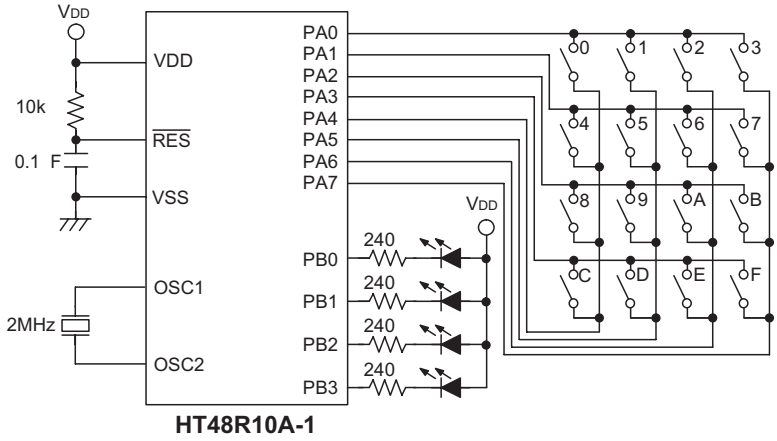
HT48 & HT46 键盘扫描程序

文件编码：HA0011s

简介：

这是一个 4×4 的键盘阵列，总共有 16 个键，如图所示每个键都有一个 16 进制的代码。键盘扫描程序扫描键盘阵列确认是哪一个键被按下了，确定键后用 LED 显示它的 2 进制代码。如图，这有 4 个 LED，表示的值是从 0000H 到 1111H。 在扫描过程中，如果同时有两个键被按下的话，那么只有第一个被扫描到的键会被检测到并显示。使用这种方法的编码键盘可以把每一个值指定给键



电路设计：

PA0~PA3 设置为输出口，PA4~PA7 设置为输入口，这样组成了一个 4×4 的阵列，程序扫描哪一个键被按下，查表确定它的值。PB0~PB3 定义为输出口，它输出 4 位二进制码，16 个值每个值对应一个键。

扫描过程:

以第一行第一列为例。先向 PA 口输出 0FEH，既扫描第一行。如果第一行有键按下，则读入的 PA 口键值的高位不会为 F；如果 PA 口高位不为 F，则第一行有键按下，转入列扫描。如果第一列有键按下，则 PA.4 为 0，否则为 1。这样可以确定按下键的行和列，确定其编码。

程序清单：

```
#include ht48r10a-1.inc
; ------------------------------------------------------------data
data .section 'datat'              ; 数据段
    temp       db  ?               ; 暂时数据寄存器
    temp2      db  ?               ;用于保存键盘扫描码以检测列值
    disp       db  ?               ; 键值显示寄存器
    count1     db  ?               ; 延时计数指针
    mask       db  ?               ; 屏蔽寄存器
    matrix     db  ?               ; 键盘阵列寄存器r
; ------------------------------------------------------------code
```

```
code .section at 0 'code'          ; 程序段
    org    00h
    jmp    start
start:                             ; 程序开始
    clr    pbc                     ; 设置PB口为输出口
    mov    a,0f0h        ; (1)     ; 设置PA高4位为输入口、低四位为输出口
    mov    pac,a
    clr    pa                      ; 清PA 口
    clr    pb                      ; 清PB 口
keyloop:                           ; 键扫描循环
    mov    a,0feh        ; (2)     ; 扫描第一行是否被按下
    mov    matrix,a                ; 将第一行的代码送matrix
    mov    pa,a                    ; 输出扫描码到PA 口
    mov    a,pa                    ; 读入PA口 的状态到ACC
    xor    a,0feh                  ; 判断高4位有无0?如有,则有键按下,ACC值应改变
    sz     acc                     ; 第一行是否有键按下:如果有键按下,则ACC不为0
    jmp    get_key                 ; 有键按下跳到读键值
    mov    a,0fdh        ; (2)     ; 扫描第二行是否被按下
    mov    matrix,a                ; 将第二行的代码送matrix
    mov    pa,a
    mov    a,pa
    xor    a,0fdh
    sz     acc
    jmp    get_key
    mov    a,0fbh        ; (2)     ; 扫描第三行是否被按下
    mov    matrix,a                ; 将第三行的代码送matrix
    mov    pa,a
    mov    a,pa
    xor    a,0fbh
    sz     acc
    jmp    get_key
    mov    a,0f7h        ; (2)     ; 扫描第四行是否被按下
    mov    matrix,a                ; 将第四行的代码送matrix
    mov    pa,a
    mov    a,pa
    xor    a,0f7h
    sz     acc
    jmp    get_key
    jmp    keyloop                 ; 跳回键循环扫描
get_key:                           ; 取键值
    call   key_in        ; (3)     ; 调用key_in 子程序
    mov    pb,a          ; (11)    ; 从PB口显示键值
    jmp    keyloop                 ; 跳回键循环扫描
```

```
key_in proc                          ; 键值读入子程序
    mov    a,pa                      ;读取PA口数据
    mov    temp,a        ; (4)       ; 将PA口的状态读入的 temp寄存器中
    mov    temp2,a                   ; 扫描值送入temp2用于检测列值
    call   delays        ; (5)       ; 调用延时子程序
get_release:                         ; 等待键松开
    mov    a,pa                      ; 将PA口的主状态值送ACC
    and    a,0f0h                    ; 屏蔽ACC高四位，取按键状态
    xor    a,0f0h
    sz     acc           ; (6)       ; 等键松开 , 键如松开则acc=0
    jmp    get_release
    mov    a,0fh                     ; 取屏蔽寄存器的低四位
    andm   a,matrix
    mov    a,0
get_row:                             ; 取行数
    rrc    matrix        ; (7)       ; 右移 matrix 指针
    sz     status.0                  ; 检查并取键行
    jmp    get_column1               ; 如果找到键行，跳到 get_next
    clr    c                         ; 如果还未找到键行，清carry_c
    add    a,4h          ; (8)       ; 加4到显示指针
    jmp    get_row                   ; 跳回get_row
get_colmn1:
    mov    temp,a
    mov    a,0f0h
    andm   a,temp2                   ;取键盘扫描码的高4位,检测列值
    swap   temp2                     ;交换,把列值放到低4位上
    mov    a,0h          ; (9)
get_column:
    rrc    temp2                     ;逐位检测,到该位为0为止
    snz    status.0                  ;为0,则说明该列有键按下
    jmp    next
    clr    c
    add    a,1h
    jmp    get_column                ; 取出列值
next:
    add    a,temp
    xor    a,0ffh                    ; 计算键值并求出显示码
key_in endp
delays proc                          ; 延时子程序
    mov    a,0ffh
    mov    count1,a
d1:
    sdz    count1
```

```
    jmp     d1
    ret
delays endp
```

程序说明：

（1）段定义了哪些口是输入口,哪些口是输出口。程序循环扫描是否有键按下然后进行处理，这段代码首先从一行一行的扫描这一行是否有键按下，因为有四个键在同一行上，程序必须要确定这一行是哪一个键被按下。例如：在（2）段中，第一行中 1-3 键被按下，程序跳出循环扫描，跳到（3）段代码去确认这一行是那一个键被按下。如果第一行没有键按下，它会去检测 4-7 键，以此类推。当程序进入到这段代码，它会将按键状态存在暂时寄存器中，如（4）所示。然后是一段时间延时（5），考虑到键会松开，代码检测键是否松开（6），程序一直循环直到键完全松开。下一段程序确定那一行的键被按了（7），由 4 位二进制码控制程序一行一行的扫描（8）。键所在的行就确定了，行乘 4 的值加上键所在的列这样键的值就确定了（9）。真正求解到键值是先求行值 ，然后求列值。

在（1）中的查表指令是根据键值求出显示代码，然后送到 PB 由 LED 显示已按的键。