https://www.halvorsen.blog

Internet of Things and Raspberry Pi

Hans-Petter Halvorsen

Table of Contents

- Introduction
- Internet of Things
- Raspberry Pi
- Raspberry Pi and Python Programming
- DAQ and IoT Sensors
- Digital Sensor Interfaces
  - SPI
  - I2C
  - 1-Wire
- NoSQL and MongoDB
- ThingSpeak
- MQTT (“IoT Communication Protocol”)
- Raspberry Pi with MATLAB

Introduction

- With Internet of Things (IoT) and Cloud Services Datalogging has reach a new era.
- The Data are typically stored in the Cloud using traditional SQL databases or more modern systems like NoSQL databases or different IoT cloud services (e.g., ThingSpeak, MongoDB Atlas, etc.).
- We will use Raspberry Pi. Raspberry Pi is popular to use in different IoT applications.
- We will primarily use Python, but also MATLAB as programming languages.

Topics

- Internet of Things (IoT)
- Microcomputers, Raspberry Pi and Linux
- Python (Raspberry Pi + Python are Powerful!)
- IoT Sensors, Digital Interfaces: SPI/I2C
- NoSQL (MongoDB)
- ThingSpeak (IoT Cloud Service)
- MQTT (IoT Communication Protocol)
- Raspberry Pi with MATLAB

Delivery

- Retrieve data from **Temperature Sensors**, e.g., TMP36 or/and an I2C/1-Wire Temperature sensor. Use the Python programming language.
- **Lowpass Filter** to remove noise from the signal
- **Alarm** Handling
- The data should be stored in a **MongoDB** database (NoSQL), either locally or in the cloud.
- The data should be also be stored in the cloud service **ThingSpeak**
- **MQTT** Communication
- **Cyber Security**: Give an overview of how Raspberry Pi OS (Linux) handles Security.

For more details, see the web site

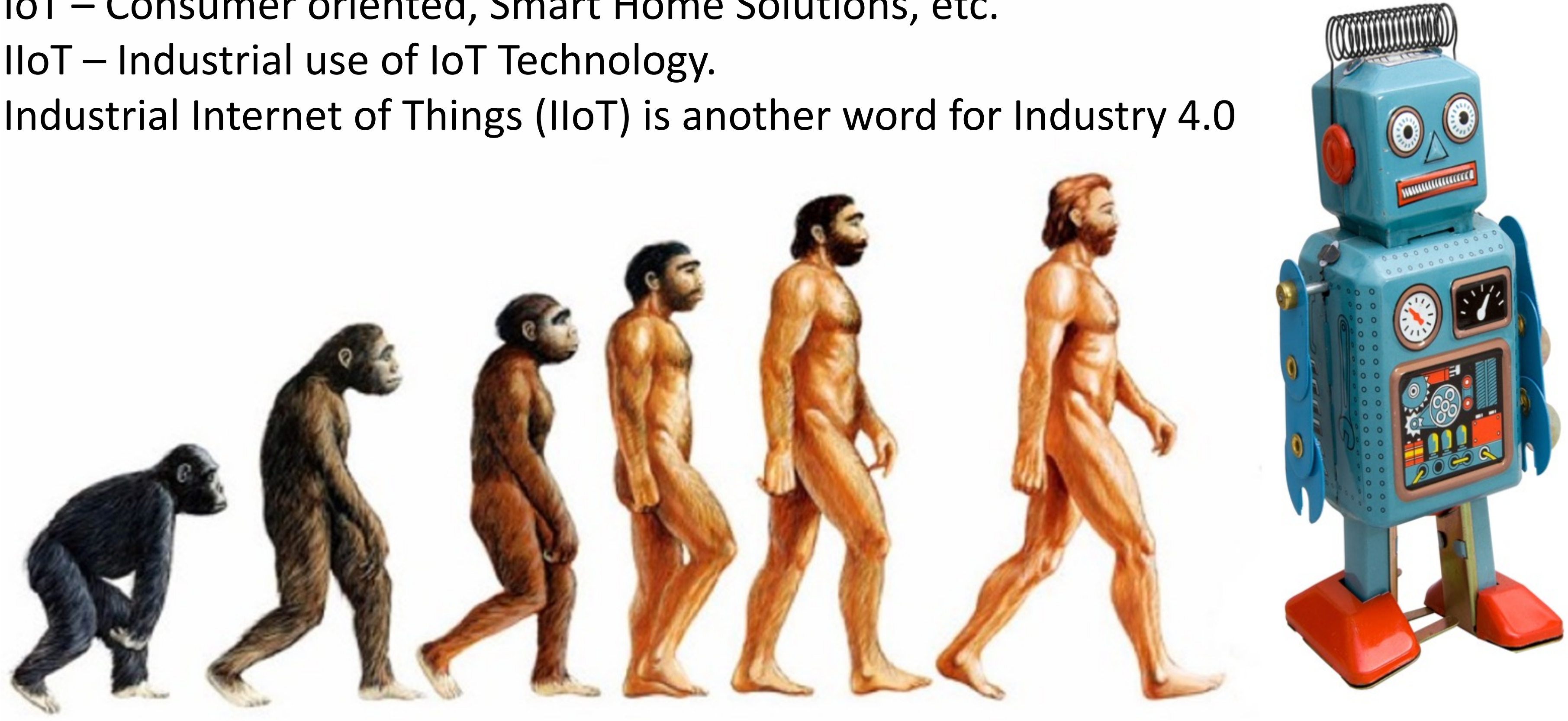
Internet of Things

Internet of Things (IoT)

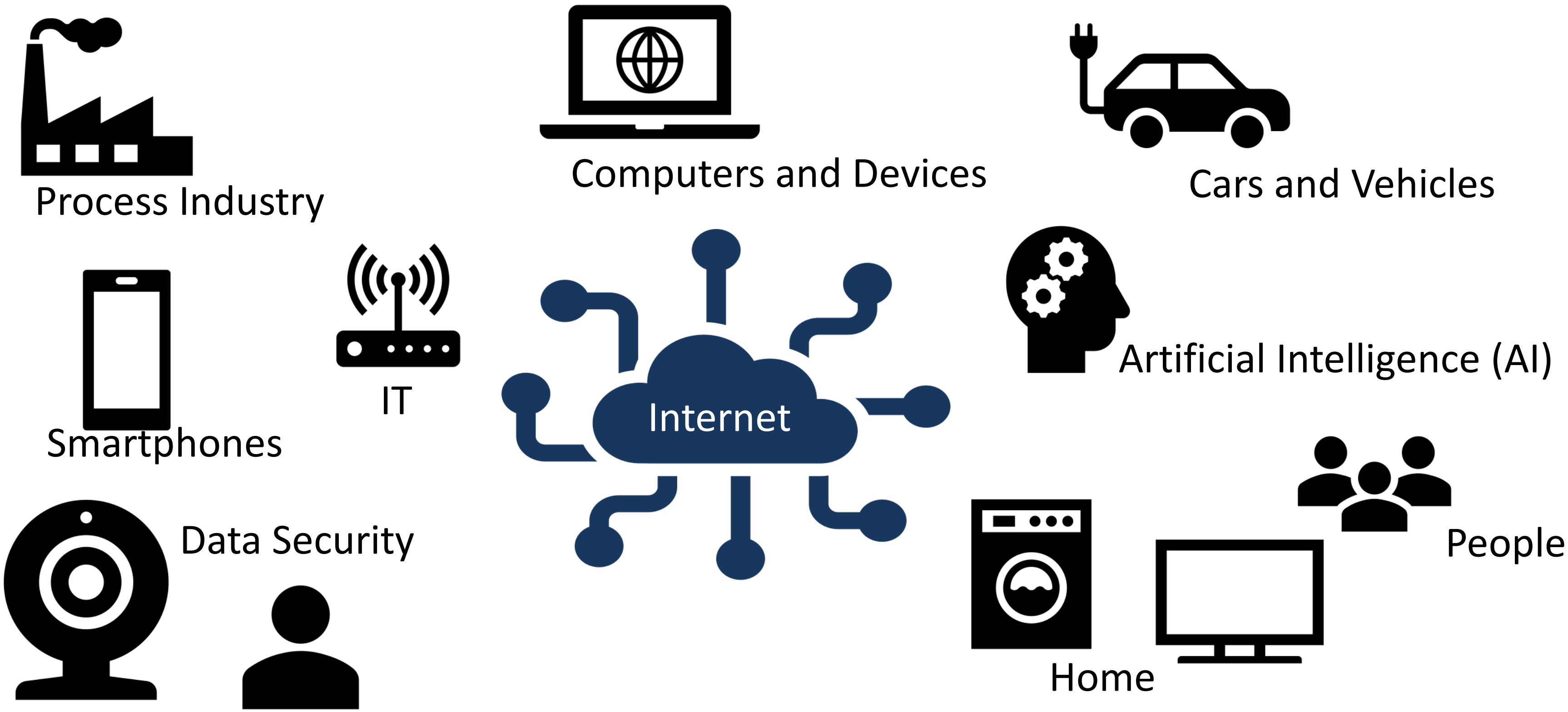
IoT – Consumer oriented, Smart Home Solutions, etc.

IIoT – Industrial use of IoT Technology.

Industrial Internet of Things (IIoT) is another word for Industry 4.0



Internet of Things (IoT)



Process Industry

Computers and Devices

Cars and Vehicles

Smartphones

IT

Internet

Artificial Intelligence (AI)

Data Security

People

Home

Soon everything will be connected to the Internet – even your Coffee Maker

Internet of Things (IoT)

Relevant Topics:

Sensor Technology

Datalogging and Monitoring

Machine Learning

Database Systems

Internet of Things (IoT)

Cloud Computing

Industrial Internet of Things
and Industry 4.0

Cyber Security

Raspberry Pi

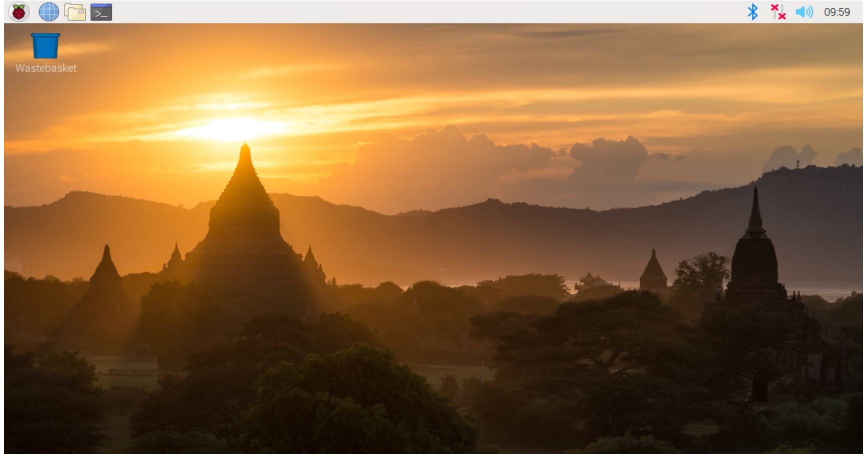
Hans-Petter Halvorsen

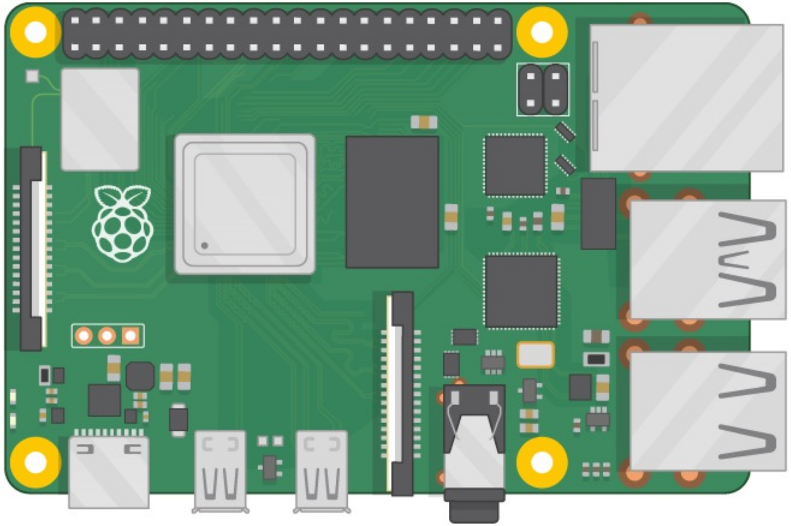
Table of Contents

Raspberry Pi

Raspberry Pi is a tiny (about 9x6cm), low-cost ($35+), single-board computer that supports embedded Linux operating systems

The recommended Operating System is called Raspberry Pi OS (Linux based)



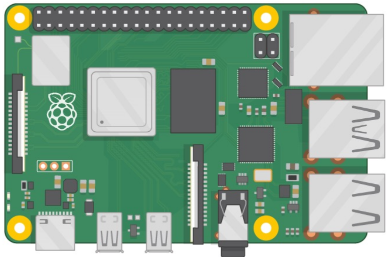


https://www.raspberrypi.org

Raspberry Pi vs. Arduino

- Raspberry PI is a Microcomputer
- It has an ordinary Operating System (OS)
- You can connect USB devices, Keyboard, Mouse, Monitors, etc.
- It has a “hard-drive“ in form of a microSD card
- RP has Bluetooth, Wi-Fi, and Ethernet connection
- RP has basically all the features an ordinary computer has but in a much smaller package
- Uptill 8 Gb RAM
- RP runs Linux applications

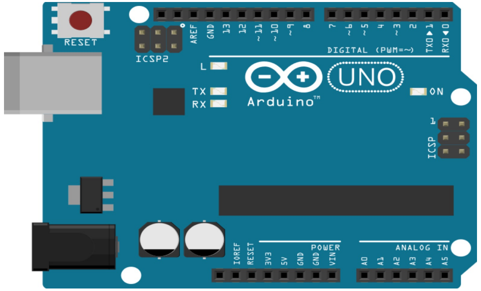
- Arduino is a Microcontroller
- Arduino has a Bootloader and not an ordinary operating system
- Arduino is NOT a computer, only a small controller, whose purpose is to control things
- No Bluetooth, Wi-Fi (some models have), and Ethernet (but can be provided as so-called Shields)
- Very little RAM (a few Kb)
- Inexpensive



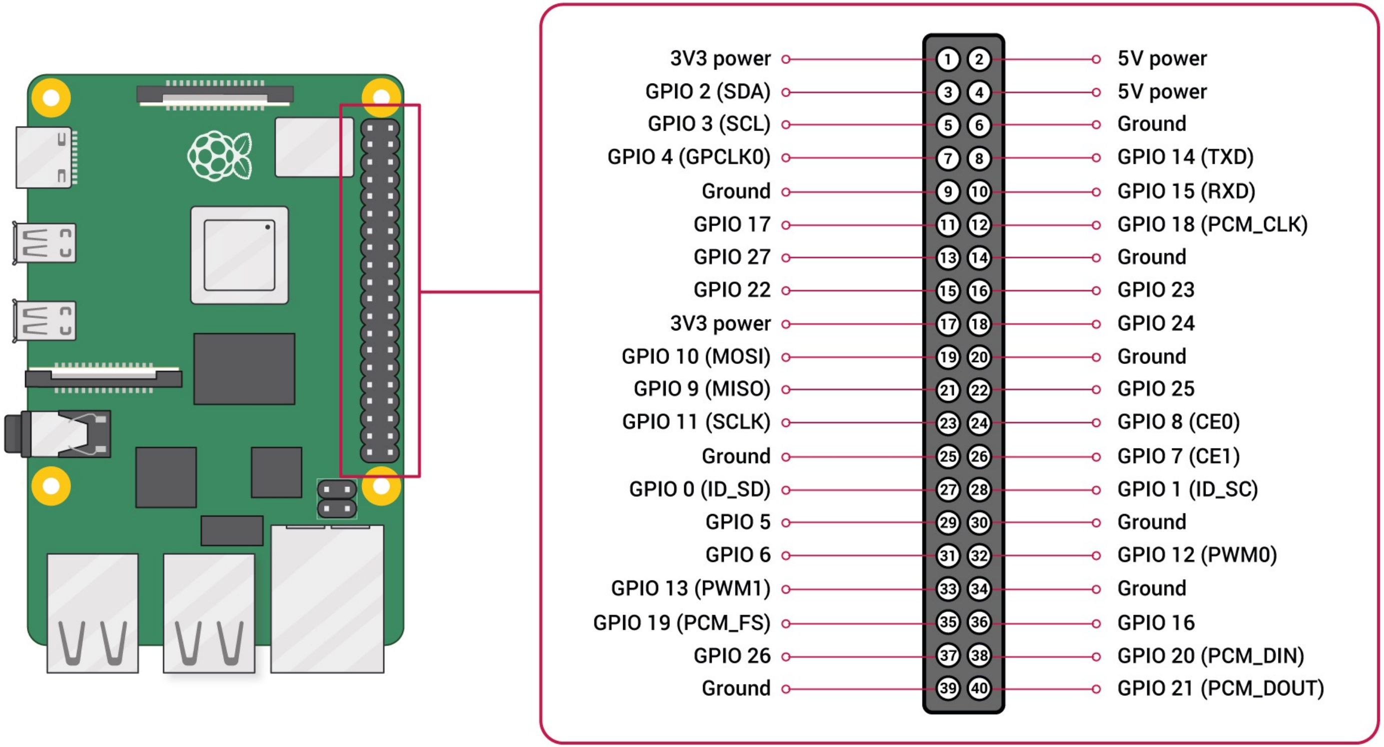
Both have Digital Pins

Both have SPI and I2C

Arduino (UNO) has also Analog Input Pins



Raspberry Pi GPIO



3V3 power	1	2	5V power
GPIO 2 (SDA)	3	4	5V power
GPIO 3 (SCL)	5	6	Ground
GPIO 4 (GPCLK0)	7	8	GPIO 14 (TXD)
Ground	9	10	GPIO 15 (RXD)
GPIO 17	11	12	GPIO 18 (PCM_CLK)
GPIO 27	13	14	Ground
GPIO 22	15	16	GPIO 23
3V3 power	17	18	GPIO 24
GPIO 10 (MOSI)	19	20	Ground
GPIO 9 (MISO)	21	22	GPIO 25
GPIO 11 (SCLK)	23	24	GPIO 8 (CE0)
Ground	25	26	GPIO 7 (CE1)
GPIO 0 (ID_SD)	27	28	GPIO 1 (ID_SC)
GPIO 5	29	30	Ground
GPIO 6	31	32	GPIO 12 (PWM0)
GPIO 13 (PWM1)	33	34	Ground
GPIO 19 (PCM_FS)	35	36	GPIO 16
GPIO 26	37	38	GPIO 20 (PCM_DIN)
Ground	39	40	GPIO 21 (PCM_DOUT)

A powerful feature of the Raspberry Pi is the GPIO (general-purpose input/output) pins. The Raspberry Pi has a 40-pin GPIO header as seen in the image

Raspberry Pi OS

- In order make your Raspberry Pi up and running you need to install an Operating System (OS)
- The OS for Raspberry Pi is called “Raspberry Pi OS“ (previously known as Raspbian)
- Raspberry Pi runs a version of an operating system called Linux (Windows and macOS are other operating systems).
- To install the necessary OS, you need a microSD card
- Then you use the “Raspberry Pi Imager“ in order to download the OS to the microSD card.

https://www.raspberrypi.org/software/

Start using Raspberry Pi

Raspberry Pi OS

- Put the microSD card into the Raspberry Pi
- Connect Monitor, Mouse and Keyboard
- Connect Power Supply
- Follow the Instructions on Screen to setup Wi-Fi

https://www.raspberrypi.org/software/

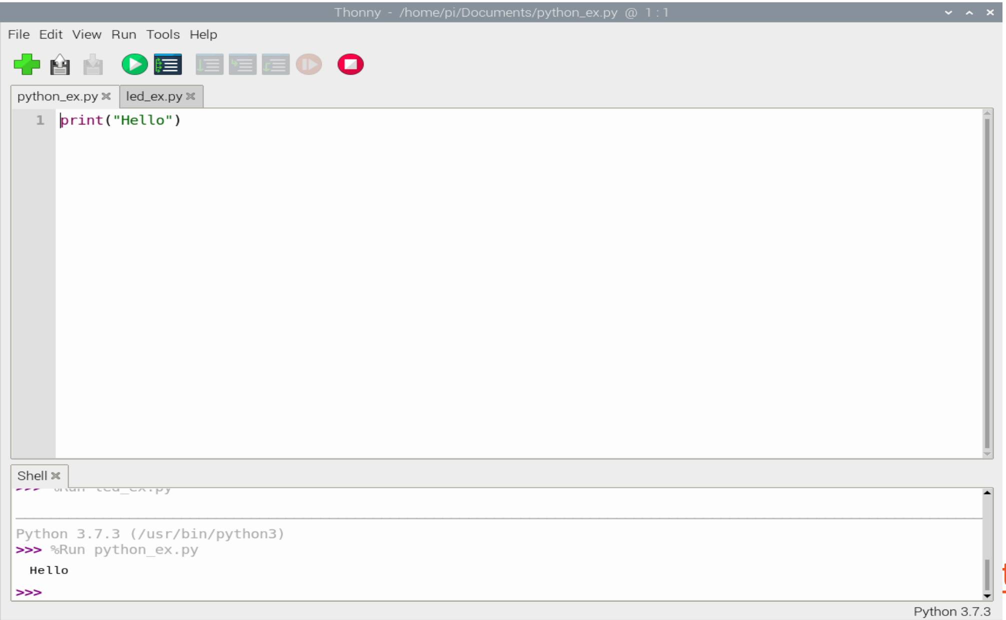
Raspberry Pi and Python Programming

Hans-Petter Halvorsen

Table of Contents

Raspberry Pi and Python

The Raspberry Pi OS comes with a basic Python Editor called Thonny


Thonny - /home/pi/Documents/python_ex.py @ 1 : 1
File Edit View Run Tools Help
python_ex.py
led_ex.py
Shell
Python 3.7.3

```
print("Hello")
```

```
>>> %Run led_ex.py

Python 3.7.3 (/usr/bin/python3)
>>> %Run python_ex.py
 Hello
>>>
```

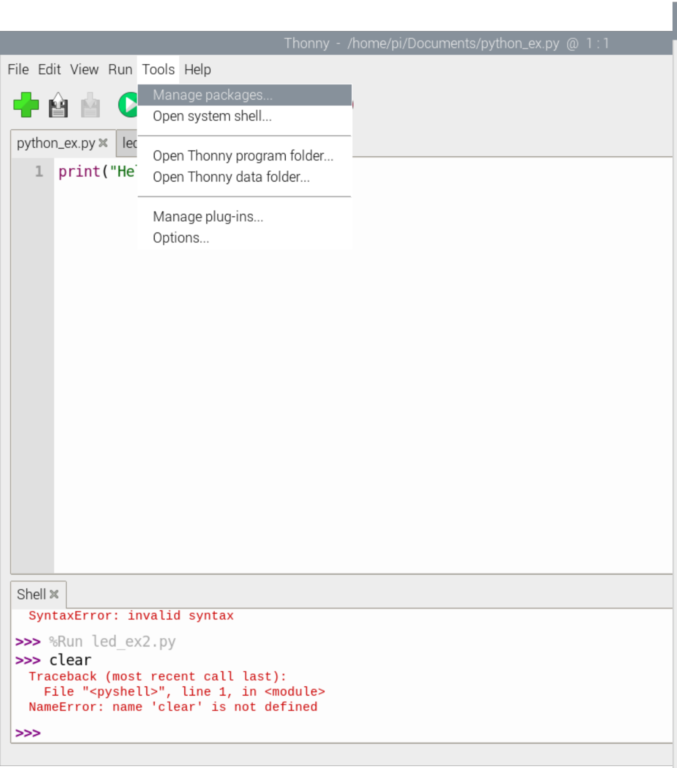
Raspberry Pi + Python are a powerful combination!

But you can install and use other Python Editors if you prefer

tps://www.raspberrypi.org/documentation/usage/python/

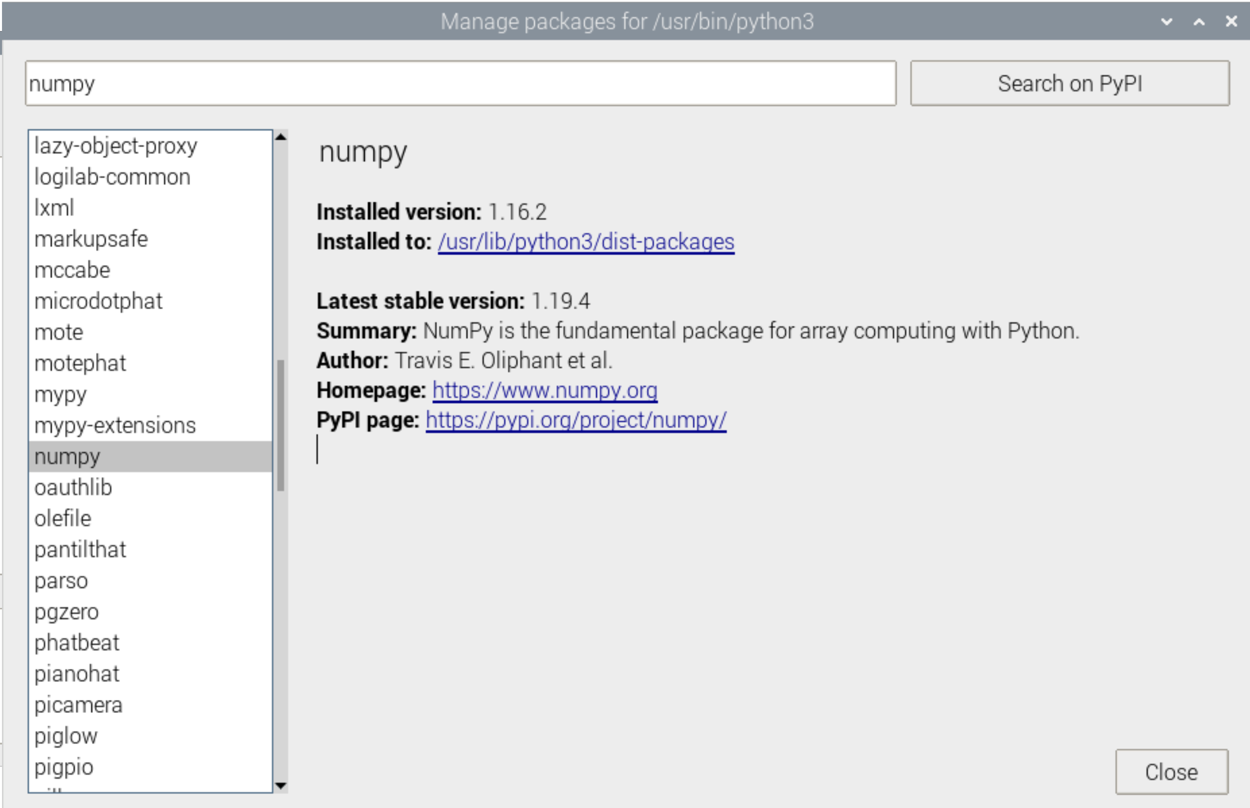
Python Packages with Thonny

Tools -> Manage packages…



```
print("He
```

```
SyntaxError: invalid syntax

>>> %Run led_ex2.py
>>> clear
Traceback (most recent call last):
  File "<pyshell>", line 1, in <module>
NameError: name 'clear' is not defined

>>>
```



numpy

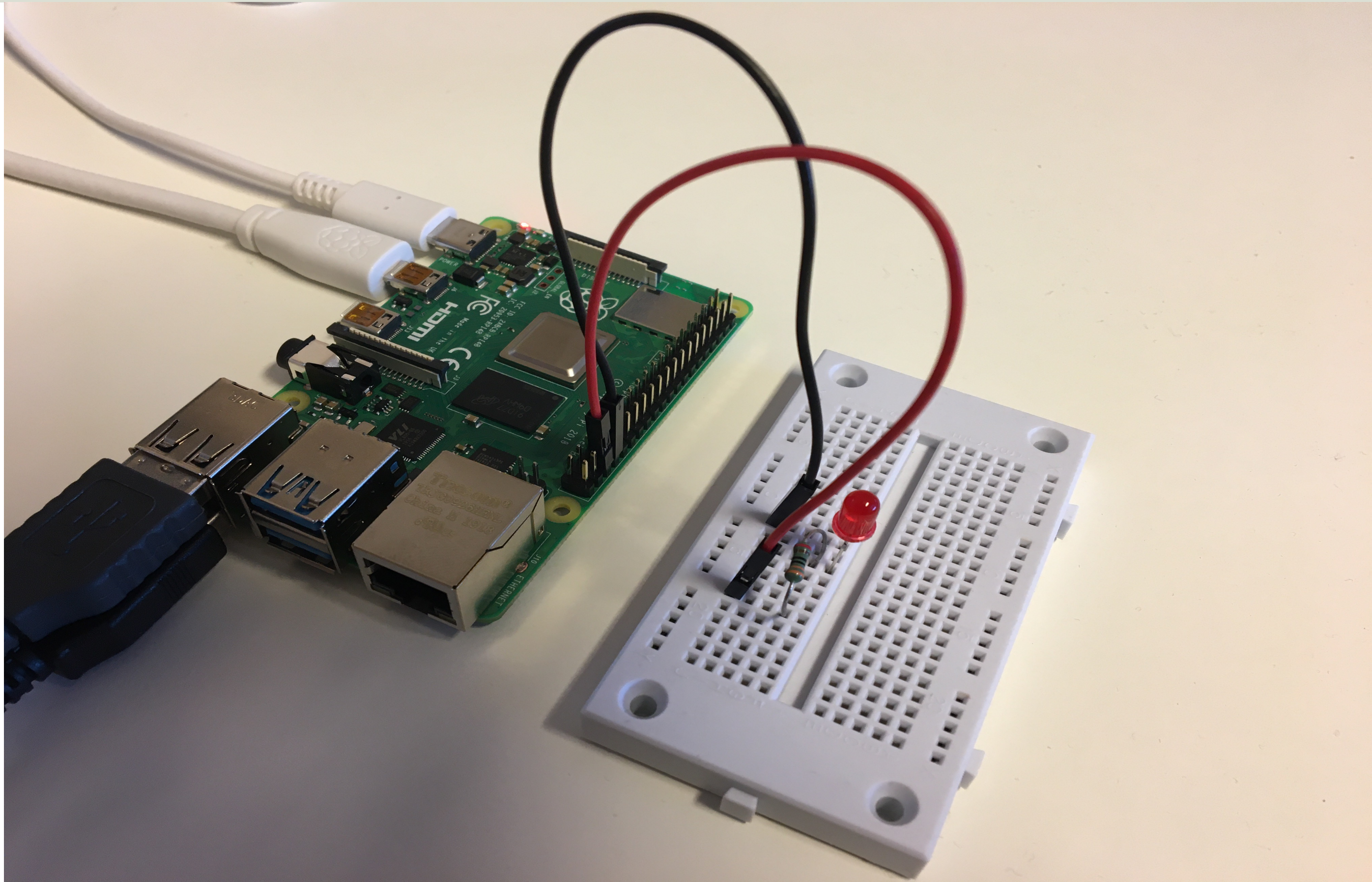
lazy-object-proxy
logilab-common
lxml
markupsafe
mccabe
microdotphat
mote
motephat
mypy
mypy-extensions
numpy
oauthlib
olefile
pantilthat
parso
pgzero
phatbeat
pianohat
picamera
piglow
pigpio

numpy

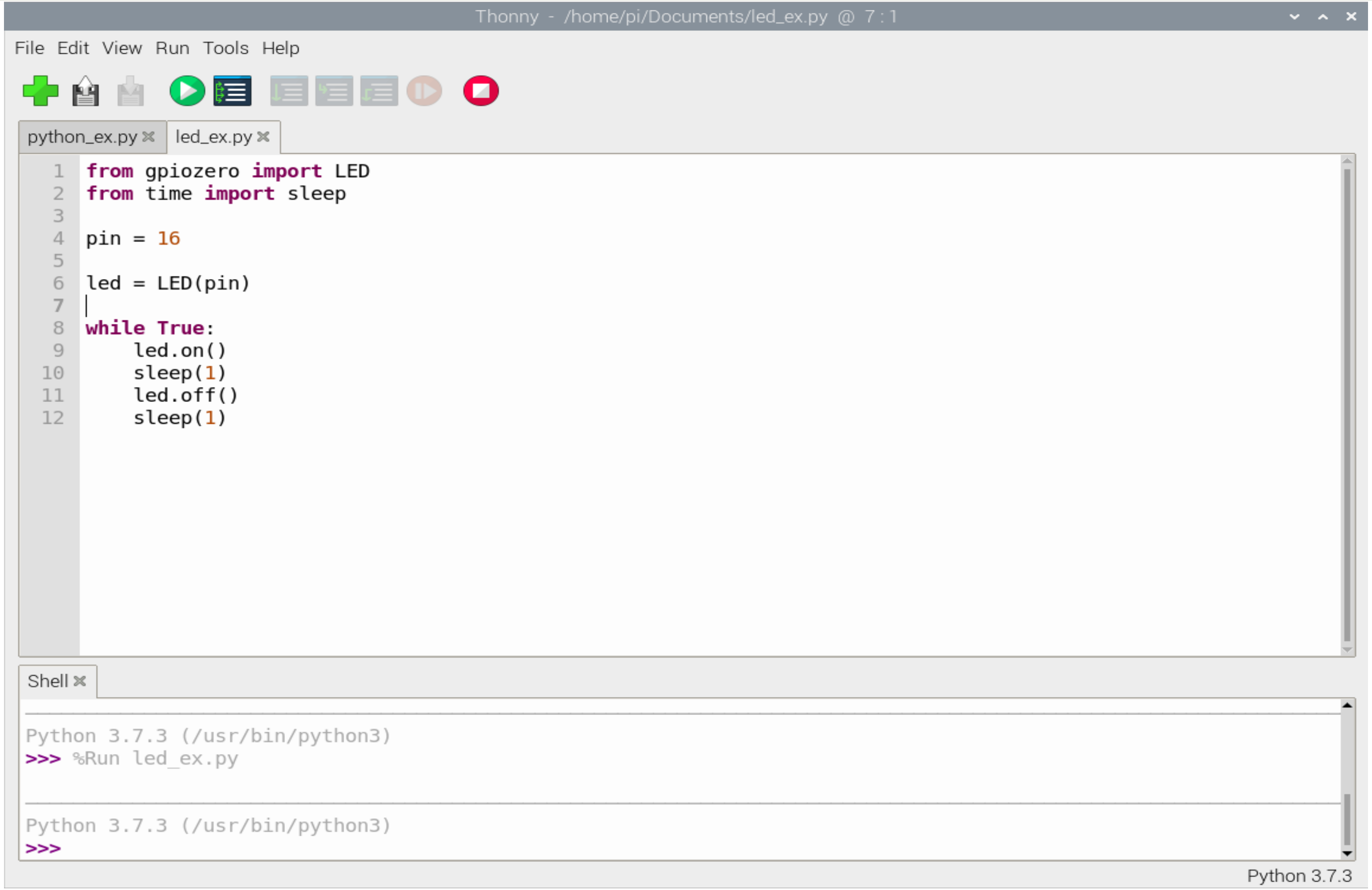
Installed version: 1.16.2
Installed to: /usr/lib/python3/dist-packages

Latest stable version: 1.19.4
Summary: NumPy is the fundamental package for array computing with Python.
Author: Travis E. Oliphant et al.
Homepage: https://www.numpy.org
PyPI page: https://pypi.org/project/numpy/

LED Example: Setup and Wiring



LED Example: Python Code



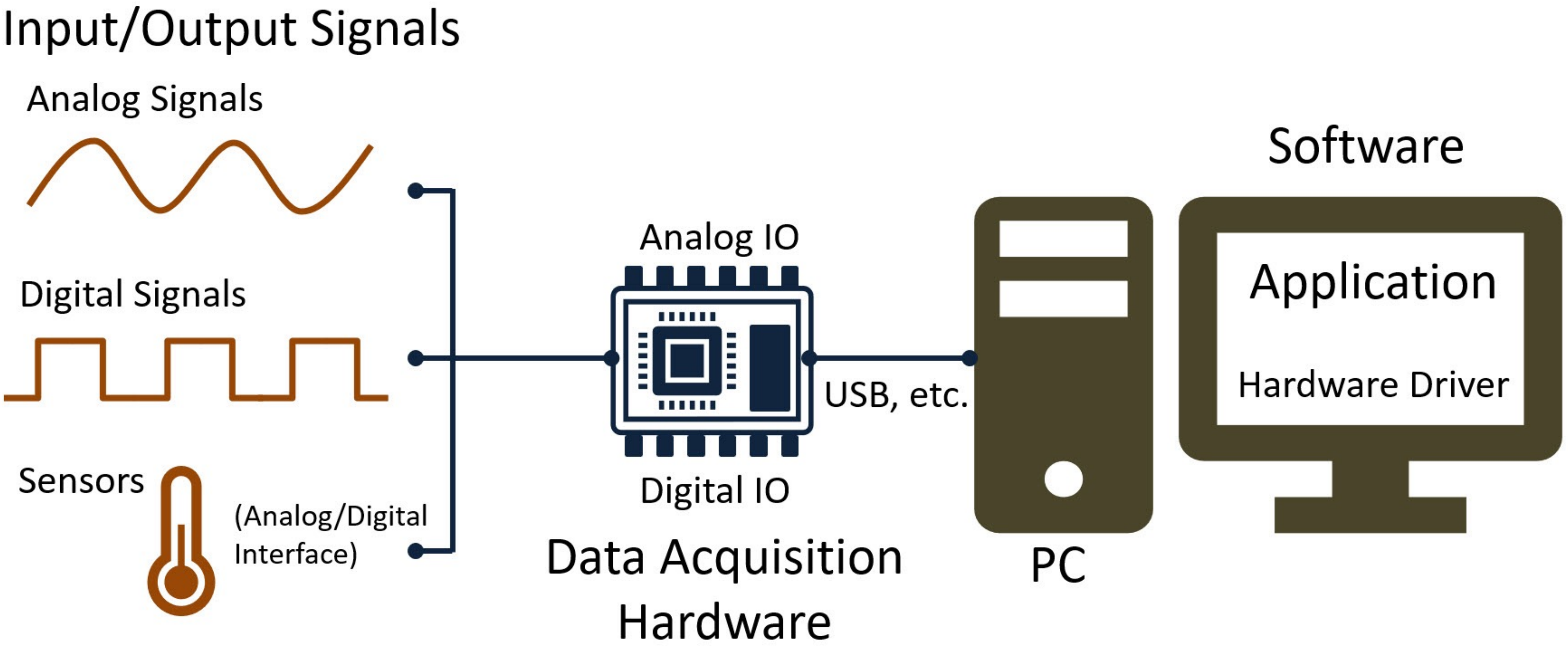
```python
from gpiozero import LED
from time import sleep

pin = 16

led = LED(pin)

while True:
    led.on()
    sleep(1)
    led.off()
    sleep(1)
```

```
Python 3.7.3 (/usr/bin/python3)
>>> %Run led_ex.py

Python 3.7.3 (/usr/bin/python3)
>>>
```

DAQ and IoT Sensors

Table of Contents

DAQ and IoT Sensors



DAQ and IoT Sensors

- DAQ (Data Acquisition) and Sensors are needed and used in all IoT applications.
- DAQ is the process of getting data from the sensors into your software.
- Here will some popular IoT sensors be presented.

IoT Sensors

- IoT sensors comes in many flavors.
- Below, some IoT sensors are presented they can be programmed with Python.
- IoT Sensor Examples: TMP36, Thermistor 10K, TC74, BME280, and DHT11/22.

GPIO Python Libraries

- **GPIO Zero**
  - https://pypi.org/project/gpiozero/
- **RPi.GPIO**
  - https://pypi.org/project/RPi.GPIO/
- **smbus** (used for I2C communication)
- **CircuitPython** - Typically, you would use the Python GPIO Zero Library, but it does not work so well with SPI/I2C Sensors

https://www.halvorsen.blog

Digital Sensor Interfaces

Hans-Petter Halvorsen

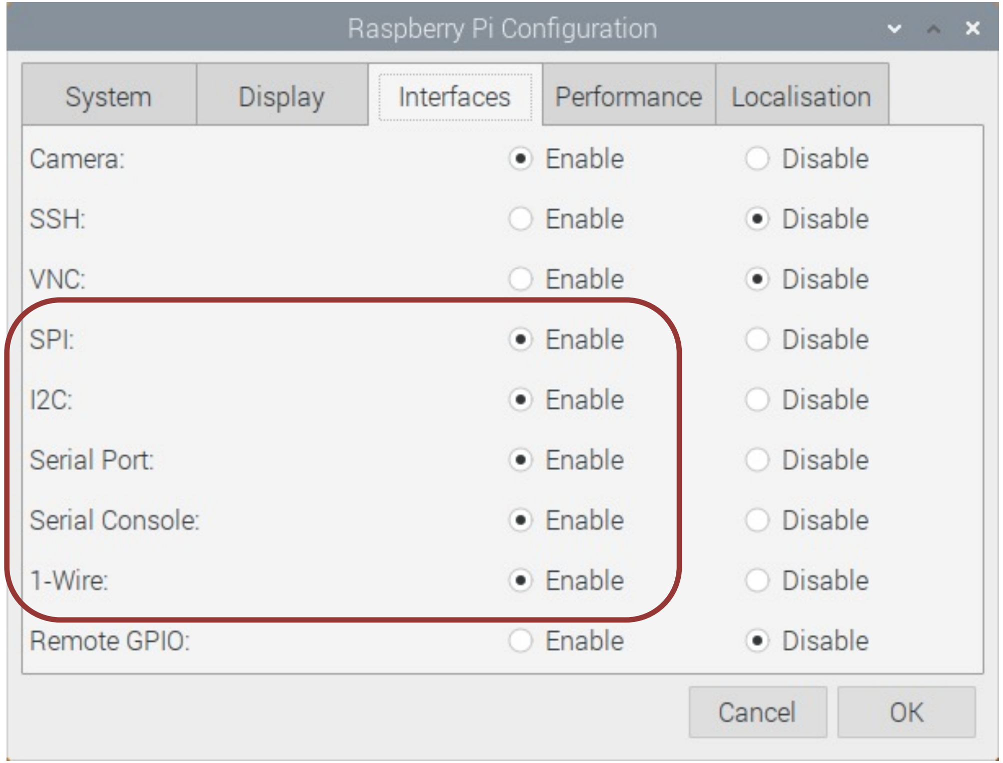
Table of Contents

Digital Interfaces

- Raspberry Pi has only Digital pins
- In order to connect and use Sensors we typically need to use one or more of these digital interfaces:
  - **SPI** Interface
  - **I2C** Interface
  - **1-Wire** Interface

Enable Access to Interfaces

- SPI Interface
- I2C Interface
- 1-Wire Interface



Camera:	● Enable	○ Disable
SSH:	○ Enable	● Disable
VNC:	○ Enable	● Disable
SPI:	● Enable	○ Disable
I2C:	● Enable	○ Disable
Serial Port:	● Enable	○ Disable
Serial Console:	● Enable	○ Disable
1-Wire:	● Enable	○ Disable
Remote GPIO:	○ Enable	● Disable

CircuitPython and Adafruit-Blinka

- CircuitPython adds the Circuit part to the Python part.
- Letting you program in Python and talk to Circuitry like sensors, motors, and LEDs!
- Typically, you would use the Python GPIO Zero Library, but it does not work with SPI/I2C Sensors
- On Raspberry Pi we need to install Adafruit-Blinka. This is a CircuitPython API that can be used on Linux devices such as the Raspberry Pi
- Adafruit-Blinka: https://pypi.org/project/Adafruit-Blinka/

https://learn.adafruit.com/circuitpython-on-raspberrypi-linux/

SPI

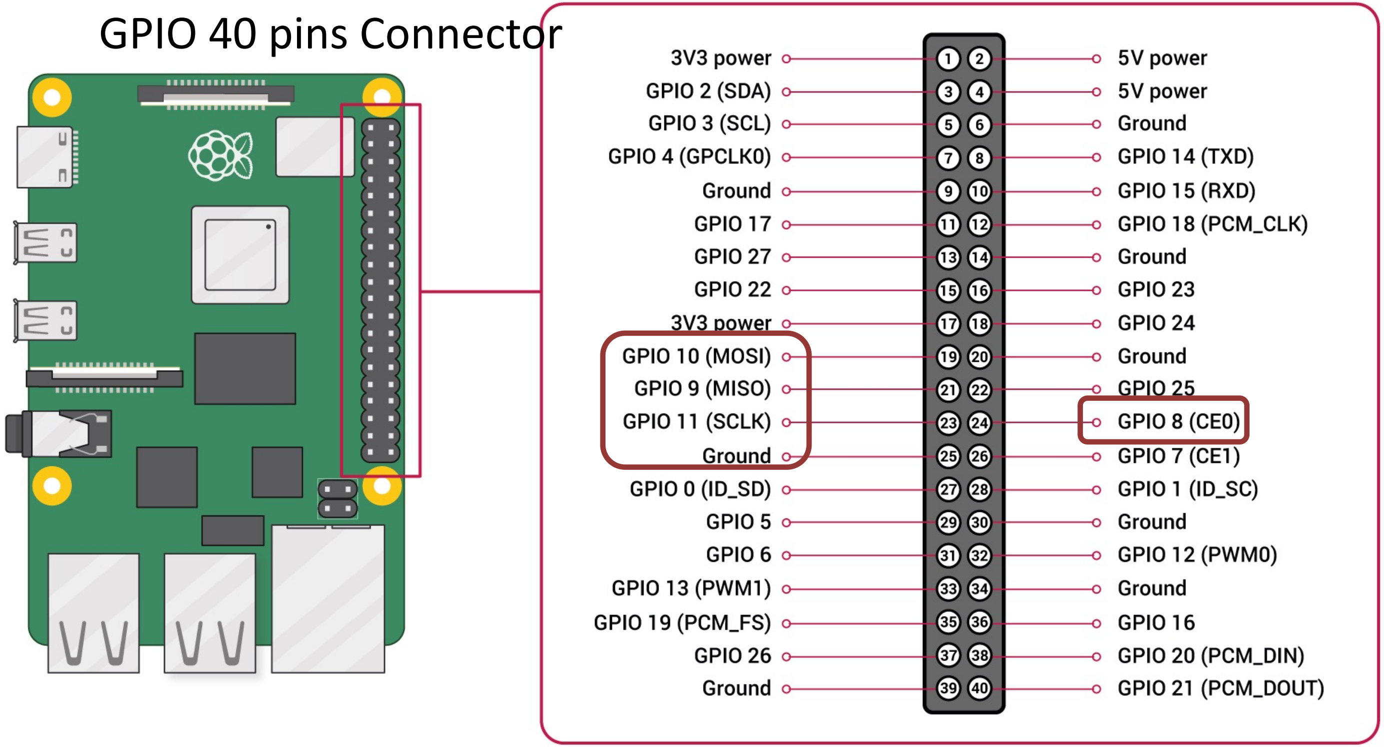
SPI

- Serial Peripheral Interface (SPI)
- SPI is an interface to communicate with different types of electronic components like Sensors, Analog to Digital Converts (ADC), etc. that supports the SPI interface
- Thousands of different Components and Sensors supports the SPI interface

https://www.raspberrypi.org/documentation/hardware/raspberrypi/spi/

SPI Wiring on Raspberry Pi

GPIO 40 pins Connector



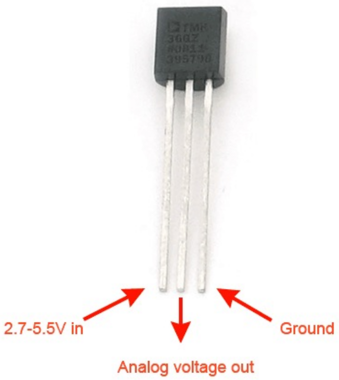
3V3 power	1	2	5V power
GPIO 2 (SDA)	3	4	5V power
GPIO 3 (SCL)	5	6	Ground
GPIO 4 (GPCLK0)	7	8	GPIO 14 (TXD)
Ground	9	10	GPIO 15 (RXD)
GPIO 17	11	12	GPIO 18 (PCM_CLK)
GPIO 27	13	14	Ground
GPIO 22	15	16	GPIO 23
3V3 power	17	18	GPIO 24
GPIO 10 (MOSI)	19	20	Ground
GPIO 9 (MISO)	21	22	GPIO 25
GPIO 11 (SCLK)	23	24	GPIO 8 (CE0)
Ground	25	26	GPIO 7 (CE1)
GPIO 0 (ID_SD)	27	28	GPIO 1 (ID_SC)
GPIO 5	29	30	Ground
GPIO 6	31	32	GPIO 12 (PWM0)
GPIO 13 (PWM1)	33	34	Ground
GPIO 19 (PCM_FS)	35	36	GPIO 16
GPIO 26	37	38	GPIO 20 (PCM_DIN)
Ground	39	40	GPIO 21 (PCM_DOUT)

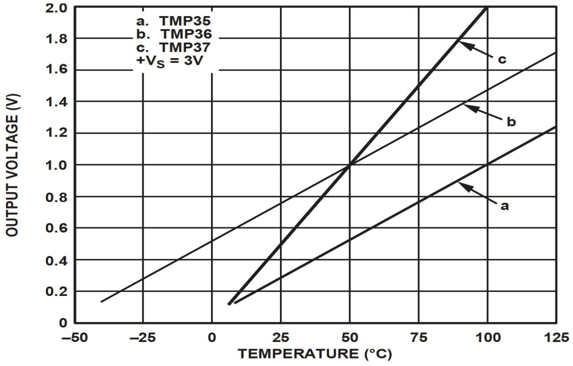
Analog Temperature Sensors

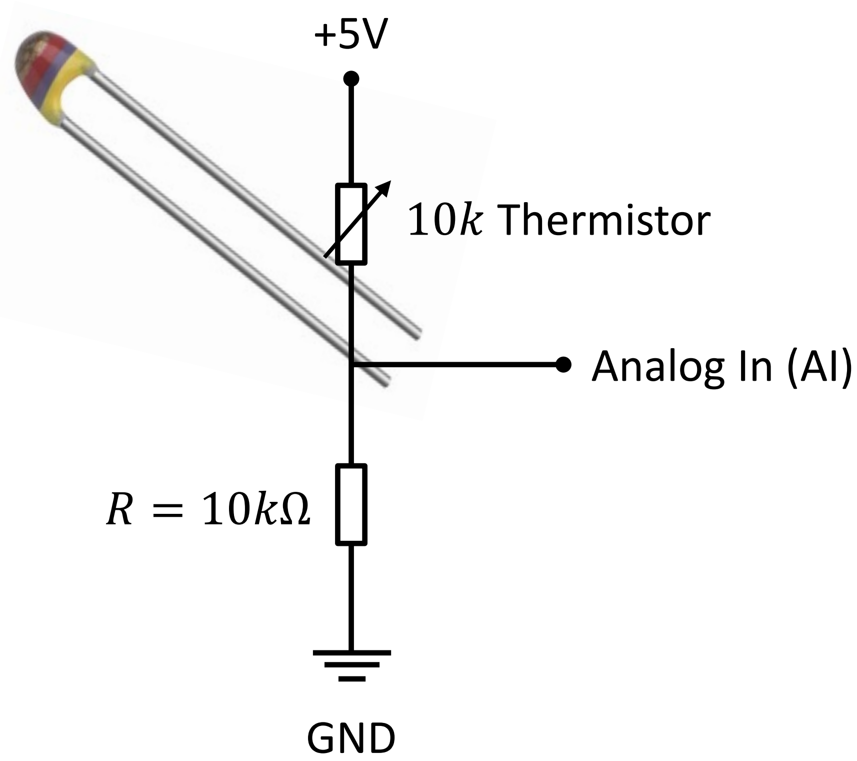
TMP36 Temperature Sensor

10k Thermistor Temperature Sensor

Note! Raspberry Pi has no Analog In Channels. You need to use an external ADC







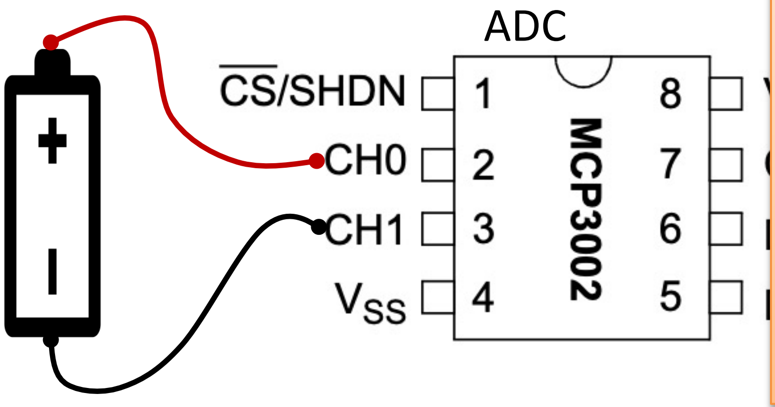
Example: Read Data from ADC

The **MCP3002** is a 10-bit analog to digital converter with 2 channels (0-1).

For test purpose we start by wiring a 1.5V Battery to the CH0 (+) and CH1(-) pins on the ADC

Note! WE have set **differential=True** (meaning CH0 is “+“ and CH1 is “-“)

1.5V Battery



```
from gpiozero import MCP3002
from time import sleep

adc = MCP3002(channel=0, differential=True)

N = 20

for x in range(N):
    adcdata = adc.value #Value between 0 and 1
    #print(adcdata)
    voltvalue = adcdata * 5 #Value 0-5V
    print(voltvalue)
    sleep(1)
```

Measure temperature with an ADC

TMP36 Temperature Sensor

Wire a TMP36 temperature sensor to the first channel of an MCP3002 analog to digital converter and the other pins to +5V and GND

```
from gpiozero import MCP3002
from time import sleep

adc = MCP3002(channel=0, differential=False)

N = 10

for x in range(N):
    adcdata = adc.value #Value between 0 and 1
    #print(adcdata)

    voltvalue = adcdata * 5 #Value between 0V and 5V
    #print(voltvalue)

    tempC = 100*voltvalue-50 #Temperature in Celsius
    tempc = round(tempC,1)
    print(tempC)

    sleep(1)
```

I2C

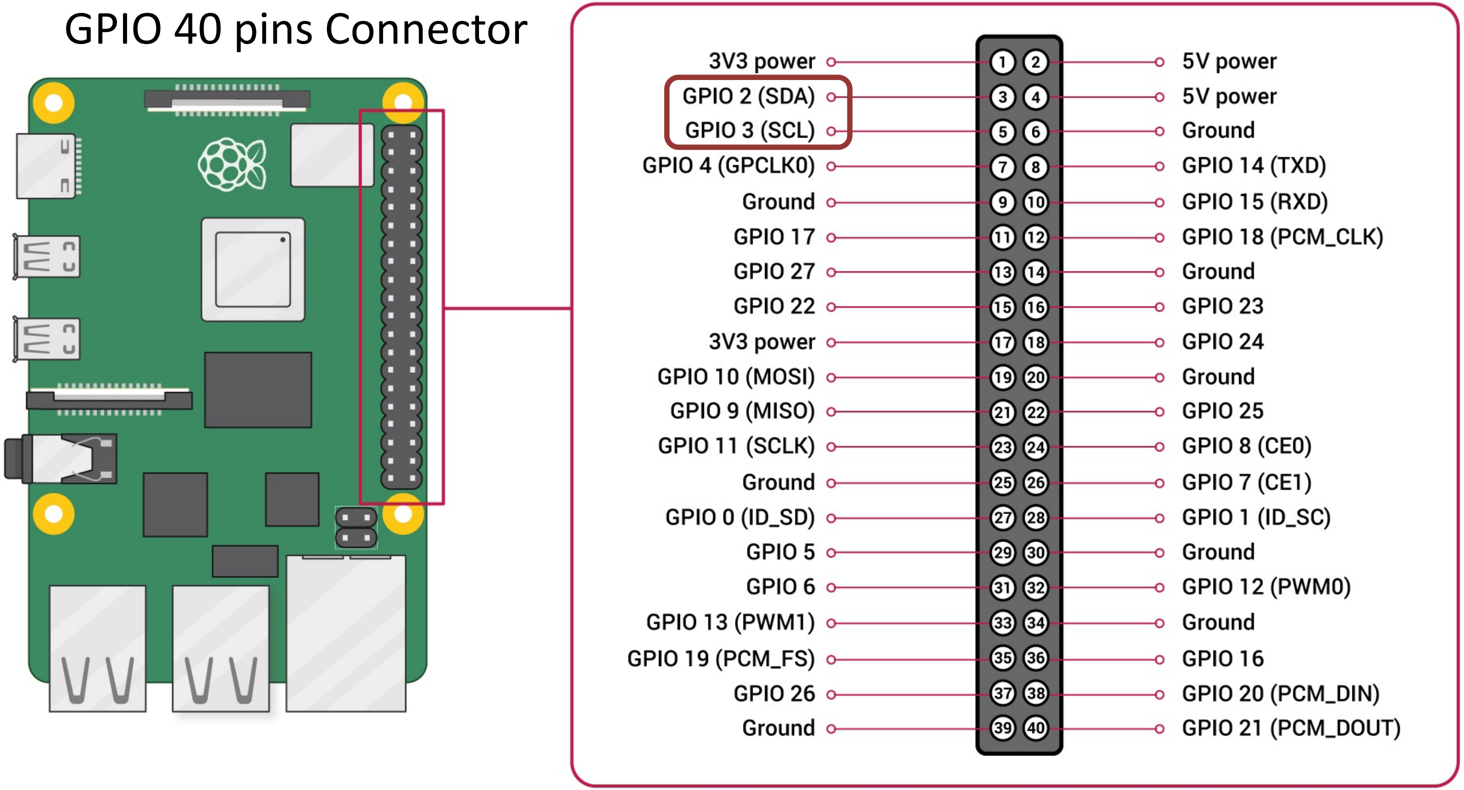
Table of Contents

I2C

- I2C is a multi-drop bus
- 2-Wire Protocol (SCL + SDA)
- Multiple devices can be connected to the I2C pins on the Raspberry Pi
- Each device has its own unique I2C address

I2C Wiring on Raspberry Pi

GPIO 40 pins Connector



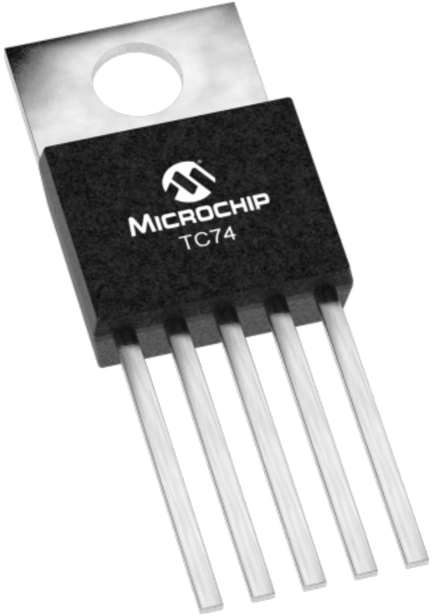
3V3 power	1	2	5V power
GPIO 2 (SDA)	3	4	5V power
GPIO 3 (SCL)	5	6	Ground
GPIO 4 (GPCLK0)	7	8	GPIO 14 (TXD)
Ground	9	10	GPIO 15 (RXD)
GPIO 17	11	12	GPIO 18 (PCM_CLK)
GPIO 27	13	14	Ground
GPIO 22	15	16	GPIO 23
3V3 power	17	18	GPIO 24
GPIO 10 (MOSI)	19	20	Ground
GPIO 9 (MISO)	21	22	GPIO 25
GPIO 11 (SCLK)	23	24	GPIO 8 (CE0)
Ground	25	26	GPIO 7 (CE1)
GPIO 0 (ID_SD)	27	28	GPIO 1 (ID_SC)
GPIO 5	29	30	Ground
GPIO 6	31	32	GPIO 12 (PWM0)
GPIO 13 (PWM1)	33	34	Ground
GPIO 19 (PCM_FS)	35	36	GPIO 16
GPIO 26	37	38	GPIO 20 (PCM_DIN)
Ground	39	40	GPIO 21 (PCM_DOUT)

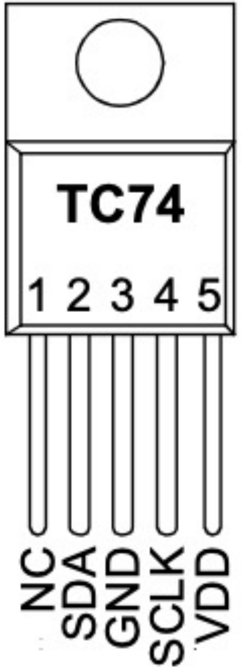
Note! The I2C pins include a fixed 1.8 kΩ pull-up resistor to 3.3v.

TC74 Temperature Sensor

SMBus/I2C Interface

TC74A0-5.0VAT





- The TC74 acquires and converts temperature information from its onboard solid-state sensor with a resolution of ±1°C.
- It stores the data in an internal register which is then read through the serial port.
- The system interface is a slave SMBus/I2C port, through which temperature data can be read at any time.

Datasheet: https://ww1.microchip.com/downloads/en/DeviceDoc/21462D.pdf

TC74 Python Code Example

This code shows the basic reading of the Sensor Data.

You can add a For Loop or a While Loop for reading Sensor Data at specific intervals.

You can plot the Data using matplotlib, save data to a File or send data to a cloud service like ThingSpeak, etc.

```
import smbus

channel = 1
address = 0x48

bus = smbus.SMBus(channel)

data = bus.read_byte_data(address, 0)
print(data)
```

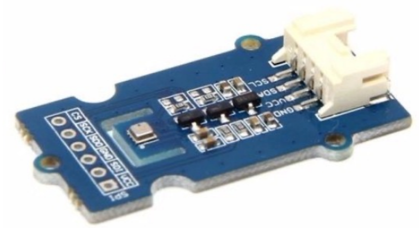
Or just:

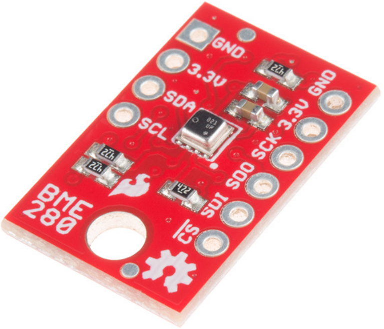
```
data = bus.read_byte(address)
print(data)
```

This gives the Temperature Value in Degrees Celsius, e.g., `22`

BME280

- BME280 is a Digital Humidity, Pressure and Temperature Sensor from Bosch
- The sensor provides both SPI and I2C interfaces
- Adafruit, Grove Seeed, SparkFun, etc. have breakout board bords for easy connection to Arduino, Raspberry Pi, etc.





BME280 Python Example

https://circuitpython.readthedocs.io/projects/bme280/en/latest/

```
import time
import board
import busio
import adafruit_bme280

# Create library object using our Bus I2C port
i2c = busio.I2C(board.SCL, board.SDA)
bme280 = adafruit_bme280.Adafruit_BME280_I2C(i2c)

# OR create library object using our Bus SPI port
# spi = busio.SPI(board.SCK, board.MOSI, board.MISO)
# bme_cs = digitalio.DigitalInOut(board.D10)
# bme280 = adafruit_bme280.Adafruit_BME280_SPI(spi, bme_cs)

# change this to match the location's pressure (hPa) at sea level
bme280.sea_level_pressure = 1013.25

while True:
    print("\nTemperature: %0.1f C" % bme280.temperature)
    print("Humidity: %0.1f %%" % bme280.relative_humidity)
    print("Pressure: %0.1f hPa" % bme280.pressure)
    print("Altitude = %0.2f meters" % bme280.altitude)
    time.sleep(2)
```

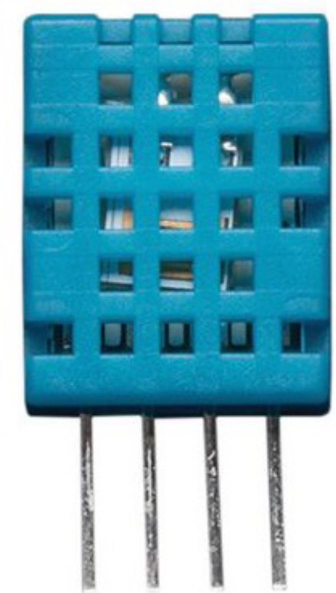
1-Wire

DHT11/DHT22

They are Breadboard friendly and easy to wire. They use a single-wire to send data.

DHT11

- Good for 20-80% humidity readings with 5% accuracy
- Good for 0-50°C temperature readings ±2°C accuracy
- 1 Hz sampling rate (once every second)
- Price: a few bucks



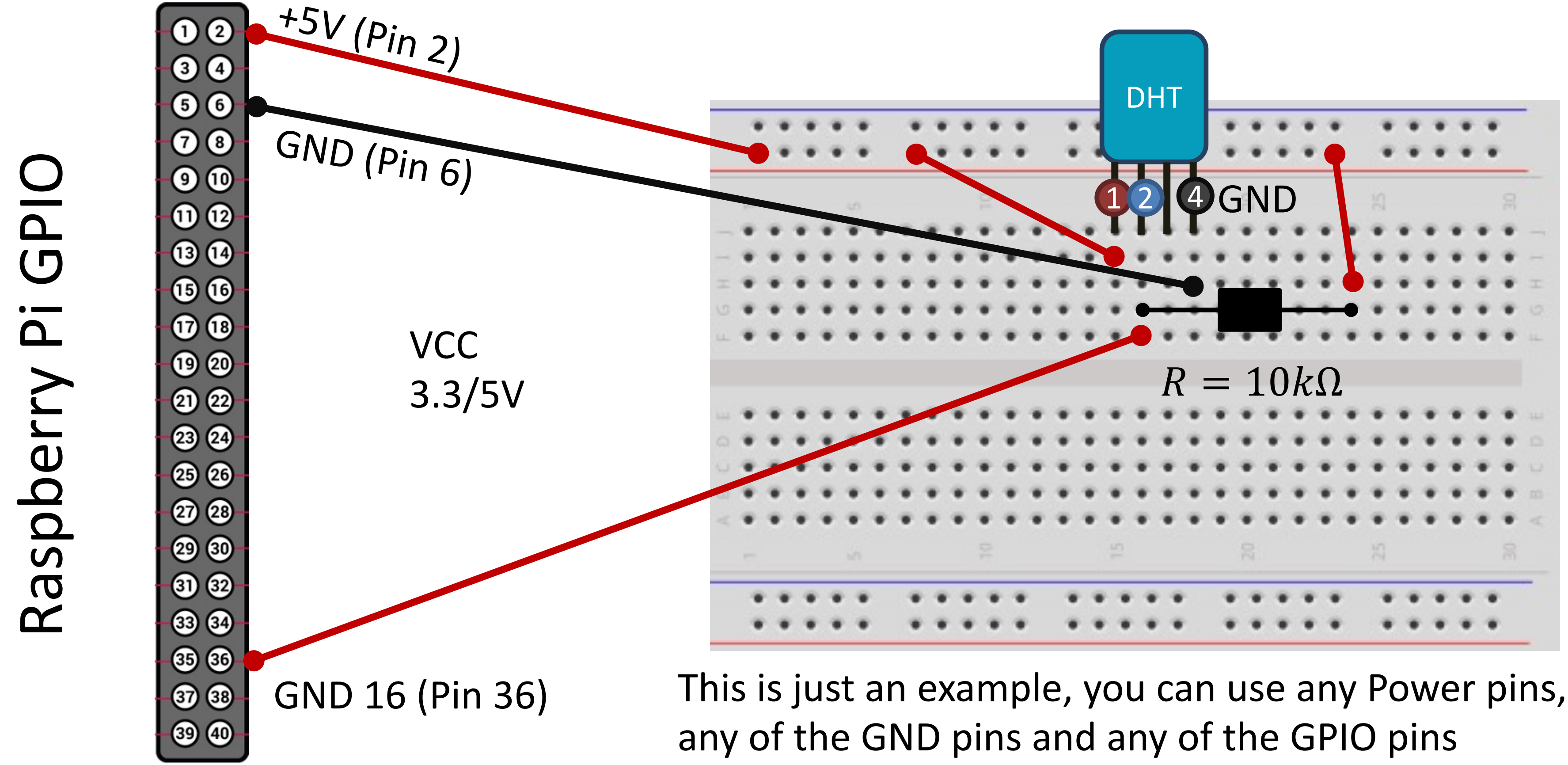
DHT22

DHT22 is more precise, more accurate and works in a bigger range of temperature and humidity, but its larger and more expensive

- 0-100% RH
- -40-125°C

Typically you need a 4.7K or 10K resistor, which you will want to use as a pullup from the data pin to VCC. This is included in the package

DHT11/DHT22



This is just an example, you can use any Power pins, any of the GND pins and any of the GPIO pins

DHT11/DHT22 Python Example

```
import time
import board
import adafruit_dht

dhtDevice = adafruit_dht.DHT22(board.D18, use_pulseio=False)

while True:
    try:
        temperature_c = dhtDevice.temperature
        humidity = dhtDevice.humidity
        print(
            "Temp: {:.1f} C    Humidity: {}% ".format(
            temperature_c, humidity
            )
        )

    except RuntimeError as error:
        # Errors happen fairly often, DHT's are hard to read, just keep going
        print(error.args[0])
        time.sleep(2.0)
        continue
    except Exception as error:
        dhtDevice.exit()
        raise error

    time.sleep(2.0)
```

Errors happen fairly often, DHT's are hard to read because it needs precise timing. That’s why you should use **try** in your code

https://learn.adafruit.com/dht-humidity-sensing-on-raspberry-pi-with-gdocs-logging/python-setup

NoSQL and MongoDB

MongoDB

- MongoDB is a cross-platform document-oriented database program.
- MongoDB is a NoSQL database program
- MongoDB uses JSON-like documents
- Home Page: https://www.mongodb.com/

Software:

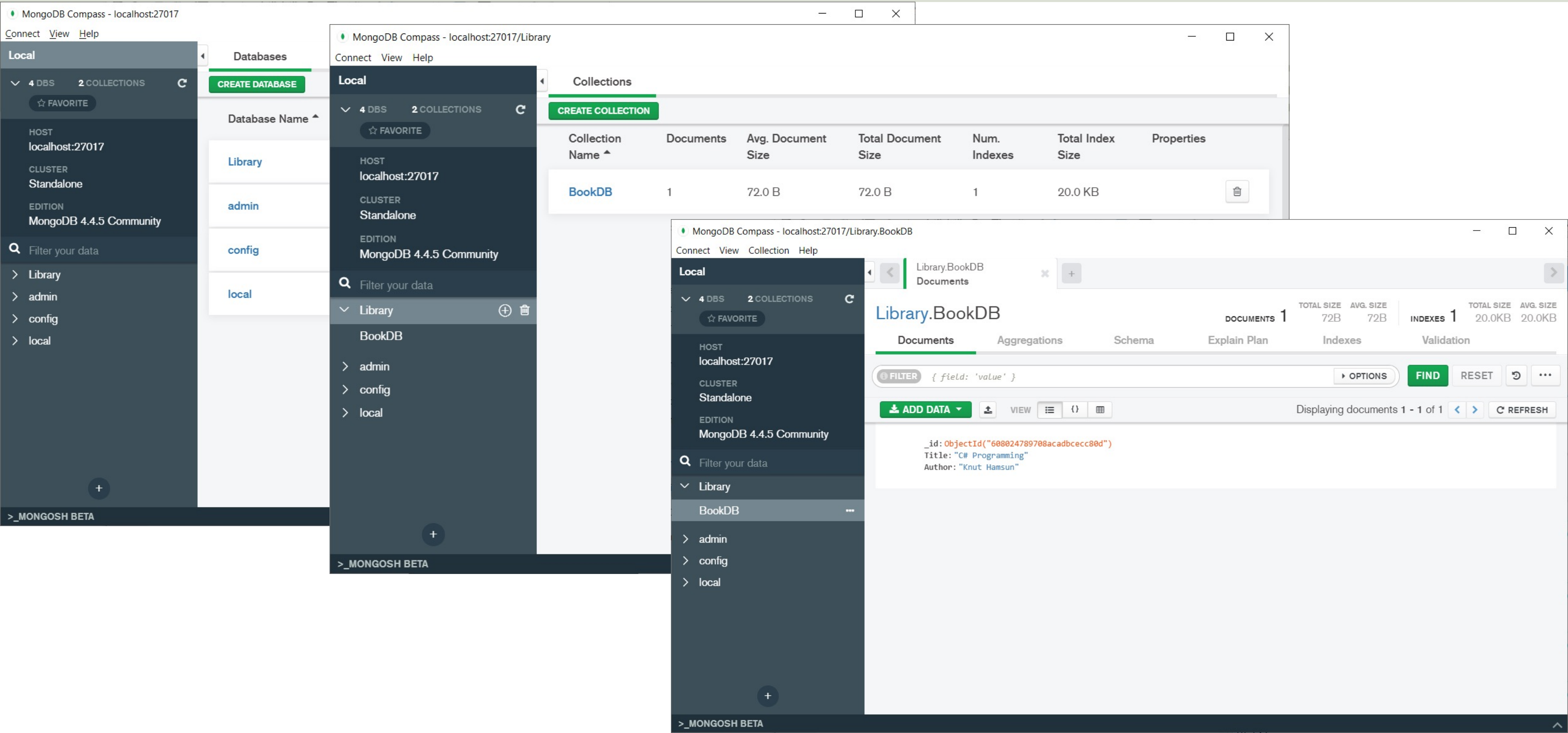
- MongoDB Community Server – Free version of the MongoDB Server which can be installed locally on your computer or a server
- MongoDB Atlas – Premade MongoDB ready to use in the Cloud
- MongoDB Compass – GUI for connecting to and manipulating your MongoDB database
- PyMongo – MongoDB Driver for Python

SQL vs MongoDB

Note the following:

- A **collection** in MongoDB is the same as a table in SQL databases.
- A **document** in MongoDB is the same as a record in SQL databases.

MongoDB Compass



HOST
localhost:27017

CLUSTER
Standalone

EDITION
MongoDB 4.4.5 Community

Collection Name	Documents	Avg. Document Size	Total Document Size	Num. Indexes	Total Index Size	Properties
BookDB	1	72.0 B	72.0 B	1	20.0 KB	

Library.BookDB

DOCUMENTS 1 TOTAL SIZE 72B AVG. SIZE 72B INDEXES 1 TOTAL SIZE 20.0KB AVG. SIZE 20.0KB

Displaying documents 1 - 1 of 1

```
_id: ObjectId("608024789708acadbcecc80d")
Title: "C# Programming"
Author: "Knut Hamsun"
```

PyMongo

- The PyMongo package contains tools for interacting with MongoDB database from Python
- The PyMongo package is a native Python driver for MongoDB
- Install using PIP: pip install pymongo
- https://pypi.org/project/pymongo/

Python

Python script that creates a Database (“Library”), a Collection (“BookDB”) and a Document.

In a SQL database we use the INSERT statement to insert data in a table.

In MongoDB we use the **insert_one()** and **insert_many()** methods to insert data into a collection.

```
import pymongo

client = pymongo.MongoClient("mongodb://localhost:27017/")
database = client["Library"]
collection = database["Book"]

document = { "Title": "C# Programming", "Author": "Knut Hamsun" }

x = collection.insert_one(document)
```

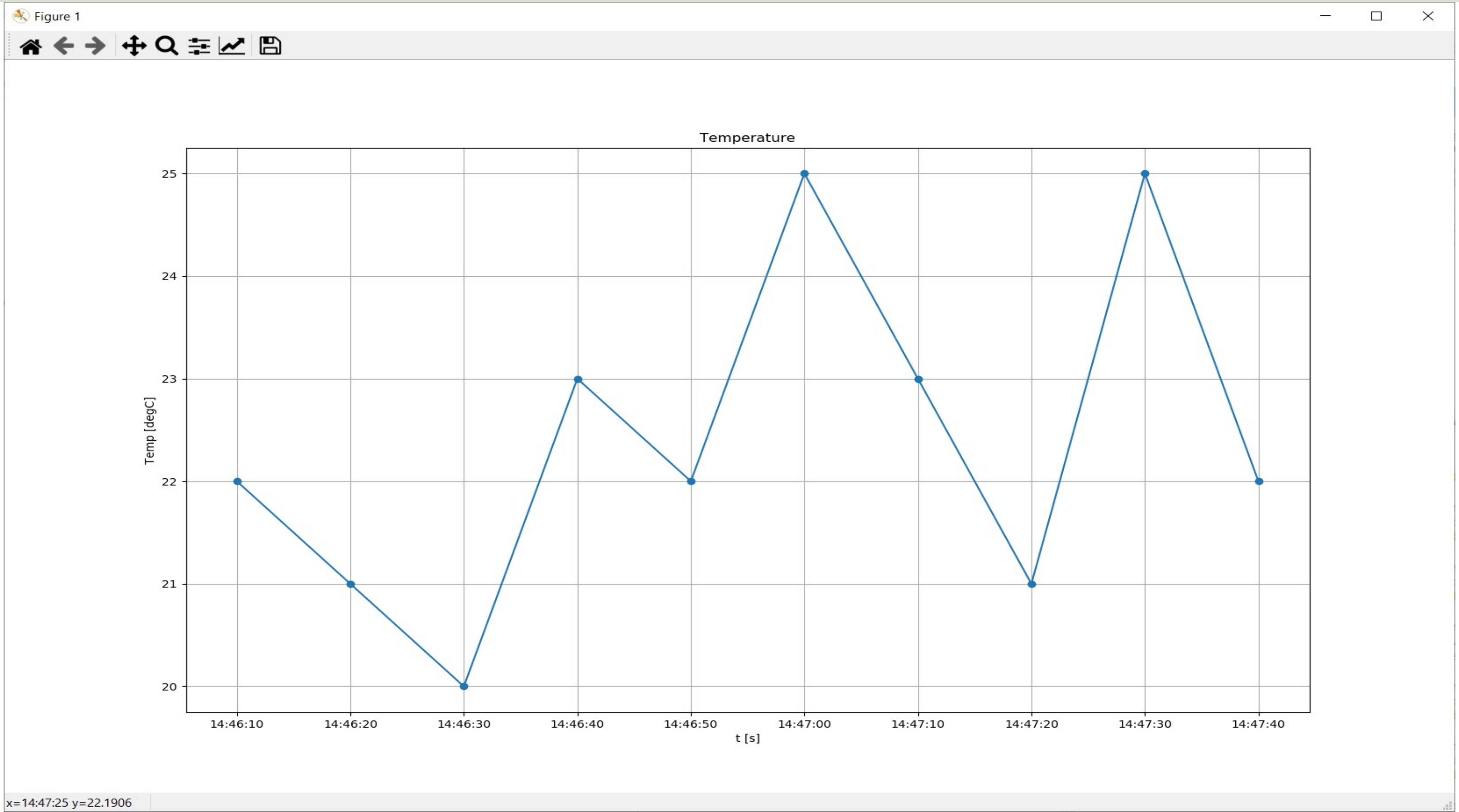
Logging Data Example

```
import pymongo
import random
import time
from datetime import datetime

# Create Database
client = pymongo.MongoClient("mongodb://localhost:27017/")
database = client["MeasurementSystem"]
collection = database["MeasurementData"]

Ts = 10 # Sampling Time
N = 10
for k in range(N):
    # Generate Random Data
    LowLimit = 20
    UpperLimit = 25
    MeasurementValue = random.randint(LowLimit, UpperLimit)

    #Find Date and Time
    now = datetime.now()
    datetimeformat = "%Y-%m-%d %H:%M:%S"
    MeasurementDateTime = now.strftime(datetimeformat)

    # Insert Data into Database
    document = { "MeasurementValue": MeasurementValue, "MeasurementDateTime":
MeasurementDateTime }
    x = collection.insert_one(document)

    # Wait
    time.sleep(Ts)
```

Plotting Data

```python
import pymongo
import matplotlib.pyplot as plt
from datetime import datetime

# Connect to Database
client = pymongo.MongoClient("mongodb://localhost:27017/")
database = client["MeasurementSystem"]
collection = database["MeasurementData"]

t = []
data = []

# Retrieving and Formatting Data
for document in collection.find():
    MeasurementValue = document["MeasurementValue"]
    MeasurementDateTime = document["MeasurementDateTime"]

    timeformat = "%Y-%m-%d %H:%M:%S"
    MeasurementDateTime = datetime.strptime(MeasurementDateTime, timeformat)

    data.append(MeasurementValue)
    t.append(MeasurementDateTime)

# Plotting
plt.plot(t, data, 'o-')
plt.title('Temperature')
plt.xlabel('t [s]')
plt.ylabel('Temp [degC]')
plt.grid()
plt.show()
```

Plotted Data



ThingSpeak

Table of Contents

ThingSpeak

- ThingSpeak is an IoT analytics platform service that lets you collect and store sensor data in the cloud and develop Internet of Things applications.
- ThingSpeak has a free Web Service (REST API) that lets you collect and store sensor data in the cloud and develop Internet of Things applications.
- It works with Arduino, Raspberry Pi, MATLAB and LabVIEW, Python, etc.

https://thingspeak.com

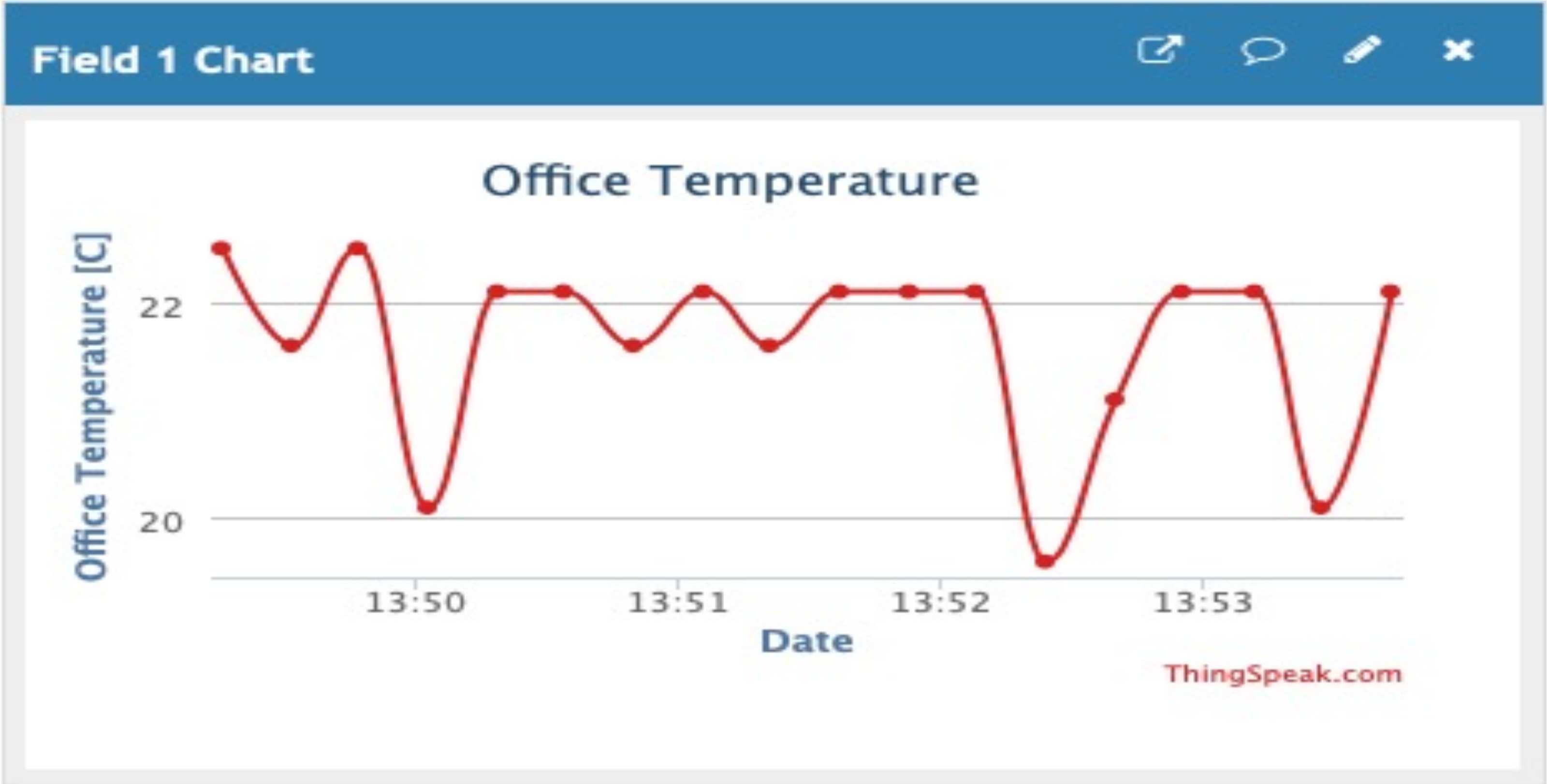
Write TMP36 Data

A Free ThingSpeak Channel can only be updated every 15 sec

```
import thingspeak
import time
from gpiozero import MCP3002

adc = MCP3002(channel=0, differential=False)

channel_id = xxxxxxx
write_key  = ”xxxxxxxxxxxxxxxxxx”

channel = thingspeak.Channel(id=channel_id, api_key=write_key)

N = 10
for x in range(N):
    #Get Sensor Data
    adcdata = adc.value #Scaled Value between 0 and 1
    voltvalue = adcdata * 5 # Value between 0V and 5V
    tempC = 100*voltvalue-50 # Temperature in Celsius
    tempC = round(tempC,1)
    print(tempC)

    #Write to ThingSpeak
    response = channel.update({'field1': tempC})
    time.sleep(15)
```

Write TMP36 Data

Here we see the Temperature Data in ThingSpeak:



MQTT

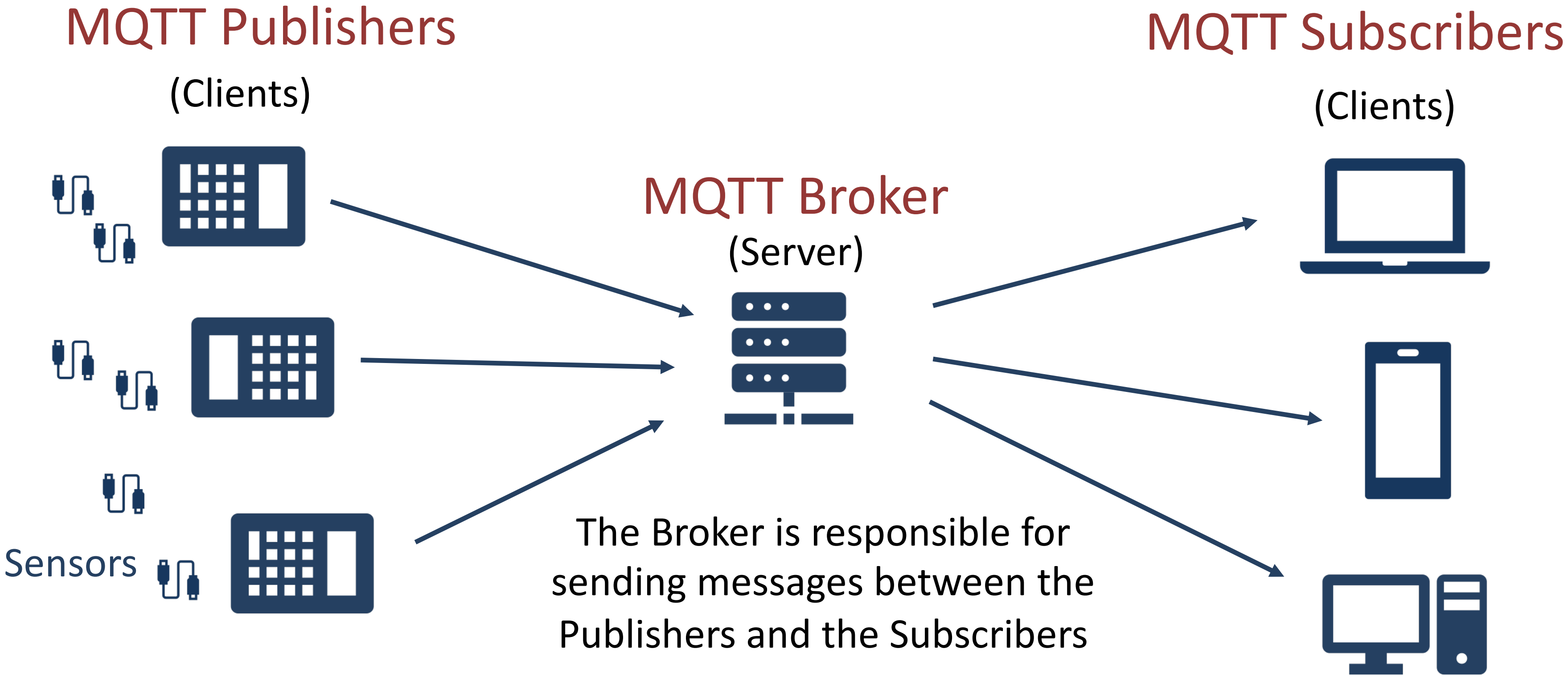
MQTT

MQTT is a popular IoT Communication Protocol

- Message Queueing Telemetry Transport (MQTT) is an IoT connectivity protocol
- MQTT is used in applications with thousands of sensors
- MQTT is efficient in terms of bandwidth, battery, and resources
- MQTT uses a publish/subscribe model
- MQTT can be implemented using standard HTTP calls
- M2M (machine to machine) Communication

https://mqtt.org/

MQTT Scenario



MQTT Publishers

(Clients)

Sensors

MQTT Broker

(Server)

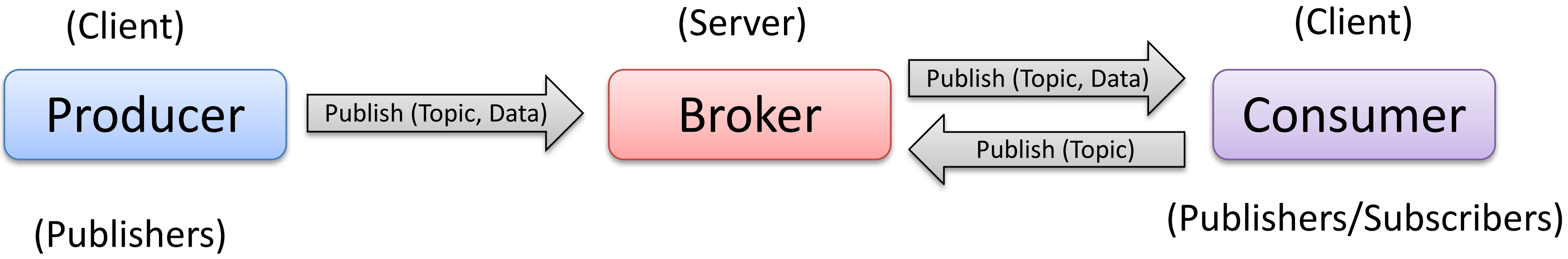
MQTT Subscribers

(Clients)

The Broker is responsible for sending messages between the Publishers and the Subscribers

Publish/Subscribe Model

Typically, we have what we call **Producers** (Publishers), and we have **Consumers**, which can be both Publishers and Subscribers.



(Client)

Producer

(Publishers)

Publish (Topic, Data)

(Server)

Broker

Publish (Topic, Data)

Publish (Topic)

(Client)

Consumer

(Publishers/Subscribers)

An MQTT Client Publishes
a Message to the Broker

Other Clients can Subscribe to
the Broker to receive Messages

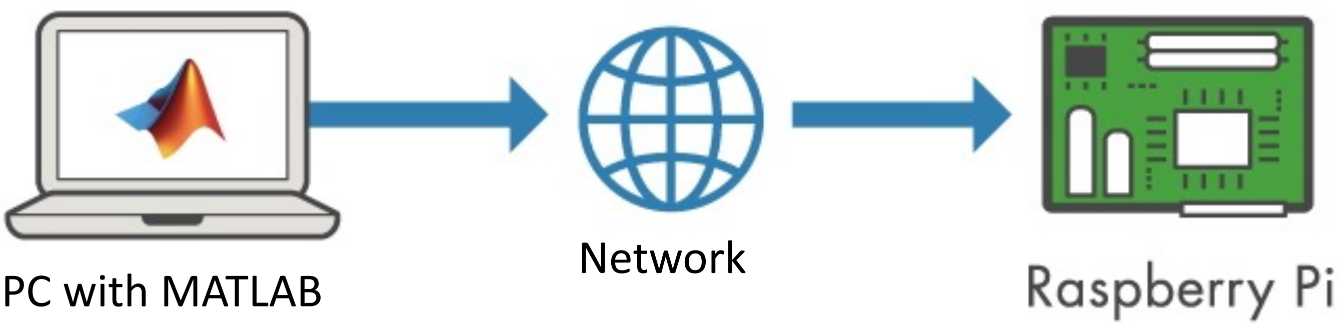
https://www.halvorsen.blog

Raspberry Pi with MATLAB

Hans-Petter Halvorsen

Table of Contents

Raspberry Pi + MATLAB



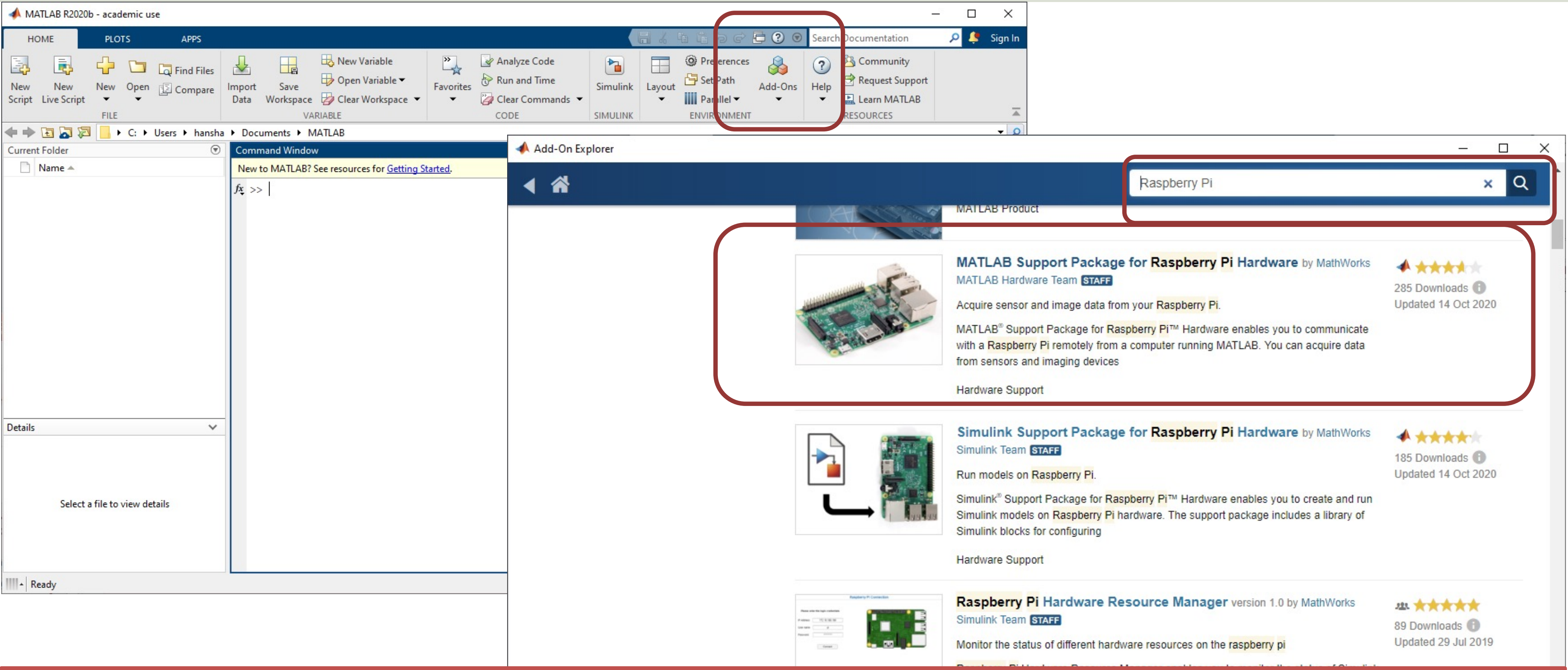
PC with MATLAB

Network

With MATLAB support package for Raspberry Pi, the Raspberry Pi is connected to a computer running MATLAB. Processing is done on the computer with MATLAB.

https://mathworks.com/hardware-support/raspberry-pi-matlab.html

MATLAB Support Package for Raspberry Pi



New to MATLAB? See resources for Getting Started.

fx >>

Select a file to view details

MATLAB Support Package for Raspberry Pi Hardware by MathWorks
MATLAB Hardware Team STAFF

Acquire sensor and image data from your Raspberry Pi.

MATLAB® Support Package for Raspberry Pi™ Hardware enables you to communicate with a Raspberry Pi remotely from a computer running MATLAB. You can acquire data from sensors and imaging devices

Hardware Support

285 Downloads
Updated 14 Oct 2020

Simulink Support Package for Raspberry Pi Hardware by MathWorks
Simulink Team STAFF

Run models on Raspberry Pi.

Simulink® Support Package for Raspberry Pi™ Hardware enables you to create and run Simulink models on Raspberry Pi hardware. The support package includes a library of Simulink blocks for configuring

Hardware Support

185 Downloads
Updated 14 Oct 2020

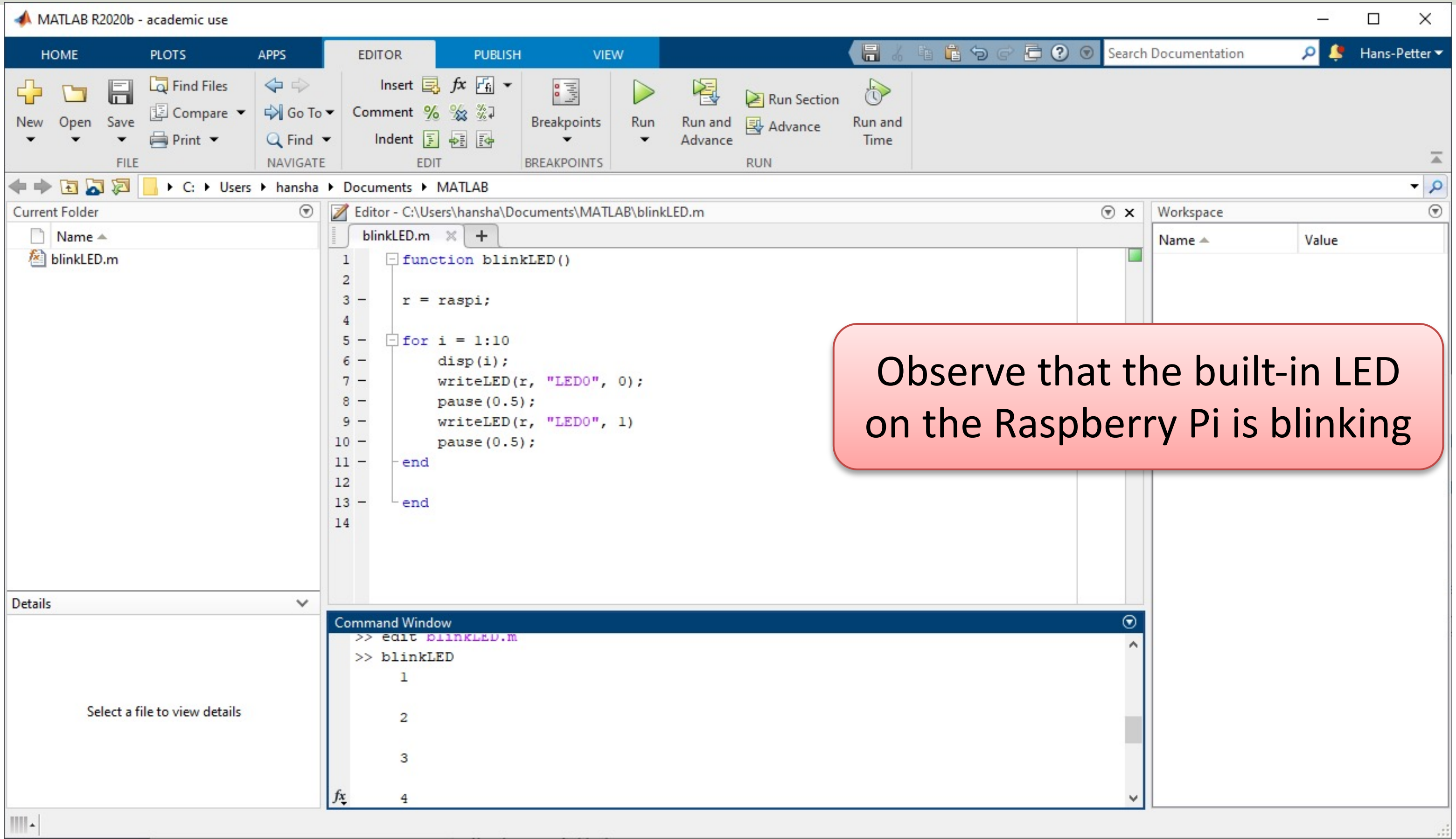
Raspberry Pi Hardware Resource Manager version 1.0 by MathWorks
Simulink Team STAFF

Monitor the status of different hardware resources on the raspberry pi

89 Downloads
Updated 29 Jul 2019

Getting Started with MATLAB Support Package for Raspberry Pi: https://youtu.be/32ByiUdOwsw

MATLAB Example



```
function blinkLED()

r = raspi;

for i = 1:10
    disp(i);
    writeLED(r, "LED0", 0);
    pause(0.5);
    writeLED(r, "LED0", 1)
    pause(0.5);
end

end
```

```
>> edit blinkLED.m
>> blinkLED
     1

     2

     3

     4
```

Observe that the built-in LED on the Raspberry Pi is blinking

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

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

Web: https://www.halvorsen.blog