西安郵電學院
XI'AN INSTITUTE OF POST & TELECOMMUNICATIONS

题目:灭火机器人小车制作

目　录

第一章　引言

1.1 课题背景

随着自动控制技术的高速发展，单片机应用日益广泛！其已广泛的渗透到工业自动化、仪器仪表、军事装备、广告设计等各类产业以及人们日常生活的方方面面。目前，基于单片机的自动控制系统已广泛应用于机械、电子、石油、化工、家用电器等轻工业领域。单片机体系结构中支持运用 C 语言进行编程，而且单片机 C 语言编译器提供了良好的程序与单片机之间的配合，避免了汇编语言的冗长，大大缩短的程序代码的长度和有效的优化了程序。灭火机器人是一个集信号检测、传输、处理和控制于一体的控制系统，代表了智能机器人系统的发展方向。

1.2 实现功能

制造一个计算机控制的机器人在一间平面结构房子模型里运动，找到一根蜡烛并尽快将它熄灭，这个工作受多个因素影响，它模拟了现实家庭中机器人处理火警的过程,那个蜡烛代表家里燃起的火源，机器人必须找到并熄灭它。设计一个自主控制的机器人，在一套平面结构的模拟的房子里运动，

1.3 模拟房子介绍

模拟房子平面图单位：mm

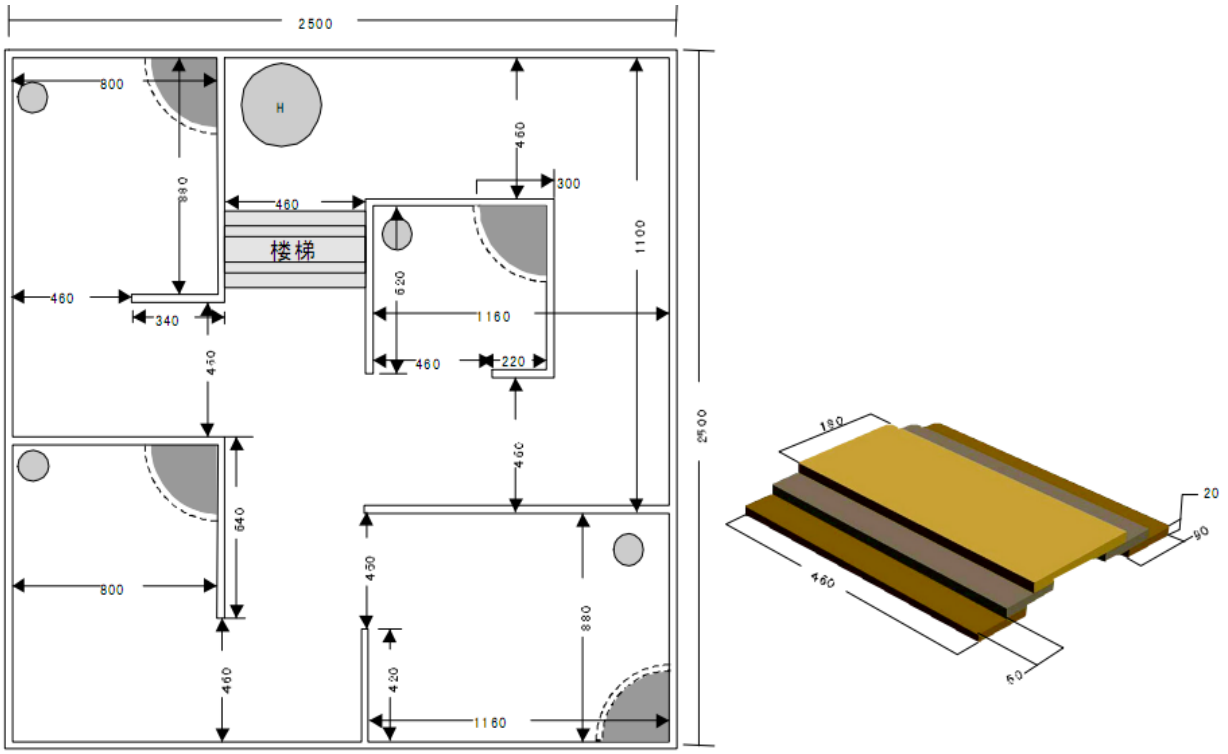


图 1-1 灭火机器人比赛场地（国际赛制）

比赛场地的墙壁 33cm 高，由木头做成。墙壁刷成白色。比赛场地的地板将是被漆成

黑色的光滑木制表面。在所有的房间和走廊的地板上，可能会铺有小地毯，不会有粗毛地毯。场地中所有的走廊和门口宽都是 46cm。门口并没有门，而是一个 46cm 的开口，将会有一个白色的 2.5cm 宽的白色带子或白漆印迹表示房间入口。

第二章 硬件设计基础知识

2.1 MSP430F149 单片机相关知识

2.1.1 MSP430F149 单片机概述

MSP430F149 是美国 TI 公司生产的 MSP430 系列超低功耗微控制器中的一种。

MSP430F149 的 CPU 结构如图 1-2 所示，主要具有以下功能模块：基础时钟、看门狗定时器、Timer_A、Timer_B、6 个 8 位并行端口（其中 P1 和 P2 具有中断功能）、模拟比较器 Comp_A、12 位 A/D 转换器、2 通道串行通信接口（通过软件选择 UART/SPI 模式）、1 个硬件乘法器、60KB 的 Flash 以及 2KB 的 RAM。

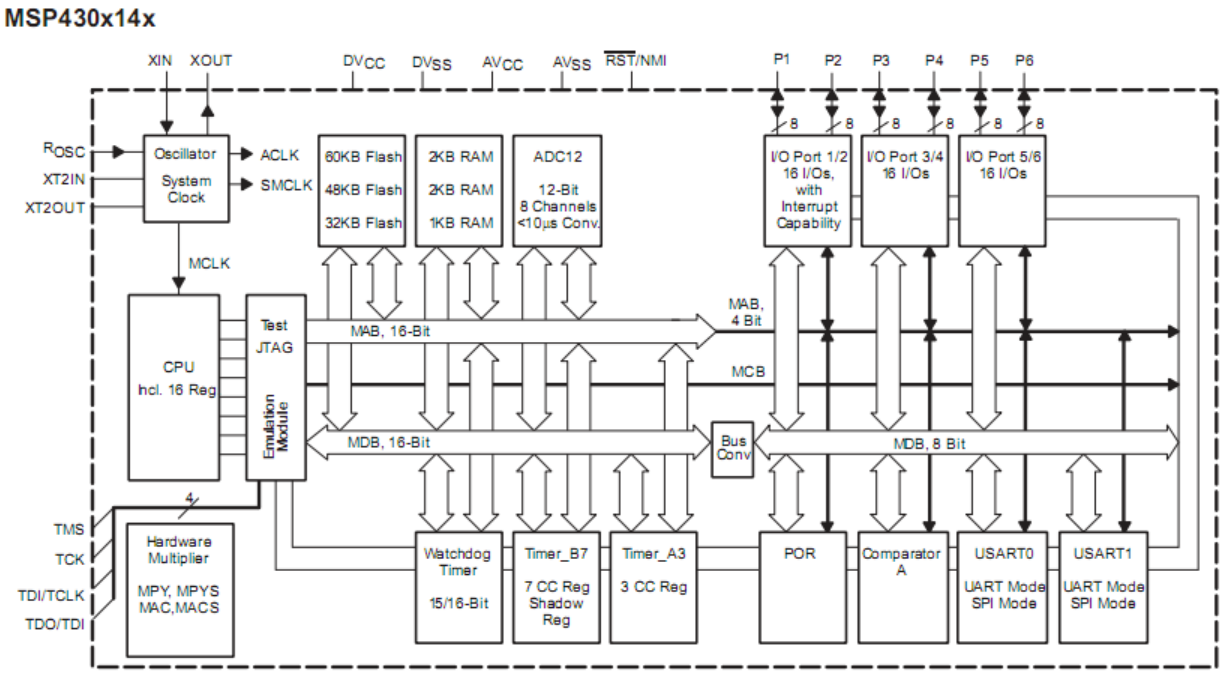


图 2-1　MSP430F149 的 CPU 结构

2.1.2 MSP430F149 单片机特点

- 电源电压范围：1.8V---3.6V;
- 超低功耗
- 待机模式：1.6uA
- 关闭模式（RAM 保持）：0.1uA；
- 活动模式：280uA@1MHz，2.2V；
- 五种省电模式；

- 快速唤醒模式（6us 内从待机模式唤醒）；
- 16 位 RISC 结构；
- 内部集成看门狗定时器防止程序跑飞；
- 快速的数据处理能力内部自带硬件乘法器；
- 高处理速度，指令周期 125ns（8M 工作模式）
- 多个 I/O 支持中断模式满足系统对外部中断的需求；
- 通用串口支持 SPI、SCI 模式；
- 带内部参考，采样保持和自动扫描特性的 12 位 A/D 转换器；
- 有 3 个捕捉/比较寄存器的 16 位定时器 Time_A 支持 PWM 和 CAP 功能；
- 有 3 个捕捉/比较寄存器的 16 位定时器 Time_B 支持 PWM 和 CAP 功能；
- 片内集成模拟比较器；
- 片上集成 60KB 的 Flash 和 2KB 的 RAM 同时提供 256 字节的信息 Flash；
- 串行在线编程，无需外部编程电压，安全熔丝可编程代码保护单片机；
- 可用封装：64 脚方形扁平封装（QFP）。

2.1.3 MSP430F149 单片机引脚图

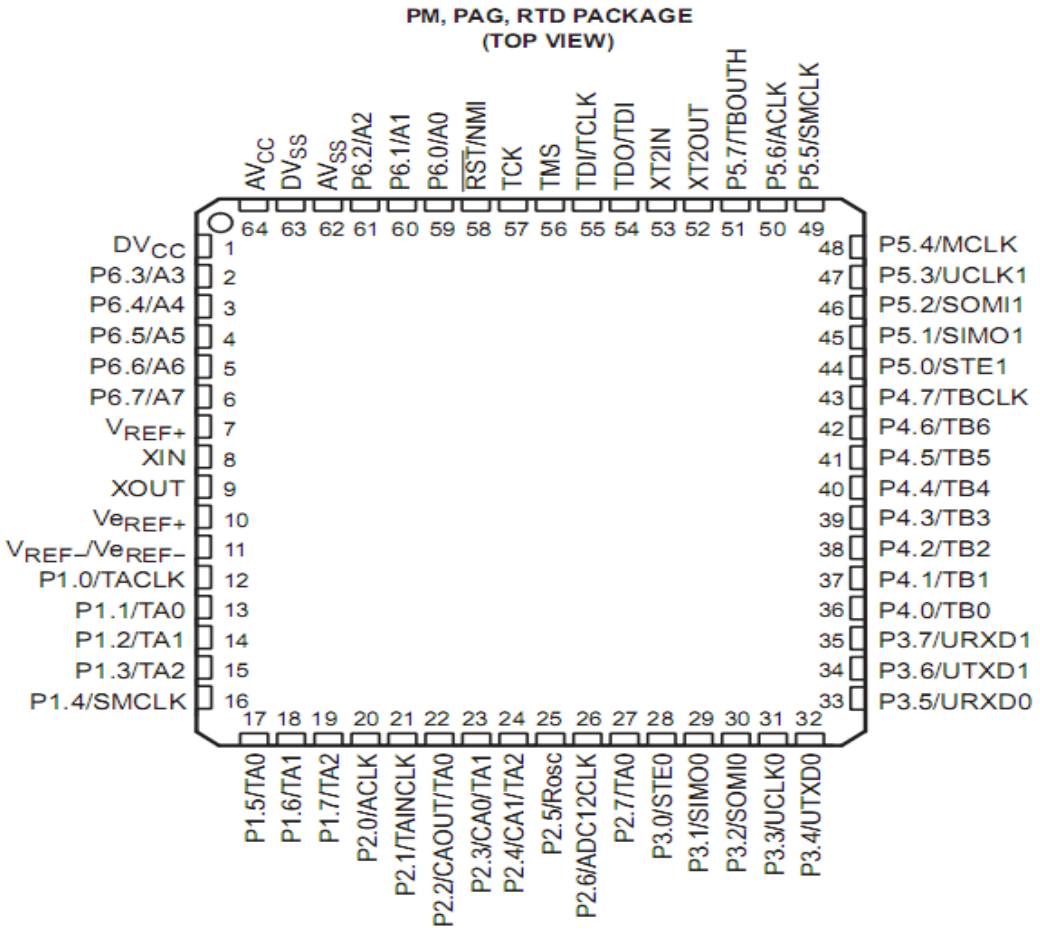


图 2-2　MSP430F149 的 CPU 结构

引脚功能说明：（MSP430F149 单片机的 I/O 口都具有第二功能）

- DVcc：数字电源正端；
- AVcc：模拟电源正端（使用内部 ADC 时应与 Dvcc 隔离以提高 ADC 采样精度）；
- DVss：数字电源负端；
- AVss：模拟电源负端；
- V_{REF+}：ADC 内部参考电压正端；
- Ve_{REF+}：ADC 外部参考电压正端；
- V_{REF-}/Ve_{REF-}：ADC 参考电压负端；
- RST：复位输入，非屏蔽中断输入端口，引导装载程序启动（Flash 器件）低电平产生复位信号；
- TDO/TDI：测试数据输出端口，编程数据输入端口；
- TDI/CLK：测试数据输入或测试时钟输入，器件保护熔丝连接到该引脚；
- TMS：选择测试模式，用作一个器件编程和测试输入端口；
- TCK：用作器件编程测试或音带装载程序启动时钟输入端口；
- XIN：晶体振荡器 XT1 的输入端口（标准 32768Hz）；
- XOUT：晶体振荡器 XT1 的输出端口；
- XT2IN：晶体振荡器 XT2 的输入端口（标准 8MHz）；
- XT2OUT：晶体振荡器 XT2 的输出端口；
- P1 口：Timer_A；捕获；比较；TACLK 和 SMCLK 信号输入；
- P2 口：Timer_A；捕获；比较；ACLK 信号输入；
- P3 口：USART/UART/SPI 方式；
- P4 口：Timer_B；捕获；比较；
- P5 口：USART/UART/SPI 方式；MCLK、SMCLK、ACLK 输出；
- P6 口：12 位 ADC 模拟输入；

2.2 撞检测传感器 HS0038

2.2.1 HS0038 简介：

HS0038B -系列微型接收机红外遥控器控制系统。 PIN 二极管和前置上组装引线框架，环氧包被设计成红外过滤器。该解调输出信号可直接解码的微处理器。HS0038B 是标准的红外遥控接收器系列，支持所有主要传输代码。

2.2.2 HS0038 特点：

1、光检测器和放大器一体封装

2、内部可集成 PCM 频率过滤器

3、与 TTL 和 CMOS 电平兼容

4、改进的屏蔽电场，抗干扰能力强

2.2.3 HS0038 外形：

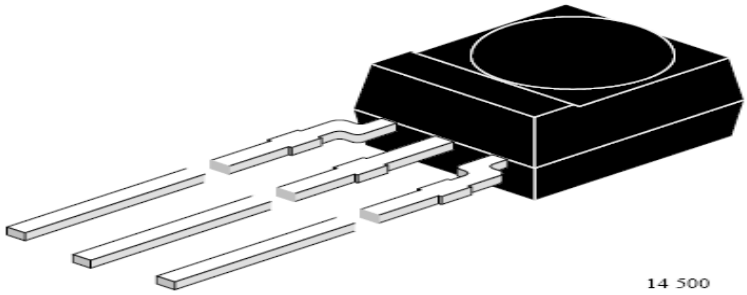


图 2-3　HS0038 实物图

2.2.4 HS0038 与单片机连接原理图：

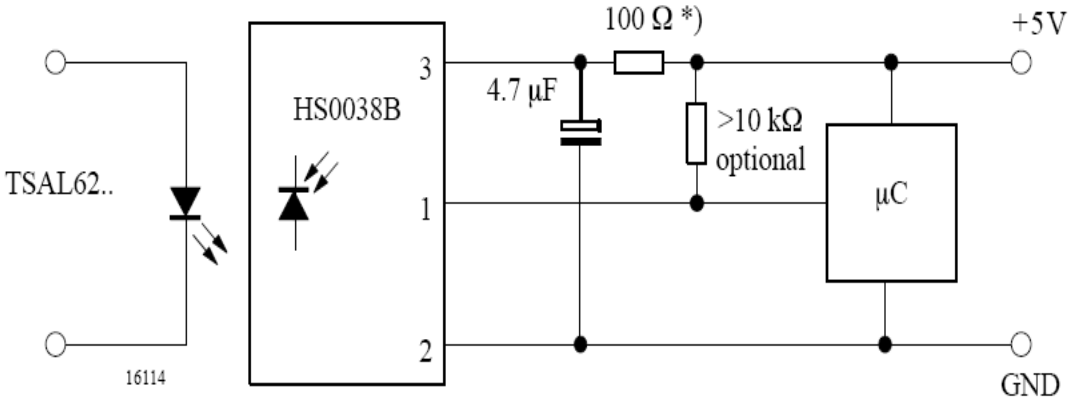


图 2-4　HS0038 电路

2.2.5 检测原理：

红外发射管发射出经过调制过的 38KHZ 的红外光，当前方没有障碍物时，接收器收不到红外光，相反当前方有障碍物时，接受器可以收到红外光。根据此原理，机器人可以感知前方的路况从而决定是否前行。

2.3 地面灰度检测传感器 ST188

2.3.1 ST188 特点：

1、采用高发射功率红外二极管和高灵敏度光电晶体管组成。

2、检测距离可调整范围大，4--13 mm 可用。

3、采用非接触方式。

2.3.2 应用范围：

1、IC 卡电度表脉冲数据采样。

2、集中抄表系统数据采集。

3、传真机纸张检测。

4、地面灰度检测，正反转速测量、行程测量等。

2.3.3 外形尺寸（单位mm）：

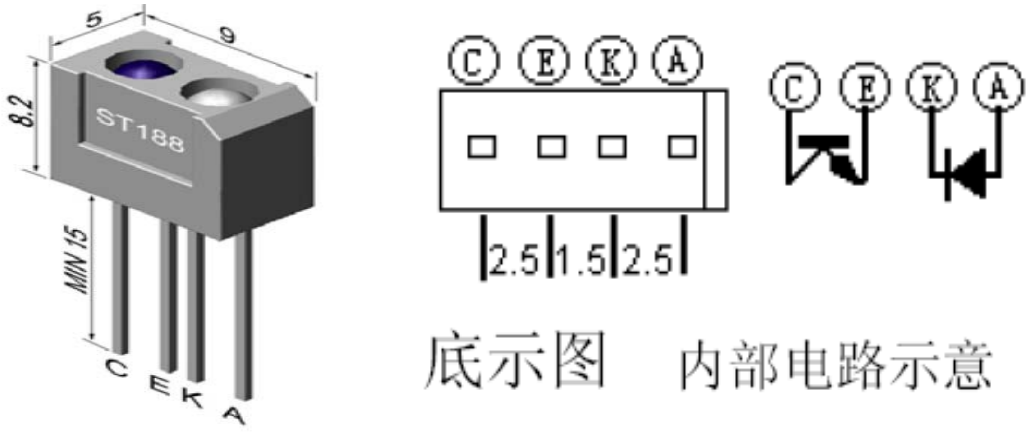


图 2-5　ST188 实物图

2.3.4 ST188 与单片机连接原理图：

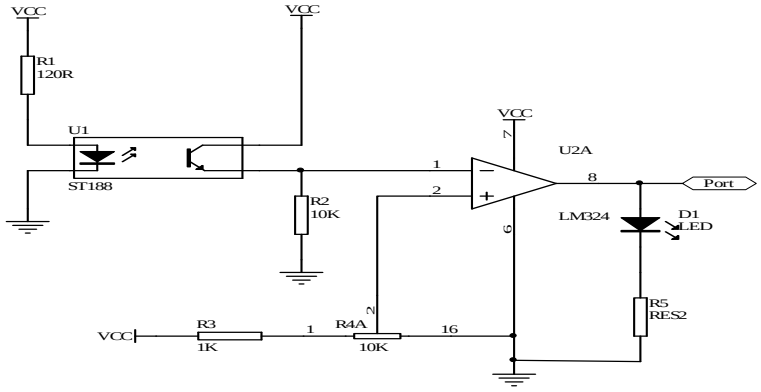


图 2-6　ST188 电路图

2.3.5 检测原理：

ST188 是红外收发一体的器件，发射管发射出红外光线，接收管就可以根据接收的红外光线的强弱，感知地面的灰度。由于此模拟房间的地面被处理成为黑白两种颜色，通过比较器设置灰度的门限值，可以很方便的感知地面的颜色，从而做出相应的决策。

2.4 左右碰撞检测传感器 SHARP GP2D12

2.4.1 选择GP2D12 的原因：

1、由于机器人在活动中（不包含工业机器人）这个传感器最常用，几乎每家国外的机器人配件供应商都提供。使用 Google 英文版搜索一下“MiniSumo”，你将会发现 GP2D12 使用是多么普遍。

2、它的测距范围和小车的“个头”及运动速度匹配，对于 10cm 见方、运动速度

10—30cm/s 的小个头，能“看到”几米开外的东西的意义不大，而 10—80cm 正是它所要关注的范围。

2.4.2 SHARP GP2D12 的主要技术参数：

1、Range 范围：10 to 80cm

2、Update frequency /period 刷新频率/周期：25Hz/40ms

3、Direction of the measured distance 测距方向性：Very directional due to the IR LED

4、Max admissible angle on flat surface最大允许角度：>40^0

5、Power supply voltage 电源电压：4.5---5.5V

6、Noise on the analog output 模拟输出噪声：<200mV

7、Mean consumption 平均功耗：35mA

8、Peak consumption 峰值功耗：about 200 mA

2.4.3 外形：

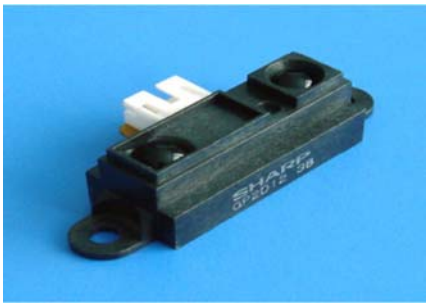


图 2-7 SHARP GP2D12 实物图

2.4.4 工作原理：

它是由一个红外发射管和一个 PSD(Position Sensing Device 位置敏感检测装置)以及相应的计算电路构成，Sharp 公司的 PSD 很有特色，它可以检测到光点落到它上面的微小位移，分辨率达微米，GP2D12 正是利用这个特性实现了几何方式测距。

红外发射管发射的光束，遇到障碍物反射回来，落在 PSD 上，构成一个三角形，借助于 PSD 可以测量得到三角形的底，而两个底角是固定的，由发射管确定，此时便可通过底边推算出高，也就是我们所要的距离，如下图所示：

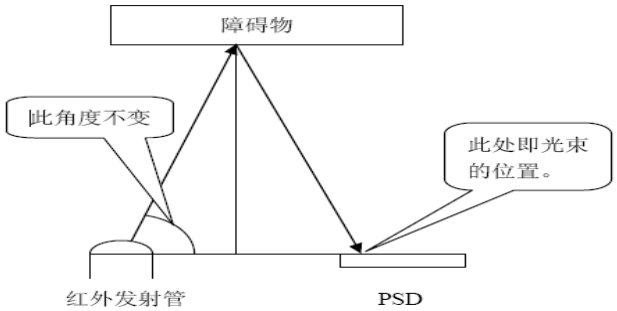


图 2-8 SHARP GP2D12 工作原理

从图中可以看出，这是一个顶角极锐的等腰三角形，底边只有 2cm，高却要有 10-80cm，所以 PSD 的分辨率必须极高，否侧微小的偏差都会带来距离的巨大误差。从这一点也可以看出，它的测距离结果很难稳定，精确，毕竟比值太大。

因为 PSD 的尺寸有限，从图中很容易理解为何他的测量距离超出范围后就不可能是有效数据，连趋势都得不到。

从上述原理描述可以知道，它不是连续测量，得到底边长度后，必须经过计算才能得到距离值，然后转换为模拟信号输出。

2.4.5 GP2D12 与单片机连接：

GP2D12 直接接到单片机的 AD 口，软件上需要解决两个问题：一是信号的线性化，因为输出与距离的关系是非线性的，为便于程序中使用距离信息，必须将模拟信号转换为相应的距离值。二是滤波，因为按照上述原理的测量分析，GP2D12 的输出噪声很大；此外，还由于测量的非连续性，导致连续的距离变化对应的输出为阶跃信号，也需要通过滤波将其平滑。

2.5 电机驱动芯片 L298N

2.5.1 L298N简介：

L298N 是 SGS 公司的产品，内部包含 4 通道逻辑驱动电路。是一种二相和四相电机的专用驱动器，即内含二个 H 桥的高电压大电流双全桥式驱动器，接收标准 TTL 逻辑电平信号，可驱动 46V、2A 以下的电机。其引脚排列如图 1 中 U4 所示，1 脚和 15 脚可单独引出连接电流采样电阻器，形成电流传感信号。L298 可驱动 2 个电机，OUT1、OUT2 和 OUT3、OUT4 之间分别接 2 个电动机。5、7、10、12 脚接输入控制电平，控制电机的正反转，ENA，ENB 接控制使能端，控制电机的停转。也利用单片机产生 PWM 信号接到 ENA，ENB 端子，对电机的转速进行调节。

2.5.2 L298N的逻辑功能：

表 2-1　SHARP GP2D12 实物图

ENA(B)	IN1(IN3)	IN2(IN4)	电机运行情况
H	H	L	正转
H	L	H	反转
H	同IN2(IN4)	同IN1(IN3)	快速停止
L	X	X	停止

2.5.3 外形：

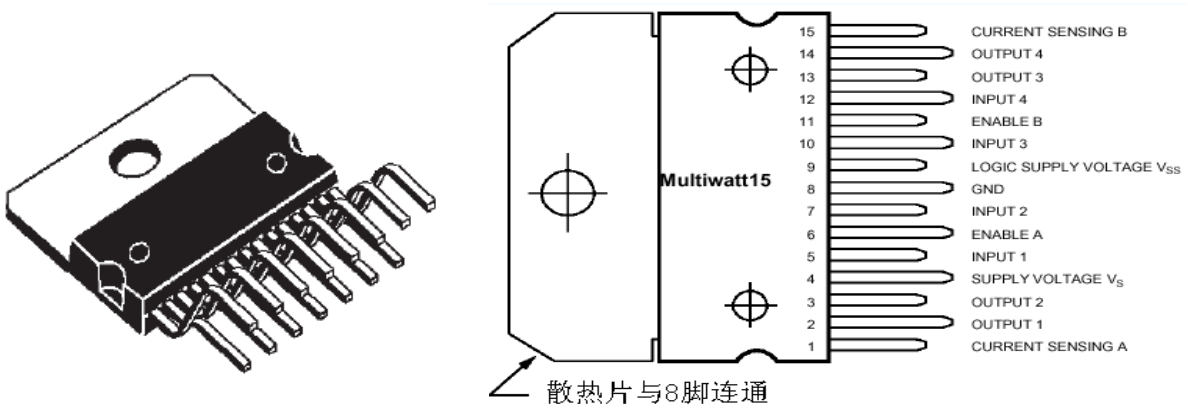


图 2-9　L298N 实物图

2.5.4 L298N与单片机之间的连接：

由于一片 L298N 可以直接驱动两个电机，但是为了加大驱动力，我们采用两路并联的方式来驱动电机。

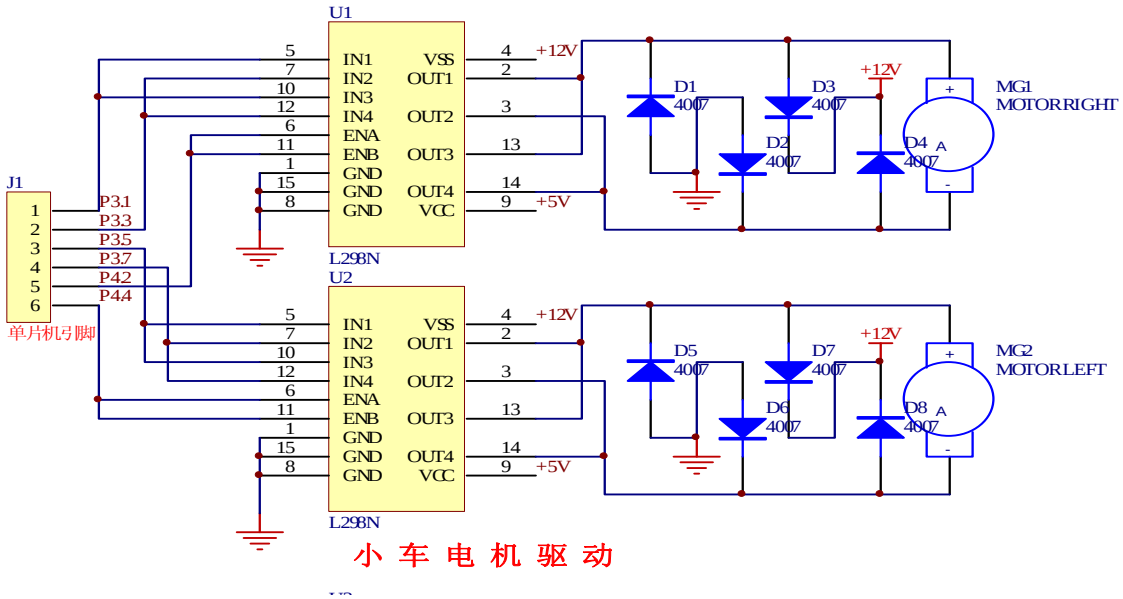


图 2-10　L298N 电路图

2.6 电源管理芯片 LM7805CV、 LM7812CV

电源是任何一个系统稳定运行的前提条件，为了使机器人运行稳定，单片机和电机的供电系统采用独立供电的方法。由于所有的传感器系统供电采用的是 5V 的电源，而车体要良好的运行电机的供电电压应该达到 12V，所以在电源的处理上采用了稳压芯片 7805CV 和 7812CV。

- LM7805CV 的技术指标如下表：

表 2-2　稳压芯片 7805 参数

Electrical Characteristics (LM7805)
(V_I=10V, I_O=500mA, 0°C ≤T_J≤125 °C, unless otherwise specified. (Note 1)

Parameter	Symbol	Conditions	MIN	TYP	MAX	UNIT
Output Voltage	V_O	T_J = 25 °C	4.8	5.0	5.2	V
Line Regulation	ΔV_O	V_I = 7V to 25V T_J = 25 °C		3	100	mV
		V_I = 8V to 12V T_J = 25 °C		1	50	
Load Regulation	ΔV_O	I_O = 5mA to 1.5A, 25 °C		15	100	mV
		I_O = 250mA to 750mA, 25 °C		5	50	
Ripple Rejection	RR	V_I = 8V to 18V, f=120Hz	62	78		dB
Output Noise Voltage	V_N	F= 10Hz to 100Hz T_J = 25 °C		40		μV
Dropout Voltage	V_D	T_J = 25 °C		2.0		V
Quiescent Current		T_J = 25 °C		4.2	8	mA
Quiescent Current Change	ΔI_Q	V_I = 7V to 25V, T_J = 25 °C			1.3	mA
		I_O = 5mA to 1A, T_J = 25 °C			0.5	

- LM7812CV 的技术指标如下表：

表 2-3　稳压芯片 7812 参数

Electrical Characteristics (LM7812)
(V_I=19V, I_O=500mA, 0°C ≤T_J≤125 °C, unless otherwise specified. (Note 1)

Parameter	Symbol	Conditions	MIN	TYP	MAX	UNIT
Output Voltage	V_O	T_J = 25 °C	11.50	12	12.5	V
Line Regulation	ΔV_O	V_I = 14.5V to 30V T_J = 25 °C		10	240	mV
		V_I = 16V to 22V T_J = 25 °C		3.0	120	
Load Regulation	ΔV_O	I_O = 5mA to 1.5A, 25 °C		12	240	mV
		I_O = 250mA to 750mA, 25 °C		4	120	
Ripple Rejection	RR	V_I = 15V to 25V, f=120Hz	55	71		dB
Output Noise Voltage	V_N	F= 10Hz to 100Hz TJ = 25 °C		75		μV
Dropout Voltage	V_D	T_J = 25 °C		2.0		V
Quiescent Current		T_J = 25 °C		4.3	8.0	mA
Quiescent Current Change	ΔI_Q	V_I = 14.5V to 30V, T_J = 25 °C			1.0	mA
		I_O = 5mA to 1A, T_J = 25 °C			0.5	

2.7 LM358 运算放大器

2.7.1　功能特性概述

LM358 内部包括有两个独立的、高增益、内部频率补偿的双运算放大器，适合于电源电压范围很宽的单电源使用，也适用于双电源工作模式，在推荐的工作条件下，电源电流与电源电压无关。它的使用范围包括传感放大器、直流增益模块和其他所有可用单电源供电的使用运算放大器的场合。

2.7.2　引脚功能说明

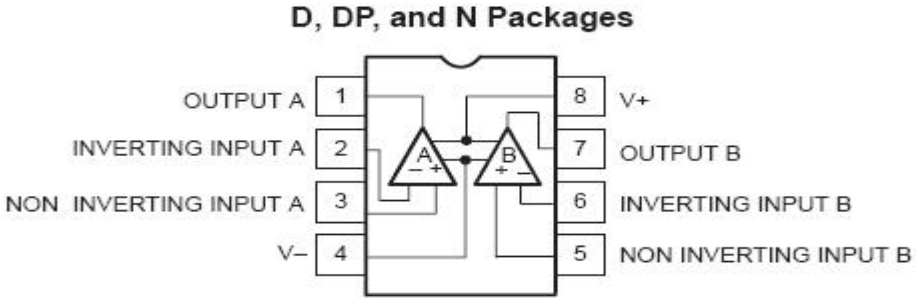


图 2-11　LM358 运算放大器管脚图

Vin-	Vin+		Vout
a	b	a>b	H
a	b	a<b	L

表 2-4　LM358 用作比较器时工作特性

2.7.3　电路图

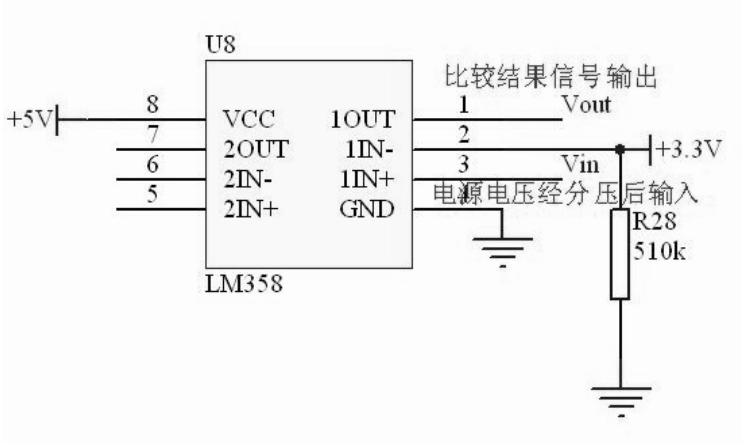


图 2-12　电压比较电路

2.8　火焰传感器

2.8.1 火焰传感器实物图

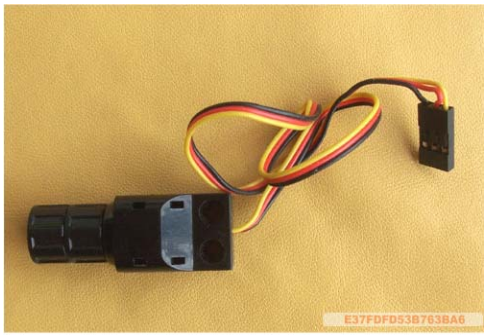


图 2-13　火焰传感器实物图

2.8.2 火焰传感器功能

此传感器本品可广泛应用于灭火机器人比赛中测量火焰值、足球比赛时，用于确定足球的方向。

2.8.3 火焰传感器使用

此传感器具有优良的火焰探测性能，可根据可见光、红外光强弱变化输出电平的大小。其输出端口是一个四针的插头，其中黑色线为地线、红色线为电源线（+5V）、黄色线为信号线，用于输出测量的红外光强度电平、棕色线为信号线，用于输出可见光强度电平。

第三章 系统设计

4.1 系统总体设计

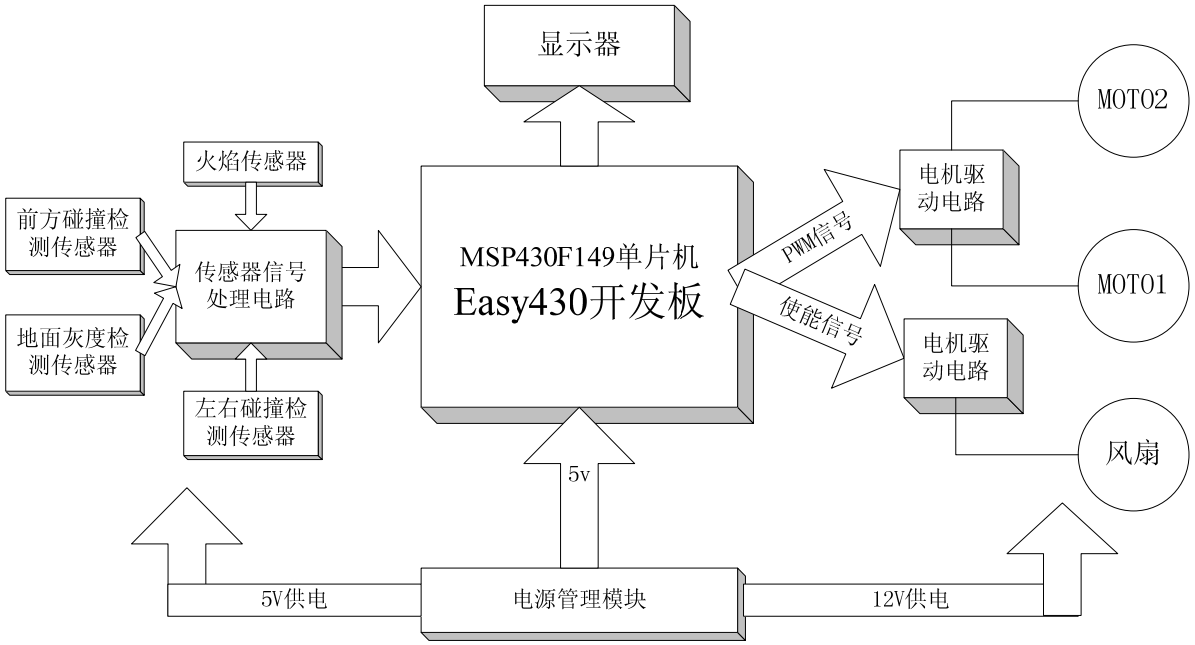


图 3-1 系统总体设计框图

本次设计的目的是设计一个在规定区域能自主搜索火源并实施灭火的智能机器人小车，本次设计使用的主控芯片使用了 EASY430 开发板上的 MSP430F149 单片机，所以设计重点在传感器和电机驱动上。系统总体设计框图如图 3-1：

4.2 硬件设计

4.2.1 电源设计

这次在电源设计上，使用了双电源独立供电的模式，所有使用 5V 的器件使用 LM7505 稳压芯片的输出；三个电机使用稳压芯片 LM7812 的输出电压。本次电源部分设计如图 3-2：

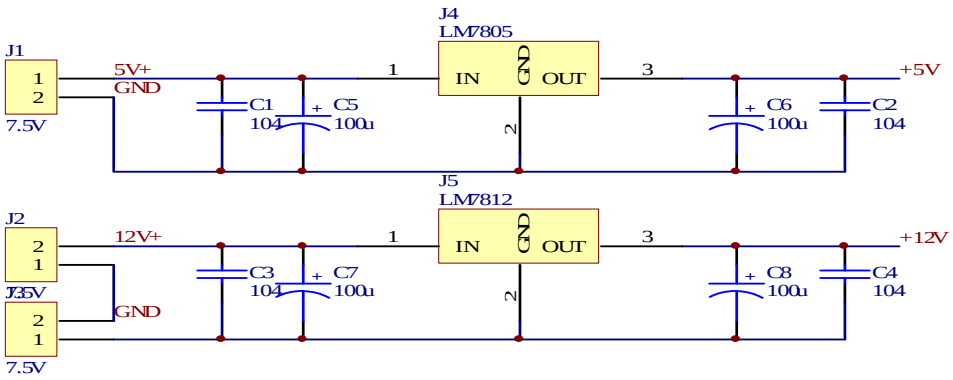


图 3-2 电源部分电路图

4.2.2 电机驱动设计

因为这次设计的是机器人小车，首先得动，而且要快，所以在小车电机的驱动使用上，采用单片 L298N 两路输出并联驱动一个电机；风扇电机用 L298N 的单路输出驱动；设计如图 3-3：

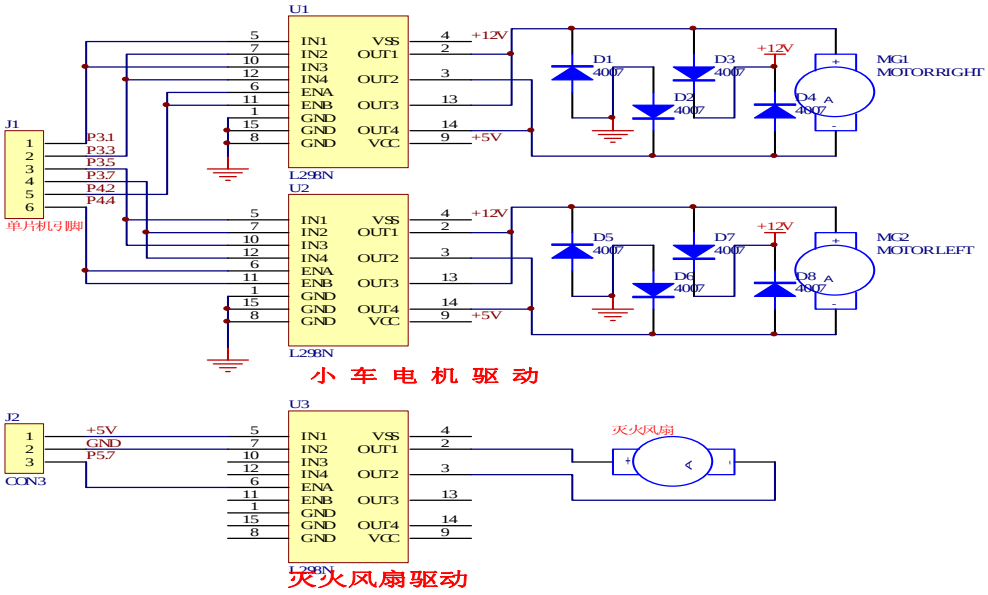


图 3-3 电机驱动部分电路图

4.2.3 传感器接口设计

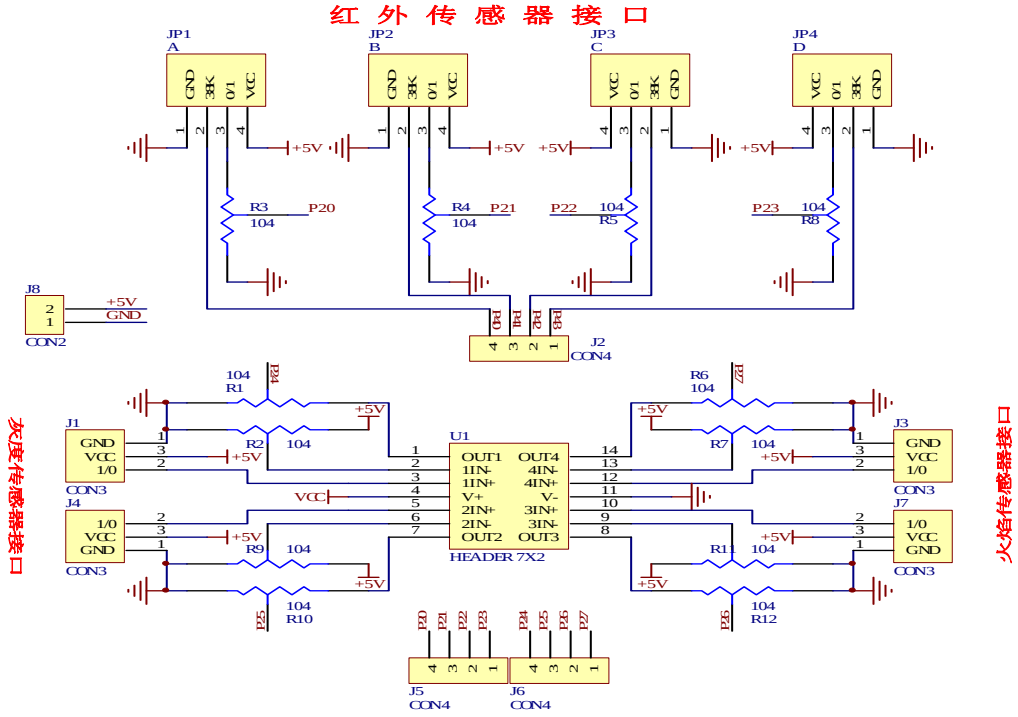


图 3-4 传感器部分电路图

4.3 软件设计

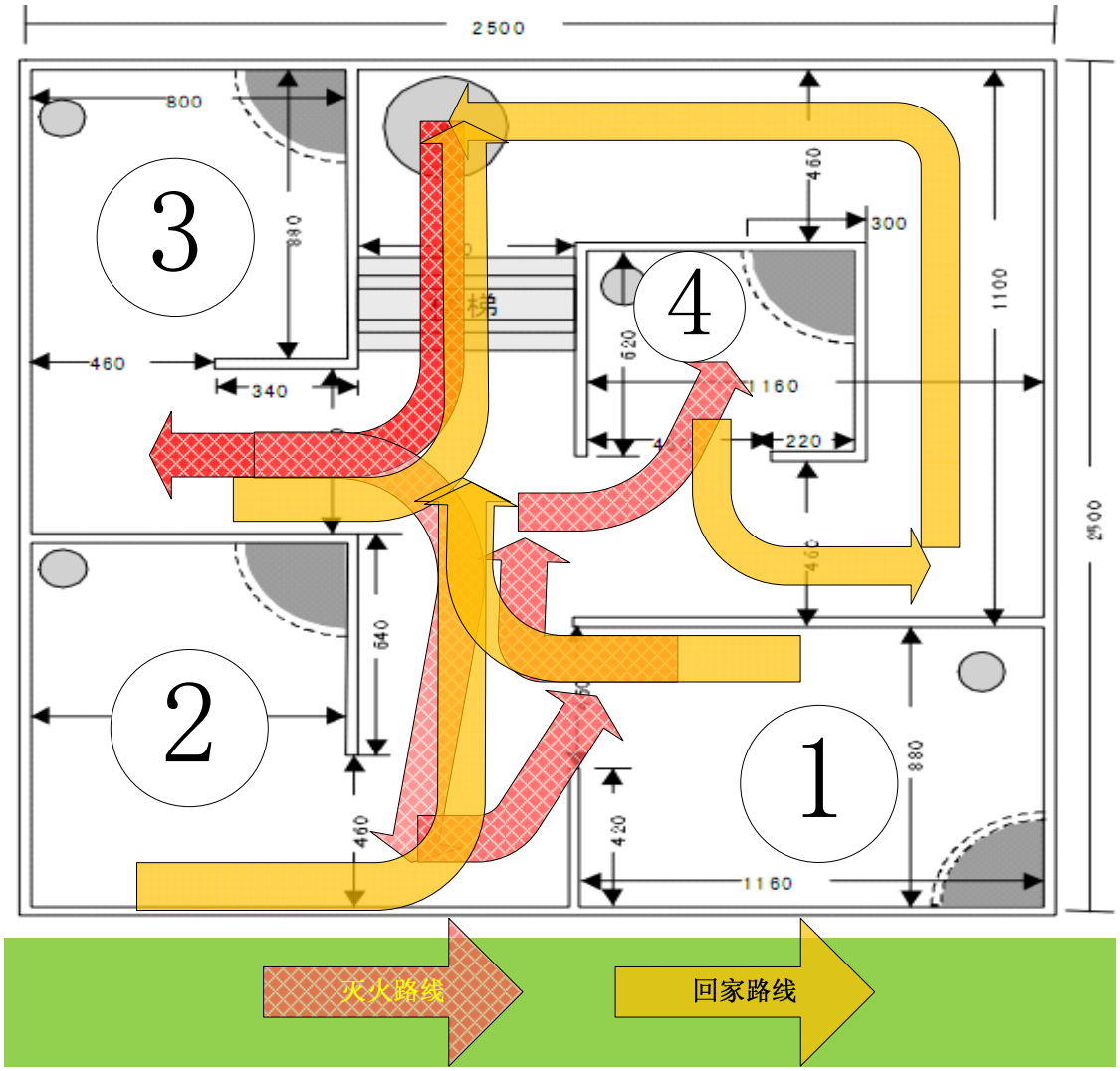


图 3-5 灭火小车灭火行进路线

灭火机器人行进路线分析：

当小车处于起点，小车要开始搜索房间有两种路径可以选择，一是不过台阶，绕着 4 号房间向外搜索。二是直接过台阶，然后开始搜索。显然直接过台阶可以节省很多的时间，路径更短，因为我们制作的小车为履带结构，结合我们小车的特点和前面分析，我们选择过台阶。

过台阶后，小车处于 3 号和 4 号房间中间，由图可知，沿着右走的方案比较好，因此我们采用是右手规则，首先搜索的是 3 号房间，如图中的红色箭头。当在 3 号房间发现火源时，小车进入房间并灭火，灭火后按原路返回；如没有发现火源，小车继续按右手规则搜索房间，直到搜索 4 号房间，不管有没有搜索到火源，从 4 号房间出来都绕着 4 号房间返回起点，因为回家过程中的时间不记入总时间，而绕行比较安全，小车比较好控制。

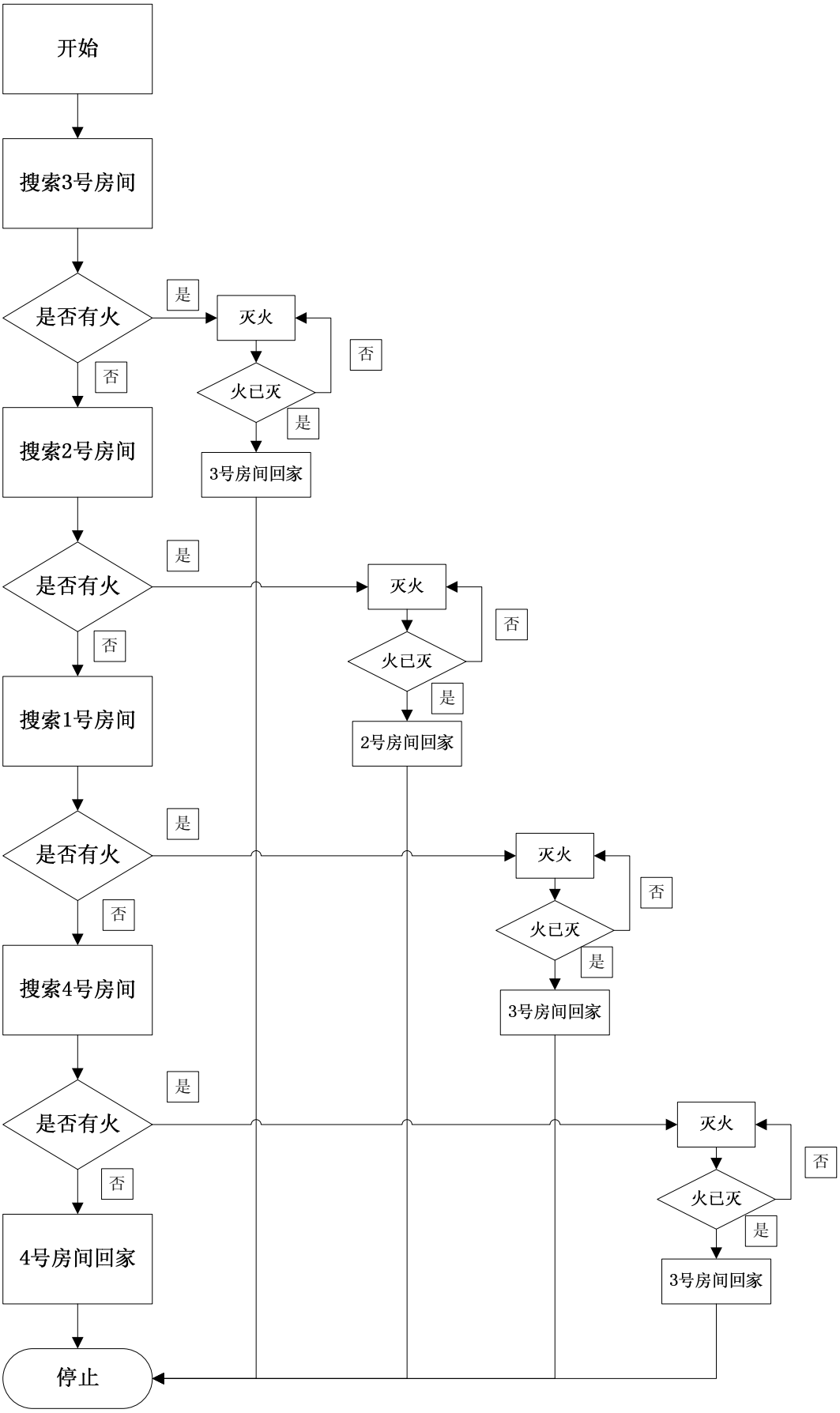


图 3-6 灭火小车软件设计流程图

第四章 调试记录

- 前方传感器检测最佳距离 23cm ，500R 的电位器逆时钟旋转可加大发射管的发射功率，检测距离可变远。
- 地面灰度传感器：测试距离 2.5cm,黑地面输出电压 1.3-1.5V；白纸输出 3.8-4.5V；
- 前方火焰传感器最远测试距离 2.5m，此次使用有效距离 0.8m，输出电压 0.6V，探测角度±30°。
- 转弯：

动作	延时常数	动作	延时常数
原地右转 90	60	原地左转 90	60
右后转 180	232	左后转 180	232

- 电池电压:5V 供电的电压不得低于 7.2V，12V 供电的电压不得低于 14.5V（在 15.7V 左右电机运行最稳定）。
- SHARP 传感器输出值：

SHARP GP2D12 红外测距传感器特性						
	左	右	后		平均值	ADC12 转换值
距离(cm)	电压(v)	电压(v)	电压(v)		电压(v)	参考电压(3.3v)
10.0	2.40	2.48	2.41		2.44	0x0bd3
11.0	2.22	2.32	2.25		2.27	0x0b00
12.0	2.08	2.15	2.07		2.12	0x0a46
13.0	1.93	2.00	1.93		1.97	0x098c
14.0	1.81	1.86	1.82		1.84	0x08eb
15.0	1.75	1.76	1.71		1.76	0x0888
16.0	1.66	1.67	1.60		1.67	0x0818
17.0	1.58	1.59	1.53		1.59	0x07b5
18.0	1.49	1.51	1.47		1.50	0x0745
19.0	1.42	1.44	1.40		1.43	0x06ee
20.0	1.36	1.38	1.32		1.37	0x06a4
21.0	1.30	1.31	1.26		1.31	0x0659
22.0	1.25	1.25	1.21		1.25	0x060f
23.0	1.21	1.21	1.16		1.21	0x05dd
24.0	1.18	1.17	1.11		1.18	0x05b8
25.0	1.12	1.12	1.06		1.12	0x056d
26.0	1.08	1.10	1.02		1.09	0x0548
27.0	1.05	1.05	1.00		1.05	0x0516
28.0	1.02	1.01	0.96		1.02	0x04f1
29.0	0.99	0.99	0.92		0.99	0x04cc
30.0	0.95	0.95	0.91		0.95	0x049a
31.0	0.93	0.92	0.87		0.93	0x0482
32.0	0.91	0.90	0.85		0.91	0x0469

33.0	0.88	0.88	0.83		0.88	0x0444
34.0	0.86	0.86	0.79		0.86	0x042b
35.0	0.84	0.84	0.77		0.84	0x0412
36.0	0.82	0.80	0.75		0.81	0x03ed
37.0	0.80	0.78	0.73		0.79	0x03d4
38.0	0.78	0.77	0.71		0.78	0x03c7
39.0	0.76	0.75	0.69		0.76	0x03af
40.0	0.74	0.74	0.62		0.74	0x0396
41.0	0.73	0.72	0.60		0.73	0x0389
42.0	0.72	0.70	0.56		0.71	0x0371
43.0	0.70	0.68	0.58		0.69	0x0358
44.0	0.68	0.67	0.58		0.68	0x034b
45.0	0.67	0.66	0.56		0.67	0x033f
46.0	0.66	0.64	0.54		0.65	0x0326
47.0	0.65	0.63	0.53		0.64	0x031a

第五章 实验心得

周会泉：

本次的灭火机器人小车制作主要涉及到单片机开发,机器人组成和原理,电机与驱动,传感器知识,程序算法设计等.通过本次机器人小车的制作,我对机器人小车的组成和原理,传感器有了全新的认识,我熟悉了整个机器人小车的制作过程.

在参加灭火机器人比赛前,我们的机器人小车是一个非常成功的作品,在实验室模拟环境下运行相当稳定,表现也非常出色.但在参加完比赛后,我们的机器人小车却是一个失败的作品.下面将主要介绍本次参加比赛的调试经验和比赛经验.以备将来做进一步的改进.

一,在调试时,尽量在自己的比赛场地调试,虽然在现场比赛时,所有的比赛场地采用的都是相同的材料,各个部分看起来都是一样的,但是实际中却会有很大差异.如果不在自己的比赛场地调试,严格来说,现场调试将没有任何意义.

二,千万不要以为自己的小车有多快,其实别人的小车更快,所以在制作小车时追求速度是必要的.在保证速度的同时也一定要保证自己小车的稳定性.

三,在实际比赛时,可以根据实际情况加上各种模式,比如非推测导航模式实际上是在四号房间边上的巷道加上一个小斜坡,而模糊模式是在起点位置靠左加上两片直径约 5,6 厘米的镜子,其实这些对小车并没有很大的影响.

四,如有可能,尽量采用相关公司的成品小车,这样小车会更稳定,我们也只需要将主要精力放在程序设计上,这样会增大成功的可能性.

五,去现场比赛时,尽量多点人去,也尽量提前到达比赛现场进行调试,如果有足够时间调试,自己制作的小车同样不逊于某些公司制作的机器人小车.

六,在制作小车时,要充分考虑到各种不利情况.这要求制作的机器人的适应能力好,到达

现场时需要调整的参数越少越好.

七,在控制机器人小车精确转弯时一定要使用相关硬件器件进行控制,比如指南针.或者采用好的算法不需要进行精确转弯.

总体来说,就参加比赛而言,我们的机器人小车是一个失败的作品.但对我和我的队友来说,我们是成功的,我们制作的机器人小车也是成功的.因为灭火机器人比赛中我们的小车是唯一的一个自己动手制作并搭建的.

陈云：

此次设计的主体是小车，使用最多的是传感器，因为传感器、电机驱动等电路已经有现成的电路，所以在硬件设计的时候，遇到的问题比较少，设计也比较顺利；但调试遇到的问题还是比较多的。

在整个调试过程中，我总共做了三副板子，第一副全是自己焊的，做测试电路用的；问题大部分出在了第二副板子上，由于当时做得部分 PCB 电路板出了问题，所以使用的是 PCB 和自焊的结合，但系统很不稳定，主要是驱动芯片是插在上面的，没有固定，结果稍微动一下就要软件修改参数，更别提拆一次车，再装上参数就得全改；第三副板子就是参赛用得，效果比较好。这给我的启示是：做东西宁可慢些，但系统必须稳定。

在智能车的设计中，电源部分可以说是核心的核心；电源设计显得尤为重要，特别是使用电池供电的系统。电池在充电后，电压会变的很高，额定 7.2V 电压冲完电电压会达到 8.5V，但在使用初，电压降的会很快，对系统的稳定性造成很大威胁，所以必须使用稳压芯片，并且电池电压（正常供电时电压）至少应该比稳压芯片输出电压高 2V；另外，电源部分的滤波电容也是非常重要的，一般采用 100uF 的有极性电容和 104 无极性电容构成滤波电容组。

在这次设计中，我充分感受到经验的力量。因为我们这次使用的主控芯片引脚只能接受小于 3.3V 的电压信号，所以 5V 器件的输出全部要进行降压，我们使用了电位器进行分压降压。在使用红外传感器的时候，我用了 10k 的电位器进行降压，但输出一直不正确，起初我以为是自己电路出问题，于是去测电路，结果半天也没发现什么问题，最后张宁波学长（曾参加过多次比赛，有丰富的软硬件经验）过来，我把情况说了一下，他就直接说，传感器的输出使用了 10k 的上拉电阻，让我将分压电阻换成 50k 的电位器后，问题就解决了。通过这件事，我感到经验有时会让我们事半功倍，甚至可以走捷径，像我们这次使用的比较器，将模拟信号转换成数字信号。

这次是我第二次和同学合作一起作比赛，虽然我们各有分工，但也只是相对，在真正做东西的时候，我们需要充分的发挥自己的作用，互相协作，促进项目设计；一个人的想法是很有限的，如果孤立，很难成事！可以说，我们在一个团队中，需要相互协作，相互信任，相互学习，共同提高，体现“1+1>2”的团队精神。

翟小兵：

2008 年的第九届“广茂达”杯全国机器人大赛刚刚结束，我有太多的感触和体会。

我们是在九月三号开始全国机器人大赛的具体制作过程，由于在暑假刚参加陕西省 TI 杯模拟与数字混合电路设计大赛，所以对于电路方面的东西我们就轻车熟路了很多，电路的设计在很大程度上就讲求经验。我们以前在书本中所学的知识很多通过到实践中去后，会得到不一样的收获，能帮助我们进一步理解知识，想得更透彻。而且很多知识到实践中去，才能真正体会到它的用处，所碰到的问题和想其解决之道。这次大赛让我对这一点有更深刻的认识。

刚了解到灭火机器人的资料后，在我们的头脑中只是依稀存在着印象而已，不能体会到其中真正如何完成，觉得很含糊。对那个灭火场地该如何跑完更是一头雾水，看到那一个一个的不规律的房间心里就直发毛了。

接下来就是从一个一个的功能来实现了，最先要解决的是车子问题。车子不仅要跑的快而且性能要稳定。现在的学习我们要借助最好的老师“网络”。在网上几乎可以搜到你想解决的各类问题，而且有各种方案供你参考，这样你的实验才会事半功倍。我们就是在网上找到合适的车子，然后买回来的，不然在实验室根本想不出更好的方法。其次就是选择各个功能的组成电路了，不同的电路有不同的优势，要中和这些来考虑了。在这个过程中最好通过面包板来将各种电路实验一遍找出性能最稳定的一种电路就行，然后再去制版就不会出现原理性的问题了。还有就是制版的问题。在画 PCB 板图的时候要注意两点，一就是要细心，图在原理图时最好不要出低级错误，这个可以多检查几遍，也可以找其他人来帮忙检查。还有在画元器件的封装时最好多核查几遍，这个也要讲求准确。二就是在元件布局和走线的时候要按照电气规则来画，这样的电路在以后才会性能更高。而且在高要求的电路中你自己的技能才能得到提高。

最后就是系统的整体调试。这在过程整个实验中是最关键的，也是最麻烦的，我们要调节各个参数，保证车子能正常完成各个功能。同时还要考虑出现的各种不良因素，将其在一一除掉，使系统的性能得到优化和提高。

这次比赛让我懂得了很多专业知识，但是让我体会更多的是在一个团队中最重要的团结协作。大家在互相讨论中，在彼此协作中我们相互交流，相互鼓励才得到今天的成果。所以说没有最强的个人，只有更强的团队。

附录 1：程序清单

//灭火机器人
//算法:3-2-1-4

附 1.1 main函数

```
//预处理
#include<msp430x14x.h>
#include"595.c"                  //数码管显示
#include"motor.c"                //电机驱动
#include"pose.c"                 //调姿函数
#include"search.c"               //房间搜索
#include"home.c"                 //回家
#include"cpu.c"                  //中断函数

//TimerA 初始化
void TimerAInit(void)
{
  TACTL=TASSEL1+ID1+MC1;     //SMCLK,4M,连续计数模式.
  TACCTL0=CCIE;                  //开中断
  TACCR0=1000;                   //1000us,1ms.
}
//P2 初始化
void Port2Init(void)
{
  P2SEL=0x00;
  P2DIR=0x00;
}
//ADC12 初始化
void AdcInit(void)
{
  P6SEL|=BIT0+BIT1+BIT2;          //P6.0,P6.1,P6.2.
  P6DIR&=~(BIT0+BIT1+BIT2);     //输入

  ADC12CTL0&=~ENC;                //清 ENC 位
  ADC12CTL0|=ADC12ON+MSC;         //采样保持时间为 4 个 ADC12CLK.
  ADC12CTL1|=SHP+CONSEQ_3;       //转换开始地址 0,多通道多次转换.

  ADC12MCTL0|=INCH_0;             //VR+=AVcc,VR-=AVss.
  ADC12MCTL1|=INCH_1;             //VR+=AVcc,VR-=AVss.
  ADC12MCTL2|=INCH_2+EOS;         //VR+=AVcc,VR-=AVss.
  ADC12CTL0|=ENC;                 //ADC 转换使能
}
```

```
//主函数
void main(void)
{
  WDTCTL=WDTPW+WDTHOLD;
  SCLKInit();                          //系统时钟
  TimerAInit();                        //定时器

  P1DIR=0xff;                          //P1   LED 指示灯
  Port2Init();                         //P2   不再使用中断,使用定时扫描.//传感器中断
  P3DIR|=0x55;                         //P3   电机驱动
  TimerB7Init();                       //P4   PWM
  P5DIR|=BIT0+BIT1+BIT3+BIT7; //P5   数码管,灭火风扇.
  AdcInit();                           //P6   A/D 转换

  Room=3;                              //先搜索房间 3
  P1OUT=0x00;                          //开所有传感器指示灯
  P1OUT|=BIT0;                         //开红灯,关绿灯.
  Forward();                           //小车前进
  UpMode;                              //定时器 B 增计数模式
  Out595(Room);                        //显示正准备搜索房间号
  P5OUT&=~BIT7;                        //关风扇
  ADC12CTL0|=ADC12SC;                  //ADC 开始转换

  _EINT();

  while(1)
  {
    switch(Room)
    {
      case 1:
        SearchRoom1();                 //搜索 1 号房间
        FindFire(Room);                //找火源
        if(FireFlag)
        {
          //靠近火源
          Forward();                   //前进,准备灭火.
          for(Num=7;Num>0;Num--)
            Delay(0);
          while(!LineFlag);            //等待小车压到白线
          Stop();                      //小车停止
          //灭火
          while(FireFlag)              //判断火是否已灭
          {
            FanRun;                    //启动风扇灭火
```

```
            for(Num=15;Num>0;Num--)
              Delay(0);                    //延时
        }
        FanStop;                           //关风扇
        //回家
        Gohome1();
    }
    else
    {
        Forward();
        while(!(IrdaLF|IrdaRF));   //等待找到 2 号房间的墙壁
        TurnRight90();
        Room=4;                            //搜索房间 4
    }
    break;
case 2:
    SearchRoom2();                         //搜索 2 号房间
    FindFire(Room);                        //找火源
    if(FireFlag)
    {
        //靠近火源
        Forward();                         //前进,准备灭火.
        while(!LineFlag);                  //等待小车压到白线
        Stop();                            //小车停止
        //灭火
        while(FireFlag)                    //判断火是否已灭
        {
            FanRun;                        //启动风扇灭火
            for(Num=15;Num>0;Num--)
              Delay(0);                    //延时
        }
        FanStop;                           //关风扇
        //回家
        Gohome2();
    }
    else
    {
        Forward();                         //前进
        while(!(IrdaLF|IrdaRF))    //等待到达墙
        {
            PoseSharpRB();                 //调姿
        }
        TBCCR2=96;
        TBCCR4=104;
```

```
        TurnLeft90();
        Room=1;                          //搜索房间 1
    }
    break;
case 3:
    SearchRoom3();                       //搜索 3 号房间
    FindFire(Room);
    if(FireFlag)
    {
        //靠近火源
        Forward();                       //前进,准备灭火.
        for(Num=7;Num>0;Num--)           //延时消除白线误判
            Delay(0);
        while(!LineFlag);                //等待小车压到白线
        Stop();                          //小车停止
        //灭火
        while(FireFlag)                  //判断火是否已灭
        {
            FanRun;                      //启动风扇灭火
            for(Num=15;Num>0;Num--)
                Delay(0);                //延时
        }
        FanStop;                         //关风扇
        //回家
        Gohome3();
    }
    else
    {
        Forward();                       //前进
        while(!(IrdaLF|IrdaRF))          //等待到达墙
        {
            PoseSharpR();                //调姿
        }
        TBCCR2=96;
        TBCCR4=104;
        TurnLeft90();                    //左转 90 度
        Room=2;                          //搜索房间 2
    }
    break;
case 4:
    SearchRoom4();                       //搜索 4 号房间
    FindFire(Room);
    if(FireFlag)
    {
```

```
    //靠近火源
    Forward();                          //前进,准备灭火.
    for(Num=7;Num>0;Num--)       //延时消除白线误判
      Delay(0);
    while(!LineFlag);               //等待小车压到白线
    Stop();                             //小车停止
    //灭火
    while(FireFlag)                  //判断火是否已灭
    {
      FanRun;                           //启动风扇灭火
      for(Num=15;Num>0;Num--)
        Delay(0);                       //延时
    }
    FanStop;                            //关风扇
    //回家
    Gohome4();
}
else
{
    Out595(H);

    Forward();
    while(!(IrdaLF|IrdaRF));   //等待到达 1 号房间的墙壁
    TurnLeft90();

    Forward();
    while(!(IrdaLF|IrdaRF))     //等待到墙壁
    {
      PoseSharpRB();
    }
    TBCCR2=96;
    TBCCR4=104;
    TurnLeft90();

    Forward();
    while(!(IrdaLF|IrdaRF))     //等待到墙壁
    {
      PoseSharpRB();
    }
    TBCCR2=96;
    TBCCR4=104;
    TurnLeft90();

    Forward();
```

```
            while(!(IrdaFlag&LineFlag))//等待到起点白圈
            {
              PoseSharpRB();
            }
            TBCCR2=96;
            TBCCR4=104;

            P1OUT=0xff;
            End();
          }
          break;
        default:
          break;
      }
    }
}
```

附 1.2 电机与单片机接口和定义

```
//小车转弯函数
//输入接 P3,使能接 P4.
#ifndef MOTOR_C
#define MOTOR_C
#include<msp430x14x.h>
#define uchar unsigned char
#define uint   unsigned int

//电机
#define L1 BIT0
#define L2 BIT2        //左电机
#define R1 BIT4
#define R2 BIT6        //右电机
#define LeftStop         P3OUT|=L1+L2              //左电机停止
#define LeftForward      P3OUT=(P3OUT&~L2)|L1   //左电机前进
#define LeftBack         P3OUT=(P3OUT&~L1)|L2   //左电机后退
#define RightStop        P3OUT|=R1+R2              //右电机停止
#define RightForward   P3OUT=(P3OUT&~R2)|R1   //右电机前进
#define RightBack        P3OUT=(P3OUT&~R1)|R2   //右电机后退

//电池+稳压芯片
//有两个参数可能需要调整
#define TurnLeft90()   TurnLeft(22800)          //左转 90 度   //??
#define TurnRight90() TurnRight(22000)         //右转 90 度   //??
```

```
//控制定时器 B 工作方式
#define StopMode TBCTL&=~(MC0+MC1)
#define UpMode      TBCTL|=MC0

//系统时钟初始化//system clock
//MCLK,4M//SMCLK,4M.
void SCLKInit(void)
{
   uchar i;
   BCSCTL1&=~XT2OFF;

   do
   {
      IFG1&=~OFIFG;
      for(i=0xff;i>0;i--);
   }
   while((IFG1&OFIFG));
   //4M
   BCSCTL2=SELM_2+SELS;
}
//TimerB7 初始化
void TimerB7Init(void)
{
   P4SEL|=0x7e;                //0111 1110
   P4DIR|=0x7e;                //输出 PWM 方波

   TBCTL=TBSSEL1;               //SMCLK
   TBCCR0=104;                 //38kHz,105-1.
   //红外,38kHz.
   TBCCTL1=OUTMOD_7;          //复位/置位模式
   TBCCR1=52;                  //占空比约 50%
   TBCCTL3=OUTMOD_7;          //复位/置位模式
   TBCCR3=52;                  //占空比约 50%
   TBCCTL5=OUTMOD_7;          //复位/置位模式
   TBCCR5=52;                  //占空比约 50%
   TBCCTL6=OUTMOD_7;          //复位/置位模式
   TBCCR6=52;                  //占空比约 50%
   //电机调速 PWM//11.86v
   //有一个参数可能需要调整
   TBCCTL2=OUTMOD_7;          //复位/置位模式,左.
   TBCCR2=96;                  //输出 PWM//??
   TBCCTL4=OUTMOD_7;          //置位/置位模式,右.
   TBCCR4=104;                 //几乎输出高电平
```

```
}

//延时函数//7+4*x+3
//延时 1ms//Delay(2000)
void Delay(uint x)
{
   while(--x);
}

//小车停止
void Stop(void)
{
   LeftStop;
   RightStop;
}
//小车前进
void Forward(void)
{
   LeftForward;
   RightForward;
}
//小车后退
void Back(void)
{
   LeftBack;
   RightBack;
}
//小车左转 x 度
void TurnLeft(uint x)
{
   uchar i;

   Stop();
   Delay(50000);
   LeftBack;
   RightForward;
   for(i=20;i>0;i--)
   {
      Delay(x);
   }
   Stop();
   Delay(1000);
}
//小车右转 x 度
```

```
void TurnRight(uint x)
{
   uchar i;

   Stop();
   Delay(50000);
   RightBack;
   LeftForward;
   for(i=20;i>0;i--)
   {
      Delay(x);
   }
   Stop();
   Delay(1000);
}
//寻找火源
void FindFire(uchar room)
{
   uchar num=0;
   Stop();
   Delay(50000);

   while(!(P2IN&0x82))          //扫描是否有火焰
   {
      Stop();
      RightBack;
      LeftForward;
      Delay(5000);              //5ms

      num++;
      if(num>164)               //??
      {
         Stop();
         Delay(50000);
         break;
      }
   }

   if(P2IN&0x82)
   {
      switch(room)
      {
         case 1:
            TurnLeft(2800);     //??
```

```
            break;
        case 2:
            TurnLeft(2000);          //??
            break;
        case 3:
            TurnLeft(2000);          //??
            break;
        case 4:
                break;
        default:
            break;
      }
   }
}

#endif
```

附 1.3 姿势调整

```
#include<msp430x14x.h>
#include"motor.c"
#define uchar unsigned char
#define uint    unsigned int

//变量声明
uchar IrdaFlag;                          //红外检测到前方障碍
uchar IrdaLF,IrdaRF,IrdaL,IrdaR;    //四个红外避障传感器检测到障碍物标志
uchar LineFlag;                          //检测到白线
uchar LineL,LineR;                       //两个光敏传感器检测到白线
uchar FireFlag;                          //检测到火焰
uchar FireL,FireR;                       //两个火焰传感器检测到火焰
uchar Room;                                 //房间号
uchar Num;                                  //延时计数变量
uint SharpL,SharpR,SharpB;               //左右夏普红外避障传感器采样平均值

//SHARP 红外测距传感器转换值表
//0,1cm 没有测量,以 0x0000 代替;2cm 以 2.5cm 的代替.
uint sharp[]={
0x0000,0x0000,0x048e,0x0745,0x0999,      //0-4cm
0x094e,0x0a15,0x0c1e,0x0ccc,0x0c8d,      //5-9cm
0x0bd3,0x0b00,0x0a46,0x098c,0x08eb,      //10-14cm
0x0888,0x0818,0x07b5,0x0745,0x06ee,      //15-20cm
0x06a4,0x0659,0x060f,0x05dd,0x05b8,      //21-24cm
0x056d,0x0548,0x0516,0x04f1,0x04cc,      //25-29cm
```

```
0x049a,0x0482,0x0469,0x0444,0x042b,     //30-34cm
0x0412,0x03ed,0x03d4,0x03c7,0x03af,     //35-39cm
0x0396,0x0389,0x0371,0x0358,0x034b,     //40-44cm
0x033f,0x0326,0x031a,0x030d,0x0301,     //45-49cm
0x0301                                  //50cm
};

//根据两 SHARP 红外测距传感器调整姿势
//以左为准
void PoseSharpL(void)
{
  if(SharpL>sharp[13])            //SHARP 距墙小于 13cm
  {
    TBCCR2=96;
    TBCCR4=82;
  }
  else if(SharpL<sharp[25])     //SHARP 距墙大于 25cm
  {
    TBCCR2=82;
    TBCCR4=104;
  }
  else                                //其他
  {
    TBCCR2=96;
    TBCCR4=104;
  }
}
//以右为准
void PoseSharpR(void)
{
  if(SharpR>sharp[13])            //SHARP 距墙小于 13cm
  {
    TBCCR2=82;
    TBCCR4=104;
  }
  else if(SharpR<sharp[25])     //SHARP 距墙大于 25cm
  {
    TBCCR2=96;
    TBCCR4=82;
  }
  else                                //其他
  {
    TBCCR2=96;
    TBCCR4=104;
```

```
    }
}
//以右边两个为标准
void PoseSharpRB(void)
{
    //<13cm
    if(SharpR>sharp[13])
    {
        TBCCR2=82;       //左
        TBCCR4=104;      //右
    }
    //13-15cm
    else if((SharpR<=sharp[13])&&(SharpR>=sharp[15]))
    {
        if(SharpB>=sharp[12])
        {
            TBCCR2=96;
            TBCCR4=82;
        }
        else if(SharpB<=sharp[16])
        {
            TBCCR2=82;
            TBCCR4=104;
        }
        else
        {
            TBCCR2=96;
            TBCCR4=104;
        }
    }
    //16-18cm
    else if((SharpR<=sharp[16])&&(SharpR>=sharp[18]))
    {
        if(SharpB>=sharp[15])
        {
            TBCCR2=96;
            TBCCR4=82;
        }
        else if(SharpB<=sharp[19])
        {
            TBCCR2=82;
            TBCCR4=104;
        }
        else
```

```
    {
      TBCCR2=96;
      TBCCR4=104;
    }
}
//19-21cm
else if((SharpR<=sharp[19])&&(SharpR>=sharp[21]))
{
    if(SharpB>=sharp[18])
    {
      TBCCR2=96;
      TBCCR4=82;
    }
    else if(SharpB<=sharp[22])
    {
      TBCCR2=82;
      TBCCR4=104;
    }
    else
    {
      TBCCR2=96;
      TBCCR4=104;
    }
}
//22-24cm
else if((SharpR<=sharp[22])&&(SharpR>=sharp[24]))
{
    if(SharpB>=sharp[21])
    {
      TBCCR2=96;
      TBCCR4=82;
    }
    else if(SharpB<=sharp[25])
    {
      TBCCR2=82;
      TBCCR4=104;
    }
    else
    {
      TBCCR2=96;
      TBCCR4=104;
    }
}
//>24cm
```

```
    else
    {
      if(SharpB>sharp[23])
      {
        TBCCR2=96;
        TBCCR4=82;
      }
      else
      {
        TBCCR2=96;
        TBCCR4=104;
      }
    }
}
```

附 1.4 中断函数

```
//P2 口中断,ADC 中断不可行,有待深入学习.
//改成定时中断,进行定时扫描和采集.
//*****************************************
//本文件 cpu.c 中没有需要调整的参数
//*****************************************

//红外指示灯
#define IrdaLedL0 P1OUT&=~BIT2
#define IrdaLedL1 P1OUT|=BIT2
#define IrdaLedR0 P1OUT&=~BIT3
#define IrdaLedR1 P1OUT|=BIT3
//火焰指示灯
#define FireLedL0 P1OUT&=~BIT4
#define FireLedL1 P1OUT|=BIT4
#define FireLedR0 P1OUT&=~BIT5
#define FireLedR1 P1OUT|=BIT5
//光敏指示灯
#define LineLedL0 P1OUT&=~BIT6
#define LineLedL1 P1OUT|=BIT6
#define LineLedR0 P1OUT&=~BIT7
#define LineLedR1 P1OUT|=BIT7

//定时器 A 中断
#pragma vector=TIMERA0_VECTOR
__interrupt void TimerA(void)
{
```

```
//    uint left,right,back;
    TACCR0+=1000;
//P2,红外避障,火焰,光敏.
    if((P2IN&0x41)!=0x41) //红外    0100 0001
    {
        IrdaFlag=1;
        //右前
        if(!(P2IN&0x01))
        {
            IrdaRF=1;
            IrdaLedR1;
        }
        else
        {
            IrdaRF=0;
            IrdaLedR0;
        }
        //左前
        if(!(P2IN&0x40))
        {
            IrdaLF=1;
            IrdaLedL1;
        }
        else
        {
            IrdaLF=0;
            IrdaLedL0;
        }
    }
    else
    {
        IrdaFlag=0;
        IrdaRF=0;
        IrdaLF=0;

        IrdaLedL0;
        IrdaLedR0;
    }

    if(P2IN&0x28)               //光敏    0010 1000
    {
        LineFlag=1;
        //左
        if(P2IN&0x20)
```

```
    {
      LineL=1;
      LineLedL1;
    }
    else
    {
      LineL=0;
      LineLedL0;
    }
    //右
    if(P2IN&0x08)
    {
      LineR=1;
      LineLedR1;
    }
    else
    {
      LineR=0;
      LineLedR0;
    }
  }
  else
  {
    LineFlag=0;
    LineL=0;
   LineR=0;

    LineLedL0;
    LineLedR0;
  }
  if(P2IN&0x82)              //火焰   1000 0010
  {
    FireFlag=1;
    //左
    if(P2IN&0x02)
    {
      FireL=1;
      FireLedL1;
    }
    else
    {
      FireL=0;
      FireLedL0;
    }
```

```
      //右
      if(P2IN&0x80)
      {
         FireR=1;
         FireLedR1;
      }
      else
      {
         LineR=0;
         FireLedR0;
      }
   }
   else
   {
      FireFlag=0;
      FireL=0;
      FireR=0;

      FireLedL0;
      FireLedR0;
   }

   //ADC,红外测距.
   SharpL=ADC12MEM0;
   SharpR=ADC12MEM1;
   SharpB=ADC12MEM2;

   ADC12CTL0|=ADC12SC;
}
```

附 1.5 房间搜索

```
//基于 3-2-1-4
//*****************************************
//本文件 search.c 中需要调整的参数如下:
//1,观察各搜索房间函数中的相关延时,如需要则调整.
//*****************************************

#include<msp430x14x.h>
#include"595.c"
#include"motor.c"
//系统指示灯
#define Gre0 P1OUT&=~BIT0
#define Gre1 P1OUT|=BIT0
```

```
#define Red0 P1OUT&=~BIT1
#define Red1 P1OUT|=BIT1

//搜索房间 1//测试成功
//有一个参数可能需要调整
void SearchRoom1(void)
{
   Red0;
   Gre1;
   Out595(1);

   Forward();                    //前进
   for(Num=3;Num>0;Num--)
      Delay(0);                  //延时以清除刚转弯时对右边路口的误判
   while(SharpR>sharp[40])     //等待找到房间入口
   {
      PoseSharpR();
   }
   TBCCR2=96;
   TBCCR4=104;
   for(Num=4;Num>0;Num--)       //
      Delay(0);                  //延时以等待小车处于房间中间
   TurnRight90();                //右转准备进入房间

   Forward();
   while(!LineFlag);            //等待到达房间入口白线
   Stop();
}
//搜索房间 2//测试成功
//有两个参数可能需要调整
void SearchRoom2(void)
{
   Red0;
   Gre1;
   Out595(2);                    //显示房间号

   Forward();                    //前进
   for(Num=3;Num>0;Num--)
      Delay(0);                  //延时
   while(SharpR>sharp[40])     //等待到达转弯处
   {
      PoseSharpRB();
   }
   TBCCR2=96;
```

```
  TBCCR4=104;
  for(Num=5;Num>0;Num--)      //
    Delay(0);                  //延时以等待车身到达路口中央
  TurnRight90();               //右转

  Forward();                   //转弯后继续前进
  for(Num=10;Num>0;Num--)     //
    Delay(0);                  //延时以清除刚转弯时对右边路口的误判
  while(SharpB<sharp[40]);   //等待后面的 SHARP 检测到墙壁
  while(SharpR>sharp[40])     //等待找到房间入口
  {
    PoseSharpR();
  }
  TBCCR2=96;
  TBCCR4=104;
  while(!(IrdaLF|IrdaRF));   //等待找到墙壁
  TurnRight90();               //右转准备进入房间

  Forward();
  while(!(IrdaLF|IrdaRF))
  {
    PoseSharpL();
  }
  TBCCR2=96;
  TBCCR4=104;
  Stop();
}
//搜索房间 3//测试成功
//有两个参数可能需要调整
void SearchRoom3(void)
{
  Red0;
  Gre1;
  Out595(3);                   //显示房间号

  Forward();                   //前进
  for(Num=3;Num>0;Num--)
    Delay(0);                  //延时
  while(SharpR>sharp[40])     //等待到达转弯处
  {
    PoseSharpRB();
  }
  TBCCR2=96;
  TBCCR4=104;
```

```
  while(SharpL>sharp[40]);   //等待到达路口中央
  TurnRight90();             //右转
  Forward();                 //转弯后继续前进
  for(Num=10;Num>0;Num--)    //
    Delay(0);                //延时以清除刚转弯时对右边路口的误判
  while(!(IrdaLF|IrdaRF))    //等待找到墙壁
  {//&,同时检测到墙壁,以防止由于其中一个坏了而引起逻辑混乱.
    PoseSharpL();
  }
  TBCCR2=96;
  TBCCR4=104;
  TurnRight90();             //右转准备进入房间

  Forward();
  while(!LineFlag);          //等待小车运行到房间入口白线
  Stop();                    //停止
}
//搜索房间 4//测试成功
//有两个参数可能需要调整
void SearchRoom4(void)
{
  Red0;
  Gre1;
  Out595(4);

  Forward();                 //前进
  for(Num=3;Num>0;Num--)
    Delay(0);                //延时以清除误判
  while(SharpL>sharp[40])    //等待找到转弯
  {
    PoseSharpL();
  }
  TBCCR2=96;
  TBCCR4=104;
  TurnRight90();             //右转准备进入房间
  Forward();                 //转弯后继续前进
  for(Num=16;Num>0;Num--)    //
    Delay(0);                //延时以等待小车处于房间中间
  TurnLeft90();              //左转准备进入房间

  Forward();
  while(!LineFlag);          //等待小车运行到房间入口白线
  Stop();                    //停止
}
```

附 1.6 回家程序

```
//****************************************
//本文件 home.c 中需要调整的参数如下:
//1,各回家函数灭火完后的转弯时间,共四个.
//****************************************

//风扇开关
#define FanStop     P5OUT&=~BIT7
#define FanRun      P5OUT|=BIT7
//结束
void End(void)
{
   Stop();                          //电机停止
   StopMode;                        //定时器停止
   LPM4;                            //进入低功耗模式
}
//从房间 1 回家//测试成功
//有一个参数可能需要调整
void Gohome1(void)
{
   Red1;
   Gre0;
   Out595(H);

   //比正对着应偏大点
   TurnLeft(28000);              //对准右边的墙壁//??

   Forward();
   while(!(IrdaLF|IrdaRF));   //等待到达墙壁附近
   TurnLeft90();                 //左转对准房间出口

   Forward();
   while(SharpR>sharp[40])
   {
      PoseSharpRB();
   }
   TBCCR2=96;
   TBCCR4=104;
   while(!(IrdaLF|IrdaRF));   //等待小车出房间直到碰到墙
   TurnRight90();                //右转

   Forward();                       //前进
   while(SharpL>sharp[40])     //等待到达左边转弯
```

```
    {
      PoseSharpL();
    }
    TBCCR2=96;
    TBCCR4=104;
    for(Num=5;Num>0;Num--)
      Delay(0);

    while(SharpR<sharp[40]);     //等待找到 4 号房间墙壁
    for(Num=3;Num>0;Num--)
      Delay(0);
    while(SharpL<sharp[40])      //等待找到 3 号房间墙壁
    {
      PoseSharpR();
    }
    TBCCR2=96;
    TBCCR4=104;
    for(Num=3;Num>0;Num--)
      Delay(0);

    while(!(IrdaFlag&LineFlag))//等待到起点白圈
    {
      PoseSharpL();
    }
    TBCCR2=96;
    TBCCR4=104;

    P1OUT=0xff;
    End();
}
//从房间 2 回家//测试成功
//有两个参数可能需要调整
void Gohome2(void)
{
    Red1;
    Gre0;
    Out595(H);

    //比正对着应偏小点
    TurnRight(33000);              //??
    Forward();                     //前进
    while(SharpL>sharp[40])     //到达房间出口
    {
      PoseSharpL();
```

```
}
TBCCR2=96;
TBCCR4=104;
while(!(IrdaLF|IrdaRF));   //等待到墙
TurnLeft90();                    //左转

Forward();
while(!LineFlag)                 //等待到白线
{
  PoseSharpRB();
}
TBCCR2=96;
TBCCR4=104;
for(Num=6;Num>0;Num--)       //??
  Delay(0);
TurnLeft90();                    //左转

Forward();
while(SharpR>sharp[40]);   //等待找到 1 号房间入口
for(Num=3;Num>0;Num--)       //延时
  Delay(0);

while(SharpL>sharp[40])     //等待到达左边转弯
{
  PoseSharpL();
}
TBCCR2=96;
TBCCR4=104;
for(Num=5;Num>0;Num--)
  Delay(0);

while(SharpR<sharp[40]);   //等待找到 4 号房间墙壁
for(Num=3;Num>0;Num--)
  Delay(0);
while(SharpL<sharp[40])     //等待找到 3 号房间墙壁
{
  PoseSharpR();
}
TBCCR2=96;
TBCCR4=104;
for(Num=3;Num>0;Num--)
  Delay(0);

while(!(IrdaFlag&LineFlag))//等待到起点白圈
```

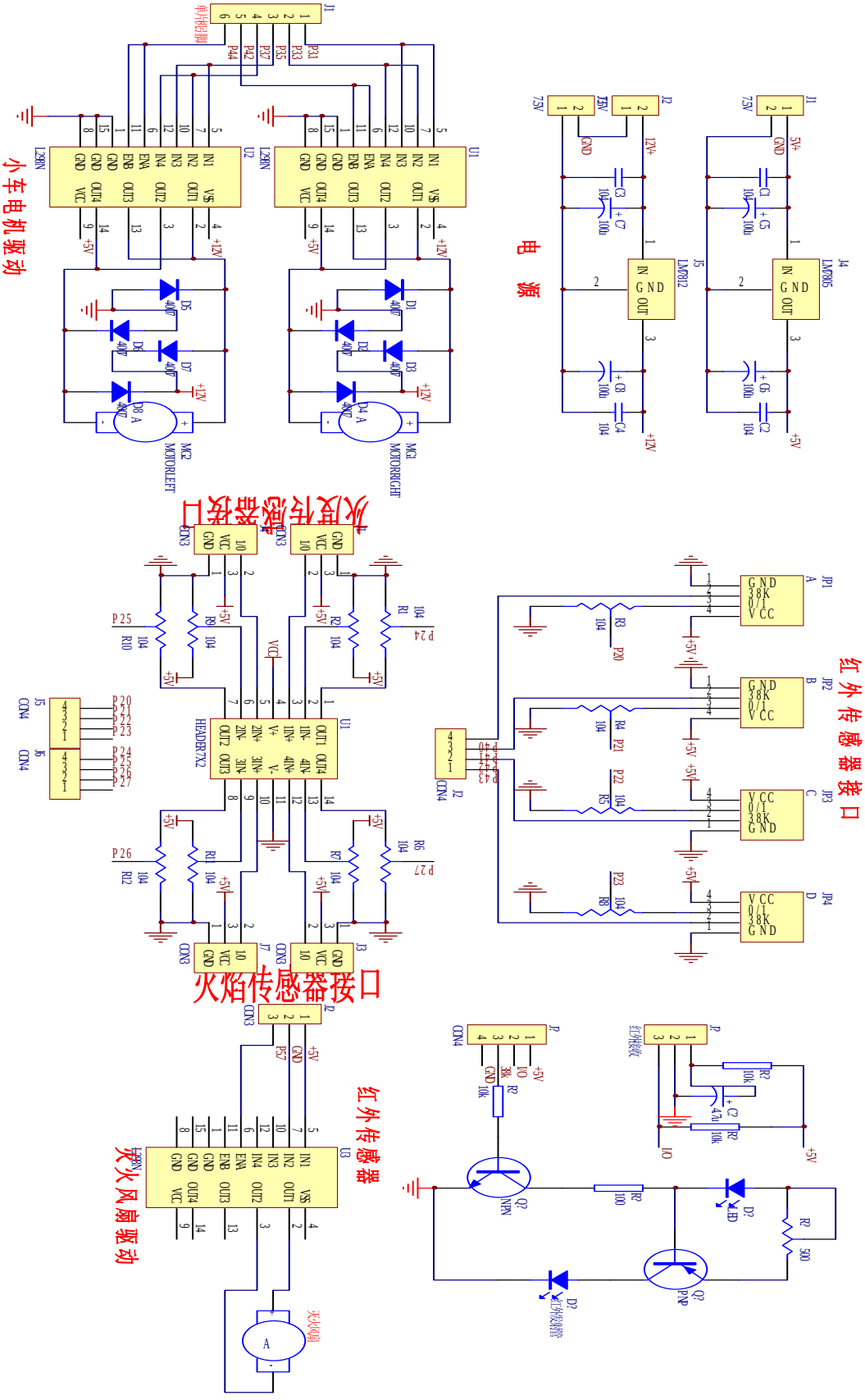
```
    {
      PoseSharpL();
    }
    TBCCR2=96;
    TBCCR4=104;

    P1OUT=0xff;
    End();
}
//从房间 3 回家//测试成功
//有两个参数可能需要调整
void Gohome3(void)
{
    Red1;
    Gre0;
    Out595(H);

    //正对
    TurnLeft(31000);                //对准右边的墙壁//??

    Forward();
    while(!(IrdaLF|IrdaRF));    //等待到达墙壁附近
    TurnLeft90();                   //左转对准房间出口

    Forward();
    while(!(IrdaLF|IrdaRF))     //等待小车运行到墙
    {
      PoseSharpRB();
    }
    TBCCR2=96;
    TBCCR4=104;
    TurnLeft90();                   //左转

    Forward();                      //转弯后继续前进
    while(SharpR>sharp[40])     //等待找到转弯
    {
      PoseSharpRB();
    }
    TBCCR2=96;
    TBCCR4=104;
    for(Num=5;Num>0;Num--)       //??
      Delay(0);                     //延时以等待小车处于路口中央
    TurnLeft90();                   //左转准备回家
```

```
    Forward();                          //前进
    while(SharpL<sharp[40])
    {
      PoseSharpR();
    }
    TBCCR2=96;
    TBCCR4=104;
    for(Num=3;Num>0;Num--)
      Delay(0);
    while(!(IrdaFlag&LineFlag))//等待到起点白圈
    {
      PoseSharpL();
    }
    TBCCR2=96;
    TBCCR4=104;

    P1OUT=0xff;
    End();
}
//从房间 4 回家//测试成功
//有一个参数可能需要调整
void Gohome4(void)
{
    Red1;
    Gre0;
    Out595(H);
    //比正对着应偏小点
    TurnLeft(28000);                //左转对准左边的墙//??
    Forward();
    while(!(IrdaLF|IrdaRF));   //等待找到墙
    TurnLeft90();                   //左转准备出房间
    Forward();
    while(SharpR>sharp[40])     //等待车身出房间
    {
      PoseSharpRB();
    }
    TBCCR2=96;
    TBCCR4=104;

    while(!(IrdaLF|IrdaRF));   //等待到达 1 号房间的墙壁
    TurnLeft90();

    Forward();
    while(!(IrdaLF|IrdaRF))     //等待到墙壁
```

```
    {
      PoseSharpRB();
    }
    TBCCR2=96;
    TBCCR4=104;
    TurnLeft90();

    Forward();
    while(!(IrdaLF|IrdaRF))     //等待到墙壁
    {
      PoseSharpRB();
    }
    TBCCR2=96;
    TBCCR4=104;
    TurnLeft90();
    Forward();
    while(!(IrdaFlag&LineFlag))//等待到起点白圈
    {
      PoseSharpRB();
    }
    TBCCR2=96;
    TBCCR4=104;
    P1OUT=0xff;
    End();
}
```

附录 2:（硬件设计电路图）

附图 总体电路图



附录 3:（参考文献）

[1] Texas Instruments Incorporated MSP430F14X C code Example

[2]Texas Instruments Incorporated MSP430F1XX Family slau049e.pdf

[3]Texas Instruments Incorporated MSP430F149 Datasheet

[4] 《国际赛制机器人灭火比赛规则》.PDF

[5] 沈建华,杨艳琴,骁曙. MSP430 系列 16 位超低功耗单片机原理与应用. 北京：清华大学出版社，2004.11

[6] 谢兴红,林凡强,吴雄英. MSP430 单片机基础与实践. 北京：北京航空航天大学出版社，2008.1

[7] 秦龙. MSP430 单片机常用模块与综合系统实例精讲. 北京：电子工业出版社，2007.7

[8] 李全利,迟荣强. 单片机原理及接口技术. 北京：高等教育出版社，2004.1

[9] 谭浩强. C 程序设计(第二版). 北京：清华大学出版社，1999.12

[10] 童诗白,华成英. 模拟电子技术基础(第三版). 北京：高等教育出版社，2003.12

[11] 康华光. 电子技术基础 数字部分(第四版). 北京： 高等教育出版社，1900.1

[12] 黄智伟. 全国大学生电子设计竞赛电路设计. 北京:北京航空航天大学出版社,2006.12

[13] 黄智伟. 全国大学生电子设计竞赛系统设计. 北京:北京航空航天大学出版社,2006.12

[14] 文艳,谭鸿. Protel 99 SE 电子电路设计. 北京：机械工业出版社，2006.8

附录 4:（灭火小车及灭火场地）