AN578

Use of the SSP Module in the I²C™ Multi-Master Environment

Author: *Scott Fink*
Microchip Technology Inc.

INTRODUCTION

The Inter-IC (I^2C) bus is a two-wire serial interface developed by Philips/Signetics. The specification supports data transmission up to 400 Kbps.

The I^2C 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.

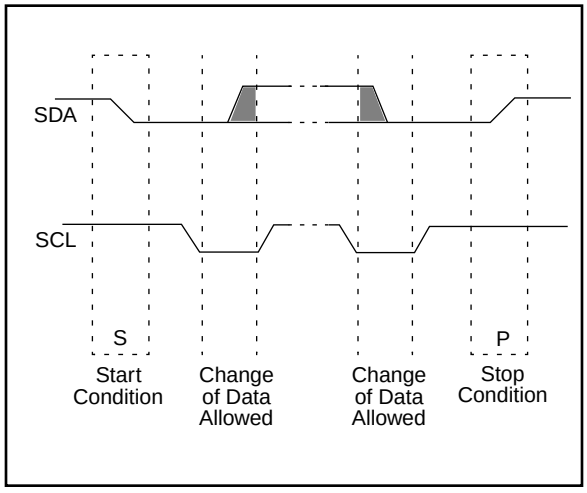
Each device attached to the I^2C bus is assigned a unique address. When a Master wishes to initiate a data transfer, it first transmits the address of the device that it wishes to "talk" to. All devices "listen" to see if this is their address. Within this address, a bit specifies whether the Master 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 the maximum bus loading specification. For complete bus specifications, refer to Philips/Signetics document "The I^2C-bus and How to Use It" (www.semiconductors.philips.com).

INITIATING AND TERMINATING DATA TRANSFER

During times of no data transfer (idle time), both the SCL and SDA lines are pulled high. A Master device which wishes to take control of the bus must first generate a START condition. A START 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. A STOP is defined as a low to high transition of SDA while SCL is high. Because the START and STOP conditions are defined as transitions of the SDA when the SCL line 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.

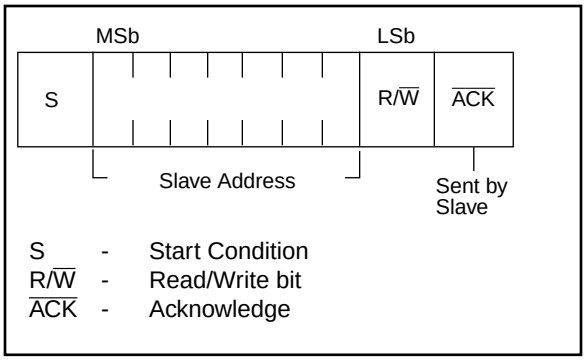
FIGURE 1: START AND STOP CONDITIONS



ADDRESSING I²C DEVICES

There are two address formats. The simplest of these is the 7-bit address format with a R/$\overline{W}$ bit (Figure 2). The more complex is the 10-bit address with a R/$\overline{W}$ bit (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

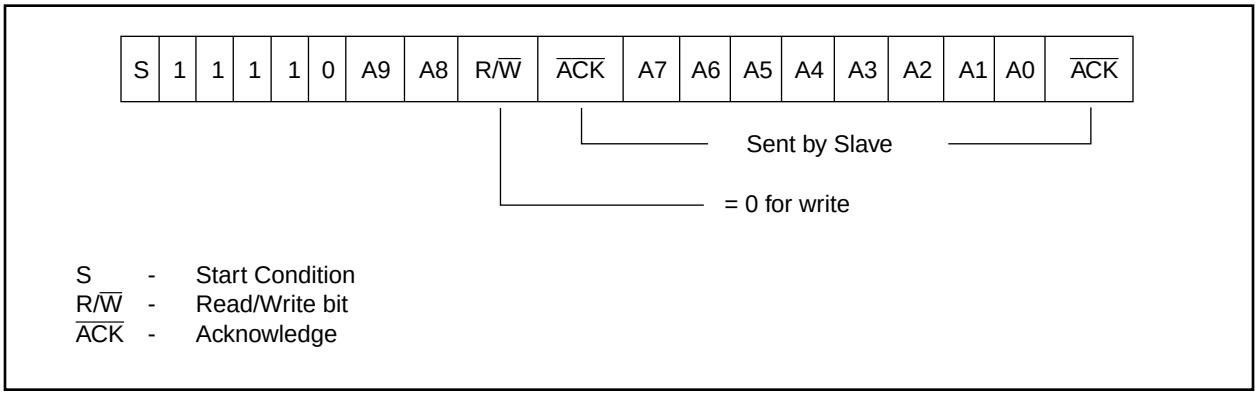
All data is transmitted as bytes, with no limit to the number of bytes transmitted per data transfer. After each byte, the slave-receiver generates an acknowledge bit ($\overline{ACK}$) by pulling the SDA line low. When a slave-receiver doesn't acknowledge the slave address or received data, the master aborts the transfer. Whether the $\overline{ACK}$ bit is generated or not, the SDA line 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^2C 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 (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 (Figure 5).

FIGURE 4: MULTI-MASTER ARBITRATION

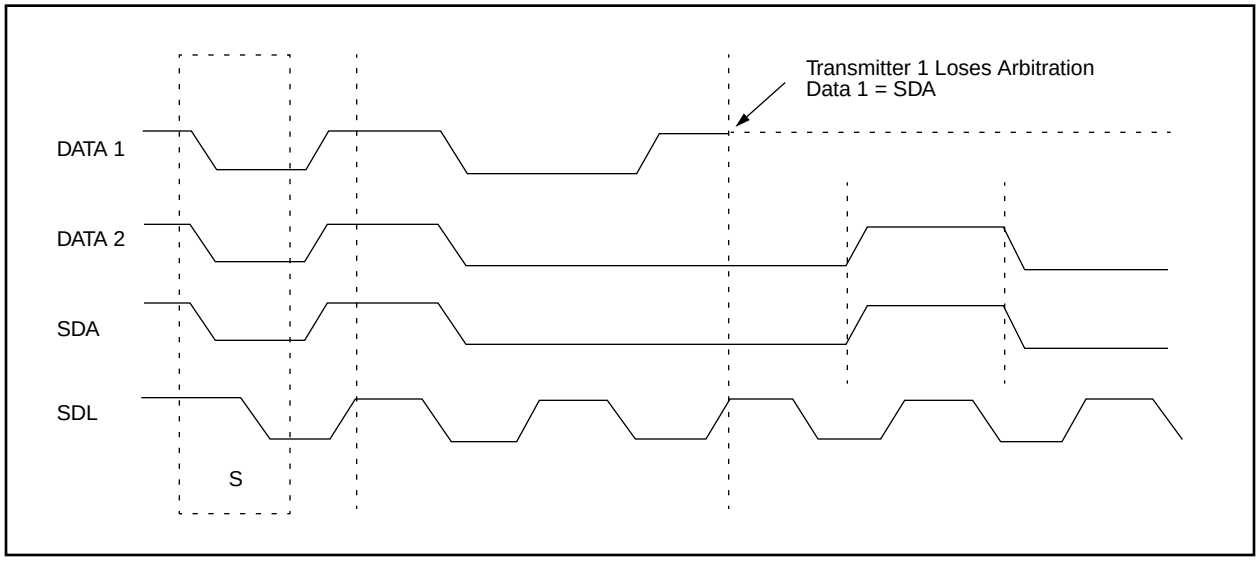
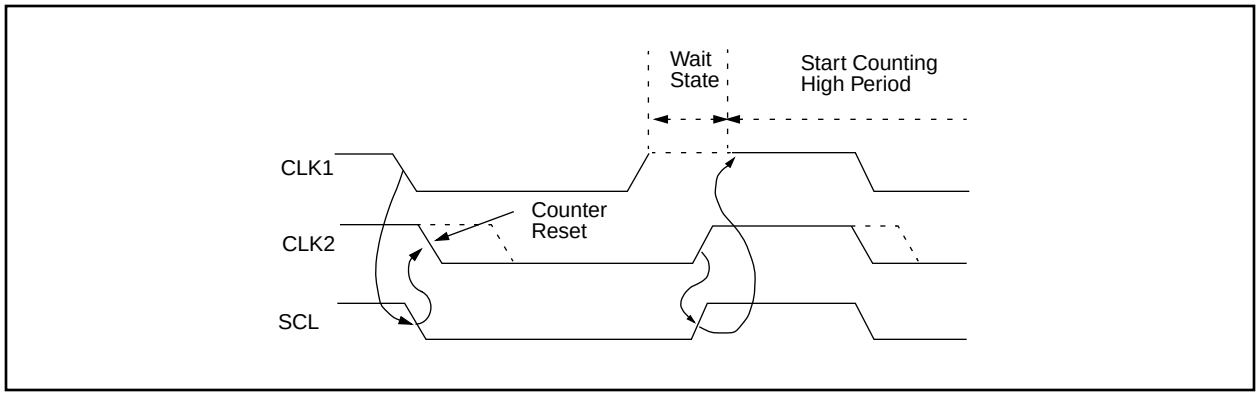


FIGURE 5: CLOCK SYNCHRONIZATION



IMPLEMENTATION IN THE PIC16CXXX

This Application Note uses the PIC16CXXX in a Multi-Master I^2C environment. The PIC16CXXX acts as both a Master and a Slave on the bus.

Hardware

The demonstration hardware consists of a keypad multiplexed with eight LEDs on PORTB and connections to the I^2C bus through the RC3/SCL and RC4/SDA pins (Figure 6).

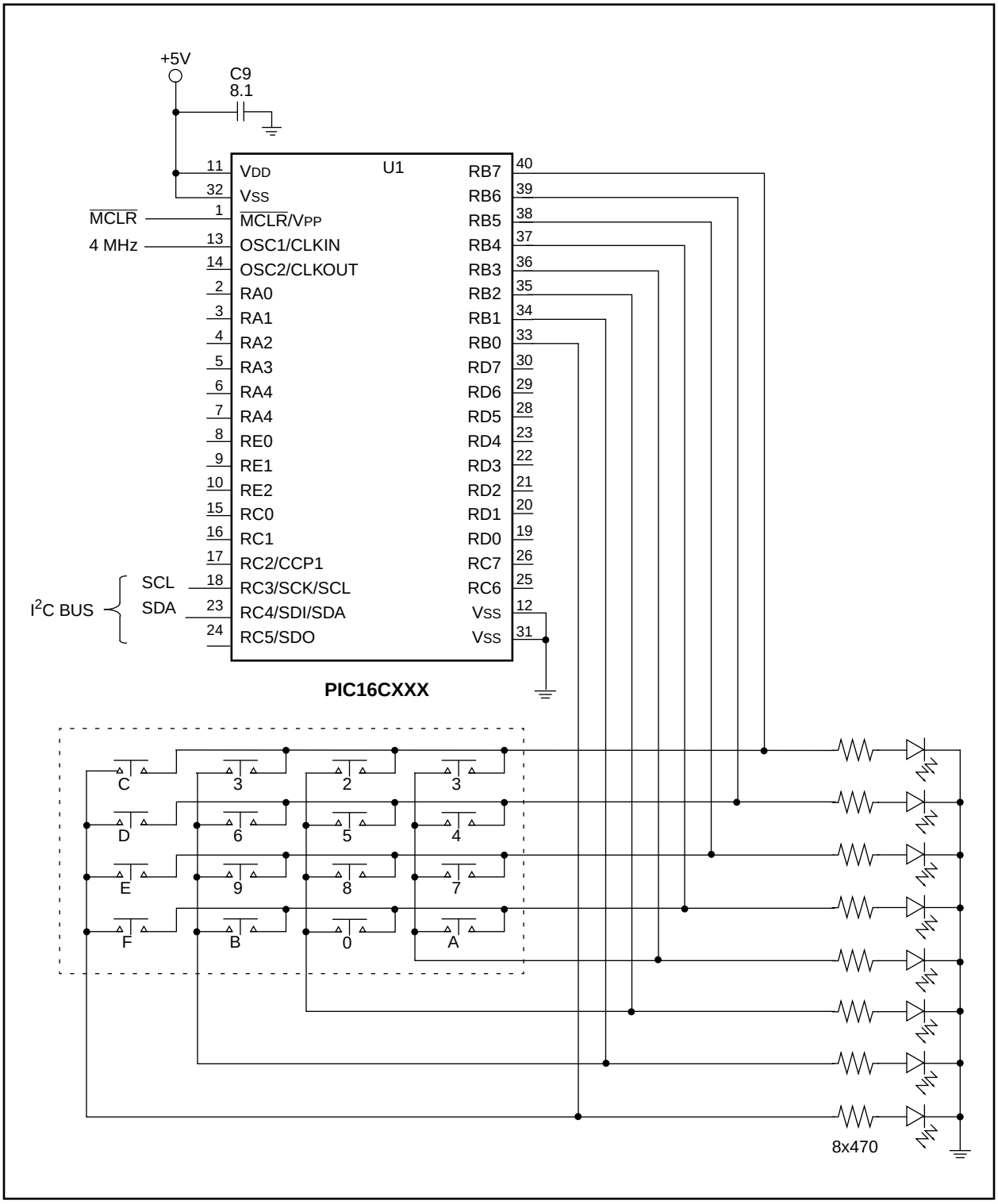
Software

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^2C 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 module, and that the buffer is full, the data word is read and output to the LEDs.

FIGURE 6: HARDWARE



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

APPENDIX A: IICMULT.ASM

```
MPASM 01.40 Released          IICMULT.ASM   1-16-1997  17:01:20         PAGE  1


LOC  OBJECT CODE     LINE SOURCE TEXT
  VALUE

                     00001         TITLE " Demonstration of I2C MultiMaster mode"
                     00002 ;
                     00003                 LIST P=16C64
                     00004                 ERRORLEVEL  -302
                     00005 ;
                     00006 ;*****************************************************************
                     00007 ;**  Two wire/I2C Bus MultiMaster sample routines for Microchip's
                     00008 ;**  PIC16C64 8-bit CMOS single chip microcomputer
                     00009 ;**  Revised Version  (3/06/94).
                     00010 ;       Program:          IICMULT.ASM
                     00011 ;       Revision Date:
                     00012 ;                         1-15-97      Compatibility with MPASMWIN 1.40
                     00013 ;
                     00014 ;**
                     00015 ;**     Part used = PIC16C64
                     00016 ;**Note: 1)All timings are based on a reference crystal frequency of
                     00017 ;**    4MHz which is equivalent to an instruction cycle time of 1 usec.
                     00018 ;**      2) Address and literal values are read in decimal unless
                     00019 ;**        otherwise specified.
                     00020 ;*****************************************************************
                     00021 ;
                     00022 ;-----------------------------------------------------------------
                     00023 ;       File Register Assignment
                     00024 ;-----------------------------------------------------------------
                     00025 ;
                     00026         include <p16C64.inc>
                     00001         LIST
                     00002 ;P16C64.INC Standard Header File, Version 1.01 Microchip Technology
                     00238         LIST
                     00027 ;
  00000020           00028 FLAG            EQU     20h     ; Common flag bits register
  00000021           00029 EEPROM          EQU     21h     ; Bit buffer
  00000022           00030 ERCODE          EQU     22h     ; Error code (to indicate bus status)
  00000023           00031 ADDR            EQU     23h     ; Address register
  00000024           00032 DATAI           EQU     24h     ; Stored data input register
  00000025           00033 DATAO           EQU     25h     ; Stored data output register
  00000026           00034 SLAVE           EQU     26h     ; Device address (1010xxx0)
  00000027           00035 TXBUF           EQU     27h     ; TX buffer
  00000028           00036 RXBUF           EQU     28h     ; RX buffer
  00000029           00037 COUNT           EQU     29h     ; Bit counter
  0000002A           00038 TEMP            EQU     2Ah     ; Temporary storage
  0000002B           00039 ROW             EQU     2Bh     ; Keypad row
  0000002C           00040 NEW_KEY         EQU     2Ch     ; Storage for latest key
  0000002D           00041 OLD_KEY         EQU     2Dh     ; Storage for last key pressed
  0000002E           00042 DISPVAL         EQU     2Eh     ; Value displayed on LEDs
  0000002F           00043 TEMP1           EQU     2Fh     ; Scratchpad register
  00000030           00044 TEMP_W          EQU     30h     ; Storage for W register
  00000031           00045 TEMP_STAT       EQU     31h     ; Storage for STATUS register
                     00046 ;
                     00047 ;-----------------------------------------------------------------
                     00048 ;                     Bit Assignments
                     00049 ;-----------------------------------------------------------------
                     00050
                     00051 ; FLAG Bits
```

```
                    00052
  00000000          00053 ERR_1   EQU     0               ; Error flag
                    00054
                    00055 ; EEPROM Bits
                    00056
  00000007          00057 DI      EQU     7               ; EEPROM input
  00000006          00058 DO      EQU     6               ; EEPROM output
                    00059
                    00060 ; I2C Device Bits
                    00061
  00000004          00062 SDA     EQU     4               ; RB7, data in/out
  00000003          00063 SCL     EQU     3               ; RB6, serial clock
                    00064
                    00065 ;End of files/bits equate
                    00066
0000                00067         ORG     00h             ; Reset Vector
0000 2810           00068         goto    starting
                    00069
0004                00070         ORG     04h             ; Interrupt Vector
0004 28EB           00071         goto    service_int
                    00072
0010                00073         ORG     10h             ; Begining of Program space
0010                00074 starting
0010 01AE           00075         clrf    DISPVAL         ; Blank out LEDs
0011 01A0           00076         clrf    FLAG            ; Clear error register
0012 0187           00077         clrf    PORTC           ; Set SDA, SCL low when not tri-stated
0013 3008           00078         movlw   08h
0014 00AB           00079         movwf   ROW             ; Set initial row to be strobed
0015 303E           00080         movlw   B'00111110'     ; I2C 7 bit slave mode with master
0016 0094           00081         movwf   SSPCON          ; mode enabled
0017 1683           00082         bsf     STATUS,RP0      ; Select Bank1
0018 0186           00083         clrf    TRISB           ; Set PORT_B to all outputs
0019 3008           00084         movlw   B'00001000'     ; Enable SSP interrupt
001A 008C           00085         movwf   PIE1
001B 30A2           00086         movlw   b'10100010'     ; Slave address
001C 0093           00087         movwf   SSPADD
001D 1587           00088         bsf     TRISC, 3        ; Set SCL high
001E 1607           00089         bsf     TRISC, 4        ; Set SDA high
001F 1283           00090         bcf     STATUS,RP0      ; Select Bank0
0020 118C           00091         bcf     PIR1,3          ; Clear SSP interrupt flag
0021 30C0           00092         movlw   B'11000000'     ; Enable interrupts
0022 008B           00093         movwf   INTCON
                    00094
0023                00095 KbdWait
0023 01AA           00096         clrf    TEMP
0024 216B           00097         call    SetupDelay
0025 2102           00098         call    ScanKbd         ; Check for key pressed
0026 1283           00099         bcf     STATUS,RP0
0027 082C           00100         movf    NEW_KEY,W       ; Get latest key
0028 39FF           00101         andlw   0FFh            ; Key pressed?
0029 1903           00102         btfsc   STATUS,Z
002A 2823           00103         goto    KbdWait         ; No, go check again
002B 00A5           00104         movwf   DATAO           ; Yes, output it on I2C bus
002C 30A6           00105         movlw   B'10100110'     ; Address of device being addressed
002D 00A6           00106         movwf   SLAVE
002E 1683           00107         bsf     STATUS,RP0
002F                00108 CheckAgain
002F 1A14           00109         btfsc   SSPSTAT,4       ;If STOP bit received last...
0030 2833           00110         goto    Goxmit          ;  OK to transmit
0031 1994           00111         btfsc   SSPSTAT,3       ;If START bit recieved last...
0032 282F           00112         goto    CheckAgain      ;  wait for STOP bit
0033                00113 Goxmit
0033 1283           00114         bcf     STATUS,RP0
0034 138B           00115         bcf     INTCON,7        ; Disable interrupts
0035 203F           00116         call    WRBYTE          ; Output byte
0036 1C20           00117         btfss   FLAG,ERR_1      ; Check for error
```

```
0037 283D           00118         goto    Checkout        ; No error, go on
0038 0822           00119         movf    ERCODE,W        ; Get error code
0039 0086           00120         movwf   PORTB           ; Put error code on LEDs
003A 00AE           00121         movwf   DISPVAL
003B 1020           00122         bcf     FLAG,ERR_1      ; Clear error flag
003C 282F           00123         goto    CheckAgain
003D                00124 Checkout
003D 178B           00125         bsf     INTCON,7        ; Enable interrupts
003E 2823           00126         goto    KbdWait
                    00127
                    00128 ;--------------------------------------------------------------------
                    00129 ;       BYTE-WRITE, write one byte to I2C (Master Mode)
                    00130 ;--------------------------------------------------------------------
                    00131 ;       Input   :       DATAO   = data to be written
                    00132 ;                       ADDR    = destination address
                    00133 ;                       SLAVE   = device address (1010xxx0)
                    00134 ;       Output  :       Data written to EEPROM device
                    00135 ;--------------------------------------------------------------------
                    00136 ;
003F                00137 WRBYTE
003F 1283           00138         bcf     STATUS,RP0
0040 0826           00139         movf    SLAVE,W         ; Send SLAVE address
0041 00A7           00140         movwf   TXBUF           ;  to TX buffer
0042 20BF           00141         call    BSTART          ; Generate START bit
0043 2071           00142         call    TX              ; Output SLAVE data address
0044 1283           00143         bcf     STATUS,RP0
0045 0825           00144         movf    DATAO,W         ; Move DATA
0046 00A7           00145         movwf   TXBUF           ; into transmit buffer
0047 2071           00146         call    TX              ; Output DATA and detect acknowledgement
0048 20CD           00147         call    BSTOP           ; Generate STOP bit
0049 0008           00148         return
                    00149
                    00150 ;--------------------------------------------------------------------
                    00151 ;       BYTE-READ, read one byte from I2C (Master Mode)
                    00152 ;--------------------------------------------------------------------
                    00153 ;       Input   :       ADDR    = source address
                    00154 ;                       SLAVE   = device address (1010xxx0)
                    00155 ;       Output  :       DATAI   = data read from serial EEPROM
                    00156 ;--------------------------------------------------------------------
                    00157
004A                00158 RDBYTE
004A 1283           00159         bcf     STATUS,RP0
004B 0826           00160         movf    SLAVE,W         ; Move SLAVE address
004C 00A7           00161         movwf   TXBUF           ;  into buffer (R/W = 0)
004D 20BF           00162         call    BSTART          ; Generate START bit
004E 2071           00163         call    TX              ; Output SLAVE address. Check ACK.
004F 1283           00164         bcf     STATUS,RP0
0050 0823           00165         movf    ADDR,W          ; Put slave data address into
0051 00A7           00166         movwf   TXBUF           ;  Xmit buffer
0052 2071           00167         call    TX              ; Output WORD address. Check ACK.
0053 20BF           00168         call    BSTART          ; START READ
0054 1283           00169         bcf     STATUS,RP0
0055 0826           00170         movf    SLAVE,W         ;
0056 00A7           00171         movwf   TXBUF
0057 1427           00172         bsf     TXBUF,0         ; Specify READ mode (R/W = 1)
0058 2071           00173         call    TX              ; Output SLAVE address
0059 205F           00174         call    RX              ; READ in data and acknowledge
005A 20CD           00175         call    BSTOP           ; Generate STOP bit
005B 1283           00176         bcf     STATUS,RP0
005C 0828           00177         movf    RXBUF,W         ; Save data from buffer
005D 00A4           00178         movwf   DATAI           ; to DATAI file.
005E 0008           00179         return
                    00180
                    00181 ;--------------------------------------------------------------------
                    00182 ;       RECEIVE eight data bits subroutine
                    00183 ;--------------------------------------------------------------------
```

```
                  00184 ;       Input   :       None
                  00185 ;       Output  :       RXBUF = 8-bit data received
                  00186 ;-------------------------------------------------------------------
                  00187
005F              00188 RX
005F 1283         00189         bcf     STATUS,RP0
0060 3008         00190         movlw   .8              ; 8 bits of data
0061 00A9         00191         movwf   COUNT
0062 01A8         00192         clrf    RXBUF
                  00193 ;
0063              00194 RXLP
0063 0DA8         00195         rlf     RXBUF, F        ; Shift data to buffer
0064 1C03         00196         btfss   3,0
0065 1028         00197         bcf     RXBUF,0         ; carry ---> f(0)
0066 1803         00198         btfsc   3,0
0067 1428         00199         bsf     RXBUF,0
0068 2087         00200         call    BITIN
0069 1283         00201         bcf     STATUS,RP0
006A 1BA1         00202         btfsc   EEPROM,DI
006B 1428         00203         bsf     RXBUF,0         ; Input bit =1
006C 0BA9         00204         decfsz  COUNT, F        ; 8 bits?
006D 2863         00205         goto    RXLP
006E 1721         00206         bsf     EEPROM,DO       ; Set acknowledge bit = 1
006F 209E         00207         call    BITOUT          ; to STOP further input
0070 3400         00208         retlw   0
                  00209
                  00210 ;-------------------------------------------------------------------
                  00211 ;       TRANSMIT 8 data bits subroutine
                  00212 ;-------------------------------------------------------------------
                  00213 ;       Input   :       TXBUF
                  00214 ;       Output  :       Data X'mitted to EEPROM device
                  00215 ;-------------------------------------------------------------------
                  00216
0071              00217 TX
0071 1283         00218         bcf     STATUS,RP0
0072 3008         00219         movlw   .8
0073 00A9         00220         movwf   COUNT
                  00221 ;
0074              00222 TXLP
0074 1321         00223         bcf     EEPROM,DO       ; Shift data bit out.
0075 1BA7         00224         btfsc   TXBUF,7         ; If shifted bit = 0, data bit = 0
0076 1721         00225         bsf     EEPROM,DO       ; Otherwise data bit = 1
0077 209E         00226         call    BITOUT          ; Serial data out
0078 1283         00227         bcf     STATUS,RP0
0079 0DA7         00228         rlf     TXBUF, F        ; Rotate TXBUF left
007A 1C03         00229         btfss   3,0             ; f(6) ---> f(7)
007B 1027         00230         bcf     TXBUF,0         ; f(7) ---> carry
007C 1803         00231         btfsc   3,0             ; carry ---> f(0)
007D 1427         00232         bsf     TXBUF,0
007E 0BA9         00233         decfsz  COUNT, F        ; 8 bits done?
007F 2874         00234         goto    TXLP            ; No.
0080 2087         00235         call    BITIN           ; Read acknowledge bit
0081 1283         00236         bcf     STATUS,RP0
0082 3003         00237         movlw   3
0083 1BA1         00238         btfsc   EEPROM,DI       ; Check for acknowledgement
0084 20DE         00239         call    ERR             ; No acknowledge from device
0085 1283         00240         bcf     STATUS,RP0
0086 3400         00241         retlw   0
                  00242
                  00243 ;-------------------------------------------------------------------
                  00244 ;       Single bit receive from I2C to PIC
                  00245 ;-------------------------------------------------------------------
                  00246 ;       Input   :       None
                  00247 ;       Output  :       Data bit received
                  00248 ;-------------------------------------------------------------------
                  00249
```

```
0087                00250 BITIN
0087 1683           00251         bsf     STATUS,RP0
0088 1607           00252         bsf     TRISC,SDA       ; Set SDA for input
0089 1283           00253         bcf     STATUS,RP0
008A 13A1           00254         bcf     EEPROM,DI
008B 1683           00255         bsf     STATUS,RP0
008C 1587           00256         bsf     TRISC,SCL       ; Clock high
008D 3001           00257         movlw   1
008E 1283           00258         bcf     STATUS,RP0
008F 1987           00259         btfsc   PORTC,SCL       ; Skip if SCL  is high
0090 2895           00260         goto    BIT1
0091 1283           00261         bcf     STATUS,RP0
0092 1C20           00262         btfss   FLAG,ERR_1      ; Remain as first error encountered
0093 00A2           00263         movwf   ERCODE          ; Save error code
0094 1420           00264         bsf     FLAG,ERR_1      ; Set error flag
0095                00265 BIT1
0095 1283           00266         bcf     STATUS,RP0
0096 1E07           00267         btfss   PORTC,SDA       ; Read SDA pin, for ACK low
0097 289A           00268         goto    ACKOK
0098 1283           00269         bcf     STATUS,RP0
0099 17A1           00270         bsf     EEPROM,DI       ; DI = 1
009A                00271 ACKOK
009A 1683           00272         bsf     STATUS,RP0
009B 0000           00273         nop                     ; Delay
009C 1187           00274         bcf     TRISC,SCL       ; Return SCL to low
009D 3400           00275         retlw   0
                    00276
                    00277 ;-------------------------------------------------------------------
                    00278 ;       Single bit data transmit from PIC to I2C
                    00279 ;-------------------------------------------------------------------
                    00280 ;       Input   :       EEPROM register, bit DO
                    00281 ;       Output  :       Bit transmitted over I2C
                    00282 ;                       Error bits set as necessary
                    00283 ;-------------------------------------------------------------------
                    00284
009E                00285 BITOUT
009E 1283           00286         bcf     STATUS,RP0
009F 1F21           00287         btfss   EEPROM,DO
00A0 28AB           00288         goto    BIT0
00A1 1683           00289         bsf     STATUS,RP0
00A2 1607           00290         bsf     TRISC,SDA       ; Output bit 0
00A3 3002           00291         movlw   2
00A4 1283           00292         bcf     STATUS,RP0
00A5 1A07           00293         btfsc   PORTC,SDA       ; Check for error code 2
00A6 28B0           00294         goto    CLK1
00A7 1C20           00295         btfss   FLAG,ERR_1      ; Remain as first error encountered
00A8 00A2           00296         movwf   ERCODE          ; Save error code
00A9 1420           00297         bsf     FLAG,ERR_1      ; Set error flag
00AA 28B0           00298         goto    CLK1            ; SDA locked low by device
                    00299 ;
00AB                00300 BIT0
00AB 1683           00301         bsf     STATUS,RP0
00AC 1207           00302         bcf     TRISC,SDA       ; Output bit 0
00AD 0000           00303         nop                      ; Delay
00AE 0000           00304         nop
00AF 0000           00305         nop
00B0                00306 CLK1
00B0 1683           00307         bsf     STATUS,RP0
00B1 1587           00308         bsf     TRISC,SCL
00B2 3001           00309         movlw   1               ; Error code 1
00B3 1283           00310         bcf     STATUS,RP0
00B4 1987           00311         btfsc   PORTC,SCL       ; SCL locked low?
00B5 28BA           00312         goto    BIT2            ; No.
00B6 1283           00313         bcf     STATUS,RP0
00B7 1C20           00314         btfss   FLAG,ERR_1      ; Yes.
00B8 00A2           00315         movwf   ERCODE          ; Save error code
```

```
00B9 1420          00316         bsf     FLAG,ERR_1      ; Set error flag
00BA               00317 BIT2
00BA 0000          00318         nop
00BB 0000          00319         nop
00BC 1683          00320         bsf     STATUS,RP0
00BD 1187          00321         bcf     TRISC,SCL       ; Return SCL to low
00BE 3400          00322         retlw   0
                   00323
                   00324 ;-------------------------------------------------------------------
                   00325 ;       START bit generation routine
                   00326 ;-------------------------------------------------------------------
                   00327 ;       input   : none
                   00328 ;       output  : initialize bus communication
                   00329 ;-------------------------------------------------------------------
                   00330
                   00331 ;Generate START bit (SCL is high while SDA goes from high to low
                   00332 ;transition) and check status of the serial clock.
00BF               00333 BSTART
00BF 1683          00334         bsf     STATUS,RP0
00C0 1607          00335         bsf     TRISC,SDA       ; Make sure SDA is high
00C1 1587          00336         bsf     TRISC,SCL       ; Set clock high
00C2 3001          00337         movlw   1               ; Ready error status code 1
00C3 1283          00338         bcf     STATUS,RP0
00C4 1D87          00339         btfss   PORTC,SCL       ; Locked?
00C5 20DE          00340         call    ERR             ; SCL locked low by device, flag error
00C6 1683          00341         bsf     STATUS,RP0
00C7 1207          00342         bcf     TRISC,SDA       ; SDA goes low during SCL high
00C8 0000          00343         nop                     ; Timing adjustment, 1.5uS @2MHz
00C9 0000          00344         nop
00CA 0000          00345         nop
00CB 1187          00346         bcf     TRISC,SCL       ; Start clock train
00CC 3400          00347         retlw   0
                   00348
                   00349 ;-------------------------------------------------------------------
                   00350 ;       STOP bit generation routine
                   00351 ;-------------------------------------------------------------------
                   00352 ;       Input   :       None
                   00353 ;       Output  :       Bus communication, STOP condition
                   00354 ;-------------------------------------------------------------------
                   00355
                   00356 ;Generate STOP bit (SDA goes from low to high during SCL high state)
                   00357 ;and check bus conditions.
                   00358
00CD               00359 BSTOP
00CD 1683          00360         bsf     STATUS,RP0
00CE 1207          00361         bcf     TRISC,SDA     ; Return SDA to low
00CF 1587          00362         bsf     TRISC,SCL     ; Set SCL high
00D0 0000          00363         nop
00D1 0000          00364         nop
00D2 0000          00365         nop
00D3 3001          00366         movlw   1             ; Ready error code 1
00D4 1283          00367         bcf     STATUS,RP0
00D5 1D87          00368         btfss   PORTC,SCL     ; High?
00D6 20DE          00369         call    ERR           ; No, SCL locked low by device
00D7 1683          00370         bsf     STATUS,RP0
00D8 1607          00371         bsf     TRISC,SDA     ; SDA goes low to high during SCL high
00D9 3004          00372         movlw   4             ; Ready error code 4
00DA 1E07          00373         btfss   TRISC,SDA     ; High?
00DB 20DE          00374         call    ERR           ; No, SDA bus not release for STOP
00DC 1283          00375         bcf     STATUS,RP0
00DD 3400          00376         retlw   0
                   00377
                   00378 ;-------------------------------------------------------------------
                   00379 ;    Two wire/I2C - CPU communication error status table and subroutine
                   00380 ;-------------------------------------------------------------------
                   00381 ;  input  :     W-reg   = error code
```

```
                  00382 ;  output :     ERCODE  = error code
                  00383 ;           FLAG(ERR_1) = 1
                  00384 ;
                  00385 ;         code                  error status mode
                  00386 ;        ------      ------------------------------------------------
                  00387 ;         1   :    SCL locked low by device (bus is still busy)
                  00388 ;         2   :    SDA locked low by device (bus is still busy)
                  00389 ;         3   :    No acknowledge from device (no handshake)
                  00390 ;         4   :    SDA bus not released for master to generate STOP bit
                  00391 ;--------------------------------------------------------------------
                  00392 ;
                  00393 ;Subroutine to identify the status of the serial clock (SCL) and serial
                  00394 ;(data SDA) condition according to the error status table.Codes
                  00395 ;generated are useful for bus/device diagnosis.
                  00396
00DE              00397 ERR
00DE 1283         00398         bcf     STATUS,RP0
00DF 1C20         00399         btfss   FLAG,ERR_1      ; Keep first error reported
00E0 00A2         00400         movwf   ERCODE          ; Save error code
00E1 1420         00401         bsf     FLAG,ERR_1      ; Set error flag
00E2 3400         00402         retlw   0
                  00403
                  00404 ;--------------------------------------------------------------------
                  00405 ;       DELAY, Provide a 1.54mS delay
                  00406 ;--------------------------------------------------------------------
                  00407 ;       Input   :       None
                  00408 ;       Output  :       None
                  00409 ;--------------------------------------------------------------------
                  00410
00E3              00411 delay
00E3 1283         00412         bcf     STATUS,RP0
00E4 01AA         00413         clrf    TEMP            ;clear last location
00E5              00414 dly1
00E5 0000         00415         nop
00E6 0000         00416         nop
00E7 0000         00417         nop
00E8 0BAA         00418         decfsz  TEMP, F         ;reduce count
00E9 28E5         00419         goto    dly1            ;loop
00EA 3400         00420         retlw   0
                  00421
                  00422 ;--------------------------------------------------------------------
                  00423 ; Interrupt service routine
                  00424 ;  Only the SSP interrupt is enabled.  This routine will read the I2C
                  00425 ;  data and output it on the LEDs.
                  00426 ;--------------------------------------------------------------------
                  00427
00EB              00428 service_int
00EB 118C         00429         bcf     PIR1,3          ; Clear SSP interrupt
00EC 00B0         00430         movwf   TEMP_W          ; Save W register
00ED 0E03         00431         swapf   STATUS,W        ; Get STATUS register
00EE 00B1         00432         movwf   TEMP_STAT       ; Save STATUS register
00EF 1683         00433         bsf     STATUS,RP0
00F0 1C14         00434         btfss   SSPSTAT,0       ; Check Buffer Full Flag
00F1 28FA         00435         goto    IntOut          ; No data received, so exit
00F2 1283         00436         bcf     STATUS,RP0
00F3 0813         00437         movf    SSPBUF,W        ; Get I2C data
00F4 1683         00438         bsf     STATUS,RP0
00F5 1E94         00439         btfss   SSPSTAT,5       ; If Address received last...
00F6 28FA         00440         goto    IntOut          ;  exit without saving it
00F7 1283         00441         bcf     STATUS,RP0
00F8 0086         00442         movwf   PORTB           ; Display received data on LEDs
00F9 00AE         00443         movwf   DISPVAL
00FA              00444 IntOut
00FA 1683         00445         bsf     STATUS,RP0
00FB 158C         00446         bsf     PIE1,3          ; Re-enable SSP interrupt
00FC 1283         00447         bcf     STATUS,RP0
```

```
00FD 0E31           00448         swapf   TEMP_STAT,W     ; Restore STATUS register
00FE 0083           00449         movwf   STATUS
00FF 0EB0           00450         swapf   TEMP_W,1
0100 0E30           00451         swapf   TEMP_W,W        ; Restore W register
0101 0009           00452         retfie
                    00453
                    00454 ;-----------------------------------------------------------------
                    00455 ; Keyboard scan routine
                    00456 ;  This routine scans the keypad connected to PORT_B, and
                    00457 ;  returns the pressed key in New_Key.
                    00458 ;-----------------------------------------------------------------
                    00459
0102                00460 ScanKbd
0102 1283           00461         bcf     STATUS,RP0
0103 01AC           00462         clrf    NEW_KEY         ; Clear key register
0104 082D           00463         movf    OLD_KEY,w       ; If key was pressed last pass through
0105 1D03           00464         btfss   STATUS,Z        ;  goto Debounce
0106 291C           00465         goto    Debounce
0107                00466 Kbdloop
0107 1003           00467         bcf     STATUS,C
0108 0DAB           00468         rlf     ROW, F          ; Select next row to strobe
0109 1C03           00469         btfss   STATUS,C
010A 290D           00470         goto    Notdone
010B 3010           00471         movlw   010h            ; Start over at first row
010C 00AB           00472         movwf   ROW
010D                00473 Notdone
010D 0186           00474         clrf    PORTB
010E 1683           00475         bsf     STATUS,RP0
010F 300F           00476         movlw   00Fh            ; Set PORT_B for keypad read
0110 0086           00477         movwf   TRISB
0111 1283           00478         bcf     STATUS,RP0
0112 082B           00479         movf    ROW,W           ; Output Row
0113 0086           00480         movwf   PORTB           ; Read colums
0114 300F           00481         movlw   0Fh             ; Mask out rows
0115 0506           00482         andwf   PORTB,W         ; Check for Key press
0116 1903           00483         btfsc   STATUS,Z
0117 2933           00484         goto    KBDOUT          ; No key pressed, exit
0118 0186           00485         clrf    PORTB
0119 042B           00486         iorwf   ROW,W           ; Key pressed, save it
011A 00AC           00487         movwf   NEW_KEY
011B 00AD           00488         movwf   OLD_KEY
011C                00489 Debounce
011C 0186           00490         clrf    PORTB
011D 1683           00491         bsf     STATUS,RP0
011E 0186           00492         clrf    TRISB
011F 1283           00493         bcf     STATUS,RP0
0120 082E           00494         movf    DISPVAL,W       ; Set LEDs
0121 0086           00495         movwf   PORTB
0122 01AA           00496         clrf    TEMP
0123 216B           00497         call    SetupDelay
0124 216B           00498         call    SetupDelay      ; Delay for key debounce
0125 0186           00499         clrf    PORTB
0126 1683           00500         bsf     STATUS,RP0
0127 300F           00501         movlw   0Fh             ; Set PORT_B for keypad read
0128 0086           00502         movwf   TRISB
0129 1283           00503         bcf     STATUS,RP0
012A 082B           00504         movf    ROW,W
012B 0086           00505         movwf   PORTB           ; Output Row
012C 082D           00506         movf    OLD_KEY,W
012D 0606           00507         xorwf   PORTB,w         ; Compare key with last key pressed
012E 1903           00508         btfsc   STATUS,Z
012F 2933           00509         goto    KBDOUT
0130 0186           00510         clrf    PORTB           ; Key released, clear registers
0131 01AC           00511         clrf    NEW_KEY
0132 01AD           00512         clrf    OLD_KEY
0133                00513 KBDOUT
```

```
0133 082E           00514         movf    DISPVAL,W       ; Set LEDs
0134 0086           00515         movwf   PORTB
0135 1683           00516         bsf     STATUS,RP0
0136 0186           00517         clrf    TRISB
0137 1283           00518         bcf     STATUS,RP0
0138                00519 KEY_DEC
0138 300F           00520         movlw   00Fh
0139 052C           00521         andwf   NEW_KEY,W       ; Get column of key pressed
013A 00AA           00522         movwf   TEMP
013B 1903           00523         btfsc   STATUS,Z        ; If no key pressed, exit
013C 2955           00524         goto    DECOUT
013D 30FF           00525         movlw   0FFh            ; Initialize the W register
013E                00526 DECL1
013E 3E01           00527         addlw   001h            ; Count column
013F 0CAA           00528         rrf     TEMP, F         ; Rotate column until it is found
0140 1C03           00529         btfss   STATUS,C
0141 293E           00530         goto    DECL1
0142 00AA           00531         movwf   TEMP
0143 0E2C           00532         swapf   NEW_KEY,W       ; Get row of key pressed
0144 390F           00533         andlw   00Fh
0145 1903           00534         btfsc   STATUS,Z
0146 2955           00535         goto    DECOUT          ;If no key pressed, exit
0147 00AF           00536         movwf   TEMP1
0148 30FF           00537         movlw   0FFH
0149                00538 DECL2
0149 3E01           00539         addlw   001h            ; Count row
014A 0CAF           00540         rrf     TEMP1, F        ; Rotate row until it is found
014B 1C03           00541         btfss   STATUS,C
014C 2949           00542         goto    DECL2
014D 00AF           00543         movwf   TEMP1
014E 1003           00544         bcf     STATUS,C
014F 0DAF           00545         rlf     TEMP1, F        ; Move row to upper nibble
0150 0DAF           00546         rlf     TEMP1, F
0151 082F           00547         movf    TEMP1,W
0152 072A           00548         addwf   TEMP,W          ; Add column to row value
0153 2156           00549         call    DEC_TABL        ; Get ASCII value of key
0154 00AC           00550         movwf   NEW_KEY
0155                00551 DECOUT
0155 0008           00552         return
0156                00553 DEC_TABL
0156 00AA           00554         movwf   TEMP            ; Save key value
0157 3001           00555         movlw   01h
0158 008A           00556         movwf   PCLATH          ; Setup for second page of RAM
0159 082A           00557         movf    TEMP,W
015A 0782           00558         addwf   PCL, F          ; Jump, return with ASCII value
015B 3446           00559         retlw   'F'
015C 3442           00560         retlw   'B'
015D 3430           00561         retlw   '0'
015E 3441           00562         retlw   'A'
015F 3445           00563         retlw   'E'
0160 3439           00564         retlw   '9'
0161 3438           00565         retlw   '8'
0162 3437           00566         retlw   '7'
0163 3444           00567         retlw   'D'
0164 3436           00568         retlw   '6'
0165 3435           00569         retlw   '5'
0166 3434           00570         retlw   '4'
0167 3443           00571         retlw   'C'
0168 3433           00572         retlw   '3'
0169 3432           00573         retlw   '2'
016A 3431           00574         retlw   '1'
                    00575
                    00576 ;*****************************************************************
                    00577 ;*This routine is a software delay.                               *
                    00578 ;*At 4Mhz clock, the loop takes 3uS, so initialize TEMP with     *
                    00579 ;*a value of 3 to give 9uS, plus the move etc should result in   *
```

```
                  00580 ;*a total time of > 10uS.                                 *
                  00581 ;*********************************************************************
                  00582
016B              00583 SetupDelay
016B 0BAA         00584         decfsz  TEMP, F
016C 296B         00585         goto    SetupDelay
016D 0008         00586         return
                  00587
                  00588         END


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

0000 : X---X----------- XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0040 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0080 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
00C0 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0100 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0140 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXX-- ----------------

All other memory blocks unused.

Program Memory Words Used:   352
Program Memory Words Free:  1696


Errors   :     0
Warnings :     0 reported,     0 suppressed
Messages :     0 reported,    29 suppressed
```

Note the following details of the code protection feature on PICmicro® MCUs.

- The PICmicro family meets the specifications contained in the Microchip Data Sheet.
- Microchip believes that its family of PICmicro microcontrollers is one of the most secure products of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the PICmicro microcontroller in a manner outside the operating specifications contained in the data sheet. The person doing so may be engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable”.
- Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our product.

If you have any further questions about this matter, please contact the local sales office nearest to you.

DNV Certification, Inc. USA — ANSI-RAB QMS ACCREDITED — DNV — DNV MSC The Netherlands Accredited by the RvA — ISO 9001 / QS-9000 REGISTERED FIRM

Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999. The Company’s quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELOQ® code hopping devices, Serial EEPROMs and microperipheral products. In addition, Microchip’s quality system for the design and manufacture of development systems is ISO 9001 certified.

WORLDWIDE SALES AND SERVICE

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200 Fax: 480-792-7277
Technical Support: 480-792-7627
Web Address: http://www.microchip.com

Rocky Mountain
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7966 Fax: 480-792-7456

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

Boston
2 Lan Drive, Suite 120
Westford, MA 01886
Tel: 978-692-3848 Fax: 978-692-3821

Chicago
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 630-285-0071 Fax: 630-285-0075

Dallas
4570 Westgrove Drive, Suite 160
Addison, TX 75001
Tel: 972-818-7423 Fax: 972-818-2924

Detroit
Tri-Atria Office Building
32255 Northwestern Highway, Suite 190
Farmington Hills, MI 48334
Tel: 248-538-2250 Fax: 248-538-2260

Kokomo
2767 S. Albright Road
Kokomo, Indiana 46902
Tel: 765-864-8360 Fax: 765-864-8387

Los Angeles
18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 949-263-1888 Fax: 949-263-1338

New York
150 Motor Parkway, Suite 202
Hauppauge, NY 11788
Tel: 631-273-5305 Fax: 631-273-5335

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

Toronto
6285 Northam Drive, Suite 108
Mississauga, Ontario L4V 1X5, Canada
Tel: 905-673-0699 Fax: 905-673-6509

ASIA/PACIFIC

Australia
Microchip Technology Australia Pty Ltd
Suite 22, 41 Rawson Street
Epping 2121, NSW
Australia
Tel: 61-2-9868-6733 Fax: 61-2-9868-6755

China - Beijing
Microchip Technology Consulting (Shanghai) Co., Ltd., Beijing Liaison Office
Unit 915
Bei Hai Wan Tai Bldg.
No. 6 Chaoyangmen Beidajie
Beijing, 100027, No. China
Tel: 86-10-85282100 Fax: 86-10-85282104

China - Chengdu
Microchip Technology Consulting (Shanghai) Co., Ltd., Chengdu Liaison Office
Rm. 2401, 24th Floor,
Ming Xing Financial Tower
No. 88 TIDU Street
Chengdu 610016, China
Tel: 86-28-6766200 Fax: 86-28-6766599

China - Fuzhou
Microchip Technology Consulting (Shanghai) Co., Ltd., Fuzhou Liaison Office
Unit 28F, World Trade Plaza
No. 71 Wusi Road
Fuzhou 350001, China
Tel: 86-591-7503506 Fax: 86-591-7503521

China - Shanghai
Microchip Technology Consulting (Shanghai) Co., Ltd.
Room 701, Bldg. B
Far East International Plaza
No. 317 Xian Xia Road
Shanghai, 200051
Tel: 86-21-6275-5700 Fax: 86-21-6275-5060

China - Shenzhen
Microchip Technology Consulting (Shanghai) Co., Ltd., Shenzhen Liaison Office
Rm. 1315, 13/F, Shenzhen Kerry Centre,
Renminnan Lu
Shenzhen 518001, China
Tel: 86-755-2350361 Fax: 86-755-2366086

Hong Kong
Microchip Technology Hongkong Ltd.
Unit 901-6, Tower 2, Metroplaza
223 Hing Fong Road
Kwai Fong, N.T., Hong Kong
Tel: 852-2401-1200 Fax: 852-2401-3431

India
Microchip Technology Inc.
India Liaison Office
Divyasree Chambers
1 Floor, Wing A (A3/A4)
No. 11, O'Shaugnessey Road
Bangalore, 560 025, India
Tel: 91-80-2290061 Fax: 91-80-2290062

Japan
Microchip Technology Japan K.K.
Benex S-1 6F
3-18-20, Shinyokohama
Kohoku-Ku, Yokohama-shi
Kanagawa, 222-0033, Japan
Tel: 81-45-471- 6166 Fax: 81-45-471-6122

Korea
Microchip Technology Korea
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku
Seoul, Korea 135-882
Tel: 82-2-554-7200 Fax: 82-2-558-5934

Singapore
Microchip Technology Singapore Pte Ltd.
200 Middle Road
#07-02 Prime Centre
Singapore, 188980
Tel: 65-334-8870 Fax: 65-334-8850

Taiwan
Microchip Technology Taiwan
11F-3, No. 207
Tung Hua North Road
Taipei, 105, Taiwan
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

EUROPE

Denmark
Microchip Technology Nordic ApS
Regus Business Centre
Lautrup hoj 1-3
Ballerup DK-2750 Denmark
Tel: 45 4420 9895 Fax: 45 4420 9910

France
Microchip Technology SARL
Parc d'Activite du Moulin de Massy
43 Rue du Saule Trapu
Batiment A - ler Etage
91300 Massy, France
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79

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

Italy
Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Taurus 1 V. Le Colleoni 1
20041 Agrate Brianza
Milan, Italy
Tel: 39-039-65791-1 Fax: 39-039-6899883

United Kingdom
Arizona Microchip Technology Ltd.
505 Eskdale Road
Winnersh Triangle
Wokingham
Berkshire, England RG41 5TU
Tel: 44 118 921 5869 Fax: 44-118 921-5820

01/18/02