```
    .area   RAM    (REL)                ;RAM area is relative

    .globl  TickCtr

TickCtr:     .ds     1                  ;7.983mS tick counter

    .area   CODE    (REL)               ;program area CODE is relative

    .globl MainLoop

    .include ".\Hardware.asm"

    ;--------------------------------------
    ;Main program loop
    ;--------------------------------------

MainLoop:                               ;
    xor     a                           ;
    ld      (FlashCursor),a             ;disable flashing cursor by defaul

.if USE_PU_LED_CODES
    ld      a,#LEDMLINIT
    out     (LEDPORT), a
.endif

    call    DetectExpansion             ;Detect expansion ports

.if USE_PU_LED_CODES
    ld      a,#LEDMLDEDONE
    out     (LEDPORT), a
.endif


.if SUPPORT_HARD_DISK
    in      a,(SWITCHPORT)              ;read in the motherboard switches
    and     #SW_MONITOR_ENABLED         ;check if monitor enabled switch i
    jr      z,SkipMonitor               ;if (ROM monitor is enabled - swit
.endif

MLRepeat:                               ;  Repeat
    ld      a,(FlashCursor)             ;
    or      a                           ;
    jr      z,MLNoKybd                  ;    if (cursor needs to be flashe
    ld      a,(KybdBase)                ;      get keyboard expansion base
    or      a                           ;
```

```
    jr      z,MLNoKybd                  ;      if (keyboard is present)
    add     a,#ExpXOffset               ;
    ld      c,a                         ;
    in      a,(c)                       ;        read in tick counter
    cpl                                 ;        invert it
    ld      (TickCtr),a                 ;        store it in memory
                                        ;      endif
MLNoKybd:                               ;    endif

.if USE_PU_LED_CODES
    ld      a,#LEDMLBFMONITOR
    out     (LEDPORT), a
.endif
                                        ;
    call    Monitor                     ;    Process user commands

.if USE_PU_LED_CODES
    ld      a,#LEDMLFCNDONE
    out     (LEDPORT), a
.endif
                                        ;
    jp      MLRepeat                    ;  Until (forever)

.if SUPPORT_HARD_DISK
SkipMonitor:                            ;else
    call    BootDrive                   ;  load and execute OS from boot s
    jp      MLRepeat                    ;  *jump to ROM monitor if boot lo
.endif
```