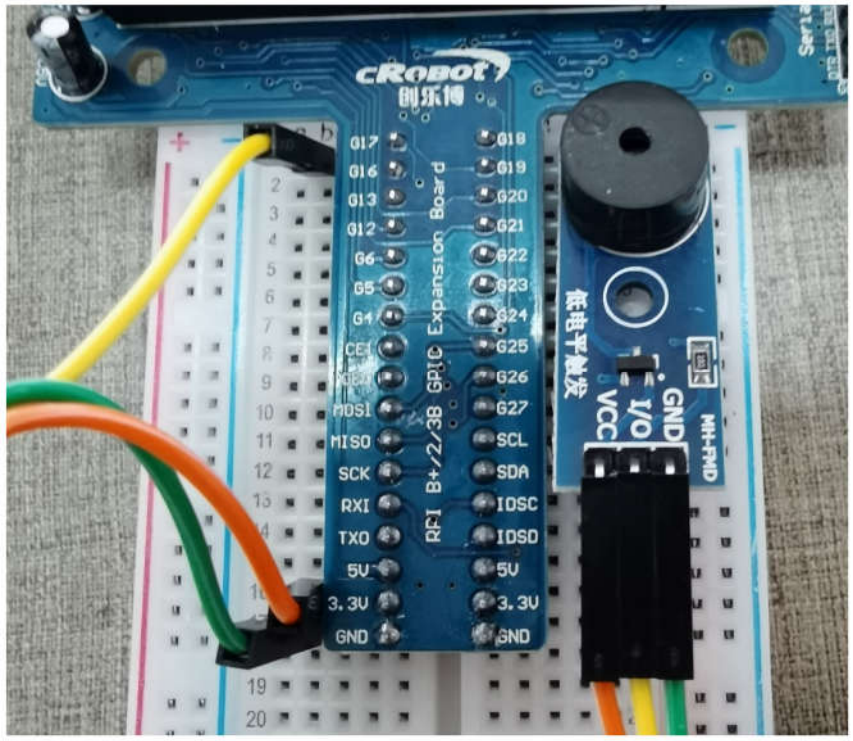
智能系统与控制

树莓派：PWM-蜂鸣器奏乐

(Pulse-width modulation,脉冲宽度调制)



于泓
鲁东大学
信息与电气工程学院

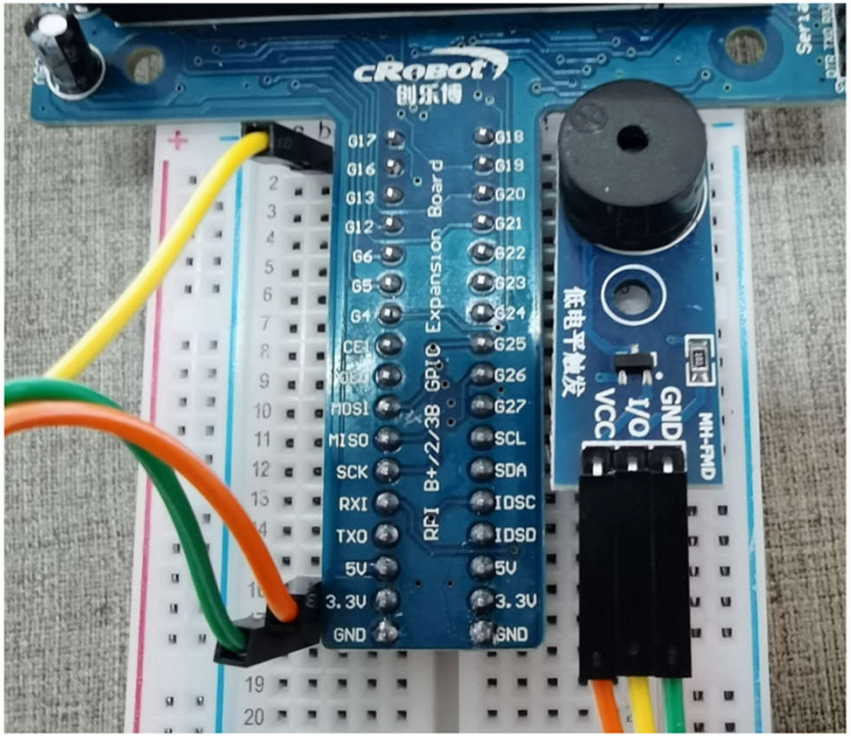
2021.10.19

无源蜂鸣器

- 蜂鸣器是一种简单低廉的音频信号装置，可以分为有源和无源两种。
- **有源蜂鸣器**内置震荡源通电时会发出单一频率的声音。
- **无缘蜂鸣器**内部没有震荡源所以接入直流电后不会发出声音，需要接入一定频率的方波来进行驱动。因此通过控制输入无源蜂鸣器内方波的频率，可以控制其发出不同频率的声响。
- 在本节中我们将介绍如何利用树莓派的 GPIO 的 PWM功能，输出不同频率的方波，控制无源蜂鸣器演奏一段简单的音乐

蜂鸣器的 VCC 与 GND 分别与扩展板的 3.3v 和 GND 相连接。

蜂鸣器的信号线 I/O 与树莓派的 **GPIO17** 相连接。



```python
#-*- coding: utf-8 -*
import RPi.GPIO as GPIO
import time
from pin_dic import pin_dic


class Buzzer_Song(object):
    def __init__(self,pin_buzzer,delay_beat=0.5):

        # 设置蜂鸣器引脚模式
        self.pin_buzzer = pin_buzzer
        GPIO.setup(self.pin_buzzer,GPIO.OUT)

        # 创建PWM对象初始频率440占空比50%
        self.Buzzer = GPIO.PWM( pin_buzzer , 440)
        self.Buzzer.start(50)

        self.note2freq = {"cl1":131,"cl2":147 ,'cl3':165 ,"cl4":175 ,"cl5":196 ,"cl6":211 ,"cl7":248,
                          "cm1":262,"cm2":294 ,'cm3':330 ,"cm4":350 ,"cm5":393 ,"cm6":441 ,"cm7":495,
                          "ch1":525,"ch2":589 ,'ch3':661 ,"ch4":700 ,"ch5":786 ,"ch6":882 ,"ch7":990
                          }
        self.delay_beat = delay_beat

    def play_song(self,notes,beats):

        for note,beat in zip(notes,beats):
            self.Buzzer.ChangeFrequency(self.note2freq[note])
            time.sleep(self.delay_beat*beat)

    def destory(self):
        self.Buzzer.stop()
        GPIO.output(self.pin_buzzer, GPIO.LOW)
        GPIO.cleanup()
```

IO 引脚

节拍时长控制

音符到频率的转换字典

切换频率，演奏音乐

```
if __name__ == "__main__":

    # 设置引脚编号模式
    GPIO.setmode(GPIO.BOARD)

    # 定义buzzer引脚
    pin_buzzer = pin_dic ['G17']
    m_buzzer_song = Buzzer_Song(pin_buzzer)

    notes = ['cm1' ,'cm1' , 'cm1' , 'cl5' , 'cm3' , 'cm3' , 'cm3' , 'cm1' ,
             'cm1' , 'cm3' , 'cm5' , 'cm5' , 'cm4' , 'cm3' , 'cm2' , 'cm2' ,
             'cm3' , 'cm4' , 'cm4' , 'cm3' , 'cm2' , 'cm3' , 'cm1' , 'cm1' ,
             'cm3' , 'cm2' , 'cl5' , 'cl7', 'cm2' , 'cm1']
    beats = [1 , 1 , 2 , 2 , 1 , 1 , 2 , 2 ,
             1 , 1 , 2 , 2 , 1 , 1 , 3 , 1 ,
             1 , 2 , 2 , 1 , 1 , 2 , 2 , 1 ,
             1 , 2 , 2 , 1 , 1 , 3]


    # 循环演奏音乐
    try:
        while True:
            m_buzzer_song.play_song(notes,beats)
    except KeyboardInterrupt:
        print('\n Ctrl + C QUIT')
    finally:
        m_buzzer_song.destory()
```

音符

节拍