Arduino 入门版使用教程

DFRduino Starter kit User Manual

DFROBOT
DRIVE THE FUTURE

版本号:V 0.22

最后修订日：2010 09 10

目录

介绍

什么是 Arduino 基础套装？

Arduino 基础套装是精心为初学者设计的一款学习工具。它可以带您走进丰富多彩的电子世界，让您体验到电子科技无穷的乐趣。在整个实验过程中无须焊接，直接在面包板上插拔元件即可，非常适合学习。另外，本品还附带了十节实验课程。这十节课程的编排完全是从初学者的角度考虑，每一节实验都配有图文结合的实验说明文档和非常有趣的例子程序。而且每一节实验除了文档上讲的方法外，还有很大可供学习者发挥的空间。Arduino 基础套装可以说是一款超值的学习工具，实验盒里宝贝多多。

元件清单

- Arduino 328控制板 1个
- 原形开发扩展板 1个
- 面包板 1个
- LED 灯 红黄绿 各2个
- 蜂鸣器 1个
- 按键开关 4个
- 数码管 1个
- 倾斜开关 1个
- 光敏电阻 1个
- 红外接收管 1个
- 电阻 220 欧、1K、10K 各 3 个
- LM35 温度传感器 1 个
- USB 线 1条
- 多彩面包线 20条
- 6节5号电池盒 1个
- mini 遥控器 1个
- Arduino 基础套装教程 1本
- 教程及其开发软件光盘 1张

Arduino 介绍篇

概 述

什么是 Arduino？

Arduino 是一块基于开放原始代码的 Simple i/o 平台，并且具有开发语言和开发环境都很简单、易理解的特点。让您可以快速使用 Arduino 做出有趣的东西。Arduino 可以配合一些电子元件使用例如：本产品实验盒中的 LED 灯、蜂鸣器、按键、光敏电阻等等。Arduino 开发环境界面基于开放原始码原则，可以让您免费下载使用开发出更多令人惊奇的互动作品。

特色描述

- 开放原始码的电路图设计，开发界面免费下载，也可依需求自己修改!!
- 下载程序简单、方便。
- 可简单地与传感器、各式各样的电子元件连接（如：LED 灯、蜂鸣器、按键、光敏电阻等等），做出各种各样有趣的东西。
- 使用高速的微处理控制器(ATMEGA328)。
- 开发语言和环境都非常的简单、易理解，非常适合初学者学习。

性能描述

- Digital I/O 数字输入/输出端共 0~13。
- Analog I/O 模拟输入/输出端共 0~5。
- 支持 ISP 下载功能。
- 输入电压：接上 USB 时无须外部供电或外部5V~9V 直流电压输入。
- 输出电压：5V 直流电压输出和3.3V 直流电压输出和外部电源输入。
- 采用 Atmel Atmega328微处理控制器。
- Arduino 大小尺寸：宽70mm X 高54mm。

Arduino C 语言介绍

Arduino 语言是建立在 C/C++基础上的，其实也就是基础的 C 语言，Arduino 语言只不过把相关的一些参数设置都函数化，不用我们去了解他的底层，让我们不了解 AVR 单片机（微控制器）的朋友也能轻松上手。那么这里我就简单的注释一下 Arduino 语言。

关键字：

- if
- if...else
- for
- switch case
- while
- do... while
- break
- continue
- return
- goto

语法符号：

- ;
- {}
- //
- /* */

运算符：

- =
- +
- -
- *
- /
- %
- ==
- !=

- <
- >
- <=
- >=
- &&
- ||
- !
- ++
- --
- +=
- -=
- *=
- /=

数据类型：

- boolean 布尔类型
- char 字符类型
- byte 字节类型
- int 整数类型
- unsigned int 无符号整型
- long 长整型
- unsigned long 无符号长整型
- float 实数类型
- double
- string
- array
- void

常量：

- HIGH | LOW 表示数字 IO 口的电平，HIGH 表示高电平（1），LOW 表示低电平（0）。
- INPUT | OUTPUT 表示数字 IO 口的方向，INPUT 表示输入（高阻态），OUTPUT 表示输出（AVR 能提供 5V 电压 40mA 电流）。

- true | false true 表示真（1），false 表示假（0）。

以上为基础 c 语言的关键字和符号，大家可以了解，具体使用可以结合实验的程序。

结构

- void setup() 初始化变量，管脚模式，调用库函数等
- void loop() 连续执行函数内的语句

功能

数字 I/O

- pinMode(pin, mode) 数字 IO 口输入输出模式定义函数，pin 表示为 0～13， mode 表示为 INPUT 或 OUTPUT。
- digitalWrite(pin, value) 数字 IO 口输出电平定义函数，pin 表示为 0～13，value 表示为 HIGH 或 LOW。比如定义 HIGH 可以驱动 LED。
- int digitalRead(pin) 数字 IO 口读输入电平函数，pin 表示为 0～13，value 表示为 HIGH 或 LOW。比如可以读数字传感器。

模拟 I/O

- int analogRead(pin) 模拟 IO 口读函数 ,pin 表示为 0～5(Arduino Diecimila 为 0～5，Arduino nano 为 0～7)。比如可以读模拟传感器（10 位 AD，0～5V 表示为 0～1023)。
- analogWrite(pin, value) - PWM 数字 IO 口 PWM 输出函数，Arduino 数字 IO 口标注了 PWM 的 IO 口可使用该函数，pin 表示 3, 5, 6, 9, 10, 11，value 表示为 0～255。比如可用于电机 PWM 调速或音乐播放。

时间函数

- delay(ms) 延时函数（单位 ms)。
- delayMicroseconds(us) 延时函数（单位 us)。

数学函数

- min(x, y) 求最小值
- max(x, y) 求最大值
- abs(x) 计算绝对值
- constrain(x, a, b) 约束函数，下限 a，上限 b，x 必须在 ab 之间才能返回。
- map(value, fromLow, fromHigh, toLow, toHigh) 约束函数，value 必须在 fromLow 与 toLow 之间和 fromHigh 与 toHigh 之间。
- pow(base, exponent) 开方函数，base 的 exponent 次方。
- sq(x) 平方
- sqrt(x) 开根号

Arduino 使用介绍

有了以上作为基础，下面我们就要开始实际操作了。下面将分步骤介绍：

1、准备好你的 Arduino 板

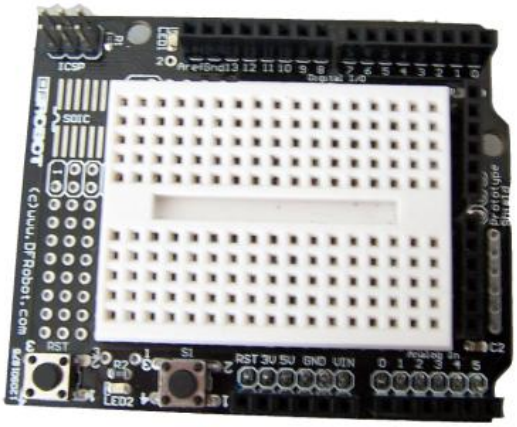
·首先从实验盒中拿出 Prototype shield 扩展板如图：

图表 1

·接着从实验盒拿出面包板如图：

图表 2

将面包板反过来大家可以看到，面包板的后面带有双面胶，将双面胶的白色部分揭下来，然后把面包板粘贴到 Prototype shield 扩展板上，如下图：

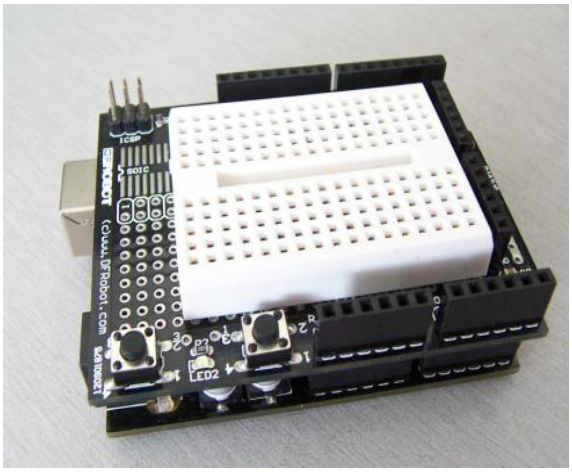


图表 3

·然后从实验盒中拿出 Arduino 328 控制板如图：

图表 4

把贴有面包板的 Prototype shield 扩展板插在 Arduino 328 控制板上如图：



图表 5

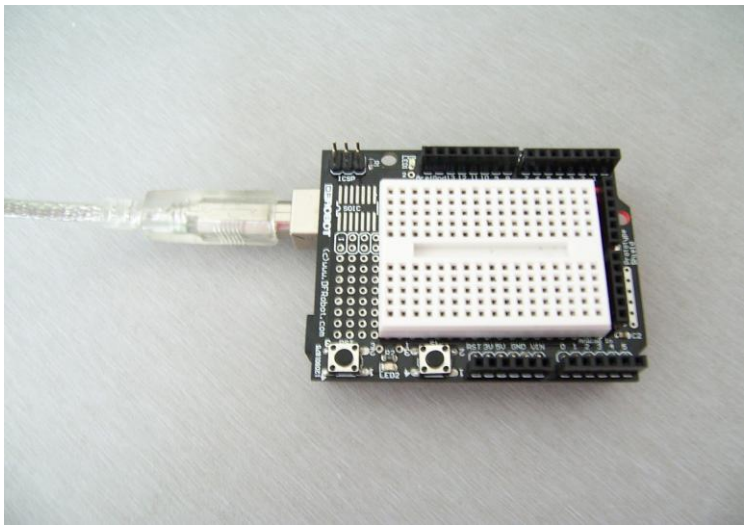
这样板子就连接好了。

2、下载 Arduino 开发环境

由于本产品光盘中已经带有 Arduino0018 软件压缩包，所以就不用下载了。在光盘中可以找到 Arduino0018 软件压缩包，解压即可。

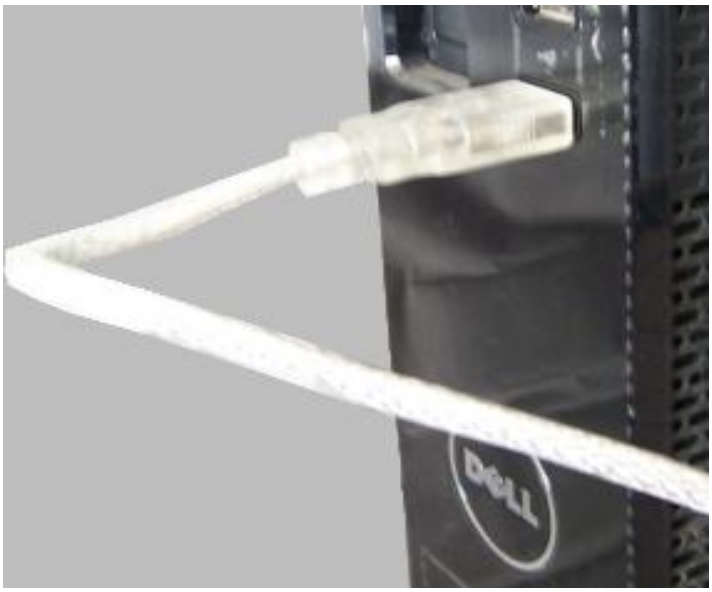
3、安装 USB 驱动

·首先连接下载程序用的下载线。首先从实验盒中拿出下载线如下图：
将数据线的圆口一端插在 Arduino328 板子上如图：



图表 6

·将数据线的扁口一端插在电脑的 USB 接口上，如下图所示：

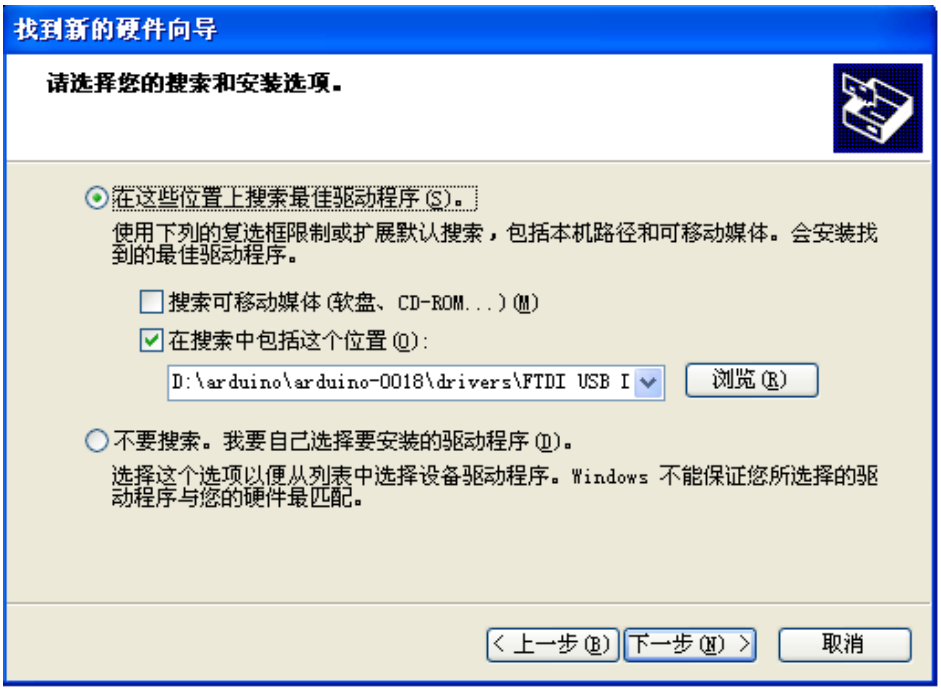


图表 7

插好后，Arduino328 控制板上的电源指示灯会被点亮，电脑上会出现一个对话框如图：

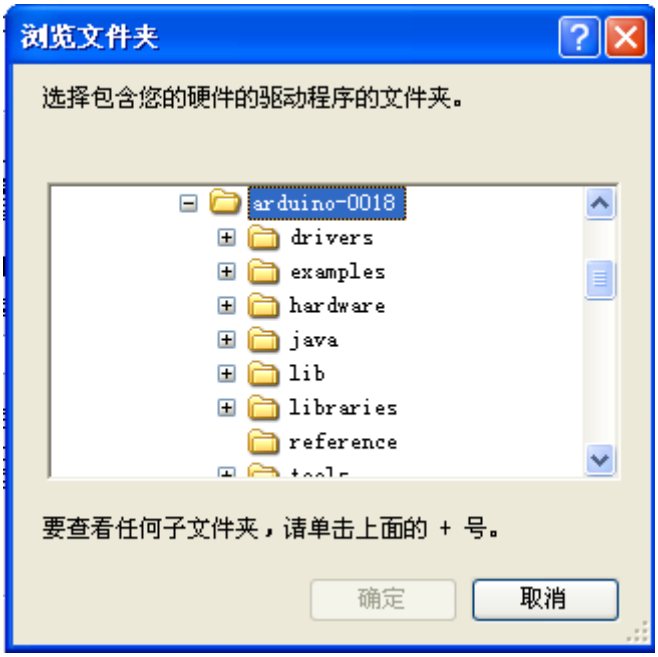
图表 8

·选择从列表或指定位置安装，点击下一步，出现如下图：



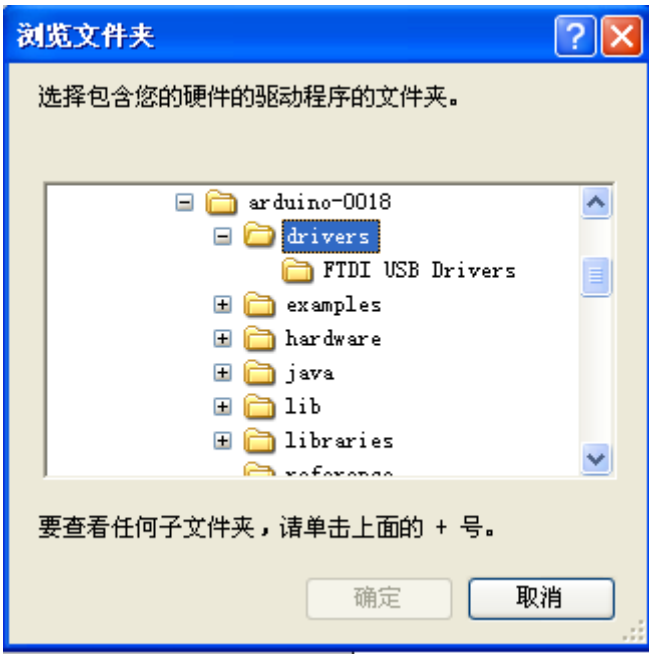
图表 9

然后点击浏览，在出现的浏览文件夹对话框中点击光盘，在光盘下找到 arduino0018 文件夹，点击打开，会看见有 drivers 文件夹如下图所示：



图表 10

·点击 drivers 文件夹，会看到 FTDI USB Drivers 文件夹，如图：



图表 11

然后点击这个文件夹，接着点击确定，点击下一步，会出现如图对话框：

图表 12

·这时我们只要等待即可，稍后会出现如下图对话框：

图表 13

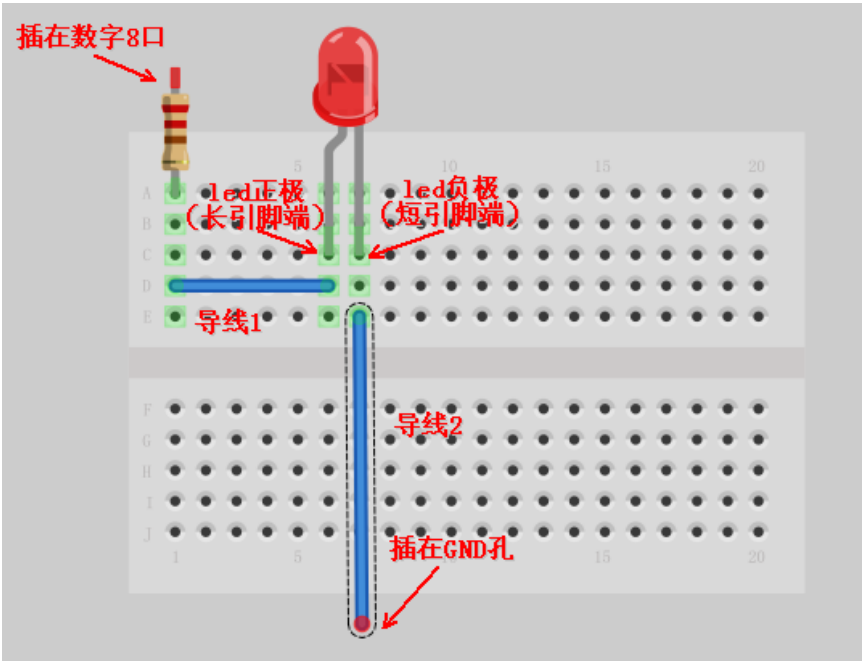
点击完成，这样驱动就安装好了，下次再将数据线插到电脑就不会出现安装驱动对话框了，插上数据线就可以下载程序了。

4、连接 LED 灯电路

·从实验盒中取出一个 220Ω的电阻，将它的一端插在数字 8 口上，电阻的另一端插在面包板上。再从实验盒中去取出一个发光二极管如下图：

图表 14

将发光二极管插在面包板上，插法如图：

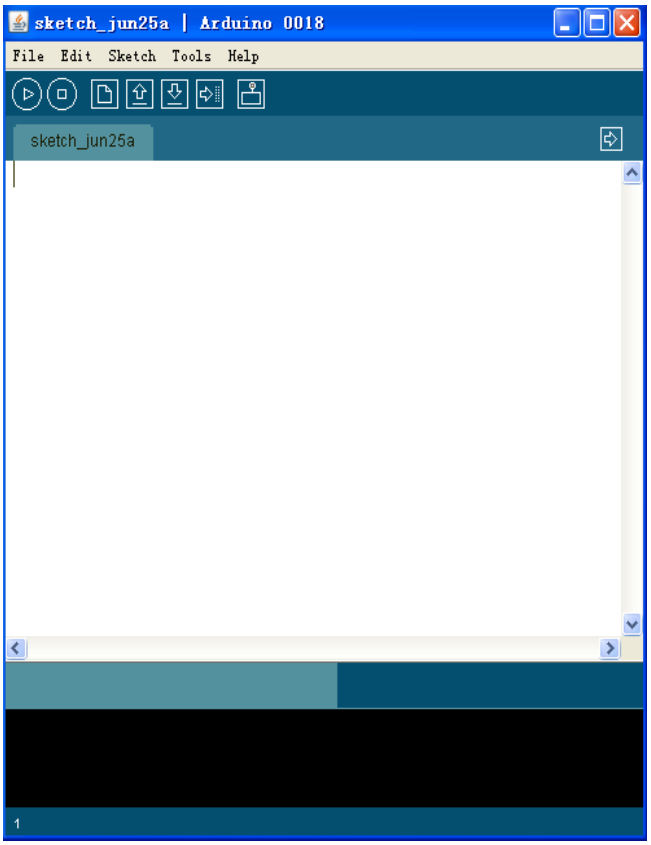


图表 15

·接着按照图上说明插上一根导线，这样电路就连接好了。

5、打开 arduino 开发环境

打开 arduino0018 文件夹，里面有一个标有 arduino.exe 图标，双击打开会出现如下界面：



图表 16

·Arduino 0018 开发编译环境很简洁，各个功能键功能描述如下：

图表 17

·打开软件后，我们就可以在窗口的空白处编写程序了。这里就不讲解怎么编程了，因为实验里会具体的介绍。点击叉关闭软件。我们直接打开 led 灯实验程序如图：

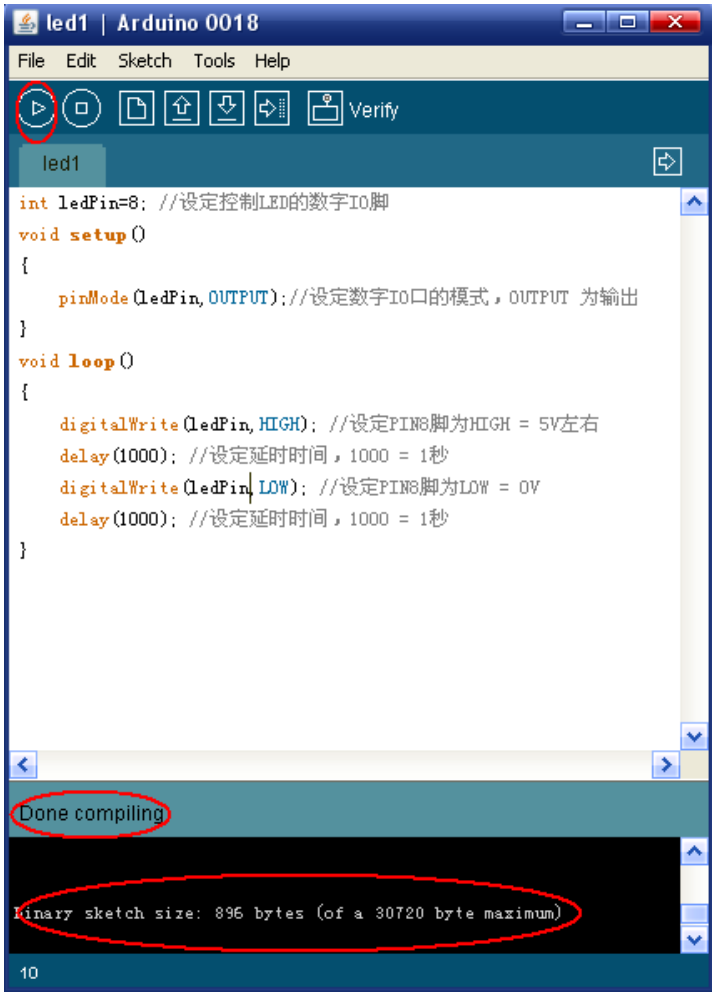
图表 18

7、编译程序

·点击编译按钮，这时编译按钮会变成黄色，下面出现英文 compiling.....，这表示软件正在对你所写的程序进行编译，如下图所示：

图表 19

等待一会，会看到编译按钮恢复原来的状态，下面出现 Done compiling，最下面一段文字说明编写的程序共有 896 字节数。这表明，程序编译成功，并且没有语法上的错误。如下图所示：



图表 20

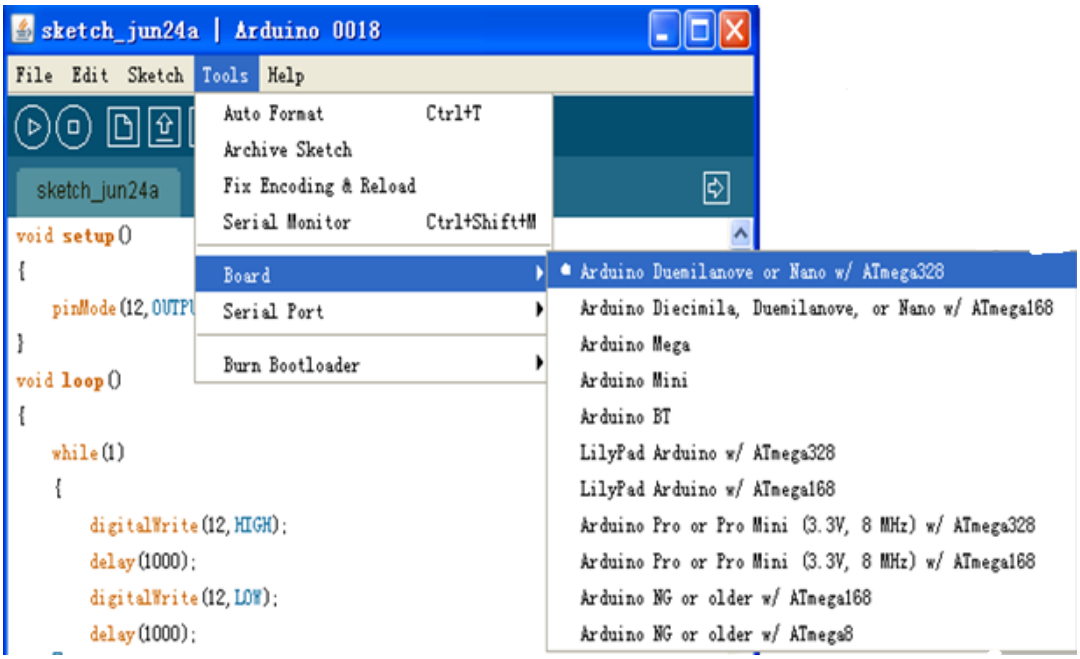
·下面看看程序有语法错误时会出现什么状态，将程序中 pinMode（ledPin,OUTPUT）后面的分号去掉，点击编译按钮，编译完成后会出现如下图所示状态：

图表 21

1 处告诉我们是因为在“}”附近缺少分号而出现的错误。2 处用文字告诉我们错误是出现在 void setup（ ）的一个“}”附近。3 处用黄颜色将“}”覆盖，表示错误就在这附近。从程序中看到错误确实在大括号附近，将分号添上后就会编译成功。以后编写程序出现错误时，就可以通过看下面信息栏里的提示调试程序。

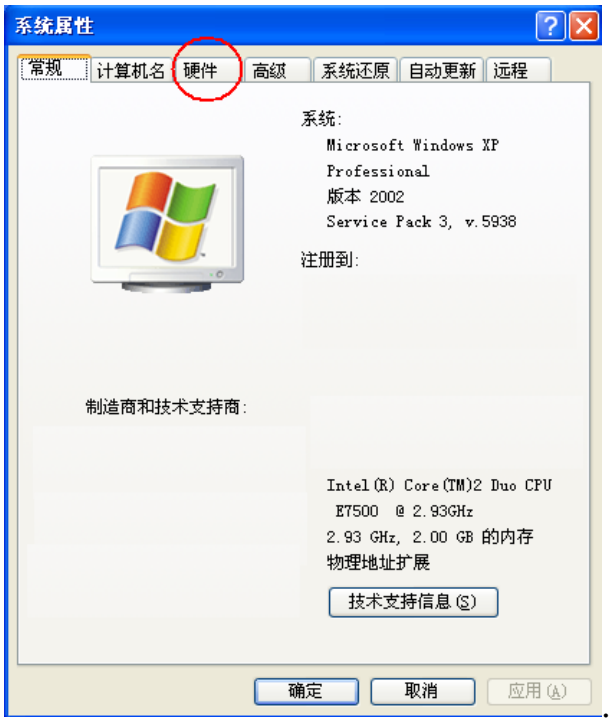
8、下载程序

·下载程序前先将板子型号和 com 口选好。点击 Tools->Board 选择开发板型号，如图：



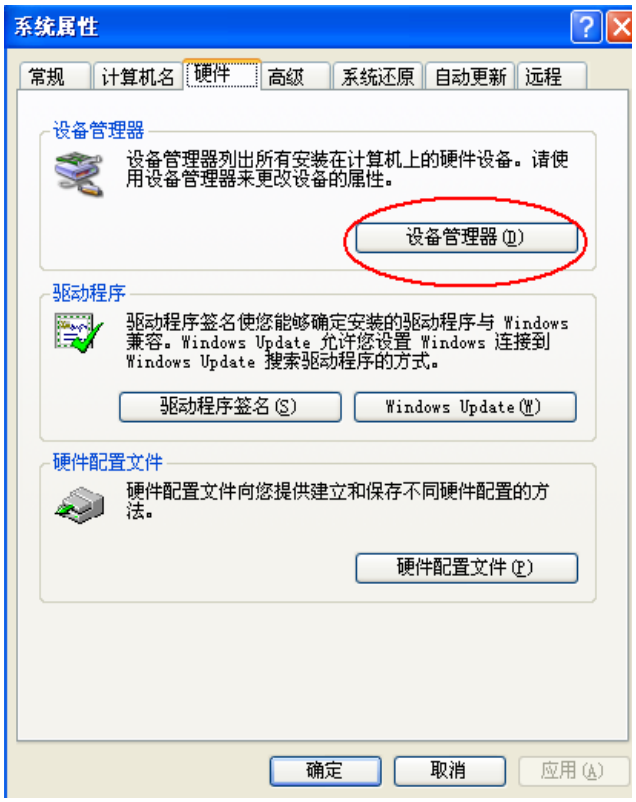
图表 22

·这里我们使用的是 Arduino 328 控制板，所以点击第一个即可。接下来选择串口，首先看一下我们的串口是 COM 几，右键点击我的电脑的图标，选择属性，会出现如下对话框：



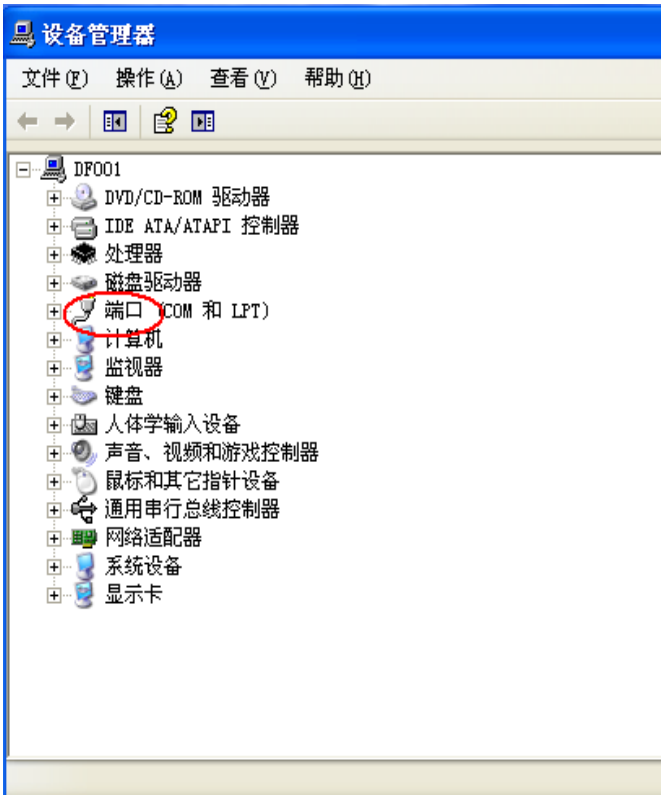
图表 23

·点击硬件，出现如图对话框：



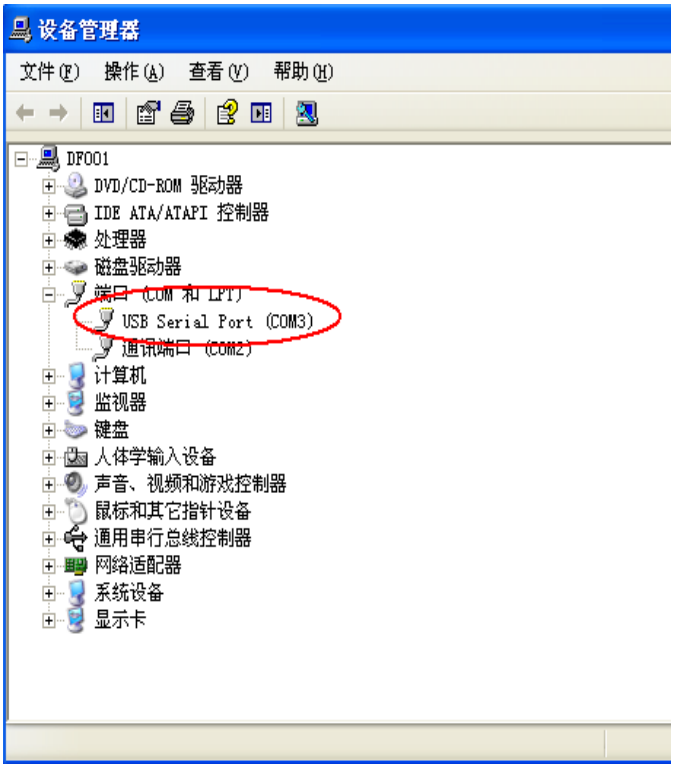
图表 24

·点击设备管理器，出现下图：



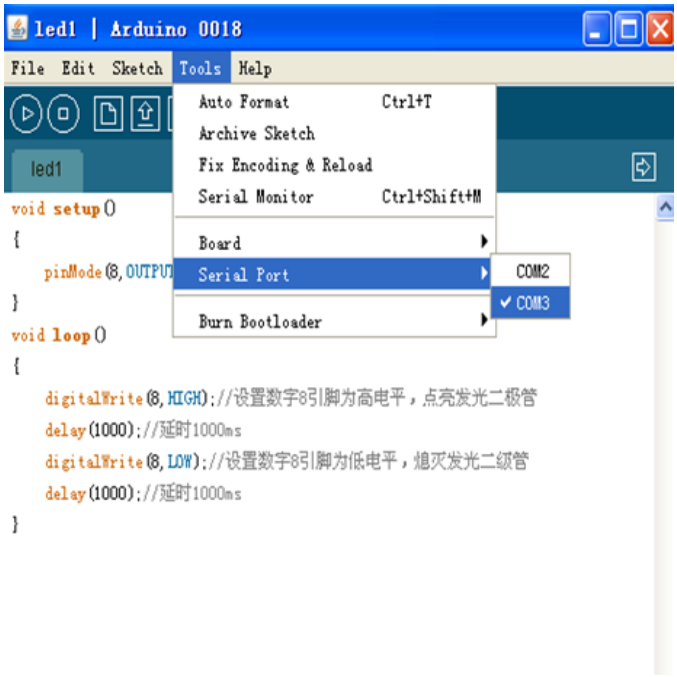
图表 25

·双击端口，出现下图：



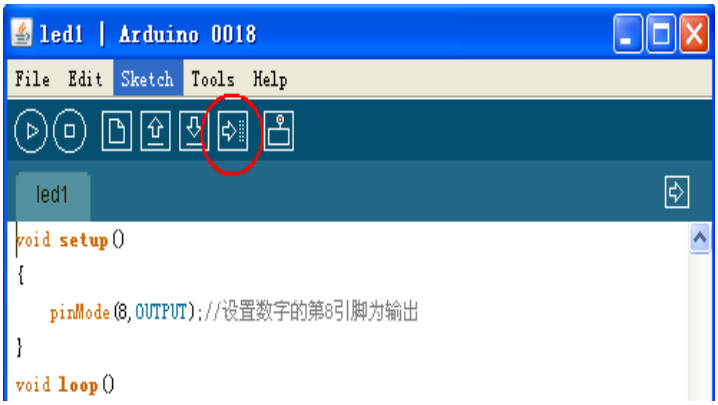
图表 26

·会看到有一个写着 USB Serial Port（COM3），COM3 这个就是我们的串口号。把这个号记住，关闭窗口，回到 arduino 软件窗口，点击 Serial Port，选择刚才记住的 COM 口号——COM3，如图：



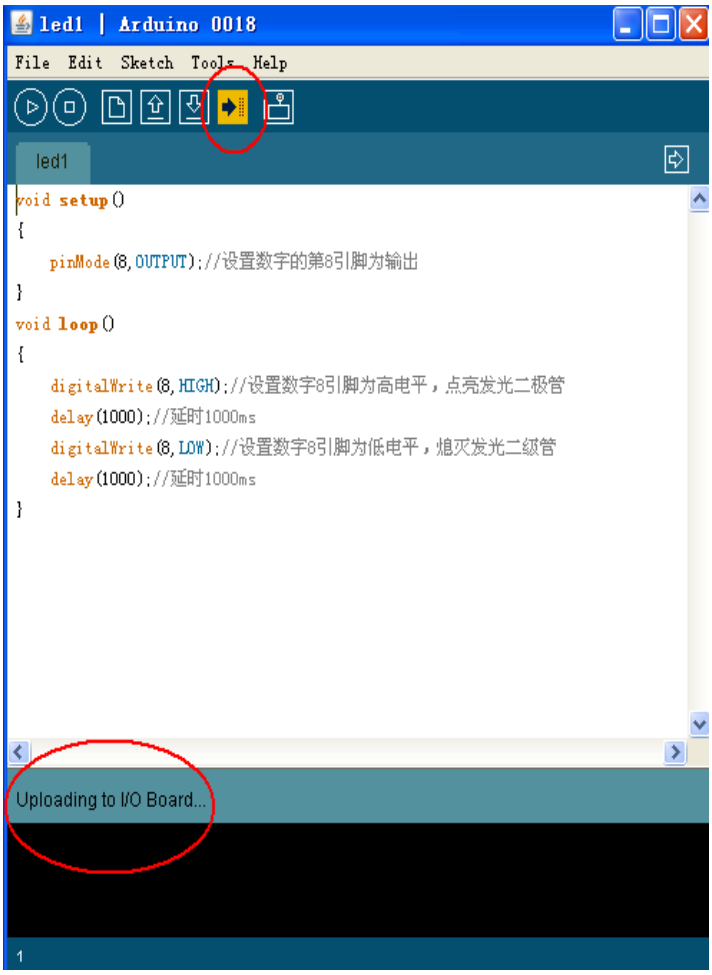
图表 27

·这样板子型号和 COM 口就选好了。接下来点击 arduino 软件上的下载按钮，如图：



图表 28

点击之后下载按钮变成橙色，软件下方出现 Uploading to I/O Board，同时板子上标有 TX 和 RX 的灯会亮，如图所示：



图表 29

程序下载完毕后，下载按钮恢复原来的颜色，下面出现 Done Uploading，如图：

图表 30

如果没有显示 Done Uploading，而是出现了红色的字，表示下载失败，可以检查一下 USB 线是否连接好、电源开关是否打开、COM 口是否选对等等。如果出现上图，表示程序下载成功了，如果你看到面包板上的 led 灯亮 1s、灭 1s 的在闪烁，恭喜你，你的 Arduino 板开始工作啦！

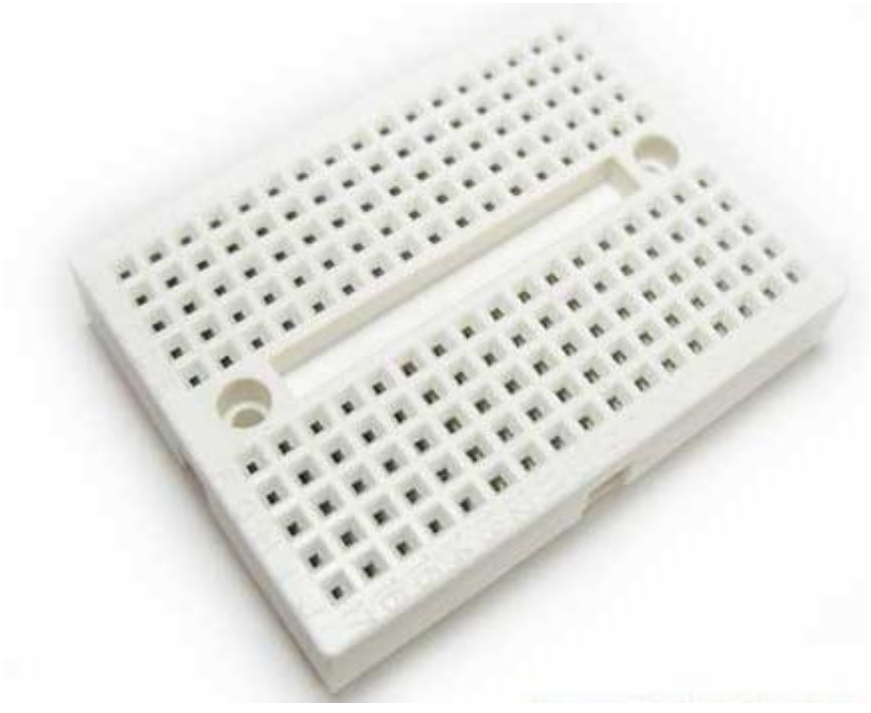
面包板使用介绍

1、面包板介绍

套件中包含的面包板具有170个插孔，此面包板可以配合 Arduino 各种型号的ProtoShield（原型扩展板），自带双面粘胶，可以粘贴到各种开发板、扩展板上，也可粘贴到各种轮式机器人或履带式机器人基板上实现个性化功能调试，体积小巧，仅有45mm×35mm（1.8"x1.4"）大小，是 Arduino 互动媒体爱好者、机器人发烧友、电子爱好者和电子工程师必备用品。

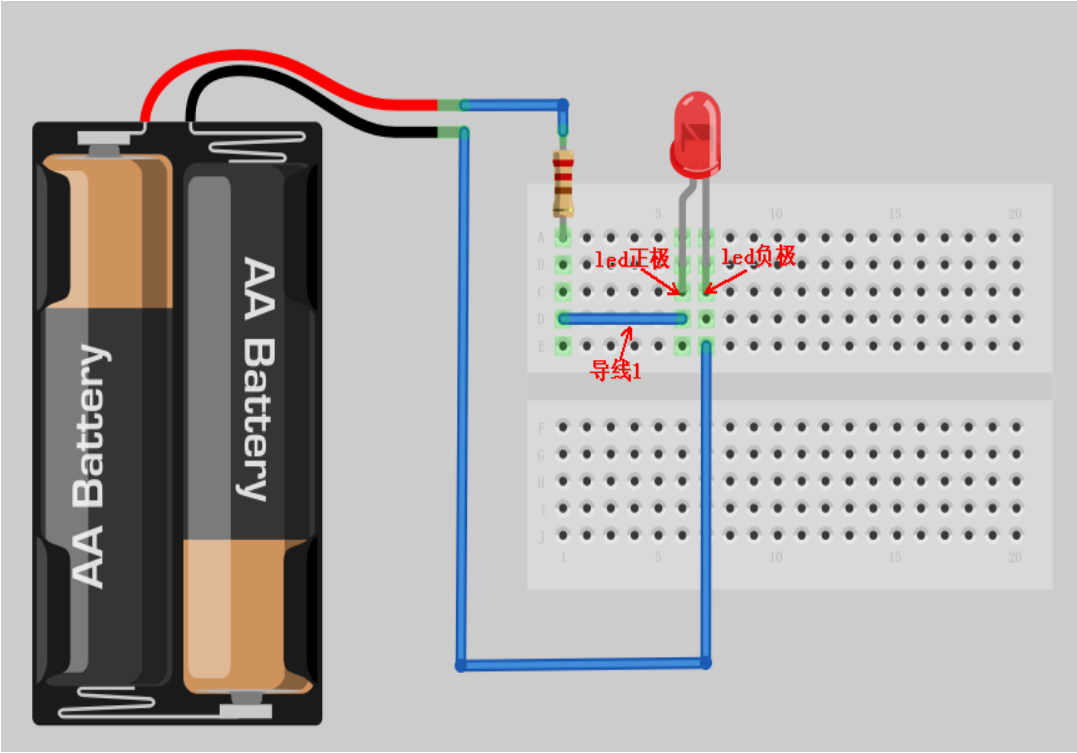
2、面包板的使用

面包板（集成电路实验板）是电路实验中一种常用的具有多孔插座的插件板，在进行电路实验时，可以根据电路连接要求，在相应孔内插入电子元器件的引脚以及导线等，使其与孔内弹性接触簧片接触，由此连接成所需的实验电路。下图为本迷你面包板的示意图。



图表 31

它一共具有170个插孔，10行17列。以中间的长槽为界分成上、下两部分，每一部分都是5行17列。从图中可以看到在每一部分中的每一列有5个插孔，这5个插孔的底部是一个金属簧片，因此插入这5个孔内的导线就被金属簧片连接在一起。例如下图：



图表 32

电阻和红色发光二极管是连通的，因为导线1的一端插在了电阻所在的列，这列的5个孔是连通的，所以导线和电阻是通的，导线的另一端插在了发光二级管正极所在的列，所以最终电阻和发光二极管是连通的。将电源正负极接好后发光二极管就会被点亮。每一部分的每一列的五个插孔都是通过金属簧片相连的，而每一行的17个孔是不通的，所以横排上的器件要连通的话，需用导线连接。

***注意**：插入面包板上孔内引脚或导线铜芯直径为 0.4～0.6mm，即比大头针的直径略微细一点。元器件引脚或导线头要沿面包板的板面垂直方向插入方孔，应能感觉到有轻微、均匀的摩擦阻力，在面包板倒置时，元器件应能被簧片夹住而不脱落。面包板应该在通风、干燥处存放，特别要避免被电池漏出的电解液所腐蚀。要保持面包板清洁，焊接过的元器件不要插在面包板上。*

实验篇

第一节 多彩 led 灯实验

一、发光二极管介绍

1、什么是发光二极管

发光二极管简称为 LED。由镓（Ga）与砷（AS）、磷（P）的化合物制成的二极管，当电子与空穴复合时能辐射出可见光，因而可以用来制成发光二极管，在电路及仪器中作为指示灯，或者组成文字或数字显示。磷砷化镓二极管发红光，磷化镓二极管发绿光，碳化硅二极管发黄光。

图 1.1 各种颜色的发光二极管

它是半导体二极管的一种，可以把电能转化成光能；常简写为 LED。发光二极管与普通二极管一样是由一个 PN 结组成，也具有单向导电性。当给发光二极管加上正向电压后，从 P 区注入到 N 区的空穴和由 N 区注入到 P 区的电子，在 PN 结附近数微米内分别与 N 区的电子和 P 区的空穴复合，产生自发辐射的荧光。不同的半导体材料中电子和空穴所处的能量状态不同。当电子和空穴复合时释放出的能量多少不同，释放出的能量越多，则发出的光的波长越短。常用的是发红光、绿光或黄光的二极管。

2、工作原理

发光二极管的反向击穿电压约 5 伏。它的正向伏安特性曲线很陡，使用时必须串联限流电阻以控制通过管子的电流。限流电阻 R 可用下式计算：

R=（E-VF）/I ；

式中 E 为电源电压，VF 为 LED 的正向压降，I 为 LED 的一般工作电流。发光二极管的工作电压一般为 1.5～2.0V，其工作电流一般为 10～20mA。所以在 5v 的数字逻辑电路中，可使用 220Ω 的电阻作为限流电阻。

3、Led 灯的内部结构与连线

发光二极管的两根引线中较长的一根为正极，应连接电源正极。有的发光二极管的两根引线一样长，但管壳上有一凸起的小舌，靠近小舌的引线是正极。如下图所示：

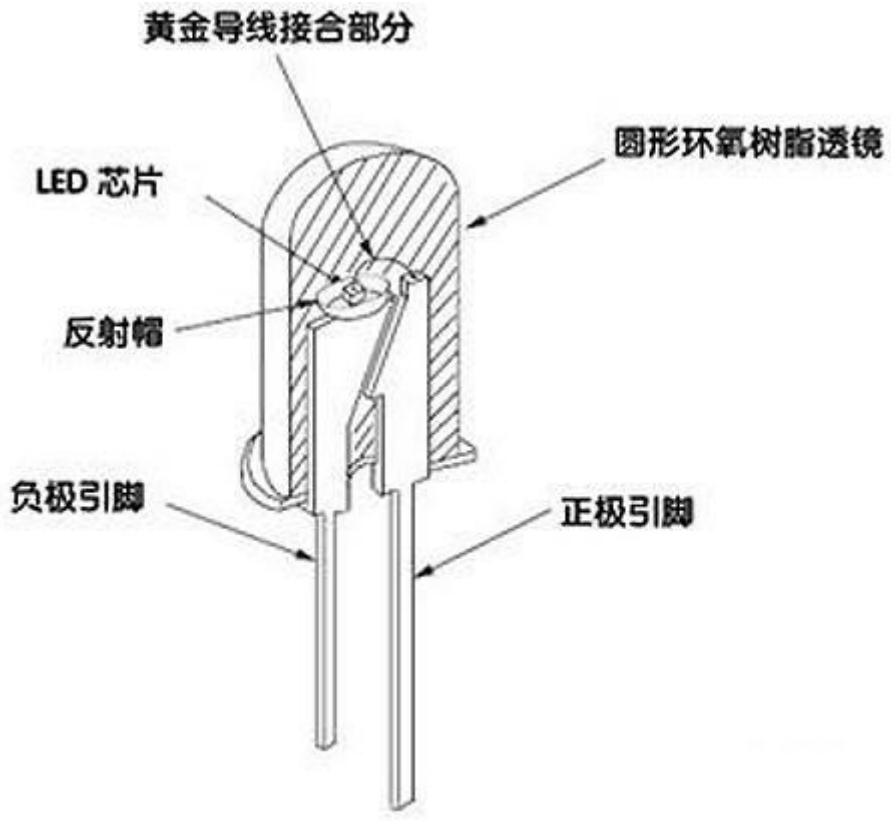


图 1.2 发光二极管内部结构图

Led 灯有两种连线方法 :当 led 灯的阳极通过限流电阻与板子上的数字 I/O 口相连，数字口输出高电平时，led 导通，发光二极管发出亮光；数字口输出低电平时，led 截

止，发光二极管熄灭。如图：

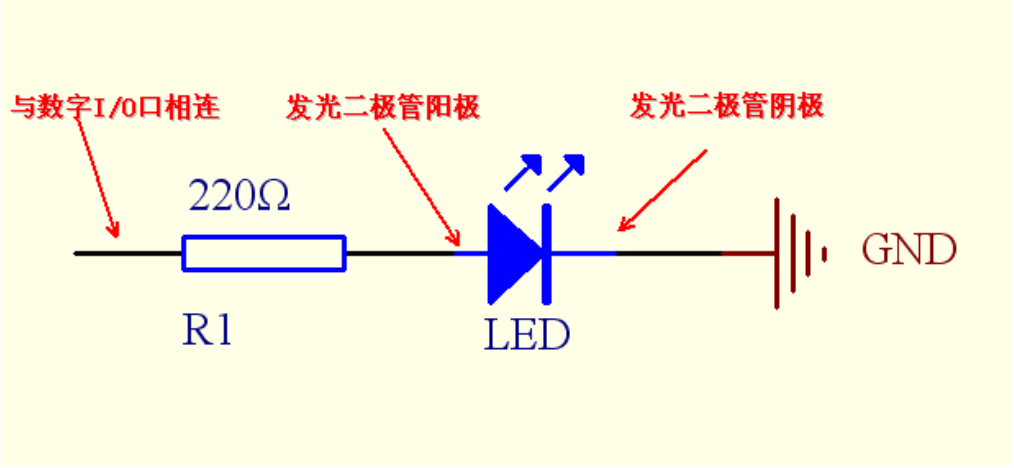


图 1.3　发光二极管接线方法 1

当 led 灯的阴极与板子上的数字 I/O 口相连时，数字口输出高电平，led 截止，发光二极管熄灭；数字口输出低电平，led 灯导通，发光二极管点亮。

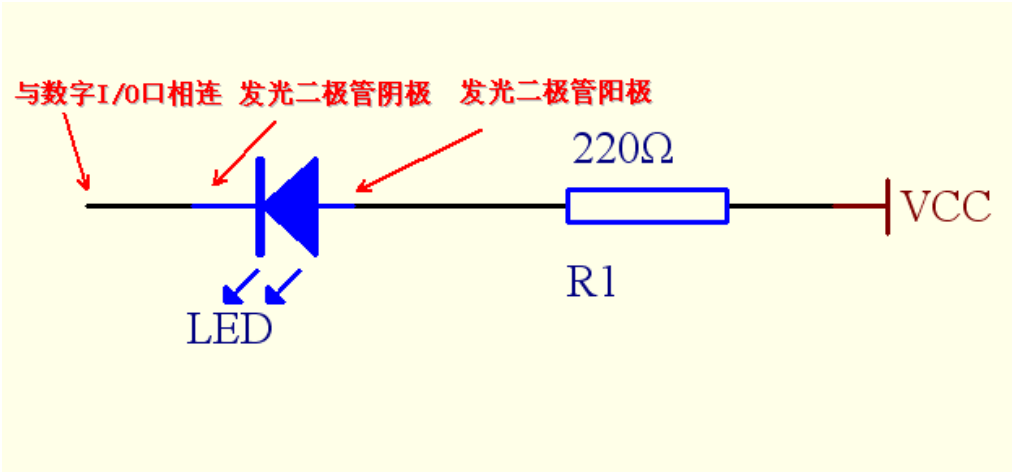


图 1.4　发光二极管接线方法 2

本实验选择了接线方法 1 连接发光二极管，将 220Ω 电阻的一端插在 Prototype Shield 扩展板上的第 8 个 digital I/O 口，电阻的另一端插在面包板上，电阻和发光二极管通过导线相连，发光二极管的负端插在面包板上与 GND 相连。具体连接如图：

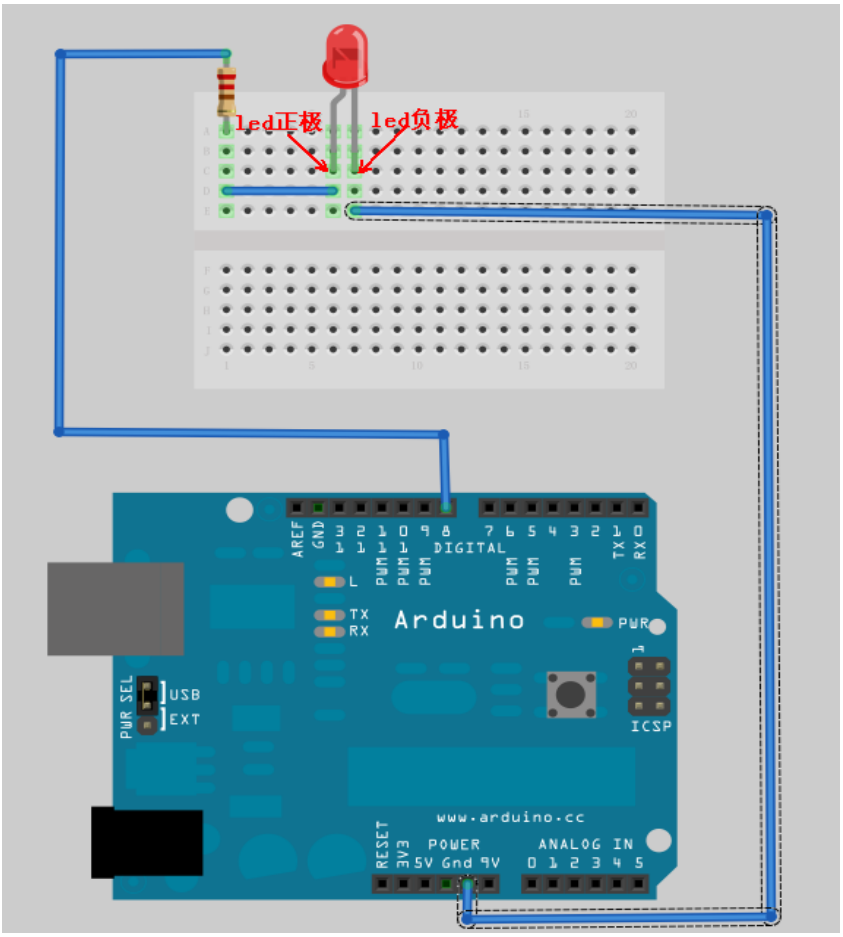


图 1.5 发光二极管的具体接线

简单的控制一个 led 灯的闪烁实验

1) 实验器件

- Led 灯：1 个
- 220Ω的电阻：1 个
- 多彩面包板实验跳线：若干

2) 实验连线

按照 Arduino 使用介绍将控制板、Prototype Shied 板子、面包板连接好，下载线插好。最后，按照图 1.5 将发光二级管连接到数字的第 8 引脚。这样我们就完成了实验的连线部分。

3) 实验原理

先设置数字 8 引脚为高电平点亮 led 灯，然后延时 1s，接着设置数字 8 引脚为低电平熄灭 led 灯，再延时 1s。这样使 led 灯亮 1s、灭 1s，在视觉上就形成闪烁状态。如果想让 led 快速闪烁，可以将延时时间设置的小一些，但不能过小，过小的话人眼就识别不出来了，看上去就像 led 灯一直在亮着；如果想让 led 慢一点闪烁，可以将延时时间设置的大一些，但也不能过大，过大的话就没有闪烁的效果了。

4) 程序代码

程序代码在简单 led 程序文件夹中，双击打开后有一个 led1 文件夹，接着双击打开后可以看见有一个 led1.pde 文件，双击图标即可打开。打开后我们可以看到这是 arduino 编程软件窗口，上面有本实验的程序代码。

程序代码如下：

```
int ledPin=8 //设定控制 LED 的数字 IO 脚

void setup()

{

    pinMode(ledPin,OUTPUT);//设定数字 IO 口的模式，OUTPUT 为输出
```

```
}

void loop()

{

    digitalWrite(ledPin,HIGH); //设定 PIN8 脚为 HIGH = 5V 左右

    delay(1000); //设定延时时间，1000 = 1 秒

    digitalWrite(ledPin,LOW); //设定 PIN8 脚为 LOW = 0V

    delay(1000); //设定延时时间，1000 = 1 秒

}
```

从 Arduino 教程中我们可以知道，Arduino 语言是以 setup()开头，loop()作为主体的一个程序构架。setup()是用来初始化变量，管脚模式，调用库函数等等，此函数只运行一次。本程序在 setup()中用数字 IO 口输入输出模式定义函数 pinMode（pin，mode），将数字的第 8 引脚设置为输出模式。

loop()函数是一个循环函数，函数内的语句周而复始的循环执行，本程序在 loop()中先用 数字 IO 口输出电平定义函数 digitalWrite(pin, value)，将数字 8 口定义为高电平，点亮 led 灯；接着调用延时函数 delay(ms)（单位 ms）延时 1000ms，让发光二极管亮 1s；再用数字 IO 口输出电平定义函数 digitalWrite(pin, value)，将数字 8 口定义为低电平，熄灭 led 灯；接着再调用延时函数 delay(ms)（单位 ms）延时 1000ms，让发光二极管熄灭 1s。因为 loop()函数是一个循环函数，所以这个过程会不断的循环。

5）下载程序

按照 arduino 教程中的程序下载方法将本程序下载到实验板中。

6）程序功能

将程序下载到实验板后我们可以观察到，发光二极管以 1s 的时间间隔不断的闪烁。

广告灯效果实验

1）实验器件

- Led 灯：6 个
- 220Ω的电阻：6 个
- 多彩面包板实验跳线：若干

2）实验连线

按照上述方法将板子和数据线连好。然后按照二级管的接线方法，将六个 LED 灯依次接到数字 1~6 引脚上。如图：

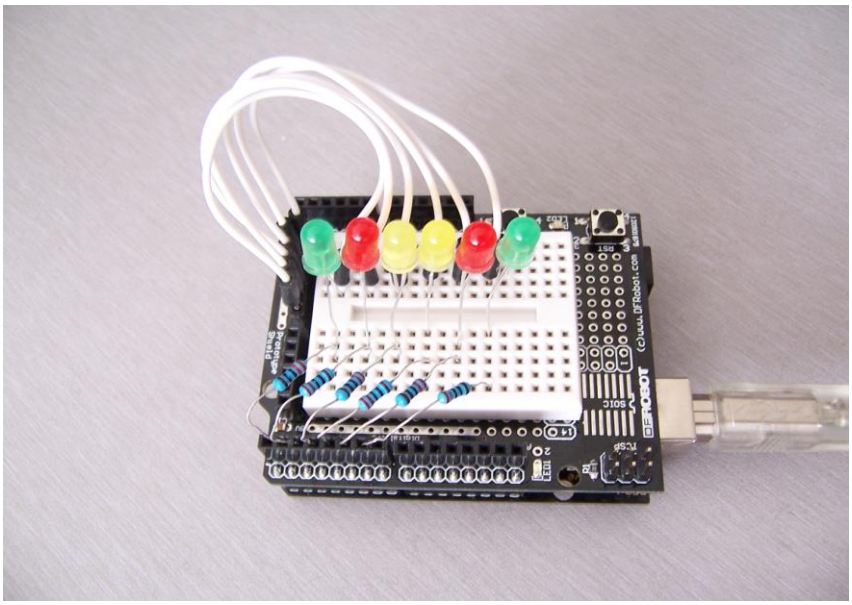


图 1.6　广告灯实验的接线

3）实验原理

在生活中我们经常会看到一些由各种颜色的 led 灯组成的广告牌，广告牌上各个位置上的 led 灯不断的亮灭变化，就形成各种不同的效果。本节实验就是利用 led 灯编程模拟广告灯的效果。

在程序中我们设置 led 灯亮灭的次序和时间，这样就可以组成不同的效果。样式一子程序：led 首先从左边的绿灯开始间隔 200ms 依次点亮六个 led 灯，如图 1.6，接着从右边的绿灯开始间隔 200ms 依次熄灭六个 led 灯。灯闪烁子程序：六个 led 灯首先全部点亮，接着延时 200ms，最后六个 led 灯全部熄灭，这个过程循环两次就实现了闪烁的效果。样式二子程序设置 k 和 j 的值让中间的两个黄灯亮先亮，接着让挨着两个

黄灯两边的红灯亮，最后让两边的绿灯亮；执行一遍后改变 k 和 j 的值让让两边的绿灯先熄灭，接着两边的红灯熄灭，最后中间的两个黄灯熄灭。样式三子程序：设置 k 和 j 的值，让两边的绿灯亮 400ms 后再熄灭，接着让两边的红灯亮 400ms 后再熄灭，最后让中间的两个黄灯亮 400ms 后再熄灭；执行一遍后改变 k 和 j 的值让两个红灯亮 400ms 后熄灭，接着让两边的绿灯亮 400ms 后熄灭。

4）程序代码

程序代码在广告灯程序文件夹中，双击打开后有一个 led2 文件夹，接着双击打开后可以看见有一个 led2.pde 文件 ,双击图标即可打开。打开后我们可以看到这是 arduino 编程软件窗口，上面有本实验的程序代码。

程序代码如下：

```
//设置控制 Led 的数字 IO 脚
int Led1 = 1;
int Led2 = 2;
int Led3 = 3;
int Led4 = 4;
int Led5 = 5;
int Led6 = 6;
//led 灯花样显示样式 1 子程序
void style_1(void)
{
  unsigned char j;
  for(j=1;j<=6;j++)//每隔 200ms 依次点亮 1~6 引脚相连的 led 灯
  {
    digitalWrite(j,HIGH);//点亮 j 引脚相连的 led 灯
    delay(200);//延时 200ms
  }
  for(j=6;j>=1;j--)//每隔 200ms 依次熄灭 6~1 引脚相连的 led 灯
  {
    digitalWrite(j,LOW);//熄灭 j 引脚相连的 led 灯
    delay(200);//延时 200ms
  }
}
//灯闪烁子程序
void flash(void)
{
  unsigned char j,k;
  for(k=0;k<=1;k++)//闪烁两次
  {
```

```
    for(j=1;j<=6;j++)//点亮 1~6 引脚相连的 led 灯
      digitalWrite(j,HIGH);//点亮与 j 引脚相连的 led 灯
    delay(200);//延时 200ms
    for(j=1;j<=6;j++)//熄灭 1~6 引脚相连的 led 灯
      digitalWrite(j,LOW);//熄灭与 j 引脚相连的 led 灯
    delay(200);//延时 200ms
  }
}
//led 灯花样显示样式 2 子程序
void style_2(void)
{
  unsigned char j,k;
  k=1;//设置 k 的初值为 1
  for(j=3;j>=1;j--)
  {
    digitalWrite(j,HIGH);//点亮灯
    digitalWrite(j+k,HIGH);//点亮灯
    delay(400);//延时 400ms
    k +=2;//k 值加 2
  }
  k=5;//设置 k 值为 5
  for(j=1;j<=3;j++)
  {
    digitalWrite(j,LOW);//熄灭灯
    digitalWrite(j+k,LOW);//熄灭灯
    delay(400);//延时 400ms
    k -=2;//k 值减 2
  }
}
//led 灯花样显示样式 3 子程序
void style_3(void)
{
  unsigned char j,k;//led 灯花样显示样式 3 子程序
  k=5;//设置 k 值为 5
  for(j=1;j<=3;j++)
  {
    digitalWrite(j,HIGH);//点亮灯
    digitalWrite(j+k,HIGH);//点亮灯
    delay(400);//延时 400ms
    digitalWrite(j,LOW);//熄灭灯
    digitalWrite(j+k,LOW);//熄灭灯
    k -=2;//k 值减 2
  }
  k=3;//设置 k 值为 3
  for(j=2;j>=1;j--)
  {
```

```
      digitalWrite(j,HIGH);//点亮灯
      digitalWrite(j+k,HIGH);//点亮灯
      delay(400);//延时 400ms
      digitalWrite(j,LOW);//熄灭灯
      digitalWrite(j+k,LOW);//熄灭灯
      k +=2;//k 值加 2
   }
}
void setup()
{
   unsigned char i;
   for(i=1;i<=6;i++)//依次设置 1~6 个数字引脚为输出模式
      pinMode(i,OUTPUT);//设置第 i 个引脚为输出模式
}
void loop()
{
   style_1();//样式 1
   flash();//闪烁
   style_2();//样式 2
   flash();//闪烁
   style_3();//样式 3
   flash();//闪烁
}
```

程序代码中用到的：

for(i=1;i<=6;i++)//依次设置 1~6 个数字引脚为输出模式
pinMode(i,OUTPUT);//设置第 i 个引脚为输出模式

这是一个 for 循环。它的一般形式为: for(<初始化>; <条件表达式>; <增量>) 语句; 初始化总是一个赋值语句，它用来给循环控制变量赋初值; 条件表达式是一个关系表达式，它决定什么时候退出循环; 增量定义循环控制变量每循环一次后 按什么方式变化。这三个部分之间用";"分开。 例如: for(i=1; i<=10; i++) 语句; 上例中先给 " i " 赋初值 1，判断 " i " 是否小于等于 10，若是则执行语句，之后值增 加 1。再重新判断，直到条件为假，即 i>10 时，结束循环。

5）下载程序

按照 arduino 教程中的程序下载方法将本程序下载到实验板中。

6）程序功能

将程序下载到实验板后我们可以观察到 ,六个 led 不断的循环执行样式一子程序—>闪烁子程序—>样式二子程序—>闪烁子程序—>样式三子程序—>闪烁子程序。

在掌握了以上两个程序后，大家可以充分发挥自己的想象，编写出自己想要的 led 灯效果，玩转多彩 led 灯。

第二节 蜂鸣器实验

一、蜂鸣器介绍

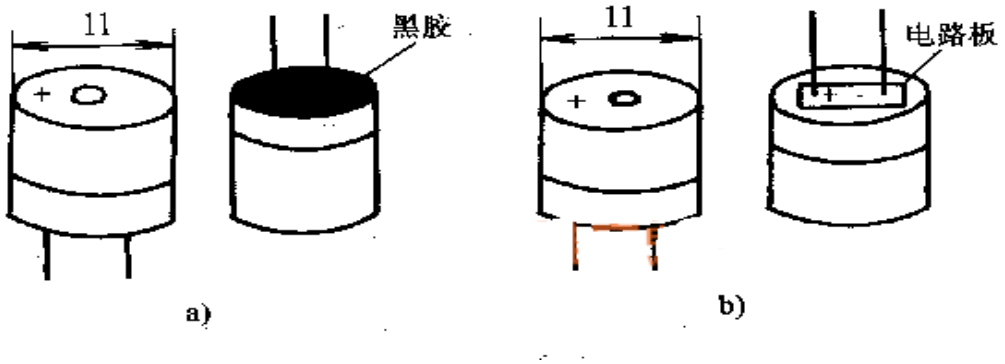
1、认识蜂鸣器

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。

图 2.1 蜂鸣器

按其驱动方式的不同，可分为：有源蜂鸣器（内含驱动线路）和无源蜂鸣器（外部驱动）

教你区分有源蜂鸣器和无源蜂鸣器,现在市场上出售的一种小型蜂鸣器因其体积小(直径只有 llmm)、重量轻、价格低、结构牢靠，而广泛地应用在各种需要发声的电器设备、电子制作和单片机等电路中。有源蜂鸣器和无源蜂鸣器的外观如图 a、b 所示。a)有源 b)无源。



从图 a、b 外观上看，两种蜂鸣器好像一样，但仔细看，两者的高度略有区别，有源蜂鸣器 a，高度为 9mm，而无源蜂鸣器 b 的高度为 8mm。如将两种蜂鸣器的引脚郡朝上放置时，可以看出有绿色电路板的一种是无源蜂鸣器，没有电路板而用黑胶封闭的一种是有源蜂鸣器。进一步判断有源蜂鸣器和无源蜂鸣器，还可以用万用表电阻档 Rxl 档测试:用黑表笔接蜂鸣器 "+"引脚,红表笔在另一引脚上来回碰触,如果触发出咔、咔声的且电阻只有 8Ω(或 16Ω)的是无源蜂鸣器;如果能发出持续声音的，且电阻在几百欧以上的，是有源蜂鸣器。 有源蜂鸣器直接接上额定电源(新的蜂鸣器在标签上都有注明)就可连续发声;而无源蜂鸣器则和电磁扬声器一样，需要接在音频输出电路中才能发声。

按构造方式的不同，可分为：电磁式蜂鸣器和压电式蜂鸣器；

压电式蜂鸣器 压电式蜂鸣器主要由多谐振荡器、压电蜂鸣片、阻抗匹配器及共鸣箱、外壳等组成。有的压电式蜂鸣器外壳上还装有发光二极管。多谐振荡器由晶体管或集成电路构成。当接通电源后(1.5~15V 直流工作电压),多谐振荡器起振,输出 1.5~2.5kHZ 的音频信号,阻抗匹配器推动压电蜂鸣片发声压电蜂鸣片由锆钛酸铅或铌镁酸铅压电陶瓷材料制成。在陶瓷片的两面镀上银电极，经极化和老化处理后，再与黄铜片或不锈钢片粘在一起。

电磁式蜂鸣器由振荡器、电磁线圈、磁铁、振动膜片及外壳等组成。接通电源后，振荡器产生的音频信号电流通过电磁线圈，使电磁线圈产生磁场。振动膜片在电磁线圈和磁铁的相互作用下，周期性地振动发声。

2、工作原理

蜂鸣器发声原理是电流通过电磁线圈，使电磁线圈产生磁场来驱动振动膜发声的，因此需要一定的电流才能驱动它，本实验用的蜂鸣器内部带有驱动电路，所以可以直接使用。当与蜂鸣器连接的引脚为高电平时，内部驱动电路导通，蜂鸣器发出声音；当与蜂鸣器连接的引脚为低电平，内部驱动电路截止，蜂鸣器不发出声音。

3、蜂鸣器的连线

本实验用的蜂鸣器内部带有驱动电路,所以可以直接将蜂鸣器的正极连接到数字口，蜂鸣器的负极连接到 GND 插口中。如下图：

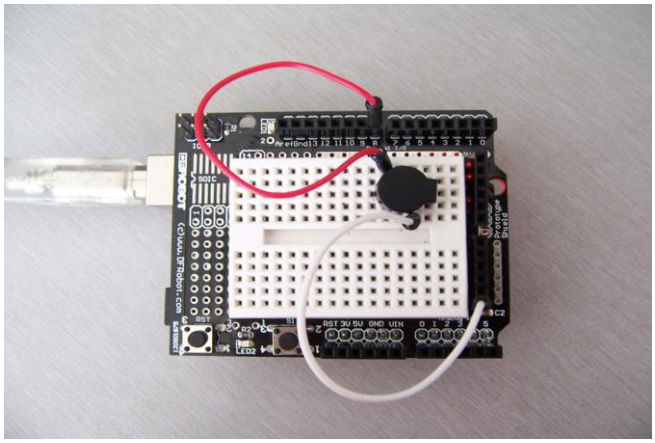


图 2.1 蜂鸣器接线图

二、蜂鸣器模拟救护车警笛声音实验

1、实验器件

- 蜂鸣器：1 个
- 多彩面包板实验跳线：若干

2、实验连线

按照 Arduino 教程将控制板、Prototype shield 板子、面包板连接好，下载线插好。然后按照蜂鸣器的接法将蜂鸣器连接到数字 7 口上，连线完毕。

3、实验原理

蜂鸣器发出声音的时间间隔不同，频率就不同，所以发出的声音就不同。根据这一原理我们通过改变蜂鸣器发出声音的时间间隔，来发出不同种声音，来模拟各种声音。

本程序首先让蜂鸣器间隔 1ms 发出一种频率的声音，循环 80 次；接着让蜂鸣器间隔 2ms 发出另一种频率的声音，循环 100 次。

4、程序代码

程序代码在蜂鸣器救护车报警声音程序文件夹中，双击打开后有一个 buzzer 文件夹，接着双击打开后可以看见有一个 buzzer.pde 文件，双击图标即可打开。打开后我们可以看到这是 arduino 编程软件窗口，上面有本实验的程序代码。

程序代码如下：

```
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
```

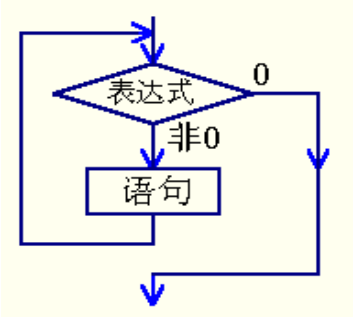
```
}
void loop()
{
  unsigned char i,j;//定义变量
  while(1)
  {
    for(i=0;i<80;i++)//输出一个频率的声音
    {
      digitalWrite(buzzer,HIGH);//发声音
      delay(1);//延时 1ms
      digitalWrite(buzzer,LOW);//不发声音
      delay(1);//延时 ms
    }
    for(i=0;i<100;i++)//输出另一个频率的声音
    {
      digitalWrite(buzzer,HIGH);//发声音
      delay(2);//延时 2ms
      digitalWrite(buzzer,LOW);//不发声音
      delay(2);//延时 2ms
    }
  }
}
```

在 loop（ ）中用的 while 也是一个循环语句，一般形式：

while(表达式)

语句

表达式是循环条件，语句是循环体。语义是：计算表达式的值，当值为真(非 0)时， 执行循环体语句。其执行过程可用下图表 ：



作用：实现“当型”循环。当“表达式”非 0（真）时，执行“语句”。“语句”是被循环执行的程序，称为“循环体”。

5、下载程序

按照 arduino 教程中的程序下载方法将本程序下载到实验板中。

6、程序功能

将程序下载到实验板后我们可以听到，蜂鸣器发出救护车警笛声。

掌握本程序后，大家可以在程序中自己改变时间间隔，调试出各种频率的声音。

第三节 数码管实验

一、数码管介绍

1、认识数码管

数码管是一种半导体发光器件，其基本单元是发光二极管.数码管按段数分为七段数码管和八段数码管，八段数码管比七段数码管多一个发光二极管单元（多一个小数点显示）；

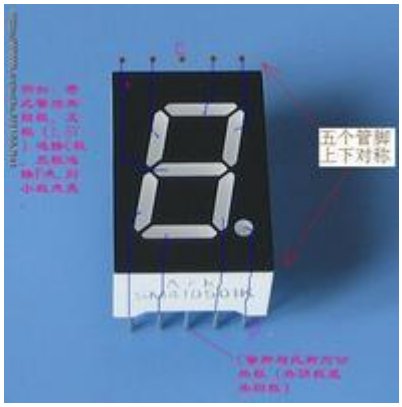


图 3.1 数码管实物图

按能显示多少个“8”可分为 1 位、2 位、4 位等等数码管；

图 3.2 各种数码管

按发光二极管单元连接方式分为共阳极数码管和共阴极数码管。共阳数码管是指将所

有发光二极管的阳极接到一起形成公共阳极(COM)的数码管。共阳数码管在应用时应将公共极 COM 接到+5V，当某一字段发光二极管的阴极为低电平时，相应字段就点亮。当某一字段的阴极为高电平时，相应字段就不亮。

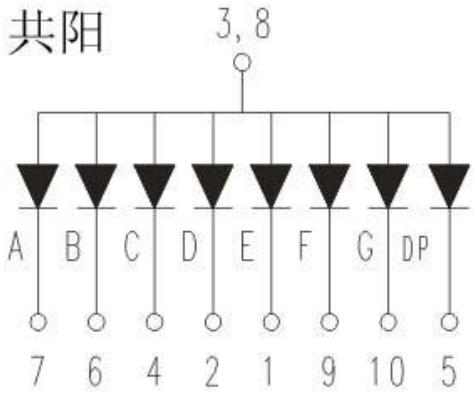


图 3.3　共阳极数码管内部结构

共阳数码管是指将所有发光二极管的阳极接到一起形成公共阳极(COM)的数码管。共阳数码管在应用时应将公共极 PWR 接到电源输入 PWR 上，当某一字段发光二极管的阴极极为低电平时，相应字段就点亮。当某一字段的阴极为高电平时，相应字段就不亮。

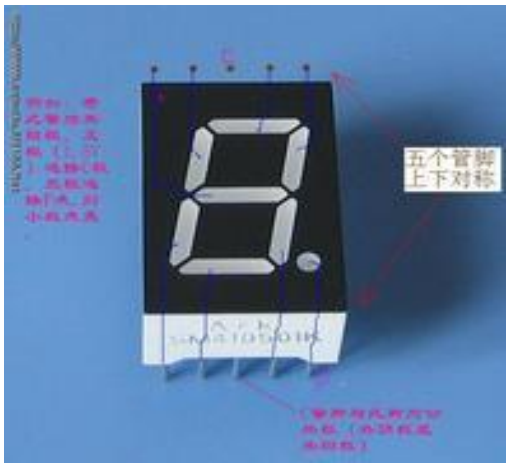


图 3.4　共阴极数码管内部结构

2、工作原理

数码管的每一段是由发光二极管组成，所以在使用时跟发光二极管一样，也要连接限流电阻，否则电流过大会烧毁发光二极管的。本实验用的是共阳极的数码管，共阳数码管在应用时应将公共极 COM 接到+5V，当某一字段发光二极管的阴极为低电平时，相应字段就点亮。当某一字段的阴极为高电平时，相应字段就不亮。

3、数码管的连线

将限流电阻的一端插到数字 I/O 中，另一端与数码管的字段引脚相连，剩下的六个字段和一个小数点依次按照这种方法接。将公共极 COM 如果是共阳极的就接到+5V，如果是共阴极的就接到 GND。

二、数码管显示数字的实验

1、实验器件

- 数码管：1 个
- 220Ω的电阻：8 个
- 多彩面包板实验跳线：若干

2、实验连线

按照 Arduino 教程将控制板、Prototype shield 板子、面包板连接好，下载线插好。按数码管的接法将数码管 g 段通过限流电阻与数字的 9 引脚相连，如图 3.4 中的(a) 图，f 段通过限流电阻与数字 8 引脚相连，共阳极与 5V 插口相连，同样的接法 a、b 分别接 7、6 引脚，e、d 分别接 10、11 引脚，第二个共阳极可以不接，c、DP 分别接 5、4 引脚，连线完毕。如下图：

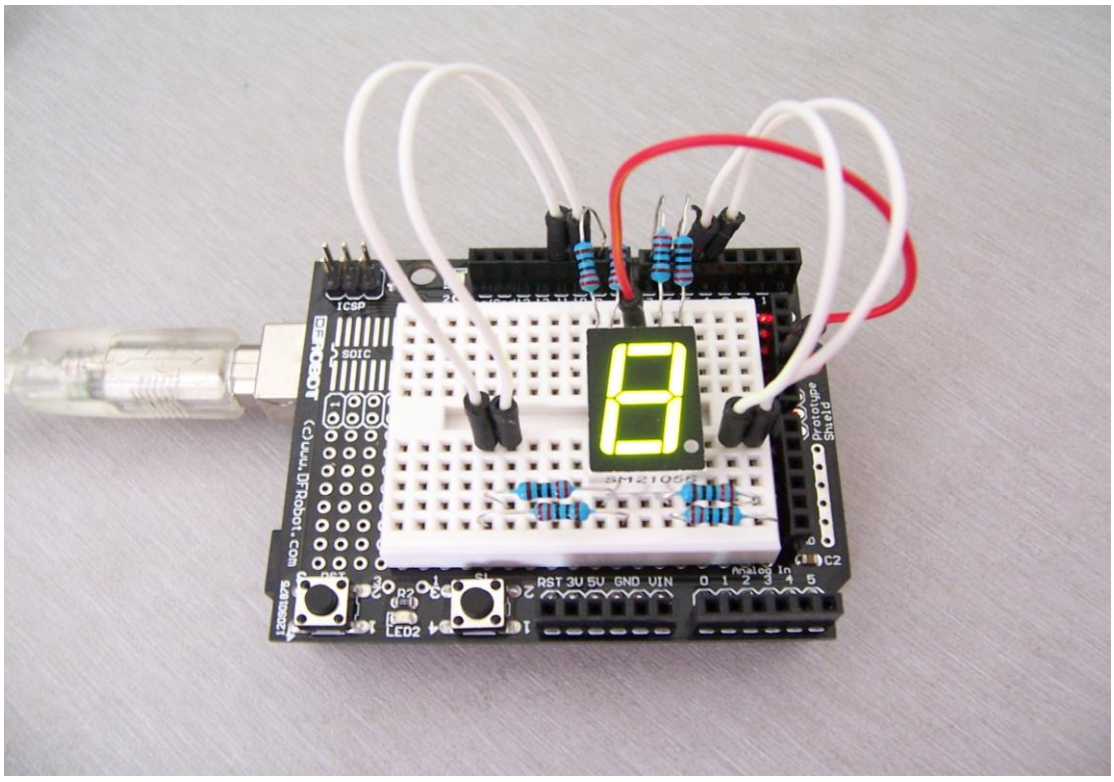


图 3.5 数码管接线图

3、实验原理

数码管共有七段显示数字的段，还有一个显示小数点的段。当让数码管显示数字时，

只要将相应的段点亮即可。例如：让数码管显示数字 1，则将 b、c 段点亮即可。

将每个数字写成一个子程序。在主程序中每隔 2s 显示一个数字，让数码管循环显示 1～8 数字。每一个数字显示的时间由延时时间来决定，时间设置的大些，显示的时间就长些，时间设置的小些，显示的时间就短。

4、程序代码

程序代码在数码管显示数字程序文件夹中，双击打开后有一个 digital_tube1 文件夹，接着双击打开后可以看见有一个 digital_tube1.pde 文件，双击图标即可打开。打开后我们可以看到这是 arduino 编程软件窗口，上面有本实验的程序代码。

程序代码如下：

```
//设置控制各段的数字 IO 脚
int a=7;
int b=6;
int c=5;
int d=11;
int e=10;
int f=8;
int g=9;
int dp=4;

//显示数字 1
void digital_1(void)
{
  unsigned char j;
  digitalWrite(c,LOW);//给数字 5 引脚低电平，点亮 c 段
  digitalWrite(b,LOW);//点亮 b 段
  for(j=7;j<=11;j++)//熄灭其余段
    digitalWrite(j,HIGH);
  digitalWrite(dp,HIGH);//熄灭小数点 DP 段
}
//显示数字 2
void digital_2(void)
{
  unsigned char j;
  digitalWrite(b,LOW);
  digitalWrite(a,LOW);
  for(j=9;j<=11;j++)
    digitalWrite(j,LOW);
  digitalWrite(dp,HIGH);
  digitalWrite(c,HIGH);
  digitalWrite(f,HIGH);
```

```
}
//显示数字 3
void digital_3(void)
{
  unsigned char j;
  digitalWrite(g,LOW);
  digitalWrite(d,LOW);
  for(j=5;j<=7;j++)
    digitalWrite(j,LOW);
  digitalWrite(dp,HIGH);
  digitalWrite(f,HIGH);
  digitalWrite(e,HIGH);
}
//显示数字 4
void digital_4(void)
{
  digitalWrite(c,LOW);
  digitalWrite(b,LOW);
  digitalWrite(f,LOW);
  digitalWrite(g,LOW);
  digitalWrite(dp,HIGH);
  digitalWrite(a,HIGH);
  digitalWrite(e,HIGH);
  digitalWrite(d,HIGH);
}
//显示数字 5
void digital_5(void)
{
  unsigned char j;
  for(j=7;j<=9;j++)
  digitalWrite(j,LOW);
  digitalWrite(c,LOW);
  digitalWrite(d,LOW);
  digitalWrite(dp,HIGH);
  digitalWrite(b,HIGH);
  digitalWrite(e,HIGH);
}
//显示数字 6
void digital_6(void)
{
  unsigned char j;
  for(j=7;j<=11;j++)
    digitalWrite(j,LOW);
  digitalWrite(c,LOW);
  digitalWrite(dp,HIGH);
  digitalWrite(b,HIGH);
```

```
}
//显示数字 7
void digital_7(void)
{
  unsigned char j;
  for(j=5;j<=7;j++)
    digitalWrite(j,LOW);
  digitalWrite(dp,HIGH);
  for(j=8;j<=11;j++)
    digitalWrite(j,HIGH);
}
//显示数字 8
void digital_8(void)
{
  unsigned char j;
  for(j=5;j<=11;j++)
    digitalWrite(j,LOW);
  digitalWrite(dp,HIGH);
}
void setup()
{
  int i;//定义变量
  for(i=4;i<=11;i++)
    pinMode(i,OUTPUT);//设置 4～11 引脚为输出模式
}
void loop()
{
  while(1)
  {
    digital_1();//数字 1
    delay(2000);//延时 2s
    digital_2();
    delay(2000);
    digital_3();
    delay(2000);
    digital_4();
    delay(2000);
    digital_5();
    delay(2000);
    digital_6();
    delay(2000);
    digital_7();
    delay(2000);
    digital_8();
    delay(2000);
  }
```

```
}
```

在 setup()前面定义了一系列的数字显示子程序，这些子程序的定义可以方便在 loop()中使用，使用时只需将子程序的名写上即可。

5、下载程序

按照 arduino 教程中的程序下载方法将本程序下载到实验板中。

6、程序功能

将程序下载到实验板后我们可以看到，数码管循环显示数字 1~8，每隔数字显示两秒钟。

掌握本程序后，大家可以发挥自己的想象，做出各种数码管实验。

第四节 按键实验

一、按键介绍

1、认识按键

按键是一种常用的控制电器元件，常用来接通或断开 ‘控制电路 （其中电流很小），从而达到控制电动机或其他电气设备运行目的的一种开关。电子产品大都有用到按键这个最基本人机接口工具，随着工业水平的提升与创新，按键外观的也变的越来越多样化及丰富的视觉效果。

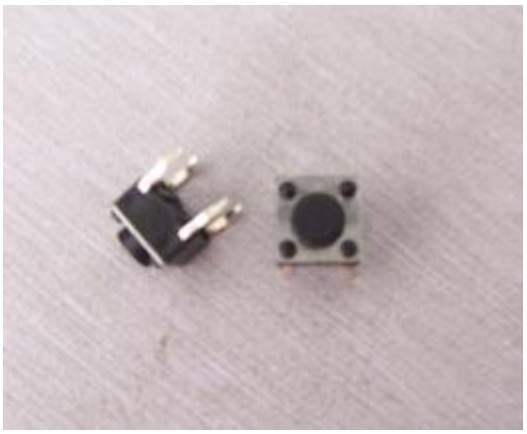


图 4.1 按键

2、工作原理

从实验盒里拿出一个按键，观察按键的背面，这时你可以看到按键背面如图：

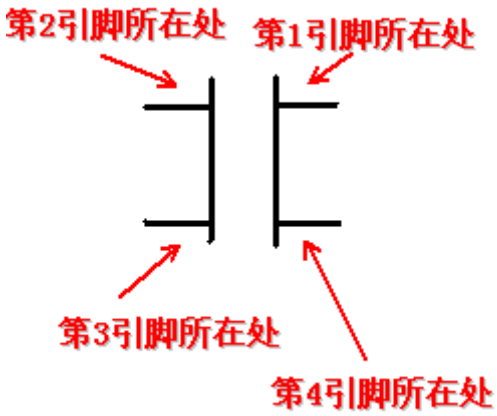


图 4.2 按键背面

这表示在没有按键按下时左侧两个引脚是导通的，右侧两个引脚也是导通的，而上侧和下侧两对引脚是没有导通的。按键正面图如下：

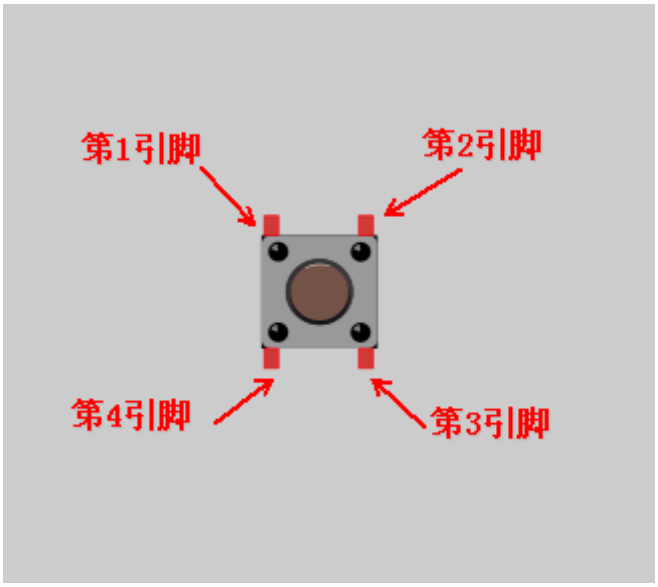


图 4.3 按键正面

本实验用到的是四脚按键，在按键没有按下时，1 和 4 引脚连通、2 和 3 引脚连通；当按键按下时，1 和 4 引脚连通、2 和 3 引脚连通、1 和 2 引脚连通、4 和 3 引脚连通。

3、按键的连线

一般用按键是为了在按键按下时控制其他器件，所以我们在连接时，将第 1 引脚连接在 5V 插口中，将第 2 引脚接在模拟口来读取电压值。或者是利用 4、3 引脚配合。

这样当按键没有按下时，模拟口的电压值为 0V 左右（用数字二进制表示为 0）；当按键按下时，模拟口的电压值为 5V 左右（用数字二进制表示为 1023）。

二、按键的实验

1、按键控制 led 等亮灭实验

1)实验器件

- 按键：1 个
- 220Ω的电阻：1 个
- Led 灯：1 个
- 多彩面包板实验跳线：若干

2)实验连线

按照 Arduino 教程将控制板、Prototype shield 板子、面包板连接好，下载线插好。将按键插在面包板的适当位置，第一引脚处插一根导线，导线的另一端插在 5V 插孔中，第二引脚处也插一根导线，导线的另一端插在模拟口的第 0 口。这样按键就连接好了。接下来将 led 灯通过限流电阻接到数字 7 引脚上。整个连线完成。如图所示：

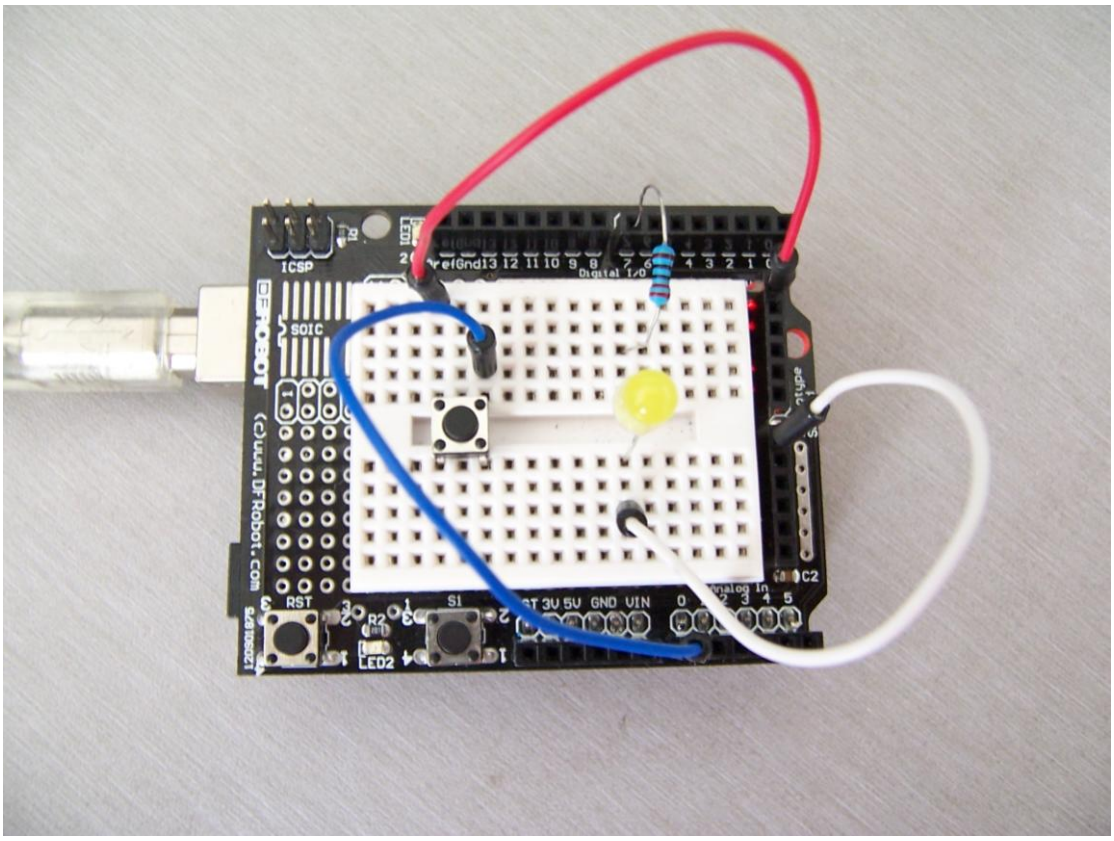


图 4.4　按键控制 led 接线图

3)实验原理

当按键没有被按下时，模拟口电压为 0V，led 灯熄灭；当按键按下时，模拟口的电压值为 5V，所以只要判断电压值大于 2.5V 即可知道按键被按下，led 灯点亮。

4）程序代码

程序代码在简单按键程序文件夹中，双击打开后有一个 key1 文件夹，接着双击打开后可以看见有一个 key1.pde 文件，双击图标即可打开。打开后我们可以看到这是 arduino 编程软件窗口，上面有本实验的程序代码。

程序代码如下：

```
int key=7;//设置控制按键的数字 IO 脚
void setup()
{
  pinMode(key,OUTPUT);//设置数字 IO 引脚为输出模式
}
void loop()
```

```
{
  int i;//定义变量
  while(1)
  {
    i=analogRead(0);//读取模拟 0 口电压值
    if(i>1000)//如果电压值大于 1000
      digitalWrite(key,HIGH);//设置第七引脚为高电平，点亮 led 灯
    else
      digitalWrite(key,LOW);//设置第七引脚为低电平，熄灭 led 灯
  }
}
```

5）下载程序

按照 arduino 教程中的程序下载方法将本程序下载到实验板中。

6）程序功能

将程序下载到实验板后，当我们按下按键，led 灯点亮；当我们不按按键，led 熄灭。

2、掷骰子实验

1）实验器件

- 按键：1 个
- 数码管：1 个
- 220Ω的电阻 8 个
- 多彩面包板实验跳线：若干

2）实验连线

将板子和数据线连好。然后按照数码管的接线方法，将数码管连接好，各引脚的接线同第三节。最后将按键接到模拟口 0。接线图如下图：

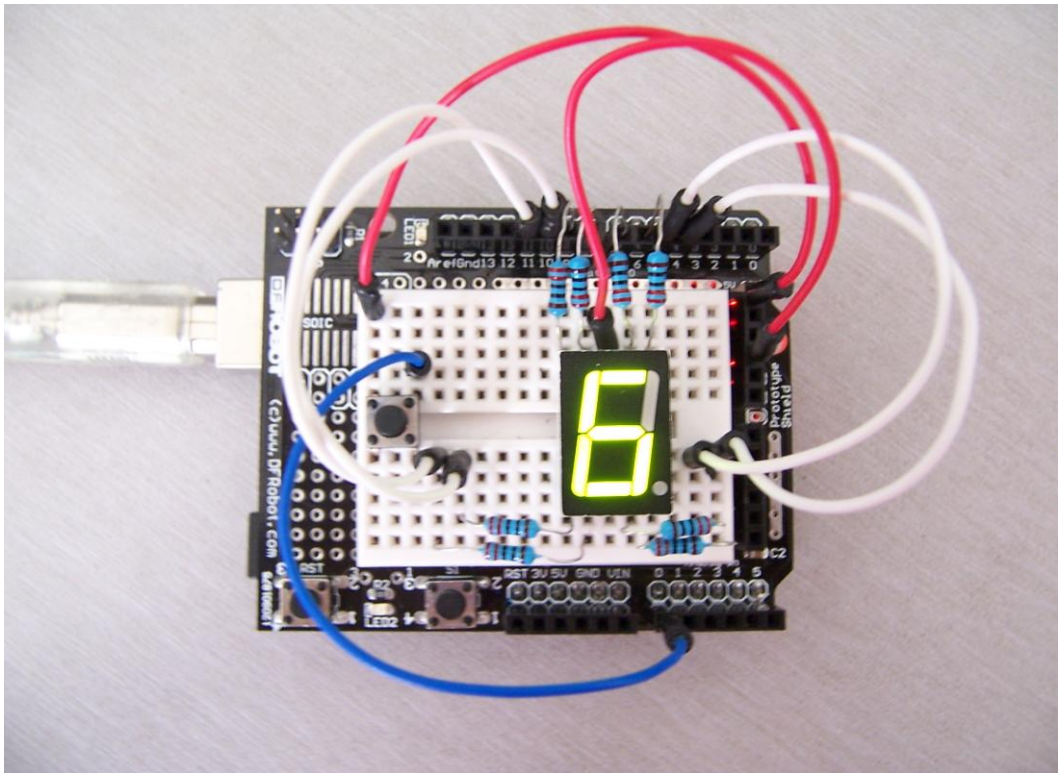


图 4.5 掷骰子实验接线

3）实验原理

让数码管快速的显示数字 1~8。当按键按下时，模拟 0 口会读到电压值为 5V，只要电压值大于 2.5V 就停留在当前显示的数字；松开按键，电压值为 0，数字继续滚动。类似于，先摇动骰子，停止时骰子上面的数字即是我们摇出的点数。

4）程序代码

程序代码在掷骰子程序文件夹中，双击打开后有一个 key2 文件夹，接着双击打开后可以看见有一个 key2.pde 文件，双击图标即可打开。打开后我们可以看到这是 arduino 编程软件窗口，上面有本实验的程序代码。大家可以参考 arduino 教程，了解 arduino 语言中的各语句的功能。结合 arduino 教程，理解程序代码。

程序代码如下：

```
//设置控制各段的数字 IO 脚
int a=7;
int b=6;
int c=5;
int d=11;
int e=10;
int f=8;
int g=9;
```

```
int dp=4;

//显示数字 1
void digital_1(void)
{
  unsigned char j;
  digitalWrite(c,LOW);//给数字 5 引脚低电平，点亮 c 段
  digitalWrite(b,LOW);//点亮 b 段
  for(j=7;j<=11;j++)//熄灭其余段
    digitalWrite(j,HIGH);
  digitalWrite(dp,HIGH);//熄灭小数点 DP 段
}
//显示数字 2
void digital_2(void)
{
  unsigned char j;
  digitalWrite(b,LOW);
  digitalWrite(a,LOW);
  for(j=9;j<=11;j++)
    digitalWrite(j,LOW);
  digitalWrite(dp,HIGH);
  digitalWrite(c,HIGH);
  digitalWrite(f,HIGH);
}
//显示数字 3
void digital_3(void)
{
  unsigned char j;
  digitalWrite(g,LOW);
  digitalWrite(d,LOW);
  for(j=5;j<=7;j++)
    digitalWrite(j,LOW);
  digitalWrite(dp,HIGH);
  digitalWrite(f,HIGH);
  digitalWrite(e,HIGH);
}
//显示数字 4
void digital_4(void)
{
  digitalWrite(c,LOW);
  digitalWrite(b,LOW);
  digitalWrite(f,LOW);
  digitalWrite(g,LOW);
  digitalWrite(dp,HIGH);
  digitalWrite(a,HIGH);
  digitalWrite(e,HIGH);
```

```
  digitalWrite(d,HIGH);
}
//显示数字 5
void digital_5(void)
{
  unsigned char j;
  for(j=7;j<=9;j++)
    digitalWrite(j,LOW);
  digitalWrite(c,LOW);
  digitalWrite(d,LOW);
  digitalWrite(dp,HIGH);
  digitalWrite(b,HIGH);
  digitalWrite(e,HIGH);
}
//显示数字 6
void digital_6(void)
{
  unsigned char j;
  for(j=7;j<=11;j++)
    digitalWrite(j,LOW);
  digitalWrite(c,LOW);
  digitalWrite(dp,HIGH);
  digitalWrite(b,HIGH);
}
//显示数字 7
void digital_7(void)
{
  unsigned char j;
  for(j=5;j<=7;j++)
    digitalWrite(j,LOW);
  digitalWrite(dp,HIGH);
  for(j=8;j<=11;j++)
    digitalWrite(j,HIGH);
}
//显示数字 8
void digital_8(void)
{
  unsigned char j;
  for(j=5;j<=11;j++)
    digitalWrite(j,LOW);
  digitalWrite(dp,HIGH);
}
void setup()
{
    int i;
    for(i=4;i<=11;i++)
```

```
    {
    pinMode(i,OUTPUT);//设置 4~11 口味输出模式
    }
}
void loop()
{
  while(1)
  {
      digital_1();//显示数字 1
      while(analogRead(0)>512);//如果读到模拟 0 口的值为 0 则说明有按键按下
      delay(200);//延时 200ms
       digital_2();
       while(analogRead(0)>512);
      delay(200);
       digital_3();
       while(analogRead(0)>512);
      delay(200);
       digital_4();
       while(analogRead(0)>512);
      delay(200);
       digital_5();
       while(analogRead(0)>512);
      delay(200);
       digital_6();
       while(analogRead(0)>512);
      delay(200);
       digital_7();
       while(analogRead(0)>512);
      delay(200);
       digital_8();
       while(analogRead(0)>512);
      delay(200);
  }
}
```

5）下载程序

按照 arduino 教程中的程序下载方法将本程序下载到实验板中。

6）程序功能

将程序下载到实验板后我们可以观察到，数码管不断的循环显示数字 1~8，当按下

按键后，数码管显示当前数字，直至松开按键。

在掌握了以上两个程序后，大家可以充分发挥自己的想象，编写出自己想要的效果。

第五节 倾斜开关实验

一、倾斜开关介绍

1、了解倾斜开关

本节实验所使用的倾斜开关是内部带有一个金属滚珠的滚珠倾斜开关，如图所示：

图 5.1 SW—200D 倾斜开关

滚珠开关：也叫碰珠开关、摇珠开关、钢珠开关、倾斜开关，倒顺开关、角度传感器。它主要是利用滚珠在开关内随不同倾斜角度的变化，达到触发电路的目的。

目前滚珠开关在市场上使用的常用型号有 SW-200D、SW-460、SW-300DA 等，本节使用的是 SW-200D 型号的。这类开关不象传统的水银开关，它功效同水银开关，但没有水银开关的环保及安全等问题。

2、工作原理

观察倾斜开关我们可以发现，倾斜开关的一端为金色导针，另一端为银色导针。金色一端为<ON>导通触发端银色一端为<OFF>开路端当受到外力摇晃而达到适当晃动力时或金色一端设置角度低于水平适当角度时导电接脚电气特性会产生短时间导通或持续导通<ON>状态 。而当电气特性要恢复开路状态<OFF>时开关设置环境必须为静止，且银色一端设置角度需低于水平 10 度。

3、倾斜开关连线

将倾斜开关银色的一端连接到 5V 插口，金色一端连接到模拟口。

二、倾斜开关控制 led 灯的亮灭

1、实验器件

- 倾斜开关：1 个
- Led 灯：1 个
- 220Ω电阻：1 个
- 多彩面包板实验跳线：若干

2、实验连线

按照 Arduino 教程将控制板、Prototype shield 板子、面包板连接好，下载线插好。然后将 led 灯连接到数字 8 引脚，倾斜开关连接到模拟 5 引脚。如图：

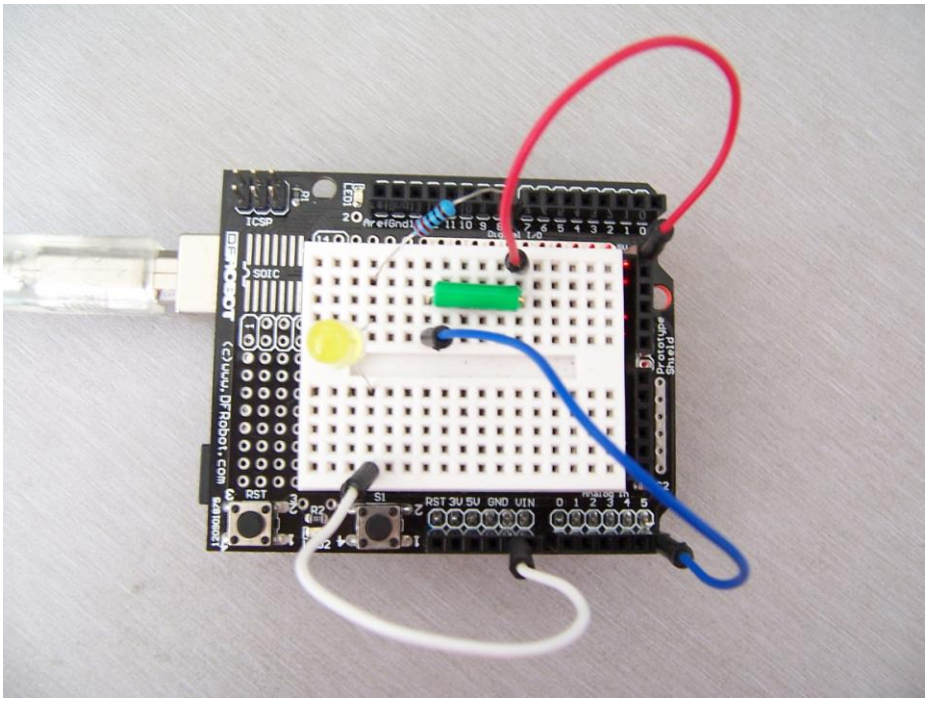


图 5.2 倾斜开关实验接线图

3、实验原理

当金色一端低于水平位置倾斜，开关导通，模拟口电压值为 5V 左右（数字二进制表示为 1023），点亮 led 灯。当银色一端低于水平位置倾斜，开关截止，模拟口电压值为 0V 左右（数字二进制表示为 0），熄灭 led 灯。在程序中模拟口电压值是否大于 2.5V 左右（数字二进制表示为 512），即可知道是否倾斜开关导通了。

4、程序代码

程序代码在 sketch_jun04d 程序文件夹中，双击打开后有一个 sketch_jun04d.pde 文件，双击图标即可打开。打开后我们可以看到这是 arduino 编程软件窗口，上面有本实验的程序代码。大家可以参考 arduino 教程，了解 arduino 语言中的各语句的功能。结合 arduino 教程，理解程序代码。

程序代码如下：

```
void setup()
{
    pinMode(8,OUTPUT);//设置数字 8 引脚为输出模式
}
void loop()
{
    int i;//定义变量 i
    while(1)
    {
        i=analogRead(5);//读取模拟 5 口电压值
        if(i>200)//如果大于 512（2.5V）
        {
            digitalWrite(8,HIGH);//点亮 led 灯
        }
        else//否则
        {
            digitalWrite(8,LOW);//熄灭 led 灯
        }
    }
}
```

5、下载程序

按照 arduino 教程中的程序下载方法将本程序下载到实验板中。

6、程序功能

将程序下载到实验板后大家可以将板子倾斜观察 led 灯的状态。当金色一端低于水平位置倾斜，开关导通，点亮 led 灯；当银色一端低于水平位置倾斜，开关截止，模拟

口电压值为 0V 左右（数字二进制表示为 0），熄灭 led 灯。

掌握本程序后，大家可以按照自己的想法实验，还可以控制其他器件例如蜂鸣器等。

第六节　光控声音实验

一、光敏电阻介绍

1、认识光敏电阻

光敏电阻又称光导管，常用的制作材料为硫化镉，另外还有硒、硫化铝、硫化铅和硫化铋等材料。这些制作材料具有在特定波长的光照射下，其阻值迅速减小的特性。这是由于光照产生的载流子都参与导电，在外加电场的作用下作漂移运动，电子奔向电源的正极，空穴奔向电源的负极，从而使光敏电阻器的阻值迅速下降。实物如图：

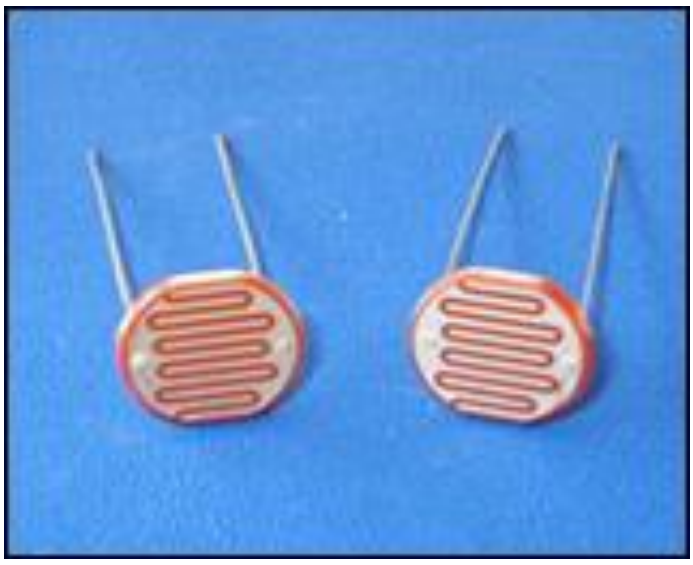


图 6.1 光敏电阻

2、工作原理

光敏电阻的工作原理是基于内光电效应。在半导体光敏材料两端装上电极引线，将其封装在带有透明窗的管壳里就构成光敏电阻，为了增加灵敏度，两电极常做成梳状。在有光照射时入射光强，电阻减小，入射光弱，电阻增大。

3、光敏电阻的连线

光敏电阻在使用中可以直接与所控制器件相连。

二、光控声音实验

1、实验器件

- 光敏电阻：1 个
- 蜂鸣器：1 个

- 多彩面包板实验跳线：若干

2、实验连线

按照 Arduino 教程将控制板、prototype 板子、面包板连接好，下载线插好。光敏电阻的一端插在数字 6 口，另一端与蜂鸣器正极相连，蜂鸣器的负极与 GND 插口相连。如图：

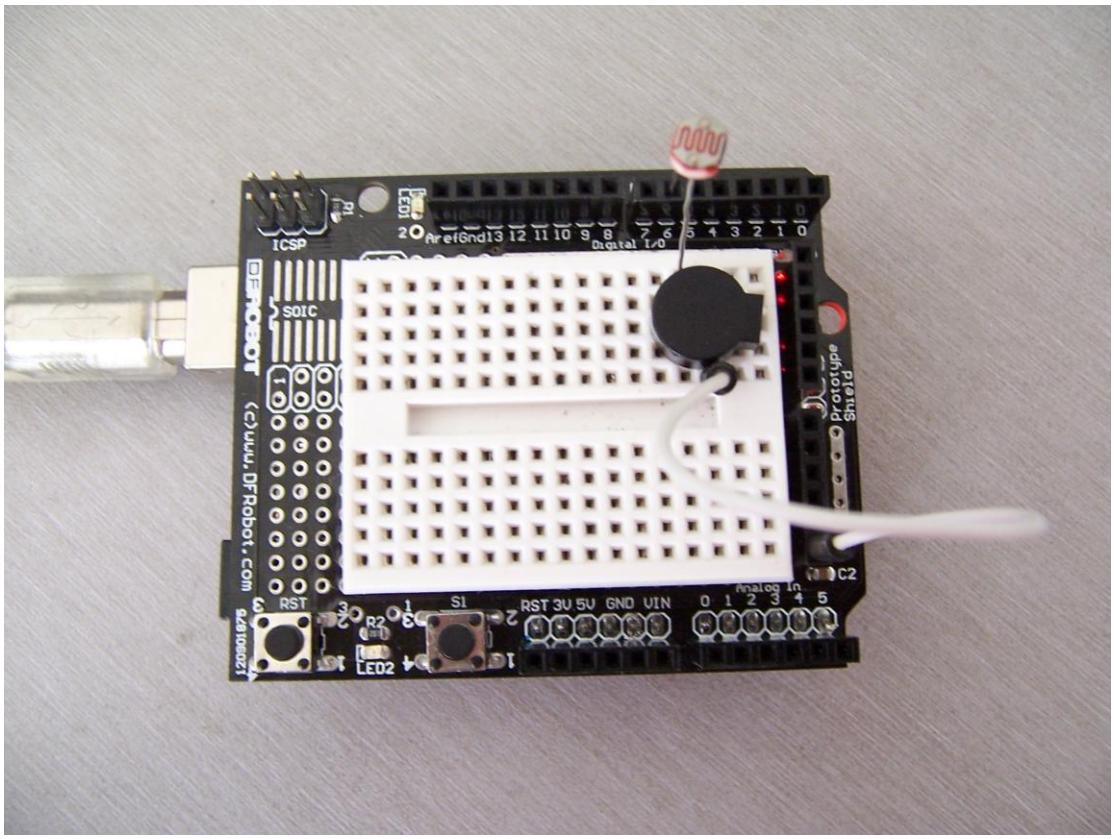


图 6.2 光敏电阻实验接线图

3、实验原理

本程序不用前面几节读取模拟口电压值的方法，直接将光敏电阻插在数字口。程序类似第二节蜂鸣器发声的程序，没有光照时，正常发出声音，但声音特别的小；当有光照时，光敏电阻的阻值减小，所以蜂鸣器两端的电压就会增大，蜂鸣器声音变大。光照越强，电阻越小，蜂鸣器越响。

4、程序代码

程序代码在 sketch_jun17b 文件夹中，双击打开后有一个 sketch_jun17b.pde 文件，双击图标即可打开。打开后我们可以看到这是 arduino 编程软件窗口，上面有本实验的程序代码。大家可以参考 arduino 教程，了解 arduino 语言中的各语句的功能。结合 arduino 教程，理解程序代码。

程序代码如下：

```
void setup()
{
      pinMode(6,OUTPUT);
}
void loop()
{
      while(1)
      {
            char i,j;
       while(1)
       {
          for(i=0;i<80;i++)       //输出一个频率的声音
            {
            digitalWrite(6,HIGH);
            delay(1);
            digitalWrite(6,LOW);
            delay(1);
            }
            for(i=0;i<100;i++)     //输出另一个频率的声音
            {
            digitalWrite(6,HIGH);
            delay(2);
            digitalWrite(6,LOW);
            delay(2);
            }
       }
      }
}
```

5、下载程序

按照 arduino 教程中的程序下载方法将本程序下载到实验板中。

6、程序功能

将程序下载到实验板后，可以用手电筒或其他发光物体照射光敏电阻，可以听到，有光照时蜂鸣器声音更大。

掌握本程序后，大家可以自己动手设计实验，也可以用光敏电阻控制 led 灯亮度。

第七节　火焰报警实验

一、火焰传感器介绍

1、认识火焰传感器

火焰传感器（即红外接收三极管）是机器人专门用来搜寻火源的传感器，本传感器对火焰特别灵敏。实物如图：

图 7.1 红外接收三极管

2、工作原理

火焰传感器利用红外线对火焰非常敏感的特点，使用特制的红外线接受管来检测火焰，然后把火焰的亮度转化为高低变化的电平信号，输入到中央处理器，中央处理器根据信号的变化做出相应的程序处理。

3、火焰传感器的连线

红外接收三极管的短引线端为负极，长引线端为正极。按照下图将负极插到 5V 插口中，然后将正极与 10K 电阻相连，电阻的另一端插到 GND 插口中，最后从火焰传感器的正极端所在列插入一根跳线，跳线的另一端插在模拟口中。如图：

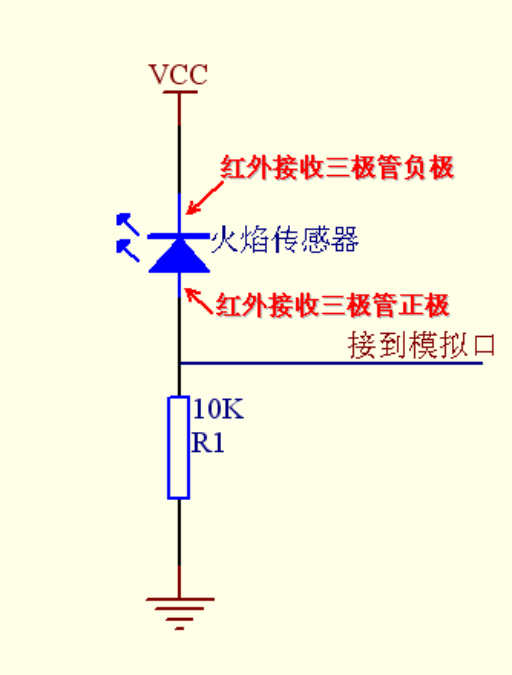


图 7.2 火焰传感器接线图

二、火焰报警实验

1、实验器件

- 火焰传感器：1 个
- 蜂鸣器：1 个
- 10K 电阻：1 个
- 多彩面包板实验跳线：若干

2、实验连线

1）蜂鸣器的连接

首先 按照 Arduino 教程将控制板、prototype 板子、面包板连接好，下载线插好。从实验盒中取出蜂鸣器，按照第二节实验蜂鸣器的连接方法，将蜂鸣器连接到数字第八口。完成蜂鸣器的连接。

2）火焰传感器的连接

从实验盒中取出火焰传感器，按照本节所讲述的火焰传感器的接线方法，将火焰传感器接到模拟 5 口。完成整个实验的连线。如下图：

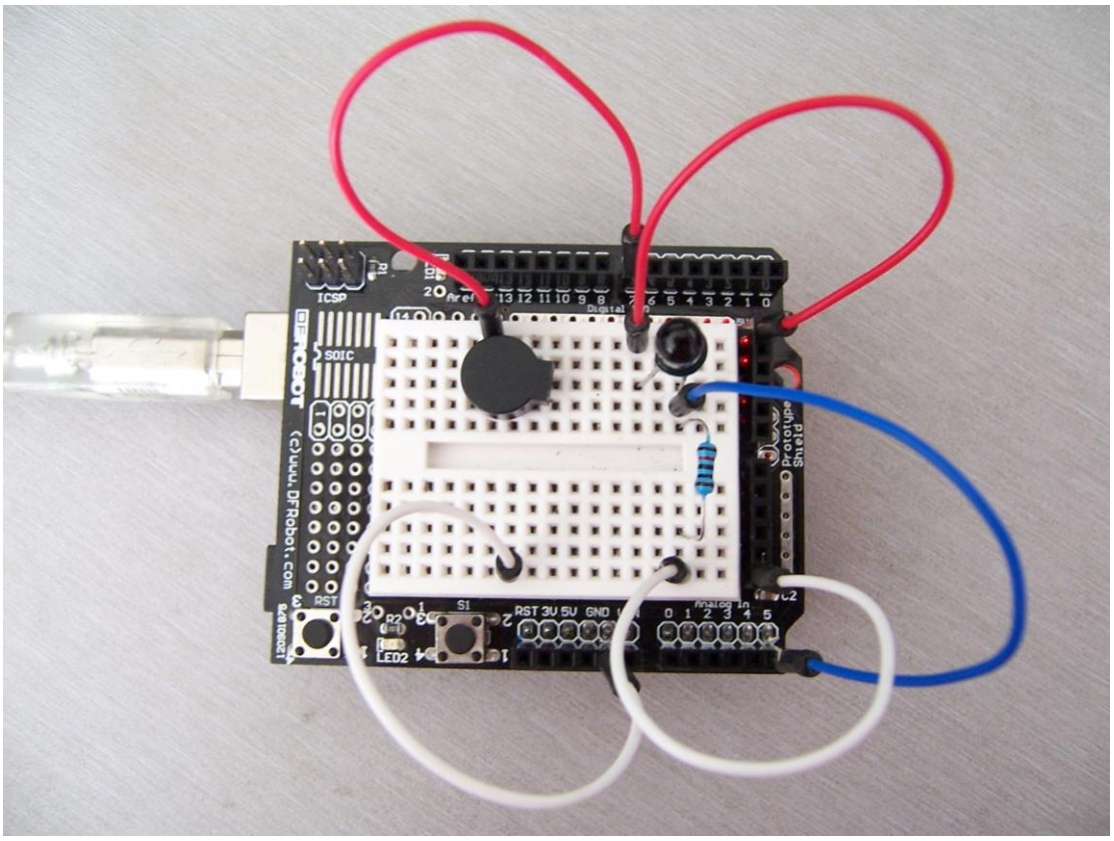


图 7.3 火焰报警接线图

3、实验原理

在有火焰靠近和没有火焰靠近两种情况下，模拟口读到的电压值是有变化的。实际用万用表测量可知，在没有火焰靠近时，模拟口读到的电压值为 0.3V 左右；当有火焰靠近时，模拟口读到的电压值为 1.0V 左右，火焰靠近距离越近电压值越大。

所以在程序一开始，我们可以先存储一个没有火焰时模拟口的电压值 i。接着不断的循环读取模拟口电压值 j、同存储的值做差值 k=j-i、差值 k 与 0.6v 做比较。差值 k 如果大于 0.6V（数字二进制值为 123），则判断有火焰靠近让蜂鸣器发出声音以作报警；如果差值小于 0.6v 则蜂鸣器不响。

4、程序代码

程序代码在火焰报警程序文件夹中，双击打开后有一个 flame 文件，双击打开后会看到 flame.pde。打开后我们可以看到这是 arduino 编程软件窗口，上面有本实验的程序代码。大家可以参考 arduino 教程，了解 arduino 语言中的各语句的功能。结合 arduino 教程，理解程序代码。

程序代码如下：

```
int g;//定义变量 g
void buzzer()//蜂鸣器发出“嘀”声音子程序
{
        for(g=0;g<80;g++)
        {
        digitalWrite(8,HIGH);//发声音
        delay(1);//延时 1ms
        digitalWrite(8,LOW);//不发声音
        delay(1);//延时 ms
        }
}
void setup()
{
    pinMode(8,OUTPUT);//设置数字 8 引脚为输出方式
}
void loop()
{
    char i,j,k;//定义变量
    i=analogRead(5);//读取没有火焰时模拟口的电压值
    while(1)
    {
      j=analogRead(5);//不断的读取模拟口的电压值，时时监测
      k=j-i;//做差值
      if(k>123)//如果差值大于 0.6（0.6 为模拟值，123 为对应的数字二进制值）说明
有火焰
      {
          buzzer();//蜂鸣器发出声音
      }
      else
      {
          digitalWrite(8,LOW);//设置数字 8 口为低电平，蜂鸣器不响
      }
    }
}
```

5、下载程序

按照 arduino 教程中的程序下载方法将本程序下载到实验板中。

6、程序功能

本程序可以模拟在有火焰时报警的情况，在没有火焰时一切正常，当有火焰时立刻报警做出提示。

第八节 抢答器实验

1、实验器件

- 按键：3 个
- 蜂鸣器：1 个
- Led 灯：红色一支、绿色一支
- 220Ω的电阻：2 个
- 多彩面包板实验跳线：若干

2、实验连线

1）按键的连接

从实验盒中取出三个按键。按照前面第四节按键的插法，将三个按键依次插到面包板合适的位置。在每个按键的第 1 引脚所在列插入一根跳线，将跳线的另一端插到 5V 插孔中；在每个按键的第 2 引脚所在列插入一根跳线，将跳线的另一端插入模拟口，按键 1 插到模拟 1 口、按键 2 插到模拟 2 口、按键 3 插到模拟 3 口中，这样按键部分就连接好了。

2）led 灯的连接

从实验盒中取出一个红色的 led 灯和一个绿色的 led 灯。按照第一节中 led 灯的连接方法，将电阻与红色的 led 灯配合连接到数字的第八引脚，将电阻与绿色的 led 灯配合连接到数字的第七引脚。这样就完成了 led 灯的连接。

3）蜂鸣器的连接

从实验盒中取出蜂鸣器。按照第二节试验中蜂鸣器的接法，将蜂鸣器连接到数字的第五引脚。这样我们就将本实验的所有器件连接好了。如下图：

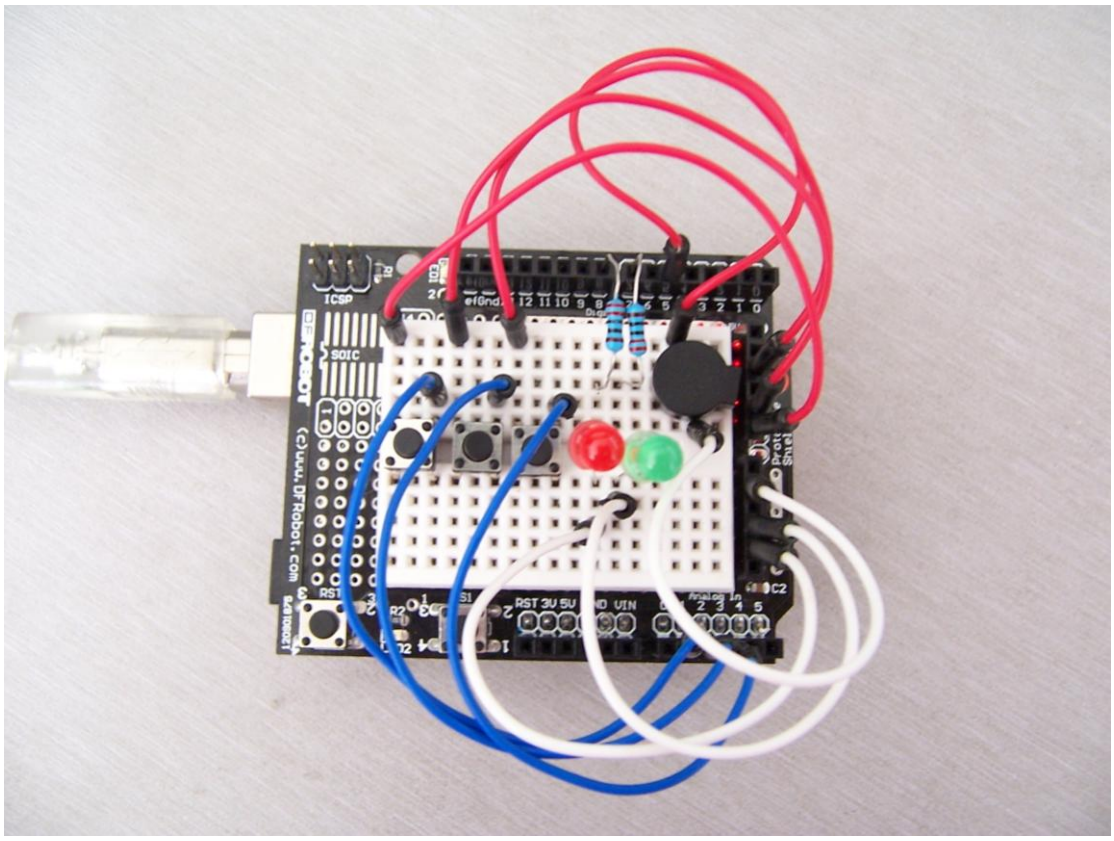


图 8.1 抢答器接线图

3、实验原理

从第四节实验中，我们可以知道按键在保持按下和保持弹起这两种状态的情况下，按键的第 2 引脚和第 4 引脚的电压值是有变化的。所以我们可以依次读取模拟口 1、2、3 的电压值，根据读取的电压值来判断按键是否被按下。

实际用万用表测量可知，当没有按键按下时，模拟口电压值为 0.0V 左右。当有按键按下时 ,模拟口的电压值为 5.0V 左右。所以我们可以认为当模拟口的电压值小于 1V(数字二进制表示为 204）时，没有按键按下，当模拟口的电压值大于 4V（数字二进制表示为 818）时，有按键按下。

按键 1 和 2 是抢答按键，按键 3 是清除按键。如果按键 1 先被按下，蜂鸣器发出提示音，红灯亮，绿灯灭；如果按键 2 先被按下，蜂鸣器发出提示音，绿灯亮，红灯灭；如果按键 3 被按下，蜂鸣器发出提示音，将红灯和绿灯都熄灭。

需要注意的是按键可能存在抖动干扰，为了更加准确的判断是否有按键被按下，在第一次判断有按键按下之后，延时 10ms 的时间躲避抖动，然后进行第二次的判断。另外，模拟口读出的电压值是用二进制表示的，这一点可以翻阅一下 Arduino 使用教

程。

4、程序代码

程序代码在 KBL 文件夹中，双击打开后有一个 KBL.pde 文件，双击图标即可打开。打开后我们可以看到这是 arduino 编程软件窗口，上面有本实验的程序代码。大家可以参考 arduino 教程，了解 arduino 语言中的各语句的功能。结合 arduino 教程，理解程序代码。

程序代码如下：

```
int RedLed=8;//定义第八引脚连接红灯
int GreenLed=7;//定义第七引脚连接绿灯
int i;//定义变量 i
int j=0;//定义变量 j

void buzzer()//蜂鸣器发出“嘀”声音子程序
{
            for(i=0;i<80;i++)
            {
            digitalWrite(5,HIGH);//发声音
            delay(1);//延时 1ms
            digitalWrite(5,LOW);//不发声音
            delay(1);//延时 ms
            }
}
void key_scan()//按键扫描子程序
{
        int key_1,key_2,key_3;//定义变量

        key_1=analogRead(2);//读取模拟第一引脚的电压值
        key_2=analogRead(3);//读取模拟第二引脚的电压值
        key_3=analogRead(4);//读取模拟第三引脚的电压值

        if(key_1<204&&key_2<204&&key_3<204)//如果各按键电压值都小于 204( 即模拟值的 1V )，可以判断没有按键按下
        {
            return;//跳出本子函数
        }
        if(key_1>818)//如果按键 1 的电压值都大于 818（即模拟值的 4V），则可以判断按键 1 被按下
        {
            delay(10);//由于有抖动，所以延时 100ms 再一次判断
            if(key_1>818)//如果按键 1 的电压值都大于 818（即模拟值的 4V），则可以判断按键 1 确实被按下
            {
```

```
                buzzer();//蜂鸣器发出声音
                digitalWrite(RedLed,HIGH);//红灯亮
                digitalWrite(GreenLed,LOW);//绿灯灭
            }
            else //否则认为是抖动干扰，不做任何动作
            {
                return;//跳出本子函数
            }
        }
        if(key_2>818)//如果按键 2 的电压值都大于 818（即模拟值的 4V），则可以判断
按键 2 被按下
        {
            delay(10);//由于有抖动，所以延时 100ms 再一次判断
            if(key_2>818)//如果按键 2 的电压值都大于 818（即模拟值的 4V），则可以判
断按键 2 确实被按下
            {
                buzzer();//蜂鸣器发出声音
                digitalWrite(RedLed,LOW);//红灯灭
                digitalWrite(GreenLed,HIGH);//绿灯亮
            }
            else //否则认为是抖动干扰，不做任何动作
            {
                return;//跳出本子函数
            }
        }
        if(key_3>818)//如果按键 3 的电压值都大于 818（即模拟值的 4V），则可以判断
按键 3 被按下
        {
            delay(10);//由于有抖动，所以延时 100ms 再一次判断
            if(key_3>818)//如果按键 3 的电压值都大于 818（即模拟值的 4V），则可以判
断按键 3 确实被按下
            {
                buzzer();//蜂鸣器发出声音
                digitalWrite(RedLed,LOW);//红灯灭
                digitalWrite(GreenLed,LOW);//绿灯灭
            }
            else   //否则认为是抖动干扰，不做任何动作
            {
                return;//跳出本子函数
            }
        }

}
void setup()
{
    for(i=5;i<=8;i++)
```

```
  {
        pinMode(i,OUTPUT);//将 5~8 引脚设置为输出模式
  }
}
void loop()
{
      while(1)
      {
          key_scan();//循环扫描按键
      }
}
```

5、下载程序

按照 arduino 教程中的程序下载方法将本程序下载到实验板中。

6、程序功能

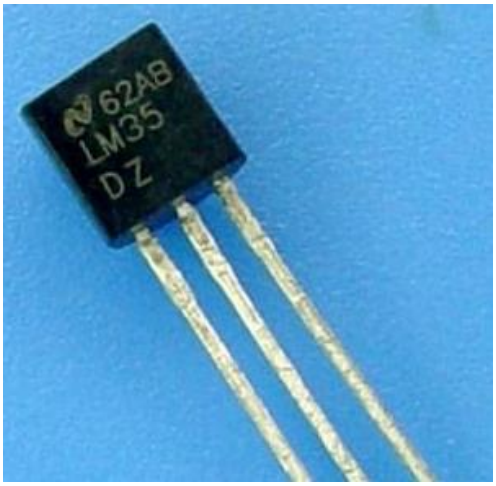
可以完成抢答器功能，两名选手各有一个按键，当比赛开始后进行抢答，谁先按下按键对应的灯就会亮起来。裁判可根据亮灯情况提示参赛选手答题，本次结束后，裁判按下按键 3 清除现在亮灯情况（即将亮灯都熄灭）。

第九节 温度报警实验

一、温度传感器介绍

1、什么是温度传感器？

温度传感器就是利用物质各种物理性质随温度变化的规律把温度转换为电量的传感器。这些呈现规律性变化的物理性质主要有体。温度传感器是温度测量仪表的核心部分，品种繁多。按测量方式可分为接触式和非接触式两大类，按照传感器材料及电子元件特性分为热电阻和热电偶两类。本实验使用的是 LM35 温度传感器。如下图：



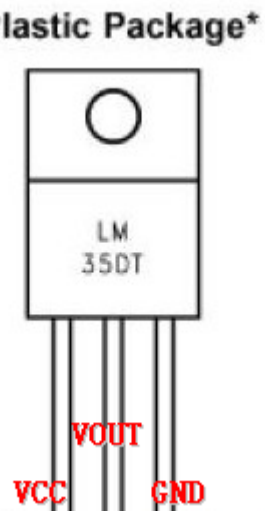
2、工作原理

LM35 温度传感器的输出电压与摄氏温标呈线性关系，0℃，时输出为 0V，每升高 1℃，输出电压增加 10mV。转换公式如下：

$$V_{\text{out\_LM35}}(T) = 10\,\text{mV}/^{\circ}\text{C} \times T^{\circ}\text{C}$$

3、LM35 的连线

LM35 的引脚示意图如下：



从实验盒中将温度传感器拿出来可以看到，温度传感器的一面是平的，另一面是半圆的。将平面对着自己，最左边的是 VCC 引脚（接+5v），中间的为 VOUT（电压值输出引脚，接板子上的模拟引脚），最右边的引脚为 GND 引脚（接板子上的 GND）。三个引脚分别接好就可以用了。

二、温度报警实验

1、实验器件

- LM35 温度传感器：1 个
- LED 灯：红黄绿各 1 个
- 220Ω电阻：3 个
- 多彩面包板实验跳线：若干

2、实验连线

首先将实验板连接好；接着按照 LM35 温度传感器连线方法将其连好，将 VOUT 连接到模拟 0 口；最后按 LED 灯的连接方法，将绿灯通过电阻接到数字 9 口，将黄灯通过电阻接到数字 10 口，将红灯通过电阻接到数字 11 口。这样温度报警实验的电路就连接好了。**（缺少连线图）**

3、实验原理

从 LM35 温度传感器的工作原理可知，温度每升高 1℃，vout 口输出的电压就增加 10MV。根据这一原理程序中实时读取模拟 0 口的电压值，如果电压值在 0.2v~0.3v（温度为 20℃~30℃）绿灯亮，表示这个环境温度是可以接受的温度；如果电压值在 0.3v~0.4v（温度为 30℃~40℃）黄灯亮，表示这个环境温度太高了；如果电压值是出了以上情况的温度即温度低于 20℃和高于 40℃，表示不正常温度，红灯亮以示警告。由于模拟口读出的电压值使用 0~1023 表示的，即 0V 对应 0、5V 对应 1023。程序中

用的都是经过计算后的，例如 0.3V 对应为 61.

4、程序代码

程序代码在温度传感器文件夹中也可以找到。.

程序代码：

```
#define LED_GREEN 9//定义与绿灯连接的引脚
#define LED_YELLOW 10//定义与黄灯连接的引脚
#define LED_RED 11//定义与红灯连接的引脚
void setup()
{
  unsigned char j;
  for(j=9;j<=11;j++)//设置与红绿黄灯连接的引脚为输出模式
  {
    pinMode(j,OUTPUT);
  }
}
void loop()
{
  int i;
  while(1)
  {
    i=analogRead(0);//读取温度传感器电压值
    if(i>41&&i<61)//温度在 20~30 度之间
    {
      digitalWrite(LED_GREEN,HIGH);//绿灯亮
      digitalWrite(LED_YELLOW,LOW);//黄灯灭
      digitalWrite(LED_RED,LOW);//红灯灭
    }
    else if(i>=61&&i<81)//温度在 30~40 度之间
    {
      digitalWrite(LED_YELLOW,HIGH);//黄灯亮
      digitalWrite(LED_GREEN,LOW);//绿灯灭
      digitalWrite(LED_RED,LOW);//红灯灭
    }
    else//温度在 20 度以下和 40 度以上
    {
      digitalWrite(LED_RED,HIGH);//红灯亮
      digitalWrite(LED_YELLOW,LOW);//黄灯灭
      digitalWrite(LED_GREEN,LOW);//绿灯灭
    }
  }
}
```

5、程序功能

将程序下载到实验板，看看那个 LED 灯亮着，就知道你的环境温度是在那个温度段了。

第十节 红外遥控

1、红外接收头介绍

一、什么是红外接收头？

红外遥控器发出的信号是一连串的二进制脉冲码。为了使其在无线传输过程中免受其他红外信号的干扰,通常都是先将其调制在特定的载波频率上,然后再经红外发射二极管发射出去,而红外线接收装置则要滤除其他杂波,只接收该特定频率的信号并将其还原成二进制脉冲码,也就是解调.

二、工作原理

内置接收管将红外发射管发射出来的光信号转换为微弱的电信号 ,此信号经由 IC 内部放大器进行放大，然后通过自动增益控制、带通滤波、解调变、波形整形后还原为遥控器发射出的原始编码，经由接收头的信号输出脚输入到电器上的编码识别电路。

三、红外接收头的引脚与连线

红外接收头有三个引脚如下图：

用的时候将 VOUT 接到模拟口，GND 接到实验板上的 GND,VCC 接到实验板上的+5v。

二、红外遥控实验

1、实验器件

- 红外遥控器：1 个
- 红外接收头：1 个
- LED 灯：1 个

- 蜂鸣器：1 个
- 220Ω电阻：1 个
- 多彩面包线：若干

2、实验连线

首先将板子连接好 ;接着将红外接收头按照上述方法接好 ,将 VOUT 接到模拟 0 口；最后将蜂鸣器接到数字 10 引脚，将红色 LED 灯通过电阻接到数字 11 引脚。这样就完成了电路部分的连接。

3、实验原理

要想对某一遥控器进行解码必须要了解该遥控器的编码方式，这就叫知己知彼，百战不殆 。本产品使用的遥控器的编码方式为：NEC 协议。下面就介绍一下 NEC 协议：

·NEC 协议介绍：特点：（1）8 位地址位，8 位命令位

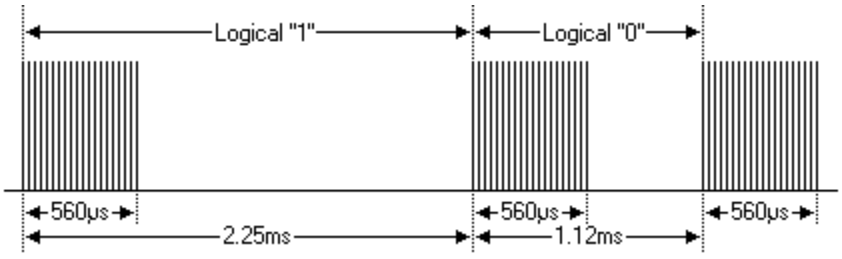
（2）为了可靠性地址位和命令位被传输两次

（3）脉冲位置调制

（4）载波频率 38khz

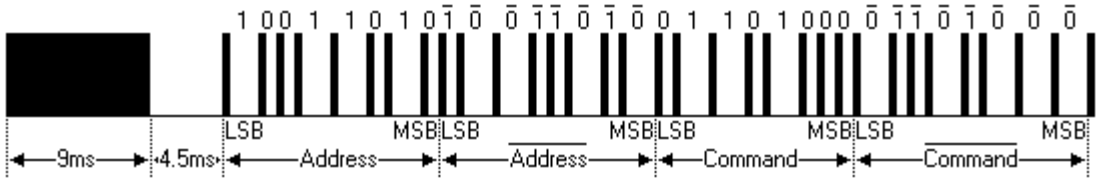
（5）每一位的时间为 1.125ms 或 2.25ms

·逻辑 0 和 1 的定义如下图：



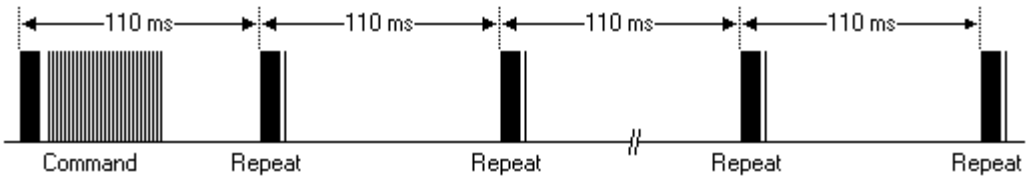
协议如下：

·按键按下立刻松开的发射脉冲：



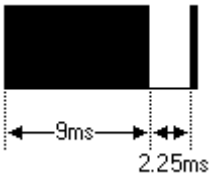
上面的图片显示了 NEC 的协议典型的脉冲序列。注意 :这是首先发送 LSB(最低位) 的协议。在上面的脉冲传输的地址为 0x59 命令为 0x16。一个消息是由一个 9ms 的高电平开始，随后有一个 4.5ms 的低电平，(这两段电平组成引导码) 然后由地址码和命令码。地址和命令传输两次。第二次所有位都取反，可用于对所收到的消息中的确认使用。总传输时间是恒定的，因为每一点与它取反长度重复。如果你不感兴趣，你可以忽略这个可靠性取反，也可以扩大地址和命令，以每 16 位！

·按键按下一段时间才松开的发射脉冲：



一个命令发送一次，即使在遥控器上的按键仍然按下。当按键一直按下时，第一个 110ms 的脉冲与上图一样，之后每 110ms 重复代码传输一次。这个重复代码是由一个 9ms 的高电平脉冲和一个 2.25ms 低电平和 560µs 的高电平组成。

·重复脉冲



本介绍是参考 http://www.sbprojects.com/knowledge/ir/nec.htm

注意：脉冲波形进入一体化接收头以后，因为一体化接收头里要进行解码、信号放大和整形，故要注意：在没有红外信号时，其输出端为高电平，有信号时为低电平，故其输出信号电平正好和发射端相反。接收端脉冲大家可以通过示波器看到，结合看到的波形理解程序。

本实验编程思想

根据 NEC 编码的特点和接收端的波形，本实验将接收端的波形分成四部分：引导码（9ms 和 4.5ms 的脉冲）、地址码 16 位（包括 8 位的地址位和 8 位的地址的取反）、命令码 16 位 (包括 8 位命令位和 8 位命令位的取反)、重复码 (9ms、2.25ms、560us

脉冲组成）。利用定时器对接收到的波形的高电平段和低电平段进行测量，根据测量到的时间来区分：逻辑“0”、逻辑“1”、引导脉冲、重复脉冲。引导码和地址码只要判断是正确的脉冲即可，不用存储，但是命令码必须存储，因为每个按键的命令码都不同，根据命令码来执行相应的动作。设置遥控器上的几个按键 VOL+：控制 LED 灯亮的；VOL-：作为控制蜂鸣器响；

几个按键的命令值

- 红色的电源键： 0xff00；
- VOL+: 0xfe01；
- VOL-: 0xf609；
- 向左的两个三角键：0xfb04；
- 向右的两个三角键：0xf906；

4、程序代码

```
#define BUZZER 10//蜂鸣器
#define LED_RED 11//红灯
#define IR_IN   8   //红外接收

int Pulse_Width=0;//存储脉宽
int   ir_code=0x00;//命令值

void timer1_init(void)//定时器初始化函数
{
  TCCR1A = 0X00;
  TCCR1B = 0X05;//给定时器时钟源
  TCCR1C = 0X00;
  TCNT1 = 0X00;
  TIMSK1 = 0X00;    //禁止定时器溢出中断
}
void remote_deal(void)//执行译码结果函数
{
  switch(ir_code)
  {
  case 0xff00://停止
    digitalWrite(LED_RED,LOW);//红灯不亮
    digitalWrite(BUZZER,LOW);//蜂鸣器不响
    break;
  case 0xfe01://VOL+
    digitalWrite(LED_RED,HIGH);//红灯亮
```

```
      break;
   case 0xf609://VOL-
      digitalWrite(BUZZER,HIGH);//蜂鸣器响
      break;
   }
}
char logic_value()//判断逻辑值“0”和“1”子函数
{
   while(!(digitalRead(8))); //低等待
   Pulse_Width=TCNT1;
   TCNT1=0;
   if(Pulse_Width>=7&&Pulse_Width<=10)//低电平 560us
   {
      while(digitalRead(8));//是高就等待
      Pulse_Width=TCNT1;
      TCNT1=0;
      if(Pulse_Width>=7&&Pulse_Width<=10)//接着高电平 560us
         return 0;
      else if(Pulse_Width>=25&&Pulse_Width<=27) //接着高电平 1.7ms
         return 1;
   }
   return -1;
}
void pulse_deal()//接收地址码和命令码脉冲函数
{
   int i;
   //执行 8 个 0
   for(i=0; i<8; i++)
   {
      if(logic_value() != 0) //不是 0
            return;
   }
   //执行 6 个 1
   for(i=0; i<6; i++)
   {
      if(logic_value()!= 1) //不是 1
            return;
   }
   //执行 1 个 0
   if(logic_value()!= 0) //不是 0
         return;
   //执行 1 个 1
   if(logic_value()!= 1) //不是 1
         return;
```

```
  //解析遥控器编码中的 command 指令
  ir_code=0x00;//清零
  for(i=0; i<16;i++ )
  {
    if(logic_value() == 1)
    {
      ir_code |=(1<<i);
    }
  }
}
void remote_decode(void)//译码函数
{
  TCNT1=0X00;
  while(digitalRead(8))//是高就等待
  {
    if(TCNT1>=1563)  //当高电平持续时间超过 100ms，表明此时没有按键按下
    {
      ir_code = 0xff00;
      return;
    }
  }

  //如果高电平持续时间不超过 100ms
  TCNT1=0X00;

  while(!(digitalRead(8))); //低等待
  Pulse_Width=TCNT1;
  TCNT1=0;
  if(Pulse_Width>=140&&Pulse_Width<=141)//9ms
  {

    while(digitalRead(8));//是高就等待
    Pulse_Width=TCNT1;
    TCNT1=0;
    if(Pulse_Width>=68&&Pulse_Width<=72)//4.5ms
    {
      pulse_deal();
      return;
    }
    else if(Pulse_Width>=34&&Pulse_Width<=36)//2.25ms
    {
      while(!(digitalRead(8)));//低等待
      Pulse_Width=TCNT1;
      TCNT1=0;
      if(Pulse_Width>=7&&Pulse_Width<=10)//560us
      {
```

```
            return;
          }
        }
    }
}
void setup()
{
  unsigned char i;
  pinMode(LED_RED,OUTPUT);//设置与红灯连接的引脚为输出模式
  pinMode(BUZZER,OUTPUT);//设置与蜂鸣器连接的引脚为输出模式
  pinMode(IR_IN,INPUT);//设置红外接收引脚为输入
}
void loop()
{
  timer1_init();//定时器初始化
  while(1)
  {
    remote_decode();   //译码
    remote_deal();     //执行译码结果
  }
}
```

五、程序功能

对遥控器发射出来的编码脉冲进行解码，根据解码结果执行相应的动作。按下前进键红灯亮，松开红灯灭；按下后退键蜂鸣器响，松开蜂鸣器停止响；这样大家就可以用遥控器遥控你的器件了，让它听你的指挥。其它按键的译码方式与这两个键一样，只要在执行译码结果的函数中写上这个按键对应的命令码和要控制的器件的动作即可。