```
; Title clk_set9.asm
; Clock speed = 32.768kHz
; Author Alan Swainston
;
; Version 9
; 25/11/2008
;
; Modified to gain more accuracy from the clock which was losing 140 secs in 24 hrs.
; Done by adding 6 seconds per hour. Further correction added per week, Minus 17 secs
; This program operates at 32.768 kHz which is divided down using a
; prescalar of 256 counting TMR0 up from 224 to give a further divide by 32 which gives
; 1 second pulses. The xtal runs too slow!!
; 180 of these pulses (seconds) are counted to increase a binary count
; every 3 minutes on PORTB. This provides the input to a DtoA converter,
; the output of which drives a meter (clock).
; In addition a switch connected to PORTA0 is used to provide a fast
; setting capability. This provides an initial increase on PORTB on the first button
; press.  A second then passes before PORTB is increased once very 30mS.


    list    P=PIC16F84
    include C:\Program Files\Microchip\MPASM Suite\p16F84.inc

    __config _CP_OFF & _WDT_OFF & _XT_OSC & _PWRTE_OFF

;***************** EQUATES AND DEFINES  ******************


dlay1   equ 20h     ; inner  timer loop counter
dlay2   equ 21h     ; middle timer loop counter
D_A     equ 23h     ; index value to be output on portB
chk_but equ 24h     ; used in "main" to check if exit from "one_sec"
THR_MIN equ 25h     ; used to check when one second has elapsed
HOUR    equ 26h     ; used to check when three minutes have elapsed
WEEK    equ 27h     ; used to count number of 3 minute blocks per hour
#define button  porta,0     ; Define button connecton

    org 0       ; Program to be assembled
                ; starting at memory location 0


; **************** INITIALISATION ROUTINE *********************

    bsf status,rp0  ; select bank 1
    movlw   01h     ; set up porta line 0 as input
    movwf   trisa ^ 80h
    movlw   00h     ; set up portb as output
    movwf   trisb ^ 80h
    movlw   b'00000111' ; Set prescalar at 256
    movwf   OPTION_REG ^ 80h
    bcf status,rp0  ; select bank 0
    clrf    portb
    clrf    D_A
    clrf    chk_but
    movlw   .180
    movwf   thr_min
    movlw   .20
    movwf   hour
    movlw   .168
    movwf   week
    call    rs_tmr0
; %%%%%%%%%%%%%%%%%%  MAIN PROGRAM START  %%%%%%%%%%%%%%%%%%%%%%

; if button is down PORTA0 is clear

start
    btfsc   button      ; check if button is pressed
    goto    op_clk      ; if not test again
    call    inc_pt1     ; increase the value on portB
    call    one_sec     ; delay for one second or detect the button up
    btfsc   chk_but,0   ; skip if button is down on exit from one_sec
    goto    start       ; go and wait for next button press
but_dn
    btfsc   button      ; check if button is down
    goto    but_up1     ; if not goto button_up1 and reset counters
    call    inc_pt2     ; increase value on portB
```

```
    call    thirty_ms    ; every 30 mS
    goto    but_dn       ; loop again
but_up1
    call    thirty_ms    ; button up so debounce
    call    rs_tmr0
    call    rs_thrmin
    call    rs_hour
    call    rs_week
    goto    start        ; go and wait for next button press
op_clk
    movf    TMR0,w       ; Has a second elapsed?
    btfss   STATUS,z
    goto    START        ; No so go back and check for button press
    call    rs_tmr0      ; Yes So reset TMR0
    decfsz  THR_MIN,f    ; Has 3 minutes elapsed
    goto    START        ; No so go back and check for button press
    call    rs_thrmin    ; Yes so reset three minute counter
    call    inc_pt2      ; Increase 12 hour counter
    goto    START        ; go and check for button press

; %%%%%%%%%%%%%%%%%%%%%%  Subroutines  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

rs_tmr0
    movlw   .224
    movwf   TMR0
    return
rs_thrmin
    decf    hour,f
    btfsc   STATUS,z     ; If 1 hour has elapsed
    goto    hr_tim_corr     ; apply +6 sec. timing correction
    movlw   .180
    movwf   thr_min
    return
hr_tim_corr
    decf    week,f
    btfsc   status,z     ; If week has elapsed
    goto    wk_tim_corr ; apply -11 sec. correction
    movlw   .174         ; reset minute counter
    movwf   thr_min
rs_hour
    movlw   .20          ; reset hour counter
    movwf   hour
    return
wk_tim_corr
    movlw   .191
    movwf   thr_min
    movlw   .20
    movwf   hour
rs_week
    movlw   .168
    movwf   week
    return

inc_pt1
    call    thirty_ms    ; debounce after button press
inc_pt2
    incf    D_A,f        ; increase D to A counter
    movlw   .240         ; and check if 12 hours has been reached
    subwf   D_A,w
    btfsc   STATUS,z
    goto    ZERO_D_A     ; if yes reset counter and portB
    incf    PORTB,f      ; else increase counter and portB by one
    return
ZERO_D_A
    clrf    D_A      ; clear counter
    clrf    PORTB        ; clear portB
    return

; %%%%%%%%%%%%%%%%%%%%%%  31 mS delay  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

thirty_ms
    clrf    dlay1        ; ensure the loop register starts at zero
iloop31 decfsz  dlay1,f      ; decrement the loop register
    goto    iloop31      ; if not go back and reduce by one more
    return
```

```
; %%%%%%%%%%%%%%%%%%%%%%%%  1 Sec. delay and button check %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

one_sec
    movlw   .10     ; set number of times through the outer loop
    movwf   dlay2       ; and store it in the register called dlay2
    clrf    dlay1       ; ensure the inner loop register starts at zero
iloop1  decfsz  dlay1,f     ; decrement the inner loop register
    goto    iloop1      ; if not go back and reduce by one more
    btfsc   button      ; skip if button is down and continue 1 second countdown
    goto    but_up      ; else exit from the one second loop
    decfsz  dlay2,f     ; inner loop now zero so reduce outer loop counter by one
    goto    iloop1      ; if not go through inner loop again
    clrf    chk_but     ; indicate to main program that 1 second has matured
    return
but_up
    call    thirty_ms   ; debounce the button
    movlw   .1
    movwf   chk_but     ; indicate to main program that button is up
    call    rs_tmr0     ; after set up start TMR0
    call    rs_thrmin   ; at the start of three minutes
    call    rs_hour     ; reset hour counter
    return


    end
```