CSCi 4907
Introduction to IoT and Edge Computing Applications

Prof. Kartik Bulusu, CS Dept.

Week 9 [03/22/2024]	• Pending midterm project demos and grading • Recap: 5 Layer IoT Architecture • Guest lecture by **Rob Shaughnessy,** CEO Pysmetis Inc.	• In-class Raspberry Pi Lab with ESP32 microcontroller • Working with MicroPython in Thonny IDE

```
git clone git@github.com:gwu-csci3907/Spring2024.git

git clone https://github.com/gwu-csci3907/Spring2024.git
```

GW School of Engineering & Applied Science
THE GEORGE WASHINGTON UNIVERSITY

Spring 2024

Photo: Kartik Bulusu

Midterm project demonstration - Status update

Midterm Project Status

Name	Project title	Hardware requirements	Status
Aleks Haskett	*Home Security System*	Pi Camera, Tracking sensor, Passive buzzer	**Approved. Needs to collect sensors / Demo completed /**
Gerald Fattah	*Smart Animal Capture System*	IR Sensor, Pyroelectric ("Passive") InfraRed (PIR) module, Pi Camera	**Approved. Needs to collect sensors / Demo completed /**
Jonathan Pang	*Proximity Alarm Door System (PADS)*	Pyroelectric ("Passive") InfraRed (PIR) module (HC-SR501), Bluetooth Tranceiver Module	**Approved. Needs to collect sensors / Demo completed /**
Oliver Kristeya	*Dungeons & Dragans (D&D) Tower Roller*	Pi Camera, 3.5 in touch screen	**Need more information on 02/23, Approved on 02/26, Needs to collect sensors**
Talia Novack	*Dish Washer Helper*	Touch switch, analog heat sensor	**Approved. Need make and model numbers of the sensors / Demo completed /**
Warren Nguyen	*Adaptive Lamp*	SenseHat, Photoresistor, Dimmable light sources	**Approved. Need make and model numbers of the sensors / Demo completed /**
Selman Eris	*Food Scanner*	Pi Camera	**Approved. Needs to collect sensors / Demo completed /**
Matthew Gouvin	*Plant Lighting Measurement Device*	Light sensors	**Approved. Need make and model numbers of the sensors / Demo completed /**
Liza Mozolyuk	*Flight Tracking Interface*	SenseHat	**Need more information on 02/23, Approved on 02/26, Needs to collect sensors / Demo completed /**
Bridget Orr	*Ukelele Tuner*	Sound sensor	**Approved. Need make and model numbers of the sensors / Demo completed /**
Georgiana Mois	*MediTrack: Smark Medication Management*	Tilt Switch, Vibration Swtich	**Approved. Need make and model numbers of the sensors / Demo completed /**
Alicia Ha	*Home Security Camera and Doorbell System*	Ultrasonic sensor, Pi Camera, PIR motion sensor, RGB LED, Passive Buzzer, Button	**Approved. Need make and model numbers of the sensors / Demo completed /**
William Mai	*Cat Detector*	Pi NOIR Camera	**Approved. Needs to collect sensors / Demo completed /**
Peter Wright	*Smart Cat Feeder*	Weight and Optical Sensor, actuator	**Approved. Need make and model numbers of the sensors / Demo completed /**
Abdulrahman Alsaleh	*Camera by sensor detection*	Pi Camera, PIR motion sensor or Ultrasonic sensor,	**Approved. Need make and model numbers of the sensors / Demo completed /**
Alvin Isaac	*Water Detection System*	Temperature, humidity and water level sensor	**Need more information on 02/23, Approved on 02/26, Needs to collect sensors / Demo completed /**

Topics to be covered today

Hardware:
- ESP32

Deliverable assigned this week:
1. One-page summary of the guest-lecture
2. Final project proposal

Expectations on student deliverables:
1. Weekly HW and / or Quizzes
2. Final project proposal
3. Final project presentation
4. Final project demo
5. Final report in a conference-style template

Topics to be covered today

Hardware:
- ESP32

Edge computing on the Pi:
Basic stats + Python

Deliverable assigned this week:
1. One-page summary of the guest-lecture
2. Final project proposal

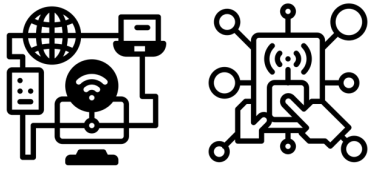
Expectations on student deliverables:
1. Weekly HW and / or Quizzes
2. Final project proposal
3. Final project presentation
4. Final project demo
5. Final report in a conference-style template

Final project proposal submissions – opened

Modifying the 5 layer IoT Architecture

Business-layer

Information-layer



Application

Processing- or
Middleware-layer /
Edge compute layer

Data processing

Communication-layer

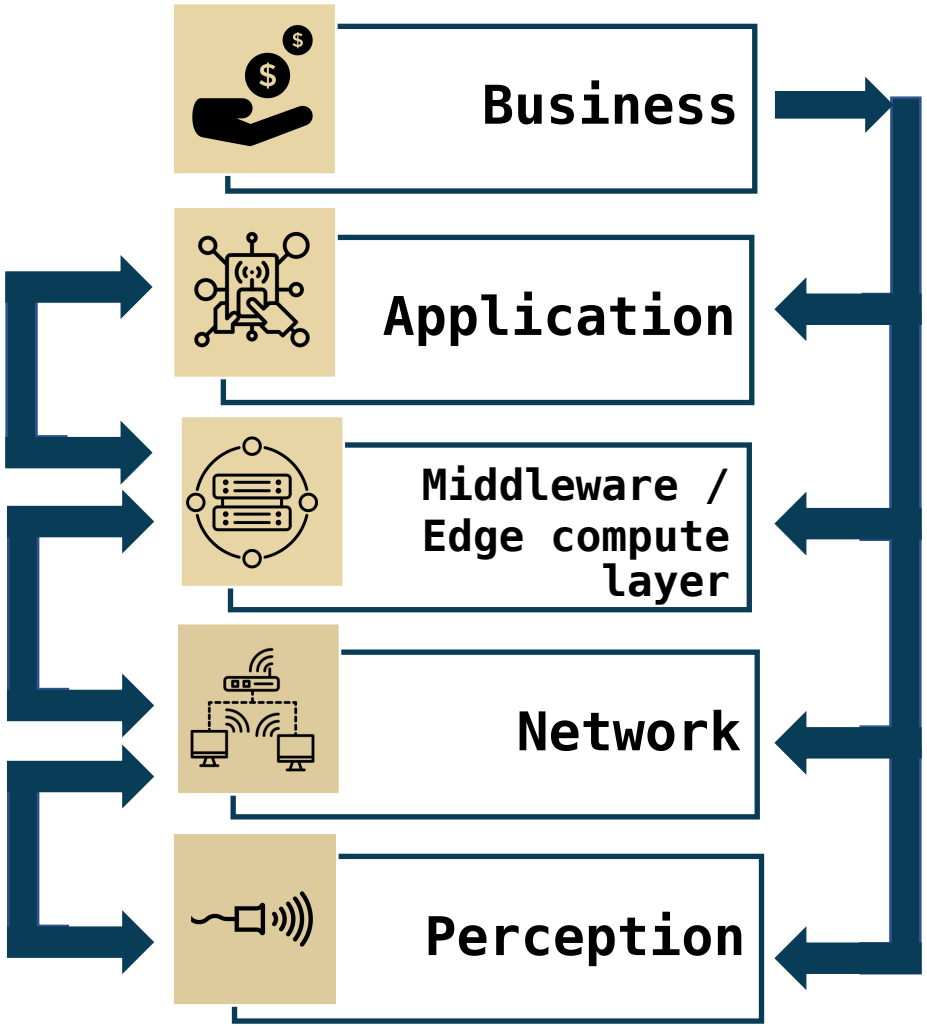
Data transfer

Sensor-layer

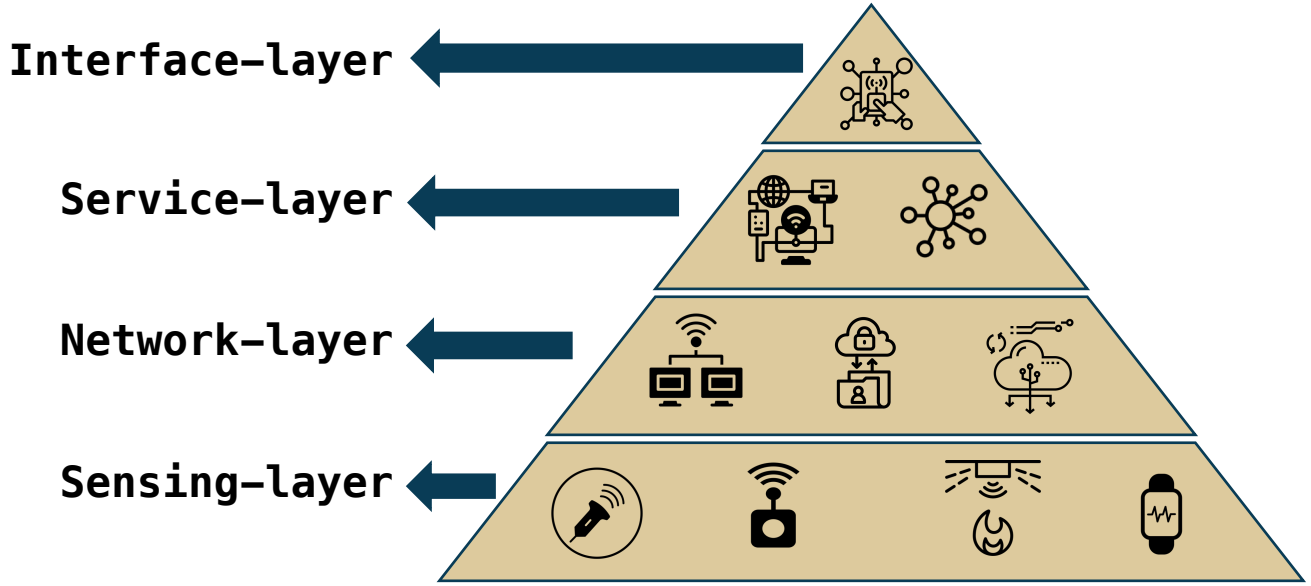
Things

Icon Sources:
sensor by Carolina Cani:, sensor by Pham Duy Phuong Hung, sensor by Tippawan Sookruay, sensor by Lorenzo: https://thenounproject.com/browse/icons/term/sensor
fire sensor by LAFS : https://thenounproject.com/browse/icons/term/fire-sensor/
Ultrasound by Shocho: https://thenounproject.com/browse/icons/term/ultrasound/
Network by Solikin:, Network by Tippawan: https://thenounproject.com/browse/icons/term/network
application by Chaowalit Koetchuea: https://thenounproject.com/browse/icons/term/application
wifi network by ProSymbols: https://thenounproject.com/browse/icons/term/wifi-network/
data transfer by Jajang Nurrahman: https://thenounproject.com/browse/icons/term/data-transfer/
transfer data by tezar tantular: https://thenounproject.com/browse/icons/term/transfer-data/
data processing by Jajang Nurrahman: https://thenounproject.com/browse/icons/term/data-processing
Business by DinosoftLab: https://thenounproject.com/browse/icons/term/business/

The 5-Layer IoT Architecture



Service-oriented IoT Architecture



Sources:
sensor by Carolina Cani:, sensor by Pham Duy Phuong Hung, sensor by Tippawan Sookruay, sensor by Lorenzo: https://thenounproject.com/browse/icons/term/sensor
wifi network by Matthias Hartmann:: https://thenounproject.com/browse/icons/term/wifi-network/
application by Chaowalit Koetchuea: https://thenounproject.com/browse/icons/term/application/
IoT Architecture layers: https://www.startertutorials.com/blog/iot-architecture-layers.html

Differences Between a Microcontroller and a Microprocessor

Source:
https://raspberrytips.com/is-raspberry-pi-a-microcontroller/
https://libre.computer/products/aml-s905x-cc/
https://www.raspberrypi.com/products/raspberry-pi-4-model-b/
https://learn.sparkfun.com/tutorials/esp32-thing-plus-usb-c-hookup-guide/introduction

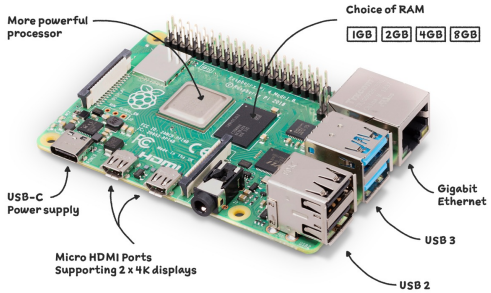
A microprocessor is the controlling unit of a micro-computer wrapped in a small chip, and it contains all the functions of a central processing unit of a computer.

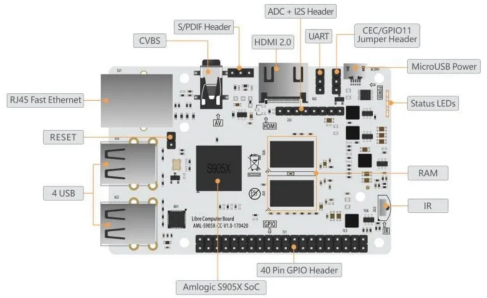
It performs Arithmetic Logical Unit (ALU) operations, and it communicates with other connected devices.

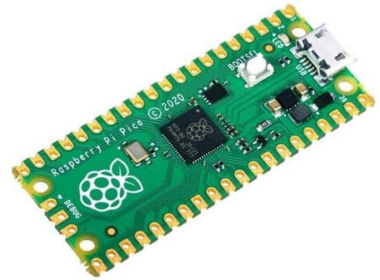
	Microprocessor	Microcontroller
Memory	Over 512 MB in general	Minimal, generally under 256 KB
Clock speed	Between 1 an 4 Ghz on average	Under 300 MHz in general
Power consumption	High	Low
Application	Complex tasks requiring calculations	Fixed and predefined task
Architecture	32 or 64 bits	8, 16 or 36 bits

A microcontroller is a chip that is optimized to control electronic devices. It is found in a single integrated circuit that is dedicated to performing a specific task.

It controls other portions of an electronic system, usually via a microprocessor unit.





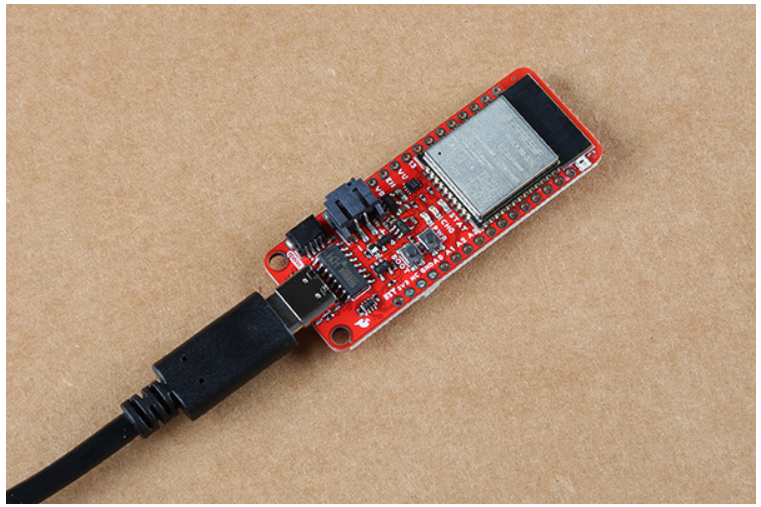


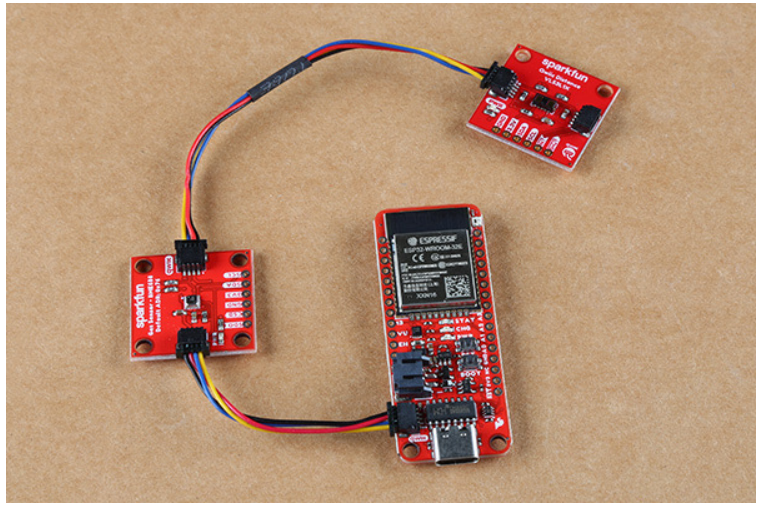
School of Engineering & Applied Science
THE GEORGE WASHINGTON UNIVERSITY

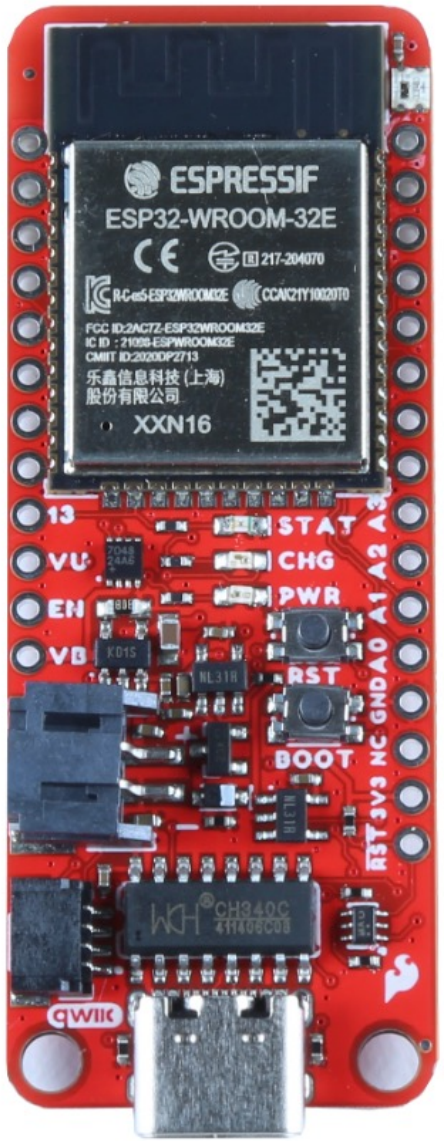
Setting up the ESP32 microcontroller with the Raspberry Pi 4B

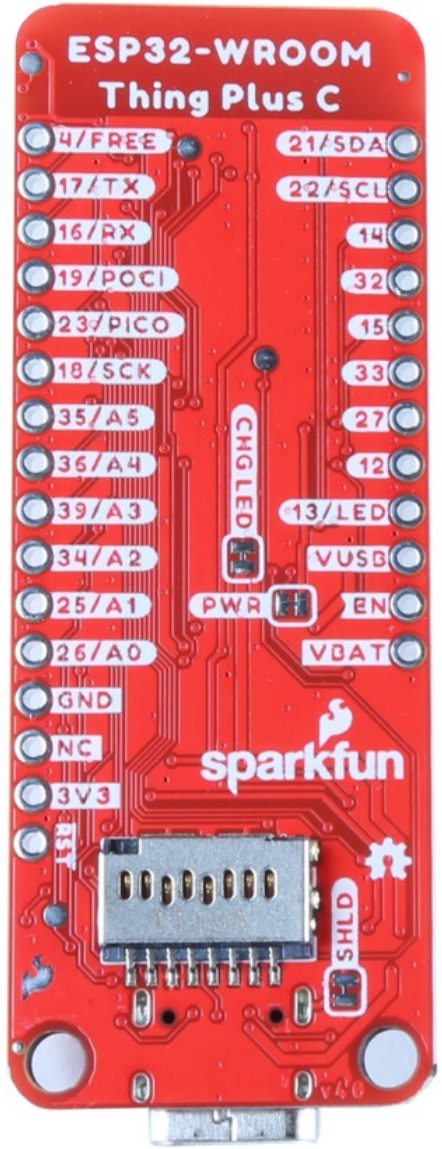
Source:
https://learn.sparkfun.com/tutorials/esp32-thing-plus-usb-c-hookup-guide/introduction

ESP32 Microcontroller – A first look





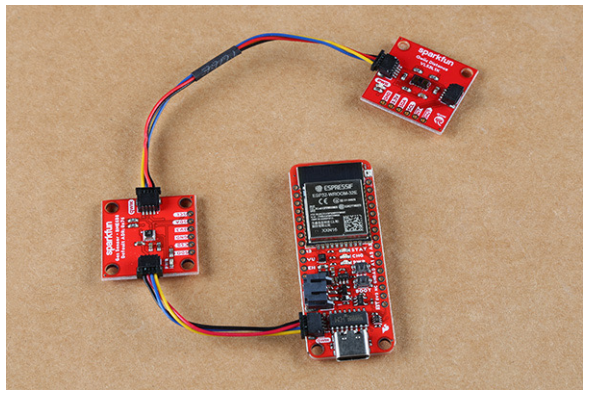


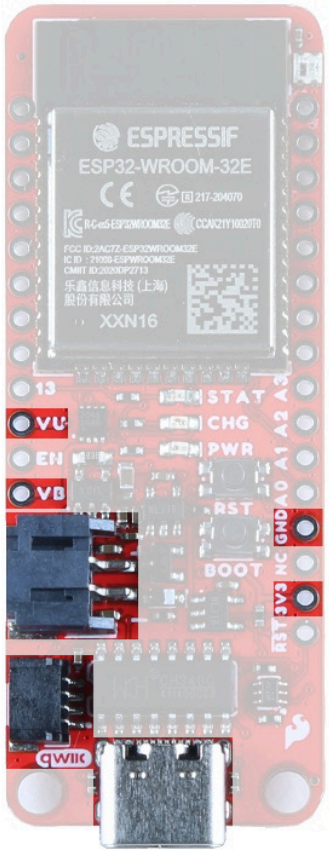


Source:
https://learn.sparkfun.com/tutorials/esp32-thing-plus-usb-c-hookup-guide/introduction

ESP32-WROOM only requires 3.3V to power the board.

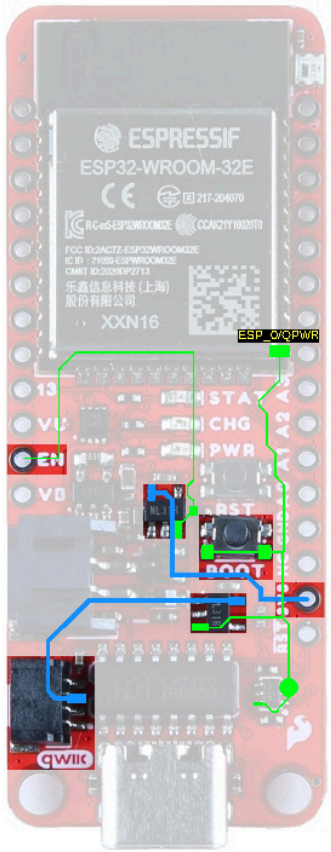
- The simplest method to power the board is through the USB-C connector.
- Alternatively, the 3V3, VBAT, and VUSB pins can also be used to supply power to the board.





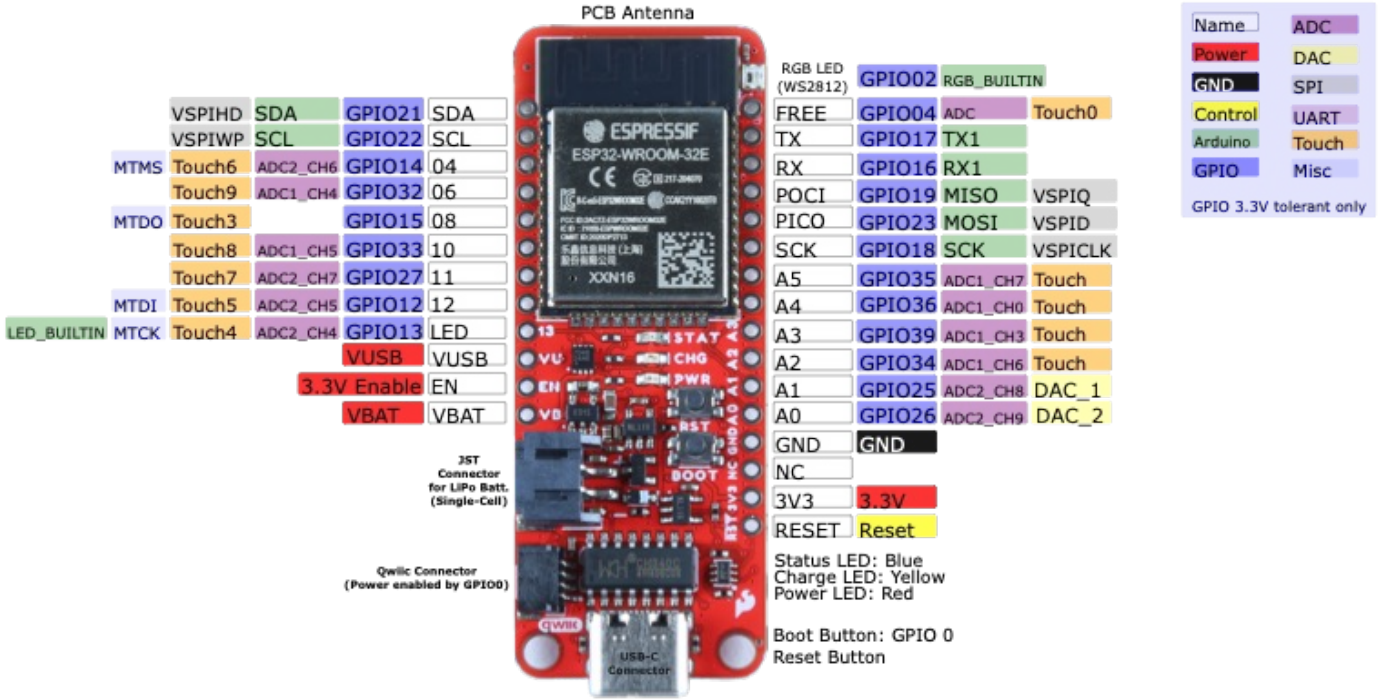
USB-C connector on the ESP32-WROOM

The BOOT button is also connected to GPIO 0



ESP32-WROOM is only compatible with **2.4GHz WiFi** networks; it will not work on the 5GHz bands.

SparkFun ESP32 Thing Plus (USB-C) (WRL-20168)



Power
ESP32 VCC Range: 3.0V-3.6V
VBAT: Direct to battery (and charger)
VUSB: Direct to USB (5V)
3V3: Direct to 3.3V regulator (700mA)
ESP32-S2 Current Consumption:
· WiFi: 380mA (peak)
· Light-Sleep: 800µA
· Deep-Sleep: 10-150µA
Power Control
EN pin - Power for board
GPIO0 - Power to Qwiic Connector

ESP32-WROOM
Dual-Core Xtensa 32-bit LX7
Up to 240MHz
520KB SRAM
16MB SPI Flash
Multiplexed I/Os allow up to
13x 12-bit ADC Channels
3x SPI Interfaces
3x UART Interfaces
2x I2C Interfaces
2x I2S Interface
16x 20-bit PWM Channels
2x 8-bit DACs
8x Capacitive Touch Inputs

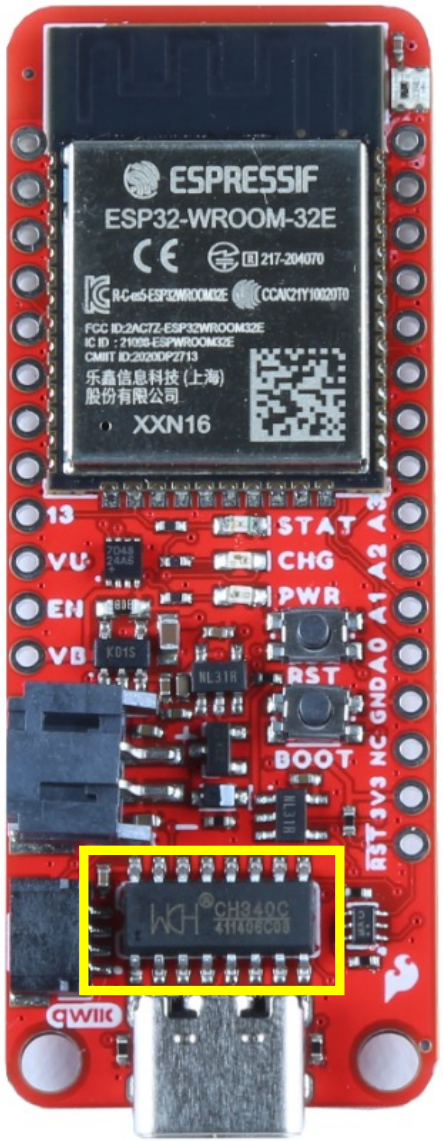
Wireless
WiFi: 802.11 b/g/n (up to 150Mbps)
WPA/WPA2/WPA2-Enterprise/SPS
Bluetooth:Bluetooth 4.2/BLE
Other*
Temp sensor (-40°C to 125°C)
Hall Sensor
SD/SDIO/MMC Host Controller
· µSD card slot CS: GPIO05 SS
CAN Bus
*On datasheet, but may not be supported yet

sparkfun ELECTRONICS

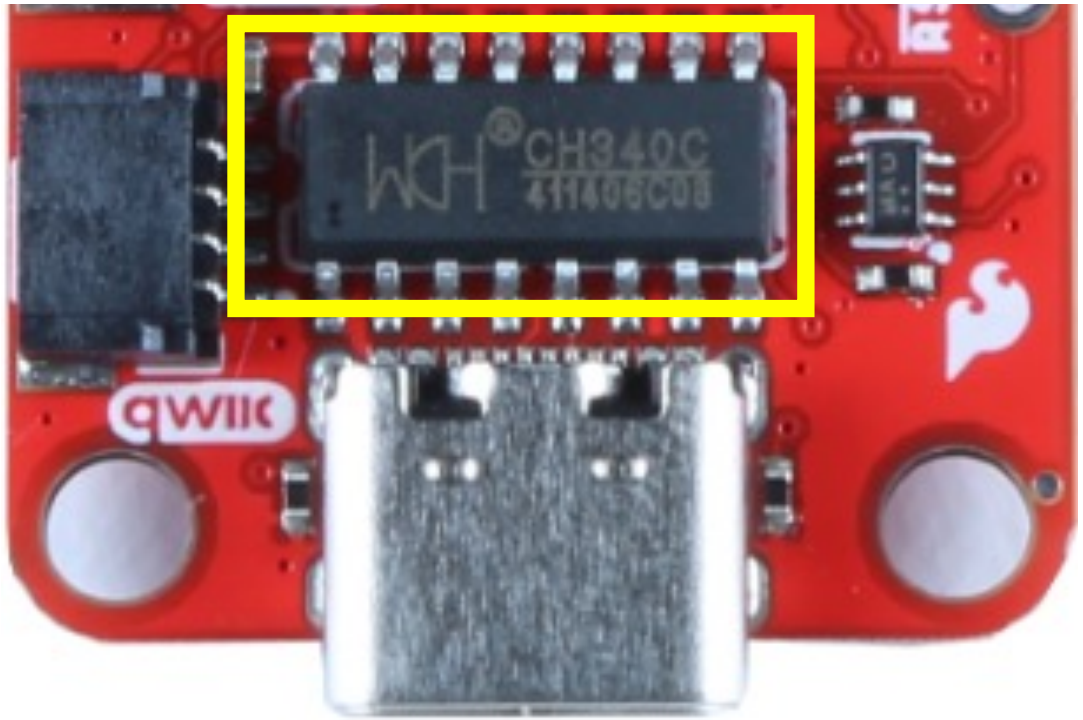
Following slides will demonstrate how to set-up the ESP32 with the Raspberry Pi 4B

Source:
https://learn.sparkfun.com/tutorials/esp32-thing-plus-usb-c-hookup-guide/introduction
https://cdn.sparkfun.com/assets/5/0/a/8/5/CH340DS1.PDF

Step 0: Install CH340 drivers for ESP32 WROOM



CH340 is a USB bus converter chip which converts USB to serial port or printer port.

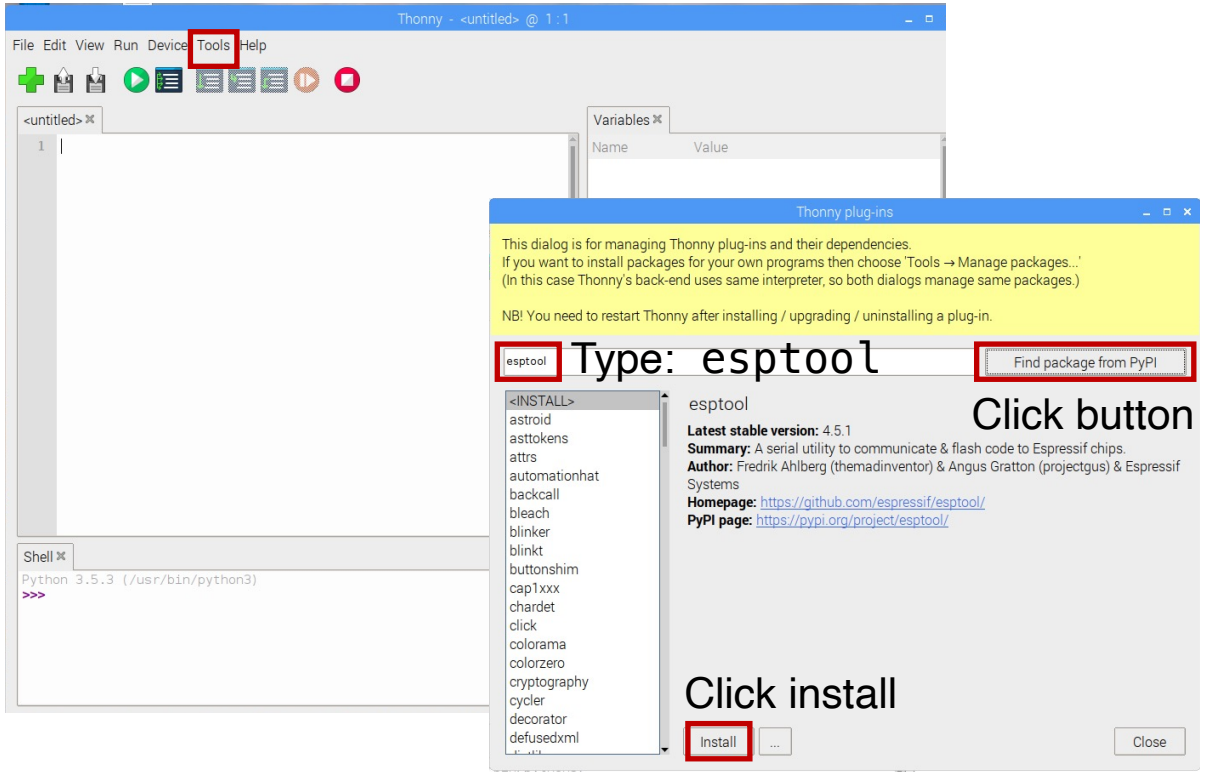



```
>>> sudo apt-get update
>>> sudo apt-get upgrade
```

Step 1: Check and Install **esptool.py** plug-in

In Thonny

Click: `Tools`



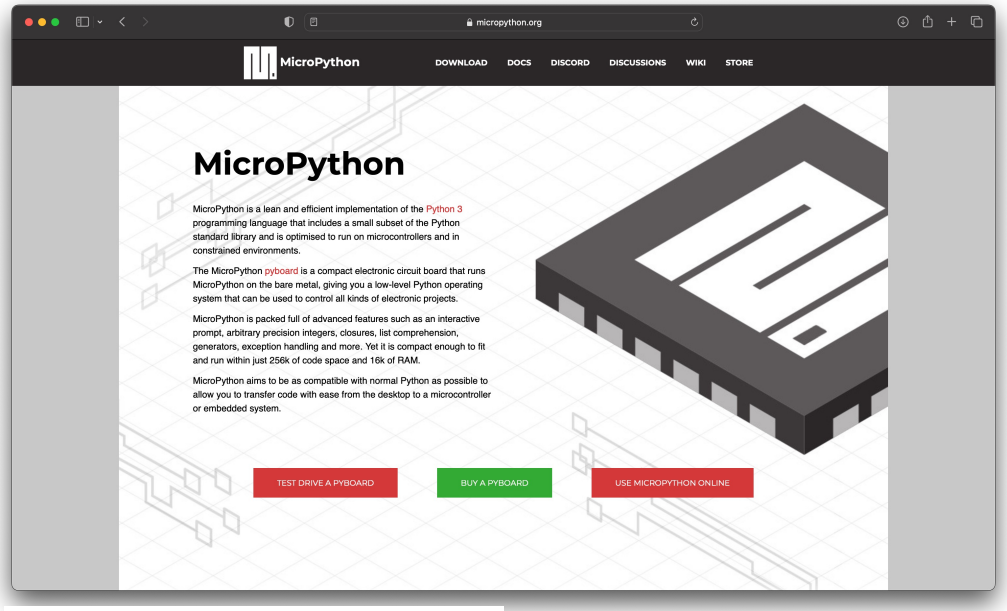
In Terminal

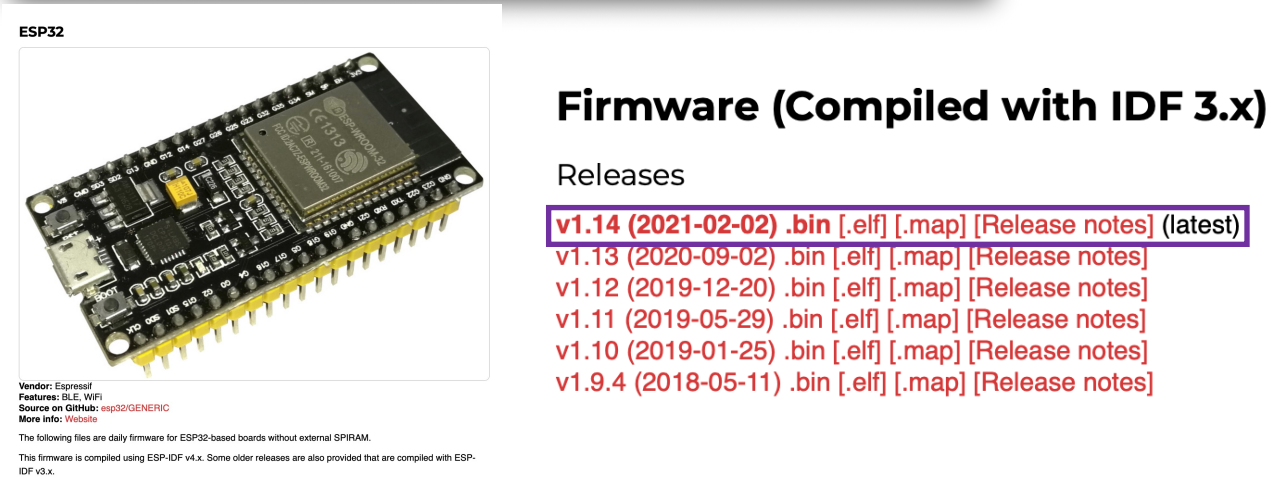
`sudo pip install esptool`

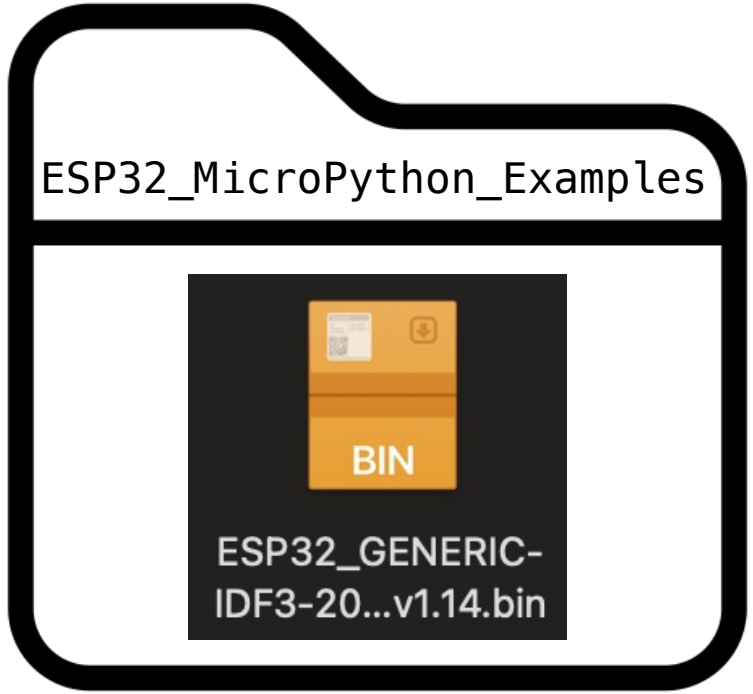
```
pi@raspberrypi:~/Downloads $ sudo pip3 install esptool
DEPRECATION: Python 3.5 reached the end of its life on September 13th, 2020. Please upgrade your Python as Python 3.5 i
s no longer maintained. pip 21.0 will drop support for Python 3.5 in January 2021. pip 21.0 will remove support for thi
s functionality.
Looking in indexes: https://pypi.org/simple, https://www.piwheels.org/simple
Collecting esptool
  Downloading https://www.piwheels.org/simple/esptool/esptool-3.3.3-py3-none-any.whl (355 kB)
     |████████████████████████████████| 355 kB 356 kB/s
Collecting reedsolo<=1.5.4,>=1.5.3
  Downloading https://www.piwheels.org/simple/reedsolo/reedsolo-1.5.4-cp35-cp35m-linux_armv7l.whl (674 kB)
     |████████████████████████████████| 674 kB 1.1 MB/s
Collecting ecdsa>=0.16.0
  Downloading https://www.piwheels.org/simple/ecdsa/ecdsa-0.18.0-py2.py3-none-any.whl (142 kB)
     |████████████████████████████████| 142 kB 1.9 MB/s
Requirement already satisfied: pyserial>=3.0 in /usr/lib/python3/dist-packages (from esptool) (3.2.1)
Collecting cryptography>=2.1.4
  Downloading https://www.piwheels.org/simple/cryptography/cryptography-3.2.1-cp35-cp35m-linux_armv7l.whl (723 kB)
     |████████████████████████████████| 723 kB 377 kB/s
Collecting bitstring<4,>=3.1.6
  Downloading https://www.piwheels.org/simple/bitstring/bitstring-3.1.9-py3-none-any.whl (39 kB)
Collecting cffi!=1.11.3,>=1.8
  Downloading https://www.piwheels.org/simple/cffi/cffi-1.15.1-cp35-cp35m-linux_armv7l.whl (318 kB)
     |████████████████████████████████| 318 kB 337 kB/s
Requirement already satisfied: six>=1.4.1 in /usr/lib/python3/dist-packages (from cryptography>=2.1.4->esptool) (1.12.0
)
Collecting pycparser
  Downloading https://www.piwheels.org/simple/pycparser/pycparser-2.21-py2.py3-none-any.whl (119 kB)
     |████████████████████████████████| 119 kB 836 kB/s
Installing collected packages: pycparser, cffi, reedsolo, ecdsa, cryptography, bitstring, esptool
  Attempting uninstall: cryptography
    Found existing installation: cryptography 1.7.1
    Uninstalling cryptography-1.7.1:
      Successfully uninstalled cryptography-1.7.1
Successfully installed bitstring-3.1.9 cffi-1.15.1 cryptography-3.2.1 ecdsa-0.18.0 esptool-3.3.3 pycparser-2.21 reedsol
o-1.5.4
pi@raspberrypi:~/Downloads $
```

Source:
Folder by Colourcreatype from https://thenounproject.com/browse/icons/term/folder/
Micropython https://micropython.org

Step 2: Download generic ESP32 firmware from `https://micropython.org/download/esp32/` [Provided to you with the github repo]



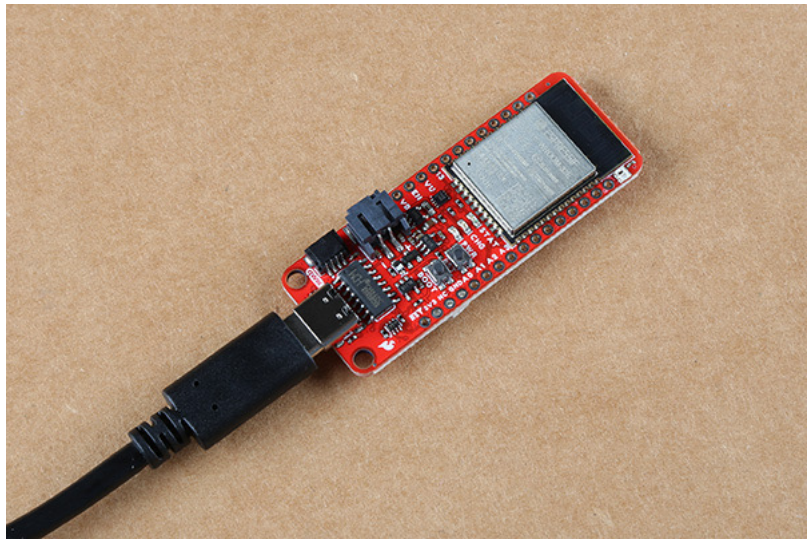




Source:
Folder by Colourcreatype from https://thenounproject.com/browse/icons/term/folder/
Micropython https://micropython.org
https://learn.sparkfun.com/tutorials/esp32-thing-plus-usb-c-hookup-guide/introduction

Flashing ESP32 using Thonny IDE

- Step 3: Connect the ESP32 microcontroller using the USB cable provided and "erase the flash"



`ls /dev/tty*`

```
pi@raspberrypi: ~
File  Edit  Tabs  Help
pi@raspberrypi:~ $ ls /dev/tty*
/dev/tty      /dev/tty19  /dev/tty3   /dev/tty40  /dev/tty51  /dev/tty62
/dev/tty0     /dev/tty2   /dev/tty30  /dev/tty41  /dev/tty52  /dev/tty63
/dev/tty1     /dev/tty20  /dev/tty31  /dev/tty42  /dev/tty53  /dev/tty7
/dev/tty10    /dev/tty21  /dev/tty32  /dev/tty43  /dev/tty54  /dev/tty8
/dev/tty11    /dev/tty22  /dev/tty33  /dev/tty44  /dev/tty55  /dev/tty9
/dev/tty12    /dev/tty23  /dev/tty34  /dev/tty45  /dev/tty56  /dev/ttyAMA0
/dev/tty13    /dev/tty24  /dev/tty35  /dev/tty46  /dev/tty57  /dev/ttyprintk
/dev/tty14    /dev/tty25  /dev/tty36  /dev/tty47  /dev/tty58  /dev/ttyS0
/dev/tty15    /dev/tty26  /dev/tty37  /dev/tty48  /dev/tty59  /dev/ttyUSB0
/dev/tty16    /dev/tty27  /dev/tty38  /dev/tty49  /dev/tty6
/dev/tty17    /dev/tty28  /dev/tty39  /dev/tty5   /dev/tty60
/dev/tty18    /dev/tty29  /dev/tty4   /dev/tty50  /dev/tty61
pi@raspberrypi:~ $
```

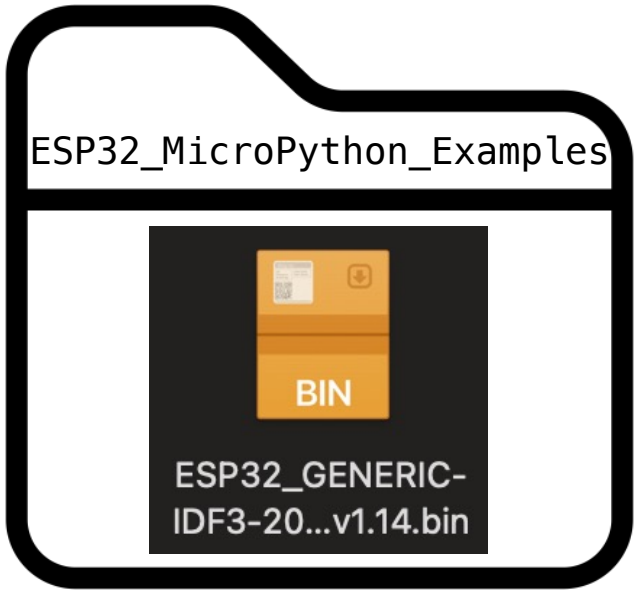
`sudo esptool.py –-port /dev/ttyUSB0 erase_flash`

```
pi@raspberrypi: ~
File  Edit  Tabs  Help
pi@raspberrypi:~ $ sudo esptool.py --port /dev/ttyUSB0 erase_flash
esptool.py v4.7.0
Serial port /dev/ttyUSB0
Connecting....
Detecting chip type... Unsupported detection protocol, switching and trying agai
n...
Connecting....
Detecting chip type... ESP32
Chip is ESP32-D0WD-V3 (revision v3.0)
Features: WiFi, BT, Dual Core, 240MHz, VRef calibration in efuse, Coding Scheme
None
Crystal is 40MHz
MAC: 94:e6:86:92:d2:7c
Uploading stub...
Running stub...
Stub running...
Erasing flash (this may take a while)...
Chip erase completed successfully in 55.3s
Hard resetting via RTS pin...
```

Source:
Folder by Colourcreatype from https://thenounproject.com/browse/icons/term/folder/
Micropython https://micropython.org

Step 4: Deploy the new firmware using the downloaded `binary file`

`sudo esptool.py --chip esp32 --port /dev/ttyUSB0 --baud 460800 write_flash -z 0x1000 ESP32_GENERIC-IDF3-20210202-v1.14.bin`



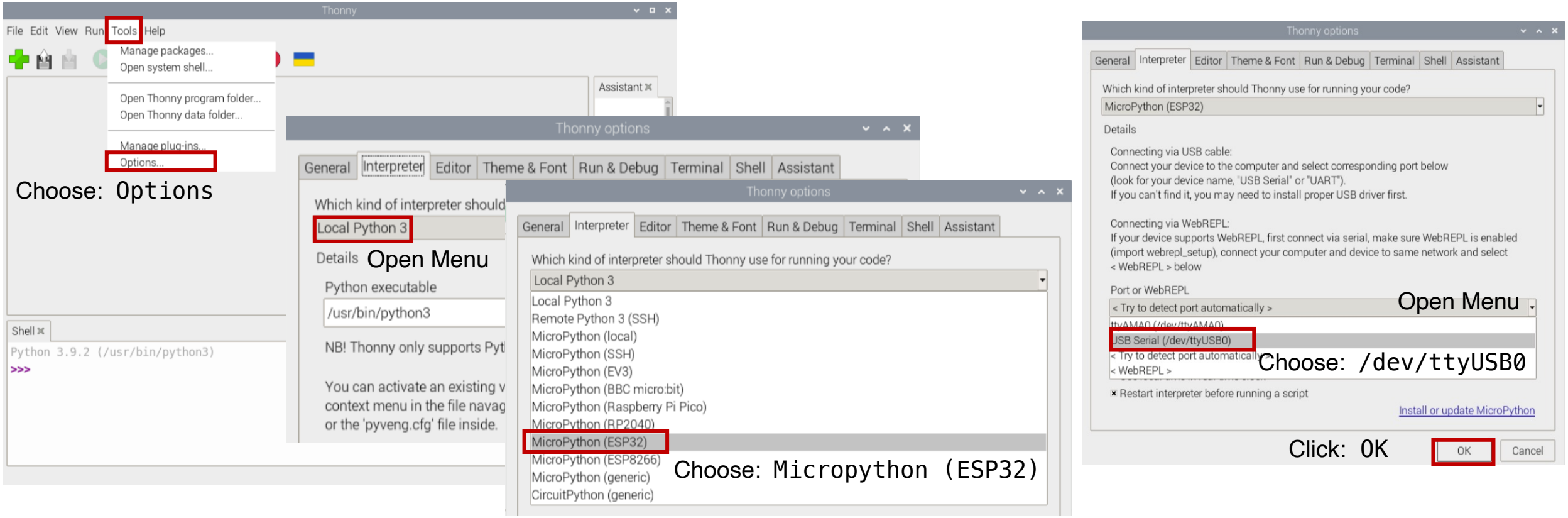

```
pi@raspberrypi:~/Desktop/Spring2024/MicropythonESP32 $ esptool.py --chip esp32 --port /dev/ttyUSB0 --baud 46080
0 write_flash -z 0x1000 ESP32_GENERIC-IDF3-20210202-v1.14.bin
esptool.py v4.6.2
Serial port /dev/ttyUSB0
Connecting....
Chip is ESP32-D0WD-V3 (revision v3.0)
Features: WiFi, BT, Dual Core, 240MHz, VRef calibration in efuse, Coding Scheme None
Crystal is 40MHz
MAC: 94:e6:86:92:d2:7c
Uploading stub...
Running stub...
Stub running...
Changing baud rate to 460800
Changed.
Configuring flash size...
Flash will be erased from 0x00001000 to 0x00161fff...
Compressed 1445632 bytes to 925476...
Wrote 1445632 bytes (925476 compressed) at 0x00001000 in 21.8 seconds (effective 531.0 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
pi@raspberrypi:~/Desktop/Spring2024/MicropythonESP32 $
```

Step 5: Start MicroPython interpreter on Thonny ID

Sources:
https://micropython.org
https://en.wikipedia.org/wiki/MicroPython
https://upload.wikimedia.org/wikipedia/commons/4/4e/Micropython-logo.svg

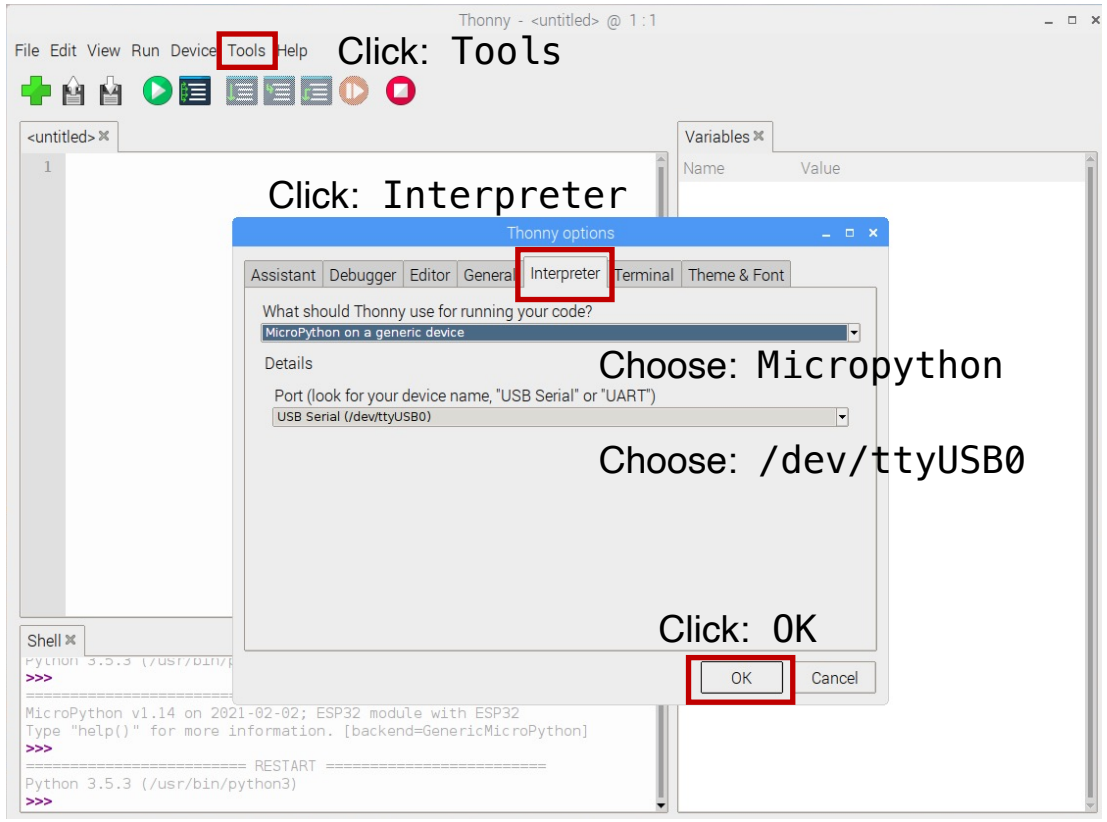
MicroPython is a software implementation of a programming language largely compatible with Python 3, written in C, that is optimized to run on a microcontroller.

Click: Tools

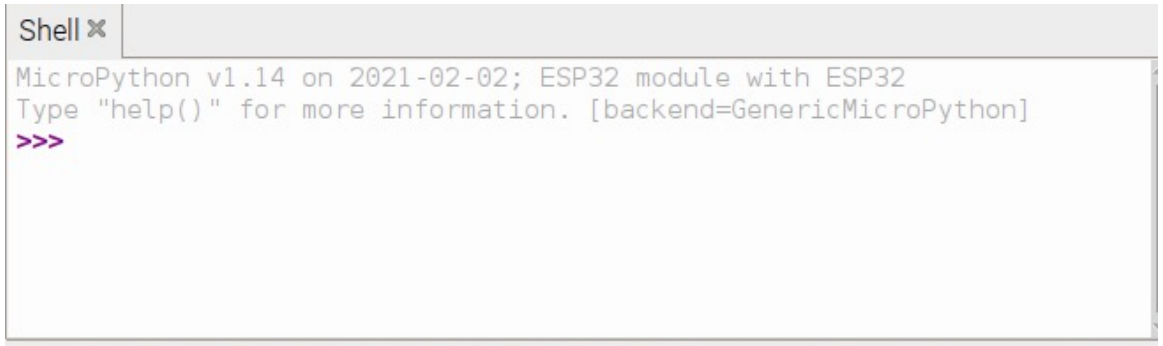


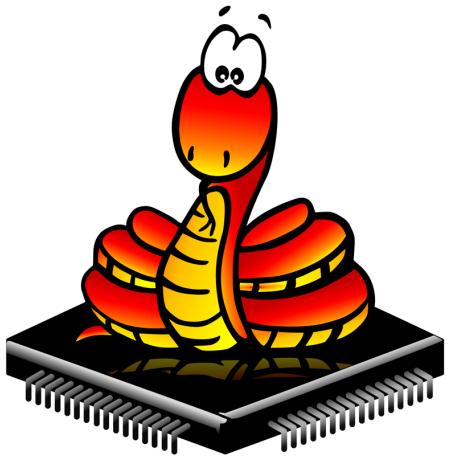
Step 6: Repeat step 5 to set up MicroPython interpreter on Thonny IDE

Sources:
https://micropython.org
https://en.wikipedia.org/wiki/MicroPython
https://upload.wikimedia.org/wikipedia/commons/4/4e/Micropython-logo.svg



Thonny is set to use the Micropython interpreter





School of Engineering & Applied Science
THE GEORGE WASHINGTON UNIVERSITY

GW

Setting up the ESP32

- You will need to execute Python codes using the Micropython interpreter on Thonny
- Download the codes provided to you on the shared google drive
- You will need two codes that should be flashed to the ESP32 from the Raspberry Pi 4B
  - boot.py
  - main.py
- You can work in groups if you like to complete the graded in-class exercise [10 points]