Interfacing to an LCD Module

INTRODUCTION

This application note interfaces a PIC16CXX device to the Hitachi LM032L LCD character display module. This module is a two line by twenty character display. LCD modules are useful for displaying text information from a system. In large volume applications, the use of custom LCD displays become economical. These routines should be a good starting point for users whose application implement a custom LCD. This source code should be compatible with the PIC16C5X devices, after modifications for the special function register initialization, but has not been verified on those devices.

OPERATION

The Hitachi® LM032L LCD character display module can operate in one of two modes. The first (and default) mode is the 4-bit data interface mode. The second is the 8-bit data interface mode. When operating in 4-bit mode, two transfers per character / command are required. 8-bit mode, though easier to implement (less program memory) requires four additional I/O lines. The use of 8-bit mode is strictly a program memory size / I/O trade-off. The three most common data interfaces from the microcontroller are:

1. An 8-bit interface.
2. A 4-bit interface, with data transfers on the high nibble of the port.
3. A 4-bit interface, with data transfers on the low nibble of the port.

The LCD module also has three control Signal, Enable (E), Read / Write (R_W), and Register Select (RS). These functions of these control signals are show in Table 1.

TABLE 1: CONTROL SIGNAL FUNCTIONS

Control Signal	Function
E	Causes data / control state to be latched Rising Edge = Latches control state (RS and R_W) Falling Edge = Latches data
RS	Register Select Control 0 = LCD in command mode 1 = LCD in data mode
R_W	Read / Write control 0 = LCD to read data 1 = LCD to write data

A single source file, with conditional assemble is used to generate these three options. This requires two flags. The flags and their results are shown in Table 2.

TABLE 2: CONDITIONAL ASSEMBLY FLAGS

Flags		
Four_bit	Data_HI	Result
1	0	4-bit mode. Data transferred on the low nibble of the port.
1	1	4-bit mode. Data transferred on the high nibble of the port.
0	X	8-bit mode.

Figure 1A through Figure 1C show the block diagrams for the three different data interfaces. The LCD_CNTL and LCD_DATA lines are user definable to their port assignment. This is accomplished with EQUate statements in the source code. See Appendices B - D.

FIGURE 1A: 8-BIT DATA INTERFACE

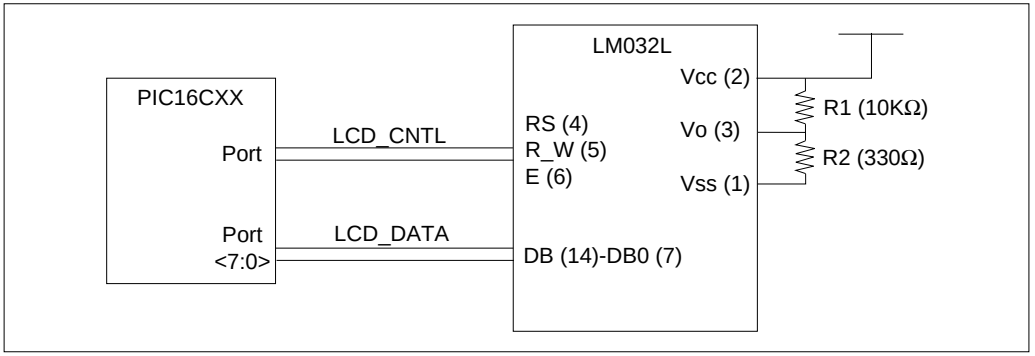


FIGURE 1B: 4-BIT MODE. DATA TRANSFERRED ON THE HIGH NIBBLE OF THE PORT

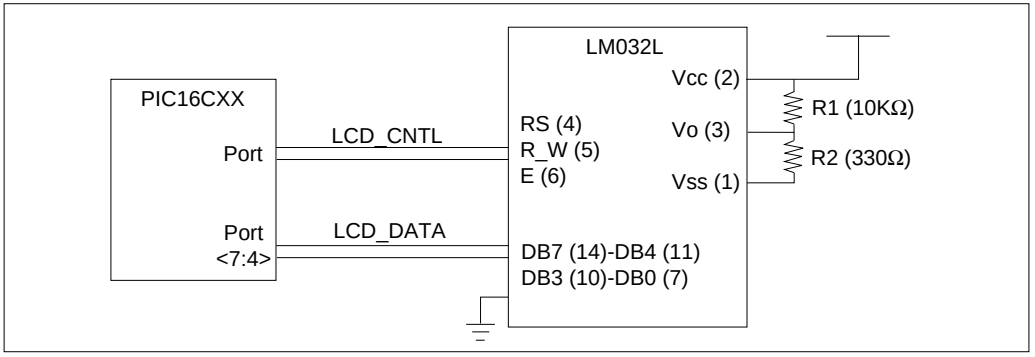
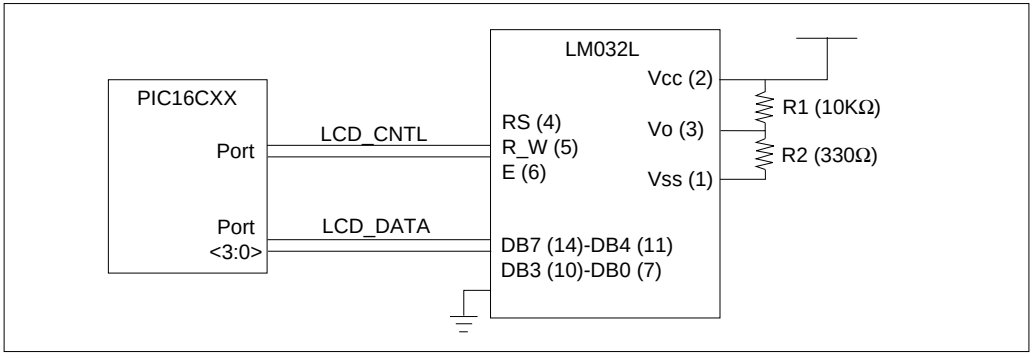


FIGURE 1C: 4-BIT MODE. DATA TRANSFERRED ON THE LOW NIBBLE OF THE PORT



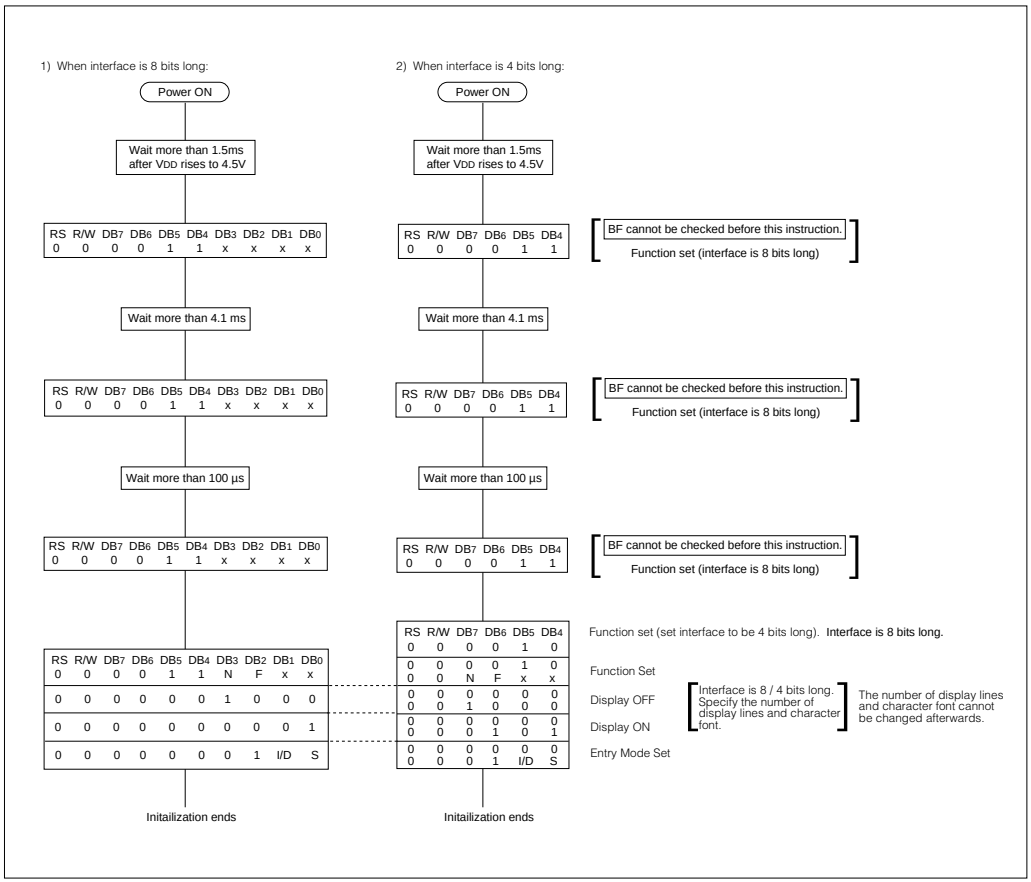
LCD ‘s (drivers) are slow devices when compared to microcontrollers. Care must be taken from having communication occur to quickly. The timing requirements of the LM032L are shown in Appendix A. It is recommended that the complete specifications of the LM032L be acquired from Hitachi or an Hitachi distributor. The literature number is CE-E613Q and M24T013 for the display driver.

When the module powers up, the default data transfer mode is 8-bit. The initialization sequence only requires commands that are 4-bit in length. The last initialization command then needs to be sent to the display to specify the data transfer width (4- or 8-bit). Then a delay of 4.6 ms must be executed before the LCD module can be initialized. Some of the LCD module commands are:

- 1 or 2 lines of characters
- Display on /off
- Clear display
- Increment / do not increment character address pointer after each character
- Load character address pointer

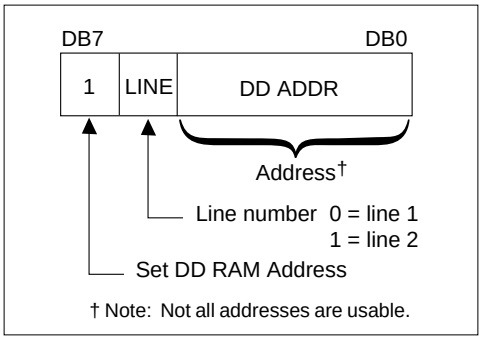
The initialization flow for the module is shown in Figure 2.

FIGURE 2: INITIALIZATION FLOW FOR LCD MODULE



After initialization, each character address is individually addressable. Figure 3 shows the structure of the command to specify the character address.

FIGURE 3: CHARACTER ADDRESS COMMAND FORMAT



The Hatachi Display Drive (HD44780A) has 80 bytes of RAM. The LM032L modules only use 40 bytes of the available RAM (2 x 20 characters). It is possible to use the remaining RAM locations for storage of other information.

Figure 4 shows the display data positions supported by the display driver as well as the characters actually displayed by the module (the shaded addresses).

The program example implemented here uses the auto character increment feature. This automatically increments the character address pointer after each character is written to the display.

CONCLUSION

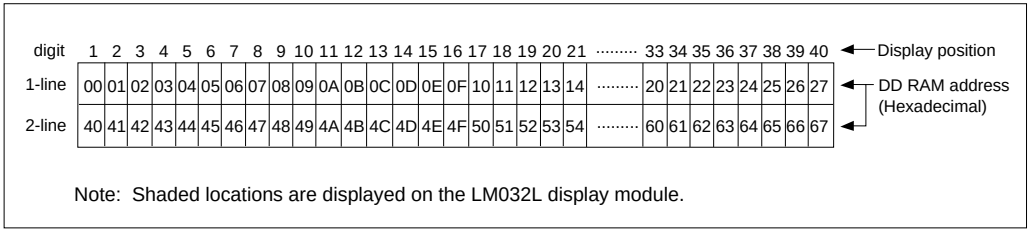
The Hitachi LM032L character display module is useful for the display of information. The selection of 4-bit or 8-bit data transfer mode is strictly a program memory size / I/O resource trade-off. The supplied code is easily used in one of three common data interfaces. The source is easily modifiable to the designers specific application needs. Other display modules / drivers maybe implemented with the appropriate modifications. Table 4 shows the resource requirements for the three subroutines SEND_CHAR, SEND_COMMAND, and BUSY_CHECK in the various data interface modes.

TABLE 4: RESOURCE REQUIREMENTS

Mode	Program Memory	Data Memory	Verified On
8-bit	32	3	PICDEM-2 *
4-bit, Data transferred on the high nibble of the port	53	3	PICDEM-2 *
4-bit, Data transferred on the high nibble of the port	53	3	Low Power Real Time Clock Board (AN582)

* Jumper J6 must be removed.

FIGURE 4: DISPLAY DRIVER (DD) RAM LOCATIONS



Note: Shaded locations are displayed on the LM032L display module.

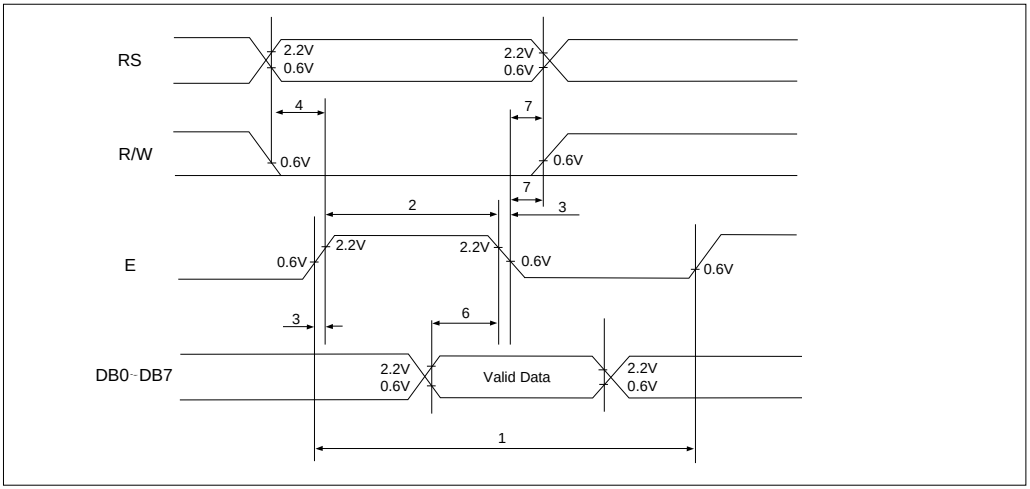
Written By: *Mark Palmer - Sr. Application Engineer*
Code By: *Mark Palmer / Scott Fink - Sr. Application Engineers*

APPENDIX A: LM032L TIMING REQUIREMENTS

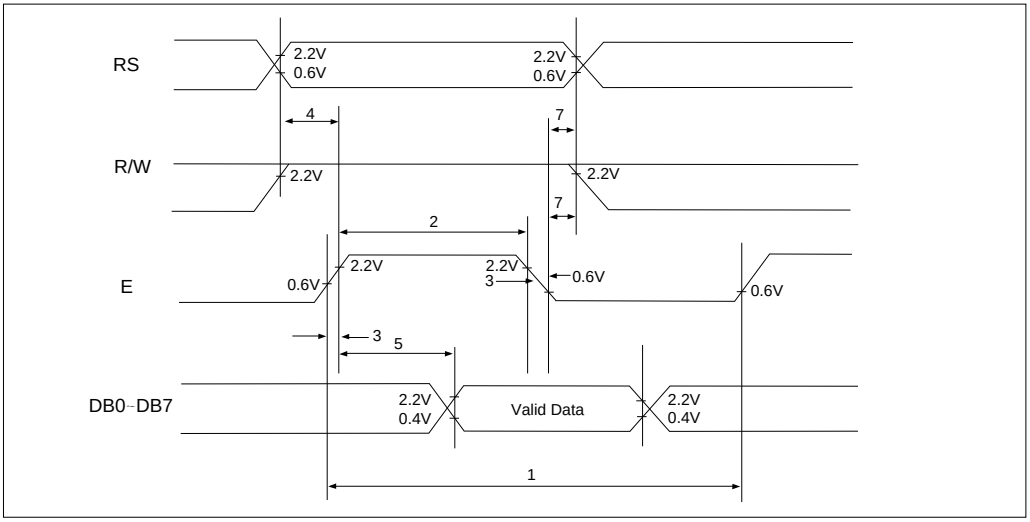
TIMING CHARACTERISTICS

Parm #	Symbol	Characterisitics	Min.	Typ.	Max.	Unit
1	t_{cyc}	Enable cycle time	1.0	—	—	µs
2	PW_{EH}	Enable pulse width	450	—	—	ns
3	t_{Er}, t_{Ef}	Enable rise / fall time	—	—	25	ns
4	t_{AS}	RS, R/W set up time	140	—	—	ns
5	t_{DDR}	Data delay time	—	—	320	ns
6	t_{DSW}	Data set up time	195	—	—	ns
7	t_H	Hold time	20	—	—	ns

DATA WRITE INTERFACE TIMING



DATA READ INTERFACE TIMING



Note: Refer to Hitachi documentation for most current timing specifications.

LM032L PIN CONNECTION

Pin No.	Symbol	Level	Function	
1	V_{SS}	—	0V	
2	V_{DD}	—	+5V	Power Supply
3	V_O	—	—	
4	RS	H/L	L: Instruction Code Input H: Data Input	
5	R/W	H/L	H: Data Read (LCD module→MPU) L: Data Write (LCD module←MPU)	
6	E	H,H→L	Enable Signal	
7	DB0	H/L	Data Bus Line Note (1), (2)	
8	DB1	H/L		
9	DB2	H/L		
10	DB3	H/L		
11	DB4	H/L		
12	DB5	H/L		
13	DB6	H/L		
14	DB7	H/L		

Notes:

In the HD44780, the data can be sent in either a 4-bit 2-operation or a 8-bit 1-operation, so that it can interface to both 4- and 8-bit MPUs.

(1) When interface data is 4-bits long, data is transferred using only 4 buses of DB~DB_7 and DB_0~DB_3 are not used. Data transfer between the HD44780 and the MPU completes when 4-bit data is tranferred twice. Data of the higher order 4 bits (contents of DB_4~DB_7 when interface data is 8-bits long) is tranferred first and then lower order 4 bits (contents of DB_4~DB_7 when interface data is 8-bits long).

(2) When interface data is 8-bits long, data is transferred using 8 data buses of DB0~DB_7.

APPENDIX B: 8-BIT DATA INTERFACE LISTING

```
MPASM 00.00.68 Intermediate    LM032L.ASM   6-8-1994  0:53:47                PAGE  1


LOC  OBJECT CODE     LINE SOURCE TEXT

                     0001         LIST P=16C64, F=INHX8M
                     0002 ;
                     0003 ; This program interfaces to a Hitachi (LM032L) 2 line by 20 character display
                     0004 ; module. The program assembles for either 4-bit or 8-bit data interface, depending
                     0005 ; on the value of the 4bit flag. LCD_DATA is the port which supplies the data to
                     0006 ; the LM032L, while LCD_CNTL is the port that has the control lines ( E, RS, R_W ).
                     0007 ; In 4-bit mode the data is transfer on the high nibble of the port ( PORT<7:4> ).
                     0008 ;
                     0009 ;       Program = LM032L.ASM
                     0010 ;       Revision Date:   5-10-94
                     0011 ;
                     0012 ;
                     0013        include <C74_reg.h>
                     0233
                     0013
                     0014        include <lm032l.h>
                     0014
                     0015 ;
0000                 0016 Four_bit         EQU     FALSE       ; Selects 4- or 8-bit data transfers
0001                 0017 Data_HI          EQU     TRUE       ; If 4-bit transfers, Hi or Low nibble of PORT
                     0018 ;
                     0019 ;
                     0020     if ( Four_bit && !Data_HI )
                     0021 ;
                     0022 LCD_DATA         EQU     PORTB
                     0023 LCD_DATA_TRIS    EQU     TRISB
                     0024 ;
                     0025     else
                     0026 ;
0008                 0027 LCD_DATA         EQU     PORTD
0088                 0028 LCD_DATA_TRIS    EQU     TRISD
                     0029 ;
                     0030     endif
                     0031 ;
```

```
0005                    0032 LCD_CNTL    EQU     PORTA
                        0033 ;
                        0034 ;
                        0035 ;
                        0036 ; LCD Display Commands and Control Signal names.
                        0037 ;
                        0038     if ( Four_bit && !Data_HI )
                        0039 ;
                        0040 E           EQU     0               ; LCD Enable control line
                        0041 R_W         EQU     1               ; LCD Read/Write control line
                        0042 RS          EQU     2               ; LCD Register Select control line
                        0043 ;
                        0044     else
                        0045 ;
0003                    0046 E           EQU     3               ; LCD Enable control line
0002                    0047 R_W         EQU     2               ; LCD Read/Write control line
0001                    0048 RS          EQU     1               ; LCD Register Select control line
                        0049 ;
                        0050     endif
                        0051 ;
                        0052 ;
0030                    0053 TEMP1       EQU     0x030
                        0054 ;
                        0055        org     RESET_V             ; RESET vector location
0000 2808               0056 RESET       GOTO    START           ;
                        0057 ;
                        0058 ; This is the Periperal Interrupt routine. Should NOT get here
                        0059 ;
                        0061         org     ISR_V               ; Interrupt vector location
                        0062 PER_INT_V
0004 1283               0063 ERROR1      BCF     STATUS, RP0     ; Bank 0
0005 1407               0064             BSF     PORTC, 0
0006 1007               0065             BCF     PORTC, 0
0007 2804               0066             GOTO    ERROR1
                        0067 ;
                        0068 ;
                        0069 ;
                        0070 START                               ; POWER_ON Reset (Beginning of program)
0008 0183               0071             CLRF    STATUS          ; Do initialization (Bank 0)
0009 018B               0072             CLRF    INTCON
000A 018C               0073             CLRF    PIR1
000B 1683               0074             BSF     STATUS, RP0     ; Bank 1
000C 3000               0075             MOVLW   0x00            ; The LCD module does not like to work w/ weak pull-ups
000D 0081               0076             MOVWF   OPTION_R        ;
```

```
000E 018C            0077             CLRF    PIE1            ; Disable all peripheral interrupts
000F 30FF            0078             MOVLW   0xFF            ;
0010 009F            0079             MOVWF   ADCON1          ; Port A is Digital.
                     0080 ;
                     0081 ;
0011 1283            0082             BCF     STATUS, RP0     ; Bank 0
0012 0185            0083             CLRF    PORTA           ; ALL PORT output should output Low.
0013 0186            0084             CLRF    PORTB
0014 0187            0085             CLRF    PORTC
0015 0188            0086             CLRF    PORTD
0016 0189            0087             CLRF    PORTE
0017 1010            0088             BCF     T1CON, TMR1ON   ; Timer 1 is NOT incrementing
                     0089 ;
0018 1683            0090             BSF     STATUS, RP0     ; Select Bank 1
0019 0185            0091             CLRF    TRISA           ; RA5 -  0 outputs
001A 30F0            0092             MOVLW   0xF0            ;
001B 0086            0093             MOVWF   TRISB           ; RB7 - 4 inputs, RB3 - 0 outputs
001C 0187            0094             CLRF    TRISC           ; RC Port are outputs
001D 1407            0095             BSF     TRISC, T1OSO    ; RC0 needs to be input for the oscillator to function
001E 0188            0096             CLRF    TRISD           ; RD Port are outputs
001F 0189            0097             CLRF    TRISE           ; RE Port are outputs
0020 140C            0098             BSF     PIE1, TMR1IE    ; Enable TMR1 Interrupt
0021 1781            0099             BSF     OPTION_R, RBPU  ; Disable PORTB pull-ups
0022 1283            0100             BCF     STATUS, RP0     ; Select Bank 0
                     0101 ;
                     0103 ;
                     0104 ; Initilize the LCD Display Module
                     0105 ;
0023 0185            0106             CLRF    LCD_CNTL        ; ALL PORT output should output Low.
                     0107
                     0108 DISPLAY_INIT
                     0109     if ( Four_bit && !Data_HI )
                     0110             MOVLW   0x02            ; Command for 4-bit interface low nibble
                     0111     endif
                     0112 ;
                     0113     if ( Four_bit && Data_HI )
                     0114             MOVLW   0x020           ; Command for 4-bit interface high nibble
                     0115     endif
                     0116 ;
                     0117     if ( !Four_bit )
0024 3038            0118             MOVLW   0x038           ; Command for 8-bit interface
                     0119     endif
                     0120 ;
0025 0088            0121             MOVWF   LCD_DATA        ;
0026 1585            0122             BSF     LCD_CNTL, E     ;
```

```
0027 1185            0123             BCF     LCD_CNTL, E     ;
                     0124 ;
                     0125 ; This routine takes the calculated times that the delay loop needs to
                     0126 ; be executed, based on the LCD_INIT_DELAY EQUate that includes the
                     0127 ; frequency of operation. These uses registers before they are needed to
                     0128 ; store the time.
                     0129 ;
0028 3006            0130 LCD_DELAY   MOVLW   LCD_INIT_DELAY  ;
0029 00B3            0131             MOVWF   MSD             ; Use MSD and LSD Registers to Initilize LCD
002A 01B4            0132             CLRF    LSD             ;
002B 0BB4            0133 LOOP2       DECFSZ  LSD             ; Delay time = MSD * ((3 * 256) + 3) * Tcy
002C 282B            0134             GOTO    LOOP2           ;
002D 0BB3            0135             DECFSZ  MSD             ;
                     0136 END_LCD_DELAY
002E 282B            0137             GOTO    LOOP2           ;
                     0138 ;
                     0139 ; Command sequence for 2 lines of 5x7 characters
                     0140 ;
                     0141 CMD_SEQ
                     0142 ;
                     0143     if ( Four_bit )
                     0144         if ( !Data_HI )
                     0145             MOVLW   0X02            ; 4-bit low nibble xfer
                     0146         else
                     0147             MOVLW   0X020           ; 4-bit high nibble xfer
                     0148         endif
                     0149 ;
                     0150     else                            ; 8-bit mode
002F 3038            0151             MOVLW   0X038
                     0152     endif
                     0153 ;
0030 0088            0154             MOVWF   LCD_DATA        ; This code for both 4-bit and 8-bit modes
0031 1585            0155             BSF     LCD_CNTL, E     ;
0032 1185            0156             BCF     LCD_CNTL, E     ;
                     0157 ;
                     0158     if ( Four_bit )                 ; This code for only 4-bit mode (2nd xfer)
                     0159         if ( !Data_HI )
                     0160             MOVLW   0x08            ; 4-bit low nibble xfer
                     0161         else
                     0162             MOVLW   0x080           ; 4-bit high nibble xfer
                     0163         endif
                     0164             MOVWF   LCD_DATA        ;
                     0165             BSF     LCD_CNTL, E     ;
                     0166             BCF     LCD_CNTL, E     ;
```

```
                     0167     endif
                     0168 ;
                     0169 ; Busy Flag should be valid after this point
                     0170 ;
0033 300C            0171             MOVLW   DISP_ON         ;
0034 206A            0172             CALL    SEND_CMD        ;
0035 3001            0173             MOVLW   CLR_DISP        ;
0036 206A            0174             CALL    SEND_CMD        ;
0037 3006            0175             MOVLW   ENTRY_INC       ;
0038 206A            0176             CALL    SEND_CMD        ;
0039 3080            0177             MOVLW   DD_RAM_ADDR     ;
003A 206A            0178             CALL    SEND_CMD        ;
                     0179 ;
                     0181 ;
                     0182 ;Send a message the hard way
003B 304D            0183             movlw   'M'
003C 2061            0184             call    SEND_CHAR
003D 3069            0185             movlw   'i'
003E 2061            0186             call    SEND_CHAR
003F 3063            0187             movlw   'c'
0040 2061            0188             call    SEND_CHAR
0041 3072            0189             movlw   'r'
0042 2061            0190             call    SEND_CHAR
0043 306F            0191             movlw   'o'
0044 2061            0192             call    SEND_CHAR
0045 3063            0193             movlw   'c'
0046 2061            0194             call    SEND_CHAR
0047 3068            0195             movlw   'h'
0048 2061            0196             call    SEND_CHAR
0049 3069            0197             movlw   'i'
004A 2061            0198             call    SEND_CHAR
004B 3070            0199             movlw   'p'
004C 2061            0200             call    SEND_CHAR
                     0201
004D 30C0            0202             movlw   B'11000000'     ;Address DDRam first character, second line
004E 206A            0203             call    SEND_CMD
                     0204
                     0205 ;Demonstration of the use of a table to output a message
004F 3000            0206             movlw   0               ;Table address of start of message
                     0207 dispmsg
0050 00B0            0208             movwf   TEMP1           ;TEMP1 holds start of message address
0051 2085            0209             call    Table
0052 39FF            0210             andlw   0FFh            ;Check if at end of message (zero
0053 1903            0211             btfsc   STATUS,Z        ;returned at end)
0054 2859            0212             goto    out
0055 2061            0213             call    SEND_CHAR       ;Display character
0056 0830            0214             movf    TEMP1,w         ;Point to next character
```

```
0057 3E01           0215            addlw   1
0058 2850           0216            goto    dispmsg
                    0217 out
                    0218 loop
0059 2859           0219            goto    loop            ;Stay here forever
                    0220 ;
                    0221 ;
                    0222 INIT_DISPLAY
005A 300C           0223            MOVLW   DISP_ON             ; Display On, Cursor On
005B 206A           0224            CALL    SEND_CMD            ; Send This command to the Display Module
005C 3001           0225            MOVLW   CLR_DISP            ; Clear the Display
005D 206A           0226            CALL    SEND_CMD            ; Send This command to the Display Module
005E 3006           0227            MOVLW   ENTRY_INC           ; Set Entry Mode Inc., No shift
005F 206A           0228            CALL    SEND_CMD            ; Send This command to the Display Module
0060 0008           0229            RETURN
                    0230 ;
                    0232 ;
                    0233 ;*****************************************************************
                    0234 ;* The LCD Module Subroutines                                    *
                    0235 ;*****************************************************************
                    0236 ;
                    0237    if ( Four_bit )      ; 4-bit Data transfers?
                    0238 ;
                    0239        if ( Data_HI )   ; 4-bit transfers on the high nibble of the PORT
                    0240 ;
                    0241 ;*****************************************************************
                    0242 ;*SendChar - Sends character to LCD                            *
                    0243 ;*This routine splits the character into the upper and lower   *
                    0244 ;*nibbles and sends them to the LCD, upper nibble first.       *
                    0245 ;*****************************************************************
                    0246 ;
                    0247 SEND_CHAR
                    0248            MOVWF   CHAR            ;Character to be sent is in W
                    0249            CALL    BUSY_CHECK      ;Wait for LCD to be ready
                    0250            MOVF    CHAR, w
                    0251            ANDLW   0xF0            ;Get upper nibble
                    0252            MOVWF   LCD_DATA        ;Send data to LCD
                    0253            BCF     LCD_CNTL, R_W   ;Set LCD to read
                    0254            BSF     LCD_CNTL, RS    ;Set LCD to data mode
                    0255            BSF     LCD_CNTL, E     ;toggle E for LCD
                    0256            BCF     LCD_CNTL, E
                    0257            SWAPF   CHAR, w
                    0258            ANDLW   0xF0            ;Get lower nibble
                    0259            MOVWF   LCD_DATA        ;Send data to LCD
                    0260            BSF     LCD_CNTL, E     ;toggle E for LCD
                    0261            BCF     LCD_CNTL, E
```

```
                    RETURN
;
              else             ; 4-bit transfers on the low nibble of the PORT
;
;*******************************************************************
;* SEND_CHAR - 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
                    MOVWF   CHAR            ; Character to be sent is in W
                    CALL    BUSY_CHECK      ; Wait for LCD to be ready
                    SWAPF   CHAR, W
                    ANDLW   0x0F            ; Get upper nibble
                    MOVWF   LCD_DATA        ; Send data to LCD
                    BCF     LCD_CNTL, R_W   ; Set LCD to read
                    BSF     LCD_CNTL, RS    ; Set LCD to data mode
                    BSF     LCD_CNTL, E     ; toggle E for LCD
                    BCF     LCD_CNTL, E
                    MOVF    CHAR, W
                    ANDLW   0x0F            ; Get lower nibble
                    MOVWF   LCD_DATA        ; Send data to LCD
                    BSF     LCD_CNTL, E     ; toggle E for LCD
                    BCF     LCD_CNTL, E
                    RETURN
;
              endif
          else
;
;******************************************************************
;* SEND_CHAR - Sends character contained in register W to LCD    *
;* This routine sends the entire character to the PORT           *
;* The data is transmitted on the PORT<7:0> pins                 *
;******************************************************************
;
SEND_CHAR
0061 00B6           MOVWF   CHAR            ; Character to be sent is in W
0062 2073           CALL    BUSY_CHECK      ; Wait for LCD to be ready
0063 0836           MOVF    CHAR, w
0064 0088           MOVWF   LCD_DATA        ; Send data to LCD
0065 1105           BCF     LCD_CNTL, R_W   ; Set LCD in read mode
0066 1485           BSF     LCD_CNTL, RS    ; Set LCD in data mode
0067 1585           BSF     LCD_CNTL, E     ; toggle E for LCD
0068 1185           BCF     LCD_CNTL, E
0069 0008           RETURN
```

```
0309 ;
0310     endif
0311 ;
0313 ;
0314 ;*****************************************************************
0315 ;* SendCmd - Sends command to LCD                                *
0316 ;* This routine splits the command into the upper and lower      *
0317 ;* nibbles and sends them to the LCD, upper nibble first.        *
0318 ;* The data is transmitted on the PORT<3:0> pins                 *
0319 ;*****************************************************************
0320 ;
0321     if ( Four_bit )      ; 4-bit Data transfers?
0322 ;
0323         if ( Data_HI )   ; 4-bit transfers on the high nibble of the PORT
0324 ;
0325 ;*****************************************************************
0326 ;* SEND_CMD - Sends command to LCD                               *
0327 ;* This routine splits the command into the upper and lower      *
0328 ;* nibbles and sends them to the LCD, upper nibble first.        *
0329 ;*****************************************************************
0330
0331 SEND_CMD
0332             MOVWF   CHAR            ; Character to be sent is in W
0333             CALL    BUSY_CHECK      ; Wait for LCD to be ready
0334             MOVF    CHAR,w
0335             ANDLW   0xF0            ; Get upper nibble
0336             MOVWF   LCD_DATA        ; Send data to LCD
0337             BCF     LCD_CNTL,R_W    ; Set LCD to read
0338             BCF     LCD_CNTL,RS     ; Set LCD to command mode
0339             BSF     LCD_CNTL,E      ; toggle E for LCD
0340             BCF     LCD_CNTL,E
0341             SWAPF   CHAR,w
0342             ANDLW   0xF0            ; Get lower nibble
0343             MOVWF   LCD_DATA        ; Send data to LCD
0344             BSF     LCD_CNTL,E      ; toggle E for LCD
0345             BCF     LCD_CNTL,E
0346             RETURN
0347 ;
0348         else            ; 4-bit transfers on the low nibble of the PORT
0349 ;
0350 SEND_CMD
0351             MOVWF   CHAR            ; Character to be sent is in W
0352             CALL    BUSY_CHECK      ; Wait for LCD to be ready
0353             SWAPF   CHAR, W
0354             ANDLW   0x0F            ; Get upper nibble
0355             MOVWF   LCD_DATA        ; Send data to LCD
0356             BCF     LCD_CNTL, R_W   ; Set LCD to read
```

```
                         BCF     LCD_CNTL, RS    ; Set LCD to command mode
                         BSF     LCD_CNTL, E     ; toggle E for LCD
                         BCF     LCD_CNTL, E
                         MOVF    CHAR, W
                         ANDLW   0x0F            ; Get lower nibble
                         MOVWF   LCD_DATA        ; Send data to LCD
                         BSF     LCD_CNTL, E     ; toggle E for LCD
                         BCF     LCD_CNTL, E
                         RETURN
           ;
                     endif
                 else
           ;
           ;***************************************************************
           ;* SEND_CND - Sends command contained in register W to LCD    *
           ;* This routine sends the entire character to the PORT        *
           ;* The data is transmitted on the PORT<7:0> pins              *
           ;***************************************************************

           SEND_CMD
006A 00B6                MOVWF   CHAR            ; Command to be sent is in W
006B 2073                CALL    BUSY_CHECK      ; Wait for LCD to be ready
006C 0836                MOVF    CHAR, w
006D 0088                MOVWF   LCD_DATA        ; Send data to LCD
006E 1105                BCF     LCD_CNTL, R_W   ; Set LCD in read mode
006F 1085                BCF     LCD_CNTL, RS    ; Set LCD in command mode
0070 1585                BSF     LCD_CNTL, E     ; toggle E for LCD
0071 1185                BCF     LCD_CNTL, E
0072 0008                RETURN
           ;
                 endif
           ;
           ;
                 if ( Four_bit )      ; 4-bit Data transfers?
           ;
                     if ( Data_HI )   ; 4-bit transfers on the high nibble of the PORT
           ;
           ;********************************************************************
           ;* This routine checks the busy flag, returns when not busy        *
           ;*  Affects:                                                       *
           ;*      TEMP - Returned with busy/address                          *
           ;********************************************************************
           ;
           BUSY_CHECK
                         BSF     STATUS, RP0     ; Select Register page 1
                         MOVLW   0xFF            ; Set Port_D for input
                         MOVWF   LCD_DATA_TRIS
```

```
            BCF     STATUS, RP0     ; Select Register page 0
            BCF     LCD_CNTL, RS    ; Set LCD for Command mode
            BSF     LCD_CNTL, R_W   ; Setup to read busy flag
            BSF     LCD_CNTL, E     ; Set E high
            BCF     LCD_CNTL, E     ; Set E low
            MOVF    LCD_DATA, W     ; Read upper nibble busy flag, DDRam address
            ANDLW   0xF0            ; Mask out lower nibble
            MOVWF   TEMP
            BSF     LCD_CNTL, E     ; Toggle E to get lower nibble
            BCF     LCD_CNTL, E
            SWAPF   LCD_DATA, w     ; Read lower nibble busy flag, DDRam address
            ANDLW   0x0F            ; Mask out upper nibble
            IORWF   TEMP            ; Combine nibbles
            BTFSC   TEMP, 7         ; Check busy flag, high = busy
            GOTO    BUSY_CHECK      ; If busy, check again
            BCF     LCD_CNTL, R_W
            BSF     STATUS, RP0     ; Select Register page 1
            MOVLW   0x0F
            MOVWF   LCD_DATA_TRIS   ; Set Port_D for output
            BCF     STATUS, RP0     ; Select Register page 0
            RETURN
;
        else            ; 4-bit transfers on the low nibble of the PORT
;
;*******************************************************************
;* This routine checks the busy flag, returns when not busy        *
;*  Affects:                                                       *
;*      TEMP - Returned with busy/address                          *
;*******************************************************************
;
BUSY_CHECK
            BSF     STATUS, RP0     ; Bank 1
            MOVLW   0xFF            ; Set PortB for input
            MOVWF   LCD_DATA_TRIS
            BCF     STATUS, RP0     ; Bank 0
            BCF     LCD_CNTL, RS    ; Set LCD for Command mode
            BSF     LCD_CNTL, R_W   ; Setup to read busy flag
            BSF     LCD_CNTL, E     ; Set E high
            BCF     LCD_CNTL, E     ; Set E low
            SWAPF   LCD_DATA, W     ; Read upper nibble busy flag, DDRam address
            ANDLW   0xF0            ; Mask out lower nibble
            MOVWF   TEMP            ;
            BSF     LCD_CNTL, E     ; Toggle E to get lower nibble
            BCF     LCD_CNTL, E
            MOVF    LCD_DATA, W     ; Read lower nibble busy flag, DDRam address
            ANDLW   0x0F            ; Mask out upper nibble
            IORWF   TEMP, F         ; Combine nibbles
```

```
                     0452             BTFSC   TEMP, 7         ; Check busy flag, high = busy
                     0453             GOTO    BUSY_CHECK      ; If busy, check again
                     0454             BCF     LCD_CNTL, R_W
                     0455             BSF     STATUS, RP0     ; Bank 1
                     0456             MOVLW   0xF0            ;
                     0457             MOVWF   LCD_DATA_TRIS   ; RB7 - 4 = inputs, RB3 - 0 = output
                     0458             BCF     STATUS, RP0     ; Bank 0
                     0459             RETURN
                     0460 ;
                     0461         endif
                     0462     else
                     0463 ;
                     0464 ;*************************************************************
                     0465 ;* This routine checks the busy flag, returns when not busy   *
                     0466 ;*  Affects:                                                  *
                     0467 ;*      TEMP - Returned with busy/address                     *
                     0468 ;*************************************************************
                     0469 ;
                     0470 BUSY_CHECK
0073 1683            0471             BSF     STATUS,RP0      ; Select Register page 1
0074 30FF            0472             MOVLW   0xFF            ; Set port_D for input
0075 0088            0473             MOVWF   LCD_DATA_TRIS
0076 1283            0474             BCF     STATUS, RP0     ; Select Register page 0
0077 1085            0475             BCF     LCD_CNTL, RS    ; Set LCD for command mode
0078 1505            0476             BSF     LCD_CNTL, R_W   ; Setup to read busy flag
0079 1585            0477             BSF     LCD_CNTL, E     ; Set E high
007A 1185            0478             BCF     LCD_CNTL, E     ; Set E low
007B 0808            0479             MOVF    LCD_DATA, w     ; Read busy flag, DDram address
007C 00B5            0480             MOVWF   TEMP
007D 1BB5            0481             BTFSC   TEMP, 7         ; Check busy flag, high=busy
007E 2873            0482             GOTO    BUSY_CHECK
007F 1105            0483             BCF     LCD_CNTL, R_W
0080 1683            0484             BSF     STATUS, RP0     ; Select Register page 1
0081 3000            0485             MOVLW   0x00
0082 0088            0486             MOVWF   LCD_DATA_TRIS   ; Set port_D for output
0083 1283            0487             BCF     STATUS, RP0     ; Select Register page 0
0084 0008            0488             RETURN
                     0489 ;
                     0490     endif
                     0492 ;
                     0493 Table
0085 0782            0494             addwf   PCL       ;Jump to char pointed to in W reg
0086 344D            0495             retlw   'M'
0087 3469            0496             retlw   'i'
0088 3463            0497             retlw   'c'
0089 3472            0498             retlw   'r'
008A 346F            0499             retlw   'o'
008B 3463            0500             retlw   'c'
```

```
008C 3468              retlw   'h'
008D 3469              retlw   'i'
008E 3470              retlw   'p'
008F 3420              retlw   ' '
0090 3454              retlw   'T'
0091 3465              retlw   'e'
0092 3463              retlw   'c'
0093 3468              retlw   'h'
0094 346E              retlw   'n'
0095 346F              retlw   'o'
0096 346C              retlw   'l'
0097 346F              retlw   'o'
0098 3467              retlw   'g'
0099 3479              retlw   'y'
          Table_End
009A 3400              retlw   0
          ;
                   if ( (Table & 0x0FF) >= (Table_End & 0x0FF) )
                  MESSG   "Warning - User Definded: Table Table crosses page boundry in computed jump"
               endif
          ;



               end
```

```
MPASM 00.00.68 Intermediate    LM032L.ASM   6-8-1994  0:53:47                PAGE 14


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

0000 : X-XXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0040 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX

0080 : XXXXXXXXXXXXXXXX XXXXXXXXXX-- -------- --------
00C0 : -------- -------- -------- --------

All other memory blocks unused.


Errors   :    0
Warnings :   13
```

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 C: 4-BIT DATA INTERFACE, HIGH NIBBLE LISTING

```
MPASM 00.00.68 Intermediate    LM032L.ASM   6-8-1994  0:59:12                PAGE  1


LOC  OBJECT CODE     LINE SOURCE TEXT

                     0001         LIST P=16C64, F=INHX8M
                     0002 ;
                     0003 ; This program interfaces to a Hitachi (LM032L) 2 line by 20 character display
                     0004 ; module. The program assembles for either 4-bit or 8-bit data interface, depending
                     0005 ; on the value of the 4bit flag. LCD_DATA is the port which supplies the data to
                     0006 ; the LM032L, while LCD_CNTL is the port that has the control lines ( E, RS, R_W ).
                     0007 ; In 4-bit mode the data is transfer on the high nibble of the port ( PORT<7:4> ).
                     0008 ;
                     0009 ;       Program = LM032L.ASM
                     0010 ;       Revision Date:   5-10-94
                     0011 ;
                     0012 ;
                     0013        include <C74_reg.h>
                     0233
                     0013
                     0014        include <lm032l.h>
                     0014
                     0015 ;
0001                 0016 Four_bit         EQU     TRUE       ; Selects 4- or 8-bit data transfers
0001                 0017 Data_HI          EQU     TRUE       ; If 4-bit transfers, Hi or Low nibble of PORT
                     0018 ;
                     0019 ;
                     0020     if ( Four_bit && !Data_HI )
                     0021 ;
                     0022 LCD_DATA         EQU     PORTB
                     0023 LCD_DATA_TRIS    EQU     TRISB
                     0024 ;
                     0025     else
                     0026 ;
0008                 0027 LCD_DATA         EQU     PORTD
0088                 0028 LCD_DATA_TRIS    EQU     TRISD
                     0029 ;
                     0030     endif
                     0031 ;
0005                 0032 LCD_CNTL         EQU     PORTA
                     0033 ;
                     0034 ;
```

```
                    0035 ;
                    0036 ; LCD Display Commands and Control Signal names.
                    0037 ;
                    0038     if ( Four_bit && !Data_HI )
                    0039 ;
                    0040 E                              EQU             0                               ; LCD Enable control line
                    0041 R_W                            EQU             1                               ; LCD Read/Write control line
                    0042 RS                             EQU             2                               ; LCD Register Select control
line
                    0043 ;
                    0044     else
                    0045 ;
0003                0046 E                              EQU             3                               ; LCD Enable control line
0002                0047 R_W                            EQU             2                               ; LCD Read/Write control line
0001                0048 RS                             EQU             1                               ; LCD Register Select control
line
                    0049 ;
                    0050     endif
                    0051 ;
                    0052 ;
0030                0053 TEMP1       EQU     0x030
                    0054 ;
                    0055         org     RESET_V                 ; RESET vector location
0000 2808           0056 RESET       GOTO    START               ;
                    0057 ;
                    0058 ; This is the Periperal Interrupt routine. Should NOT get here
                    0059 ;
                    0061         org     ISR_V               ; Interrupt vector location
                    0062 PER_INT_V
0004 1283           0063 ERROR1      BCF     STATUS, RP0         ; Bank 0
0005 1407           0064             BSF     PORTC, 0
0006 1007           0065             BCF     PORTC, 0
0007 2804           0066             GOTO    ERROR1
                    0067 ;
                    0068 ;
                    0069 ;
                    0070 START                              ; POWER_ON Reset (Beginning of program)
0008 0183           0071             CLRF    STATUS         ; Do initialization (Bank 0)
0009 018B           0072             CLRF    INTCON
000A 018C           0073             CLRF    PIR1
000B 1683           0074             BSF     STATUS, RP0     ; Bank 1
000C 3000           0075             MOVLW   0x00            ; The LCD module does not like to work w/ weak pull-ups
000D 0081           0076             MOVWF   OPTION_R        ;
000E 018C           0077             CLRF    PIE1            ; Disable all peripheral interrupts
000F 30FF           0078             MOVLW   0xFF            ;
0010 009F           0079             MOVWF   ADCON1          ; Port A is Digital.
                    0080 ;
                    0081 ;
```

```
0011 1283          0082            BCF     STATUS, RP0     ; Bank 0
0012 0185          0083            CLRF    PORTA           ; ALL PORT output should output Low.
0013 0186          0084            CLRF    PORTB
0014 0187          0085            CLRF    PORTC
0015 0188          0086            CLRF    PORTD
0016 0189          0087            CLRF    PORTE
0017 1010          0088            BCF     T1CON, TMR1ON   ; Timer 1 is NOT incrementing
                   0089 ;
0018 1683          0090            BSF     STATUS, RP0     ; Select Bank 1
0019 0185          0091            CLRF    TRISA           ; RA5 -  0 outputs
001A 30F0          0092            MOVLW   0xF0            ;
001B 0086          0093            MOVWF   TRISB           ; RB7 - 4 inputs, RB3 - 0 outputs
001C 0187          0094            CLRF    TRISC           ; RC Port are outputs
001D 1407          0095            BSF     TRISC, T1OSO    ; RC0 needs to be input for the oscillator to function
001E 0188          0096            CLRF    TRISD           ; RD Port are outputs
001F 0189          0097            CLRF    TRISE           ; RE Port are outputs
0020 140C          0098            BSF     PIE1, TMR1IE    ; Enable TMR1 Interrupt
0021 1781          0099            BSF     OPTION_R, RBPU  ; Disable PORTB pull-ups
0022 1283          0100            BCF     STATUS, RP0     ; Select Bank 0
                   0101 ;
                   0103 ;
                   0104 ; Initilize the LCD Display Module
                   0105 ;
0023 0185          0106            CLRF    LCD_CNTL        ; ALL PORT output should output Low.
                   0107
                   0108 DISPLAY_INIT
                   0109     if ( Four_bit && !Data_HI )
                   0110            MOVLW   0x02            ; Command for 4-bit interface low nibble
                   0111     endif
                   0112 ;
                   0113     if ( Four_bit && Data_HI )
0024 3020          0114            MOVLW   0x020           ; Command for 4-bit interface high nibble
                   0115     endif
                   0116 ;
                   0117     if ( !Four_bit )
                   0118            MOVLW   0x038           ; Command for 8-bit interface
                   0119     endif
                   0120 ;
0025 0088          0121            MOVWF   LCD_DATA        ;
0026 1585          0122            BSF     LCD_CNTL, E     ;
0027 1185          0123            BCF     LCD_CNTL, E     ;
                   0124 ;
                   0125 ; This routine takes the calculated times that the delay loop needs to
                   0126 ; be executed, based on the LCD_INIT_DELAY EQUate that includes the
                   0127 ; frequency of operation. These uses registers before they are needed to
                   0128 ; store the time.
                   0129 ;
0028 3006          0130 LCD_DELAY   MOVLW   LCD_INIT_DELAY  ;
```

```
0029 00B3           0131             MOVWF   MSD             ; Use MSD and LSD Registers to Initilize LCD
002A 01B4           0132             CLRF    LSD             ;
002B 0BB4           0133 LOOP2       DECFSZ  LSD             ; Delay time = MSD * ((3 * 256) + 3) * Tcy
002C 282B           0134             GOTO    LOOP2           ;
002D 0BB3           0135             DECFSZ  MSD             ;
                    0136 END_LCD_DELAY
002E 282B           0137             GOTO    LOOP2           ;
                    0138 ;
                    0139 ; Command sequence for 2 lines of 5x7 characters
                    0140 ;
                    0141 CMD_SEQ
                    0142 ;
                    0143     if ( Four_bit )
                    0144         if ( !Data_HI )
                    0145             MOVLW   0X02            ; 4-bit low nibble xfer
                    0146         else
002F 3020           0147             MOVLW   0X020           ; 4-bit high nibble xfer
                    0148         endif
                    0149 ;
                    0150     else                            ; 8-bit mode
                    0151             MOVLW   0X038
                    0152     endif
                    0153 ;
0030 0088           0154             MOVWF   LCD_DATA        ; This code for both 4-bit and 8-bit modes
0031 1585           0155             BSF     LCD_CNTL, E     ;
0032 1185           0156             BCF     LCD_CNTL, E     ;
                    0157 ;
                    0158     if ( Four_bit )                 ; This code for only 4-bit mode (2nd xfer)
                    0159         if ( !Data_HI )
                    0160             MOVLW   0x08            ; 4-bit low nibble xfer
                    0161         else
0033 3080           0162             MOVLW   0x080           ; 4-bit high nibble xfer
                    0163         endif
0034 0088           0164             MOVWF   LCD_DATA        ;
0035 1585           0165             BSF     LCD_CNTL, E     ;
0036 1185           0166             BCF     LCD_CNTL, E     ;
                    0167     endif
                    0168 ;
                    0169 ; Busy Flag should be valid after this point
                    0170 ;
0037 300C           0171             MOVLW   DISP_ON         ;
0038 2074           0172             CALL    SEND_CMD        ;
0039 3001           0173             MOVLW   CLR_DISP        ;
003A 2074           0174             CALL    SEND_CMD        ;
003B 3006           0175             MOVLW   ENTRY_INC       ;
003C 2074           0176             CALL    SEND_CMD        ;
003D 3080           0177             MOVLW   DD_RAM_ADDR     ;
003E 2074           0178             CALL    SEND_CMD        ;
```

```
          ;
          ;
          ;Send a message the hard way
003F 304D            movlw   'M'
0040 2065            call    SEND_CHAR
0041 3069            movlw   'i'
0042 2065            call    SEND_CHAR
0043 3063            movlw   'c'
0044 2065            call    SEND_CHAR
0045 3072            movlw   'r'
0046 2065            call    SEND_CHAR
0047 306F            movlw   'o'
0048 2065            call    SEND_CHAR
0049 3063            movlw   'c'
004A 2065            call    SEND_CHAR
004B 3068            movlw   'h'
004C 2065            call    SEND_CHAR
004D 3069            movlw   'i'
004E 2065            call    SEND_CHAR
004F 3070            movlw   'p'
0050 2065            call    SEND_CHAR

0051 30C0            movlw   B'11000000'     ;Address DDRam first character, second line
0052 2074            call    SEND_CMD

          ;Demonstration of the use of a table to output a message
0053 3000            movlw   0               ;Table address of start of message
          dispmsg
0054 00B0            movwf   TEMP1           ;TEMP1 holds start of message address
0055 209B            call    Table
0056 39FF            andlw   0FFh            ;Check if at end of message (zero
0057 1903            btfsc   STATUS,Z        ;returned at end)
0058 285D            goto    out
0059 2065            call    SEND_CHAR       ;Display character
005A 0830            movf    TEMP1,w         ;Point to next character
005B 3E01            addlw   1
005C 2854            goto    dispmsg
          out
          loop
005D 285D            goto    loop            ;Stay here forever
          ;
          ;
          INIT_DISPLAY
005E 300C            MOVLW   DISP_ON             ; Display On, Cursor On
005F 2074            CALL    SEND_CMD            ; Send This command to the Display Module
0060 3001            MOVLW   CLR_DISP            ; Clear the Display
0061 2074            CALL    SEND_CMD            ; Send This command to the Display Module
0062 3006            MOVLW   ENTRY_INC           ; Set Entry Mode Inc., No shift
```

```
0063 2074           0228            CALL    SEND_CMD            ; Send This command to the Display Module
0064 0008           0229            RETURN
                    0230 ;
                    0231 ;
                    0232 ;*****************************************************************
                    0233 ;* The LCD Module Subroutines                                    *
                    0234 ;*****************************************************************
                    0235 ;
                    0236     if ( Four_bit )      ; 4-bit Data transfers?
                    0237 ;
                    0238         if ( Data_HI )   ; 4-bit transfers on the high nibble of the PORT
                    0239 ;
                    0240 ;*****************************************************************
                    0241 ;*SendChar - Sends character to LCD                            *
                    0242 ;*This routine splits the character into the upper and lower    *
                    0243 ;*nibbles and sends them to the LCD, upper nibble first.        *
                    0244 ;*****************************************************************
                    0245 ;
                    0246 SEND_CHAR
0065 00B6           0247            MOVWF   CHAR            ;Character to be sent is in W
0066 2083           0248            CALL    BUSY_CHECK      ;Wait for LCD to be ready
0067 0836           0249            MOVF    CHAR, w
0068 39F0           0250            ANDLW   0xF0            ;Get upper nibble
0069 0088           0251            MOVWF   LCD_DATA        ;Send data to LCD
006A 1105           0252            BCF     LCD_CNTL, R_W   ;Set LCD to read
006B 1485           0253            BSF     LCD_CNTL, RS    ;Set LCD to data mode
006C 1585           0254            BSF     LCD_CNTL, E     ;toggle E for LCD
006D 1185           0255            BCF     LCD_CNTL, E
006E 0E36           0256            SWAPF   CHAR, w
006F 39F0           0257            ANDLW   0xF0            ;Get lower nibble
0070 0088           0258            MOVWF   LCD_DATA        ;Send data to LCD
0071 1585           0259            BSF     LCD_CNTL, E     ;toggle E for LCD
0072 1185           0260            BCF     LCD_CNTL, E
0073 0008           0261            RETURN
                    0262 ;
                    0263         else            ; 4-bit transfers on the low nibble of the PORT
                    0264 ;
                    0265 ;*****************************************************************
                    0266 ;* SEND_CHAR - Sends character to LCD                          *
                    0267 ;* This routine splits the character into the upper and lower   *
                    0268 ;* nibbles and sends them to the LCD, upper nibble first.       *
                    0269 ;* The data is transmitted on the PORT<3:0> pins                *
                    0270 ;*****************************************************************
                    0271 ;
                    0272 SEND_CHAR
                    0273            MOVWF   CHAR            ; Character to be sent is in W
                    0274            CALL    BUSY_CHECK      ; Wait for LCD to be ready
                    0275            SWAPF   CHAR, W
```

```
            ANDLW   0x0F             ; Get upper nibble
            MOVWF   LCD_DATA         ; Send data to LCD
            BCF     LCD_CNTL, R_W    ; Set LCD to read
            BSF     LCD_CNTL, RS     ; Set LCD to data mode
            BSF     LCD_CNTL, E      ; toggle E for LCD
            BCF     LCD_CNTL, E
            MOVF    CHAR, W
            ANDLW   0x0F             ; Get lower nibble
            MOVWF   LCD_DATA         ; Send data to LCD
            BSF     LCD_CNTL, E      ; toggle E for LCD
            BCF     LCD_CNTL, E
            RETURN
;
        endif
    else
;
;*****************************************************************
;* SEND_CHAR - Sends character contained in register W to LCD    *
;* This routine sends the entire character to the PORT           *
;* The data is transmitted on the PORT<7:0> pins                 *
;*****************************************************************
;
SEND_CHAR
            MOVWF   CHAR             ; Character to be sent is in W
            CALL    BUSY_CHECK       ; Wait for LCD to be ready
            MOVF    CHAR, w
            MOVWF   LCD_DATA         ; Send data to LCD
            BCF     LCD_CNTL, R_W    ; Set LCD in read mode
            BSF     LCD_CNTL, RS     ; Set LCD in data mode
            BSF     LCD_CNTL, E      ; toggle E for LCD
            BCF     LCD_CNTL, E
            RETURN
;
    endif
;
;
;*******************************************************************
;* 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                   *
;*******************************************************************
;
    if ( Four_bit )      ; 4-bit Data transfers?
;
        if ( Data_HI )   ; 4-bit transfers on the high nibble of the PORT
;
;*******************************************************************
```

```
          ;* SEND_CMD - Sends command to LCD                                   *
          ;* This routine splits the command into the upper and lower         *
          ;* nibbles and sends them to the LCD, upper nibble first.           *
          ;*********************************************************************

          SEND_CMD
0074 00B6               MOVWF   CHAR            ; Character to be sent is in W
0075 2083               CALL    BUSY_CHECK      ; Wait for LCD to be ready
0076 0836               MOVF    CHAR,w
0077 39F0               ANDLW   0xF0            ; Get upper nibble
0078 0088               MOVWF   LCD_DATA        ; Send data to LCD
0079 1105               BCF     LCD_CNTL,R_W    ; Set LCD to read
007A 1085               BCF     LCD_CNTL,RS     ; Set LCD to command mode
007B 1585               BSF     LCD_CNTL,E      ; toggle E for LCD
007C 1185               BCF     LCD_CNTL,E
007D 0E36               SWAPF   CHAR,w
007E 39F0               ANDLW   0xF0            ; Get lower nibble
007F 0088               MOVWF   LCD_DATA        ; Send data to LCD
0080 1585               BSF     LCD_CNTL,E      ; toggle E for LCD
0081 1185               BCF     LCD_CNTL,E
0082 0008               RETURN
          ;
                  else             ; 4-bit transfers on the low nibble of the PORT
          ;
          SEND_CMD
                        MOVWF   CHAR            ; Character to be sent is in W
                        CALL    BUSY_CHECK      ; Wait for LCD to be ready
                        SWAPF   CHAR, W
                        ANDLW   0x0F            ; Get upper nibble
                        MOVWF   LCD_DATA        ; Send data to LCD
                        BCF     LCD_CNTL, R_W   ; Set LCD to read
                        BCF     LCD_CNTL, RS    ; Set LCD to command mode
                        BSF     LCD_CNTL, E     ; toggle E for LCD
                        BCF     LCD_CNTL, E
                        MOVF    CHAR, W
                        ANDLW   0x0F            ; Get lower nibble
                        MOVWF   LCD_DATA        ; Send data to LCD
                        BSF     LCD_CNTL, E     ; toggle E for LCD
                        BCF     LCD_CNTL, E
                        RETURN
          ;
                  endif
              else
          ;
          ;***************************************************************
          ;* SEND_CND - Sends command contained in register W to LCD    *
          ;* This routine sends the entire character to the PORT        *
          ;* The data is transmitted on the PORT<7:0> pins              *
```

```
          ;*****************************************************************
          SEND_CMD
                      MOVWF   CHAR             ; Command to be sent is in W
                      CALL    BUSY_CHECK       ; Wait for LCD to be ready
                      MOVF    CHAR, w
                      MOVWF   LCD_DATA         ; Send data to LCD
                      BCF     LCD_CNTL, R_W    ; Set LCD in read mode
                      BCF     LCD_CNTL, RS     ; Set LCD in command mode
                      BSF     LCD_CNTL, E      ; toggle E for LCD
                      BCF     LCD_CNTL, E
                      RETURN
          ;
              endif
          ;
          ;
              if ( Four_bit )       ; 4-bit Data transfers?
          ;
                  if ( Data_HI )    ; 4-bit transfers on the high nibble of the PORT
          ;
          ;********************************************************************
          ;* This routine checks the busy flag, returns when not busy        *
          ;*  Affects:                                                        *
          ;*      TEMP - Returned with busy/address                           *
          ;********************************************************************
          ;
          BUSY_CHECK
0083 1683             BSF     STATUS, RP0      ; Select Register page 1
0084 30FF             MOVLW   0xFF             ; Set Port_D for input
0085 0088             MOVWF   LCD_DATA_TRIS
0086 1283             BCF     STATUS, RP0      ; Select Register page 0
0087 1085             BCF     LCD_CNTL, RS     ; Set LCD for Command mode
0088 1505             BSF     LCD_CNTL, R_W    ; Setup to read busy flag
0089 1585             BSF     LCD_CNTL, E      ; Set E high
008A 1185             BCF     LCD_CNTL, E      ; Set E low
008B 0808             MOVF    LCD_DATA, W      ; Read upper nibble busy flag, DDRam address
008C 39F0             ANDLW   0xF0             ; Mask out lower nibble
008D 00B5             MOVWF   TEMP
008E 1585             BSF     LCD_CNTL, E      ; Toggle E to get lower nibble
008F 1185             BCF     LCD_CNTL, E
0090 0E08             SWAPF   LCD_DATA, w      ; Read lower nibble busy flag, DDRam address
0091 390F             ANDLW   0x0F             ; Mask out upper nibble
0092 04B5             IORWF   TEMP             ; Combine nibbles
0093 1BB5             BTFSC   TEMP, 7          ; Check busy flag, high = busy
0094 2883             GOTO    BUSY_CHECK       ; If busy, check again
0095 1105             BCF     LCD_CNTL, R_W
0096 1683             BSF     STATUS, RP0      ; Select Register page 1
0097 300F             MOVLW   0x0F
```

```
0098 0088        MOVWF   LCD_DATA_TRIS   ; Set Port_D for output
0099 1283        BCF     STATUS, RP0     ; Select Register page 0
009A 0008        RETURN
           ;
               else             ; 4-bit transfers on the low nibble of the PORT
           ;
           ;*******************************************************************
           ;* This routine checks the busy flag, returns when not busy        *
           ;*  Affects:                                                       *
           ;*      TEMP - Returned with busy/address                          *
           ;*******************************************************************
           ;
           BUSY_CHECK
                 BSF     STATUS, RP0     ; Bank 1
                 MOVLW   0xFF            ; Set PortB for input
                 MOVWF   LCD_DATA_TRIS
                 BCF     STATUS, RP0     ; Bank 0
                 BCF     LCD_CNTL, RS    ; Set LCD for Command mode
                 BSF     LCD_CNTL, R_W   ; Setup to read busy flag
                 BSF     LCD_CNTL, E     ; Set E high
                 BCF     LCD_CNTL, E     ; Set E low
                 SWAPF   LCD_DATA, W     ; Read upper nibble busy flag, DDRam address
                 ANDLW   0xF0            ; Mask out lower nibble
                 MOVWF   TEMP            ;
                 BSF     LCD_CNTL, E     ; Toggle E to get lower nibble
                 BCF     LCD_CNTL, E
                 MOVF    LCD_DATA, W     ; Read lower nibble busy flag, DDRam address
                 ANDLW   0x0F            ; Mask out upper nibble
                 IORWF   TEMP, F         ; Combine nibbles
                 BTFSC   TEMP, 7         ; Check busy flag, high = busy
                 GOTO    BUSY_CHECK      ; If busy, check again
                 BCF     LCD_CNTL, R_W
                 BSF     STATUS, RP0     ; Bank 1
                 MOVLW   0xF0            ;
                 MOVWF   LCD_DATA_TRIS   ; RB7 - 4 = inputs, RB3 - 0 = output
                 BCF     STATUS, RP0     ; Bank 0
                 RETURN
           ;
               endif
           else
           ;
           ;**************************************************************
           ;* This routine checks the busy flag, returns when not busy   *
           ;*  Affects:                                                  *
           ;*      TEMP - Returned with busy/address                     *
           ;**************************************************************
           ;
           BUSY_CHECK
```

```
                  BSF     STATUS,RP0      ; Select Register page 1
                  MOVLW   0xFF            ; Set port_D for input
                  MOVWF   LCD_DATA_TRIS
                  BCF     STATUS, RP0     ; Select Register page 0
                  BCF     LCD_CNTL, RS    ; Set LCD for command mode
                  BSF     LCD_CNTL, R_W   ; Setup to read busy flag
                  BSF     LCD_CNTL, E     ; Set E high
                  BCF     LCD_CNTL, E     ; Set E low
                  MOVF    LCD_DATA, w     ; Read busy flag, DDram address
                  MOVWF   TEMP
                  BTFSC   TEMP, 7         ; Check busy flag, high=busy
                  GOTO    BUSY_CHECK
                  BCF     LCD_CNTL, R_W
                  BSF     STATUS, RP0     ; Select Register page 1
                  MOVLW   0x00
                  MOVWF   LCD_DATA_TRIS   ; Set port_D for output
                  BCF     STATUS, RP0     ; Select Register page 0
                  RETURN
           ;
               endif
           ;
           Table
009B 0782         addwf   PCL       ;Jump to char pointed to in W reg
009C 344D         retlw   'M'
009D 3469         retlw   'i'
009E 3463         retlw   'c'
009F 3472         retlw   'r'
00A0 346F         retlw   'o'
00A1 3463         retlw   'c'
00A2 3468         retlw   'h'
00A3 3469         retlw   'i'
00A4 3470         retlw   'p'
00A5 3420         retlw   ' '
00A6 3454         retlw   'T'
00A7 3465         retlw   'e'
00A8 3463         retlw   'c'
00A9 3468         retlw   'h'
00AA 346E         retlw   'n'
00AB 346F         retlw   'o'
00AC 346C         retlw   'l'
00AD 346F         retlw   'o'
00AE 3467         retlw   'g'
00AF 3479         retlw   'y'
           Table_End
00B0 3400         retlw   0
           ;
                   if ( (Table & 0x0FF) >= (Table_End & 0x0FF) )
                  MESSG   "Warning - User Definded: Table Table crosses page boundry in computed jump"
```

```
                      0520     endif
                      0521 ;
                      0522
                      0523
                      0524
                      0525     end
                      0526
                      0527
                      0528
                      0529
                      0530



MPASM 00.00.68 Intermediate    LM032L.ASM   6-8-1994  0:59:12                PAGE 14


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

0000 : X—XXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX
0040 : XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX

0080 : XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX X———————
00C0 : ———————— ———————— ———————— ————————

All other memory blocks unused.


Errors   :    0
Warnings :   13
```

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 D: 4-BIT DATA INTERFACE, LOW NIBBLE LISTING

```
MPASM 00.00.68 Intermediate    LM032L.ASM   6-8-1994  5:29:26                PAGE  1


LOC  OBJECT CODE     LINE SOURCE TEXT

                     0001         LIST P=16C64, F=INHX8M
                     0002 ;
                     0003 ; This program interfaces to a Hitachi (LM032L) 2 line by 20 character display
                     0004 ; module. The program assembles for either 4-bit or 8-bit data interface, depending
                     0005 ; on the value of the 4bit flag. LCD_DATA is the port which supplies the data to
                     0006 ; the LM032L, while LCD_CNTL is the port that has the control lines ( E, RS, R_W ).
                     0007 ; In 4-bit mode the data is transfer on the high nibble of the port ( PORT<7:4> ).
                     0008 ;
                     0009 ;       Program = LM032L.ASM
                     0010 ;       Revision Date:   5-10-94
                     0011 ;
                     0012 ;
                     0013        include <C74_reg.h>
                     0233
                     0013
                     0014        include <lm032l.h>
                     0014
                     0015 ;
0001                 0016 Four_bit         EQU     TRUE       ; Selects 4- or 8-bit data transfers
0000                 0017 Data_HI          EQU     FALSE       ; If 4-bit transfers, Hi or Low nibble of PORT
                     0018 ;
                     0019 ;
                     0020     if ( Four_bit && !Data_HI )
                     0021 ;
0006                 0022 LCD_DATA         EQU     PORTB
0086                 0023 LCD_DATA_TRIS    EQU     TRISB
                     0024 ;
                     0025     else
                     0026 ;
                     0027 LCD_DATA         EQU     PORTD
                     0028 LCD_DATA_TRIS    EQU     TRISD
                     0029 ;
                     0030     endif
                     0031 ;
0005                 0032 LCD_CNTL         EQU     PORTA
                     0033 ;
                     0034 ;
                     0035 ;
```

```
                    0036 ; LCD Display Commands and Control Signal names.
                    0037 ;
                    0038     if ( Four_bit && !Data_HI )
                    0039 ;
0000                0040 E                               EQU             0                               ; LCD Enable control line
0001                0041 R_W                             EQU             1                               ; LCD Read/Write control line
0002                0042 RS                              EQU             2                               ; LCD Register Select control
line
                    0043 ;
                    0044     else
                    0045 ;
                    0046 E                               EQU             3                               ; LCD Enable control line
                    0047 R_W                             EQU             2                               ; LCD Read/Write control line
                    0048 RS                              EQU             1                               ; LCD Register Select control
line
                    0049 ;
                    0050     endif
                    0051 ;
                    0052 ;
0030                0053 TEMP1       EQU     0x030
                    0054 ;
                    0055         org     RESET_V                 ; RESET vector location
0000 2808           0056 RESET       GOTO    START               ;
                    0057 ;
                    0058 ; This is the Peripheral Interrupt routine. Should NOT get here
                    0059 ;
                    0061          org     ISR_V               ; Interrupt vector location
                    0062 PER_INT_V
0004 1283           0063 ERROR1      BCF     STATUS, RP0         ; Bank 0
0005 1407           0064             BSF     PORTC, 0
0006 1007           0065             BCF     PORTC, 0
0007 2804           0066             GOTO    ERROR1
                    0067 ;
                    0068 ;
                    0069 ;
                    0070 START                                ; POWER_ON Reset (Beginning of program)
0008 0183           0071             CLRF    STATUS          ; Do initialization (Bank 0)
0009 018B           0072             CLRF    INTCON
000A 018C           0073             CLRF    PIR1
000B 1683           0074             BSF     STATUS, RP0     ; Bank 1
000C 3000           0075             MOVLW   0x00            ; The LCD module does not like to work w/ weak pull-ups
000D 0081           0076             MOVWF   OPTION_R        ;
000E 018C           0077             CLRF    PIE1            ; Disable all peripheral interrupts
000F 30FF           0078             MOVLW   0xFF            ;
0010 009F           0079             MOVWF   ADCON1          ; Port A is Digital.
                    0080 ;
                    0081 ;
0011 1283           0082             BCF     STATUS, RP0     ; Bank 0
```

```
0012 0185                  CLRF    PORTA            ; ALL PORT output should output Low.
0013 0186                  CLRF    PORTB
0014 0187                  CLRF    PORTC
0015 0188                  CLRF    PORTD
0016 0189                  CLRF    PORTE
0017 1010                  BCF     T1CON, TMR1ON    ; Timer 1 is NOT incrementing
                      ;
0018 1683                  BSF     STATUS, RP0      ; Select Bank 1
0019 0185                  CLRF    TRISA            ; RA5 -  0 outputs
001A 30F0                  MOVLW   0xF0             ;
001B 0086                  MOVWF   TRISB            ; RB7 - 4 inputs, RB3 - 0 outputs
001C 0187                  CLRF    TRISC            ; RC Port are outputs
001D 1407                  BSF     TRISC, T1OSO     ; RC0 needs to be input for the oscillator to function
001E 0188                  CLRF    TRISD            ; RD Port are outputs
001F 0189                  CLRF    TRISE            ; RE Port are outputs
0020 140C                  BSF     PIE1, TMR1IE     ; Enable TMR1 Interrupt
0021 1781                  BSF     OPTION_R, RBPU   ; Disable PORTB pull-ups
0022 1283                  BCF     STATUS, RP0      ; Select Bank 0
                      ;
                      ;
                      ; Initilize the LCD Display Module
                      ;
0023 0185                  CLRF    LCD_CNTL         ; ALL PORT output should output Low.

                      DISPLAY_INIT
                          if ( Four_bit && !Data_HI )
0024 3002                  MOVLW   0x02             ; Command for 4-bit interface low nibble
                          endif
                      ;
                          if ( Four_bit && Data_HI )
                           MOVLW   0x020            ; Command for 4-bit interface high nibble
                          endif
                      ;
                          if ( !Four_bit )
                           MOVLW   0x038            ; Command for 8-bit interface
                          endif
                      ;
0025 0086                  MOVWF   LCD_DATA         ;
0026 1405                  BSF     LCD_CNTL, E      ;
0027 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.
                      ;
0028 3006             LCD_DELAY   MOVLW   LCD_INIT_DELAY  ;
0029 00B3                  MOVWF   MSD              ; Use MSD and LSD Registers to Initilize LCD
```

```
002A 01B4           0132             CLRF    LSD             ;
002B 0BB4           0133 LOOP2       DECFSZ  LSD             ; Delay time = MSD * ((3 * 256) + 3) * Tcy
002C 282B           0134             GOTO    LOOP2           ;
002D 0BB3           0135             DECFSZ  MSD             ;
                    0136 END_LCD_DELAY
002E 282B           0137             GOTO    LOOP2           ;
                    0138 ;
                    0139 ; Command sequence for 2 lines of 5x7 characters
                    0140 ;
                    0141 CMD_SEQ
                    0142 ;
                    0143     if ( Four_bit )
                    0144         if ( !Data_HI )
002F 3002           0145             MOVLW   0X02            ; 4-bit low nibble xfer
                    0146         else
                    0147             MOVLW   0X020           ; 4-bit high nibble xfer
                    0148         endif
                    0149 ;
                    0150     else                            ; 8-bit mode
                    0151             MOVLW   0X038
                    0152     endif
                    0153 ;
0030 0086           0154             MOVWF   LCD_DATA        ; This code for both 4-bit and 8-bit modes
0031 1405           0155             BSF     LCD_CNTL, E     ;
0032 1005           0156             BCF     LCD_CNTL, E     ;
                    0157 ;
                    0158     if ( Four_bit )                 ; This code for only 4-bit mode (2nd xfer)
                    0159         if ( !Data_HI )
0033 3008           0160             MOVLW   0x08            ; 4-bit low nibble xfer
                    0161         else
                    0162             MOVLW   0x080           ; 4-bit high nibble xfer
                    0163         endif
0034 0086           0164             MOVWF   LCD_DATA        ;
0035 1405           0165             BSF     LCD_CNTL, E     ;
0036 1005           0166             BCF     LCD_CNTL, E     ;
                    0167     endif
                    0168 ;
                    0169 ; Busy Flag should be valid after this point
                    0170 ;
0037 300C           0171             MOVLW   DISP_ON         ;
0038 2074           0172             CALL    SEND_CMD        ;
0039 3001           0173             MOVLW   CLR_DISP        ;
003A 2074           0174             CALL    SEND_CMD        ;
003B 3006           0175             MOVLW   ENTRY_INC       ;
003C 2074           0176             CALL    SEND_CMD        ;
003D 3080           0177             MOVLW   DD_RAM_ADDR     ;
003E 2074           0178             CALL    SEND_CMD        ;
                    0179 ;
```

```
                        0181 ;
                        0182 ;Send a message the hard way
003F 304D               0183            movlw   'M'
0040 2065               0184            call    SEND_CHAR
0041 3069               0185            movlw   'i'
0042 2065               0186            call    SEND_CHAR
0043 3063               0187            movlw   'c'
0044 2065               0188            call    SEND_CHAR
0045 3072               0189            movlw   'r'
0046 2065               0190            call    SEND_CHAR
0047 306F               0191            movlw   'o'
0048 2065               0192            call    SEND_CHAR
0049 3063               0193            movlw   'c'
004A 2065               0194            call    SEND_CHAR
004B 3068               0195            movlw   'h'
004C 2065               0196            call    SEND_CHAR
004D 3069               0197            movlw   'i'
004E 2065               0198            call    SEND_CHAR
004F 3070               0199            movlw   'p'
0050 2065               0200            call    SEND_CHAR
                        0201
0051 30C0               0202            movlw   B'11000000'     ;Address DDRam first character, second line
0052 2074               0203            call    SEND_CMD
                        0204
                        0205 ;Demonstration of the use of a table to output a message
0053 3000               0206            movlw   0               ;Table address of start of message
                        0207 dispmsg
0054 00B0               0208            movwf   TEMP1           ;TEMP1 holds start of message address
0055 209B               0209            call    Table
0056 39FF               0210            andlw   0FFh            ;Check if at end of message (zero
0057 1903               0211            btfsc   STATUS,Z        ;returned at end)
0058 285D               0212            goto    out
0059 2065               0213            call    SEND_CHAR       ;Display character
005A 0830               0214            movf    TEMP1,w         ;Point to next character
005B 3E01               0215            addlw   1
005C 2854               0216            goto    dispmsg
                        0217 out
                        0218 loop
005D 285D               0219            goto    loop            ;Stay here forever
                        0220 ;
                        0221 ;
                        0222 INIT_DISPLAY
005E 300C               0223            MOVLW   DISP_ON             ; Display On, Cursor On
005F 2074               0224            CALL    SEND_CMD            ; Send This command to the Display Module
0060 3001               0225            MOVLW   CLR_DISP            ; Clear the Display
0061 2074               0226            CALL    SEND_CMD            ; Send This command to the Display Module
0062 3006               0227            MOVLW   ENTRY_INC           ; Set Entry Mode Inc., No shift
0063 2074               0228            CALL    SEND_CMD            ; Send This command to the Display Module
```

```
0064 0008           0229            RETURN
                    0230 ;
                    0232 ;
                    0233 ;*****************************************************************
                    0234 ;* The LCD Module Subroutines                                    *
                    0235 ;*****************************************************************
                    0236 ;
                    0237    if ( Four_bit )      ; 4-bit Data transfers?
                    0238 ;
                    0239        if ( Data_HI )   ; 4-bit transfers on the high nibble of the PORT
                    0240 ;
                    0241 ;*****************************************************************
                    0242 ;*SendChar - Sends character to LCD                             *
                    0243 ;*This routine splits the character into the upper and lower    *
                    0244 ;*nibbles and sends them to the LCD, upper nibble first.        *
                    0245 ;*****************************************************************
                    0246 ;
                    0247 SEND_CHAR
                    0248            MOVWF   CHAR            ;Character to be sent is in W
                    0249            CALL    BUSY_CHECK      ;Wait for LCD to be ready
                    0250            MOVF    CHAR, w
                    0251            ANDLW   0xF0            ;Get upper nibble
                    0252            MOVWF   LCD_DATA        ;Send data to LCD
                    0253            BCF     LCD_CNTL, R_W   ;Set LCD to read
                    0254            BSF     LCD_CNTL, RS    ;Set LCD to data mode
                    0255            BSF     LCD_CNTL, E     ;toggle E for LCD
                    0256            BCF     LCD_CNTL, E
                    0257            SWAPF   CHAR, w
                    0258            ANDLW   0xF0            ;Get lower nibble
                    0259            MOVWF   LCD_DATA        ;Send data to LCD
                    0260            BSF     LCD_CNTL, E     ;toggle E for LCD
                    0261            BCF     LCD_CNTL, E
                    0262            RETURN
                    0263 ;
                    0264        else            ; 4-bit transfers on the low nibble of the PORT
                    0265 ;
                    0266 ;*****************************************************************
                    0267 ;* SEND_CHAR - Sends character to LCD                           *
                    0268 ;* This routine splits the character into the upper and lower    *
                    0269 ;* nibbles and sends them to the LCD, upper nibble first.       *
                    0270 ;* The data is transmitted on the PORT<3:0> pins                *
                    0271 ;*****************************************************************
                    0272 ;
                    0273 SEND_CHAR
0065 00B6           0274            MOVWF   CHAR            ; Character to be sent is in W
0066 2083           0275            CALL    BUSY_CHECK      ; Wait for LCD to be ready
0067 0E36           0276            SWAPF   CHAR, W
0068 390F           0277            ANDLW   0x0F            ; Get upper nibble
```

```
0069 0086           0278            MOVWF   LCD_DATA        ; Send data to LCD
006A 1085           0279            BCF     LCD_CNTL, R_W   ; Set LCD to read
006B 1505           0280            BSF     LCD_CNTL, RS    ; Set LCD to data mode
006C 1405           0281            BSF     LCD_CNTL, E     ; toggle E for LCD
006D 1005           0282            BCF     LCD_CNTL, E
006E 0836           0283            MOVF    CHAR, W
006F 390F           0284            ANDLW   0x0F            ; Get lower nibble
0070 0086           0285            MOVWF   LCD_DATA        ; Send data to LCD
0071 1405           0286            BSF     LCD_CNTL, E     ; toggle E for LCD
0072 1005           0287            BCF     LCD_CNTL, E
0073 0008           0288            RETURN
                    0289 ;
                    0290        endif
                    0291    else
                    0292 ;
                    0293 ;*****************************************************************
                    0294 ;* SEND_CHAR - Sends character contained in register W to LCD    *
                    0295 ;* This routine sends the entire character to the PORT           *
                    0296 ;* The data is transmitted on the PORT<7:0> pins                 *
                    0297 ;*****************************************************************
                    0298 ;
                    0299 SEND_CHAR
                    0300            MOVWF   CHAR            ; Character to be sent is in W
                    0301            CALL    BUSY_CHECK      ; Wait for LCD to be ready
                    0302            MOVF    CHAR, w
                    0303            MOVWF   LCD_DATA        ; Send data to LCD
                    0304            BCF     LCD_CNTL, R_W   ; Set LCD in read mode
                    0305            BSF     LCD_CNTL, RS    ; Set LCD in data mode
                    0306            BSF     LCD_CNTL, E     ; toggle E for LCD
                    0307            BCF     LCD_CNTL, E
                    0308            RETURN
                    0309 ;
                    0310    endif
                    0311 ;
                    0313 ;
                    0314 ;*****************************************************************
                    0315 ;* SendCmd - Sends command to LCD                               *
                    0316 ;* This routine splits the command into the upper and lower      *
                    0317 ;* nibbles and sends them to the LCD, upper nibble first.        *
                    0318 ;* The data is transmitted on the PORT<3:0> pins                 *
                    0319 ;*****************************************************************
                    0320 ;
                    0321    if ( Four_bit )      ; 4-bit Data transfers?
                    0322 ;
                    0323        if ( Data_HI )   ; 4-bit transfers on the high nibble of the PORT
                    0324 ;
                    0325 ;*****************************************************************
```

```
          ;* SEND_CMD - Sends command to LCD                                *
          ;* This routine splits the command into the upper and lower       *
          ;* nibbles and sends them to the LCD, upper nibble first.         *
          ;*******************************************************************

          SEND_CMD
                      MOVWF   CHAR            ; Character to be sent is in W
                      CALL    BUSY_CHECK      ; Wait for LCD to be ready
                      MOVF    CHAR,w
                      ANDLW   0xF0            ; Get upper nibble
                      MOVWF   LCD_DATA        ; Send data to LCD
                      BCF     LCD_CNTL,R_W    ; Set LCD to read
                      BCF     LCD_CNTL,RS     ; Set LCD to command mode
                      BSF     LCD_CNTL,E      ; toggle E for LCD
                      BCF     LCD_CNTL,E
                      SWAPF   CHAR,w
                      ANDLW   0xF0            ; Get lower nibble
                      MOVWF   LCD_DATA        ; Send data to LCD
                      BSF     LCD_CNTL,E      ; toggle E for LCD
                      BCF     LCD_CNTL,E
                      RETURN
          ;
                  else             ; 4-bit transfers on the low nibble of the PORT
          ;
          SEND_CMD
0074 00B6             MOVWF   CHAR            ; Character to be sent is in W
0075 2083             CALL    BUSY_CHECK      ; Wait for LCD to be ready
0076 0E36             SWAPF   CHAR, W
0077 390F             ANDLW   0x0F            ; Get upper nibble
0078 0086             MOVWF   LCD_DATA        ; Send data to LCD
0079 1085             BCF     LCD_CNTL, R_W   ; Set LCD to read
007A 1105             BCF     LCD_CNTL, RS    ; Set LCD to command mode
007B 1405             BSF     LCD_CNTL, E     ; toggle E for LCD
007C 1005             BCF     LCD_CNTL, E
007D 0836             MOVF    CHAR, W
007E 390F             ANDLW   0x0F            ; Get lower nibble
007F 0086             MOVWF   LCD_DATA        ; Send data to LCD
0080 1405             BSF     LCD_CNTL, E     ; toggle E for LCD
0081 1005             BCF     LCD_CNTL, E
0082 0008             RETURN
          ;
                  endif
              else
          ;
          ;**************************************************************
          ;* SEND_CND - Sends command contained in register W to LCD    *
          ;* This routine sends the entire character to the PORT        *
          ;* The data is transmitted on the PORT<7:0> pins              *
```

```
0374 ;*****************************************************************
0375
0376 SEND_CMD
0377             MOVWF   CHAR            ; Command to be sent is in W
0378             CALL    BUSY_CHECK      ; Wait for LCD to be ready
0379             MOVF    CHAR, w
0380             MOVWF   LCD_DATA        ; Send data to LCD
0381             BCF     LCD_CNTL, R_W   ; Set LCD in read mode
0382             BCF     LCD_CNTL, RS    ; Set LCD in command mode
0383             BSF     LCD_CNTL, E     ; toggle E for LCD
0384             BCF     LCD_CNTL, E
0385             RETURN
0386 ;
0387     endif
0388 ;
0390 ;
0391     if ( Four_bit )      ; 4-bit Data transfers?
0392 ;
0393         if ( Data_HI )   ; 4-bit transfers on the high nibble of the PORT
0394 ;
0395 ;*********************************************************************
0396 ;* This routine checks the busy flag, returns when not busy        *
0397 ;*  Affects:                                                       *
0398 ;*      TEMP - Returned with busy/address                          *
0399 ;*********************************************************************
0400 ;
0401 BUSY_CHECK
0402             BSF     STATUS, RP0     ; Select Register page 1
0403             MOVLW   0xFF            ; Set Port_D for input
0404             MOVWF   LCD_DATA_TRIS
0405             BCF     STATUS, RP0     ; Select Register page 0
0406             BCF     LCD_CNTL, RS    ; Set LCD for Command mode
0407             BSF     LCD_CNTL, R_W   ; Setup to read busy flag
0408             BSF     LCD_CNTL, E     ; Set E high
0409             BCF     LCD_CNTL, E     ; Set E low
0410             MOVF    LCD_DATA, W     ; Read upper nibble busy flag, DDRam address
0411             ANDLW   0xF0            ; Mask out lower nibble
0412             MOVWF   TEMP
0413             BSF     LCD_CNTL, E     ; Toggle E to get lower nibble
0414             BCF     LCD_CNTL, E
0415             SWAPF   LCD_DATA, w     ; Read lower nibble busy flag, DDRam address
0416             ANDLW   0x0F            ; Mask out upper nibble
0417             IORWF   TEMP            ; Combine nibbles
0418             BTFSC   TEMP, 7         ; Check busy flag, high = busy
0419             GOTO    BUSY_CHECK      ; If busy, check again
0420             BCF     LCD_CNTL, R_W
0421             BSF     STATUS, RP0     ; Select Register page 1
0422             MOVLW   0x0F
```

```
                      MOVWF   LCD_DATA_TRIS   ; Set Port_D for output
                      BCF     STATUS, RP0     ; Select Register page 0
                      RETURN
           ;
                  else            ; 4-bit transfers on the low nibble of the PORT
           ;
           ;*******************************************************************
           ;* This routine checks the busy flag, returns when not busy        *
           ;*  Affects:                                                       *
           ;*      TEMP - Returned with busy/address                          *
           ;*******************************************************************
           ;
           BUSY_CHECK
0083 1683             BSF     STATUS, RP0     ; Bank 1
0084 30FF             MOVLW   0xFF            ; Set PortB for input
0085 0086             MOVWF   LCD_DATA_TRIS
0086 1283             BCF     STATUS, RP0     ; Bank 0
0087 1105             BCF     LCD_CNTL, RS    ; Set LCD for Command mode
0088 1485             BSF     LCD_CNTL, R_W   ; Setup to read busy flag
0089 1405             BSF     LCD_CNTL, E     ; Set E high
008A 1005             BCF     LCD_CNTL, E     ; Set E low
008B 0E06             SWAPF   LCD_DATA, W     ; Read upper nibble busy flag, DDRam address
008C 39F0             ANDLW   0xF0            ; Mask out lower nibble
008D 00B5             MOVWF   TEMP            ;
008E 1405             BSF     LCD_CNTL, E     ; Toggle E to get lower nibble
008F 1005             BCF     LCD_CNTL, E
0090 0806             MOVF    LCD_DATA, W     ; Read lower nibble busy flag, DDRam address
0091 390F             ANDLW   0x0F            ; Mask out upper nibble
0092 04B5             IORWF   TEMP, F         ; Combine nibbles
0093 1BB5             BTFSC   TEMP, 7         ; Check busy flag, high = busy
0094 2883             GOTO    BUSY_CHECK      ; If busy, check again
0095 1085             BCF     LCD_CNTL, R_W
0096 1683             BSF     STATUS, RP0     ; Bank 1
0097 30F0             MOVLW   0xF0            ;
0098 0086             MOVWF   LCD_DATA_TRIS   ; RB7 - 4 = inputs, RB3 - 0 = output
0099 1283             BCF     STATUS, RP0     ; Bank 0
009A 0008             RETURN
           ;
                  endif
              else
           ;
           ;**************************************************************
           ;* This routine checks the busy flag, returns when not busy   *
           ;*  Affects:                                                  *
           ;*      TEMP - Returned with busy/address                     *
           ;**************************************************************
           ;
           BUSY_CHECK
```

3

```
                      BSF     STATUS,RP0      ; Select Register page 1
                      MOVLW   0xFF            ; Set port_D for input
                      MOVWF   LCD_DATA_TRIS
                      BCF     STATUS, RP0     ; Select Register page 0
                      BCF     LCD_CNTL, RS    ; Set LCD for command mode
                      BSF     LCD_CNTL, R_W   ; Setup to read busy flag
                      BSF     LCD_CNTL, E     ; Set E high
                      BCF     LCD_CNTL, E     ; Set E low
                      MOVF    LCD_DATA, w     ; Read busy flag, DDram address
                      MOVWF   TEMP
                      BTFSC   TEMP, 7         ; Check busy flag, high=busy
                      GOTO    BUSY_CHECK
                      BCF     LCD_CNTL, R_W
                      BSF     STATUS, RP0     ; Select Register page 1
                      MOVLW   0x00
                      MOVWF   LCD_DATA_TRIS   ; Set port_D for output
                      BCF     STATUS, RP0     ; Select Register page 0
                      RETURN
           ;
                  endif
           ;
           Table
009B 0782             addwf   PCL       ;Jump to char pointed to in W reg
009C 344D             retlw   'M'
009D 3469             retlw   'i'
009E 3463             retlw   'c'
009F 3472             retlw   'r'
00A0 346F             retlw   'o'
00A1 3463             retlw   'c'
00A2 3468             retlw   'h'
00A3 3469             retlw   'i'
00A4 3470             retlw   'p'
00A5 3420             retlw   ' '
00A6 3454             retlw   'T'
00A7 3465             retlw   'e'
00A8 3463             retlw   'c'
00A9 3468             retlw   'h'
00AA 346E             retlw   'n'
00AB 346F             retlw   'o'
00AC 346C             retlw   'l'
00AD 346F             retlw   'o'
00AE 3467             retlw   'g'
00AF 3479             retlw   'y'
           Table_End
00B0 3400             retlw   0
           ;
                   if ( (Table & 0x0FF) >= (Table_End & 0x0FF) )
                  MESSG   "Warning - User Definded: Table Table crosses page boundry in computed jump"
```

```
                    0520     endif
                    0521 ;
                    0522
                    0523
                    0524
                    0525     end
                    0526
                    0527
                    0528
                    0529
                    0530
```

```
MPASM 00.00.68 Intermediate    LM032L.ASM   6-8-1994  5:29:26                PAGE 14


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

0000 : X—XXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0040 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX

0080 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX X———————
00C0 : ———————— ———————— ———————— ————————

All other memory blocks unused.


Errors   :    0
Warnings :   13
```

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 E: LM032L.H

```
                    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 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 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
```

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