

河南工业大学

# 实 验 报 告

课程设计名称： 单片机原理与应用

专 业 班 级： 物联网工程 1902 班

小 组 成 员： 201916070216 王源 201916070213 王众

指 导 教 师： 刘宏焕

课程实验时间： 2021 年 5 月 20 日 ~2021 年 6 月 7 日

## **专题题目：基于单片机的温度控制系统的设计**

### **【专题目的】**

为了实现基于 STC89C52 单片机的温度控制与报警系统，能够实现如下功能：

1. 能够准确无误地读取当前温度。
2. 通过温度来调节报警温度区间的上下限，当第一次按下第一个键即 KEY SET 键时，进入调节温度上限模式，然后按下第二个键即 KEY UP 键时，升高温度的上限，按下第三个键即 KEY DOWN 键时，降低温度的下限，第二次按下 KEY SET 键时进入调节温度下限模式，同理第二个键是升高，第三个键是降低。第三次按下 KEY SET 键时显示此时正常的温度。
3. 当目前温度超过设置的报警温度区间时，即超过上限或低于下限时，蜂鸣器发出声响，并且红灯闪烁，数码管闪烁，温度越高，闪烁频率越快，蜂鸣器叫声越急促。
4. 当目前温度处于设置的报警温度区间时，绿灯亮，蜂鸣器不报警，显示当前温度时，绿灯亮。

## 设计简介：

---

本设计是基于单片机的温度控制系统的设计，主要实现以下功能：

- 可实现通过 DS18B20 测得当前的环境温度。
- 可实现通过继电器控制加热片和制冷片，使得温度稳定在设定的上下限范围内。
- 可实现通过按键调整上下限温度阈值。
- 可实现通过 LCD1602 显示当前的温度值及上下限阈值。

## 【应用场域与相关背景知识】

### 概要：

温度是生产、生活及科学研究等方面中的一个重要参数，在很多场合起着极为关键的作用，需要精确控制。因此，高精度温度控制器具有广阔的市场前景和迫切的应用需求。研究和设计了一个由单片机控制的具有一定智能水平的温度控制系统，能够按照实际需要设定温度控制的范围，并根据在温度调整过程中的温度变化情况，输出智能控制信号，实现温度的精确控制。

该设计可以适用于需要温控检测的场所，例如工厂车间或是大楼防火警报，在温度过高或者温度过低时能发出警报提醒工作人员及时检查设备或疏散人群。在温度控制系统中应用较为广泛的场所有食品行业，档案管理，温室大棚，动物养殖，药品存储，烟草行业，工控行业等领域运用广泛。

该设计中使用了 STC89C52 单片机，本来想使用自己的开发板的，但是因为开发板是已经集成好了的，在开发设计的过程中发现许多的线路冲突，比如数码管与 LED 灯会冲突，外部中断 0 与独立按键冲突等，并且我使用到的元器件在开发板上也没有焊接，自己手动买元器件焊板子很复杂。结合多种因素考虑，我选择利用了 Proteus 8 Professional 来完成我的课程设计。

如下是本设计所用到的硬件清单：

名称	型号	数量
单片机	STC89C52	1
显示屏	LCD1602	1
测温模块	DS18B20	1
继电器	/	2
加热片	/	1
制冷片	/	1
蜂鸣器	/	1
独立按键	/	3
杜邦线	/	若干

所需硬件的基本介绍：

1.单片机芯片→STC89C52:

STC89C52 是 [STC](#) 公司生产的一种低功耗、高性能 CMOS8 位微控制器，具有 8K 字节系统可编程 [Flash 存储器](#)。STC89C52 使用经典的 MCS-51 内核，但是

做了很多的改进使得芯片具有传统的 51 单片机不具备的功能。在单芯片上，拥有灵巧的 8 位 CPU 和在系统可编程 Flash，使得 STC89C52 为众多嵌入式控制应用系统提供高灵活、有效的解决方案。

具有以下标准功能： 8k 字节 Flash，512 字节 RAM， 32 位 I/O 口线，看门狗定时器，内置 4KB EEPROM，MAX810 复位电路，3 个 16 位定时器/计数器，4 个外部中断，一个 7 向量 4 级中断结构（兼容传统 51 的 5 向量 2 级中断结构），全双工串行口。另外 STC89C52 可降至 0Hz 静态逻辑操作，支持 2 种软件可选择节电模式。空闲模式下，CPU 停止工作，允许 RAM、定时器/计数器、串口、中断继续工作。掉电保护方式下，RAM 内容被保存，振荡器被冻结，单片机一切工作停止，直到下一个中断或硬件复位为止。最高运作频率 35MHz，6T/12T 可选。

## 2.显示屏→LCD1602:

### 2.1 介绍

LCD1602 液晶显示器是广泛使用的一种字符型液晶显示模块。它是由字符型液晶显示屏（LCD）、控制驱动主电路 HD44780 及其扩展驱动电路 HD44100，以及少量电阻、电容元件和结构件等装配在 PCB 板上而组成。不同厂家生产的 LCD1602 芯片可能有所不同，但使用方法都是一样的。为了降低成本，绝大多数制造商都直接将裸片做到板子上。

### 2.2 引脚功能：

表1 LCD引脚功能表					
编号	符号	引脚说明	标号	符号	引脚说明
1	VSS	电源地	9	D2	数据
2	VDD	电源正极	10	D3	数据
3	VL	液晶显示偏压	11	D4	数据
4	RS	数据/命令选择	12	D5	数据
5	R/W	读/写选择	13	D6	数据
6	E	使能信号	14	D7	数据
7	D0	数据	15	BLA	背光源正极
8	D1	数据	16	BLK	背光源负极

### 2.3 指令集：

表2 LCD1602控制指令											
序号	指令	RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0
1	清屏	0	0	0	0	0	0	0	0	0	1
2	光标复位	0	0	0	0	0	0	0	0	1	x
3	输入方式设置	0	0	0	0	0	0	0	1	I/D	S
4	显示开关控制	0	0	0	0	0	0	1	D	C	B
5	光标或字符移位控制	0	0	0	0	0	1	S/C	R/L	x	x
6	功能设置	0	0	0	0	1	DL	N	F	x	x
7	字符发生存储器地址设置	0	0	0	1	字符发生存储器地址					
8	数据存储器地址设置	0	0	1	显示数据存储器地址						
9	读忙标志或地址	0	1	BF	计数器地址						
10	写入数据至CGRAM或DDRAM	1	0	要写入的数据内容							
11	从CGRAM或DDRAM中读取数据	1	1	读取的数据内容							

2.4 连接方式：

LCD1602与单片机的连接有两种方式，一种是直接控制方式，另一种是所谓的间接控制方式。它们的区别只是所用的数据线的数量不同，其他都一样。

**1.直接控制方式**

LCD1602的8根数据线和3根控制线E，RS和RW与单片机相连后即可正常工作。一般应用中只须往LCD1602中写入命令和数据，因此，可将LCD1602的RW读/写选择控制端直接接地，这样可节省1根数据线。VO引脚是液晶对比度调试端，通常连接一个10kΩ的电位器即可实现对对比度的调整；也可采用将一个适当大小的电阻从该引脚接地的方法进行调整，不过电阻的大小应通过调试决定。

**2.间接控制方式**

间接控制方式也称为四线制工作方式，是利用HD44780所具有的4位数据总线的功能，将电路接口简化的一种方式。为了减少接线数量，只采用引脚DB4~DB7与单片机进行通信，先传数据或命令的高4位，再传低4位。采用四线并口通信，可以减少对微控制器I/O的需求，当设计产品过程中单片机的I/O资源紧张时，可以考虑使用此方法。 [1]

3.测温模块→ DS18B20:

**3.1 介绍：**

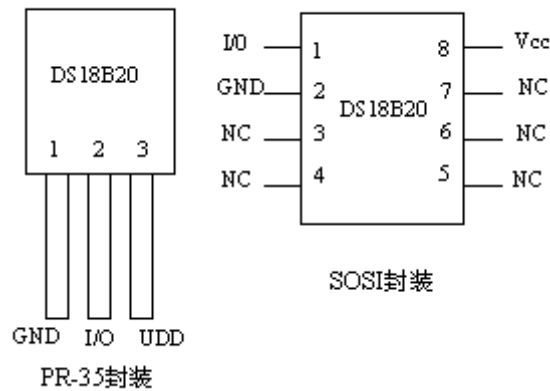
DS18B20 是常用的数字温度传感器，其输出的是数字信号，具有体积小，硬件开销低，抗干扰能力强，精度高的特点。 [1] DS18B20 数字温度传感器接线方便，封装成后可应用于多种场合，如管道式，螺纹式，磁铁吸附式，不锈钢封装式，型号多种多样，有 LTM8877，LTM8874 等等。

主要根据应用场合的不同而改变其外观。封装后的 DS18B20 可用于电缆沟测温，高炉水循环测温，锅炉测温，机房测温，农业大棚测温，洁净室测温，弹药库测温等各种非极限温度场合。耐磨耐碰，体积小，使用方便，封装形式多样，适用于各种狭小空间设备数字测温和控制领域。

### 3.2 外形和内部结构:

DS18B20 内部结构主要由四部分组成: 64 位光刻 ROM、温度传感器、非挥发的温度报警触发器 TH 和 TL、配置寄存器。

DS18B20 的外形及管脚排列如下:



### 3.3 DS18B20 引脚定义:

- (1)DQ 为数字信号输入/输出端;
- (2)GND 为电源地;
- (3)VDD 为外接供电电源输入端 (在寄生电源接线方式时接地)。

### 3.4 DS18B20 工作原理:

DS18B20 的读写时序和测温原理与 DS1820 相同, 只是得到的温度值的位数因分辨率不同而不同, 且温度转换时的延时时间由 2s 减为 750ms。高温度系数晶振 随温度变化其振荡率明显改变, 所产生的信号作为计数器 2 的脉冲输入。计数器 1 和温度寄存器被预置在  $-55^{\circ}\text{C}$  所对应的一个基数值。计数器 1 对 低温度系数晶振产生的脉冲信号进行减法计数, 当计数器 1 的预置值减到 0 时, 温度寄存器的值将加 1, 计数器 1 的预置将重新被装入, 计数器 1 重 新开始对低温度系数晶振产生的脉冲信号进行计数, 如此循环直到计数器 2 计数到 0 时, 停止温度寄存器值的累加, 此时温度寄存器中的数值即 为所测温度。图 3 中的斜率累加器用于补偿和修正测温过程中的非线性, 其输出用于修正计数器 1 的预置值。

这里还需要继电器两个, 加热片, 制冷片, 蜂鸣器各一个, 独立按键三个, 杜邦线若干, 型号都与不做要求, 故这里不再展开讨论。

## 【设计的基本思路】

### 1.温控系统硬件设计：

温度控制的基本原理是在需要进行温度控制的场合用传感器测量其温度值，与控制器内存储的温度值进行比较，当测得的温度高于或低于设定值时，启动加热或降温设备，使温度回归到设定值范围内。

#### 1.1 系统总体结构设计：

本温控箱以单片机 STC89C52 作为温控中心，用温度传感器 DS18B20 作为温度测量单元，将采集的温度值经过串行通信方式传输到温控中心进行判断，并进行智能处理。当测得的温度  $T$  低于设定的最低温度  $T_L$  时，单片机发出控制信号，启动加热器件；当测得的温度  $T$  高于设定的温度  $T_H$  时，单片机发出控制信号，启动降温器件，将温度保持在设定的范围内，完成温控工作。本温控器带有 LCD1602 显示模块和按键输入模块，可显示实时温度值和现场设定温度控制范围。温控系统主要由温度检测模块、单片机控制模块、温度显示模块、温控执行模块（继电器及加热、降温器件）等部分组成。



## 1.2 温度检测单元设计

为提高测温精度，降低成本，本温控箱采用较成熟的 DS18B20 温度传感器来完成温控箱内部和外部的温度检测。DS18B20 是由 Dallas 公司生产的一线式数字温度传感器，它将温度感测、信号变换、数据存储、A/D 转换等功能集成于一体，其温度检测范围宽，达到  $-55^{\circ}\text{C} \sim +125^{\circ}\text{C}$ ，可以用一线总线方式连接微处理器，以编程方式（9~12 位）转换精度，测温分辨率达  $0.0625^{\circ}\text{C}$ 。DS18B20 温度传感器的工作电源可从外部输入，也可采用寄生电源方式工作；多个 DS18B20 可以并联连接到 CPU，实现多个 DS18B20 与 CPU 的通信，因此连线少，可节省引线 and 逻辑电路，减少 CPU 端口的占用，但以增加软件复杂性为代价，对读写的数据位有着严格的时序要求。

DS18B20 温度传感器具有体积小、功能强、精度高、连接方便、抗干扰性好等优点，在工业控制、智能家居等环境中得到较广泛的应用。

## 1.3 温度控制执行部分设计：

由于单片机的输出功率较小，不宜直接驱动继电器，否则会造成单片机功耗过大，加重单片机内部电源的负担，易导致单片机工作不稳定。为安全平稳控制继电器，本温控系统采用固态继电器 SSR-40DA，固态继电器也称作固态开关 SSR（Solid State Relay），它是利用现代微电子技术 with 电力电子技术相结合而发展起来的一种新型无触点电子开关，集光电耦合、大功率双向晶闸管及触发电

路、阻容吸收回路于一体，用于代替传统的电磁式继电器，实现对单相或者三相电动机的正反转控制，或者其他控制。无触点无动作噪音，具有开关速度快、无火花干扰和可靠性高等优点。

#### 1.4 温度显示模块：

温度显示模块采用 1602 型字符型液晶显示器。1602 型显示器具有功耗低、体积小、显示内容丰富、超薄轻巧等优点，在袖珍式仪表和低功耗应用系统中应用广泛，是一种专门用于显示字母、数字、符号等点阵式的 LCD，显示的格式为 16×2 行。在模块内部已经存储了 160 个不同的点阵字符图形，这些字符包括：英文字母的大小写、阿拉伯数字、常用的符号等，每一个字符都有一个固定的代码。

#### 1.5 加热 / 降温执行模块：

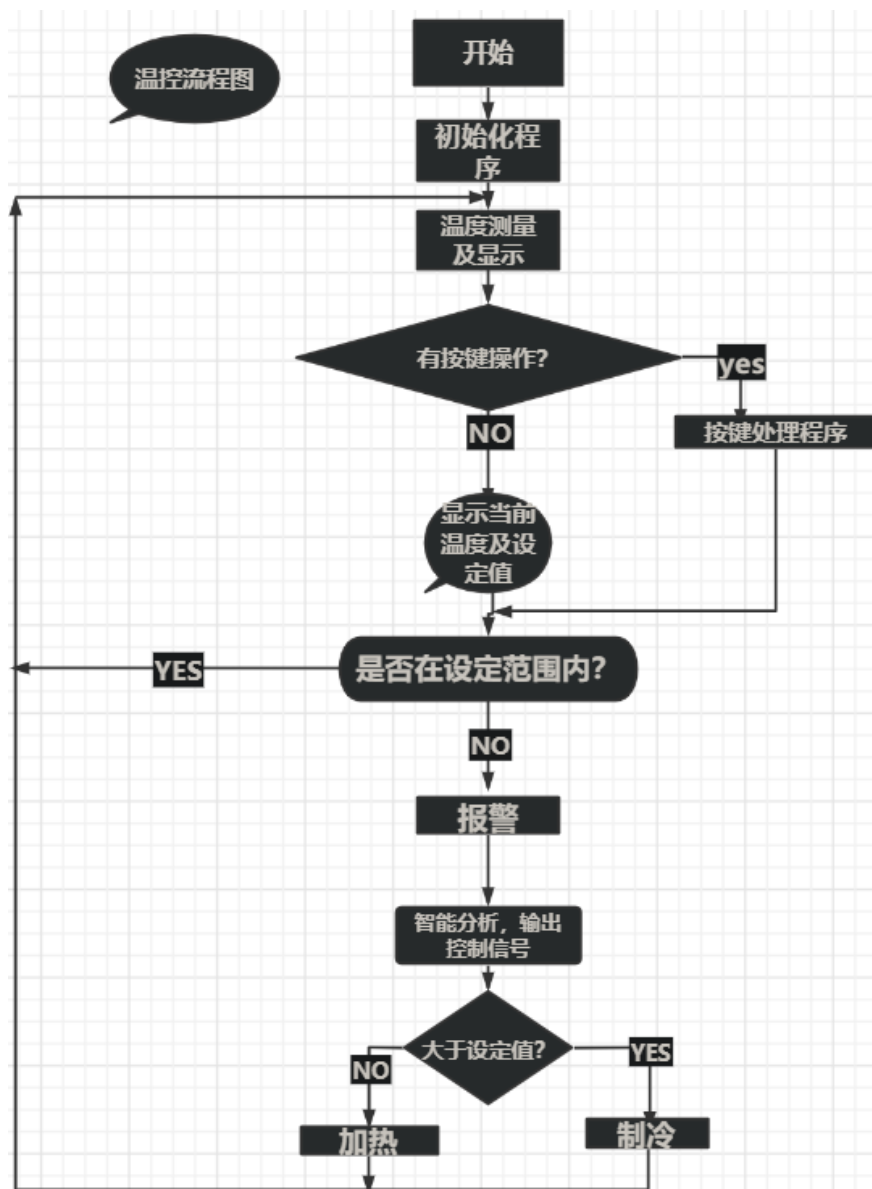
当单片机检测到温度不在调控范围以内时，需要启动加热或降温器件使温度回到温控范围内。一般加热的方式为电热丝和风扇，本系统用加热片为加热器件，以降温片为降温器件。为使温度变化过程平稳，通常要对加热或降温器件的功率进行调整。功率调整的方法一般用可控硅，具体的方式有调相和 PWM。调相就是调整加在负载上的电压的导通角，PWM 是通过调整单位时间内加在负载上的电压次数来改变负载功率。为降低对电网的污染和对其他用电器件的干扰，本系统采用 PWM 方式对温控器件进行调整。

### 1.6 报警电路：

本温控箱采用声光报警方式进行异常状态报警,以晶体管和蜂鸣器构成声音报警电路,以红、绿色发光二极管构成光线报警电路。在系统正常工作时,只有绿色发光二极管点亮;当系统测得的温度超出设定的温度范围,绿色发光二极管熄灭,红色二极管点亮,同时由单片机控制蜂鸣器发出报警声,数秒后停止声音报警。

## 2．温控系统软件设计：

### 2.1 控制流程图：



## 2.2 软件设计：

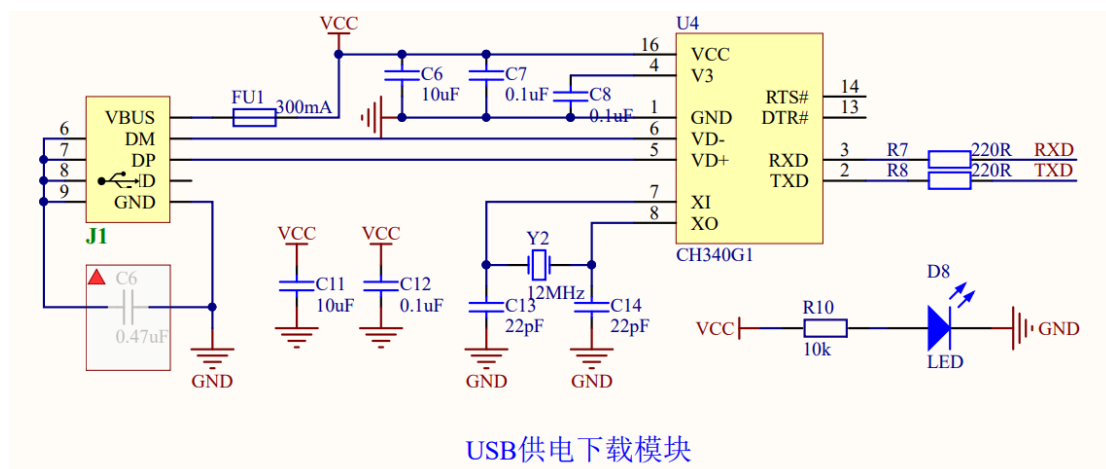
在进行智能控制时，根据设定的温度范围及探测到的系统温度，确定加热或降温时的结束温度，如果环境温度高于设定温度的上限，则降温器件停止工作时的温度由单片机根据公式  $T_h - 0.8(T_h - T_l)$  计算出来。当环境温度低于设定温度的下限，则加热器件停止工作时的温度由单片机根据公式  $T_l + 0.8(T_h - T_l)$  计算出来。

当环境温度在设定温度的上、下限之间，则降温器件停止工作时的温度由单片机根据公式  $T_l + 0.5 (T_h + T_l)$  计算出来，通过此种方式进行温度调控，能有效减少加热或降温器件的启停次数，延长系统寿命，同时也使温度变化过程更平稳。在调温过程中以 PID 方式对系统温度进行控制，即在控制过程中，将测得实际温度值与设定值进行比较，经单片机计算后得到温度的偏差值、偏差变化率等，根据温度值、偏差值、偏差变化率算出控制增量，以控制加热器件或风扇的导通时间，达到温度控制的目的。

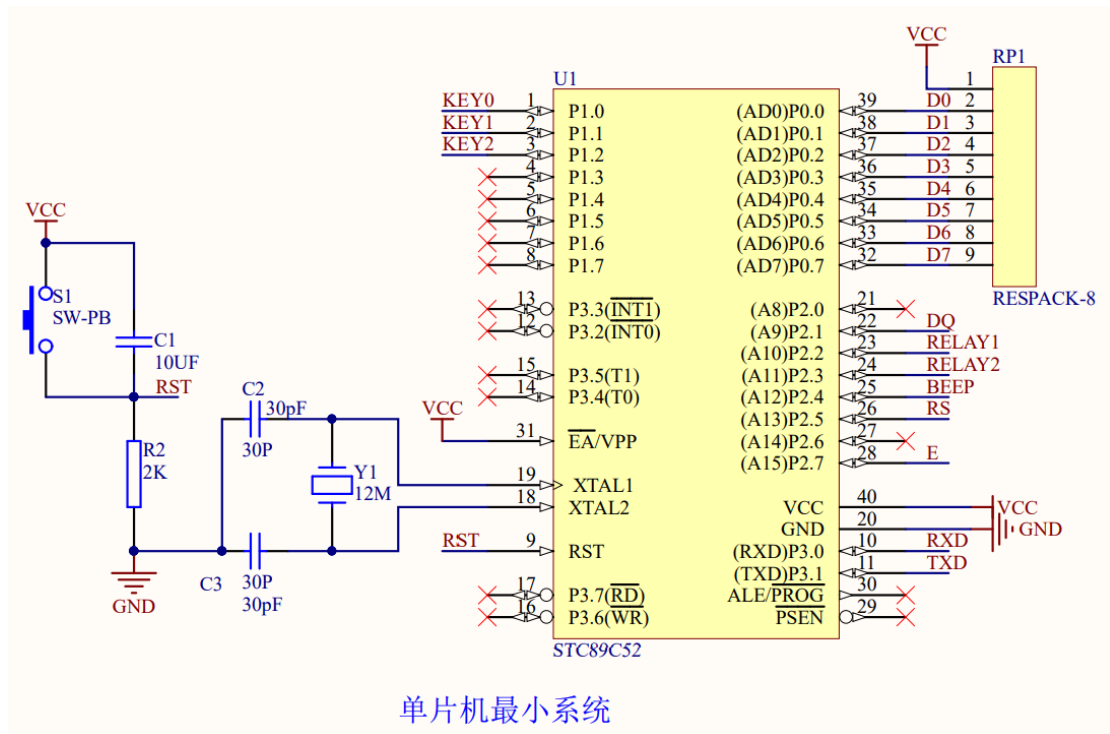
报警程序用于输出报警信号，控制报警电路实现声光报警。

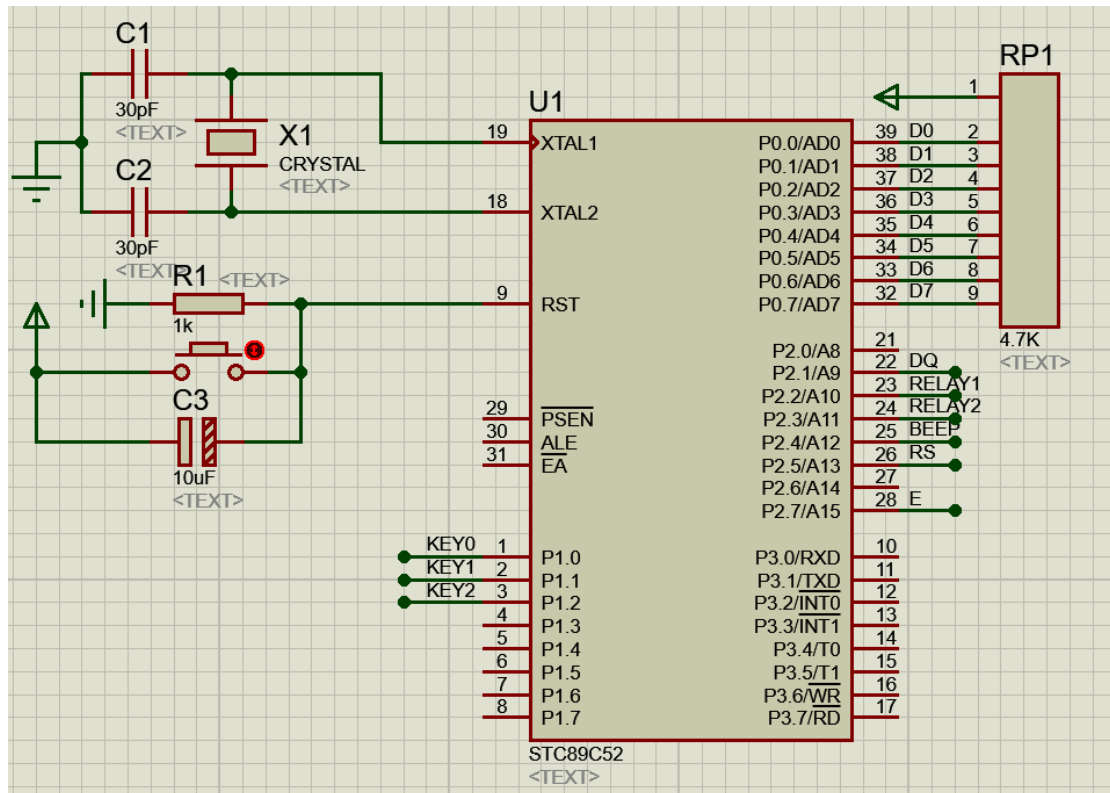
## 【各模块代码及分析】

### USB 的供电下载原理图：



### 单片机的最小系统原理图：





该最小系统包含了 stc89c52 主芯片及外部晶振电路与外部复位电路。三个部分组成我们单片机的最小系统。

### 1. 初始设定代码：

```
1 #include <reg52.h>
2 #define uint unsigned int
3 #define uchar unsigned char //宏定义
4 #define LCD1602 P0
5 sbit SET=P1^0; //定义调整键
6 sbit DEC=P1^1; //定义减少键
7 sbit ADD=P1^2; //定义增加键
8 sbit BUZZ=P2^4; //定义蜂鸣器
9 sbit ALAM=P2^3; //定义继电器
10 sbit ALAM1=P2^2;
11 sbit DQ=P2^1; //定义DS18B20总线I/O
12 sbit RS = P2^5;
13 sbit EN = P2^7;
14 bit shanshuo_st; //闪烁间隔标志
15 bit beep_st; //蜂鸣器间隔标志
16 uchar x=0; //计数器
17
18 uchar code tab1[]={"Now Tem: . C "};
19 uchar code tab2[]={"TH: C TL: C"};
20 uint c;
21 uchar Mode=0; //状态标志
22 signed char TH=40; //上限报警温度，默认值为40
23 signed char TL=10; //下限报警温度，默认值为10
```

## 2.DS18B20 模块的延时子程序：

```
//=====DS18B20=====
//=====DS18B20=====
//=====DS18B20=====
/*****延时子程序*****/
void Delay_DS18B20(int num)
{
    while(num--);
}
void delay(uint xms)//延时函数，有参函数
{
    uint x,y;
    for(x=xms;x>0;x--)
        for(y=110;y>0;y--);
}
```

## 3.初始化 DS18B20：



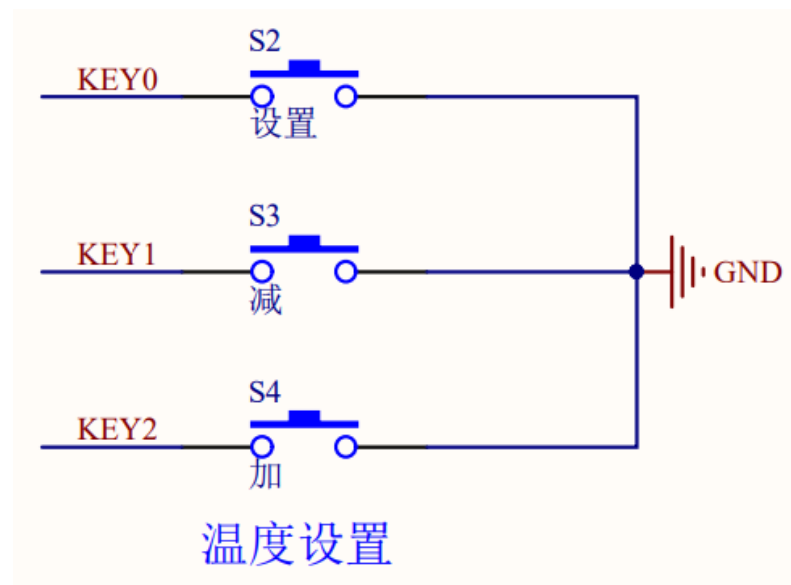
```

/*****初始化DS18B20*****/
void Init_DS18B20(void)
{
    unsigned char x=0;
    DQ = 1;          //DQ复位
    Delay_DS18B20(8); //稍做延时
    DQ = 0;          //单片机将DQ拉低
    Delay_DS18B20(80); //精确延时，大于480us
    DQ = 1;          //拉高总线
    Delay_DS18B20(14);
    x = DQ;          //稍做延时后，如果x=0则初始化成功，x=1则初始化失败
    Delay_DS18B20(20);
}

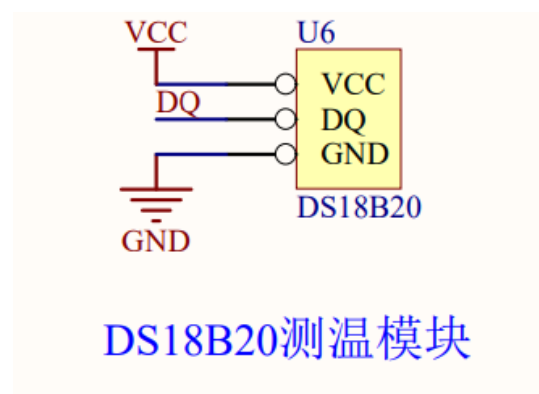
```

#### 4. 读写字节+读温度模块：

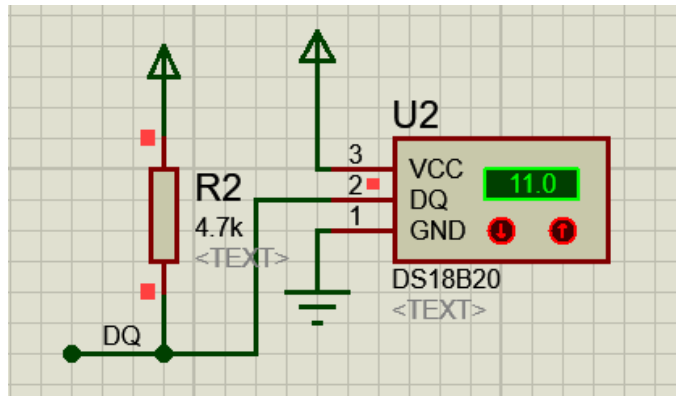
温度设置原理图：



测温模块原理图：



DS18B20 proteus 仿真：



Ds18b20 用来检测温度的值。

```
/******读一个字节*****/  
unsigned char ReadOneChar(void)  
{  
    unsigned char i=0;  
    unsigned char dat = 0;  
    for (i=8;i>0;i--)  
    {  
        DQ = 0;        // 给脉冲信号  
        dat>>=1;  
        DQ = 1;        // 给脉冲信号  
        if(DQ)  
            dat|=0x80;  
        Delay_DS18B20(4);  
    }  
    return(dat);  
}
```

```

/*****写一个字节*****/
void WriteOneChar(unsigned char dat)
{
    unsigned char i=0;
    for (i=8; i>0; i--)
    {
        DQ = 0;
        DQ = dat&0x01;
        Delay_DS18B20(5);
        DQ = 1;
        dat>>=1;
    }
}

```

```

/*****读取温度*****/
unsigned int ReadTemperature(void)
{
    unsigned char a=0;
    unsigned char b=0;
    unsigned int t=0;
    float tt=0;
    Init_DS18B20();
    WriteOneChar(0xCC); //跳过读序号列号的操作
    WriteOneChar(0x44); //启动温度转换
    Init_DS18B20();
    WriteOneChar(0xCC); //跳过读序号列号的操作
    WriteOneChar(0xBE); //读取温度寄存器
    a=ReadOneChar(); //读低8位
    b=ReadOneChar(); //读高8位
    t=b;
    t<<=8;
    t=t|a;
    tt=t*0.0625;
    // t= tt*10+0.5; //放大10倍输出并四舍五入
    t= tt*10+0.5;
    return(t);
}

```

```

/****读取温度****/
void check_wendu(void)
{
    c=ReadTemperature()-5;          //获取温度值并减去DS18B20的温漂误差
    if(c>1200)
        c=1200;
}

```

## 5. 写入数据模块：

```

/*****液晶写入指令函数与写入数据函数，以后可调用*****/

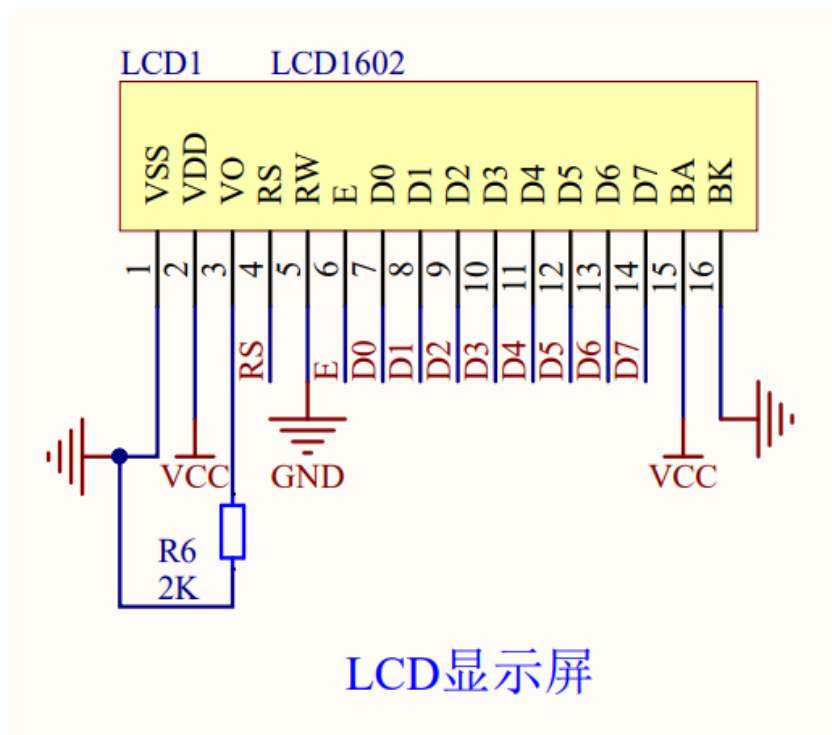
void write_1602com(uchar com)****液晶写入指令函数****
{
    RS=0; //数据/指令选择置为指令
    // rw=0; //读写选择置为写
    LCD1602=com; //送入数据
    delay(1);
    EN=1; //拉高使能端，为制造有效的下降沿做准备
    delay(1);
    EN=0; //en由高变低，产生下降沿，液晶执行命令
}

void write_1602dat(uchar dat)****液晶写入数据函数****
{
    RS=1; //数据/指令选择置为数据
    // rw=0; //读写选择置为写
    LCD1602=dat; //送入数据
    delay(1);
    EN=1; //en置高电平，为制造下降沿做准备
    delay(1);
    EN=0; //en由高变低，产生下降沿，液晶执行命令
}

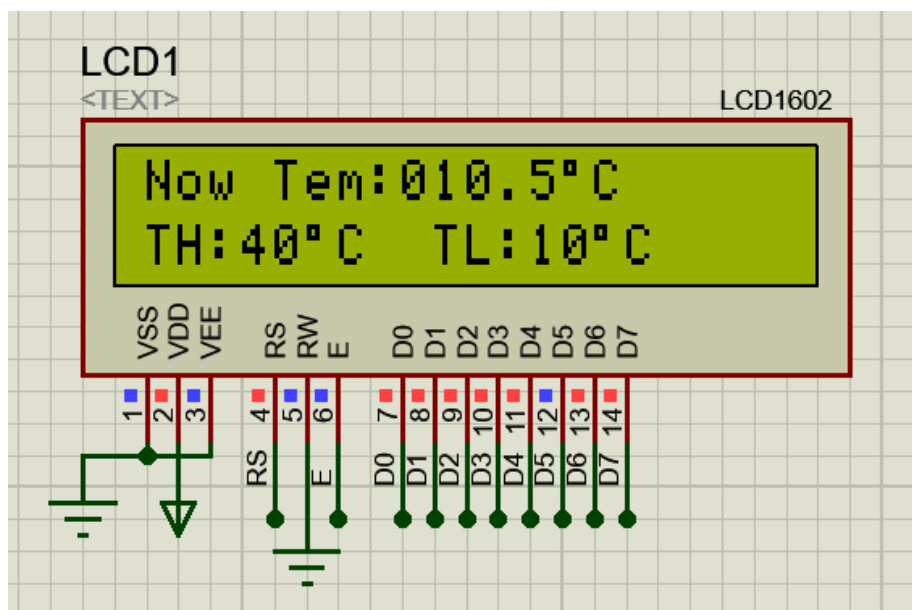
```

## 6. 初始化 lcd 模块：

### Lcd1602 原理图：



Lcd1602 显示屏 proteus 仿真：



```

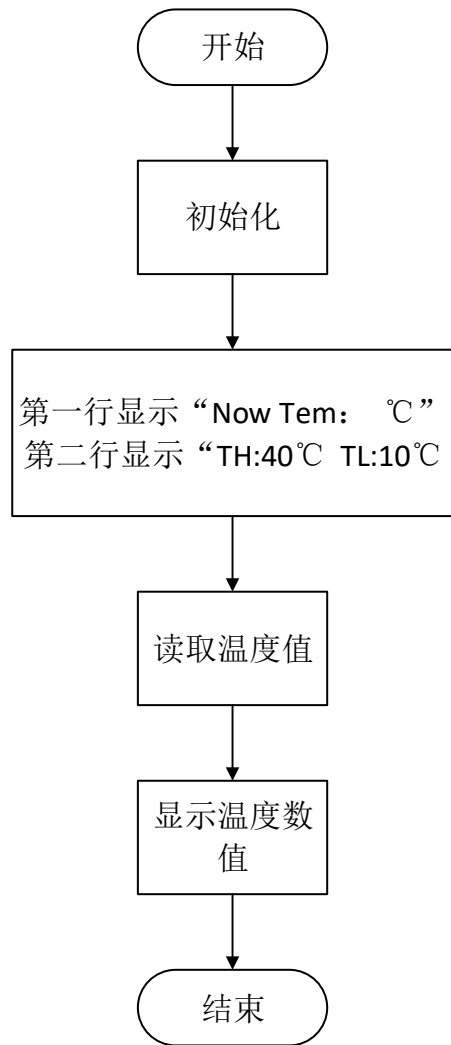
void lcd_init()/**液晶初始化函数***/
{
    uchar a;
    write_1602com(0x38);//设置液晶工作模式，意思：16*2行显示，5*7点阵，8位数据
    write_1602com(0x0c);//开显示不显示光标
    write_1602com(0x06);//整屏不移动，光标自动右移
    write_1602com(0x01);//清显示

    write_1602com(0x80);//日历显示固定符号从第一行第1个位置之后开始显示
    for(a=0;a<16;a++)
    {
        write_1602dat(tab1[a]);//向液晶屏写日历显示的固定符号部分
        delay(3);
    }
    write_1602com(0x80+0x40);//时间显示固定符号写入位置，从第2个位置后开始显示
    for(a=0;a<16;a++)
    {
        write_1602dat(tab2[a]);//写显示时间固定符号，两个冒号
        delay(3);
    }
}

```

## 7.显示模块：

初始化 LCD1602 刚开始第一行显示“Now Tem: °C” 第二行显示“TH:40°C TL:10°C”。后面读取实时环境温度值后填入第一行的冒号后，第二行的上下限值显示可通过按键进行调整。



(LCD 显示运行流程图)

```

void display()
{
    if (Mode==0)
    {
        write_1602com(0x80+8);
        write_1602dat(c/1000+0x30);
        write_1602dat((c%1000)/100+0x30);
        write_1602dat(((c%1000)%100)/10+0x30);
        write_1602com(0x80+12);
        write_1602dat(((c%1000)%100)%10+0x30);
        write_1602com(0x80+13);
        write_1602dat(0xdf);
        write_1602com(0x80+0x40+3);
        write_1602dat(TH/10+0x30);
        write_1602dat(TH%10+0x30);
        write_1602dat(0xdf);
        write_1602com(0x80+0x40+12);
        write_1602dat(TL/10+0x30);
        write_1602dat(TL%10+0x30);
        write_1602dat(0xdf);
    }
}

```

## 8. 初始化定时器模块：

```

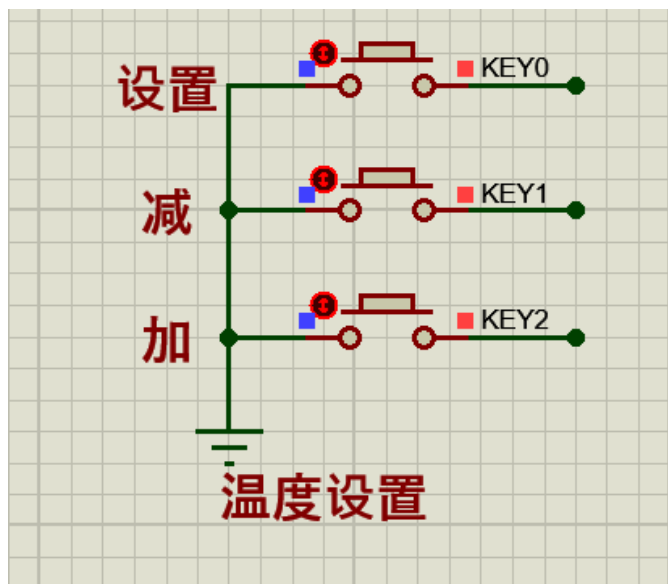
/*****初始化定时器0*****/
void InitTimer(void)
{
    TMOD=0x1;
    TH0=0x3c;
    TL0=0xb0;    //50ms（晶振12M）
    EA=1;        //全局中断开关
    TR0=1;
    ET0=1;        //开启定时器0
}

```

## 9. 设置功能键模块：

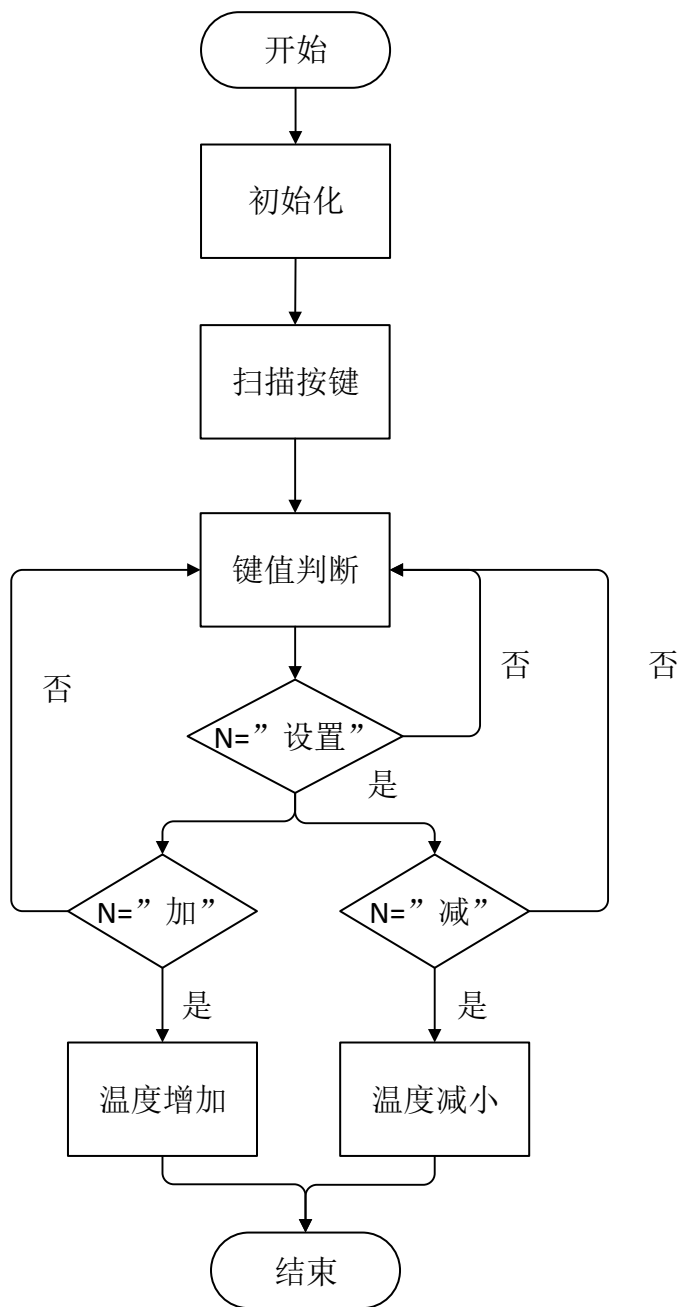


温度设置器的 proteus 仿真：



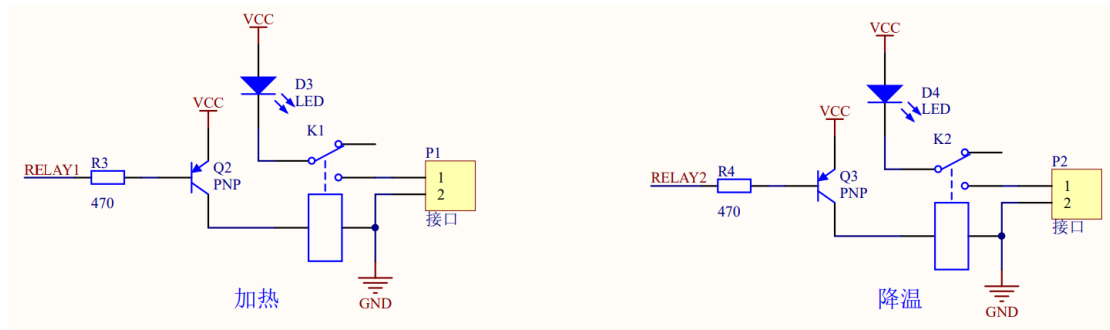
通过设置按键调节光标移动到 TL 或者 TL 范围内，通过加或减设置温度上限跟温度下限，超出温度上限蜂鸣器会报警，继电器会控制制冷片工作以降低温度；同理，低于温度下限蜂鸣器也报警，继电器会控制加热作以升高温度。

初始化按键，如果检测到“设置”按键按下，则进入修改最高温度值和最低温度值模式，此模式通过按键“加”和按键“减”调节最高温度值和最低温度值，再次按下“设置”按键，则退出修改模式。

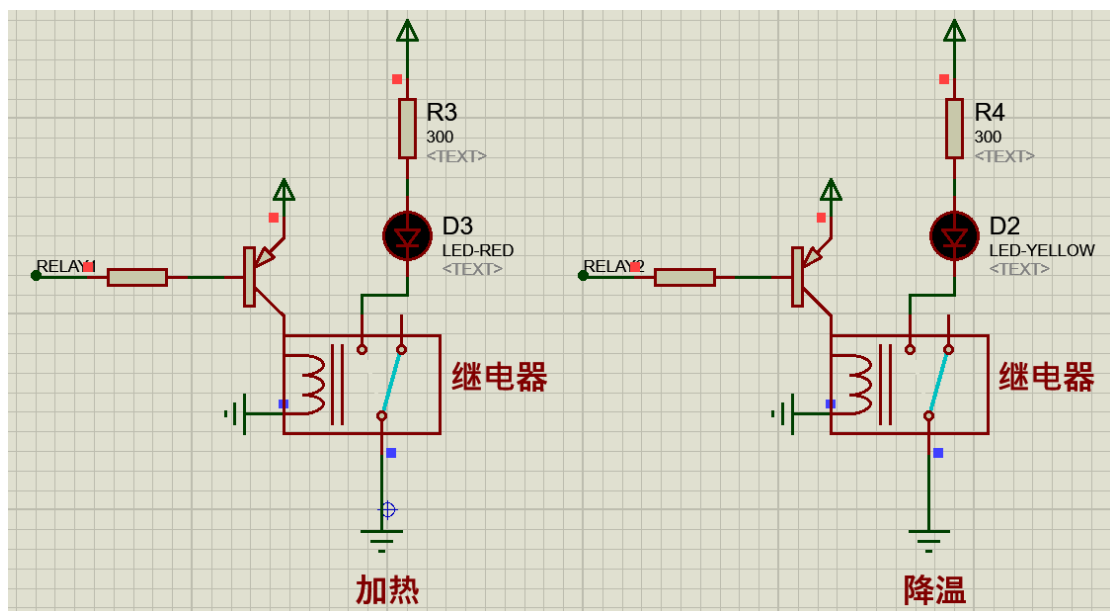


( 上下限温度阈值调整流程图 )

## 加热制冷原理图：



## 继电器 proteus 仿真：



这两部分主要是通过两个继电器来控制外围的加热片及制冷片工作，起到加热或制冷的作用。

```

void KEY()
{
    //功能键
    if (SET==0)
    {
        BUZZ=0;
        delay(10);
        if (SET==0)
        {
            Mode++;
            if (Mode==3)
            {
                Mode=0;
                BUZZ=1;
            }
        }
        while (SET==0)
        {
            if (Mode==0)
            {
                // write_1602com(0x80+0x40+6);
                write_1602com(0x0c);
            }
            else if (Mode==1)
            {
                write_1602com(0x80+0x40+4);
                write_1602com(0x0f);
            }
            else
            {
                write_1602com(0x80+0x40+13);
                write_1602com(0x0f);
            }
        }
    }
}

```

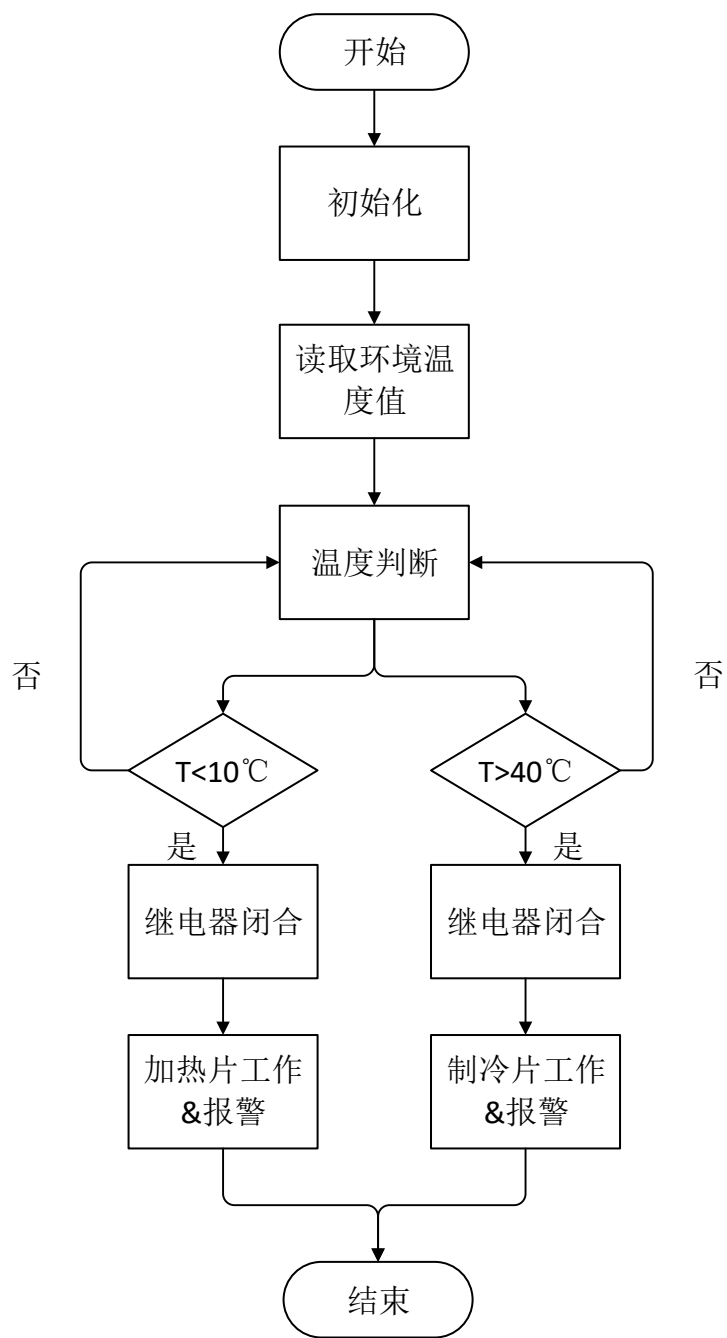
```

//增加
if (ADD==0&&Mode==1)
{
    BUZZ=0;
    delay(10);
    if (ADD==0)
    {
        TH++;
        if (TH>=99)
            TH=99;
        write_1602com(0x80+0x40+3);
        write_1602dat(TH/10+0x30);
        write_1602dat(TH%10+0x30);
        write_1602com(0x80+0x40+4);
        BUZZ=1;
    }
    while (ADD==0);
}

```

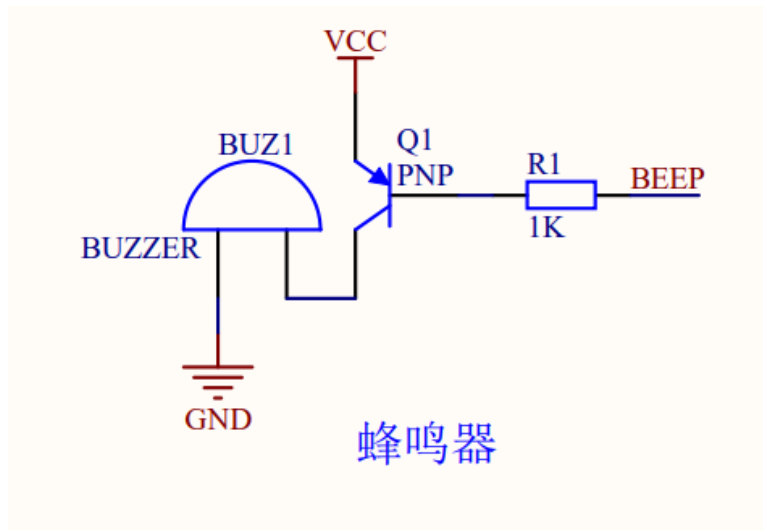
## 10. 报警子程序模块：

初始化 DS18B20 温度传感器，读取温度值，当室内环境温度小于 10℃时，继电器 1 闭合，加热片工作，蜂鸣器报警；当室内环境温度大于 40℃时，继电器 2 闭合，制冷片工作，蜂鸣器报警。

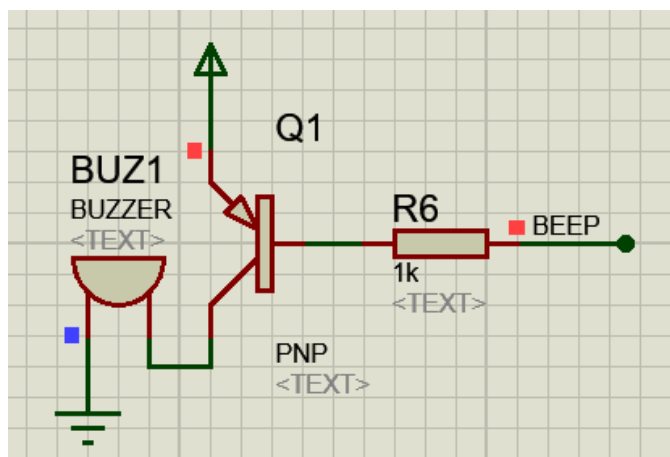


(升降温&报警流程图)

蜂鸣器原理图：



蜂鸣器 proteus 仿真：



当 ds18b20 检测到温度大于 40 摄氏度时或者小于 10 摄氏度时，蜂鸣器会间断地报警。

```

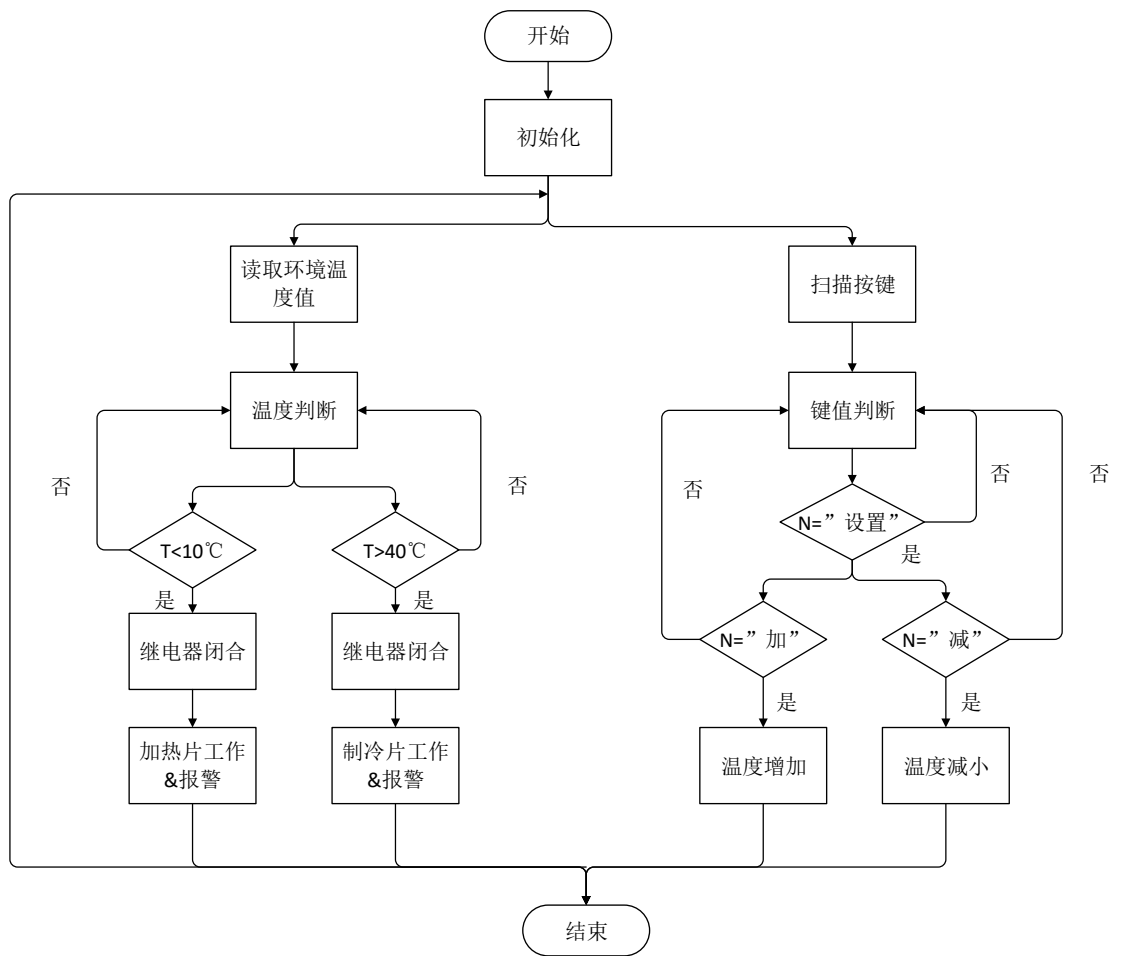
/*****报警子程序*****/
void Alarm()
{
    if(x>=10) {beep_st=~beep_st;x=0;}
    if(Mode==0)
    {
        if((c/10)>=TH)
        {
            ALAM=0;
            ALAM1=1;
            if(beep_st==1)
            BUZZ=0;
            else
            BUZZ=1;
        }
        else if((c/10)<TL)
        {
            ALAM1=0;
            ALAM=1;
            if(beep_st==1)
            BUZZ=0;
            else
            BUZZ=1;
        }
        else
        {
            BUZZ=1;
            ALAM=1;
            ALAM1=1;
        }
    }
}

```

## 11. 主函数模块：

首先对液晶显示、传感器等进行初始化操作，然后通过 DS18B20 温度传感器实时检测室内温度的变化，并且通过液晶屏将温度实时显示出来。当室内环境温度一旦小于 10℃，继电器闭合，加热片工作，蜂鸣器报警；当室内环境温度大于 40℃时，另一个继电器闭合，制冷片工作，蜂鸣器报警。另一方面，如果检测到“设置”按键按下，则进入修改最高温度值和最低温度值模式，此模式通过按键“加”和按键“减”调节最高温度值和最低温度值，再次按下“设置”按键，则退出修改模式。





( 程序总体流程图 )

```

/*****主函数*****/
void main(void)
{
    uint z;
    delay(1);
    lcd_init();
    delay(1);
    InitTimer();    //初始化定时器

    for(z=0;z<100;z++)
    {
        check_wendu();
        delay(1);
    }
    while(1)
    {
        display();
        KEY();
        Alarm();
        check_wendu();
    }
}

/*****定时器0中断服务程序*****/
void timer0(void) interrupt 1
{
    TH0=0x3c;
    TL0=0xb0;
    x++;
}

```

```

,
//减少
if (DEC==0&&Mode==1)
{
    BUZZ=0;
    delay(10);
    if (DEC==0)
    {
        TH--;
        if (TH==TL)
            TH=TL+1;
        write_1602com(0x80+0x40+3);
        write_1602dat(TH/10+0x30);
        write_1602dat(TH%10+0x30);
        write_1602com(0x80+0x40+4);
        BUZZ=1;
    }
    while (DEC==0);
}
if (ADD==0&&Mode==2)
{
    BUZZ=0;
    delay(10);
    if (ADD==0)
    {
        TL++;
        if (TL==TH)
            TL=TH-1;
        write_1602com(0x80+0x40+12);
        write_1602dat(TL/10+0x30);
        write_1602dat(TL%10+0x30);
        write_1602com(0x80+0x40+13);
        BUZZ=1;
    }
,

```

```

,
//减少
if (DEC==0&&Mode==2)
{
    BUZZ=0;
    delay(10);
    if (DEC==0)
    {
        TL--;
        if (TL<=0)
            TL=0;
        write_1602com(0x80+0x40+12);
        write_1602dat(TL/10+0x30);
        write_1602dat(TL%10+0x30);
        write_1602com(0x80+0x40+13);
        BUZZ=1;
    }
    while (DEC==0);
}
}

```

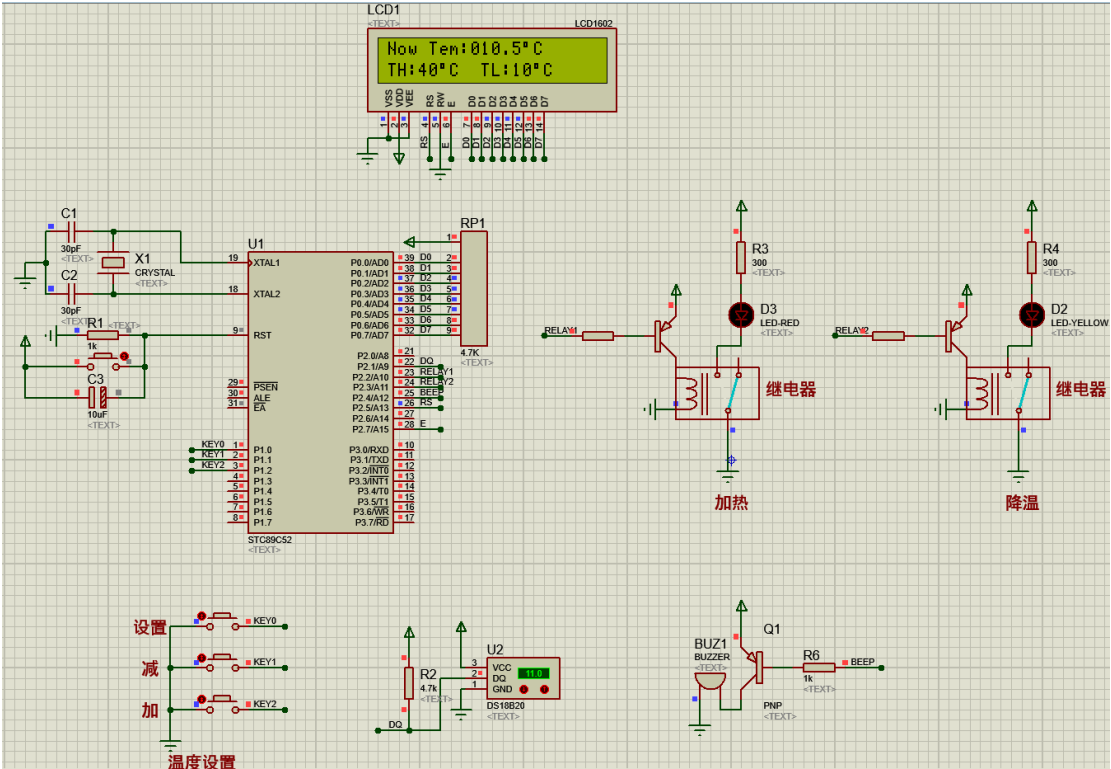
### 【总结】

1. 所有的原理图都是通过 Altium Designer 软件绘制。
2. 所有的仿真图都是通过 proteus 软件绘制。
3. 在Keil5创建好project编辑好.c文件，然后编译生成.hex文件。
4. 在proteus仿真中找到元件并连接好后，双击STC89C52芯片，选择.hex文件。
5. 双击DS18B20设置温度值，先设置为10.5℃
6. 按下KEY0进入温度上限设定模式，按下KEY1上限温度增，按下KEY2上限温度减。


7. 再次按下KEY0进入温度下限设定模式，按下KEY1下限温度增，按下KEY2下限温度减。

8. 再次按下KEY0进入温度显示。

附上我的完整proteus仿真图：



【实验结果】

附件源代码： My\_test.c

【实验体会】

在此次的期末专题设计中，我碰到了许多的困难，比如我原来是尝试温度控制加上红外遥控控制的，但是遇到了温度传感器读取温度时会产生中断，使

红外遥控无法使用，我查阅了许多资料，但还是没有解决这个问题，于是我就取消了红外遥控，但是现阶段我解决不了的事情，我觉得在以后不断学习中我肯定可以回过头来，解决现在没解决的问题。在设计中我不断回顾以前学过的知识，更加熟悉了单片机的各种端口的作用，在查阅了购买的板子的开发手册后，由于买的板子是集成性质的，发现许多我想要用到的接口都会冲突，于是我转而使用仿真软件，完成了这次设计，我能更加熟练的操作各种软件以及对单片机的认识有了更进一步的提高。在以后的日子中，我会更加努力学习，达到自己的目标。总的来说，这次期末专题的实践让我受益匪浅，也更增加了我对单片机的兴趣，以后想着往嵌入式方向更进一步！