安徽建筑大学
ANHUI JIANZHU UNIVERSITY

电子设计竞赛

参赛小组人员：年志豪　胡建

2022 年 6 月 17 日

摘要

随着时代的进步和发展，单片机技术已经普及到我们生活，工作，科研，各个领域，已经成为一种比较成熟的技术。本文介绍了 DS18B20 数字温度传感器，DS1302 时钟模块，LCD1602 以及矩阵键盘在主控 STC89C52 控制下的硬件连接及软件编程，并给出了软件流程图以及硬件原理图。

该系统由上位机和下位机两大部分组成。下位机中包含四种模式（学号显示模式-温度及时间显示模式-时间修改模式-数据回查模式），此四种模式分别对应 4 个交互界面（由显示终端 LCD1602 完成）。利用矩阵键盘中 S15 和 S16 来完成界面切换（S16 用于学号显示模式-温度及时间显示模式切换，S15 用于时间修改模式-数据回查模式切换）。配置定时器 0，50ms 产生一次中断，在中断服务函数中定义计时计数变量 Count_i，计数为 200 时（即 10s）通过串口向上位机发送一次温度时间数据。计数为 400 时（即 20s）记录一次时间与温度。在时间修改模式中，通过矩阵键盘 S1 S5 S2 S6 S3 S7 S4 S8 S9 S13 S10 S14 分别完成对年，月，日，时，分，秒的增减，实现时间修改。在数据回查模式中，通过矩阵键盘 S8 与 S12 完成对数据的翻页查询。此外按下 S11 将会把查询界面的数据通过串口发送给上位机，实现数据的回显功能。上位机部分使用了通用 PC。

该系统实现了温度测量，时间显示，时间修改，数据记录，数据回显等功能。

关键词：温度测量　STC89C52　DS18B20　DS1302　单总线系统

一、前言

了解到该系统的需求，为了将数据信息显示出来，采用了一块 LCD1602 液晶显示屏，并通过编写函数完成初始化，能够完成显示目的。其次考虑的是完成温度的测量，本系统采用 DS18B20 温度传感器采集温度数据，并配置实现单总线通信读取温度数据，并通过 LCD1602 将温度数据显示出来。计时模块采用 DS1302 实时时钟，该时钟内部带有备用电源，可以使时间数据掉电不丢失，并通过软件编写程序，实现了通过按键设置显示时间的功能。为了完成与上位机的通信，配置了 UART 串口函数，最终，通过串口可以实现向上位机发送历史温度以及时间数据，达到数据保存目的。至此该系统总体需求均已实现。

二、系统方案设计

方案一：

由于本设计是测温电路，可以使用热敏电阻之类的器件利用其感温效应，在将随被测温度变化的电压或电流采集过来，进行 A/D 转换后，就可以用单片机进行数据的处理，在显示电路上，就可以将被测温度显示出来，这种设计需要用到 A/D 转换电路，感温电路比较麻烦。

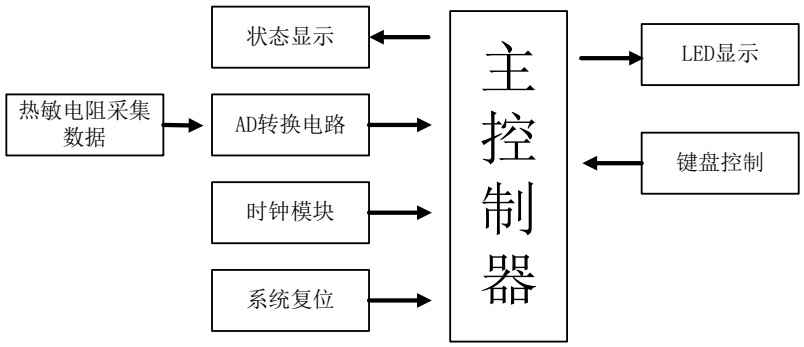


图 2.1 方案一流程图

方案二 ：

对于测温器件的重新选择，考虑到用温度传感器，在单片机电路设计中，大多都是使用传感器，所以可以采用一只温度传感器 DS18B20，此传感器，可以很容易直接读取被测温度值，进行转换，就可以满足设计要求。

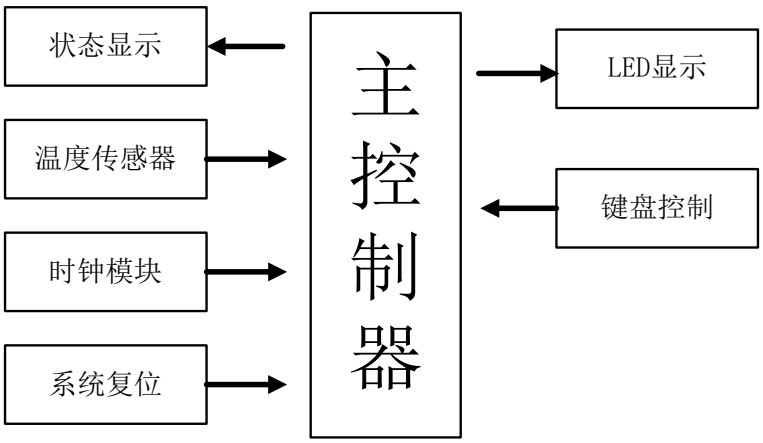


图 2.2 方案二方框图

通过上述两种方案的设计以及流程图，可以看出，方案一通过热敏电阻采集的温度数据需要额外搭载 AD 转换电路进行转化，方案二只需通过温度传感器即可采集准确的温度数据。采用热敏电阻，可满足 40 摄氏度至 90 摄氏度测量范围，但热敏电阻精度、重复性、可靠性较差，对于检测 1 摄氏度的信号是不适用的，DS18B20 测温范围为-55℃到+125℃，在-10℃到+85℃范围内误差为±0.4°，精确性上优于热敏电阻。同时软件设计方面方案二简易程度要优于方案一，因此通过比较，最终选择方案二作为本系统的设计方案。

三、理论分析与计算

3.1 温度数据

DS18B20 测温范围为-55℃到+125℃，在-10℃到+85℃范围内误差为±0.4°。返回 16 位二进制温度数值。

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
低字节	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}
	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
高字节	S	S	S	S	S	2^6	2^5	2^4

图 3.1 温度寄存器

由图 3.1 可知 16 位温度数值寄存器中，最低位表示 0.0625，即最终温度数据可以精确到四位小数。

3.2 定时器配置

定时器初始化时需要配值相关寄存器以及计数初值。本系统使用挂载 12MHz 时钟源的定时器 0，配置 50ms 的定时器，计算初值公式如下：

T=（65535-n）/12MHz （1）

其中 n 为定时器初值，T 为所需的定时周期。对于 50ms 定时周期，算得初值为 15536。

四、系统电路设计

4.1 LCD1602 液晶屏

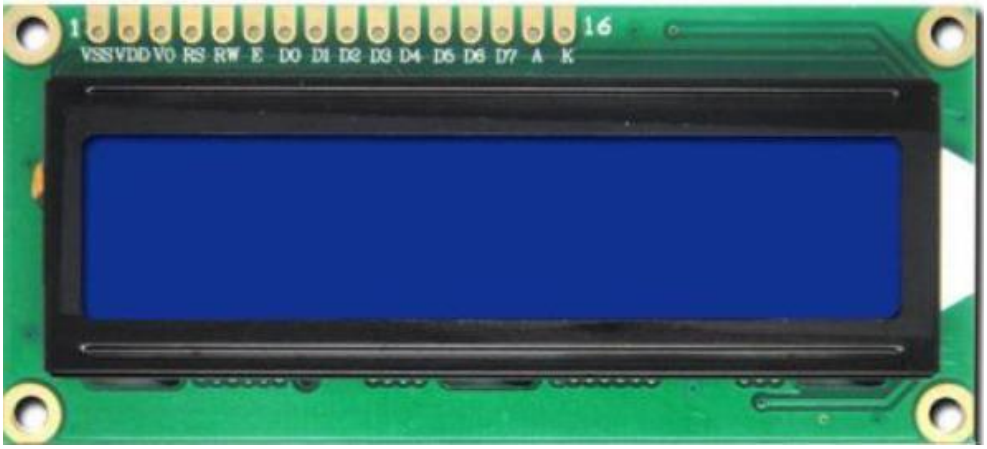


图 4.1 LCD1602 实物图

主要参数：

- 显示字符:16×2 个字符
- 工作电压:4.5~5V
- 工作电流:2.0mA
- 工作温度：-20° C~70° C
- 模块最佳工作电压:5.0V
- 单个字符尺寸 2.95×4.35（W×Hmm）
- 引脚：16 脚

作为显示端显示相关信息，如学号、温度、时间等。

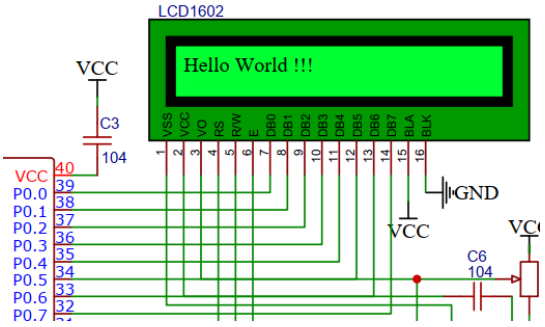


图 4.2 LCD1602 接线图

注：模块工作时序和控制，状态以及数据寄存器配置读写，见附件 1：**LCD1602 中文数据手册**

4.2 DS18B20 温度传感器

DS18B20 是一款常用的高精度的单总线数字温度测量芯片。具有体积小，硬件开销低，抗干扰能力强，精度高的特点。测温范围为-55℃到+125℃，在-10℃到+85℃范围内误差为±0.4° 。宽电压供电，电压 2.5V~5.5V。

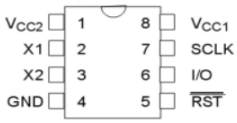
DS18B20 内部含有 EEPROM ，通过配置寄存器可以设定数字转换精度和报警温度，在系统掉电以后，它仍可保存分辨率及报警温度的设定值。

注：模块工作时序和控制，状态以及数据寄存器配置读写，见附件 2： **DS18B20 用户手册**

4.3 DS1302 实时时钟

DS1302 是由美国 DALLAS 公司推出的具有涓细电流充电能力的低功耗实时时钟芯片。它可以对年、月、日、周、时、分、秒进行计时，并且具有闰年补偿等多种功能。

DS1302 工作电压为 2.0V～5.5V。采用三线接口与 CPU 进行同步通信，并可采用突发方式一次传送多个字节的时钟信号或 RAM 数据。DS1302 内部有一个 31×8 的用于临时性存放数据的 RAM 寄存器。DS1302 增加了主电源/后备电源双电源引脚，同时提供了对后备电源进行涓细电流充电的能力。



引脚	说明
Vcc2	主电源
Vcc1	后备电源（断电后保证时钟正常运行）
x1,x2	外接32.768KHZ晶振
GND	接地
RST	复位引脚（低电平有效）
I/O	数据输入/输出引脚
SCLK	串行时钟输入引脚

图 4.3 DS1302 的引脚功能图

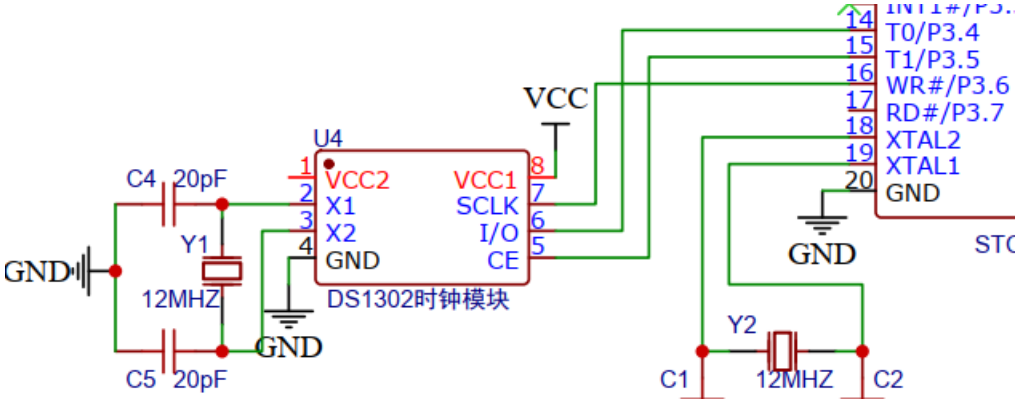


图 4.4 DS1302 实时时钟接线图

注：模块工作时序和控制，状态以及数据寄存器配置读写，见附件 3： **DS1302 数据手册**

4.4 矩阵键盘

通过矩阵键盘控制系统实现相应功能

扫描原理：

先从 P1 口的高四位（四个行）输出高电平，低四位（四个列）输出低电平，假设有按键按下，从 P1 口的高四位读取键盘状态。判断高四位的四行哪一行变成了低电平，就知道是第几行，再从 P1 口的低四位（四个列）输出高电平，高四位（四个行）输出低电平，从 P1 口的低四位读取键盘状态。判断低四位的四列哪一行变成了低电平，就知道是第几列，将两次读取结果组合起来就可以得到当前按键的特征编码。使用上述方法我们得到 16 个键的特征编码。

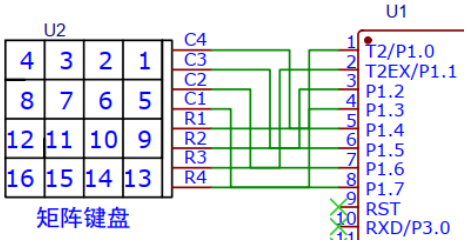


图 4.5 矩阵键盘 接线图

五、 系统软件设计

整个系统的功能是由硬件电路配合软件来实现的，当硬件基本定型后，软件的功能也就基本定下来了。从软件的功能不同可分为两大类：一是监控软件（主程序），它是整个控制系统的核心，专门用来协调各执行模块和操作者的关系。二是执行软件（子程序），它是用来完成各种实质性的功能如测量、计算、显示、通讯等。每一个执行软件也就是一个小的功能执行模块。

首先要根据系统的总体功能和键盘设置选择一种最合适的监控程序结构，然后根据实时性的要求，合理地安排监控软件和各执行模块之间地调度关系。

5.1 主程序方案

主程序调用了 5 个子程序，分别是 LCD 显示程序、键盘扫描以及按键处理程序、温度读取程序、中断控制程序、单片机与 PC 机串口通讯程序。

（1）键盘扫描电路及按键处理程序：实现键盘的输入按键的识别及相关处理。（2）温度读取程序：对温度芯片送过来的数据进行处理，进行判断和显示。

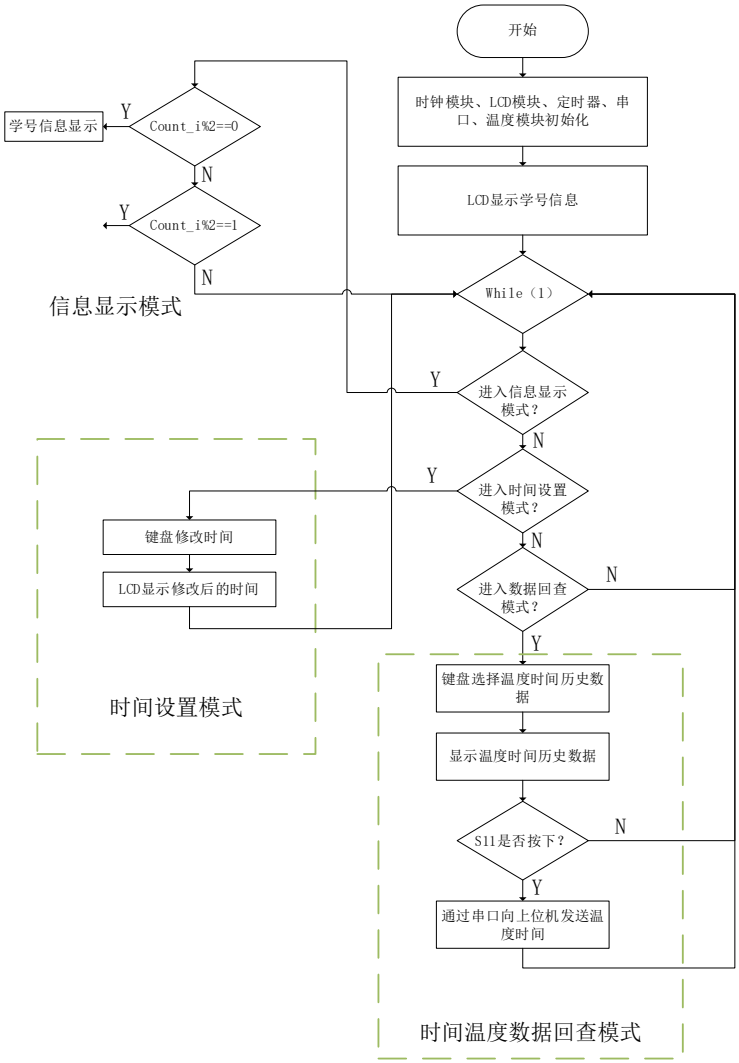
（3）LCD 显示程序：控制系统的显示部分，显示信息、温度数据以及时间等。

（4）中断控制程序：实现定时记录历史数据以及发送数据功能。

（5）串口通讯程序：实现 PC 机与单片机通讯，将温度数据传送给 PC 机。

各个功能程序以子程序的形式写好，当写主程序的时候，只需要调用子程序，然后在寄存器的分配上作一下调整，消除寄存器冲突和 I/O 冲突即可。

主程序流程图如下图:



六、 系统测试

6.1 分步调试

1、测试环境及工具

测试温度：0~100 摄氏度。（模拟不同温度值环境）

测试仪器及软件：数字万用表，温度计 0~100 摄氏度，串口调试助手。

测试方法：目测。

2、测试方法

使系统运行，观察系统硬件检测是否正常（包括单片机最小系统，键盘电路，显示电路，温度测试电路等）。系统自带测试表格数据，观察显示数据是否相符合即可。

采用温度传感器测试不同温度下的水温，目测显示电路是否正常。并记录各温度值，与实际温度值比较，得出系统的温度指标。

测试数据回查功能是否正常运行，检查回查后的数据是否正确。

使用串口调试助手与单片机通讯，观察单片机与串口之间传输数据正确否。

3、测试结果

记录次数	实时温度(^C)	实时时间(时：分：秒)	回查温度(^C)	回查时间(时：分：秒)
1	29.75	0：0：20	29.75	0：0：21
2	29.25	0：0：40	29.25	0：0：43
3	29.25	0：1：00	29.25	0：1：05
4	29.75	0：1：20	29.75	0：1：26
5	29.00	0：1：40	29.00	0：1：48
6	29.00	0：2：00	29.00	0：2：10
7	28.50	0：2：20	28.50	0：2：32
8	30.50	0：2：40	30.50	0：2：53
9	30.25	0：3：00	30.25	0：3：15
10	29.25	0：3：20	29.25	0：3：37

4、测试结果分析

自检正常，多组温度显示正常，回查数据正常，串口传输数据正确。

因为芯片是塑料封装，所以对温度的感应灵敏度不是相当高，需要一个很短的时间才能达到稳定。对于定时器，本系统设置了 50ms 定时器，定时次数最大为 400，即 20s 记录一次数据，当记录 200 个数时（即 10s），单片机向串口发送一次时间温度数据。但是实测来看平均记录周期略微有所误差，主要原因在于系统执行程序功能需要一定时间，改进的方法有简化系统程序代码等。

$$50ms * 200 = 10s \quad (2)$$

6.2 统一调试

将硬件及软件结合起来进行系统的统一调试。实现 PC 机与单片机通讯，两者可以实时更新显示温度值与时间。

七、结论

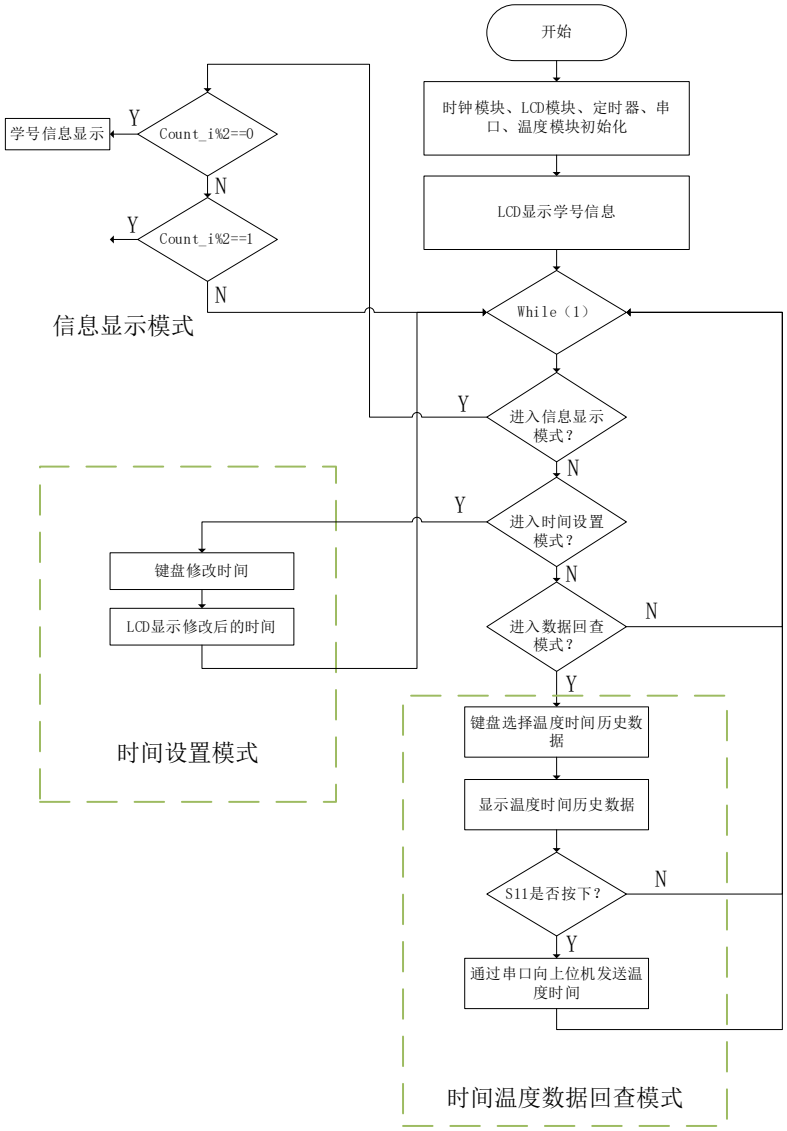
在此次项目中，熟悉了 DS18B20 温度传感器 •DS1302 实时时钟，矩阵键盘以及 LCD1602 模块的用法。提高了串口通信，SPI 通信，单总线系统的理解。对于主程序逻辑构思更加清晰，增加了一定的实战经验。

此项目设计基本完成了预期的目标，系统在硬件自动测试，键盘操作，实时显示方面做的比较好。但是由于能力有限，设计成果并不是很完美，还存在下面问题：液晶显示不够清晰，串口通讯不稳定，未对温度数值统计处理以及存储。如加入温度数据处理函数使得温度数据显示更加稳定准确；修改 LCD 屏显函数使其显示更加清除稳定等。由于经济实力有限，缺少语音播报模块，未完成对温度的播报要求。

附录

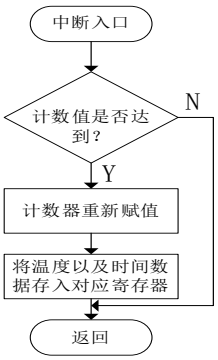
一、 程序流程图

(1）主程序流程

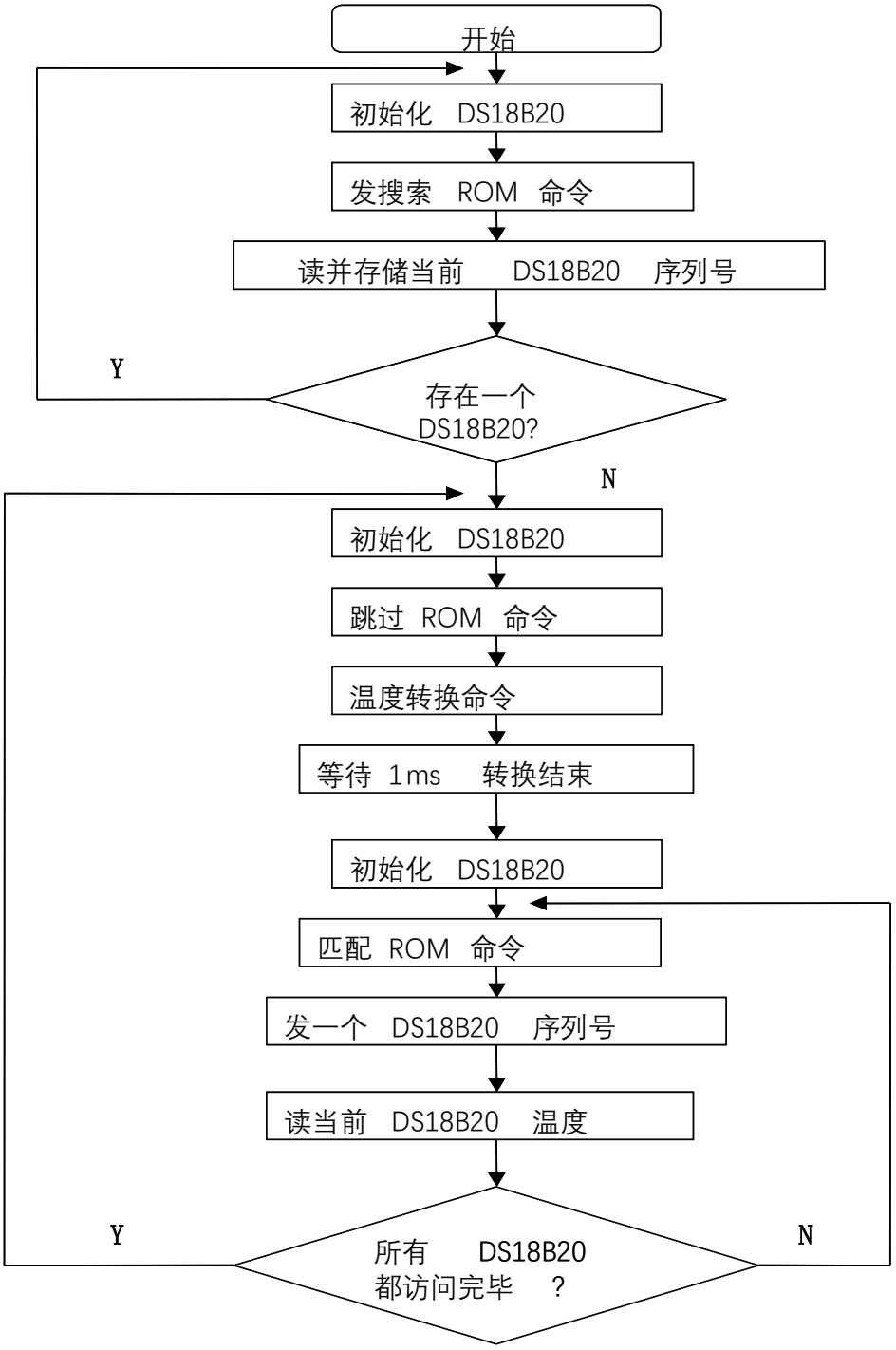


（2）子程序流程图

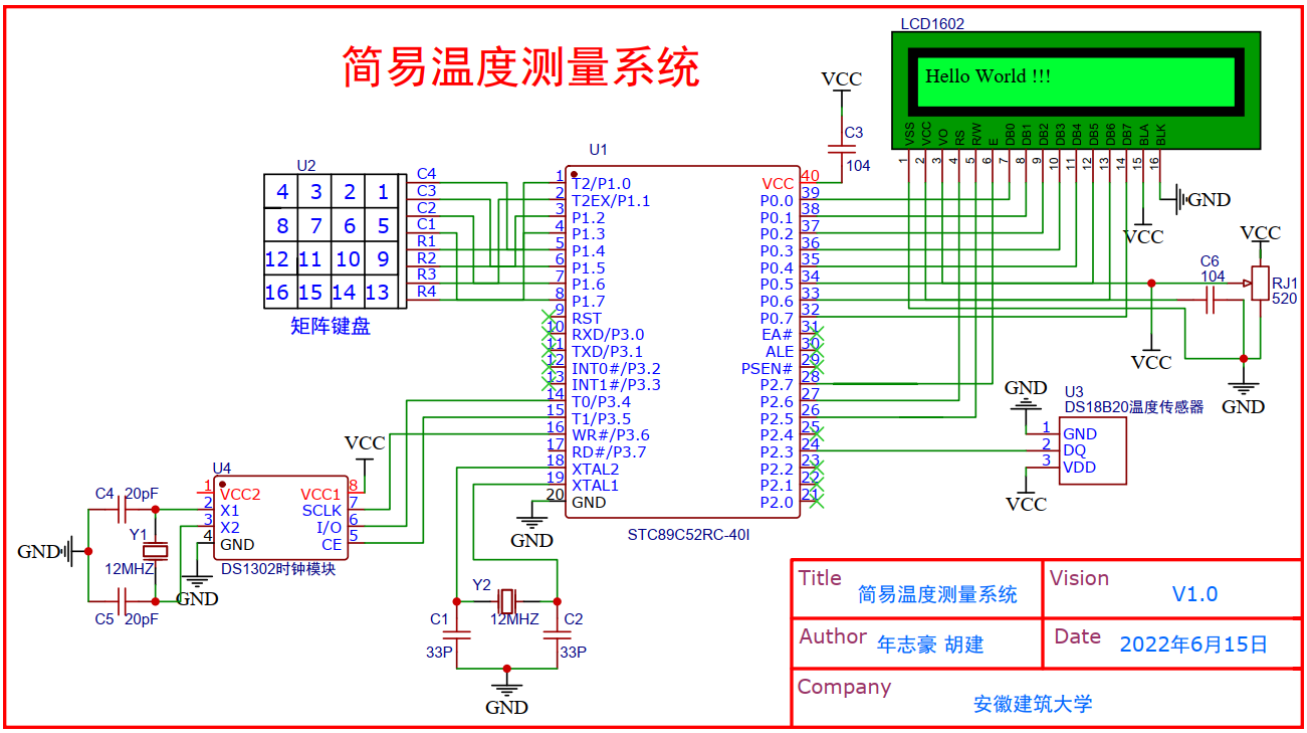
1．中断流程图


中断入口
计数值是否达到？
N
Y
计数器重新赋值
将温度以及时间数据存入对应寄存器
返回

2. 温度读取流程图


开始
初始化 DS18B20
发搜索 ROM 命令
读并存储当前 DS18B20 序列号
Y
存在一个 DS18B20?
N
初始化 DS18B20
跳过 ROM 命令
温度转换命令
等待 1ms 转换结束
初始化 DS18B20
匹配 ROM 命令
发一个 DS18B20 序列号
读当前 DS18B20 温度
Y
所有 DS18B20 都访问完毕 ？
N

二、 硬件原理图



三、 程序代码

Main.c （主函数）

```
/*****头文件区**********************/
#include <REGX52.H>
#include "LCD1602.h"
#include "DS18B20.h"
#include "Delay.h"
#include "MatrixKey.h"
#include "DS1302.h"
#include "Timer.h"
#include "UART.h"
/************************************************/
/*****变量定义区**************************************************/
u8 Count_k=0;
u8 Count_i=0;
u8 Count_j=0;
/*****************/
/******************
* Name           main
* Functiom      .......
* Input          Non
* Output        Non
* Author         年志豪 胡建
```

```
* University     安徽建筑大学
* Decoding       GB2312
* Time           2022 年 6 月 14 日
********************/
void main()
{
DS1302_Init();        //时钟模块初始化
LCD_Init();          //LCD1602 模块初始化
Timer0Init();          //定时器 0 初始化
UartInit();            //串口初始化
DS1302_SetTime();     //时钟时间赋初值
DS18B20_ConvertT();  //上电先转换一次温度，防止第一次读数据错误
LCD_ShowString(1,1,"N1:20210040216");//学号显示
LCD_ShowString(2,1,"N2:20203100142");
while(1)
{
matrixkey_return=MatrixKey();
if(matrixkey_return==15)
{
Count_k++;
}
if(Count_k%3==0)
{
if(matrixkey_return==16)
{
Count_i++;                          //模式控制计数器变量自增
matrixkey_return=0;                 //矩阵键值清零
}
if(Count_i%2==0)                      //模式 1(学号显示模式)
{
LCD_Init();                         //LCD1602 模块初始化
LCD_ShowString(1,1,"N1:20210040216");
LCD_ShowString(2,1,"N2:20203100142");
}
if(Count_i%2==1)                      //模式 2(温度时间显示模式)
{
LCD_Init();                        //LCD1602 模块初始化
DS18B20_ReadT();                   //读取温度
DS18B20_ConvertT();                //转换温度
LCD1602_Show_T();                  //显示温度
DS1302_ReadTime();                 //读取时间
LCD1602_Show_Time();               //显示时间
}
Delay(500);
```

```
}
if(Count_k%3==1)                    //模式3(时间设置模式)
{
matrixkey_return=0;
LCD_Init();
LCD_ShowString(1,1,"MODE:Setting Time");
LCD1602_Show_Time();
while(1)
{
matrixkey_return=MatrixKey();
if(matrixkey_return==1){DS1302_Time[0]++;matrixkey_return=0;}
if(matrixkey_return==5){DS1302_Time[0]--;matrixkey_return=0;}

if(matrixkey_return==2){DS1302_Time[1]++;matrixkey_return=0;}
if(matrixkey_return==6){DS1302_Time[1]--;matrixkey_return=0;}

if(matrixkey_return==3){DS1302_Time[2]++;matrixkey_return=0;}
if(matrixkey_return==7){DS1302_Time[2]--;matrixkey_return=0;}

if(matrixkey_return==4){DS1302_Time[3]++;matrixkey_return=0;}
if(matrixkey_return==8){DS1302_Time[3]--;matrixkey_return=0;}

if(matrixkey_return==9){DS1302_Time[4]++;matrixkey_return=0;}
if(matrixkey_return==13){DS1302_Time[4]--;matrixkey_return=0;}

if(matrixkey_return==10){DS1302_Time[5]++;matrixkey_return=0;}
if(matrixkey_return==14){DS1302_Time[5]--;matrixkey_return=0;}

DS1302_SetTime();
LCD1602_Show_Time();

if(matrixkey_return==15||matrixkey_return==16){break;}
```

```
98.}
99.Count_k++;
100.  }
101.  if(Count_k%3==2)                              //模式4(时间温度数
   据回查模式)
102.  {
103.  matrixkey_return=0;
104.  LCD_Init();
105.  LCD_ShowString(1,1,"MODE:");
106.  LCD_ShowString(1,8,"Record");
107.  LCD_ShowNum(1,6,(Count_j%10)+1,2);
108.  LCD_ShowNum(2,1,Time_Record_Array[Count_j%10][0],2);
109.  LCD_ShowString(2,3,":");
110.  LCD_ShowNum(2,4,Time_Record_Array[Count_j%10][1],2);
111.  LCD_ShowString(2,6,":");
112.  LCD_ShowNum(2,7,Time_Record_Array[Count_j%10][2],2);
113.  while(1)
114.  {
115.  matrixkey_return=MatrixKey();
116.  if(matrixkey_return==8){Count_j++;matrixkey_return=0;}
117.  if(matrixkey_return==12){Count_j--;matrixkey_return=0;}
118.  LCD_ShowNum(1,6,(Count_j%10)+1,2);
119.  LCD_ShowNum(2,1,Time_Record_Array[Count_j%10][0],2);
120.  LCD_ShowString(2,3,":");
121.  LCD_ShowNum(2,4,Time_Record_Array[Count_j%10][1],2);
122.  LCD_ShowString(2,6,":");
123.  LCD_ShowNum(2,7,Time_Record_Array[Count_j%10][2],2);
124.
125.
126.  LCD_ShowChar(2,11,'+'); //显示正号
127.  LCD_ShowNum(2,12,Temperature_Record_Array[Count_j%10],2);
     //显示温度整数部分
128.  LCD_ShowChar(2,14,'.');  //显示小数点
129.  LCD_ShowNum(2,15,(unsigned long)(Temperature_Record_Array
   [Count_j%10]*10000)%10000,2);//显示温度小数部分
130.  if(matrixkey_return==11)
131.  {
132.  printf("\n\n 温度时间回显 S11 触发\n");
133.  printf("第%2.0f 次时间温度记录\n",(float)(Count_j%10)+1);
134.  printf("温
   度: %2.4f^C  ",Temperature_Record_Array[Count_j%10]);
135.  printf("时
   间: %2.0f:%2.0f:%2.0f\n\n\n",(float)Time_Record_Array[Count_
```

```
   j%10][0],(float)Time_Record_Array[Count_j%10][1],(float)Tim
   e_Record_Array[Count_j%10][2]);
136.  }
137.  if(matrixkey_return==15){matrixkey_return=0;Count_k++;bre
   ak;}
138.  }
139.  }
140.  }
141.  }
```

DS18B20.c （温度传感器）

```
1. #include <REGX52.H>
2. #include "OneWire.h"
3. #include "LCD1602.h"
4. #include "DS18B20.h"
5. //DS18B20 指令
6. #define DS18B20_SKIP_ROM   0xCC
7. #define DS18B20_CONVERT_T   0x44
8. #define DS18B20_READ_SCRATCHPAD  0xBE
9. float T;
10./**
11.   * @brief  DS18B20 开始温度变换
12.   * @param  无
13.   * @retval 无
14.   */
15.void DS18B20_ConvertT(void)
16.{
17. OneWire_Init();
18. OneWire_SendByte(DS18B20_SKIP_ROM);
19. OneWire_SendByte(DS18B20_CONVERT_T);
20.}
21.
22./**
23.   * @brief  DS18B20 读取温度
24.   * @param  无
25.   * @retval 温度数值
26.   */
27.float DS18B20_ReadT(void)
28.{
29. unsigned char TLSB,TMSB;
30. int Temp;
31. OneWire_Init();
32. OneWire_SendByte(DS18B20_SKIP_ROM);
```

```
33.  OneWire_SendByte(DS18B20_READ_SCRATCHPAD);
34.  TLSB=OneWire_ReceiveByte();
35.  TMSB=OneWire_ReceiveByte();
36.  Temp=(TMSB<<8)|TLSB;
37.  T=Temp/16.0;
38.  return T;
39.}
40.
41.void LCD1602_Show_T(void)
42.{
43.  LCD_ShowString(1,1,"T:");
44.  if(T<0)    //如果温度小于0
45.  {
46.   LCD_ShowChar(1,3,'-'); //显示负号
47.   T=-T;   //将温度变为正数
48.  }
49.  else    //如果温度大于等于0
50.  {
51.   LCD_ShowChar(1,3,'+'); //显示正号
52.  }
53.  LCD_ShowNum(1,4,T,3);  //显示温度整数部分
54.  LCD_ShowChar(1,7,'.');  //显示小数点
55.  LCD_ShowNum(1,8,(unsigned long)(T*10000)%10000,4);//显示温
     度小数部分
56.  LCD_ShowString(1,12,"^C");
57.}
```

DS1302.c （实时时钟）

```
1.  #include <REGX52.H>
2.  #include "DS1302.h"
3.  #include "LCD1602.h"
4.  //引脚定义
5.  sbit DS1302_SCLK=P3^6;
6.  sbit DS1302_IO=P3^4;
7.  sbit DS1302_CE=P3^5;
8.
9.  //寄存器写入地址/指令定义
10.#define DS1302_SECOND  0x80
11.#define DS1302_MINUTE  0x82
12.#define DS1302_HOUR   0x84
13.#define DS1302_DATE   0x86
14.#define DS1302_MONTH  0x88
```

```
#define DS1302_DAY   0x8A
#define DS1302_YEAR   0x8C
#define DS1302_WP     0x8E

//时间数组，索引0~6分别为年、月、日、时、分、秒、星期
unsigned char DS1302_Time[]={22,06,01,0,0,0,6};
/**
  * @brief  DS1302初始化
  * @param  无
  * @retval 无
  */
void DS1302_Init(void)
{
 DS1302_CE=0;
 DS1302_SCLK=0;
}

/**
  * @brief  DS1302写一个字节
  * @param  Command 命令字/地址
  * @param  Data 要写入的数据
  * @retval 无
  */
void DS1302_WriteByte(unsigned char Command,Data)
{
 unsigned char i;
 DS1302_CE=1;
 for(i=0;i<8;i++)
 {
  DS1302_IO=Command&(0x01<<i);
  DS1302_SCLK=1;
  DS1302_SCLK=0;
 }
 for(i=0;i<8;i++)
 {
  DS1302_IO=Data&(0x01<<i);
  DS1302_SCLK=1;
  DS1302_SCLK=0;
 }
 DS1302_CE=0;
}

/**
  * @brief  DS1302读一个字节
```

```
59.   * @param  Command 命令字/地址
60.   * @retval 读出的数据
61.   */
62.unsigned char DS1302_ReadByte(unsigned char Command)
63.{
64. unsigned char i,Data=0x00;
65. Command|=0x01; //将指令转换为读指令
66. DS1302_CE=1;
67. for(i=0;i<8;i++)
68. {
69.  DS1302_IO=Command&(0x01<<i);
70.  DS1302_SCLK=0;
71.  DS1302_SCLK=1;
72. }
73. for(i=0;i<8;i++)
74. {
75.  DS1302_SCLK=1;
76.  DS1302_SCLK=0;
77.  if(DS1302_IO){Data|=(0x01<<i);}
78. }
79. DS1302_CE=0;
80. DS1302_IO=0; //读取后将 IO 设置为 0，否则读出的数据会出错
81. return Data;
82.}
83.
84./**
85.   * @brief  DS1302 设置时间，调用之后，DS1302_Time 数组的数字会被设置到 DS1302 中
86.   * @param  无
87.   * @retval 无
88.   */
89.void DS1302_SetTime(void)
90.{
91. DS1302_WriteByte(DS1302_WP,0x00);
92. DS1302_WriteByte(DS1302_YEAR,DS1302_Time[0]/10*16+DS1302_Time[0]%10);//十进制转 BCD 码后写入
93. DS1302_WriteByte(DS1302_MONTH,DS1302_Time[1]/10*16+DS1302_Time[1]%10);
94. DS1302_WriteByte(DS1302_DATE,DS1302_Time[2]/10*16+DS1302_Time[2]%10);
95. DS1302_WriteByte(DS1302_HOUR,DS1302_Time[3]/10*16+DS1302_Time[3]%10);
96. DS1302_WriteByte(DS1302_MINUTE,DS1302_Time[4]/10*16+DS1302_Time[4]%10);
```

```
97.  DS1302_WriteByte(DS1302_SECOND,DS1302_Time[5]/10*16+DS1302
   _Time[5]%10);
98.  DS1302_WriteByte(DS1302_DAY,DS1302_Time[6]/10*16+DS1302_Ti
   me[6]%10);
99.  DS1302_WriteByte(DS1302_WP,0x80);
100.    }
101.
102.    /**
103.      * @brief  DS1302 读取时间，调用之后，DS1302 中的数据会被
   读取到 DS1302_Time 数组中
104.      * @param  无
105.      * @retval 无
106.      */
107.    void DS1302_ReadTime(void)
108.    {
109.     unsigned char Temp;
110.     Temp=DS1302_ReadByte(DS1302_YEAR);
111.     DS1302_Time[0]=Temp/16*10+Temp%16;//BCD 码转十进制后读取
112.     Temp=DS1302_ReadByte(DS1302_MONTH);
113.     DS1302_Time[1]=Temp/16*10+Temp%16;
114.     Temp=DS1302_ReadByte(DS1302_DATE);
115.     DS1302_Time[2]=Temp/16*10+Temp%16;
116.     Temp=DS1302_ReadByte(DS1302_HOUR);
117.     DS1302_Time[3]=Temp/16*10+Temp%16;
118.     Temp=DS1302_ReadByte(DS1302_MINUTE);
119.     DS1302_Time[4]=Temp/16*10+Temp%16;
120.     Temp=DS1302_ReadByte(DS1302_SECOND);
121.     DS1302_Time[5]=Temp/16*10+Temp%16;
122.     Temp=DS1302_ReadByte(DS1302_DAY);
123.     DS1302_Time[6]=Temp/16*10+Temp%16;
124.    }
125.    void LCD1602_Show_Time(void)
126.    {
127.     LCD_ShowNum(2,1,DS1302_Time[0],2);
128.     LCD_ShowString(2,3,",");
129.     LCD_ShowNum(2,4,DS1302_Time[1],2);
130.     LCD_ShowString(2,6,",");
131.     LCD_ShowNum(2,7,DS1302_Time[2],2);
132.     LCD_ShowNum(2,9,DS1302_Time[3],2);
133.     LCD_ShowString(2,11,":");
134.     LCD_ShowNum(2,12,DS1302_Time[4],2);
135.     LCD_ShowString(2,14,":");
136.     LCD_ShowNum(2,15,DS1302_Time[5],2);
137.    }
```