Introduction

Some electrical appliance often make a buzzing sound under electrical conditions. This is actually the sound of a buzzer and the annoying bell in schools is just a bigger buzzer. The buzzer is divided into active buzzer and passive buzzer. The distinction between an "active" buzzer and a "passive" buzzer is based on whether it has an internal oscillator rather than whether it needs to be powered. As long as you electrify it, the active buzzer will buzz but it has a fixed frequency. The passive buzzer has no internal oscillator, so the internal oscillator will not buzz when it is powered on, and it needs a square wave of 2~5 kHz to drive and then different frequency waveform will make the buzzer produce relevant sound.

Active buzzer Passive buzzer

Experimental purpose

Raspberry Pi can be used to create a lot of interactive work and its most common function is sound and light display. We have used LEDs in the previous experiment and now we will use a buzzer to play two sounds of different frequencies. As long as the frequency matches the score, we can hear beautiful music.

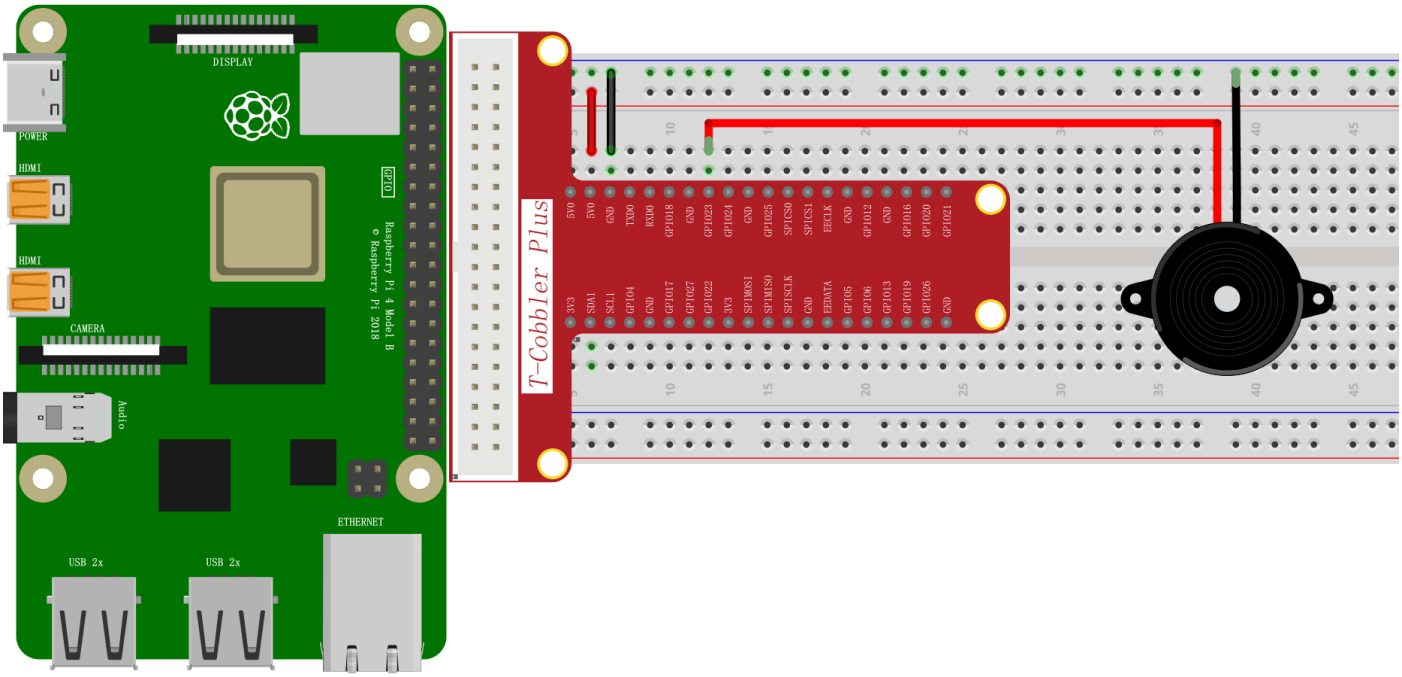
Component list

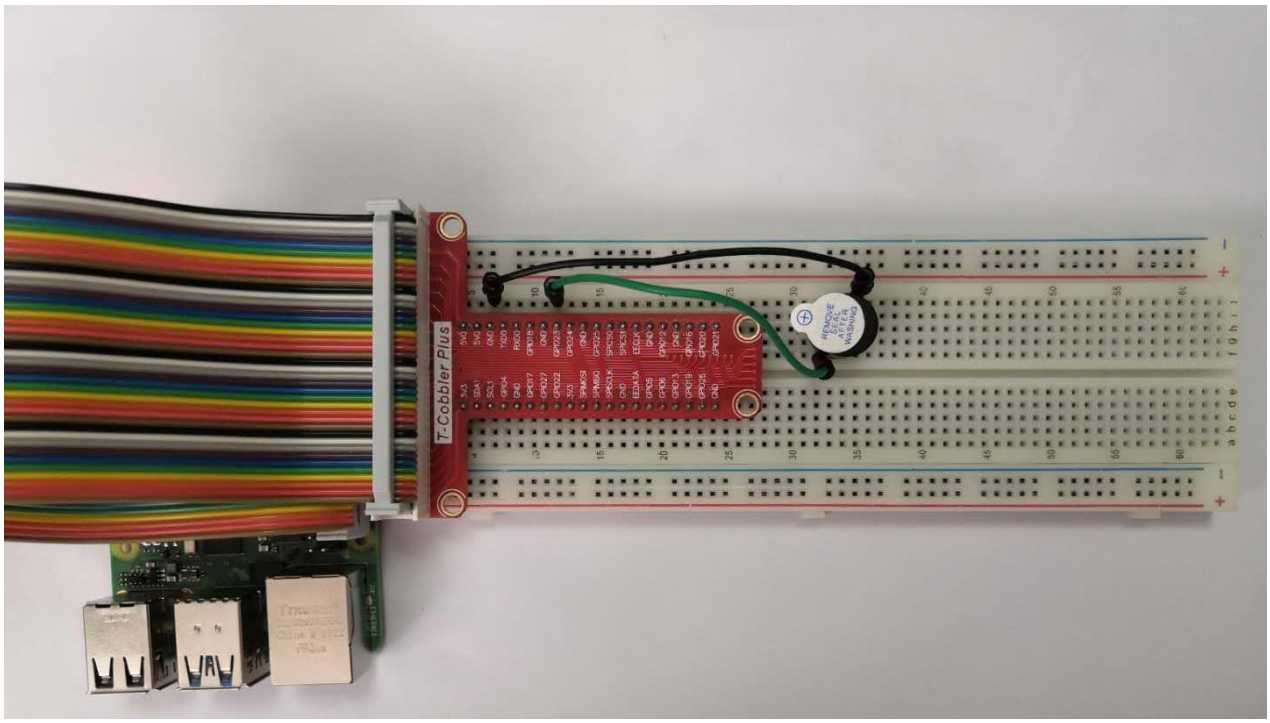
- Raspberry Pi main board
- T-Cobbler Plus expansion board
- Breadboard

- USB data cable
- Active buzzer *1，Passive buzzer * 1
- Several jumper wires

Wiring

Raspberry Pi	buzzer
IO4(wiringPi)/23(BCM)	+
GND	-





Note: The buzzer has both a cathode and an anode. In the above physical diagram, we can see that the buzzer has two colors of wiring and which are green and black. The circuit connection and programming of this experiment are very simple and the program is similar to the former. Since the control interface in the buzzer is also a digital interface, its output high-level voltage and low-level voltage can control the sound of the buzzer.

Experiment 1 Demo： Active buzzer C++ test program

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

int main ()
{
    int buzzer = 4;
    wiringPiSetup();
    pinMode(buzzer, OUTPUT);
    while(1)
    {
        digitalWrite(buzzer, HIGH);
        delay(500);
        digitalWrite(buzzer, LOW);
        delay(500);
    }
}
```

Experiment 1 Demo： Active buzzer Python test program

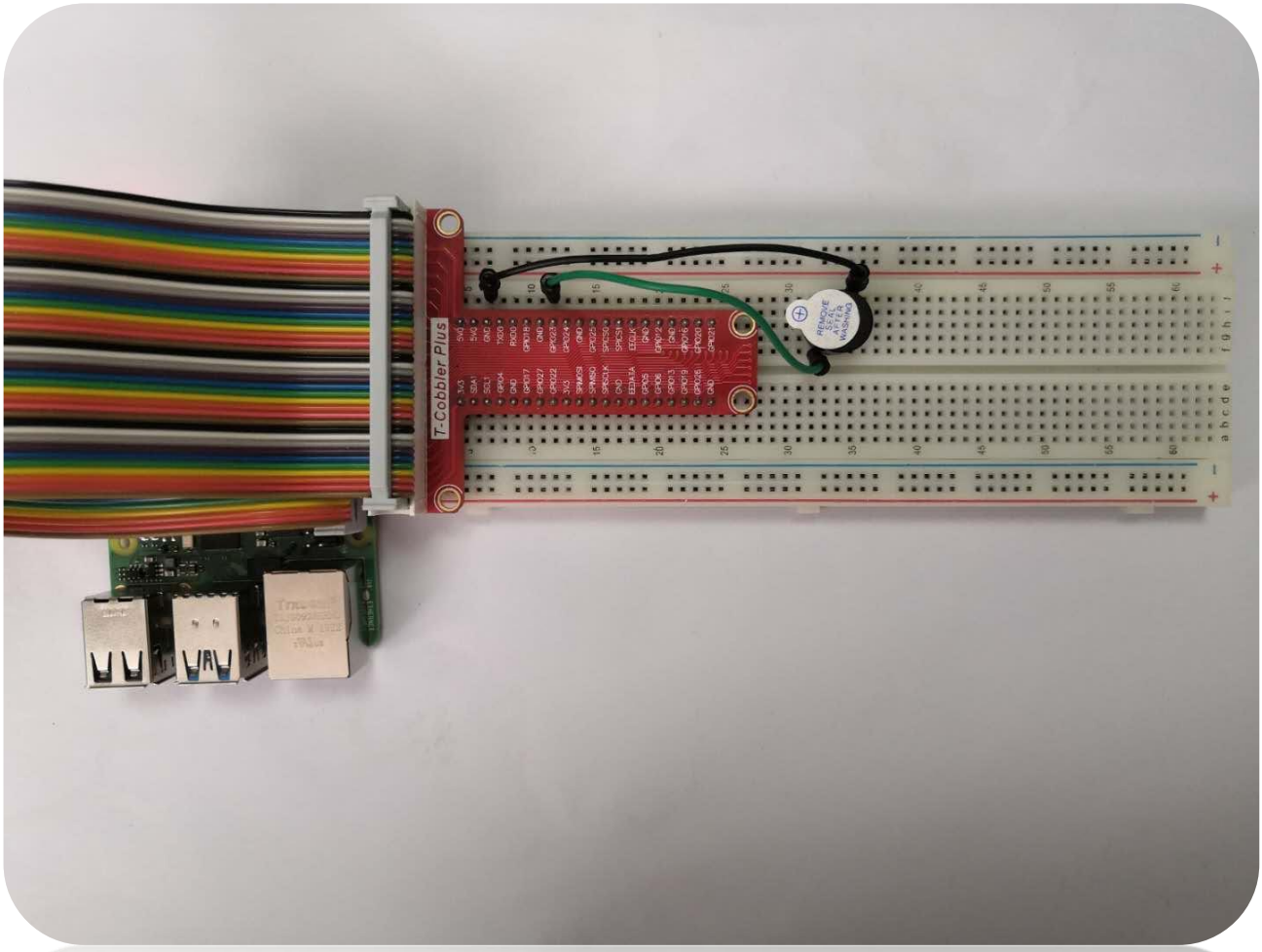
```
import time
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(23, GPIO.OUT)
p = GPIO.PWM(23, 100)
p.start(0)
try:
    while 1:
        for dc in range(0, 100):
            p.ChangeDutyCycle(dc)
            time.sleep(1)
        for dc in range(0, 100, 5):
            p.ChangeDutyCycle(dc)
            time.sleep(1)
except KeyboardInterrupt:
    pass
p.stop()
GPIO.cleanup()
```

Experiment 1 Demo： Active buzzer Java test program

```
import com.pi4j.wiringpi.Gpio;

public class ActiveBuzzer {
    static int buzzer_pin  = 4;

    static {
        // setup wiring pi
        if (Gpio.wiringPiSetup() == -1) {
            System.out.println(" ==>> GPIO SETUP FAILED");
        }

        Gpio.pinMode(buzzer_pin, Gpio.OUTPUT);
    }

    public static void main(String args[]) {

        for ( ; ;) {
            Gpio.digitalWrite(buzzer_pin, Gpio.HIGH);
            Gpio.delay(5);
            Gpio.digitalWrite(buzzer_pin,Gpio.LOW);
            Gpio.delay(5);
        }
    }
}
```

Experiment 1 active buzzer experimental effect:

In this experiment we used an active buzzer to make a beep sound.


T-Cobbler Plus

Experiment 2 Demo： Passive buzzer C++ test program

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

int main ()
{
    int buzzer=4;
    wiringPiSetup();
    pinMode(buzzer, OUTPUT);
    while(1)
    {
        for(int i=0;i<800;i++)  // 1k HZ
        {
            //声音频率设置
            digitalWrite(buzzer, HIGH);
            delay(0.5);
            digitalWrite(buzzer, LOW);
            delay(0.5);
        }
        delay(1000);
        for(int i=0;i<800;i++)   // 250 HZ
        {
            //声音频率设置
            digitalWrite(buzzer, HIGH);
            delay(2);
            digitalWrite(buzzer, LOW);
            delay(2);
        }
        delay(1000);
    }
}
```

Experiment 2 Demo： Passive buzzer Python test program

```
import time
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(23, GPIO.OUT)

p = GPIO.PWM(23, 100)
p.start(0)
try:
    while 1:
        for dc in range(0, 100):
            p.ChangeDutyCycle(dc)
            time.sleep(1)
        for dc in range(0, 100, 5):
            p.ChangeDutyCycle(dc)
            time.sleep(1)
except KeyboardInterrupt:
    pass
p.stop()
GPIO.cleanup()
```

Experiment 2 Demo：Passive buzzer Java test program

```
import com.pi4j.wiringpi.Gpio;

public class Buzzer {
    static int buzzer_pin  = 4;

    static {
        // setup wiring pi
        if (Gpio.wiringPiSetup() == -1) {
            System.out.println(" ==>> GPIO SETUP FAILED");
        }

        Gpio.pinMode(buzzer_pin, Gpio.OUTPUT);
    }

    public static void main(String args[]) {

        for ( ; ;) {
            //  500HZ
            for (int i = 0; i < 800; i++) {
                Gpio.digitalWrite(buzzer_pin, Gpio.HIGH);
                Gpio.delay(1);
                Gpio.digitalWrite(buzzer_pin,Gpio.LOW);
                Gpio.delay(1);
            }
            Gpio.delay(1000);

            //  250HZ
            for (int i = 0; i < 800; i++) {
                Gpio.digitalWrite(buzzer_pin, Gpio.HIGH);
                Gpio.delay(2);
                Gpio.digitalWrite(buzzer_pin,Gpio.LOW);
                Gpio.delay(2);
            }
            Gpio.delay(1000);
        }
    }
}
```

Experiment 2 passive buzzer experimental effect:

Our passive buzzer has no distinction between positive and negative poles. We can hear two beeps with different frequencies.