HT47R20A-1 红外载波输出（IR-carrier） 使用介绍

文件编码：HA0035s

本文主要介绍 HT47R20A-1 红外载波输出（IR-carrier）的使用及注意事项。

介绍

HT47R20A-1 提供了一个与 PA2 共用引脚的红外载波发射端口，可以由掩膜选择来决定。

当选择红外输出时，置 PA2 为“ 0 ”（CLR PA.2）可以允许红外载波输出，置 PA2 为“ 1 ”（SET PA.2）则禁止红外载波输出并且 PA2 口输出为低电平。红外载波的输出频率为系统时钟的 12 分频，当系统频率为 480kHz 时，载波频率为 40kHz。

PA2	功能
0 （CLR PA.2）	PA2=红外载波输出端
1 (SET PA.2)	PA2=0

红外载波输出（IR-carrier）的使用

下面的程序介绍了红外载波输出的具体使用。此程序使 PA2 口产生一个周期性的红外载波输出，根据不同的编码规则，可以更改输出周期，从而达到数字信号传输的功能。

掩膜设定：

红外载波输出允许：IR output 设定为 PA2 IR output enable

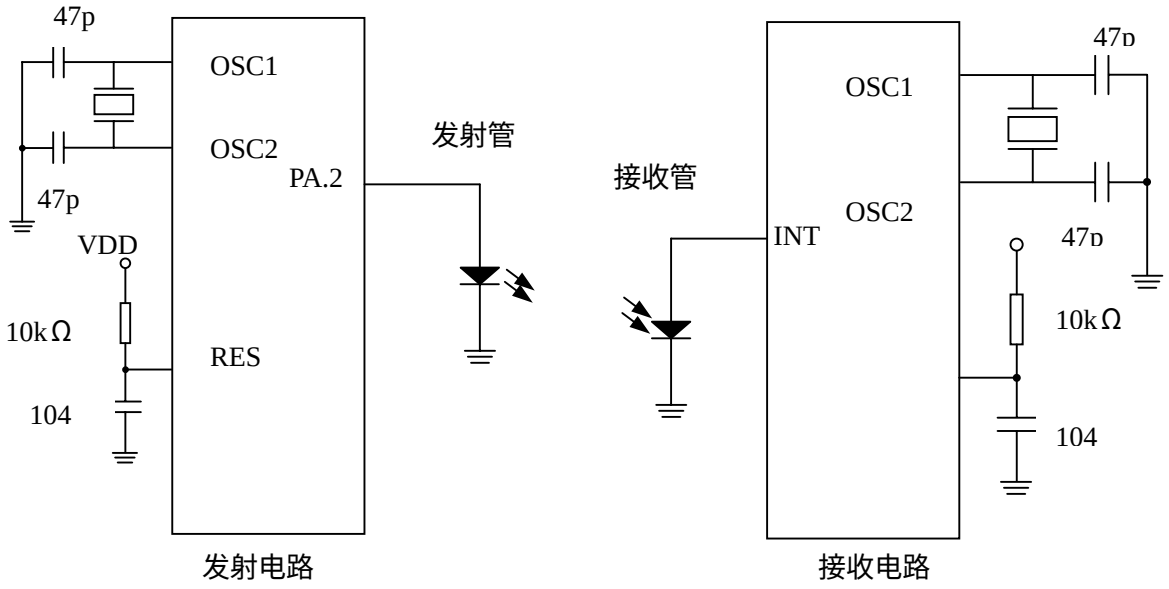
程序清单：

```
include ht47r20a-1.inc
data .section 'data'
count1 db ?
count2 db ?

code .section at 0 'code'
       org    00h
       jmp    start
;-----------------------------------------
       org    20h
start:
       clr    intc0
       clr    intc1
loop:  clr    pa.2          ;红外载波输出允许
       call   delay         ;延时
```

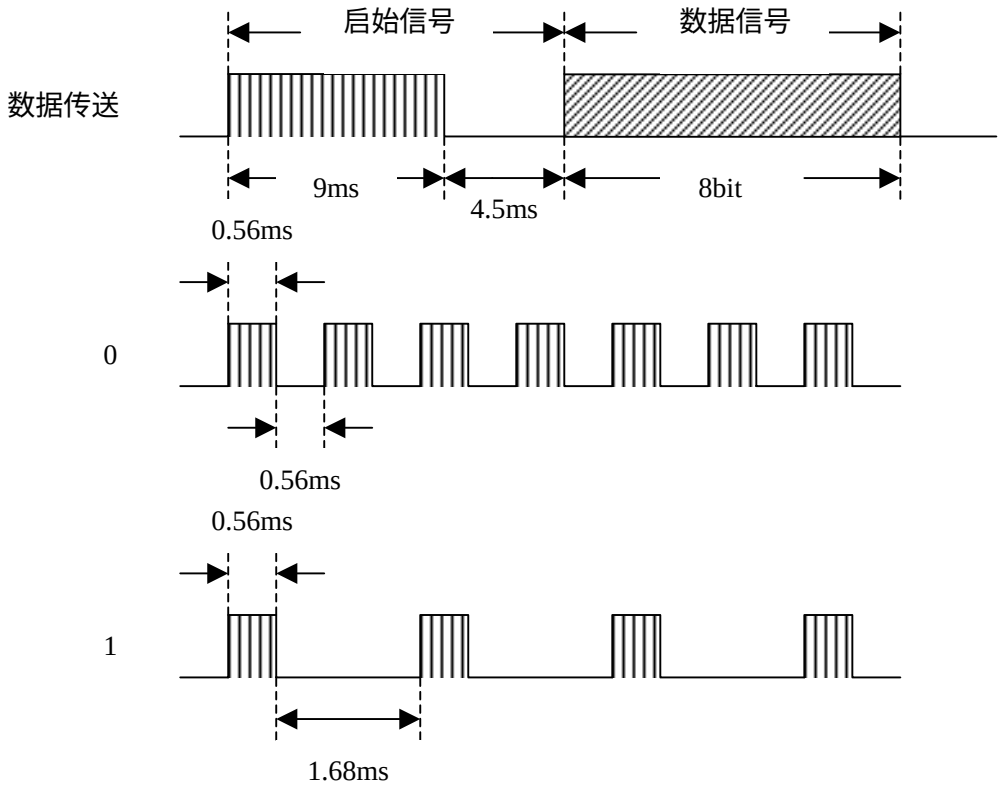
```
        set     pa.2            ；关闭红外载波输出
        call    delay           ；延时
        jmp     loop
；----------------------------------------
delay proc                      ；延时子程序
        mov     a,04h
        mov     count1,a
        mov     count2,a
d:      sdz     count1
        jmp     d
        sdz     count2
        jmp     d
        ret
delay endp
```

下面以具体例子来说明 IR 的发射与接收。
硬件电路：



用两个 HT47R20A-1 单片机来实现 IR 的发射和接收。其编码方式为：1 位启始信号+8 位数据信号。

数据传送协议和时序图如下：



软件部分：

在掩膜选项中选择 PA2 IR output enable

1．IR 的发射：

数据信号的发射主要通过控制 IR 的开关状态来完成。以下程序能完成 8bit 的信号的发送，其编码为：1 位启始信号+8 位数据信号，并且循环发射。

```
include ht47r20a-1.inc
data .section 'data'
SEND_DATA     equ    12h              ；要发送的数据
buffer    db  ?                       ；发送控制寄存器
count     db  ?                       ；发送 bit 的位数控制
d_count   db  ?                       ；延时变量

code .section at 0 'code'
       org    00h
       jmp    start
       org    04h
       reti
       org    08h
       reti
```

```
       org    0ch
       reti
       org    10h
       reti
;--------------------------------
       org    20h
start: clr    intc0
       clr    intc1
loop:  clr    pa.2                    ;发送启始位
       call   delay1                  ;延时 9ms
       call   delay1
       set    pa.2
       call   delay1                  ;延时 4.5ms
       mov    a,08h                   ;总共发射一个字节
       mov    count,a
       mov    a,SEND_DATA             ;将数据送至控制寄存器
       mov    buffer,a
       clr    c
next_bit:
       rlc    buffer                  ;从高位到低位发送
       snz    c
       jmp    send_0                  ;发送 0
       jmp    send_1                  ;发送 1
back:  sdz    count                   ;发送返回
       jmp    next_bit
       jmp    loop
;--------------------------------
send_0:                               ;发送 0
       clr    pa.2                    ;载波允许
       call   delay2                  ;延时 0.56ms
       set    pa.2                    ;载波关闭
       call   delay2                  ;延时 0.56ms
       jmp    back                    ;返回
;--------------------------------
send_1:                               ;发送 1
       clr    pa.2                    ;载波允许
       call   delay2                  ;延时 0.56ms
       set    pa.2                    ;载波关闭
       call   delay2                  ;延时 1.68ms
       call   delay2
       call   delay2
       jmp    back                    ;返回
;--------------------------------
```

```
delay1 proc                             ；延时子程序，延时 4.5ms
       mov    a,0b2h
       mov    d_count,a
d1:    sdz    d_count
       jmp    d1
       ret
delay1 endp
；--------------------------------
delay2 proc                             ；延时子程序，延时 0.56ms
       mov    a,14h
       mov    d_count,a
d2:    sdz    d_count
       jmp    d2
       ret
delay2 endp
；--------------------------------
```

2．IR 的接收：

数据的接收由外部中断来完成，通过测量两次中断的时间间隔来判断输入信号是 0 还是 1。其接收时的编码方式必须和发射时的编码一样，即：1 位启始信号+8 位数据信号。

（1）以下程序能完成 8bit 数据的接收，利用 HT47R20A-1 来接收。

```
include ht47r20a-1.inc
data .section 'data'
ACCEPT_DATA   db ?                      ；用来存放接收到的数据
time_counth   db ?                      ；存放两次外部中断的时间间隔
time_countl   db ?
bit_count     db ?                      ；接收时的位控制
begin_flag       dbit                   ；启始标志

code .section at 0 'code'
       org    00h
       jmp    start
       org    04h
       jmp    accept_int                ；外部中断，用来接收数据
       org    08h
       reti
       org    0ch
       reti
       org    10h
       reti
；--------------------------------
       org    20h
start: clr    intc0
```

```
        clr     intc1
        clr     adcr.1                  ；定时/计数器允许
        set     tmrc.5                  ；定时/计数器时钟为指令时钟
        clr     ACCEPT_DATA
        clr     begin_flag              ；清除启始标志
        mov     a,08h
        mov     bit_count,a             ；总共接收 8bit
        set     tmrc.4                  ；打开定时/计数器
        set     intc0.1                 ；外部中断允许
        set     intc0.0                 ；总中断允许
loop:   sz      bit_count               ；检测是否接收完毕
        jmp     loop
        jmp     $
；--------------------------------
accept_int:
        clr     tmrc.4                  ；关闭定时/计数器
        mov     a,tmrah                 ；读取定时/计数器的值
        mov     time_counth,a
        mov     a,tmral
        mov     time_countl,a
        clr     acc                     ；定时/计数器重新赋值
        mov     tmral,a
        mov     tmrah,a
        mov     tmrbl,a
        mov     tmrbh,a
        set     tmrc.4                  ；打开定时/计数器，重新计数
；--------------------------------
        sz      begin_flag              ；判断启始标志
        jmp     accept_0                ；有启始标志，则以下接收的是数据
        mov     a,40h                   ；否则判断接收到的是否为启始信号
        sub     a,time_countl
        mov     a,06h
        sbc     a,time_counth
        sz      c
        reti
        mov     a,68h
        sub     a,time_countl
        mov     a,06h
        sbc     a,time_counth
        sz      c
        set     begin_flag              ；是启始信号，则置位启始标志
        reti
accept_0:                               ；接收 0
```

```
        mov     a,72h                   ；判断接收到的是否为“0”信号
        sub     a,time_countl
        mov     a,0h
        sbc     a,time_counth
        sz      c
        reti
        mov     a,9ah
        sub     a,time_countl
        mov     a,0h
        sbc     a,time_counth
        snz     c
        jmp     accept_1
        clr     c
        jmp     accept
accept_1:                               ；接收 1
        mov     a,0f7h                  ；判断接收到的是否为“1”信号
        sub     a,time_countl
        mov     a,0h
        sbc     a,time_counth
        sz      c
        reti
        mov     a,1fh
        sub     a,time_countl
        mov     a,01h
        sbc     a,time_counth
        snz     c
        reti
        set     c
accept:                                 ；接收该信号
        sz      bit_count               ；判断是否满 8bit
        jmp     _accept
        reti
_accept:
        rlc     ACCEPT_DATA             ；将接收到的信号放入 ACCEPT_DATA
        dec     bit_count
        reti
```

（2）以下是利用 HT48R70A-1 来完成对以上 IR 发送器的接收的程序。

掩膜设置：clk 是 480KHZ.程序如下：

```
include ht48r70a-1.inc
data .section 'data'
ACCEPT_DATA   db ?                      ;存放接收到的字节。
time_counth   db ?                      ；保存记时器高位字节
time_countl   db ?                      ；保存记时器低位字节
```

```
    bit_count     db ?                          ；接收字节位记数器
    begin_flag    dbit                        ；接收到头信息标志位

code .section at 0 'code'
    org    00h
    jmp    start
    org    04h
    jmp    accept_int                          ；外部中断入口
    org    08h
    reti
    org    0ch
    reti
    org    10h
    reti
;*****************************
    org    20h
start:
    clr    intc                                ；关闭外部中断
    mov    a,0a0h
    mov    tmr0c,a                             ；设置记时器工作方式
    clr    ACCEPT_DATA                         ；清除各标志位
    clr    begin_flag                          ；设置读入bit个数
    mov    a,08h
    mov    bit_count,a
    set    tmr0c.4                             ；记时器开始记时
    set    intc.1                              ；开中断
    set    intc.0
loop:
    sz     bit_count                           ；看是否已收满信息
    jmp    loop                                ；否，循环等待中断
    jmp    $                                   ；程序结束
;---------------------------------
accept_int:                                    ；外部中断服务
    clr    tmr0c.4                             ；记时器停止记时
    mov    a,tmr0h                             ；读取记时器值到2个存储单元
    mov    time_counth,a
    mov    a,tmr0l
    mov    time_countl,a
    clr    acc                                 ；付初值给记时器
    mov    tmr0l,a
    mov    tmr0h,a
    mov    tmr1l,a
    mov    tmr1h,a
```

```
    mov     a,0a0h                          ；设置记时器工作方式
    mov     tmr0c,a
    set     tmr0c.4                         ；记时器开始记时
;--------------------------------
    sz      begin_flag                      ；是否已经收到头信息
    jmp     accept_0                        ；有启始标志，则以下接收的是数据
    mov     a,40h                           ；否则判断接收到的是否为启始信号

    sub     a,time_countl
    mov     a,06h
    sbc     a,time_counth
    sz      c
    reti                                    ；不是头信息，返回。
    mov     a,68h
    sub     a,time_countl
    mov     a,06h
    sbc     a,time_counth
    sz      c                               ；是否是头信息？
    set     begin_flag                      ；是头信息，设置头信息接收到标志
    reti
accept_0:                                   ；接收数据信息
    mov     a,72h
    sub     a,time_countl
    mov     a,0h
    sbc     a,time_counth
    sz      c
    reti                                    ；无效的干扰，返回
    mov     a,9ah
    sub     a,time_countl
    mov     a,0h
    sbc     a,time_counth
    snz     c                               ；接收信息是否为“0“
    jmp     accept_1                        ；跳判断接收的bit是否为“1“
    clr     c                               ；接收信息是“0“
    jmp     accept
accept_1:                                   ；判断接收信息是否为“1“
    mov     a,0f7h
    sub     a,time_countl
    mov     a,0h
    sbc     a,time_counth
    sz      c
    reti                                    ；无效的干扰，返回
    mov     a,1fh
```

```
    sub     a,time_countl
    mov     a,01h
    sbc     a,time_counth
    snz     c
    reti                                    ；无效的干扰，返回
    set     c                               ；接收的信息是“1“
accept:
    sz      bit_count
    jmp     _accept
    reti
_accept:
    rlc     ACCEPT_DATA                     ；接收信息到ACCEPT_DATA
    dec     bit_count                       ;成功接收一位信息
    reti
```

程序说明：

以上程序只是一个简单的使用介绍，其编码方式非常简单，并且只能完成 8bit 数据的发送和接收，不能直接应用到实际的系统中。其主要目的不是编码和解码的算法，而是通过这个例程来介绍 IR 发射和接收的使用方法。