AN586

Macros for Page and Bank Switching

Author:	*Mark Palmer* *Microchip Technology Inc.*
Contributions:	*Mike Morse* *Sr. Field Applications Engineer (Dallas)*

INTRODUCTION

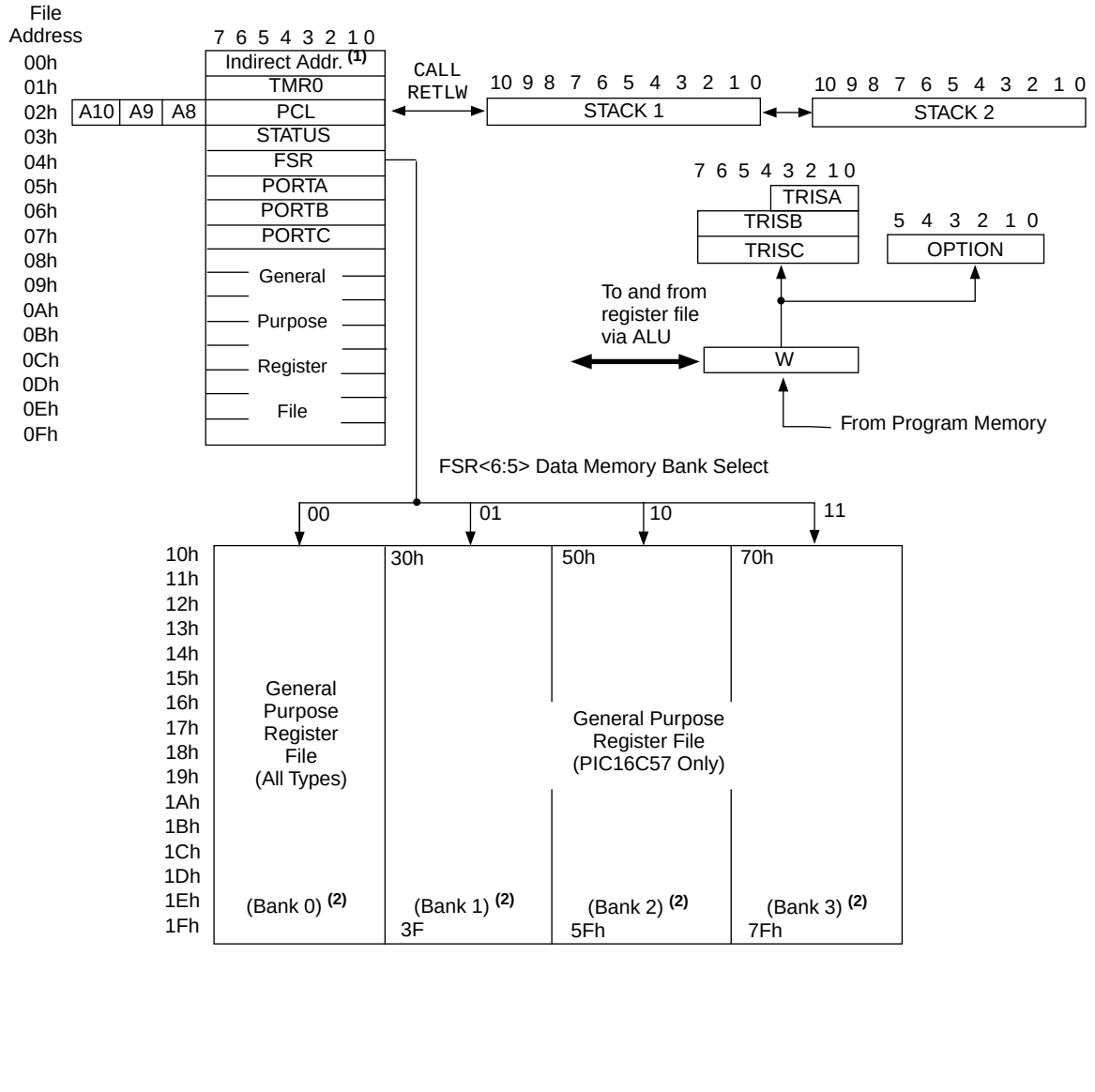
This application note discusses the use of the MPASM assembler's conditional assembly to automatically switch between program memory pages or to set the data memory banks. These macros, along with the long call technique (Application Note AN581), ease the development of software. Though the use of these macros can simplify program memory paging and data memory banking with minimal software overhead, the use of these macros without thought can cause unnecessary (duplicate) instructions to be used, by setting page or bank bits unnecessarily.

The PIC16C5X family of devices has an architecture in which program memory has up to four pages of program memory (512 words / page) and four banks of data memory (16 bytes / bank). Two bits in the STATUS register, PA1:PA0, are used to manage the program memory pages. Two bits of the FSR register, bits 6 and 5, manage the data memory banks. We will call the FSR<5> bit RP0 and the FSR<6> bit RP1 (for Register Page 0 and 1). The naming of these bits RP1 and RP0 should not be confused with the similarly named bits in the PIC16CXX family (PIC16C64, PIC16C71, etc.). The RP bits for the PIC16CXX family are found in the STATUS register, as opposed to the FSR register for the PIC16C5X family. The use of these macros can be modified to support the PIC16CXX family.

The program memory organization is shown in Figure 1 and the data memory organization is shown in Figure 2. To use the macros for the data memory, the data memory locations must be EQUated for the absolute address, and not the relative address in the bank. The relative address is the lower 5-bits of the data memory address.

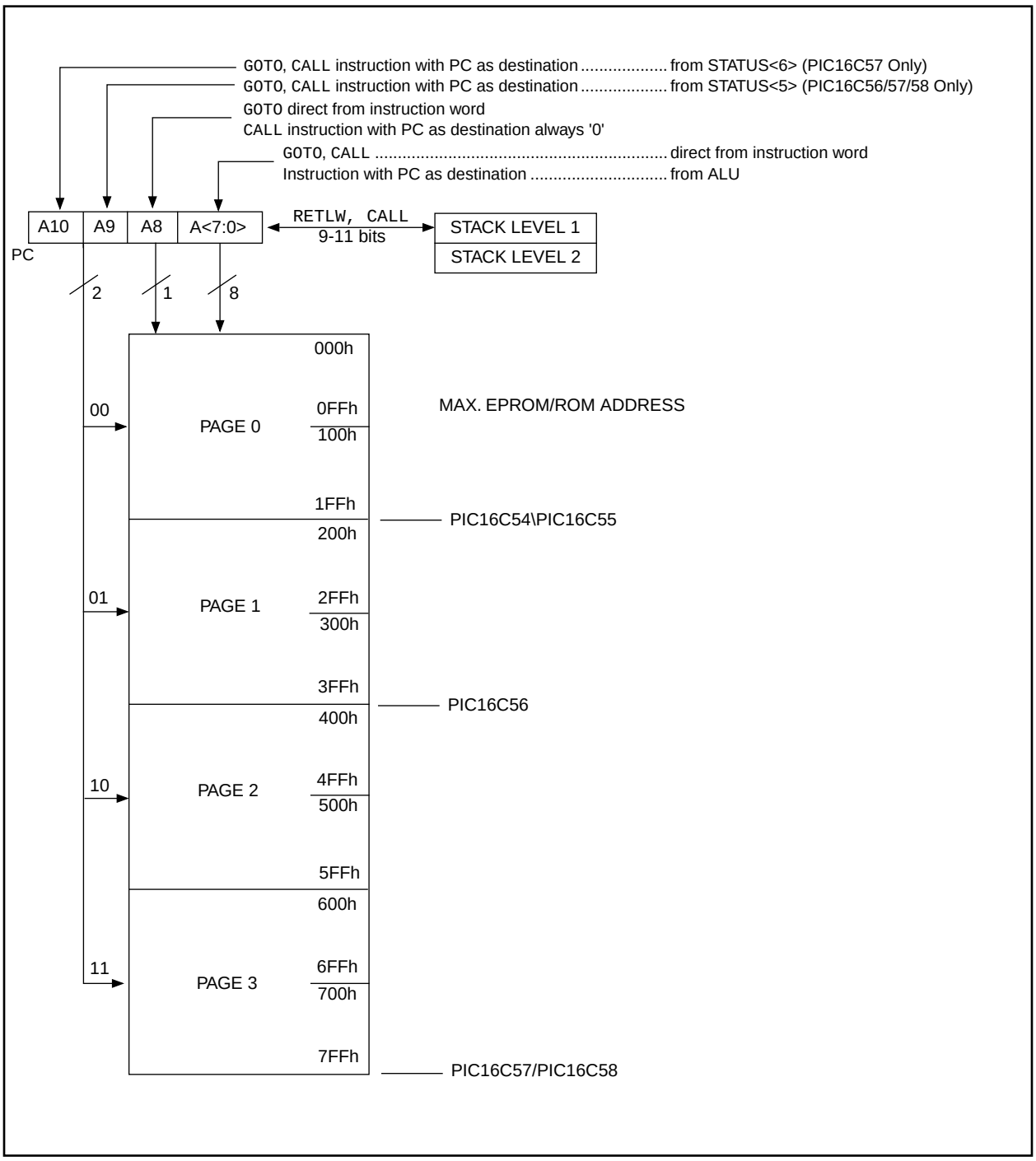
When the address of the data memory has the MSb (bit4) of the direct address cleared, or FSR<4> cleared (for indirect addressing), the address 0h through 0Fh is accessed. That is, when accessing addresses 0h through 0Fh the bank selection (FSR<6:5>) bits are ignored. This means that data memory addresses `'xxx0 xxxx'b` access the data memory address 0xh (x is anywhere from 0 - F).

FIGURE 1: PROGRAM MEMORY ORGANIZATION



Note 1: Not a physically implemented register.
2: Bank 0 is available on all microcontrollers while Bank 1, Bank 2, and Bank 3 are only available on the PIC16C57 and PIC16C58.

FIGURE 2: DATA MEMORY MAP



The use of MPASM's conditional assembly, allows the selection of source code to be assembled based on the address of the symbol / label. The macros supplied are show in Table 1.

They can be grouped into three categories:

1. Configuring of the program memory pages.
2. Configuring of the data memory banks.
3. Other.

TABLE 1: MACROS

Program Calling Paging	Operands	Operation
CALLM	address	Sets page bits, then CALLs the specified routine
GOTOM	address	Sets page bits, then GOTOs the specified address
PAGE_MAC	address	Sets the specified page bits
Data Memory Banking		
ADDWF_MAC	Reg, dest	Sets Bank bits, then executes the ADDWF
ANDWF_MAC	Reg, dest	Sets Bank bits, then executes the ANDWF
BCF_MAC	Reg, bit	Sets Bank bits, then executes the BCF
BSF_MAC	Reg, bit	Sets Bank bits, then executes the BSF
BTFSC_MAC	Reg, bit	Sets Bank bits, then executes the BTFSC
BTFSS_MAC	Reg, bit	Sets Bank bits, then executes the BTFSS
CLRF_MAC	Reg	Sets Bank bits, then executes the CLRF
COMF_MAC	Reg, dest	Sets Bank bits, then executes the COMF
DECF_MAC	Reg, dest	Sets Bank bits, then executes the DECF
DECFSZ_MAC	Reg, dest	Sets Bank bits, then executes the DECFSZ
INCF_MAC	Reg, dest	Sets Bank bits, then executes the INCF
INCFSZ_MAC	Reg, dest	Sets Bank bits, then executes the INCFSZ
IORWF_MAC	Reg, dest	Sets Bank bits, then executes the IORWF
MOVF_MAC	Reg, dest	Sets Bank bits, then executes the MOVF
MOVWF_MAC	Reg	Sets Bank bits, then executes the MOVWF
RLF_MAC	Reg, dest	Sets Bank bits, then executes the RLF
RRF_MAC	Reg, dest	Sets Bank bits, then executes the RRF
SUBWF_MAC	Reg, dest	Sets Bank bits, then executes the SUBWF
SWAPF_MAC	Reg, dest	Sets Bank bits, then executes the SWAPF
XORWF_MAC	Reg, dest	Sets Bank bits, then executes the XORWF
BANK_MAC	Reg	Sets the specified Bank bits
Other		
SAVE_W_STATUS	-	Saves the W and STATUS registers
RESTORE_W_STATUS	-	Restores the W and STATUS registers

These macros (Appendix A) ease the development of programs, but care should be taken in their use so that redundant instructions are not caused. An example of this (Example 1) is if you wanted to do the operations, INCF and BTFSS, on data memory location CNTR (in Bank 3) and the FSR was pointing to some other bank. The use of the macros for both operations would cause six program memory locations to be assembled, while with some thought only four words are needed (Example 2).

CONCLUSION

The use of these macros simplify program development by managing the memory resources of the PIC16C5X device. If the application program becomes too large for the device's program memory, it is recommended to study the listing file for any unnecessary code due to non-optimum usage of these macros. The MAC_TST.ASM file, is supplied to show how these macros work in a program.

EXAMPLE 1: GENERATION OF UNNECESSARY CODE

```
INCF_MAC     CNTR, F     —>     BSF     FSR, 5
                                BSF     FSR, 6
                                INCF    CNTR, F
BTFSS_MAC    CNTR, 5     —>     BSF     FSR, 5     ; Unnecessary, already in bank
                                BSF     FSR, 6     ; Unnecessary, already in bank
                                BTFSS   CNTR, 5
```

EXAMPLE 2: GENERATION OF OPTIMUM CODE

```
INCF_MAC     CNTR, F     —>     BSF     FSR, 5
                                BSF     FSR, 6
                                INCF    CNTR, F
BTFSS        CNTR, 5     —>     BTFSS   CNTR, 5
```

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

APPENDIX A: MACRO FILE

```
MPASM 01.40 Released         LCALL_5X.ASM   1-16-1997  17:13:33         PAGE  1


LOC  OBJECT CODE     LINE SOURCE TEXT
  VALUE

                     00001         LIST    P = 16C57, n = 66
                     00002         ERRORLEVEL  -302, -306
                     00003 ;
                     00004 ;               Program = LCALL_5x.asm
                     00005 ;               Revision Date:  5-07-94
                     00006 ;                               1-15-97    Compatibility with MPASMWIN 1.40
                     00007 ;
                     00008 ;
  00000000           00009 P1_TOP          EQU     0x0000
  000001FF           00010 P1_BOTTOM       EQU     0x01FF
  00000200           00011 P2_TOP          EQU     0x0200
  000003FF           00012 P2_BOTTOM       EQU     0x03FF
  00000400           00013 P3_TOP          EQU     0x0400
  000005FF           00014 P3_BOTTOM       EQU     0x05FF
  00000600           00015 P4_TOP          EQU     0x0600
  000007FF           00016 P4_BOTTOM       EQU     0x07FF
  000007FF           00017 RESET_V         EQU     0x07FF
                     00018 ;
  00000003           00019 STATUS          EQU     0x03        ; Status Register
  00000005           00020 PA0             EQU     0x05        ; Program Memory Page Address bit 0
  00000006           00021 PA1             EQU     0x06        ; Program Memory Page Address bit 1
                     00022 ;
  00000004           00023 FSR             EQU     0x04        ; FSR Register
  00000005           00024 RP0             EQU     0x05        ; Direct Addressing Register Bank bit 0
  00000006           00025 RP1             EQU     0x06        ; Direct Addressing Register Bank bit 1
                     00026 ;
  00000010           00027 BANK0_T         EQU     0x10
  00000030           00028 BANK1_T         EQU     0x30
  00000050           00029 BANK2_T         EQU     0x50
  00000070           00030 BANK3_T         EQU     0x70
                     00031 ;
0000                 00032                 org     P1_TOP
                     00033 ;
0000 0A07            00034 P1_CALL_1_V     GOTO    P1_CALL_1
0001 0A0A            00035 P1_CALL_2_V     GOTO    P1_CALL_2
```

```
0002 0A0D           00036 P1_CALL_3_V     GOTO    P1_CALL_3
0003 0A1F           00037 P1_CALL_4_V     GOTO    P1_CALL_4
0004 0A64           00038 P1_CALL_5_V     GOTO    P1_CALL_5
                    00039 ;
                    00040 ;
  00000010          00041 D_address       EQU     0x10
  00000001          00042 F               EQU     1
  00000000          00043 W               EQU     0
                    00044 ;
                    00045      include    <AUTO_PG.MAC>
                    00408      list
                    00409
                    00046 ;
0005 090D           00047 START           CALL    P1_CALL_3
                    00048 ;
0006 0A06           00049 LOOP            GOTO    LOOP
                    00050 ;
                    00051 ;
0007                00052 P1_CALL_1
                    00053 ;
                    00054     if ( (P1_CALL_1_V & 0x0600) != (P1_CALL_1 & 0x0600) )
                    00055        MESSG   “ERROR - User Defined: CALL VECTOR and CALL routine NOT in same page”
                    00056     endif
                    00057 ;
0007 0000           00058                 NOP
0008 0000           00059                 NOP
0009 0800           00060 P1_CALL_1_END   RETURN
                    00061 ;
                    00062     if ( (P1_CALL_1 & 0x0600) != (P1_CALL_1_END & 0x0600) )
                    00063        MESSG   “Warning - User Defined: Call routine crosses page boundry”
                    00064     endif
                    00065 ;
                    00066
                    00067
000A                00068 P1_CALL_2
                    00069 ;
                    00070     if ( (P1_CALL_2_V & 0x0600) != (P1_CALL_2 & 0x0600) )
                    00071        MESSG   “ERROR - User Defined: CALL VECTOR and CALL routine NOT in same page”
                    00072     endif
                    00073 ;
000A 0000           00074                 NOP
000B 0000           00075                 NOP
000C 0800           00076 P1_CALL_2_END   RETURN
                    00077 ;
                    00078     if ( (P1_CALL_2 & 0x0600) != (P1_CALL_2_END & 0x0600) )
                    00079        MESSG   “Warning - User Defined: Call routine crosses page boundry”
                    00080     endif
```

```
                    00081 ;
                    00082
                    00083
000D                00084 P1_CALL_3
                    00085 ;
                    00086     if ( (P1_CALL_3_V & 0x0600) != (P1_CALL_3 & 0x0600) )
                    00087        MESSG   “ERROR - User Defined: CALL VECTOR and CALL routine NOT in same page”
                    00088     endif
                    00089 ;
000D 0000           00090                 NOP
000E 0000           00091                 NOP
000F 0800           00092 P1_CALL_3_END   RETURN
                    00093 ;
                    00094     if ( (P1_CALL_3 & 0x0600) != (P1_CALL_3_END & 0x0600) )
                    00095        MESSG   “Warning - User Defined: Call routine crosses page boundry”
                    00096     endif
                    00097 ;
                    00098
                    00099     CALLM       P2_CALL_3_V
                        M ;
                        M     if ( ( P2_CALL_3_V & 0x0200 ) == 0x0200 )
0010 05A3               M                 BSF     STATUS, PA0       ; Set PA0 for Program Memory Page
                        M     else
                        M                 BCF     STATUS, PA0       ; Clear PA0 for Program Memory Page
                        M     endif
                        M ;
                        M     if ( ( P2_CALL_3_V & 0x0400 ) == 0x0400 )
                        M                 BSF     STATUS, PA1       ; Set PA1 for Program Memory Page
                        M     else
0011 04C3               M                 BCF     STATUS, PA1       ; Clear PA1 for Program Memory Page
                        M     endif
                        M ;
0012 0905               M                 CALL    P2_CALL_3_V
0013 0000           00100                 nop
0014 0000           00101                 nop
                    00102 ;
                    00103             CALLM       P4_CALL_2_V
                        M ;
                        M      if ( ( P4_CALL_2_V & 0x0200 ) == 0x0200 )
0015 05A3               M                 BSF     STATUS, PA0       ; Set PA0 for Program Memory Page
                        M      else
                        M                 BCF     STATUS, PA0       ; Clear PA0 for Program Memory Page
                        M      endif
                        M ;
                        M     if ( ( P4_CALL_2_V & 0x0400 ) == 0x0400 )
0016 05C3               M                 BSF     STATUS, PA1       ; Set PA1 for Program Memory Page
                        M      else
```

```
                        M                BCF     STATUS, PA1       ; Clear PA1 for Program Memory Page
                        M      endif
                        M ;
0017 0901               M                CALL    P4_CALL_2_V
0018 0000            00104               nop
0019 0000            00105               nop
                     00106 ;
                     00107            CALLM       P3_CALL_1_V
                        M ;
                        M      if ( ( P3_CALL_1_V & 0x0200 ) == 0x0200 )
                        M                BSF     STATUS, PA0       ; Set PA0 for Program Memory Page
                        M      else
001A 04A3               M                BCF     STATUS, PA0       ; Clear PA0 for Program Memory Page
                        M      endif
                        M ;
                        M      if ( ( P3_CALL_1_V & 0x0400 ) == 0x0400 )
001B 05C3               M                BSF     STATUS, PA1       ; Set PA1 for Program Memory Page
                        M      else
                        M                BCF     STATUS, PA1       ; Clear PA1 for Program Memory Page
                        M      endif
                        M ;
001C 0900               M                CALL    P3_CALL_1_V
001D 0000            00108               nop
001E 0000            00109               nop
                     00110
                     00111 ;
001F                 00112 P1_CALL_4
                     00113 ;
                     00114     if ( (P1_CALL_4_V & 0x0600) != (P1_CALL_4 & 0x0600) )
                     00115        MESSG   “ERROR - User Defined: CALL VECTOR and CALL routine NOT in same page”
                     00116     endif
                     00117 ;
001F 0000            00118                NOP
0020 0000            00119                NOP
0021 0800            00120 P1_CALL_4_END  RETURN
                     00121 ;
                     00122     if ( (P1_CALL_4 & 0x0600) != (P1_CALL_4_END & 0x0600) )
                     00123        MESSG   “Warning - User Defined: Call routine crosses page boundry”
                     00124     endif
                     00125 ;
0022 0000            00126               nop
0023 0000            00127               nop
                     00128
                     00129               ADDWF_MAC      D_address, F
                        M ;
                        M      if ( ( D_address & 0x020 ) == 0x020 )
                        M                BSF     FSR, RP0          ; Set RP0 for Data Memory Page
```

```
                        M      else
0024 04A4               M                 BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                        M      endif
                        M ;
                        M      if ( ( D_address & 0x040 ) == 0x040 )
                        M                 BSF     FSR, RP1          ; Set RP1 for Data Memory Bank
                        M      else
0025 04C4               M                 BCF     FSR, RP1          ; Clear RP1 for Data Memory Bank
                        M      endif
0026 01F0               M                 ADDWF    D_address, F
                     00130                ANDWF_MAC      D_address, F
                        M ;
                        M      if ( ( D_address & 0x020 ) == 0x020 )
                        M                 BSF     FSR, RP0          ; Set RP0 for Data Memory Bank
                        M      else
0027 04A4               M                 BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                        M      endif
                        M ;
                        M      if ( ( D_address & 0x040 ) == 0x040 )
                        M                 BSF     FSR, RP1          ; Set RP1 for Data Memory Bank
                        M      else
0028 04C4               M                 BCF     FSR, RP1          ; Clear RP1 for Data Memory Bank
                        M      endif
0029 0170               M                 ANDWF    D_address, F
                     00131                BCF_MAC        D_address, 7
                        M ;
                        M      if ( ( D_address & 0x020 ) == 0x020 )
                        M                 BSF     FSR, RP0          ; Set RP0 for Data Memory Bank
                        M      else
002A 04A4               M                 BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                        M      endif
                        M ;
                        M      if ( ( D_address & 0x040 ) == 0x040 )
                        M                 BSF     FSR, RP1          ; Set RP1 for Data Memory Bank
                        M      else
002B 04C4               M                 BCF     FSR, RP1          ; Clear RP1 for Data Memory Bank
                        M      endif
002C 04F0               M                 BCF     D_address, 7
                     00132                BSF_MAC        D_address, 7
                        M ;
                        M      if ( ( D_address & 0x020 ) == 0x020 )
                        M                 BSF     FSR, RP0          ; Set RP0 for Data Memory Bank
                        M      else
002D 04A4               M                 BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                        M      endif
                        M ;
                        M      if ( ( D_address & 0x040 ) == 0x040 )
```

```
                        M                BSF     FSR, RP1          ; Set RP1 for Data Memory Bank
                        M      else
002E 04C4               M                BCF     FSR, RP1          ; Clear RP1 for Data Memory Bank
                        M      endif
002F 05F0               M                BSF     D_address, 7
                    00133                BTFSC_MAC       D_address, 7
                        M ;
                        M      if ( ( D_address & 0x020 ) == 0x020 )
                        M                BSF     FSR, RP0          ; Set RP0 for Data Memory Bank
                        M      else
0030 04A4               M                BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                        M      endif
                        M ;
                        M      if ( ( D_address & 0x040 ) == 0x040 )
                        M                BSF     FSR, RP1          ; Set RP1 for Data Memory Bank
                        M      else
0031 04C4               M                BCF     FSR, RP1          ; Clear RP1 for Data Memory Bank
                        M      endif
0032 06F0               M                BTFSC    D_address, 7
                    00134                BTFSS_MAC       D_address, 7
                        M ;
                        M      if ( ( D_address & 0x020 ) == 0x020 )
                        M                BSF     FSR, RP0          ; Set RP0 for Data Memory Bank
                        M      else
0033 04A4               M                BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                        M      endif
                        M ;
                        M      if ( ( D_address & 0x040 ) == 0x040 )
                        M                BSF     FSR, RP1          ; Set RP1 for Data Memory Bank
                        M      else
0034 04C4               M                BCF     FSR, RP1          ; Clear RP1 for Data Memory Bank
                        M      endif
0035 07F0               M                BTFSS    D_address, 7
                    00135                CLRF_MAC        D_address
                        M ;
                        M      if ( ( D_address & 0x020 ) == 0x020 )
                        M                BSF     FSR, RP0          ; Set RP0 for Data Memory Bank
                        M      else
0036 04A4               M                BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                        M      endif
                        M ;
                        M      if ( ( D_address & 0x040 ) == 0x040 )
                        M                BSF     FSR, RP1          ; Set RP1 for Data Memory Bank
                        M      else
0037 04C4               M                BCF     FSR, RP1          ; Clear RP1 for Data Memory Bank
                        M      endif
0038 0070               M                CLRF    D_address
```

```
                    00136                COMF_MAC       D_address, F
                        M ;
                        M      if ( ( D_address & 0x020 ) == 0x020 )
                        M                BSF     FSR, RP0          ; Set RP0 for Data Memory Bank
                        M      else
0039 04A4               M                BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                        M      endif
                        M ;
                        M      if ( ( D_address & 0x040 ) == 0x040 )
                        M                BSF     FSR, RP1          ; Set RP1 for Data Memory Bank
                        M      else
003A 04C4               M                BCF     FSR, RP1          ; Clear RP1 for Data Memory Bank
                        M      endif
003B 0270               M                COMF    D_address, F
                    00137                DECF_MAC       D_address, F
                        M ;
                        M      if ( ( D_address & 0x020 ) == 0x020 )
                        M                BSF     FSR, RP0          ; Set RP0 for Data Memory Bank
                        M      else
003C 04A4               M                BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                        M      endif
                        M ;
                        M      if ( ( D_address & 0x040 ) == 0x040 )
                        M                BSF     FSR, RP1          ; Set RP1 for Data Memory Bank
                        M      else
003D 04C4               M                BCF     FSR, RP1          ; Clear RP1 for Data Memory Bank
                        M      endif
003E 00F0               M                DECF    D_address, F
                    00138                DECFSZ_MAC     D_address, F
                        M ;
                        M      if ( ( D_address & 0x020 ) == 0x020 )
                        M                BSF     FSR, RP0          ; Set RP0 for Data Memory Bank
                        M      else
003F 04A4               M                BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                        M      endif
                        M ;
                        M      if ( ( D_address & 0x040 ) == 0x040 )
                        M                BSF     FSR, RP1          ; Set RP1 for Data Memory Bank
                        M      else
0040 04C4               M                BCF     FSR, RP1          ; Clear RP1 for Data Memory Bank
                        M      endif
0041 02F0               M                DECFSZ  D_address, F
                    00139                INCF_MAC       D_address, F
                        M ;
                        M      if ( ( D_address & 0x020 ) == 0x020 )
                        M                BSF     FSR, RP0          ; Set RP0 for Data Memory Bank
                        M      else
```

```
0042 04A4               M               BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                        M       endif
                        M ;
                        M       if ( ( D_address & 0x040 ) == 0x040 )
                        M               BSF     FSR, RP1          ; Set RP1 for Data Memory Bank
                        M       else
0043 04C4               M               BCF     FSR, RP1          ; Clear RP1 for Data Memory Bank
                        M       endif
0044 02B0               M               INCF    D_address, F
                     00140              INCFSZ_MAC     D_address, F
                        M ;
                        M       if ( ( D_address & 0x020 ) == 0x020 )
                        M               BSF     FSR, RP0          ; Set RP0 for Data Memory Bank
                        M       else
0045 04A4               M               BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                        M       endif
                        M ;
                        M       if ( ( D_address & 0x040 ) == 0x040 )
                        M               BSF     FSR, RP1          ; Set RP1 for Data Memory Bank
                        M       else
0046 04C4               M               BCF     FSR, RP1          ; Clear RP1 for Data Memory Bank
                        M       endif
0047 03F0               M               INCFSZ  D_address, F
                     00141              IORWF_MAC      D_address, F
                        M ;
                        M       if ( ( D_address & 0x020 ) == 0x020 )
                        M               BSF     FSR, RP0          ; Set RP0 for Data Memory Bank
                        M       else
0048 04A4               M               BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                        M       endif
                        M ;
                        M       if ( ( D_address & 0x040 ) == 0x040 )
                        M               BSF     FSR, RP1          ; Set RP1 for Data Memory Bank
                        M       else
0049 04C4               M               BCF     FSR, RP1          ; Clear RP1 for Data Memory Bank
                        M       endif
004A 0130               M               IORWF    D_address, F
                     00142              MOVF_MAC       D_address, F
                        M ;
                        M       if ( ( D_address & 0x020 ) == 0x020 )
                        M               BSF     FSR, RP0          ; Set RP0 for Data Memory Bank
                        M       else
004B 04A4               M               BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                        M       endif
                        M ;
                        M       if ( ( D_address & 0x040 ) == 0x040 )
                        M               BSF     FSR, RP1          ; Set RP1 for Data Memory Bank
```

```
                        M      else
004C 04C4               M                BCF     FSR, RP1          ; Clear RP1 for Data Memory Bank
                        M      endif
004D 0230               M                MOVF    D_address, F
                    00143                MOVWF_MAC       D_address
                        M ;
                        M      if ( ( D_address & 0x020 ) == 0x020 )
                        M                BSF     FSR, RP0          ; Set RP0 for Data Memory Bank
                        M      else
004E 04A4               M                BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                        M      endif
                        M ;
                        M      if ( ( D_address & 0x040 ) == 0x040 )
                        M                BSF     FSR, RP1          ; Set RP1 for Data Memory Bank
                        M      else
004F 04C4               M                BCF     FSR, RP1          ; Clear RP1 for Data Memory Bank
                        M      endif
0050 0030               M                MOVWF    D_address
                    00144                RLF_MAC         D_address, F
                        M ;
                        M      if ( ( D_address & 0x020 ) == 0x020 )
                        M                BSF     FSR, RP0          ; Set RP0 for Data Memory Bank
                        M      else
0051 04A4               M                BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                        M      endif
                        M ;
                        M      if ( ( D_address & 0x040 ) == 0x040 )
                        M                BSF     FSR, RP1          ; Set RP1 for Data Memory Bank
                        M      else
0052 04C4               M                BCF     FSR, RP1          ; Clear RP1 for Data Memory Bank
                        M      endif
0053 0370               M                RLF     D_address, F
                    00145                RRF_MAC         D_address, F
                        M ;
                        M      if ( ( D_address & 0x020 ) == 0x020 )
                        M                BSF     FSR, RP0          ; Set RP0 for Data Memory Bank
                        M      else
0054 04A4               M                BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                        M      endif
                        M ;
                        M      if ( ( D_address & 0x040 ) == 0x040 )
                        M                BSF     FSR, RP1          ; Set RP1 for Data Memory Bank
                        M      else
0055 04C4               M                BCF     FSR, RP1          ; Clear RP1 for Data Memory Bank
                        M      endif
0056 0330               M                RRF     D_address, F
                    00146                SUBWF_MAC       D_address, F
```

```
                        M ;
                        M       if ( ( D_address & 0x020 ) == 0x020 )
                        M               BSF     FSR, RP0          ; Set RP0 for Data Memory Bank
                        M       else
0057 04A4               M               BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                        M       endif
                        M ;
                        M       if ( ( D_address & 0x040 ) == 0x040 )
                        M               BSF     FSR, RP1          ; Set RP1 for Data Memory Bank
                        M       else
0058 04C4               M               BCF     FSR, RP1          ; Clear RP1 for Data Memory Bank
                        M       endif
0059 00B0               M               SUBWF   D_address, F
                      00147             SWAPF_MAC       D_address, F
                        M ;
                        M       if ( ( D_address & 0x020 ) == 0x020 )
                        M               BSF     FSR, RP0          ; Set RP0 for Data Memory Bank
                        M       else
005A 04A4               M               BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                        M       endif
                        M ;
                        M       if ( ( D_address & 0x040 ) == 0x040 )
                        M               BSF     FSR, RP1          ; Set RP1 for Data Memory Bank
                        M       else
005B 04C4               M               BCF     FSR, RP1          ; Clear RP1 for Data Memory Bank
                        M       endif
005C 03B0               M               SWAPF   D_address, F
                      00148             XORWF_MAC       D_address, F
                        M ;
                        M       if ( ( D_address & 0x020 ) == 0x020 )
                        M               BSF     FSR, RP0          ; Set RP0 for Data Memory Bank
                        M       else
005D 04A4               M               BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                        M       endif
                        M ;
                        M       if ( ( D_address & 0x040 ) == 0x040 )
                        M               BSF     FSR, RP1          ; Set RP1 for Data Memory Bank
                        M       else
005E 04C4               M               BCF     FSR, RP1          ; Clear RP1 for Data Memory Bank
                        M       endif
005F 01B0               M               XORWF   D_address, F
                      00149             PAGE_MAC        P1_CALL_4
                        M ;
                        M       if ( ( P1_CALL_4 & 0x0200 ) == 0x0200 )
                        M               BSF     STATUS, PA0       ; Set PA0 for Program Memory Page
                        M       else
0060 04A3               M               BCF     STATUS, PA0       ; Clear PA0 for Program Memory Page
```

```
                         M      endif
                         M ;
                         M      if ( ( P1_CALL_4 & 0x0400 ) == 0x0400 )
                         M                BSF     STATUS, PA1       ; Set PA1 for Program Memory Page
                         M      else
0061 04C3                M                BCF     STATUS, PA1       ; Clear PA1 for Program Memory Page
                         M      endif
                         M ;
                      00150               BANK_MAC       D_address
                         M ;
                         M      if ( ( D_address & 0x020 ) == 0x020 )
                         M                BSF     FSR, RP0          ; Set RP0 for Data Memory Bank
                         M      else
0062 04A4                M                BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                         M      endif
                         M ;
                         M      if ( ( D_address & 0x040 ) == 0x040 )
                         M                BSF     FSR, RP1          ; Set RP1 for Data Memory Bank
                         M      else
0063 04C4                M                BCF     FSR, RP1          ; Clear RP1 for Data Memory Bank
                         M      endif
                      00151
                      00152
                      00153
0064                  00154 P1_CALL_5
                      00155 ;
                      00156     if ( (P1_CALL_5_V & 0x0600) != (P1_CALL_5 & 0x0600) )
                      00157        MESSG   “ERROR - User Defined: CALL VECTOR and CALL routine NOT in same page”
                      00158     endif
                      00159 ;
0064 0000             00160               NOP
0065 0000             00161               NOP
                      00162 ;
0200                  00163               org     P2_TOP       ; This is to force an intentional User Defined Message
                      00164 ;
0200 0000             00165               NOP
0201 0000             00166               NOP
0202 0800             00167 P1_CALL_5_END  RETURN
                      00168 ;
                      00169     if ( (P1_CALL_5 & 0x0600) != (P1_CALL_5_END & 0x0600) )
Message[301]: MESSAGE: (Warning - User Defined: Call routine crosses page boundry)
                      00170        MESSG   “Warning - User Defined: Call routine crosses page boundry”
                      00171     endif
                      00172 ;
                      00173
0203 0A08             00174 P2_CALL_1_V    GOTO    P2_CALL_1
0204 0A0B             00175 P2_CALL_2_V    GOTO    P2_CALL_2
```

```
0205 0A13           00176 P2_CALL_3_V    GOTO    P2_CALL_3
0206 0A19           00177 P2_CALL_4_V    GOTO    P2_CALL_4
0207 0A1C           00178 P2_CALL_5_V    GOTO    P2_CALL_5
                    00179
                    00180
0208 0000           00181 P2_CALL_1      NOP
0209 0000           00182                NOP
020A 0800           00183                RETURN
                    00184 ;
                    00185     if ( (P2_CALL_1_V & 0x0600) != (P2_CALL_1 & 0x0600) )
                    00186        MESSG   "ERROR - User Defined: CALL VECTOR and CALL routine NOT in same page"
                    00187     endif
                    00188
                    00189 ;
020B 0000           00190 P2_CALL_2      NOP
020C 0000           00191                NOP
020D 0800           00192                RETURN
                    00193 ;
                    00194     if ( (P2_CALL_2_V & 0x0600) != (P2_CALL_2 & 0x0600) )
                    00195        MESSG   "ERROR - User Defined: CALL VECTOR and CALL routine NOT in same page"
                    00196     endif
                    00197
                    00198 ;
                    00199                CALLM   P1_CALL_3_V
                        M ;
                        M      if ( ( P1_CALL_3_V & 0x0200 ) == 0x0200 )
                        M                BSF     STATUS, PA0       ; Set PA0 for Program Memory Page
                        M      else
020E 04A3               M                BCF     STATUS, PA0       ; Clear PA0 for Program Memory Page
                        M      endif
                        M ;
                        M      if ( ( P1_CALL_3_V & 0x0400 ) == 0x0400 )
                        M                BSF     STATUS, PA1       ; Set PA1 for Program Memory Page
                        M      else
020F 04C3               M                BCF     STATUS, PA1       ; Clear PA1 for Program Memory Page
                        M      endif
                        M ;
0210 0902               M                CALL    P1_CALL_3_V
0211 0000           00200                nop
0212 0000           00201                nop
                    00202
0213 0000           00203 P2_CALL_3      NOP
0214 0000           00204                NOP
0215 0800           00205                RETURN
                    00206 ;
                    00207     if ( (P2_CALL_3_V & 0x0600) != (P2_CALL_3 & 0x0600) )
                    00208        MESSG   "ERROR - User Defined: CALL VECTOR and CALL routine NOT in same page"
```

```
                    00209     endif
                    00210
                    00211 ;
                    00212               GOTOM      P1_CALL_2
                        M ;
                        M      if ( ( P1_CALL_2 & 0x0200 ) == 0x0200 )
                        M               BSF     STATUS, PA0       ; Set PA0 for Program Memory Page
                        M      else
0216 04A3               M               BCF     STATUS, PA0       ; Clear PA0 for Program Memory Page
                        M      endif
                        M ;
                        M      if ( ( P1_CALL_2 & 0x0400 ) == 0x0400 )
                        M               BSF     STATUS, PA1       ; Set PA1 for Program Memory Page
                        M      else
0217 04C3               M               BCF     STATUS, PA1       ; Clear PA1 for Program Memory Page
                        M      endif
                        M ;
0218 0A0A               M               GOTO    P1_CALL_2
                    00213
                    00214 ;
0219 0000           00215 P2_CALL_4     NOP
021A 0000           00216               NOP
021B 0800           00217               RETURN
                    00218 ;
                    00219     if ( (P2_CALL_4_V & 0x0600) != (P2_CALL_4 & 0x0600) )
                    00220        MESSG   “ERROR - User Defined: CALL VECTOR and CALL routine NOT in same page”
                    00221     endif
                    00222
                    00223 ;
021C 0000           00224 P2_CALL_5     NOP
021D 0000           00225               NOP
021E 0800           00226               RETURN
                    00227 ;
                    00228     if ( (P2_CALL_5_V & 0x0600) != (P2_CALL_5 & 0x0600) )
                    00229        MESSG   “ERROR - User Defined: CALL VECTOR and CALL routine NOT in same page”
                    00230     endif
                    00231
                    00232 ;
                    00233         MOVF_MAC     BANK0_T, 0
                        M ;
                        M      if ( ( BANK0_T & 0x020 ) == 0x020 )
                        M               BSF     FSR, RP0          ; Set RP0 for Data Memory Bank
                        M      else
021F 04A4               M               BCF     FSR, RP0          ; Clear RP0 for Data Memory Bank
                        M      endif
                        M ;
                        M      if ( ( BANK0_T & 0x040 ) == 0x040 )
```

```
                        M               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
                        M       else
0220 04C4               M               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
                        M       endif
0221 0210               M               MOVF    BANK0_T, 0
                      00234           MOVWF_MAC    BANK1_T
                        M ;
                        M       if ( ( BANK1_T & 0x020 ) == 0x020 )
0222 05A4               M               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
                        M       else
                        M               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
                        M       endif
                        M ;
                        M       if ( ( BANK1_T & 0x040 ) == 0x040 )
                        M               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
                        M       else
0223 04C4               M               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
                        M       endif
0224 0030               M               MOVWF    BANK1_T
                      00235           MOVF_MAC     BANK2_T, 0
                        M ;
                        M       if ( ( BANK2_T & 0x020 ) == 0x020 )
                        M               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
                        M       else
0225 04A4               M               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
                        M       endif
                        M ;
                        M       if ( ( BANK2_T & 0x040 ) == 0x040 )
0226 05C4               M               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
                        M       else
                        M               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
                        M       endif
0227 0210               M               MOVF    BANK2_T, 0
                      00236           MOVWF_MAC    BANK3_T
                        M ;
                        M       if ( ( BANK3_T & 0x020 ) == 0x020 )
0228 05A4               M               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
                        M       else
                        M               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
                        M       endif
                        M ;
                        M       if ( ( BANK3_T & 0x040 ) == 0x040 )
0229 05C4               M               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
                        M       else
                        M               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
                        M       endif
022A 0030               M               MOVWF    BANK3_T
```

```
                    00237
                    00238 ;
0400                00239               org     P3_TOP
                    00240 ;
0400 0A05           00241 P3_CALL_1_V   GOTO    P3_CALL_1
0401 0A08           00242 P3_CALL_2_V   GOTO    P3_CALL_2
0402 0A0B           00243 P3_CALL_3_V   GOTO    P3_CALL_3
0403 0A0E           00244 P3_CALL_4_V   GOTO    P3_CALL_4
0404 0A05           00245 P3_CALL_5_V   GOTO    P3_CALL_5
                    00246
                    00247
0405 0000           00248 P3_CALL_1     NOP
0406 0000           00249               NOP
0407 0800           00250               RETURN
                    00251 ;
                    00252     if ( (P3_CALL_1_V & 0x0600) != (P3_CALL_1 & 0x0600) )
                    00253        MESSG   “ERROR - User Defined: CALL VECTOR and CALL routine NOT in same page”
                    00254     endif
                    00255
                    00256 ;
0408 0000           00257 P3_CALL_2     NOP
0409 0000           00258               NOP
040A 0800           00259               RETURN
                    00260 ;
                    00261     if ( (P3_CALL_2_V & 0x0600) != (P3_CALL_2 & 0x0600) )
                    00262        MESSG   “ERROR - User Defined: CALL VECTOR and CALL routine NOT in same page”
                    00263     endif
                    00264
                    00265 ;
040B 0000           00266 P3_CALL_3     NOP
040C 0000           00267               NOP
040D 0800           00268               RETURN
                    00269 ;
                    00270     if ( (P3_CALL_3_V & 0x0600) != (P3_CALL_3 & 0x0600) )
                    00271        MESSG   “ERROR - User Defined: CALL VECTOR and CALL routine NOT in same page”
                    00272     endif
                    00273
                    00274 ;
040E 0000           00275 P3_CALL_4     NOP
040F 0000           00276               NOP
0410 0800           00277               RETURN
                    00278 ;
                    00279     if ( (P3_CALL_4_V & 0x0600) != (P3_CALL_4 & 0x0600) )
                    00280        MESSG   “ERROR - User Defined: CALL VECTOR and CALL routine NOT in same page”
                    00281     endif
                    00282
                    00283 ;
```

```
0600                00284              org     P4_TOP
                    00285 ;
0600 0A08           00286 P4_CALL_1_V  GOTO    P4_CALL_1
0601 0A0B           00287 P4_CALL_2_V  GOTO    P4_CALL_2
0602 0A0E           00288 P4_CALL_3_V  GOTO    P4_CALL_3
0603 0A11           00289 P4_CALL_4_V  GOTO    P4_CALL_4
0604 0A14           00290 P4_CALL_5_V  GOTO    P4_CALL_5
                    00291
                    00292 ;
0605 0000           00293 P3_CALL_5    NOP            ; This is to force an intentional User Defined Message
0606 0000           00294              NOP
0607 0800           00295              RETURN
                    00296 ;
                    00297     if ( (P3_CALL_5_V & 0x0600) != (P3_CALL_5 & 0x0600) )
Message[301]: MESSAGE: (ERROR - User Defined: CALL VECTOR and CALL routine NOT in same page)
                    00298        MESSG   “ERROR - User Defined: CALL VECTOR and CALL routine NOT in same page”
                    00299     endif
                    00300
                    00301 ;
0608 0000           00302 P4_CALL_1    NOP
0609 0000           00303              NOP
060A 0800           00304              RETURN
                    00305 ;
                    00306     if ( (P4_CALL_1_V & 0x0600) != (P4_CALL_1 & 0x0600) )
                    00307        MESSG   “ERROR - User Defined: CALL VECTOR and CALL routine NOT in same page”
                    00308     endif
                    00309
                    00310 ;
060B 0000           00311 P4_CALL_2    NOP
060C 0000           00312              NOP
060D 0800           00313              RETURN
                    00314 ;
                    00315     if ( (P4_CALL_2_V & 0x0600) != (P4_CALL_2 & 0x0600) )
                    00316        MESSG   “ERROR - User Defined: CALL VECTOR and CALL routine NOT in same page”
                    00317     endif
                    00318
                    00319 ;
060E 0000           00320 P4_CALL_3    NOP
060F 0000           00321              NOP
0610 0800           00322              RETURN
                    00323 ;
                    00324     if ( (P4_CALL_3_V & 0x0600) != (P4_CALL_3 & 0x0600) )
                    00325        MESSG   “ERROR - User Defined: CALL VECTOR and CALL routine NOT in same page”
                    00326     endif
                    00327
                    00328 ;
0611 0000           00329 P4_CALL_4    NOP
```

```
0612 0000           00330              NOP
0613 0800           00331              RETURN
                    00332 ;
                    00333     if ( (P4_CALL_4_V & 0x0600) != (P4_CALL_4 & 0x0600) )
                    00334        MESSG   “ERROR - User Defined: CALL VECTOR and CALL routine NOT in same page”
                    00335     endif
                    00336
                    00337 ;
0614 0000           00338 P4_CALL_5    NOP
0615 0000           00339              NOP
0616 0800           00340              RETURN
                    00341 ;
                    00342     if ( (P4_CALL_5_V & 0x0600) != (P4_CALL_5 & 0x0600) )
                    00343        MESSG   “ERROR - User Defined: CALL VECTOR and CALL routine NOT in same page”
                    00344     endif
                    00345 ;
                    00346 ;
07FF                00347              org     RESET_V
                    00348 ;
07FF 0A05           00349              GOTO    START           ; Goto the begining of the program
                    00350
                    00351         end

MEMORY USAGE MAP (‘X’ = Used,  ‘-’ = Unused)

0000 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0040 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXX---------- ----------------
0200 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXX----- ----------------
0400 : XXXXXXXXXXXXXXXX X--------------- ---------------- ----------------
0600 : XXXXXXXXXXXXXXXX XXXXXX---------- ---------------- ----------------
07C0 : ---------------- ---------------- ---------------- ---------------X

All other memory blocks unused.

Program Memory Words Used:   186
Program Memory Words Free:  1862


Errors   :     0
Warnings :     0 reported,     0 suppressed
Messages :     2 reported,    10 suppressed
```

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: 630-285-0071 Fax: 630-285-0075

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.
Two Prestige Place, Suite 150
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: 91-80-229-0061 Fax: 91-80-229-0062

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
RM 406 Shanghai Golden Bridge Bldg.
2077 Yan'an Road West, Hongiao District
Shanghai, PRC 200335
Tel: 86-21-6275-5700
Fax: 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 Müchen, 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

5/8/97

MICROCHIP