触摸按键扫描范例程序

```
;*********************************************
;触摸按键扫描范例程序
;*********************************************
;程序出口：
;KEYF 对应 8 个触摸按键
;KEYF,0-KEY1,KEYF~KEY2.....KEYF,7~KEY8
;KEYF,0=1 表示 KEY1 按下。
;KEYF,1=1 表示 KEY2 按下。
;KEYF,2=1 表示 KEY3 按下。
;KEYF,3=1 表示 KEY4 按下。
;KEYF,4=1 表示 KEY5 按下。
;KEYF,5=1 表示 KEY6 按下。
;KEYF,6=1 表示 KEY7 按下。
;KEYF,7=1 表示 KEY8 按下。

;B_KEY_EN=1      按键单次按下标志
;B_KEY_LONG=1 按键长按标志
;B_KEY_UP=0      按键放开标志
;*********************************************
;#INCLUDE      CMS69F6162.H
;*********************************************
KEYF            EQU         08H         ;按键标志位,按键输出接口必须用
BANK1,BANK0 公用寄存器,该地址不能更改
KFLAG       EQU         0EH         ;按键标志位,按键输出接口必须用
BANK1,BANK0 公用寄存器,该地址不能更改
B_KEY_EN    EQU         KFLAG,0
```

```
B_KEY_LONG EQU     KFLAG,1
B_KEY_UP   EQU     KFLAG,2
BANK1      DATA0L  EQU     ?
BANK1      DATA0H  EQU     ?
BANK1      DATA1L  EQU     ?          ;触摸按键 K1 求和保存低位
BANK1      DATA1H  EQU     ?          ;触摸按键 K1 求和保存高位
BANK1      DATA2L  EQU     ?          ;触摸按键 K2 求和保存低位
BANK1      DATA2H  EQU     ?          ;触摸按键 K2 求和保存高位
BANK1      DATA3L  EQU     ?          ;触摸按键 K3 求和保存低位
BANK1      DATA3H  EQU     ?          ;触摸按键 K3 求和保存高位
BANK1      DATA4L  EQU     ?          ;触摸按键 K4 求和保存低位
BANK1      DATA4H  EQU     ?          ;触摸按键 K4 求和保存高位
BANK1      DATA5L  EQU     ?          ;触摸按键 K5 求和保存低位
BANK1      DATA5H  EQU     ?          ;触摸按键 K5 求和保存高位
BANK1      DATA6L  EQU     ?          ;触摸按键 K6 求和保存低位
BANK1      DATA6H  EQU     ?          ;触摸按键 K6 求和保存高位
BANK1      DATA7L  EQU     ?          ;触摸按键 K7 求和保存低位
BANK1      DATA7H  EQU     ?          ;触摸按键 K7 求和保存高位


BANK1      KEYC    EQU     ?          ;按键有效次数
BANK1      KOLD    EQU     ?          ;旧的键值
BANK1      KDATA   EQU     ?          ;有按键后保存的按键值
BANK1      KSUB    EQU     ?

BANK1      KODAT0L     EQU     ?          ;按键旧值（必须顺序排列）
BANK1      KODAT1L     EQU     ?
BANK1      KODAT01H    EQU     ?
```

```
BANK1       K0DAT2L      EQU        ?
BANK1       K0DAT3L      EQU        ?
BANK1       K0DAT23H EQU        ?

BANK1       K1DAT0L      EQU        ?          ;按键旧值（必须顺序排列）
BANK1       K1DAT1L      EQU        ?
BANK1       K1DAT01H EQU        ?
BANK1       K1DAT2L      EQU        ?
BANK1       K1DAT3L      EQU        ?
BANK1       K1DAT23H EQU        ?

BANK1       K2DAT0L      EQU        ?          ;按键旧值（必须顺序排列）
BANK1       K2DAT1L      EQU        ?
BANK1       K2DAT01H EQU        ?
BANK1       K2DAT2L      EQU        ?
BANK1       K2DAT3L      EQU        ?
BANK1       K2DAT23H EQU        ?

BANK1       K3DAT0L      EQU        ?          ;按键旧值（必须顺序排列）
BANK1       K3DAT1L      EQU        ?
BANK1       K3DAT01H EQU        ?
BANK1       K3DAT2L      EQU        ?
BANK1       K3DAT3L      EQU        ?
BANK1       K3DAT23H EQU        ?

BANK1       K4DAT0L      EQU        ?          ;按键旧值（必须顺序排列）
BANK1       K4DAT1L      EQU        ?
BANK1       K4DAT01H EQU        ?
BANK1       K4DAT2L      EQU        ?
```

```
BANK1       K4DAT3L      EQU        ?
BANK1       K4DAT23H EQU        ?

BANK1       K5DAT0L      EQU        ?          ;按键旧值（必须顺序排列）
BANK1       K5DAT1L      EQU        ?
BANK1       K5DAT01H EQU        ?
BANK1       K5DAT2L      EQU        ?
BANK1       K5DAT3L      EQU        ?
BANK1       K5DAT23H EQU        ?

BANK1       K6DAT0L      EQU        ?          ;按键旧值（必须顺序排列）
BANK1       K6DAT1L      EQU        ?
BANK1       K6DAT01H EQU        ?
BANK1       K6DAT2L      EQU        ?
BANK1       K6DAT3L      EQU        ?
BANK1       K6DAT23H EQU        ?

BANK1       K7DAT0L      EQU        ?          ;按键旧值（必须顺序排列）
BANK1       K7DAT1L      EQU        ?
BANK1       K7DAT01H EQU        ?
BANK1       K7DAT2L      EQU        ?
BANK1       K7DAT3L      EQU        ?
BANK1       K7DAT23H EQU        ?

BANK1       ADTIME       EQU        ?
BANK1       KSTOPL       EQU        ?

BANK1       KCOUNT       EQU        ?
BANK1       K_TEMPL   EQU        ?
```

```
BANK1       K_TEMPH  EQU          ?
BANK1       K_TEMP       EQU          ?
BANK1       CD_COUNT EQU          ?
BANK1       KLONGC       EQU          ?
DATAL       EQU          DATAOL      ;计算时将 DATAOL、DATAOH 放入 DATAL 跟
DATAH 中，避免经常操作 MP
DATAH       EQU          DATAOH      ;Key0 第一个处理，可以当做临时存放其它
用

;*********************************************************
BANKSEL_BANK1 MACRO         ;设置 BANK1
        SETB      FLAGS,4
ENDM

BANKSEL_BANK0 MACRO         ;设置 BANK0
        CLRB      FLAGS,4
ENDM

C_KCOUNT    EQU         .8          ;按键个数
C_KSAVE         EQU         .2          ;保存几个按键旧值(按键旧值必须顺序
排列)
C_KEYC      EQU         .4          ;连续检测到多少次认为有按键
C_FRIST_TIME EQU          .80         ;按键初次长按确认时间
C_LONG_TIME     EQU         .5          ;
;*********************************************************
;按键通道定义表
C_P00_CH      EQU           0028H           ;P0,0
C_P01_CH      EQU           0029H           ;P0,1
C_P02_CH      EQU           002AH           ;P0,2
```

```
C_P03_CH      EQU         002BH          ;P0,3
C_P04_CH      EQU         002CH          ;P0,4
C_P05_CH      EQU         002DH          ;P0,5
C_P06_CH      EQU         002EH          ;P0,6
C_P07_CH      EQU         002FH          ;P0,7
C_P10_CH      EQU         0128H          ;P1,0
C_P11_CH      EQU         0129H          ;P1,1
C_P12_CH      EQU         012AH          ;P1,2
C_P13_CH      EQU         012BH          ;P1,3
C_P14_CH      EQU         012CH          ;P1,4
C_P16_CH      EQU         012DH          ;P1,6
C_P20_CH      EQU         012EH          ;P2,0
C_P21_CH      EQU         012FH          ;P2,1
;-------------------------------------------------------------
;选择触摸通道；只需要将上面定义的对应 IO 通道定义在下面 8 个按键里
C_KEY1_CH     EQU         C_P00_CH
C_KEY2_CH     EQU         C_P01_CH
C_KEY3_CH     EQU         C_P02_CH
C_KEY4_CH     EQU         C_P03_CH
C_KEY5_CH     EQU         C_P04_CH
C_KEY6_CH     EQU         C_P05_CH
C_KEY7_CH     EQU         C_P06_CH
C_KEY8_CH     EQU         C_P07_CH
;*************************************************************
; 触摸通道口电容选择
C_KCAP_0      EQU         B'00000010'  ;按键口不接电容
C_KCAP_1      EQU         B'00001010'  ;按键口接电容 c*1
C_KCAP_2      EQU         B'00010010'  ;按键口接电容 c*2
C_KCAP_3      EQU         B'00011010'  ;按键口接电容 c*3
```

```
C_KCAP_4      EQU       B'00100010' ;按键口接电容 c*4
C_KCAP_5      EQU       B'00101010' ;按键口接电容 c*5
C_KCAP_6      EQU       B'00110010' ;按键口接电容 c*6
C_KCAP_7      EQU       B'00111010' ;按键口接电容 c*7

C_KEY1_ADJ    EQU       C_KCAP_0
C_KEY2_ADJ    EQU       C_KCAP_0
C_KEY3_ADJ    EQU       C_KCAP_0
C_KEY4_ADJ    EQU       C_KCAP_0
C_KEY5_ADJ    EQU       C_KCAP_0
C_KEY6_ADJ    EQU       C_KCAP_0
C_KEY7_ADJ    EQU       C_KCAP_0
C_KEY8_ADJ    EQU       C_KCAP_0
;************************************************************
;按键灵敏度设置常量
C_KEY1_LMD    EQU        .6       ;K1 灵敏度
C_KEY2_LMD    EQU        .6       ;K2 灵敏度
C_KEY3_LMD    EQU        .6       ;K3 灵敏度
C_KEY4_LMD    EQU        .6       ;K4 灵敏度
C_KEY5_LMD    EQU        .6       ;K5 灵敏度
C_KEY6_LMD    EQU        .6       ;K6 灵敏度
C_KEY7_LMD    EQU        .6       ;K7 灵敏度
C_KEY8_LMD    EQU        .6       ;K8 灵敏度

;*****************************************

;---------------------------------------
;按键检测子程序，重键模式
;1.25ms 进一次
```

```
;每进一次检测一遍所有按键
;按键出口：KEYF

;-----------------------------------------
_KSCAN:
        BANKSEL_BANK1
        CLR             KCOUNT
        CLR         KDATA
__KSCAN_LOOP:
        CLR             KEY_C
        CLR             KEY_C1
        CLR           KSTOPL
        LDIA            B'10010010'             ;放电
        LD          P1CH,A
        CLRB        P1,5
        NOP
        LDIA            B'10001010'             ;设置 P1.5 为触摸用电容口
        LD          P1CH,A
        LDIA            TABLE_KEY_ADJ$H         ; 将 地 址 的 高 位 赋 给
"TABLE_SPH"
        LD           TABLE_SPH,A
        LDIA            TABLE_KEY_ADJ$L         ; 将 地 址 的 低 位 赋 给
"TABLE_SPL"
        ADDA          KCOUNT
        LD          TABLE_SPL,A
        SZB           FLAGS,C
        INCR          TABLE_SPH
        TABLEA
        ORIA          002H
```

```
        LD          KEY_C1,A


         LDIA          TABLE_KEY_CH$H        ; 将 地 址 的 高 位 赋 给
"TABLE_SPH"
        LD          TABLE_SPH,A
        LDIA          TABLE_KEY_CH$L        ; 将 地 址 的 低 位 赋 给
"TABLE_SPL"
        ADDA          KCOUNT
        LD          TABLE_SPL,A
        SZB           FLAGS,C
        INCR          TABLE_SPH
        TABLEA
        LD          KEY_C,A                 ;开启触摸检测
        SZB           TABLE_DATAH,0
        SETB          KEY_C1,6
        NOP
        CLR         ADTIME
        SETB            KEY_C,7
__KSCAN_WAIT:
        SZB         KEY_C1,7                ;判断触摸完成标志
        JP              ___KSCAN_WAIT_OK
        SZDECR      ADTIME
        JP          __KSCAN_WAIT
        JP            ___KSCAN_WAIT_ERR
___KSCAN_WAIT_OK:
         CALL         GET_SUM_ADD
```

```
        LD          A,KEY_DATAL
        ADDR        IAR
        INCR            MP                  ;MP=DATA0H
        SZB             FLAGS,C
        INCR            IAR
        LD          A,KEY_DATAH
        ADDR        IAR

        INCR            KCOUNT              ;每个按键加 1 次
        LDIA            C_KCOUNT            ;按键个数
        SUBA        KCOUNT
        SNZB        FLAGS,C
        JP              __KSCAN_LOOP
        CLR             KCOUNT


        INCR            CD_COUNT            ;是否已经 16 次求和
        LDIA            .8
        SUBA        CD_COUNT
        SNZB        FLAGS,C
        JP              _KEY_BACK
        CLR             CD_COUNT
        JP            __KSCAN_JUGE
;*********************************************
;*********************************************
___KSCAN_WAIT_ERR:    ;一次检测不到触摸完成直接清初值
         CALL         _CLR_KEY_SUM_RAM
         CLR          KEYF
         CALL         _CLR_NO_KEY_DAT
```

```
        JP          _KEY_BACK
;-----------------------------------------
;上面部分按键检测完成
;下面进行按键键值判断
;将 16 位减法转换成 16 位加法
;-----------------------------------------
__KSCAN_JUGE:
        CLR         KCOUNT
        CLR         KDATA               ;表示本次检测有没有按键
        CLR         KSUB
_KEY_LOOP:
        CALL        GET_SUM_ADD
        LD          A,IAR               ;将 DATAOH、DATAOL 赋给 DATAH、DATAL
        LD          DATAL,A
        INCR        MP
        LD          A,IAR
        LD          DATAH,A

        RRCR        DATAH               ;除以 16
        RRCR        DATAL
        RRCR        DATAH
        RRCR        DATAL
        RRCR        DATAH
        RRCR        DATAL
        ;RRCR       DATAH
        ;RRCR       DATAL

        LDIA        0FH
        ANDR        DATAH
```

```
        COMR        DATAL               ;取反以做 16 位加法
        COMR        DATAH
;--------------------------------------
_GET_KEYF:                              ;获得当前检测按键的位，放入
KEYF_BAK。
         INCA         KCOUNT
         CALL         GET_KEY_DAT
         SUBA         KEYF
        SZB            FLAGS,Z
        JP             __KEY_HAVE_CHECK  ;检测的按键原来有按键
;--------------------------------------
_KEY_NO_CHK:                            ;检测的按键原来没有按键,判断按键按
下
        CLR            CD_COUNT            ;表示取几次旧值减去当前值

_KEY_SUB_LOOP:
         CALL         GET_CMDAT_ADD
        LD          A,CD_COUNT
        CALL           _GET_KEY_OLD       ;根据 COUNT 获得旧值保存值
K_TEMPL 和 K_TEMPH
_KEY_SUB:                               ;16 位减法程序,用新值加上旧值的补
码
        SETB           FLAGS,C                ;补码加 1
        LD          A,DATAL
        ADDCR       K_TEMPL
        LD          A,DATAH
        ADDCR       K_TEMPH
        SNZB        FLAGS,C
```

```
        JP          __KEY_NO_CHK_LONG
        ;JP         _KEY_MOVE           ;C 为 0，旧值小于新值
_KEY_DOWN:                              ;C 为 1，旧值大于等于新值
        TESTZ       K_TEMPH
        LDIA            0FFH
        SNZB        FLAGS,Z
        LD          K_TEMPL,A           ;如果高位不为 0 则赋低位 FFH，以方便只用低位判断

        CALL          GET_KEY_DOWN_DAT
        SUBA        K_TEMPL
        SNZB        FLAGS,C                 ;差值大于灵敏度表格表示有按键
        JP              __KEY_NO_CHK_LONG
        ;JP         _KEY_MOVE

        INCR            CD_COUNT            ;判断几次旧值，1 表示新值只比 KODAT1L 跟 KODAT1H 小
        LDIA            2;C_KSAVE           ;2 表示比 KODAT1L、KODAT1H,及 KODAT2L，KODAT2H 小
        SUBA        CD_COUNT
        SNZB        FLAGS,C
        JP              _KEY_SUB_LOOP

        CALL          GET_CMDAT_ADD
        INCR            MP                  ;MP=KODAT01H
        INCR            MP                  ;MP=KODAT2L
        LD          A,IAR
        INCR            MP                  ;MP=KODAT3L
```

```
        LD          IAR,A
        INCR            MP                  ;MP=KODAT23H
        LDIA            0FH
        ANDR        IAR
        SWAPA       IAR
        ORR         IAR
_KEY_HAVE:                                  ;确认有按键
__KEY_HAVE_WATER:
         INCA          KCOUNT
         CALL          GET_KEY_DAT
         SUBA          KEYF
        SZB             FLAGS,Z
        JP              _KEY_NEXT
        LD          A,K_TEMPL
        SUBA        KSUB
        SZB             FLAGS,C
        JP              _KEY_NEXT
        INCA          KCOUNT
        LD          KDATA,A                 ;KDATA 表示相应的临时按键标志位
        LD          A,K_TEMPL
        LD          KSUB,A
        JP              _KEY_NEXT
;*********************************************
__KEY_NO_CHK_LONG:                          ;判断保存的旧值（慢按键）
         CALL          GET_CMDAT_ADD
        LDIA            .3
        ADDR        MP                  ;MP=KODAT3L
        LD          A,IAR
        LD          K_TEMPL,A
```

```
        LDIA        .1
        ADDR        MP                  ;MP=KODAT23H
        SWAPA       IAR
        ANDIA       0FH
        LD          K_TEMPH,A

        SETB        FLAGS,C             ;补码加 1
        LD          A,DATAL
        ADDCR       K_TEMPL
        LD          A,DATAH
        ADDCR       K_TEMPH
        SNZB        FLAGS,C
        JP          _KEY_OLD_CHK        ;C 为 0，旧值小于新值

        TESTZ       K_TEMPH
        LDIA        0FFH
        SNZB        FLAGS,Z
        LD          K_TEMPL,A           ;如果高位不为 0 则赋低位 FFH，以方便只用低位判断

        CALL        GET_KEY_DOWN_DAT
        SUBA        K_TEMPL
        SNZB        FLAGS,C             ;差值大于灵敏度表格表示有按键
        JP          _KEY_OLD_CHK
        JP          _KEY_HAVE
;-------------------------------------
__KEY_HAVE_CHECK:                       ;检测的按键已经按下，判断按键放开
        CALL        GET_CMDAT_ADD
```

```
        LDIA        .3
        ADDR        MP                  ;MP=KODAT3L
        LD          A,IAR
        LD          K_TEMPL,A
        INCR        MP                  ;MP=KODAT23H
        SWAPA       IAR
        ANDIA       0FH
        LD          K_TEMPH,A
;KODAT3L,KODAT3H 还保存旧值，判断是否比旧值小于灵敏度值
_HAVE__KEY_SUB:                         ;16 位减法程序,用新值加上旧值的
补码
        SETB        FLAGS,C             ;补码加 1
        LD          A,DATAL
        ADDCR       K_TEMPL
        LD          A,DATAH
        ADDCR       K_TEMPH
        SNZB        FLAGS,C
        JP          _HAVE_KEY_OPEN      ;C 为 0，旧值小于新值,按键放
开
_HAVE__KEY_DOWN:                        ;C 为 1，旧值大于等于新值
        TESTZ       K_TEMPH
        LDIA        0FFH
        SNZB        FLAGS,Z
        LD          K_TEMPL,A           ;如果高位不为 0 则赋低位 FFH，以方便
只用低位判断

         CALL       GET_KEY_DOWN_DAT

        HSUBIA      02H
```

```
        SUBA        K_TEMPL
        SZB             FLAGS,C                 ;差值大于灵敏度表格表示有按键
        JP              _HAVE_KEY_RISE          ;差值大于灵敏度时，按键继续有效，判断上升沿
_HAVE_KEY_OPEN:                                 ;按键比旧值小的值小于灵敏度，按键放开
         CLR            KEYF
         CALL           GET_CMDAT_ADD
        DECR        MP                  ;MP=KODATOL
        LDIA            .6
        LD          CD_COUNT,A
_CLR_KOLD_LOOP:
        CLR             IAR
        INCR            MP
        SZDECR      CD_COUNT
        JP              _CLR_KOLD_LOOP
        JP              _KEY_NEXT

_HAVE_KEY_RISE:                                 ;确检测上升沿

         CALL           GET_CMDAT_ADD

        LD          A,IAR
        LD          K_TEMPL,A
        INCR            MP
        SWAPA       IAR
        ANDIA       0FH
        LD          K_TEMPH,A
```

```
        SETB        FLAGS,C                 ;补码加 1
        LD          A,DATAL
        ADDCR       K_TEMPL
        LD          A,DATAH
        ADDCR       K_TEMPH
        SZB         FLAGS,C
        JP          _HAVE_KEY_CONTINUE      ;C 为 0，旧值小于新值,判断按键放开
__HAVE_KEY_RISE_CHK:                        ;C 为 1，旧值大于等于新值,按键继续有效
        COMR        K_TEMPH
        COMR        K_TEMPL

        TESTZ       K_TEMPH
        LDIA        0FFH
        SNZB        FLAGS,Z
        LD          K_TEMPL,A

         CALL       GET_KEY_DOWN_DAT
         HSUBIA     02H
        SUBA        K_TEMPL
        SZB         FLAGS,C
        JP          _HAVE_KEY_OPEN
_HAVE_KEY_CONTINUE:                         ;按键继续有效

_HAVE_KEY_CONT_1:
        LDIA        .2                      ;KODAT3L、KODAT3H 不做处理，其它依次移位
```

```
        LD          CD_COUNT,A
         CALL           GET_CMDAT_ADD
        JP              __KEY_MOVE_KHAVE
;-------------------------------------
;连续 4 次值相差不超过 1，则保存至 KODAT3L 和 KODAT3H
_KEY_OLD_CHK:
        CLR             CD_COUNT            ;表示取几次旧值减去当前值

__KEY_NO_CHK_LOOP:
         CALL           GET_CMDAT_ADD
        LD          A,IAR
        LD          K_TEMPL,A
        INCR            MP                  ;MP=KODAT01H
        SWAPA       IAR
        ANDIA       0FH
        LD          K_TEMPH,A
        CALL            _CHK_DATA_SAME
        LD          K_TEMP,A
        SNZB        K_TEMP,0
        JP              _KEY_MOVE
        INCR            MP                  ;MP=KODAT2L
        LD          A,IAR
        LD          K_TEMPL,A
        INCR            MP                  ;MP=KODAT3L
        INCR            MP                  ;MP=KODAT23L

        LDIA            0FH
        ANDA        IAR
        LD          K_TEMPH,A
```

```
CALL        _CHK_DATA_SAME
LD          K_TEMP,A
SNZB        K_TEMP,0
JP          _KEY_MOVE

LDIA        .3
SUBR        MP              ;MP=KODAT01H
LDIA        0FH
ANDA        IAR
LD          K_TEMPH,A
LDIA        .2
SUBR        MP              ;MP=KODAT0L
LD          A,IAR
LD          K_TEMPL,A

CALL        _CHK_DATA_SAME
LD          K_TEMP,A
SNZB        K_TEMP,0
JP          _KEY_MOVE

LDIA        .4
ADDR        MP              ;MP=KODAT3L
COMA        DATAL

LD          IAR,A

INCR        MP              ;MP=KODAT23H
COMA        DATAH
```

```
        LD          IAR,A
        SWAPR       IAR

         CALL          GET_CMDAT_ADD
        JP              __KEY_MOVE_KHAVE

_CHK_DATA_SAME:
        SETB            FLAGS,C                 ;补码加 1
        LD          A,DATAL
        ADDCR       K_TEMPL
        LD          A,DATAH
        ADDCR       K_TEMPH
        SNZB        FLAGS,C
        JP              __CHK_DATA_SAME_1 ;C 为 0，旧值小于新值
        TESTZ       K_TEMPH
        SNZB        FLAGS,Z
        RET         .0          ;如果高位不为 0 则赋低位 FFH，以方便只用低位判断
        LDIA        .2
        SUBA        K_TEMPL
        SZB             FLAGS,C                 ;差值大于灵敏度表格表示有按键
        RET         .0
        RET         .1
__CHK_DATA_SAME_1:
        LDIA            .1
        ADDR        K_TEMPL
        SNZB        FLAGS,C
        RET         .0
```

```
        LDIA        .1
        ADDR        K_TEMPH
        SNZB        FLAGS,C
        RET         .0
        RET         .1
;-------------------------------------------------
_KEY_MOVE:                          ;没有按键，按键旧值依次保存
         CALL       GET_CMDAT_ADD
        JP          __KEY_MOVE_KHAVE
__KEY_MOVE_KHAVE:                   ;一直有按键的时候移位，最旧的值不变，
其它移位
        LD          A,IAR
        INCR        MP              ;MP=KODAT01H
        INCR        MP              ;MP=KODAT2L
        LD          IAR,A

        DECR        MP              ;MP=KODAT01H
        SWAPA       IAR
        ANDIA       0FH
        LD          K_TEMP,A

        LDIA        .3
        ADDR        MP              ;MP=KODAT23H
        LDIA        0F0H
        ANDR        IAR
        LD          A,K_TEMP
        ORR         IAR

        LDIA        .5
```

```
        SUBR        MP                  ;MP=KODAT0L
        LD          A,IAR
        INCR            MP                  ;MP=KODAT1L
        LD          IAR,A

        INCR            MP                  ;MP=KODAT01H
        SWAPR       IAR                 ;KODAT0H 放入 KODAT1H
        LDIA            0F0H
        ANDR        IAR

        COMA        DATAH
        ANDIA       0FH
        ORR         IAR

        LDIA            .2
        SUBR        MP                  ;MP=KODAT0L
        COMA        DATAL
        LD          IAR,A
_KEY_NEXT:
        INCR            KCOUNT
        LDIA            C_KCOUNT            ;按键个数
        SUBA        KCOUNT
        SNZB        FLAGS,C
        JP              _KEY_LOOP
         CALL           _CLR_KEY_SUM_RAM
        JP              _K_MAINK
;-----------------------------------------———
;*********************************************
_CLR_NO_KEY_DAT:
```

```
        CLR             KCOUNT
__CLR_NO_KEY_DAT_LOOP:

         INCA           KCOUNT
         CALL           GET_KEY_DAT
         SUBA           KEYF
        SZB              FLAGS,Z
        JP               __CLR_NO_KEY_DAT_NEXT
         CALL           GET_CMDAT_ADD
        CLR              IAR
        DECR        MP                  ;MP=KODAT0L
        CLR              IAR
        INCR             MP                 ;MP=KODAT1L
        INCR             MP                 ;MP=KODAT01H
        CLR              IAR
        INCR             MP                 ;MP=KODAT2L
        CLR              IAR
        INCR             MP                 ;MP=KODAT3L
        CLR              IAR
        INCR             MP                 ;MP=KODAT23H
        CLR              IAR
__CLR_NO_KEY_DAT_NEXT:
        INCR             KCOUNT
        LDIA             C_KCOUNT        ;按键个数
        SUBA        KCOUNT
        SNZB        FLAGS,C
        JP               __CLR_NO_KEY_DAT_LOOP
        CLR              KCOUNT
        RET
```

```
;*************************************************
TABLE_KEY_ADJ:
        DW          C_KEY1_ADJ
        DW          C_KEY2_ADJ
        DW          C_KEY3_ADJ
        DW          C_KEY4_ADJ
        DW          C_KEY5_ADJ
        DW          C_KEY6_ADJ
        DW          C_KEY7_ADJ
        DW          C_KEY8_ADJ
;-------------------------------------
;*************************************
TABLE_DOWN:
        DW          C_KEY1_LMD
        DW          C_KEY2_LMD
        DW          C_KEY3_LMD
        DW          C_KEY4_LMD
        DW          C_KEY5_LMD
        DW          C_KEY6_LMD
        DW          C_KEY7_LMD
        DW          C_KEY8_LMD
;*****************************************
TABLE_KEY_CH:
        DW          C_KEY1_CH
        DW          C_KEY2_CH
        DW          C_KEY3_CH
        DW          C_KEY4_CH
        DW          C_KEY5_CH
        DW          C_KEY6_CH
```

```
        DW          C_KEY7_CH
        DW          C_KEY8_CH
;-----------------------------------------
TABLE_KEY_DAT:                  ;按键旧值表格
        DW          K0DAT1L
        DW          K1DAT1L
        DW          K2DAT1L
        DW          K3DAT1L
        DW          K4DAT1L
        DW          K5DAT1L
        DW          K6DAT1L
        DW          K7DAT1L
;-----------------------------------------
TABLE_KEY_SUM_ADD:              ;按键求和保存值表格
        DW          DATA0L
        DW          DATA1L
        DW          DATA2L
        DW          DATA3L
        DW          DATA4L
        DW          DATA5L
        DW          DATA6L
        DW          DATA7L
;-----------------------------------------
;-----------------------------------------
TABLE_KEY_YX_DATA:
        DW          000H
        DW          001H
        DW              002H
        DW              004H
```

```
        DW              008H
        DW              010H
        DW              020H
        DW              040H
        DW              080H
;***********************************************
_CLR_KEY_SUM_RAM:
        BANKSEL_BANK1
        CLR             DATA0L
        CLR             DATA0H
        CLR             DATA1L
        CLR             DATA1H
        CLR             DATA2L
        CLR             DATA2H
        CLR             DATA3L
        CLR             DATA3H
        CLR             DATA4L
        CLR             DATA4H
        CLR             DATA5L
        CLR             DATA5H
        CLR             DATA6L
        CLR             DATA6H
        CLR             DATA7L
        CLR             DATA7H
        CLR         CD_COUNT
        RET
;-----------------------------------------
;根据 ACC，获得旧值，入口 MP=K0DAT1L
;出口 K_TEMPL、K_TEMPH 存放旧值低位、高 4 位
```

```
;ACC=0，MP=KODAT1L,+0
;ACC=1，MP=KODAT2L，+2
;ACC=2，MP=KODAT3L，+3
_GET_KEY_OLD:
        LD          K_TEMP,A
        TESTZ       K_TEMP
        SNZB        FLAGS,Z
        INCA        K_TEMP
        ADDR        MP

        LD          A,IAR
        LD          K_TEMPL,A           ;低位放入 K_TEMPL

        LDIA            .1
        SZB             K_TEMP,0
        LDIA            .2
        ADDR        MP                  ;MP=KODAT1H

        LD          A,IAR
        SNZB        K_TEMP,0                ;=1 表示 KODAT2H，不用高低交换
        SWAPA       IAR                 ;KODAT1H 保存在高 4 位
        ANDIA       0FH
        LD          K_TEMPH,A           ;高位放入 K_TEMPH
        RET
;*********************************************
;---------------------------------------
GET_SUM_ADD:
        LDIA        TABLE_KEY_SUM_ADD$H  ; 将 地 址 的 高 位 赋 给
"TABLE_SPH"
```

```
        LD          TABLE_SPH,A
        LDIA            TABLE_KEY_SUM_ADD$L  ; 将 地 址 的 低 位 赋 给
"TABLE_SPL"
        ADDA        KCOUNT
        LD          TABLE_SPL,A
        SZB          FLAGS,C
        INCR         TABLE_SPH
        TABLEA
        LD          MP,A                    ;MP=DATA0L
        RET
;-----------------------------------------
GET_CMDAT_ADD:
         LDIA           TABLE_KEY_DAT$H
        LD          TABLE_SPH,A
        LDIA            TABLE_KEY_DAT$L
        ADDA        KCOUNT
        LD          TABLE_SPL,A
        SZB          FLAGS,C
        INCR         TABLE_SPH
        TABLEA
        LD          MP,A                    ;MP=KODAT1L
        RET
;-----------------------------------------
GET_KEY_DOWN_DAT:
        LDIA            TABLE_DOWN$H
        LD          TABLE_SPH,A
        LDIA            TABLE_DOWN$L
        ADDA        KCOUNT
        LD          TABLE_SPL,A
```

```
        SZB         FLAGS,C
        INCR        TABLE_SPH
        TABLEA
        RET
;-----------------------------------------
GET_KEY_DAT:
        ADDIA       TABLE_KEY_YX_DATA$L
        LD        TABLE_SPL,A
        LDIA        00H
        SZB         FLAGS,C
        LDIA        01H
        ADDIA       TABLE_KEY_YX_DATA$H
       LD         TABLE_SPH,A
        TABLEA
        RET
;-----------------------------------------
_K_MAINK:
        TESTZ      KDATA
        SZB           FLAGS,Z
        JP            _KNO

        LD         A,KDATA                ;不为 0 则判断是否同一按键
        SUBA       KOLD
        SNZB       FLAGS,Z
        JP            _KFLASH

        INCR          KEYC
        LDIA          C_KEYC              ;跟上次相同则再检测 2 次
        SUBA       KEYC
```

```
        SNZB        FLAGS,C
        JP              __K_MAINK_BACK
        CLR             KEYC

         LD             A,KDATA
         CALL           GET_KEY_DAT
        LD          KEYF,A
        SETB            B_KEY_EN
        CLR             KOLD
        CALL            _CLR_NO_KEY_DAT        ;检测到按键清除没有按键的按键保存值
        JP              __K_MAINK_BACK
_KFLASH:
        LD          A,KDATA                    ;跟上次不同，保存这次的 KCOUNT 值到 KOLD
        LD          KOLD,A
        CLR             KEYC
        JP              __K_MAINK_BACK
_KNO:
        SZR         KEYF
        JP          _KSCAN_LONG
        CLR             KEYC
        CLR             KOLD
        CLR         KLONGC
        CLRB        B_KEY_UP
        CLRB        B_KEY_LONG
        JP              __K_MAINK_BACK
_KSCAN_LONG:
        INCR            KLONGC
```

```
        LDIA            C_FRIST_TIME
        SZB         B_KEY_LONG
        LDIA            C_LONG_TIME
        SUBA        KLONGC
        SNZB        FLAGS,C
        JP          __K_MAINK_BACK
        CLR         KLONGC
        SETB            B_KEY_LONG
        SETB            B_KEY_EN
        JP          __K_MAINK_BACK
__K_MAINK_BACK:
_KEY_BACK:
        BANKSEL_BANK0
        RET
;***********************************************
```

```
;*********************************************
;应用实例
;*********************************************
;*********************************************
;*********************************************
        #INCLUDE        CMS69F6162.H
;*********************************************
;*********************************************

GPR0        EQU         25H    ;中断现场保护寄存器必须用 BANK1,BANK0 公用寄存器
GPR1        EQU         26H    ;中断现场保护寄存器必须用 BANK1,BANK0 公用寄存器
TCOUNT      EQU         ?      ;系统计时，125US 加 1
;-------------------------------------------
;工作 RAM
;;*************************************************************
;*************************************************************
        ORG         00H
        JP              START
        ORG         01H
;-------------------------------------------
INT_IN:
        LD          GPR0,A          ;中断现场保护
        SWAPR       GPR0
        SWAPA       FLAGS
        LD          GPR1,A
;-------------------------------------------
INT_TMR1:                           ;只用到 1 个中断，TMR1 中断每 125US 进一
```

```
次
        BANKSEL_BANK0
        INCR            TCOUNT
        CLR             INT_FLAG
;-----------------------------------------
INT_BACK:
        SWAPA       GPR1
        LD          FLAGS,A
        SWAPA       GPR0            ;中断返回
        RETI
;*****************************************

;*****************************************
;*****************************************

;*****************************************
START:
        NOP
        CLRWDT                              ;上电清 WDT（必要）
;----------初始化 I/O----------------------;设置 IO 口的初始状态
        CLR             P0
        SET         P1
        LDIA            03H
        LD          P2,A

        LDIA            B'11111111'
        LD          P0CH,A
        LDIA            B'11111111'         ;P0,0-P0,7 为触摸按键检测口
        LD          P0CL,A
```

```
        LDIA        B'01110010'     ;P1,6 检测 AD（灵敏度调节）
        LD          P1CH,A          ;P1.5 灵敏度电容，先输出低放电
        LDIA        B'10101010'     ;P1,0 BCD 码高位，其它口未用
        LD          P1CL,A

        LDIA        B'00010010'     ;P2.0 编码输出
        LD          P2C,A           ;P2.2 为防水/抗干扰选择

;-----------清寄存器 BANK0--------------------------
CLR_RAM:                            ;间接寻址清掉所有 RAM
        BANKSEL_BANK0
        LDIA        0FH
        LD          MP,A
CLR_LOOP:
        INCR        MP
        CLR         IAR
        LDIA        07FH
        SUBA        MP
        SNZB        FLAGS,C
        JP          CLR_LOOP
;***************清寄存器 BANK1************************
CLR_RAM_1:
         BANKSEL_BANK1
        LDIA        02FH
        LD          MP,A
CLR_LOOP_1:
        INCR        MP
        CLR         IAR
```

```
        LDIA            07FH
        SUBA        MP
        SNZB        FLAGS,C
        JP              CLR_LOOP_1


;-------------上电延时-----------------------
INIT_DELAY:
         BANKSEL_BANK0
        LDIA            .0
        LD          TCOUNT,A
INIT_LOOP:                              ;上电延时 3*256*256*5/2=500MS
        CLRWDT
        SZDECR      TCOUNT
        JP              INIT_LOOP
;-------------------------------------------
;-----------初始化 RAM-------------------------
INIT_RAM:
         BANKSEL_BANK0
         CLR            TCOUNT
        CLR         PWM8DATA
        CLR         PWM10DATA
        CLR         TABLE_SPL   ;
        CLR         BUZCON
        CLR         INT_EXT         ;
        CLR         TABLE_SPH   ;
        CLR           08H
        CLR           0EH
        CLR           0FH
        CLR           25H
```

```
        CLR         26H
        CLR         27H
        CLR         2AH
        CLR         2BH

        LDIA          .6
        LD        TMR1,A            ; TMR1 初值设定
        LDIA          80H               ; TMR1 设定为定时器，分频比 1：1
        LD        TMR1C,A
        SETB          TMR1C,4               ; TMR1 开启

        LDIA          03H
        LD        SYS_GEN,A         ;中断使能，AD 禁止
        LDIA          08H               ;TMR1 中断使能，其它中断关闭
        LD        INT_EN,A

;*******************************************
;*******************************************
;*******************************************
MAIN:
         CALL       SET_SYS_1
         BANKSEL_BANK0
        LDIA          .20               ;125us*10=1.25ms 一个主循环
        SUBA      TCOUNT
        SNZB      FLAGS,C
        JP            MAIN
;*******************************************
        CLR           TCOUNT
        CLRWDT
```

```
;*********************************************
;*********************************************
MAIN_SUB:
        CALL            SET_SYS
        CALL        KSCAN               ;触摸扫描程序
        CALL          MAINK
        JP              MAIN
;*********************************************
;*********************************************
SET_SYS:
         BANKSEL_BANK0
        LDIA            B'11111111'
        LD          P0CH,A
        LDIA            B'11111111'         ;P0,0-P0,7 为触摸按键检测口
        LD          P0CL,A
        LDIA            B'10010001'         ;P1,6 检测 AD（灵敏度调节）
        LD          P1CH,A          ;P1.5 灵敏度电容，先输出低放电
        CLR         28H
        CLR         29H
SET_SYS_1:
        LDIA            03H
        LD          SYS_GEN,A       ;中断使能，AD 禁止
        LDIA            08H                 ; TMR1 中断使能，其它中断关闭
        LD          INT_EN,A
        LDIA            .6
        LD          TMR1,A              ; TMR1 初值设定
        LDIA            90H
        SUBA        TMR1C               ;TMR1 在没有乱的情况下不进行写操作，
以免影响定时
```

```
        SZB             FLAGS,Z
        JP              SET_SYS_BACK
        LDIA            80H                 ; TMR1 设定为定时器，分频比 1：1
        LD          TMR1C,A
        SETB            TMR1C,4             ; TMR1 开启
SET_SYS_BACK:
        RET
;****************************************
;****************************************
MAINK:
        SNZB        B_KEY_EN
        JP          MAINK_BACK
        CLRB        B_KEY_EN
        SZB         B_KEY_LONG
        JP          MAINK_LONG
        NOP
        JP          MAINK_BACK
MAINK_LONG:
        NOP
        JP          MAINK_BACK
MAINK_BACK:
         RET
;********************************************
KSCAN：
;插入”触摸按键检测范例程序”文件
         #INCLUDE  "触摸按键扫描范例程序.ASM"
;********************************************
        END
```