实时时钟（RTC）的主要功能是在系统掉电的情况下，利用后备电源使时钟继续运行，从而不会丢失时间信息。

s3c2440 内部集成了 RTC 模块，而且用起来也十分简单。其内部的寄存器 BCDSEC，BCDMIN，BCDHOUR，BCDDAY，BCDDATE，BCDMON 和 BCDYEAR 分别存储了当前的秒，分，小时，星期，日，月和年，表示时间的数值都是 BCD 码。这些寄存器的内容可读可写，并且只有在寄存器 RTCCON 的第 0 位为 1 时才能进行读写操作。为了防止误操作，当不进行读写时，要把该位清零。当读取这些寄存器时，能够获知当前的时间；当写入这些寄存器时，能够改变当前的时间。另外需要注意的是，因为有所谓的“一秒误差”，因此当读取到的秒为 0 时，需要重新再读取一遍这些寄存器的内容，才能保证时间的正确。

下面的程序就是一个简单的 RTC 的例子。系统上电，或按键时，会在 LCD 上显示当前时间的全部信息。并且可以通过 UART，在 PC 机上改变 RTC，它们之间的协议为：当 PC 机发送 0xAA 时，表示其后的 7 个数据分别为年、月、日、星期、小时、分和秒，在 s3c2440 接收到这些数据后，发送 0xAA 以示确认；当 PC 机发送其他数据时，对于 s3c2440 来说没有任何意义，s3c2440 只是把该数据再传回给 PC 机。为了简洁和突出重点，在下面程序中，会省略掉以前文章中介绍过的内容，如不清楚，请翻看本博客中的其他相关内容。

```
unsigned char date_buffer[7];                    //分别表示年、月、日、星期、小时、分、秒

unsigned char flag;              //用于表示更新 LCD 显示时间的标志


//获取时间

void get_date(void)

{

        rRTCCON = 1;


        date_buffer[0] = rBCDYEAR    ;              //年
```

```
        date_buffer[1] = rBCDMON   ;              //月
        date_buffer[2] = rBCDDATE   ;             //日
        date_buffer[3] = rBCDDAY   ;              //星期
        date_buffer[4] = rBCDHOUR   ;             //小时
        date_buffer[5] = rBCDMIN   ;              //分
        date_buffer[6] = rBCDSEC   ;              //秒
        //当秒为 0 时，重新读取
        if(date_buffer[6]==0)
        {
            date_buffer[0] = rBCDYEAR   ;         //年
            date_buffer[1] = rBCDMON   ;          //月
            date_buffer[2] = rBCDDATE   ;         //日
            date_buffer[3] = rBCDDAY   ;          //星期
            date_buffer[4] = rBCDHOUR   ;         //小时
            date_buffer[5] = rBCDMIN   ;          //分
            date_buffer[6] = rBCDSEC   ;          //秒
        }

        rRTCCON = 0;
}

//设置时间
```

```
void set_date(void)
{
        rRTCCON = 1 ;

        rBCDYEAR = date_buffer[0] ;                  //年
        rBCDMON    = date_buffer[1] ;                //月
        rBCDDATE = date_buffer[2] ;                   //日
        rBCDDAY    = date_buffer[3] ;                 //星期
        rBCDHOUR = date_buffer[4] ;                   //小时
        rBCDMIN    = date_buffer[5] ;                          //分
        rBCDSEC    = date_buffer[6] ;                 //秒

        rRTCCON = 0 ;
}

//在 LCD 上显示时间
void show_date(void)
{
        unsigned char String[ ]="当前时间：";
        unsigned char String_Y[ ]="年";
        unsigned char String_M[ ]="月";
        unsigned char String_D[ ]="日";
```

```
unsigned char ToWeek[ ][7] = {"星期日","星期一","星期二","星期三","星期四","星期五","星期六"};


unsigned char qh,wh;

const unsigned char *mould;

int length = sizeof(String);

int k,xx;


get_date();                    //获取时间


//在 LCD 上写“当前时间：”这几个字

for(k=0,xx=0;k<length-1;k++)

{

        qh=String[k]-0xa0;

        wh=String[k+1]-0xa0;

        mould = & __HZK[ (    ( qh - 1 )*94 + wh- 1 )*32 ];

        Draw_Text16(30+xx,50,0x0f,mould);

        xx+=16;

        k++;

}


//在下一行显示具体的时间，形式如：2010 年 04 月 03 日星期六  21:52:28
```

```
            //年
            //人为加上“20”
            mould = & __ASCII['2'*16];
Draw_ASCII(30,70,0x0,mould);
            mould = & __ASCII['0'*16];
Draw_ASCII(38,70,0x0,mould);

            qh=(date_buffer[0]>>4)+48;
wh=(date_buffer[0]&0x0f)+48;
mould = & __ASCII[qh*16];
            Draw_ASCII(46,70,0x0,mould);
mould = & __ASCII[wh*16];
            Draw_ASCII(54,70,0x0,mould);

            qh=String_Y[0]-0xa0;
            wh=String_Y[1]-0xa0;
            mould = & __HZK[ (    ( qh - 1 )*94 + wh- 1 )*32 ];
            Draw_Text16(62,70,0x0f,mould);

            //月
qh=(date_buffer[1]>>4)+48;
            wh=(date_buffer[1]&0x0f)+48;
```

```
mould = & __ASCII[qh*16];
            Draw_ASCII(78,70,0x0,mould);
mould = & __ASCII[wh*16];
            Draw_ASCII(86,70,0x0,mould);

            qh=String_M[0]-0xa0;
            wh=String_M[1]-0xa0;
            mould = & __HZK[ (    ( qh - 1 )*94 + wh- 1 )*32 ];
            Draw_Text16(94,70,0x0f,mould);

            //日
            qh=(date_buffer[2]>>4)+48;
wh=(date_buffer[2]&0x0f)+48;
            mould = & __ASCII[qh*16];
Draw_ASCII(110,70,0x0,mould);
            mould = & __ASCII[wh*16];
            Draw_ASCII(118,70,0x0,mould);

            qh=String_D[0]-0xa0;
            wh=String_D[1]-0xa0;
            mould = & __HZK[ (    ( qh - 1 )*94 + wh- 1 )*32 ];
            Draw_Text16(126,70,0x0f,mould);
```

```
        //星期
        for(k=0,xx=0;k<7-1;k++)
{
                qh=ToWeek[date_buffer[3]][k]-0xa0;
                wh=ToWeek[date_buffer[3]][k+1]-0xa0;
                mould = & __HZK[ (    ( qh - 1 )*94 + wh- 1 )*32 ];
                Draw_Text16(142+xx,70,0x0f,mould);
                xx+=16;
                k++;
}
//小时
        qh=(date_buffer[4]>>4)+48;
wh=(date_buffer[4]&0x0f)+48;
        mould = & __ASCII[qh*16];
Draw_ASCII(194,70,0x0,mould);
        mould = & __ASCII[wh*16];
Draw_ASCII(202,70,0x0,mould);

        mould = & __ASCII[ ':'*16 ];
        Draw_ASCII(210,70,0x0f,mould);
```

```
        //分
        qh=(date_buffer[5]>>4)+48;
wh=(date_buffer[5]&0x0f)+48;
        mould = & __ASCII[qh*16];
Draw_ASCII(218,70,0x0,mould);
mould = & __ASCII[wh*16];
        Draw_ASCII(226,70,0x0,mould);

        mould = & __ASCII[ ':'*16 ];
        Draw_ASCII(234,70,0x0f,mould);

        //秒
        qh=(date_buffer[6]>>4)+48;
wh=(date_buffer[6]&0x0f)+48;
        mould = & __ASCII[qh*16];
Draw_ASCII(242,70,0x0,mould);
        mould = & __ASCII[wh*16];
Draw_ASCII(250,70,0x0,mould);
}

//按键中断，用于更新 LCD 上显示的时间
void __irq Key4_ISR(void)
```

```
{
        rSRCPND = rSRCPND | 0x1;
        rINTPND = rINTPND | 0x1;
        flag=1;                 //置标志
}

//UART 中断，用于修改时间
void __irq uartISP(void)
{
        char ch;
        static char temp;
        static char count;
        rSUBSRCPND |= 0x3;
        rSRCPND |= 0x1<<28;
        rINTPND |= 0x1<<28;

        if(rUTRSTAT0 & 1)       //接收数据处理部分
        {

                ch = rURXH0;                    //接收字节数据
                if(ch==0xaa&&temp==0)       //表示接收到修改时间的命令
                {
```

```
            temp=1;
            count=0;
        }
        else if(temp==1)                //接收时间数据
        {
            date_buffer[count]=ch;          //依次存入时间数组内
            count++;
            if(count==7)                //7 个时间数据全部接收完毕
            {
                rUTXH0=0xaa;                //发送 0xAA
                set_date();                 //设置时间
                count=0;
                temp=0;
                flag=1;                 //更新 LCD 上显示的时间
            }
        }
        else
            rUTXH0=ch;              //不是接收时间的命令

    }
}
```

```
void Main(void)
{
//初始化寄存器，内容省略
……
        Brush_Background(0xffffff);             //LCD 背景为白色
        show_date();                //在 LCD 上显示时间

        flag=0;                 //清标志

        while(1)
        {
            if(flag)                //需要更新 LCD 上的显示时间
            {
            Brush_Background(0xffffff);
                show_date();
                flag=0;                 //清标志
            }
}
}
```