https://www.halvorsen.blog

DAQ with Python

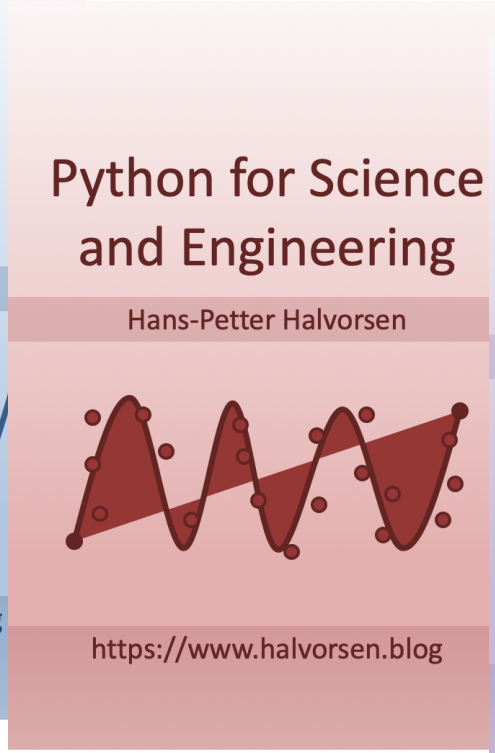
Exemplified by reading Temperature Data using NI USB TC-01 Thermocouple

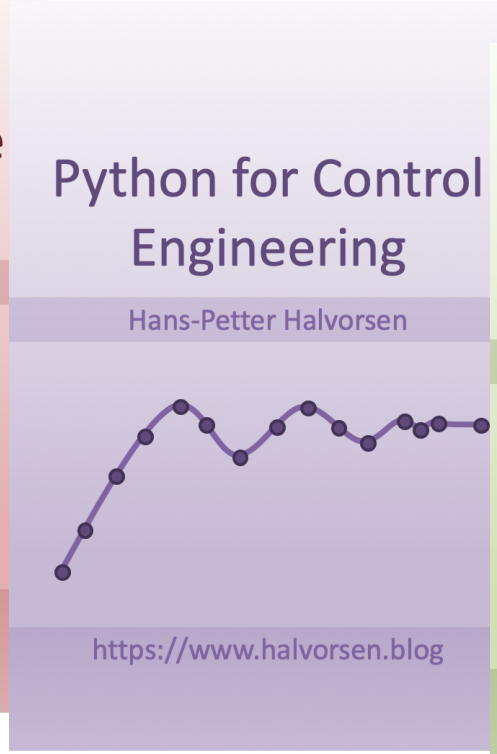
Hans-Petter Halvorsen

Free Textbook with lots of Practical Examples

https://www.halvorsen.blog/documents/programming/python/

Additional Python Resources





https://www.halvorsen.blog/documents/programming/python/

Contents

- How can we use NI Hardware with Python?
- What is DAQ?
- TC-01 Thermocouple Device
- DAQmx
- nidaqmx Python API
- Python Examples
  - Logging Data, Plotting Data, Save Data to File

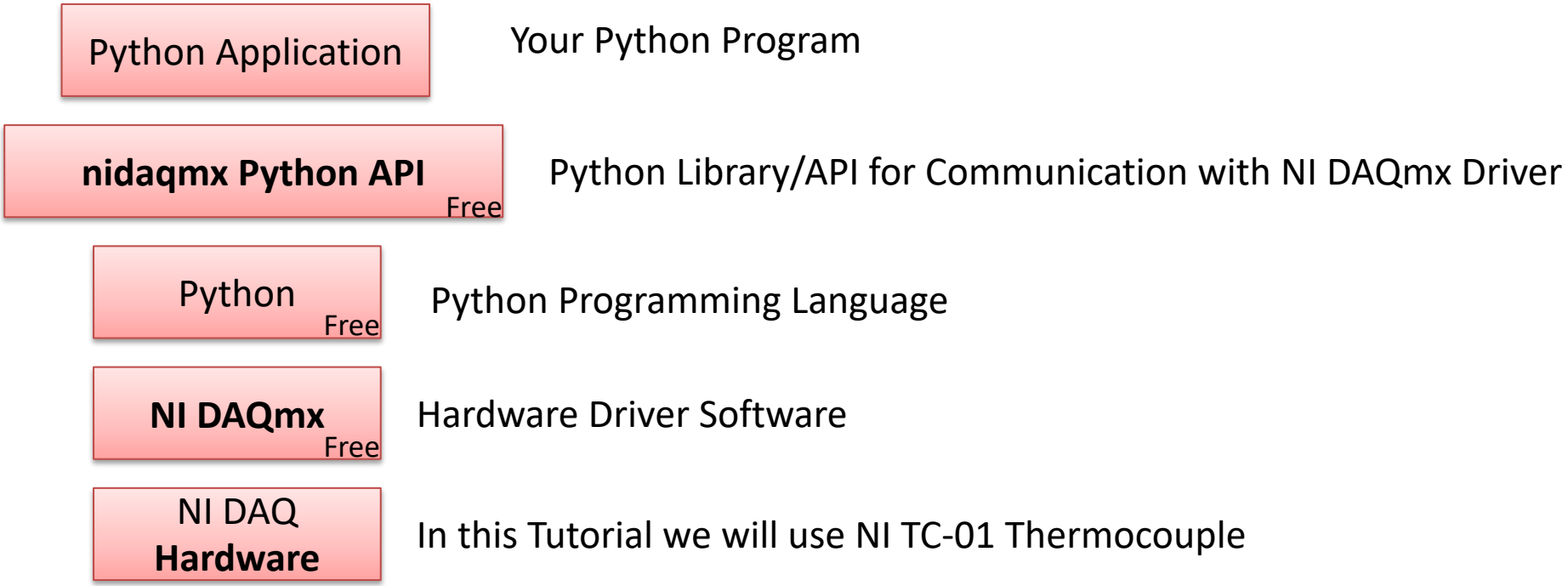
Note! The Python Examples provided will work for all NI-DAQ Devices using the NI-DAQmx Driver, which are several hundreds different types. We will use the NI TC-01 Thermocouple DAQ Device as an Example. The basic DAQ concepts can also be applied to other types of hardware from other vendors.

How can we use NI Hardware with Python?

- NI is a company that manufacture and sell both Hardware and Software
- The are most famous for their LabVIEW software
- LabVIEW is popular Graphical Programming Language
- Typically you use LabVIEW in combination with NI DAQ Hardware, but the NI-DAQmx can also be used from C, C#, Python, etc.
- Control NI DAQ Device with Python and NI DAQmx
  - https://knowledge.ni.com/KnowledgeArticleDetails?id=kA00Z000000P8o0SAC

NI DAQ Device with Python

How to use a NI DAQ Device with Python



LabVIEW

- In this Tutorial we will use Python and not LabVIEW
- But if you want to learn more about LabVIEW, you may take a look at my LabVIEW resources:
- https://halvorsen.blog/documents/programming/labview/labview.php

NI DAQ Hardware

Some Examples

TC-01 Thermocouple

myDAQ

NI-DAQmx
Hardware Driver

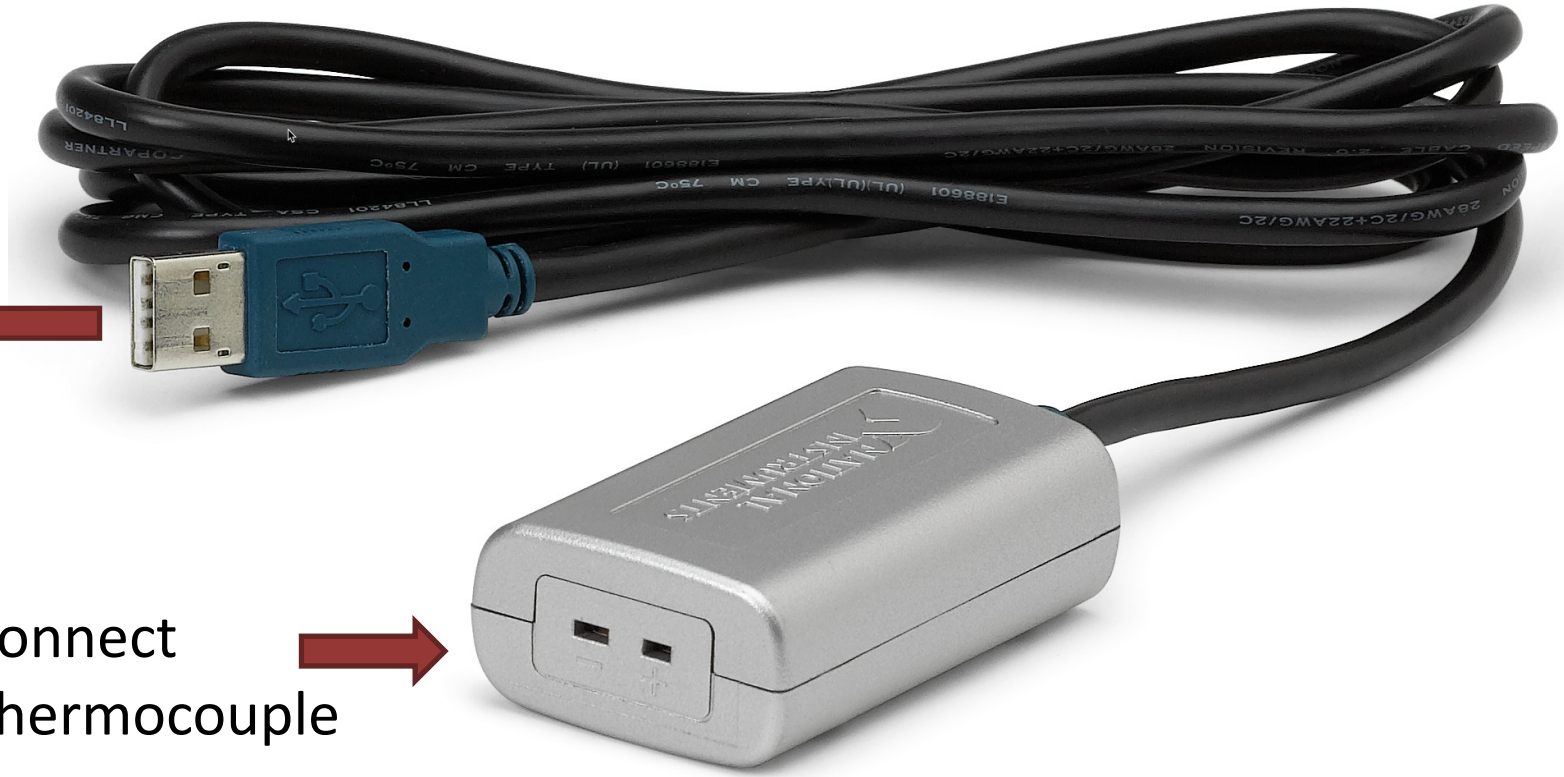
USB-6001

USB-6008

cDAQ

Note! The Python Examples provided will work for all NI-DAQ Devices using the NI-DAQmx Driver, which is several hundreds different types

NI USB TC-01 Thermocouple



https://www.ni.com/en-no/support/model.usb-tc01.html

NI USB TC-01 Thermocouple

J-Type Exposed-Junction Thermocouple

J-Type Grounded Probe Thermocouple

https://www.ni.com/en-no/support/model.usb-tc01.html

NI USB TC-01 Thermocouple

- USB TC-01 Thermocouple is a DAQ Hardware manufactured by NI
  www.ni.com
- It measures Temperature using the Thermocouple principle
- The USB-TC01 is compatible with J, K, R, S, T, N, E, and B thermocouples
- https://www.ni.com/en-no/support/model.usb-tc01.html

Thermocouple

- A Thermocouple is a sensor used to measure temperature.
- Thermocouples consist of two wire legs made from different metals.
- The wires legs are welded together at one end, creating a junction.
- This junction is where the temperature is measured.
- When the junction experiences a change in temperature, a voltage is created.
- The voltage can then be interpreted using thermocouple reference tables to calculate the temperature

https://www.thermocoupleinfo.com

Data Acquisition (DAQ)

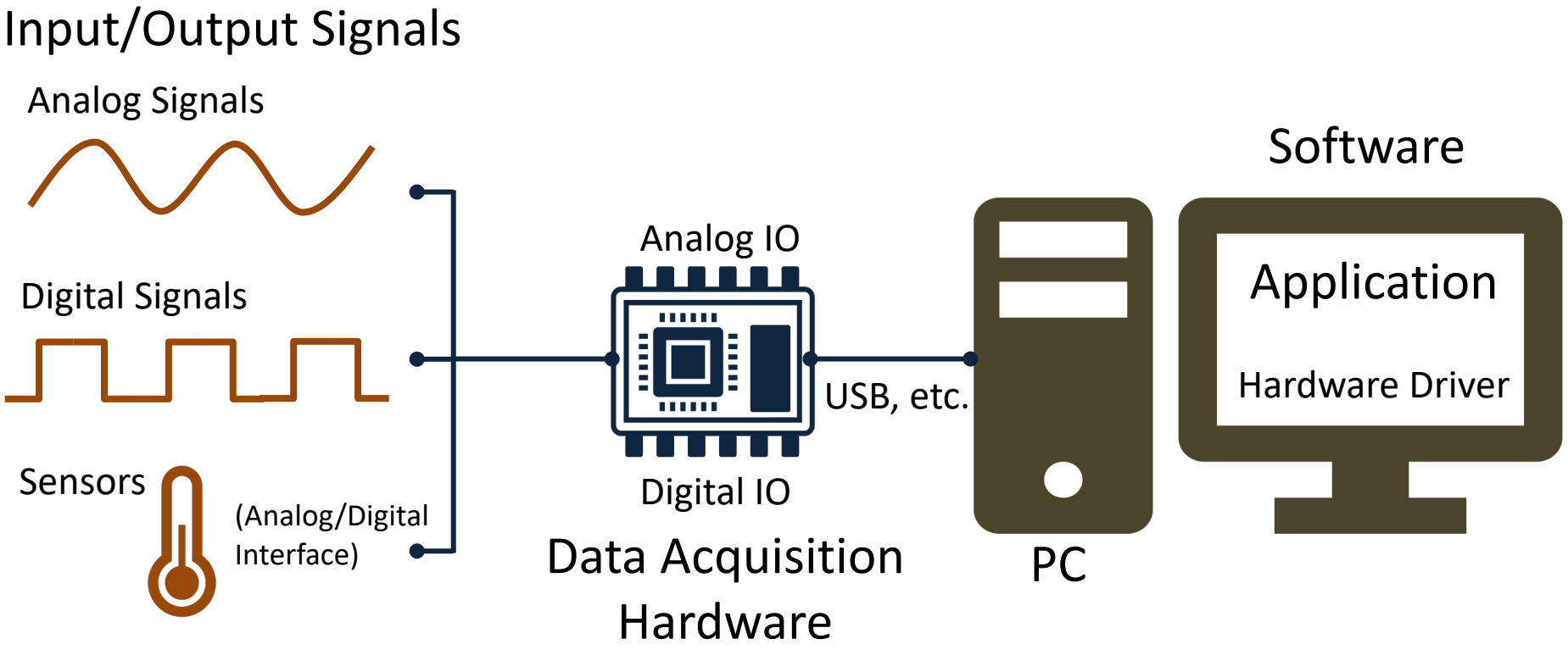
- To read sensor data you typically need a DAQ (Data Acquisition) device connected to you PC
- You can also use devices like Arduino , Raspberry Pi, etc.
- In all cases you will typically need to install a driver from the vendor of the DAQ device or the sensor you are using

DAQ System

A DAQ System consists of 4 parts:

- Physical input/output signals, sensors
- DAQ device/hardware
- Driver software
- Your software application (Application Software) - in this case your Python application

DAQ System

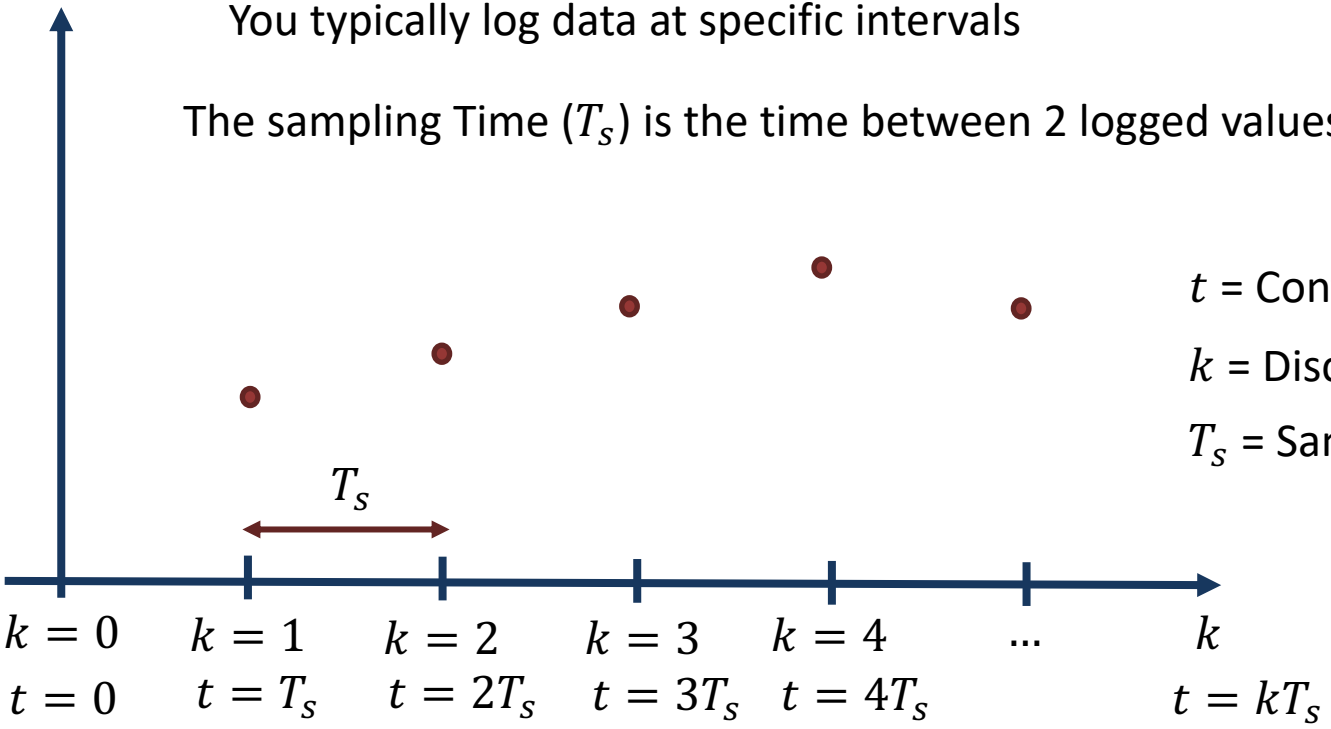

Input/Output Signals
Analog Signals
Digital Signals
Sensors
(Analog/Digital Interface)
Analog IO
Digital IO
Data Acquisition Hardware
USB, etc.
PC
Software
Application
Hardware Driver

Digital Signals

A computer can only deal with discrete signals

You typically log data at specific intervals

The sampling Time (T_s) is the time between 2 logged values



t = Continuous Time

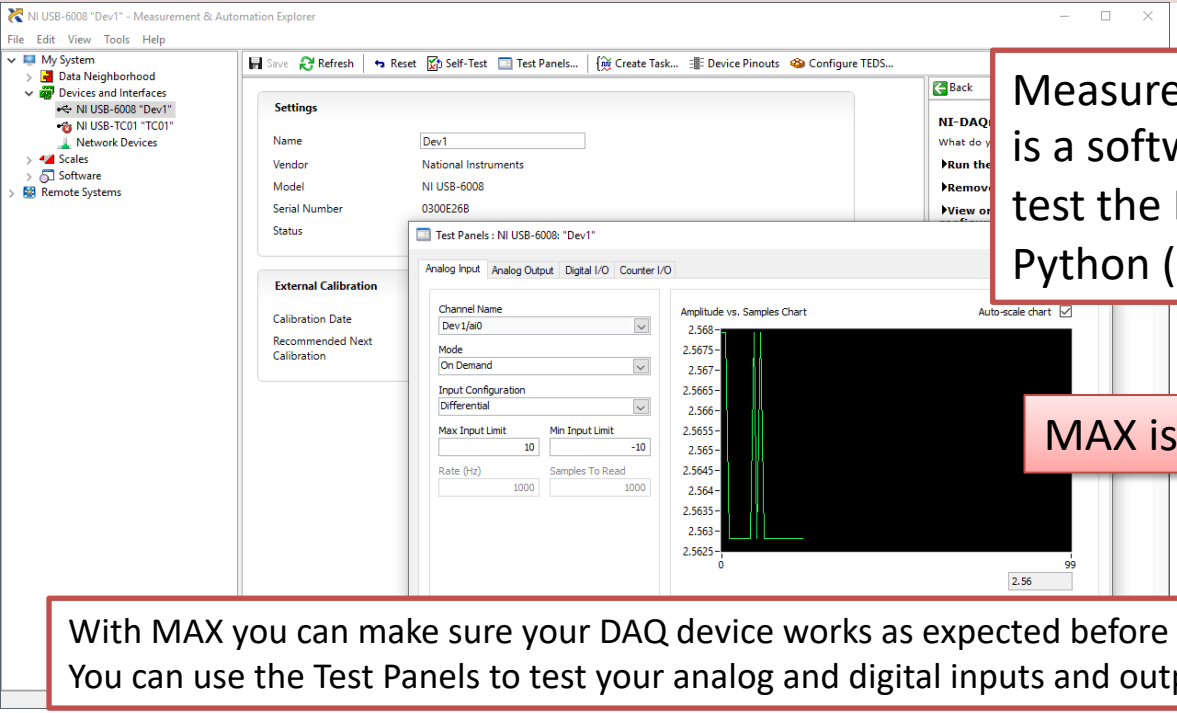
k = Discrete Time

T_s = Sampling Time

NI-DAQmx

- NI-DAQmx is the software you use to communicate with and control your NI data acquisition (DAQ) device.
- NI-DAQmx supports only the Windows operating system.
- Typically you use LabVIEW in combination with NI DAQ Hardware, but the NI-DAQmx can also be used from C, C#, Python, etc.
- The NI-DAQmx Driver is Free!
- Visit the ni.com/downloads to download the latest version of NI-DAQmx

Measurement & Automation Explorer (MAX)



Measurement & Automation Explorer (MAX) is a software you can use to configure and test the DAQ device before you use it in Python (or other programming languages).

MAX is included with NI-DAQmx software

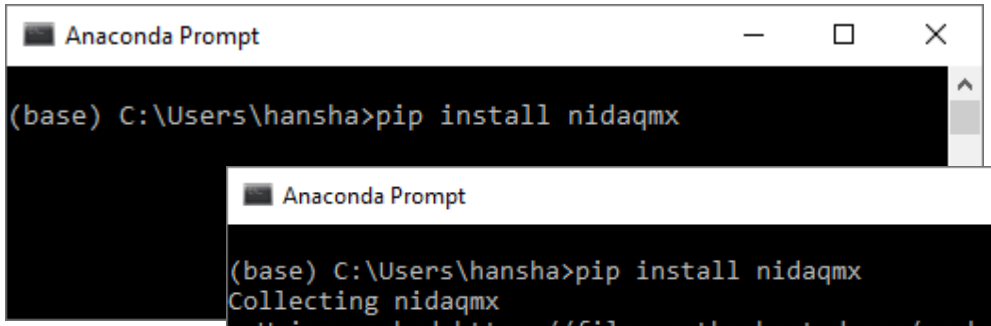
With MAX you can make sure your DAQ device works as expected before you start using it in your Python program. You can use the Test Panels to test your analog and digital inputs and outputs channels.

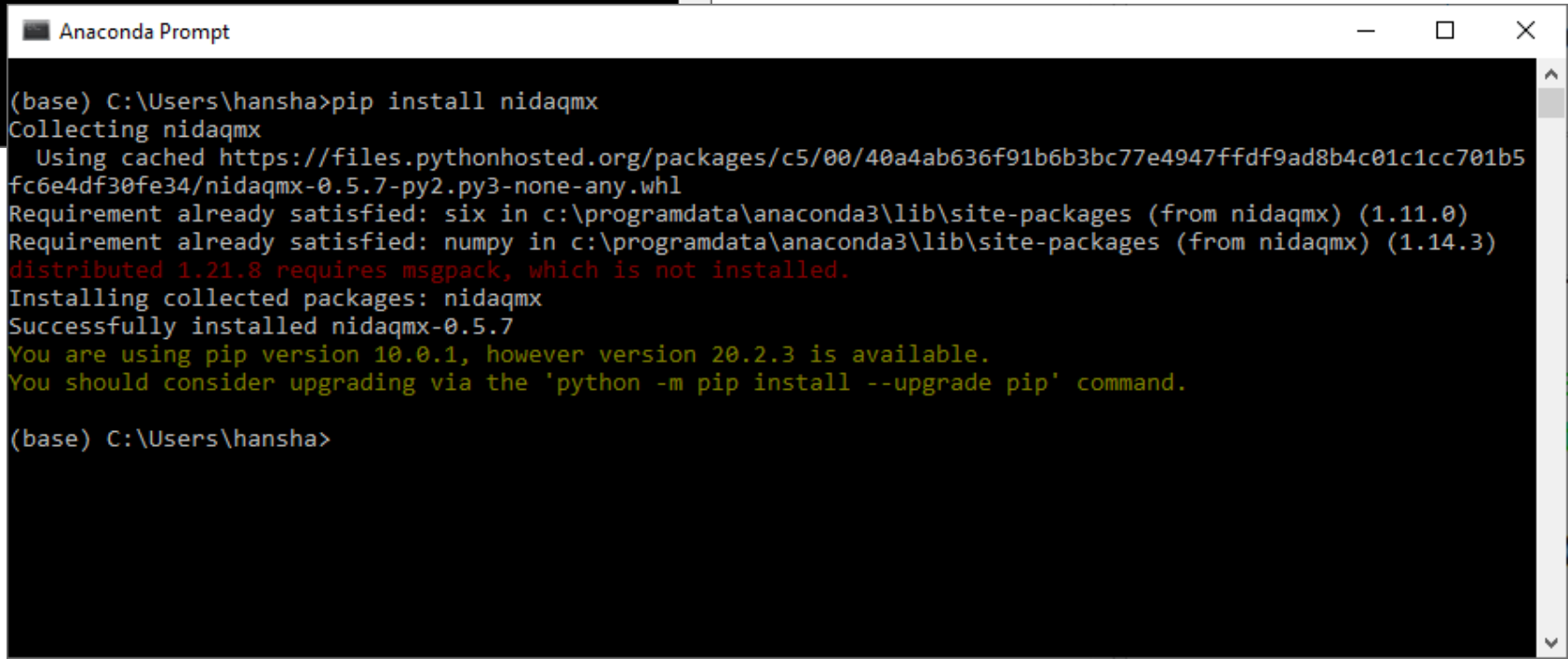
nidaqmx Python API

- Python Library/API for Communication with NI DAQmx Driver
- Running **nidaqmx** requires NI-DAQmx or NI-DAQmx Runtime
- Visit the ni.com/downloads to download the latest version of NI-DAQmx
- nidaqmx can be installed with **pip**:
  `pip install nidaqmx`
- https://github.com/ni/nidaqmx-python

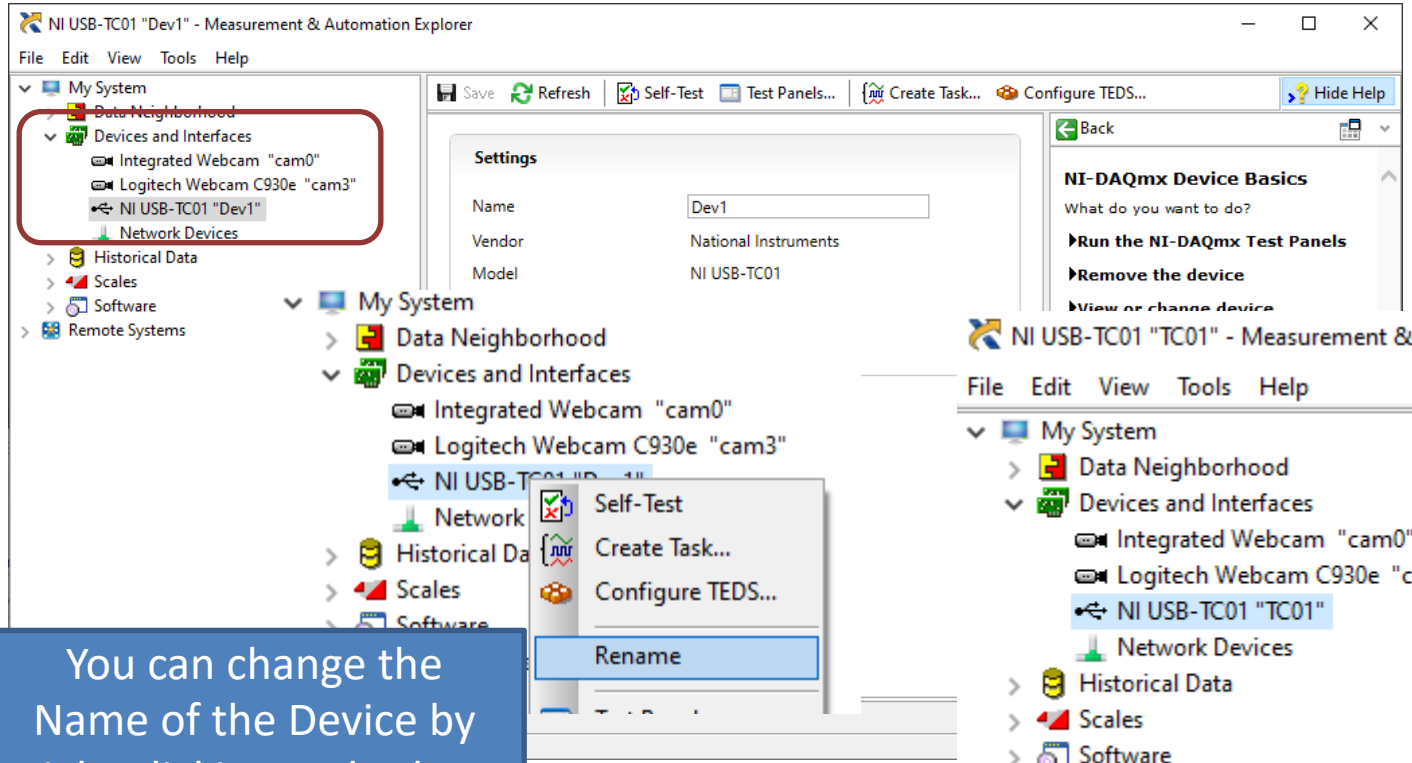
nidaqmx Python Package

Installation using PIP

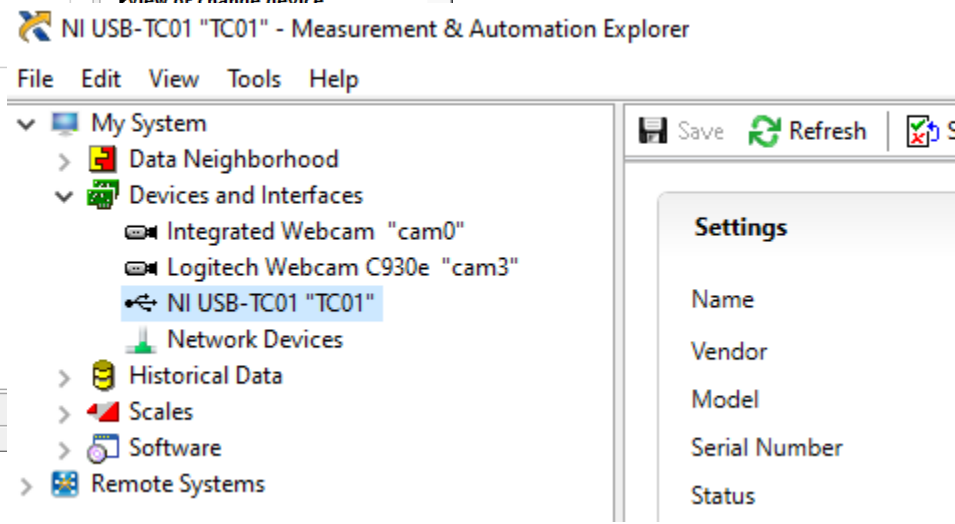




nidaqmx Python Package



MAX



You can change the Name of the Device by right-clicking and select “Rename”

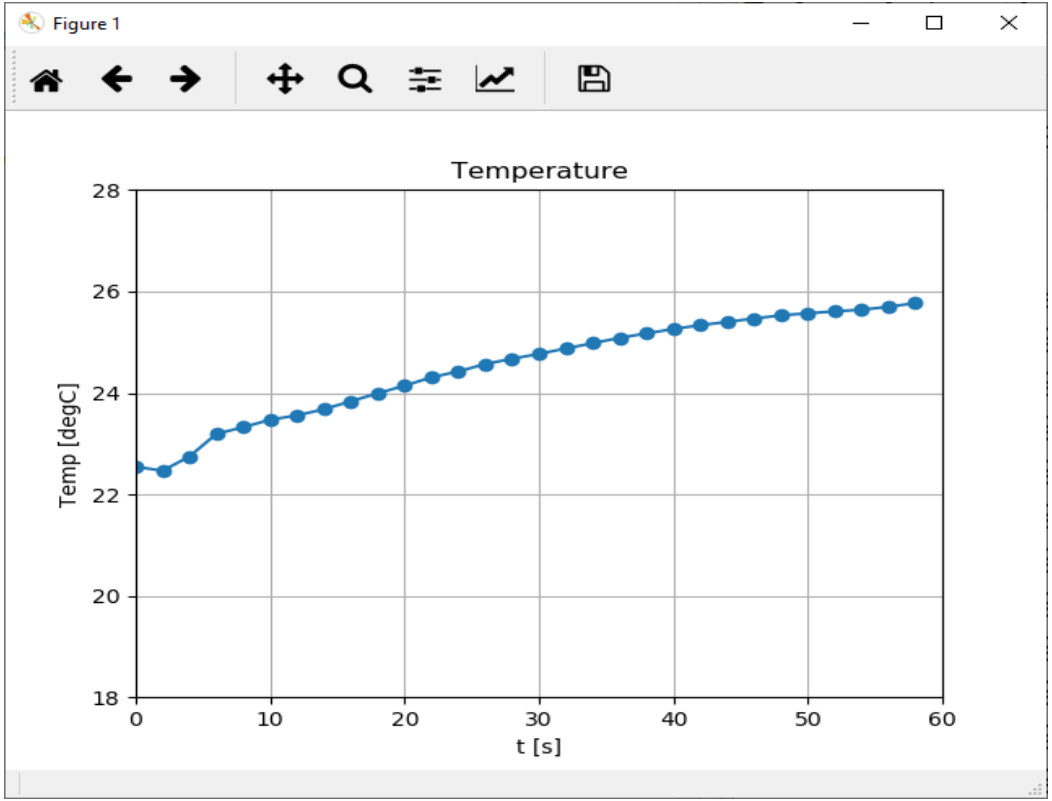
Basic Python Example

```
import nidaqmx

task = nidaqmx.Task()

task.ai_channels.add_ai_thrmcpl_chan("TC01/ai0")

task.start()

value = task.read()

print(round(value,1))

task.stop()
task.close()
```

For Loop

```
import nidaqmx
import time

# Initialize DAQ Device
task = nidaqmx.Task()
task.ai_channels.add_ai_thrmcpl_chan("TC01/ai0")
task.start()

Ts = 5 # Sampling Time

N = 60
for k in range(N):
    value = task.read()
    print("T =", round(value,1), "[degC]")
    time.sleep(Ts)

# Terminate DAQ Device
task.stop()
task.close()
```

Plotting




```
import nidaqmx
import time
import numpy as np
import matplotlib.pyplot as plt

# Initialize Logging
Tstop = 60 # Logging Time [seconds]
Ts = 2 # Sampling Time [seconds]
N = int(Tstop/Ts)
data = []

# Initialize DAQ Device
task = nidaqmx.Task()
task.ai_channels.add_ai_thrmcpl_chan("TC01/ai0")
task.start()

#Logging Temperature Data from DAQ Device
for k in range(N):
    value = task.read()
    print("T =", round(value,1), "[degC]")
    data.append(value)
    time.sleep(Ts)

# Terminate DAQ Device
task.stop()
task.close()

# Plotting
t = np.arange(0,Tstop,Ts)
plt.plot(t,data, "-o")
plt.title('Temperature')
plt.xlabel('t [s]')
plt.ylabel('Temp [degC]')
plt.grid()
Tmin = 18; Tmax = 28
plt.axis([0, Tstop, Tmin, Tmax])
plt.show()
```

Log to File

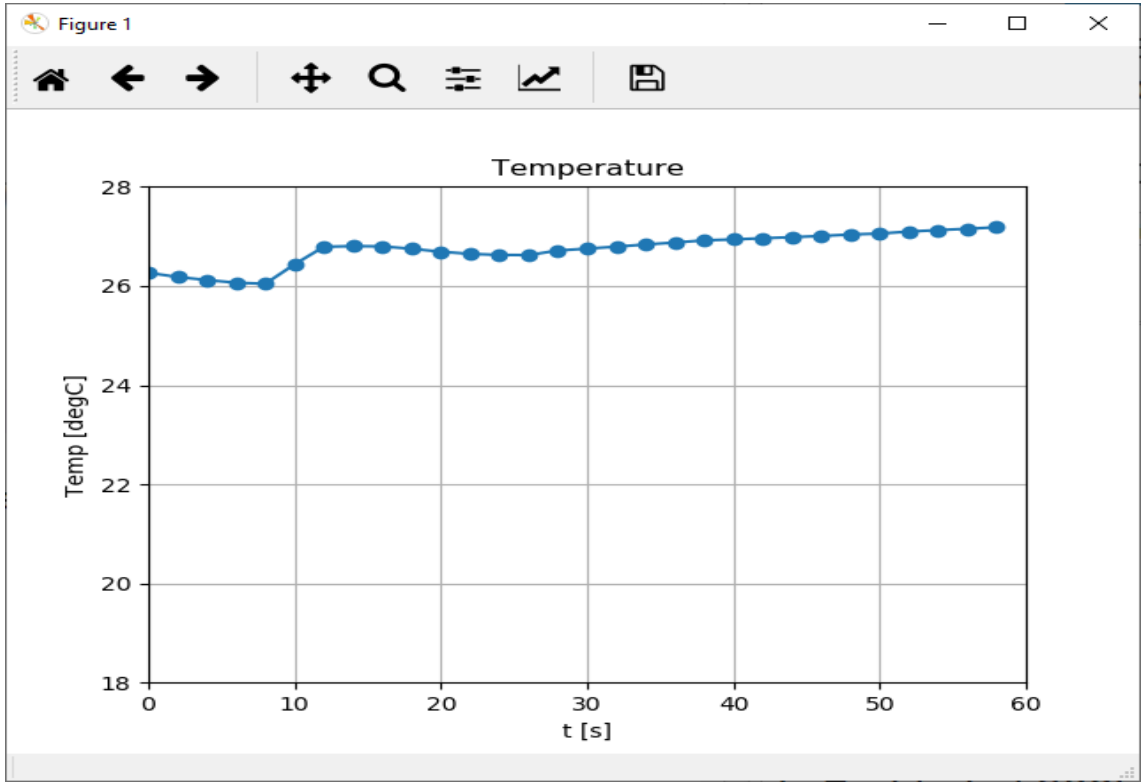
Python Code:

```
import nidaqmx
import time
import numpy as np
import matplotlib.pyplot as plt

# Initialize Logging
Tstop = 10 # Logging Time [seconds]
Ts = 2 # Sampling Time [seconds]
N = int(Tstop/Ts)
data = []

# Initialize DAQ Device
task = nidaqmx.Task()
task.ai_channels.add_ai_thrmcpl_chan("TC01/ai
0")
task.start()

# Open File
file = open("tempdata.txt", "w")

# Write Data Function
def writefiledata(t, x):
    time = str(t)
    value = str(round(x, 2))
    file.write(time + "\t" + value)
    file.write("\n")
```

Cont.

```
# Logging Temperature Data from DAQ Device
for k in range(N):
    value = task.read()
    print("T =", round(value,1), "[degC]")
    data.append(value)
    time.sleep(Ts)
    writefiledata(k*Ts, value)

# Terminate DAQ Device
task.stop()
task.close()

# Close File
file.close()

# Plotting
t = np.arange(0,Tstop,Ts)
plt.plot(t,data, "-o")
plt.title('Temperature')
plt.xlabel('t [s]')
plt.ylabel('Temp [degC]')
plt.grid()
Tmin = 18; Tmax = 28
plt.axis([0, Tstop, Tmin, Tmax])
plt.show()
```

Log to File



tempdata.txt - Notepad

File Edit Format View Help

0	26.27
2	26.18
4	26.12
6	26.07
8	26.04
10	26.44
12	26.79
14	26.8
16	26.8
18	26.75
20	26.69
22	26.65
24	26.63
26	26.62
28	26.71
30	26.75
32	26.79
34	26.84
36	26.88
38	26.92
40	26.94
42	26.96
44	26.98
46	27.01
48	27.04
50	27.06
52	27.1
54	27.13
56	27.15
58	27.18

Ln 100% Windows (CRLF) UTF-8

Real-time Plotting

Python Code:

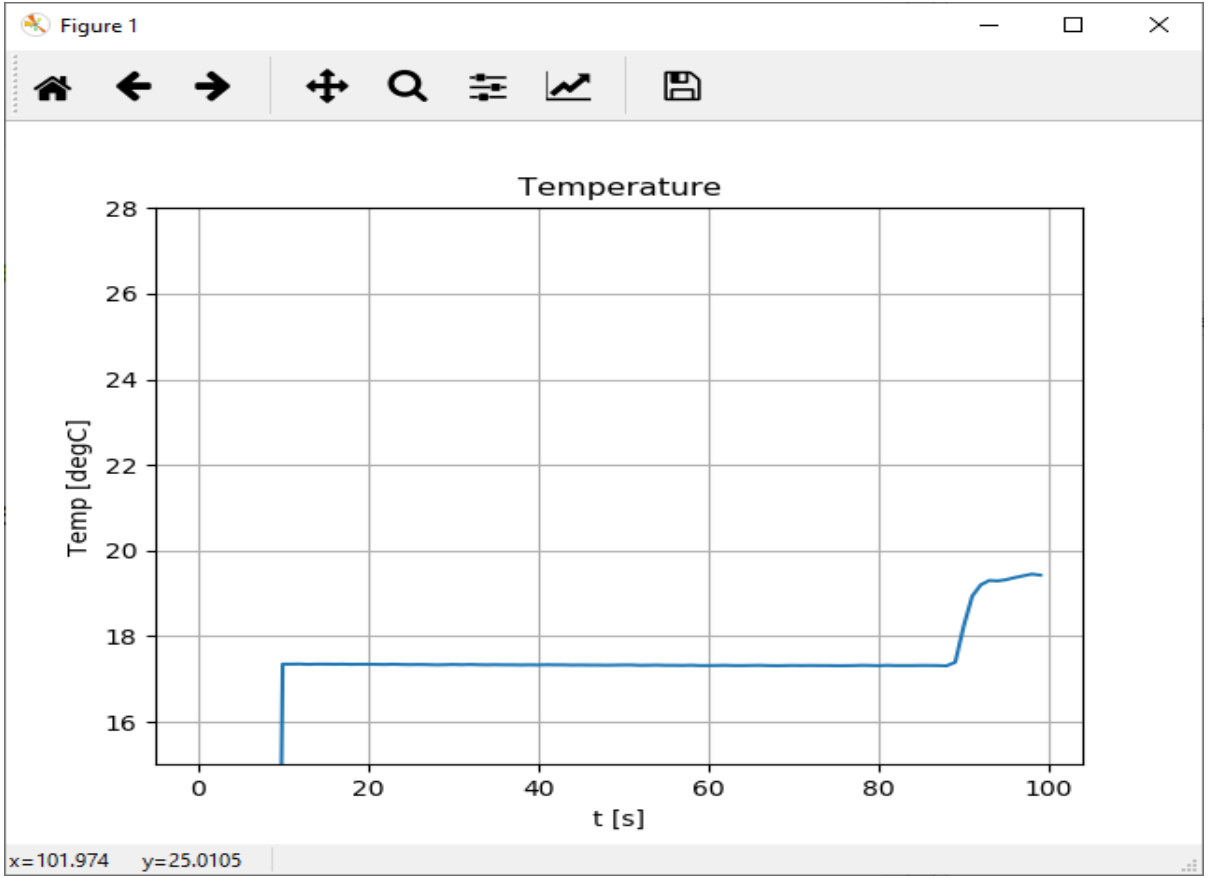
```
import nidaqmx
import time
import matplotlib.pyplot as plt
import matplotlib.animation as animation

# Read from DAQ Device
def readdaq():
    task = nidaqmx.Task()
    task.ai_channels.add_ai_thrmcpl_chan("TC01/ai0")
    task.start()
    value = task.read()
    task.stop()
    task.close()
    return value

# Write Data Function
def writefiledata(t, x):
    # Open File
    file = open("tempdata.txt", "a")

    # Write Data
    time = str(t)
    value = str(round(x, 2))
    file.write(time + "\t" + value)
    file.write("\n")

    # Close File
    file.close()

# Initialize Logging
Ts = 1 # Sampling Time [seconds]
N = 100
k = 1
x_len = N  # Number of points to display
Tmin = 15; Tmax = 28
y_range = [Tmin, Tmax]  # Range of possible Y values to display
data = []

# Create figure for plotting
fig = plt.figure()
ax = fig.add_subplot(1, 1, 1)
xs = list(range(0, N))
ys = [0] * x_len
ax.set_ylim(y_range)
```

This Code is getting advanced, so I will not go into details. Lets just run the code and observe the results

Cont.

```
# Create a blank line. We will update the line in animate
line, = ax.plot(xs, ys)

# Configure Plot
plt.title('Temperature')
plt.xlabel('t [s]')
plt.ylabel('Temp [degC]')
plt.grid()

#Logging Temperature Data from DAQ Device
def logging(i, ys):
    value = readdaq()
    print("T =", round(value,1), "[degC]")
    data.append(value)
    time.sleep(Ts)
    global k
    k = k + 1
    writefiledata(k*Ts, value)

    # Add y to list
    ys.append(value)
    # Limit y list to set number of items
    ys = ys[-x_len:]
    # Update line with new Y values
    line.set_ydata(ys)
    return line,

ani = animation.FuncAnimation(fig,
    logging,
    fargs=(ys,),
    interval=100,
    blit=True)

plt.show()
```

Real-time Plotting



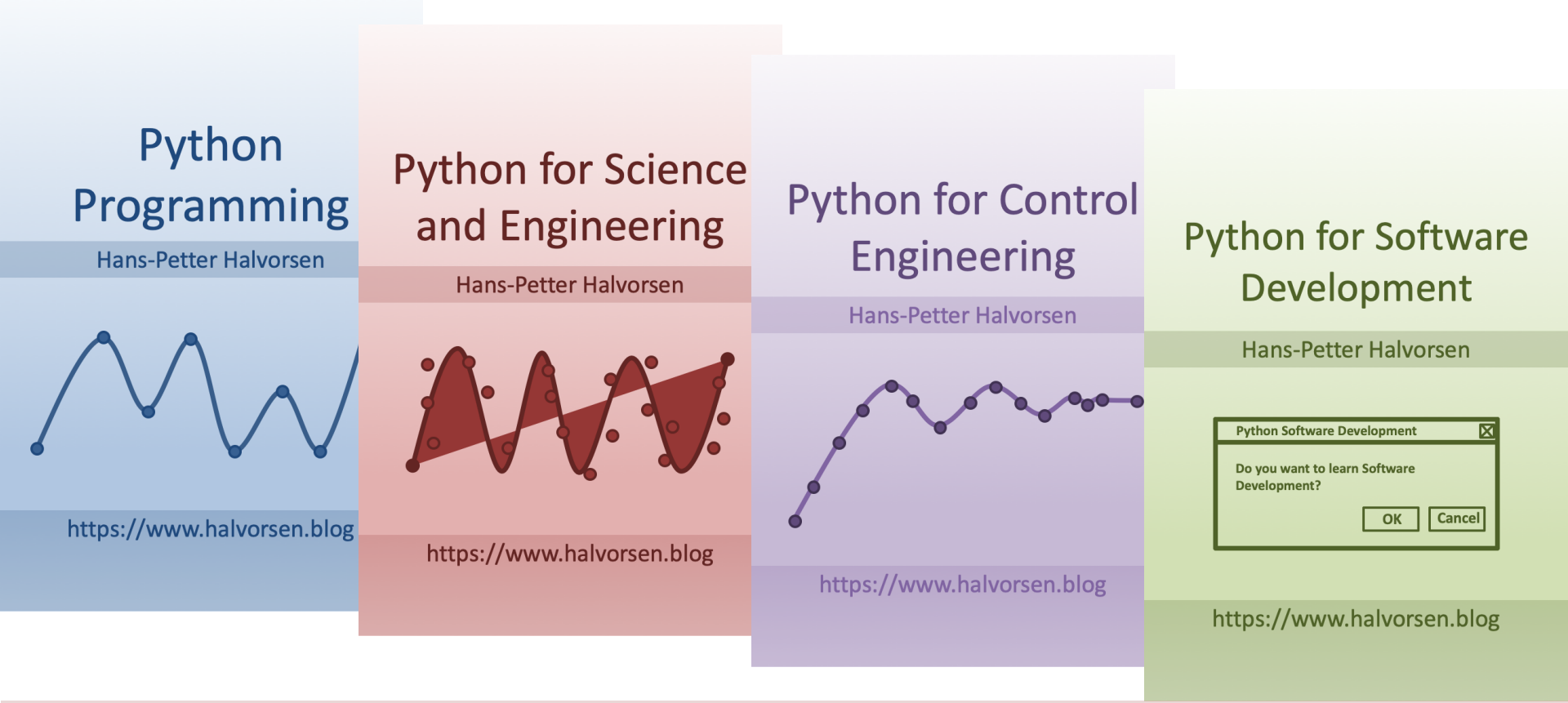
The Plot is updated in Real-time.

You stop the Logging when you close the Figure

DAQ

- I have made lots of DAQ resources for other Topics and Programming Language
- Here you find more information, resources, videos and examples regarding DAQ:
- https://www.halvorsen.blog/documents/technology/daq

Additional Python Resources



https://www.halvorsen.blog/documents/programming/python/

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: https://www.halvorsen.blog