学号：53160831　　姓名：张志帆

实验五 重量测量

一、实验目的和要求

掌握点阵式液晶显示屏的原理和控制方法，掌握点阵字符的显示方法。

掌握模拟/数字（A/D）转换方式，

进一步掌握使用 C51 语言编写程序的方法，使用 C51 语言编写实现重量测量的功能。

二、实验原理

1. 参考附录六，学习点阵式液晶显示屏的控制方法。

2. 在液晶显示中，自定义图形和文字的字模对应的字节表需要使用专门的字模软件来生成。可以使用 PCtoLCD2002 字模软件提取。

3. 字符点阵等数据，需要定义在 code 数据段中，具体原理参见示例程序设计部分。

4. 向 LCM 输出一个命令或数据时，应当在选通信号为高时准备好数据，然后延迟若干指令周期，再将选通信号置为低

5 与 A/D 转换相关的寄存器

ADC_POWER：ADC 电源控制位，0 关 1 开。

SPEED1,SPEED0：模数转换器速度控制位，控制 A/D 转换所需时间。

ADC_FLAG：模数转换结束标志位，AD 转换完后，ADC_FLAG=1，一定要软件清 0。

ADC_START：模数转换器（ADC）转换启动控制位，1 开始转换，转换结束后为 0。

CHS2/CHS1/CHS0：模拟输入通道选择，选择使用 P1.0~P1.7 作为 A/D 输入

6. 重量传感器采用压敏电阻。利用压敏电阻采集应变,产生变化的阻值。利用放大电路将其转化为电压值，通过数模转换将电压值转化成 CPU 处理的数字信号。传感器根据编制的程序将数字信号转换为砝码重量显示输出

三、实验器材

单片机测控实验系统

重量测量实验板/砝码

Keil 开发环境

STC-ISP 程序下载工具

四、实验内容

参考辅助材料，学习 C51 语言使用

编写 C51 程序，使用重量测量实验板测量标准砝码的重量，将结果（以克计）显示到液晶屏上。误差可允许的范围之间

五、实验步骤

1. 阅读实验原理，掌握 YM12864C 的控制方式，编写出基本的输出命令和数据的子程序；

2. 掌握点阵字模的构成方式。使用字模软件 PCtoLCD2002，设定正确的输出模式，生成点阵数据

3. 使用 C51 语言编写重量测量程序；

4. 调零，满量程校准；

5. 将编译后的程序下载到 51 单片机；

6. 在托盘中放上相应重量的法码，使显示值为正确重量

六、思考题

1.调零的原理，软件调零和硬件调零的区别。

答:

硬件调零:使用外接电路或者改变压敏电阻的初始阻止等方式实现的调零。通过附加电路或者对压敏电阻调整;

软件调零:在不适用任何外接电路的情况下，对采集的数据进行数学处理从而实现调零的过程。

使用软件对 A/D 采集值进行调整来达到在未放置物体的时候显示 0。

2.A/D 和 D/A 信号的转换原理。

答:A/D 逐次逼近法:由-个比较器、D/A 转换器、缓冲寄存器及控制逻辑电路组成。初始化时将逐次逼近寄存器各位清零;

转换开始时，先将逐次逼近寄存器最高位置 1,送入 D/A 转换器，

经 D/A 转换后生成的模拟量送入比较器，称为 Vo,与送入比较器的待转换的模拟量 Vi 进行比较，若 Vo<Vi,该位 1 被保留，否则被清除。然后再置逐次逼近寄存器次高位为 1,

将寄存器中新的数字量送 D/A 转换器，输出的 Vo 再与 Vi 比较，若 Vo<Vi,该位 1 被保留，否则被清除。

重复此过程，直至逼近寄存器最低位。

转换结束后，将逐次逼近寄存器中的数字量送入缓冲寄存器，得到数字量的输出。D/A 转换:将二进制数的每位按权大小转换为相对应的模拟量， 然后将代表的各位的模拟量想家，就得到对应数字量对应的模拟量。

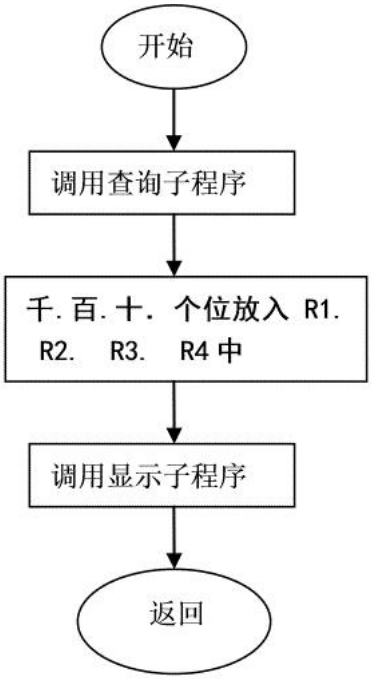
3.I²C 总线在信号通讯过程中的应用。

答:主器件用于启动总线传送数据，并产生时钟以开放传送的器件，此时任何被寻址的器件均被认为是从器件.在总线上主和从、发和收的关系不是恒定的，而取决于此时数据传送方向。如果主机要发送数据给从器件，则主机首先寻址从器件，然后主动发送数据至从器件，最后由主机终止数据传送。

如果主机要接收从器件的数据，首先由主器件寻址从器件，然后主机接收从器件发送的数据，最后由主机终止接收过程，在这种情况下，主机负责产生定时时钟和终止数据传送。

七、流程图

总流程图：



查询子程序流程图：

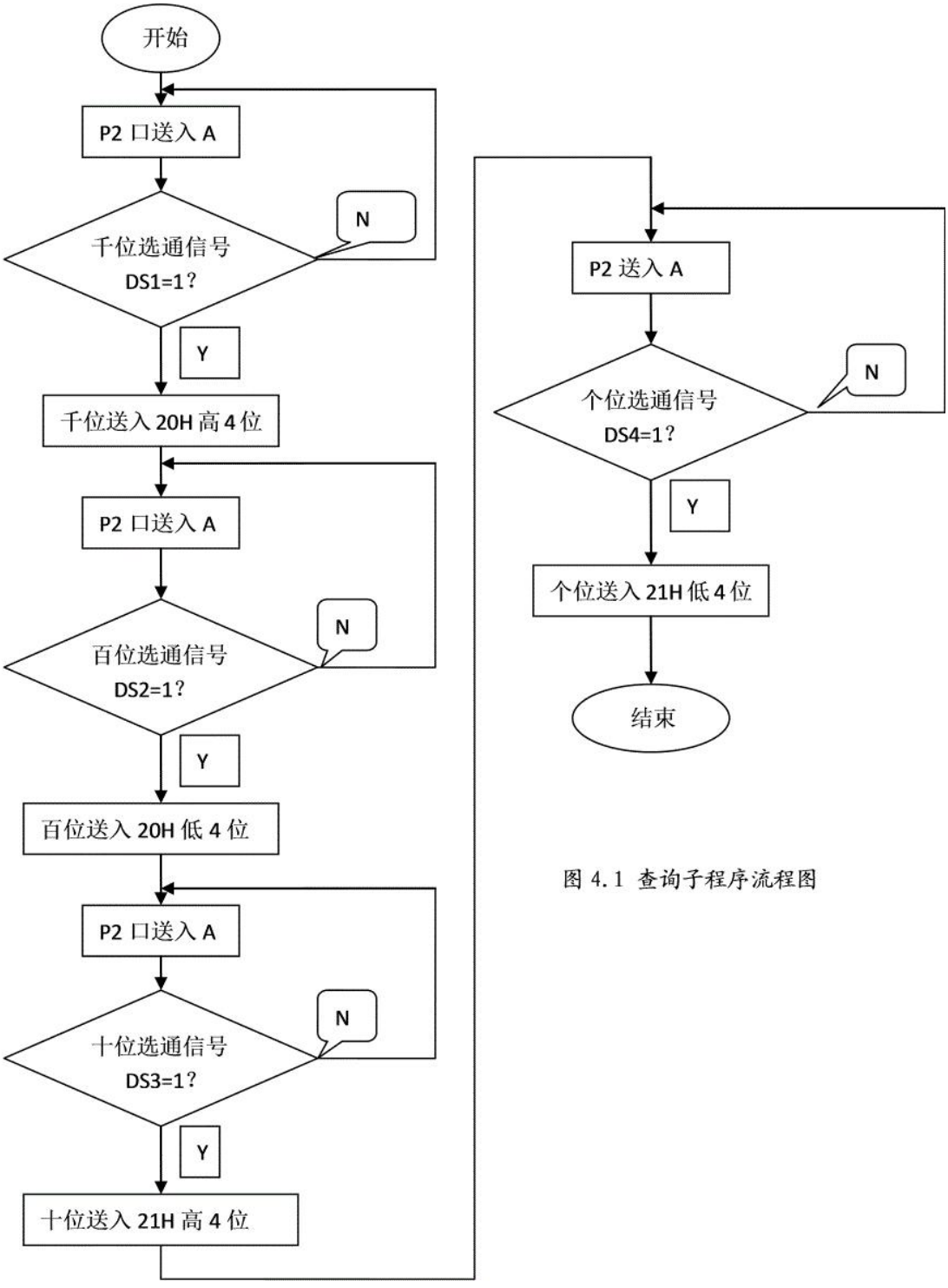


图 4.1 查询子程序流程图

代码：

```
#include <reg52.h>
#include <intrins.h>

#define uchar unsigned char
#define uint unsigned int
```

```
    sbit CS1=P1^7;///左半边
    sbit CS2=P1^6;///右半边
    sbit E=P3^3;///使能信号
    sbit RW=P3^4;///读写操作选择
    sbit RS=P3^5;///寄存器选择(数据/指令)
    sbit RES=P1^5;///复位 低电平有效
    sbit BUSY=P2^7;

    /**Declare SFR associated with the ADC */
    sfr ADC_CONTR = 0xBC;  ///ADC control register
    sfr ADC_RES = 0xBD;  ///ADC hight 8-bit result register
    sfr ADC_LOW2 = 0xBE;  ///ADC low 2-bit result register
    sfr P1ASF = 0x9D;  ///P1 secondary function control register
    sfr AURX1 = 0xA2; ///AURX1 中的 ADRJ 位用于转换结果寄存器的数据格式调整控制

    /**Define ADC operation const for ADC_CONTR*/
    #define ADC_POWER 0x80  ///ADC power control bit
    #define ADC_FLAG 0x10  ///ADC complete flag
    #define ADC_START 0x08  ///ADC start control bit
    #define ADC_SPEEDLL 0x00  ///540 clocks
    #define ADC_SPEEDL 0x20  ///360 clocks
    #define ADC_SPEEDH 0x40  ///180 clocks
    #define ADC_SPEEDHH 0x60  ///90 clocks

    uchar ch = 0;  ///ADC channel NO.0


    uchar code zima[20][32]=
    {
    0x00,0x00,0xC0,0xE0,0x30,0x10,0x08,0x08,0x08,0x08,0x08,0x18,0x30,0xE0,0xC0,
0x00,
    0x00,0x00,0x07,0x0F,0x18,0x10,0x20,0x20,0x20,0x20,0x20,0x10,0x18,0x0F,0x07,
0x00,///*"0"*0/

    0x00,0x00,0x00,0x10,0x10,0x10,0x10,0xF0,0xF8,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,
    0x00,0x00,0x00,0x20,0x20,0x20,0x20,0x3F,0x3F,0x20,0x20,0x20,0x20,0x00,0x00,
0x00,///*"1"*1/

    0x00,0x00,0x60,0x50,0x10,0x08,0x08,0x08,0x08,0x08,0x08,0x98,0xF0,0x70,0x00,
0x00,
    0x00,0x00,0x20,0x30,0x28,0x28,0x24,0x24,0x22,0x22,0x21,0x20,0x30,0x18,0x00,
0x00,///*"2"*2/
```

0x00, 0x00, 0x30, 0x30, 0x08, 0x08, 0x88, 0x88, 0x88, 0x88, 0x58, 0x70, 0x30, 0x00, 0x00, 0x00,
0x00, 0x00, 0x18, 0x18, 0x20, 0x20, 0x20, 0x20, 0x20, 0x20, 0x31, 0x11, 0x1F, 0x0E, 0x00, 0x00, ///*"3"*3/

0x00, 0x00, 0x00, 0x00, 0x00, 0x80, 0x40, 0x20, 0x10, 0xF0, 0xF8, 0xF8, 0x00, 0x00, 0x00, 0x00,
0x00, 0x04, 0x06, 0x05, 0x05, 0x04, 0x24, 0x24, 0x24, 0x3F, 0x3F, 0x3F, 0x24, 0x24, 0x24, 0x00, ///*"4"*4/

0x00, 0x00, 0x00, 0xC0, 0x38, 0x88, 0x88, 0x88, 0x88, 0x88, 0x88, 0x88, 0x08, 0x08, 0x00, 0x00,
0x00, 0x00, 0x18, 0x29, 0x21, 0x20, 0x20, 0x20, 0x20, 0x20, 0x30, 0x11, 0x1F, 0x0E, 0x00, 0x00, ///*"5"*5/

0x00, 0x00, 0x80, 0xE0, 0x30, 0x10, 0x98, 0x88, 0x88, 0x88, 0x88, 0x88, 0x98, 0x10, 0x00, 0x00,
0x00, 0x00, 0x07, 0x0F, 0x19, 0x31, 0x20, 0x20, 0x20, 0x20, 0x20, 0x20, 0x11, 0x1F, 0x0E, 0x00, ///*"6"*6/

0x00, 0x00, 0x30, 0x18, 0x08, 0x08, 0x08, 0x08, 0x08, 0x88, 0x48, 0x28, 0x18, 0x08, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x38, 0x3E, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, ///*"7"*7/

0x00, 0x00, 0x70, 0x70, 0xD8, 0x88, 0x88, 0x08, 0x08, 0x08, 0x08, 0x98, 0x70, 0x70, 0x00, 0x00,
0x00, 0x0C, 0x1E, 0x12, 0x21, 0x21, 0x20, 0x21, 0x21, 0x21, 0x23, 0x12, 0x1E, 0x0C, 0x00, 0x00, ///*"8"*8/

0x00, 0xE0, 0xF0, 0x10, 0x08, 0x08, 0x08, 0x08, 0x08, 0x08, 0x18, 0x10, 0xF0, 0xC0, 0x00, 0x00,
0x00, 0x00, 0x11, 0x33, 0x22, 0x22, 0x22, 0x22, 0x22, 0x32, 0x11, 0x1D, 0x0F, 0x03, 0x00, 0x00, ///*"9"*9/

0x08, 0x08, 0x0A, 0xEA, 0xAA, 0xAA, 0xAA, 0xFF, 0xA9, 0xA9, 0xA9, 0xE9, 0x08, 0x08, 0x08, 0x00,
0x40, 0x40, 0x48, 0x4B, 0x4A, 0x4A, 0x4A, 0x7F, 0x4A, 0x4A, 0x4A, 0x4B, 0x48, 0x40, 0x40, 0x00, ///*"重"*10/

0x40, 0x40, 0x40, 0xDF, 0x55, 0x55, 0x55, 0xD5, 0x55, 0x55, 0x55, 0xDF, 0x40, 0x40, 0x40, 0x00,
0x40, 0x40, 0x40, 0x57, 0x55, 0x55, 0x55, 0x7F, 0x55, 0x55, 0x55, 0x57, 0x50, 0x40, 0x40, 0x00, ///*"量"*11/

```
    0x00,0x00,0x00,0x00,0x00,0x00,0xC0,0xC0,0xC0,0xC0,0x00,0x00,0x00,0x00,0x00,
0x00,
    0x00,0x00,0x00,0x00,0x00,0x00,0x30,0x30,0x30,0x30,0x00,0x00,0x00,0x00,0x00,
0x00,///*":"*12/

    0x00,0x04,0x04,0xE4,0x24,0x24,0x24,0x3F,0x24,0x24,0x24,0xE4,0x04,0x04,0x00,
0x00,
    0x00,0x00,0x80,0x43,0x31,0x0F,0x01,0x01,0x01,0x3F,0x41,0x43,0x40,0x40,0x70,
0x00,///*"克"*13/

    };



    void send_byte(uchar dat ,uchar cs1,uchar cs2);
    void send_all(uint page,uint lie,uint offset);
    void delay(uint x);
    void init_adc();
    void init_yejing();
    void calibrate();
    int get_ad_result();
    void clearscreen();

    int cweight;
    int weight;




    void main()
    {
        init_yejing();
        init_adc();
        calibrate();///校准

        while(1)
        {
            weight=(get_ad_result()-cweight)/2-50;
              weight += weight/10;///真实重量
            clearscreen();

            send_all(1,1,10);///重
            send_all(1,2,11);///量
```

```
            send_all(1,3,12);///:
            send_all(4,3,weight/100);///百
            send_all(4,4,(weight/10)%10);///十
            send_all(4,5,weight%10);///个
            send_all(4,6,13);///克

            delay(50000);

        }

    }

    void init_yejing()
    {
        send_byte(192,1,1);///设置起始行
        send_byte(63,1,1);///打开显示开关
    }

    void send_byte(uchar dat,uchar cs1,uchar cs2)
    {
        P2=0xff;
        CS1=cs1; CS2=cs2;
        RS=0; RW=1; E=1;
        while(BUSY) ;

        ///送数据或控制字
        E=0;
        RS=!(cs1&&cs2),RW=0;
        P2=dat;
        E=1; delay(3); E=0;

        CS1=CS2=0;
    }

    void send_all(uint page,uint lie,uint offset)
    {
        uint i,j,k=0;
        for(i=0;i<2;++i)
        {
            send_byte(184+i+page,1,1);///选择页面
            send_byte(64+lie*16-(lie>3)*64,1,1);///选择列号
            for(j=0;j<16;++j)
                send_byte(zima[offset][k++],lie<4,lie>=4);///送数
        }
```

```
    }

    void init_adc()
    {
        P1ASF = 1; ///Set  P1.0 as analog input port
        AURX1 |= 0X04; ///AURX1 中的 ADRJ 位用于转换结果寄存器的数据格式调整控制

        ADC_RES = ADC_LOW2 = 0; ///Clear previous result

        ADC_CONTR = ADC_POWER | ADC_SPEEDLL | ADC_START | ch;  ///ch=0 ADC channel
NO.0
        delay(4);  ///ADC power-on delay and Start A/D conversion
    }

    int get_ad_result()
    {
        int ADC_result;
        ADC_RES = ADC_LOW2 = 0; ///Clear previous result
        ADC_CONTR = ADC_POWER | ADC_SPEEDLL | ch | ADC_START;
        _nop_(); _nop_(); _nop_(); _nop_(); _nop_(); _nop_(); ///Must wait before
inquiry
        while (!(ADC_CONTR & ADC_FLAG)); ///Wait complete flag
        ADC_result = (ADC_RES & 0x03) *256 + ADC_LOW2;///ADC_RES 中存高 2 位
        ADC_CONTR &= ~ADC_FLAG; ///Close ADC flag 位置 0
        return ADC_result;  ///Return ADC result

    }

    void calibrate()
    {
        cweight=(get_ad_result()-0)/2;
    }

    void delay(uint x)
    {
        while(x--) ;
    }

    void clearscreen()
    {
        int i,j;
        for(i=0;i<8;++i)
        {
            send_byte(184+i,1,1);///页
```

```
            send_byte(64,1,1);///列
                for(j=0;j<64;++j)
                    {
                        send_byte(0x00,0,1);
                        send_byte(0x00,1,0);
                    }
        }
    }
```

实验六　直流电机脉宽调制调速

一、实验目的和要求

掌握脉宽调制调速的原理与方法，学习频率/周期测量的方法，了解闭环控制的原理

二、实验原理

对于直流电机来说，其转速由输入电压决定，因此具有平滑调速的效果；相比而言，交流电机的转速由交流电频率和电机结构决定，难以改变速度。当然，交流电机构造简单，没有换向器，所以容易制造高转速、高电压、大电流、大容量的电机；而直流电机一般用在负荷小，但要求转速连续可调的场合，如伺服电机。

脉宽调制（Pulse Width Modulation，PWM）是一种能够通过开关量输出达到模拟量输出效果的方法。使用 PWM 可以实现频率调制、电压调制等效果，并且需要的外围器件较少，特别适合于单片机控制领域。这里只关心通过 PWM 实现电压调制，从而控制直流电机转速的效果。也称作脉宽调制调速。

PWM 的基本原理是通过输出一个很高频率的 0/1 信号，其中 1 的比例为 δ（也叫做占空比），在外围积分元件的作用下，使得总的效果相当于输出 δ ×A（A 为高电平电压）的电压。通过改变占空比就可以调整输出电压，从而达到模拟输出并控制电机转速的效果。

使用单片机实现 PWM，就是根据预定的占空比 δ 来输出 0 和 1，这里 δ 就是控制变量。最简单的办法就是以某个时间单位（如 0.1ms，相当于 10kHz）为基准，在前 N 段输出 1，后 M-N 段输出 0，总体的占空比就是 N/M。这种方法由于 0 和 1 分布不均匀，所以要求基准频率要足够高，否则会出现颠簸现象。

要达到更稳定的效果，可以采用累加进位法如果将总的周期内的 0 和 1 均匀分散开。设置一个累加变量 x，每次加 N，若结果大于 M，则输出 1，并减去 M；否则输出 0。这样整体的占空比也是 N/M。在实验中取 M=256 可以使程序更加简单。

另外，由于本实验板的设计，输出 0 使电机工作。因此对于本实验，上面所说的 0 和 1 要翻转过来用。

在本实验板中，电机每转动一次，与之相连的偏心轮将遮挡光电对管一次，因此会产生一个脉冲，送到 INT0。要测量转速，既可以测量相邻两次中断之间的时间；也可以测量一秒种之内发生的中断次数。显然，后一种方法更加简单。

进行转速控制时，涉及到三个变量：预期转速，实际转速和控制变量。这里控制变量就是占空比。我们并不能够预先精确知道某个控制变量的值会导致多少的实际转速，因为这里有很多内部和外部因素起作用（如摩擦力，惯性等），但可以确定就是随着控制变量的增加，实际转速会增加。

反馈控制的基本原理就是根据实际结果与预期结果之间的差值，来调节控制变量的值。当实际转速高于预期转速时，我们需要减少控制变量，以降低速度；反之则需要调高控制变量。

本实验的转速控制可以使用简单的比例控制算法，也就是当转速 S 大于预定值时，将输出 0 的个数减少；当转速小于预定值时，将输出 0 的个数增加。改变值正比于测量出的差值。也可自行使用其他更加复杂的算法。

实验中采用的电机最大转速在 200 转/s 左右，转速小于 40 转/s 左右将不稳定，可能会停转。

三、实验器材

单片机测控实验系统

直流电机调速实验模块

Keil 开发环境

STC-ISP 程序下载工具

四、实验内容

1. 在液晶显示屏上显示出直流电机的：当前转速、低目标转速、高目标转速。

2. 固定向 P1.1 输出 0，然后测量每秒钟电机转动的转数，将其显示在数码管，每秒刷新一次即可。

3. 使用脉宽调制的方法，动态调整向 P1.1 输出的内容，使得电机转速能够稳定在一个预定值附近，同时实时显示当前转速。

4. 根据输入修改电机得目标转速值，设置两个转速目标值：低转速和高转速。

5. 每隔一秒钟读取两个开关的状态，如果 S1 按下，动态调整输出，使得电机转速能够稳定到低转速目标值附近，如果 S2 按下，动态调整输出，使得电机转速能够稳定到高转速目标值附近。交替显示目标值和当前转速值。

五、实验步骤

4.1 建立工程，实现实验内容 1

预习附录四，学习 C51 编程方法。设计实现一个进行显示的 C51 程序。

建立工程，实现实验内容 1。

将例子程序补充完整。建立一个新的工程，然后加入一个 C 语言文件，输入上述例子程序，编译并下载执行调试。

4.2 编写中断程序，测量电机转速

编写中断程序，测量直流电机的转速。

按照实验原理，电机转速就是一秒钟之内 INT0 的中断个数。编写带有中断

的 C51 程序，包括一个能够实现 1 秒钟的定时器中断和一个外部中断。注意外部中断要设置边沿触发方式。程序框架参考附录四。

4.3 完成控制转速程序

按照脉宽调制的原理，再添加一个快速的定时中断（0.1ms 左右），在这个中断里面动态改变 P1.1 的输出，宏观上输出有效（0）的比例就是预定的控制变量。这个控制变量增大，电机转速就应该提高，但由于各种内部和外部因素，它们之间不存在简单的函数关系，因此必须根据测量出来的实际转速进行动态调整。

首先将电机转速控制在一个预定数值附近，在每一个 1 秒钟中断测量出当前转速之后，将其与目标值相对比，如果不够则增加控制变量，否则减少之，这样就能逐步达到稳定转速的目的。同时将速度显示出来。

4.4 完成整体实验内容

在上面程序的基础上，再加上根据开关状态改变预定转速的代码。同时，在主程序中交替显示目标值和当前转速值，显示一个内容之后等待一段时间（可以由延时代码实现），然后再显示另一个并延时。要显示的内容都是在中断中被修改的。

六、思考题

1. 讨论脉宽调速和电压调速的区别、优缺点和应用范围。

答:脉宽调速:是一种能够通过开关量输出达到模拟量输出效果的方法。PWM 的基本原理是通过输出一个很高频率的 0/1 信号，其中 1 的比例为 δ (也叫做占空比)，通过改变占空比就可以调整输出电压，从而达到模拟输出并控制电机转速的效果。并且需要的外围器件较少，特别适合于单片机控制领域。

电压调速:直接改变电压模拟量从而改变电机转速的方法。电压便于平滑性调节，可以实现无级调速，损耗小，调速经济性好。

2.说明程序原理中累加进位法的正确性。

答:设置一个累加变量 x，每次加 N,若结果大于 M,则输出 1, 并减去 M:否则输出 0。这样整体的占空比也是 N/M.每次循环中都有 M/N 次输出，而其中只有一次输出为 1,即总共输出了 N 个 1,而且总输出次数是 M 次，1 的比例就是 N/M。

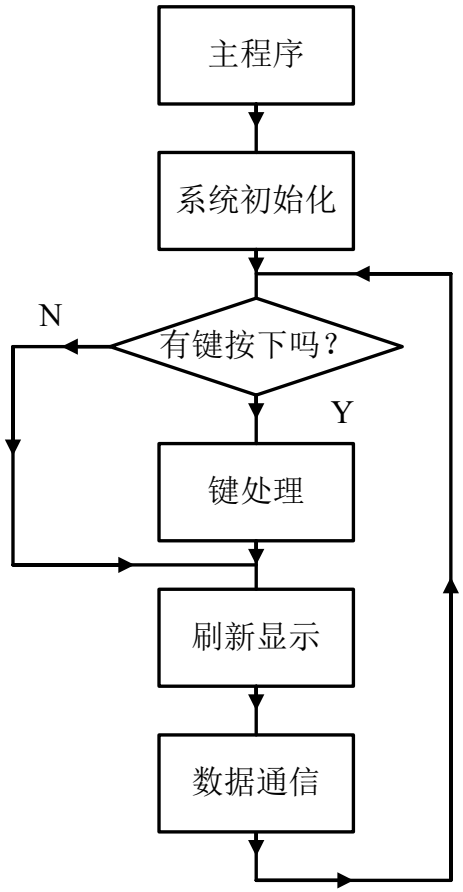
3.计算转速测量的最大可能误差，讨论减少误差的办法。

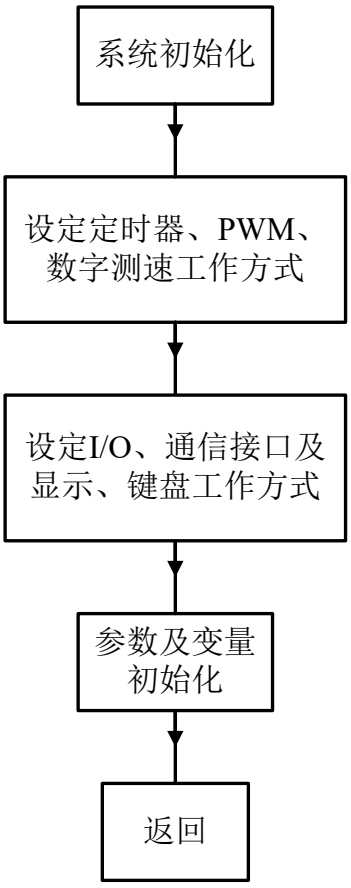
答:减少误差的方法:

外部因素:减少摩擦力

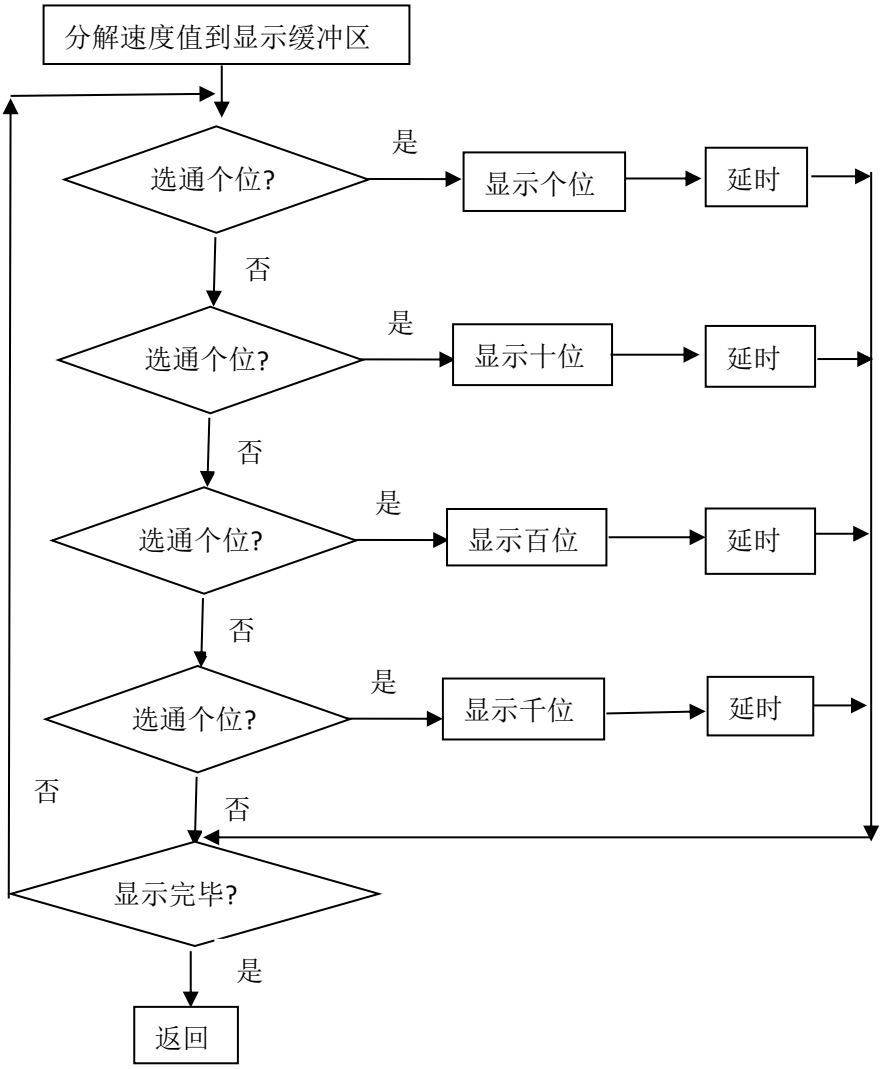
内部因素:让电机的转速保持在 200~40 转/s 之间的速度,并保持一个比较低的速度(速度不要过快)。

七、流程图

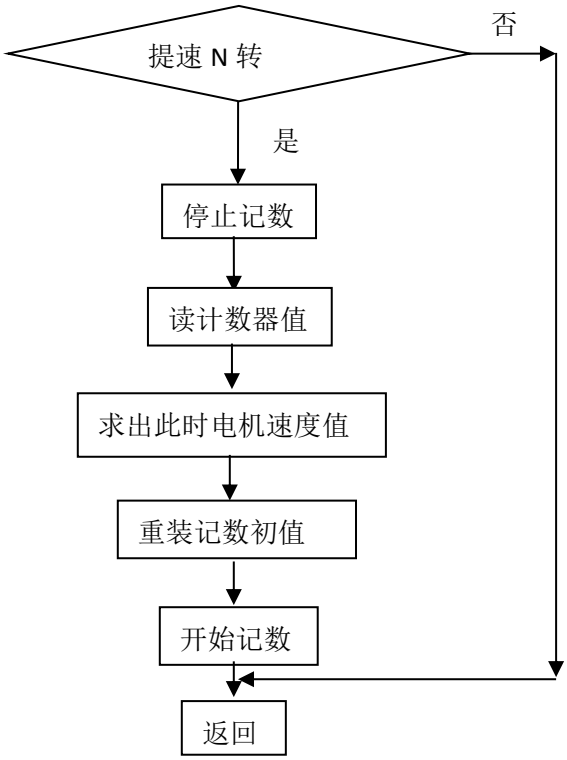

主程序
系统初始化
N
有键按下吗？
Y
键处理
刷新显示
数据通信


系统初始化
设定定时器、PWM、
数字测速工作方式
设定I/O、通信接口及
显示、键盘工作方式
参数及变量
初始化
返回

显示子程序：


分解速度值到显示缓冲区
选通个位?
是
显示个位
延时
否
选通个位?
是
显示十位
延时
否
选通个位?
是
显示百位
延时
否
选通个位?
是
显示千位
延时
否
否
显示完毕?
是
返回

测速子程序：



代码：

```
#include <reg52.h>
#include <intrins.h>

#define uchar unsigned char
#define uint unsigned int

sfr P4=0xC0;
sfr P4SW=0xBB;
///sfr IE=0xA8; ///中断允许寄存器

sbit sclk=P4^4;
sbit sdata=P4^5;
sbit swh1=P3^6;
sbit swh2=P3^7;
sbit motor=P1^1;

uchar tab[15]=
      {0xC0,0xF9,0xA4,0xB0,0x99,
            0x92,0x82,0x0F8,0x80,0x90};
```

```
uchar tspeed=0;///累加转数
uchar cspeed=0;///当前速度
uchar xspeed=100;///期望速度

uchar t1_cnt=0; ///1s 延时控制变量  50ms*20 次
int N=100;///占空比
int M=256;
int X=0;///起始变量


init();
void sendbyte(uchar ch);
void display(uchar n);
void delay1();
void delay2();
//void ex_int0() interrupt 0 ;
//void t1_int() interrupt 3 ;
//void t0_int() interrupt 1 ;


uchar flag1=0;
uchar flag2=0;


void main()
{
	init();
	motor=0;
	while(1)
	{
		display(cspeed);
		delay2();
		display(xspeed);
		delay1();
	}
}



init()
{
	P4SW=0x30;

	IT0=1; ///设置 INT0 为边沿触发
```

```
    EA=1;
    ET1=1;
    ET0=1;
    EX0=1;   ///中断允许

    TMOD=0x11; ///设置定时器 0 和定时器 1 的工作方式
    TH1=0x3C;
    TL1=0xB0;   ///50ms 计数值
    TH0=0xFF;
    TL0=0x9C;   ///0.1ms 计数值

    TR0=1;
    TR1=1;    ///启动定时器
}


void ex_int0() interrupt 0   ///外部中断 INT0
{
    tspeed++;
}

void t1_int() interrupt 3     ///50ms 定时器中断  T1
{
    if(++t1_cnt<20)
    {
        if(swh1==0)
        {
            flag1=1;
        }
        if(swh1==1 && flag1==1)
        {
            xspeed++;
            flag1=0;
        }


        if(swh2==0)
            flag2=1;
        if(swh2==1 && flag2==1)
        {
            xspeed--;
            flag2=0;
        }
```

```
            return;
        }
        t1_cnt=0;
        cspeed=tspeed;
        tspeed=0;
        if(cspeed>xspeed) N++;
        if(cspeed<xspeed) N--;
}

void t0_int() interrupt 1    ///0.1ms 定时器中断  T0
{
         X+=N;
         if(X>M)
         {
               motor=1; ///不转
               X-=M;
         }
         else
               motor=0; ///转
}

void sendbyte(uchar ch)
{
        uchar shape,c;
        shape=tab[ch];
        for(c=0;c<8;c++)
        {
              sclk=0;
              sdata=shape & 0x80;
              sclk=1;
              shape <<= 1;
        }
}

void display(uchar n)
{
        sendbyte(n%10);             ///个
        sendbyte((n/10)%10);    ///十
        sendbyte(n/100);           ///百
}

void delay1()
{
```

```
    int i,j;
    for(i=0;i<1000;i++)
        for(j=0;j<500;j++);
}

void delay2()
{
    int i,j;
    for(i=0;i<1000;i++)
        for(j=0;j<1000;j++);
}
```

实验八　温度测量与控制

一、实验目的和要求

1.学习 DS18B20 温度传感器的编程结构。

2.了解温度测量的原理。

3. 掌握 PID 控制原理及实现方法。

3. 加深 C51 编程语言的理解和学习。

二、实验原理

本实验使用的 DS18B20 是单总线数字温度计，测量范围从—55℃到+125℃，增量值为 0.5 ℃。

用于贮存测得的温度值的两个 8 位存贮器 RAM 编号为 0 号和 1 号。

1 号存贮器存放温度值的符号，如果温度为负（℃），则 1 号存贮器 8 位全为 1，否则全为 0。

0 号存贮器用于存放温度值的补码 LSB(最低位)的 1 表示 0.5℃ 。

将存贮器中的二进制数求补再转换成十进制数并除以 2，就得到被测温度值。

温度检测与控制系统由加热灯泡，温度二极管，温度检测电路，控制电路和继电器组成。温度二极管和加热灯泡封闭在一个塑料保温盒内，温度二极管监测保温盒内的温度，用温控实验板内部的 A/D 转换器 ADC7109 检测二极管两端的电压，通过电压和温度的关系，计算出盒内空气的实际温度。

相关背景知识参见 DS18B20 中文资料

实验原理见附录七。

本实验使用 STC89C516RD+单片机实验板。单片机的 P1.4 与 DS18B20 的 DQ 引脚相连，进行数据和命令的传输。

单片机的 P1.1 连接热电阻。当 P1.1 为高电平时，加热热电阻。

温度控制的方法采用 PID 控制实现。

三、实验器材

单片机测控实验系统
温控实验模块
Keil 开发环境
STC-ISP 程序下载工具

四、实验内容

掌握使用传感器测量与控制温度的原理与方法，使用 C51 语言编写实现温度控制的功能，使用超声波/温度实验板测量温度，将温度测量的结果（单位为摄氏度）显示到液晶屏上。

编程实现测量当前教室的温度，显示在 LCM 液晶显示屏上。

通过 S1 设定一个高于当前室温的目标温度值。

编程实现温度的控制，将当前温度值控制到目标温度值并稳定的显示

五、实验步骤

1. 预习，参考附录三，预习 DS18B20 的编程结构，编程时注意 DS18B20 的时间要求，必须准确满足。根据实验原理附录中的流程图进行编程。

2. 将编译后的程序下载到 51 单片机，观察温度的测量结果。

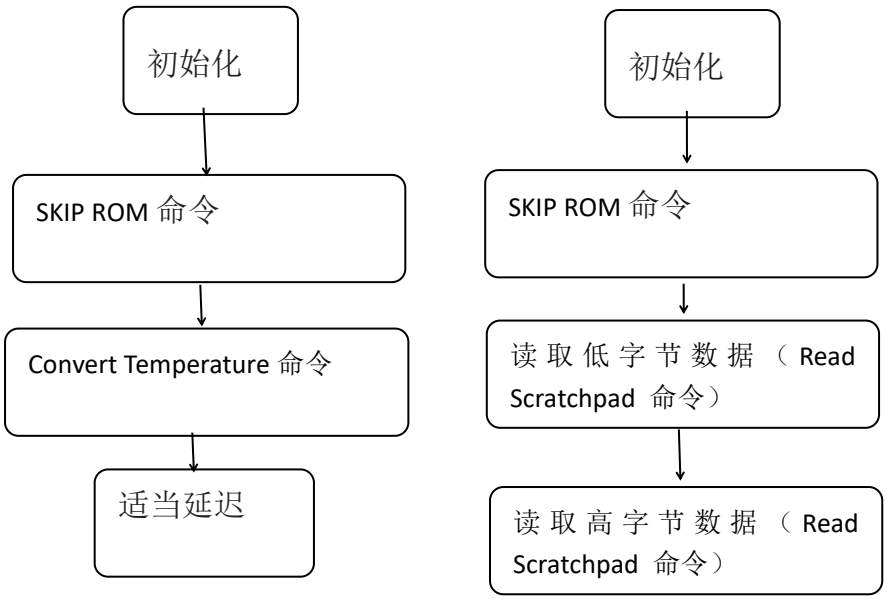
3. 程序调试

六、思考题

1. 进行精确的延时的程序有几种方法？各有什么优缺点？。

直接让单片机做空循环，死等。 2，利用定时器的溢出间隔，如果时间上不够，可以在溢出中断中配合软件计数器来实现。 前者浪费 cpu，后者比较高效，但是占用一个定时器

七、流程图



代码：

```
#include <reg52.h>
#include <intrins.h>

#define uchar unsigned char
#define uint unsigned int


uchar code zima[20][32]=
{
0x00,0x00,0xC0,0xE0,0x30,0x10,0x08,0x08,0x08,0x08,0x08,0x18,0x30,0xE0,0xC0,0x00,
0x00,0x00,0x07,0x0F,0x18,0x10,0x20,0x20,0x20,0x20,0x20,0x10,0x18,0x0F,0x07,0x00,///*"0
"*0/

0x00,0x00,0x00,0x10,0x10,0x10,0x10,0xF0,0xF8,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x20,0x20,0x20,0x20,0x3F,0x3F,0x20,0x20,0x20,0x20,0x00,0x00,0x00,///*"1
"*1/

0x00,0x00,0x60,0x50,0x10,0x08,0x08,0x08,0x08,0x08,0x08,0x98,0xF0,0x70,0x00,0x00,
0x00,0x00,0x20,0x30,0x28,0x28,0x24,0x24,0x22,0x22,0x21,0x20,0x30,0x18,0x00,0x00,///*"
2"*2/

0x00,0x00,0x30,0x30,0x08,0x08,0x88,0x88,0x88,0x88,0x58,0x70,0x30,0x00,0x00,0x00,
0x00,0x00,0x18,0x18,0x20,0x20,0x20,0x20,0x20,0x20,0x31,0x11,0x1F,0x0E,0x00,0x00,///*"
3"*3/

0x00,0x00,0x00,0x00,0x00,0x80,0x40,0x20,0x10,0xF0,0xF8,0xF8,0x00,0x00,0x00,0x00,
0x00,0x04,0x06,0x05,0x05,0x04,0x24,0x24,0x24,0x3F,0x3F,0x3F,0x24,0x24,0x24,0x00,///*"4
"*4/

0x00,0x00,0x00,0xC0,0x38,0x88,0x88,0x88,0x88,0x88,0x88,0x88,0x08,0x08,0x00,0x00,
0x00,0x00,0x18,0x29,0x21,0x20,0x20,0x20,0x20,0x20,0x20,0x30,0x11,0x1F,0x0E,0x00,0x00,///*"
5"*5/

0x00,0x00,0x80,0xE0,0x30,0x10,0x98,0x88,0x88,0x88,0x88,0x88,0x98,0x10,0x00,0x00,
0x00,0x00,0x07,0x0F,0x19,0x31,0x20,0x20,0x20,0x20,0x20,0x20,0x11,0x1F,0x0E,0x00,///*"6
"*6/

0x00,0x00,0x30,0x18,0x08,0x08,0x08,0x08,0x08,0x88,0x48,0x28,0x18,0x08,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x38,0x3E,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x00,///*"
7"*7/
```

```
    0x00,0x00,0x70,0x70,0xD8,0x88,0x88,0x08,0x08,0x08,0x08,0x98,0x70,0x70,0x00,0x00,
    0x00,0x0C,0x1E,0x12,0x21,0x21,0x20,0x21,0x21,0x21,0x23,0x12,0x1E,0x0C,0x00,0x00,///*"
8"*8/

    0x00,0xE0,0xF0,0x10,0x08,0x08,0x08,0x08,0x08,0x08,0x18,0x10,0xF0,0xC0,0x00,0x00,
    0x00,0x00,0x11,0x33,0x22,0x22,0x22,0x22,0x22,0x32,0x11,0x1D,0x0F,0x03,0x00,0x00,///*"
9"*9/

    0x08,0x08,0x0A,0xEA,0xAA,0xAA,0xAA,0xFF,0xA9,0xA9,0xA9,0xE9,0x08,0x08,0x08,0x00,
    0x40,0x40,0x48,0x4B,0x4A,0x4A,0x4A,0x7F,0x4A,0x4A,0x4A,0x4B,0x48,0x40,0x40,0x00,///*
"重"*10/

    0x40,0x40,0x40,0xDF,0x55,0x55,0x55,0xD5,0x55,0x55,0x55,0xDF,0x40,0x40,0x40,0x00,
    0x40,0x40,0x40,0x57,0x55,0x55,0x55,0x7F,0x55,0x55,0x55,0x57,0x50,0x40,0x40,0x00,///*"
量"*11/

    0x00,0x00,0x00,0x00,0x00,0x00,0xC0,0xC0,0xC0,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,
    0x00,0x00,0x00,0x00,0x00,0x00,0x30,0x30,0x30,0x30,0x00,0x00,0x00,0x00,0x00,0x00,///*":
"*12/

    0x00,0x04,0x04,0xE4,0x24,0x24,0x24,0x3F,0x24,0x24,0x24,0xE4,0x04,0x04,0x00,0x00,
    0x00,0x00,0x80,0x43,0x31,0x0F,0x01,0x01,0x01,0x3F,0x41,0x43,0x40,0x40,0x70,0x00,///*"
克"*13/

    0x10,0x21,0x86,0x70,0x00,0x7E,0x4A,0x4A,0x4A,0x4A,0x4A,0x7E,0x00,0x00,0x00,0x00,
    0x02,0xFE,0x01,0x40,0x7F,0x41,0x41,0x7F,0x41,0x41,0x7F,0x41,0x41,0x7F,0x40,0x00,///*"
温",14*/

    0x00,0x00,0xFC,0x04,0x24,0x24,0xFC,0xA5,0xA6,0xA4,0xFC,0x24,0x24,0x24,0x04,0x00,
    0x80,0x60,0x1F,0x80,0x80,0x42,0x46,0x2A,0x12,0x12,0x2A,0x26,0x42,0xC0,0x40,0x00,///*"
度",15*/

    };



    sbit CS1=P1^7;///左半边
    sbit CS2=P1^6;///右半边
    sbit E=P3^3;///使能信号
    sbit RW=P3^4;///读写操作选择
    sbit RS=P3^5;///寄存器选择(数据/指令)
    sbit RES=P1^5;///复位 低电平有效
    sbit BUSY=P2^7;
```

```
sbit De=P1^1; ///加热
sbit DQ=P1^4; ///DS18B20 单数据总线
uchar TPH,TPL; ///温度值高位  低位
unsigned int t; ///温度值
unsigned int t1=30; ///目标温度值

sbit swh1=P3^6;
sbit swh2=P3^7;
uchar flag1=0;
uchar flag2=0;



void send_byte(uchar dat ,uchar cs1,uchar cs2);
void send_all(uint page,uint lie,uint offset);
void delay(uint x);
void init_yejing();
void clearscreen();

void DelayXus(uchar n); ///微秒级延时
void ow_rest(); ///复位
void write_byte(char dat);
unsigned char read_bit(void);







void main(void)
{
    init_yejing();

    t=0;

while(1)
{


      if(swh1==0)
      {
            flag1=1;
      }
```

```
        if(swh1==1 && flag1==1)
        {
              t1++;
              flag1=0;
        }


        if(swh2==0)
              flag2=1;
        if(swh2==1 && flag2==1)
        {
              t1--;
              flag2=0;
        }



 if(t<t1)
 De=1;
 else De=0;
 ow_rest(); ///设备复位

 write_byte(0xCC); ///跳过 ROM 命令

 write_byte(0x44); ///开始转换命令
 while (!DQ); ///等待转换完成

 ow_rest(); ///设备复位
 write_byte(0xCC); ///跳过 ROM 命令
 write_byte(0xBE); ///读暂存存储器命令
 TPL = read_bit(); ///读温度低字节
 TPH = read_bit(); ///读温度高字节


 t=TPH; ///取温度高位
 t<<=8; ///高位 8 位
 t|=TPL; ///加上温度低位
 t*=0.625; ///实际温度 可直接显示

  t=t/10;


 send_all(1,1,14);///温
send_all(1,2,15);///度
```

```
    send_all(1,3,12);///:

     send_all(4,2,t1/10);///十
    send_all(4,3,t1%10);///个

    send_all(4,5,t/10);///十
    send_all(4,6,t%10);///个

     delay(50000);

     clearscreen();




    }
}




void DelayXus(uchar n)
{
    while (n--)
    {
        _nop_();
        _nop_();
    }
}

unsigned char read_bit(void)///读位
{
    uchar i;
    uchar dat = 0;
    for (i=0; i<8; i++) ///8 位计数器
    {
        dat >>= 1;
        DQ = 0; ///开始时间片
        DelayXus(1); ///延时等待
        DQ = 1; ///准备接收
        DelayXus(1); ///接收延时
        if (DQ) dat |= 0x80; ///读取数据
        DelayXus(60); ///等待时间片结束
    }
```

```
    return dat;
}
void ow_rest()///复位
{
    CY = 1;
    while (CY)
    {
        DQ = 0; ///送出低电平复位信号
        DelayXus(240); ///延时至少 480us
        DelayXus(240);
        DQ = 1; ///释放数据线
        DelayXus(60); ///等待 60us
        CY = DQ; ///检测存在脉冲,DQ 为 0 转换完成
        DelayXus(240); ///等待设备释放数据线
        DelayXus(180);
    }
}

void write_byte(char dat)///写字节
{
    uchar i;
    for (i=0; i<8; i++) ///8 位计数器
    {
        DQ = 0; ///开始时间片
        DelayXus(1); ///延时等待
        dat >>= 1; ///送出数据
        DQ = CY;
        DelayXus(60); ///等待时间片结束
        DQ = 1; ///恢复数据线
        DelayXus(1); ///恢复延时
    }
}









void init_yejing()
{
    send_byte(192,1,1);///设置起始行
    send_byte(63,1,1);///打开显示开关
}
```

```
void send_byte(uchar dat,uchar cs1,uchar cs2)
{
    P2=0xff;
    CS1=cs1; CS2=cs2;
    RS=0; RW=1; E=1;
    while(BUSY) ;

    ///送数据或控制字
    E=0;
    RS=!(cs1&&cs2),RW=0;
    P2=dat;
    E=1; delay(3); E=0;

    CS1=CS2=0;
}

void send_all(uint page,uint lie,uint offset)
{
    uint i,j,k=0;
    for(i=0;i<2;++i)
    {
        send_byte(184+i+page,1,1);///选择页面
        send_byte(64+lie*16-(lie>3)*64,1,1);///选择列号
        for(j=0;j<16;++j)
            send_byte(zima[offset][k++],lie<4,lie>=4);///送数
    }
}



void delay(uint x)
{
    while(x--) ;
}

void clearscreen()
{
    int i,j;
    for(i=0;i<8;++i)
    {
        send_byte(184+i,1,1);///页
        send_byte(64,1,1);///列
            for(j=0;j<64;++j)
                {
                    send_byte(0x00,0,1);
```

```
                    send_byte(0x00,1,0);
                }
    }
}
```

三次实验的体会与收获

通过这三次实验，掌握了点阵式液晶显示屏的原理和控制方法，掌握了点阵字符的显示方法。掌握了模拟/数字（A/D）转换方式，进一步掌握了使用 C51 语言编写程序的方法，使用 C51 语言编写实现重量测量的功能。掌握了脉宽调制调速的原理与方法，学习了频率/周期测量的方法，了解了闭环控制的原理，学习了 DS18B20 温度传感器的编程结构。了解了温度测量的原理。掌握了 PID 控制原理及实现方法。 加深了 C51 编程语言的理解和学习。

在一开始做课程设计的时候由于缺乏经验和资料，没有找准目标，思路和设计都缺乏针对性，在一些小细节上浪费了很多时间。通过和同学的交流，找准了此次课程设计的重点，在网上有目的的找到了相关的资料，在同学的帮助和自己的努力下完成了最后的课程设计。虽然还有很多地方需要完善，但通过本次课程设计使我对自己的动手能力更加的有自信，同时也了解到人多力量大的道理，和同学的交流以及上网搜索资料能更快更好的完成任务