

Interfacing 93CX6 Serial EEPROMs to PIC16C5X Microcontrollers

Authors: Stan D'Souza
 Microchip Technology Inc.
 Bob Ward
 Microchip Technology Inc.

INTRODUCTION

Microchip Technology Inc.'s popular 93C46/56/66 and 93LC46/56/66 Serial EEPROMs feature a three/four wire serial interface bus. The attractive price and simple interface make it the ideal device for additional memory space. This application note is intended for design engineers who wish to incorporate a pre-packaged serial EEPROM interface driver into their application.

THE HARDWARE CONNECTION

A typical 4-wire hardware connection is illustrated in Figure 1 and a typical 3-wire connection is illustrated in Figure 2. Since all I/O ports on the PIC16C5X are configurable as input and/or output, a 3-wire interface makes optimum utilization of the I/O pins by having a common connection for the DI and DO lines of the Serial EEPROM. The port pin on the PIC16C5X connected to these pins, has a default setting as an output and is configured, when needed, as an input during program execution.

THE SOFTWARE CONNECTION

An example interface driver is listed in Appendix A. A flow diagram is given in Figure 2. The interface driver is written to minimize both overhead to the calling program as well as the program space necessary for its inclusion into the user's code. The driver has been written as a generic driver to service all 93 Series Serial EEPROMs made by Microchip. Processor resources which must be made available to the driver prior to being called are:

1. Two levels of processor stack.
2. Six register locations (four for command/data passing and two for software counters).
3. The File Select Register (FSR), which is used to pass a command/data string pointer to the driver.

Note: The four command/data passing registers have to be defined consecutively in order for the FSR to access them successfully in the program execution.

FIGURE 1: 4-WIRE CONNECTION

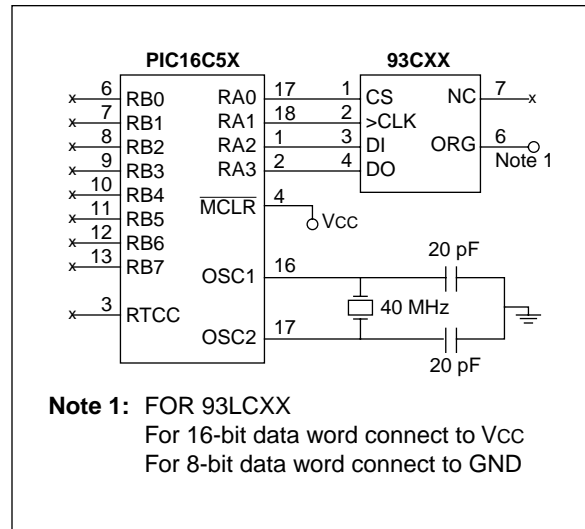
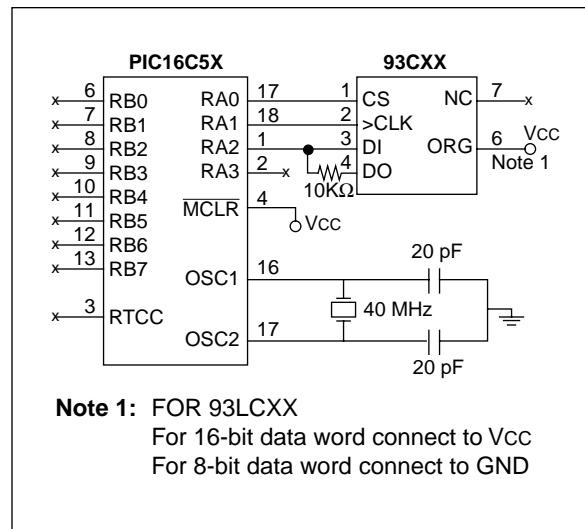


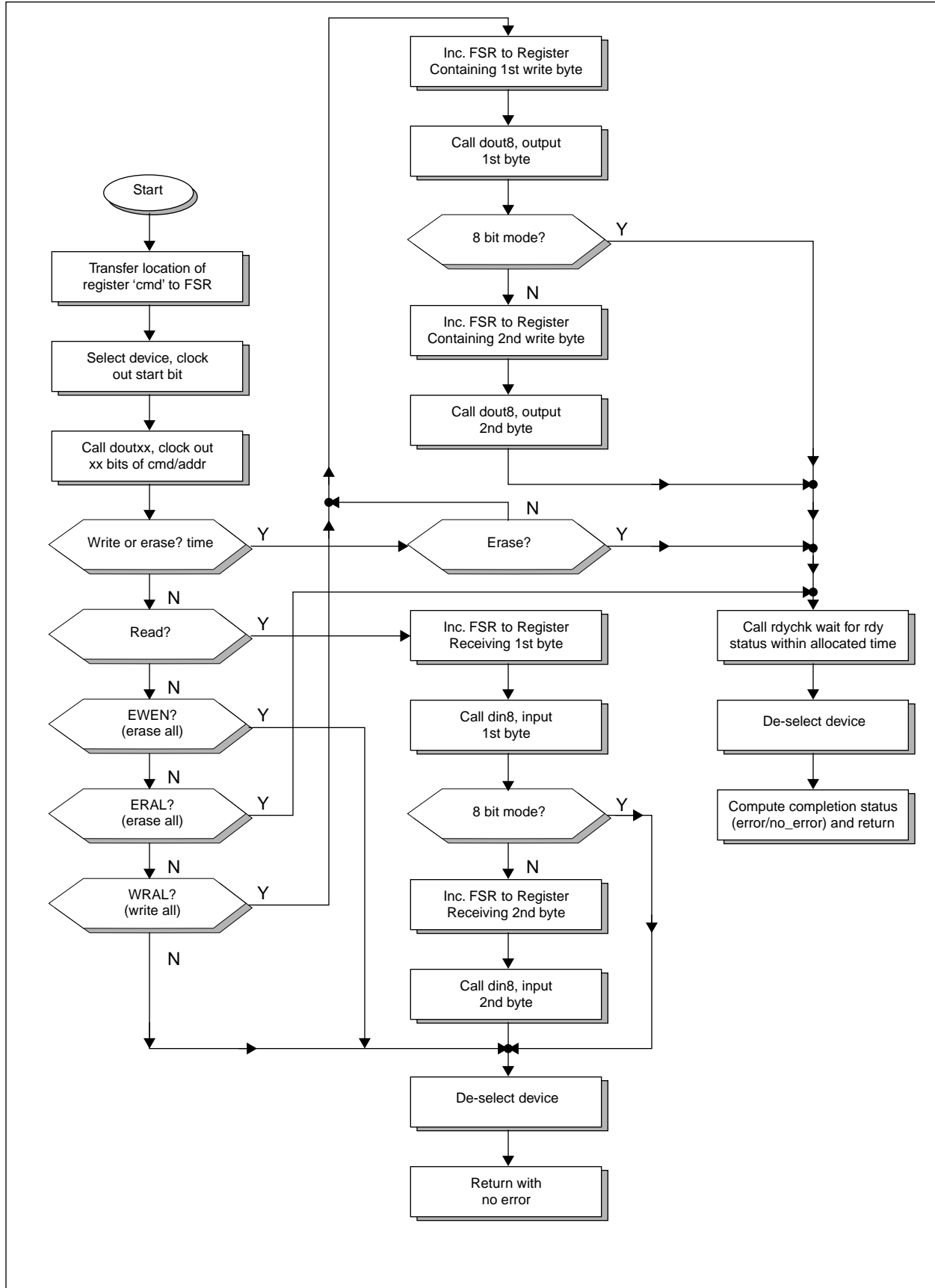
FIGURE 2: 3-WIRE CONNECTION



The user should take the following steps when using the routines provided in Appendix A.

- A) Specify and define a 3-/4-wire interface by defining the common connection to the DI/DO lines and setting the equate 'wire3' TRUE or FALSE (4-wire is automatically assumed if 3-wire is false).
- B) Specify and define if 16-bit or 8-bit data organization is used, by setting equate 'org8' TRUE or FALSE.

FIGURE 3: FLOW CHART



- C) The user should assemble the source file by specifying which type of serial EEPROM is being used. This is done by defining the equate S93C46, S93LC46, etc., as TRUE. Only one device can be TRUE, the rest have to be defined FALSE.
- D) The user would invoke the driver as follows:
1. Load the register defined as 'cmd' with the 93CX6 Command Opcode (only the four upper bits are used in this register; see Figure 3).
 2. If necessary, load the register defined 'addr' with the lower 8/7/6 bit address of the location.
 3. If necessary, load the 9th bit of the address as bit 3 of the register defined as 'cmd' (Figure 3).

Note: READ, WRITE and ERASE commands need to have an address associated with the command and the 9th bit of the address is only required when using the 93C56/66 or 93LC56/66 devices in the 8 bit mode (ORG tied to GND).

4. If necessary, load the register defined as 'highb' and 'lowb' with the 16 bits of data, the most significant byte loaded in 'highb'. In 8 bit mode, 'highb' should be loaded with the 8 bit data.

Note: Only the WRITE and WRAL commands require data to be loaded in the 'highb', 'lowb' locations.

5. Call the driver sub-routine 'see'.
6. Upon completion, the driver will return a completion status in the W register

(error/no error). Only commands requiring a status check are capable of returning a valid error/no error status, in all other cases a no error is returned.

7. If the READ command is executed, the 16/8 bit data will be loaded in the 'highb' and 'lowb' registers, where 'highb' contains the MSB in the 16 bit mode and 8 bit data in the 8-bit mode. The Example interface assumes a 4 MHz oscillator clock which gives us a 1 μ S instruction cycle. If a higher clock speed is used, additional NOPs have to be included in the code in order to meet the minimum clock speed requirements of the 93 Series Serial EEPROMs (see data sheet for further details).

Listing in Appendix B is for an interface to 93C46 Serial EEPROM only.

SUMMARY

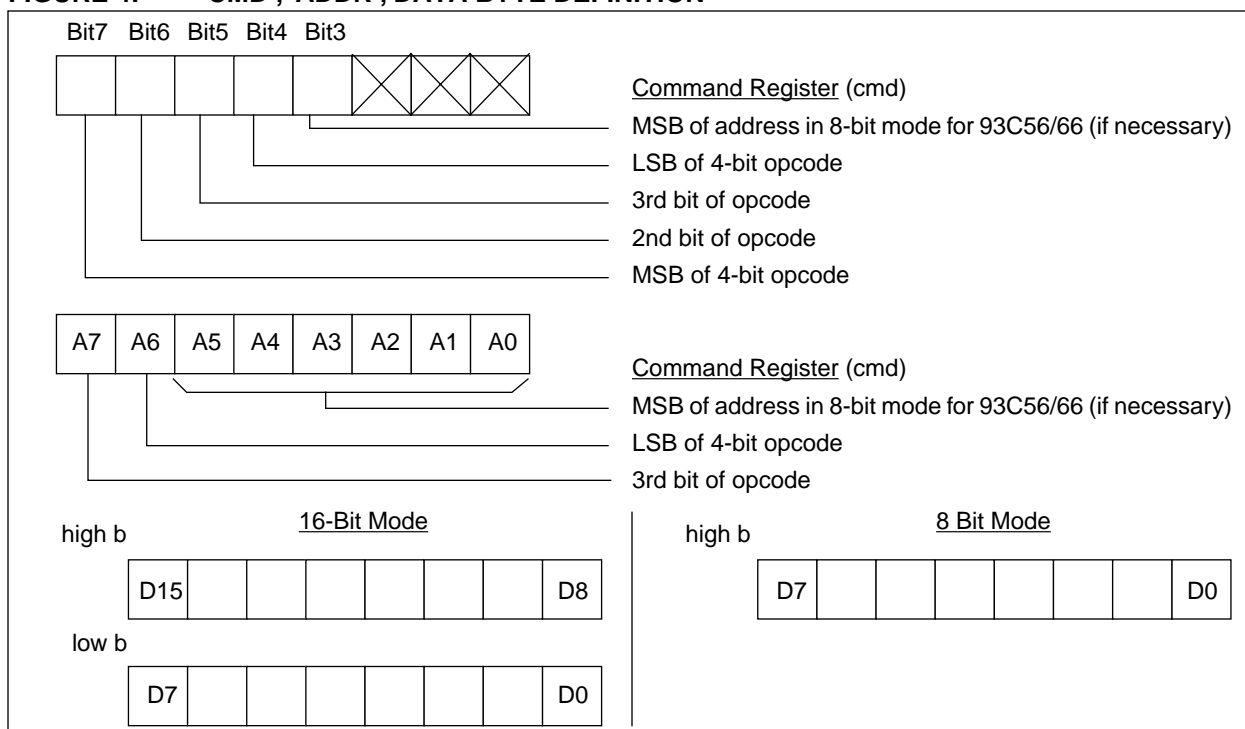
The 93 Series Serial EEPROMs are a simple and versatile method of increasing read/write memory capability in a PIC16C5X application. The 'generic' code in Appendix A makes it easy to incorporate in any PIC16C5X application. Any of the 93 Series Serial EEPROMs can be applied, while at the same time using a minimal amount of I/O, code and RAM resources.

Code size: 6 bytes of RAM.

Appendix A listing: 127 bytes program code (max.)
100 bytes program code (min.)

Appendix B listing: 86 bytes program code

FIGURE 4: 'CMD', 'ADDR', DATA BYTE DEFINITION



AN530

Please check the Microchip Worldwide Website at www.microchip.com for the latest version of the source code.

APPENDIX A: 93XCXX.ASM

MPASM 01.40 Released

93XCXX.ASM 1-16-1997 13:13:47

PAGE 1

```
LOC OBJECT CODE      LINE SOURCE TEXT
VALUE

00001          00001          TITLE          "R/W EEPROM"
00002          00002          LIST P = 16C54
00003          00003          ;Serial Eprom interface to PIC16C5X.
00004          00004          ;
00005          00005          ;
00006          00006          ;      Program:          93XCXX.ASM
00007          00007          ;      Revision Date:    8/26/92
00008          00008          ;                          10/31/92      Comments clarified REV 2.0
00009          00009          ;                          1-13-97      Compatibility with MPASMWIN 1.40
00010          00010          ;
00011          00011          ;*****
00012          00012          ;
00013          00013          ;Define Equates:
00014          00014          ;
000001FF      00015          PIC54 EQU 1FFH
00016          00016          ;
00017          00017          PAGE
00018          00018          ;
0000          00019          ORG 0
0000          00020          START
0000 0A7B     00021          goto main          ;run test program
00022          00022          ;
00023          00023          PAGE
00024          00024          ;
00000001     00025          TRUE EQU 1
00000000     00026          FALSE EQU 0
00000001     00027          F EQU 1
00000000     00028          S93C46 EQU FALSE
00000000     00029          S93LC46 EQU FALSE
00000000     00030          S93C56 EQU FALSE
00000000     00031          S93LC56 EQU FALSE
00000001     00032          S93C66 EQU TRUE
00000000     00033          S93LC66 EQU FALSE
00000001     00034          wire3 equ TRUE          ; for four-wire setup equate to FALSE
00000000     00035          org8 EQU FALSE
00036          00036          ;
00037          00037          IF S93LC46
00038          00038          IF org8
00039          00039          LC468 EQU TRUE
00040          00040          XC46 EQU FALSE
00041          00041          H16 EQU FALSE
00042          00042          H8 EQU FALSE
00043          00043          ELSE
00044          00044          LC468 EQU FALSE
00045          00045          XC46 EQU TRUE
00046          00046          H16 EQU FALSE
00047          00047          H8 EQU FALSE
00048          00048          ENDIF
00049          00049          ELSE
00050          00050          ENDIF
00051          00051          IF S93C46
00052          00052          LC468 EQU FALSE
00053          00053          XC46 EQU TRUE
00054          00054          H16 EQU FALSE
00055          00055          H8 EQU FALSE
00056          00056          ENDIF
```

```

00057     IF          S93C56 + S93C66 + S93LC56 + S93LC66
00058             IF          org8
00059 H8          EQU          TRUE
00060 H16         EQU          FALSE
00061 LC468      EQU          FALSE
00062 XC46       EQU          FALSE
00063             ELSE
00000001      00064 H16         EQU          TRUE
00000000      00065 H8          EQU          FALSE
00000000      00066 LC468      EQU          FALSE
00000000      00067 XC46       EQU          FALSE
00068             ENDIF
00069             ELSE
00070             ENDIF
00071
00072
00073
00074 ;*****
00075 ;*                               Register Assignments                               *
00076 ;*****
00077
00000000      00078 indir    equ     0          ;Use this register as source/destination
00079                                     ;for indirect addressing.
00000002      00080 pc        equ     2          ;PIC Program Counter.
00000003      00081 status   equ     3          ;PIC Status Register.
00000004      00082 fsr      equ     4          ;File Select Register.
00000005      00083 serial   equ     5          ;Port used for 93CX6 control.
00084                                     ;The following four registers must be
00085                                     ;located consecutively in memory.
0000001A      00086 cmd      equ     1a         ;This register contains the 4 bit
00087                                     ;command op code for 93CX6 as follows:
00088                                     ;bit 7 msb of command op code
00089                                     ;bit 6 next bit of op code
00090                                     ;bit 5 next bit of op code
00091                                     ;bit 4 lsb of op code
00092                                     ;bit 3 A8 of address in case of
00093                                     ;56/66 in 8 bit mode.
0000001B      00094 addr     equ     1b         ;memory address of lower 7/8 bits
0000001C      00095 highb    equ     1c         ;Used in read/write routines to store the
00096                                     ;upper byte of a 16 bit data word,
00097                                     ;or the data in a 8 bit data word
0000001D      00098 lowb     equ     1d         ;Used in read/write routines to store the
00099                                     ;lower byte of a 16 bit data word,
00100                                     ;or not used in 8 bit data word.
00101
0000001E      00102 cnthi    equ     1e         ;Used as the upper byte of a sixteen bit
00103                                     ;loop counter in RDYCHK routine.
0000001F      00104 cnt      equ     1f         ;Used as the lower byte of a sixteen bit
00105                                     ;loop counter in RDYCHK routine, and
00106                                     ;elsewhere as an eight bit counter.
0000001E      00107 temp_cmd  equ     1e         ;doubles as a temp register for cmd
0000001F      00108 temp_addr equ     1f         ;doubles as a temp register for addr
00109
00110 ;*****
00111 ;* Bit Assignments: The following assignments are for 3-wire. For
00112 ;* 4-wire assign din and dout to separate pins.
00113
00000000      00114 carry   equ     0          ;Carry Flag of Status Register.
00000002      00115 zflag    equ     2          ;Zero Flag of Status Register.
00116
00000002      00117 cs       equ     2          ;Port pin tied to CS on 93CX6.
00000001      00118 din      equ     1          ;Port pin tied to DI on 93CX6. 3-wire
00000001      00119 dout     equ     1          ;Port pin tied to DO on 93CX6. 3-wire
00000003      00120 clock    equ     3          ;Port pin tied to CLK on 93CX6.
00121
00122 ;*****

```

```

00123 ;*                               General Assignments                               *
00124 ;*****
00125
00000000 00126 no_err equ 0
00000001 00127 err1 equ 1
00000020 00128 tries equ 20 ;After issuing a WRITE, ERASE, ERAL, or WRAL
00129 ;command, the approximate number of machine
00130 ;cycles X 256 to wait for the RDY status.
00131 ;This value must be adjusted for operating
00132 ;frequencies other than 4 MHz.
00133
00000080 00134 read equ b'10000000' ;read command op code
00000040 00135 write equ b'01000000' ;write command op code
000000C0 00136 erase equ b'11000000' ;erase command op code
00000030 00137 ewen equ b'00110000' ;erase enable command op code
00000000 00138 ewds equ b'00000000' ;erase disable command op code
00000020 00139 eral equ b'00100000' ;erase all command op code
00000010 00140 wral equ b'00010000' ;write all command op code
00141
00142 ;*****
00143 ;*                               Macro Definitions                               *
00144 ;*****
00145
00146 sel MACRO ;Selects the 93CX6 device.
00147 bsf serial,cs ;Chip Select (CS) = '1' to select
00148 ENDM ;the device
00149
00150 dsel MACRO ;De-select the 93CX6 device.
00151 bcf serial,cs ;Chip Select (CS) = '0' to de-select
00152 ;the device.
00153 ENDM
00154
00155 strtbt MACRO ;Issue the Start Bit to the 93CX6.
00156 bsf serial,din ;Start Bit = '1'.
00157 clkkit ;Clock it out.
00158 ENDM
00159
00160 clkkit MACRO ;Clocks a serial data bit into or out
00161 ;of the 93CX6 device.
00162 bsf serial,clock ;Clock (CLK) = '1'.
00163
00164 nop ;Adjust the number of nop instructions
00165 ;between the assertion and de-assertion of
00166 ;CLK in proportion to the PIC operating
00167 ;frequency. Refer to the 93CX6 data for the
00168 ;minimum CLK period.
00169
00170 bcf serial,clock ;Clock (CLK) = '0'.
00171 ENDM
00172 ;
00173 PAGE
00174 ;*****
00175 ;*                               DOUTx                               *
00176 ;*****
00177 ;doutxx, outputs up to 11 bits of op code/data, depending on whether
00178 ;a 46/56/66 serial eeprom is being used. The number of bits over 8 are
00179 ;saved in the cmd register and the rest in the addr register. Before
00180 ;calling this routine the cmd and the addr registers should be loaded
00181 ;as follows:
00182 ;cmd reg.bits 7|6|5|4|3|2|1|0
00183 ;-----|---|---|---|---|
00184 ; X|X|X|X|X|X|X|Y --> not used
00185 ; X|X|X|X|X|X|Y|X --> mot used
00186 ; X|X|X|X|X|Y|X|X --> not used
00187 ; X|X|X|X|Y|X|X|X --> 9th bit of address if necessary
00188 ; X|X|X|Y|X|X|X|X --> lsb of command op code

```

```

00189 ;          X|X|Y|X|X|X|X|X| --> 3rd bit of command op code
00190 ;          X|Y|X|X|X|X|X|X| --> 2nd bit of command op code
00191 ;          Y|X|X|X|X|X|X|X| --> msb of command op code
00192 ;
00193 ;addr reg. 6/7/8 bits of address if necessary.
00194
00195     IF      H8
00196 dout11
00197         bcf    serial,din
00198         rlf    indir      ;rotate thru carry
00199         btfs   status,carry ;set?
00200         bsf    serial,din  ;yes, set output to 1
00201         clkit          ;clk data
00202     ENDIF
00203     IF      H16+H8
0001      00204 dout10
0001 0360      00205         rlf    indir, F      ;rotate thru carry
0002 0425      00206         bcf    serial,din      ;set output to 0
0003 0603      00207         btfs   status,carry ;set?
0004 0525      00208         bsf    serial,din  ;else set output to 1
0005 0565      00209         clkit          ;clk data
0006 0000      M          ;of the 93CX6 device.
0006 0000      M          ;Adjust the number of nop instructions
0006 0000      M          ;between the assertion and de-assertion of
0006 0000      M          ;CLK in proportion to the PIC operating
0006 0000      M          ;frequency. Refer to the 93CX6 data for the
0006 0000      M          ;minimum CLK period.
0007 0465      M          bcf    serial,clock ;Clock (CLK) = '0'.
00210     ENDIF
00211     IF      H8+H16+LC468
0008      00212 dout9
0008 0360      00213         rlf    indir, F      ;rotate thru carry
0009 0425      00214         bcf    serial,din      ;set output to 0
000A 0603      00215         btfs   status,carry ;set?
000B 0525      00216         bsf    serial,din  ;else set output to 1
000C 0565      00217         clkit          ;clk data
000C 0565      M          ;of the 93CX6 device.
000D 0000      M          bcf    serial,clock ;Clock (CLK) = '1'.
000D 0000      M          ;Adjust the number of nop instructions
000D 0000      M          ;between the assertion and de-assertion of
000D 0000      M          ;CLK in proportion to the PIC operating
000D 0000      M          ;frequency. Refer to the 93CX6 data for the
000D 0000      M          ;minimum CLK period.
000E 0465      M          bcf    serial,clock ;Clock (CLK) = '0'.
000F 02A4      00218         incf   fsr, F      ;inc pointer
00219     ENDIF
0010      00220 dout8
0010 0C08      00221         movlw  8          ;Initialize loop counter.
0011 003F      00222         movwf  cnt
00223 ;
0012 0425      00224 d_o_8 bcf    serial,din      ;Assume that the bit to be transferred
00225         ;is a '0'. Hence, de-assert DI.
0013 0360      00226         rlf    indir, F      ;Rotate the actual bit to be
00227         ;transferred into the carry bit.
0014 0603      00228         btfs   status,carry ;Test the carry, if our assumption was
00229         ;correct, skip the next instruction.
0015 0525      00230         bsf    serial,din  ;No, actual bit was a '1'. Assert DI.
00231         clkit          ;Clock the 93CX6.
0016 0565      M          ;of the 93CX6 device.
0016 0565      M          bcf    serial,clock ;Clock (CLK) = '1'.
0016 0565      M

```

AN530

```
0017 0000      M      nop                ;Adjust the number of nop instructions
              M                ;between the assertion and de-assertion of
              M                ;CLK in proportion to the PIC operating
              M                ;frequency. Refer to the 93CX6 data for the
              M                ;minimum CLK period.
              M
0018 0465      M      bcf      serial,clock ;Clock (CLK) = '0'.
0019 02FF      00232   decfsz  cnt, F      ;Repeat until cnt = 0.
001A 0A12      00233   goto   d_o_8      ;Cnt still > 0.
001B 0360      00234   rlf    indir, F   ;Restore register to its original condition.
001C 0425      00235   bcf    serial,din  ;make sure din is low
001D 0800      00236   retlw  no_err     ;Exit with good status.
              00237
              00238 ;*****
              00239 ;*                DIN8                *
              00240 ;*****
00241                ;Din8 will input 8 bits of data from the
00242                ;93CX6. Before calling this routine,
00243                ;the FSR must point to the register
00244                ;being used to hold the incoming data.
001E          00245   din8
              00246       IF      wire3
00247 ;set up the RAL as a input before proceeding
001E 0C02      00248       movlw  b'00000010' ;set up porta
001F 0005      00249       tris   serial ;
              00250       ENDIF
0020 0C08      00251       movlw  8 ;Initialize loop counter.
0021 003F      00252       movwf  cnt ;
              00253
0022          00254   d_i_8
0022 0360      00255       rlf    indir, F ;Make room for the incoming bit in the
              00256                ;destination register.
0023 0400      00257       bcf    indir,0 ;Assume that the incoming bit is a '0' and
              00258                ;clear the LSB of the destination register.
              00259       clkit ;Clock a bit in the 93CX6.
              M                ;of the 93CX6 device.
0024 0565      M      bsf    serial,clock ;Clock (CLK) = '1'.
              M
0025 0000      M      nop                ;Adjust the number of nop instructions
              M                ;between the assertion and de-assertion of
              M                ;CLK in proportion to the PIC operating
              M                ;frequency. Refer to the 93CX6 data for the
              M                ;minimum CLK period.
              M
0026 0465      M      bcf    serial,clock ;Clock (CLK) = '0'.
0027 0625      00260   btfsc  serial,dout ;Test the incoming bit, if our assumption
              00261                ;was correct, skip the next instruction.
0028 0500      00262       bsf    indir,0 ;No, actual bit is a '1'. Set the LSB of
              00263                ;the destination register.
0029 02FF      00264       decfsz  cnt, F ;Repeat until cnt = 0.
002A 0A22      00265       goto   d_i_8 ;Cnt still > 0.
              00266       IF      wire3
00267 ;setup RAL back to output
002B 0C00      00268       movlw  0 ;set RAL as output
002C 0005      00269       tris   serial ; /
              00270       ENDIF
002D 0800      00271       retlw  no_err ;Exit with good status.
              00272
              00273 ;*****
              00274 ;*                RDYCHK                *
              00275 ;*****
00276                ;Rdychk will read the 93CX6 READY/BUSY status
00277                ;and wait for RDY status within the allotted
00278                ;number of processor cycles. If RDY status
00279                ;is not present after this set period, the
00280                ;routine will return with an error status.
```

```

00281
002E      00282 rdychk
00283      IF      wire3
00284 ;set up RA1 as a input before proceeding
002E 0C02 00285      movlw  b'00000010'      ;set up porta
002F 0005 00286      tris   serial      ;
00287      ENDIF
0030 0C20 00288      movlw  tries      ;Initialize time-out counter.
0031 003E 00289      movwf  cnthi      ;
0032 007F 00290      clrf   cnt      ;
00291      dsel   ;De-select the 93CX6.
0033 0445      M      bcf   serial,cs      ;Chip Select (CS) = '0' to de-select
          M          ;the device.
00292
00293 ;      nop      ;NOTE: Check the 93CX6 data sheet for
00294      ;minimum CS low time. Depending upon
00295      ;processor frequency, a nop(s) may be
00296      ;between the assertion and de-assertion of
00297      ;Chip Select.
00298
00299      sel      ;Re-select the 93CX6.
0034 0545      M      bsf   serial,cs      ;Chip Select (CS) = '1' to select
0035 0625 00300 notrdy  btfsz  serial,dout      ;If DO is a '0', 93CX6 has yet to completed
00301      ;the last operation (still busy).
0036 0A3E 00302      goto   rdynoerr      ;skip to no error
0037 02FF 00303      decfsz cnt, F      ;No, not yet ready. Decrement the LSB of
00304      ;our 16 bit timer and check for expiration.
0038 0A35 00305      goto   notrdy      ;Still some time left. Try again.
0039 02FE 00306      decfsz cnthi, F      ;Least significant byte expired - decrement
00307      ;and check for expiration of the MSB.
003A 0A35 00308      goto   notrdy      ;Still some time left. Try again.
00309      IF      wire3
00310 ;setup RA1 back to output
003B 0C00 00311      movlw  0      ;set RA1 as output
003C 0005 00312      tris   serial      ; /
00313      ENDIF
003D 0801 00314      retlw  err1      ;RDY status was not present in the allotted
00315      ;time, return with error status.
003E
00316 rdynoerr
00317      IF      wire3
00318 ;setup RA1 back to output
003E 0C00 00319      movlw  0      ;set porta as output
003F 0005 00320      tris   serial      ; /
00321      ENDIF
0040 0800 00322      retlw  no_err
00323
00324 ;*****
00325 ;*          SEE          *
00326 ;*****
00327
00328      ;See will control the entire operation of a
00329      ;93CX6 device. Prior to calling the routine,
00330      ;load a valid command/memory address into
00331      ;location cmd, and for WRITE or WRAL
00332      ;commands, load registers highb and lowb with
00333      ;16 bits of write data. Upon exit, the W
00334      ;register will contain the completion status.
00335      ;Only 93CX6 instructions which require a
00336      ;status check can return with an error as the
00337      ;completion status. The values that denote
00338      ;the completion status are defined as
00339      ;variables 'error' and 'no_err' in the
00340      ;general assignments section.
00341
0041      00342 see
0041 021A      00343      movf   cmd,W      ;save cmd

```

AN530

```
0042 003E      00344      movwf    temp_cmd
0043 021B      00345      movf     addr,W          ;save addr
0044 003F      00346      movwf   temp_addr      ;
0045 0C1A      00347      movlw   cmd             ;Load W with the location of the cmd
                                00348      ;register.
0046 0024      00349      movwf   fsr            ;Transfer that information into the
                                00350      ;File Select Register. The fsr now
                                00351      ;points to location cmd.
                                00352      sel
0047 0545      M          bsf     serial,cs      ;Chip Select (CS) = '1' to select
                                00353      strtbt
0048 0525      M          bsf     serial,din   ;Start Bit = '1'.
                                M          clklt             ;Clock it out.
                                M          ;of the 93CX6 device.
0049 0565      M          bsf     serial,clock ;Clock (CLK) = '1'.
                                M
004A 0000      M          nop
                                M          ;Adjust the number of nop instructions
                                M          ;between the assertion and de-assertion of
                                M          ;CLK in proportion to the PIC operating
                                M          ;frequency. Refer to the 93CX6 data for the
                                M          ;minimum CLK period.
                                M
004B 0465      M          bcf     serial,clock ;Clock (CLK) = '0'.
                                00354 ;
004C 06FA      00355      btfsc   cmd,7          ;bit 7 = 0?
                                00356      IF      XC46
                                00357      goto    sca8          ;xfer 8 bit cmd/adr
                                00358      ENDIF
                                00359      IF      LC468
                                00360      goto    sca9          ;xfer 9 bit cmd/adr
                                00361      ENDIF
                                00362      IF      H16
004D 0A73      00363      goto    sca10         ;xfer 10 bit cmd/adr
                                00364      ENDIF
                                00365      IF      H8
                                00366      goto    sca11        ;xfer 11 bit cmd/adr
                                00367      ENDIF
                                00368
                                00369 ;
                                goto    set_cmd_addr ;no then set cmd/addr
004E 07DA      00370      btfss   cmd,6          ;bit 6 = 0 ?
                                00371      IF      XC46
                                00372      goto    sc8          ;xfer 8 bit cmd/adr
                                00373      ENDIF
                                00374      IF      LC468
                                00375      goto    sc9          ;xfer 9 bit cmd/adr
                                00376      ENDIF
                                00377      IF      H16
004F 0A75      00378      goto    sc10         ;xfer 10 bit cmd/adr
                                00379      ENDIF
                                00380      IF      H8
                                00381      goto    sc11        ;xfer 11 bit cmd/adr
                                00382      ENDIF
                                00383 ;
                                goto    set_cmd      ;yes then set cmd
                                00384      IF      XC46
                                00385      goto    sca8          ;xfer 8 bit cmd/adr
                                00386      ENDIF
                                00387      IF      LC468
                                00388      goto    sca9          ;xfer 9 bit cmd/adr
                                00389      ENDIF
                                00390      IF      H16
0050 0A73      00391      goto    sca10         ;xfer 10 bit cmd/adr
                                00392      ENDIF
                                00393      IF      H8
                                00394      goto    sca11        ;xfer 11 bit cmd/adr
                                00395      ENDIF
00396
```

```

00397 ;      goto      set_cmd_addr      ;else set cmd/addr
0051      00398 see1
0051 021E      00399      movf      temp_cmd,W      ;retore cmd
0052 003A      00400      movwf     cmd              ;      /
0053 021F      00401      movf      temp_addr,W      ;restore addr
0054 003B      00402      movwf     addr              ;      /
0055 06DA      00403      btfsc    cmd,6              ;Check for a WRITE or ERASE command.
0056 0A5F      00404      goto     see2              ;Yes, parse the command further.
0057 06FA      00405      btfsc    cmd,7              ;Check for a READ command.
0058 0A69      00406      goto     read_            ;Yes, process READ command.
0059 06BA      00407      btfsc    cmd,5              ;Check for a EWEN or ERAL command.
005A 0A66      00408      goto     see3              ;Yes, parse the command further.
005B 069A      00409      btfsc    cmd,4              ;Check for a WRAL command.
005C 0A6E      00410      goto     write_           ;Yes, process WRITE/WRAL command.
00411
00412 exit_    dsel
005D 0445      M          bcf      serial,cs      ;No further processing required; 93CX6
M          ;Chip Select (CS) = '0' to de-select
00413      ;the device.
00414 ;      ;command completed.
005E 0800      00415      retlw    no_err           ;Return with good completion status.
00416
005F 07FA      00417 see2    btfss    cmd,7              ;Check for a ERASE command.
0060 0A6E      00418      goto     write_           ;No, process WRITE command.
0061 092E      00419 exit2_  call     rdychk           ;ERASE command requires a status check.
00420      dsel
0062 0445      M          bcf      serial,cs      ;Chip Select (CS) = '0' to de-select
M          ;the device.
0063 01E2      00421      addwf    pc, F            ;Compute completion status from
00422      ;results of status check.
0064 0800      00423      retlw    no_err           ;Return with good completion status.
0065 0801      00424      retlw    err1            ;Return with bad completion status.
00425
0066 069A      00426 see3    btfsc    cmd,4              ;Check for a EWEN command.
0067 0A5D      00427      goto     exit_            ;Yes, no further processing required,
00428      ;exit now.
0068 0A61      00429      goto     exit2_           ;No, ERAL command which requires a
00430      ;status check.
00431
0069 02A4      00432 read_  incf     fsr, F            ;Increment the File Select Register to
00433      ;point to the register receiving the upper
00434      ;byte of the incomming 93CX6 data word.
006A 091E      00435      call     din8             ;Input the upper byte.
00436      IF      org8
00437      ELSE
006B 02A4      00438      incf     fsr, F            ;Increment the File Select Register to point
00439      ;to the register receiving the lower byte.
006C 091E      00440      call     din8             ;Input 8 more bits.
00441      ENDIF
006D 0A5D      00442      goto     exit_            ;No further processing required, exit now.
00443
006E 02A4      00444 write_  incf     fsr, F            ;Increment the File Select Register to point
00445      ;to the upper byte of the 16 bit 93CX6 data
00446      ;word to be transmitted.
006F 0910      00447      call     dout8            ;Output that byte.
00448      IF      org8
00449      ELSE
0070 02A4      00450      incf     fsr, F            ;Increment the File Select Register to point
00451      ;to the lower byte.
0071 0910      00452      call     dout8            ;Output the lower byte of the 16 bit 93CX6
00453      ;data word.
00454      ENDIF
0072 0A61      00455      goto     exit2_           ;Exit with a status check.
00456
00457 ;
00458 ;

```

```

00459     IF      XC46
00460 sca8
00461     movlw   b'11000000'    ;clr all but hi 2
00462     andwf   cmd,W          ;save in w
00463     iorwf   addr          ;mask in addr.
00464     incf    fsr            ;inc FSR
00465     call    dout8          ;output
00466     goto    seel          ;return
00467     ENDIF
00468     IF      LC468
00469 ;
00470 sca9
00471     bcf     addr,7          ;clr hi of addr
00472     btfsc   cmd,6          ;xfer cmd's bit 6
00473     bsf     addr,7          ;to addr's bit 7
00474     call    dout9          ;output
00475     goto    seel          ;return
00476     ENDIF
00477     IF      H16
00478 ;
00479 sca10
00480     call    dout10         ;output cmd reg
00481     goto    seel          ;return
00482     ENDIF
00483 ;
00484     IF      H8
00485 scall
00486     bcf     cmd,5           ;xfer cmd's bit 3 to
00487     btfsc   cmd,3           ;cmd's bit 5
00488     bsf     cmd,5           ;
00489     call    dout11         ;output
00490     goto    seel          ;return
00491 ;
00492     ENDIF
00493 ;
00494 ;
00495 ;
00496     IF      XC46
00497 sc8
00498     movf    cmd,W           ;get command
00499     movwf   addr           ;save in addr
00500     incf    fsr            ;inc pointer
00501     call    dout8          ;output
00502     goto    seel          ;return
00503 ;
00504     ENDIF
00505     IF      LC468
00506 sc9
00507     rlf     cmd,W           ;rotate cmd left
00508     movwf   addr           ;save in addr
00509     call    dout9          ;xfer 9 bits
00510     goto    seel          ;return
00511     ENDIF
00512     IF      H16
00513 ;
00514 sca10
00515     rlf     cmd, F          ;rotate cmd
00516     rlf     cmd,W          ;left twice
00517     movwf   addr           ;save in addr
00518     clrfs  cmd             ;clear command
00519     call    dout10         ;xfer 10 bits
00520     goto    seel          ;return
00521     ENDIF
00522 ;
00523     IF      H8
00524 sc11

```

```

00525         bcf     addr,7           ;xfer cmd's bit 4
00526         btfsc  cmd,4            ;to addr's bit 7
00527         bsf     addr,7           ;
00528         call   dout11           ;xfer 11 bits
00529         goto   see1             ;return
00530         ENDIF
00531 ;*****
00532 ;*                               Test Program                               *
00533 ;*****
00534 ;
007B          00535 main                ;We've include a sample program to
00536                ;exercise the PIC to 93C66 interface
00537                ;using a simple erase, write and verify
00538                ;routine.8 bit organization has been
00539                ;used with a 3 wire interface.
00540
007B 0065     00541         clr     serial           ;Clear the port tied to the 93C66 device.
007C 0CF0     00542         movlw  b'11110000'         ;Intialize the data direction register
007D 0005     00543         tris   serial           ;for that port.
00544
007E 0C30     00545         movlw  ewen            ;Load W with the Erase/Write Enable command.
007F 003A     00546         movwf  cmd             ;Transfer W into cmd register.
0080 0941     00547         call   see             ;Enable the 93C66 device.
00548
0081 0C20     00549         movlw  eral            ;Load W with the Erase All command.
0082 003A     00550         movwf  cmd             ;Transfer W into cmd register.
0083 0941     00551         call   see             ;Erase the 93C66.
0084 0F01     00552         xorlw  err1           ;Check completion status.
0085 0643     00553         btfsc  status, zflag      ;Test for error condition.
0086 0AA6     00554         goto   errloop        ;Yes, bad completion status, error-out.
00555
00556                ;Write loop:
000000AA     00557 tstptrn equ      0xAA           ;Define the test pattern to be written.
00558
0087 0C40     00559         movlw  write          ;Load W with the Write command.
0088 003A     00560         movwf  cmd             ;Transfer W into cmd register.
0089 0CAA     00561         movlw  tstptrn        ;Intialize the 93C66 data registers with
00562                ;write data.
008A 003C     00563         movwf  highb          ;load in high byte only
00564                ;since 8 bit low byte is ignored
008B 007B     00565         clr     addr            ;start at addr 0
008C 0941     00566 test1   call   see             ;Write data word into 93C66 device.
008D 0F01     00567         xorlw  err1           ;Check completion status.
008E 0643     00568         btfsc  status,zflag      ;Test for error condition.
008F 0AA6     00569         goto   errloop        ;Yes, bad completion status, error-out.
0090 03FB     00570         incfsz  addr, F           ;No, increment the 8 bit memory address
00571                ;field.
0091 0A8C     00572         goto   test1          ;write another location
0092 077A     00573         btfss  cmd,3             ;see if all done
0093 0A95     00574         goto   wrt_nxt_pg        ;no then write next page
0094 0A97     00575         goto   read_tst         ;read written data
0095
0095 057A     00576 wrt_nxt_pg bsf     cmd,3             ;set page bit
0096 0A8C     00578         goto   test1          ;No, write another location.
00579
00580                ;Read loop:
0097
0097 0C80     00581 read_tst  movlw  read            ;Load W with the Read command.
0098 003A     00583         movwf  cmd             ;Transfer W into cmd register.
0099 0941     00584 test2   call   see             ;Read addressed data word from 93C66 device.
009A 0CAA     00585         movlw  tstptrn        ;Load W with the pattern written.
009B 009C     00586         subwf  highb,W          ;Verify the data read against what was
00587                ;written.
009C 0743     00588         btfss  status,zflag      ;Same?
009D 0AA6     00589         goto   errloop        ;No, error-out.
009E 03FB     00590         incfsz  addr, F           ;Yes, both byte correct, increment the 8 bit

```

AN530

```
00591 ;memory address field.
009F 0A99 00592 goto test2 ;do next byte
00A0 077A 00593 btfss cmd,3 ;check page bit
00A1 0AA3 00594 goto rd_nxt_pg ;no then chk next page
00A2 0AA5 00595 goto allok ;all done!!!
00A3 00596 rd_nxt_pg
00A3 057A 00597 bsf cmd,3 ;set page bit
00A4 0A99 00598 goto test2 ;No, read another location.
00599
00A5 0AA5 00600 allok goto allok ;Home safe!
00601 ;
00602 ;
00A6 00603 errloop
00A6 0AA6 00604 goto errloop
00605 ;
00606 ;
00607 ;
00608 ;KEY DEFINITIONS
00609 ;
01FF 00610 ORG PIC54
01FF 00611 SYS_RESET
01FF 0A00 00612 GOTO START
00613 ;
00614 END
MEMORY USAGE MAP ('X' = Used, '-' = Unused)

0000 : XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX
0040 : XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX
0080 : XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXX-----
01C0 : -----X
```

All other memory blocks unused.

Program Memory Words Used: 168
Program Memory Words Free: 344

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

Please check the Microchip Worldwide Website at www.microchip.com for the latest version of the source code.

APPENDIX B: 93C46.ASM

MPASM 01.40 Released

93C46.ASM 1-16-1997 13:13:14

PAGE 1

```

LOC OBJECT CODE      LINE SOURCE TEXT
VALUE

00001 ;*****
00002 ;*                               PICALC Directives Section          *
00003 ;*****
00004
00005      SUBTTL  "93C46 3 WIRE INTERFACE ROUTINE"
00006      LIST    P=16C54,N=40,C=132
00007
00008 ;*****
00009 ;
00010 ;
00011 ;      Program:           93C46.ASM
00012 ;      Revision Date:
00013 ;                               1-13-97      Compatibility with MPASMWIN 1.40
00014 ;
00015 ;*****
00016 ;
00017
00018 ;*****
00019 ;*                               Register Assignments          *
00020 ;*****
00021
00000000 00022 indir  equ    0x00      ;Use this register as source/destination for
00023                                     ;indirect addressing.
00000002 00024 pc     equ    0x02      ;PIC Program Counter.
00000003 00025 status equ    0x03      ;PIC Status Register.
00000004 00026 fsr    equ    0x04      ;File Select Register.
00000005 00027 serial equ    0x05      ;Port used for 93C46 control. Port A is
00028                                     ;4 bits wide, we'll use 3 or 4 of Port A.
00000001 00029 F      EQU    1
00030                                     ;The following three registers must be
00031                                     ;located consecutively in memory.
00000010 00032 cmd     equ    0x10      ;This register contains the 2 bit 93C46
00033                                     ;command is the upper 2 bit positions and
00034                                     ;memory address in the lower 6.
00000011 00035 highb   equ    0x11      ;Used in read/write routines to store the
00036                                     ;upper byte of a 16 bit 93C46 data word.
00000012 00037 lowb    equ    0x12      ;Used in read/write routines to store the
00038                                     ;lower byte of a 16 bit 93C46 data word.
00039
00000013 00040 cnthi   equ    0x13      ;Used as the upper byte of a sixteen bit loop
00041                                     ;counter in RDYCHK routine.
00000014 00042 cnt     equ    0x14      ;Used as the lower byte of a sixteen bit loop
00043                                     ;counter in RDYCHK routine, and elsewhere as
00044                                     ;an eight bit counter.
00045
00046 ;*****
00047 ;* Bit Assignments: The following assignment is for 3-wire setup.
00048 ;* For 4-wire setup assign din and dout to seperate pins.
00049
00000000 00050 carry   equ    0          ;Carry Flag of Status Register.
00000002 00051 zflag   equ    2          ;Zero Flag of Status Register.
00052
00053 ;For the 3 wire interface, connect the din and dout to the same
00054 ;i/o line of the PIC16C5X.
00000000 00055 cs     equ    0          ;Port pin tied to CS on 93C46.
00000001 00056 din    equ    1          ;Port pin tied to DI on 93C46. 3-wire

```

AN530

```
00000001    00057 dout    equ    1            ;Port pin tied to DO on 93C46. 3-wire
00000002    00058 clock   equ    2            ;Port pin tied to CLK on 93C46.
00059
00060 ;*****
00061 ;*                               General Assignments                               *
00062 ;*****
00063
00000000    00064 no_err   equ    0            ;
00000001    00065 err1     equ    1            ;
00000004    00066 tries   equ    0x04         ;After issuing a WRITE, ERASE, ERAL, or WRAL
00067 ;command, the approximate number of machine
00068 ;cycles X 256 to wait for the RDY status.
00069 ;This value must be adjusted for operating
00070 ;frequencies other than 4 MHz.
00071
00000080    00072 read     equ    0x80         ;93C46 Read command.
00000040    00073 write    equ    0x40         ;93C46 Write command.
000000C0    00074 erase    equ    0xC0         ;93C46 Erase command.
00000030    00075 ewen     equ    0x30         ;93C46 Erase/Write Enable command.
00000000    00076 ewds     equ    0x00         ;93C46 Erase/Write Disable command.
00000020    00077 eral     equ    0x20         ;92CXX Erase All command.
00000010    00078 wral     equ    0x10         ;92CXX Write All command.
00079
00080 ;*****
00081 ;*                               Macro Definitions                               *
00082 ;*****
00083
00084 sel      MACRO                               ;Selects the 93C46 device.
00085         bsf    serial,cs                    ;Chip Select (CS) = '1' to select the device.
00086         ENDM
00087
00088 dsel     MACRO                               ;De-select the 93C46 device.
00089         bcf    serial,cs                    ;Chip Select (CS) = '0' to de-select the
00090 ;device.
00091         ENDM
00092
00093 strtbt   MACRO                               ;Issue the Start Bit to the 93C46.
00094         bsf    serial,din                    ;Start Bit = '1'.
00095         clkit                               ;Clock it out.
00096         ENDM
00097
00098 clkit    MACRO                               ;Clocks a serial data bit into or out of the
00099 ;93C46 device.
00100         bsf    serial,clock                 ;Clock (CLK) = '1'.
00101
00102         nop                                ;Adjust the number of nop instructions
00103 ;between the assertion and de-assertion of
00104 ;CLK in proportion to the PIC operating
00105 ;frequency. Refer to the 93C46 data for the
00106 ;minimum CLK period.
00107
00108         bcf    serial,clock                 ;Clock (CLK) = '0'.
00109         ENDM
00110
00111 ;*****
00112 ;*                               Power-On/Reset Entry Point                       *
00113 ;*****
00114
01FF      00115 reset_   org    0x1FF
01FF 0A56 00116         goto    main
00117
00118 ;*****
00119 ;*                               93C46 Routines                                   *
00120 ;*****
0000      00121         org    0x000                    ;Locate all subroutines in the lower half of
00122 ;a Program Memory Page.
```

```

00123
00124 ;*****
00125 ;*                               DOUT8                               *
00126 ;*****
00127                               ;Dout8 will output 8 bits of data to the
00128                               ;93C46. Before calling this routine, the FSR
00129                               ;must point to the byte being transmitted.
00130
0000 0C08 00131 dout8   movlw   0x08           ;Initialize loop counter.
0001 0034 00132         movwf  cnt             ;
00133
0002 0425 00134 d_o_8   bcf     serial,din      ;Assume that the bit to be transfered is a
00135                               ;'0'. Hence, de-assert DI.
0003 0360 00136         rlf     indir, F        ;Rotate the actual bit to be transferred into
00137                               ;the carry bit.
0004 0603 00138         btfsc  status,carry    ;Test the carry, if our assumption was
00139                               ;correct, skip the next instruction.
0005 0525 00140         bsf     serial,din      ;No, actual bit was a '1'. Assert DI.
00141         clkkit                          ;Clock the 93C46.
00142         M                               ;93C46 device.
0006 0545 00143         bsf     serial,clock    ;Clock (CLK) = '1'.
00144         M
00145         M
0007 0000 00146         nop                      ;Adjust the number of nop instructions
00147         M                               ;between the assertion and de-assertion of
00148         M                               ;CLK in proportion to the PIC operating
00149         M                               ;frequency. Refer to the 93C46 data for the
00150         M                               ;minimum CLK period.
00151         M
0008 0445 00152         bcf     serial,clock    ;Clock (CLK) = '0'.
0009 02F4 00153         decfsz cnt, F          ;Repeat until cnt = 0.
000A 0A02 00154         goto   d_o_8         ;Cnt still > 0.
000B 0360 00155         rlf     indir, F        ;Restore register to its original condition.
000C 0800 00156         retlw  no_err         ;Exit with good status.
00157
00147 ;*****
00148 ;*                               DIN8                               *
00149 ;*****
00150                               ;Din8 will input 8 bits of data from the
00151                               ;93C46. Before calling this routine, the FSR
00152                               ;must point to the register being used to
00153                               ;hold the incoming data.
000D 0C08 00154 din8   movlw   0x08           ;Initialize loop counter.
000E 0034 00155         movwf  cnt             ;
00156 ;for the 3 wire interface the direction of the i/o line connected to
00157 ;din and dout has to be converted from an output to an input.
000F 0C02 00158         movlw  b'00000010'    ;convert RA1 to an input
0010 0005 00159         tris   serial        ; /
00160
00161 d_i_8   clkkit                          ;Clock a bit out of the 93C46.
00162         M                               ;93C46 device.
0011 0545 00163         bsf     serial,clock    ;Clock (CLK) = '1'.
00164         M
0012 0000 00165         nop                      ;Adjust the number of nop instructions
00166         M                               ;between the assertion and de-assertion of
00167         M                               ;CLK in proportion to the PIC operating
00168         M                               ;frequency. Refer to the 93C46 data for the
00169         M                               ;minimum CLK period.
00170         M
0013 0445 00171         bcf     serial,clock    ;Clock (CLK) = '0'.
0014 0360 00172         rlf     indir, F        ;Make room for the incoming bit in the
00173         00173                               ;destination register.
0015 0400 00174         bcf     indir,0         ;Assume that the incoming bit is a '0' and
00175         00175                               ;clear the LSB of the destination register.
0016 0625 00176         btfsc  serial,dout    ;Test the incoming bit, if our assumption
00177         00177                               ;was correct, skip the next instruction.
0017 0500 00178         bsf     indir,0         ;No, actual bit is a '1'. Set the LSB of the

```

```

00169                                ;destination register.
0018 02F4 00170      decfsz  cnt, F      ;Repeat until cnt = 0.
0019 0A11 00171      goto    d_i_8      ;Cnt still > 0
00172 ;for a 3 wire interface, convert the RAl line back to an output
001A 0C00 00173      movlw   0          ;make RAl to an output
001B 0005 00174      tris    serial      ; /
001C 0800 00175      retlw   no_err     ;Exit with good status.
00176
00177 ;*****
00178 ;*                                RDYCHK                                *
00179 ;*****
00180                                ;Rdychk will read the 93C46 READY/BUSY status
00181                                ;and wait for RDY status within the allotted
00182                                ;number of processor cycles. If RDY status
00183                                ;is not present after this set period, the
00184                                ;routine will return with an error status.
00185
001D 0C04 00186 rdychk  movlw   tries     ;Initialize time-out counter.
001E 0033 00187      movwf   cnthi      ;
001F 0074 00188      clrf    cnt        ;
00189 ;for a 3 wire interface, make the RAl line an input
0020 0C02 00190      movlw   b'00000010' ;
0021 0005 00191      tris    serial      ;
00192      dsel                                ;De-select the 93C46.
0022 0405      M      bcf     serial,cs    ;Chip Select (CS) = '0' to de-select the
      M                                ;device.
00193
00194 ;      nop                                ;NOTE: Check the 93C46 data sheet for
00195                                ;minimum CS low time. Depending upon
00196                                ;processor frequency, a nop(s) may be
00197                                ;between the assertion and de-assertion of
00198                                ;Chip Select.
00199
00200      sel                                ;Re-select the 93C46.
0023 0505      M      bsf     serial,cs    ;Chip Select (CS) = '1' to select the device.
0024 0625 00201 notrdy  btfsc   serial,dout ;If DO is a '0', 93C46 has yet to completed
00202                                ;the last operation (still busy).
0025 0A2D 00203      goto    no_error    ;Otherwise RDY status is present within the
00204                                ;alloted time, and return with good status.
0026 02F4 00205      decfsz  cnt, F      ;No, not yet ready. Decrement the LSB of our
00206                                ;16 bit timer and check for expiration.
0027 0A24 00207      goto    notrdy     ;Still some time left. Try again.
0028 02F3 00208      decfsz  cnthi, F    ;Least significant byte expired - decrement
00209                                ;and check for expiration of the MSB.
0029 0A24 00210      goto    notrdy     ;Still some time left. Try again.
00211 ;for a 3 wire interface, convert RAl line back to an ouput
002A 0C00 00212      movlw   0          ;convert RAl to an output
002B 0005 00213      tris    serial      ; /
002C 0801 00214      retlw   err1      ;RDY status was not present in the allotted
00215                                ;time, return with error status.
002D
00216 no_error
00217 ;for a 3 wire interface, convert RAl line back to an ouput
002D 0C00 00218      movlw   0          ;convert RAl to an output
002E 0005 00219      tris    serial      ; /
002F 0800 00220      retlw   no_err     ;
00221
00222
00223 ;*****
00224 ;*                                SEE                                *
00225 ;*****
00226
00227                                ;See will control the entire operation of a
00228                                ;93C46 device. Prior to calling the routine,
00229                                ;load a valid command/memory address into
00230                                ;location cmd, and for WRITE or WRAl
00231                                ;commands, load registers highb and lowb with

```

```

00232                ;16 bits of write data. Upon exit, the W
00233                ;register will contain the completion status.
00234                ;Only 93C46 instructions which require a
00235                ;status check can return with an error as the
00236                ;completion status. The values that denote
00237                ;the completion status are defined as
00238                ;variables 'error' and 'no_err' in the
00239                ;general assignments section.
00240
0030 0C10          00241 see    movlw   cmd      ;Load W with the location of the cmd
00242                ;register.
0031 0024          00243        movwf   fsr      ;Transfer that information into the File
00244                ;Select Register. The fsr now points to
00245                ;location cmd.
00246                sel
0032 0505          M        bsf     serial,cs   ;Chip Select (CS) = '1' to select the device.
00247                strtbt
0033 0525          M        bsf     serial,din  ;Start Bit = '1'.
M                clklt
M                ;Clock it out.
M                ;93C46 device.
0034 0545          M        bsf     serial,clock ;Clock (CLK) = '1'.
M
0035 0000          M        nop
M                ;Adjust the number of nop instructions
M                ;between the assertion and de-assertion of
M                ;CLK in proportion to the PIC operating
M                ;frequency. Refer to the 93C46 data for the
M                ;minimum CLK period.
M
0036 0445          M        bcf     serial,clock ;Clock (CLK) = '0'.
0037 0900          00248        call   dout8   ;Transmit the 2 bit command and six bit
00249                ;address.
0038 06D0          00250        btfsc   cmd,6     ;Check for a WRITE or ERASE command.
0039 0A42          00251        goto    see2    ;Yes, parse the command further.
003A 06F0          00252        btfsc   cmd,7     ;Check for a READ command.
003B 0A4C          00253        goto    read_   ;Yes, process READ command.
003C 06B0          00254        btfsc   cmd,5     ;Check for a EWEN or ERAL command.
003D 0A49          00255        goto    see3    ;Yes, parse the command further.
003E 0690          00256        btfsc   cmd,4     ;Check for a WRAL command.
003F 0A51          00257        goto    write_  ;Yes, process WRITE/WRAL command.
00258
0040 0405          00259 exit_   dsel
M                bcf     serial,cs   ;No further processing required; 93C46
M                ;Chip Select (CS) = '0' to de-select the
00260                ;device.
0041 0800          00261        retlw  no_err  ;command completed.
00262                ;Return with good completion status.
0042 07F0          00263 see2    btfss   cmd,7     ;Check for a ERASE command.
0043 0A51          00264        goto    write_  ;No, process WRITE command.
0044 091D          00265 exit2_  call   rdychk  ;ERASE command requires a status check.
00266                dsel
0045 0405          M        bcf     serial,cs   ;De-select the 93C46.
M                ;Chip Select (CS) = '0' to de-select the
0046 01E2          00267        addwf  pc, F    ;device.
00268                ;Compute completion status from results of
0047 0800          00269        retlw  no_err  ;status check.
0048 0801          00270        retlw  err1    ;Return with good completion status.
00271                ;Return with bad completion status.
0049 0690          00272 see3    btfsc   cmd,4     ;Check for a EWEN command.
004A 0A40          00273        goto    exit_   ;Yes, no further processing required, exit
00274                ;now.
004B 0A44          00275        goto    exit2_  ;No, ERAL command which requires a status
00276                ;check.
00277
004C 02A4          00278 read_   incf   fsr, F    ;Increment the File Select Register to point
00279                ;to the register receiving the upper byte of
00280                ;the incoming 93C46 data word.

```

AN530

```
004D 090D      00281      call    din8          ;Input the upper byte.
004E 02A4      00282      incf   fsr, F        ;Increment the File Select Register to point
                                00283                        ;to the register receiving the lower byte.
004F 090D      00284      call    din8          ;Input 8 more bits.
0050 0A40      00285      goto   exit_         ;No further processing required, exit now.
                                00286
0051 02A4      00287 write_ incf   fsr, F        ;Increment the File Select Register to point
                                00288                        ;to the upper byte of the 16 bit 93C46 data
                                00289                        ;word to be transmitted.
0052 0900      00290      call    dout8        ;Output that byte.
0053 02A4      00291      incf   fsr, F        ;Increment the File Select Register to point
                                00292                        ;to the lower byte.
0054 0900      00293      call    dout8        ;Output the lower byte of the 16 bit 93C46
                                00294                        ;data word.
0055 0A44      00295      goto   exit2_       ;Exit with a status check.
                                00296
                                00297 ;*****
00298 ;*                Test Program                *
00299 ;*****
00300 ;
0056           00301 main          ;We've include a sample program to exercise
                                00302                        ;the PIC to 93C46 interface using a simple
                                00303                        ;erase, write and varify routine.
                                00304
0056 0065      00305      clrf   serial        ;Clear the port tied to the 93C46 device.
0057 0CF4      00306      movlw  b'11110100'   ;Intialize the data direction register for
0058 0005      00307      tris   serial        ;that port.
                                00308
0059 0C30      00309      movlw  ewen          ;Load W with the Erase/Write Enable command.
005A 0030      00310      movwf  cmd           ;Transfer W into cmd register.
005B 0930      00311      call   see           ;Enable the 93C46 device.
                                00312
005C 0C20      00313      movlw  eral          ;Load W with the Erase All command.
005D 0030      00314      movwf  cmd           ;Transfer W into cmd register.
005E 0930      00315      call   see           ;Erase the 93C46.
005F 0F01      00316      xorlw  err1          ;Check completion status.
0060 0643      00317      btfsc  status, zflag ;Test for error condition.
0061 0A81      00318      goto   errloop      ;Yes, bad completion status, error-out.
                                00319
                                00320                        ;Write loop:
0000001F      00321 loopcnt equ    0x1F ;Define an unused location for our test
                                00322                        ;program loop counter.
000000AA      00323 tstptrn equ    0xAA  ;Define the test pattern to be written.
                                00324
0062 0C40      00325      movlw  .64           ;Initialize that counter.
0063 003F      00326      movwf  loopcnt      ;
0064 0C40      00327      movlw  write         ;Load W with the Write command.
0065 0030      00328      movwf  cmd           ;Transfer W into cmd register.
0066 0CAA      00329      movlw  tstptrn      ;Intialize the 93C46 data registers with
                                00330                        ;write data.
0067 0031      00331      movwf  highb        ;
0068 0032      00332      movwf  lowb         ;
0069 0930      00333 test1  call   see           ;Write data word into 93C46 device.
006A 0F01      00334      xorlw  err1          ;Check completion status.
006B 0643      00335      btfsc  status,zflag ;Test for error condition.
006C 0A81      00336      goto   errloop      ;Yes, bad completion status, error-out.
006D 02B0      00337      incf   cmd, F        ;No, increment the 6 bit memory address
                                00338                        ;field.
006E 02FF      00339      decfsz loopcnt, F   ;Have we written all 64 locations?
006F 0A69      00340      goto   test1        ;No, write another location.
                                00341
                                00342                        ;Read loop:
                                00343
0070 0C40      00344      movlw  .64           ;Initialize loop counter.
0071 003F      00345      movwf  loopcnt      ;
0072 0C80      00346      movlw  read          ;Load W with the Read command.
```

```

0073 0030      00347      movwf  cmd          ;Transfer W into cmd register.
0074 0930      00348 test2   call   see          ;Read addressed data word from 93C46 device.
0075 0CAA      00349      movlw  tstptrn     ;Load W with the pattern written.
0076 0091      00350      subwf  highb,0     ;Verify the data read against what was
                        00351                        ;written.
0077 0743      00352      btfss  status,zflag ;Same?
0078 0A81      00353      goto  errloop     ;No, error-out.
0079 0CAA      00354      movlw  tstptrn     ;Repeat with the lower byte read.
007A 0092      00355      subwf  lowb,0     ;
007B 0743      00356      btfss  status,zflag ;Same?
007C 0A81      00357      goto  errloop     ;No, error-out.
007D 02B0      00358      incf  cmd, F      ;Yes, both byte correct, increment the 6 bit
                        00359                        ;memory address field.
007E 02FF      00360      decfsz loopcnt, F ;Have we read all 64 locations?
007F 0A74      00361      goto  test2       ;No, read another location.
                        00362
0080 0A80      00363 alloc   goto  alloc      ;Home safe!
                        00364
0081 0A81      00365 errloop goto  errloop     ;Bad news!
                        00366
                        00367      END          ;Thats all folks!

```

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

```

0000 : XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX
0040 : XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX
0080 : XX-----

```

93C46 3 WIRE INTERFACE ROUTINE

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

```

01C0 : -----X

```

All other memory blocks unused.

Program Memory Words Used: 131

Program Memory Words Free: 381

```

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

```

AN530

NOTES:

NOTES:

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.

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. 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.

Trademarks


The Microchip name and logo, the Microchip logo, FilterLab, KEELOQ, microID, MPLAB, PIC, PICmicro, PICMASTER, PICSTART, PRO MATE, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

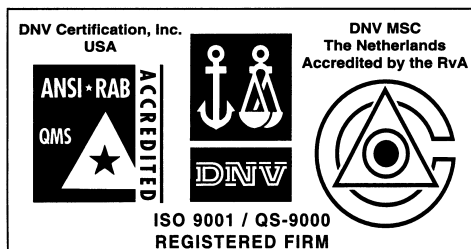
dsPIC, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, microPort, Migratable Memory, MPASM, MPLIB, MPLINK, MPSIM, MXDEV, PICC, PICDEM, PICDEM.net, rPIC, Select Mode and Total Endurance are trademarks of Microchip Technology Incorporated in the U.S.A.

Serialized Quick Turn Programming (SQTP) is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2002, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

 Printed on recycled paper.



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.



MICROCHIP

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: 65-6334-8870 Fax: 65-6334-8850

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 høj 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 - 1er 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

03/01/02