数字钟项目

硬件总体设计说明书

编制单位：侏罗纪工作室

作　　者

发布日期：2011-1-22

审 核 人：

批 准 人：

目 录

1. 引言

1.1. 编写目的：

20 世纪末，电子技术获得了飞速的发展，在其推动下，现代电子产品几乎渗透了社会的各个领域，有力地推动了社会生产力的发展和社会信息化程度的提高，同时也使现代电子产品性能进一步提高，产品更新换代的节奏也越来越快。 时间对人们来说总是那么宝贵，工作的忙碌性和繁杂性容易使人忘记当前的时间。忘记了要做的事情，当事情不是很重要的时候，这种遗忘无伤大雅。但是，一旦重要事情，一时的耽误可能酿成大祸。例如，许多火灾都是由于人们一时忘记了关闭煤气或是忘记充电时间等造成的。而钟表的数字化给人们生产生活带来了极大的方便。数字钟是通过数字电路实现时,分,秒数字显示的计时装置,广泛用于个人家庭、车站、码头办公室等公共场所,成为人们日常生活中不可少的必需品。由于数字集成电路的发展和石英晶体振荡器的广泛应用,使得数字钟的精度,远远超过老式钟表,钟表的数字化给人们生产生活带来了极大的方便，而且大大地扩展了钟表原先的报时功能，诸如定时自动报警、按时自动打铃、时间程序自动控制、定时广播、自动起闭路灯、定时开关烤箱、通断动力设备、甚至各种定时电气的自动启用等。所有这些，都是以钟表数字化为基础的。因此，研究数字钟及扩大其应用，有着非常现实的意义。

1.2. 背景

单片机自 20 世纪 70 年代问世以来，以其极高的性能价格比，受到人们的重视和关注，应用很广、发展很快。而 51 单片机是各单片机中最为典型和最有代表性的一种。

本设计以 AT89C51 芯片为核心，辅以必要的外围电路，设计了一个结构简单，功能齐全的电子时钟，它由 5V 直流电源供电。在硬件方面，除了 CPU 外，使用八个七段 LED 数码管来进行显示，LED 采用的是动态扫描显示，使用 74LS245 芯片进行驱动。通过 LED 能够较为准确地显示时、分、秒。四个简单的按键实现对时间的调整。软件方面采用 C 语言编程。整个电子钟系统能完成

时间的显示、调时、校时和三组定时闹钟的功能。

选用单片机最小系统应用程序,添加比较程序、时间调整程序及蜂鸣程序，通过时间比较程序触发蜂鸣，实现闹钟功能，完成设计所需求的软件环境。介绍并使用 Keil 单片机模拟调试软件，测试程序的可行性并用 Proteus 进行仿真，并且利用 Protel 软件来绘制 PCB 板。本设计应解决的主要问题有两大方面，即硬件电路设计和软件设计两大方面。其中硬件电路部分又可分为四个模块：键盘模块、显示模块、计时模块和发声模块。硬件电路部分致力于低成本、低功耗和易实现性。软件部分则应做到代码的精简、准确、易读懂。最后通过硬软件的结合实现数字钟的精确计时、校时、三组闹钟设置和定时报警功能。

1.3. 定义

单片机：单片机是一种集成在电路芯片，是采用超大规模集成电路技术把具有数据处理能力的中央处理器 CPU 随机存储器 RAM、只读存储器 ROM、多种 I/O 口和中断系统、定时器/计时器等功能（可能还包括显示驱动电路、脉宽调制电路、模拟多路转换器、A/D 转换器等电路）集成到一块硅片上构成的一个小而完善的计算机系统。

定时器：51 单片机内有两个 16 位的可编程的定时器/计数器，即定时器 T0 和定时器 T1。

中断：51 单片机内有 5 个中断源，这里运用的是定时器中断。CPU 一旦设置开启定时功能后，定时器便在晶振的作用下开始自动计时，当定时器的计数器计满后，会产生中断。

闹钟：预置一个定时时间，当走时到定时的时间后，闹铃响铃。

LED：8 位共阴极数码管。

1.4. 参考资料

《单片机原理及接口技术》 第三版 北京航空航天大学出版社 李朝青

《新概念 51 单片机 C 语言教程》 电子工业出版社 郭天翔

2. 总体设计

2.1 开发与运行环境

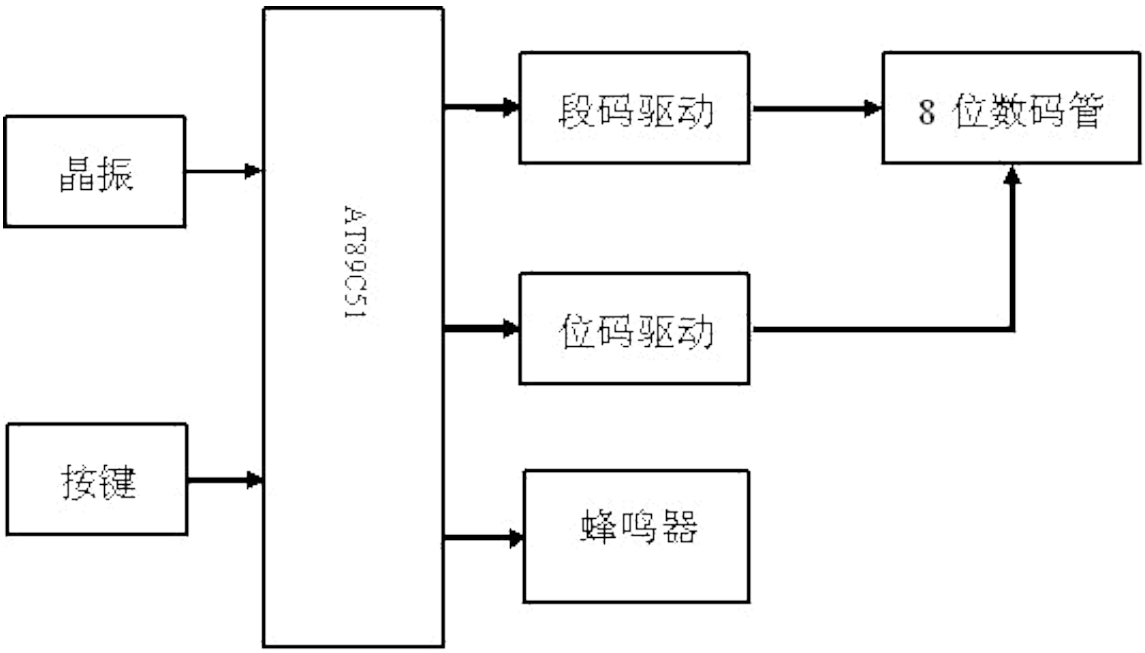
在硬件方面，除了 CPU 外，使用八个七段 LED 数码管来进行显示，LED 采用的是动态扫描显示，使用 74LS245 芯片进行驱动。通过 LED 能够较为准确地显示时、分、秒。四个简单的按键实现对时间的调整。软件方面采用 C 语言编程。使用 Keil 单片机模拟调试软件，测试程序的可行性并用 Proteus 进行仿真，并且利用 Protel 软件来绘制 PCB 板。

2.2 硬件功能描述

硬件部分设置了的三个按键 S1、S2、S3、S4。当按键 S1 第一次按下时，停止计时进入闹钟 1 的秒设置，当按键 S1 第二、第三次按下时，分别进入闹钟 1 的分设置和时设置，当按 S1 第四　、第五、第六次按下时分别进入闹钟 2 的秒、分、时设置，当按 S1 第七　、第八、第九次按下时分别进入闹钟 3 的秒、分、时设置，当按 S1 第十 、第二一、第十二次按下时分别进入时间的秒、分、时设置，在 S1 按下的各阶段，可用按键 S2、S3 进行时间和闹铃时间的时、分、秒进行加减设置；当按键 S1 第十三次按下时恢复到时间显示功能。当显示的时间和定时设置的时间一致时，蜂鸣器发出等时间断蜂鸣声，闹铃时间设置为 60 秒。在各个闹钟设置阶段，如果有 S4 按下，则相应闹钟功能关闭或开启；如在闹铃时有 S4 按下则提前停止闹铃。

另外，闹铃电路有音乐闹钟的扩展的功能(可以将蜂鸣器换成扬声器再加一段音乐程序或利用音乐芯片即可实现)。因时间有限，扩展功能还未能及时实现，比如音乐闹铃。

2.3 硬件结构



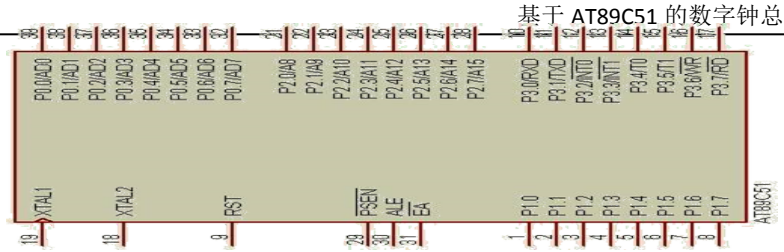
3. 硬件模块设计

3.1. 描述

3.1.1. AT89C51 单片机简介

AT89C51 是一款单片封装的微控制器，适合于许多要求高集成度、低成本的场合。可以满足多方面的性能要求。AT89C51 采用了高性能的处理器结构，指令执行时间只需 2 到 4 个时钟周期。6 倍于标准 51 单片机器件。AT89C51 集成了许多系统级的功能，这样可大大减少元件的数目和电路板面积并降低系统的成本。

AT89C51 单片机内部主要有以下部件：8031CPU、振荡电路、总线控制部件、中断控制部件、片内 Flash 存储器、并行 I/O 接口、定时器和串行 I/O 接口。



3.1.2. 键盘电路的设计

方案一：4×4 矩阵式键盘。如果选择此方案，那么在修改时钟或设置闹铃时间时就可以直接从键盘输入，方便、快捷，但程序较为复杂。

方案二：独立式按键。如果设置过多按键，将会占用较多 I/O 口，而且会给布线带来不便，因此，此方案适用于按键较少的情况。如果选择此方案，由于按键较少，在修改时间或设置闹铃时间时就不能直接输入，只能通过加或减完成，稍为麻烦一些，但其程序简单。

由于并不需要经常修改时间和设置闹铃时间，而且方案二的程序简单，按键少、成本低，因此，选择方案二。

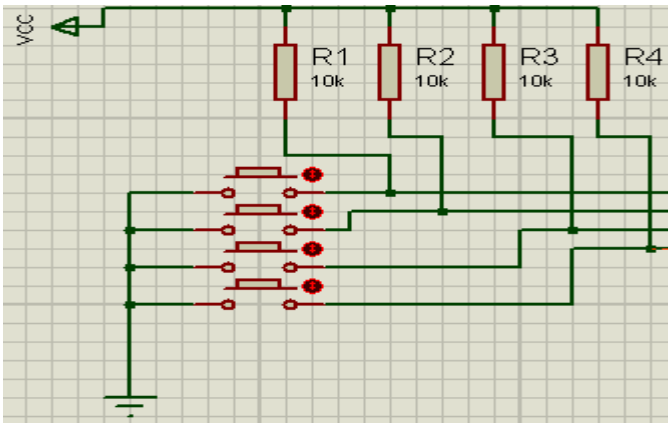
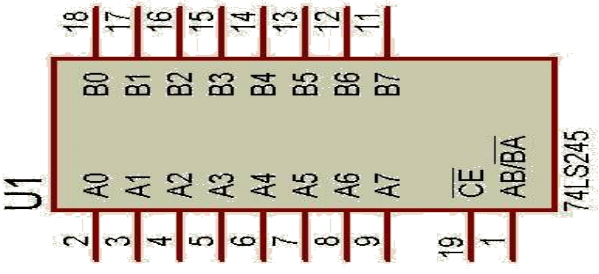


图 3-1 独立按键

图 3-2 键盘输入电路

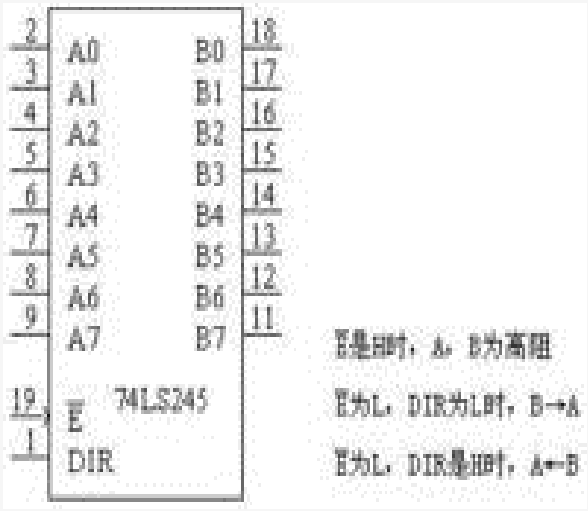
3.1.3. 段码驱动电路

由于通过数码管公共及的电流较大且避免过多地使用分立元件，采用了一



片 74LS245 来驱动段码，用 P2 口作位码驱动。

图 3-3 段码驱动器 74LS245



74LS245 是我们常用的芯片，用来驱动 led 或者其他的设备，它是 8 路同相三态双向总线收发器，可双向传输数据。

*74LS245 还具有双向三态功能，既可以输出，也可以输入数据。

*当 8051 单片机的 P0 口总线负载达到或超过 P0 最大负载能力时，必须接入 74LS245 等总线驱动器。

*当片选端/CE 低电平有效时，DIR=“0”，信号由 B 向 A 传输；（接收）

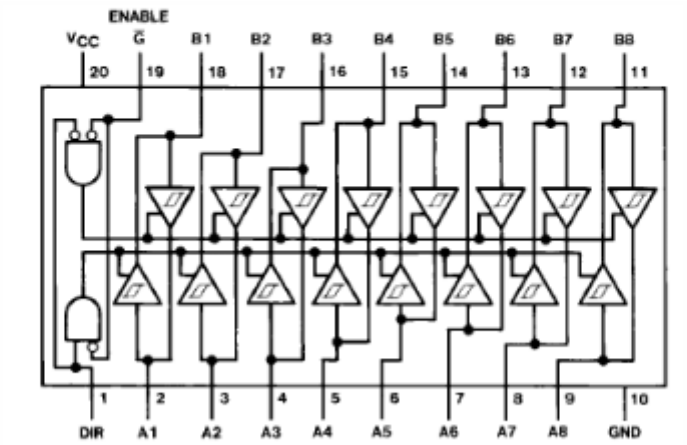
*DIR=“1”，信号由 A 向 B 传输；（发送）当/CE 为高电平时，A、B 均为高阻态。

由于 P2 口始终输出地址的高 8 位，接口时 74LS245 的三态控制端/1G 和/2G 接地，P2 口与驱动器输入线对应相连。P0 口与 74LS245 输入端相连,/E 端接地，保证数据现畅通。8051 的/RD 和/PSEN 相与后接 DIR，使得/RD 且/PSEN 有效时，74LS245 输入（P0.i←Di），其它时间处于输出（P0.i→Di）。

引出端符号:

A	A 总线端
B	B 总线端
/G	三态允许端(低电平有效)
DIR	方向控制端

逻辑图:



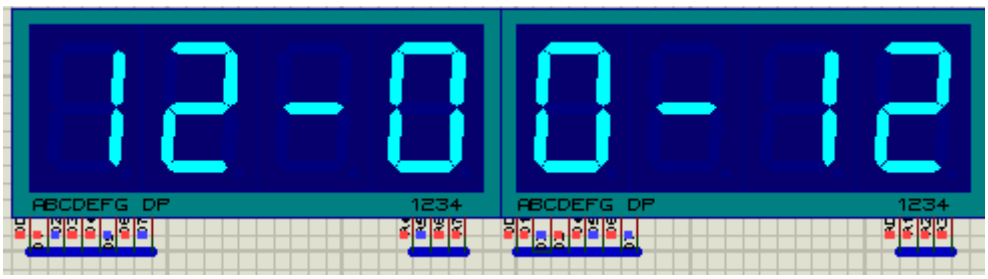
双列直插封装

3.1.4. 显示器的选择

方案一：液晶显示器。如果选择此方案，将会降低系统的功耗，这样就可以用电池供电，便于携带。但液晶显示器的驱动电路复杂，使用起来有一定的难度。

方案二：用数码管作为显示器。数码管的驱动电路简单，使用方便，如果选择了此方案，那么在夜间看时间的时候就不需要有光源，非常方便。其缺点是功耗较大。

由于数码管使用起来较为方便，在夜间看时间也很方便，因此我选择了方案二。



3.1.5. 蜂鸣器驱动电路

发音部分是通过三极管放大驱动蜂鸣器工作，再通过软件这时产生等时时间方波驱动蜂鸣器发出间断嘀声，这样就可以省去硬件振荡电路，降低成本。

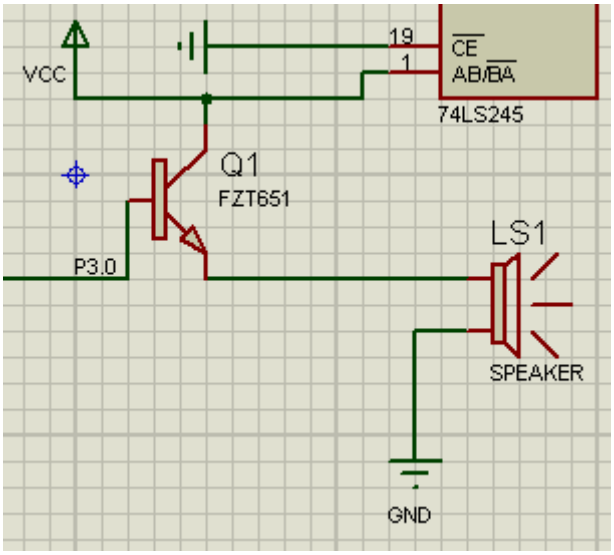


图 3-4 蜂鸣器驱动电路

3.2. 功能

1. 键盘模块

在此次设计时用了四个按键，分别为 S1, S2, S3, S4. 当按键 S1 第一次按下时，停止计时进入闹钟 1 的秒设置，当按键 S1 第二、第三次按下时，分别进入闹钟 1 的分设置和时设置，当按 S1 第四 、第五、第六次按下时分别进入闹钟 2 的秒、分、时设置，当按 S1 第七 、第八、第九次按下时分别进入闹钟 3 的秒、分、时设置，当按 S1 第十 、第二一、第十二次按下时分别进入时间的秒、分、时设置，在 S1 按下的各阶段，可用按键 S2、S3 进行时间和闹铃时间的时、分、秒进行加减设置；当按键 S1 第十三次按下时恢复到时间显示功能。当显示的时间和定时设置的时间一致时，蜂鸣器发出等时间断蜂鸣声，闹铃时间设置为 60 秒。在各个闹钟设置阶段，如果有 S4 按下，则相应闹钟功能关闭或开启；如在闹铃时有 S4 按下则提前停止闹铃。

2. 段码驱动电路

74LS245 是我们常用的芯片，用来驱动 led 或者其他的设备，它是 8 路同相三态双向总线收发器，可双向传输数据。

*74LS245 还具有双向三态功能，既可以输出，也可以输入数据。

*当 8051 单片机的 P0 口总线负载达到或超过 P0 最大负载能力时，必须接入 74LS245 等总线驱动器。

*当片选端/CE 低电平有效时，DIR=“0”，信号由 B 向 A 传输；（接收）

*DIR=“1”，信号由 A 向 B 传输；（发送）当/CE 为高电平时，A、B 均为高阻态。

在设计硬件电路时，74LS245 的 DIR=“1”，信号由 A 向 B 传输。由 P0 口送信号至 74LS245，再由其驱动数码管的段选，实现数字的显示。

3. 蜂鸣器电路

发音部分是通过三极管放大驱动蜂鸣器工作，再通过软件这时产生等时时间方波驱动蜂鸣器发出间断嘀声，这样就可以省去硬件振荡电路，降低成本。

4. 嵌入式软件设计

4.1. 流程逻辑

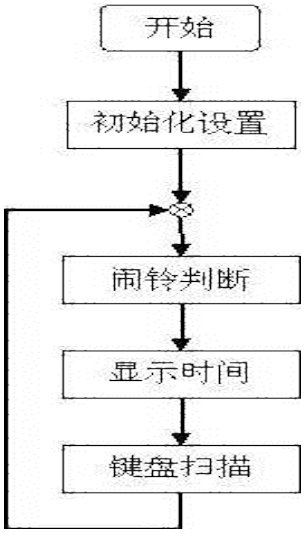


图 4-1 主程序流程图

4.2. 算法

```
//*****************************头文件*********************************
#include<reg51.h>
#include<intrins.h>
//*****************************宏定义*********************************
#define uchar unsigned char
#define uint    unsigned int
//*****************************位声明*********************************
sbit key1=P1^0;
sbit key2=P1^1;
sbit key3=P1^2;
sbit key4=P1^3;
sbit fmq=P3^0;
//************************数码管显示的数值*************************

uchar code table[]={0x3f,0x06,0x5b,0x4f,0x66,0x6d,
                    0x7d,0x07,0x7f,0x6f,0x40,0x00};
//*****************************函数声明*******************************
void jia();
void jian();
//********************数组定义，数组内含有 8 个数值*****************
uchar table1[8],table2[8],table3[8],table4[8];
//*************************时间显示初始值**************************
uchar shi=12,fen=0,miao=0;
//*************************定义全局变量*****************************
uchar shi1,fen1,miao1,shi2,fen2,miao2,shi3,fen3,miao3;
uchar shi4,fen4,miao4;

uchar flag, flag1, wss, cnt, cnt1, alm1, alm2, alm3;
```

```
//      1 秒     等时    位闪  次数  校时      闹 1    闹 2      闹 3
uint flag2;
//     蜂鸣
//**********************延时函数，用于动态扫描数码管*****************
void delay(uchar i)
{    uchar x,y;
     for(x=i;x>0;x--)
        for(y=120;y>0;y--);
  }
//*******************************初始化函数************************
void init()
{    TMOD=0x01;              //工作方式 1
     TH0=0x3c;               //定时时间为：50ms    (65536-50000)/256
     TL0=0x0b0;              //(65536-50000)%256
     ET0=1;                  //打开定时器
     EA=1;                   //开总中断
     TR0=1;                  //启动定时器
  }
   //********************显示子函数，用于显示时间数值*****************
void display()
{    uchar i,j;
     if(cnt!=10||wss==0)
        { table1[0]=miao%10;            //分离秒的个位与十位
           table1[1]=miao/10;
          }
      else
      { table1[0]=table1[1]=11;}
      if(cnt!=11||wss==0)
      { table1[3]=fen%10;               //分离分的个位与十位
           table1[4]=fen/10;
```

```
        }
    else
    { table1[3]=table1[4]=11;}
    if(cnt!=12||wss==0)
    { table1[6]=shi%10;              //分离时的个位与十位
       table1[7]=shi/10;
        }
    else
    { table1[6]=table1[7]=11;}
   table1[2]=table1[5]=10;
   j=0xfb;
   for(i=0;i<=7;i++)                //从秒到时的扫描
     {   P2=j;
         P0=table[table1[i]];   //显示数值
         delay(10);
         j=_cror_(j,1);            //循环右移
        }
  }
  //*******************显示子函数，用于显示定时 1 时间
*****************
void display1()
{   uchar i,j;
    if(alm1==0)
      { if(cnt!=1||wss==0)
          { table2[0]=miao1%10;            //以下含义同上
             table2[1]=miao1/10;
           }
        else
          { table2[0]=table2[1]=11;}
        if(cnt!=2||wss==0)
```

```
            { table2[3]=fen1%10;
                table2[4]=fen1/10;
              }
          else
            { table2[3]=table2[4]=11;}
          if(cnt!=3||wss==0)
            { table2[6]=shi1%10;
                table2[7]=shi1/10;
              }
          else
            { table2[6]=table2[7]=11;}
          }
      else
          table2[0]=table2[1]=table2[3]=table2[4]=table2[6]=table2[7]=10;
     table2[2]= table2[5]=10;
     j=0xfb;
     for(i=0;i<=7;i++)
        {   P2=j;
            P0=table[table2[i]];
           delay(10);
            j=_cror_(j,1);
           }
  }
//********************显示子函数，用于显示定时 2 时间
******************
void display2()
{   uchar i,j;
     if(alm2==0)
        { if(cnt!=4||wss==0)
```

```
            { table3[0]=miao2%10;              //以下含义同上
                table3[1]=miao2/10;
             }
          else
            { table3[0]=table3[1]=11;}
          if(cnt!=5||wss==0)
            { table3[3]=fen2%10;
                table3[4]=fen2/10;
              }
           else
            { table3[3]=table3[4]=11;}
           if(cnt!=6||wss==0)
            { table3[6]=shi2%10;
                table3[7]=shi2/10;
              }
           else
            { table3[6]=table3[7]=11;}
          }
     else
          table3[0]=table3[1]=table3[3]=table3[4]=table3[6]=table3[7]=10;
    table3[2]= table3[5]=10;
    j=0xfb;
    for(i=0;i<=7;i++)
       {    P2=j;
             P0=table[table3[i]];
           delay(10);
             j=_cror_(j,1);
           }
}
```

```
 //***************显示子函数，用于显示定时 3 时间数值****************//
void display3()
{   uchar i,j;
    if(alm3==0)
      {   if(cnt!=7||wss==0)
            { table4[0]=miao3%10;            //分离秒的个位与十位
              table4[1]=miao3/10;
              }
          else
            { table4[0]=table4[1]=11;}
          if(cnt!=8||wss==0)
            { table4[3]=fen3%10;             //分离分的个位与十位
              table4[4]=fen3/10;
             }
          else
            {  table4[3]=table4[4]=11;}
          if(cnt!=9||wss==0)
            { table4[6]=shi3%10;             //分离时的个位与十位
              table4[7]=shi3/10;
             }
          else
            {  table4[6]=table4[7]=11;}
           }
     else
         table4[0]=table4[1]=table4[3]=table4[4]=table4[6]=table4[7]=10;
    table4[2]= table4[5]=10;
    j=0xfb;                              //从秒到时的扫描
    for(i=0;i<=7;i++)
      {   P2=j;
```

```
            P0=table[table4[i]];   //显示数值
           delay(10);
            j=_cror_(j,1);            //循环右移
           }
   }
 //**********************时间子函数 ****************************//
void shijian()
{   if(flag>=20)                    //判断是否到一秒
       { wss=~wss;
         flag=0;                        //到了，则标志位清零
         if(cnt1!=0)
            { miao4++;               //秒加 1
              if( miao4>59)       //判断秒是否到 60s
                { miao4=0;          //到了，则清零
                  fen4++;           //分加 1
                  if(fen4>59)   //以下含义同上
                    { fen4=0;
                      shi4++;
                      if(shi4>23)
                        shi4=0;

                    }
                 }
            }
         else
         { miao++;                //秒加 1
           if( miao>59)       //判断秒是否到 60s
             { miao=0;          //到了，则清零
               fen++;           //分加 1
```

```
                if(fen>59)    //以下含义同上
                  { fen=0;
                    shi++;
                    if(shi>23)
                      shi=0;

                    }
                }
            }
        }
    }
//*************************键盘扫描子函数************************//
void key_scan()
{   if(key1==0)
      {   while(!key1)              //防止掉显
            {   if(cnt==1||cnt==2||cnt==3)
                  { display1(); }
                if(cnt==4||cnt==5||cnt==6)
                  { display2(); }
                if(cnt==7||cnt==8||cnt==9)
                  { display3(); }
                if(cnt==0||cnt==10||cnt==11||cnt==12||cnt==13)
                  { display(); }
                }
          cnt++;        //记下按键 key1 按下的次数
        if(cnt==10&&cnt1==0)
          { miao4=miao;
            fen4=fen;
            shi4=shi;
            cnt1++;
```

```
        }
    if(cnt==13)
        { cnt=0;
          if(cnt1==1)
              { miao=miao4;
                fen=fen4;
                shi=shi4;
                }
          cnt1=0;
        }
    }
if(key2==0)                    //判断 key2 是否按下
{   while(!key2)               //防止掉显
        {   if(cnt==1||cnt==2||cnt==3)
                { display1(); }
         if(cnt==4||cnt==5||cnt==6)
                { display2(); }
         if(cnt==7||cnt==8||cnt==9)
                { display3(); }
         if(cnt==0||cnt==10||cnt==11||cnt==12||cnt==13)
                { display(); }
          }
     jia();
    }
if(key3==0)                    //判断 key3 是否按下
{   while(!key3)               //防止掉显
        {   if(cnt==1||cnt==2||cnt==3)
                { display1(); }
         if(cnt==4||cnt==5||cnt==6)
                { display2(); }
```