环境检查

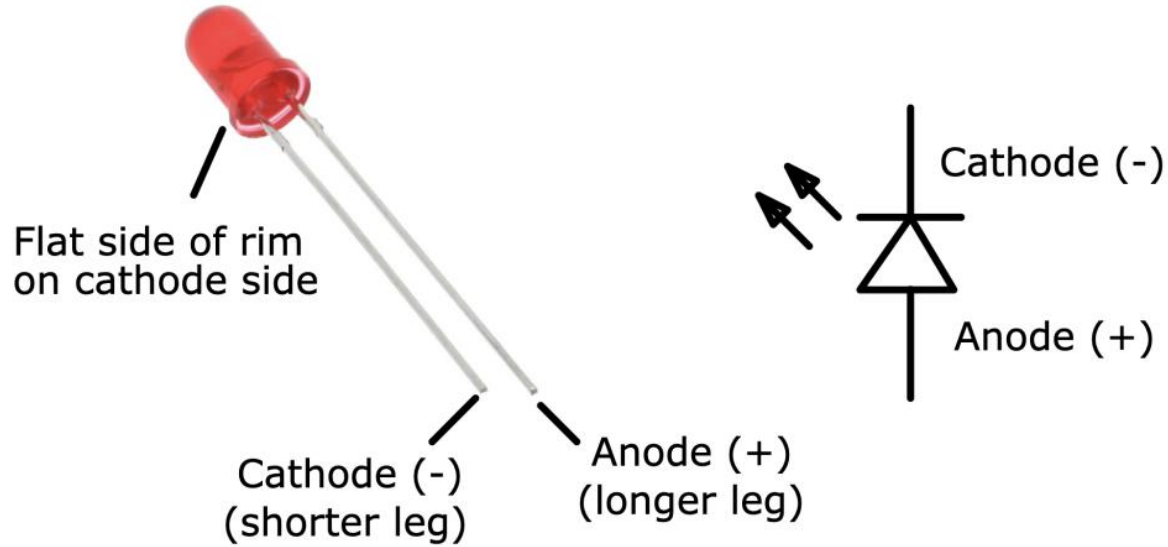
- python --version
- python3 --version
- pip list
- pip install xxxxxx
- pip uninstall xxxxxx
- pip freeze > requirements.txt
- pip install –r requirements.txt

实验2 输入输出-GPIO

2024年3月7日

实验目标

- 读懂简单电路原理图
- Python编程点亮LED
- Python编程按键控制
- 体验IoT控制LED


Flat side of rim
on cathode side
Cathode (-)
(shorter leg)
Anode (+)
(longer leg)
Cathode (-)
Anode (+)


GND


GND

I/O(Input/Output)

- 数字IO：　两种状态

  ON = HIGH = TRUE = 1

  OFF = LOW = FALSE= 0

- 模拟IO：处理的是打开、关闭或介于两者之间的状态

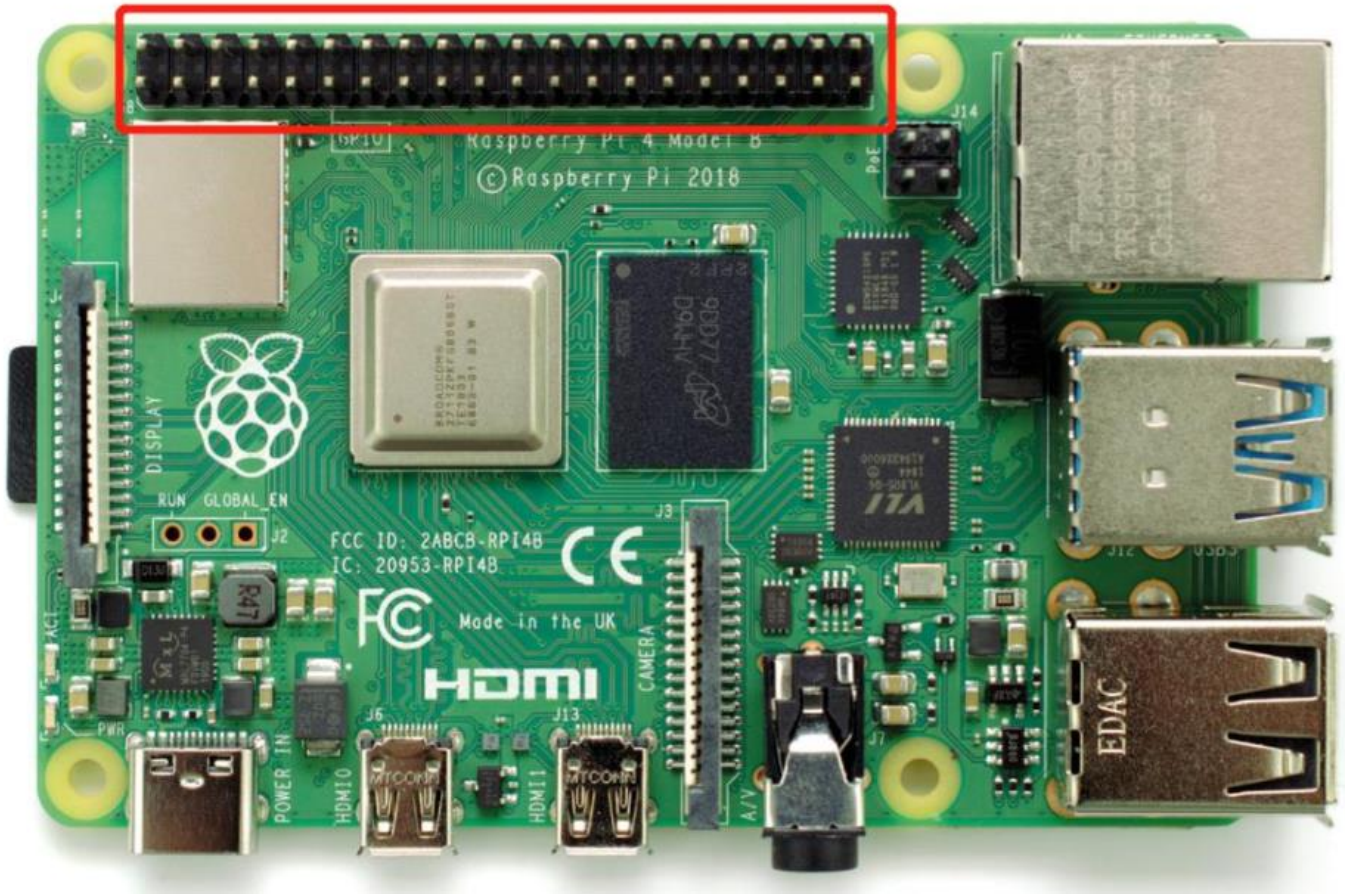
  电位器

  光敏电阻等

  树莓派本身并不具备模拟IO，需要外部ADC

GPIO

- GPIO：General Purpose Input Output，通用输入输出



Alternate Function						Alternate Function
	3.3V PWR	1		2	5V PWR	
I2C1 SDA	GPIO 2	3		4	5V PWR	
I2C1 SCL	GPIO 3	5		6	GND	
	GPIO 4	7		8	UART0 TX	
	GND	9		10	UART0 RX	
	GPIO 17	11		12	GPIO 18	
	GPIO 27	13		14	GND	
	GPIO 22	15		16	GPIO 23	
	3.3V PWR	17		18	GPIO 24	
SPI0 MOSI	GPIO 10	19		20	GND	
SPI0 MISO	GPIO 9	21		22	GPIO 25	
SPI0 SCLK	GPIO 11	23		24	GPIO 8	SPI0 CS0
	GND	25		26	GPIO 7	SPI0 CS1
	Reserved	27		28	Reserved	
	GPIO 5	29		30	GND	
	GPIO 6	31		32	GPIO 12	
	GPIO 13	33		34	GND	
SPI1 MISO	GPIO 19	35		36	GPIO 16	SPI1 CS0
	GPIO 26	37		38	GPIO 20	SPI1 MOSI
	GND	39		40	GPIO 21	SPI1 SCLK

GPIO Python库

- GPIO Zero库
  官方文档：https://gpiozero.readthedocs.io/en/stable/
- 常用的功能进行封装：LED，Button，Motor……
- Raspbian系统默认安装了GPIO Zero库
- 使用方法：

1. import gpiozero，然后通过 gpiozero.LED 来操作；
2. from gpiozero import LED，直接从 gpiozero 中引入 LED 模块，然后直接通过 LED 来操作

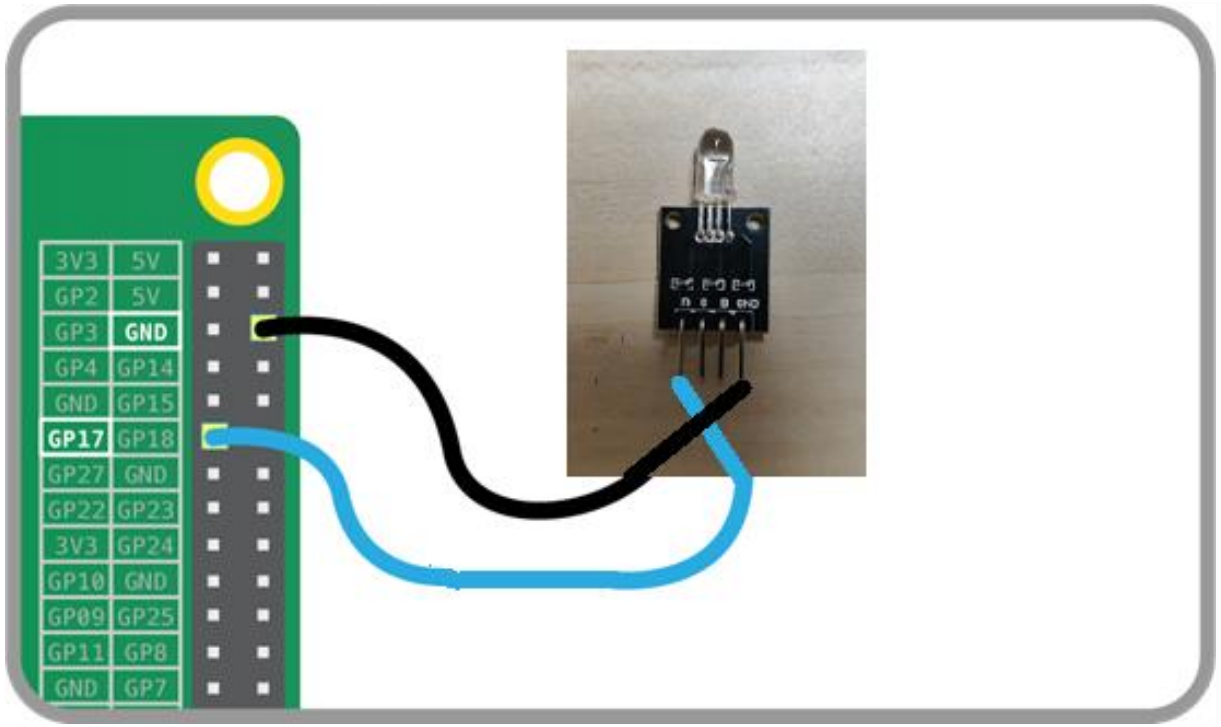
其他的库

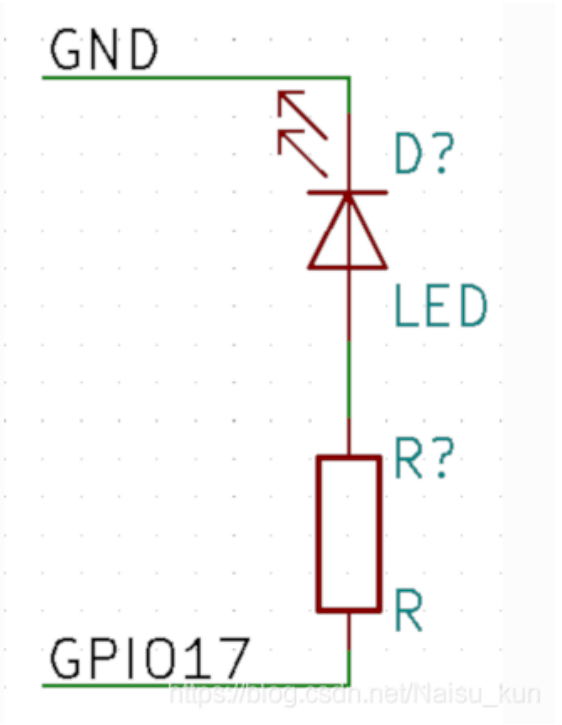
- RPi.GPIO
- PiGPIO

```
import pigpio
from time import sleep
GPIO_PIN = 17
pi = pigpio.pi()
pi.set_mode(GPIO_PIN, pigpio.OUTPUT)
while True:
    pi.write(GPIO_PIN, 1) # 1 = High = On # (4)
    sleep(1) # 1 second
    pi.write(GPIO_PIN, 0) # 0 = Low = Off # (5)
    sleep(1) # 1 second
```

LED

- 实验器件：树莓派4B，LED模块，杜邦线
- 实验目的：点亮LED
- LED模块：





示例代码（Vscode下编写python代码）

```python
from gpiozero import LED
from time import sleep

red = LED(17)

while True:
    red.on()
    sleep(1)
    red.off()
    sleep(1)
```

```python
from gpiozero import LED
from signal import pause

red = LED(17)

red.blink()

pause()
```

代码里的 while True: 让程序一直保持运行。退出按 Ctrl+Z

构造函数
gpiozero.LED(pin)
LED 对象在 gpiozero 模块下。 【pin】树莓派 GPIO， BCM 编码；
使用方法
LED.on()
点亮 LED， 输出高电平。
LED.off()
关闭 LED， 输出低电平。
LED.toggle()
LED 亮灭状态翻转，输出电平翻转
更多使用方法，请参考官方文档：

构造函数
time
时间模块，直接 import 使用。
使用方法
time.sleep(S)
延时。 S：延时秒数，可以输入 0.1，即 100ms。

LED亮度调节

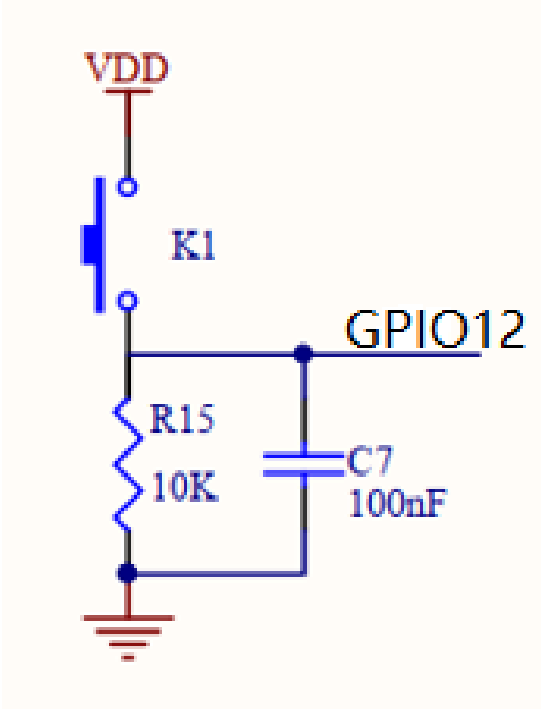
- 使用 PWM 方式来调节亮度。gpiozero 中需要导入 PWMLED，值的范围为 0 - 1

```python
from gpiozero import PWMLED
from time import sleep

led = PWMLED(17)

while True:
    led.value = 0  # off
    sleep(1)
    led.value = 0.5  # half brightness
    sleep(1)
    led.value = 1  # full brightness
    sleep(1)
```

```python
from gpiozero import PWMLED
from signal import pause

led = PWMLED(17)

led.pulse()

pause()
```

按键--Button

- 按键接入IO口，按键接通后判断该口的高低电平。当按键没有被按下时，IO口为低电平； 当按键按下时，IO口为高电平，所以只要判断口的状态即可知道按键是否被按下。使用逻辑判断的方法来控制LED亮或者灭。



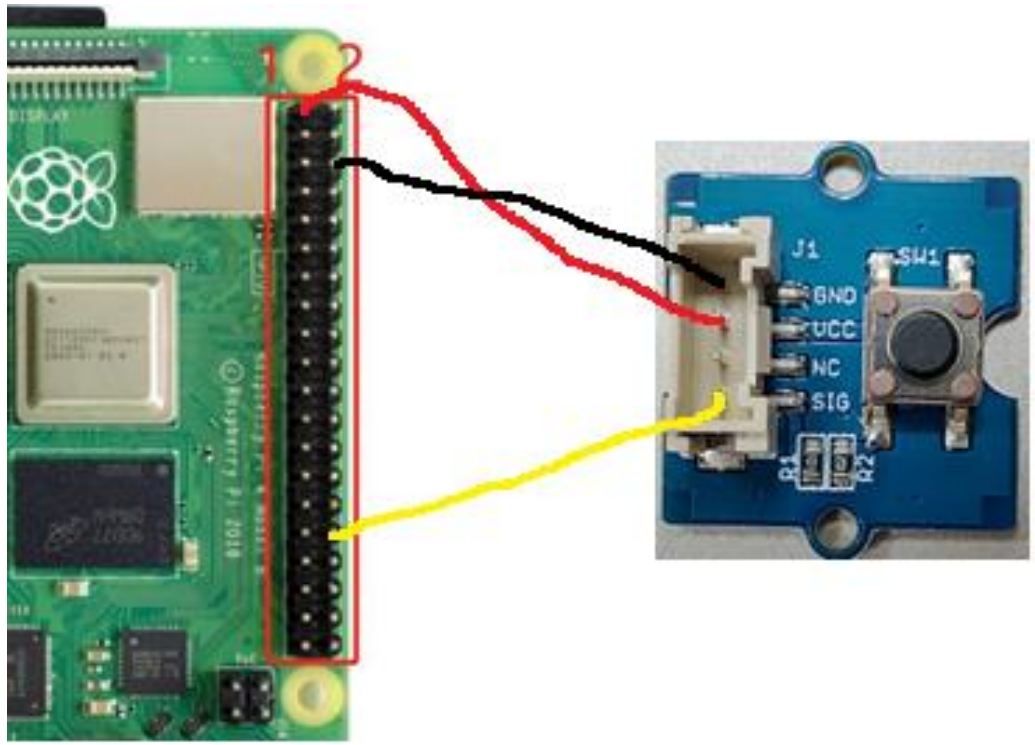
按键去抖

- 按键按下就是按下了，为什么会抖动？
- 因为按键都是机械式的，两个金属片在接触的瞬间，从微秒级的时间段来看，会存在接触-断开-再接触这样的轻微的抖动。直到两个金属片牢牢的接触到一起之后，抖动才会消失。所谓按键去抖动，就是通过延时来消除掉接触再断开这种异常状态的。
- 设置 bounce_time

```
button = button(12, bounce_time=0.2)
```

Button

- 实验器件：树莓派4B，按键模块，杜邦线
- 实验目的：检测按键
- 按键模块：



注意

- 按键模块Vcc接3.3v！！！

```
for i in range(3):
    print("GPIO pin 3V3")
```

- button模块按下是高电平3.3v，弹起是0v

13.1.1. Button

class **gpiozero.Button**(*pin*, ***, *pull_up=True*, *active_state=None*, *bounce_time=None*, *hold_time=1*, *hold_repeat=False*, *pin_factory=None*) [source]

Extends `DigitalInputDevice` and represents a simple push button or switch.

Connect one side of the button to a ground pin, and the other to any GPIO pin. Alternatively, connect one side of the button to the 3V3 pin, and the other to any GPIO pin, then set *pull_up* to `False` in the `Button` constructor.

构造函数
gpiozero.Button(pin,pull_up=False)
Button 对象在 gpiozero 模块下。 【pin】树莓派 GPIO， BCM 编码； 【pull_up】配置上拉/下拉/悬空： True:配置上拉； False:下拉； None:悬空。
使用方法
Button.is_press
判断按键是否被按下， 返回值： 【True】 即【0】， 被按下； 【False】 即【1】， 没被按下。
更多 Button 对象介绍，请看官方文档： https://gpiozero.readthedocs.io/en/stable/api_input.html#button

```
button = Button(12,False)
```

构造函数
gpiozero.Button(pin， pull_up=False)
Button 对象在 gpiozero 模块下。 【pin】树莓派 GPIO， BCM 编码； 【pull_up】配置上拉/下拉/悬空： True:配置上拉； False:下拉； None:悬空。
使用方法
Button.when_pressed = fun
当按键被按下时，执行函数 fun()：
更多 Button 对象介绍，请看官方文档： https://gpiozero.readthedocs.io/en/stable/api_input.html#button

示例代码

```python
from gpiozero import Button
from time import sleep

button = Button(2)

while True:
    if button.is_pressed:
        print("Button is pressed")
    else:
        print("Button is not pressed")
    sleep(1)
```

```python
from gpiozero import Button
from signal import pause

def say_hello():
    print("Hello!")

button = Button(2)

button.when_pressed = say_hello    # 此处是 say_hello 而不是 say_hello()

pause()
```

IoT控制LED

- 运行例程dweet_led.py
- 注意：需要更改thing name和GPIO

```
INFO:main:Created new thing name a8e38712                    # (1)
LED Control URLs - Try them in your web browser:
 On     : https://dweet.io/dweet/for/a8e38712?state=on       # (2)
 Off    : https://dweet.io/dweet/for/a8e38712?state=off
 Blink  : https://dweet.io/dweet/for/a8e38712?state=blink

INFO:main:LED off
Waiting for dweets. Press Control+C to exit.
```

作业

- 按键控制呼吸灯的开启及关闭，红绿蓝交替呼吸灯效果
- Traffic light：https://gpiozero.readthedocs.io/en/stable/recipes.html#traffic-lights
- buzzer
- 要求：按键按一下，交替呼吸灯开启
  按键再按一下，关闭
  如此循环
- 阅读参考书相关介绍，理解dweet_led.py并整理进实验报告

报告需要上传至lnt.xmu.edu.cn平台，要求同时提交相关代码及实验结果视频展示文件。截止时间：2024年3月14日