



SKU:KIT0150

Get Started with Raspberry Pi

Learning Objectives

- Get to know Raspberry Pi

Learning Contents

Introduction

What is Raspberry Pi?






The Raspberry Pi is a low cost, credit-card sized computer that plugs into a computer monitor or TV, and uses a standard keyboard and mouse. It is a capable little device that enables people of all ages to explore computing, and to learn how to program in languages like Scratch and Python. It's capable of doing everything you'd expect a desktop computer to do, from browsing the internet and playing high-definition video, to making spreadsheets, word-processing, and playing games. With the release of Windows 10 IoT, We will also be able to use the Raspberry Pi to run Windows.

Raspberry Pi was developed by the Raspberry Pi Foundation, a UK charity, with Eben·Upton in Cambridge University as the project leader. In March 2012, he officially released the world's smallest desktop computer, also known as a card computer, which is only the size of a credit card but has all the basic functions of a computer. This is the Raspberry Pi computer board. The foundation aims to improve the education of computer science and related subjects in schools to make computers interesting. And it expects that this computer will continue to be developed and applied to more fields, whether in developing or developed countries. In 2006, the early concept of Raspberry Pi was based on Atmel's ATmega644 microcontroller. The first batch of 10,000 sets of Raspberry Pi boards were made by manufacturers in Taiwan and mainland China.

It is an ARM-based microcomputer motherboard with SD/MicroSD card as the memory hard disk. There are 1/2/4 USB ports and a 10/100 Ethernet port around the card motherboard (A type has no network port), which can be connected to a Keyboard, mouse, and network cable, as well as a TV output interface for video analog signals and an HDMI high-definition video output

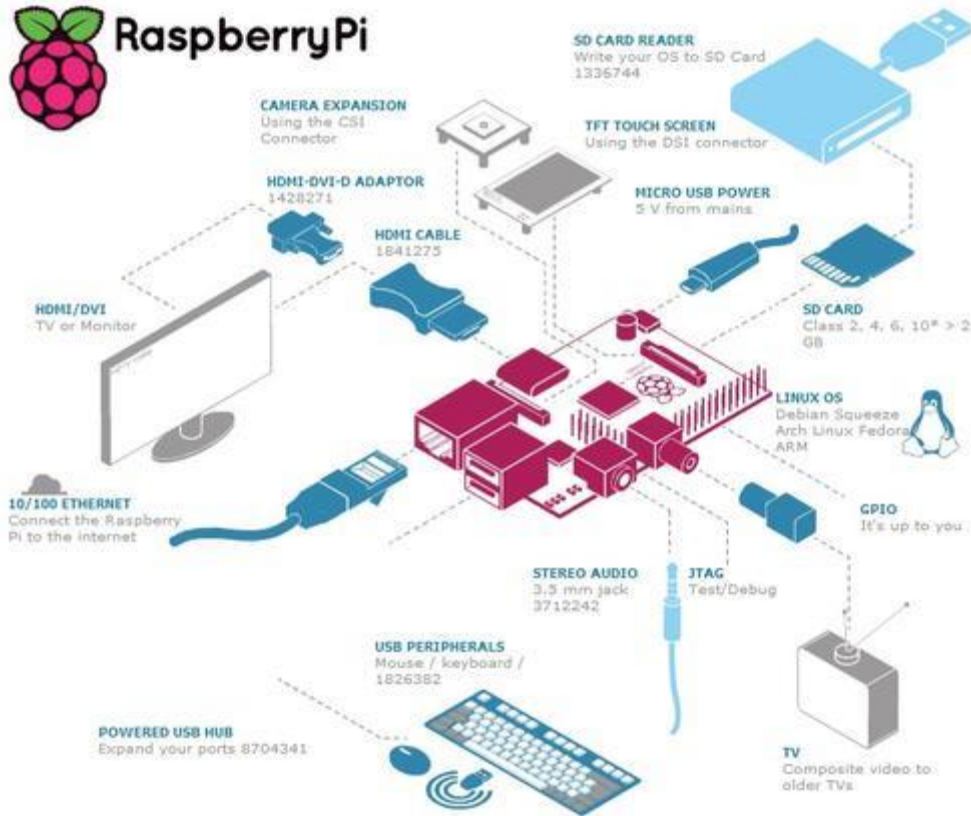
interface. The above components are all integrated on a motherboard that is only slightly larger than a credit card. It has all the basic functions of a PC. But the Raspberry Pi B model only provides a computer board without memory, power supply, keyboard, case or connection.

Differences between Raspberry Pi models

Name	Raspberry Pi 4 Module B (1GB, 2GB, 4GB)	Raspberry Pi 3 Model B+	Raspberry Pi 3 Model B	Raspberry Pi 3 Model A+ (E14)	Raspberry Pi 2 Model B
SKU	DFR0617, DFR0618, DFR0619	DFR0567	DFR0431	DFR0594	DFR0353
Price	\$35, \$45, \$55	\$39.95	\$39.95	\$25	Discontinued
Image					
SOC	BCM2711	BCM2837	BCM2837	BCM2837	BCM2836
CPU	1.5GHz, 64 bits quad-core Cortex A-72	1.4GHz, 64bits quad-core Cortex A-53	1.2GHz, 64bits quad-core Cortex A-53	1.4GHz, 64bits quad-core ARM Cortex-A53	900MHz quad-core ARM Cortex-A7
SD Card Slot	Micro SD	Micro SD	Micro SD	Micro SD	Micro SD
RAM	1GB, 2GB, 4GB	1GB	1GB	512MB	1GB
GPIO	40-pin	40-pin	40-pin	40-pin	40-pin
GPU	VideoCore IV	VideoCore IV	VideoCore IV	VideoCore IV	VideoCore IV
Max Resolution	4K, 60Hz	1080P30	R1080P30	1080P30	1080P
Video Input	Dual-lane CSI camera port	CSI camera port	CSI camera port	CSI camera port	CSI camera port
Video Output	R2 HDMI ports, dual-lane DSI port	Full-size HDMI	HDMI, DSI port	HDMI, DSI port	HDMI, DSI port
USB Port	2xUSB3.0 + 2xUSB2.0	4xUSB2.0	4xUSB2.0	1xUSB2.0	4xUSB2.0
Ethernet	Gigabit Ethernet	Gigabit Ethernet	100 Mbit Ethernet	None	100 Mbit Ethernet
Wireless Network	802.11ac(2.4/5GHz), Bluetooth 5.0	802.11b/g/n/ac(2.4/5GHz), Bluetooth 4.2	802.11n(2.4GHz), Bluetooth 4.0	802.11ac(2.4/5GHz), Bluetooth 4.2	None
Charging Port	USB Type-C	micro USB	micro USB	micro USB	micro USB
Power Supply	3A, 5V	2.5A, 5V	2.5A, 5V	2.5A, 5V	1.8A, 5V
Power Via PoE	Supported	Supported	Unsupported	Unsupported	Unsupported
Dimension	88*58*19.5mm/ 3.5*2.3*0.76"	82*56*19.5mm/ 3.2*2.3*0.76"	87*58*19mm/ 3.43*2.28*0.75"	65*56mm/ 2.56*2.20"	87*58*19mm/ 3.43*2.28*0.75"
Weight	46g	50g	45g	45g	45g

[Selection guide](#)

Raspberry Pi Peripherals



Summary

As everyone has a preliminary understanding of Raspberry Pi, now we are going to learn how to use the 37 PCs Sensor kit on Raspberry Pi 4B.

Preparation

Hardware

- [Gravity: 37 Pcs Sensor Set](#)
- [Raspberry Pi 4 Model B](#)
- [IO Expansion HAT for Raspberry Pi 4B/3B+](#)
- [8GB + SanDisk Class10 SD/MicroSD Memory Card](#)
- [5V@3A USB Power Supply](#)
- [8.9 IPS Touch Display](#)
- Card Reader

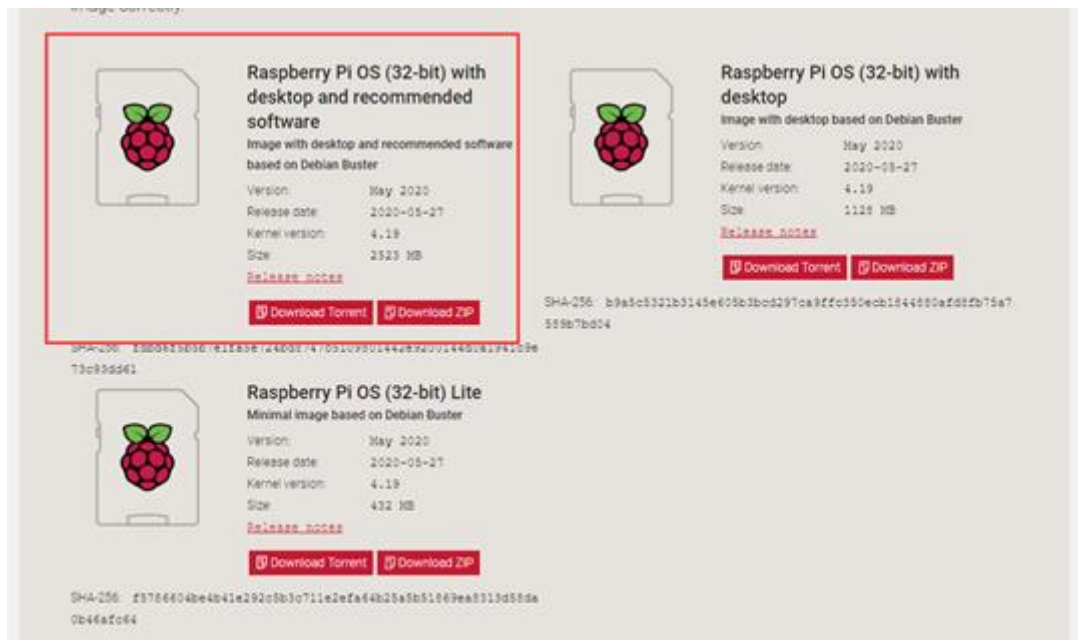
Install Raspberry Pi System

It is recommended to use a SanDisk Class10 SD card with memory storage of 8G and above. The SD card purchased online may already have image burned inside. However, as system is constantly updating and there are so many systems for Raspberry Pi now (you can't play them all), so burning OS by yourself is a necessary skill. Now we will take the Raspbian system as an example.

Download the Latest Disc Image

- Enter Raspberry Pi official web: <http://www.raspberrypi.org/downloads/>
- Find the latest Imager to download.

It is recommended to download Raspberry Pi OS (32-bit) with desktop and recommended software.



Burn to write raspbian image to SD Card on Windows

- Download Win32DiskImager: Baidu Cloud Link (<https://pan.baidu.com/s/1i3oY1Hr>)

It will be installed like this:

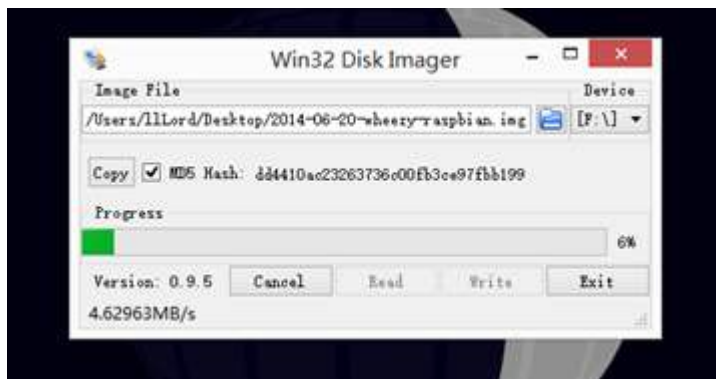


- Insert your SD card and remember the drive letter

- Open the burning software above



- Select the disc images downloaded before and the corresponding drive letter. Be careful not to get it wrong. And then you can go ahead and "WRITE"



- If it can't operate properly, format the SD card and try again
- After doing this, remove the SD card from your computer and insert it into the Raspberry Pi, then it's ready to use now.

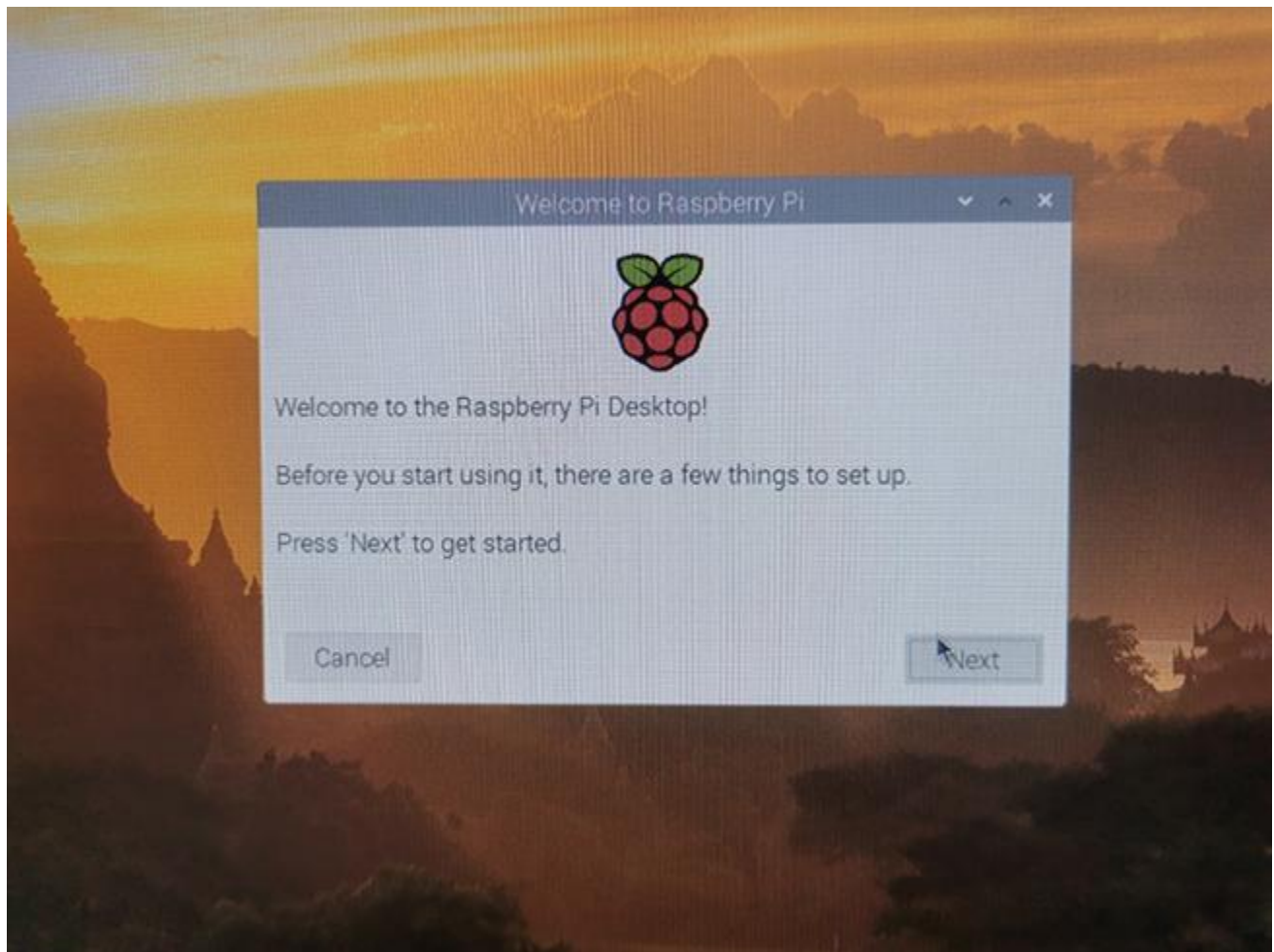


First Time Booting Up

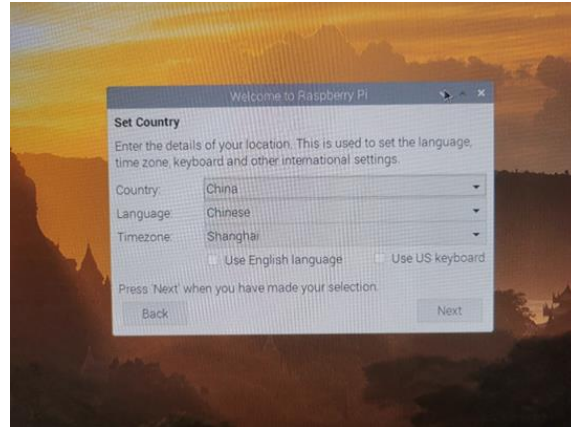
- Connect the Raspberry Pi to the screen as shown in the picture below



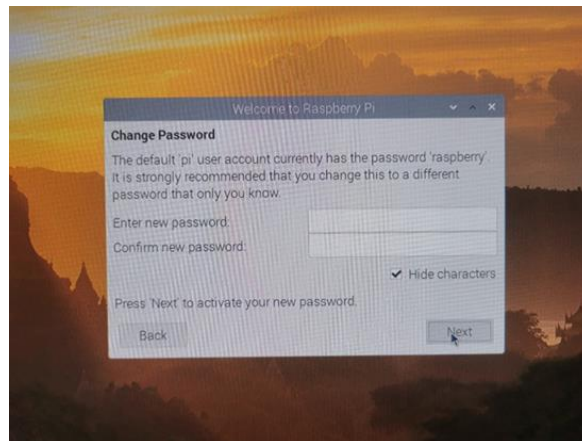
- Insert the Raspberry Pi expansion board and the burned SD card into the Raspberry Pi. Then power it on.
- Press 'Next' to setup your Raspberry Pi



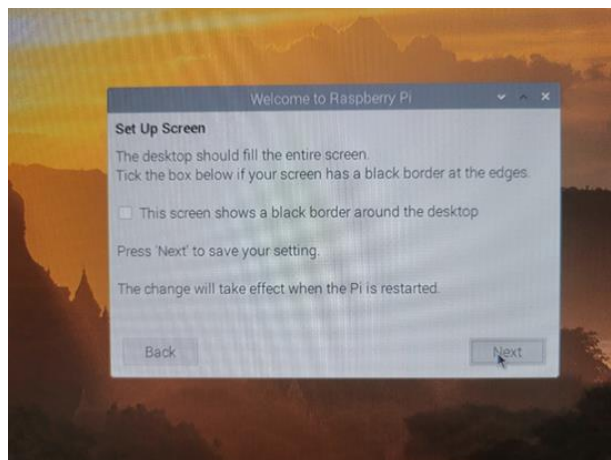
- Select your location and press 'Next'.



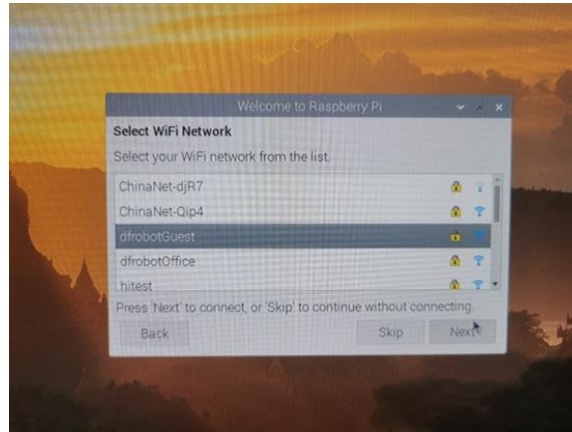
- Set password (If not set, 'raspberry' will be default password). Then press 'Next'.



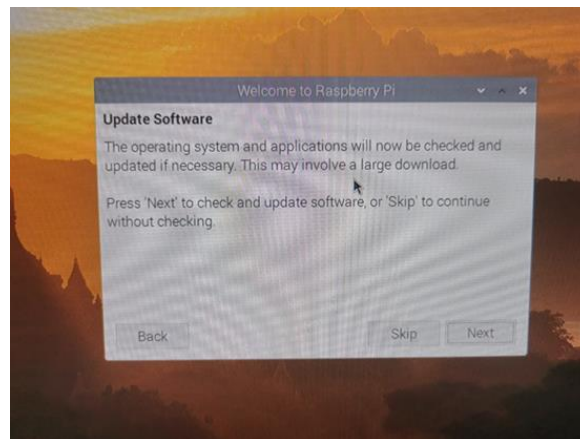
- Select this according to your needs and press 'Next' to save your setting.



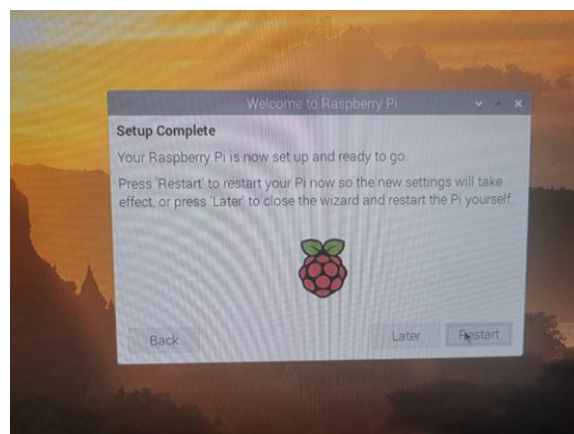
- Select your WIFI network and press 'Next' to enter your password, or press 'Skip' to skip this step.



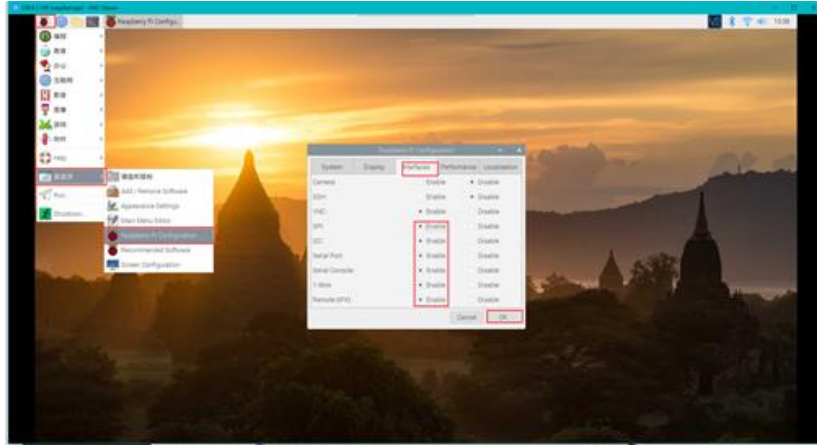
- Press 'Skip' or 'Next' according to your needs.



- Press 'Restart' to restart your Raspberry Pi



- Enable interfaces of Raspberry Pi such as I2C, SPI, GPIO, serial port, etc. Press 'OK' to restart it (Ignore if they are already enabled).



- Install Python dependency libraries and git, and you need to get your Raspberry Pi connect to internet for this step(skip if installed). In the terminal, type the following commands and press 'Enter'

```

pi@raspberrypi ~
文件(F) 编辑(E) 标签(T) 帮助(H)
pi@raspberrypi:~$ sudo apt-get install build-essential python-dev python-smbus git
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
build-essential 已经是最新版 (12.6)。
git 已经是最新版 (1:2.20.1-2+deb10u1)。
python-dev 已经是最新版 (2.7.16-1)。
python-dev 已设置为手动安装。
python-smbus 已经是最新版 (4.1-1)。
升级了 0 个软件包，新安装了 0 个软件包，要卸载 0 个软件包，有 0 个软件包未被升级
pi@raspberrypi:~$

```

`sudo apt-get install build-essential python-dev python-smbus git`

- Update WiringPi Library. In the terminal, type the following instructions in order and press 'Enter'

```

cd /tmp
wget https://project-downloads.drogon.net/wiringpi-latest.deb //Down load wiringpi library
sudo dpkg -i wiringpi-latest.deb //Install wiringpi Library
gpio readall //Read GPIO code number

```

```

pi@raspberrypi:~$ sudo apt-get install build-essential python-dev python-smbus git
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
build-essential 已经是最新版 (12.6)。
git 已经是最新版 (1:2.20.1-2+deb10u1)。
python-dev 已经是最新版 (2.7.16-1)。
python-dev 已设置为手动安装。
python-smbus 已经是最新版 (4.1-1)。
升级了 0 个软件包，新安装了 0 个软件包，要卸载 0 个软件包，有 0 个软件包未被升级。
pi@raspberrypi:~$ cd /tmp
pi@raspberrypi:/tmp$ wget https://project-downloads.drogon.net/wiringpi-latest.deb
--2020-08-21 18:55:19-- https://project-downloads.drogon.net/wiringpi-latest.deb
正在解新主机 project-downloads.drogon.net (project-downloads.drogon.net)... 188.246.205.22, 2a03:9000:10:7b::2
正在连接 project-downloads.drogon.net (project-downloads.drogon.net)[188.246.205.22]:443... 已连接。
已发出 HTTP 请求，正在等待响应... 200 OK
长度：52260 (51K) [application/x-debian-package]
正在保存为：“wiringpi-latest.deb”

wiringpi-latest.deb 100%[=====] 51.04K 34.8KB/s 用时 1.5s

2020-08-21 18:55:24 (34.8 KB/s) - 已保存 “wiringpi-latest.deb” [52260/52260]

pi@raspberrypi:/tmp$ sudo dpkg -i wiringpi-latest.deb
(正在读取数据库 ... 系统当前共安装了有 153919 个文件和目录。)
准备解压 wiringpi-latest.deb ...
正在解压 wiringpi (2.52) 并覆盖 (2.50) ...
正在设置 wiringpi (2.52) ...
正在处理用于 man-db (2.8.5-2) 的触发器 ...
pi@raspberrypi:/tmp$ gpio readall
-----Pi 4B-----
| BCM | wPi | Name | Mode | V | Physical | V | Mode | Name | wPi | BCM | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 8 | 3.3v | | | 1 | 2 | | | 5v | | |
| 3 | 9 | SCL_1 | ALTO | 1 | 3 | 4 | | | 5v | | |
| 4 | 7 | GPIO_7 | IN | 1 | 7 | 8 | 1 | ALTS | Tx0 | 15 | 14 |
| | | 0v | | | 9 | 10 | 1 | ALTS | Rx0 | 16 | 15 |
| 17 | 0 | GPIO_0 | IN | 0 | 11 | 12 | 0 | IN | GPIO_1 | 1 | 18 |
| 27 | 2 | GPIO_2 | IN | 0 | 13 | 14 | | | 0v | | |
| 22 | 3 | GPIO_3 | IN | 0 | 15 | 16 | 0 | IN | GPIO_4 | 4 | 23 |
| | | 3.3v | | | 17 | 18 | 0 | IN | GPIO_5 | 5 | 24 |
| 10 | 12 | MOSI | ALTO | 0 | 19 | 20 | | | 0v | | |
| 9 | 13 | MISO | ALTO | 0 | 21 | 22 | 0 | IN | GPIO_6 | 6 | 25 |
| 11 | 14 | SCLK | ALTO | 0 | 23 | 24 | 1 | OUT | CE0 | 10 | 8 |
| | | 0v | | | 25 | 26 | 1 | OUT | CE1 | 11 | 7 |
| 0 | 30 | SDA_0 | IN | 1 | 27 | 28 | 1 | IN | SCL_0 | 31 | 1 |
| 5 | 21 | GPIO_21 | IN | 1 | 29 | 30 | | | 0v | | |
| 6 | 22 | GPIO_22 | IN | 1 | 31 | 32 | 0 | IN | GPIO_26 | 26 | 12 |
| 13 | 23 | GPIO_23 | IN | 0 | 33 | 34 | | | 0v | | |
| 19 | 24 | GPIO_24 | IN | 0 | 35 | 36 | 0 | IN | GPIO_27 | 27 | 16 |
| 26 | 25 | GPIO_25 | IN | 0 | 37 | 38 | 0 | IN | GPIO_28 | 28 | 20 |
| | | 0v | | | 39 | 40 | 0 | IN | GPIO_29 | 29 | 21 |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| BCM | wPi | Name | Mode | V | Physical | V | Mode | Name | wPi | BCM |
-----Pi 4B-----
pi@raspberrypi:/tmp$

```

- All preparations are completed now.

Lesson 1: Digital LED Light Emitting Module

Learning Goals

Let's start our Raspberry Pi journey from lighting up an LED!

In terms of hardware, you'll learn about Raspberry Pi LED and GPIO output, which is important for later projects. In this process, you'll also come into contact with Python programming and you may find that programming isn't as complicated as you think.

Now let's use Raspberry Pi to control a digital LED light emitting module in this basic project.

Learning Contents - LED Flicker

Guide: In the first project, we will learn about the following contents:

1. Internal circuit resolution of digital LED light emitting module
2. Basic uses of Thonny Python IDE
3. Basic Python code for operating GPIO.

Hardware

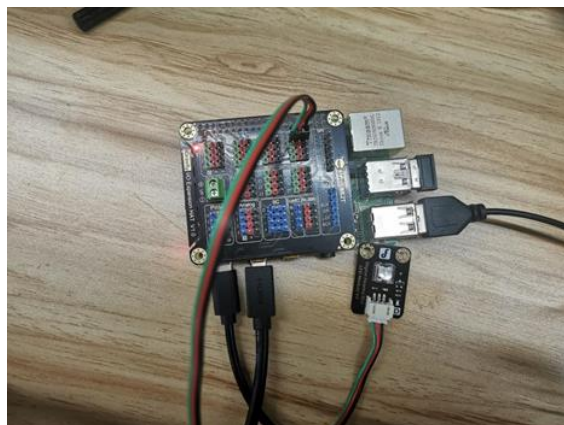
- [Gravity: 37 Pcs Sensor Set](#)
- [Raspberry Pi 4 Model B](#)
- [IO Expansion HAT for Raspberry Pi 4B/3B+](#)
- [8GB + SanDisk Class10 SD/MicroSD Memory Card](#)
- [5V@3A USB Power Supply](#)
- [8.9 IPS Touch Display](#)

Connection

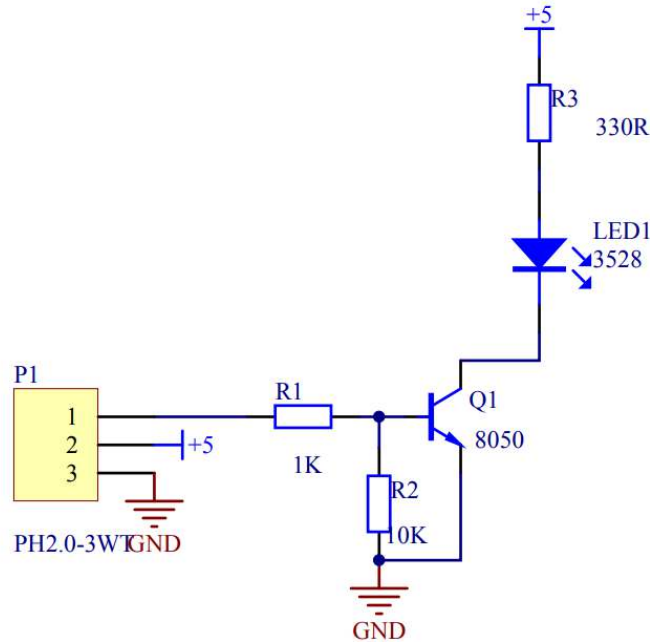
- Connect the necessary peripherals such as the screen, power, keyboard and mouse to your Raspberry Pi.



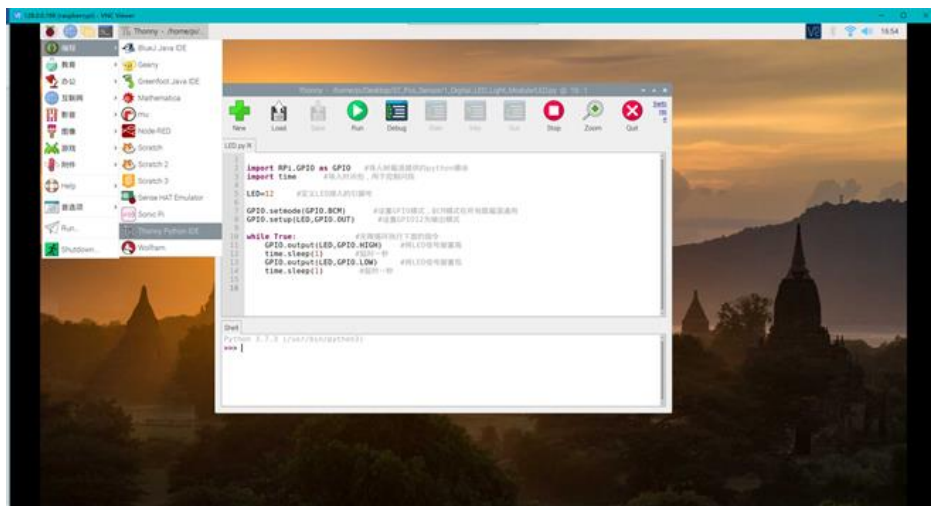
- Install the IO expansion board on the Raspberry Pi. And connect the LED light-emitting module to the digital port 12 of the expansion board. Then start it up.



- By analyzing the LED light-emitting module circuit, we can see that the signal pin is directly connected to the base of an NPN transistor, whose collector and emitter are respectively connected to the negative electrode of the LED and the negative electrode of the power supply, and the positive electrode of the LED is connected to a current-limiting resistor and then connects to VCC. That is to say, when the signal pin appears at a high level, the collector and emitter of the transistor are connected, then the circuit forms a loop, lightening up the LED. When the signal pin is in low level, the LED will be off.



- Open Thonny Python IDE to copy the following program into it

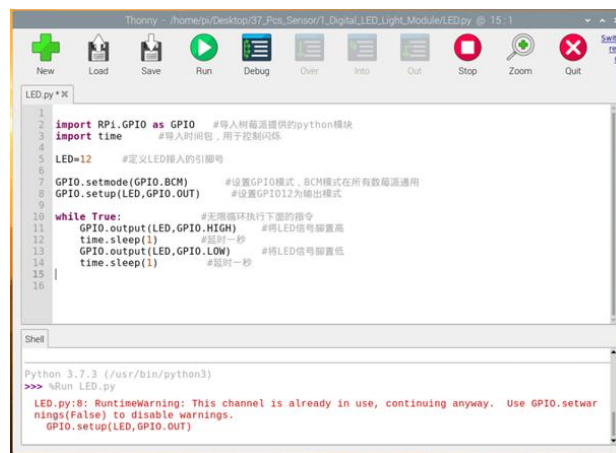


Code

```
import RPi.GPIO as GPIO # Import the python module provided by the Raspberry Pi
import time # Import time package to control flicker

LED=12 # Define the pin number to which the LED is connected

GPIO.setmode(GPIO.BCM) # Set GPIO mode, BCM mode is universally available to
all Raspberry Pi
GPIO.setup(LED,GPIO.OUT) # Set GPIO12 to output mode
while True: # Execute the following commands in an infinite loop
    GPIO.output(LED,GPIO.HIGH) # Set the LED signal pin high (i.e. turn on the
LED)
time.sleep(1) # Delay one second
    GPIO.output(LED,GPIO.LOW) # Set the LED signal pin low (i.e. turn off the
LED)
time.sleep(1) # Delay one second
```



```
LED.py *K
1
2 import RPi.GPIO as GPIO #导入树莓派提供的python模块
3 import time #导入时间包,用于控制闪烁
4
5 LED=12 #定义LED接入的引脚号
6
7 GPIO.setmode(GPIO.BCM) #设置GPIO模式,BCM模式在所有树莓派通用
8 GPIO.setup(LED,GPIO.OUT) #设置GPIO12为输出模式
9
10 while True: #无限循环执行下面的指令
11     GPIO.output(LED,GPIO.HIGH) #将LED信号脚置高
12     time.sleep(1) #延时一秒
13     GPIO.output(LED,GPIO.LOW) #将LED信号脚置低
14     time.sleep(1) #延时一秒
15
16

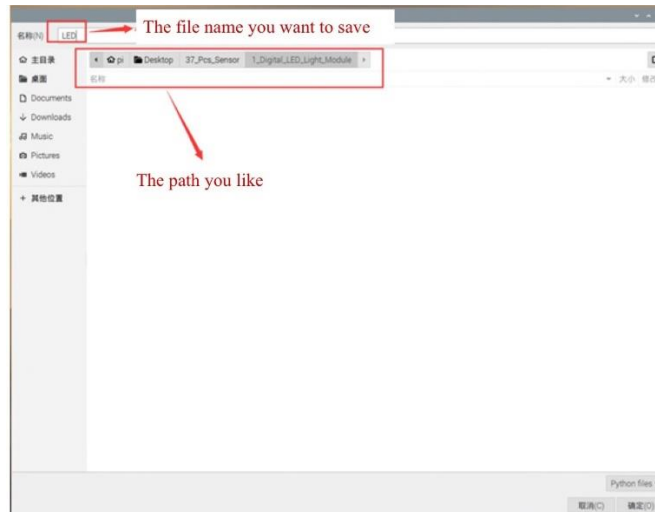
Shell

Python 3.7.3 (/usr/bin/python3)
>>> %Run LED.py
LED.py:8: RuntimeWarning: This channel is already in use, continuing anyway. Use GPIO.setwarnings(False) to disable warnings.
GPIO.setup(LED,GPIO.OUT)
```

- Click 'Save' to set the file name and the path you like to save



```
<untitled> *K
1 import RPi.GPIO as GPIO #导入树莓派提供的python模块
2 import time #导入时间包,用于控制闪烁
3
4 LED=12 #定义LED接入的引脚号
5
6 GPIO.setmode(GPIO.BCM) #设置GPIO模式,BCM模式在所有树莓派通用
7 GPIO.setup(LED,GPIO.OUT) #设置GPIO12为输出模式
8
9 while True: #无限循环执行下面的指令
10     GPIO.output(LED,GPIO.HIGH) #将LED信号脚置高
11     time.sleep(1) #延时一秒
12     GPIO.output(LED,GPIO.LOW) #将LED信号脚置低
13     time.sleep(1) #延时一秒
14
```

- Click 'Run', then you can see the LED is flickering



Lesson 2: Digital Push Button Module

Learning Goals

In the last tutorial, we have learned how to simply control the on and off of the LED module. In this tutorial, we will learn about Raspberry Pi buttons and GPIO inputs based on the previous tutorial, which is very important for future projects. In this process, you will be exposed to new Python programs.

Learning Contents – Control LED by the Button

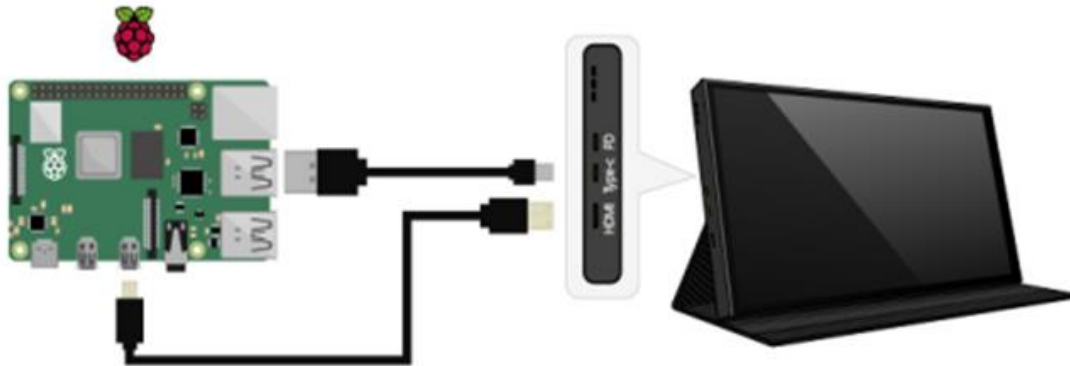
Guide: In this project, we will learn the basic principles of the button module. And we will also consolidate the basic uses of Thonny Python IDE we learned before, as well as the basic Python code for operating GPIO.

Hardware

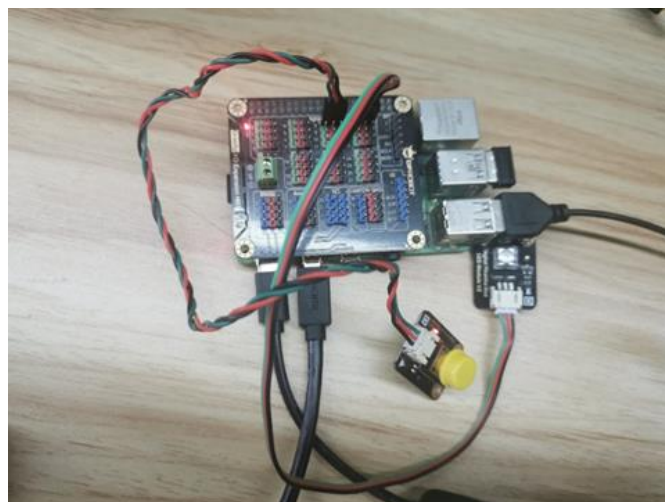
- [Gravity: 37 Pcs Sensor Set](#)
- [Raspberry Pi 4 Model B](#)
- [IO Expansion HAT for Raspberry Pi 4B/3B+](#)
- [8GB + SanDisk Class10 SD/MicroSD Memory Card](#)
- [5V@3A USB Power Supply](#)
- [8.9 IPS Touch Display](#)

Connection

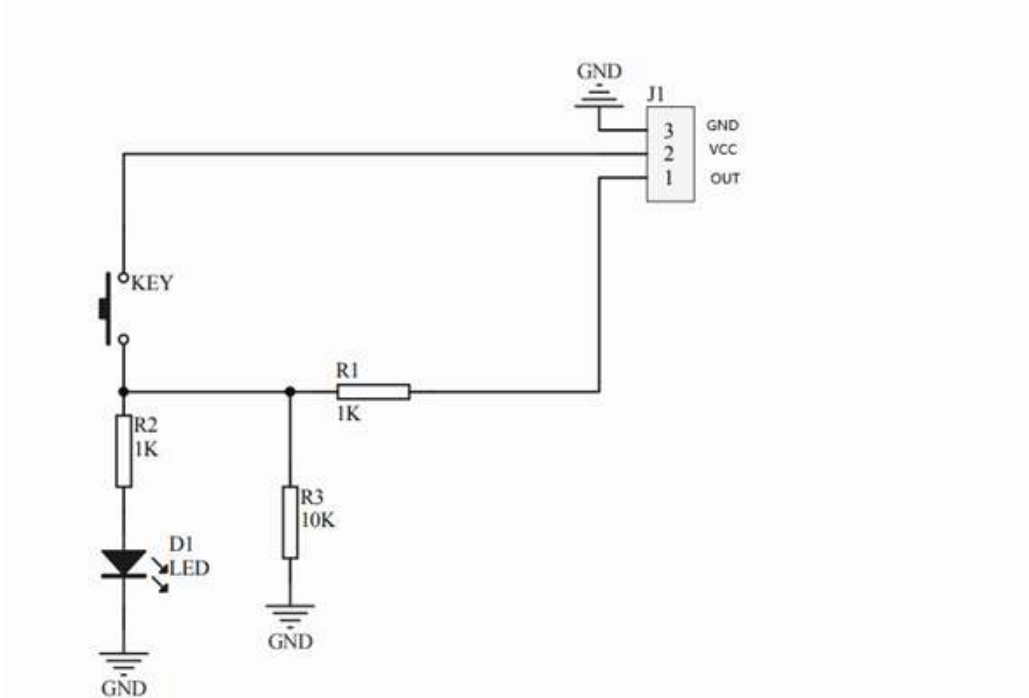
- Connect the necessary peripherals such as the screen, power, keyboard and mouse to your Raspberry Pi.



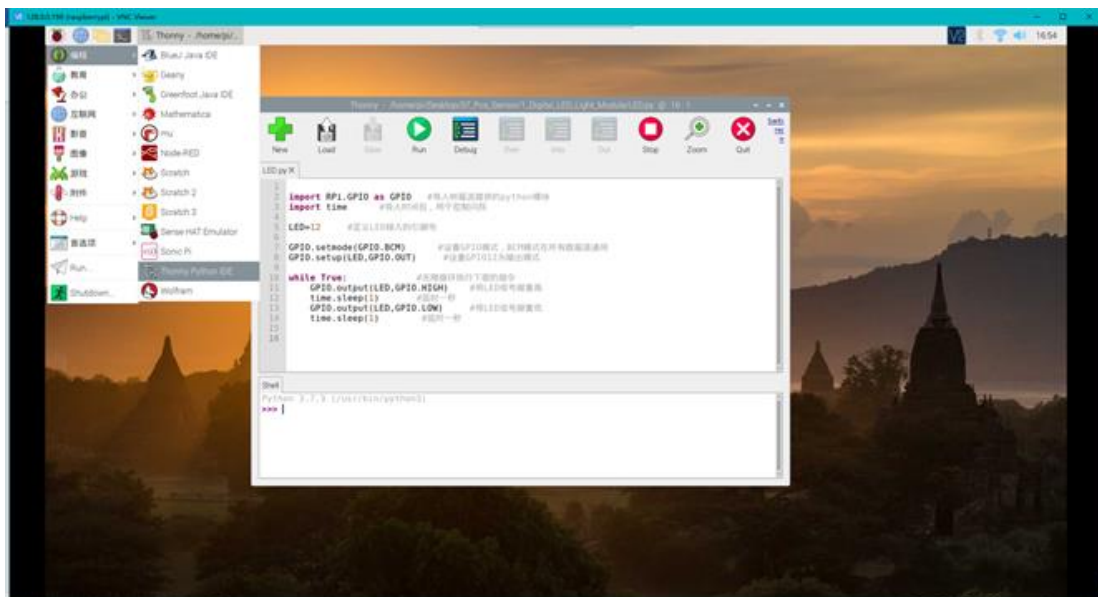
- Install the Raspberry Pi IO expansion board on your 'Pi' and connect the LED light-emitting module to digital port 12 on the expansion board, and the digital push button module to the digital port 8. Then power on.



- By analyzing the schematic circuit diagram of this module, we can see that when release the button, the signal output pin is at low level and the indicator light is off. When the button is pressed, the signal pin is at high level and the indicator light is on. They are corresponding to the two states of the LED light-emitting module. In this case, the button can be used to control the LED light through a simple judgment by the Raspberry Pi.



- Open Thonny Python IDE to copy the following program into it



Code

```
import RPi.GPIO as GPIO    # Import the python module provided by the Raspberry Pi
import time                # Import time package to control flicker

LED=12                    # Define the pin number to which the LED is connected
Blue=8                    # Define the pin number to which the button is connected

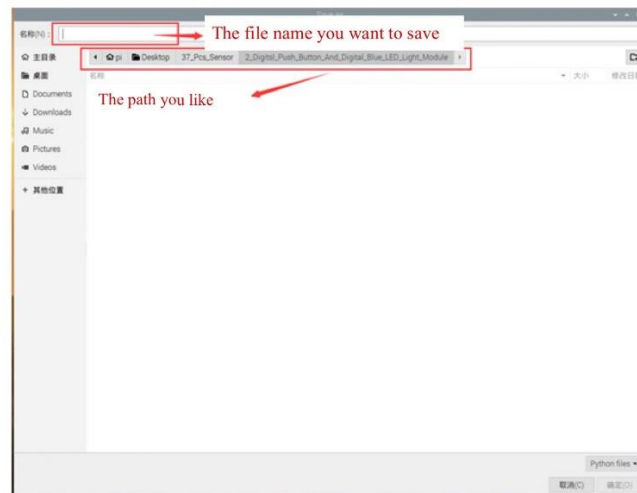
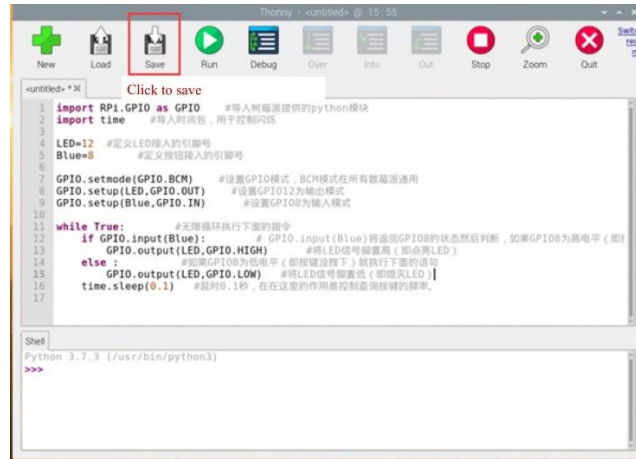
GPIO.setmode(GPIO.BCM)    # Set GPIO mode, BCM mode is universally available to all
Raspberry Pi
GPIO.setup(LED,GPIO.OUT)  # Set GPIO12 to output mode
GPIO.setup(Blue,GPIO.IN)  #Set GPIO8 to input mode

while True:              # Execute the following commands in an infinite loop
    if GPIO.input(Blue): # GPIO.input(Blue) will return to the state of GPIO8 and
judge it. If GPIO8 is high (that is, the button is pressed), execute the following
statement
        GPIO.output(LED,GPIO.HIGH)    # Set the LED signal pin high (i.e. light up
the LED)
    else :                  # If GPIO8 is low (that is, the button is released), execute the
following statement
        GPIO.output(LED,GPIO.LOW)    # Set the LED signal pin low (i.e. turn off the
LED)
time.sleep(0.1)           # Delay one second, here is to control the frequency for querying
key status
```



```
Thonny - <untitled> @ 16:49
New Load Save Run Debug Over Info Out Stop Zoom Quit
<untitled> *X
1 import RPi.GPIO as GPIO    #导入树莓派提供的python模块
2 import time                #导入时间包,用于控制闪烁
3
4 LED=12                    #定义LED接入的引脚号
5 Blue=8                    #定义按钮接入的引脚号
6
7 GPIO.setmode(GPIO.BCM)    #设置GPIO模式,BCM模式在所有树莓派通用
8 GPIO.setup(LED,GPIO.OUT)  #设置GPIO12为输出模式
9 GPIO.setup(Blue,GPIO.IN)  #设置GPIO8为输入模式
10
11 while True:              #无限循环执行下面的指令
12     if GPIO.input(Blue): # GPIO.input(Blue)将返回GPIO8的状态然后判断,如果GPIO8为高电平(即
13         GPIO.output(LED,GPIO.HIGH)    #将LED信号脚置高(即点亮LED)
14     else :                #如果GPIO8为低电平(即按键没按下)就执行下面的语句
15         GPIO.output(LED,GPIO.LOW)    #将LED信号脚置低(即熄灭LED)
16     time.sleep(0.1)       #延时0.1秒,在这里的作用是控制查询按键的频率。
17
Shell
Python 3.7.3 (/usr/bin/python3)
>>>
```

- Click 'Save' to set the file name and the save path you like



- Click 'Run', you can see that the light is on and off with the button pressed and released.



Lesson 3: LM35 Analog Linear Temperature Sensor

Learning Contents

Introduction to LM35 Analog Linear Temperature Sensor

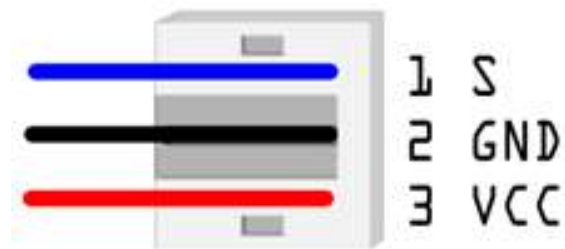
Based on LM35 semiconductor, this sensor produced by National Semiconductor Corporation can be used to detect ambient temperature. It offers a measurement range from $-40\text{ }^{\circ}\text{C}$ to $150\text{ }^{\circ}\text{C}$ and a sensitivity of $10\text{ mV}/^{\circ}\text{C}$. And its output voltage is proportional to the temperature. Moreover, if used in combination with sensor-specific expansion of Arduino board, this sensor can be really easy to achieve interactive effects related to ambient temperature perception.

Commonly-used sensors for temperature measurement include thermocouples, platinum resistance, thermal resistance and temperature semiconductor chips. Thermocouples are commonly used in high temperature measurement. Platinum resistance temperature modules are used in measurement of 800 degrees Celsius, while the thermal resistance and semiconductor temperature sensor are suitable for measuring the temperature of 100-200 degrees or below. With good linearity and high sensitivity, the semiconductor temperature sensor is easy to use.

Precautions

The port layout of the new analog sensor has the following two improvements. Please refer to the blog [How to change the layout of the data cable connector](#). When using this sensor on the IO expansion board, you may need to adjust the layout of the connector. For your convenience, we will make more improvements, so stay tuned.

old version

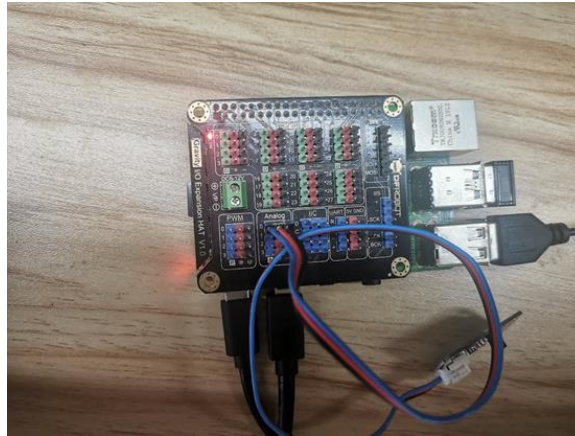


new version



Use LM35 Analog Linear Temperature Sensor on Your Raspberry Pi

- Connect the sensor to the analog pin 0 on the expansion board



- Open Thonny Python IDE to copy the following program into it

```
Thonny - /home/pi/Desktop/17 Pin Sensor/3 Analog LM35 Temperature Sensor/Analog LM35 Temperature Sensor.py @ 7:20
+ New Load Save Run Debug Over Help Out Stop Zoom Quit Switch to remote mode
Analog LM35 Temperature Sensor.py R demo_adc.py R
1 import time
2 from DFRobot_RaspberryPi_Expansion_Board import DFRobot_Expansion_Board_IIC as Board
3
4 board = Board(1, 0x10) # Select i2c bus 1, set address to 0x10
5
6
7 def board_detect():
8     l = board.detect()
9     print("Board list conform:")
10    print(l)
11
12    """ print last operate status, users can use this variable to determine the result of a function """
13    def print_board_status():
14        if board.last_operate_status == board.STA_OK:
15            print("board status: everything ok")
16        elif board.last_operate_status == board.STA_ERR:
17            print("board status: unexpected error")
18        elif board.last_operate_status == board.STA_ERR_DEVICE_NOT_DETECTED:
19            print("board status: device not detected")
20        elif board.last_operate_status == board.STA_ERR_PARAMETER:
21            print("board status: parameter error")
22        elif board.last_operate_status == board.STA_ERR_CMD_FUNC_ERR:
23            print("board status: command function error")
24
25    board_detect()
26    print_board_status()
27
28    while True:
29        board_detect()
30        print_board_status()
31        time.sleep(1)
32
33    Shell
34
35    Temperature = 25 ℃
36    Temperature = 25 ℃
37    Temperature = 25 ℃
```

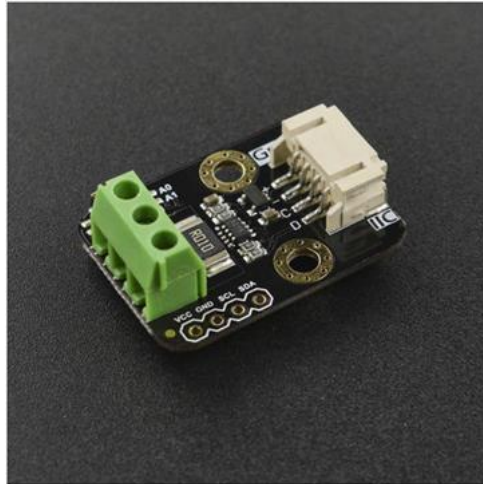
Lesson 4: I2C Digital Wattmeter

Introduction

I2C Digital Wattmeter is a high-resolution, high-precision, large-scale measurement module that can measure the voltage, current and power of various electronic modules and electrical equipment within 26V 8A, and the maximum relative error is no more than $\pm 0.2\%$ (A simple manual calibration is required before using). It can be used for power consumption measurement or battery life evaluation of solar energy systems, battery coulombmeters, motors, controllers or electronic modules.

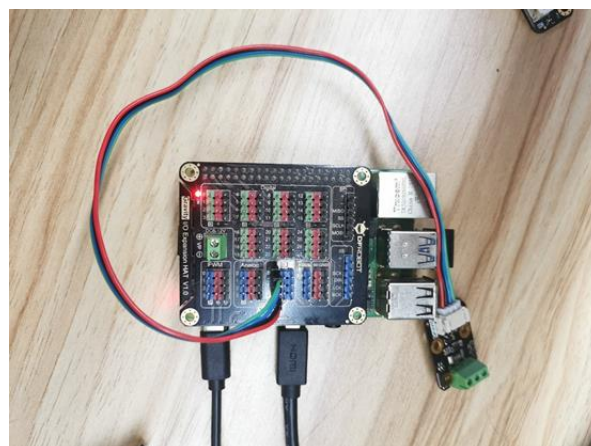
This module adopts TI INA219 zero temperature drift current/power monitoring chip and 2W high power low temperature drift 10m Ω alloy sampling resistor. The voltage and current resolution can reach 4mV and 1mA respectively. Under the full-scale measurement condition,

the maximum relative error of voltage and current measurement can be superior to $\pm 0.2\%$. It also provides four I2C addresses that can be configured via the 2P DIP switch. The module accurately measures bi-directional high-side currents (current flowing through the power supply or battery positive), which is especially useful in solar or battery fuel gauge applications where the battery needs to be charged and discharged. This status can be simply determined by positive or negative current readings. In the motor applications, the current can be monitored in real time by monitoring whether the motor current is too large due to overload. In addition, you can use this module to measure the power consumption of various electronic modules or the entire project to evaluate battery life.

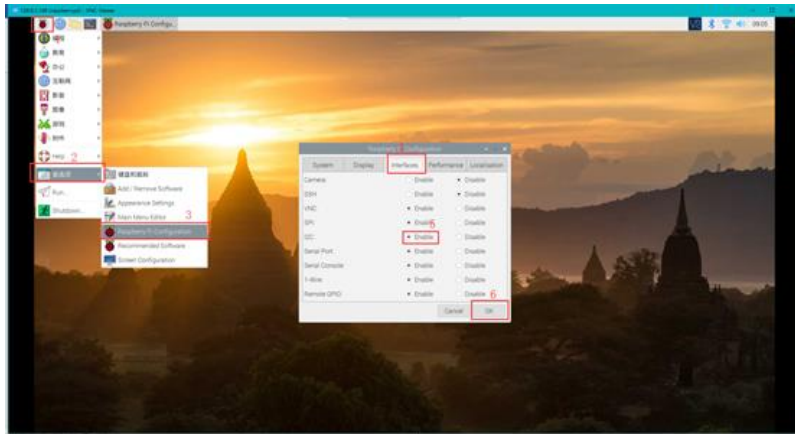


Use I2C Digital Wattmeter on Your Raspberry Pi

- Power the Raspberry Pi on and install the Raspberry Pi expansion board correctly
- Connect the sensor to the IIC interface on the expansion board



- Configure to enable I2C and restart the Raspberry Pi. If configured, you can skip this step. Configure the Raspberry Pi according to the following procedure and restart it.



- Install I2C libraries and tools, and you need to get your Raspberry Pi connect to internet for this step(skip if installed). In the terminal, type the following instructions and press 'Enter'

```
sudo apt-get install i2c-tools
```

- When the I2C device is connected, the I2C address can be checked by the following command. In the terminal, type the following instructions and press 'Enter'

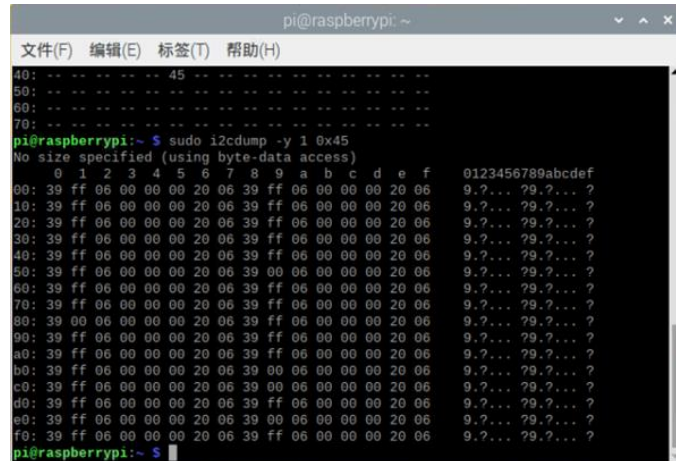
```
sudo i2cdetect -y -a 1
```

```
pi@raspberrypi: ~  
文件(F) 编辑(E) 标签(T) 帮助(H)  
d0: 39 ff 06 00 00 00 20 06 39 00 06 00 00 00 20 06 9.?.?.?.?.?.?.?  
e0: 39 ff 06 00 00 00 20 06 39 ff 06 00 00 00 20 06 9.?.?.?.?.?.?.?  
f0: 39 ff 06 00 00 00 20 06 39 ff 06 00 00 00 20 06 9.?.?.?.?.?.?.?  
pi@raspberrypi:~$ sudo i2cdetect -y -a 1  
 0 1 2 3 4 5 6 7 8 9 a b c d e f  
00: 00 -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
10: 10 -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
40: -- -- -- -- -- 45 -- -- -- -- -- -- -- -- -- --  
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
70: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
pi@raspberrypi:~$ sudo i2cdetect -y -a 1  
 0 1 2 3 4 5 6 7 8 9 a b c d e f  
00: 00 -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
10: 10 -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
40: -- -- -- -- -- 45 -- -- -- -- -- -- -- -- -- --  
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
70: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  
pi@raspberrypi:~$
```

- Read all register data of I2C device. In the terminal, type the following instructions and press 'Enter'

```
sudo i2cdump -y 1 0x45
```

-y means cancelling the user interaction process and directly executing the command 1 is the I2C device number 0x45 is I2C device address



```
pi@raspberrypi: ~
文件(F) 编辑(E) 标签(T) 帮助(H)
40: -- -- -- -- 45 -- -- -- -- --
50: -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- -- --
pi@raspberrypi: ~ $ sudo i2cdump -y 1 0x45
No size specified (using byte-data access)
 0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f  0123456789abcdef
00: 39 ff 06 00 00 00 20 06 39 ff 06 00 00 00 20 06  9.?...?9.?...?
10: 39 ff 06 00 00 00 20 06 39 ff 06 00 00 00 20 06  9.?...?9.?...?
20: 39 ff 06 00 00 00 20 06 39 ff 06 00 00 00 20 06  9.?...?9.?...?
30: 39 ff 06 00 00 00 20 06 39 ff 06 00 00 00 20 06  9.?...?9.?...?
40: 39 ff 06 00 00 00 20 06 39 ff 06 00 00 00 20 06  9.?...?9.?...?
50: 39 ff 06 00 00 00 20 06 39 00 06 00 00 00 20 06  9.?...?9.?...?
60: 39 ff 06 00 00 00 20 06 39 ff 06 00 00 00 20 06  9.?...?9.?...?
70: 39 ff 06 00 00 00 20 06 39 ff 06 00 00 00 20 06  9.?...?9.?...?
80: 39 00 06 00 00 00 20 06 39 ff 06 00 00 00 20 06  9.?...?9.?...?
90: 39 ff 06 00 00 00 20 06 39 ff 06 00 00 00 20 06  9.?...?9.?...?
a0: 39 ff 06 00 00 00 20 06 39 ff 06 00 00 00 20 06  9.?...?9.?...?
b0: 39 ff 06 00 00 00 20 06 39 00 06 00 00 00 20 06  9.?...?9.?...?
c0: 39 ff 06 00 00 00 20 06 39 00 06 00 00 00 20 06  9.?...?9.?...?
d0: 39 ff 06 00 00 00 20 06 39 ff 06 00 00 00 20 06  9.?...?9.?...?
e0: 39 ff 06 00 00 00 20 06 39 00 06 00 00 00 20 06  9.?...?9.?...?
f0: 39 ff 06 00 00 00 20 06 39 ff 06 00 00 00 20 06  9.?...?9.?...?
pi@raspberrypi: ~ $
```

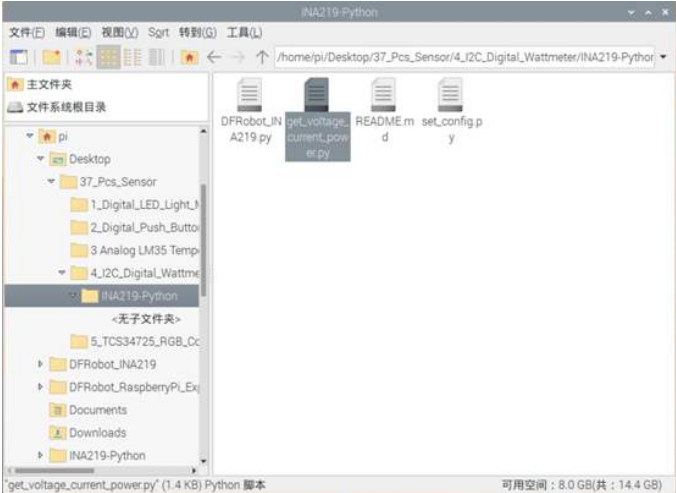
- Install Python dependency libraries and git, and you need to get your Raspberry Pi connect to internet for this step(skip if installed). In the terminal, type the following instructions and press 'Enter'

```
sudo apt-get install build-essential python-dev python-smbus git
```

- Install the drive library. In the terminal, type the following instructions in order and press 'Enter'

```
cd \~
git clone https://github.com/DFRobotdl/37_Pcs_Sensor.git
```

- Find get_voltage_current_power.py in 37_Pcs_Sensor\4_I2C_Digital_Wattmeter, open it with Thonny Pyth IDE and run it, then you can see the printed data.



The screenshot shows a Python IDE window titled 'Thonny - /home/pi/Desktop/37_Pcs_Sensor4_I2C_Digital_Wattmeter/INA219_Python/get_voltage_current_power.py @ 52:1'. The code in the editor is as follows:

```
1 # -*- coding: utf-8 -*-
2
3 """
4 file get_voltage_current_power.py
5 SEN0291 Wattmeter Sensor
6 This sensor can detect Voltage ,Current,and Power
7 The module has four I2C, these addresses are:
8
9 ADDRESS 1 0x40 A0 = 0 A1 = 0
10 ADDRESS 2 0x41 A0 = 1 A1 = 0
11 ADDRESS 3 0x44 A0 = 0 A1 = 1
12 ADDRESS 4 0x45 A0 = 1 A1 = 1
13
14 Copyright [DFRobot](http://www.dfrobot.com), 2016
15 Copyright GNU Lesser General Public License
16 version V1.0
17 date 2019-2-27
18 """
19
20 import time
21 from DFRobot_INA219 import INA219
22
```

The Shell window below the code displays the following output:

```
Power : 0 mW
Shunt Voltage : -3.00 mV
Bus Voltage : 0.829 V
Current : -3 mA
Power : 0 mW
```

- Click 'run', it will show current, voltage, and other information.

Lesson 5: TCS34725 I2C Color Sensor

Learning Contents

Lead-in

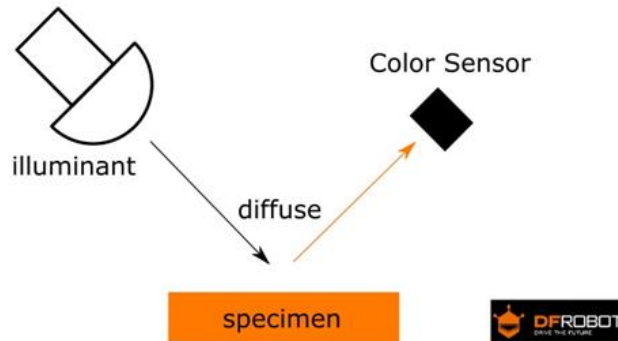
Want to know the secret to chameleons' ability to change color? Want know how the most popular electronic color picking pen picks color? Don't think too complicated, because their principle is really simple. Nature, is the best teacher of mankind. There are so many interesting ideas inspired by creatures.

Through tens of millions of years of derivation, the chameleon has formed a biological instinct, which can be perfectly hidden in the surroundings by changing the protective color of its skin. This is a process from "color picking" to "color matching".

Today, electronic color picking pens use the same principle. First, obtain the RGB three primary color values by detecting the color of the object. Then blend these values to get the color of the object. It's just like kneading plasticine when you were a kid. After knowing a certain ratio, you can knead the color you want.

Introduction to I2C Color Sensor

TCS34725 is a low-cost and cost-effective RGB full-color sensor. The sensor recognizes the surface color of an object through optical sensing. It supports the three primary colors of red, green, and blue (RGB), supports bright light sensing, and can output the corresponding specific values to help restore the true colors.



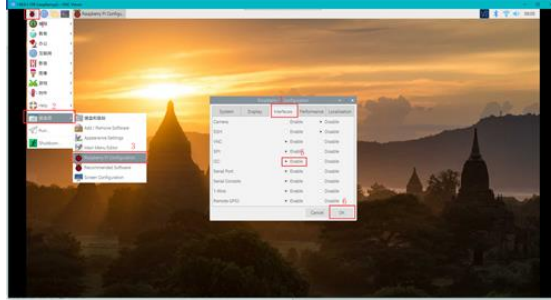
For a higher precision, it specifically employs a IR blocking filter at bottom to avoid interference from the surroundings, minimizing the infrared spectrum component of the incident light. In this case, it gets a more accurate color management. Besides, this sensor also includes four ultra-bright LEDs to allow itself to work without external light resources. And the module works via I2C bus, featuring PH2.0-4P and XH2.54 (breadboard) interfaces to meet different using requirements.

Use I2C Color Sensor on Your Raspberry Pi

- Power the Raspberry Pi on and install the Raspberry Pi expansion board correctly
- Connect the sensor to corresponding IIC interface on the expansion board



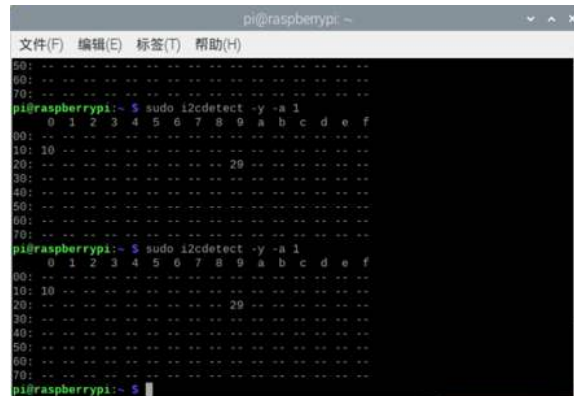
- Configure to enable I2C and restart the Raspberry Pi. If configured, you can skip this step. Configure the Raspberry Pi according to the following procedure and restart it.



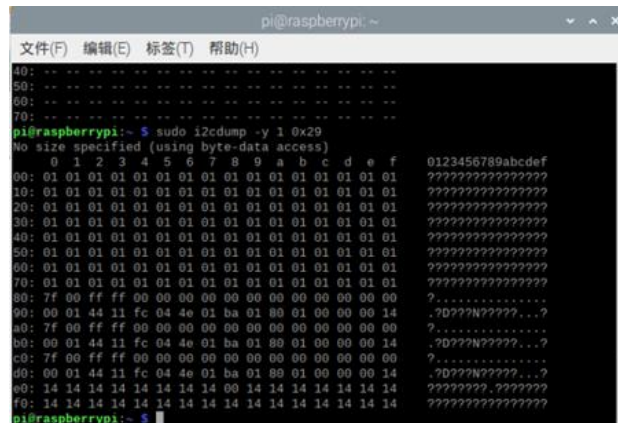
- Install I2C libraries and tools, and you need to get your Raspberry Pi connect to internet for this step(skip if installed). In the terminal, type the following instructions and press 'Enter'

`sudo apt-get install i2c-tools`

- When the I2C device is connected, the I2C address can be checked by the following command. In the terminal, type the following instructions and press 'Enter' `sudo i2cdetect -y -a 1`



- Read all register data of I2C device. In the terminal, type the following instructions and press 'Enter' `sudo i2cdump -y 1 0x29`
- y means cancelling the user interaction process and directly executing the command 1 is the I2C device number 0x29 is I2C device address



- Open Thonny Python IDE to copy the following program into it

```
import smbus
import time
bus = smbus.SMBus(1)

addr = 0x29

CDATAL = 0x94
CDATAH = 0x95
RDATAL = 0x96
RDATAH = 0x97
GDATAL = 0x98
GDATAH = 0x99
BDATAL = 0x9a
BDATAH = 0x9b

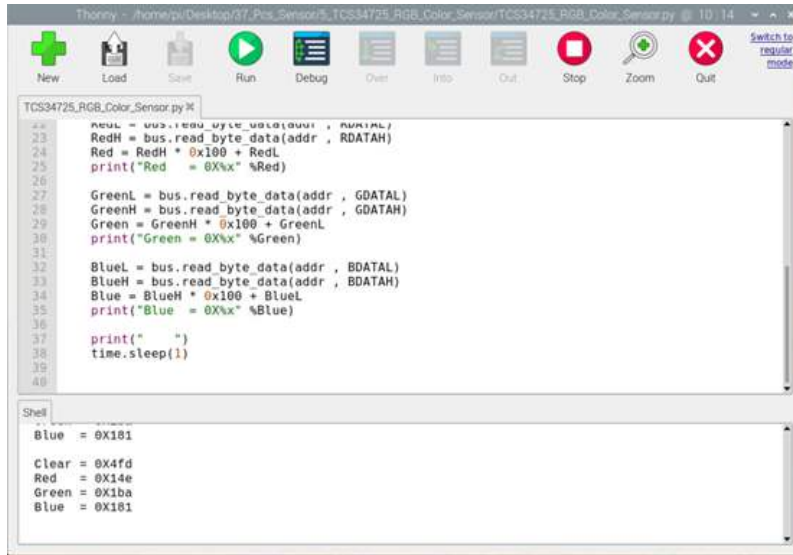
while True:
    ClearL = bus.read_byte_data(addr , CDATAL)
    ClearH = bus.read_byte_data(addr , CDATAH)
    Clear = ClearH * 0x100 + ClearL
    print("Clear = 0X%x" %Clear)

    RedL = bus.read_byte_data(addr , RDATAL)
    RedH = bus.read_byte_data(addr , RDATAH)
    Red = RedH * 0x100 + RedL
    print("Red   = 0X%x" %Red)

    GreenL = bus.read_byte_data(addr , GDATAL)
    GreenH = bus.read_byte_data(addr , GDATAH)
    Green = GreenH * 0x100 + GreenL
    print("Green = 0X%x" %Green)

    BlueL = bus.read_byte_data(addr , BDATAL)
    BlueH = bus.read_byte_data(addr , BDATAH)
    Blue = BlueH * 0x100 + BlueL
    print("Blue  = 0X%x" %Blue)

    print("    ")
    time.sleep(1)
```



```
TCS34725_RGB_Color_Sensor.py
23 RedH = bus.read_byte_data(addr, RDATAH)
24 Red = RedH * 0x100 + RedL
25 print("Red = 0x%x" %Red)
26
27 GreenL = bus.read_byte_data(addr, GDATAH)
28 GreenH = bus.read_byte_data(addr, GDATAH)
29 Green = GreenH * 0x100 + GreenL
30 print("Green = 0x%x" %Green)
31
32 BlueL = bus.read_byte_data(addr, BDATAH)
33 BlueH = bus.read_byte_data(addr, BDATAH)
34 Blue = BlueH * 0x100 + BlueL
35 print("Blue = 0x%x" %Blue)
36
37 print(" ")
38 time.sleep(1)
39
40
Shell
Blue = 0x181
Clear = 0x4fd
Red = 0x14e
Green = 0x1ba
Blue = 0x181
```

- Click 'Run', it will print the color information.

Lesson 6: Analog Rotation Sensor

Learning Contents

Introduction to Analog Rotation Sensor V2

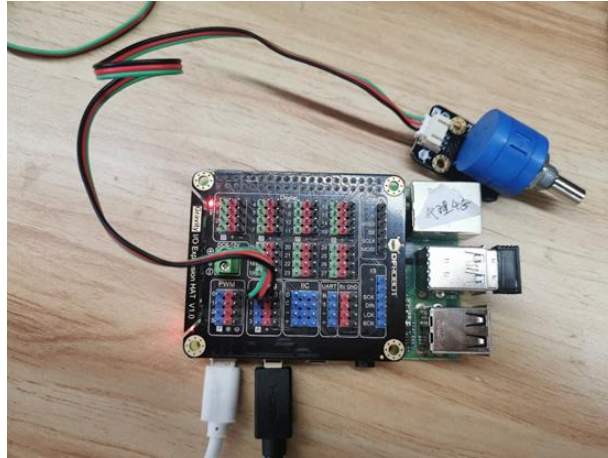
As the rotation angle of the ordinary potentiometer is only 300 degrees at most, the accuracy is quite low after distributing the 5V power supply of Arduino to every 1 degree.

So if you want to make a project with precise control of angle or analog quantity, this precision angle sensor is a good choice. Based on a multi-turn precision potentiometer, this sensor can be rotated about 10 turns and subdivide the voltage into 1024 parts. What's more, it can be combined with the sensor expansion board through the 3P connection line, accurately sensing small changes in rotation.

*This product has been updated. Please refer to the product wiki at the bottom of the page for the specific line sequence of the V2 version. For V1 version, please connect according to the "1, 2, 3" marked on the module's silk screen.

Use Analog Rotation Sensor on Your Raspberry Pi

- Power the Raspberry Pi on and install the Raspberry Pi expansion board correctly
- Connect the sensor to the analog port 0 on the expansion board

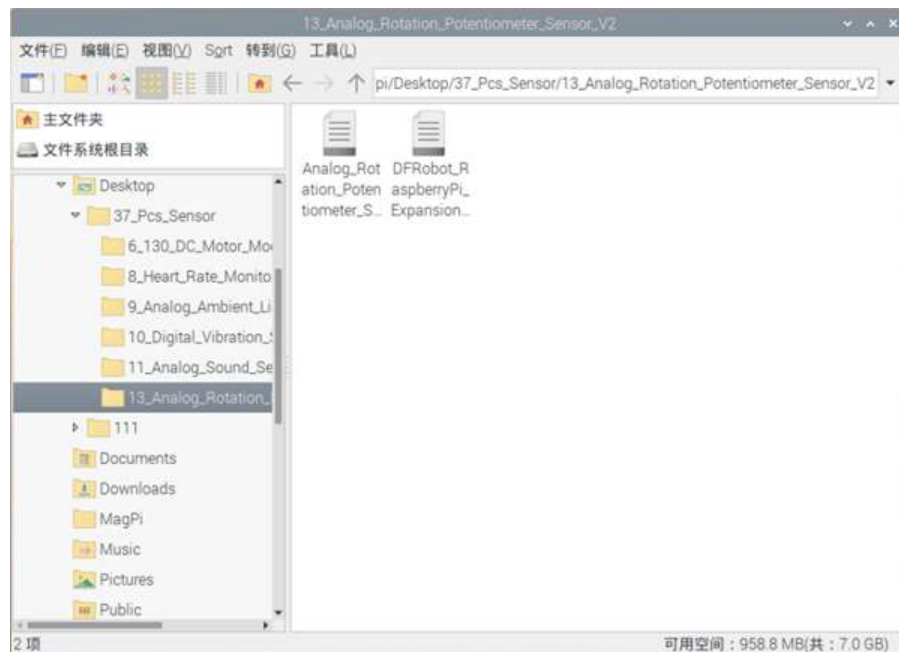


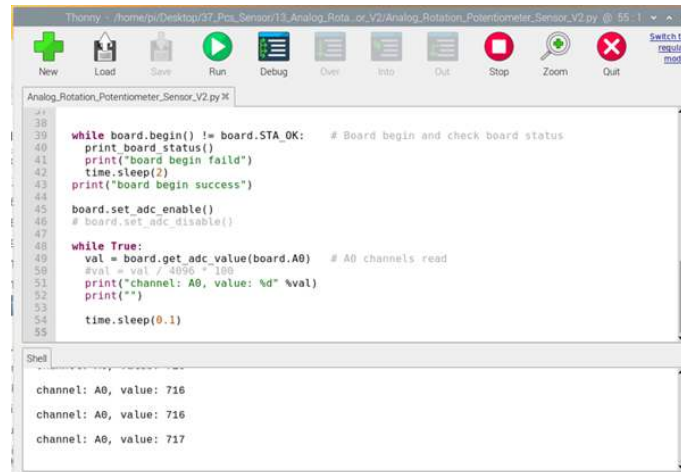
- Install Python dependency libraries and git, and you need to get your Raspberry Pi connect to internet for this step(skip if installed). In the terminal, type the following instructions and press 'Enter'

```
sudo apt-get install build-essential python-dev python-smbus git
```

- Install the drive library and program. In the terminal, type the following instructions and press 'Enter'

```
cd ~  
git clone https://github.com/DFRobotdl/37_Pcs_Sensor.git
```
- Find `Analog_Rotation_Potentiometer_Sensor_V2.py` in `37_Pcs_Sensor\6_Analog_Rotation_Potentiometer_Sensor`, open it with Thonny Pyth IDE and run it, you can see the printed analog value, which will change if you rotate the sensor.





```
38
39 while board.begin() != board.STA_OK: # Board begin and check board status
40     print("board status")
41     print("board begin failed")
42     time.sleep(2)
43     print("board begin success")
44
45 board.set_adc_enable()
46 # board.set_adc_disable()
47
48 while True:
49     val = board.get_adc_value(board.A0) # A0 channel's read
50     #val = val / 4096 * 100
51     print("channel: A0, value: %d" %val)
52     print("")
53
54     time.sleep(0.1)
55
```

Shell

```
channel: A0, value: 716
channel: A0, value: 716
channel: A0, value: 717
```

Lesson 7: 130 DC Motor Fan

Learning Contents

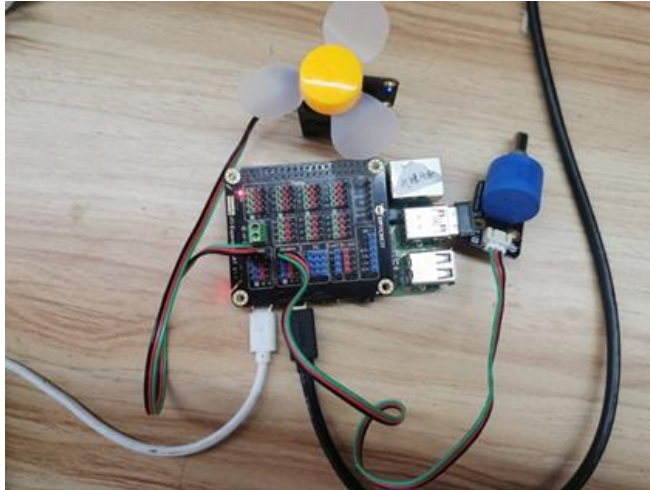
Introduction

This module is really interesting. It can be easily driven by Arduino without an additional motor driver board. You can also use the PWM pulse width to adjust its speed, which is suitable for light applications or small DIY. Simple, but useful.

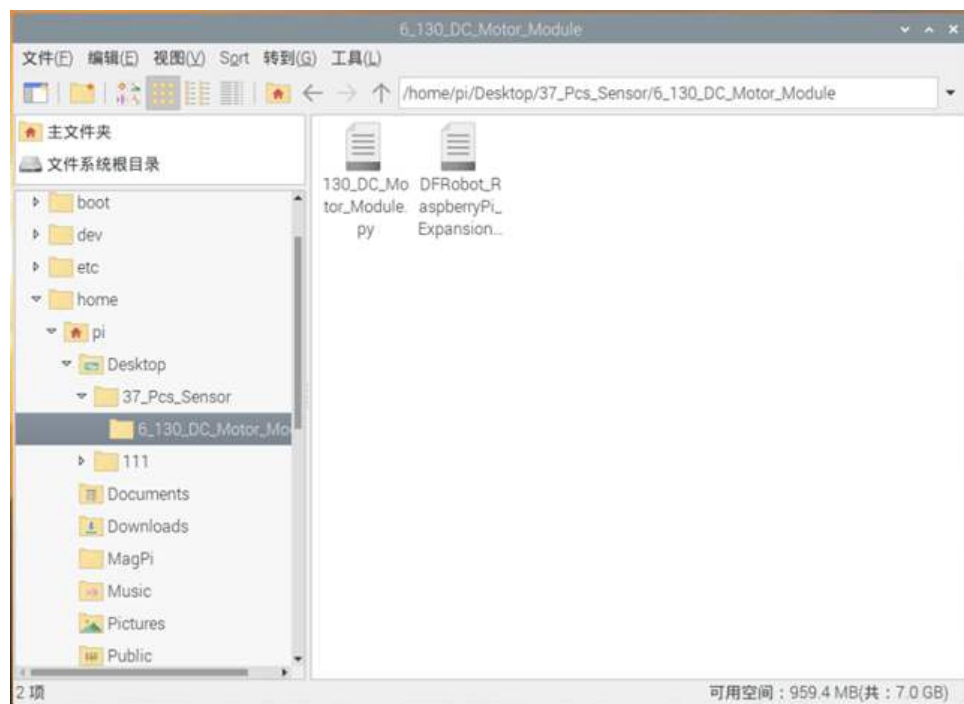


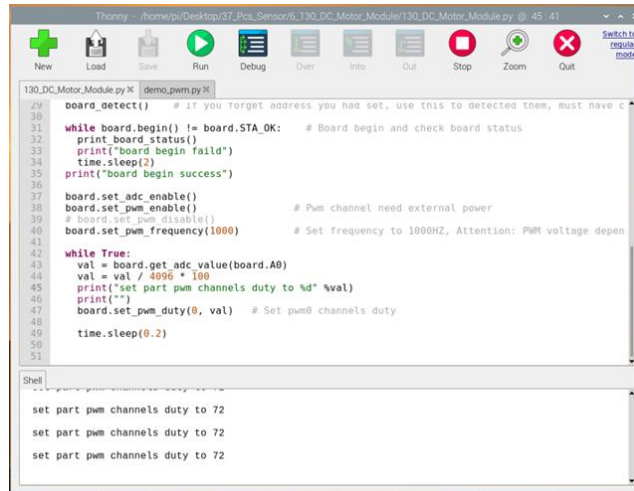
Use 130 DC Motor Fan on Your Raspberry Pi

- Power the Raspberry Pi on and install the Raspberry Pi expansion board correctly
- Connect the fan to PWM port 0 on the expansion board, and the analog rotation sensor to analog port 0



- Install Python dependency libraries and git, and you need to get your Raspberry Pi connect to internet for this step(skip if installed). In the terminal, type the following instructions and press 'Enter' `sudo apt-get install build-essential python-dev python-smbus git`
- Install the drive library and program. In the terminal, type the following instructions in order and press 'Enter' `cd \~`
`git clone https://github.com/DFRobotd1/37_Pcs_Sensor.git`
- Find 130_DC_Motor_Module.py in \37_Pcs_Sensor\7_130_DC_Motor_Module folder, open it with Thonny Pyth IDE and run it, the fan speed can be adjusted through the rotation sensor.





```
29 board_detect() # If you forget address you had set, use this to detected them, must have c
30
31 while board.begin() != board.STA_OK: # Board begin and check board status
32     print_board_status()
33     print("board begin faild")
34     time.sleep(2)
35     print("board begin success")
36
37 board.set_adc_enable()
38 board.set_pwm_enable() # Pwm channel need external power
39 # board.set_pwm_disable()
40 board.set_pwm_frequency(1000) # Set frequency to 1000HZ, Attention: PWM voltage depen
41
42 while True:
43     val = board.get_adc_value(board.A0)
44     val = val / 4096 * 100
45     print("set part pwm channels duty to %d" %val)
46     print("")
47     board.set_pwm_duty(0, val) # Set pwm0 channels duty
48
49     time.sleep(0.2)
50
51
```

```
Shell
set part pwm channels duty to 72
set part pwm channels duty to 72
set part pwm channels duty to 72
```

Lesson 8: JoyStick

Learning Contents

Introduction to JoyStick

The JoyStick produced by DFRobot is made with original high-quality metal PS2 rocker potentiometer. With (X, Y) 2 axis analog output and (Z) 1 button digital output, it can maintain good contact and mechanical properties no matter how you torture it. The 3 signals are respectively connected to the Arduino sensor expansion board through the 3P line, occupying only 3 ports for its control.

Features of its new version:

- Operating Voltage: 3.3 V/5 V, suitable for more 3.3 V controllers.
- Standard Size: The distance between two mounting holes with a diameter of 3mm is several times that of 5mm
- Easy to Identify: The two analog output interfaces are marked with S, while the digital interface is marked with D
- High-quality connectors, resistant to repeated plugging and unplugging
- Immersion gold technology, not only improving the quality of PCB board, but also being with golden fonts.

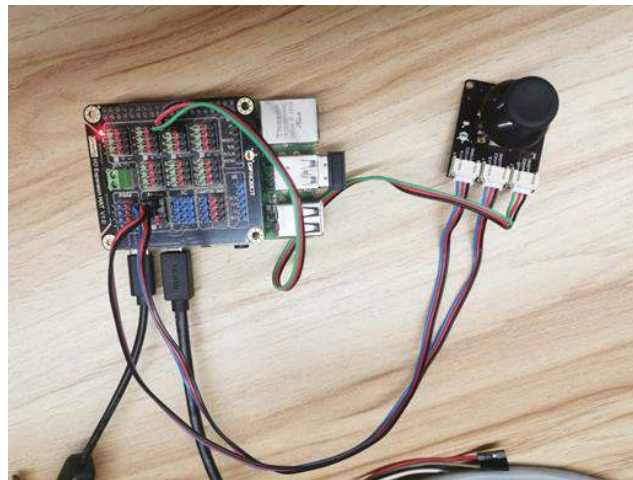
Precautions

The layout of the new version of the analog sensor port has the following two improvements. Please refer to the blog [How to change the layout of the data cable connector](#). When using this sensor on the IO expansion board, you may need to adjust the layout of the connector. For your convenience, we will make more improvements, so stay tuned.

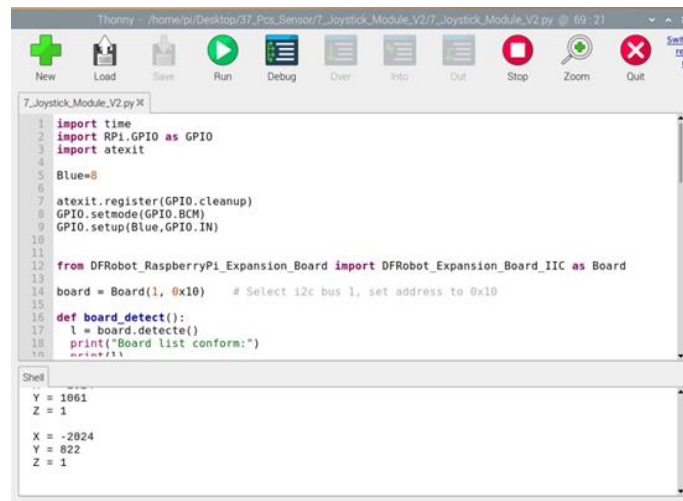
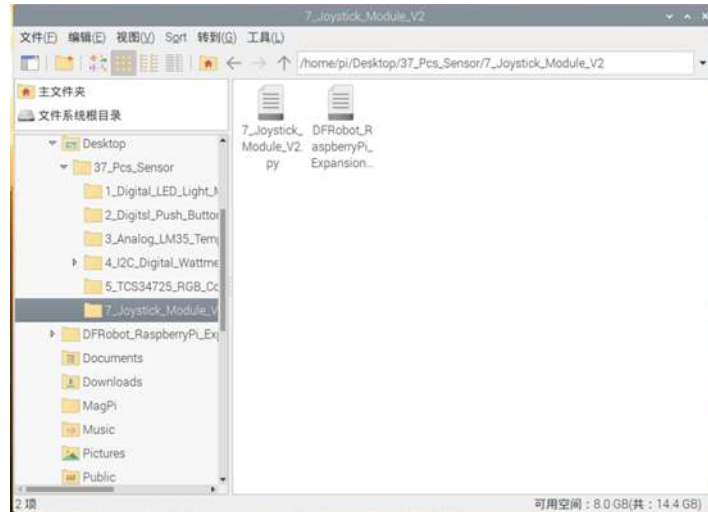


Use JoyStick on Your Raspberry Pi

- Power the Raspberry Pi on and install the Raspberry Pi expansion board correctly
- Connect the X-axis output port of the JoyStick to the analog port 0 on the expansion board, the Y-axis output port to the analog port 1, and the Z-axis output to the digital port 8



- Install Python dependency libraries and git, and you need to get your Raspberry Pi connect to internet for this step(skip if installed). In the terminal, type the following instructions and press 'Enter' `sudo apt-get install build-essential python-dev python-smbus git`
- Download drive library and program. In the terminal, type the following instructions and press 'Enter' `cd \~
git clone https://github.com/DFRobotdl/37_Pcs_Sensor.git`
- Find 7_Joystick_Module_V2.py in \37_Pcs_Sensor\8_Joystick_Module_V2, open and run it with Thonny Pyth IDE, you can see the printed X, Y, Z axis data. The value will change if you flip or press the joystick.



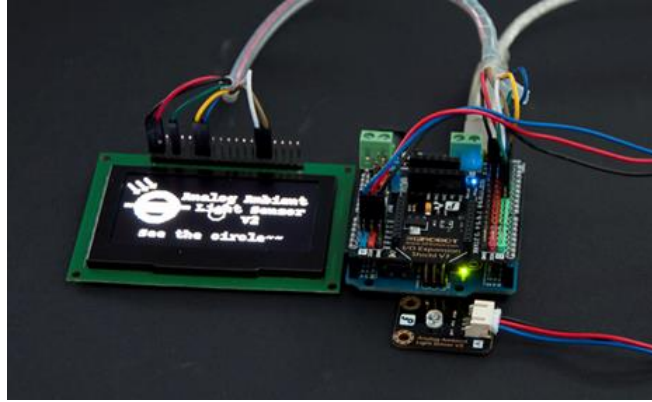
Lesson 9: Analog Light Sensor

Learning Contents

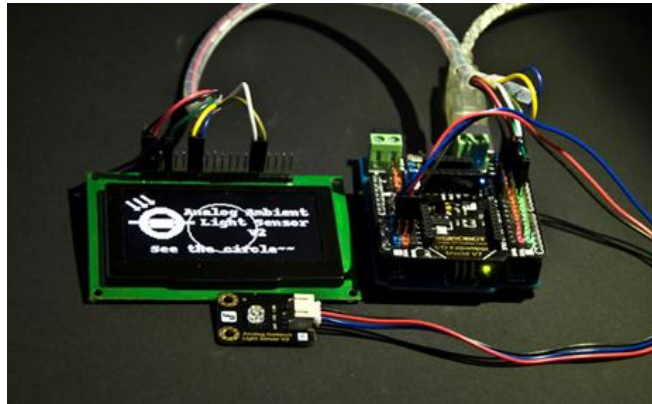
Introduction

Based on PT550 environmentally friendly photodiode, this light sensor can be used to detect the intensity of ambient light. It is usually used to produce interactive works that produce special effects with changes in light intensity. The entire module is connected to the IO expansion board with a 3P analog data cable. As long as the color is corresponding, it will not be inserted wrong, really convenient.

- Wider Operating Voltage: 3.3 V~5 V
- Standard fixing hole design, two 3mm fixing holes separated by 5cm.



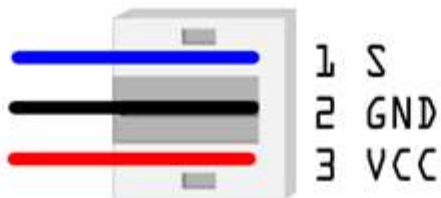
This LCD screen can display the detected light intensity of the sensor, which is reflected by the size of the circle



Precautions

The layout of the new version of the analog sensor port has the following two improvements. Please refer to our blog "[How to change the data line connector layout](#)" instructions. When using the sensor on the IO expansion board, you may need to adjust the layout of the connector. For your convenience, we will make more improvements, so stay tuned.

old version

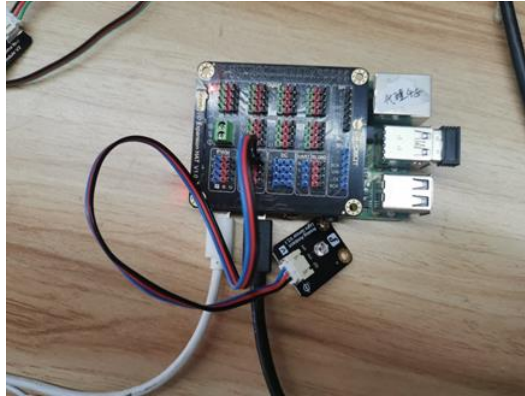


new version

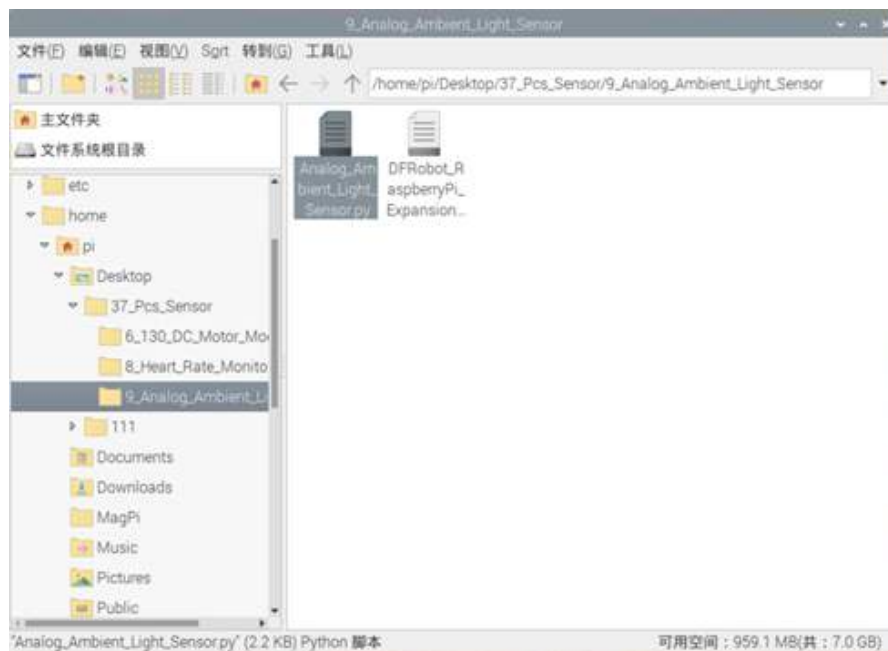


Use Analog Light Sensor on Your Raspberry Pi

- Power the Raspberry Pi on and install the Raspberry Pi expansion board correctly
- Connect the sensor to analog port 0 on the expansion board



- Install Python dependency libraries and git, and you need to get your Raspberry Pi connect to internet for this step(skip if installed). In the terminal, type the following instructions and press 'Enter' `sudo apt-get install build-essential python-dev python-smbus git`
- Install the drive library and program. In the terminal, type the following instructions and press 'Enter' `cd \~`
`git clone https://github.com/DFRobotdl/37_Pcs_Sensor.git`
- Find `Analog_Ambient_Light_Sensor.py` in `\37_Pcs_Sensor\9_Analog_Ambient_Light_Sensor`, open and run it with Thonny Python IDE, you can see the printed corresponding value of the light



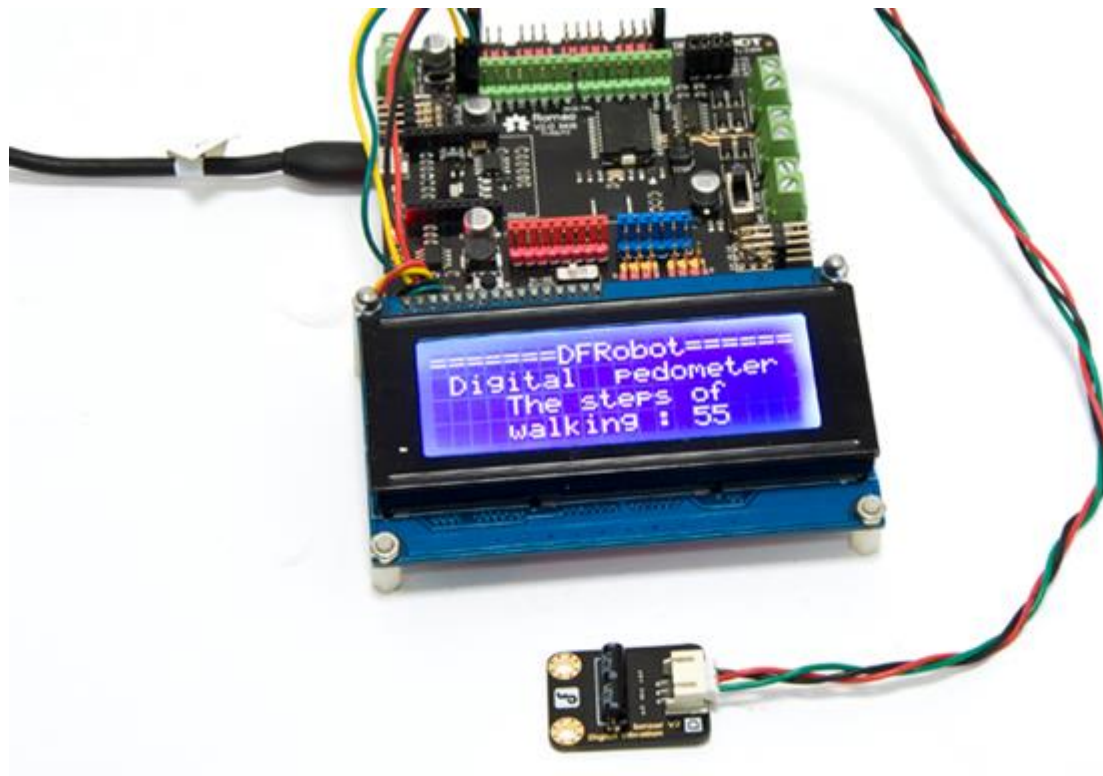
```
Thonny - PyGame/DeskTop/07_PiC_Sensor/8_Analog_Ambient_Light_Sensor/Analog_Ambient_Light_Sensor.py @ 74.1
+ New Load Save Run Debug Over File Help Stop Zoom Quit
Analog_Ambient_Light_Sensor.py
49 if board.last_operate_status != board.STA_OK:
50     print("set board address failed")
51 else:
52     print("set board address success")
53
54
55 while board.begin() != board.STA_OK: # Board begin and check board status
56     print_board_status()
57     print("board begin failed")
58     time.sleep(2)
59     print("board begin success")
60
61 board.set_adc_enable()
62 # board.set_adc_channel()
63
64 while True:
65     val = board.get_adc_value(board.A0) # All channels read
66     # val = board.get_adc_value(board.A1) # A1 channels read
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
Shell
channel: A0, value: 2968
channel: A0, value: 2943
channel: A0, value: 3002
```

Lesson 10: Digital Vibration Sensor

Learning Contents

Introduction

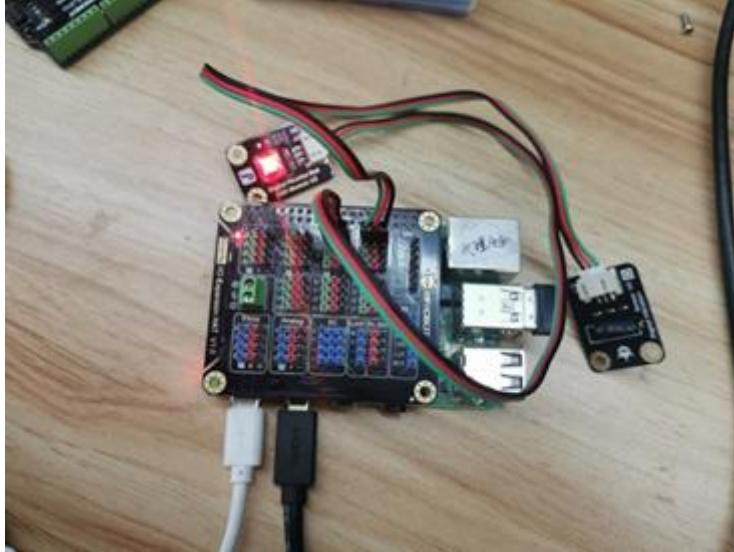
What's the simplest way to check vibration with electric devices? Use a vibration switch to turn on and off the circuit through vibration to generate a signal. Despite a simple structure, you can make full use of this vibration sensor with creative thinking, like step counting, Crash warning light and so on. As long as you have ideas, the usage of simple components will change endlessly. As long as you have ideas, the usage of simple components can be varied an infinite number of times.



This is a very simple pedometer that can count the number of steps you take.

Use Digital Vibration Sensor on Your Raspberry Pi

- Connect the digital LED light-emitting module to the pin 12 on extension board, and connect the digital vibration sensor to the pin 8.



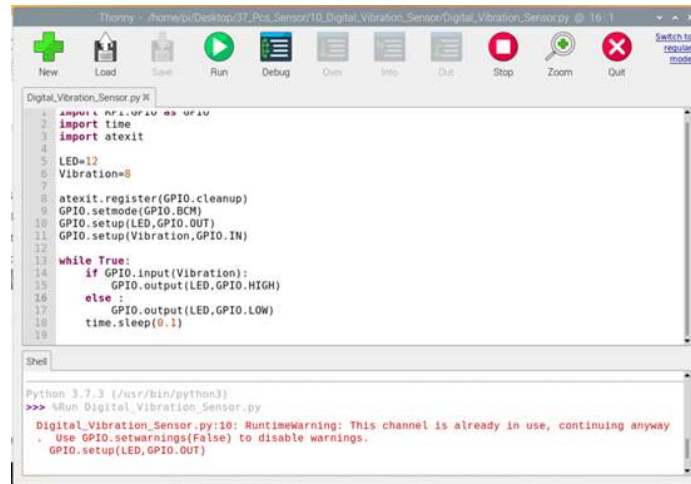
- Open Thonny Python IDE to copy the following program into it

```
import RPi.GPIO as GPIO
import time
import atexit

LED=12
Vibration=8

atexit.register(GPIO.cleanup)
GPIO.setmode(GPIO.BCM)
GPIO.setup(LED,GPIO.OUT)
GPIO.setup(Vibration,GPIO.IN)

while True:
    if GPIO.input(Vibration):
        GPIO.output(LED,GPIO.HIGH)
    else :
        GPIO.output(LED,GPIO.LOW)
    time.sleep(0.1)
```



```
1 import RPi.GPIO as GPIO
2 import time
3 import atexit
4
5 LED=12
6 Vibration=8
7
8 atexit.register(GPIO.cleanup)
9 GPIO.setmode(GPIO.BCM)
10 GPIO.setup(LED,GPIO.OUT)
11 GPIO.setup(Vibration,GPIO.IN)
12
13
14 while True:
15     if GPIO.input(Vibration):
16         GPIO.output(LED,GPIO.HIGH)
17     else :
18         GPIO.output(LED,GPIO.LOW)
19         time.sleep(0.1)
```

Shell

```
Python 3.7.3 (/usr/bin/python3)
>>> !run Digital_Vibration_Sensor.py
Digital_Vibration_Sensor.py:10: RuntimeWarning: This channel is already in use, continuing anyway
  Use GPIO.setwarnings(False) to disable warnings.
  GPIO.setup(LED,GPIO.OUT)
```

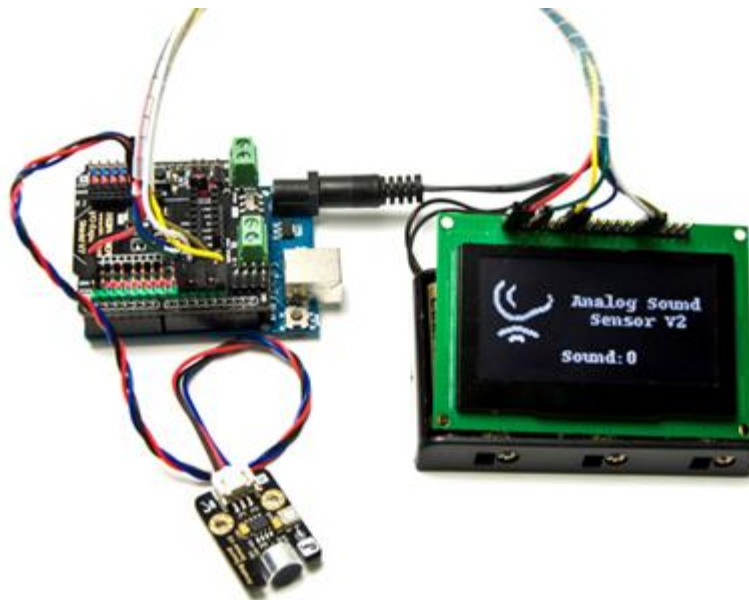
- Click 'Run', you can observe that when the sensor detects vibration, the LED will go out.

Lesson 11: Analog Sound Sensor

Learning Contents

Introduction

This is a simple and affordable microphone through which the Arduino can sense the level of the sound and convert it into an analog signal. That is, the volume is reflected by the feedback voltage value. The analog data line directly corresponds to the analog port on the IO expansion board V7. Plug in and burn the code, you can use it.



The sensor reads 0 when completely muted. When there is music nearby, it will read various readings with the volume.



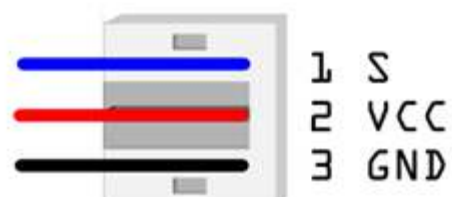
Precautions

The layout of the new version of the analog sensor port has the following two improvements. Please refer to our blog "[How to change the data line connector layout](#)" instructions. When using the sensor on the IO expansion board, you may need to adjust the layout of the connector. For your convenience, we will make more improvements, so stay tuned.

old version

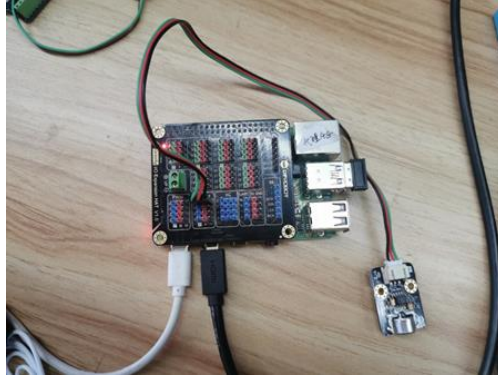


new version

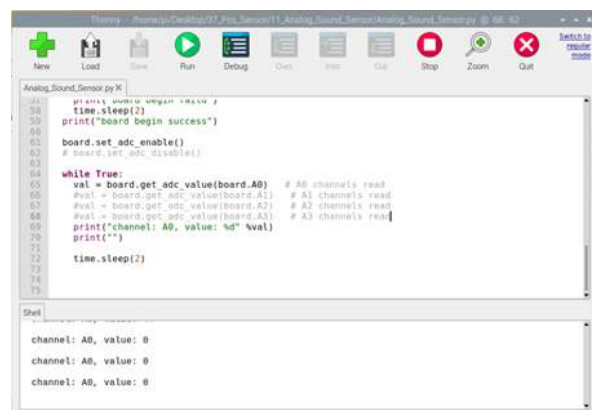
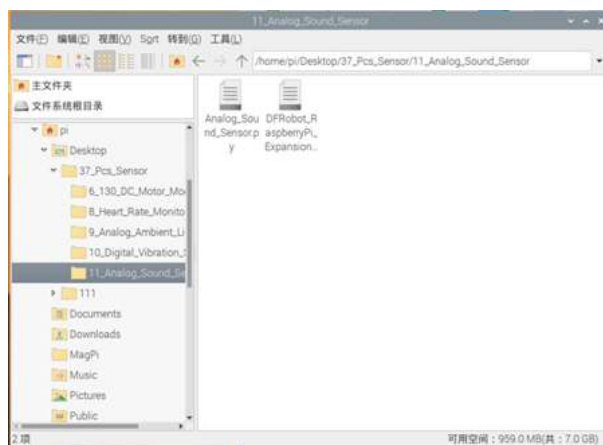


Use Analog Sound Sensor on Your Raspberry Pi

- Power the Raspberry Pi on and install the Raspberry Pi expansion board correctly
- Connect the sensor to analog port 0 on the expansion board



- Install Python dependency libraries and git, and you need to get your Raspberry Pi connect to internet for this step(skip if installed). In the terminal, type the following instructions and press 'Enter' `sudo apt-get install build-essential python-dev python-smbus git`
- Download the drive library and program. In the terminal, type the following instructions and press 'Enter' `cd \~`
`git clone https://github.com/DFRobotdl/37_Pcs_Sensor.git`
- Find `Analog_Sound_Sensor.py` in `\37_Pcs_Sensor\11_Analog_Sound_Sensor`, open and run it with Thonny Python IDE, you can see the printed sound values. The smaller value, the louder volume. And vice versa.

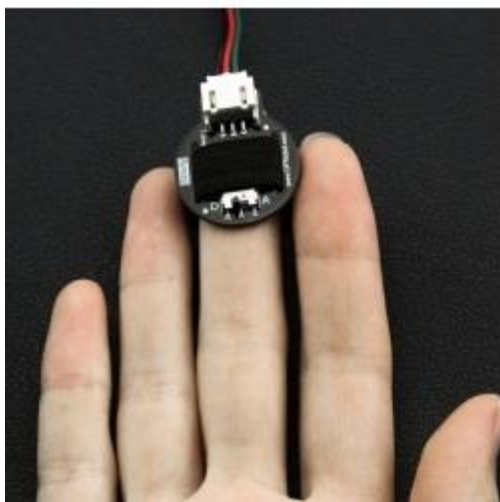
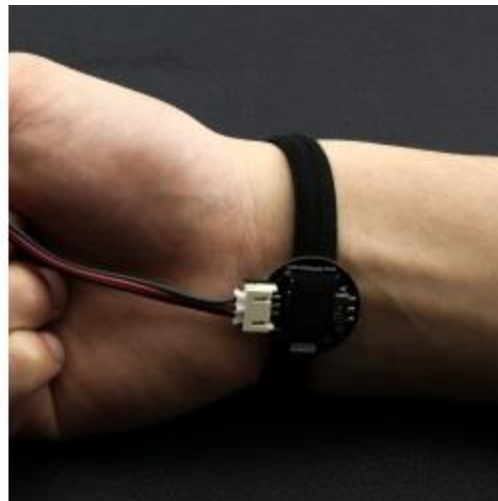


Lesson 12: Heart Rate Monitor Sensor

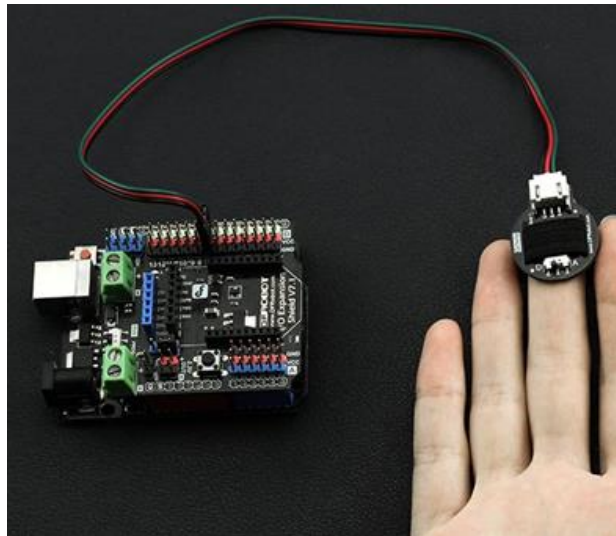
Learning Contents

Introduction

The DFRobot heart rate sensor, despite a tiny size of just a thumb, can monitor human heart rates. Compatible with Arduino main controllers, this plug-and-play module is really convenient to use that it's equipped with Gravity 3-Pin interface. It uses PPG (Photo Plethysmo Graphy) to measure heart rate, a low-cost optical technology that monitors the human heart rate by detecting blood oxygen levels in subcutaneous capillaries. Besides, this technology features fast response, stable performance and strong adaptability. As being equipped with two mounting holes, the sensor can be wrapped on your finger, wrist, earlobe or other areas where it has contact with your skin.



The module has two signal output modes: square wave and pulse wave, which can be freely switched through the on-board switch. The pulse wave outputs a continuous heart rate waveform, while the square wave outputs a corresponding square wave according to heart rates.

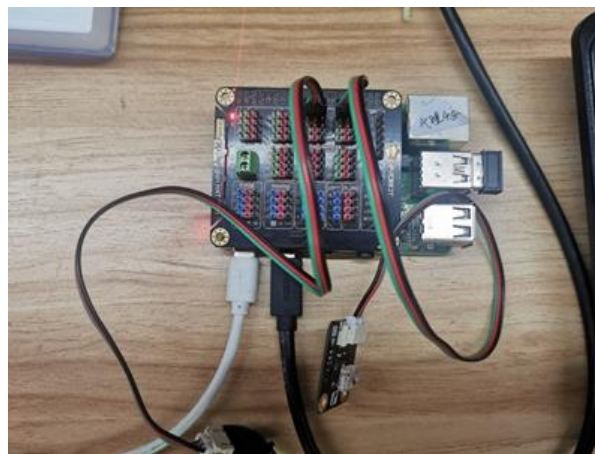


Precautions

1. This is a static heart rate sensor. Please do not move or press too tightly during its measurement.
2. This product is NOT a professional medical device and should not be used to diagnose or treat medical conditions.

Use Heart Rate Sensor on Your Raspberry Pi

- Connect the digital LED light-emitting module to the pin 12 on extension board, the heart rate sensor to the pin 8, and the analog light sensor to the analog port 0.



- Open Thonny Python IDE and copy the following program into it.

```

import RPi.GPIO as GPIO
import time
import atexit

LED=12
Heart_Rate=8

atexit.register(GPIO.cleanup)
GPIO.setmode(GPIO.BCM)
GPIO.setup(LED,GPIO.OUT)
GPIO.setup(Heart_Rate,GPIO.IN)

while True:
    if GPIO.input(Heart_Rate):
        GPIO.output(LED,GPIO.HIGH)
    else :
        GPIO.output(LED,GPIO.LOW)
    time.sleep(0.1)

```

The screenshot shows a Python IDE window titled 'Thonny' with the following code in the editor:

```

1 import RPi.GPIO as GPIO
2 import time
3 import atexit
4
5 LED=12
6 Heart_Rate=8
7
8 atexit.register(GPIO.cleanup)
9 GPIO.setmode(GPIO.BCM)
10 GPIO.setup(LED,GPIO.OUT)
11 GPIO.setup(Heart_Rate,GPIO.IN)
12
13 while True:
14     if GPIO.input(Heart_Rate):
15         GPIO.output(LED,GPIO.HIGH)
16     else :
17         GPIO.output(LED,GPIO.LOW)
18     time.sleep(0.1)
19

```

The Shell window shows the execution output:

```

Python 3.7.3 (/usr/bin/python3)
>>> %Run Heart_Rate_Monitor_Sensor_and_Digital_Blue_LED_Light_Module.py
Heart_Rate_Monitor_Sensor_and_Digital_Blue_LED_Light_Module.py:10: RuntimeWarning: This channel is already in use, continuing anyway. Use GPIO.setwarnings(False) to disable warnings.
  GPIO.setup(LED,GPIO.OUT)

```

- Then the LED will go on and off with your heartbeat.

Lesson 13 Conductivity Sensor Switch

Preface

Lead-in

In this section we will introduce the conductivity switch sensor. We can use it to test the conductivity of the object, and can also use it as a switching device for special occasions.

Introduction

The conductivity switch is to conduct through two exposed conductors. When the two sides are connected through a certain medium, they are turned on. And the sensor can detect whether there is an object connection between the two poles. When the resistance value of the connected object is less than 10M, it can output a high level. You can use it to make a fruit piano, music wind chime, handshake sensor light and other interesting applications.

Hardware

- [Gravity: 37 Pcs Sensor Set](#)
- [Raspberry Pi 4 Model B](#)
- [IO Expansion HAT for Raspberry Pi 4B/3B+](#)
- [8GB + SanDisk Class10 SD/MicroSD Memory Card](#)
- [5V@3A USB Power Supply](#)
- [8.9 IPS Touch Display](#)

Learning Contents

Connection

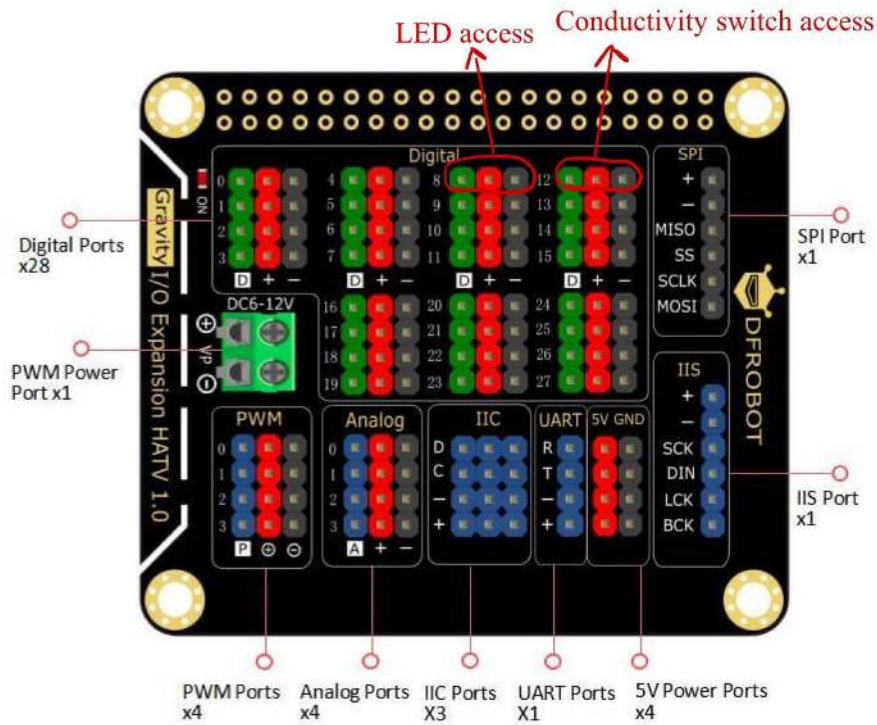
- Connect the Raspberry Pi correctly to devices such as the screen, power, keyboard and mouse.

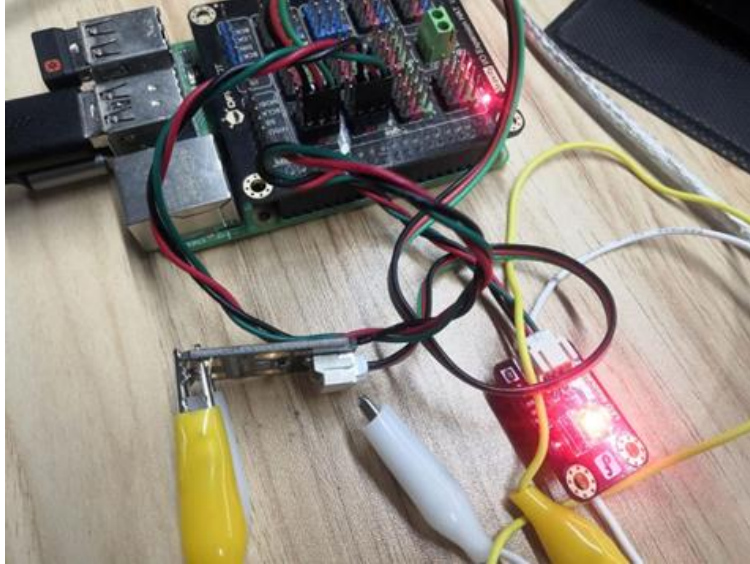


- Install the Raspberry Pi IO expansion board on the Raspberry Pi and connect the LED light module to the No. 12 digital port on the board, and the conductivity switch sensor module to the No. 8 digital port. When we touch to connect the two alligator clips with our hands or other conductive objects, we can see the corresponding phenomenon.



- Expansion Board Connection

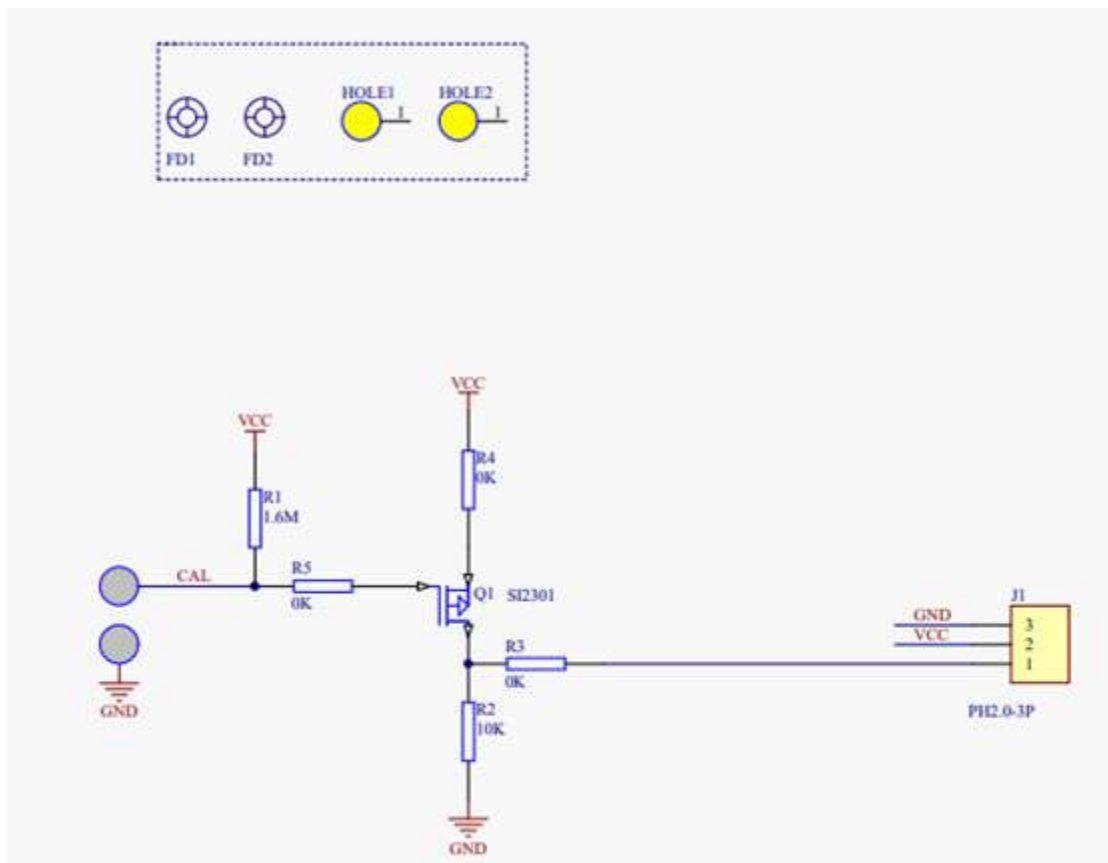




Operating Principle

Change the level signal as a switch by conducting two metal sheets.

Schematic



Software

- Open Thonny Python IDE to copy the following program into it



```
import RPi.GPIO as GPIO    # Import the python module provided by the Raspberry Pi
import time                # Import time package to control flicker

LED=12                    # Define the pin number to which the LED is connected
Electrical_switch = 8     # Define the pin number to which the
switch is connected

GPIO.setmode(GPIO.BCM)   # Set GPIO mode, BCM mode is common to all
Raspberry Pi
GPIO.setup(Electrical_switch,GPIO.IN)    # Set GPIO12 to output mode
GPIO.setup(LED,GPIO.OUT)                # Set GPIO8 to input mode
GPIO.output(LED,GPIO.HIGH)              #Define LED original value

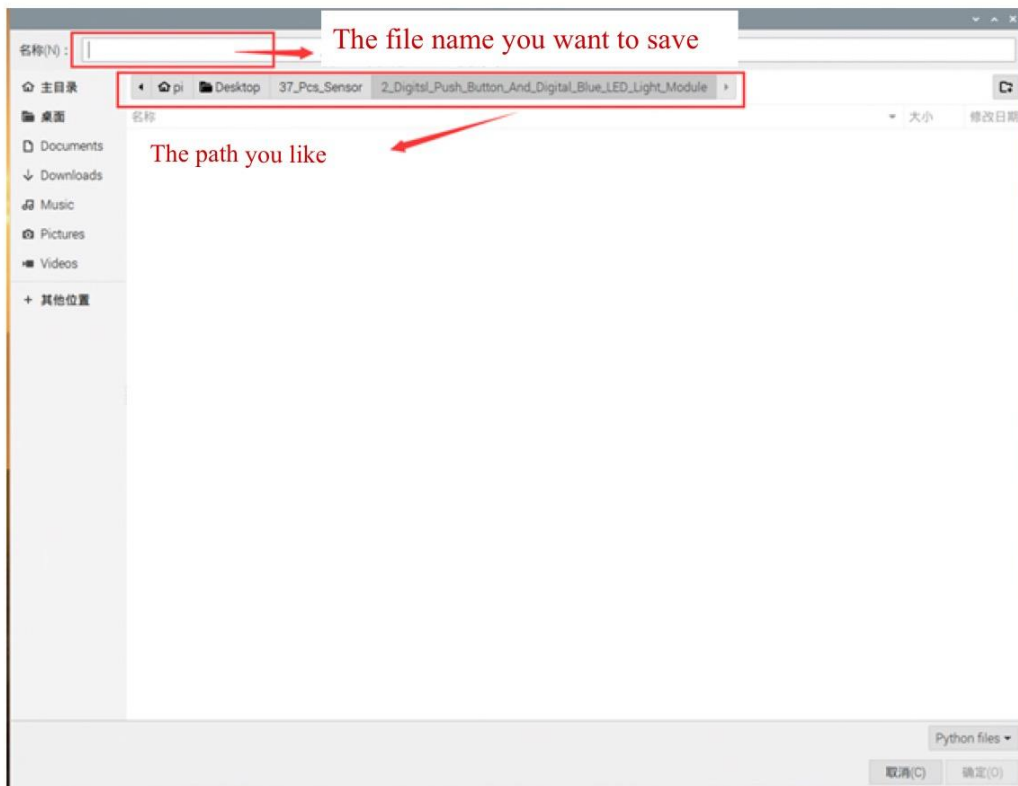
while True:                        # Execute the following commands in an
infinite loop
    key = GPIO.input(Electrical_switch)
    if (key ):                    # Judge whether the button is pressed
        GPIO.output(LED,GPIO.LOW)    # Button pressed, start the micro vibrator
    else :                        # If GPIO8 is low (that is, the button is released),
execute the following statement
        GPIO.output(LED,GPIO.HIGH)   # Not start the micro vibrator
    time.sleep(0.01)              # Delay one second, here is to control the frequen
cy of the query key
```

- Click 'Save' after copying



```
1 import RPi.GPIO as GPIO #导入树莓派提供的python模块
2 import time #导入时间包,用于延时时间
3
4 LED = 12 #定义LED灯接入引脚
5 Electrical_switch = 8 #定义电导开关接入引脚
6
7 GPIO.setmode(GPIO.BCM) #设置GPIO模式,BCM模式在所有树莓派通用
8 GPIO.setup(Electrical_switch,GPIO.IN) #设置GPIO12为输入模式
9 GPIO.setup(LED,GPIO.OUT) #设置GPIO18为输出模式
10 GPIO.output(LED,GPIO.HIGH) #定义LED初始值
11
12 while True: #无限循环执行下面的指令
13     key = GPIO.input(Electrical_switch) #判断按钮是否按下
14     if (key ): #按钮按下,启动微型驱动器
15         GPIO.output(LED,GPIO.LOW) #如果GPIO18为低电平(即按钮没按下)就执行下面的语句
16     else : #未启动微型驱动器
17         GPIO.output(LED,GPIO.HIGH)
18     time.sleep(0.01) #等待0.1秒,在这里的作用是控制查询按钮的频率。
```

- Set the file name and the save path you like



- Press 'Run' or 'Stop' the program



```
1 import RPi.GPIO as GPIO #导入树莓派提供的python模块
2 import time #导入时间包，用于延时时间
3
4 LED = 12 #定义LED灯接入引脚
5 Electrical_switch = 8 #定义电导开关接入引脚
6
7 GPIO.setmode(GPIO.BCM) #设置GPIO模式，BCM模式在所有树莓派通用
8 GPIO.setup(Electrical_switch,GPIO.IN) #设置GPIO12为输入模式
9 GPIO.setup(LED,GPIO.OUT) #设置GPIO8为输出模式
10 GPIO.output(LED,GPIO.HIGH) #定义LED初始值
11
12 while True: #无限循环执行下面的指令
13     key = GPIO.input(Electrical_switch) #判断按键是否按下
14     if (key ): #按键按下，启动微型驱动器
15         GPIO.output(LED,GPIO.LOW) #如果GPIO8为低电平（即按键按下）就执行下面的语句
16     else : #未启动微型驱动器
17         GPIO.output(LED,GPIO.HIGH) #等待0.1秒，在这里的作用是控制LED灯的频率。|
18     time.sleep(0.01)
```

- After running it, check the corresponding phenomenon

<http://https://www.bilibili.com/video/BV1z54y1b763/>

Lesson 14: Analog Voltage Monitoring Module

Preface

Introduction

Based on the resistance divider principle, this voltage monitoring sensor can be used with the Arduino sensor expansion board to detect the voltage and power level, monitoring the power of interactive media works or the robot power supply.

This module can reduce the input voltage by 5 times. Since the maximum Arduino analog input voltage is 5 V, its input voltage cannot be greater than $5 \text{ V} \times 5 = 25 \text{ V}$.

Hardware

- [Gravity: 37 Pcs Sensor Set](#)
- [Raspberry Pi 4 Model B](#)
- [IO Expansion HAT for Raspberry Pi 4B/3B+](#)
- [8GB + SanDisk Class10 SD/MicroSD Memory Card](#)
- [5V@3A USB Power Supply](#)
- [8.9 IPS Touch Display](#)

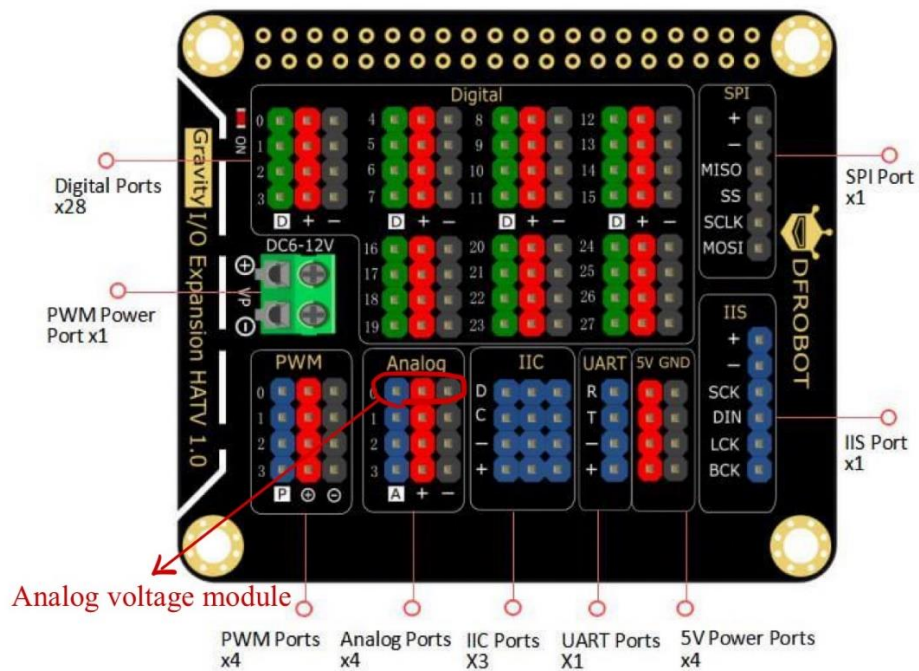
Learning Contents

Connection

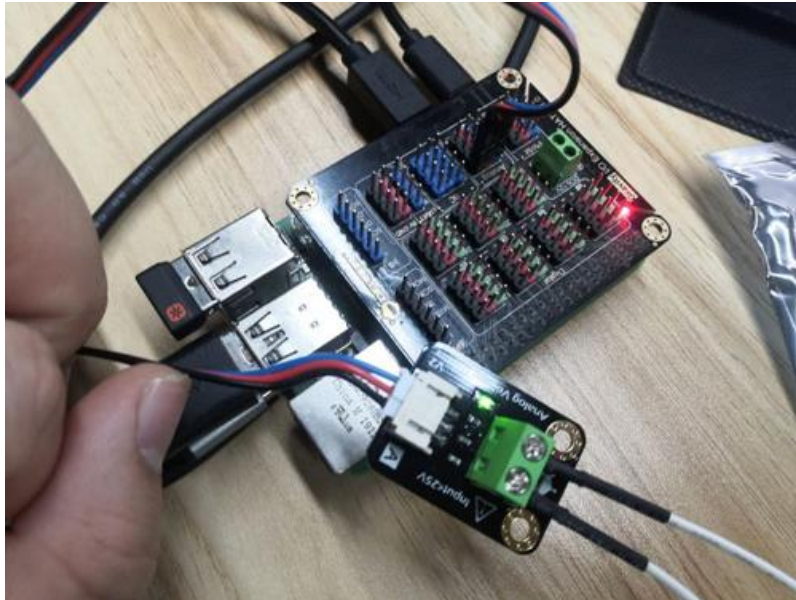
- Connect the Raspberry Pi correctly to devices such as the screen, power, keyboard and mouse.



- Connect the pin of this module to pin 0 on the Analog board of the Raspberry Pi expansion board. The schematic diagram is as follows.

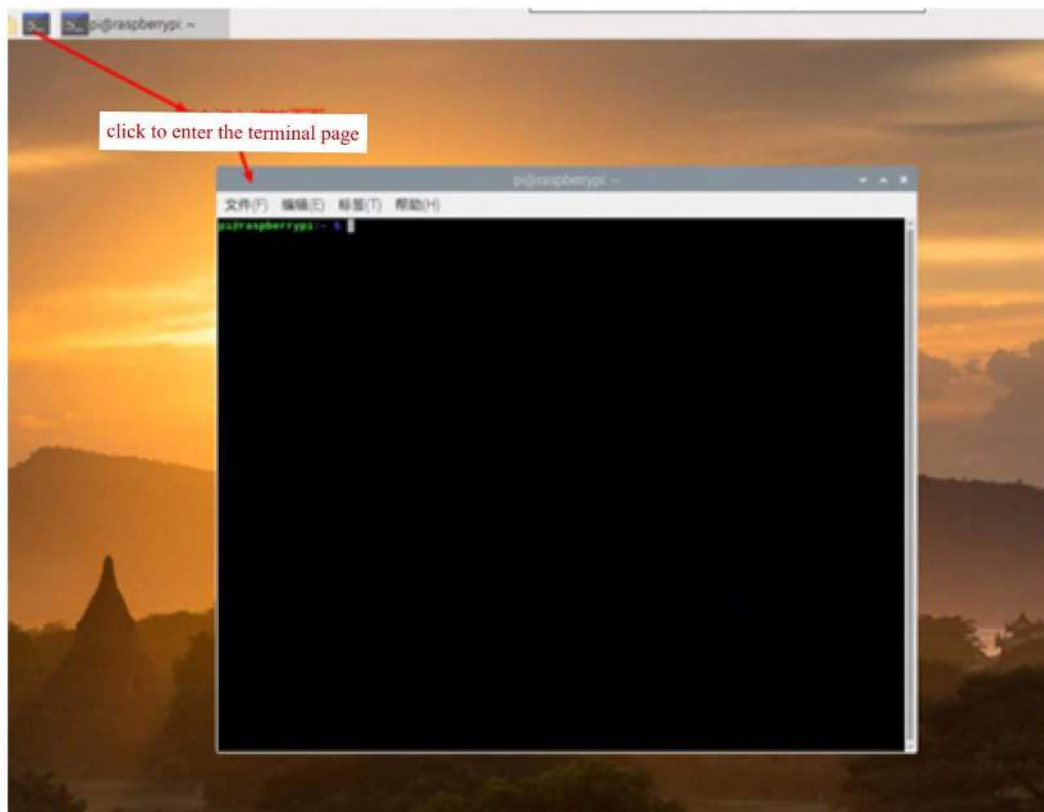


- Connect a power to power it on. The wiring diagram is as follows.

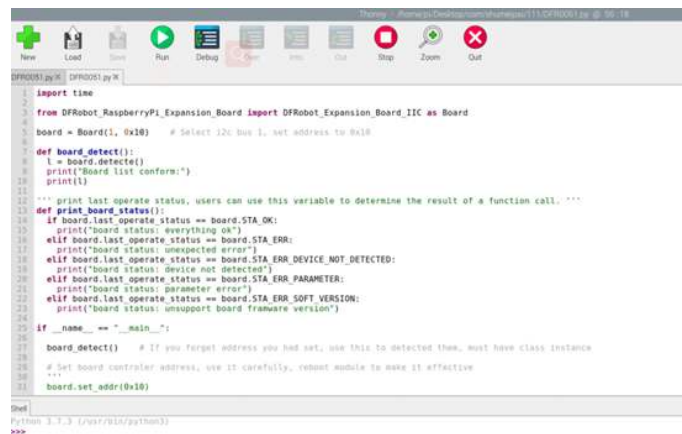
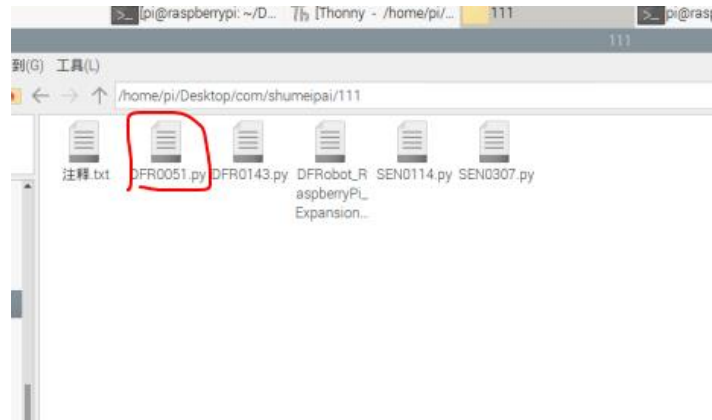


Program

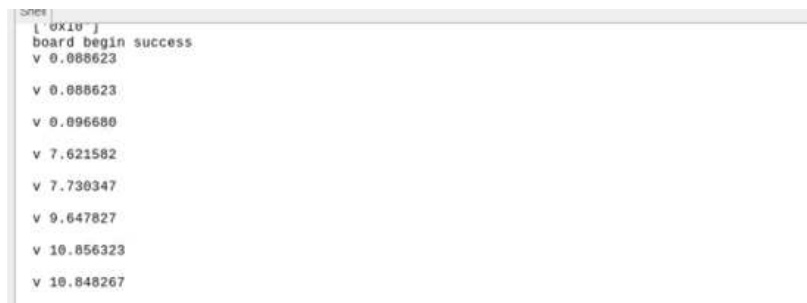
- Open the system to enter the terminal



- Install Python dependency libraries and git, and you need to get your Raspberry Pi connect to internet for this step(skip if installed). In the terminal, type the following instructions and press 'Enter' `sudo apt-get install build-essential python-dev python-smbus git`
- Download the drive library and program. `git clone https://github.com/DFRobotdl/111.git`
- Find DFR0051.py in the file you just downloaded before, click to run it



- Adjust the input voltage, then the analog port will read it.



Lesson 15: Analog Capacitive Soil Moisture Sensor

Preface

Introduction

This is a simple moisture sensor that can be used to detect soil moisture. When the soil is short of water, its output value will decrease, otherwise it will increase. This kind of sensor is mainly used to measure the relative water content of soil, do soil moisture monitoring, agricultural irrigation and forestry protection.

Hardware

- [Gravity: 37 Pcs Sensor Set](#)
- [Raspberry Pi 4 Model B](#)
- [IO Expansion HAT for Raspberry Pi 4B/3B+](#)
- [8GB + SanDisk Class10 SD/MicroSD Memory Card](#)
- [5V@3A USB Power Supply](#)
- [8.9 IPS Touch Display](#)

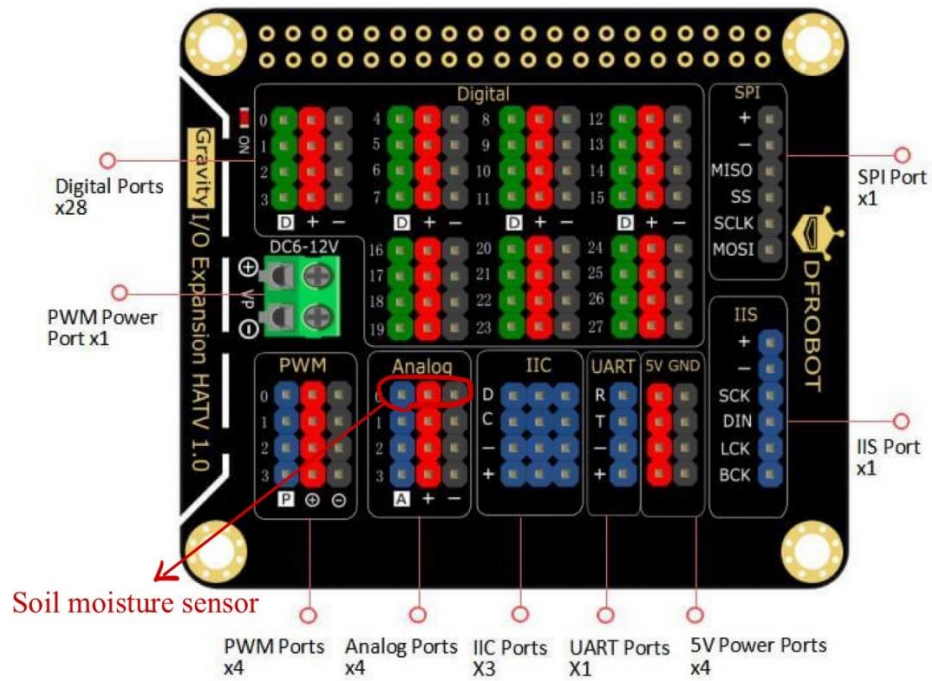
Learning Contents

Connection

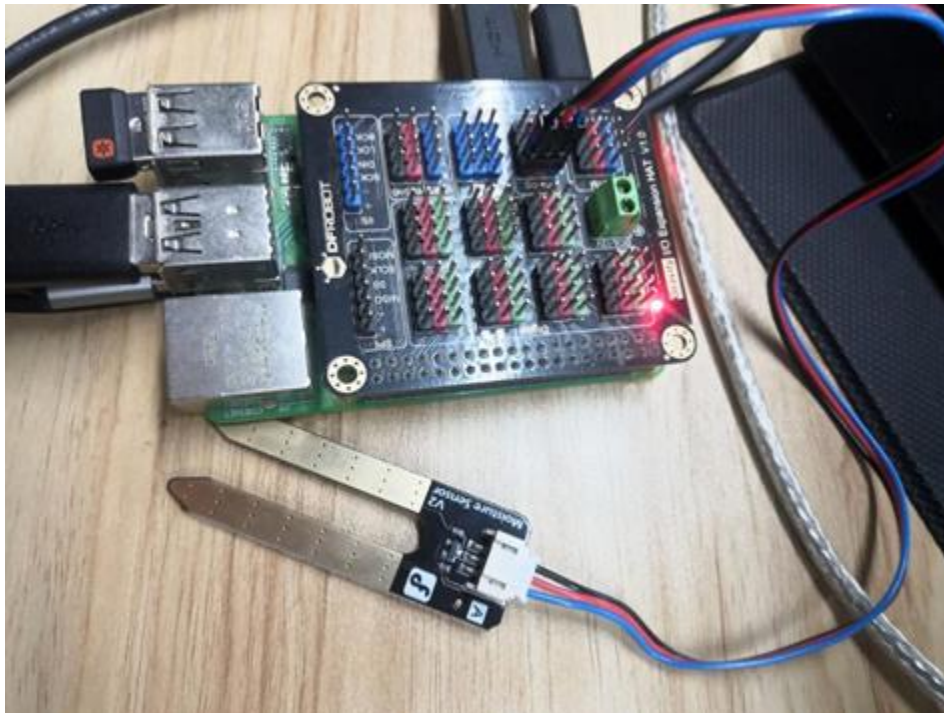
- Connect the Raspberry Pi correctly to devices such as the screen, power, keyboard and mouse.



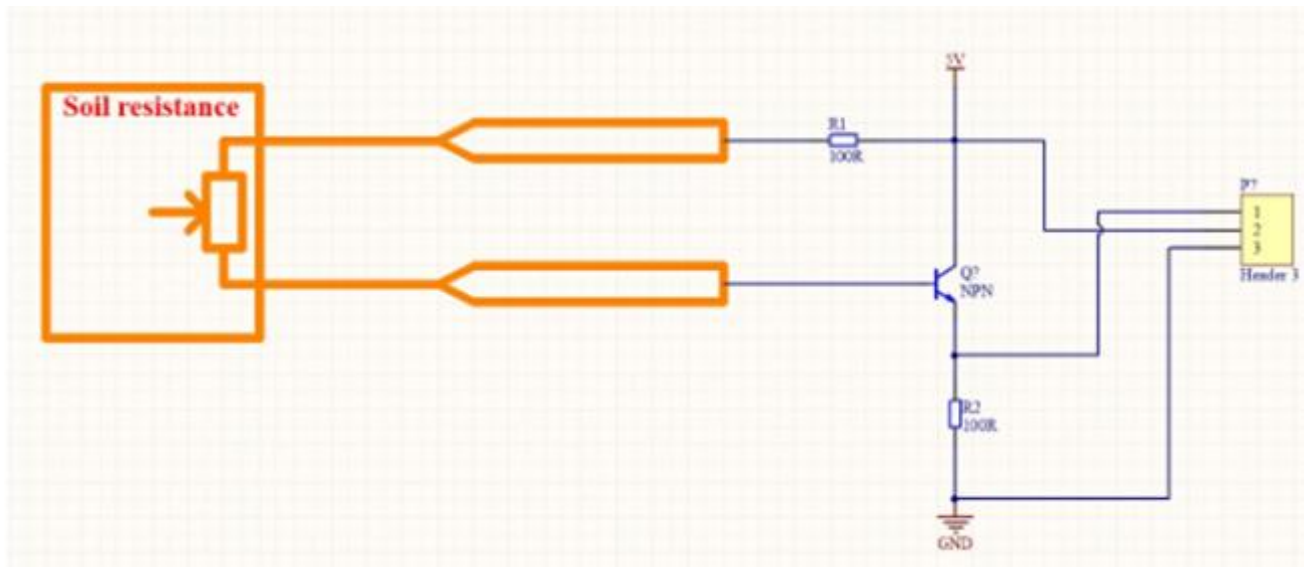
- Connect the sensor to analog port A0 on Raspberry Pi expansion board. The wiring diagram is as follows.



- Find components and connect



Schematic and operating principle



The soil moisture sensor judges the soil moisture content by its water level.

As shown in the figure, when the soil moisture sensor probe is suspended in the air, the base of the triode is open, and the output of the triode is 0. When it is inserted into the soil, the resistance value of the soil is different due to the different moisture content. Then the base of the triode will provide a variable conduction current. The conduction current from the collector to the emitter of the triode is controlled by the base, converted into a voltage after the pull-down resistor of the emitter.

Program

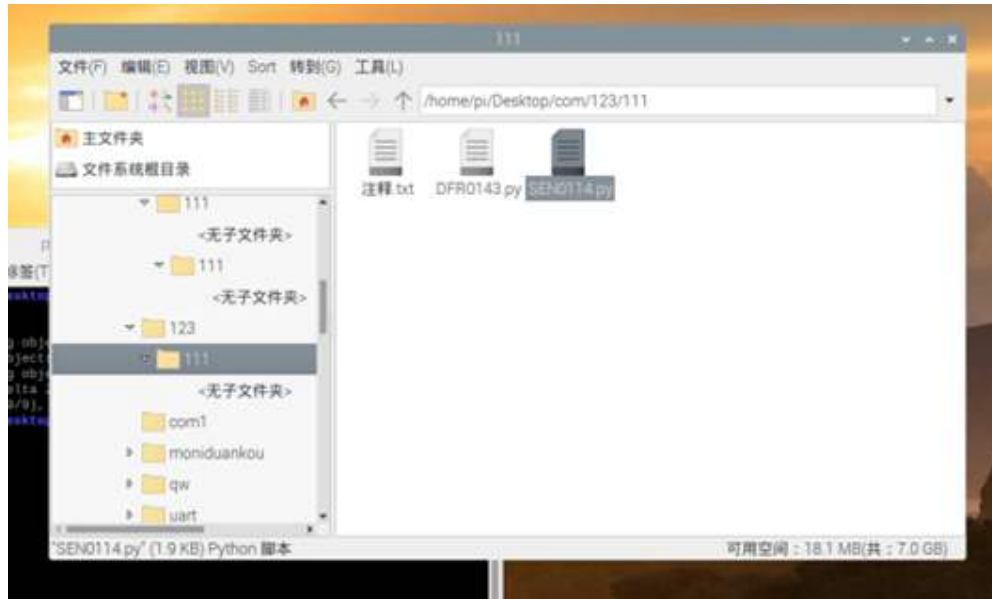
- Install Python dependency libraries and git, and you need to get your Raspberry Pi connect to internet for this step(skip if installed). In the terminal, type the following instructions and press 'Enter'

```
sudo apt-get install build-essential python-dev python-smbus git
```

- Download the drive library and program.

```
git clone https://github.com/DFRobotdl/111.git
```

- Find SEN0114.py in the file you downloaded before, open and run it with, you can see the printed data.



```
Thonny - Home/pi
Thonny - Home/pi/Desktop/com/123/111/SEN0114.py @ 30.1%

New Load Save Run Debug View Help Shell Out Stop Zoom Out

SEN0114.py X 运行
1 import time
2 from DFRobot_RaspberryPi_Expansion_Board import DFRobot_Expansion_Board_IIC as Board
3
4 board = Board(1, 0x10) # Select I2c bus 1, set address to 0x10
5
6 def board_detect():
7     l = board.detect()
8     print("Board list conform:")
9     print(l)
10
11
12 ''' print last operate status, users can use this variable to determine the result of a function call. '''
13 def print_board_status():
14     if board.last_operate_status == board.STA_OK:
15         print("board status: everything ok")
16     elif board.last_operate_status == board.STA_ERR:
17         print("board status: unexpected error")
18     elif board.last_operate_status == board.STA_ERR_DEVICE_NOT_DETECTED:
19         print("board status: device not detected")
20     elif board.last_operate_status == board.STA_ERR_PARAMETER:
21         print("board status: parameter error")
22     elif board.last_operate_status == board.STA_ERR_SOFT_VERSION:
23         print("board status: unsupport board framwure version")
24
25 if __name__ == "__main__":
26
27     board_detect() # If you forget address you had set, use this to detected them, must have class instance
28
29 # Set board controller address, use it carefully, reboot module to make it effective
30 ...
31 board.set_addr(0x10)
32
33 Shell
```

```
soil moisture: A0, value: 0
soil moisture: A0, value: 0
soil moisture: A0, value: 0
soil moisture: A0, value: 38
soil moisture: A0, value: 785
soil moisture: A0, value: 880
soil moisture: A0, value: 644
soil moisture: A0, value: 325
soil moisture: A0, value: 638
```

Lesson 16: URM09 Ultrasonic Sensor

Preface

Introduction

This is an open dual-probe ultrasonic distance measurement module. Equipped with Gravity standard PH2.0-3P vertical mount interface, it outputs analog voltage, compatible with arduino, Raspberry Pi, and other controllers with logic level of 3.3 V or 5 V.

This module comes with temperature compensation to avoid ambient temperature from affecting its measurements. With its analog voltage value output, it can directly read the temperature value by ADC conversion, simplifying the operation and reducing the difficulty. Besides, this sensor has been tested that its effective measurement range for flat walls is 2 – 500 cm, its resolution is 1cm, and the error is about $\pm 1\%$. Its dual-probe greatly reduces the detection blind area. And the on-board status indicator light is convenient to check the test progress.

Thanks to its small size, this convenient sensor that allows plug and play has strong environmental applicability, high accuracy, wide measurement range, quite suitable for outdoor environments, especially those with rapid temperature changes. It is also an excellent choice for robots to avoid obstacles automatically, car reversing alarms, doorbells, warning alarms, subway safety line prompts, bank and cash machine one-meter line prompts, and so on.

Preparation

Lead-in

URM09 ultrasonic sensor adopts analog voltage output, so we can get the corresponding test distance through conversion. In this section, we will use Thonny Python IDE basic usage and routine adc to do a simple ultrasonic distance measurement.

Hardware

- [Gravity: 37 Pcs Sensor Set](#)
- [Raspberry Pi 4 Model B](#)
- [IO Expansion HAT for Raspberry Pi 4B/3B+](#)
- [8GB + SanDisk Class10 SD/MicroSD Memory Card](#)
- [5V@3A USB Power Supply](#)
- [8.9 IPS Touch Display](#)

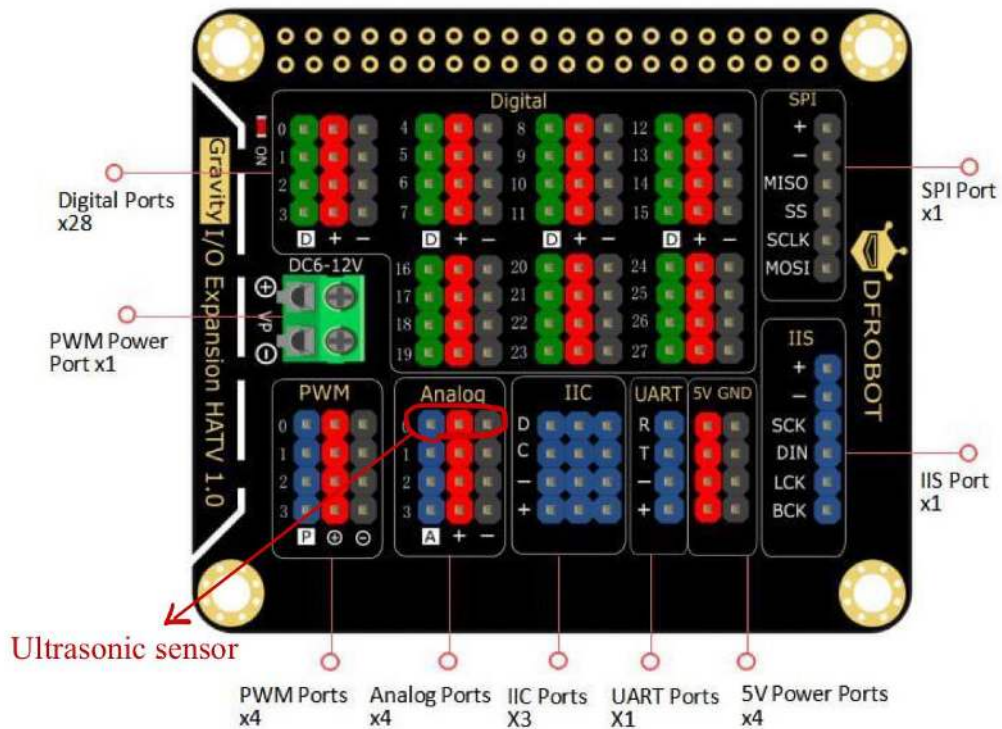
Learning Contents

Connection

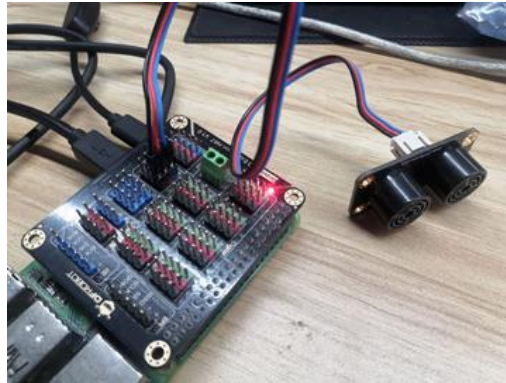
- Connect the Raspberry Pi correctly to devices such as the screen, power, keyboard and mouse.



- Connect this module to analog port A0 on Raspberry Pi expansion board.



- Find this position to connect the sensor correctly.



When we move the ultrasonic sensor, we can receive feedback to detect different distance values

Software

- Install Python dependency libraries and git, and you need to get your Raspberry Pi connect to internet for this step(skip if installed). In the terminal, type the following instructions and press 'Enter'

```
sudo apt-get install build-essential python-dev python-smbus git
```

- Install the drive library and program.

```
git clone https://github.com/DFRobotdl/111.git
```

- Find SEN0307.py in the file below, open and run it.

```

1 import time
2
3 from DFRobot_RaspberryPi_Expansion_Board import DFRobot_Expansion_Board_IIC as Board
4
5 board = Board(1, 0x18) # Select I2C Bus 1, set address to 0x18
6
7 def board_detect():
8     l = board.detect()
9     print("Board list conform:")
10    print(l)
11
12    ''' print last operate status, users can use this variable to determine the result of a function call. '''
13
14 def print_board_status():
15     if board.last_operate_status == board.STA_OK:
16         print("board status: everything ok")
17     elif board.last_operate_status == board.STA_ERR:
18         print("board status: unexpected error")
19     elif board.last_operate_status == board.STA_ERR_DEVICE_NOT_DETECTED:
20         print("board status: device not detected")
21     elif board.last_operate_status == board.STA_ERR_PARAMETER:
22         print("board status: parameter error")
23     elif board.last_operate_status == board.STA_ERR_SOFTWARE_VERSION:
24         print("board status: unsupported board firmware version")
25
26 if __name__ == "__main__":
27
28     board_detect() # If you forget address you had set, use this to detected them, you'll have class instance
29     # Set board controller address, use it carefully, report module to make it effective
30     '''
31     board.set_addr(0x18)
32
'''

```

- Get result.

```
Shell
Python 3.7.3 (/usr/bin/python3)
>>> !lsun SEN0307.py
Board list conform:
['0x10']
board begin success
distance: A0, value: 1999
distance: A0, value: 8
distance: A0, value: 8
distance: A0, value: 12
```

Lesson 17: Digital Touch Sensor

Preface

Introduction

This is a touch switch module based on capacitive sensing that can sense the direct contact of the human body or metal on its metal surface. In addition to the direct touch, the contact with a certain thickness of plastic, glass and other materials can also be sensed. And its sensitivity depends on the size of the contact surface and the thickness of the covering material.

Preparation

Lead-in

We are already clear about the control of Led. Now we are going to use Thonny Python IDE and the basic Python code operating GPIO to control Led by this digital analog sensors.

Hardware

- [Gravity: 37 Pcs Sensor Set](#)
- [Raspberry Pi 4 Model B](#)
- [IO Expansion HAT for Raspberry Pi 4B/3B+](#)
- [8GB + SanDisk Class10 SD/MicroSD Memory Card](#)
- [5V@3A USB Power Supply](#)
- [8.9 IPS Touch Display](#)

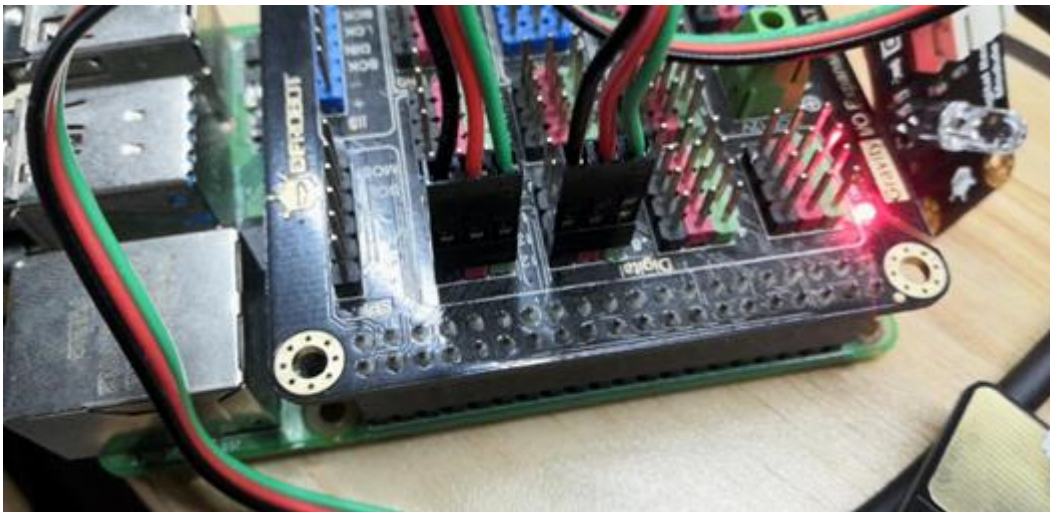
Learning Contents - Control LED with Digital Touch Sensor

Connection

- Connect the Raspberry Pi correctly to devices such as the screen, power, keyboard and mouse.



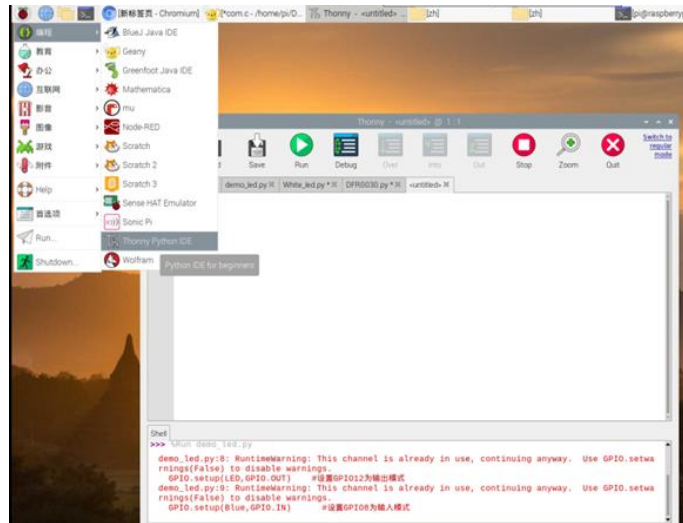
- Install the Raspberry Pi IO expansion board on the Raspberry Pi, connect the LED light-emitting module digital port 12 on the board, and the digital analog sensor to port 8, then turn on the Pi.



**If there is a metal object or a finger touches the metal piece, pin 8 inputs a high level and triggers a high level on the pin 12, and the LED is on. When there is no metal object or finger touch, the pin 12 is low and the LED is off. **

Programm

- Open Thonny Python IDE to copy the following program into it



```

import RPi.GPIO as GPIO    # Import the python module provided by the Raspberry Pi
import time                # Import time package for touch time detection

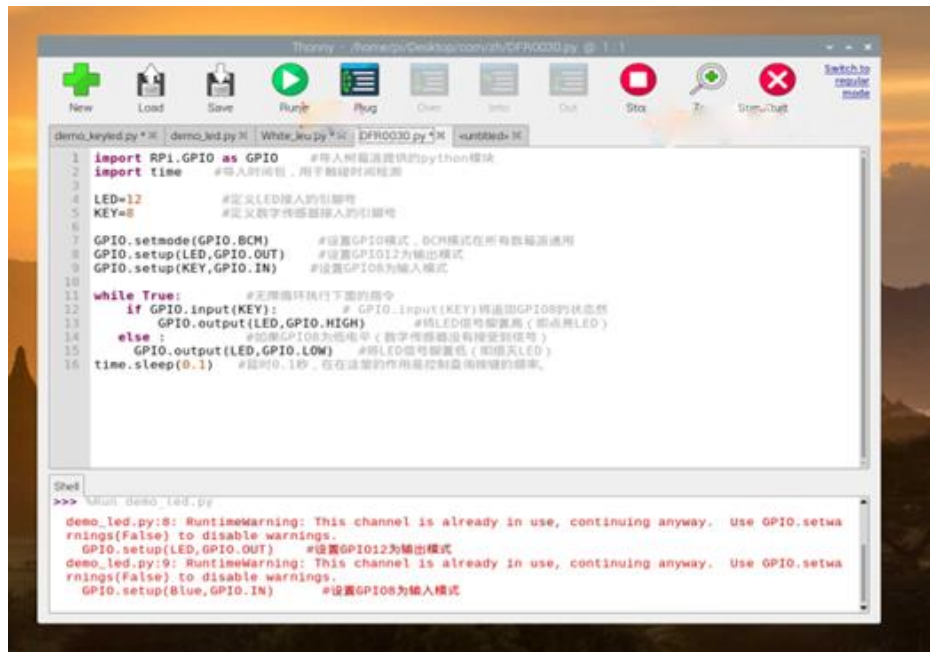
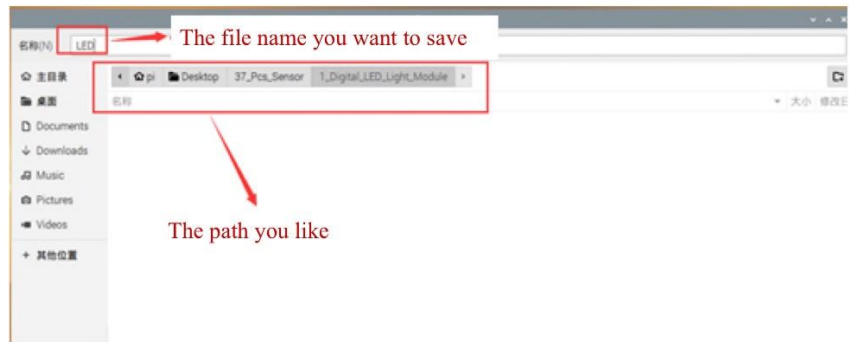
LED=12                    # Define the pin number to which the LED is connected
KEY=8                     # Define the pin number to which the sensor is connected

GPIO.setmode(GPIO.BCM)   # Set GPIO mode, BCM mode is common to all Raspberry Pi
GPIO.setup(LED,GPIO.OUT) # Set GPIO12 to output mode
GPIO.setup(KEY,GPIO.IN)  # Set GPIO8 to input mode

while True:              # Execute the following commands in an infinite loop
    if GPIO.input(KEY):  # GPIO.input(KEY) will return the state of GPIO and
        judge, if GPIO8 is high (i.e. The sensor received signal)
            GPIO.output(LED,GPIO.HIGH) #Set LED signal pin high (Light LED on)
    else :                # If GPIO8 is low (Not receive signal)
        GPIO.output(LED,GPIO.LOW)    #Set LED signal pin low (Turn LED off)
    time.sleep(0.1)      # Delay one second, here is to control the frequency of the query
    key

```





Lesson 18: Digital Steel Ball Inclination Sensor

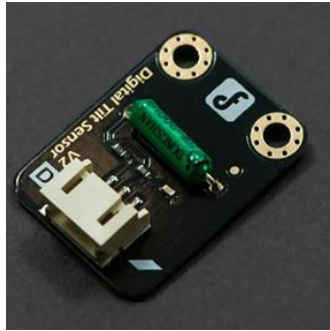
Preface

Introduction

This is a digital module based on a steel ball switch. It utilizes the characteristics of the steel ball to roll it towards the bottom through gravity, thereby closing or opening the switch. So it can also be used as a simple tilt sensor.

This module can also be used in combination with the Raspberry Pi sensor expansion board. In this case, it can be used to make very interesting interactive works, which is safer than using a mercury switch.

Operating Principle



It utilizes the characteristics of the steel ball to roll it towards the bottom through gravity, thereby closing or opening the switch.

Requirements

Hardware

- [Gravity: 37 Pcs Sensor Set](#)
- [Raspberry Pi 4 Model B](#)
- [IO Expansion HAT for Raspberry Pi 4B/3B+](#)
- [8GB + SanDisk Class10 SD/MicroSD Memory Card](#)
- [5V@3A USB Power Supply](#)
- [8.9 IPS Touch Display](#)

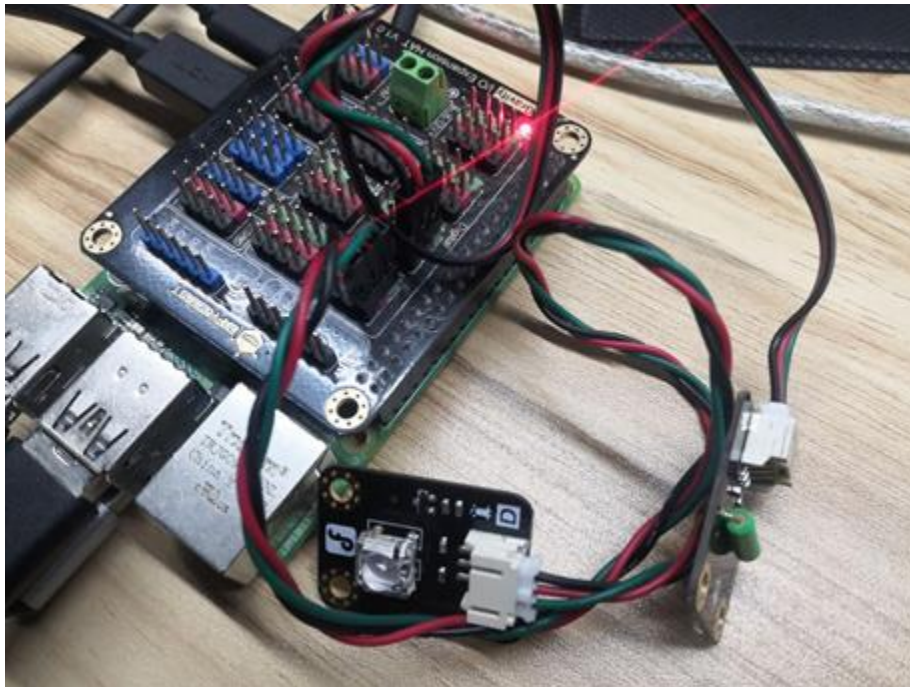
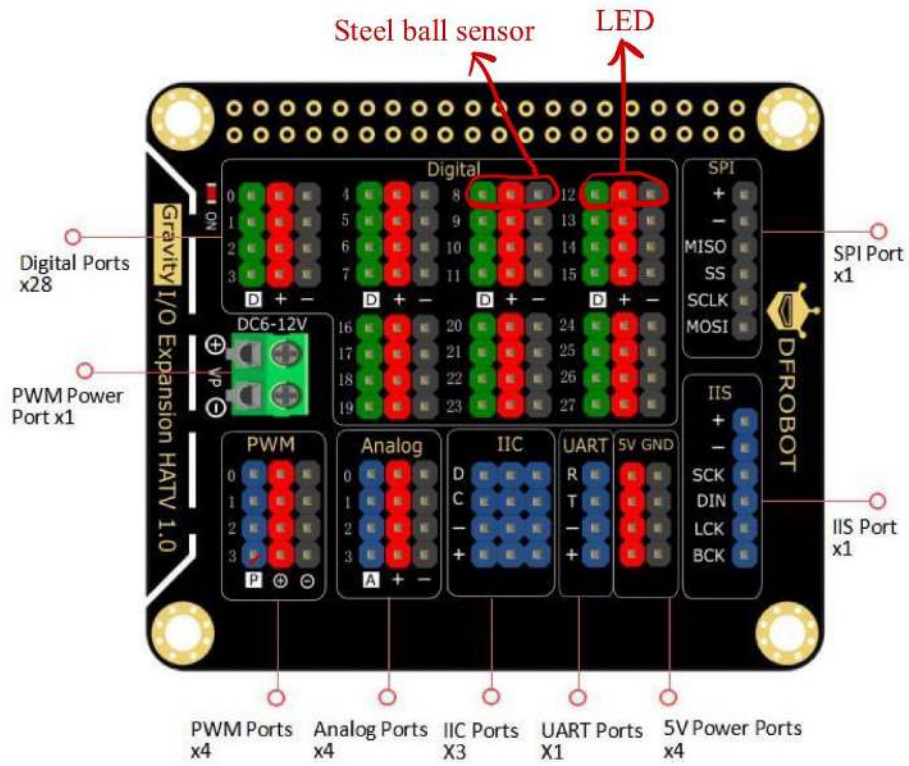
Learning Contents

Connection

- Connect the Raspberry Pi correctly to devices such as the screen, power, keyboard and mouse.

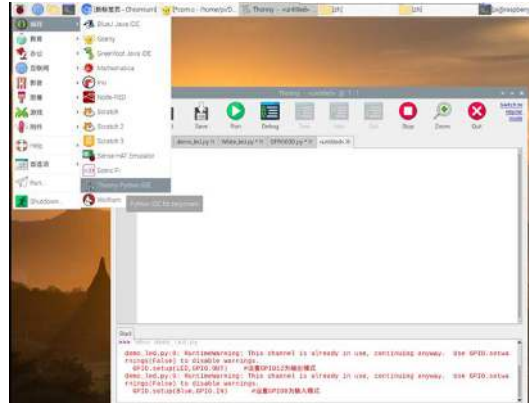


- Connect this sensor to pin 8 on the expansion board. For convenience, connect a led switch to pin 12.



Program

- Open Thonny Python IDE to copy the following program into it

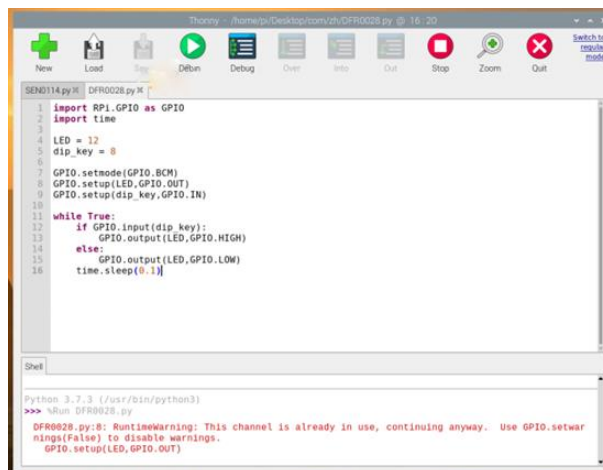


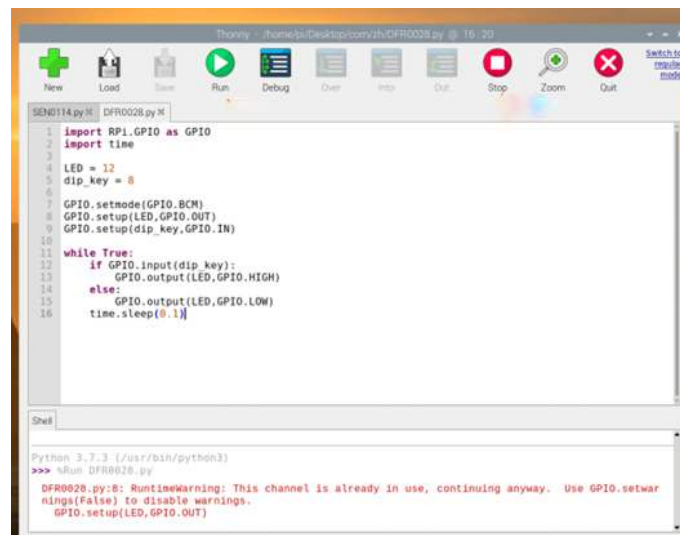
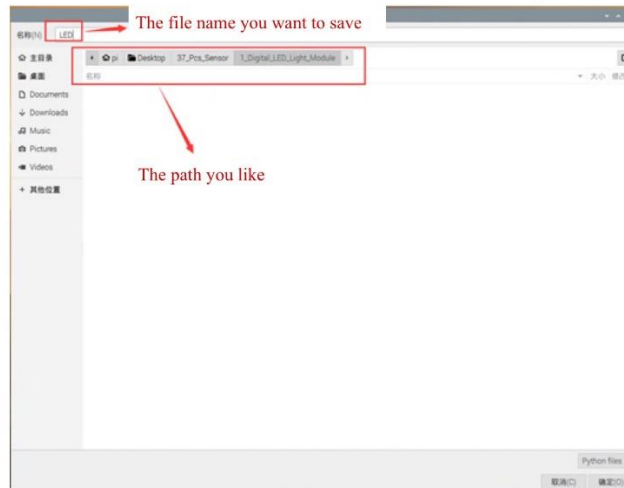
```
import RPi.GPIO as GPIO
import time

LED = 12
dip_key = 8

GPIO.setmode(GPIO.BCM)
GPIO.setup(LED,GPIO.OUT)
GPIO.setup(dip_key,GPIO.IN)

while True:
    if GPIO.input(dip_key):
        GPIO.output(LED,GPIO.HIGH)
    else:
        GPIO.output(LED,GPIO.LOW)
    time.sleep(0.1)
```





Lesson 19: Digital SMD Magnetic Induction Sensor

Preface

Introduction

This is a magnetic sensor based on high-quality reed tube that can sense the magnetic force within 3cm (the detection distance varies with the magnitude of the magnetic force). With our IO sensor expansion board V7, it can quickly build magnetic interaction projects.

The reed switch is disconnected in an environment without a magnetic field. When the magnetic force is strong enough, the reeds can be contacted and conducted. This process is very fast, making it a highly efficient and reliable switching element.

Requirements

Hardware

- [Gravity: 37 Pcs Sensor Set](#)
- [Raspberry Pi 4 Model B](#)
- [IO Expansion HAT for Raspberry Pi 4B/3B+](#)
- [8GB + SanDisk Class10 SD/MicroSD Memory Card](#)
- [5V@3A USB Power Supply](#)
- [8.9 IPS Touch Display](#)

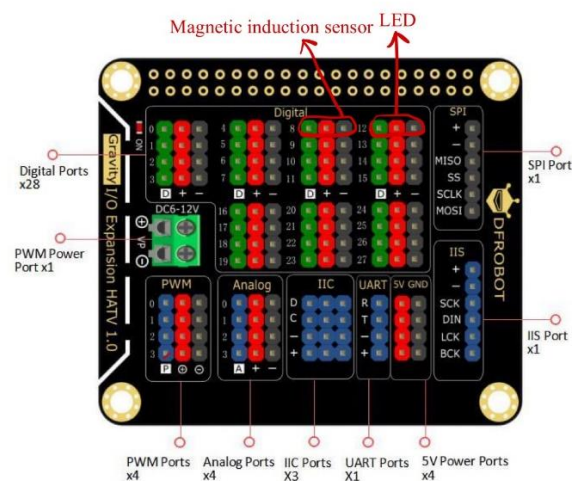
Learning Contents

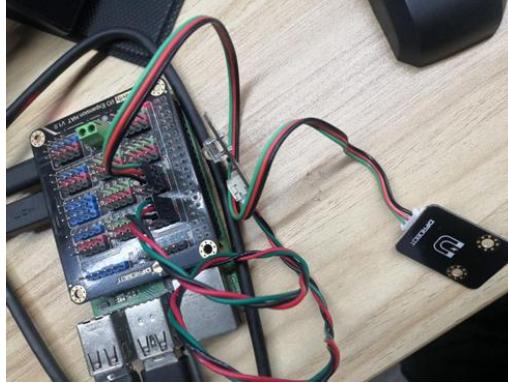
Connection

- Connect the Raspberry Pi correctly to devices such as the screen, power, keyboard and mouse.



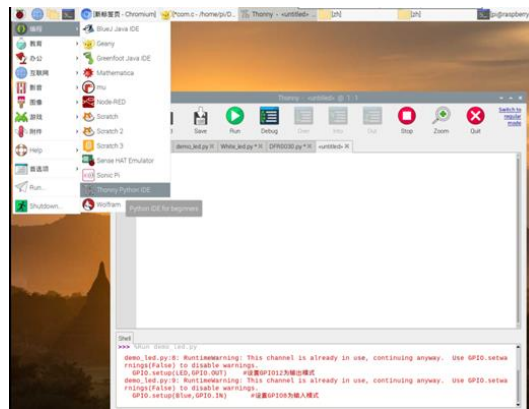
- Connect the LED to pin 12 on the expansion board, and the sensor to pin 8.





Program

- Open Thonny Python IDE to copy the following program into it



```
import RPi.GPIO as GPIO    #Import the python module provided by the Raspberry Pi
import time                #Import time package to detect induction time

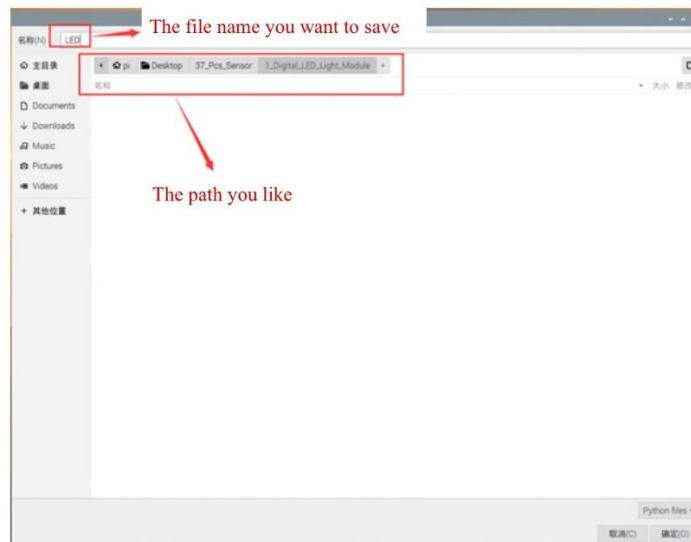
LED = 12                   #Define the pin number to which the LED is connected
magnetic_key = 8           #Define the pin number to which the sensor is connected

GPIO.setmode(GPIO.BCM)    #Set GPIO mode, BCM mode is common to all Raspberry Pi
GPIO.setup(LED,GPIO.OUT)  #Set GPIO12 to output mode
GPIO.setup(magnetic_key ,GPIO.IN)    #设Set GPIO8 to input mode

while True:               #Execute the following commands in an infinite loop
    if GPIO.input(magnetic_key ):           #GPIO.input(magnetic_key )
        GPIO.output(LED,GPIO.HIGH)        #Set the LED signal high (Turn off LED)
    else :                                #If GPIO8 is low (The sensor didn't receive signal)
        GPIO.output(LED,GPIO.LOW)         #Set LED signal low (Turn on LED)
    time.sleep(0.1)    #Delay one second, here is to control the frequency of the query key
```



```
1 import RPi.GPIO as GPIO #导入树莓派提供的python模块
2 import time #导入时间包,用于感应时间检测
3
4 LED = 12 #定义LED接入的引脚号
5 magnetic_key = 8 #定义数字传感器接入的引脚号
6
7 GPIO.setmode(GPIO.BCM) #设置GPIO模式,BCM模式在所有数据源通用
8 GPIO.setup(LED,GPIO.OUT) #设置GPIO12为输出模式
9 GPIO.setup(magnetic_key,GPIO.IN) #设置GPIO8为输入模式
10
11 while True: #无限循环执行下面的指令
12     if GPIO.input(magnetic_key): #GPIO.input(magnetic_key)
13         GPIO.output(LED,GPIO.HIGH) #将LED信号脚置高(点亮LED)
14     else: #如果GPIO8为低电平(数字磁感应没有接收到信号)
15         GPIO.output(LED,GPIO.LOW) #将LED信号脚置低(熄灭LED)
16     time.sleep(0.1) #延时0.1秒,在这里的作用是控制查询磁感应的频率。
```



```
Python 3.7.3 (/usr/bin/python3)
>>> %Run DFR0033.py
DFR0033.py:8: RuntimeWarning: This channel is already in use, continuing anyway. Use GPIO.setwarnings(False) to disable warnings.
  GPIO.setup(LED,GPIO.OUT) #设置GPIO12为输出模式
```

Lesson 20: Mini Vibration Module

Preface

Lead-in

We have already introduced the digital push button module. Now let's use the module and the miniature vibration module to control its vibration to make a simple project.

Introduction

The vibration module uses a vibration motor as the excitation source. The motor is equipped with a set of adjustable eccentric blocks at one end of the rotor shaft, and the excitation force is obtained by the centrifugal force generated by the high-speed rotation of the shaft and the eccentric block.

Requirements

Hardware

- [Gravity: 37 Pcs Sensor Set](#)
- [Raspberry Pi 4 Model B](#)
- [IO Expansion HAT for Raspberry Pi 4B/3B+](#)
- [8GB + SanDisk Class10 SD/MicroSD Memory Card](#)
- [5V@3A USB Power Supply](#)
- [8.9 IPS Touch Display](#)

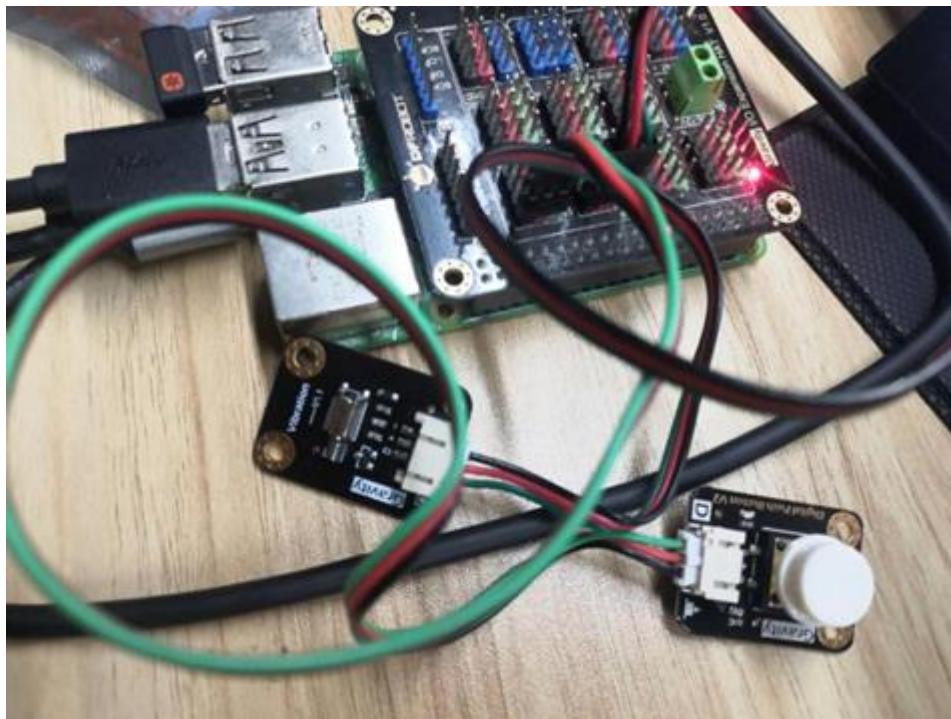
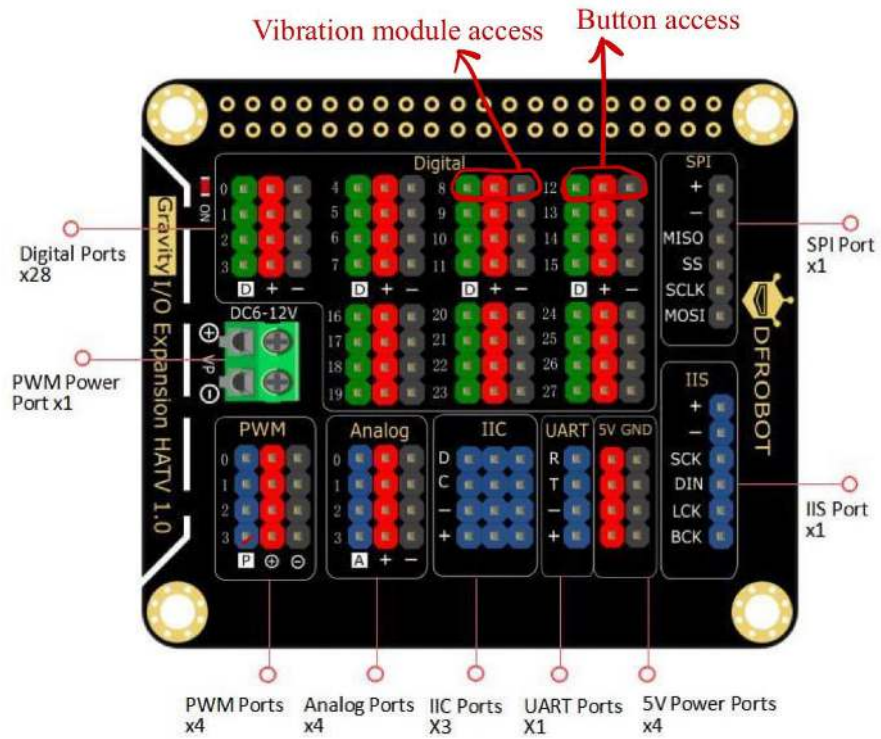
Learning Contents

Connection

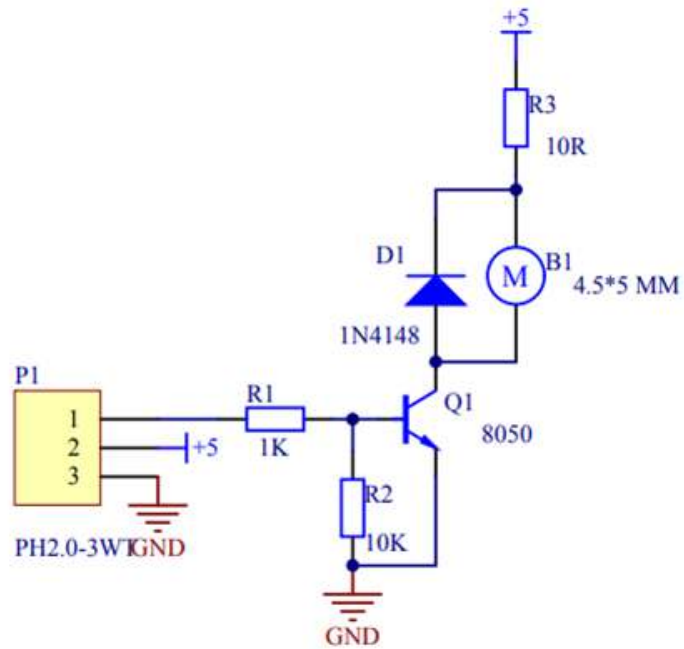
- Connect the Raspberry Pi correctly to devices such as the screen, power, keyboard and mouse.



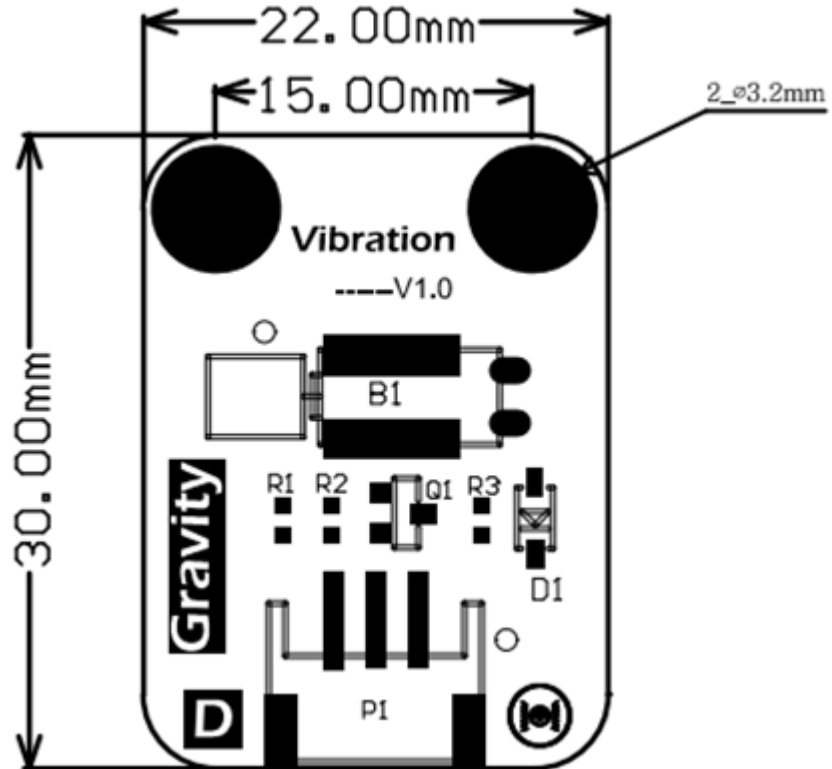
- Connect the IO expansion board with the Raspberry Pi. Connect the push button module to digital port 18, the vibration module to digital port 8.



Schematic



Original Layout



Program

- Open Thonny Python IDE to copy the following program into it



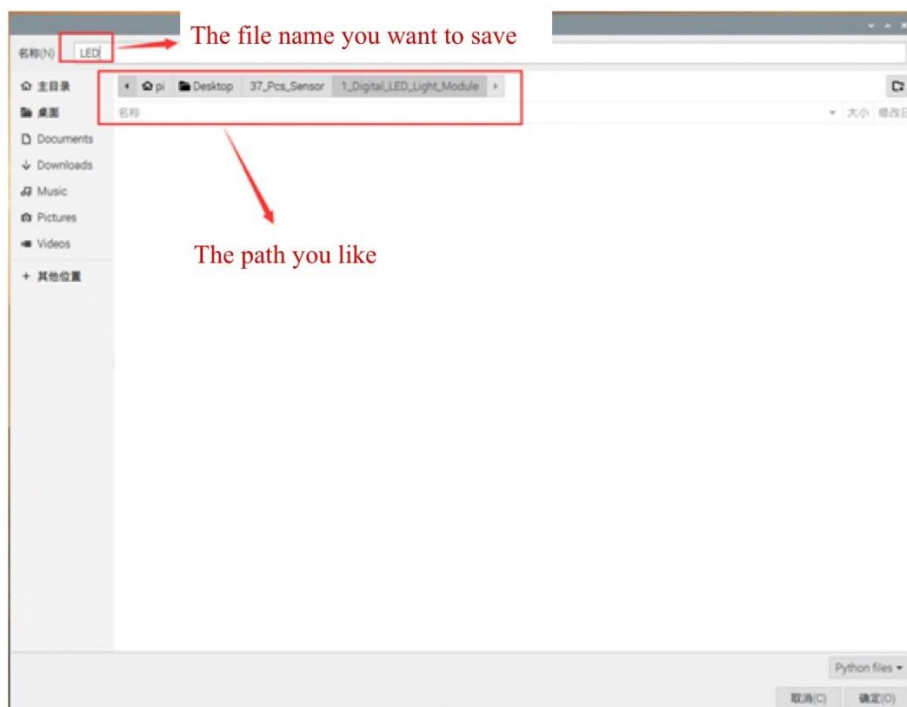
```
import RPi.GPIO as GPIO # Import the python module provided
by the Raspberry Pi
import time #Import time package to control
flicker

button = 12 #Define the pin number to which the
button is connected
vibration = 8 #Define the pin number to which the
vibration is connected

GPIO.setmode(GPIO.BCM) #Set GPIO mode, BCM mode is common to all
Raspberry Pi
GPIO.setup(button,GPIO.IN) #Set GPIO12 to input mode
GPIO.setup(vibration,GPIO.OUT) # Set GPIO8 to output mode
while True: # Set GPIO8 to input mode
    key = GPIO.input(button)
    if (key): #Judge whether the button is pressed
        GPIO.output(vibration,GPIO.HIGH) #Button pressed, start the micro
        vibrator
        time.sleep(5) #Wait for 5 s
        GPIO.output(vibration,GPIO.LOW) #Turn off the micro vibrator
    else : #If GPIO8 is low(that is, the button
is released), execute the following statement
        GPIO.output(vibration,GPIO.LOW) #Not start the micro vibrator
        time.sleep(0.1) #Delay one second, here is to contro
l the frequency of the query key
```

The screenshot shows the Thonny IDE interface. The top toolbar includes icons for New, Load, Save, Run, Debug, Over, Info, Out, Stop, Zoom, and Quit. The main window displays a Python script named 'DFR0440.py' with the following code and comments:

```
1 import RPi.GPIO as GPIO #导入树莓派提供的python模块
2 import time #导入时间包，用于控制闪烁
3
4 button = 12 #定义按钮接入引脚
5 vibration = 8 #定义振动器接入引脚
6
7 GPIO.setmode(GPIO.BCM) #设置GPIO模式，BCM模式在所有树莓派通用
8 GPIO.setup(button,GPIO.IN) #设置GPIO12为输入模式
9 GPIO.setup(vibration,GPIO.OUT) #设置GPIO8为输出模式
10
11 while True: #无限循环执行下面的指令
12     key = GPIO.input(button) #判断按钮是否按下
13     if (key ): #按钮按下，启动微型振动器
14         GPIO.output(vibration,GPIO.HIGH) #等待5s
15         time.sleep(5) #关闭微型振动器
16         GPIO.output(vibration,GPIO.LOW) #如果GPIO8为低电平（即按钮没按下）就执行下面的语句
17     else : #未启动微型振动器
18         GPIO.output(vibration,GPIO.LOW) #等待0.1秒，在这里的作用是控制查询按钮的频率。|
19 time.sleep(0.1)
```





```
DFR0440.py X
1 import RPi.GPIO as GPIO #导入树莓派提供的python模块
2 import time #导入时间包，用于控制闪烁
3
4 button = 12 #定义按钮接入引脚
5 vibration = 8 #定义振动器接入引脚
6
7 GPIO.setmode(GPIO.BCM) #设置GPIO模式，BCM模式在所有树莓派通用
8 GPIO.setup(button,GPIO.IN) #设置GPIO12为输入模式
9 GPIO.setup(vibration,GPIO.OUT) #设置GPIO8为输出模式
10
11 while True: #无限循环执行下面的命令
12     key = GPIO.input(button)
13     if (key ): #判断按钮是否按下
14         GPIO.output(vibration,GPIO.HIGH) #按钮按下，启动微型振动器
15         time.sleep(5) #等待5s
16         GPIO.output(vibration,GPIO.LOW) #关闭微型振动器
17     else : #如果GPIO8为低电平（即按钮没按下）就执行下面的语句
18         GPIO.output(vibration,GPIO.LOW) #关闭微型振动器
19     time.sleep(0.1) #等待0.1秒，在这里的作用是控制查询按钮的速率。|
```

- Click to [watch the effect](#).

FAQ

For any questions, advice or cool ideas to share, please visit the [DFRobot Forum](#).