

# Microcontroller

Introduction HackLab

<https://github.com/KABK-HackLab/microcontroller-intro>



## What is a MicroController

### MCU (MicroController Unit)

A small self-contained computer on a single chip. The microcontroller is the core of an embedded system.

Embedded systems are specialized, computer-based systems combining hardware and software with the purpose of interacting with the physical world using IO Ports

The Microcontroller contains:

- Processor (that executes our code)
- Flash Memory (for program storage)
- Ram memory (for processing, variables)
- IO (Input / Output Ports) via 'pins'

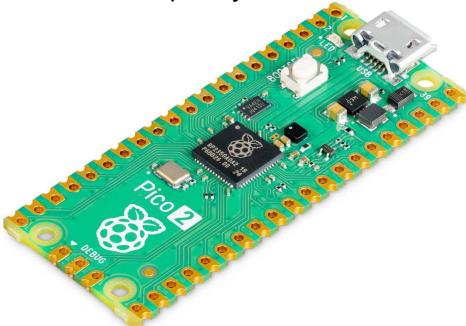


**Arduino** Well known, large community, open source software library. Various boards ranging in price and performance. Arduino is both a microcontroller board and a software development environment (IDE)

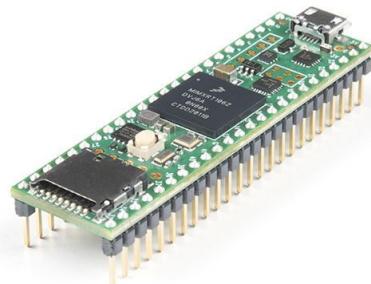


**ESP32** Includes build in Wifi and Bluetooth. Compatible with arduino IDE.

**Raspberry Pico** Cheap, Not the fastest board, but has interesting set advanced features (e.g. dual core). Not to be confused with the Raspberry Pi.



**Teensy** Powerful platform that extends on the arduino platform with advanced features, performance and software libraries. Expensive



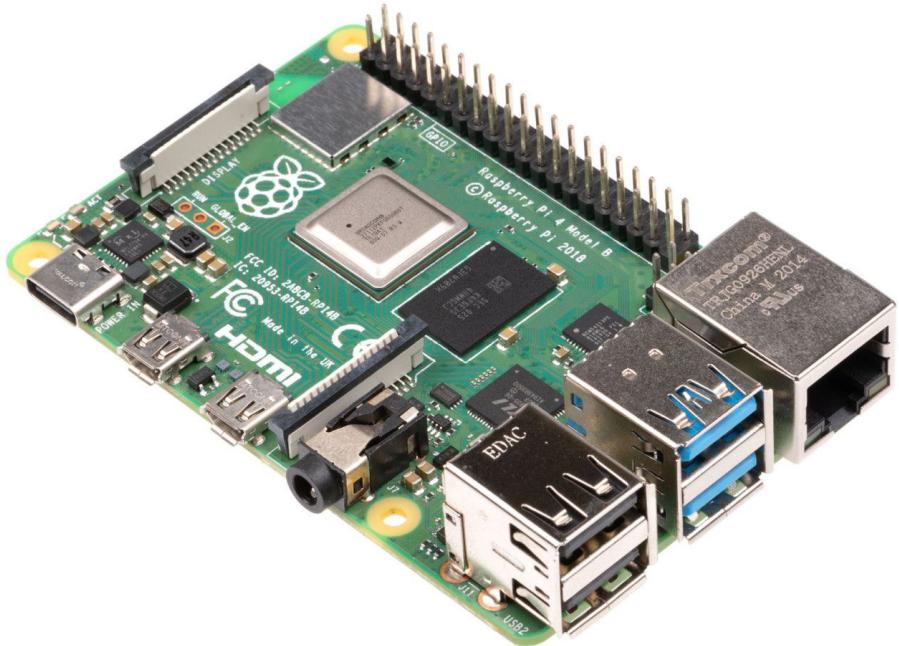
There are various brands and development environments.

While a MCU is only a single chip, most of them come on a microcontroller-board that makes it easier to work with. For example it features a usb connector to upload code.

The microcontroller-board is often just called microcontroller.

What microcontroller is best depends on your project and personal preference.

Each one has their own advantages and disadvantages.



# Raspberry Pi

The Raspberry Pi is not a microcontroller!

It is called a single-board computer and is actually a complete mini computer. In contrast to a microcontroller this can run a complete linux desktop system with graphics, audio, network, storage.

Similar to a microcontroller it features an IO port that can be used to communicate with the other hardware.

A microcontroller does not have an OS.

It only runs the code we put upload to the chip ourselves.

DEMO TIME

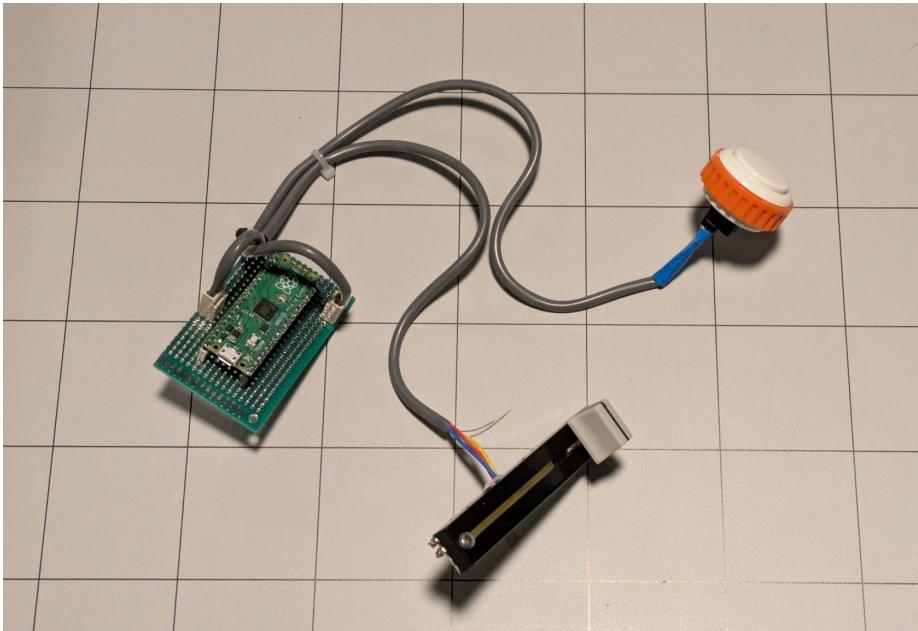
## Create a basic music instrument

Microcontroller functions as the interface between

**software:** audio synthesis software, running laptop

and

**hardware:** buttons / faders



```
sketch_may18a | Arduino 1.8.8
File Edit Sketch Tools Help
sketch_may18a
1 void setup() {
2 // put your setup code here, to run once:
3
4 }
5
6 void loop() {
7 // put your main code here, to run repeatedly:
8
9 }
```

Arduino/Genuino Uno on COM8

## Arduino IDE

For use with arduino and compatible boards. Used the the c/c++ programming language with specialised software libraries for writing code for microcontrollers.

The arduino ecosystem features an extensive range of libraries to connect a wide range of hardware. Sensors, displays, shields (extension boards), etc

There are various programming development environments.

Similar to the microcontroller boards, each one has their own advantages and disadvantages.

```
Thonny - Fblink.py @ 3:1
File Edit View Run Tools Help
blinky.py x
3 import time
4 from machine import Pin
5
6 led = Pin(25, Pin.OUT)
7
8 while True:
9     led.value(0) # yellow LED on
10    time.sleep_ms(100)
11    led.value(1) # yellow LED off
12    time.sleep_ms(600)
13
Shell x
>>> %Run -c $EDITOR_CONTENT
MicroPython (Pololu 3pi+ 2040 Robot) • COM4
```

## Thonny

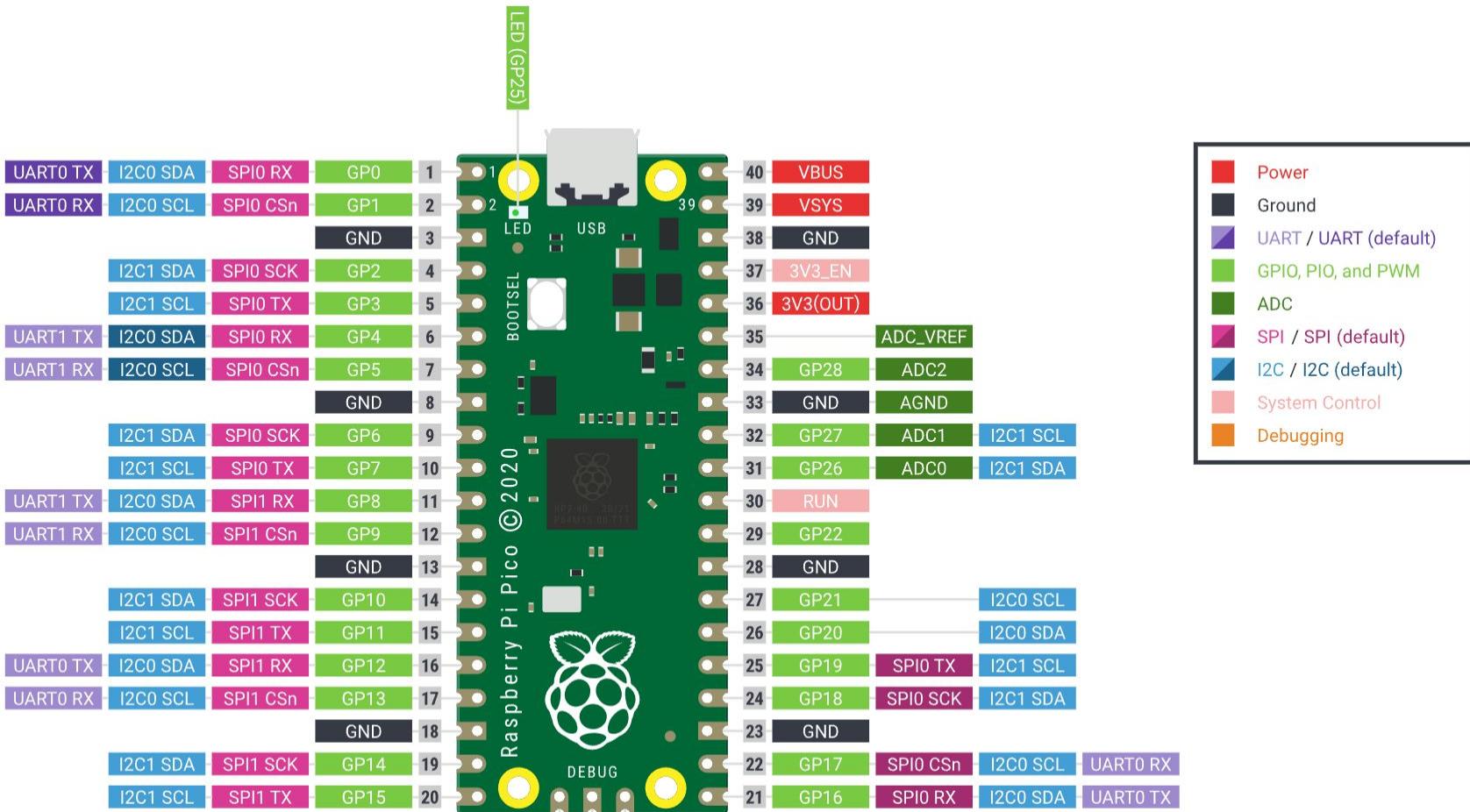
Focussed on the python programming language.

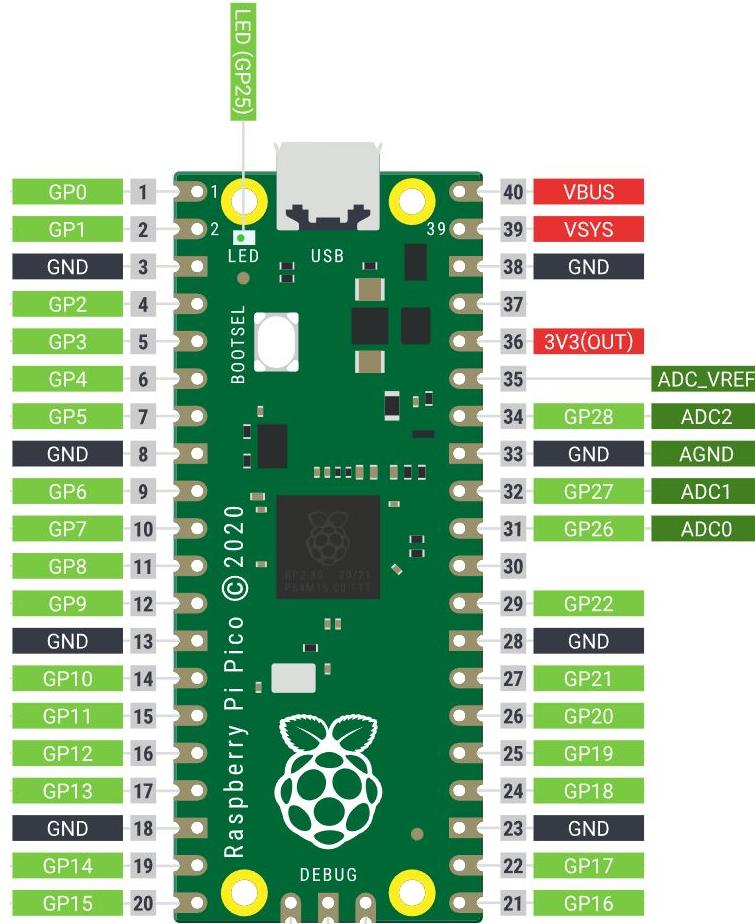
Can be used to write python code for PC's as well as microcontrollers.

Python is arguably easier for beginners, but less performant than the C/C++ language

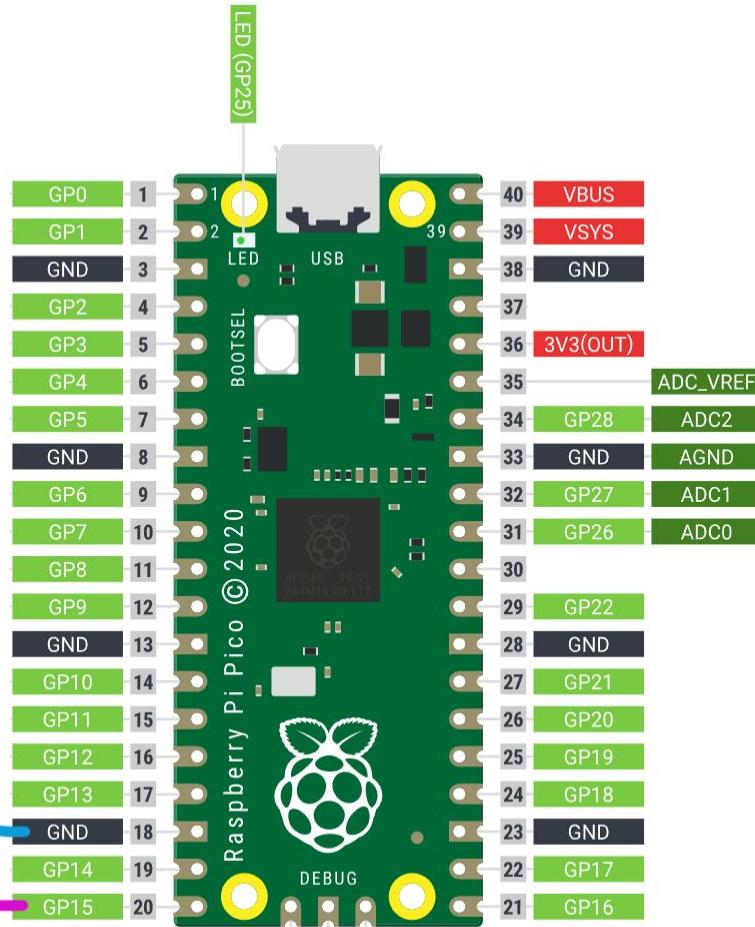
```
File Edit Selection View Go Run Terminal ...
main.cpp
PROJECT TASKS
  General
    o Build
      o Upload
      o Monitor
      o Upload and Monitor
      o Devices
      o Clean
    > Advanced
    > Remote Development
    > Miscellaneous
    > envesp32dot-devkit-v1
QUICK ACCESS
Ln 19 Col 2 Spaces:2 UFT-8 CRLF C++ Go Live Win32
```

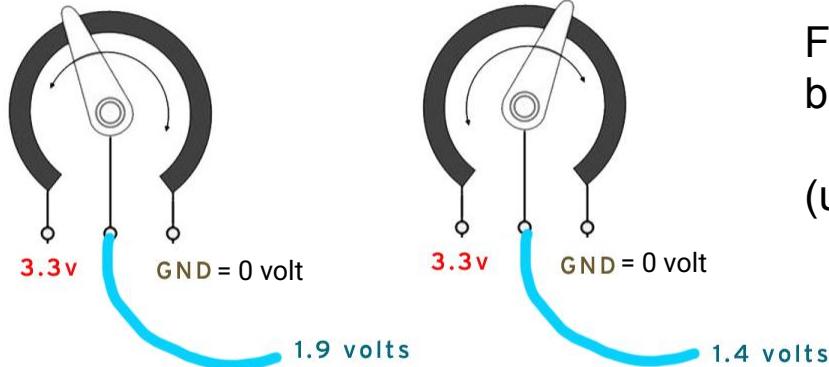
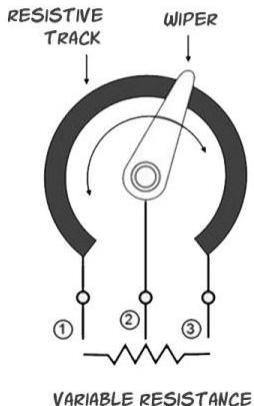
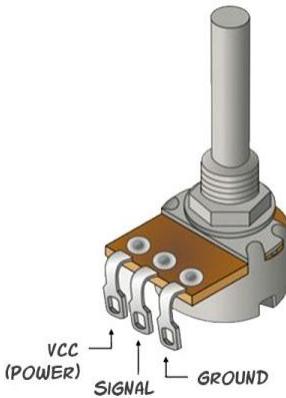
A red arrow points to the 'Upload' option in the sidebar under the 'PROJECT TASKS' section.





digital input





## Potmeter overview

The Fader works exactly the same, different style / housing.

Potmeter: output is the middle pin (pin 2)

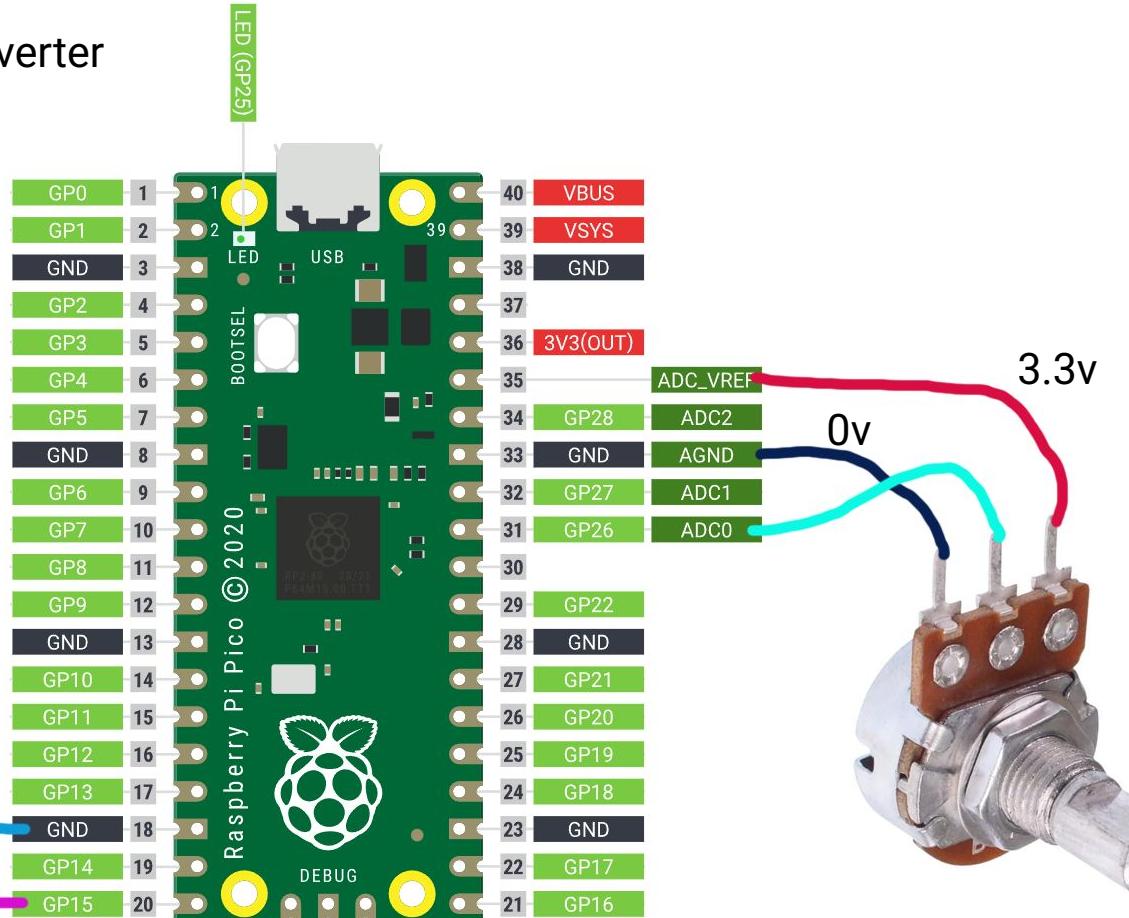
Fader: output position can vary between designs.

(usually labeled: pin 2)

ADC = Analog to Digital Converter

ADC can read a voltage  
between 0 and 3.3volt

digital input



analog input

## [ Coding Time ]

Download circuitpython for the raspberry pico:

<https://circuitpython.org/downloads>

Install/upload circuitpython to the board:

- 1) Hold the bootsel button and connect the usb cable, the pico should show up as a drive.
- 2) Copy the utf file to the pico, The pico will restart

<https://learn.adafruit.com/getting-started-with-raspberry-pi-pico-circuitpython/circuitpython>

Install Thonny:

<https://thonny.org/>

In Thonny go to Tools - Options - Interpreter and select circuitpython.