

数字电子钟设计说明报告

单片机原理及应用课内实践



2014-4-8

刘子洋 2011301430021

电子信息学院 测控技术与仪器专业

目录

[一、设计概述 4](#_Toc384708225)

[1.进行设计的目的 4](#_Toc384708226)

[2.设计简介 4](#_Toc384708227)

[二、硬件部分 5](#_Toc384708228)

[1.MCU 5](#_Toc384708229)

[复位电路 5](#_Toc384708230)

[时钟源 5](#_Toc384708231)

[特殊功能引脚 5](#_Toc384708232)

[2.LCD12864 5](#_Toc384708233)

[电源,复位及片选 5](#_Toc384708234)

[背光 5](#_Toc384708235)

[对比度 5](#_Toc384708236)

[IO 6](#_Toc384708237)

[3.DS1302 6](#_Toc384708238)

[电源 6](#_Toc384708239)

[IO 6](#_Toc384708240)

[时钟源 6](#_Toc384708241)

[4.按键控制 6](#_Toc384708242)

[简述 6](#_Toc384708243)

[按键连接 6](#_Toc384708244)

[编码器 7](#_Toc384708245)

[5.电源 7](#_Toc384708246)

[6.ISP编程口 7](#_Toc384708247)

[三、程序部分 8](#_Toc384708248)

[1.整体程序结构及工程文件简介 8](#_Toc384708249)

[程序整体文件结构 8](#_Toc384708250)

[main.c main.h 8](#_Toc384708251)

[COMMON.c COMMON.h 8](#_Toc384708252)

[IO.h 8](#_Toc384708253)

[DS1302.c DS1302.h 8](#_Toc384708254)

[LCD12864.c LCD12864.h 9](#_Toc384708255)

[LCDClock.c LCDClock.h 9](#_Toc384708256)

[INT.c INT.h 9](#_Toc384708257)

[SYS.c SYS.h 9](#_Toc384708258)

[程序简易框图 9](#_Toc384708259)

[其它杂项信息 10](#_Toc384708260)

[2.通用子程序（COMMON） 10](#_Toc384708261)

[3.LCD12864驱动 10](#_Toc384708262)

[底层数据收发 10](#_Toc384708263)

[通讯函数 10](#_Toc384708264)

[4.DS1302 驱动 11](#_Toc384708265)

[5.电子表控制程序(LCDClock) 11](#_Toc384708266)

[全局变量 11](#_Toc384708267)

[显示时间及日期函数 12](#_Toc384708268)

[时间设定及更新函数 13](#_Toc384708269)

[用户层面的数据操作函数 13](#_Toc384708270)

[6.中断(INT) 14](#_Toc384708271)

[全局变量 14](#_Toc384708272)

[中断初始化及使能函数 14](#_Toc384708273)

[中断服务函数 14](#_Toc384708274)

[\*\*7.菜单功能及按键命令协调框架程序(SYS) 15](#_Toc384708275)

[简述 15](#_Toc384708276)

[标志变量 16](#_Toc384708277)

[菜单函数 17](#_Toc384708278)

[四、待解决问题 18](#_Toc384708279)

[复位电路不可靠 18](#_Toc384708280)

[按键中断偶尔不灵 18](#_Toc384708281)

[时间到页面模糊 18](#_Toc384708282)

[蜂鸣器不响 18](#_Toc384708283)

[五、设计过程简述及感想 19](#_Toc384708284)

[设计过程简述 19](#_Toc384708285)

[设计感想 19](#_Toc384708286)

[设计参考资料 20](#_Toc384708287)

[附: 设备原理图及程序源码 20](#_Toc384708288)

# 一、设计概述

## 1.进行设计的目的

锻炼实际运用能力，并通过实际设计过程来学习使用51系列控制器

## 2.设计简介

该电子表由Atmel公司AT89S52微控制器进行所有部件的协调和控制；通过LCD12864（串行方式）进行状态显示；通过74LS148编码器进行按键管理。同时由DS1302时钟芯片进行主时间的控制和记录。

主计时时钟由DS1302时钟芯片提供。

功能计时器，秒表等常用功能的计时由AT89S52内部定时器完成。内部定时器可进行精度调校。最高精度可达0.5%

另有蜂鸣器，开关，时钟芯片电池，ISP编程口等辅助部件。

硬件基于万能实验板。软件于Keil C51环境下开发。

# 二、硬件部分

## 1.MCU

### 复位电路

复位电路采用上电+按键复位

按键复位由按键产生正脉冲实现

上电复位由电容模拟按键产生正跳变实现

### 时钟源

晶振使用22.1184方形晶振 为便于更换插在排母上与MCU连接 同时接2个30pF电容

### 特殊功能引脚

由于使用内部程序存储器, EA脚始终接+5V

同时PSEN始终接地

## 2.LCD12864

### 电源,复位及片选

由于LCD12864 在设备开启时需要始终运行， 因此RST，CS始终接+5v

### 背光

该显示屏无背光时基本无法看到文字，因此背光电源始终接+5v

### 对比度

液晶电压V0由电位器调整。以适当调整对比度

### IO

由于使用串行通信，因此PSB始终接地

## 3.DS1302

### 电源

VCC2为主电源，与设备电源连接

VCC1低功耗电源，与电池相连

根据DS1302芯片手册，当VCC2电压低于VCC1时，由VCC1供电。此时为设备关闭的状态，用于时间记录

高于VCC1+0.1v时DS1302由VCC2供电。此时为设备开启时的状态

### IO

与MCU通讯口由上拉电阻接+5V以防驱动力不够造成通讯错误

### 时钟源

根据官方手册，X1 X2 接32768Hz晶振并接两个6pF电容去耦

为了防止干扰信号,晶振外壳用铁丝固定并用焊锡连接并接地，晶振下方全部用焊锡填满接地，并靠近芯片

## 4.按键控制

### 简述

此设备原定在AT89C2051上设计,因此IO口紧张 为节省IO口,采用74LC148管理按键中断。优点是原理简单并且自带优先级

### 按键连接

按键由4.7k电阻上拉连接+5v，同时连接编码器输入端

按键另一端接地

为了防止误动作，0号输入没有按键。因此设备上实际为7个按键。详细说明在程序说明部分

### 编码器

由于编码器始终需要工作，因此EI端接地

GS作为有任何输入即输出低电平的输出端，连接至MCU外部中断EX0入口P3.2

A2，A1，A0用来分辨具体是哪位有按键输入，与MCU的IO连接

MCU进入EX0中断后在ISR中会对A2，A1，A0进行分析来确定是哪个按键别按下

## 5.电源

电源由USB线引入，由一个带锁开关控制。同时用104电容并联以去除电源毛刺

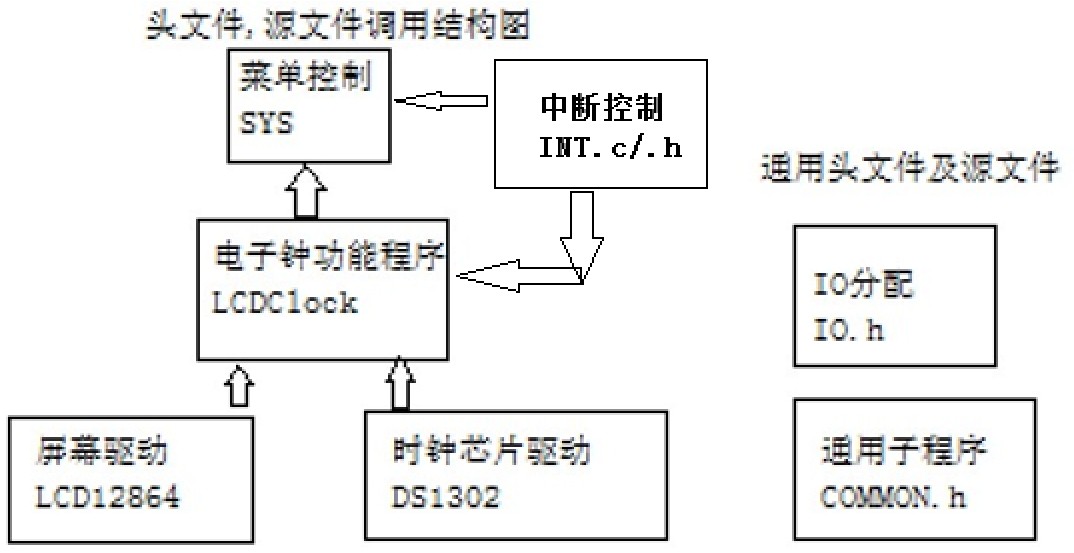
## 6.ISP编程口

ISP编程口采用直接与对应IO及功能口连接的方式。因此在用ISP编程线与计算机连接时，设备不会正常工作

# 三、程序部分

## 1.整体程序结构及工程文件简介

### 程序整体文件结构



### main.c main.h

主函数文件,为程序入口

### COMMON.c COMMON.h

包含一些常用子程序及宏定义

例如U8定义为unsigned char

BCD码与16进制的转换

软件延时函数等

### IO.h

集中用于分配IO口

### DS1302.c DS1302.h

用于存放DS1302的通讯驱动

### LCD12864.c LCD12864.h

用于存放LCD12864的驱动

### LCDClock.c LCDClock.h

用于协调LCD12864及DS1302以组合成数字钟的功能

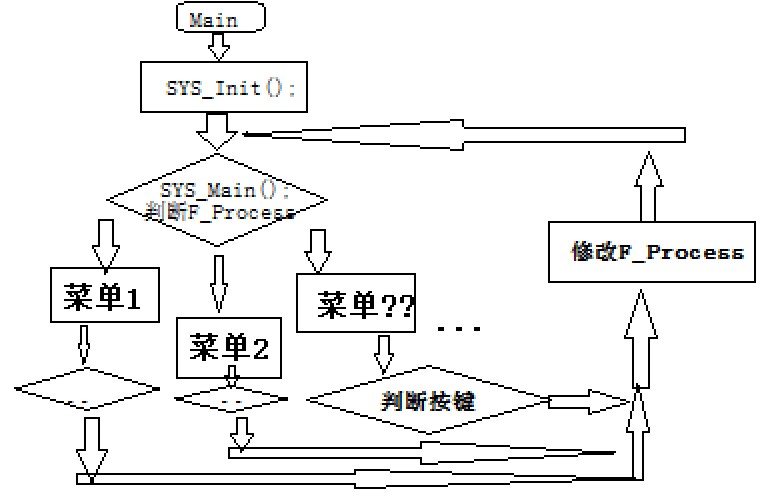
### INT.c INT.h

存放与中断相关的初始化及对应的ISR程序

### SYS.c SYS.h

总程序框架, 用于协调所有菜单控制功能及设备的运行

### 程序简易框图



### 其它杂项信息

1. 为了节省data低128字节空间, 所有局部变量用idata定义在128-256空间内

2. 程序总大小 6408/8000

## 2.通用子程序（COMMON）

#define u8 unsigned char

#define u16 int

为便于记忆及使用将uchar和int分别替换为u8和u16

#define flag unsigned char

所有标志位的类型,为了便于分析程序

#define msg unsigned char

所有消息的类型, 为了便于分析

void delayms(u16 t);

软件延时 通过空操作来延时

u8 Tf\_BCD\_H(u8 BCDdata);

BCD格式数据转换为16进制或10进制数

u8 Tf\_H\_BCD(u8 HEXdata);

16进制数据转换为BCD格式数据

以上两函数主要用于与DS1302通讯

void TIM\_Delayms(u16 t);

利用定时器的延时,与软件延时效果相同,精度稍高

## 3.LCD12864驱动

### 底层数据收发

void LCD\_SendU8(u8 u8Data);

底层数据收发函数.用于用串行模式发送8位数据

void LCD\_RetU8(u8 u8Data);

底层数据收发函数.用于用串行模式接收8位数据

### 通讯函数

void LCD\_WaitBF();

判忙函数 若设备BF忙标志位为1,则等待直至其为0

void LCD\_WRCmd(u8 u8Cmd);

写命令函数

此函数由底层数据收发函数外加设定WR RS为00组成

用于写控制字

void LCD\_WRDat(u8 u8Dat);

写命令函数

此函数由底层数据收发函数外加设定WR RS为01组成

void LCD\_Init();

初始化函数 包含了清屏, 设置模式等一系列初始化命令

void LCD\_CLR();

字符显示模式的清屏函数, 用字库中的空白填满整个屏幕

优点是比绘图方式速度快

void LCD\_NLet(u8 ads, u8\* table, u8 n);

用字库在地址为ads的位置显示ASCII字符, 字符放在table中,n为全角字符数

void LCD\_NChar(u8 ads, u8\* table, u8 n);

用字库在地址为ads的位置显示中文汉字, 字符放在table中,n为半角字符数

## 4.DS1302 驱动

void DS1302\_SendU8(u8 dat);

u8 DS1302\_GetU8();

遵照手册时序图的底层8位数据发送接收函数

void DS1302\_WR(u8 REG, u8 dat);

u8 DS1302\_RD(u8 REG);

数据读写函数

## 5.电子表控制程序(LCDClock)

包含所有与电子钟相关的功能函数

### 全局变量

extern u8 YEAR;

extern u8 MONTH;

extern u8 DATE;

extern u8 HOUR;

extern u8 MINUTE;

extern u8 SECOND;

存放与DS1302直接通讯用的BCD格式的时间8位数据

extern flag F\_HourMode;

显示方式标志位 24小时模式为0, 12小时模式为1

extern flag F\_AMPM;

AMPM表示为,为1表示PM

extern u8 uYEAR;

extern u8 uMONTH;

extern u8 uDATE;

extern u8 uHOUR;

extern u8 uMINUTE;

extern u8 uSECOND;

extern u8 uSECD100;

用户层面的时间数据, 用于调整时间等 格式为16进制或10进制

extern u8 tHOUR;

extern u8 tMINUTE;

extern u8 tSECOND;

extern u8 tSECD100;

秒表专用的时间变量. 单独定义是为了让秒表可以在其它界面下继续运行 16进制

extern u8 aHOUR;

extern u8 aMINUTE;

闹钟设定时间变量,16进制

extern u8 cHOUR;

extern u8 cMINUTE;

闹钟监视的目前时钟变量16进制

### 显示时间及日期函数

void Clock\_DispDate(u8 year,u8 month,u8 date);

根据输入的10进制数,以设定好的格式显示日期

void Clock\_DispTime(u8 hour,u8 min,u8 sec);

根据输入的10进制数,以设定好的格式显示时间

### 时间设定及更新函数

void Clock\_SetTime(u8 year,u8 month,u8 date,u8 hour,u8 min,u8 sec,u8 mode);

根据输入的10进制数设定时间并写入DS1302

void Clock\_UpdateTime();

由DS1302读取BCD格式程序保存

void Clock\_ShowTime();

void Clock\_ShowDate();

显示当前MCU获取的时间及日期

以上4个函数由获取数据+显示时间来组合完成

### 用户层面的数据操作函数

void Clock\_ShowSetT();

显示当前设定模式下调整到的时间

例如设定时手动将时间调整到某个时间

这些时间数据不会被写入DS1302

void Clock\_GetForSet();

将从DS1302读取来的BCD数据转换格式并复制到用户层面的时间变量中

void Clock\_Tim\_CLR();

清除秒表

void Clock\_Tim\_Disp();

秒表时间显示

秒表由定时器0来完成计时 精度可达0.1%

void Clock\_rvTim\_Disp();

显示倒计时时钟函数

void Clock\_Alarm\_Disp();

闹钟设定时间显示函数

void Clock\_Alarm\_CLR();

闹钟复位函数 此函数会在每次闹钟响时自动调用

void Clock\_Alarm\_Chk();

读取并翻译当前时间,以便于闹钟设定时间比对

## 6.中断(INT)

### 全局变量

extern u16 t0ms;

用于记录定时器T0运行的毫秒数

extern u16 t1ms;

用于记录定时器T1运行的毫秒数

extern u8 TIM\_Modify;

用于存放初值修正值.

修正值用于修正初值数值以使其能尽量接近每一毫秒一次中断

### 中断初始化及使能函数

void INT\_EX0\_Init();

外部中断初始化函数 主要包含相关寄存器的置位操作

void INT\_TIM0\_Init();

定时器中断初始化函数

void INT\_TIM0\_Cmd();

定时器启动函数 主要包含重装技术值及TR0的置位

void INT\_TIM1\_Init();

void INT\_TIM1\_Cmd();

### 中断服务函数

void ISR\_EX0() interrupt 0

外部中断0的ISR, 完成的工作有

1.自动定时复位计数清0 防止设备自动回到主界面

2.读取按键编码IO A2,A1,A0 并将其分别赋予权相加,赋值给消息msg\_Keys

3.软件延时100毫秒以防止抖动产生误动作

void ISR\_TIM0() interrupt 1

定时器0的溢出中断ISR ,主要为秒表及定时器(倒计时)工作

1.每次溢出时判断t0ms的大小 即过了多少毫秒

2.根据标志位F\_TMode来确定是秒表还是倒计时定时器. 如果是秒表,则每次t0ms达到一定数值,将其清零, 并使以t开头的时间变量增加并进位. 如果是倒计时, 则使这些变量减少并在适当时候借位. 当无位可借时,会将时间到标志位F\_Timeout置1

3.重装计数值

void ISR\_TIM1() interrupt 3

定时器1的溢出中断ISR,主要用于设备菜单辅助功能. 计时原理及模式与T0相同.

## \*\*7.菜单功能及按键命令协调框架程序(SYS)

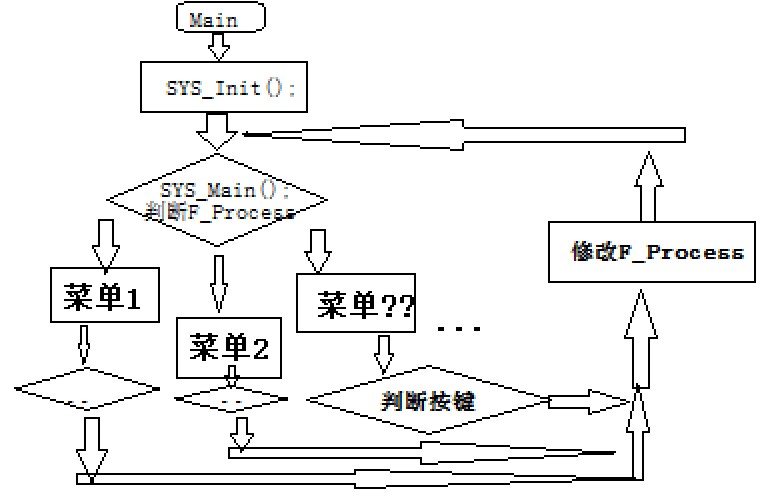
### 简述

电子钟许多功能的实现具有实时性, 在某一时刻必须在任何情况下都能响应某个事件的发生. 并且要给用户操作的空间. SYS.c与SYS.h中的程序就是为了实现这些功能.

主要工作方式为:

主函数是一个while(1)死循环, 每次循环会运行一些检测运行状态标志变量F\_Process. 根据该标志位的数值来决定各个菜单子程序的执行. 菜单子程序不包括循环部分. 在各个菜单的状态下, 每个子程序都包含检测按键消息msg\_Keys 的函数, 根据检测到的数值来修改F\_Process. 在下一次主函数死循环检测到F\_Process时程序会分配到另一个相应的菜单函数中. 例如F\_Process一直为0时, 程序每次循环都会分配去执行待机画面函数. 当某次检测到msg\_Keys = 1时, F\_Process会被置1, 在之后死循环检测F\_Process时, 会执行主菜单函数SYS\_Menu();

由于MCU执行的循环速度非常快, 因此人的感觉上即为 按键之后,或者发生某一事件之后,画面立即改变到另一菜单.同时, 每次进入死循环时, 程序会将F\_Process与上一次的F\_Process进行比对 若不同, 会进行清屏等操作. 这样就实现了某些只需要在切换页面时才需要的功能仅执行一次. 这样做的主要优点是能实现近似实时的响应, 同时便于以后添加功能. 并且节省中断通道



### 标志变量

extern flag F\_Timeup;

时间到标变量, 闹钟到达设定时间或倒计时器时间变为0时该标志位置1

extern u16 F\_Process;

主程序的步骤标志变量 用于判断每次循环进入的子程序

extern flag F\_TMode;

计时器工作方式标志变量 0表示为秒表,1表示为倒计时器

extern flag F\_Return;

当10秒内无动作会使其置1, 表明需要返回待机界面

extern flag F\_AlarmEN;

闹钟标志变量, 1表示闹钟开启

extern msg msg\_Keys;

按键消息 1-7对应7个按键 为8时表示无按键按下

由于干扰问题有时会产生并非需要的中断. 而此时msg\_Keys一定为0. 因此编码器0输入端不接按键. 而当发生中断并且msg\_Keys = 0时,程序除了将msg\_Keys改为默认无按键按下的8之外不做任何事

### 菜单函数

void SYS\_Main();

主循环函数 每次循环根据F\_Process决定运行某一个菜单子程序

void SYS\_Init();

初始化函数. 用于所有变量, 定时器, 中断的初始化

void SYS\_CLR();

重新初始化函数

void SYS\_Pointer();

菜单选中项标签函数

void SYS\_Prep();

准备函数. 在每次循环扫描F\_Process之前调用.

主要功能有 更新数据, 判断页面改变后清屏, 根据当前页面复位某一功能等

void SYS\_Menu();

主菜单函数

void SYS\_Default();

待机画面

void SYS\_SubM\_Fts();

功能选择页面函数

void SYS\_SubM\_Set();

时间设置函数. 进入该页面会使SYS\_Prep();将DS1302收回的暂存BCD格式数据HOUR等转换为16进制并存入uHOUR等变量. 之后以uHOUR等变量为基础进行时间设置的显示, 按确定将把数据送入DS1302

void SYS\_SSubM\_TIM();

秒表界面

void SYS\_SSubM\_rvTIM();

倒计时计时器界面

void SYS\_SSubM\_SetAlarm();

闹钟设置界面

void SYS\_SV\_Alarm();

闹钟监视函数. 于主函数扫描之前调用

void SYS\_Success();

成功界面 进入该界面后3秒会将F\_Process清0

void SYS\_Timeup();

时间到界面函数. 进入该界面会清除所有闹钟, 计时器状态变量

void SYS\_Info();

程序信息界面

# 四、待解决问题

由于时间紧迫, 有许多问题还没有想到解决的办法. 这些问题会在以后尽力解决

### 复位电路不可靠

复位电路为上电+按键复位. 但实际情况为上电后有一定几率单片机什么都不做. 并且有极少数的情况上电后, DS1302中的时间中的某位会被抹掉

### 按键中断偶尔不灵

有些时候 按键会无效 需要重启设备才能修正这个错误

### 时间到页面模糊

在时间到页面会明显有刷屏感 且非常模糊

### 蜂鸣器不响

用2N5551驱动蜂鸣器, 在外独立搭电路接5v会响 但接在设备上不会工作

猜想可能是由于AT89S52内置上拉电阻过大导致.

# 五、设计过程简述及感想

### 设计过程简述

除了最后一周, 每晚5个小时左右, 总共历时1个月. 第一周主要是熟悉C51. 在开发板的AT89S52上初次尝试LCD12864 因为时第一次用串行设备. 花费了一周多的时间才懂. 后来用AT89C2051在万能板上搭了个最小系统, 成功运行. 之后的LCD编程都是以这块板为基础. 因为自己刻意多放了许多电源排针, LCD插起电源非常方便. 之后的一周花在了DS1302的调试上. 因为这个芯片任何地方程序有错都不会工作, 这就导致查错比较麻烦. 第三周主要是写SYS菜单程序. 由于程序容量此时超过了2000,因此重新回到开发板上用40脚的AT89S52进行调试. 最后一周主要是将开发板上的系统迁移出来到正式的实物上. 期间遇到了各种干扰, 电压不稳, 驱动力不足等问题. 实物电路搭好之后, 继续添加,改进功能程序. 最后测试并做一些修改尽量提高设备稳定性

### 设计感想

此次课程设计最大的感想就是. 在开发过程中遇到的许多问题都是看似难以想象, 其实通过思考都可以很快解决. 几乎所有解决不了的问题都可以归结为 ”时机未到” 即功夫没下够. 一旦花费了够多的时间, 都是可以解决的. 现在回想起刚着手的时候, 当初的确是难以想象如何把这些外设协调起来. 尤其是屏幕. 但通过花费足够时间, 这些全都实现了. 整个开发过程98%都是在查错与重新尝试上. 几乎没有一件事是可以一次成功的. 这次设计之后我有些习惯了这种节奏. 在设计的前期, 发现异常的时候我会非常不舒服. 但在后期的时候, 发现错误之后反而觉得很正常.

但这次开发中许多教训也让我明白, 有时候不能盲目地尝试. 例如在搭电路时没有看AT89S52的官方芯片手册. 结果上拉电阻过小导致电平无法压低. 设备不工作, 还以为是单片机最小系统没搭好, 花了一天时间才查出是上拉电阻问题, 之后又花了大量的时间在修改电路上. 回想如果当初能仔细通读一遍芯片手册, 也许会节省许多不必要花费的时间.

从知识和认识上, 此次课程设计让我完全改变了对单片机定位的认识. 现在我明白了单片机为什么又叫控制器, 并且更加喜欢用控制器这个名称. 通过这次设计, 我发现MCU芯片在许多设备中, 起到的作用正是控制协调外设一起工作. 是开发人员指挥外设用的指挥棒. 而不是工作系统的全部. 学习MCU并没有花费太多的时间, 反而是大量时间花费在如何指挥这些外设上. 就好像玩游戏的时候, 鼠标是必不可少的的. 因为没有鼠标, 无法进行有效的操作. 但玩游戏许多时间都是花费在如何玩好上, 而一般人只花了少量时间在学习如何用好鼠标上.

设计的过程中还是充满挑战的乐趣的. 特别是每当解决一个异常问题时瞬间心中的成就感. 以后有空闲时间的时候, 我会继续尽量多地做各种设计. 通过利用理论知识来设计实际应用是一个效率非常高的学习方法. 因为在设计过程中接触次数越多的知识, 越是那些重要的知识.

# 设计参考资料

<单片机原理及接口技术> 李朝青

AT89S52官方芯片手册

LCD12864 手册

DS1302 芯片手册

74LS148 手册

2N5551 芯片手册

慧净51系列视频教程

# 附: 设备原理图及程序源码

### main.c

#include<reg52.h>

#include "COMMON.h"

#include "SYS.h"

#include "IO.h"

#include "LCD12864.h"

#include "INT.h"

void main(void)

{

SYS\_Init();

while(1)

{

SYS\_Main();

}

}

### COMMON.h

//keys msg

#define u8 unsigned char

#define u16 int

#define flag unsigned char

#define msg unsigned char

void delayms(u16 t);

u8 Tf\_BCD\_H(u8 BCDdata);

u8 Tf\_H\_BCD(u8 HEXdata);

void TIM\_Delayms(u16 t);

### COMMON.C

#include "COMMON.h"

#include<reg52.h>

#include "INT.h"

void delayms(u16 t)

{

int i;

int j;

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

for(j=0;j<246;j++);

}

u8 Tf\_BCD\_H(u8 BCDdata)

{

u8 temp,Hdata;

temp = (BCDdata>>4)&0x0f;

BCDdata &= 0x0f;

Hdata = BCDdata+temp\*10;

return Hdata;

}

u8 Tf\_H\_BCD(u8 HEXdata)

{

u8 bcdH,bcdL;

bcdH = ((HEXdata/10)<<4)&0xf0;

bcdL = HEXdata%10;

bcdL = bcdH+bcdL;

return bcdL;

}

void TIM\_Delayms(u16 t)

{

TR0 = 0;

INT\_TIM0\_Cmd();

while(1)

{

if(t0ms > t)break;

}

}

### IO.h

#include<reg52.h>

#include "COMMON.h"

/\*------------------------------IO------------------------------\*/

#define BF 0x80

//URT

//sbit LCD\_CS = P1^2;

sbit LCD\_SID = P2^7;

sbit LCD\_SCLK = P2^6;

//DS1302

sbit DS1302\_IO = P1^3;

sbit DS1302\_CS = P1^4;

sbit DS1302\_SCLK = P1^2;

//key

sbit Key\_A0 = P2^3;

sbit Key\_A1 = P2^5;

sbit Key\_A2 = P2^4;

sbit AlarmBeep = P1^1;

### LCD12864.h

#include "COMMON.h"

void LCD\_SendU8(u8 u8Data);

u8 LCD\_RetU8();

void LCD\_WaitBF();

void LCD\_WRCmd(u8 u8Cmd);

void LCD\_WRDat(u8 u8Dat);

void LCD\_Init();

void LCD\_CLR();

void LCD\_NLet(u8 ads, u8\* table, u8 n);

void LCD\_NChar(u8 ads, u8\* table, u8 n);

### LCD12864.c

#include<reg52.h>

#include "COMMON.h"

#include "LCD12864.h"

#include "IO.h"

void LCD\_SendU8(u8 u8Data)

{

u8 idata i;

for(i=0;i<8;i++)

{

LCD\_SCLK = 0;

u8Data=u8Data<<1;

LCD\_SID = CY;

LCD\_SCLK = 1;

LCD\_SCLK = 0;

}

}

u8 LCD\_RetU8()

{

u8 idata i,u8H,u8L;

u8H = 0;

u8L = 0;

for(i=0;i<8;i++)

{

u8H=u8H<<1;

LCD\_SCLK = 0;

LCD\_SCLK = 1;

LCD\_SCLK = 0;

if(LCD\_SID)u8H = 1;

}

for(i=0;i<8;i++)

{

u8L=u8L<<1;

LCD\_SCLK = 0;

LCD\_SCLK = 1;

LCD\_SCLK = 0;

if(LCD\_SID) u8L = 1;

}

return ((0xf0&u8H)+(0x0f&u8L));

}

void LCD\_WaitBF()

{

LCD\_SendU8(0xfc);

while(0x80&LCD\_RetU8());

}

void LCD\_WRCmd(u8 u8Cmd)

{

LCD\_WaitBF();

LCD\_SendU8(0xf8);

LCD\_SendU8(0xf0&u8Cmd);

LCD\_SendU8(0xf0&u8Cmd<<4);

}

void LCD\_WRDat(u8 u8Dat)

{

LCD\_WaitBF();

LCD\_SendU8(0xfa);

LCD\_SendU8(0xf0&u8Dat);

LCD\_SendU8(0xf0&u8Dat<<4);

}

void LCD\_Init()

{

delayms(20);

LCD\_WRCmd(0x30);

delayms(20);

LCD\_WRCmd(0x06);

delayms(20);

LCD\_WRCmd(0x01);

delayms(20);

LCD\_WRCmd(0x0C);

delayms(20);

LCD\_WRCmd(0x02);

delayms(20);

}

void LCD\_CLR()

{

u8 idata CLRtable[16] = {0x20,0x20,0x20,0x20,

0x20,0x20,0x20,0x20,

0x20,0x20,0x20,0x20,

0x20,0x20,0x20,0x20};

LCD\_NLet(0x80, CLRtable, 8);

LCD\_NLet(0x90, CLRtable, 8);

LCD\_NLet(0x88, CLRtable, 8);

LCD\_NLet(0x98, CLRtable, 8);

}

void LCD\_NLet(u8 ads, u8\* table, u8 n)

{

u8 idata i;

LCD\_WRCmd(ads);

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

{

LCD\_WRDat(table[2\*i]);

LCD\_WRDat(table[2\*i+1]);

}

}

void LCD\_NChar(u8 ads, u8\* table, u8 n)

{

u8 idata i;

LCD\_WRCmd(ads);

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

{

LCD\_WRDat(table[i]);

}

}

### DS1302.h

void DS1302\_SendU8(u8 dat);

u8 DS1302\_GetU8();

void DS1302\_WR(u8 REG, u8 dat);

u8 DS1302\_RD(u8 REG);

### DS1302.c

#include<reg52.h>

#include "COMMON.h"

#include "DS1302.h"

#include "IO.h"

void DS1302\_SendU8(u8 U8dat)

{

u8 idata i;

DS1302\_SCLK=0;

for(i=0;i<8;i++)

{

if(U8dat&0x01)

DS1302\_IO=1;

else

DS1302\_IO=0;

DS1302\_SCLK=1;

U8dat>>=1;

DS1302\_SCLK=0;

}

}

u8 DS1302\_GetU8()

{

u8 idata i;

u8 idata u8Dat;

DS1302\_SCLK=0;

for(i=0;i<8;i++)

{

u8Dat>>=1;

if(DS1302\_IO)

u8Dat|=0x80;

else

DS1302\_IO&=0x7f;

DS1302\_SCLK=1;

DS1302\_SCLK=0;

}

return u8Dat;

}

void DS1302\_WR(u8 REG, u8 dat)

{

u8 idata temp;

temp = dat;

DS1302\_CS=1;

DS1302\_SendU8(REG);

DS1302\_SendU8(dat);

DS1302\_CS=0;

}

u8 DS1302\_RD(u8 REG)

{

u8 idata Dat;

DS1302\_CS=1;

DS1302\_SendU8(REG);

Dat = DS1302\_GetU8();

DS1302\_CS=0;

return Dat;

}

### LCDClock.h

#include "COMMON.h"

//----------------------DS1302Time---------------------//

extern u8 YEAR;

extern u8 MONTH;

extern u8 DATE;

extern u8 HOUR;

extern u8 MINUTE;

extern u8 SECOND;

extern flag F\_HourMode;

extern flag F\_AMPM;

extern u8 uYEAR;

extern u8 uMONTH;

extern u8 uDATE;

extern u8 uHOUR;

extern u8 uMINUTE;

extern u8 uSECOND;

extern u8 uSECD100;

extern u8 tHOUR;

extern u8 tMINUTE;

extern u8 tSECOND;

extern u8 tSECD100;

extern u8 aHOUR;

extern u8 aMINUTE;

extern u8 cHOUR;

extern u8 cMINUTE;

void Clock\_DispDate(u8 year,u8 month,u8 date);

void Clock\_DispTime(u8 hour,u8 min,u8 sec);

void Clock\_SetTime(u8 year,u8 month,u8 date,u8 hour,u8 min,u8 sec,u8 mode);

void Clock\_UpdateTime();

void Clock\_ShowTime();

void Clock\_ShowDate();

void Clock\_ShowSetT();

void Clock\_GetForSet();

void Clock\_Tim\_CLR();

void Clock\_Tim\_Disp();

void Clock\_rvTim\_Disp();

void Clock\_Alarm\_Disp();

void Clock\_Alarm\_CLR();

void Clock\_Alarm\_Chk();

### LCDClock.c

#include<reg52.h>

#include "LCD12864.h"

#include "COMMON.h"

#include "LCDClock.h"

#include "DS1302.h"

#include "IO.h"

#include "INT.h"

#include "SYS.h"

flag F\_HourMode;

flag F\_AMPM;

u8 YEAR;

u8 MONTH;

u8 DATE;

u8 DAY;

u8 HOUR;

u8 MINUTE;

u8 SECOND;

u8 uYEAR;

u8 uMONTH;

u8 uDATE;

u8 uHOUR;

u8 uMINUTE;

u8 uSECOND;

u8 uSECD100;

u8 tHOUR;

u8 tMINUTE;

u8 tSECOND;

u8 tSECD100;

u8 aHOUR;

u8 aMINUTE;

u8 cHOUR;

u8 cMINUTE;

flag F\_uHourMode;

flag F\_uAMPM;

void Clock\_DispDate(u8 year,u8 month,u8 date)

{

u8 idata datet[10];

datet[0] = 0x30+(month/10);

datet[1] = 0x30+(month%10);

datet[2] = 0x2F;

datet[3] = 0x30+(date/10);

datet[4] = 0x30+(date%10);

datet[5] = 0x2F;

datet[6] = 0x32;

datet[7] = 0x30;

datet[8] = 0x30+(year/10);

datet[9] = 0x30+(year%10);

LCD\_NLet(0x80,datet,5);

}

void Clock\_DispTime(u8 hour,u8 min,u8 sec)

{

u8 idata time[8];

u8 idata AMPM[2];

time[0] = 0x30+(hour/10);

if(!(hour/10)) time[0] = 0x20;

time[1] = 0x30+(hour%10);

time[2] = 0x3A;

time[3] = 0x30+(min/10);

time[4] = 0x30+(min%10);

time[5] = 0x3A;

time[6] = 0x30+(sec/10);;

time[7] = 0x30+(sec%10);

LCD\_NLet(0x89,time,4);

if(F\_HourMode)

{

AMPM[1] = 0x4D;

if(F\_AMPM)

AMPM[0] = 0x50;

else

AMPM[0] = 0x41;

}

else

{

AMPM[0] = 0x20;

AMPM[1] = 0x20;

}

LCD\_NLet(0x8E,AMPM,1);

}

void Clock\_SetTime(u8 year,u8 month,u8 date,u8 hour,u8 min,u8 sec,u8 mode)

{

u8 idata i;

F\_HourMode = mode;

for(i=0;i<2;i++)

{

SECOND = Tf\_H\_BCD(sec);

MINUTE = Tf\_H\_BCD(min);

if(!F\_HourMode)

{

HOUR = Tf\_H\_BCD(hour);

}

else

{

if(hour>12)

{

hour -= 12;

HOUR = Tf\_H\_BCD(hour)|0xA0;

F\_AMPM = 1;

hour += 12;

}

else

{

HOUR = Tf\_H\_BCD(hour)|0x80;

F\_AMPM = 0;

}

}

YEAR = Tf\_H\_BCD(year);

MONTH = Tf\_H\_BCD(month);

DATE = Tf\_H\_BCD(date);

DS1302\_WR(0x8E,0x00);

DS1302\_WR(0x80,0x80);

DS1302\_WR(0x80,SECOND);

DS1302\_WR(0x82,MINUTE);

DS1302\_WR(0x84,HOUR);

DS1302\_WR(0x86,DATE);

DS1302\_WR(0x88,MONTH);

DS1302\_WR(0x8C,YEAR);

DS1302\_WR(0x8E,0x80);

}

}

void Clock\_UpdateTime()

{

SECOND = DS1302\_RD(0x81);

MINUTE = DS1302\_RD(0x83);

HOUR = DS1302\_RD(0x85);

F\_AMPM = ((HOUR&0x20)>>5)&0x01;

DATE = DS1302\_RD(0x87);

MONTH = DS1302\_RD(0x89);

YEAR = DS1302\_RD(0x8D);

}

void Clock\_ShowTime()

{

if(!F\_HourMode)

Clock\_DispTime(Tf\_BCD\_H(HOUR),Tf\_BCD\_H(MINUTE),Tf\_BCD\_H(SECOND));

else

{

u8 idata hour12;

hour12 = Tf\_BCD\_H(HOUR&0x1f);

Clock\_DispTime(hour12,Tf\_BCD\_H(MINUTE),Tf\_BCD\_H(SECOND));

}

}

void Clock\_ShowDate()

{

Clock\_DispDate(Tf\_BCD\_H(YEAR),Tf\_BCD\_H(MONTH),Tf\_BCD\_H(DATE));

}

void Clock\_ShowSetT()

{

Clock\_DispTime(uHOUR,uMINUTE,uSECOND);

Clock\_DispDate(uYEAR,uMONTH,uDATE);

}

void Clock\_GetForSet()

{

uYEAR = Tf\_BCD\_H(YEAR);

uMONTH = Tf\_BCD\_H(MONTH);

uDATE = Tf\_BCD\_H(DATE);

if(!F\_HourMode)uHOUR = Tf\_BCD\_H(HOUR);

else uHOUR = Tf\_BCD\_H(HOUR&0x1f);

uMINUTE = Tf\_BCD\_H(MINUTE);

uSECOND = Tf\_BCD\_H(SECOND);

}

void Clock\_Tim\_CLR()

{

t0ms = 0;

tSECD100 = 0;

tSECOND = 0;

tMINUTE = 0;

tHOUR = 0;

}

void Clock\_Tim\_Disp()

{

u8 idata time[12];

time[0] = 0x30+(tHOUR/10);

if(!(tHOUR/10)) time[0] = 0x20;

time[1] = 0x30+(tHOUR%10);

time[2] = 0x3A;

time[3] = 0x30+(tMINUTE/10);

time[4] = 0x30+(tMINUTE%10);

time[5] = 0x3A;

time[6] = 0x30+(tSECOND/10);

time[7] = 0x30+(tSECOND%10);

time[8] = 0x3A;

time[9] = 0x30+(tSECD100/10);

time[10] = 0x30+(tSECD100%10);

time[11] = 0x20;

LCD\_NLet(0x89,time,6);

}

void Clock\_rvTim\_Disp()

{

u8 idata time[8];

time[0] = 0x30+(tHOUR/10);

if(!(tHOUR/10)) time[0] = 0x20;

time[1] = 0x30+(tHOUR%10);

time[2] = 0x3A;

time[3] = 0x30+(tMINUTE/10);

time[4] = 0x30+(tMINUTE%10);

time[5] = 0x3A;

time[6] = 0x30+(tSECOND/10);

time[7] = 0x30+(tSECOND%10);

LCD\_NLet(0x89,time,4);

}

void Clock\_Alarm\_Disp()

{

u8 idata time[6];

u8 idata status[] = "闹钟状态";

u8 idata aEnable[] = "启用";

u8 idata aDisable[] = "禁用";

time[0] = 0x30+(aHOUR/10);

if(!(aHOUR/10)) time[0] = 0x20;

time[1] = 0x30+(aHOUR%10);

time[2] = 0x3A;

time[3] = 0x30+(aMINUTE/10);

time[4] = 0x30+(aMINUTE%10);

time[5] = 0x20;

LCD\_NLet(0x8A,time,3);

LCD\_NChar(0x91,status,8);

if(F\_AlarmEN)LCD\_NChar(0x96,aEnable,4);

else LCD\_NChar(0x96,aDisable,4);

}

void Clock\_Alarm\_CLR()

{

aHOUR = 0;

aMINUTE = 0;

F\_AlarmEN = 0;

}

void Clock\_Alarm\_Chk()

{

u8 idata temp;

if(!F\_HourMode)cHOUR = Tf\_BCD\_H(HOUR);

else

{

temp = Tf\_BCD\_H(HOUR&0x1f);

if(F\_AMPM&&temp<12) cHOUR = temp+12;

else cHOUR = temp;

}

cMINUTE = Tf\_BCD\_H(MINUTE);

}

### INT.h

#include "COMMON.h"

extern u16 t0ms;

extern u16 t1ms;

extern u8 TIM\_Modify;

void INT\_EX0\_Init();

void INT\_TIM0\_Init();

void INT\_TIM0\_Cmd();

void INT\_TIM1\_Init();

void INT\_TIM1\_Cmd();

### INT.c

#include <reg52.h>

#include "INT.h"

#include "SYS.h"

#include "LCDClock.h"

#include "IO.h"

u16 t0ms;

u16 t1ms;

u8 TIM\_Modify;

//--------------------EX0--------------------//

void INT\_EX0\_Init()

{

IE |= (1<<7)|(1<<0);

TCON &= 0<<0;

}

void ISR\_EX0() interrupt 0

{

u8 idata A2,A1,A0;

t1ms = 0;

if(!Key\_A2) A2 = 1;

else A2 = 0;

if(!Key\_A1) A1 = 1;

else A1 = 0;

if(!Key\_A0) A0 = 1;

else A0 = 0;

msg\_Keys = (A2<<2)+(A1<<1)+A0;

delayms(100);

}

//--------------------TIM0--------------------//

void INT\_TIM0\_Init()

{

IE |= (1<<7)|(1<<1);

TMOD |= 0x01;

t0ms = 0;

TIM\_Modify = 0x0A;

}

void INT\_TIM0\_Cmd()

{

t0ms = 0;

TH0 = 0xF8;

TL0 = 0xCD + TIM\_Modify;

TCON |= 1<<4;

}

void ISR\_TIM0() interrupt 1

{

TH0 = 0xF8;

TL0 = 0xCD + TIM\_Modify;

if(!F\_TMode)

{

if(t0ms>10)

{

if(tSECD100==99)

{

if(tSECOND == 59)

{

if(tMINUTE == 59)

{

if(tHOUR == 99)tHOUR = 0;

else tHOUR++;

tMINUTE = 0;

}

else tMINUTE++;

tSECOND = 0;

}

else tSECOND++;

tSECD100 = 0;

}

else tSECD100++;

t0ms = 0;

}

else t0ms++;

}

else if(F\_TMode==1)

{

if(t0ms>=1000)

{

if(!tSECOND)

{

if(!tMINUTE)

{

if(!tHOUR)

{

F\_Timeup = 1;

}

else tHOUR--;

tMINUTE = 59;

}

else tMINUTE--;

tSECOND = 59;

}

else tSECOND--;

t0ms = 0;

}

else t0ms++;

}

}

void INT\_TIM1\_Init()

{

IE |= (1<<7)|(1<<3);

TMOD |= 0x10;

t1ms = 0;

TIM\_Modify = 0x0A;

}

void INT\_TIM1\_Cmd()

{

t1ms = 0;

TH1 = 0xF8;

TL1 = 0xCD + TIM\_Modify;

TCON |= 1<<6;

TR1 = 1;

}

void ISR\_TIM1() interrupt 3

{

TH0 = 0xF8;

TL0 = 0xCD + TIM\_Modify;

t1ms++;

if(t1ms>=10000)

{

t1ms = 0;

F\_Return = 1;

}

}

### SYS.h

#include<reg52.h>

#include<common.h>

extern flag F\_Timeup;

extern u16 F\_Process;

extern flag F\_Alarm;

extern flag F\_TMode;

extern flag F\_Return;

extern flag F\_AlarmEN;

extern u16 sysTimer;

extern u8 NumBit;

extern msg msg\_Keys;

void SYS\_Main();

void SYS\_Init();

void SYS\_CLR();

void SYS\_Pointer();

void SYS\_Prep();

void SYS\_Menu();

void SYS\_Default();

void SYS\_CLRScrOnce();

void SYS\_SubM\_Fts();

void SYS\_SubM\_Set();

void SYS\_SSubM\_TIM();

void SYS\_SSubM\_rvTIM();

void SYS\_SSubM\_SetAlarm();

void SYS\_SV\_Alarm();

void SYS\_Success();

void SYS\_Timeup();

void SYS\_Info();

### SYS.c

#include<reg52.h>

#include "LCD12864.h"

#include "COMMON.h"

#include "LCDClock.h"

#include "DS1302.h"

#include "SYS.h"

#include "INT.h"

#include "IO.h"

msg msg\_Keys;

u16 F\_Process;

flag F\_Timeup;

flag F\_TMode;

flag F\_Return;

flag F\_AlarmEN;

flag idata F\_Selected;

flag idata F\_TIMState;

u8 idata temp;

u8 idata Ttemp;

void SYS\_Init()

{

AlarmBeep = 0;

LCD\_Init();

INT\_EX0\_Init();

INT\_TIM0\_Init();

INT\_TIM1\_Init();

INT\_TIM1\_Cmd();

Clock\_Alarm\_CLR();

F\_HourMode = 1;

F\_Process = 0;

msg\_Keys = 8;

F\_Selected = 1;

F\_Return = 0;

F\_AlarmEN = 0;

uYEAR = 0;

uMONTH = 0;

uDATE = 0;

uHOUR = 0;

uMINUTE = 0;

uSECOND = 0;

F\_Timeup = 0;

F\_TIMState = 0;

Clock\_Tim\_CLR();

}

void SYS\_Prep()

{

if(!msg\_Keys)

{

msg\_Keys = 8;

}

if(temp!=F\_Process)

{

//-------------------------菜单改变服务-----------------//

LCD\_CLR();

F\_Selected = 1;

Clock\_GetForSet();

if(Ttemp!=F\_TMode)

{

Clock\_Tim\_CLR();

F\_TIMState = 0;

}

}

if(F\_Timeup)

{

F\_Timeup = 0;

F\_Process = 1120;

}

if(F\_Return&&F\_Process)

{

F\_Return = 0;

F\_Process = 0;

}

temp = F\_Process;

Ttemp = F\_TMode;

}

void SYS\_Main()

{

while(1)

{

Clock\_Alarm\_Chk();

if(cHOUR==aHOUR&&cMINUTE==aMINUTE&&F\_AlarmEN&&F\_Process!=113)

{

F\_Process = 1120;

Clock\_Alarm\_CLR();

F\_AlarmEN = 0;

}

SYS\_Prep();

switch(F\_Process)

{

case 0:

{

SYS\_Default();

break;

}

case 1:

{

SYS\_Menu();

break;

}

case 11:

{

SYS\_SubM\_Fts();

break;

}

case 12:

{

SYS\_SubM\_Set();

break;

}

case 13:

{

SYS\_Init();

F\_Process = 121;

break;

}

case 14:

{

SYS\_Info();

break;

}

case 111:

{

SYS\_SSubM\_TIM();

break;

}

case 112:

{

SYS\_SSubM\_rvTIM();

break;

}

case 113:

{

SYS\_SSubM\_SetAlarm();

break;

}

case 1120:

{

SYS\_Timeup();

break;

}

case 121:

{

SYS\_Success();

delayms(1000);

F\_Process = 0;

break;

}

default: ;

}

}

}

//------------------------菜单指针-----------------------//

void SYS\_Pointer()

{

u8 idata sign[2];

u8 idata signCLR[2];

sign[0] = 0x20;

sign[1] = 0x10;

signCLR[0] = 0x20;

signCLR[1] = 0x20;

switch(F\_Selected)

{

case 1:

{

LCD\_NLet(0x80,sign,1);

LCD\_NLet(0x90,signCLR,1);

LCD\_NLet(0x88,signCLR,1);

LCD\_NLet(0x98,signCLR,1);

break;

}

case 2:

{

LCD\_NLet(0x90,sign,1);

LCD\_NLet(0x80,signCLR,1);

LCD\_NLet(0x88,signCLR,1);

LCD\_NLet(0x98,signCLR,1);

break;

}

case 3:

{

LCD\_NLet(0x88,sign,1);

LCD\_NLet(0x80,signCLR,1);

LCD\_NLet(0x90,signCLR,1);

LCD\_NLet(0x98,signCLR,1);

break;

}

case 4:

{

LCD\_NLet(0x98,sign,1);

LCD\_NLet(0x80,signCLR,1);

LCD\_NLet(0x90,signCLR,1);

LCD\_NLet(0x88,signCLR,1);

break;

}

default: ;

}

}

//------------------------待机界面-----------------------//

void SYS\_Default()

{

u8 idata AlarmS[2];

u8 idata AlarmNS[2];

AlarmS[0] = 0x0E;

AlarmS[1] = 0x20;

AlarmNS[0] = 0x20;

AlarmNS[1] = 0x20;

//msg\_Keys = 8;

AlarmBeep = 0;

Clock\_UpdateTime();

Clock\_ShowTime();

Clock\_ShowDate();

if(F\_AlarmEN) LCD\_NLet(0x87,AlarmS,1);

else LCD\_NLet(0x87,AlarmNS,1);

switch(msg\_Keys)

{

case 1:

{

F\_Process = 1;

msg\_Keys = 8;

delayms(100);

break;

}

default:;

}

}

//------------------------主菜单-----------------------//

void SYS\_Menu()

{

u8 idata txt1[]="常用功能";

u8 idata txt2[]="调整时间日期";

u8 idata txt3[]="复位计时";

u8 idata txt4[]="程序信息";

LCD\_NChar(0x81,txt1,8);

LCD\_NChar(0x91,txt2,12);

LCD\_NChar(0x89,txt3,8);

LCD\_NChar(0x99,txt4,8);

SYS\_Pointer();

switch(msg\_Keys)

{

case 1:

{

switch(F\_Selected)

{

case 1:

{

F\_Process = 11;

break;

}

case 2:

{

F\_Process = 12;

break;

}

case 3:

{

F\_Process = 13;

break;

}

case 4:

{

F\_Process = 14;

break;

}

}

}

case 3:

{

if(F\_Selected == 4)F\_Selected = 1;

else F\_Selected++;

msg\_Keys = 8;

break;

}

case 2:

{

if(F\_Selected == 1)F\_Selected = 4;

else F\_Selected--;

msg\_Keys = 8;

break;

}

case 6:

{

F\_Process = 0;

delayms(100);

msg\_Keys = 8;

break;

}

case 7:

{

F\_Process = 0;

delayms(100);

msg\_Keys = 8;

break;

}

default:;

}

}

//------------------------功能菜单-----------------------//

void SYS\_SubM\_Fts()

{

u8 idata txt1[]="秒表";

u8 idata txt2[]="定时器";

u8 idata txt3[]="设置闹钟";

LCD\_NChar(0x81,txt1,4);

LCD\_NChar(0x91,txt2,6);

LCD\_NChar(0x89,txt3,8);

SYS\_Pointer();

switch(msg\_Keys)

{

case 1:

{

switch(F\_Selected)

{

case 1:

{

F\_Process = 111;

F\_TMode = 0;

msg\_Keys = 8;

break;

}

case 2:

{

F\_Process = 112;

F\_TMode = 1;

msg\_Keys = 8;

break;

}

case 3:

{

F\_Process = 113;

msg\_Keys = 8;

break;

}

case 4: ;

}

}

case 3:

{

if(F\_Selected == 4)F\_Selected = 1;

else F\_Selected++;

msg\_Keys = 8;

break;

}

case 2:

{

if(F\_Selected == 1)F\_Selected = 4;

else F\_Selected--;

msg\_Keys = 8;

break;

}

case 6:

{

F\_Process = 1;

delayms(100);

msg\_Keys = 8;

break;

}

case 7:

{

F\_Process = 0;

delayms(100);

msg\_Keys = 8;

break;

}

default:;

}

}

//-----------------------时间设置界面-------------------------//

void SYS\_SubM\_Set()

{

u8 idata PtLeft[2] = {0x1E,0x20};

u8 idata PtRight[2] = {0x20,0x1E};

u8 idata PtCLR[2] = {0x20,0x20};

//按键分配

switch(msg\_Keys)

{

case 1:

{

if(F\_HourMode&&F\_AMPM&&uHOUR!=12) uHOUR+=12;

Clock\_SetTime(uYEAR,uMONTH,uDATE,uHOUR,uMINUTE,uSECOND,F\_HourMode);

msg\_Keys = 8;

F\_Process = 121;

break;

}

case 2:

{

if(F\_Selected == 1)F\_Selected = 7;

else F\_Selected--;

msg\_Keys = 8;

break;

}

case 3:

{

if(F\_Selected == 7)F\_Selected = 1;

else F\_Selected++;

msg\_Keys = 8;

break;

}

case 4:

{

//------------------------------------------------------------

switch(F\_Selected)

{

case 1:

{

if(uMONTH == 1)uMONTH = 12;

else if(uMONTH == 0)uMONTH = 9;

else uMONTH --;

break;

}

case 2:

{

if(uDATE == 1)

{

if(uMONTH == 2)

{

if(!(uYEAR%4))uDATE = 29;

else uDATE = 28;

}

else if(uMONTH == 4||uMONTH == 6||uMONTH == 9||uMONTH == 11)uDATE = 30;

else uDATE = 31;

}

else if(uDATE == 0)uDATE = 9;

else uDATE--;

break;

}

case 3:

{

if(!uYEAR)uYEAR = 99;

else uYEAR --;

break;

}

case 4:

{

if(!F\_HourMode&&!uHOUR) uHOUR = 23;

else if(F\_HourMode&&!uHOUR&&!F\_AMPM) uHOUR = 11;

else if(F\_HourMode&&uHOUR == 1&&F\_AMPM) uHOUR = 12;

else if(F\_HourMode&&!uHOUR&&F\_AMPM) uHOUR = 1;

else uHOUR --;

break;

}

case 5:

{

if(!uMINUTE)uMINUTE = 59;

else uMINUTE --;

break;

}

case 6:

{

if(!uSECOND)uSECOND = 59;

else uSECOND --;

break;

}

case 7:

{

if(F\_HourMode&&F\_AMPM) F\_AMPM = 0;

else if(F\_HourMode &&(!F\_AMPM)) F\_HourMode = 0;

else if(!F\_HourMode)

{

F\_HourMode = 1;

F\_AMPM = 1;

}

else

{

F\_HourMode = 0;

}

break;

}

default:;

}

//-------------------------------------------------------------

msg\_Keys = 8;

break;

}

case 5:

{

//------------------------------------------------------------

switch(F\_Selected)

{

case 1:

{

if(uMONTH == 12)uMONTH = 1;

else uMONTH++;

break;

}

case 2:

{

if(uMONTH == 2)

{

if(!(uYEAR%4))

{

if(uDATE == 29)uDATE = 1;

else uDATE++;

}

else

{

if(uDATE == 28)uDATE = 1;

else if(uDATE == 29)uDATE = 1;

else uDATE++;

}

}

else if(uMONTH==4||uMONTH==6||uMONTH==9||uMONTH==11)

{

if(uDATE == 30) uDATE = 1;

else uDATE++;

}

else

{

if(uDATE == 31)uDATE = 1;

else uDATE++;

}

break;

}

case 3:

{

if(uYEAR == 99)uYEAR = 0;

else uYEAR++;

break;

}

case 4:

{

if(!F\_HourMode&&uHOUR==23)uHOUR = 0;

else if(F\_HourMode&&!F\_AMPM&&uHOUR==11)uHOUR=0;

else if(F\_HourMode&&F\_AMPM&&uHOUR==12)uHOUR=1;

else if(F\_HourMode&&!F\_AMPM&&uHOUR==12)uHOUR=11;

else uHOUR++;

break;

}

case 5:

{

if(uMINUTE==59)uMINUTE=0;

else uMINUTE++;

break;

}

case 6:

{

if(uSECOND==59)uSECOND=0;

else uSECOND++;

break;

}

case 7:

{

if(!F\_HourMode)

{

F\_HourMode = 1;

F\_AMPM = 0;

}

else if(F\_HourMode&&!F\_AMPM)

{

F\_AMPM = 1;

}

else F\_HourMode = 0;

break;

}

default:;

}

//-------------------------------------------------------------

msg\_Keys = 8;

break;

}

case 6:

{

F\_Process = 1;

delayms(100);

msg\_Keys = 8;

break;

}

case 7:

{

F\_Process = 0;

delayms(100);

msg\_Keys = 8;

break;

}

default:;

}

//指针显示

switch(F\_Selected)

{

case 1:

{

LCD\_NLet(0x90,PtRight,1);

LCD\_NLet(0x92,PtCLR,1);

LCD\_NLet(0x94,PtCLR,1);

LCD\_NLet(0x99,PtCLR,1);

LCD\_NLet(0x9B,PtCLR,1);

LCD\_NLet(0x9C,PtCLR,1);

LCD\_NLet(0x9E,PtCLR,1);

break;

}

case 2:

{

LCD\_NLet(0x90,PtCLR,1);

LCD\_NLet(0x92,PtLeft,1);

LCD\_NLet(0x94,PtCLR,1);

LCD\_NLet(0x99,PtCLR,1);

LCD\_NLet(0x9B,PtCLR,1);

LCD\_NLet(0x9C,PtCLR,1);

LCD\_NLet(0x9E,PtCLR,1);

break;

}

case 3:

{

LCD\_NLet(0x90,PtCLR,1);

LCD\_NLet(0x92,PtCLR,1);

LCD\_NLet(0x94,PtRight,1);

LCD\_NLet(0x99,PtCLR,1);

LCD\_NLet(0x9B,PtCLR,1);

LCD\_NLet(0x9C,PtCLR,1);

LCD\_NLet(0x9E,PtCLR,1);

break;

}

case 4:

{

LCD\_NLet(0x90,PtCLR,1);

LCD\_NLet(0x92,PtCLR,1);

LCD\_NLet(0x94,PtCLR,1);

LCD\_NLet(0x99,PtRight,1);

LCD\_NLet(0x9B,PtCLR,1);

LCD\_NLet(0x9C,PtCLR,1);

LCD\_NLet(0x9E,PtCLR,1);

break;

}

case 5:

{

LCD\_NLet(0x90,PtCLR,1);

LCD\_NLet(0x92,PtCLR,1);

LCD\_NLet(0x94,PtCLR,1);

LCD\_NLet(0x99,PtCLR,1);

LCD\_NLet(0x9B,PtLeft,1);

LCD\_NLet(0x9C,PtCLR,1);

LCD\_NLet(0x9E,PtCLR,1);

break;

}

case 6:

{

LCD\_NLet(0x90,PtCLR,1);

LCD\_NLet(0x92,PtCLR,1);

LCD\_NLet(0x94,PtCLR,1);

LCD\_NLet(0x99,PtCLR,1);

LCD\_NLet(0x9B,PtCLR,1);

LCD\_NLet(0x9C,PtRight,1);

LCD\_NLet(0x9E,PtCLR,1);

break;

}

case 7:

{

LCD\_NLet(0x90,PtCLR,1);

LCD\_NLet(0x92,PtCLR,1);

LCD\_NLet(0x94,PtCLR,1);

LCD\_NLet(0x99,PtCLR,1);

LCD\_NLet(0x9B,PtCLR,1);

LCD\_NLet(0x9C,PtCLR,1);

LCD\_NLet(0x9E,PtRight,1);

break;

}

default:;

}

Clock\_ShowSetT();

}

//------------------------成功消息提示---------------------//

void SYS\_Success()

{

u8 idata txt1[]="操作成功";

LCD\_NChar(0x92,txt1,8);

}

//------------------------秒表------------------------//

void SYS\_SSubM\_TIM()

{

u8 idata txt1[]="秒表";

LCD\_NChar(0x80,txt1,4);

switch(msg\_Keys)

{

case 1:

{

if(!F\_TIMState)

{

INT\_TIM0\_Cmd();

TR0 = 1;

F\_TIMState = 2;

}

else if(F\_TIMState == 1)

{

F\_TIMState = 2;

}

else if(F\_TIMState == 2)

{

F\_TIMState = 1;

}

delayms(100);

msg\_Keys = 8;

break;

}

case 2:

{

if(F\_TIMState == 1)

{

Clock\_Tim\_CLR();

}

msg\_Keys = 8;

break;

}

case 6:

{

F\_Process = 11;

delayms(100);

msg\_Keys = 8;

break;

}

case 7:

{

F\_Process = 0;

delayms(100);

msg\_Keys = 8;

break;

}

default:;

}

switch(F\_TIMState)

{

case 1:

{

TR0 = 0;

break;

}

case 2:

{

TR0 = 1;

break;

}

default:;

}

Clock\_Tim\_Disp();

}

void SYS\_SSubM\_rvTIM()

{

u8 idata txt1[]="定时器";

u8 idata PtLeft[2] = {0x1E,0x20};

u8 idata PtRight[2] = {0x20,0x1E};

u8 idata PtCLR[2] = {0x20,0x20};

LCD\_NChar(0x80,txt1,6);

switch(msg\_Keys)

{

case 1:

{

if(!F\_TIMState)

{

INT\_TIM0\_Cmd();

TR0 = 1;

F\_TIMState = 2;

//开始走时

LCD\_NLet(0x99,PtCLR,1);

LCD\_NLet(0x9B,PtCLR,1);

LCD\_NLet(0x9C,PtCLR,1);

}

else if(F\_TIMState == 1)

{

F\_TIMState = 2; //继续

}

else if(F\_TIMState == 2)

{

F\_TIMState = 1; //停止

}

delayms(100);

msg\_Keys = 8;

break;

}

case 2:

{

if(F\_TIMState == 1)

{

F\_TIMState = 0;

}

else if(!F\_TIMState)

{

if(F\_Selected==1) F\_Selected = 3;

else F\_Selected --;

}

msg\_Keys = 8;

break;

}

case 3:

{

if(F\_Selected==3) F\_Selected = 1;

else F\_Selected ++;

msg\_Keys = 8;

break;

}

case 4:

{

switch(F\_Selected)

{

case 1:

{

if(tHOUR == 0) tHOUR = 99;

else tHOUR --;

break;

}

case 2:

{

if(tMINUTE == 0) tMINUTE = 59;

else tMINUTE --;

break;

}

case 3:

{

if(tSECOND == 0) tSECOND = 59;

else tSECOND --;

break;

}

}

msg\_Keys = 8;

break;

}

case 5:

{

switch(F\_Selected)

{

case 1:

{

if(tHOUR == 99) tHOUR = 0;

else tHOUR ++;

break;

}

case 2:

{

if(tMINUTE == 59) tMINUTE = 0;

else tMINUTE ++;

break;

}

case 3:

{

if(tSECOND == 59) tSECOND = 0;

else tSECOND ++;

break;

}

}

msg\_Keys = 8;

break;

}

case 6:

{

F\_Process = 11;

delayms(100);

msg\_Keys = 8;

break;

}

case 7:

{

F\_Process = 0;

delayms(100);

msg\_Keys = 8;

break;

}

default:;

}

switch(F\_TIMState)

{

case 0:

{

//显示设置指针

switch(F\_Selected)

{

case 1:

{

LCD\_NLet(0x99,PtRight,1);

LCD\_NLet(0x9B,PtCLR,1);

LCD\_NLet(0x9C,PtCLR,1);

break;

}

case 2:

{

LCD\_NLet(0x99,PtCLR,1);

LCD\_NLet(0x9B,PtLeft,1);

LCD\_NLet(0x9C,PtCLR,1);

break;

}

case 3:

{

LCD\_NLet(0x99,PtCLR,1);

LCD\_NLet(0x9B,PtCLR,1);

LCD\_NLet(0x9C,PtRight,1);

break;

}

default:;

}

TR0 = 0;

break;

}

case 1:

{

TR0 = 0;

break;

}

case 2:

{

TR0 = 1;

break;

}

default:;

}

Clock\_rvTim\_Disp();

}

void SYS\_Timeup()

{

u8 idata txt1[]="时间到";

LCD\_NChar(0x8A,txt1,6);

TR0 = 0;

F\_TIMState = 0;

Clock\_Tim\_CLR();

AlarmBeep = 1;

if(msg\_Keys>=1&&msg\_Keys<=7)

{

F\_Process = 0;

}

}

void SYS\_SSubM\_SetAlarm()

{

u8 idata title[] = "闹钟设置";

u8 idata PtLeft[2] = {0x1E,0x20};

u8 idata PtRight[2] = {0x20,0x1E};

u8 idata PtCLR[2] = {0x20,0x20};

u8 idata sign[2];

u8 idata signCLR[2];

sign[0] = 0x20;

sign[1] = 0x10;

signCLR[0] = 0x20;

signCLR[1] = 0x20;

LCD\_NChar(0x80,title,8);

switch(msg\_Keys)

{

case 1:

{

F\_Process = 0;

msg\_Keys = 8;

break;

}

case 2:

{

if(F\_Selected==1) F\_Selected = 3;

else F\_Selected --;

msg\_Keys = 8;

break;

}

case 3:

{

if(F\_Selected==3) F\_Selected = 1;

else F\_Selected ++;

msg\_Keys = 8;

break;

}

case 4:

{

switch(F\_Selected)

{

case 1:

{

if(aHOUR == 0) aHOUR = 23;

else aHOUR --;

break;

}

case 2:

{

if(aMINUTE == 0) aMINUTE = 59;

else aMINUTE --;

break;

}

case 3:

{

F\_AlarmEN = ~F\_AlarmEN;

}

}

msg\_Keys = 8;

break;

}

case 5:

{

switch(F\_Selected)

{

case 1:

{

if(aHOUR == 23) aHOUR = 0;

else aHOUR ++;

break;

}

case 2:

{

if(aMINUTE == 59) aMINUTE = 0;

else aMINUTE ++;

break;

}

case 3:

{

F\_AlarmEN = ~F\_AlarmEN;

}

}

msg\_Keys = 8;

break;

}

case 6:

{

F\_Process = 11;

delayms(100);

msg\_Keys = 8;

break;

}

case 7:

{

F\_Process = 0;

Clock\_Alarm\_CLR();

delayms(100);

msg\_Keys = 8;

break;

}

default:;

}

switch(F\_Selected)

{

case 1:

{

LCD\_NLet(0x9A,PtRight,1);

LCD\_NLet(0x9C,PtCLR,1);

LCD\_NLet(0x90,PtCLR,1);

break;

}

case 2:

{

LCD\_NLet(0x9A,PtCLR,1);

LCD\_NLet(0x9C,PtLeft,1);

LCD\_NLet(0x90,PtCLR,1);

break;

}

case 3:

{

LCD\_NLet(0x9A,PtCLR,1);

LCD\_NLet(0x9C,PtCLR,1);

LCD\_NLet(0x90,sign,1);

break;

}

default:;

}

Clock\_Alarm\_Disp();

}

void SYS\_Info()

{

u8 idata txt1[] = "版本信息";

u8 idata txt2[4];

u8 idata txt3[8]={0x50,0x61,0x6E,0x65,0x76,0x69,0x6C,0x61};

txt2[0] = 0x76;

txt2[1] = 0x30;

txt2[2] = 0x2E;

txt2[3] = 0x39;

LCD\_NChar(0x80,txt1,8);

LCD\_NLet(0x8A,txt2,2);

LCD\_NLet(0x9C,txt3,4);

switch(msg\_Keys)

{

case 1:

{

F\_Process = 0;

msg\_Keys = 8;

break;

}

case 6:

{

F\_Process = 1;

delayms(100);

msg\_Keys = 8;

break;

}

case 7:

{

F\_Process = 0;

Clock\_Alarm\_CLR();

delayms(100);

msg\_Keys = 8;

break;

}

default:;

}

}