

单片机应用**课程设计任务书**

**2018－2019学年 第 1 学期~第 16周**

|  |  |  |  |
| --- | --- | --- | --- |
| **题目1** | 基于 STC8A8K64S4A12单片机的带整点报时及双显示的电子钟 | | |
| 内容及要求：设计基于 STC8A8K64S4A12单片机的带整点报时及双显示的电子钟，购买元件、焊接装置、设计程序；该电子钟具有如下功能：要求规定的共阴数码管和LCD双重显示，先设计普通电子钟，完成后再修改设计，使用24小时制，7个数码位显示**h h. m m.s s.s ，**RT1602显示在第2行的开始处**，**其中要显示的秒含小数点后1位。开机时间规定11时59分00.0秒，用蜂鸣器间歇发声来模拟报时，间歇周期2秒，响1秒、歇1秒，共计响5声；撰写设计报告；回答老师提的问题。 | | | |
| **进度安排**   1. 方案论证 1.0天 2. 分析、设计、调试、运行 3天 3. 检查、整理、写设计报告、小结 1.0天 | | | |
| 学生姓名：冯烽 李玉林 袁世强 | | | |
| 指导时间2018年12月17日至2018年12月21日 | | | 指导地点：F401 |
| 任务下达 | 2018年 12月14日 | 任务完成 | 2018年12 月21日 |
| 考核方式 | 1.评阅 🗹 2.答辩 □ 3. 实际操作 🗹 4.其它🗹 | | |
| 指导教师 | 赵文龙 | 系（部）主任 | 刘 君 |

注：1、此表一组一表二份，课程设计小组组长一份；任课教师授课时自带一份备查。

1. 课程设计结束后与“课程设计小结”、“学生成绩单”一并交院教务存档。

**组内分工：**

**冯烽：软件设计及调试**

**袁世强：硬件焊接**

**李玉林：文稿的撰写**

**摘要**

设计基于 STC8A8K64S4A12单片机的带整点报时及双显示的电子钟，购买元件、焊接装置、设计程序；该电子钟具有如下功能：要求规定的共阴数码管和LCD双重显示，先设计普通电子钟，完成后再修改设计，使用24小时制，7个数码位显示**h h. m m.s s.s ，**RT1602显示在第2行的开始处**，**其中要显示的秒含小数点后1位。开机时间规定11时59分00.0秒，用蜂鸣器间歇发声来模拟报时，间歇周期2秒，响1秒、歇1秒，共计响5声。

关键词：单片机、电子钟、共阴数码管、LCD。

**目录**

**1.设计方案和工作原理……………………..…………………1**

**2.硬件设计……………………………..………………………2**

**2.1 系统原理图……………………..………………………2**

**2.2 芯片STC8A8K64S4A12工作电路…………………...3**

**3.软件设计调试………………..………………………………4**

**3.1 数码管扫描显示……………………………..…………4**

**3.2 RT1602液晶显示………………………………..……...5**

**4.调试……………………………………….……………….....7**

**4.1 数码管扫描显示……………………………..…………7**

**4.2 RT1602液晶显示……………………………….………8**

**5.结论……………………………………….………………….9**

**6.参考文献…………………………..……….……………….10**

**附录A 焊接实物图片………………………………..………11**

**附录B 实验程序………………………………..……………12**

# 1. 设计方案和工作原理

设计方案：

电子钟亦称数显钟(数字显示钟)，是一种用数字电路技术实现时、分、秒计时的装置，与机械时钟相比，直观性为其主要显著特点，且因非机械驱动，具有更长的使用寿命，相较石英钟的石英机芯驱动，更具准确性。电子钟已成为人们日常生活中必不可少的必需品，给人们的生活、学习、工作、娱乐带来极大方便。

电子钟的时钟功能:具有可选的24h(小时)或12h(小时)的计时方式，显示时、分、秒;具有快速校准时、分、秒的功能;能设置起闹时刻、响闹时间为1min(分钟)，超过1min(分钟)自动停止;具有人工止闹功能，止闹后不再重新操作，将不再发生起闹;具有整点报时功能。随着科技的进步与发达，部分电子钟还带投影功能，同时衍生为许多其他产品的辅助功能。

工作原理：

电子钟是一个将" 时"，"分"，"秒"显示于人的视觉器官的计时装置。它的计时周期为24小时，显示满刻度为23时59分59秒，具有校时功能和报时功能。秒信号产生器是整个系统的时基信号，它直接决定计时系统的精度，一般用石英晶体振荡器加分频器来实现。将标准秒信号送入"秒计数器"，"秒计数器"采用60进制计数器，每累计60秒发出一个"分脉冲"信号，该信号将作为"分计数器"的时钟脉冲。"分计数器"也采用60进制计数器，每累计60分钟，发出一个"时脉冲"信号，该信号将被送到"时计数器"。"时计数器"采用24进制计时器，可实现对一天24小时的累计。将"时"、"分"、"秒"计数器的输出状态用七段显示译码器译码，通过七段显示器显示出来。整点报时电路时根据计时系统的输出状态产生一脉冲信号，然后去触发一音频发生器实现报时。校时电路时用来对"时"、"分"、"秒"显示数字进行校对调整。

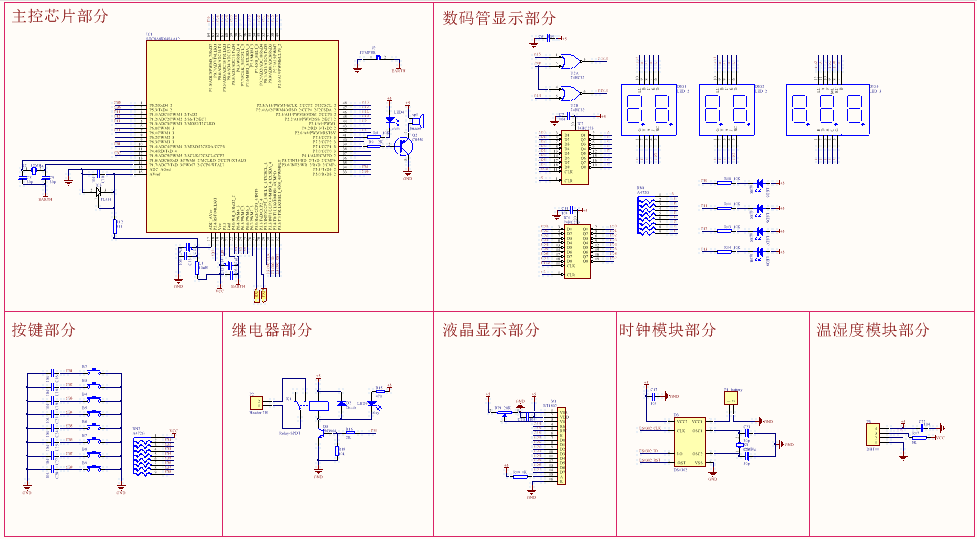
应用 ：

目前，在国内，电子钟因LCD数字显示效用直接有效，所以大多运用在城市的主要营业场所，以及车站、码头等公共场所。在对公共场所的电子钟设定的时候，使用者还可根据周边的气候、温度等对LCD屏进行设置。同时，因为LCD的显示耗电量很省，所以能够保持持续的工作效果。

# 2.硬件设计

## 2.1系统原理图

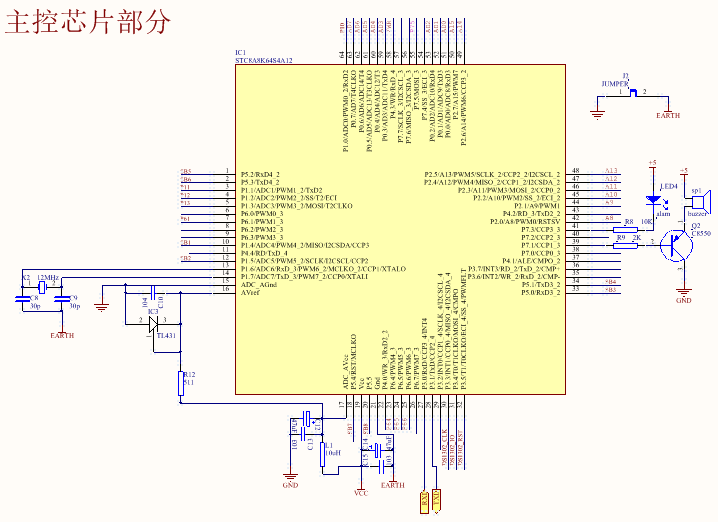
系统板的原理图如图2-1所示，其中由STC8A8K64S4A12芯片作为控制器；一个LED及蜂鸣器用作报警；两个两位一体数码管和一个三位一体数码管以及四个LED用作显示时间；RT1602液晶工作电路显示日期，温湿度信息；8个按键开关及滤波电路用于设置时间、选择模式；程序下载电路用于给芯片程序烧写；继电器控制电路用于控制外围装置；DHT11温湿度模块工作电路用以实时采集环境的温湿度数据；DS1302实时时钟工作电路在系统断电情况下采用备用电池供电，保证时钟的连续运行。



### 图2-1 系统板原理图

## 2.2 芯片STC8A8K64S4A12工作电路

芯片STC8A8K64S4A12引脚资源使用情况如图2-2所示，P1.0、P1.1、P1.2、P1.3分别连接数码管之间的LED5、LED6、LED7、LED8四个LED；P2.6、P2.7、P4.8连接74HC32的1、2、4脚，用以选择锁存器；P1.4、P1.5、P5.0、P5.1、P5.2、P5.3、P5.4、P5.5连接按键SB2、SB3、SB4、SB5、SB6、SB7、SB8、SB9八个按键，用以设置时间；P0口连接两个锁存器74HC273的输入引脚，配合74HC32来作为数码管的段码及位码控制；P7.3连接LED4；P7.1连接蜂鸣器；DS1302实时时钟模块接口分别为：P3.3对应SCLK，P3.4对应DSIO，P3.5对应RST；RT1602液晶模块接口分别为：P6.4对应RS，P6.5对应RW，P6.6对应EN，P0口对应数据IO；温湿度模块DHT11温湿度模块接口为：P7.5对应数据IO;

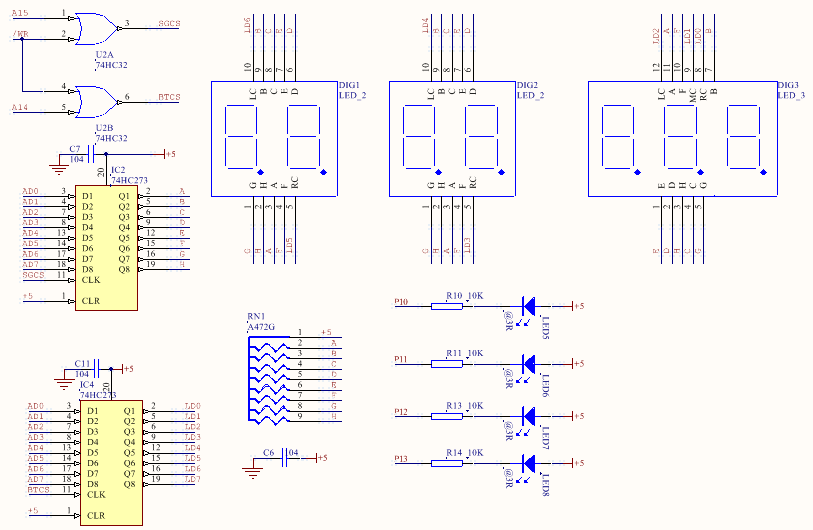


### 图2-2 STC8A8K64S4A12工作电路

# 3．软件设计

## 3.1 数码管扫描显示

从图3-1可知，扫描显示有挂载在总线上的锁存器端口，分别是锁存器IC2和IC4，它们均为8位输出端口。地址译码采用线选法译码，显示段码端口地址由写信号/WR和地址线A15，显示位码端口地址由写信号/WR和地址线A14，其余地址线无效，我们假设为1。故显示段码端口地址为0111,1111,1111,1111B，即0x7FFF，进行写操作，即可输出显示段码，锁存于IC2这个8位锁存器中。显示位码端口地址为1011,1111,1111,1111B，即0xBFFF，进行写操作，即可输出显示段码，锁存于IC4这个8位锁存器中。



### 图3-1 数码管扫描显示电路

采用8段共阴数码管，则扫描显示需要锁存8位段码，选用74HC273芯片能够实现，其中段码的LSB位对应段A，段码的MSB位对应段H，则小数点不亮时数符0~9的段码如表3-1所示。

### 表3-1 小数点不亮时共阴数码管段码表

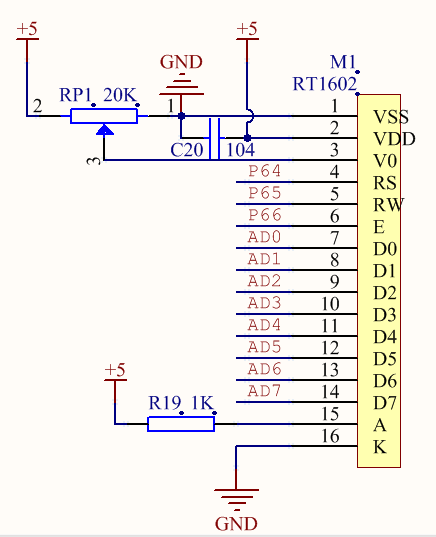
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 数符 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 段码 | 0x3f | 0x06 | 0x5b | 0x4f | 0x66 | 0x6d | 0x7d | 0x07 | 0x7f | 0x6f |

涉及到符号时，+号段码为0x00；-号段码为0x40；小数点单独控制，需要小数点的位，对应段码获得后，再或上0x80即可。

## 3.2 RT1602液晶显示

由图3-2可知，RP1是调节显示的对比度，一般上电要隐约看见1行16片点阵型黑影表示调节正常，如果一点都看不到点阵型黑影表示对比度调节不合适，有可能显示程序正确但看不见显示信息，所以硬件调试时必须先把对比度调好。

寄存器选择引脚RS连P6.4，读写操作选择RW连P6.5,使能信号连P6.6，数据IO连接P0口。



### 图3-2 LCD1602液晶显示电路原理图

配置不同的寄存器组合，即可对1602 实现不同的操作，具体如表3-2所示

表3-2 寄存器选择功能

|  |  |  |
| --- | --- | --- |
| RS | R/W | 操 作 |
| 0 | 0 | 指令寄存器(IR)写入 |
| 0 | 1 | 忙标志和地址计数器读出 |
| 1 | 0 | 数据寄存器(DR)写入 |
| 1 | 1 | 数据寄存器读出 |

**备注：**忙标志为"1"时，表明正在进行内部操作，此时不能输入指令或数据，要等内部操作结束，忙标志为"0"时。

要想1602正确工作，还得满足相应的时序条件，时序图如下：

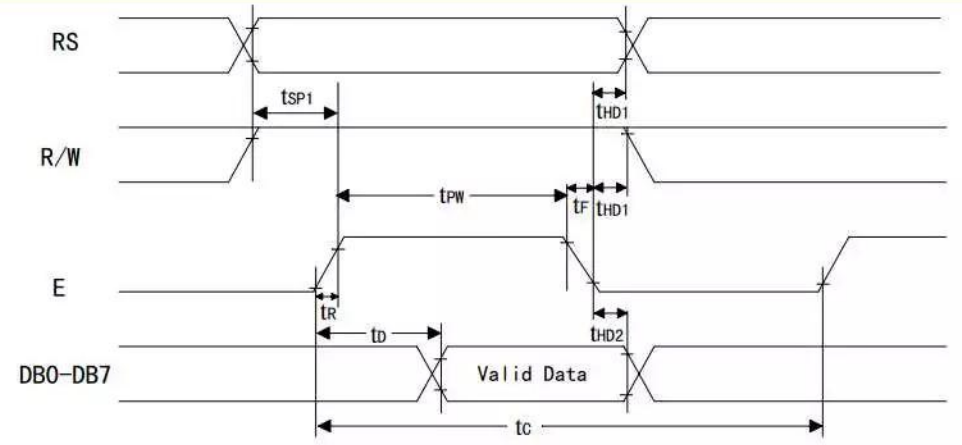
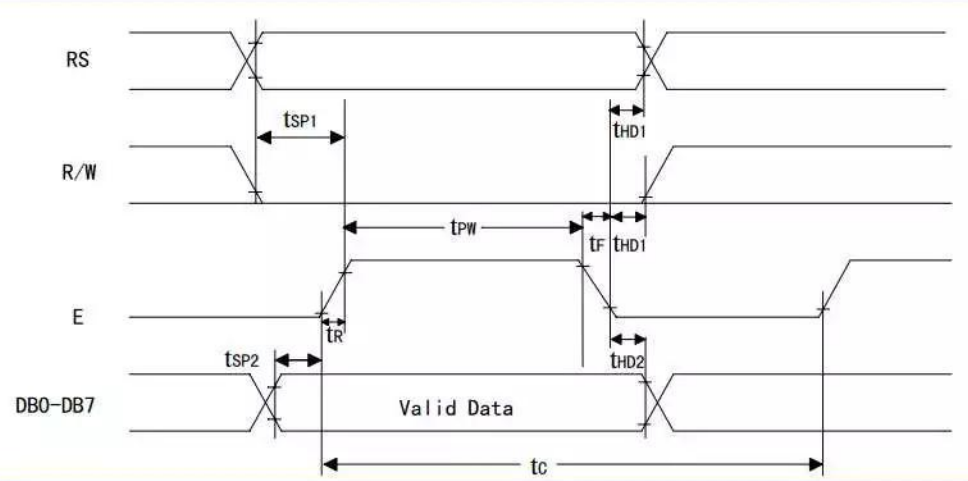


图3-3 1602读操作时序 

### 图3-4 1602写操作时序

### **4 调试**

## 4.1数码管扫描显示调试

为了显示时分秒及毫秒，共计需要7个数码管，则扫描显示需要1个8位锁存器锁存7位数码公共端信息，选用SN74ABT273芯片能够实现，其中位码的LSB位对8号数码管，段码的MSB位对1号数码管，编号从1到7对应功能和位码如表2所示。

### 表4-1 编号从1到7数码管对应功能和位码

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数码管编号 | 显示内容 | 二进制位码 | 十六进制 | M |
| 1 | 时（十位） | 01111111B | 0x7f | 7 |
| 2 | 时（个位） | 10111111B | 0xbf | 6 |
| 3 | 分（十位） | 11011111B | 0xdf | 5 |
| 4 | 分（个位） | 11101111B | 0xef | 4 |
| 5 | 秒（十位） | 11110111B | 0xf7 | 3 |
| 6 | 秒（十位） | 11111011B | 0xfb | 2 |
| 7 | 毫秒 | 11111101B | 0xfd | 1 |

程序中可以用如下方法定义端口：

#define Displaydata XBYTE [0x7FFF] //数码管段码锁存器端口地址

#define Displaybit XBYTE [0xBFFF]//数码管位码锁存器端口地址

可以考虑用查表的方法实现数字到段码的查表法以及扫描位控到位码，具体表格定义如下：

uchar code Disptab[10]={0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f}；

//段码表

uchar code Dispbit[8]={0xfe,0xfd,0xfb,0xf7,0xef,0xdf,0xbf,0x7f}；

//共阴位码表

扫描显示中经常需要考虑每位要显示一定的时间，经常使用软件延时的方法，但这种方法不好，我们采用定时器定时的方法，在中断服务程序中执行显示操作，具体方法如下。

Displaydata=dispbuf[M]； //查段码表送数显的段端口

Displaybit=Dispbit[M]； //查位码表送数显的位端口

M++； //修改下一次的显示位

## 4.2. RT1602液晶显示调试

相应程序：

sbit RS=P6^4; // RS位定义

sbit RW=P6^5; // RW位定义

sbit EN=P6^6; // EN位定义

/\*\*\*\*\*\*\*\*指令延时\*\*\*\*\*\*\*\*/

void delayms(unsigned char t)

{

unsigned char i,j;

for(i=0;i<10;i++)

for(j=0;j<t;j++);

}

/\*\*\*\*\*\*\*\*LCD忙检测\*\*\*\*\*\*\*\*\*\*/

unsigned char lazy\_detector( )

{

unsigned char busy\_flag;

RS=0;

RW=1;

EN=1;

busy\_flag=(P0 & 0x80)>>8;

EN=0;

return busy\_flag;

}

/\*\*\*\*\*\*\*写指令\*\*\*\*\*\*/

void command\_write(unsigned char data1)

{

while(lazy\_detector());

RS=0;

RW=0;

EN=0;

delayms(1);

P0=data1;

delayms(3);

EN=1;

delayms(1);

EN=0;

delayms(1);

}

/\*\*\*\*\*写数据\*\*\*\*\*\*\*\*/

void data\_write(unsigned char data2)

{

while(lazy\_detector());

RS=1;

RW=0;

EN=0;

delayms(1);

P0=data2;

delayms(3);

EN=1;

delayms(1);

EN=0;

}

/\*\*\*\*\*\*\*\*1602初始化\*\*\*\*\*\*\*\*/

void init\_1602()

{

delayms(15);

command\_write(0x38);

delayms(1);

command\_write(0x38);

delayms(1);

command\_write(0x0c);

delayms(1);

command\_write(0x0c);

delayms(1);

command\_write(0x06);

delayms(1);

command\_write(0x06);

delayms(1);

command\_write(0x01);

delayms(1);

command\_write(0x01);

delayms(1);

}

//1602液晶显示初始化

void Display\_init()

{

unsigned char i=0;

delayms(15);

command\_write(0x80);

for(i=0;i<16;i++)

{

data\_write(' ');

}

command\_write(0x80);

for(i=0;i<16;i++)

{

data\_write(' ');

}

command\_write(0xc0);

for(i=0;i<16;i++)

{

data\_write(' ');

}

}

# 5 结论

关于这次课程设计，我们花费了比较多的心思，既是对课程理论内容的一次复习和巩固，还让我们丰富了更多与该专业相关的其他知识。在真正设计之前我们做了相当丰富的准备，首先巩固了一下课程理论，再一遍熟悉课程知识的构架，然后结合加以理论分析、总结、有了一个清晰的思路和一个完整的软件流程之后才着手设计。在设计程序时，我们不能妄想一次就将整个程序设计好，不断改进是程序设计的必经之路，养成注释程序的好习惯是非常必要的。我觉得在课程设计过程中遇到问题很正常，但我们应该将每次遇到的问题记录下来，并分析清楚，以免下次再碰到同样的问题的课程设计又错了。

除了对此次设计的准备工作之外，我们还学到了很多平时难得的东西。首先是团队协作，在这次设计当中，难免产生观点和意见的分歧，以及分工明细、时间安排不合理等问题，通过这次设计，我们体会到了团队合作的重要性及其力量之强大，还有让我们处理事情更加有条理，思路更加清晰明了，发现、提出、分析、解决问题和实践能力的提高将受益于我们在以后的学习、工作和生活中。

参考文献

### [1]李朝清.单片机原理与接口技术[M].北京航空航天大学出版社。2005.10.

### [2]耿永刚.单片机与接口应用技术[M].华东师范大学出版社，2008.4.

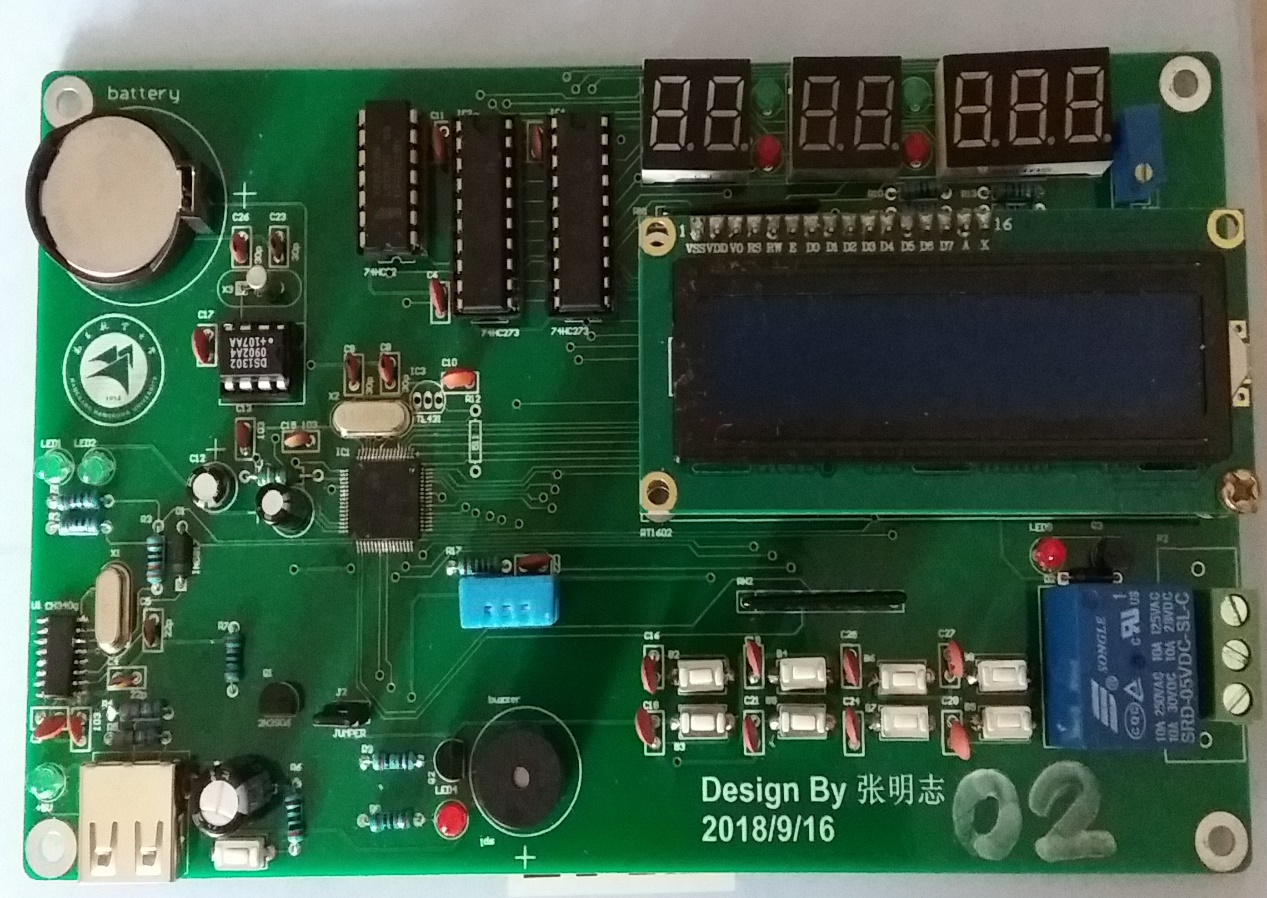
### [3]范立南， 谢了殿，单片机原理及应用教程[M].北京人学出版社，2006.1.

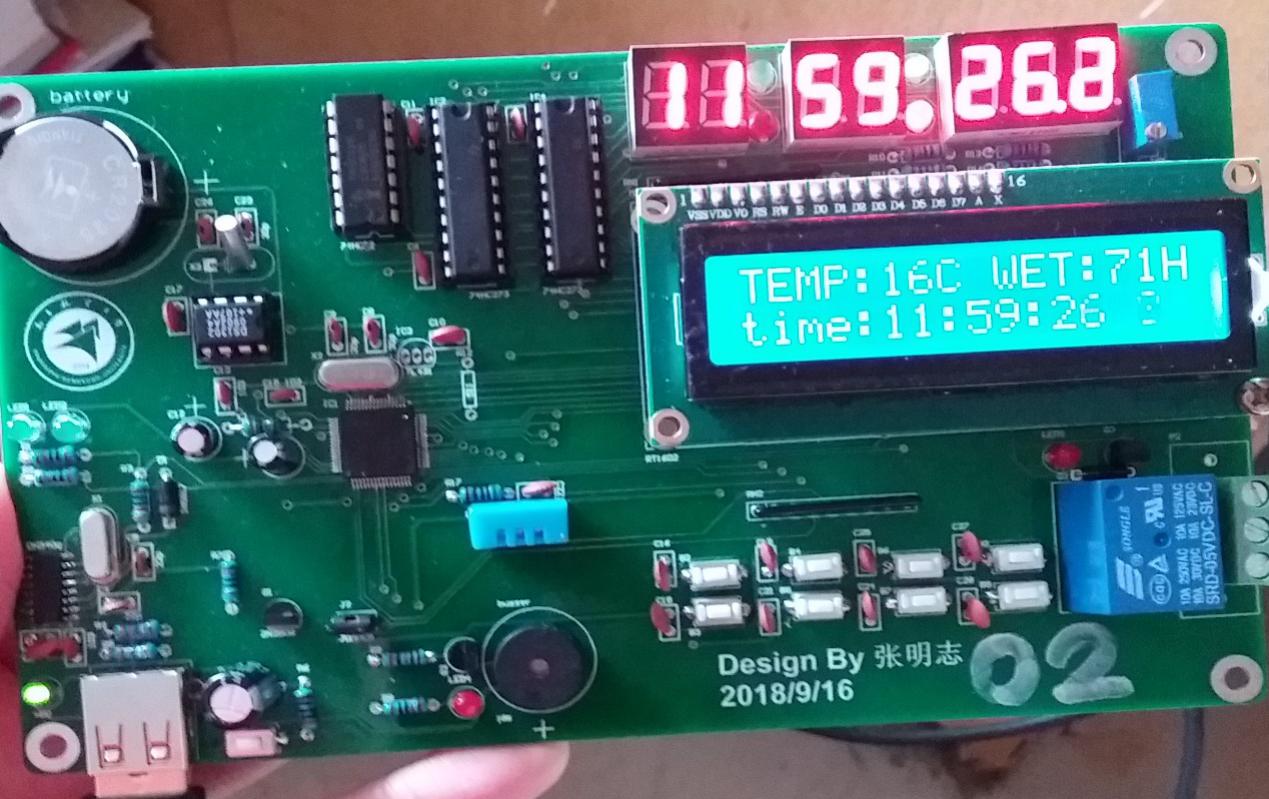
### [4]梅丽风，王艳秋，汪毓铎.单片机原理及接口技术[M].清华人学出版社，2003.5.

### [5]何立民.单片机应用系统设计[M].北京航空航天大学出版社2002.

### [6]胡汉才.单片机原理及其接口技术[W].清华大学出版社，2004.10.

附录A 焊接实物图片

****

****

**附录B 实验程序**

#include "STC8xxxx.H"

#include"1602.h"

#include <stdio.h>

#include <absacc.h>

#define Displaydata XBYTE [0x7fff] //段码端口

#define Displaybit XBYTE [0xbfff] //位码端口

u8 code Stab[10] = {0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f};//段码表

u8 code Btab[8] = {0xfe,0xfd,0xfb,0xf7,0xef,0xdf,0xbf,0x7f}; //位码表

u8 buf[7] = {0x3f,0xbf,0x3f,0x6f,0x6d,0x06,0x06};//显示缓冲区定义并初始化11:59:00.0

u8 hour = 11,minute = 59,second = 0,ms100 = 0;//时、分、秒、小数秒定义并初始化

u16 i = 0,One\_ms\_number = 0; //1ms计数器

u8 M = 0,P1\_value = 0xf3; //扫描显示位计数器和P1口的4个LED指示灯的值LED0、LED1不亮LED2、LED3亮

bit ms\_flag = 0; //新的毫秒标志

u8 code a[]="time: : : "; //显示时间

u8 code b[]="TEMP: WET:"; //显示温湿度

extern u8 rec\_dat[9]; //用于显示的接收数据数组

extern u8 TIME[7]; //时钟芯片用

extern u8 code WRITE\_RTC\_ADDR[];

sbit beep=P7^1;

void delay(u8 s)

{

u8 i,j;

for(i=s;i>0;i--)

for(j=110;j>0;j--);

}

/\*\*\*\*\*\*\*\*\*\*\* T0初始化 \*\*\*\*\*\*\*\*\*\*\*/

void T0\_Init(void)

{ AUXR &= 0x7f; //定时器时钟12T模式

TMOD = 0x21; //设置定时器0工作在方式1定时

TL0 = 0x67; //低字节，1ms溢出一次的时间常数FC67H

TH0 = 0xFC; //高字节，(65536-FC67H)/(11.0592/12)=999.4微秒

EA = 1; //开中断

ET0 = 1; //

TR0 = 1; //定时器0开始计时

}

/\*\*\*\*\*\*\*\*\*\*\*时钟初始化\*\*\*\*\*\*\*\*\*\*\*/

void clock\_init(void)

{ P\_SW2 = 0x80;

CKSEL = 0x00; //选择内部IRC

P\_SW2 = 0x00;

P\_SW2 = 0x80;

XOSCCR = 0xc0; //启动外部晶体

while(!(XOSCCR & 1)); //等待外部晶体稳定

CLKDIV = 0x00;

CKSEL = 0x01; //选择外部晶体

P\_SW2 = 0x00;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*指令延时\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void delayms(unsigned char t)

{

unsigned char i,j;

for(i=0;i<10;i++)

for(j=0;j<t;j++);

}

/\*\*\*\*\*\*\*\*LCD忙检测\*\*\*\*\*\*\*\*\*\*/

unsigned char lazy\_detector( )

{

unsigned char busy\_flag;

RS=0;

RW=1;

EN=1;

busy\_flag=(P0 & 0x80)>>8;

EN=0;

return busy\_flag;

}

/\*\*\*\*\*\*\*\*\*\*\*写指令\*\*\*\*\*\*\*\*\*\*/

void command\_write(unsigned char data1)

{

while(lazy\_detector());

RS=0;

RW=0;

EN=0;

delayms(1);

P0=data1;

delayms(3);

EN=1;

delayms(1);

EN=0;

delayms(1);

}

/\*\*\*\*\*\*\*\*\*\*写数据\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void data\_write(unsigned char data2)

{

while(lazy\_detector());

RS=1;

RW=0;

EN=0;

delayms(1);

P0=data2;

delayms(3);

EN=1;

delayms(1);

EN=0;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*1602初始化\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void init\_1602()

{

delayms(15);

command\_write(0x38);

delayms(1);

command\_write(0x38);

delayms(1);

command\_write(0x0c);

delayms(1);

command\_write(0x0c);

delayms(1);

command\_write(0x06);

delayms(1);

command\_write(0x06);

delayms(1);

command\_write(0x01);

delayms(1);

command\_write(0x01);

delayms(1);

}

//显示时分秒函数

void write\_sfm(u8 a,u8 b) //1602刷新时分秒函数，4为时，7为分，10为秒

{

u8 shi,ge;

shi=b/10;

ge=b%10;

command\_write(0x80+0x40+a);

data\_write(0x30+shi);

data\_write(0x30+ge);

}

void main()

{

int j=0,c,d=0;

unsigned char t[2]={0},h[2]={0};

BUS\_SPEED = 0x03; //降低总线速度

T0\_Init(); //T0初始化

for(i = 0;i < 65535;i ++) //延时，等待内部振荡器稳定

clock\_init(); //时钟初始化，选用外部晶体

init\_1602();

P1 = P1\_value; //4个发光二极管亮

//-----液晶显示固定值-----

command\_write(0x01);

command\_write(0x80);

for(c=-3;c<13;c++)

{

data\_write(b[c]);

delay(10);

}

command\_write(0x80+0x40);

for(c=0;c<13;c++)

{

data\_write(a[c]);

delay(10);

}

write\_sfm(11,second);

write\_sfm(8,minute);

write\_sfm(5,hour);

////////////////////////

while(1)

{

if(ms\_flag == 1)

{ ms\_flag = 0; //到了1ms，标志清0

One\_ms\_number ++; //1ms计数加1

if(One\_ms\_number > 99) //100ms到了

{ One\_ms\_number -= 100; //清除100ms

ms100 += 1; //小数秒加1

if(ms100 > 9) //小数秒加到10

{ ms100 = 0; //小数秒清0

second ++; //秒+1

if(second > 59) //1分钟时间到

{ second = 0; //秒清零

minute ++; //分钟加1

if(minute > 59) //1小时时间到

{ minute = 0; //分钟清零

hour ++; //小时加1

if(hour > 23) hour = 0; //超过了1天则小时清零

buf[6] = Stab[hour / 10]; //取出小时的十位段码送给7号数码管缓存

buf[5] = Stab[hour % 10] | 0x80;//取出小时的个位段码送给6号数码管缓存

write\_sfm(5,hour); //更新1602时显示

} //小时修改结束

buf[4] = Stab[minute / 10];//取出分钟的十位段码送给5号数码管缓存

buf[3] = Stab[minute % 10] | 0x80;//取出小时的个位段码送给4号数码管缓存

j=0;

write\_sfm(8,minute); //更新1602分显示

} //分钟修改结束

buf[2] = Stab[second / 10]; //取出秒的十位段码送给3号数码管缓存

buf[1] = Stab[second % 10] | 0x80;//取出秒的个位段码送给2号数码管缓存 小数点一直亮

write\_sfm(11,second); //更新1602秒显示

P1\_value ^= 0x0f; //变反

P1 = P1\_value; //4个发光二极管交替亮

//蜂鸣器响5下

j++;

if(j>10)beep=1;

else beep=~beep;

} //秒修改结束

buf[0] = Stab[ms100]; //小数秒段码送给1号数码管缓存

command\_write(0x80+0x40+14);

data\_write(0x30+ms100); //更新1602小数秒显示

} //小数秒修改结束

} //毫秒标志存在

} //主循环

}

/\*\*\*\*\*\*\*\*\*T0中断服务程序\*\*\*\*\*\*\*\*\*\*/

void t0(void)interrupt 1 using 1 //每1ms中断一次，12T模式

{ TL0 = 0x67; //装T0时间常数低字节

TH0 = 0xFC; //装T0时间常数高字节

if(M > 6) M = 0;

Displaybit = 0xff; //关闭显示，消隐

Displaydata = buf[M]; //对应数码管的段码值送给段码端口U2

Displaybit = Btab[M]; //对应数码管的位码值送给位码端口U3

M ++; //修改到下一次要显示的数码管

ms\_flag = 1; //设置新毫秒标志

}