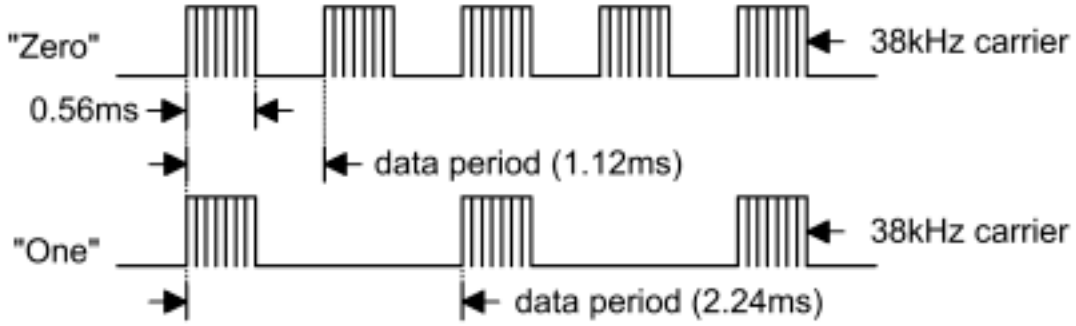
HT48RA0 发射 HT6221 码

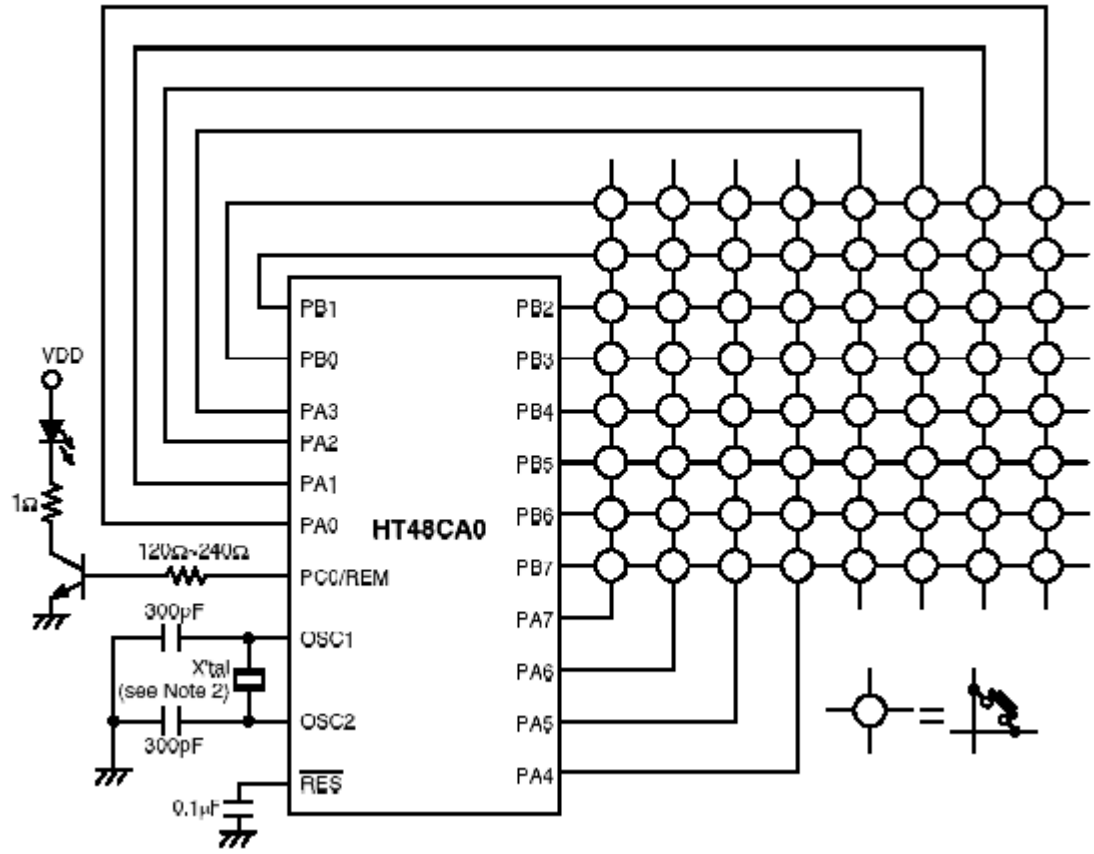
作　者：　盛扬半导体（上海）有限公司软件部
时　间：　2001/8/6
适用单片机：HT48CA0、HT48RA0

简介：

HT48RA0 是一款遥控发射单片机，可以编制各种各样的遥控码，PC.0 口作为载波输出。本文介绍如何用 HT48CA0 发射 HT6221 码。HT6221 采用 PPM（Pulse Position Modulation）进行编码，1.12ms 为 0，2.24ms 为 1，如下图：

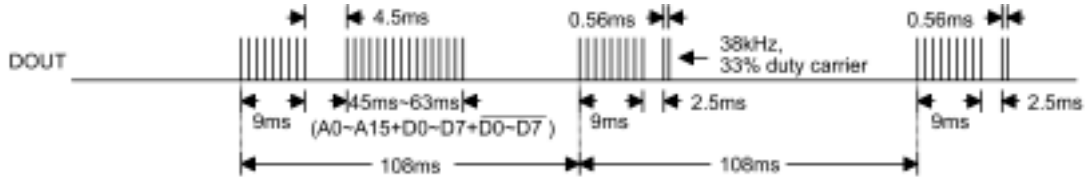


HT48CA0 应用电路图



发码过程：

这组程序的发码过程是这样的：先发射 9mS 头码，再发射 4.5mS 空码。4.5mS 空码发送完后，开始发送 16 位地址码，地址码从低位向高位发送，先发送低 8 位，低 8 位发送完后，发送高 8 位。接着，发送数据码。发射完后，发送数据反码，所有码发送完后，计算 108mS 还剩下多少，剩余的时间发射空码出去。这样完成第一个 108mS。第二个 108mS 先发送 9mS 载波，9mS 载波发送完后，发射 2.5mS 空码，再发送 0.56mS 载波，接着发送 108-9-2.5-0.56=95.94mS 空码。这样完成第二组 108mS 码。以后的发码都是重复第二组码。这就是完整的 HT48RA0 模仿 HT6221 发射的码。如下图：



例程：

```
; file name : send_code.asm
; 作者：盛扬半导体（上海）有限公司软件部
; 说明：在掩模选项中要将 PC.0 设为载波输出
#include ht48ca0.inc
address_code_l    equ    [21h]
address_code_h    equ    [22h]
carrier           equ    pc.0
delay_1           equ    [23h]
delay_2           equ    [24h]
data_8            equ    [25h]
inverse_code      equ    [26h]       ;反码暂存器
record_0          equ    [27h]       ;记录发送了多少个 0
record_1          equ    [28h]       ;记录发送了多少个 1
mem1              equ    [29h]
mem2              equ    [2ah]
mem3              equ    [2bh]
mem4              equ    [2ch]
mem5              equ    [2dh]
ram_code          equ    [2eh]        ;内存中要发送的码
datm              equ    [2fh]
m_180_1           equ    [30h]
m_180_2           equ    [31h]
m_180_3           equ    [32h]        ;记录 180ms 的剩余时间
mmx               equ    [33h]
;*****************************************************************

four    .section   'code'
   org    00h
   jmp    start
start:
```

```
    mov     a,20h
    mov     mp,a
clr_ram:                                ;清内存
    clr     [00h]
    inc     mp
    sdz     acc
    jmp     clr_ram
send_remote_code:
    mov     a,0f0h
    mov     address_code_l,a            ;要发送的地址低位码
    mov     a,0fh
    mov     address_code_h,a            ;要发送的地址高位码
    mov     a,012h
    mov     ram_code,a                  ;要发送的数据码
    cpla    ram_code
    mov     inverse_code,a              ;要发送的数据反码
;*****************************************************************

send_9_ms:
    clr     carrier                     ;开始发送 9mS 头码
    mov     a,0ch
    mov     delay_1,a
    mov     a,0afh
    mov     delay_2,a
delay_9:
    sdz     delay_2
    jmp     delay_9
    sdz     delay_1
    jmp     delay_9
    set     carrier                     ;9mS 头码结束
;*****************************************************************

    mov     a,06h                       ;发送 4.5mS 头码
    mov     delay_1,a
    mov     a,0d7h
    mov     delay_2,a
delay_4_5:
    sdz     delay_2
    jmp     delay_4_5
    sdz     delay_1
    jmp     delay_4_5                   ;4.5mS 头码结束
;*****************************************************************

    mov     a,08h
```

```
    mov     data_8,a
address_low_out:
    sz      address_code_l.0            ;发送低位地址码
    jmp     one_send                    ;从低向高发送
    call    send_0
    rr      address_code_l
    inc     record_0
    sdz     data_8
    jmp     address_low_out
    jmp     for_adhigh_out              ;继续处理高位地址码
one_send:
    call    send_1
    rr      address_code_l
    inc     record_1
    sdz     data_8
    jmp     address_low_out
    jmp     for_adhigh_out
;*****************************************************************

for_adhigh_out:
    mov     a,08h
    mov     data_8,a
address_high_out:
    sz      address_code_h.0            ;发送高位地址码
    jmp     one_send_h
    call    send_0
    rr      address_code_h
    inc     record_0
    sdz     data_8
    jmp     address_high_out
    jmp     data_out
one_send_h:
    call    send_1
    rr      address_code_h
    inc     record_1
    sdz     data_8
    jmp     address_high_out
    jmp     data_out                    ;继续处理数据码
;*****************************************************************

data_out:
    mov     a,08h
    mov     data_8,a
plus_data_out:
```

```
    sz      ram_code.0                  ;发送数据码
    jmp     data_one_send
    call    send_0
    rr      ram_code
    inc     record_0
    sdz     data_8
    jmp     plus_data_out
    jmp     inverse_code_out
data_one_send:
    call    send_1
    rr      ram_code
    inc     record_1
    sdz     data_8
    jmp     plus_data_out
    jmp     inverse_code_out            ;继续处理数据反码
;*************************************************************

inverse_code_out:
    mov     a,08h
    mov     data_8,a
inverse_data_send:                      ;发送数据反码
    sz      inverse_code.0
    jmp     inv_one
    call    send_0
    rr      inverse_code
    inc     record_0
    sdz     data_8
    jmp     inverse_data_send
    jmp     out_sky                 ;计算剩余时间
inv_one:
    call    send_1
    rr      inverse_code
    inc     record_1
    sdz     data_8
    jmp     inverse_data_send
    jmp     out_sky
;*************************************************************
out_sky:
    set     carrier                 ;计算 108mS 已经去掉多长时间
    mov     a,04h
    mov     mem2,a
    mov     a,60h
    mov     mem1,a
    mov     a,record_0              ;加上 0 码所去掉的时间
```

```
    mov     datm,a
    call    one_mul_two
    mov     a,mem3
    addm    a,m_180_1
    mov     a,mem4
    adcm    a,m_180_2
    mov     a,mem5
    adcm    a,m_180_3
    mov     a,08h
    mov     mem2,a
    mov     a,0c0h
    mov     mem1,a
    mov     a,record_1              ;加上 1 码所去掉的时间
    mov     datm,a
    call    one_mul_two
    mov     a,mem3
    addm    a,m_180_1
    mov     a,mem4
    adcm    a,m_180_2
    mov     a,mem5
    adcm    a,m_180_3
    mov     a,94h
    addm    a,m_180_1
    mov     a,11h
    adcm    a,m_180_2
    clr     acc
    adcm    a,m_180_3               ;加上 4.5mS
    mov     a,28h
    addm    a,m_180_1
    mov     a,23h
    addm    a,m_180_2
    clr     acc
    adcm    a,m_180_3               ;加上 9mS
    mov     a,0e0h
    subm    a,m_180_1
    mov     a,0a5h
    sz      c
    jmp     c_clr_1
    set     c
    jmp     two_sub                 ;剩余下的时间存在 m_180_1、2、3 中
c_clr_1:
    clr     c
two_sub:
    sbcm    a,m_180_2
```

```
    mov     a,01h
    sz      c
    jmp     c_clr_2
    set     c
c_clr_2:
    clr     c
    clr     mmx
get_mmx:
    sdz     m_180_2                 ;mmx 为 delay 时间暂存器
    jmp     $+2
    jmp     delay_no_carrier
    sdz     m_180_2
    jmp     $+2
    jmp     delay_no_carrier
    sdz     m_180_2
    jmp     $+2
    jmp     delay_no_carrier
    inc     mmx
    jmp     get_mmx
delay_no_carrier:
    sdz     m_180_1
    jmp     $-1
    sdz     mmx
    jmp     $-3
;*****************************************************************

send_9_ms_r:
    clr    carrier                  ;连续发后继码
    mov    a,0ch                    ;9mS 码
    mov    delay_1,a
    mov    a,0afh
    mov    delay_2,a
delay_9_s:
    sdz    delay_2
    jmp    delay_9_s
    sdz    delay_1
    jmp    delay_9_s
    set    carrier
send_25_ms:
    mov    a,04h                    ;2.5mS 后继码
    mov    delay_1,a
    mov    a,03eh
    mov    delay_2,a
del_25:
```

```
    sdz    delay_2
    jmp    del_25
    sdz    delay_1
    jmp    del_25
    clr    carrier

    mov    a,0bah
    mov    delay_1,a
    sdz    delay_1
    jmp    $-1
    set    carrier                  ;0.56 mS 载波

    clr    delay_1
    mov    a,7ch
    mov    delay_2,a
    sdz    delay_1
    jmp    $-1
    sdz    delay_2
    jmp    $-3                      ;完整的 108 mS
    jmp    send_9_ms_r               ;再重新发码
;*****************************************************************

send_0    proc
    clr    carrier                  ;发射 0 码子程序
    mov    a,0bah
    mov    delay_1,a
delay_5_6:
    sdz    delay_1
    jmp    delay_5_6
    set    carrier
    mov    a,0bah
    mov    delay_1,a
delay_5_6_2:
    sdz    delay_1
    jmp    delay_5_6_2
    ret
send_0    endp
;*****************************************************************
send_1    proc
    clr    carrier
    mov    a,0bah
    mov    delay_1,a
del1_5_6:
    sdz    delay_1                  ;发射 1 码子程序
```

```
    jmp    del1_5_6
    set    carrier
    mov    a,03h
    mov    delay_1,a
    mov    a,02dh
    mov    delay_2,a
delay_108:
    sdz    delay_2
    jmp    delay_108
    sdz    delay_1
    jmp    delay_108
    ret
send_1     endp
;*****************************************************************

one_mul_two    proc
    clr    mem3                    ;一个字节乘以两个字节子程序
    clr    mem4                    ; mem3,4,5 存放结果
    clr    mem5                    ; mem1,mem2 存放乘数
    mov    a,08h
    mov    data_8,a
tmul:
    sz     datm.7                  ; mul 存放被乘数
    jmp    tmu_1
    clr    c
    rlc    mem3
    rlc    mem4
    rlc    mem5
    clr    c
    rl     datm
    sdz    data_8
    jmp    tmul
    ret
tmu_1:
    clr    c
    rlc    mem3
    rlc    mem4
    rlc    mem5
    clr    c
    mov    a,mem1
    addm   a,mem3
    mov    a,mem20
    adcm   a,mem4
    sz     c
```

```
    inc    mem5
    clr    c
    rl     datm
    sdz    data_8
    jmp    tmul
    ret
one_mul_two    endp
```

注意事项:

1、在有载波输出时，只需要 clr pc.0。

2、第一组 108mS 码发送完后，在以后的发码过程中，要不断检测何时停止发码。

3、使用 HT48CA0 时，要注意它没有指令 reti，在芯片中没有外部、内部中断。

4、由于只有一级堆栈，所以不能进行子程序嵌套。

注：关于 6221 码的接收在 HA0040s.doc（利用 HT48 接收解码 HT6221 码）一文中有叙述，可参考相关文档 HA0040s.doc。