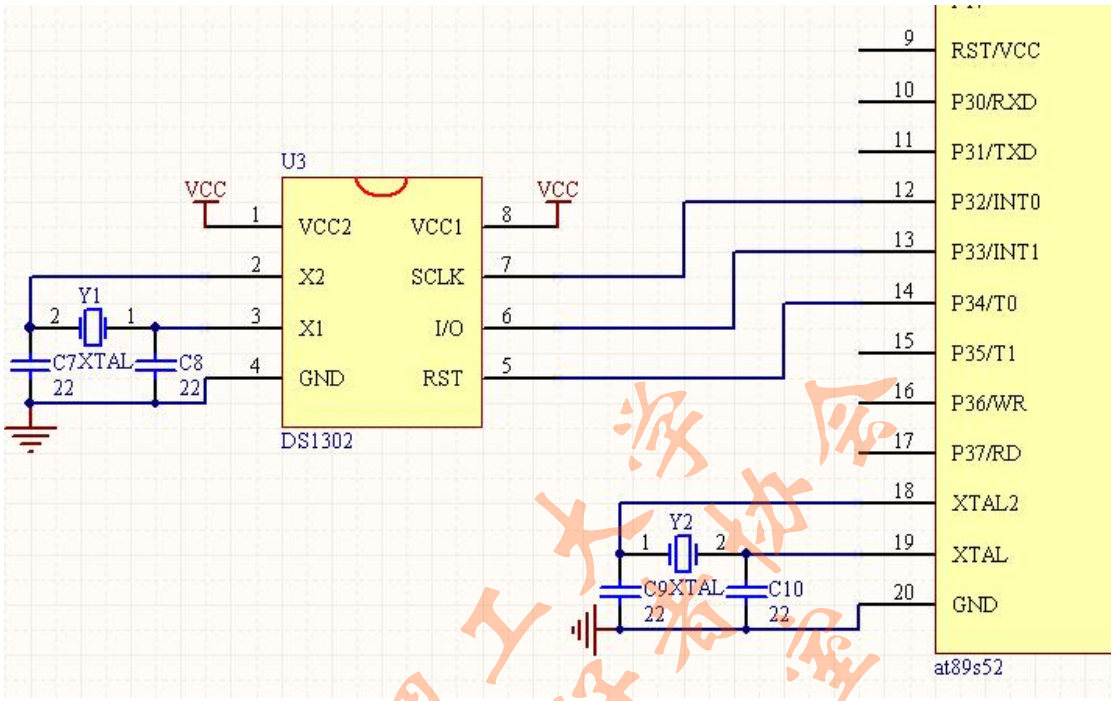
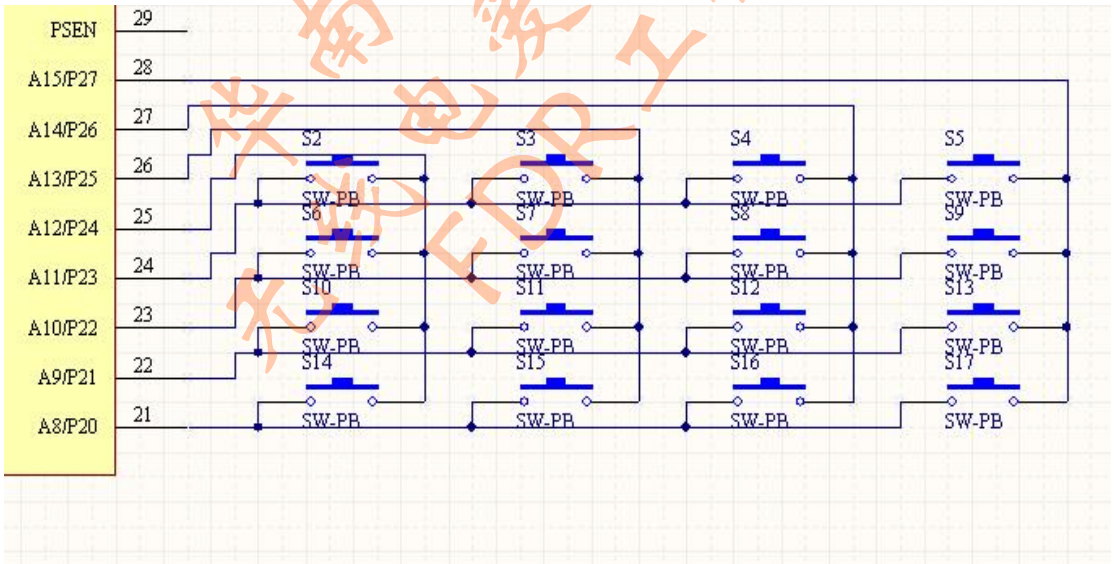
实验十四: DS1302 实时时钟实验

一、 硬件原理图：

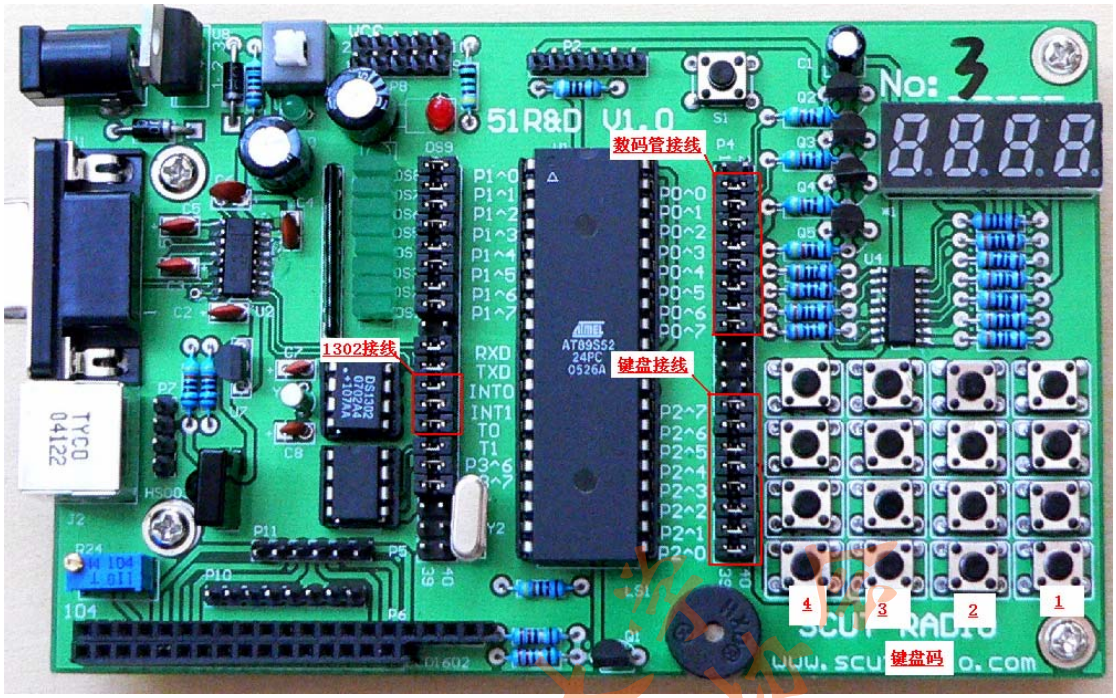


Ds1302 接线图



键盘接线

二、硬件接线图：



三、实验原理：

DS1302 是美国DALLAS公司推出的一种高性能、低功耗、带RAM的实时时钟电路，它可以对年、月、日、周日、时、分、秒进行计时，具有闰年补偿功能，工作电压为 2.5V～5.5V。采用三线接口与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或RAM数据。DS1302 内部有一个 31×8 的用于临时性存放数据的RAM寄存器。DS1302 是DS1202 的升级产品，与DS1202 兼容，但增加了主电源/后背电源双电源引脚，同时提供了对后背电源进行涓细电流充电的能力。

2.1 引脚功能及结构 图 1 示出DS1302 的引脚排列,其中V_{cc1}为后备电源，V_{CC2}为主电源。在主电源关闭的情况下，也能保持时钟的连续运行。DS1302 由V_{cc1}或V_{cc2}两者中的较大者供电。当V_{cc2}大于V_{cc1}＋0.2V时，V_{cc2}给DS1302 供电。当V_{cc2}小于V_{cc1}时，DS1302 由V_{cc1}供电。X1 和X2 是振荡源，外接 32.768kHz晶振。RST是复位/片选线，通过把RST输入驱动置高电平来启动所有的数据传送。RST输入有两种功能：首先，RST接通控制逻辑，允许地址/命令序列送入移位寄存器；其次，RST提供终止单字节或多字节数据的传送手段。当RST为高电平时，所有的数据传送被初始化，允许对DS1302 进行操作。如果在传送过程中RST置为低电平，则会终止此次数据传送，I/O引脚变为高阻态。上电运行时，在V_{cc}≥2.5V之前，RST必须保持低电平。只有在SCLK为低电平时，才能将RST置为高电平。I/O为串行数据输入输出端(双向)，后面有详细说明。SCLK始终是输入端。

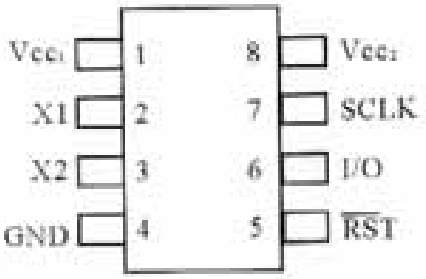


图 1　所示为 DS1302 的引脚图

2.2 DS1302 的控制字节　DS1302 的控制字如图 2 所示。控制字节的最高有效位(位 7)必须是逻辑 1，如果它为 0，则不能把数据写入 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 指示操作单元的地址;最低有效位(位 0)如为 0 表示要进行写操作，为 1 表示进行读操作，控制字节总是从最低位开始输出。

2.3 数据输入输出(I/O)

在控制指令字输入后的下一个 SCLK 时钟的上升沿时，数据被写入 DS1302，数据输入从低位即位 0 开始。同样，在紧跟 8 位的控制指令字后的下一个 SCLK 脉冲的下降沿读出 DS1302 的数据，读出数据时从低位 0 位到高位 7。 DS1302 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,其日历、时间寄存器及其控制字见表 1。

表 1　日历、时间寄存器及其控制字

寄存器名称	命令字		取值范围	各位内容							
	写操作	读操作		7	6	5	4	3	2	1	0
秒寄存器	80H	81H	00～59	CH	10SEC				SEC		
分寄存器	82H	83H	00～59	0	10MIN				MIN		
时寄存器	84H	85H	01～12 或 00～23	12/24	0	10 HR			HR		
日寄存器	86H	87H	01～28,29,30,31	0	0	10DATE			DATE		
月寄存器	88H	89H	01～12	0	0	0	10M		MONTH		
周寄存器	8AH	8BH	01～07	0	0	0	0	0	DAY		
年寄存器	8CH	8DH	00～99	10YEAR					YEAR		

2.4 程序功能：对 ds1302 进行写时间，并读出来放在数码管上显示，键盘右

下角为第一个键，底排键从右到左键码为 1~4。按下 1，显示分秒；按下 2，显示星期和小时；按下 3，显示月日；按下 4，显示年份。默认显示分秒。

四、程序代码：

```
#include<reg52.h>

#define LEDPORT P0          //数码管接线
sbit SCLK=P3^2;             //ds1302 接线
sbit IO=P3^3;
sbit RST=P3^4;

sbit ACC7=ACC^7;
sbit ACC0=ACC^0;

unsigned char times[7]={25,33,23,6,11,2,0x07};//用于存放时间
unsigned char KEY=0;       //用于读取键盘值

void write_1302_byte(unsigned char d);              //向 1302 写入一字节数据 d
unsigned char read_1302_byte(void);                 //从 1302 读出一字节数据并返回
void write_1302_reg(unsigned char add,unsigned char d);     //向 1302 地址 add 处写入数据 d
unsigned char read_1302_reg(unsigned char add);             //从 1302 地址 add 处读出数据并返回
void write_1302_time(unsigned char *time);          //设置时间
void read_1302_time(unsigned char time[]);          //读出时间

/*=========================================================*/
/*                  向 1302 当前地址处写入 1 字节数据                  */
/*=========================================================*/
void write_1302_byte(unsigned char d)
{
    unsigned char i;
    ACC=d;                  //将 8 位数据写入 ACC
    for(i=8;i>0;i--)      //循环 8 次将 ACC8 位数据从低位到高位写入 1302
    {
        IO=ACC0;           //将 ACC 的最低位转送到 IO 口
        SCLK=1;            //上升沿写入 1 位数据
        SCLK=0;
        ACC=ACC>>1;          //将 ACC 次低位变为最低位
    }
}
/*=========================================================*/
```

```
/*=========================================================*/
/*                从 1302 的当前地址处读出 1 字节数据
*/
/*=========================================================*/
unsigned char read_1302_byte(void)
{
    unsigned char i;
    for(i=8;i>0;i--)
    {
        ACC=ACC>>1;
        ACC7=IO;
        SCLK=1; //时钟下降沿读 1 位数据
        SCLK=0;

    }
    return (ACC);
}
/*=========================================================*/

/*=========================================================*/
/*                    将数据 Data 写入 1302 寄存器 add 处                    */
/*=========================================================*/
void write_1302_reg(unsigned char add,unsigned char d)
{
    RST= 0;
    SCLK=0;
    RST= 1;
    write_1302_byte(add);  //将 1302 的指针移到 add 处
    write_1302_byte(d);          //往 add 处写入数据
    SCLK=1;
    RST=0;                  //停止写数据
}
/*=========================================================*/

/*=========================================================*/
/*                        读寄存器 add 处的值                        */
/*=========================================================*/
unsigned char read_1302_reg(unsigned char add)
{
    unsigned char d;
    RST=0;
    SCLK=0;
    RST=1;
```

```
    write_1302_byte(add);
    d=read_1302_byte();
    SCLK=1;
    RST=0;
    return (d);
}
/*=====================================================*/

/*=====================================================*/
/*                设置时间,time[0]-time[6]数据存放顺序为:秒,分,时,日,月,星期,
年                 */
/*=====================================================*/
void write_1302_time(unsigned char *time)
{
    unsigned char add=0x80;          //地址 add 初值(秒),偶数地址为只写
    unsigned char i;
    unsigned char timeBCD[7];     //用以存放 time 数据的 BCD 码值
    bdata unsigned char l,h;     //单位变量,分别存放时间数据(8421BCD  码)的
低 4 位和高 4 位
    for(i=0;i<7;i++)                 //将时间数据转为 BCD 码放入 timeBCD 中
    {
        l=time[i]%10;
        h=time[i]/10;
        timeBCD[i]=h*16+l;
    }
    write_1302_reg(0x8e,0x00);       //禁止写
    for(i=0;i<7;i++)
    {
        write_1302_reg(add,timeBCD[i]); //将时间数据写入 1302 对应的寄存器
        add+=2;                          //地址移动,确保地址为偶数
    }
    write_1302_reg(0x8e,0x80);           //允许写
}
/*=============================================*/

/*=====================================================*/
/*                                                读取当前时间
*/
/*=====================================================*/
void read_1302_time(unsigned char time[])
{
    unsigned char i;
    unsigned char add=0x81;          //设置地址 add 初值(秒),奇数为只读
    bdata unsigned char l,h;         //单位变量,分别存放时间数据(8421BCD 码)
```

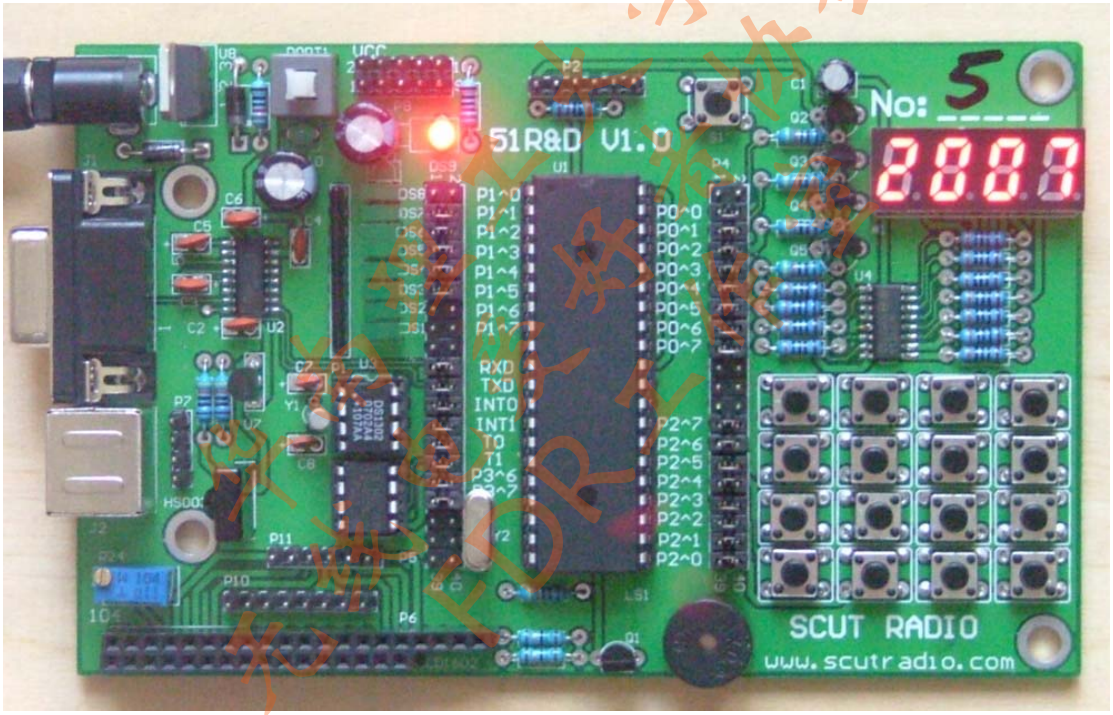
```
的低 4 位和高 4 位,用以进行十进制转换
    write_1302_reg(0x8e,0x00);   //禁止写
     for(i=0;i<7;i++)
     {
          time[i]=read_1302_reg(add);   //将时间数据(8421BCD 码)从 1302 对应
的寄存器中读出
          l=time[i]&0x0f;              //l 存放 time 的低 4 位,即个位
          h=(time[i]>>4)&0x0f;       //h 存放 time 的高 4 位,即十位
          time[i]=h*10+l;             //时间数据以十进制形式放进 time[i]中
          add+=2;                      //地址移动,确保地址为奇数
     }
}
/*=============================================*/
void delayms(unsigned char i)//延时函数
{
    unsigned char j;
    while(i--)
    for(j=101;j>1;j--);
}
/*=============================================*/
/*                  读键盘值                   */
/*=============================================*/
//将读到到的键盘值放入全局变量 KEY 中
void keyread()
{
    unsigned char x,y,z;
    P2=0xf0;              //列扫描
    P2=0xf0;              //列扫描
    if(P2!=0xf0)          //若有按下信号
    {
          delayms(5);
          y=P2;             //保存行扫描时有键按下时状态
          P2=0x0f;          //行扫描
           delayms(5);
          x=P2;             //保存列扫描时有键按下时状态
          z=x|y;            //取出键值
          switch(z)         //判断键值
          {
              case 0xe7: KEY=4;break;
              case 0xd7: KEY=8;break;
              case 0xb7: KEY=12;break;
              case 0x77: KEY=16;break;
              case 0xeb: KEY=3;break;
              case 0xdb: KEY=7;break;
```

```
                case 0xbb: KEY=11;break;
                case 0x7b: KEY=15;break;
                case 0xed: KEY=2;break;
                case 0xdd: KEY=6;break;
                case 0xbd: KEY=10;break;
                case 0x7d: KEY=14;break;
                case 0xee: KEY=1;break;
                case 0xde: KEY=5;break;
                case 0xbe: KEY=9;break;
                case 0x7e: KEY=13;break;
            }
            delayms(5);
        }
        P2=0xff;
}
/*-------------数码管显示函数--------------------
参数:number  要显示的数字
参数:pos       要显示的位置(从左到右为 1--4)
返回值:无
-----------------------------------------------*/
void display(unsigned char number,unsigned char pos)//数码管显示函数
{
    unsigned char temp;                    //临时变量
    if(number<10 && pos<5 && pos>0)//确定数据合法
    {
        temp=(number<<4)&0xf0;        //获得要显示的数据
        temp+=1<<(pos-1);             //送位置位
        LEDPORT=temp;                 ////送显示数据,开始显示
    }
}
/***********显示月日和时分秒的函数**********************/
void display_hour_and_minute(unsigned char count,unsigned char i)
{
    if(i==0)
    {
        if(count>=10)
        {
            display(count/10,1);
            delayms(1);
            display(count%10,2);
        }
        else    display(count%10,1);
    }
    else
```

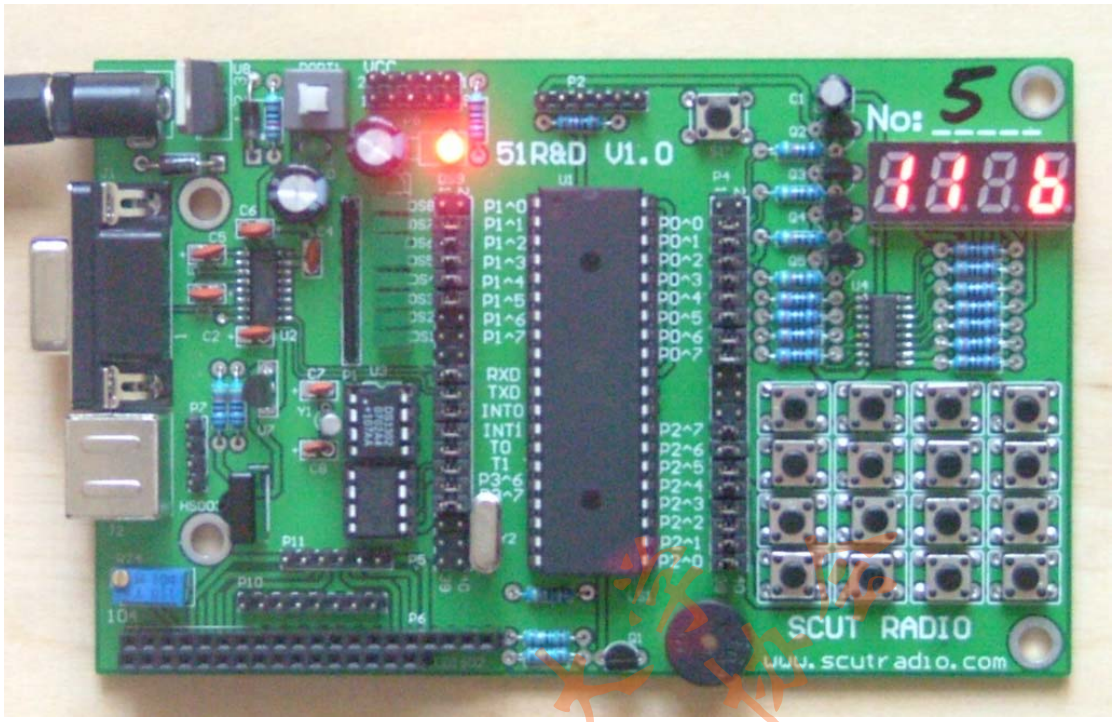
```
    {
        if(count>=10)
        {
            display(count/10,3);
            delayms(1);
            display(count%10,4);
        }
        else    display(count%10,4);
    }
    delayms(1);
}
/************显示年和星期的函数****************/
void display_year_and_weak(unsigned char count,unsigned char i)
{
    if(i==0)
    {
        display(2,1);
        delayms(1);
        display(0,2);
        delayms(1);
        display(count/10,3);
        delayms(1);
    }
    display(count%10,4);
    delayms(1);
}
void main()         //主函数
{
    unsigned int j=2000;
    write_1302_time(times);          //写初始时间
    KEY=1;
    while(1)
    {
        read_1302_time(times);
        keyread();
        while(KEY==1 && j--)                    //显示分秒
        {
            display_hour_and_minute(times[0],1);
            display_hour_and_minute(times[1],0);
        }
        while(KEY==2 && j--)                    //显示星期和小时
        {
            display_hour_and_minute(times[2],1);
            display_hour_and_minute(times[5],0);
```

```
            }
            while(KEY==3 && j--)                    //显示月和日
            {
                display_hour_and_minute(times[3],1);
                display_hour_and_minute(times[4],0);
            }
            while(KEY==4 && j--)                    //显示年号
            {
            display_year_and_weak(times[6],0);
            delayms(1);
            }
            j=50;
        }
}
```

五、效果图：



图一



图二