instr_model ):
    global instr
    cmd_str = "USB?::%s::%s::%s::INSTR" % (vendor_id,device_id,instr_model)
    instr = resources.open_resource( cmd_str, timeout=500, chunk_size=102400 )
    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:
        str = None
    return str[:-1]

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

if vendor_id == '0x1AB1' and device_id == '0x04B0':
    instr = selectInstrument( vendor_id, device_id, instr_model)
    print ( cmdRead("*IDN?") )
else:
    print ('No Rigol oscilloscope instrument found !!!')
    sys.exit(-1)

# connect to the remote instrument and send SCPI commands
cmdWrite( "SYSTem:REMote" )       # change from LOCAL to REMOTE
#cmdWrite( "*RST", 1.0 )           # reset the instrument
cmdWrite( ":RUN" )                # enter 'RUN' mode

# clear all measurements settings
cmdWrite( ":MEAS:CLE ALL" )

# turn on display for both CH1 and CH2
cmdWrite( ":CHAN1:DISP 1" )
cmdWrite( ":CHAN2:DISP 1" )
```

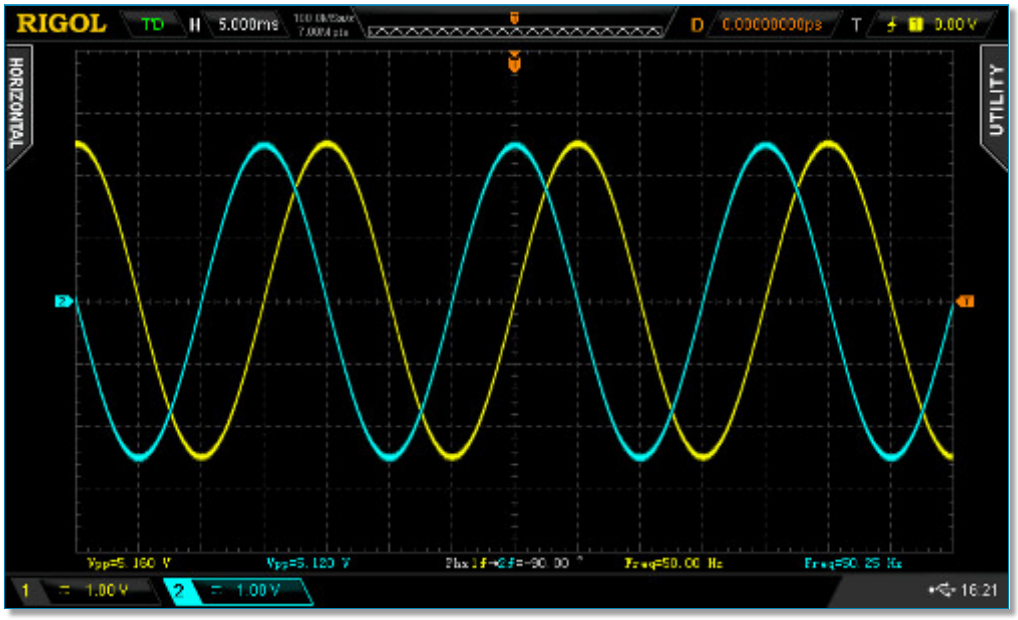
```
# set 1x probe for both CH1 and CH2
cmdWrite( ":CHAN1:PROB 1" )
cmdWrite( ":CHAN2:PROB 1" )

if USE_PROBE_10X:
    cmdWrite( ":CHAN1:PROB 10" )     # CH1 10x probe
    cmdWrite( ":CHAN2:PROB 10" )     # CH2 10x probe
time.sleep(0.5)

# select DC coupling for both CH1 and CH2
cmdWrite( ":CHAN1:COUP DC" )
cmdWrite( ":CHAN2:COUP DC" )

# set volt/div scale = 1.0V for both CH1 and CH2
cmdWrite( ":CHAN1:SCAL {:.1f}".format(1.0) )
cmdWrite( ":CHAN2:SCAL {:.1f}".format(1.0) )

# set vertical offset to 0.0V for both CH1 and CH2
cmdWrite( ":CHAN1:OFFS {:.3f}".format(0.0) )
cmdWrite( ":CHAN2:OFFS {:.3f}".format(0.0) )

# set time scale = 0.005 sec
cmdWrite( ":TIM:MAIN:SCAL {:.6f}".format( 0.005 ) )

# set trigger source = CH1, trigger mode = EDGE,
#     trigger sweep = AUTO, trigger level = 0.0V
cmdWrite( ":TRIG:EDG:SOUR CHAN1" )
cmdWrite( ":TRIG:MODE EDGE" )
cmdWrite( ":TRIG:SWE AUTO" )
cmdWrite( ":TRIG:EDG:LEV {:.3f}".format(0.0) )

# enable frequency measurement on both CH1 and CH2
cmdWrite( ":MEAS:FREQ CHAN1" )
cmdWrite( ":MEAS:FREQ CHAN2" )

# enable period measurement on both CH1 and CH2
cmdWrite( ":MEAS:PER CHAN1" )
cmdWrite( ":MEAS:PER CHAN2" )

# enable Vpp (Voltage peak-to-peak)  measurement on both CH1 and CH2
cmdWrite( ":MEAS:VPP CHAN1" )
cmdWrite( ":MEAS:VPP CHAN2" )

# set PSA=CHAN1 and PSB=CHAN2 for phase and delay measurement
cmdWrite( ":MEAS:SET:PSA CHAN1" )
cmdWrite( ":MEAS:SET:PSB CHAN2" )

# enable measurement of phase difference (rising-edge to rising-edge)
cmdWrite( ":MEAS:RPH CHAN1,CHAN2" )
```

```
time.sleep(1.0)

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

def read_vpp( channel ):
    resp = cmdRead( ":MEAS:VPP? CHAN{:d}".format( channel ), 0.2 )
    vpp = float(resp)
    if vpp > TOO_LARGE_VALUE:
        print ( "CHAN{:d} Vpp: ---- V".format( channel) )
    else:
        print ( "CHAN{:d} Vpp: {:.3f} V".format( channel, vpp ) )

def read_freq( channel ):
    resp = cmdRead( ":MEAS:FREQ? CHAN{:d}".format( channel ), 0.2 )
    freq = float(resp)
    if freq > TOO_LARGE_VALUE:
        print ( "CHAN{:d} Freq: ---- Hz".format( channel ) )
    else:
        print ( "CHAN{:d} Freq: {:.1f} Hz".format( channel, freq ) )

def read_phase_diff():
    resp = cmdRead( ":MEAS:RPH? CHAN1,CHAN2", 0.2 )
    phase_delay = float( resp )
    if phase_delay > TOO_LARGE_VALUE:
        print ( "Cannot measure the phase difference" )
    else:
        print ( "Phase difference CHAN1->CHAN2: {:.1f} deg.".format( phase_delay) )

# perform the measurements 3 times
print (50*'-')
for i in range(3):
    read_vpp(1)
    read_vpp(2)
    read_freq(1)
    read_freq(2)
    read_phase_diff()
    print (50*'-')

time.sleep(1.0)

if resources != None:
    resources.close()
    del resources
if instr != None:
    instr.close()
    del instr

sys.exit(0)
##############################################################################
```

จากโค้ดตัวอย่างข้างบน เมื่อนำไปใช้งานกับเครื่อง DS2072A จะตั้งค่าการวัดและแสดงสัญญาณดังนี้

- ตั้งค่าเพื่อใช้ตัวคูณ 10x สำหรับสายวัด (Probe) ทั้งสอง 2 ช่องอินพุต (CHAN1 และ CHAN2)
- ตั้งค่า Volt/div ไว้ที่ 1V per div (1 โวลต์ต่อหนึ่งช่อง) และ Vertical Offset = 0.0V ทั้งสอง 2 ช่องอินพุต
- ตั้งค่า Time/div ไว้ที่ 5 msec per div (5 มิลลิวินาทีต่อหนึ่งช่อง) ซึ่งจะใช้สำหรับวัดสัญญาณอินพุต 50Hz (ถ้าจะวัดสัญญาณแบบมีคาบที่มีความถี่ต่างไป จะต้องปรับค่า Time/div ให้เหมาะสม)
- ตั้งค่า Trigger เป็นโหมด AUTO, ค่า Trigger level = 0.0V, Trigger type แบบ Edge และเลือก Trigger Source = CHAN1
- แสดงการวัดค่าความถี่ (Frequency) แรงดัน (Voltage peak-to-peak: Vpp) และวัดค่าความต่างเฟสระหว่างสัญญาณอินพุต 2 ช่องสัญญาณ

สำหรับสัญญาณอินพุต จะได้จากเครื่องกำเนิดสัญญาณ โดยสร้างสัญญาณเป็นรูปไซน์ จำนวน 2 ช่องสัญญาณ มีความถี่ 50Hz และแอมพลิจูด 2.5V (Vpp = 5V) เหมือนกัน แต่มีความต่างเฟส 90 องศา (degrees)



รูปภาพคลื่นสัญญาณจำนวน 2 สัญญาณ แสดงผลโดยเครื่องออสซิลโลสโคป DS2072A

ตัวอย่างข้อความเอาต์พุตจากการทำงานของโค้ด Python (ทดลองรันโดยใช้คอมพิวเตอร์ที่ติดตั้งระบบปฏิบัติการ Ubuntu 16.04)

```
Found #devices: 1
>> USB0::6833::1200::DS2D154700765::0::INSTR
0x1AB1 0x04B0 DS2D154700765
------------------------------------------------
RIGOL TECHNOLOGIES,DS2302A,DS2D154700765,00.03.05.SP3
------------------------------------------------
CHAN1 Vpp: 5.160 V
CHAN2 Vpp: 5.120 V
CHAN1 Freq: 50.0 Hz
CHAN2 Freq: 50.3 Hz
Phase difference CHAN1->CHAN2: -90.0 deg.
------------------------------------------------
CHAN1 Vpp: 5.160 V
CHAN2 Vpp: 5.120 V
CHAN1 Freq: 50.3 Hz
CHAN2 Freq: 50.3 Hz
Phase difference CHAN1->CHAN2: -90.0 deg.
------------------------------------------------
CHAN1 Vpp: 5.120 V
CHAN2 Vpp: 5.120 V
CHAN1 Freq: 50.0 Hz
CHAN2 Freq: 50.3 Hz
Phase difference CHAN1->CHAN2: -91.8 deg.
------------------------------------------------
```

สรุปผลการเรียนรู้ที่คาดหวัง

- **ได้เรียนรู้ตัวอย่างการเขียนโค้ดภาษา** Python **ในการติดต่อสื่อสารและเชื่อมต่อกับเครื่องมือวัดออสซิลโลสโคป** Rigol DS2000A Series **ผ่านพอร์ต** USB
- **ได้เรียนรู้ตัวอย่างการใช้คำสั่ง** SCPI **สำหรับเครื่องมือวัด** Rigol DS2000A Series **เช่น การตั้งค่าต่าง ๆ สำหรับการวัดและแสดงผลสัญญาณของเครื่องออสซิลโลสโคป การอ่านค่าความถี่ หรือคาบของสัญญาณในแต่ละช่องสัญญาณ การอ่านค่าแรงดันของสัญญาณ การอ่านค่าความต่างเฟสของสัญญาณอินพุต** 2 **สัญญาณ เป็นต้น**