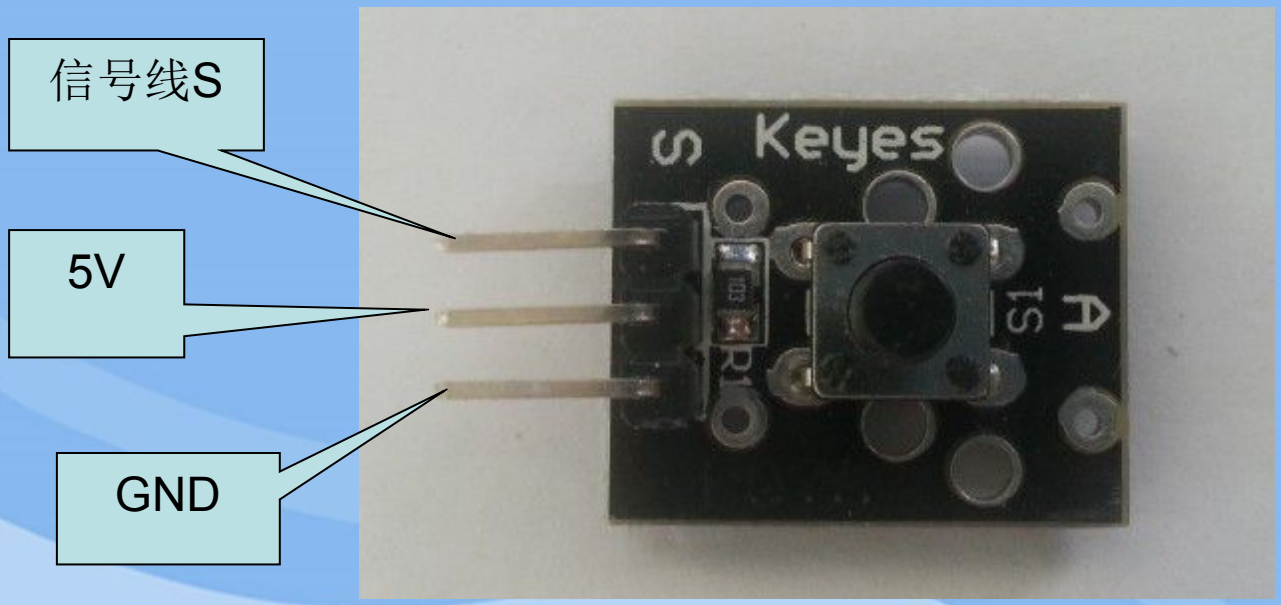

Arduino传感器认识
与应用

1、按钮开关

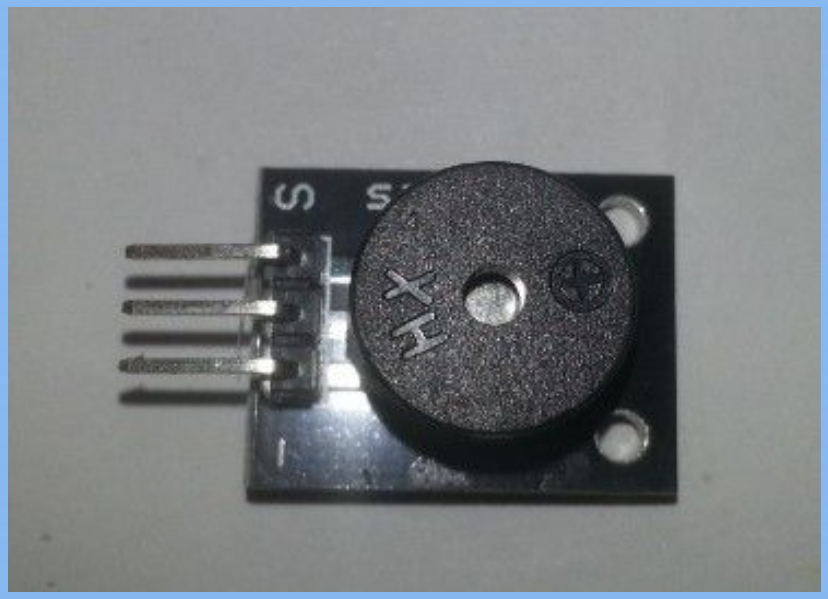


开关例程

- 按键开关模块和数字13 接口自带LED 搭建简单电路，制作按键提示灯
- 利用数字13 接口自带的LED，将按键开关传感器接入数字3接口，当按键开关传感器感测到有按键信号时，LED 亮,反之则灭.

```
int Led=13;   //定义LED 接口
int buttonpin=3;       //定义按键开关传感器接口
int val;               //定义数字变量val
void setup()
{
     pinMode(Led,OUTPUT);//定义LED 为输出接口
    pinMode(buttonpin,INPUT);//定义按键开关传感器为输入接口
}
void loop()
{
    val=digitalRead(buttonpin);//将数字接口3的值读取赋给val
   if(val==HIGH)//当按键开关传感器检测有信号时，LED 闪烁
      {
       digitalWrite(Led,HIGH)
      }
  else
    {
     digitalWrite(Led,LOW)
     }
}
```

2、无源蜂鸣器



■蜂鸣器的作用 蜂鸣器是一种一体化结构的电子讯响器，采用直流电压供电，广泛应用于计算机、打印机、复印机、报警器、电子玩具、汽车电子设备、电话机、定时器等电子产品中作发声器件。

■这里的“源”不是指电源。而是指震荡源。 也就是说，有源蜂鸣器内部带震荡源，所以只要一通电就会叫。而无源内部不带震荡源，所以如果用直流信号无法令其鸣叫。必须用2K~5K的方波去驱动它。

实验原理

- 使用数字I/O口，通过高低电平的变换，来实现方波。第一个例子是持续1毫秒的低电平，和1毫秒的高电平，实现一个500HZ的方波信号，持续80个方波，然后切换为持续2毫秒的低电平，和2毫秒的高电平，实现一个250HZ的方波信号.持续100个方波。两种声音交替发出。

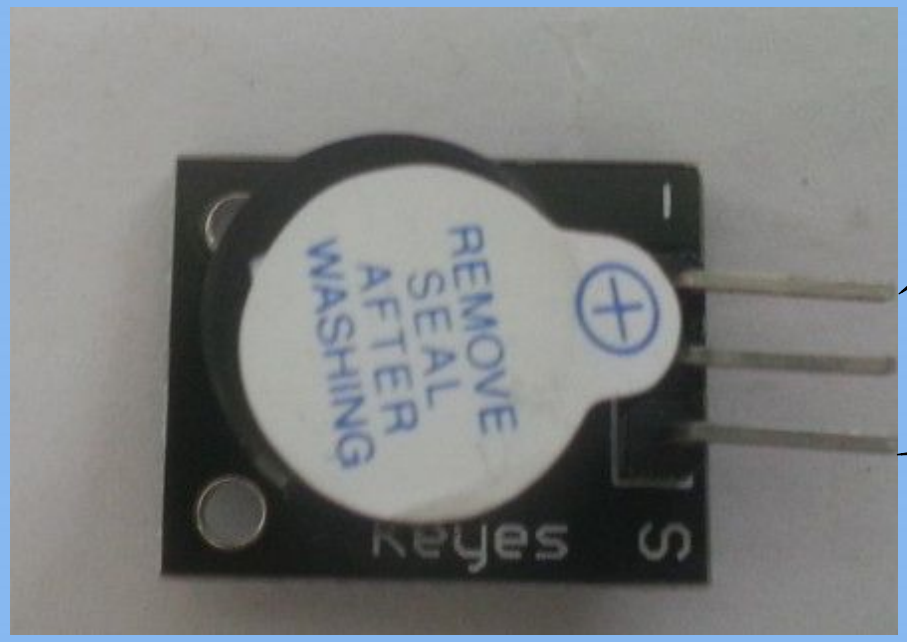
```
int buzzer=8;//设置控制蜂鸣器的数字IO脚
void setup()
{     pinMode(buzzer,OUTPUT);//设置数字IO脚模式，OUTPUT为输出
}
void loop()
{   unsigned char i,j;//定义变量
    for(i=0;i<80;i++)//辐出一个频率的声音
     {
        digitalWrite(buzzer,HIGH);  //发声音
        delay(1);     //延时1ms
        digitalWrite(buzzer,LOW);   //不发声音
        delay(1);  //延时ms
     }
for(i=0;i<100;i++)     //输出出另一个频率的声音
  {     digitalWrite(buzzer,HIGH);   //发声音
        delay(2);//延时2ms
        digitalWrite(buzzer,LOW);    //不发声音
    delay(2);//延时2ms
  }
}
```

```
int buzzer=8;//设置控制蜂鸣器的数字IO脚
void setup()
{ pinMode(buzzer,OUTPUT);//设置数字IO脚模式，OUTPUT为辐出
}
void loop()
{  unsigned char i,j;    //定义变量
    for(i=0;i<100;i++)   //输出一个频率的声音
    { digitalWrite(buzzer,HIGH);    //发声音
      delayMicroseconds(40);     // 延时40微秒
     digitalWrite(buzzer,LOW);//不发声音
     delayMicroseconds(40);   //延时40微秒
    }
   for(i=0;i<250;i++)//辐出另一个频率乩声音
   {  digitalWrite(buzzer,HIGH);//发声音
     delayMicroseconds(120);//延时120微秒

     digitalWrite(buzzer,LOW);//不发声音
     delayMicroseconds(120);//延时120微秒
    }

}
```

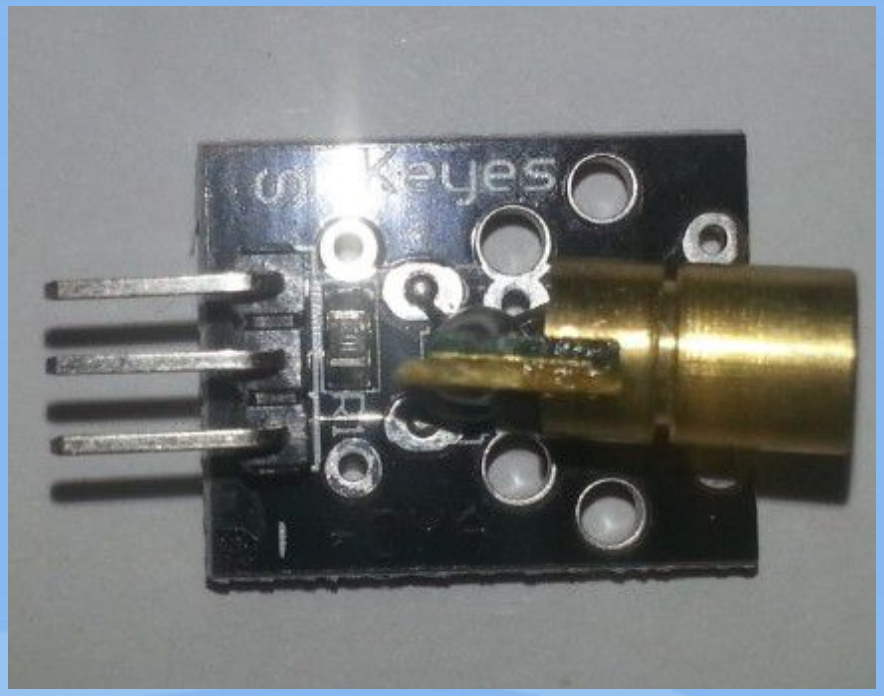
3、有源蜂鸣器



- 有源蜂鸣器内部带震荡源，所以只要一通电就会叫,只能发出固定频率的声音。

```
int speakerPin = 8;  //控制喇叭的引脚
int value = 10; //控制喇叭响的时间，可自行更改
void setup()
{
   pinMode(speakerPin, OUTPUT);
}
void loop()
{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；
}
```

4、激光传感器



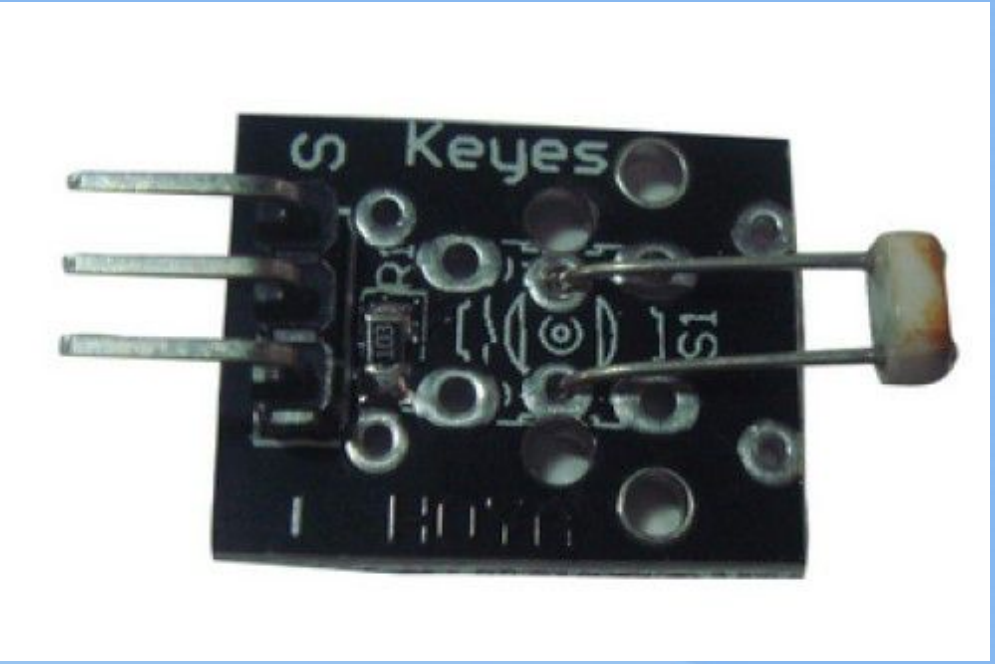
- 激光传感器通过S端来开启，可以发射持续的激光，也可以发射脉冲波。可用于玩具激光枪，或者激光测距等各种用途。

实例程序

```
void setup()
{
 pinMode(13, OUTPUT);  //   定义13脚为数字输出接口
}

void loop() {
  digitalWrite(13, HIGH);   // 打开激光头
  delay(1000);              // 延时一秒
  digitalWrite(13, LOW);    // 关闭激光头
  delay(1000);              // 延时一秒
}
```

5、光敏传感器



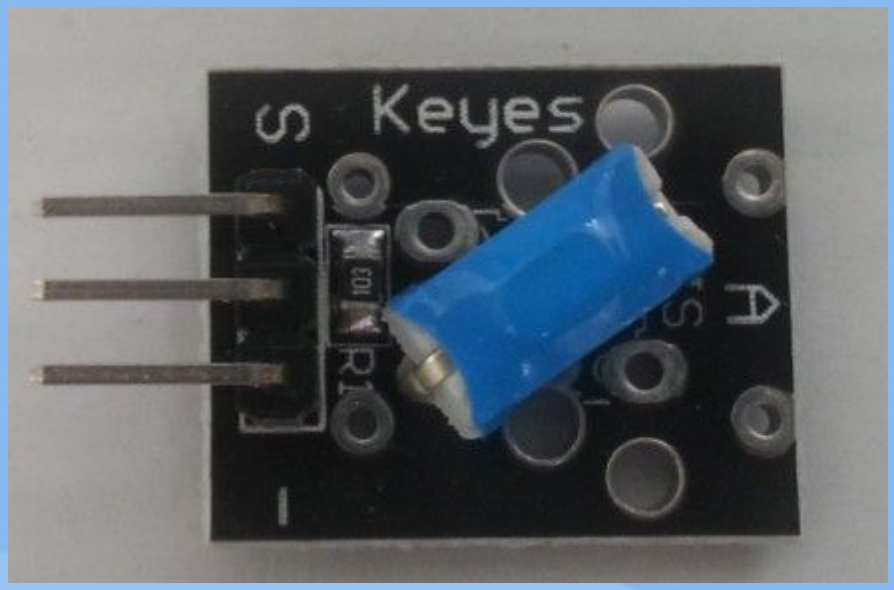
- 光敏传感器实质是一个光敏电阻，根据光的照射强度会改变其自身的阻值。

编程原理

- 将光敏电阻的S端接在一个模拟输入口，光强的变化会改变阻值，从而改变S端的输出电压。将S端的电压读出，使用串口输出到计算机显示结果。
- 因为AVR是10位的采样精度，输出值从0~1023.当光照强烈的时候，值减小，光照减弱的时候，值增加。完全遮挡光线，值最大。

```
int sensorPin = 2;
int value = 0;
void setup()
{
Serial.begin(9600); //串口波特率为9600
 }
void loop()
 {
value = analogRead(sensorPin); //读取模拟2端口
Serial.println(value, DEC); //十进制数显示结果并且换行
 delay(50);  //延时50毫秒
 }
```

6、倾斜开关



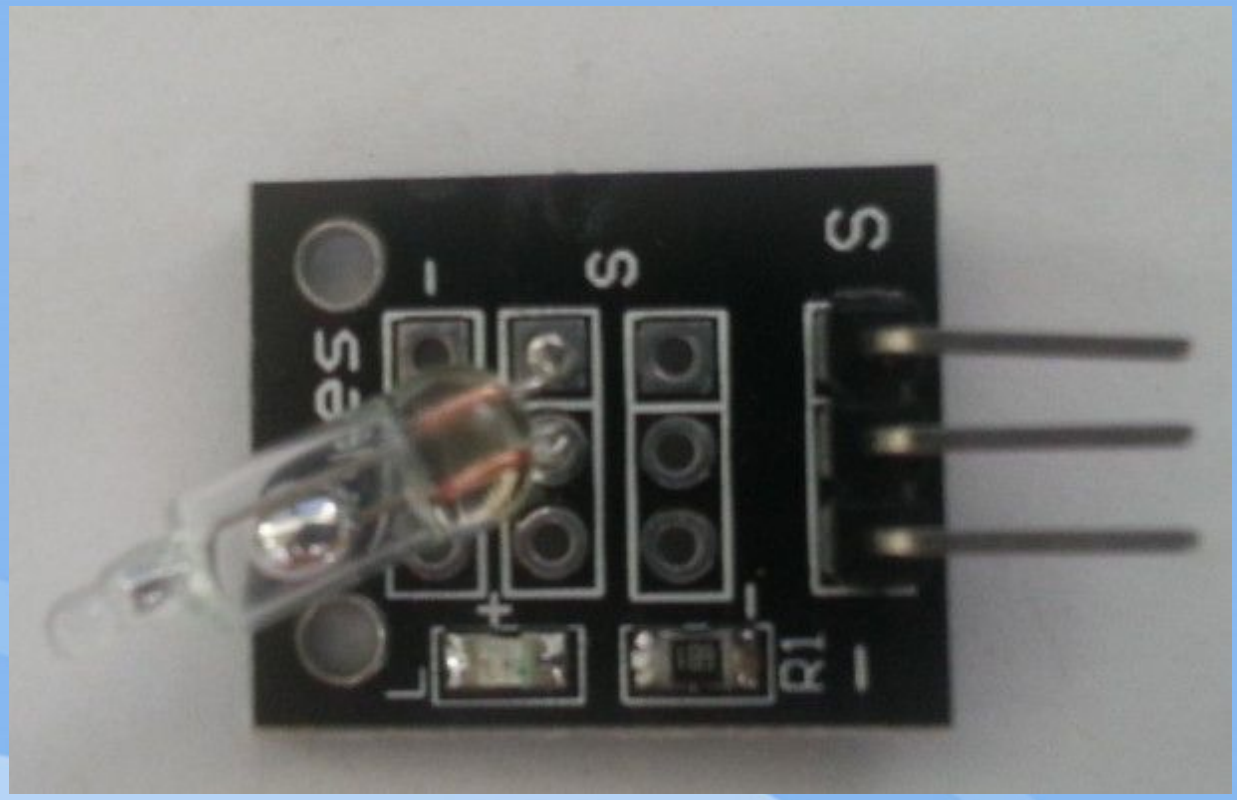
- 用于检测较小的倾斜角度

编程原理

- **倾斜开关**模块和数字13 接口自带LED 搭建简单电路，制作**倾斜**提示灯。利用数字13 接口自带的LED，将**倾斜开关**传感器接入数字3接口，当**倾斜开关**传感器感测到有倾斜信号时，LED 亮,反之则灭 。

```
int Led=13;//定义LED 接口
 int buttonpin=3; //定义倾斜开关传感器接口
int val;//定义数字变量val
void setup()
{
    pinMode(Led,OUTPUT);   //定义LED 为输出接口
    pinMode(buttonpin,INPUT);  //定义倾斜开关传感器为输出接口
}
void loop()
{
    val=digitalRead(buttonpin);   //将数字接口3的值读取赋给val
    if(val==HIGH)    //当倾斜开关传感器检测有信号时，LED 亮
   {
    digitalWrite(Led,HIGH);
    }
   else
    {
    digitalWrite(Led,LOW);
    }
}
```

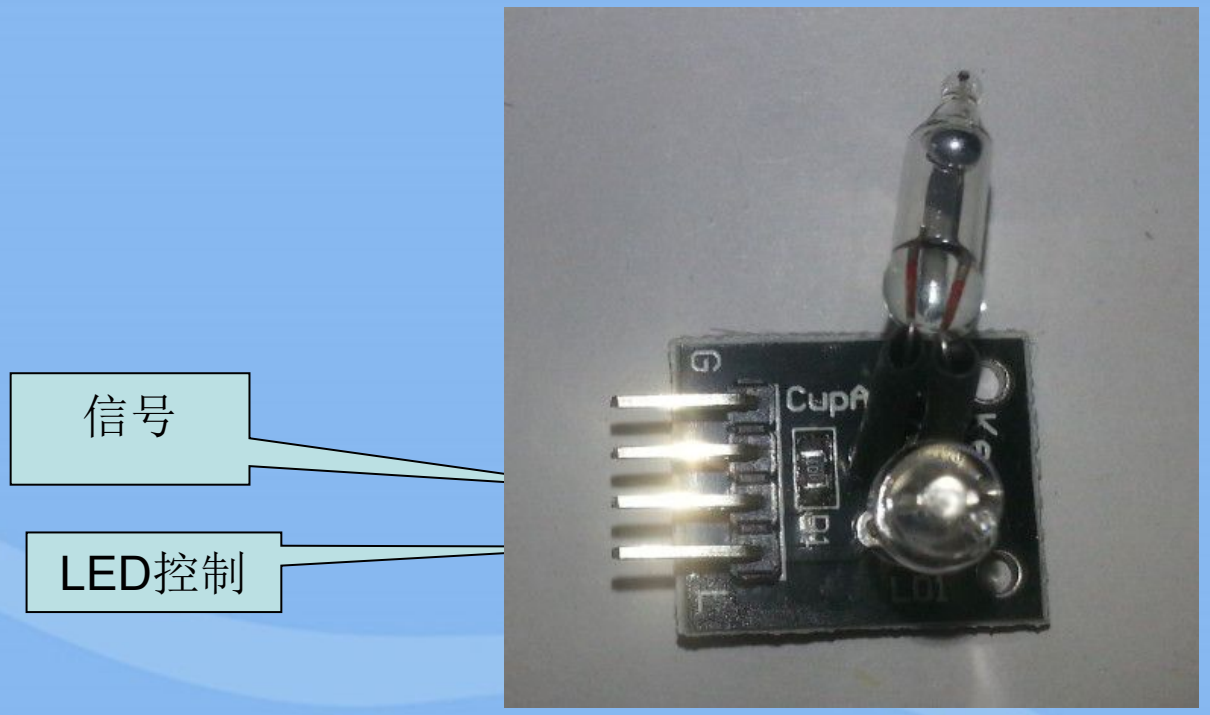
7、水银开关传感器



- 用于检测稍微大的角度，可以用于检测跌倒等。程序和上个例程通用

```
int Led=13;//定义LED 接口
 int buttonpin=3; //定义倾斜开关传感器接口
int val;//定义数字变量val
void setup()
{
    pinMode(Led,OUTPUT);   //定义LED 为输出接口
    pinMode(buttonpin,INPUT);  //定义倾斜开关传感器为输出接口
}
void loop()
{
    val=digitalRead(buttonpin);   //将数字接口3的值读取赋给val
    if(val==HIGH)    //当倾斜开关传感器检测有信号时，LED 闪烁
   {
    digitalWrite(Led,HIGH);
    }
   else
    {
    digitalWrite(Led,LOW);
    }
}
```

8、魔术光杯（一对）



- 水银开关多加了一个独立的LED，两个可以组成魔术光杯

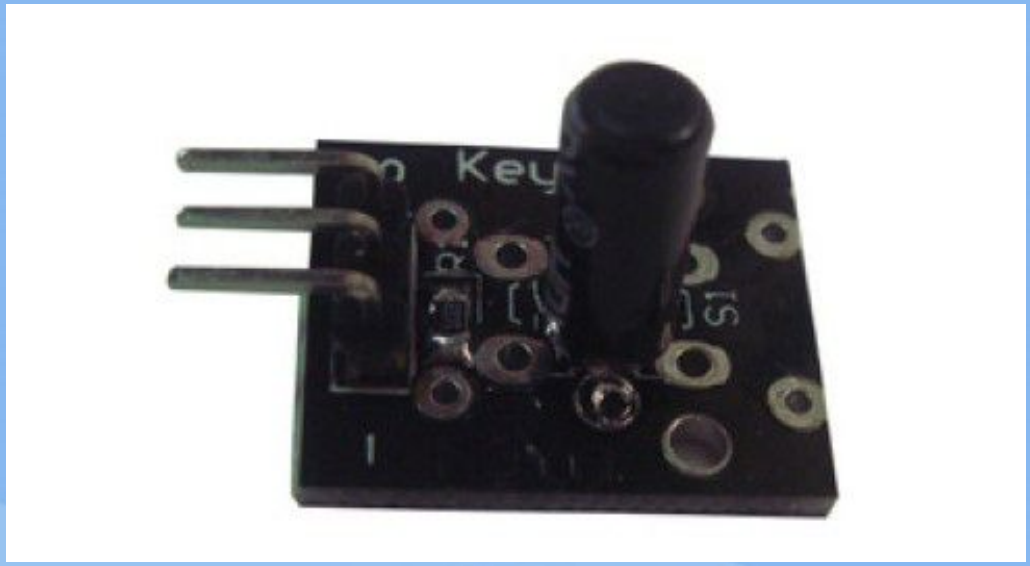
原理

- 将魔术光杯其中一个模块S脚接数字脚7，LED控制接数字脚5（PWM功能），另一个模块S脚接数字脚4，LED控制接数字脚6.
- 现象：当一个水银开关倾倒时，自己的灯会越来越暗，另一个灯会越来越亮，像心电感应一样。

```
int LedPinA = 5;
int LedPinB = 6;
int ButtonPinA = 7;
int ButtonPinB = 4;
int buttonStateA = 0;
int buttonStateB = 0;
int brightness   = 0;
void setup()
{
    pinMode(LedPinA, OUTPUT);
    pinMode(LedPinB, OUTPUT);
    pinMode(ButtonPinA, INPUT);
    pinMode(ButtonPinB, INPUT);
}
```

```
void loop()
{
      buttonStateA = digitalRead(ButtonPinA); //读取A模块
      if (buttonStateA == HIGH && brightness != 255)
       {   //当A模块检测到信号，且亮度不是最大时，亮度值增加
          brightness ++;
       }
      buttonStateB = digitalRead(ButtonPinB);
   if (buttonStateB == HIGH && brightness != 0)
   {  //当B模块检测到信号，且亮度不是最小时，亮度值减小
    brightness --;
   }
      analogWrite(LedPinA, brightness);  //  A慢渐暗
      analogWrite(LedPinB, 255 - brightness);  // B慢渐亮
      delay(25);
}
//两者相加的和为255，亮度此消彼涨的关系
```

9、震动开关

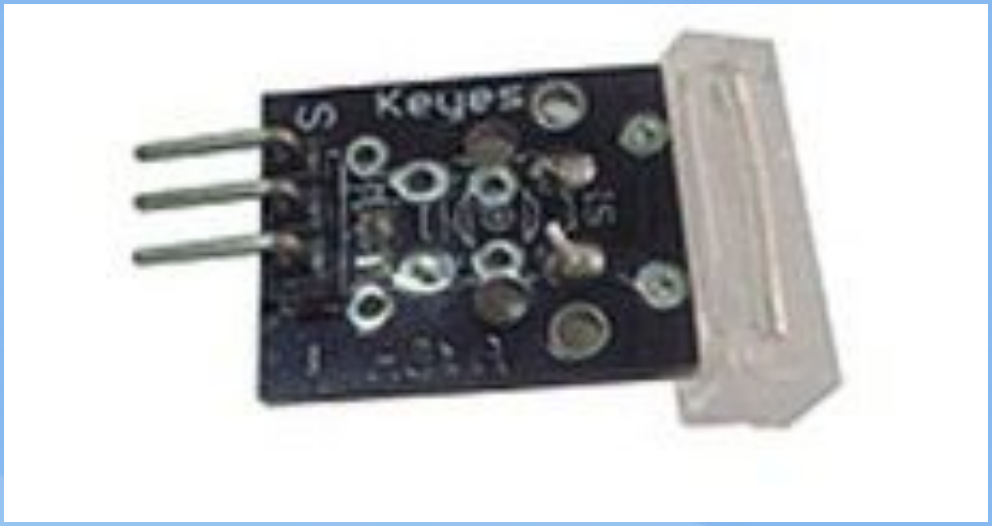


编程原理

- 震动模块和数字13 接口自带LED 搭建简单电路，制作震动闪光器。
- 利用数字13 接口自带的LED，将震动传感器接入数字3接口，当震动传感器感测到有震动信号时，LED 闪烁发光。

```
int Led=13;//定义LED 接口
int Shock=3//定义震动传感器接口
int val;//定义数字变量val
void setup()
{
  pinMode(Led,OUTPUT); //定义LED 为输出接口
  pinMode(Shock,INPUT);//定义震动传感器为输出接口
}
void loop()
{
   val=digitalRead(Shock);//将数字接口3的值读取赋给val
   if(val==HIGH)//当震动传感器检测有信号时，LED 闪烁
   {
   digitalWrite(Led,LOW);
  }
 else
  {
  digitalWrite(Led,HIGH);
  }
}
```

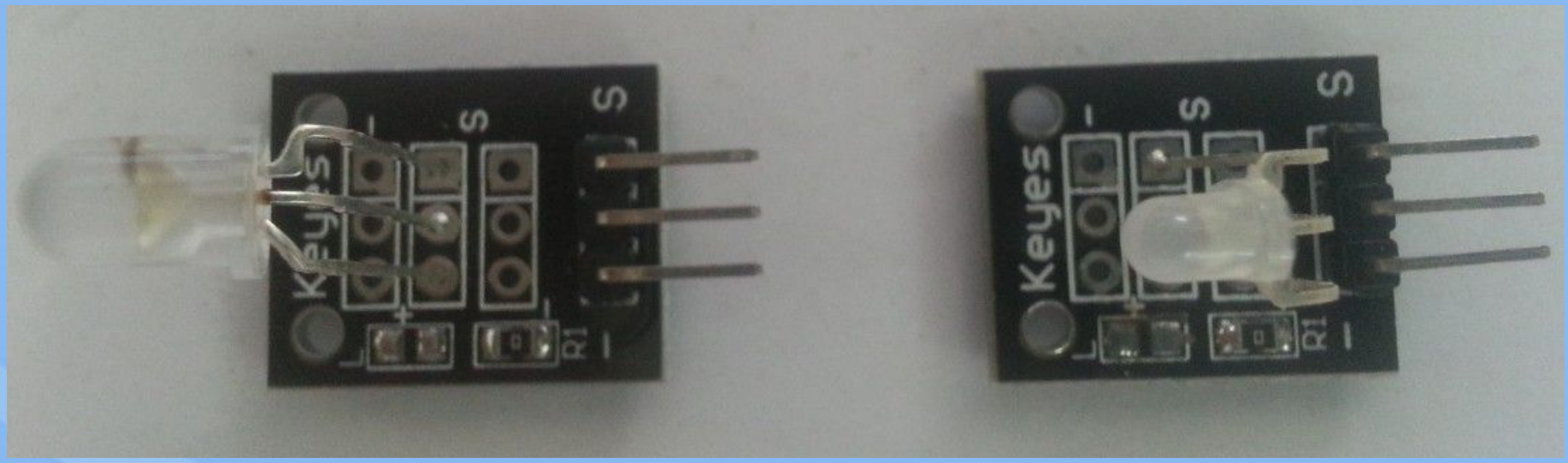
10、敲击传感器



和振动开关类似，只是能感受较小振幅的振动，更灵敏一些。
余震的时间能维持稍微久一点点，代码可以兼容振动开关

```
int Led=13;//定义LED 接口
int Shock=3//定义震动传感器接口
int val;//定义数字变量val
void setup()
{
    pinMode(Led,OUTPUT);//定义LED 为输出接口
    pinMode(Shock,INPUT);//定义震动传感器为输出接口
}
void loop()
{
   val=digitalRead(Shock);//将数字接口3的值读取赋给val
   if(val==HIGH)//当震动传感器检测有信号时，LED 闪烁
   {
    digitalWrite(Led,LOW);
    }
   else
  {
   digitalWrite(Led,HIGH);
   }
}
```

11、双色共阴LED模块



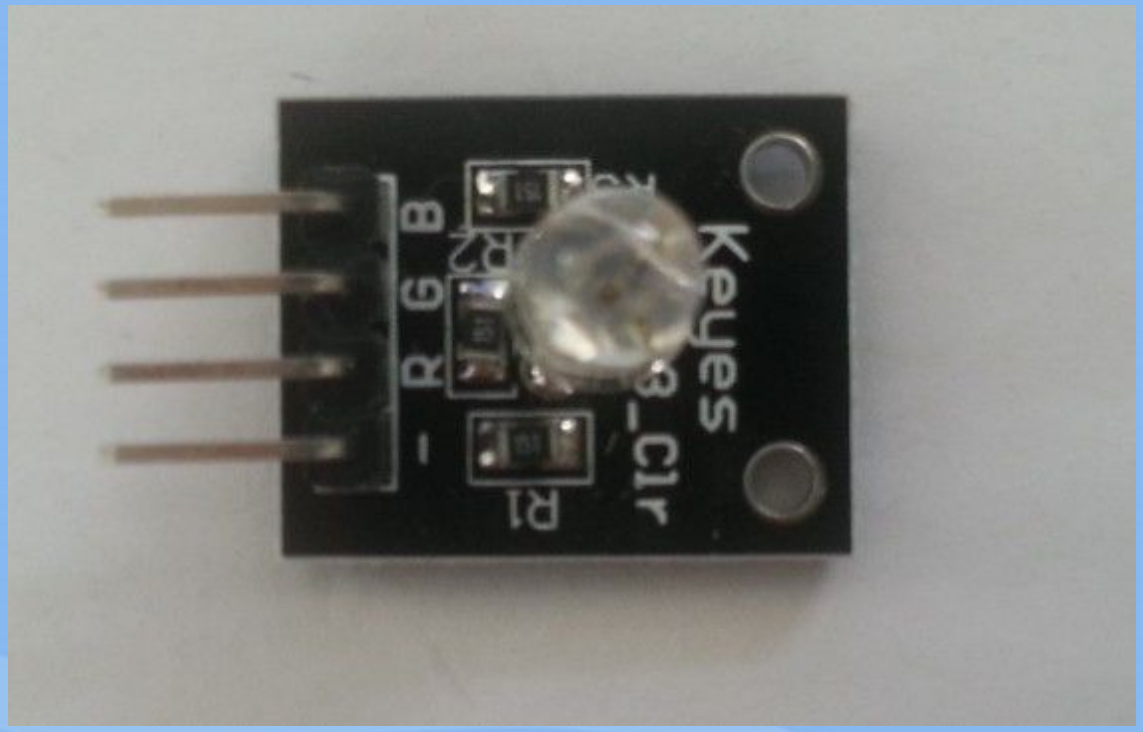
- 发光颜色：绿色+红色（左边头大一点的）黄+红（右边头小一点的）
- 产品广泛应用于电子词典、PDA、MP3、耳机、数码相机、VCD、DVD、汽车音响、通讯、计算机、充电器、功放、仪器仪表、礼品、电子玩具及移动电话等诸多领域。

编程原理

- 通过模拟端口控制LED的亮度，0~255表示从0V到5V。2种颜色的灯混合，让其值总和为255，可以看到，从红色过渡到绿色的现象，中间颜色是混合成的黄色。

```
int redpin = 11;    // 选择红灯引脚
int greenpin =10;    // 选择绿灯引脚
int val;
void setup()
{  pinMode(redpin, OUTPUT);
   pinMode(greenpin, OUTPUT);
}
void loop()
{
for(val=255; val>0; val--)
  {   analogWrite(redpin, val);
       analogWrite(greenpin, 255-val);
      delay(15);
  }
for(val=0; val<255; val++)
  {    analogWrite(redpin, val);
       analogWrite(greenpin, 255-val);
      delay(15);
  }
}
```

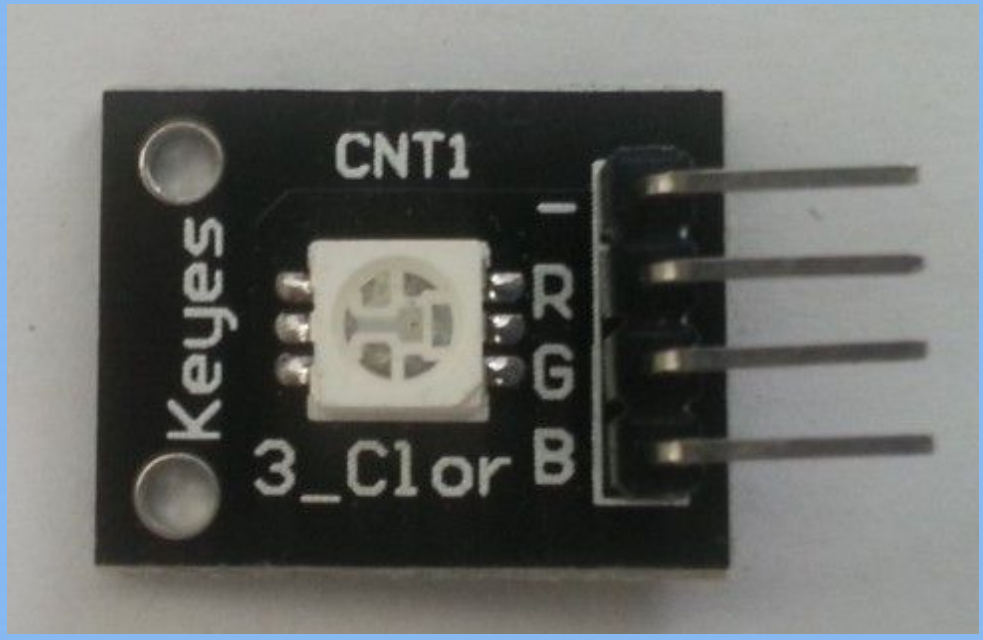
12、三色RGB模块（DIP封装）



- RGB LED模块由一个插件全彩LED制成，通过R、G、B三个引脚的PWM电压输入可以调节三种基色（红/蓝/绿）的强度从而实现全彩的混色效果

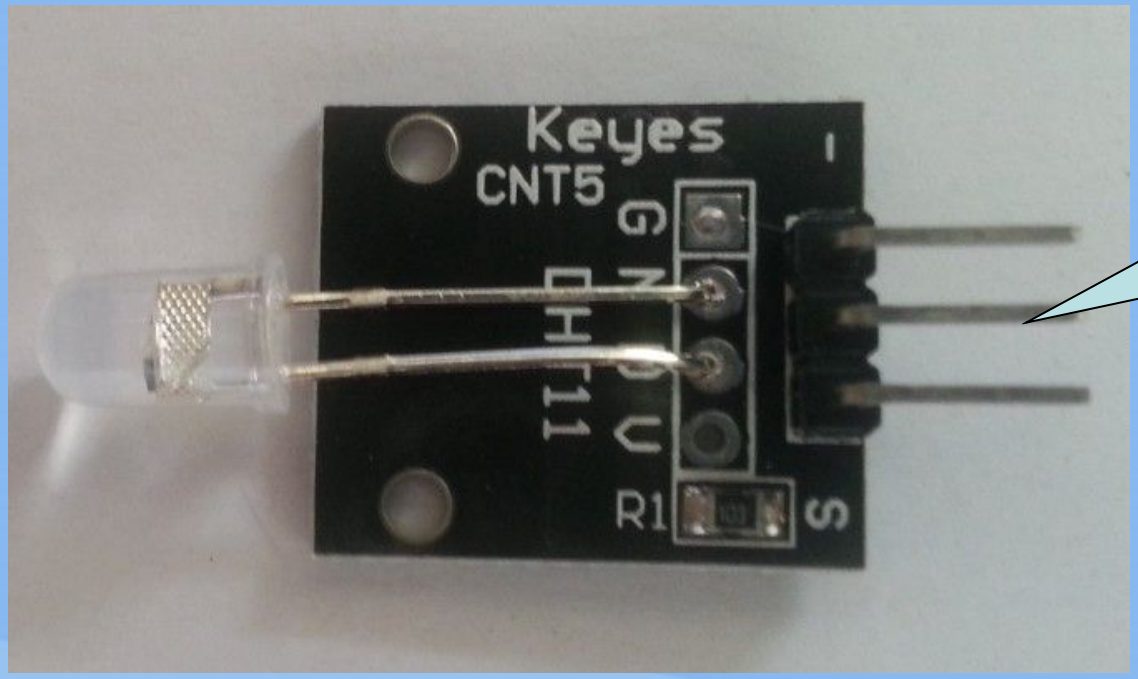
```
int redpin = 11; //select the pin for the red LED
int bluepin =10; // select the pin for the  blue LED
int greenpin =9;// select the pin for the green LED
int val;
void setup()
 {  pinMode(redpin, OUTPUT);
    pinMode(bluepin, OUTPUT);
    pinMode(greenpin, OUTPUT);
}
void loop()
{  for(val=255; val>0; val--)
   {   analogWrite(redpin, val);
       analogWrite(bluepin, 255-val);
       analogWrite(greenpin, 128-val);
        delay(2);
    }
  for(val=0; val<255; val++)
   {   analogWrite(redpin, val);
       analogWrite(bluepin, 255-val);
       analogWrite(bluepin, 128-val);
       delay(2);
    }
}
```

13 三色RGB模块（SMD封装）



- 这种LED只是封装形式不同，采用贴片封装。亮度较高。例程同上

14、7彩自动闪烁LED模块



- 通电之后能自动闪烁其中颜色。使用数字引脚直接连接，可控制其亮灭

```
void setup()
{
pinMode(13, OUTPUT);
}
void loop() {
  digitalWrite(13, HIGH);   // set the LED on
  delay(8000);              // wait for a second
  digitalWrite(13, LOW);    // set the LED off
  delay(1000);              // wait for a second
}
```

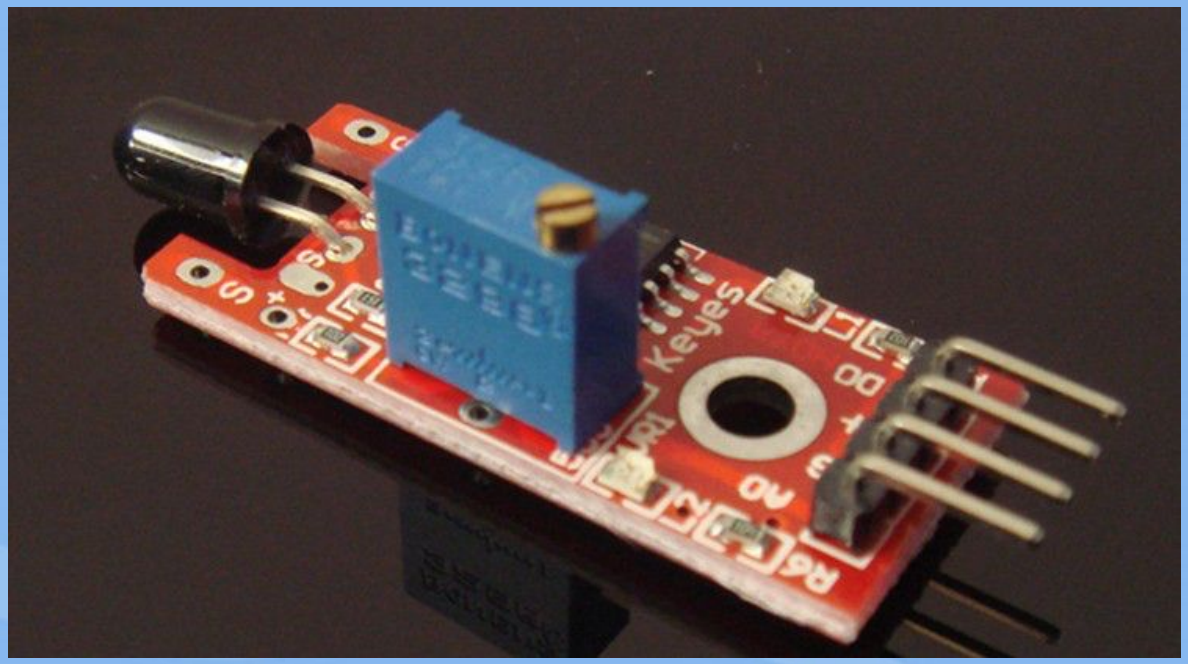
15、金属触摸传感器



- 金属触摸模块和数字13 接口自带LED 搭建简单电路，制作触摸提示灯。利用数字13 接口自带的LED，将金属触摸传感器接入数字3接口，当金属触摸传感器感测到有按键信号时，LED 亮,反之则灭.

```
int Led=13;//定义LED 接口
int buttonpin=3; //定义金属触摸传感器接口
int val;//定义数字变量val
void setup()
{
    pinMode(Led,OUTPUT);//定义LED 为输出接口
    pinMode(buttonpin,INPUT);//定义金属触摸传感器为输出接口
}
void loop()
{  val=digitalRead(buttonpin); //将数字接口3的值读取赋给val
   if(val==HIGH) //当金属触摸传感器检测有信号时，LED 亮
    {
     digitalWrite(Led,HIGH);
    }
   else
   {
     digitalWrite(Led,LOW);
    }
}
```

16、火焰传感器



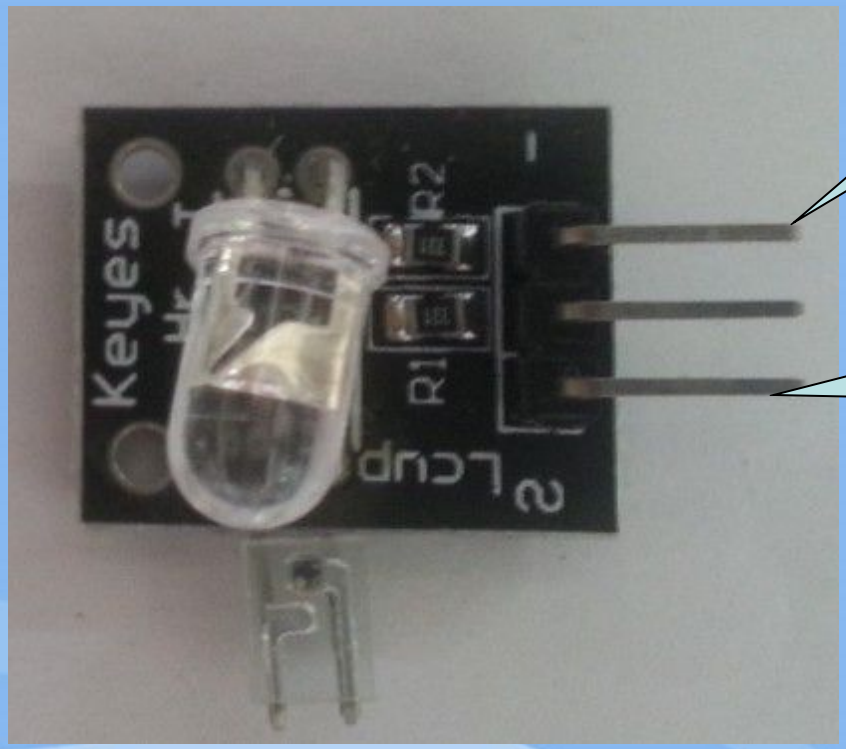
- 通过捕捉火焰中的红外波长来检测

编程原理

- **火焰模块**模块和数字13 接口自带LED 搭建简单电路，制作**火焰**提示灯
- 利用数字13 接口自带的LED，将**火焰**传感器接入数字3接口，当**火焰**传感器感测到有信号时，LED亮,反之则灭

```
int Led=13;//定义LED 接口
int buttonpin=3; //定义火焰传感器接口
int val;//定义数字变量val
void setup()
{
   pinMode(Led,OUTPUT);//定义LED 为输出接口
    pinMode(buttonpin,INPUT);//定义火焰传感器为输出接口
}
void loop()
{
    val=digitalRead(buttonpin); //将数字接口3的值读取赋给val
    if(val==HIGH)   //当火焰传感器检测有信号时，LED 亮，否则灭
      {
      digitalWrite(Led,HIGH);
      }
     else
      {
       digitalWrite(Led,LOW);
      }
}
```

17、手指测心跳模块



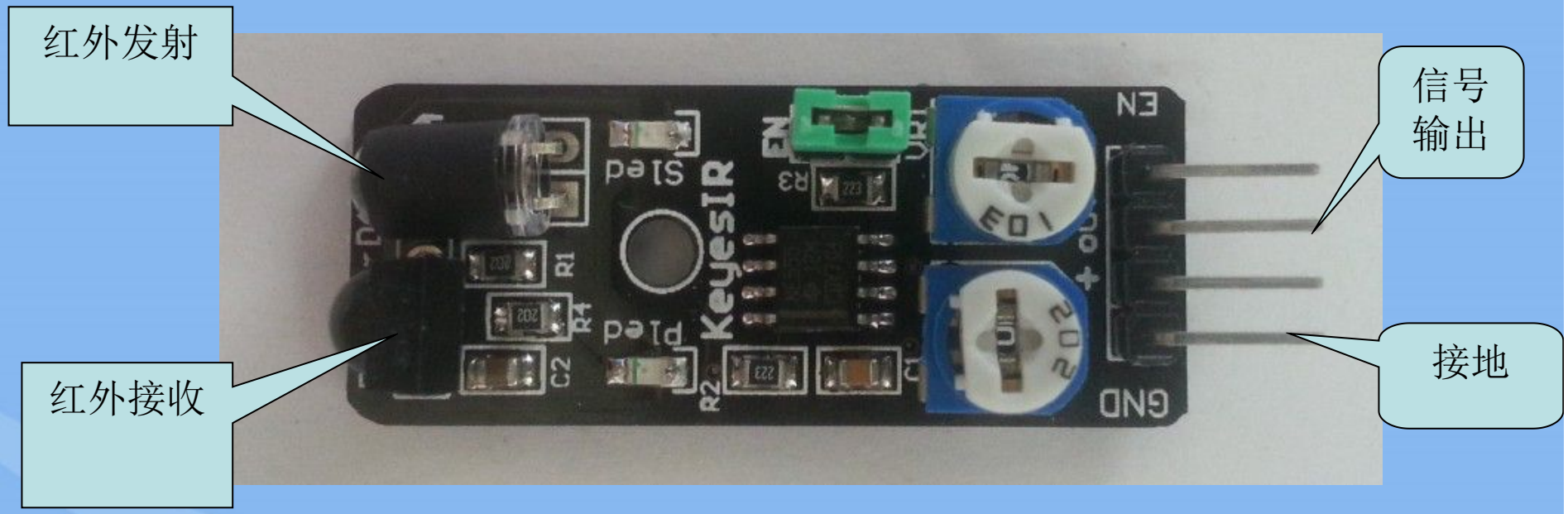
模块采用超亮红外LED和光敏晶体管来探测手指的脉搏，一个红色LED灯会随着脉搏闪动

```
int ledPin=13;   //显示灯在13引脚
int sensorPin=0;     //传感器引脚在模拟输入第0脚
double alpha=0.75; //修正值，用于增加平滑度
int period=20;
double change=0.0;
void setup()
{
        pinMode(ledPin,OUTPUT);
}
void loop()
{
        static double oldValue=0;
        static double oldChange=0;
        int rawValue=analogRead(sensorPin);  //读取传感器的值
        double value=alpha*oldValue+(1-alpha)*rawValue;
        change=value-oldValue;

        digitalWrite(ledPin,(change<0.0&&oldChange>0.0)); //输出

        oldValue=value;
        oldChange=change;
        delay(period);
}
```

18、红外避障传感器



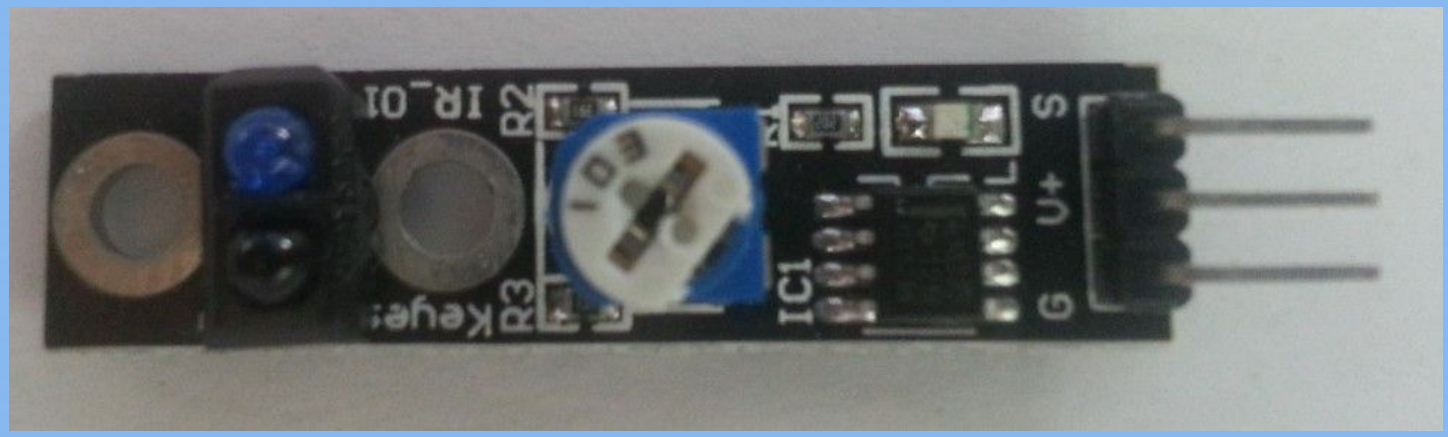
- 根据红外反射的原来来检测前方是否有物体。当前方没有物体是，红外接收不到信号。前方有物体会遮挡并反射红外光，此时能检测到信号。

编程原理

- 下面我们利用避障模块和数字13 接口自带LED 搭建简单电路，制作避障提示灯，将避障传感器接入数字3接口，当避障传感器感测到有障碍物时，输出是低电平（电路是负逻辑）信号时，LED 亮，反之没有障碍物是，输出高电平，提示灯则灭.

```
int Led=13;//定义LED 接口
int buttonpin=3; //定义避障传感器接口
int val;//定义数字变量val
void setup()
{
    pinMode(Led,OUTPUT);//定义LED 为输出接口
    pinMode(buttonpin,INPUT);//定义避障传感器为输出接口
}
void loop()
{
    val=digitalRead(buttonpin);//将数字接口3的值读取赋给val
    if(val==LOW)//当避障传感器检测有障碍物时为低电平
      {
        digitalWrite(Led,HIGH); //提示有障碍物
      }
    else
      {
       digitalWrite(Led,LOW);  //没有障碍物
      }
}
```

19、寻线传感器



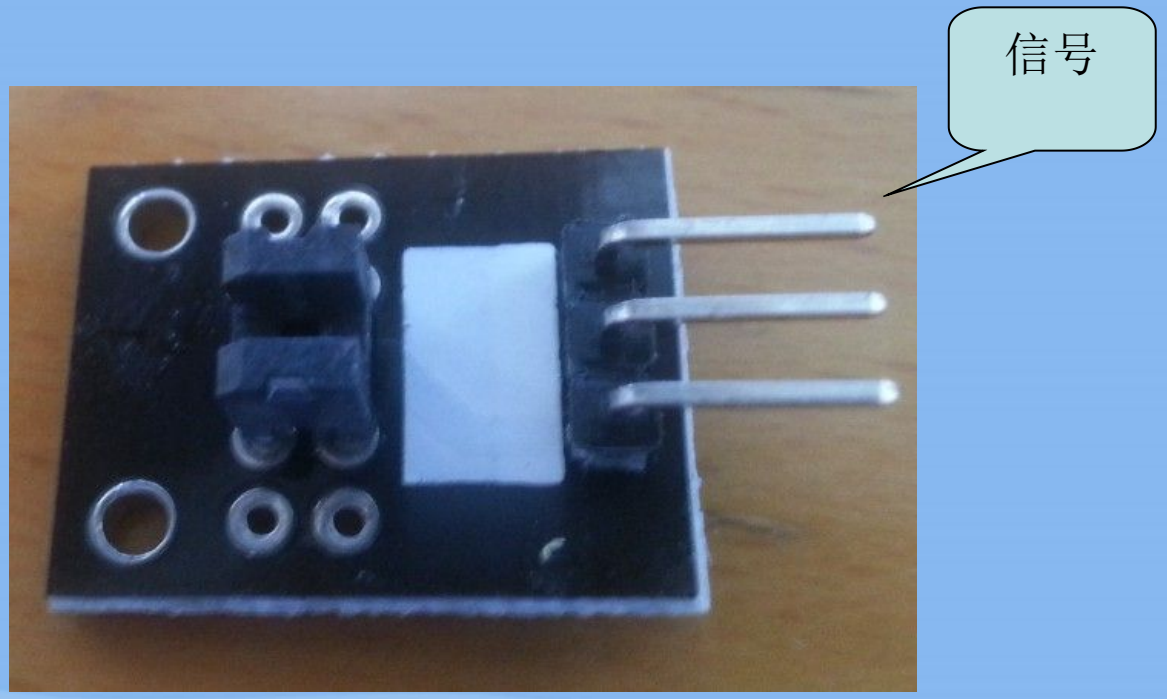
- 原理同红外避障传感器，只是发射功率比较小，遇到白色反射红外，遇到黑色被吸收红外。以此来寻找地面的黑线。

编程原理

- **寻线**模块和数字13 接口自带LED 搭建简单电路，制作**寻线**提示灯
- 利用数字13 接口自带的LED，将**寻线**传感器接入数字3接口，当**寻线**传感器感测到有反射信号时（白色），LED 亮,反之（黑线）则灭.

```
int Led=13;//定义LED 接口
int buttonpin=3; //定义寻线传感器接口
int val;//定义数字变量val
void setup()
{
    pinMode(Led,OUTPUT);//定义LED 为输出接口
    pinMode(buttonpin,INPUT);//定义寻线传感器为输出接口
}
void loop()
{
   val=digitalRead(buttonpin);//将数字接口3的值读取赋给val
   if(val==HIGH)//当寻线传感器检测有反射信号时，LED 亮
   {
     digitalWrite(Led,HIGH);
    }
  else
   {
   digitalWrite(Led,LOW);
    }
}
```

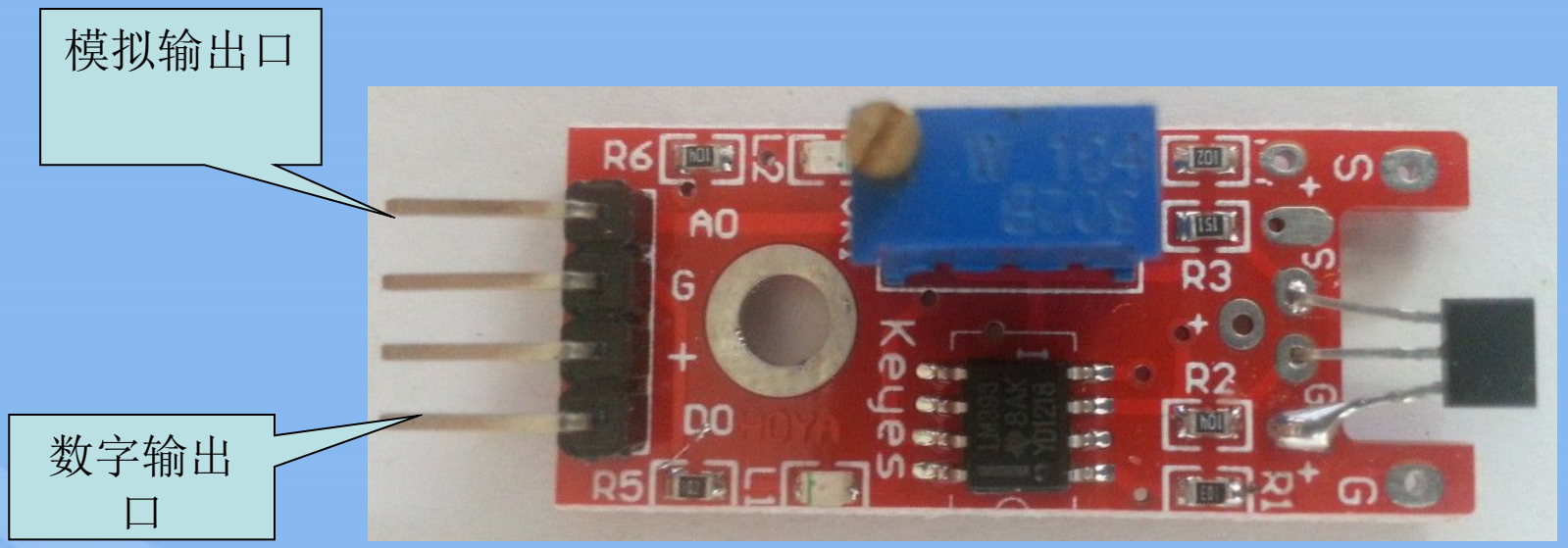
20、光折断传感器



原理同红外传感器，当没有遮挡的时候，发射的光线直接到红外接收处被吸收，此时有信号输出。如果有遮挡则无信号输出。广泛用于测速。

```
int Led=13;//定义LED 接口
int buttonpin=3; //定义光遮断传感器接口
int val;//定义数字变量val
void setup()
{
    pinMode(Led,OUTPUT);//定义LED 为输出接口
    pinMode(buttonpin,INPUT);//定义光遮断传感器为输出接口
}
void loop()
{
    val=digitalRead(buttonpin);//将数字接口3的值读取赋给val
    if(val==HIGH)//当光遮断传感器检测有信号时，LED 亮
    {
       digitalWrite(Led,HIGH);
     }
    else
     {
       digitalWrite(Led,LOW);
     }
}
```

21、线性霍尔磁力传感器



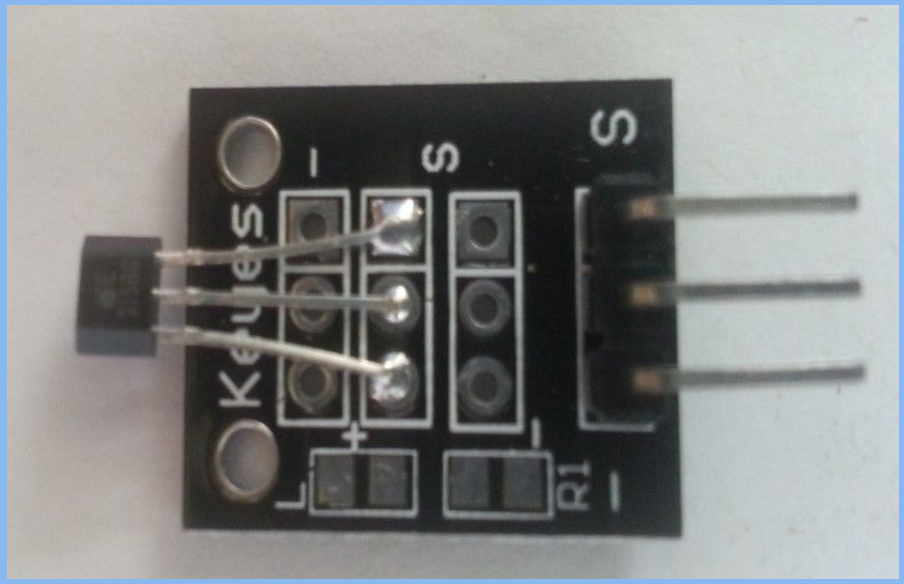
- 霍尔磁力传感器能检测到磁场，从而输出检测信号。模拟口能通过输出线性电压的变化来揭示出磁场的强弱。数字输出口是达到某个阈值才会输出高低电平。可调电阻能改变检测的灵敏度。

编程原理

- 我们选择数字口作为输出，将D接开发板数字引脚3，使用板上的13脚和LED连通，用于观察磁场的有无。当磁铁靠近时，13脚的灯灭，反之则亮。

```
int Led=13;//定义LED 接口
int buttonpin=3; //定义线性霍尔传感器接口
int val;//定义数字变量val
void setup()
{
    pinMode(Led,OUTPUT);//定义LED 为输出接口
    pinMode(buttonpin,INPUT);//定义线性霍尔传感器为输出接口
}
void loop()
{
    val=digitalRead(buttonpin);//将数字接口3的值读取赋给val
    if(val==HIGH)//当霍尔传感器检测没有磁场信号时，LED 亮
    {
       digitalWrite(Led,HIGH);
    }
    else  //当霍尔传感器检测到有磁场信号时，LED灭
    {
      digitalWrite(Led,LOW);
    }
}
```

22、模拟霍尔传感器



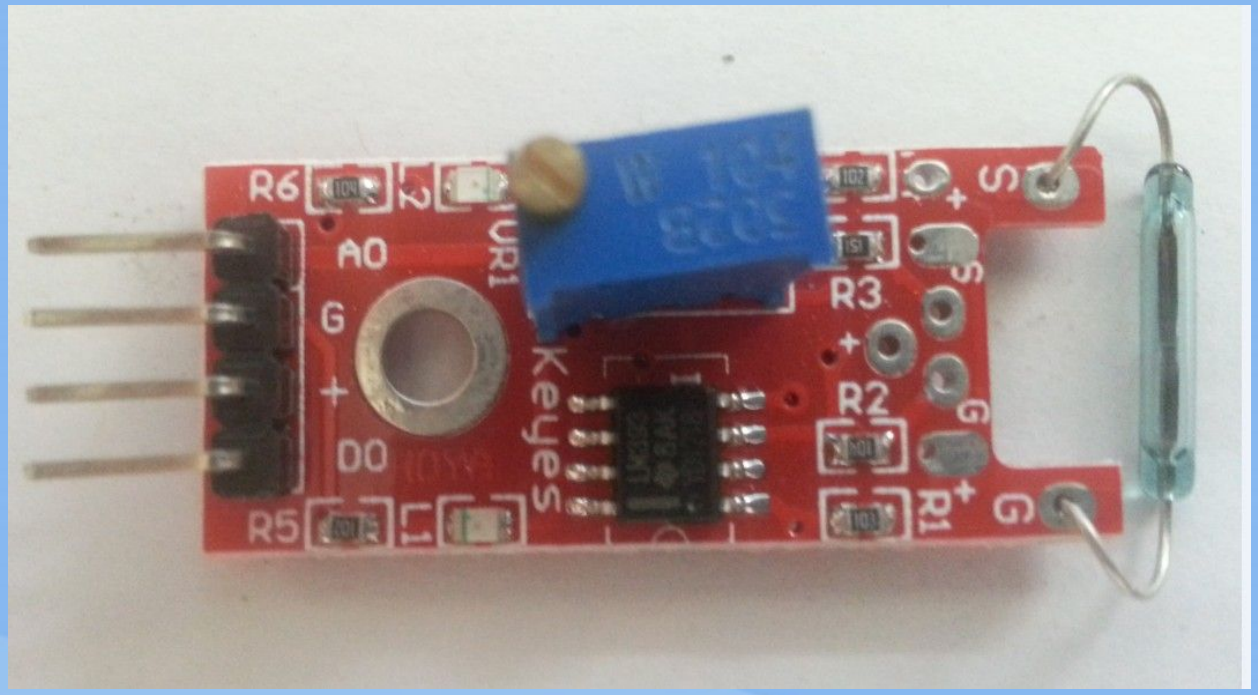
- 和线性霍尔类似。有磁场则输出数值改变

编程原理

- 传感器A0口接Arduino板模拟口的A1口，读取传感器的值。当磁铁离传感器近时，数值变大，反之数值变小。

```
int sensorPin = 1;
int value = 0;
void setup()
{
Serial.begin(9600); //串口波特率为9600
 }
void loop()
 {
value = analogRead(sensorPin); //读取模拟1端口
Serial.println(value, DEC); //十进制数显示结果并且换行
 delay(50);  //延时50毫秒
 }
```

23、大磁簧传感器



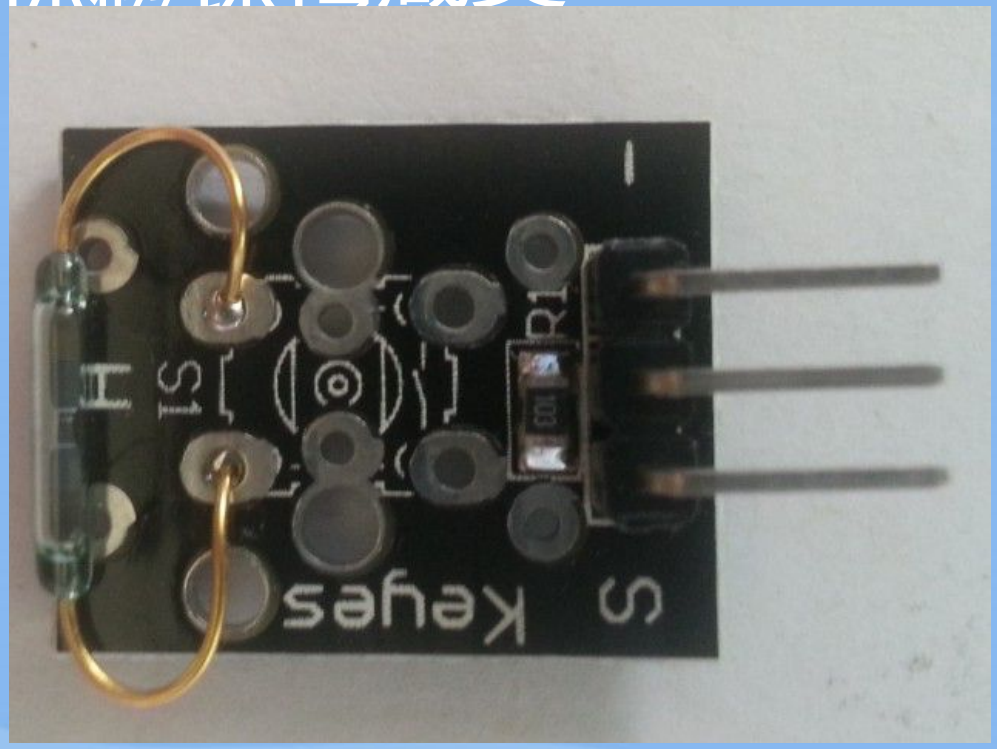
同样是检测磁场的传感器。霍尔通常可以用到智能车测速，或者流水线计数计件，磁环传感器一般只用来测试有没有磁场的。

编程原理

- **磁环**模块和数字13 接口自带LED 搭建简单电路，制作**磁场**提示灯
- 利用数字13 接口自带的LED，将**磁环**传感器接入数字3接口，当**磁环**传感器感测到有磁铁靠近时，LED 亮,反之则灭.

```
int Led=13;//定义LED 接口
int buttonpin=3; //定义磁环传感器接口
int val;//定义数字变量val
void setup()
{
    pinMode(Led,OUTPUT);//定义LED 为输出接口
     pinMode(buttonpin,INPUT);//定义磁环传感器为输出接口
}
void loop()
{
    val=digitalRead(buttonpin);//将数字接口3的值读取赋给val
    if(val==HIGH)//当磁环传感器检测有信号时，LED 亮
     {
      digitalWrite(Led,HIGH);
      }
     else //没有信号则灭
      {
       digitalWrite(Led,LOW);
       }
}
```

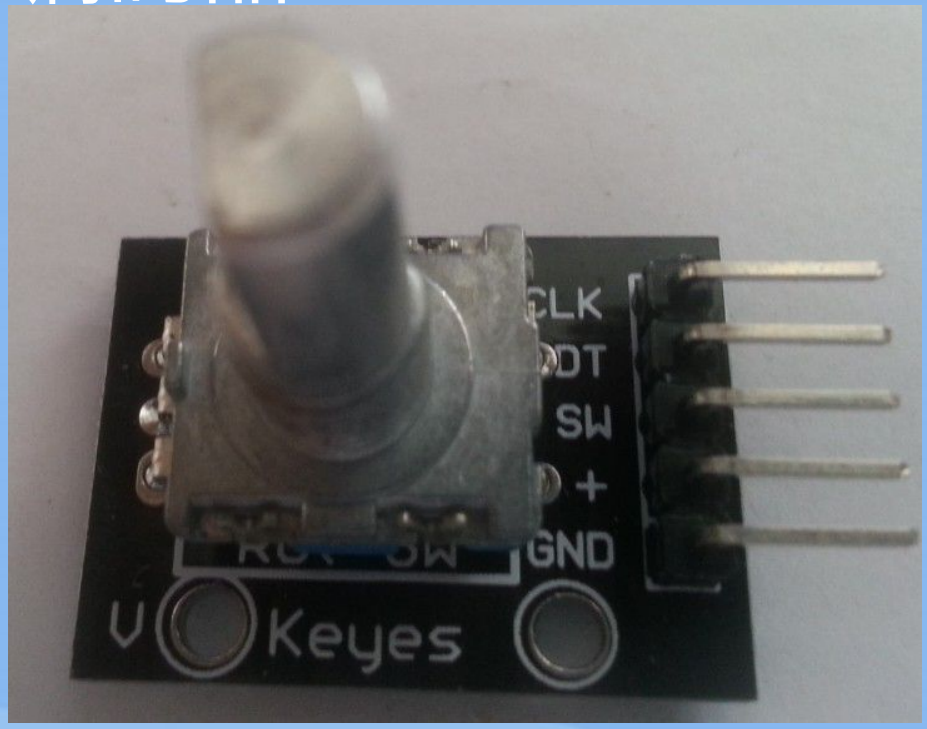
24、迷你磁环传感器



- 和大磁簧传感器是一类，差异很小，只是灵敏度更弱些。迷你磁环没有模拟输出，直接接数字接口。

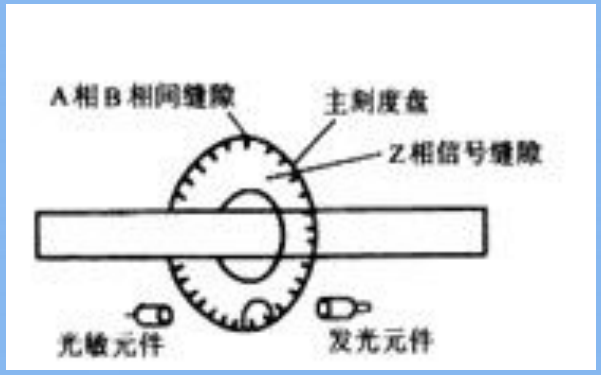
```
int Led=13;//定义LED 接口
int buttonpin=3; //定义磁环传感器接口
int val;//定义数字变量val
void setup()
{
    pinMode(Led,OUTPUT);//定义LED 为输出接口
     pinMode(buttonpin,INPUT);//定义磁环传感器为输入接口
}
void loop()
{
    val=digitalRead(buttonpin);//将数字接口3的值读取赋给val
    if(val==HIGH)//当磁环传感器检测有信号时，LED 亮
     {
      digitalWrite(Led,HIGH);
      }
     else
      {
       digitalWrite(Led,LOW);
       }
}
```

25、旋转编码器



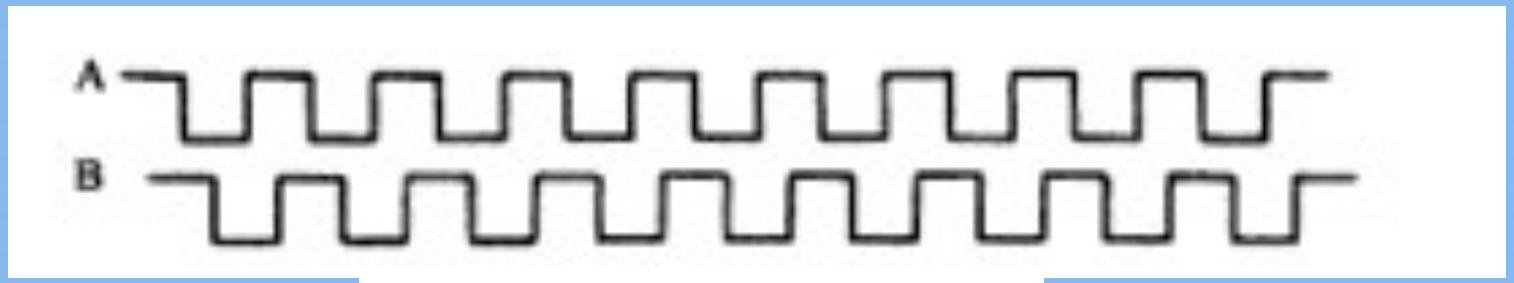
编程原理

- 旋转编码器可通过旋转可以计数正方向和反方向转动过程中输出脉冲的次数



增量编码器给出两相方波，它们的相位差90°，通常称为A通道和B通道。其中一个通道给出与转速相关的信息，与此同时，通过两个通道信号进行顺序对比，得到旋转方向的信息。

编程原理



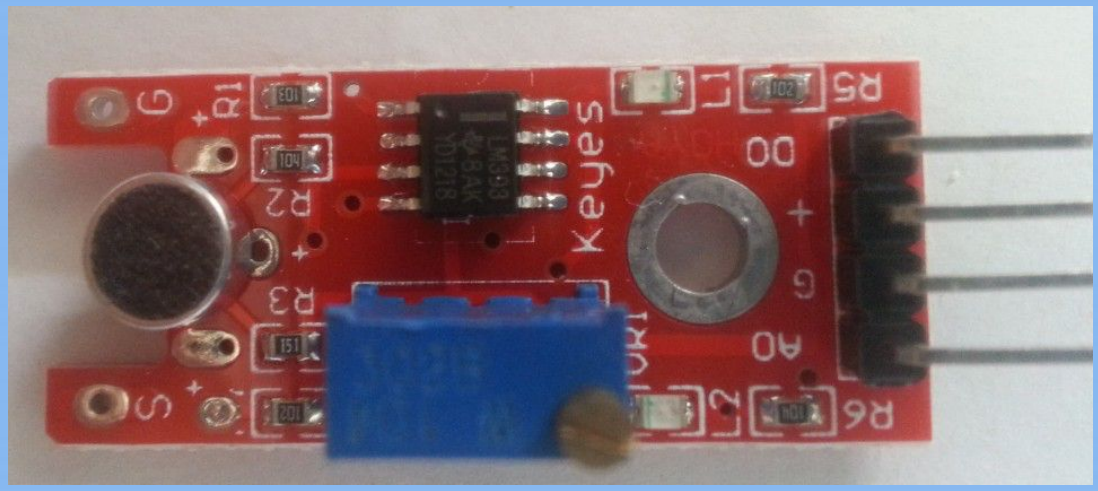
增量编码器

顺时针运动	逆时针运动
A B	A B
1 1	1 1
0 1	1 0
0 0	0 0
1 0	0 1

```
const int interruptA = 0;       // 中断 Interrupt 0 在 pin 2 上
int CLK = 2;     // PIN2 脉冲信号
int DAT = 3;     // PIN3
int SW= 4;     // PIN4 往下按压的开关信号
int LED1 = 5;    // PIN5
int LED2 = 6;    // PIN6
int COUNT = 0;
void setup()
 {  attachInterrupt(interruptA, RoteStateChanged, FALLING);
 // 高电平变为低电平触发，调用中断处理子函数RoteStateChanged（）
  pinMode(CLK, INPUT);
  digitalWrite(2, HIGH);   // 上拉电阻
  pinMode(DAT, INPUT);
  digitalWrite(3, HIGH);   // 上拉电阻
  pinMode(SW, INPUT);
  digitalWrite(4, HIGH);    // 上拉电阻
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
   Serial.begin(9600);    //设置波特率为9600
 }
```

```
void loop()
{   if  (!(digitalRead(SW)))  //如果按下按钮
    {     COUNT = 0;     //计数清零
          Serial.println("STOP COUNT = 0");  //串口输出清零
          digitalWrite(LED1, LOW);  //LED1灯灭
          digitalWrite(LED2, LOW);   //LED2灯灭
          delay (2000);   //延时2秒
    }
     Serial.println(COUNT);   //如果没有按钮，输出计数值
}
void RoteStateChanged()  //当CLK下降沿触发的时候，进入中断
{   if  (digitalRead(DAT))   // 当DAT为高电平时，是前进方向
    {   COUNT++; //计数器累加
        digitalWrite(LED1, HIGH); // LED1亮
        digitalWrite(LED2, LOW);  //LED2 灭
        delay(20);
      }
    else                  // 当 DAT是低电平是反方向滚动
    {    COUNT--;  //计数器累减
        digitalWrite(LED2, HIGH); // LED2亮
        digitalWrite(LED1, LOW);  // LED1灭
        delay(20);
    }
}
```

26、麦克风声音传感器



- 模块有2个输出：
- 1、AO，模拟量输出，实时输出麦克风的电压信号
- 2、DO，当声音强度到达某个阀值时，输出高低电平信号，【阀值-灵敏度可以通过电位器调

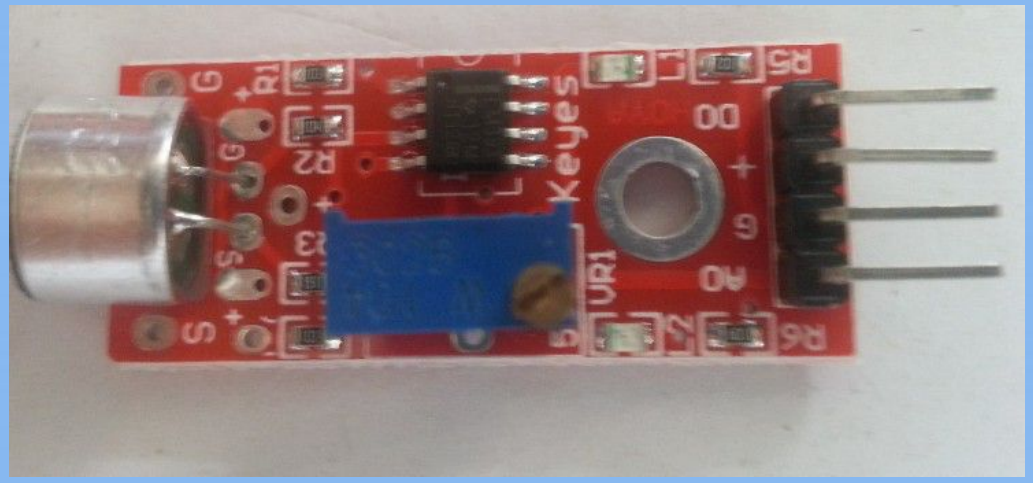
数字输出

```
nt Led=13;//定义LED 接口
int buttonpin=3 //定义传感器D0接口
int val;//定义数字变量val
void setup()
{    pinMode(Led,OUTPUT);//定义LED 为输出接口
     pinMode(buttonpin,INPUT);//定义传感器D0为输出接口
 }
void loop()
{
    val=digitalRead(buttonpin);//将数字接口3的值读取赋给val
    if(val==HIGH)//当声音检测模块检测有信号时，LED 闪烁
    {  digitalWrite(Led,HIGH)
     }
     else
    {  digitalWrite(Led,LOW)
    }
}
```

模拟输出

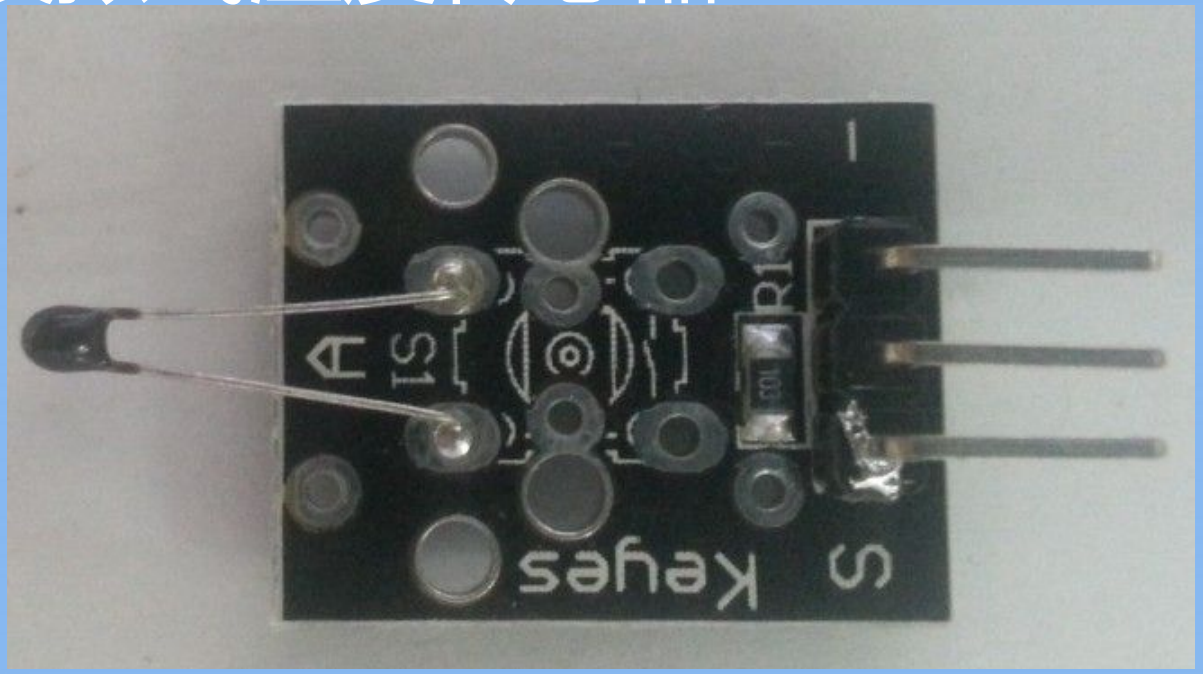
```
int sensorPin = A5;    // 选择模拟5输入端口
int ledPin = 13;       // 选择LED显示端口
int sensorValue = 0;  // 声音值变量
void setup()
{  pinMode(ledPin, OUTPUT);
   Serial.begin(9600);
}
void loop()
 {  sensorValue = analogRead(sensorPin);  //读声音传感器的值
    digitalWrite(ledPin, HIGH);  //灯闪烁
    delay(50);
    digitalWrite(ledPin, LOW); //灯闪烁
    delay(50);
    Serial.println(sensorValue, DEC);  //以10进制的形式输出声音值
}
```

27、高感度声音传感器



- 灵敏度高于前面的声音传感器。例程与前面一致。

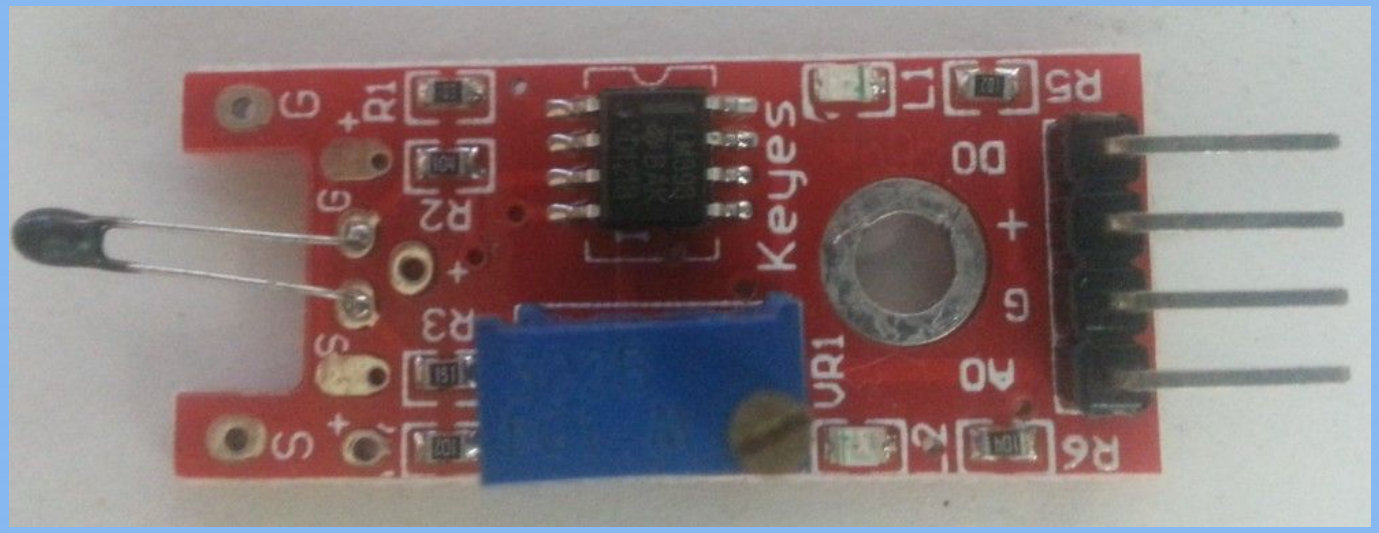
28、模拟式温度传感器



- 该模块是基于热敏电阻（阻值随外界环境温度变化而变化）的工作原理，能够实时感知周边环境温度的变化，我们把数据送到Arduino的模拟 I/O口，接来下我们只要经过简单的编程就能将传感器输出的数据转换为摄氏温度值，并加以显示，使用起来还是方便、有效的呢，借此广泛应用于园艺、家庭警报系统等装置中。

```
#include <math.h>
double Thermister(int RawADC)
{ double Temp; Temp = log(((10240000/RawADC) - 10000));
Temp = 1 / (0.001129148 + (0.000234125 + (0.0000000876741 *
Temp * Temp ))* Temp );
 Temp = Temp - 273.15;
// 转换温度值;
return temp;
}
void setup()
{ Serial.begin(9600);
}
void loop()
{ Serial.print(Thermister(analogRead(0))); // 输出转换好的温度值
Serial.println("c");
delay(500);
 }
```

29、数字温度传感器

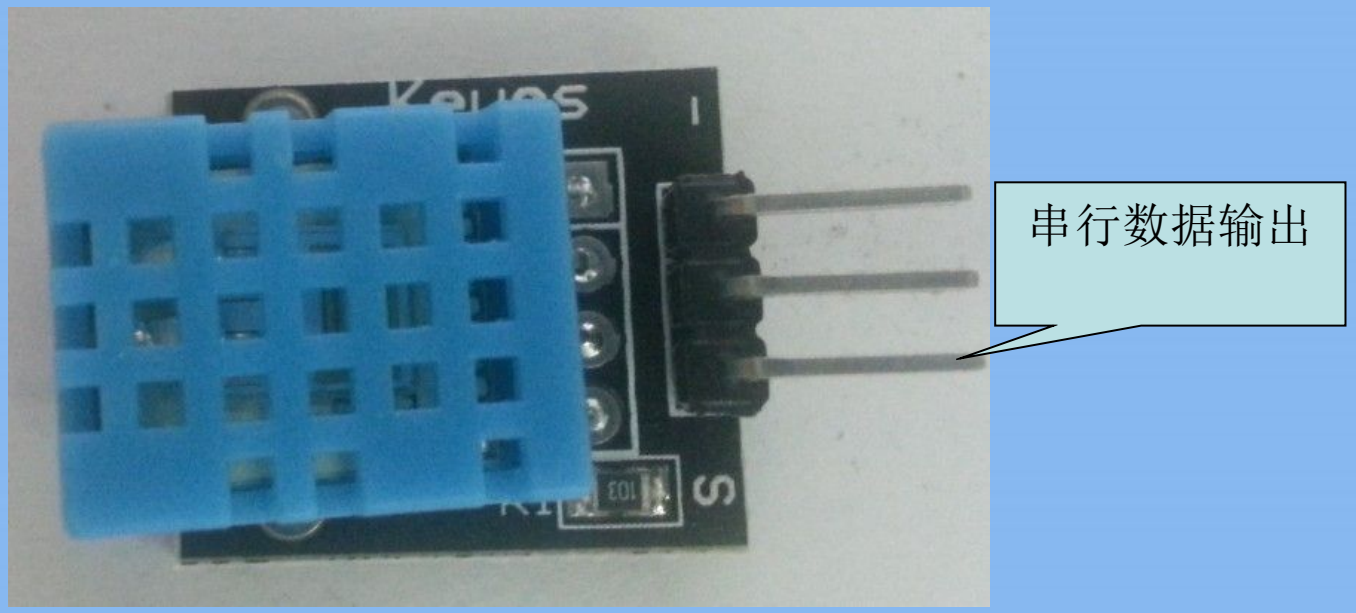


- 和前面的传感器一样，只是增加了数字输出。通过可调电阻调节阈值。达到某个值的时候，输出高电平，低于某个值的时候输出低电平。

编程原理

- **数字温度**模块和数字13 接口自带LED 搭建简单电路，制作**温度**提示灯
- 利用数字13 接口自带的LED，将**数字温度**传感器接入数字3接口，当**数字温度**传感器感测到高于某个值时，LED 亮,反之则灭.
- 可以通过调节可调电阻来设定阈值。

30、温湿度传感器



- DHT11数字温湿度传感器是一款含有已校准数字信号输出的温湿度复合传感器，它应用专用的数字模块采集技术和温湿度传感技术。

编程原理

- DHT11是一款数字式的温湿度传感器，使用一根信号线传输数据。其读取步骤如下：
- 1、将该引脚改为输出模式，先将数据线的电平拉低（将该引脚置LOW)持续时间超过18MS以上。
- 2、再将该引脚置为高电平,持续时间40US.
- 3、再把该引脚设置为读取模式，此时读到低电平后，延时80US，再读到高电平后80US，开始能接收到有效数据。
- 4、数据总共有5个字节，忽略校验位，有四位是有效数据。第0字节是湿度的整数位，第1字节是湿度的小数为，第2字节是温度的整数位，第3字节是温度的小数位。

```
int DHpin = 8;  //数字第8引脚读取
byte dat[5];  //设置5个字节的数组
byte read_data()
 { byte data;
     for(int i=0; i<8; i++)
     { if(digitalRead(DHpin) == LOW)
        { while(digitalRead(DHpin) == LOW); //等待50us；
          delayMicroseconds(30); //判断高电平的持续时间，以判定数据是‘0’还是‘1’；
          if(digitalRead(DHpin) == HIGH) data |= (1<<(7-i)); //高位在前，低位在后；
          while(digitalRead(DHpin) == HIGH); //数据‘1’，等待下一位的接收；
          }
     }
   return data;
 }
```

//拉低总线，发开始信号；
//延时要大于18ms，以便DHT11能检测到开始信号；

//等待DHT11响应；
//改为输入读取模式

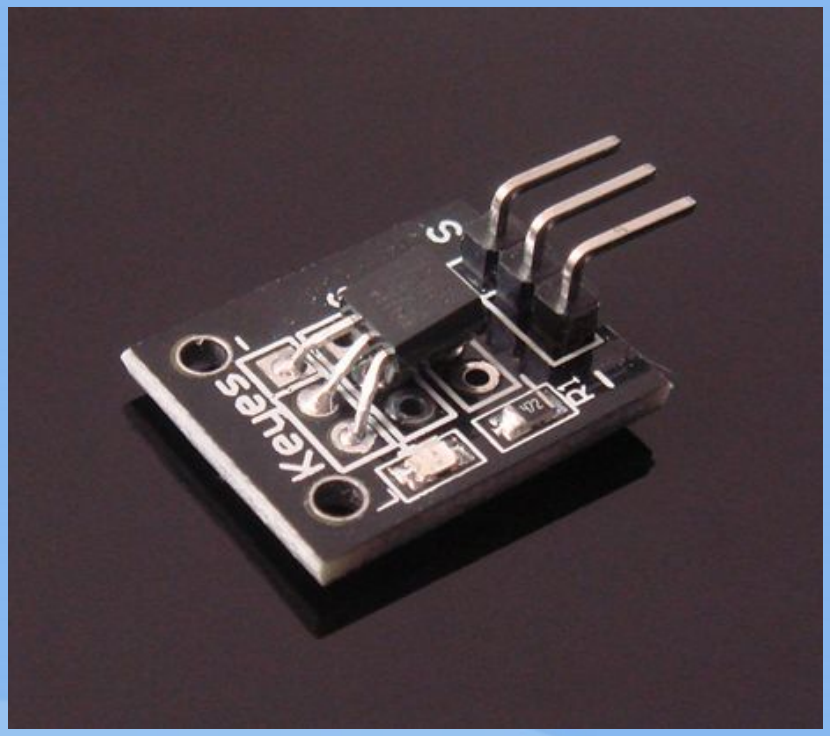
//DHT11发出响应，拉低总线80us；

//DHT11拉高总线80us后开始发送数据；
//接收温湿度数据，校验位不考虑；

//改为输出模式
//发送完一次数据后释放总线，等待主机的下一次开始信号；

```
void setup()
{ Serial.begin(9600);
pinMode(DHpin,OUTPUT);
}
void loop()
{ start_test();
 Serial.print("Current humdity = ");
 Serial.print(dat[0], DEC); //显示湿度的整数位；
 Serial.print('.');
 Serial.print(dat[1],DEC); //显示湿度的小数位；
 Serial.println('%');
 Serial.print("Current temperature = ");
 Serial.print(dat[2], DEC); //显示温度的整数位；
 Serial.print('.');
 Serial.print(dat[3],DEC); //显示温度的小数位；
 Serial.println('C');
 delay(700);
}
```

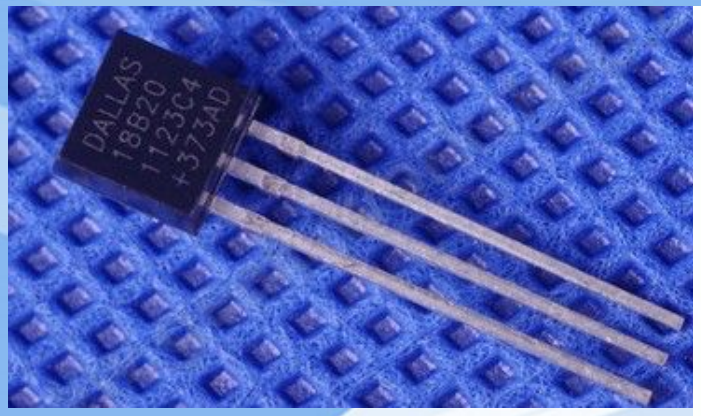
31、DS18b20 数字温度传感器模块

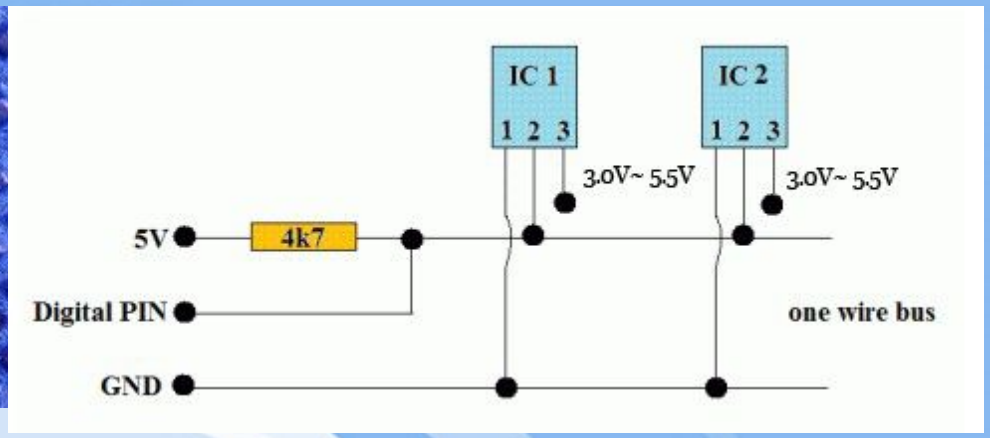


DS18B20数字温度传感器有别于传统的AD采集的温度传感器，其采用单线制总线，能直接输出温度数据。

芯片介绍

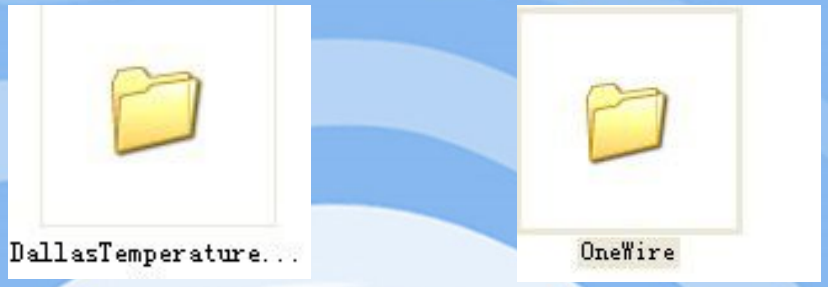
- DS18x20系列数字温度传感器主要有DS18S20和DS18B20(DS18S20只有9位一种工作模式，分辨率只到0.5摄氏度，DS18B20有9、10、11、12位四种工作可编程控制的模式，分辨率最高为0.0625摄氏度。)，都是由美国Dallas半导体公司(现在改名叫Maxim)生产的。
- 这个系列最大的特点就是采用了Maxim的专利技术1-Wire。顾名思义，1-Wire就是采用单一信号线，但可像I2C，SPI一样，同时传输时钟（clock）又传输数据（data），而且数据传输是双向的。1-Wire 使用较低的数据传输速率，通常是用来沟通小型device，如数位温度计。通过1-Wire技术可以在单一信号线的基础上构成传感器网络，Maxim起名”MicroLan”。





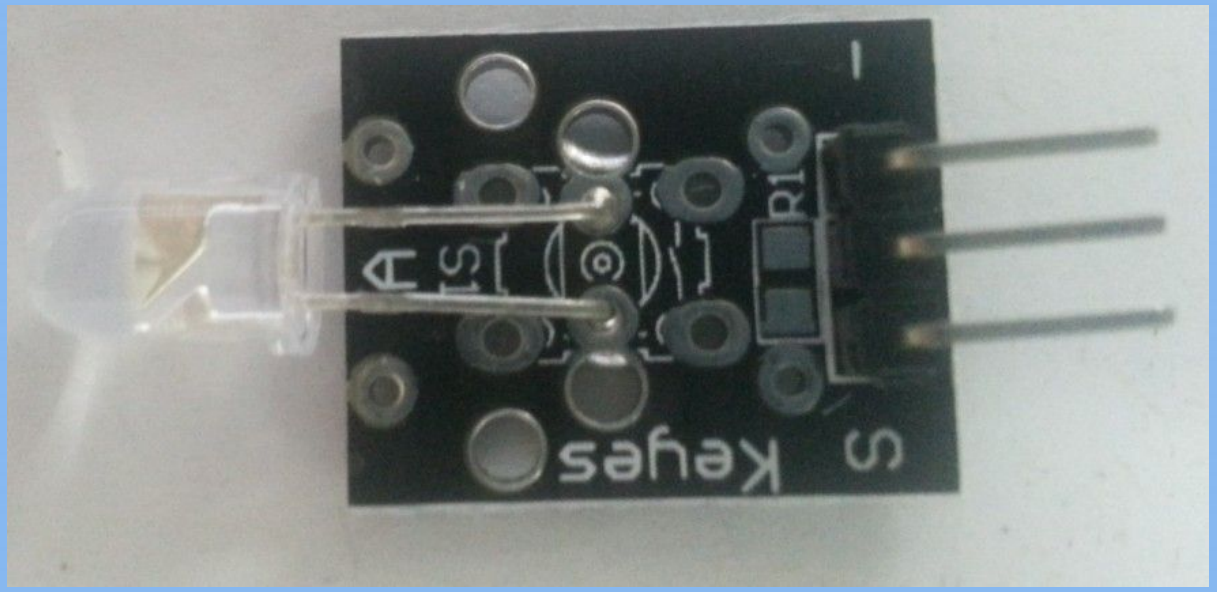
编程原理

- 厂家已经提供可供调用的库函数，只需要把库函数拷贝下来，放到arduino\libraries下面。这两个库函数分别是DallasTemperature和onewire两个文件夹。
- 本库函数有两个版本，一个支持arduino0023或以下版本编译通过，另一个支持1.01或更高版本。



```
#include <OneWire.h>
#include <DallasTemperature.h>
#define ONE_WIRE_BUS 2   // 数据线接数据口2
OneWire oneWire(ONE_WIRE_BUS);  //实例化一个对象
DallasTemperature sensors(&oneWire);  //实例化一个对象
void setup(void)
{  Serial.begin(9600);  //串口波特率9600
  Serial.println("Dallas Temperature IC Control Library Demo");
  sensors.begin(); //调用该对象的方法，启动传感器初始化
}
void loop(void)
{
  Serial.print(“Requesting temperatures...”);
  sensors.requestTemperatures(); // 发送命令去读取温度
  Serial.println("DONE");
  Serial.print(“Temperature for the device 1 (index 0) is: ”);
  Serial.println(sensors.getTempCByIndex(0));  //显示索引号为0的传感器温度。（可在总线上接多个传感器，根据索引号地址来区分）
}
```

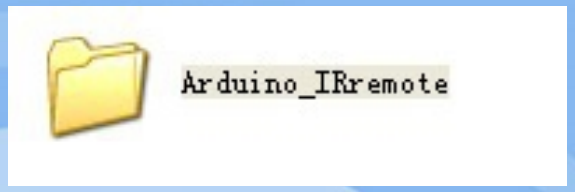
32、红外发射



- 使用单片机产生38K的调制信号来发射。业界通用的标准，只是码的含义不同。同一个码，SONY和JVC的厂家的定义就不一样。

编程原理

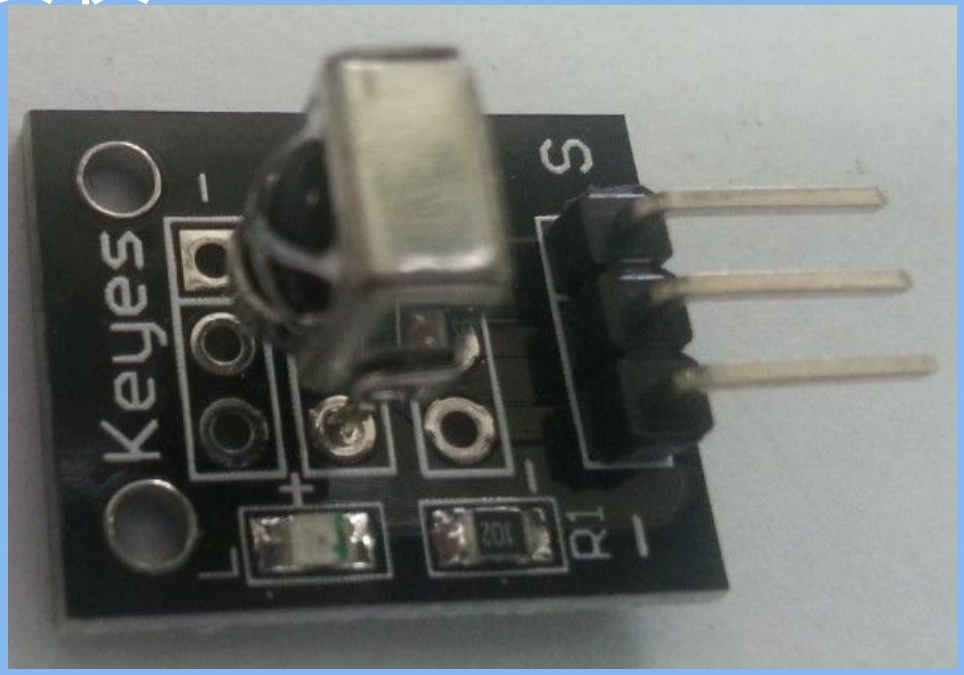
- 经提供可供调用的库函数，只需要把库函数拷贝下来，放到arduino\libraries下面。该函数的文件夹名是Arduino_IRremote。然后调用IRremote.h，实例化一个对象IRsend即可使用其方法。使用Arduino1.04以上版本。



//将红外S端接数字第3脚（PWM）

```
#include <IRremote.h>
IRsend irsend; //实例化一个对象
void setup()
{
  Serial.begin(9600);
}
void loop()
{  if (Serial.read() != -1)
    {  for (int i = 0; i < 3; i++)
       {   irsend.sendSony(0xa90, 12); // Sony TV power code
           delay(40);
        }
    }
}
```

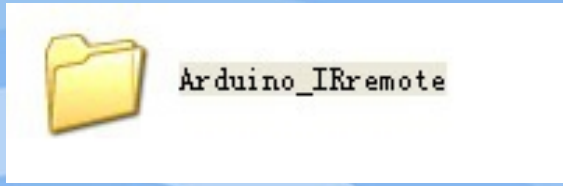
33、红外接收



- 红外接收头，和家用遥控器使用的红外接收头一致，接收38K的调制信号。通过单片机解码。

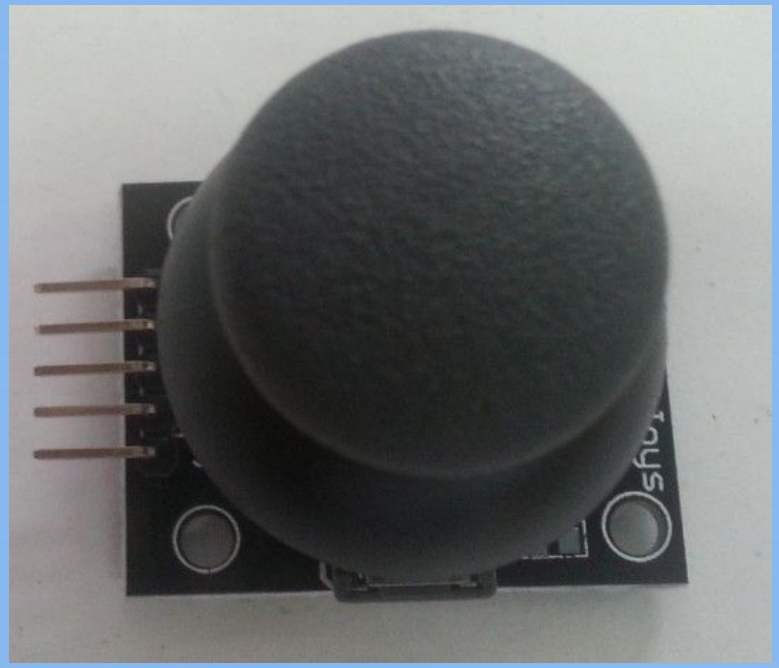
编程原理

- 经提供可供调用的库函数，只需要把库函数拷贝下来，放到arduino\libraries下面。该函数的文件夹名是Arduino_IRremote。然后调用IRremote.h，实例化一个对象IRrecv即可使用其方法。使用Arduino1.04以上版本。



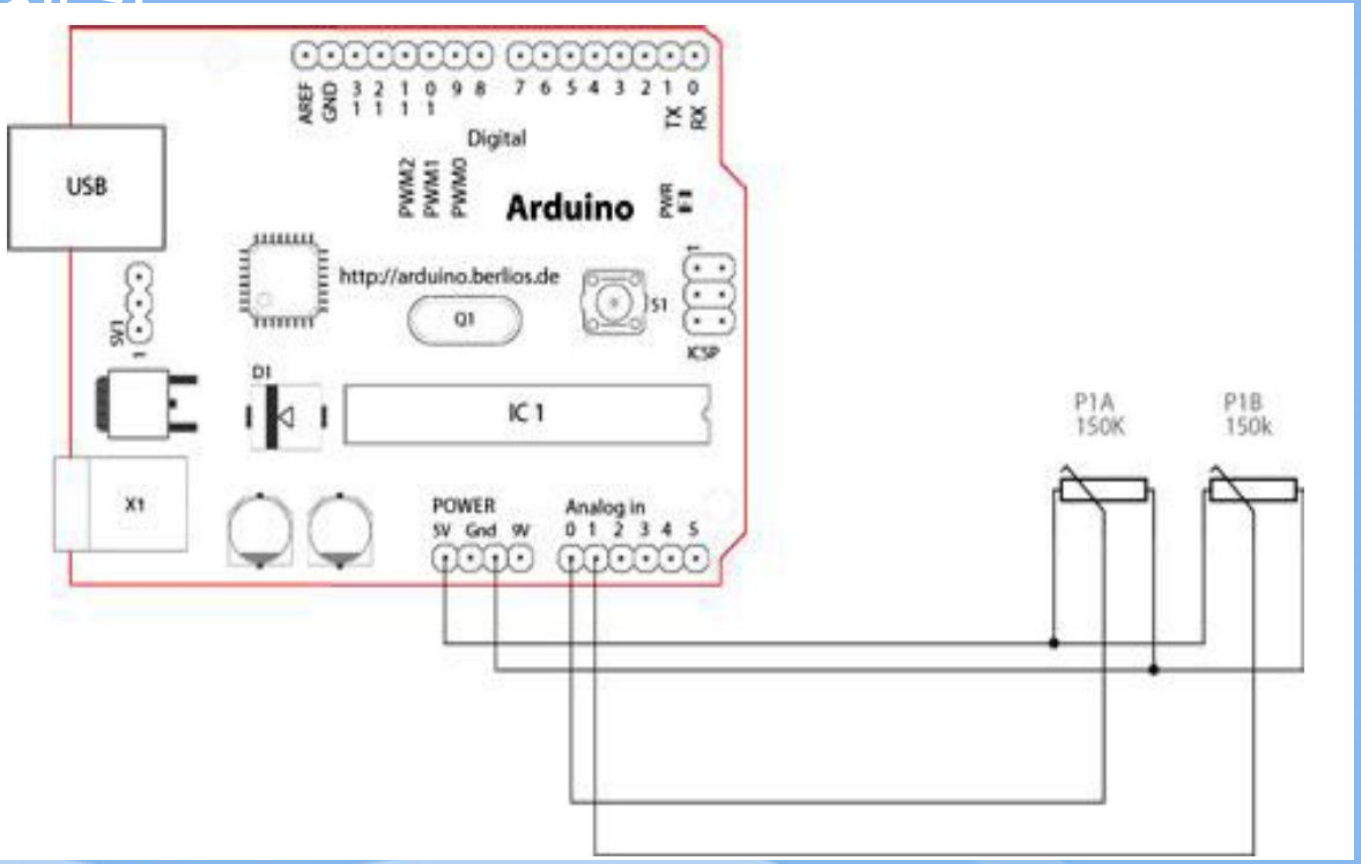
```
#include <IRremote.h>
int RECV_PIN = 11; //定义红外接收器的引脚为11
IRrecv irrecv(RECV_PIN); //实例化一个对象，并使用第11脚进行接收
decode_results results;
void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn(); // 初始化红外接收器
}
void loop()
{
  if (irrecv.decode(&results))
  {    Serial.println(results.value, HEX); //以16进制换行输出接收代码
   Serial.println(); //为了便于观看输出结果增加一个空行
 irrecv.resume(); // 接收下一个值
  }
  }
```

34、Joystick PS2 摇杆



- 具有(X,Y)2轴模拟输出，(Z)1路按钮数字输出。配合Arduino传感器扩展板可以制作遥控器等互动作品。

接线图



VRX接模拟0，VRY接模拟1，SW是按钮，接模拟口2

编程原理

- 该操纵杆本身是两个电位器（可调电阻）加一个按钮开关组成的。拨动电位器，使得阻值发生变化，从而输出电压改变。输出的是模拟量接模拟口。按钮输出的值由于电压低于2.5V，无法输出高电平，按下后，输出0V，因此也用模拟口来读取。

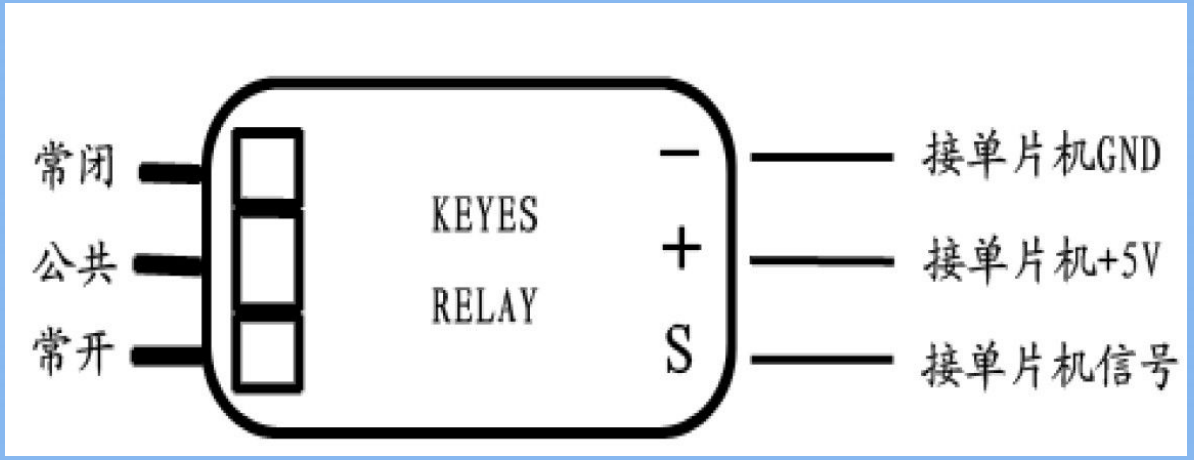
```
int Xaxis=A0;   //定义X轴由模拟0端口读取
int Yaxis=A1;  //定义Y轴由模拟1端口读取
int Zsw=A2;     //定义Z按钮由模拟2端口读取（因为开关最大值低于2.5V）
int value = 0;  //该变量读取模拟口的值
void setup()
{
    Serial.begin(9600);
 }
void loop()
 { value = analogRead(Xaxis);  //读取模拟端口0
   Serial.print("X:");
   Serial.print(value, DEC);
   value = analogRead(Yaxis); //读取模拟端口1
   Serial.print(" | Y:");
   Serial.print(value, DEC);
   value = analogRead(Zsw); //读取模拟端口2
   Serial.print(" | Z: ");
   Serial.println(value, DEC);
   delay(100);
 }
```

35、继电器



- 继电器适合驱动大功率的电器，如电灯、电风扇甚至空调。单片机接继电器可以实现弱电控制强电。

编程原理



- 如上图所示，如果接一个灯泡要点亮的话，将火线接在公共和常开端（没有控制信号时是断开的），然后S接单片机接数字口3，将按钮接数字口4.按下按钮则继电器工作，电灯亮起。

```
int relayPin=3;
int buttonPin=4;
void setup()
 { pinMode(relayPin,OUTPUT); //定义端口属性为输出；
   pinMode(buttonPin,INPUT);//定义端口属性为输入；
 }
void loop()
 { if(digitalRead(buttonPin)==HIGH)
    digitalWrite(relay,HIGH); //继电器导通；
  else
   digitalWrite(relay,LOW); //继电器开关断开；
   delay(1000);
}
```