



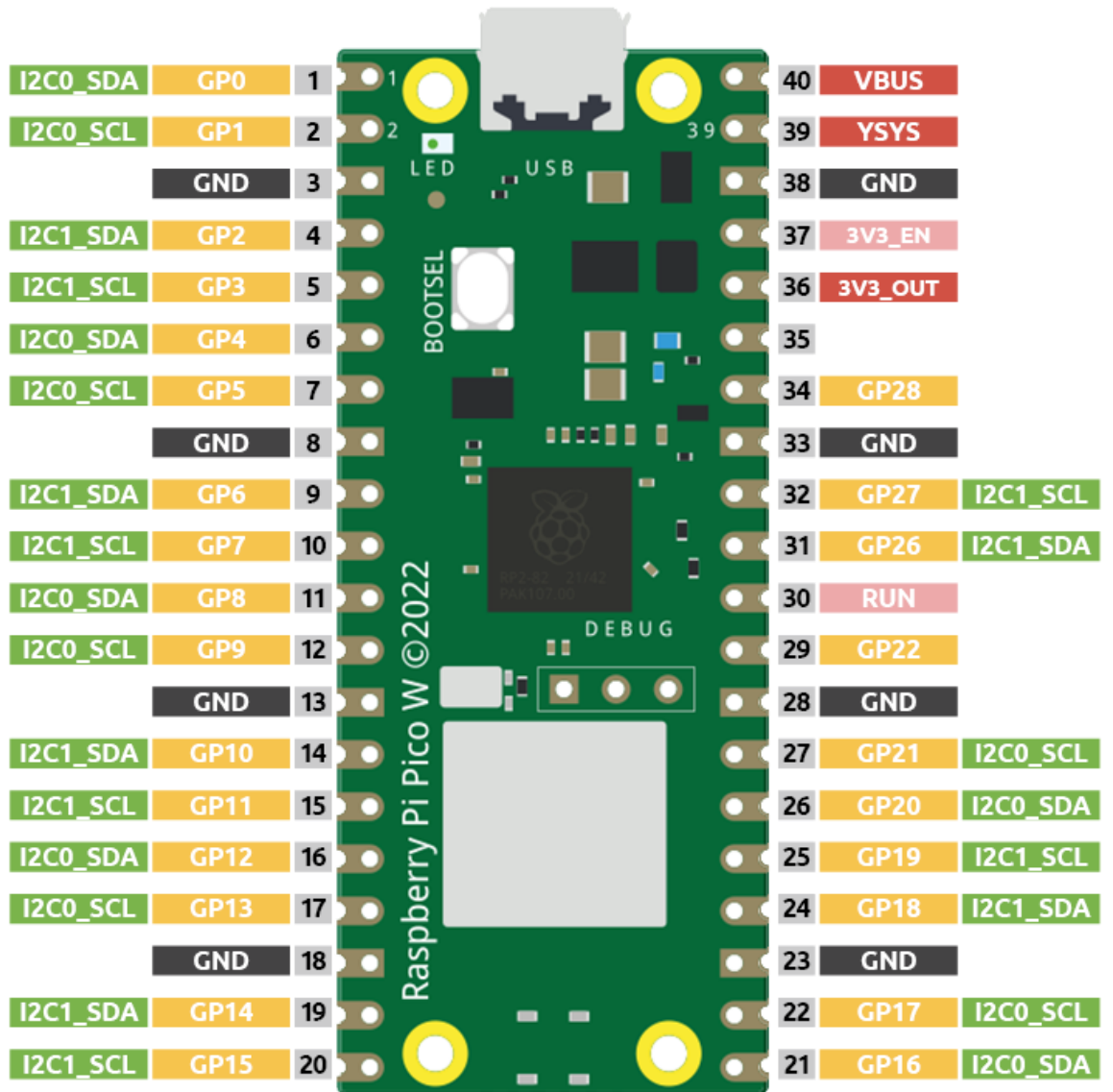
# Physical computing with the Pico

## Worksheet

### CONTENTS

Pi Pico Pin Diagram.....	2
DEMO PROJECT : ONBOARD LED LIGHT.....	3
DEMO PROJECT : CONTROLLING A BUTTON.....	4
DEMO PROJECT : TABLE LAMP .....	5
RGB LED .....	6
DEMO PROJECT : FLOWLIGHTS .....	7
DEMO PROJECT : MOTION DETECTION.....	8
DEMO PROJECT : HEARTBEAT .....	9
DEMO PROJECT : TRAFFIC LIGHTS.....	10
DEMO PROJECT: MONITORING TEMPERATURE ONBAORD PICO .....	12
DEMO PROJECT: MONITORING TEMPERATURE USING DS18B20 TEMPERATURE SENSOR .....	13
IR SENSOR .....	14
SERVO MOTOR.....	15
LCD screen 16x02 display .....	16
LED STRIP .....	17

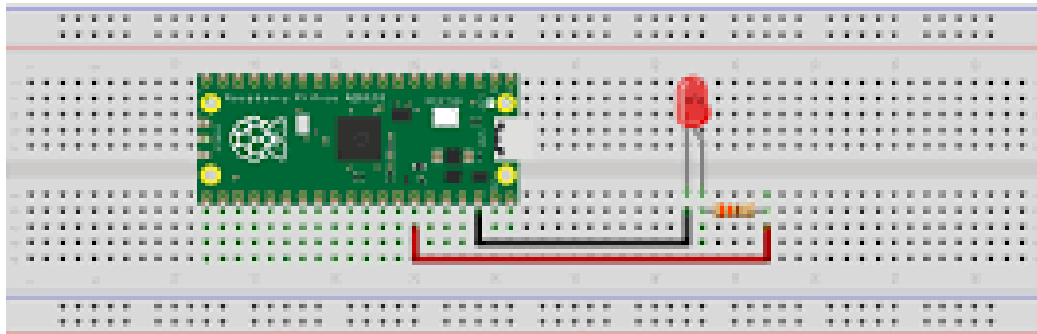
# Pi Pico Pin Diagram



## DEMO PROJECT : ONBOARD LED LIGHT

■ Gnd

■ GP28



Thonny - Raspberry Pi Pico :: /LED.PY @ 5:19

File Edit View Run Tools Help

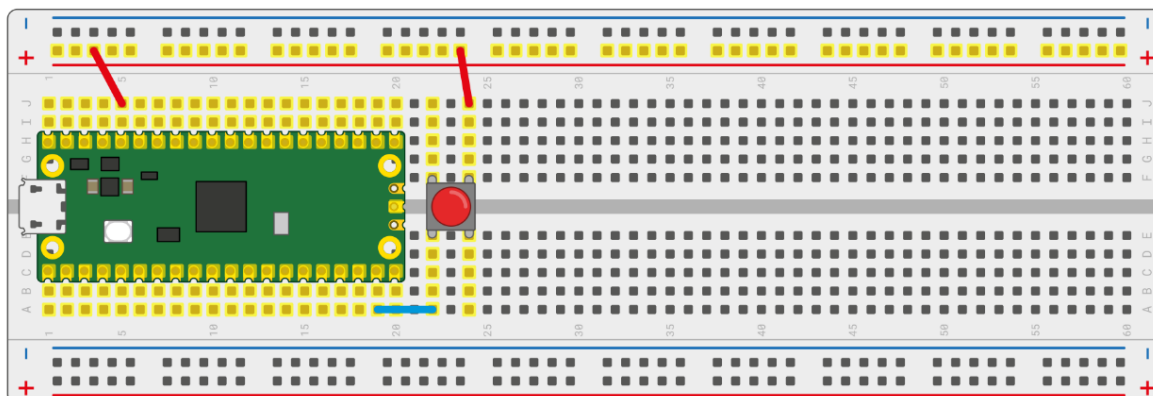


[ LED.PY ] \* x

```
1 import machine
2 import utime
3
4
5 led = machine.Pin(25, machine.Pin.OUT)
6 #pin 25 represents a LED on the pico board.
7
8 while True:
9
10     led.value(1)
11
12     utime.sleep(1)
13
14     led.value(0)
15
16     utime.sleep(1)
17
```

# DEMO PROJECT : CONTROLLING A BUTTON

Button

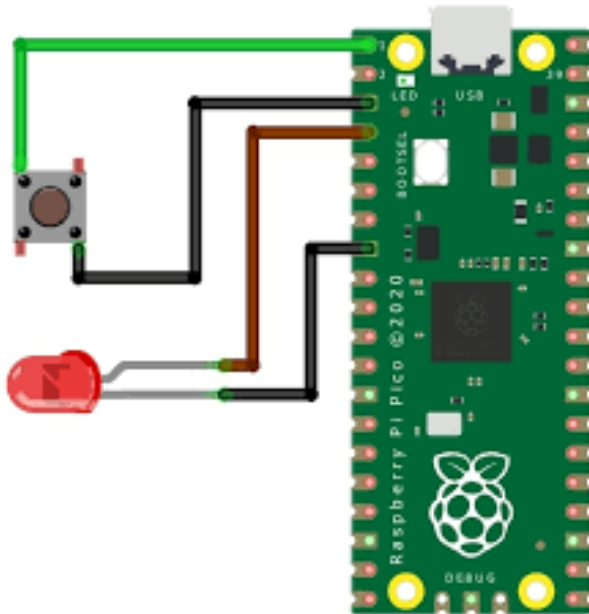


```
File Edit View Run Tools Help
Button.py x
1 import machine
2 import utime
3
4 button = machine.Pin(14, machine.Pin.IN, machine.Pin.PULL_DOWN)
5
6 while True:
7     if button.value() == 1:
8         print("You pressed the button!")
9         utime.sleep(1)
```

## BUTTON AND LED

```
Thonny - Raspberry Pi Pico :: /buttonAndLed.py @ 4:12
File Edit View Run Tools Help
[ buttonAndLed.py ] x
1 from machine import Pin
2 import time
3
4 led = Pin(15, Pin.OUT)
5 button = Pin(13, Pin.IN, Pin.PULL_UP) #Create button object from Pin13 , Set GP13 to input
6
7
8 while True:
9     if not button.value():
10        led.value(1) #Set led turn on
11    else:
12        led.value(0) #Set led turn off
13
```

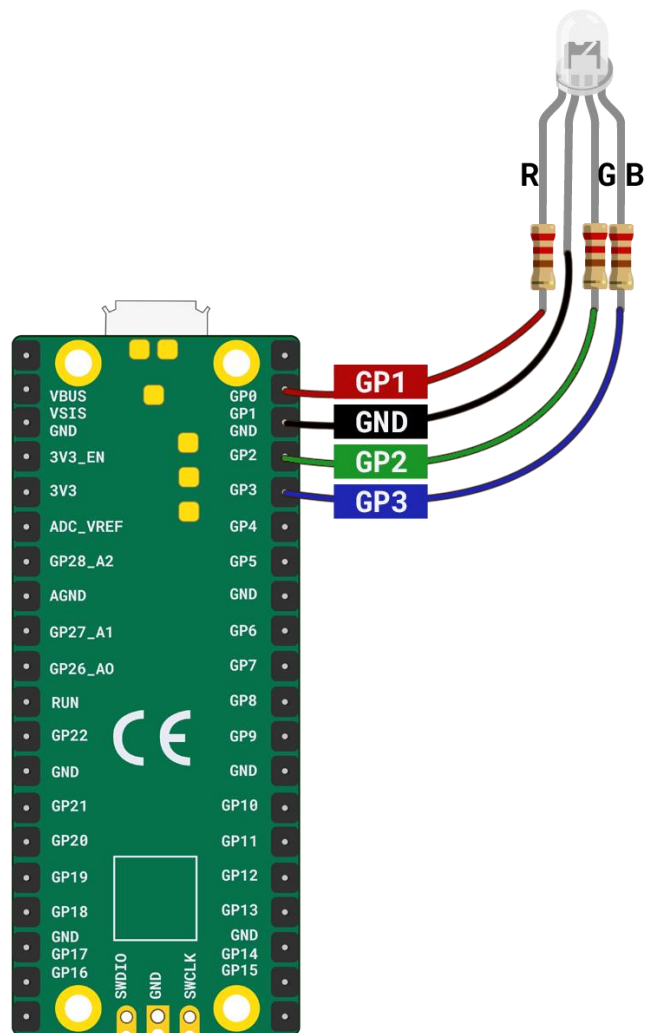
## DEMO PROJECT : TABLE LAMP



File Edit View Run Tools Help

```
02.2_TableLamp.py * x
1 from machine import Pin
2 import time
3
4 led = Pin(15, Pin.OUT)
5 button = Pin(13, Pin.IN, Pin.PULL_UP) #Create button object from Pin13 , Set GP13 to input
6
7 def reverseGPIO():
8     if led.value():
9         led.value(0) #Set led turn on
10    else:
11        led.value(1) #Set led turn off
12
13
14    while True:
15        if not button.value():
16            time.sleep_ms(20)
17            if not button.value():
18                reverseGPIO()
19                while not button.value():
20                    time.sleep_ms(20)
21
22
```

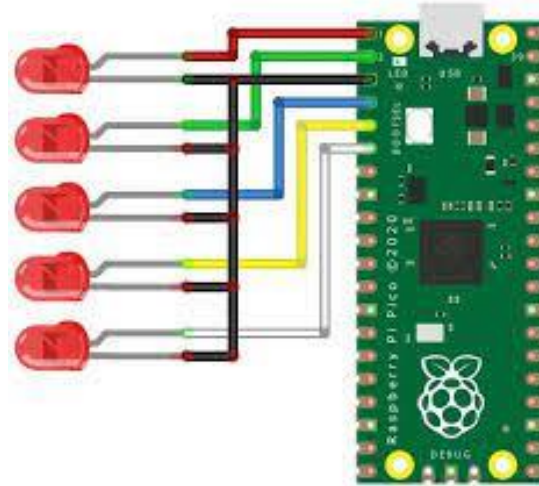
## RGB LED



[ pico\_RGB.py ] ×

```
1 from piczero import RGBLED
2 from time import sleep
3
4 rgb = RGBLED(red = 0, green = 2, blue = 3)
5
6 while True:
7     rgb.color = (255, 0, 0)
8     sleep(0.5)
9     rgb.color = (0, 255, 0)
10    sleep(0.5)
11    rgb.color = (0, 0, 255)
12    sleep(0.5)
```

## DEMO PROJECT : FLOWLIGHTS



Thonny - Raspberry Pi Pico :: /main.py @ 22:14

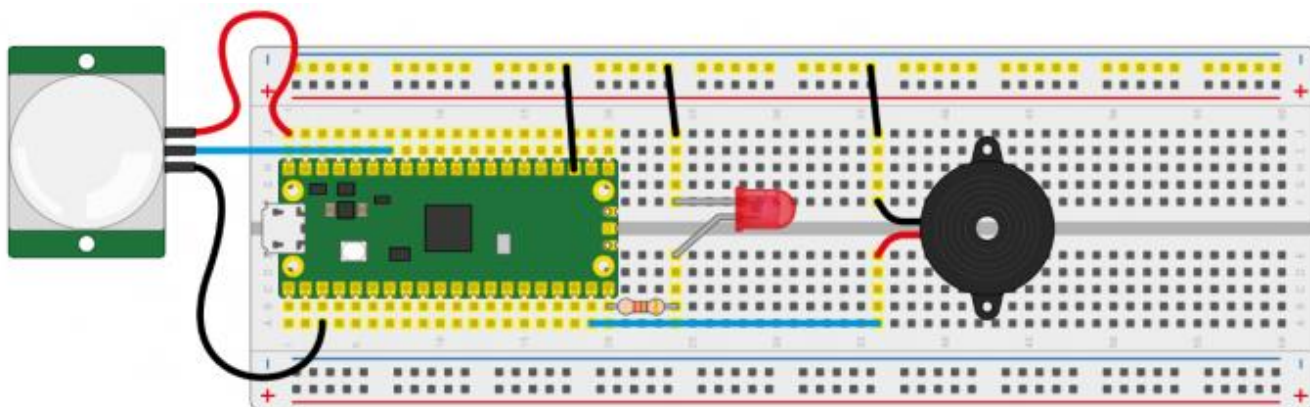
File Edit View Run Tools Help



[main.py] x

```
1 from machine import Pin
2 import time
3
4 pins = [0, 1, 2, 4, 5, 6, 7, 9, 10, 11, 12]
5 def showLed():
6     for pin in pins:
7         print(pin)
8         led = Pin(pin, Pin.OUT)
9         led.value(1)
10        time.sleep_ms(100)
11        led.value(0)
12        time.sleep_ms(50)
13    for pin in reversed(pins):
14        print(pin)
15        led = Pin(pin, Pin.OUT)
16        led.value(1)
17        time.sleep_ms(100)
18        led.value(0)
19        time.sleep_ms(50)
20
21 while True:
22     showLed()
```

## DEMO PROJECT : MOTION DETECTION



Thonny - Raspberry Pi Pico :: /main.py @ 22:1

File Edit View Run Tools Help



[ main.py ] ×

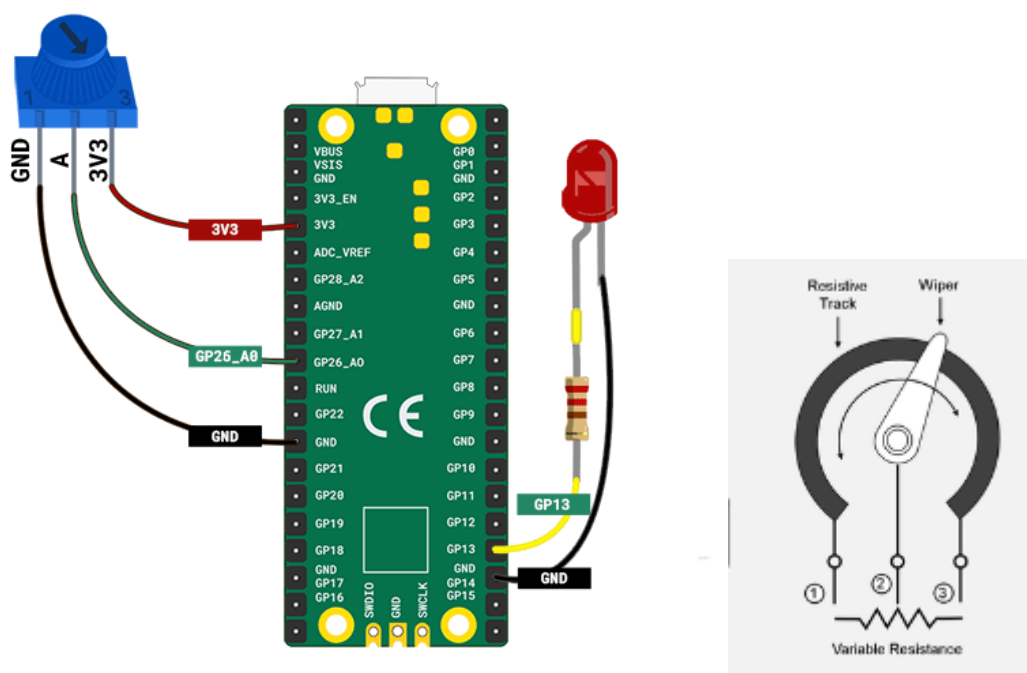
```
1 import machine
2 import utime
3
4 pir_sensor = machine.Pin(16, machine.Pin.IN)
5 led = machine.Pin(15, machine.Pin.OUT)
6 buzzer = machine.Pin(14, machine.Pin.OUT)
7
8 while True:
9
10     if pir_sensor.value()==1:
11         print("ALARM! Motion detected!") #print the message.
12         for i in range(50):
13             led.toggle()
14             buzzer.toggle()
15             utime.sleep(0.1)
16     else:
17         led.value(0)
18         buzzer.value(0)
19
20
21
22
```

Shell ×

```
>>> %RUN -C $EDITOR_CONTENT
```

```
ALARM! Motion detected!
ALARM! Motion detected!
```

# DEMO PROJECT : HEARTBEAT



Thony - C:\Users\MARTIAL\Documents\MEGA\CODE CLUB\PROJECTS\heartbeat.py @ 14:98

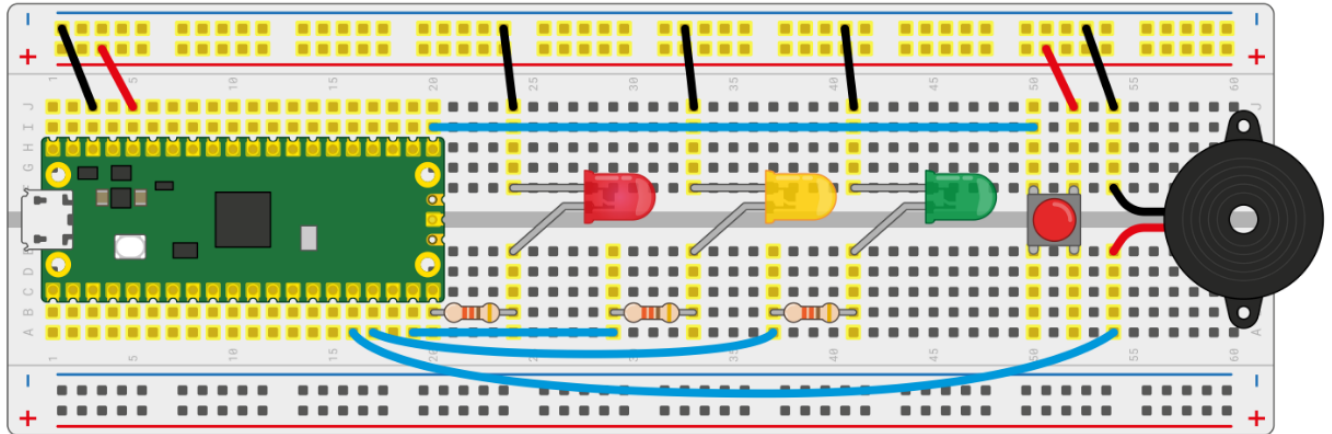
File Edit View Run Tools Help



heartbeat.py x

```
1 from picozero import Pot, LED # Add LED
2 from time import sleep
3
4 dial = Pot(0)
5 led = LED(13) # Make sure this is the correct pin
6
7 while True:
8     bpm = heart_min + dial.value * heart_range
9     print(bpm)
10    beat = 60/bpm
11    brighter_time = beat / 2 # Spend half a beat getting brighter
12    dimmer_time = beat / 2 # Spend half a beat getting dimmer
13
14    led.pulse(brighter_time, dimmer_time, n=1, wait=True) # Pulse 1 time, waiting until finished
```

## DEMO PROJECT : TRAFFIC LIGHTS



Thonny - Raspberry Pi Pico :: /main.py @ 52 : 1

File Edit View Run Tools Help

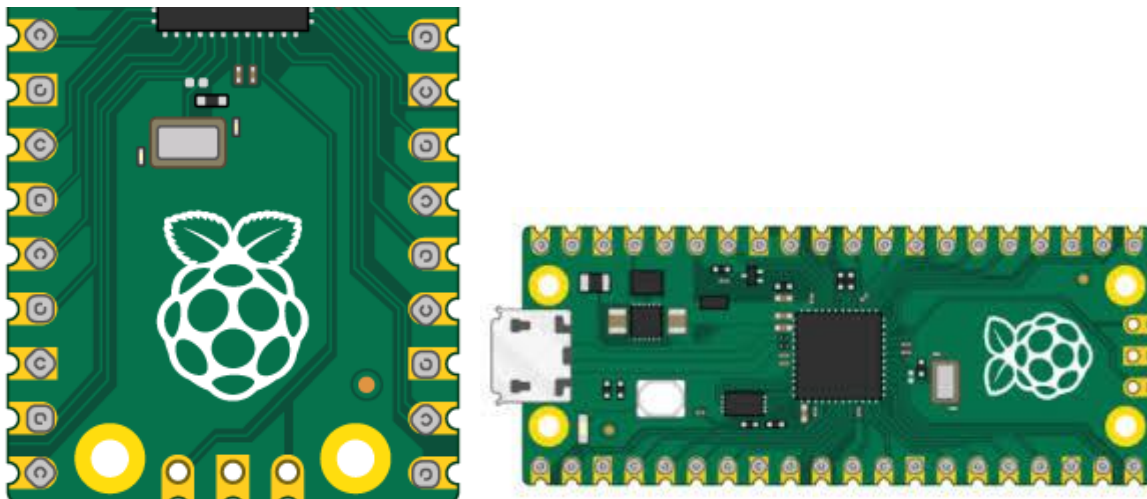


heartbeat.py × [main.py] ×

```
1 #python libraries to use the pico
2 import machine
3 import utime
4 import _thread
5
6 #configuring the pin on the pico
7 led_red = machine.Pin(16, machine.Pin.OUT)
8 led_yellow = machine.Pin(14, machine.Pin.OUT)
9 led_green = machine.Pin(13, machine.Pin.OUT)
10 button = machine.Pin(18, machine.Pin.IN, machine.Pin.PULL_DOWN)
11 buzzer = machine.Pin(15, machine.Pin.OUT)
12
13 global button_pressed
14 button_pressed = False
15
16 def button_reader_thread():
17     global button_pressed
18     while True:
19         if button.value() == 1:
20             button_pressed = True
21             utime.sleep(0.01)
22
23 _thread.start_new_thread(button_reader_thread, ())
24
```

```
25 while True:
26     #configuring the button pressed
27     if button_pressed == True:
28         led_red.value(1)
29         print("BUTTON PRESSED ")
30         for i in range(30):
31             buzzer.value(1)
32             utime.sleep(0.2)
33             buzzer.value(0)
34             utime.sleep(0.2)
35         global button_pressed
36         button_pressed = False
37
38     #configuring the led lights for the traffic lights
39     led_red.value(1)
40     utime.sleep(5)
41     led_red.value(0)
42     led_yellow.value(1)
43     utime.sleep(5)
44     led_red.value(0)
45     led_yellow.value(0)
46     led_green.value(1)
47     utime.sleep(5)
48     led_green.value(0)
49     led_yellow.value(1)
50     utime.sleep(5)
51     led_yellow.value(0)
52
```

## DEMO PROJECT: MONITORING TEMPERATURE ONBOARD PICO



Thonny - Raspberry Pi Pico :: /tem.py @ 14:1

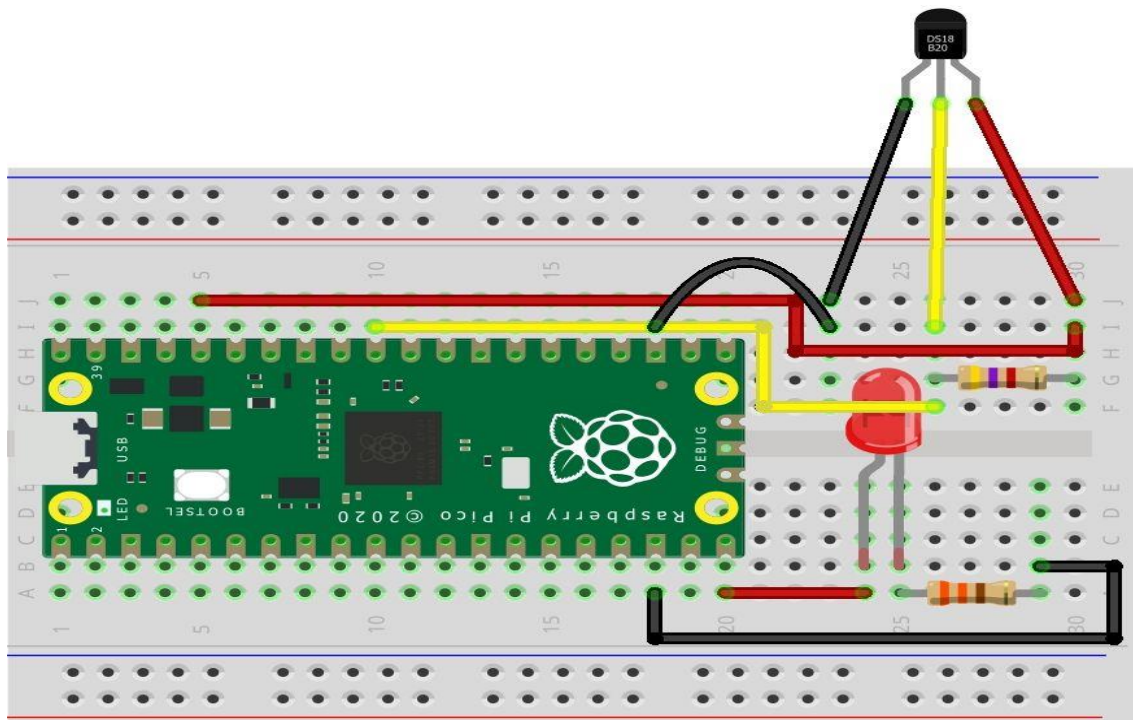
File Edit View Run Tools Help



tem.py x

```
1 import machine
2 import time
3
4 adcpin = 4
5 sensor = machine.ADC(adcpin)
6
7
8 def ReadTemperature():
9     adc_value = sensor.read_u16()
10    volt = (3.3/65535) * adc_value
11    temperature = 27 - (volt - 0.706)/0.001721
12    return round(temperature, 1)
13
14 while True:
15     temperature = ReadTemperature()
16     print(temperature)
17     time.sleep(5)
18
```

# DEMO PROJECT: MONITORING TEMPERATURE USING DS18B20 TEMPERATURE SENSOR



Thonny - Raspberry Pi Pico :: /tem.py @ 9:1

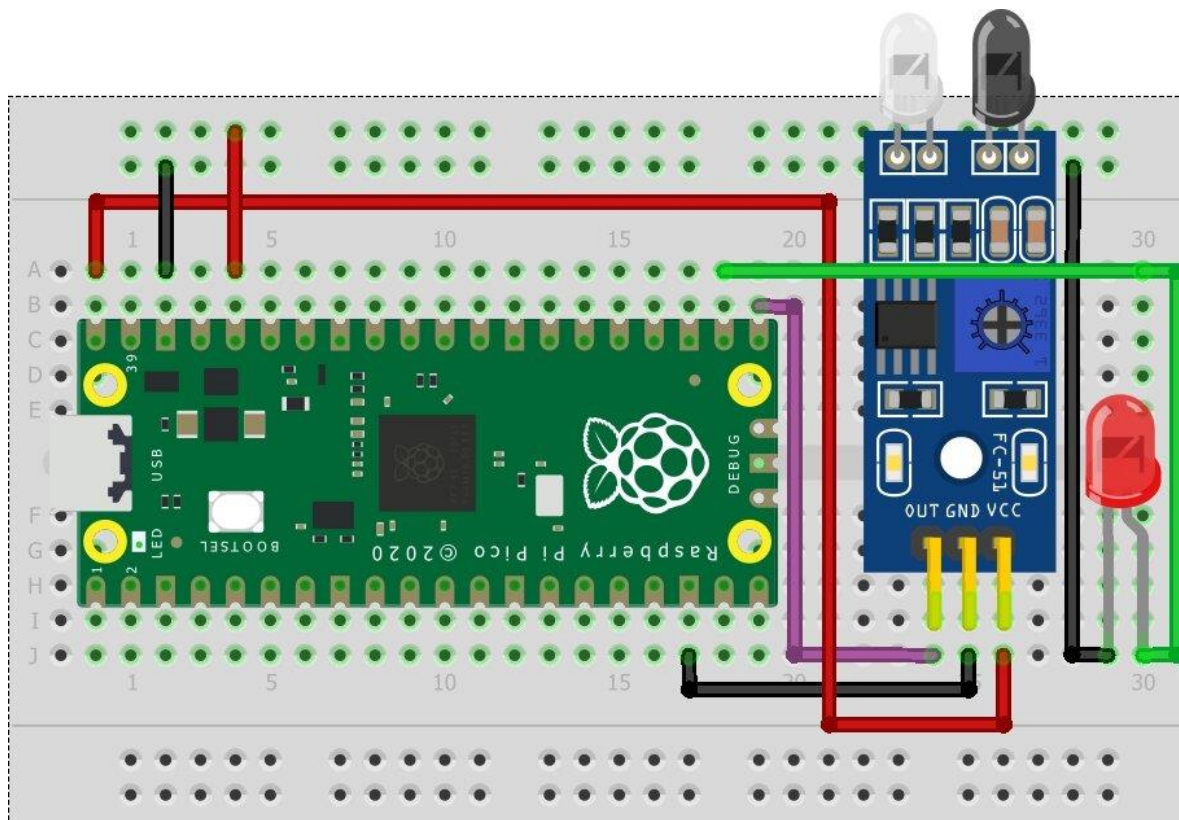
File Edit View Run Tools Help



[tem.py]\* x

```
1 import onewire, ds18x20, time
2 from machine import Pin
3
4 SensorPin = Pin(26, Pin.IN)
5 alert = Pin(15, Pin.OUT)
6 sensor = ds18x20.DS18X20(onewire.OneWire(SensorPin))
7 roms = sensor.scan()
8 print(roms)
9
10 while True:
11     sensor.convert_temp()
12     time.sleep(2)
13     for rom in roms:
14         temperature = round(sensor.read_temp(rom),1)
15         if temperature <= 20:
16             print("Warning the temperature is",temperature,"C")
17             for i in range(10):
18                 alert.toggle()
19                 time.sleep(0.5)
20         else:
21             print(temperature,"C")
22     time.sleep(5)
```

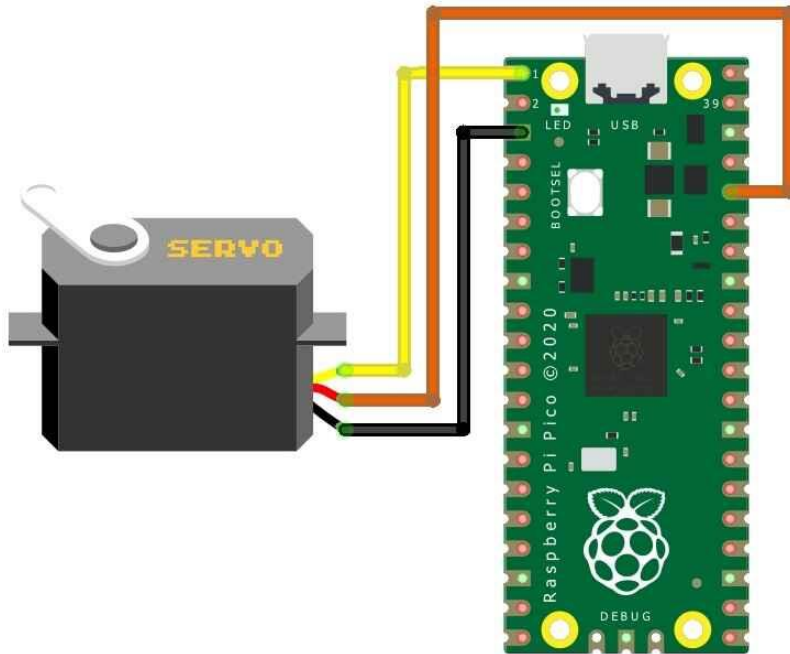
## IR SENSOR



## CODE

```
1 from machine import Pin
2 import utime
3
4 buzzer = Pin(14, Pin.OUT) # Define buzzer pin as output
5 sensor = Pin(28, Pin.IN) # Define sensor pin as input
6
7 buzzer.value(0) # Initially turn off the buzzer
8
9 while True:
10     if sensor.value() == 0:
11         buzzer.value(1) # Turn on buzzer if sensor detects something
12     else:
13         buzzer.value(0) # Turn off buzzer if sensor doesn't detect anything
14         utime.sleep_ms(10) # Add a small delay (optional)
```

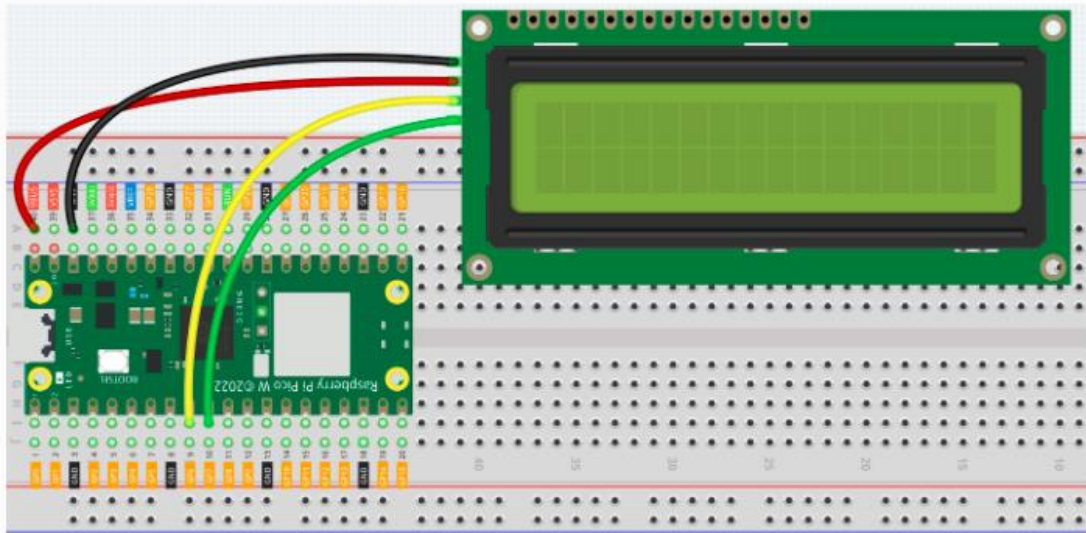
## SERVO MOTOR



servo\_motor.py ] ×

```
1 from time import sleep
2 from machine import Pin, PWM
3 pwm = PWM(Pin(1))
4 pwm.freq(50)
5
6 while True:
7     for position in range(1000,9000,50):
8         pwm.duty_u16(position)
9         sleep(0.01)
10    for position in range(9000,1000,-50 ):
11        pwm.duty_u16(position)
12        sleep(0.01)
13
```

## LCD screen 16x02 display



### Needed libraries

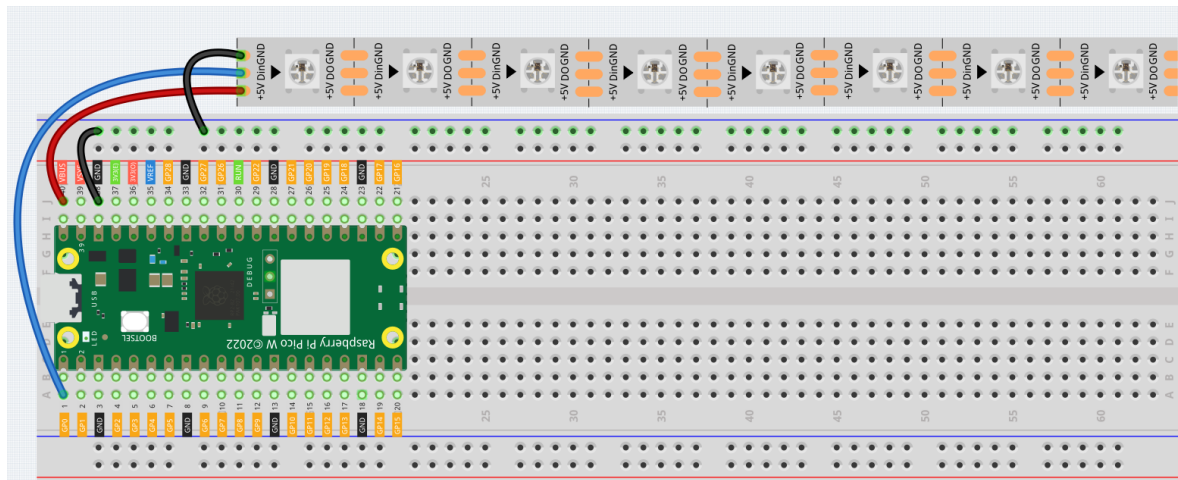
1. I2C\_LCD
2. LCD\_API

18.1\_IIC\_LCD1602.py ×

```
1 import time
2 from machine import I2C, Pin
3 from I2C_LCD import I2CLcd
4
5 i2c = I2C(1, sda=Pin(0), scl=Pin(1), freq=400000)
6 devices = i2c.scan()
7
8 try:
9     if devices != [0x27]:
10         lcd = I2CLcd(i2c, devices[0], 2, 16)
11         lcd.move_to(0, 0)
12         lcd.putstr("Hello, world!")
13         count = 0
14         while True:
15             lcd.move_to(0, 1)
16             lcd.putstr("Counter:%d" %(count))
17             time.sleep(1)
18             count += 1
19     else:
20         print("No address found")
21 except:
22     pass
```

# LED STRIP

## Wiring



**Library:** ws2812.py

```
1 import machine
2 from ws2812 import WS2812
3
4 ws = WS2812(machine.Pin(0),8)
5
6 ws[0] = [64,154,227]
7 ws[1] = [128,0,128]
8 ws[2] = [50,150,50]
9 ws[3] = [255,30,30]
10 ws[4] = [0,128,255]
11 ws[5] = [99,199,0]
12 ws[6] = [128,128,128]
13 ws[7] = [255,100,0]
14 ws.write()
```