



## Use of the SSP Module in the I<sup>2</sup>C™ Multi-Master Environment

### INTRODUCTION

The Inter-IC (I<sup>2</sup>C) bus is a two-wire serial interface developed by Phillips/Sigmetics™. The specification supports data transmission up to 400 Kbps.

The I<sup>2</sup>C interface employs a comprehensive protocol to ensure reliable transmission and reception of data. When the bus is active, one device is the Master (generates the clock and the handshaking signals), while all the other devices are Slaves. The current bus Master can both read from and write to any of the Slave units by addressing them individually. On a Multi-Master bus the Masters follow an arbitration scheme to ensure that the bus is not corrupted.

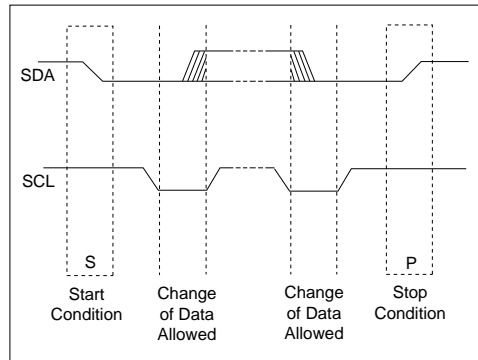
Each device attached to the I<sup>2</sup>C bus is assigned a unique address. When a Master wishes to initiate a data transfer, it first transmits the address of the device that it wishes to "talk" to. All devices "listen" to see if this is their address. Within this address, a bit specifies whether the Master wishes to read from or write to the Slave device.

The output stages of each device on the bus attached to the clock (SCL) and data (SDA) lines must have an open-drain or open-collector in order to perform the wired-AND function of the bus. External pull-up resistors are used to ensure a high level when no device is pulling the line down. The only limitation on the number of devices that may be attached to the bus is limited by the maximum bus loading specification. For complete bus specifications, refer to Philips/Sigmetics™ document "The I<sup>2</sup>C-bus and how to use it". The order number for this document is 98-8080-575.

### INITIATING AND TERMINATING DATA TRANSFER

During times of no data transfer (idle time), both SCL and SDA are pulled high. A Master device which wishes to take control of the bus must first generate a START condition. The START condition is defined as a high to low transition of SDA when SCL is high. When the Master has completed all data transmissions and wishes to relinquish the bus, it generates a STOP condition. The STOP condition is defined as a low to high transition of SDA when SCL is high. Because the START and STOP conditions are defined as transitions of the SDA when the SCL is high, the SDA line can only change when SCL is low during the actual data transmission. Figure 1 shows the relationship between SCL and SDA for the various conditions.

FIGURE 1: START AND STOP CONDITIONS

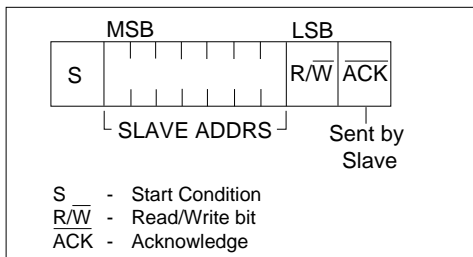


# I<sup>2</sup>C Multi-Master Environment

## ADDRESSING I<sup>2</sup>C DEVICES

There are two address formats. The simplest is the 7-bit address format with a R/W bit (see Figure 2). The more complex is the 10-bit address with a R/W bit (see Figure 3). For 10-bit addressing, two bytes must be transmitted with the first five bits specifying this to be a 10-bit address. Only 7-bit addressing is used in this application note.

**FIGURE 2: 7-BIT ADDRESS FORMAT**



## TRANSFER ACKNOWLEDGE

Slave as receiver:

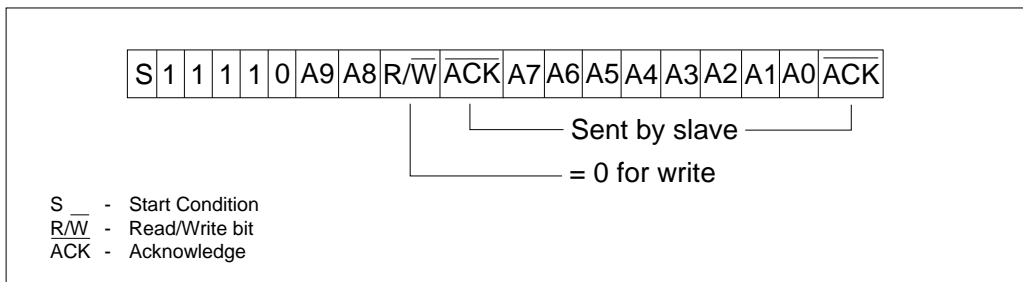
All data is transmitted as bytes, with no limit to the number of bytes transmitted per data transfer. After each byte, the slave-receiver generates an acknowledge bit (ACK) by pulling the SDA line low. When a slave-receiver doesn't acknowledge the slave address or received data, the master aborts the transfer. Whether the ACK bit is generated or not, SDA must be released by the slave so that the master can generate the STOP condition.

Master as receiver:

If the master is receiving the data, it generates an acknowledge signal for each received byte of data except for the last byte. To signal the end of data to the slave-transmitter, the master does not generate an acknowledge. The slave then releases the SDA line so the master can generate the STOP condition. The master can also generate the STOP condition during the acknowledge pulse for valid termination of data transfer.

If the slave needs to delay the transmission of the next byte, holding the SCL line low will force the master into a wait state. Data transfer continues when the slave releases the SCL line. This allows the slave to move the received data or fetch the data it needs to transfer before allowing the clock to start. This wait state technique can also be implemented at the bit level.

**FIGURE 3: 10-BIT ADDRESS FORMAT**



## MULTI-MASTER

The I<sup>2</sup>C protocol allows a system to have more than one master. When two or more masters try to transfer data at the same time, arbitration and synchronization occur.

Arbitration:

Arbitration takes place on the SDA line while the SCL line is high. The master which transmits a high when the other master transmits a low loses arbitration (see Figure 4) and turns off its data output stage. A master which lost arbitration can generate clock pulses until the end of the data byte where it lost arbitration. When the master devices are addressing the same device, arbitration continues into the data.

Clock Synchronization:

Clock synchronization occurs after the devices have started arbitration. This is performed using a wired-AND connection to the SCL line. A high to low transition on the SCL line causes the concerned devices to start counting off their low period. Once a device clock has gone low, it will hold the SCL line low until its SCL high state is reached. The low to high transition of this clock may not change the state of the SCL line, if another device clock is still within its low period. The SCL line is held low by the device with the longest low period. Devices with shorter low periods enter a high wait-state, until the SCL line comes high. When the SCL line comes high, all devices start counting off their high periods. The first device to complete its high period will pull the SCL line low. The SCL line high time is determined by the device with the shortest high period. This is shown in Figure 5.

FIGURE 4: MULTI-MASTER ARBITRATION

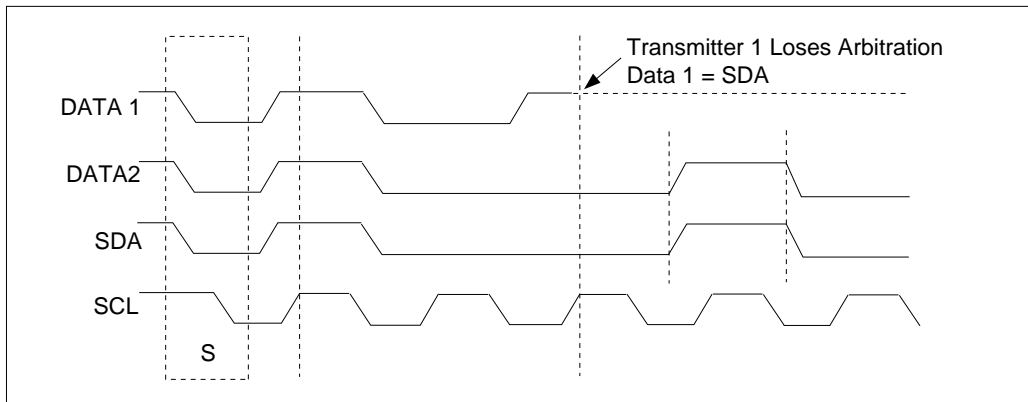
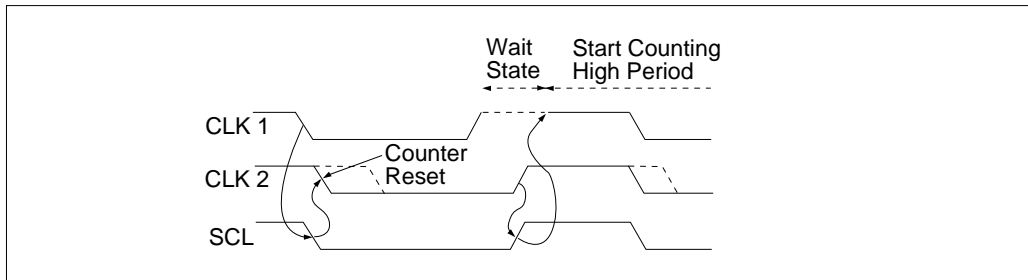


FIGURE 5: CLOCK SYNCHRONIZATION



# I<sup>2</sup>C Multi-Master Environment

---

## IMPLEMENTATION IN THE PIC16CXX

This Application Note uses the PIC16CXX in a Multi-Master I<sup>2</sup>C environment. The PIC16CXX acts as both a Master and a Slave on the bus.

### Hardware:

The demonstration hardware consists of a keypad multiplexed with eight LEDs on Port B and connections to the I<sup>2</sup>C through the RC3/SCL and RC4/SDA pins. (See Figure 6).

### Software:

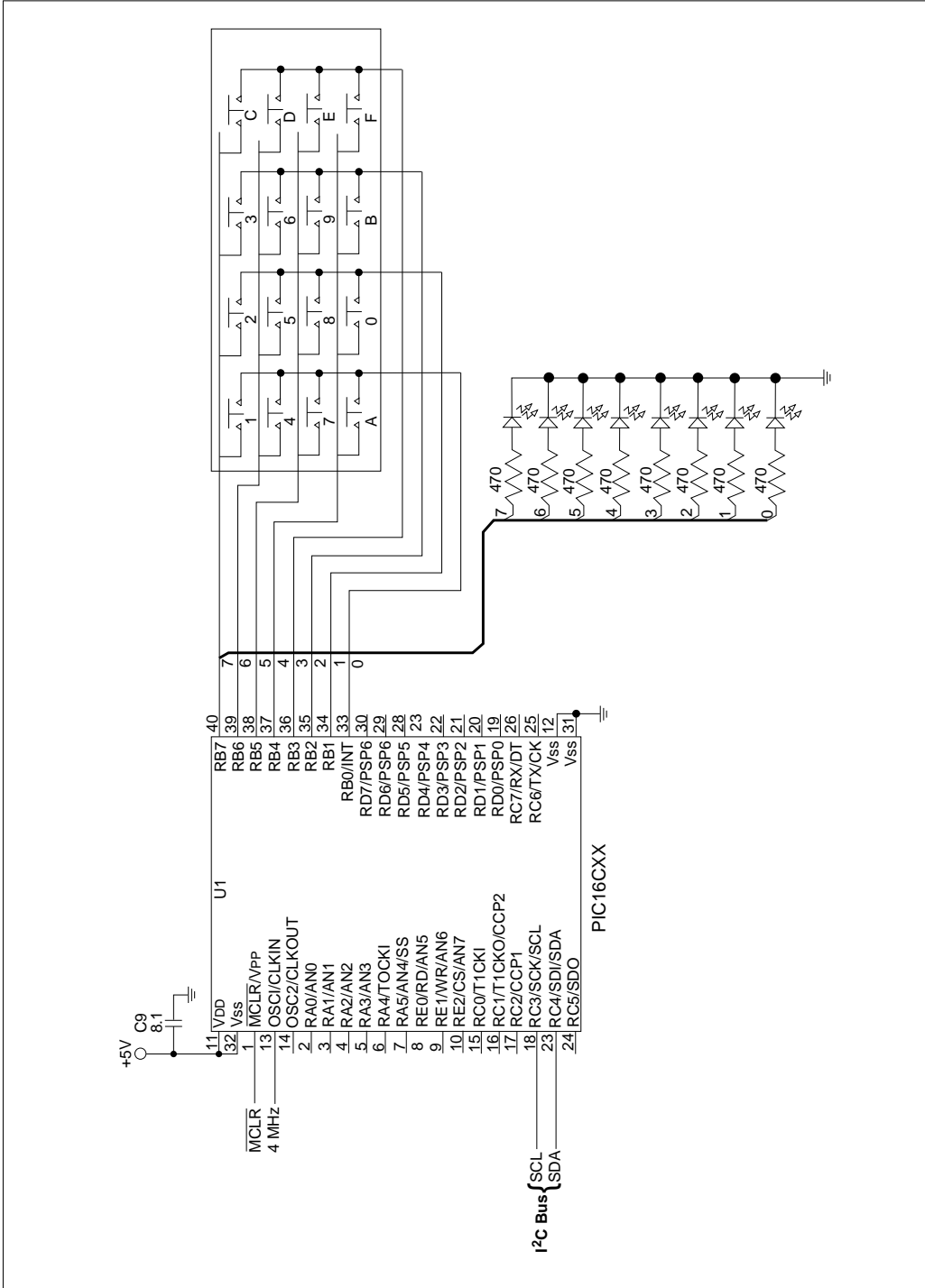
The software transmits in Master mode and receives data in Slave mode.

The transmit routine scans the keypad, debounces the keypresses, and transmits their encoded value in Master mode over the I<sup>2</sup>C bus to slave address A6. Before transmitting, the status of the Synchronous Serial Port (SSP) is checked. No data is sent until the SSP status register indicates that a Stop bit was received. If the transmission causes any errors, the error code will be displayed on the LEDs, and transmission will again be attempted.

Data received in Slave mode at address A2 from the bus is displayed on the LEDs. Slave reception is interrupt driven. When a complete word is received with the proper address, the processor is interrupted, and the program verifies that an SSP interrupt was received. Once it has been verified that the interrupt was caused by the SSP, and that the buffer is full, the data word is read and output to the LEDs.

*Author: Scott Fink, Logic Products Division*

FIGURE 6: HARDWARE USED FOR THIS AP-NOTE



# I<sup>2</sup>C Multi-Master Environment

MPASM 01.00.02 Alpha SCAPPNOT\IIC 5-10-1994 13:2:14  
Demonstration of I2C MultiMaster mode

PAGE 1

```
LOC OBJECT CODE      LINE SOURCE TEXT

                                0001      TITLE " Demonstration of I2C MultiMaster mode"
                                0002 ;
                                0003      LIST P=16C64, F=INHX8M
                                0004 ;
                                0005
                                ;*****
0006 ;** Two wire/I2C Bus MultiMaster sample routines for Microchip's
0007 ;** PIC16C64 8-bit CMOS single chip microcomputer
0008 ;** Revised Version (3/06/94).
0009 ;**
0010 ;** Part used = PIC16C64
0011 ;** Note: 1) All timings are based on a reference crystal frequency of
0012 ;**          4MHz which is equivalent to an instruction cycle time of
0013 ;**          of 1 usec.
0014 ;**          2) Address and literal values are read in decimal unless
0015 ;**          otherwise specified.
                                ;*****
0016 ;
0017 ;-----
0018 ;          File Register Assignment
0019 ;-----
0020 ;
0021 include "\picmastr\picdemii\picreg.equ"
0022 ;
0023 FLAG EQU 20h ; Common flag bits register
0024 EEPROM EQU 21h ; Bit buffer
0025 ERCODE EQU 22h ; Error code (to indicate bus status)
0026 ADDR EQU 23h ; Address register
0027 DATAI EQU 24h ; Stored data input register
0028 DATAO EQU 25h ; Stored data output register
0029 SLAVE EQU 26h ; Device address (1010xxx0)
0030 TXBUF EQU 27h ; TX buffer
0031 RXBUF EQU 28h ; RX buffer
0032 COUNT EQU 29h ; Bit counter
0033 TEMP EQU 2Ah ; Temporary storage
0034 ROW EQU 2Bh ; Keypad row
0035 NEW_KEY EQU 2Ch ; Storage for latest key
0036 OLD_KEY EQU 2Dh ; Storage for last key pressed
0037 DISPVAL EQU 2Eh ; Value displayed on LEDs
0038 TEMP1 EQU 2Fh ; Scratchpad register
0039 TEMP_W EQU 30h ; Storage for W register
0040 TEMP_STAT EQU 31h ; Storage for STATUS register
0041 ;
0042 ;-----
0043 ;          Bit Assignments
0044 ;-----
0045
0046 ; FLAG Bits
0047
0000 0048 ERROR EQU 0 ; Error flag
0049
0050 ; EEPROM Bits
```

# I<sup>2</sup>C Multi-Master Environment

MPASM 01.00.02 Alpha SCAPPNOT\IIC 5-10-1994 13:2:14  
Demonstration of I2C MultiMaster mode

PAGE 2

```
LOC OBJECT CODE          LINE SOURCE TEXT
                                0051
0007          0052 DI      EQU      7          ; EEPROM input
0006          0053 DO      EQU      6          ; EEPROM output
                                0054
                                0055 ; I2C Device Bits
                                0056
0004          0057 SDA     EQU      4          ; RB7, data in/out
0003          0058 SCL     EQU      3          ; RB6, serial clock
                                0059
                                0060 ;End of files/bits equate
                                0061
                                0062          ORG      00h          ; Reset Vector
0000 2810          0063          goto    starting
                                0064
                                0065          ORG      04h          ; Interrupt Vector
0004 28EB          0066          goto    service_int
                                0067
                                0068          ORG      10h          ; Begining of Program space
                                0069 starting
0010 01AE          0070          clrf    DISPVAL      ; Blank out LEDs
0011 01A0          0071          clrf    FLAG          ; Clear error register
0012 0187          0072          clrf    PORT_C      ; Set SDA, SCL low when not is tri-state
0013 3008          0073          movlw   08h
0014 00AB          0074          movwf  ROW          ; Set initial row to be strobed
0015 303E          0075          movlw  B'00111110' ; I2C 7 bit slave mode with master
0016 0094          0076          movwf  SSPCON      ; mode enabled
0017 1683          0077          bsf    STATUS,RP0  ; Select page 1
0018 0186          0078          clrf  TRIS_B      ; Set PORT_B to all outputs
0019 3008          0079          movlw  B'00001000' ; Enable SSP interrupt
001A 008C          0080          movwf  PIE1
001B 30A2          0081          movlw  b'10100010' ; Slave address
001C 0093          0082          movwf  SSPADD
001D 1587          0083          bsf    TRIS_C,3    ; Set SCL high
001E 1607          0084          bsf    TRIS_C,4    ; Set SDA high
001F 1283          0085          bcf    STATUS,RP0  ; Select page 0
0020 118C          0086          bcf    PIR1,3      ; Clear SSP interrupt flag
0021 30C0          0087          movlw  B'11000000' ; Enable interrupts
0022 008B          0088          movwf  INTCON
                                0089
                                0090 KbdWait
0023 01AA          0091          clrf  TEMP
0024 216B          0092          call  SetupDelay
0025 2102          0093          call  ScanKbd      ; Check for key pressed
0026 1283          0094          bcf    STATUS,RP0
0027 082C          0095          movf  NEW_KEY,W    ; Get latest key
0028 39FF          0096          andlw  0FFh       ; Key pressed?
0029 1903          0097          btfsz STATUS,Z
002A 2823          0098          goto  KbdWait     ; No, go check again
002B 00A5          0099          movwf  DATA0     ; Yes, output it on I2C bus
002C 30A6          0100          movlw  B'10100110' ; Address of device being addressed
002D 00A6          0101          movwf  SLAVE
002E 1683          0102          bsf    STATUS,RP0
                                0103 CheckAgain
```

3

# I<sup>2</sup>C Multi-Master Environment

MPASM 01.00.02 Alpha SCAPPNOT\IIC 5-10-1994 13:2:14  
Demonstration of I2C MultiMaster mode

PAGE 3

```
LOC OBJECT CODE      LINE SOURCE TEXT
002F 1A14             0104      btfsc   SSPSTAT,4      ;If STOP bit received last...
0030 2833             0105      goto    Goxmit         ; OK to transmit
0031 1994             0106      btfsc   SSPSTAT,3      ;If START bit recieved last...
0032 282F             0107      goto    CheckAgain     ; wait for STOP bit
                        0108      Goxmit
0033 1283             0109      bcf     STATUS,RP0
0034 138B             0110      bcf     INTCON,7       ; Disable interrupts
0035 203F             0111      call    WRBYTE         ; Output byte
0036 1C20             0112      btfss   FLAG,ERROR     ; Check for error
0037 283D             0113      goto    Checkout       ; No error, go on
0038 0822             0114      movf    ERCODE,W       ; Get error code
0039 0086             0115      movwf   PORT_B         ; Put error code on LEDs
003A 00AE             0116      movwf   DISPVAL
003B 1020             0117      bcf     FLAG,ERROR     ; Clear error flag
003C 282F             0118      goto    CheckAgain
                        0119      Checkout
003D 178B             0120      bsf     INTCON,7       ; Enable interrupts
003E 2823             0121      goto    KbdWait
                        0122
                        0123 ;-----
0124 ;          BYTE-WRITE, write one byte to I2C (Master Mode)
0125 ;-----
0126 ;          Input   :          DATAO= data to be written
0127 ;                  ADDR   = destination address
0128 ;                  SLAVE  = device address (1010xxx0)
0129 ;          Output :          Data written to EEPROM device
0130 ;-----
0131 ;
0132      WRBYTE
003F 1283             0133      bcf     STATUS,RP0
0040 0826             0134      movf    SLAVE,W        ; Send SLAVE address
0041 00A7             0135      movwf   TXBUF          ; to TX buffer
0042 20BF             0136      call    BSTART         ; Generate START bit
0043 2071             0137      call    TX             ; Output SLAVE data address
0044 1283             0138      bcf     STATUS,RP0
0045 0825             0139      movf    DATAO,W       ; Move DATA
0046 00A7             0140      movwf   TXBUF          ; into transmit buffer
0047 2071             0141      call    TX             ; Output DATA and detect
                        ; acknowledgement
0048 20CD             0142      call    BSTOP          ; Generate STOP bit
0049 0008             0143      return
                        0144
                        0145 ;-----
0146 ;          BYTE-READ, read one byte from I2C (Master Mode)
0147 ;-----
0148 ;          Input   :          ADDR   = source address
0149 ;                  SLAVE  = device address (1010xxx0)
0150 ;          Output :          DATAI  = data read from serial EEPROM
0151 ;-----
0152
0153      RDBYTE
004A 1283             0154      bcf     STATUS,RP0
004B 0826             0155      movf    SLAVE,W        ; Move SLAVE address
004C 00A7             0156      movwf   TXBUF          ; into buffer (R/W = 0)
```



# I<sup>2</sup>C Multi-Master Environment

MPASM 01.00.02 Alpha SCAPPNOT\IIC 5-10-1994 13:2:14  
Demonstration of I2C MultiMaster mode

PAGE 4

```
LOC OBJECT CODE      LINE SOURCE TEXT
004D 20BF            0157      call    BSTART          ; Generate START bit
004E 2071            0158      call    TX              ; Output SLAVE address. Check ACK.
004F 1283            0159      bcf    STATUS,RP0      ;
0050 0823            0160      movf   ADDR,W          ; Put slave data address into
0051 00A7            0161      movwf  TXBUF           ; Xmit buffer
0052 2071            0162      call    TX              ; Output WORD address. Check ACK.
0053 20BF            0163      call    BSTART         ; START READ
0054 1283            0164      bcf    STATUS,RP0      ;
0055 0826            0165      movf   SLAVE,W         ;
0056 00A7            0166      movwf  TXBUF           ;
0057 1427            0167      bsf    TXBUF,0         ; Specify READ mode (R/W = 1)
0058 2071            0168      call    TX              ; Output SLAVE address
0059 205F            0169      call    RX              ; READ in data and acknowledge
005A 20CD            0170      call    BSTOP          ; Generate STOP bit
005B 1283            0171      bcf    STATUS,RP0      ;
005C 0828            0172      movf   RXBUF,W         ; Save data from buffer
005D 00A4            0173      movwf  DATAI          ; to DATAI file.
005E 0008            0174      return
0175
0176 ;-----
0177 ; RECEIVE eight data bits subroutine
0178 ;-----
0179 ; Input   :      None
0180 ; Output  :      RXBUF = 8-bit data received
0181 ;-----
0182
0183 RX
005F 1283            0184      bcf    STATUS,RP0
0060 3008            0185      movlw  .8              ; 8 bits of data
0061 00A9            0186      movwf  COUNT
0062 01A8            0187      clrf  RXBUF
0188 ;
0189 RXLPL
0063 0DA8            0190      rlf   RXBUF           ; Shift data to buffer
0064 1C03            0191      btfs  3,0
0065 1028            0192      bcf   RXBUF,0         ; carry -> f(0)
0066 1803            0193      btfs  3,0
0067 1428            0194      bsf   RXBUF,0
0068 2087            0195      call  BITIN
0069 1283            0196      bcf   STATUS,RP0
006A 1BA1            0197      btfs  EEPROM,DI
006B 1428            0198      bsf   RXBUF,0         ; Input bit =1
006C 0BA9            0199      decfsz COUNT          ; 8 bits?
006D 2863            0200      goto  RXLP
006E 1721            0201      bsf   EEPROM,DO       ; Set acknowledge bit = 1
006F 209E            0202      call  BITOUT          ; to STOP further input
0070 3400            0203      retlw 0
0204
0205 ;-----
0206 ; TRANSMIT 8 data bits subroutine
0207 ;-----
0208 ; Input   :      TXBUF
0209 ; Output  :      Data X'mitted to EEPROM device
```

3

# I<sup>2</sup>C Multi-Master Environment

MPASM 01.00.02 Alpha SCAPPNOT\IIC 5-10-1994 13:2:14  
Demonstration of I2C MultiMaster mode

PAGE 5

```
LOC OBJECT CODE      LINE SOURCE TEXT
                                0210 ;-----
                                0211
                                0212 TX
0071 1283             0213      bcf     STATUS,RP0
0072 3008             0214      movlw  .8
0073 00A9             0215      movwf  COUNT
                                0216 ;
                                0217 TXLP
0074 1321             0218      bcf     EEPROM,DO      ; Shift data bit out.
0075 1BA7             0219      btfsz  TXBUF,7        ; If shifted bit = 0, data bit = 0
0076 1721             0220      bsf     EEPROM,DO      ; Otherwise data bit = 1
0077 209E             0221      call   BITOUT         ; Serial data out
0078 1283             0222      bcf     STATUS,RP0
0079 0DA7             0223      rlf    TXBUF         ; Rotate TXBUF left
007A 1C03             0224      btfsz  3,0           ; f(6) -> f(7)
007B 1027             0225      bcf    TXBUF,0       ; f(7) -> carry
007C 1803             0226      btfsz  3,0           ; carry -> f(0)
007D 1427             0227      bsf    TXBUF,0
007E 0BA9             0228      decfsz COUNT        ; 8 bits done?
007F 2874             0229      goto  TXLP         ; No.
0080 2087             0230      call   BITIN        ; Read acknowledge bit
0081 1283             0231      bcf    STATUS,RP0
0082 3003             0232      movlw  3
0083 1BA1             0233      btfsz  EEPROM,DI     ; Check for acknowledgement
0084 20DE             0234      call   ERR          ; No acknowledge from device
0085 1283             0235      bcf    STATUS,RP0
0086 3400             0236      retlw  0
                                0237
                                0238 ;-----
                                0239 ;      Single bit receive from I2C to PIC
                                0240 ;-----
                                0241 ;      Input   :      None
                                0242 ;      Output  :      Data bit received
                                0243 ;-----
                                0244
                                0245 BITIN
0087 1683             0246      bsf    STATUS,RP0
0088 1607             0247      bsf    TRIS_C,SDA    ; Set SDA for input
0089 1283             0248      bcf    STATUS,RP0
008A 13A1             0249      bcf    EEPROM,DI
008B 1683             0250      bsf    STATUS,RP0
008C 1587             0251      bsf    TRIS_C,SCL    ; Clock high
008D 3001             0252      movlw  1
008E 1283             0253      bcf    STATUS,RP0
008F 1987             0254      btfsz  PORT_C,SCL    ; Skip if SCL is high
0090 2895             0255      goto  BIT1
0091 1283             0256      bcf    STATUS,RP0
0092 1C20             0257      btfsz  FLAG,ERROR    ; Remain as first error encountered
0093 00A2             0258      movwf  ERCODE        ; Save error code
0094 1420             0259      bsf    FLAG,ERROR    ; Set error flag
                                0260 BIT1
0095 1283             0261      bcf    STATUS,RP0
0096 1E07             0262      btfsz  PORT_C,SDA    ; Read SDA pin, for ACK low
```

# I<sup>2</sup>C Multi-Master Environment

MPASM 01.00.02 Alpha SCAPPNOT\IIC 5-10-1994 13:2:14  
Demonstration of I2C MultiMaster mode

PAGE 6

```
LOC OBJECT CODE      LINE SOURCE TEXT
0097 289A            0263      goto    ACKOK
0098 1283            0264      bcf     STATUS,RP0
0099 17A1            0265      bsf     EEPROM,DI      ; DI = 1
                        0266 ACKOK
009A 1683            0267      bsf     STATUS,RP0
009B 0000            0268      nop
                        ; Delay
009C 1187            0269      bcf     TRIS_C,SCL    ; Return SCL to low
009D 3400            0270      retlw  0
                        0271
                        0272 ; _____
0273 ;          Single bit data transmit from PIC to I2C
0274 ; _____
0275 ;          Input   :      EEPROM register, bit D0
0276 ;          Output  :      Bit transmitted over I2C
0277 ;                      Error bits set as necessary
0278 ; _____
0279
0280 BITOUT
009E 1283            0281      bcf     STATUS,RP0
009F 1F21            0282      btfs   EEPROM,DO
00A0 28AB            0283      goto   BIT0
00A1 1683            0284      bsf     STATUS,RP0
00A2 1607            0285      bsf     TRIS_C,SDA    ; Output bit 0
00A3 3002            0286      movlw  2
00A4 1283            0287      bcf     STATUS,RP0
00A5 1A07            0288      btfs   PORT_C,SDA    ; Check for error code 2
00A6 28B0            0289      goto   CLK1
00A7 1C20            0290      btfs   FLAG,ERROR    ; Remain as first error encountered
00A8 00A2            0291      movwf  ERCODE        ; Save error code
00A9 1420            0292      bsf     FLAG,ERROR    ; Set error flag
00AA 28B0            0293      goto   CLK1          ; SDA locked low by device
                        0294 ;
0295 BIT0
00AB 1683            0296      bsf     STATUS,RP0
00AC 1207            0297      bcf     TRIS_C,SDA    ; Output bit 0
00AD 0000            0298      nop
                        ; Delay
00AE 0000            0299      nop
00AF 0000            0300      nop
                        0301 CLK1
00B0 1683            0302      bsf     STATUS,RP0
00B1 1587            0303      bsf     TRIS_C,SCL
00B2 3001            0304      movlw  1              ; Error code 1
00B3 1283            0305      bcf     STATUS,RP0
00B4 1987            0306      btfs   PORT_C,SCL    ; SCL locked low?
00B5 28BA            0307      goto   BIT2          ; No.
00B6 1283            0308      bcf     STATUS,RP0
00B7 1C20            0309      btfs   FLAG,ERROR    ; Yes.
00B8 00A2            0310      movwf  ERCODE        ; Save error code
00B9 1420            0311      bsf     FLAG,ERROR    ; Set error flag
                        0312 BIT2
00BA 0000            0313      nop
00BB 0000            0314      nop
00BC 1683            0315      bsf     STATUS,RP0
```

3

# I<sup>2</sup>C Multi-Master Environment

MPASM 01.00.02 Alpha SCAPPNOT\IIC 5-10-1994 13:2:14  
Demonstration of I2C MultiMaster mode

PAGE 7

```
LOC OBJECT CODE      LINE SOURCE TEXT
00BD 1187            0316      bcf     TRIS_C,SCL      ; Return SCL to low
00BE 3400            0317      retlw   0
                                0318
                                0319 ;-----
                                0320 ;          START bit generation routine
                                0321 ;-----
                                0322 ;          input   : none
                                0323 ;          output  : initialize bus communication
                                0324 ;-----
                                0325
                                0326 ;Generate START bit (SCL is high while SDA goes from high to low
; transition)
                                0327 ;and check status of the serial clock.
                                0328 BSTART
00BF 1683            0329      bsf     STATUS,RP0
00C0 1607            0330      bsf     TRIS_C,SDA      ; Make sure SDA is high
00C1 1587            0331      bsf     TRIS_C,SCL      ; Set clock high
00C2 3001            0332      movlw  1                ; Ready error status code 1
00C3 1283            0333      bcf     STATUS,RP0
00C4 1D87            0334      btfs   PORT_C,SCL      ; Locked?
00C5 20DE            0335      call   ERR              ; SCL locked low by device, flag error
00C6 1683            0336      bsf     STATUS,RP0
00C7 1207            0337      bcf     TRIS_C,SDA      ; SDA goes low during SCL high
00C8 0000            0338      nop
                                ; Timing adjustment, 1.5uS @2MHz
00C9 0000            0339      nop
00CA 0000            0340      nop
00CB 1187            0341      bcf     TRIS_C,SCL      ; Start clock train
00CC 3400            0342      retlw   0
                                0343
                                0344 ;-----
                                0345 ;          STOP bit generation routine
                                0346 ;-----
                                0347 ;          Input   :      None
                                0348 ;          Output  :      Bus communication, STOP condition
                                0349 ;-----
                                0350
                                0351 ;Generate STOP bit (SDA goes from low to high during SCL high state)
                                0352 ;and check bus conditions.
                                0353
                                0354 BSTOP
00CD 1683            0355      bsf     STATUS,RP0
00CE 1207            0356      bcf     TRIS_C,SDA      ; Return SDA to low
00CF 1587            0357      bsf     TRIS_C,SCL      ; Set SCL high
00D0 0000            0358      nop
00D1 0000            0359      nop
00D2 0000            0360      nop
00D3 3001            0361      movlw  1                ; Ready error code 1
00D4 1283            0362      bcf     STATUS,RP0
00D5 1D87            0363      btfs   PORT_C,SCL      ;High?
00D6 20DE            0364      call   ERR              ; No, SCL locked low by device
00D7 1683            0365      bsf     STATUS,RP0
00D8 1607            0366      bsf     TRIS_C,SDA      ; SDA goes from low to high during SCL high
00D9 3004            0367      movlw  4                ; Ready error code 4
00DA 1E07            0368      btfs   TRIS_C,SDA      ;High?
```

# I<sup>2</sup>C Multi-Master Environment

MPASM 01.00.02 Alpha SCAPPNOT\IIC 5-10-1994 13:2:14  
Demonstration of I2C MultiMaster mode

PAGE 8

```
LOC OBJECT CODE      LINE SOURCE TEXT
00DB 20DE            0369      call    ERR          ; No, SDA bus not release for STOP
00DC 1283            0370      bcf     STATUS,RP0
00DD 3400            0371      retlw   0
0372
0373 ;-----
0374 ; Two wire/I2C - CPU communication error status table and subroutine
0375 ;-----
0376 ; input  :      W-reg   = error code
0377 ; output :      ERCODE  = error code
0378 ;          FLAG(ERROR) = 1
0379 ;
0380 ;          code          error status mode
0381 ;-----
0382 ;          1 :   SCL locked low by device (bus is still busy)
0383 ;          2 :   SDA locked low by device (bus is still busy)
0384 ;          3 :   No acknowledge from device (no handshake)
0385 ;          4 :   SDA bus not released for master to generate STOP bit
0386 ;-----
0387 ;
0388 ;Subroutine to identify the status of the serial clock (SCL) and serial
0389 ;data (SDA) condition according to the error status table. Codes
0390 ;generated are useful for bus/device diagnosis.
0391
0392 ERR
00DE 1283            0393      bcf     STATUS,RP0
00DF 1C20            0394      btfsz  FLAG,ERROR    ; Keep first error reported
00E0 00A2            0395      movwf  ERCODE        ; Save error code
00E1 1420            0396      bsf    FLAG,ERROR    ; Set error flag
00E2 3400            0397      retlw   0
0398
0399 ;-----
0400 ;          DELAY, Provide a 1.54mS delay (@ 4 MHz clock)
0401 ;-----
0402 ;          Input   :      None
0403 ;          Output  :      None
0404 ;-----
0405
0406 delay
00E3 1283            0407      bcf     STATUS,RP0
00E4 01AA            0408      clrf   TEMP          ;clear last location
0409 dly1
00E5 0000            0410      nop
00E6 0000            0411      nop
00E7 0000            0412      nop
00E8 0BAA            0413      decfsz TEMP          ;reduce count
00E9 28E5            0414      goto  dly1          ;loop
00EA 3400            0415      retlw   0
0416
0417 ;-----
0418 ; Interrupt service routine
0419 ; Only the SSP interrupt is enabled. This routine will read the I2C
0420 ; data and output it on the LEDs.
0421 ;-----
```

3

# I<sup>2</sup>C Multi-Master Environment

MPASM 01.00.02 Alpha SCAPPNOT\IIC 5-10-1994 13:2:14  
Demonstration of I2C MultiMaster mode

PAGE 9

```
LOC  OBJECT CODE      LINE SOURCE TEXT
                                0422
                                0423 service_int
00EB 118C          0424      bcf      PIR1,3      ; Clear SSP interrupt
00EC 00B0          0425      movwf   TEMP_W      ; Save W register
00ED 0E03          0426      swapf  STATUS,W    ; Get STATUS register
00EE 00B1          0427      movwf  TEMP_STAT   ; Save STATUS register
00EF 1683          0428      bsf    STATUS,RP0
00F0 1C14          0429      btfss  SSPSTAT,0   ; Check Buffer Full Flag
00F1 28FA          0430      goto   IntOut      ; No data received, so exit
00F2 1283          0431      bcf    STATUS,RP0
00F3 0813          0432      movf   SSPBUF,W    ; Get I2C data
00F4 1683          0433      bsf    STATUS,RP0
00F5 1E94          0434      btfss  SSPSTAT,5   ; If Address received last...
00F6 28FA          0435      goto   IntOut      ; exit without saving it
00F7 1283          0436      bcf    STATUS,RP0
00F8 0086          0437      movwf  PORT_B      ; Display received data on LEDs
00F9 00AE          0438      movwf  DISPVAL
                                0439 IntOut
00FA 1683          0440      bsf    STATUS,RP0
00FB 158C          0441      bsf    PIE1,3      ; Re-enable SSP interrupt
00FC 1283          0442      bcf    STATUS,RP0
00FD 0E31          0443      swapf  TEMP_STAT,W ; Restore STATUS register
00FE 0083          0444      movwf  STATUS
00FF 0EB0          0445      swapf  TEMP_W,1
0100 0E30          0446      swapf  TEMP_W,W    ; Restore W register
0101 0009          0447      retfie
                                0448
                                0449 ;-----
0450 ; Keyboard scan routine
0451 ; This routine scans the keypad connected to PORT_B, and
0452 ; returns the pressed key in New_Key.
0453 ;-----
0454
0455 ScanKbd
0102 1283          0456      bcf    STATUS,RP0
0103 01AC          0457      clrf  NEW_KEY      ; Clear key register
0104 082D          0458      movf  OLD_KEY,W    ; If key was pressed last pass through
0105 1D03          0459      btfss  STATUS,Z    ; goto Debounce
0106 291C          0460      goto  Debounce
                                0461 Kbdloop
0107 1003          0462      bcf    STATUS,C
0108 0DAB          0463      rlf   ROW          ; Select next row to strobe
0109 1C03          0464      btfss  STATUS,C
010A 290D          0465      goto  Notdone
010B 3010          0466      movlw 010h        ; Start over at first row
010C 00AB          0467      movwf  ROW
                                0468 Notdone
010D 0186          0469      clrf  PORT_B
010E 1683          0470      bsf   STATUS,RP0
010F 300F          0471      movlw 00Fh        ; Set PORT_B for keypad read
0110 0086          0472      movwf  TRIS_B
0111 1283          0473      bcf   STATUS,RP0
0112 082B          0474      movf  ROW,W        ; Output Row
```

# I<sup>2</sup>C Multi-Master Environment

MPASM 01.00.02 Alpha SCAPPNOT\IIC 5-10-1994 13:2:14  
Demonstration of I2C MultiMaster mode

PAGE 10

```
LOC  OBJECT CODE      LINE SOURCE TEXT
0113 0086          0475      movwf  PORT_B          ; Read colums
0114 300F          0476      movlw  0Fh            ; Mask out rows
0115 0506          0477      andwf  PORT_B,W      ; Check for Key press
0116 1903          0478      btfs   STATUS,Z
0117 2933          0479      goto   KBDOUT        ; No key pressed, exit
0118 0186          0480      clrf   PORT_B
0119 042B          0481      iorwf  ROW,W         ; Key pressed, save it
011A 00AC          0482      movwf  NEW_KEY
011B 00AD          0483      movwf  OLD_KEY
          0484      Debounce
011C 0186          0485      clrf   PORT_B
011D 1683          0486      bsf    STATUS,RP0
011E 0186          0487      clrf   TRIS_B
011F 1283          0488      bcf    STATUS,RP0
0120 082E          0489      movf   DISPVAL,W    ; Set LEDs
0121 0086          0490      movwf  PORT_B
0122 01AA          0491      clrf   TEMP
0123 216B          0492      call   SetupDelay
0124 216B          0493      call   SetupDelay    ; Delay for key debounce
0125 0186          0494      clrf   PORT_B
0126 1683          0495      bsf    STATUS,RP0
0127 300F          0496      movlw  0Fh            ; Set PORT_B for keypad read
0128 0086          0497      movwf  TRIS_B
0129 1283          0498      bcf    STATUS,RP0
012A 082B          0499      movf   ROW,W
012B 0086          0500      movwf  PORT_B        ; Output Row
012C 082D          0501      movf   OLD_KEY,W
012D 0606          0502      xorwf  PORT_B,w      ; Compare key with last key pressed
012E 1903          0503      btfs   STATUS,Z
012F 2933          0504      goto   KBDOUT
0130 0186          0505      clrf   PORT_B        ; Key released, clear registers
0131 01AC          0506      clrf   NEW_KEY
0132 01AD          0507      clrf   OLD_KEY
          0508      KBDOUT
0133 082E          0509      movf   DISPVAL,W    ; Set LEDs
0134 0086          0510      movwf  PORT_B
0135 1683          0511      bsf    STATUS,RP0
0136 0186          0512      clrf   TRIS_B
0137 1283          0513      bcf    STATUS,RP0
          0514      KEY_DEC
0138 300F          0515      movlw  00Fh
0139 052C          0516      andwf  NEW_KEY,W    ; Get column of key pressed
013A 00AA          0517      movwf  TEMP
013B 1903          0518      btfs   STATUS,Z      ; If no key pressed, exit
013C 2955          0519      goto   DECOUT
013D 30FF          0520      movlw  0FFh         ; Initialize the W register
          0521      DECL1
013E 3E01          0522      addlw  001h         ; Count column
013F 0CAA          0523      rrf    TEMP          ; Rotate column until it is found
0140 1C03          0524      btfs   STATUS,C
0141 293E          0525      goto   DECL1
0142 00AA          0526      movwf  TEMP
0143 0E2C          0527      swapf  NEW_KEY,W    ; Get row of key pressed
```

3

# I<sup>2</sup>C Multi-Master Environment

MPASM 01.00.02 Alpha SCAPPNOT\IIC 5-10-1994 13:2:14  
Demonstration of I2C MultiMaster mode

PAGE 11

```
LOC OBJECT CODE      LINE SOURCE TEXT
0144 390F            0528      andlw  00Fh
0145 1903            0529      btfsc  STATUS,Z
0146 2955            0530      goto   DECOUT      ;If no key pressed, exit
0147 00AF            0531      movwf  TEMP1
0148 30FF            0532      movlw  0FFH
                   0533      DECL2
0149 3E01            0534      addlw  001h      ; Count row
014A 0CAF            0535      rrf    TEMP1     ; Rotate row until it is found
014B 1C03            0536      btfss  STATUS,C
014C 2949            0537      goto   DECL2
014D 00AF            0538      movwf  TEMP1
014E 1003            0539      bcf    STATUS,C
014F 0DAF            0540      rlf    TEMP1     ; Move row to upper nibble
0150 0DAF            0541      rlf    TEMP1
0151 082F            0542      movf   TEMP1,W
0152 072A            0543      addwf  TEMP,W    ; Add column to row value
0153 2156            0544      call  DEC_TABL  ; Get ASCII value of key
0154 00AC            0545      movwf  NEW_KEY
                   0546      DECOUT
0155 0008            0547      return
                   0548      DEC_TABL
0156 00AA            0549      movwf  TEMP      ; Save key value
0157 3001            0550      movlw  01h
0158 008A            0551      movwf  PCLATH    ; Setup for second page of RAM
0159 082A            0552      movf   TEMP,W
015A 0782            0553      addwf  PCL,W     ; Jump, return with ASCII value
015B 3446            0554      retlw  'F'
015C 3442            0555      retlw  'B'
015D 3430            0556      retlw  '0'
015E 3441            0557      retlw  'A'
015F 3445            0558      retlw  'E'
0160 3439            0559      retlw  '9'
0161 3438            0560      retlw  '8'
0162 3437            0561      retlw  '7'
0163 3444            0562      retlw  'D'
0164 3436            0563      retlw  '6'
0165 3435            0564      retlw  '5'
0166 3434            0565      retlw  '4'
0167 3443            0566      retlw  'C'
0168 3433            0567      retlw  '3'
0169 3432            0568      retlw  '2'
016A 3431            0569      retlw  '1'
0570
0571 ;*****
0572 ;*This routine is a software delay. *
0573 ;*At 4Mhz clock, the loop takes 3uS, so initialize TEMP with *
0574 ;*a value of 3 to give 9uS, plus the move etc should result in *
0575 ;*a total time of > 10uS. *
0576 ;*****
0577
0578 SetupDelay
016B 0BAA            0579      decfsz TEMP
016C 296B            0580      goto   SetupDelay
```



MPASM 01.00.02 Alpha SCAPPNOT\IIC 5-10-1994 13:2:14  
Demonstration of I2C MultiMaster mode

PAGE 12

LOC	OBJECT CODE	LINE	SOURCE TEXT
016D	0008	0581	return
		0582	
		0583	END
		0584	

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

```
0000 : X-X----- XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX
0040 : XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX

0080 : XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX
00C0 : XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX

0100 : XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX
0140 : XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX- -----
```

All other memory blocks unused.

Errors : 0  
Warnings : 0

# I<sup>2</sup>C Multi-Master Environment

---

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.