

```

#include <iom16v.h>
#include <macros.h>
#define uchar unsigned char
#define uint unsigned int
//头文件, 宏定义
const uchar table[]={'X','X','.','X'};
// 1602液晶第一行显示的内容放入数组table
const uchar table1[]="SUMHS";
// 1602液晶第二行显示的内容放入数组table1
uchar temp[]={0,1,2};
//温度传感器获取的温度值存入数组temp
void delayus(uint US)
{
    uint i;
    US=US*5/4;                                //5/4是在8MHz晶振下, 通过软件仿真反复
    实验得到的数值
    for( i=0;i<US;i++);
}
//微秒延时函数
void delayms(uint ms)
{
    uint i,j;
    for(i=0;i<ms;i++)
    {
        for(j=0;j<1141;j++);
    }
}
//毫秒延时函数
void write_com(uchar com)
{
    PORTD&=~BIT(4);
    PORTD&=~BIT(5);
    PORTB=com;
    PORTD|=BIT(6);
    delayms(1);
    PORTD&=~BIT(6);
}
// 1602液晶写指令函数
void write_dat(uchar dat)
{
    PORTD|=BIT(4);
    PORTD&=~BIT(5);
    PORTB=dat;
    PORTD|=BIT(6);
    delayms(1);
    PORTD&=~BIT(6);
}
//1602液晶写数据函数
void B20_init(void)
{
    DDRA|=BIT(5);                            //配置为输出

```

```

PORTA&=~BIT(5);          //拉低
delayus(600);             //等待600微秒
PORTA|=BIT(5);            //释放总线
delayus(60);              //等待60微秒
DDRA&=~BIT(5);            //配置为输入
while((PINA&(BIT(5)))); //等待DS18B20拉低
while(!(PINA&(BIT(5)))); //等待DS18B20释放总线
}
//温度传感器18B20初始化函数
uchar B20_readB(void)
{
    uchar i,retd=0;
    for(i=0;i<8;i++)      //位计数值
    {
        retd>>=1;        //右移，准备接受新的数据位
        DDRA|=BIT(5);     //配置为输出
        PORTA&=~BIT(5);   //拉低，启动读数据位
        PORTA|=BIT(5);    //释放总线
        delayus(5);       //等待5微秒
        DDRA&=~BIT(5);   //配置为输入，开始读取数据位
        if(PINA&BIT(5))   //该位是否为高
        {
            retd|=0x80;    //是就将此位置高
        }
        delayus(50);      //等待50微秒
    }
    return retd;          //将读到的一个字节返
回
}
//温度传感器18B20读数据函数
void B20_writeB(uchar wrd)
{
    uchar i;
    for(i=0;i<8;i++)      //位计数值
    {
        DDRA|=BIT(5);     //配置为输出
        PORTA&=~BIT(5);   //拉低，启动写数据位
        delayus(1);       //等待1微秒
        if(wrd&0x01)       //此位数据是否为高
        {
            PORTA|=BIT(5); //是高则将单总线拉高
        }
        else
        {
            PORTA&=~BIT(5); //是低则将单总线拉低
        }
        delayus(50);      //等待50微秒
    }
}

```

```

    PORTA|=BIT(5);                //释放总线
    wrd>>=1;                      //右移, 为
写入新的数据位做准备
}
    delayus(50);                  //等待50微秒
}
//温度传感器18B20写数据函数
uint Read_temp(void)
{
    uchar templ,temph;
    uint temp;
    B20_init();                   //初始化, 每次写命令都从初始化开始
    B20_writeB(0xcc);             //跳过ROM
    B20_writeB(0x44);             //启动温度转换
    B20_init();                   //初始化, 每次写命令都从初始化开始
    B20_writeB(0xcc);             //跳过ROM
    B20_writeB(0xbe);             //读寄存器
    templ=B20_readB();            //读温度低字节
    temph=B20_readB();            //读温度高字节
    temp=templ+temph*256;         //将温度整理成16位变量
    return temp;                  //返回16位变量
}
//读取当前温度函数, 返回16位变量
uchar* Num_BCD(uint num)
{
    uchar i,chr[3];
    uchar *rept;
    rept=&(chr[0]);               //返回指针指向BCD码数组
    for(i=0;i<3;i++)
    {
        chr[2-i]=num%10;          //对10取余数 (其实是求模, 但是对于正数, 取余与求
模是相等的)
        num/=10;                  //除以10, 为取出下一位做准备
    }
    return rept;                  //返回指针
}
//BCD码转换函数
void main()
{
    uint t;
    uchar *temppt;
    uchar i;
        DDRB=0xFF;
        DDRD|=BIT(4)|BIT(5)|BIT(6);
        PORTD&=~BIT(6);

        write_com(0x38);
        delayms(5);
        write_com(0x01);
        delayms(5);

```

```

        write_com(0X0C);
        delayms(5);
        write_com(0X06);
        delayms(5);
        write_com(0X80+0X00);
        for(i=0;i<16;i++)
        {
            write_dat(table[i]);
            delayms(5);
        }
        write_com(0X80+0X40);
        delayms(5);
        for(i=0;i<16;i++)
        {
            write_dat(table1[i]);
            delayms(5);
        }
    }
while(1)
{
    t=Read_temp();    //读取温度值
    t*=0.625;          //转换成实际温度的10倍
    temppt=Num_BCD(t); //将实际温度的10倍转换成BCD码
    temp[0]=*(temppt+0);
    temp[1]=*(temppt+1);
    temp[2]=*(temppt+2);
    write_com(0X80+0X00);
    delayms(5);
    write_dat(temp[0]+0x30);
    delayms(5);
    write_dat(temp[1]+0x30);
    delayms(5);
    write_dat(0x2e);
    delayms(5);
    write_dat(temp[2]+0x30);
    delayms(5);
    write_dat(0xdf);
    write_dat(0x43);
    delayms(1);
}

}

```