Raspberry Pi and GPIO

Giuseppe Scanniello

Simone Romano

Michelangelo Esposito

General information

A **general purpose** development board, much like Arduino. Available in different models, with increase functionalities.

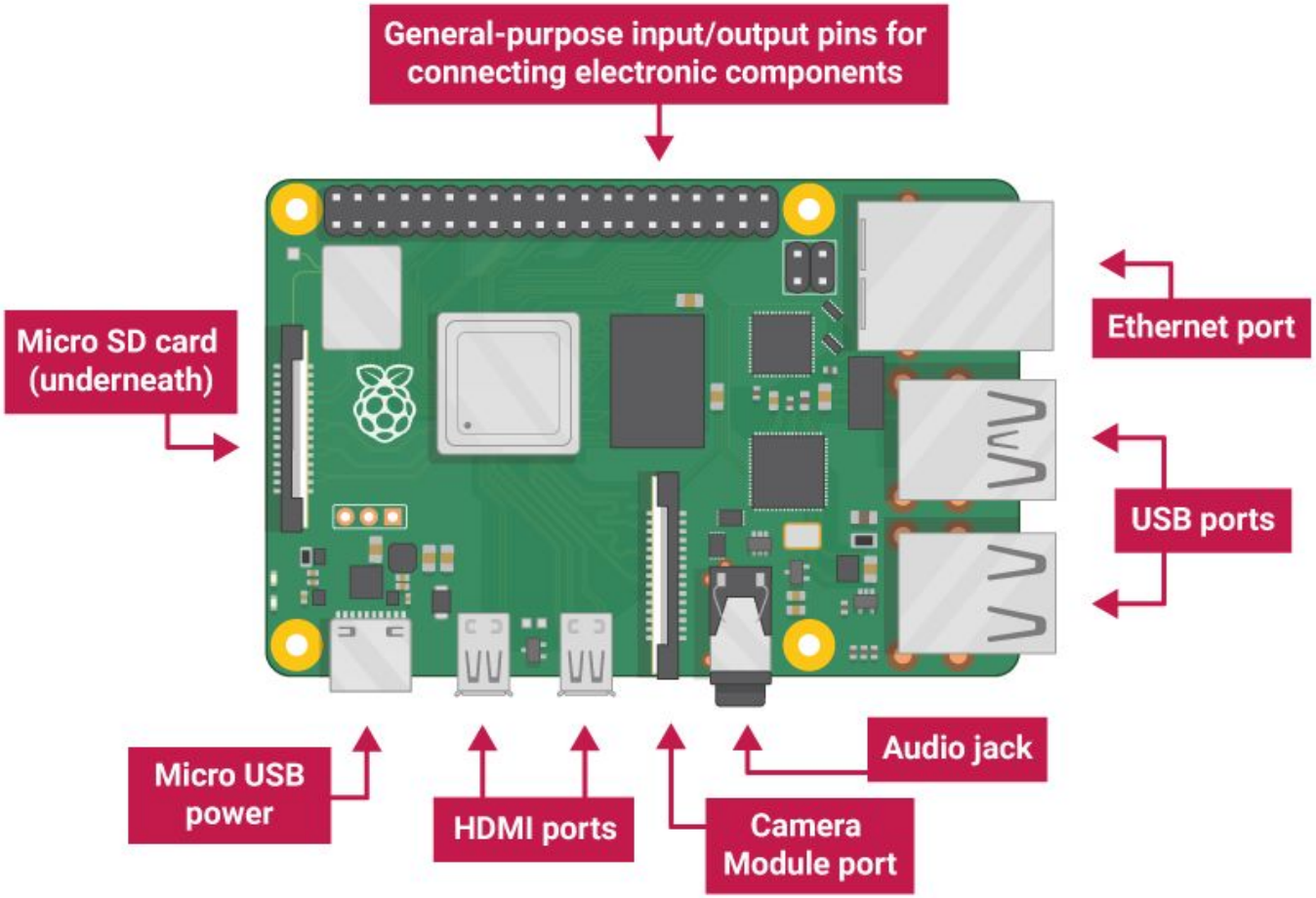
Raspbian OS based on Linux, stored on a micro SD card.

Programmable in Python, MicroPython or C++.

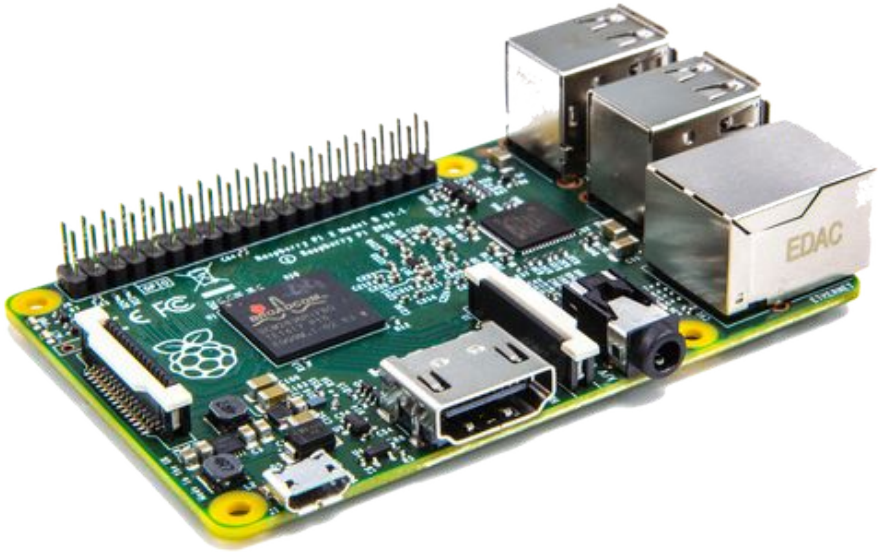
Top models (i.e. Raspberry Pi 4B) are effectively mini computers.

- **Mouse/keyboard support**
- **HMDI Display output**
- **Up to 8GB RAM.**

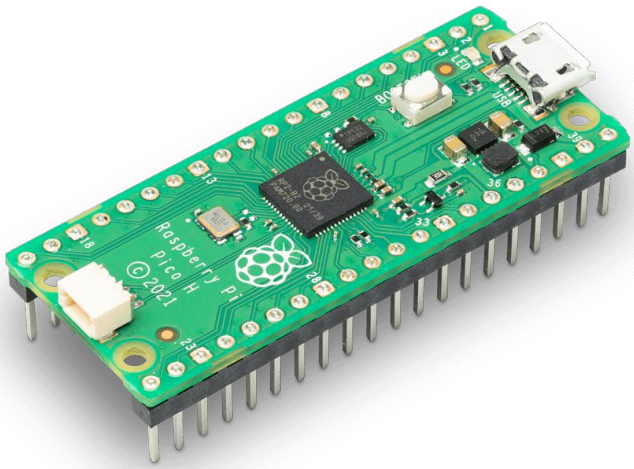
General information



Two of the models



Raspberry Pi 4

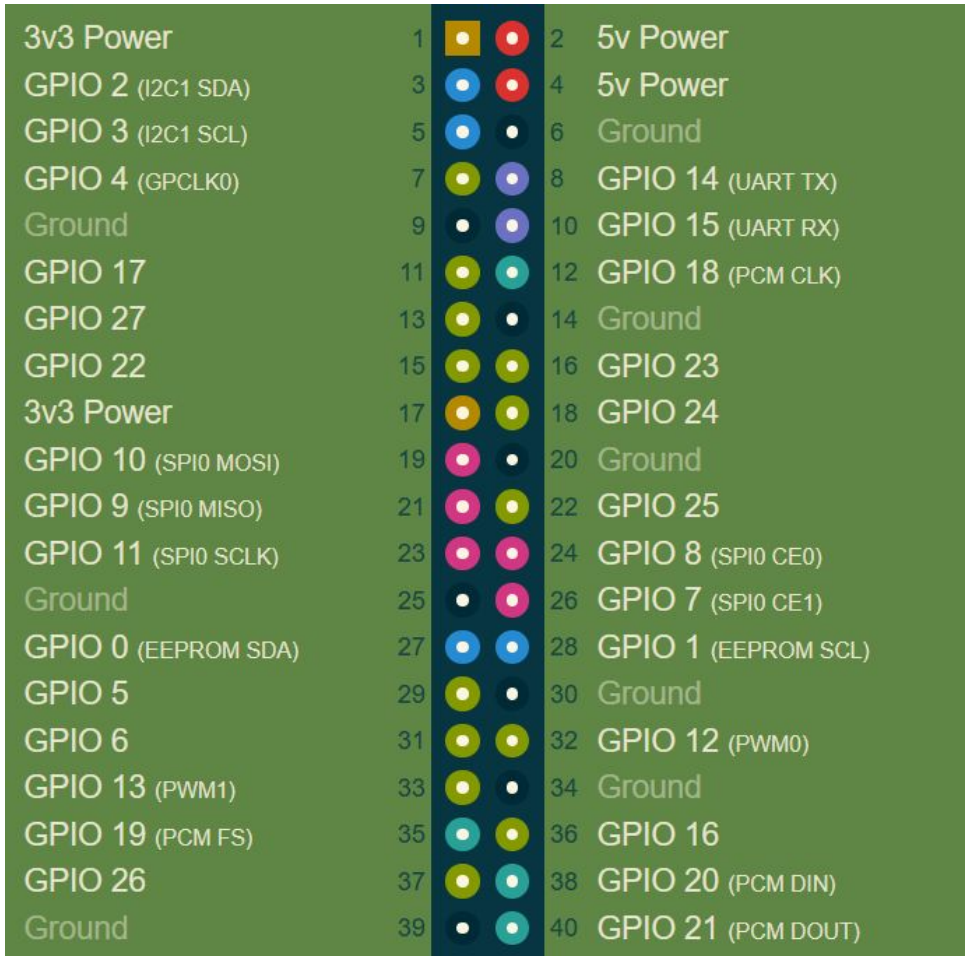


Raspberry Pi Pico

Raspberry Pi pinout

Model-independent

- **Power pins** (2, 4, 17)
- **Ground pins** (6, 9, 25, …)
- **GPIO pins** (3, 5, 7, …)



General Purpose Input-Output

Uncommitted digital signal pins which may be used as inputs or outputs and are controllable via software.

- Most commonly used as a mean of communication with other hardware (i.e. sensors and actuators)

General Purpose Input-Output

```
def blink():
    # Refer to the pins by the pin number
    GPIO.setmode(GPIO.BOARD)
    # Refer to the pin by their GPIO number
    #GPIO.setmode(GPIO.BCM)

    # Pin 18 (GPIO24) if we use BOARD
    # GPIO 18 if we use BCM
    LED_PIN = 18

    GPIO.setup(LED_PIN, GPIO.OUT)

    try:
        while True:
            GPIO.output(LED_PIN, GPIO.HIGH)
            print('LED ON')
            time.sleep(1)

            GPIO.output(LED_PIN, GPIO.LOW)
            print('LED OFF')
            time.sleep(1)
    except KeyboardInterrupt:
        GPIO.cleanup()
```

Setting the pin mode

Setting LED_PIN as an output pin

Turn on the LED

Turn off the LED

A (mocked) GPIO library

How can we use the GPIO functionalities **without** the actual board?

Create a mocked implementation. Then, based on hardware availability either import the mock or the real one.

```
try:
    import RPi.GPIO as GPIO
except:
    import mock.GPIO as GPIO
```

A (mocked) GPIO library

```
def output(channel, value):
    """
    Output to a GPIO channel or list of channels
    channel - either board pin number or BCM number depending on which mode is set.
    value   - 0/1 or False/True or LOW/HIGH

    """
    logger.info("Output channel : {} with value : {}".format(channel, value))

def input(channel):
    """
    Input from a GPIO channel.  Returns HIGH=1=True or LOW=0=False
    channel - either board pin number or BCM number depending on which mode is set.
    """
    logger.info("Reading from channel {}".format(channel))
```

Library credits to https://github.com/codenio/Mock.GPIO

A (mocked) GPIO library

```
# We mock the GPIO input function
@patch('mock.GPIO.input')
def test_low_temperature_activation(self, mock_input):
    mock_input.return_value = 15
    temperature = TemperatureSensor.read_temperature()

    # Thermostat requires a certain temperature range to activate
    thermostat = Thermostat()
    result = thermostat.activate(temperature)
    self.assertTrue(result)
```