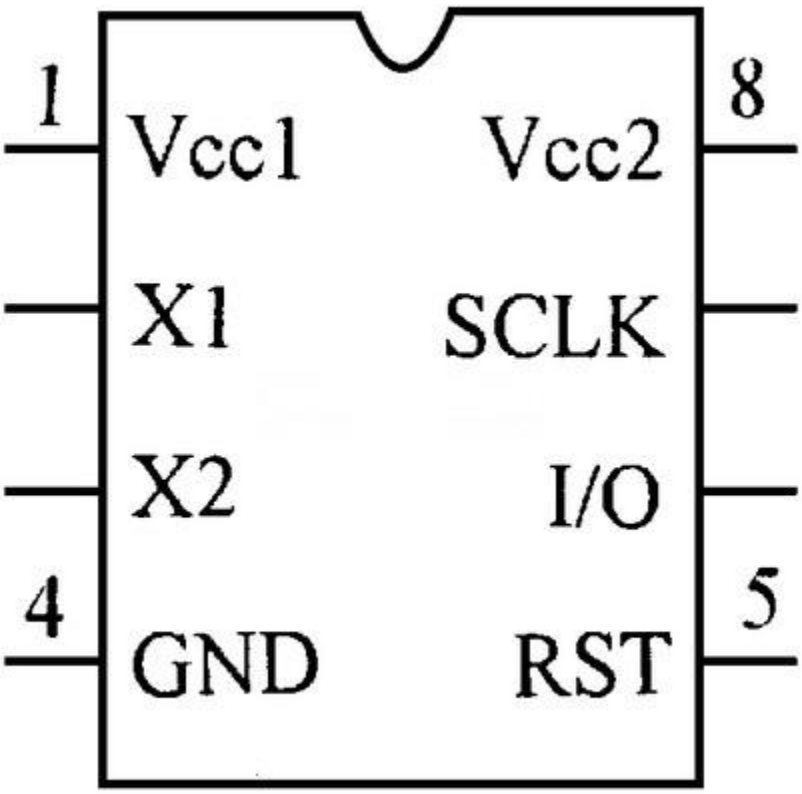
DS1302 是美国 DALLAS 公司推出的一种高性能、低功耗的实时时钟芯片，附加 31 字节静态 RAM，采用 SPI 三线接口与 CPU 进行同步通信，并可采用突发方式一次传送多个字节的时钟信号和 RAM 数据。实时时钟可提供秒、分、时日、星期、月和年，一个月小与 31 天时可以自动调整，且具有闰年补偿功能。

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，提供了对后背电源进行涓细电流充电的能力。

1.管脚图：



Vcc1：主电源；Vcc2：备份电源。当 Vcc2>Vcc1+0.3V 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、开始控制字访问移位寄存器的控制逻辑；第二、CE 提供结束单字节或多字节数据传输的方法。

2.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	—

DS1302 有关日历、时间的寄存器共有 12 个，其中有 7 个寄存器（读时 81h～8Dh，写时 80h～8Ch）存放的数据格式为 BCD 码形式。

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，且位 5 是为 1 时，表示 PM；在 24 小时模式时，位 55 是第二个 10 小时位。

秒寄存器（81h、80h）的位 77 定义为时钟暂停标志（CH）。当该位置为 1 时，时钟振荡器停止，DS1302 处于低功耗状态；当该位置为 0 时，时钟开始运行。

控制寄存器（8Fh、8Eh）的位 77 是写保护位（WP），其它 7 位均置为 0。在任何的对时钟和 RAM 的写操作之前，WP 位必须为 0。

当 WP 位为 11 时，写保护位防止对任一寄存器的写操作。

3.时序图：

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。

控制字：

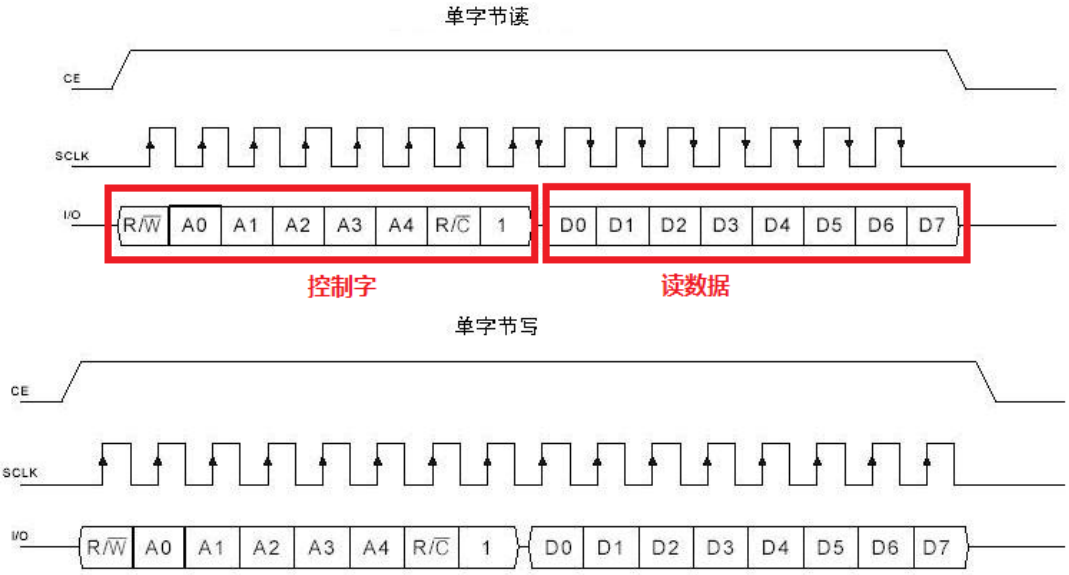
7	6	5	4	3	2	1	0
1	RAM $\overline{CK}$	A4	A3	A2	A1	A0	RD $\overline{WR}$

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；

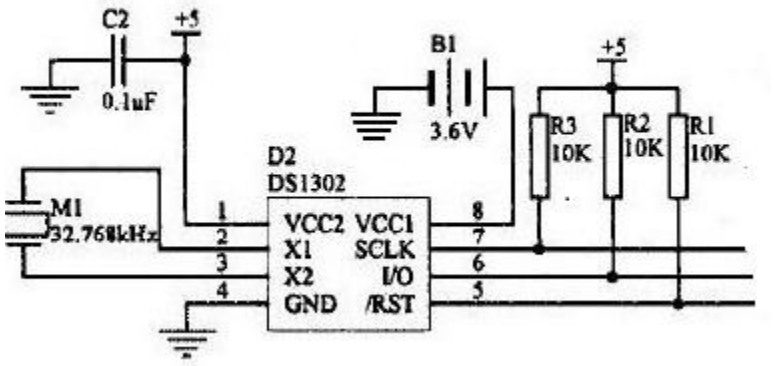
位 5 至位 1（A4～A0）：指示操作单元的地址；

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。

3. 原理图：



4. 程序控制：

```
/*  (Ds1302+LCD1602) */
#include<reg52.h>
#include<intrins.h>
sbit DS1302_CLK=P1^1;
sbit DS1302_IO=P1^0;
sbit DS1302_RST=P1^2;
```

```
sbit lcden=P2^2;
sbit lcdrw=P2^1;
sbit lcdrs=P2^0;


#define uchar unsigned char
#define uint unsigned int
#define RS_CLR lcdrs=0
#define RS_SET lcdrs=1

#define RW_CLR lcdrw=0
#define RW_SET lcdrw=1

#define EN_CLR lcden=0
#define EN_SET lcden=1

#define DataPort P2


unsigned char second,minute,hour,week,day,month,year;
unsigned char table[]="0123456789";
//unsigned char table1[]="          ";
unsigned char table2[]="                  ";
unsigned char table3[]="Time: ";
unsigned char table4[]="Date: ";
unsigned int time[]={
0x13,0x05,0x18,0x03,0x00,0x00,0x00
};
uchar num,count,date,shi;
```

```
unsigned char receive[3]={0xff,0xff,0xff};
unsigned char ComBuf[3]={0x02,0x90,0x08};



void delay(uint z)
{
  uint x,y;
  for(x=z;x>0;x--)
    for(y=110;y>0;y--);
}
void DelayUs2x(unsigned char t)
{
 while(--t);
}
/*------------------------------------------------
 mS 延时函数，含有输入参数 unsigned char t，无返回值
 unsigned char 是定义无符号字符变量，其值的范围是
 0~255 这里使用晶振 12M，精确延时请使用汇编
------------------------------------------------*/
void DelayMs(unsigned char t)
{

 while(t--)
 {
     //大致延时 1mS
     DelayUs2x(245);
    DelayUs2x(245);
 }
```

```
}
//向 DS1302 送一个字节//
void InputByte(unsigned char byte1)
{
 char i;
 for(i=8;i>0;i--)
   {
     DS1302_IO=(bit)(byte1&0x01);
     DS1302_CLK=1;
     _nop_();
     DS1302_CLK=0;
     byte1>>=1;
   }
    return;
}
//读 DS1302 一个字节//
unsigned char outputbyte(void)
{
   unsigned char i;
   unsigned char ucdat=0;
   for(i=8;i>0;i--)
    {
     DS1302_IO=1;
     ucdat>>=1;
     if(DS1302_IO)ucdat|=0x80;
     DS1302_CLK=1;
     _nop_();
     DS1302_CLK=0;
    }
```

```
    return(ucdat);

}
//向 DS1302 某地址写一个字节数据//
void write_ds1302(unsigned char addr,unsigned char TDat)
{
 DS1302_RST=0;
 _nop_();
 DS1302_CLK=0;
 _nop_();
 DS1302_RST=1;
 InputByte(addr);
 _nop_();
 InputByte(TDat);
 DS1302_CLK=1;
 _nop_();
 DS1302_RST=0;
}
//读 DS1302 地址子程序//
unsigned char read_ds1302(unsigned char addr)
{
   unsigned char timedata;
   DS1302_RST=0;
   _nop_();
   DS1302_CLK=0;
   _nop_();
   DS1302_RST=1;
   InputByte(addr);
   timedata=outputbyte();
```

```
    DS1302_CLK=1;
    _nop_();
    DS1302_RST=0;
    return(timedata);
}
//DS1302 的初始化//
void initial_ds1302()
{
  write_ds1302(0x8e,0x00);//写保护寄存器
  write_ds1302(0x8c,time[0]);//年
  write_ds1302(0x88,time[1]);//月
  write_ds1302(0x86,time[2]);//日
  write_ds1302(0x8A,time[3]);//星期
  write_ds1302(0x84,time[4]);//时
  write_ds1302(0x82,time[5]);//分
  write_ds1302(0x80,time[6]);//秒
  write_ds1302(0x8e,0x80);//写保护寄存器
}
//读 DS1302 时间//
void read_time()
{
 second=read_ds1302(0x81);
 minute=read_ds1302(0x83);
 hour=read_ds1302(0x85);
 week=read_ds1302(0x8B);
 day=read_ds1302(0x87);
 month=read_ds1302(0x89);
 year=read_ds1302(0x8d);
```

```
}

//液晶显示程序//
void write_com(uchar com)//写指令//
{
 lcdrs=0;
 lcdrw=0;
 P0=com;
 delay(5);
 lcden=1;
 delay(5);
 lcden=0;
}
void write_data(uchar date)   //写数据//
{
 lcdrs=1;
 lcdrw=0;
 P0=date;
 delay(5);
 lcden=1;
 delay(5);
 lcden=0;
}
void init()
{

  lcden=0;
  write_com(0x38);    //置初值//
  write_com(0x0c);
```

```
  write_com(0x06);
  write_com(0x81);
}
void Display(void)
{
    num=0;
    count=0;
    for(date=0;date<6;date++)
      {
         write_data(table3[date]);
      }
    //write_com(0x81);
    write_data(table[hour/16]);   //写第 1 行数据//
    write_data(table[hour%16]);
    write_data(':');

    write_data(table[minute/16]);
    write_data(table[minute%16]);
    write_data(':');

    write_data(table[second/16]);
    write_data(table[second%16]);
    //write_com(0x01);
    write_com(0x80+0x41);
    for(shi=0;shi<6;shi++)
      {
         write_data(table4[shi]);
      }
    write_data(table[year/16]);
```

```
    write_data(table[year%16]);
    write_data('-');
    write_data(table[month/16]);
    write_data(table[month%16]);
    write_data('-');
    write_data(table[day/16]);
    write_data(table[day%16]);

    write_com(0x80);
    for(count=0;count<16;count++);
      {
        write_data(table2[count]);
      }

}
/*-----------------------------------------------
              判忙函数
-----------------------------------------------*/
 bit LCD_Check_Busy(void)
 {
 DataPort= 0xFF;
 RS_CLR;
 RW_SET;
 EN_CLR;
 _nop_();
 EN_SET;
 return (bit)(DataPort & 0x80);
 }
/*-----------------------------------------------
```

写入命令函数

----------*/

```
 void LCD_Write_Com(unsigned char com)
 {
 while(LCD_Check_Busy()); //忙则等待
 RS_CLR;
 RW_CLR;
 EN_SET;
 DataPort= com;
 _nop_();
 EN_CLR;
 }
```

/*----------

写入数据函数

----------*/

```
 void LCD_Write_Data(unsigned char Data)
 {
 while(LCD_Check_Busy()); //忙则等待
  RS_SET;
  RW_CLR;
  EN_SET;
  DataPort= Data;
  _nop_();
  EN_CLR;
 }
```

/*----------

清屏函数

----------*/

```
void LCD_Clear(void)
{
LCD_Write_Com(0x01);
DelayMs(5);
}
/*------------------------------------------
                写入字符串函数
------------------------------------------*/
void LCD_Write_String(unsigned char x,unsigned char y,unsigned char
*s)
{
if (y == 0)
   {
    LCD_Write_Com(0x80 + x);     //表示第一行
   }
else
   {
   LCD_Write_Com(0xC0 + x);      //表示第二行
   }
while (*s)
   {
LCD_Write_Data( *s);
s ++;
   }
}
/*------------------------------------------
                写入字符函数
------------------------------------------*/
```

```
void LCD_Write_Char(unsigned char x,unsigned char y,unsigned char Data)
{
if (y == 0)
   {
   LCD_Write_Com(0x80 + x);
   }
else
   {
   LCD_Write_Com(0xC0 + x);
   }
LCD_Write_Data( Data);
}
/*------------------------------------------------
              初始化函数
------------------------------------------------*/
void LCD_Init(void)
{
  LCD_Write_Com(0x38);    /*显示模式设置*/
  DelayMs(5);
  LCD_Write_Com(0x38);
  DelayMs(5);
  LCD_Write_Com(0x38);
  DelayMs(5);
  LCD_Write_Com(0x38);
  LCD_Write_Com(0x08);    /*显示关闭*/
  LCD_Write_Com(0x01);    /*显示清屏*/
  LCD_Write_Com(0x06);    /*显示光标移动设置*/
  DelayMs(5);
```

```
   LCD_Write_Com(0x0C);     /*显示开及光标设置*/
   }
void lcdxxmain(void)
{




  LCD_Write_Char(7,0,'0');
  LCD_Write_Char(8,0,'0');
  LCD_Write_String(1,1,"cumt j1b201");

 }
 void lcdshizhongmain(void)
 {


    read_time();
    Display();

 }
void main(void)
{
  int i,j;
  initial_ds1302();
   init();
  while(1)
  {
  for(i=0;i<100;i++)
  {
```

```
        lcdshizhongmain();
    }
        LCD_Init();
       LCD_Clear();//清屏
   for(j=0;j<10000;j++)
   {
     lcdxxmain();
   }
    }


}
```