PIC16F870 TO PIC16F882

The PIC16F870 is no longer in production at Microchip, but is currently available from Digi-Key. For our books "Microcontrol'n Apps" and "Time'n and Count'n", the PIC16F882 may be used instead, with minor philosophy/code changes. For our purposes, they are pin-for-pin compatible.

The F882 is a good substitute. It has all of the features of the F870, plus more, so it can be used as an F870 simply by ignoring the additional features (almost). The new features that we will need to adapt to are:

1) The F882 has two configuration words instead of one.
2) The port A analog pins are the same. There are six new ones on port B, which we will ignore. Because there are a total of eleven A/D pins, thier selection cannot be handled by four bits in ADCON1. Two new registers handle that. They are ANSEL for port A which we will use, and ANSELH for port B which we will ignore. The two registers are used to enable analog I/O on a one-bit-per-pin basis.
3) The A/D converter module and comparator are disabled on reset (default) and require no attention, unless your application involves A/D.
4) The A/D conversion go/done bit is bit 1 in the ADCON0 register, not bit 2 as in the F870.
5) The F882 has a very useful internal clock-oscillator. The examples in "Time'n and Count'n" require the use of the external crystal-controlled oscillator (as shown) as precise timing is the subject of the book. The examples in "Microcontrol'n Apps" do not require the use of the external crystal-controlled oscillator, but it will be retained to avoid having to change the code and schematics. If you wish, you may use the internal 4 MHz option instead which requires changing the oscillator selection bits in Configuration Register 1.

PINS AND FUNCTIONS

The methods for selecting pin functions are summarized in the following table:

PIC16F882 I/O

<table>
<tr><td>RA0,1,2,3,4,5 options</td><td rowspan="3">Always I/O</td><td rowspan="3">Select via software using control registers</td></tr>
<tr><td>RB0,1,2,3,4,5 options</td></tr>
<tr><td>RC0,1,2,3,4,5,6,7 opt</td></tr>
<tr><td>RB3/PGM</td><td>Digital I/O vs. Program</td><td rowspan="6">Select via config register 1</td></tr>
<tr><td>RE3/$\overline{\text{MCLR}}$/Vpp</td><td>Digital Input vs. $\overline{\text{MCLR}}$</td></tr>
<tr><td>RA6/OSC2/CLKOUT</td><td>Digital I/O vs. OSC2 vs. clock output</td></tr>
<tr><td>RA7/OSC1/CLKIN</td><td>Digital I/O vs. OSC1 vs. clock input</td></tr>
<tr><td>RB6/ICSPCLK</td><td>Digital I/O vs. Debug</td></tr>
<tr><td>RB7/ICSPDAT</td><td>Digital I/O vs. Debug</td></tr>
</table>

FILE REGISTERS

Two "new" file registers required for A/D applications are used in the F882. They are:

ANSEL	0x188
ANSELH	0x189

The file register used for selecting the frequency of the internal clock-oscillator (when used) is:

OSCCON	0x8F

Details on their use follow.

CONFIGURATION BITS

There are two configuration registers in the F882. The configuration bits and what they control are shown in the following diagrams:

Configuration Register 1

$\overline{\text{DEBUG}}$	LVP	FCMEN	IESO	BOREN1	BOREN0	$\overline{\text{CPD}}$	$\overline{\text{CP}}$	MCLRE	$\overline{\text{PWRTE}}$	WDTEN	FOSC2	FOSC1	FOSC0
13													0

Bit 13 **$\overline{\text{DEBUG}}$:** In-Circuit Debugger mode bit
1 = In-Circuit Debugger disabled, RB7 and RB6 are I/O pins
0 = In-Circuit Debugger enabled, RB7 and RB6 are dedicated to the debugger

Bit 12 **LVP:** Low voltage in-circuit serial programming enable bit
1 = Enabled - RB3 is programming pin
0 = Disabled - RB3 is digital I/O pin

Bit 11 **FCMEN:** Fail-safe clock monitor enabled bit
1 = Fail-safe clock monitor is enabled
0 = Fail-safe clock monitor is disabled

Bit 10 **IESO:** Internal/external switchover bit
1 = Internal/external switchover mode is enabled
0 = Internal/external switchover mode is disabled

Bits 9,8 **BOREN<1:0>:** Brown-out reset selection bits
11 = BOR enabled
10 = BOR enabled during operation and disabled in sleep
01 = BOR controlled by SBOREN bit of PCON register
00 = BOR disabled

Bit 7 **$\overline{\text{CPD}}$:** Data code protection bit
1 = Data memory code protection disabled
0 = Data memory code protection enabled

Bit 6 **$\overline{\text{CP}}$:** Code protection bit
1 = Program memory code protection disabled
0 = Program memory code protection enabled

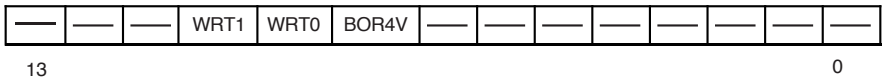
Bit 5 **MCLRE:** R3E/$\overline{\text{MCLR}}$ pin function select bit
1 = $\overline{\text{MCLR}}$
0 = RE3 is digital input pin

Bit 4 **PWRTE:** Power-up timer enable bit
1 = Disabled
0 = Enabled

Bit 3 **WDTEN:** Watchdog timer enable bit
1 = Enabled
0 = Disabled

```
Bits 2,1,0    FOSC3:FOSC2:FOSC1:FOSC0: Oscillator selection bits
              111 = EXTRC oscillator - CLKOUT on RA6 pin
              110 = EXTRC oscillator - I/O on RA6 pin
              101 = INTRC oscillator - CLKOUT on RA6, I/O on RA7
              100 = INTRC oscillator - I/O on RA6, RA7 pins
              011 = EXTCLK - I/O on RA6 pin
              010 = HS oscillator - high speed crystal/resonator
              001 = XT oscillator - crystal/resonator
              000 = LP oscillator - low power crystal
```

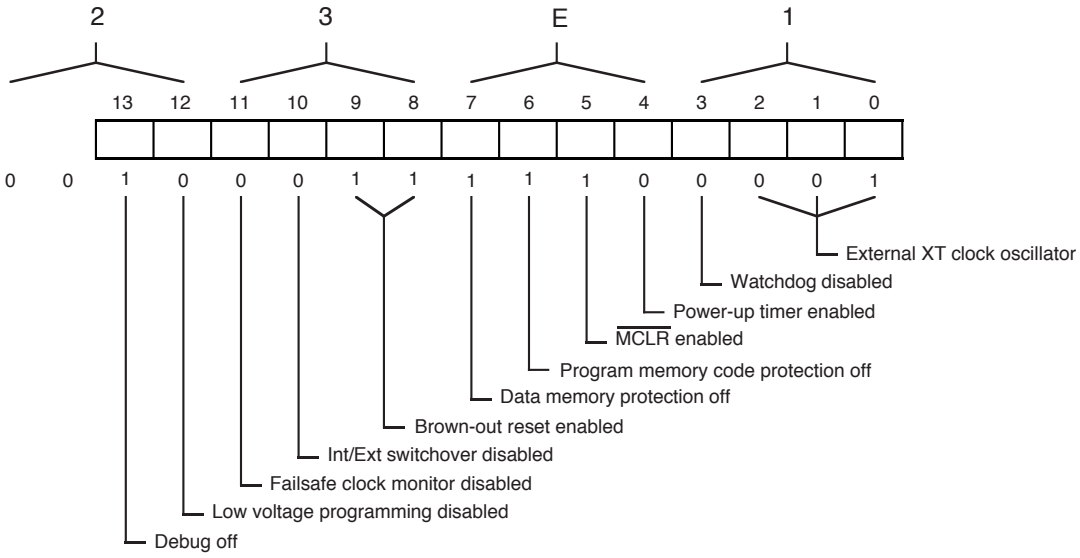
Configuration Register 2



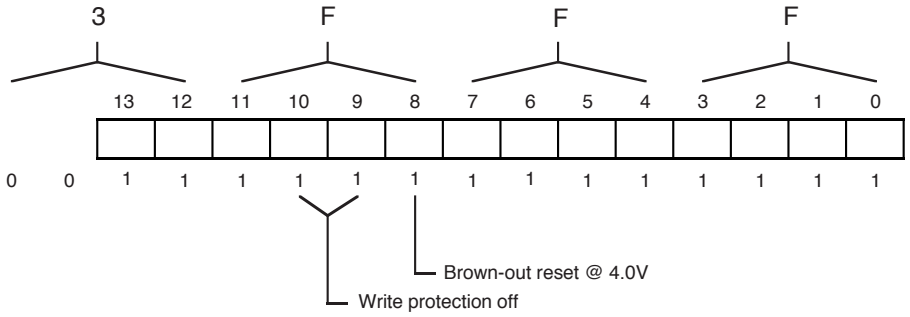

```
Bits 10,9     WRT1:WRT0: Flash program memory self write enable bits
              00 = 0x0000 to 0x03FF write-protected, 0x0400 to 0x07FF
                   may be modified by EECON control
              01 = 0x0000 to 0x00FF write-protected, 0x0100 to 0x07FF
                   may be modified by EECON control
              11 = Write protection off

Bit 8         BOR4V: Brown-out reset selection bit
              1 = Brown-out reset set to 4.0V
              0 = Brown-out reset set to 2.1V
```

For our examples, we will use a 4 MHz crystal-controlled external oscillator in a can as shown in the books. The configuration words are determined as follows:

Configuration Register 1



Configuration Register 2



Since there are two configuration registers in the F882, a configuration register address must be given for each configuration word.

```
__config    h'2007',    h'23e1'     ;config reg 1
__config    h'2008',    h'3fff'     ;config reg 2
```

A/D CONVERSION

For A/D applications, the pins which will be analog need to be selected, The A/D applications in "Microcontrol'n Apps" require one analog input, AN0. This is selected by setting the ANS0 bit (bit 0) in the ANSEL register.

ANSEL - Port A

ANS7	ANS6	ANS5	ANS4	ANS3	ANS2	ANS1	ANS0	0x188
7							0	

Power-on reset 00000000

```
1 = Selected
0 = Not selected
```

Notice that the ANSEL register comes up 00000000 on reset, so port A pins are digital I/O (default).

Although we will not use it, the ANSELH register is shown below for reference. It operates as shown above.

ANSELH - Port B

—	—	ANS13	ANS12	ANS11	ANS10	ANS9	ANS8	0x1F
7							0	

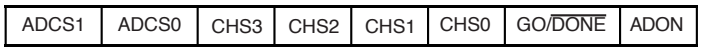
Power-on reset 00000000

Both the analog and comparator functions come up "off" on reset.

Notice that the go/$\overline{\text{done}}$ bit is bit 1 in the the F882 ADCON0 register. It is bit 2 in the F870 ADCON0 register.

The ADCON0 register controls the A/D module.

ADCON0



Power-on reset 00000000

Bits 7,6 **ADCS1:ADCS0** A/D conversion clock select bits
00 = Fosc/2
01 = Fosc/8
10 = Fosc/32
11 = Frc (clock derived from a dedicated internal oscillator = 500 kHz max)

Bits 5-2 **CHS3:CHS0:** Analog channel select bits
0000 = AN0
0001 = AN1
0010 = AN2
0011 = AN3
0100 = AN4
0101 = AN5
0110 = AN6
0111 = AN7
1000 = AN8
1001 = AN9
1010 = AN10
1011 = AN11
1100 = AN12
1101 = AN13
1110 = CVref
1111 = Fixed ref (0.6 volt fixed reference)

Bit 1 **GO/$\overline{\text{DONE}}$:** A/D conversion status bit
1 = A/D conversion in progress (setting this bit starts the A/D conversion)(this bit is automatically cleared by hardware when the A/D conversion is complete)
0 = A/D conversion not in progress

Bit 0 **ADON:** A/D converter enable bit
1 = A/D converter is enabled
0 = A/D converter is disabled and consumes no operating current

The ADFM conversion format bit in the ADCON1 register is used in the A/D examples.

ADCON1

Power-on reset 00000000

```
Bit 7        ADFM: A/D conversion result format bit
             1 = Right justified
             0 = Left justified

Bit 6        Unimplemented: Read as "0"

Bit 5        VCFG1: Voltage reference bit
             1 = Vref- pin
             0 = Vss

Bit 4        VCFG0: Voltage reference bit
             1 = Vref+ pin
             0 = Vdd

Bit 3-0      Unimplemented: Read as "0"
```

CODE CHANGES REQUIRED

Code changes needed to use the F882 in lieu of the F870 are presented below.

The following code snipit contains the items that need to be substituted in the F870 code examples in the books.

```
       list    p=16f882
      __config    h'2007',    h'23e1'     ;config reg 1
      __config    h'2008',    h'3fff'     ;config reg 2
;-----------------------------------------------------
;at device programming time, select:
;     code protection off
;     write protection off
;     watchdog timer disabled
;     external clock osc XT
;        RA6 clock out, n.c.
         Use 4 MHz crystal-controlled oscillator
;     mclr enabled
;     power-up timer enabled
;     brown-out reset enabled @ 4.0v
;     low voltage programming disabled
;     debug disabled
;     fail safe clock monitor disabled
;     int/ext switchover disabled
;A/D is disabled on reset (default)
;comparator is disabled on reset (default)
;=====================================================
```

```
;for A/D examples:

      bsf   ansel,0     ;select AN0 analog pin

;go/done bit is bit 1 in adcon0 register, not bit 2 as in F870



;     cpu equates (memory map)
ansel       equ   0x188
;     bit equates
rp1   equ   6

            org     0x000
start bsf   status,rp0  ;switch to bank 3
      bsf   status,rp1  ;   "   "   "   "
      bsf   ansel,0     ;select AN0 analog pin
      bcf   status,rp1  ;switch to bank 1
      bcf   status,rp0  ;switch to bank 0
```

Following is a simple F882 example (external clock oscillator):

```
;=======F882testxt.ASM=====================3/24/11==
        list    p=16f882
        __config            h'2007',      h'23e1'     ;config reg 1
        __config            h'2008',      h'3fff'     ;config reg 2
        radix   hex
;-----------------------------------------------------
;       cpu equates (memory map)
status  equ     0x03
portc   equ     0x07
trisc   equ     0x87
;       bit equates
rp0     equ     5
;-----------------------------------------------------
        org     0x000
;
start   bsf     status,rp0  ;switch to bank 1
        movlw   b'00000000'   ;port C outputs
        movwf   trisc
        bcf     status,rp0  ;switch back to bank 0
        movlw   0x0f          ;load w with bit pattern
        movwf   portc       ;load port C with contents of W
circle  goto    circle      ;done
;
        end
;-----------------------------------------------------
;at device programming time, select:
;       code protection off
;       write protection off
;       watchdog timer disabled)
;       external clock osc XT, 4 MHz
;               RA6 clock out, n.c.
```

```
;       mclr enabled
;       power-up timer enabled
;       brown-out reset enabled @ 4.0v
;       low voltage programming disabled
;       debug disabled
;       fail safe clock monitor disabled
;       int/ext switchover disabled
;=====================================================
```

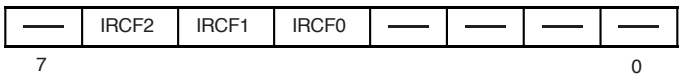
CLOCK OSCILLATOR

The internal clock oscillator may be used (most common), or an external clock oscillator may be used as is done in these books. The details of various external oscillator circuits and components as well as internal clock oscillator use options are given in the Microchip data sheet.

At device programming time, the part must be told via configuration bits whether an internal vs. external clock oscillator will be used.

When the internal clock-oscillator has been selected via the configuration register, the frequency is selected using three bits in the ADCON0 register.

OSCCON



Power-on reset 01100000

```
      Bits 6,5,4   IRCF2:IRCF1:IRCF0: Internal oscillator
                        frequency selection bits
                   111 = 8 MHz
                   110 = 4 Mhz (default)
                   101 = 2 Mhz
                   100 = 1 MHz
                   011 = 500 KHz
                   010 = 250 KHz
                   001 = 125 KHz
                   000 = 31 KHz
```

For experimentation on your own, you can easily use the 4 MHz internal clock oscillator. You can rely on the default or write `b'01100000'` to the OSCCON oscillator control register.

```
;     cpu equates (memory map)
osccon      equ   0x8f


            movlw       b'01100000'       ;4 MHz clock oscillator
            movwf       osccon
```

For example:

```
;=======F882test.ASM=========================1/2/11==
        list    p=16f882
        __config            h'2007',      h'23e4'     ;config reg 1
        __config            h'2008',      h'3fff'     ;config reg 2
        radix   hex
;-----------------------------------------------------
;       cpu equates (memory map)
status  equ     0x03
portc   equ     0x07
trisc   equ     0x87
;       bit equates
rp0     equ     5
;-----------------------------------------------------
        org     0x000
;
start   bsf     status,rp0  ;switch to bank 1
        movlw   b'00000000'   ;port C outputs
        movwf   trisc
        bcf     status,rp0  ;switch back to bank 0
        movlw   0x0f          ;load w with bit pattern
        movwf   portc       ;load port C with contents of W
circle  goto    circle      ;done
;
        end
;-----------------------------------------------------
;at device programming time, select:
;       code protection off
;       write protection off
;       watchdog timer disabled
;       internal clock osc selected, 4 MHz
;               i/o on RA7, 6
;       mclr enabled
;       power-up timer enabled
;       brown-out reset enabled @ 4.0v
;       low voltage programming disabled
;       debug disabled
;       fail safe clock monitor disabled
;       int/ext switchover disabled
;=====================================================
```

IN-CIRCUIT DEBUGGING

In-circuit debugging using a PICkit 2 or PICkit 3 is outlined in another Update available on this web site.