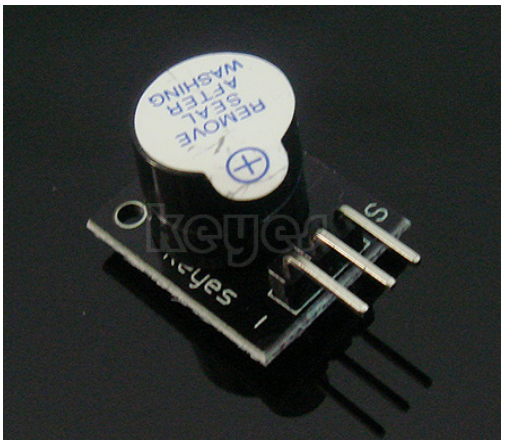
蜂鸣器模块



一、相关介绍

相信大家对蜂鸣器都不会陌生，我们在很多方案中都会用到蜂鸣器，大部分都是使用蜂鸣器来做提示或报警，比如按键按下、开始工作、工作结束或是故障等等。这里对单片机在蜂鸣器驱动上的应用作一下描述。

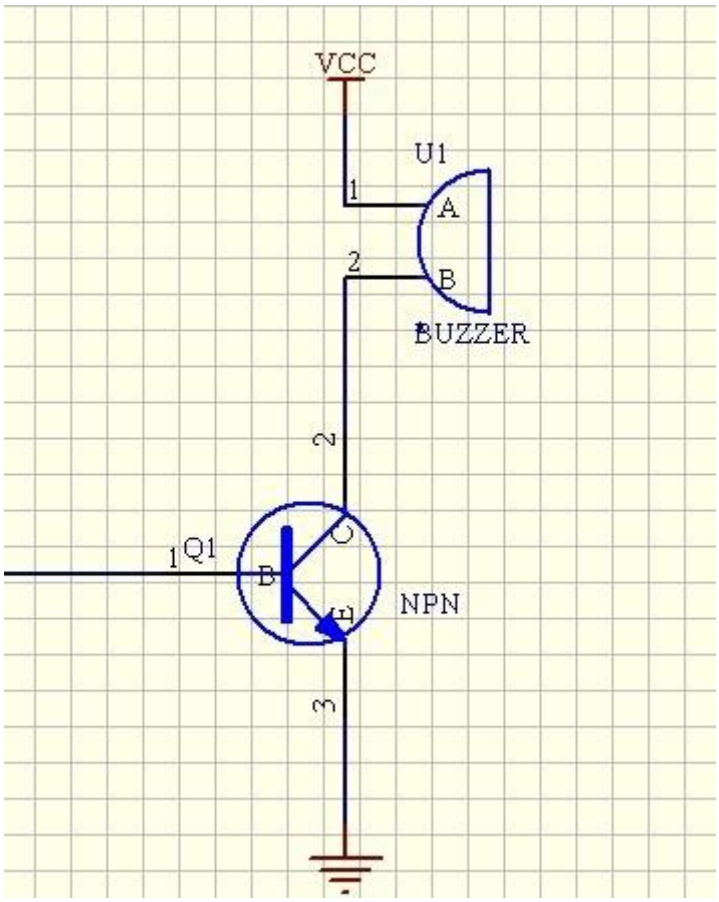
二、驱动方式

惯用驱动蜂鸣器的方式有两种：一种是 PWM 输出口直接驱动，另一种是利用 I/O 定时翻转电平产生驱动波形对蜂鸣器进行驱动。

PWM 输出口直接驱动是利用 PWM 输出口本身可以输出一定的方波来直接驱动蜂鸣器。在单片机的软件设置中有几个系统寄存器是用来设置 PWM 口的输出的，可以设置占空比、周期等等，通过设置这些寄存器产生符合蜂鸣器要求的频率的波形之后，只要打开 PWM 输出，PWM 输出口就能输出该频率的方波，这个时候利用这个波形就可以驱动蜂鸣器了。比如频率为 2000Hz

的蜂鸣器的驱动，可以知道周期为 500μs，这样只需要把 PWM 的周期设置为 500μs，占空比电平设置为 250μs，就能产生一个频率为 2000Hz 的方波，通过这个方波再利用三极管就可以去驱动这个蜂鸣器了。

而利用 I/O 定时翻转电平来产生驱动波形的方式会比较麻烦一点，必须利用定时器来做定时，通过定时翻转电平产生符合蜂鸣器要求的频率的波形，这个波形就可以用来驱动蜂鸣器了。比如为 2500Hz 的蜂鸣器的驱动，可以知道周期为 400μs，这样只需要驱动蜂鸣器的 I/O 口每 200μs 翻转一次电平就可以产生一个频率为 2500Hz，占空比为 1/2duty 的方波，再通过三极管放大就可以驱动这个蜂鸣器了。



三、模块使用

大家看看到模块后应该明白，其实他非常方便使用，一个电源端，一个地端，还有一个就是信号输入端。我们只要把电源、地线接好，那信号线接上 IO 口就行

四、模块功能测试

硬件要求

Arduino 控制器 × 1

USB 数据线 × 1

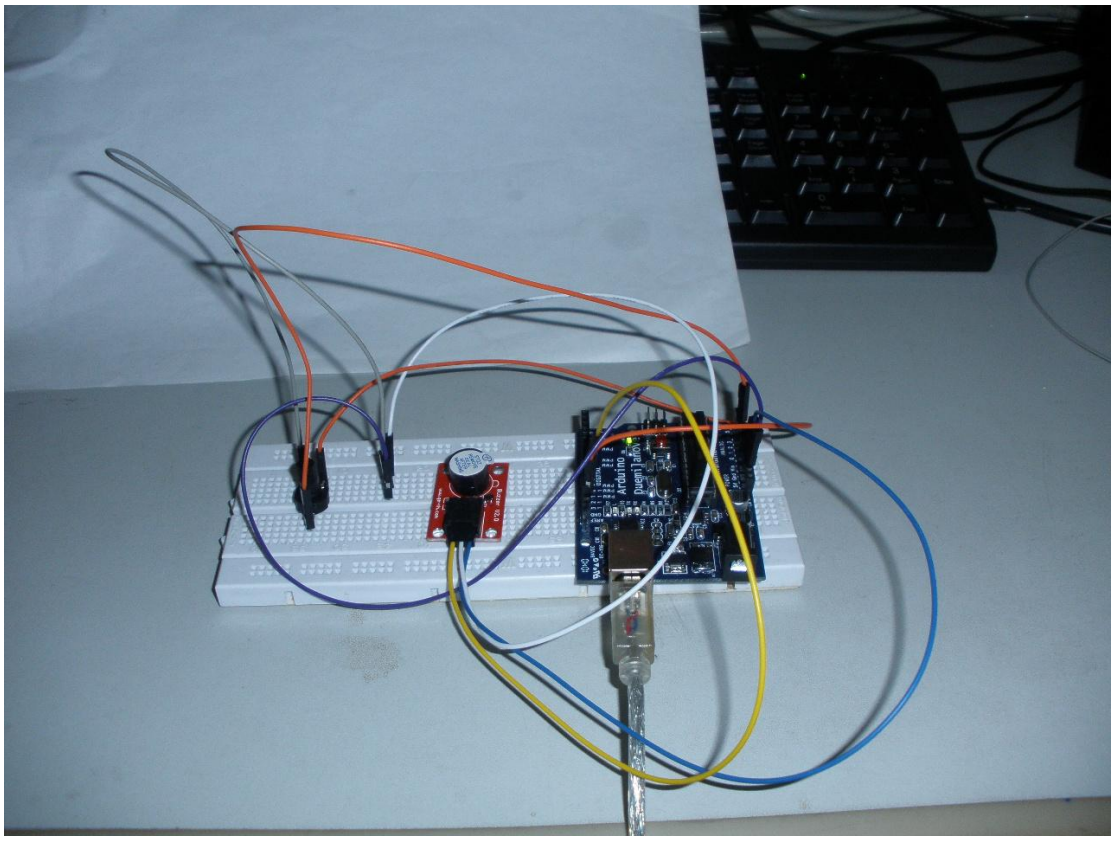
蜂鸣器模块 × 1

可调电位器（10K） × 1

我们下面的测试例子主要是学会如何控制蜂鸣器的发声，和一些简单的应用，当然还有

用两种不同的驱动方式驱动蜂鸣器的发声，大家可以比较下效果，一方便以后的使用。

具体的接法下面有



下面的这个测试代码是一个关于使用模拟量控制蜂鸣器显示频率的程序：

程序说明：第 10 引脚为控制蜂鸣器的引脚。

第 3 引脚为模拟量引脚，所使用的可调电阻为 10K。

功能：调动可调电阻器可以听到蜂鸣器响明显的频率变化。

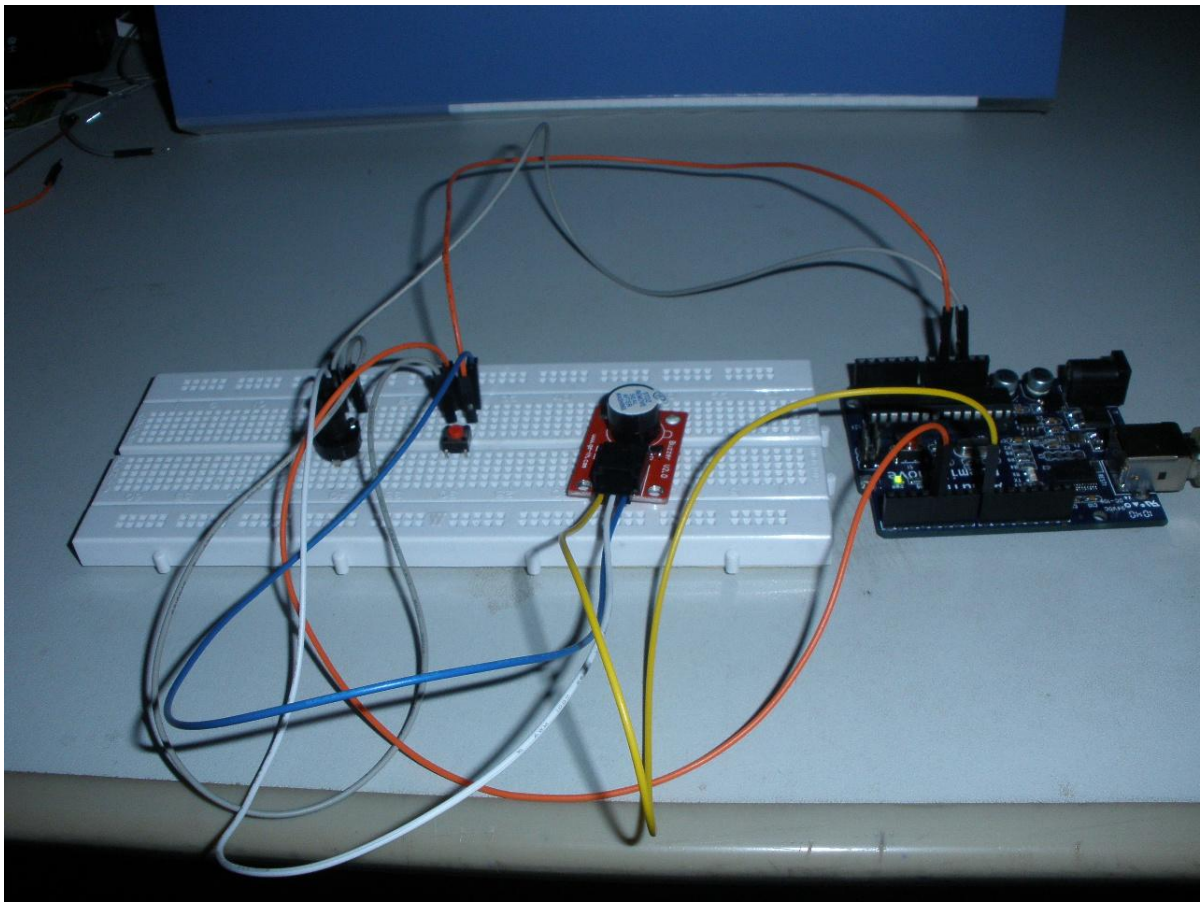
```
int speakerPin = 8;//控制喇叭的引脚
int potPin = 4;//控制可调电阻器的引脚
int value = 0;
void setup() {
   pinMode(speakerPin, OUTPUT);
}
void loop() {
```

```
  value = analogRead(potPin);读电阻器引脚的值
  digitalWrite(speakerPin, HIGH);
  delay(value);调节喇叭响的时间;
  digitalWrite(speakerPin, LOW);
  delay(value);调节喇叭不响的时间;
}
```

在这里我们可以认为的调节电位器来达到延时不同时间的效果，从而改变蜂鸣器的发声频率，大家可以试下，看到底是不是那样的，嗯~~~~

下面我们加了一个按键开关控制蜂鸣器，这样我们就能模拟一个简单的门铃，当按键被按下时喇叭就可以发出响声了。

实物连接如下：



示例代码：

```
const int buttonPin = 4;          //  按键引脚;
const int speakerPin =    8;           //蜂鸣器引脚;
// variables will change:
int buttonState = 0;                 //  读取按键引脚的一个值
void setup()
{
   //设置按键引脚为输入模式，蜂鸣器引脚为输出模式；
   pinMode(speakerPin, OUTPUT);
   pinMode(buttonPin, INPUT);
}
void loop(){
   //  读取按键一个初值，这里在电路中我接了是在默认高电平，所以初值为高；
   buttonState = digitalRead(buttonPin);
   /*  如果按键为高，则蜂鸣器不响；
  因为我刚开始接在硬件电路中初值为高，所以 if 条件成立，蜂鸣器不响
  */
   if (buttonState == HIGH) {
      digitalWrite(speakerPin,LOW);
```

```
  }
  else {
   //这里按键的值为低电平（也是按键被按下时），蜂鸣器响起
    digitalWrite(speakerPin,HIGH);
  }
}
```

这个程序比较简单，相信大家一看就明白了，为了增加大家对蜂蜜器的认识我为大家写了一小段用 PWM 控制蜂鸣器的代码。

下面这个程序是利用 PWM（脉冲宽度调节）控制蜂鸣器的，下载到单片机可以听到蜂鸣器发出不同的音调，我们只要根据相关曲目调节出音符(0,1,2,3,4,5,6,7)就可以让蜂鸣器唱歌了。

程序如下：

```
int speakerPin = 8;
byte song_table[] = {  30,  30,  30,  40,  50,  60,  70,  80,  90, 100,110, 120, 130, 140, 150, 160, 170, 180, 190, 200, 210, 220, 230, 240, 250, 250, 240, 230, 220, 210, 200, 190, 180, 170, 160, 150, 140, 130, 120, 110, 100,  90,  80,  70,  60,  50,  40,  30,  30,  30 };
int MAX = 50;
int count = 0;
void setup() {
```

```
    pinMode(speakerPin, OUTPUT);
}
void loop() {
    analogWrite(speakerPin,song_table[count]);
    count ++;
    if (count > MAX) {
        count = 0;
    }
    delay(50);
}
```

具体的实物连接可用上例的。

五、结束语

由于蜂鸣器的控制相对比较简单，我们也不做过多的介绍了，大家会用就行，当然我们上面的测试实例相对粗略，蜂鸣器的发声效果可能不是很好，有读者慢慢领会。。。。。