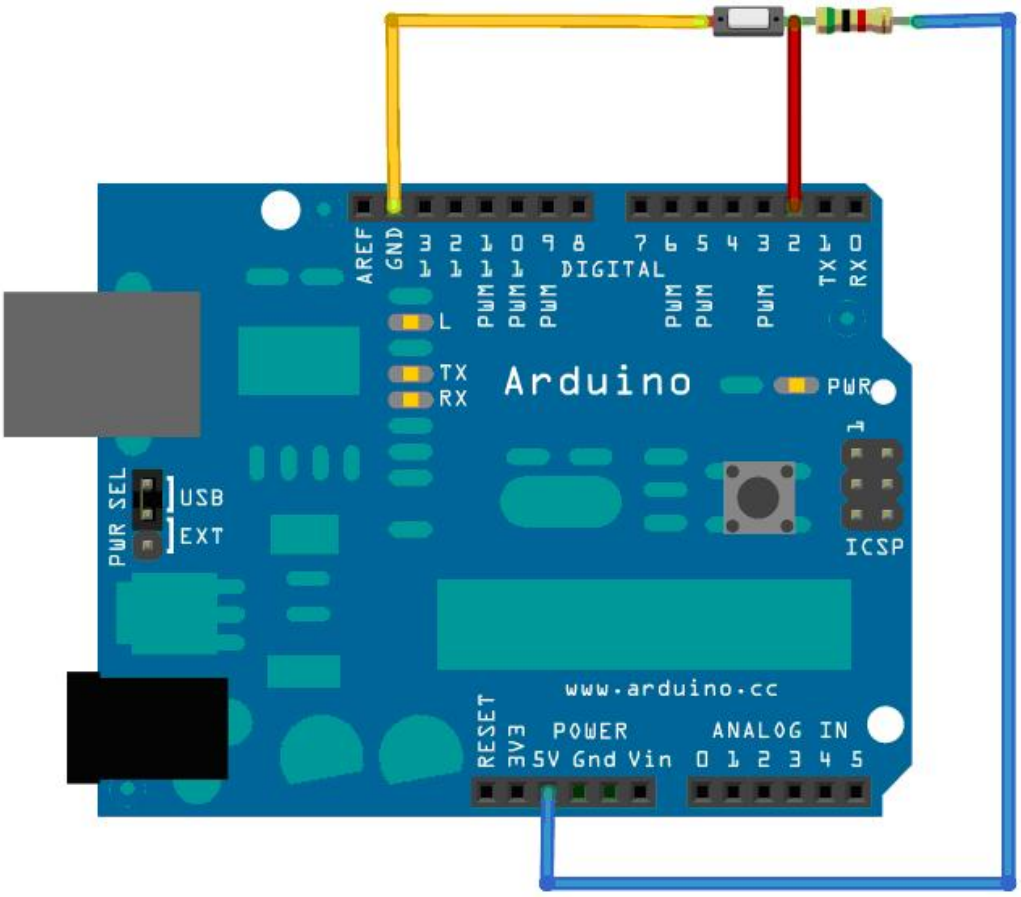
开源传媒原创精品文档系列

作者：晨光熹微
策划：玉非璞

零基础学 Arduino 的十堂课



首发论坛	http://www.52solution.com/bbs/forumdisplay.php?fid=26
微博	开源硬件的星星之火：http://weibo.com/itneers
QQ 群	150183353
邮箱	晨光熹微：51216463@qq.com；玉非璞：Qiang0204@qq.com；
购买	硬件宝库：http://shop67239743.taobao.com/
网址	http://osmt.b2b.youboy.com/spzs.html

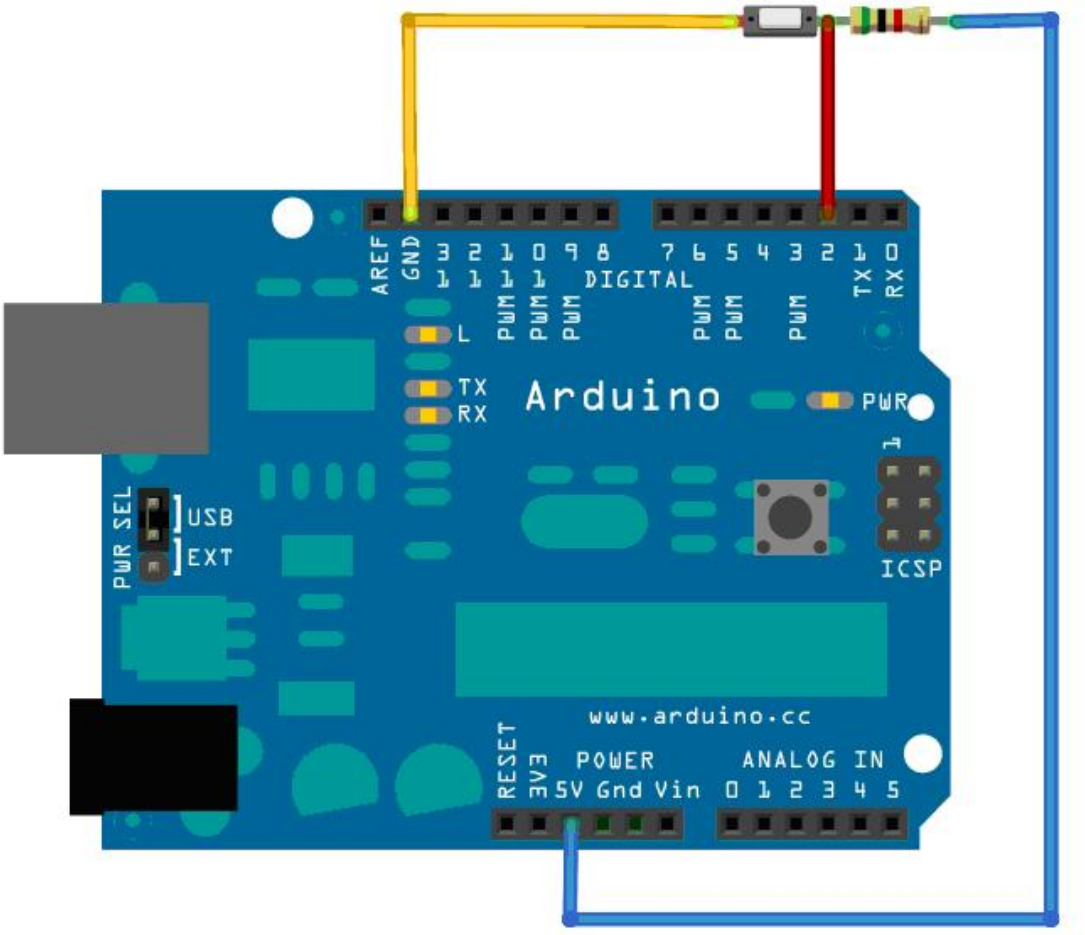
目　录

第一堂课：信号输入

之前的内容都在介绍 I/O 口的输出功能，这次来介绍一下 I/O 口的信号输入
获取引脚是高电平还是低电平的信息需要使用 digitalRead 函数。
函数定义如下：
int digitalRead(uint8_t pin)
函数的参数是所获取信息的引脚号，返回值为引脚的状态。

这里用一个实例说明一下函数的用法，实例的功能是在 Arduino 上外接一个按键，用以控制引脚 13 上的 led 的亮灭（Arduino 板上 13 号引脚带一个 led）

按键占用引脚 2，连接效果如图所示（*为保持按键没有按下时引脚上保持高电平，所以在引脚上加了一个上拉电阻*）



第一堂课接线图

程序如下（直接粘贴到 Arduino 开发环境下就可直接应用）：

```
        void setup()
{
    //设置 13 号引脚为输出
    pinMode(13, OUTPUT);
    //设置 2 号引脚为输入
    pinMode(2, INPUT);
}

void loop()
{
    //判断按键是否按下
    if(LOW = = digitalRead(2))
    {
      //延时去抖
      delay(50);
      if(LOW = = digitalRead(2))
      {
          //点亮 LED
          digitalWrite(13,HIGH);
          while(1)
          {
              //判断是否松开按键
              if(HIGH = = digitalRead(2))
              {
                  //延时去抖
                  delay(50);
                  if(HIGH = = digitalRead(2))
                  break;
              }
          }
          //熄灭 LED
          digitalWrite(13,LOW);
    }
  }
}
```

程序中紫色的部分均调用 digitalRead *函数，在使用* digitalRead *函数前要将引脚置为输入*——pinMode(2, INPUT);

另外在处理按键相应的程序时，为防止按键抖动造成误操作，一般都作一个延时去抖处理

第二堂课：I/O 输入组件

按键作为 I/O 口输入器件在上一次已经看到了使用的效果。其实际上就是用来控制线路的通断，在加上上拉电阻等器件的配合达到改变 I/O 口信号的效果。

倾斜开关也叫滚珠开关、钢珠开关。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，类似于传统的水银开关。如图所示的是较常用的 SW-200D 型倾斜开关。

观察倾斜开关可以收现，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角度低于水平角度时，会产生短时间导通或持续导通<ON>状态 ；而要恢复开路状态<OFF>时开关环境必须为静止，且银色一端角度需低于水平 10 度。

倾斜开关的连接方式与按键相同，按键传递的是按压的信号，而倾斜开关传递的是角度信号。
可以参照上次内容是试试倾斜开关的使用哦

程序和信号输入里的内容类似，在处理倾斜开关时也要考虑去抖的问题。

程序如下（直接粘贴到 Arduino 开发环境下就可直接应用）：

```
        void setup()
{
    //设置 13 号引脚为输出，用于 led 显示
    pinMode(13, OUTPUT);
    //设置 2 号引脚为倾斜开关输入
    pinMode(2, INPUT);
}

void loop()
{
    //判断倾斜开关是否导通
    if(LOW = = digitalRead(2))
    {
      //延时去抖
      delay(50);
      if(LOW = = digitalRead(2))
      {
          //点亮 LED
          digitalWrite(13,HIGH);
          while(1)
          {
```

```
                //判断倾斜开关是否断开
                if(HIGH = = digitalRead(2))
                {
                    //延时去抖
                    delay(50);
                    if(HIGH = = digitalRead(2))
                    break;
                }
            }
            //熄灭 LED
            digitalWrite(13,LOW);
        }
    }
}
```

程序效果是当改变倾斜开关的角度时，LED 会相应的点亮和熄灭

第三堂课：I/O 输出

什么是 Arduino 及它的历史这里就不讲了,想知道的可以去 baidu
本教程使用的是 ArduinoDuemilanove
软件版本是 0021。

在 Arduino 的开发环境中设置串口（Tools—>SerialPort—>*硬件使用的串口号*）
选择正确的 Arduino 板系列型号，（Tools—>Board—>*Arduino Duemilanove or Nano w/ATmege328*）

一般在 C 语言中要求必须有一个主函数，即 main 函数，且只能有一个主函数，程序执行是从主函数开始的。但在 Arduino 中，主函数 main 函数在内部定义了，使用者只需要完成以下两个函数就能够完成 Arduino 程序的编写，这连个函数分别负责 Arduino 程序的初始化部分和执行部分。他们是

- ❑void setup（）
- ❑void loop（）

两个函数均为无返回值的函数，setup()函数用于初始化,一般放在程序开头，主要工作是用于设置一些引脚的输出/输入模式,、初始化串口等，该函数只在上电或重启时执行一次.；loop()函数用于执行程序，loop（）函数是一个死循环，其中的代码将被循环执行，来完成程序的功能。

由于在 Arduino 板上的 pin13 脚连接了一个 LED，我们就使用这个 LED 了解一下 Arduino 引脚的输出。
I/O 的输出实际上就是两步操作，1、设置引脚为输出；2、设置输出高或者输出低
在加上延时就能看到效果了。

程序如下（直接粘贴到 Arduino 开发环境下就可直接应用）：

```
        void setup( )
{

 // 设置引脚 13 为输出，
 //使用函数 pinMode(pin, mode)，
 //     pin 表示 14 个 Arduino 引脚为 0～13，
 //     mode 表示输入或输出，可选参数为 INPUT 戒 OUTPUT

 pinMode(13, OUTPUT);
```

```
}

void loop( )
{

    //设置引脚输出高电平，使用函数 digitalWrite(pin, value)：数字 IO 口输出电平定义
函数，
    //     pin 表示 14 个引脚为 0～13
    //     value 表示输出电平，高电平为 HIGH，低电平为 LOW
    digitalWrite(13, HIGH);

    //延时 1 秒，使用延时函数 delay（ms）
    //     ms 表示延时时间，单位是 ms，1000ms=1s
    delay(1000);



    digitalWrite(13, LOW);     // set the LED off
    delay(1000);               // wait for a second
}
```

选择开发环境中的 Upload（一个向右的箭头）就可以看到 LED 闪烁的效果了

重点：3 个函数

数字 IO 口辎入辎出模式定义函数　　pinMode(pin, mode)

数字 IO 口输出电平定义函数　　digitalWrite(pin, value)

延时函数　　delay（ms）

第三堂课：I/O 输出组建（数码管、蜂鸣器、LED）

Arduino 除了能驱动 LED 之外，还可以通过 I/O 输出驱动蜂鸣器和数码管。

LED（发光二极管），能将电能转化为光能，其也具有单向导电性，反向击穿电压约 5V。它的正向伏安特性曲线很陡，使用时必须串连限流电阻，在 5V 的电路中一般使用 400 欧左右的电阻。
LED 的两根引脚中较长的一根为正极。有两种接法，1、当 led 的正极通过限流电阻与 Arduino 的 I/O 口相连，另一端接地，此时 Arduino 输出 高电平时 led 点亮，输出低电平时 led 熄灭。2、当 led 的负极与 Arduino 的 I/O 口相连，另一端通过限流电阻接 5V 电压，此时 Arduino 输 出低时 led 点亮，输出高电平时 led 熄灭。

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。也可以采用上面说到两种接法，不同的是不需要接限流电阻。

数码管可以看成是多个 led 的集合，led 的公共脚接在一起，若公共脚是正极的称为共阳极数码管，公共脚是负极的称为公阴极数码管。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小数点。

接下来我们使用一个公阳极的数码管循环显示 1 到 8 八个数字。电路连接上数码管的公共引脚接 5V 电压，其他引脚串连限流电阻接到 Arduino 的 I/O 口上。数码管上 7 个短线段分别用 A、B、C、D、E、F、G 表示。如下

```
     a
    ——
f |  g  | b
    ——
e |  d  | c
    ——
```

Arduino 的连接对应关系如下（引脚输出低电亮数码段）
a —— 7 引脚
b —— 6 引脚
c —— 5 引脚
d —— 11 引脚
e —— 10 引脚
f —— 8 引脚
g —— 9 引脚

程序如下（直接粘贴到 Arduino 开发环境下就可直接应用）：

```
        //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;


//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);
}
//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(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(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(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(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(b,HIGH);
}
//显示数字 7
void digital_7(void)
{
  unsigned char j;
  for(j=5;j<=7;j++)
      digitalWrite(j,LOW);
  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);
}
void setup()
{
  int i;//定义变量
  for(i=5;i<=11;i++)
     pinMode(i,OUTPUT);//设置 5～11 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);
  }
}
```

第五堂课：交通灯

通过前面两次 I/O 输出的介绍，这次我们就来实现一个交通灯的例子

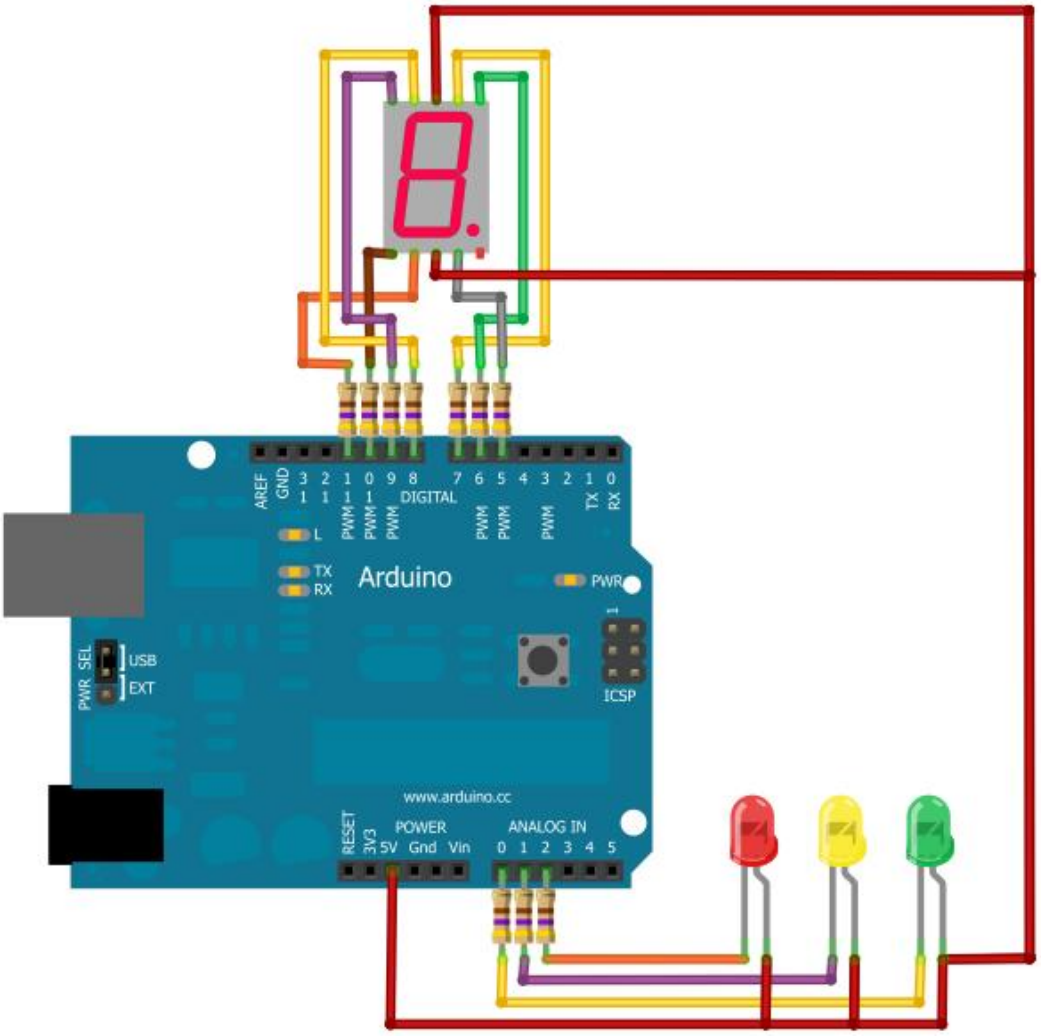
实物连接效果如图所示：

数码管的控制引脚不变，3 个 led 灯连接到模拟端口 0、1、2，分别对应绿、黄、红，引脚输出低点亮 LED。

说明：当 6 个模拟口作为数字口的时候，其对应序号是接着其他的数字口的，即模拟口 0--5 对应就是数字口 14--19。

实例功能如下：

红灯亮 9 秒，接着绿灯亮 9 秒，接着黄灯亮 3 秒，接着红灯亮 9 秒，如此循环。数码管显示当前 LED 灯熄灭剩余秒数



第五堂课接线图

程序如下（直接粘贴到 Arduino 开发环境下就可直接应用，程序中略去了数码管显示 0-9 数字子函数，可参考第四堂课）：

```
        //设置控制各段及 LED 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 ledG = 14;
int ledY = 15;
int ledR = 16;


//引脚设置、初始化
void setup()
{
 int i;//定义变量
 for(i=5;i<=16;i++)
    pinMode(i,OUTPUT);//设置 5～16 引脚为输出模式
}


//程序执行部分
void loop()
{
 while(1)
 {
    //红灯
    digitalWrite(ledR,LOW);
    digitalWrite(ledY,HIGH);
    digitalWrite(ledG,HIGH);

    digital_9();     //显示 9
    delay(1000);     //延时 1s
    digital_8();     //显示 8
    delay(1000);     //延时 1s
    digital_7();     //显示 7
    delay(1000);     //延时 1s
    digital_6();     //显示 6
    delay(1000);     //延时 1s
    digital_5();     //显示 5
    delay(1000);     //延时 1s
    digital_4();     //显示 4
    delay(1000);     //延时 1s
    digital_3();     //显示 3
    delay(1000);     //延时 1s
```

```
    digital_2();    //显示 2
    delay(1000);    //延时 1s
    digital_1();    //显示 1
    delay(1000);    //延时 1s

    //绿灯
    digitalWrite(ledR,HIGH);
    digitalWrite(ledY,HIGH);
    digitalWrite(ledG,LOW);

    digital_9();    //显示 9
    delay(1000);    //延时 1s
    digital_8();    //显示 8
    delay(1000);    //延时 1s
    digital_7();    //显示 7
    delay(1000);    //延时 1s
    digital_6();    //显示 6
    delay(1000);    //延时 1s
    digital_5();    //显示 5
    delay(1000);    //延时 1s
    digital_4();    //显示 4
    delay(1000);    //延时 1s
    digital_3();    //显示 3
    delay(1000);    //延时 1s
    digital_2();    //显示 2
    delay(1000);    //延时 1s
    digital_1();    //显示 1
    delay(1000);    //延时 1s

    //黄灯
    digitalWrite(ledR,HIGH);
    digitalWrite(ledY,LOW);
    digitalWrite(ledG,HIGH);

    digital_3();    //显示 3
    delay(1000);    //延时 1s
    digital_2();    //显示 2
    delay(1000);    //延时 1s
    digital_1();    //显示 1
    delay(1000);    //延时 1s
  }
}
```

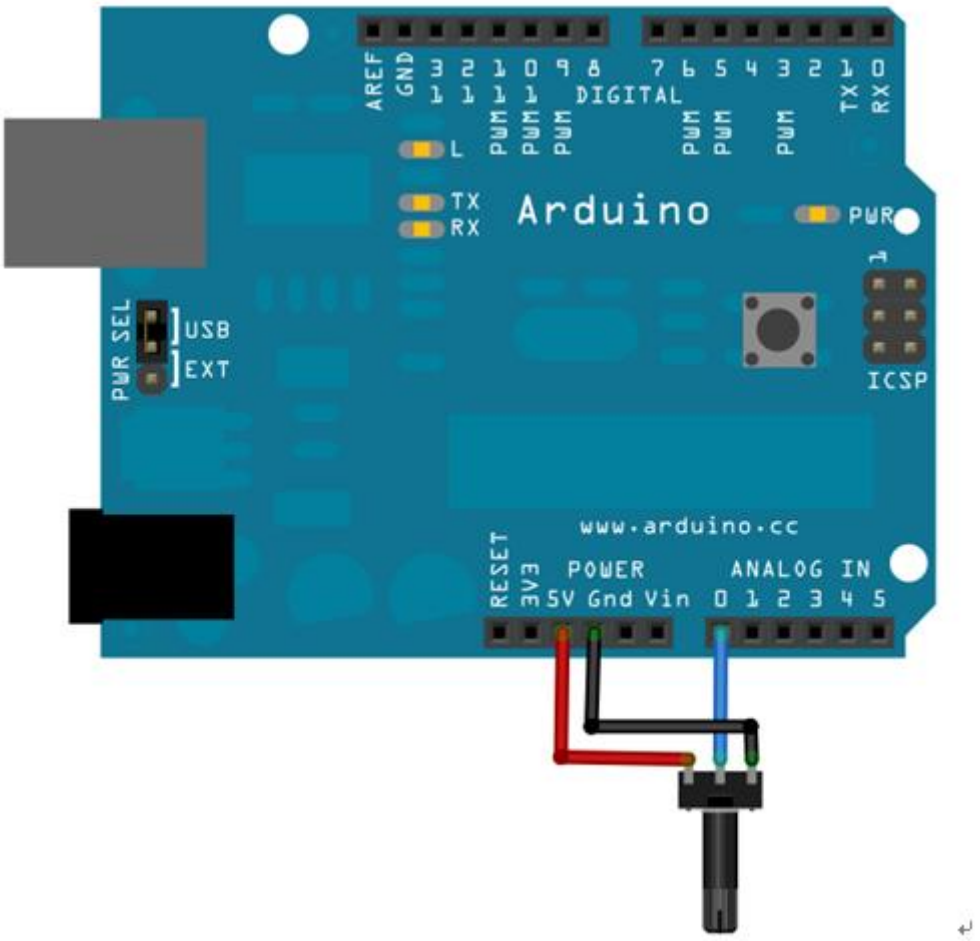
第六堂课：模拟量采集（温度、光线、火焰）

模拟量采集需要用到模拟量器件，这里主要指一些随着环境变化输出电压值随之变化的器件，如火焰传感器、部分温度传感器、可调电阻等等

火焰传感器 和 LM35 温度传感器 的实物图及典型电路连接见附件 1 和 附件 2，这里我们使用可调电阻举例说明一下

Arduino 中模拟量采集要使用 6 个具有 ADC 功能的模拟 I/O 口，使用功能函数 analogRead（）读取引脚的模拟量电压值，每读一次需要花 100 微妙的时间。

将 Arduino 的 0 号模拟口接至可调电位器的中点，电位器另外两端分别连接+5V 和地，USB 口连接至计算机用于传送采样数据。程序设计 Arduino 每 1 秒进行一次 A/D 转换，并将结果传给计算机。



第六堂课接线图

程序如下（直接粘贴到 Arduino 开发环境下就可直接应用）：

```
        void setup()
{
 //设置串口波特率为 9600bps
 Serial.begin(9600);
}

void loop()
{
 //延时 1 秒
 delay(1000);
 //进行 A/D 转换并传输数据
 Serial.print(analogRead(0), DEC);
}
```

第七堂课：串口（串口的使用）

Arduino 上的串口占用的是引脚 0 和引脚 1，对于 Arduino 来说，这两个引脚的任务太重了，下载程序用它，和计算机通信用它、与别的器件进行串行通信也用它。好在使用上还是很方便的。
咱们从输出和输入两方面来说 Arduino 串口的使用

1、输出

Arduino 的输出基本就用两个函数 print 和 println，区别在于后者比前者多了回车换行
下面贴一段“Hello Arduino”的代码，各位可以把代码运行的效果截图贴出来。

```
        void setup()
{
 Serial.begin(9600);
}



void loop()
{

    Serial.println("Hello Arduino");
    //或者  Serial.print("Hello Arduino");
    delay(5000);//延时 5 秒
}
```

在 Arduino 开发环境下带有 Serial Monitor 按钮，可以直接看到效果
**** Serial.begin(9600);*的作用是设置串行波特率，这个设置要与 Serial Monitor 界面中的设置相同* ****

2、输入

串行数据的输入相对要麻烦点，多了一步判断是否有数据收到的操作，使用函数 Serial.available(),之后使用函数 Serial.read()提取收到的数据。
这里把上面的函数做一个调整，只有在收到 S 后才发送“Hello Arduino”

程序如下（直接粘贴到 Arduino 开发环境下就可直接应用）

```
        void setup()
{
    Serial.begin(9600); // 9600 bps
}
```

```
void loop()
{
 if ( Serial.available())
 {
    if('S' == Serial.read())
    {
       Serial.println("Hello Arduino");
       }
  }
}
```

重点：5 个函数

```
//初始化串口
Serial.begin()

//串口发送数据
Serial.println()
Serial.print()

//串口接收数据
Serial.available()
Serial.read()
```

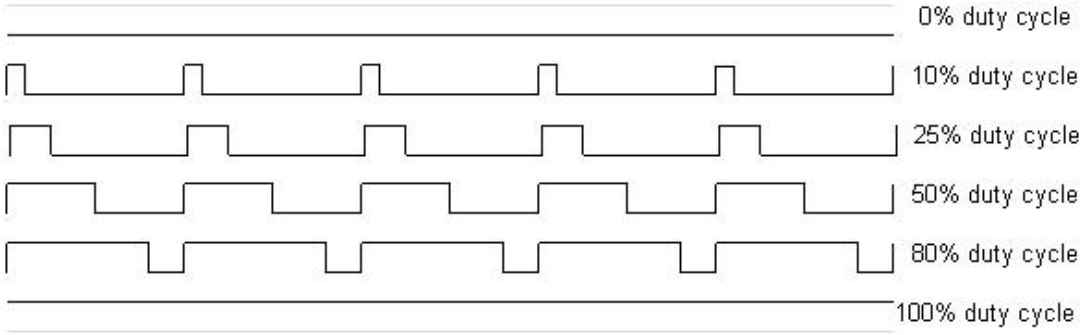
第八堂课：PWM

最近做一个东西用到了 avr 单片机管脚的 pwm，以前还真没仔细琢磨过 pwm，这回碰上了，发现这个 pwm 还真不简单。

PWM 是啥玩意儿？

PWM 是“怕玩命”的缩写，英文写法是“Pulse-width modulation”，也有些外行人士把它翻译成“脉冲宽度调制”。Arduino 有很多种版本，这篇文章里是以 ATmega168 为例，有用过其他型号的兄弟请补充。

PWM 是用占空比不同的方波，来模拟“模拟输出”的一种方式。靠，这个太拗口了，简而言之就是电脑只会输出 0 和 1，那么想输出 0.5 怎么办呢？于是输出 01010101….，平均之后的效果就是 0.5 了。早这么说就了然了嘛。



PWM 有神马作用？

举几个例子说明：

1.通过简单的滤波电路，就可以生成真正的模拟输出量；
2.控制灯光亮度，调节电机转速；请注意这和 1 不是重复的，因为不需要滤波就可以实现
3.控制舵机角度，这个请参考 Arduino 开发板实验三：舵机控制
4.输出信号，例如接喇叭的时候可以发声

如何产生 PWM？

Arduino 有三种方式可以产生 PWM。第一种：

用 analogWrite(pin, val)命令

其中 pin 是腿的编号，传说中只能用 3,5,6,9,10,11 这几条；val 是 0~255 的整数值，对应电压从 0 到+5V。注意，那几个脚的编号，指的 是 ATmega168 的 pin 编号，Arduino 的板子会用这几个管脚支持更多路的 PWM 输出，例如我的 Arduino Mega168 就支持 0~13 共 14 个 PWM 输出。

具体的使用可以看下面的示例代码：

```
        int pin = 8; //0~13

void setup()
{
    pinMode(pin, OUTPUT);
}

void loop()
{
    analogWrite(pin, 128);
    delay(500);
}
```

这种方式产生的方波周期大概是 20ms 左右（50Hz），不需要占用额外的 cpu 命令时间。据说 99%的同学看到这里就可以下课了，技术宅请继续看第二种方式：

手动用代码实现 PWM

```
        int pin = 38;  //这个可以随意点

void setup()
{
    pinMode(pin, OUTPUT);
}
void loop()
{
 digitalWrite(pin, HIGH);
 delayMicroseconds(100);
 digitalWrite(pin, LOW);
 delayMicroseconds(1000 - 100);
}
```

上面这段代码会产生一个 PWM=0.1 的，周期为 1ms 的方波（1000Hz），这种方式的优缺点很明显：

1、PWM 的比例可以更精确；
2、周期和频率可控制；

3、所有的 pin 脚都可以输出，不局限于那几个脚；
4、缺点：CPU 干不了其他事情了；

好吧，缺点只有一个，却非常致命，以至于上面这些基本都是废话。但是对于周期比较大的 PWM，可以用算法模拟 CPU 的多任务系统，从而在输出 PWM 的同时做点兼职。

那么能不能既调节 PWM 的频率和周期，又不要占用额外的 CPU 时间呢？请看第三种方式：

使用 PWM 寄存器

ATmega168 有三个时钟，名字分别叫 Timer0, Timer1 和 Timer2。每个时钟都使用了两个寄存器，其中一个是设定值例如 128，另一个则从 0 开始不断递增，到 1024 之后溢出回到 0。那么当两 个值相同的时候，Timer 就会把某个管脚反相。不同的 Timer 之间频率是相同的，占空比则根据设置值不同。

占空比有了，那么周期怎么控制呢？有一种叫做时钟控制器的东东，这个控制器可以设置周期为 CPU 周期的某个倍数，例如 1,8,64,256,1024 等等，Timer0 和 Timer1 共用一个控制器，Timer2 和它们是独立的。

Atmega 168/328 的时钟们

ATmega328P 有三个时钟，Timer0，Timer1 和 Timer2。每个时钟都有两个比较寄存器，可以同时支持两路输出。其中比较寄存器用于控 制 PWM 的占空比，具体的原理等会儿会介绍。大多数情况下，每个时钟的两路输出会有相同的频率，但是可以有不同的占空比（取决于那两个比较寄存器的设置）

每个时钟都有一个“预定标器”，它的作用是设置 timer 的时钟周期，这个周期一般是有 Arduino 的系统时钟除以一个预设的因子来实现的。这个因子一 般是 1,8,64,256 或 1024 这样的数值。Arduino 的系统时钟周期是 16MHz，所以这些 Timer 的频率就是系统时钟除以这个预设值的标定 值。需要注意的是，Timer2 的时钟标定值是独立的，而 Timer0 和 Timer1 使用的是相同的。这些时钟都可以有多种不同的运行模式。常见的模式包 括“快速 PWM”和“相位修正 PWM”，这两种 PWM 的定义也会在后面解释。这些时钟可以从 0 计数到 255，也可以计数到某个指定的值。例如 16 位 Timer1 就可以支持计数到 16 位（2 个字节）。

除了比较寄存器外，还有一些其他的寄存器用来控制时钟。例如 TCCRnA 和 TCCRnB 就是用来设置时钟的计数位数。这些寄存器包含了很多位（bit），它们分别的作用如下：

脉冲生成模式控制位（WGM）：用来设置时钟的模式
时钟选择位（CS）：设置时钟的预定标器
输出模式控制位（COMnA 和 COMnB）：使能/禁用/反相 输出 A 和输出 B
输出比较器（OCRnA 和 OCRnB）：当计数器等于这两个值时，输出值根据不同的模式进行变化。不同时钟的这些设置位稍有不同，所以使用的时候需要查一下资料。其中 Timer1 是一个 16 位的时钟，Timer2 可以使用不同的预定标器。

文中实现 pwm 的例子，请点击全文下载查看。
地址：http://www.52solution.com/bbs/viewthread.php?tid=1736&extra=page%3D2

第九堂课：模拟量输出

Arduino 通过 PWM 的方式在引脚上输出一个模拟量，较多的应用在 LED 亮度控制、电机转速控制等方面。PWM 方式是通过对一系列脉冲的宽度进行调制，来等效的获得所需要的波形或电压。

关于 PWM 可以参考一下 nine_09y 的帖子 PWM 的秘密 。

在 Arduino 中应用 **analogWrite** 函数实现 PWM 输出。
在 Arduino 中执行该操作后，应该等待一定时间后才能对该引脚进行下一次的操作。Arduino 中的 PWM 的频率大约为 490Hz.。该函数支持以下引脚:3, 5, 6, 9, 10, 11。

我们可以在 PWM 支持的引脚上连接一个 led，使用 analogWrite 实现一个 led 逐渐变亮又逐渐熄灭的效果。

程序如下（这里我们应用的是引脚 5，程序直接粘贴到 Arduino 开发环境下就可直接应用）

```
        void setup()
{
  pinMode(5, OUTPUT);
}

void loop()
{
    //led 渐亮
    for(int i = 0; i<255 ; i++)
    {
          analogWrite(5,i);
          delay(100);
    }

    //led 渐灭
    for( i = 255; i>0 ; i--)
     {
          analogWrite(5,i);
         delay(100);
    }
}
```

****

重点

analogWrite

第十堂课：红外遥控器

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.

红外接收头如附件中所示：

这里举一个 Arduino 接受遥控器信号的例子。红外接受头占用 Arduino 的数 8。
（也可以选择 nine_09y 的帖子 Arduino 电子积木之红外发射接收编解码中的模块)

先来看看遥控器的编码方式,
遥控器发出的二进制脉冲码载波频率是 38kHz，采用脉冲宽度调制，每一位的时间为 1.125ms 或 2.25ms ， 逻辑 0 和逻辑 1 的定义如附件中图 2 所示。

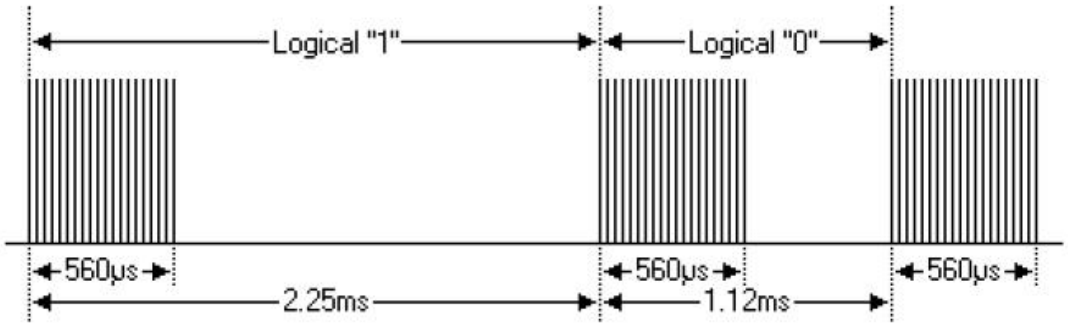


图 2：遥控器的编码方式

而一个消息是由一个 9ms 的高电平开始，随后有一个 4.5ms 的低电平,之后就是信息码了.

图：遥控器

例子接收图中遥控器的 VOL+按钮的信号(信息码 0xfe01),然后让引脚 13 上的 LED 闪烁一下.

程序如下（直接粘贴到 Arduino 开发环境下就可直接应用）：

```
        #define LED_RED 13//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


//定时器初始化函数
void timer1_init(void)
{
 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
}


//执行译码结果函数
void remote_deal(void)
{
 switch(ir_code)
 {
 case 0xfe01://VOL+
    digitalWrite(LED_RED,HIGH);//灯亮
    delay(500);
    digitalWrite(LED_RED,LOW);//灯不亮
    break;
 }
}


//判断逻辑值“0”和“1”子函数
char logic_value()
{
 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;
}


//接收命令码脉冲函数
void pulse_deal()
{
 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;


  ir_code=0x00;//清零
  for(i=0; i<16;i++ )
  {
     if(logic_value() == 1)
     {
       ir_code |=(1<<i);
     }
  }
}


//译码函数
void remote_decode(void)
{
  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(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
  }
}
```

关于红外遥控的学习,个人建议先用示波器捕捉遥控器发出的二进制编码,通过捕捉到的内容能够解读出按键的信息码,然后再认真看一下程序中的译码子函数

红外遥控的程序调起来可能会比较麻烦,可能需要反复的调试几次
需要多些耐心

练成：环境监视器（温度，湿度，光照，下雨指数）

这里使用的温湿度传感器是 SHT1X 温湿度传感器

如图：

图：温湿度传感器

SHT1x 温湿度传感器是瑞士 Sensirion 公司推出的单片数字温湿度集成传感器。采用 CMOS 过程微加工专利技术（CMOSens technology），确保产品具有极高的可靠性和出色的长期稳定性。该传感器由 1 个电容式聚合体测湿元件和 1 个能隙式测温元件组成，并与 1 个 14 位 A/D 转换器以及 1 个 2-wire 数字接口在单芯片中无缝结合，使得该产品具有功耗低、反应快、抗干扰能力强等优点。在对环境温度与湿度测量要求高的情况下使用，该产品具有极高的可靠性和出色的稳定性。其技术规格如下：

❑全部校准，数字输出；
❑接口简单（2-wire）,响应速度快；
❑超低功耗，自动休眠；
❑出色的长期稳定性；
❑超小体积（表面贴装）；
❑湿度范围 0—100%RH，温度范围-40℃—128.8℃
❑测湿精度±4.5%RH，测温精度±0.5℃（25℃）
❑模块尺寸：32X17mm

传感器采用 2-wire 接口，数字输出，所以需要占用两个数字口

举个应用的例子：（例子中占用 Arduino 的数字口 9（接 SHT1x 温湿度传感器的 SCK）和数字口 10（接 SHT1x 温湿度传感器的 DATA））

代码如下：

```
#include <SHT1x.h>
```

```
#define dataPin  10
#define clockPin 9

//定义 SHT1x 类的对象 sht1x
SHT1x sht1x(dataPin, clockPin);

void setup()
{
   Serial.begin(9600);                  //  波特率 9600 bps
}

void loop()
{
          float temp_c;                              //定义温度值变量
          float humidity;                            //定义湿度值变量

          //  读取温湿度值
           temp_c = sht1x.readTemperatureC();
           humidity = sht1x.readHumidity();

          //通过串口输出温度值
          Serial.print("Temperature: ");
          Serial.print(temp_c);

           //通过串口输出湿度值
          Serial.print("      Humidity: ");
          Serial.print(humidity);
          Serial.println("%");

          //2 秒采样一次
          delay(2000);

}
```

//

需要定义一个 **SHT1x 的对象 sht1x**。

获取温度值使用函数 **sht1x.readTemperatureC()**
获取湿度值使用函数 **sht1x.readHumidity()**

另外还可以直接获取华式温度，使用函数 sht1x.readTemperatureF()

作业一：掷骰子

一、设计思路

掷骰子，主要是骰子点数的随机性。应该引用 random 函数。又因为筛子的点数是 1~6,那么最后确定用到的函数应该是 random(min,max)。

题目中给出的是使用 led 灯，用六个 led 灯来表示不同的数字，当然需要上拉电阻避免超过 led 灯的最大电流。

同样还有倾斜开关。利用晃动时不同的角度里面的滚珠的滚动来形成不同的电压值因此而触发电路，通过读取倾斜开关的金色端得电压值来确定灯得亮灭。

由于六个 led 灯形成的点数不够直观，我又加上了数码管来显示，同样也需要上拉电阻配合使用，这样更清楚的显示。

蜂鸣器是因为发出的声音间隔不一样，形成的频率就会不一样，还有可以通过循环的次数来解决题目中蜂鸣器的要求。

二、物品清单

面包板、扩展板和控制板的组合版，USB 接线，导线，led 灯，数码管，倾斜开关，蜂鸣器，220 欧姆电阻。

三、照片

（led 灯从左到右分别代表数字是 1~6，接的 IO 口 8~13，数码管接的 IO 口是 0~6，蜂鸣器的 IO 口是 7）

（最后一个随机数是 3，那么数码管显示为 3）

四、代码

```
int buzzer=7;//蜂鸣器正极是数字端 7
int a=0;
int b=1;
int c=2;
int d=3;
int e=4;
int f=5;
int g=6;//因为 dp 脚一直是灭，所以直接高电平
void buzzerZ()//蜂鸣器发出“嘀”声音子程序
{       int i;
          for(i=0;i<=2;i++)      //响三声，在最后一个随机数字出来时候
          {
          digitalWrite(buzzer,HIGH);//发声音
          delay(125);//延时 125ms
```

```
            digitalWrite(buzzer,LOW);//不发声音
            delay(125);//延时 125ms，因为说是 4Hz 的频率
            }
}
//显示数字 1
void digital_1(void)
{
  digitalWrite(c,LOW);//给数字 5 引脚低电平，点亮 c 段
  digitalWrite(b,LOW);//点亮 b 段
  digitalWrite(f,HIGH);
  digitalWrite(d,HIGH);
  digitalWrite(a,HIGH);
  digitalWrite(g,HIGH);
  digitalWrite(e,HIGH);

}
//显示数字 2
void digital_2(void)
{
  digitalWrite(b,LOW);
  digitalWrite(a,LOW);
  digitalWrite(g,LOW);
  digitalWrite(e,LOW);
  digitalWrite(d,LOW);
  digitalWrite(c,HIGH);
  digitalWrite(f,HIGH);
}
//显示数字 3
void digital_3(void)
{
  unsigned char j;
  digitalWrite(g,LOW);
  digitalWrite(d,LOW);
  digitalWrite(a,LOW);
  digitalWrite(b,LOW);
  digitalWrite(c,LOW);
  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(a,HIGH);
    digitalWrite(e,HIGH);
    digitalWrite(d,HIGH);
}
//显示数字 5
void digital_5(void)
{
    digitalWrite(a,LOW);
    digitalWrite(f,LOW);
    digitalWrite(g,LOW);
    digitalWrite(c,LOW);
    digitalWrite(d,LOW);
    digitalWrite(b,HIGH);
    digitalWrite(e,HIGH);
}
//显示数字 6
void digital_6(void)
{
    digitalWrite(a,LOW);
    digitalWrite(f,LOW);
    digitalWrite(g,LOW);
    digitalWrite(d,LOW);
    digitalWrite(c,LOW);
    digitalWrite(e,LOW);
    digitalWrite(b,HIGH);
}
void digital(void)//全灭
{
    digitalWrite(a,HIGH);
    digitalWrite(f,HIGH);
    digitalWrite(g,HIGH);
    digitalWrite(d,HIGH);
    digitalWrite(c,HIGH);
    digitalWrite(e,HIGH);
    digitalWrite(b,HIGH);
}
void setup()
{
        int i,j;
        pinMode(buzzer,OUTPUT);
        for(i=8;i<=13;i++)
    {
```

```
                pinMode(i,OUTPUT);//将 8~13 引脚设置为输出模式
            }
            for(j=0;j<=6;j++)
              pinMode(j,OUTPUT);//设置 0~6 引脚为输出模式
        }
        void loop()
        {
            int i,j,k,l,m;//定义变量 i,j,k,l,m
            while(1)
            {
            // i=analogRead(5);//读取模拟 5 口电压值
                for(k=1;k<=99;k++)//因为从每 10ms 到 1s 产生的随机数，需要 100 秒，
所以要循环 100 次，但是最后一次需要定下来随机数，不再改变，所以要单独写
                {   for(l=10;l<1000/l;l=l+10)//因为如果是 10ms 产生一次，那么 1s 需要
50 次，那么按此累加的时候循环的次数是 1000/周期
                  {
                    digital();
                   i=analogRead(5);//读取模拟 5 口电压值(倾斜开关)
                    if(i>206)//如果大于 206（1.25V）
                  {
                    j=random(8,13);
                    digitalWrite(j,HIGH);//点亮 led 灯
                     delay(10*k);//间隔时间
                   // else//否则
                  //j=random(8,13);
                     digitalWrite(j,LOW);//熄灭 led 灯
                  }
                  }
              }
                  delay(1000);
                  m=random(8,13);
                  digitalWrite(m,HIGH);//最后一次的随机数字不再灭（如果重新摇骰
子，需要按键 reset）
                  buzzerZ();//蜂鸣器
                  if(m==13)//根据灯亮的情况来显示数码管的数字
                  { digital_1();}//数字 1
                else if(m==12)
                {digital_2();}
                else if(m==11)
                {digital_3();}
                else if(m==10)
                {digital_4();}
                else if(m==9)
                {digital_5();}
```

```
            else if(m==8)
            {digital_6();}
                while(1){}
        }
    }
```

因为题目要求的最开始 10ms 出一个随机数 1s 之后变换为 20ms 出现，一直到 1s 中出现最后一个随机数后确定点数的显示。那么应该用循环来实现。从最开始到结束需要 1 分多钟。

视频：http://v.youku.com/v_show/id_XMjg2MjAwOTc2.html

作业二：蜂鸣器唱歌

【Arduino 作业】喇叭、蜂鸣器唱歌
设计思路 ：

要生产音频脉冲，只要算出某一音频的周期(1/频率)，可以利用定时器计时的方式得到此频率的脉冲。而 Arduino 平台“封装”了新的数字输出函数 tone()。更简易的实现喇叭和蜂鸣器唱歌。
utone(pin, frequency)，Arduino 会向指定 pin 发送制定频率的方波，执行 noTone()函数来停止。
utone(pin, frequency, duration 方法多了一个参数，代表发送方波持续的时间，到时自动停止发送信号，就不需要 noTone()函数。

利用 tone()函数播放音乐，只需要查表了解各个音符对应的频率，还要求个人稍微能看懂音乐谱子的节拍。本人也是上网找了谱子后，慢慢请教慢慢翻译的，都怪小学没有学好音乐课啊。惭愧……

物料清单 ：
Arduino 328 控制板 1 块
8Ω 0.5W 的喇叭（或者蜂鸣器） 1 个（ATmega328 的驱动能力足够，直接拉电流就 ok!）
12Ω 电阻(限流) 1 个
代码

```
/*
* Copyright (c) 2011,个人学习共享
* All rights reserved.
*
* 文件名称：music.pde
* 文件标识：
* 摘　　要：基于 Arduino 328 控制板+Speaker，播放儿歌《小星星》和《国际歌》背景音乐
*
* 当前版本：1.0
* 作　　者：张嘉
* 完成日期：2011 年 7 月 16 日
*
*/
int song[] = {

  /* 儿歌《小星星》*/
  277,277,415,415,466,466,415,
  370,370,330,330,311,311,277,
  415,415,370,370,330,330,311,
  415,415,370,370,330,330,311,
  277,277,415,415,466,466,415,
```

```
    370,370,330,330,311,311,277,

    /*《国际歌》*/
    370,494,466,554,494,370,311,415,330,
    415,554,494,466,415,370,330,311,
    370,494,466,554,494,370,311,415,330,
    415,554,494,466,554,659,494,
    622,554,466,415,466,494,415,466,370,
    370,330,370,415,415,554,494,466,
    554,554,466,370,370,330,370,622,494,
    415,466,494,466,554,494,415,370,
    622,554,494,370,311,415,330,
    554,494,466,415,370,
    370,622,554,370,494,466,
    466,415,415,415,554,554,
    622,554,494,370,311,415,330,330,
    554,494,466,415,370,622,311,622,
    740,659,622,554,622,659,
    659,622,622,554,554,494,

};

int noteDurations[] = {

    2,2,2,2,2,2,1,
    2,2,2,2,2,2,1,
    2,2,2,2,2,2,1,
    2,2,2,2,2,2,1,
    2,2,2,2,2,2,1,
    2,2,2,2,2,2,1,
    1,1,2,2,2,2,2,1,2,
    2,1,2,2,2,2,2,1,
    1,1,2,2,2,2,2,1,2,
    2,2,2,1,1,1,1,2,
    2,2,1,2,2,2,2,1,2,
    2,2,2,2,1,1,1,2,
    2,1,2,2,2,2,2,1,2,
    2,2,2,1,1,2,1,2,
    2,2,1,1,2,1,2,
    2,2,1,1,2,2,
    1,1,1,1,1,1,
    2,1,2,1,2,1,2,
    2,2,1,1,2,1,2,2,
    2,2,1,1,1,1,2,1,
```

```
    1,1,1,1,2,2,
    1,1,2,1,2,1,

};
void setup()
{
    for (int thisNote = 0; thisNote <154; thisNote++)
    {
    int noteDuration = 1000/noteDurations[thisNote];// 计算每个节拍的时间，以一个节拍一秒为例，四分之一拍就是 1000/4 毫秒，八分之一拍就是 1000/8 毫秒
    tone(8, song[thisNote],noteDuration);
    int pauseBetweenNotes = noteDuration * 1.30; // 每个音符间的停顿间隔，以该音符的 130%为佳
    delay(pauseBetweenNotes);
    noTone(8);
    }
}
void loop()
{
    setup();//反复唱

}
```

视频地址：http://v.youku.com/v_show/id_XMjg1OTU5NTcy.html

作业三：水温提示杯

1、概述

这是我拿到 Arduino 起步套件后发的第二个作业，发的是一个水温提示杯，这个作业还是很简单的，如果我这个小菜鸟有什么考虑的不周到的地方，请各位大神指点一下我啊，毕竟学到东西最重要，奖金神马的都是浮云哟！

2、设计思路

区别于其它传感器，LM35 传感器的引脚图如下：

它的工作原理是：温度每升高 1℃，Vout 口输出的电压就增加 10mv。根据这一原理，实时读取模拟 5 口的电压值。如果电压值在 0.1 以下（21 为 AD 转换后的值），说明温度在 10℃以下，根据要求绿灯亮；如果电压值在 0.1v-0.4v 之间，说明温度在 10℃-40℃范围内，黄灯亮；如果电压值在 0.4v 以上（81 为 AD 转换后的值），说明温度在 40℃以上，红灯亮。由于模拟口读出的电压值使用 0-1023 表示，所以程序中用的数值都是经过计算的。

3、物品清单

面包板、扩展板和控制板的组合版，USB 接线，导线，温度传感器，LED

4、照片

电路连接图：

温度在 1℃左右的凉水：

温度在常温下的水：

温度在 90℃左右的开水：

5、源代码

```
#define LED_GREEN 10//定义与绿灯连接的引脚
#define LED_YELLOW 9//定义与黄灯连接的引脚
#define LED_RED 8//定义与红灯连接的引脚
void setup()
{
  unsigned char j;
  for(j=8;j<=10;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(5);//读取温度传感器电压值
      if(i<21)//温度在 10 度以下
      {
        digitalWrite(LED_GREEN,HIGH);//绿灯亮
        digitalWrite(LED_YELLOW,LOW);//黄灯灭
        digitalWrite(LED_RED,LOW);//红灯灭
      }
      else if(i>=21&&i<=81)//温度在 10~40 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 if(i>81)//温度在 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
      }
      else
      {
        digitalWrite(LED_RED,LOW);//红灯灭
        digitalWrite(LED_YELLOW,LOW);//黄灯灭
        digitalWrite(LED_GREEN,LOW);//绿灯灭
      }
    }
  }
```

作业四：莫尔斯码模拟器

方案设计：

1.通过蜂鸣器的脉宽调制产生不同的电压，实现 di、da 蜂鸣器发音。

2.串口通信，接收数据。

3.switch.....case 函数来识别，产生不同的声音。

作者

柳智

完成时间

2011 年 7 月 18 日

代码

```
char incomingByte = 0; // for incoming serial data
void setup() {
Serial.begin(9600); // opens serial port, sets data rate to 9600 bps
        pinMode(8,OUTPUT);
}
void myfun()
{
   switch(incomingByte)
   {
    case 'a':di();da();break;
    case 'b':da();di();di();break;
    case 'c':di();da();di();da();break;
    case 'd':da();di();di();break;
    case 'e':di();break;
    case 'f':di();di();da();di();break;
    case 'g':da();da();di();break;
    case 'h':di();di();di();di();break;
    case 'i':di();di();break;
    case 'j':di();da();da();da();break;
    case 'k':da();di();da();break;
    case 'l':di();da();di();di();break;
    case 'm':da();da();break;
    case 'n':da();di();break;
    case 'o':da();da();da();break;
    case 'p':di();da();da();di();break;
    case 'r':di();da();di();break;
    case 's':di();di();di();break;
    case 't':da();break;
    case 'u':di();di();da();break;
    case 'v':di();di();di();da();break;
    case 'w':di();da();da();break;
    case 'x':da();di();di();da();break;
```

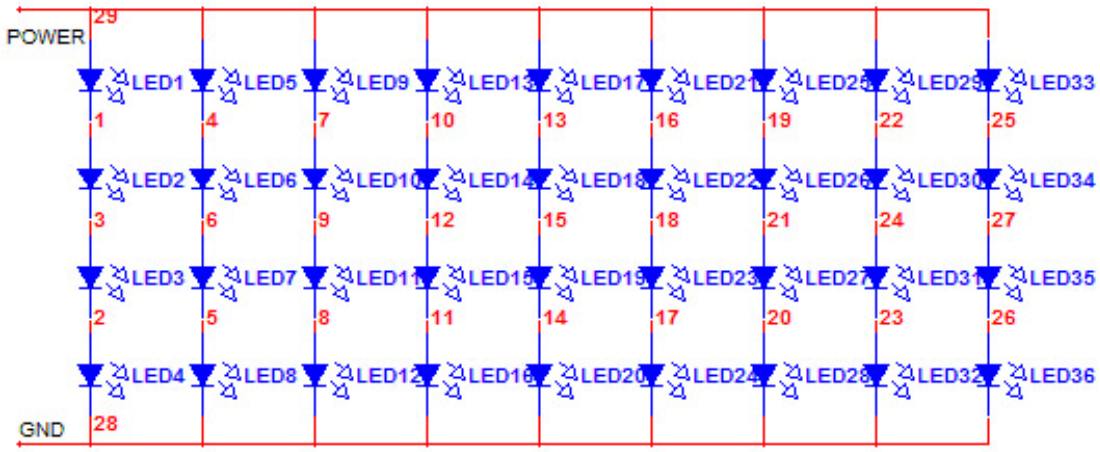
```
      case 'y':da();di();da();da();break;
      case 'z':da();da();di();di();break;
      case '1':di();da();da();da();da();break;
      case '2':di();di();da();da();da();break;
      case '3':di();di();di();da();da();break;
      case '4':di();di();di();di();da();break;
      case '5':di();di();di();di();di();break;
      case '6':da();di();di();di();di();break;
      case '7':da();di();di();di();break;
      case '8':da();da();da();di();di();break;
      case '9':da();da();da();da();di();break;
      case '0':da();da();da();da();da();break;
    }
    digitalWrite(8,HIGH);//key=1;
    delay(100);

}
void di()
{
digitalWrite(8,LOW);//key=0;
delay(100);
digitalWrite(8,HIGH);//key=1;
delay(100);
}
void da()
{ int i;
for(i=0;i<20;i++)
{
    digitalWrite(8,LOW);//key=0;
    delay(10);
    digitalWrite(8,HIGH);//key=1;
    delay(6);
}
digitalWrite(8,HIGH);//key=1;
delay(100);

}
void loop() {
            digitalWrite(8,HIGH);
if (Serial.available() > 0) {
    incomingByte = Serial.read();
                        myfun();
}
}
```

作业五：红外遥控 LED 台灯

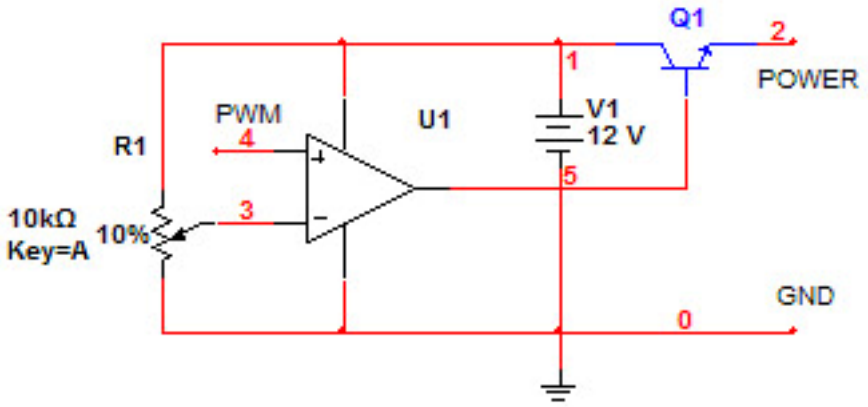
一、设计思路

总体思路：首先红外遥控需要一个遥控器与接收管，接受到的信号通过 I/0 口读，逻辑分析并控制 PWM 的占空比输出，然后经过驱动电路驱动 LED 显示。
红外遥控器发送格式为 9ms 低电平加 4.5ms 高电平表示起始，后面跟着 8 位系统编码，接着其反码，然后是 command 码与其反码，通过逻辑分析其值来进行控制。
接收管连接如下：

每次按下，接受并成功解码后，先将蜂鸣器发声 0.1s，然后输出改变后的 PWM 波形。
LED 灯结构如下：



每个 LED 工作电压 3.2~3.4V，工作电流 20mA 左右，所以总驱动电压 12~13V 就差不多了。
驱动管选用的 13003 效果比较好，电压转换用的运放做的电压比较器。电路如下：



二、物料清单

Arduino 开发板、跳线若干、蜂鸣器一只、红外接收管一只、13003 一只、LM358P 一片、10K 滑动变阻器一个、面包版一块、LED 灯阵、改造灯座。

三、源代码

```
#define BUZZER 10//蜂鸣器
#define IR_IN    8    //红外接收
int Pulse_Width=0;//存储脉宽
int    ir_code=0x00;//命令值
int pm=0;
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 timer2_init(int pm)
{
   switch (pm)
      {
            case 0:
            OCR2A =0X00;
            TCNT2 =0X00;
            TCCR2A =0X00;
            TCCR2B =0X00;
            TIMSK2=0X00;
            break;
            case 1:
            OCR2A =0X3F;
            TCNT2 =0X00;
            TCCR2A =0X83;
            TCCR2B =0X04;
            TIMSK2=0X00;
            break;
            case 2:
            OCR2A =0X7F;
            TCNT2 =0X00;
            TCCR2A =0X83;
            TCCR2B =0X04;
            TIMSK2=0X00;
            break;
            case 3:
            OCR2A =0XBF;
```

```
                TCNT2 =0X00;
                TCCR2A =0X83;
                TCCR2B =0X04;
                TIMSK2=0X00;
                break;
                case 4:
                OCR2A =0XFF;
                TCNT2 =0X00;
                TCCR2A =0X83;
                TCCR2B =0X04;
                TIMSK2=0X00;
                break;
        }
}
void remote_deal(void)//执行译码结果函数
{
   switch(ir_code)
   {
   case 0xff00://停止
      break;
   case 0xea15://VOL+
      digitalWrite(10,HIGH);
      delay(100);
      digitalWrite(10,LOW);
      ir_code=0xff00;
      pwm_1(1);
      timer2_init(pm);
      break;
   case 0xf807://VOL-
      digitalWrite(10,HIGH);
      delay(100);
      digitalWrite(10,LOW);
      ir_code=0xff00;
      pwm_1(0);
      timer2_init (pm);
      break;
   }
}
void pwm_1(int x)
{
   if(x==1)
        {
            if(pm!=4) pm=pm+1;
            else if(pm==4) pm=0;
```

```
        }
    else if(x==0)
        {
          if (pm!=0) pm=pm-1;
          else if (pm==0) pm=4;
        }
}
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&&ulse_Width<=10)//低电平 560us
   {
      while(digitalRead(8));//是高就等待
      Pulse_Width=TCNT1;
      TCNT1=0;
      if(Pulse_Width>=7&&ulse_Width<=10)//接着高电平 560us
         return 0;
      else if(Pulse_Width>=25&&ulse_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 8 个 1
   for(i=0; i<8; i++)
   {
      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&&ulse_Width<=141)//9ms
            {
              while(digitalRead(8));//是高就等待
              Pulse_Width=TCNT1;
              TCNT1=0;
              if(Pulse_Width>=68&&ulse_Width<=72)//4.5ms
              {
                pulse_deal();
                return;
              }
              else if(Pulse_Width>=34&&ulse_Width<=36)//2.25ms
              {
                while(!(digitalRead(8)));//低等待
                Pulse_Width=TCNT1;
                TCNT1=0;
                if(Pulse_Width>=7&&ulse_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
    timer2_init(0);
    while(1)
    {
        remote_decode();    //译码
        remote_deal();      //执行译码结果
//      delay(10);
    }
}
```

作业六：禁烟器

1、概述

拿到 Anduino 起步套装已经快 10 天了，这段时间都是在研究基本的一些东西，比如：把附赠光盘里面的东西都学习了一遍，把里面的实例都按照要求实现了一遍，觉得里面的东西都是一些很基础的东西，上手也挺快的，本来这个作业还是比较简单的，但是由于作业不是按照难度排列的，加上前面的有些作业的程序编写起来还是有点棘手，所以这个简单的作业到现在才交。现在把自己的写的一个很简单的小程序发上来，希望各位专家和高手给我指出可以改进的地方，督促我这个小菜鸟更进一步。

2、设计思路

我们从教程中知道，在火焰传感器的接线电路接好后，有火焰靠近和没有火焰靠近两种情冴下，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，我先存储一个没有火焰时模拟口的电压值 i。接着不断得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 0.3v 做比较（取 0.3 是实验得出的，因为作业要求是要在 1m 范围内检测打火机的火焰，为了使得程序比较准确，所以取得阈值为 0.3，而不是想教程里面取得 0.6）。差值 k 如果大于 0.3V（数字二迮制值为 62），则判断有火焰靠近，让蜂鸣器发出 2HZ 频率的声音以作报警，同时 LED 灯亮示警；如果差值小于 0.3v 则蜂鸣器不响，LED 灯熄灭。

3、物品清单

面包板、扩展板和控制板的组合版，USB 接线，导线，火焰传感器，蜂鸣器，LED

4、源代码

```
int g;//定义变量 g
void buzzer()//蜂鸣器鸣叫 5 声，频率 2Hz
{
        for(g=0;g<5;g++) //蜂鸣器鸣叫 5 声
        {
          digitalWrite(8,HIGH);//发声音
          delay(250);//延时 250ms
          digitalWrite(8,LOW);//不发声音
          delay(250);//延时 250ms
        }
}
void setup()
{
    pinMode(8,OUTPUT);//设置数字 8 引脚为输出方式
    pinMode(10,OUTPUT);//设置数字 10 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>62)//如果差值大于 0.3（0.3 为模拟值，62 为对应的数字二进制值）
说明有火焰
      {
        digitalWrite(10,HIGH);
        buzzer();//蜂鸣器发出声音
      }
      else
      {
        digitalWrite(10,LOW);
        digitalWrite(8,LOW);//设置数字 8 口为低电平，蜂鸣器不响
      }
    }
}
```

作业七：自动控制廊灯

1、设计思路

通过教程以及相关资料得知：光敏电阻具有关照越强电阻越小的特性。根据这一特性，以及题目的要求（功过光敏期间，感知环境亮度，从而控制 LED 灯的亮灭，环境越亮 LED 越暗，反之越亮），可以设计出如下所示的外围电路（其中 5 为模拟 IO 口 5，6 为数字 IO 口 6）。

通过万用表对二极管的测量得知：当二极管两端电压为 0.19V 时，二极管几乎不发光。在二极管电压允许范围内（0~3V），电压越大，发光二极管发出的光越强；光敏电阻的变化范围是 0.3 千欧~~200 千欧。根据实际情况，当关照十分强的情况下可以使灯完全熄灭。所以在程序设计时，输入的电压为小于 0.39V 时控制板输出低电平，不点亮发光二极管（digitalWrite(6,LOW)），其他情况下按输出按题目的要求处理，即环境越亮 LED 越暗，反之越亮。

2、物品清单

面包板、扩展板和控制板的组合版，USB 接线，导线，光敏电阻，10k 电阻。

3、源代码

```
        void setup()
{

pinMode(6,OUTPUT);
//设置数字 6 引脚为输出方式
}
void loop()
{

int i;
//定义变量

while(1)

{

i=analogRead(5);
//不断的读取模拟口的电压值，时时监测

if(i<80)
digitalWrite(6,LOW);
//如果差值小于 0.39V（0.39V 为模拟值，80 为对应的数字二进制值）说明光照非常强，灯熄灭。
```

```
else
analogWrite(6,i/4);
//否则十位和八位进行转换，并输出模拟量。

}
}
```

作业八：打地鼠游戏机

一、项目要求：

打地鼠游戏机（LED，按键，蜂鸣器）

现象描述：作品中 9 个 LED 对应 9 个按键，刚开始 0.5 秒生成一个 1 至 9 的随机数，对应点亮相应的 LED，在下次随机数生成之前判断相应的按键是否按下，若按键错误，计错误一次，蜂鸣器鸣叫 0.2 秒，重新生成随机数。完成 10 次操作后生成随机数频率变为 0.4 秒一次，又完成 10 次操作后频率变为 0.3 秒一次，依次类推，0.1 秒的情况下完成 10 次操作游戏结束。蜂鸣器鸣叫 5 声，频率 5Hz。在游戏过程中，错误 3 次游戏也结束，蜂鸣器鸣叫 3 声，频率 5Hz。

二、所用器材：

- Arduino 328 控制板一个
- 面包板 二个
- LED 9 个
- 按键 9 个
- 蜂鸣器 1 个
- 接线若干

源代码

```
int led1=1;
int led2=2;
int led3=3;
int led4=4;
int led5=5;
int led6=6;
int led7=7;
int led8=8;
int led9=9;
int buzzer=10;
int p1=11;
int p2=12;
int p3=13;

void led(void)
{   int num;
   num=rand()%9+1;
if(num==1)        digitalWrite(led1,HIGH);
if(num==2)        digitalWrite(led2,HIGH);
if(num==3)        digitalWrite(led3,HIGH);
if(num==4)        digitalWrite(led4,HIGH);
if(num==5)        digitalWrite(led5,HIGH);
if(num==6)        digitalWrite(led6,HIGH);
if(num==7)        digitalWrite(led7,HIGH);
```

```
if(num==8)        digitalWrite(led8,HIGH);
if(num==9)        digitalWrite(led9,HIGH);

   }

int key_scan()
{   int i;
     int j=rand()%9+1;
     digitalWrite(p1,LOW);
     digitalWrite(p2,HIGH);
     digitalWrite(p3,HIGH);
    if(analogRead(0)>512)
     { if((rand()%9+1)==1) ;
          else
          {i++;
          digitalWrite(buzzer,HIGH);
          delay(200);
          digitalWrite(buzzer,LOW);}
     }

      if(analogRead(1)>512)
     { if((rand()%9+1)==4) ;
        else i++;
          digitalWrite(buzzer,HIGH);
          delay(200);
          digitalWrite(buzzer,LOW);
     }

     if(analogRead(2)>512)
      { if((rand()%9+1)==7) ;
      else{ i++;
      digitalWrite(buzzer,HIGH);
      delay(200);
      digitalWrite(buzzer,LOW);}
      }


     digitalWrite(p1,HIGH);
     digitalWrite(p2,LOW);
     digitalWrite(p3,HIGH);
     if(analogRead(0)>512)
        {if((rand()%9+1)==2) ;
         else
        { i++;
```

```
   digitalWrite(buzzer,HIGH);
    delay(200);
    digitalWrite(buzzer,LOW);
  }}

if(analogRead(1)>512)
  { if((rand()%9+1)==5) ;
    else
    {i++;
    digitalWrite(buzzer,HIGH);
    delay(200);
    digitalWrite(buzzer,LOW);}
  }

if(analogRead(2)>512)
  { if((rand()%9+1)==8) ;
    else
   { i++;
    digitalWrite(buzzer,HIGH);
    delay(200);
    digitalWrite(buzzer,LOW);}
  }


digitalWrite(p1,HIGH);
digitalWrite(p2,HIGH);
digitalWrite(p3,LOW);
if(analogRead(0)>512)
{ if((rand()%9+1)==3) ;
else { i++;
    digitalWrite(buzzer,HIGH);
    delay(200);
    digitalWrite(buzzer,LOW);}
  }

  if(analogRead(1)>512)
{ if((rand()%9+1)==6) ;
  else
   { i++;
     digitalWrite(buzzer,HIGH);
     delay(200);
     digitalWrite(buzzer,LOW);}
  }
```

```
        if(analogRead(2)>512)
         { if((rand()%9+1)==9) ;
            else
                {   i++;
                  digitalWrite(buzzer,HIGH);
                  delay(200);
                  digitalWrite(buzzer,LOW);
    }
         }
         digitalWrite(led1,LOW);
         digitalWrite(led2,LOW);
         digitalWrite(led3,LOW);
         digitalWrite(led4,LOW);
         digitalWrite(led5,LOW);
         digitalWrite(led6,LOW);
         digitalWrite(led7,LOW);
         digitalWrite(led8,LOW);
         digitalWrite(led9,LOW);
         return i;
}

int judge()
{ int j;
       j++;
   if(j<=10)
   delay(500);
   if(j<=20&j>10)
   delay(400);
   if(j<=30&j>20)
   delay(300);
   if(j<=40&j>30)
   delay(200);
   if(j<50&j>40)
   delay(100);
return j;
}

void setup()
{
   int i;//定义变量
   for(i=1;i<=13;i++)
     pinMode(i,OUTPUT);//设置 1～13 引脚为输出模式
}
```

```
void loop()
{   int i,j;
   while(1)
   {   digitalWrite( buzzer,LOW);
       led();
    j=judge();
    i=key_scan();
    if(i=3)
    { if(i--)     {
                  digitalWrite(buzzer,HIGH);
                  delay(100);
                  digitalWrite(buzzer,LOW);
                  delay(100);}
                  break;}
    if(j=50)
          { for(j=5;j>0;j--){
                  digitalWrite(buzzer,HIGH);
                  delay(100);
                  digitalWrite(buzzer,LOW);
                  delay(100);}
                  break;}
   }
}
```

创意方案：基于 Arduino 的音乐频谱显示器

【方案概述】

在家庭影院、卡拉 OK 等音响系统中，实时显示音乐信号的频谱将为音乐系统增添不少色彩。这里采用 Arduino 来实现这一效果，具体系统设计原理图如图 1：

整个系统分前端放大、ADC 采集转换、FFT 频谱计算、LED 点阵驱动四部分组成。下面分不分对各个不分进行说明：

1.前端放大电路

直接从电脑音频孔取出的音频信号是个交流信号，电压幅值约 100mv~500mv 左右，而 ADC 只能采样正电压，为了充分利用 ADC 的采样精度，需要将音频信号放大并叠加一个直流分量（有点类似数字示波器的前端电路），在设计中我直接用了一个音频集成功放 LM386，原理图如下（参数以网上实际参数为准）：

测试从 386 的 5 脚取出信号，刚好有 2.5V 的直流电压，另外放大倍数约 20 倍。

2.ADC 采样

直接接入 atmega8 的模拟输入端即可。这里要注意采样频率，按照采样定理，采样频率要大于信号频率的两倍，这些上课时大家都学过，就不展开了。而我们音频信号频率范围在 20Hz~20kHz 之间，按照最大的频率来算，要求 ADC 采样频率不得小于 40kHz。

在 Arduino 的教程中介绍其 ADC 完成一次采样转换的时间要 100μ 秒，约 10K 的采样率，不符合上述对音频信号采样最低频率的要求，在网上找到解决办法如下：

3、源代码

```
#define FASTADC 1
        // defines for setting and clearing register bits
        #ifndef cbi
        #define cbi(sfr, bit) (_SFR_BYTE(sfr) &= ~_BV(bit))
        #endif
        #ifndef sbi
        #define sbi(sfr, bit) (_SFR_BYTE(sfr) |= _BV(bit))
        #endif
        //。。。。。。。。
        //在 setup()函数中添加
          #if FASTADC
            //set prescale to 16
          sbi(ADCSRA,ADPS2) ;
          cbi(ADCSRA,ADPS1) ;
          cbi(ADCSRA,ADPS0) ;
```

```
#endif
```

如此转换频率理论达到 60K 左右，添加一些其他处理代码后测试得到的一次采样率约为 48K，符合要求。

3.LED 点阵屏驱动

为了美观，采用方形发光二极管（蓝色）来焊接的一个 8*13 的 LED 点阵屏，原理图跟常见的 8*8 的一样。由于开发板 IO 口的限制，只有 14 个数字 IO 口，所以采用 2 块 74HC595 串转并，用来扫描 13 个公共端，数据段直接由数字 IO 驱动，这里要注意限流：一是 LED 的最大电流，二是 595 最大灌电流，三是 atmega8 的最大输出电流。

```
  #define FASTADC 1
// defines for setting and clearing register bits
#ifndef cbi
#define cbi(sfr, bit) (_SFR_BYTE(sfr) &= ~_BV(bit))
#endif
#ifndef sbi
#define sbi(sfr, bit) (_SFR_BYTE(sfr) |= _BV(bit))
#endif

//设置 595 的控制 IO 脚
int SCLK = 8;
int LCLK = 9;
int SDI = 10;

int SN = 64;
int analogPin=0;
int offset=500;
struct compx                                    //定义复数结构体
{
        float re;
        float im;
};
struct compx s[ 64 ];
unsigned char tab[9]={0x00,0x01,0x03,0x07,0x0f,0x1f,0x3f,0x7f,0xff};
unsigned int tab2[13]={0xfffe,0xfffd,0xfffb,0xfff7,
                       0xffef,0xffdf,0xffbf,0xff7f,
                       0xfeff,0xfdff,0xfbff,0xf7ff,0xefff};
struct compx cpx_multiplication(struct compx,struct compx);          //复数乘
法函数的声明
```

```
        void FFT(struct compx xin[],int N);
//FFT 函数的声明
        void ADC_SAMPLE();

        void init595()
        {
            digitalWrite(SCLK,LOW);
            digitalWrite(LCLK,LOW);
            digitalWrite(SDI,LOW);
        }

        void send_led(unsigned char dat)
        {
          for(int i=0;i<8;i++)
          {
            digitalWrite(i,dat&0x01);
            dat = dat>>1;
          }
        }

        void send_595(unsigned int dat)
        {
          unsigned char i;
          for(i=0;i<13;i++)
          {
            //SDI = dat&0x01;
            digitalWrite(SDI,dat&0x01);
            digitalWrite(SCLK,HIGH);
            digitalWrite(SCLK,LOW);
            dat = dat>>1;
          }
        }

        void setup(){
          unsigned char i;
          #if FASTADC
            //set prescale to 16
          sbi(ADCSRA,ADPS2) ;
          cbi(ADCSRA,ADPS1) ;
          cbi(ADCSRA,ADPS0) ;
        #endif

          for(i=0;i<=10;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
```

```
        digitalWrite(i,LOW);
    }
    init595();
}

void loop(){
    int val[64];
    unsigned char i,j;
    ADC_SAMPLE();

    FFT(s,SN);
    for(i=0,j=0,m=0;i<SN/2;i+=2){
        val[j]=int(sqrt(pow(s[i].re,2)+pow(s[i].im,2))/40);
    if(j<13){
        if(val[j]>8)
                val[j] = 8;
          send_led(tab[val[j]]);
          send_595(tab2[j]);
          digitalWrite(LCLK,HIGH);
          digitalWrite(LCLK,LOW);
    }
    j+=1;
    }
}

/*复数乘法*/

struct compx cpx_multiplication (struct compx a1,struct compx b2)
{
            struct compx b3;
            b3.re=a1.re*b2.re-a1.im*b2.im;
            b3.im=a1.re*b2.im+a1.im*b2.re;
            return(b3);
}

/*FFT 函数*/
void FFT(struct compx xin[],int N)
{
            int f,m,nv2,nm1,i,k,j=1,l;
            struct compx v,w,t;
            int le,lei,ip;
               float pi=3.14159265;
            nv2=N/2;
            f=N;
```

```
                for(m=1;(f=f/2)!=1;m++){;}

                nm1=N-1;
                for(i=0;i<nm1;i++)
//倒序操作
                {
                        if(i<j)
                        {
                                t=xin[j];
                                xin[j]=xin[i];
                                xin[i]=t;
                        }
                        k=nv2;
//k 为倒序中相应位置的权值
                        while(k<j)
                        {
                                j=j-k;
                                k=k/2;
                        }
                        j=j+k;
                }
        /************************************************/
                           for(l=1;l<=m;l++)
                             {
                        le=pow(2,l);
//乘方
                              lei=le/2;

                              v.re=1.0;
                        v.im=0.0;
                             w.re=cos(pi/lei);
//旋转因子
                              w.im=(-1)*sin(pi/lei);

                        for(j=1;j<=lei;j++)
//控制蝶形运算的级数
                              {
                                for(i=j-1;i<N;i=i+le)
//控制每级蝶形运算的次数
                                        {
                                           ip=i+lei;
                                                t=cpx_multiplication(xin[ ip ],v);
```

```
                                                    xin[ ip ].re=xin[ i ].re-t.re;      //蝶
形计算
                                                    xin[ ip ].im=xin[ i ].im-t.im;
                                                    xin[ i ].re=xin[ i ].re+t.re;
                                                    xin[ i ].im=xin[ i ].im+t.im;
                                                }
                                                v=cpx_multiplication(v,w);
                                            }
                                        }

        }

        /*ADC 采样*/
        void ADC_SAMPLE()
        {
            int i;
            for(i=0;i<SN;i++)
           {
              s[i].re=float((analogRead(analogPin)-offset))/6;        // read the input pin
              s[i].im=0;
           }
        }
```

创意方案：基于 Arduino 的 LED 广告屏

【方案概述】

LED 显示屏采用了低电压扫描驱动，具有耗电省、使用寿命长、成本低、亮度高、视角大、可视距离远、防水、规格品种多等优点，可以满足各种不同应用场景的需求，发展前景非常广阔，被公认为最具增长潜力也是发展最快的的 LED 应用市场。2008 年 LED 显示屏市场规模约 100 亿元。随着北京奥运会、上海世博会、广州亚运会等重大赛会的举办和筹备，体育场馆、机场、车站、银行、医院、公共广场、商业场所、居民社区的大面积应用，LED 显示屏的市场应用空间不断扩大。此外，已架设的大型 LED 显示屏幕每 10 年将历经一次换机潮，随着人们生活水平的提高，户外 led 显示屏将逐渐应用于各个行业。

说明：1、LED 模块为大小为 16*64LED 点阵显示屏，采用 74HC154 和 74HC595 级联驱动方式。

2、主控制器采用 Arduino 328 主板

【程序代码 1（静态显示）】

```
    int DS=1;
int ST_CP=2;
int SH_CP=3;
int A=4;
int B=5;
int C=6;
int D=7;

//unsigned char display_1[]={
//0x10,0x04,0x60,0x04,0x02,0x7E,0x8C,0x01,0x00,0x00,0x88,0x1F,0x88,0x08,0
xFF,0x08,
//0x88,0x08,0x88,0x9F,0x00,0x60,0xFE,0x1F,0x22,0x42,0x22,0x82,0xFE,0x7F,0
x00,0x00,/*"湖",0*/
//
//0x00,0x20,0x20,0x60,0x20,0x20,0x20,0x10,0x20,0x10,0xFF,0xFF,0x00,0x00,0
x00,0x00,
//0x00,0x00,0xFF,0x3F,0x40,0x40,0x20,0x40,0x10,0x40,0x08,0x40,0x00,0x78,0
x00,0x00,/*"北",1*/
//
//0x00,0x00,0xFC,0x87,0x00,0x40,0x00,0x30,0xFF,0x0F,0x00,0x00,0x02,0x00,0
xE2,0x1F,
//0x22,0x00,0x22,0x00,0xFE,0xFF,0x22,0x08,0x22,0x10,0xE2,0x0F,0x02,0x00,0
x00,0x00,/*"师",2*/
//
```

```
        //0x00,0x00,0xFE,0xFF,0x22,0x04,0x5A,0x08,0x86,0x07,0x10,0x80,0x0C,0x41,
0x24,0x31,
        //0x24,0x0F,0x25,0x01,0x26,0x01,0x24,0x3F,0x24,0x41,0x14,0x41,0x0C,0x71,0
x00,0x00,/*"院",3*/
        //};
        unsigned char display_1[]={
        0x40,0x80,0x48,0x60,0x48,0x1F,0x48,0x20,0xFF,0x7F,0x48,0x44,0x48,0x44,0x
00,0x40,
        0xC4,0x4F,0x44,0x50,0x44,0x50,0x44,0x50,0xFC,0x50,0x00,0x5C,0x00,0x40,0
x00,0x00,/*"起",0*/

        0x80,0x80,0xFC,0x7F,0x96,0x02,0xE5,0x4C,0x84,0x80,0xFC,0x7F,0x00,0x80,0
x08,0x60,
        0xC8,0x1F,0x49,0x00,0x4A,0x00,0xC8,0x3F,0x08,0x40,0x08,0x40,0x00,0x78,0
x00,0x00,/*"航",1*/

        0x00,0x00,0x00,0x00,0xF8,0x1F,0x88,0x08,0x88,0x08,0x88,0x08,0x88,0x08,0x
FF,0x7F,
        0x88,0x88,0x88,0x88,0x88,0x88,0x88,0x88,0xF8,0x9F,0x00,0x80,0x00,0xF0,0x
00,0x00,/*"电",2*/

        0x80,0x00,0x82,0x00,0x82,0x00,0x82,0x00,0x82,0x00,0x82,0x40,0x82,0x80,0x
E2,0x7F,
        0xA2,0x00,0x92,0x00,0x8A,0x00,0x86,0x00,0x82,0x00,0x80,0x00,0x80,0x00,0x
00,0x00,/*"子",3*/
        };
        void hc595_senddat(unsigned char dat)
        {
        unsigned char i;
        for(i=0;i<8;i++)
        {
        digitalWrite(DS,dat&0x80);// DS=dat&0x80;
        digitalWrite(SH_CP,HIGH);// SH_CP=1;
        digitalWrite(SH_CP,LOW);// SH_CP=0;
        dat<<=1;
        }
        }

        void setup()
        {
        unsigned char i;
        for(i=1;i<=7;i++)//依次设置 1~7 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()
        {
        unsigned char i;
        char j;
        digitalWrite(SH_CP,LOW); //SH_CP=0;
        digitalWrite(ST_CP,LOW); //ST_CP=0;
        while(1)
        {
        for(i=0;i<16;i++)
        {
        for(j=3;j>=0;j--)
        {
        hc595_senddat(~display_1[32*j+2*i+1]);
        hc595_senddat(~display_1[32*j+2*i]);
        }
        switch(i)
        {
        case
0:digitalWrite(A,LOW);digitalWrite(B,LOW);digitalWrite(C,LOW);digitalWrite(D,LOW);
break; //译码器 ABCD 地址端为 0000
        case
1:digitalWrite(A,HIGH);digitalWrite(B,LOW);digitalWrite(C,LOW);digitalWrite(D,LOW)
; break; //1000
        case
2:digitalWrite(A,LOW);digitalWrite(B,HIGH);digitalWrite(C,LOW);digitalWrite(D,LOW)
; break; //0100
        case
3:digitalWrite(A,HIGH);digitalWrite(B,HIGH);digitalWrite(C,LOW);digitalWrite(D,LOW
); break; //1100
        case
4:digitalWrite(A,LOW);digitalWrite(B,LOW);digitalWrite(C,HIGH);digitalWrite(D,LOW)
; break; //0010
        case
5:digitalWrite(A,HIGH);digitalWrite(B,LOW);digitalWrite(C,HIGH);digitalWrite(D,LOW
); break; //1010
        case
6:digitalWrite(A,LOW);digitalWrite(B,HIGH);digitalWrite(C,HIGH);digitalWrite(D,LOW
); break; //0110
        case
7:digitalWrite(A,HIGH);digitalWrite(B,HIGH);digitalWrite(C,HIGH);digitalWrite(D,LOW
); break; //1110
        case
8:digitalWrite(A,LOW);digitalWrite(B,LOW);digitalWrite(C,LOW);digitalWrite(D,HIGH)
; break; //0001
```

```
        case
9:digitalWrite(A,HIGH);digitalWrite(B,LOW);digitalWrite(C,LOW);digitalWrite(D,HIGH
); break; //1001
        case
10:digitalWrite(A,LOW);digitalWrite(B,HIGH);digitalWrite(C,LOW);digitalWrite(D,HIG
H); break; //0101
        case
11:digitalWrite(A,HIGH);digitalWrite(B,HIGH);digitalWrite(C,LOW);digitalWrite(D,HIG
H); break; //1101
        case
12:digitalWrite(A,LOW);digitalWrite(B,LOW);digitalWrite(C,HIGH);digitalWrite(D,HIG
H); break; //0011
        case
13:digitalWrite(A,HIGH);digitalWrite(B,LOW);digitalWrite(C,HIGH);digitalWrite(D,HIG
H); break; //1011
        case
14:digitalWrite(A,LOW);digitalWrite(B,HIGH);digitalWrite(C,HIGH);digitalWrite(D,HIG
H); break; //0111
        case
15:digitalWrite(A,HIGH);digitalWrite(B,HIGH);digitalWrite(C,HIGH);digitalWrite(D,HIG
H); break; //1111
        }
        digitalWrite(ST_CP,HIGH);//ST_CP=1;
        digitalWrite(ST_CP,LOW);//ST_CP=0;
        }
        }
        }
```

【程序代码 2（静态显示）】

```
          int DS=1;
        int ST_CP=2;
        int SH_CP=3;
        int A=4;
        int B=5;
        int C=6;
        int D=7;
        unsigned char display_1[]={
        0x20,0x80,0x20,0x80,0x20,0x40,0x20,0x20,0x20,0x10,0x20,0x0C,0x20,0x03,0x
FF,0x00,
        0x20,0x03,0x20,0x0C,0x20,0x10,0x20,0x20,0x20,0x40,0x20,0x80,0x20,0x80,0x
00,0x00,/*"大",0*/
```

```
0x40,0x04,0x30,0x04,0x11,0x04,0x96,0x04,0x90,0x04,0x90,0x44,0x91,0x84,0x96,0x7E,
0x90,0x06,0x90,0x05,0x98,0x04,0x14,0x04,0x13,0x04,0x50,0x04,0x30,0x04,0x00,0x00,/*"学",1*/
0x80,0x40,0x40,0x40,0x30,0x42,0x1E,0x42,0x10,0x42,0x10,0x42,0x10,0x42,0xFF,0x7F,
0x10,0x42,0x10,0x42,0x10,0x42,0x10,0x42,0x10,0x42,0x10,0x40,0x00,0x40,0x00,0x00,/*"生",2*/
0x00,0x00,0x00,0x00,0xF8,0x1F,0x88,0x08,0x88,0x08,0x88,0x08,0x88,0x08,0xFF,0x7F,
0x88,0x88,0x88,0x88,0x88,0x88,0x88,0x88,0xF8,0x9F,0x00,0x80,0x00,0xF0,0x00,0x00,/*"电",3*/
0x80,0x00,0x82,0x00,0x82,0x00,0x82,0x00,0x82,0x00,0x82,0x40,0x82,0x80,0xE2,0x7F,
0xA2,0x00,0x92,0x00,0x8A,0x00,0x86,0x00,0x82,0x00,0x80,0x00,0x80,0x00,0x00,0x00,/*"子",4*/
0x10,0x04,0x0C,0x84,0x04,0x84,0x84,0x44,0x14,0x47,0x64,0x24,0x05,0x14,0x06,0x0C,
0xF4,0x07,0x04,0x0C,0x04,0x14,0x04,0x24,0x04,0x44,0x14,0x84,0x0C,0x04,0x00,0x00,/*"实",5*/
0x02,0x08,0xFA,0x18,0x82,0x48,0x82,0x84,0xFE,0x44,0x80,0x3F,0x40,0x40,0x20,0x44,
0x50,0x58,0x4C,0x41,0x43,0x4E,0x4C,0x60,0x50,0x58,0x20,0x47,0x40,0x40,0x00,0x00,/*"验",6*/
0x10,0x40,0x0C,0x40,0x24,0x48,0x24,0x49,0xA4,0x49,0x64,0x49,0x25,0x49,0x26,0x7F,
0x24,0x49,0x24,0x49,0xA4,0x49,0x24,0x4B,0x24,0x48,0x14,0x40,0x0C,0x40,0x00,0x00,/*"室",7*/
0x40,0x00,0x50,0x00,0x4E,0x3E,0x48,0x02,0x48,0x02,0xFF,0xFF,0x48,0x12,0x48,0x22,
0x48,0x1E,0x40,0x00,0xF8,0x0F,0x00,0x40,0x00,0x80,0xFF,0x7F,0x00,0x00,0x00,0x00,/*"制",8*/
0x00,0x01,0x80,0x00,0x60,0x00,0xF8,0xFF,0x07,0x00,0x40,0x00,0x30,0x00,0x0F,0x00,
0xF8,0xFF,0x88,0x08,0x88,0x08,0x88,0x08,0x88,0x08,0x08,0x08,0x08,0x00,0x00,0x00,/*"作",9*/
0x08,0x20,0xF8,0x3F,0x00,0x21,0x80,0x00,0x80,0x00,0x80,0x20,0x00,0x3F,0x00,0x20,/*"h",10*/
0x00,0x00,0x80,0x00,0x80,0x00,0xE0,0x1F,0x80,0x20,0x80,0x20,0x00,0x00,0x00,0x00,/*"t",11*/
0x00,0x00,0x80,0x00,0x80,0x00,0xE0,0x1F,0x80,0x20,0x80,0x20,0x00,0x00,0x00,0x00,/*"t",12*/
0x80,0x80,0x80,0xFF,0x00,0xA1,0x80,0x20,0x80,0x20,0x00,0x11,0x00,0x0E,0x00,0x00,/*"p",13*/
```

```
	0x00,0x00,0x00,0x00,0x00,0x00,0xC0,0x30,0xC0,0x30,0x00,0x00,0x00,0x00,0x00,0x00,/*":",14*/
	0x00,0x00,0x00,0x60,0x00,0x18,0x00,0x06,0x80,0x01,0x60,0x00,0x18,0x00,0x04,0x00,/*"/",15*/
	0x00,0x00,0x00,0x60,0x00,0x18,0x00,0x06,0x80,0x01,0x60,0x00,0x18,0x00,0x04,0x00,/*"/",16*/
	0x80,0x0F,0x80,0x30,0x00,0x0C,0x80,0x03,0x00,0x0C,0x80,0x30,0x80,0x0F,0x80,0x00,/*"w",17*/
	0x80,0x0F,0x80,0x30,0x00,0x0C,0x80,0x03,0x00,0x0C,0x80,0x30,0x80,0x0F,0x80,0x00,/*"w",18*/
	0x80,0x0F,0x80,0x30,0x00,0x0C,0x80,0x03,0x00,0x0C,0x80,0x30,0x80,0x0F,0x80,0x00,/*"w",19*/
	0x00,0x00,0x00,0x30,0x00,0x30,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,/*".",20*/
	0x00,0x00,0x00,0x0E,0x00,0x11,0x80,0x20,0x80,0x20,0x80,0x20,0x00,0x11,0x00,0x00,/*"c",21*/
	0x00,0x00,0x00,0x1F,0x80,0x22,0x80,0x22,0x80,0x22,0x80,0x22,0x00,0x13,0x00,0x00,/*"e",22*/
	0x00,0x00,0x00,0x1F,0x80,0x22,0x80,0x22,0x80,0x22,0x80,0x22,0x00,0x13,0x00,0x00,/*"e",23*/
	0x00,0x00,0x80,0x00,0x80,0x00,0xE0,0x1F,0x80,0x20,0x80,0x20,0x00,0x00,0x00,0x00,/*"t",24*/
	0x00,0x00,0x00,0x30,0x00,0x30,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,/*".",25*/
	0x08,0x20,0xF8,0x3F,0x00,0x21,0x80,0x00,0x80,0x00,0x80,0x20,0x00,0x3F,0x00,0x20,/*"h",26*/
	0x08,0x00,0xF8,0x3F,0x00,0x11,0x80,0x20,0x80,0x20,0x00,0x11,0x00,0x0E,0x00,0x00,/*"b",27*/
	0x80,0x20,0x80,0x3F,0x00,0x21,0x80,0x00,0x80,0x00,0x80,0x20,0x00,0x3F,0x00,0x20,/*"n",28*/
	0x80,0x00,0x80,0x1F,0x00,0x20,0x00,0x20,0x00,0x20,0x80,0x10,0x80,0x3F,0x00,0x20,/*"u",29*/
	0x00,0x00,0x00,0x30,0x00,0x30,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,/*".",30*/
	0x00,0x00,0x00,0x1F,0x80,0x22,0x80,0x22,0x80,0x22,0x80,0x22,0x00,0x13,0x00,0x00,/*"e",31*/
	0x00,0x00,0x00,0x0E,0x00,0x11,0x80,0x20,0x80,0x20,0x88,0x10,0xF8,0x3F,0x00,0x20,/*"d",32*/
	0x80,0x00,0x80,0x1F,0x00,0x20,0x00,0x20,0x00,0x20,0x80,0x10,0x80,0x3F,0x00,0x20,/*"u",33*/
	0x00,0x00,0x00,0x30,0x00,0x30,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,/*".",34*/
	0x00,0x00,0x00,0x0E,0x00,0x11,0x80,0x20,0x80,0x20,0x80,0x20,0x00,0x11,0x00,0x00,/*"c",35*/
```

```
	0x80,0x20,0x80,0x3F,0x00,0x21,0x80,0x00,0x80,0x00,0x80,0x20,0x00,0x3F,0x00,0x20,/*"n",36*/
	0x00,0x00,0x00,0x60,0x00,0x18,0x00,0x06,0x80,0x01,0x60,0x00,0x18,0x00,0x04,0x00,/*"/",37*/
	0x08,0x00,0xF8,0x3F,0x00,0x11,0x80,0x20,0x80,0x20,0x00,0x11,0x00,0x0E,0x00,0x00,/*"b",38*/
	0x08,0x00,0xF8,0x3F,0x00,0x11,0x80,0x20,0x80,0x20,0x00,0x11,0x00,0x0E,0x00,0x00,/*"b",39*/
	0x00,0x00,0x00,0x33,0x80,0x24,0x80,0x24,0x80,0x24,0x80,0x24,0x80,0x19,0x00,0x00,/*"s",40*/
	0x00,0x00,0x00,0x60,0x00,0x18,0x00,0x06,0x80,0x01,0x60,0x00,0x18,0x00,0x04,0x00,/*"/",41*/
	0x00,0x00,0x80,0x20,0x98,0x20,0x98,0x3F,0x00,0x20,0x00,0x20,0x00,0x00,0x00,0x00,/*"i",42*/
	0x80,0x20,0x80,0x3F,0x00,0x21,0x80,0x00,0x80,0x00,0x80,0x20,0x00,0x3F,0x00,0x20,/*"n",43*/
	0x00,0x00,0x00,0x0E,0x00,0x11,0x80,0x20,0x80,0x20,0x88,0x10,0xF8,0x3F,0x00,0x20,/*"d",44*/
	0x00,0x00,0x00,0x1F,0x80,0x22,0x80,0x22,0x80,0x22,0x80,0x22,0x00,0x13,0x00,0x00,/*"e",45*/
	0x00,0x00,0x80,0x20,0x80,0x31,0x00,0x2E,0x80,0x0E,0x80,0x31,0x80,0x20,0x00,0x00,/*"x",46*/
	0x00,0x00,0x00,0x30,0x00,0x30,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,/*".",47*/
	0x80,0x80,0x80,0xFF,0x00,0xA1,0x80,0x20,0x80,0x20,0x00,0x11,0x00,0x0E,0x00,0x00,/*"p",48*/
	0x08,0x20,0xF8,0x3F,0x00,0x21,0x80,0x00,0x80,0x00,0x80,0x20,0x00,0x3F,0x00,0x20,/*"h",49*/
	0x80,0x80,0x80,0xFF,0x00,0xA1,0x80,0x20,0x80,0x20,0x00,0x11,0x00,0x0E,0x00,0x00,/*"p",50*/
	0x00,0x00,0x00,0x01,0x00,0x01,0x00,0x01,0x00,0x01,0x00,0x01,0x00,0x01,0x00,0x01,/*"-",51*/
	0x00,0x00,0x00,0x01,0x00,0x01,0x00,0x01,0x00,0x01,0x00,0x01,0x00,0x01,0x00,0x01,/*"-",52*/
	0x40,0x80,0x48,0x60,0x48,0x1F,0x48,0x20,0xFF,0x7F,0x48,0x44,0x48,0x44,0x00,0x40,
	0xC4,0x4F,0x44,0x50,0x44,0x50,0x44,0x50,0xFC,0x50,0x00,0x5C,0x00,0x40,0x00,0x00,/*"起",53*/
	0x80,0x80,0xFC,0x7F,0x96,0x02,0xE5,0x4C,0x84,0x80,0xFC,0x7F,0x00,0x80,0x08,0x60,
	0xC8,0x1F,0x49,0x00,0x4A,0x00,0xC8,0x3F,0x08,0x40,0x08,0x40,0x00,0x78,0x00,0x00,/*"航",54*/
	0x00,0x00,0x00,0x00,0xF8,0x1F,0x88,0x08,0x88,0x08,0x88,0x08,0x88,0x08,0xFF,0x7F,
```

```
        0x88,0x88,0x88,0x88,0x88,0x88,0x88,0x88,0xF8,0x9F,0x00,0x80,0x00,0xF0,0x
00,0x00,/*"电",55*/
        0x80,0x00,0x82,0x00,0x82,0x00,0x82,0x00,0x82,0x00,0x82,0x40,0x82,0x80,0x
E2,0x7F,
        0xA2,0x00,0x92,0x00,0x8A,0x00,0x86,0x00,0x82,0x00,0x80,0x00,0x80,0x00,0x
00,0x00,/*"子",56*/
        };
        void    hc595_senddat(unsigned char dat)
        {
        unsigned char i;
        for(i=0;i<8;i++)
        {
              digitalWrite(DS,dat&0x80);// DS=dat&0x80;
              digitalWrite(SH_CP,HIGH);// SH_CP=1;
              digitalWrite(SH_CP,LOW);// SH_CP=0;
               dat<<=1;
        }
        }
        void setup()
        {
          unsigned char i;
          for(i=1;i<=7;i++)//依次设置 1~7 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()
        {
        unsigned char i,n;
        unsigned int k,m;
        char j;
        m=sizeof(display_1);
        digitalWrite(SH_CP,LOW);      //SH_CP=0;
        digitalWrite(ST_CP,LOW);       //ST_CP=0;
        while(1)
        {
          for(k=0;k<m;k=k+2)
          for(n=0;n<2;n++)
          for(i=0;i<16;i++)
          {
           for(j=3;j>=0;j--)
           {
            hc595_senddat(~display_1[(32*j+2*i+1+k)%m]);
            hc595_senddat(~display_1[(32*j+2*i+k)%m]);
           }
           switch(i)
```

```
{
case 0:digitalWrite(A,LOW);digitalWrite(B,LOW);digitalWrite(C,LOW);digitalWrite(D,LOW); break;  //译码器 ABCD 地址端为 0000
case 1:digitalWrite(A,HIGH);digitalWrite(B,LOW);digitalWrite(C,LOW);digitalWrite(D,LOW); break; //1000
case 2:digitalWrite(A,LOW);digitalWrite(B,HIGH);digitalWrite(C,LOW);digitalWrite(D,LOW); break;  //0100
case 3:digitalWrite(A,HIGH);digitalWrite(B,HIGH);digitalWrite(C,LOW);digitalWrite(D,LOW); break; //1100
case 4:digitalWrite(A,LOW);digitalWrite(B,LOW);digitalWrite(C,HIGH);digitalWrite(D,LOW); break; //0010
case 5:digitalWrite(A,HIGH);digitalWrite(B,LOW);digitalWrite(C,HIGH);digitalWrite(D,LOW); break; //1010
case 6:digitalWrite(A,LOW);digitalWrite(B,HIGH);digitalWrite(C,HIGH);digitalWrite(D,LOW); break; //0110
case 7:digitalWrite(A,HIGH);digitalWrite(B,HIGH);digitalWrite(C,HIGH);digitalWrite(D,LOW); break; //1110
case 8:digitalWrite(A,LOW);digitalWrite(B,LOW);digitalWrite(C,LOW);digitalWrite(D,HIGH); break;  //0001
case 9:digitalWrite(A,HIGH);digitalWrite(B,LOW);digitalWrite(C,LOW);digitalWrite(D,HIGH); break; //1001
case 10:digitalWrite(A,LOW);digitalWrite(B,HIGH);digitalWrite(C,LOW);digitalWrite(D,HIGH); break; //0101
case 11:digitalWrite(A,HIGH);digitalWrite(B,HIGH);digitalWrite(C,LOW);digitalWrite(D,HIGH); break; //1101
case 12:digitalWrite(A,LOW);digitalWrite(B,LOW);digitalWrite(C,HIGH);digitalWrite(D,HIGH); break; //0011
case 13:digitalWrite(A,HIGH);digitalWrite(B,LOW);digitalWrite(C,HIGH);digitalWrite(D,HIGH); break; //1011
```

```
                                                case
14:digitalWrite(A,LOW);digitalWrite(B,HIGH);digitalWrite(C,HIGH);digitalWrite(D,HIG
H); break; //0111
                                                case
15:digitalWrite(A,HIGH);digitalWrite(B,HIGH);digitalWrite(C,HIGH);digitalWrite(D,HIG
H); break; //1111
                                            }
                digitalWrite(ST_CP,HIGH);//ST_CP=1;
                digitalWrite(ST_CP,LOW);//ST_CP=0;
              }
            }
            }
```

【视频】

大学生电子实验室制作
http://v.youku.com/v_show/id_XMjg3MDUwMzYw.html

52 方案秀，汇集热门方案，分享设计思想
http://v.youku.com/v_show/id_XMjg3MDk3OTQw.html

这是一本非常全面的讲述 Arduino 应用开发的书，它从 Arduino 的 I/O 板和最基本的 C 语言入门开始，仔细介绍了 Arduino 库和十多个第三方开源库及其对应的外围设备的使用与编程。第三部分用两个份量充足的例子展示了 Arduino 作品开发的过程，也体现了 Arduino 的强大潜力。本书适合 Arduino 的初学者作为循序渐进的教材，也适合深入的开发者进一步学习，并用作参考手册。

——浙江大学计算机学院 Arduino 创新教学实践者 翁恺老师

Arduino 是一个开源硬件平台，电子专业的学生完全可以通过查资料、买元件、做 PCB 板、焊电路，制作自己的 Arduino 硬件模块；同时，很多厂商也开发了各种各样的 Arduino 外围功能电路供学生选择，无论是电机驱动、无线通信、音乐播放，还是各种传感器（压力、速度、倾角、方向等），这些均为学生在学习和设计与自动控制、物联网、无线传感网相关的知识提供了不同的学习途径，并且使得学习电子知识变得相对容易。另一方面，Arduino 的代码语法简单易懂，对于学过 C 语言程序设计、甚至没有任何编程经验的读者来说，Arduino 程序也是简单易读的。因此，我觉得这本书非常适合作为 Arduino 爱好者的参考教材，并且很适合初学者入门学习。同时，通过全面系统介绍 Arduino 及其开发方法，本书也为电子和计算机类专业低年级学生打开了一扇兴趣之门，书中丰富的实例更是增强学生动手能力不可多得的素材。

——西安邮电大学计算机学院周立功“3+1”创新教育实验班班主任 马博老师

这是一本关于 Arduino 及其开发方法的书。本书内容涵盖广泛，但又不失重点。通过系统的理论知识介绍和精彩的实例讲解，将 Arduino 这个开源硬件平台阐释的淋漓尽致。这本书的内容包括 Arduino 的前世今生、C 语言基础、Arduino 开发平台的使用以及实战项目。与之前看过的几本国外直接翻译的 Arduino 书籍相比，这本书更加生动。读者即使原来没有在嵌入式平台上编写过软件，通过这本书也可以学会 Arduino 的开发方法，实现自己的产品创意，这也是 Arduino 的魔力所在。本书作者有非常丰富的 Arduino 项目开发经验，后面几个章节的项目记录了作者使用 Arduino 的开发心得。通过阅读此书相信你也可以 DIY 出充满创意的产品原型！

——DFRobot 创始人，Arduino 首批引入者之一 庄明波

内容提要

目前，在国内关注 Arduino 的人越来越多，但介绍 Arduino 的书籍却很少。Arduino 是一个注重实际动手操作应用的产品，所以本书以实际应用为纽带将各个章节联系起来。本书首先介绍 Arduino 的一些基础知识，接着针对具体应用介绍了一些扩展板以及 Arduino 扩展库，最后应用之前的内容完成了具有视频监控功能的履带车、遥控机械臂以及双足机器人的制作。内容是循序渐进，使读者深刻的理解 Arduino 的优点，本书会引领您走入 Arduino 的精彩世界。

作者简介

程晨（网名晨光熹微） 国内较早接触 Arduino 的硬件工程师，在 Arduino 的应用上有着丰富的实战经验。对 Arduino 的底层代码进行了长达一年的学习与研究，同时使用过大量的 Arduino 的类库，对于 Arduino 的架构和实现原理有着非常深入的理解和认识。同时在 PC 端、手机端的应用程序开发方面也有一定的经验。应用 Arduino 进行过多款交互式制作。

前言

2011 年举行的 Google I/O 开发者大会上，Google 发布了基于 Arduino 的 Android Open Accessory 标准和 ADK 工具，这使得大家对 Arduino 的前景十分看好。Phillip Torrone 大胆的预测 Google 将用 Android+Arduino 的形式掀起自己的“Kinect 模式”浪潮。目前，在国

内关注 Arduino 的人越来越多，但介绍 Arduino 的书籍却很少。本人由于工作的关系，接触 Arduino 较早，所以希望通过自己的努力让更多的人了解 Arduino，在 2011 年近一年的时间里，通过不断的学习、查阅 Arduino 相关知识，终于完成了书稿的撰写工作。但在书稿完成之后，心中却一直忐忑不安，Arduino 是一个介于软件与硬件之间的一个产品，系统性不是很强，加上本人水平有限，拙著中一定存在不少的缺点与漏洞，为此，本人先为书中的不足之处致以真诚的歉意，同时诚挚的欢迎广大的读者提出宝贵的意见并不吝赐教。

本书的内容及面向的读者

Arduino 是一个注重实际动手操作应用的产品，所以本书以实际应用为纽带将各个章节联系起来。本书共 9 章，首先介绍 Arduino 的一些基础知识，接着针对具体应用介绍了一些扩展板以及 Arduino 扩展库，最后应用之前的内容完成了具有视频监控功能的履带车、遥控机械臂以及双足机器人的制作。内容是循序渐进，使读者深刻的理解 Arduino 的优点。

由于 Arduino 本身简单易用的特点，本书所面向的读者是所有有兴趣使用 Arduino 进行项目开发的人。

当然根据读者的情况不同，本书的阅读方式也不同。

如果读者是一个之前没有进行过单片机开发也没有进行过软件开发的人，现在想使用 Arduino 来实现自己的一些想法，那么首先要阅读本书的前两章，了解一些简单的编程思想以及程序结构，接着看一下第三章的目录，了解 Arduino 都有哪些基本函数，书中的内容都可以不用看，当你之后用到这些函数遇到问题时可以在翻过头来看一看相应的函数说明。然后翻开第四章，将 Arduino 接到你的电脑上，根据书上的内容，边学习边实践，完成第四章的内容，在这一章里 4.5 节可以跳过不看。5、6、7 章会帮助你了解 Arduino 周边都有什么资源能够帮助你尽快的实现你的想法，这 3 章的内容也可以采用跳跃式的阅读方式，8、9 章的内容会告诉你前三章的内容是如何结合起来的，建议按照书中的内容至少动手实践完成一个项目的制作。

如果读者之前进行过 AVR 单片机的开发，想了解 Arduino 一些底层的知识，那么第二章的知识就可以跳过了，在简单的翻阅一下第三章的内容后，直接进入第四章，把 Arduino 连到电脑上实践一把，再翻过来阅读第三章 Arduino 的基本函数，结合自身已有的 AVR 单片机的知识，了解 Arduino 底层的工作机制，需要说明的是这里需要读者自己花一些精力，可能还需要学习一些 C++方面的知识，第五章 Arduino 扩展板的内容本书中也对硬件原理有详细的介绍，若读者之前学习过，这一章可以选择性的学习，第六章 Arduino 的扩展库，如果读者也想开发一些 Arduino 扩展板，并以库的形式提供扩展板的软件资源，那么建议先学习最后一节，再从第一节开始学习，深入的了解这些扩展库是如何与 Arduino 结合在一起的，剩下几个章节的内容，如果用开发单片机的思路来完成也是不难的，所以阅读的重点是看看如何用 Arduino 的思路进行项目的制作。

如果读者之前是做纯电脑软件开发的，C++用的很好那种，那么在阅读完第一章后，可以直接跳到第四章，感受一下 Arduino 给纯软件开发人员带来的那种完成硬件制作的感觉，然后仔细阅读第五章，看看目前都有哪些扩展板可以为我们所用，控制电机、控制液晶之类

的，硬件知识哪怕我们不用，也还是要了解一些的。接下来第六章，可以仔细阅读一下与硬件关系不太大的扩展库以及如何创建自己的库，在今后底层硬件库不断丰富完善的情况下，开发一些注重应用、与底层关系不是太紧密的库时就是我们的用武之地了。7、8、9 章的内容会告诉我们前面的知识是如何结合起来的——用纯软件的思路，同样建议按照书中的内容至少动手实践完成一个项目的制作，纯软件的人开发硬件也是很 easy 的。

致谢

首先要感谢本书的编辑张国强先生，是他对 Arduino 的关注促成了这本书的出版，同时本人在撰写书稿时也对本书提出了宝贵的写作建议，并对书稿进行了仔细的审阅。

其次要感谢让我了解 Arduino 的庄明波先生，他不但在技术上给了我很多的指导，同时也无私的提供了大量的 Arduino 扩展板的资料以及实物，供本人在 Arduino 的程序调试中使用，当我在技术上遇到问题时，就会和他共同探讨研究。

最后要感谢现在正捧着这本书的您，感谢您肯花费时间和精力阅读本书，由于时间有限，书中难免存在疏漏与错误，诚恳的希望您批评指正，您的意见和建议将是我巨大的财富。希望在 Arduino 的领域结识更多的朋友。

关于 Arduino

Arduino 是源自意大利的一个开源代码的硬件项目，该平台包括一块具备简单 I/O 功能的电路板以及一套程序开发环境软体。Arduino 可以用来开发交互产品，比如它可以读取大量的开关和传感器信号，并且可以控制各式各样的电灯、电机和其他物理设备，Arduino 也可以开发出与 PC 相连的周边装置，能在运作时与 PC 上的软体进行沟通。Arduino 的硬体电路板可以自行焊接组装，也可以购买已经组装好的模块，而程序开发环境的软件则可以从网上免费下载与使用。2011 年举行的 Google I/O 开发者大会上，Google 发布了基于 Arduino 的 Android Open Accessory 标准和 ADK 工具，这使得更多的人开始关注 Arduino，国内也有不少工程师加入到 Arduino 爱好者的阵营中来。

如果你也想方便快捷的开发自己的交互式产品，无论是在嵌入式方面还是移动终端方面，甚至是航模、玩具方面，那么开始阅读这本书吧，本书会引领您走入由 Arduino 构建的精彩世界。

除了书中提到的机械臂、遥控小车、双足机器人之外，让我们看看其他人都用 Arduino 作了什么

●3D 打印机

三维立体打印的技术是利用普通打印机的原理，通过计算机的控制，用激光注射器将原料一层一层累积起来，最后将计算机上的蓝图变成实物。3D 打印机在 90 年代中期就出现了，但价格一直很昂贵，现在 Arduino 让 3D 打印机变成了普通人都可以使用的设备。具体资料可以访问 http://www.douban.com/note/125874839

●互动式烹饪锅

有没有想过使用手机也能烹饪，有人就用 Arduino 做了一个互动式烹饪锅，想想坐在沙发上就能做菜是多么惬意的事情呀。具体资料可以访问：

http://toydl.kmd.keio.ac.jp/index.php/en/projects/yaminabe-yammy

●远程遥控的电源插排

智能家居的第一步恐怕就是控制家电的电源了吧，不能在家电内部作改造，改造一个可远程遥控的电源插排还是可以的，Arduino 又出场了。具体资料可以访问：

http://www.practicalarduino.com/projects/appliance-remote-control

●示波器/逻辑分析仪

用 Arduino 制作的模拟口制作一台示波器/逻辑分析仪，虽然带宽不是很宽，但还是感觉很酷。具体资料可以访问 http://www.practicalarduino.com/projects/scope-logic-analyzer

●引擎信息采集系统

通过一个基于 Arduino 的平台记录一台汽车出行一趟每时每刻的速度值、温度、转速等信息，同时结合 google earth 制作出这台汽车在城市中的行驶情况。具体资料可以访问

http://www.practicalarduino.com/projects/vehicle-telemetry-platform

●私人气象站

使用 Arduino 通过无线的方式采集所居住环境的温度、湿度、风力等等信息，而不是依靠国家和城市的气象服务，让我们真正了解周围环境的变化。具体资料可以访问 http://www.practicalarduino.com/projects/weather-station-receiver

●八音盒

仅仅使用蜂鸣器就可以让 Arduino 发出动听的声音，自动动手制作一个八音盒送人一定会让别人的礼品黯然失色。具体资料可以访问：

http://www.52solution.com/bbs/viewthread.php?tid=1768&extra=page%3D1

●自动狗食盆

主人外出，家里宠物每人照看，那就自己动手做个自动狗食盆吧，让 Arduino 来照看你的宠物吧，当狗粮不足时还可以发出声音提示补充饲料。具体资料可以访问：

http://www.wshop.idv.tw/interaction/?p=379

●两轮平衡车

使用 Arduino 和乐高积木搭建自己的两轮平衡车。具体资料可以访问：

http://www.52solution.com/bbs/viewthread.php?tid=1845&extra=page%3D1

●还有更多的应用等待你去发现或实现.....

嵌入式计算系统设计原理（第2版）
作者：Wayne Wolf
译者：李仁发 等
ISBN：978-7-111-27068-3
定价：55.00元

高性能嵌入式计算
作者：Wayne Wolf
译者：鞠大鹏 王海霞 汪东升 等
ISBN：978-7-111-28822-0
定价：65.00元

嵌入式系统导论：CPS方法
作者：Edward Ashford Lee, Sanjit Arunkumar Seshia
译者：李实英 贾香 李仁发
ISBN：978-7-111-36021-6
定价：55.00元

嵌入式微控制器与处理器设计
作者：Greg Osborn
译者：宋廷强 高树静
ISBN：978-7-111-32281-8
定价：59.00元

编译原理（原书第2版）
作者：Alfred V. Aho, Monica S. Lam 等
译者：赵建华 等
ISBN：978-7-111-25121-7
定价：89.00元

人工智能：复杂问题求解的结构和策略（原书第6版）
作者：George F. Luger
译者：郭茂祖 刘扬 玄萍 王春宇 等
ISBN：978-7-111-28345-4
定价：79.00元

C++程序设计原理与实践
作者：Bjarne Stroustrup
译者：王刚 刘晓光 吴英 李涛
ISBN：978-7-111-30322-0
定价：108.00元

C程序设计语言（第2版·新版）
作者：Brian W. Kernighan, Dennis M. Ritchie
译者：徐宝文 李志
ISBN：978-7-111-12806-0
定价：30.00元