AN582

Low-Power Real-Time Clock

Author: *Mark Palmer*
Microchip Technology Inc.

INTRODUCTION

This application note uses the Timer1 module, from a mid-range PIC16CXXX microcontroller, to control a low-power real-time clock. Timer1 was chosen because it has its own crystal which allows the module to operate during sleep. The two events that will wake the device from sleep (for this application) are a keypress and a Timer1 overflow.

OPERATION

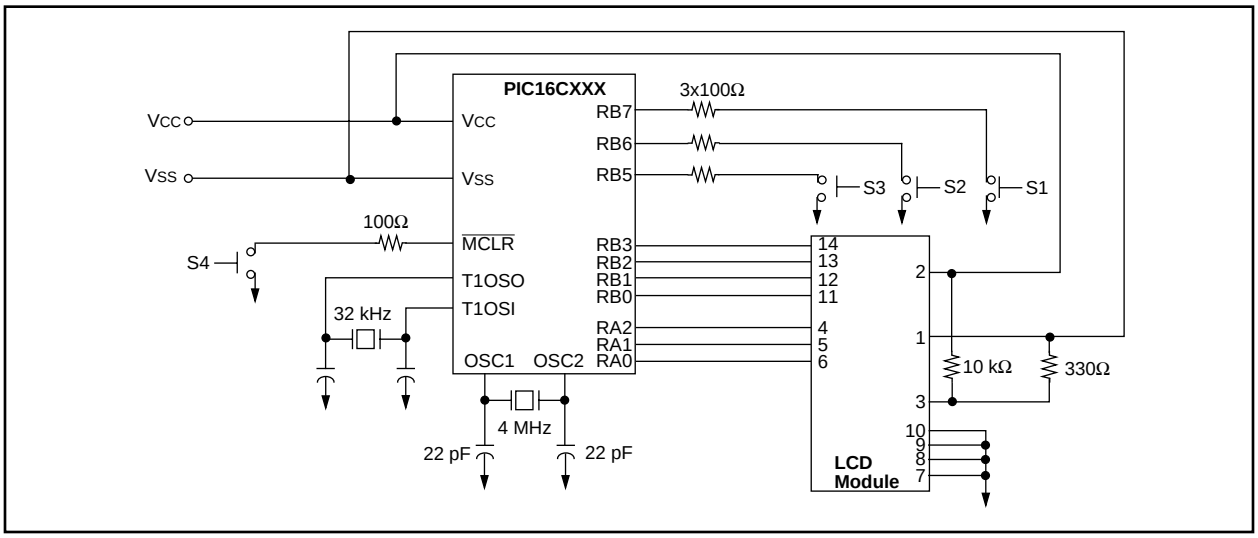
Upon power-up, the device is initialized with the display starting at 12:00 PM, and Timer1 is configured to generate an interrupt (every second). The Timer1 overflow interrupt wakes the device from sleep. This causes the time registers (HRS, MIN, SECS) to be updated. If the SECS register contains an even value (SECS<0> = 0), the colon (":") is not displayed. This gives a visual indication for each second. Then the device returns to sleep.

There are three keys for the setting of the clock. The SELECT_UNITS Key (S1) selects which units are to be modified (hours, minutes, off). The selected units are blanked for a second then flashed for one second. The INC Key (S2) increments the selected units. While incrementing, the selected units values are displayed. Upon key release, the Timer counts out one second and begins flashing the selected units. The CLR_MIN Key (S3) clears the minutes and seconds. CLR_MIN is useful for exactly setting the time to the "top of the hour" as announced in radio broadcasts. After the INC or SELECT_UNITS keys are depressed, the user has ten seconds to depress the next key. If no keypress is detected within ten seconds, the unit returns to the clock mode.

To simplify the design time and minimize cost, a standard Hitachi LCD display module is used. Most applications that require LCDs use a custom LCD display. The LCD interface software would need to be modified to suit the specific LCD display driver being used.

Figure 1 is a block diagram of the design. The RA2:RA0 pins are the control signals to the LCD display, RB3:RB0 acts as a 4-bit data bus, and RB7:RB5 are the input switches. The OSC1 pin is connected to an RC network, which generates an approximate 4 MHz device frequency. Because Timer1 operates asynchronously to the device, the device's oscillator can be configured for RC mode. RC oscillator mode is the least expensive and has the quickest start-up time. Timer1 is where an accurate frequency is required. Timer1's crystal is connected to the T1OSI and T1OSO pins. A good choice for a crystal is a 32.786 kHz (watch) crystal. Table 1 is a list of the components and their part numbers.

FIGURE 1: CLOCK BLOCK DIAGRAM



Relative to most microelectronics, LCD's are slow devices. A good portion of the time spent in the Interrupt Service Routine, is talking to and updating the LCD module. To minimize power consumption, the device should be in SLEEP mode as much as possible.

By using the conditional assembly, if a flag (called Debug) is true, the total time spent in the subroutine can be seen on the PORTD<0> pin (the high time). Measuring this time on an oscilloscope displayed a typical time of 800 µs that the device is awake. This 800 µs operation is out of the 1 second time that the device needs to service the interrupt (a Timer1 overflow).

The accuracy of a real-time clock using Timer1 depends on the accuracy of the crystal being used. The more accurate the crystal, the higher the cost. So as always there is a cost / performance trade-off to be made. A crystal rated with an accuracy of 20 PPM (parts per million), could cause an error of about 1.7 seconds per day. For many applications, this should be adequate (said from someone who doesn't wear a watch).

The program written for this application note shows one method for a real-time clock. Trade-offs between code size, current consumption and desired operation have been made. Some possible alternative implementations are:

1. When displaying the time, update only the characters that changed.
2. Turn off the display during sleep
3. LCD module data interface of 8-bits, as-opposed-to the 4-bit interface.

Alternative 1 can reduce the time awake by keeping track of which characters need to be updated. The majority of the time it will be only the position which contains either the ":" or the " ". Next would be the ones place of the minutes, then the tens place of the minutes, etc. The display would only need to be completely updated 2 times every 24 hours. This would reduce the amount of time talking with the LCD display at the cost of some program / data memory.

Depending on the requirements of the application and the characteristics of the display, Alternative 2 could be implemented by turning the power off and on (at a given rate) to the display. This technique may lead to a lower system current consumption. Evaluation of the desired display / display driver is recommended.

Alternative 3 uses the LCD module in an 8-bit mode, which will reduce the size of the display routines (save about 20 words of program memory) at the cost of four additional I/O lines. For some applications this may be a good trade-off to get the additional program memory space. The percentage of operating time saved is slight and should not give substantial power savings.

TABLE 1: LIST OF COMPONENTS†

Description	Part Number	Manufacturer	Quantity
LCD Module (2 x 20 Characters)	LM032L	Hitachi	1
Switches	EVQPADO4M	Panasonic	4
Microcontroller	PIC16C64 / 74	Microchip	1
32.768 kHz Crystal	NC26 / NC38	FOX	1
4 MHz Crystal	ECS-40-20-1	ECS	1

† Most components available from DigiKey.

CONCLUSION

The Timer1 module allows many applications to include a real-time clock at minimal system cost. This time function can be useful in consumer applications (display time) as well as in industrial applications (data time stamp). The accuracy of the time is strictly dependent on the accuracy of the crystal. Table 2 shows the program resource requirements.

TABLE 2: PROGRAM RESOURCE REQUIREMENTS

Resource			Words / Bytes	Cycles
Program Memory	Initialization		61	61
	Clock Operation	Increment Time WC	106	35 + Display
		Key Input WC		35 + Display Time
Data Memory	Display[2]		208	526[1]
	Variables		5	N.A.
	Scratch RAM		4	N.A.

[1] Dependent on LCD Module (re; BUSY_CHECK subroutine).
[2] Assumes worst case (WC) numbers and best case response from LCD module.

Please check the Microchip BBS for the latest version of the source code. Microchip's Worldwide Web Address: www.microchip.com; Bulletin Board Support: MCHIPBBS using CompuServe® (CompuServe membership not required).

APPENDIX A: SOURCE CODE LISTING (CLOCK_01.LST)

```
MPASM 01.40 Released            CLOCK.ASM   1-16-1997  17:05:59         PAGE  1


LOC  OBJECT CODE     LINE SOURCE TEXT
  VALUE

                     00001         LIST    P = 16C74, n = 66
                     00002         ERRORLEVEL  -302
                     00003 ;
                     00004 ;*****************************************************************************
                     00005 ;
                     00006 ; This program implements a real time clock using the TMR1 module of the
                     00007 ; PIC16CXXX family. A LCD display module is used to display (update) the time
                     00008 ; every second. Three keys are used to set the time.
                     00009 ;
                     00010 ;       Program = CLOCK.ASM
                     00011 ;       Revision Date:   5-15-94
                     00012 ;                        1-15-97    Compatibility with MPASMWIN 1.40
                     00013 ;
                     00014 ;*****************************************************************************
                     00015 ;
                     00016 ;
                     00017 ; HARDWARE SETUP
                     00018 ;   LCD Control Lines
                     00019 ;       RA0 = E     (Enable)
                     00020 ;       RA1 = RW    (Read/Write)
                     00021 ;       RA2 = RS    (Register Select)
                     00022 ;   LCD Data Lines
                     00023 ;       RB<3:0>
                     00024 ;   Switch Inputs
                     00025 ;       RB7 = Select Hour / Minute / Off
                     00026 ;       RB6 = Increment Hour / Minute
                     00027 ;       RB5 = Reset Minutes to 00
                     00028 ;
                     00029             INCLUDE <p16c74.inc>
                     00001         LIST
                     00002 ; P16C74.INC  Standard Header File, Version 1.00    Microchip Technology, Inc.
                     00318         LIST
                     00030
  00000000           00031 FALSE                   EQU             0
  00000001           00032 TRUE                    EQU             1
```

```
                    00033
                    00034             INCLUDE <CLOCK.h>
                    00076         list
                    00035 ;
  00000006          00036 LCD_DATA        EQU     PORTB           ; The LCD data is on the lower 4-bits
  00000086          00037 LCD_DATA_TRIS   EQU     TRISB           ; The TRIS register for the LCD data
  00000005          00038 LCD_CNTL        EQU     PORTA           ; Three control lines
                    00039 ;
  00000000          00040 PICMaster       EQU     FALSE           ; A Debugging Flag
  00000000          00041 Debug           EQU     FALSE           ; A Debugging Flag
  00000001          00042 Debug_PU        EQU     TRUE            ; A Debugging Flag
                    00043 ;
                    00044 ;
                    00045 ; Reset address. Determine type of RESET
                    00046 ;
0000                00047         org     RESET_V                 ; RESET vector location
0000 1683           00048 RESET       BSF     STATUS, RP0         ; Bank 1
0001 188E           00049             BTFSC   PCON, NOT_POR       ; Power-up reset?
0002 290C           00050             GOTO    START               ; YES
0003 295E           00051             GOTO    OTHER_RESET         ; NO, a WDT or MCLR reset
                    00052 ;
                    00053 ; This is the Periperal Interrupt routine. Need to determine the type
                    00054 ; of interrupt that occurred. The following interrupts are enabled:
                    00055 ;   1.  PORTB Change (RBIF)
                    00056 ;   2.  TMR1 Overflow Interrupt (T1IF)
                    00058     page
0004                00059         org     ISR_V                   ; Interrupt vector location
0004                00060 PER_INT_V
                    00061         if ( Debug )
                    00062             bsf     PORTD, 0            ; Set high, use to measure total
                    00063         endif                           ;     time in Int Service Routine
                    00064 ;
0004 1283           00065             BCF         STATUS, RP0     ; Bank 0
0005 180C           00066             BTFSC       PIR1, TMR1IF    ; Timer 1 overflowed?
0006 2843           00067             GOTO        T1_OVRFL        ; YES, Service the Timer1 Overflow Interrupt
0007 1C0B           00068             BTFSS       INTCON, RBIF    ; NO, Did PORTB change?
0008 28D0           00069             GOTO        ERROR1          ; NO, Error Condition - Unknown Interrupt
                    00070 ;
0009                00071 PORTB_FLAG                            ; Are any of PORTB's inputs active?
0009 0806           00072             MOVF        PORTB, W        ;
000A 39E0           00073             ANDLW       0xE0            ; Keep only the 3 switch values
000B 00B5           00074 DEBOUNCE    MOVWF       TEMP            ;
000C 3002           00075             MOVLW       DB_HI_BYTE      ; This is the debounce delay
000D 08B3           00076             MOVF        MSD, F          ;
000E 01B4           00077             CLRF        LSD             ;
000F 0BB4           00078 KB_D_LP1    DECFSZ      LSD, F          ;
0010 280F           00079             GOTO        KB_D_LP1        ;
```

```
0011 0BB3           00080             DECFSZ      MSD, F          ;
0012 280F           00081             GOTO        KB_D_LP1        ;
0013 0806           00082 END_DELAY   MOVF        PORTB, W        ;
0014 39E0           00083             ANDLW       0xE0            ; Keep only the 3 switch values
0015 02B5           00084             SUBWF       TEMP, F         ;
0016 1D03           00085             BTFSS       STATUS, Z       ; Is the Zero bit set?
                    00086                                         ;    (switches were the same on 2 reads)
0017 280B           00087             GOTO        DEBOUNCE        ; NO, Try another read
0018 00B5           00088 KEY_MATCH   MOVWF       TEMP            ; YES, need to see which is depressed.
                    00089 ;
0019 3080           00090             MOVLW       0x80            ; Since doing key inputs, clear TMR1
001A 008F           00091             MOVWF       TMR1H           ;   for 1 sec overflow.
001B 018E           00092             CLRF        TMR1L           ;
001C 100C           00093             BCF         PIR1, TMR1IF    ; Clear Timer 1 Interrupt Flag
                    00094
001D 1FB5           00095             BTFSS       TEMP, HR_MIN_SW         ; Is the hour-min-off switch depressed?
001E 2826           00096             GOTO        SELECT_UNITS            ; YES, specify the units selected
001F 1F35           00097             BTFSS       TEMP, INC_SW            ; Is the inc switch depressed?
0020 282B           00098             GOTO        INC_UNIT                ; YES, Increment the selected Units
0021 1EB5           00099             BTFSS       TEMP, CLR_MIN_SW        ; Is the clear minute switch depressed?
0022 2835           00100             GOTO        CLR_MIN                 ; YES, clear the minutes.
                    00101 ;
                    00102 ; No key match occured, or finished with PortB interrupt and need to clear interrupt condition.
                    00103 ;
0023                00104 CLR_RB                                        ; No RB<7:5> keys are depressed (rising edge Int.)
0023 0886           00105             MOVF        PORTB, F        ; Clear the PORTB mismatch condition
0024 100B           00106             BCF         INTCON, RBIF    ; Clear the PORTB Int Flag
                    00107         if ( Debug )
                    00108             bcf         PORTD, 0        ; Set low, use to measure total
                    00109                                         ;     time in Int Service Routine
                    00110         endif
0025 0009           00111             RETFIE                      ; Return / Enable Global Interrupts
                    00112 ;
                    00113     page
0026                00114 SELECT_UNITS
0026 30FF           00115             MOVLW       0xFF            ;
0027 00C0           00116             MOVWF       WAIT_CNTR       ; WAIT_CNTR has LSb set after each SELECT UNIT key press.
0028 0AA0           00117             INCF        FLAG_REG, F             ; Increment the pointer to the MIN_UNIT:HR_UNIT
0029 1620           00118             BSF         FLAG_REG, KEY_INPUT     ;
002A 2875           00119             GOTO        DISPLAY                 ; Flash the Display of the selected unit
                    00120 ;
002B                00121 INC_UNIT
002B 01C0           00122             CLRF        WAIT_CNTR               ; WAIT_CNTR is cleared to zero after each key press.
002C 1820           00123             BTFSC       FLAG_REG, HR_UNIT       ; Are the hour units selected?
002D 285C           00124             GOTO        INC_HRS                 ; YES, Increment the hour units
002E 1CA0           00125             BTFSS       FLAG_REG, MIN_UNIT      ; Are the minute units selected?
002F 2823           00126             GOTO        CLR_RB                  ; NO, Not a valid key. Clear flags
```

```
                    00127 ;
0030 0AB1           00128               INCF        MIN, F                  ; YES, Increment the minute units
0031 303C           00129               MOVLW       0x3C                    ; This is Decimal 60
0032 0231           00130               SUBWF       MIN, W                  ; MIN - 60 = ?
0033 1D03           00131               BTFSS       STATUS, Z               ; MIN = 60?
0034 2875           00132               GOTO        DISPLAY                 ; NO, display time
                    00133                                                   ; YES, MIN = 0 (use code from CLR_MIN)
0035 01B1           00134 CLR_MIN       CLRF        MIN                     ; MIN = 0
0036 3004           00135               MOVLW       0x04                    ; Clear the seconds
0037 00B2           00136               MOVWF       SECS                    ; Initial Second count = 4
0038 3080           00137               MOVLW       0x80                    ; Clear Timer 1, for 1 sec overflow
0039 008F           00138               MOVWF       TMR1H                   ;
003A 018E           00139               CLRF        TMR1L                   ;
003B 100C           00140               BCF         PIR1, TMR1IF            ; Clear the TMR1 overflow interrupt.
003C 01C0           00141               CLRF        WAIT_CNTR               ; WAIT_CNTR is cleared to zero after each key press.
003D 1AB5           00142               BTFSC       TEMP, CLR_MIN_SW        ; Is the clear minute switch depressed?
003E 2875           00143               GOTO        DISPLAY                 ; NO. Rollover from increment key
003F 10A0           00144               BCF         FLAG_REG, MIN_UNIT      ; YES, Clear ALL relevant flags
0040 1020           00145               BCF         FLAG_REG, HR_UNIT       ;
0041 1220           00146               BCF         FLAG_REG, KEY_INPUT     ;
0042 2875           00147               GOTO        DISPLAY                 ;
                    00148 ;
                    00149     page
                    00150 ;
0043                00151 T1_OVRFL
0043 100C           00152               BCF         PIR1, TMR1IF            ; Clear Timer 1 Interrupt Flag
0044 1E20           00153               BTFSS       FLAG_REG, KEY_INPUT     ; Are we using the key inputs?
0045 284F           00154               GOTO        INC_TIME                ; NO, Need to Increment the time
0046 0AC0           00155               INCF        WAIT_CNTR, F            ; YES,
0047 300A           00156               MOVLW       0x0A                    ; 10 counts x 1 seconds
0048 0240           00157               SUBWF       WAIT_CNTR, W            ; Has the 10 Sec wait for key expired?
0049 1D03           00158               BTFSS       STATUS, Z               ; Is the result 0?
004A 2875           00159               GOTO        DISPLAY                 ; NO, Display value
004B 01C0           00160               CLRF        WAIT_CNTR               ; YES, Clear WAIT_CNTR
004C 1220           00161               BCF         FLAG_REG, KEY_INPUT     ;
004D 1020           00162               BCF         FLAG_REG, HR_UNIT       ;
004E 10A0           00163               BCF         FLAG_REG, MIN_UNIT      ;
                    00164 ;
                    00165 ;
004F 3080           00166 INC_TIME          MOVLW   0x80                    ;
0050 008F           00167                   MOVWF   TMR1H           ; 1 Second Overflow
0051 0AB2           00168                   INCF    SECS, F         ;
0052 1F32           00169                   BTFSS   SECS, 6         ;
0053 2875           00170                   GOTO    DISPLAY         ;
0054 3004           00171                   MOVLW   0x04            ;
0055 00B2           00172                   MOVWF   SECS            ;
0056 0AB1           00173                   INCF    MIN, F          ;
```

```
0057 303C          00174                MOVLW   0x3C            ; W = 60d
0058 0231          00175                SUBWF   MIN, W          ;
0059 1D03          00176                BTFSS   STATUS, Z       ;
005A 2875          00177                GOTO    DISPLAY         ;
005B 01B1          00178                CLRF    MIN             ;
005C 0AB0          00179 INC_HRS        INCF    HRS, F          ;
                   00180
005D 300C          00181                MOVLW   0x0C            ; It is now 12:00, Toggle AM / PM
005E 0230          00182                SUBWF   HRS, W          ;
005F 1D03          00183                BTFSS   STATUS, Z       ;
0060 2867          00184                GOTO    CK_13           ; Need to check if HRS = 13
0061 1FA0          00185                BTFSS   FLAG_REG, AM    ; Was it AM or PM
0062 2865          00186                GOTO    SET_AM          ; Was PM, Needs to be AM
0063 13A0          00187                BCF     FLAG_REG, AM    ; It is PM
0064 2875          00188                GOTO    DISPLAY         ;
0065 17A0          00189 SET_AM         BSF     FLAG_REG, AM    ; It is AM
0066 2875          00190                GOTO    DISPLAY         ;
                   00191
0067 300D          00192 CK_13          MOVLW   0x0D                    ; Check if HRS = 13
0068 0230          00193                SUBWF   HRS, W                  ;
0069 1D03          00194                BTFSS   STATUS, Z               ;
006A 2875          00195                GOTO    DISPLAY                 ;
006B 01B0          00196                CLRF    HRS                     ;
006C 0AB0          00197                INCF    HRS, F                  ;
006D 2875          00198                GOTO    DISPLAY                 ;
                   00199 ;
                   00200         page
006E               00201 INIT_DISPLAY
006E 300C          00202                MOVLW   DISP_ON                 ; Display On, Cursor On
006F 20E3          00203                CALL    SEND_CMD                ; Send This command to the Display Module
0070 3001          00204                MOVLW   CLR_DISP                ; Clear the Display
0071 20E3          00205                CALL    SEND_CMD                ; Send This command to the Display Module
0072 3006          00206                MOVLW   ENTRY_INC               ; Set Entry Mode Inc., No shift
0073 20E3          00207                CALL    SEND_CMD                ; Send This command to the Display Module
0074 0008          00208                RETURN
                   00209 ;
0075               00210 DISPLAY
0075 3080          00211                MOVLW   DD_RAM_ADDR             ;
0076 20E3          00212                CALL    SEND_CMD                ;
                   00213 ;
0077 1A20          00214                BTFSC   FLAG_REG, KEY_INPUT     ; Do we need to flash the selected units?
0078 287D          00215                GOTO    FLASH_UNITS             ; YES, we need to flash selected units
0079 20A4          00216                CALL    LOAD_HRS                ; NO, do a normal display
007A 20AD          00217                CALL    LOAD_COLON              ;
007B 20B2          00218                CALL    LOAD_MIN                ;
007C 28BB          00219                GOTO    LOAD_AM                 ;
                   00220 ;
```

```
007D                00221 FLASH_UNITS
007D 018A           00222                 CLRF    PCLATH                  ; This clears PCLATH, This table in 1st
007E 0820           00223                 MOVF    FLAG_REG, W             ;    256 bytes of program memory
007F 3903           00224                 ANDLW   0x03                    ; only HR_UNIT and MIN_UNIT bit can be non-zero
0080                00225 UNIT_TBL
0080 0782           00226                 ADDWF   PCL, F                  ; HR_UNIT:MIN_UNIT
0081 289F           00227                 GOTO    NO_UNITS                ;       0 0        - Display everything.
0082 2887           00228                 GOTO    HR_UNITS                ;       0 1        - Flash the hour units
0083 2893           00229                 GOTO    MIN_UNITS               ;       1 0        - Flash the minute units
0084                00230 UNIT_TBL_END
0084 30FC           00231                 MOVLW   0xFC                    ;       1 1        - Need to clear FLAG_REG<HR_UNIT:MIN_UN
                          IT>
0085 05A0           00232                 ANDWF   FLAG_REG, F             ;
0086 289F           00233                 GOTO    NO_UNITS                ;       0 0        - Display everything.
                    00234 ;
                    00235         if ( (UNIT_TBL & 0x0FF) >= (UNIT_TBL_END & 0x0FF) )
                    00236        MESSG   "Warning: Table UNIT_TBL crosses page boundry in computed jump"
                    00237     endif
                    00238 ;
                    00239 ;
0087                00240 HR_UNITS
0087 1C40           00241                 BTFSS   WAIT_CNTR, 0            ; If WAIT_CNTR is odd,
                    00242                                                 ;    hour digits are displayed as blank
0088 288D           00243                 GOTO    SKIP_BLK_HRS            ;
0089 3020           00244                 MOVLW   ' '                     ;
008A 20D4           00245                 CALL    SEND_CHAR               ;
008B 3020           00246                 MOVLW   ' '                     ;
008C 20D4           00247                 CALL    SEND_CHAR               ;
008D                00248 SKIP_BLK_HRS
008D 1C40           00249                 BTFSS   WAIT_CNTR, 0            ; WAIT_CNTR was even, display hour digits
008E 20A4           00250                 CALL    LOAD_HRS                ;
                    00251 ;
008F 303A           00252                 MOVLW   ':'                     ; : always on, display all other character
0090 20D4           00253                 CALL    SEND_CHAR               ;
0091 20B2           00254                 CALL    LOAD_MIN                ;
0092 28BB           00255                 GOTO    LOAD_AM                 ;
                    00256 ;
                    00257         page
0093                00258 MIN_UNITS
0093 20A4           00259                 CALL    LOAD_HRS                ; Display hours
0094 303A           00260                 MOVLW   ':'                     ; : always on
0095 20D4           00261                 CALL    SEND_CHAR               ;
0096 1C40           00262                 BTFSS   WAIT_CNTR, 0            ; If WAIT_CNTR is odd,
                    00263                                                 ;    minute digits are displayed as blank
0097 289C           00264                 GOTO    SKIP_BLK_MIN            ;
0098 3020           00265                 MOVLW   ' '                     ;
0099 20D4           00266                 CALL    SEND_CHAR               ;
```

```
009A 3020           00267                MOVLW   ' '                 ;
009B 20D4           00268                CALL    SEND_CHAR           ;
009C                00269 SKIP_BLK_MIN
009C 1C40           00270                BTFSS   WAIT_CNTR, 0        ; WAIT_CNTR was even, display minute digits
009D 20B2           00271                CALL    LOAD_MIN            ;
009E 28BB           00272                GOTO    LOAD_AM             ;
                    00273 ;
009F                00274 NO_UNITS
009F 20A4           00275                CALL    LOAD_HRS            ; Display all character
00A0 303A           00276                MOVLW   ':'                 ;
00A1 20D4           00277                CALL    SEND_CHAR           ;
00A2 20B2           00278                CALL    LOAD_MIN            ;
00A3 28BB           00279                GOTO    LOAD_AM             ;
                    00280 ;
00A4                00281 LOAD_HRS
00A4 0830           00282                MOVF    HRS, W              ; Load the Wreg with the value
00A5 20C7           00283                CALL    BIN_2_BCD           ;   to convert to BCD
00A6 0833           00284                MOVF    MSD, W              ; Load the MSD value into the Wreg
00A7 2400           00285                CALL    NUM_TABLE           ; Get the ASCII code
00A8 20D4           00286                CALL    SEND_CHAR           ; Send this Character to the Display
                    00287 ;
00A9 0834           00288                MOVF    LSD, W              ; Load the LSD value into the Wreg
00AA 2400           00289                CALL    NUM_TABLE           ; Get the ASCII code
00AB 20D4           00290                CALL    SEND_CHAR           ; Send this Character to the Display
00AC 0008           00291                RETURN
                    00292 ;
00AD 3020           00293 LOAD_COLON     MOVLW   ' '                 ; ASCII value for a Blank space
00AE 1832           00294                BTFSC   SECS, 0             ; Is it an EVEN or ODD second
00AF 3E1A           00295                ADDLW   ':' - ' '           ; Is ODD, Second colon is ON.
                    00296                                            ;   Add delta offset of ASCII Characters
00B0 20D4           00297                CALL    SEND_CHAR           ; Send this Character to the Display
00B1 0008           00298                RETURN
                    00299 ;
00B2                00300 LOAD_MIN
00B2 0831           00301                MOVF    MIN, W              ; Load the Wreg with the value
00B3 20C7           00302                CALL    BIN_2_BCD           ;   to convert to BCD
00B4 0833           00303                MOVF    MSD, W              ; Load the MSD value into the Wreg
00B5 2400           00304                CALL    NUM_TABLE           ; Get the ASCII code
00B6 20D4           00305                CALL    SEND_CHAR           ; Send this Character to the Display
                    00306 ;
00B7 0834           00307                MOVF    LSD, W              ; Load the LSD value into the Wreg
00B8 2400           00308                CALL    NUM_TABLE           ; Get the ASCII code
00B9 20D4           00309                CALL    SEND_CHAR           ; Send this Character to the Display
00BA 0008           00310                RETURN
                    00311 ;
                    00312         page
00BB 3020           00313 LOAD_AM        MOVLW   ' '                 ; ASCII value for a Blank space
```

```
00BC 20D4          00314                CALL    SEND_CHAR              ; Send this Character to the Display
00BD 3041          00315                MOVLW   'A'                    ; ASCII value for a Blank space
00BE 1FA0          00316                BTFSS   FLAG_REG, AM           ; Is it AM or PM
00BF 3E0F          00317                ADDLW   'P' - 'A'              ; Is PM, Add delta offset of ASCII Characters
00C0 20D4          00318                CALL    SEND_CHAR              ; Send this Character to the Display
00C1 304D          00319                MOVLW   'M'
00C2 20D4          00320                CALL    SEND_CHAR              ; Send this Character to the Display
                   00321 ;
00C3 1683          00322                BSF     STATUS, RP0            ; Bank 1
00C4 1381          00323                BCF     OPTION_REG, NOT_RBPU           ; Turn on PORTB Pull-up
00C5 1283          00324                BCF     STATUS, RP0            ; Bank 0
00C6 2823          00325                GOTO    CLR_RB                 ; You've displayed the time, Clear RBIF
                   00326 ;
                   00327 ;
                   00328 ;*************************************************************************
                   00329 ; The BIN_2_BCD routine converts the binary number, in the W register, to a
                   00330 ; binary coded decimal (BCD) munber. This BCD number is stored MSD:LSD. This
                   00331 ; routine is used by the DISPLAY subroutine, to convert the time values.
                   00332 ;*************************************************************************
                   00333 ;
00C7 01B3          00334 BIN_2_BCD      CLRF    MSD                    ; This value contain the 10's digit value
00C8 00B4          00335                MOVWF   LSD                    ; This value contain the 1's digit value
00C9 300A          00336 TENS_SUB       MOVLW   .10                    ; A decimal 10
00CA 0234          00337                SUBWF   LSD, W                 ;
00CB 1C03          00338                BTFSS   STATUS, C              ; Did this subtract cause a Negative Result?
00CC 3400          00339                RETLW   0                      ; YES, Return from this Routine
00CD 00B4          00340                MOVWF   LSD                    ; No, move the result into LSD
00CE 0AB3          00341                INCF    MSD, F                 ; Increment the most significat digit
00CF 28C9          00342                GOTO    TENS_SUB               ;
                   00343 ;
                   00344 ;
                   00345 ; Should NEVER get here
                   00346 ;
00D0 1283          00347 ERROR1         BCF     STATUS, RP0        ; Bank 0
                   00348 ;
                   00349           if ( Debug )
                   00350               BSF     PORTD, 1
                   00351               BCF     PORTD, 1
                   00352           else
00D1 1407          00353               BSF     PORTC, 0
00D2 1007          00354               BCF     PORTC, 0
                   00355           endif
00D3 28D0          00356               GOTO    ERROR1
                   00357 ;
                   00358           page
                   00359 ;
                   00360 ;*****************************************************************
```

```
                    00361 ;* SendChar - Sends character to LCD                          *
                    00362 ;* This routine splits the character into the upper and lower  *
                    00363 ;* nibbles and sends them to the LCD, upper nibble first.      *
                    00364 ;* The data is transmitted on the PORT<3:0> pins               *
                    00365 ;*********************************************************************
                    00366
00D4                00367 SEND_CHAR
00D4 00B6           00368             MOVWF   CHAR                ; Character to be sent is in W
00D5 20F2           00369             CALL    BUSY_CHECK          ; Wait for LCD to be ready
00D6 0E36           00370             SWAPF   CHAR, W
00D7 390F           00371             ANDLW   0x0F                ; Get upper nibble
00D8 0086           00372             MOVWF   LCD_DATA            ; Send data to LCD
00D9 1085           00373             BCF     LCD_CNTL, RW        ; Set LCD to read
00DA 1505           00374             BSF     LCD_CNTL, RS        ; Set LCD to data mode
00DB 1405           00375             BSF     LCD_CNTL, E         ; toggle E for LCD
00DC 1005           00376             BCF     LCD_CNTL, E
00DD 0836           00377             MOVF    CHAR, W
00DE 390F           00378             ANDLW   0x0F                ; Get lower nibble
00DF 0086           00379             MOVWF   LCD_DATA            ; Send data to LCD
00E0 1405           00380             BSF     LCD_CNTL, E         ; toggle E for LCD
00E1 1005           00381             BCF     LCD_CNTL, E
00E2 0008           00382             RETURN
                    00383
                    00384 ;*********************************************************************
                    00385 ;* SendCmd - Sends command to LCD                                 *
                    00386 ;* This routine splits the command into the upper and lower       *
                    00387 ;* nibbles and sends them to the LCD, upper nibble first.         *
                    00388 ;* The data is transmitted on the PORT<3:0> pins                  *
                    00389 ;*********************************************************************
                    00390
00E3                00391 SEND_CMD
00E3 00B6           00392             MOVWF   CHAR                ; Character to be sent is in W
00E4 20F2           00393             CALL    BUSY_CHECK          ; Wait for LCD to be ready
00E5 0E36           00394             SWAPF   CHAR, W
00E6 390F           00395             ANDLW   0x0F                ; Get upper nibble
00E7 0086           00396             MOVWF   LCD_DATA            ; Send data to LCD
00E8 1085           00397             BCF     LCD_CNTL, RW        ; Set LCD to read
00E9 1105           00398             BCF     LCD_CNTL, RS        ; Set LCD to command mode
00EA 1405           00399             BSF     LCD_CNTL, E         ; toggle E for LCD
00EB 1005           00400             BCF     LCD_CNTL, E
00EC 0836           00401             MOVF    CHAR, W
00ED 390F           00402             ANDLW   0x0F                ; Get lower nibble
00EE 0086           00403             MOVWF   LCD_DATA            ; Send data to LCD
00EF 1405           00404             BSF     LCD_CNTL, E         ; toggle E for LCD
00F0 1005           00405             BCF     LCD_CNTL, E
00F1 0008           00406             RETURN
                    00407     page
```

```
                    00408 ;*******************************************************************
                    00409 ;* This routine checks the busy flag, returns when not busy        *
                    00410 ;*  Affects:                                                       *
                    00411 ;*      TEMP - Returned with busy/address                          *
                    00412 ;*******************************************************************
                    00413
00F2                00414 BUSY_CHECK
                    00415 ;
                    00416         if ( Debug )
                    00417             BSF     PORTD, 3
                    00418             BCF     PORTD, 3
                    00419         endif
00F2 0186           00420             CLRF    LCD_DATA            ;** Have PORTB<3:0> output low
00F3 1683           00421             BSF     STATUS, RP0         ; Bank 1
00F4 1781           00422             BSF     OPTION_REG, NOT_RBPU      ; Turn off PORTB Pull-up
00F5 30FF           00423             MOVLW   0xFF                ; Set PortB for input
00F6 0086           00424             MOVWF   LCD_DATA_TRIS
00F7 1283           00425             BCF     STATUS, RP0         ; Bank 0
00F8 1105           00426             BCF     LCD_CNTL, RS        ; Set LCD for Command mode
00F9 1485           00427             BSF     LCD_CNTL, RW        ; Setup to read busy flag
00FA 1405           00428             BSF     LCD_CNTL, E         ; Set E high
00FB 1005           00429             BCF     LCD_CNTL, E         ; Set E low
00FC 0E06           00430             SWAPF   LCD_DATA, W         ; Read upper nibble busy flag, DDRam address
00FD 39F0           00431             ANDLW   0xF0                ; Mask out lower nibble
00FE 00B5           00432             MOVWF   TEMP                ;
00FF 1405           00433             BSF     LCD_CNTL, E         ; Toggle E to get lower nibble
0100 1005           00434             BCF     LCD_CNTL, E
0101 0806           00435             MOVF    LCD_DATA, W         ; Read lower nibble busy flag, DDRam address
0102 390F           00436             ANDLW   0x0F                ; Mask out upper nibble
0103 04B5           00437             IORWF   TEMP, F             ; Combine nibbles
0104 1BB5           00438             BTFSC   TEMP, 7             ; Check busy flag, high = busy
0105 28F2           00439             GOTO    BUSY_CHECK          ; If busy, check again
0106 1085           00440             BCF     LCD_CNTL, RW
0107 1683           00441             BSF     STATUS, RP0         ; Bank 1
0108 30F0           00442             MOVLW   0xF0                ;
0109 0086           00443             MOVWF   LCD_DATA_TRIS       ; RB7 - 4 = inputs, RB3 - 0 = output
010A 1283           00444             BCF     STATUS, RP0         ; Bank 0
010B 0008           00445             RETURN
                    00446 ;
                    00447     page
                    00448 ;
                    00449 ;*******************************************************************************
                    00450 ;*****      Start program here, Power-On Reset occurred.
                    00451 ;*******************************************************************************
                    00452 ;
010C                00453 START                               ; POWER_ON Reset (Beginning of program)
010C 1283           00454             BCF     STATUS, RP0     ; Bank 0
```

```
010D 300C          00455             MOVLW   0x0C            ; Decimal 12
010E 00B0          00456             MOVWF   HRS             ; HOURS = 12
010F 01B1          00457             CLRF    MIN             ; MIN   = 00
0110 3000          00458             MOVLW   0x00            ;
0111 00A0          00459             MOVWF   FLAG_REG        ; PM light is on
0112 3004          00460             MOVLW   0x04            ; Initial value of seconds (64d - 60d)
0113 00B2          00461             MOVWF   SECS            ; This allows a simple bit test to see if 60
                   00462                                     ;   secs has elapsed.
0114 3080          00463             MOVLW   0x80            ; TIM1H:TMR1L = 0x8000 gives 1 second
0115 008F          00464             MOVWF   TMR1H           ;   overflow, at 32 KHz.
0116 018E          00465             CLRF    TMR1L           ;
                   00466 ;
0117               00467 MCLR_RESET                          ; A Master Clear Reset
0117 0183          00468             CLRF    STATUS          ; Do initialization (Bank 0)
0118 018B          00469             CLRF    INTCON
0119 018C          00470             CLRF    PIR1
011A 1683          00471             BSF     STATUS, RP0     ; Bank 1
011B 3000          00472             MOVLW   0x00            ; The LCD module does not like to work w/ weak pull-ups
011C 0081          00473             MOVWF   OPTION_REG        ;
011D 018C          00474             CLRF    PIE1            ; Disable all peripheral interrupts
011E 30FF          00475             MOVLW   0xFF            ;
011F 009F          00476             MOVWF   ADCON1          ; Port A is Digital (for 16C7x devices).
                   00477 ;
                   00478 ;
0120 1283          00479             BCF     STATUS, RP0     ; Bank 0
0121 0185          00480             CLRF    PORTA           ; ALL PORT output should output Low.
0122 0186          00481             CLRF    PORTB
0123 0187          00482             CLRF    PORTC
0124 0188          00483             CLRF    PORTD
0125 0189          00484             CLRF    PORTE
0126 1010          00485             BCF     T1CON, TMR1ON   ; Timer 1 is NOT incrementing
                   00486 ;
0127 1683          00487             BSF     STATUS, RP0     ; Select Bank 1
0128 0185          00488             CLRF    TRISA           ; RA5 -  0 outputs
0129 30F0          00489             MOVLW   0xF0            ;
012A 0086          00490             MOVWF   TRISB           ; RB7 - 4 inputs, RB3 - 0 outputs
012B 0187          00491             CLRF    TRISC           ; RC Port are outputs
012C 1407          00492             BSF     TRISC, T1OSO    ; RC0 needs to be input for the oscillator to function
012D 0188          00493             CLRF    TRISD           ; RD Port are outputs
012E 0189          00494             CLRF    TRISE           ; RE Port are outputs
012F 140C          00495             BSF     PIE1, TMR1IE    ; Enable TMR1 Interrupt
0130 1381          00496             BCF     OPTION_REG, NOT_RBPU  ; Enable PORTB pull-ups
0131 1283          00497             BCF     STATUS, RP0     ; Select Bank 0
0132 0886          00498             MOVF    PORTB, F        ; Need to clear 1st RBIF, due to
0133 100B          00499             BCF     INTCON, RBIF    ;    set up of PORTB
                   00500 ;
                   00501     page
```

```
                        00502 ;
                        00503 ; Initilize the LCD Display Module
                        00504 ;
0134 0185               00505             CLRF    LCD_CNTL        ; ALL PORT output should output Low.
                        00506
0135                    00507 DISPLAY_INIT
0135 3002               00508             MOVLW   0x02            ; Command for 4-bit interface
0136 0086               00509             MOVWF   LCD_DATA        ;
0137 1405               00510             BSF     LCD_CNTL, E     ;
0138 1005               00511             BCF     LCD_CNTL, E     ;
                        00512 ;
                        00513 ; This routine takes the calculated times that the delay loop needs to
                        00514 ; be executed, based on the LCD_INIT_DELAY EQUate that includes the
                        00515 ; frequency of operation. It uses registers before they are needed to
                        00516 ; store the time.
                        00517 ;
0139 3006               00518 LCD_DELAY   MOVLW   LCD_INIT_DELAY  ;
013A 00B3               00519             MOVWF   MSD             ; Use MSD and LSD Registers to Initialize LCD
013B 01B4               00520             CLRF    LSD             ;
013C 0BB4               00521 LOOP2       DECFSZ  LSD, F          ; Delay time = MSD * ((3 * 256) + 3) * Tcy
013D 293C               00522             GOTO    LOOP2           ;
013E 0BB3               00523             DECFSZ  MSD, F          ;
013F                    00524 END_LCD_DELAY
013F 293C               00525             GOTO    LOOP2           ;
                        00526 ;
                        00527 ; Command sequence for 2 lines of 5x7 characters
                        00528 ;
0140 3002               00529 CMD_SEQ     MOVLW   0X02
0141 0086               00530             MOVWF   LCD_DATA
0142 1405               00531             BSF     LCD_CNTL, E     ;
0143 1005               00532             BCF     LCD_CNTL, E     ;
0144 3008               00533             MOVLW   0x08            ;
0145 0086               00534             MOVWF   LCD_DATA        ;
0146 1405               00535             BSF     LCD_CNTL, E     ;
0147 1005               00536             BCF     LCD_CNTL, E     ;
                        00537 ;
                        00538 ; Busy Flag should be valid after this point
                        00539 ;
0148 300C               00540             MOVLW   DISP_ON         ;
0149 20E3               00541             CALL    SEND_CMD        ;
014A 3001               00542             MOVLW   CLR_DISP        ;
014B 20E3               00543             CALL    SEND_CMD        ;
014C 3006               00544             MOVLW   ENTRY_INC       ;
014D 20E3               00545             CALL    SEND_CMD        ;
014E 3080               00546             MOVLW   DD_RAM_ADDR     ;
014F 20E3               00547             CALL    SEND_CMD        ;
                        00548 ;
```

```
                    00549     page
                    00550 ;
                    00551 ; Initialize the Special Function Registers (SFR) interrupts
                    00552 ;
0150 018C           00553             CLRF    PIR1            ;
0151 300E           00554             MOVLW   0x0E
0152 0090           00555             MOVWF   T1CON           ; RC1 is overridden by TCKO
0153 170B           00556             BSF     INTCON, PEIE    ; Enable Peripheral Interrupts
0154 158B           00557             BSF     INTCON, RBIE    ; Disable PORTB<7:4> Change Interrupts
0155 178B           00558             BSF     INTCON, GIE     ; Enable all Interrupts
                    00559 ;
0156 206E           00560             CALL    INIT_DISPLAY    ;
0157 2075           00561             CALL    DISPLAY         ;
                    00562 ;
0158 300E           00563             MOVLW   0x0E
0159 0090           00564             MOVWF   T1CON           ; Enable T1 Oscillator, Ext Clock, Async, prescaler = 1
015A 1410           00565             BSF     T1CON, TMR1ON   ; Turn Timer 1 ON
                    00566 ;
                    00567         if ( PICMaster )
                    00568 lzz         goto    lzz             ; Loop waiting for interrupts (for use with PICMASTER)
                    00569         else
                    00570 ;
015B 0063           00571 SLEEP_LP    SLEEP                   ; Wait for Change on PORTB interrupt. or TMR1 timeout
015C 0000           00572             NOP                     ;
015D 295B           00573             GOTO    SLEEP_LP        ;
                    00574 ;
                    00575         endif
                    00576 ;
                    00577 ; Here is where you do things depending on the type of RESET (Not a Power-On Reset).
                    00578 ;
015E 1E03           00579 OTHER_RESET   BTFSS  STATUS,NOT_TO  ; WDT Time-out?
015F 28D0           00580 WDT_TIMEOUT   GOTO   ERROR1         ; YES, This is error condition
                    00581         if ( Debug_PU )
0160 290C           00582             goto   START            ; MCLR reset, Goto START
                    00583         else
                    00584             GOTO   MCLR_RESET       ; MCLR reset, Goto MCLR_RESET
                    00585         endif
                    00586 ;
                    00587         if (Debug )
                    00588 END_START     NOP                   ; END label for debug
                    00589         endif
                    00590 ;
                    00591     page
                    00592 ;
0400                00593     org     TABLE_ADDR
                    00594 ;
0400 00B5           00595 NUM_TABLE       MOVWF   TEMP        ; Store value to TEMP register
```

```
0401 3004           00596                MOVLW   HIGH (TABLE_ADDR)   ; Ensure that PCLATH high has the
0402 008A           00597                MOVWF   PCLATH              ;   correct value
0403 0835           00598                MOVF    TEMP, W             ; Value into table
0404 390F           00599                ANDLW   0x0F                ; Mask to 4-bits (00 - 0Fh)
0405 0782           00600 NUM_TBL        ADDWF   PCL, F              ; Determine Offset into table
0406 3430           00601                RETLW   '0'                 ; ASCII value of "0" in W register
0407 3431           00602                RETLW   '1'                 ; ASCII value of "1" in W register
0408 3432           00603                RETLW   '2'                 ; ASCII value of "2" in W register
0409 3433           00604                RETLW   '3'                 ; ASCII value of "3" in W register
040A 3434           00605                RETLW   '4'                 ; ASCII value of "4" in W register
040B 3435           00606                RETLW   '5'                 ; ASCII value of "5" in W register
040C 3436           00607                RETLW   '6'                 ; ASCII value of "6" in W register
040D 3437           00608                RETLW   '7'                 ; ASCII value of "7" in W register
040E 3438           00609                RETLW   '8'                 ; ASCII value of "8" in W register
040F 3439           00610                RETLW   '9'                 ; ASCII value of "9" in W register
                    00611                                            ; Any enter after is in error (Display an E)
0410 3445           00612                RETLW   'E'                 ; ASCII value of "E" in W register
0411 3445           00613                RETLW   'E'                 ; ASCII value of "E" in W register
0412 3445           00614                RETLW   'E'                 ; ASCII value of "E" in W register
0413 3445           00615                RETLW   'E'                 ; ASCII value of "E" in W register
0414 3445           00616                RETLW   'E'                 ; ASCII value of "E" in W register
0415 3445           00617 NUM_TBL_END    RETLW   'E'                 ; ASCII value of "E" in W register
                    00618 ;
                    00619     if ( (NUM_TBL & 0xFF00) != (NUM_TBL_END & 0xFF00) )
                    00620        MESSG   "Warning: Table NUM_TBL crosses page boundry in computed jump"
                    00621     endif
                    00622 ;
                    00623 ;
07FF                00624     org     PMEM_END                       ; End of Program Memory
07FF 28D0           00625             GOTO    ERROR1                 ; If you get here your program was lost
                    00626
                    00627     end



MEMORY USAGE MAP ('X' = Used,  '-' = Unused)


0000 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0040 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0080 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
00C0 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0100 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0140 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX X--------------- ----------------
0400 : XXXXXXXXXXXXXXXX XXXXXX---------- ---------------- ----------------
07C0 : ---------------- ---------------- ---------------- ---------------X
```

```
All other memory blocks unused.

Program Memory Words Used:   376
Program Memory Words Free:  3720


Errors   :     0
Warnings :     0 reported,     0 suppressed
Messages :     0 reported,    16 suppressed
```

Please check the Microchip BBS for the latest version of the source code. Microchip's Worldwide Web Address: www.microchip.com; Bulletin Board Support: MCHIPBBS using CompuServe® (CompuServe membership not required).

APPENDIX B: CLOCK_01.H INCLUDE FILE

```
   nolist
;*************************************************************************
;
;  This is the custom Header File for the real time clock application note
;  PROGRAM: CLOCK.H
;  Revision:5-10-94
;
;*************************************************************************
; This is used for the ASSEMBLER to recalculate certain frequency
; dependant variables. The value of Dev_Freq must be changed to
; reflect the frequency that the device actually operates at.
;
Dev_Freq         EQU   D'4000000'     ; Device Frequency is 4 MHz
DB_HI_BYTE       EQU  (HIGH ((( Dev_Freq / 4 ) * 1 / D'1000' ) / 3 ) ) + 1
LCD_INIT_DELAY   EQU  (HIGH ((( Dev_Freq / 4 ) * D'46' / D'10000' ) / 3 ) ) + 1
INNER_CNTR       EQU  40  RAM Location
OUTER_CNTR       EQU  41              ; RAM Location
;
T1OSO            EQU 0                ; The RC0 / T1OSO / T1CKI
;
RESET_V          EQU  0x0000   ; Address of RESET Vector
ISR_V            EQU  0x0004   ; Address of Interrupt Vector
PMEM_END         EQU  0x07FF   ; Last address in Program Memory
TABLE_ADDR       EQU  0x0400   ; Address where to start Tables
;
HR_MIN_SW        EQU  0x7      ; The switch to select the units
INC_SW           EQU  0x6      ; The switch to increment the selected units
CLR_MIN_SW       EQU  0x5      ; The switch to clear the minutes and seconds
;
FLAG_REG         EQU  0x020    ; Register which contains flag bits
;
;  +------+-------+-------+-----------+-------+-------+----------+---------+
;  |  AM  |  ---  |  ---  | KEY_INPUT |  ---  |  ---  | MIN_UNIT | HR_UNIT |
;  +------+-------+-------+-----------+-------+-------+----------+---------+
;
AM               EQU  0x07    ; Flag to specify if AM or PM
;
KEY_INPUT        EQU  0x04    ; Flag to specify if doing key inputs
;
MIN_UNIT         EQU  0x01    ; Flags to specify which units to operate on
HR_UNIT          EQU  0x00    ;  (HRS, MIN, or none)
;
HRS              EQU  0x030   ; Holds counter value for HOURS
MIN              EQU  0x031   ; Holds counter value for MINUTES
SECS             EQU  0x032   ; Holds counter value for SECONDS
MSD              EQU  0x033   ; Temporary register, Holds MSD of BIN to BCD conversion
LSD              EQU  0x034   ; Temporary register, Holds LSD of BIN to BCD conversion
TEMP             EQU  0x035   ; Temporary register
CHAR             EQU  0x036   ; Temporary register, Holds value to send to LCD module.
;
WAIT_CNTR        EQU  0x040   ; Counter that holds wait time for key inputs
;
;
; LCD Display Commands and Control Signal names.
;
E                EQU  0       ; LCD Enable control line
R_W              EQU  1       ; LCD Read/Write control line
RS               EQU  2       ; LCD Register Select control line
;
;
; LCD Module commands
;
```

```
DISP_ON          EQU  0x00C   ; Display on
DISP_ON_C        EQU  0x00E   ; Display on, Cursor on
DISP_ON_B        EQU  0x00F   ; Display on, Cursor on, Blink cursor
DISP_OFF         EQU  0x008   ; Display off
CLR_DISP         EQU  0x001   ; Clear the Display
ENTRY_INC        EQU  0x006   ;
ENTRY_INC_S      EQU  0x007   ;
ENTRY_DEC        EQU  0x004   ;
ENTRY_DEC_S      EQU  0x005   ;
DD_RAM_ADDR      EQU  0x080   ; Least Significant 7-bit are for address
DD_RAM_UL        EQU  0x080   ; Upper Left corner of the Display
;

    list
```

Please check the Microchip BBS for the latest version of the source code. Microchip's Worldwide Web Address: www.microchip.com; Bulletin Board Support: MCHIPBBS using CompuServe® (CompuServe membership not required).

APPENDIX C: C74_REG.H INCLUDE FILE

```
    NOLIST
;
;   File =   C64_reg.h
;   Rev. History:     08-04-93 by MP
;                     10-18-93 by MP to make Page ok
;                     11-15-93 by MP to have correct pages for SFR
;
; EQUates for Special Function Registers
;
;
INDF            EQU     00
TMR0            EQU     01
OPTION_R        EQU     81
PCL             EQU     02
STATUS          EQU     03
FSR             EQU     04
PORTA           EQU     05
TRISA           EQU     85
PORTB           EQU     06
TRISB           EQU     86
PORTC           EQU     07
TRISC           EQU     87
PORTD           EQU     08
TRISD           EQU     88
PORTE           EQU     09
TRISE           EQU     89
PCLATH          EQU     0A
INTCON          EQU     0B
PIR1            EQU     0C
PIE1            EQU     8C
TMR1L           EQU     0E
PCON            EQU     8E
TMR1H           EQU     0F
T1CON           EQU     10
TMR2            EQU     11
T2CON           EQU     12
PR2             EQU     92
SSPBUF          EQU     13
SSPADD          EQU     93
SSPCON          EQU     14
SSPSTAT         EQU     94
CCPR1L          EQU     15
CCPR1H          EQU     16
CCP1CON         EQU     17
RCSTA           EQU     18
TXSTA           EQU     98
TXREG           EQU     19
SPBRG           EQU     99
RCREG           EQU     1A
CCPR2L          EQU     1B
CCPR2H          EQU     1C
CCP2CON         EQU     1D
ADRES           EQU     1E
ADCON0          EQU     1F
ADCON1          EQU     9F
;
;*********************************************
;************   Bit Definitions   ************
```

```
;*******************************************
;
; STATUS register (Address 03/83)
;
IRP             EQU     7
RP1             EQU     6
RP0             EQU     5
TO              EQU     4
PD              EQU     3
Z               EQU     2
DC              EQU     1
C               EQU     0
;
; INTCON register (Address 0B/8B)
;
GI              EQU     7
PEIE            EQU     6
T0IE            EQU     5
INTE            EQU     4
RBIE            EQU     3
T0IF            EQU     2
INTF            EQU     1
RBIF            EQU     0
;
; PIR1 register (Address 0C)
;
PSPIF           EQU     7
SSPIF           EQU     3
CCP1IF          EQU     2
TMR2IF          EQU     1
TMR1IF          EQU     0
;
; PIE1 register (Address 8C)
;
PSPIE           EQU     7
SSPIE           EQU     3
CCP1IE          EQU     2
TMR2IE          EQU     1
TMR1IE          EQU     0
;
; OPTION register (Address 81)
;
RBPU            EQU     7
INTEDG          EQU     6
T0CS            EQU     5
T0SE            EQU     4
PSA             EQU     3
PS2             EQU     2
PS1             EQU     1
PS0             EQU     0
;
; PCON register (Address 8E)
;
POR             EQU     1
;
; TRISE register (Address 89)
;
IBF             EQU     7
OBF             EQU     6
IBOV            EQU     5
PSPMODE         EQU     4
TRISE2          EQU     2
TRISE1          EQU     1
TRISE0          EQU     0
;
; T1CON register (Address 10)
```

```
;
T1CKPS1         EQU      5
T1CKPS0         EQU      4
T1OSCEN         EQU      3
T1INSYNC        EQU      2
TMR1CS          EQU      1
TMR1ON          EQU     0
;
; T2CON register (Address 12)
;
TOUTPS3         EQU     6
TOUTPS2         EQU     5
TOUTPS1         EQU     4
TOUTPS0         EQU     3
TMR2ON          EQU     2
T2CKPS1         EQU     1
T2CKPS0         EQU     0
;
; SSPCON register (Address 14)
;
WCOL            EQU     7
SSPOV           EQU     6
SSPEN           EQU     5
CKP             EQU     4
SSPM3           EQU     3
SSPM2           EQU     2
SSPM1           EQU     1
SSPM0           EQU     0
;
; SSPSTAT register (Address 94)
;
DA              EQU     5
P               EQU     4
S               EQU     3
RW              EQU     2
UA              EQU     1
BF              EQU     0
;
; CCP1CON register (Address 17)
;
CCP1X           EQU     5
CCP1Y           EQU     4
CCP1M3          EQU     3
CCP1M2          EQU     2
CCP1M1          EQU     1
CCP1M0          EQU     0
;
; RCSTA register (Address 18)
;
SPEN           EQU      7
RC89           EQU      6
SREN           EQU      5
CREN           EQU      4
FERR           EQU      2
OERR           EQU      1
RCD8           EQU      0
;
; TXSTA register (Address 98)
;
CSRC           EQU      7
TX89           EQU      6
TXEN           EQU      5
SYNC           EQU      4
BRGH           EQU      2
TRMT           EQU      1
TXD8           EQU      0
```

```
;
; CCP2CON register (Address 1D)
;
CCP2X         EQU      5
CCP2Y         EQU      4
CCP2M3        EQU      3
CCP2M2        EQU      2
CCP2M1        EQU      1
CCP2M0        EQU      0
;
; ADCON0 register (Address 1F)
;
ADCS1         EQU      7
ADCS0         EQU      6
CHS2          EQU      5
CHS1          EQU      4
CHS0          EQU      3
GO            EQU      2
DONE          EQU      2
ADON          EQU      0
;
; ADCON1 register (Address 9F)
;
PCFG2         EQU      2
PCFG1         EQU      1
PCFG0         EQU      0
;
;*******************************************
;**** Bits for destination control
;****    W = W register is destination
;****    F = File register is destination
;*******************************************
;
W             EQU      0
F             EQU      1
;
FALSE         EQU      0
TRUE          EQU      1


    LIST
```