



**PRODUCT NAME**

TM58PR10

**TITLE**

TM58PR10 遥控器编码(M50560 格式)

**APPLICATION NOTE**

- 1.功能简介
- 2.DEMO 程序
- 3.应用线路图

## 功能简介：

对于遥控器产品来说，如何利用端口的设置来进行正确的发码显得尤为重要。对于新推出的遥控器 MCU (TM58PR10) 来说，以 REM 端口为载波发射端口，只需控制此端口，将发射的数据送入其中，便能完成你所需要的发码任务。

Note:

1. 在该 MCU 中，port B 及 port C 端口均有使用了下拉及唤醒功能。注意，port B 端口在任何情况下，都只能设置为输入，在 port C 端口，pc1 端口属于 I/O 口（作输入模式可外接 Led 显示），pc0 口作输入及 OFF mode port，而 pc2 与 port B 端口具有同样的功能设置。
2. 与 port B 与 port C 端口的下拉及唤醒功能寄存器 PSTAT(地址为 24H)，其具体设置定义请参考以下：

Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PSTAT		PB Wake-up	PC Wake-up	PB Pull-down	PC1/ PC0 Pull-down	PC2 Pull-down	PC1 mode	X	PC0 mode
Setting	0	OFF	OFF	OFF	OFF	Disable	$\overline{LED}$	0	OFF
	1	ON	ON	ON	ON	Enable	PC1	0	IN
After cold reset		0	0	1	0	0	0	0	0

b0: 说明 PC<sub>0</sub> 端口的输入模式。

0 = OFF (高阻抗); 1 = IN (输入模式).

b1: 保留;

b2: 说明 PC1/LED 端口的 I/O 模式。

0 = LED (输出模式); 1 = PC 1 (输入模式)

b3: 说明 PC2 下拉电阻的使用。

0 = 禁止 (无下拉); 1 = 允许 (有下拉).

b4: 说明在 PC1/ PC0 端口输入模式下拉电阻的使用。

0 = OFF (不使用); 1 = ON (使用).

b5: 说明 PB 端口下拉电阻的使用。

0 = OFF (不使用); 1 = ON (使用).

b6: 说明 PC 端口的唤醒功能。

0 = OFF; 1 = ON.

芯片可通过改变 PC 端口的状态被唤醒，PC 端口设成输入模式。

b7: 说明 PB 端口唤醒功能。

0 = OFF; 1 = ON.

芯片可以通过 PB 端口的状态被唤醒。

下面是以 TM58PR10 扫描 8\*11 的 DEMO 程序及线路图，其发码的格式为 M50560,程序的功能及设置请参考程序中的注释。

```

;-----
;ic :tm58pr10_24pin
;title :m50560. asm
; 振荡频率:4M
;key count:88key
;date :2004-11-03
;-----
;code_head:                high:8.0ms   low:4.0ms

;client_code   8bit
;               code_0   high:500us  low:500us
;               code_1   high:500us  low:1500us

;separation_bit: 1bit
;               high:500us  low:4.0ms

;data_code:    8bit
;               code_0   high:500us  low:500us
;               code_1   high:500us  low:1500us

;stop_bit_code 1bit
;               high:500us  low:21.76ms
;Note:
;               Last send repeat code from the lead_code to the stop_bit_code.

;=====
;special register
;=====
iar          equ    00h
tmr0         equ    01h
pc           equ    02h
status      equ    03h
bsr         equ    04h
porta       equ    05h
portb       equ    06h
portc       equ    07h
buf         equ    12h
modl        equ    20h
modh        equ    21h
irq         equ    22h
pgs         equ    23h
pstat       equ    24h
cg_ctrl     equ    25h
tmr_ctrl    equ    26h
osc_adj     equ    27h

;=====
;status bit

```

```

;=====
c          equ      0
dc         equ      1
z          equ      2
pd         equ      3
;-----
a          equ      00h
m          equ      01h
;=====
;general ram
;=====
verify    equ      09h
key_value equ      10h
count     equ      11h
temp1     equ      14h
temp2     equ      15h
tm_r      equ      15h
scan_temp equ      16h
temp      equ      17h
temp_buf  equ      18h
status_buf equ     19h
acc_buf   equ      1ah
; ;-----
;Program start here.
;-----
        org      3feh      ;中断程序入口地址
        lgoto    tmr0_int  ;进入中断子程序
        org      3ffh
        movla   7fh        ;程序在 ROM 中的起始地址
        org      000h
        movam   27h
        lgoto   main      ;进入主程序
;-----

;-----延时 10ms 子程序-----
delay_10ms:          ;延时 10ms
        movla   .10
        movam   temp1
loop:
        movla   0fah
        movam   temp2

        nop
        nop
        decmsz  temp2,m
        lgoto   $-3
        decmsz  temp1,m
        lgoto   loop

```

```

        retla        00h
;-----
s_sleep:
        movla       0ffh        ;portb and portc 唤醒
        movam      porta       ;porta 输出高电平，使得有键按下后使 portb
                                ;portc 端口电平变化

        nop
        nop
        sleep      ;没有按键按下，进入睡眠状态
        nop        ;sleep 唤醒后回到 3FFH
        nop

;-----主程序-----
main:
        clra              ;porta 口设置为输出，为低电平
        iodir          porta
        movam         porta

        clrm          portb   ;portb 口设置为输入状态
        clrm          portc   ;portc 口设置为输入状态

        movla       0fdh
        movam      pstat      ;对 portb/portc 的唤醒与下拉功能进行设置

        movla       46h       ;clock source=4MHz,1/3duty,output=38kHz
        movam      modl      ;modl=2*4M*(1-1/3)*(1/38k)=46h(8.75us)
        movla       8ch
        movam      modh      ;modh=2*4M*(1-2/3)*(1/38k)=8ch(17.50us)

;-----
start:
        clrm         key_value ;清行计数值和按键值
        clrm         count

start1:
        lcall        key_scan  ;调用 key_scan
        movm         key_value,a
        movam        verify    ;送键值到 verify
        movla       0ffh
        xoram        key_value,a ;判断是否有按键按下
        btmsc        status,z
        lgoto       s_sleep    ;没有按键按下，进入 s_sleep
        lcall        delay_10ms ;调用延时抖消
        lcall        key_scan   ;再次进行按键扫描
        movm         key_value,a
        xoram        verify,a  ;与开始扫描的键值进行校验
        btmsc        status,z

```

```

lgoto    s_sleep      ;不等则误判，进入 s_sleep

movm     key_value,a  ;相同则将键值送入 temp_buf 存储
movam    temp_buf
lcall    send_lead_code;调用发码子程序

```

;;-----按键扫描子程序-----

```

key_scan:                                ;按键扫描处理
    movla    0ffh
    movam    key_value
    movla    .8
    movam    count      ;按键扫描 8 次
    movla    b'10000000'
    movam    scan_temp
    movam    porta
    lgoto    $+5
next_column:
    bcm      status,0
    rrm      scan_temp,1
    movm     scan_temp,a
    movam    porta
    lgoto    $+1
    lgoto    $+1
    lgoto    $+1
    lgoto    $+1
    lgoto    $+1
    lgoto    $+1

    btmsc    portb,0
    lgoto    row_0      ;有键按下，调用表格
    btmsc    portb,1    ;没有按键按下，继续扫描
    lgoto    row_1
    btmsc    portb,2
    lgoto    row_2
    btmsc    portb,3
    lgoto    row_3
    btmsc    portb,4
    lgoto    row_4
    btmsc    portb,5
    lgoto    row_5
    btmsc    portb,6
    lgoto    row_6
    btmsc    portb,7
    lgoto    row_7

    btmsc    portc,0
    lgoto    row_8
    btmsc    portc,1

```

```

lgoto    row_9
btmsc   portc,2
lgoto    row_10

decmsz   count,1
lgoto    next_column
ret      ;key_scan 结束返回
;-----
send_lead_code:
movla    b'00011010' ;tmr0 采用内部时钟计数,预分频器分配给 tmr0
movam    tmr_ctrl    ;分频比为 1:4
movla    .155        ;给 tmr0 重装值
movam    tmr0
movla    B'10000000' ;开启中断功能
movam    irq
clrm     tm_r
bsm      cg_ctrl,3   ;启动载波
movla    .14
xoram    tm_r,a      ;定时发码为 9.00ms
btms     status,z
lgoto    $-3
bcm      cg_ctrl,3   ;停止载波发送
bcm      cg_ctrl,2
movla    .185        ;4.00ms
movam    tmr0
clrm     tm_r
movla    .7
xoram    tm_r,a
btms     status,z
lgoto    $-3

bcm      cg_ctrl,2
clrm     tm_r      ;
clrm     irq      ;清中断子程序

;-----;send_client_code(010100)
send_client_code_loop_st:
;发送客户码(010100)
lcall    code_0     ;调用 code_0 子程序
lcall    code_1     ;调用 code_1 子程序
lcall    code_0     ; code_0
lcall    code_1     ; code_1
lcall    code_0     ; code_0
lcall    code_0     ; code_0
lcall    code_1     ; code_1
lcall    code_0     ; code_0
;-----

```

send\_separation\_code:

```

    movla    b'00011010' ;tmr0 采用内部时钟计数,预分频器分配给 tmr0
    movam    tmr_ctrl    ;分频比为 1:4
    movla    .115        ;给 tmr0 重装值
    movam    tmr0
    movla    B'10000000' ;开启中断使能
    clrm     tm_r
    movam    irq
    bsm      cg_ctrl,3   ;启动载波
    movla    .1
    xoram    tm_r,a      ;定时发送载波时间为 500us
    btms     status,z
    lgoto    $-3
    bcm      cg_ctrl,3   ;时间到，停止载波发送
    bcm      cg_ctrl,2
    movla    .185
    movam    tmr0
    clrm     tm_r
    movla    .7          ;4.00ms
    xoram    tm_r,a
    btms     status,z
    lgoto    $-3

    bcm      cg_ctrl,2
    clrm     irq
    clrm     tm_r

```

-----

send\_key\_code

```

    movla    .8          ;发送键码(8bit)
    movam    count

```

send\_code\_loop:

```

    rrm      temp_buf,m  ;键值循环左移
    btmsc    status,c    ;判断发码为 1 还是为 0
    lgoto    send_code_1

```

send\_code\_0:

```

    lcall    code_0      ;为 0 则发 send_0
    lgoto    send_code_end

```

send\_code\_1:

```

    lcall    code_1      ;为 1 则发 send_1

```

send\_code\_end:

```

    decmsz   count,m     ;判断发码是否结束
    lgoto    send_code_loop

```

```

    movla    b'00011010' ;tmr0 采用内部时钟计数,预分频器分配给 tmr0

```

```

movam    tmr_ctrl    ;分频比为 1:4
movla    .115        ;给 tmr0 重装值
movam    tmr0
movla    B'10000000' ;开启中断使能
movam    irq
clrm     tm_r
bsm      cg_ctrl,3   ;启动载波
movla    .1
xoram    tm_r,a      ;定时发送载波时间为 500US
btmss    status,z
lgoto    $-3

bcm      cg_ctrl,3
;-----;读取时间，时间到发连续码
read_wait:
btmsc    portb,0     ;判断按键是否抬起,没有则等待，读取时间
lgoto    read_wait_nt
btmsc    portb,1
lgoto    read_wait_nt
btmsc    portb,2
lgoto    read_wait_nt
btmsc    portb,3
lgoto    read_wait_nt
btmsc    portb,4
lgoto    read_wait_nt
btmsc    portb,5
lgoto    read_wait_nt
btmsc    portb,6
lgoto    read_wait_nt
btmsc    portb,7
lgoto    read_wait_nt

btmsc    portc,0
lgoto    read_wait_nt
btmsc    portc,1
lgoto    read_wait_nt
btmsc    portc,2
lgoto    read_wait_nt
lgoto    dsp_key_end ;按键抬起，发结束码

read_wait_nt:
movm     tm_r,a      ;读取时间约为 21.76MS
xorla    .40
btmss    status,z
lgoto    read_wait

clrm     tm_r

```

```

movla    b'00011010'
movam    tmr_ctrl
clrm     irq
movm     buf,a
movam    temp_buf

lgoto    send_lead_code;发送重复码

```

```

dsp_key_end:                ;发码结束
clrm     tm_r
movla    b'00011010'
movam    tmr_ctrl
clrm     irq                ;清中断标志
lgoto    s_sleep           ;进入 s_sleep 状态

```

-----码发送子程序-----

```

code_0:
movla    b'00011010'      ;tmr0 采用内部时钟计数,预分频器分配给 tmr0
movam    tmr_ctrl         ;分频比为 1:4
movla    .115
movam    tmr0             ;给 tmr0 重装值
movla    b'10000000'      ;开放 tmr0 中断功能
movam    irq
clrm     tm_r
bsm     cg_ctrl,3         ;启动载波
movla    .1               ;定时发送载波时间 500us
xoram    tm_r,a
btmss   status,z
lgoto    $-3
bcm     cg_ctrl,3         ;停止载波发送
bcm     cg_ctrl,2
movla    .118
movam    tmr0
clrm     tm_r
movla    .1               ;500us
xoram    tm_r,a
btmss   status,z
lgoto    $-3

clrm     tm_r
clrm     irq              ;清中断标志
ret

```

-----

```

code_1:
movla    b'00011010'      ;tmr0 采用内部时钟计数,预分频器分配给 tmr0
movam    tmr_ctrl         ;分频比为 1:4

```

```

movla      .120      ;给 tmr0 重装值
movam      tmr0      ;开放 tmr0 中断功能
movla      b'10000000'
movam      irq
clrm      tm_r
bsm        cg_ctrl,3 ;启动载波

movla      .1
xoram      tm_r,a    ;定时发送载波时间 500us
btmss      status,z
lgoto     $-3
bcm        cg_ctrl,3 ;停止载波发送
bcm        cg_ctrl,2
movla      .110
movam      tmr0
clrm      tm_r
movla      .3
xoram      tm_r,a    ;1.50ms
btmss      status,z
lgoto     $-3

clrm      tm_r
clrm      irq        ;清中断标志
ret

```

-----tmr0 interrupt-----

```

tmr0_int
movam      acc_buf   ;将累加器内容送入 a_temp 中进行保护
swapm      status,a  ;将状态寄存器 status 高低半字节内容
                    ;相互交换结果送累加器中
movam      status_buf ;最后将结果送入 status_temp 中进行保护

incm      tm_r,m
movla      .138      ;给 tmr0 重装值
movam      tmr0
tmr0_int_end
bcm        irq,0     ;清 tmr0 中断请求标志
swapm      status_buf,a ;将 status_temp 寄存器高低半字节内容相
                    ;互交换结果送累加器中

movam      status    ;累加器内容送状态寄存器 status,以恢复
                    ;进中断前的 status 状态

movm      acc_buf,a
reti

```

```

; ;-----
org      200h

```

```
-----  
row_0:  
    movm    count,a  
    lcall   row_0_tab  
    movam   key_value  
    retla  0  
row_1:  
    movm    count,a  
    lcall   row_1_tab  
    movam   key_value  
    retla  0  
row_2:  
    movm    count,a  
    lcall   row_2_tab  
    movam   key_value  
    retla  0  
row_3:  
    movm    count,a  
    lcall   row_3_tab  
    movam   key_value  
    retla  0  
row_4:  
    movm    count,a  
    lcall   row_4_tab  
    movam   key_value  
    retla  0  
row_5:  
    movm    count,a  
    lcall   row_5_tab  
    movam   key_value  
    retla  0  
row_6:  
    movm    count,a  
    lcall   row_6_tab  
    movam   key_value  
    retla  0  
row_7:  
    movm    count,a  
    lcall   row_7_tab  
    movam   key_value  
    retla  0  
row_8:  
    movm    count,a  
    lcall   row_8_tab  
    movam   key_value  
    retla  0  
row_9:
```

```
    movm    count,a
    lcall   row_9_tab
    movam   key_value
    retla   0
row_10:
    movm    count,a
    lcall   row_10_tab
    movam   key_value
    retla   0

;-----
row_0_tab:
    addam   pc,    1
    nop
    retla   00h    ;/sw01
    retla   01h    ;/sw02
    retla   02h    ;/sw03
    retla   03h    ;/sw04
    retla   04h    ;/sw05
    retla   05h    ;/sw06
    retla   06h    ;/sw07
    retla   07h    ;/sw08

row_1_tab:
    addam   pc,    1
    nop
    retla   08h    ;/sw09
    retla   09h    ;/sw10
    retla   0ah    ;/sw11
    retla   0bh    ;/sw12
    retla   0ch    ;/sw13
    retla   0dh    ;/sw14
    retla   0eh    ;/sw15
    retla   0fh    ;/sw16

row_2_tab:
    addam   pc,    1
    nop
    retla   10h    ;/sw17
    retla   11h    ;/sw18
    retla   12h    ;/sw19
    retla   13h    ;/sw20
    retla   14h    ;/sw21
    retla   15h    ;/sw22
    retla   16h    ;/sw23
    retla   17h    ;/sw24

row_3_tab:
    addam   pc,    1
```

---

```

nop
retla      18h      ;/sw25
retla      19h      ;/sw26
retla      1ah      ;/sw27
retla      1bh      ;/sw28
retla      1ch      ;/sw29
retla      1dh      ;/sw30
retla      1eh      ;/sw31
retla      1fh      ;/sw32

row_4_tab:
  addam     pc,      1
  nop
  retla     20h      ;/sw33
  retla     21h      ;/sw34
  retla     22h      ;/sw35
  retla     23h      ;/sw36
  retla     24h      ;/sw37
  retla     25h      ;/sw38
  retla     26h      ;/sw39
  retla     27h      ;/sw40

row_5_tab:
  addam     pc,      1
  nop
  retla     28h      ;/sw41
  retla     29h      ;/sw42
  retla     2ah      ;/sw43
  retla     2bh      ;/sw44
  retla     2ch      ;/sw45
  retla     2dh      ;/sw46
  retla     2eh      ;/sw47
  retla     2fh      ;/sw48

row_6_tab:
  addam     pc,      1
  nop
  retla     30h      ;/sw49
  retla     31h      ;/sw50
  retla     32h      ;/sw51
  retla     33h      ;/sw52
  retla     34h      ;/sw53
  retla     35h      ;/sw54
  retla     36h      ;/sw55
  retla     37h      ;/sw56

row_7_tab:
  addam     pc,      1
  nop
```

---

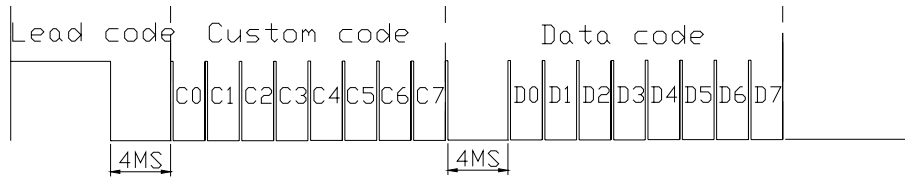
```
retla      38h      ;/sw57
retla      39h      ;/sw58
retla      3ah      ;/sw59
retla      3bh      ;/sw60
retla      3ch      ;/sw61
retla      3dh      ;/sw62
retla      3eh      ;/sw63
retla      3fh      ;/sw64
row_8_tab:
  addam    pc,      1
  nop
  retla    40h      ;/sw65
  retla    41h      ;/sw66
  retla    42h      ;/sw67
  retla    43h      ;/sw68
  retla    44h      ;/sw69
  retla    45h      ;/sw70
  retla    46h      ;/sw71
  retla    47h      ;/sw72

row_9_tab:
  addam    pc,      1
  nop
  retla    48h      ;/sw73
  retla    49h      ;/sw74
  retla    4ah      ;/sw75
  retla    4bh      ;/sw76
  retla    4ch      ;/sw77
  retla    4dh      ;/sw78
  retla    4eh      ;/sw79
  retla    4fh      ;/sw80

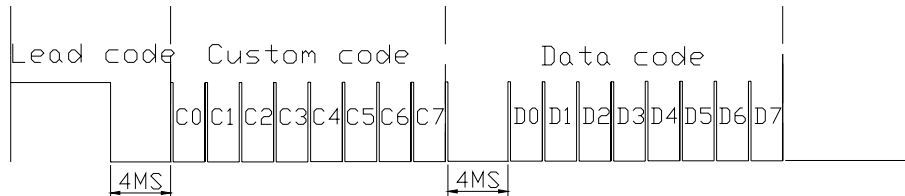
row_10_tab:
  addam    pc,      1
  nop
  retla    51h      ;/sw81
  retla    52h      ;/sw82
  retla    53h      ;/sw83
  retla    54h      ;/sw84
  retla    55h      ;/sw85
  retla    56h      ;/sw86
  retla    57h      ;/sw87
  retla    58h      ;/sw88
;-----
end
```

Note: M50560 发码的波形如下所示：

A single pulse, modulated with 4MHz.

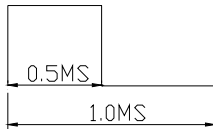


REPEAT CODE

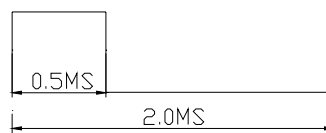


BIT DESCRIPTION

BIT "0" (ms)



BIT "1" (ms)



应用线路图如下：

