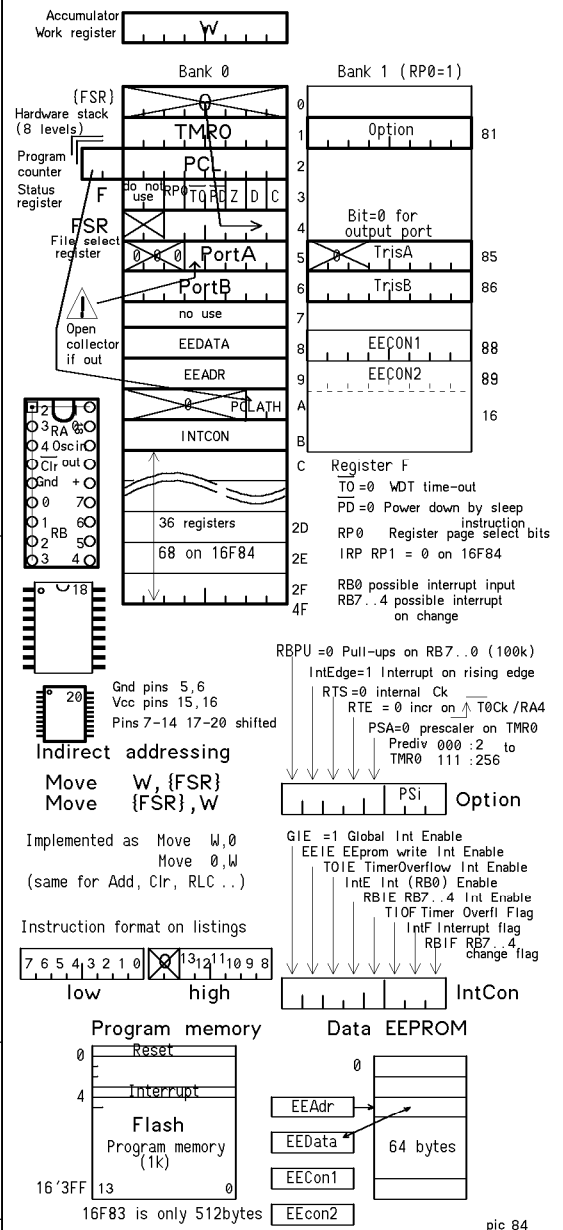


| Code 14 bits Microchip         | CALM                       | Flags                                   |
|--------------------------------|----------------------------|-----------------------------------------|
| 110000 kkkkkkkk movlw val      | Move #Val,W                | [none]                                  |
| 000000 lfffffffff movwf reg    | Move W,Reg                 | [none]                                  |
| 001000 0fffffffff movf reg,0   | Move Reg,W                 | [Z]                                     |
| 001000 1fffffffff movf reg,1   | Test Reg                   | [Z] Move Reg,Reg                        |
| 000000 01100010 option         | Move W,Option <sup>1</sup> | [none]                                  |
| 000000 01100101 tris porta     | Move W,TrisA <sup>1</sup>  | [none]                                  |
| 000000 01100110 tris portb     | Move W,TrisB <sup>1</sup>  | [none]                                  |
| 111110 kkkkkkkk addlw val      | Add #Val,W                 | [C, D, Z] #Val+W -> W                   |
| 000111 0fffffffff addwf reg,0  | Add Reg,W                  | [C, D, Z] Reg+W -> W                    |
| 000111 1fffffffff addwf reg,1  | Add W,Reg                  | [C, D, Z] Reg+W -> Reg                  |
| 111100 kkkkkkkk sublw val      | Sub #Val,W                 | [C <sup>3</sup> , D, Z] #Val+(-W) -> W  |
| 000010 0fffffffff subwf reg,0  | Sub W,Reg,W <sup>2</sup>   | [C <sup>3</sup> , D, Z] Reg+(-W) -> W   |
| 000010 1fffffffff subwf reg,1  | Sub W,Reg                  | [C <sup>3</sup> , D, Z] Reg+(-W) -> Reg |
| 111001 kkkkkkkk andlw val      | And #Val,W                 | [Z]                                     |
| 000101 0fffffffff andwf reg,0  | And Reg,W                  | [Z]                                     |
| 000101 1fffffffff andwf reg,1  | And W,Reg                  | [Z]                                     |
| 111000 kkkkkkkk iorlw val      | Or #Val,W                  | [Z]                                     |
| 000100 0fffffffff iorwf reg,0  | Or Reg,W                   | [Z]                                     |
| 000100 1fffffffff iorwf reg,1  | Or W,Reg                   | [Z]                                     |
| 111010 kkkkkkkk xorlw val      | Xor #Val,W                 | [Z]                                     |
| 000110 0fffffffff xorwf reg,0  | Xor Reg,W                  | [Z]                                     |
| 000110 1fffffffff xorwf reg,1  | Xor W,Reg                  | [Z]                                     |
| 001110 0fffffffff swapf reg,0  | Swap Reg,W                 | [none]                                  |
| 001110 1fffffffff swapf reg,1  | Swap Reg                   | [none]                                  |
| 000001 1fffffffff clrf reg     | Clr Reg                    | [Z=1]                                   |
| 000001 00000000 clrw           | Clr W                      | [Z=1]                                   |
| 000000 01100100 clrwdt         | Clr WDT                    | [T0=1, PD=1]                            |
| 000000 01100011 sleep          | Clr WDT,OSC                | [T0=1, PD=0]                            |
| 001001 1fffffffff comf reg,1   | Not Reg                    | [Z]                                     |
| 001001 0fffffffff comf reg,0   | Not Reg,W                  | [Z]                                     |
| 001010 1fffffffff incf reg,1   | Inc Reg                    | [Z] Reg+1 -> Reg                        |
| 001010 0fffffffff incf reg,0   | Inc Reg,W                  | [Z] Reg->Reg Reg+1->W                   |
| 000011 1fffffffff decf reg,1   | Dec Reg                    | [Z] Reg-1 -> Reg                        |
| 000011 0fffffffff decf reg,0   | Dec Reg,W                  | [Z] Reg->Reg Reg-1->W                   |
| 001101 1fffffffff rlf reg,1    | RLC Reg                    | [C]                                     |
| 001101 0fffffffff rlf reg,0    | RLC Reg,W                  | [C]                                     |
| 001100 1fffffffff rrf reg,1    | RRC Reg                    | [C]                                     |
| 001100 0fffffffff rrf reg,0    | RRC Reg,W                  | [C]                                     |
| 0100bb bfffffffff bcf reg,b    | Clr Reg:#b                 | [none]                                  |
| 0101bb bfffffffff bsf reg,b    | Set Reg:#b                 | [none]                                  |
| 001111 1fffffffff incfsz reg,1 | IncSkip,EQ Reg             | [none]                                  |
| 001111 0fffffffff incfsz reg,0 | IncSkip,EQ Reg,W           | [none]                                  |
| 001011 1fffffffff decfsz reg,1 | DecSkip,EQ Reg             | [none]                                  |
| 001011 0fffffffff decfsz reg,0 | DecSkip,EQ Reg,W           | [none]                                  |
| 0110bb bfffffffff btfsc reg,b  | TestSkip,BC Reg:#b         | [none]                                  |
| 0111bb bfffffffff btfss reg,d  | TestSkip,BS Reg:#b         | [none]                                  |
| 1010kk kkkkkkkk goto addr      | Jump Addr                  | [none]                                  |
| 1000kk kkkkkkkk call addr      | Call Addr                  | [none]                                  |
| 110100 kkkkkkkk retlw val      | RetMove #Val,W             | [none]                                  |
| 000000 00001000 return         | Ret                        | [none]                                  |
| 000000 00001001 retfie         | Reti                       | [none]                                  |
| 000000 00000000 nop            | Nop                        | [none]                                  |
| bcf status,0                   | CLRC equiv to              | Clr F:#0                                |
| bsf status,0                   | SETC equiv to              | Set F:#0                                |
|                                | CLRZ SETZ CLRD SETD        | Clr F:#2 etc.                           |
| btfsc status,0                 | Skip,CS equiv to           | TestSkip,BS F:#0                        |
| btfss status,0                 | Skip,CC equiv to           | TestSkip,BC F:#0                        |
| btfsc status,1                 | Skip,DS equiv to           | TestSkip,BS F:#1                        |
| btfss status,1                 | Skip,DC equiv to           | TestSkip,BC F:#1                        |
| btfsc status,2                 | Skip,EQ Skip,ZSequiv to    | TestSkip,BS F:#2                        |
| btfss status,2                 | Skip,NE Skip,ZCequiv to    | TestSkip,BC F:#2                        |

16F84/16C84 Registers



Meaning of non trivial instructions

Swap Reg      Swap Reg,W      Swap Reg,W

IncSkip,EQ Reg      Increment reg and skip next instruction if result is equal to zero

IncSkip,EQ Reg,W      Copy Reg into W, increment W and skip next instruction if result is equal to zero

TestSkip,BC Reg:#b      Test bit number b in Reg and skip if this bit is clear (zero)

,BS      ... if this bit is set (one)

RetMove #Val,W      Return from routine with immediate value Val prepared into W

ClrC      Clear Carry bit

SetC      Set Carry bit

Skip,CS      Skip next instruction if Carry bit is set (one)

picinstnt

Not available on PIC 16C5x and PIC 12C50x

Reg is one of the processor register or user variable (addr 0 to 16'2F)

Note 1: Not available on 17C family (select bank 1)

Note 2: Calm assembler accepts the simpler but confusing Sub #Val,W and Sub Reg,W – do not use

Note 3: Processor add the 2-s complement; Carry value is inverted compared to most other μP

All instructions execute in 4 clock cycles (1 μS at 4 MHz) except the Skip, Jump, Call and Ret

Compare for equality: Xor W,Reg or Xor Reg,W      Compare for Inequality: Sub instruction, but be careful