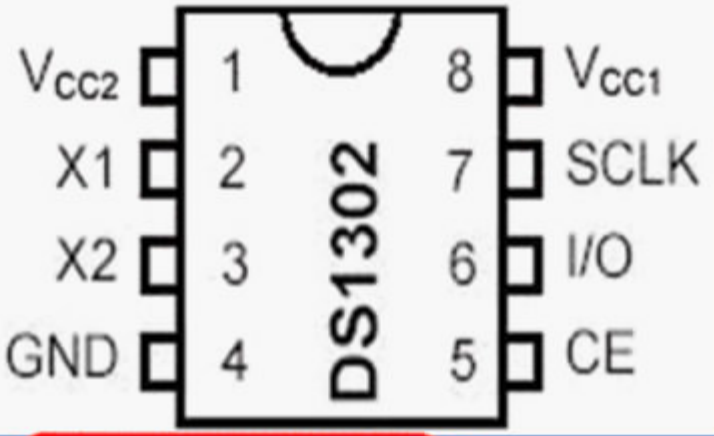
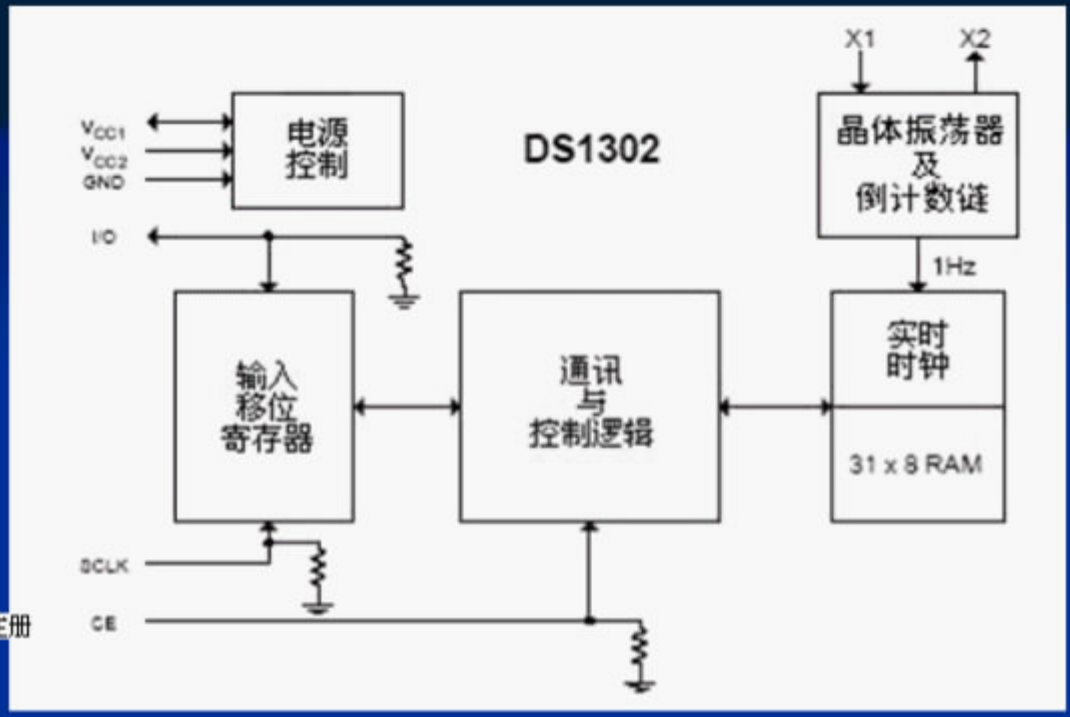
DS1302时钟芯片

DS1302是美国DALLAS公司推出的一种高性能、低功耗的实时时钟芯片，附加31字节静态RAM，采用SPI三线接口与CPU进行通信，并可采用突发方式一次传送多个字节的时钟信号和RAM数据。实时时钟可提供秒、分、时、日、星期、月和年，一个月小与31天时可以自动调整，且具有闰年补偿功能。工作电压宽达2.5～5.5V。采用双电源供电（主电源和备用电源），可设置备用电源充电方式，提供了对后备电源进行涓细电流充电的能力。



屏幕录像专家 未注册

DS1302引脚分配图



各引脚的功能为：

8 、Vcc1：备用电池端；

1、Vcc2：5V电源。当Vcc2>Vcc1+0.2V时，

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

7、 SCLK：串行时钟，输入；

6、I/O：数据输入输出口；

5、CE/RST：复位脚

2 3、X1、X2是外接晶振脚 （32.768KHZ的晶振）

4 地（GND）

DS1302有关日历、时间的寄存器

读寄存器	写寄存器	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	范围
81h	80h	CH	10秒			秒				00–59
83h	82h		10分			分				00–59
85h	84h	12/$\overline{24}$	0	10 AM/PM	时	时				1–12/0–23
87h	86h	0	0	10日		日				1–31
89h	88h	0	0	0	10月	月				1–12
8Bh	8Ah	0	0	0	0	0	周日			1–7
8Dh	8Ch	10年				年				00–99
8Fh	8Eh	WP	0	0	0	0	0	0	0	—

屏幕录像专家 未注册

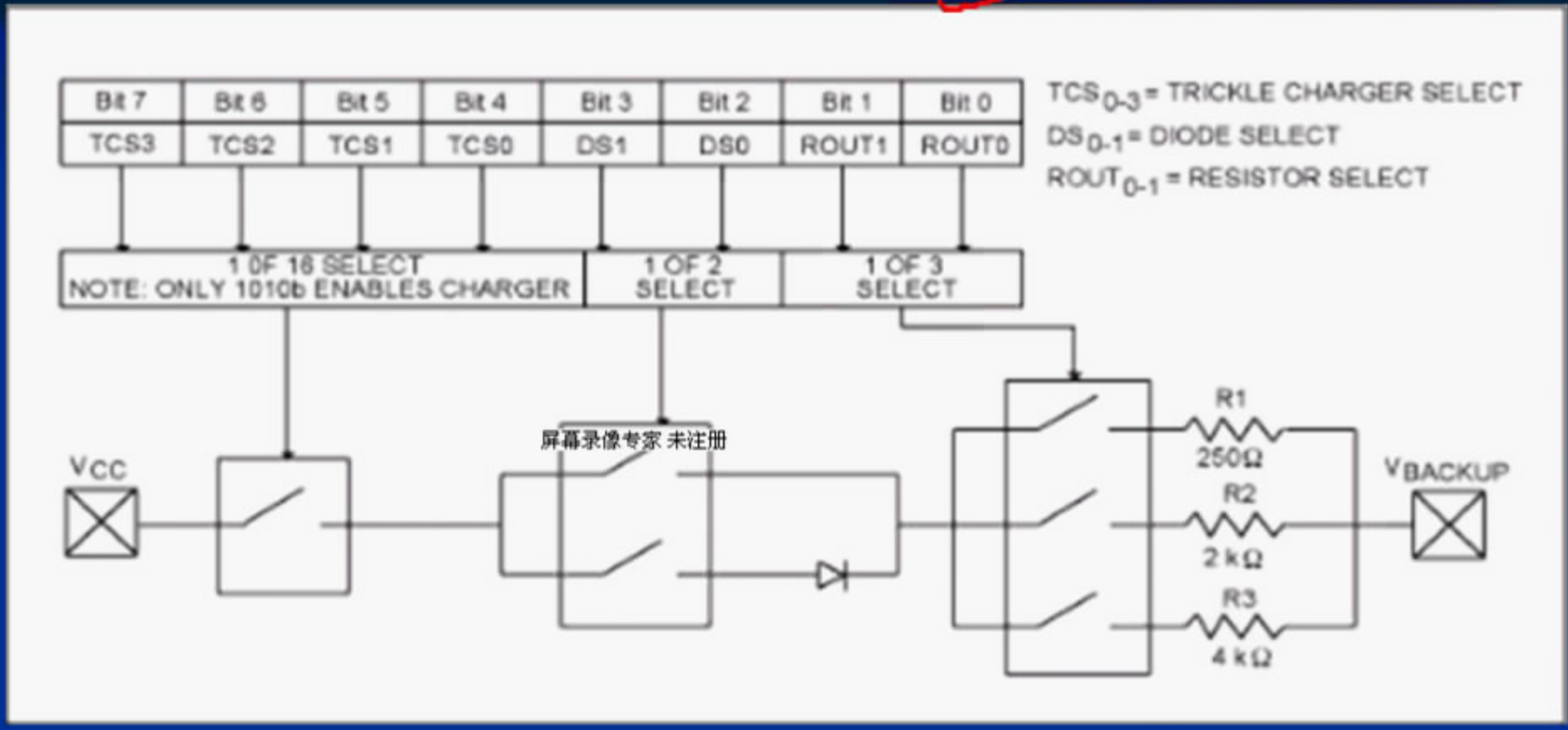
这张表是DS1302内部的7个与时间、日期有关的寄存器图和一个写保护寄存器，
我们要做的就是将初始设置的时间、日期数据写入这几个寄存器，然后再不断地读取
这几个寄存器来获取实时时间和日期。这几个寄存器的说明如下：

1、秒寄存器（81h、80h）的位7定义为时钟暂停标志（CH）。当初始上电时该位置为1，
时钟振荡器停止，DS1302处于低功耗状态；只有将秒寄存器的该位置改写为0时，时钟才能开始运行。

2、小时寄存器（85h、84h）的位7用于定义DS1302是运行于12小时模式还是24小时模式。当为高时，选择12小时模式。在12小时模式时，位5是 ，当为1时，表示PM。在24小时模式时，
位5是第二个10小时位

3、控制寄存器（8Fh、8Eh）的位7是写保护位（WP），其它7位均置为0。在对任何的时钟和RAM的写操作之前，WP位必须为0。当WP位为1时，写保护位防止对任一寄存器的写操作。
也就是说在电路上电的初始态WP是1，这时是不能改写上面任何一个时间寄存器的，
只有首先将WP改写为0，才能进行其它寄存器的写操作。

DS1302充电寄存器　寄存器地址是08H　读操作是91H 写操作是90H



2-3位:　DS=01 为一个二极管
DS=10 为二个二极管
DS=00 无二极管,不能充电
DS=11 无二极管,不能充电

0-1位:　RS=00 无电阻,不能充电
RS=01 2K
RS=10 4K
RS=11 8K

4-7位:　TCS=1010 能充电
其它组合都不能充电

10100101　一个二极管　电阻2K
10100110　一个二极管　电阻4K
10100111　一个二极管　电阻8K
10101001　两个二极管　电阻2K
10101010　两个二极管　电阻4K
10101011　两个二极管　电阻8K
01011100　初始化电源

DS1302读写时序

DS1302是SPI总线驱动方式。它不仅要向寄存器写入控制字，
还需要读取相应寄存器的数据。
要想与DS1302通信，首先要先了解DS1302的控制字。DS1302的控制字如图

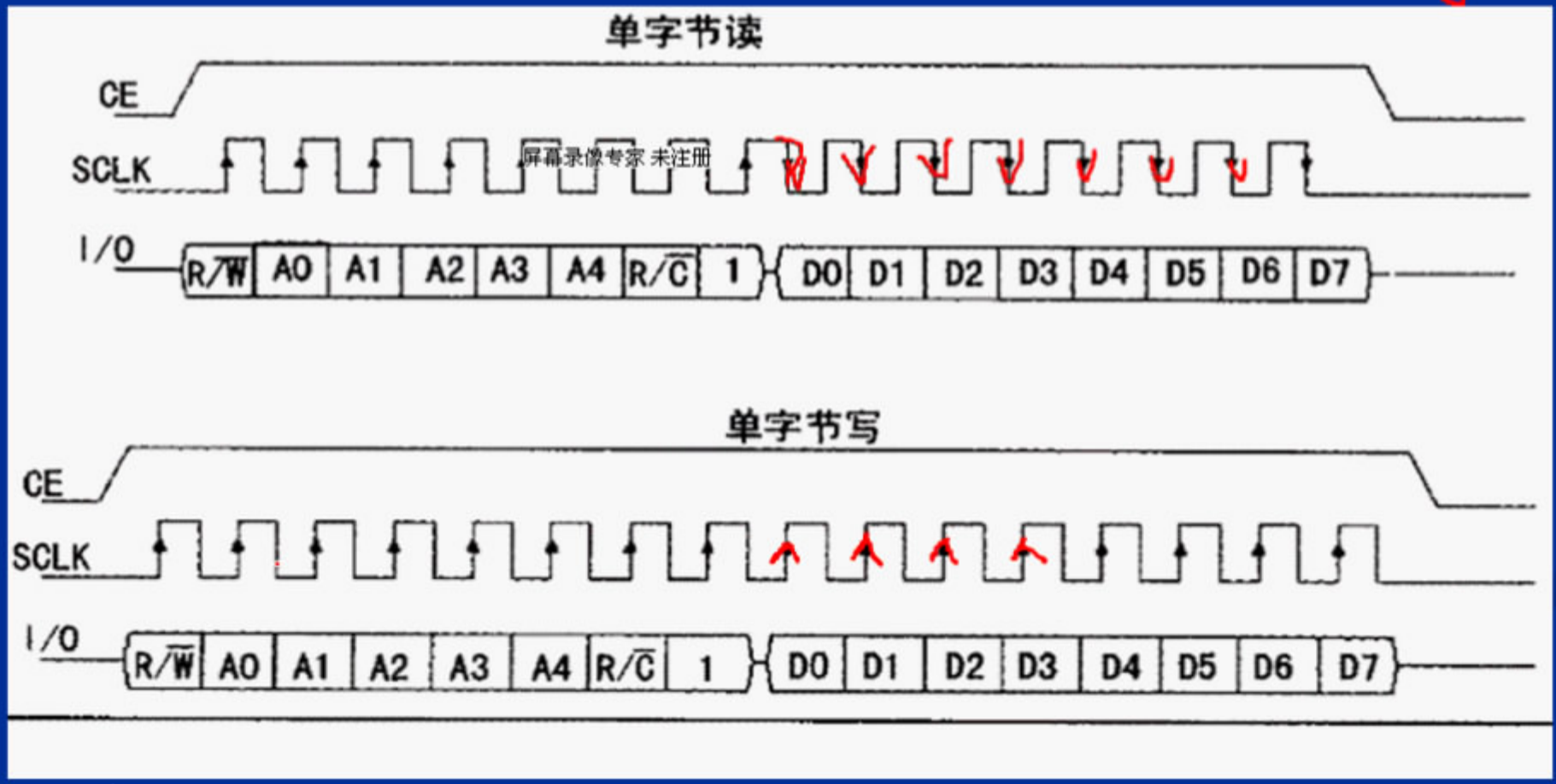
7	6	5	4	3	2	1	0
1	RAM $\overline{CK}$	A4	A3	A2	A1	A0	RD $\overline{WR}$

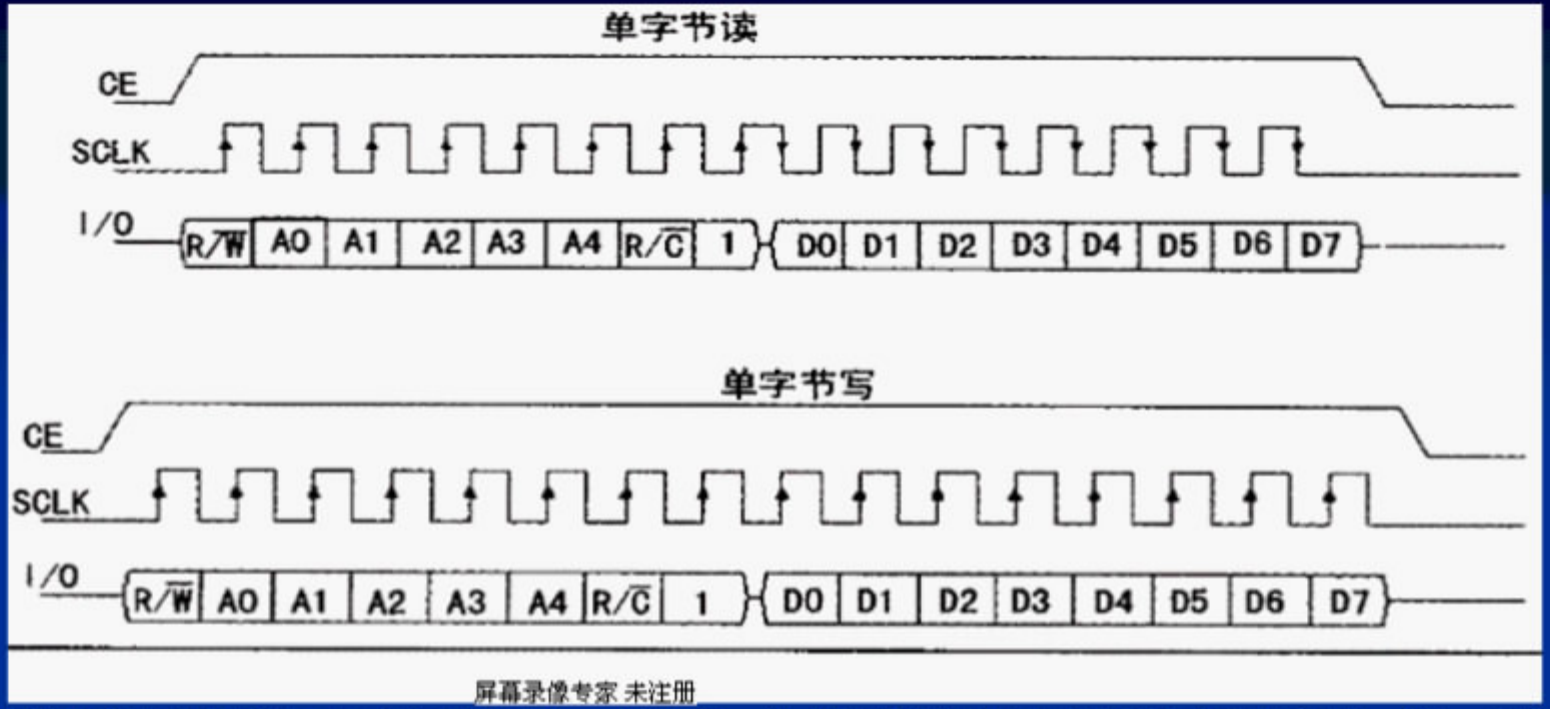
屏幕录像专家 未注册

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。
读数据：
读数据时在紧跟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，
读出的数据是从最低位到最高位。
写数据：
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，数据输入也是从最低位（0位）开始。

位0（最低有效位）：为1表示进行读操作。 如为0，表示要进行写操作，
控制字后 SCLK下降沿 读数据 SCLK上升沿写数据
数据是低位在前 高位在后

7	6	5	4	3	2	1	0
1	RAM $\overline{CK}$	A4	A3	A2	A1	A0	RD $\overline{WR}$





屏幕录像专家 未注册

DS1302的数据读写是通过I/O串行进行的。当进行一次读写操作时最少得读写两个字节，第一个字节是控制字节，就是一个命令，告诉DS1302是读还是写操作，是对RAM还是对CLOK寄存器操作，以及操作的址。第二个字节就是要读或写的数据了。我们先看

单字节写：在进行操作之前先得将CE（也可说是RST）置高电平，然后单片机将控制字的位0放到I/O上，当I/O的数据稳定后，将SCLK置高电平，DS1302检测到SCLK的上升沿后就将I/O上的数据读取，然后单片机将SCLK置为低电平，再将控制字的位1放到I/O上，如此反复，将一个字节控制字的8个位传给DS1302。接下来就是传一个字节的数据给DS1302，当传完数据后，单片机将CE置为低电平，操作结束。

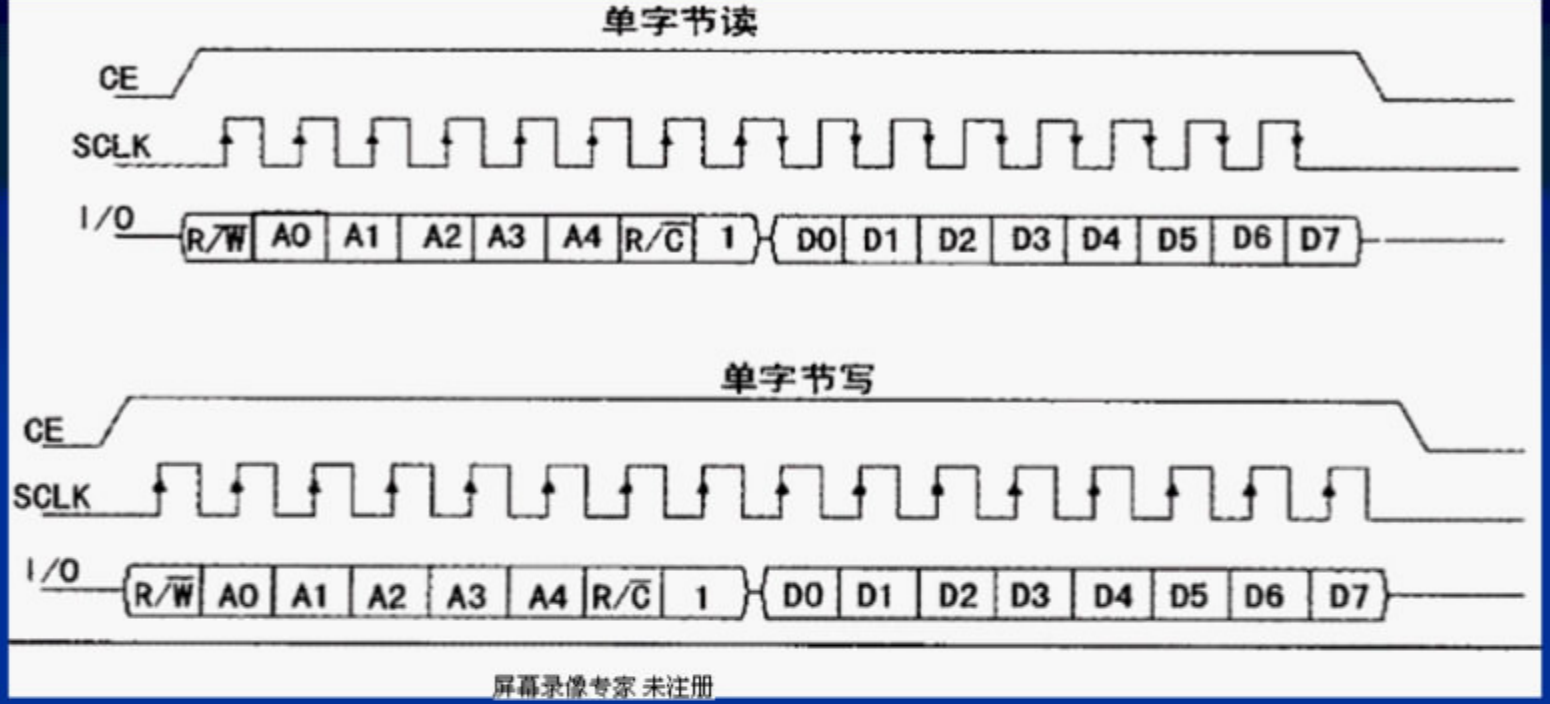
单字节读操作的一开始写控制字的过程和上面的单字节写操作是一样，但是单字节读操作在写控制字的最后一个位，SCLK还在高电平时，DS1302就将数据放到I/O上，单片机将SCLK置为低电平后数据锁存，单机机就可以读取I/O上的数据。如此反复，将一个字节的数据读入单片机。

读与写操作的不同就在于，写操作是在SCLK低电平时单片机将数据放到IO上，当SCLK上升沿时，DS1302读取。而读操作是在SCLK高电平时DS1302放数据到IO上，将SCLK置为低电平后，单片机就可从IO上读取数据。

操作DS1302

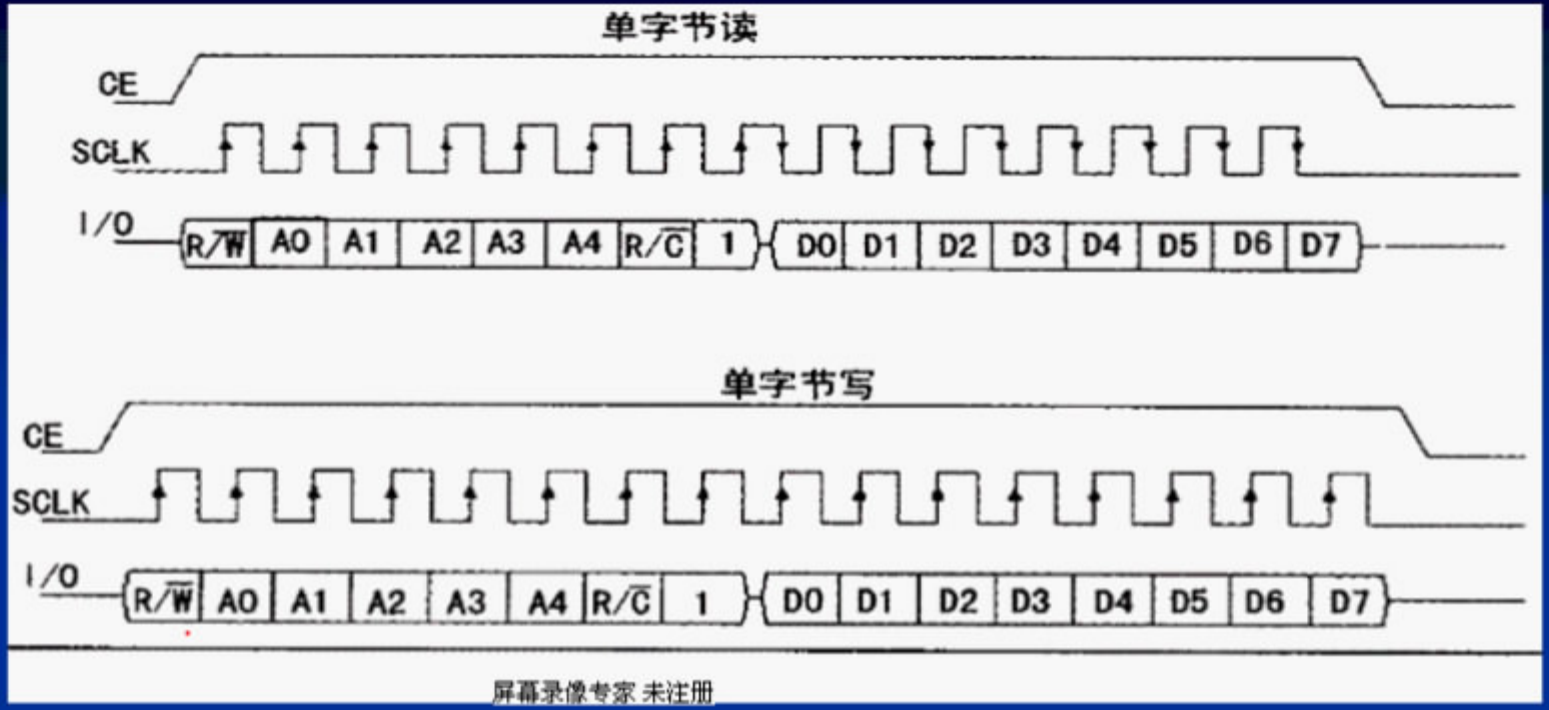
读寄存器	写寄存器	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	范围
81h	80h	CH	10 秒			秒				00–59
83h	82h		10 分			分				00–59
85h	84h	12/$\overline{24}$	0	10 AM/PM	时	时				1–12/0–23
87h	86h	0	0	10 日		日				1–31
89h	88h	0	0	0	10 月	月				1–12
8Bh	8Ah	0	0	屏幕录像专家 未注册	0	0	周日			1–7
8Dh	8Ch	10 年				年				00–99
8Fh	8Eh	WP	0	0	0	0	0	0	0	—

- 我们现在就来编程看一下，如何对DS1302进行操作把时钟信息显示在1602LCD上。
  1 首先要通过8EH将写保护去掉，将日期，时间的初值写时各个寄存器。
  2 然后就可以对80H、82H、84H、86H、88H、8AH、8CH进行初值的写入。同时也通过秒寄存器将位7的CH值改成0，这样DS1302就开始走时运了。
  3 将写保护寄存器再写为80H，防止误改写寄存器的值。
  4 不断读取80H—8CH的值，将它们格式化后显示到1602LCD液晶上



屏幕录像专家 未注册

```
void Write1302(unsigned char dat)
{
 unsigned char i;
 SCLK=0;          //拉低SCLK，为脉冲上升沿写入数据做好准备
 delaynus(2);     //稍微等待，使硬件做好准备
 for(i=0;i<8;i++)     //连续写8个二进制位数据
  {
                    DATA=dat&0x01;   //取出dat的第0位数据写入1302 低位在前，高位在后
                    delaynus(2);     //稍微等待，使硬件做好准备
                    SCLK=1;          //上升沿写入数据
                    delaynus(2);     //稍微等待，使硬件做好准备
                    SCLK=0;          //重新拉低SCLK，形成脉冲
                    dat>>=1;         //将dat的各数据位右移1位，准备写入下一个数据位
          }
```




```
unsigned char Read1302(void)
{
  unsigned char i,dat;
            delaynus(2);      //稍微等待，使硬件做好准备
            for(i=0;i<8;i++)  //连续读8个二进制位数据
             {
                        dat>>=1;      //将dat的各数据位右移1位，因为先读出的是字节的最低位
                        if(DATA==1)   //如果读出的数据是1
                        dat|=0x80;    //将1取出，写在dat的最高位
                        SCLK=1;       //将SCLK置于高电平，为下降沿读出
                        delaynus(2);  //稍微等待
                        SCLK=0;       //拉低SCLK，形成脉冲下降沿
                        delaynus(2);  //稍微等待
             }
  return dat;       //将读出的数据返回
```

读寄存器	写寄存器	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	范围
81h	80h	CH	10 秒			秒				00–59
83h	82h		10 分			分				00–59
85h	84h	12/24	0	10 AM/PM	时	时				1–12/0–23
87h	86h	0	0	10 日		日				1–31
89h	88h	0	0	0	10 月	月				1–12
8Bh	8Ah	0	0	0	0	0	周日			1–7
8Dh	8Ch	10 年				年				00–99
8Fh	8Eh	WP	0	0	0	0	0	0	0	—

屏幕录像专家 未注册

DS1302初始化操作　　设置初始时间为 09年6月18日 23点59分55秒

```
flag= ReadSet1302(0x81);//读秒寄存器最高位，读出时钟状态
    if(flag&0x80)//判断时钟是否关闭，如内部关闭就初始化，如内部还在走时 那就不初始化
        {
     WriteSet1302(0x8E,0x00);              //根据写状态寄存器命令字，写入不保护指令
 WriteSet1302(0x80,((55/10)<<4|(55%10)));  //根据写秒寄存器命令字，写入秒的初始值
    WriteSet1302(0x82,((59/10)<<4|(59%10)));  //根据写分寄存器命令字，写入分的初始值
    WriteSet1302(0x84,((23/10)<<4|(23%10)));//根据写小时寄存器命令字，写入小时的初始值
    WriteSet1302(0x86,((18/10)<<4|(18%10)));//根据写日寄存器命令字，写入日的初始值
    WriteSet1302(0x88,((6/10)<<4|(6%10)));//根据写月寄存器命令字，写入月的初始值
    WriteSet1302(0x8c,((9/10)<<4|(9%10)));  //根据写年寄存器命令字，写入年的初始值
    WriteSet1302(0x90,0xa5);              //打开充电功能 选择2K电阻充电方式
    WriteSet1302(0x8E,0x80);              //根据写状态寄存器命令字，写入保护指令
```

BCD码

- BCD码　用4位二进制数来表示1位十进制数中的0~9这10个数码，
- 简称BCD码（Binary-Coded Decimal），简称BCD
- 非压缩的BCD码，

屏幕录像专家 未注册

非压缩的BCD码用8位二进制数表示一个十进制数位，其中低4位是BCD码，高4位是0。例如，十进制数78表示成压缩的BCD码为　0000 0111 0000 1000

- 压缩的BCD码：

压缩的BCD码用4位二进制数表示一个十进制数位，整个十进制数用一串BCD码来表示。

- 例如，十进制数59表示成压缩的BCD码为0101 1001

读寄存器	写寄存器	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	范围
81h	80h	CH	10秒			秒				00–59
83h	82h		10分			分				00–59
85h	84h	12/$\overline{24}$	0	10 AM/PM	时	时				1–12/0–23
87h	86h	0	0	10日		日				1–31
89h	88h	0	0	0	10月	月				1–12
8Bh	8Ah	0	0	0	0	0	周日			1–7
8Dh	8Ch	10年				年				00–99
8Fh	8Eh	WP	0	0	0	0	0	0	0	—

```
ReadValue = ReadSet1302(0x81);  //从秒寄存器读数据
  second=((ReadValue&0x70)>>4)*10 + (ReadValue&0x0F);//将读出数据转化
          DisplaySecond(second);        //显示秒
          ReadValue = ReadSet1302(0x83); //从分寄存器读
  minute=((ReadValue&0x70)>>4)*10 + (ReadValue&0x0F); //将读出数据转化
          DisplayMinute(minute);                //显示分
  ReadValue = ReadSet1302(0x85); //从分寄存器读
  hour=((ReadValue&0x70)>>4)*10 + (ReadValue&0x0F); //将读出数据转化
          DisplayHour(hour);              //显示小时
  ReadValue = ReadSet1302(0x87); //从分寄存器读
  day=((ReadValue&0x70)>>4)*10 + (ReadValue&0x0F); //将读出数据转化
          DisplayDay(day);              //显示日
          ReadValue = ReadSet1302(0x89); //从分寄存器读
  month=((ReadValue&0x70)>>4)*10 + (ReadValue&0x0F); //将读出数据转化
          DisplayMonth(month);        //显示月
          ReadValue = ReadSet1302(0x8d); //从分寄存器读
  year=((ReadValue&0xF0)>>4)*10 + (ReadValue&0x0F); //将读出数据转化
```