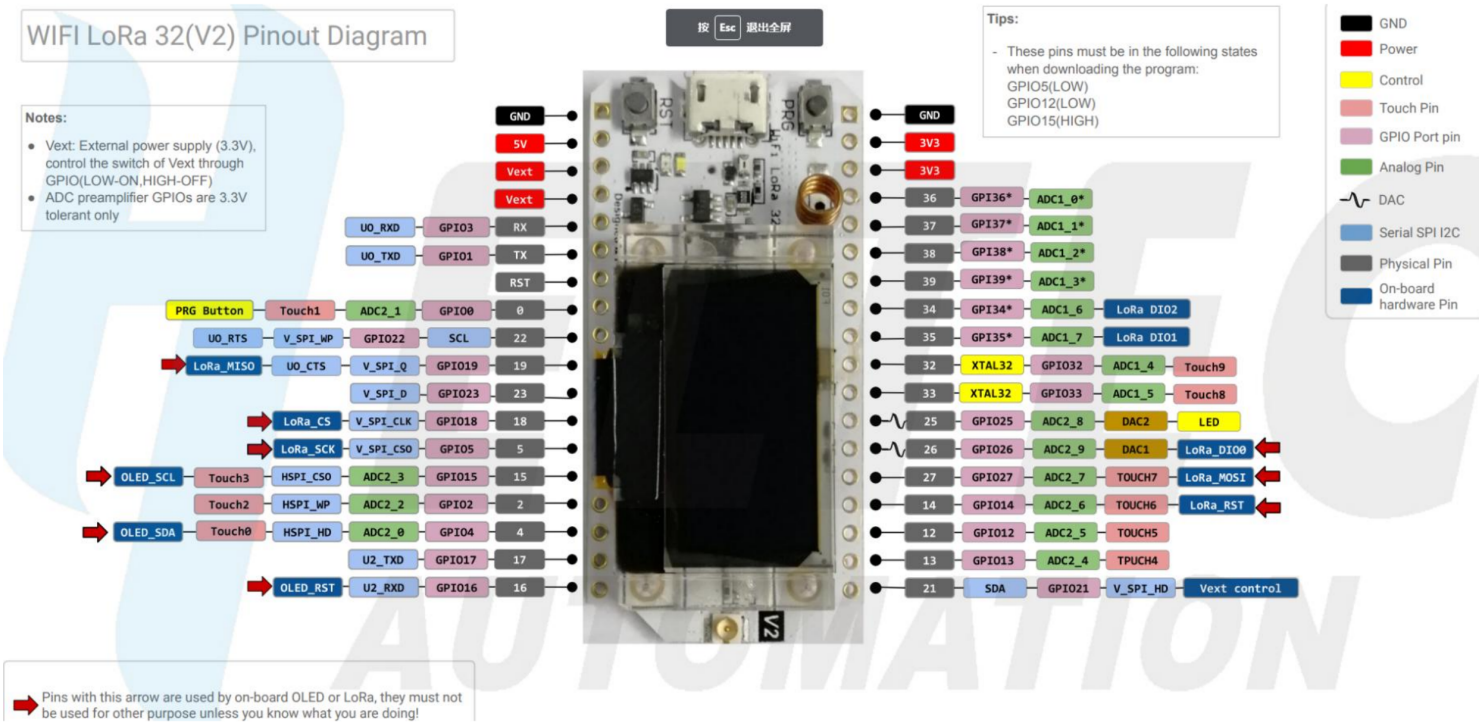
ESP32 et utilisation de µPython

Schéma d'interconnexion



https://resource.heltec.cn/download/WiFi_Kit_32/WIFI_Kit_32_pinoutDiagram_V2.pdf

Exploitation du bouton « *PROG* »

```
from machine import Pin
from utime import sleep

button = Pin(?? , Pin.IN)
led = Pin(?? , Pin.OUT)

while 1:
    if button.value() == 0:
        print("bouton appuye")
        led.value(1)
    sleep(0.01)
    led.value(0)
```

broche ❶

broche ❷

1. Identifiez la broche connectée à la led ❷ et au bouton ❶ sur le schéma de d'interconnexion.
2. Essayez le programme suivant :

```
from machine import Pin

def handle_interrupt(pin):
    print("bouton !")

bouton = Pin(0, Pin.IN)
bouton.irq(trigger=Pin.IRQ_RISING, handler=handle_interrupt)
```

Comment fonctionne-t-il par rapport au programme précédent ?

Exploitation de l'écran OLED

Vous récupérerez la bibliothèque suivante : `https://raw.githubusercontent.com/micropython/micropython-lib/master/micropython/drivers/display/ssd1306/ssd1306.py`

Vous l'installerez sur l'ESP32.

Vous essaierez le programme suivant :

```
import ssd1306
from machine import SoftI2C, Pin
import time

def init_oled():
   rst = Pin(16, Pin.OUT)
   rst.value(1)
   scl = Pin(15, Pin.OUT, Pin.PULL_UP)
   sda = Pin(4, Pin.OUT, Pin.PULL_UP)
   i2c = SoftI2C(scl=scl, sda=sda, freq=450000)
   oled = ssd1306.SSD1306_I2C(128, 64, i2c, addr=0x3c)
   test_screen(oled)
   screen = [" "," "," "," "," "]
   return oled, screen

def write_screen(oled, screen):
   oled.fill(0)
   for row, text in enumerate(screen):
      print("{} - {}".format(row, text))
      line = 12 * row
      oled.text(text, 0, line, 1)
   oled.show()

def test_screen(oled):
   oled.fill(0)
   oled.fill_rect(0, 0, 32, 32, 1)
   oled.fill_rect(2, 2, 28, 28, 0)
   oled.vline(9, 8, 22, 1)
   oled.vline(16, 2, 22, 1)
   oled.vline(23, 8, 22, 1)
   oled.fill_rect(26, 24, 2, 4, 1)
   oled.text('MicroPython', 40, 0, 1)
   oled.text('SSD1306', 40, 12, 1)
   oled.text('OLED 128x64', 40, 24, 1)
   oled.text(str(int(time.time())),40, 36, 1)
   oled.show()

   time.sleep(3)
   oled.fill(0)
   oled.show()
```

3. Que fait-il ? Que fait la fonction `test_screen()` ?

4. Essayez le programme suivant :

```
from machine import Pin, ADC, Timer
adc = ADC(Pin(13,Pin.IN))

def envoyer_valeur(t):
    v = adc.read()
    print(v)

t = Timer(-1)
t.init(period=1000,mode=Timer.PERIODIC, callback=envoyer_valeur)
```

   Que fait-il ?

   Écrivez un chrono précis à la seconde près et enregistrant deux temps intermédiaire sur appui sur le bouton de l'ESP32.

   L'affichage des différents temps se fera sur l'écran OLED.