第 3 章

生活便捷类开发案例

3.1 项目 9：“懒人”垃圾桶

设计者：黄婷，托雅，徐淼

3.1.1 项目背景

“垃圾桶”作为家居生活中不可或缺的必需品，它伴随着人类走过了每一个时代。特别是在如今的智能居家、万物互联快速发展的情况下，更是必不可少。另外，随着人们环保意识和审美水平的普遍提高，垃圾桶的种类和数量也在不断地增加，人们更加注重它的美观和实用性，而产品也向着“小巧”和“智能化”方向发展。所以，本项目在现有的基础上，试图开发垃圾桶的一些新功能。例如，垃圾桶可以实现语音识别功能、避障功能和测满功能。

3.1.2 创意描述

本项目的作品最重要的创新点在于应用了语音识别模块，可以在室内较安静的环境中实现垃圾桶“随叫随到”的功能。同时，应用了红外避障模块，使垃圾桶在受到“召唤”之后，可以准确地来到使用者面前。最后，本项目运用了超声波测距模块，以此来测量垃圾桶内垃圾的高度，并在超过限定的高度后，发出警报。

3.1.3 功能及总体设计

本项目的“懒人”垃圾桶可以通过用户的语音进行控制，达到随叫随到的效果，因此在普通的垃圾桶上，除了加入语音识别的功能之外，还将智能控制小车与垃圾桶结合起来。最后，为了使垃圾桶更加智能，加入了超声波避障以及人体感应的功能。

1. 功能介绍

本项目主要有以下四大功能：

(1) 语音识别。识别语音“发动”而开启自动避障模式；识别语音“刹车”而制动。

(2) 自动避障。前进过程中，根据安装在车体前、左、右的三个红外探测器进行障碍物

检测，进而完成自动避障功能。

（3）垃圾高度自动检测报警。垃圾桶桶壁的超声波测距模块每 2s 检测一次，如果垃圾达到限定的高度，则触发蜂鸣器，给出提示音。

（4）人体识别。垃圾桶的车体自动避障过程中，如果周围有人走动，则自动刹车。

2．总体设计

本项目的“懒人”垃圾桶主要完成语音识别下的智能小车自动避障、垃圾达到限定高度自动报警功能的智能垃圾桶的搭建，共包括语音识别、电机驱动、红外避障、人体感应、超声波测距、车体共六个模块。

1）整体框架图

项目整体框架图如图 3-1 所示。

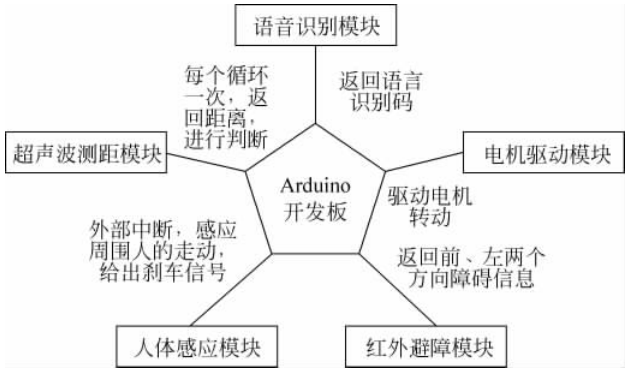

图 3-1　整体框架图

2）系统流程图

系统流程图如图 3-2 所示。

3）总电路图

系统总电路接线图如图 3-3 所示，系统总电路原理图如图 3-4 所示。

3．模块介绍

本项目的“懒人”垃圾桶主要由语音识别模块、红外避障模块、电机驱动模块、超声波测距（垃圾高度报警）模块、人体感应模块和车体组成。

1）语音识别模块

功能介绍：通过识别输入的语音，返回指定信号代码给 Arduino 开发板，供单片机根据返回值执行相应程序功能。本程序中完成“发动”两个汉字的识别，返回十六进制数 0X00 给 Arduino 开发板，开发板接收到 0X00 后，驱动小车进行智能避障；完成“刹车”两个汉字的识别，返回十六进制数 0X01 给 Arduino 开发板，单片机收到 0X01 后，驱动小车制动。该模块有 VCC、GND、ICR、RX、TX、MIP（MIC＋输出）、MIN（MIC-输出）共七个引脚，其中 MIP 与 MIN 两个引脚为麦克风的连接引脚，ICR 与 RX 在本项目中没有使用。语音识别模块引脚说明如表 3-1 所示。

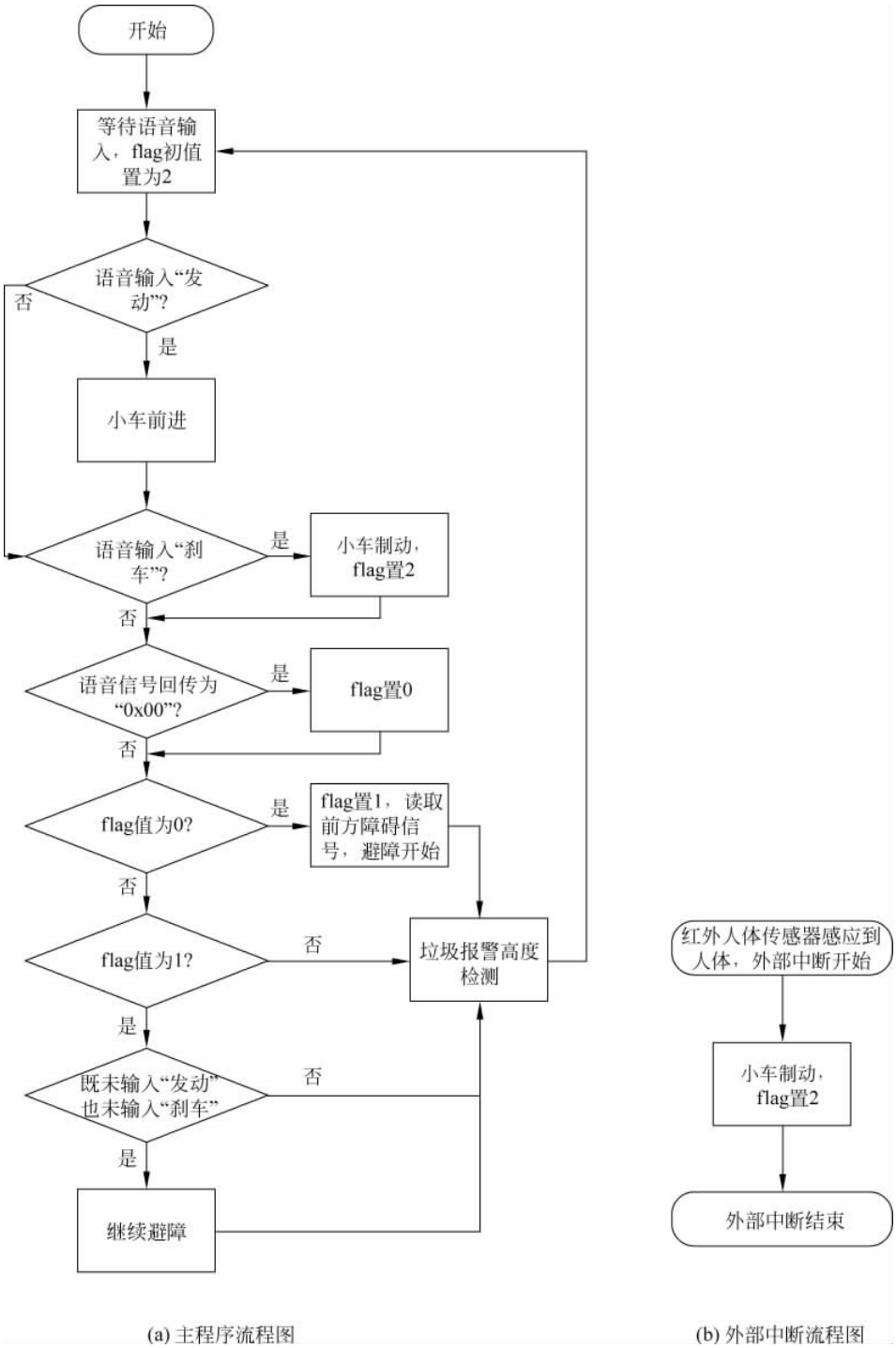
开始
等待语音输入，flag初值置为2
语音输入“发动”?
否
是
小车前进
语音输入“刹车”?
是
小车制动，flag置2
否
语音信号回传为“0x00”?
是
flag置0
否
flag值为0?
是
flag置1，读取前方障碍信号，避障开始
否
flag值为1?
否
垃圾报警高度检测
是
既未输入“发动”也未输入“刹车”
否
是
继续避障
(a) 主程序流程图
红外人体传感器感应到人体，外部中断开始
小车制动，flag置2
外部中断结束
(b) 外部中断流程图

图 3-2　系统流程图

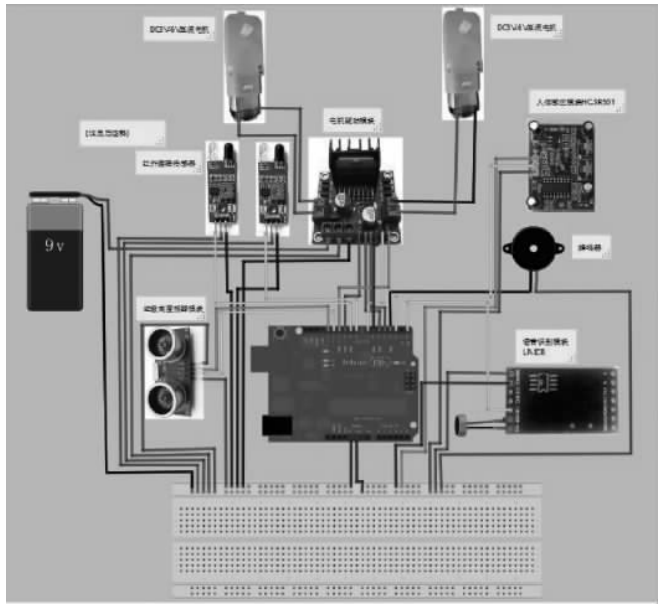

图 3-3　总电路接线图

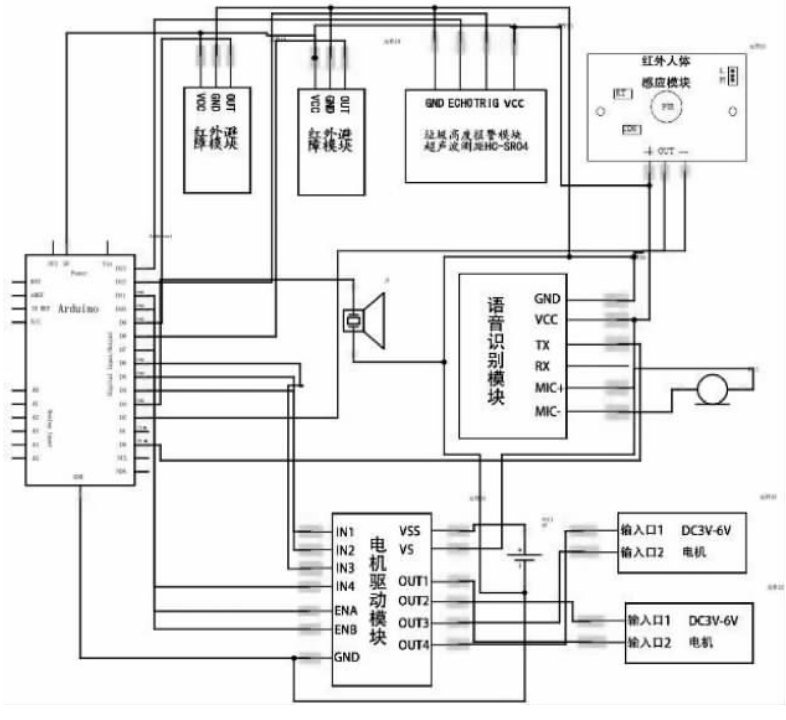

图 3-4　总电路原理图

表 3-1　语音识别模块引脚说明

引　脚	用　途
VCC	电源正极
GND	接地
TX	识别语音后发送特定返回码，接 Arduino 开发板 0 号端口
MIP	MIC 正极
MIN	MIC 负极

元器件清单：语音识别模块元器件及其数量如表 3-2 所示。

表 3-2　语音识别模块元器件清单

元器件名称	数　量
语音识别模块 LP-ICR	1
测试 MIC	1
RS232-TTL 模块	1
杜邦线	若干
VGA-USB 线	1
上位机语音识别烧写软件	1

电路图：语音识别模块电路连接图如图 3-5 所示，语音识别模块电路原理图如图 3-6 所示。

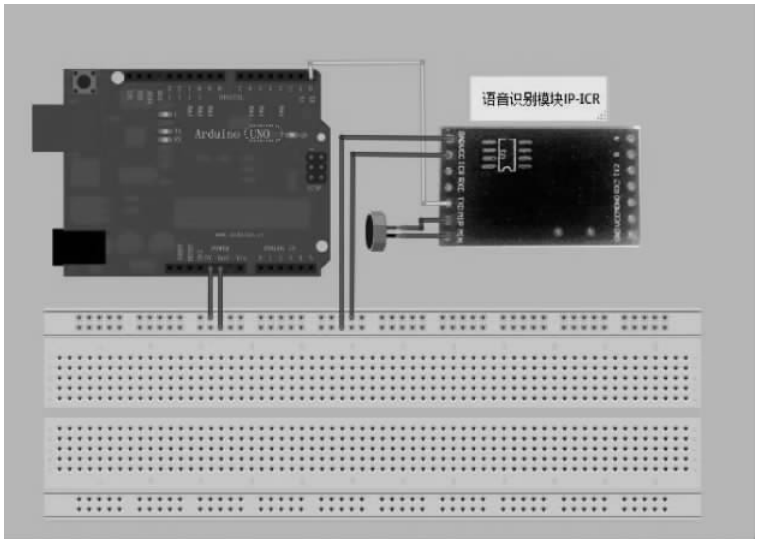

图 3-5　语音识别模块电路连接图

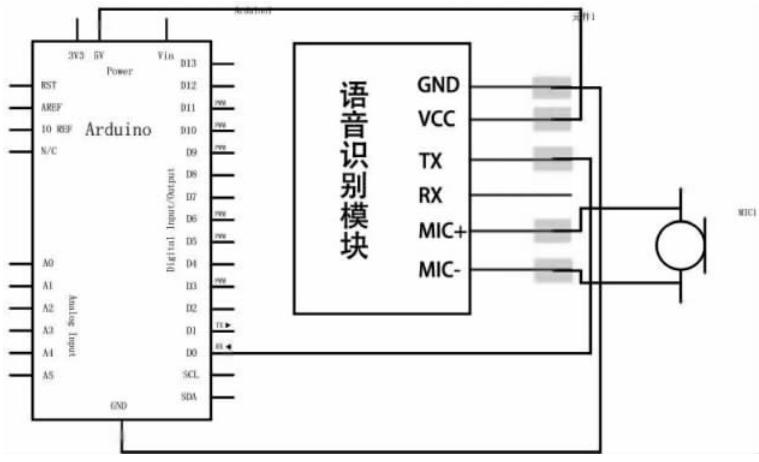

图 3-6　语音识别模块电路原理图

相关代码：

（1）在上位机语音识别软件上的烧写代码，语法与 Arduino 开发板不同：

```
{d1}          //打开调试模式
{c0}          //清除原有语句列表
{a0ni hao}    //添加"你好"识别语句，该模块识别普通话的"你好"发音，不用考虑音调。
              //可以指定返回值，默认为该识别语句十六进制下的语句序号
{d0}          //关闭调试模式
```

（2）Arduino 使用代码，根据不同返回值确定调用什么函数：

```
//主循环函数中读取语音输入信号，根据输入信号选择发动、刹车、默认退出
if(Serial.available())
  {
    inByte = Serial.read();        //读取语音输入信号返回值
    switch(inByte)
      {
        case 0x00:                 //对应"发动"，小车启动，自动避障开始
          forward();               //开始前进
          break;
        case 0x01:                 //对应"刹车"，小车制动
          brake();
          flag = 2;
          break;
        default:                   //默认情况，退出此次判断
          break;
      }
  }
```

2）红外避障模块

功能介绍：小车运行后，位于小车车前、左两个方向各有一个红外传感器，当前方有障碍物时，返回低电平信号；当前方无障碍物时，返回高电平信号。根据红外传感器的原理进行避障，逻辑：直行＞左转＞右转＞后退（＞表示优先），以 1 代表有障碍物，以 0 代表无障碍物，以 X 代表不考虑。

使用两个红外传感器，避障逻辑如下：在主函数的每个循环，检测一次车前的红外传感器，如果检测到有障碍物，则避障开始。检测一次车体左方的传感器信号，若无障碍，则左转；若有障碍，则右转。然后，再次检测车前方的传感器，若无障碍，则本次避障结束，小车恢复前行；若有障碍，则小车直接右转，且右转后前行。

小车右转是在前方、左方都有障碍物的前提下进行的，右转后只需再判断前方障碍就可以决定是右转后直接前行还是继续右转（两次右转等效于后转）。

红外传感器避障实现逻辑如表 3-3 所示。红外避障模块引脚说明如表 3-4 所示。

表 3-3　红外传感器避障实现逻辑

前方	左方	右　方	操　作
0	X	X	直行
1	0	X	左转
1	1	0	右转
1	1	1	后转

表 3-4　红外避障模块引脚说明

引　脚	用　途
VCC	电源正极
GND	接地
OUT	输出，接 Arduino 数字端口，无障碍输出 0，有障碍输出 1

元器件清单：红外避障模块所需元器件及其数量如表 3-5 所示。

表 3-5　红外避障模块元器件清单

元器件名称	数　量
红外传感器模块	2
杜邦线	若干

电路图：红外避障模块电路连接图如图 3-7 所示，红外避障模块电路原理图如图 3-8 所示。

相关代码：

在主循环函数中，首先读取语音输入信号，根据输入信号选择发动、刹车、默认退出。由于没有语音输入信号时，串口读不到数据，为防止小车在停止状态下出现避障行为，设置 flag 值来标记小车当前的运行状态。flag 值为 0 对应输入发动的第一个循环，遇到障碍信

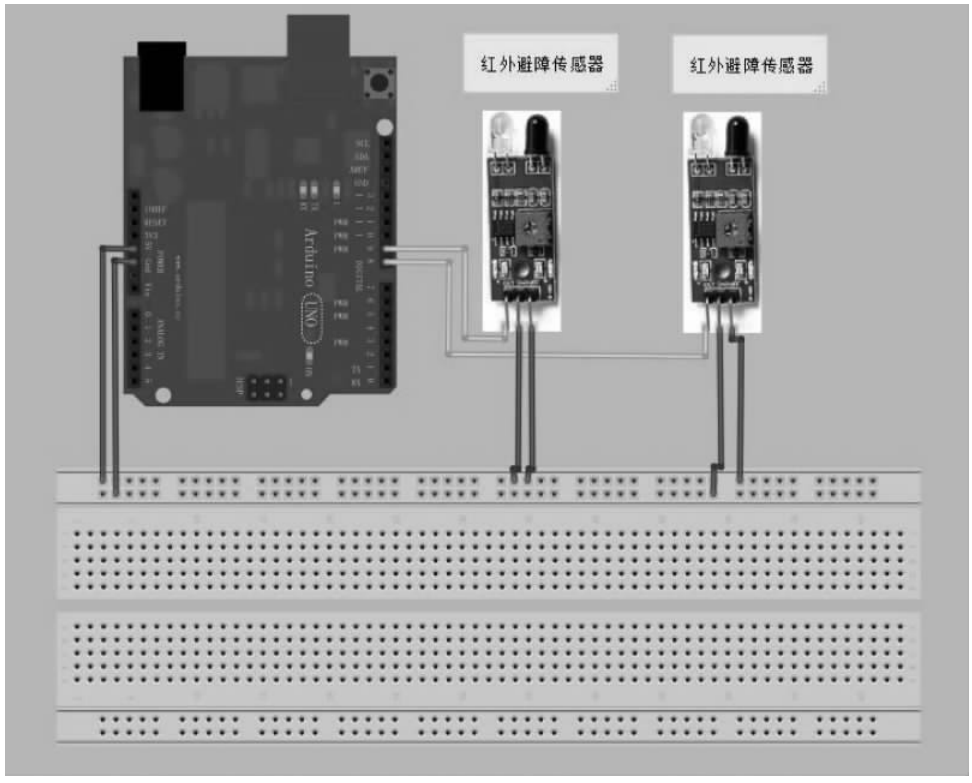

图 3-7　红外避障模块电路连接图

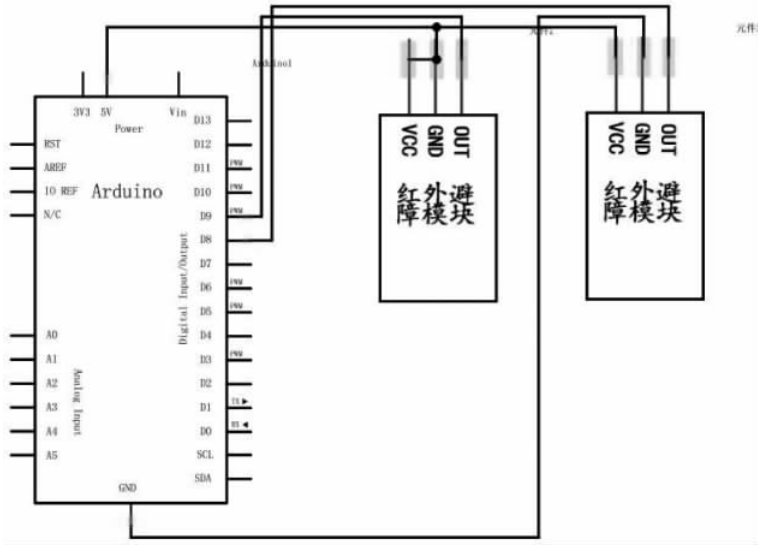

图 3-8　红外避障模块电路原理图

号小车正常进行避障；flag 值为 1 对应输入过发动、没输入刹车但在正常循环中没有输入的情况，遇到障碍信号小车正常进行避障；2 为 flag 的默认值，对应没输入过发动且读不到信号的情况，遇到障碍信号小车不进行避障。在每一次循环中：

```
int infrared_front = 8;     //前方红外避障传感器
int infrared_left = 9;      //左侧红外避障传感器
int inByte = 0X02;          //语音信号返回值,默认为 0X02
int flag = 2;               //一个标签变量,用于判断是否为刚输入语音"发动"。"发动"情况
                            //下为 0,"刹车"情况下为 1
pinMode(infrared_front ,INPUT);
pinMode(infrared_left ,INPUT);
 if (0X00 == inByte)        //输入"发动",将 flag 置 0
 {
    flag = 0;
 }
 switch(flag)
 {
    case 0:                 //第一次输入"发动",进行避障
      {
        sig_front = digitalRead(infrared_front);   //检测前方障碍信号
        if (0 == sig_front)                        //前方有障碍,避障开始
       {
         avoidence();
       }
        flag = 1;                                  //flag 置 1,表示自动避障已经开始
        break;
      }
    case 1://在输入"发动",未输入其他指令时,语音接收可能存在扰乱的情况
    {
   if ((inByte != 0X00)&(inByte != 0X01)) //既不是"发动",也不是"刹车",是干扰,继续进行避障
       {
        sig_front = digitalRead(infrared_front);
        if (0 == sig_front)
          {
           avoidence();
          }
       }
       break;
    }
    default:
      break;
 }void avoidence()                                  //避障模块
{
    brake();                                       //避障前先进行短时刹车,减少电机损耗
    delay(500);
    boolean sig_left = digitalRead(infrared_left); //读取左侧障碍物信号
```

```
    if (1 == sig_left)                              //左侧无障碍物,左转
    {
      turn_left();
    }
    else                                            //左侧有障碍物,右转
    {
      turn_right();
    }
    brake();
    delay(500);
    sig_front = digitalRead(infrared_front);        //读取前方障碍物
    if (0 == sig_front)              //前方有障碍物,在右转的情况下继续右转,等效于向后转
    {
      turn_right();
    }
    brake();
    delay(100);
    forward();                       //恢复转完前状态,继续前进
}
```

每个循环先判断当前小车运行状态,如果在前进过程中,则读取位于小车前方、左方的两个红外线传感器的探测值,总体逻辑:前方传感器返回值优先级左侧传感器返回值。前方无障碍物则继续前行;前方有障碍物、左侧无障碍物则左转;前方有障碍物、左侧也有障碍物则右转,且右转后不前行,继续查看前方红外传感器的返回值:无障碍物则前行,有障碍物则继续右转(两次右转相当于后转)。

```
void avoidence()                                    //避障模块
{
    brake();                                        //避障前先进行短时刹车,减少电机损耗
    delay(500);
    boolean sig_left = digitalRead(infrared_left); //读取左侧障碍物信号
    if (1 == sig_left)                              //左侧无障碍物,左转
    {
      turn_left();
    }
    else                                            //左侧有障碍物,右转
    {
      turn_right();
    }
    brake();
    delay(500);
    sig_front = digitalRead(infrared_front);        //读取前方障碍物
    if (0 == sig_front)              //前方有障碍物,在右转的情况下继续右转,等效于向后转
    {
      turn_right();
    }
```

```
    brake();
    delay(100);
    forward();                          //恢复转完前状态,继续前进
}
```

3）电机驱动模块

功能介绍：驱动电机转动，配合避障模块完成自动避障功能。模块共有 6 个信号输入端对信号输入进行控制，3 个供电输入端（5V、12V 可选）对模块自身进行供电，4 个供电输出端驱动电机进行正向或反向旋转。

对每个电机，由 1 个输入使能端、2 个接口端进行控制，以 1 代表高电平，0 代表低电平，X 代表不考虑。控制逻辑如表 3-6 所示。

表 3-6　电机驱动模块控制小车运动实现逻辑

使能端 ENA	输入端口 IN1	输入端口 IN2	电 机 操 作
0	X	X	自由
1	1	0	向前转
1	0	1	向后转
1	0	0	制动
1	1	1	制动

共有 10 个引脚需要连接，引脚说明如表 3-7 所示。

表 3-7　电机驱动模块引脚说明

引　　脚	用　　途
ENA/ENB	使能端口，控制电机自由或转动
IN1～IN4	IN1 与 IN2 为一组，IN3 与 IN4 为一组，连接 Arduino 引脚，以此控制电机转动方向
GND	接地
5V 供电端	接 VCC，供电
输出 A/输出 B	各两个输出端，对应电机两个端口

元器件清单：该模块所需的元器件及其数量如表 3-8 所示。

表 3-8　电机驱动模块实现功能所需元器件

元器件名称	数　　量
电机驱动模块	1
DC3V-6V 直流电机	2
电线	4
Arduino 开发板	1

电路图：电机驱动模块电路连接图如图 3-9 所示，电机驱动模块电路原理图如图 3-10 所示。

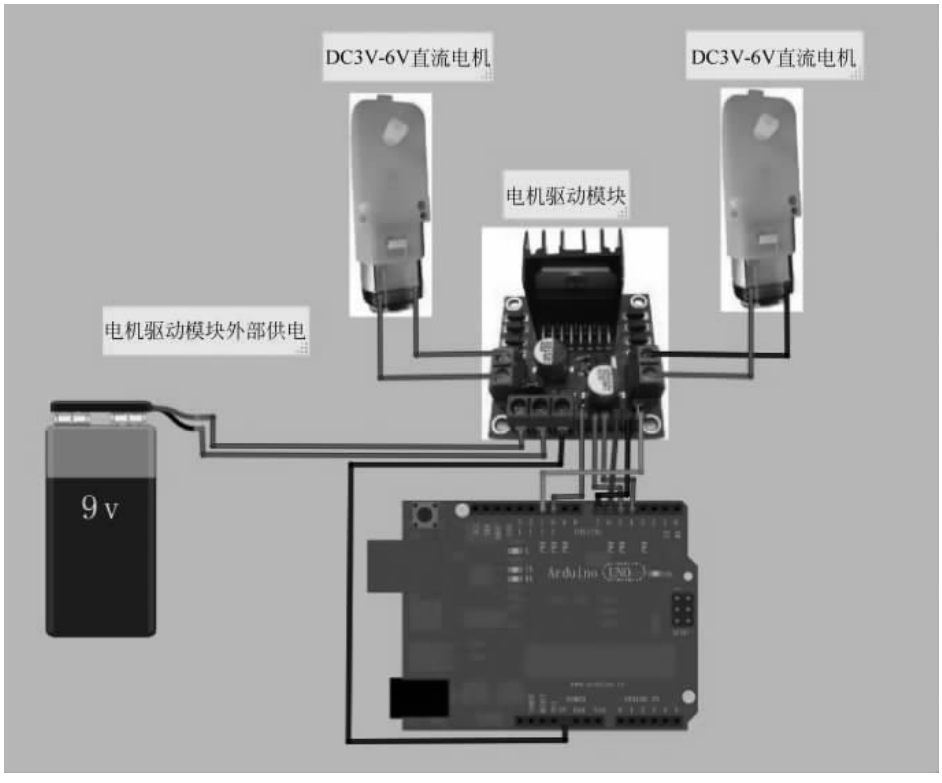

图 3-9　电机驱动模块电路连接图

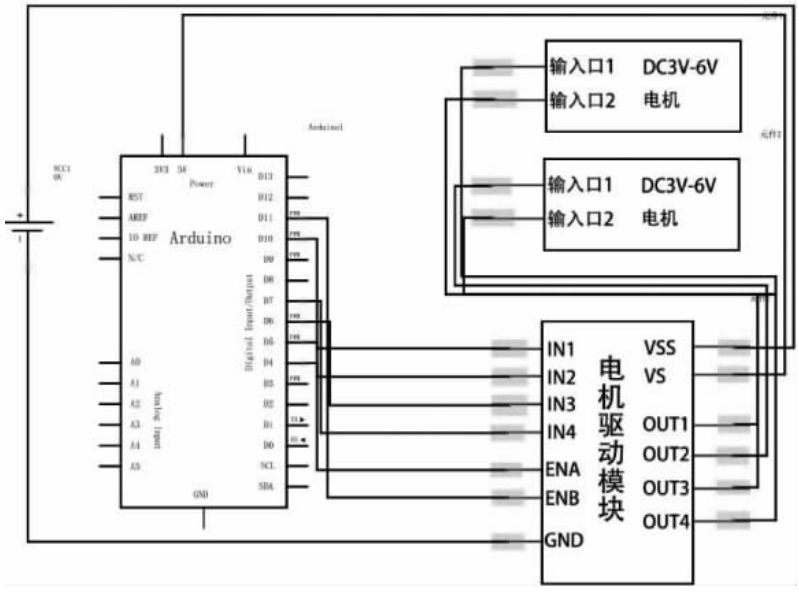

图 3-10　电机驱动模块电路原理图

相关代码：

单独控制电机转动方向的函数有三个，即电机前转、后转、制动。DV3V-6V 的电机两个接口端一个接口接高电平，另一个接口接低电平，电机就会向一个方向转动。通过分别控制两个电机的转动方向来配合避障模块进行小车移动。

```
int ENA = 10;                          //左电机使能
int ENB = 11;                          //右电机使能
int engine_a1 = 5;                     //左电机控制端口 1
int engine_a2 = 4;                     //左电机控制端口 2
int engine_b1 = 6;                     //右电机控制端口 1
int engine_b2 = 7;                     //右电机控制端口 2

pinMode(ENA ,OUTPUT);
pinMode(ENB ,OUTPUT);
pinMode(engine_a1 ,OUTPUT);
pinMode(engine_a2 ,OUTPUT);
pinMode(engine_b1 ,OUTPUT);
pinMode(engine_b2 ,OUTPUT);
void engine_left_forward()
                //通过控制电机控制端口来决定电机的转动方向，左电机向前转，下同
{
  digitalWrite(engine_a1 ,LOW);
  digitalWrite(engine_a2 ,HIGH);
}
void engine_left_backward()            //左电机向后转
{
  digitalWrite(engine_a1 ,HIGH);
  digitalWrite(engine_a2 ,LOW);
}
void engine_left_brake()               //左电机制动
{
  digitalWrite(engine_a1 ,LOW);
  digitalWrite(engine_a2 ,LOW);
}
void engine_right_forward()            //右电机向前转
{
  digitalWrite(engine_b1 ,HIGH);
  digitalWrite(engine_b2 ,LOW);
}
void engine_right_backward()           //右电机向后转
{
  digitalWrite(engine_b1 ,LOW);
  digitalWrite(engine_b2 ,HIGH);
```

```
}
void engine_right_brake()              //右电机制动
{
  digitalWrite(engine_b1 ,LOW);
  digitalWrite(engine_b2 ,LOW);
}
void forward()                         //小车前进
{
    analogWrite(ENA,184);
    analogWrite(ENB,184);
    engine_left_forward();
    engine_right_forward();
}
void turn_left()                       //小车左转
{
  analogWrite(ENA,184);
  analogWrite(ENB,184);
  engine_left_backward();
  engine_right_forward();
  delay(410);
}
void turn_right()                      //小车右转
{
  analogWrite(ENA,184);
  analogWrite(ENB,184);
  engine_left_forward();
  engine_right_backward();
  delay(410);
}
void brake()                           //小车制动
{
  analogWrite(ENA,184);
  analogWrite(ENB,184);
  engine_left_brake();
  engine_right_brake();
}
```

4）超声波测距(垃圾高度报警)模块

功能介绍：将垃圾高度报警模块置于垃圾桶侧壁，当垃圾达到一定高度时，触发超声波测距的信号，给出报警提示音或提示灯。运行过程中，进行 2s 的定时中断，检测垃圾平面是否到达超声波模块限定的高度，如果达到，则蜂鸣器有提示音。该模块的引脚说明如表 3-9 所示。其中，测试距离＝(ECHO 引脚高电平持续时间 * 声速)/2。

表 3-9　超声波测距模块引脚说明

引脚	用　途
VCC	接电源正极
GND	接地
TRIG	连接 Arduino 一个引脚，以一个 10μs 脉冲为信号，接到此信号后发送超声波进行探测，此时 ECHO 引脚为高电平
ECHO	连接 Arduino 一个引脚，有超声波信号返回后 ECHO 变为低电平

元器件清单：该模块所使用的元器件及其数量如表 3-10 所示。

表 3-10　超声波测距模块所需元器件

元器件名称	数　量
超声波测距模块 HC-SR04	1
杜邦线	若干
蜂鸣器	1
Arduino 开发板	1

电路图：超声波测距模块接线图如图 3-11 所示，超声波测距模块电路原理图如图 3-12 所示。

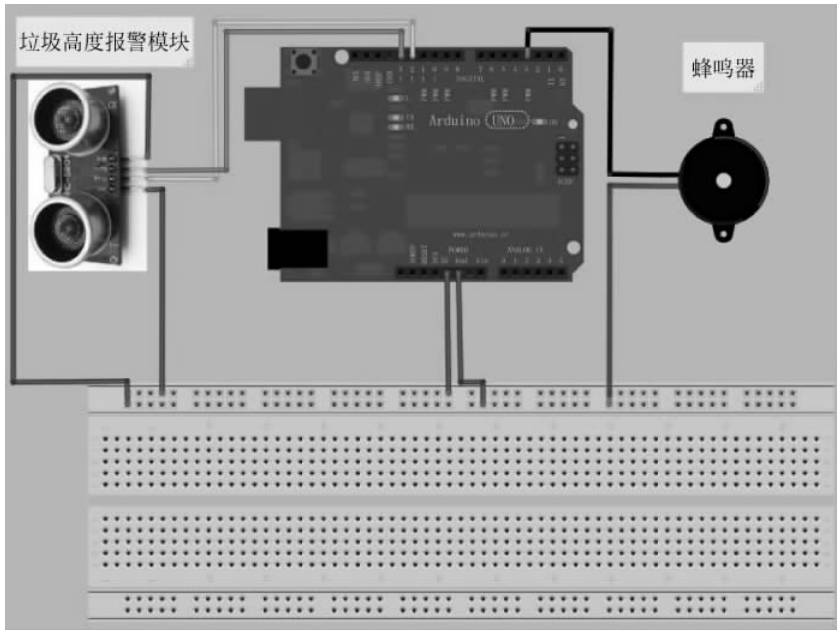

图 3-11　超声波测距模块接线图

相关代码：每个循环进行一次垃圾高度的报警检测，采用超声波测距模块完成此功能。超声波测距模块有 4 个引脚，即 VCC、GND、TRIG、ECHO。VCC 接 5V 供电；GND 接地；

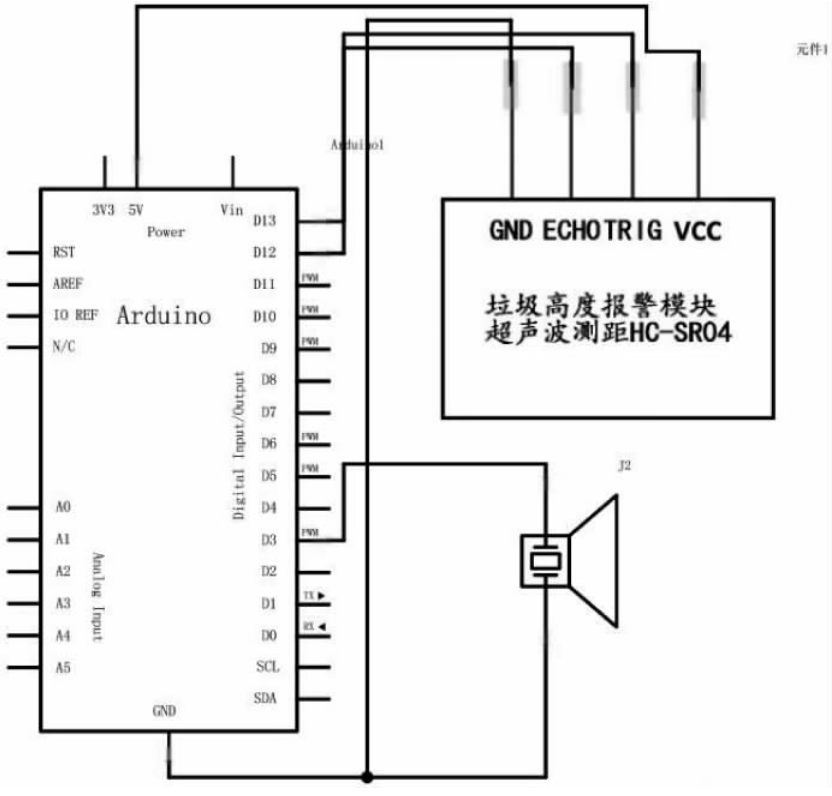

图 3-12　超声波测距模块电路原理图

TRIG 为输入引脚，接收 10μs 以上的脉冲后发送超声波；ECHO 为输出引脚，从 TRIG 发出超声波开始输出高电平，接收到返回的超声波为止，高电平时间即为超声波往返于模块与障碍物的时间，根据时间与声速计算障碍物的距离，首先判断障碍物是桶壁还是垃圾，桶壁直径约 15cm，如果距离小于 15cm，则判断障碍物是垃圾，蜂鸣器发出 100ms 的 500Hz 声音。

```
int sonic_trig = 12;                    //超声波测距触发信号
int sonic_echo = 13;                    //超声波测距返回信号接收
int buzzer = 3;                         //报警提示蜂鸣器

pinMode(sonic_trig ,OUTPUT);
pinMode(sonic_echo ,INPUT);
pinMode(buzzer ,OUTPUT);

void check_sonic()
{
    digitalWrite(sonic_trig ,LOW);
    delayMicroseconds(2);
```

```
    digitalWrite(sonic_trig ,HIGH);
    delayMicroseconds(10);
    digitalWrite(sonic_trig,LOW);                    //发一个 10us 的高脉冲去触发 TrigPin
    float distanceL = pulseIn(sonic_echo ,HIGH);//接收高电平时间
    distanceL = distanceL/58.0;                      //计算距离,公式由模块自定
    if (distanceL < 1.0)                             //测垃圾桶报警高度内壁直径约为 15cm
    {
      int i;
      for(i = 0;i < 50;i++)                          //输出频率为 500Hz 的报警音
      {
        digitalWrite(buzzer,HIGH);
        delay(1);
        digitalWrite(buzzer,LOW);
        delay(1);
      }
    }
}
```

5）人体感应模块

功能介绍：小车运动过程中，进行外部上升沿中断，进行人体感应模块的检测。如果人体感应模块感应到周围有人，则调用 brake()函数刹车，蜂鸣器报警 1s。该模块在周围没有人时持续输出低电平，有人时则输出高电平。注意：本模块在人进入时产生高电平，在人离开后自动变为低电平；如果两次检测之间，人在范围内但没有运动，则无法给出检测结果。该模块引脚说明如表 3-11 所示。

表 3-11　人体感应模块引脚说明

引　脚	用　途
VCC	电源正极，供电
GND	接地
OUT	接 Arduino 引脚，检测到周围有人时输出高电平信号

通过跳帽选择可重复触发模式和不可重复触发模式，螺母调节感应范围和锁定时间，理论值分别为 3～7m、0.5～300s。

元器件清单：人体感应模块所需元器件及其数量如表 3-12 所示。

表 3-12　人体感应模块元器件清单

元器件名称	数　量
人体感应模块 HC-SR501	1
杜邦线	若干
Arduino 开发板	1

电路图：人体感应模块接线图如图 3-13 所示，人体感应模块电路原理图如图 3-14 所示。

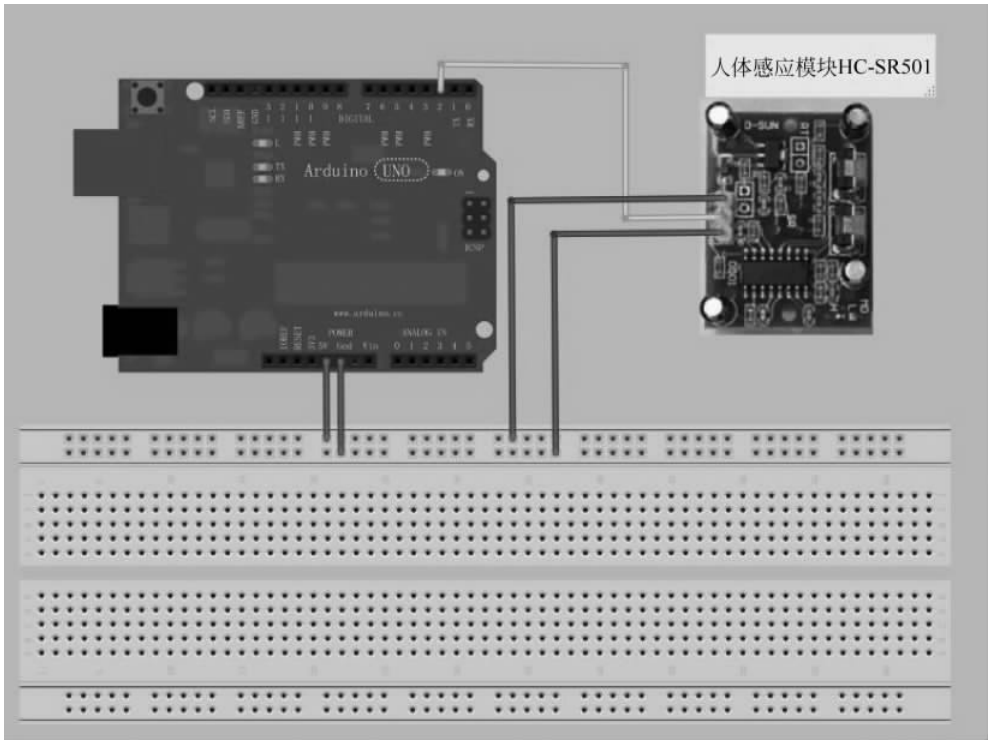

图 3-13　人体感应模块接线图

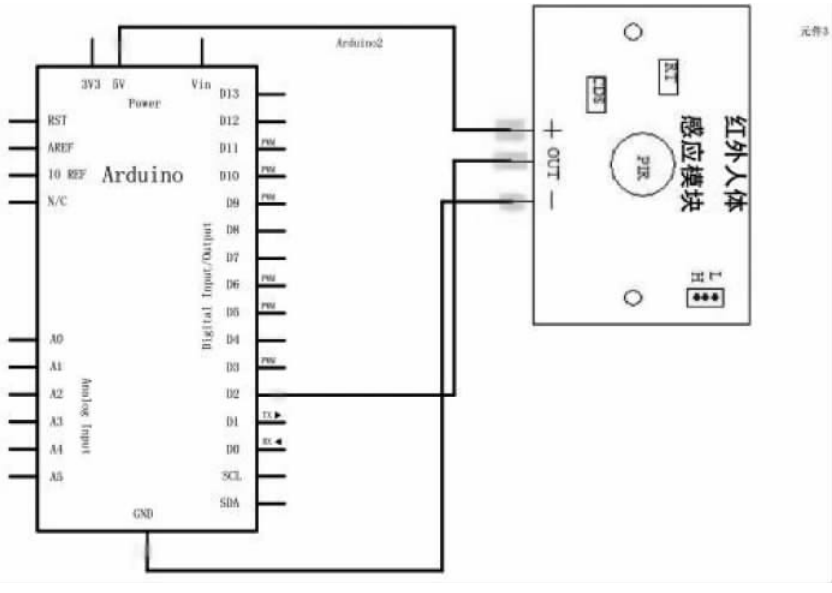

图 3-14　人体感应模块电路原理图

相关代码：

```
//外部中断：收到中断信号后小车刹车，蜂鸣器响 1s，将 flag 置 2
volatile int state = LOW;          //外部中断变量，人体红外感应器无人时为 0，有人时为 1
int infrared_sensor = 2;           //人体红外感应器
attachInterrupt(0 ,interrupt ,RISING);
pinMode(infrared_sensor ,INPUT);
void interrupt()
{
  brake();
  digitalWrite(buzzer ,HIGH);
  delay(1000);
  digitalWrite(buzzer ,LOW);
  flag = 2;
}
```

6）车体

功能介绍：各个模块的载体，搭载各个模块完成自动避障等功能。

元器件清单：车体组装所需元器件及其数量如表 3-13 所示。

表 3-13　车体组装所需元器件

元器件名称	数　量
车架	1
轮子	2
螺钉	若干
螺母	若干

3.1.4　产品展示

电机驱动模块实物图如图 3-15 所示。电机驱动模块用于驱动电机转动，配合避障模块完成自动避障功能。模块共有 6 个信号输入端对信号输入进行控制，3 个供电输入端对模块自身进行供电，4 个供电输出端驱动电机进行正向或反向旋转。

语音识别模块实物图如图 3-16 所示。语音识别模块通过识别输入的语音，返回指定信号代码给 Arduino 开发板，开发板根据返回值执行相应程序功能。本程序中完成“发动”与“刹车”两句话的识别。

人体感应模块实物图如图 3-17 所示。人体感应模块通过感应周围是否有人进入来调节输出端的电平。当周围有人时，输出端输出高电平，驱动中断，蜂鸣器报警；当周围没有人时持续输出低电平。

超声波测距模块实物图如图 3-18 所示。模块通过每隔 2s 发出超声波脉冲来检查垃圾是否达到最大限度。当达到垃圾桶设置的最大限度时，对发射出的超声波返回接收机的时间进行测算，如果在测定范围内，则蜂鸣器报警。本项目测量了安装超声波测距模块的垃圾

桶切面直径，调整精确了超声波测距距离范围，避免功能错误。

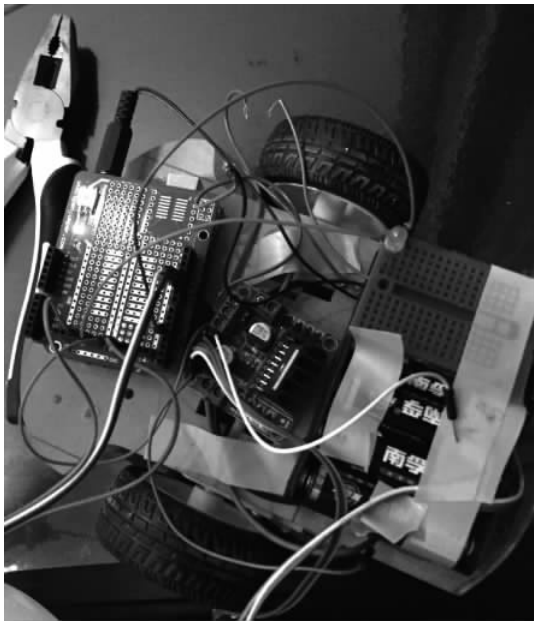

图 3-15　电机驱动模块实物图

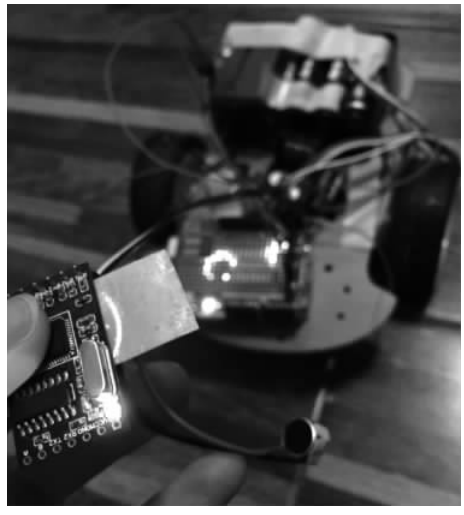

图 3-16　语音识别模块实物图

图 3-17　人体感应模块实物图

图 3-18　超声波测距模块实物图

红外避障模块实物图如图 3-19 所示。车体只安装了两个红外避障探头，分别位于车头的前端和左端。探测时采取前端优先于左端，左端优先于右端的逻辑。有障碍物时，输出低电平；无障碍物时，输出高电平。两次左转等于一次后转。避障测量距离设置恰到好处，靠近障碍物而不完全到达障碍物。最终自动无控制找到自己的主人。

最终成果展示图如图 3-20 所示。本项目实现了发出“发动”口令，小车启动，并自动避障，到人所在处附近时，发出“刹车”口令，小车停止；同时，实现了垃圾的测满功能，当垃圾到达一定高度时，蜂鸣器发出警报提示。

(a) 仅前方有障碍物时，左转

(b) 前方与左侧都有障碍物时，右转

图 3-19　红外避障模块实物图

图 3-20　最终成果展示图

3.1.5　故障及问题分析

(1) 问题：语音识别模块 LP-ICR 与 Arduino 开发板进行通信产生不同步错误。

LP-ICR 的 TX 接 Arduino 的 RX，通过判断识别语音返回的代码来决定执行响应的程序。在程序编译通过后，下载的过程中总是显示错误：not in sync：error：0x30。

解决方案：通过查找资料知道原因，由于使用 0 号端口与 LP-ICR 模块进行通信，在上传的过程中，其他模块对 Arduino 开发板有干扰，导致不能同步。在程序上传过程前，先将所有与 Arduino 开发板有通信的模块移除，上传结束后断电，再将相应模块重新接入，重新接通电源，语音识别模块与 Arduino 开发板间通信便可正常进行。

(2) 问题：电机驱动模块两个电机转速不同。

解决方案：尝试两种解决办法，第一种通过软件的方式，采用说明中提供的 PWM 调速

的方式，通过设置不同的通电时间比，将两轮的速度调整到同一程度；第二种方法是硬件的方式，更换电机。第一种解决办法尝试过程中发现，在本项目中加入到 Arduino 的逻辑框架中难度较大，最终选用第二种方法解决了问题。

(3) 问题：初步设计时达到的效果为，人发声“发动”，小车开启自动避障；人发声“刹车”，小车制动停止前进。遇到的问题为，小车未发动之前，没有输入，不进入 switch 判断句，小车发动后，在本次 loop 未完成之前，无法进行实时检测语音输入。由于需要对语音输入的语句判断后，才能决定调用什么功能，而外部中断只有简单的低电平、上升沿、下降沿和电平变化四种判断方式，无法对串行输入数据进行解析，所以也已无法使用外部中断。

解决方案：使用定时中断，每 2s 读取一次串行输入的数据判断是否有语句输入，由于 Arduino 开发板串行通信有缓存，所以可以采取这种方式来达到类似于实时的效果。

(4) 问题：避障中断返回，逻辑出现歧义。避障模块，根据车体前方红外传感器的低电平进行外部中断，在设计中出现了未发送“发动”命令，但车体前方感应到障碍物进行避障的情形。

解决方案：先尝试将 loop()函数中的接收串口输入的变量定为全局变量，在接收到中断信号后，先进行该变量与语音输入“发动”返回值是否相等的判断，相等的情况下才进行避障。如此解决了逻辑上存在的歧义。

在避障模块改变方法后，设计也随之做出了调整。增加了一个 flag 变量值，设某一次循环中语音检测到“发动”信号，则将这个循环定义为第一个循环；在某一个循环中检测到“刹车”信号，定义该循环为最后一个循环。flag 值与循环之间的关系如表 3-14 所示。

表 3-14　flag 值功能实现

flag 值	含　义
2	默认值，对应“发动”、“刹车”意外的干扰或者无输入。小车不避障
0	循环中检测到“发动”，进入第一个循环。小车进行正常避障
1	第一个与最后一个循环之间的其他循环。小车进行正常避障

由于在主函数的 loop 循环中，没有语音输入时检测不到信号，设置 flag 值就将没有语音输入时是否应该进行避障这个问题成功解决。

(5) 问题：使用中断能够达到近乎实时的转弯效果，但实际使用中，出现前方有障碍物小车在转弯的过程中，还没转到相应角度就结束中断，返回向前行驶的状态。

原因如下：小车前方的红外避障传感器探测距离有限且受障碍物材料影响波动比较大，而中断原理在于中断信号结束后立刻返回中断前状态，而不是所调用函数调用结束后返回中断前状态，导致想要完成的小车左转直角弯的运转还没完成便结束。这是小车运行状态受中断原理以及红外避障传感器硬件设施的影响。

解决方案：虽然使用中断能达到近乎实时的效果，但弊端却是不可接受的。对于避障转弯，采取放弃中断的方式，使用程序原本的 loop 函数，每个循环进行一次判定的方式。虽然这样，几乎所有判断都在 loop 函数中显得比较冗杂，每次所转角度也是固定的，不能自动

灵活控制，但比较两种方法，所达到的效果相对来说是可以接受的。

原始避障模式逻辑如下：

在主函数的每个循环检测一次车前方的红外传感器，如果检测到有障碍物，则避障开始。检测车体左方的传感器信号，若无障碍，则左转；若有障碍，则检测车体右方的传感器信号，若无障碍，则前行，若有障碍，则后转。

新避障判断模式：新的避障模式从 3 个红外传感器降为 2 个红外传感器，去掉了车体右侧的传感器。在主函数的每个循环检测一次车前方的红外传感器，如果检测到有障碍物，则避障开始。检测车体左方的传感器信号，若无障碍，则左转；若有障碍，则右转。再次检测车前方的传感器，若无障碍，则本次避障结束，小车恢复前行；若有障碍，则小车直接右转，且右转后前行。

与原始逻辑区别在于，小车右转是在前方、左方都有障碍物的前提下进行的，右转后只需再判断前方障碍就可以决定是右转后直接前行还是继续右转（两次右转等效于后转）。

3.1.6 元器件清单

完成本项目所用的元器件清单及其数量如表 3-15 所示。

表 3-15 “懒人”垃圾桶设计元器件清单

元器件名称	数 量
语音识别模块 LP-ICR	1
测试 MIC	1
RS232-TTL 模块	1
VGA-USB 线	1
上位机语音识别烧写软件	1
红外传感器模块	2
电机驱动模块	1
DC3V-6V 直流电机	2
电线	4
超声波测距模块 HC-SR04	1
蜂鸣器	1
人体感应模块 HC-SR501	1
杜邦线	若干
Arduino 开发板	1
扩展板	1
车架	1
轮子	2
螺钉螺母	若干

参考文献

[1] Arduino 中文社区. Arduino 定时器的使用[J/OL]. http://www.Arduino.cn/thread-2890-1-1.html
[2] Arduino 中文社区. 通过手机控制蓝牙小车[J/OL]. http://www.Arduino.cn/forum.php?mod=viewthread&tid=6590&highlight=%E8%93%9D%E7%89%99
[3] Arduino 中文社区. 用方便面盒子做的语音识别[J/OL]. http://www.Arduino.cn/thread-4546-2-1.html
[4] Arduino 中文社区. Arduino 教程——外部中断的使用[J/OL]. http://www.Arduino.cn/thread-2421-1-1.html
[5] 李永华,高英,陈青云. Arduino 软硬件协同设计实战指南[M]. 北京：清华大学出版社,2015.

3.2 项目 10：星伞

设计者：田筱,刘羽蝉,董冰

3.2.1 项目背景

雨天天色较暗,大街上车水马龙。若是极为阴沉的天气,那么一位行人在雨中撑起一把黑色的大雨伞,很容易与昏暗的背景融为一体,在路上行走的时候,非常不利于司机师傅及时准确地辨认,具有极大的交通隐患,甚至导致交通事故的发生。

基于对此场景的分析,本项目研究并设计制作星伞,简单来说,就是会发光的雨伞。用发蓝色光的 LED 在伞面构成 8×8 的点阵,布满整个伞面,使得在雨天使用它的行人能够更准确、更清晰地显示自己的位置,及时引起司机师傅的注意,确保雨天该行人的交通安全。同时,用户可以自己修改相应的模式,根据自己的创意想法及需求,设计属于自己的变幻图案。

目前市场上的雨伞缺乏创意,千篇一律,最多也只是在布面图案上做一些改变,星伞打破了雨伞外观设计沉闷的现状,为雨伞的设计注入了灵感与活力。它兼具实用性和美观性,可作为雨具携带,也可放在屋内或门前起到一定的装饰作用；它在相隔一段距离的地方也可以显示文字或图案,甚至简单动画,起到一定的提示作用,星伞适合所有浪漫的场合,如告白、求婚、联欢等。

3.2.2 创意描述

因为现在 LED 显示屏随处可见,8×8 LED 点阵看起来很普通,但是,将 8×8 LED 点阵固定到伞面上,这就是本项目的创新点。

每一个小的产品都有它自己的特色,在不同的环境下它的作用也不同。在傍晚的广告牌上也许一眼望去它并不能吸引你,但当你在雨中发现随行人移动闪烁显示图案的雨伞,你一定会心动。

虽然本项目只完成了基本创意的功能,但是,拓展性非常强,后续结合更多其他功能上的创意,例如,可以为星伞做红外线感应开关、对雨水的重力感应触发点阵产生随机点亮的

变幻效果，但由于纯手工制作耗时太长，有兴趣的读者可以自己加入相关功能。

3.2.3 功能及总体设计

本项目的目的是制作一把可以发光的雨伞，这样，在昏暗的环境下可以确保使用者更加安全。在达到这样目的的同时，还要让雨伞看起来美观，因此，主要设计点在于 LED 灯的形状以及闪烁的频率。

1. 功能介绍

雨伞面上 4 块对称的 8×8 LED 点阵实现多种图案，图案的代码可在 Arduino 源程序中任意改变，显示时间也可自由改变。一次可显示 9 种不同的图案，既可以是静态的图案，也可以是动态变化的图案。

2. 总体设计

要实现星伞的主要功能，最主要的部分就在于 LED 点阵的设计，除了实现雨伞的功能之外，LED 点阵还使得雨伞的外观更加漂亮，因此必须对 LED 点阵的闪烁频率、显示的图案进行精心的设计。

1）整体框架图

项目的整体框架图如图 3-21 所示。

2）系统流程图

系统流程图如图 3-22 所示。

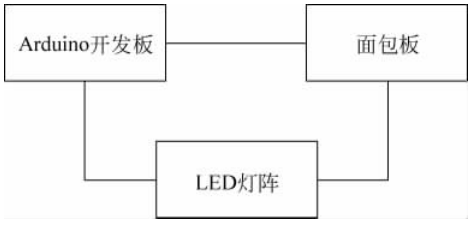

图 3-21 整体框架图

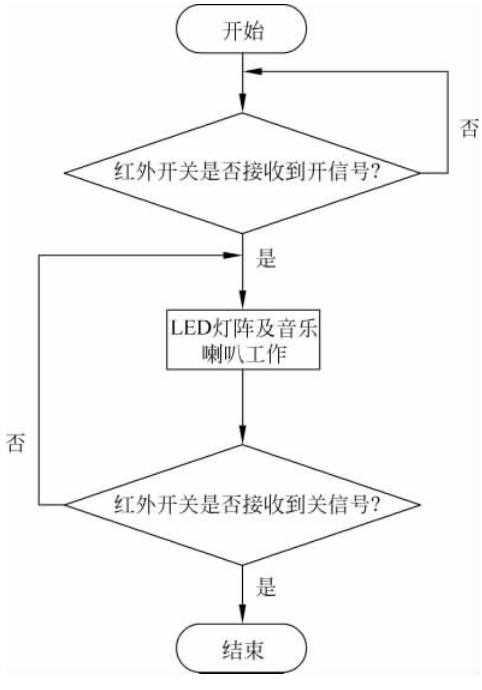

图 3-22 系统流程图

开始时，检测红外开关是否接收到启动信号，若没有，则一直等待启动信号，否则启动LED灯阵及音乐喇叭开始工作，直到接收到红外开关传来的关闭信号才结束一切工作。

3）总电路图

系统总电路图如图3-23所示。

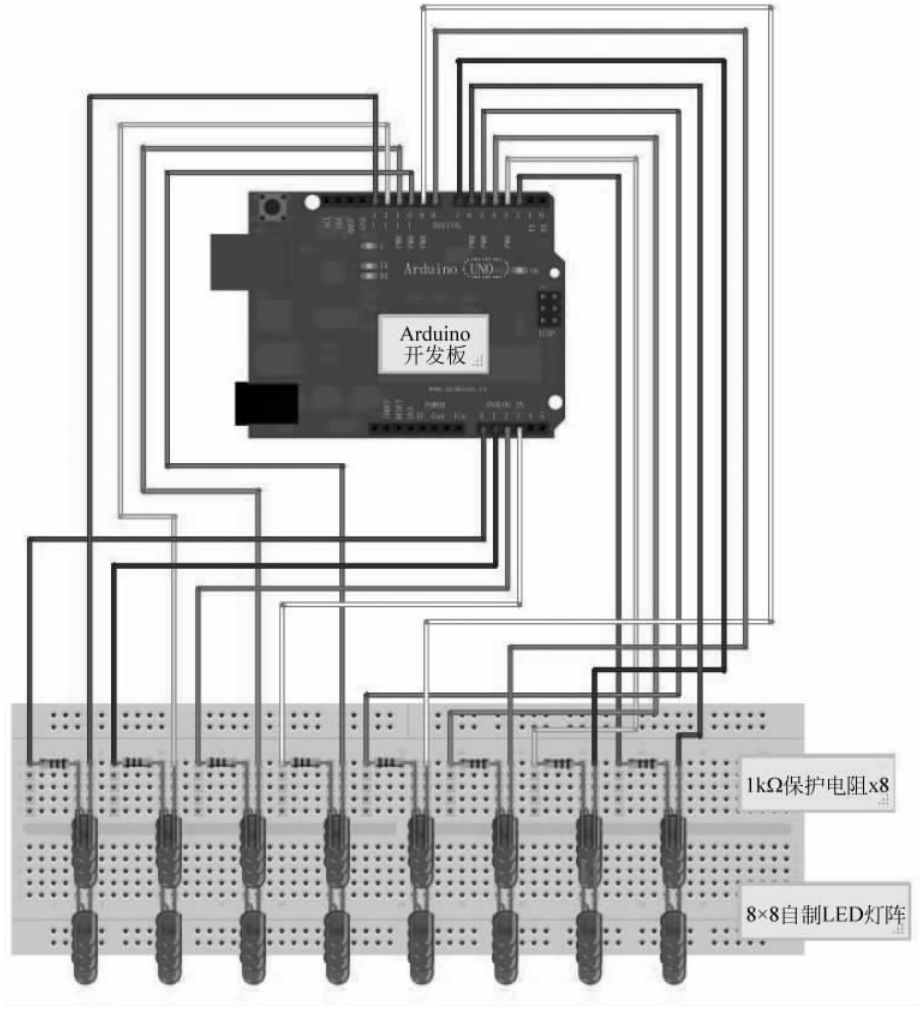

图3-23　总电路图

电路分为相同的四块，每一块均为8×8 LED点阵，图3-23描述的是其中一块电路图。每8个LED灯阳极相接形成一串，外加电阻接到Arduino相应端口，8串LED灯对应阴极分别相接引出8条接线分别接到相应端口，共用到16个Arduino端口。利用面包板，四块LED灯阵所对应的接线并联，接入相同的Arduino接口，使得四面点阵同时输出相同的图案。

3. 模块介绍

本项目通过 LED 一个模块即可实现其功能。下面介绍 LED 模块。

功能介绍：构造一个 8×8 点阵，实现图形的任意变化。

元器件清单：Arduino 开发板、Mini 面包板、256 个 LED、软导线、热缩管、若干导线。

电路图：该模块的电路图如图 3-23 所示。

相关代码：

```
#define display_array_size 8                                          //八位字节
#define data_null 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00             //全灭
#define data_ascii_A 0x18,0x18,0x18,0xff,0xff,0x18,0x18,0x18
//显示"十字交叉" 具体说明如下：标有"1"的位置在点阵上显示灯亮,"0"的位置显示灯灭
data_ascii_A
{
      {0, 0, 0, 1, 1, 0, 0, 0},                                       //0x18
      {0, 0, 0, 1, 1, 0, 0, 0},                                       //0x18
      {0, 0, 0, 1, 1, 0, 0, 0},                                       //0x18
      {1, 1, 1, 1, 1, 1, 1, 1},                                       //0xff
      {1, 1, 1, 1, 1, 1, 1, 1},                                       //0xff
      {0, 0, 0, 1, 1, 0, 0, 0},                                       //0x18
      {0, 0, 0, 1, 1, 0, 0, 0},                                       //0x18
      {0, 0, 0, 1, 1, 0, 0, 0},                                       //0x18
  }


#define data_ascii_B 0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff  //全亮
#define data_ascii_C 0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff  //全亮
#define data_ascii_D 0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff  //全亮
#define data_ascii_E 0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff  //全亮
#define data_ascii_F 0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff  //全亮
#define data_ascii_G 0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff  //全亮
#define data_ascii_H 0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff  //全亮
#define data_ascii_I 0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff  //全亮

byte data_ascii[][display_array_size] =
{
    data_null,
    data_ascii_A,
    data_ascii_B,
    data_ascii_C,
    data_ascii_D,
    data_ascii_E,
    data_ascii_F,
    data_ascii_G,
    data_ascii_H,
    data_ascii_I,
```

```
};
//行引脚设置
const int row1 = 2;
const int row2 = 3;
const int row3 = 4;
const int row4 = 5;
const int row5 = 17;
const int row6 = 16;
const int row7 = 15;
const int row8 = 14;
//列引脚设置
const int col1 = 6;
const int col2 = 7;
const int col3 = 8;
const int col4 = 9;
const int col5 = 10;
const int col6 = 11;
const int col7 = 12;
const int col8 = 13;

//扫描实现 LED 灯的亮灭
void displayNum(byte rowNum,int colNum)
{
    int j;
    byte temp = rowNum;
    for(j = 2; j < 6; j++)              //将 row1 至 row4 引脚设置为低电平
    {
      digitalWrite(j, LOW);
    }
    digitalWrite(row5, LOW);            //将 row5 至 row8 引脚设置为低电平
    digitalWrite(row6, LOW);
    digitalWrite(row7, LOW);
    digitalWrite(row8, LOW);
    for(j = 6;j < 14;j++)               //将 col1 至 col8 引脚设置为高电平
    {
    digitalWrite(j, HIGH);
    }
    switch(colNum)                      //将选择的列引脚设置为低电平
    {
     case 1: digitalWrite(col1, LOW);break;
     case 2: digitalWrite(col2, LOW);break;
     case 3: digitalWrite(col3, LOW);break;
     case 4: digitalWrite(col4, LOW);break;
     case 5: digitalWrite(col5, LOW);break;
     case 6: digitalWrite(col6, LOW);break;
     case 7: digitalWrite(col7, LOW);break;
     case 8: digitalWrite(col8, LOW);break;
```

```
      default:break;
    }

    for(j = 1 ;j < 9; j++)
    {
      temp = (0x80)&(temp);            //第四位置 0
    if( temp == 0)
    {
      temp = rowNum << j;              //将 rowNum 左移 j 位
      continue;
      }

      switch(j)                        //将选择的列引脚设置为高电平
      {
        case 1: digitalWrite(row1, HIGH);break;
        case 2: digitalWrite(row2, HIGH);break;
        case 3: digitalWrite(row3, HIGH);break;
        case 4: digitalWrite(row4, HIGH);break;
        case 5: digitalWrite(row5, HIGH);break;
        case 6: digitalWrite(row6, HIGH);break;
        case 7: digitalWrite(row7, HIGH);break;
        case 8: digitalWrite(row8, HIGH);break;
        default:break;
      }
     temp = rowNum << j;               //将 rowNum 左移 j 位
    }
}

void setup()
{
    int i = 0;
    for( i = 2; i < 18; i++)           //分配输出引脚
    {
    pinMode(i, OUTPUT);
    }

    for( i = 2; i < 18; i++)           //所有输出引脚置低电平,灯全灭
    {
    digitalWrite(i, LOW);
    }
}

void loop()                            //实现全灭和 A 至 I 共 9 种图案的显示
{
    int t1;
    int l;
    int arrage;
```

```
    for( arrage = 0; arrage < 10; arrage++)
    {
        for( l = 0; l < 512; l++)      //设置每个图形显示时间
        {
            for(t1 = 0; t1 < 8; t1++) //循环扫描行列实现一个图案的显示
            {
                displayNum(data_ascii[arrage][t1],(t1 + 1));
            }
        }
    }
}
```

3.2.4　产品展示

整体实物图如图 3-24 所示。实物内部图如图 3-25 所示。最终演示效果图如图 3-26 所示。

图 3-24　整体实物图

在图 3-24 中，可以看见在透明的雨伞下面的黑色罩面上整齐布上了 8×8 灯阵，伞一共八面，每间隔一面布上灯阵，共使用了 4 个 8×8 灯阵从图 3-24 中只能看见其中两个灯阵。

在图 3-25 中，可以清楚看见 Arduino 的 16 个端口被整齐地接上，用了两块 Mini 电路板，一块通过电阻接共阳极，一块接共阴极，用到了多个自制排线，内部结构虽然东西多却不乱。

从图 3-26 可以看到最终的演示效果，伞面上四块 LED 灯阵共同显示相同图案，而图案的形状与种类多变，可根据个人喜好自行设计。图 3-26 只截取了其中一个显示图形，是一个爱心形状。

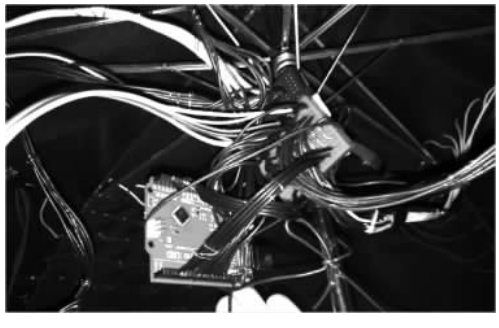

图 3-25　实物内部图

图 3-26　最终演示效果图

3.2.5 故障及问题分析

(1) 问题：选购何种 LED 灯可以满足要求？

解决方案：若要将 LED 灯连接，则需要它引脚相对较长；若要将 LED 灯固定在伞面，则不能采用圆头的 LED，研究过贴片 LED、不同规格大小的 LED 后，选择了 2mm×5mm×7mm 长方扁形长短引脚 LED，它更方便连接，而且解决了圆头 LED(平时普通物理实验中用到的普通种类)固定在伞面上会造成伞面凸起的情况。

(2) 问题：如何将 8×8 LED 点阵固定在伞面上？

解决方案：综合讨论后，决定买一把透明的、非常结实的雨伞，将完整的厚黑布裁剪成两块，每一块黑布的大小恰好可以覆盖半把伞(即 4 个伞面)，之后再将 8×8 LED 点阵分别固定在每个伞面上。

(3) 问题：选购何种导线可以满足要求？

解决方案：将所有 LED 的负引脚用剥线钳拧至与正引脚成 90°角，以 8 个 LED 为一组将它们的正引脚依次距离相等间隔固定在同一根长、硬导线上。以 8 条连在一起的硬导线为一组，将导线沿伞架引出到雨伞中心的伞骨部分(所有的共阳极为纵向连接)作为 0～7 引脚，所以这一部分共用到 32 根长导线(包括固定、做插线接头用到的长硬导线，总共用掉 70 多根)。

共阴极连接相对较为困难，在电子市场上对比，找到了软导线，由多股极细的线并股而成，实现了横向共阴极用软线连接，最终能够将伞收起。

(4) 问题：如何连接 8×8 LED 点阵？

解决方案：考虑到要将 8×8 LED 点阵固定在伞面上，软面背景下的连接并不能用焊接，于是将所有导线与导线之间的连接或二极管与导线之间的连接处套上长度适中的一截热缩管，并用吹风机对其进行适当加热，经过一段时间，热缩管会紧缩，从而起到软连接的作用。为保证横向共阴极连接的软度，不影响后期收伞，把负引脚长度减去一半，扩大软导线的连接面积。

3.2.6 元器件清单

完成本项目所需的元器件及其数量如表 3-16 所示。

表 3-16 星伞项目元器件清单

元器件名称	数 量
Arduino 开发板	1
LED 灯	256
电阻	8
面包板	2
导线	若干
热缩管	若干

参考文献

[1]　豆丁网. 完整光立方 LED 原理图[J/OL]. http://www.docin.com/p-438970001.html
[2]　豆丁网. 8×8 点阵引脚分布图[J/OL]. http://www.docin.com/p-247586836.html
[3]　李永华,高英,陈青云. Arduino 软硬件协同设计实战指南[M]. 北京：清华大学出版社,2015.

3.3　项目 11：强密码生成器

设计者：曹爽,田原,王建勇

3.3.1　项目背景

人们在上网时,无论是使用计算机,还是用手机登录,经常需要输入密码。然而,设置密码是一个难题,密码短虽然简单易记,但是容易被破解；密码长虽然相对安全,但是难以记忆。简单密码在生活中无处不在,破解工具五花八门；而设置强密码又有输入不便、验证方式不通用等问题。

2011 年 12 月 22 日,北京警方接到 CSDN 公司报案,称其公司服务器被入侵,核心数据遭到泄露。随后,大量用户数据被公布于网络上。自此之后,密码问题日益被人们关注。图 3-27 所示为关键词“密码”的百度搜索趋势。

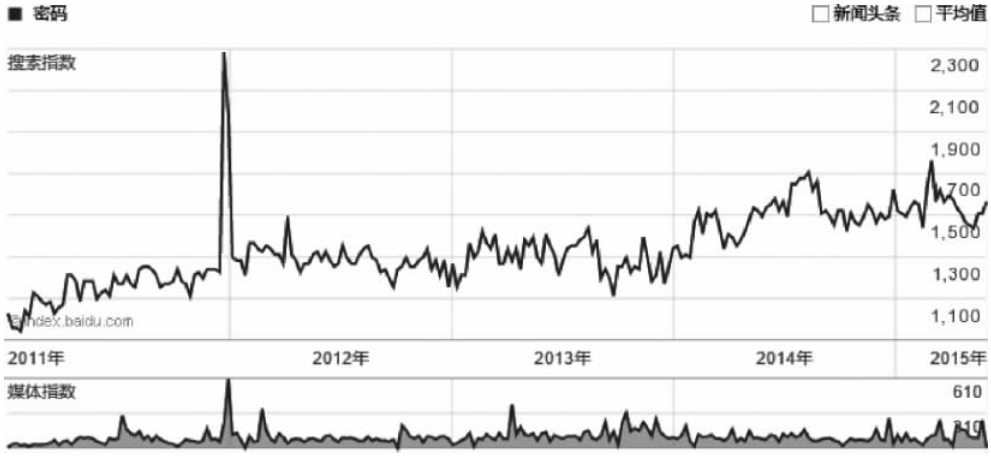

图 3-27　关键词“密码”的百度搜索趋势

从图 3-27 中可以看出,密码问题的关注度逐年升高。显然,强密码相比于普通密码更加安全,那么,为什么得不到广泛应用呢？答案很简单,输入麻烦,难以记忆。用户往往需要花费输入普通密码数倍的时间去输入强密码。而且,过长的密码也难以记忆,经常容易出现输入错误等问题。

由此,本项目利用 RFID 模块设计一款便捷、安全、通用的强密码输入设备,只需刷

RFID卡即可自动输入16位强密码，适用于任何能连接USB的设备。如果说二维码的发明解决了输入长网址的难题，那么，本项目的产品不仅完成了输入强密码的功能，还避免了用户记忆长密码之苦，同时也提高了密码安全性。

3.3.2 创意描述

在生活中，RFID卡主要应用于刷卡消费、身份识别、门禁系统等，每一张RFID卡有全球独一无二的序列号。根据这个特点，考虑利用该序列号通过算法生成一串16位的强密码，这样每张卡就对应唯一的强密码不会重复。此外，RFID卡可以扩展到类似的射频卡，如北京邮电大学的校园一卡通，这样会更加方便，贴近生活实际。

3.3.3 功能及总体设计

1. 功能介绍

本产品最终实现的功能为强密码生成器，即刷RFID卡时，输出卡片对应的唯一强密码，且不同的卡片输出的强密码不同。

2. 总体设计

要实现上述功能，首先需要进行设计的就是密码生成器，这是整个项目的核心。除此之外，还需要通过RFID读出卡中的内容。另外，为用户设计相应的提醒部分也是必要的。

1）整体框架图

项目整体框架图如图3-28所示。

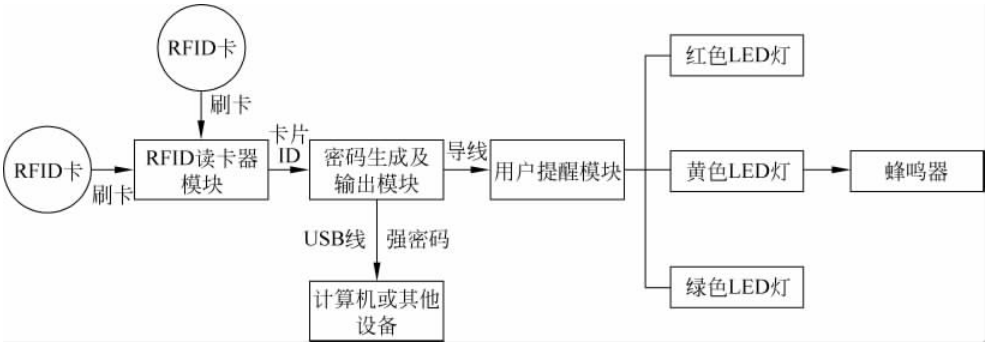

图3-28 整体框架图

整个强密码生成器分为三个模块。首先用RFID卡在读卡器模块处刷卡，之后读卡器模块读取卡片ID并传给密码生成及输出模块，由该模块通过USB线将生成的强密码传入计算机或其他设备。同时，密码生成及输出模块通过导线与用户提醒模块相连，用来显示强密码生成器的工作状态（连接异常、正常工作、输出密码），并通过LED灯和蜂鸣器提醒用户工作状态。

2）系统流程图

系统流程图如图3-29所示。

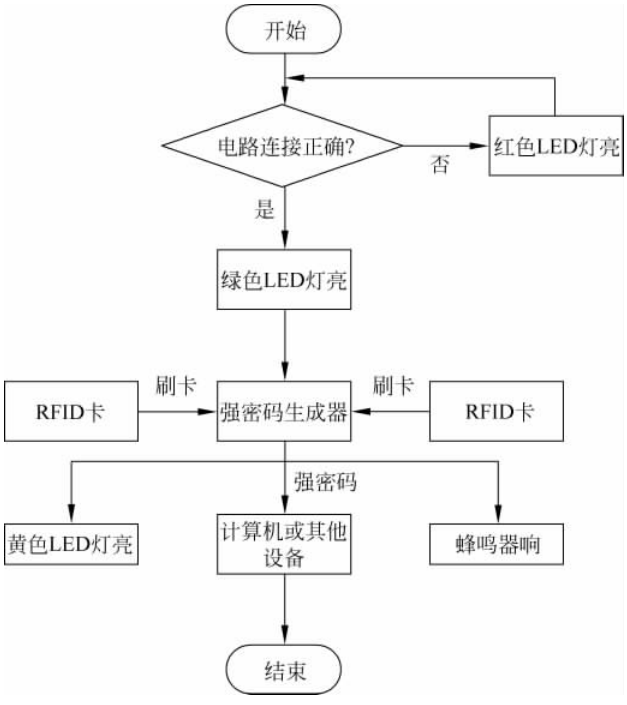

图 3-29　系统流程图

强密码生成器的工作流程为：先检查电路连接是否正确，待电路可以正常工作（绿色LED灯亮）后，刷卡生成强密码，并输入到计算机或其他设备中。电路工作状态通过 LED 灯和蜂鸣器来提醒用户。

3）总电路图

系统总电路图如图 3-30 所示。

引脚连接说明如下：

MFRC-522 读卡器与 Arduino Leonardo 开发板连接方法如表 3-17 所示。

表 3-17　MFRC-522 读卡器与 Arduino Leonardo 开发板连接方法

MFRC-522 读卡器引脚	Arduino Leonardo 开发板引脚
SDA	10
SCK	ICSP3
MOSI	ICSP4
MISO	ICSP1
RST	ICSP5
3.3V	3.3V
GND	GND

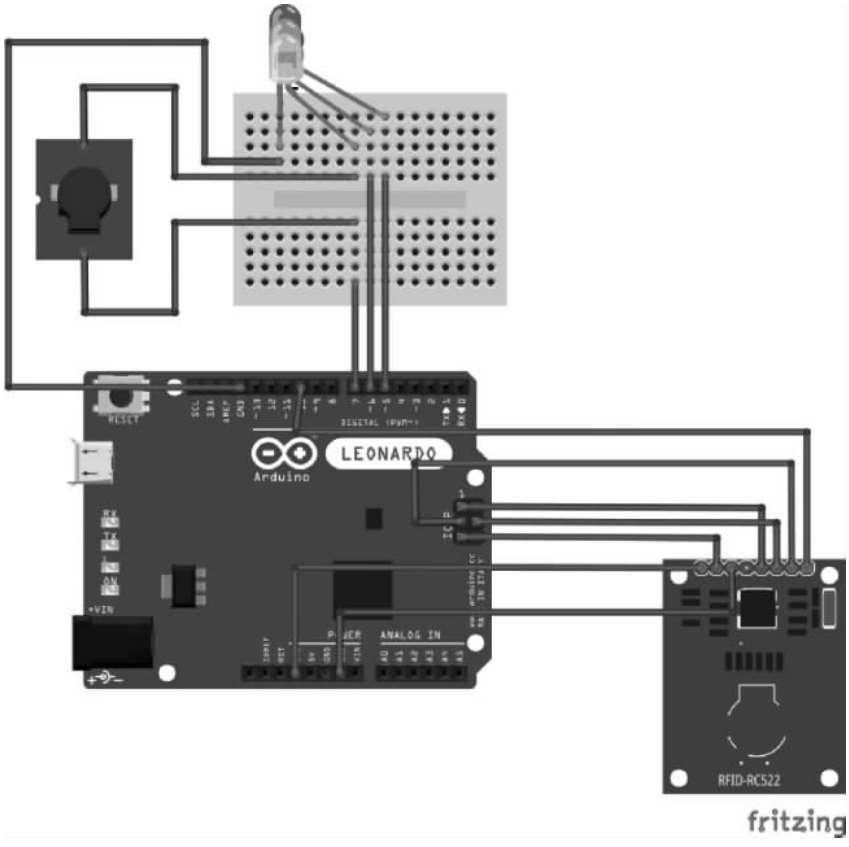

图 3-30　总电路图

小面包板上各元件与 Arduino Leonardo 开发板连接方法如表 3-18 所示。

表 3-18　小面包板上各元件与 Arduino Leonardo 开发板连接方法

小面包板上各元件	Arduino Leonardo 开发板引脚
红色 LED 正极	5
绿色 LED 正极	6
蜂鸣器正极	7
红色 LED 负极	GND
绿色 LED 负极	GND
黄色 LED 负极	GND

除此之外，黄色 LED 正极与蜂鸣器负极相连，即黄色 LED 与蜂鸣器串联，黄灯亮同时蜂鸣器响，黄灯灭同时蜂鸣器停。

3. 模块介绍

本项目主要分为三个模块：RFID 读卡器模块、密码生成及输出模块、用户提醒模块。

1）RFID 读卡器模块

功能介绍：读取 RFID 卡片信息，指示初始化是否正常，读取 RFID 读卡器的版本号，在连接异常时返回的版本号不正确。通过红、绿 LED 指示工作状态。

元器件清单：MFRC-522 读卡器。

电路图：RFID 模块电路图如图 3-30 所示。

相关代码：

```
byte v = mfrc522.PCD_ReadRegister(mfrc522.VersionReg);
if (v == 0x91 || v == 0x92)
    digitalWrite(GREEN, HIGH);
else
    digitalWrite(RED, HIGH);
…
//新卡片出现之后再继续执行
if ( ! mfrc522.PICC_IsNewCardPresent()) {
    return;
}
//读取到卡片序列号之后再执行
if ( ! mfrc522.PICC_ReadCardSerial()) {
    return;
}
```

2）密码生成及输出模块

功能介绍：针对传入的 RFID 卡片 ID，生成强密码。

元器件清单：Arduino Leonardo 开发板。

电路图：密码生成及输出模块的电路如图 3-30 所示。

相关代码：

```
void PrintPassword() {
    unsigned char * hash = MD5::make_hash((char *)mfrc522.uid.uidByte);
//使用 MD5 算法对卡片 UID 生成摘要
char *pwd = (char *) malloc (sizeof(char) * 17);   //存放密码的字符串
    static const char trans[95] =                  //转义数组
    "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890[];',./{}:\"<>?\\| =
-+_)(*&^%$#@!~";
    for (int i = 0; i != 16; ++i)
    pwd[i] = trans[*(hash + i) % 94];              //将 MD5 生成的摘要转换为强密码字符串
    pwd[16] = '\0';                                //防止极端条件下的异常
    Keyboard.print(pwd);                           //打印密码
    free(hash);                                    //避免内存泄漏
    free(pwd);
}
```

3）用户提醒模块

功能介绍：给用户直观地展示产品工作状态，LED 红灯亮为接触不良，绿灯亮为正常工作，黄色灯亮并蜂鸣为正在输出密码。

元器件清单：黄色 LED、红色 LED、绿色 LED、蜂鸣器、小面包板。

电路图：图 3-30 中面包板部分。

相关代码：

```
//指示初始化是否正常,读取 RFID 读卡器的版本号
//在连接异常时返回的版本号不正确,通过红、绿 LED 指示工作状态
byte v = mfrc522.PCD_ReadRegister(mfrc522.VersionReg);
if (v == 0x91 || v == 0x92)
    digitalWrite(GREEN, HIGH);
else
    digitalWrite(RED, HIGH);
…
digitalWrite(WHITE, HIGH);                    //黄灯亮指示正在生成密码
```

3.3.4 产品展示

整体实物图如图 3-31 所示，内部图如图 3-32 所示，插上 USB 线之后的效果如图 3-33 所示，电路正常工作时的效果如图 3-34 所示，刷卡时的效果如图 3-35 所示，演示结果正确如图 3-36 所示，方形卡刷卡时的效果如图 3-37 所示，演示结果错误如图 3-38 所示。

图 3-31 整体实物图

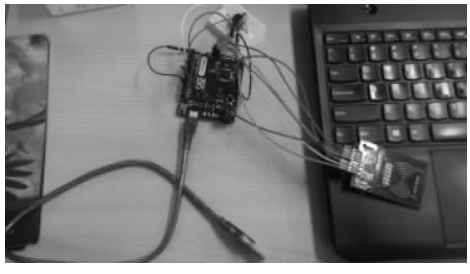

图 3-32 内部图

外部接口为 USB 线，封面上有三个 LED 灯显示工作状态，并有感应区负责感应 RFID 卡。

如图 3-33 所示，插上 USB 线之后，Arduino 开发板上的灯会亮。

如图 3-34 所示，电路正常工作时，面包板上的绿色 LED 灯亮。

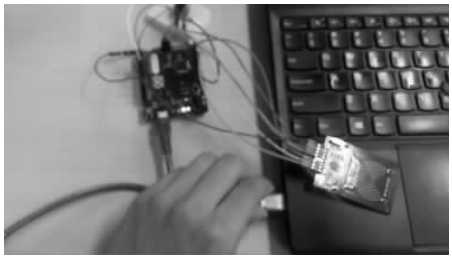

图 3-33 插上 USB 线之后的效果

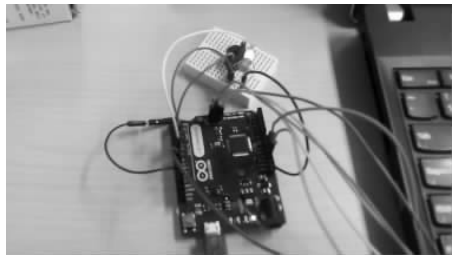

图 3-34 电路正常工作时的效果

如图 3-35 所示，演示时利用之前申请的邮箱，并设置密码为用圆形卡生成的密码。刷圆形卡时，屏幕上光标处会输入对应的强密码，同时面包板上的黄色 LED 灯亮（正常工作时绿色 LED 灯仍然亮）、蜂鸣器响。

如图 3-35 所示，单击“登录”按钮，可以登录到邮箱，说明密码正确。

如图 3-37 所示，使用另一张方形的卡，同样输入了一串强密码。

图 3-35　刷卡时的效果

图 3-36　演示结果正确

如图 3-38 所示，使用方形卡登录时显示“账号或密码错误”，说明方形卡对应的强密码与之前圆形卡对应的强密码不同，这就体现出每张卡密码的唯一性。

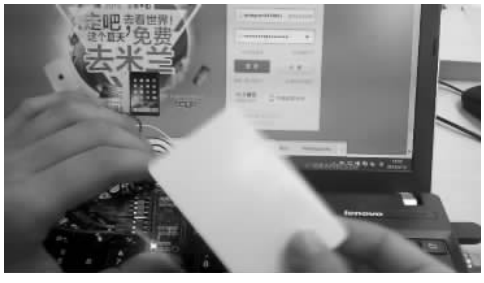

图 3-37　方形卡刷卡时的效果

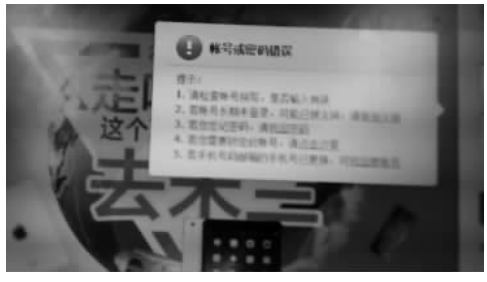

图 3-38　演示结果错误

3.3.5　故障及问题分析

（1）问题：如何获取标识一张 RFID 卡的唯一 ID?

解决方案：在 Github 上有专为 MFRC522 编写的 API，通过运行其提供的例子，学习 MFRC-522 类的使用方法，找到其存储 ID 的成员变量以及从卡中获取该 ID 的方法，也用到了判断 RFID 识别设备是否连接正常的函数。

（2）问题：如何将 ID 转换为密码?

解决方案：找到 MD5 的类库 API，可以根据所给的字符串指针通过相关哈希算法生成 16B32 位十六进制数。为了使其可以作为密码输入（char 类型字符串），设计一个根据该 16B 数据的指针，生成 16 个字符（包含键盘上所有特殊字符）串的算法。算法的原理设计正确，但是，第一次运行时却出现不同的卡是同一个密码的结果，查看代码调试，发现是把

MD5 返回的指针名误带双引号作为参数传递，去掉双引号后功能正常。

（3）问题：线路接触不良。

解决方案：焊接、换新导线。

3.3.6 元器件清单

完成本项目所需要的元器件及其数量如表 3-19 所示。

表 3-19 强密码生成器设计元器件清单

元器件名称	数 量
Arduino Leonardo 开发板	1
MFRC-522 读卡器	1
黄色 LED	1
红色 LED	1
绿色 LED	1
蜂鸣器	1
小面包板	1
包装盒	1
USB 线	1
导线	若干

参考文献

[1] 广州周立功单片机发展有限公司. MFRC_522 中文资料[J/OL]. http://wenku.baidu.com/view/4d5b2ceb680203d8cf2f241b.html

[2] 电子发烧友论坛. RFID-RC522 速成教程[J/OL]. http://bbs.elecfans.com/jishu_414027_1_1.html

[3] Wordpress 3.8.2 补丁分析 HMAC timing attack[J/OL]. http://drops.wooyun.org/papers/1404

[4] 维基百科. MD5[J/OL]. http://zh.wikipedia.org/zh-cn/MD5

[5] 李永华，高英，陈青云. Arduino 软硬件协同设计实战指南[M]. 北京：清华大学出版社，2015.

3.4 项目 12：智能教室

设计者：刘广泽，郭垚，刘玮康

3.4.1 项目背景

随着社会、经济水平的发展，人们对生活品质的要求也越来越高，要求生活环境舒适化、安全化、人性化、智能化。智能家居是通信技术、计算机技术和控制技术向传统家电产业渗透发展的必然结果。智能家居以住宅为平台，将物联网技术和传统家居有机地融合在一起，将与家居生活有关的设施集成，构建高效的住宅设施与家庭日程事务的管理系统，从而提升家居安全性、便利性、舒适性、艺术性。

本项目将物联网以及智能家居的理念应用于教室中，目的就是实现传统教室的信息化与智能化，即将教室内的情况统一进行信息采集，对于采集的信息，对其进行相应的分析与处理，并通过互联网传输，将用户所需要的信息呈现出来。

3.4.2　创意描述

本项目将传统的教室与物联网技术相结合，将各类传感器与互联网相连接，与智能处理相结合，将从传感器获得的信息进行分析、加工和处理，并将所需要的信息呈现给用户，以适应用户的不同需求，实现环境和状态信息的实时共享以及智能化的收集、传递、处理、执行。

3.4.3　功能及总体设计

基于上述创意，在总体上，该项目分为电路部分、传输部分和网络部分，三部分相结合以实现物联网。电路部分主要是收集信息，而传输部分是将收集到的信息传输到网络，网络部分进行信息处理和呈现。

1．功能介绍

本项目具体实现的功能包括教室内人数及剩余座位数的实时统计、教室内环境数据的监测，并通过网页和 LCD 液晶显示屏两种方式将以上统计结果呈现给用户。

2．总体设计

要实现上述功能，如前所述，主要将该项目分为电路部分、传输部分和网络部分。

1）整体框架图

该项目整体框架图如图 3-39 所示。

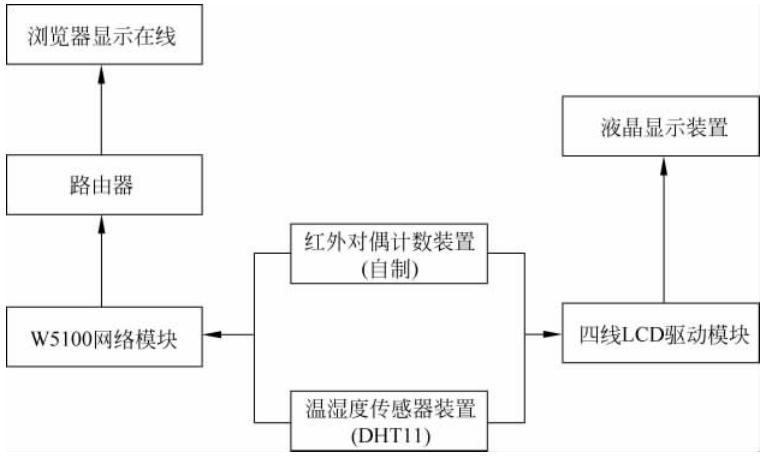

图 3-39　整体框架图

2）系统流程图

系统流程图如图 3-40 所示。

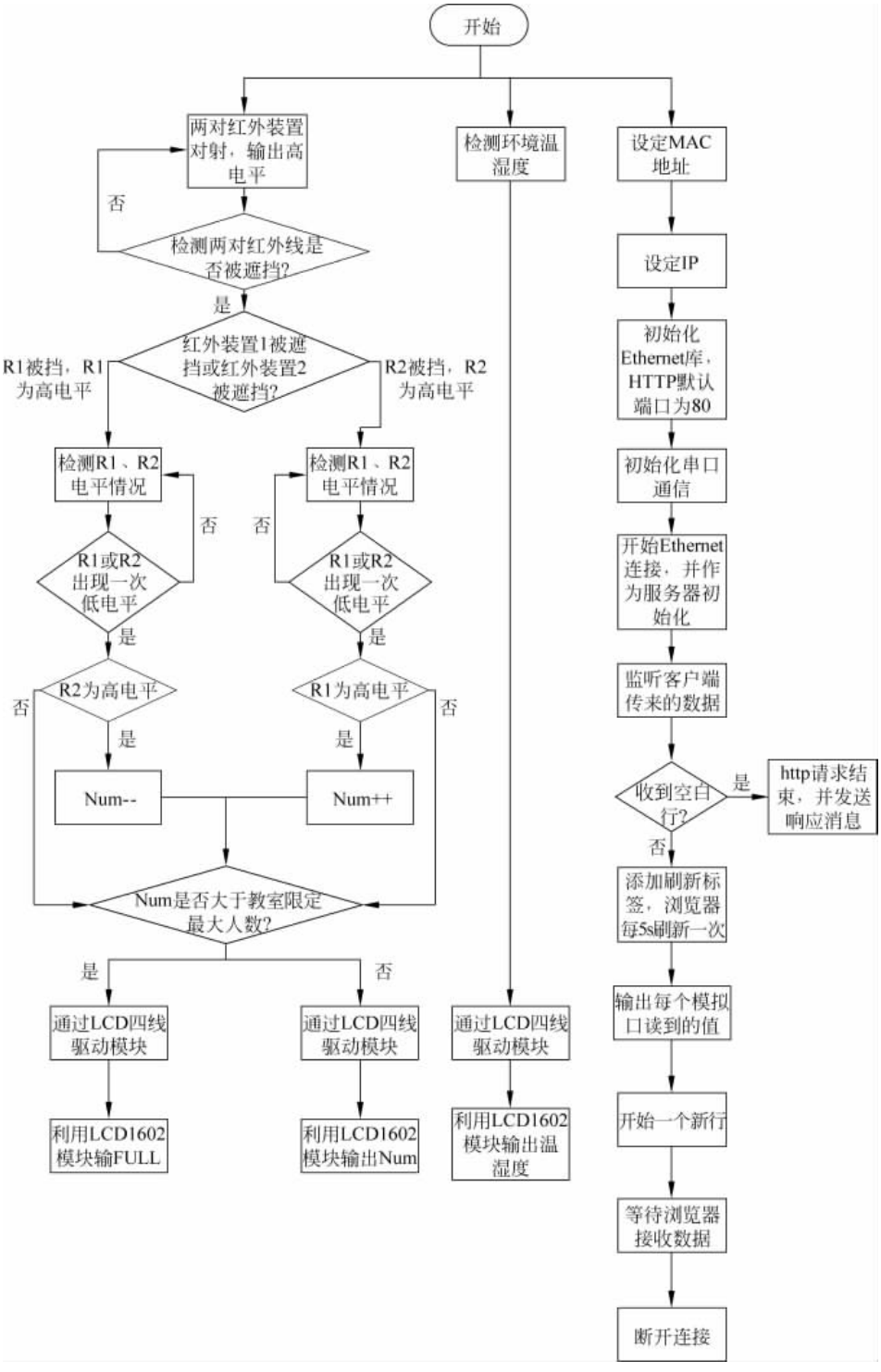
开始
两对红外装置对射，输出高电平
否
检测两对红外线是否被遮挡?
是
红外装置1被遮挡或红外装置2被遮挡?
R1被挡，R1为高电平
R2被挡，R2为高电平
检测R1、R2电平情况
否
R1或R2出现一次低电平
是
R2为高电平
否
是
Num--
检测R1、R2电平情况
否
R1或R2出现一次低电平
是
R1为高电平
否
是
Num++
Num是否大于教室限定最大人数?
是
否
通过LCD四线驱动模块
利用LCD1602模块输FULL
通过LCD四线驱动模块
利用LCD1602模块输出Num
检测环境温湿度
通过LCD四线驱动模块
利用LCD1602模块输出温湿度
设定MAC地址
设定IP
初始化Ethernet库，HTTP默认端口为80
初始化串口通信
开始Ethernet连接，并作为服务器初始化
监听客户端传来的数据
收到空白行?
是
http请求结束，并发送响应消息
否
添加刷新标签，浏览器每5s刷新一次
输出每个模拟口读到的值
开始一个新行
等待浏览器接收数据
断开连接

图 3-40　系统流程图

3）总电路图

系统总电路图如图 3-41 所示。

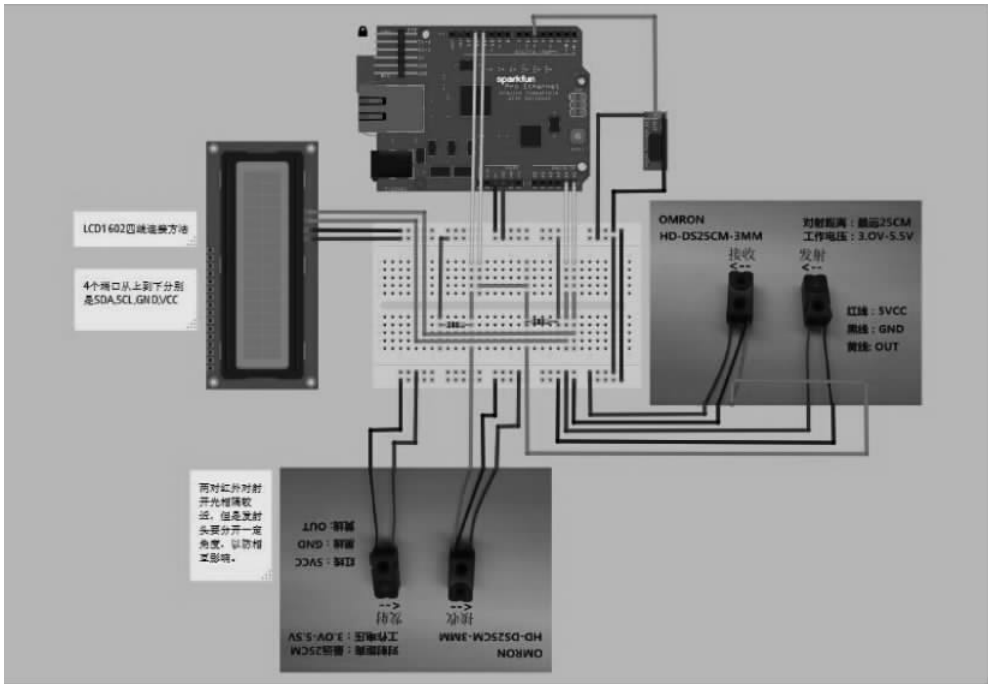

图 3-41　总电路图

3. 模块介绍

该项目主要分为三个模块：红外对射模块、温湿度传感器模块和 W5100 网络模块。

1）红外对射模块

功能介绍：红外对射全名叫“主动红外入侵探测器”(active infrared intrusion detectors)，其基本的构造包括发射端、接收端、光束强度指示灯、光学透镜等。其侦测原理是利用经 LED 红外光发射二极体发射的脉冲红外线，再经光学镜面做聚焦处理使光线传至很远距离，由受光器接收。当红外脉冲射束被遮断时，电压由高电平变为低电平。

元器件清单：该模块所需的元器件及其数量如表 3-20 所示。

表 3-20　红外对射模块元器件清单

元器件名称	数　量
Arduino 开发板	1
红外对射	2
1kΩ 电阻	2
LCD1602 显示屏	1
导线	若干

电路图：该模块电路连接图如图 3-42 所示。

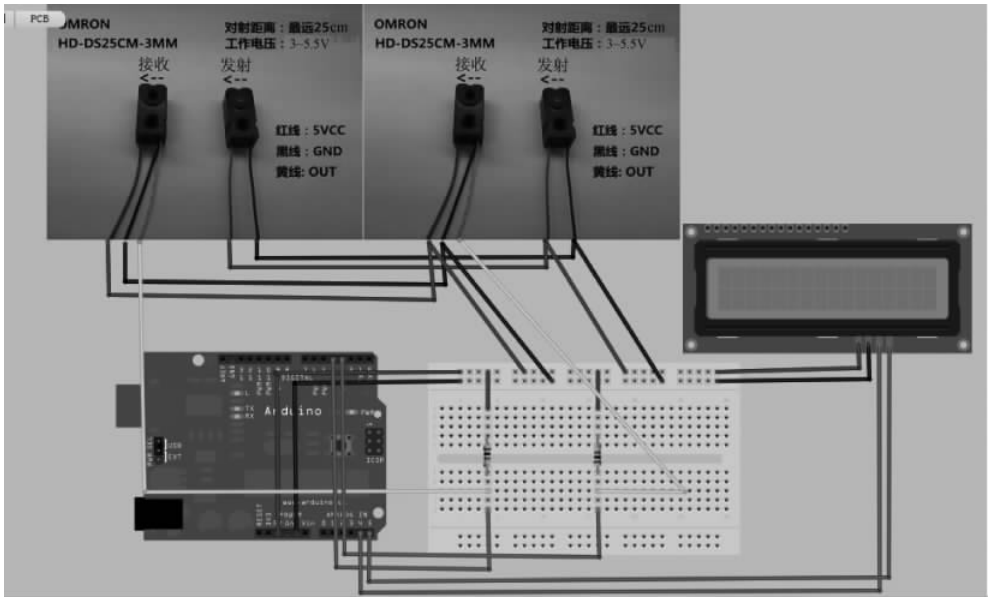

图 3-42　红外对射模块电路连接图

相关代码：

```
LiquidCrystal_I2C lcd(0x27,16,2);
#define N 120
int R1,R2;
int num = 0;
int sit;
void setup()
{
  lcd.init();
  lcd.backlight();
  num = 0;
  R1 = R2 = HIGH;
}
void loop()
{
  //当有人触碰,退出循环
  for(R1 = digitalRead(8), R2 = digitalRead(9);R1 == HIGH&&R2 == HIGH;)
  {
    R1 = digitalRead(8);
    R2 = digitalRead(9);
  }
  //人进入的情况
```

```
  if(R1 == 0&&R2 == 1)
  {
    sit = 0;
  }
  //人出去的情况
  else if(R2 == 0&&R1 == 1)
  {
    sit = 1;
  }
  switch(sit)
  {
    case 0:
      for(;R2 == HIGH;)
      {
        R2 = digitalRead(9);  //要是两个红外之间距离较近,使得人一触到 1,必然会触 10
      }
      //沿触发,当人有进出动作时,必有沿出现
      for(R1 = digitalRead(8), R2 = digitalRead(9);R1 == LOW&&R2 == LOW;)
      {
        R1 = digitalRead(8);
        R2 = digitalRead(9);
      }
      if(R1 == HIGH)
      {
        num++;
      }
        break;
    case 1:
      for(;R1 == HIGH;)
      {
        R1 = digitalRead(8);
      }
      for(R1 = digitalRead(8), R2 = digitalRead(9);R1 == LOW&&R2 == LOW;)
      {
        R1 = digitalRead(8);
        R2 = digitalRead(9);
      }
      if(R2 == HIGH)
      {
        num--;
      }
        break;
  }
}
```

2）温湿度传感器模块

功能介绍：温湿度传感器是指能将温度量和湿度量转换成容易被测量处理的电信号的设备或装置。

元器件清单：该模块所需的元器件及其数量如表 3-21 所示。

表 3-21　温湿度传感器模块元器件清单

元器件名称	数　量
Arduino 开发板	1
温湿度传感器	1
LCD1602 显示屏	1
导线	若干

电路图：该模块电路连接图如图 3-43 所示。

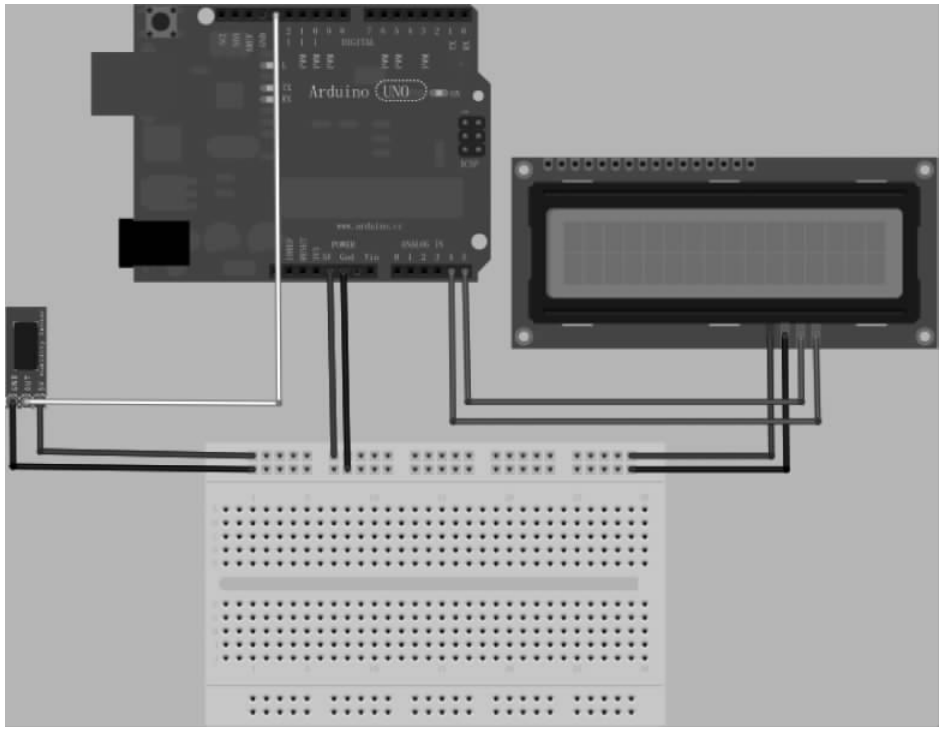

图 3-43　温湿度传感器模块电路连接图

相关代码：

```
#include <dht11.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2);
```

```
dht11 DHT11;
#define DHT11PIN 5
void setup()
{
  Serial.begin(9600);                                    //串口初始化
  delay(100);
}

void loop ()
{
  //获取传感器的数据
  DHT11.read(DHT11PIN);
  //传输到 LCD 屏
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("T:");
  lcd.print((int)DHT11.temperature);
  lcd.print("'C");
  lcd.setCursor(0,1);
  lcd.print("H:");
  lcd.print((int)DHT11.humidity);
  lcd.print("%");
}
```

3）W5100 网络模块

功能介绍：W5100 是一款多功能的单片网络接口芯片，内部集成有 10/100Mbps 以太网控制器，主要应用于高集成、高稳定、高性能和低成本的嵌入式系统中，使用 W5100 可以实现 Internet 连接。W5100 与 IEEE802.3 10BASE-T 和 802.3u 100BASE-TX 兼容，W5100 内部集成了全硬件的且经过多年市场验证的 TCP/IP 协议栈、以太网介质传输层（MAC）和物理层（PHY）。全硬件 TCP/IP 协议栈支持 TCP、UDP、IPv4、ICMP、ARP、IGMP 和 PPPoE，这些协议已经在很多领域经过了多年的验证。W5100 内部还集成有 16KB 存储器用于数据传输，使用 W5100 不需要考虑以太网的控制，只需要进行简单的端口编程。

元器件清单：该模块所需的元器件及其数量，如表 3-22 所示。

表 3-22　W5100 网络模块元器件清单

元器件名称	数　量
Arduino 开发板	1
W5100 网张模块	1

相关代码：

```
#include <SPI.h>
```

```
#include <Ethernet.h>
#include <dht11.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,111,2);
dht11 DHT11;
#define DHT11PIN 5
int R1,R2;
int num = 0;
int sit;

  //设定 MAC 地址、IP 地址,IP 地址需要参考本地网络设置
byte mac[] = { 0xDE, 0x3D, 0x3E, 0xEF, 0x3E, 0x3D };
IPAddress ip(192,168,1,170);
  //初始化 Ethernet 库,HTTP 默认端口为 80
EthernetServer server(80);
void setup() {
  //开始 Ethernet 连接,并作为服务器初始化
  Ethernet.begin(mac, ip);
  server.begin();
}
void loop() {
  DHT11.read(DHT11PIN);
  //监听客户端传来的数据
  EthernetClient client = server.available();
  if (client) {
    Serial.println("new client");
    //一个 Http 请求结尾必须带有回车换行
    boolean currentLineIsBlank = true;
    while (client.connected()) {
      if (client.available()) {
        char c = client.read();
        Serial.write(c);
        //如果收到空白行,说明 Http 请求结束,并发送响应消息
        if (c == '\n' && currentLineIsBlank) {
          //发送标准的 HTTP 响应
          client.println("HTTP/1.1 200 OK");
          client.println("Content-Type: text/html");
          client.println("Connection: close");
          client.println();
          client.println("<!DOCTYPE HTML>");
          client.println("<html><head><meta
              charset=\"UTF-8\"><title>BUPT_Classroom</title>");
          client.println("<body><div
          align=\"center\"><h1>BUPT_Classroom</h1>");
```

```
            //添加一个 meta 刷新标签，浏览器会每 5s 刷新一次
            //如果此处刷新频率设置过高,可能会出现网页卡死的状况
            client.println("<meta http-equiv=\"refresh\" content=\"5\">");
            //显示温湿度
            client.print("Humidity (%): ");
            client.println((float)DHT11.humidity,2);
            client.println("<br />");
            client.print("Temperature (oC): ");
            client.println((float)DHT11.temperature, 2);
            client.println("<br />");
            client.println("<br />");
            //显示座位数及学生人数
            client.println("Student Number: ");
            client.println(num);
            client.println("<br />");
            client.println("Available Seats:");
            client.print(N-num);
            client.println("</body>");
            client.println("</html>");
            break;
          }
          if (c == '\n') {
            //已经开始一个新行
            currentLineIsBlank = true;
          }
          else if (c != '\r') {
            //在当前行已经得到一个字符
            currentLineIsBlank = false;
          }
        }
      }

      //等待浏览器接收数据
      delay(1);
      //断开连接
      client.stop();
      Serial.println("client disonnected");
    }
}
```

3.4.4　产品展示

项目整体实物图如图 3-44 所示。

最终演示效果图如图 3-45 所示。

图 3-44　整体实物图

图 3-45　最终演示效果图

3.4.5　故障及问题分析

（1）问题：LCD 屏数据显示出现乱码，可能是数据刷新的速度太快，LCD 屏反应的时间过短。

解决方案：将红外计数的数据推送延迟，给 LCD 屏一个反应的时间。

（2）问题：红外计数不准确。代码中给两个红外对射的数据读取加了一个估测的时间间隔，也由于这个时间间隔的不确定性，使得在实验中，对射无法准时地获取到人经过的

信号。

解决方案：用两个 for 循环作为触发的开关，舍去了估测的时限，但是在物理装配上，必须得保证两个对射的距离较近，来规避出现只触及一个红外，而不触及另一个红外的极端情况。计数的准确性大大提高，计数的速率很快，同时也能解决另外一个极端情况：当一个人同时触到两个红外（只触及到一个红外的情况物理上已避免）。

（3）问题：数据无法通过 W5100 网络模块传输。最初用 W5100＋YEELINK 来传送数据，但是有些数据不是经过传感器得到的，无法传到 YEELINK 中。

解决方案：用 W5100 建立简单的网页，并将数据传递到该网页上。

3.4.6　元器件清单

完成该项目所需的元器件及其数量，如表 3-23 所示。

表 3-23　智能教室设计元器件清单

元器件名称	数　量
Arduino 开发板	1
红外对射	2
W5100 网络模块	1
温湿度传感器	1
LCD1602 显示屏	1
1kΩ 电阻	2
导线	若干

参考文献

[1] Bradski G，Kaehler A. 于仕琪，刘瑞祯，译. 学习 OpenCV（中文版）[M]. 北京：清华大学出版社，2009.

[2] Robert Laganiere. 张静，译. OpenCV2 计算机视觉编程手册[M]. 北京：科学出版社，2013.

[3] 宋楠，韩广义. Arduino 开发从零开始学：学电子的都玩这个[M]. 北京：清华大学出版社，2014.

[4] Stephen Prata. 张海龙，袁国忠，译. C++ Primer Plus 中文版[M]. 第 6 版. 北京：人民邮电出版社，2012.

[5] 李永华，高英，陈青云. Arduino 软硬件协同设计实战指南[M]. 北京：清华大学出版社，2015.

3.5　项目 13：智能分类垃圾桶

设计者：曾波，王丽娜，边雯皓

3.5.1　项目背景

将可回收垃圾和不可回收垃圾混在一起处理，既是对可再利用资源的一种浪费，也是对

环境的一种极大伤害。如今，环境问题已经引起广泛关注，许多人已经具有垃圾分类的意识，并且街道的垃圾箱通常也有可回收与不可回收之分，但人们仍不能很好地实现垃圾的正确分类。基于目前的情况，本项目研究一种可自动识别垃圾种类的智能垃圾桶，帮助人们完成正确的垃圾分类。此外，本项目通过实现自动测满以及自动开盖的功能，使垃圾分类更加智能化。

3.5.2 创意描述

本项目的智能分类垃圾桶的创意主要有三点：有人放垃圾时，垃圾桶自动判别垃圾的种类；有人靠近垃圾桶时，垃圾桶自动开盖，之后自动关闭；实现自动测满。

3.5.3 功能及总体设计

在智能分类垃圾桶上添加红外感应模块，实现有人靠近时，自动开盖；在垃圾桶的顶部安放超声波测距模块，以测试垃圾桶是否已满；通过图像识别对放入的垃圾进行简单的分类识别。

1. 功能介绍

该智能垃圾桶实现的主要功能有：红外感应，检测有人靠近时，自动开盖；识别特定颜色的物体，简单分类；自动测满功能。

2. 总体设计

要实现上述功能，将项目分为三部分进行设计：垃圾桶顶超声波测距部分、红外感应部分和识别部分。

1）整体框架图

项目整体框架图如图 3-46 所示。

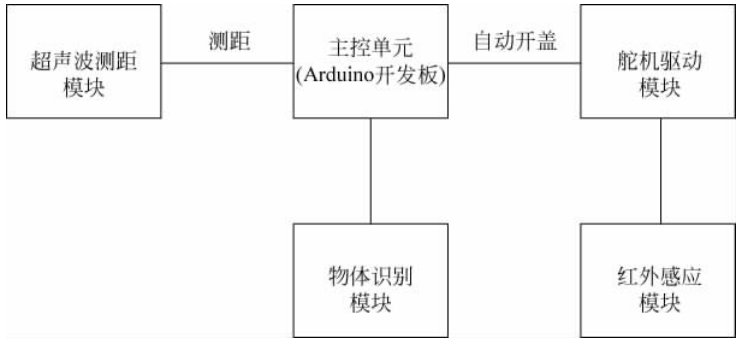

图 3-46 整体框架图

2）系统流程图

系统流程图如图 3-47 所示。

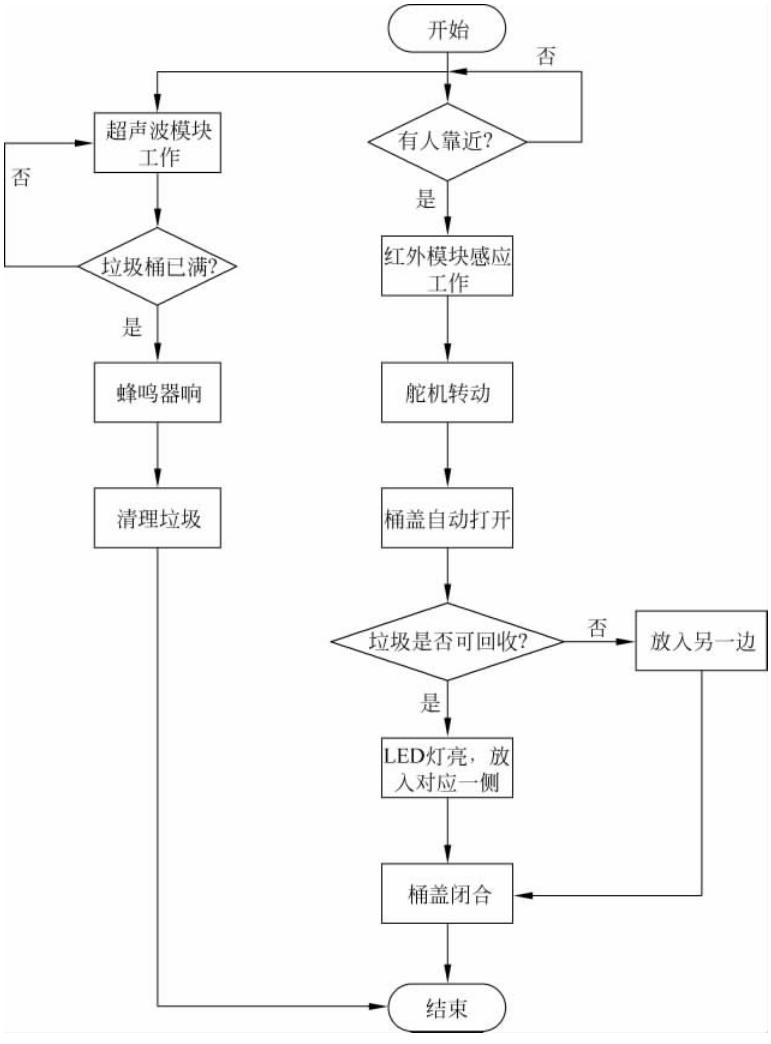

图 3-47　系统流程图

3）总电路图

系统总电路图如图 3-48 所示。

3．模块介绍

该项目主要分为三个模块：超声波测距模块、红外感应模块、识别模块。

1）超声波测距模块

功能介绍：将超声波测距模块装在垃圾桶顶部，通过设定的距离实现对桶内垃圾高度的测定，当垃圾的高度达到设置的参数时，蜂鸣器将发出声音，完成测满功能。

元器件清单：该模块所需的元器件及其数量如表 3-24 所示。

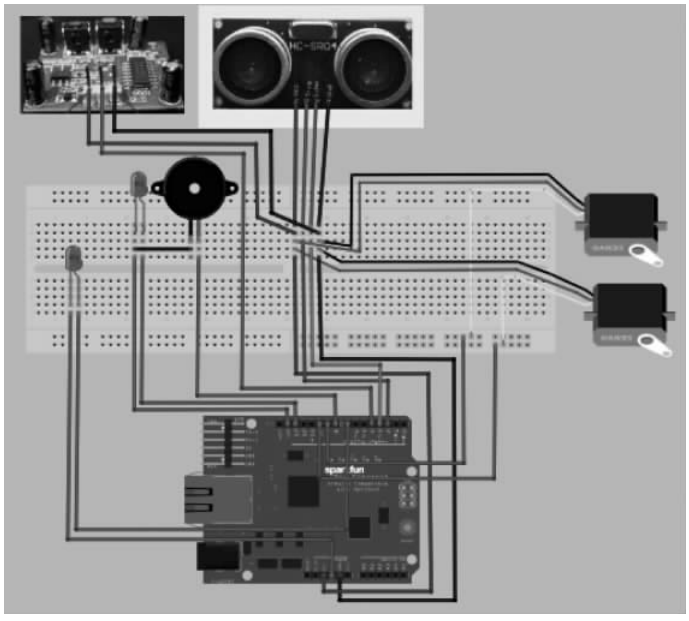

图 3-48　总电路图

表 3-24　超声波测距模块元器件清单

元器件名称	数　量
Arduino 开发板	1
超声波测距模块	1
面包板	1
蜂鸣器	1

电路图：该模块电路图如图 3-49 所示。

如图 3-49 所示，超声波测距模块有 GND 端和 VCC 端，分别接在板上的 GND 和 5V 端。

相关代码：

```
const int TrigPin = 2;
const int EchoPin = 3;
float cm;
void setup()
{
Serial.begin(9600);
pinMode(TrigPin, OUTPUT);
pinMode(EchoPin, INPUT);
pinMode(8,OUTPUT);
}
void loop()
{
digitalWrite(8, LOW);
```

```
digitalWrite(TrigPin, LOW);                    //低电平发一个短时间脉冲
delayMicroseconds(10);
digitalWrite(TrigPin, HIGH);
delayMicroseconds(10);
digitalWrite(TrigPin, LOW);

cm = pulseIn(EchoPin, HIGH) / 58.0;            //将回波时间换算成 cm
cm = (int(cm * 100.0)) / 100.0;                //保留两位小数
if (cm >= 2 && cm <= 10)
digitalWrite(8, HIGH);
}
```

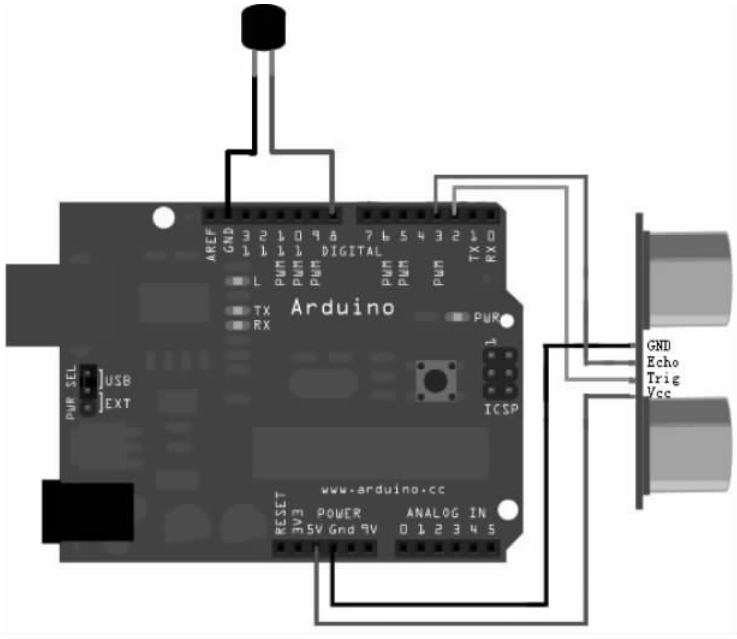

图 3-49　超声波测距模块电路图

2）红外感应模块

功能介绍：当有人靠近垃圾桶时，红外感应模块工作，启动舵机，垃圾桶自动开盖。

元器件清单：该模块所需的元器件及其数量如表 3-25 所示。

表 3-25　红外感应模块元器件清单

元器件名称	数　量
Arduino 开发板	1
红外感应模块	1
面包板	1
舵机	2

电路图：该模块电路图如图 3-50 所示。

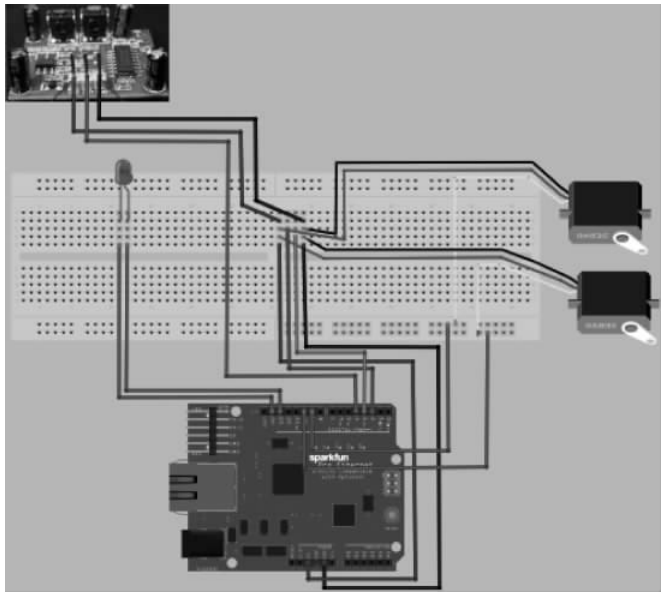

图 3-50　红外感应模块电路图

如图 3-50 所示，结合红外感应模块和 Arduino 开发板，当检测到有人时，Arduino 开发板发出指令，驱动舵机。其中，红外感应有 VCC 端和 GND 端，分别接在开发板上的 5V 和 GND 端。

相关代码：

```
#include <Servo.h>
Servo myservo1;
Servo myservo2;
int pos1 = 0;
int pos2 = 180;
int Sensor = 4;
int ina = 13;
void setup() {
     myservo1.attach(9);
     myservo2.attach(10);
     pinMode(ina, OUTPUT);                          //13 引脚定义为输出
     Serial.begin(9600);                            //设置下载程序的串口的波特率为 9600
     pinMode(Sensor, INPUT);                        //2 引脚定义为输入
}
void loop() {
    int SensorState = digitalRead(Sensor);          //读取 2 引脚的电平
    while(SensorState == 1){
    SensorState = digitalRead(Sensor);              //当 2 引脚为高电平时，设置 13 引脚为高
```

```
    digitalWrite(ina,HIGH);
    for(pos1 = 0,pos2 = 180; pos1 < 180,pos2 > 0; pos1 += 1,pos2 -= 1)
  {  myservo1.write(pos1);
     myservo2.write(pos2);
     delay(100);
  }
    delay(1000);                                    //延时 1000ms
    }
    while(SensorState == 0){                        //当 2 引脚为低电平时,设置 13 引脚为低
       SensorState = digitalRead(Sensor);
       digitalWrite(ina,LOW);
    }
    delay(100);                                     //延时 100ms
}
```

3）识别模块

功能介绍：通过 OpenCV 调用计算机摄像头，通过摄像头识别指定颜色的物体（针对农夫山泉的水瓶，将颜色设定为红色）。

元器件清单：该模块所需的元器件及其数量如表 3-26 所示。

表 3-26　识别模块元器件清单

元器件名称	数　量
Arduino 开发板	1
摄像头	1
OpenCV 工具	—
计算机	1

相关代码：OpenCV 工具将图片转化为二值图，且将图片中的红色部分变为二值图中的白色部分。

```
IplImage * colorSearch(string pic)
{    IplImage * img = cvLoadImage(pic.c_str());
                                                    //cvLoadImage("E:\\testface\\color2.jpg");
//测试图片
    //在 HSV 空间中处理图像
    IplImage *channelH = cvCreateImage(cvGetSize(img), IPL_DEPTH_8U, 1);
    //创建 H 通道
    IplImage *channelS = cvCreateImage(cvGetSize(img), IPL_DEPTH_8U, 1);
    //创建 S 通道
    IplImage *channelV = cvCreateImage(cvGetSize(img), IPL_DEPTH_8U, 1);
    //创建 V 通道
    IplImage *HSV = cvCreateImage(cvGetSize(img), IPL_DEPTH_8U, 3);
    //创建 HSV 图像
    cvCvtColor(img, HSV, CV_BGR2HSV);               //将 RGB 空间转换为 HSV 空间
    //分离通道,HSV->H、S、V
for (int i = 0; i < HSV->height; i++)
```

```
{
unsigned char * pHSV = (unsigned char * )(HSV -> imageData + i * HSV -> widthStep);
unsigned char * pH = (unsigned char * )(channelH -> imageData + i * channelH -> widthStep);
unsigned char * pS = (unsigned char * )(channelS -> imageData + i * channelS -> widthStep);
unsigned char * pV = (unsigned char * )(channelV -> imageData + i * channelV -> widthStep);
        for (int j = 0; j < HSV -> width; j++)
        {
            pH[j] = pHSV[3 * j + 0];
            pS[j] = pHSV[3 * j + 1];
            pV[j] = pHSV[3 * j + 2];
        }
    }

    //把 0 < H < 8 || 160 < H < 180(红色)的地方设置为 1,其他地方设置为 0
    //色调(H),饱和度(S),亮度(V)
    //只有 H 通道表示颜色,只需对 H 通道进行处理
//cvThreshold( const CvArr * src, CvArr * dst, double threshold, double max_value, int
threshold_type )
    //对灰度图像进行阈值操作得到二值图像
    //src: 原始数组 (单通道 , 8 位或 32 位浮点数)
    //dst: 输出数组,必须与 src 的类型一致,或者为 8 位
    //threshold: 阈值
    //max_value: 使用 CV_THRESH_BINARY 和 CV_THRESH_BINARY_INV 的最大值
    //threshold_type: 阈值类型 threshold_type = CV_THRESH_BINARY:如果 src(x,y)> threshold 0,
dst(x,y) = max_value, 否则 dst(x,y) = 0
    //threshold_type = CV_THRESH_BINARY_INV:如果 src(x,y)> threshold,dst(x,y) = 0; 否则,dst
(x,y) = max_value

    cvThreshold(channelH, channelS, 8, 1, CV_THRESH_BINARY_INV);
    //0 < H < 8 的二值图像
    cvThreshold(channelH, channelH, 160, 1, CV_THRESH_BINARY);
    //160 < H < 180 的二值图像
    //将 H、S 通道合并,并转换为二值图像
for (int i = 0; i < channelH -> height; i++)
{
    unsigned char * pH = (unsigned char * )(channelH -> imageData + i * channelH ->
widthStep);
    unsigned char * pS = (unsigned char * )(channelS -> imageData + i * channelS ->
widthStep);
        for (int j = 0; j < channelH -> width; j++)
        {
            if (pH[j] || pS[j])
            {
                pH[j] = (unsigned char)255;
            }
        }
    }
```

3.5.4　产品展示

整体实物图如图 3-51 所示，最终演示效果图如图 3-52 所示。

图 3-51　整体实物图

如图 3-51 所示，左侧为所识别的指定垃圾，右侧则为其他。

图 3-52　最终演示效果图

如图 3-52 所示，当有人靠近垃圾桶时，垃圾桶自动开盖。

3.5.5　故障及问题分析

（1）问题：摄像头识别红色物体时可以正常运行，但当物体没有红色时，程序会出现异常中断。

解决方案：编写一个判断函数，int panduan(IplImage * img)，当二值图中没有红色时，判断函数返回 0；当二值图中有红色时，判断函数返回 1。

（2）问题：Arduino 开发板在给特定端口高电平时，不能驱动蜂鸣器等器件。

解决方案：在电机的驱动程序里面，通过相关代码的调整，可以直接在 Arduino 开发板上驱动蜂鸣器。

(3) 问题：当红外感应模块工作时，无明显现象。

解决方案：连接 LED 灯，观察 LED 灯的亮灭与电机转动是否同步，确定红外感应模块是否正常工作。

(4) 问题：颜色识别部分代码程序可以和超声波测距模块程序很好地融合，但无法融合舵机控制程序。原因在于 Servo 库文件确实和 PWM 输出有冲突，无法同时使用。

解决方案：使用两块 Arduino 开发板，在第二块 Arduino 开发板中，单独下载舵机控制程序。

3.5.6 元器件清单

完成本项目所需要的元器件及其数量如表 3-27 所示。

表 3-27 智能垃圾桶设计元器件清单

元器件名称	数 量	元器件名称	数 量
Arduino 开发板	1	蜂鸣器	1
网线	1	舵机	2
数据线	1	纸盒	2
面包板	1	超声波测距模块	1
摄像头	1	人体红外感应模块	1
计算机	1	杜邦线	若干

参考文献

[1] Bradski G，Kaehler A. 于仕琪，刘瑞祯，译. 学习 OpenCV(中文版)[M]. 北京：清华大学出版社，2009.

[2] 谭浩强. C 语言程序设计[M]. 第 4 版. 北京：清华大学出版社，2010.

[3] CSDN 论坛. 使用 OpenCV 进行图像的颜色识别[J/OL]. http://blog.csdn.net/scottly1/article/details/23046847.

[4] 李永华，高英，陈青云. Arduino 软硬件协同设计实战指南[M]. 北京：清华大学出版社，2015.

3.6 项目 14：语音控制台灯

设计者：陈英杰，杨文哲

3.6.1 项目背景

近年来，随着科学技术的发展和网络时代的到来，人类社会已经进入了信息时代，信息的交换、人与人之间的交流变得更加密切。然而，社会的发展已经不仅仅满足于人与人之间的沟通，更希望通过某些方式和手段，实现人与物之间的沟通，让身边的物品也能“听懂”人类的语言。

为了能够实现这一目的，本项目利用简单常见的生活用品——台灯作为切入点。通过研究制造出语音台灯，可以更加方便地来调整和控制灯光，甚至不需要动手，只需要随口一个命令，就能随心所欲掌控台灯的状态。例如，当我们说“打开”时，它可以自动打开；当我们说“关闭”时，它也能自动关闭；当我们说“闪烁”时，它就会完成一闪一闪的功能，等等。

3.6.2　创意描述

台灯是日常生活中大家都熟悉的物品，声控灯也是屡见不鲜，但是，一个不仅能听见声音，还能听懂我们说话的内容，根据所接收到的语音指令，能做出相应动作的台灯，目前并不多见，这正是本项目所要实现的功能。

如果大家看过“皮克斯”电影公司出品的电影，那么对片头的那个一跳一跳的小台灯一定有印象。因此，本项目的语音台灯起名为“皮卡斯”，它可以对人类的语音进行识别，并且做出回应的动作，不必是特定的人，“皮卡斯”是一盏任何人的话它都能听懂的智能台灯。

3.6.3　功能及总体设计

基于上述创意，采用语音识别模块，实现台灯“听懂”人类的语音，是本项目研究和设计中最重要的部分。

1. 功能介绍

可以通过语音控制台灯实现开灯、关灯、抬升灯头、降低灯头、延时打开、延时关闭、闪烁、亮度变暗、逐渐变暗的功能。

2. 总体设计

首先，要让台灯“听懂”人类所说的话，为台灯添加语音识别部分；为了可以让台灯随着语音的控制而移动，需要在台灯上添加步进电机，从而实现上述的抬升灯头、降低灯头的功能。

1）整体框架图

项目整体框架图如图 3-53 所示。

如图 3-53 所示，Arduino 为中心模块，与语音控制模块、升压模块、步进电机分别连接。步进电机连接台灯灯头，实现动作；升压模块连接台灯灯泡，实现照明。

2）系统流程图

系统流程图如图 3-54 所示。

如图 3-54 所示，Arduino 为语音控制模块、升压模块、步进电机提供驱动电压，语音控制模块将接收到的语音信息转化为电信号，为升压模块和步进电机提供指令，再由升压模块控制台灯的照明，由步进电机控制台灯的动作。

3）总电路图

系统总电路图如图 3-55 所示。

如图 3-55 所示，Arduino 开发板的 5V 和 GND 分别接在面包板上作为 5V 输入端和

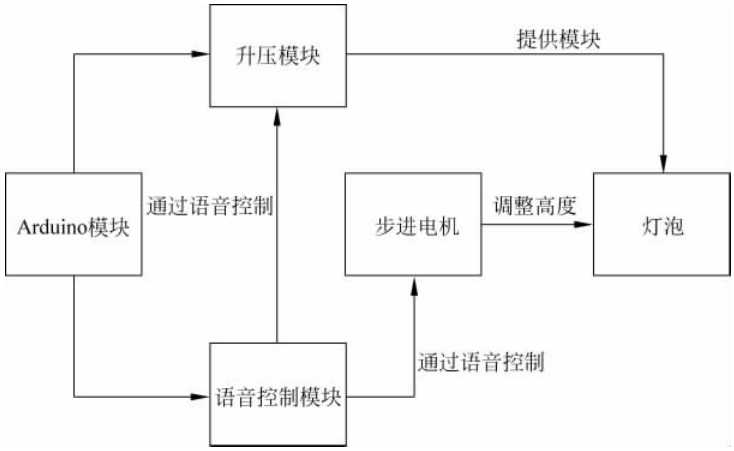

图 3-53　整体框架图

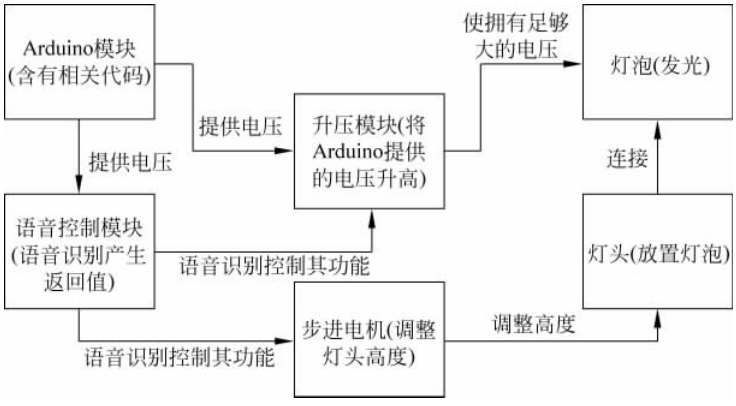

图 3-54　系统流程图

GND 端；语音控制模块 VCC 接 5V 输入端，GND 接 GND 端，MIC 口接麦克风，RX0 接 Arduino 开发者的 RX0 输入口；L298N “＋”(正极)接 5V 输入端，“－”(负极)接 GND 端，A1、B1、C1、D1 输入端分别接在 Arduino 开发板的 2、3、4、5 输出口；步进电机接 L298N 的 A1、B1、C1、D1 输出端；升压模块 IN＋(输入正极)接 Arduino 开发板输出端口 10，IN-(输入负极)接 Arduino 开发板输出端 GND，OUT＋(输出正极)接灯的正极，OUT-(输出负极)接灯的负极。

3. 模块介绍

该项目主要分为三个模块：语音控制模块、升压模块、步进电机模块。

1) 语音控制模块

功能介绍：运用 LP_COMM V2.22 将特定语句写入语音控制模块，为每一语句设置返回值。

如图 3-56(a)所示，a0 为写入的意思，ni hao 是写入的语句的拼音发音，0x00 为设置该语句对应返回值 00。然后，由麦克风接收语音信号，通过模块识别特定的语句来产生不同的返回值，如图 3-56(b)所示，最下方框中的部分为测试时不同语句获得的返回值显示。当语句成功写入以后，就能通过不同的返回值来作为不同的动作指令，借助 Arduino 编程使台灯完成不同的动作。

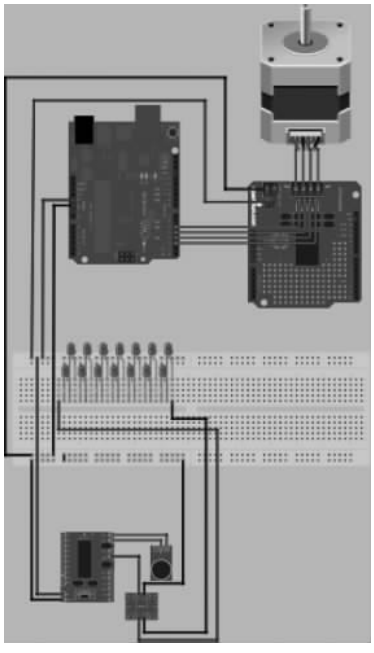

图 3-55　总电路图

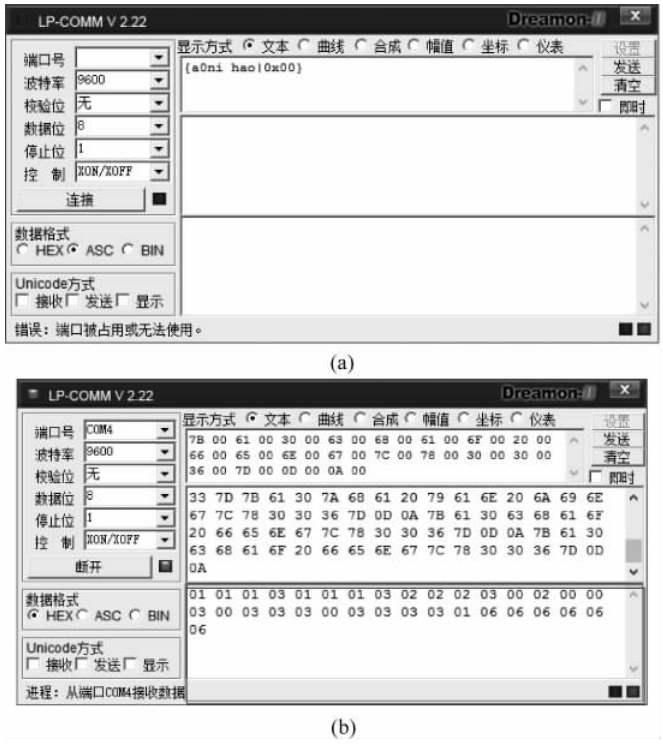

图 3-56　特定语句写入语音控制模块

元器件清单：非特定人语音控制模块 LD3320、小型麦克风。

电路图：语音控制模块的引脚图如图 3-57 所示，语音控制模块的连接图如图 3-58 所示。

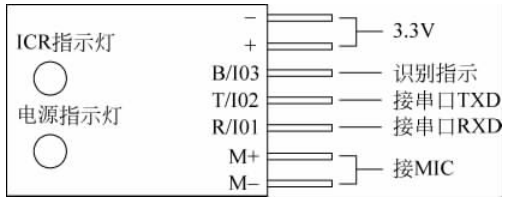

图 3-57　语音控制模块引脚图

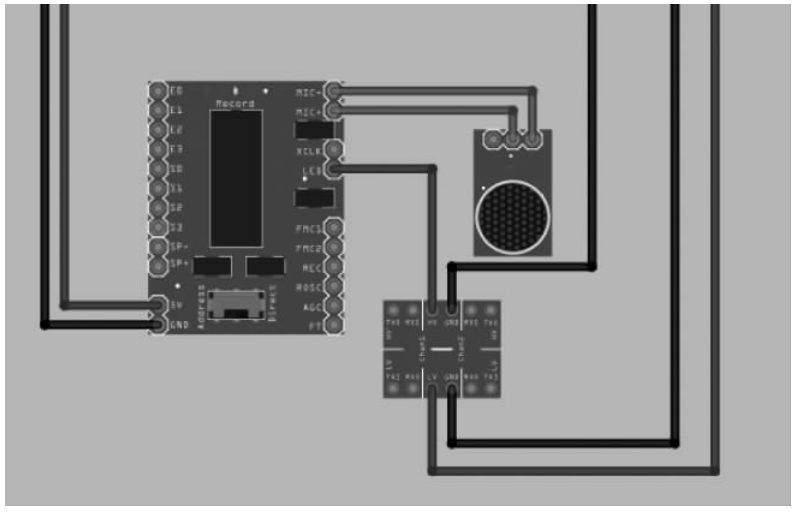

图 3-58　语音控制模块连接图

相关代码：台灯所有的指令接收都是通过语音控制模块来实现的，这一部分的代码主要是设置各个返回值对应的 Arduino 输出信号。例如，当接收到返回值为 00 的语音信号(case 0x00:)，让数字端口 10 对应输出高电压(digitalWrite(10, HIGH);)。

```
void loop()
{
  int a; a = 512;
  if(Serial.available())
  {
    int inByte = Serial.read();
    switch(inByte)
    {
      case 0x00:
        digitalWrite(10, HIGH);
        break;
      //返回值为 00 时，打开灯
      case 0x01:
        digitalWrite(10, LOW);
        break;
      //返回值为 01 时，关闭灯
      case 0x02:
       while(a -- )
      {
       for(int i = 2;i < 6;i++)
       {
         digitalWrite(i,1);
         delay(3);
         digitalWrite(i,0);
```

```
  }
}
break;
//返回值为 02 时,控制步进电机顺时针旋转一周
case 0x03:
 while(a--)
 {
 for(int i=5;i>1;i--)
  {
  digitalWrite(i,1);
  delay(3);
  digitalWrite(i,0);
   }
  }
 break;
//返回值为 03 时,控制步进电机逆时针旋转一周
case 0x04:
 delay(1000);
 digitalWrite(10, HIGH);
 break;
//返回值为 04 时,延迟 5s 后台灯打开
case 0x05:
 delay(1000);
 digitalWrite(10, LOW);
 break;
//返回值为 05 时,延迟 5s 后台灯关闭
case 0x06:
 for(int count=0;count<3;count++)
 {
 digitalWrite(10, HIGH);
 delay(1000);
 digitalWrite(10, LOW);
 delay(1000);
 }
 break;
//返回值为 06 时,台灯闪烁三次
case 0x07:
 n=200;
 analogWrite(10,n);
 break;
//返回值为 07 时,台灯亮度减小
case 0x08:
 while (n>=0)
 {
  n=n-10;
  analogWrite(10,n);
  delay (300);
```

```
            }
            break;
           //返回值为 08 时,台灯逐渐变暗并且最终熄灭
        }
      }
    }
```

2）升压模块

功能介绍：由于 Arduino 开发板的输出电压最大只有 5V，无法驱动大量的 LED 来实现台灯的功能，所以使用了升压模块，通过将输出电压升高来驱动更多的灯，完成台灯的功能。升压模块可以将输入为 3.5～18V 的电压提升至 4～24V，以极大地提升 Arduino 的驱动能力，让台灯功能得以更好地实现。

元器件清单：ITEAD 直流升压电路电源模块、LED 发光二极管 4×7 个。

电路图：升压模块的端口图如图 3-59 所示，升压模块的实物图如图 3-60 所示，升压模块在电路中的连接图如图 3-61 所示。

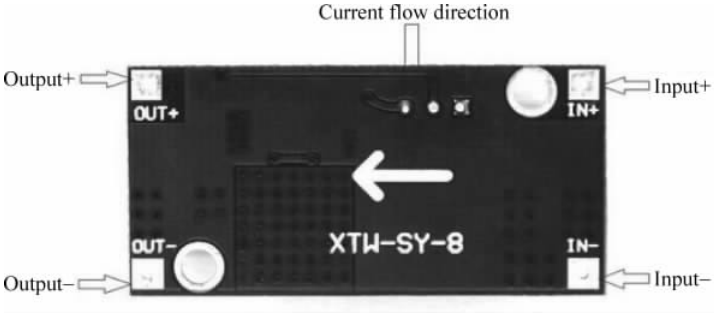

图 3-59　升压模块端口图

图 3-60　升压模块实物图

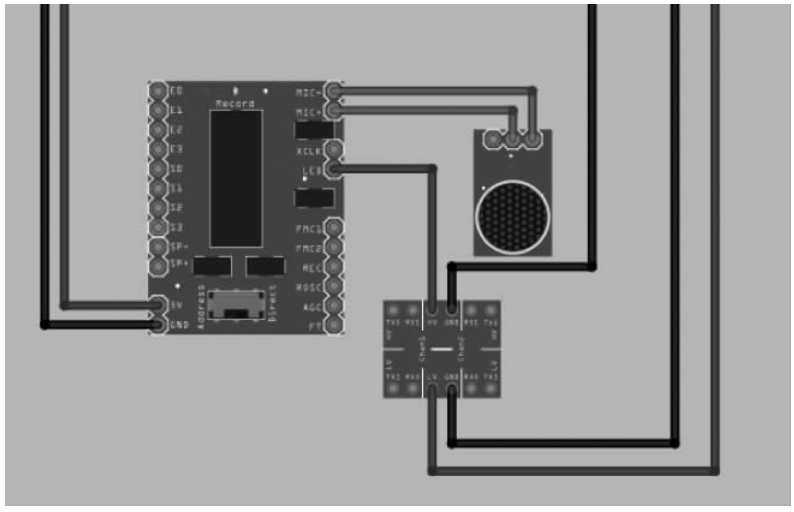

图 3-61　升压模块电路连接图

3）步进电机模块

功能介绍：步进电机是将电脉冲信号转变为角位移或线位移的开环控制元器件。在非超载的情况下，电机的转速、停止的位置只取决于脉冲信号的频率和脉冲数，而不受负载变化的影响，当步进驱动器接收到一个脉冲信号，它就驱动步进电机按设定的方向转动一个固定的角度，称为“步距角”，它的旋转是以固定的角度一步一步运行的。可以通过控制脉冲个数来控制角位移量，从而达到准确定位的目的；同时，可以通过控制脉冲频率来控制电机转动的速度和加速度，从而达到调速的目的。

为了实现台灯灯头的动作部分，将一个步进电机固定在灯架上，用黑色丝线将步进电机的转动部分和台灯的灯头部分连接起来，当步进电机转动时，通过力的传递牵引台灯的灯头，使其发生上下动作。

元器件清单：步进电机、黑色丝线、L298N。

电路图：步进电机的电路连接图如图 3-62 所示。

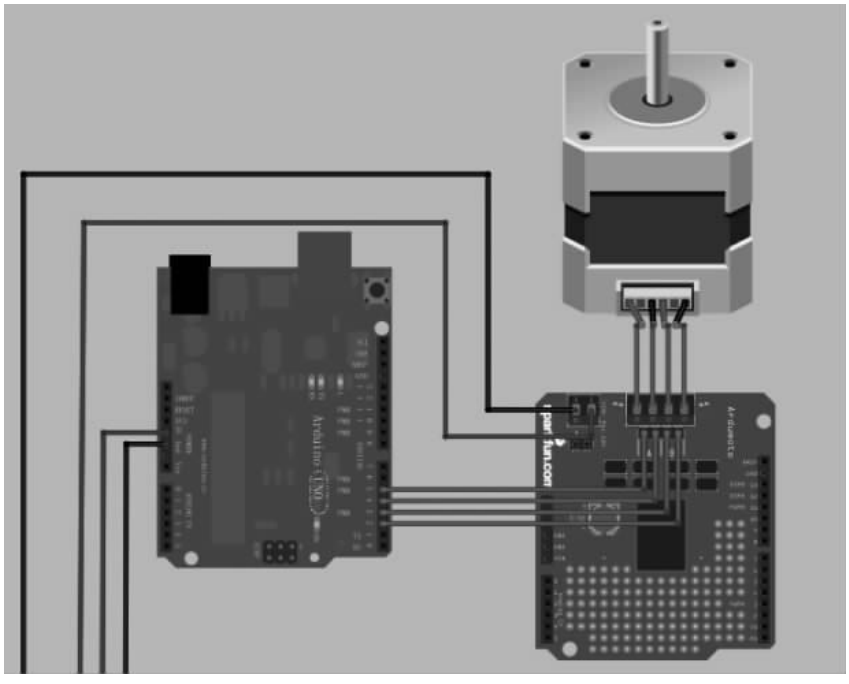

图 3-62　步进电机电路连接图

相关代码：台灯动作部分是通过语音模块来实现的，这一部分的代码主要是设置步进电机的动作。例如，当返回值为 02 时(case 0x02:)，Arduino 的数字端口 2、3、4、5 从低电平到高电平(digitalWrite(i,1);)再从高电平到低电平(digitalWrite(i,0);)交替变换，使得转子发生转动，当转动部分旋转一周以后停下。

```
int a; a = 512;
```

```
case 0x02:
  while(a--)
 {
  for(int i=2;i<6;i++)
  {
    digitalWrite(i,1);
    delay(3);
    digitalWrite(i,0);
   }
 }
 break;
 //返回值为02时,控制步进电机顺时针旋转一周,牵引灯头降低
 case 0x03:
  while(a--)
  {
  for(int i=5;i>1;i--)
   {
   digitalWrite(i,1);
   delay(3);
   digitalWrite(i,0);
    }
   }
  break;
 //返回值为03时,控制步进电机逆时针旋转一周,牵引灯头抬高
```

3.6.4 产品展示

整体实物连接图如图 3-63 所示。

最终效果演示图如图 3-64 所示。当接收到语音信号打开时,台灯打开。

图 3-63 整体实物连接图

图 3-64 最终演示效果图

3.6.5　故障及问题分析

（1）问题：由 Arduino 连接语音控制模块时可以写入语句，但是无返回值。

解决方案：购买了 TTL 转 232 模块，将语音控制模块通过该模块接入计算机，再利用 LP_COMM V2.22 串口调试软件，获取了语音控制模块的返回值，以此完成了对语音控制模块的调试。

（2）问题：步进电机如何带动灯头移动？

解决方案：通过多次尝试，最终决定用线将灯头和步进电机的转动部分连接起来，利用力的传递，通过转动部分转动的扭力给拉线提供拉力，带动灯头运动。

（3）问题：代码调试的时候关于 Arduino 输出电压大小的值不知如何设置。

解决方案：查询了相关书籍和资料，了解 Arduino 输出不同大小的电压值的方法。

3.6.6　元器件清单

完成该项目所需的元器件及其数量如表 3-28 所示。

表 3-28　语音控制台灯元器件清单

元器件名称	数　量
Arduino 开发板	1
LED 灯	28
杜邦线	若干
语音控制模块	1
步进电机	1
升压模块	1

参考文献

[1] Bradski G，Kaehler A. 于仕琪，刘瑞祯，译. 学习 OpenCV（中文版）[M]. 北京：清华大学出版社，2009.

[2] Robert Laganiere. 张静，译. OpenCV2 计算机视觉编程手册[M]. 北京：科学出版社，2013.

[3] 宋楠，韩广义. Arduino 开发从零开始学：学电子的都玩这个[M]. 北京：清华大学出版社，2014.

[4] Stephen Prata. 张海龙，袁国忠，译. C++ Primer Plus 中文版[M]. 第 6 版. 北京：人民邮电出版社，2012.

[5] 李永华，高英，陈青云. Arduino 软硬件协同设计实战指南[M]. 北京：清华大学出版社，2015.