

电子线路设计训练专题实验 2 实验报告

西安交通大学



基于 C8051F020 智能控制器的 模拟直升机垂直升降控制系统设计 实验报告

自动化 94--胡欣盈--2194323176

目 录

一、 模拟直升机垂直升降控制系统简介	3
1.1 系统结构	3
1.2 系统功能	3
二、 实验设备	4
三、 硬件结构	4
3.1 C8051F020	5
3.1.1 C8051F020 单片机简介	5
3.1.2 C8051F020 单片机特性	6
3.1.3 C8051F020 引脚说明	6
3.2 LCD 显示电路	8
3.2.1 字符显示 RAM (DDRAM)	8
3.2.2 绘图 RAM (GDRAM)	9
3.3 按键电路	9
3.4 LED 显示电路	10
3.5 直升机垂直升降模拟对象	11
3.5.1 原理图	11
3.5.2 SS49E 线性霍尔效应传感器	13
四、 软件实现	13
4.1 功能介绍	13
4.2 系统流程图	14
4.3 主要函数代码	16
4.3.1 LED 显示	16
4.3.2 按键中断	17
4.3.3 LCD 显示	17
4.3.4 PID 控制算法	18
4.3.5 滤波器设计	18
五、 结果展示与分析	21
六、 实验总结	24
七、 参考文献	24

模拟直升机垂直升降控制系统设计

一、模拟直升机垂直升降控制系统简介

1.1 系统结构

模拟直升机垂直升降控制系统主要由 C8051F020 单片机、按键和显示等模块以及直升机垂直升降模拟对象组成，如图 1-1 所示。显示功能由液晶和数码管实现，液晶屏和数码管分别实现显示控制主菜单界面、当前霍尔电压的数值及变化曲线等功能，按键用于切换液晶屏显示内容、设置 PID 参数、增加和减少霍尔电压设定值等功能。

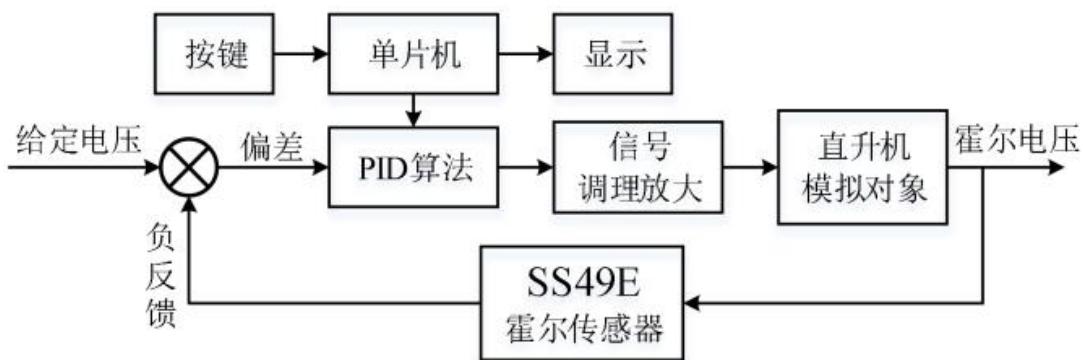


图 1-1 模拟直升机垂直升降控制系统框图

1.2 系统功能

系统设有以下 3 大板块：

1. LCD 显示区：LCD 用来显示个人信息与菜单，指引按键功能，在控制中也能作为示波器使用。
2. 键盘控制区：3 个按键在不同子菜单下被赋予了不同功能。
3. LED 提示区：3 组 LED 分别被赋予了显示电压设定值（用键盘可以增大、减小）、电压测量值、控制电压值的功能。

二、实验设备

基于 C8051F020 的智能控制器、直升机垂直升降模拟对象、计算机、Keil C51 编程软件等。

三、硬件结构

本次实验所用硬件以基于 C8051F020 的智能控制器为主，它是以 C8051F020 单片机为核心部件，增加了按键模块、数码管和液晶屏显示模块、AD 和 DA 输入输出信号调理电路等。智能控制器系统框图如图 3-1 所示，智能控制器实物板实物如图 3-2 所示。

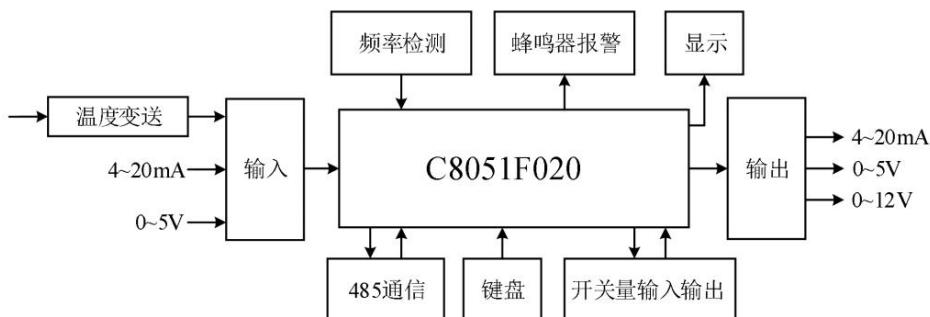


图 3-1 智能控制器系统框图

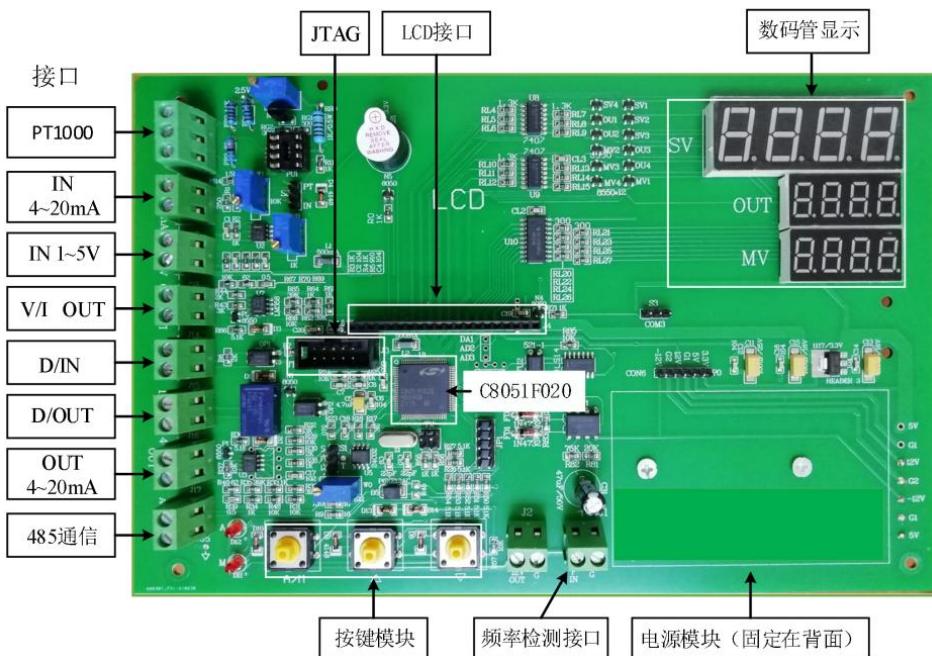


图 3-2 智能控制器实物板

3.1 C8051F020

C8051F 系列单片机是美国 Silabs 公司把 80C51 系列单片机从微控制器（MCU）时代推向片上系统（SOC）时代的产物。C8051F020 单片机是一种混合信号系统级 MCU 芯片，片内含 CIP-51 的 CPU 内核，其指令系统与 MCS-51 完全兼容。支持双时钟，其工作电压为 2.7~3.6V。

3.1.1 C8051F020 单片机简介

C8051F020 单片机的硬件原理框图如图 3-3 所示，它以 8051 内核为中心，通过 SFR 总线、外部数据存储器总线、系统时钟线、复位线等与 64KB 闪存、数字功能模块（如 UART、SPI、定时器等）、模拟功能模块（如比较器、A/D、D/A 等）、片上时钟系统和 JTAG 逻辑电路等相连。片内 JTAG 调试电路允许安装在应用系统上的产品对 MCU 进行非侵入式、全速、在系统调试。单片机的 64 个数字 I/O 引脚能够处理繁杂的键盘与液晶任务，增强了单片机对外围接口的处理能力。

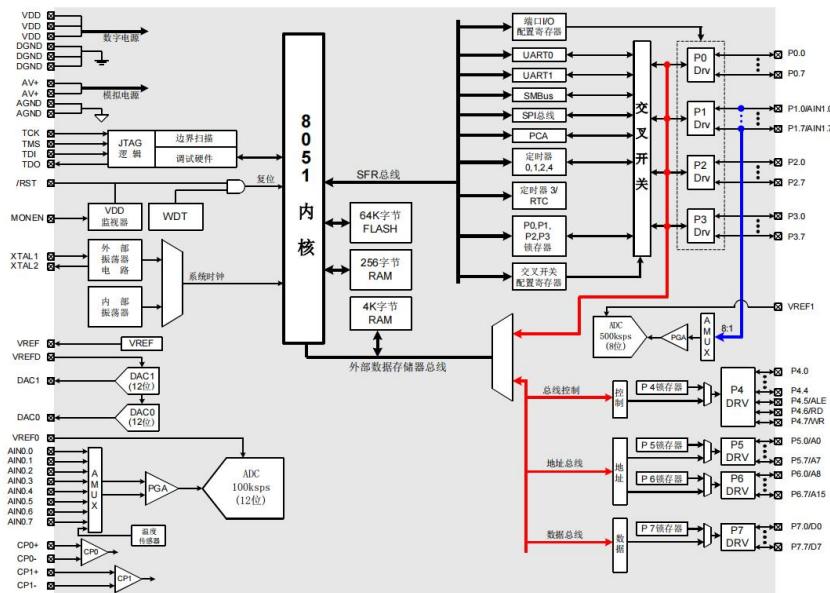


图 3-3 C8051F020 智能控制器硬件原理框图

3.1.2 C8051F020 单片机特性

- ① 高速、流水线结构的 8051 兼容的 CIP-51 内核。
 - ② 全速、非侵入式的在系统调试接口 JTAG。
 - ③ 64K 字节可在系统编程的 FLASH 存储器。
 - ④ 4352 (4096+256) 字节的片内 RAM。
 - ⑤ 可寻址 64K 字节地址空间的外部数据存储器接口。
 - ⑥ 真正 12 位 100ksps 的 8 通道 ADC0，带 PGA 和模拟多路开关。
 - ⑦ 真正 8 位 500ksps 的 ADC1，带 PGA 和 8 通道模拟多路开关。
 - ⑧ 两个 12 位 DAC，具有可编程数据更新方式。54
 - ⑨ 硬件实现的 SPI、SMBus/I2C 和两个 UART 串行接口。
 - ⑩ 5 个通用的 16 位定时器。
 - ⑪ 具有 5 个捕捉/比较模块的可编程计数器/定时器阵列 (PCA)。
 - ⑫ 片内看门狗定时器、VDD 监视器和温度传感器。
 - ⑬ 具有片内 VDD 监视器、看门狗定时器和时钟振荡器的
- C8051F020 是真正能独立工作的片上系统。

3.1.3 C8051F020 引脚说明

基于 C8051F020 单片机的特性，智能控制器在设计时，选取该单片机作为控制、计算、显示的核心部件。C8051F020 单片机有 100 个引脚，封装为 TQFP-100，其引脚如图 3-4 所示。C8051F020 单片机低端口 (P0、P1、P2、P3) 既可以按位寻址，也可以按字节寻址，高端口 (P4、P5、P6、P7) 只能按字节寻址，所有引脚都可以被配置为开漏或推挽输出方式。C8051F020 单片机有大量的数字资源需要通过 P0、

P1、P2 和 P3 端口才能使用。P0、P1、P2 和 P3 中的每个引脚即可定义为通用的 I/O 端口引脚，也可以分配给一个数字外设或功能（例如：UART0 或 INT1）。这种资源分配的灵活性是通过使用优先权交叉开关实现的。

交叉开关按优先权顺序将端口 P0、P1、P2、P3 引脚分配给单片机的数字外设（UART、SMBus、PCA、定时器等），端口引脚的分配顺序是从 P0.0 开始可以一直分配到 P3.7。当交叉开关配置寄存器 XBR0、XBR1 和 XBR2 中外设的对应允许位被设置为逻辑 ‘1’ 时，交叉开关将端口引脚分配给外设。因为 UART0 有最高优先权，所以当 UART0EN 位被设置为逻辑 ‘1’ 时，其引脚将总是被分配到 P0.0 和 P0.1。被交叉开关分配的端口引脚输出状态受使用数字外设的控制，向端口寄存器（或相关端口位）写入时，对引脚的状态没有影响，但在执行读-修改-写的读周期，所读的值是端口数据寄存器的内容，而不是端口引脚的状态。因为交叉开关寄存器影响外设的引脚，所以在外设被配置前，由系统的端口初始化代码配置。一旦端口在初始化时进行了交叉开关配置，则在程序运行过程中，不再对其重新编程。

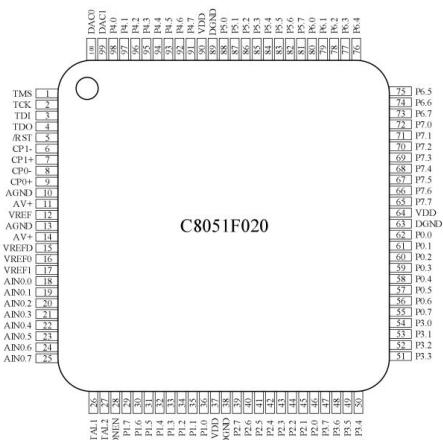


图 3-4 C8051F020 引脚图

3.2 LCD 显示电路

图 3-5 液晶显示电路中，智能控制器选用 HS12864-15B 汉字图形型液晶，带中文字库。液晶显示采用串口通信模式，可以显示字母、数字符号、中文字型及图形，具有绘图及文字画面混合显示功能。该液晶共有 20 个引脚，E、RW、RS 分别接在单片机的 P1.3、P1.4、P1.5 引脚，引脚说明如图 3-6 所示，没有列出的引脚是空接状态。P1.3、P1.4、P1.5 引脚在系统端口初始化时被设置为推挽模式，一旦端口在初始化时进行了交叉开关配置，则在程序运行过程中，端口不能进行修改。RW 引脚（P1.4）在端口初始化时被配置成推挽输出，则不能读取 LCD 返回的数据。如果需要液晶显示温度变化曲线，软件编程时设置一个“虚拟屏幕”，对数据处理后，再刷新到真实的液晶屏显示。

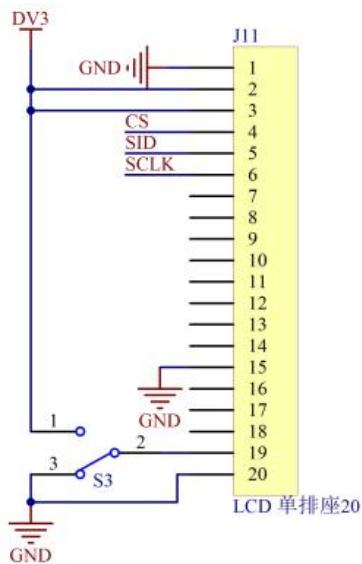


图 3-5 液晶显示电路

管脚号	名称	电平	功能
1	GND	0V	电源地
2	VCC	+5V	模块电源输入
3	VO	-	对比度（亮度）调整
4	RS	H/L	片选端，高电平有效
5	RW	H/L	串行数据线
6	E	H/L	串行时钟输入
15	PSB	L	L:串口方式
17	RST	H/L	复位端，低电平有效
19	A	VDD	背光电压
20	K	GND	背光源负端0V

图 3-6 12864-15B 液晶引脚说明

3.2.1 字符显示 RAM (DDRAM)

HS12864-15B 液晶的控制 IC 为 ST7920。ST7920 内置 2M 位中文字

型 ROM(CGROM)，总共提供 8192 个中文字型(16x16 点阵)，16K 位半宽字型 ROM(HCGROM)，总共提供 126 个符号字型(16x8 点阵)，64x16 位字型产生 RAM(CGRAM)，用来用户自定义字型。显示中文字符时将 16 位数据送入 DDRAM 中，先写高 8 位(D16~D8)，再写低 8 位(D7~D0)，可显示 4 行，每行显示 8 个汉字，共显示 32 个汉字。DDRAM 在液晶模块中的地址为 80H~9FH，字符显示的 RAM 地址与字符在屏幕上的显示区域是一一对应的关系，如图 3-7 所示。

	列 1	列 2	列 3	列 4	列 5	列 6	列 7	列 8
行 1	80H	81H	82H	83H	84H	85H	86H	87H
行 2	90H	91H	92H	93H	94H	95H	96H	97H
行 3	88H	89H	8AH	8BH	8CH	8DH	8EH	8FH
行 4	98H	99H	9AH	9BH	9CH	9DH	9EH	9FH

图 3-7 字符显示的 DDRAM 地址与显示区域对应关系

3.2.2 绘图 RAM (GDRAM)

绘图 GDRAM 由扩充指令进行设置。横坐标将 128 点分为 16 点一列，共 8 列，纵坐标将 64 点分为 64 行。写入数据时，先写入垂直地址，再写入水平地址，最后连续写入两字节 8bit 数据，先高 8 位，后低 8 位。在编程显示图片时，先将其调整成合适尺寸的图片，然后通过取模软件对图片取模，保存成点阵数组。显示时，数据以长度为[1024]大小的数组形式送入 GDRAM，数组内每个元素为一个字节，即 8 位二进制数。

3.3 按键电路

智能控制器的按键电路，如图 3-8 所示。三个按键信号 A8、A9、A10 分别接在 C5051F020 单片机的 P5.0、P5.1、P5.2 引脚，中断信

号 INT1 接在 P0.3 引脚。按键由外部中断信号触发，低电平有效，按键按下触发中断，进入按键中断服务程序，完成一定功能后再回到中断前正在执行的程序。

INT1 中断信号和数码管 P5 的低三位复用，假设某一按键按下，P5 对应的位置为逻辑 ‘0’，此时这条线路形成通路，才能将低电平信号和 INT1 连通。如果在按键扫描时，让 P5 的三个端口状态轮流为逻辑 ‘0’，读取 P5 端口的值，就可以建立 P5 的端口和三个按键一一对应的关系。共阳极数码管显示的位选是轮流使能的过程，故将数码管的位选端口与 INT1 的端口复用。

INT1 端口对应的 P0.3 端口，在应用前需要由交叉开关进行设置。交叉开关是 C8051F020 中端口配置的特色，可通过交叉开关寄存器来控制端口为普通的数据输入输出端口，是具有特定功能的端口。P0、P1 等的全部或部分端口被交叉开关配置成推挽模式，保证正常输出。

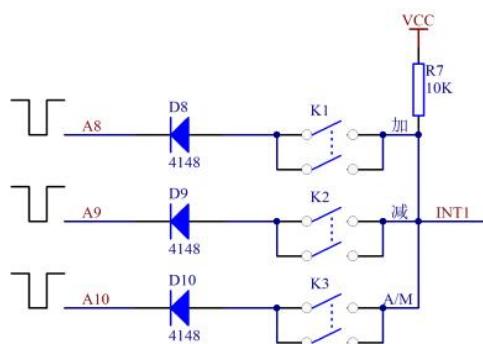


图 3-8 按键电路

3.4 LED 显示电路

智能控制器设置了三组四位数码管，其中一组数码管的显示电路，如图 3-9 所示。三组数码管的段选信号 Q0~Q7，通过 74HC245 分别接在单片机的 P7.0~P7.7 引脚，三组共 12 位数码管的位选信号

LED21~LED24、LED25~LED28、LED29~LED32，通过7407驱动器分别接在单片机的P5.0~P5.3、P5.4~P5.7、P6.0~P6.3引脚。数码管采用动态扫描显示方式，数码管为共阳极接法，位选信号为逻辑‘0’表示该位对应的数码管被选中，数码管显示内容由段选信号决定，利用余辉效应可以分时复用P7端口，来点亮特定的数码管，显示对应的数字。在实际应用中，采用两种方式可以达成余晖的效果。一种方式是在程序中设置延时，来制造余晖的效果。另一种方式是程序在一次循环中需要执行很多指令的情况下，这样即使设置了延迟，程序执行的时间往往远超过期望的延迟，则余晖的效果会变成闪烁，严重影响数码管的显示。在这种情况下，可以考虑采用定时器中断进行特定周期的触发以达成稳定显示。

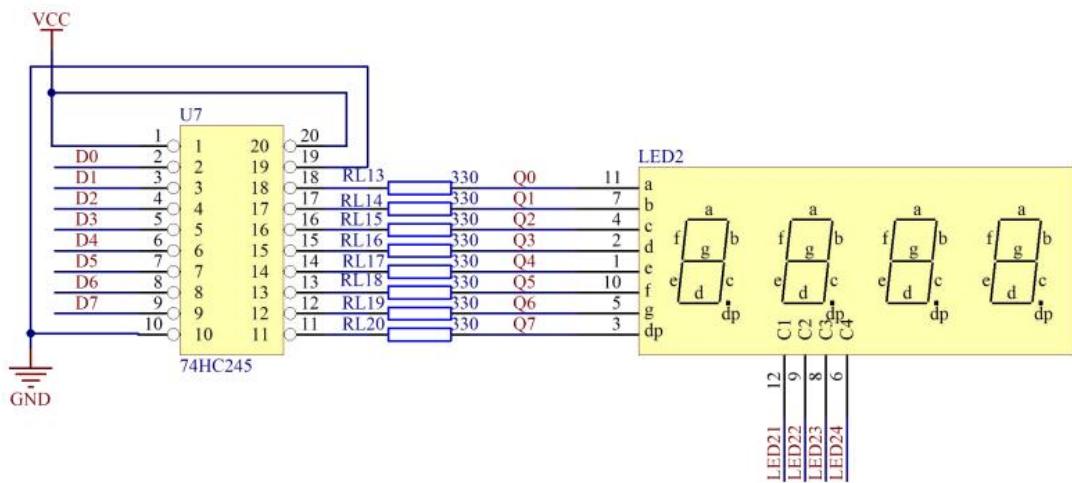


图 3-9 数码管显示电路

3.5 直升机垂直升降模拟对象

3.5.1 原理图

直升机垂直升降模拟对象系统原理图如图3-10所示，实物如图3-11所示，接口说明如图3-12所示。

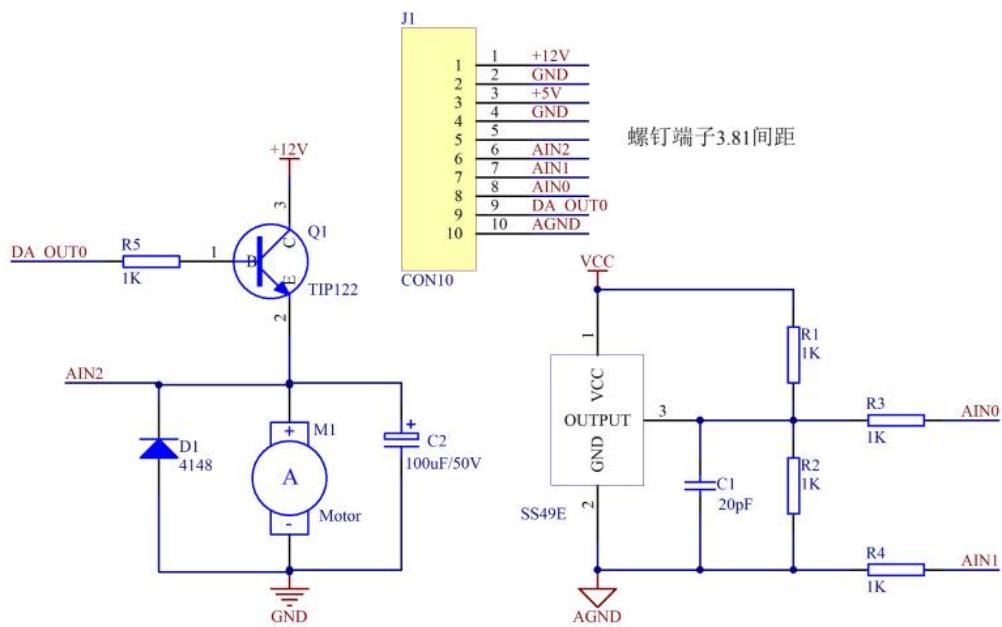


图 3-10 直升机垂直升降模拟对象系统原理图



图 3-11 直升机垂直升降模拟对象实物图

端子号	端子名称	功能
1	+12V	+12V电源
2	GND	数字地
3	+5V	+5V电源
4	GND	数字地
5	AIN3	空接
6	AIN2	空接
7	AIN1	空接
8	AIN0	霍尔传感器电路输出信号检测端子
9	DA_OUT	模拟量控制信号端子
10	AGND	模拟地

图 3-12 直升机垂直升降模拟对象接口说明

3.5.2 SS49E 线性霍尔效应传感器

特性：SS49E 线性霍尔传感器具有体积小，用途广泛等特点。SS49E 可由永磁体或电磁铁进行操作，电源电压控制线性输出，可根据磁场强度的不同做成线性变化。SS49E 内部集成了低噪声输出电路，省去了外部滤波器的使用。器件包含了薄膜电阻，增加了温度的稳定性和精度。SS49E 的工作电压为 4.5V~6V。

引脚：SS49E 线性霍尔传感器引脚说明如图 3-13 所示。

SOT 引脚	SPI 引脚	引脚名称	功能
1	1	VDD	电源引脚
2	3	OUT	开漏输出引脚
3	2	GND	地引脚

图 3-13 SS49E 线性霍尔传感器引脚说明

极限参数：SS49E 线性霍尔传感器极限参数如图 3-14 所示。

参数	符号	数值	单位
电源电压（工作状态）	VCC	8.0	V
输出电流	Iout	20	mA
工作温度范围	TA	-40~150	°C
储存温度	TS	-65~150	°C

图 3-14 SS49E 线性霍尔传感器极限参数

四、软件实现

4.1 功能介绍

- 1、LCD 显示初始菜单与个人信息
- 2、按任意键进入直升机控制系统子页面，按右键回到初始界面
- 3、直升机控制系统子页面中，LCD 显示控制系统子菜单，LED 显

示电压设定值、电压测量值、控制电压值，LCD 可作为示波器使用，按键可实现更改 PID 参数设定、波形显示、回退等功能。

4.2 系统流程图

在软件设计中程序分别完成 LCD 初始化及显示、按键输入检测、倒计时运算、得分计算、局数统计、LED 显示等功能。对主程序进行初始化，其他程序选择模块化的方式实现。首先对每个模块进行调度，再逐一加入主程序中，最后完成整个软件部分的设计。

以 LCD 显示为单位，整个系统共有 8 个菜单，各菜单及其菜单下不同按键功能如下所示：

初始菜单：mainflag=0

LCD: WriteStr(0, 0, "电子线路设计实验");
WriteStr(1, 0, "自动化94 胡欣盈");
WriteStr(2, 0, "2194323176");
WriteStr(3, 0, "按任意键开始实验");

任意按键：

```
if (botflag==0) {mainflag=3;LcdClear();botflag=4;}  
if (botflag==1) {mainflag=3;LcdClear();botflag=4;}  
if (botflag==2) {mainflag=3;LcdClear();botflag=4;}  
if (mainflag==3) {page3show();}//进入实验二界面
```

系统子菜单：mainflag=3; showflag2=0

LCD: WriteStr(0, 0, "实验二");
WriteStr(1, 0, "左：示波器");
WriteStr(2, 0, "中：飞机控制");
WriteStr(3, 0, "右：返回");

按键：

```
if (botflag==2) {mainflag=0;LcdClear();botflag=4;}  
if (botflag==1) {showflag2=2;LcdInit();botflag=4;}  
if (botflag==0) {showflag2=1;LcdInit();botflag=4;}
```

示波器页面：mainflag=3; showflag2=1

LCD: WriteStr(1, 0, "按左键开始");
WriteStr(2, 0, "左：上 中：下");//设定值
WriteStr(3, 0, "右：停止&&返回");

按键：

```
if (botflag==0) {setting=setting+50;botflag=4;}  
if (botflag==1) {setting=setting-50;botflag=4;}  
if (botflag==2) {showflag2=0;LcdInit();botflag=4;}
```

LED：显示设定值与控制量

控制子菜单: mainflag=3; showflag2=2

LCD: WriteStr(0, 0, "滤波器");
WriteStr(2, 0, "左:next 中: 控制");
WriteStr(3, 0, "右: 返回");
LedDispNum();
switch(filtertype2)
{
 case 0:
 WriteStr(1, 0, "算法平均值滤波 ");
 break;
 case 1:
 WriteStr(1, 0, "中位数平均滤波 ");
 break;
 case 2:
 WriteStr(1, 0, "滑动平均值滤波 ");
 break;
}
按键: if (botflag==0) [filtertype2=(filtertype2+1)%3;botflag=4;]
if (botflag==1) [showflag2=3;LcdInit();ImageShow(gImage_white);botflag=4;]
if (botflag==2) [showflag2=0;LcdInit();botflag=4;]

(4)

PID参数菜单: mainflag=3; showflag2=3

LCD: WriteStr(0, 0, "Kp: ");
WriteStr(1, 0, "Ki: ");
WriteStr(2, 0, "Kd: ");
WriteStr(3, 0, "左:Next 中: 确定");
WriteStr(0, 4, ckp);
WriteStr(1, 4, cki);
WriteStr(2, 4, ckd);
按键: if (botflag==0) {paratype=(paratype+1)%3;botflag=4;}
if (botflag==1) {showflag2=paratype+30;LcdInit();botflag=4;}
if (botflag==2) {showflag2=0;LcdInit();botflag=4;}

(5)

P参数控制菜单: mainflag=3; showflag2=30

LCD: WriteStr(0, 0, "Kp: ");
WriteStr(1, 0, "左: 加 ");
WriteStr(2, 0, "中: 减 ");
WriteStr(3, 0, "右: 返回 ");
WriteStr(0, 4, ckp);
按键: if (botflag==0) {kp=kp+0.1;ckp[0]=ckp[0]+1;WriteStr(0, 4, ckp);botflag=4;}
if (botflag==1) {kp=kp-0.1;ckp[0]=ckp[0]-1;WriteStr(0, 4, ckp);botflag=4;}
if (botflag==2) {showflag2=3;LcdInit();botflag=4;}

(6)

I参数控制菜单: mainflag=3; showflag2=31

LCD: WriteStr(0, 0, "Ki: ");
WriteStr(1, 0, "左: 加 ");
WriteStr(2, 0, "中: 减 ");
WriteStr(3, 0, "右: 返回 ");
WriteStr(0, 4, cki);
按键: if (botflag==0) {ki=ki+0.01;cki[3]=cki[3]+1;WriteStr(0, 4, cki);botflag=4;}
if (botflag==1) {ki=ki-0.01;cki[3]=cki[3]-1;WriteStr(0, 4, cki);botflag=4;}
if (botflag==2) {showflag2=3;LcdInit();botflag=4;}

(7)

D参数控制菜单: mainflag=3; showflag2=32

LCD: WriteStr(0, 0, "Kd: ");
WriteStr(1, 0, "左: 加 ");
WriteStr(2, 0, "中: 减 ");
WriteStr(3, 0, "右: 返回 ");
WriteStr(0, 4, ckd);
按键: if (botflag==0) {kd=kd+0.001;ckd[4]=ckd[4]+1;WriteStr(0, 4, ckd);botflag=4;}
if (botflag==1) {kd=kd-0.001;ckd[4]=ckd[4]-1;WriteStr(0, 4, ckd);botflag=4;}
if (botflag==2) {showflag2=3;LcdInit();botflag=4;}

(8)

各个菜单之间的转变与按键控制内容如下图 4-1 所示：

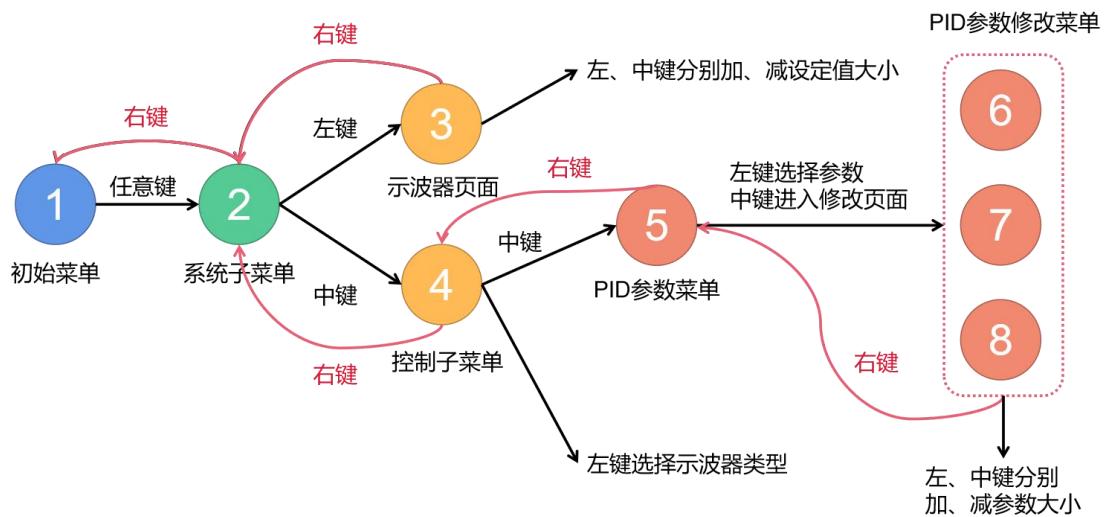


图 4-1 系统流程图

4.3 主要函数代码

本设计程序采用 C 语言编程，程序分为多个模块，由主程序、LED 显示和 LCD 系列程序等模块组成。具体程序见附件，此处简要说明各模块函数功能。

4.3.1 LED 显示

在主程序文件中定义了 12 个变量 (w1、x1、y1、z1、w2、x2、y2、z2、w3、x3、y3、z3)， wxyz 分别表示每个 LED 的 4 位数值，下标指示是第几个 LED 的显示数值。3 个 LED 分别记录滤波器输出、设定值、控制量。在 led.c 文件中外部引用这三个变量，并使三组 LED 数码管分别显示其数值即可。通过修改例程中的 LedDispNum() 函数可实现该功能，即将 3 个 4 位数分别存在对应的 3 个 4 位数组中，通过片选将数组中的每一个数赋予对应的数码管，使数码管在主程序中不断刷新即可实现 3 组 LED 显示。

4.3.2 按键中断

在主程序文件中定义了按键中断对应的函数 INT1_ISR(void) 如图 4-2 所示，通过 botflag 的数值指示左、中、右三大按键，每次读取按键中断后都及时将 botflag 置为 4 以保证程序正常运行。

```
void INT1_ISR(void) interrupt 2
{
    Delay_ms(1);
    switch(P5)
    {
        case 0xfb: //左边那个键
            botflag=0;
            Delay_ms(50);
            break;

        case 0xfd: //中间那个键
            botflag=1;
            Delay_ms(50);
            break;

        case 0xfe: //右边那个键
            botflag=2;
            Delay_ms(50);
            break;
    }
}
```

图 4-2 按键中断

4.3.3 LCD 显示

LCD 字符显示与例程基本相同，通过 WriteStr (uchar row, uchar col, uchar *puts) 函数可实现不同的 LCD 显示。切换菜单时通过 LcdInit() 清屏。

LCD 显示波形则通过 LcdShowPoint (unsigned char x) 函数实现，函数内容如图 4-3 所示。首先利用 ADC0 中断服务实现模数转换，得到测量值，并将其存入变量 ADC0_result 中，通过计算得到采样点位置并将其传入 LcdShowPoint 函数中，即可在 LCD 上画出点阵图。

```

void LcdShowPoint(unsigned char x)//lcd画电压点图
{
    unsigned char i;
    unsigned char col=x/16;
    unsigned char off=x%16;
    unsigned char row=wavevalue[x]/128;
    //unsigned char row=wavevalue[x]/157;

    unsigned char datah=0;
    unsigned char data1=0;

    for(i=0;i<8;i++)
    {
        if(i<=off&&wavevalue[col*16+i]/128==row) datah|=0x80>>i;
        if(i+8<=off&&wavevalue[col*16+8+i]/128==row) data1|=0x80>>i;
    }

    WriteCommand(0x34);
    WriteCommand(0x80+31-row);
    WriteCommand(0x80+col);
    WriteCommand(0x30);
    WriteData(datah);
    WriteData(data1);
    WriteCommand(0x32);
    WriteCommand(0x36);
}

```

图 4-3 电压点阵图实现函数

4. 3. 4 PID 控制算法

根据实验指导书中介绍的 PID 控制原理，采用位置式 PID 控制算法，可写出如图 4-4 所示算法以实现 PID 控制。

```

//控制算法
error=settings-result2;
sumerror+=error;
derror=lasterror-preerror;
preerror=lasterror;
lasterror=error;
output=result2+kp*error+ki*sumerror+kd*derror;

```

图 4-4 PID 算法

4. 3. 5 滤波器设计

考虑到实验室环境较为嘈杂，为了更好的测量霍尔电压来进行控制，设计了 3 种不同的滤波器对采样数据进行处理。在实际处理时可根据需要通过按键中断进行滤波器的选择。

(1) 算术平均滤波

连续取 N 个采样值进行平均运算。N 值较大时，信号平滑度较高，但灵敏度较低；N 值较小时，信号平滑度较低，但灵敏度较高。N 值选 12 左右。具体算法实现如图 4-5 所示。

它适应于对一般具有随机干扰的信号进行滤波，这样信号的特点是有一个平均值，信号在某一数值范围附近上下波动。但对于测量速度较慢或要求数据计算速度较快的实时控制并不适用，比较浪费 RAM。

```
void Average_filter(void)//算术平均滤波
{
    long i = 0, t = 0;
    long sum = 0;
    for ( i = 0 ; i < 10 ; i++ )
    {
        //sum = sum + (int)(ADC0_result[1]-270)*1.101;
        sum = sum + v[i];
    }
    t = sum /10;
    w1 = t*5/4096;
    x1 = (t*50)/4096-w1*10;
    y1 = (t*500)/4096-w1*100-x1*10;
    z1 = (t*5000)/4096-w1*1000-x1*100-y1*10;
    /*w1 = t/1000;
    x1 = (t - w1*1000)/100;
    y1 = (t - w1*1000-x1*100)/10;
    z1 = t - w1*1000-x1*100-y1*10; */
    result=(int)(t);
    result2=t*5000/4096;
    //return t;
}
```

图 4-5 算术平均滤波算法

(2) 中位值平均滤波

采样 N 个数据，去掉一个最大值和一个最小值，然后计算 N-2 个数据的算术平均值。N 值选 3-14 左右，对于偶然出现的脉冲性干扰，可消除由于脉冲干扰所引起的采样值偏差。但测量速度较慢，和算法平均滤波一样，浪费 RAM。具体算法实现如图 4-6 所示。

```

void Median_average_filter(void)//中值平均滤波
{
    long i,sum, average,max,min;
    average = 0;
    max = v[0];
    min = max;
    sum = 0;
    for ( i = 0; i<10;i++)
    {
        sum = sum + v[i];
        if(v[i] > max)
        {
            max = v[i];
        }
        if(v[i] < min)
        {
            min = v[i];
        }
    }
    sum = sum - max -min;
    average = sum /8 ;
    w1 = average*5/4096;
    x1 = (average*50)/4096-w1*10;
    y1 = (average*500)/4096-w1*100-x1*10;
    z1 = (average*5000)/4096-w1*1000-x1*100-y1*10;
    /*w1 = average/1000;
    x1 = (average-w1*1000)/100;
    y1 = (average - w1*1000-x1*100)/10;
    z1 = average - w1*1000-x1*100-y1*10; */
    result=(int)(average);
    result2=average*5000/4096;
    //return average;
}

```

图 4-6 中位值平均滤波算法

(3) 滑动平均滤波

把连续取 N 个采样值看成一个队列，队列的长度固定为 N，每次采样到一个新数据放入队尾，并扔掉原来队首的一次数据(先进先出)。把队列中的 N 个数据进行算术平均运算，就可获得新的滤波结果。N 值一般选 12.

对周期性干扰有良好的抑制作用，平滑度高，适应于高频振荡的系统。但灵敏度低，对偶然出现的脉冲性干扰的抑制作用较差。不易消除由于脉冲干扰所引起打的采样值偏差，不适用于脉冲干扰比较严重的场合浪费 RAM，具体算法实现如图 4-7 所示。

```

void Sliding_filter(void)//滑动平均滤波
{
    long s;
    // int count;
    // int sum=0;
    // i++;
    // v[i] = (int)(ADC0_result[1]-270)*1.101;
    // if ( i == 12 ) i = 0;
    // for ( count = 1;count < 13;count++) sum = sum + v[count];

    slidesum-=v[samplepos];
    slidesum+=v[(samplepos+9)%10];
    s = slidesum /9;
    //if(s<0) s=0;
    if(s>4095) s=4095;
    //s = s/4096*5000;
    w1 = s*5/4096;
    x1 = (s*50)/4096-w1*10;
    y1 = (s*500)/4096-w1*100-x1*10;
    z1 = (s*5000)/4096-w1*1000-x1*100-y1*10;
    /*w1 = s/1000;
    x1 = (s-w1*1000)/100;
    y1 = (s - w1*1000-x1*100)/10;
    z1 = s - w1*1000-x1*100-y1*10;*/
    result=(int)(s);
    result2=s*5000/4096;
    //return s;
}

```

图 4-7 滑动平均滤波

五、结果展示与分析

系统初始化后在 LCD 屏幕上显示主菜单，内含个人信息与控制指引，如图 5-1 所示，按任意键进入系统子菜单如图 5-1 所示。



图 5-1 主菜单

图 5-2 系统子菜单

在系统子菜单下首先按中键进行飞机控制，进入控制子菜单，在此页面上按左键进行滤波器选择，选好的滤波器会显示在 LCD 上，如图 5-3、5-4、5-5 所示。

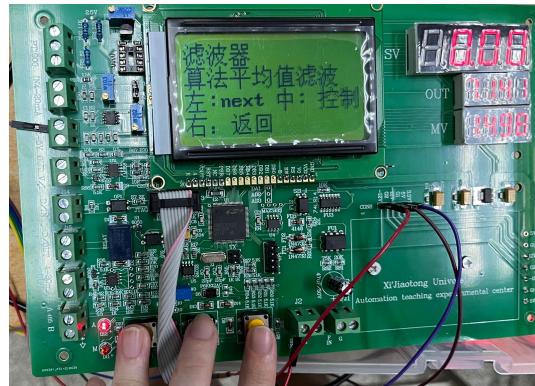


图 5-3 控制子菜单 (1)



图 5-4 控制子菜单 (2)



图 5-5 控制子菜单 (3)

确认选择后，按中键进行 PID 参数页面，该页面显示了当前 PID 参数。如图 5-6 所示。若要改变 PID 参数，则可通过左键选中需要改变的参数，首次打开时默认当前选项为 P。选中参数后按右键进入参数修改页面如图 5-7、5-8、5-9 所示。

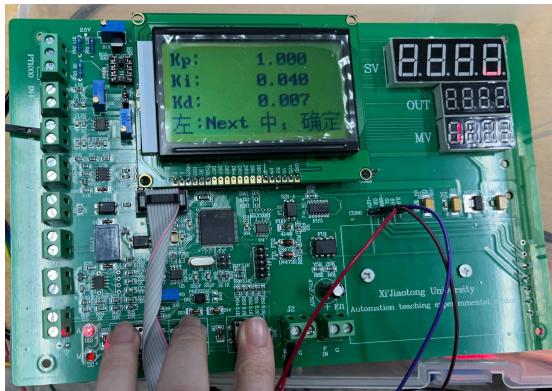


图 5-6 PID 参数页面

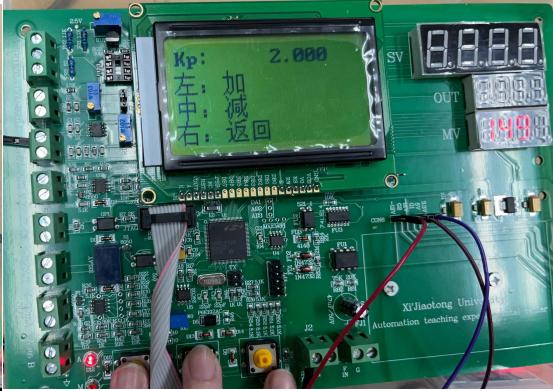


图 5-7 P 修改页面

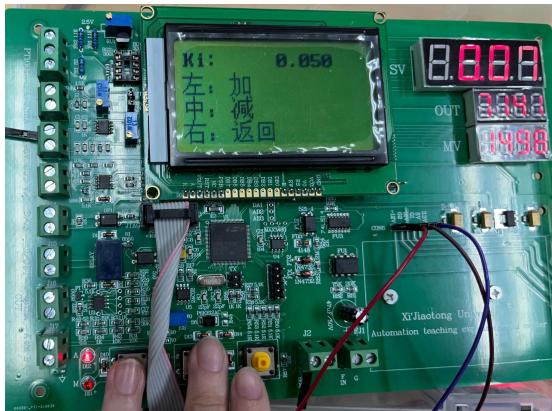


图 5-8 I 修改页面

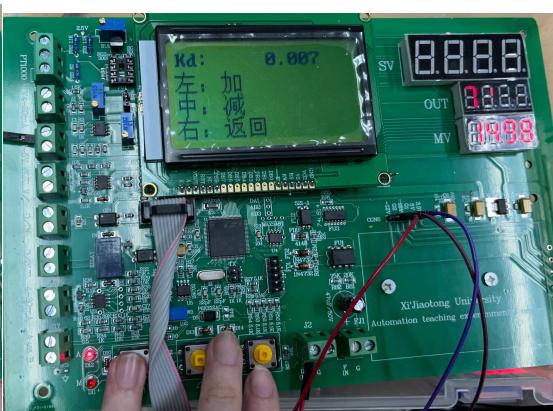


图 5-9 D 修改页面

在参数修改页面按左、中键可控制参数的加、减，改完后按右键返回，连续返回到系统子菜单后按左键可通过示波器观察电压输出波形，如图 5-10 所示。此时 LED 分别显示滤波器计算后的采样值、设定值、控制量，按左、中键可控制设定值加、减 0.5，观察波形变化。

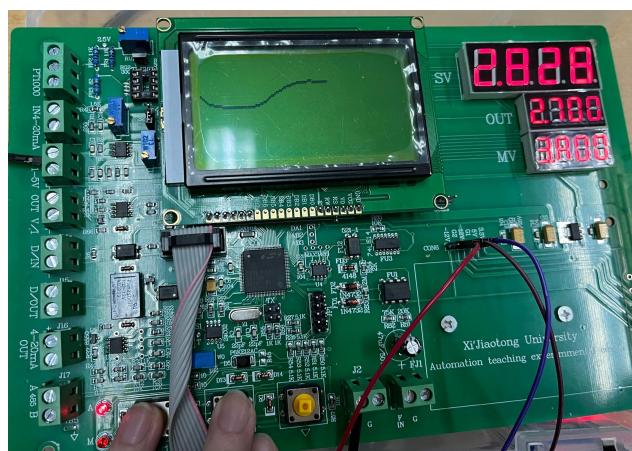


图 5-10 LCD 示波器

六、实验总结

通过本次实验，很大程度的提高了我的理解力和动手能力，也拓宽了知识面。在图书馆和网上查阅资料使我了解了更多的资料，也方便了我们以后的使用。在查询的资料中让我对单片机与控制系统有了更深入的了解，能把学到的知识用活，而不只局限于理论方面。

学习例程让我对整个系统运行与引脚定义有了大致的了解，在自己进行系统编程时，我也学习了例程的方法，充分使用了结构化的思想。这样一来，程序调试也更方便，功能模块可以逐一地调试，充分体现了结构化编程的优势。当每个模块都完成时，将其功能互相整合就完成了整体的设计。

经历了从最初的学习例程到设计实现完整的 PID 控制系统的过程，我从根本上提高了对专业的认识及兴趣，对于我们工科学生来说，学习这些对我们以后的工作有巨大帮助。同时也感谢刘老师在实验中的悉心指导与耐心讲解，让我获益匪浅，也在实践中得以学习与成长。

七、参考文献

[1] 刘美兰, 刘瑞玲, 刘源等, 电子线路设计训练实验教程. 西安交通大学出版社.

[2] 杨国林, c 语言程序设计[J]. 内蒙古大学出版社. 2001. 9

[3] 郭天样, 新概念 51 单片机 C 语言程宁 [J]. 电子工业出版社
-2009. 1