DS1302 中文资料

DS1302 是 DALLAS 公司推出的涓流充电时钟芯片内含有一个实时时钟/日历和 31 字节静态 RAM

可通过简单的串行接口与单片机进行通信

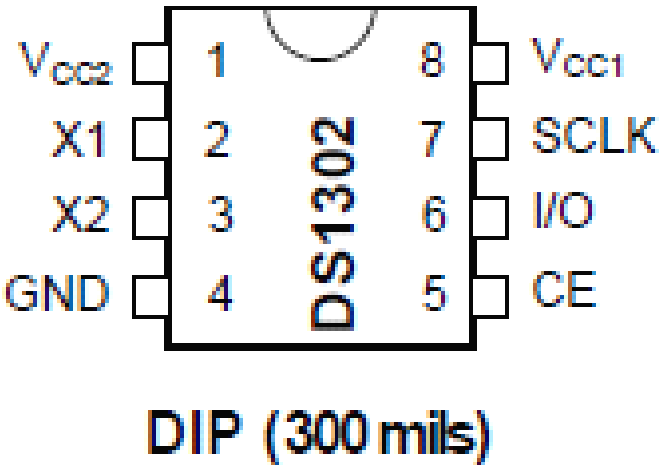
可提供:

--秒分时日日期月年的信息

--每月的天数和闰年的天数可自动调整

--可通过 AM/PM 指示决定采用 24 或 12 小时格式

--保持数据和时钟信息时功率小于 1mW



DIP (300 mils)

DS1302 引脚

X1 X2 32.768KHz 晶振管脚

GND 地

CE 复位脚

I/O 数据输入/输出引脚

SCLK 串行时钟

Vcc1,Vcc2 电源供电管脚

各引脚的功能为:

Vcc1: 主电源; Vcc2: 备份电源。当Vcc2>Vcc1+0.2V时, 由Vcc2向DS1302供电，当Vcc2< Vcc1时，由Vcc1向DS1302供电。

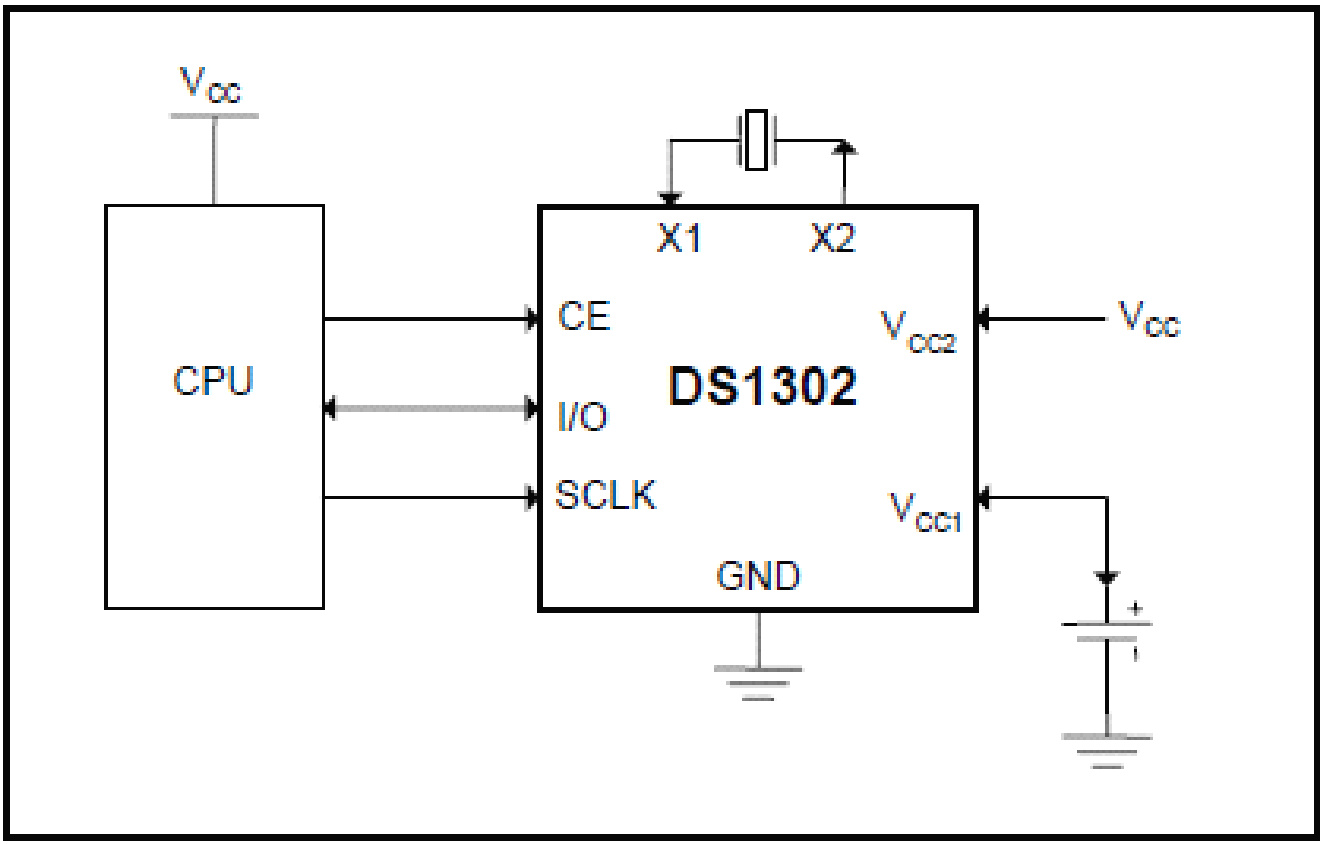
SCLK: 串行时钟，输入，控制数据的输入与输出;

I/O: 三线接口时的双向数据线;

CE: 输入信号，在读、写数据期间，必须为高。该引脚有两个功能:第一，CE开始控制字访问移位寄存器的控制逻辑；其次，CE 提供结束单字节或多字节数据传输的方法。

参考电路：

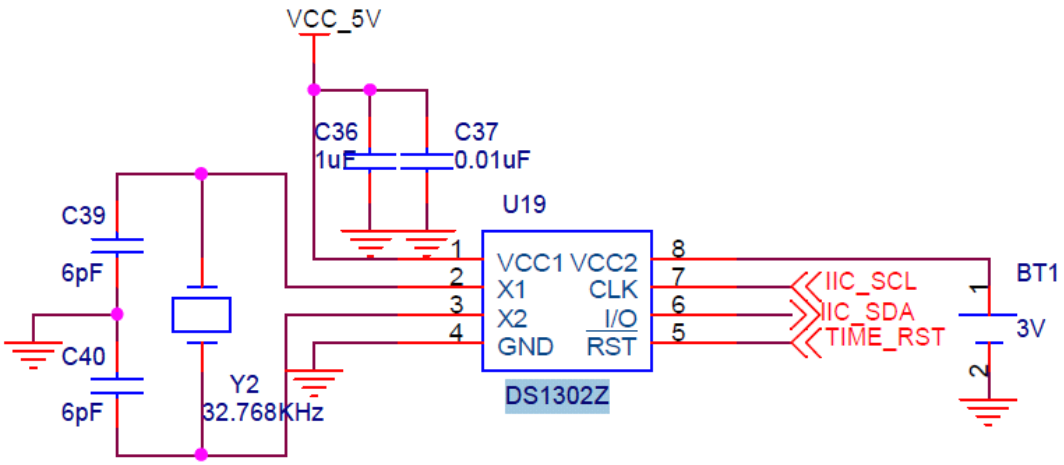
TYPICAL OPERATING CIRCUIT



如上图所示：

DS1302 与单片机的连接也仅需要 3 条线：CE 引脚、SCLK 串行时钟引脚、I/O 串行数据引脚，Vcc2 为备用电源，外接 32.768kHz 晶振，为芯片提供计时脉冲。

实际电路:

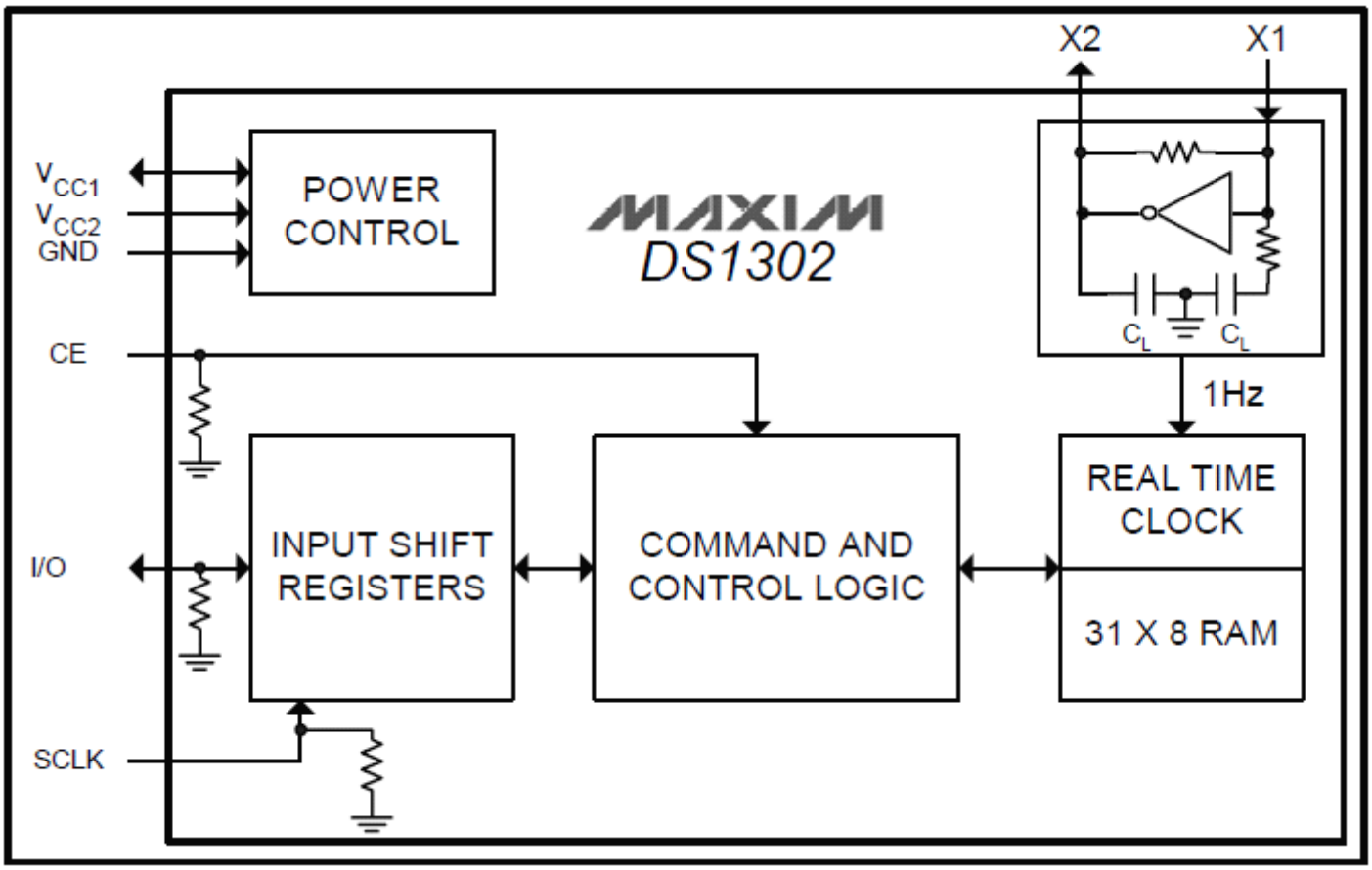


Vcc_5V：为电路中的主电源；Vcc2，也就是 BT1 为备份电源。当 Vcc2>Vcc1+0.2V 时，

由 Vcc2 向 DS1302 供电，当 Vcc2< Vcc1 时，由 Vcc1 向 DS1302 供电

CLK 和 I/O 虽然和 IIC 总线接在一条引脚上，但 DS1302 其实并不是使用 IIC 总线，而是一种三线式总线，

DS1302 内部结构：



DS1302 内部包括：

Power control:电源控制模块

Input shift registers：输入移位寄存器

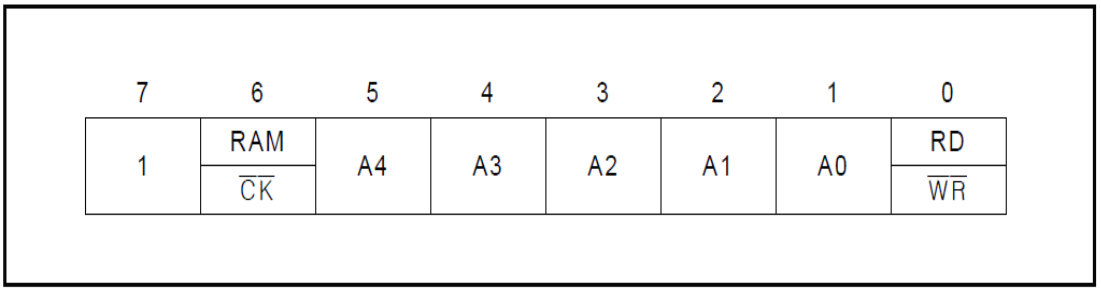
Command and control logic：通讯与逻辑控制器

Oscillator and divider：晶体振荡器及分频器

DS1302 的内部主要组成部分虽然有：移位寄存器、控制逻辑、振荡器、实时时钟以及 RAM。虽然数据分成两种，但是对单片机的程序而言，其实是一样的，就是对特定的地址进行读写操作。

DS1302 控制字：

Figure 3. Address/Command Byte



控制字的最高有效位（位7）必须是逻辑1，如果它为0，则不能把数据写入到DS1302中。

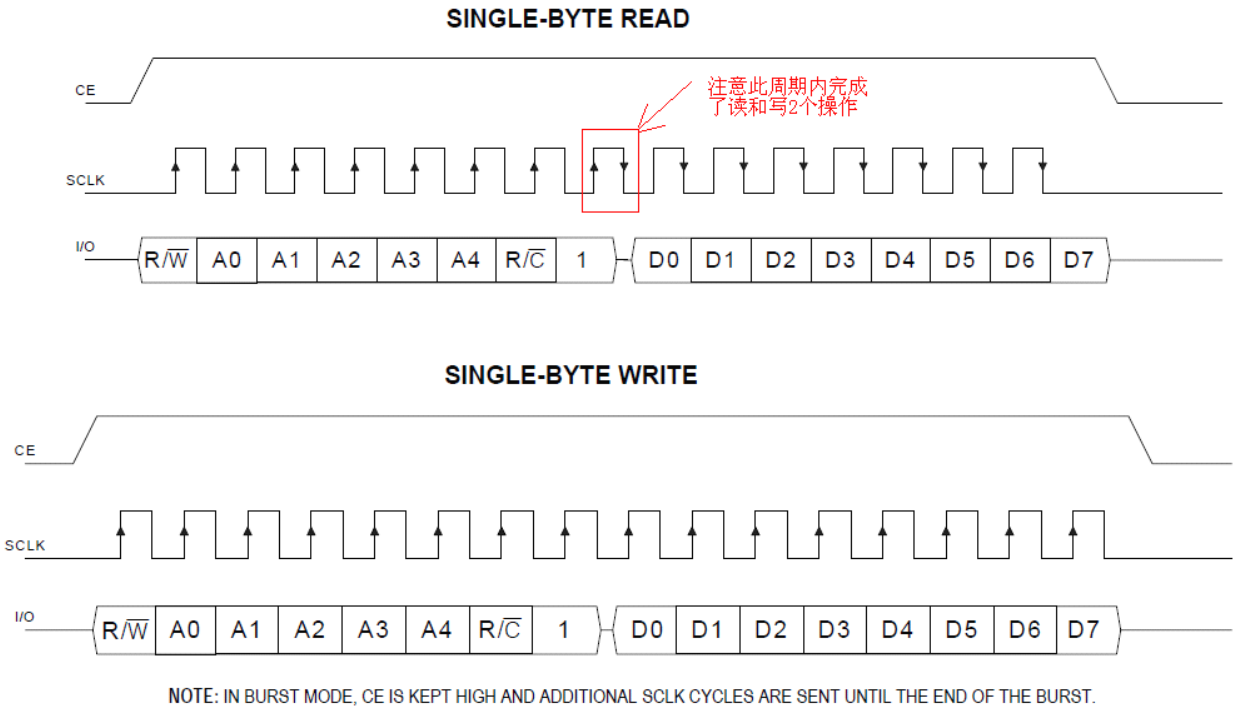
位6：如果为0，则表示存取日历时钟数据，为1表示存取RAM数据；

位5至位1（A4～A0）：指示操作单元的地址；

位0（最低有效位）：如为0，表示要进行写操作，为1表示进行读操作。

控制字总是从最低位开始输出。在控制字指令输入后的下一个 SCLK 时钟的上升沿时，数据被写入 DS1302，数据输入从最低位（0 位）开始。同样，在紧跟 8 位的控制字指令后的下一个 SCLK 脉冲的下降沿，读出 DS1302 的数据，读出的数据也是从最低位到最高位。

DS1302 时序：



如图，所示

CE 输入驱动高启动所有的数据传输。

CE 输入有两个功能。首先，CE 打开控制逻辑，允许访问的移位寄存器的地址/命令序列。其次，CE 提供了一个终止单字节或多字节数据传输方法。

一个时钟周期是由一个下降沿之后的上升沿序列。对于数据传输而言，数据必须在有效的时钟的上升沿输入，在时钟的下降沿输出。如果 CE 为低，所有的 I/O 引脚变为高阻抗状态，数据传输终止。

对于数据输入：

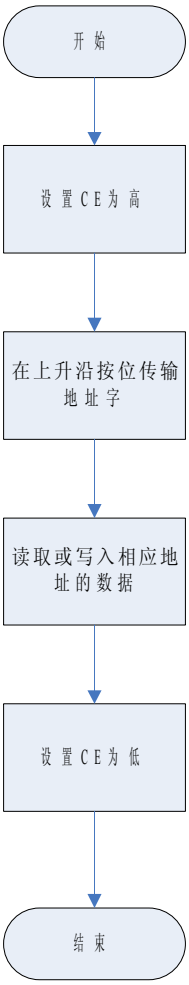
开始的 8 个 SCLK 周期，输入写命令字节，数据字节在后 8 个 SCLK 周期的

上升沿输入。数据输入位 0 开始。

对于数据输出：

开始的 8 个 SCLK 周期，输入一个读命令字节，数据字节在后 8 个 SCLK 周期的下降沿输出。注意，第一个数据字节的第一个下降沿发生后，命令字的最后一位被写入（Note that the first data bit to be transmitted occurs on the first falling edge after the last bit of the command byte is written.），命令字节的最后一位被写入。当 CE 仍为高时。如果还有额外的 SCLK 周期，DS1302 将重新发送数据字节，这使 DS1302 具有连续突发读取的能力。

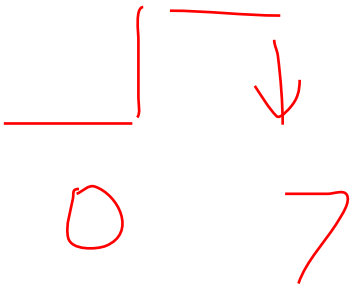
DS1302 驱动程序分析




```
/***************************************************
*名称: uchar DS1302Read()
*说明: 先写地址，后读数据
*功能: 从 cmd 相应地址中读取一个字节的数据
*调用: DS1302WriteByte(),DS1302ReadByte()
*输入: cmd:要写入的控制字节
*输出: dat:读取的数据
***************************************************/
uchar DS1302Read(uchar cmd)
{
    uchar dat;
    RST=0;//初始 CE 线置为 0
    SCLK=0;//初始时钟线置为 0
    RST=1;//初始 CE 置为 1，传输开始
    DS1302WriteByte(cmd);//传输命令字，要读取的时间/日历地址
    dat=DS1302ReadByte();//读取要得到的时间/日期
```

```
    SCLK=1;//时钟线拉高
    RST=0;//读取结束，CE 置为 0，结束数据的传输
    return dat;//返回得到的时间/日期
}
/*************************************************
*名称: DS1302Write
*说明: 先写地址，后写数据
*功能: 向 cmd 相应地址中写一个字节的数据
*调用: DS1302WriteByte()
*输入: cmd:要写入的控制字,dat:要写入的数据
*输出: 无
*************************************************/
void DS1302Write(uchar cmd, uchar dat)
{
    RST=0; //初始 CE 线置为 0
    SCLK=0; //初始时钟线置为 0
    RST=1; //初始 CE 置为 1，传输开始
    DS1302WriteByte(cmd); /传输命令字，要写入的时间/日历地址
    DS1302WriteByte(dat); //写入要修改的时间/日期
    SCLK=1; //时钟线拉高
    RST=0; //读取结束，CE 置为 0，结束数据的传输
}
/*************************************************
*名称: DS1302WriteByte
*说明: 无
*功能: 写入 8bit 数据
*调用: delayus()
*输入: dat:要写入的数据
*输出: 无
*************************************************/
void DS1302WriteByte(uchar dat)
{
    uchar i;
    SCLK=0;//初始时钟线置为 0
    delayus(2);
    for(i=0;i<8;i++)//开始传输 8 个字节的数据
     {
        SDA=dat&0x01;//取最低位，注意 DS1302 的数据和地址都是从最低位开始传输的
        delayus(2);
        SCLK=1;//时钟线拉高，制造上升沿，SDA 的数据被传输
        delayus(2);
        SCLK=0;//时钟线拉低，为下一个上升沿做准备
        dat>>=1;//数据右移一位，准备传输下一位数据
     }
```

```
}
/*****************************************************
*名称: DS1302ReadByte()
*说明: 无
*功能: 读取 8bit 的数据
*调用: delayus()
*输入: 无
*输出: dat:读取的数据
*****************************************************/
uchar DS1302ReadByte()
{
    uchar i,dat;
    delayus(2);
    for(i=0;i<8;i++)
     {
        dat>>=1;//要返回的数据右移一位
        if(SDA==1)//当数据线为高时，证明该位数据为 1
          dat|=0x80;//要传输数据的当前值置为 1,若不是,则为 0
        SCLK=1;//拉高时钟线
        delayus(2);
        SCLK=0;//制造下降沿
       delayus(2);
     }
    return dat;//返回读取出的数据
}
```



寄存器和 RAM

对 DS1302 的操作就是对其内部寄存器的操作，DS1302 内部共有 12 个寄存器，其中有：

7 个寄存器与日历、时钟相关，存放的数据位为 BCD 码形式。

此外，DS1302 还有年份寄存器、控制寄存器、充电寄存器、时钟突发寄存器及与 RAM 相关的寄存器等。

时钟突发寄存器可一次性顺序读写除充电寄存器以外的寄存器。

如下所示：

DS1302 数据地址和传输格式

RTC

READ	WRITE	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	RANGE
81h	80h	CH	10 Seconds			Seconds				00–59
83h	82h		10 Minutes			Minutes				00–59
85h	84h	12/$\overline{24}$	0	10 $\overline{\text{AM/PM}}$	Hour	Hour				1–12/0–23
87h	86h	0	0	10 Date		Date				1–31
89h	88h	0	0	0	10 Month	Month				1–12
8Bh	8Ah	0	0	0	0	0	Day			1–7
8Dh	8Ch	10 Year				Year				00–99
8Fh	8Eh	WP	0	0	0	0	0	0	0	—
91h	90h	TCS	TCS	TCS	TCS	DS	DS	RS	RS	—

如图所示，时钟日历包含在 7 个读/写寄存器内，读/写寄存器中的数据是二——十进制的 BCD 码。

秒寄存器的 BIT7 定义为时间暂停位，当 BIT1 为 1 时，时钟振荡器停止工作，DS1302 进入低功耗模式，电源消耗小于 100 微安，当 BIT1 为 0 时，时钟振荡器启动，DS1302 正常工作。

小时寄存器的 BIT7 定义为 12 或 24 小时工作模式选择位，当 BIT7 为高时，

为 12 小时工作模式，此时 BIT5 为 AM/PM 位，低电平标示 AM，高电平标示 PM，在 24 小时模式下，BIT5 为第二个 10 小时位标示(20~23 时)。

写保护寄存器的 BIT7：WP 是写保护位，工作时，出 WP 外的其他位都置为 0，对时钟/日历寄存器或 RAM 进行写操作之前，WP 必须为 0，当 WP 为高电平的时候，不能对任何时钟/日历寄存器或 RAM 进行写操作。

关于突发模式（burst mode 或称多字节传输模式），突发模式可以指定任何的时钟/日历或者 RAM 寄存器为突发模式，和以前一样，第 6 位指定时钟或 RAM 而 0 位指定读或写。突发模式的实质是指一次传送多个字节的时钟信号和 RAM 数据。如下图所示

工作模式寄存器		读寄存器	写寄存器
时钟突发模式寄存器	CLOCK BURST	BFh	BEh
RAM 突发模式寄存器	RAM BURST	FFh	FEh

在时钟/日历寄存器中的 9 至 31 和在 RAM 寄存器的地址 31 不能存储数据。突发模式的读取或写入从地址的位 0 开始。

如下程序所示:

```
/*-------------------------------------------------------------------------------
函数名称: DS1302_NReadRam(unsigned char *rstr)
函数功能: 多字节突发模式读 RAM,
      DS1302_NRRAM 一次可进行31个片内 RAM 单元读
输入参数: *rstr：存放读到的 N 个数据
输出参数:       无
-------------------------------------------------------------------------------*/
void DS1302_burst_Read(unsigned char *rstr)
{
      unsigned char i;
      RST=0;
      SCLK=0;
      RST=1;     //CE 拉高，传输开始
      DS1302_WriteByte(0XFF);     // 写0XFF，多字节突发方式读 RAM 具体细节见上一节
```

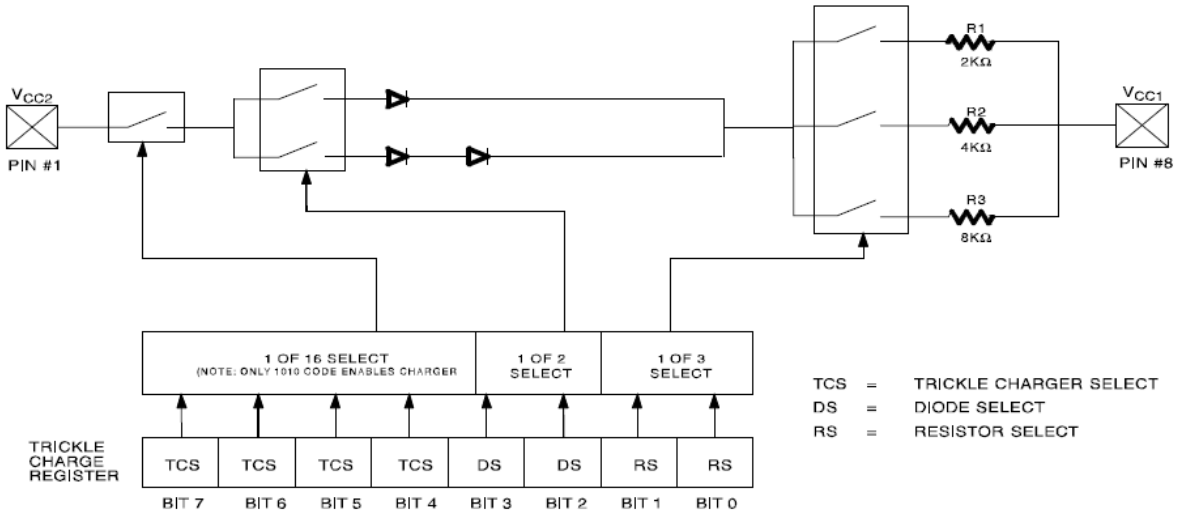
```
    for(i=0;i<31;i++)       //连续读取个31 字节
    {
        *rstr=DS1302ReadByte(address) ;//此处的 ADDRESS 指的是你需要进行连续读取的地址
        rstr++;
    }
    RST=0;//CE 信号拉低，传输结束
    CLK=1;
}

/*-------------------------------------------------------------------------------
函数名称: DS1302_NReadRam(unsigned char *rstr)
函数功能: 多字节突发模式写 RAM，
    DS1302_NRRAM 一次可进行 31 个片内 RAM 单元写
 输入参数: *wstr：要被写入的 N 个数据
输出参数：  无
-------------------------------------------------------------------------------*/

 void DS1302_burstWrite(unsigned char *wstr)
{
    unsigned char i;
    unsigned char *tmpstr;
    tmpstr=wstr ;
    DS1302Write(0x8e,0x00);//写保护关
    RST=0;
    SCLK=0;
    RST=1;
    DS1302_WriteByte(0XFE);       // 写  0XFE ， 多 字 节 突 发 方 式 写 RAM, 具 体 细 节 见 上 节
    for(i=0;i<31;i++)       //连续写入31 字节
    {
        DS1302_WriteByte(*tmpstr) ;
        tmpstr++;
    }
    RST=0;
    SCLK=1;
    DS1302Write(0x8e,0x80);     //开写保护
}
```

91h	90h	TCS	TCS	TCS	TCS	DS	DS	RS	RS	—

此寄存器为 DS1302 充电模式控制位，结构如下所示



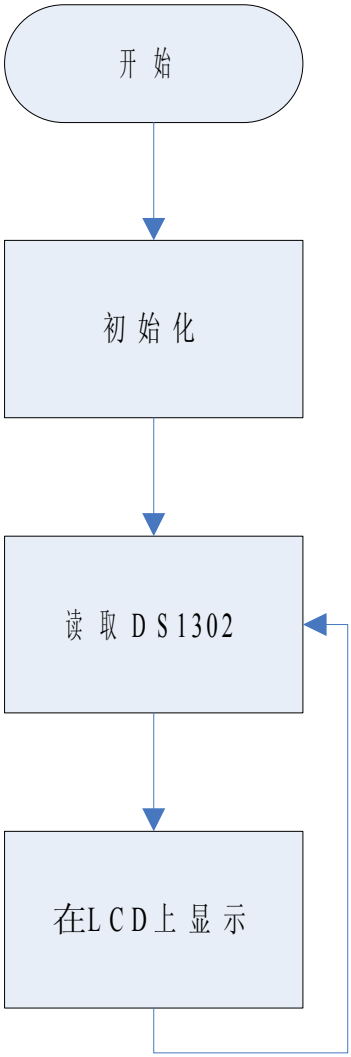
TCS 位：涓流充电选择位（BIT4 至 7）控制涓流充电器的选择。为防止偶然的因素使其工作，只有 1010 模式才能使涓流充电器工作。所有其它的模式将禁止涓流充电。在 DS1302 上电后。涓流充电将被禁止。

DS 位:该二极管选择（BIT2 和 3）选择是一个二极管还是两个二极管之间在 Vcc2 和 Vcc1 连接。如果 DS 为 01，则选择一个二极管。如果 DS 为 10，则两个二极管被选中。如果 DS 为 00 或 11，充电器被禁止，与 TCS 无关。

RS 位（BIT0 和 1）:选择是在 Vcc2 和 Vcc1 之间的连接电阻。电阻的选择如下所示：

00	无	无
01	R1	2kΩ
10	R2	4kΩ
11	R3	8kΩ

程序流程设计



几个数组，用来定义要显示的汉字信息

unsigned char Chinese_1[17] = {0xc4,0xea, 0xd4,0xc2, 0xc8,0xd5, 0xca,0xb1, 0xb7,0xd6, 0xc3,0xeb, 0xd0,0xc7, 0xc6,0xda, ' '};//年 月 日 时 分 秒 星 期

unsigned char Chinese_2[15] = {0xd2,0xbb, 0xb6,0xfe, 0xc8,0xfd, 0xcb,0xc4, 0xce,0xe5, 0xc1,0xf9, 0xcc,0xec, ' '};//一二三四五六天

unsigned char Cursor[6] ={0x81,0x83,0x85,0x90,0x92,0x94};//定义光标移动的位置

主函数部分

主函数部分只需要按照流程图的需要调用各个模块即可

```
/*************************************************
*名称: void main()
*说明: 无
*功能: 读取 8bit 的数据
*调用: delayms()
*输入: 无
*输出: dat:读取的数据
*************************************************/
void main()
{
    Delayms(50);
    EX1 = 1;//外部中断开
    EA = 1;   //全局中断开
    Initialize_device();//初始化 CH452 LCD DS1302 等器件
    while(1)//while 中的值只循环显示小时，分钟，秒这几个常变量，有其他需要，可以继续添加
    {
            Display_Clock();//显示你需要从 DS1302 中读取的值
            Delayms(500);
      }
    };
}
```

DS1302 的初始化部分

只需要调用一次，写入一个初始值即可

```
/*************************************************
*名称: void Init_DS1302(void)
*说明: 给 1302 写入一个初始的值
*功能: 写入日期，和时钟的值
*调用: DS1302Write（）
*输入: 无
*输出: 无
```

```
*************************************************/
void Init_DS1302(void)
{
    DS1302Write(0x8e,0x00);//写保护关
    DS1302Write(DS1302_SECOND_WRITE,0x00);   //初始秒值为 0
    DS1302Write(DS1302_MINUTE_WRITE,0x20);//初始分钟值为 0
    DS1302Write(DS1302_HOUR_WRITE,0x09); //初始为 24 小时模式 初始时间为 0 点
    DS1302Write(DS1302_DAY_WRITE,0x25);   //2011 年 1 月 1 日 星期 6
    DS1302Write(DS1302_MONTH_WRITE,0x12);
    DS1302Write(DS1302_YEAR_WRITE,0x10);
    DS1302Write(DS1302_WEEK_WRITE,0x06);
    DS1302Write(0x90,0x01);                  //充电
    DS1302Write(0xc0,0xf0);                  //初始化一次标示
    DS1302Write(0x8e,0x80);
}
/*************************************************
*名称: Initialize_device()
*说明: 无
*功能: 初始化实验用到的器件
*调用: delayms()， Init_1602();Dispaly_Menu();Init8259a();
*输入: 无
*输出:无
*************************************************/
Void Initialize_device()
{
    CH452_Write(CH452_SYSON2);//初始化 CH452
    Init_1602();                 //初始化 LCD
    Init8259a();                 //初始化 8259a
    Dispaly_Menu();//显示年月日等文字
    Delayms(50);
//  Init_DS1302();//初始化 DS1302，只需要执行一次，设定好后，不需要再次执行
}
```

由于 DS1302 的很多数值都是十位数，所以需要把十位数分成 2 次在 LCD12864 上显示

```
//-------------------------------------------------
//                  函数名称： void Split_display()
//                  函数功能：把一个十位数分二次显示至传递来的位置
//                  入口参数： unsigned char address
//                  出口参数：无
//-------------------------------------------------
void Split_display(unsigned char address)
```

```
{
    unsigned char i;
    i = DS1302Read(address);//读取  十位
    i = i/16 + '0';
    Write_data(i);//显示十位
    Delayms(3);
    i = DS1302Read(address);//读取  个位
    i = i%16 + '0';
    Write_data(i);//显示个位
    Delayms(5);
}
```

屏幕显示部分

显示屏幕上的文字信息

显示效果为：

20XX 年 XX 月 XX 日
XX 时 XX 分 XX 秒
星期 X

```
//-------------------------------------------------
//                      函数名称：void Dispaly_Menu()
//                      函数功能：显示屏幕上的所有元素
//                      入口参数：无
//                      出口参数：无
//-------------------------------------------------
void Dispaly_Menu()
{
    Write_com(0x01);
    Delayms(100);
    Write_com(0x80);
    Delayms(100);
    Write_data('2');
    Delayms(5);
    Write_data('0');
    Delayms(1);


    Split_display(DS1302_YEAR_READ);// 显示年份
    Write_data(Chinese_1[0]);//显示“年”
    Delayms(3);
    Write_data(Chinese_1[1]);
    Delayms(3);
```

```
Split_display(DS1302_MONTH_READ);//显示月份
Write_data(Chinese_1[2]);//显示“月”
Delayms(3);
Write_data(Chinese_1[3]);
Delayms(3);


Split_display(DS1302_DAY_READ);//显示日期
Write_data(Chinese_1[4]);//显示“日”
Delayms(3);
Write_data(Chinese_1[5]);
Delayms(3);
Write_data(Chinese_1[16]);
Delayms(3);


Write_com(0x90); //换第二行
Split_display(DS1302_HOUR_READ);//显示小时
Write_data(Chinese_1[6]);//显示“时”
Delayms(3);
Write_data(Chinese_1[7]);
Delayms(3);

Split_display(DS1302_MINUTE_READ);//显示分钟
Write_data(Chinese_1[8]);//显示“分”
Delayms(3);
Write_data(Chinese_1[9]);
Delayms(3);//


c = DS1302Read(DS1302_SECOND_READ);//读取  秒的十位
c = c&0x7f;
c = c/16 + '0';
Write_data(c);//显示秒的十位
Delayms(3);
c = DS1302Read(DS1302_SECOND_READ);//读取  秒的个位
c = c%16 + '0';
Write_data(c);
Delayms(5);//显示秒
Write_data(Chinese_1[10]);//显示“秒”
Delayms(3);
Write_data(Chinese_1[11]);
Delayms(3);
Write_data(Chinese_1[16]);
```

```
    Delayms(3);


    Write_com(0x88);//换第三行
                    //显示"星期"
    Write_data(Chinese_1[12]);
    Delayms(3);
    Write_data(Chinese_1[13]);
    Delayms(3);
    Write_data(Chinese_1[14]);
    Delayms(3);
    Write_data(Chinese_1[15]);
    Delayms(3);
    Write_data(Chinese_2[(DS1302Read(0x8b)-1)*2]);
    Delayms(3);
    Write_data(Chinese_2[(DS1302Read(0x8b)-1)*2+1]);
    Delayms(3);
    Write_data(Chinese_1[16]);
    Delayms(3);


    Write_com(0x81);
  Delayms(5000);
}

/*************************************************
*名称: void Display_Clock();
*说明: 无
*功能: 显示屏幕上改变的数值
*调用: delayms()，Write_com();
*输入: 无
*输出: dat:读取的数据
*************************************************/
Void Display_Clock()
{
        Write_com(Cursor[3]);//小时的位置，再次紧紧列出了几个最长改变的时间变量，如果有需要，可以另外加上其他的变量，如年，月
        Split_display(DS1302_HOUR_READ);//显示小时
        Delayms(5);//

        Write_com(Cursor[4]); //分钟的位置
        Delayms(3);
        Split_display(DS1302_MINUTE_READ);//显示分钟
        Delayms(5);//
```

```
        Write_com(Cursor[5]); //秒的位置
        c = DS1302Read(DS1302_SECOND_READ);//读取  秒的十位
        c = c&0x7f;
        c = c/16 + '0';
        Write_data(c);//显示秒的十位
        Delayms(3);
        c = DS1302Read(DS1302_SECOND_READ);//读取  秒的个位
        c = c%16 + '0';
        Write_data(c);//显示秒
}
```

思考题：

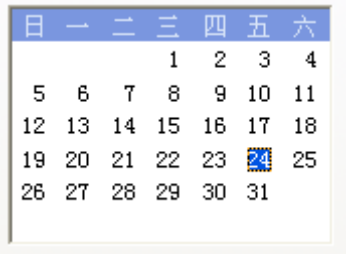
1.如何用 4*4 键盘动态改变 DS1302 中的时间/日历?

提示：

```
void Inter8259a() interrupt 2
{
//用于检验最后结果的位数
    uchar int_numb;
    int_numb = ADR_8259AE;
    int_numb = ADR_8259AE;
    #ifdef MODE8085
    int_numb = ADR_8259AE;
    #endif
    int_numb = int_numb & 0x07;
    switch(int_numb)
    {
        case 2:
            KeyBuff = CH452_Read() - 0x40; //从CH452的 DIG和SIG中读取按键编码 CH452 所提供的实际按键代码是读取的按键编码加上40H
            KeyBuff = Key_Val_Tab[3-(KeyBuff/8)][3-(KeyBuff%4)]; //把按键编码转换为数组
            if(KeyBuff=='0')
            {
                switch(flag)
                {
                    case 0:
                        flag = 1;//结束时间修改模式
                        Write_com(0x0c);//  使能设置 关闭光标闪烁
                        DS1302Write(DS1302_SECOND_WRITE, 0x00);//DS1302结束暂停模式
                        break;
                    case 1:
                        Write_com(0x0f);//改变为光标显示模式 显示游标 并游标位置反白显示
                        flag = 0;//改变标示位
                        Delayms(50);
                        DS1302Write(DS1302_SECOND_WRITE, 0x80);//DS1302进入暂停模式
                        break;
                     default :
                         break;
                }
            }

            if(flag==0)
            {
                    Change_Location();//检测要修改的时间的位置
                    Change_time();    //根据按键修改时间
            }
                break;
             break;
          default :
              break;
```

2.如何显示日历式日期，如：



日	一	二	三	四	五	六
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

提示：

```
//-----------------------------------------------
//                    函数名称：int WeekDay(int year, int month, int day)
//                    函数功能：由日期计算今天为星期几
//                    入口参数：int year, int month, int day
//                    出口参数：(int)sum%7
//-----------------------------------------------

int WeekDay(int year, int month, int day) //根据输入的日期，返回对应的星期
{
    int i;
    int run=0, ping=0;
    long sum;
    for(i=1; i {
    if(i%4==0 && i%100!=0 || i%400==0)
    run++;
    else
    ping++;
}
    sum = 366*run + 365*ping;
    for(i=1; i sum += MonthDays(year, i);
    sum =sum + day; //计算总天数
    return (int)sum%7;
}
  //-----------------------------------------------
//                    函数名称：int MonthDays(int year, int month)
//                    函数功能：根据输入的年号和月份，返回该月的天数
//                    入口参数：int year, int month, int day
//                    出口参数：(int)sum%7
//-----------------------------------------------

int MonthDays(int year, int month)//
{
switch(month)
{
```

```
        case 1:
        case 3:
        case 5:
        case 7:
        case 8:
        case 10:
        case 12: return 31; //一三五七八十腊，三十一天永不差
        case 4:
        case 6:
        case 9:
        case 11: return 30;//其他月份自然是30了
        case 2://计算闰月
                    if(year%4==0 && year%100!=0 || year%400==0)
                  return 29;
                  else
                  return 28;
        default:
              Return 0;
}
```