**项目编号：20151245**

** （华东）**

**国家大学生创新创业训练计划**

**结 题 总 结**

**项目名称： “智”行天下**

**项目级别：**  **校级**

**负 责 人：** **宋真**

**所在院部：** **计算机与通信工程学院专业班级：物联网1401**

**指导教师： 成玲**

**教务处 制**

基于STM32的智能灭火机器人设计

摘 要

本设计是针对智能灭火机器人的制作与研究，以ARM公司生产STM32F4 DISCOVERY单片机为主控核心，集电机驱动模块、电源管理模块、蓝牙串口通信模块[1]、android控制端模块于一体的智能灭火机器人。蓝牙串口通信模块采用了FBT06\_LPDB针插蓝牙模块，与主控板进行串口通信，同时与android手机进行通信；android控制端模块具有开启蓝牙、搜索蓝牙、控制小车等功能。本设计制作的灭火机器人具有简易灭火功能，用户可通过android控制端进行控制小车的运动，达到实现现场灭火的目的。

**关键词**：灭火；蓝牙串口通信；android控制

**Design of intelligent fire extinguishing robot based on STM32**

**Abstract**

The design is for intelligent fire-fighting robot production and research, ARM production STM32F4 DISCOVERY microcontroller as the main core, set the motor drive module, power management module, Bluetooth serial communication module, android control module in one of the intelligent fire extinguishing robot. Bluetooth serial communication module uses FBT06\_LPDB[2]Bluetooth module, and the main control board for serial communication, and with the android mobile phone communication; android console module has Bluetooth, search Bluetooth, control car and other functions. The design of the fire extinguishing robot has a simple fire function, the user can control the car through the android control, to achieve the purpose of the scene fire[3] **.**

**Keywords**：Extinguishing；Bluetooth Serial Communication；Android Control

目 录

[第1章 引言 1](#_Toc479862769)

[第2章 设计要求及设计方案 2](#_Toc479862770)

[2.1 设计要求 2](#_Toc479862771)

[2.1.1 基本要求 2](#_Toc479862772)

[2.1.2 发挥部分 2](#_Toc479862773)

[2.2 设计方案 2](#_Toc479862774)

[第3章 设计的主体内容 3](#_Toc479862775)

[3.1 系统硬件设计 3](#_Toc479862776)

[3.1.1电机驱动模块 3](#_Toc479862777)

[3.2 系统软件设计 5](#_Toc479862778)

[3.2.1 主控板程序设计 5](#_Toc479862779)

[3.2.2 android客户端程序设计 9](#_Toc479862780)

[3.3 系统调试 14](#_Toc479862781)

[3.3.1 系统调试时遇到的问题及解决方案 14](#_Toc479862782)

[3.3.2 调试心得 14](#_Toc479862783)

[3.3.3 系统调试的结果 15](#_Toc479862784)

[第4章 系统创新与结果分析 16](#_Toc479862785)

[4.1 系统创新 16](#_Toc479862786)

[4.2 评测与结论 16](#_Toc479862787)

[第5章 结论 17](#_Toc479862788)

[心得体会与感悟 18](#_Toc479862789)

[参考文献 20](#_Toc479862790)

[附 录 21](#_Toc479862791)

[附录一 21](#_Toc479862792)

[附录二 22](#_Toc479862793)

[附录三 23](#_Toc479862794)

# 第1章 引言

关于机器人，最早出现在科幻小说中，是人们对于未来世界的一种美好幻想。如今随着现代高端技术的发展，各种类型的机器人层出不穷，并且在众多的领域得到了广泛的应用。从最初只能运行简单程序执行单一动作的机器人，发展到可以完成复杂任务代替手工的智能机器人。

随着经济的发展，越来越多的高楼拔地而起，由于高层建筑的特殊性，发生火灾时，不能快速高效的灭火。相关资料显示，在过去五年时间，我国消防人员在执行火灾救援工作时伤亡惨烈，为了达到尽快救助火灾中的受害者，并最大限度的保证消防员的安全，灭火机器人研究被提到了议事日程。

一些发达国家将研究开发灭火机器人视为经济发展的重要保证，不仅是因为灭火机器人可以代替消防人员深入到达有毒、浓烟的高危险性火灾现场进行侦查检验、执行危险的灭火任务，提高了消防部队的灭火救援能力，更重要的是灭火机器人的研究避免更大的经济损失，保障消防人员的人身安全。因此从二十世纪八十年代开始，许多国家开始进行有关灭火机器人的研究。

我国灭火机器人的研究始于八十年代末期，如今已经取得了长足发展，但是仍处于初级阶段。与发达国家相比，我国灭火机器人在技术上还有很大的发展空间。

本课题主要是以实现手机端控制机器人实现灭火功能为目的，对灭火机器人的设计进行初步的探索。

# 第2章 设计要求及设计方案

## 2.1 设计要求

### 2.1.1 基本要求

（1）在遥控模式下，通过手机端控制智能灭火车启动，并控制小车的速度使其前进。

（2）出现火源，控制灭火车到达火源处并将其完全扑灭。

### 2.1.2 发挥部分

设置在自动模式下，小车能过自动寻轨前进、发现并扑灭火源。

## 2.2 设计方案

该系统分为电机驱动模块、电源管理模块、主控板、蓝牙通信模块、等4个模块，如图2.1所示：

stm32f4

主控板

电源管理模块

红外避障模块

蓝牙通信模块

其他输入模块

其它待扩展模块

风扇灭火模块

电机驱动模块

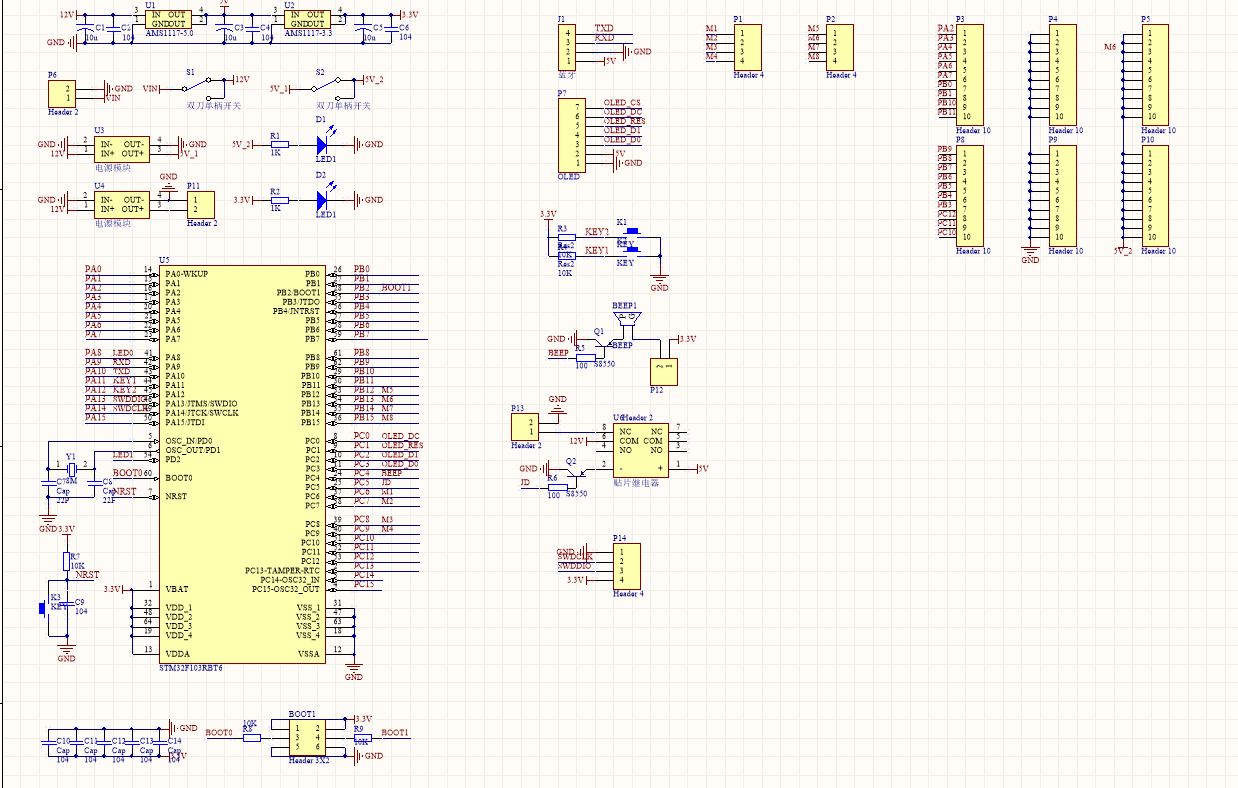
指示灯

图2.1 系统模块图

# 第3章 设计的主体内容

在开始进行设计时我们做了大量的准备工作，查阅灭火机器人的有关资料，了解灭火机器人的发展及不同的实现方案。之后，我们开始动手实践，下面从两个大的方面：系统的硬件设计和系统的软件设计，介绍我们的研究成果。

## 3.1 系统硬件设计

****图3.1原理图总览[4] [5]

### 3.1.1电机驱动模块

（1）L298N的封装

H 桥电路[6]虽然有着诸多的优点，但是在实际制作过程中，由于元件较多，电路的搭建也较为麻烦，增加了硬件设计的复杂度。

由于H 桥电路有诸多的优点，但是在实际制作过程中电路又比较麻烦，因此在本设计中我们采用H 桥集成电机驱动芯片L298。L298N 的工作原理和以上介绍的H 桥相同，引脚图如图3.2所示：

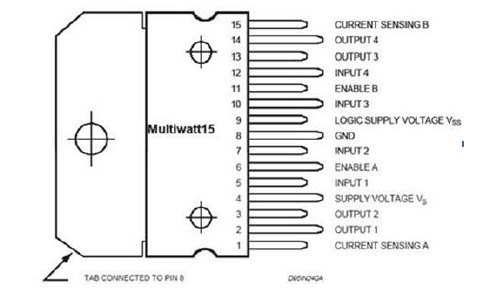


图3.2 L298N封装图

（2）L298N的原理图设计

L298N 是ST 公司生产的一种高电压、大电流电机驱动芯片。该芯片采用15 脚封装。主要特点是：工作电压高，最高工作电压可达46V；输出电流大，瞬间 峰值电流可达3A，持续工作电流为2A；额定功率25W。内含两个H 桥的高电压大电流全桥式驱动器，可以用来驱动直流电动机和步进电动机、继电器线圈等感性负载；采用标准逻辑电平信号控制；具有两个使能控制端，在不受输入信号影响的情况下允许或禁止器件工作有一个逻辑电源输入端，使内部逻辑电路部分在低电 压下工作；可以外接检测电阻，将变化量反馈给控制电路。使用L298N 芯片驱动电机，该芯片可以驱动一台两相步进电机或四相步进电机，也可以驱动两台直流电机。L298 的参考电路图如图3.3所示。

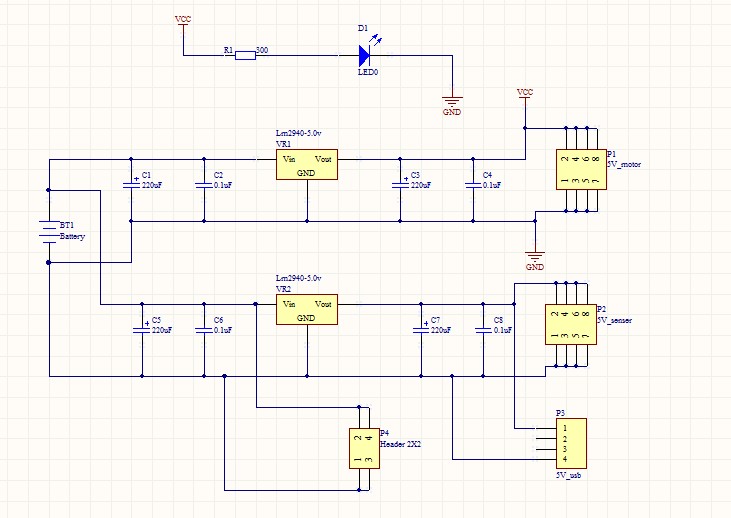


图3.3 电机驱动电路原理图

## 3.2 系统软件设计

### 3.2.1 主控板程序设计

（1） main程序设计

主函数主要分为延迟时间初始化、串口接收模块程序、电机初始化三部分。主函数的流程图如图3.4所示[7]：

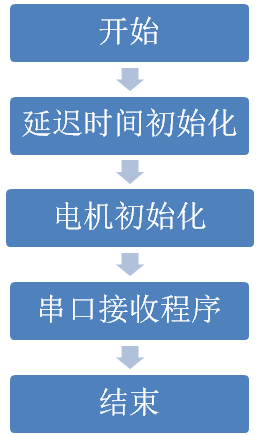


图3.4 主函数流程图

主函数程序代码如下[8]：

int main(void)

{

Stm32\_Clock\_Init(9); //系统时钟

uart\_init(72,19200); //串口初始化波特率19200

delay\_init(72); //延时初始化

init();//各模块初始化

while(1)

{

Get\_Control();

}

}

1. 功能函数

extern void Up(); //前进

extern void Left();//左转

extern void Right();//右转

extern void Back(); //后退

extern void Stop();//停车

extern void Add\_speed(); //加速

extern void Cut\_speed(); //减速

extern void Diaotou();//调头

extern void Auto(); //自动模式

（3）串口接收模块程序

if(USART\_RX\_STA&0x8000)

{

len=USART\_RX\_STA&0x3FFF;

for(t=0;t<len;t++)

{

USART3->DR=USART\_RX\_BUF[t];

while((USART3->SR&0X40)==0);

}

USART\_RX\_STA=0;

if(\*USART\_RX\_BUF==':')

{

Mobile\_Con\_Flag=0;

Up();

}

else if(\*USART\_RX\_BUF!=NULL)

{

Stop();

Mobile\_Con\_Flag=1;

}

}

主控板接收到蓝牙从串口传来的数据后存入Res变量，然后通过分支程序来选择执行前进、后退、左转、右转和停止等功能。该模块的程序流程图如图3.5所示：

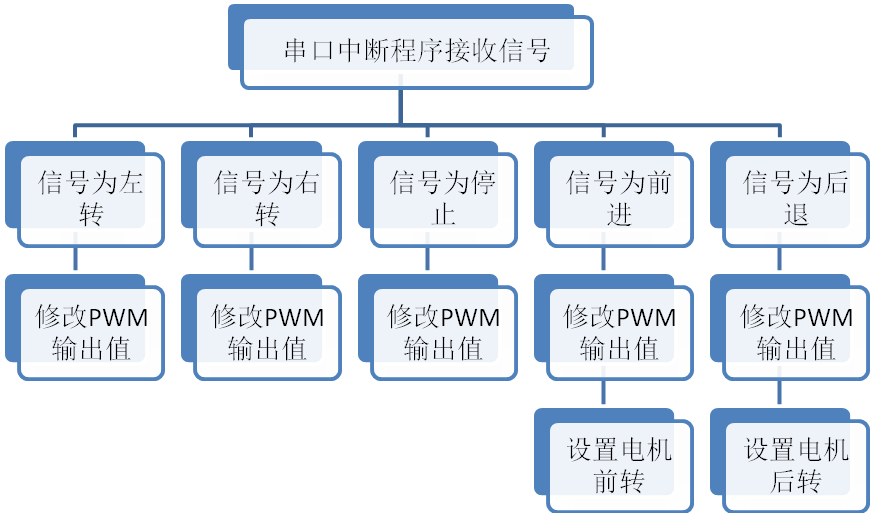


图3.5 串口接收程序流程图

（4）修改PWM输出值程序[9] [10]

该程序是基于“4\_PWM的实现”中的程序改编的。代码如下：

void Change\_PWM(int duty1,int duty2,int duty3,int duty4)

{

SCB->AIRCR=0x05AF00;// 中断优先级分组 抢占：响应=3:1

RCC->AHB1ENR|=(1<<2);// 打开GPIOC时钟

GPIOC->MODER|=0x000AA000;// pc6789第二功能，推挽输出

GPIOC->OSPEEDR|=0x000FF000;//输出速度为100m

GPIOC->PUPDR|=0x00055000;//上拉

GPIOC->AFR[0]|=0x22000000;//pc6789的第二功能为AF2

GPIOC->AFR[1]|=0x00000022;

RCC->APB1ENR|=(1<<1);//打开TIM3时钟

TIM3->PSC=83;//对84M时钟进行84分频，使得计数频率为1M

TIM3->ARR=10000;//周期为10ms

TIM3->EGR|=1;//产生一次更新时间

TIM3->CCMR1|=0x6060;//PWM模式1

TIM3->CCMR2|=0x6060;//PWM模式2

TIM3->CCR1=duty1;//1路PWM

TIM3->CCR2=duty2;//2路PWM

TIM3->CCR3=duty3;//3路PWM

TIM3->CCR4=duty4;//4路PWM

TIM3->CCER|=0x1111;//使能比较输出

TIM3->CCMR1|=0x0808;//启动预装载

TIM3->CCMR2|=0x8080;

TIM3->CR1|=1;//开始计时

}

(5)设置电机转向程序

改程序将电机驱动模块的8个输入端口接到了主控板的8个GPIO口，通过推挽输出，从而控制电机的转向，代码如下：

void Direction(int direction)

{

SysTick\_Config(SystemCoreClock / 1000); //时钟中断设为1ms

RCC->AHB1ENR |= 0x00000005; //使能GPIOA和GPIOD时钟

RCC->APB2ENR |= (1<<14); //使能syscfg时钟

if(direction==0)

{

GPIOA->MODER &= 0xffff0000; //设置PA0，1，2，3为输出

GPIOA->MODER |= 0x00005555;

GPIOA->OTYPER &= 0xFFFFff00; //设置PA0，1，2，3为推挽输出

GPIOA->OSPEEDR &= 0xffff0000; //设置PA0，1，2，3的输出速度为100M

GPIOA->OSPEEDR |= 0x0000ffff;

SYSCFG->CMPCR = 0x00000001; //使用IO补偿单元

GPIOA->PUPDR &= 0xffffff00; //设置PA0，1，2，3无上拉，无下拉

GPIOA->BSRRH = 0x00ff; //复位GPIOA\_BSRRH寄存器

GPIOA->BSRRL = 0x0055;

}

else

{

GPIOA->MODER &= 0xffff0000; //设置PA0，1，2，3为输出

GPIOA->MODER |= 0x0000005555;

GPIOA->OTYPER &= 0xFFFFff00; //设置PA0，1，2，3为推挽输出

GPIOA->OSPEEDR &= 0xffff0000; //设置PA0，1，2，3的输出速度为100M

GPIOA->OSPEEDR |= 0x0000ffff;

SYSCFG->CMPCR = 0x00000001; //使用IO补偿单元

GPIOA->PUPDR &= 0xffffff00; //设置PA0，1，2，3无上拉，无下拉

GPIOA->BSRRH = 0x00ff; //复位GPIOA\_BSRRH寄存器

GPIOA->BSRRL = 0x00AA;

}

}

### 3.2.2 android客户端程序设计

（1）控制界面的布局

控制界面主要运用了线性布局、相对布局和表格布局。整体采用线性布局，局部采用相对布局，而控制按钮采用了表格布局。控制界面的布局如图3.6所示：

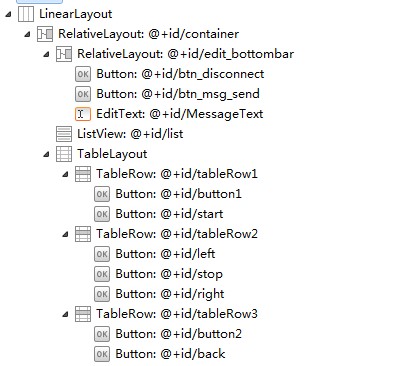


图3.6 控制界面的布局

（2）布局的代码如下：

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<LinearLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"fill\_parent"*

android:orientation=*"vertical"* >

<RelativeLayout

android:id = *"@+id/container"*

android:orientation=*"vertical"*

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"fill\_parent"*

>

<RelativeLayout

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"wrap\_content"*

android:id= *"@+id/edit\_bottombar"*

android:layout\_alignParentBottom = *"true"*>

<Button android:id=*"@+id/btn\_disconnect"*

android:layout\_width=*"65dp"*

android:layout\_height=*"wrap\_content"*

android:layout\_alignParentLeft =*"true"*

android:text=*"断开"*/>

<Button android:id=*"@+id/btn\_msg\_send"*

android:layout\_width=*"65dp"*

android:layout\_height=*"wrap\_content"*

android:layout\_alignParentRight =*"true"*

android:text=*"发送"*/>

</RelativeLayout>

<TableLayout

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"* >

<TableRow

android:id=*"@+id/tableRow1"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"* >

<Button

android:id=*"@+id/button1"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:text=*"Button"*

android:visibility=*"invisible"* />

<Button

android:id=*"@+id/start"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:layout\_marginRight=*"0dp"*

android:text=*"start"*

android:width=*"120px"* />

</TableRow>

android客户端的界面如图3.7所示：



图3.7 android控制界面

（3）发送按钮的代码

sendButton= (Button)findViewById(R.id.*btn\_msg\_send*);

sendButton.setOnClickListener(**new** OnClickListener() {

@Override

**public** **void** onClick(View arg0) {

// **TODO** Auto-generated method stub

String msgText =editMsgView.getText().toString();//获取编辑框内的内容

**if** (msgText.length()>0) {

sendMessageHandle(msgText);//发送编辑框的内容给串口

editMsgView.setText("");//清空编辑框

editMsgView.clearFocus();

//close InputMethodManager

InputMethodManager imm = (InputMethodManager)getSystemService(Context.*INPUT\_METHOD\_SERVICE*);

imm.hideSoftInputFromWindow(editMsgView.getWindowToken(), 0);

}

**else**

Toast.*makeText*(mContext, "发送内容不能为空！", Toast.*LENGTH\_SHORT*).show();

}

});

（4）控制按钮的代码

以左转按钮为例：

sendButton= (Button)findViewById(R.id.*left*);

sendButton.setOnClickListener(**new** OnClickListener() {

@Override

**public** **void** onClick(View arg0) {

String msgText ="1"; // 发送左转命令“l”

**if** (msgText.length()>0) {

sendMessageHandle(msgText);//发送“l”给串口

editMsgView.setText("");//清空编辑框

editMsgView.clearFocus();

//close InputMethodManager

InputMethodManager imm = (InputMethodManager)getSystemService(Context.*INPUT\_METHOD\_SERVICE*);

imm.hideSoftInputFromWindow(editMsgView.getWindowToken(), 0);

}

**else**

Toast.*makeText*(mContext, "发送内容不能为空！", Toast.*LENGTH\_SHORT*).show();

}

});

## 3.3 系统调试

### 3.3.1 系统调试时遇到的问题及解决方案

系统调试时，遇到的问题主要有以下两个问题：

a.驱动电路部分：

加电调试时，无法正常运行，断电细查后，才发现，原来把稳压管方向接反。编写程序调试时，出现2个电机一个能正常转动，但是另外一个电机却不按照程序控制，自行无规律乱动的情况，并且在检查软件程序无误的情况下，还是不能控制小车的右电机正常动作，后来细心的研读程序，一项一项的查找问题所在，最后，检查到，原来是在接线的时候，由于不够仔细，出现短路现象，造成了硬件电路的错误。

b.实际场地调试部分

在无火源的情况下，小车处于巡逻模式，根据前面、左面、右面超声波传感器反馈给单片机的距离进行比较进而控制小车的前进、左转、右转以及后退。小车能成功在房间进行巡逻，如果遇到障碍物小车会根据距离的比较避开障碍物。试验中，我们分别把脚和木板当做障碍物，有时小车前面的超声波检测距离反应时间比较长会撞到障碍物，所以小车具有后退功能，后退之后比较左右距离的大小自行调整行走方向。

　　一旦发现火源小车立即切换成灭火模式，小车将装在车前的火焰传感器的输出模拟电压值反馈给单片机，单片机进行a/d转换后比较6路输出电压值的最小值使小车朝着火焰传感器输出的最小模拟值的方向走。同时，小车通过前面的超声波检测小车与火源的距离使小车准确地停车并打开风扇灭火。灭火之后小车自动切换成巡逻模式。

### 3.3.2 调试心得

最好把驱动电机系统设计成可以承受智能技术预计最终重量的两倍，这样增加模块不会太影响控制性能。黑线用作地线，红色用作电源（VCC)，白线用作马达馈线等等养成习惯, 如果不管什么颜色的电线都随手抓来，那么调试起来就会很困难。硬件和软件开发应该同步进行，它们实际上只是同一个问题的两个不同面。先做硬件并进行硬件调试，然后依据设计要求和硬件来编写程序并进行软件调试。

小车第一次在场地跑时，由于电池电量不足没能成功，当时使用的是若干节干电池串联组成的电池组构成的电源，后来改为3.7V锂电池板串联而成的电池组，为整个电路供电。但是小车的电机承受尖峰电流的能力太弱，所以烧坏了几个电机，只能降低PWM占空比，然后再进行调试。

在正式跑赛道时，车底盘上的传感器灵敏度太高，刚刚从出发点出发就显示检测到三根白线，开始进行灭火动作，后来经过调节传感器距离地面的高度，使得ST168只能够检测到纯白色的部分，其余灰色或非纯白色的部分LM324的输出电平都不会发生变化，经过调整后小车能够完成基本功能。同时诺基亚5110能够实时显示小车当下执行的操作。

### 3.3.3 系统调试的结果

经过系统调试以后，不断地改善程序，基本上实现了总体设计中所要求的功能。系统调试的最终结果如图3-8所示：



图3-8灭火智能小车实物效果图

# 第4章 系统创新与结果分析

## 4.1 系统创新

传统的遥控车模型功能单一，只能实现前进后退等基本功能，我们研究组设计实现了可以自动运行也可人为干预的遥控+自动的小车模型，并且加入了寻找和熄灭火源的有实际应用意义的功能。我们设计的创新点主要有以下几点：

a.我们自行开发了一个基于Android和基于C#的手机和电脑客户端，在小车上安装接收蓝牙信号的HC-05蓝牙接收模块，然后就可以通过客户端发送蓝牙信号，来对蓝牙小车进行控制控制，其接收可达15米，完全能适应对小车的要求。其中这个HC-05蓝牙接收模块是低耗能，这样就把更多的能量用在小车的驱动上。

b.小车需要很大的马力和很好的灵活性以应对不同的地形。这辆车的车轮使用四驱的直流电机来驱动的，用PWM波来控制小车的速度，可以很方便的更改其速度，有主控板通过推挽输出来控制电机的翻转以让车子进行后退的速度。这样就可以胜任小车的行驶要求。

c.为了让小车在人眼看不到的地方也能自由行驶，我们加入了红外光电开关管，设计了小车循墙自动行驶功能。

d.同时我们安装了火焰传感器模块和舵机传动模块，操作人员不需要进行操作，小车就能自动识别火焰方向，并通过舵机转到指定方向，打开风扇进行精确灭火。

## 4.2 评测与结论

首先，给电源模块上12v的电源，然后打开电机驱动模块开关，同时将主控板的供电端连接到电源管理模块。然后，在android手机上安装“蓝牙通信”应用程序后，打开该APP，然后选择“允许打开蓝牙”。点击设备列表中的“开始搜索按钮”，在设备列表中选择蓝牙模块的名字进行连接。

完成上述工作以后，就可以在手机上通过按下“前进”、“左转”、“后退”、“右转”、“后退”通过蓝牙给小车发送“前进”、“左转”、“停止”、“右转”、“后退”，“灭火”，“加速”，“减速”，“自动模式”，“遥控模式”9个命令。小车可以解析命令做出相应动作。

# 第5章 结论

小车的特点能实现自动寻路找到火源，但由于火焰传感器的检测角度为60度，超声波测距模块的性能不是很好，所以有时不能在场地准确确定火源的位置，比如直线驶向火源，再加上超声波测距存在一定偏差并且反映给单片机都一定延时，导致小车有时不能在前正方停下并熄灭火源，而是先撞一下再避开然后启动风扇灭火。为解决这一问题，目前是在左前右三面装上了光电开关传感器，解决了上述问题。

利用软件不断调试，减小小车直行速度，加上延时，并增加小车停车时距离火源的距离。为了更精确找到火源的位置，考虑再加个烟雾传感器或者多个红外火焰传感器排列成一定角度，实现对场地的所有方位巡逻。除此之外，还有小车转弯速度过高，转弯幅度太大而造成原地打转，或者是小车在碰到墙面时不能及时避开造成偶尔的卡顿现象，通过更加细化转弯的逻辑关系小车的行走路线的确改善了不少，未来在控制机器小车精确转弯时可能利用相关硬件器件进行控制比如指南针，或者采用好的算法比如pid。我们通过不断调试，解决了小车会卡在墙壁等问题。从最终综合测试结果来看，本系统能够很好的完成题目的设计要求，并结合我们自己的发散思维，进行了特色创新。

完成上述工作的同时，就可以在手机上通过按下“前进”、“左转”、“后退”、“右转”、“后退”通过蓝牙给小车发送“前进”、“左转”、“停止”、“右转”、“后退”，“灭火”，“加速”，“减速”，“自动模式”，“遥控模式”9个命令。小车可以完美解析上述命令做出相应动作。

# 心得体会与感悟

时光荏苒，二年的历程就这样在今天划上了句号。这两年中，我们整个大创小组都全身心的投入到这个大创项目中，与导师之间的联系，同学之间激烈的讨论，相关资料的搜索，最终都以面前的这个灭火机器人呈现给各位。两年的时间说长不长说短不短。刚开始的时候，我们的大创小组并没有什么具体的目标，因为当时选择的大创项目并不是灭火机器人，在经历过几个月的思考和讨论后，并且和导师进行一些必要的沟通之后，我们决定将这个大创项目修改为灭火机器人。说实话在刚开始的时候感觉灭火机器人这个方面的大创项目挺有意义的，因为当时就在想新闻上很多消防官兵在进行消防工作时很多方面的因素导致身体甚至生命受到威胁，而且当今社会人工智能这么火的情况下，感觉如果能以机器人的形式去进行相应的消防工作，就会很切合人工智能这个主题，所以灭火机器人成为了我们这个大创项目的最终目的。当时定下这个大创项目时，大家心里都觉得坦然了很多，因为我们有了共同的目标，就是面前的这台微型灭火机器人。

我们先是寻找了一些相关的论文，证明了这个灭火机器人的可行性。因为我们这个大创小组里面有做过飞思卡尔智能车的同学，所以关于这些四轮车的行进以及一些避障，PCB板的制作等功能都可以实现。然后就是对火源的感知，以及灭火器的摆动，同时可以实现通过手机端对其进行控制，达到远距离操控进行工作的目的。在进行整体的这些部分的构想后，然后就利用假期的时间开始灭火机器人的制作。

毕竟理想和现实总是有偏差的，所以说走过的这两年可以说是不断的修正理想和现实之间的误差。因为我们在刚开始PCB板的制作过程中就遇到了很多的麻烦，虽然与飞思卡尔的智能车大同小异，但是考虑到各种传感器以及电机的影响使PCB板布线就产生很大的麻烦。经过好多次的设计，使PCB板的布线趋近合理。解决了这个难题之后，然后开始了板子的焊接以及车模的组装。然后对于这个灭火原理的实现的，最终是选择了让人难以接受的吹风的方式，因为在选择的过程中，发现液体二氧化碳和干粉灭火的方式难以实现，而且原料的填充和试验现场的限制，最终还是选择了风力灭火的方式。以为风力的大小以及方向都可以通过程序得到实现，实现起来更加的方便。当然也是考虑大创项目的最终结果是在于最终灭火机器人的模型，对于灭火的方式以后可以稍加完善。

这次大学生创新创业让我们有很多感慨，毕竟天外有天人外有人，而且对于自身的知识储备方面也是不够的，过程中有额外差了很多的资料，但是最终我们能将这两年的结晶以灭火机器人这样一种形式呈现给大家也让我们心情很轻松。大学生活的两年很珍贵，最终的大创结题也仅此一次，这份经历很伴随我们很久很久，也让我们掌握了很多的技能。

谢谢导师的陪伴，还有各位评委老师的付出！

# 参考文献

[1] 蔡莉莎, 韩宝如. [传感器技术及应用](http://211.87.177.4/opac/openlink.php?title=%E4%BC%A0%E6%84%9F%E5%99%A8%E6%8A%80%E6%9C%AF%E5%8F%8A%E5%BA%94%E7%94%A8)[M]. 北京:中国电力出版社,2016

[2]Joseph Yiu.The Dedinitive Guide to the ARM Cortex-M3[S],2007

[3] Michael A. Miller. Data and Network Communications[M].第一版，科学出版社，2002.

[4] 郭天祥. 新概念51单片机C语言教程:入门、提高、开发、拓展[M]. 北京:电子工业出版社，2009

[5]  刘火良，杨森. [单片机](http://baike.baidu.com/item/%E5%8D%95%E7%89%87%E6%9C%BA)与嵌入式:STM32库开发实战指南[M]. 北京:机械工业出版社，2013.5

[6] 初永丽, 王雪琪, 范丽杰. [模拟电子技术基础](http://211.87.177.4/opac/openlink.php?title=%E6%A8%A1%E6%8B%9F%E7%94%B5%E5%AD%90%E6%8A%80%E6%9C%AF%E5%9F%BA%E7%A1%80)[M]. 西安:西安电子科技大学出版社,2016

[7]  李志明，檀永，徐石明，丁孝华，桑林. STM32嵌入式系统开发实战指南[M]. 北京:机械工业出版社，2013.5

[8]  (美国)[霍顿](http://baike.baidu.com/item/%E9%9C%8D%E9%A1%BF)(Ivor Horton). C语言入门经典. 北京:清华大学出版社[M]，2008.4.1

[9] 黄智伟, 黄国玉. [Altium Designer原理图与PCB设计](http://211.87.177.4/opac/item.php?marc_no=0000508618)[M], 北京:人民邮电出版社,2016

[10]解璞, 胡仁. [Altium Designer 16从入门到精通](http://211.87.177.4/opac/openlink.php?title=Altium+Designer+16%E4%BB%8E%E5%85%A5%E9%97%A8%E5%88%B0%E7%B2%BE%E9%80%9A)[M]. 北京:机械工业出版社, 2016

# 附 录

## 附录一

软件源代码：

int main(void)

{

Stm32\_Clock\_Init(9); //ÏµÍ³Ê±ÖÓÉèÖÃ

uart\_init(72,19200); //´®¿Ú³õÊ¼»¯Îª19200

delay\_init(72); //ÑÓÊ±³õÊ¼»¯

init();

while(1)

{

if(F1||F2||F4||F5||F3)

{

JDQ=1;

while(F1||F2||F4||F5||F3)

{

Turn\_Fan();

}

JDQ=0;

}

Servo\_PWM=Servo\_Mid;

Get\_Control();

}

}

void Get\_Control()

{

u16 t;

u16 len;

u16 times=0;

if(USART\_RX\_STA&0x8000)

{

len=USART\_RX\_STA&0x3FFF;//µÃµ½´Ë´Î½ÓÊÕµ½µÄÊý¾Ý³¤¶È

for(t=0;t<len;t++)

{

USART3->DR=USART\_RX\_BUF[t];

while((USART3->SR&0X40)==0);//µÈ´ý·¢ËÍ½áÊø

}

USART\_RX\_STA=0;

}

Mobile\_Con(USART\_RX\_BUF);

if(Mobile\_Con\_Flag)

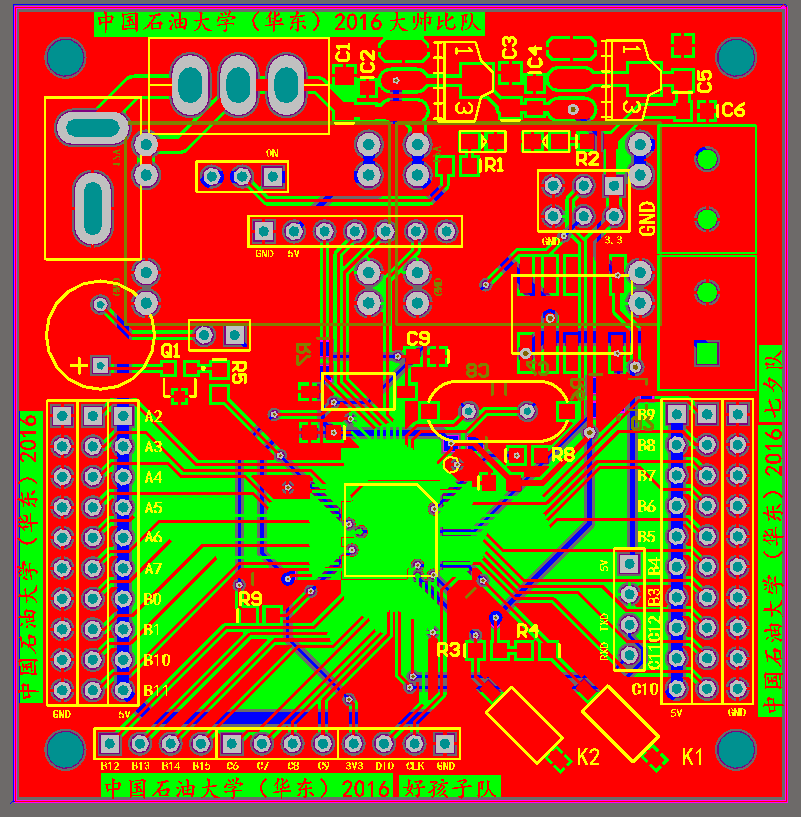
{

Auto();

}

}

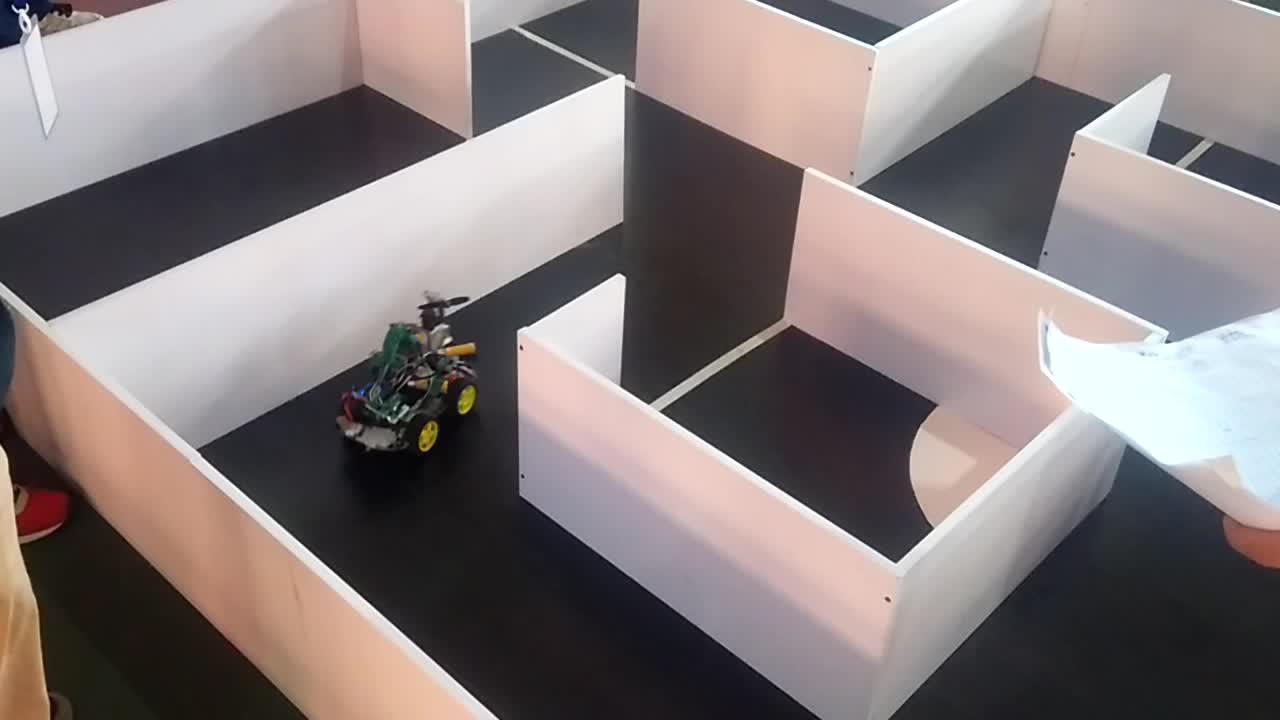
## 附录二



PCB效果图

## 

## 附录三



系统设计实物图