Preface

About SunFounder

SunFounder is a technology company focused on Raspberry Pi and Arduino open source community development. Committed to the promotion of open source culture, we strive to bring the fun of electronics making to people all around the world and enable everyone to be a maker. Our products include learning kits, development boards, robots, sensor modules and development tools. In addition to high quality products, SunFounder also offers video tutorials to help you build your own project. If you have interest in open source or making something cool, welcome to join us! Visit www.sunfounder.com for more!

About This Kit

This sensor kit is suitable for the Raspberry Pi model B+, 2 model B, and 3 model B. It includes dozens of different modules for you to learn and we provide corresponding lessons which are simple and useful for better understanding. Hope you can learn their applications quickly and use them in your own projects!

In this book, we will show you circuits with both realistic illustrations and schematic diagrams. You can go to our official website www.sunfounder.com to download related code by clicking **LEARN** -> **Get Tutorials**.

Free Support

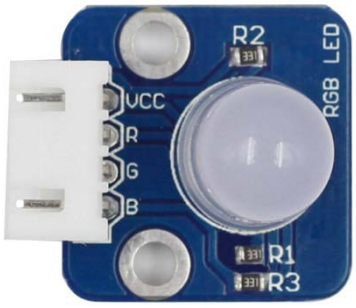
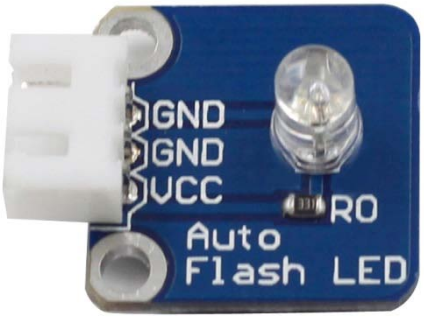
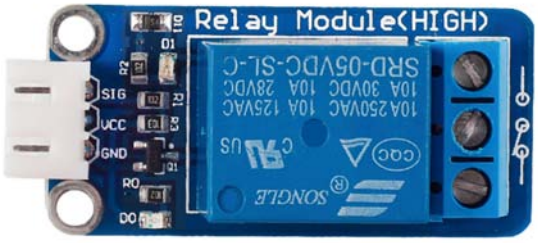
If you have any **TECHNICAL questions**, add a topic under **FORUM** section on our website and we'll reply as soon as possible.

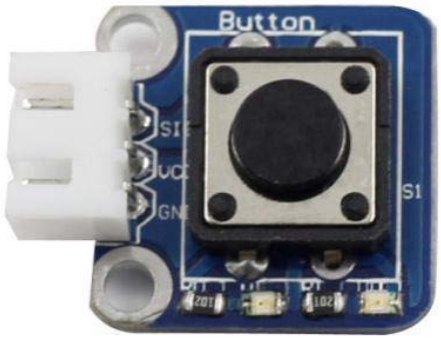
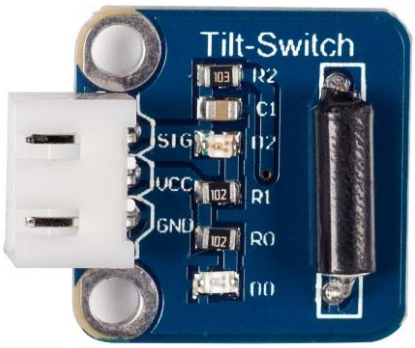
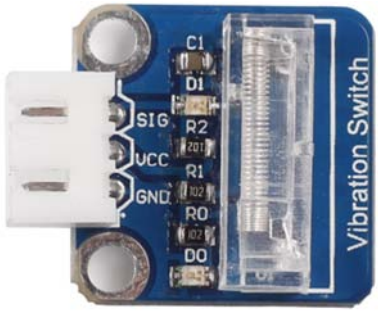
For **NON-TECH questions** like order and shipment issues, please **send an email to service@sunfounder.com**. You're also welcomed to share your projects on FORUM.

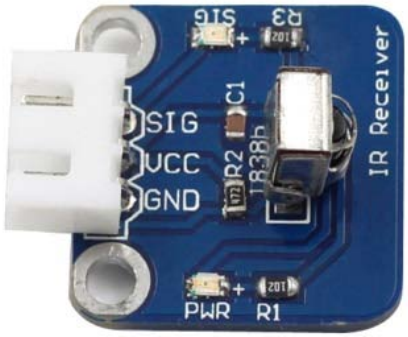
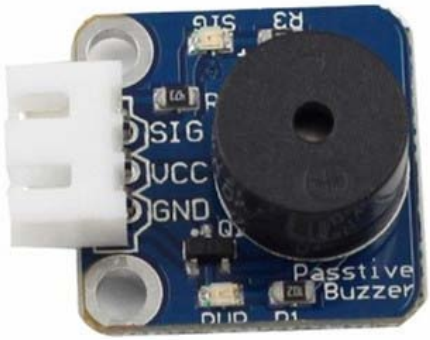
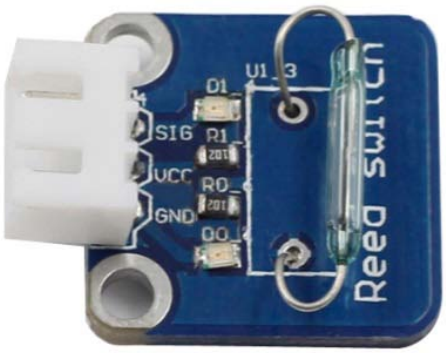
Reprint 2.0

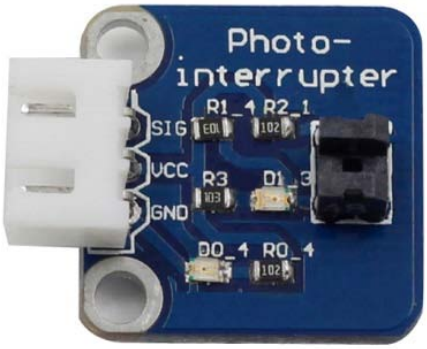
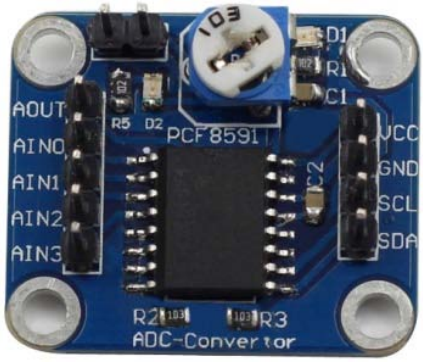
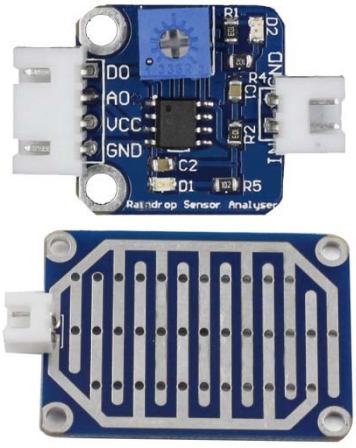
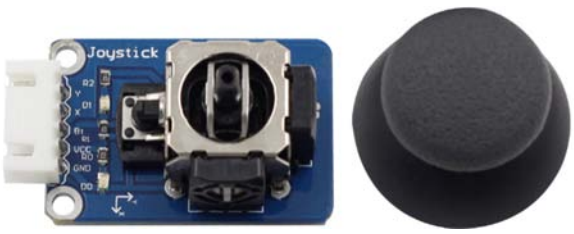
Contents

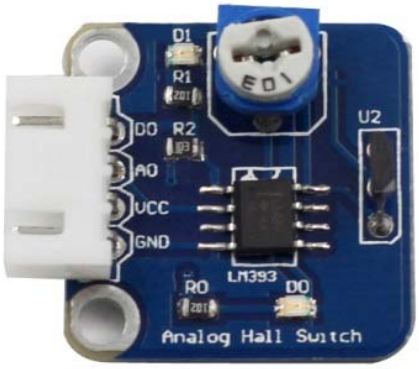
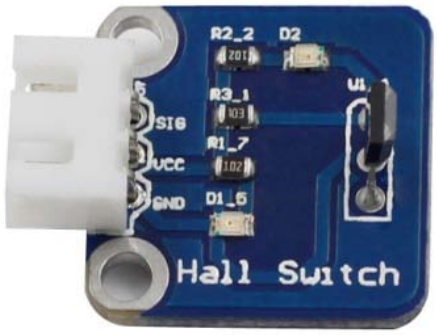
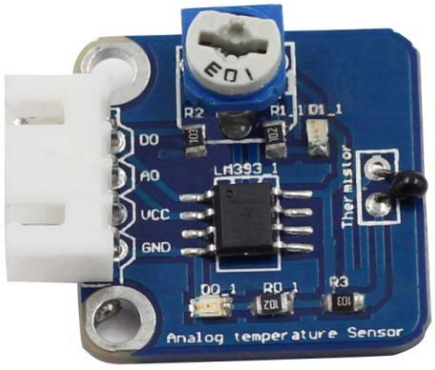
Components List

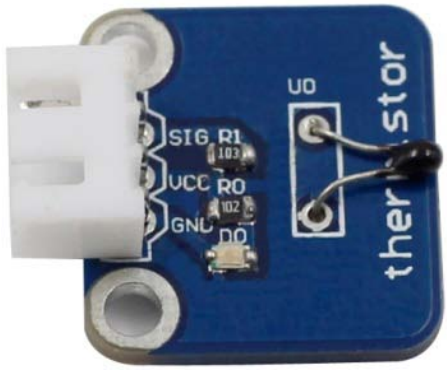
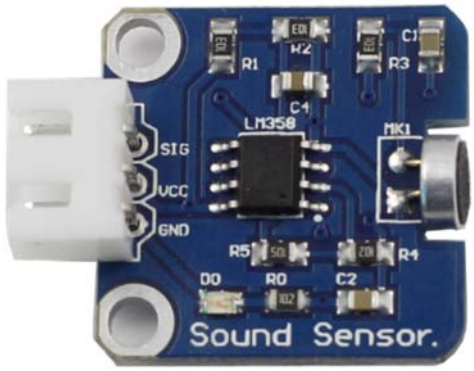
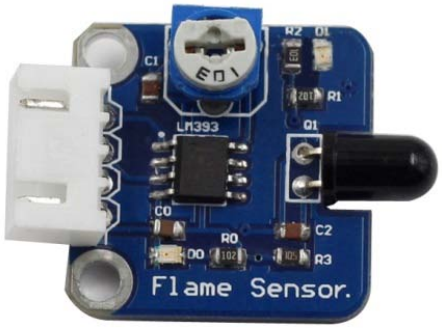
No.	Name	Qty.	Component
1	Dual-Color LED	1	
2	RGB LED	1	
3	Auto Flash LED	1	
4	Relay Module	1	

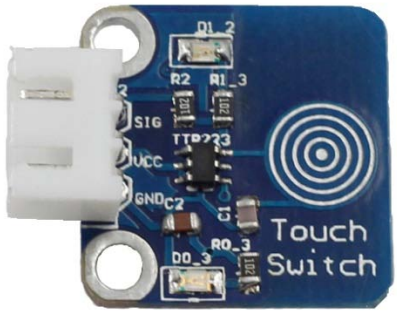
5	Laser Emitter	1	
6	Button	1	
7	Tilt-Switch	1	
8	Vibration Switch	1	

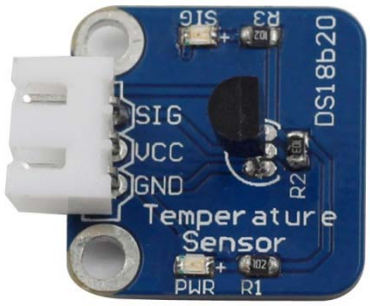
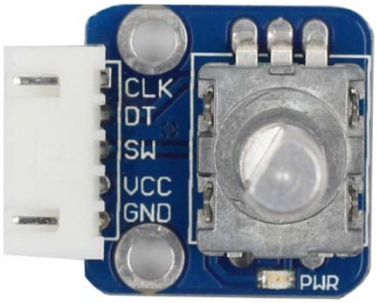
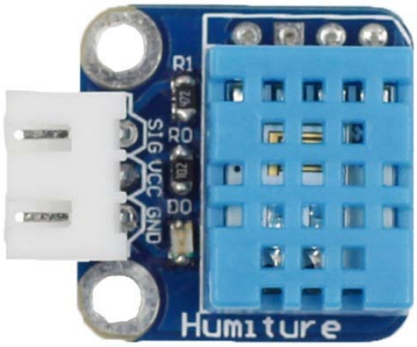
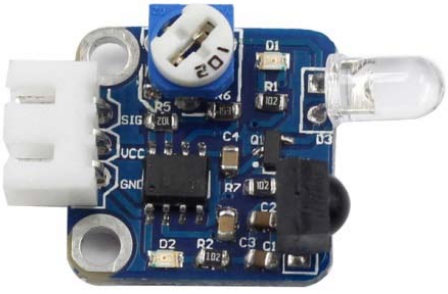
9	IR Receiver	1	
10	Active Buzzer	1	
11	Passive Buzzer	1	
12	Reed Switch	1	

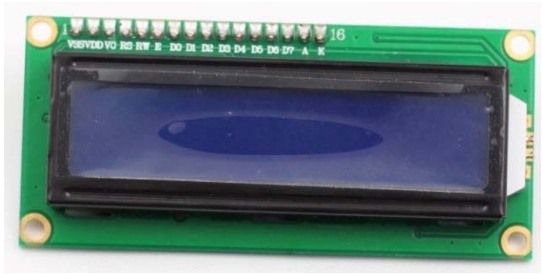
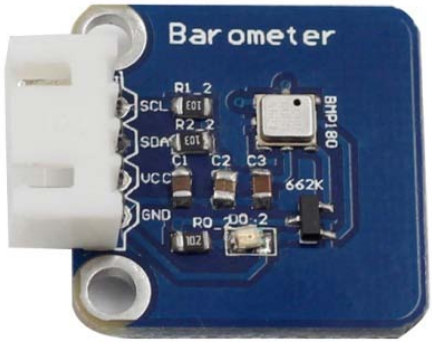
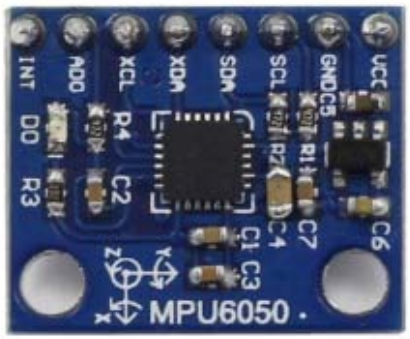
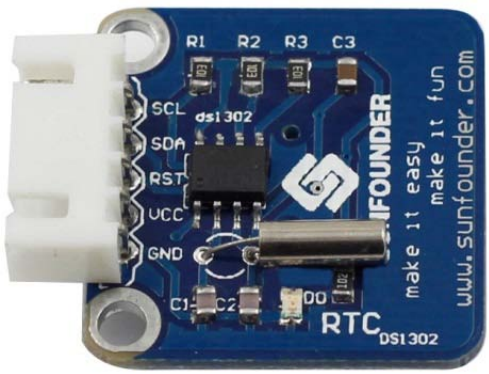
13	Photo-interrupter	1	
14	AD/DA Converter PCF8591	1	
15	Raindrop Sensor	1	
16	Joystick PS2	1	

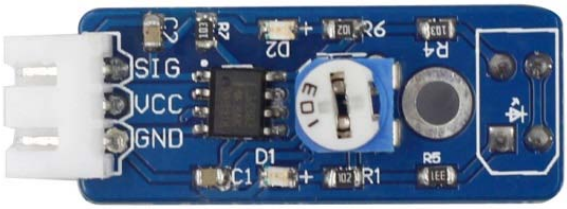
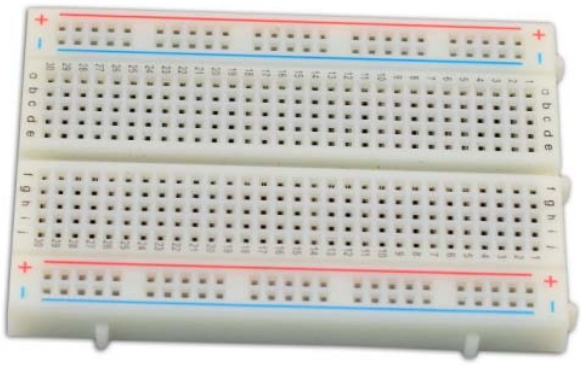
17	Potentiometer	1	
18	Analog Hall Sensor	1	
19	Hall Switch Sensor	1	
20	Analog Temperature Sensor	1	

21	Thermistor	1	
22	Sound Sensor	1	
23	Photoresistor	1	
24	Flame Sensor	1	

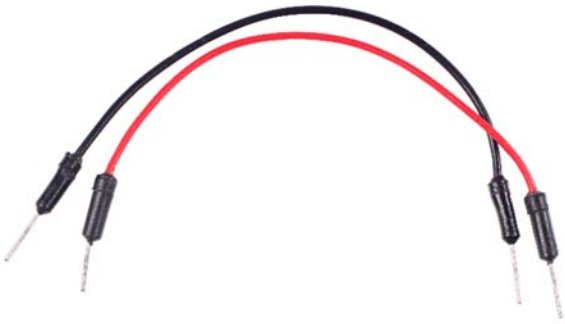
25	Gas Sensor	1	
26	Remote Control	1	
27	Touch Switch	1	
28	Ultrasonic	1	

29	Temperature Sensor DS18B20	1	
30	Rotary Encoder	1	
31	Humiture Sensor	1	
32	IR Obstacle Module	1	

33	I2C LCD1602 Module	1	
34	Barometer-BMP180	1	
35	MPU6050 Module	1	
36	RTC-DS1302 Module	1	

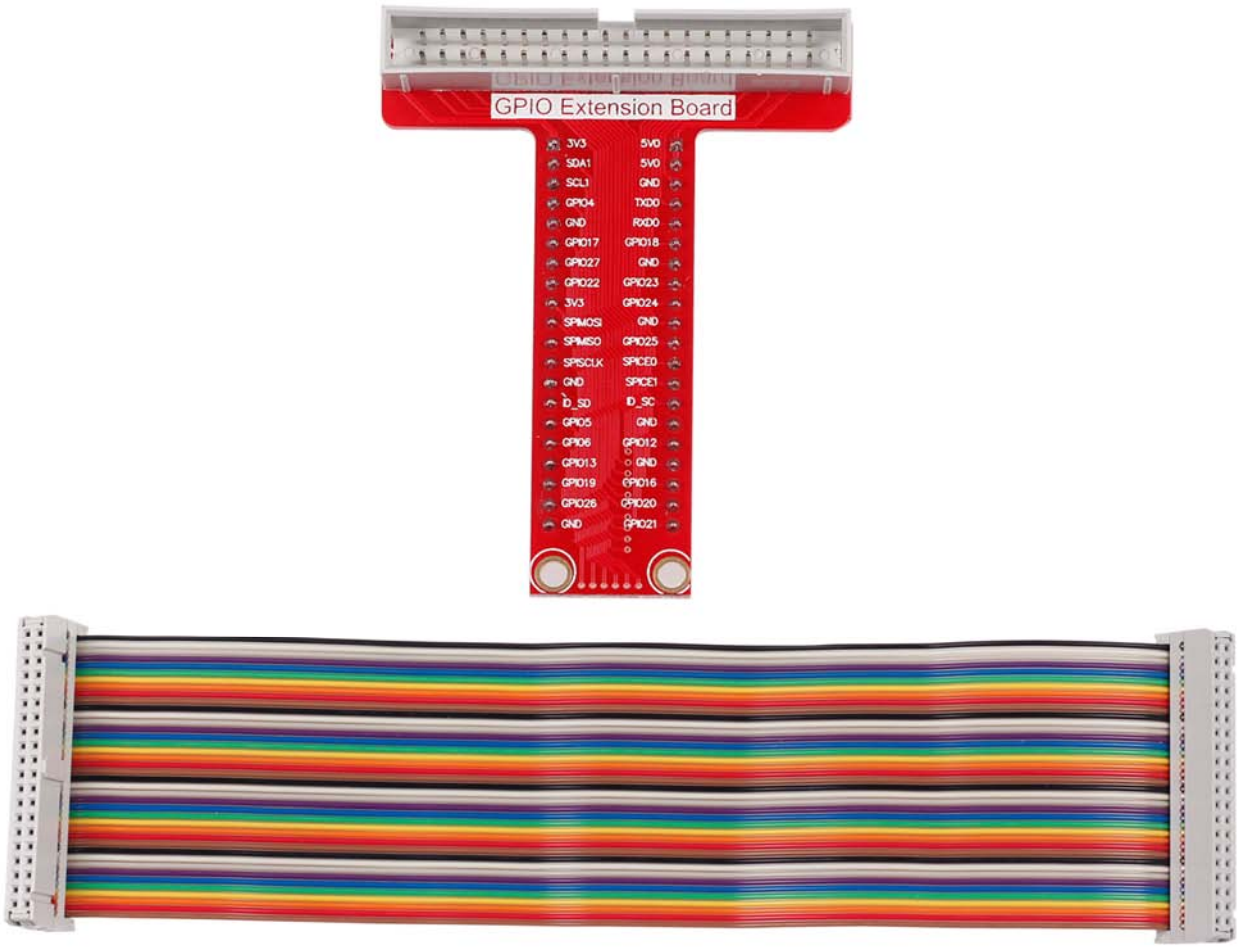
37	Tracking Sensor	1	
38	Breadboard	1	
39	T-Cobbler	1	
40	40-pin Ribbon Cable for T-Cobbler	1	

41	2-Pin Anti-reverse Cable	2	
42	3-Pin Anti-reverse Cable	5	
43	4-Pin Anti-reverse Cable	5	
44	5-Pin Anti-reverse Cable	5	

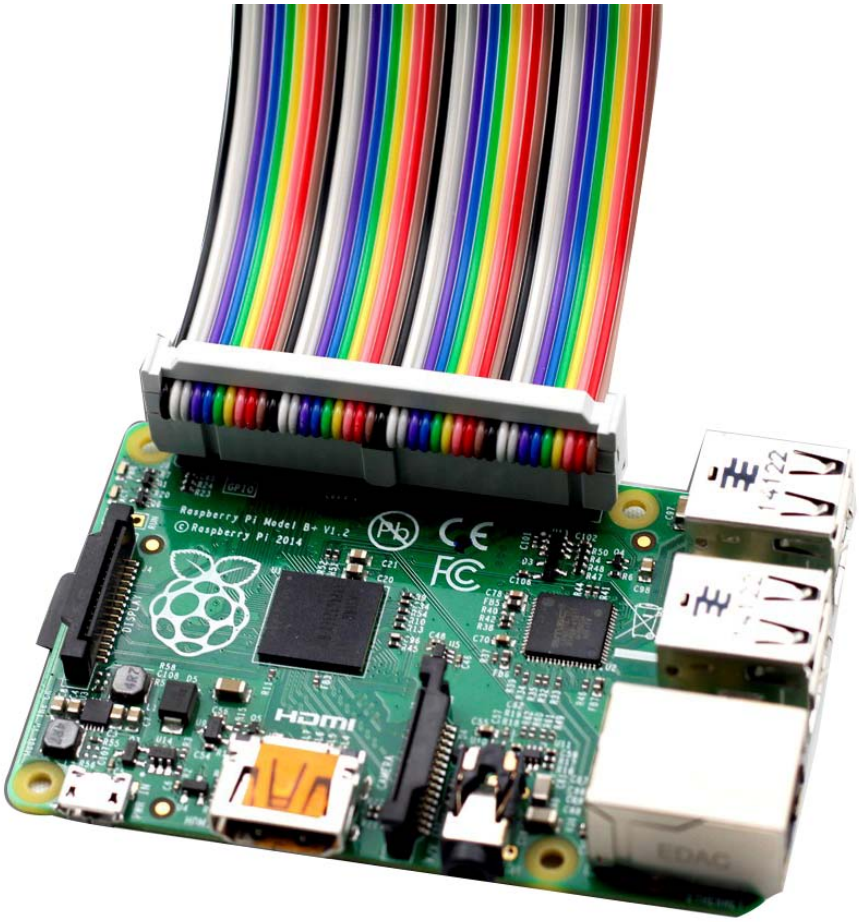
45	2-Pin Ribbon Cable (Female to Female)	1	
46	Jumper wires (Male to Female)	20	
47	Jumper wires (Male to Male)	10	

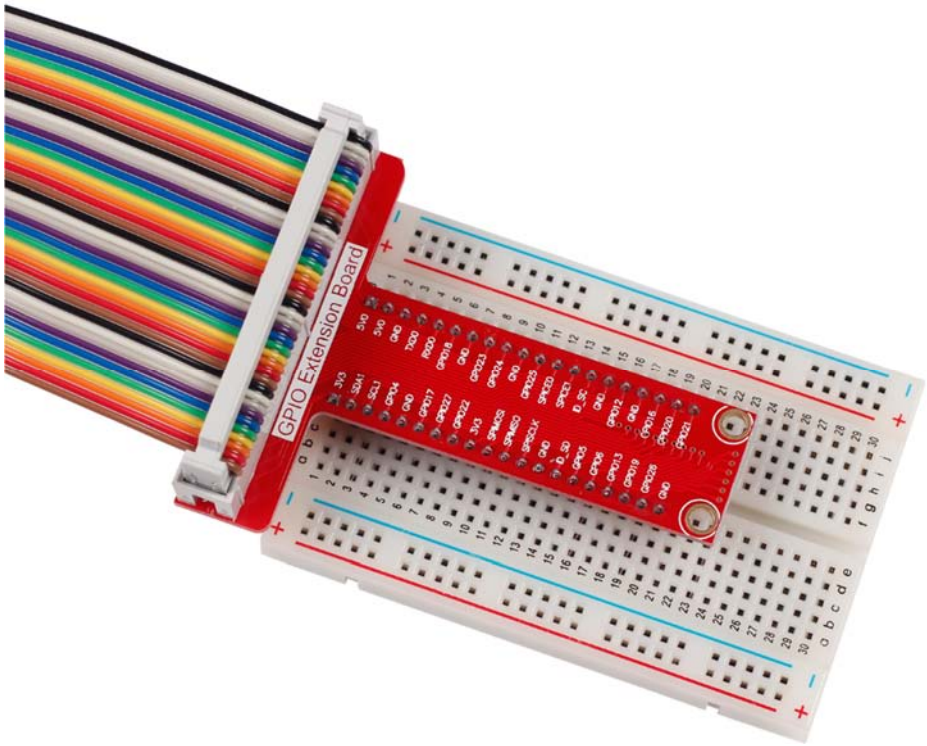
Notice

We can easily lead out pins of the Raspberry Pi to breadboard by GPIO Extension Board to avoid GPIO damage caused by frequent plugging in or out. This is our 40-pin GPIO Extension Board and GPIO cable for Raspberry Pi model B+ and Raspberry Pi 2 model B.

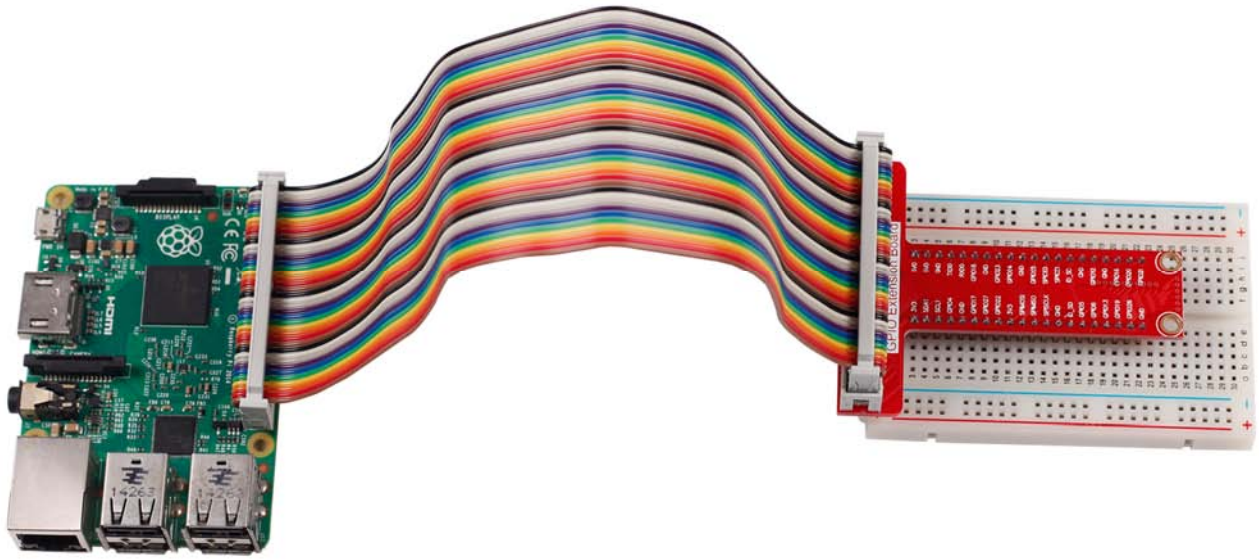


Connect the GPIO cable to the Raspberry Pi B+ like this:



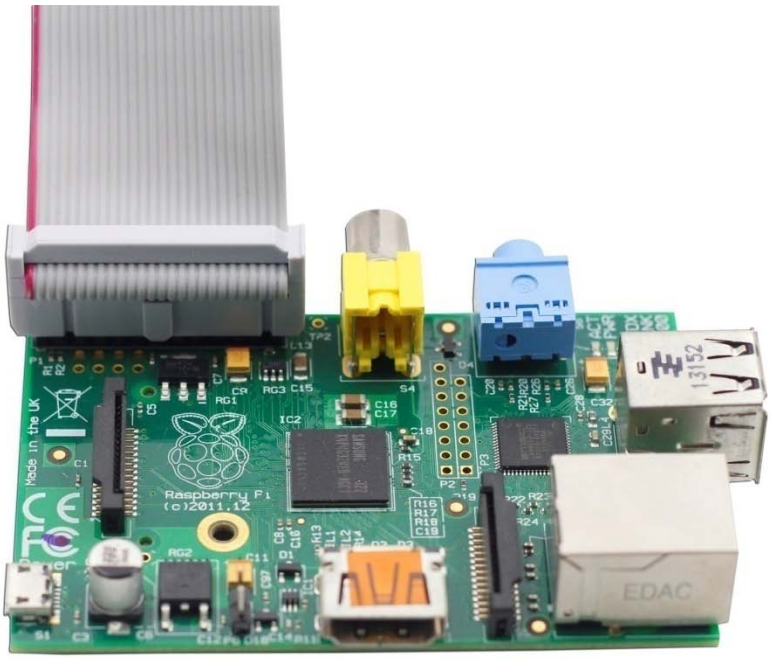


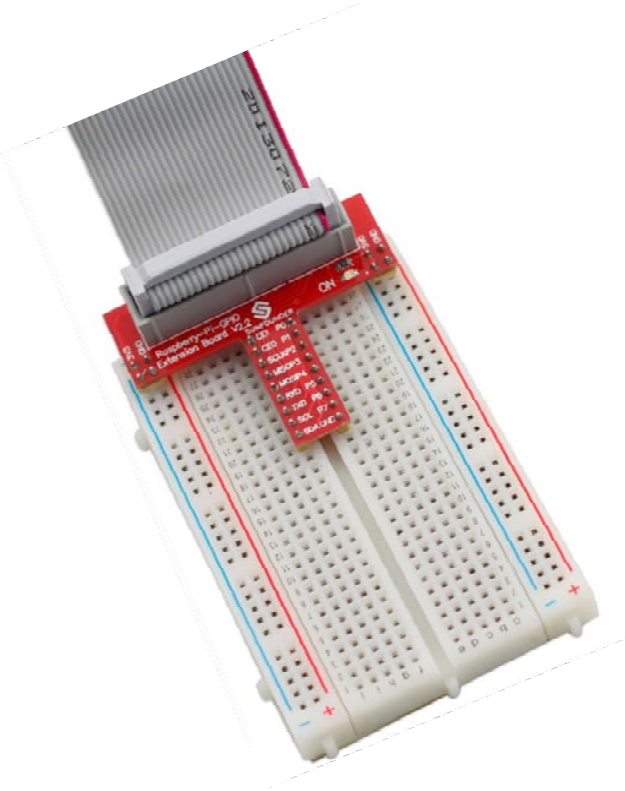
After connection, it is shown as follows:



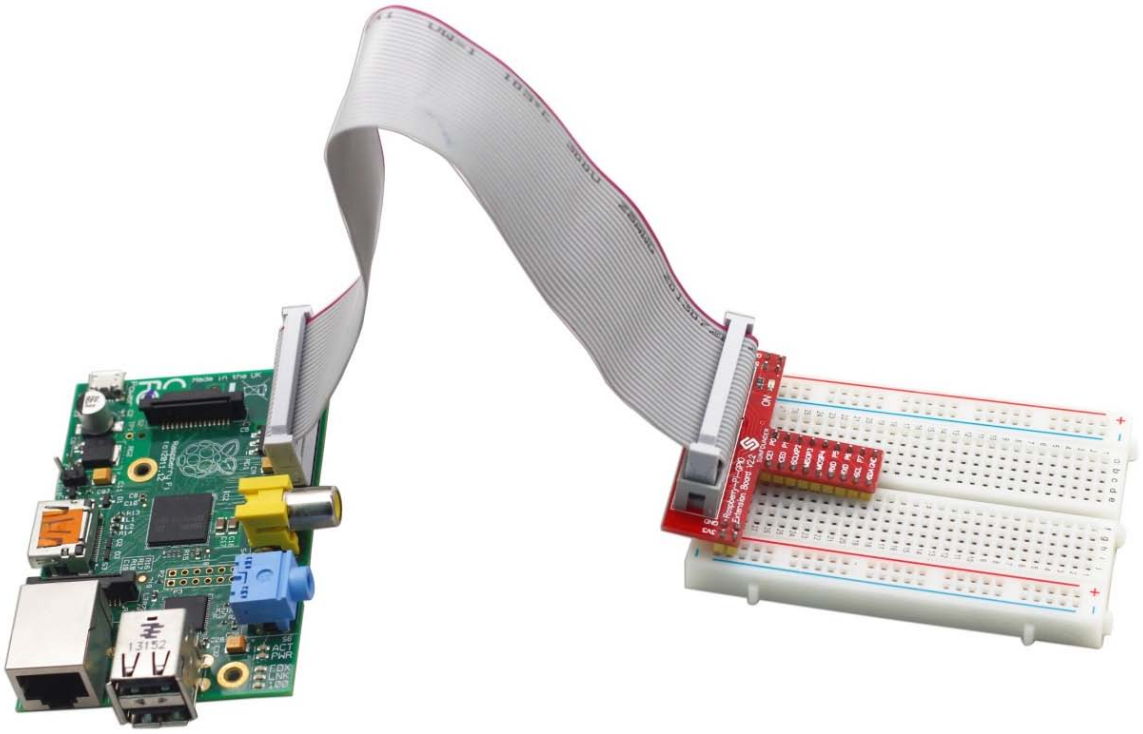
This is our 26-pin GPIO Extension Board and GPIO cable for Raspberry Pi B.

Connect the GPIO cable to the Raspberry Pi B like this:





After connection, it is shown as follows:



Introduction to Pin Number of Raspberry Pi

wiringPi Pin	BCM GPIO	Name	Header	Name	BCM GPIO	wiringPi Pin
–	–	3.3v	1 | 2	5v	–	–
8	R1:0/R2:2	SDA0	3 | 4	5v	–	–
9	R1:1/R2:3	SCL0	5 | 6	0V	–	–
7	4	GPIO7	7 | 8	TXD	14	15
–	–	0V	9 | 10	RXD	15	16
0	17	GPIO0	11 | 12	GPIO1	18	1
2	R1:21/R2:27	GPIO2	13 | 14	0V	–	–
3	22	GPIO3	15 | 16	GPIO4	23	4
–	–	3.3v	17 | 18	GPIO5	24	5
12	10	MOSI	19 | 20	0V	–	–
13	9	MISO	21 | 22	GPIO6	25	6
14	11	SCLK	23 | 24	CE0	8	10
–	–	0V	25 | 26	CE1	7	11
30	0	SDA. 0	27 | 28	SCL. 0	1	31
21	5	GPIO. 21	29 | 30	0V	–	–
22	6	GPIO. 22	31 | 32	GPIO. 26	12	26
23	13	GPIO. 23	33 | 34	0V	–	–
24	19	GPIO. 24	35 | 36	GPIO. 27	16	27
25	26	GPIO. 25	37 | 38	GPIO. 28	20	28
		0V	39 | 40	GPIO. 29	21	29
wiringPi Pin	BCM GPIO	Name	Header	Name	BCM GPIO	wiringPi Pin

For RPi B

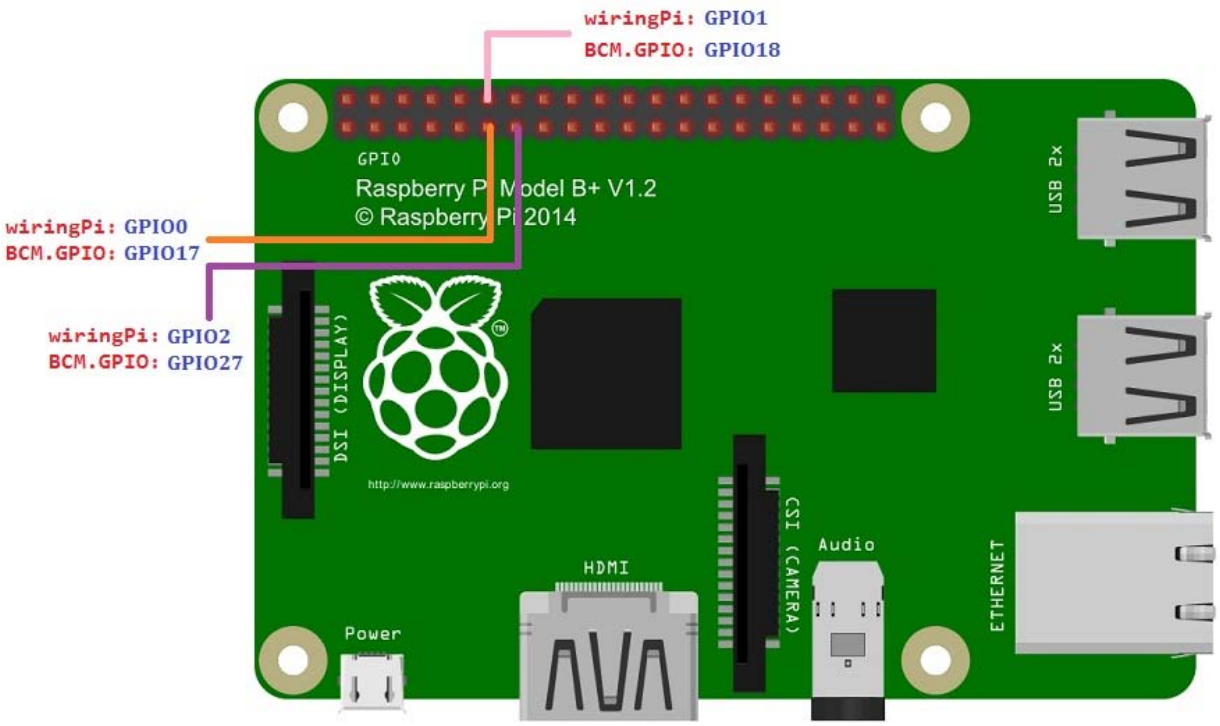
For RPi B+/ 2 model B

Currently, there are three methods of pin numbering for Raspberry Pi.

1. Numbering based on the physical location of pins
2. Numbering appointed by C language GPIO library wiringPi
3. Numbering appointed by BCM2835 SOC

If we want to operate Raspberry Pi GPIOs in C language based on the wiringPi library, choose numbering appointed by wiringPi. You can see from the above diagram that GPIO0 in wiringPi corresponds to pin 11 numbered by physical location, and GPIO30, to pin 27.

The picture below demonstrates the physical numbers of the three pins 11, 12, and 13 in detail:



If you are a C user, please install wiringPi.

WiringPi introduction:

WiringPi is a GPIO access library written in C for the BCM2835 used in the Raspberry Pi. It's released under the GNU LGPLv3 license and is usable from C and C++ and many other languages with suitable wrappers. It's designed to be familiar to people who have used the Arduino "wiring" system.

How to install

Step 1: Get the source code of wiringPi

```
git clone git://git.drogon.net/wiringPi
```

Step 2: Install wiringPi

```
cd wiringPi
git pull origin
./build
```

Press Enter, with the script *build*, the source code of wiringPi will be compiled automatically and installed to the appropriate directory of Raspberry Pi OS.

Step 3: Test whether wiringPi is installed successfully or not

WiringPi includes many GPIO commands which enable you to control all kinds of interfaces on Raspberry Pi. You can test whether the wiringPi library is installed successfully or not by the following instructions.

```
gpio -v
```

```
root@raspberrypi:/home/wiringPi# gpio -v
gpio version: 2.21
Copyright (c) 2012-2014 Gordon Henderson
This is free software with ABSOLUTELY NO WARRANTY.
For details type: gpio -warranty

Raspberry Pi Details:
  Type: Model B, Revision: 2, Memory: 512MB, Maker: Sony
```

If the message above appears, the wiringPi is installed successfully.

```
$gpio readall
```

```
root@raspberrypi:/home/wiringPi# gpio readall
 +-----+-----+---------+------+---+-Model B2-+---+------+---------+-----+-----+
 | BCM | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | BCM |
 +-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
 |     |     |    3.3v |      |   |  1 || 2  |   |      | 5v      |     |     |
 |   2 |   8 |   SDA.1 | ALT0 | 1 |  3 || 4  |   |      | 5V      |     |     |
 |   3 |   9 |   SCL.1 | ALT0 | 1 |  5 || 6  |   |      | 0v      |     |     |
 |   4 |   7 | GPIO. 7 |   IN | 1 |  7 || 8  | 1 | ALT0 | TxD     | 15  | 14  |
 |     |     |      0v |      |   |  9 || 10 | 1 | ALT0 | RxD     | 16  | 15  |
 |  17 |   0 | GPIO. 0 |   IN | 1 | 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 | ALT0 | 0 | 19 || 20 |   |      | 0v      |     |     |
 |   9 |  13 |    MISO | ALT0 | 0 | 21 || 22 | 0 | IN   | GPIO. 6 | 6   | 25  |
 |  11 |  14 |    SCLK | ALT0 | 0 | 23 || 24 | 1 | ALT0 | CE0     | 10  | 8   |
 |     |     |      0v |      |   | 25 || 26 | 1 | ALT0 | CE1     | 11  | 7   |
 +-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
 |  28 |  17 | GPIO.17 |   IN | 0 | 51 || 52 | 0 | IN   | GPIO.18 | 18  | 29  |
 |  30 |  19 | GPIO.19 |   IN | 0 | 53 || 54 | 0 | IN   | GPIO.20 | 20  | 31  |
 +-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
 | BCM | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | BCM |
 +-----+-----+---------+------+---+-Model B2-+---+------+---------+-----+-----+
```

If you are a Python user, you can program GPIOs with API provided by RPi.GPIO.

RPi.GPIO Introduction

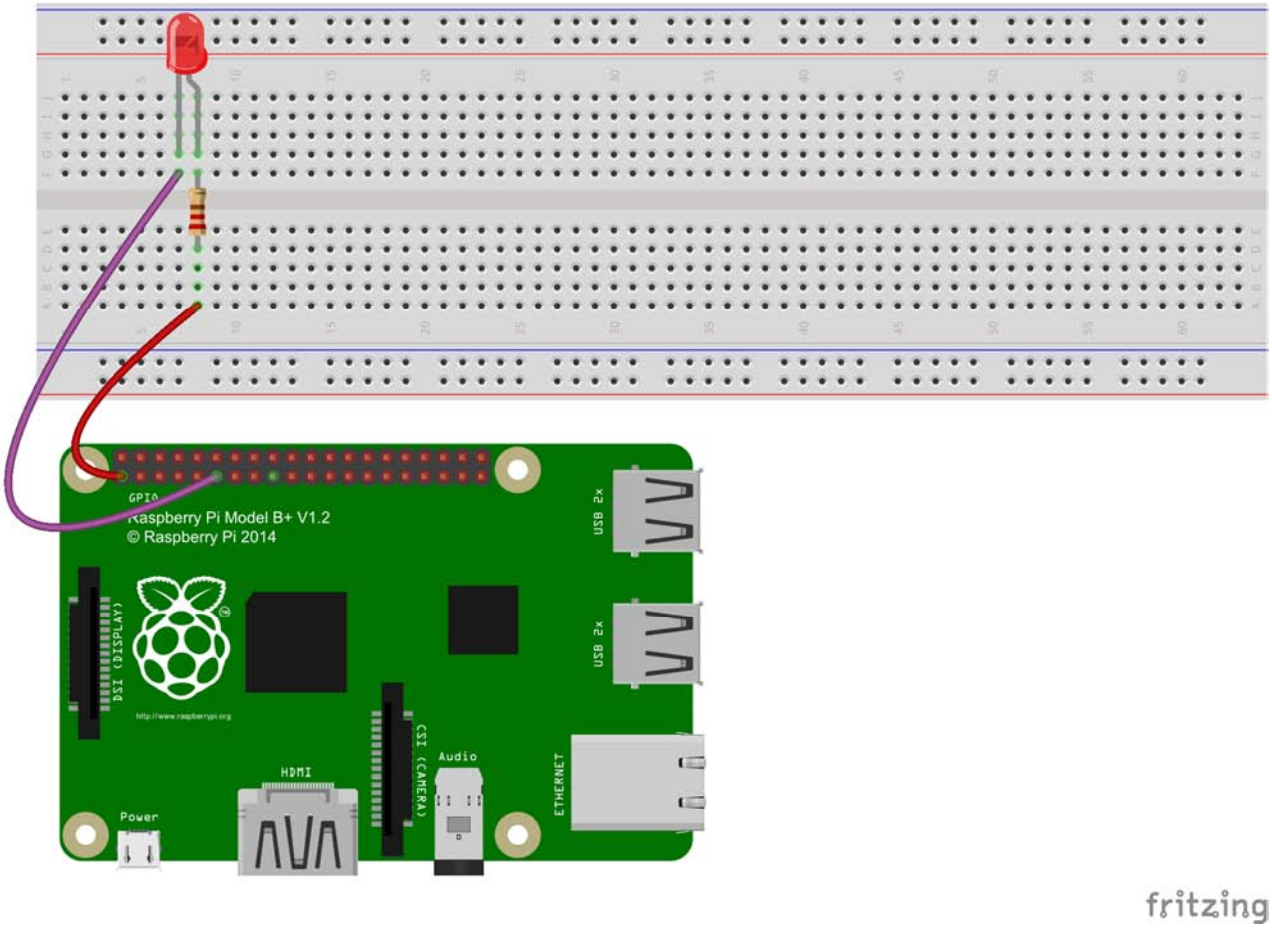
RPi.GPIO is a module to control Raspberry Pi GPIO channels. This package provides a class to control the GPIO on a Raspberry Pi. For examples and documents, visit http://sourceforge.net/p/raspberry-gpio-python/wiki/Home/

Raspbian OS image of Raspberry Pi installs RPi.GPIO by default, so you can use it directly.

You might want to install the python-dev package.

```
sudo apt-get install python-dev
```

Next, use C and Python language to program. You will learn how to use wiringPi and RPi.GPIO module to control Raspberry Pi GPIOs. Take blinking LED for example. First, connect circuit according to the following diagram:



Note: The resistance of the resistor here is 220Ω. If the resistance is too high, the LED will be too dim or even won't light up.

For C language users (based on the wiringPi library):

Step 1: Create a C file named *led.c*

```
touch led.c
```

Step 2: Use nano or other code edit tools to open led.c, write down the following code and save it.

```
#include <wiringPi.h>
#include <stdio.h>

#define  LEDPin    0

int main(void)
{
  if(wiringPiSetup() == -1){//when the wiringPi initialization fails, print
message to the screen.
    printf("setup wiringPi failed !");
    return 1;
  }

  pinMode(LEDPin, OUTPUT);

  while(1){
    digitalWrite(LEDPin, LOW);   //led on
    printf("led on...\n");
    delay(500);
    digitalWrite(LEDPin, HIGH);  //led off
    printf("...led off\n");
    delay(500);
  }
  return 0;
}
```

Step 3: Compile

```
gcc led.c -o led -lwiringPi
```

Step 4: Run

```
sudo ./led
```

You should see the LED blinking. Press Ctrl+C to terminate the program.

For Python users (based on the RPi.GPIO python module):

Step 1: Create a Python file named *led.py*

```
touch led.py
```

Step 2: Use nano or other code edit tools to open *led.py*, write down the following code and save it.

```
#!/usr/bin/env python
import RPi.GPIO as GPIO
import time
```

```
LEDPin = 11    # pin11

GPIO.setmode(GPIO.BOARD)       # Numbers GPIOs by physical location
GPIO.setup(LEDPin, GPIO.OUT)   # Set LEDPin's mode as output
GPIO.output(LEDPin, GPIO.HIGH) # Set LEDPin as high(+3.3V) to turn off the led

try:
  while True:
    print '...led on'
    GPIO.output(LEDPin, GPIO.LOW)  # led on
    time.sleep(0.5)
    print 'led off...'
    GPIO.output(LEDPin, GPIO.HIGH) # led off
    time.sleep(0.5)
except KeyboardInterrupt:  # When 'Ctrl+C' is pressed, the following code will
be  executed.
    GPIO.output(LEDPin, GPIO.HIGH)     # led off
    GPIO.cleanup()                     # Release resource
```

Step 3: Run

```
sudo python led.py
```

You should see the LED blinking. Press Ctrl+C to terminate the program.

Get Source Code

Method 1: From Github.
Change directory to */home/pi*

```
cd /home/pi/
```

Then clone the repository from github

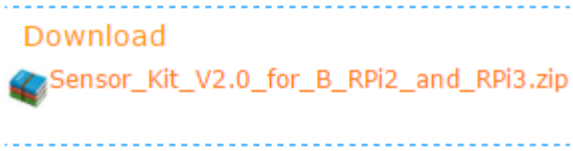
```
git clone https://github.com/sunfounder/SunFounder_SensorKit_for_RPi2.git
```

Method 2: From our website

Download the source code from our website. www.sunfounder.com.

Click **LEARN**->**Get tutorials**, Find and click on *Sensor Kit v2.0 for B+*.

Then on the page of the kit, click the package *Sensor_Kit_V2.0_fo_B_RPi2_and_RPi3.zip* to download.



On your computer, unzip the file and move the folders extracted to */home/pi/* on your Raspberry Pi.

Lesson 1 Dual-Color LED

Introduction

A dual-color light emitting diode (LED) is capable of emitting two different colors of light, typically red and green, rather than only one color. It is housed in a 3mm or 5mm epoxy package. It has 3 leads; common cathode or common anode is available. A dual-color LED features two LED terminals, or pins, arranged in the circuit in anti-parallel and connected by a cathode/anode. Positive voltage can be directed towards one of the LED terminals, causing that terminal to emit light of the corresponding color; when the direction of the voltage is reversed, the light of the other color is emitted. In a dual-color LED, only one of the pins can receive voltage at a time. As a result, this type of LED frequently functions as indicator lights for a variety of devices, including televisions, digital cameras, and remote controls.

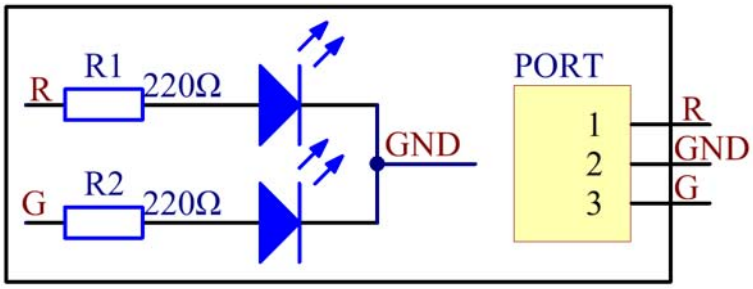
Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires
- 1 * Network cable (or USB wireless network adapter)
- 1 * Dual-color LED module
- 1 * 3-Pin anti-reverse cable

Experimental Principle

Connect pin R and G to GPIOs of Raspberry Pi, program the Raspberry Pi to change the color of the LED from red to green, and then use PWM to mix into other colors.

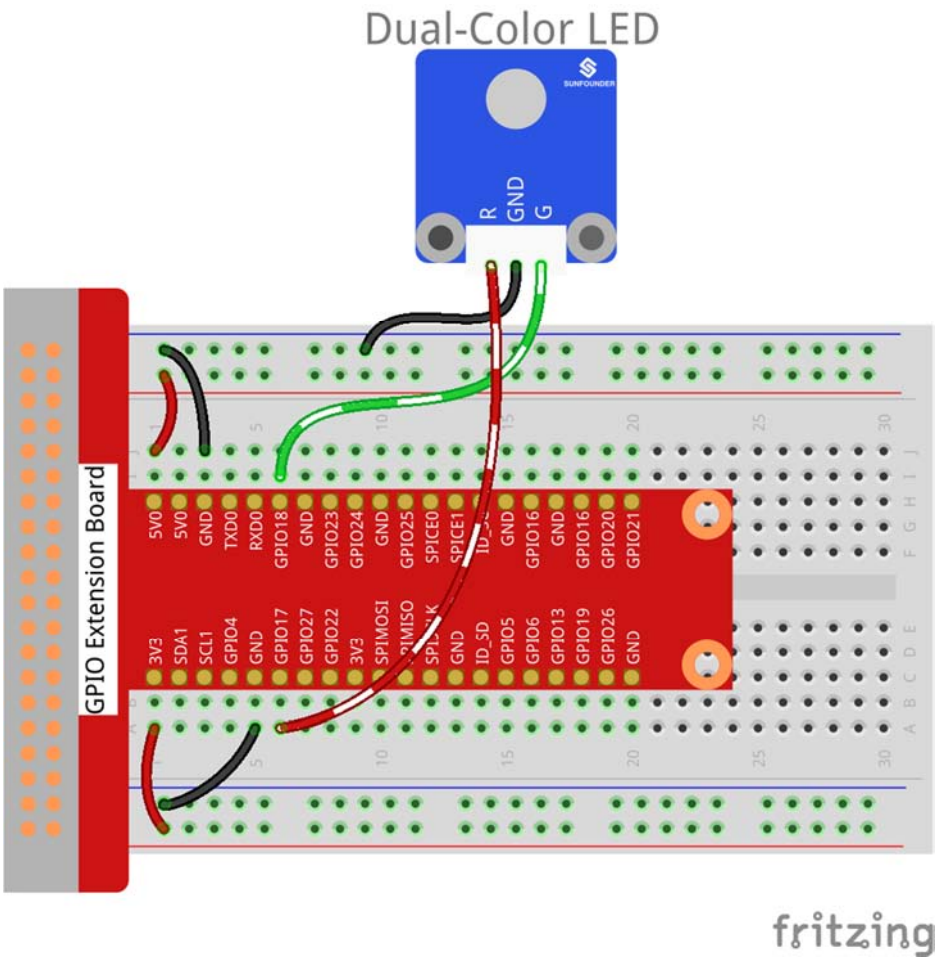
The schematic diagram of the module is as shown below:



Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	Dual-Color LED Module
GPIO0	GPIO17	R
GND	GND	GND
GPIO1	GPIO18	G



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/01_dule_color_led/
```

Step 3: Compile

```
gcc dule_color_led.c -lwiringPi -lpthread
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

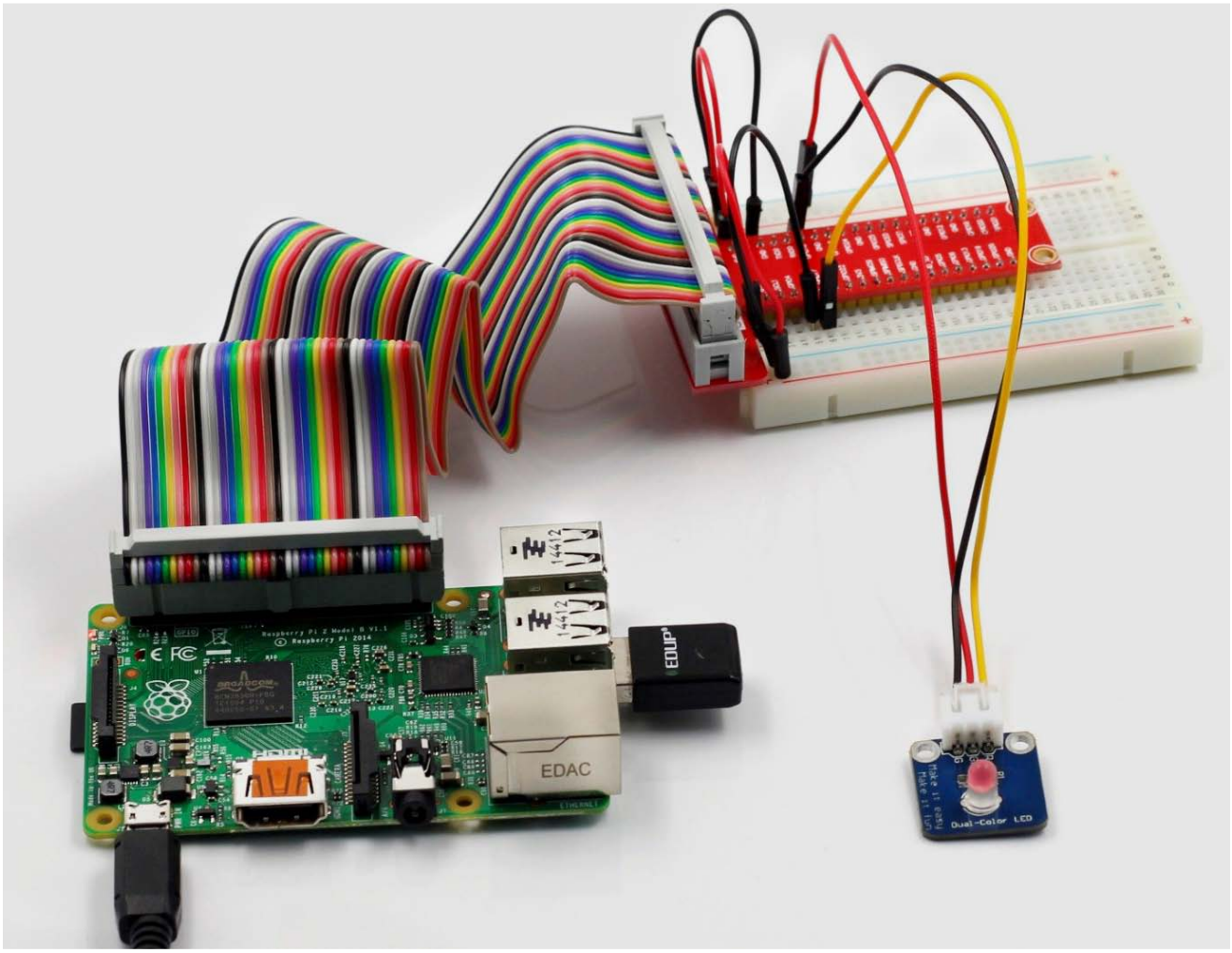
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 01_dule_color_led.py
```

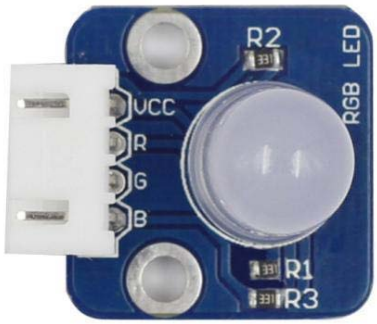
You can see the dual-color LED render green, red, and mixed colors.



Lesson 2 RGB LED Module

Introduction

RGB LED modules can emit various colors of light. Three LEDs of red, green, and blue are packaged into a transparent or semitransparent plastic shell with four pins led out. The three primary colors of red, green, and blue can be mixed and compose all kinds of colors by brightness, so you can make an RGB LED emit colorful light by controlling the circuit.



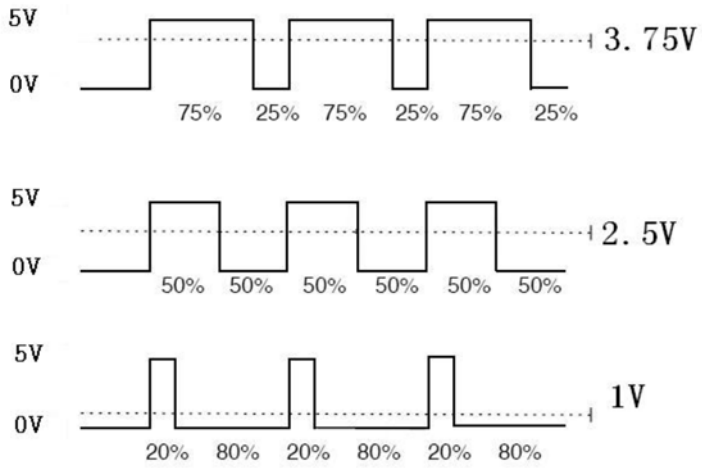
Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * RGB LED module
- 1 * 4-Pin anti-reverse cable

Experimental Principle

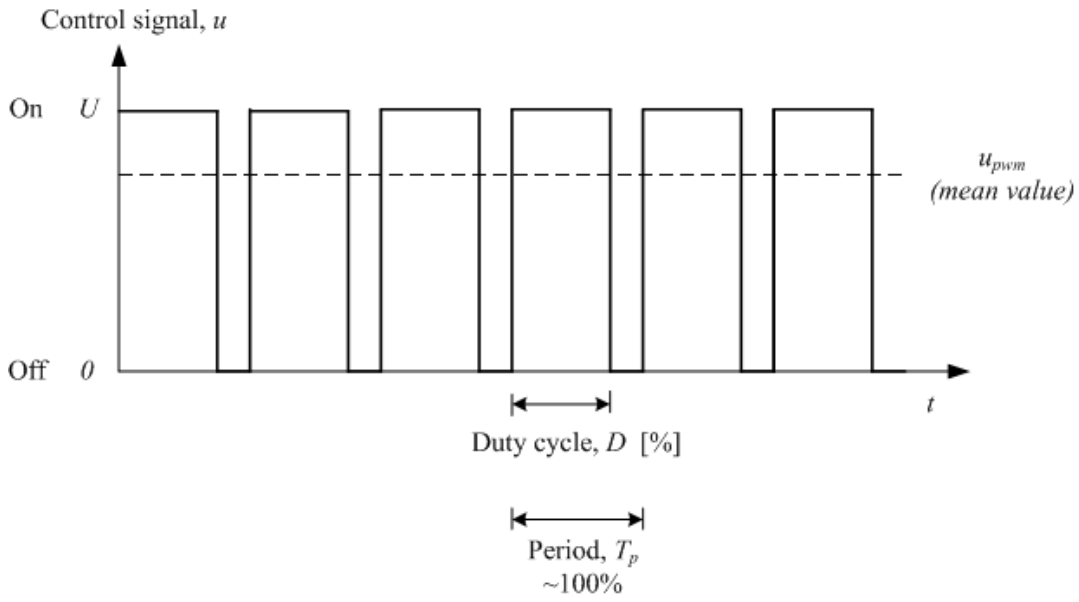
In this experiment, we will use PWM technology to control the brightness of RGB.

Pulse Width Modulation, or PWM, is a technique for getting analog results with digital means. Digital control is used to create a square wave, a signal switched between on and off. This on-off pattern can simulate voltages in between full on (5 Volts) and off (0 Volts) by changing the portion of the time the signal spends on versus the time that the signal spends off. The duration of "on time" is called the pulse width. To get varying analog values, you change, or modulate, that pulse width. If you repeat this on-off pattern fast enough with an LED for example, the result is as if the signal is a steady voltage between 0 and 5v controlling the brightness of the LED.



We can see from the top oscillogram that the amplitude of DC voltage output is 5V. However, the actual voltage output is only 3.75V through PWM, for the high level only takes up 75% of the total voltage within a period.

Here are the three basic parameters of PWM:



1. The term **duty cycle** describes the proportion of "on" time to the regular interval or "period" of time

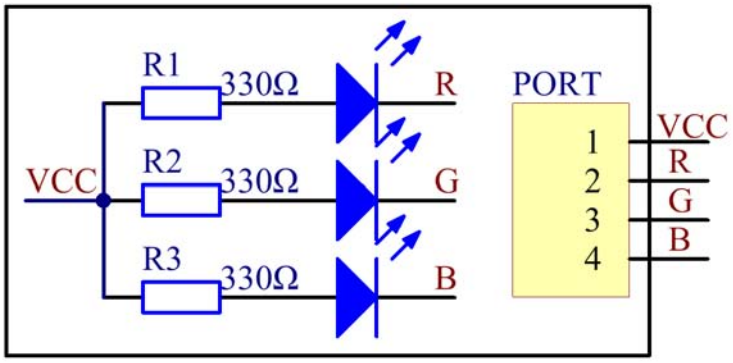
2. **Period** describes the reciprocal of pulses in one second.

3. The voltage amplitude here is 0V-5V.

Here we input any value between 0 and 255 to the three pins of the RGB LED to make it display different colors.

RGB LEDs can be categorized into common anode LED and common cathode LED. In this experiment, we use a common cathode RGB LED.

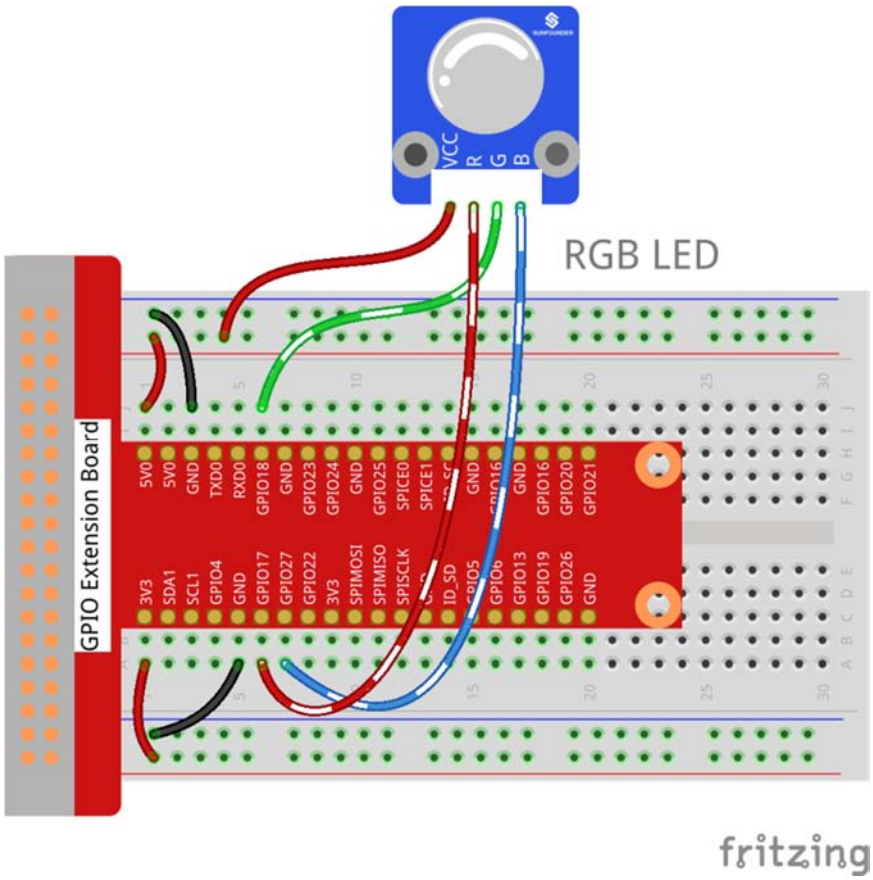
The schematic diagram of the module is as shown below:



Experimental Procedures

Step 1: Build the circuit according to the following method

Raspberry Pi	T-Cobbler	RGB LED Module
5V	5V0	VCC
GPIO0	GPIO17	R
GPIO1	GPIO18	G
GPIO2	GPIO27	B



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/02_rgb_led/
```

Step 3: Compile

```
gcc rgb_led.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

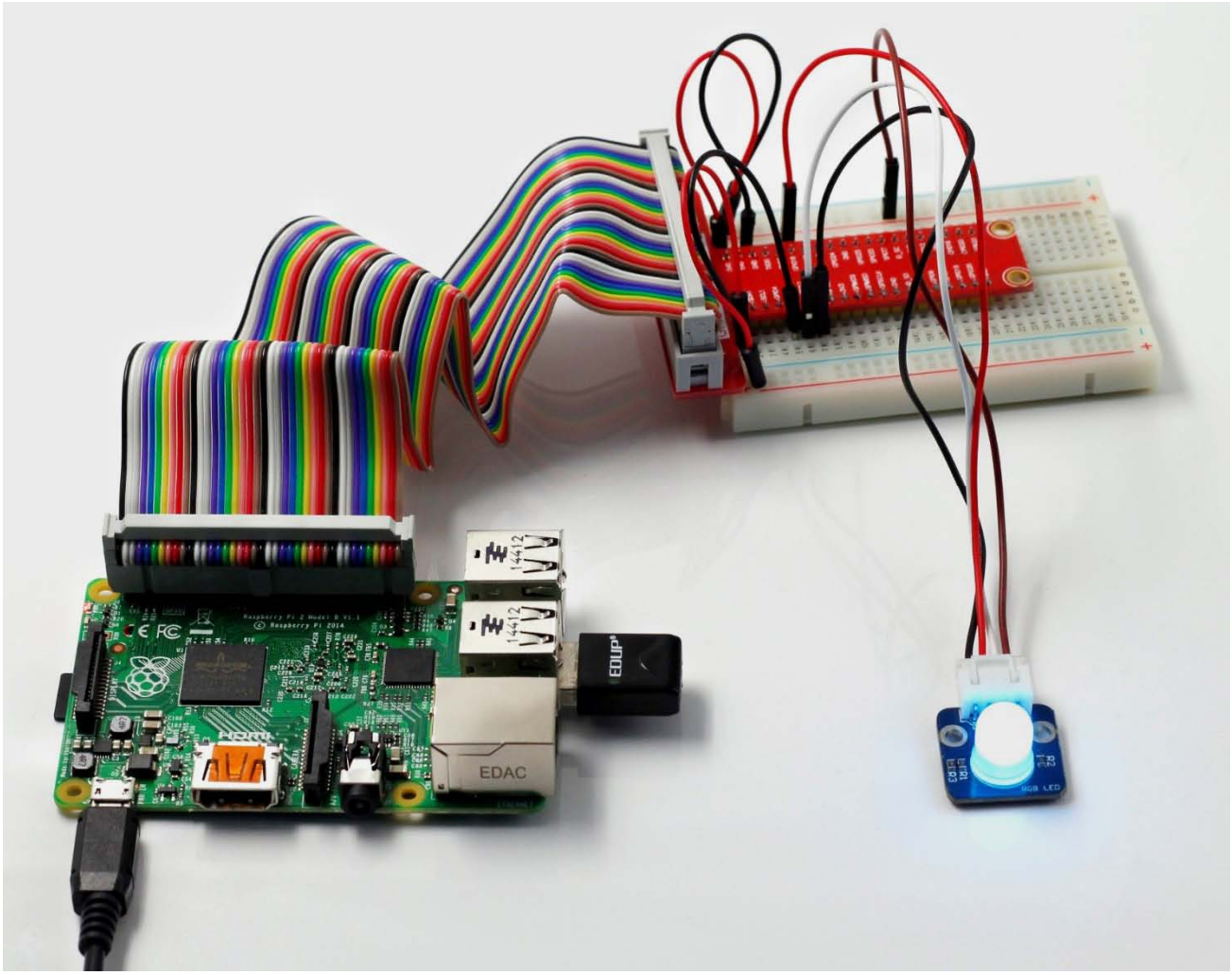
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 02_rgb_led.py
```

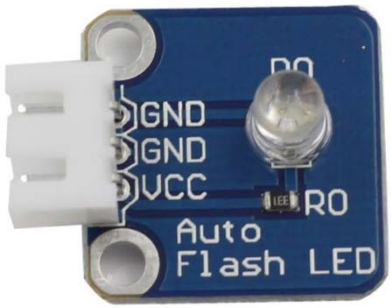
You will see the RGB LED light up, and display different colors in turn.



Lesson 3 7-Color Auto-flash LED

Introduction

On the 7-Color Auto-flash LED module, the LED can automatically flash built-in colors after power on. It can be used to make quite fascinating light effects.



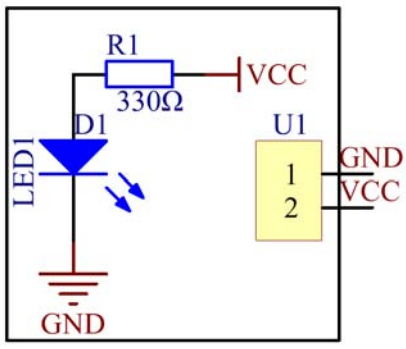
Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * 7-color auto-flash LED module
- 1 * 3-Pin anti-reverse cable

Experimental Principle

When it is power on, the 7-color auto-flash LED will flash built-in colors.

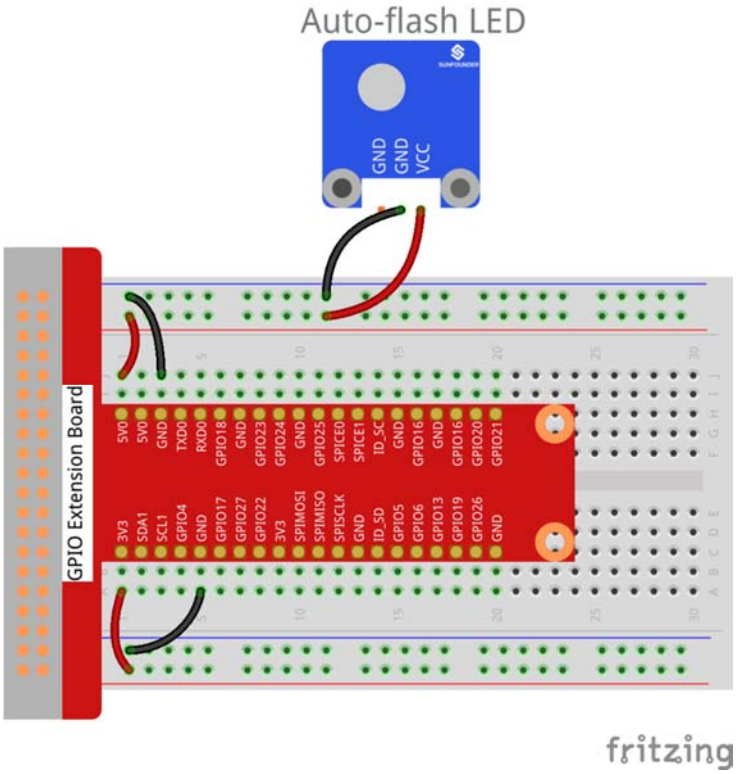
The schematic diagram of the module is as shown below:



Experimental Procedures

Build the circuit

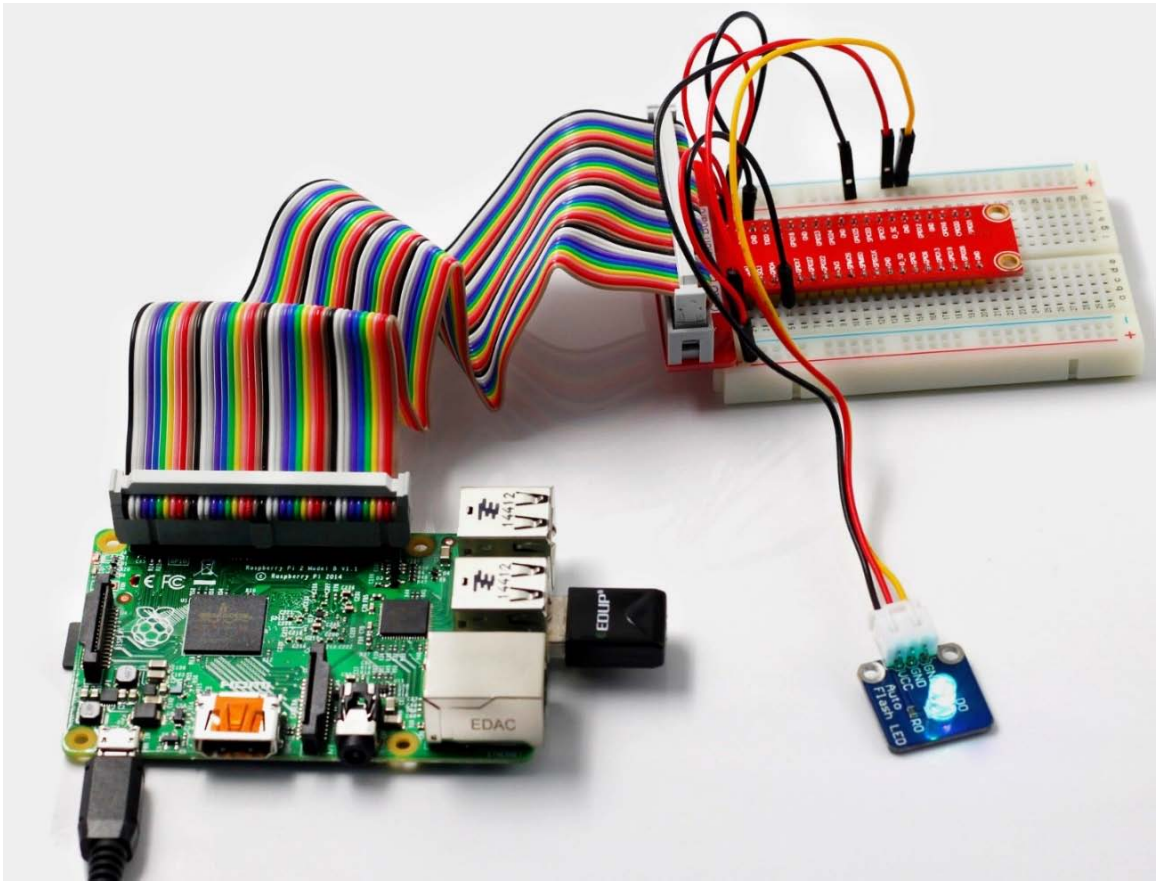
Raspberry Pi	T-Cobbler	Auto-flash LED Module
GND	GND	GND
5V	5V0	VCC



Note:

There are two "GND" pins on the module. You only need to connect one of them.

Now, you will see 7-color auto-flash LED flashing seven colors.



Lesson 4 Relay Module

Introduction

Relay is a device which is used to provide connection between two or more points or devices in response to the input signal applied. It is suitable for driving high power electric equipment, such as light bulbs, electric fans and air conditioning. You can use a relay to control high voltage with low voltage by connecting it to Raspberry Pi.

Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * Relay module
- 1 * Dual-color LED module
- 2 * 3-Pin anti-reverse cable

Experimental Principle

Relay – There are 5 parts in every relay:

1. **Electromagnet** – It consists of an iron core wounded by coil of wires. When electricity is passed through, it becomes magnetic. Therefore, it is called electromagnet.

2. **Armature** – The movable magnetic strip is known as armature. When current flows through them, the coil is it energized thus producing a magnetic field which is used to make or break the normally open (N/O) or normally close (N/C) points. And the armature can be moved with direct current (DC) as well as alternating current (AC).

3. **Spring** – When no currents flow through the coil on the electromagnet, the spring pulls the armature away so the circuit cannot be completed.

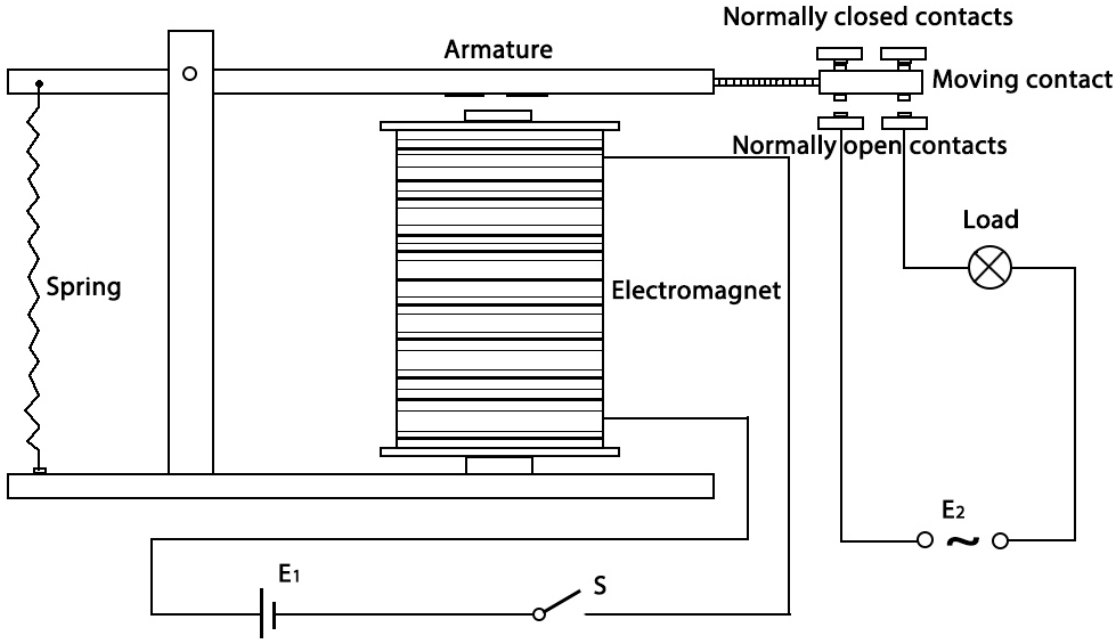
4. Set of electrical **contacts** – There are two contact points:

- Normally open - connected when the relay is activated, and disconnected when it is inactive.
- Normally close – not connected when the relay is activated, and connected when it is inactive.

5. Molded frame – Relays are covered with plastic for protection.

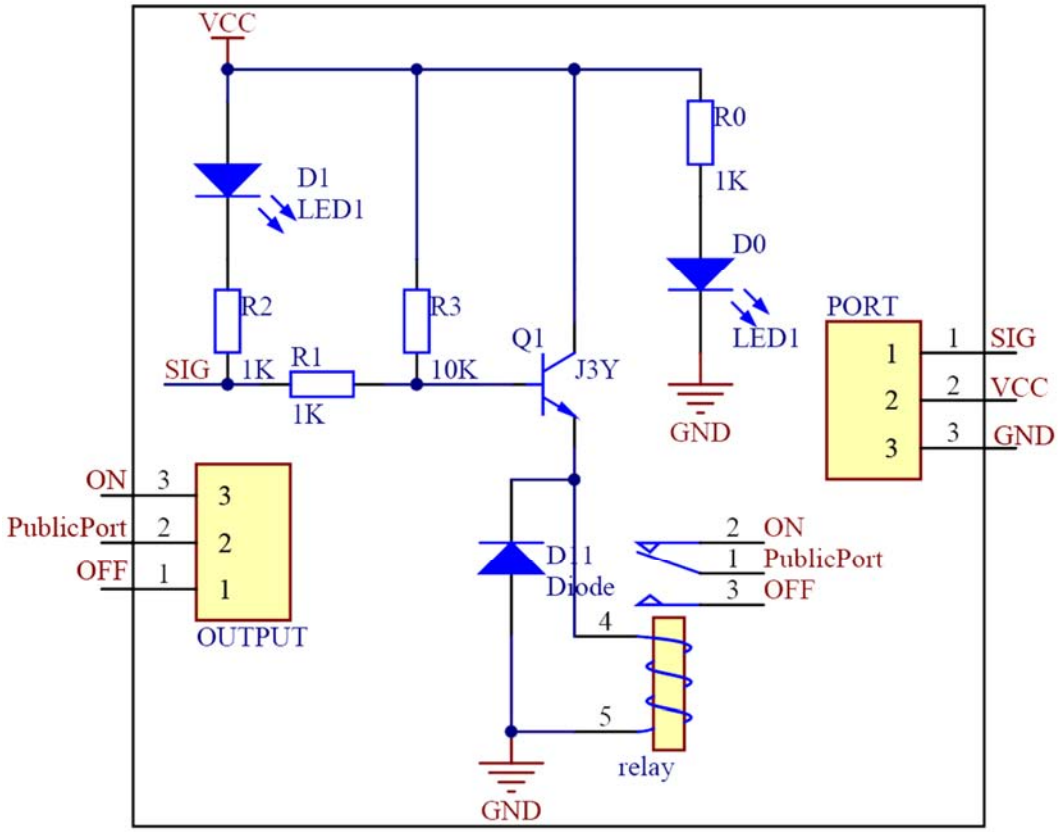
Working of Relay

The working principle of relay is simple. When power is supplied to the relay, currents start flowing through the control coil; as a result, the electromagnet starts energizing. Then the armature is attracted to the coil, pulling down the moving contact together thus connecting with the normally open contacts. So the circuit with the load is energized. Then breaking the circuit would a similar case, as the moving contact will be pulled up to the normally closed contacts under the force of the spring. In this way, the switching on and off of the relay can control the state of a load circuit.



Connect the base electrode of the transistor to GPIO0. When we make GPIO0 output high level (3.3V) by programming, the transistor will conduct because of current saturation. The normally open contact of the relay will be closed, while the normally closed contact of the relay will be broken; when we make it output low level (0V), the transistor will be cut off, and the relay will recover to initial state.

The schematic diagram

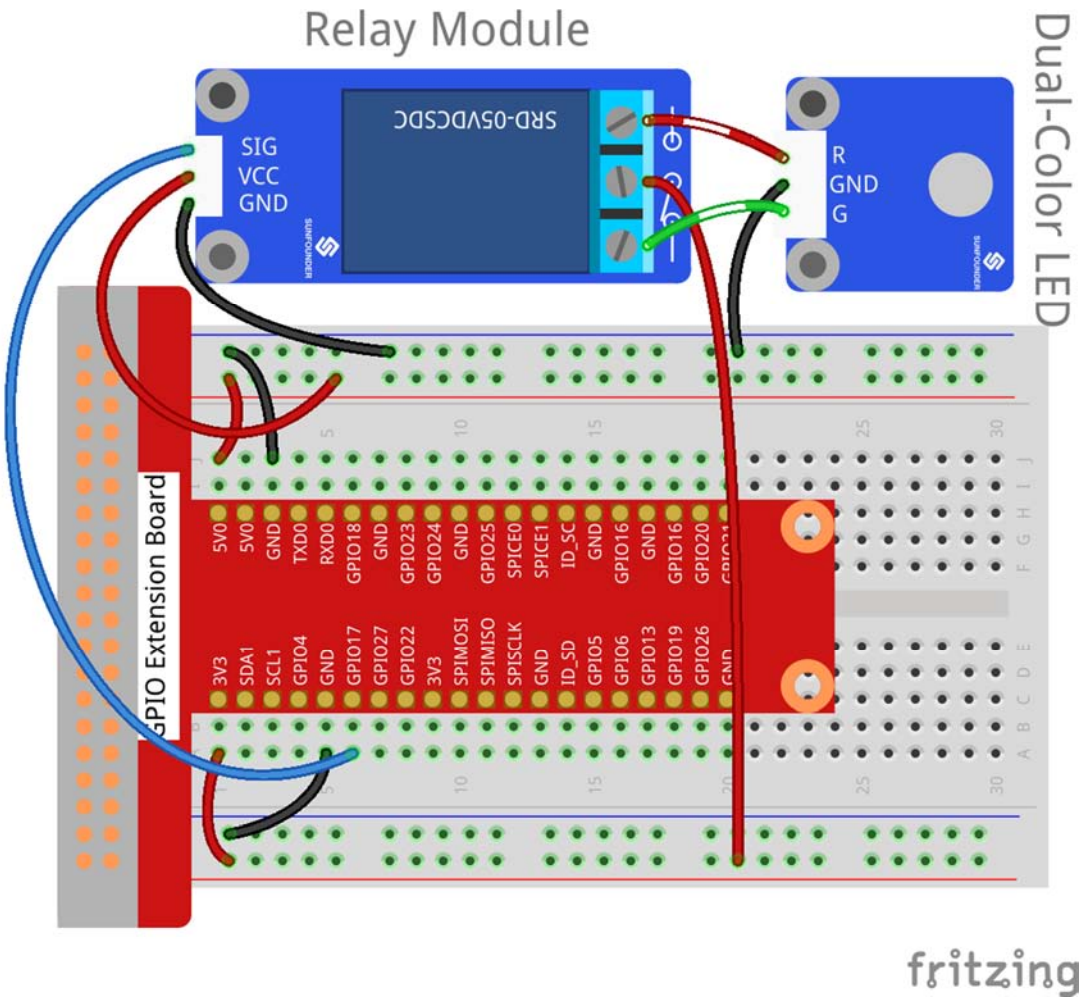


Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	Relay Module
GPIO0	GPIO17	SIG
5V	5V0	VCC
GND	GND	GND
3V3	3V3	COM

Dual-color LED Module	T-Cobbler	Relay Module
R	*	Normal Open
GND	GND	*
G	*	Normal Close



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/04_relay/
```

Step 3: Compile

```
gcc relay.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

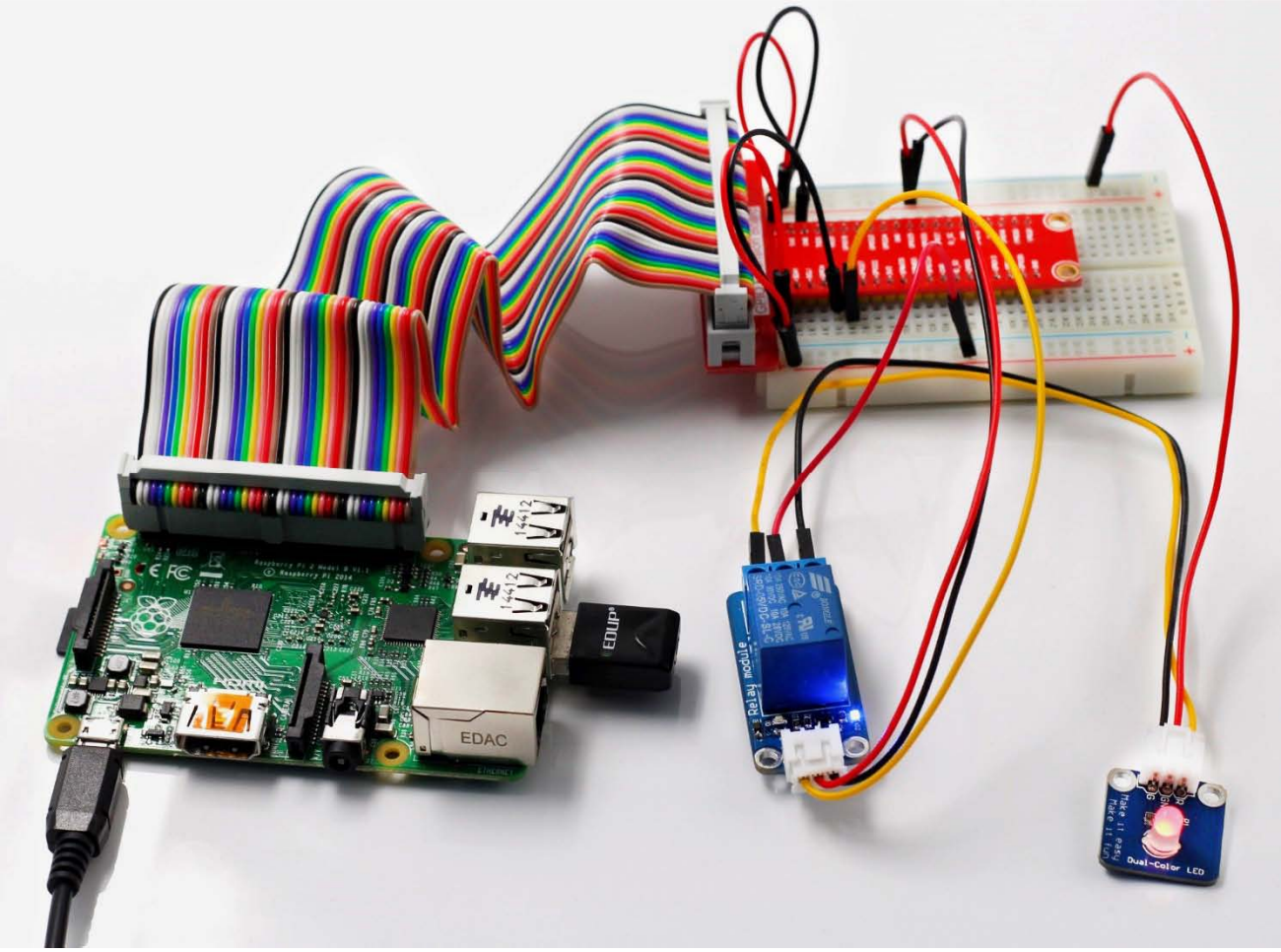
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 04_relay.py
```

Now, you may hear the ticktock. That's the normally closed contact opened and the normally open contact closed. You can attach a high voltage device you want to control, like a 220V bulb, to the output port of the relay. Then the relay will act as an automatic switch.



Lesson 5 Laser Emitter Module

Introduction

Laser is widely used in medical treatment, military, and other fields due to its good directivity and energy concentration.

Components

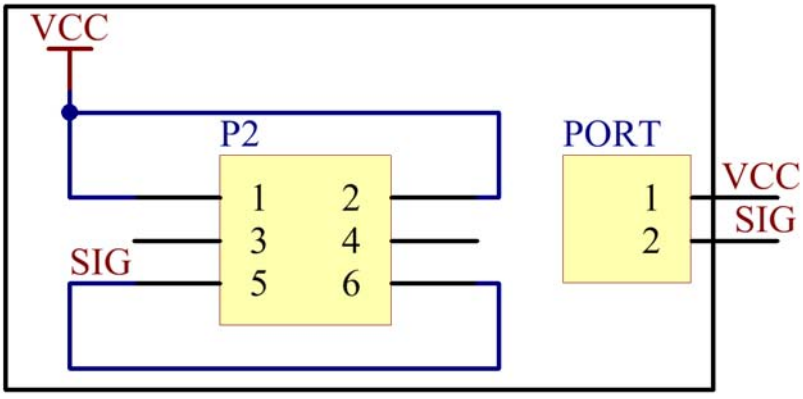
- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * Laser Emitter module
- 1 * 2-Pin anti-reverse cable

Experimental Principle

A laser is a device that emits light through a process of optical amplification based on the stimulated emission of electromagnetic radiation. Lasers differ from other sources of light because they emit light coherently.

Spatial coherence allows a laser to be focused to a tight spot, enabling applications like laser cutting and lithography, and a laser beam to stay narrow over long distances (collimation), enabling applications like laser pointers. Lasers can also have high temporal coherence which allows them to have a very narrow spectrum, i.e., they only emit light of a single color. And its temporal coherence can be used to produce pulses of light—as short as a femtosecond.

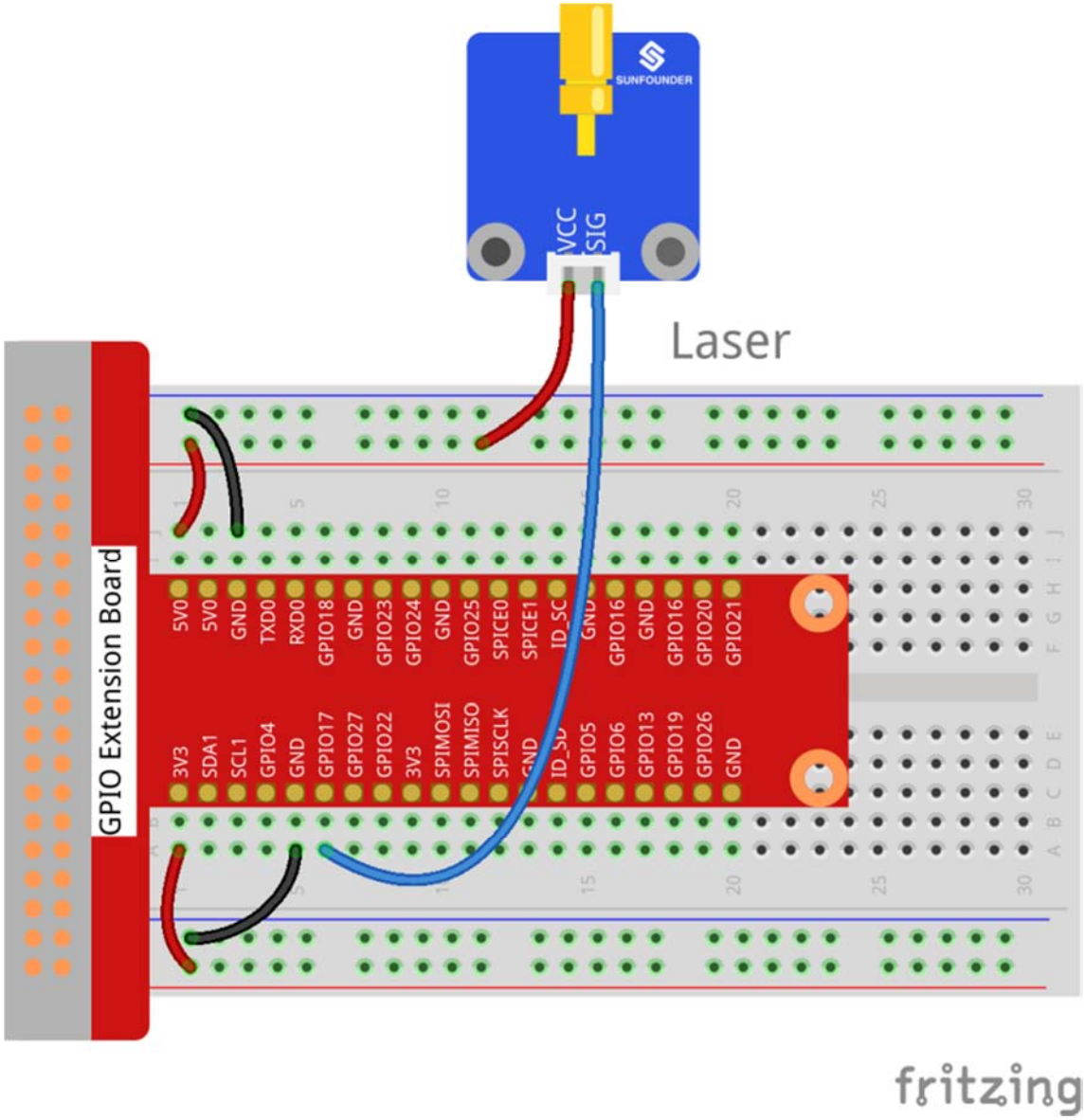
The schematic diagram:



Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	Laser Emitter Module
5V	5V0	VCC
GPIO0	GPIO17	SIG



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/05_laser/
```

Step 3: Compile

```
gcc laser.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

Step 2: Change directory

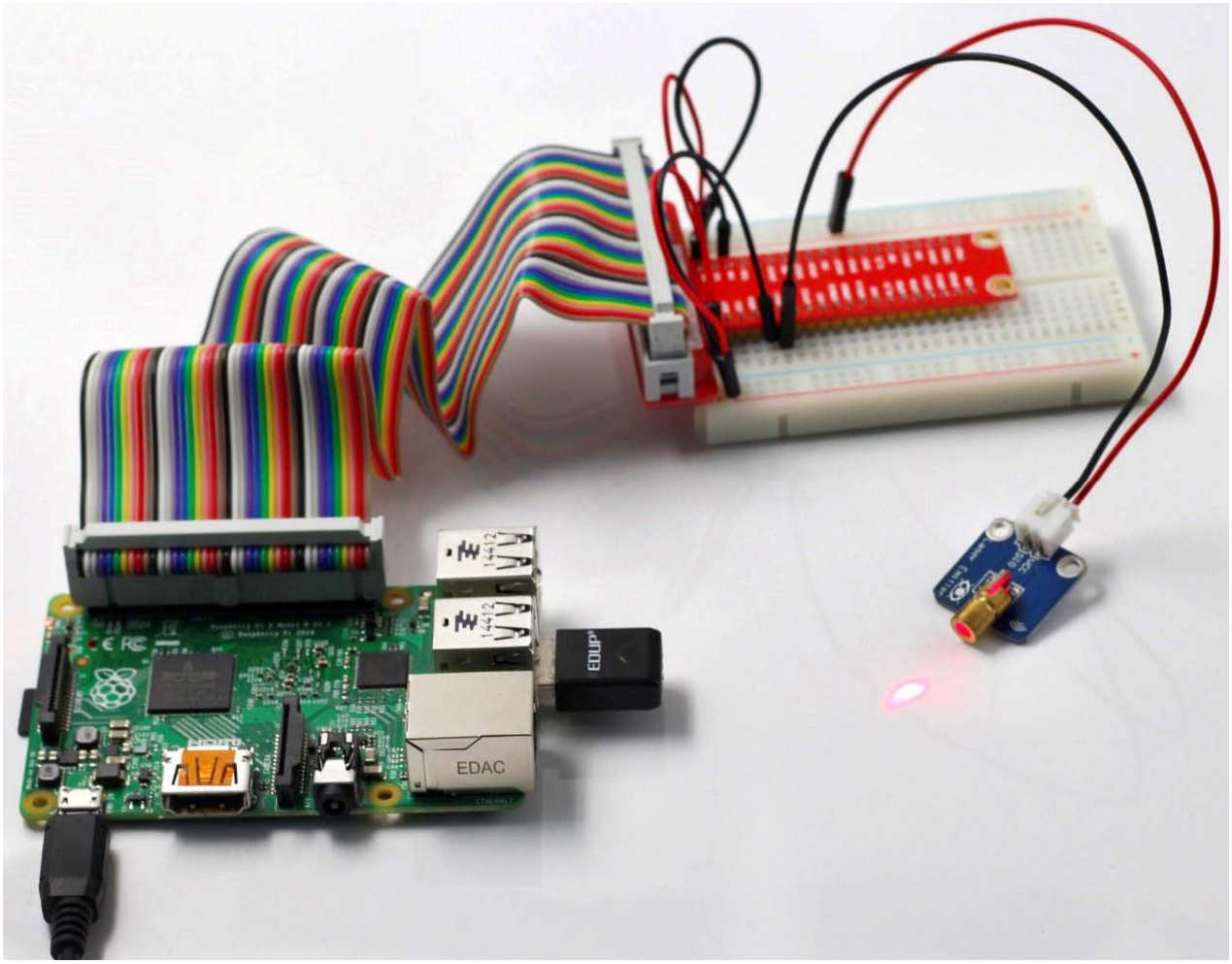
```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 05_laser.py
```

Now you can see the module send out Morse signals.

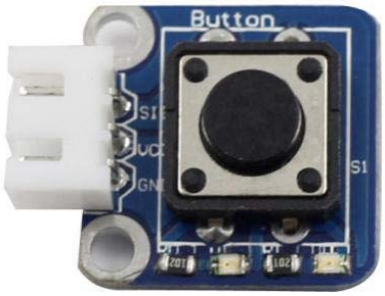
Note: Do NOT look directly at the laser head. It can cause great harm to your eyes. You can point the laser beam to the table and see the light spot flashing on the table.



Lesson 6 Button Module

Introduction

In this lesson, we will use button module to control a dual-color LED module.



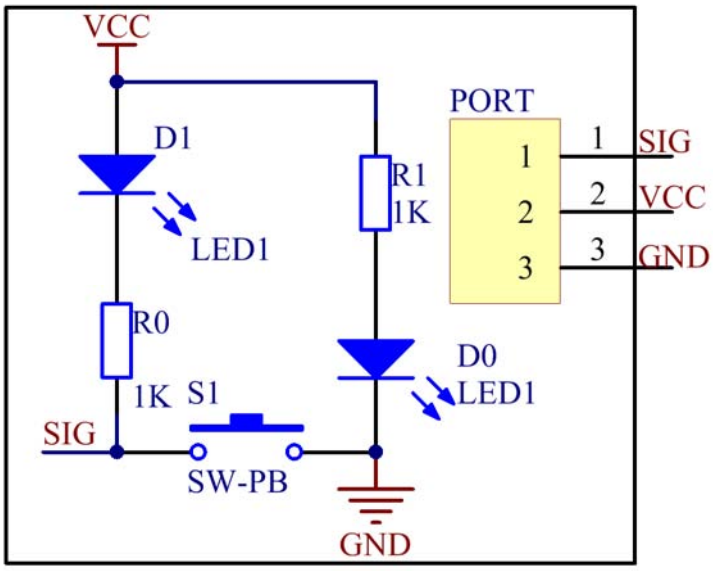
Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * Button module
- 1 * Dual-color LED module
- 2 * 3-Pin anti-reverse cable

Experimental Principle

Use a normally open button as an input device of Raspberry Pi. When the button is pressed, the General Purpose Input/Output (GPIO) connected to the button will change to low level (0V). You can detect the state of the GPIO through programming. That is, if the GPIO turns into low level, it means the button is pressed, so you can run the corresponding code. In this experiment, we will print a string on the screen and control an LED.

The schematic diagram:

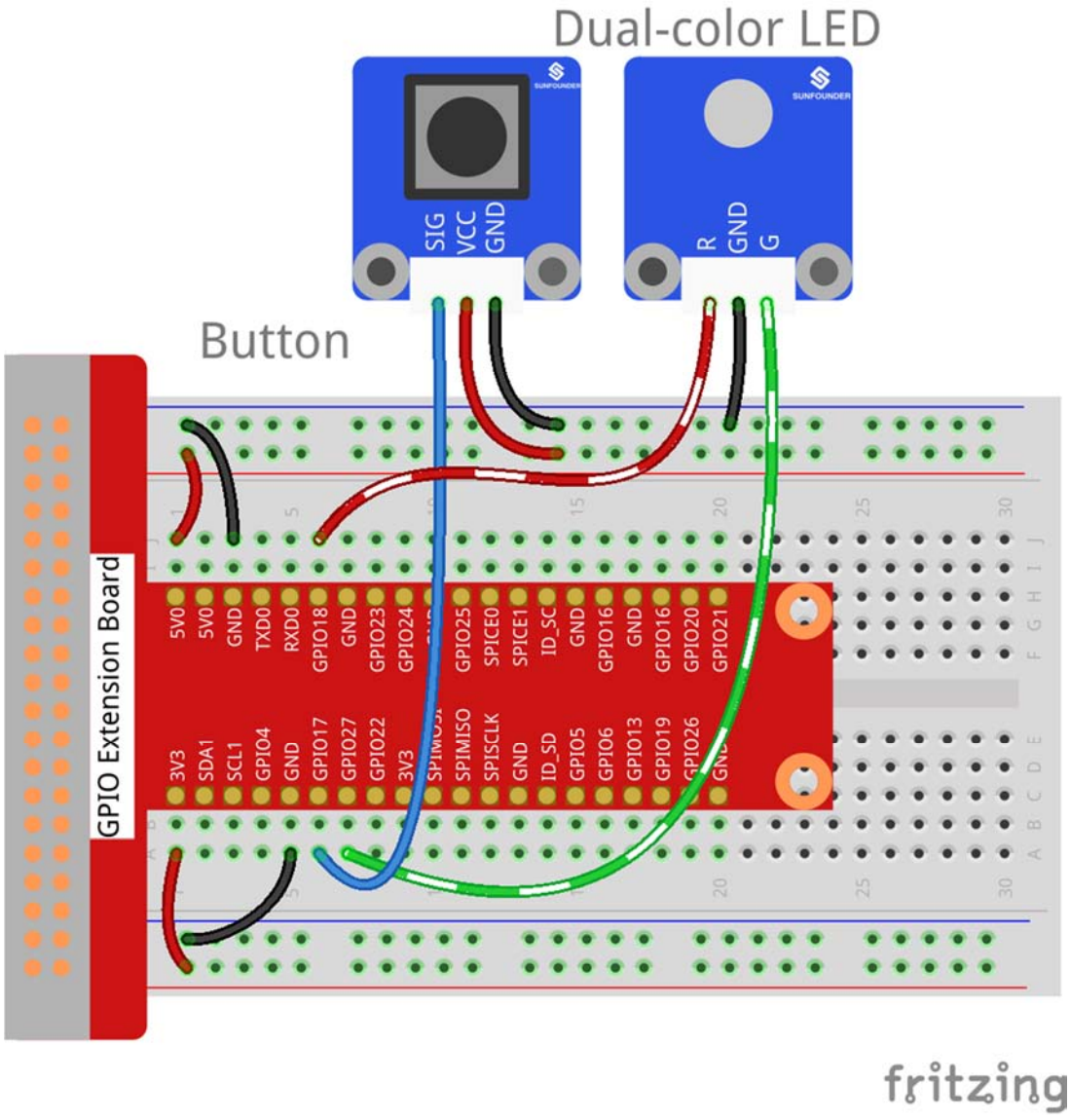


Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	Button Module
GPIO0	GPIO17	SIG
5V	5V0	VCC
GND	GND	GND

Raspberry Pi	T-Cobbler	Dual-Color LED Module
GPIO1	GPIO18	R
GND	GND	GND
GPIO2	GPIO27	G



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/06_button/
```

Step 3: Compile

```
gcc button.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

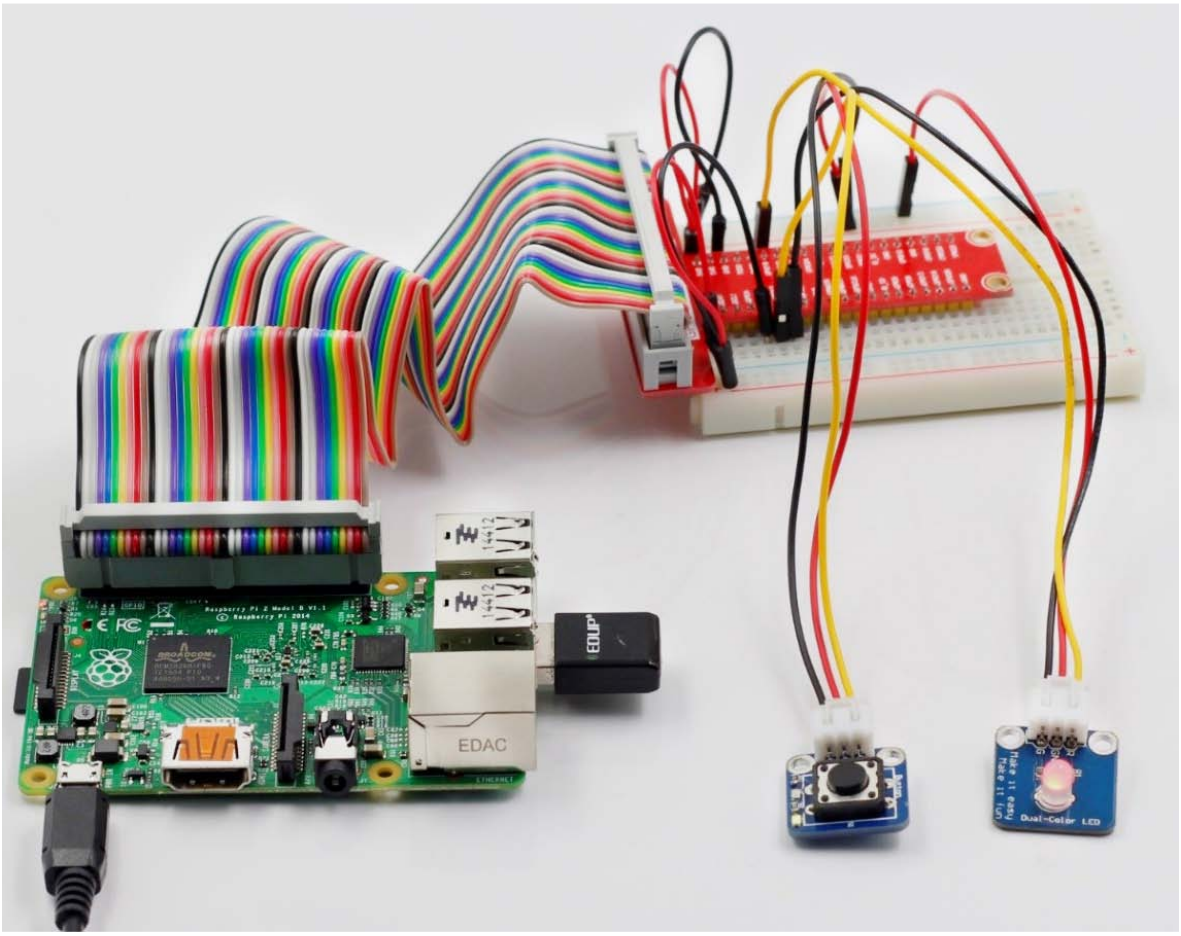
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 06_button.py
```

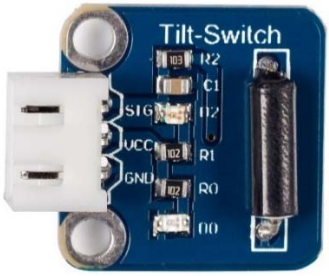
The LED on the module will emit green light. If you press the button, "Button pressed" will be printed on the screen and the LED will emit red light. If you release the button, "Button released" will be printed on the screen and the LED will flash green again.



Lesson 7 Tilt-Switch Module

Introduction

The tilt-switch module (as shown below) in this kit is a ball tilt-switch with a metal ball inside. It is used to detect inclinations of a small angle.



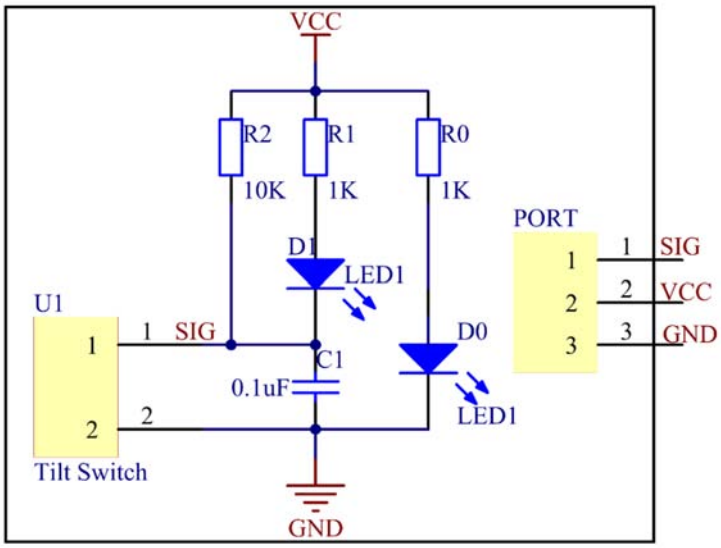
Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * Dual-color LED module
- 1 * Tilt-switch module
- 2 * 3-Pin anti-reverse cable

Experimental Principle

The principle is very simple. The ball in the tilt-switch changes with different angle of inclination to trigger the circuit. When the ball in tilt switch runs from one end to the other end due to shaking caused by external force, the tilt switch will conduct and the LED will emit red light, otherwise it will break and the LED will emit green light.

The schematic diagram:

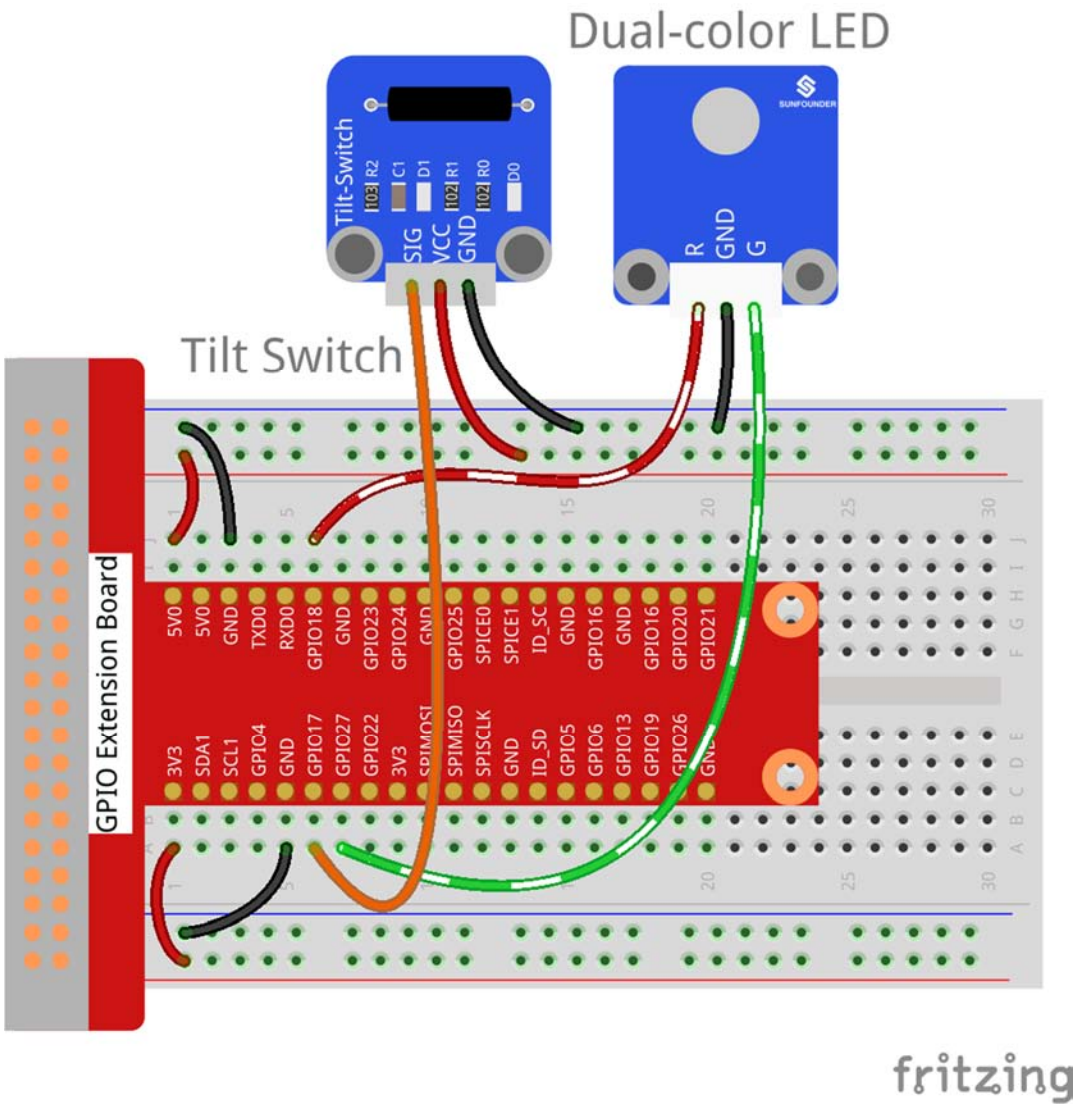


Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	Tilt Switch Module
GPIO0	GPIO17	SIG
5V	5V0	VCC
GND	GND	GND

Raspberry Pi	T-Cobbler	Dual-Color LED Module
GPIO1	GPIO18	R
GND	GND	GND
GPIO2	GPIO27	G



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/07_tilt_switch/
```

Step 3: Compile

```
gcc tilt_switch.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

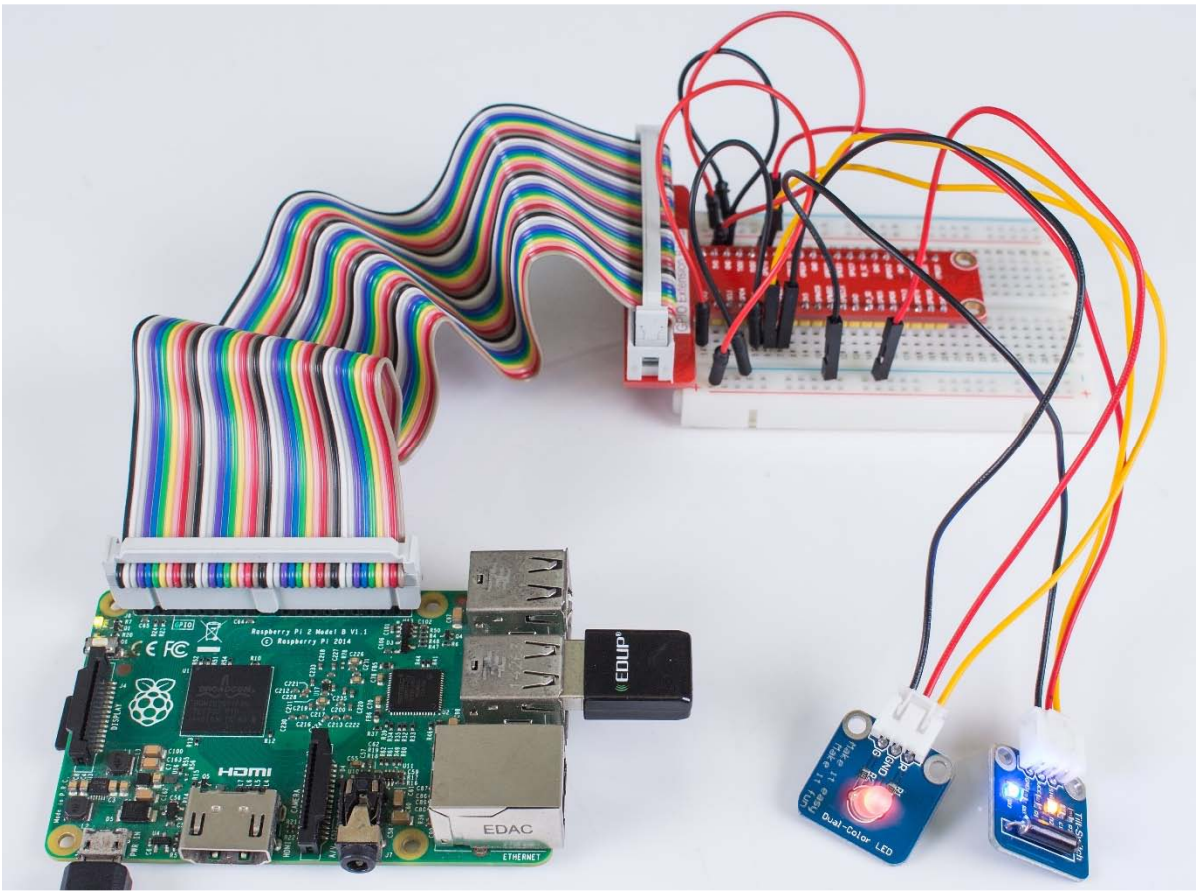
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 07_tilt_switch.py
```

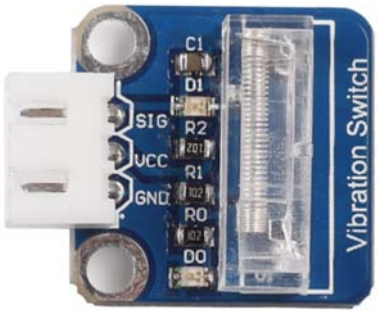
Place the tilt switch module horizontally, and the LED will flash green. If you tilt it, "Tilt!" will be printed on the screen and the LED will change to red. Place it horizontally again, and the LED will flash green again.



Lesson 8 Vibration Switch

Introduction

A vibration switch, also called spring switch or shock sensor, is an electronic switch which induces shock force and transfers the result to a circuit device thus triggering it to work. It contains the following parts: conductive vibration spring, switch body, trigger pin, and packaging agent.



Components

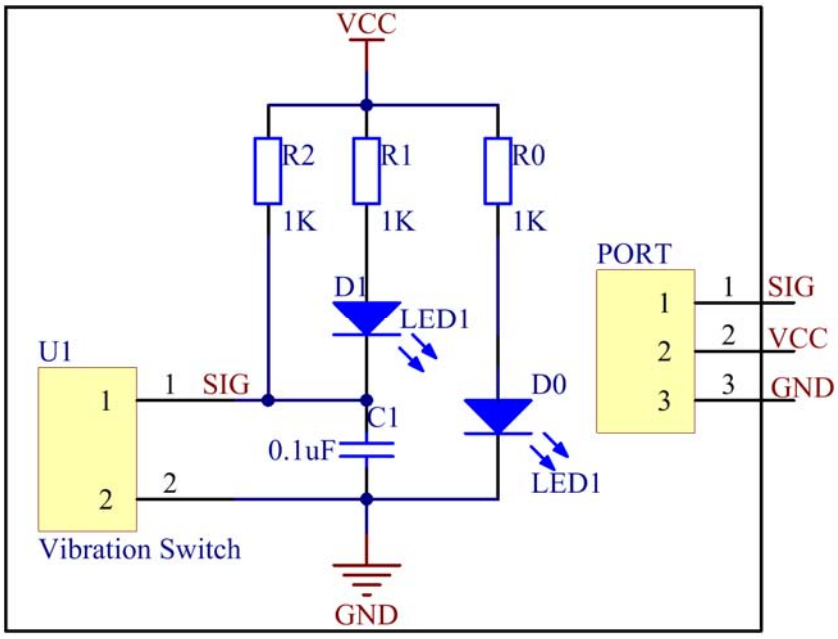
- 1 * Raspberry Pi
- 1 * Breadboard
- 1 * Network cable (or USB wireless network adapter)
- 1 * Dual-color LED module
- 1 * Vibration switch module
- 2 * 3-Pin anti-reverse cable

Experimental Principle

In a vibration switch module, the conductive vibration spring and trigger pin are precisely placed in the switch and fixed by adhesive. Normally, the spring and the trigger pin are separated. Once the sensor detects shock, the spring will vibrate and contact with the trigger pin, thus conducting and generating trigger signals.

In this experiment, connect a dual-color LED module to the Raspberry Pi to indicate the changes. When you knock or tap the vibration sensor, it will get turned on and the dual-color LED will flash red. Tap it again and the LED will change to green – just between the two colors for each tap or knock.

The schematic diagram:

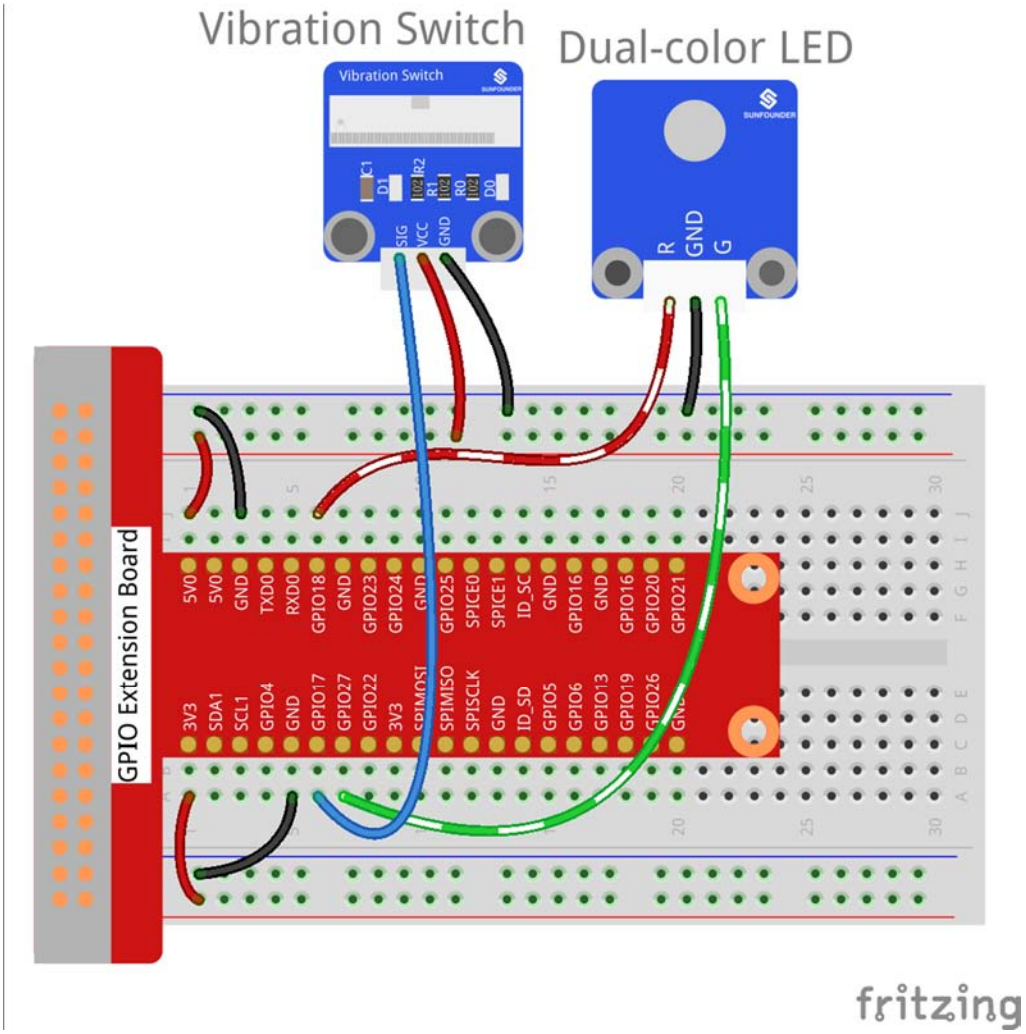


Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	Vibration Switch Module
GPIO0	GPIO17	SIG
5V	5V0	VCC
GND	GND	GND

Raspberry Pi	T-Cobbler	Dual-Color LED Module
GPIO1	GPIO18	R
GND	GND	GND
GPIO2	GPIO27	G



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/08_vibration_switch/
```

Step 3: Compile

```
gcc vibration_switch.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

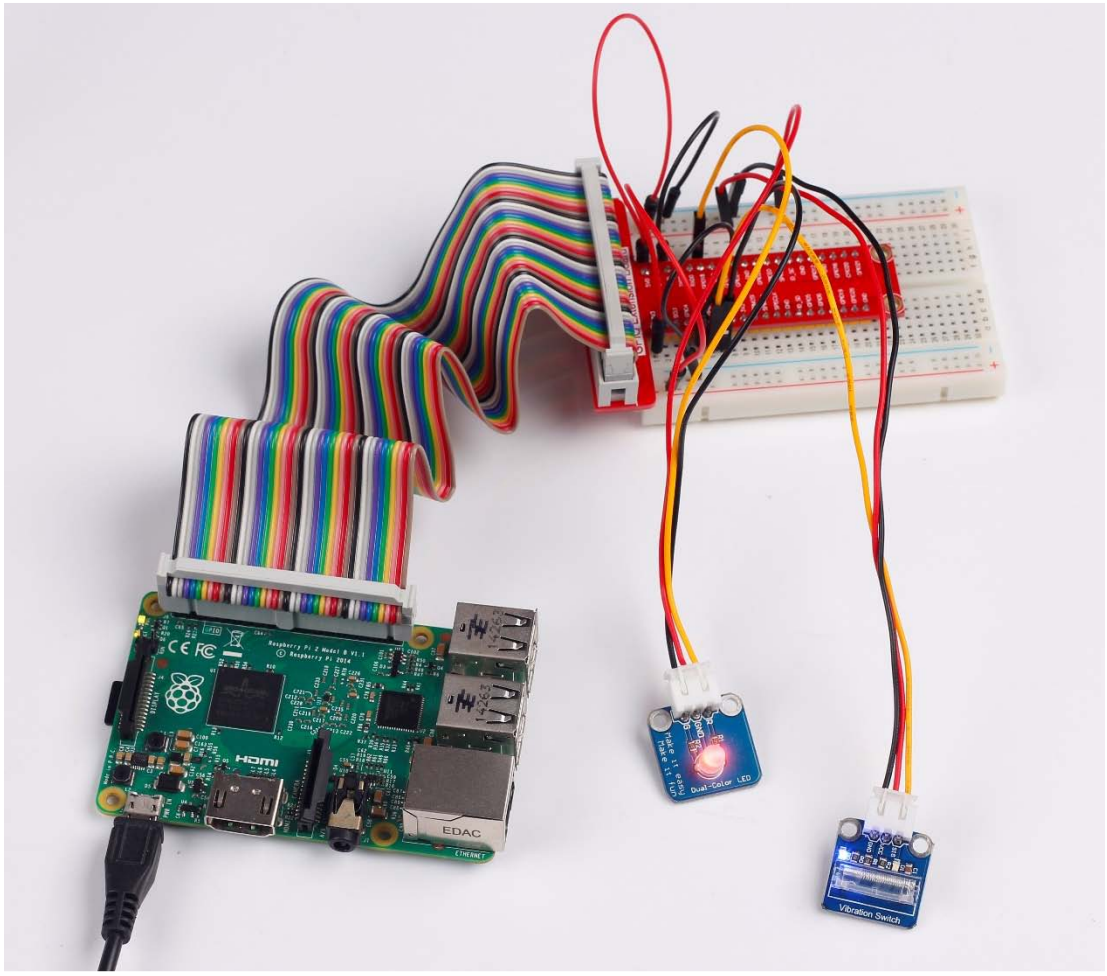
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 08_vibration_switch.py
```

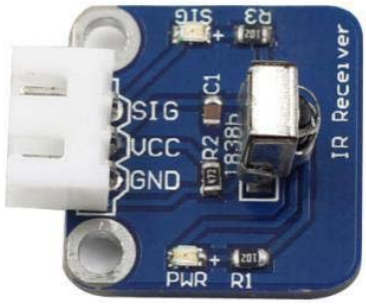
Now tap or knock the module and you can see the dual-color LED flash red. Tap the sensor again, and the LED will change to green. Each tap or knock would make it change between red and green.



Lesson 9 IR Receiver Module

Introduction

An infrared-receiver (as shown below) is a component which receives infrared signals and can independently receive infrared rays and output signals compatible with TTL level. It is similar with a normal plastic-packaged transistor in size and is suitable for all kinds of infrared remote control and infrared transmission.

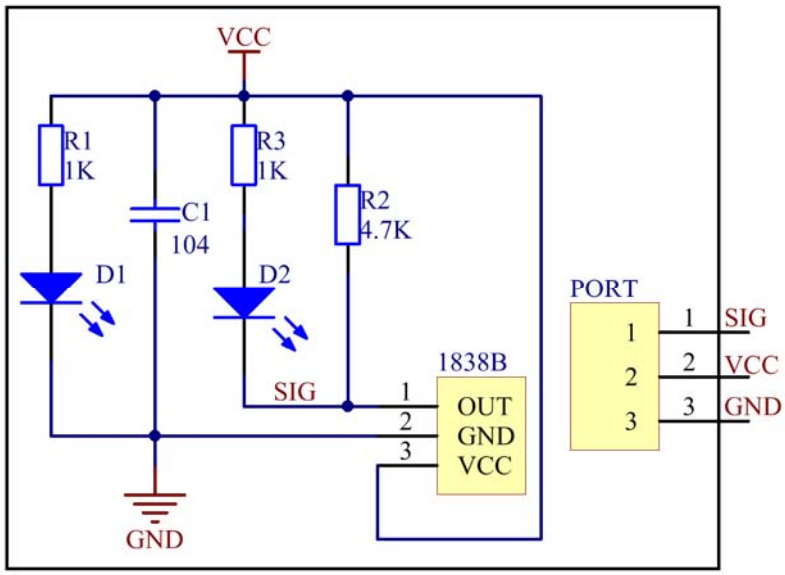


Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * IR receiver module
- 1 * IR Remote Controller
- 1 * 3-Pin anti-reverse cable

Experimental Principle

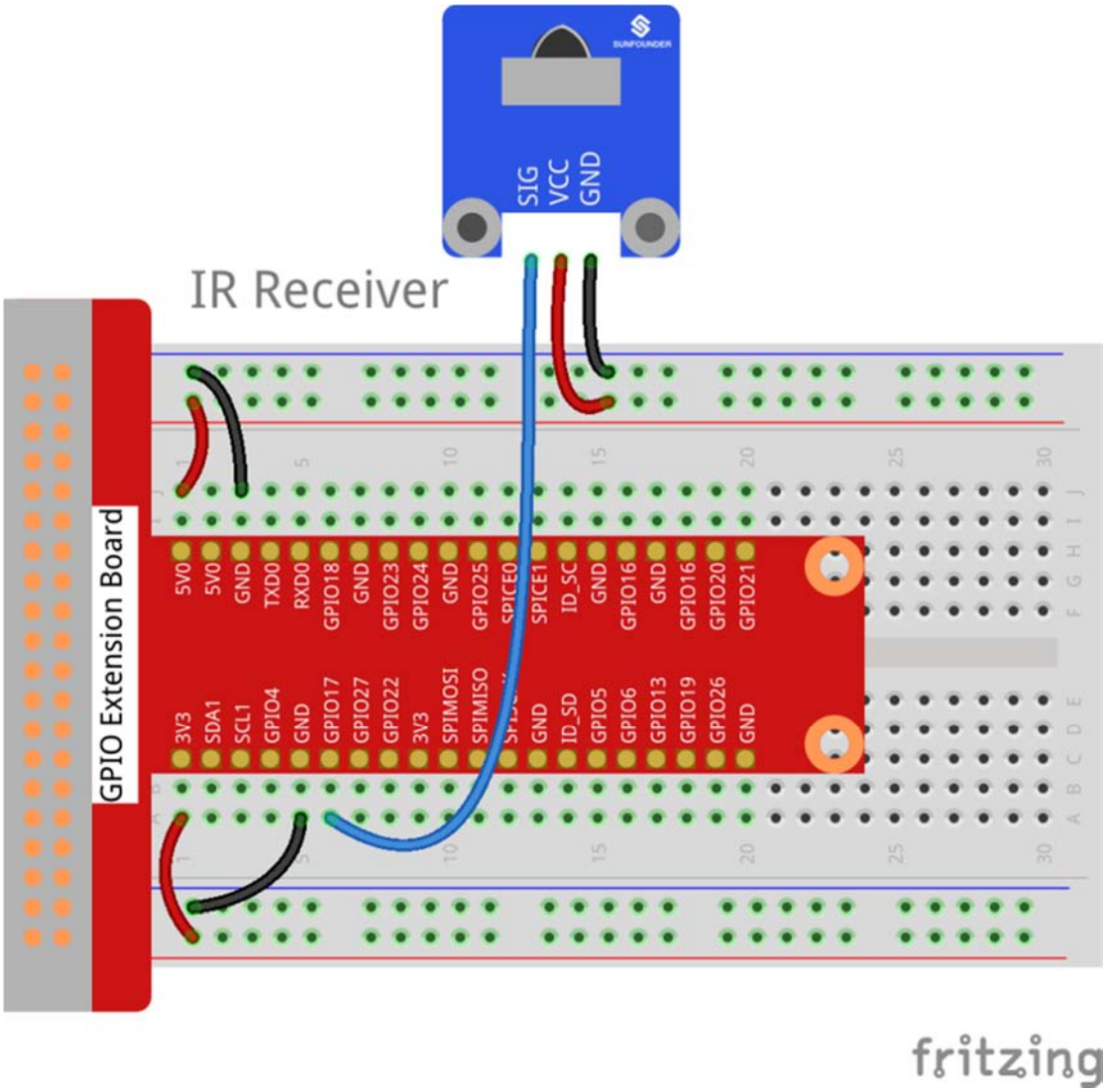
In this experiment, send signals to IR receiver by pressing buttons on the IR remote controller. The counter will add 1 every time it receives signals; in other words, the increased number indicates IR signals are received. The schematic diagram:



Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	IR Receiver Module
GPIO0	GPIO17	SIG
5V	5V0	VCC
GND	GND	GND



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/09_ir_receiver/
```

Step 3: Compile

```
gcc ir_receiver.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

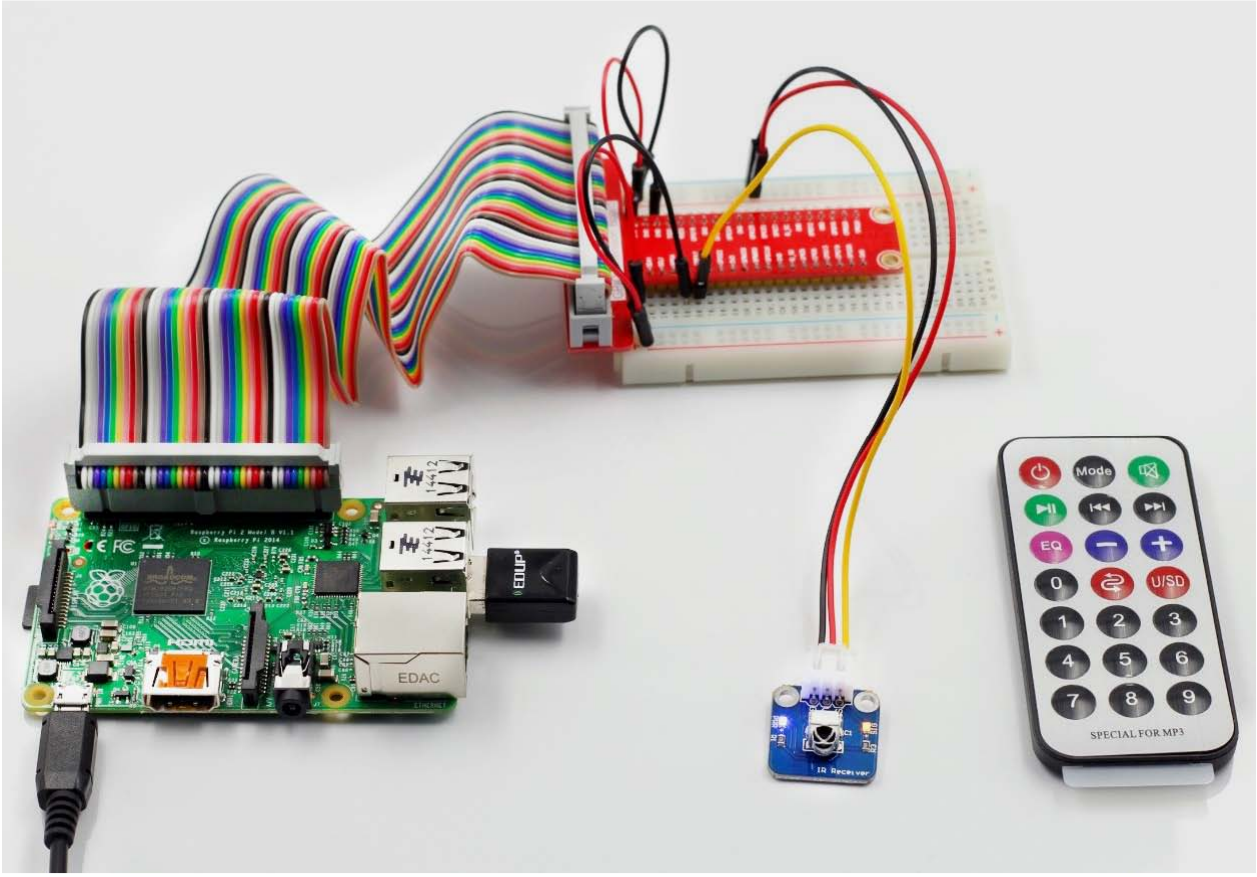
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 09_ir_receiver.py
```

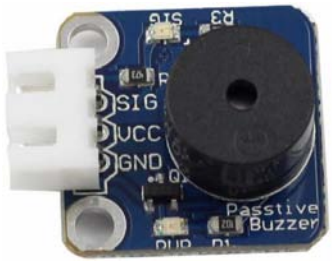
Press any key of the remote. Then you can see the LED on the module blinking, and "Received infrared. cnt = xxx" printed on the screen. ""xxx" means the time you pressed the key(s).



Lesson 10 Buzzer Module

Introduction

Buzzers can be categorized as active and passive ones (See the following picture).



Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * Passive buzzer module
- 1 * Active buzzer module
- 1 * 3-Pin anti-reverse cable

Experimental Principle

Place the pins of two buzzers face up and you can see the one with a green circuit board is a passive buzzer, while the other with a black tape, instead of a board, is an active buzzer.

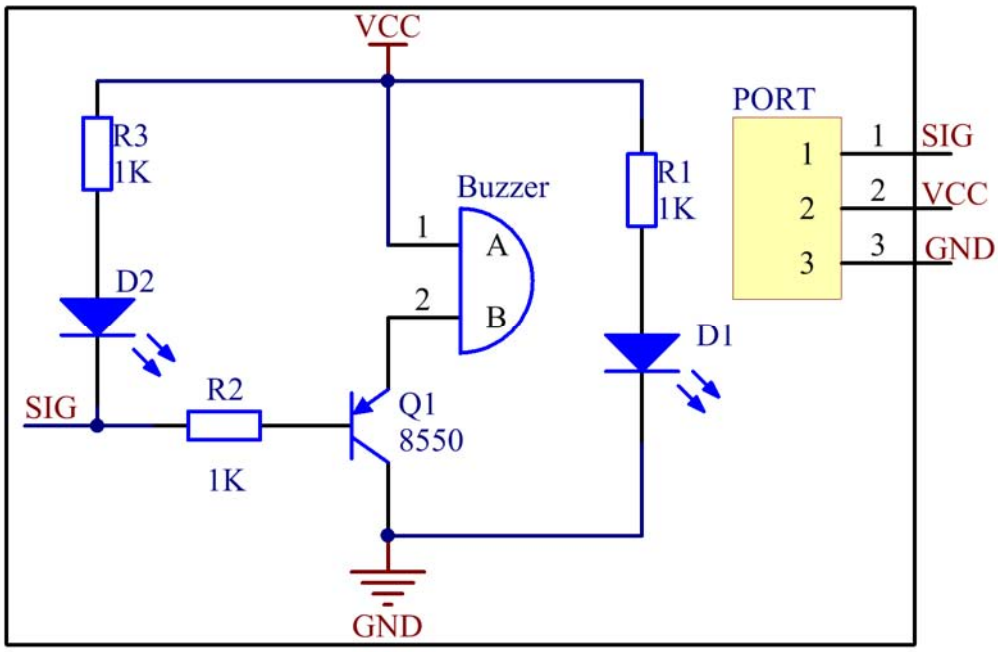
Active buzzer

Passive buzzer

The difference between an active buzzer and a passive buzzer is:

An active buzzer has a built-in oscillating source, so it will make sounds when electrified. But a passive buzzer does not have such source, so it will not beep if DC signals are used; instead, you need to use square waves whose frequency is between 2K and 5K to drive it. The active buzzer is often more expensive than the passive one because of multiple built-in oscillating circuits.

The schematic diagram:



Experimental Procedures

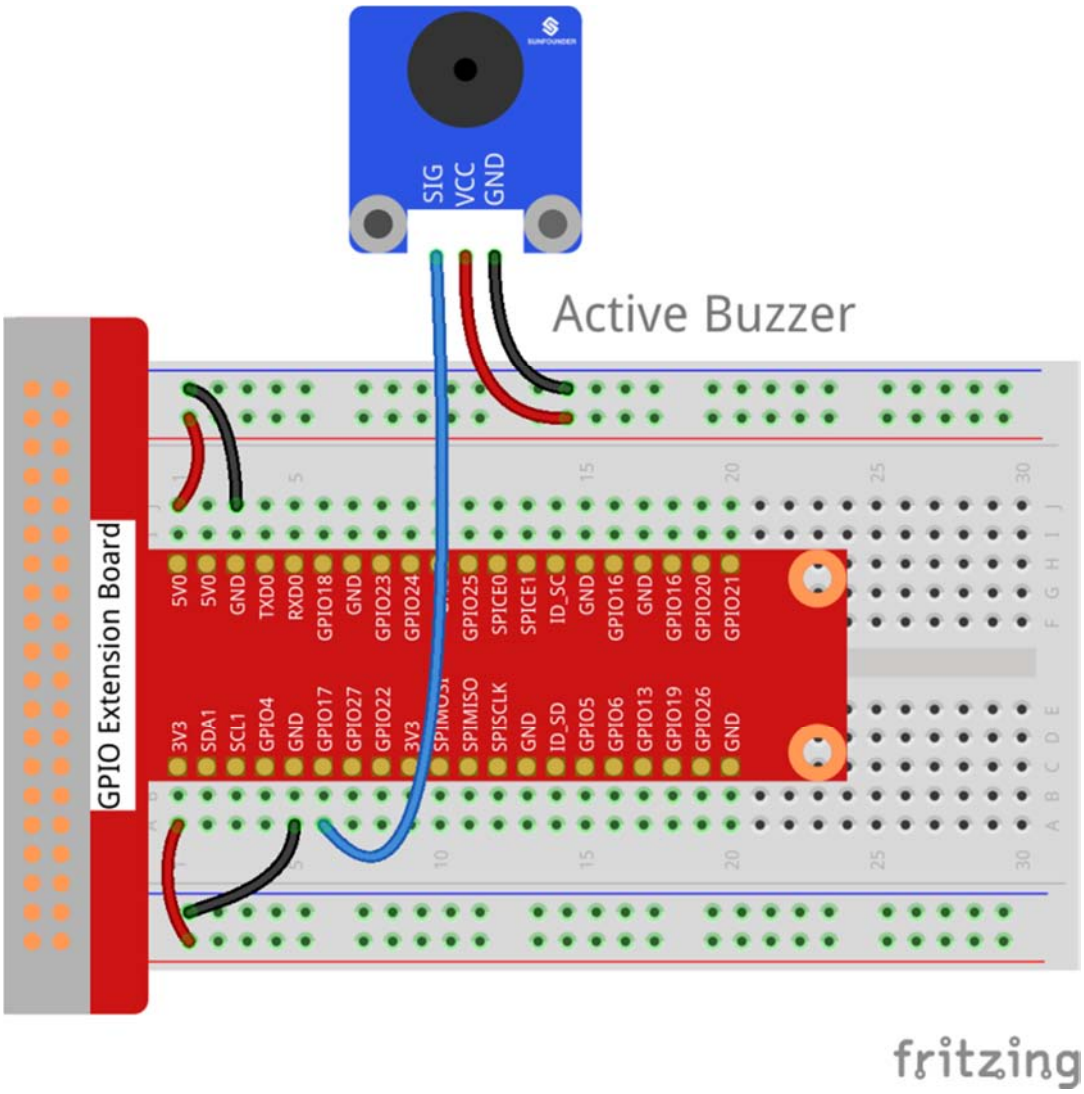
Active Buzzer

Note:

The active buzzer has built-in oscillating source, so it will beep as long as it is wired up, but it can only beep with fixed frequency.

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	Active Buzzer Module
GPIO0	GPIO17	SIG
5V	5V0	VCC
GND	GND	GND



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/10_active_buzzer/
```

Step 3: Compile

```
gcc active_buzzer.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

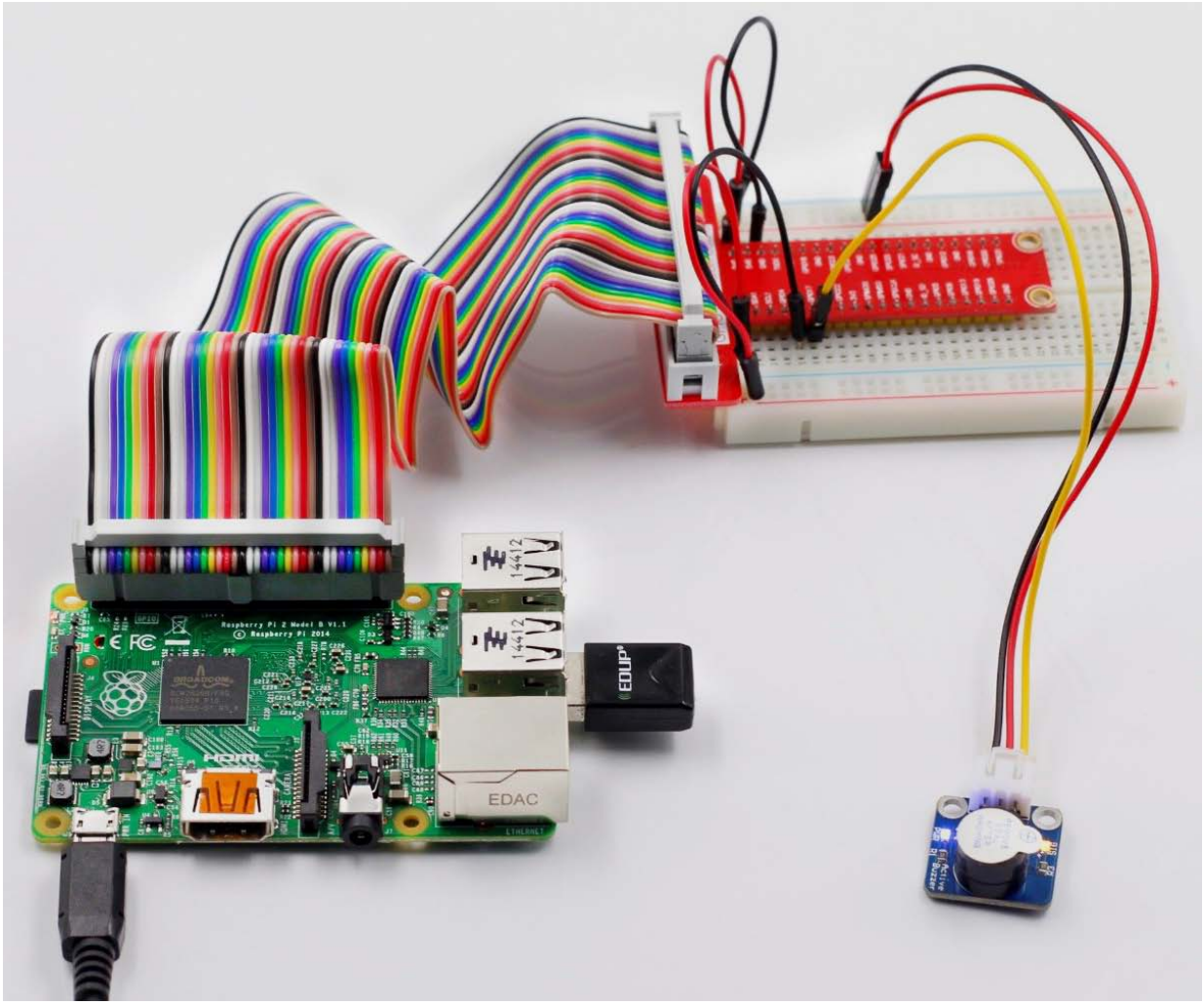
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 10_active_buzzer.py
```

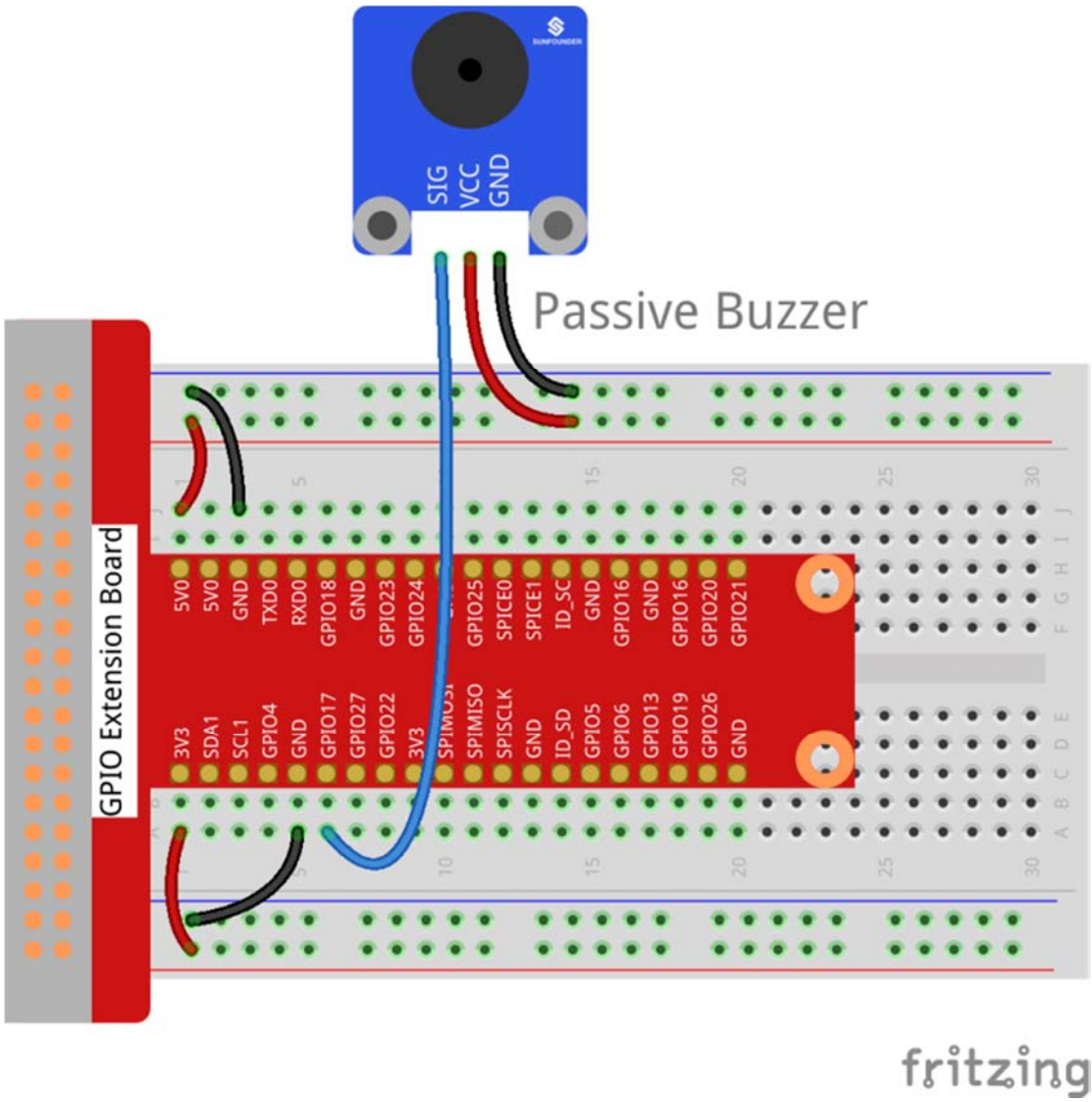
Now you can hear the active buzzer beeping.



Passive Buzzer

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	Passive Buzzer Module
GPIO0	GPIO17	SIG
5V	5V0	VCC
GND	GND	GND



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/10_passive_buzzer/
```

Step 3: Compile

```
gcc passive_buzzer.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

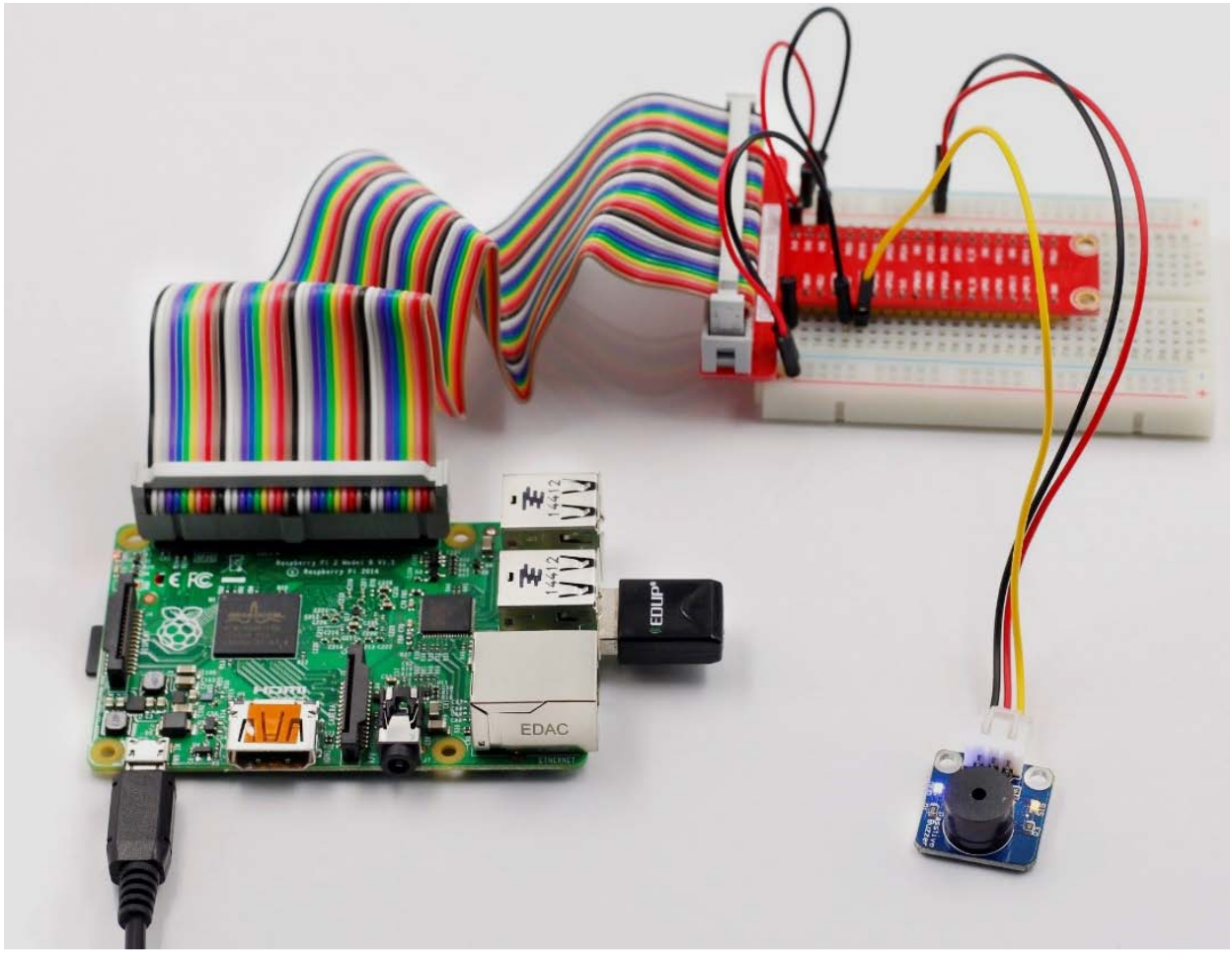
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 10_passive_buzzer.py
```

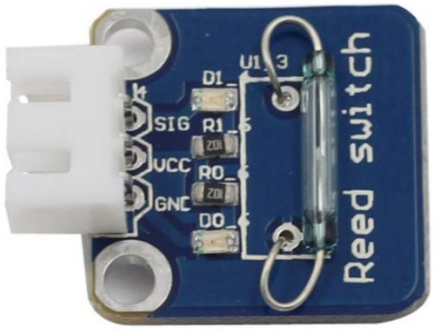
Now you can hear the passive buzzer playing music.



Lesson 11 Reed Switch

Introduction

A reed switch (as shown below) is used to detect the magnetic field. Hall sensors are generally used to measure the speed of intelligent vehicles and count in assembly lines, while reed switches are often used to detect the existence of a magnetic field.



Components

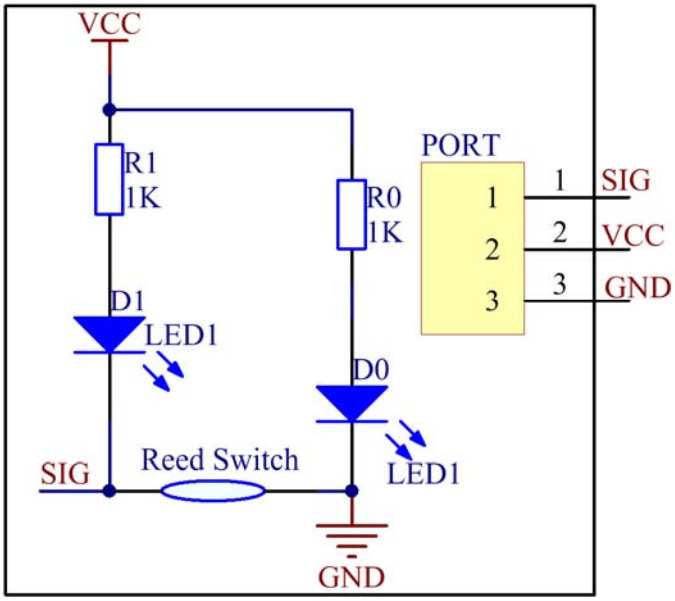
- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * Reed switch module
- 1 * Dual-color LED module
- 2 * 3-Pin anti-reverse cable
- 1 * Magnet (Self provided)

Experimental Principle

A reed switch is a type of line switch component that realizes control by magnetic signals. It induces by a magnet. The “switch" here means dry reed pipe, which is a kind of contact passive electronic switch component with the advantage of simple structure, small size, and convenient control. The shell of a reed switch is commonly a sealed glass pipe in which two iron elastic reed electroplates are equipped and inert gases are filled. Normally, the two reeds made of special materials in the glass tube are separated. However, when a magnetic substance approaches the glass tube, the two reeds in the glass tube are magnetized to attract each other and contact under the function of magnetic field lines. As a result, the two reeds will pull together to connect the circuit connected with the nodes.

After external magnetic force disappears, the two reeds will be separated with each other because they have the same magnetism, so the circuit is also disconnected. Therefore, as a line switch component controlling by magnetic signals, the dry reed pipe can be used as a sensor to count, limit positions and so on. At the same time, it is widely used in a variety of communication devices.

The schematic diagram:

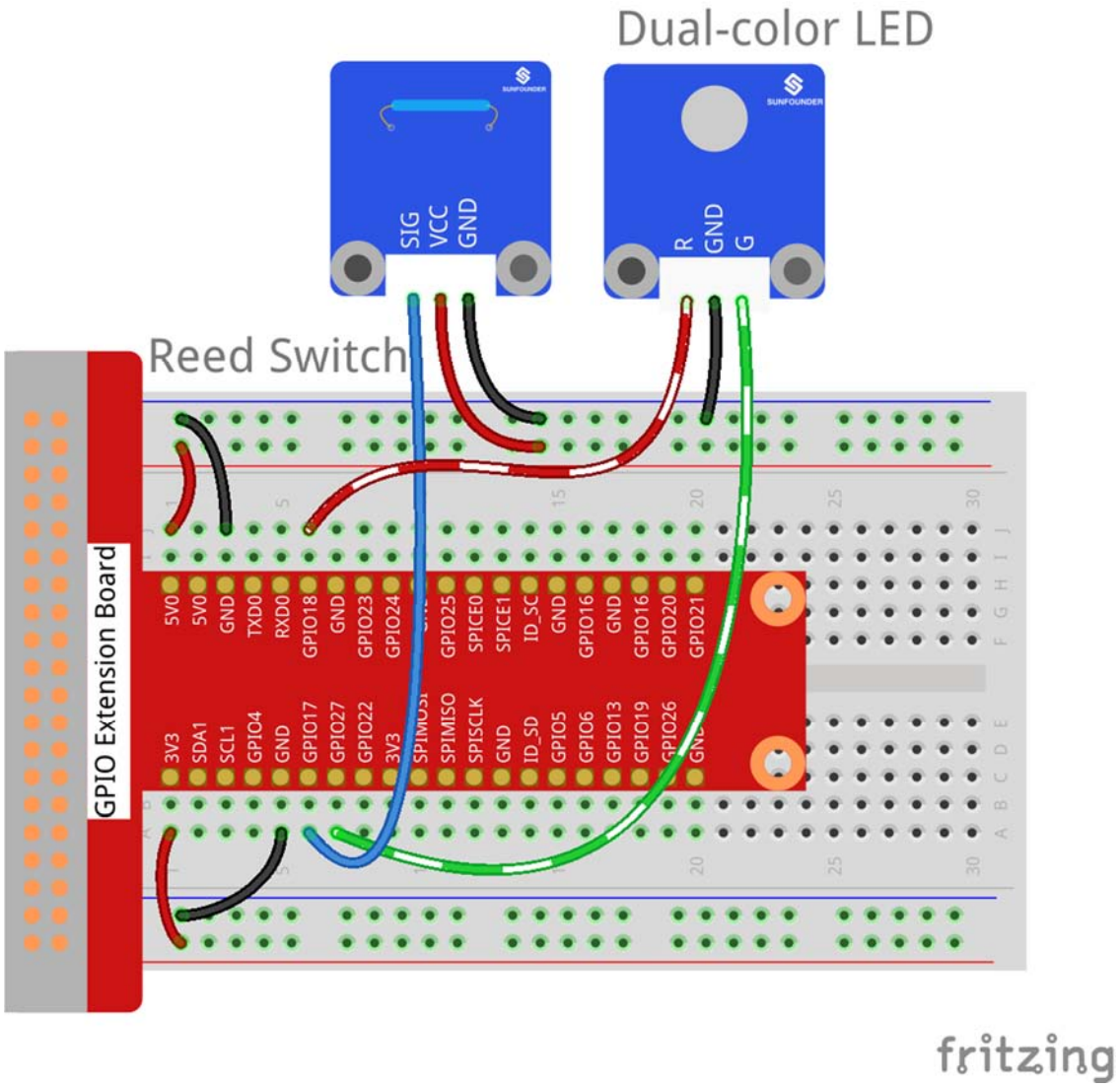


Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	Reed Switch Module
GPIO0	GPIO17	SIG
5V	5V0	VCC
GND	GND	GND

Raspberry Pi	T-Cobbler	Dual-color LED Module
GPIO1	GPIO18	R
GND	GND	GND
GPIO2	GPIO27	G



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/11_reed_switch/
```

Step 3: Compile

```
gcc reed_switch.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

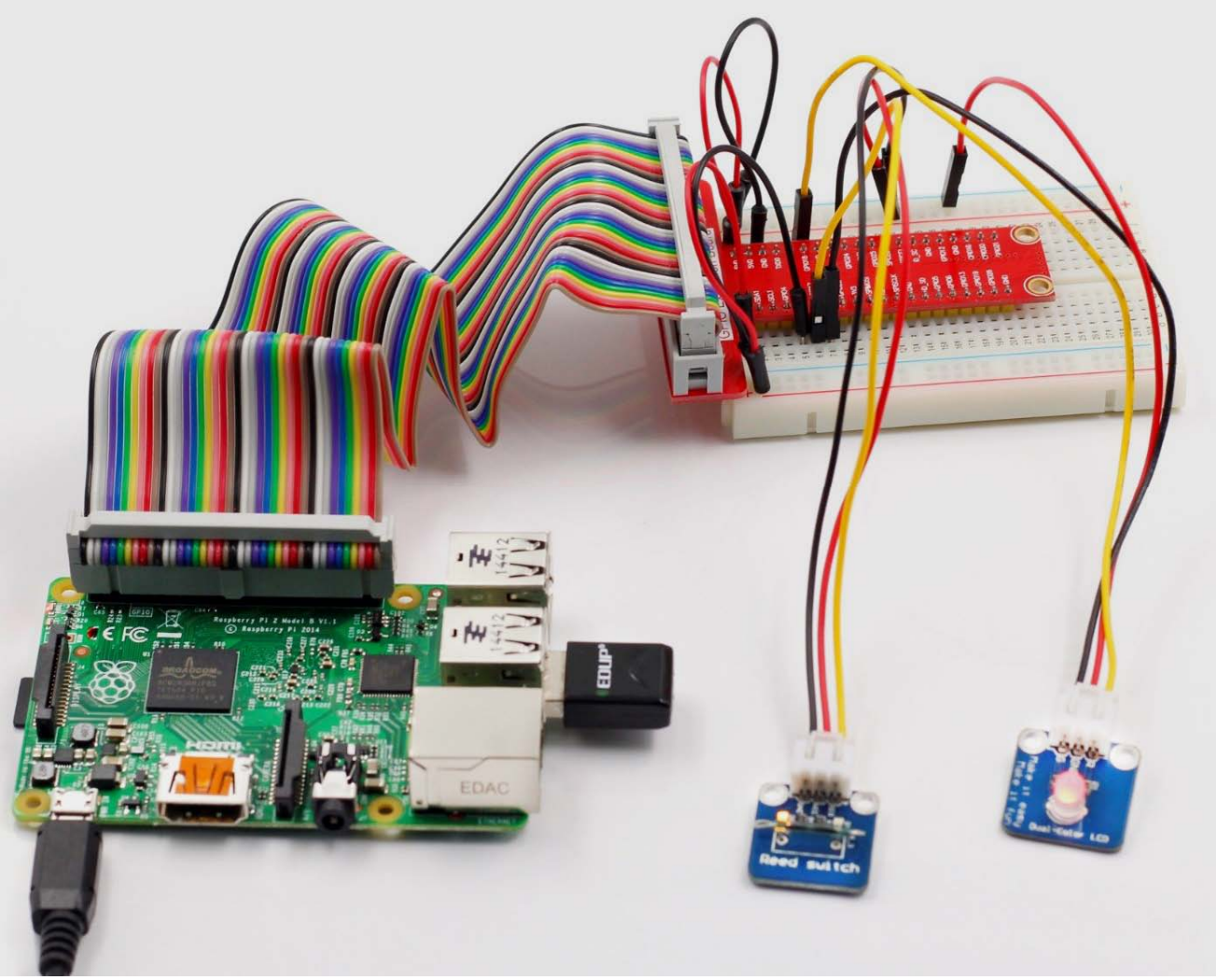
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 11_reed_switch.py
```

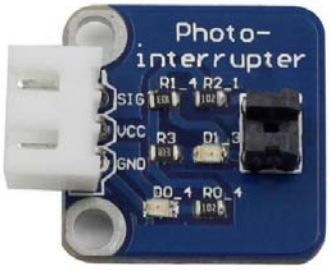
Then the LED will flash green. Place a magnet near the reed switch, "Detected Magnetic Material!" will be printed on the screen and the LED will change to red. Move away the magnet, the LED will turn green again.



Lesson 12 Photo-interrupter

Introduction

A photo-interrupter (as shown below) is a sensor with a light-emitting component and light-receiving component packaged and placed on face-to-face. It applies the principle that light is interrupted when an object passes through the sensor. Therefore, photo-interrupters are widely used in speed measurement.

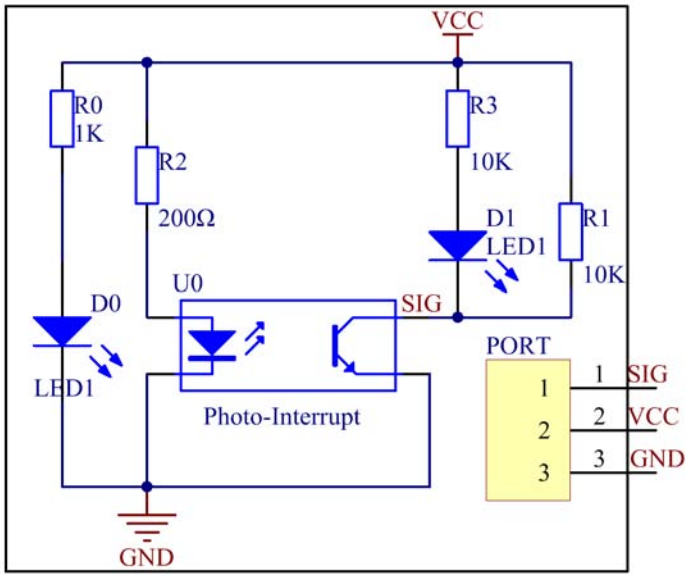


Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male)
- 1 * Network cable (or USB wireless network adapter)
- 1 * Dual-color LED module
- 1 * Photo-interrupter module
- 2 * 3-Pin anti-reverse cable

Experimental Principle

Basically a photo-interrupter consists of two parts: transmitter and receiver. The transmitter (e.g., an LED or a laser) emits light and then the light goes to the receiver. If that light beam between the transmitter and receiver is interrupted by an obstacle, the receiver will detect no incident light even for a moment and the output level will change. In this experiment, we will turn an LED on or off by using this change. The schematic diagram:

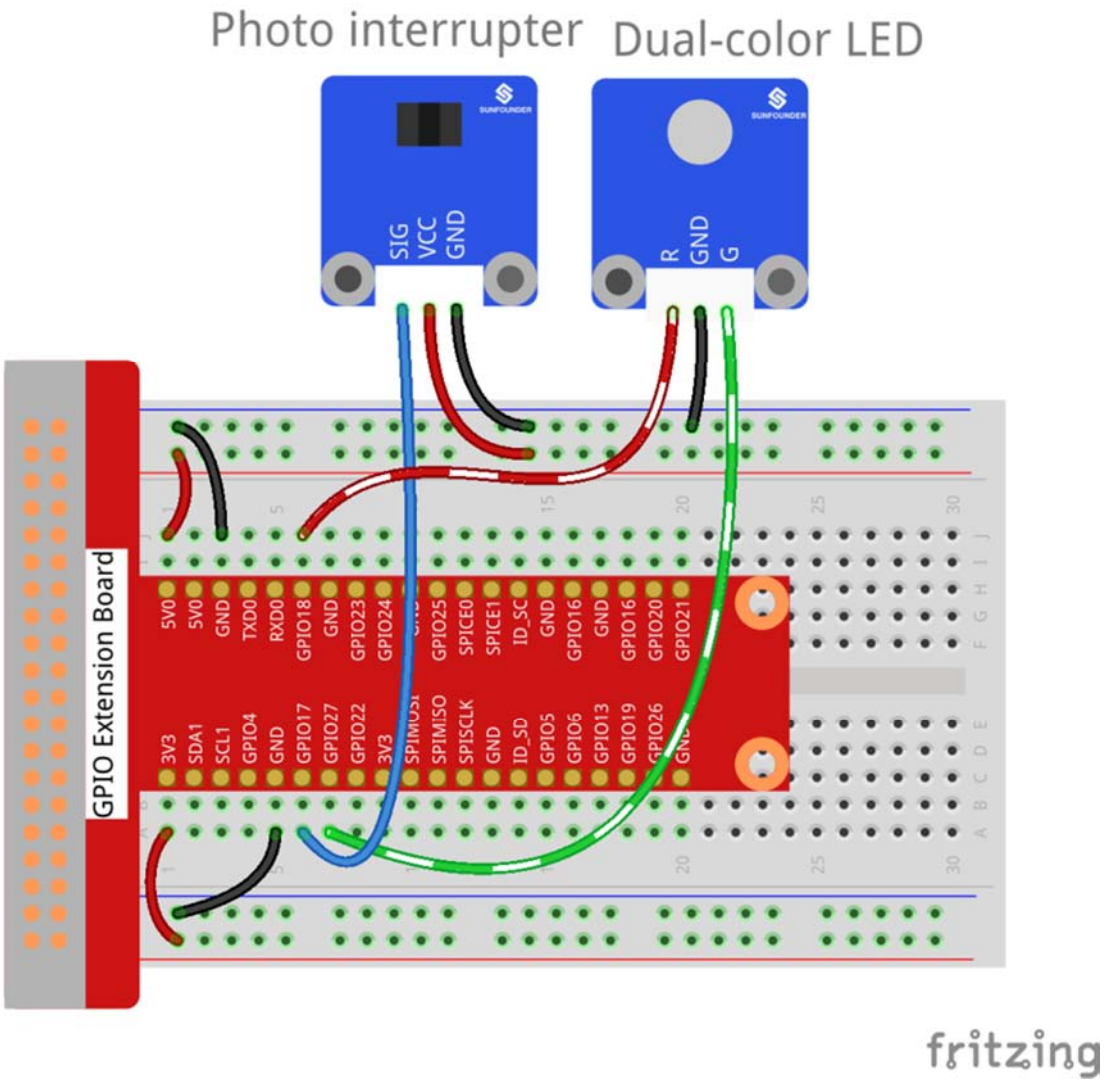


Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	Photo-interrupter Module
GPIO0	GPIO17	SIG
5V	5V0	VCC
GND	GND	GND

Raspberry Pi	T-Cobbler	Dual-color LED Module
GPIO1	GPIO18	R
GND	GND	GND
GPIO2	GPIO27	G



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/12_photo_interrupter/
```

Step 3: Compile

```
gcc photo_interrupter.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

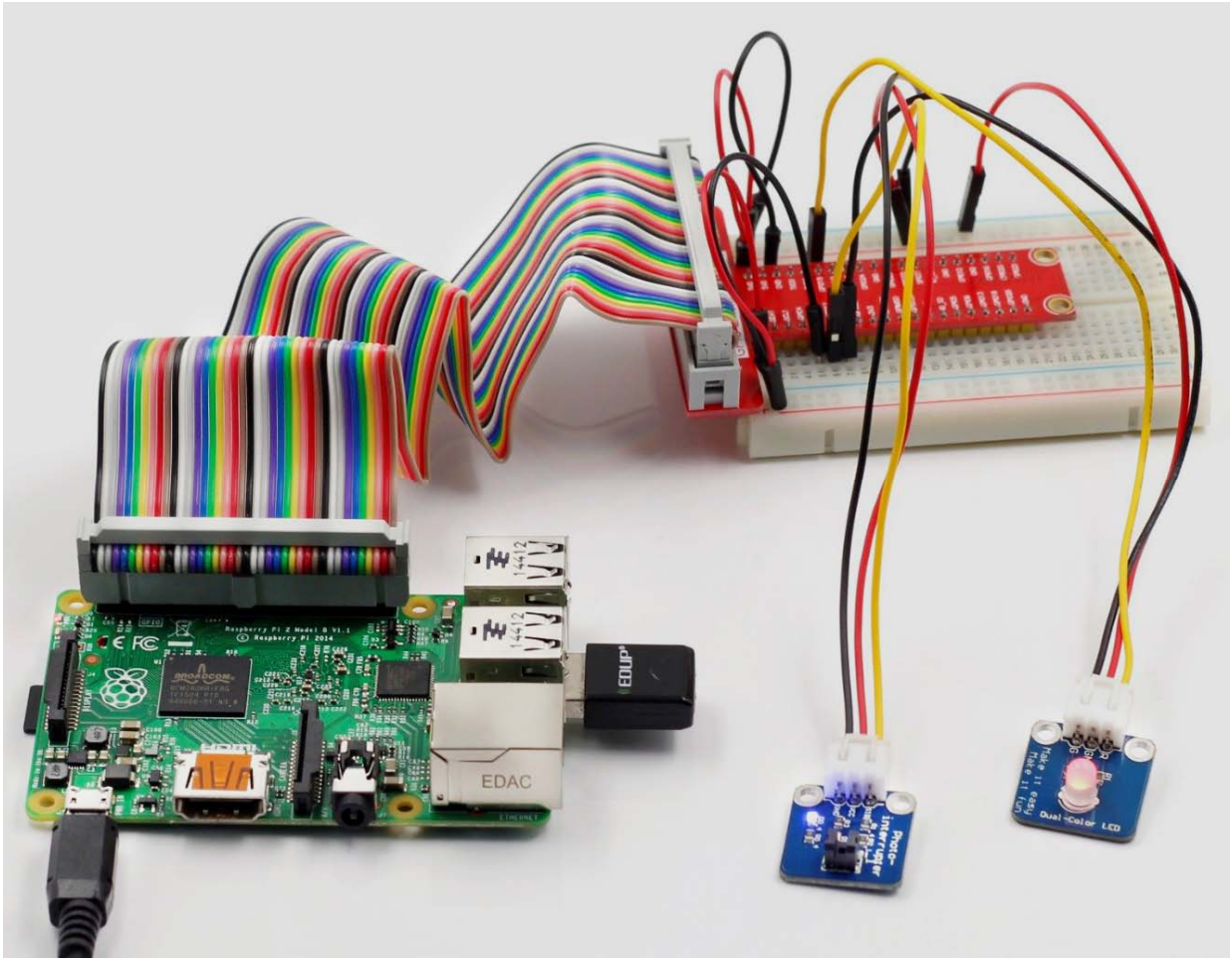
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 12_photo_interrupter.py
```

Now the LED will light up green. Stick a piece of paper in the gap of photo interrupter. Then "Light was blocked" will be printed on the screen and the LED will flash red. Remove the paper, and the LED will turn green again.



Lesson 13 PCF8591

Introduction

The PCF8591 is a single-chip, single-supply low-power 8-bit CMOS data acquisition device with four analog inputs, one analog output and a serial I2C-bus interface. Three address pins A0, A1 and A2 are used for programming the hardware address, allowing the use of up to eight devices connected to the I2C-bus without additional hardware. Address, control and data to and from the device are transferred serially via the two-line bidirectional I2C-bus.

The functions of the device include analog input multiplexing, on-chip track and hold function, 8-bit analog-to-digital conversion and an 8-bit digital-to-analog conversion. The maximum conversion rate is given by the maximum speed of the I2C-bus.

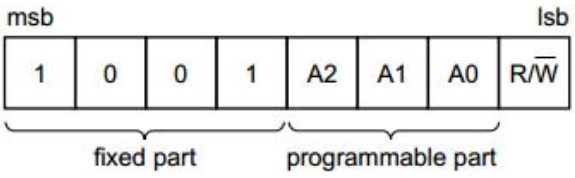
Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male)
- 1 * Network cable (or USB wireless network adapter)
- 1 * PCF8591 module
- 1 * Dual-Color LED module
- 1 * 3-Pin anti-reverse cable
- Several Jumper wires (Male to Female)

Experimental Principle

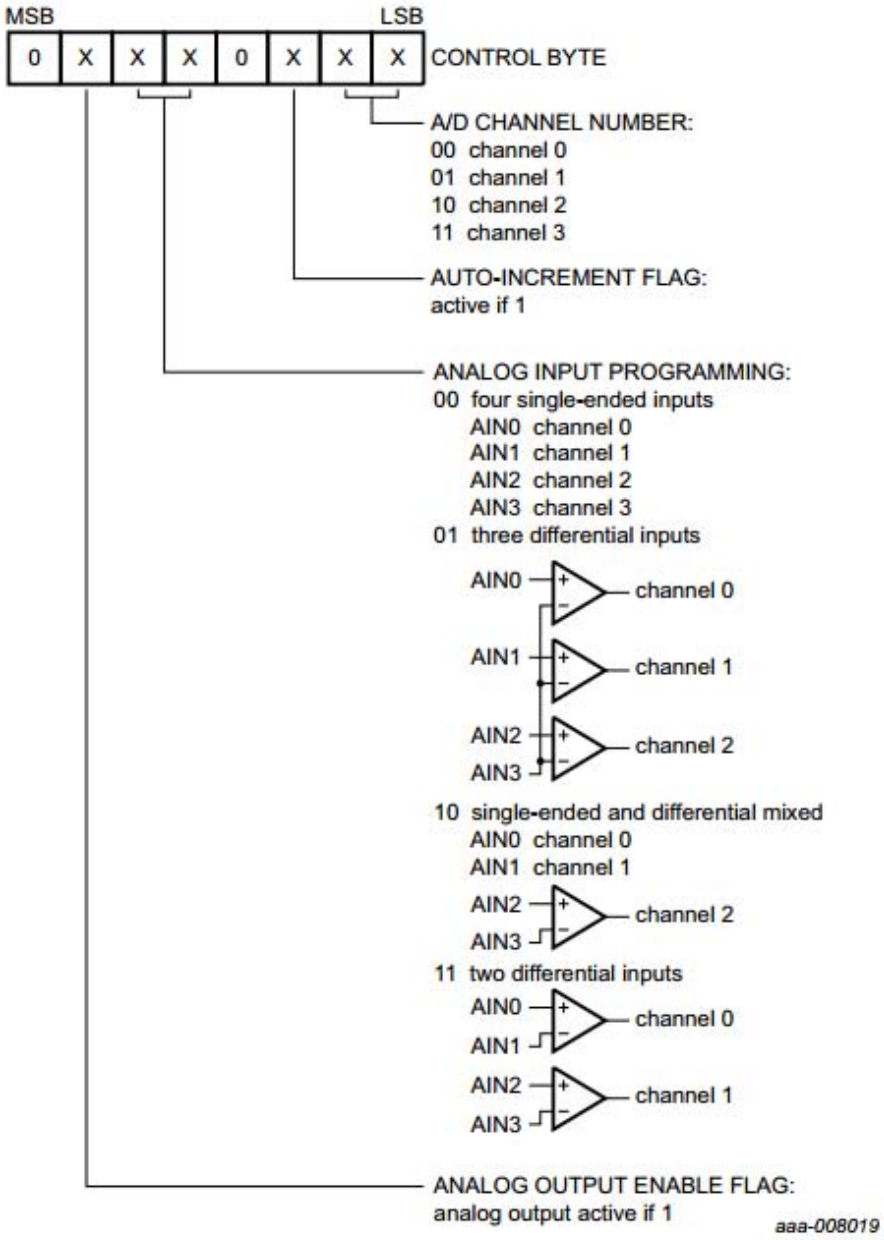
Addressing:

Each PCF8591 device in an I2C-bus system is activated by sending a valid address to the device. The address consists of a fixed part and a programmable part. The programmable part must be set according to the address pins A0, A1 and A2. The address always has to be sent as the first byte after the start condition in the I2C-bus protocol. The last bit of the address byte is the read/write-bit which sets the direction of the following data transfer (see as below).



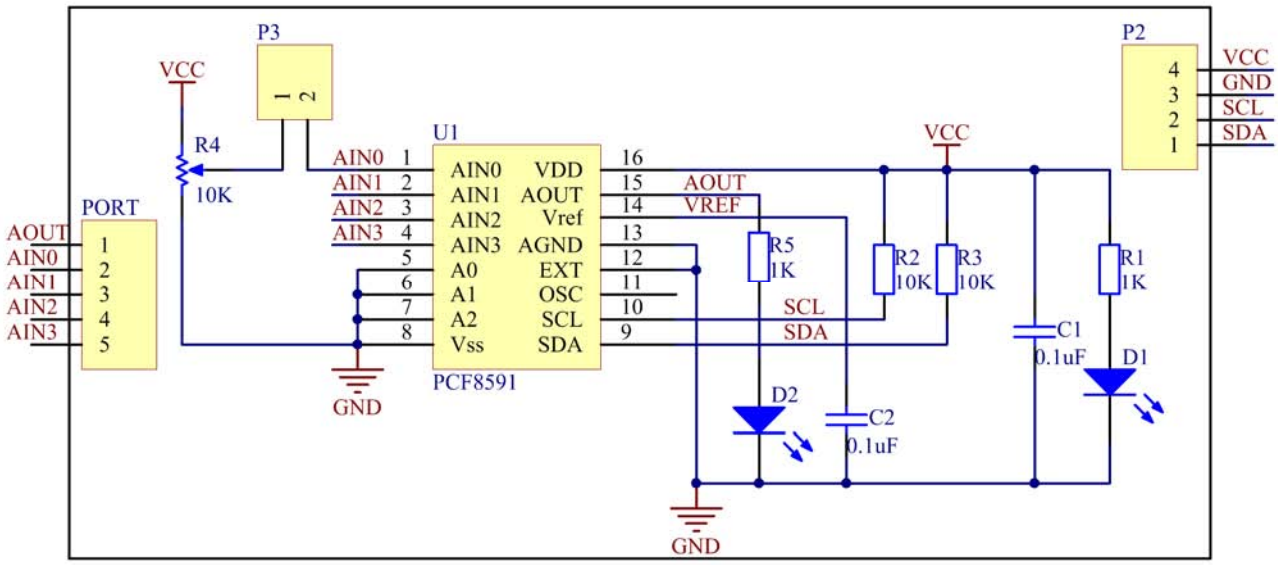
Control byte:

The second byte sent to a PCF8591 device will be stored in its control register and is required to control the device function. The upper nibble of the control register is used for enabling the analog output, and for programming the analog inputs as single-ended or differential inputs. The lower nibble selects one of the analog input channels defined by the upper nibble (see Fig.5). If the auto-increment flag is set, the channel number is incremented automatically after each A/D conversion. See the figure below.



In this experiment, the AIN0 (Analog Input 0) port is used to receive analog signals from the potentiometer module, and AOUT (Analog Output) is used to output analog signals to the dual-color LED module so as to change the luminance of the LED.

The schematic diagram:



Experimental Procedures

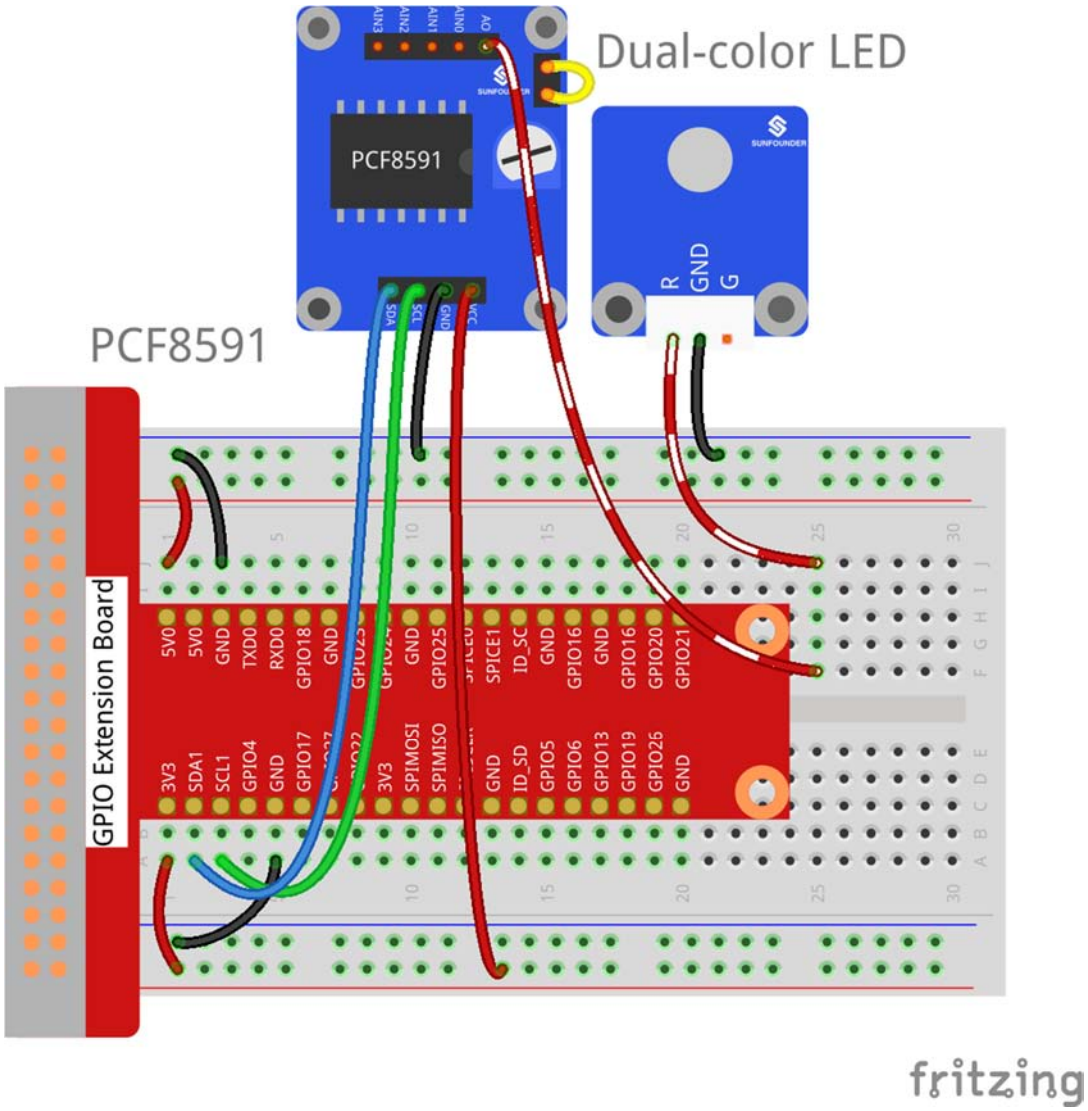
Step 1: Build the circuit

Raspberry Pi	T-Cobbler	PCF8591 Module
SDA	SDA1	SDA
SCL	SCL1	SCL
3V3	3V3	VCC
GND	GND	GND

Dual-Color Module	T-Cobbler	PCF8591 Module
R	*	AOUT
GND	GND	GND
G	*	*

Note:

Connect the two pins next to the potentiometer of the PCF8591 module with the yellow jumper cap attached.



Step 2: Setup I2C (see Appendix 1. If you have set I2C, skip this step.)

For C language users:

Step 3: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/13_pcf8591/
```

Step 4: Compile

```
gcc pcf8591.c -lwiringPi
```

Step 5: Run

```
sudo ./a.out
```

For Python users:

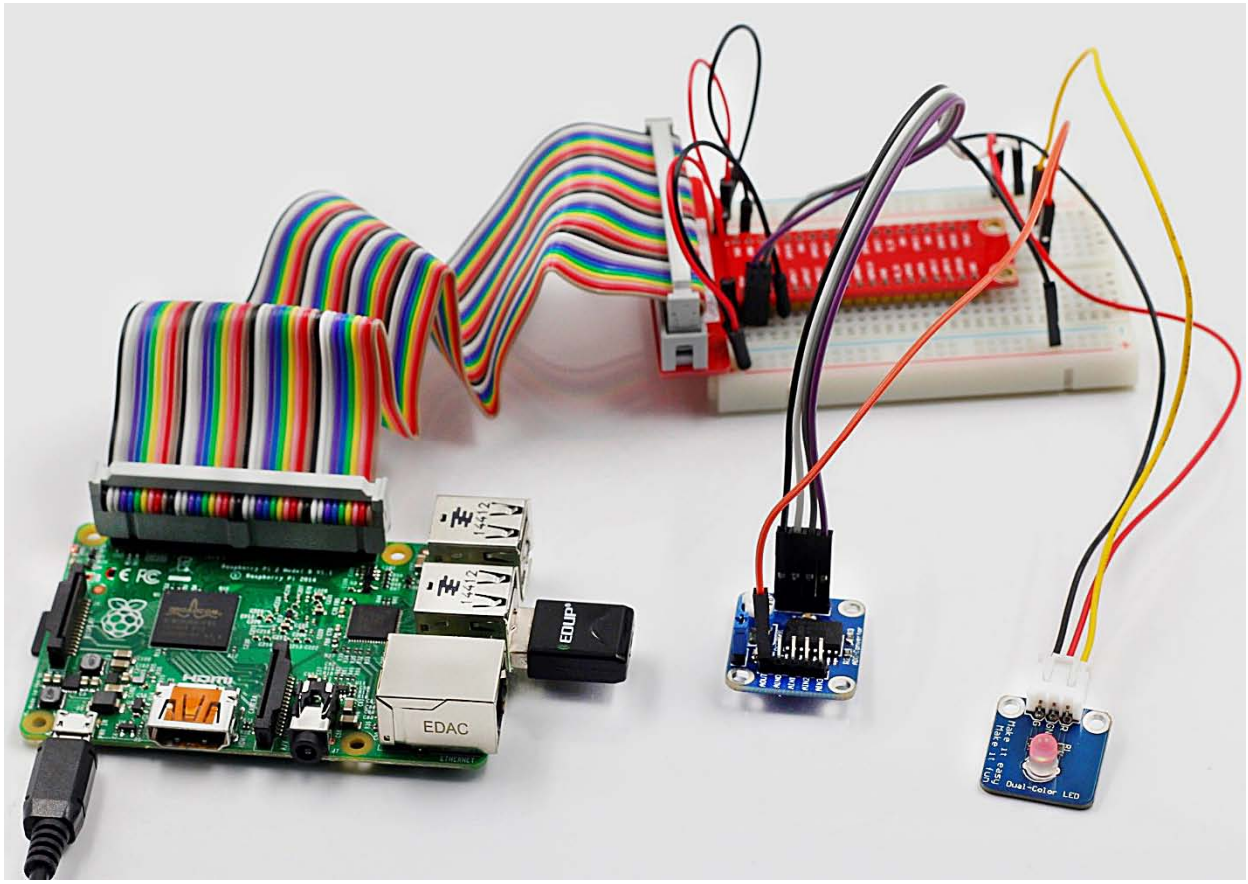
Step 3: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 4: Run

```
sudo python 13_pcf8591.py
```

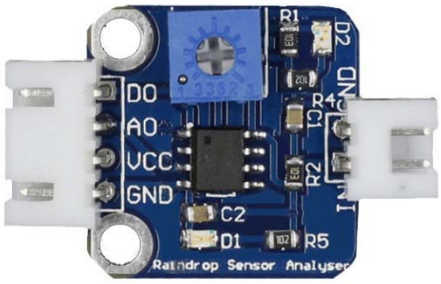
Now, turn the knob of the potentiometer on PCF8591, and you can see the luminance of the LED change and a value between 0 and 255 printed on the screen.



Lesson 14 Rain Detection Module

Introduction

The rain detection module detects rain on the board. Place the rain detection board in the open air. When it is raining, the rain detection module will sense the raindrops and send signals to the Raspberry Pi.



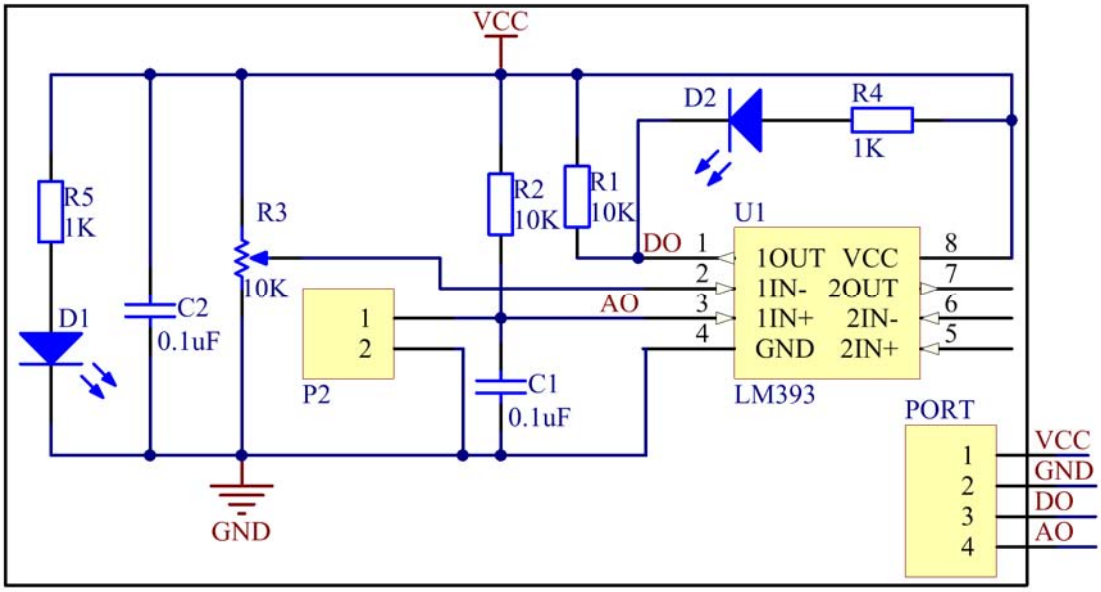
Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * Rain Detection module
- 1 * PCF8591
- 1 * LM393
- 1 * 2-Pin ribbon cable
- 1 * 4-Pin anti-reverse cable
- Several Jumper wires (Male to Female)
- 1 * glass of water (Self provided)

Experimental Principle

There are two metal wires that are close to each other but do not cross on the rain detection board. When rain drops on the board, the two metal wires will conduct, thus there is a voltage between the two metal wires.

The schematic diagram:



Experimental Procedures

Step 1: Build the circuit

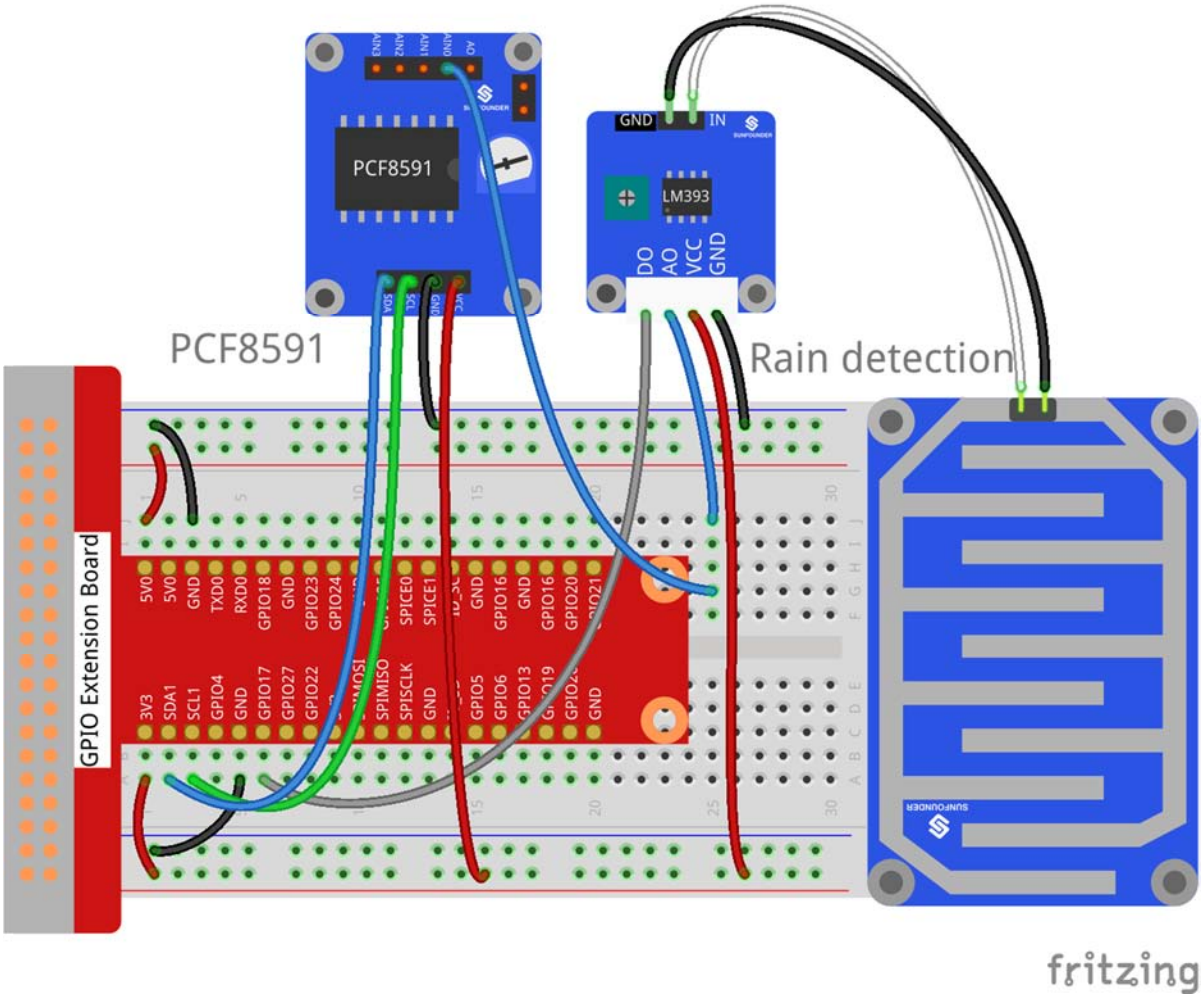
Raspberry Pi	T-Cobbler	PCF8591 Module
SDA	SDA1	SDA
SCL	SCL1	SCL
3V3	3V3	VCC
GND	GND	GND

LM393	T-Cobbler	PCF8591 Module
DO	GPIO17	*
AO	*	AIN0
VCC	3V3	VCC
GND	GND	GND

Rain Detection Board	LM393
-	IN
-	GND

Note:

The two pins on the rain detection board are exactly the same. You can connect them to pin IN and GND on LM393.



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/14_rain_detector/
```

Step 3: Compile

```
gcc rain_detector.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

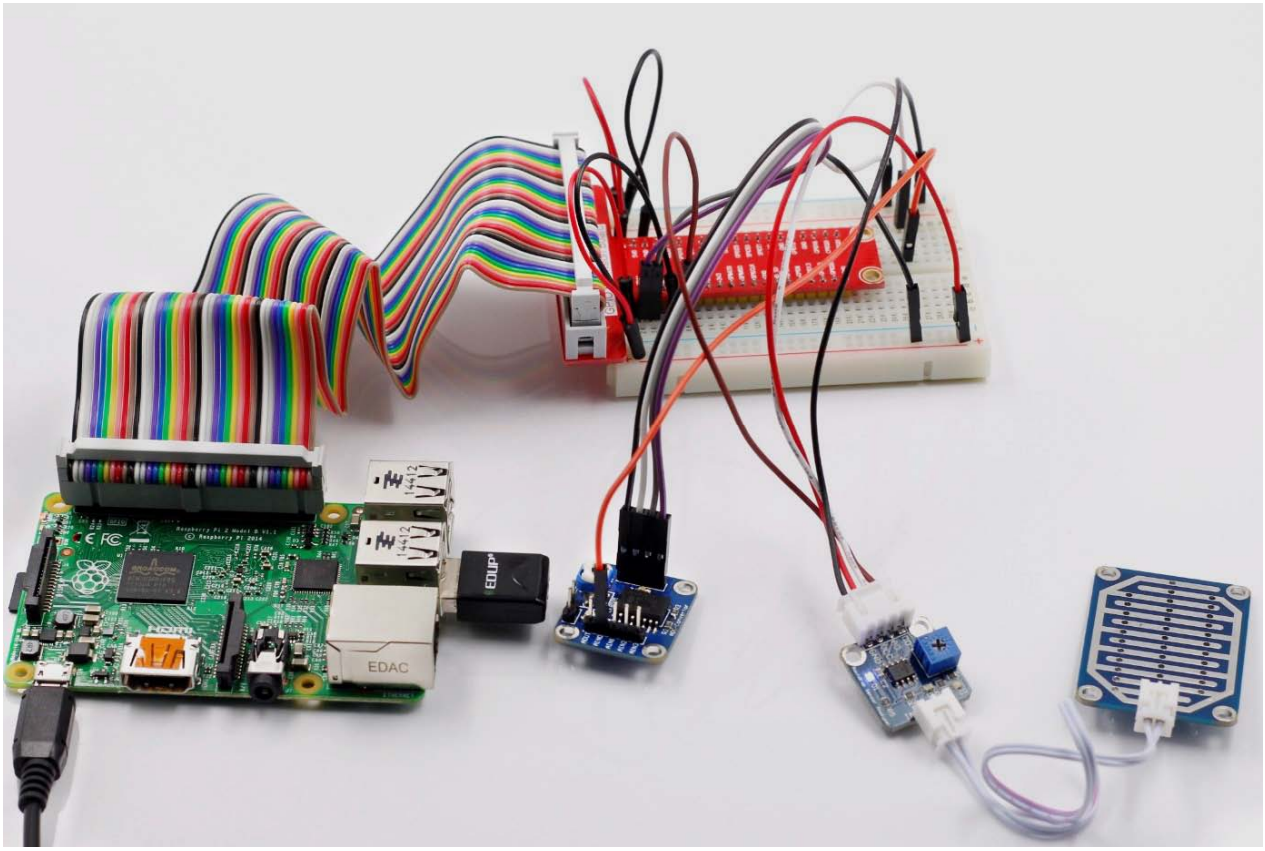
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 14_rain_detector.py
```

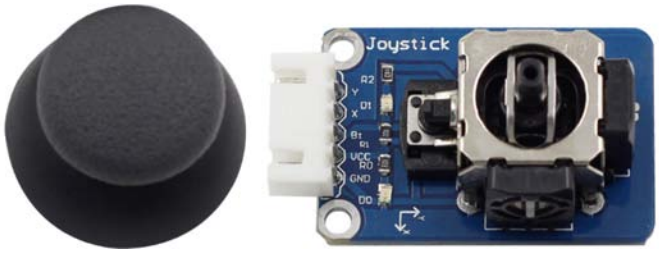
Now drop some water onto the rain detection board until "**raining**" displayed on the screen. You can adjust the potentiometer on LM393 to detect the threshold of rainfall.



Lesson 15 Joystick PS2

Introduction

There are five operation directions for joystick PS2: up, down, left, right and press-down.



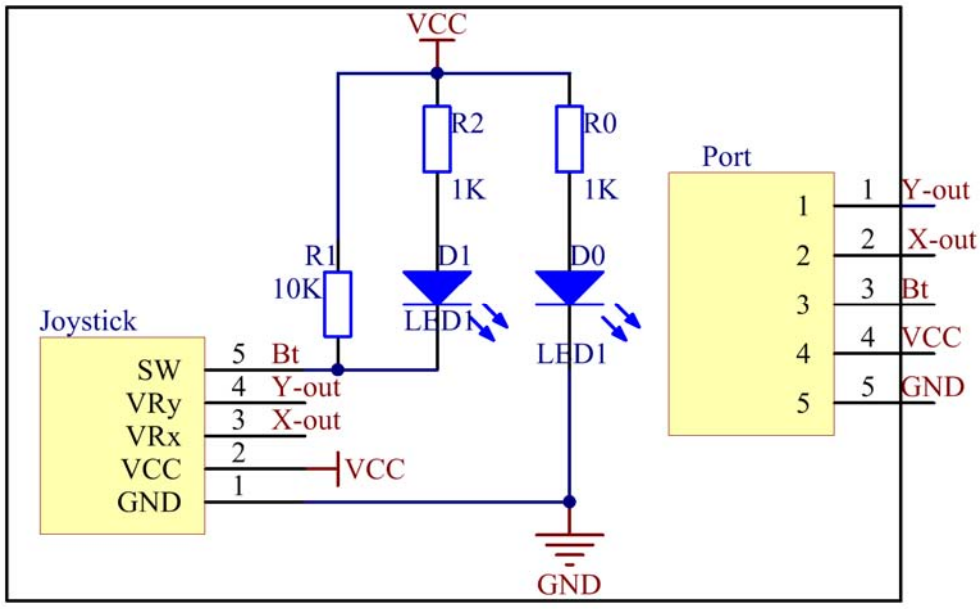
Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * PCF8591
- 1 * Joystick PS2 module
- 1 * 5-Pin anti-reverse cable
- Several Jumper wires (Male to Female)

Experimental Principle

This module has two analog outputs (corresponding to **X** and **Y** coordinates) and one digital output representing whether it is pressed on Z axis.

In this experiment, we connect pin X and Y to the analog input ports of the A/D convertor so as to convert analog quantities into digital ones. Then program on Raspberry Pi to detect the moving direction of the Joystick. The schematic diagram:

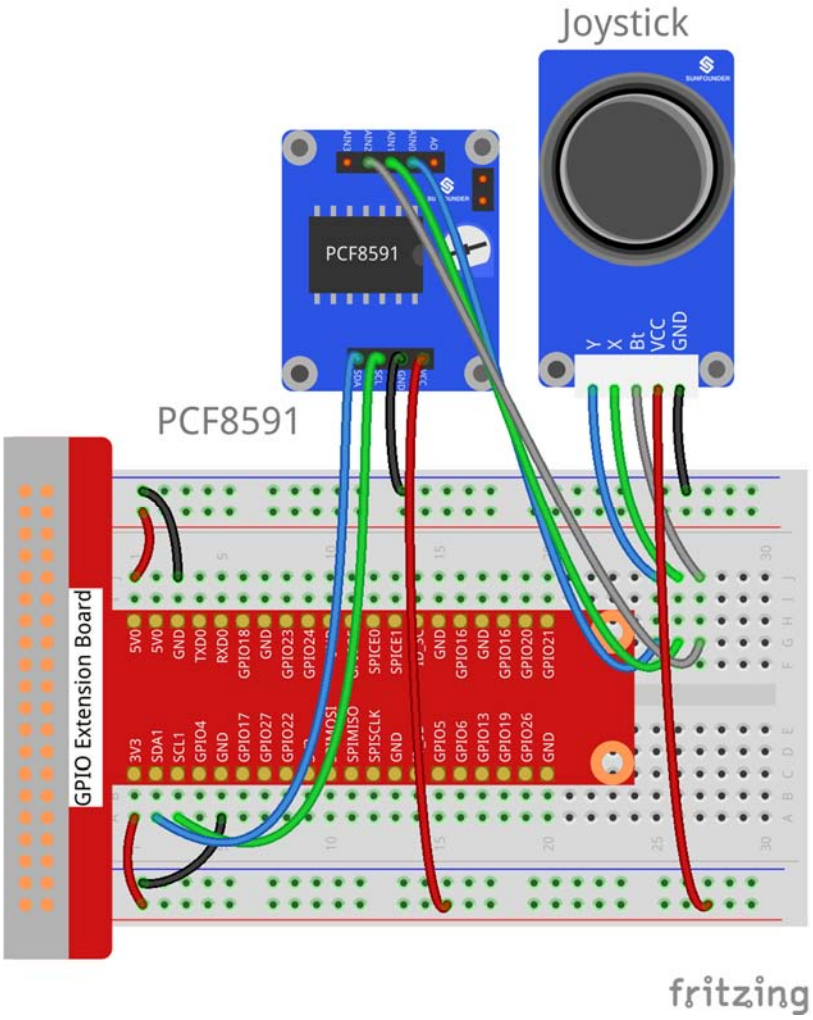


Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	PCF8591 Module
SDA	SDA1	SDA
SCL	SCL1	SCL
3.3V	3V3	VCC
GND	GND	GND

Joystick PS2	T-Cobbler	PCF8591 Module
Y	*	AIN0
X	*	AIN1
Bt	*	AIN2
VCC	3V3	VCC
GND	GND	GND



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/15_joystick_PS2/
```

Step 3: Compile

```
gcc joystick_PS2.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

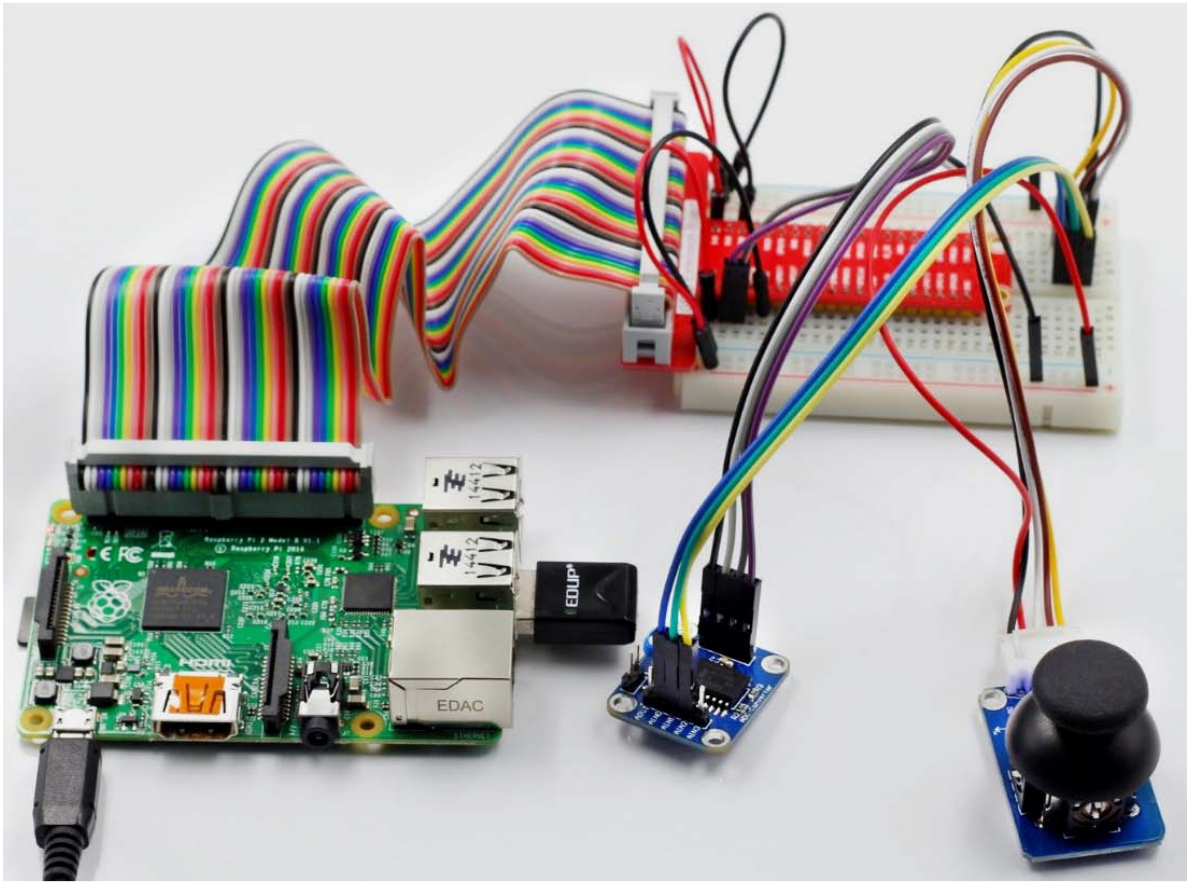
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 15_joystick_PS2.py
```

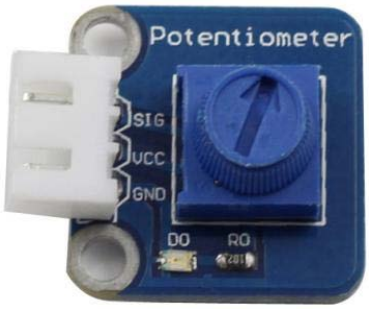
Now push the rocker upwards, and a string "**up**" will be printed on the screen; push it downwards, and "**down**" will be printed; if you push it left, "**Left**" will be printed on; If you push it right, and "**Right**" will be printed; If you press down the cap, "**Button Pressed**" will be printed on the screen.



Lesson 16 Potentiometer Module

Introduction

A potentiometer is a device which is used to vary the resistance in an electrical circuit without interrupting the circuit.



Components

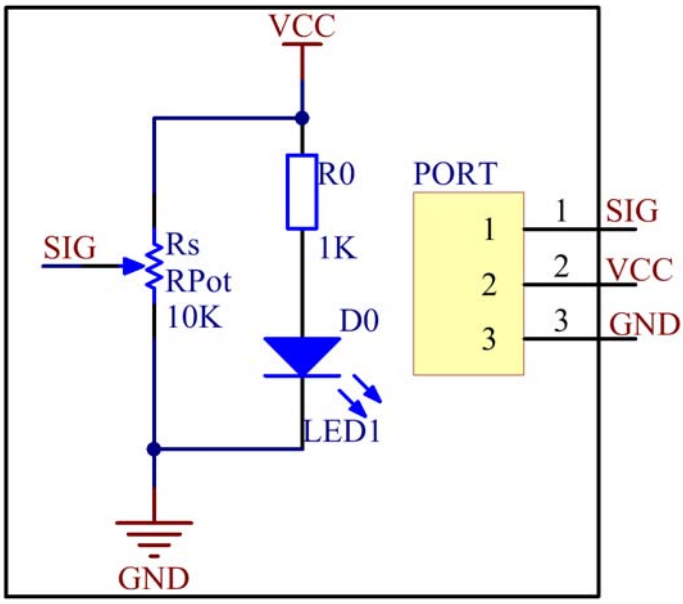
- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * Potentiometer module
- 1 * Dual-Color LED module
- 2 * 3-Pin anti-reverse cable
- Several Jumper wires (Male to Female)

Experimental Principle

An analog potentiometer is an analog electronic component. What's the difference between an analog one and a digital one? Simply put, a digital potentiometer refers to just two states like on/off, high/low levels, i.e. either 0 or 1, while a digital one supports analog signals like a number from 1 to 1000. The signal value changes over time instead of keeping an exact number. Analog signals include light intensity, humidity, temperature, and so on.

In this experiment, PCF8591 is used to read the analog value of the potentiometer and output the value to LED. Connect pin SIG of the potentiometer to pin AIN0 of PCF8591. Connect pin R or Pin G of the Dual-Color LED to pin AOUT of PCF8591 to observe the change of LED.

The schematic diagram:



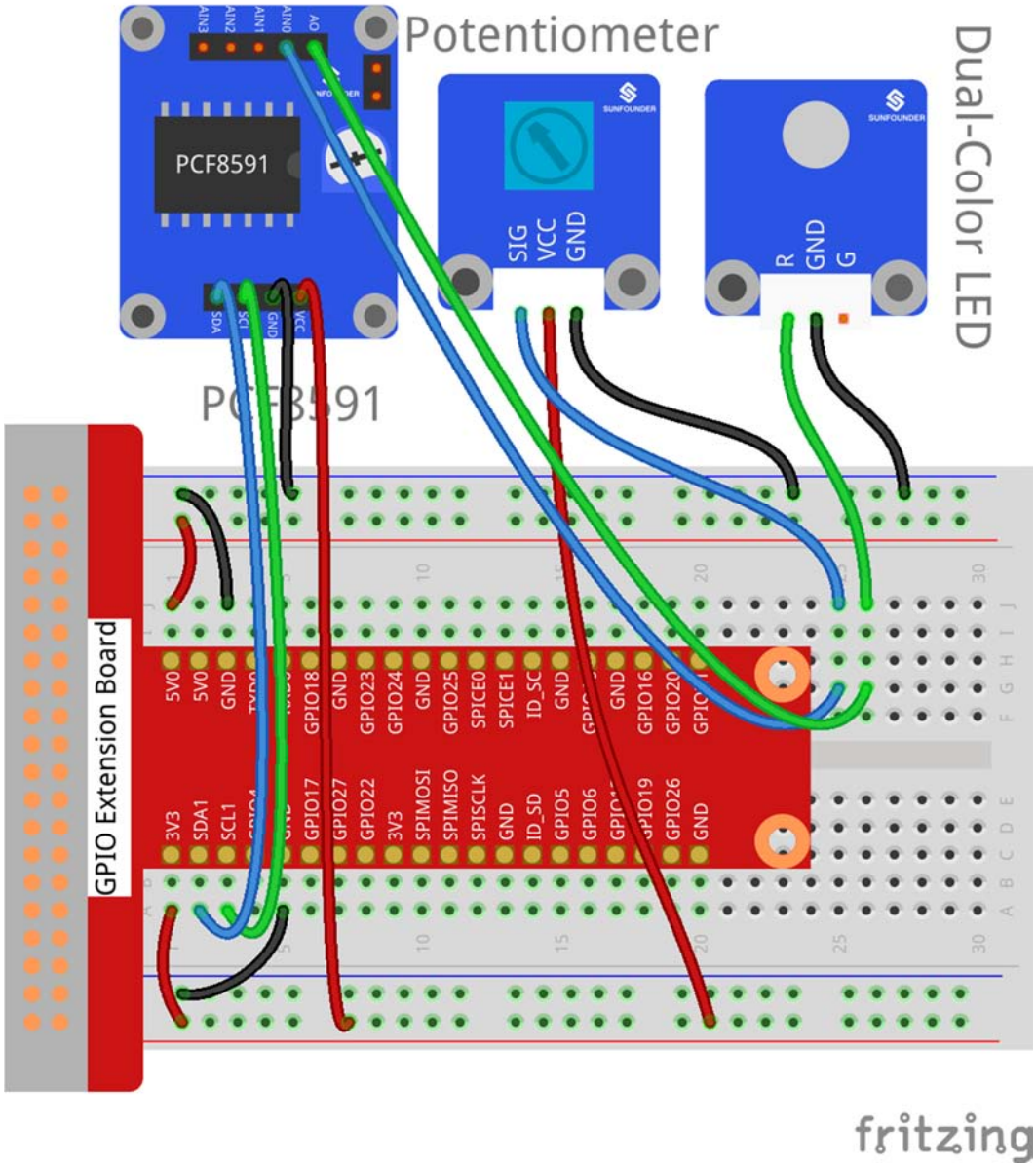
Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	PCF8591 Module
SDA	SDA1	SDA
SCL	SCL1	SCL
3V3	3V3	VCC
GND	GND	GND

Potentiometer	T-Cobbler	PCF8591 Module
SIG	*	AIN0
VCC	3V3	VCC
GND	GND	GND

Dual-Color Module	T-Cobbler	PCF8591 Module
R	*	AOUT
GND	GND	GND
G	*	*



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/16_potentiometer/
```

Step 3: Compile

```
gcc potentiometer.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

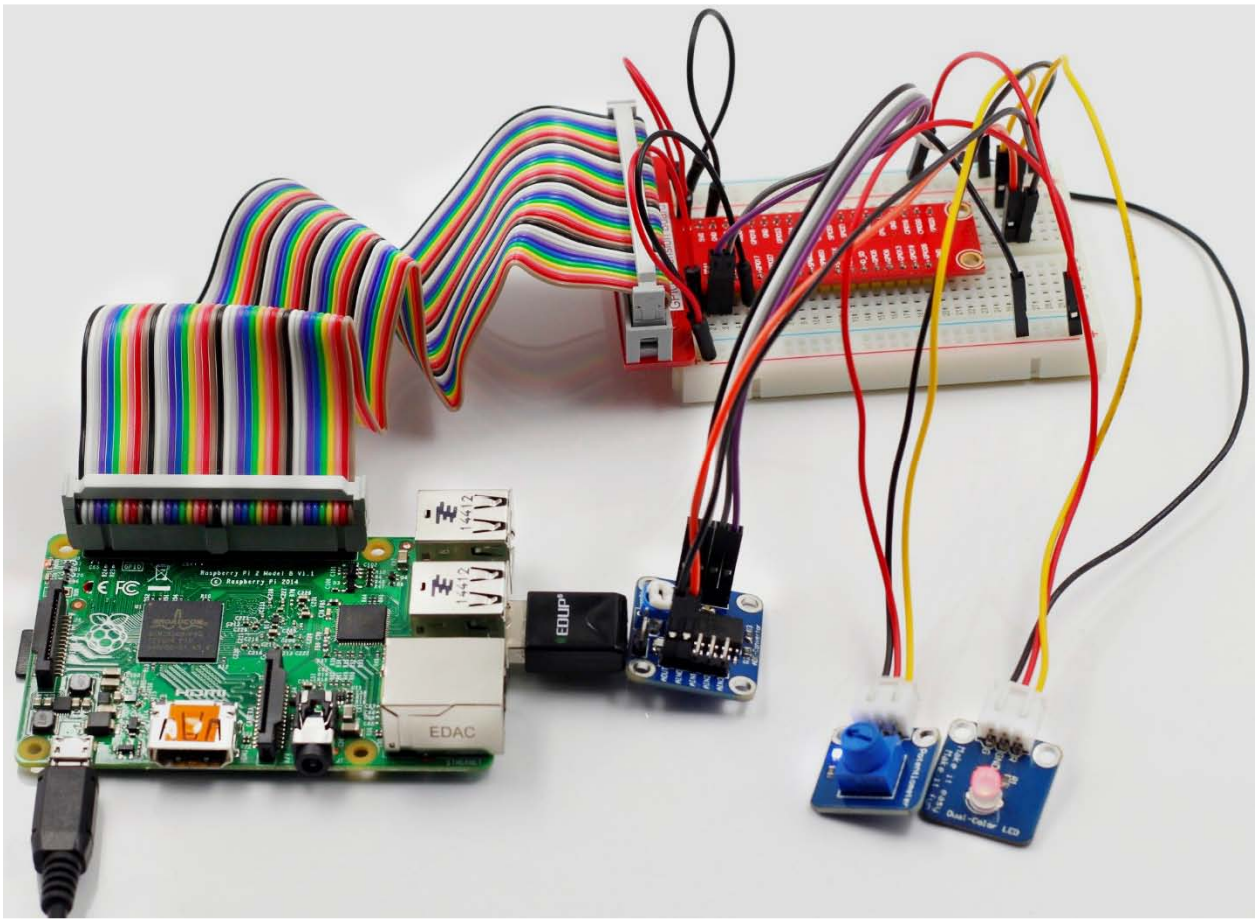
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 16_potentiometer.py
```

Turn the knob of the potentiometer, and you can see the value printed on the screen change from 0 (minimum) to 255 (maximum).

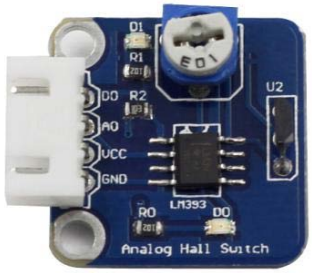


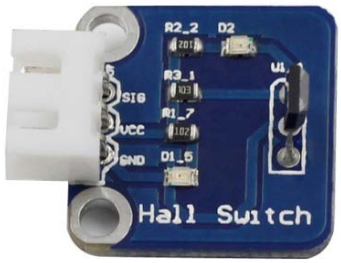
Lesson 17 Hall Sensor

Introduction

Based on Hall Effect, a Hall sensor is a one that varies its output voltage in response to a magnetic field. Hall sensors are used for proximity switching, positioning, speed detection, and current sensing applications.

Hall sensors can be categorized into linear (analog) Hall sensors and switch Hall sensors. A switch Hall sensor consists of voltage regulator, Hall element, differential amplifier, Schmitt trigger, and output terminal and it outputs digital values. A linear Hall sensor consists of Hall element, linear amplifier, and emitter follower and it outputs analog values. If you add a comparator to a linear (analog) Hall sensor it will be able to output both analog and digital signals.





Components

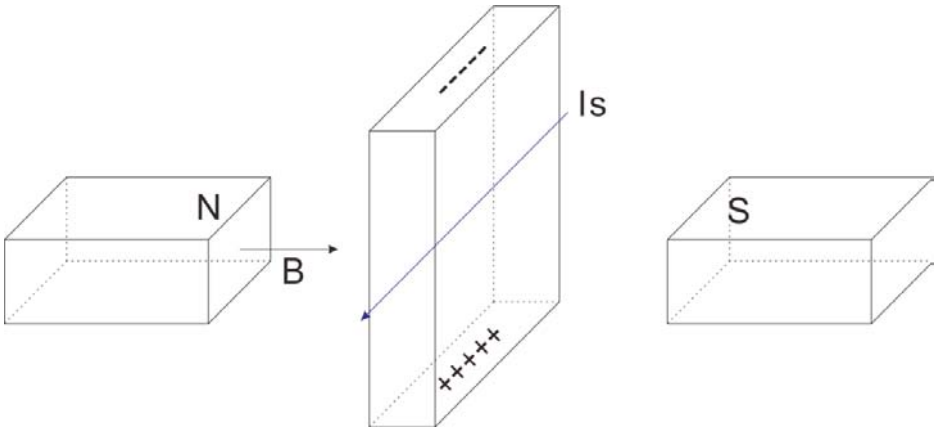
- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * Analog Hall Switch module
- 1 * Dual-color LED module
- 1 * Switch hall module
- 1 * PCF8591
- 2 * 3-Pin anti-reverse cable
- 1 * 4-Pin anti-reverse cable
- Several Jumper wires (Male to Female)

Experimental Principles

Hall Effect

Hall Effect is a kind of electromagnetic effect. It was discovered by Edwin Hall in 1879 when he was researching conductive mechanism about metals. The effect is seen when a conductor is passed through a uniform magnetic field. The natural electron drift of the charge carriers causes the magnetic field to apply a Lorentz force (the force exerted on a charged particle in an electromagnetic field) to these charge carriers. The result is what is

seen as a charge separation, with a buildup of either positive or negative charges on the bottom or on the top of the plate.

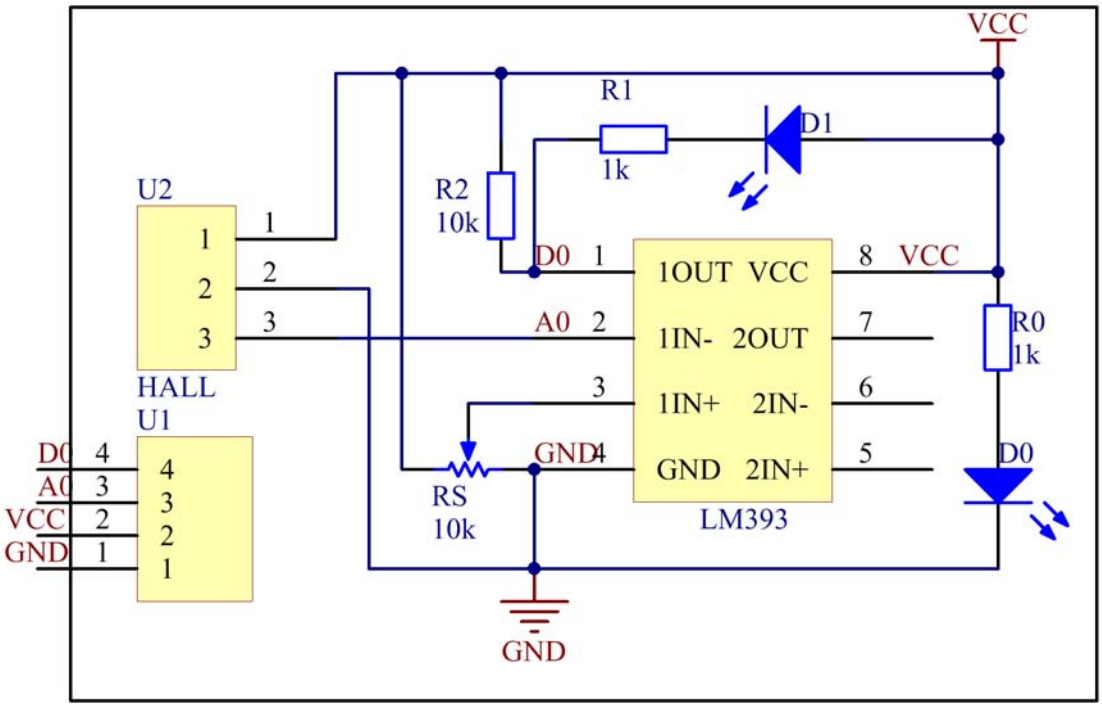


Hall sensor

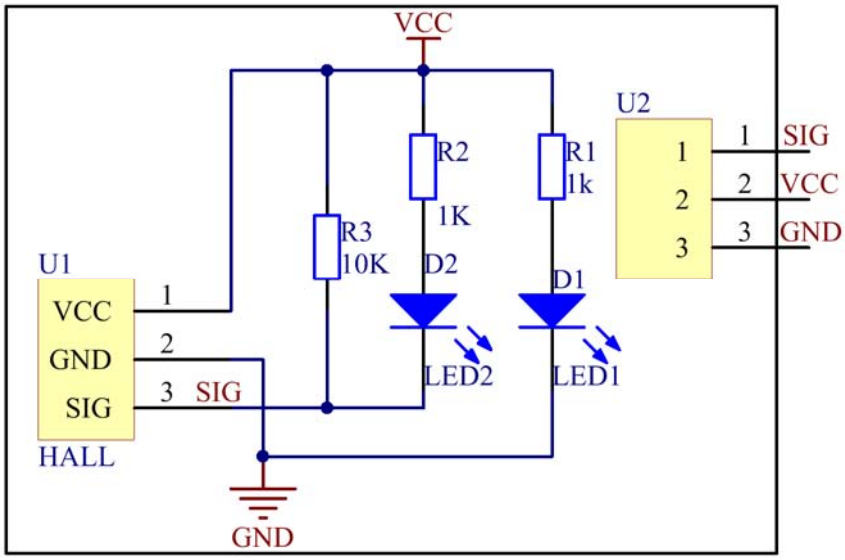
A Hall sensor is a kind of magnetic field sensor based on it.

Electricity carried through a conductor will produce a magnetic field that varies with current, and a Hall sensor can be used to measure the current without interrupting the circuit. Typically, the sensor is integrated with a wound core or permanent magnet that surrounds the conductor to be measured.

The schematic diagram of the analog Hall sensor module:



The schematic diagram of the Switch hall module:



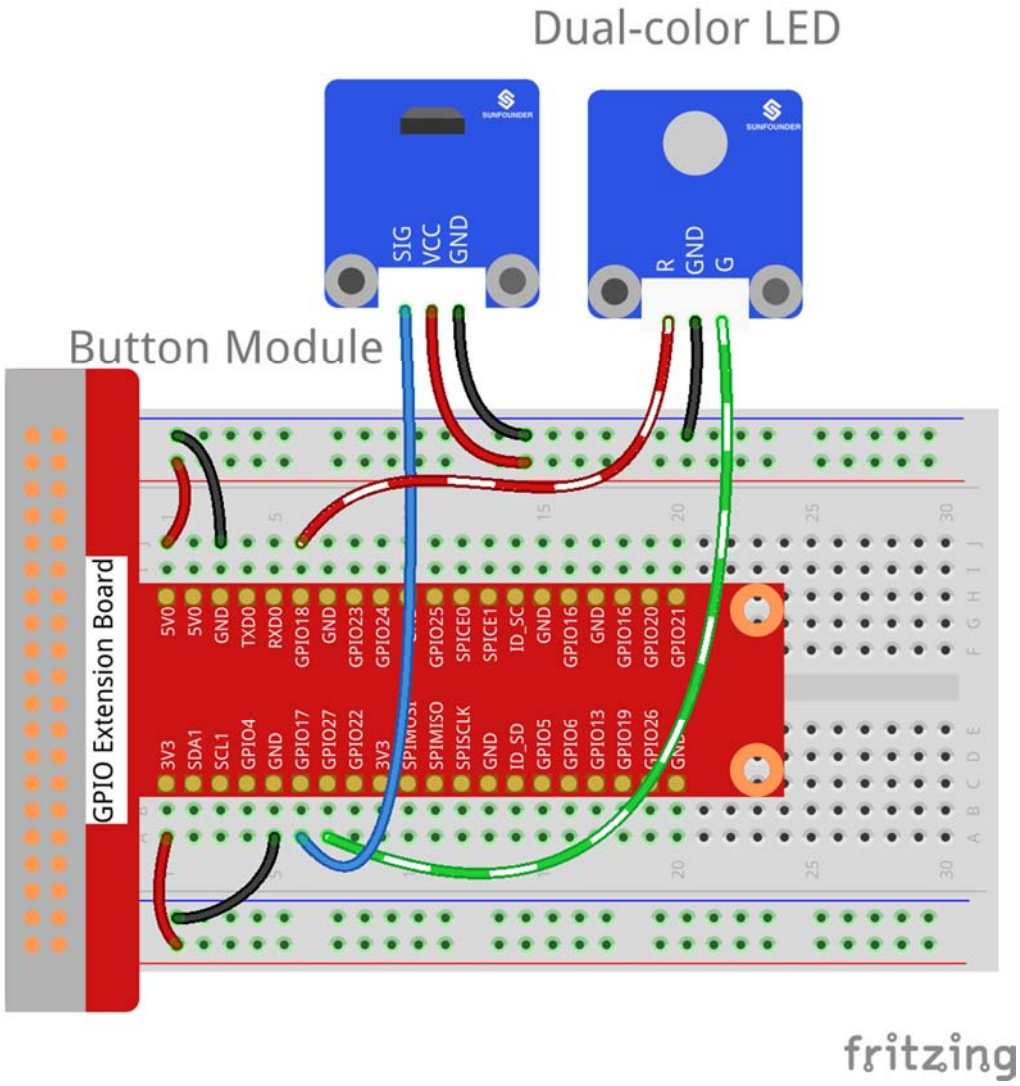
Experimental Procedures

For switch Hall sensor, take the following steps.

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	Switch Hall Module
GPIO0	GPIO17	SIG
5V	5V0	VCC
GND	GND	GND

Raspberry Pi	T-Cobbler	Dual-color LED Module
GPIO1	GPIO18	R
GND	GND	GND
GPIO2	GPIO27	G



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/17_switch_hall/
```

Step 3: Compile

```
gcc switch_hall.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

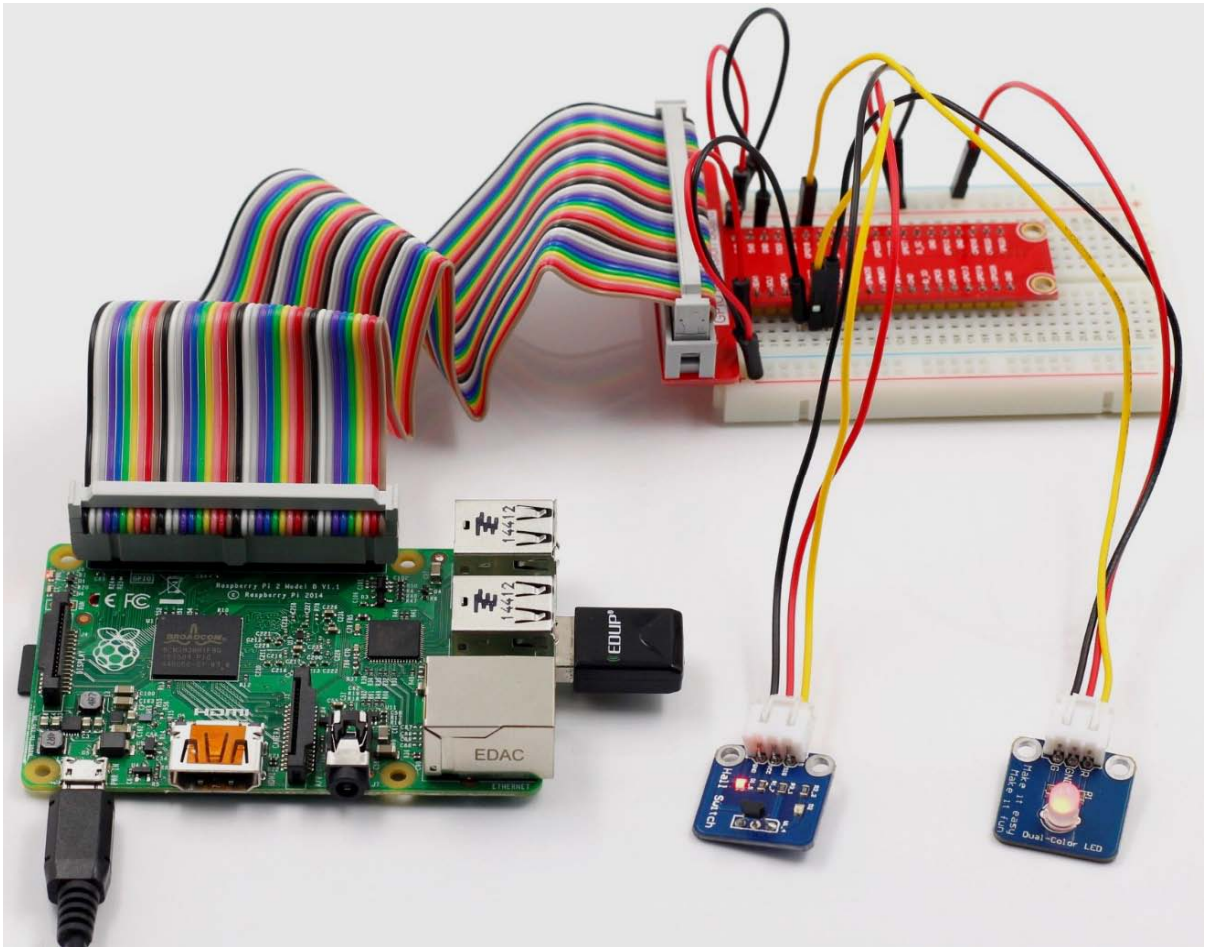
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 17_switch_Hall.py
```

Put a magnet close to the Switch Hall sensor. Then a string "**Detected magnetic materials**" will be printed on the screen and the LED will light up.

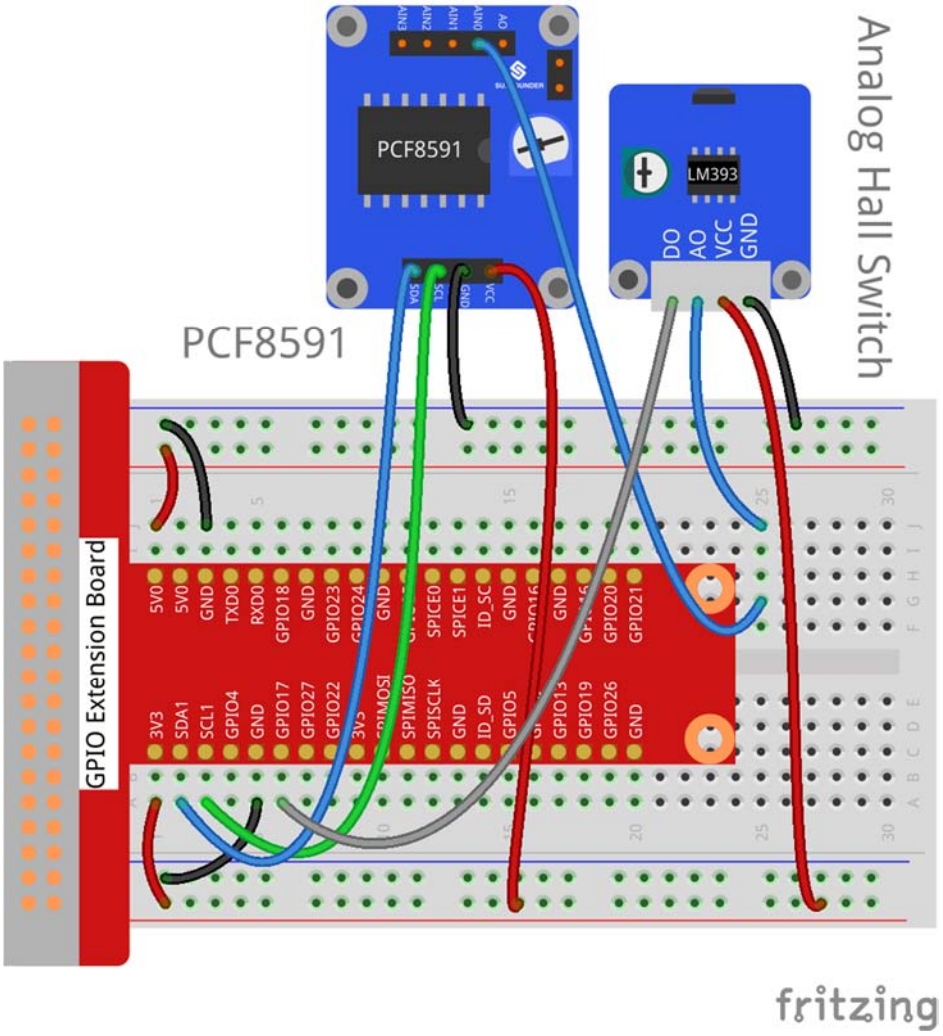


For **Analog Hall Switch**, take the following steps

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	PCF8591 module
SDA	SDA1	SDA
SCL	SCL1	SCL
3V3	3V3	VCC
GND	GND	GND

Analog Hall Switch	T-Cobbler	PCF8591 module
DO	GPIO17	*
AO	*	AIN0
VCC	3V3	VCC
GND	GND	GND



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/17_analog_hall_switch/
```

Step 3: Compile

```
gcc analog_hall_switch.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

Step 2: Change directory

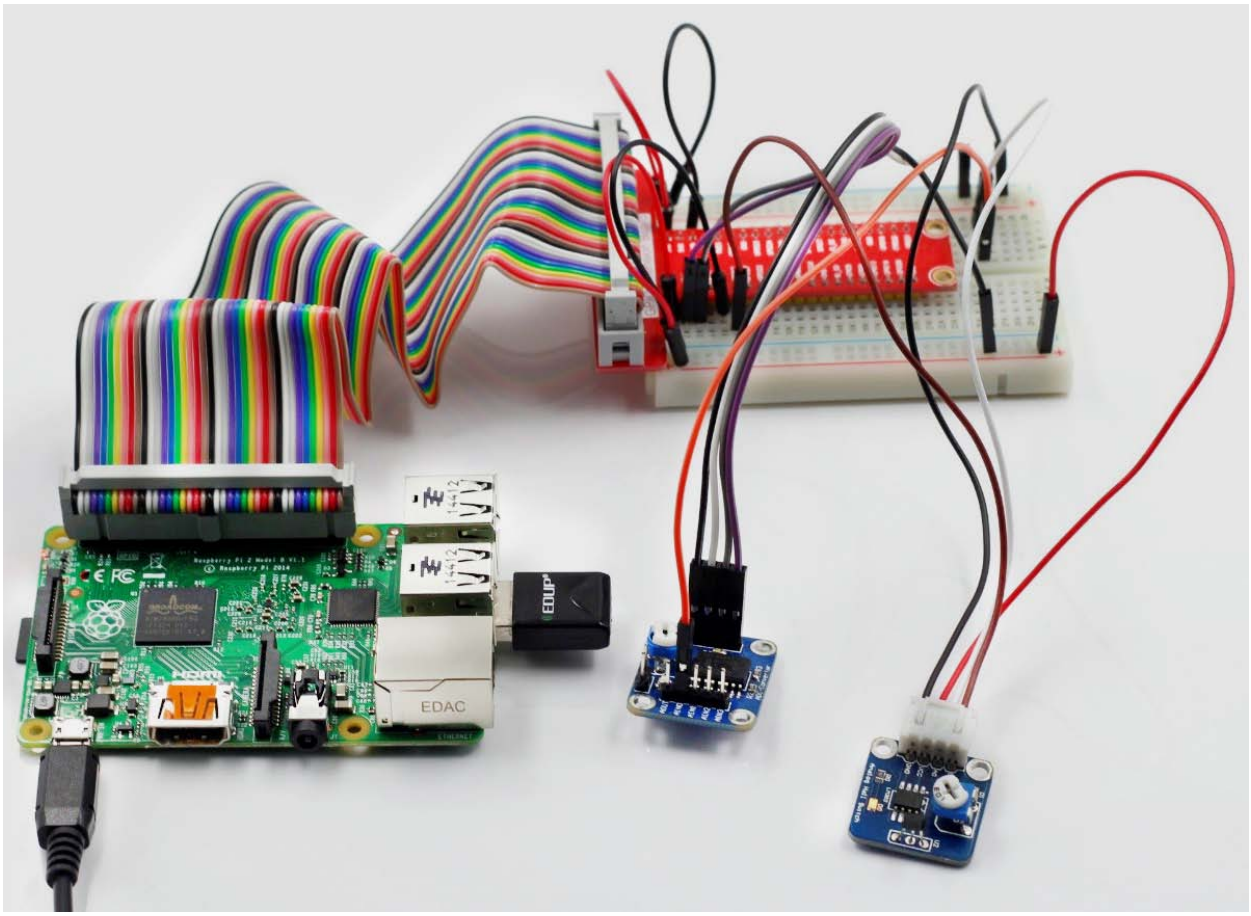
```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 17_analog_hall_switch.py
```

Now "Current intensity of magnetic field : xxx " will be displayed on the screen. Put the magnet close to the analog Hall sensor, with the north magnetic pole towards the sensor, and then " Magnet: North." will be displayed. Move the magnet away, and " Magnet: None." will be printed. If the magnet approaches the sensor with the south magnetic pole towards it, " Magnet: South." will be printed on the screen.

Note: Pin D0 of the Analog Hall Sensor will output "0" only when the south pole of the magnet approaches it, otherwise it will output "1".

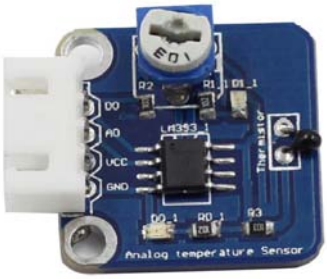


Lesson 18 Temperature Sensor

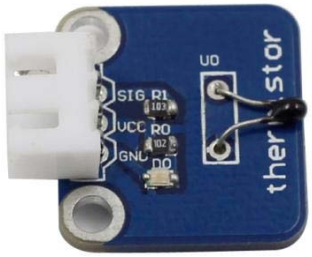
Introduction

A temperature sensor is a component that senses temperature and converts it into output signals. By material and component features, temperature sensors can be divided into two types: thermal resistor and thermocouple. Thermistor is one kind of the former type. It is made of semiconductor materials; most thermistors are negative temperature coefficient (NTC) ones, the resistance of which decreases with rising temperature. Since their resistance changes acutely with temperature changes, thermistors are the most sensitive temperature sensors.

There are two kinds of thermistor module in this kit (as shown below).



Analog temperature sensor



Thermistor

Components

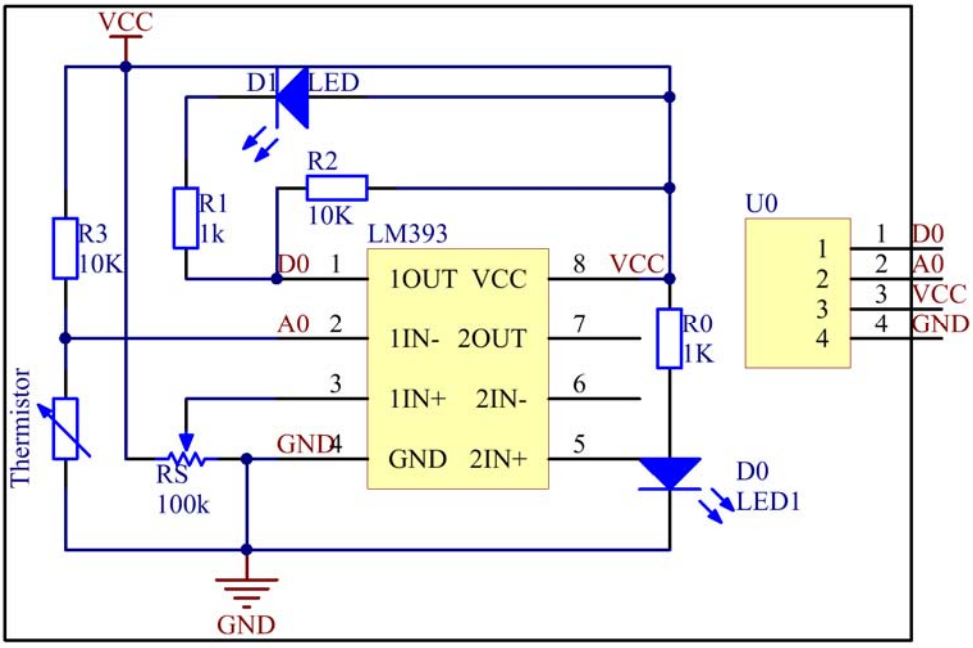
- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * Analog-temperature Sensor module
- 1 * Thermistor module
- 1 * PCF8591
- 1 * 3-Pin anti-reverse cable
- 1 * 4-Pin anti-reverse cable
- Several Jumper wires (Male to Female)

Experimental Principle

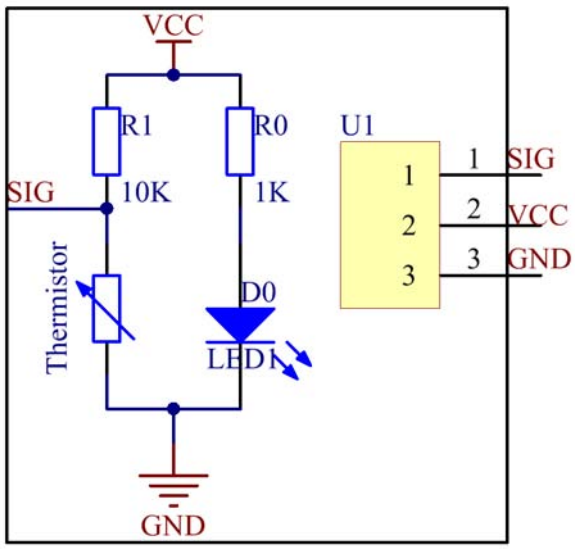
This module is based on the principle of the thermistor, whose resistance varies significantly with ambient temperature. When the ambient temperature increases, the resistance of the thermistor decreases; when decreases, it increases. It can detect surrounding temperature changes in a real-time manner.

In this experiment, we use an analog-digital converter PCF8591 to convert analog signals into digital ones.

The schematic diagram for analog temperature sensor:



The schematic diagram for the thermistor module:



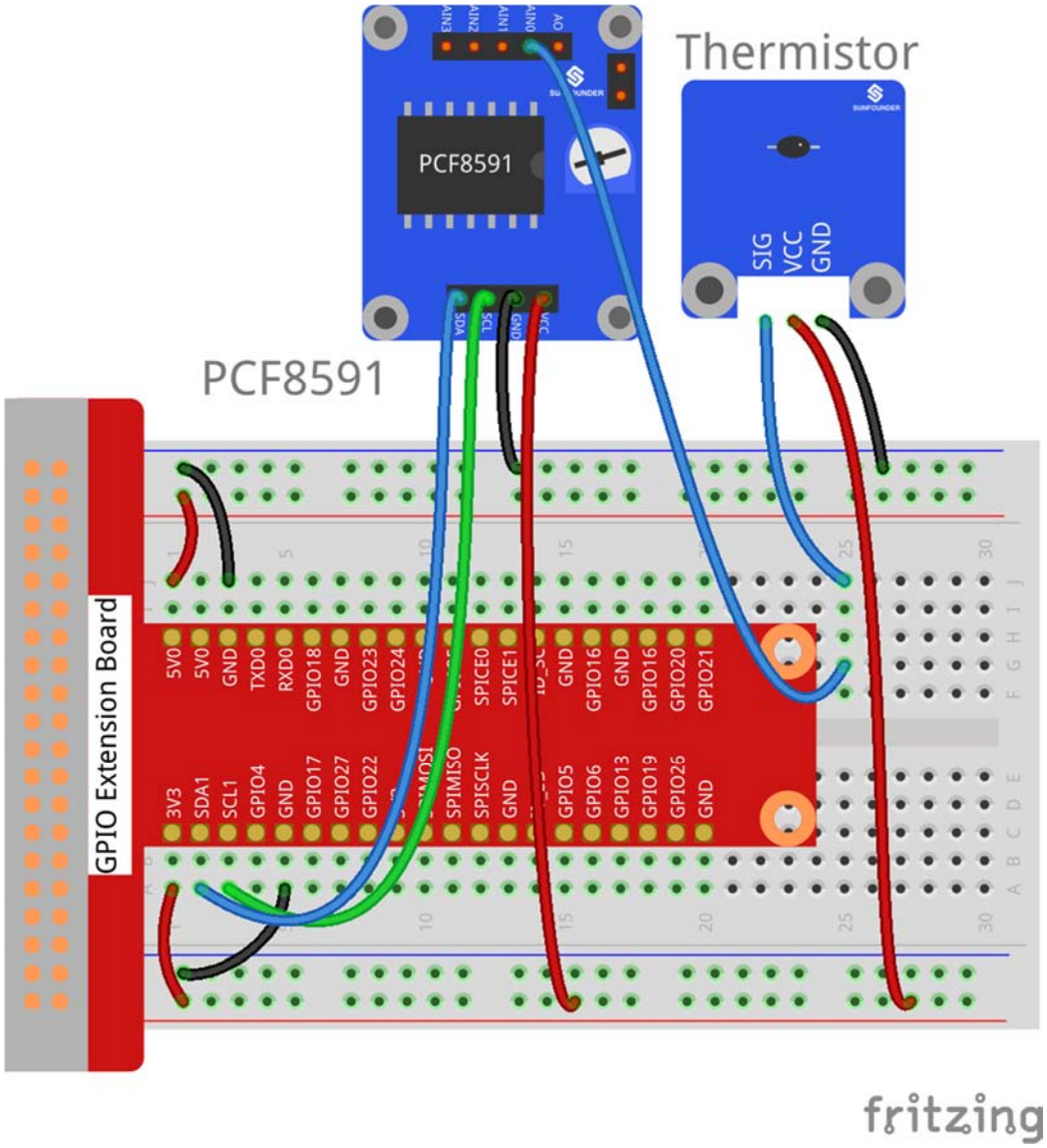
Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	PCF8591 Module
SDA	SDA1	SDA
SCL	SCL1	SCL
3V3	3V3	VCC
GND	GND	GND

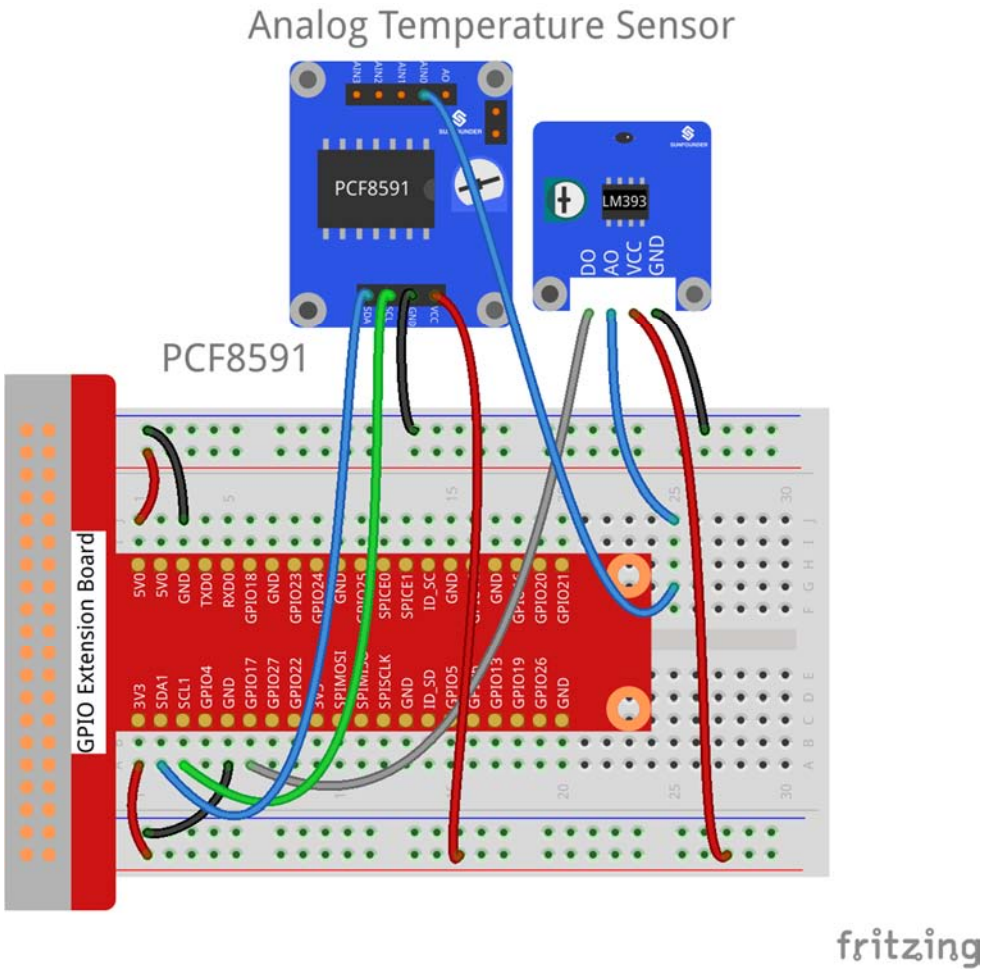
For thermistor module:

Thermistor Module	T-Cobbler	PCF8591 Module
SIG	*	AIN0
VCC	3V3	VCC
GND	GND	GND



For analog temperature sensor module：

Analog Temperature Module	T-Cobbler	PCF8591 Module
DO	GPIO17	*
AO	*	AIN0
VCC	3V3	VCC
GND	GND	GND



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/18_thermistor/
```

Step 3: Compile

```
gcc thermistor.c -lwiringPi -lm
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 18_thermistor.py
```

Now touch the thermistor and you can see the value of current temperature printed on the screen change accordingly.

Temperature alarm setting:

If you use the **Analog Temperature Sensor** module, uncomment the line under **1**:

For C language:

```
55        // For a threshold, uncomment one of the code for
56        // which module you use. DONOT UNCOMMENT BOTH!
57        //----------------------------------------
58        // 1. For Analog Temperature module(with DO)
59        tmp = digitalRead(DO);
60
61        // 2. For Thermister module(with sig pin)
62        // if (temp > 33) tmp = 0;
63        // else if (temp < 31) tmp = 1;
```

For Python

```
41        ##################################################
42        # 1. For Analog Temperature module(with DO)
43        tmp = GPIO.input(DO);
44
45        # 2. For Thermister module(with sig pin)
46        #if temp > 33:
47        #    tmp = 0;
48        #elif temp < 31:
49        #    tmp = 1;
50        ##################################################
```

If you use the **Thermistor module**, uncomment the line under **2**:

For C language:

```
55        // For a threshold, uncomment one of the code for
56        // which module you use. DONOT UNCOMMENT BOTH!
57        //----------------------------------------
58        // 1. For Analog Temperature module(with DO)
59        // tmp = digitalRead(DO);
```

```
60
61          // 2. For Thermister module(with sig pin)
62          if (temp > 33) tmp = 0;
63          else if (temp < 31) tmp = 1;
64          //----------------------------------------
```

For Python

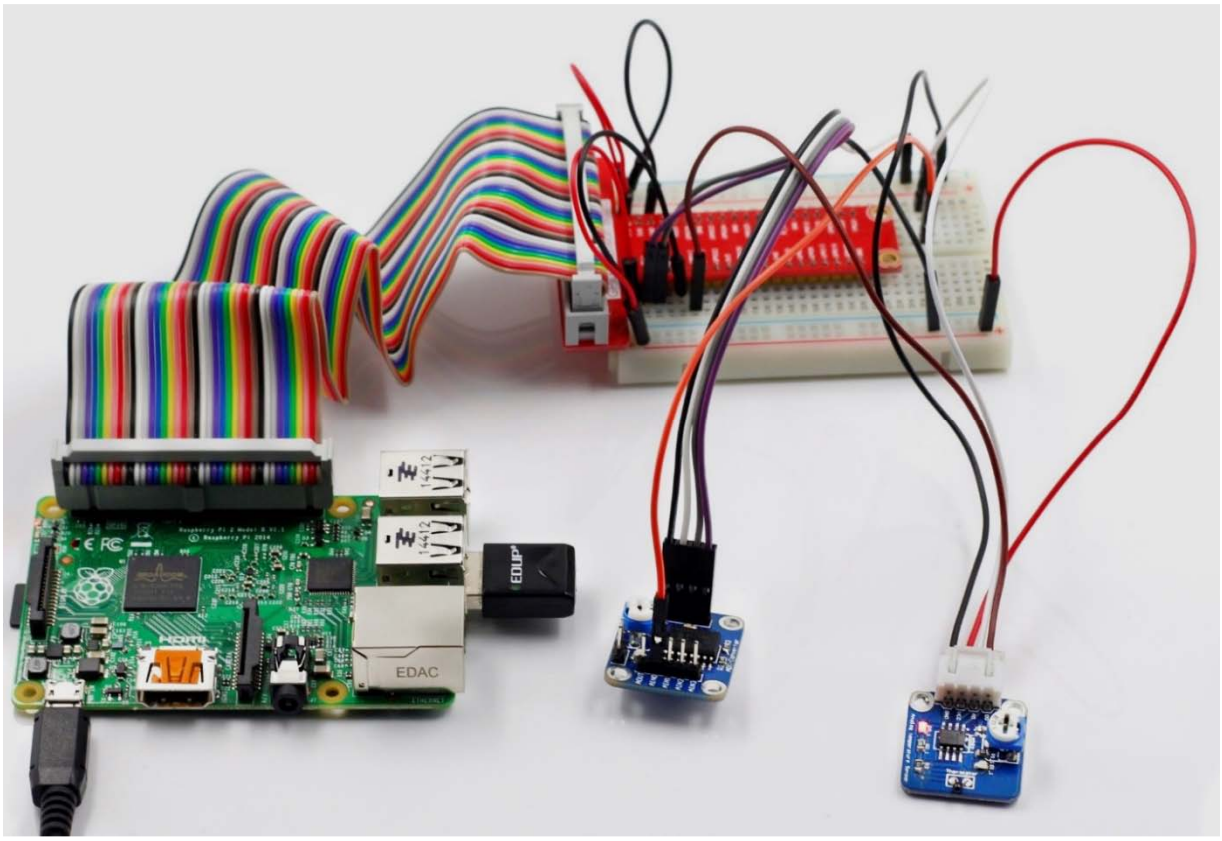
```
41          ##################################################
42          # 1. For Analog Temperature module(with DO)
43          #tmp = GPIO.input(DO);
44          #
45          # 2. For Thermister module(with sig pin)
46          if temp > 33:
47              tmp = 0;
48          elif temp < 31:
49              tmp = 1;
50          ##################################################
```

After editing the code, repeat step 2, 3, and 4 (or step 2, 3 for Python users).

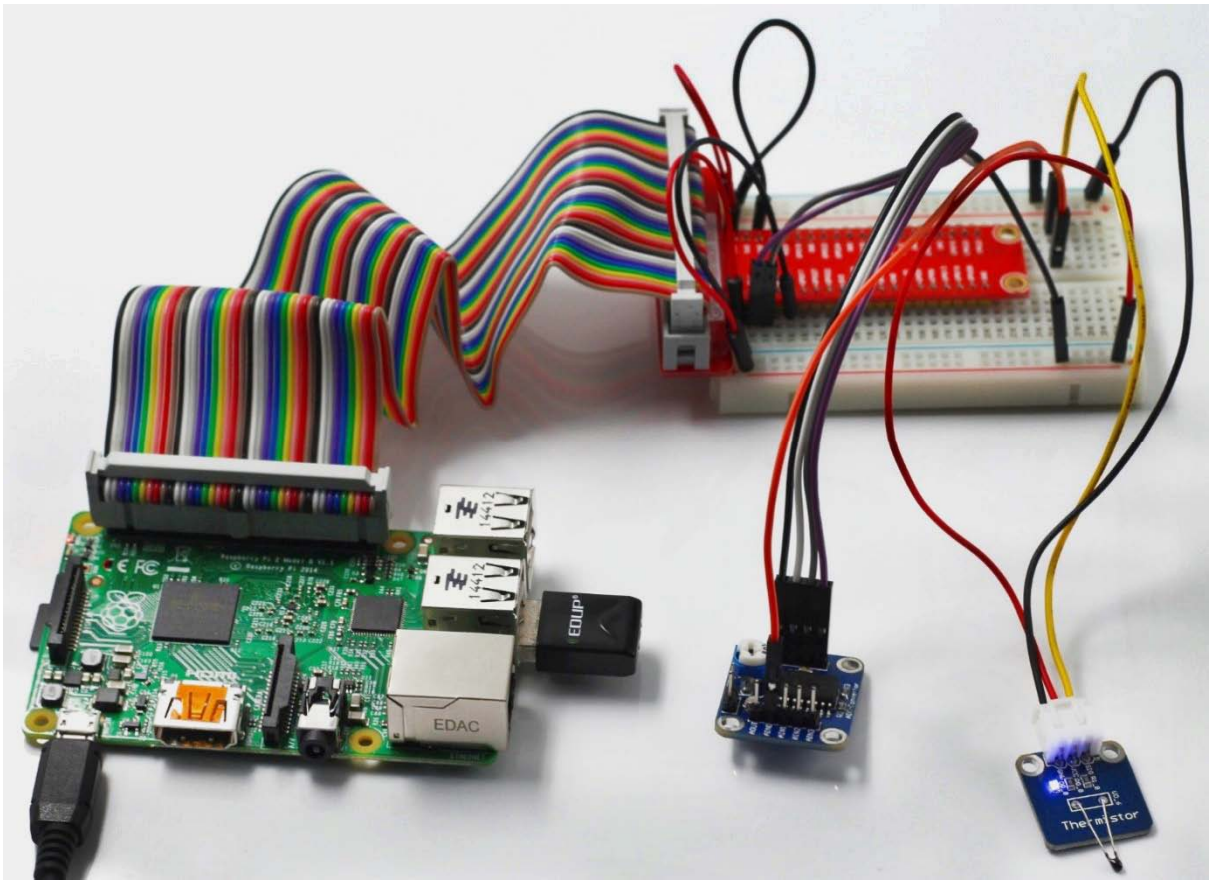
You can still see temperature value printed on the screen constantly. If you pinch the thermistor for a while, its temperature will rise slowly. "Too Hot!" will be printed on the screen. Release your fingers, and let it stay in the open air for a while, or blow on the module. When the temperature drops down slowly, "Better" will be printed.

Note: The analog temperature sensor adjusts alarm temperature by the potentiometer on the module. The thermistor changes the alarm temperature by program.

The physical picture for analog temperature sensor:



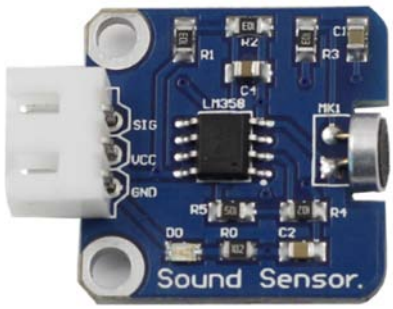
The physical picture for thermistor module:



Lesson 19 Sound Sensor

Introduction

Sound sensor is a component that receives sound waves and converts them into electrical signal. It detects the sound intensity in ambient environment like a microphone.



Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * PCF8591
- 1 * Sound sensor module
- 1 * 3-Pin anti-reverse cable
- Several Jumper wires (Male to Female)

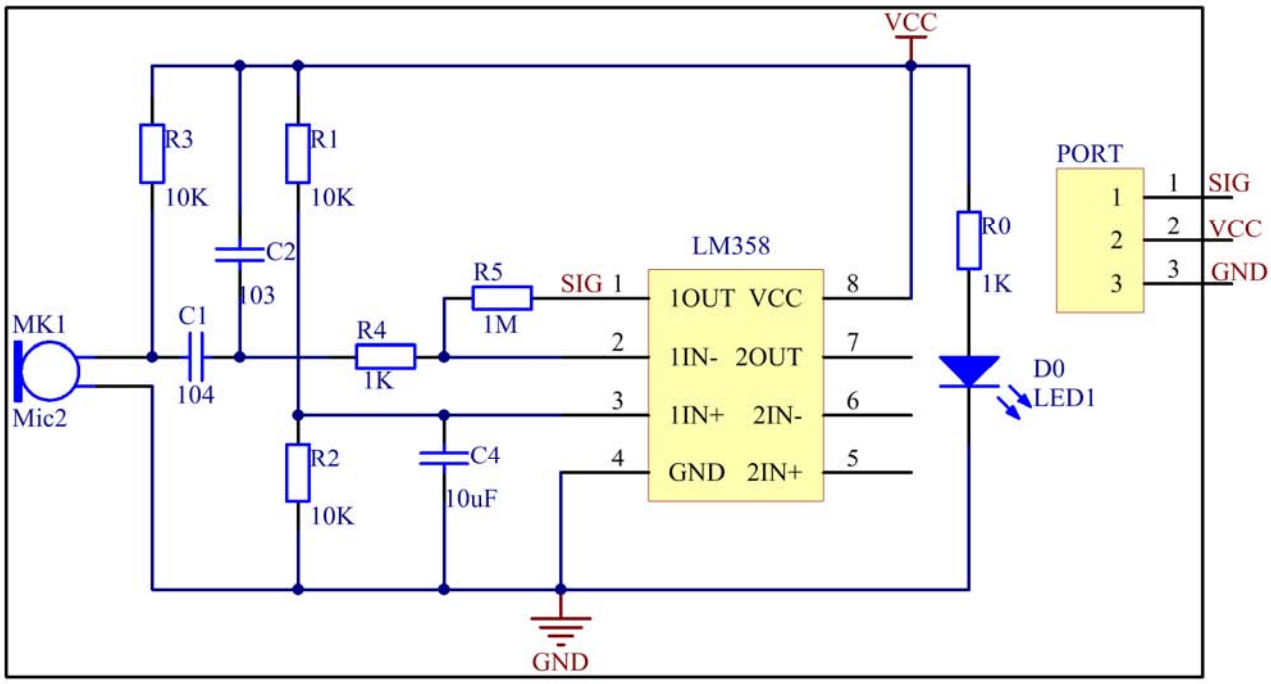
Experimental Principle

The microphone on the sensor module can convert audio signals into electrical signals (analog quantity), then convert analog quantity into digital quantity by PCF8591 and transfer them to MCU.

LM358 is a dual-channel operational amplifier. It contains two independent, high gain, and internally compensated amplifiers, but we will only use one of them in this experiment. The microphone transforms sound signals into electrical signals and then sends out the signals to pin 2 of LM358 and outputs them to pin 1 (that's, pin SIG of the module) via the external circuit. Then use PCF8591 to read analog values.

PCF8591 is an 8-bit resolution, 4-channel A/D，1-channel D/A conversion chip. We connect the output terminal (SIG) to AIN0 of PCF8591 so as to detect the strength of voice signal in a real-time manner.

The schematic diagram:

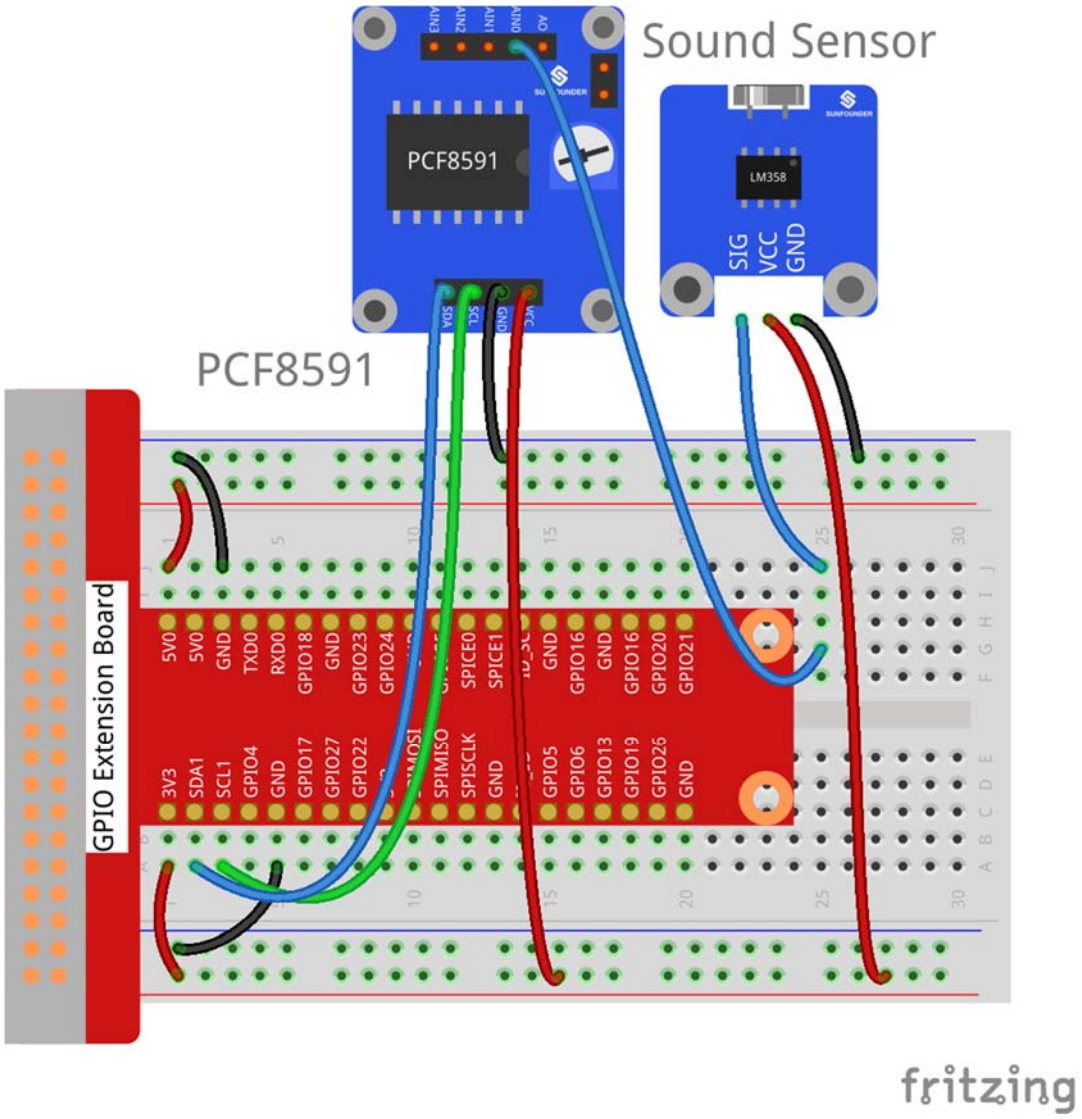


Experimental Procedures

Step 1: Build the circuit according to the following method

Raspberry Pi	T-Cobbler	PCF8591 Module
SDA	SDA1	SDA
SCL	SCL1	SCL
3V3	3V3	VCC
GND	GND	GND

Sound Sensor Module	T-Cobbler	PCF8591 Module
SIG	*	AIN0
VCC	3V3	VCC
GND	GND	GND



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/19_sound_sensor/
```

Step 3: Compile

```
gcc sound_sensor.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

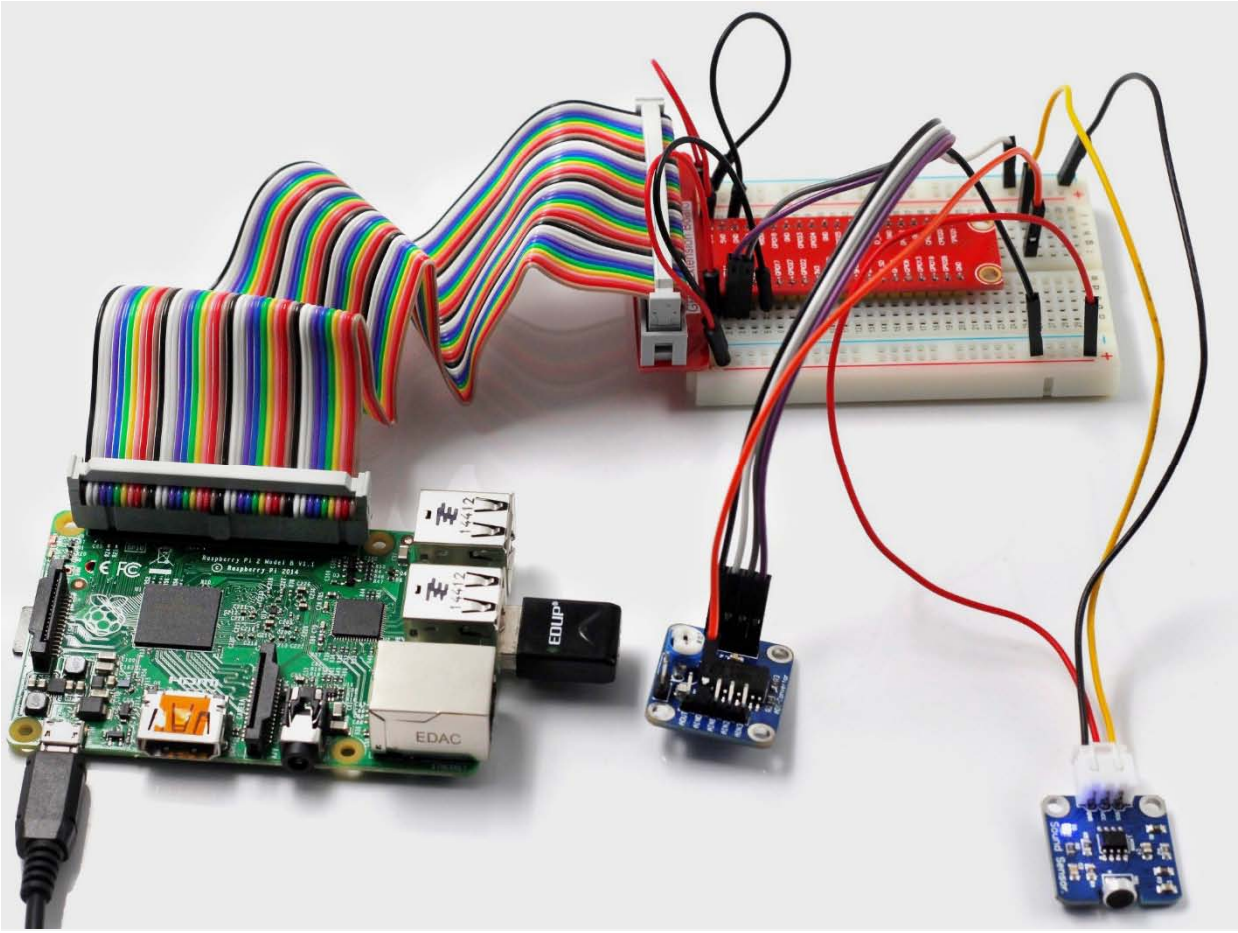
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 19_sound_sensor.py
```

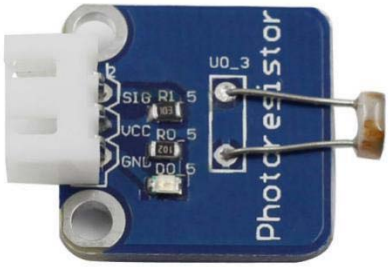
Now, speak close to or blow to the microphone, and you can see "Voice In!! ***" printed on the screen.



Lesson 20 Photoresistor Module

Introduction

A photoresistor is a light-controlled variable resistor. The resistance of a photoresistor decreases with increasing incident light intensity.

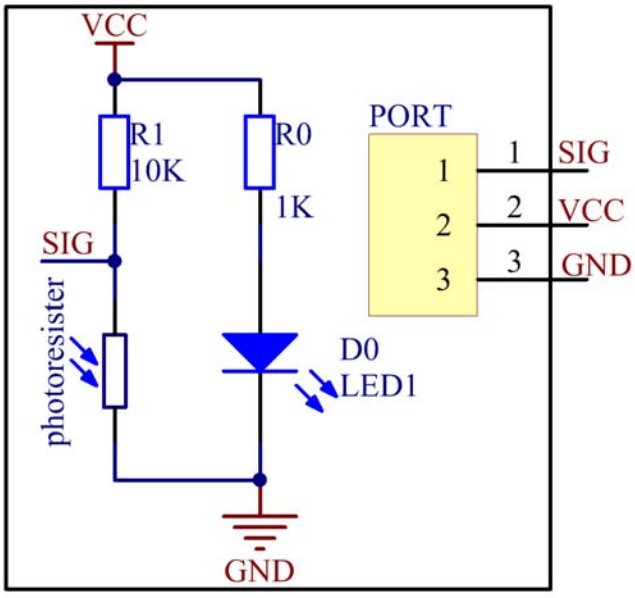


Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * PCF8591
- 1 * Photoresistor module
- 1 * 3-Pin anti-reverse cable
- Several Jumper wires (Male to Female)

Experimental Principle

With light intensity increasing, the resistance of a photoresistor will decrease. Thus the output voltage changes. Analog signals collected by the photoresistor are converted to digital signals through PCF8591. Then these digital signals are transmitted to Raspberry Pi and printed on the screen. The schematic diagram:

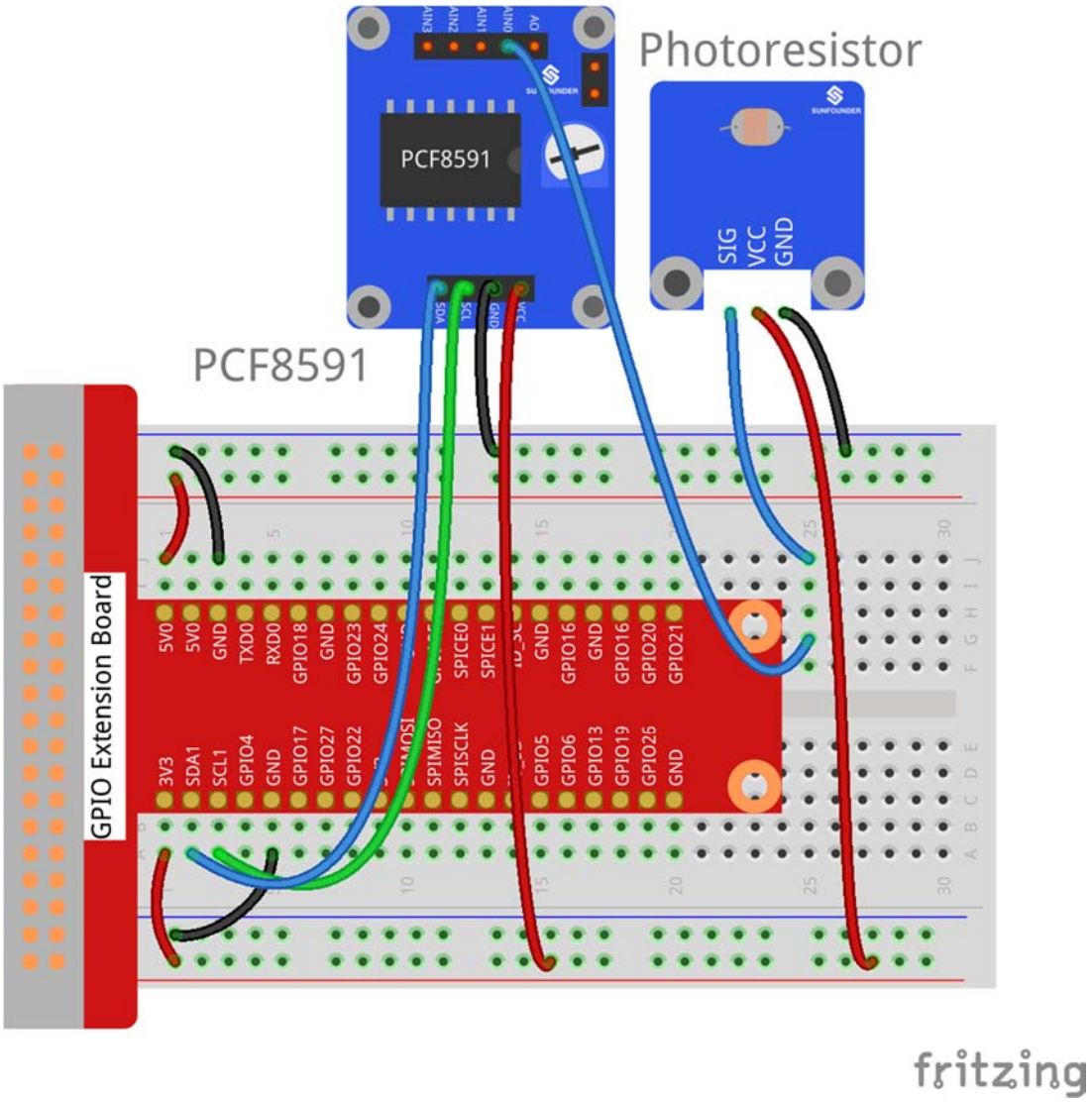


Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	PCF8591 Module
SDA	SDA1	SDA
SCL	SCL1	SCL
3V3	3V3	VCC
GND	GND	GND

Photoresistor	T-Cobbler	PCF8591 Module
SIG	*	AIN0
VCC	3V3	VCC
GND	GND	GND



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/20_photoresistor/
```

Step 3: Compile

```
gcc photoresistor.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

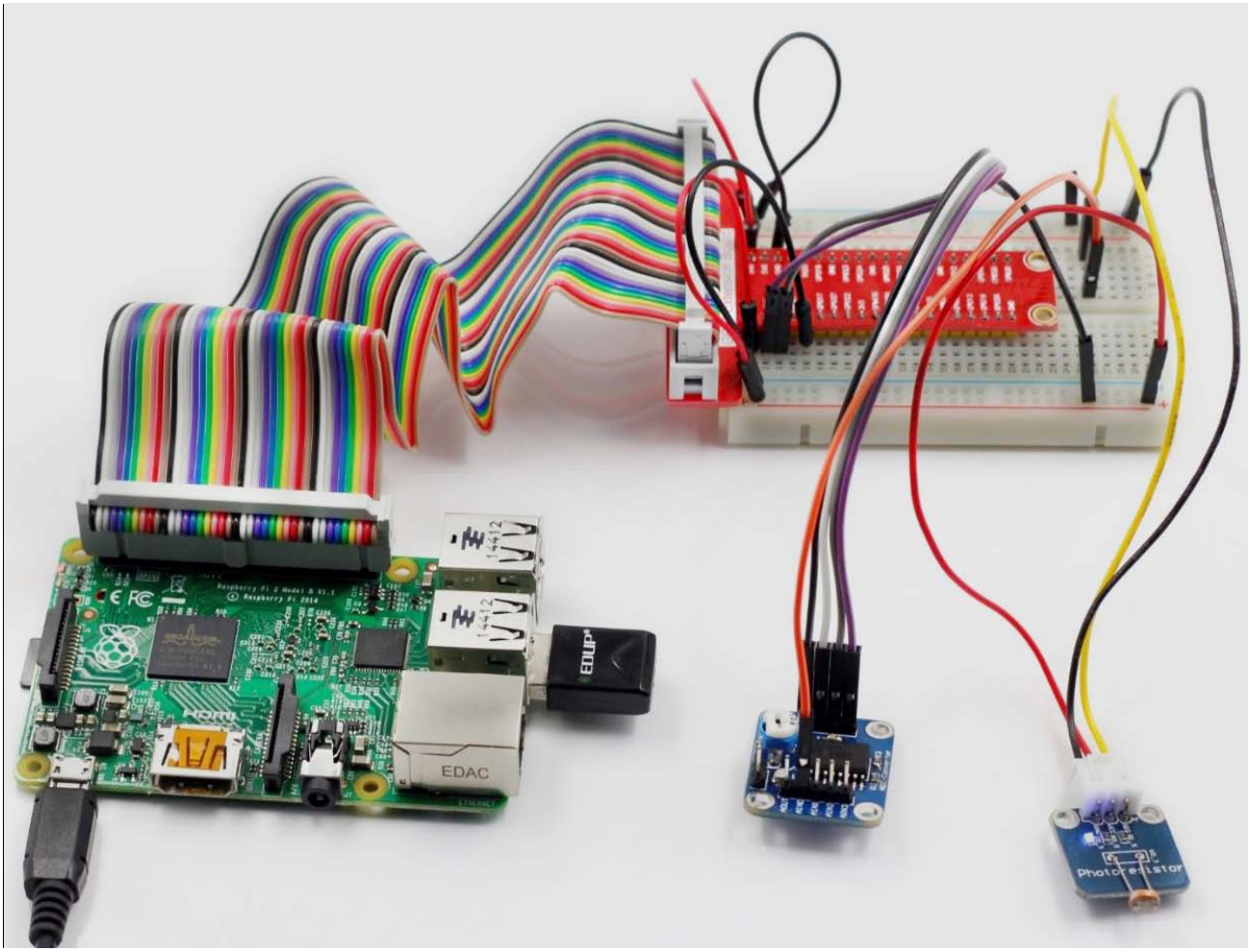
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 20_photoresistor.py
```

Now, change light intensity (e.g. cover the module with a pad), and the value printed on the screen will change accordingly.



Lesson 21 Flame Sensor

Introduction

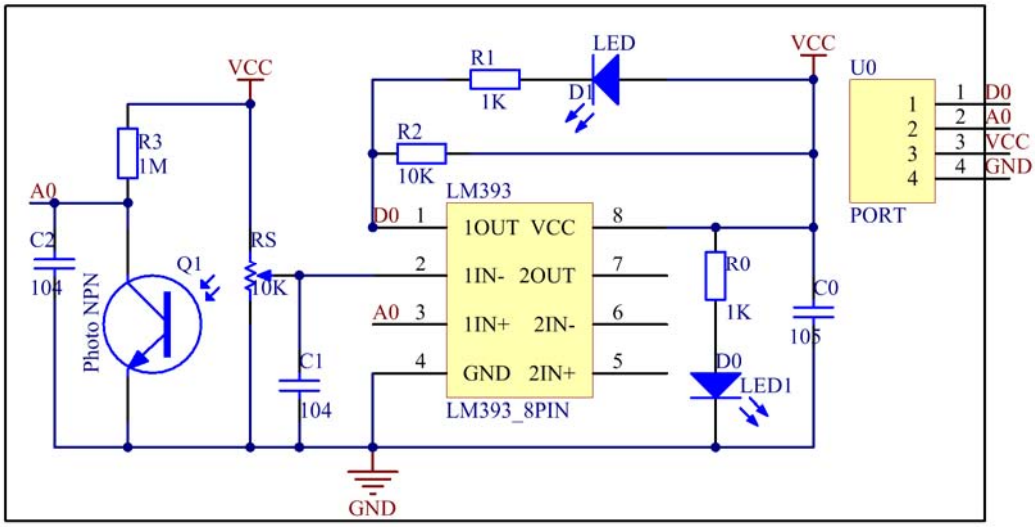
A flame sensor (as shown below) performs detection by capturing infrared rays with specific wavelengths from flame. It can be used to detect and warn of flames.

Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * Flame sensor module
- 1 * PCF8591
- 1 * 4-Pin anti-reverse cable
- Several Jumper wires (Male to Female)

Experimental Principle

There are several types of flame sensors. In this experiment, we will use a far-infrared flame sensor. It can detect infrared rays with wavelength ranging from 700nm to 1000nm. A far-infrared flame probe converts the strength changes of external infrared light into current changes. And then it convert analog quantities into digital ones. In this experiment, connect pin D0 of the Flame Sensor module to a GPIO of Raspberry Pi to detect by programming whether any flame exists. The schematic diagram:

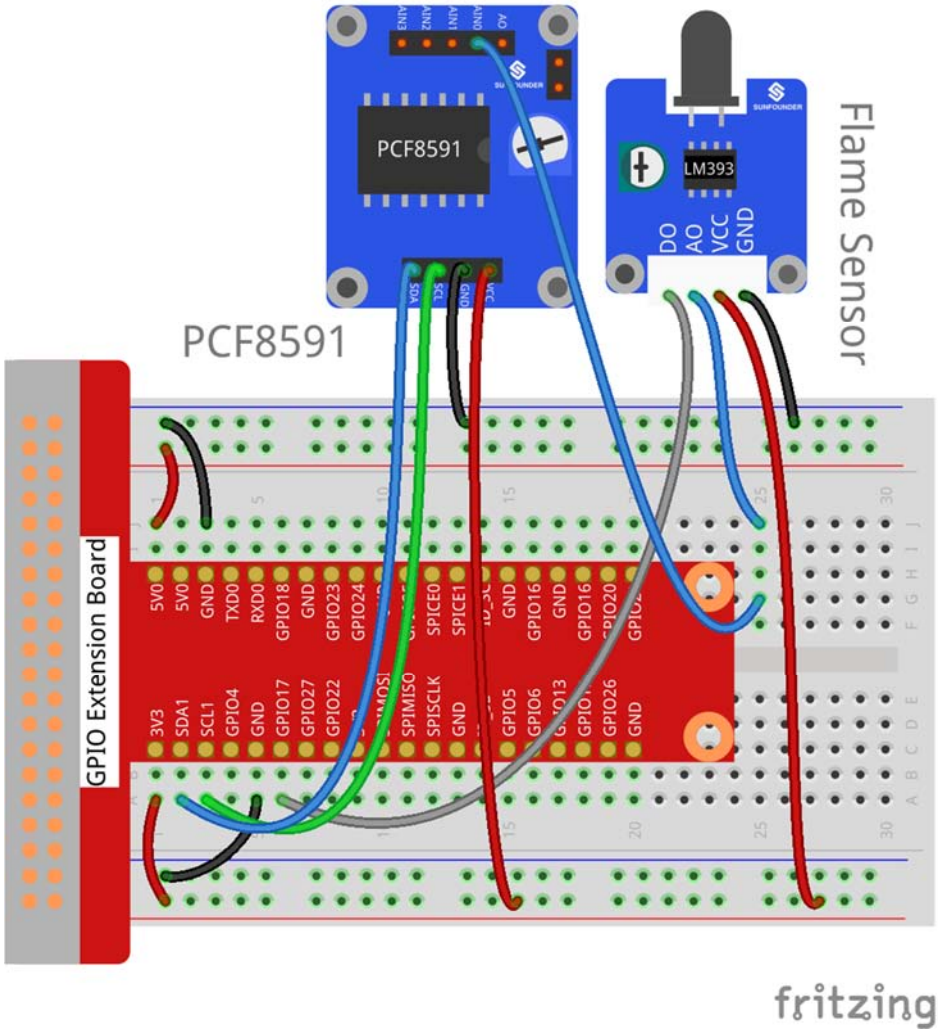


Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	PCF8591 Module
SDA	SDA1	SDA
SCL	SCL1	SCL
3V3	3V3	VCC
GND	GND	GND

Flame Sensor	T-Cobbler	PCF8591 Module
DO	GPIO17	*
AO	*	AIN0
VCC	3V3	VCC
GND	GND	GND



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/21_flame_sensor/
```

Step 3: Compile

```
gcc flame_sensor.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

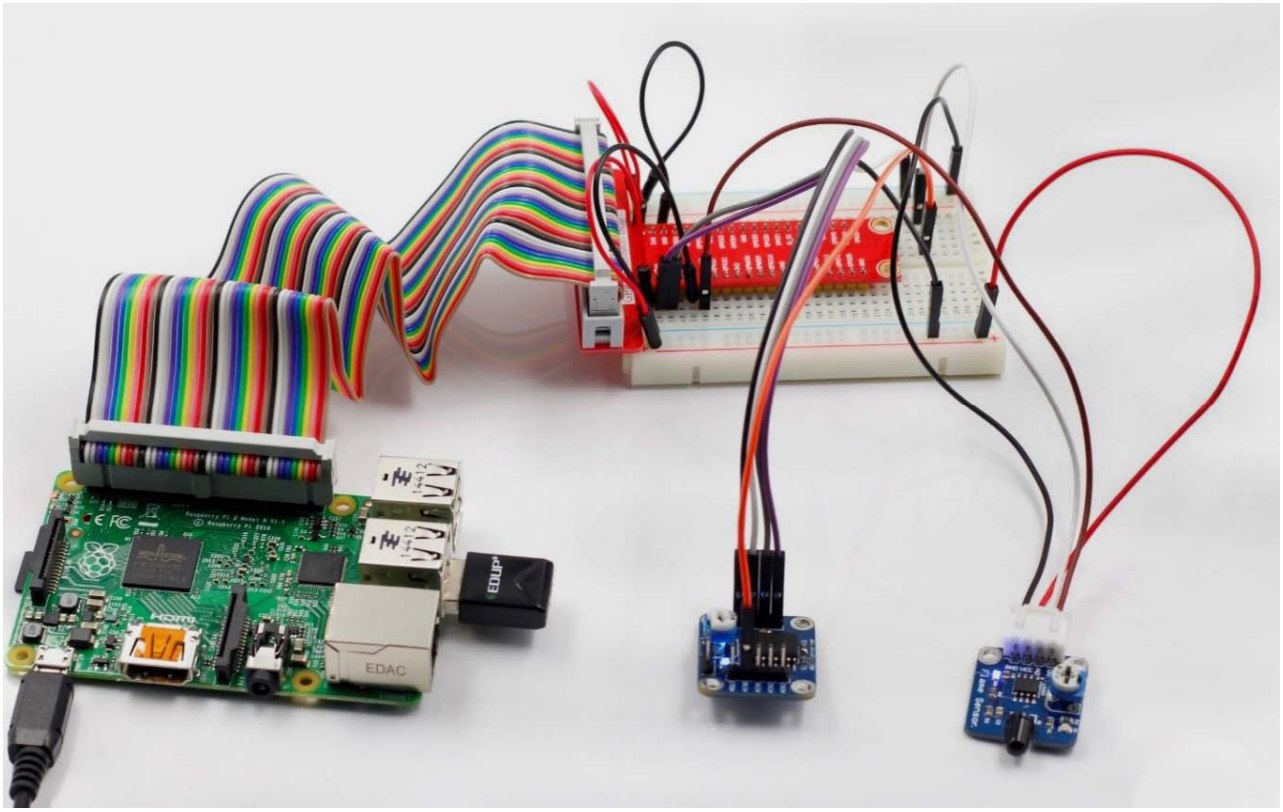
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 21_flame_sensor.py
```

Now, ignite a lighter near the sensor, within the range of 80cm, and "Fire!" will be displayed on the screen. If you put out the lighter or just move the flames away from the flame sensor, "Safe~" will be displayed then.



Lesson 22 Gas Sensor

Introduction

Gas Sensor MQ-2 is a sensor for flammable gas and smoke by detecting the concentration of combustible gas in the air. They are used in gas detecting equipment for smoke and flammable gasses in household, industry or automobile.

Components

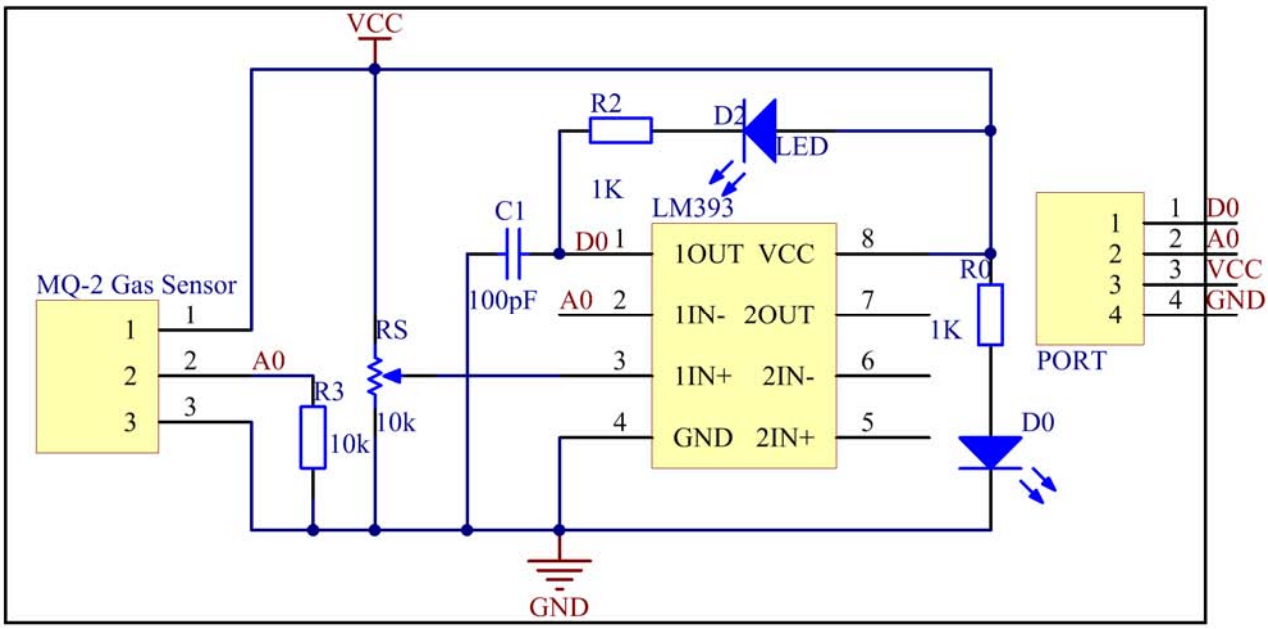
- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * Active Buzzer module
- 1 * PCF8591
- 1 * Gas sensor module
- 1 * 3-Pin anti-reverse cable
- 1 * 4-Pin anti-reverse cable
- Several Jumper wires (Male to Female)

Experimental Principle

MQ-2 gas sensor is a kind of surface ion type and N-type semiconductors, which uses tin oxide semiconductor gas sensitive material. When ambient temperature is in 200 ~ 300℃, tin oxide will adsorb oxygen in the air and form oxygen anion adsorption to decrease electron density in semiconductor so as to increase its resistance. When in contact with the smoke, if grain boundary barrier is modulated by the smoke and changed, it could cause surface conductivity change. So you can gain the information of the smoke existence, The higher the smoke concentration is, the more conductive the material becomes, thus the lower the output resistance is.

In this experiment, if harmful gases reach a certain concentration, the buzzer will beep to warn.

The schematic diagram:



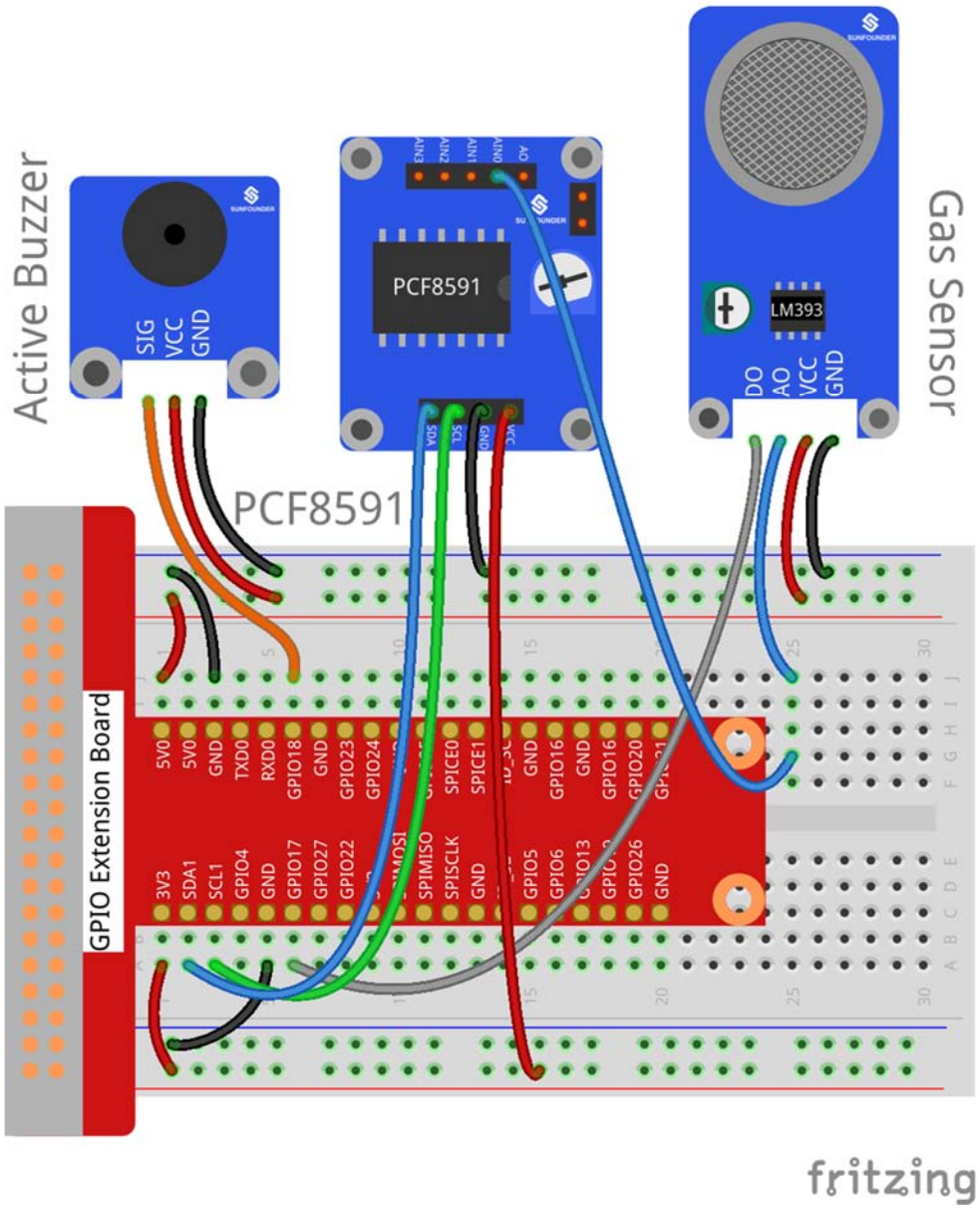
Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	PCF8591 Module
SDA	SDA1	SDA
SCL	SCL1	SCL
3V3	3V3	VCC
GND	GND	GND

Gas Sensor Module	T-Cobbler	PCF8591 Module
DO	GPIO17	*
AO	*	AIN0
VCC	5V0	VCC
GND	GND	GND

Raspberry Pi	T-Cobbler	Active Buzzer Module
GPIO1	GPIO18	SIG
5V	3V3	VCC
GND	GND	GND



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/22_gas_sensor/
```

Step 3: Compile

```
gcc gas_sensor.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

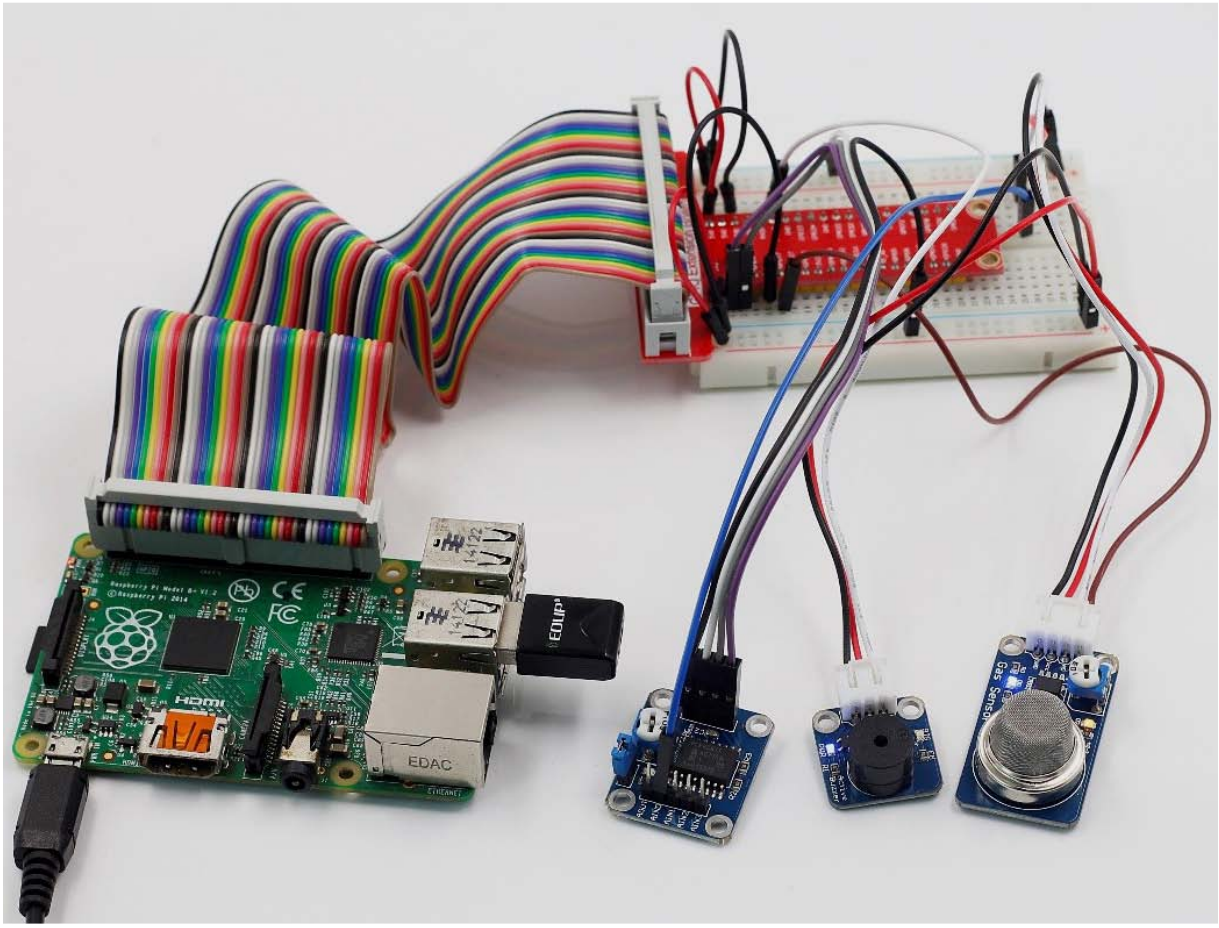
```
sudo python 22_gas_sensor.py
```

Place a lighter close to the MQ-2 gas sensor, and press the switch to release gasses. A value between 0 and 255 will be displayed on the screen. If harmful gases reach a certain concentration, the buzzer will beep, and "Danger Gas!" will be printed on the screen.

You can also turn the shaft of the potentiometer on the module to raise or reduce the concentration threshold.

The MQ-2 gas sensor needs to be heated up for a while. Wait until the value printed on screen stays steady and the sensor gets warm, which means it can work normally and sensitively at that time.

Note: It is normal that the gas sensor generates heat. Actually, the higher the temperature is, the sensor is more sensitive.



Lesson 23 IR Remote Control

Introduction

Each button of an IR remote control (as shown below) has a string of specific encoding. When a button is pressed, the IR transmitter in the remote control will send out the corresponding IR encoding signals. On the other side, when the IR receiver receives certain encoding signals, it will decode them to identify which button is pressed.

Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * IR Receiver
- 1 * RGB LED module
- 1 * IR Remote Control
- 1 * 3-Pin anti-reverse cable
- 1 * 4-Pin anti-reverse cable

Experimental Principle

In this experiment, we use the lirc library to read infrared signals returned by buttons of the remote control and translate them to button values. Then use liblircclient-dev (C) and pylirc (Python) to simplify the process for reading values from the remote control. In this experiment use 9 buttons on the top of the remote to control the color of the RGB LED module. Each row represents one color, and each column represents the brightness.

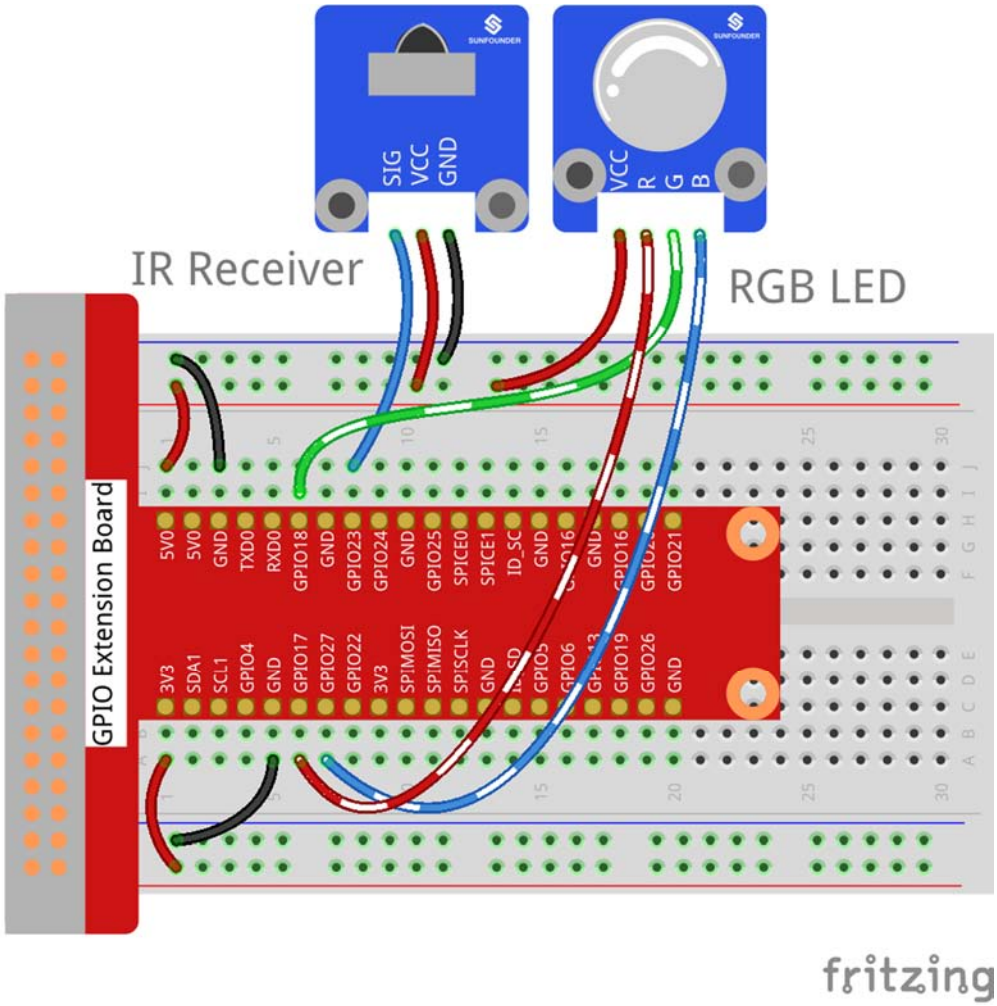
	OFF	Dark	Bright
Red			
Green			
Blue			

Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	IR Receiver Module
GPIO4	GPIO23	SIG
5V	5V0	VCC
GND	GND	GND

Raspberry Pi	T-Cobbler	RGB LED Module
5V	5V0	VCC
GPIO0	GPIO17	R
GPIO1	GPIO18	G
GPIO2	GPIO27	B



Step 2: Download the *LIRC* library

```
sudo apt-get install lirc
```

Step 3: Set up lirc:

Open your */etc/modules* file:

```
sudo nano /etc/modules
```

Add this to the end:

```
lirc_dev
lirc_rpi gpio_in_pin=23 gpio_out_pin=22
```

Press Ctrl +O and Ctrl +X, save and exit .

Open the */etc/lirc/hardware.conf* file:

```
sudo nano /etc/lirc/hardware.conf
```

Modify the file as shown below:

```
# Run "lircd --driver=help" for a list of supported drivers.
DRIVER="default"
# usually /dev/lirc0 is the correct setting for systems using udev
DEVICE="/dev/lirc0"
MODULES="lirc_rpi"
```

Press Ctrl +O and Ctrl +X, save and exit.

Copy the configuration file to*/home/pi* and */etc/lirc*:

```
cd /home/pi/SunFounder_SensorKit_for_RPi2
cp lircd.conf /home/pi
sudo cp lircd.conf /etc/lirc/
```

Open the */boot/config.txt* file:

```
sudo nano /boot/config.txt
```

Add the following line to the end:

```
dtoverlay=lirc-rpi:gpio_in_pin=23,gpio_out_pin=22
```

Press Ctrl +O and Ctrl +X, save and exit.

Reboot the Raspberry Pi after the change.

```
sudo reboot
```

Step 4: Test the IR receiver

Check if lirc module is loaded:

```
ls /dev/li*
```

You should see this:

```
/dev/lirc0      /dev/lircd
```

Run the command to stop lircd and start outputting raw data from the IR receiver:

```
irw
```

When you press a button on the remote, you can see the button name printed on the screen.

```
pi@raspberrypi:~ $ irw
0000000000000001 00 KEY_CHANNELDOWN /home/pi/lircd.conf
0000000000000002 00 KEY_CHANNEL /home/pi/lircd.conf
0000000000000003 00 KEY_CHANNELUP /home/pi/lircd.conf
0000000000000004 00 KEY_PREVIOUS /home/pi/lircd.conf
0000000000000005 00 KEY_NEXT /home/pi/lircd.conf
0000000000000006 00 KEY_PLAYPAUSE /home/pi/lircd.conf
0000000000000007 00 KEY_VOLUMEDOWN /home/pi/lircd.conf
0000000000000008 00 KEY_VOLUMEUP /home/pi/lircd.conf
0000000000000009 00 KEY_EQUAL /home/pi/lircd.conf
000000000000000a 00 KEY_NUMERIC_0 /home/pi/lircd.conf
0000000000000014 00 BTN_0 /home/pi/lircd.conf
0000000000000015 00 BTN_1 /home/pi/lircd.conf
000000000000000b 00 KEY_NUMERIC_1 /home/pi/lircd.conf
000000000000000c 00 KEY_NUMERIC_2 /home/pi/lircd.conf
000000000000000d 00 KEY_NUMERIC_3 /home/pi/lircd.conf
000000000000000e 00 KEY_NUMERIC_4 /home/pi/lircd.conf
000000000000000f 00 KEY_NUMERIC_5 /home/pi/lircd.conf
0000000000000010 00 KEY_NUMERIC_6 /home/pi/lircd.conf
0000000000000011 00 KEY_NUMERIC_7 /home/pi/lircd.conf
0000000000000012 00 KEY_NUMERIC_8 /home/pi/lircd.conf
0000000000000013 00 KEY_NUMERIC_9 /home/pi/lircd.conf
```

If it does not appear, somewhere may be incorrectly configured. Check again that you've connected everything and haven't crossed any wires.

For C language users:

Step 5: Download LIRC client library:

```
sudo apt-get install liblircclient-dev
```

Step 6: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/23_ircontrol/
```

Step 7: Create a *lirc* directory under */etc/lirc/*:

```
sudo mkdir /etc/lirc/lirc/
```

Copy the *lircrc* file to */etc/lirc/lirc/*:

```
sudo cp lircrc /etc/lirc/lirc/
```

Step 8: Compile

```
gcc ircontrol.c -lwiringPi -llirc_client
```

Step 9: Run

```
sudo ./a.out
```

For Python users:

Step 5: Download *pylirc*:

```
sudo apt-get install python-pylirc
```

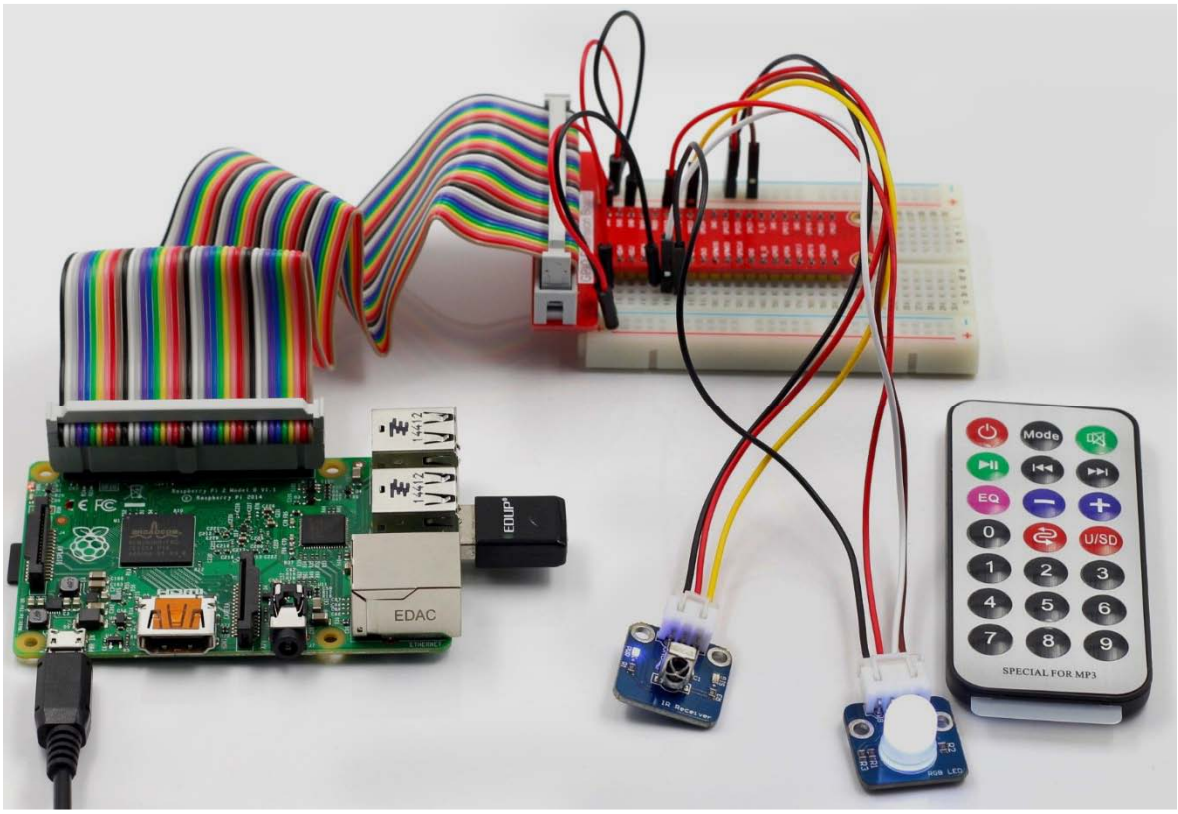
Step 6: Change directory:

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 7: Run

```
sudo python 23_ircontrol.py
```

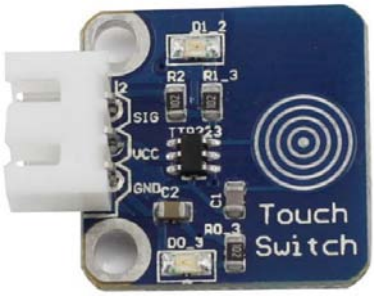
Each of the top three rows of buttons on the remote control represents a kind of color, i.e. red, green, and blue, top to bottom. Each column represents off, light, and dark. For example, press the second button (light) on the first row (red), and the LED will flash light red. You can use the remote to generate 27 colors in total (including all the LEDs off). Try to change the color of the RGB LED with the 9 buttons!



Lesson 24 Touch Switch

Introduction

A touch sensor operate with the conductivity of human body. When you touch the metal on the base electrode of the transistor, the level of pin SIG will turn over.

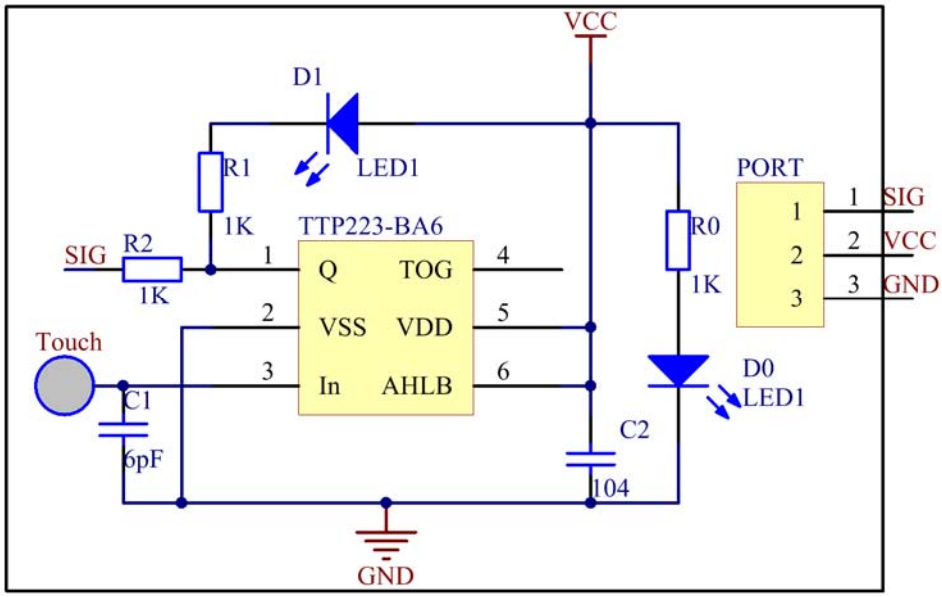


Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * Touch sensor module
- 1 * Dual-Color LED module
- 2 * 3-Pin anti-reverse cable

Experimental Principle

In this experiment, touch the base electrode of the transistor by fingers to make it conduct as human body itself is a kind of conductor and an antenna that can receive electromagnetic waves in the air. These electromagnetic wave signals collected from the human body are amplified by the transistor and processed by the comparator on the module to output steady signals. The schematic diagram:

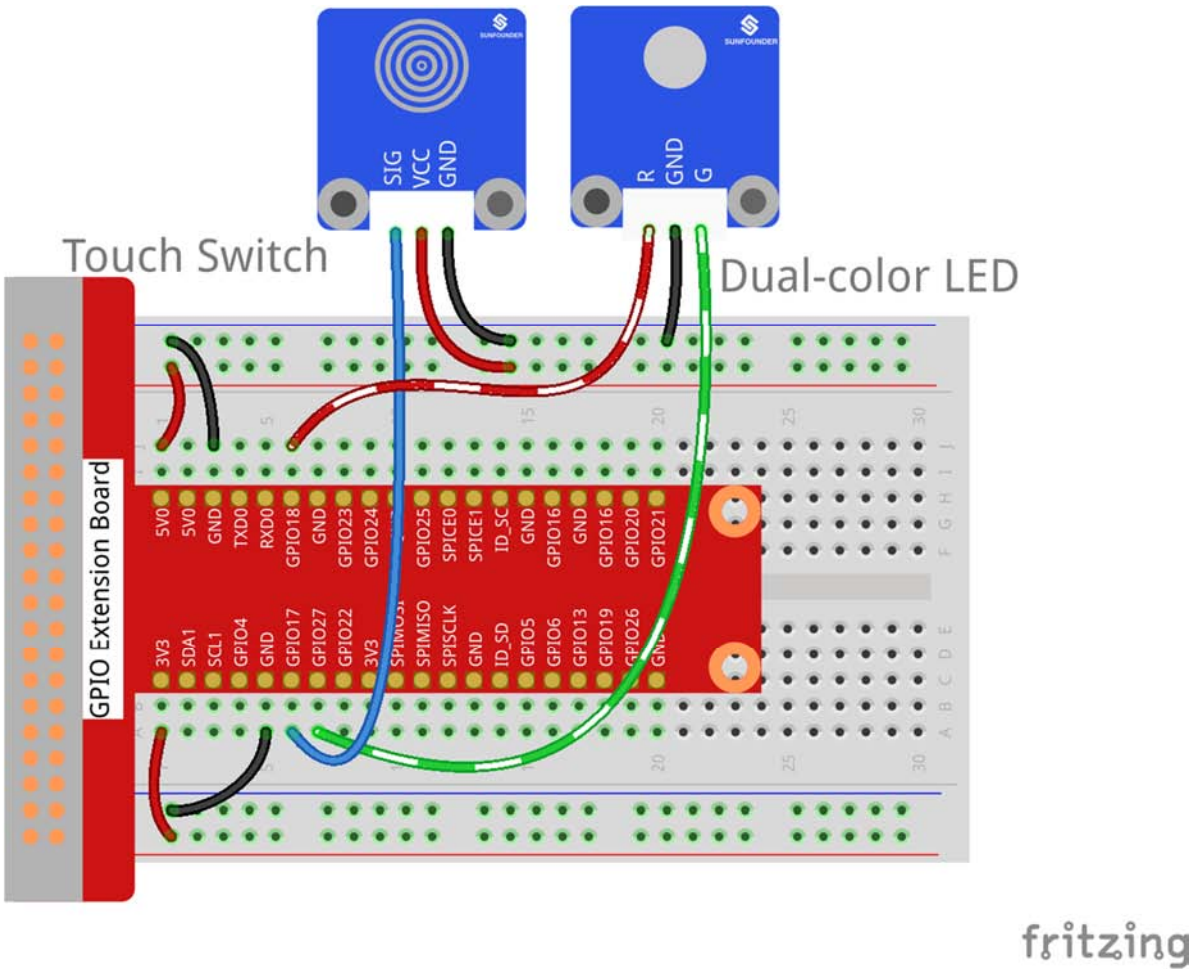


Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	Touch Sensor Module
GPIO0	GPIO17	SIG
5V	5V0	VCC
GND	GND	GND

Raspberry Pi	T-Cobbler	Dual-Color LED Module
GPIO1	GPIO18	R
GND	GND	GND
GPIO2	GPIO27	G



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/24_touch_switch/
```

Step 3: Compile

```
gcc touch_switch.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

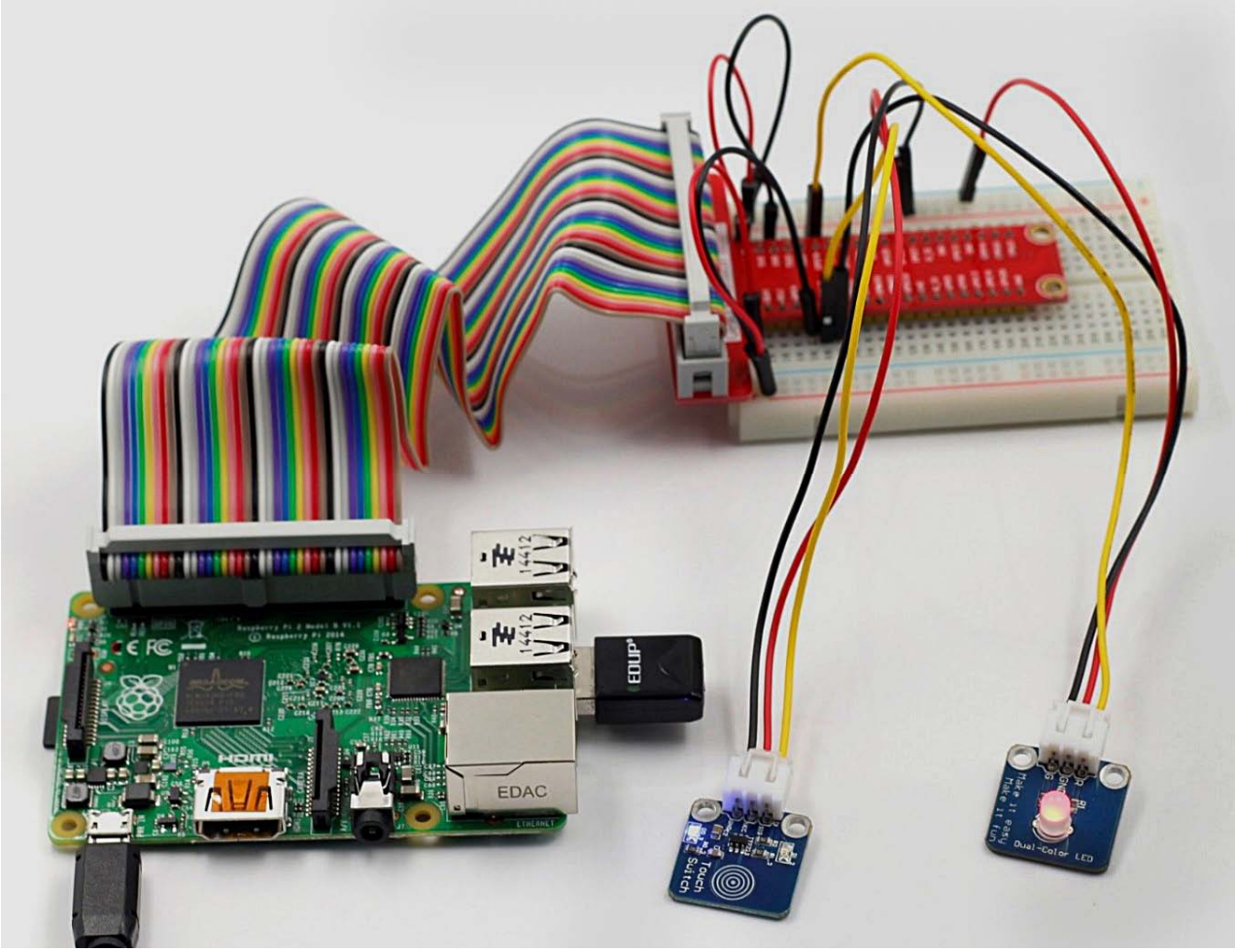
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 24_touch_switch.py
```

Now, touch the metal disk, you can see the LED change its colors and "ON" and "OFF" printed on the screen.



Lesson 25 Ultrasonic Ranging Module

Introduction

The ultrasonic sensor uses sound to accurately detect objects and measure distances. It sends out ultrasonic waves and converts them into electronic signals.

Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * Ultrasonic ranging module
- Several jumper wires (Male to Female)

Experimental Principle

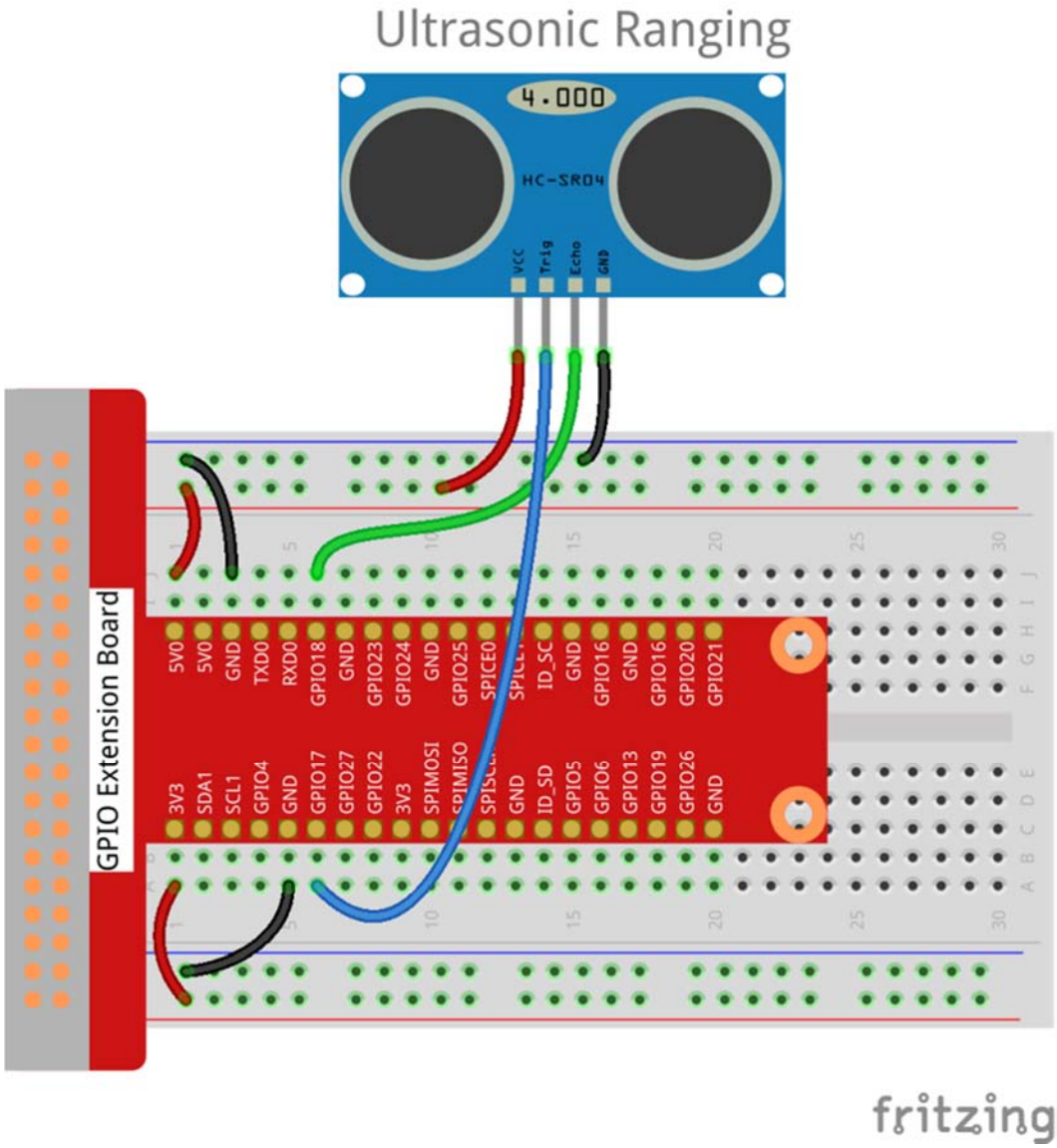
This sensor works by sending a sound wave out and calculating the time it takes for the sound wave to get back to the ultrasonic sensor. By doing this, it can tell us how far away objects are relative to the ultrasonic sensor.

Test distance = (high level time * velocity of sound (340M/S)) / 2 **(in meters)**

Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	Ultrasonic Ranging Module
5V	5V0	VCC
GPIO0	GPIO17	Trig
GPIO1	GPIO18	Echo
GND	GND	GND



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/25_ultrasonic_ranging/
```

Step 3: Compile

```
gcc ultrasonic_ranging.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

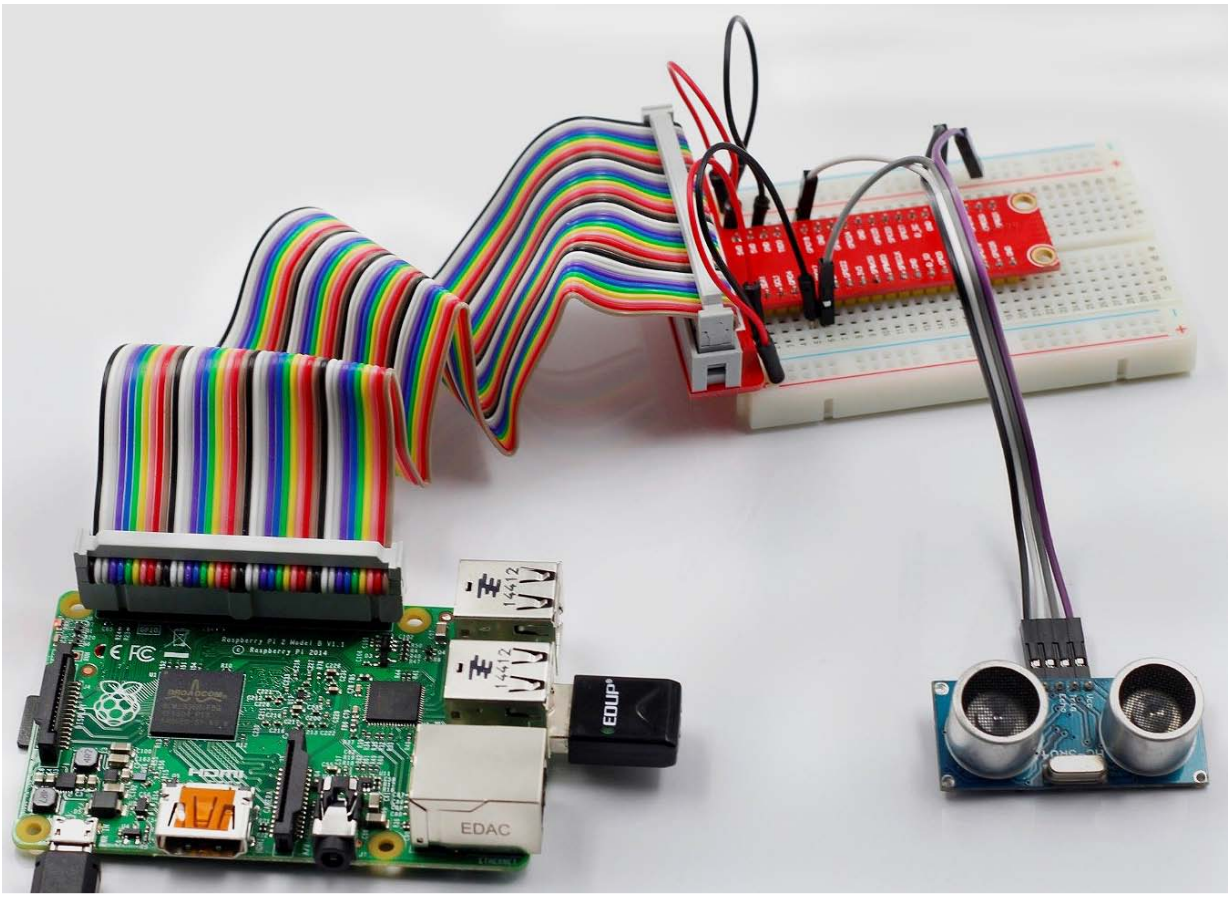
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 25_ultrasonic_ranging.py
```

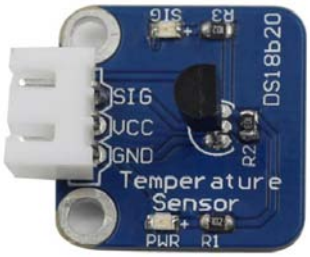
Now you can see the distance between the ultrasonic ranging module and the obstacle (like your palm) in front on the screen. Sway your hand over the ultrasonic ranging module slowly and observe the distance printed on the screen.



Lesson 26 DS18B20 Temperature Sensor

Introduction

Temperature Sensor DS18B20 is a commonly used digital temperature sensor featured with small size, low-cost hardware, strong anti-interference capability and high precision. The digital temperature sensor is easy to wire and can be applied a various occasions after packaging. Different from conventional AD collection temperature sensors, it uses a 1-wire bus and can directly output temperature data.



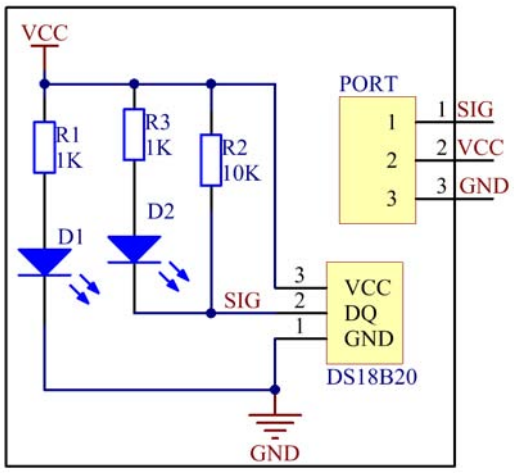
Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * DS18B20 Temperature Sensor module
- 1 * 3-Pin anti-reverse cable

Experimental Principle

With a unique single-wire interface, DS18B20 requires only one pin for a two-way communication with a microprocessor. It supports multi-point networking to measure multi-point temperatures. Eight sensors can be connected at most, because it will consume too much power supply and cause low voltage thus harming the stability of transmission.

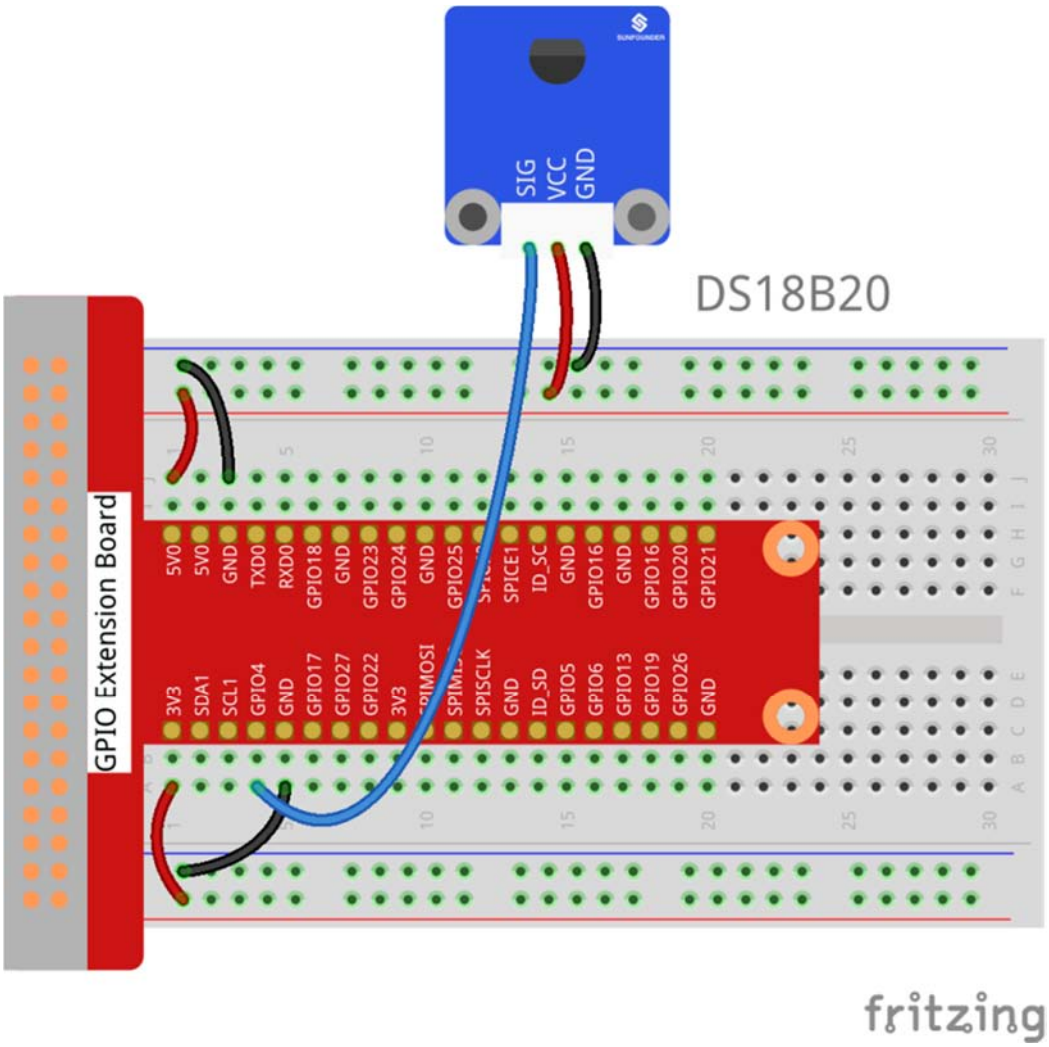
When using the DS18B20, you need to connect a 10KΩ resistor to the middle pin DQ to pull up the level. The schematic diagram of the module is as shown below:



Experimental Procedures

Step 1 : Build the circuit according to the following method

Raspberry Pi	T-Cobbler	DS18B20 Temperature Sensor
GPIO7	GPIO4	SIG
5V	5V0	VCC
GND	GND	GND



Step 2 : Upgrade your kernel

```
sudo apt-get update
sudo apt-get upgrade
```

Step 3 : You can edit that file with nano

```
sudo nano /boot/config.txt
```

Then scroll to the bottom and type

```
dtoverlay=w1-gpio
```

Then reboot with

```
sudo reboot.
```

Mount the device drivers and confirm whether the device is effective or not

```
sudo modprobe w1-gpio
sudo modprobe w1-therm
cd /sys/bus/w1/devices/
ls
```

The result is as follows:

```
root@rasberrypi:/sys/bus/w1/devices# ls
28-00000495db35 w1_bus_master1
```

28-00000495db35 is an external temperature sensor device, but it may vary with every client. This is the serial number of your ds18b20.

Step 4 : Check the current temperature

```
cd 28-00000495db35
ls
```

The result is as follows:

```
root@rasberrypi:/sys/bus/w1/devices/28-00000495db35# ls
driver  id  name    power   subsystem   uevent  w1_slave
```

```
cat w1_slave
```

The result is as follows:

```
root@raspberrypi:/sys/bus/w1_slave/28-00000495db35# cat w1_slave
a3 01 4b 46 7f ff 0d 10 ce : crc=ce YES
a3 01 4b 46 7f ff 0d 10 ce t=26187
```

The second line t=26187 is current temperature value. If you want to convert it to degree Celsius, you can divide by 1000, that is, the current temperature is 26187/1000=26.187 ℃.

For C language users:

Step 2: Change directory and edit

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/26_ds18b20/
nano ds18b20.c
```

Find the following line, replace "28-031467805fff" with your sensor address. Save and exit.

```
char* addr = "/sys/bus/w1/devices/28-031467805fff/w1_slave";
```

Step 6: Compile

```
gcc ds18b20.c
```

Step 7: Run

```
sudo ./a.out
```

For Python users:

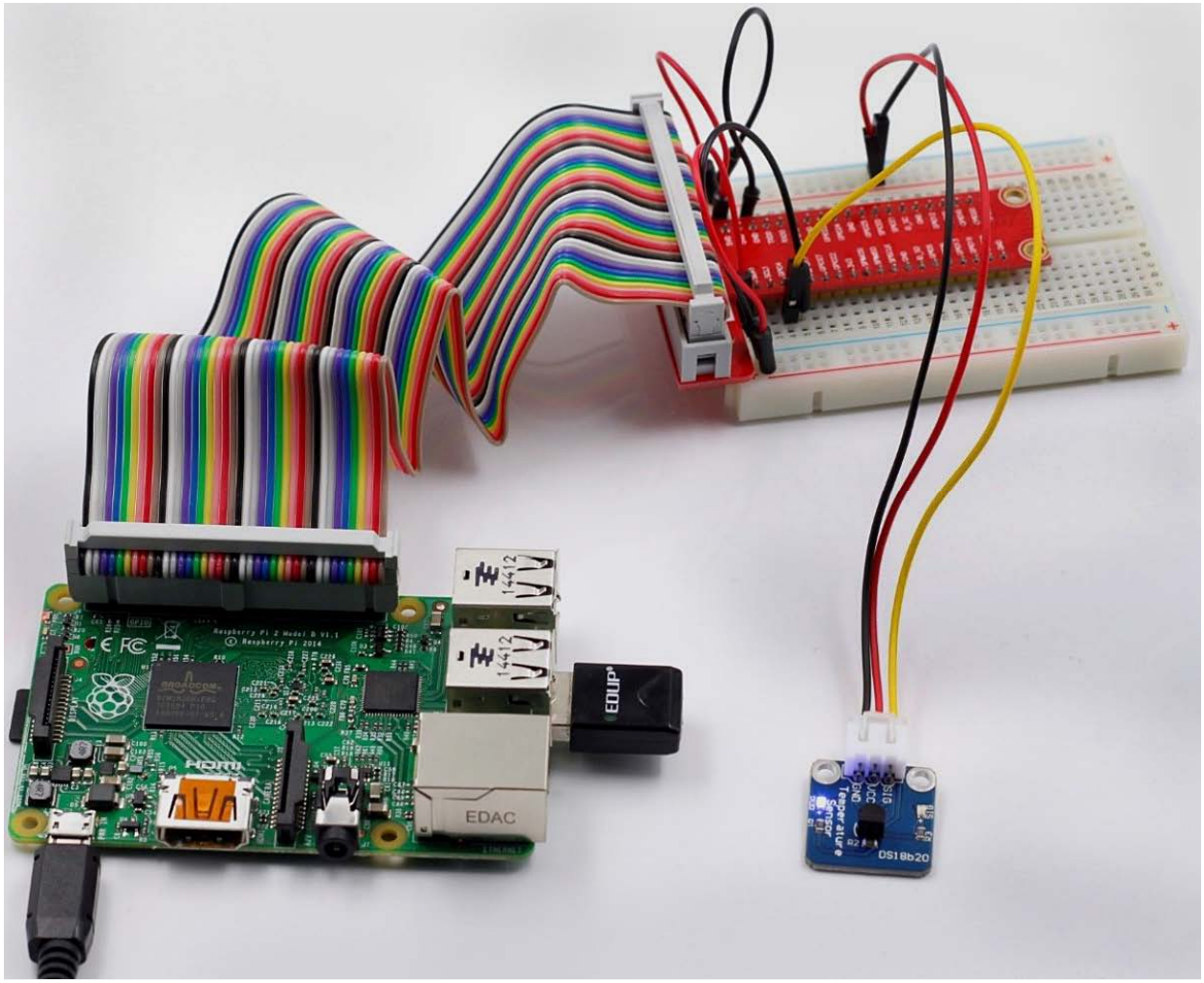
Step 5: Change directory and edit

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
nano 26_ds18b20.py
```

Step 6: Run

```
sudo python 26_ds18b20.py
```

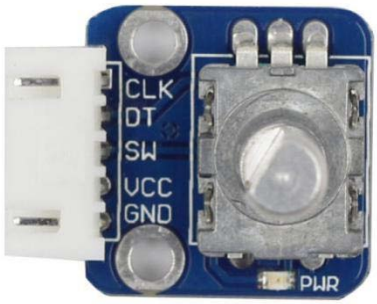
Now, you can see the current temperature value displayed on the screen.



Lesson 27 Rotary Encoder Module

Introduction

A rotary encoder is an electro-mechanical device that converts the angular position or motion of a shaft or axle to analog or digital code. Rotary encoders are usually placed at the side which is perpendicular to the shaft. They act as sensors for detecting angle, speed, length, position, and acceleration in automation field.



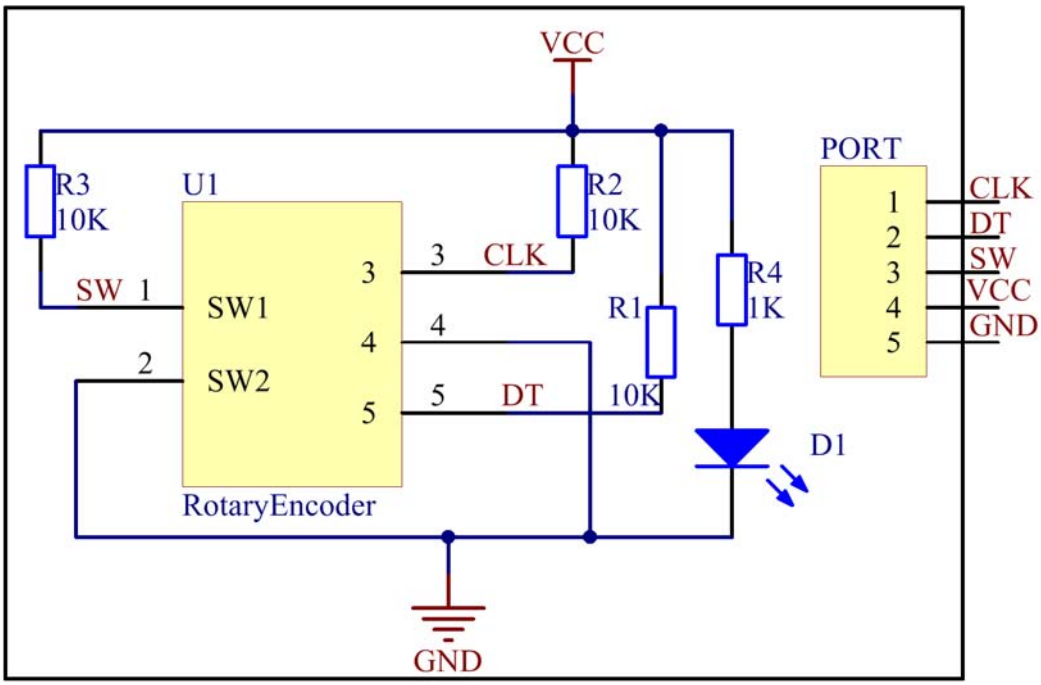
Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * Rotary Encoder module
- 1 * 5-Pin anti-reverse cable

Experimental Principle

Most rotary encoders have 5 pins with three functions of turning left & right and pressing down. Pin 1 and pin 2 are switch wiring terminals used to press. They are similar to buttons previously mentioned, so we will no longer discuss them in this experiment. Pin 4 is generally connected to ground. Pin 3 and pin 5 are first connected to pull-up resistor and then to the microprocessor. In this experiment, they are connected to GPIO0 and GPIO1 of Raspberry Pi. When it is rotated left and right, there will be pulse inputs in pin 1 and pin 3.

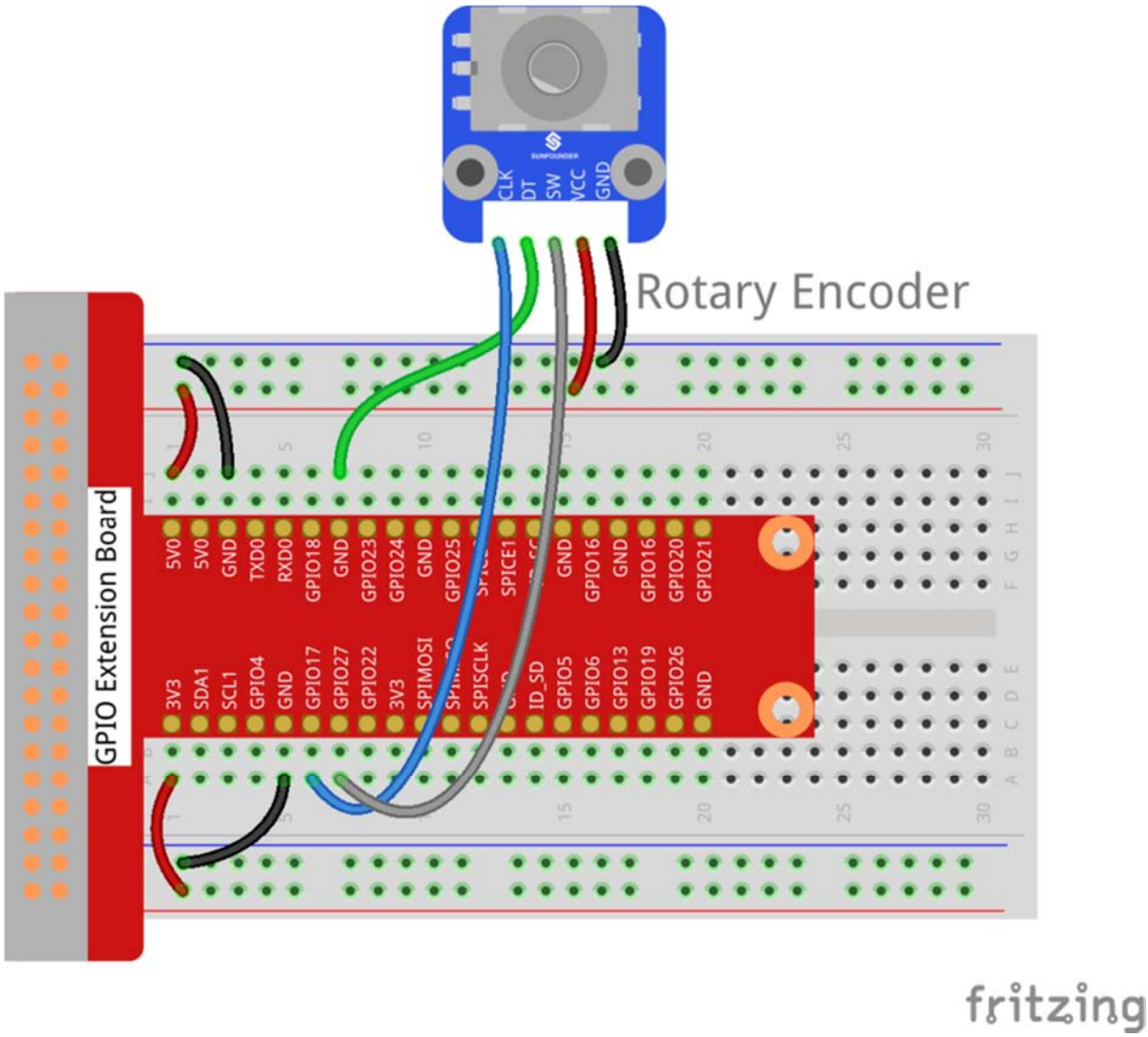
It shows that if output 1 is high and output 2 is high, then the switch rotates clockwise; if output 1 is high and output 2 is low, then the switch rotates counterclockwise. As a result, during SCM programming, if output 1 is high, then you can tell whether the rotary encoder rotates left or right as long as you know the state of output 2.



Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	Rotary Encoder Module
GPIO0	GPIO17	CLK
GPIO1	GPIO18	DT
GPIO2	GPIO27	SW
5V	5V0	VCC
GND	GND	GND



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/27_rotary_encoder/
```

Step 3: Compile

```
gcc rotary_encoder.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

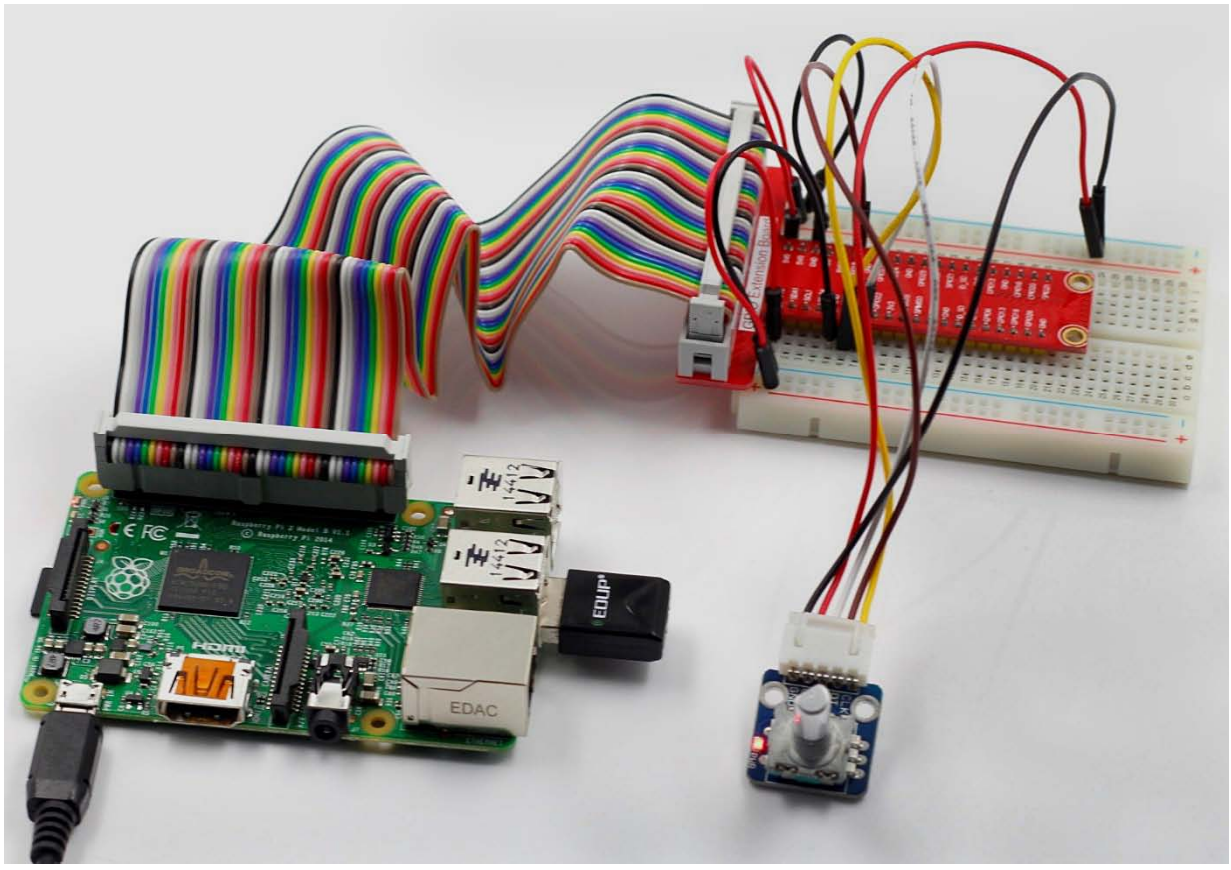
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 27_rotary_encoder.py
```

Now rotate the shaft of the rotary encoder, and the value printed on the screen will change. Rotate the rotary encoder clockwise, the value will increase; Rotate it counterclockwise, the value will decrease; Press the rotary encoder, the value will be reset to 0.

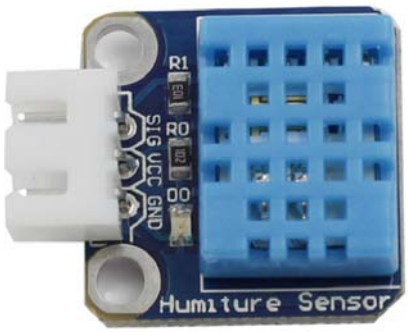


Lesson 28 Humiture Sensor

Introduction

The digital temperature and humidity sensor DHT11 is a composite sensor that contains a calibrated digital signal output of temperature and humidity. The technology of a dedicated digital modules collection and the temperature and humidity sensing technology are applied to ensure that the product has high reliability and excellent long-term stability.

The sensor includes a resistive sense of wet component and an NTC temperature measurement device, and is connected with a high-performance 8-bit microcontroller.

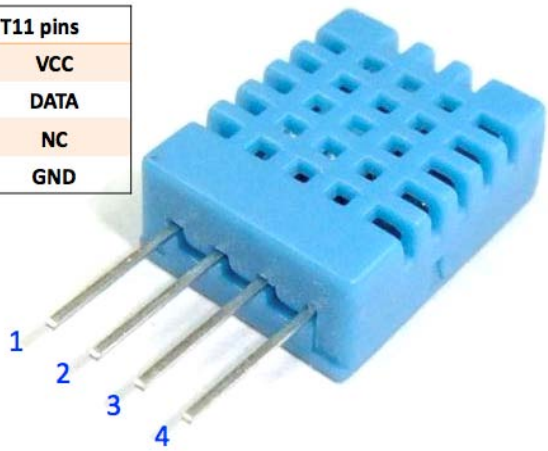


Components

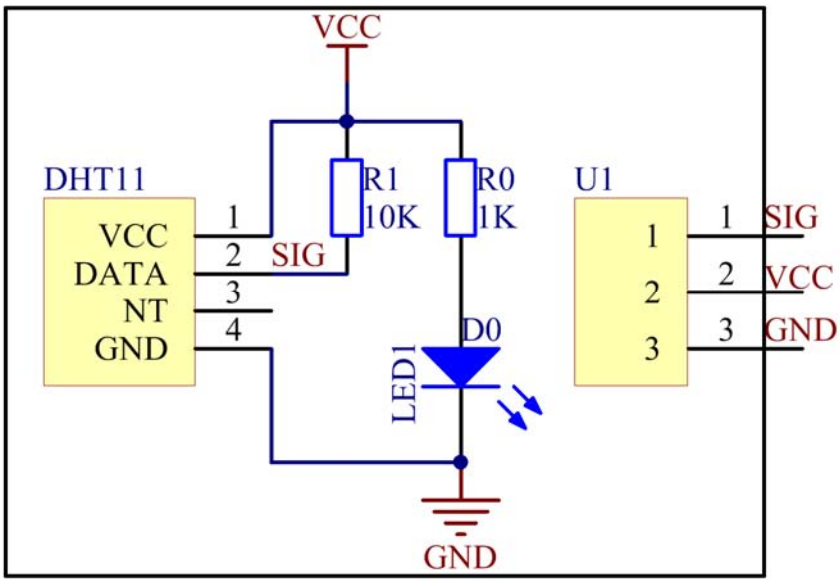
- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * Humiture module
- 1 * 3-Pin anti-reverse cable

Experimental Principle

DHT11 pins	
1	VCC
2	DATA
3	NC
4	GND



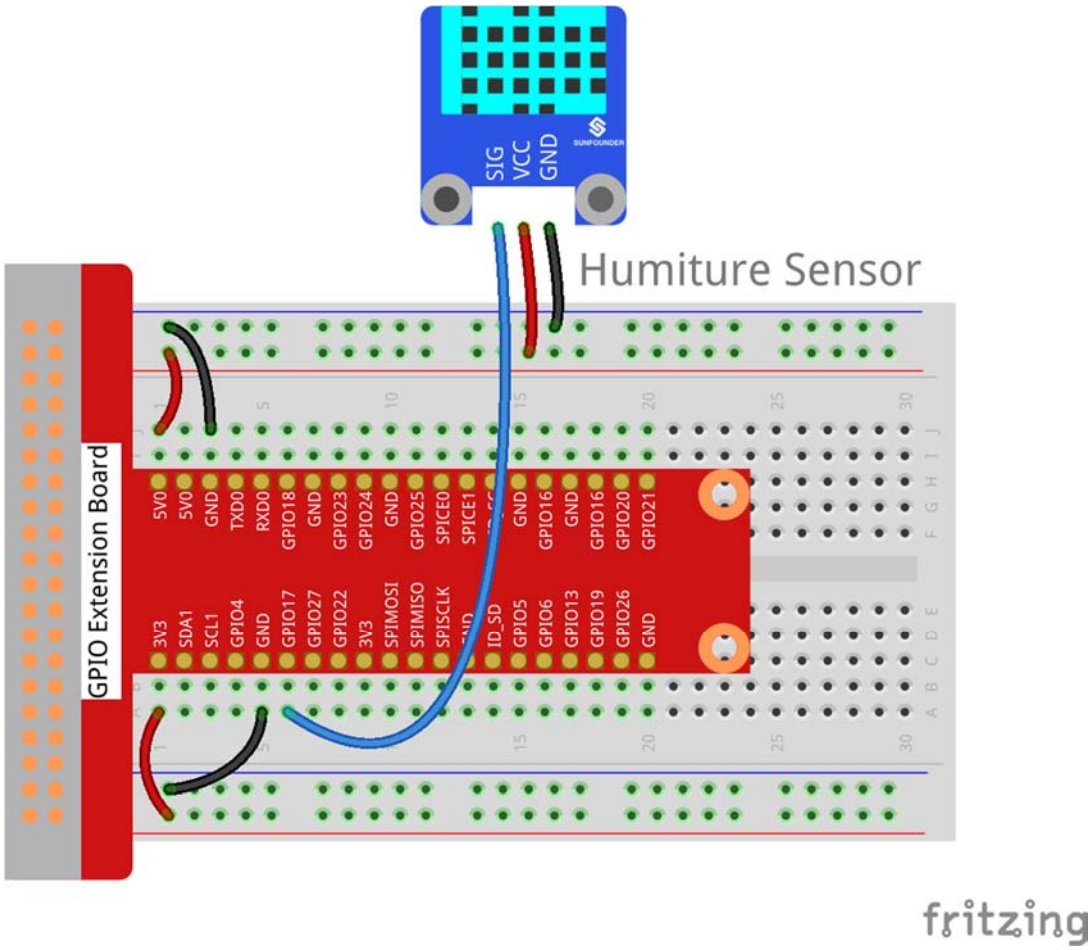
Only three pins are available for use: VCC, GND, and DATA. The communication process begins with the DATA line sending start signal to DHT11, and DHT11 receives the signal and returns an answer signal, then the host receives the answer signal and begins to receive 40-bit humiture data (8-bit humidity integer + 8-bit humidity decimal + 8-bit temperature integer + 8-bit temperature decimal + 8-bit checksum). For more information, please refer to the datasheet of DHT11.



Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	Humiture Module
GPIO0	GPIO17	SIG
5V	5V0	VCC
GND	GND	GND



Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/28_humiture/
```

Step 3: Compile

```
gcc humiture.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/SunFounder_DHT11/
```

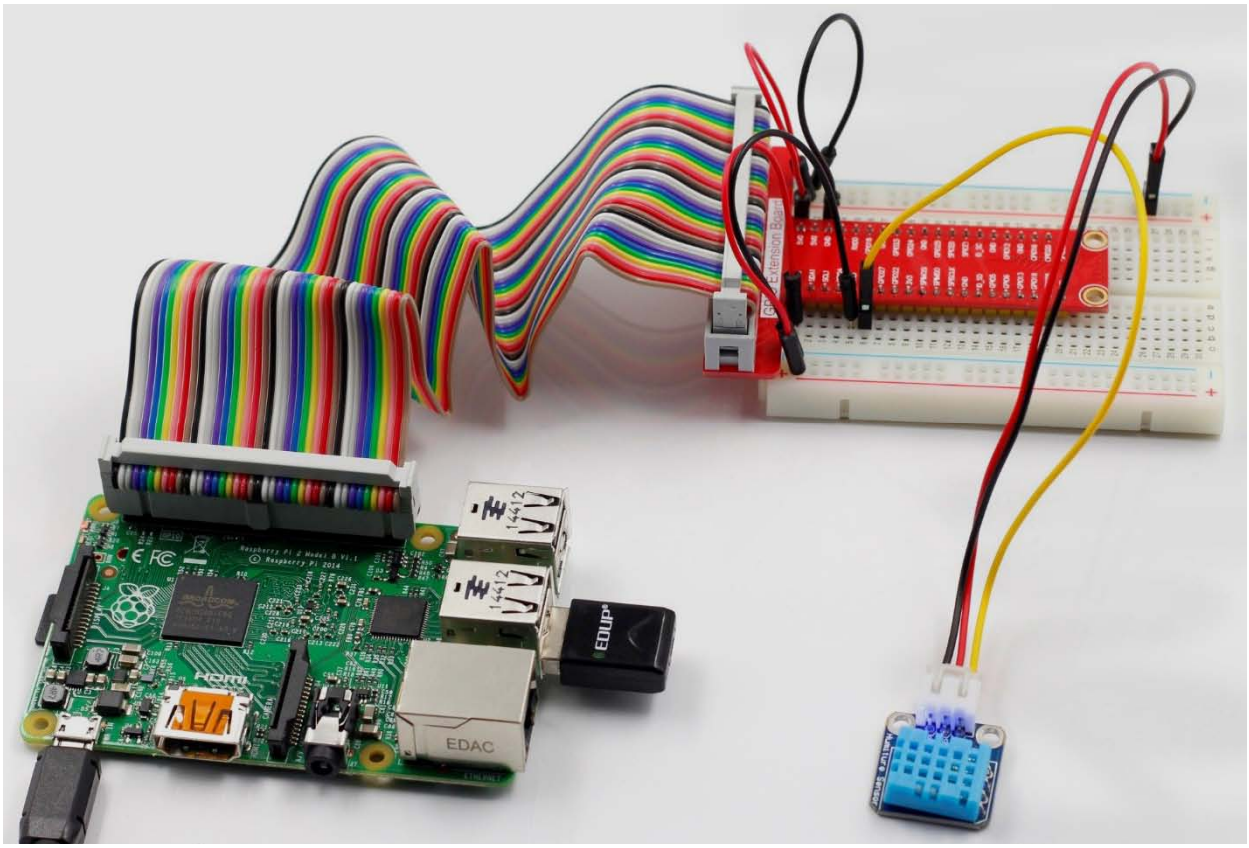
Step 3：Update submodule SunFounder_DHT11

```
git submodule update --init
```

Step 4: Run

```
sudo python dht11.py
```

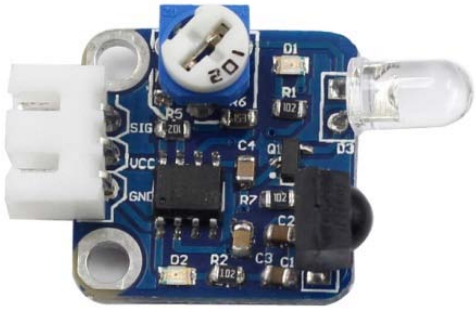
Now, you can see humidity and temperature value printed on the screen.


EDAC

Lesson 29 IR Obstacle Avoidance Module

Introduction

An IR obstacle avoidance module (as shown below) uses infrared reflection principle to detect obstacles. When there is no object ahead, infrared-receiver cannot receive signals; when there is an object ahead, it will block and reflect infrared light, then infrared-receiver can receive signals.



Components

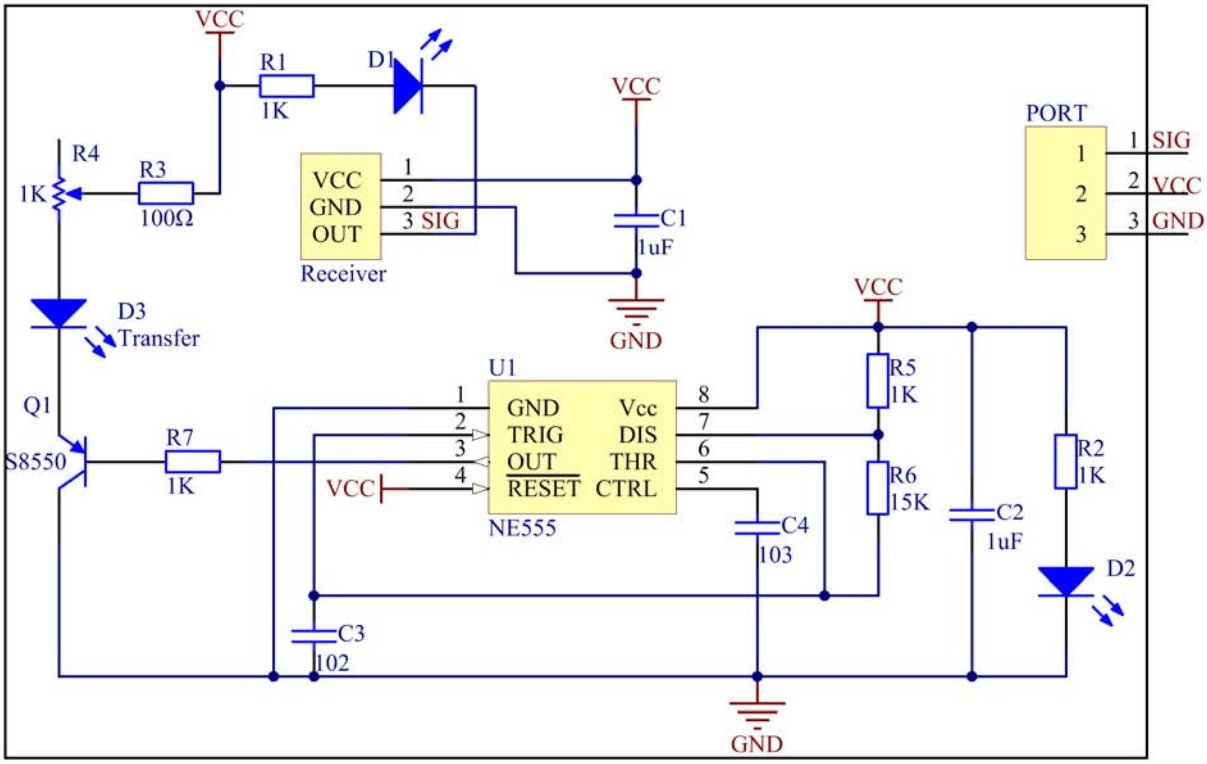
- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * IR Obstacle module
- 1 * 3-Pin anti-reverse cable

Experimental Principle

An obstacle avoidance sensor mainly consists of an infrared-transmitter, an infrared-receiver and a potentiometer. According to the reflecting feature of an object, if there is no obstacle, emitted infrared ray will weaken with the propagation distance and finally disappear. If there is an obstacle, when infrared ray encounters an obstacle, it will be reflected back to the infrared-receiver. Then the infrared-receiver detects this signal and confirms an obstacle exists ahead.

Note: The detection distance of the infrared sensor is adjustable - you may adjust it by the potentiometer.

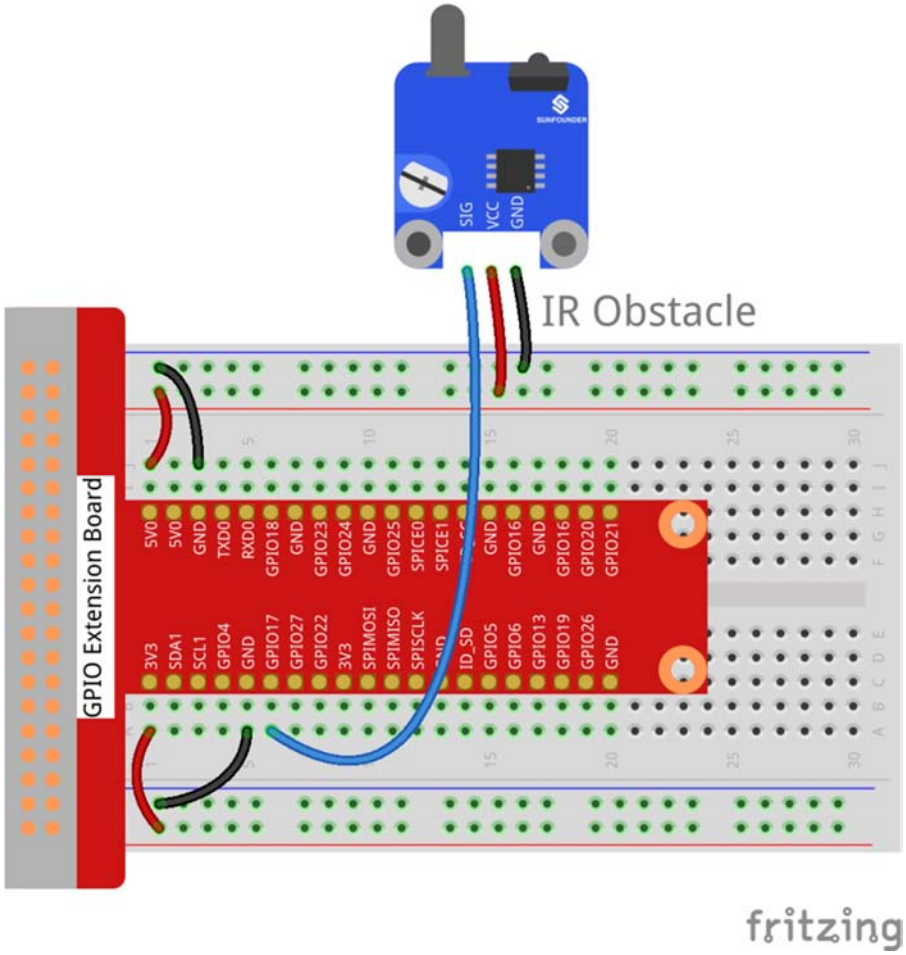
The schematic diagram:



Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	IR Obstacle Module
GPIO0	GPIO17	SIG
5V	5V0	VCC
GND	GND	GND



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/30_ir_obstacle/
```

Step 3: Compile

```
gcc ir_obstacle.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

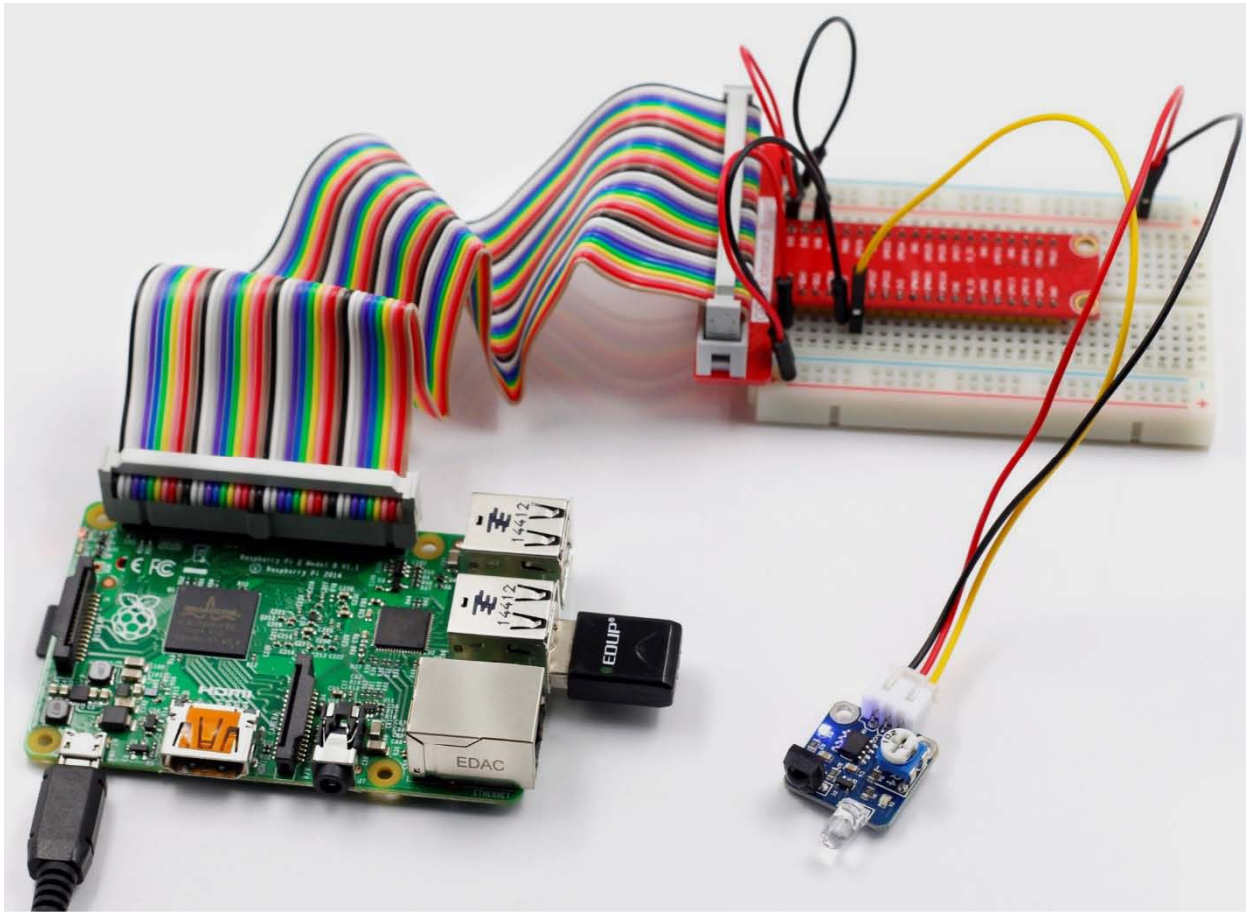
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 30_ir_obstacle.py
```

Now, if there is an obstacle ahead, a string “Detected Barrier!” will be printed on the screen.

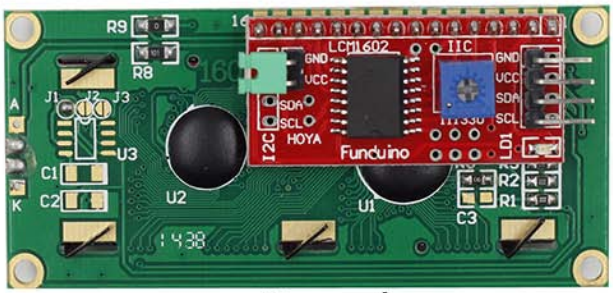


Lesson 30 I2C LCD1602

Introduction

LCD1602 is a character type liquid crystal display, which can display 32 (16*2) characters at the same time. It has 16 pins, of which at least 7 would be used each time. You can use a PCF8574 I2C chip to expand I/O ports so only two GPIO ports would be occupied.

Front



Back

Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * I2C LCD1602
- Several jumper wires (Male to Female)

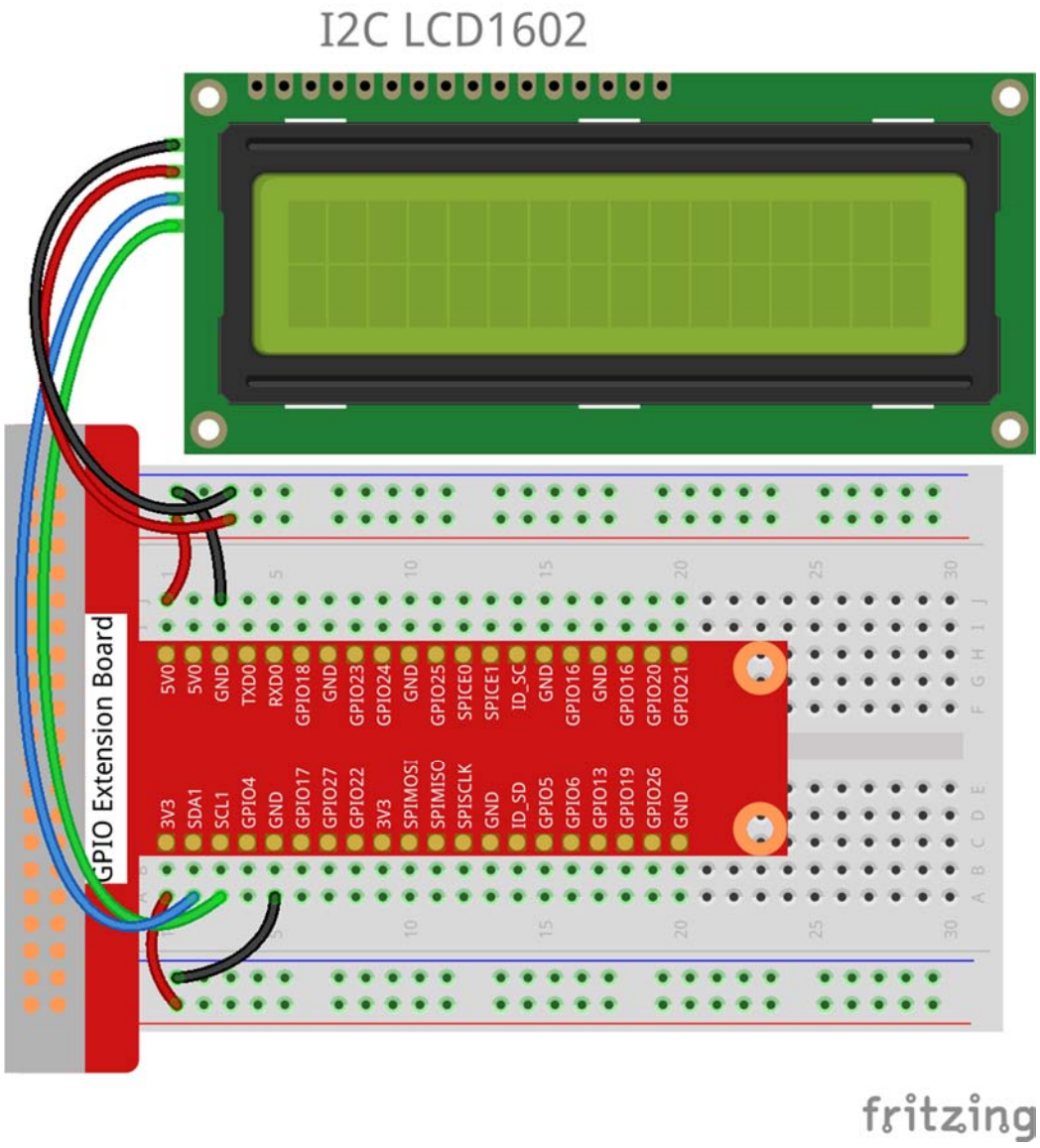
Experimental Principle

In this experiment, I2C is used to configure LCD so that you can control the LCD1602 to display characters. The I2C slave address of I2C LCD1602 here is 0x27.

Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	I2C LCD1602 Module
SCL	SCL1	SCL
SDA	SDA1	SDA
5V	5V0	VCC
GND	GND	GND



Step 2: Setup I2C (see Appendix 1. If you have set I2C, skip this step.)

For C language users:

Step 3: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/30_i2c_lcd1602/
```

Step 4: Compile

```
gcc i2c_lcd1602.c -lwiringPi
```

Step 5: Run

```
sudo ./a.out
```

For Python users:

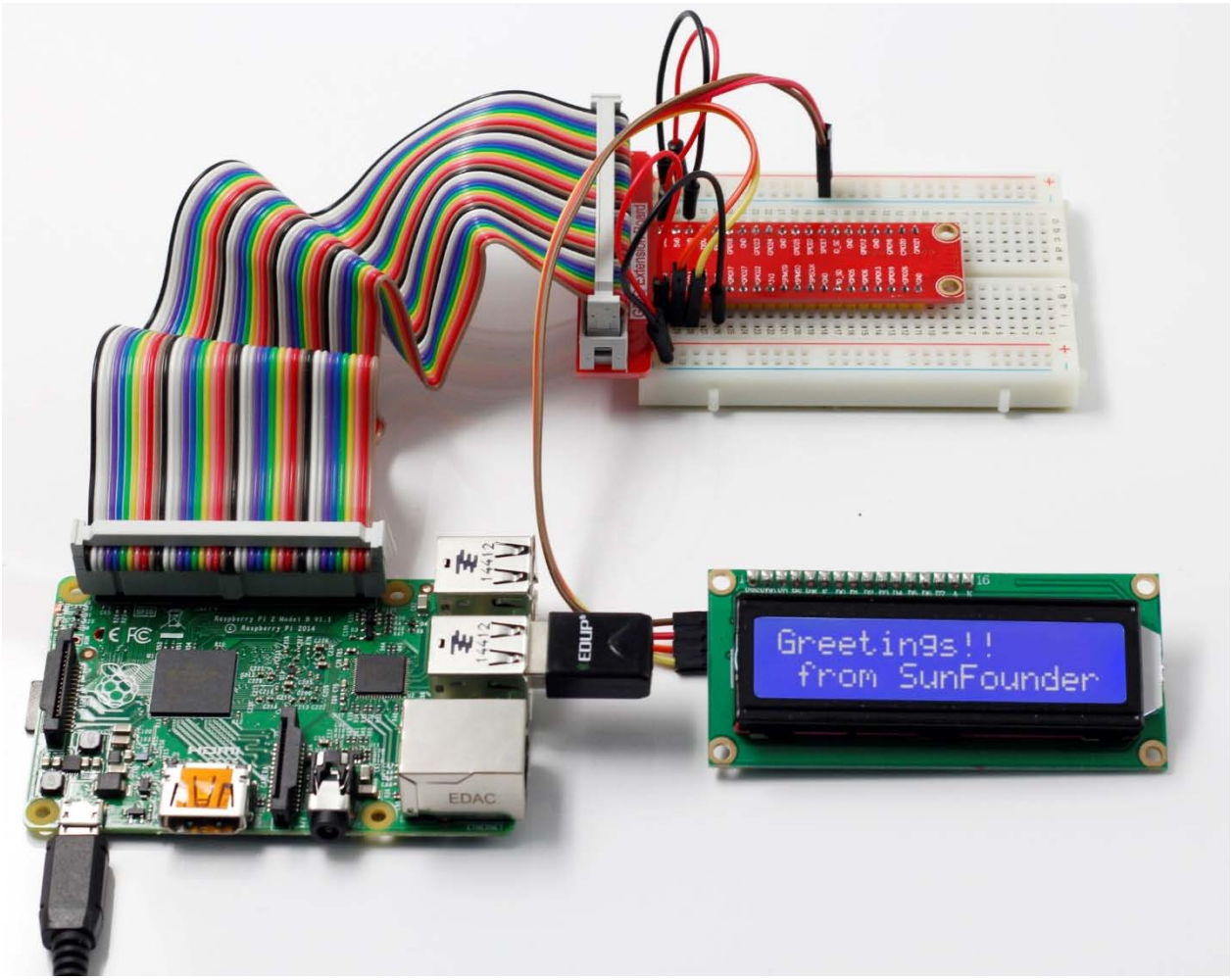
Step 3: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 4: Run

```
sudo python 30_i2c_lcd1602.py
```

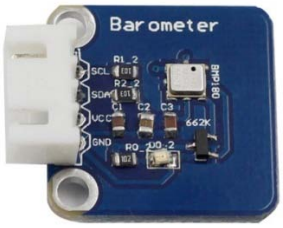
Now you can see "Greetings! From SunFounder" displayed on the LCD.



Lesson 31 Barometer

Introduction

The BMP180 barometer is the new digital barometric pressure sensor, with a very high performance, which enables applications in advanced mobile devices, such as smart phones, tablets and sports devices. It complies with the BMP085 but boasts many improvements, like a smaller size and more digital interfaces.



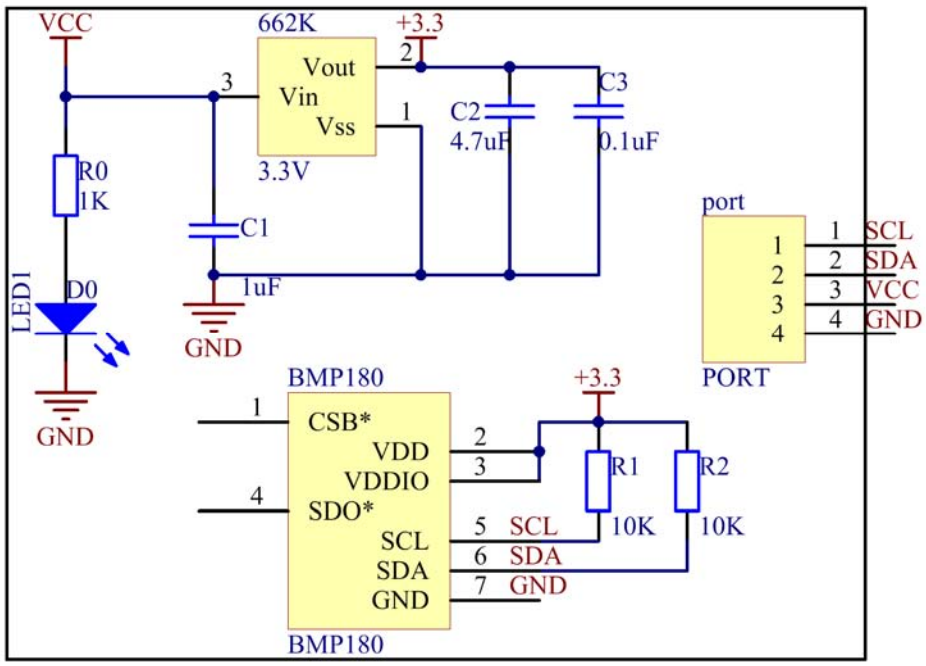
Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * Barometer module
- 1 * 4-Pin anti-reverse cable

Experimental Principle

Use a barometer to measure air pressure and temperature.

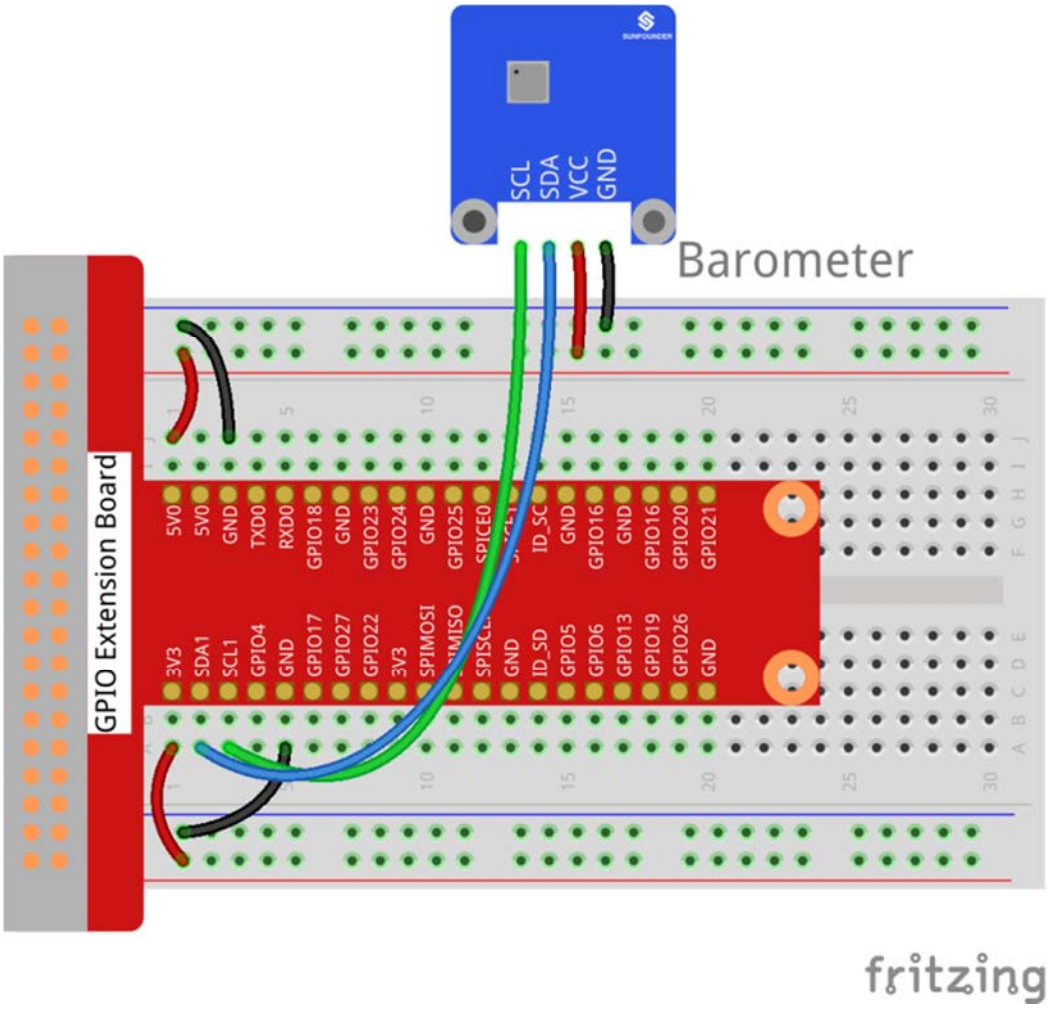
The schematic diagram:



Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	Barometer Module
SCL	SCL1	SCL
SDA	SDA1	SDA
5V	5V0	VCC
GND	GND	GND



Step2: Setup I2C (see Appendix 1. If you have set I2C, skip this step.)

For C language users:

Step 3: Download libi2c-dev

```
sudo apt-get install libi2c-dev
```

Step 4: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/31_barometer/
```

Step 5: Compile

```
gcc barometer.c bmp180.c -lm
```

Step 6: Run

```
sudo ./a.out
```

For Python users:

Step 3: Install smbus for I2C

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

Step 4: We'll need to install some utilities for the Raspberry Pi to communicate over I2C.

```
sudo apt-get install build-essential python-dev python-smbus
cd ~
git clone https://github.com/adafruit/Adafruit_Python_BMP.git
cd Adafruit_Python_BMP
sudo python setup.py install
```

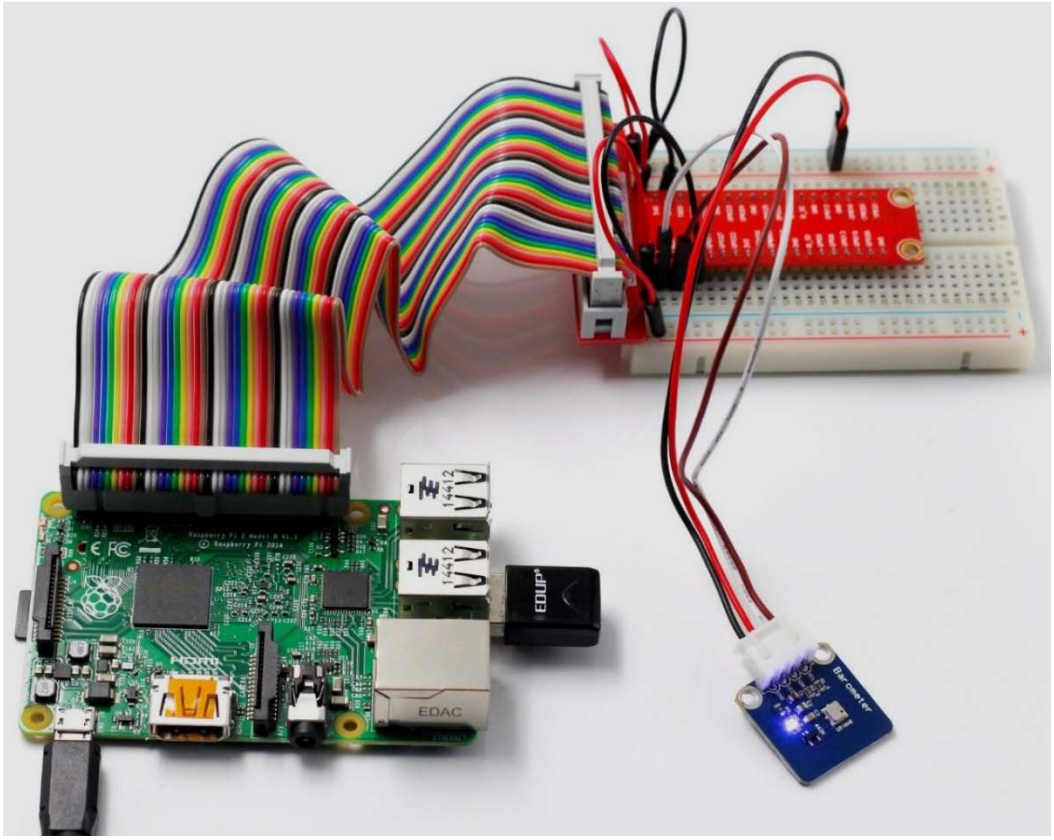
Step 5: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 6: Run

```
sudo python 31_barometer.py
```

Now you can see the temperature and pressure value displayed on the screen.



Lesson 32 MPU6050 Gyro Acceleration Sensor

Introduction

The MPU-6050 is the world's first and only 6-axis motion tracking devices designed for the low power, low cost, and high performance requirements of smartphones, tablets and wearable sensors.

Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * MPU-6050 module
- 4 * Jumper wires (M to F) (preferable to the 4-pin anti-reverse cable)

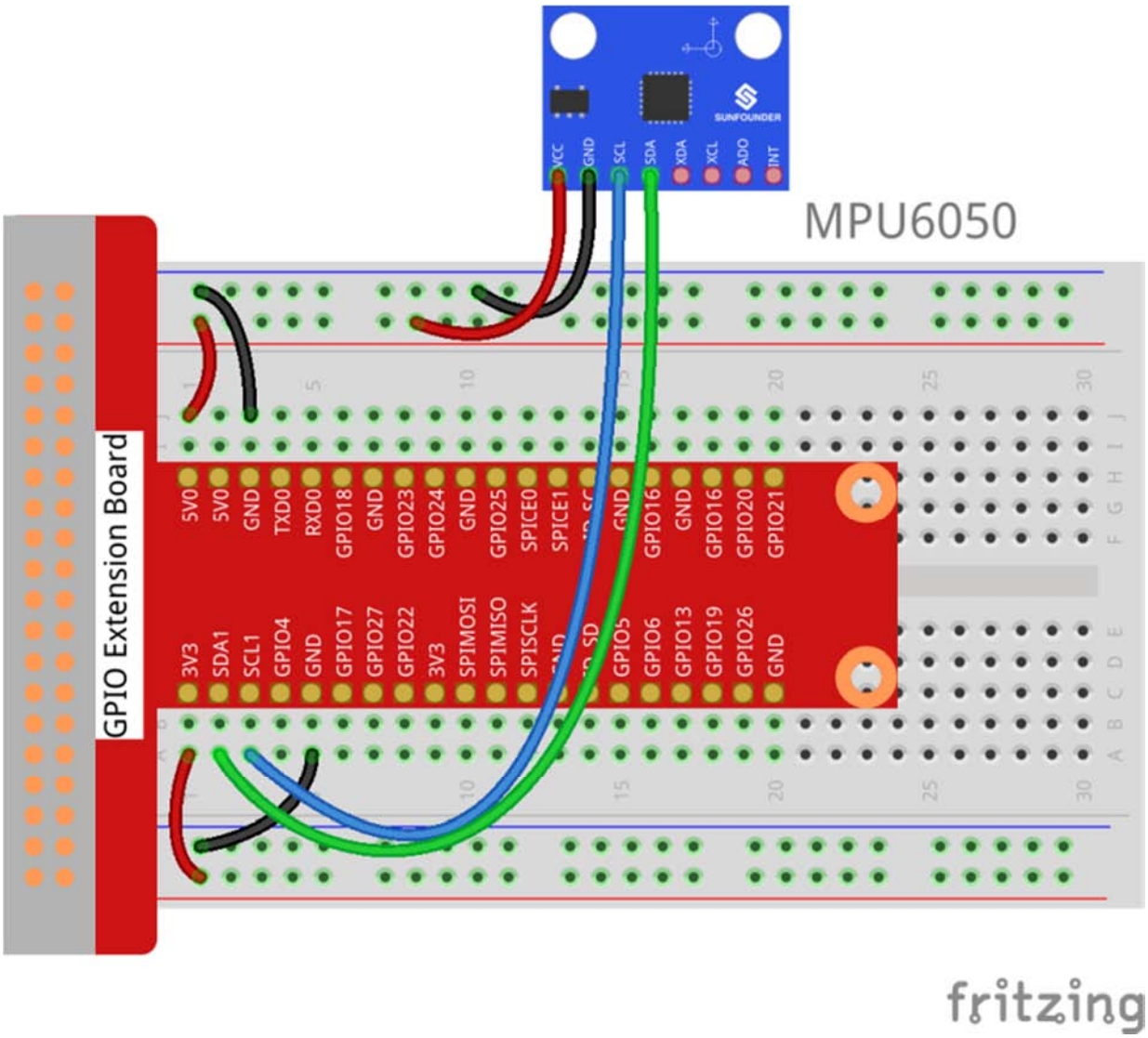
Experimental Principle

In this experiment, use I2C to obtain the values of the three-axis acceleration sensor and three-axis gyroscope for MPU6050 and display them on the screen.

Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	MPU-6050 Module
SCL	SCL1	SCL
SDA	SDA1	SDA
5V	5V0	VCC
GND	GND	GND



Step2: Setup I2C (see Appendix 1. If you have set I2C, skip this step.)

For C language users:

Step 3: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/32_mpu6050/
```

Step 4: Compile

```
gcc 32_mpu6050_accel.c -lwiringPi -lm
gcc 32_mpu6050_gyro.c -lwiringPi -lm
```

Step 5: Run

```
sudo ./a.out
```

For Python users:

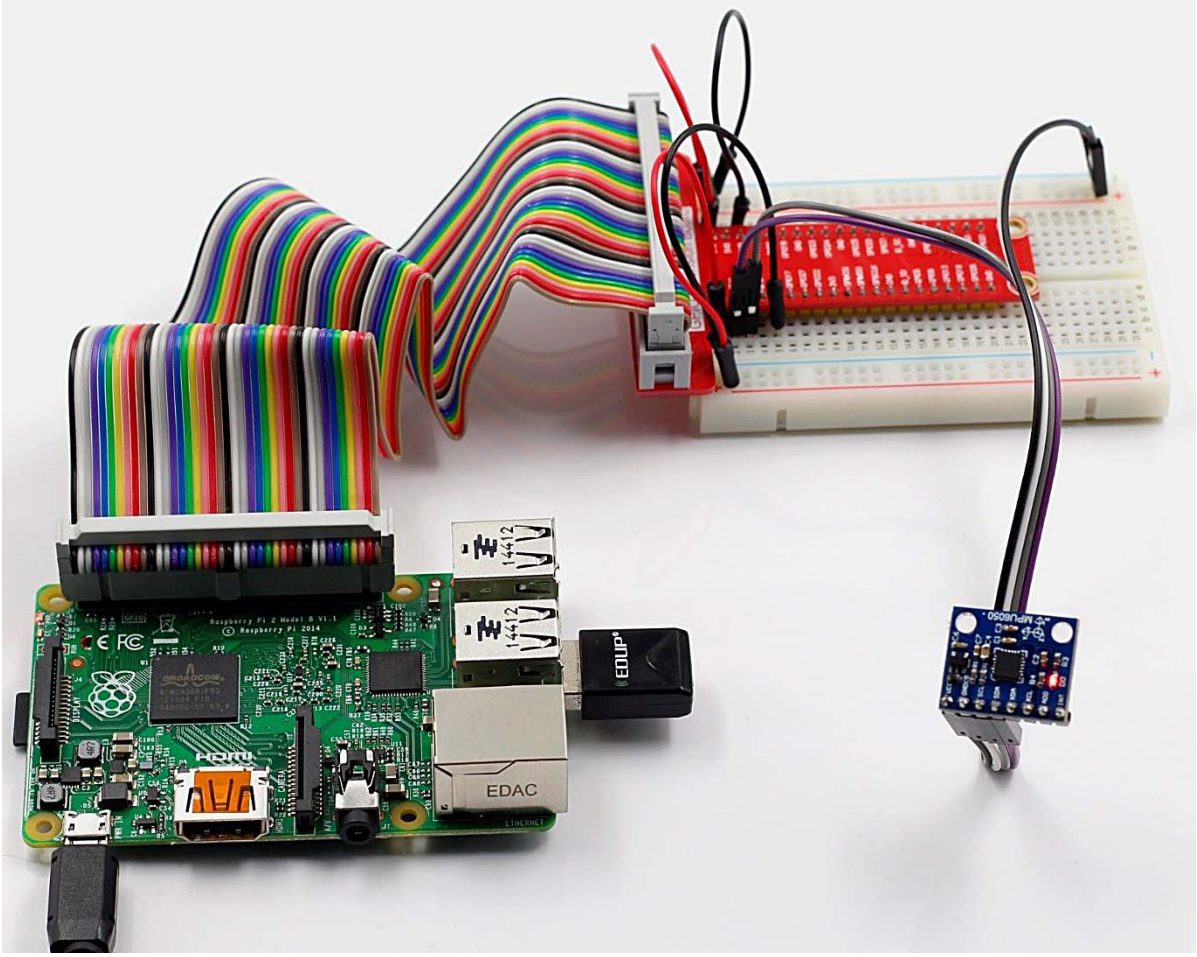
Step 3: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 4: Run

```
sudo python 32_mpu6050.py
```

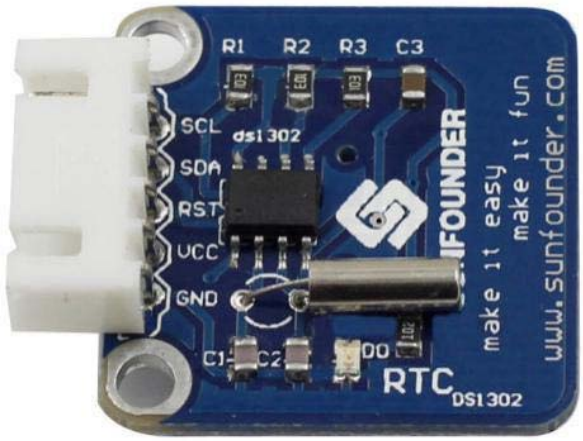
Now you can see the values of the acceleration sensor, gyroscope, and XY-axis rotation read by MPU6050 printed on the screen constantly.



Lesson 33 RTC DS1302

Introduction

DS1302 is a trickle charging clock chip, launched by DALLAS in America. With a built-in real-time clock/calendar and a 31-byte static RAM, it can communicate with MCU through simple serial interfaces. The real-time clock/calendar circuit provides information about second, minute, hour, day, week, month, and year. DS1302 can automatically adjust the number of days per month and days in leap year. You can determine to use a 24-hour or 12-hour system by AM/PM selection.



Components

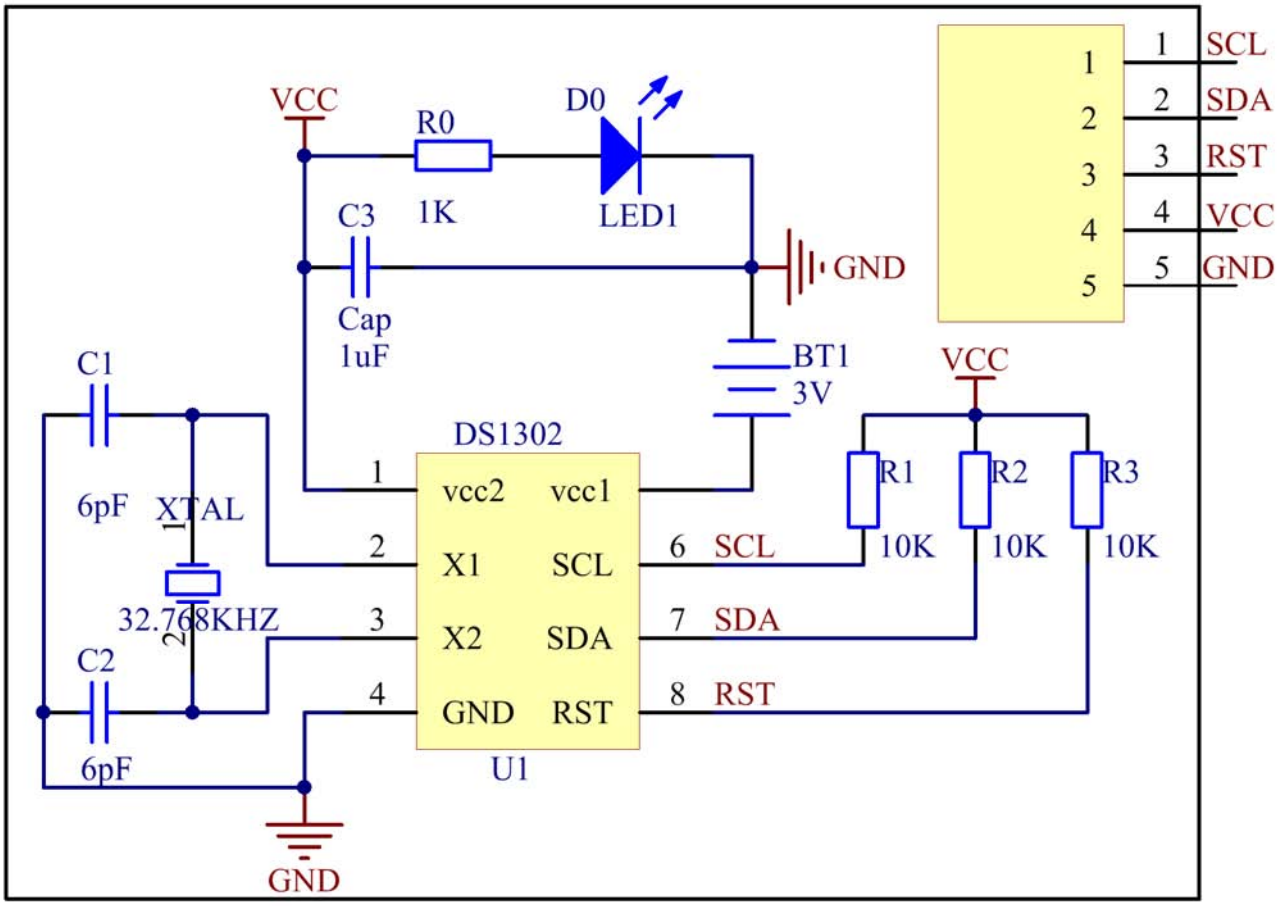
- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * DS1302 RTC module
- 1 * 5-Pin anti-reverse cable

Experimental Principle

Interfacing the DS1302 with a microprocessor is simplified by using synchronous serial communication. Only three wires are required to communicate with the clock/RAM: RST, serial data (SDA) and serial clock (SCL). SDA can be transferred to and from the clock/RAM one byte at a time or in a burst of up to 31 bytes.

After the time of the DS1302 is set manually, the MCU starts to read the accurate time and date returned by DS1302.

The schematic diagram:

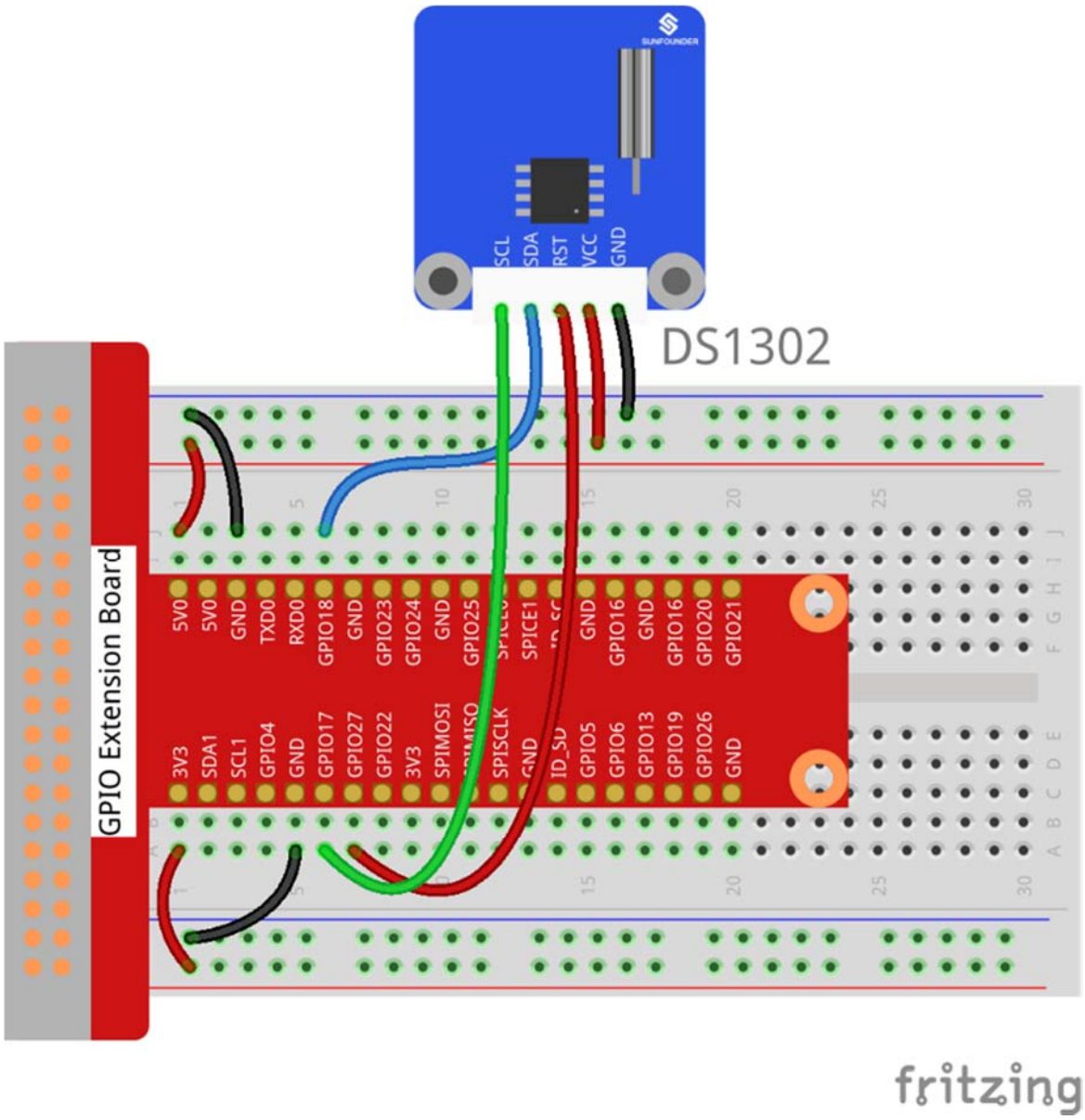


Experimental Procedures

For C language users:

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	RTC DS1302 Module
GPIO0	GPIO17	SCL
GPIO1	GPIO18	SDA
GPIO2	GPIO27	RST
5V	5V0	VCC
GND	GND	GND



Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/33_ds1302/
```

Step 3: Compile:

```
gcc rtc_ds1302.c -lwiringPi -lwiringPiDev
```

Step 4: Set up time by:

```
sudo ./a.out -sdsc
```

Set year, month, date as YYMMDD

Set hour, minute, second as HHMMSS(24-hour clock)

Set weekday (0 as Sunday)

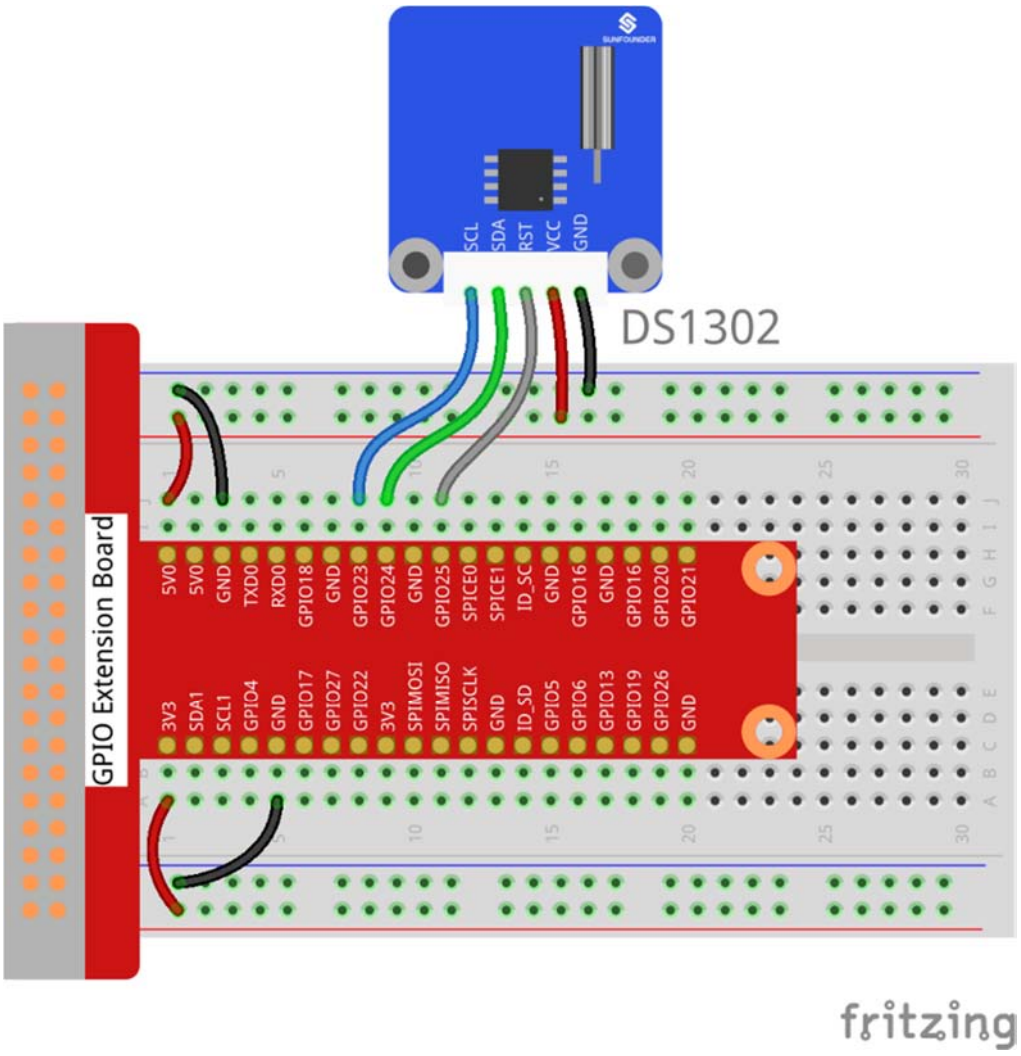
Step 5: Run:

```
sudo ./a.out
```

For Python users:

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	RTC DS1302 Module
GPIO4	GPIO23	SCL
GPIO5	GPIO24	SDA
GPIO6	GPIO25	RST
5V	5V0	VCC
GND	GND	GND



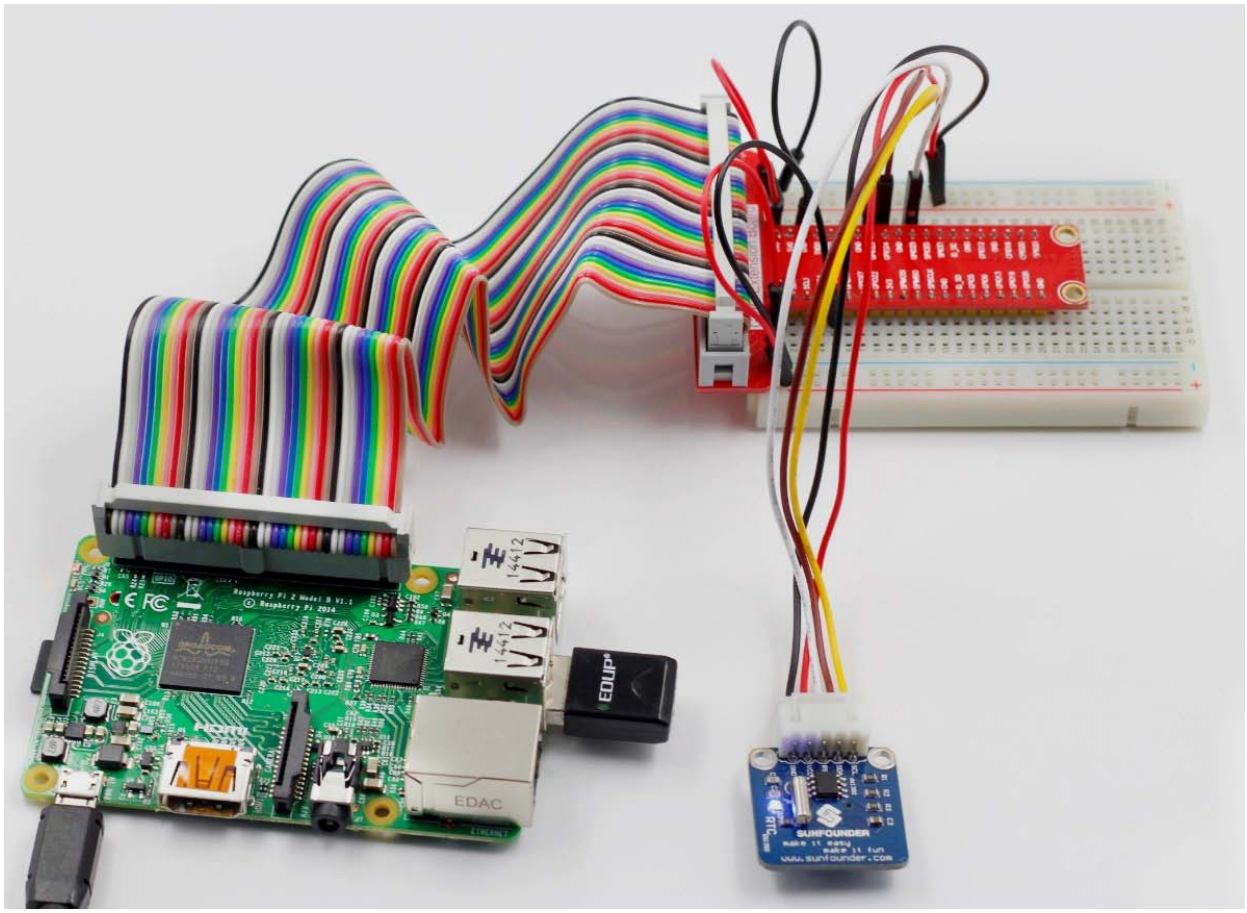
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 33_ds1302.py
```

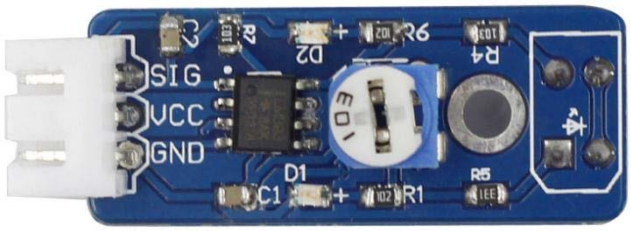
Now you can see the time on the screen.



Lesson 34 Tracking Sensor

Introduction

The infrared tracking sensor uses a TRT5000 sensor. The blue LED of TRT5000 is the emission tube and after electrified it emits infrared light invisible to human eye. The black part of the sensor is for receiving; the resistance of the resistor inside changes with the infrared light received.

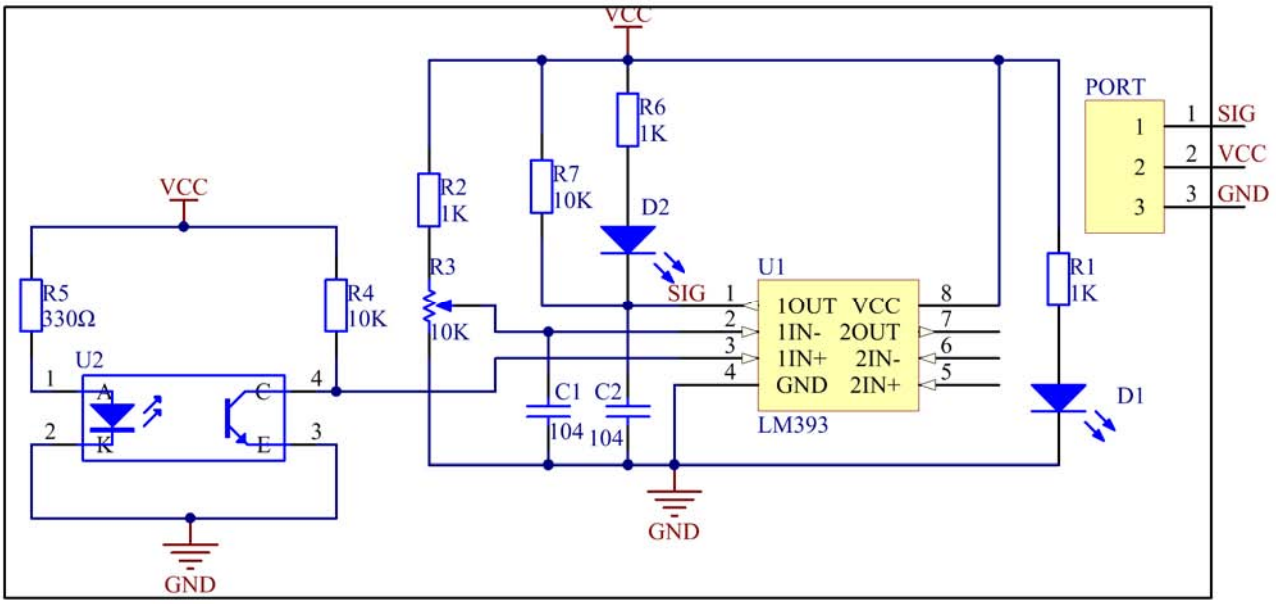


Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * Tracking sensor module
- 1 * 3-Pin anti-reverse cable

Experimental Principle

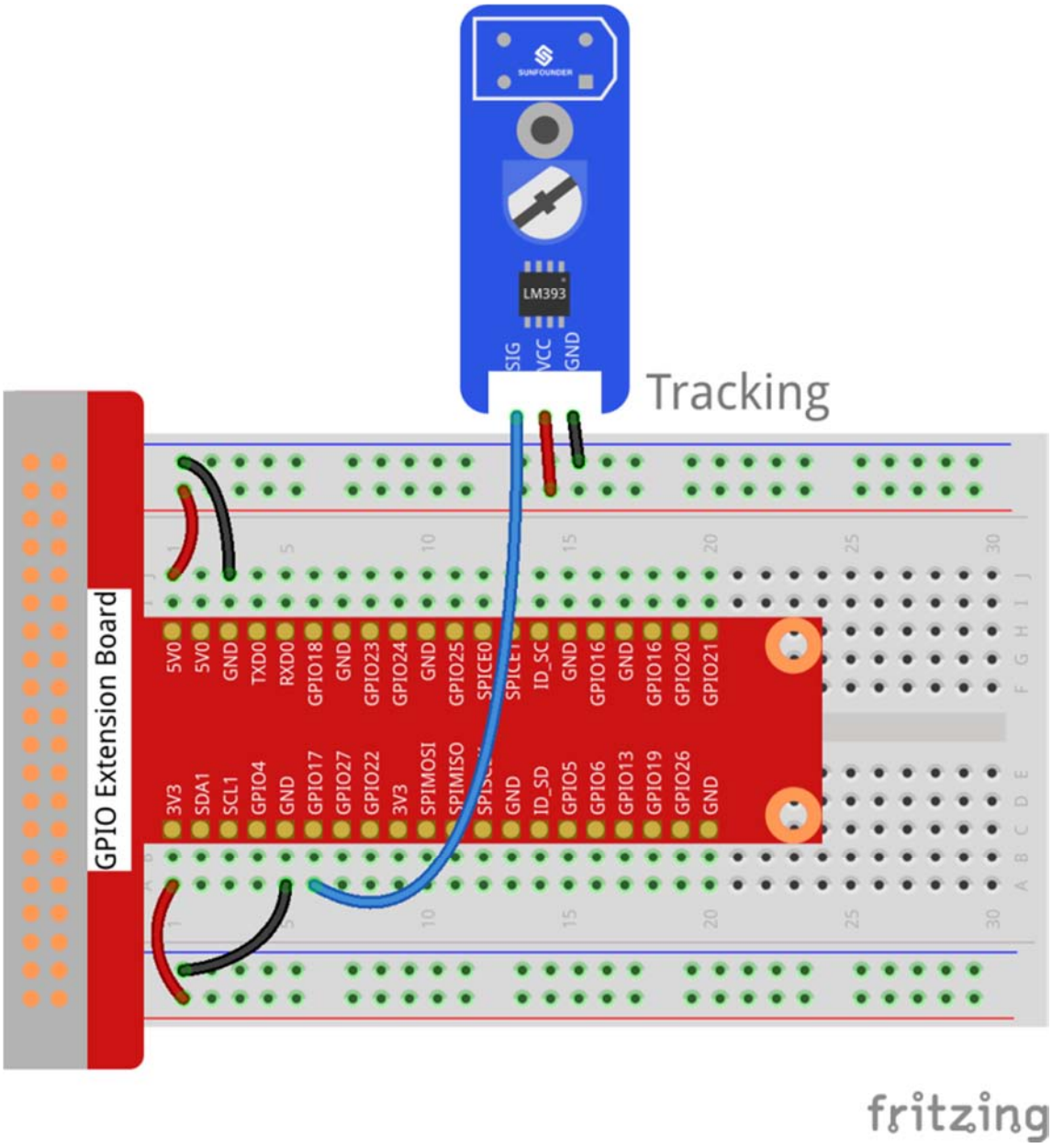
When the infrared transmitter emits rays to a piece of paper, if the rays shine on a white surface, they will be reflected and received by the receiver, and pin SIG will output low level; If the rays encounter black lines, they will be absorbed, thus the receiver gets nothing, and pin SIG will output high level. The schematic diagram of the module is as shown below:



Experimental Procedures

Step 1: Build the circuit

Raspberry Pi	T-Cobbler	Tracking Sensor Module
GPIO0	GPIO17	SIG
5V	5V0	VCC
GND	GND	GND



For C language users:

Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/34_tracking/
```

Step 3: Compile

```
gcc tracking.c -lwiringPi
```

Step 4: Run

```
sudo ./a.out
```

For Python users:

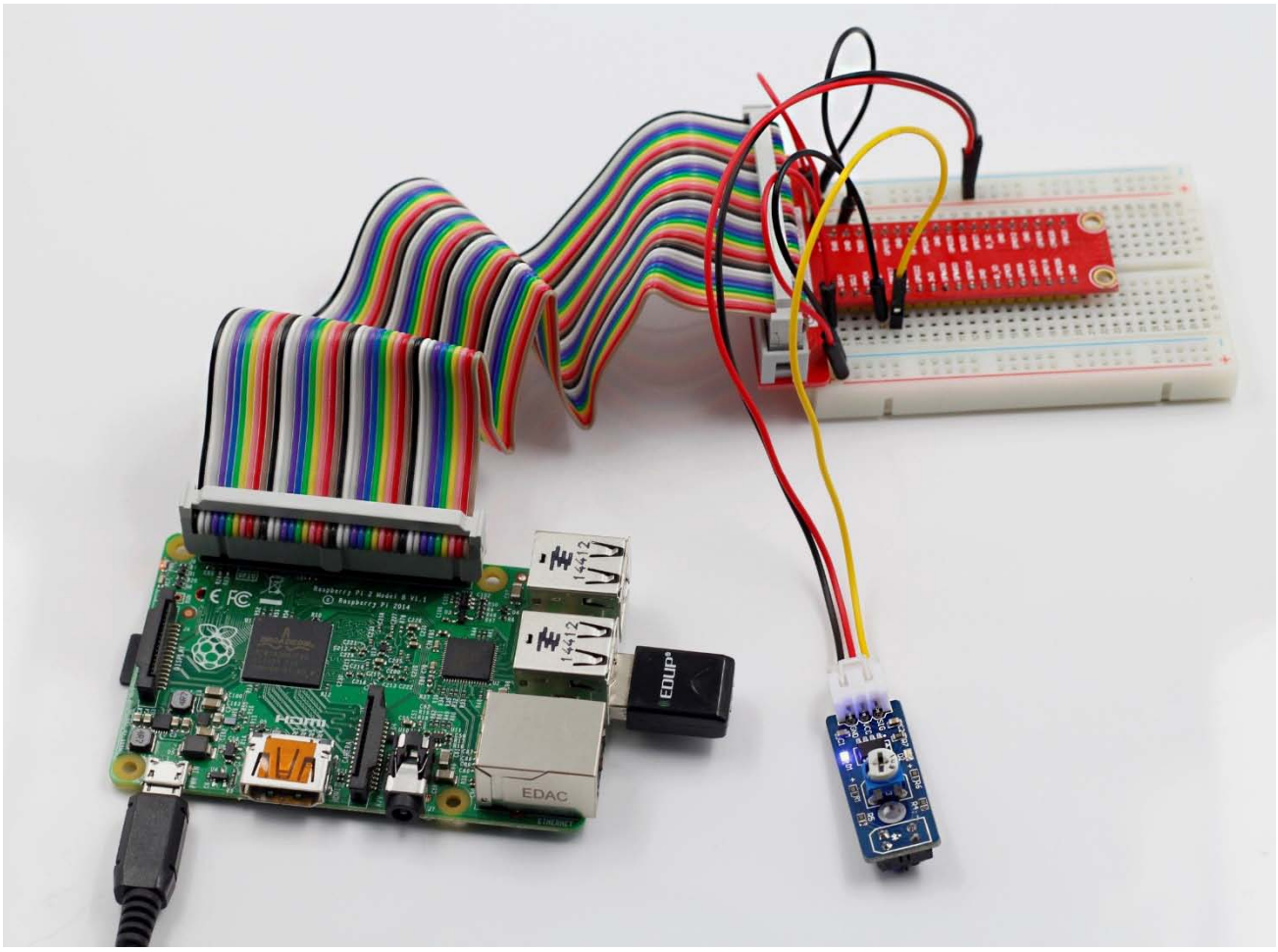
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 3: Run

```
sudo python 34_tracking.py
```

When the tracking sensor encounters black lines, a string "Black Line is detected" will be printed on the screen.



Lesson 35 Intelligent Temperature Measurement System

Introduction

In this experiment, we will use some modules together to build an intelligent temperature measurement system.

Components

- 1 * Raspberry Pi
- 1 * Breadboard
- 4 * Jumper wires (Male to Male, 2 red and 2 black)
- 1 * Network cable (or USB wireless network adapter)
- 1 * Active Buzzer
- 1 * RGB LED Module
- 1 * DS18B20 Temperature Sensor
- 1 * PCF8591
- 1 * Joystick PS2
- 2 * 3-Pin anti-reverse cable
- 1 * 4-Pin anti-reverse cable
- 1 * 5-Pin anti-reverse cable
- Several Jumper wires (Male to Female)

Experimental Principle

It is similar with lesson 26. The only difference is that we can adjust the lower limit and upper limit value by joystick PS2 when programming.

As mentioned previously, joystick PS2 has five operation directions: up, down, left, right and press-down. Well, in this experiment, we will use the left and right directions to control the upper limit value and up/down direction to control the lower limit. If you press down the joystick, the system will log out.

Experimental Procedures

Step 1: Build the circuit

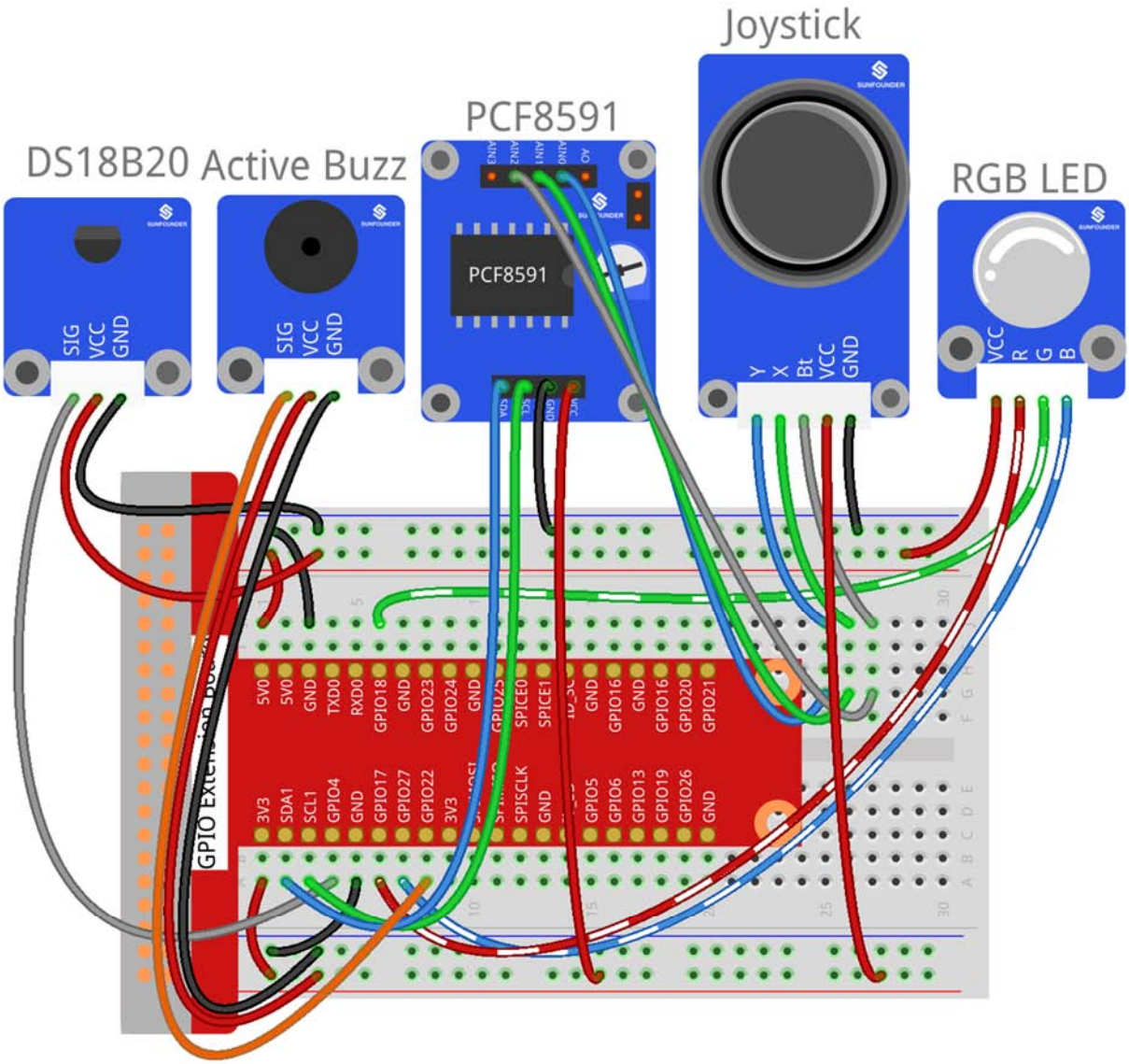
Raspberry Pi	T-Cobbler	DS18B20 Module
GPIO7	GPIO4	SIG
5V	5V0	VCC
GND	GND	GND

Raspberry Pi	T-Cobbler	PCF8591 Module
SDA	SDA1	SDA
SCL	SCL1	SCL
3V3	3V3	VCC
GND	GND	GND

Joystick PS2	T-Cobbler	PCF8591 Module
Y	*	AIN0
X	*	AIN1
Bt	*	AIN2
VCC	3V3	*
GND	GND	*

Raspberry Pi	T-Cobbler	RGB LED Module
GPIO0	GPIO17	R
GPIO1	GPIO18	G
GPIO2	GPIO27	B
5V	5V0	VCC

Raspberry Pi	T-Cobbler	Active Buzzer Module
GPIO3	GPIO22	SIG
3V	3V3	VCC
GND	GND	GND



For C language users:

Step 2: Check the address of your sensor

```
ls /sys/bus/w1/devices/
```

It may be like this:

```
28-031467805fff  w1_bus_master1
```

Copy or write down 28-XXXXXXX. It is the address of your sensor.

Step 2: Change directory and edit

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/C/35_expand02/
nano temp_monitor.c
```

Find the function *float tempRead(void)*, and the line "fd = open(XXXXXX)". Replace "28-031467805ff" with your sensor address.

```
float tempRead(void)
{
float temp;
int i, j;
int fd;
int ret;

char buf[BUFSIZE];
char tempBuf[5];

fd = open("/sys/bus/w1/devices/28-031467805fff/w1_slave", O_RDONLY);

if(-1 == fd){
perror("open device file error");
return 1;
}
```

Save and exit.

Step 4: Compile

```
gcc temp_monitor.c -lwiringPi
```

Step 5: Run

```
sudo ./a.out
```

For Python users:

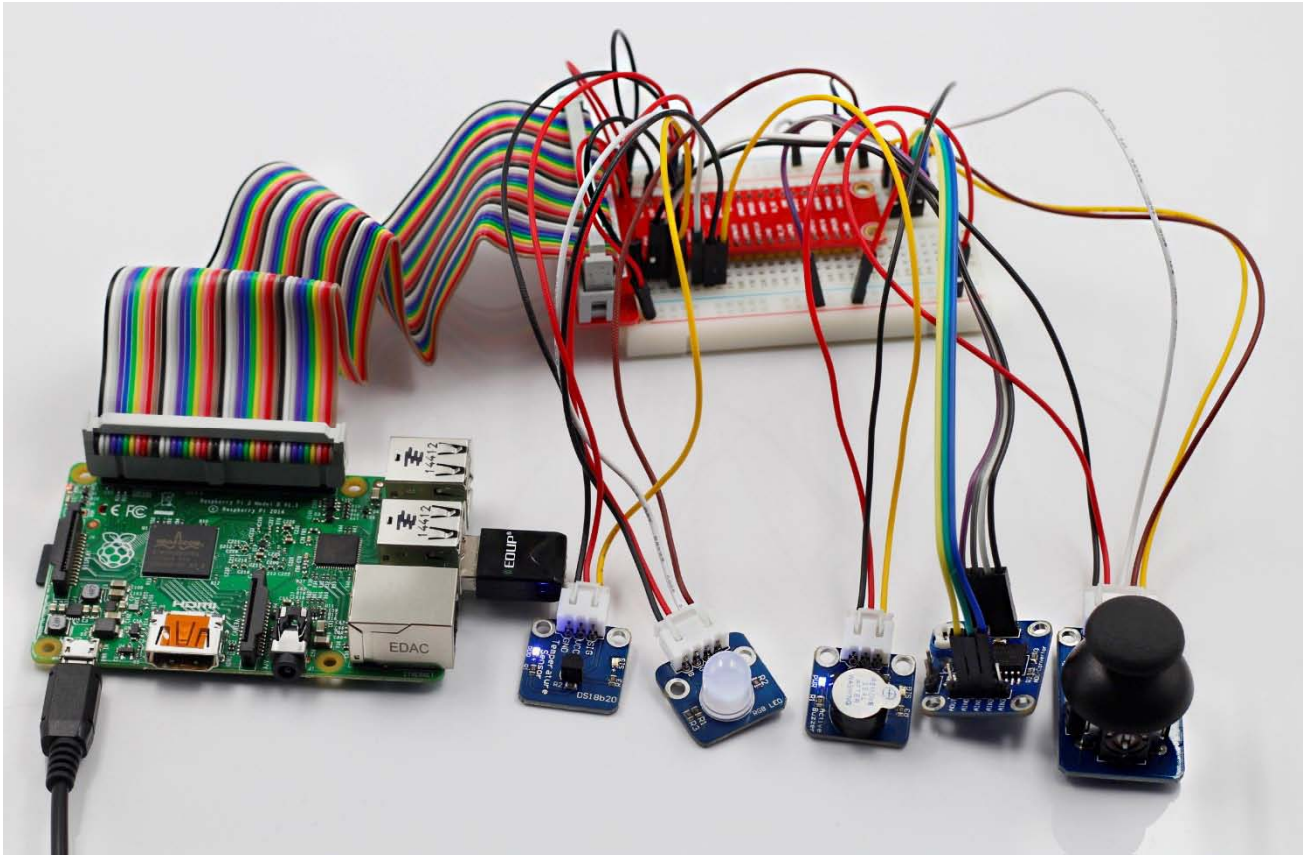
Step 2: Change directory

```
cd /home/pi/SunFounder_SensorKit_for_RPi2/Python/
```

Step 4: Run

```
sudo python 35_temp_monitor.py
```

Now, you can pull the shaft of the joystick left and right to set the upper limit value, and up and down to set the lower limit value. Then, if the ambient temperature reaches the upper limit value or lower limit value, the buzzer will beep in a different frequency to warn.

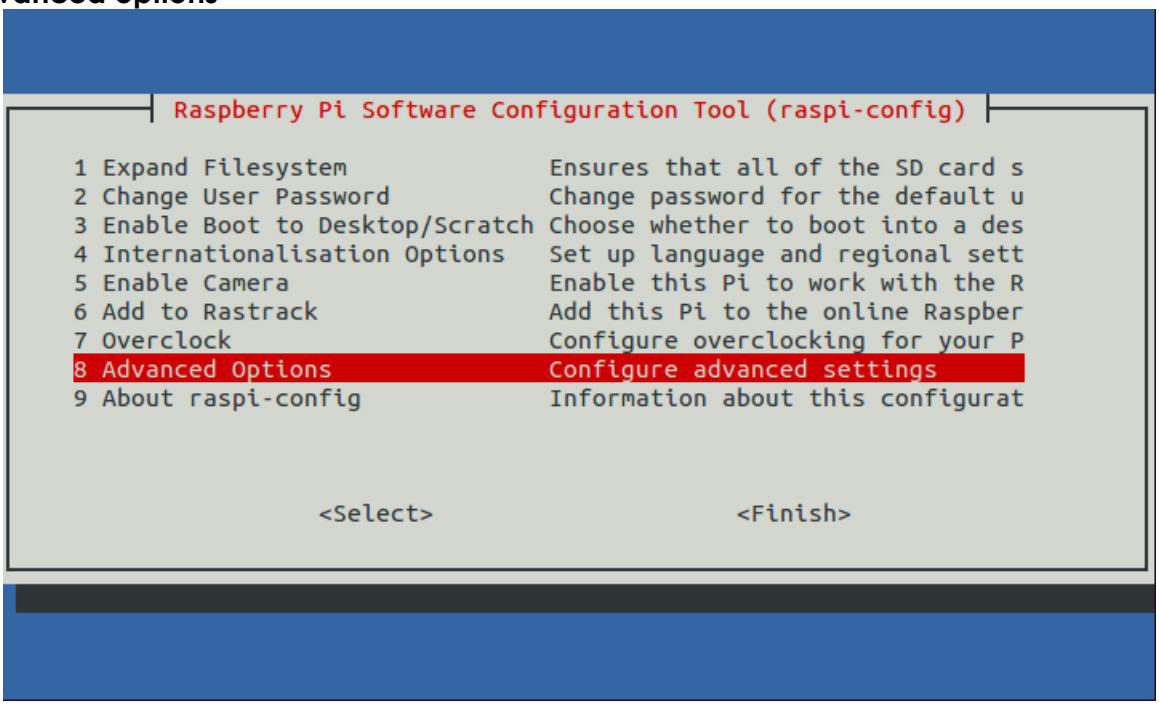


Appendix 1: I2C Configuration

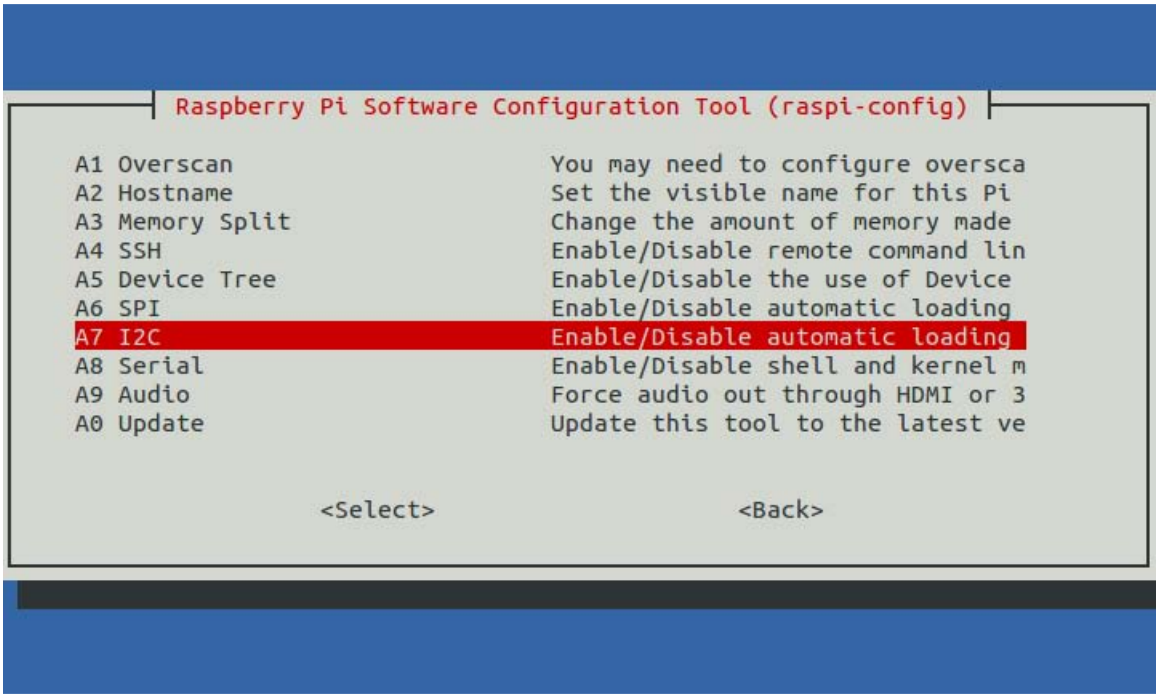
Step 1: Enable the I2C port of your Raspberry Pi (If you have enabled it, skip this; if you do not know whether you have done that or not, please continue):

```
sudo raspi-config
```

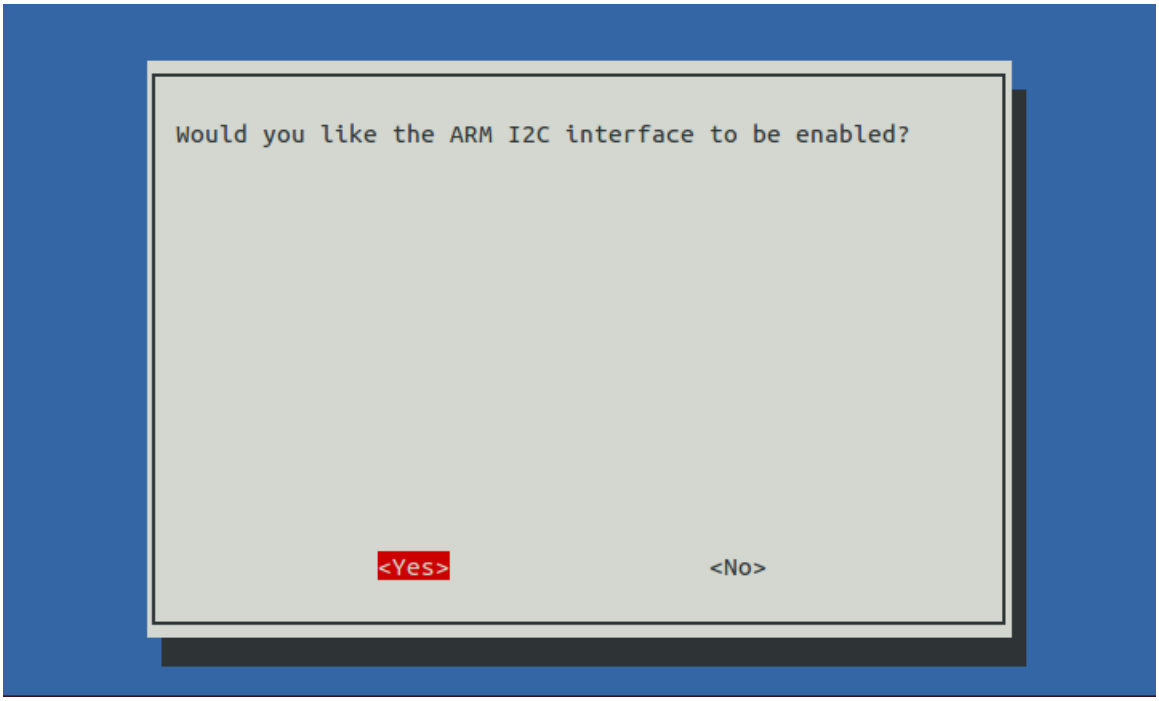
8 Advanced options



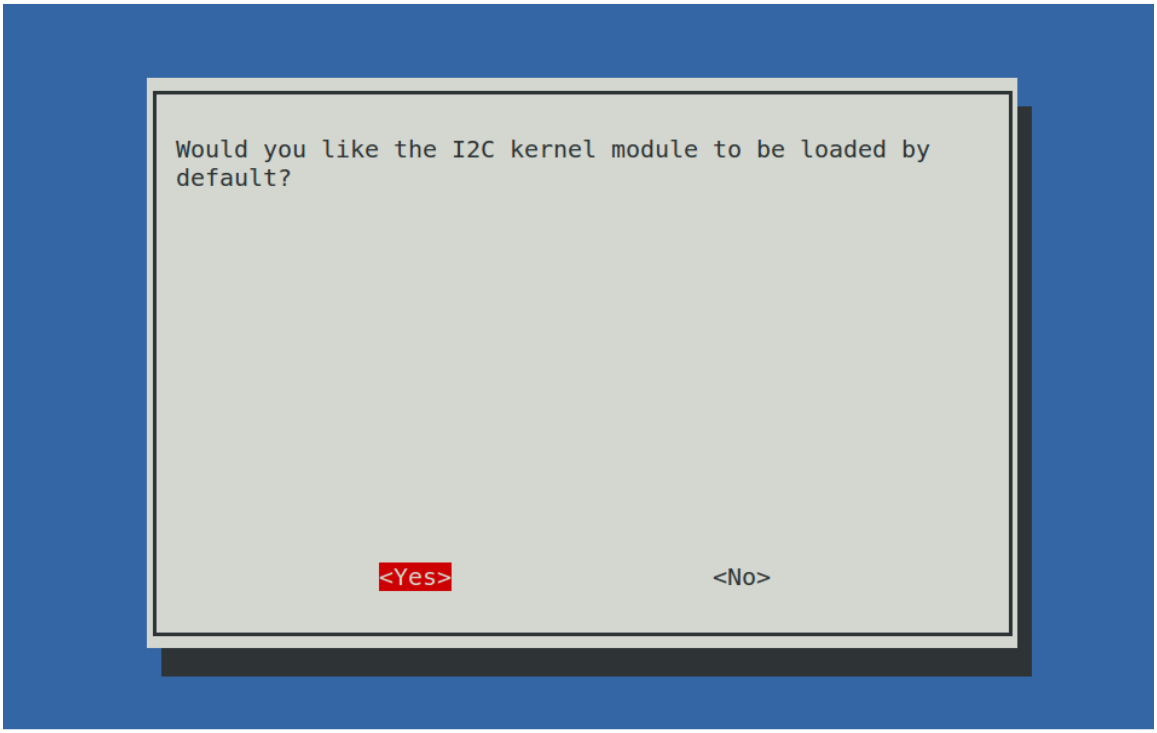
A7 I2C



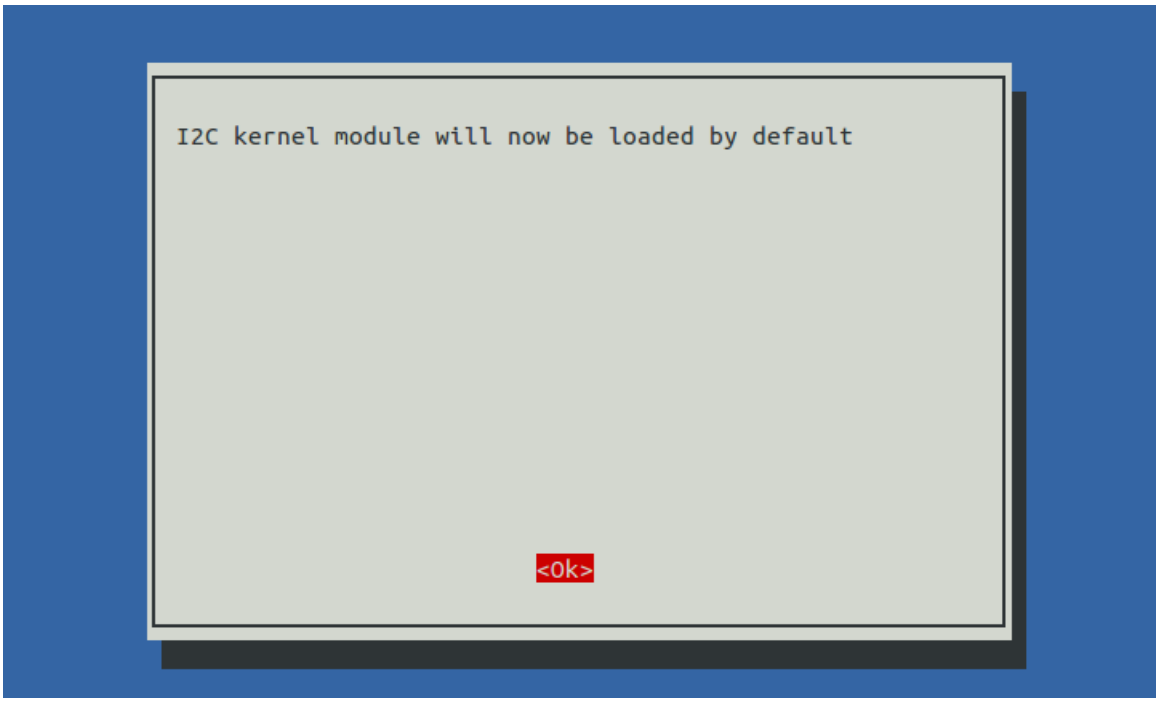
<Yes>


Would you like the ARM I2C interface to be enabled?
<Yes>
<No>

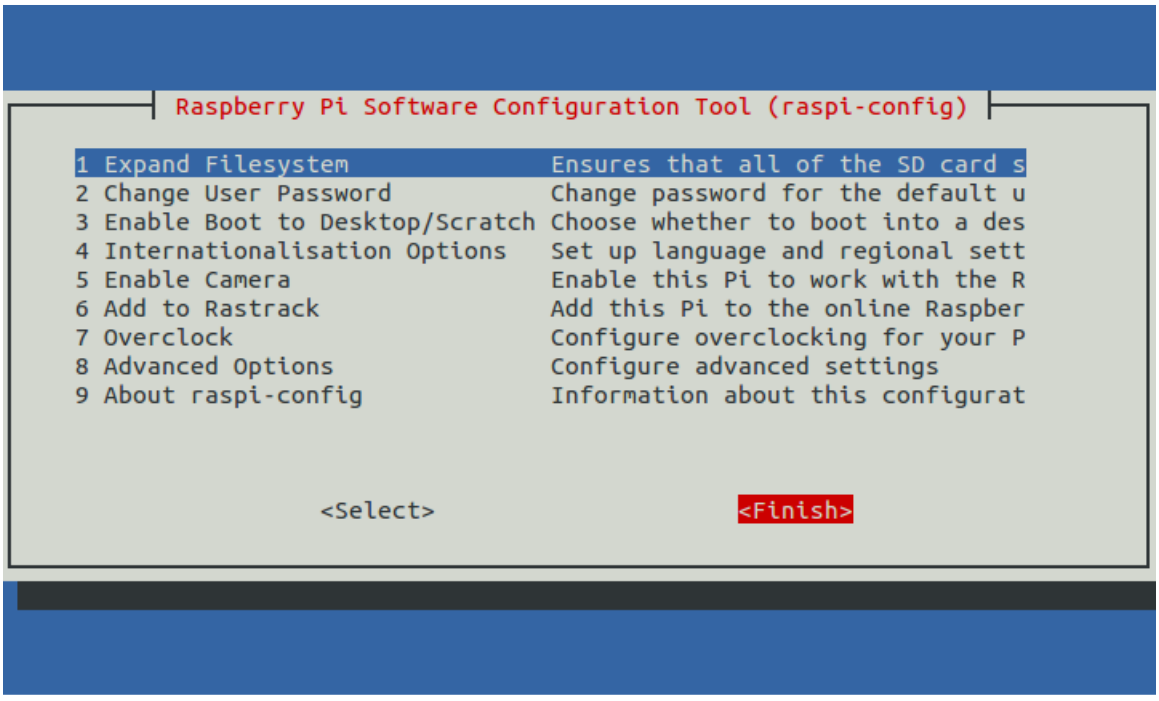
<Yes>


Would you like the I2C kernel module to be loaded by default?
<Yes>
<No>

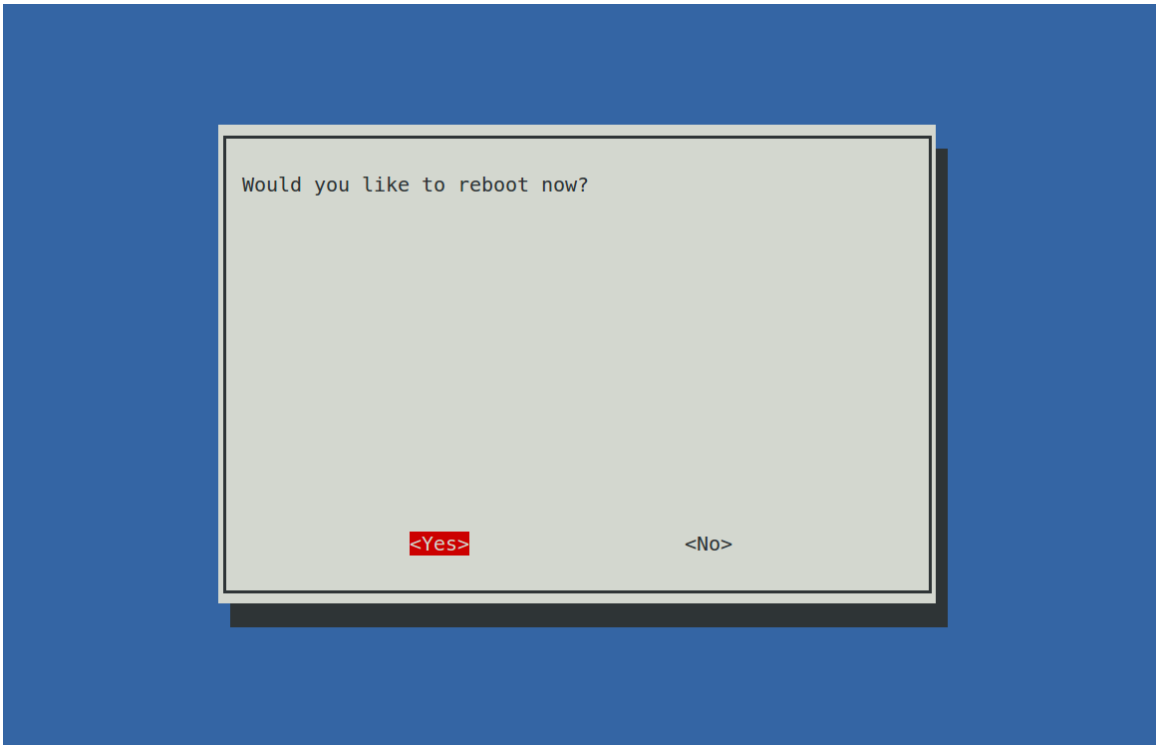
<Ok>


I2C kernel module will now be loaded by default
<Ok>

<Finish>


Raspberry Pi Software Configuration Tool (raspi-config)
1 Expand Filesystem Ensures that all of the SD card s
2 Change User Password Change password for the default u
3 Enable Boot to Desktop/Scratch Choose whether to boot into a des
4 Internationalisation Options Set up language and regional sett
5 Enable Camera Enable this Pi to work with the R
6 Add to Rastrack Add this Pi to the online Raspber
7 Overclock Configure overclocking for your P
8 Advanced Options Configure advanced settings
9 About raspi-config Information about this configurat
<Select>
<Finish>

<Yes> (If you do not see this page, continue to the next step)



Step 2: Check that the i2c modules are loaded and active:

```
lsmod | grep i2c
```

Then the following code will appear (the number may be different)

```
i2c_dev                 6276    0
i2c_bcm2708             4121    0
```

Step 3: Install i2c-tools

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

Step 4: Check the address of the I2C device:

```
i2cdetect -y 1      # For Raspberry Pi 2
i2cdetect -y 0      # For Raspberry Pi 1
pi@raspberrypi ~ $ i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- 48 -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

If there's an I2C device connected, the results will be similar as shown above – since the address of the device is 0x48, `48` is printed.

Step 5:

For C language users: Install libi2c-dev

```
sudo apt-get install libi2c-dev
```

For Python users: Install smbus for I2C

```
sudo apt-get install python-smbus
```

Copyright Notice