Low-Power Real Time Clock

INTRODUCTION

This application note implements a low-power real time clock using the Timer1 module of the PIC16CXX family of processors. Timer1 can operate from its own crystal source, which allows the timer to increment while the device is in sleep mode. The device is placed into sleep to minimize the current consumption. Only the events that require processing will wake the device from sleep. These are a key input and a Timer1 overflow.

OPERATION

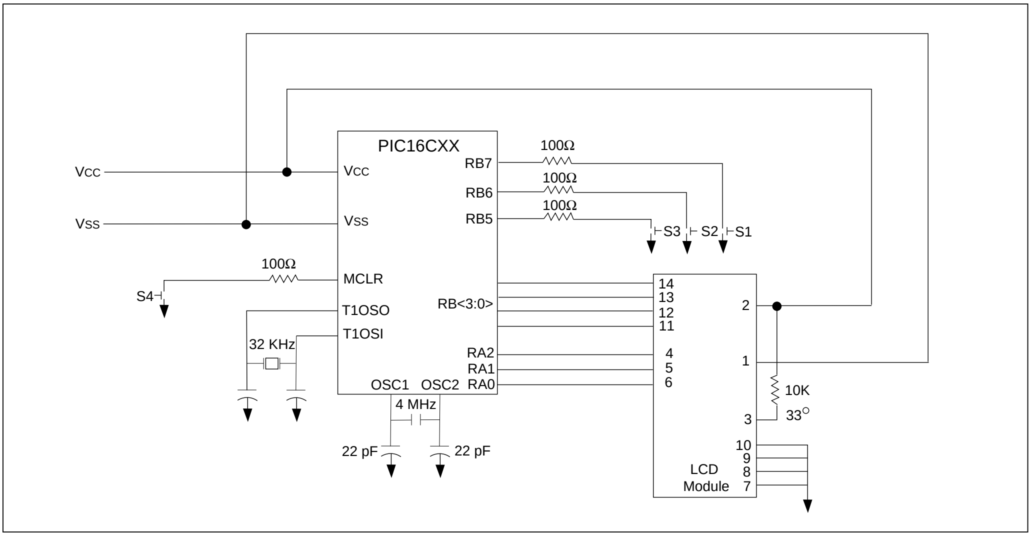
Upon power-up, the device goes into an initial state. This state sets the display to 12:00 PM and waits for Timer1 to generate an interrupt (every second). The Timer1 overflow interrupt wakes the device from sleep This causes the time registers (HRS, MIN, SECS) to be updated. If the SECS register contains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

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 are displayed. After a key has not been depressed for more then one second, the selected units will begin to flash. 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. After no key has been depressed for ten seconds, the unit returns to the clock mode.

To simplify the design time, a standard Hitachi LCD display module is used. In most applications requiring a LCD display, a custom LCD display is used. 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. RA<2:0> are the control signals to the LCD display, RB<3:0> is the 4-bit data bus, and RB<7:5> is the input switches. The OSC1 pin is connected to an RC network, which generates approximately a 4 MHz device frequency. The device frequency does not need to be stable, since the Timer1 module operates asynchronously. This allows the device's oscillaor to be configures for RC mode. This oscillator mode is the least expensive and has the quickest start-up time. Timer1 is where the accurate frequency is required. This 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 number.

FIGURE 1: CLOCK BLOCK DIAGRAM



Relative to most microelectronics, the 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 the sleep mode as much as possible.

By using the conditional assemble, 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 uS that the device is awake. This 800 uS operation is out of the 1 second time that the device needs to service the interrupt (a TMR1 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 presented here 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, not 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 hrs. 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 upon the desired display / display driver is recommended.

Alternative 3 uses the LCD module in an 8-bit mode, 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 Characteristics)	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 W.C.	106	35 + Display
		Key Input W.C.		35 + Display Time
Data Memory	Display‡		208	526†
	Variables		5	N.A.
	Scratch RAM		4	N.A.

† Dependent on LCD Module (re: BUSY_CHECK subroutine)

‡ Assumes worst case numbers and best case response from LCD module.

Author: Mark Palmer - Sr. Application Engineer
Logic Products Division

APPENDIX A: SOURCE CODE LISTING (CLOCK_01.LST)

```
MPASM 01.01 Released     CLOCK.ASM   5-13-1994  13:11:9                 PAGE  1


LOC  OBJECT CODE     LINE SOURCE TEXT

                     0001         LIST    P = 16C74,  F = INHX8M, n = 66
                     0002 ;
                     0003 ;******************************************************************************
                     0004 ;
                     0005 ; This program implements a real time clock using the TMR1 module of the
                     0006 ; PIC16Cxx family. A LCD display module is used to display (update) the
                     0007 ; time every second. Three keys are used to set the time.
                     0008 ;
                     0009 ;       Program = CLOCK.ASM
                     0010 ;       Revision Date:   5-15-94
                     0011 ;
                     0012 ;******************************************************************************
                     0013 ;
                     0014 ;
                     0015 ; HARDWARE SETUP
                     0016 ;   LCD Control Lines
                     0017 ;       RA0 = E     (Enable)
                     0018 ;       RA1 = R_W   (Read/Write)
                     0019 ;       RA2 = RS    (Register Select)
                     0020 ;   LCD Data Lines
                     0021 ;       RB<3:0>
                     0022 ;   Switch Inputs
                     0023 ;       RB7 = Select Hour / Minute / Off
                     0024 ;       RB6 = Increment Hour / Minute
                     0025 ;       RB5 = Reset Minutes to 00
                     0026 ;
                     0027             INCLUDE <C74_reg.h>
                     0233
                     0027
                     0028             INCLUDE <CLOCK.h>
                     0028
                     0029 ;
0006                 0030 LCD_DATA        EQU     PORTB       ; The LCD data is on the lower 4-bits
0086                 0031 LCD_DATA_TRIS   EQU     TRISB       ; The TRIS register for the LCD data
0005                 0032 LCD_CNTL        EQU     PORTA       ; Three control lines
                     0033 ;
0000                 0034 PICMaster       EQU     FALSE        ; A Debugging Flag
0000                 0035 Debug           EQU     FALSE        ; A Debugging Flag
```

```
0001                 0036 Debug_PU        EQU     TRUE         ; A Debugging Flag
                     0037 ;
                     0038 ;
                     0039 ; Reset address. Determine type of RESET
                     0040 ;
                     0041         org     RESET_V                 ; RESET vector location
0000 1683            0042 RESET       BSF     STATUS, RP0         ; Bank 1
0001 188E            0043             BTFSC   PCON, POR           ; Power-up reset?
0002 290C            0044             GOTO    START               ; YES
0003 295F            0045             GOTO    OTHER_RESET         ; NO, a WDT or MCLR reset
                     0046 ;
                     0047 ; This is the Periperal Interrupt routine. Need to determine the type
                     0048 ; of interrupt that occurred. The following interrupts are enabled:
                     0049 ;   1.  PORTB Change (RBIF)
                     0050 ;   2.  TMR1 Overflow Interrupt (T1IF)
                     0051 ;
                     0053         org     ISR_V              ; Interrupt vector location
                     0054 PER_INT_V
                     0055         if ( Debug )
                     0056             bsf     PORTD, 0       ; Set high, use to measure total
                     0057         endif                      ;    time in Int Service Routine
                     0058 ;
0004 1283            0059             BCF     STATUS, RP0    ; Bank 0
0005 180C            0060             BTFSC   PIR1, TMR1IF   ; Timer 1 overflowed?
0006 2843            0061             GOTO    T1_OVRFL       ; YES, Service the Timer1 Overflow Interrupt
0007 1C0B            0062             BTFSS   INTCON, RBIF   ; NO, Did PORTB change?
0008 28D0            0063             GOTO    ERROR1         ; NO, Error Condition - Unknown Interrupt
                     0064 ;
                     0065 PORTB_FLAG                         ; Are any of PORTB’s inputs active?
0009 0806            0066             MOVF    PORTB, W       ;
000A 39E0            0067             ANDLW   0xE0           ; Keep only the 3 switch values
000B 00B5            0068 DEBOUNCE    MOVWF   TEMP           ;
000C 3002            0069             MOVLW   DB_HI_BYTE     ; This is the debounce delay
000D 08B3            0070             MOVF    MSD            ;
000E 01B4            0071             CLRF    LSD            ;
000F 0BB4            0072 KB_D_LP1    DECFSZ  LSD            ;
0010 280F            0073             GOTO    KB_D_LP1       ;
0011 0BB3            0074             DECFSZ  MSD            ;
0012 280F            0075             GOTO    KB_D_LP1       ;
0013 0806            0076 END_DELAY   MOVF    PORTB, W       ;
0014 39E0            0077             ANDLW   0xE0           ; Keep only the 3 switch values
0015 02B5            0078             SUBWF   TEMP, F        ;
0016 1D03            0079             BTFSS   STATUS, Z      ; Is the Zero bit set?
                     0080                                    ;   (switches were the same on 2 reads)
0017 280B            0081             GOTO    DEBOUNCE       ; NO, Try another read
0018 00B5            0082 KEY_MATCH   MOVWF   TEMP           ; YES, need to see which is depressed.
                     0083 ;
0019 3080            0084             MOVLW   0x80           ; Since doing key inputs, clear TMR1
```

```
001A 008F                  MOVWF   TMR1H          ;   for 1 sec overflow.
001B 018E                  CLRF    TMR1L          ;
001C 100C                  BCF     PIR1, TMR1IF   ; Clear Timer 1 Interrupt Flag

001D 1FB5                  BTFSS   TEMP, HR_MIN_SW   ; Is the hour-min-off switch depressed?
001E 2826                  GOTO    SELECT_UNITS      ; YES, specify the units selected
001F 1F35                  BTFSS   TEMP, INC_SW      ; Is the inc switch depressed?
0020 282B                  GOTO    INC_UNIT          ; YES, Increment the selected Units
0021 1EB5                  BTFSS   TEMP, CLR_MIN_SW  ; Is the clear minute switch depressed?
0022 2835                  GOTO    CLR_MIN           ; YES, clear the minutes.
                    ;
                    ; No key match occured, or finished with PortB interrupt and need to clear interrupt condition.
                    ;
                    CLR_RB                        ; No RB<7:5> keys are depressed (rising edge Int.)
0023 0886                  MOVF    PORTB, F       ; Clear the PORTB mismatch condition
0024 100B                  BCF     INTCON, RBIF   ; Clear the PORTB Int Flag
                       if ( Debug )
                           bcf     PORTD, 0    ; Set low, use to measure total
                       endif                   ;   time in Int Service Routine
0025 0009                  RETFIE              ; Return / Enable Global Interrupts
                    ;
                    SELECT_UNITS
0026 30FF                  MOVLW   0xFF                 ;
0027 00C0                  MOVWF   WAIT_CNTR            ; WAIT_CNTR has LSb set after each SELECT UNIT key press.
0028 0AA0                  INCF    FLAG_REG, F          ; Increment the pointer to the MIN_UNIT:HR_UNIT
0029 1620                  BSF     FLAG_REG, KEY_INPUT ;
002A 2875                  GOTO    DISPLAY              ; Flash the Display of the selected unit
                    ;
                    INC_UNIT
002B 01C0                  CLRF    WAIT_CNTR            ; WAIT_CNTR is cleared to zero after each key press.
002C 1820                  BTFSC   FLAG_REG, HR_UNIT    ; Are the hour units selected?
002D 285C                  GOTO    INC_HRS              ; YES, Increment the hour units
002E 1CA0                  BTFSS   FLAG_REG, MIN_UNIT  ; Are the minute units selected?
002F 2823                  GOTO    CLR_RB               ; NO, Not a valid key. Clear flags
                    ;
0030 0AB1                  INCF    MIN, F               ; YES, Increment the minute units
0031 303C                  MOVLW   0x3C                 ; This is Decimal 60
0032 0231                  SUBWF   MIN, W               ; MIN - 60 = ?
0033 1D03                  BTFSS   STATUS, Z            ; MIN = 60?
0034 2875                  GOTO    DISPLAY              ; NO, display time
                                                        ; YES, MIN = 0 (use code from CLR_MIN)
0035 01B1           CLR_MIN     CLRF    MIN             ; MIN = 0
0036 3004                  MOVLW   0x04                 ; Clear the seconds
0037 00B2                  MOVWF   SECS                 ; Initial Second count = 4
0038 3080                  MOVLW   0x80                 ; Clear Timer 1, for 1 sec overflow
0039 008F                  MOVWF   TMR1H                ;
003A 018E                  CLRF    TMR1L                ;
003B 100C                  BCF     PIR1, TMR1IF         ; Clear the TMR1 overflow interrupt.
```

```
003C 01C0          0134          CLRF    WAIT_CNTR          ; WAIT_CNTR is cleared to zero after each key press.
003D 1AB5          0135          BTFSC   TEMP, CLR_MIN_SW   ; Is the clear minute switch depressed?
003E 2875          0136          GOTO    DISPLAY            ; NO. Rollover from increment key
003F 10A0          0137          BCF     FLAG_REG, MIN_UNIT ; YES, Clear ALL relevant flags
0040 1020          0138          BCF     FLAG_REG, HR_UNIT  ;
0041 1220          0139          BCF     FLAG_REG, KEY_INPUT ;
0042 2875          0140          GOTO    DISPLAY            ;
                   0141 ;
                   0143 ;
                   0144 T1_OVRFL
0043 100C          0145          BCF     PIR1, TMR1IF         ; Clear Timer 1 Interrupt Flag
0044 1E20          0146          BTFSS   FLAG_REG, KEY_INPUT  ; Are we using the key inputs?
0045 284F          0147          GOTO    INC_TIME             ; NO, Need to Increment the time
0046 0AC0          0148          INCF    WAIT_CNTR, F         ; YES,
0047 300A          0149          MOVLW   0x0A                 ; 10 counts x 1 seconds
0048 0240          0150          SUBWF   WAIT_CNTR, W         ; Has the 10 Sec wait for key expired?
0049 1D03          0151          BTFSS   STATUS, Z            ; Is the result 0?
004A 2875          0152          GOTO    DISPLAY              ; NO, Display value
004B 01C0          0153          CLRF    WAIT_CNTR            ; YES, Clear WAIT_CNTR
004C 1220          0154          BCF     FLAG_REG, KEY_INPUT  ;
004D 1020          0155          BCF     FLAG_REG, HR_UNIT    ;
004E 10A0          0156          BCF     FLAG_REG, MIN_UNIT   ;
                   0157 ;
                   0158 ;
004F 3080          0159 INC_TIME MOVLW   0x80           ;
0050 008F          0160          MOVWF   TMR1H          ; 1 Second Overflow
0051 0AB2          0161          INCF    SECS, F        ;
0052 1F32          0162          BTFSS   SECS, 6        ;
0053 2875          0163          GOTO    DISPLAY        ;
0054 3004          0164          MOVLW   0x04           ;
0055 00B2          0165          MOVWF   SECS           ;
0056 0AB1          0166          INCF    MIN, F         ;
0057 303C          0167          MOVLW   0x3C           ; W = 60d
0058 0231          0168          SUBWF   MIN, W         ;
0059 1D03          0169          BTFSS   STATUS, Z      ;
005A 2875          0170          GOTO    DISPLAY        ;
005B 01B1          0171          CLRF    MIN            ;
005C 0AB0          0172 INC_HRS  INCF    HRS, F         ;
005D 300C          0173          MOVLW   0x0C           ; It is now 12:00, Toggle AM / PM
005E 0230          0174          SUBWF   HRS, W         ;
005F 1D03          0175          BTFSS   STATUS, Z      ;
0060 2867          0176          GOTO    CK_13          ; Need to check if HRS = 13
0061 1FA0          0177          BTFSS   FLAG_REG, AM   ; Was it AM or PM
0062 2865          0178          GOTO    SET_AM         ; Was PM, Needs to be AM
0063 13A0          0179          BCF     FLAG_REG, AM   ; It is PM
0064 2875          0180          GOTO    DISPLAY        ;
0065 17A0          0181 SET_AM   BSF     FLAG_REG, AM   ; It is AM
0066 2875          0182          GOTO    DISPLAY        ;
```

```
                    ;
0067 300D           CK_13        MOVLW   0x0D             ; Check if HRS = 13
0068 0230                        SUBWF   HRS, W           ;
0069 1D03                        BTFSS   STATUS, Z        ;
006A 2875                        GOTO    DISPLAY          ;
006B 01B0                        CLRF    HRS              ;
006C 0AB0                        INCF    HRS, F           ;
006D 2875                        GOTO    DISPLAY          ;
                    ;
                    INIT_DISPLAY
006E 300C                        MOVLW   DISP_ON          ; Display On, Cursor On
006F 20E3                        CALL    SEND_CMD         ; Send This command to the Display Module
0070 3001                        MOVLW   CLR_DISP         ; Clear the Display
0071 20E3                        CALL    SEND_CMD         ; Send This command to the Display Module
0072 3006                        MOVLW   ENTRY_INC        ; Set Entry Mode Inc., No shift
0073 20E3                        CALL    SEND_CMD         ; Send This command to the Display Module
0074 0008                        RETURN
                    ;
                    DISPLAY
0075 3080                        MOVLW   DD_RAM_ADDR      ;
0076 20E3                        CALL    SEND_CMD         ;
                    ;
0077 1A20                        BTFSC   FLAG_REG, KEY_INPUT ; Do we need to flash the selectected units?
0078 287D                        GOTO    FLASH_UNITS         ; YES, we need to flash selected units
0079 20A4                        CALL    LOAD_HRS            ; NO, do a normal display
007A 20AD                        CALL    LOAD_COLON          ;
007B 20B2                        CALL    LOAD_MIN            ;
007C 28BB                        GOTO    LOAD_AM             ;
                    ;
                    FLASH_UNITS
007D 018A                        CLRF    PCLATH              ; This clears PCLATH, This table in 1st
007E 0820                        MOVF    FLAG_REG, W         ;    256 bytes of program memory
007F 3903                        ANDLW   0x03                ; only HR_UNIT and MIN_UNIT bit can be non-zero
                    UNIT_TBL
0080 0782                        ADDWF   PCL                 ; HR_UNIT:MIN_UNIT
0081 289F                        GOTO    NO_UNITS            ;    0 0  - Display everything.
0082 2887                        GOTO    HR_UNITS            ;    0 1  - Flash the hour units
0083 2893                        GOTO    MIN_UNITS           ;    1 0  - Flash the minute units
                    UNIT_TBL_END
0084 30FC                        MOVLW   0xFC                ;    1 1  - Need to clear FLAG_REG
0085 05A0                        ANDWF   FLAG_REG, F         ;                    <HR_UNIT:MIN_UNIT>
0086 289F                        GOTO    NO_UNITS            ;    0 0  - Display everything.
                    ;
                        if ( (UNIT_TBL & 0x0FF) >= (UNIT_TBL_END & 0x0FF) )
                           MESSG   “Warning: Table UNIT_TBL crosses page boundry in computed jump”
                        endif
                    ;
                    ;
                    0232 HR_UNITS
0087 1C40           0233             BTFSS   WAIT_CNTR, 0    ; If WAIT_CNTR is odd,
                    0234                                     ;    hour digits are displayed as blank
0088 288D           0235             GOTO    SKIP_BLK_HRS    ;
0089 3020           0236             MOVLW   ‘ ‘             ;
008A 20D4           0237             CALL    SEND_CHAR       ;
008B 3020           0238             MOVLW   ‘ ‘             ;
008C 20D4           0239             CALL    SEND_CHAR       ;
                    0240 SKIP_BLK_HRS
008D 1C40           0241             BTFSS   WAIT_CNTR, 0    ; WAIT_CNTR was even, display hour digits
008E 20A4           0242             CALL    LOAD_HRS        ;
                    0243 ;
008F 303A           0244             MOVLW   ‘:’             ; : always on, display all other character
0090 20D4           0245             CALL    SEND_CHAR       ;
0091 20B2           0246             CALL    LOAD_MIN        ;
0092 28BB           0247             GOTO    LOAD_AM         ;
                    0248 ;
                    0250 MIN_UNITS
0093 20A4           0251             CALL    LOAD_HRS        ; Display hours
0094 303A           0252             MOVLW   ‘:’             ; : always on
0095 20D4           0253             CALL    SEND_CHAR       ;
0096 1C40           0254             BTFSS   WAIT_CNTR, 0    ; If WAIT_CNTR is odd,
                    0255                                     ;   minute digits are displayed as blank
0097 289C           0256             GOTO    SKIP_BLK_MIN    ;
0098 3020           0257             MOVLW   ‘ ‘             ;
0099 20D4           0258             CALL    SEND_CHAR       ;
009A 3020           0259             MOVLW   ‘ ‘             ;
009B 20D4           0260             CALL    SEND_CHAR       ;
                    0261 SKIP_BLK_MIN
009C 1C40           0262             BTFSS   WAIT_CNTR, 0    ; WAIT_CNTR was even, display minute digits
009D 20B2           0263             CALL    LOAD_MIN        ;
009E 28BB           0264             GOTO    LOAD_AM         ;
                    0265 ;
                    0266 NO_UNITS
009F 20A4           0267             CALL    LOAD_HRS        ; Display all character
00A0 303A           0268             MOVLW   ‘:’             ;
00A1 20D4           0269             CALL    SEND_CHAR       ;
00A2 20B2           0270             CALL    LOAD_MIN        ;
00A3 28BB           0271             GOTO    LOAD_AM         ;
                    0272 ;
                    0273 LOAD_HRS
00A4 0830           0274             MOVF    HRS, W          ; Load the Wreg with the value
00A5 20C7           0275             CALL    BIN_2_BCD       ;   to convert to BCD
00A6 0833           0276             MOVF    MSD, W          ; Load the MSD value into the Wreg
00A7 2400           0277             CALL    NUM_TABLE       ; Get the ASCII code
00A8 20D4           0278             CALL    SEND_CHAR       ; Send this Character to the Display
                    0279 ;
00A9 0834           0280             MOVF    LSD, W          ; Load the LSD value into the Wreg
```

```
00AA 2400               CALL    NUM_TABLE       ; Get the ASCII code
00AB 20D4               CALL    SEND_CHAR       ; Send this Character to the Display
00AC 0008               RETURN
           ;
00AD 3020  LOAD_COLON   MOVLW   ‘ ‘             ; ASCII value for a Blank space
00AE 1832               BTFSC   SECS, 0         ; Is it an EVEN or ODD second
00AF 3E1A               ADDLW   ‘:’ - ‘ ‘       ; Is ODD, Second colon is ON.
                                                ;   Add delta offset of ASCII Characters
00B0 20D4               CALL    SEND_CHAR       ; Send this Character to the Display
00B1 0008               RETURN
           ;
           LOAD_MIN
00B2 0831               MOVF    MIN, W          ; Load the Wreg with the value
00B3 20C7               CALL    BIN_2_BCD       ;   to convert to BCD
00B4 0833               MOVF    MSD, W          ; Load the MSD value into the Wreg
00B5 2400               CALL    NUM_TABLE       ; Get the ASCII code
00B6 20D4               CALL    SEND_CHAR       ; Send this Character to the Display
           ;
00B7 0834               MOVF    LSD, W          ; Load the LSD value into the Wreg
00B8 2400               CALL    NUM_TABLE       ; Get the ASCII code
00B9 20D4               CALL    SEND_CHAR       ; Send this Character to the Display
00BA 0008               RETURN
           ;
00BB 3020  LOAD_AM      MOVLW   ‘ ‘             ; ASCII value for a Blank space
00BC 20D4               CALL    SEND_CHAR       ; Send this Character to the Display
00BD 3041               MOVLW   ‘A’             ; ASCII value for a Blank space
00BE 1FA0               BTFSS   FLAG_REG, AM    ; Is it AM or PM
00BF 3E0F               ADDLW   ‘P’ - ‘A’       ; Is PM, Add delta offset of ASCII Characters
00C0 20D4               CALL    SEND_CHAR       ; Send this Character to the Display
00C1 304D               MOVLW   ‘M’
00C2 20D4               CALL    SEND_CHAR       ; Send this Character to the Display
           ;
00C3 1683               BSF     STATUS, RP0     ; Bank 1
00C4 1381               BCF     OPTION_R, RBPU  ; Turn on PORTB Pull-up
00C5 1283               BCF     STATUS, RP0     ; Bank 0
00C6 2823               GOTO    CLR_RB          ; You’ve displayed the time, Clear RBIF
           ;
           ;
           ;*****************************************************************************
           ; The BIN_2_BCD routine converts the binary number, in the W register, to a
           ; binary coded decimal (BCD) munber. This BCD number is stored MSD:LSD. This
           ; routine is used by the DISPLAY subroutine, to convert the time values.
           ;*****************************************************************************
           ;
00C7 01B3  BIN_2_BCD    CLRF    MSD          ; This value contain the 10’s digit value
00C8 00B4               MOVWF   LSD          ; This value contain the 1’s digit value
00C9 300A  TENS_SUB     MOVLW   .10          ; A decimal 10
00CA 0234               SUBWF   LSD, W       ;
00CB 1C03               BTFSS   STATUS, C    ; Did this subtract cause a Negative Result?
```

```
00CC 3400                        RETLW   0            ; YES, Return from this Routine
00CD 00B4                        MOVWF   LSD          ; No, move the result into LSD
00CE 0AB3                        INCF    MSD, F       ; Increment the most significat digit
00CF 28C9                        GOTO    TENS_SUB     ;
                     ;
                     ;
                     ; Should NEVER get here
                     ;
00D0 1283            ERROR1      BCF     STATUS, RP0         ; Bank 0
                     ;
                             if ( Debug )
                                 bsf     PORTD, 1
                                 bcf     PORTD, 1
                             else
00D1 1407                        BSF     PORTC, 0
00D2 1007                        BCF     PORTC, 0
                             endif
00D3 28D0                        GOTO    ERROR1
                     ;
                     ;
                     ;*******************************************************************
                     ;* SendChar - Sends character to LCD                               *
                     ;* This routine splits the character into the upper and lower      *
                     ;* nibbles and sends them to the LCD, upper nibble first.          *
                     ;* The data is transmitted on the PORT<3:0> pins                   *
                     ;*******************************************************************

                     SEND_CHAR
00D4 00B6                        MOVWF   CHAR           ; Character to be sent is in W
00D5 20F2                        CALL    BUSY_CHECK     ; Wait for LCD to be ready
00D6 0E36                        SWAPF   CHAR, W
00D7 390F                        ANDLW   0x0F           ; Get upper nibble
00D8 0086                        MOVWF   LCD_DATA       ; Send data to LCD
00D9 1085                        BCF     LCD_CNTL, R_W  ; Set LCD to read
00DA 1505                        BSF     LCD_CNTL, RS   ; Set LCD to data mode
00DB 1405                        BSF     LCD_CNTL, E    ; toggle E for LCD
00DC 1005                        BCF     LCD_CNTL, E
00DD 0836                        MOVF    CHAR, W
00DE 390F                        ANDLW   0x0F           ; Get lower nibble
00DF 0086                        MOVWF   LCD_DATA       ; Send data to LCD
00E0 1405                        BSF     LCD_CNTL, E    ; toggle E for LCD
00E1 1005                        BCF     LCD_CNTL, E
00E2 0008                        RETURN

                     ;*******************************************************************
```

```
             ;* SendCmd - Sends command to LCD                                  *
             ;* This routine splits the command into the upper and lower        *
             ;* nibbles and sends them to the LCD, upper nibble first.          *
             ;* The data is transmitted on the PORT<3:0> pins                   *
             ;*******************************************************************

             SEND_CMD
00E3 00B6                MOVWF   CHAR            ; Character to be sent is in W
00E4 20F2                CALL    BUSY_CHECK      ; Wait for LCD to be ready
00E5 0E36                SWAPF   CHAR, W
00E6 390F                ANDLW   0x0F            ; Get upper nibble
00E7 0086                MOVWF   LCD_DATA        ; Send data to LCD
00E8 1085                BCF     LCD_CNTL, R_W   ; Set LCD to read
00E9 1105                BCF     LCD_CNTL, RS    ; Set LCD to command mode
00EA 1405                BSF     LCD_CNTL, E     ; toggle E for LCD
00EB 1005                BCF     LCD_CNTL, E
00EC 0836                MOVF    CHAR, W
00ED 390F                ANDLW   0x0F            ; Get lower nibble
00EE 0086                MOVWF   LCD_DATA        ; Send data to LCD
00EF 1405                BSF     LCD_CNTL, E     ; toggle E for LCD
00F0 1005                BCF     LCD_CNTL, E
00F1 0008                RETURN
             ;*******************************************************************
             ;* This routine checks the busy flag, returns when not busy        *
             ;*  Affects:                                                       *
             ;*      TEMP - Returned with busy/address                          *
             ;*******************************************************************

             BUSY_CHECK
             ;
                     if ( Debug )
                         bsf     PORTD, 3
                         bcf     PORTD, 3
                     endif
00F2 0186                CLRF    LCD_DATA         ;** Have PORTB<3:0> output low
00F3 1683                BSF     STATUS, RP0      ; Bank 1
00F4 1781                BSF     OPTION_R, RBPU   ; Turn off PORTB Pull-up
00F5 30FF                MOVLW   0xFF             ; Set PortB for input
00F6 0086                MOVWF   LCD_DATA_TRIS
00F7 1283                BCF     STATUS, RP0      ; Bank 0
00F8 1105                BCF     LCD_CNTL, RS     ; Set LCD for Command mode
00F9 1485                BSF     LCD_CNTL, R_W    ; Setup to read busy flag
00FA 1405                BSF     LCD_CNTL, E      ; Set E high
00FB 1005                BCF     LCD_CNTL, E      ; Set E low
00FC 0E06                SWAPF   LCD_DATA, W      ; Read upper nibble busy flag, DDRam address
```

```
00FD 39F0            0423            ANDLW   0xF0              ; Mask out lower nibble
00FE 00B5            0424            MOVWF   TEMP              ;
00FF 1405            0425            BSF     LCD_CNTL, E       ; Toggle E to get lower nibble
0100 1005            0426            BCF     LCD_CNTL, E
0101 0806            0427            MOVF    LCD_DATA, W       ; Read lower nibble busy flag, DDRam address
0102 390F            0428            ANDLW   0x0F              ; Mask out upper nibble
0103 04B5            0429            IORWF   TEMP, F           ; Combine nibbles
0104 1BB5            0430            BTFSC   TEMP, 7           ; Check busy flag, high = busy
0105 28F2            0431            GOTO    BUSY_CHECK        ; If busy, check again
0106 1085            0432            BCF     LCD_CNTL, R_W
0107 1683            0433            BSF     STATUS, RP0       ; Bank 1
0108 30F0            0434            MOVLW   0xF0              ;
0109 0086            0435            MOVWF   LCD_DATA_TRIS     ; RB7 - 4 = inputs, RB3 - 0 = output
010A 1283            0436            BCF     STATUS, RP0       ; Bank 0
010B 0008            0437            RETURN
                     0438 ;
                     0440 ;
                     0441 ;******************************************************************************
                     0442 ;*****      Start program here, Power-On Reset occurred.
                     0443 ;******************************************************************************
                     0444 ;
                     0445 START                                ; POWER_ON Reset (Beginning of program)
010C 1283            0446            BCF     STATUS, RP0       ; Bank 0
010D 300C            0447            MOVLW   0x0C              ; Decimal 12
010E 00B0            0448            MOVWF   HRS               ; HOURS = 12
010F 01B1            0449            CLRF    MIN               ; MIN   = 00
0110 3000            0450            MOVLW   0x00              ;
0111 00A0            0451            MOVWF   FLAG_REG          ; PM light is on
0112 3004            0452            MOVLW   0x04              ; Initial value of seconds (64d - 60d)
0113 00B2            0453            MOVWF   SECS              ; This allows a simple bit test to see if 60
                     0454                                      ;   secs has elapsed.
0114 3080            0455            MOVLW   0x80              ; TIM1H:TMR1L = 0x8000 gives 1 second
0115 008F            0456            MOVWF   TMR1H             ;   overflow, at 32 KHz.
0116 018E            0457            CLRF    TMR1L             ;
                     0458 ;
                     0459 MCLR_RESET                           ; A Master Clear Reset
0117 1283            0460            BCF     STATUS, RP0       ; Bank 0
0118 0183            0461            CLRF    STATUS            ; Do initialization (Bank 0)
0119 018B            0462            CLRF    INTCON
011A 018C            0463            CLRF    PIR1
011B 1683            0464            BSF     STATUS, RP0       ; Bank 1
011C 3000            0465            MOVLW   0x00              ; The LCD module does not like to work w/ weak pull-ups
011D 0081            0466            MOVWF   OPTION_R          ;
011E 018C            0467            CLRF    PIE1              ; Disable all peripheral interrupts
011F 30FF            0468            MOVLW   0xFF              ;
0120 009F            0469            MOVWF   ADCON1            ; Port A is Digital.
                     0470 ;
                     0471 ;
```

```
0121 1283               BCF     STATUS, RP0     ; Bank 0
0122 0185               CLRF    PORTA           ; ALL PORT output should output Low.
0123 0186               CLRF    PORTB
0124 0187               CLRF    PORTC
0125 0188               CLRF    PORTD
0126 0189               CLRF    PORTE
0127 1010               BCF     T1CON, TMR1ON   ; Timer 1 is NOT incrementing
                  ;
0128 1683               BSF     STATUS, RP0     ; Select Bank 1
0129 0185               CLRF    TRISA           ; RA5 -  0 outputs
012A 30F0               MOVLW   0xF0            ;
012B 0086               MOVWF   TRISB           ; RB7 - 4 inputs, RB3 - 0 outputs
012C 0187               CLRF    TRISC           ; RC Port are outputs
012D 1407               BSF     TRISC, T1OSO    ; RC0 needs to be input for the oscillator to function
012E 0188               CLRF    TRISD           ; RD Port are outputs
012F 0189               CLRF    TRISE           ; RE Port are outputs
0130 140C               BSF     PIE1, TMR1IE    ; Enable TMR1 Interrupt
0131 1381               BCF     OPTION_R, RBPU  ; Enable PORTB pull-ups
0132 1283               BCF     STATUS, RP0     ; Select Bank 0
0133 0886               MOVF    PORTB, F        ; Need to clear 1st RBIF, due to
0134 100B               BCF     INTCON, RBIF    ;    set up of PORTB
                  ;
                  ;
                  ; Initilize the LCD Display Module
                  ;
0135 0185               CLRF    LCD_CNTL        ; ALL PORT output should output Low.

                  DISPLAY_INIT
0136 3002               MOVLW   0x02            ; Command for 4-bit interface
0137 0086               MOVWF   LCD_DATA        ;
0138 1405               BSF     LCD_CNTL, E     ;
0139 1005               BCF     LCD_CNTL, E     ;
                  ;
                  ; This routine takes the calculated times that the delay loop needs to
                  ; be executed, based on the LCD_INIT_DELAY EQUate that includes the
                  ; frequency of operation. These uses registers before they are needed to
                  ; store the time.
                  ;
013A 3006         LCD_DELAY   MOVLW   LCD_INIT_DELAY  ;
013B 00B3               MOVWF   MSD             ; Use MSD and LSD Registers to Initilize LCD
013C 01B4               CLRF    LSD             ;
013D 0BB4         LOOP2       DECFSZ  LSD             ; Delay time = MSD * ((3 * 256) + 3) * Tcy
013E 293D               GOTO    LOOP2           ;
013F 0BB3               DECFSZ  MSD             ;
                  END_LCD_DELAY
0140 293D               GOTO    LOOP2           ;
                  ;
                  ; Command sequence for 2 lines of 5x7 characters
```

```
             ;
0141 3002    CMD_SEQ     MOVLW   0X02
0142 0086                MOVWF   LCD_DATA
0143 1405                BSF     LCD_CNTL, E     ;
0144 1005                BCF     LCD_CNTL, E     ;
0145 3008                MOVLW   0x08            ;
0146 0086                MOVWF   LCD_DATA        ;
0147 1405                BSF     LCD_CNTL, E     ;
0148 1005                BCF     LCD_CNTL, E     ;
             ;
             ; Busy Flag should be valid after this point
             ;
0149 300C                MOVLW   DISP_ON         ;
014A 20E3                CALL    SEND_CMD        ;
014B 3001                MOVLW   CLR_DISP        ;
014C 20E3                CALL    SEND_CMD        ;
014D 3006                MOVLW   ENTRY_INC       ;
014E 20E3                CALL    SEND_CMD        ;
014F 3080                MOVLW   DD_RAM_ADDR     ;
0150 20E3                CALL    SEND_CMD        ;
             ;
             ;
             ; Initialize the Special Function Registers (SFR) interrupts
             ;
0151 018C                CLRF    PIR1            ;
0152 300E                MOVLW   0x0E
0153 0090                MOVWF   T1CON           ; RC1 is overridden by TCKO
0154 170B                BSF     INTCON, PEIE    ; Enable Peripheral Interrupts
0155 158B                BSF     INTCON, RBIE    ; Disable PORTB<7:4> Change Interrupts
0156 178B                BSF     INTCON, GIE     ; Enable all Interrupts
             ;
0157 206E                CALL    INIT_DISPLAY    ;
0158 2075                CALL    DISPLAY         ;
             ;
0159 300E                MOVLW   0x0E
015A 0090                MOVWF   T1CON           ; Enable T1 Oscillator, Ext Clock, Async, prescaler = 1
015B 1410                BSF     T1CON, TMR1ON   ; Turn Timer 1 ON
             ;
                     if ( PICMaster )
             lzz         goto    lzz             ; Loop waiting for interrupts (for use with PICMASTER)
                     else
             ;
015C 0063    SLEEP_LP    SLEEP                   ; Wait for Change on PORTB interrupt. or TMR1 timeout
015D 0000                NOP                     ;
015E 295C                GOTO    SLEEP_LP        ;
             ;
                     endif
             ;
```

```
                ; Here is where you do things depending on the type of RESET (Not a Power-On Reset).
                ;
015F 1E03       OTHER_RESET   BTFSS  STATUS, TO     ; WDT Time-out?
0160 28D0       WDT_TIMEOUT   GOTO   ERROR1         ; YES, This is error condition
                        if ( Debug_PU )
0161 290C                   goto   START            ; MCLR reset, Goto START
                        else
                            GOTO   MCLR_RESET       ; MCLR reset, Goto MCLR_RESET
                        endif
                ;
                        if (Debug )
                END_START     NOP                   ; END lable for debug
                        endif
                ;
                ;
                    org     TABLE_ADDR
                ;
0400 00B5       NUM_TABLE       MOVWF   TEMP                ; Store value to TEMP register
0401 3004                       MOVLW   HIGH (TABLE_ADDR)   ; Ensure that the PCLATH high has the
0402 008A                       MOVWF   PCLATH              ;   correct value
0403 0835                       MOVF    TEMP, W             ; Value into table
0404 390F                       ANDLW   0x0F                ; Mask to 4-bits (00 - 0Fh)
0405 0782       NUM_TBL         ADDWF   PCL, F              ; Determine Offset into table
0406 3430                       RETLW   ‘0’                 ; ASCII value of “0” in W register
0407 3431                       RETLW   ‘1’                 ; ASCII value of “1” in W register
0408 3432                       RETLW   ‘2’                 ; ASCII value of “2” in W register
0409 3433                       RETLW   ‘3’                 ; ASCII value of “3” in W register
040A 3434                       RETLW   ‘4’                 ; ASCII value of “4” in W register
040B 3435                       RETLW   ‘5’                 ; ASCII value of “5” in W register
040C 3436                       RETLW   ‘6’                 ; ASCII value of “6” in W register
040D 3437                       RETLW   ‘7’                 ; ASCII value of “7” in W register
040E 3438                       RETLW   ‘8’                 ; ASCII value of “8” in W register
040F 3439                       RETLW   ‘9’                 ; ASCII value of “9” in W register
                                                            ; Any enter after is in error (Display an E)
0410 3445                       RETLW   ‘E’                 ; ASCII value of “E” in W register
0411 3445                       RETLW   ‘E’                 ; ASCII value of “E” in W register
0412 3445                       RETLW   ‘E’                 ; ASCII value of “E” in W register
0413 3445                       RETLW   ‘E’                 ; ASCII value of “E” in W register
0414 3445                       RETLW   ‘E’                 ; ASCII value of “E” in W register
0415 3445       NUM_TBL_END     RETLW   ‘E’                 ; ASCII value of “E” in W register
                ;
                    if ( (NUM_TBL & 0x0FF) >= (NUM_TBL_END & 0x0FF) )
                       MESSG   “Warning: Table NUM_TBL crosses page boundry in computed jump”
                    endif
                ;
                ;
                    org     PMEM_END            ; End of Program Memory
07FF 28D0                   GOTO    ERROR1          ; If you get here your program was lost
```

```
                     0619
                     0620     end
                     0621
                     0622


MPASM 01.01 Released     CLOCK.ASM   5-13-1994  13:11:9                 PAGE 15


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 XX-------------- ----------------

0400 : XXXXXXXXXXXXXXXX XXXXXX---------- ---------------- ----------------
0440 : ---------------- ---------------- ---------------- ----------------

0780 : ---------------- ---------------- ---------------- ----------------
07C0 : ---------------- ---------------- ---------------- ---------------X

All other memory blocks unused.


Errors   :    0
Warnings :   16
```

```
NOTE:   Special Function Register data memory locations in Bank 1, are
        specified by their true address in the file C74_REG.H.
        The use of the MPASM assembler will generate a warning message,
        when these lables are used with direct addressing.
```

APPENDIX B: CLOCK_01.H INCLUDE FILE

```
       nolist
;*****************************************************************************
;
;  This is the custom Header File for the real time clock application note
;       PROGRAM:       CLOCK_01.H
;       Revision:      5-04-94
;
;*****************************************************************************
; 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          ; Temp. register, Holds Most Significant
                                             ;   Digit of BIN to BCD conversion
LSD                   EQU     0x034          ; Temporary register, Holds Least Significant
                                             ;   Digit 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 coner of the Display
;
       list
```

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
RTCC                  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 Deffinitions   ************
;**********************************************
;
; 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)
;
GIE                   EQU            7
PEIE                  EQU            6
RTIE                  EQU            5
INTE                  EQU            4
RBIE                  EQU            3
RTIF                  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
RTS                   EQU            5
RTE                   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
```

NOTES:

WORLDWIDE SALES & SERVICE

AMERICAS

Corporate Office

Microchip Technology Inc.
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 602 786-7200 Fax: 602 786-7277
Technical Support: 602 786-7627
Web: http://www.mchip.com/microhip

Atlanta

Microchip Technology Inc.
500 Sugar Mill Road, Suite 200B
Atlanta, GA 30350
Tel: 770 640-0034 Fax: 770 640-0307

Boston

Microchip Technology Inc.
5 Mount Royal Avenue
Marlborough, MA 01752
Tel: 508 480-9990 Fax: 508 480-8575

Chicago

Microchip Technology Inc.
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 708 285-0071 Fax: 708 285-0075

Dallas

Microchip Technology Inc.
14651 Dallas Parkway, Suite 816
Dallas, TX 75240-8809
Tel: 214 991-7177 Fax: 214 991-8588

Dayton

Microchip Technology Inc.
35 Rockridge Road
Englewood, OH 45322
Tel: 513 832-2543 Fax: 513 832-2841

Los Angeles

Microchip Technology Inc.
18201 Von Karman, Suite 455
Irvine, CA 92715
Tel: 714 263-1888 Fax: 714 263-1338

New York

Microchip Technology Inc.
150 Motor Parkway, Suite 416
Hauppauge, NY 11788
Tel: 516 273-5305 Fax: 516 273-5335

AMERICAS (continued)

San Jose

Microchip Technology Inc.
2107 North First Street, Suite 590
San Jose, CA 95131
Tel: 408 436-7950 Fax: 408 436-7955

ASIA/PACIFIC

Hong Kong

Microchip Technology
Unit No. 3002-3004, Tower 1
Metroplaza
223 Hing Fong Road
Kwai Fong, N.T. Hong Kong
Tel: 852 2 401 1200 Fax: 852 2 401 3431

Korea

Microchip Technology
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku,
Seoul, Korea
Tel: 82 2 554 7200 Fax: 82 2 558 5934

Singapore

Microchip Technology
200 Middle Road
#10-03 Prime Centre
Singapore 188980
Tel: 65 334 8870 Fax: 65 334 8850

Taiwan

Microchip Technology
10F-1C 207
Tung Hua North Road
Taipei, Taiwan, ROC
Tel: 886 2 717 7175 Fax: 886 2 545 0139

EUROPE

United Kingdom

Arizona Microchip Technology Ltd.
Unit 6, The Courtyard
Meadow Bank, Furlong Road
Bourne End, Buckinghamshire SL8 5AJ
Tel: 44 0 1628 851077 Fax: 44 0 1628 850259

France

Arizona Microchip Technology SARL
2 Rue du Buisson aux Fraises
91300 Massy - France
Tel: 33 1 69 53 63 20 Fax: 33 1 69 30 90 79

Germany

Arizona Microchip Technology GmbH
Gustav-Heinemann-Ring 125
D-81739 Muenchen, Germany
Tel: 49 89 627 144 0 Fax: 49 89 627 144 44

Italy

Arizona Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Pegaso Ingresso No. 2
Via Paracelso 23, 20041
Agrate Brianza (MI) Italy
Tel: 39 039 689 9939 Fax: 39 039 689 9883

JAPAN

Microchip Technology Intl. Inc.
Benex S-1 6F
3-18-20, Shin Yokohama
Kohoku-Ku, Yokohama
Kanagawa 222 Japan
Tel: 81 45 471 6166 Fax: 81 45 471 6122

9/22/95

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip's products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights. The Microchip logo and name are registered trademarks of Microchip Technology Inc. All rights reserved. All other trademarks mentioned herein are the property of their respective companies.