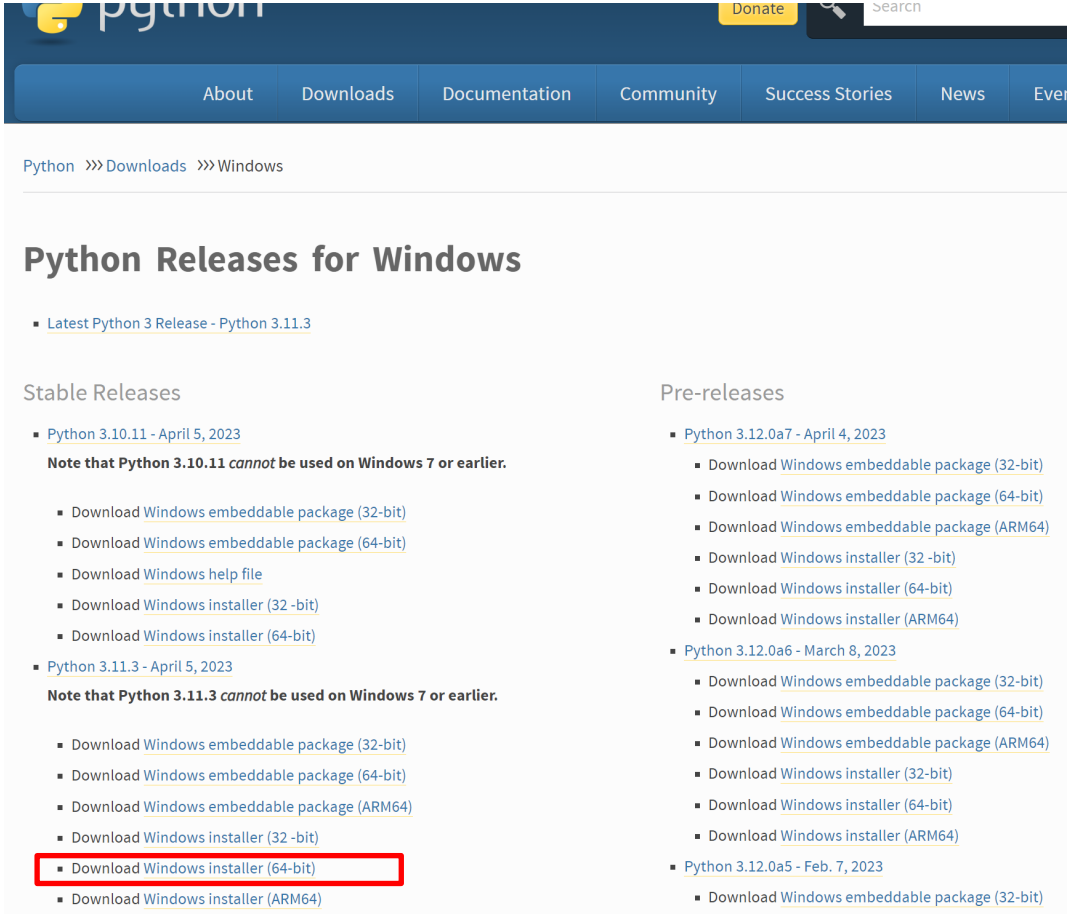
# Python을 이용한
# EEboard 제어 및 실습

**Shinwoong Kim**

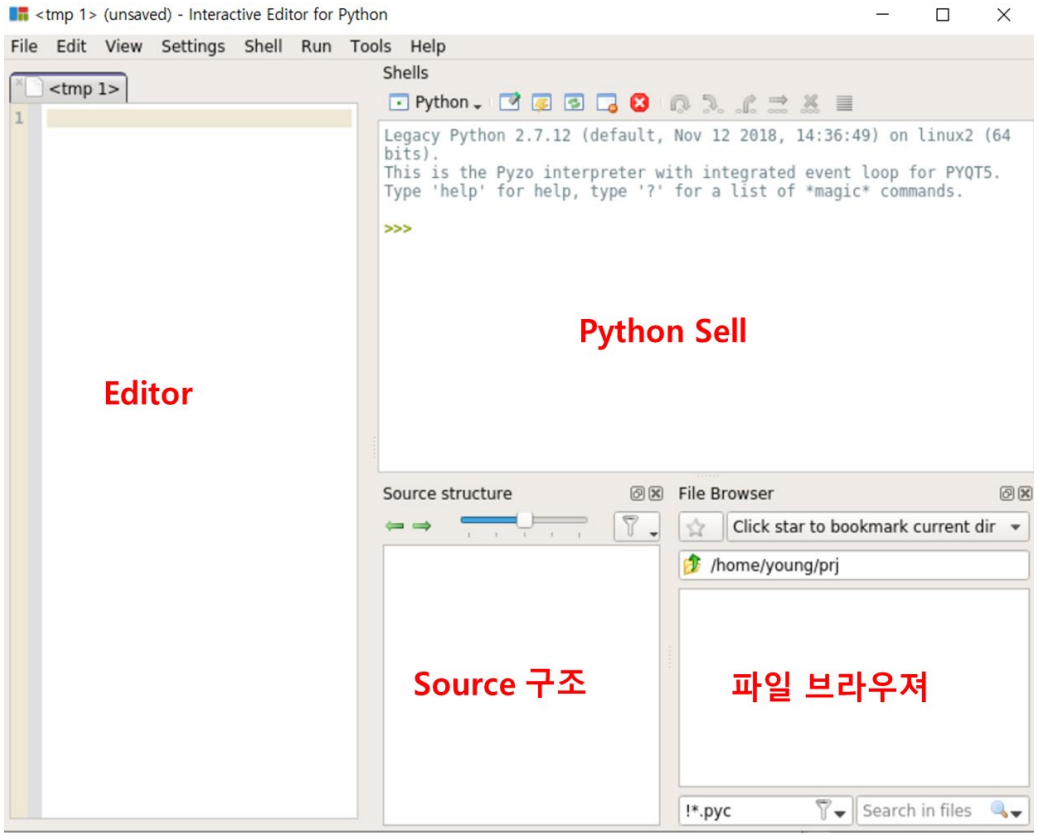
# Python

- **Python 설치**
  - ✓ https://www.python.org/
  - ✓ Downloads → Windows (OS 및 32bit/64bit을 적절히 선택)



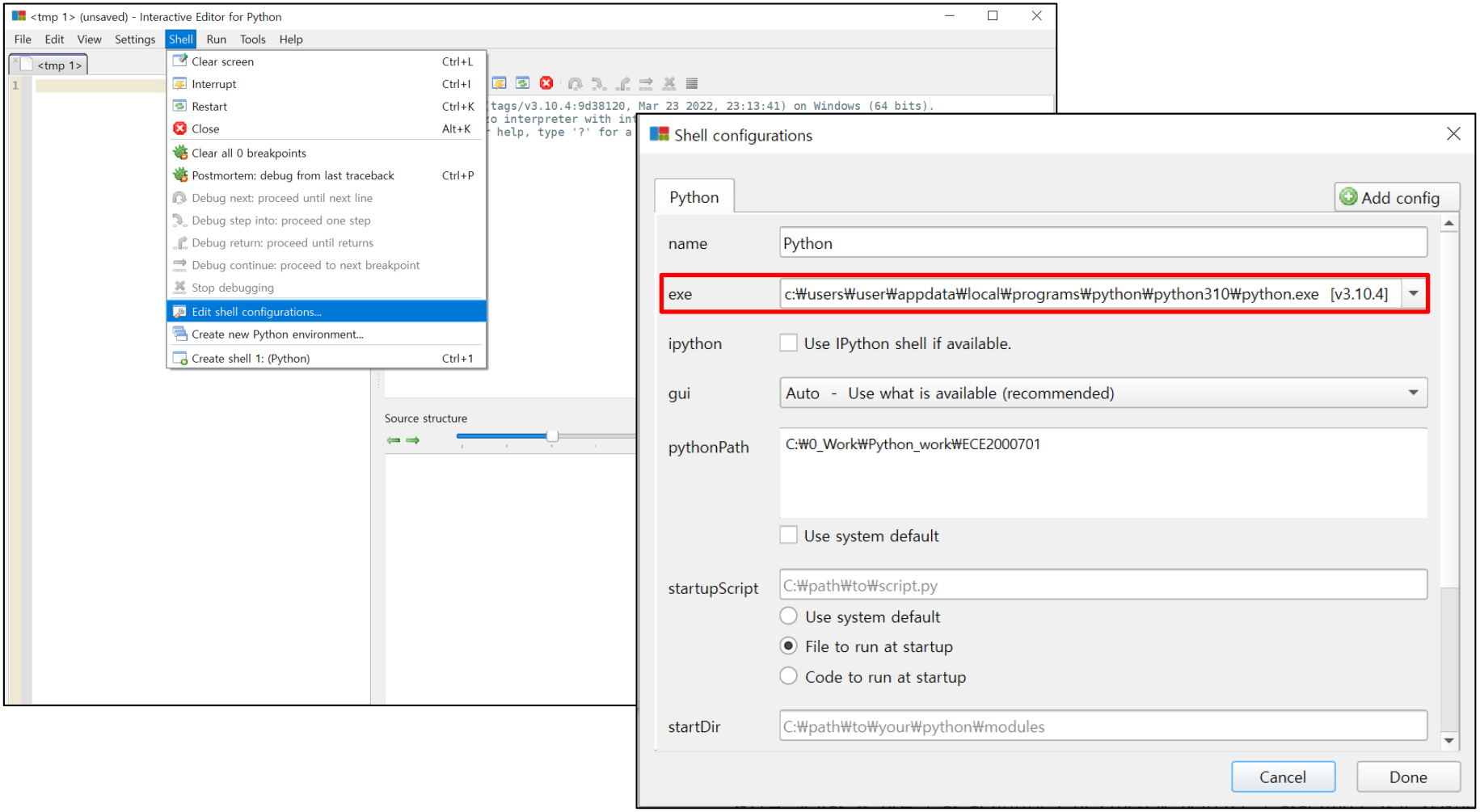
# Python 개발 환경

- **Pyzo**
  - ✓ 개발 및 debugging 을 체계적으로 관리할 수 있는 통합 환경(IDE) tool이 다양하게 제공
  - ✓ 본 실습에서는 Pyzo를 사용함 (https://pyzo.org/)



# Python 개발 환경

- 기본적으로 Python v3 이상 지원 가능
  - ✓ Shell -> Edit shell configurations…



# Python Package for EEBoard

- **Python class package**
  - ✓ Digilent 사의 Waveforms를 설치하면 EEboard를 제어하기 위한 SDK는 자동으로 설치가 되고, EEboard를 제어하려면 SDK를 이용하여 필요한 Python Class를 설계 해야함

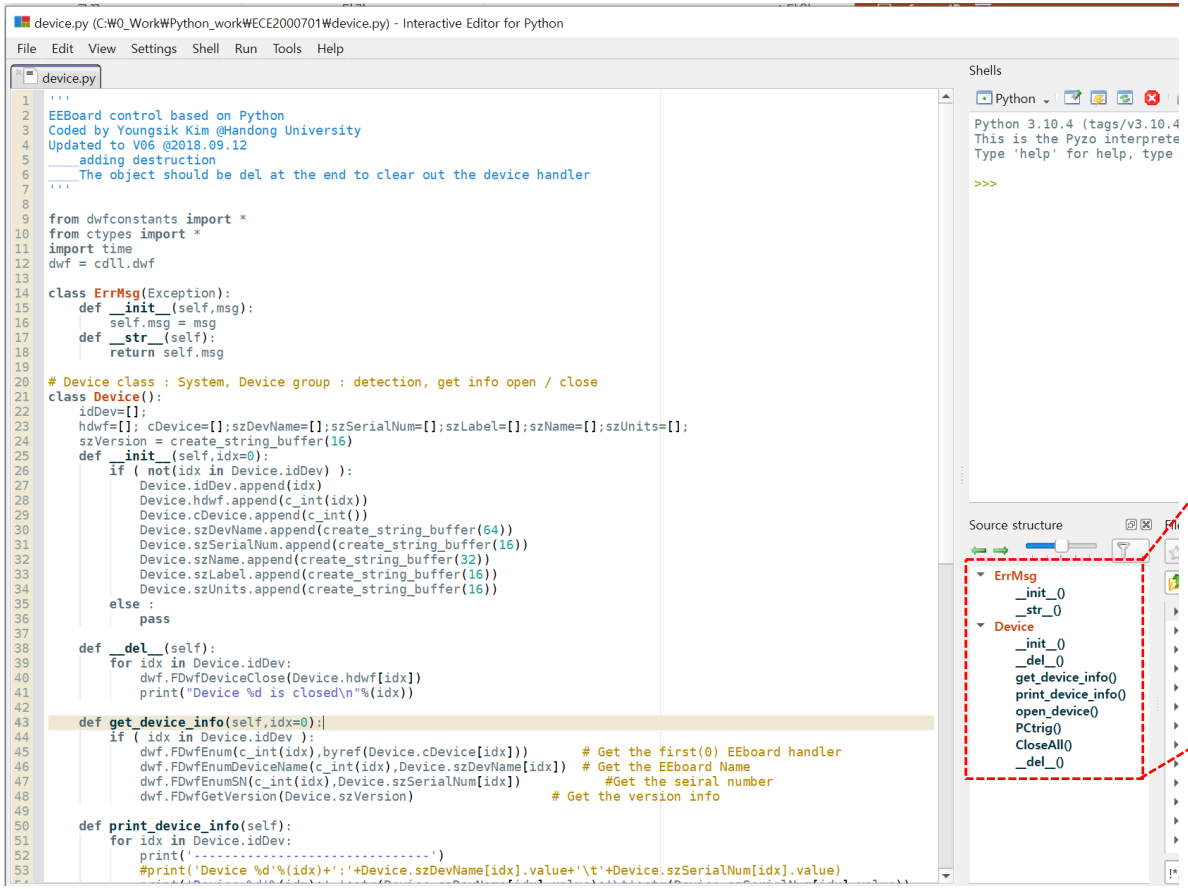
```
.
|-- Readme.txt   : this file
|-- eeboard
|   |-- __init__.py : package init file
|   |-- awg.py : AWG object
|   |-- device.py: Electronic Explorer board base class
|   |-- digitalio.py : digital static io object
|   |-- dwfconstants.py : constants provided from Digilentinc
|   |-- logic.py   : logic analyzer object
|   |-- pattern.py : digital pattern generation object
|   |-- power.py: Power supply and voltagemeter boject
|   `-- scope.py : analog scope object
`-- examples
    |-- 01_Device.py : check out the device connection.
    |-- 02_Power.py : Vref1, Vref2, VP+ and Voltage meters are tested
    |-- 03_AWG_Scope.py: AWG and Scope test
    |-- 04_DigitalIO.py : Static digital IO test
    |-- 05_Pattern_Logic03.py : 4bit counter and meausre with 8-bit 1000 samples with Single Acquisition Mode
    |-- 06_Pattern04.py D22 and D21 as output terminal, 1khz clock 50% duty output
    `-- 07_Pattern
```

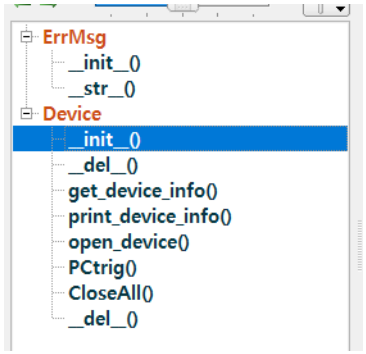
https://github.com/young1329/EEboard

  - ✓ eeboard: EEboard제어 클래스 정의 패키지
  - ✓ Example: 패키지를 시험하기 위한 test-bench 파일
  - ✓ 본 실습에서는 device.py와 power.py class만을 사용할 예정

# Python Package for EEBoard

- **Device class**
  - ✓ EEboard 제어를 위한 모든 클래스의 base class가 됨

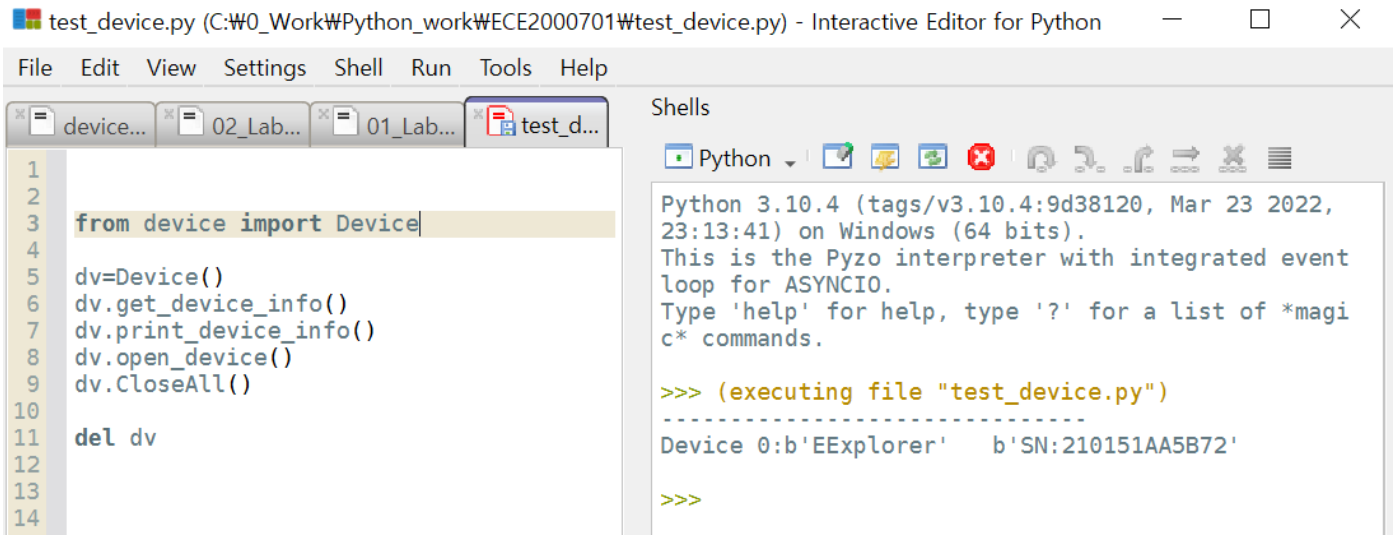



Device class의 구조

# Python Package for EEBoard

- **Example: Device class**
  - ✓ 해당 코드는 EEboard에서 Device 객체를 import한 후 dv 객체를 생성
  - ✓ 그리고 dv.get_device_info()로 EEboard의 정보를 가져와 dv.print_device_info() 메서드로 EEboard의 정보를 출력함



# Python Package for EEBoard

- **Power class**
  - ✓ Device 클래스를 상속 하여 다음을 제어
    - 전원공급 Vcc, VP+, VP- 포트
    - 기준 전압을 출력하는 Vref1, Vref2 포트
    - 전압을 측정하는 $V_{MTR1}$~$V_{MTR4}$ 포트

```
from device import *

channels = {'Vcc':0, 'VP+':1, 'VP-':2, 'Vref1':3, 'Vref2':4,
            'Vmtr1':5,'Vmtr2':6, 'Vmtr3':7, 'Vmtr4':8}
nodes = {'Enable':0, 'Voltage':1,'Current':2,'Power':3}

class Power(Device):
    def __init__(self,idx=0):
        self.idx = idx;
        self.nChannel = c_int()
        self.szChannel = create_string_buffer(32)
        self.szLabel = create_string_buffer(16)
        self.szNode = create_string_buffer(32)
        self.szUnits = create_string_buffer(16)
        #Current for power supply
        self.crntVP = c_double()
        self.crntVN = c_double()
        self.crntVCC = c_double()

        Device.__init__(self,idx)
        #Vmtr Variables
        self.Vmtr = [c_double(), c_double(), c_double(), c_double()]
        #print ('pwr idx is %d'%self.idx)
```


Power class의 구조

# Python Package for EEBoard

- **Example: Power class**

  ✓ 해당 코드는 Vref1, Vref2, VP+, Vcc 포트를 제어하여 전압을 출력함

01_Lab09_Power_example.py | device.py | power.py

```
    Vmtr1, Vmtr2, Vmtr3, Vmtr4 : measure and print the v
    You can check it by connecting Vmtr1 --> Vre1, Vmtr2
'''

from power import Power

import time

pwr=Power()

pwr.get_device_info()
pwr.print_device_info()

pwr.open_device()
pwr.reset_analogIO()
print('========================================\n')

#get the number of channels for AnalogIO
pwr.get_number_of_channels()
# get the number of node for channel 1
pwr.get_nodes_of_channels(1)
# figure out channel 1, and node 2
pwr.what_is_channel_node(1,2)
print('========================================\n')


# Configure VP+=5.0V with 50mA current
pwr.set_channel_voltage('Vref1',4.0)
pwr.set_channel_voltage('Vref2',-3.0)
pwr.set_channel_voltage('VP+',5.0)
pwr.set_channel_current('VP+',50e-3)
pwr.set_channel_voltage('Vcc',3.3)

pwr.enable_channel('Vref1')
pwr.enable_channel('Vref2')
pwr.enable_channel('VP+')
pwr.enable_channel('Vcc')

pwr.analogIO_ON()

time.sleep(1)

pwr.measure_vmtr()

pwr.analogIO_OFF()


print('Vmtr1=%.2f V\n'%(pwr.get_vmtr(0)))
print('Vmtr2=%.2f V\n'%(pwr.get_vmtr(1)))
print('Vmtr3=%.2f V\n'%(pwr.get_vmtr(2)))
print('Vmtr4=%.2f V'%(pwr.get_vmtr(3)))

del pwr
```

✓ EEboard에서 아래 같이 연결 하는 것이 필요

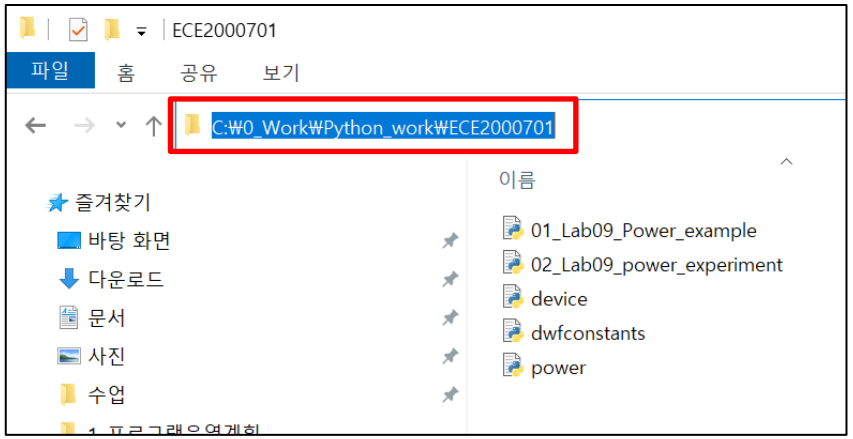
- Vref1 → $V_{MTR1}$
- Vref2 → $V_{MTR2}$
- VP+ → $V_{MTR3}$
- Vcc → $V_{MTR4}$

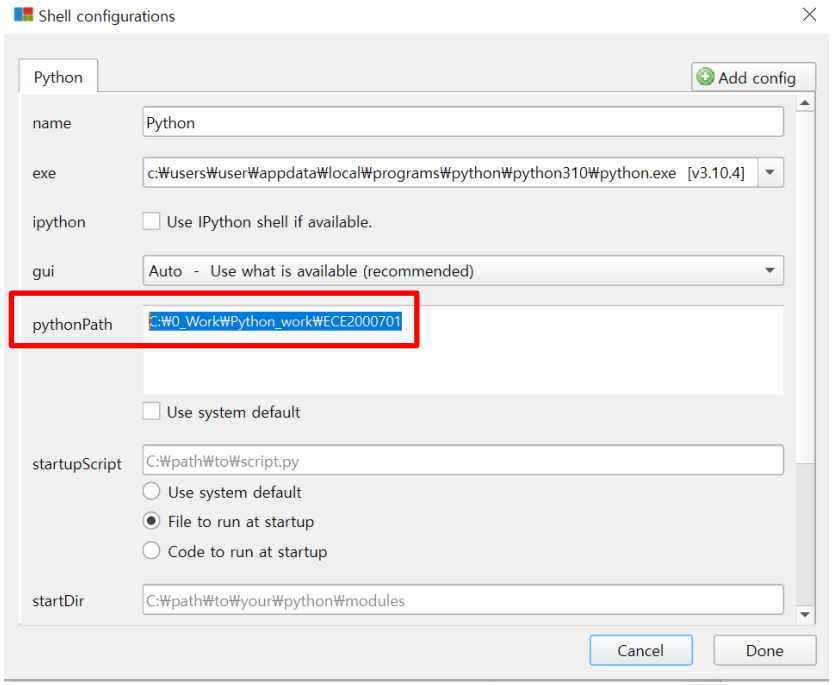
```
>>> (executing file "01_Lab09_Power_example.py")
------------------------------
Device 0:b'EExplorer'   b'SN:210151AA5B72'
========================================

Numberof Channel is 9

Number of nodes of the channel(1, b'Positive Supply', b'VP+') is 4

Channel(1): b'Positive Supply', b'VP+'

Node(2) : b'Current', b'A'

========================================

Vmtr1=4.01 V

Vmtr2=-3.09 V

Vmtr3=4.99 V

Vmtr4=3.33 V
```

실행 결과

# 실습 1

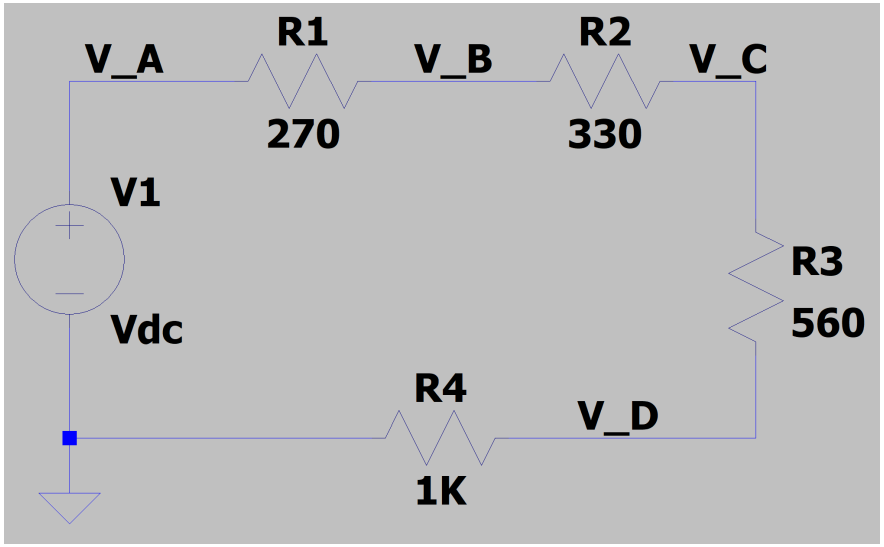
- **01_Lab09_Power_example.py**
  - ✓ 해당 python 코드를 그대로 실행하여 EEboard가 제어되는지 확인
- **[참고]**
  - ✓ Python 코딩을 수행하는 작업 폴더의 경로를 'pythonPath'에 설정 필요
    - Pyzo tool에서 Shell -> Edit shell configurations에서 확인



# 실습 2

- **다음과 같이 회로를 꾸미고 Python으로 EEboard를 제어**
  - ✓ 전압 소스로는 VP+를 사용하고, 4개의 저항으로 직렬 연결 회로 구성
  - ✓ 각 node의 이름을 V_A, V_B, V_C, V_D로 정의하고 Volt meter에 연결
    - V_A → $V_{MTR1}$, V_B → $V_{MTR2}$
    - V_C → $V_{MTR3}$, V_D → $V_{MTR4}$
  - ✓ VP+의 전압이 1.0V, 3.0V, 5.0V 각각 세가지 경우에서
    - 모든 node의 전압을 측정 및 이를 그래프로 표현하기



# 실습 2

02_Lab09_power_experiment.py

```python
'''
Lab is based on the Code by Youngsik Kim @ CSEE . HGU
Test the Power supply and Voltage meter function
'''


from power import Power
import time
import matplotlib.pyplot as plt

pwr=Power()

pwr.get_device_info()
pwr.print_device_info()

pwr.open_device()
pwr.reset_analogIO()
print('========================================\n')


# Configure VP+ with 50mA current limit
pwr.set_channel_current('VP+',50e-3)
pwr.enable_channel('VP+')
pwr.analogIO_ON()

# Vraible declaration
Vdc_list = [1.0, 3.0, 5.0]
V_A_list = []
V_B_list = []
V_C_list = []
V_D_list = []

# Experiment
print('*** Experiment****')
for vdc in Vdc_list:
    print('Vdc=%.2f (V)'%(vdc))
    pwr.set_channel_voltage('VP+',vdc)
    time.sleep(0.5)
    pwr.measure_vmtr()

    V_A_list.append(pwr.get_vmtr(0))
    V_B_list.append(pwr.get_vmtr(1))
    V_C_list.append(pwr.get_vmtr(2))
    V_D_list.append(pwr.get_vmtr(3))

pwr.analogIO_OFF()
print('========================================\n')
print('*** Finished****')
del pwr

# Data representation
plt.plot(Vdc_list, V_A_list, 'o--', Vdc_list, V_B_list, 'o--', Vdc_list, V_C_list, 'o--', Vdc_list, V_D_lis
plt.xlim([1,5])
plt.ylim([0,5])
plt.xlabel('Vdc(V)')
plt.ylabel('Node voltage(V)')
plt.legend(('V_A', 'V_B', 'V_C', 'V_D'))
plt.grid()
plt.show()
```

<실행 결과>

```
>>> (executing file "02_Lab09_power_experiment.py")
------------------------------
Device 0:b'EExplorer'   b'SN:210151AA5B72'
============================================

*** Experiment****
Vdc=1.00 (V)
Vdc=3.00 (V)
Vdc=5.00 (V)
============================================

*** Finished****
```