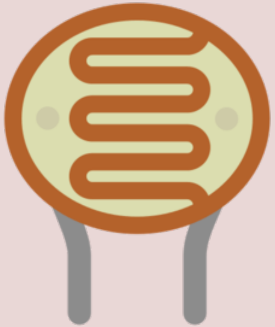
https://www.halvorsen.blog



Light Sensors with Python

Exemplified by using NI USB-6008 I/O Module

Hans-Petter Halvorsen

Free Textbook with lots of Practical Examples

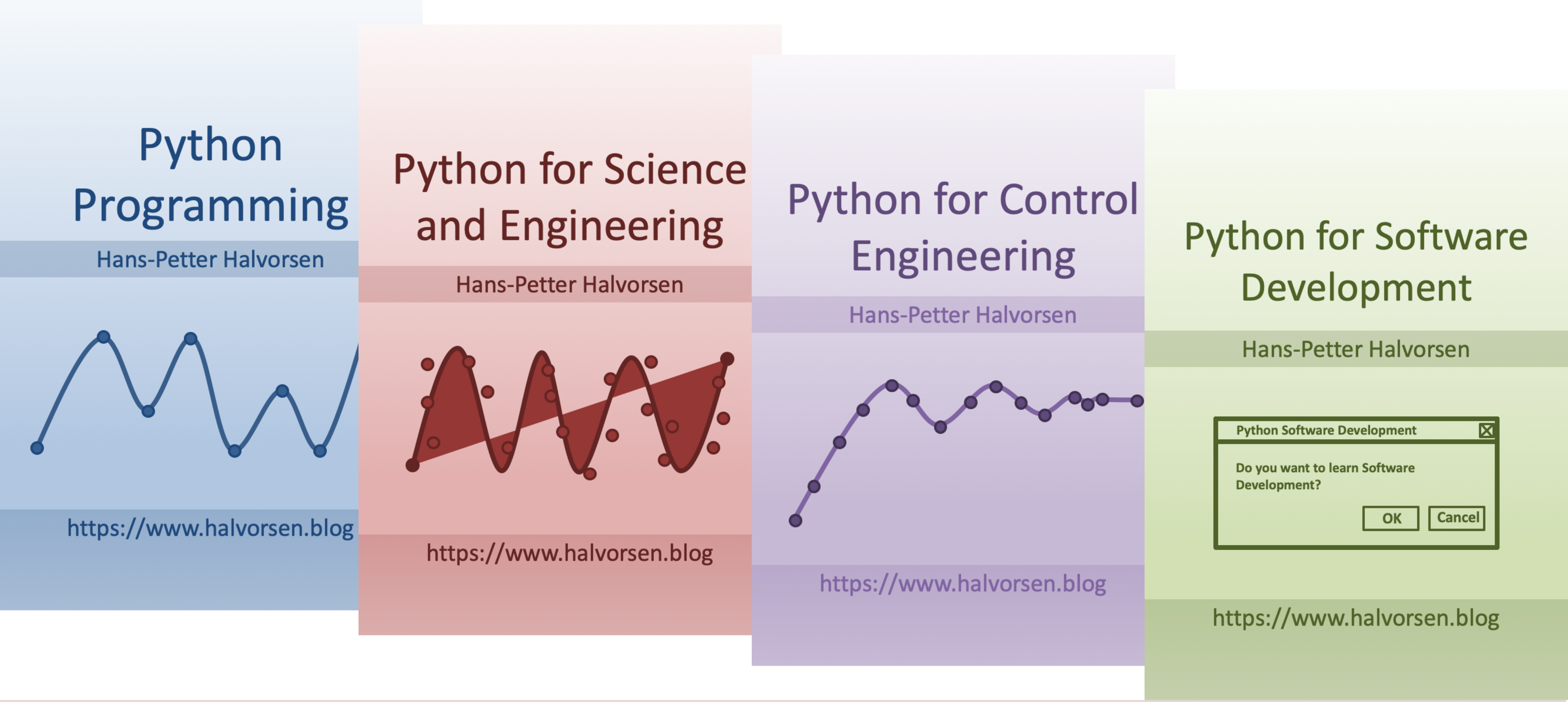
Python for Science and Engineering

Hans-Petter Halvorsen

https://www.halvorsen.blog

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

Additional Python Resources



Python Programming

Hans-Petter Halvorsen

https://www.halvorsen.blog

Python for Science and Engineering

Hans-Petter Halvorsen

https://www.halvorsen.blog

Python for Control Engineering

Hans-Petter Halvorsen

https://www.halvorsen.blog

Python for Software Development

Hans-Petter Halvorsen

Python Software Development ☒

Do you want to learn Software Development?

OK Cancel

https://www.halvorsen.blog

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

Contents

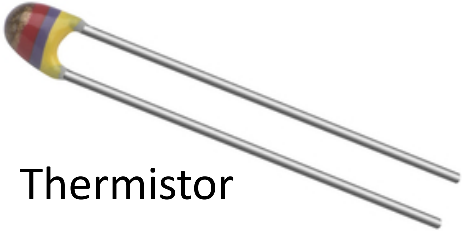
- DAQ and I/O Modules
- NI-DAQ
- Light Sensors
  - Light sensor, Photocell (Photo resistor), LDR (light-dependent resistor)
- Python Examples

Note! The Python Examples provided will work for all NI-DAQ Devices using the NI-DAQmx Driver, which is several hundreds different types. We will use the NI USB-6008 DAQ Device or I/O Module as an Example

Equipment

USB-6008 (or similar DAQ Device)

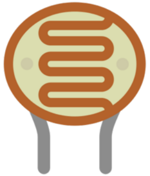
Push Button



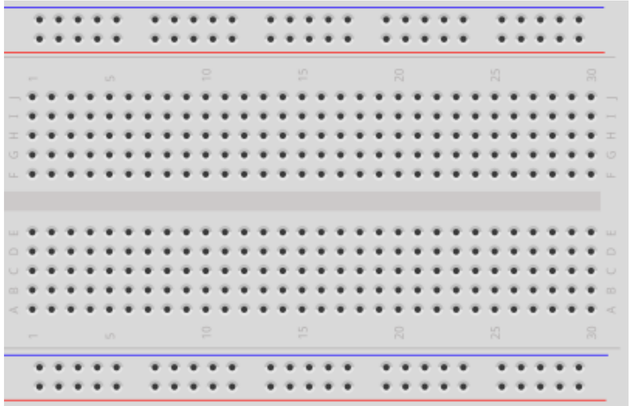
Thermistor

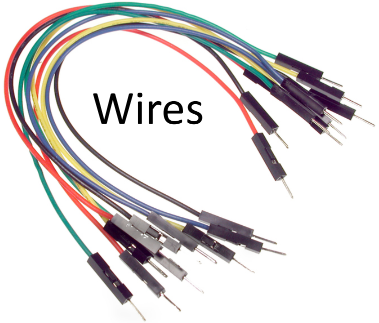
TMP36

Light Sensor

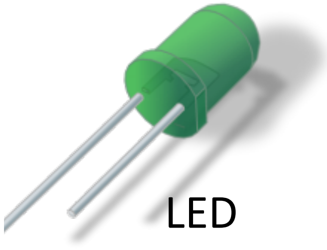


Breadboard

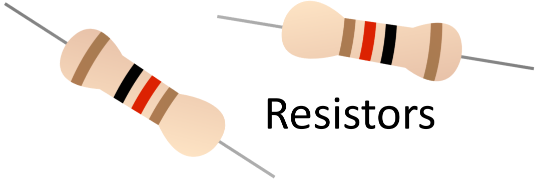




Wires



LED

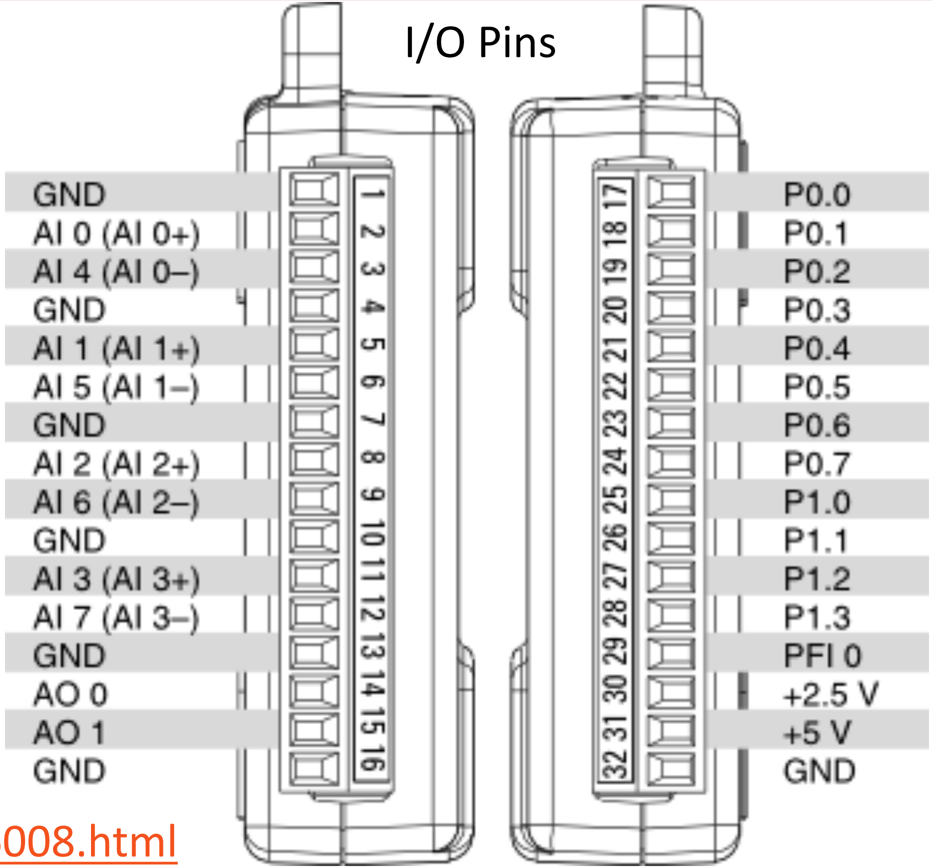


Resistors

NI USB-6008

We will use NI USB-6008 in our examples

I/O Pins

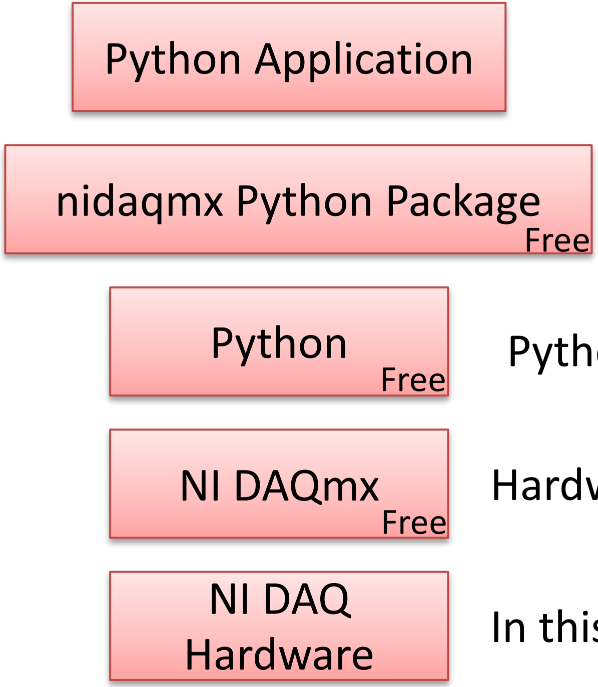


Signal	Pin	Pin	Signal
GND	1	17	P0.0
AI 0 (AI 0+)	2	18	P0.1
AI 4 (AI 0–)	3	19	P0.2
GND	4	20	P0.3
AI 1 (AI 1+)	5	21	P0.4
AI 5 (AI 1–)	6	22	P0.5
GND	7	23	P0.6
AI 2 (AI 2+)	8	24	P0.7
AI 6 (AI 2–)	9	25	P1.0
GND	10	26	P1.1
AI 3 (AI 3+)	11	27	P1.2
AI 7 (AI 3–)	12	28	P1.3
GND	13	29	PFI 0
AO 0	14	30	+2.5 V
AO 1	15	31	+5 V
GND	16	32	GND

http://www.ni.com/en-no/support/model.usb-6008.html

NI DAQ Device with Python

How to use a NI DAQ Device with Python



Python Application — Your Python Program

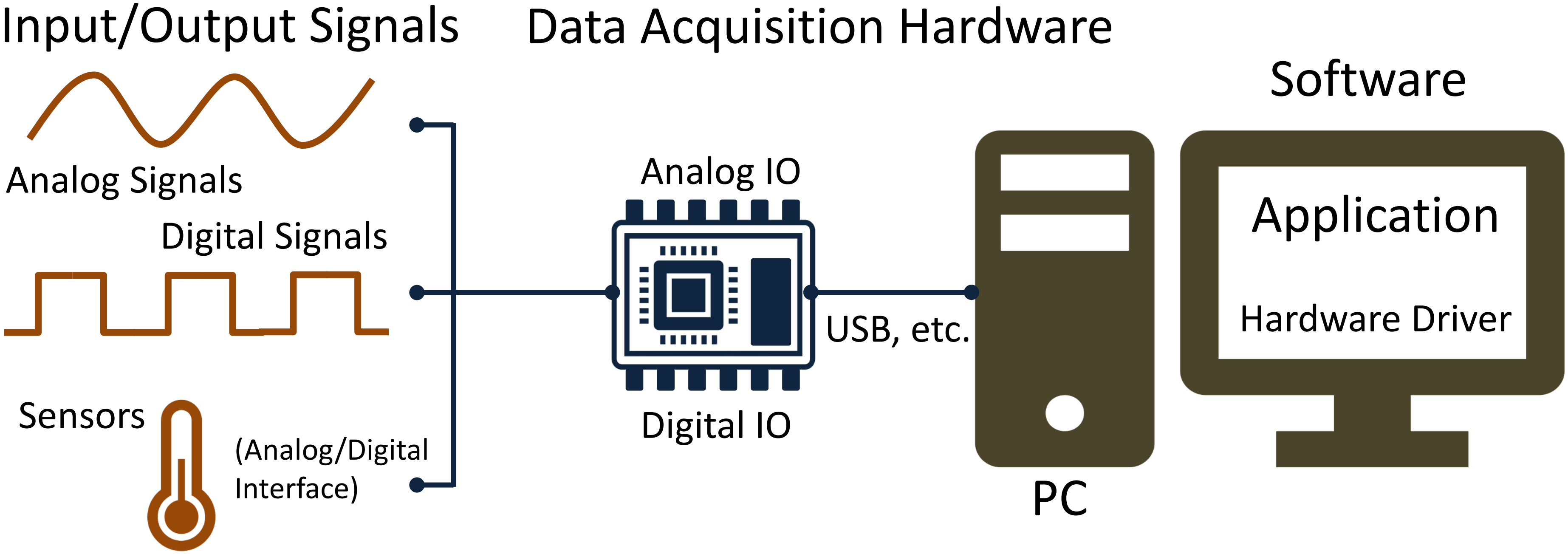
nidaqmx Python Package (Free) — Python Library/API for Communication with NI DAQmx Driver

Python (Free) — Python Programming Language

NI DAQmx (Free) — Hardware Driver Software

NI DAQ Hardware — In this Tutorial we will use USB-6008 DAQ Device or I/O Module

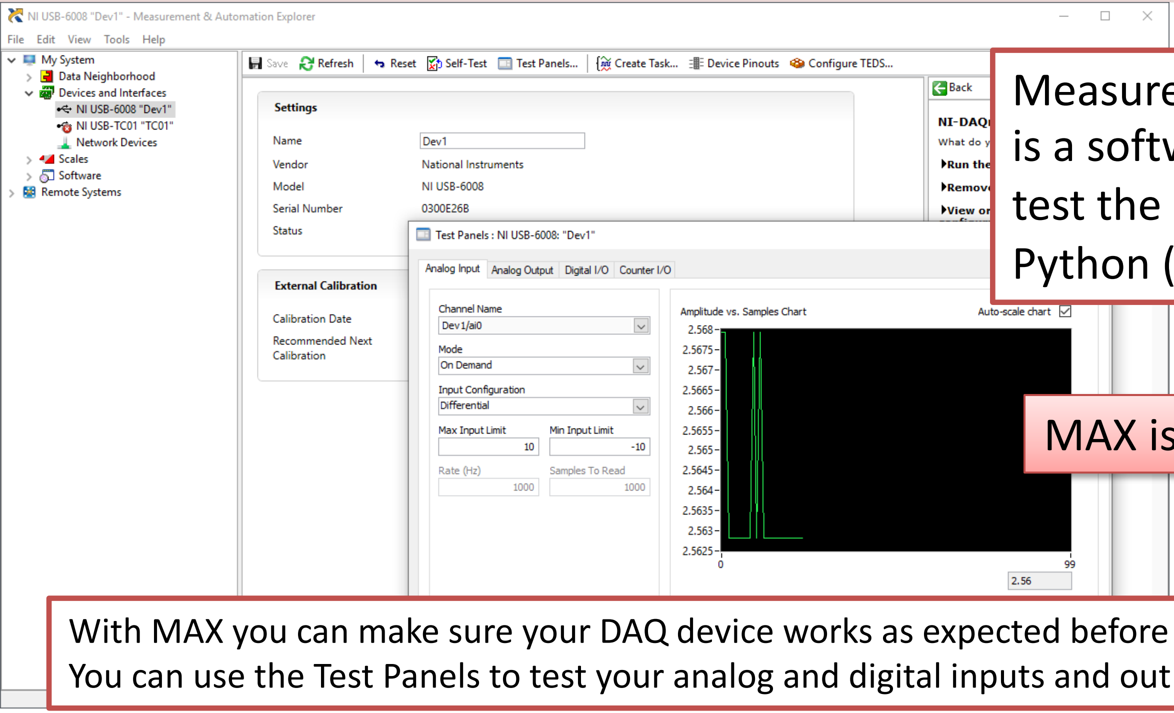
DAQ System


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

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)



NI USB-6008 "Dev1" - Measurement & Automation Explorer

Settings

Name: Dev1

Vendor: National Instruments

Model: NI USB-6008

Serial Number: 0300E26B

Status

External Calibration

Calibration Date

Recommended Next Calibration

Test Panels : NI USB-6008: "Dev1"

Channel Name: Dev1/ai0

Mode: On Demand

Input Configuration: Differential

Max Input Limit: 10

Min Input Limit: -10

Rate (Hz): 1000

Samples To Read: 1000

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

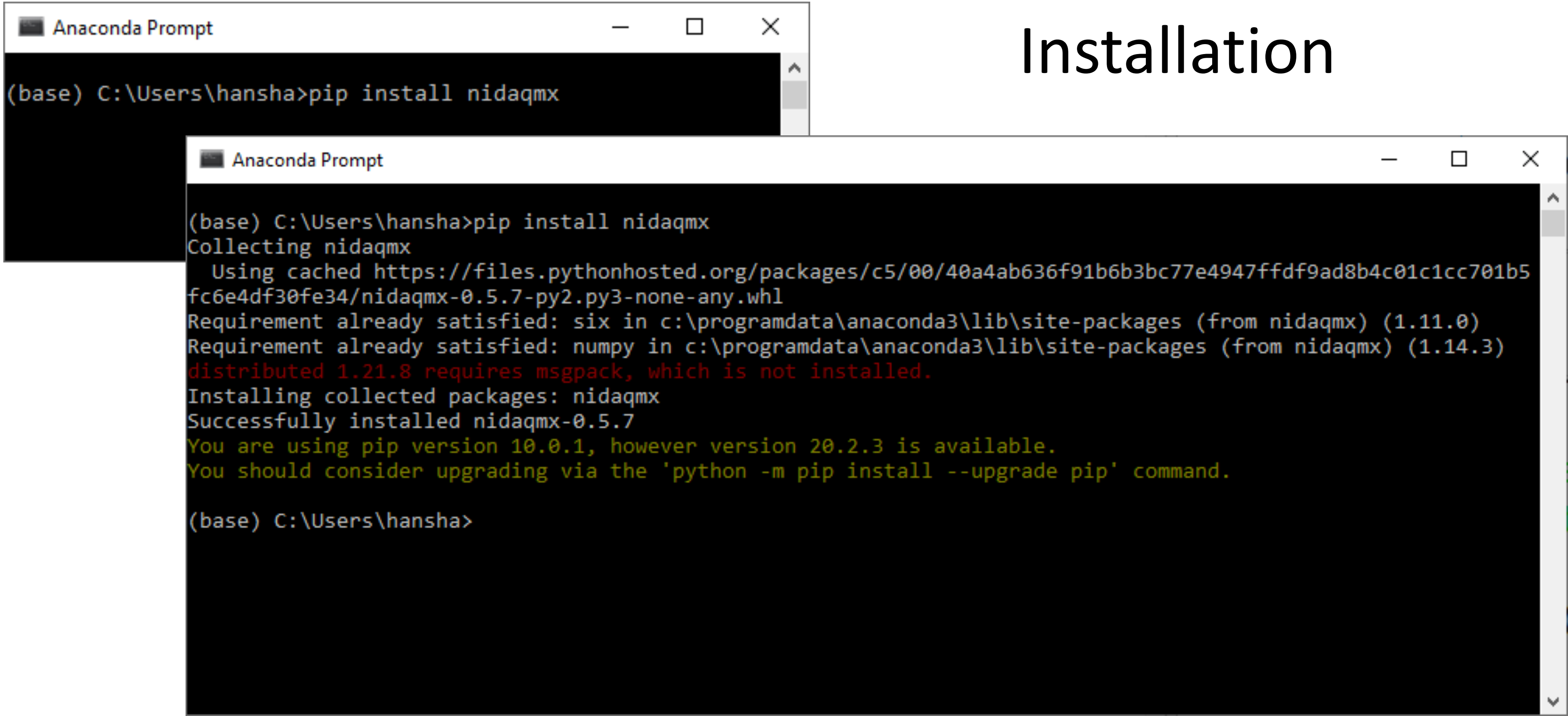
With MAX you can make sure your DAQ device works as expected before you start using it in your Python program. You can use the Test Panels to test your analog and digital inputs and outputs channels.

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



```
(base) C:\Users\hansha>pip install nidaqmx
```

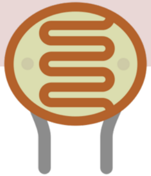
```
(base) C:\Users\hansha>pip install nidaqmx
Collecting nidaqmx
  Using cached https://files.pythonhosted.org/packages/c5/00/40a4ab636f91b6b3bc77e4947ffdf9ad8b4c01c1cc701b5
fc6e4df30fe34/nidaqmx-0.5.7-py2.py3-none-any.whl
Requirement already satisfied: six in c:\programdata\anaconda3\lib\site-packages (from nidaqmx) (1.11.0)
Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\site-packages (from nidaqmx) (1.14.3)
distributed 1.21.8 requires msgpack, which is not installed.
Installing collected packages: nidaqmx
Successfully installed nidaqmx-0.5.7
You are using pip version 10.0.1, however version 20.2.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.

(base) C:\Users\hansha>
```

https://www.halvorsen.blog

Light Sensor with Python

Hans-Petter Halvorsen

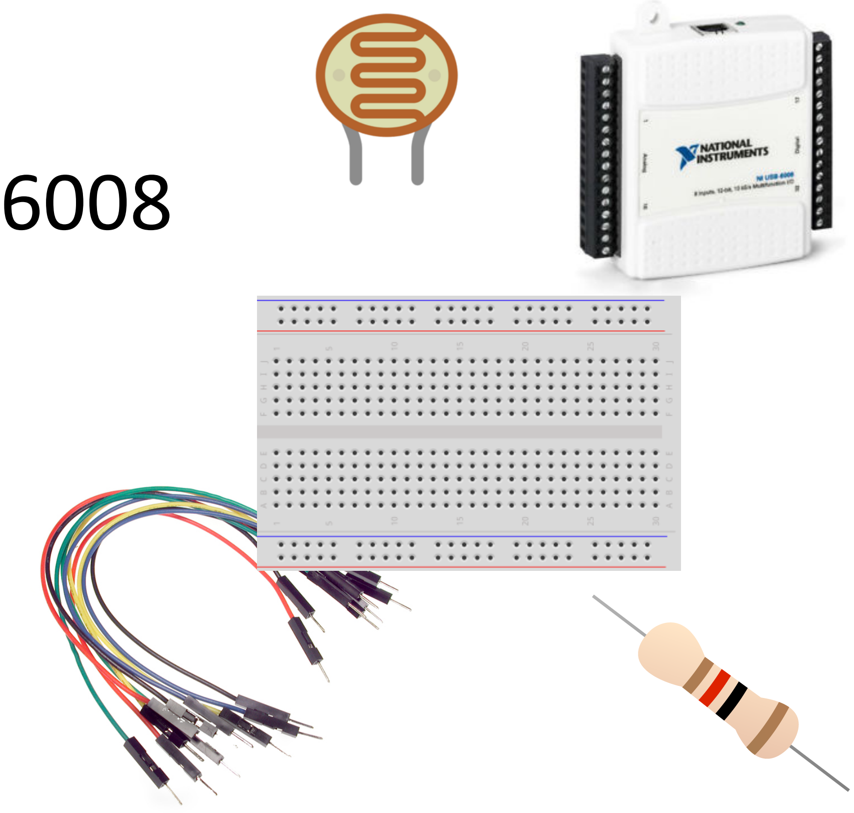
Light Sensor



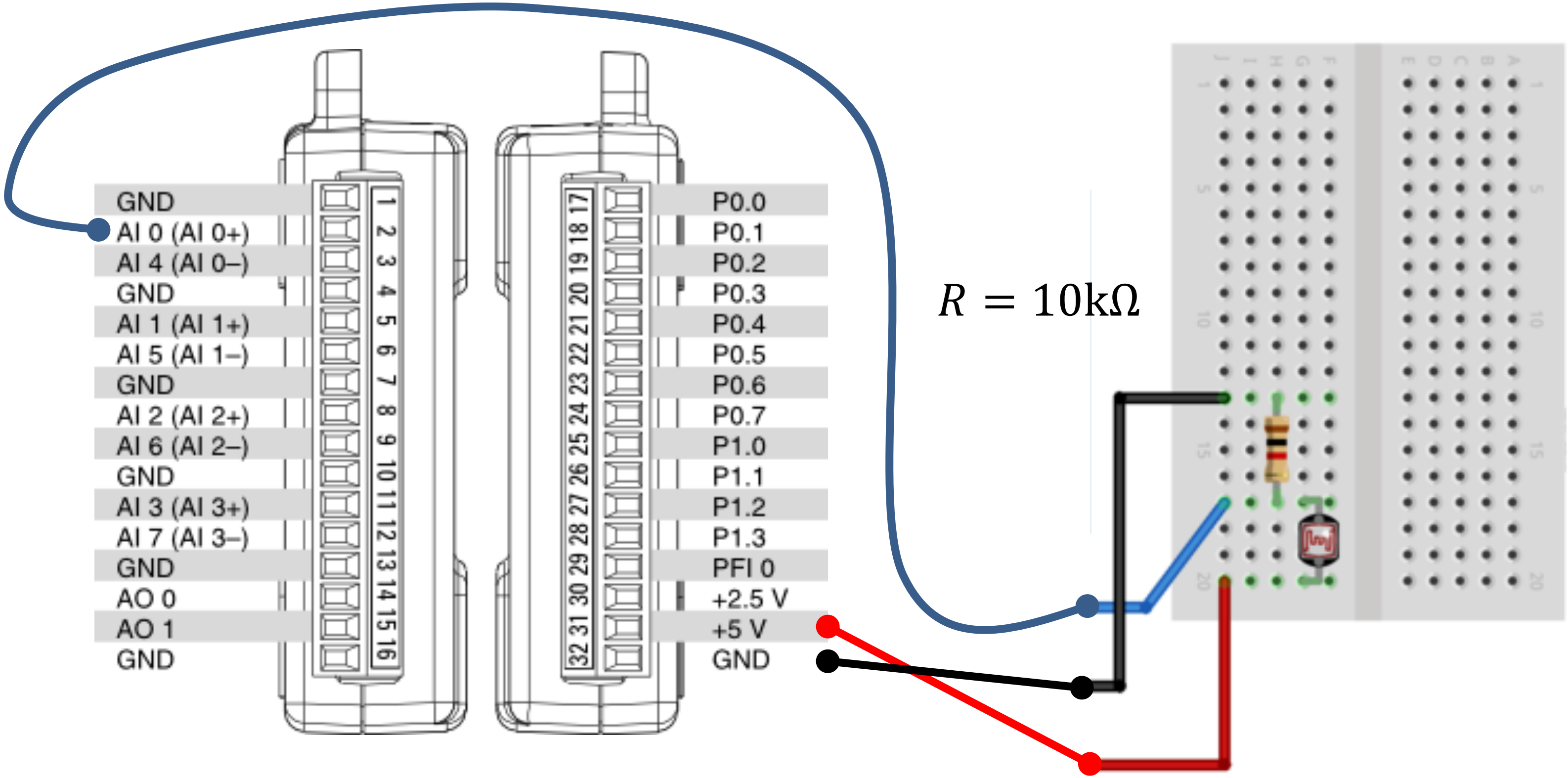
- Light sensor, Photocell (Photo resistor), LDR (light-dependent resistor)
- A light sensor / photocell is a sensor used to detect light.
- The resistance changes with the change in light intensity

Necessary Equipment

- PC
- DAQ Module, e.g., USB-6008
- Breadboard
- Light Sensor
- Wires (Jumper Wires)
- Resistors, $R = 10\ k\Omega$



Hardware Setup

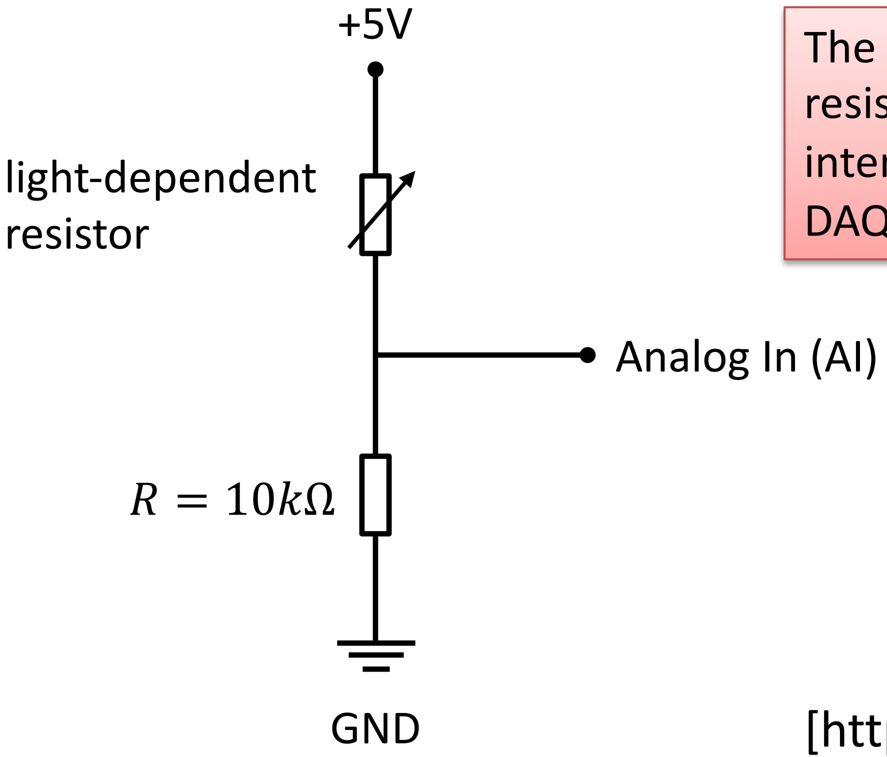


$R = 10\text{k}\Omega$

Made with Fritzing.org

Voltage Divider

The wiring is called a “Voltage Divider”



The light sensor is a light-dependent resistor. The resistance changes with the change in light intensity. We can only measure voltage with the DAQ device, so we need to use a “Voltage Divider”

[https://en.wikipedia.org/wiki/Voltage_divider]

Python Code

```
import nidaqmx

from nidaqmx.constants import (
    TerminalConfiguration)

task = nidaqmx.Task()
task.ai_channels.add_ai_voltage_chan("Dev1/ai0",
             terminal_config=TerminalConfiguration.RSE)
task.start()

value = task.read()
print(value)

task.stop
task.close()
```

Python Code – For Loop

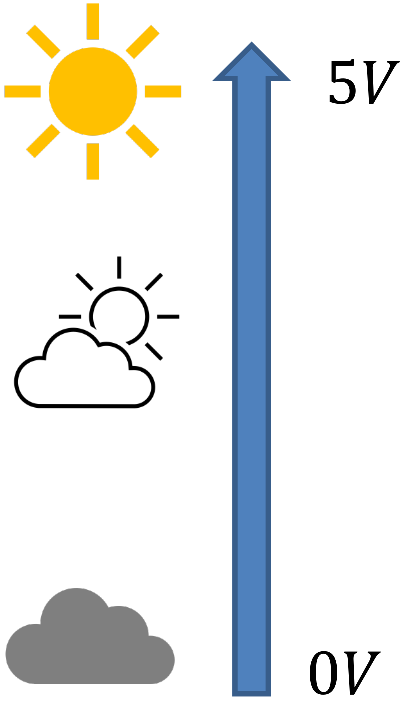
```
import nidaqmx
import time

from nidaqmx.constants import (
    TerminalConfiguration)

task = nidaqmx.Task()
task.ai_channels.add_ai_voltage_chan("Dev1/ai0",
          terminal_config=TerminalConfiguration.RSE)
task.start()

N = 60
for k in range(N):
    Vout = task.read()
    print(Vout)
    time.sleep(1)

task.stop
task.close()
```

Light Sensor Results

- The resistance changes with the change in light intensity.
- We measure the the voltage (using a Voltage Divider)
- When the Light Intensity gets Higher, the Voltage Level gets Higher

The Light Sensor has not very high accuracy, but you can typically use it to automatically turn on a light when it get dark outside (or inside), typically used in streetlights, etc.

High Light Intensity



Low Light Intensity

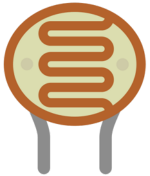
Light Sensor Example

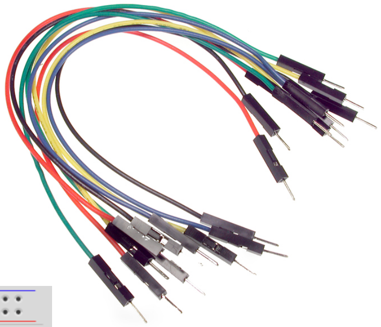
- The Light Sensor has not very high accuracy, but you can typically use it to automatically turn on a light when it get dark outside (or inside)
- In this example we will use a light sensor to measure the light intensity of the room.
  - If it's dark, we will turn on the light (LED)
  - If it's bright, we'll turn off the light (LED)

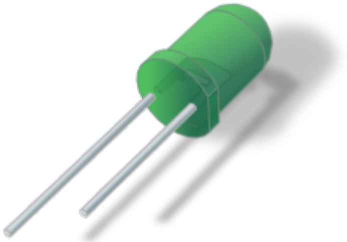
Necessary Equipment

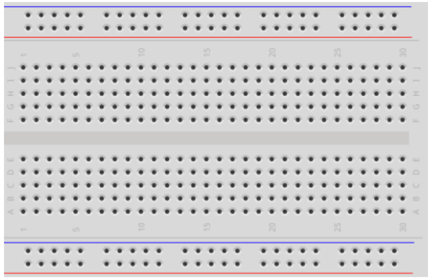
- PC
- DAQ Module, e.g., USB-6008
- Breadboard
- Light Sensor
- Wires (Jumper Wires)
- Resistors

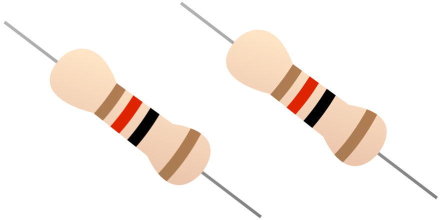
  $$R = 270\Omega$$
  $$R = 10k\Omega$$



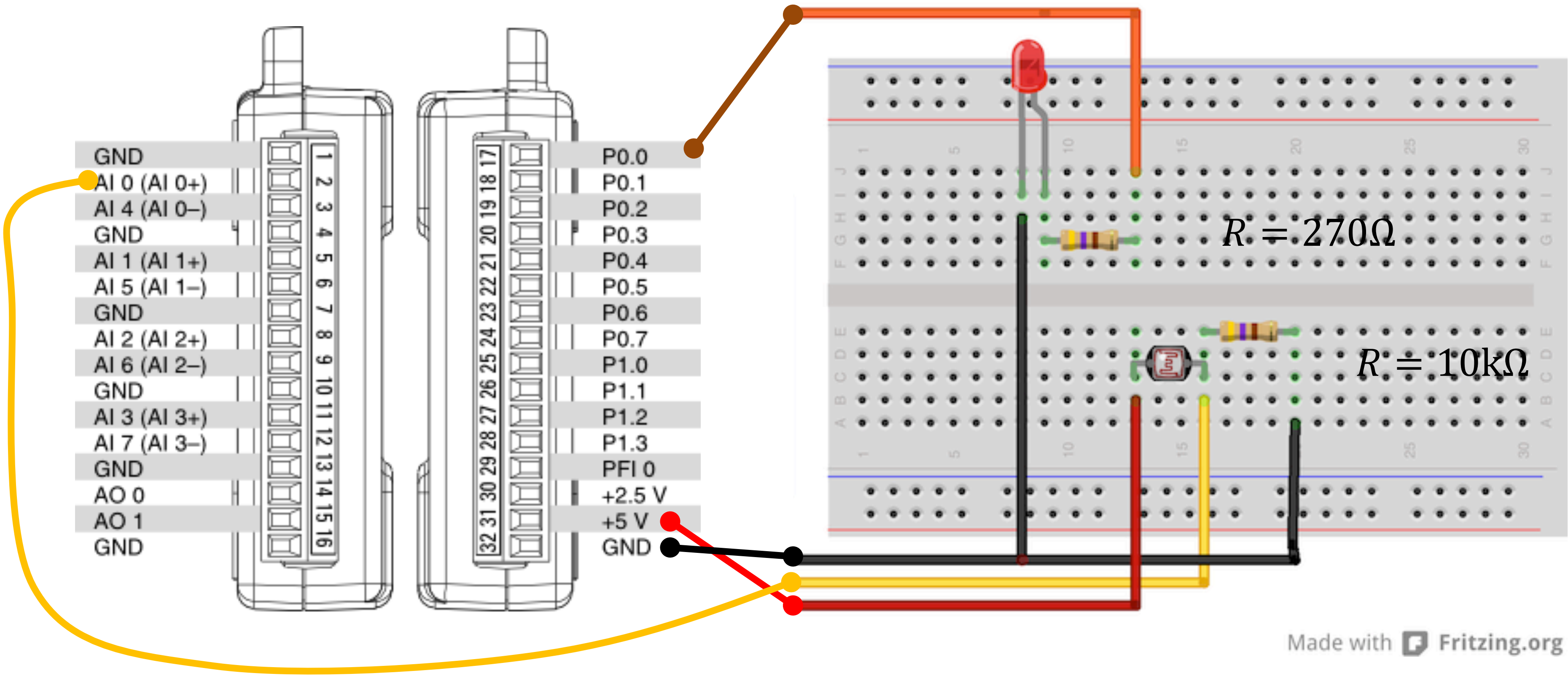








Hardware Setup



Python Code

If it's dark, we will turn on the light (LED)
If it's bright, we'll turn off the light (LED)

In the Example a the “Bright Level” is set to 0.2V

This value needs to be adjusted (“trial and error“) depending on the use of the application.

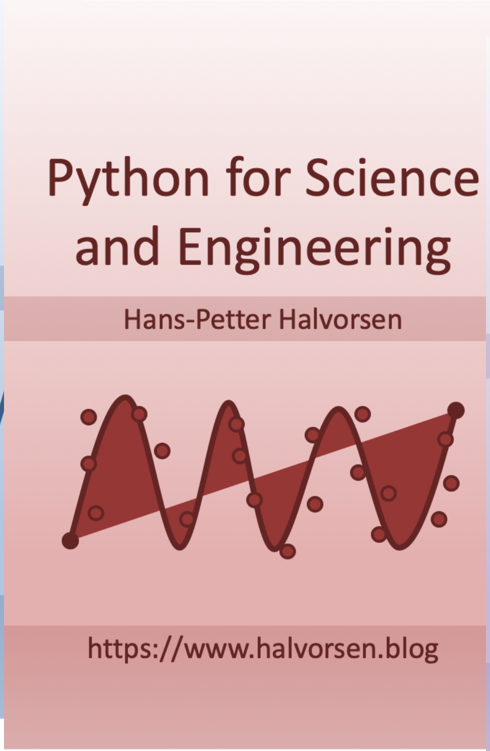
```python
import nidaqmx
import time

from nidaqmx.constants import (
    TerminalConfiguration)

task_ai = nidaqmx.Task()
task_ai.ai_channels.add_ai_voltage_chan("Dev1/ai0",
            terminal_config=TerminalConfiguration.RSE)
task_ai.start()

task_do = nidaqmx.Task()
task_do.do_channels.add_do_chan("Dev1/port0/line0")
task_do.start()

brightlevel = 0.2
N = 60
for k in range(N):
    Vout = task_ai.read()
    print(round(Vout,2))

    task_do.write(True)

    if Vout < brightlevel:
        task_do.write(True)
    else:
        task_do.write(False)
    time.sleep(1)

task_do.write(False)

task_ai.stop; task_ai.close()
task_do.stop; task_do.close()
```

Additional Python Resources

Python Programming

Hans-Petter Halvorsen

https://www.halvorsen.blog



Python for Science and Engineering

Hans-Petter Halvorsen

https://www.halvorsen.blog

Python for Control Engineering

Hans-Petter Halvorsen

https://www.halvorsen.blog

Python for Software Development

Hans-Petter Halvorsen

Python Software Development ☒

Do you want to learn Software Development?

https://www.halvorsen.blog

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