# 第三章 按键控制开关灯

我们要结合数字输入功能，制作一个可控制的LED。实现按住按键时，点亮LED，放开按键后，熄灭LED的效果。

实验中我们将用到按键或者按键模块，常见的有2脚按键和4脚按键，其内部如图2-21所示。当按下按键时，就会接通按键两端，放开时，两端会再次断开。

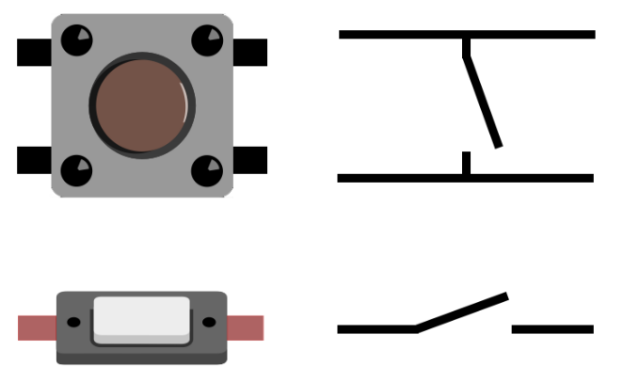


图2- 21 按键

* 实验所需材料：

Arduino 2560、按键模块、LED模块

* **连接示意图**

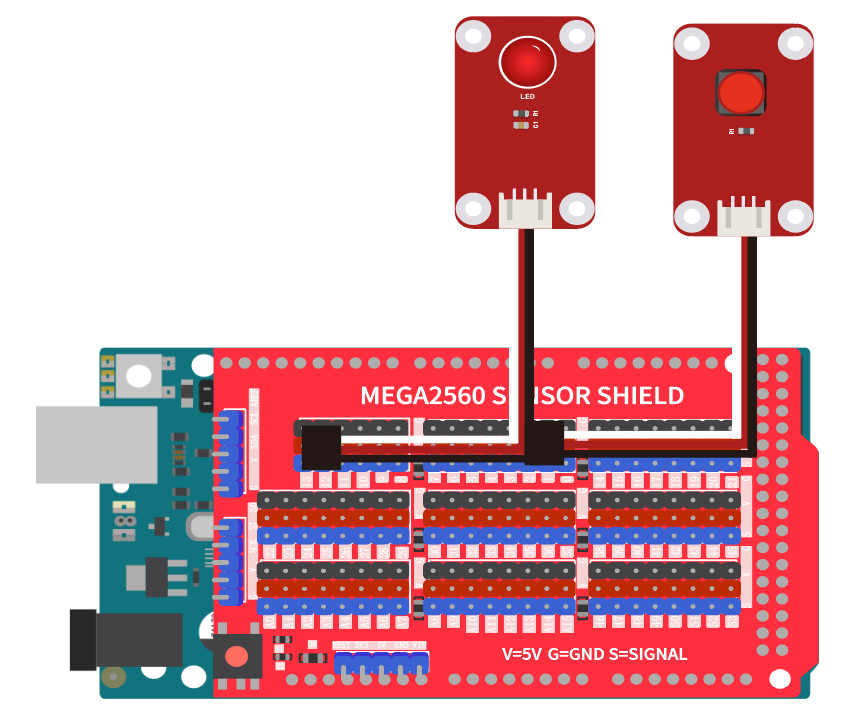


图2- 22 按键控制LED实验连接示意图

* 实现方法分析：

当未按下按键时，2号引脚检测到的输入电压为低电平；当按下按键时，会导通2号引脚和VCC，此时2号引脚检测到的输入电压为高电平。通过判断按键是否被按下，来控制LED的亮灭。

你可以在IDE的文件菜单>示例>02.Digital>Button中找到以下程序：

|  |
| --- |
| /\*  Button  通过2号引脚连接的按键，控制13号引脚连接的LED  备注：大多数Arduino的13号引脚上都连接了名为L的LED.  created 2005  by DojoDave <http://www.0j0.org>  modified 30 Aug 2011  by Tom Igoe  This example code is in the public domain.  http://www.arduino.cc/en/Tutorial/Button  \*/  // 设置各引脚别名  const int buttonPin = 2; // 连接按键的引脚  const int ledPin = 13; // 连接LED的引脚  // 变量定义  int buttonState = 0; // 存储按键状态的变量  void setup() {  // 初始化LED引脚为输出状态  pinMode(ledPin, OUTPUT);  // 初始化按键引脚为输入状态  pinMode(buttonPin, INPUT);  }  void loop(){  // 读取按键状态并存储在变量中  buttonState = digitalRead(buttonPin);  // 检查按键是否被按下  // 如果按键按下，那buttonState应该为高电平  if (buttonState == HIGH) {  // 点亮LED  digitalWrite(ledPin, HIGH);  }  else {  // 熄灭LED  digitalWrite(ledPin, LOW);  }  } |

下载并运行程序，按下按键，你会观察到LED会被点亮，松开按键，LED又会熄灭。

新的按键控制方式

接下来我们要对控制程序做一个升级，完成一个新的控制效果。按一下按键，点亮LED；再按一下按键，熄灭LED。

连线图同图2-22。

程序代码如下：

|  |
| --- |
| /\*  OpenJumper Button Module  按键控制LED  www.openjumper.com  \*/  int buttonPin = 2;  int ledPin = 13;  boolean ledState=false; // 记录LED状态  boolean buttonState=true; // 记录按键状态  void setup()  {  // 初始化I/O口  pinMode(buttonPin, INPUT\_PULLUP);  pinMode(ledPin,OUTPUT);  }  void loop()  {  // 等待按键按下  while(digitalRead(buttonPin)==HIGH){}  // 当按键按下时，点亮或熄灭LED  if(ledState==true)  {  digitalWrite(ledPin,LOW);  ledState=!ledState;  }  else  {  digitalWrite(ledPin,HIGH);  ledState=!ledState;  }  delay(500);  } |

下载该程序，每按一下按键，LED状态都会改变。

以上程序中，使用了

|  |
| --- |
| while(digitalRead(buttonPin)==HIGH){} |

因为在初始化时，我们已经将buttonPin引脚设为了输入上拉状态。如果没有按下按键，使用digitalRead(buttonPin) 读出的值始终为高电平，这个循环也将一直运行；当你按下按键后，digitalRead(buttonPin) 读出了低电平，while循环的判断条件为假，程序会退出这个循环，并开始运行此后的语句。这样我们就实现了一个等待用户按下按键的效果。

程序末尾有一个delay(500) 的延时，它在这里是极其重要的。你可以尝试删去这个延时操作，再下载程序到Arduino。你会发现按键经常出现控制失灵的情况。这是因为程序运行的非常快，没有了延时操作，你按下按键到放开按键的间隔时间虽然极短，但loop中的语句可能已经运行了很多次，很难确定你放开按键时正在运行的loop() 循环是点亮还是熄灭LED。正是这样的原因，程序变得不那么好用了。

上面程序中，我们使用延时操作来使两次按键间产生一定的间隔时间，在间隔时间内Arduino会忽略按键按下情况，从而达到区分两次按键的目的。