

GYJ-0118_STC12C5608AD 串口六位数码管 DS1302 产品

使用说明书



【简要说明】

- 一、 尺寸：长81mmX 宽 72mmX 高 20mm
- 二、 主要芯片：单片机 STC12C5608AD 继电器 光耦
- 三、 工作电压：有直流 12V 及 24V 可选。
- 四、 特点：

电路结构简单，稳定可靠，采用最新款 STC 单片机，运行速度快，单片机预留扩展接口。

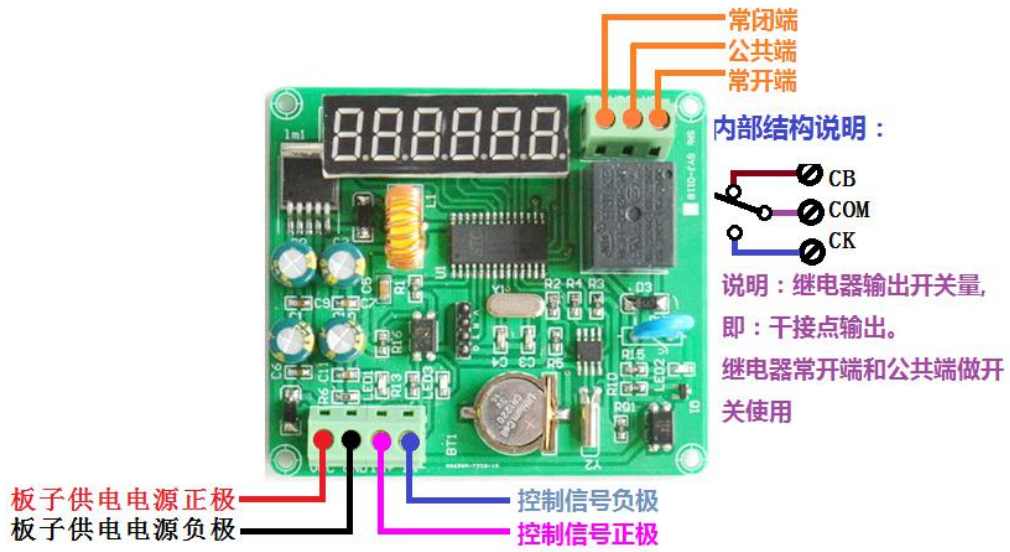
- 1、具有输入信号指示灯，继电器吸合指示灯，电源指示灯。
- 2、板子功耗小于 3W
- 3、额定切换电流 10A 以内，切换电压 250V 以内
- 4、单路最大切换功率 500W 额定功率 300W
- 5、继电器寿命 1000000 次以上。
- 6、电器绝缘电阻 100M

- 7、触电耐压 1000V
 - 8、继电器最大吸合时间 15mS 毫秒
 - 9、继电器最大释放时间 5mS 毫秒
 - 10、工作温度-40 度至 +70 度
 - 11、工作湿度 40% ~ 80%RH
 - 12、光电隔离输入，光电隔离输出
 - 13、输入支持 NPN 和 PNP 接法（即：高电平控制或者低电平控制）
 - 14、输出开关量输出（即：干接点输出）
 - 15、具有 DS1302 时钟芯片，可以编程时钟控制。
 - 16、客户可以自己编程提供参考例程及单片机资料。
 - 17、可以选择使用内部 EEPROM 作为存储单元
 - 18、电路具有，防反接保护、过流保护、续流保护等
 - 19、单片机可以自行更换，可以选择替换型的 STC 系列单片机
 - 20、我们提供电路相关的，原理图、例程、开发环境、下载软件等相关资料
- 适用场合：工业控制、产品开发、项目设计，自动化改造等

【标注说明】



【接线说明】



【输入控制设备】



输入控制也可用以下设备

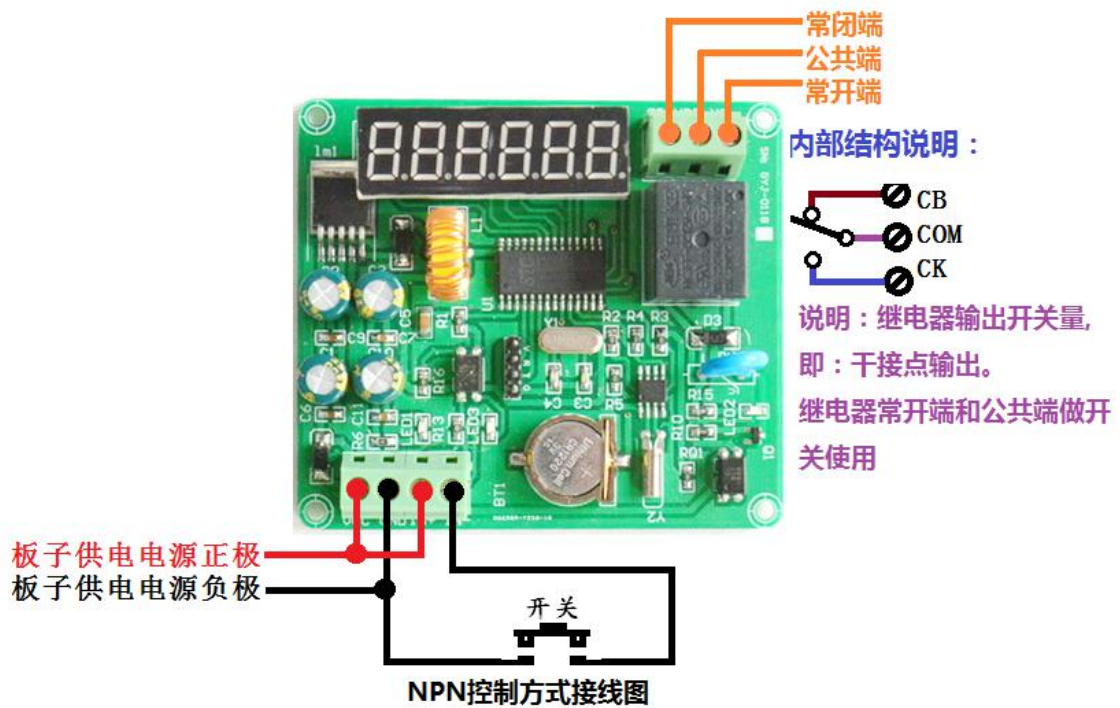


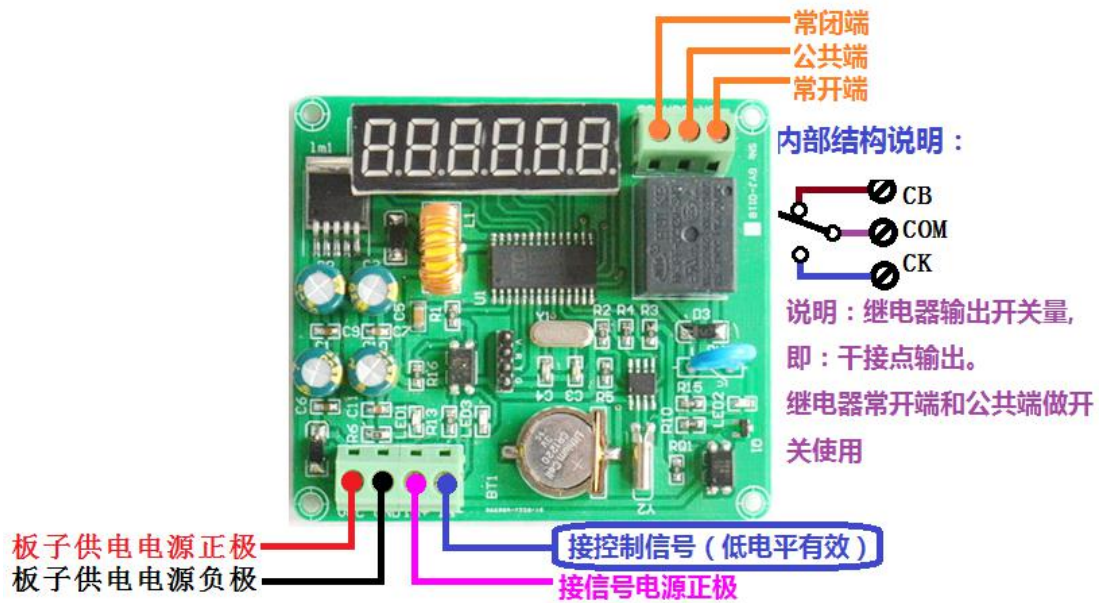
【输出控制设备】

可以控制以下设备：



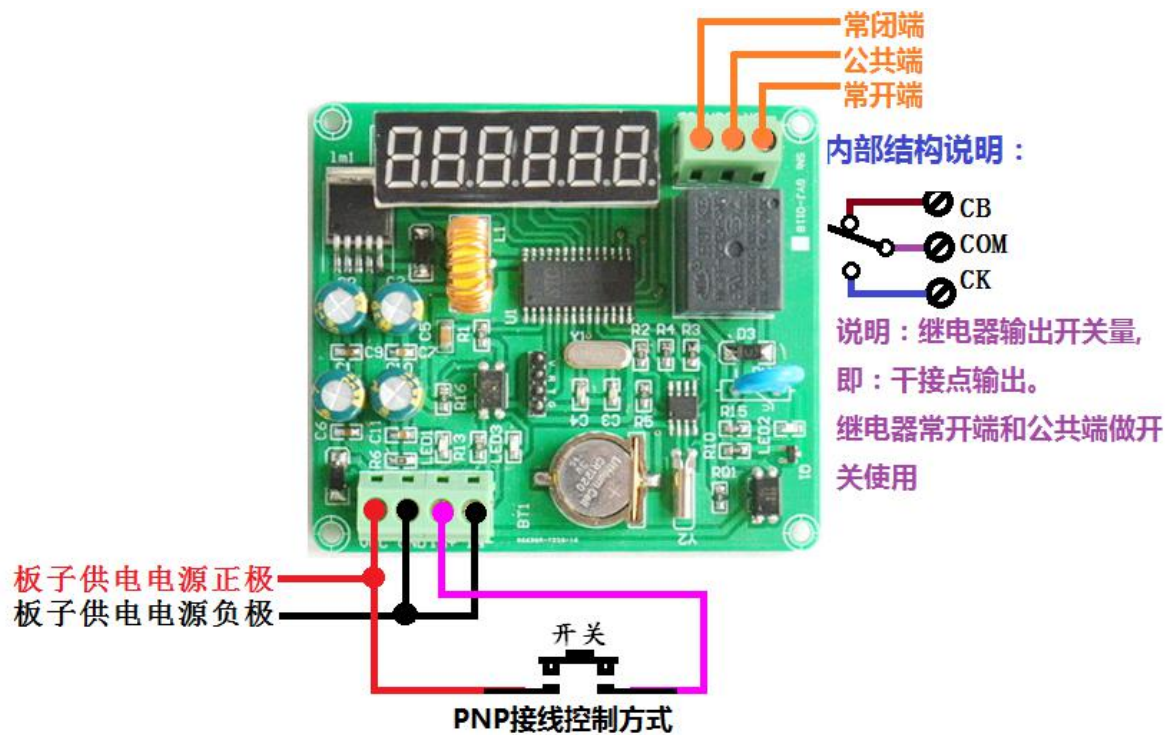
【输入接线方式举例说明】NPN 方式

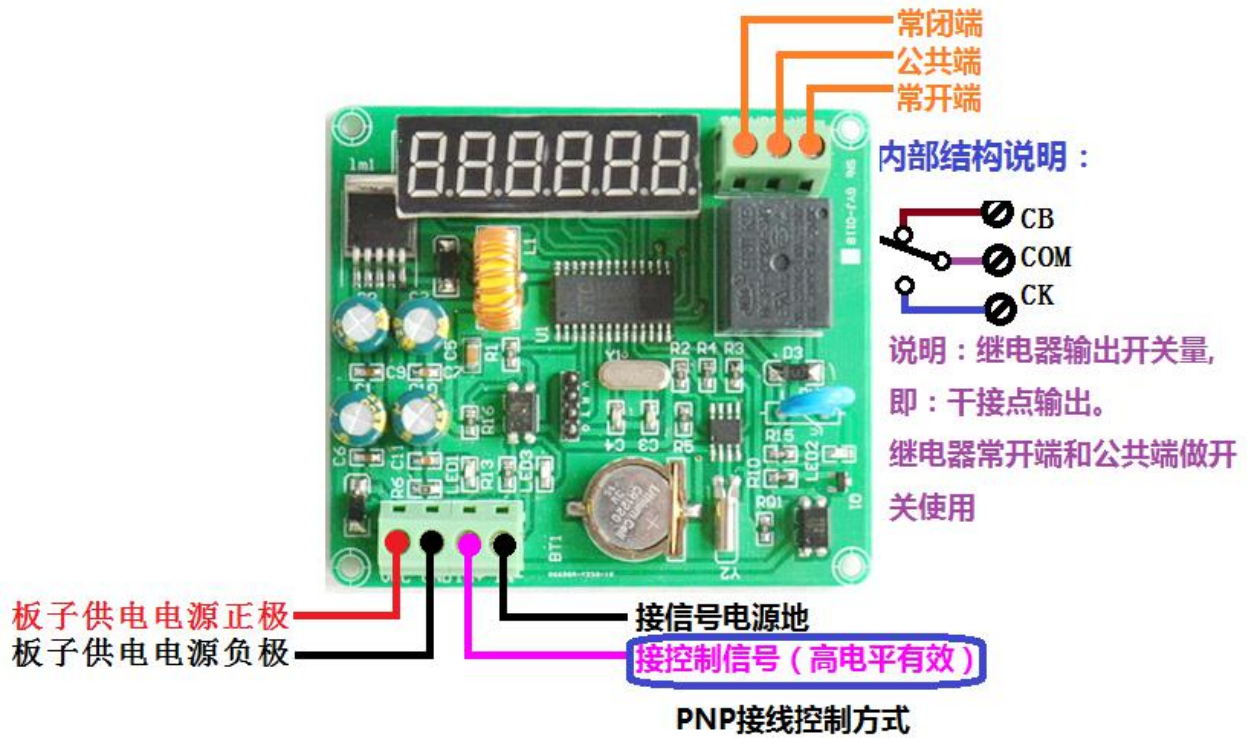




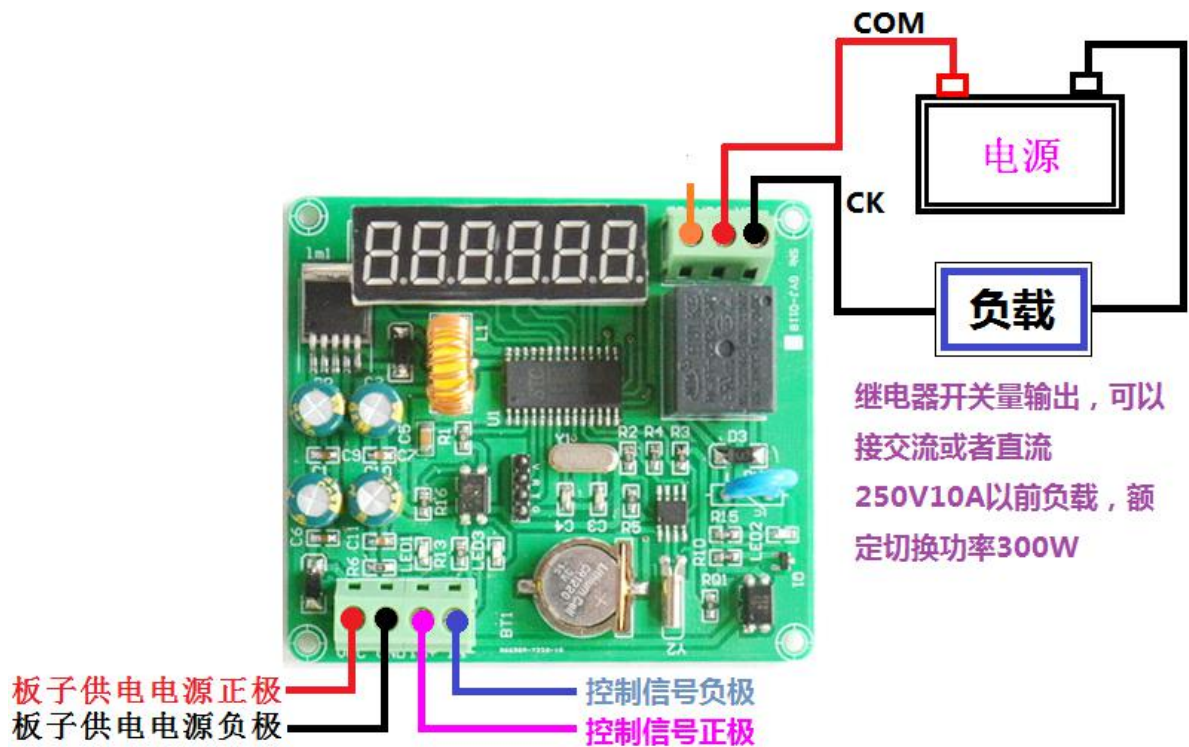
NPN控制方式接线图

【输入接线方式举例说明】PNP方式





【输出举例说明】（开关量输出、干接点输出）

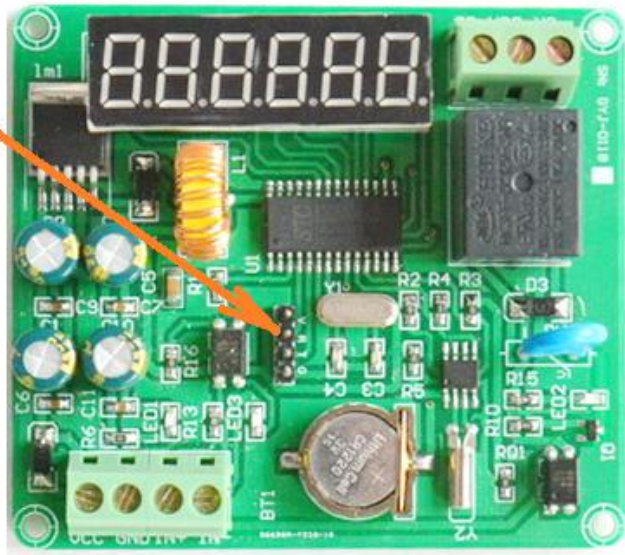


【下载口说明】也可以 UART 通信



连接图

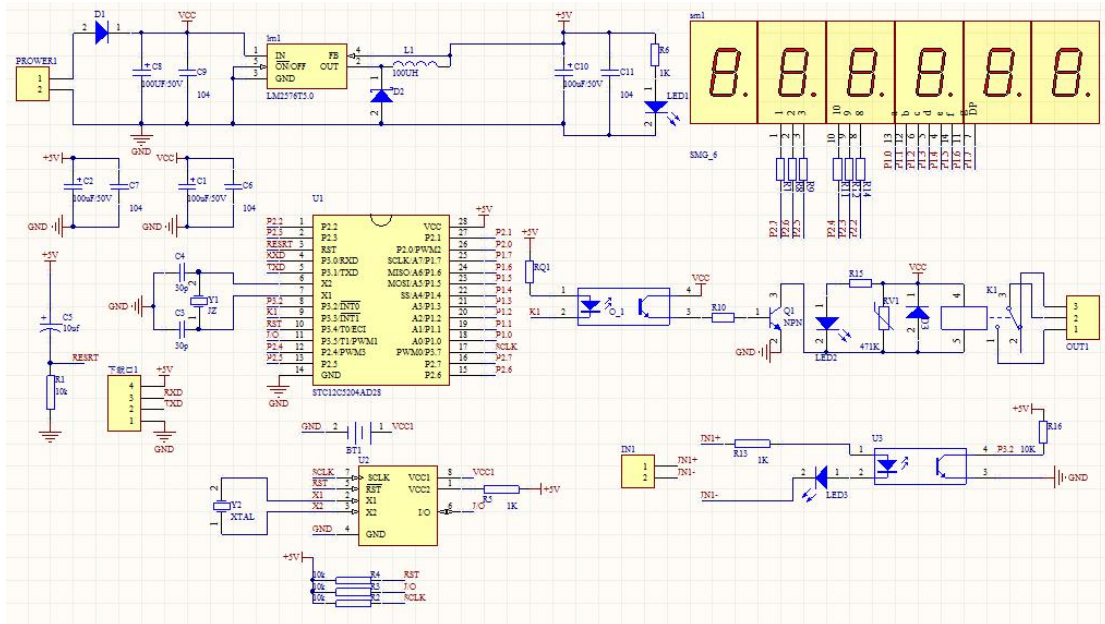
STC下载线 工控板
 GND-----G (GND)
 RXD-----T (P3.1)
 TXD-----R (P3.0)
 5V0-----V (+5V)



【原理图】 (提供 PDF 格式的原理图及 PCB 图)更清晰

免费提供与此工控板有关的：[资料](#)、[例程](#)、[原理图](#)

[芯片资料](#)、[软件](#)。



DS1302 时钟显示控制板

【DS1302 时钟显示控制板】

功能 1 描述

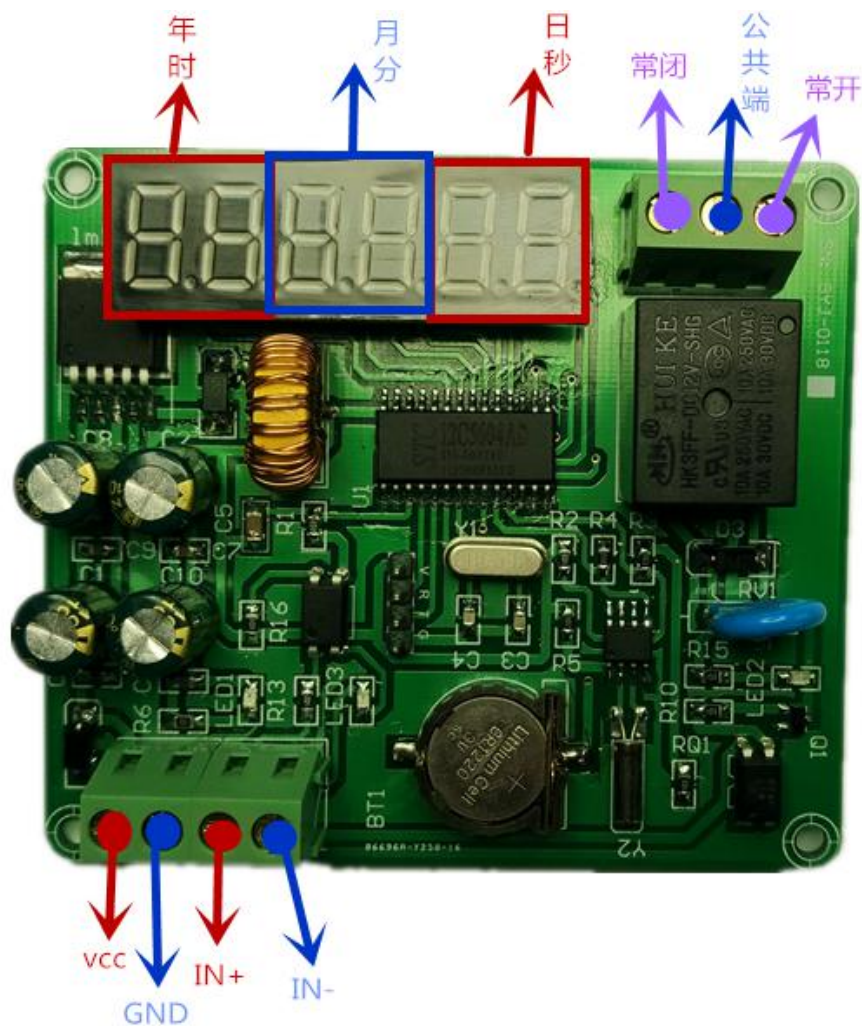
该程序主要基于 stc12c5604ad 开发，佩带有 ds1302 时钟芯片，并在六位数码管上显示年月日时分秒，使用串口发送数据，并使用触发端和输出端实现简单的控制。

下载程序之后，使用如下软件：



配置波特率及串口，串口根据客户需要进行配置，下拉找到相应的串口，配置之后点击“进入控制页面”，进入如下页面：





首先点击“写入时间”，这是数码管会显示写入的时间，先显示年月日，过五秒钟之后会停留显示时分秒。

当点击“写入时间”之后，软件会把“写入闹钟 1”的内容及按钮显示出来。这时可以设置闹钟 1 的时间，这个时间必须大于现在的时间，如果不大于，请修改。

当点击“写入闹钟 1”之后，且时间大于系统运行时间，继电器会吸合，数码管会显示写入的闹钟 1 的时间，先显示闹钟的年月日，过五秒钟之后显示闹钟时分秒，再过五秒显示系统运行的年月日，最后过五秒停留显示时分秒。软件中的“写入闹钟 2”的内容及按钮显示出来，这时可以设置闹钟 2 的时间，这个时间必须大于那闹钟 1 的时间，如果不大于，请修改。

当点击“写入闹钟 2”之后，且时间大于闹钟 1 的时间，数码管会显示写入的闹钟 2 的时间，先显示闹钟的年月日，过五秒钟之后显示闹钟时分秒，再过五秒显示系统运行的年月日，最后过五秒停留显示时分秒。

到此，软件使用完毕。

时间写入完成之后，板子开始自行工作，这是可以切换为外部供电，供电电压为 12V，上电之后，数码管先显示年月日，过五秒之后停留显示时分秒。此时继电器是吸合的，当到达时间，继电器断开，这时进行触发，继电器吸合，当时间到达第二时间之后，继电器断开。一个工作流程结束，如想再次运行，请再次使用上位机下发时间。

```

/*****
#include "STC12C56.H" //库文件
#include <intrins.H>
#define uchar unsigned char//宏定义无符号字符型
#define uint unsigned int //宏定义无符号整型
/*****串口配置*****/
unsigned char TORH = 0; //T0 重载值的高字节
unsigned char TORL = 0; //T0 重载值的低字节
extern void ConfigTimer0(unsigned int ms);
extern void ConfigUART(unsigned int baud);
xdata unsigned char sendBuf[7]={0x00,0x00,0x00,0x00,0x00}; //发送缓冲区
unsigned char sendPosi=0; //发送状态指针
unsigned int crcData=0; //CRC16 效验结果十六位
unsigned char sendCount=0; //需要发送的总个数
bit flag_zx=0;
bit bt1ms=0; //一毫秒定时器
unsigned char receCount=0; //接收的数据
unsigned char receTimeOut=0; //接收超时
xdata unsigned char receBuf[7]={0x00,0x00,0x00,0x00,0x00}; //接收缓冲区
void clear_receBuf();//清空发送缓冲区
void time0out(void); //定时把数据缓冲器归零，等待下次接收
void beginSend(void); //开始发送函数，发送前先写需要发送的
个数个发送的数组
void Input(); //获取输入函数
/*****串口配置*****/
/*****数码管配置*****/
uint tcnt1=0;//用于刷新数码管
uint bainum1,shinum1,genum1;//数码管上显示
uint bainum2,shinum2,genum2;//数码管上显示
uchar display1,display2,display3;//数码管上显示
xdata uchar seg7code[10]={0xc0,0xf9,0xa4,0xb0,0x99,0x92,0x82,0xf8,0x80,0x90}; //显示段码
数码管字跟
sbit C0 = P2^2; //第一位数码管
sbit C1 = P2^3; //第二位数码管
sbit C2 = P2^4; //第 3 位数码管
sbit C3 = P2^5; //第 4 位数码管
sbit C4 = P2^6; //第 3 位数码管
sbit C5 = P2^7; //第 4 位数码管
void z16to10();//十六进制转换成十进制进行数码管显示
void displaycut();//显示切换
/*****数码管配置*****/
/*****DS1302 配置*****/
sbit DS1302_CLK = P3^7;
sbit DS1302_IO = P3^5;

```

```

sbit DS1302_RST = P3^4;
xdata unsigned char second,minute,hour,week,day,month,year; //秒、分、时、星期、日、
月、年
xdata unsigned char time[]={0x00,0x00,0x00,0x00,0x00,0x00,0x00}; //初始时间数组
xdata unsigned char settime1[]={0x00,0x00,0x00,0x00,0x00,0x00,0x00}; //设置时间数组 1
xdata unsigned char settime2[]={0x00,0x00,0x00,0x00,0x00,0x00,0x00}; //设置时间数组 2
void read_time();//读 DS1302 时间子程序
void initial_ds1302();//初始化 DS1302 子程序
unsigned char read_ds1302(unsigned char addr);//读 DS1302 地址子程序
void write_ds1302(unsigned char addr,unsigned char TDat);//向 DS1302 某地址写一字节数据子
程序
unsigned char outputbyte(void);//读 DS1302 一个字节子程序
void InputByte(unsigned char byte1);//向 DS1302 送一字节数据子程序
/*****DS1302 配置*****/
/*****EEPROM 配置*****/
#define CMD_IDLE 0 //EEPROM 无操作
#define CMD_READ 1 //读取字节
#define CMD_PROGRAM 2 //写入字节
#define CMD_ERASE 3 //擦除字节
#define ENABLE_IAP 0X83 //编程周期由晶振决定（如果<12MHZ 选用此项）
#define IAP_ADDRESS 0X0000 //内部 EEPROM 地址
void IapEraseSector(uint addr);//擦除一个字节函数
void IapProgramByte(uint addr,uchar dat);//写入一个字节函数
uchar IapReadByte(uint addr);//读取一个字节函数
void IapIdle();//操作函数
bit flag_eeeprom1=1;//掉电存储的标志位
void saveeeepro();//掉电存储
/*****EEPROM 配置*****/
xdata uint fd1=0,fd2=0,fd3=0,fd4=0;//用于切换数码管显示部分
sbit ins=P3^2;//输入
sbit outs=P3^3;//输出
void comparesetime();//比较设置的时间
uint fcst1=0,fcst2=0,fcst3=0,fcst4=0;
uint fo1=1;
static uchar presstime=0;//时间值按键用到的
uint kf0=1;//按键用的标志位
uint kt0=0;//按键用的标志位
void comparesetime(){//比较设置的时间
    if((fcst1==1)&&(fcst2==1)){//fcst1 表示时间 fcst2 表示设置的时间 1
        //second,minute,hour,week,day,month,year; //秒、分、时、星期、日、月、
年
        if((second>=settime1[0])&&(minute>=settime1[1])&&(hour>=settime1[2])&&(day>
=settime1[4])&&(month>=settime1[5])&&(year>=settime1[6])){

```



```

settime1[4]=IapReadByte(0x09);//掉电存储用到
settime1[5]=IapReadByte(0x0a);//掉电存储用到
settime1[6]=IapReadByte(0x0b);//掉电存储用到
settime2[0]=IapReadByte(0x0c);//掉电存储用到
settime2[1]=IapReadByte(0x0d);//掉电存储用到
settime2[2]=IapReadByte(0x0e);//掉电存储用到
settime2[3]=IapReadByte(0x10);//掉电存储用到
settime2[4]=IapReadByte(0x11);//掉电存储用到
settime2[5]=IapReadByte(0x12);//掉电存储用到
settime2[6]=IapReadByte(0x13);//掉电存储用到
EA = 1;           //开总中断
TR0=1;
fd1=1;
ConfigTimer0(1); //配置 T0 定时 1ms
ConfigUART(9600); //配置波特率为 9600
while(1){
    read_time();//读取时间 second,minute,hour,week,day,month,year; //秒、分、时、
    星期、日、月、年
    Input();
    time0out();
    comparesetime();//比较设置的时间
    displaycut();//显示切换
    if(fo1==1){
        outs=1;
    }else if(fo1==0){
        outs=0;
    }
    if((fcst2==0)&&(fcst3==1)){
        if(ins==0){//当按键 1 按下
            if(kf0){//当标志位 kf0=1 的时候进行
                if(presstime>100){//当时间值大于一百的时候
                    kf0=0;//标志位 kf0=0
                    kt0=1;//标志位 kt0=0
                    flag_eeeprom1=1;
                    presstime=0;//时间清零，重新进行
                    fcst4=1;
                }
            }
        }else{//当按键松开
            if(kt0){//当标志位 kt0=1 的时候进行
                if(presstime>100){//当时间值大于一百的时候
                    kf0=1;//标志位 kf0=1
                    kt0=0;//标志位 kt0=0
                    presstime=0;//时间清零，重新进行

```

```

    }
}
if((flag_eeeprom1==1)){//加掉电存储，必需加到这，不然有一些
小问题
    flag_eeeprom1=0;
    saveeeepro();//掉电存储
}
} //按键 1 结束
}
}
}
}

```

```

void displaycut() { //显示切换
    if(fd3==0){
        if(fd4==0){
            display1=second;
            display2=minute;
            display3=hour;
        } else if(fd4==1){
            display1=settime1[0];
            display2=settime1[1];
            display3=settime1[2];
        } else if(fd4==2){
            display1=settime2[0];
            display2=settime2[1];
            display3=settime2[2];
        }
    } else if(fd3==1){
        if(fd4==0){
            display1=day;
            display2=month;
            display3=year;
        } else if(fd4==1){
            display1=settime1[4];
            display2=settime1[5];
            display3=settime1[6];
        } else if(fd4==2){
            display1=settime2[4];
            display2=settime2[5];
            display3=settime2[6];
        }
    }
}
}
}
}

```

```

        z16to10();//十六进制转换成十进制进行数码管显示
    }

void z16to10(){//十六进制转换成十进制进行数码管显示
    bainum1=display1%16;
    shinum1=display1/16;
    genum1= display2%16;
    bainum2=display2/16;
    shinum2=display3%16;
    genum2= display3/16;
}

void Input(){//输入检测函数
    if(flag_zx==1){
        flag_zx=0;
        sendCount=7;
        sendBuf[0]=receBuf[0];
        sendBuf[1]=receBuf[1];
        sendBuf[2]=receBuf[2];
        sendBuf[3]=receBuf[3];
        sendBuf[4]=receBuf[4];
        sendBuf[5]=receBuf[5];
        sendBuf[6]=receBuf[6];
        if(receBuf[0]==0x00){
            fd1=1;
            fd4=0;
            fcst1=1;
            time[6]=receBuf[1];
            time[5]=receBuf[2];
            time[4]=receBuf[3];
            time[3]=0x00;
            time[2]=receBuf[4];
            time[1]=receBuf[5];
            time[0]=receBuf[6];
            initial_ds1302();//初始化时间
        }
        if(receBuf[0]==0x01){
            fd1=1;
            fd4=1;
            fcst2=1;
            flag_eeprom1=1;
            settime1[6]=receBuf[6];
            settime1[5]=receBuf[5];
            settime1[4]=receBuf[4];
            settime1[3]=0x00;

```

```

        setttime1[2]=receBuf[3];
        setttime1[1]=receBuf[2];
        setttime1[0]=receBuf[1];
    }
    if(receBuf[0]==0x02){
        fd1=1;
        fd4=2;
        fcst3=1;
        flag_eeeprom1=1;
        setttime2[6]=receBuf[6];
        setttime2[5]=receBuf[5];
        setttime2[4]=receBuf[4];
        setttime2[3]=0x00;
        setttime2[2]=receBuf[3];
        setttime2[1]=receBuf[2];
        setttime2[0]=receBuf[1];
    }
        if((flag_eeeprom1==1)){//加掉电存储
            flag_eeeprom1=0;
            saveeeepro();//掉电存储
        }
    beginSend();
    clear_receBuf();//清空发送缓冲区
}
}
}

```

void InterruptTimer0() interrupt 1{//T0 中断服务函数，执行串口接收监控、并且进行数码管显示

```

    TH0 = TORH; //重新加载重载值
    TL0 = TORL;
    bt1ms=1;
    tent1++;//用于刷新数码管
    presstime++;
    if(fd1==1){
        fd2++;
        if(fd2==1){
            fd3=1;
        }
    }
    if(fd2==5000){
        fd3=0;
        if(fd4==0){
            fd3=0;
            fd1=0;
        }
    }
}

```

```

        fd2=0;
    }
}
if(fd2==9000){
    fd4=0;
    fd2=0;
}
}
switch(tcnt1){
    /******数码管 1-开始******/
    case 1: P2&=0X03; P1=seg7code[bainum1];C0=1;break;
    case 2: P2&=0X03; P1=seg7code[shinum1];C1=1;break;
    case 3: P2&=0X03; P1=seg7code[genum1]; C2=1;break;
    case 4: P2&=0X03; P1=0x7f;if(fd4==0){C2=1;}break;//小数点
    /******数码管 1-结束******/
    /******数码管 2-开始******/
    case 5: P2&=0X03; P1=seg7code[bainum2];C3=1;break;
    case 6: P2&=0X03; P1=seg7code[shinum2];C4=1;break;
    case 7: P2&=0X03; P1=0x7f;if(fd4==0){C4=1;}break;//小数点
    case 8: P2&=0X03; P1=seg7code[genum2]; C5=1;break;
    /******数码管 2-结束******/
    default: tcnt1=0; P2&=0X03; break;
}
}
}
/******串口配置开始******/
void ConfigTimer0(unsigned int ms){//配置并启动 T0, ms-T0 定时时间
    unsigned long tmp; //临时变量
    tmp = 11059200 / 12; //定时器计数频率
    tmp = (tmp * ms) / 1000; //计算所需的计数值
    tmp = 65536 - tmp; //计算定时器重载值
    tmp = tmp + 33; //补偿中断响应延时造成的误差
    TORH = (unsigned char)(tmp>>8); //定时器重载值拆分为高低字节
    TORL = (unsigned char)tmp;
    TMOD &= 0xF0; //清零 T0 的控制位
    TMOD |= 0x01; //配置 T0 为模式 1
    TH0 = TORH; //重新加载重载值
    TL0 = TORL;
    ET0 = 1; //使能 T0 中断
    TR0 = 1; //启动 T0
}
}
void ConfigUART(unsigned int baud){//串口配置函数, baud-通信波特率
    SCON = 0x50; //配置串口为模式 1
    TMOD &= 0x0F; //清零 T1 的控制位
    TMOD |= 0x20; //配置 T1 为模式 2
}
}

```

```

    TH1 = 256 - (11059200/12/32)/baud; //计算 T1 重载值
    TL1 = TH1; //初值等于重载值
    ET1 = 0; //禁止 T1 中断
    ES = 1; //使能串口中断
    TR1 = 1; //启动 T1
}
void timeOut(void){//接收缓冲区归零
    if(bt1ms==1){
        bt1ms=0;
        if(receCount>0){
            receTimeOut--;
            if((receTimeOut==0)&&(receCount>0)){
                receCount=0;
            }
        }
    }
}
void clear_receBuf(){//清空发送缓冲区
    unsigned char i;
    for(i=0;i<8;i++){
        receBuf[i]=0;
    }
}
void beginSend(void){//开始发送
    sendPosi=0;
    if(sendCount>1)
        sendCount--;
    SBUF=sendBuf[0];
}

void InterruptUART() interrupt 4{//串口中断服务函数
    if(RI){ //接收到新字节 当字节发送到结束位的一半的时候 RI 接收标志变为 1
        进入串口中断
        RI = 0; //清零接收中断标志位
        receTimeOut=10;
        receBuf[receCount]= SBUF;
        receCount++;
        if(receCount>6){
            flag_zx=1;
        }
        receCount &=0xff;
    }
    if(TI) { //字节发送完毕
        TI = 0; //清零发送中断标志位
    }
}

```

```

        if(sendPosi<sendCount){
            sendPosi++;
            SBUF=sendBuf[sendPosi];
        }
    }
}
/*****          串    口    配    置    结    束
*****/
/*****DS1302 开始*****/
void InputByte(unsigned char byte1){//向 DS1302 送一字节数据子程序
    char i;
    for(i=8;i>0;i--){
        DS1302_IO=(bit)(byte1&0x01);
        DS1302_CLK=1;
        _nop_();
        DS1302_CLK=0;
        byte1>>=1;
    }
    return;
}
unsigned char outputbyte(void){//读 DS1302 一个字节子程序
    unsigned char i;
    unsigned 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);
}
void write_ds1302(unsigned char addr,unsigned char TDat){//向 DS1302 某地址写一字节数据子
程序
    DS1302_RST=0;
    _nop_();
    DS1302_CLK=0;
    _nop_();
    DS1302_RST=1;
    InputByte(addr);
    _nop_();
    InputByte(TDat);
    DS1302_CLK=1;

```

```

        _nop_();
        DS1302_RST=0;
    }
unsigned char read_ds1302(unsigned char addr){//读 DS1302 地址子程序
    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);
}
void initial_ds1302(){//初始化 DS1302 子程序
    write_ds1302(0x8e,0x00);    //写保护寄存器，在对时钟或 RAM 写前 WP 一定
    要为 0
    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
}
void read_time(){//读 DS1302 时间子程序
    second=read_ds1302(0x81);    //秒寄存器
    minute=read_ds1302(0x83);    //分
    hour=read_ds1302(0x85);    //时
    week=read_ds1302(0x8B);    //星期
    day=read_ds1302(0x87);    //日
    month=read_ds1302(0x89);    //月
    year=read_ds1302(0x8d);    //年
}
/*****EEPROM 开始*****/
void saveepro(){//掉电存储
    IapEraseSector(0);    //擦除 0x01 地址中的数据    一定要先
    擦除再写进 同一地址
    IapProgramByte(0x01,fcst1);    //擦除完成就可以写入了
    IapProgramByte(0x02,fcst2);    //擦除完成就可以写入了
}

```

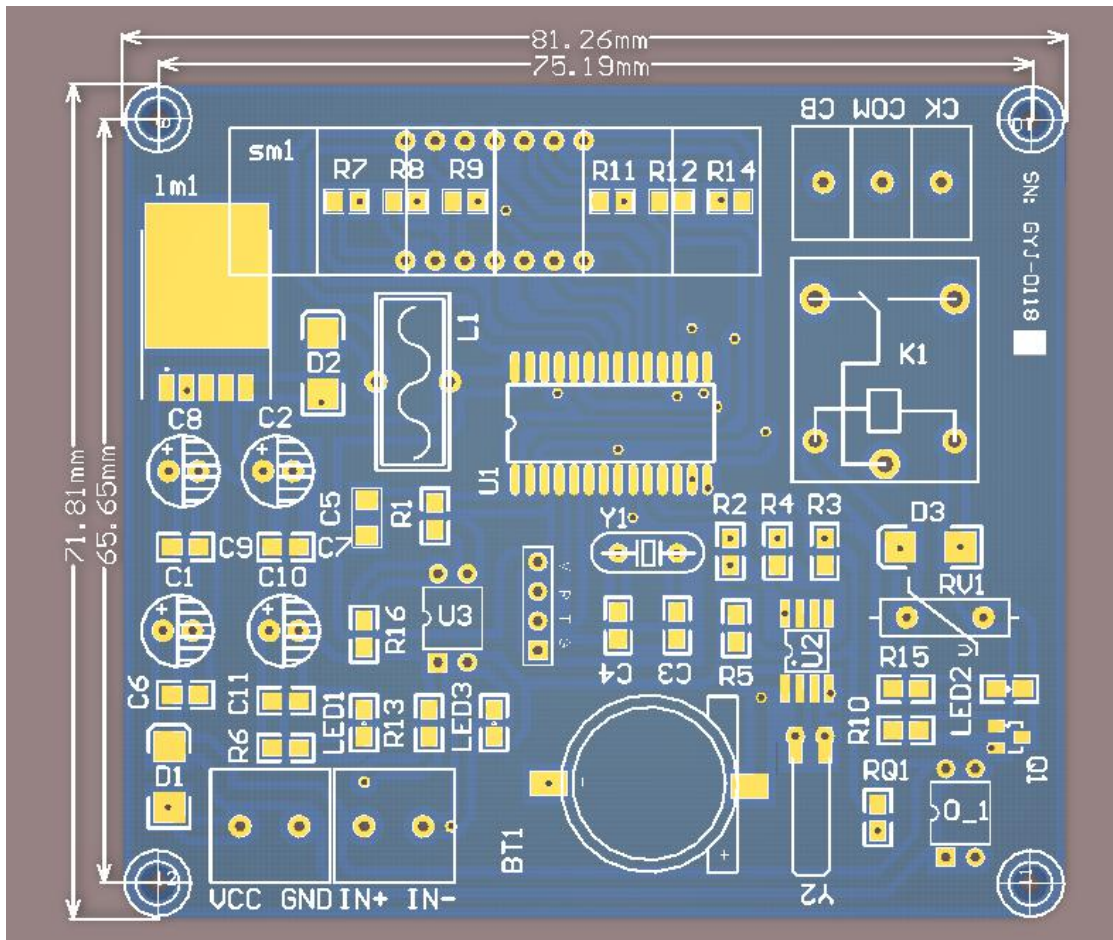
```

        IapProgramByte(0x03,fcst3);        //擦除完成就可以写入了
        IapProgramByte(0x04,fcst4);        //擦除完成就可以写入了
        IapProgramByte(0x05,settime1[0]); //擦除完成就可以写入了
        IapProgramByte(0x06,settime1[1]); //擦除完成就可以写入了
        IapProgramByte(0x07,settime1[2]); //擦除完成就可以写入了
        IapProgramByte(0x08,settime1[3]); //擦除完成就可以写入了
        IapProgramByte(0x09,settime1[4]); //擦除完成就可以写入了
        IapProgramByte(0x0a,settime1[5]); //擦除完成就可以写入了
        IapProgramByte(0x0b,settime1[6]); //擦除完成就可以写入了
        IapProgramByte(0x0c,settime2[0]); //擦除完成就可以写入了
        IapProgramByte(0x0d,settime2[1]); //擦除完成就可以写入了
        IapProgramByte(0x0e,settime2[2]); //擦除完成就可以写入了
        IapProgramByte(0x10,settime2[3]); //擦除完成就可以写入了
        IapProgramByte(0x11,settime2[4]); //擦除完成就可以写入了
        IapProgramByte(0x12,settime2[5]); //擦除完成就可以写入了
        IapProgramByte(0x13,settime2[6]); //擦除完成就可以写入了
    }
}
void IapIdle(){//操作函数
    IAP_CONTR=0;
    IAP_CMD=0;
    IAP_TRIG=0;
    IAP_ADDRH=0X80;
    IAP_ADDRL=0;
}
uchar IapReadByte(uint addr){//读取一个字节函数
    uchar dat;
    IAP_CONTR=ENABLE_IAP;
    IAP_CMD=CMD_READ;
    IAP_ADDRL=addr;
    IAP_ADDRH=addr>>8;
    IAP_TRIG=0X46;
    IAP_TRIG=0XB9;
    _nop_();
    _nop_();
    _nop_();
    dat=IAP_DATA;
    IapIdle();
    return dat;
}
void IapProgramByte(uint addr,uchar dat){//写入一个字节函数
    IAP_CONTR=ENABLE_IAP;
    IAP_CMD=CMD_PROGRAM;
    IAP_ADDRL=addr;
    IAP_ADDRH=addr>>8;
}

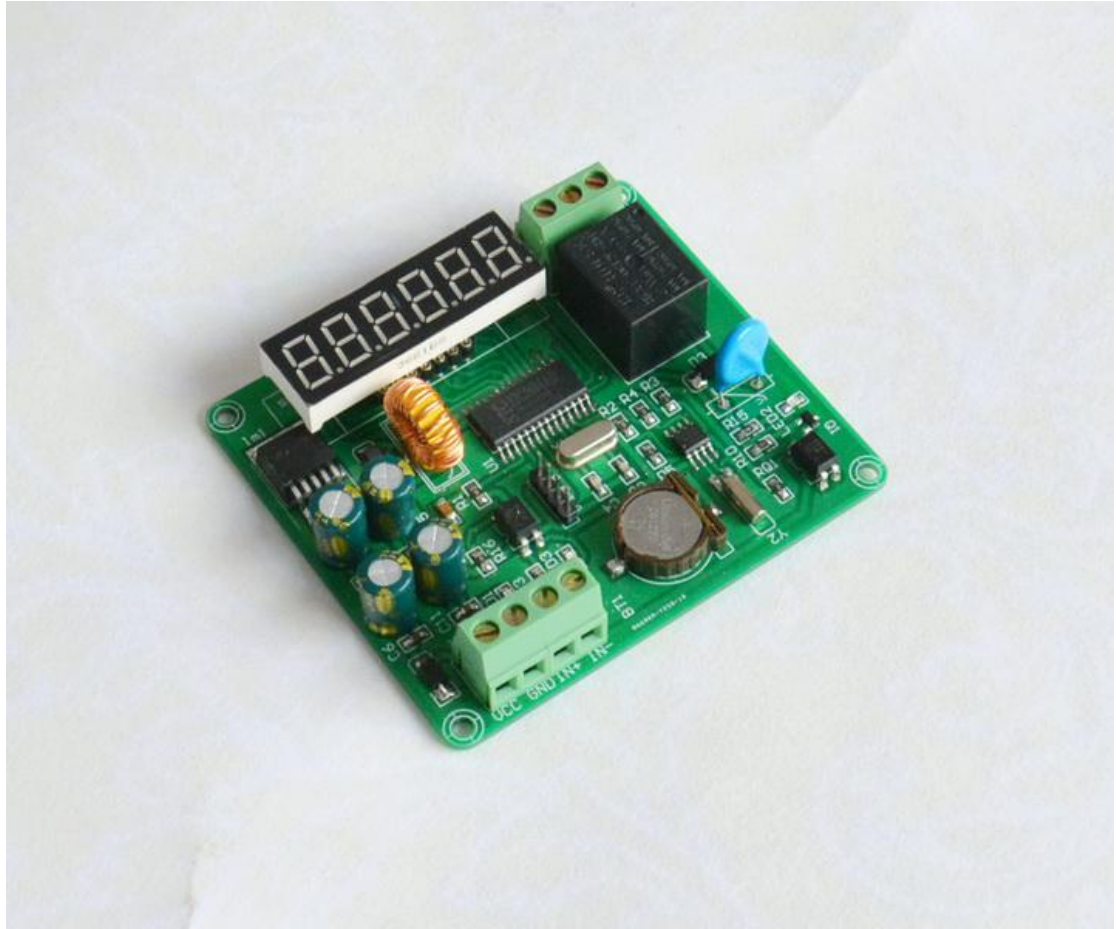
```

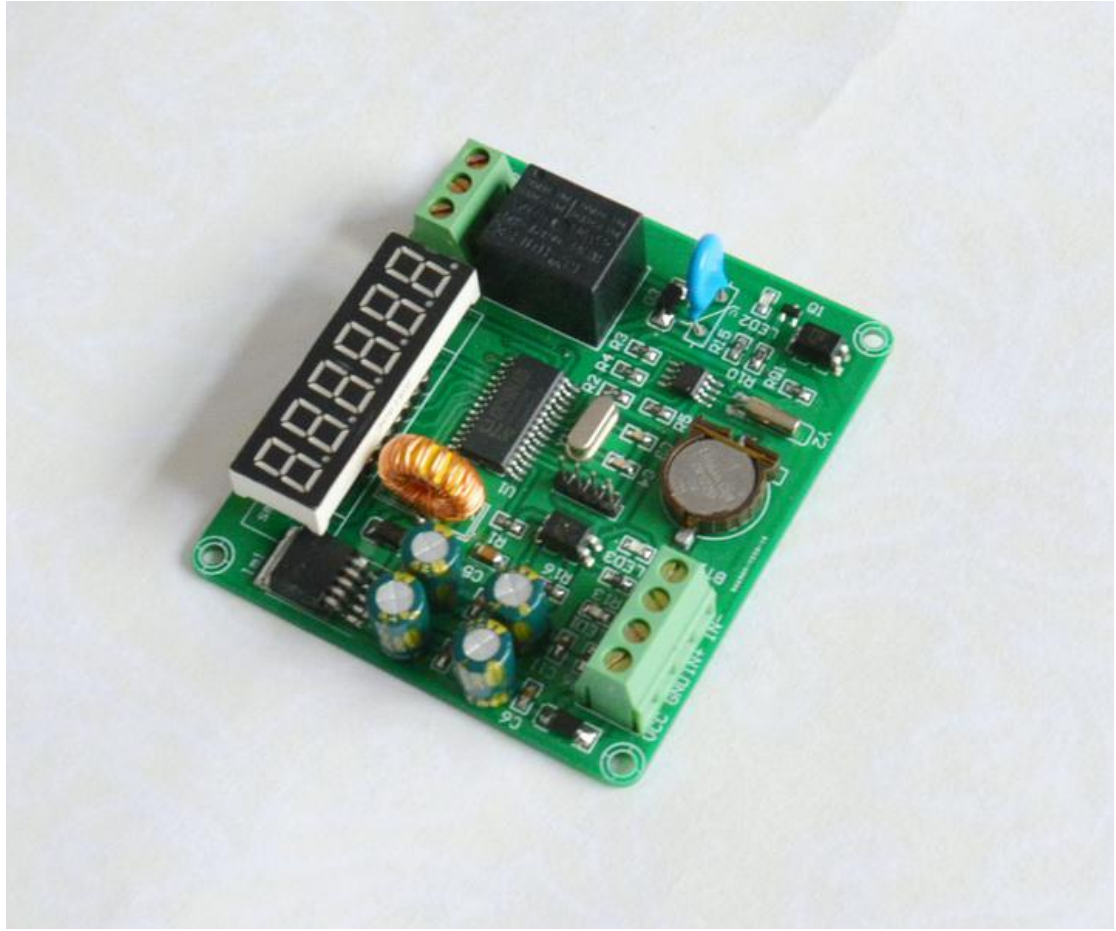
```
IAP_DATA=dat;
IAP_TRIG=0X46;
IAP_TRIG=0XB9;
_nop_();
_nop_();
_nop_();
IapIdle();
}
void IapEraseSector(uint addr){//擦除一个字节函数
    IAP_CONTR=ENABLE_IAP;
    IAP_CMD=CMD_ERASE;
    IAP_ADDRL=addr;
    IAP_ADDRH=addr>>8;
    IAP_TRIG=0X46;
    IAP_TRIG=0XB9;
    _nop_();
    _nop_();
    _nop_();
    IapIdle();
}
/*****EEPROM 结束*****/
```

【尺寸图】



【图片展示】









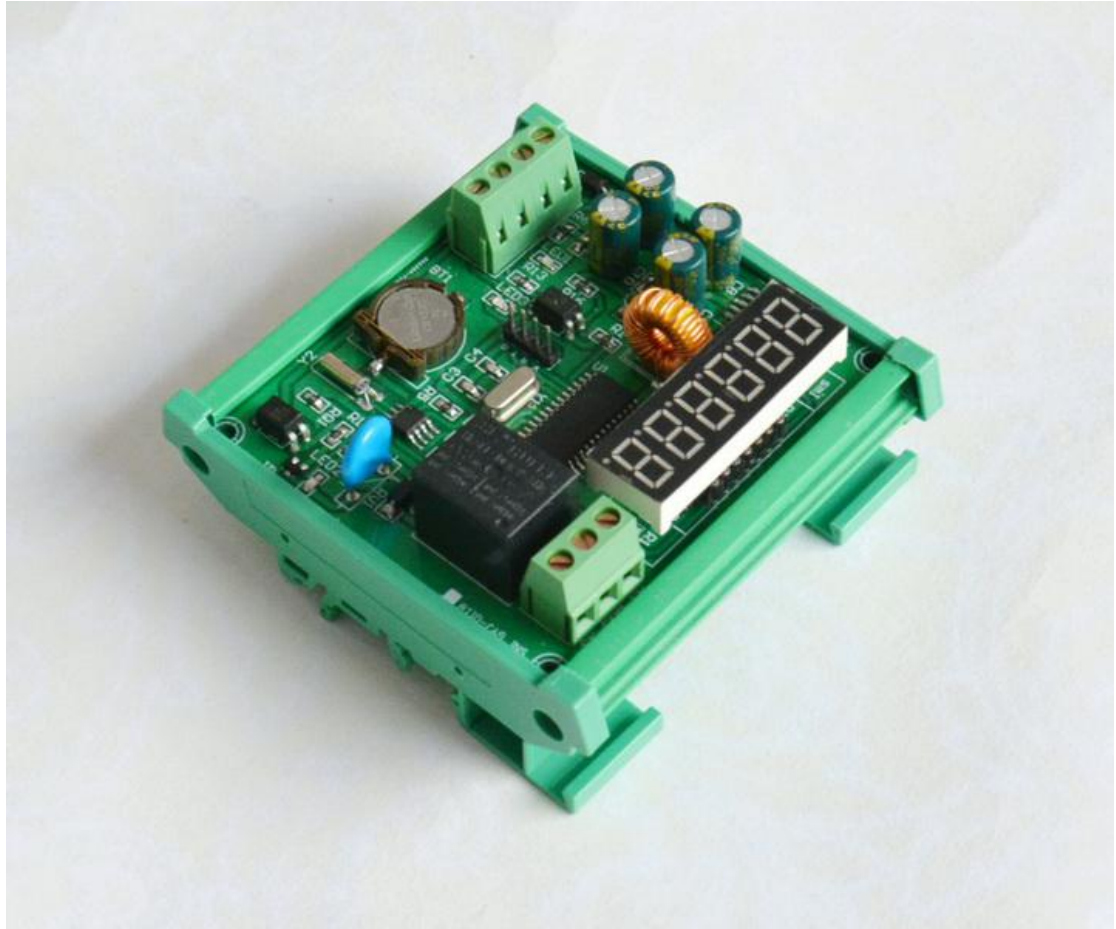


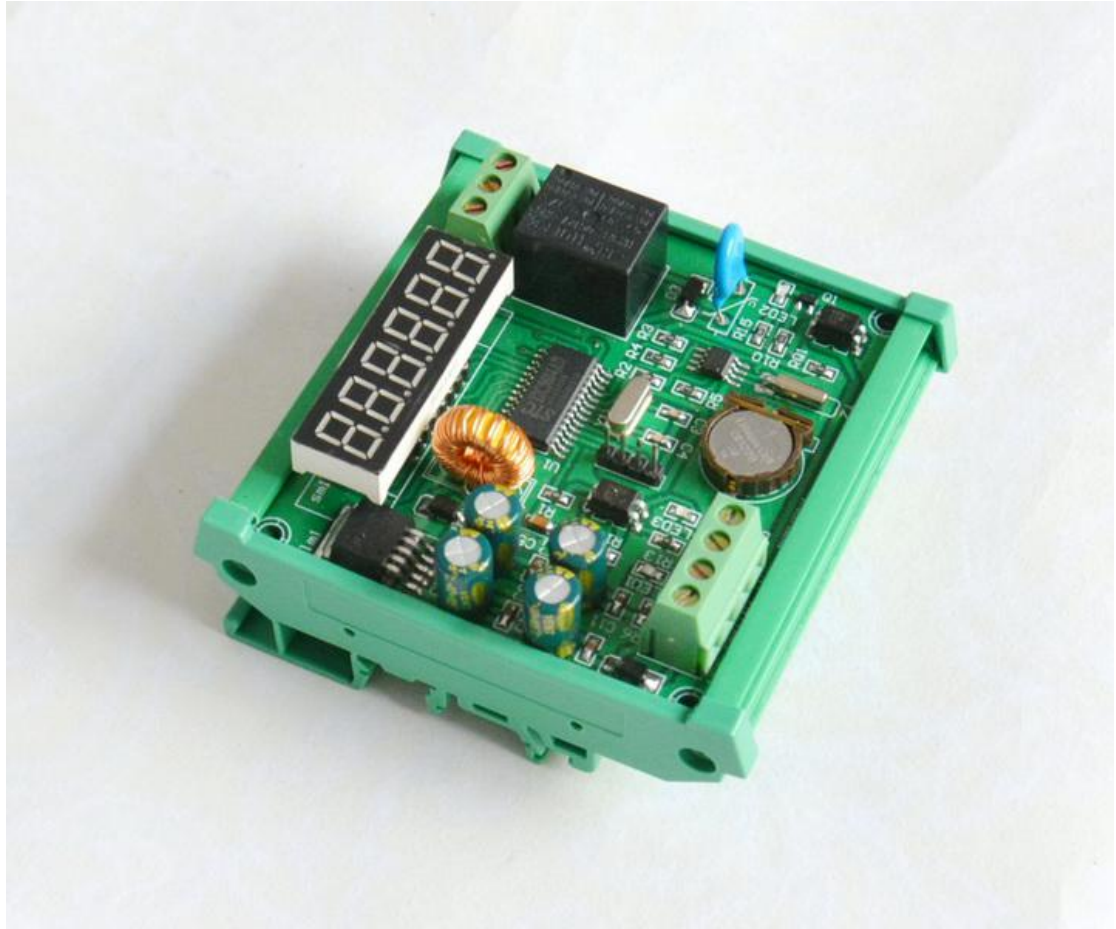


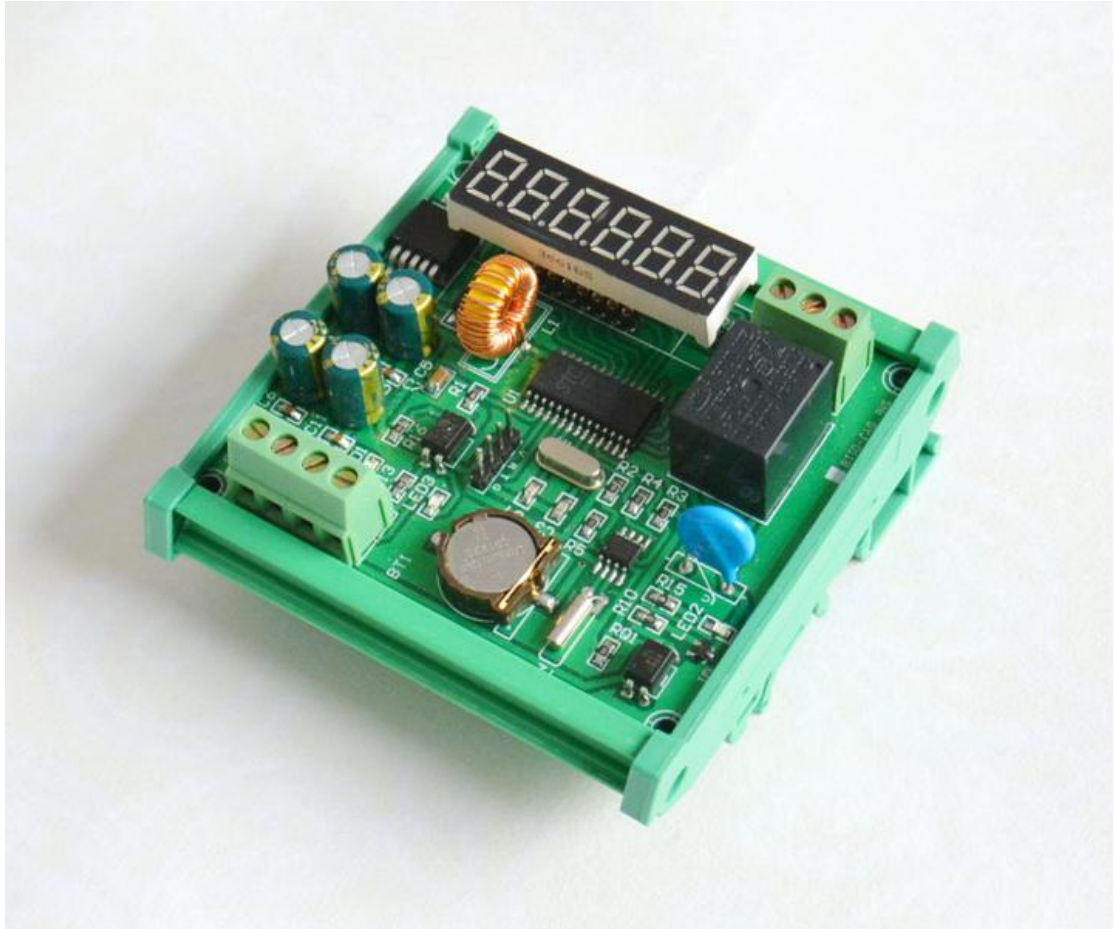
【加装配套外壳效果】











在线销售产品技术支持联系信息: 13603455408 QQ: 115451619

电路设计 项目定制 产品开发: 15981910271 (微信同号)

产品有售淘宝 1店: <https://ourhc.taobao.com>

产品有售淘宝 2店: <https://g88888.taobao.com>

产品有售淘宝企业店: <https://shop404420384.taobao.com>

SMT 贴片加工 电子元件焊接 联系生产部: 卢经理 电话

13503710441 (微信同号) 零元起步 不限数量, 不限价格!