Macros for Page and Bank Switching

INTRODUCTION

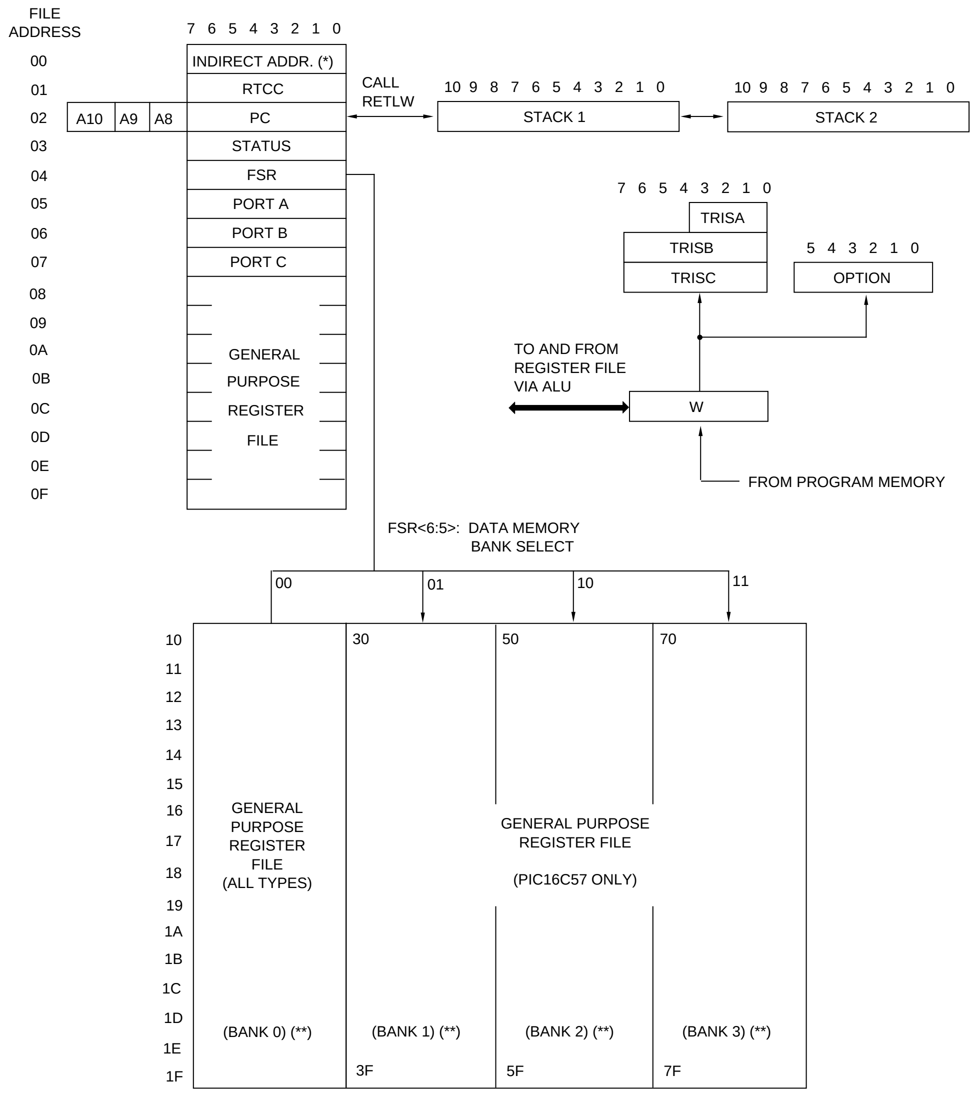
This application note discusses the use of the MPASM assembler's conditional assembly to automatically switch between program memory pages or to set the data memory banks. These macros, along with the long call technique (see Application Note AN581), ease the development of software. Though the use of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

The PIC16C5X family of devices has an architecture where the 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 and PA0, are used to manage the program memory page. Two bits of the FSR register, bits 6 and 5, manage the data memory bank. 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.

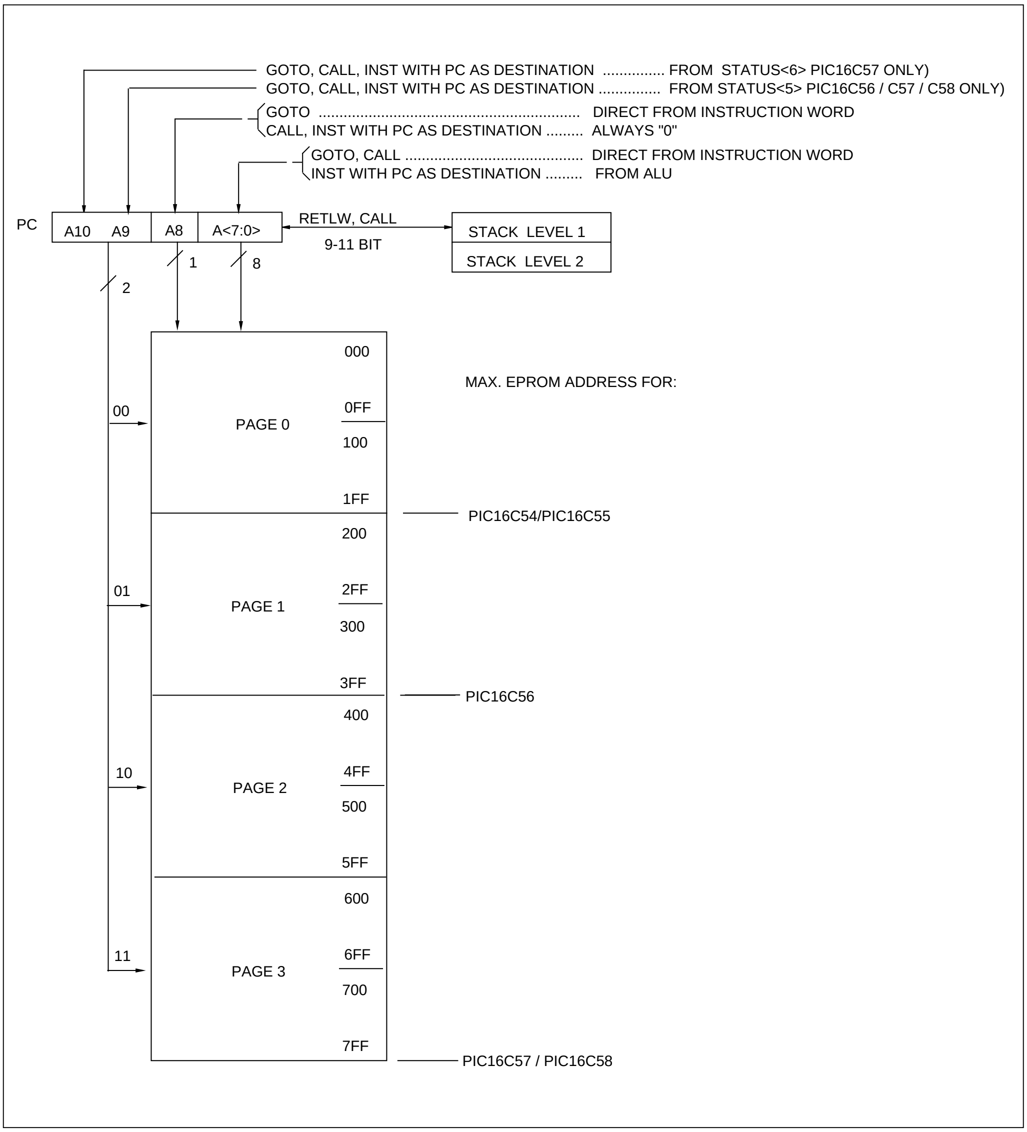
When the address of the data memory has the MSb (bit 4) of the direct address is cleared, or FSR<4> cleared (for indirect addressing), the address 0h through 0Fh is accessed. That is when accessing addresses 0h through 0Fh, the bank selection (FSR<6:5>) bits are ignored. This means that data memory addresses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

FIGURE 1: PROGRAM MEMORY ORGANIZATION



(*) NOT A PHYSICALLY IMPLEMENTED REGISTER. SEE SECTION 4.0 OR DETAILS
(**) BANK 0 IS AVAILABLE ON ALL MICROCONTROLLERS WHILE BANK 1 TO BANK 3 ARE ONLY AVAILABLE ON THE PIC16C57 AND PIC16C58.

FIGURE 2: DATA MEMORY MAP


GOTO, CALL, INST WITH PC AS DESTINATION .... FROM STATUS<6> PIC16C57 ONLY)
GOTO, CALL, INST WITH PC AS DESTINATION .... FROM STATUS<5> PIC16C56 / C57 / C58 ONLY)
GOTO .... DIRECT FROM INSTRUCTION WORD
CALL, INST WITH PC AS DESTINATION .... ALWAYS "0"
GOTO, CALL .... DIRECT FROM INSTRUCTION WORD
INST WITH PC AS DESTINATION .... FROM ALU
PC
A10
A9
A8
A<7:0>
RETLW, CALL
9-11 BIT
STACK LEVEL 1
STACK LEVEL 2
1
8
2
000
MAX. EPROM ADDRESS FOR:
00
PAGE 0
0FF
100
1FF
PIC16C54/PIC16C55
200
01
PAGE 1
2FF
300
3FF
PIC16C56
400
10
PAGE 2
4FF
500
5FF
600
11
PAGE 3
6FF
700
7FF
PIC16C57 / PIC16C58

Macros for Page and Bank Switching

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 (see 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 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 (see Example 1).

CONCLUSION

The use of these macros simplify the program development by managing the memory resources of the PIC16C5X device. If the application program becomes too large for the desired device 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 1A: 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 1B: GENERATION OF OPTIMUM CODE

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

Written By: Mark Palmer - Sr. Application Engineer
Contributions by: Mike Morse - Sr. Field Application Engineer (Dallas)

APPENDIX A: MACRO FILE

```
   nolist
;*************************************************************************
; This file contains MACROs to ease in the use of the Program Memory
; paging and the Data Memory bank switching for the PIC16C5x devices
;
;    File Name:   AUTO_PG.MAC
;    REVISION:    5-20-94
;*************************************************************************
;
PAGE1_OR_3   EQU       0x0200         ; Program Memory in page 1 or page 3
PAGE2_OR_3   EQU       0x0400         ; Program Memory in page 2 or page 3
;
BANK1_OR_3   EQU       0x020          ; Data Memory in Bank 1 or Bank 3
BANK2_OR_3   EQU       0x040          ; Data Memory in Bank 2 or Bank 3
;
;******************************************************************************
;**                            CALLM     program_address
;** Configures the PA1 and PA0 bits as required, ensures that the CALLed
;** “routine” is in the first 256 locations of the program memory page.
;** If the “routine” is in the second 256 locations of the program memory page,
;** an User Defined ERROR Message is placed in the LISTING file. MPASM
;** presently only places this message in the listing file (i.e. no indication
;** is shown when MPASM completes execution in the ERROR / WARNING listed.
;**
;******************************************************************************
;
CALLM          macro       routine
;
     if ( ( routine & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     STATUS, PA0       ; Set PA0 for Program Memory Page
     else
               BCF     STATUS, PA0       ; Clear PA0 for Program Memory Page
     endif
;
     if ( ( routine & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     STATUS, PA1       ; Set PA1 for Program Memory Page
     else
               BCF     STATUS, PA1       ; Clear PA1 for Program Memory Page
     endif
;
     if ( ( routine & 0x0100 ) == 0x0100 )
       MESSG  “Error - User Defined: CALLed routine in 2nd 256 locations of the
             program memory page”
     endif
;
               CALL    routine
               endm
```

```
;
;*******************************************************************************
;**                             GOTOM     program_address
;** Configures the PA1 and PA0 bits as required, and GOTOs the specified
;** locations of the program memory page.
;**
;*******************************************************************************
;
GOTOM          macro       routine
;
     if ( ( routine & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     STATUS, PA0       ; Set PA0 for Program Memory Page
     else
               BCF     STATUS, PA0       ; Clear PA0 for Program Memory Page
     endif
;
     if ( ( routine & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     STATUS, PA1       ; Set PA1 for Program Memory Page
     else
               BCF     STATUS, PA1       ; Clear PA1 for Program Memory Page
     endif
;
               GOTO    routine
               endm
;
;*******************************************************************************
;**                   ADDWF_MAC     data_address, destination
;** Configures the FSR<6:5> bits as required for the Data Memory addressing
;** and then executes the “ADDWF data_address, destination” instruction. The
;** data_address must be the absolute address and NOT the relative address in
;** the data memory page.
;**
;*******************************************************************************
;
ADDWF_MAC      macro       address, d
;
     if ( ( address & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
     else
               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
     endif
;
     if ( ( address & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
     else
               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
     endif
               ADDWF    address, d
               endm
```

```
;
;******************************************************************************
;**                   ANDWF_MAC     data_address, destination
;** Configures the FSR<6:5> bits as required for the Data Memory addressing
;** and then executes the “ANDWF data_address, destination” instruction. The
;** data_address must be the absolute address and NOT the relative address in
;** the data memory page.
;**
;******************************************************************************
;
ANDWF_MAC      macro       address, d
;
     if ( ( address & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
     else
               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
     endif
;
     if ( ( address & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
     else
               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
     endif
               ANDWF    address, d
               endm
;
;******************************************************************************
;**                   BCF_MAC     data_address, bit
;** Configures the FSR<6:5> bits as required for the Data Memory addressing
;** and then executes the “BCF data_address, bit” instruction. The
;** data_address must be the absolute address and NOT the relative address in
;** the data memory page.
;**
;******************************************************************************
;
BCF_MAC        macro       address, b
;
     if ( ( address & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
     else
               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
     endif
;
     if ( ( address & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
     else
               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
     endif
               BCF     address, b
               endm
```

```
;
;*****************************************************************************
;**                   BSF_MAC     data_address, bit
;** Configures the FSR<6:5> bits as required for the Data Memory addressing
;** and then executes the “BSF data_address, bit” instruction. The
;** data_address must be the absolute address and NOT the relative address in
;** the data memory page.
;**
;*****************************************************************************
;
BSF_MAC        macro       address, b
;
     if ( ( address & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
     else
               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
     endif
;
     if ( ( address & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
     else
               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
     endif
               BSF     address, b
               endm
;
;*****************************************************************************
;**                   BTFSC_MAC     data_address, bit
;** Configures the FSR<6:5> bits as required for the Data Memory addressing
;** and then executes the “BTFSC data_address, bit” instruction. The
;** data_address must be the absolute address and NOT the relative address in
;** the data memory page.
;**
;*****************************************************************************
;
BTFSC_MAC      macro       address, b
;
     if ( ( address & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
     else
               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
     endif
;
     if ( ( address & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
     else
               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
     endif
               BTFSC     address, b
               endm
```

```
;
;******************************************************************************
;**                    BTFSS_MAC     data_address, bit
;** Configures the FSR<6:5> bits as required for the Data Memory addressing
;** and then executes the “BTFSS data_address, bit” instruction. The
;** data_address must be the absolute address and NOT the relative address in
;** the data memory page.
;**
;******************************************************************************
;
BTFSS_MAC      macro       address, b
;
     if ( ( address & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
     else
               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
     endif
;
     if ( ( address & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
     else
               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
     endif
               BTFSS    address, b
               endm
;
;******************************************************************************
;**                    CLRF_MAC     data_address
;** Configures the FSR<6:5> bits as required for the Data Memory addressing
;** and then executes the “CLRF data_address” instruction. The data_address
;** must be the absolute address and NOT the relative address in
;** the data memory page.
;**
;******************************************************************************
;
CLRF_MAC       macro       address
;
     if ( ( address & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
     else
               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
     endif
;
     if ( ( address & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
     else
               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
     endif
               CLRF    address
               endm
```

```
;
;*****************************************************************************
;**                    COMF_MAC     data_address, destination
;** Configures the FSR<6:5> bits as required for the Data Memory addressing
;** and then executes the “COMF data_address, destination” instruction. The
;** data_address must be the absolute address and NOT the relative address in
;** the data memory page.
;**
;*****************************************************************************
;
COMF_MAC       macro       address, d
;
     if ( ( address & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
     else
               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
     endif
;
     if ( ( address & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
     else
               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
     endif
               COMF    address, d
               endm
;
;*****************************************************************************
;**                    DECF_MAC     data_address, destination
;** Configures the FSR<6:5> bits as required for the Data Memory addressing
;** and then executes the “DECF data_address, destination” instruction. The
;** data_address must be the absolute address and NOT the relative address in
;** the data memory page.
;**
;*****************************************************************************
;
DECF_MAC       macro       address, d
;
     if ( ( address & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
     else
               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
     endif
;
     if ( ( address & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
     else
               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
     endif
               DECF    address, d
               endm
```

```
;
;*****************************************************************************
;**                   DECFSZ_MAC     data_address, destination
;** Configures the FSR<6:5> bits as required for the Data Memory addressing
;** and then executes the “DECFSZ data_address, destination” instruction. The
;** data_address must be the absolute address and NOT the relative address in
;** the data memory page.
;**
;*****************************************************************************
;
DECFSZ_MAC     macro       address, d
;
     if ( ( address & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
     else
               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
     endif
;
     if ( ( address & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
     else
               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
     endif
               DECFSZ  address, d
               endm
;
;*****************************************************************************
;**                   INCF_MAC     data_address, destination
;** Configures the FSR<6:5> bits as required for the Data Memory addressing
;** and then executes the “INCF data_address, destination” instruction. The
;** data_address must be the absolute address and NOT the relative address in
;** the data memory page.
;**
;*****************************************************************************
;
INCF_MAC       macro       address, d
;
     if ( ( address & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
     else
               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
     endif
;
     if ( ( address & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
     else
               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
     endif
               INCF    address, d
               endm
```

```
;
;*****************************************************************************
;**                    INCFSZ_MAC     data_address, destination
;** Configures the FSR<6:5> bits as required for the Data Memory addressing
;** and then executes the “INCFSZ data_address, destination” instruction. The
;** data_address must be the absolute address and NOT the relative address in
;** the data memory page.
;**
;*****************************************************************************
;
INCFSZ_MAC     macro       address, d
;
     if ( ( address & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
     else
               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
     endif
;
     if ( ( address & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
     else
               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
     endif
               INCFSZ  address, d
               endm
;
;*****************************************************************************
;**                    IORWF_MAC     data_address, destination
;** Configures the FSR<6:5> bits as required for the Data Memory addressing
;** and then executes the “IORWF data_address, destination” instruction. The
;** data_address must be the absolute address and NOT the relative address in
;** the data memory page.
;**
;*****************************************************************************
;
IORWF_MAC      macro       address, d
;
     if ( ( address & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
     else
               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
     endif
;
     if ( ( address & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
     else
               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
     endif
               IORWF    address, d
               endm
```

```
;
;*****************************************************************************
;**                   MOVF_MAC     data_address, destination
;** Configures the FSR<6:5> bits as required for the Data Memory addressing
;** and then executes the “MOVF data_address, destination” instruction. The
;** data_address must be the absolute address and NOT the relative address in
;** the data memory page.
;**
;*****************************************************************************
;
MOVF_MAC       macro       address, d
;
     if ( ( address & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
     else
               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
     endif
;
     if ( ( address & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
     else
               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
     endif
               MOVF    address, d
               endm
;
;*****************************************************************************
;**                   MOVWF_MAC     data_address
;** Configures the FSR<6:5> bits as required for the Data Memory addressing
;** and then executes the “BSF data_address” instruction. The
;** data_address must be the absolute address and NOT the relative address in
;** the data memory page.
;**
;*****************************************************************************
;
MOVWF_MAC      macro       address
;
     if ( ( address & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
     else
               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
     endif
;
     if ( ( address & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
     else
               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
     endif
               MOVWF    address
               endm
```

```
;
;*****************************************************************************
;**                    RLF_MAC     data_address, destination
;** Configures the FSR<6:5> bits as required for the Data Memory addressing
;** and then executes the “RLF data_address, destination” instruction. The
;** data_address must be the absolute address and NOT the relative address in
;** the data memory page.
;**
;*****************************************************************************
;
RLF_MAC         macro       address, d
;
     if ( ( address & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
     else
               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
     endif
;
     if ( ( address & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
     else
               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
     endif
               RLF     address, d
               endm
;
;*****************************************************************************
;**                    RRF_MAC     data_address, destination
;** Configures the FSR<6:5> bits as required for the Data Memory addressing
;** and then executes the “RRF data_address, destination” instruction. The
;** data_address must be the absolute address and NOT the relative address in
;** the data memory page.
;**
;*****************************************************************************
;
RRF_MAC         macro       address, d
;
     if ( ( address & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
     else
               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
     endif
;
     if ( ( address & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
     else
               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
     endif
               RRF     address, d
               endm
```

```
;
;******************************************************************************
;**                  SUBWF_MAC     data_address, destination
;** Configures the FSR<6:5> bits as required for the Data Memory addressing
;** and then executes the “SUBWF data_address, destination” instruction. The
;** data_address must be the absolute address and NOT the relative address in
;** the data memory page.
;**
;******************************************************************************
;
SUBWF_MAC      macro       address, d
;
     if ( ( address & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
     else
               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
     endif
;
     if ( ( address & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
     else
               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
     endif
               SUBWF   address, d
               endm
;
;******************************************************************************
;**                  SWAPF_MAC     data_address, destination
;** Configures the FSR<6:5> bits as required for the Data Memory addressing
;** and then executes the “SWAPF data_address, destination” instruction. The
;** data_address must be the absolute address and NOT the relative address in
;** the data memory page.
;**
;******************************************************************************
;
SWAPF_MAC      macro       address, d
;
     if ( ( address & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
     else
               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
     endif
;
     if ( ( address & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
     else
               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
     endif
               SWAPF   address, d
               endm
```

```
;
;*******************************************************************************
;**                   XORWF_MAC     data_address, destination
;** Configures the FSR<6:5> bits as required for the Data Memory addressing
;** and then executes the “XORWF data_address, destination” instruction. The
;** data_address must be the absolute address and NOT the relative address in
;** the data memory page.
;**
;*******************************************************************************
;
XORWF_MAC      macro       address, d
;
     if ( ( address & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
     else
               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
     endif
;
     if ( ( address & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
     else
               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
     endif
               XORWF   address, d
               endm
;
;*******************************************************************************
;**                   PAGE_MAC     program_address
;** Configures the PA1 and PA0 bits as required
;**
;*******************************************************************************
;
PAGE_MAC          macro       routine
;
     if ( ( routine & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     STATUS, PA0       ; Set PA0 for Program Memory Page
     else
               BCF     STATUS, PA0       ; Clear PA0 for Program Memory Page
     endif
;
     if ( ( routine & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     STATUS, PA1       ; Set PA1 for Program Memory Page
     else
               BCF     STATUS, PA1       ; Clear PA1 for Program Memory Page
     endif
;
               endm
```

```
;
;*******************************************************************************
;**                  BANK_MAC     program_address
;** Configures the FSR<6:5> bits as required for the Data Memory addressing.
;** The data_address must be the absolute address and NOT the relative address
;** in the data memory page.
;**
;*******************************************************************************
;
BANK_MAC      macro       address
;
     if ( ( address & PAGE1_OR_3 ) == PAGE1_OR_3 )
               BSF     FSR, RP0          ; Set RP0 for Data Memory Page
     else
               BCF     FSR, RP0          ; Clear RP0 for Data Memory Page
     endif
;
     if ( ( address & PAGE2_OR_3 ) == PAGE2_OR_3 )
               BSF     FSR, RP1          ; Set RP1 for Data Memory Page
     else
               BCF     FSR, RP1          ; Clear RP1 for Data Memory Page
     endif
               endm
;
;*******************************************************************************
;**                  SAVE_W_AND_STATUS
;** Saves the contects of the W register and the STATUS register to two
;** temporary RAM locations W_TEMP and STATUS_TEMP. These temporary RAM
;** locations should be in the NON-Banked part of Data Memory (8h to Fh).
;** This Macro generates a User Defined Warning (seen only in listing file)
;** if the Data RAM location is in Banked RAM.
;**
;*******************************************************************************
;
SAVE_W_AND_STATUS      macro
;
               MOVWF   W_TEMP
               SWAPF   W_TEMP, F
               SWAPF   STATUS, W
               MOVWF   STATUS_TEMP
     if ( ( W_TEMP & 0x0F0 ) != 0x00 )
       MESSG “Warning - User Defined: W_TEMP register is defined to be in BANKed
             memory”
     endif
;
     if ( ( STATUS_TEMP & 0x0F0 ) != 0x00 )
       MESSG “Warning - User Defined: STATUS_TEMP register is defined to be in BANKed
             memory”
     endif
               endm
```

```
;
;******************************************************************************
;**                    RESTORE_W_AND_STATUS
;** Saves the contects of the W register and the STATUS register to two
;** temporary RAM locations W_TEMP and STATUS_TEMP. These temporary RAM
;** locations should be in the NON-Banked part of Data Memory (8h to Fh).
;** This Macro generates a User Defined Warning (seen only in listing file)
;** if the Data RAM location is in Banked RAM.
;**
;******************************************************************************
;
RESTORE_W_AND_STATUS      macro
;
               SWAPF   STATUS_TEMP, W
               MOVWF   STATUS
               SWAPF   W_TEMP, W
     if ( ( W_TEMP & 0x0F0 ) != 0x00 )
       MESSG “Warning - User Defined: W_TEMP register is defined to be in BANKed
             memory”
     endif
;
     if ( ( STATUS_TEMP & 0x0F0 ) != 0x00 )
       MESSG “Warning - User Defined: STATUS_TEMP register is defined to be in BANKed
             memory”
     endif
               endm
;

     list
```

NOTES:

WORLDWIDE SALES & SERVICE

AMERICAS

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

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

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

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

Dallas
Microchip Technology Inc.
14651 Dallas Parkway, Suite 816
Dallas, TX 75240-8809
Tel: 214 991-7177 Fax: 214 991-8588

Dayton
Microchip Technology Inc.
35 Rockridge Road
Englewood, OH 45322
Tel: 513 832-2543 Fax: 513 832-2841

Los Angeles
Microchip Technology Inc.
18201 Von Karman, Suite 455
Irvine, CA 92715
Tel: 714 263-1888 Fax: 714 263-1338

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

AMERICAS (continued)

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

ASIA/PACIFIC

Hong Kong
Microchip Technology
Unit No. 3002-3004, Tower 1
Metroplaza
223 Hing Fong Road
Kwai Fong, N.T. Hong Kong
Tel: 852 2 401 1200 Fax: 852 2 401 3431

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

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

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

EUROPE

United Kingdom
Arizona Microchip Technology Ltd.
Unit 6, The Courtyard
Meadow Bank, Furlong Road
Bourne End, Buckinghamshire SL8 5AJ
Tel: 44 0 1628 851077 Fax: 44 0 1628 850259

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

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

Italy
Arizona Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Pegaso Ingresso No. 2
Via Paracelso 23, 20041
Agrate Brianza (MI) Italy
Tel: 39 039 689 9939 Fax: 39 039 689 9883

JAPAN

Microchip Technology Intl. Inc.
Benex S-1 6F
3-18-20, Shin Yokohama
Kohoku-Ku, Yokohama
Kanagawa 222 Japan
Tel: 81 45 471 6166 Fax: 81 45 471 6122

9/22/95