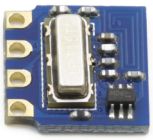
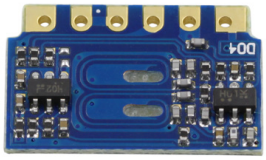
遥控模块软件编解码说明





	型号	频率	调制方式	灵敏度	发射功率	电压	电流	尺寸
发射模块 	H34A-315	315Mhz	ASK		15db@12V	4.2-12V	7mA	10.5*10.5*2.6
	H34B-315	315Mhz	ASK		12db@3V	2-4.2V	7mA	10.5*10.5*2.6
	H34A-433	433Mhz	ASK		15db@12V	4.2-12V	7mA	10.5*10.5*2.6
	H34B-433	433Mhz	ASK		12db@3V	2-4.2V	7mA	10.5*10.5*2.6
接收模块 	H3V4F	433Mhz	ASK	-102db		2.7-5.3V	0.28mA	12.0*18.0*2.10
	H3V3E	315Mhz	ASK	-102db		2.7-5.3V	0.28mA	12.0*18.0*2.10
	H5V4D	433Mhz	ASK	-102db		4.5-5.5V	1.2mA	12.0*18.0*2.10
	H5V3M	315Mhz	ASK	-105db		4.5-5.5V	1.9mA	12.0*18.0*2.10

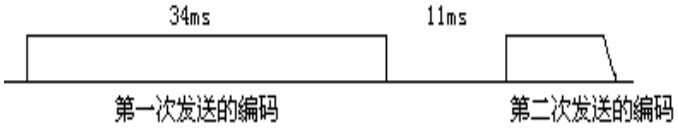
PT2262 软件解码

一 概述

PT2262/2272 是一种 CMOS 工艺制造的低功耗低价位通用编解码电路，是目前在无线通讯电路中作地址编码识别最常用的芯片之一。PT2262/2272 最多可有 12 位(A0-A11)三态地址端管脚(悬空,接高电平,接低电平),任意组合可提供 531441 地址码,PT2262 最多可有 6 位(D0-D5)数据端管脚,设定的地址码和数据码从 17 脚串行输出。

PT2262/2272 必须用相同地址码配对使用，当需要增加一个通讯机时，用户不得不求助于技术人员或厂家来设置相同地址码，客户自己设置相对比较麻烦，尤其对不懂电子的人来说。随着人们对操作的要求越来越高，PT2262/2272 的这种配对使用严重制约着使用的方便性，人们不断地要求使用一种无须请教专业人士，无须使用特殊工具，任何人都可以操作的方便的手段来弥补 PT2262/2272 的缺陷，这就是 PT2262 软件解码。

二 解码原理



上面是 PT2262 的一段波形，可以看到一组一组的字码，每组字码之间有同步码隔开，所以我们如果用单片机软件解码时，程序只要判断出同步码，然后对后面的字码进行脉冲宽度识别即可。

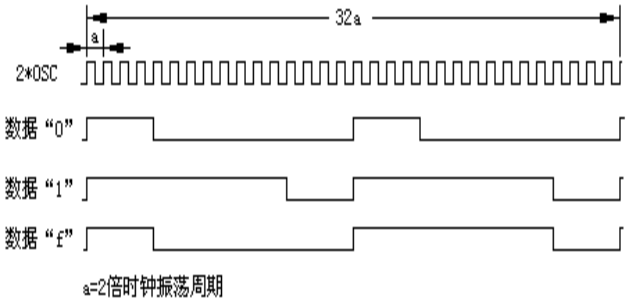
2262 每次发射时至少发射 4 组字码，2272 只有在连续两次检测到相同的地址码加数据码时才会把数据码中的“1”驱动相应的数据输出端为高电平和驱动 VT 端同步为高电平。因为无线发射的特点，第一组字码非常容易受零电平干扰，往往会产生误码，所以程序可以丢弃处理。

下面我们来仔细看一下 PT2262 的波形特征：

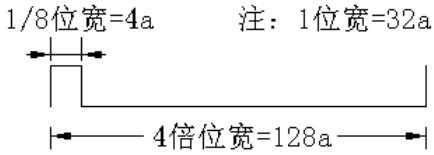
振荡频率 f=2*1000*16/Rosc(kΩ) kHz 其中 Rosc 为振荡电阻

这里我们选用的是一种比较常用的频率 f≈10 kHz, Rosc=3.3MΩ (以下同)。

下图是振荡频率与码位波形的对应关系：

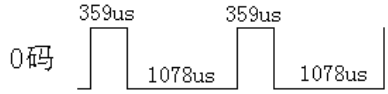


同步码头波形：

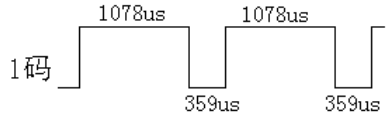


PT2262 有三种编码：0，1，和悬空(表示为 f)。

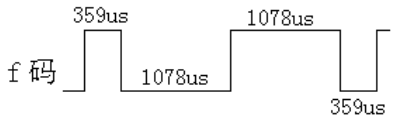
1、 数据“0”发送的码位如下：



2、 数据“1”发送的码位如下：



3、 数据“f”发送的码位如下：



有了以上具体的波形，我们就可以进行软件解码了。T2262 每次至少发送 4 次编码，首先我们可以通过检测 11ms 宽度的同步码头，有码头才开始进行编码解码，无码头则继续等待。当收到码头时，还要检测是否已经收到过码头，若无，则丢弃第一次编码的信号，以防止误码。

从编码图中可以看出，每一位码字都是从低电平开始到高电平，到低电平，再到高电平。为了检测方便，在接收端我们把编码信号进行了 180° 倒相，使码位开始的上升沿转化为下降沿，这样当我们使用 MCS51 系列单片机解码时可使用中断方式及时截获编码。从编码图中还可以看出，每一位码字都可以分成两段，我们以每段中的电平宽度来描述码位：

码位	第一段	第二段	数值表示	反码表示
0	窄	窄	00	11
1	宽	宽	11	00
f	窄	宽	01	10
无效码	宽	窄	10	01

软件解码方法 1(反码)：

从第一个下降沿开始延时 700us 左右，检测电平高低，记为 A1，再检测第二个下降沿，延时 700us 左右，检测电平高低，记为 A2，这样一个码位就可以译出来了，连续检测 12 个码位。

软件解码方法 2(反码)：

从第一个下降沿开始记时，并不断检测电平变化，一有电平变化，立即记录电平宽度 B1，再继续记时直至出现第二个下降沿，记录两个下降沿的间隔 B2，重复以上步骤，得到 B3，B4，判断 B1，B2，B3，B4 是否在各自允许的误差范围内，是则保存 B1，B3，译出一个码位，否则认为误码，丢弃。连续正确检测 12 个码位。

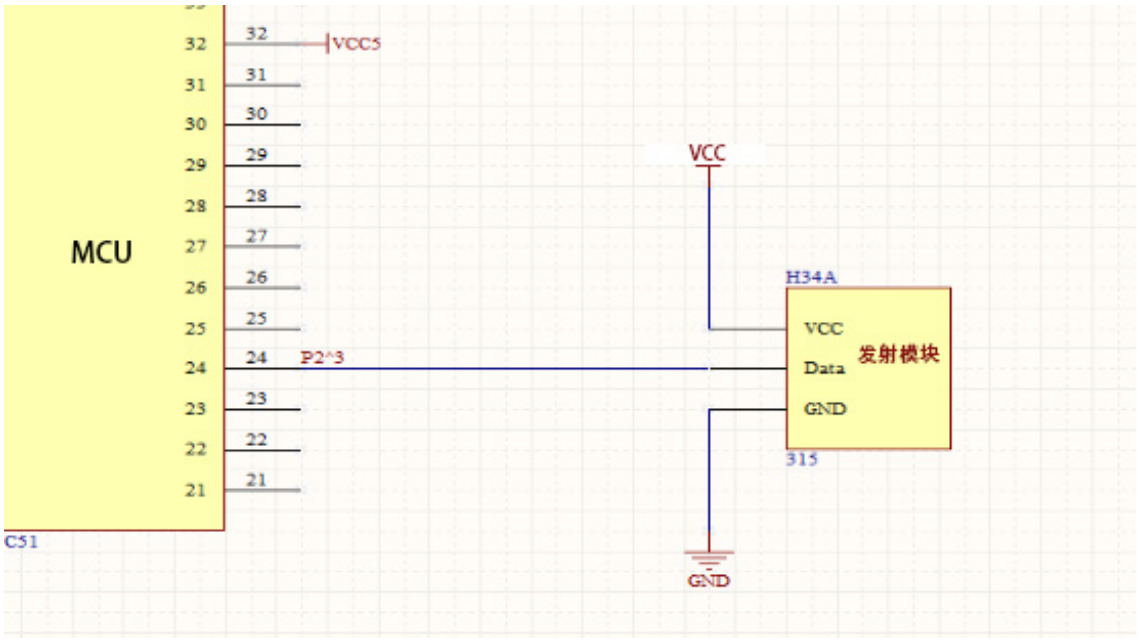
两种解码方式各有优缺点如下：

解码方式	优点	缺点
1	程序简单，CPU 开销少	解码精度差
2	程序复杂，CPU 开销大	解码精度较高

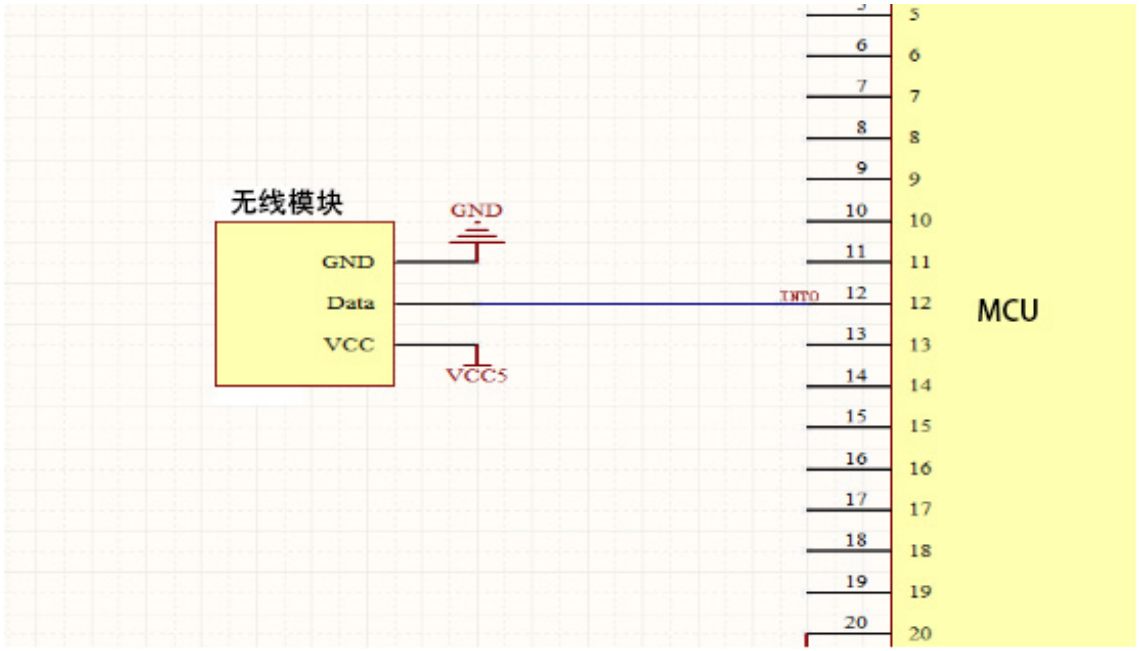
为了获得较高的解码精度，我们推荐使用方法 2，以避免大量的干扰信号的误解码。

三、硬件连接

3.1 发射部分



3.2 接收部分



四、参考解码软件 A

说明：ADD1，ADD2 中为 8 位地址，DAT0 中为 4 位数据

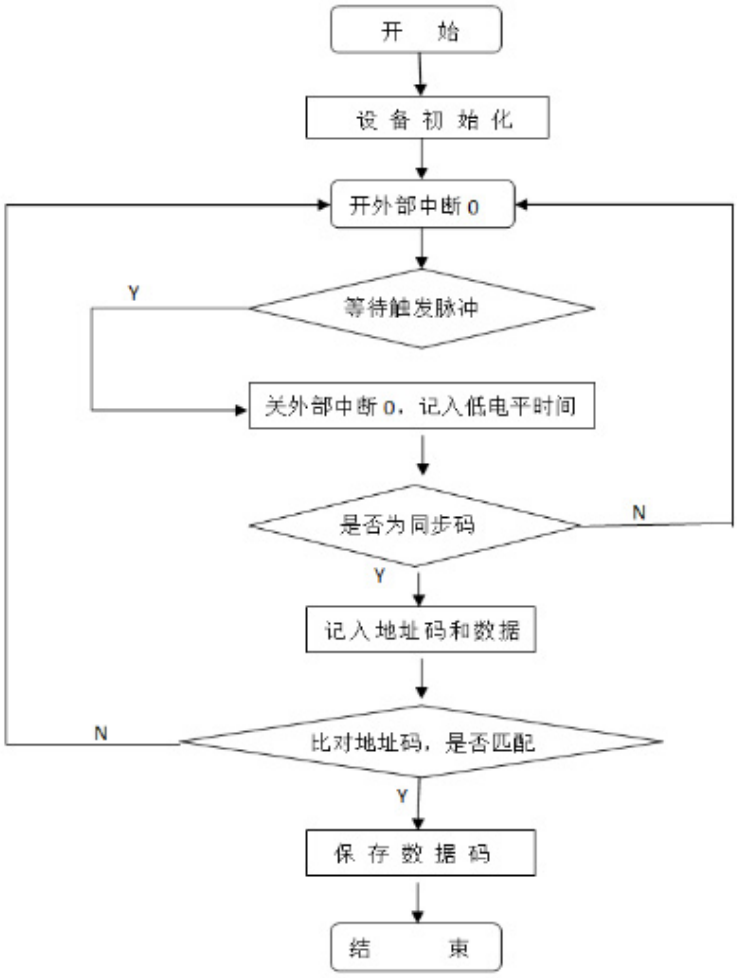
```
REMOTE: CLR     TR2                     ;探头信号检测子程序
        CLR     RECEIVE                 ;
        MOV     DETE_LOOP,#12           ;接收12位编码
REM00:  CLR     DETE_T_OVER             ;
          MOV     TH2,#0FEH               ;测第1位电平宽度
        MOV     TL2,#041H               ;
        SETB    TR2                      ;
REM01:  JB      REM,REM02               ;等待出现高电平
        JB      DETE_T_OVER,REM03       ;限时1500us,超时则认为误码
        AJMP    REM01                   ;
REM02:  MOV     A,TH2                   ;测低电平宽度,0FF为宽脉冲，0FE为窄脉冲
        CJNE    A,#0FFH,REM04           ;
        MOV     A,TL2                   ;
        CLR     C                       ;
        CJNE    A,#098H,$+3             ;
        JNC     REM03                   ;电平过宽(超过1150us)，退出
        CLR     C                       ;
        CJNE    A,#020H,$+3             ;
        JC      REM03                   ;电平过窄(小于780us)，退出
        SETB    C                       ;
        AJMP    REM05                   ;
REM03:  AJMP    REMOTE_END              ;
REM04:  CJNE    A,#0FEH,REM03           ;
        MOV     A,TL2                   ;
        CLR     C                       ;
        CJNE    A,#0C7H,$+3             ;
        JNC     REM03                   ;电平过宽(超过450us)，退出
        CLR     C                       ;
        CJNE    A,#060H,$+3             ;
        JC      REM03                   ;电平过窄(小于210us)，退出
        CLR     C                       ;
REM05:  MOV     A,DAT0                  ;存储电平值
        RLC     A                       ;
        MOV     DAT0,A                  ;
        MOV     A,ADD1                  ;
        RLC     A                       ;
        MOV     ADD1,A                   ;
REM06:  JNB     REM,REM07               ;等待出现低电平
        JB      DETE_T_OVER,REM03     ;脉冲下降沿间隔限时1500us,超时则认为误码
        AJMP    REM06                   ;
REM07:  CLR     TR2                     ;
        CLR     DETE_T_OVER             ;
        MOV     A,TH2                   ;
        CJNE    A,#0FFH,REM13           ;脉冲间隔过小
        MOV     A,TL2                   ;
        CLR     C                       ;
        CJNE    A,#050H,$+3             ;
        JC      REM13                   ;电平过窄(小于1200us)，退出
        MOV     TH2,#0FEH               ;测第2位电平宽度
        MOV     TL2,#041H               ;
        SETB    TR2                      ;
REM11:  JB      REM,REM12               ;等待出现高电平
        JB      DETE_T_OVER,REM13      ;限时1500us,超时则认为误码
```

```
        AJMP    REM11                   ;
REM12:  MOV     A,TH2                   ;测低电平宽度,0FE为宽脉冲，0FF为窄脉冲
        CJNE    A,#0FFH,REM14           ;
        MOV     A,TL2                   ;
        CLR     C                       ;
        CJNE    A,#098H,$+3             ;
        JNC     REM13                   ;电平过宽(超过1100us)，退出
        CLR     C                       ;
        CJNE    A,#020H,$+3             ;
        JC      REM13                   ;电平过窄(小于1000us)，退出
        SETB    C                       ;
        AJMP    REM15                   ;
REM13:  AJMP    REMOTE_END              ;
REM14:  CJNE    A,#0FEH,REM13           ;
        MOV     A,TL2                   ;
        CLR     C                       ;
        CJNE    A,#0C7H,$+3             ;
        JNC     REM13                   ;电平过宽(超过450us)，退出
        CLR     C                       ;
        CJNE    A,#060H,$+3             ;
        JC      REM13                   ;电平过窄(小于210us)，退出
        CLR     C                       ;
REM15:  MOV     A,TEMP                  ;存储电平值
        RLC     A                       ;
        MOV     TEMP,A                  ;
        MOV     A,ADD2                  ;
        RLC     A                       ;
        MOV     ADD2,A                  ;
REM16:  JNB     REM,REM18               ;等待出现低电平
        JB      DETE_T_OVER,REM13       ;脉冲下降沿间隔限时1500us,超时则认为误码
        AJMP    REM16                   ;
REM17:  AJMP    REM00
REM18:  CLR     TR2                     ;
        CLR     DETE_T_OVER             ;
        MOV     A,TH2
        CJNE    A,#0FFH,REM13           ;脉冲间隔过小
        MOV     A,TL2                   ;
        CLR     C                       ;
        CJNE    A,#050H,$+3             ;
        JC      REM13                   ;电平过窄(小于1200us)，退出
        DJNZ    DETE_LOOP,REM17         ;
REM19:  MOV     DETE_LOOP,#4            ;把接收的编码左移4位
REM20:  CLR     C                       ;将8位密码放在同一字节上
        MOV     A,DAT0                  ;
        RLC     A                       ;
        MOV     DAT0,A                  ;
        MOV     A,ADD1                  ;
        RLC     A                       ;
        MOV     ADD1,A                  ;
        CLR     C                       ;
        MOV     A,TEMP                  ;
        RLC     A                       ;
        MOV     TEMP,A                  ;
        MOV     A,ADD2                  ;
        RLC     A                       ;
        MOV     ADD2,A                  ;
        DJNZ    DETE_LOOP,REM20         ;
;把4 位数据编码由高4 位移到低4 位上      ;
```

```
        MOV     A,DAT0                  ;
        SWAP    A                       ;
        MOV     DAT0,A                  ;
        MOV     A,TEMP                  ;
        SWAP    A                       ;
        MOV     TEMP,A                  ;
        ANL     DAT0,#0FH               ;
        SETB    RECEIVE                 ;
REMOTE_END:                             ;
        CLR     TR2
        CLR     REMOTING                ;
        RET                             ;
```

五 参考解码软件 B

5.1 程序流程图



5.2 PT2262 编码部分程序：

```
//*********************************
#include <reg52.h>//头文件
/******* 发送数据“1”---12a 高+4a 低**************/
void sendbit_1()
{
        REM=1;
        TH1=TH1_12a;//12a
        TL1=TL1_12a;//12a
        TR1=1;
        while(TR1);//高电平
        REM=0;
        TH1=TH1_4a;//4a
        TL1=TL1_4a;//4a
        TR1=1;
        while(TR1);//低电平
}
/******* 发送数据“0”---4a 高+12a 低**************/
void sendbit_0()
{
        REM=1;
        TH1=TH1_4a;//4a
        TL1=TL1_4a;//4a
        TR1=1;
        while(TR1);//高电平
        REM=0;
        TH1=TH1_12a;//12a
        TL1=TL1_12a;//12a
        TR1=1;
        while(TR1);//低电平
}
/******* 发送数据“f”---4a 高+50ms 低**************/
//同步头
void sendRF_sof()
{
        REM=1;
        TH1=TH1_4a;//4a
        TL1=TL1_4a;//4a
        TR1=1;
        while(TR1);//高电平
        REM=0;
        TH1=(65536-20000)/256;//50ms
        TL1=(65536-20000)%256;//
        TR1=1;
        while(TR1);//低电平
}
// 发送一组编码，连续发送四次
void RFSendData(uchar ADDER, uchar Dat1)
```

```
{
        sendRF_sof();
        SendINT(ADDER);
        SendINT(Dat1);
        sendRF_sof();
        SendINT(ADDER);
        SendINT(Dat1);
        sendRF_sof();
        SendINT(ADDER);
        SendINT(Dat1);
        sendRF_sof();
        SendINT(ADDER);
        SendINT(Dat1);
}
```

5.3 PT2272 解码部分程序：

根据以上的单片机模拟编码，我们可以知道，单片机每次至少发送 4 组数据，每组数据之前都有一个同步头，而同步头之后才是地址码+数据码。所以我们首先要做的就是判断同步头。根据采集同步头的时间来确定是不是数据代码。若是，则开始连续接收 12 位数据，最后再进行判定。

5.3.1 同步头的检测：

单片机外部中断 0 引脚用于接收数据，当单片机检测到触发信号，进入中断程序后，关闭外部中断 0，开启定时器 0 记录低电平持续的时间，并检测是否为同步码。若是，则关闭外部中断，并开始接收地址码跟数据码；若不是，则重新打开中断，继续检测同步码。

```
void accept_tongbu() interrupt 0
{
 EX0=0;
 TR0=1;
 while(!INT0);     //等待同步码
 TR0=0;
 time=TH0*256+TL0;
 TH0=0;
 TL0=0;
 if((time>11000)&&(time<23000))          //检测同步码
  {
              Chuli();//开始接收地址码和数据码
  }
else
   EX0=1;
}
```

5.3.2 接收数据：

当检测到同步码时，则开始接收地址码跟数据码。当判定是正确的同步码后，随之打开定时器 0，利用 while(IR==1)，计算出高电平的时间，根据发送端编写的协议判定高电平的时间是位“1”还是“0”，连续这样接收 8 位组成一个 Byte,保存在数组 temp[]

中。连续接收两次，分别将地址码和数据码保存在 temp[1]和 temp[0]中后，然后进行判定是否所发送的数据，最后再重新打开中断。

```
bit DeCode(void)
{
    unsigned char j,i=0;
    unsigned char temp[2]={0};     //储存解码出的数据
      for(i=2;i>0;i--)   //每个数据包含两个数据码（8+8）
    {
      for(j=0;j<8;j++)  //每个码有 8 位数字
    {
    temp[i-1]=temp[i-1]<<1;  //temp 中的各数据位右移一位，因为先读出的是高位数据
    TH0=0;          //定时器清 0
    TL0=0;          //定时器清 0
     TR0=1;          //开启定时器 T0
    while(IR==1);   //如果是高电平就等待
     TR0=0;          //关闭定时器 T0
    HighTime=TH0*256+TL0;     //保存高电平宽度
    if((HighTime>700)&&(HighTime<1105))
                temp[i-1]|=1;
    while(IR==0);
    }
    }
          addless=temp[1];
          dataless=temp[0];
          return 1;

}
```

六 EV1527解码程序

```
//EV1527解码程序:单片机PIC16F630,内部4MHz,EV1527发射频率433MHz,振荡电阻270K,遥控器供电电压DC12V,周期为1.2ms。
//在上电3S钟内按开机键，程序学习遥控器识别码，1次只能学习一个遥控器；按关机键程序将擦除所有遥控器识别码。
//本程序最多可以学习42个遥控器，当遥控器个数满42个后第43个将会覆盖第1个。

#include <pic.h>
__CONFIG(0x34);
__EEPROM_DATA(0x02,0xff,0xff,0xff,0xff,0xff,0xff,0xff);


#define uchar unsigned char
#define uint  unsigned int

#define rfin         RC3
#define SLED        RA5
#define       LED                   RC1

//==================//
bit    learnf;
bit    fun_out;
bit    fun_ok;

//=======================//
//记时间变量
uint   temp0=0;
//=========================//
//RF接收变量
uchar  rec_count;
uchar  lo_buf;
uchar  hi_buf;
uchar  cs2;
uchar  cs1;
uchar  cs0;

//RF接收标志
bit    rfstart1;
bit    rfstart;
bit    rfok;
bit  rf_er;//接收失败
//-------------------------------延时-------------------------//
delay(uchar ms)
   {
     uint i,j;
     for(i=0;i<ms;i++)
     for(j=0;j<30;j++);
   }

//------------------------------ LED闪烁-------------------------//
LED_FLASH(void)
  {
    uchar i;
    for(i=0;i<6;i++)
```

```
     {
       LED=1;
       delay(30);
       LED=0;
     }
   }
//--------------------------------- 中断服务----------------------------//
void interrupt all ()
{
    GIE=0;
       temp0++;
if(TMR1IF==1)
                    {
                    TMR1IF=0;//清中断标志位
            TMR1H=0xFF;
                    TMR1L=0xAA;

//RF接受程序

if(rfin==1)
    {
      if(rfstart1==1)//是否找到同步头？如果找到同步头就由低到高开始接收
                {          //A
                  if(lo_buf>hi_buf)//判断是否低电平
                                  {

if(lo_buf>=8&&lo_buf<=12&&hi_buf>=3&&hi_buf<=4)
                                          {
                                             rec_count++;//是低电平
                                             cs0<<=1;//左移动一位

                                    }
                                     else   rf_er=1;//接收失败
                                }
                    else if(lo_buf<hi_buf)//判断是否高电平
                                  {

if(lo_buf>=3&&lo_buf<=4&&hi_buf>=8&&hi_buf<=12)//判断高电平是否大于900us和低电平
大于200us
                                    {
                                           rec_count++;//是高电平
                                           cs0<<=1;//移动一位
                                           cs0++;

                                        }
                                 else      rf_er=1;

                                  }
                else if(lo_buf==hi_buf) rf_er=1;       //高低电平周期相等，接收错
误
                if(rf_er==1)
                        {
                            rfstart=0;//接收失败
                            rec_count=0;
                            rfok=0;
                          }
                Else
```

```
                    {
                            if(rec_count==8){cs2=cs0;}
                            else if(rec_count==16){cs1=cs0;}
                        }
                if(rec_count==24)  rfok=1;//接收成功了
                        hi_buf=0;
                }       //if(rfstart1==1)函数结尾

    else    if(lo_buf>=30&&lo_buf<=120)//判断同步头
            {
                    rfstart=1;//检测到正确的同步头

                    rec_count=0;//接收位数,清除0


            }

               hi_buf++;

               lo_buf=0;
               rfstart1=0;
            rf_er=0;//接收失败标志
   }

else
  {      //输入端口为低电
        lo_buf++;
        if(rfstart==1)  rfstart1=1;  //开始接收标志
        else  hi_buf=0;
   }
}
 GIE=1;
}//TMR1IF==1函数内

//-------------------------------- 擦除EEPROM---------------------------//
C_EEPROM(void)
     {
          uchar i,b=0xff;
             for(i=0;i<=127;i++){EEPROM_WRITE(i=i,b);}
        EEPROM_WRITE(0,2);
        LED_FLASH();
      }

//-------------------------------- 保存用户码---------------------------//
void save_learn(void)
       {
       uchar i,j;
       TMR1IE=0;
       i=EEPROM_READ(0);
       EEPROM_WRITE(i,cs2);
       i++;
       EEPROM_WRITE(i,cs1);
       i++;
       j=(cs0&0xf0);
       EEPROM_WRITE(i,j);
       if(i==127) i=1;
       i++;
```

```
        EEPROM_WRITE(0,i);
        delay(2);
        TMR1IE=1;
        }
//------------------------------ 初始化-------------------------//

void init(void)//初始化函数
       {
              CMCON=0X07;  //关闭比较器
              TRISA=0x18;  //RA4 MRLC
         PORTA=0x00;
              TRISC=0x08;
         PORTC=0x00;
              GIE=0;       //关总中断
         TMR1IF=0;
              PEIE=1;
              TMR1IE=1;    //开启定时中断1
              TMR1ON=1;  //使用定时器1
         TMR1H=0xff;
              TMR1L=0xaa;//定义100uS中断一次
         GIE=1;
           }
//------------------------------ 主函数-------------------------//
main()
  {
    init();
    while(temp0<20000)  //学习或擦除
      {
        if(rfok)
         {
                if(cs0&0x01)  //学习
                {
                   save_learn();
                   LED_FLASH();
                   break;
                 }
                else if(cs0&0x02)//擦除
                 {
                   C_EEPROM();
                   LED_FLASH();
                 }
          }
       }
    while(1)
      {
       if(rfok==1)//接收完成，开始将接收到的用户码同学习到储存在EEPROM中的用户码
对比
        {
             uchar i=2,j,k,l,m,sw;
             for(m=0;m<42;m++)      //循环对比42次
                {
                 j=EEPROM_READ(i);
                 i++;
                 k=EEPROM_READ(i);
                 i++;
                 l=EEPROM_READ(i);
                 i++;
```

```
                if(j==cs2&&k==cs1&&(l==(cs0&0xf0)))    {sw=1;break;} //对比结果
相同
                else  sw=0;                                          //对比结果
不相同
              }
            if(sw==1)
             {
              if(cs0&0x04)
              LED=1;
              if(cs0&0x08)
              LED=0;
              if(cs0&0x01)
              SLED=1;
              if(cs0&0x02)
              SLED=0;
             }
          }
        }
    }
//*************************************程序结束
**********************************//
```

七、常见问题

1　接收模块杂波如何处理？

答：在接收模块一上电时，模块的数据脚会输出一个大约在 100hz 左右的正弦波（并非标准），这个波形的来源有两个部分，分别是因为这个是模块本身元器件产生的和空气中干扰造成，但在接收到信号时，就会输出一个正确的解码方波，对正常解码并无太大的影响，特别是对于那种固定解码芯片 pt2262/2272 来说，不会造成漏码和误码等问题，直接接线就好。而对于用单片机解码，大多数的客户使用的都是由外部中端来接接收模块的 data 脚，那么由于我们的接收模块在空闲状态下依旧会产生上下边沿的杂波，那么可能会不断引起单片机的误判，从而有可能漏掉对正确信号的解码，对于这样的问题，客户可采用软件屏蔽的办法，一般的编码信号都会有同步头这样的前端编码，而这也是判断正确编码的开始，所以在中断程序里面尽量只做同步头的检测就好，同时可以的话，也可以做超时的检测，这样以便能尽快退出中断，为接收下一串的信号做准备。

2　如何避免硬件干扰问题？

在无线通讯中使用单片机会对通讯系统造成严重的干扰。如果硬件设计不当，会造成原先硬件解码时通讯距离为 50 米以上，而用软件解码后可能只有十几米，因此解决硬件抗干扰问题在很大程度上可减少软件解码的误码率。

a、单片机振荡频率：大量的 MCS51 教材中推荐大家使用的是 12 MHz 及 11.0592 MHz 的晶体，这些晶体在一般场合使用没有问题，但在此却不可以，它们在 300 MHz 左右仍然能够产生较大的干扰，为解决单片机运行速度与电磁干扰的矛盾，建议采用低频率晶振如 4 MHz 或 3.58 MHz。

b、隔离：为了有效抑制单片机对接收模块的电磁干扰，建议采用电源隔离或端口隔离；端口隔离可采用三极管或比较器。采用隔离的效果非常明显。

3 高频电路的 PCB 线路如何排布效果好?

答：设计 PCB 时应注意：需要提供低阻抗电源和最小噪声辐射的地线。要求使用双面 PCB 板，并把地线平面放在底层以减少无线电的辐射和串扰；旁路电容应尽量靠近每个电源引脚 VDD；不要把 PCB 通孔与复俣地线相连；为减少电路中的分布电容，应避免平行线路的出现；线路应越短越好；为防止耦合，应独立其各组成部分；使用接地线使各信号隔离。

4 遥控距离有多远?

答：我们所说的遥控距离是收发模块天线处于垂直状态，离地 1.5 米高，工作于直线开阔地上测得的最大可解码距离，如果双方都处在较高的位置，则遥控距离还将更远。

由于工作在 UHF 频段内，电磁波沿直线传播，遇到障碍物会激剧衰减，遥控距离明显缩短，故使用时应尽量避开障碍物，或尽量架高天线。

数据速率对通信距离也有较大影响， 一般而言，速率越高，距离就越近，建议数据速率取 1.2K 以下比较好。

5 超外差和超再生模块有何区别?

答：超再生解调电路也称超再生检波电路，它实际上是工作在间歇振荡状态下的再生检波电路。一般再生检波电路在中波段工作时灵敏度很高，所以常用来制作简易晶体管收音机。对于工作于短波段的无线遥控或通信设备，再生检波的灵敏度及稳定性都不符合要求。但超再生检波在短波段却具有很高的灵敏度，在接收弱信号时放大率可达几十万倍。因此，对于希望电路简单、灵敏度高，而对选择性和信噪比要求不高的简单无线遥控通信设备(如防盗器等产品)，超再生检波电路还是颇有实用价值的。

通常超再生接收机的灵敏度约－100DBM 左右，所用器件多，稳定性差，加工复杂。

超外差式解调电路与超外差收音机相同，它是设置一本机振荡电路产生振荡信号，与接收到的载频信号混频后，得到中频(一般为 465kHz)信号，经中频放大和检波，解调出数据信号。由于载频频率是固定的，所以其电路要比收音机简单一些。

超外差接收机灵敏度可达－100～110DBM，而且外围元件少，集成化程度高，适合大规模生产。超外差接收机有声表稳频和 LC 稳频的两种，采用 LC 稳频的灵敏度高可达-105DBM 左右，但是稳定性稍差，而声表稳频的灵敏度约－100DBM 左右，稳定性好。

超外差接收机对天线的阻抗匹配要求较高，要求外接天线的阻抗必须是 50 欧姆的，否则对接收灵敏度有很大的影响，要尽可能减少天线根部到发射模块天线焊接处的引线长度，如果无法减小，可以用特性阻抗 50 欧姆的射频同轴电缆连接（天线焊点右侧有一个专门的接地焊点）。

超再生与超外差比较：

超再生式接收机具有电路简单、成本低廉的优点所以被广泛采用，而超外差接收机价格较高，温度适应性强，接收灵敏度更高，而且工作稳定可靠，抗干扰能力强，产品的一致性好，接收机本振辐射低，无二次辐射，性能指标好，容易通过 FCC 或者 CE 等标准的检测，符合工业使用规范。

6 如何选择天线?

答：天线对于无线模块很重要，直接影响通信距离，推荐以下 4 种天线。

a：线径 0.5mm 导线，433.92MHz 用 17cm，315MHz 23cm，尽量将天线拉直放，避开金属。

b：弹簧天线，我司有出售 315/433Mhz 配套的弹簧天线，效果不错。

c：对于 H3V3E 和 H3V4F，支持短天线。直径 0.5mm、长 4cm(433.92MHz)或 5cm(315MHz)。

d：PCB 天线 ，PCB 天线效果一般，设计难度较大，我司可以提供设计服务。

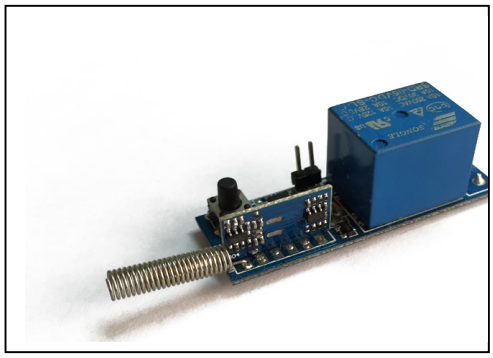
7 是否需要在 DAT 端附加上拉/下拉电阻及电容？

答：不用，可直接与编解码芯片或 MCU 连接

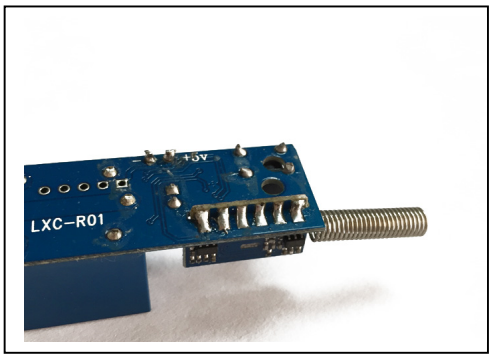
8 怎么焊接模块到 PCB 上较好？

a：直插式

在 PCB 上开槽，把模块插进去焊接引脚，此方式简单方便，如下图：



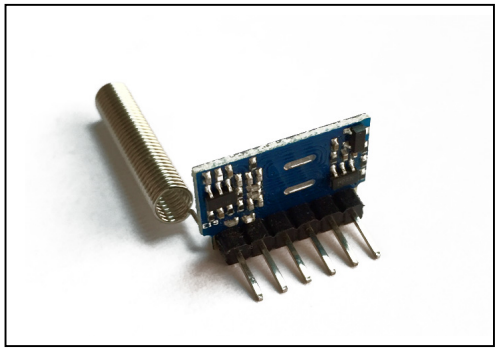
正面

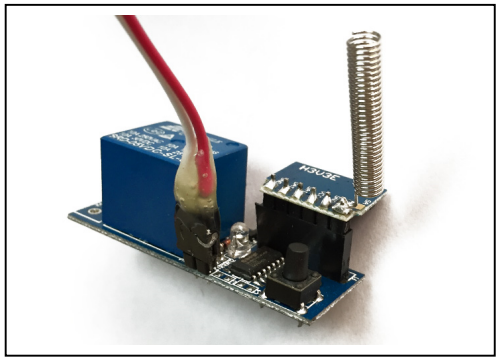


背面

b：插针式

需要自己给模块焊接上排针 ，PCB 上焊接排座，此方式成本较高，不推荐。





c：贴片式

在 PCB 上做个封装，把无线模块贴上去焊接，此方式最方便，推荐