AN618

Modifying PIC16C54A Code for the PIC16C58A

Author: *Stan D'Souza*
Logic Products Division

INTRODUCTION

Code written for a PIC16C54A device can be easily modified and used in a PIC16C58A. Therefore, a PIC16C58A may be used in place of a PIC16C54A. The PIC16C58A and the PIC16C54A are pin-compatible devices. Also, the PIC16C58A has more than enough memory (program and data) to support PIC16C54A code.

Device	Pins	EPROM/ROM	RAM
PIC16C54A	18	512	25
PIC16C58A	18	2048	73

PROGRAM MEMORY

The PIC16C54A has 512 words of EPROM program memory, while the PIC16C58A has 2048 words. Figure 1 and Figure 2 depict the memory organization for each of these devices. For this application, Page 0 of the PIC16C58A On-chip Program Memory will be used in place of the PIC16C54A On-chip Program Memory. However, the Reset vector location will change from 1FFh for the PIC16C54A to 7FFh for the PIC16C58A. (See the discussion on Reset in the next section.)

The PC, or program counter, is used on both devices to access program memory locations. Eight of the nine bits required to access the 512 words of program memory in the PIC16C54A are provided by the PCL file register (Figure 3). The most significant 9th bit is provided from the instruction word during a `GOTO` instruction. A `CALL` instruction, on the other hand, forces a '0' into the most significant 9th bit of the PC. Hence, all subroutines must reside in the top half (00h to FFh) of the program memory.

On the PIC16C58A, bits 0 through 8 of the PC operate the same as bits 0 through 8 of the PIC16C54A PC for `GOTO` and `CALL` instructions (Figure 4). The PIC16C58A PC differs from the PIC16C54A PC in the addition of two most significant bits. These 10th and 11th bits are provided by the PA0 and PA1 values of the STATUS register. The bits signify the program memory page for the PIC16C58A (i.e., 00 = Page 0, 01 = Page 1, etc.).

After a power-up reset, the PA1:PA0 bits are reset to '0'. So, if these two bits are not modified by the PIC16C54A program, the same program will correctly access the Page 0 program memory space on a PIC16C58A.

FIGURE 1: PIC16C54A PROGRAM MEMORY MAP AND STACK

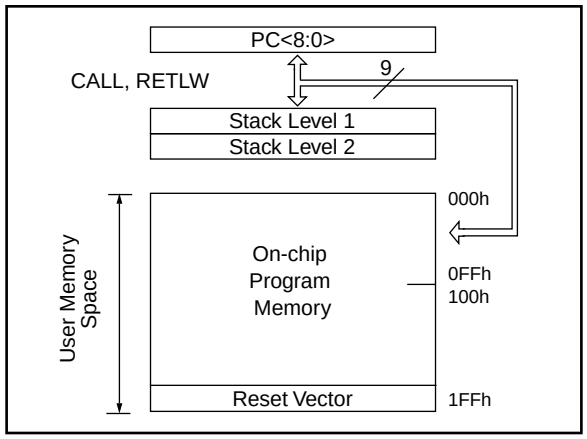


FIGURE 2: PIC16C58A PROGRAM MEMORY MAP AND STACK

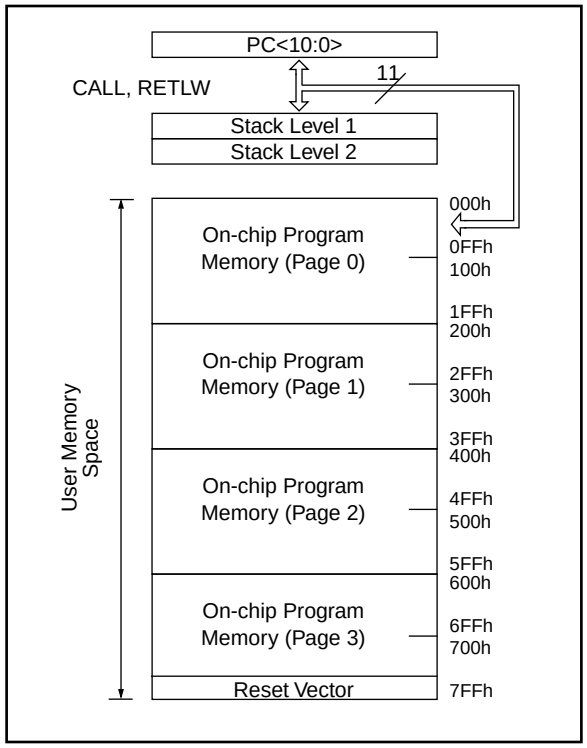


FIGURE 3: LOADING OF PIC16C54A PC FOR BRANCH INSTRUCTIONS

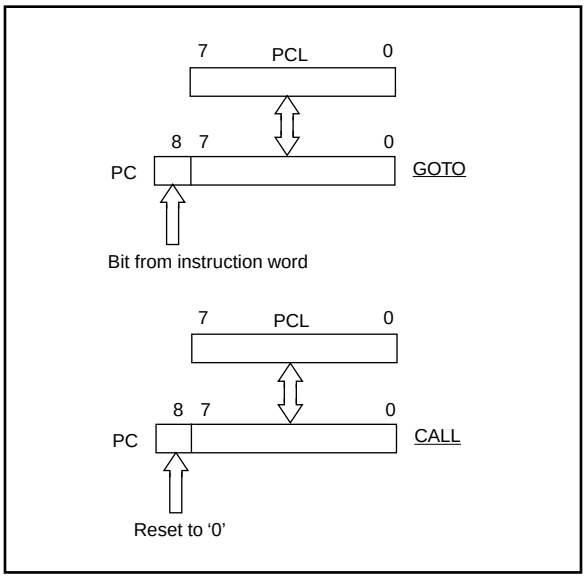
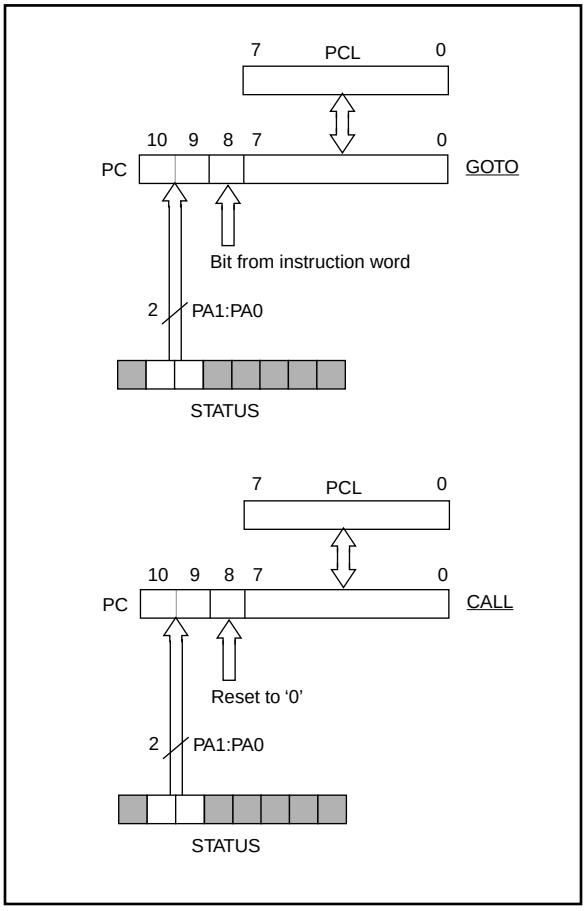


FIGURE 4: LOADING OF PIC16C58A PC FOR BRANCH INSTRUCTIONS



Reset Address

The reset address on the PIC16C54A is at location 1FFh. Normally, a "Goto Start" instruction is placed at this location, where Start is the beginning of the program. If the user has used such an instruction, then in order for the program to operate correctly on a PIC16C58A the "Goto Start" instruction should be moved to location 7FFh, the reset address for the PIC16C58A.

Operation in a Noisy Environment

What happens if the PA1 or PA0 bit value gets corrupted due to noise or some other external event? The PA1:PA0 bits are not directly mapped into PC<11:10>, but the PA1:PA0 bit values are loaded into PC<11:10> during a `CALL` or `GOTO` instruction. So, if the PA1:PA0 bit values get corrupted (ex: PA1:PA0 = 11 instead of 00), the program will go to a program memory page other than 0 when a `CALL` or `GOTO` instruction occurs. The program will then start executing "unimplemented" code space.

Normally the user would program "unimplemented" code space with the default FFFh value (XORLW FFh instruction). The PC would increment sequentially until it hit the last location (7FF) or the Reset address, which is normally recommended to be a "Goto Start" instruction. However, since the value in PA1:PA0 is still corrupted (PA1:PA0 = 11 in our example), the program execution will still be in Page 3. In other words, the program would get locked in an endless loop.

In order to avoid the above situation, each "unimplemented" page should be filled with the instruction "Goto Recover". The Recover subroutines (Three will be required for the three (3) unimplemented pages of the PIC16C58A) should clear the PA1:PA0 bits in the STATUS register and execute a "Goto Start" instruction.

EXAMPLE 1: RECOVER CODE

```
RECOVER  BCF STATUS, PA1
         BCF STATUS, PA0
         GOTO START
```

Please note that operation in a noisy environment is very application-dependent and may not affect the majority of the users. However, due to the presence of extra code space on the PIC16C58A, additional steps may have to be taken as mentioned above. Please check Appendix A for an example of the code implementation.

DATA MEMORY

Figure 5 and Figure 6 depict the data memory registers for the PIC16C54A and the PIC16C58A. Banks 1, 2 and 3, which are not present in the PIC16C54A, are R/W memory RAM locations on the PIC16C58A. These locations are accessed using the FSR<6:5> bits. On the PIC16C54A, these bit values are ignored. However, they are not cleared during a power-up reset on any PIC16C5XA device. Therefore, the FSR<6:5> bits on the PIC16C58A should be cleared at the very start of the program and always maintained in that manner during the course of the program.

Operating in a noisy environment

The FSR<6:5> bits can get corrupted due to noise or some external conditions just like the PA1:PA0 bits of program memory. As part of that recovery, the user should also clear the FSR<6:5> bits. Please see Appendix A for a code example.

SUMMARY

Code transfer from a PIC16C54A to a PIC16C58A may be accomplished easily by remembering the following points:

- The reset address has to be re-located from 1FFh to 7FFh.
- The PA1:PA0 bits of the STATUS register must be kept cleared.
- The FSR<6:5> bits must be kept cleared.

FIGURE 5: PIC16C54A REGISTER FILE MAP

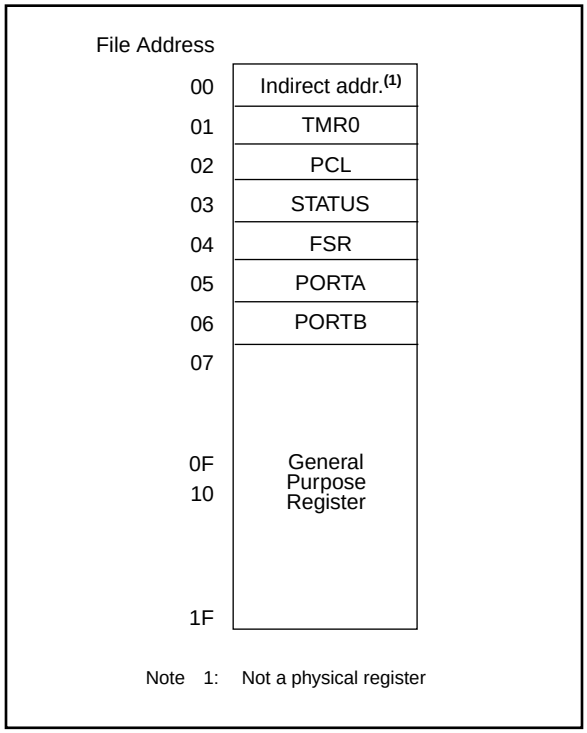
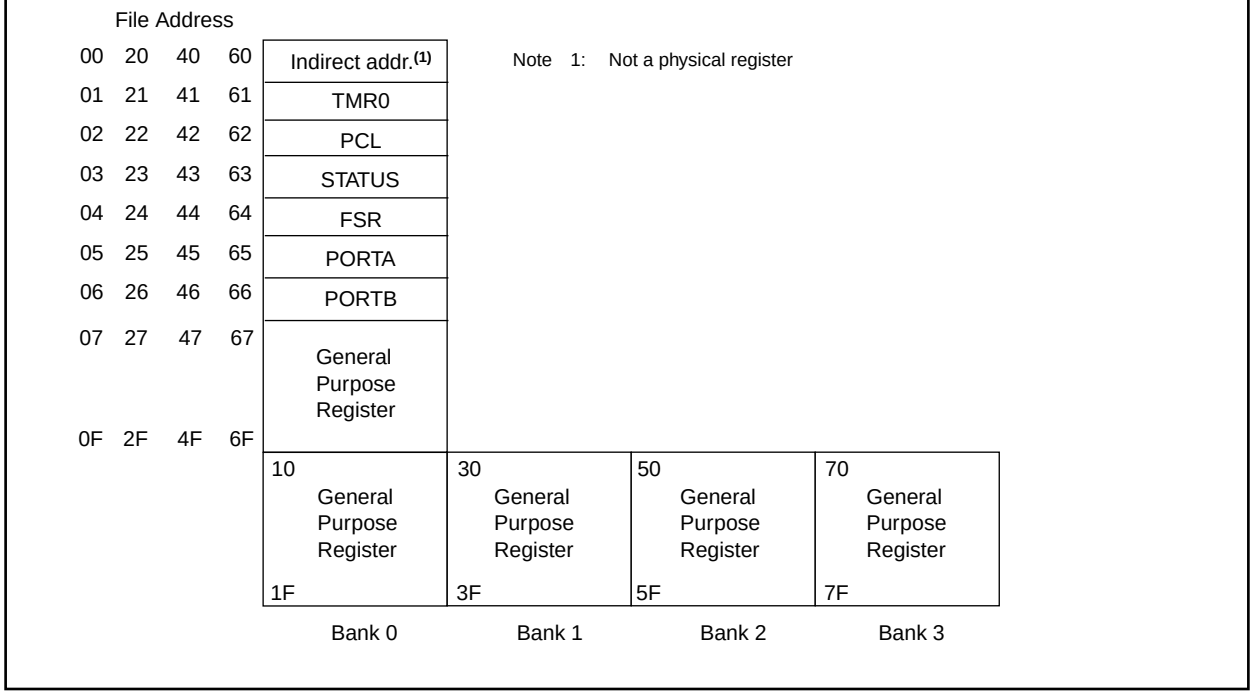


FIGURE 6: PIC16C58A REGISTER FILE MAP



APPENDIX A

```
MPASM 01.21 Released          P54TO58.ASM   8-25-1995  12:33:54         PAGE  1

LOC  OBJECT CODE     LINE SOURCE TEXT
  VALUE

                     00001 ;This program is a code example of a program written for a PIC16C54A
                     00002 ;and modified for use in a PIC16C58A.
                     00003 ;
                     00004 ;In this program the following modifications have been done:
                     00005 ;1. The reset address is relocated from 0x1ff to 0x7ff
                     00006 ;2. The FSR<6,5> bits are initialized to 0 at the start of the program
                     00007 ;3. The Recover subroutine implemented in each “un-implemented”
                     00008 ;   page of the PIC16C58A.
                     00009 ;
                     00010 ;                                              by Stan D’Souza 8/29/95.
                     00011 ;
                     00012 ;
                     00013         list p=16C58A, f=inhx8m
                     00014 ;
                     00015 ;This program written for the PIC16C54A and operates on the PICDEM1
                     00016 ;demo board. Program has been modified to work on a PIC16C58A.
                     00017 ;The program blinks each of the 8 LEDs on PortB twice, then moves to the
                     00018 ;next LED and so on. If anytime the RA1 key is pressed, the direction
                     00019 ;of rotation is changed.
                     00020 ;
                     00021 ;
  0006               00022 PORTB   equ     6
  0005               00023 PORTA   equ     5
  0003               00024 STATUS  equ     3
  0004               00025 FSR     equ     4
  0005               00026 PA0     equ     5
  0006               00027 PA1     equ     6
  0000               00028 W       equ     0
  0001               00029 F       equ     1
  0000               00030 C       equ     0
  001B               00031 pb_buf  equ     0x1b
  001C               00032 count   equ     0x1c
  001D               00033 templ   equ     0x1d
  001E               00034 temph   equ     0x1e
  001F               00035 flag    equ     0x1f
                     00036 #define KeyPressed      flag,7
                     00037 #define KeyReleased     flag,6
                     00038 #define LtoR            flag,5
                     00039 ;
                     00040 ;*****************************************************************
                     00041 ;       org     0x1ff           ;reset vector for PIC16C54A, comment out
                     00042                                 ;when moving to a PIC16C58A
                     00043 ;*****************************************************************
07FF                 00044         org     0x7ff           ;added as PIC16C58A reset vector
07FF 0A10            00045         goto    Start
                     00046 ;*****************************************************************
                     00047 ;The fill command is used to load the unused 1.5K EPROM space on the
                     00048 ;PIC16C58A with a “goto Recover” subroutine. If the program ever
                     00049 ;enters the unused 1.5K space, this command will immediately and
                     00050 ;automatically cause the program execution to go back to the first page
                     00051 ;at address 0x00 to 0x1ff.
0200                 00052         org     0x200
```

```
MPASM 01.21 Released          P54TO58.ASM   8-25-1995  12:33:54         PAGE  2


LOC  OBJECT CODE     LINE SOURCE TEXT
  VALUE

0200                 00053 Recover1
0200 04C3            00054         bcf     STATUS,PA1
0201 04A3            00055         bcf     STATUS,PA0
0202 0064            00056         clrf    FSR
0203 0A10            00057         goto    Start
0204 0A00            00058         FILL    (goto Recover1), (0x3ff - $ + 1)
                     00059         ;Fill all unused locations in page 1 with goto Recover1
                     00060 ;
                     00061 ;
0400                 00062         org     0x400
0400                 00063 Recover2
0400 04C3            00064         bcf     STATUS,PA1
0401 04A3            00065         bcf     STATUS,PA0
0402 0064            00066         clrf    FSR
0403 0A10            00067         goto    Start
0404 0A00            00068         FILL    (goto Recover2), (0x5ff - $ + 1)
                     00069         ;Fill all unused locations in page 2 with goto Recover2
                     00070 ;
                     00071 ;
                     00072 ;
0600                 00073         org     0x600
0600                 00074 Recover3
0600 04C3            00075         bcf     STATUS,PA1
0601 04A3            00076         bcf     STATUS,PA0
0602 0064            00077         clrf    FSR
0603 0A10            00078         goto    Start
0604 0A00            00079         FILL    (goto Recover3), (0x7fe - $ + 1)
                     00080         ;Fill all unused locations in page 3 with goto Recover3
                     00081 ;
                     00082 ;
0010                 00083         org     0x10
0010                 00084 Start
                     00085 ;*****************************************************************
0010 0064            00086         clrf    FSR             ;initialize the FSR register
                     00087                                 ;added for the PIC16C58A compatibility
                     00088 ;*****************************************************************
0011 0040            00089         clrw
0012 0026            00090         movwf   PORTB           ;set Port B as output and low
0013 0006            00091         tris    PORTB           ;       /
0014 007B            00092         clrf    pb_buf          ;clear buffer
0015 051B            00093         bsf     pb_buf,0        ;set up the first blink
0016 007F            00094         clrf    flag            ;clr flags
0017 05DF            00095         bsf     KeyReleased     ;       /
0018                 00096 Repeat
0018 021B            00097         movf    pb_buf,W
0019 0026            00098         movwf   PORTB
001A 092F            00099         call    delay500        ;delay for 500mS
001B 0066            00100         clrf    PORTB
001C 092F            00101         call    delay500        ;
001D 06FF            00102         btfsc   KeyPressed      ;if no key then skip
```

```
MPASM 01.21 Released          P54TO58.ASM   8-25-1995  12:33:54         PAGE  3


LOC  OBJECT CODE     LINE SOURCE TEXT
  VALUE

001E 0A46           00103         goto    ChangeDirection ;else change direction
001F 021B           00104         movf    pb_buf,W
0020 0026           00105         movwf   PORTB
0021 092F           00106         call    delay500        ;delay for 500mS
0022 0066           00107         clrf    PORTB
0023 092F           00108         call    delay500        ;
0024 06FF           00109         btfsc   KeyPressed      ;if no key then skip
0025 0A46           00110         goto    ChangeDirection
0026                00111 Rotate
0026 0403           00112         bcf     STATUS,C        ;clr carry
0027                00113 RotAgain
0027 07BF           00114         btfss   LtoR            ;left to right?
0028 0A2D           00115         goto    RotateLeft      ;rotate Port B
0029 033B           00116         rrf     pb_buf,F        ;rotate Port B
002A                00117 Done
002A 0703           00118         btfss   STATUS,C        ;carry over?
002B 0A18           00119         goto    Repeat          ;no then do again
002C 0A27           00120         goto    RotAgain        ;rotate again
002D                00121 RotateLeft
002D 037B           00122         rlf     pb_buf,F        ;rotate left
002E 0A2A           00123         goto    Done            ;check
                    00124 ;
002F                00125 delay500
002F 0C19           00126         movlw   .25             ;reload count
0030 003C           00127         movwf   count           ;       /
0031                00128 dly500lp
0031 06DF           00129         btfsc   KeyReleased     ;key released?
0032 0938           00130         call    CheckReleased   ;no then check?
0033 093E           00131         call    CheckPressed    ;else check if pressed?
0034 094E           00132         call    delay20         ;delay for 20mS
0035 02FC           00133         decfsz  count           ;see if delay over
0036 0A31           00134         goto    dly500lp        ;no then loop
0037 0800           00135         retlw   0               ;return
                    00136 ;
0038                00137 CheckReleased
0038 0625           00138         btfsc   PORTA,1         ;RA1 low?
0039 0A3B           00139         goto    RelAgain        ;no then check again
003A 0800           00140         retlw   0               ;go back
003B                00141 RelAgain
003B 0625           00142         btfsc   PORTA,1         ;RA1 low?
003C 05DF           00143         bsf     KeyReleased     ;no then key released
003D 0800           00144         retlw   0
                    00145 ;
003E                00146 CheckPressed
003E 06FF           00147         btfsc   KeyPressed      ;flag already set?
003F 0800           00148         retlw   0               ;yes then return
0040 0725           00149         btfss   PORTA,1         ;see if key low?
0041 0A43           00150         goto    PressAgain      ;check again
0042 0800           00151         retlw   0               ;else go back
0043                00152 PressAgain
0043 0725           00153         btfss   PORTA,1         ;see if low
0044 05FF           00154         bsf     KeyPressed      ;yes then set flag
```

```
MPASM 01.21 Released          P54TO58.ASM   8-25-1995  12:33:54         PAGE  4


LOC  OBJECT CODE     LINE SOURCE TEXT
  VALUE

0045 0800           00155         retlw   0
                    00156 ;
0046                00157 ChangeDirection
0046 04FF           00158         bcf     KeyPressed      ;key serviced
0047 04DF           00159         bcf     KeyReleased     ;see if key released
0048 07BF           00160         btfss   LtoR            ;check if Left to Rt.
0049 0A4C           00161         goto    TurnLeft        ;make it go left
004A 04BF           00162         bcf     LtoR
004B 0A26           00163         goto    Rotate
004C                00164 TurnLeft
004C 05BF           00165         bsf     LtoR
004D 0A26           00166         goto    Rotate
                    00167
                    00168 ;
                    00169 ;Delay loop for 20 mS
004E                00170 delay20
004E 0C14           00171         movlw   .20             ;load temph
004F 003E           00172         movwf   temph
0050                00173 dly4
0050 0CC8           00174         movlw   .200            ;load templ
0051 003D           00175         movwf   templ
0052                00176 dly
0052 0000           00177         nop
0053 0000           00178         nop
0054 02FD           00179         decfsz  templ
0055 0A52           00180         goto    dly
0056 02FE           00181         decfsz  temph
0057 0A50           00182         goto    dly4
0058 0800           00183         retlw   0
                    00184 ;
                    00185 ;
                    00186 ;
                    00187         end

All other memory blocks unused.


Errors   :    0
Warnings :    0
Messages :    0
```

Worldwide Sales & Service

AMERICAS

Corporate Office
Microchip Technology Inc.
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 602-786-7200 Fax: 602-786-7277
Technical Support: 602 786-7627
Web: http://www.microchip.com

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

Boston
Microchip Technology Inc.
5 Mount Royal Avenue
Marlborough, MA 01752
Tel: 508-480-9990 Fax: 508-480-8575

Chicago
Microchip Technology Inc.
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 708-285-0071 Fax: 708-285-0075

Dallas
Microchip Technology Inc.
14651 Dallas Parkway, Suite 816
Dallas, TX 75240-8809
Tel: 972-991-7177 Fax: 972-991-8588

Dayton
Microchip Technology Inc.
Suite 150
Two Prestige Place
Miamisburg, OH 45342
Tel: 937-291-1654 Fax: 937-291-9175

Los Angeles
Microchip Technology Inc.
18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 714-263-1888 Fax: 714-263-1338

New York
Microchip Technology Inc.
150 Motor Parkway, Suite 416
Hauppauge, NY 11788
Tel: 516-273-5305 Fax: 516-273-5335

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

Toronto
Microchip Technology Inc.
5925 Airport Road, Suite 200
Mississauga, Ontario L4V 1W1, Canada
Tel: 905-405-6279 Fax: 905-405-6253

ASIA/PACIFIC

Hong Kong
Microchip Asia Pacific
RM 3801B, Tower Two
Metroplaza
223 Hing Fong Road
Kwai Fong, N.T. Hong Kong
Tel: 852-2-401-1200 Fax: 852-2-401-3431

India
Microchip Technology India
No. 6, Legacy, Convent Road
Bangalore 560 025 India
Tel: 91-80-299-4036 Fax: 91-80-559-9840

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

Shanghai
Microchip Technology
Unit 406 Shanghai Golden Bridge Bldg.
2077 Yan'an Road West, Hongiao District
Shanghai, Peoples Republic of China
Tel: 86-21-6275-5700
Fax: 011-86 21-6275-5060

Singapore
Microchip Technology
Taiwan Singapore Branch
200 Middle Road
#10-03 Prime Centre
Singapore 188980
Tel: 65-334-8870 Fax: 65-334-8850

Taiwan, R.O.C
Microchip Technology Taiwan
10F-1C 207
Tung Hua North Road
Taipei, Taiwan, ROC
Tel: 886 2-717-7175 Fax: 886-2-545-0139

EUROPE

United Kingdom
Arizona Microchip Technology Ltd.
Unit 6, The Courtyard
Meadow Bank, Furlong Road
Bourne End, Buckinghamshire SL8 5AJ
Tel: 44-1628-851077 Fax: 44-1628-850259

France
Arizona Microchip Technology SARL
Zone Industrielle de la Bonde
2 Rue du Buisson aux Fraises
91300 Massy - France
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79

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

Italy
Arizona Microchip Technology SRL
Centro Direzionale Colleone
Palazzo Taurus 1 V. Le Colleoni 1
20041 Agrate Brianza
Milan Italy
Tel: 39-39-6899939 Fax: 39-39-6899883

JAPAN
Microchip Technology Intl. Inc.
Benex S-1 6F
3-18-20, Shin Yokohama
Kohoku-Ku, Yokohama
Kanagawa 222 Japan
Tel: 81-4-5471- 6166 Fax: 81-4-5471-6122

12/6/96

MICROCHIP

Printed on recycled paper.