

一、设计题目:

电子万年历

二、设计原理:

单片机应用系统由硬件系统和软件系统两部分组成。硬件系统是指单片机以及扩展的存储器、I/O 接口、外围扩展的功能芯片以及接口电路。软件系统包括监控程序和各种应用程序。

在单片机应用系统中,单片机是整个系统的核心,对整个系统的信息输入、处理、信息输出进行控制。与单片机配套的有相应的复位电路、时钟电路以及扩展的存储器和 I/O 接口,使单片机应用系统能够运行。

本设计利用 2kHz 的中断每 0.5 秒钟产生一个中断来产生秒的计数,进而实现时钟部分的功能。在每次时间改变时调用 LCD 显示子程序,来实现时间的显示。用实验板上自带的三个按键来实现报时功能及时间调整功能,并利用置不同的标志位来实现年月日时分秒的调整。利用 A2000 自动播放进行时间的播报。

三、设计要求:

- 1、key1——按一下播放年并进入年的调整状态,再按一下下播放月并进入月的调整状态,以此类推到日、时、分、秒;
- 2、key2——年、月、日、时、分、秒的增加;
- 3、key3——年、月、日、时、分、秒的减少;

4、复位键——按下之后复位

实现功能:

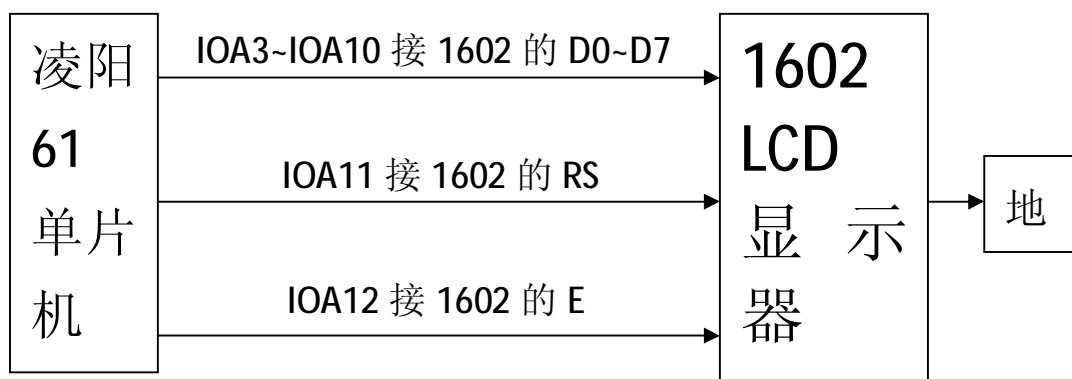
1、数字钟功能

3、报时功能、闹表功能、定时提醒功能

四、设计材料

这个设计的材料有: 1602LCD 显示器、18B20、凌阳 61 单片机、排线、排针、连接线、电池等

五、设计流程图



六、设计源程序

```
/**
//项目: e_clock
//实现功能: 1、数字钟功能
//           2、测温功能、提醒功能
//           3、报时功能、闹表功能、定时提醒功能

//作者: 郭新新
//cpu 时钟调整: P_SystemClock 重新赋值, 比如语句[P_SystemClock]=0x31(00110001),
//则将系统始终(PLL)更换为 20.480MHz, CPU 始终选择为 Fosc/2, 即 10.240MHz, 约为 0.098us,
//那么如果要实现 1us 的延时, 只要实现将延时程序中语句总的执行周期约为 10 个 CPU
//周期即可
**/

#include "spce061a.h"
#include spce061A.inc
```

```
#include "SPCE061A.H"
#include "e_clock.h"
#include "SPCE061V004.H"
#include "unSPMACRO.h"
// #include "hardware.h"
#include "a2000.h"
#include "IO_bit.h"
#include "s480.h"

//*****//
//函数声明
//*****//
int leap_year();
void delay (void);
int Remind(void);
void c_1602(int c);
void d_1602(int d);
void character(void);
void display_1602(void);
int getkey(void);
void check_clock(int uikey);
void tempture(void);
void IO_Init(void);
void voice_2000(int num);
void world_end(void);
void flag_Init(void);
void del_10us();
void PlayNum(unsigned int Num);
void PlayYear(unsigned int Year);
void PlayMon(unsigned int Mon);
void PlayDay(unsigned int Day);
void PlayHour(unsigned int Hour);
void PlayMin(unsigned int Min);
void PlaySec(unsigned int Sec);
void PlayTime(void);
void PlayDate(void);
void time_Init();
void chang_flag(unsigned int uikey);

//*****//
//闰年判断子程序
//输入参数: year (年份)
```

```
//返回参数: leap: leap=1 是闰年 leap=0 不是闰年
//*****//
int leap_year()
{
    unsigned int leap=0;

    if(year%400==0)
        leap=1;
    else if((year%4==0)&&(year%100!=0))
        leap=1;
    else
        leap=0;
    return (leap);
}

//*****//
//延时子程序
//输入参数: 无
//返回参数: 无
//演示时间: 约 1ms
//*****//
void delay (void)
{
    unsigned int i;
    unsigned int j=10;
    while(j)
    {
        j--;
        for(i=0;i<204;i++);
    }
}

//*****//
//提醒子程序
//6:50    起床
//7:30    上课
//12:40   午休
//13:45   起床
//18:50   玩自习
//23:30   睡觉
//传递参数: int hour,int minute
//返回参数: int Rem    (语音播放索引号)
```

```
/*******//
```

```
int Remind(void)
{
    unsigned int i;
    int hr[]={6,6,6};//7,23};//12,13,18,23};          //小时
    int mite[]={50,51,52,};//30,30};//40,45,50,30};    //分钟
    unsigned int Rem=0;                                //定义返回参数
    for(i=0;i<6;i++)
    {
        if(hour==hr[i]&&minute==mite[i])
            Rem=i+1;
    }
    return(Rem);
}
```

```
/*******//
```

```
//发送命令字子程序
//传递参数: c (命令字)
//返回参数: 无
//数据通过 IOA3-10 口传递
//IOA11 连 RS 寄存器选择端
//IOA12 接 E 使能信号 E=1 时读取信息 E 下降沿执行命令 ...
//将 R/W 接地, 设为只写, 不读
```

```
/*******//
```

```
void c_1602(int c)
{
    //asm("c=c<<3");
    //asm("%0<=<=%1":"=m"(c):"i"(3));
    //c<<=3;                //c 左移 3 位
    unsigned int rs=0;
    rs=*P_IOA_Data;        //读取 IOB 口数据
    rs=(rs&0xf7ff);        //指向指令寄存器
    *P_IOA_Data=rs;        //指向指令寄存器
    c<<=3;
    rs=(RS&0xf807)|c;      //得到要输出的数据
    *P_IOA_Data=rs;        //写入命令字
    rs=(rs&0xefff)|0x1000; //E 端使能端置 1
    *P_IOA_Data=rs;        //E 端即使能端置 1
    delay();
    rs=rs&0xefff;
    *P_IOA_Data=rs;        //E 端, 即使能端置 0, 创造一个下降沿
```

```
        // IORS=0X00;           //选到命令寄存器
        // IOWR=C;             //写入命令字

    delay();
}

//*****//
//写数据子程序
//传递参数: d (字模)
//返回参数: 无
//*****//
void d_1602(int d)
{
    //d<<=3;                   //c 左移 3 位
    unsigned int S;
    S=*P_IOA_Data;             //读取 IOB 口数据
    S=(S&0xf7ff)|0x0800;      //指向指令寄存器
    *P_IOA_Data=S;            //指向指令寄存器
    d<<=3;
    S=(S&0xf807)|d;           //得到要输出的数据
    *P_IOA_Data=S;            //写入命令字
    S=(S&0xefff)|0x1000;     //E 端使能端置 1
    *P_IOA_Data=S;           //E 端即使能端置 1
    delay();
    S=S&0xefff;
    *P_IOA_Data=S;           //E 端, 即使能端置 0, 创建一个下降沿
                                // IORS=0X01;           //选到数据寄存器
                                // IOWR=d;             //写入数据

    //delay();
}

//*****//
//写入年月日摄氏度电工一字模子程序
//传递参数: 无
//返回参数: 无
//*****//

void character(void)
{
    unsigned int i;
    int year[]={0x08,0x0f,0x12,0x0f,0x0a,0x1f,0x02,0x02}; //年字的字模
    int month[]={0x0f,0x09,0x0f,0x09,0x0f,0x09,0x11,0x00}; //月字的字模
    int day[]={0x0f,0x09,0x09,0x0f,0x09,0x09,0x0f,0x00}; //日字的字模
}
```

```
int temp[]={0x10,0x06,0x09,0x08,0x08,0x09,0x06,0x00};           //字符℃
int dian[]={0x04,0x1F,0x15,0x1F,0x15,0x1F,0x04,0x07};
int gong[]={0x00,0x1F,0x04,0x04,0x04,0x04,0x1F,0x00};

c_1602(0x0040);           //选择年字的地址
for(i=0;i<8;i++)         //写入年字
{
    d_1602(year[i]);
}

c_1602(0x0048);           //选择月字的地址
for(i=0;i<8;i++)         //写入月字
{
    d_1602(month[i]);
}

c_1602(0x0050);           //选择日字的地址
for(i=0;i<8;i++)         //写入日字
{
    d_1602(day[i]);
}

c_1602(0x0058);           //选择℃的地址
for(i=0;i<8;i++)         //写入℃
{
    d_1602(temp[i]);
}

c_1602(0x0060);           //选择电工一地址
for(i=0;i<8;i++)         //写入电工一
{
    d_1602(dian[i]);
}
for(i=0;i<8;i++)
{
    d_1602(gong[i]);
}
}

//*****//
//显示子程序
```

```
//传递参数: year, month, day, hour, minite, temp
//返回参数: 无
//作用: 显示年月日时分星期
//*****//
void display_1602(void)
{
    unsigned int num[10]={0x30,0x31,0x32,0x33,0x34,0x35,0x36,0x37,0x38,0x39};    //0 到 9
    字符代码
    delay();                            //延时等待复位
    c_1602(0x0038);                       //设置为 8 总线, 16*2,5*7 点阵
    c_1602(0x0001);                       //清屏
    c_1602(0x0006);                       //光标移动, 显示区不动
    c_1602(0x000c);                       //开显示, 光标闪烁
    c_1602(0x0080);                       //从第一行第一列开始显示
    d_1602(num[year/1000]);                //显示年份
    d_1602(num[year/100%10]);
    d_1602(num[year%100/10]);
    d_1602(num[year%10]);
    d_1602(0x00);                          //40;                //显示年字
    d_1602(num[month/10]);                 //显示月份
    d_1602(num[month%10]);
    d_1602(0x0001);                          //41;                //显示月字
    d_1602(num[day/10]);                   //显示日期
    d_1602(num[day%10]);
    d_1602(0x0002);                          //42;                //显示日字
    c_1602(0x008d);                         //显示星期的位置
    switch(week)
    {
        case 1:                             //Monday
            d_1602(0x004d);
            d_1602(0x006f);
            d_1602(0x006e);
            break;
        case 2:                             //Tuesday
            d_1602(0x0054);
            d_1602(0x0075);
            d_1602(0x0065);
            break;
        case 3:                             //Wednesday
            d_1602(0x0057);
            d_1602(0x0066);
            d_1602(0x0065);
            break;
    }
}
```

```
case 4: //Tuesday
    d_1602(0x0054);
    d_1602(0x0075);
    d_1602(0x0072);
    break;
case 5: //Friday
    d_1602(0x0046);
    d_1602(0x0072);
    d_1602(0x0069);
    break;
case 6: //Saturday
    d_1602(0x0053);
    d_1602(0x0061);
    d_1602('t');//0x0054);
    break;
case 7: //Sunday
    d_1602(0x0053);
    d_1602(0x0075);
    d_1602(0x006e);
    break;
default:
    break;
}
c_1602(0x00c2); //第二行第一列
d_1602(num[hour/10]); //显示小时
d_1602(num[hour%10]);
d_1602(0x003a); //显示冒号
d_1602(num[minute/10]); //显示分钟
d_1602(num[minute%10]);
//c_1602(0x00c6); //第二行第 7 列
//d_1602(num[temp/10]); //显示温度
//d_1602(num[temp%10]);
//d_1602(0x002e); //显示小数
//d_1602(num[temp*10%10]);
// d_1602(0x0003);//43); //显示℃
c_1602(0x00ca); //跳过一位
d_1602(0x0004);//5); //显示电字
d_1602(0x0005); //显示工字
d_1602(0x00b0); //显示一字
}
```

```
//*****//
//第二块 LCD 显示
//D0-D7 接 IOB0-IOB7 RS 接 IOB8 E 接 IOB9 RW 接地
//c_1602a
//*****//
void c_1602a(int code)
{
    RS=0;                //指向指令寄存器
    *P_IOB_Data>(*P_IOB_Data&0xff00)|code;    //输出数据
    E=1;
    delay();
    E=0;
}

//*****//
//
//*****//

void d_1602a(int data)
{
    RS=1;
    *P_IOB_Data>(*P_IOB_Data&0xff00)|data;
    E=1;
    delay();
    E=0;
}
//*****//
//workhard
//*****//
void workhard(unsigned int w)
{
    delay();                //延时等待复位
    c_1602a(0x0038);        //设置为 8 总线, 16*2,5*7 点阵
    c_1602a(0x0001);        //清屏
    c_1602a(0x0006);        //光标移动, 显示区不动
    c_1602a(0x000c);        //开显示, 光标闪烁
    c_1602a(0x008f-w);      //从第一行第一列开始显示
    d_1602a('l');
    c_1602a(0x0090-w);
    d_1602a('t');
    c_1602a(0x0092-w);
    d_1602a('i');
    c_1602a(0x0093-w);
}
```

```
d_1602a('s');
c_1602a(0x0095-w);
d_1602a('t');
c_1602a(0x0096-w);
d_1602a('h');
c_1602a(0x0097-w);
d_1602a('e');
c_1602a(0x0099-w);
d_1602a('t');
c_1602a(0x009a-w);
d_1602a('i');
c_1602a(0x009b-w);
d_1602a('m');
c_1602a(0x009c-w);
d_1602a('e');
c_1602a(0x009f-w);
d_1602a('t');
c_1602a(0x00a0-w);
d_1602a('o');
c_1602a(0x00a2-w);
d_1602a('w');
c_1602a(0x00a3-w);
d_1602a('o');
c_1602a(0x00a4-w);
d_1602a('r');
c_1602a(0x00a5-w);
d_1602a('k');
c_1602a(0x00a7-w);
d_1602a('h');
c_1602a(0x00a8-w);
d_1602a('a');
c_1602a(0x00a9-w);
d_1602a('r');
c_1602a(0x00aa-w);
d_1602a('d');
c_1602a(0x00ab-w);
d_1602a('!');
c_1602a(0x00ac-w);
d_1602a('!');
}

//*****//
//getup      起床
```

```
/*******//  
void getup()  
{  
    flag_wh=1;  
    m=0;  
    n=0;  
    delay(); //延时等待复位  
    c_1602a(0x0038); //设置为 8 总线, 16*2,5*7 点阵  
    c_1602a(0x0001); //清屏  
    c_1602a(0x0006); //光标移动, 显示区不动  
    c_1602a(0x000c); //开显示, 光标闪烁  
    c_1602a(0x0080); //从第一行第一列开始显示  
    d_1602a('I');  
    d_1602a('t');  
    c_1602a(0x0083);  
    d_1602a('i');  
    d_1602a('s');  
    c_1602a(0x0086);  
    d_1602a('t');  
    d_1602a('h');  
    d_1602a('e');  
    c_1602a(0x008a);  
    d_1602a('t');  
    d_1602a('i');  
    d_1602a('m');  
    d_1602a('e');  
    c_1602a(0x00c0);  
    d_1602a('t');  
    d_1602a('o');  
    c_1602a(0x00c3);  
    d_1602a('g');  
    d_1602a('e');  
    d_1602a('t');  
    c_1602a(0x00c7);  
    d_1602a('u');  
    d_1602a('p');  
    c_1602a(0x00ca);  
    d_1602a('!');  
    d_1602a('!');  
    d_1602a('!');  
    d_1602a('!');  
}
```

```
void class1()
{
    flag_wh=1;
    m=0;
    n=0;
    delay(); //延时等待复位
    c_1602a(0x0038); //设置为 8 总线, 16*2,5*7 点阵
    c_1602a(0x0001); //清屏
    c_1602a(0x0006); //光标移动, 显示区不动
    c_1602a(0x000c); //开显示, 光标闪烁
    c_1602a(0x0080); //从第一行第一列开始显示
    d_1602a('!');
    d_1602a('t');
    c_1602a(0x0083);
    d_1602a('i');
    d_1602a('s');
    c_1602a(0x0086);
    d_1602a('t');
    d_1602a('h');
    d_1602a('e');
    c_1602a(0x008a);
    d_1602a('t');
    d_1602a('i');
    d_1602a('m');
    d_1602a('e');
    c_1602a(0x00c0);
    d_1602a('f');
    d_1602a('o');
    d_1602a('r');
    c_1602a(0x00c4);
    d_1602a('c');
    d_1602a('l');
    d_1602a('a');
    //c_1602a(0x00c7);
    d_1602a('s');
    d_1602a('s');
    c_1602a(0x00c9);
    d_1602a('!');
    d_1602a('!');
    d_1602a('!');
```

```
d_1602a('!');  
}
```

```
void sleep()
```

```
{  
    flag_wh=1;  
    m=0;  
    n=0;  
    delay(); //延时等待复位  
    c_1602a(0x0038); //设置为 8 总线, 16*2,5*7 点阵  
    c_1602a(0x0001); //清屏  
    c_1602a(0x0006); //光标移动, 显示区不动  
    c_1602a(0x000c); //开显示, 光标闪烁  
    c_1602a(0x0080); //从第一行第一列开始显示  
    d_1602a('I');  
    d_1602a('t');  
    c_1602a(0x0083);  
    d_1602a('i');  
    d_1602a('s');  
    c_1602a(0x0086);  
    d_1602a('t');  
    d_1602a('h');  
    d_1602a('e');  
    c_1602a(0x008a);  
    d_1602a('t');  
    d_1602a('i');  
    d_1602a('m');  
    d_1602a('e');  
    c_1602a(0x00c0);  
    d_1602a('t');  
    d_1602a('o');  
    c_1602a(0x00c3);  
    d_1602a('s');  
    d_1602a('l');  
    d_1602a('e');  
    //c_1602a(0x00c7);  
    d_1602a('e');  
    d_1602a('p');  
    c_1602a(0x00ca);  
    d_1602a('!');
```

```
d_1602a('!');
d_1602a('!');
d_1602a('!');
}

//*****//
//display
//传递参数: Rem
//*****&//
void display(Rem)
{
    switch(Rem)
    {
        case 1:  getup(); break;
        case 2:  class1(); break;
        case 3:  sleep();  break;
        default: break;
    }
}

//*****//
//键盘扫描子程序
//传递参数: 无
//返回参数: uikey
//*****//
int getkey(void)
{
    unsigned int uikey;      //定义返回值变量
    unsigned int mid_key;   //
    uikey=*P_IOA_Data&0x0007;
// mid_key=uikey&0x0007;
if(uikey)&&0x0007)
{
    delay();                //延迟 10ms
    uikey=*P_IOA_Data;
    if((uikey&0x0007)!=0)
    {
        uikey=*P_IOA_Data&0x0007;
        mid_key=uikey;
        while(uikey&0x0007)    //等待键弹起
    }
}
```

```
        {
            uikey=*P_IOA_Data;//&0x0007;
            *P_Watchdog_Clear=0x0001;
            // uikey&=0x0007;
        }
        uikey=mid_key&0x0007;
    }
    else
        uikey=0;
}
else
    uikey=0;
return(uikey);
}

//*****//
//改变 flag 的值
//*****//
void chang_flag(unsigned int uikey)
{
    if(flag_clock==1)
    {
        flag_clock=0x0000;           //报时标记清零
        flag_yearqian=0x0001;       //年千位标记置 1
        PlayYear(year);
    }
    else if(flag_yearqian==1)
    {
        flag_yearqian=0;
        flag_yearbai=1;
        PlayYear(year);
    }
    else if(flag_yearbai==1)
    {
        flag_yearbai=0;
        flag_yearshi=1;
        PlayYear(year);
    }
    else if(flag_yearshi==1)
    {
        flag_yearshi=0;
        flag_yearge=1;
    }
}
```

```
    PlayYear(year);
}
else if(flag_yearge==1)
{
    flag_yearge=0;
    flag_month=1;
    PlayMon(month);
}
else if(flag_month==1)
{
    flag_month=0;
    flag_day=1;
    PlayDay(day);
}
else if(flag_day==1)
{
    flag_day=0;
    flag_week=1;
}
else if(flag_week)
{
    flag_week=0;
    flag_hour=1;
    PlayHour(hour);
}
else if(flag_hour==1)
{
    flag_hour=0;
    flag_minute=1;
    PlayMin(minute);
}
else if(flag_minute==1)
{
    flag_minute=0;
    flag_clock=1;
    PlayDate();
    PlayTime();
    __asm("INT IRQ");
}
}
```

```
//*****//
//时间校正子程序
//传递参数: 无
//返回参数: 无
//*****//
void check_clock(int uikey)
{
    unsigned int a=year/1000;           //年的千位
    unsigned int b=year/100%10;        //年的百位
    unsigned int c=year%100/10;        //年的十位
    unsigned int d=year%10;            //年的个位
    unsigned int _leap;                 //闰年判断

    INT_OFF();
    if(flag_clock==1)
    {
        switch(uikey)
        {
            case key2:    PlayDate();
                        PlayTime();
                        break;

            case key3:    // voice_2000(now_t);           //"现在温度"
                        // voice_2000(du);             //"度"
                        break;

            default :    break;
        }
    }

    if(flag_yearqian==1)
    {
        switch(uikey)
        {
            case key2:    c_1602(0x000c);           //光标闪烁
                        c_1602(0x0080);           //千位闪烁
                        a++;                         //千位加 1
                        if(a>9)
                            a=2;
                        year=1000*a+100*b+10*c+d;   //复得 year
                        display_1602();           //显示
                        PlayYear(year);
                        break;

            case key3:    c_1602(0x000c);
        }
    }
}
```

```
        c_1602(0x0080);
        if(a>0)
            a--;
        else
            a=9;
            year=1000*a+100*b+10*c+d;
            display_1602();
            PlayYear(year);
            break;
    default:    break;
}
}

if(flag_yearbai==1)
{
    switch(uikey)
    {
        case key2:    c_1602(0x000c);
                     c_1602(0x0081);
                     b++;
                     if(b>9)
                         b=0;
                     year=1000*a+100*b+10*c+d;
                     display_1602();
                     PlayYear(year);
                     break;
        case key3:    c_1602(0x000c);
                     c_1602(0x0081);
                     if(b>0)
                         b--;
                     else
                         b=9;
                     year=1000*a+100*b+10*c+d;
                     display_1602();
                     PlayYear(year);
                     break;
        default :    break;
    }
}

if(flag_yearshi==1)
{
    switch(uikey)
    {
```

```
        case key2:  c_1602(0x000c);
                    c_1602(0x0082);
                    c++;
                    if(c>9)
                        c=0;
                    year=1000*a+100*b+10*c+d;
                    display_1602();
                    PlayYear(year);
                    break;
        case key3:  c_1602(0x000c);
                    c_1602(0x0082);
                    if(c>0)
                        c--;
                    else
                        c=9;
                    year=1000*a+100*b+10*c+d;
                    display_1602();
                    PlayYear(year);
                    break;
        default:    break;
    }
}
}
if(flag_yearge==1)
{
    switch(uikey)
    {
        case key2:  c_1602(0x000c);
                    c_1602(0x0083);
                    d++;
                    if(d>9)
                        d=0;
                    year=1000*a+100*b+10*c+d;
                    display_1602();
                    PlayYear(year);
                    break;
        case key3:  c_1602(0x000c);
                    c_1602(0x0083);
                    if(d>0)
                        d--;
                    else
                        d=9;
                    year=1000*a+100*b+10*c+d;
                    display_1602();
```

```
        PlayYear(year);
        break;
    default:    break;
    }
}
if(flag_month==1)
{
    switch(uikey)
    {
        case key2:  c_1602(0x000c);
                    c_1602(0x0086);
                    month++;
                    if(month>12)
                        month=1;
                    display_1602();
                    PlayMon(month);
                    break;
        case key3:  c_1602(0x000c);
                    c_1602(0x0086);
                    if(month>1)
                        month--;
                    else
                        month=12;
                    display_1602();
                    PlayMon(month);
                    break;
        default:    break;
    }
}
if(flag_day==1)
{
    switch(uikey)
    {
        case key2:  c_1602(0x000c);
                    c_1602(0x0089);
                    _leap=leap_year();

if(month==1 || month==3 || month==5 || month==7 || month==8 || month==10 || month==12)
        {
            day++;
            if(day>31)
                day=1;
        }
    }
}
```

```
        if(month==4 || month==6 || month==9 || month==11)
        {
            day++;
            day=1;
        }
        if(month==2)
        {
            if(_leap==1)
            {
                day++;
                if(day>29)
                    day=1;
            }
            else
            {
                day++;
                if(day>28)
                    day=1;
            }
        }

        display_1602();
        PlayDay(day);
        break;
    case key3:    _leap=leap_year();

if(month==1 || month==3 || month==5 || month==7 || month==8 || month==10 || month==12)
    {
        if(day>1)
            day--;
        else
            day=31;
    }
    if(month==4 || month==6 || month==9 || month==11)
    {
        if(day>1)
            day--;
        else
            day=30;
    }
    if(month==2)
    {
        if(_leap==1)
```

```
        {
            if(day>1)
                day--;
            else
                day=29;
        }
    else
    {
        if(day>1)
            day--;
        else
            day=28;
    }
}
display_1602();
PlayDay(day);
break;
default: break;
}
}
if(flag_week==1)
{
    switch(uikey)
    {
        case key2: c_1602(0x000c);
                  c_1602(0x008f);
                  week++;
                  if(week>7)
                      week=1;
                  display_1602();
                  break;
        case key3: c_1602(0x000c);
                  c_1602(0x008f);
                  if(week>1)
                      week--;
                  else
                      week=7;
                  display_1602();
                  break;
        default: break;
    }
}
if(flag_hour==1)
```

```
{
    switch(uikey)
    {
        case key2: c_1602(0x000c);
                  c_1602(0x00c1);
                  hour++;
                  if(hour>23)
                      hour=0;
                  display_1602();
                  PlayHour(hour);
                  break;
        case key3: c_1602(0x000c);
                  c_1602(0x00c1);
                  if(hour>0)
                      hour--;
                  else
                      hour=23;
                  display_1602();
                  PlayHour(hour);
                  break;
        default:  break;
    }
}
if(flag_minute==1)
{
    switch(uikey)
    {
        case key2: c_1602(0x000c);
                  c_1602(0x00c4);
                  minute++;
                  if(hour>59)
                      minute=0;
                  display_1602();
                  PlayMin(minute);
                  break;
        case key3: c_1602(0x000c);
                  c_1602(0x00c4);
                  if(minute>0)
                      minute--;
                  else
                      minute=59;
                  display_1602();
    }
}
```

```
        PlayMin(minute);
        break;
    default: break;
}
}

//=====
// 语法格式:    void PlaySnd(int SndIndex);
// 实现功能:    播放一段语音
// 参数:        SndIndex - 待播报语音的序号
// 返回值:      无
//=====
void PlaySnd(int SndIndex)
{
    SACM_A2000_Initial(1);           // 初始化为自动播放
    SACM_A2000_Play(SndIndex, 3, 3); // 开始播放一段语音
    while((SACM_A2000_Status)&0x0001)!= 0) // 是否播放完毕?
    {
        SACM_A2000_ServiceLoop(); // 解码并填充队列
        *P_Watchdog_Clear = 0x01;
    }
    SACM_A2000_Stop();              // 停止播放
}

void PlayNum(unsigned int Num)
{
    unsigned int TempNum = Num;

    if(TempNum==0)                  // 数字为 0 则直接播报 0
        PlaySnd(0);//S_0);
    else
    {
        if(TempNum>=20) PlaySnd(TempNum/10); // 十位
        if(TempNum>=10) PlaySnd(10);//S_10); // "十"
        TempNum = TempNum % 10;
        if(TempNum>0) PlaySnd(TempNum);      // 个位
    }
}
}
```

```
//=====
// 语法格式:    void PlayYear(unsigned int Year);
// 实现功能:    播报年
// 参数:        Year - 待播报的年
// 返回值:      无
```

```
//=====
void PlayYear(unsigned int Year)
{
    unsigned int TempYear = Year;

    PlaySnd(TempYear/1000);           // 千位
    TempYear = TempYear % 1000;
    PlaySnd(TempYear/100);          // 百位
    TempYear = TempYear % 100;
    PlaySnd(TempYear/10);           // 十位
    TempYear = TempYear % 10;
    PlaySnd(TempYear);              // 个位
    PlaySnd(20);//S_Nian;           // "年"
}

```

```
//=====
// 语法格式:    void PlayMon(unsigned int Mon);
// 实现功能:    播报月
// 参数:        Mon - 待播报的月
// 返回值:      无
//=====
void PlayMon(unsigned int Mon)
{
    PlaySnd(Mon);
    PlaySnd(21);//S_Yue;
}

```

```
//=====
// 语法格式:    void PlayDay(unsigned int Day);
// 实现功能:    播报日
// 参数:        Day - 待播报的日
// 返回值:      无
//=====
void PlayDay(unsigned int Day)
{
    PlayNum(Day);
    PlaySnd(22);//S_Ri);
}
//=====
// 语法格式:    void PlayHour(unsigned int Hour);
// 实现功能:    播报小时
// 参数:        Hour - 待播报的时
// 返回值:      无
//=====
void PlayHour(unsigned int Hour)
{
    if(Hour<=5) PlaySnd(16);//S_LC);           // 凌晨(0~5 点)
    else if(Hour<=11) PlaySnd(17);//S_SW);      // 上午(6~11 点)
    else if(Hour<=17) PlaySnd(18);//S_XW);      // 下午(12~17 点)
    else PlaySnd(19);//S_WS);                  // 晚上(18~23 点)

    if(Hour==2 || Hour==14) PlaySnd(13);//S_Liang); // 2 点和 14 点播报"两"
    else if(Hour>12) PlaySnd(Hour-12);         // 12 小时制
    else PlaySnd(Hour);
    PlaySnd(23);//S_Dian);                     // "点"
}
//=====
// 语法格式:    void PlayMin(unsigned int Min);
// 实现功能:    播报分
// 参数:        Min - 待播报的分
// 返回值:      无
//=====
void PlayMin(unsigned int Min)
{
    if(Min==0)PlaySnd(26);//S_Zheng);          // 00 分播报"整"
    else
    {
        PlayNum(Min);                          // 播报数字
    }
}
```

```
        PlaySnd(24);//S_Fen);                                // "分"
    }
}

//=====
// 语法格式:      void PlaySec(unsigned int Sec);
// 实现功能:      播报秒
// 参数:          Sec - 待播报的秒
// 返回值:        无
//=====
void PlaySec(unsigned int Sec)
{
    PlayNum(Sec);
    PlaySnd(25);//S_Miao;
}

//=====
// 语法格式:      void PlayTime();
// 实现功能:      播报当前时间
// 参数:          无
// 返回值:        无
//=====
void PlayTime()
{
    PlaySnd(15);//S_XZSK);
    PlayHour(hour);
    PlayMin(minute);
}

//=====
// 语法格式:      void PlayDate();
// 实现功能:      播报当前日期
// 参数:          无
// 返回值:        无
//=====
void PlayDate()
{
    PlayYear(year);//TempYear);
    PlayMon(month);//TempMon);
    PlayDay(day);//TempDay);
}
```

```
//*****//  
//温度测试子程序  
//  
//*****//  
void tempture(void)  
{  
}  
  
//*****//  
//初始化 IO 口子程序  
//传递参数: 无  
//返回参数: 无  
//*****//  
void IO_Init(void)  
{  
    *P_IOA_Dir=0xffff;           //IOA 口低 3 位用作按键输入, 其它用作输出口  
    *P_IOA_Attrib=0xffff;       //带数据缓冲器  
    *P_IOA_Data=0x0000;  
    *P_IOB_Dir=0xffff;  
    *P_IOB_Attrib=0xffff;  
    *P_IOB_Data=0x0000;  
  
    *P_SystemClock=0x31;        //系统始终(PLL)更换为 20.480MHz, CPU 始终选择  
    为 Fosc/2, 即 10.240MHz, 约为 0.098us  
    *P_INT_Ctrl=C_IRQ5_2Hz;    //|C_FIQ_TMA;           //打开中断 IRQ5  
    __asm("INT IRQ");         //开中断  
}  
  
//*****//  
//时间初始化子程序  
//*****//  
void time_Init()  
{  
    year=2010;  
    month=5;  
    day=1;
```

```
hour=6;
minute=49;
week=6;
second=50;
}

//*****//
//语音播报字程序
//传递参数: 播放序列号
//返回参数: 无
//功能: 根据传递来的播放序列号播放语音
//
//*****//
void voice_2000(int num)
{
    SACM_A2000_Initial(0);          //自动初始化
    SACM_A2000_Play(num,DAC1+DAC2,Ramp_updn_on);          //双通道, 允许声音增
减播放
    while(SACM_A2000_Status()&0x01)
    {
        SACM_A2000_ServiceLoop();
        *P_Watchdog_Clear=1;
    }
    SACM_A2000_Stop();
}

//*****//
//当 year=9999 时, lcd 显示 the world end 字样
//传递参数: 无
//返回参数: 无
//*****//
void world_end(void)
{
    delay();          //延时等待复位
    c_1602(0x0038);   //设置为 8 总线, 16*2,5*7 点阵
    c_1602(0x0001);   //清屏
    c_1602(0x0006);   //光标移动, 显示区不动
    c_1602(0x000c);   //开显示, 光标闪烁
    c_1602(0x0083);   //从第一行第四列开始显示
    d_1602(0x0054);   //T
    d_1602(0x0068);   //h
}
```

```
d_1602(0x0065);           //e
c_1602(0x0088);           //跳一格
d_1602(0x0045);           //E
d_1602(0x006e);           //n
d_1602(0x0064);           //d
c_1602(0x008c);           //空一格
d_1602('o');               //o
d_1602('f');               //f
c_1602(0x00c3);           //第二行第四列开始
d_1602(0x0054);           //T
d_1602(0x0068);           //h
d_1602(0x0065);           //e
c_1602(0x00c8);           //空一格
d_1602(0x0057);           //W
d_1602(0x006f);           //o
d_1602(0x0072);           //r
d_1602(0x006c);           //l
d_1602(0x0064);           //d
d_1602(0x0021);           //!(惊叹号)
d_1602(0x0021);
d_1602(0x0021);
d_1602(0x0021);
d_1602(0x0021);

}

//*****//
//校时程序中用到的 flag 初始化
//传递参数: 无
//返回参数: 无
//*****//
void flag_Init()
{
    flag_yearqian=0x0000;    //年的千位标记
    flag_yearbai=0x0000;    //年的百位标记
    flag_yearshi=0x0000;    //年的十位标记
    flag_yearge=0x0000;     //年的个位标记
    flag_month=0x0000;      //月的标记
    flag_day=0x0000;        //天的标记
    flag_hour=0x0000;       //小时的标记
    flag_minute=0x0000;    //分钟的标记
    flag_week=0x0000;       //星期的标记
    flag_clock=0x0001;      //报时标记
}
```

```
flag_wh=0x0000;  
}
```

```
void display_wh()  
{  
  
}
```

```
//main.c
```

```
int main()  
{  
    unsigned int key;  
  
    IO_Init();           //IO 口初始化  
    character();  
    flag_Init();        //写入年月日等代码  
    time_Init();        //时间初始化  
    tempture();         //读温度  
    display_1602();     //显示  
    c_1602a(0x0001);   //清屏  
    while(1)  
    {  
  
        *P_IOA_Data=*P_IOA_Data&0xff8;  
        key=getkey();   //读取键值  
        if(key==0)  
        {  
            n++;  
        }  
    }  
}
```

```
        if(n==0xffff)
        {
            m++;
        }
        if(m==50)
        {
            __asm("INT IRQ");
            flag_Init();
            delay();
            c_1602a(0x0001);           //清屏
        }

    }
    else if(key==1)
    {
        n=0;
        m=0;
        chang_flag(key);
    }
    else
    {
        n=0;
        m=0;
        check_clock(key);           //报时&修改子程序
    }
    *P_Watchdog_Clear=0x0001;     //清看门狗
}
}
```

```
static unsigned int seccnt=0;
unsigned int Rem=0;           //温度返回变量
unsigned int _leap;         //闰年判断变量
```

```
void IRQ5(void) __attribute__((ISR));
```

```
void IRQ5(void)
{
    if(*P_INT_Ctrl&C_IRQ5_2Hz)
    {
```

```
seccnt++;
if(seccnt>=2)
{
    seccnt=0;
    if(second<59)
    {
        second++;
    }
    else
    {
        second=0;
        if(minute<59)
            minute++;
        else
        {
            minute=0;
            // PlayTime(); //整点报时
            if(hour<23)
                hour++;
            else
            {
                hour=0;
                if(week<7)
                    week++;
                else
                    week=1;
            }
        }
    }
}

if(month==1 || month==3 || month==5 || month==7 || month==8 || month==10 || month==12)
{
    if(day<31)
        day++;
    else
    {
        day=1;
        if(month<12)
            month++;
        else
        {
            month=1;
            if(year<9999)
                year++;
            else
            {

```

序

```
world_end(); //调世界末日子程

while(1)
{
    *P_Watchdog_Clear=1;
}

}

}

}
else if(month==4 | month==6 | month==9 | month==11)
{
    if(day<30)
        day++;
    else
    {
        day=1;
        month++;
    }
}
else
{
    _leap=leap_year(); //调用闰年子程序
    if(_leap==1)
    {
        if(day<29)
            day++;
        else
        {
            day=1;
            month++;
        }
    }
    else
    {
        if(day<28)
            day++;
        else
        {
            day++;
            month++;
        }
    }
}
```

```
        }
    }
}
display_1602();
Rem=Remind(); //调用提醒子程序, 每加
一分钟, 与定时比较
display(Rem); //显示
PlaySnd(27); //语音提醒
if(minute==0)
{
    //minute=0;
    PlayTime(); //整点报时
}
}
}

    *P_INT_Clear=C_IRQ5_2Hz;
}
else if(*P_INT_Ctrl&C_IRQ5_4Hz)
    *P_INT_Clear=C_IRQ5_4Hz;
}
```

```
//*****//
//语言播报中断
//*****//
__asm(".external F_FIQ_Service_SACM_A2000");
void FIQ(void) __attribute__((ISR));
void FIQ(void)
{
    if(*P_INT_Ctrl&0x2000)
    {
        *P_INT_Clear=C_FIQ_TMA;
        __asm("call F_FIQ_Service_SACM_A2000"); //A2000 终端服务程序
    }
    else if(*P_INT_Ctrl&0x0800)
```

```
*P_INT_Clear=C_FIQ_TMB;  
else  
*P_INT_Clear=C_FIQ_PWM;  
}
```