```
    .title "EL308 Lab#2"
    .sbttl "Counter example for egg timer"
    .equ __24FJ256GB110, 1
    .include "p24FJ256GB110.inc"

    .global __reset          ;The label for the first line of code.
    .global __T1Interrupt    ;Declare Timer 1 ISR name global

.bss
.section .const,psv
    stock:      .ascii "ALYAG 0,58 \1%13,73  *  "
                .ascii "VANET 2,49 \1%10,18  *  "
                .ascii "USAK 0,71 \0%-10,13  *  "
                .ascii "DARDL 0,29 \0%-4,35  *  "
    stockend:

    DArrow:     .byte 0b00001110
                .byte 0b00001110
                .byte 0b00001110
                .byte 0b00001110
                .byte 0b00011111
                .byte 0b00001110
                .byte 0b00000100
                .byte 0b00000000
    UArrow:     .byte 0b00000100
                .byte 0b00001110
                .byte 0b00011111
                .byte 0b00001110
                .byte 0b00001110
                .byte 0b00001110
                .byte 0b00001110
                .byte 0b00000000

.text                           ;Start of Code section
__reset:
    mov     #__SP_init, W15     ; Initalize the Stack Pointer
    mov     #__SPLIM_init, W0   ; Initialize the Stack Pointer Limit Register
    mov     W0, SPLIM
    nop                         ; Add NOP to follow SPLIM initialization

        ;<<insert more user code here>>

    call    init_PSV
    call    init_LED
    call    init_LCD
    call    init_keypad
    call    init_buzzer

    call    init_timer2

    mov.b   #0b01000000, W0     ; CG RAM
    call    sendcomm

    mov     #psvoffset(DArrow), W12
    mov     #16, W2
program_char:
    mov.b   [W12++], W0
;   call    senddata

    bset    PORTB,#15   ; select LCD data register
    nop
    mov     W0, PORTE   ; output data
    bset    PORTD, #4
    repeat  #15
    nop
    bclr    PORTD, #4
    call    wait_data
;   call    dly

    dec     W2, W2
    cp0     W2
    bra     NZ, program_char

    mov.b   #0b10010000, W0     ; DD RAM
    call    sendcomm
```

```
    mov     #psvoffset(stock), W10
    mov     #psvoffset(stockend), W11



    mov     #0x0018, W0     ; shift display to left
    mov     #0x0007, W0
    call    sendcomm
start_display:

    mov.b   [W10++], W0
    call    senddata

wait_1:
    btss    TMR2, #13
    bra     wait_1
    bset    PORTF, #3
wait_0:
    btsc    TMR2, #13
    bra     wait_0
    bclr    PORTF, #3

    cp      W10, W11
    bra     NZ, start_display
    mov     #psvoffset(stock), W10

    bra     start_display

; -----------------------------------------------------
; !!!!!!!!!!!!!!!!! Functions !!!!!!!!!!!!!!!!!!!!!!!!
; -----------------------------------------------------

init_timer2:
    bclr    T2CON, #TON     ; turn timer1 OFF

    bset    T2CON, #TCKPS1
    bclr    T2CON, #TCKPS0  ; set prescaler to 64

    bclr    T1CON, #TCS     ; select internal clock

    mov     #0x0000, W0
    mov     W0, TMR2        ; clear TMR1 register
    mov     #15625, W0
    mov     W0, PR2         ; set timer1 period to 32150 -> f=2e6/64/15625=2 Hz

    bset    T2CON, #TON     ; turn timer1 ON
    return


init_PSV:
    mov     #psvpage(stock), W0
    mov     W0, PSVPAG      ; set PSVPAG
    bset.b  CORCONL,#PSV    ; enable Program Space Visibility
    return

init_LED:
    bclr    TRISF, #0
    bclr    TRISF, #1
    bclr    TRISF, #2
    bclr    TRISF, #3       ; LED array
    return

init_LCD:
    bclr    TRISB, #15
    bclr    PORTD, #4       ; make sure LCD is disabled before port is set to output mode
    bclr    TRISD, #4
    bclr    TRISD, #5
    mov     #0xFF00, W0
    mov     W0, TRISE

    bclr    PORTD, #5       ; select LCD WR mode

    mov     #0x0038, W0     ; init LCD
;   mov     #0x0030, W0     ; init LCD as 1 line
    call    sendcomm
    call    dly
```

```
    call    dly
    call    dly

    mov     #0x000C, W0     ; LCD on, cursor off
    call    sendcomm
    mov     #0x0001,W0      ; clear LCD
    call    sendcomm
    return

wait_data:
    push    W0
    push    W1
    mov     TRISE, W0
    mov     #0x00FF, W1
    ior     W0, W1, W0
    mov     W0, TRISE       ; make port input
    nop

    bclr    PORTB,#15   ; select LCD command register
    nop
    bset    PORTD, #5   ; select LCD RD mode
    nop
    bset    PORTD, #4   ; enable

    mov     #0x0080, W1
wait_for_0:
    mov     PORTE, W0
    and     W0, W1, W0
    cp0     W0
    bra     NZ, wait_for_0

    bclr    PORTD, #4   ; disable
    nop

    mov     TRISE, W0
    mov     #0xFF00, W1
    and     W0, W1, W0
    mov     W0, TRISE       ; make port output
    nop

    bclr    PORTD, #5       ; select LCD WR mode

    pop     W1
    pop     W0
    return

sendcomm:
    bclr    PORTB,#15   ; select LCD command register
    mov     W0, PORTE   ; output command
    bset    PORTD, #4
    call    dly
    nop
    bclr    PORTD, #4
    call    dly
    return

senddata:
    bset    PORTB,#15   ; select LCD data register
    mov     W0, PORTE   ; output data
    bset    PORTD, #4
    call    dly
    nop
    bclr    PORTD, #4
    call    dly
    return

dly:
    mov     #0x2000,W0
dlyloop:
    sub     W0, #1, W0
    bra     NZ, dlyloop
    return

init_keypad:
    bset    TRISD,#0    ; DATA A
    bset    TRISD,#1    ; DATA B
```

```
    bset    TRISD,#2    ; DATA C
    bset    TRISD,#3    ; DATA D
    bset    TRISD,#6    ; DATA Available
    return

init_buzzer:
    bclr    PORTD, #13  ; buzzer initially OFF
    bclr    TRISD, #13  ; enable output
    return

.end                            ;End of program code in this file
```