



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		Result
Four_bit	Data_HI	
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.

Interfacing to an LCD Module

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

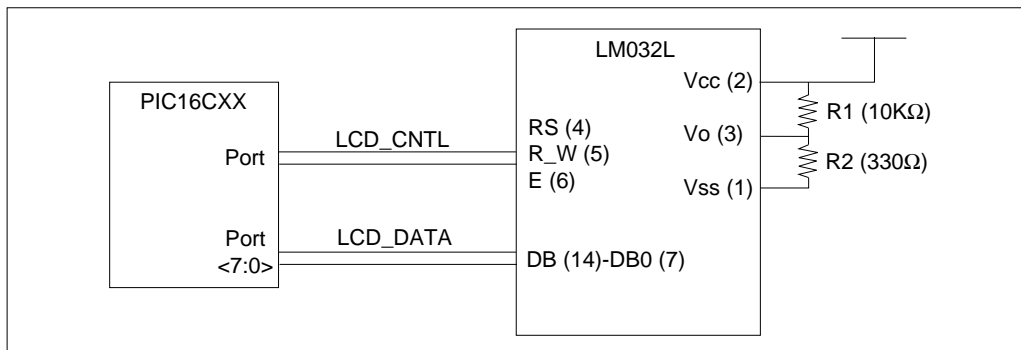


FIGURE 1B: 4-BIT MODE. DATA TRANSFERRED ON THE HIGH NIBBLE OF THE PORT

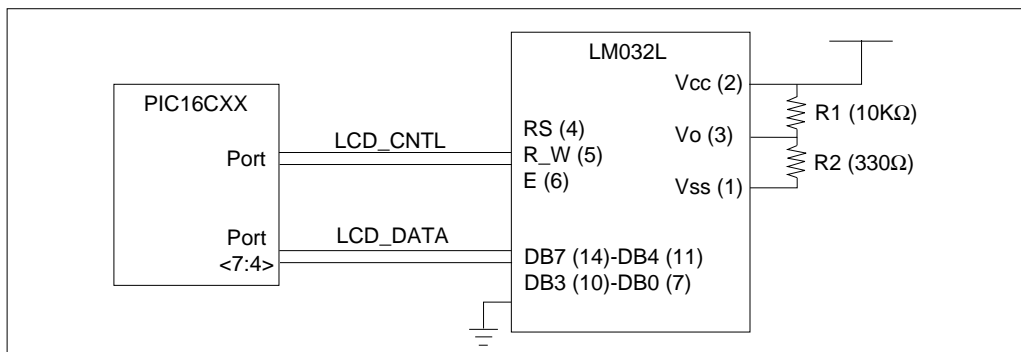
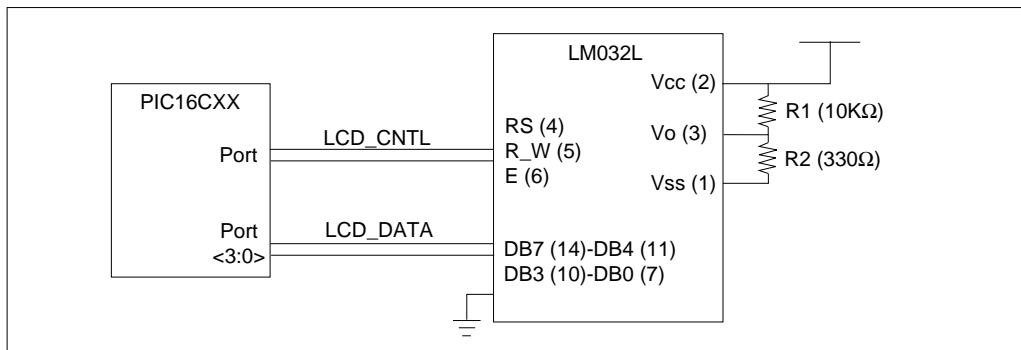


FIGURE 1C: 4-BIT MODE. DATA TRANSFERRED ON THE LOW NIBBLE OF THE PORT



Interfacing to an LCD Module

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.

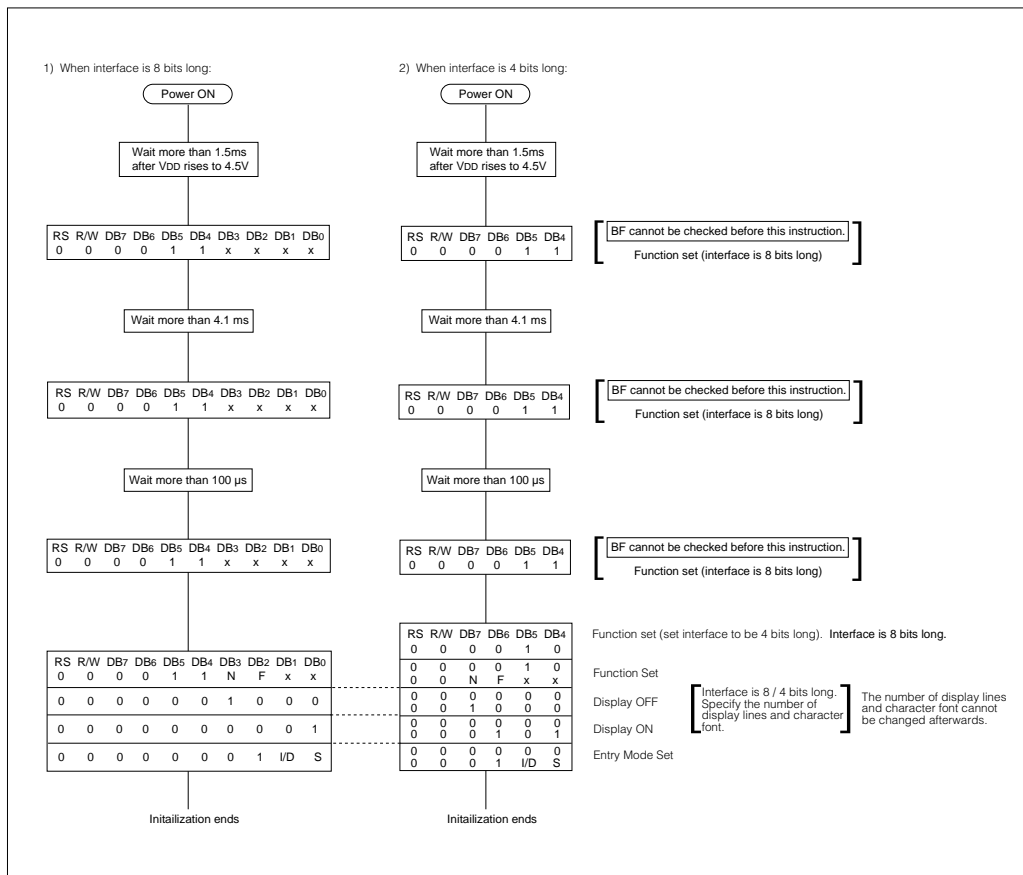
When the module powers up, the default data transfer mode is 8-bit. The initialization sequence only requires commands that are 4-bit in length. The last initialization command then needs to be sent to the display to specify the data transfer width (4- or 8-bit). Then a delay of

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.

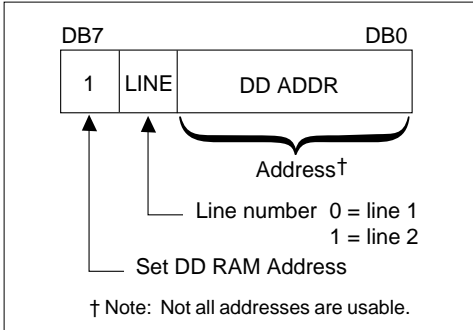
FIGURE 2: INITIALIZATION FLOW FOR LCD MODULE



Interfacing to an 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 Hitachi 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.

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

digit	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	33	34	35	36	37	38	39	40	← Display position
1-line	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	20	21	22	23	24	25	26	27	← DD RAM address (Hexadecimal)
2-line	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	50	51	52	53	54	60	61	62	63	64	65	66	67	

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

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 may be 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.

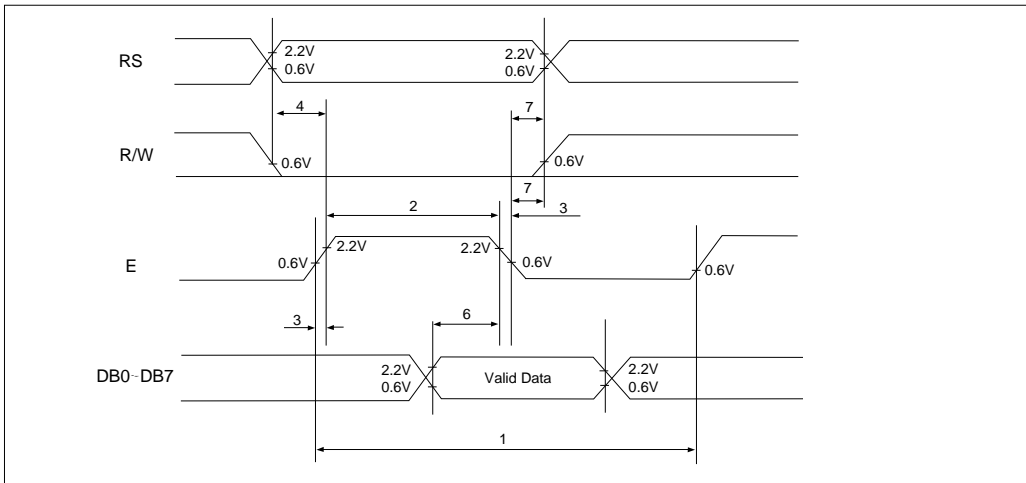
Written By: Mark Palmer - Sr. Application Engineer
Code By: Mark Palmer / Scott Fink - Sr. Application Engineers

APPENDIX A: LM032L TIMING REQUIREMENTS

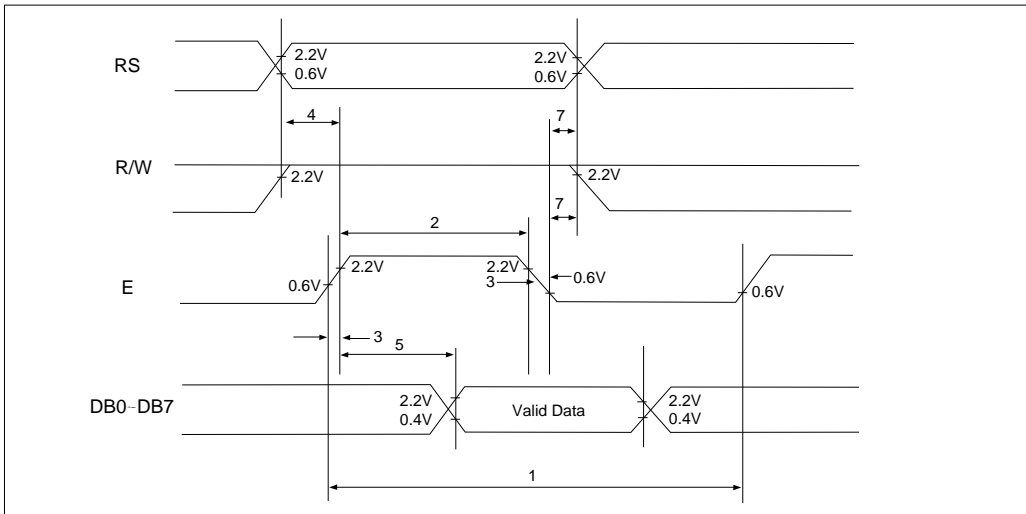
TIMING CHARACTERISTICS

Parm #	Symbol	Characterisitcs	Min.	Typ.	Max.	Unit
1	t _{cyc}	Enable cycle time	1.0	—	—	μs
2	PWEH	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.

Interfacing to an LCD Module

LM032L PIN CONNECTION

Pin No.	Symbol	Level	Function	
1	V _{SS}	—	0V	Power Supply
2	V _{DD}	—	+5V	
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 DB₄~DB₇ and DB₀~DB₃ are not used. Data transfer between the HD44780 and the MPU completes when 4-bit data is transferred twice. Data of the higher order 4 bits (contents of DB₄~DB₇ when interface data is 8-bits long) is transferred first and then lower order 4 bits (contents of DB₀~DB₃ when interface data is 8-bits long).
- (2) When interface data is 8-bits long, data is transferred using 8 data buses of DB₀~DB₇.

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_CNTRL 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
0014 include <lm032l.h>
0015 ;
0016 Four_bit EQU FALSE ; Selects 4- or 8-bit data transfers
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 ;
0027 LCD_DATA EQU PORTD
0028 LCD_DATA_TRIS EQU TRISD
0029 ;
0030 endif
0031 ;
0008
0088

```

Interfacing to an LCD Module

```
0005
0032 LCD_CNFL EQU PORTA
0033 ;
0034 ;
0035 ;
0036 ; LCD Display Commands and Control signal names.
0037 ;
0038 if ( Four_bit && !Data_HI )
0039 ;
0040 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 ;
0046 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 ;
0053 TEMP1 EQU 0x030
0054 ;
0055 org RESET_V ; RESET vector location
0056 RESET GOTO START ;
0057 ;
0058 ; This is the Peripheral Interrupt routine. Should NOT get here
0059 ;
0060 org ISR_V ; Interrupt vector location
0061 PER_INT_V
0062 ERROR1 BCF STATUS, RP0 ; Bank 0
0063 BSF PORTC, 0
0064 BCF PORTC, 0
0065 GOTO ERROR1
0066 ;
0067 ;
0068 ;
0069 ;
0070 START CLRF STATUS ; POWER_ON Reset (Beginning of program)
0071 INTCON CLRF INTCON ; Do initialization (Bank 0)
0072 CLR FIRL
0073 CLR STATUS, RP0 ; Bank 1
0074 MOVW 0x00 ; The LCD module does not like to work w/ weak pull-ups
0075 OPTION_R ;
0076
```



```

000E 018C          CLRFB    PIE1          ; Disable all peripheral interrupts
000F 30FF          MOVWLB   0xFF          ;
0010 009F          MOVWFB   ADCON1        ; Port A is Digital.
0080 ;
0081 ;
0082          BCF     STATUS, RP0          ; Bank 0
0083          CLRFB   PORTA          ; ALL PORT output should output Low.
0084          CLRFB   PORTB
0085          CLRFB   PORTC
0086          CLRFB   PORTD
0087          CLRFB   PORTE
0088          BCF     TLCON, TMR1ON      ; Timer 1 is NOT incrementing
0089 ;
0090          BSF     STATUS, RP0          ; Select Bank 1
0091          CLRFB   TRISA          ; RA5 - 0 outputs
0092          MOVWLB   0xF0
0093          MOVWFB   TRISB          ; RB7 - 4 inputs, RB3 - 0 outputs
0094          CLRFB   TRISC          ; RC Port are outputs
0095          BSF     TRISC, T1OSO       ; RC0 needs to be input for the oscillator to function
0096          CLRFB   TRISD          ; RD Port are outputs
0097          CLRFB   TRISE          ; RE Port are outputs
0098          BSF     PIE1, TMR1IE      ; Enable TMR1 Interrupt
0099          BSF     OPTION_R, RBFU    ; Disable PORTB pull-ups
0100          BCF     STATUS, RP0          ; Select Bank 0
0101 ;
0103 ;
0104 ; Initialize the LCD Display Module
0105 ;
0106          CLRFB   LCD_CNTL          ; ALL PORT output should output Low.
0107
0108 DISPLAY_INIT
0109   if ( Four_bit && !Data_HI )
0110     MOVWLB   0x02
0111   endif
0112 ;
0113   if ( !Four_bit && Data_HI )
0114     MOVWLB   0x020
0115   endif
0116 ;
0117   if ( !Four_bit )
0118     MOVWLB   0x038
0119   endif
0120 ;
0121   MOVWFB   LCD_DATA
0122   BSF     LCD_CNTL, E

```

Interfacing to an LCD Module

```
0027 1185          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 ;
0130 LCD_DELAY      MOV LW  LCD_INIT_DELAY ;
0131                MOVWF  MSD             ; Use MSD and LSD Registers to Initialize LCD
0132                CLR F                    ;
0133 LOOP2          DECFSZ LSD             ;
0134                GOTO  LOOP2            ; Delay time = MSD * ((3 * 256) + 3) * Tcy
0135                DECFSZ MSD             ;
0136 END_LCD_DELAY
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                        MOV LW  0X02      ; 4-bit low nibble xfer
0146                    else
0147                        MOV LW  0X020     ; 4-bit high nibble xfer
0148                    endif
0149 ;
0150                else
0151                    MOV LW  0X038         ; 8-bit mode
0152                endif
0153 ;
0154                MOVWF  LCD_DATA          ; This code for both 4-bit and 8-bit modes
0155                BSF   LCD_CNTL, E        ;
0156                BCF   LCD_CNTL, E        ;
0157 ;
0158                if ( Four_bit )
0159                    if ( !Data_HI )
0160                        MOV LW  0x08      ; 4-bit low nibble xfer
0161                    else
0162                        MOV LW  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 ;
0171         MOVLW    DISP_ON      ;
0172         CALL    SEND_CMD     ;
0173         MOVLW    CLR_DISP    ;
0174         CALL    SEND_CMD     ;
0175         MOVLW    ENTRY_INC   ;
0176         CALL    SEND_CMD     ;
0177         MOVLW    DD_RAM_ADDR ;
0178         CALL    SEND_CMD     ;
0179 ;
0181 ;
0182 ;send a message the hard way
0183         movlw   'W'
0184         call    SEND_CHAR
0185         movlw   'i'
0186         call    SEND_CHAR
0187         movlw   'c'
0188         call    SEND_CHAR
0189         movlw   'r'
0190         call    SEND_CHAR
0191         movlw   'o'
0192         call    SEND_CHAR
0193         movlw   'c'
0194         call    SEND_CHAR
0195         movlw   'h'
0196         call    SEND_CHAR
0197         movlw   'i'
0198         call    SEND_CHAR
0199         movlw   'p'
0200         call    SEND_CHAR
0201
0202         movlw   B'11000000' ;Address DDRam first character, second line
0203         call    SEND_CMD
0204
0205 ;Demonstration of the use of a table to output a message
0206         movlw   0 ;Table address of start of message
0207         dispmsg
0208         movwf   TEMP1 ;TEMP1 holds start of message address
0209         call    Table
0210         andlw   0FFh ;Check if at end of message (zero
0211         btfsc  STATUS,Z ;returned at end)
0212         goto    out
0213         call    SEND_CHAR ;Display character
0214         movf   TEMP1,w ;Point to next character

```



```

0262         RETURN
0263 ;
0264         else
0265 ;
0266 ;***** ; 4-bit transfers on the low nibble of the PORT
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
0274     MOVWF CHAR           ; Character to be sent is in W
0275     CALL  BUSY_CHECK    ; Wait for LCD to be ready
0276     SWAPF CHAR, W
0277     ANDLW 0x0F         ; Get upper nibble
0278     MOVWF LCD_DATA    ; Send data to LCD
0279     BCF   LCD_CNTRL, R_W ; Set LCD to read
0280     BSF   LCD_CNTRL, RS ; Set LCD in read mode
0281     BSF   LCD_CNTRL, E ; Set LCD to data mode
0282     BCF   LCD_CNTRL, E ; toggle E for LCD
0283     MOVF  CHAR, W
0284     ANDLW 0x0F         ; Get lower nibble
0285     MOVWF LCD_DATA    ; Send data to LCD
0286     BSF   LCD_CNTRL, E ; Set LCD in read mode
0287     BCF   LCD_CNTRL, E ; toggle E for LCD
0288     RETURN
0289 ;
0290     endif
0291     else
0292 ;*****
0293 ; SEND_CHAR - Sends character contained in register W to LCD
0294 ; This routine sends the entire character to the PORT
0295 ; The data is transmitted on the PORT<7:0> pins
0296 ;*****
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_CNTRL, R_W ; Set LCD in read mode
0305     BSF   LCD_CNTRL, RS ; Set LCD in data mode
0306     BSF   LCD_CNTRL, E ; toggle E for LCD
0307     BCF   LCD_CNTRL, E
0308     RETURN
0061 00B6
0062 2073
0063 0836
0064 0088
0065 1105
0066 1485
0067 1585
0068 1185
0069 0008

```

Interfacing to an LCD Module

```
0309 ;
0310     endif
0311 ;
0312 ;
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_CNTRL,R_W ; Set LCD to read
0338     BCF LCD_CNTRL,RS ; Set LCD to command mode
0339     BSF LCD_CNTRL,E ; toggle E for LCD
0340     BCF LCD_CNTRL,E
0341     SWAPF CHAR,w
0342     ANDLW 0xF0 ; Get lower nibble
0343     MOVWF LCD_DATA ; Send data to LCD
0344     BSF LCD_CNTRL,E ; toggle E for LCD
0345     BCF LCD_CNTRL,E
0346     RETURN
0347 ;
0348 ;
0349 ;     else ; 4-bit transfers on the low nibble of the PORT
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_CNTRL,R_W ; Set LCD to read
```

```

0357          LCD_CNTRL, RS      ; Set LCD to command mode
0358          BCF LCD_CNTRL, E    ; toggle E for LCD
0359          BCF LCD_CNTRL, E
0360          MOVF CHAR, W
0361          ANDLW 0x0F          ; Get lower nibble
0362          MOVWF LCD_DATA      ; Send data to LCD
0363          BSF LCD_CNTRL, E    ; toggle E for LCD
0364          BCF LCD_CNTRL, E
0365          RETURN
0366 ;
0367         endif
0368         else
0369 ;
0370 ;*****
0371 ; * SEND_CMD - Sends command contained in register W to LCD *
0372 ; * This routine sends the entire character to the PORT *
0373 ; * 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_CNTRL, R_W  ; Set LCD in read mode
0382         BCF LCD_CNTRL, RS   ; Set LCD in command mode
0383         BSF LCD_CNTRL, E    ; toggle E for LCD
0384         BCF LCD_CNTRL, E
0385         RETURN
0386 ;
0387         endif
0388 ;
0389 ;
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

```

Interfacing to an LCD Module

```
0405 STATUS, RP0 ; Select Register page 0
0406 LCD_CNTL, RS ; Set LCD for Command mode
0407 LCD_CNTL, R_W ; Setup to read busy flag
0408 LCD_CNTL, E ; Set E high
0409 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 ;
0423 MOVWF LCD_DATA, TRIS ; Set Port.D for output
0424 BCF STATUS, RP0 ; Select Register page 0
0425 RETURN ;
0426 ;
0427 ; else ; 4-bit transfers on the low nibble of the PORT
0428 ;
0429 ; *****
0430 ; * This routine checks the busy flag, returns when not busy *
0431 ; * Affects: *
0432 ; * TEMP - Returned with busy/address *
0433 ; *****
0434 ;
0435 BUSY_CHECK ; Bank 1
0436 STATUS, RP0 ; Set PortB for input
0437 MOVLW 0xFF ;
0438 MOVWF LCD_DATA, TRIS ; Bank 0
0439 BCF STATUS, RP0 ; Set LCD for Command mode
0440 BCF LCD_CNTL, RS ; Setup to read busy flag
0441 BSF LCD_CNTL, R_W ; Set E high
0442 BSF LCD_CNTL, E ; Set E low
0443 BCF LCD_CNTL, E ;
0444 SWAPF LCD_DATA, W ; Read upper nibble busy flag, DDRam address
0445 ANDLW 0xF0 ; Mask out lower nibble
0446 MOVWF TEMP ;
0447 BSF LCD_CNTL, E ; Toggle E to get lower nibble
0448 BCF LCD_CNTL, E ;
0449 MOVF LCD_DATA, W ; Read lower nibble busy flag, DDRam address
0450 ANDLW 0x0F ; Mask out upper nibble
0451 IORWF TEMP, F ; Combine nibbles
```



```

0452          BTFSC          TEMP, 7
0453          GOTO          BUSY_CHECK
0454          BCF           LCD_CNTRL, R_W
0455          BSF           STATUS, RP0
0456          MOVLW        0xF0
0457          MOVWF        LCD_DATA_TRIS
0458          BCF           STATUS, RP0
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
0471          BSF           STATUS,RP0
0472          MOVLW        0xFF
0473          MOVWF        LCD_DATA_TRIS
0474          BCF           STATUS, RP0
0475          BCF           LCD_CNTRL, RS
0476          BSF           LCD_CNTRL, R_W
0477          BSF           LCD_CNTRL, E
0478          BCF           LCD_CNTRL, E
0479          MOVF         LCD_DATA, W
0480          MOVWF        TEMP
0481          BTFSC        TEMP, 7
0482          GOTO          BUSY_CHECK
0483          BCF           LCD_CNTRL, R_W
0484          BSF           STATUS, RP0
0485          MOVLW        0x00
0486          MOVWF        LCD_DATA_TRIS
0487          BCF           STATUS, RP0
0488          RETURN
0489 ;
0490          endif
0491 ;
0492 ;
0493 Table
0494          adwlf        PCL
0495          retlw        'M'
0496          retlw        'i'
0497          retlw        'c'
0498          retlw        'r'
0499          retlw        'o'
0500          retlw        'c'

0073 1683
0074 30FF
0075 0088
0076 1283
0077 1085
0078 1505
0079 1585
007A 1185
007B 0808
007C 00B5
007D 1BB5
007E 2873
007F 1105
0080 1683
0081 3000
0082 0088
0083 1283
0084 0008

0085 0782
0086 344D
0087 3469
0088 3463
0089 3472
008A 346F
008B 3463

```

Interfacing to an LCD Module

```
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'
009A 3400      retlw 0
0517 ;
0518          if ( (Table & 0x0FF) >= (Table_End & 0x0FF) )
0519              MESSG "Warning - User Defined: Table Table crosses page boundary in computed_jump"
0520          endif
0521 ;
0522
0523
0524          end
0525
0526
0527
0528
0529
0530
```

PAGE 14

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

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

0000 : X-XXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX

0040 : XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX

0080 : XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXX-_____

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 labels 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=INH8M
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_CNTRL 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>
0014
0015 ;
0016 Four_bit EQU TRUE ; Selects 4- or 8-bit data transfers
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 ;
0027 LCD_DATA EQU PORTD
0028 LCD_DATA_TRIS EQU TRISD
0029 ;
0030 endif
0031 ;
0032 LCD_CNTRL EQU PORTA
0033 ;
0034 ;

```

```

0035 ;
0036 ; LCD Display Commands and Control Signal names.
0037 ;
0038     if ( Four_bit && !Data_HI )
0039 ;
0040 EQU     0
0041 EQU     1
0042 EQU     2
                                ; LCD Enable control line
                                ; LCD Read/Write control line
                                ; LCD Register Select control
line

0043 ;
0044     else
0045 ;
0046 EQU     3
0047 EQU     2
0048 EQU     1
                                ; LCD Enable control line
                                ; LCD Read/Write control line
                                ; LCD Register Select control
line

0030
0050     endif
0051 ;
0052 ;
0053 EQU     0x030
0054 ;
0055     org     RESET_V           ; RESET vector location
0056 RESET   GOTO   START       ;
0057 ;
0058 ; This is the Peripheral Interrupt routine. Should NOT get here
0059 ;
0060     org     ISR_V           ; Interrupt vector location
0061 PER_INT_V
0062 ERROR1   BCF   STATUS, RP0   ; Bank 0
0063         BSF   PORTC, 0
0064         BCF   PORTC, 0
0065         GOTO   ERROR1
0066 ;
0067 ;
0068 ;
0069 ;
0070 START   CLR   STATUS
0071         CLR   INTCON
0072         CLR   PIR1
0073         BSF   STATUS, RP0     ; Bank 1
0074         MOVW  0x00          ; The LCD module does not like to work w/ weak pull-ups
0075         MOVWF OPTION_R    ;
0076         CLR   PIR1         ; Disable all peripheral interrupts
0077         MOVW  0xFF        ;
0078         MOVWF ADCON1     ; Port A is Digital.
0079 ;
0080 ;
0081 ;
0004 1283
0005 1407
0006 1007
0007 2804
0008 0183
0009 018B
000A 018C
000B 1683
000C 3000
000D 0081
000E 018C
000F 30FF
0010 009F

```

Interfacing to an LCD Module

```

0011 1283          STATUS, RP0          ; Bank 0
0012 1285          CLRF PORTA          ; ALL PORT output should output Low.
0013 1286          CLRF PORTB
0014 1287          CLRF PORTC
0015 1288          CLRF PORTD
0016 1289          CLRF PORTE
0017 1010          BCF TLCON, TMR1ON   ; Timer 1 is NOT incrementing
0018 1683          ;
0019 1685          BSF STATUS, RP0     ; Select Bank 1
0020 30F0          CLRF TRISA         ; RA5 - 0 outputs
0021 30F2          MOVWF 0xF0
0022 0086          TRISB              ; RB7 - 4 inputs, RB3 - 0 outputs
0023 0187          CLRF TRISC        ; RC port are outputs
0024 1407          BSF TRISC, T1OSO   ; RC0 needs to be input for the oscillator to function
0025 0188          CLRF TRISD        ; RD port are outputs
0026 0189          CLRF TRISE        ; RE port are outputs
0027 140C          BSF PIE1, TMR1IE   ; Enable TMR1 Interrupt
0028 1781          BSF OPTION_R, RBFU ; Disable PORTB pull-ups
0029 1283          BCF STATUS, RP0    ; Select Bank 0
0030 1011          ;
0031 1013          ;
0032 1014          ; Initialize the LCD Display Module
0033 1015          ;
0034 1016          CLRF LCD_CNTRL    ; ALL PORT output should output Low.
0035 1017          ;
0036 1018          DISPLAY_INIT
0037 1019          if ( Four_bit && !Data_HI )
0038 1110          MOVWLW 0x02        ; Command for 4-bit interface low nibble
0039 1111          endif
0040 1112          ;
0041 1113          if ( !Four_bit && Data_HI )
0042 1114          MOVWLW 0x020       ; Command for 4-bit interface high nibble
0043 1115          endif
0044 1116          ;
0045 1117          if ( !Four_bit )
0046 1118          MOVWLW 0x038       ; Command for 8-bit interface
0047 1119          endif
0048 1020          ;
0049 0088          MOVWF LCD_DATA     ;
0050 1585          BSF LCD_CNTRL, E   ;
0051 1185          BCF LCD_CNTRL, E   ;
0052 1024          ;
0053 1025          ; This routine takes the calculated times that the delay loop needs to
0054 1026          ; be executed, based on the LCD_INIT_DELAY EQUate that includes the
0055 1027          ; frequency of operation. These uses registers before they are needed to
0056 1028          ; store the time.
0057 1029          ;
0058 3006          LCD_DELAY MOVWLW LCD_INIT_DELAY ;

```

```

0029 00B3          MOVWF MSD          ; Use MSD and LSD Registers to Initialize LCD
002A 01B4          CLRF LSD           ;
002B 0BB4          DEFSZ LSD          ; Delay time = MSD * ((3 * 256) + 3) * Tcy
002C 282B          GOTO LOOP2        ;
002D 0BB3          DEFSZ MSD         ;
002E 282B          GOTO LOOP2        ;
002F 3020          END_LCD_DELAY     ;
0030 0088          MOVWF LCD_DATA    ; This code for both 4-bit and 8-bit modes
0031 1585          BSF LCD_CNTRL, E   ;
0032 1185          BCF LCD_CNTRL, E   ;
0033 3080          MOVWF 0x080       ; 4-bit low nibble xfer
0034 0088          MOVWF LCD_DATA    ;
0035 1585          BSF LCD_CNTRL, E   ;
0036 1185          BCF LCD_CNTRL, E   ;
0037 300C          MOVWF DISP_ON     ;
0038 2074          CALL SEND_CMD     ;
0039 3001          MOVWF CLR_DISP    ;
003A 2074          CALL SEND_CMD     ;
003B 3006          MOVWF ENTRY_INC  ;
003C 2074          CALL SEND_CMD     ;
003D 3080          MOVWF DD_RAM_ADDR ;
003E 2074          CALL SEND_CMD     ;
003F 300C          MOVWF DISP_ON     ;
0040 2074          CALL SEND_CMD     ;
0041 3001          MOVWF CLR_DISP    ;
0042 2074          CALL SEND_CMD     ;
0043 3006          MOVWF ENTRY_INC  ;
0044 2074          CALL SEND_CMD     ;
0045 3001          MOVWF DD_RAM_ADDR ;
0046 2074          CALL SEND_CMD     ;
0047 300C          MOVWF DISP_ON     ;
0048 2074          CALL SEND_CMD     ;
0049 3001          MOVWF CLR_DISP    ;
004A 2074          CALL SEND_CMD     ;
004B 3006          MOVWF ENTRY_INC  ;
004C 2074          CALL SEND_CMD     ;
004D 3001          MOVWF DD_RAM_ADDR ;
004E 2074          CALL SEND_CMD     ;
004F 300C          MOVWF DISP_ON     ;
0050 2074          CALL SEND_CMD     ;
0051 3001          MOVWF CLR_DISP    ;
0052 2074          CALL SEND_CMD     ;
0053 3006          MOVWF ENTRY_INC  ;
0054 2074          CALL SEND_CMD     ;
0055 3001          MOVWF DD_RAM_ADDR ;
0056 2074          CALL SEND_CMD     ;
0057 300C          MOVWF DISP_ON     ;
0058 2074          CALL SEND_CMD     ;
0059 3001          MOVWF CLR_DISP    ;
005A 2074          CALL SEND_CMD     ;
005B 3006          MOVWF ENTRY_INC  ;
005C 2074          CALL SEND_CMD     ;
005D 3001          MOVWF DD_RAM_ADDR ;
005E 2074          CALL SEND_CMD     ;
005F 300C          MOVWF DISP_ON     ;
0060 2074          CALL SEND_CMD     ;
0061 3001          MOVWF CLR_DISP    ;
0062 2074          CALL SEND_CMD     ;
0063 3006          MOVWF ENTRY_INC  ;
0064 2074          CALL SEND_CMD     ;
0065 3001          MOVWF DD_RAM_ADDR ;
0066 2074          CALL SEND_CMD     ;
0067 300C          MOVWF DISP_ON     ;
0068 2074          CALL SEND_CMD     ;
0069 3001          MOVWF CLR_DISP    ;
006A 2074          CALL SEND_CMD     ;
006B 3006          MOVWF ENTRY_INC  ;
006C 2074          CALL SEND_CMD     ;
006D 3001          MOVWF DD_RAM_ADDR ;
006E 2074          CALL SEND_CMD     ;
006F 300C          MOVWF DISP_ON     ;
0070 2074          CALL SEND_CMD     ;
0071 3001          MOVWF CLR_DISP    ;
0072 2074          CALL SEND_CMD     ;
0073 3006          MOVWF ENTRY_INC  ;
0074 2074          CALL SEND_CMD     ;
0075 3001          MOVWF DD_RAM_ADDR ;
0076 2074          CALL SEND_CMD     ;
0077 300C          MOVWF DISP_ON     ;
0078 2074          CALL SEND_CMD     ;

```

Interfacing to an LCD Module

```
003F 304D      0179 ;
0040 2065      0181 ;
0041 3069      0182 ;Send a message the hard way
0042 2065      0183      movlw 'M'
0043 3063      0184      call SEND_CHAR
0044 2065      0185      movlw 'i'
0045 3072      0186      call SEND_CHAR
0046 2065      0187      movlw 'c'
0047 306F      0188      call SEND_CHAR
0048 2065      0189      movlw 'r'
0049 3063      0190      call SEND_CHAR
0050 3070      0191      movlw 'o'
0051 30C0      0192      call SEND_CHAR
0052 2074      0193      call SEND_CHAR
0053 3000      0194      call SEND_CHAR
0054 00B0      0195      movlw 'h'
0055 209B      0196      call SEND_CHAR
0056 39FF      0197      movlw 'i'
0057 1903      0198      call SEND_CHAR
0058 285D      0199      movlw 'p'
0059 2065      0200      call SEND_CHAR
0060 3001      0201
0061 2074      0202      movlw B'11000000'
0062 3006      0203      call SEND_CMD
0063 3006      0204
0064 3006      0205 ;Demonstration of the use of a table to output a message
0065 3006      0206      ;Table address of start of message
0066 3006      0207      dispmsg
0067 3006      0208      movwf TEMP1
0068 3006      0209      call Table
0069 3006      0210      andlw 0FFh
0070 3006      0211      btfsc STATUS,Z
0071 3006      0212      goto out
0072 3006      0213      call SEND_CHAR
0073 3006      0214      movf TEMP1,w
0074 3006      0215      addlw 1
0075 3006      0216      goto dispmsg
0076 3006      0217      out
0077 3006      0218      loop
0078 3006      0219      goto loop
0079 3006      0220 ;
0080 3006      0221 ;
0081 3006      0222 INIT_DISPLAY
0082 3006      0223      MOVLW DISP_ON
0083 3006      0224      CALL SEND_CMD
0084 3006      0225      MOVLW CLR_DISP
0085 3006      0226      CALL SEND_CMD
0086 3006      0227      MOVLW ENTRY_INC
0087 3006      ; Display On, Cursor On
0088 3006      ; Send This command to the Display Module
0089 3006      ; Clear the Display
0090 3006      ; Send This command to the Display Module
0091 3006      ; Set Entry Mode Inc., No shift
0092 3006      ; Stay here forever
```



```

0063 2074      CALL SEND_CMD      ; Send This command to the Display Module
0064 0008      RETURN

0228          CALL SEND_CMD      ; Send This command to the Display Module
0229          RETURN
0230 ;
0231 ;
0232 ;
0233 ;*****
0234 ; * The LCD Module Subroutines
0235 ;*****
0236 ;
0237          if ( Four_bit )      ; 4-bit Data transfers?
0238          if ( Data_HI )      ; 4-bit transfers on the high nibble of the PORT
0239          ;
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
0252          MOVWF LCD_DATA      ;Get upper nibble
0253          BCF LCD_CNTRL, R_W  ;Send data to LCD
0254          BSF LCD_CNTRL, RS   ;Set LCD to read
0255          BSF LCD_CNTRL, E    ;Set LCD to data mode
0256          BCF LCD_CNTRL, E    ;toggle E for LCD
0257          SWAPF CHAR, W
0258          ANDLW 0xF0
0259          MOVWF LCD_DATA      ;Get lower nibble
0260          BSF LCD_CNTRL, E    ;Send data to LCD
0261          BCF LCD_CNTRL, E    ;toggle E for LCD
0262          RETURN
0263 ;
0264          else
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
0274          MOVWF CHAR            ; Character to be sent is in W
0275          CALL BUSY_CHECK      ; Wait for LCD to be ready
0276          SWAPF CHAR, W
0277          ;

```

Interfacing to an LCD Module

```
0277 ANDLW 0x0F           ; Get upper nibble
0278 MOVWF LCD_DATA      ; Send data to LCD
0279 BCF LCD_CNTRL, R_W  ; Set LCD to read
0280 BSF LCD_CNTRL, RS   ; Set LCD to data mode
0281 BSF LCD_CNTRL, E   ; toggle E for LCD
0282 BCF LCD_CNTRL, E
0283 MOVF CHAR, W       ; Get lower nibble
0284 ANDLW 0x0F        ; Send data to LCD
0285 MOVWF LCD_DATA    ; Send data to LCD
0286 BSF LCD_CNTRL, E  ; toggle E for LCD
0287 BCF LCD_CNTRL, E
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_CNTRL, R_W ; Set LCD in read mode
0305 BSF LCD_CNTRL, RS  ; Set LCD in data mode
0306 BSF LCD_CNTRL, E  ; toggle E for LCD
0307 BCF LCD_CNTRL, E
0308 RETURN
0309 ;
0310 endif
0311 ;
0312 ;
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 ;*****
```

```

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          ; Get upper nibble
0077 39F0          ANDLW   0x0F          ; Send data to LCD
0078 0088          MOVWF   LCD_DATA        ; Send data to LCD
0079 1105          BCF     LCD_CNTRL,R_W      ; Set LCD to read
007A 1085          BCF     LCD_CNTRL,RS      ; Set LCD to command mode
007B 1585          BSF     LCD_CNTRL,E      ; toggle E for LCD
007C 1185          BCF     LCD_CNTRL,E
007D 0E36          SWAPF   CHAR,W          ; Get lower nibble
007E 39F0          ANDLW   0x0F          ; Send data to LCD
007F 0088          MOVWF   LCD_DATA        ; Send data to LCD
0080 1585          BSF     LCD_CNTRL,E      ; toggle E for LCD
0081 1185          BCF     LCD_CNTRL,E
0082 0008          RETURN

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          ; Get upper nibble
0335          ANDLW   0x0F          ; Send data to LCD
0336          MOVWF   LCD_DATA        ; Send data to LCD
0337          BCF     LCD_CNTRL,R_W      ; Set LCD to read
0338          BCF     LCD_CNTRL,RS      ; Set LCD to command mode
0339          BSF     LCD_CNTRL,E      ; toggle E for LCD
0340          BCF     LCD_CNTRL,E
0341          SWAPF   CHAR,W          ; Get lower nibble
0342          ANDLW   0x0F          ; Send data to LCD
0343          MOVWF   LCD_DATA        ; Send data to LCD
0344          BSF     LCD_CNTRL,E      ; toggle E for LCD
0345          BCF     LCD_CNTRL,E
0346          RETURN

0347 ;
0348          else
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          ; Get upper nibble
0354          ANDLW   0x0F          ; Send data to LCD
0355          MOVWF   LCD_DATA        ; Send data to LCD
0356          BCF     LCD_CNTRL,R_W      ; Set LCD to read
0357          BCF     LCD_CNTRL,RS      ; Set LCD to command mode
0358          BSF     LCD_CNTRL,E      ; toggle E for LCD
0359          BCF     LCD_CNTRL,E
0360          MOVF     CHAR,W          ; Get lower nibble
0361          ANDLW   0x0F          ; Send data to LCD
0362          MOVWF   LCD_DATA        ; Send data to LCD
0363          BSF     LCD_CNTRL,E      ; toggle E for LCD
0364          BCF     LCD_CNTRL,E
0365          RETURN

0366 ;
0367          endif
0368          else
0369 ;
0370 ; *****
0371 ; * SEND_CMD - Sends command contained in register W to LCD
0372 ; * This routine sends the entire character to the PORT
0373 ; * The data is transmitted on the PORT<7:0> pins

```

Interfacing to an LCD Module

```

0374 ; *****
0375
0376 SEND_CMD          CHAR          ; Command to be sent is in W
0377                  BUSY_CHECK      ; Wait for LCD to be ready
0378                  CALL            ;
0379                  MOVWF           ; Send data to LCD
0380                  CHAR, W         ;
0381                  LCD_DATA       ; Set LCD in read mode
0382                  LCD_CNTL, R_W  ; Set LCD in command mode
0383                  BCF            ; toggle E for LCD
0384                  LCD_CNTL, E    ;
0385                  BCF            ;
0386                  RETURN         ;
0387
0388                  ;
0389                  ;
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                  MOVWF         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
0083 1683
0084 30FF
0085 0088
0086 1283
0087 1085
0088 1505
0089 1585
008A 1185
008B 0808
008C 39F0
008D 00B5
008E 1585
008F 1185
0090 0E08
0091 390F
0092 04B5
0093 1BB5
0094 2883
0095 1105
0096 1683
0097 300F

```

```

0098 0088      MOVWF LCD_DATA_TRIS ; Set Port_D for output
0099 1283      BCF STATUS, RP0 ; Select Register page 0
009A 0008      RETURN

0426 ;
0427 ;
0428 ;
0429 ; ***** ; 4-bit transfers on the low nibble of the PORT
0430 ; * This routine checks the busy flag, returns when not busy *
0431 ; * Affects: *
0432 ; * TEMP - Returned with busy/address *****
0433 ; *****
0434 ;
0435 BUSY_CHECK
0436      BSF STATUS, RP0 ; Bank 1
0437      MOVLW 0xFF ; Set PortB for input
0438      MOVWF LCD_DATA_TRIS
0439      BCF STATUS, RP0 ; Bank 0
0440      BCF LCD_CNTL, RS ; Set LCD for Command mode
0441      BSF LCD_CNTL, R_W ; Setup to read busy flag
0442      BCF LCD_CNTL, E ; Set E high
0443      BCF LCD_CNTL, E ; Set E low
0444      SWAPF LCD_DATA, W ; Read upper nibble busy flag, DDRam address
0445      ANDLW 0xF0 ; Mask out lower nibble
0446      MOVWF TEMP ;
0447      BSF LCD_CNTL, E ; Toggle E to get lower nibble
0448      BCF LCD_CNTL, E ;
0449      MOVF LCD_DATA, W ; Read lower nibble busy flag, DDRam address
0450      ANDLW 0x0F ; Mask out upper nibble
0451      IORWF TEMP, F ; Combine nibbles
0452      BTFSF 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 ; * This routine checks the busy flag, returns when not busy *
0465 ; * Affects: *
0466 ; * TEMP - Returned with busy/address *****
0467 ; *
0468 ; *****
0469 ;
0470 BUSY_CHECK

```

Interfacing to an LCD Module

```

0471: STATUS,RP0           ; Select Register page 1
0472: MOVW 0xFF           ; Set port_D for input
0473: LCD_DATA_TRIS
0474: STATUS, RP0           ; Select Register page 0
0475: BCF LCD_CNTRL, RS    ; Set LCD for command mode
0476: BSF LCD_CNTRL, R_W   ; Setup to read busy flag
0477: BSF LCD_CNTRL, E     ; Set E high
0478: BCF LCD_CNTRL, E   ; Set E low
0479: MOVF LCD_DATA, W    ; Read busy flag, DDrAm address
0480: MOVWF TEMP
0481: BTFSC TEMP, 7       ; Check busy flag, high=busy
0482: GOTO BUSY_CHECK
0483: BCF LCD_CNTRL, R_W   ; Select Register page 1
0484: BSF STATUS, RP0
0485: MOVW 0x00
0486: MOVWF LCD_DATA_TRIS ; Set port_D for output
0487: BCF STATUS, RP0   ; Select Register page 0
0488: RETURN
0489: ;
0490: ; endif
0492: ;
0493: Table
0494: addwf PCL           ;Jump to char pointed to in W reg
0495: retlw 'M'
0496: retlw 'i'
0497: retlw 'c'
0498: retlw 'r'
0499: retlw 'o'
0500: retlw 'c'
0501: retlw 'h'
0502: retlw 'i'
0503: retlw 'p'
0504: retlw '\ '
0505: retlw 't'
0506: retlw 'e'
0507: retlw 'c'
0508: retlw 'h'
0509: retlw 'n'
0510: retlw 'o'
0511: retlw 'l'
0512: retlw 'o'
0513: retlw 'g'
0514: retlw 'y'
0515: Table_End
0516: retlw 0
0517: ;
0518: if ( (Table & 0x0FF) >= (Table_End & 0x0FF) )
0519: MESSG "Warning - User Defined: Table Table crosses page boundary in computed jump"

```

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

PAGE 14

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

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

```
0000 : X-XXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX
0040 : XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX
0080 : XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX 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 labels 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_CNTRL 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>
0014
0015 ;
0016 Four_bit EQU TRUE ; Selects 4- or 8-bit data transfers
0017 Data_HI EQU FALSE ; 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 ;
0027 LCD_DATA EQU PORTD
0028 LCD_DATA_TRIS EQU TRISD
0029 ;
0030 endif
0031 ;
0032 LCD_CNTRL EQU PORTA
0033 ;
0034 ;
0035 ;
0005

```



```

0036 ; LCD Display Commands and Control Signal names.
0037 ;
0038         if ( Four_bit && !Data_HI )
0039 ;
0040 EQU Enable control line
0041 EQU LCD Read/Write control line
0042 EQU LCD Register Select control
line

0043 ;
0044         else
0045 ;
0046 EQU Enable control line
0047 EQU LCD Read/Write control line
0048 EQU LCD Register Select control
line

0030
0053 TEMP1 EQU 0x030
0054 ;
0055         org RESET_V           ; RESET vector location
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
0063 ERROR1 BCF STATUS, RP0      ; Bank 0
0064         BSF PORTC, 0
0065         BCF PORTC, 0
0066         GOTO ERROR1
0067 ;
0068 ;
0069 ;
0070 START CLRF STATUS
0071         CLRF INTCON
0072         CLRF PIR1
0073         BSF STATUS, RP0      ; Bank 1
0074         MOVWF OPTION_R      ; The LCD module does not like to work w/ weak pull-ups
0075         CLRF PIR1           ; Disable all peripheral interrupts
0076         MOVWF ADCON1
0077         MOVWF ADCON1
0078 ;
0079 ; Port A is Digital.
0080 ;
0081 ;
0082         BCF STATUS, RP0      ; Bank 0
0083
0004 1283
0005 1407
0006 1007
0007 2804

0008 0183
0009 018B
000A 018C
000B 1683
000C 3000
000D 0081
000E 018C
000F 30FF
0010 009F

0011 1283

```

Interfacing to an LCD Module

```

0012 0185 PORTA          ; ALL PORT output should output Low.
0013 0186 PORTB
0014 0187 PORTC
0015 0188 PORTD
0016 0189 PORTE
0017 1010 BCF          ; Timer 1 is NOT incrementing
0018 1683 BSF          ; Select Bank 1
0019 0185 TRISA      ; RA5 - 0 outputs
001A 30F0 MOVLW      ;
001B 0086 MOVWF      ; RB7 - 4 inputs, RB3 - 0 outputs
001C 0187 CLRF       ; RC Port are outputs
001D 1407 BSF        ; RC0 needs to be input for the oscillator to function
001E 0188 CLRF       ; RD Port are outputs
001F 0189 CLRF       ; RE Port are outputs
0020 140C BSF        ; Enable TMR1 Interrupt
0021 1781 BSF        ; Disable PORTB pull-ups
0022 1283 BCF        ; Select Bank 0
0023 0185          ; ALL PORT output should output Low.
0024 3002          ; Command for 4-bit interface low nibble
0025 0086          ; Command for 4-bit interface high nibble
0026 1405          ; Command for 8-bit interface
0027 1005          ;

0083          CLRF     PORTA
0084          CLRF     PORTB
0085          CLRF     PORTC
0086          CLRF     PORTD
0087          CLRF     PORTE
0088          BCF      T1CON, TMR1ON
0089 ;
0090          BSF      STATUS, RP0
0091          CLRF     TRISA
0092          MOVLW   0xF0
0093          MOVWF   TRISB
0094          CLRF     TRISC
0095          BSF      TRISC, T1OSO
0096          CLRF     TRISD
0097          CLRF     TRISE
0098          BSF      P1EL, TMR1IE
0099          BSF      OPTION_R, RBP0
0100          BCF      STATUS, RP0
0101 ;
0102 ;
0103 ;
0104 ; Initialize the LCD Display Module
0105 ;
0106          CLRF     LCD_CNTRL
0107
0108 DISPLAY_INIT
0109   if ( Four_bit && !Data_HI )
0110     MOVLW   0x02
0111   endif
0112 ;
0113   if ( Four_bit && Data_HI )
0114     MOVLW   0x020
0115   endif
0116 ;
0117   if ( !Four_bit )
0118     MOVLW   0x038
0119   endif
0120 ;
0121     MOVWF   LCD_DATA
0122     BSF    LCD_CNTRL, E
0123     BCF    LCD_CNTRL, 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 ;
0130 LCD_DELAY   MOVLW   LCD_INIT_DELAY
0131             MOVWF   MSD

```

```

002A 01B4          CLR    LSD
002B 0BB4          DEFSZ  LSD
002C 282B          GOTO  LOOP2
002D 0BB3          DEFSZ  MSD
002E 282B          GOTO  LOOP2
002F 3002          END_LCD_DELAY
0030 0086          GOTO  LOOP2
0031 1405          ; Command sequence for 2 lines of 5x7 characters
0032 1005          ;
0033 3008          ; Delay time = MSD * ((3 * 256) + 3) * Tcy
0034 0086          ;
0035 1405          ;
0036 1005          ;
0037 300C          ;
0038 2074          ;
0039 3001          ;
003A 2074          ;
003B 3006          ;
003C 2074          ;
003D 3080          ;
003E 2074          ;
003F 300C          ;
0040 2074          ;
0041 3001          ;
0042 2074          ;
0043 3006          ;
0044 2074          ;
0045 3006          ;
0046 2074          ;
0047 3006          ;
0048 2074          ;
0049 3006          ;
004A 2074          ;
004B 3006          ;
004C 2074          ;
004D 3006          ;
004E 2074          ;
004F 3006          ;
0050 2074          ;
0051 3006          ;
0052 2074          ;
0053 3006          ;
0054 2074          ;
0055 3006          ;
0056 2074          ;
0057 3006          ;
0058 2074          ;
0059 3006          ;
005A 2074          ;
005B 3006          ;
005C 2074          ;
005D 3006          ;
005E 2074          ;
005F 3006          ;
0060 2074          ;
0061 3006          ;
0062 2074          ;
0063 3006          ;
0064 2074          ;
0065 3006          ;
0066 2074          ;
0067 3006          ;
0068 2074          ;
0069 3006          ;
006A 2074          ;
006B 3006          ;
006C 2074          ;
006D 3006          ;
006E 2074          ;
006F 3006          ;
0070 2074          ;
0071 3006          ;
0072 2074          ;
0073 3006          ;
0074 2074          ;
0075 3006          ;
0076 2074          ;
0077 3006          ;
0078 2074          ;
0079 3006          ;
007A 2074          ;
007B 3006          ;
007C 2074          ;
007D 3006          ;
007E 2074          ;
007F 3006          ;
0080 2074          ;
0081 3006          ;
0082 2074          ;
0083 3006          ;
0084 2074          ;
0085 3006          ;
0086 2074          ;
0087 3006          ;
0088 2074          ;
0089 3006          ;
008A 2074          ;
008B 3006          ;
008C 2074          ;
008D 3006          ;
008E 2074          ;
008F 3006          ;
0090 2074          ;
0091 3006          ;
0092 2074          ;
0093 3006          ;
0094 2074          ;
0095 3006          ;
0096 2074          ;
0097 3006          ;
0098 2074          ;
0099 3006          ;
009A 2074          ;
009B 3006          ;
009C 2074          ;
009D 3006          ;
009E 2074          ;
009F 3006          ;
00A0 2074          ;
00A1 3006          ;
00A2 2074          ;
00A3 3006          ;
00A4 2074          ;
00A5 3006          ;
00A6 2074          ;
00A7 3006          ;
00A8 2074          ;
00A9 3006          ;
00AA 2074          ;
00AB 3006          ;
00AC 2074          ;
00AD 3006          ;
00AE 2074          ;
00AF 3006          ;
00B0 2074          ;
00B1 3006          ;
00B2 2074          ;
00B3 3006          ;
00B4 2074          ;
00B5 3006          ;
00B6 2074          ;
00B7 3006          ;
00B8 2074          ;
00B9 3006          ;
00BA 2074          ;
00BB 3006          ;
00BC 2074          ;
00BD 3006          ;
00BE 2074          ;
00BF 3006          ;
00C0 2074          ;
00C1 3006          ;
00C2 2074          ;
00C3 3006          ;
00C4 2074          ;
00C5 3006          ;
00C6 2074          ;
00C7 3006          ;
00C8 2074          ;
00C9 3006          ;
00CA 2074          ;
00CB 3006          ;
00CC 2074          ;
00CD 3006          ;
00CE 2074          ;
00CF 3006          ;
00D0 2074          ;
00D1 3006          ;
00D2 2074          ;
00D3 3006          ;
00D4 2074          ;
00D5 3006          ;
00D6 2074          ;
00D7 3006          ;
00D8 2074          ;
00D9 3006          ;
00DA 2074          ;
00DB 3006          ;
00DC 2074          ;
00DD 3006          ;
00DE 2074          ;
00DF 3006          ;
00E0 2074          ;
00E1 3006          ;
00E2 2074          ;
00E3 3006          ;
00E4 2074          ;
00E5 3006          ;
00E6 2074          ;
00E7 3006          ;
00E8 2074          ;
00E9 3006          ;
00EA 2074          ;
00EB 3006          ;
00EC 2074          ;
00ED 3006          ;
00EE 2074          ;
00EF 3006          ;
00F0 2074          ;
00F1 3006          ;
00F2 2074          ;
00F3 3006          ;
00F4 2074          ;
00F5 3006          ;
00F6 2074          ;
00F7 3006          ;
00F8 2074          ;
00F9 3006          ;
00FA 2074          ;
00FB 3006          ;
00FC 2074          ;
00FD 3006          ;
00FE 2074          ;
00FF 3006          ;

```

Interfacing to an LCD Module

```
0181 ;
0182 ;Send a message the hard way
0183     movlw    'M'
0184     call    SEND_CHAR
0185     movlw    'i'
0186     call    SEND_CHAR
0187     movlw    'c'
0188     call    SEND_CHAR
0189     movlw    'r'
0190     call    SEND_CHAR
0191     movlw    'o'
0192     call    SEND_CHAR
0193     movlw    'c'
0194     call    SEND_CHAR
0195     movlw    'h'
0196     call    SEND_CHAR
0197     movlw    'i'
0198     call    SEND_CHAR
0199     movlw    'p'
0200     call    SEND_CHAR
0201
0202     movlw    B'11000000' ;Address DDrAm first character, second line
0203     call    SEND_CMD
0204
0205 ;Demonstration of the use of a table to output a message
0206     movlw    0 ;Table address of start of message
0207     dispmsg
0208     movwf    TEMP1 ;TEMP1 holds start of message address
0209     call    Table
0210     andlw    0FFh ;Check if at end of message (zero
0211     btfsc   STATUS,Z ;returned at end)
0212     goto    out
0213     call    SEND_CHAR ;Display character
0214     movf    TEMP1,w ;Point to next character
0215     addlw   1
0216     goto    dispmsg
0217     out
0218     loop
0219
0220 ;
0221 ;
0222 INIT_DISPLAY
0223     MOVWLW  DISP_ON ; Display On, Cursor On
0224     CALL    SEND_CMD ; Send This command to the Display Module
0225     MOVWLW  CLR_DISP ; Clear the Display
0226     CALL    SEND_CMD ; Send This command to the Display Module
0227     MOVWLW  ENTRY_INC ; Set Entry Mode Inc., No shift
0228     CALL    SEND_CMD ; Send This command to the Display Module
```

```

0064 0008
0229 RETURN
0230 ;
0231 ;
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_CNTRL, R_W ;Set LCD to read
0254 BSF LCD_CNTRL, RS ;Set LCD to data mode
0255 BSF LCD_CNTRL, E ;toggle E for LCD
0256 BCF LCD_CNTRL, E
0257 SWAPF CHAR, W
0258 ANDLW 0xF0 ;Get lower nibble
0259 MOVWF LCD_DATA ;Send data to LCD
0260 BSF LCD_CNTRL, E ;toggle E for LCD
0261 BCF LCD_CNTRL, 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
0274 MOVWF CHAR ; Character to be sent is in W
0275 CALL BUSY_CHECK ; Wait for LCD to be ready
0276 SWAPF CHAR, W
0277 ANDLW 0x0F ; Get upper nibble
0065 00B6
0066 2083
0067 0E36
0068 390F

```

Interfacing to an LCD Module

```
0069 0086          MOVWF LCD_DATA      ; Send data to LCD
006A 1085          BCF LCD_CNTRL, R_W    ; Set LCD to read
006B 1505          BSF LCD_CNTRL, RS    ; Set LCD to data mode
006C 1405          BCF LCD_CNTRL, E    ; toggle E for LCD
006D 1005          BCF LCD_CNTRL, E
006E 0836          MOVF CHAR, W
006F 390F          ANDLW 0x0F
0070 0086          MOVWF LCD_DATA      ; Get lower nibble
0071 1405          BSF LCD_CNTRL, E    ; Send data to LCD
0072 1005          BCF LCD_CNTRL, E    ; toggle E for LCD
0073 0008          RETURN

0278          MOVWF LCD_DATA      ; Send data to LCD
0279          BCF LCD_CNTRL, R_W    ; Set LCD to read
0280          BSF LCD_CNTRL, RS    ; Set LCD to data mode
0281          BCF LCD_CNTRL, E    ; toggle E for LCD
0282          BCF LCD_CNTRL, E
0283          MOVF CHAR, W
0284          ANDLW 0x0F
0285          MOVWF LCD_DATA      ; Get lower nibble
0286          BSF LCD_CNTRL, E    ; Send data to LCD
0287          BCF LCD_CNTRL, E    ; toggle E for LCD
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
0301          CALL BUSY_CHECK
0302          MOVF CHAR, W
0303          MOVWF LCD_DATA
0304          BCF LCD_CNTRL, R_W    ; Send data to LCD
0305          BSF LCD_CNTRL, RS    ; Set LCD in read mode
0306          BSF LCD_CNTRL, E    ; Set LCD in data mode
0307          BCF LCD_CNTRL, E
0308          RETURN
0309          ;
0310          endif
0311          ;
0312          ;
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
0336     MOVWF LCD_DATA      ; Get upper nibble
0337     BCF  LCD_CNTRL,R_W  ; Send data to LCD
0338     BCF  LCD_CNTRL,RS   ; Set LCD to read
0339     BSF  LCD_CNTRL,E    ; Set LCD to command mode
0340     BCF  LCD_CNTRL,E    ; toggle E for LCD
0341     SWAPF CHAR,W
0342     ANDLW 0xF0
0343     MOVWF LCD_DATA      ; Get lower nibble
0344     BSF  LCD_CNTRL,E    ; Send data to LCD
0345     BCF  LCD_CNTRL,E    ; toggle E for LCD
0346     RETURN
0347 ;
0348     else
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
0355     MOVWF LCD_DATA      ; Get upper nibble
0356     BCF  LCD_CNTRL, R_W  ; Send data to LCD
0357     BCF  LCD_CNTRL, RS   ; Set LCD to read
0358     BSF  LCD_CNTRL, E    ; Set LCD to command mode
0359     BCF  LCD_CNTRL, E    ; toggle E for LCD
0360     MOVF  CHAR, W
0361     ANDLW 0x0F
0362     MOVWF LCD_DATA      ; Get lower nibble
0363     BSF  LCD_CNTRL, E    ; Send data to LCD
0364     BCF  LCD_CNTRL, E    ; toggle E for LCD
0365     RETURN
0366 ;
0367     endif
0368     else
0369 ;
0370 ; *****
0371 ; * SEND_CMD - Sends command contained in register W to LCD
0372 ; * This routine sends the entire character to the PORT
0373 ; * The data is transmitted on the PORT<7:0> pins

```

Interfacing to an LCD Module

```
0374 ; *****
0375
0376 SEND_CMD
0377
0378     MOVWF    CHAR
0379     CALL    BUSY_CHECK
0380     MOVF    CHAR, W
0381     MOVWF    LCD_DATA
0382     BCF    LCD_CNTL, R_W
0383     BCF    LCD_CNTL, RS
0384     BCF    LCD_CNTL, E
0385     RETURN
0386 ;
0387     endif
0388 ;
0389 ;
0390 ;
0391     if ( Four_bit )
0392 ;
0393     if ( Data_HI ) ; 4-bit Data transfers?
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
0403     BSF    STATUS, RP0
0404     MOVLW    LCD_DATA_TRIS
0405     BCF    STATUS, RP0
0406     BCF    LCD_CNTL, RS
0407     BSF    LCD_CNTL, R_W
0408     BSF    LCD_CNTL, E
0409     BCF    LCD_CNTL, E
0410     MOVF    LCD_DATA, W
0411     ANDLW    0xF0
0412     MOVWF    TEMP
0413     BSF    LCD_CNTL, E
0414     BCF    LCD_CNTL, E
0415     SWAPF    LCD_DATA, W
0416     ANDLW    0x0F
0417     IORWF    TEMP, 7
0418     BTFSF    TEMP, 7
0419     GOTO    BUSY_CHECK
0420     BCF    LCD_CNTL, R_W
0421     BSF    STATUS, RP0
0422     MOVLW    0x0F
; Command to be sent is in W
; Wait for LCD to be ready
; Send data to LCD
; Set LCD in read mode
; Set LCD in command mode
; toggle E for LCD
; 4-bit transfers on the high nibble of the PORT
; Select Register page 1
; Set Port_D for input
; Select Register page 0
; Set LCD for Command mode
; Setup to read busy flag
; Set E high
; Set E low
; Read upper nibble busy flag, DDRam address
; Mask out lower nibble
; Toggle E to get lower nibble
; Read lower nibble busy flag, DDRam address
; Mask out upper nibble
; Combine nibbles
; Check busy flag, high = busy
; If busy, check again
; Select Register page 1
```



```

0423 MOVWF LCD_DATA_TRIS ; Set Port_D for output
0424 BCF STATUS, RP0 ; Select Register page 0
0425 RETURN
0426 ;
0427 ; else ; 4-bit transfers on the low nibble of the PORT
0428 ;
0429 ;*****
0430 ;* This routine checks the busy flag, returns when not busy *
0431 ;* Affects: *
0432 ;* TEMP - Returned with busy/address *
0433 ;*****
0434 ;
0435 BUSY_CHECK
0436 BSF STATUS, RP0 ; Bank 1
0437 MOVLW 0xFF ; Set PortB for input
0438 MOVWF LCD_DATA_TRIS
0439 BCF STATUS, RP0 ; Bank 0
0440 BCF LCD_CNTRL, RS ; Set LCD for Command mode
0441 BSF LCD_CNTRL, R_W ; Setup to read busy flag
0442 BCF LCD_CNTRL, E ; Set E high
0443 BCF LCD_CNTRL, E ; Set E low
0444 SWAPF LCD_DATA, W ; Read upper nibble busy flag, DDRam address
0445 ANDLW 0xF0 ; Mask out lower nibble
0446 MOVWF TEMP ;
0447 BSF LCD_CNTRL, E ; Toggle E to get lower nibble
0448 BCF LCD_CNTRL, E ; Read lower nibble busy flag, DDRam address
0449 MOVF LCD_DATA, W ; Mask out upper nibble
0450 ANDLW 0x0F ;
0451 IORWF TEMP, F ; Combine nibbles
0452 BITFSC TEMP, 7 ; Check busy flag, high = busy
0453 GOTO BUSY_CHECK ; If busy, check again
0454 BCF LCD_CNTRL, R_W ; Bank 1
0455 BSF STATUS, RP0 ;
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

```

Interfacing to an LCD Module

```

0471 BSF STATUS,RP0 ; Select Register page 1
0472 MOVLW 0xFF ; Set port_D for input
0473 MOVWF LCD_DATA_TRIS
0474 BCF STATUS,RP0 ; Select Register page 0
0475 BCF LCD_CNTRL,RS ; Set LCD for command mode
0476 BSF LCD_CNTRL,R_W ; Setup to read busy flag
0477 BSF LCD_CNTRL,E ; Set E high
0478 BCF LCD_CNTRL,E ; Set E low
0479 MOVF LCD_DATA,w ; Read busy flag, DDRAM address
0480 MOVWF TEMP
0481 BITFSC TEMP,7 ; Check busy flag, high=busy
0482 GOTO BUSY_CHECK
0483 BCF LCD_CNTRL,R_W
0484 BSF STATUS,RP0 ; Select Register page 1
0485 MOVLW 0x00
0486 MOVWF LCD_DATA_TRIS ; Set port_D for output
0487 BCF STATUS,RP0 ; Select Register page 0
0488 RETURN
0489 ;
0490 ; endif
0491 ;
0492 ;
0493 Table
0494 addwf PCL ;Jump to char pointed to in W reg
0495 retlw 'W'
0496 retlw 'i'
0497 retlw 'c'
0498 retlw 'r'
0499 retlw 'o'
0500 retlw 'c'
0501 retlw 'h'
0502 retlw 'i'
0503 retlw 'p'
0504 retlw ' '
0505 retlw 'T'
0506 retlw 'e'
0507 retlw 'c'
0508 retlw 'h'
0509 retlw 'n'
0510 retlw 'o'
0511 retlw 'l'
0512 retlw 'o'
0513 retlw 'g'
0514 retlw 'y'
0515 Table_End
0516 retlw 0
0517 ;
0518 if ( (Table & 0x0FF) >= (Table_End & 0x0FF) )
0519 MESSG "Warning - User Defined: Table Table crosses page boundary in computed_jump"

```

```

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

```

PAGE 14

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

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

```

0000 : X-XXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX
0040 : XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX
0080 : XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX
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 labels are used with direct addressing.


```

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 corner of the Display
;

```

list

Interfacing to an LCD Module

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

All rights reserved. © 1995, Microchip Technology Incorporated, USA.

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