单片机第五次实验报告

学号：53160819　班级：8 班　姓名：李申瑞

一、**实验题目**：重量测量

二、**实验目的**

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。

三、**实验内容**

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。

四、**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。

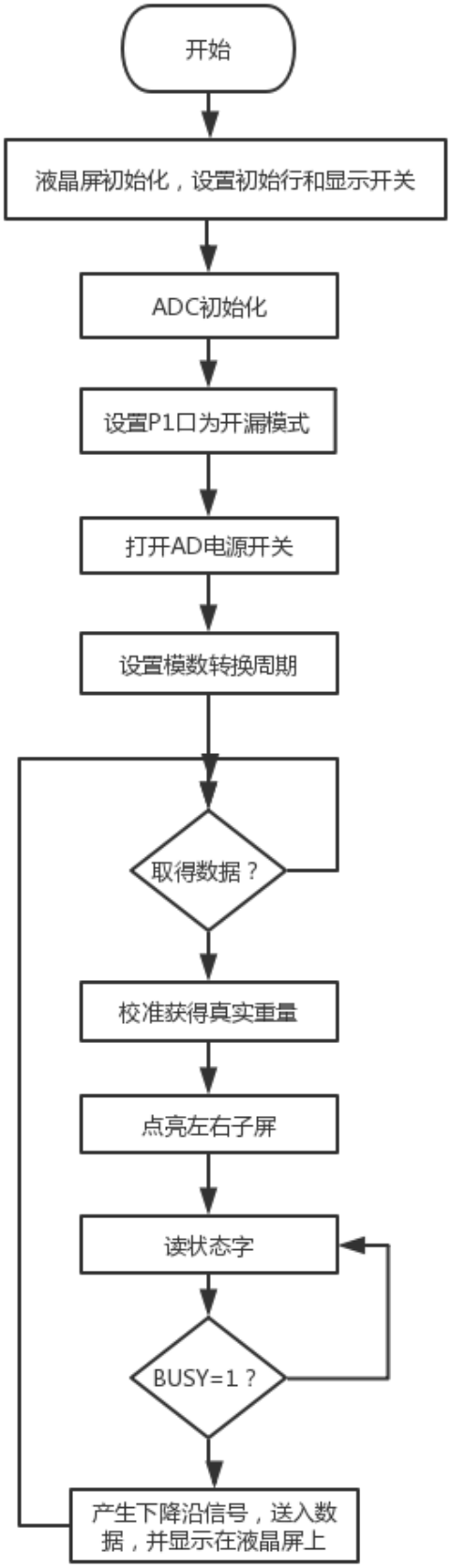
四、**实验原理**

液晶显示器 LCM，在数据或命令送入阶段，，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。

模数转换器 ADC，与其相关的寄存器：ADC_RES、ADC_RESL：A/D 转换结果寄存器，是特殊功能寄存器，用于保存 A/D 转换结果; IE：中断允许寄存器（可位寻址）；EA：CPU 的中断开放标志，EA=1，CPU 开放中断，EA=0，CPU 屏蔽所有中断申请。EADC：A/D 转换中断允许位。1 允许 0 禁止；IPH：中断优先级控制寄存器高（不可位寻址）；IP：中断优先级控制寄存器低（可位寻址）。

数据产生：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。

五、程序流程图


开始
液晶屏初始化，设置初始行和显示开关
ADC初始化
设置P1口为开漏模式
打开AD电源开关
设置模数转换周期
取得数据？
校准获得真实重量
点亮左右子屏
读状态字
BUSY=1？
产生下降沿信号，送入数据，并显示在液晶屏上

六、实验程序

```
#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

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

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"*

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

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

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,///*":"*

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

};
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);
                }
    }
}
```

七、思考题

1. 调零的原理，软件调零和硬件调零的区别。

答：调零是指在没有放置砝码的前提下，需要使液晶屏显示为 0。

软件调零是指在程序中通过减去空砝码值重力测量值显示为 0。

硬件调零是指在通过调整压敏电阻的阻值来进行调整。

2. 模/数和数/模的信号转换原理。

答：模数转换就是通过将产生的电流或电压以数字信号输出：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。；而数模转换是将输入的二进制数字信号转换为模拟信号，以电压或电流信号输出。

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。

八、思考与总结

通过此次实验，我初步掌握了液晶屏显示的原理和控制方法，通过选定行和列号来显示数据，送数据的时候要注意判断 BUSY 位和下降沿信号的产生，这是送入数据的关键；再者是了解了 ADC 模数转换器的使用，压敏电阻和放大电路产生的电压值，通过 ADC 模数转换器最后转换为数值送入液晶屏。

单片机第六次实验报告

一、**实验题目：**直流电机脉宽调制调速

二、实验目的及要求

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。

三、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。

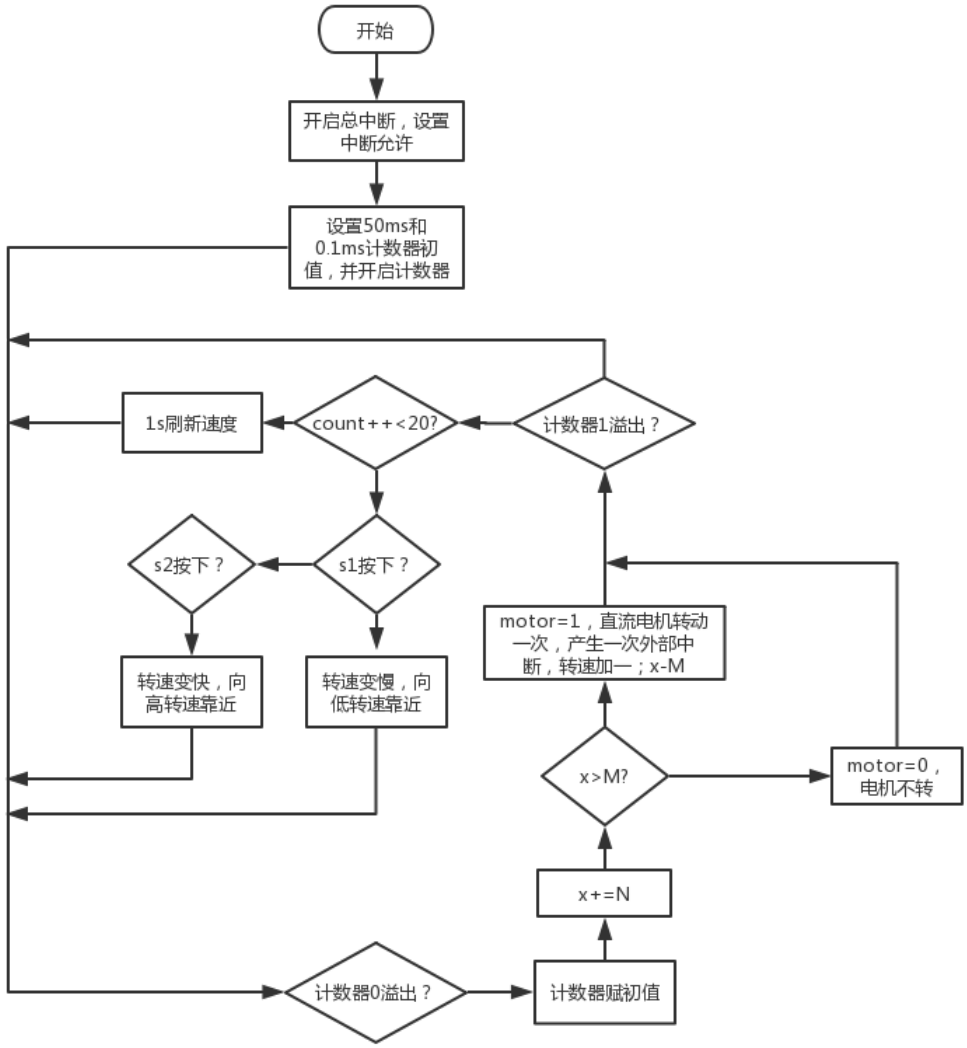
4. 根据输入修改电机得目标转速值，设置两个转速目标值：低转速和高转速。

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。

四、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,而脉宽调制（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 可以实现电压调制，从而控制直流电机转速的效果；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。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，而测量转速的方法就是计算一秒钟内 INT0 发生的中断次数。而当想要改变转速时, 就改变 N 的大小，也就是输出 0 的个数即可。

五、实验流程图



六、实验代码

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

#define uchar unsigned char
#define uint unsigned int

sfr P4=0xC0;
sfr P4SW=0xBB;

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;
uchar cspeed=0;
uchar xspeed=100;

uchar t1_cnt=0;
int N=100;
int M=256;
int X=0;

init();
void sendbyte(uchar ch);
void display(uchar n);
void delay1();
void delay2();

uchar flag1=0;
uchar flag2=0;

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

init()
{
    P4SW=0x30;

    IT0=1;
    EA=1;
    ET1=1;
    ET0=1;
    EX0=1;
```

```
    TMOD=0x11;
    TH1=0x3C;
    TL1=0xB0;
    TH0=0xFF;
    TL0=0x9C;

    TR0=1;
    TR1=1;
}

void ex_int0() interrupt 0
{
    tspeed++;
}

void t1_int() interrupt 3
{
    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
{
        X+=N;
        if(X>M)
        {
              motor=1;
              X-=M;
        }
        else
              motor=0;
}
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);
      sendbyte((n/10)%10);
      sendbyte(n/100);
}
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++);
}
```

七、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)。

八、思考与总结

通过此次实验，我初步掌握了运用 pwm 脉宽调制来控制电压，进

而控制电机转速的方法，进一步加强对液晶屏显示原理的理解。通过设置一个控制变量,用累加进位的方法输出一个很高频率的 0/1 电压。每次电机转动一次，与之相连的偏心轮将遮挡光电对管一次，就会产生一个脉冲给 INT0，转速就增加一次。计算一秒内的中断次数就是转速。

第八次单片机实验报告

一、实验题目：温度测量与控制

二、实验目的及要求

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。

三、实验内容

1.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。

2.编程实现测量当前教室的温度，显示在 LCM 液晶显示屏上。

3.通过 S1 设定一个高于当前室温的目标温度值。

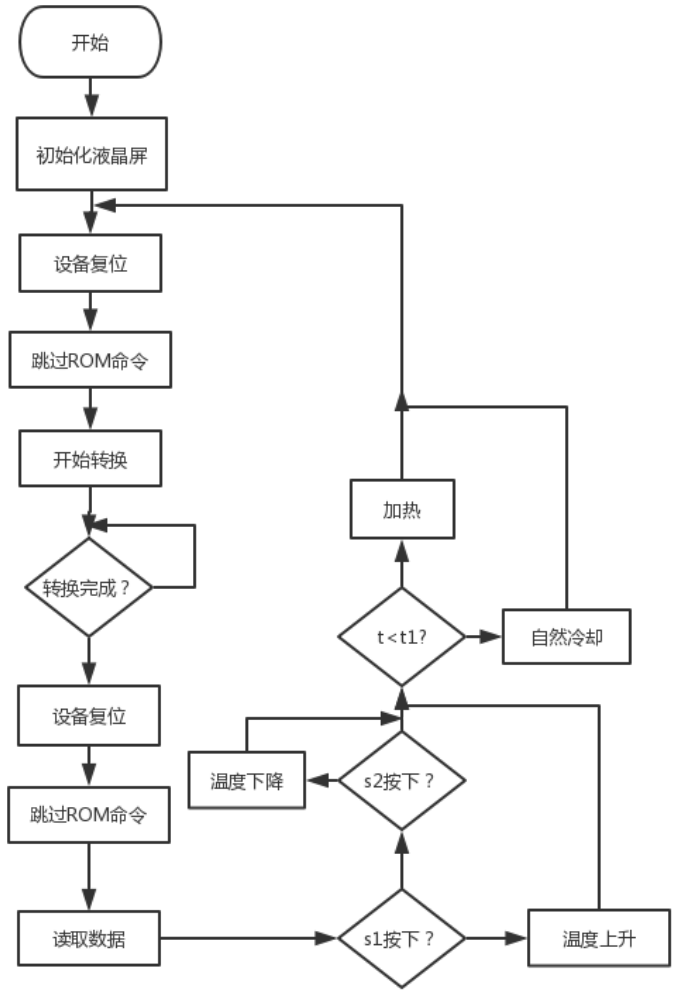
4.编程实现温度的控制，将当前温度值控制到目标温度值并稳定的显示。

四、**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。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。

五、实验流程图



六、实验程序

```
#include <reg52.h>
#include <intrins.h>
#define uchar unsigned char
#define uint unsigned int
uchar code zima[20][32]=
{
0x00,0x00,0xC0,0xE0,0x30,0x10,0x08,0x08,0x08,0x08,0x08,0x18,0x30,0xE0,0x
C0,0x00,
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,0x20,0x20,0x20,0x20,0x00,0x
00,0x00,///*"1"*1/
0x00,0x00,0x60,0x50,0x10,0x08,0x08,0x08,0x08,0x08,0x08,0x98,0xF0,0x70,0x0
0,0x00,
0x00,0x00,0x20,0x30,0x28,0x28,0x24,0x24,0x22,0x22,0x21,0x20,0x30,0x18,0x0
0,0x00,///*"2"*2/
0x00,0x00,0x30,0x30,0x08,0x08,0x88,0x88,0x88,0x88,0x58,0x70,0x30,0x00,0x0
0,0x00,
0x00,0x00,0x18,0x18,0x20,0x20,0x20,0x20,0x20,0x20,0x31,0x11,0x1F,0x0E,0x00,
0x00,///*"3"*3/
0x00,0x00,0x00,0x00,0x00,0x80,0x40,0x20,0x10,0xF0,0xF8,0xF8,0x00,0x00,0x0
0,0x00,
0x00,0x04,0x06,0x05,0x05,0x04,0x24,0x24,0x24,0x3F,0x3F,0x3F,0x24,0x24,0x
24,0x00,///*"4"*4/
0x00,0x00,0x00,0xC0,0x38,0x88,0x88,0x88,0x88,0x88,0x88,0x88,0x08,0x08,0x
00,0x00,
0x00,0x00,0x18,0x29,0x21,0x20,0x20,0x20,0x20,0x20,0x30,0x11,0x1F,0x0E,0x00,
0x00,///*"5"*5/
0x00,0x00,0x80,0xE0,0x30,0x10,0x98,0x88,0x88,0x88,0x88,0x88,0x98,0x10,0x0
0,0x00,
0x00,0x00,0x07,0x0F,0x19,0x31,0x20,0x20,0x20,0x20,0x20,0x20,0x11,0x1F,0x0E,
0x00,///*"6"*6/
0x00,0x00,0x30,0x18,0x08,0x08,0x08,0x08,0x08,0x88,0x48,0x28,0x18,0x08,0x0
0,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x38,0x3E,0x01,0x00,0x00,0x00,0x00,0x00,0x0
0,0x00,///*"7"*7/
0x00,0x00,0x70,0x70,0xD8,0x88,0x88,0x08,0x08,0x08,0x08,0x98,0x70,0x70,0x
```

```
00,0x00,
0x00,0x0C,0x1E,0x12,0x21,0x21,0x20,0x21,0x21,0x21,0x23,0x12,0x1E,0x0C,0x00,
0x00,///*"8"*8/
0x00,0xE0,0xF0,0x10,0x08,0x08,0x08,0x08,0x08,0x08,0x18,0x10,0xF0,0xC0,0x0
0,0x00,
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,0x48,0x40,0
x40,0x00,///*"重"*10/
0x40,0x40,0x40,0xDF,0x55,0x55,0x55,0xD5,0x55,0x55,0x55,0xDF,0x40,0x40,0
x40,0x00,
0x40,0x40,0x40,0x57,0x55,0x55,0x55,0x7F,0x55,0x55,0x55,0x57,0x50,0x40,0x4
0,0x00,///*"量"*11/
0x00,0x00,0x00,0x00,0x00,0x00,0xC0,0xC0,0xC0,0xC0,0x00,0x00,0x00,0x00,0
x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x30,0x30,0x30,0x30,0x00,0x00,0x00,0x00,0x
00,0x00,///*":"*12/
0x00,0x04,0x04,0xE4,0x24,0x24,0x24,0x3F,0x24,0x24,0x24,0xE4,0x04,0x04,0x
00,0x00,
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,0x00,0x00,0x
00,0x00,
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,0x26,0x42,0xC0,0x4
0,0x00,///*"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);
            }
   }
}
```

七、**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？

答：(1)、使用循环函数延时。这种方法比较精确，但它会一直占用

CPU，延时期间 CPU 只能选择等待，无法执行其他事情，降低了 CPU 的使用效率。(2)、使用定时器中断延时。这种方法是硬件延时，可以提高 CPU 的使用率，也能做到精确延时。

2. 参考其他资料，了解 DS18B20 的其他命令用法。

答： ROM 操作命令：(1) Read ROM(读 ROM)。(2) Match ROM(匹配 ROM)。(3) Skip ROM (跳过 ROM)。(4) Search ROM (搜索 ROM)。(5) Alarm search (告警搜索)。 存储器操作命令：(1) write Scratchpad (写暂存存储器)。(2) Read Scratchpad (读暂存存储器)。(3) Copy Scratchpad (复制暂存存储器)。(4) Convert Temperature (温度变换)。(5) Recall EPROM (重新调出)。(6) Read Power Supply (读电源)。

八、思考与总结

通过此次实验，我初步了解了 DS18B20 温度控制的使用方法，它的工作过程可分为初始化-ROM 操作命令-存储器操作命令-处理数据。它会返回 10 数据，取其高 2 位和低 8 位形成 8 位数据送入液晶屏。其次进一步理解了 ADC 模数转换器的使用，ADC 产生电压，通过电压与温度的关系，可以计算出真实的温度。而 PID 温度控制就是就是通过比例，积分，微分的方法来消除误差，比例的作用就是当误差产生时，增大比例系数就可以减小误差，积分的作用就是消除稳态误差，每当误差产生时就可以随时消除，而微分的作用就是让温度值更加的接近真实值。

通过这三个实验，我对于单片机从最初的懵懂到现在的基本理解，

经历了很多，同时对于单片机也产生了更多的兴趣，相信经过以后的进一步学习，可以接触到更多的关于单片机的知识，也谢谢老师的悉心教导。