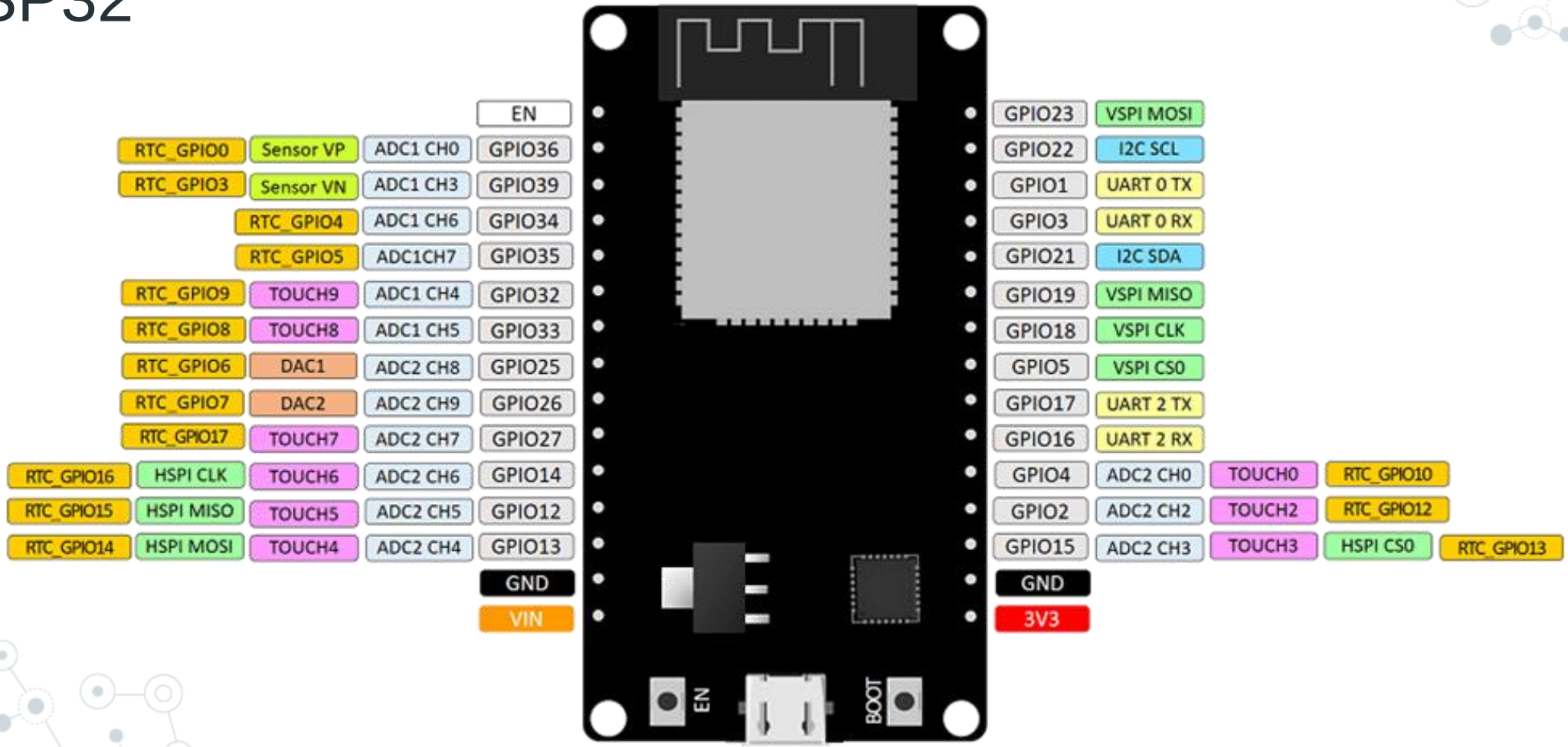
centro universitário FEI

Sistemas Embarcados

Prof. Dr. André Luiz Perin
Prof. Dr. Marco Antonio A. Melo
Prof. Dr. Rudolf T. Bühler

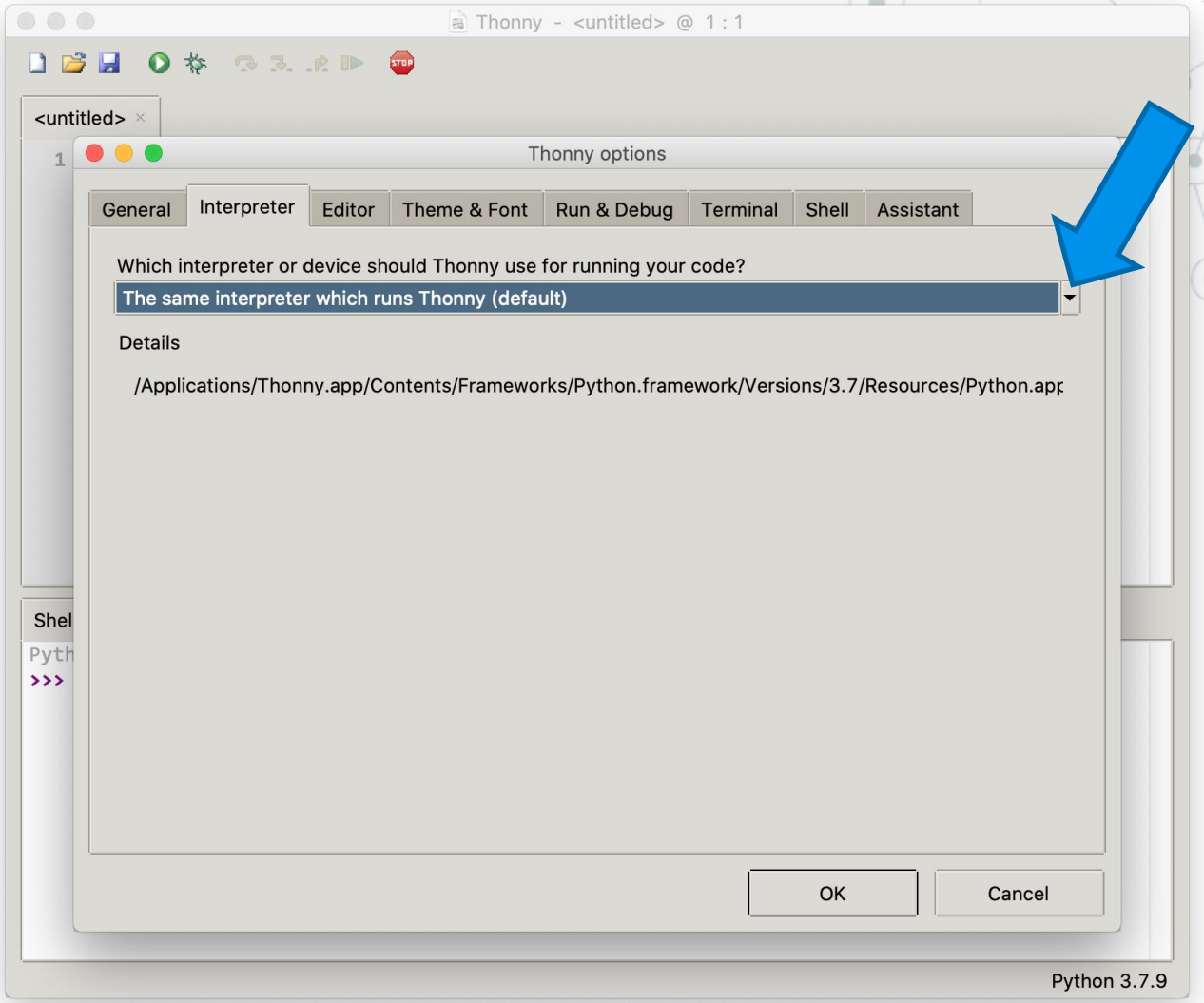
Sistemas Embarcados

- ESP32



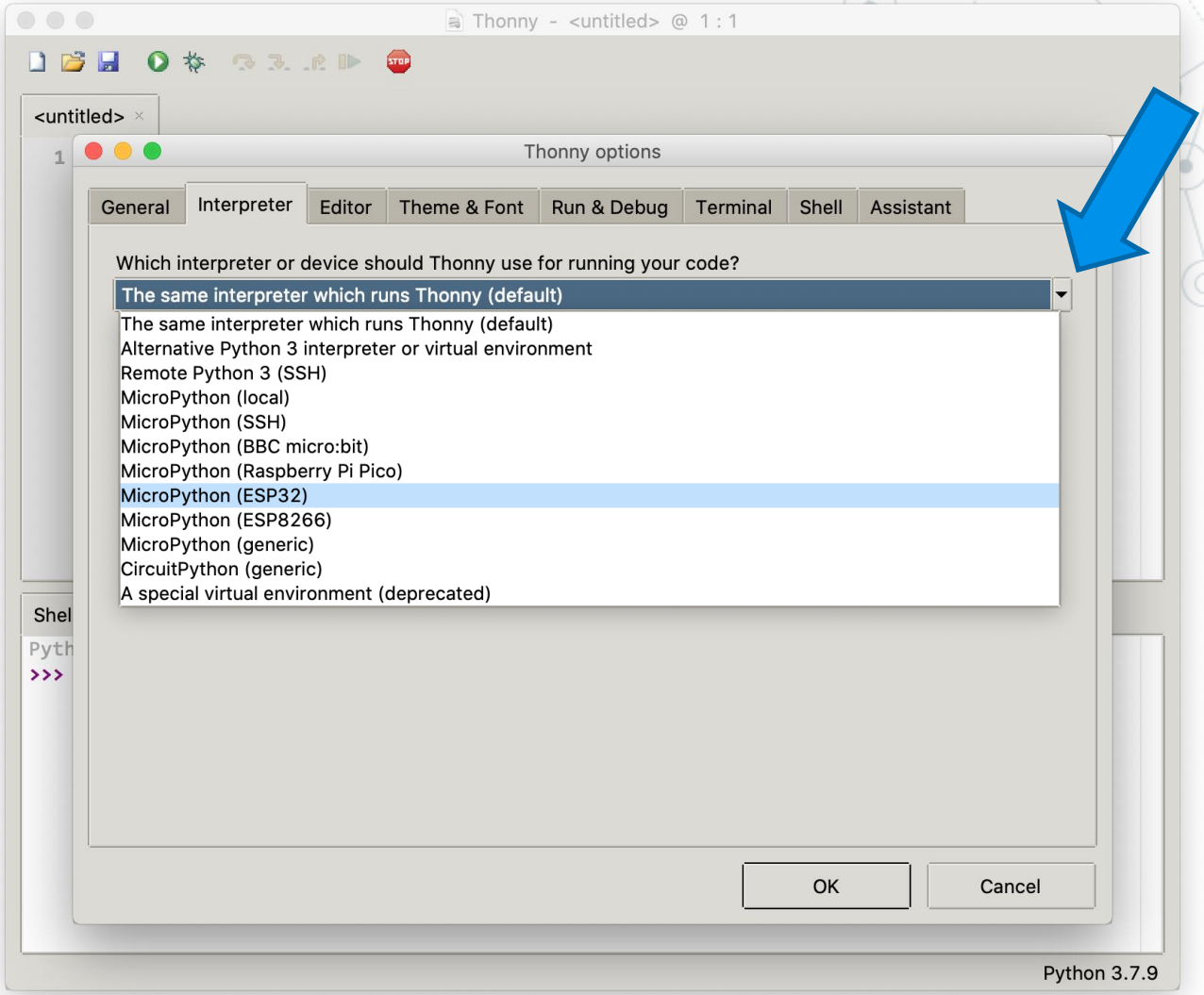
Sistemas Embarcados

- Thonny
  - Selecionar o interpretador:
    - Menu: Run/Select Interpreter



Sistemas Embarcados

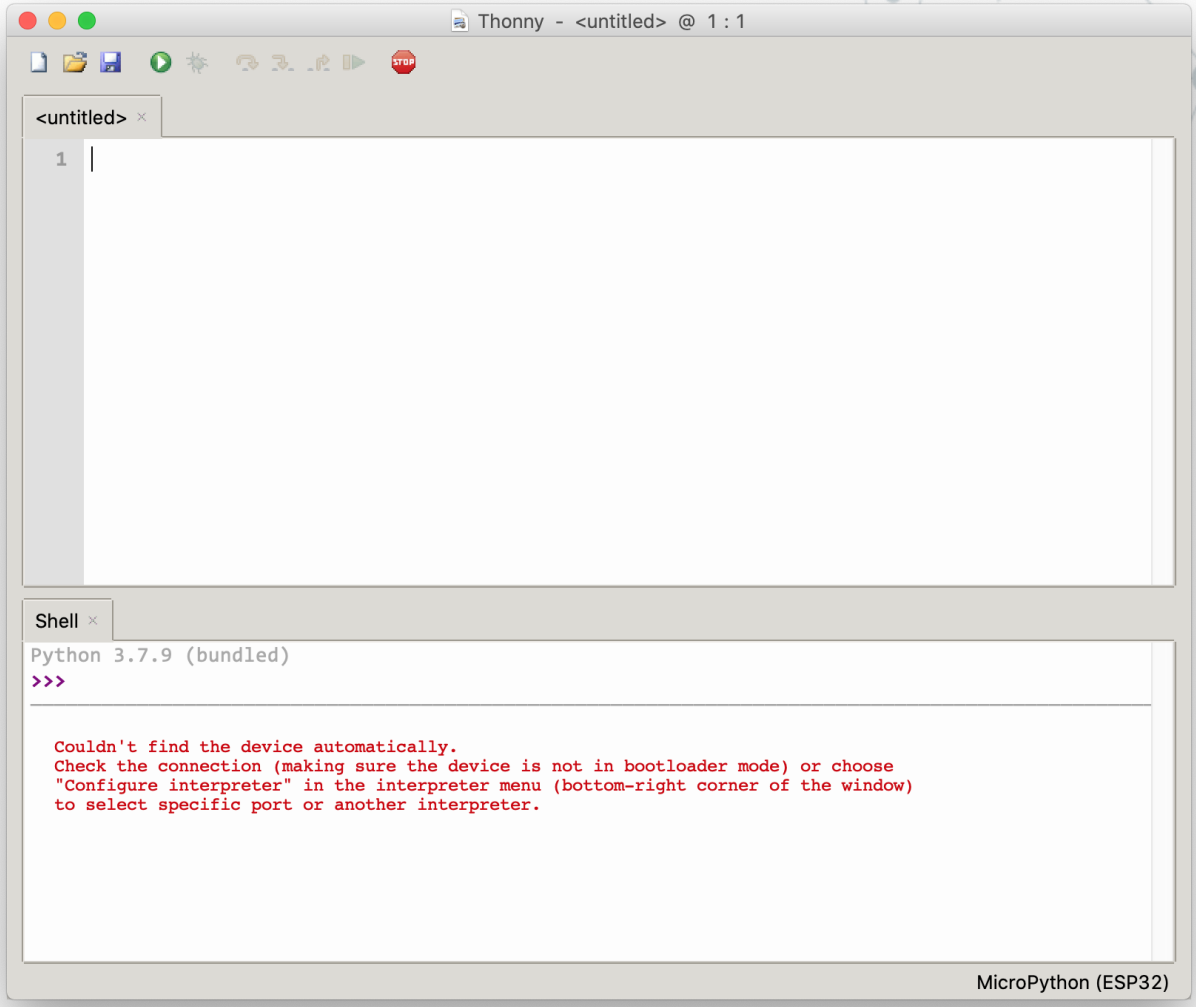
- Thonny
  - Selecionar o interpretador:
    - Menu: Run/Select Interpreter
      - Escolher MicroPython
        - ESP32 ou ESP8266



https://thonny.org

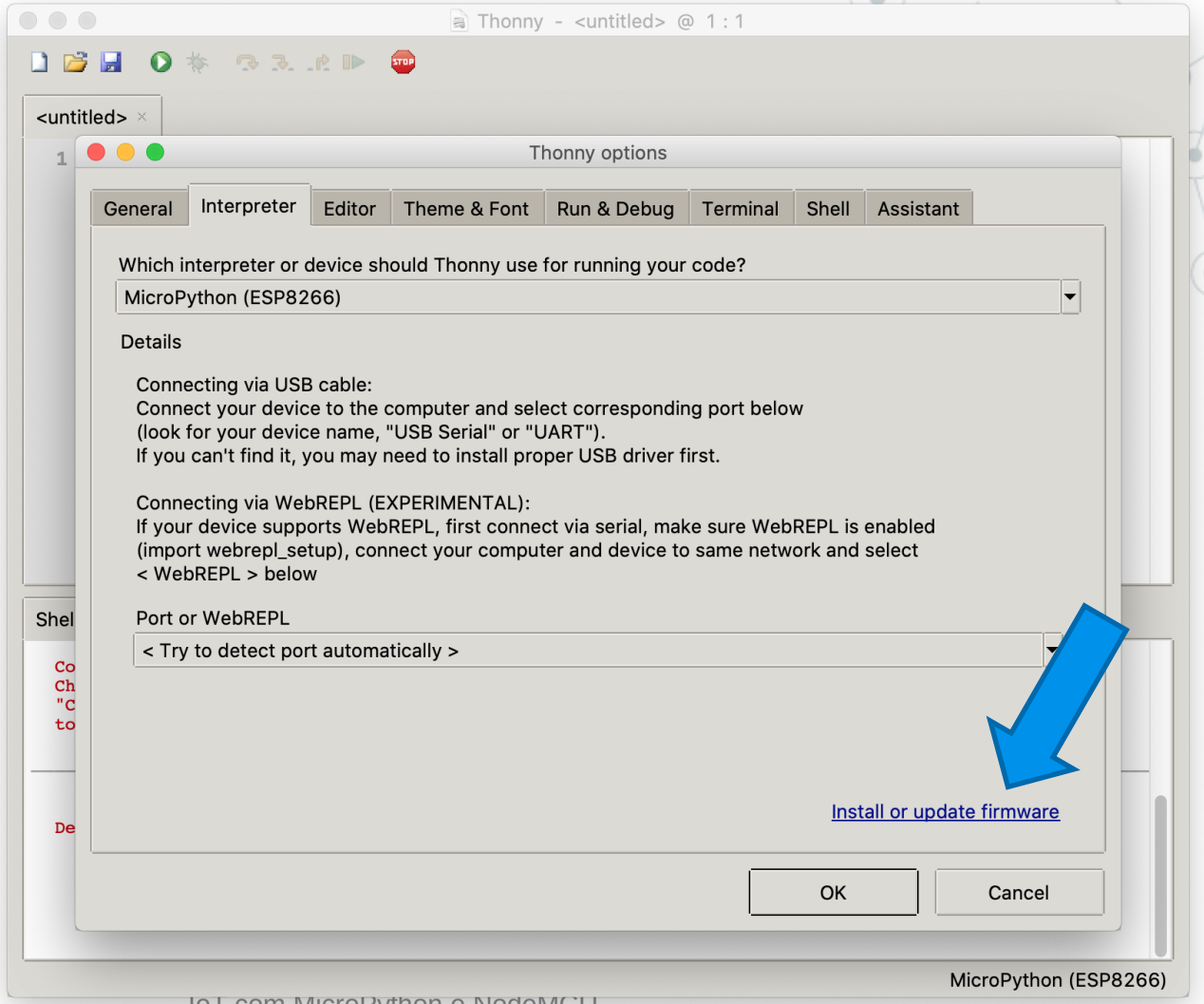
Sistemas Embarcados

- Thonny
  - Se o dispositivo não estiver conectado ou não tiver o firmware de micropython instalado aparecerá uma mensagem



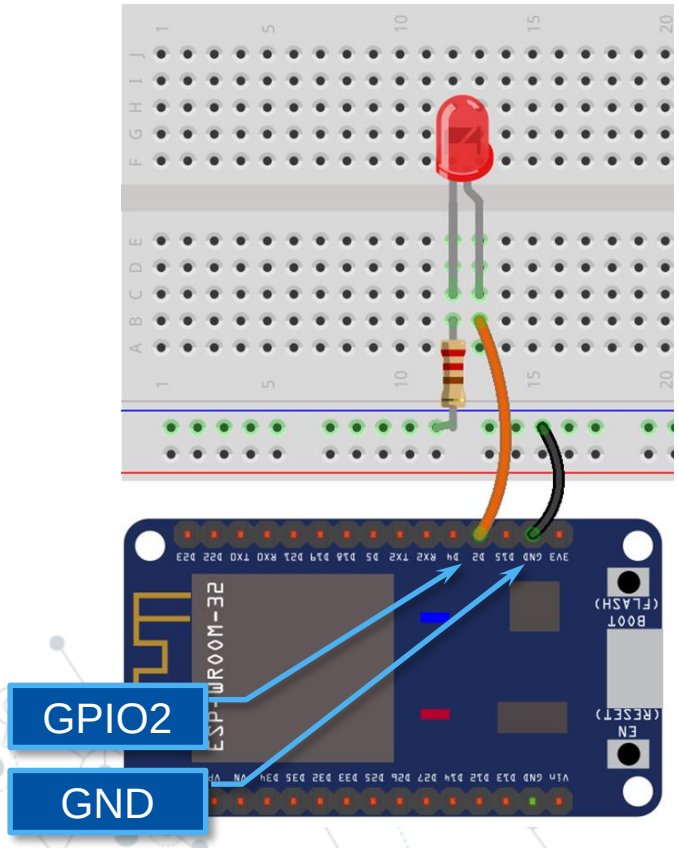
Sistemas Embarcados

- Thonny
  - Instalação do firmware do micropython
    - Clicar em “Install or update firmware”



Sistemas Embarcados

- Node MCU
  - Pisca LED



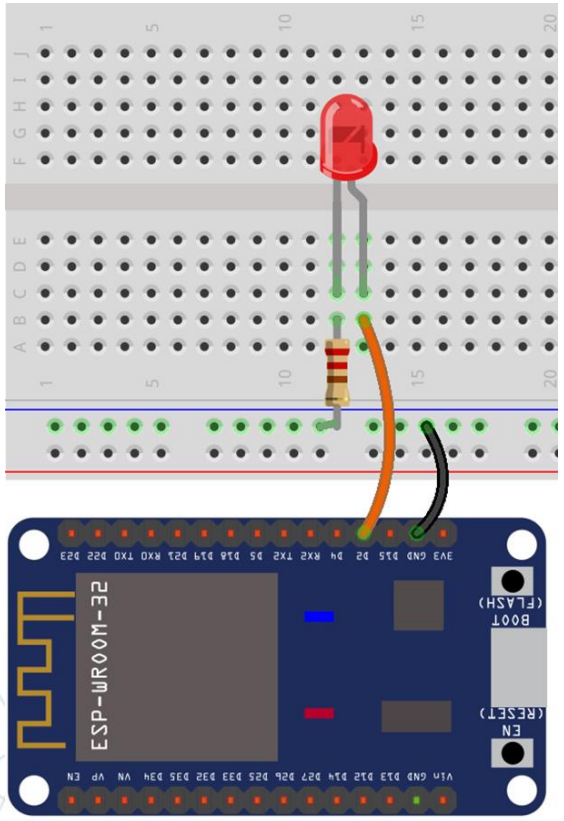
O LED deve ficar piscando em intervalos de 1s.

```
from machine import Pin
from time import sleep

led = Pin(2, Pin.OUT)

while True:
    led.value(1)
    sleep(1)
    led.value(0)
    sleep(1)
```

- Biblioteca que reconhece o ESP
- Biblioteca que implementa o *delay*
- Define a variável led (pino 2) como saída
- Enquanto for verdade (*loop* infinito)
- led recebe 1 (acende)
- Aguarda 1s
- led recebe 0 (apaga)

Sistemas Embarcados

- Node MCU
  - Pisca LED – Tratamento de exceções



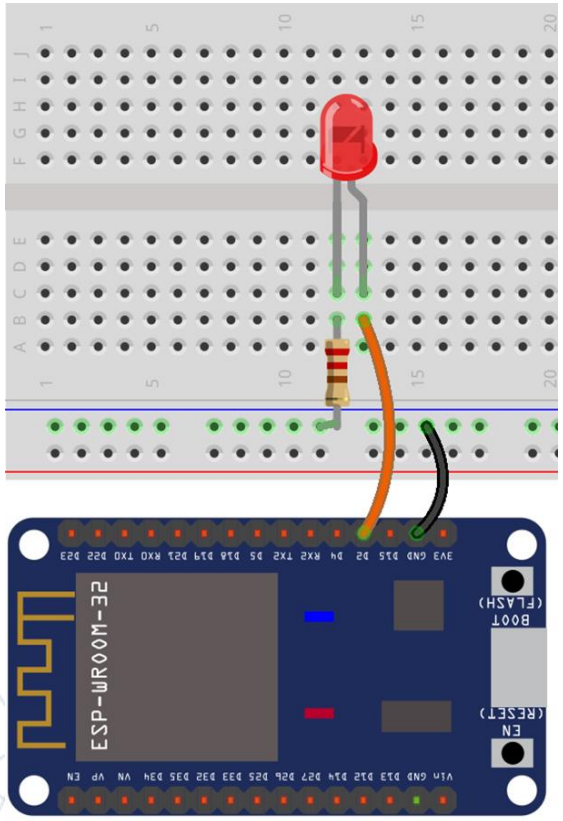
Como fazer para o LED ficar sempre apagado se a execução for interrompida?

```
from time import sleep
from machine import Pin

led = Pin(2, Pin.OUT)
try:
    while True:
        led.value(1)
        sleep(0.5)
        led.value(0)
        sleep(0.5)
except KeyboardInterrupt:
    led.value(0)
```

Tratamento de exceção

Sistemas Embarcados

- Node MCU
  - Pisca LED – Tratamento de exceções

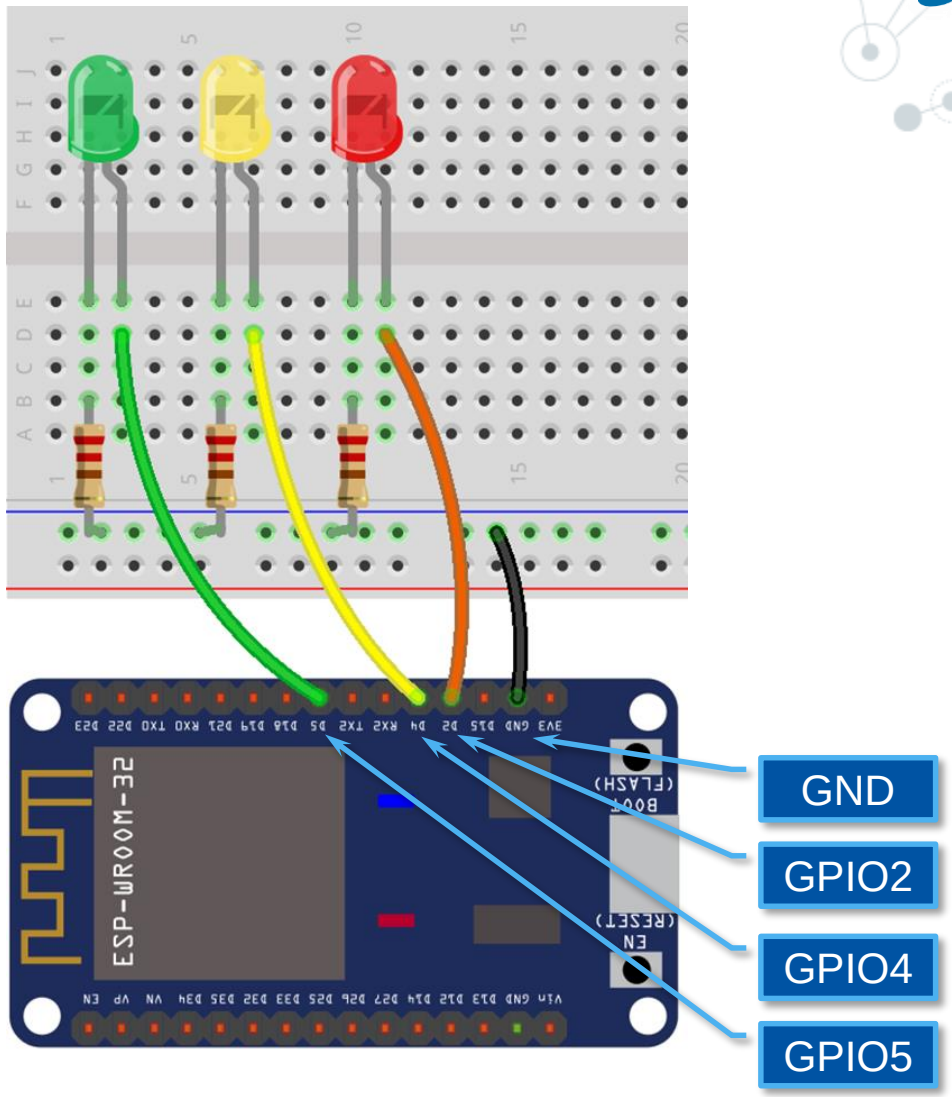


Contagem de tempo

```
from time import sleep
from machine import Pin

led = Pin(2, Pin.OUT)
cont = 0
aceso = 1
try:
   while True:
      cont += 1
      sleep(0.1)
      if cont == 10:
         aceso = not aceso
         led.value(aceso)
         cont = 0
except KeyboardInterrupt:
   led.value(0)
```

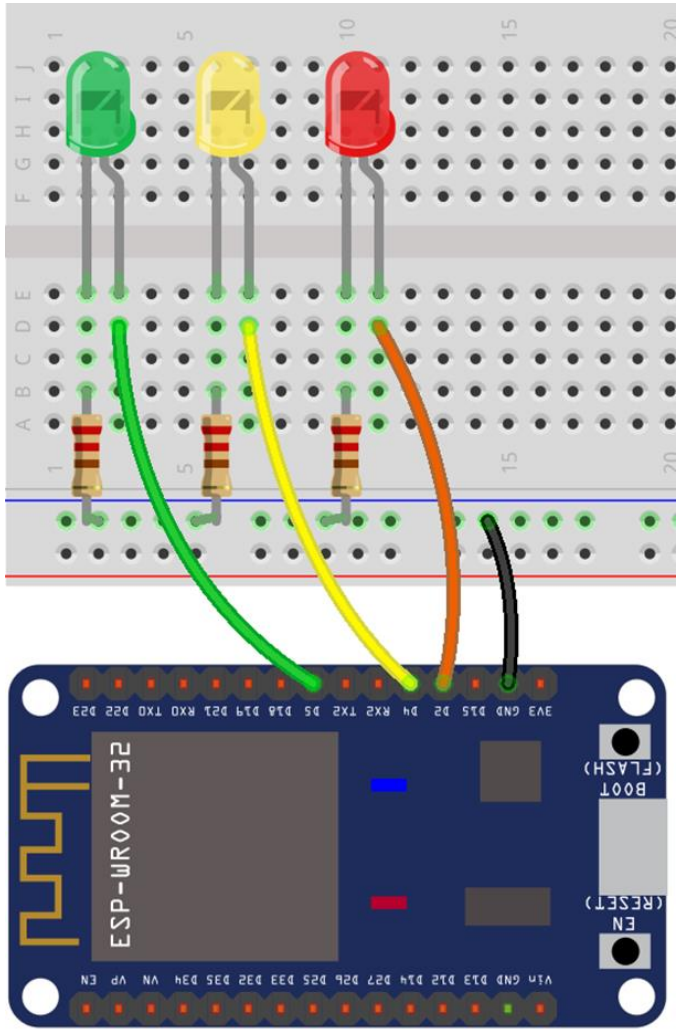
Sistemas Embarcados

- Node MCU
  - Semáforo
    - Materiais
      - Node MCU
      - 3 Resistores 220Ω
      - 1 LED vermelho
      - 1 LED verde
      - 1 LED amarelo
      - 1 Protoboard
      - Fios e jumpers



Sistemas Embarcados

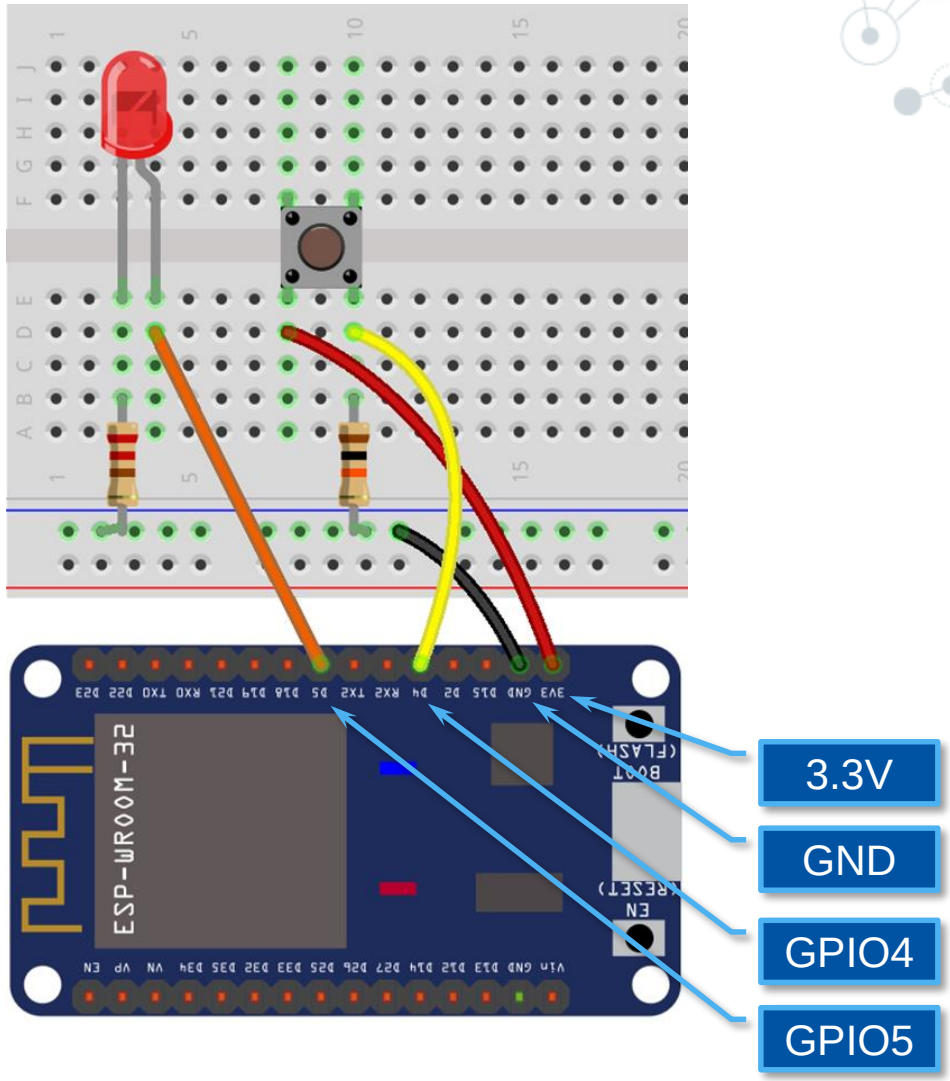
- Node MCU
  - Semáforo




```
from time import sleep
from machine import Pin

led_vm = Pin(2, Pin.OUT)
led_am = Pin(4, Pin.OUT)
led_vd = Pin(5, Pin.OUT)

try:
    while True:
        led_vm.value(1)
        sleep(0.5)
        led_vd.value(1)
        led_vm.value(0)
        sleep(0.5)
        led_am.value(1)
        led_vd.value(0)
        sleep(0.5)
        led_am.value(0)
except KeyboardInterrupt:
    led_vm.value(0)
    led_am.value(0)
    led_vd.value(0)
```

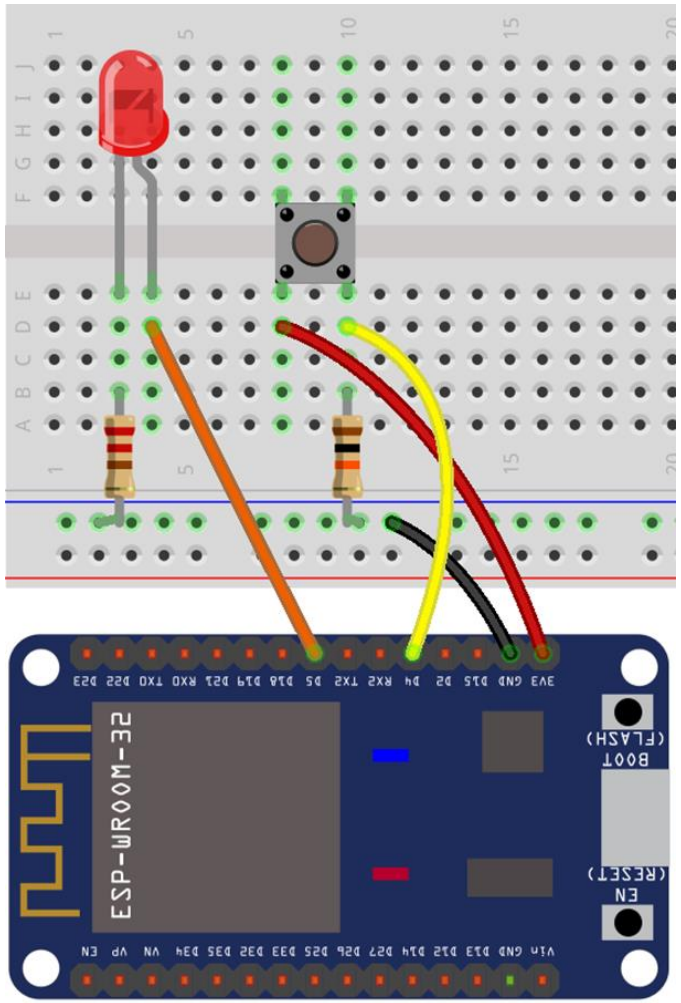
Sistemas Embarcados

- Node MCU
  - Entrada Digital
    - Materiais
      - Node MCU
      - 3 Resistores 220Ω
      - 1 Resistor 10 *k*Ω
      - 1 LED
      - 1 Chave *push button*
      - 1 Protoboard
      - Fios e jumpers



Sistemas Embarcados

- Node MCU
  - Entrada Digital

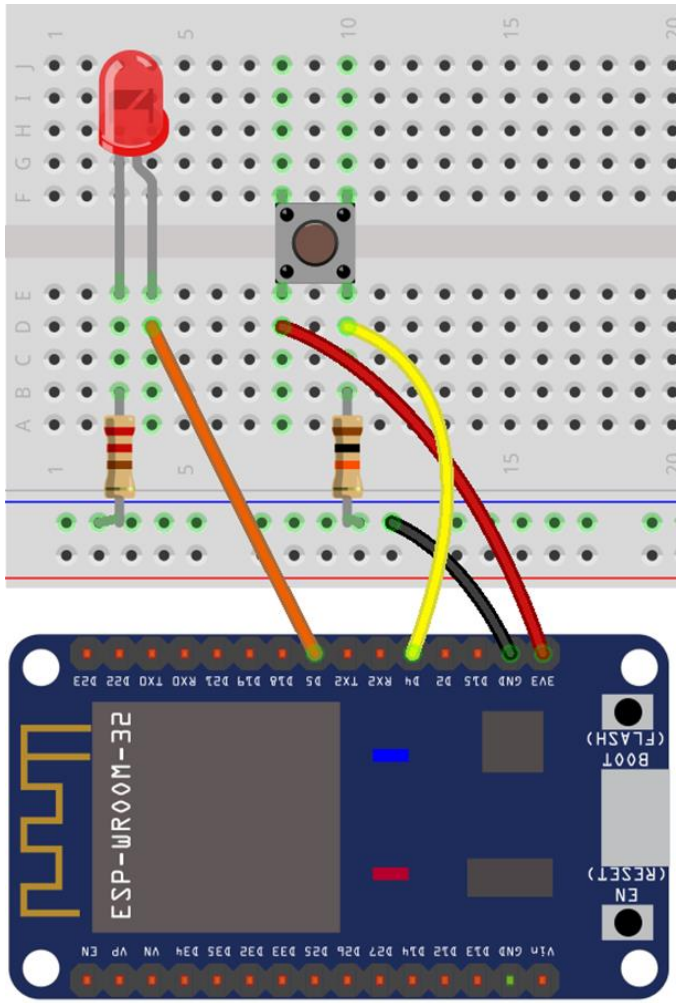


O LED deve ser aceso quando o botão for acionado.

```
from time import sleep
from machine import Pin

led = Pin(5, Pin.OUT)
botao = Pin(4, Pin.IN)

try:
   while True:
      led.value( botao.value() )
      sleep(0.2)
except KeyboardInterrupt:
   led.value(0)
```

Sistemas Embarcados

- Node MCU
  - Entrada Digital

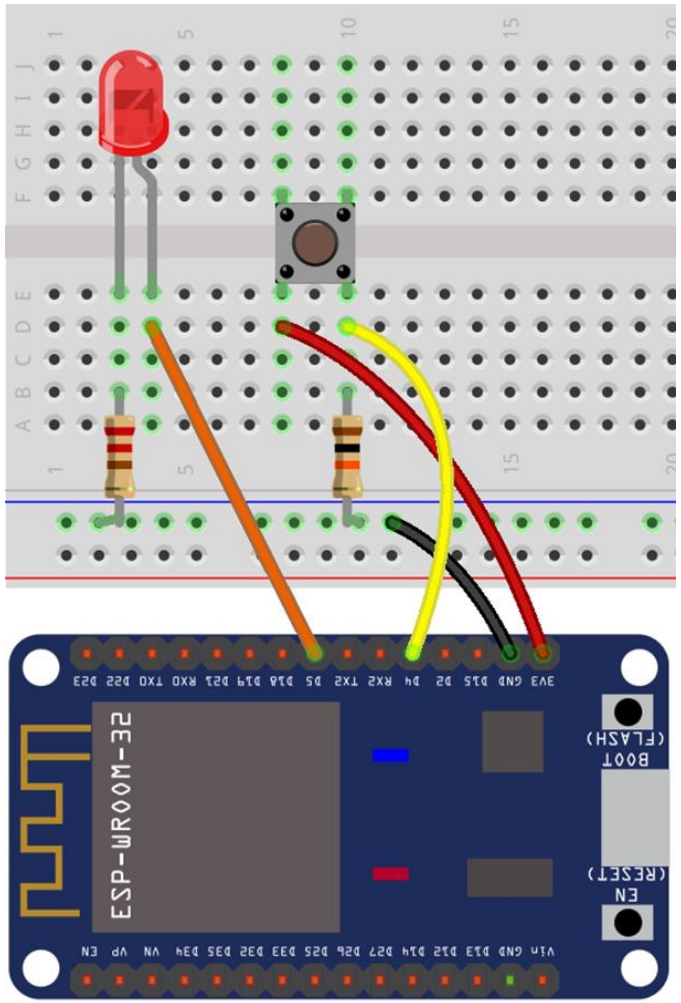


O LED deve ficar piscando e apagar quando o botão for acionado.

```
from time import sleep
from machine import Pin

led = Pin(5, Pin.OUT)
botao = Pin(4, Pin.IN)
estado = 1

while not botao.value():
    led.value(estado)
    sleep(0.2)
    estado = not estado
led.value(0)
```

Sistemas Embarcados

- Node MCU
  - Entrada Digital

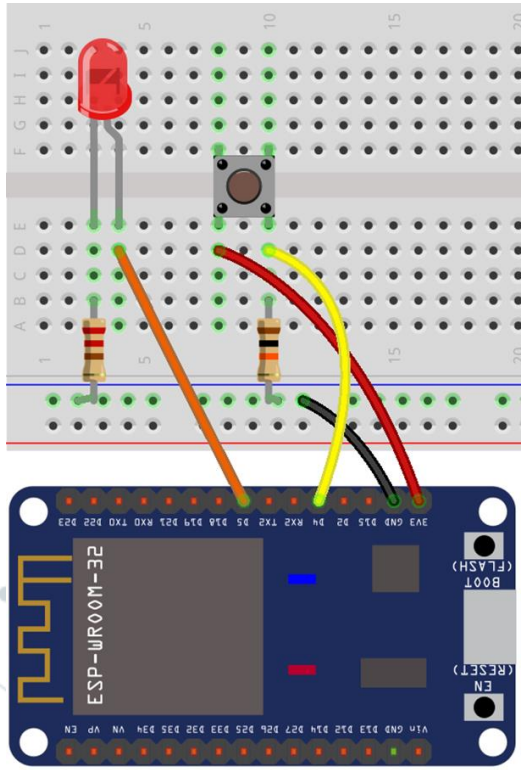


Se estiver apagado, o LED deve acender quando o botão for acionado e vice-versa.

```
from time import sleep
from machine import Pin

led = Pin(5, Pin.OUT)
botao = Pin(4, Pin.IN)
estado = 0
anterior = 0

try:
   while True:
      valor = botao.value()
      if valor == 1 and anterior == 0:
         estado = not estado
         led.value(estado)
      anterior = valor
      sleep(0.2)
except KeyboardInterrupt:
   led.value(0)
```

Sistemas Embarcados

- Node MCU
  - Entrada Digital com Interrupção



Se estiver apagado, o LED deve acender quando o botão for acionado e vice-versa.

```
from time import sleep
from machine import Pin

def handle_interrupt(pin):  # Função acionada quando houver uma interrupção
    global estado # Variável global 'estado' para poder alterá-la dentro da
função
    estado = not estado  # Alterna o 'estado' (de 0 para 1 ou de 1 para 0)
    print(estado)
    sleep(0.2)

led = Pin(5, Pin.OUT)  # Configura o pino 2 como saída (para acionar o LED)
botao = Pin(4, Pin.IN) # Configura o pino 4 como entrada (para o botão)

# Configura uma interrupção no pino do botão, acionando a função
'handle_interrupt' quando houver uma borda de subida (transição de 0 para 1)
botao.irq(trigger=Pin.IRQ_RISING, handler=handle_interrupt)

estado = 0

try:
    while True:
        led.value(estado)
except KeyboardInterrupt:
    led.value(0)
```