ECE 376 - Homework #2

Assembler, Flow Charts. Due Monday, January 23rd

Assembler Programming

1) Determine the contents of registers W, A, and B after each assembler command:

Command	W	A	B
`; Start`	15	7	3
`movlw 254` move the number 254 to W	**254**	7	3
`addwf A,F` add W to A, put the result in A	254	**5** (261 mod 256)	3
`movff A,B` copy A to B	254	5	**5**
`andlw 0x0F` W and 0x0F put the result in W	**14**	5	5
`andwf A,F` W and A put the result in A	14	**4**	5
`iorwf B,F` W or B put the result in B	14	4	**15**

2) Convert the following C code to assembler (8-bit operations)

- Note: This is an 8-bit processor. It does 8-bit operations fairly easily.

There are many solutions. A 10-instruction solution:

```
;unsigned char A, B, C;
A       equ       0
B       equ       1
C       equ       2

;A = 2*B + 6*C + 3;
        movlw     3

        addwf     B,W
        addwf     B,W

        addwf     C,W
        addwf     C,W
        addwf     C,W
        addwf     C,W
        addwf     C,W
        addwf     C,W

        movwf     A
```

Trickier Solution using the MUL command (also 10 instructions)

```
;unsigned char A, B, C;
A       equ       0
B       equ       1
C       equ       2

;A = 2*B + 6*C + 3;
        movlw     3
        movwf     A

        movf      B,W
        mullw     2
        movf      PRODL,W
        addwf     A,F

        movf      C,W
        mullw     6
        movf      PRODL,W
        addwf     A,F
```

3) Convert the following C code to assembler: (16-bit operations)

- note: 16-bit operations are *much* harder than 8-bit operations

```
; unsigned int A, B, C;

A       equ     0
B       equ     2
C       equ     4

; A = 2*B + 6*C + 3;

        movlw   3
        movwf   A
        clrf    A+1

        movf    B,W
        addwf   A,F
        movf    B+1,W
        addwfc  A+1,F

        movf    B,W
        addwf   A,F
        movf    B+1,W
        addwfc  A+1,F

        movf    C,W
        addwf   A,F
        movf    C+1,W
        addwfc  A+1,F

        movf    C,W
        addwf   A,F
        movf    C+1,W
        addwfc  A+1,F

        movf    C,W
        addwf   A,F
        movf    C+1,W
        addwfc  A+1,F

        movf    C,W
        addwf   A,F
        movf    C+1,W
        addwfc  A+1,F

        movf    C,W
        addwf   A,F
        movf    C+1,W
        addwfc  A+1,F

        movf    C,W
        addwf   A,F
        movf    C+1,W
        addwfc  A+1,F
```

4) Convert the following C code to assembler (if-statements)

- Use cpfseq statements
- prior to its use, set up the W register

```
; unsigned char A, B;

A equ 0
B equ 1

; A = A & 0x0F;

       movlw     0x0F
       andwf     A,F

;if(A == 0) B = 1;

       movlw     0
       cpfseq    A
       goto      L1
       movlw     1
       movwf     B

;if(A == 1) B = 2;
L1:
       movlw     1
       cpfseq    A
       goto      L2
       movlw     2
       movwf     B

;if(A == 2) B = 4;
L2:
       movlw     2
       cpfseq    A
       goto      L3
       movlw     4
       movwf     B

;if(A == 3) B = 8;
L3:
       movlw     3
       cpfseq    A
       goto      L4
       movlw     8
       movwf     B
L4:
       nop
```

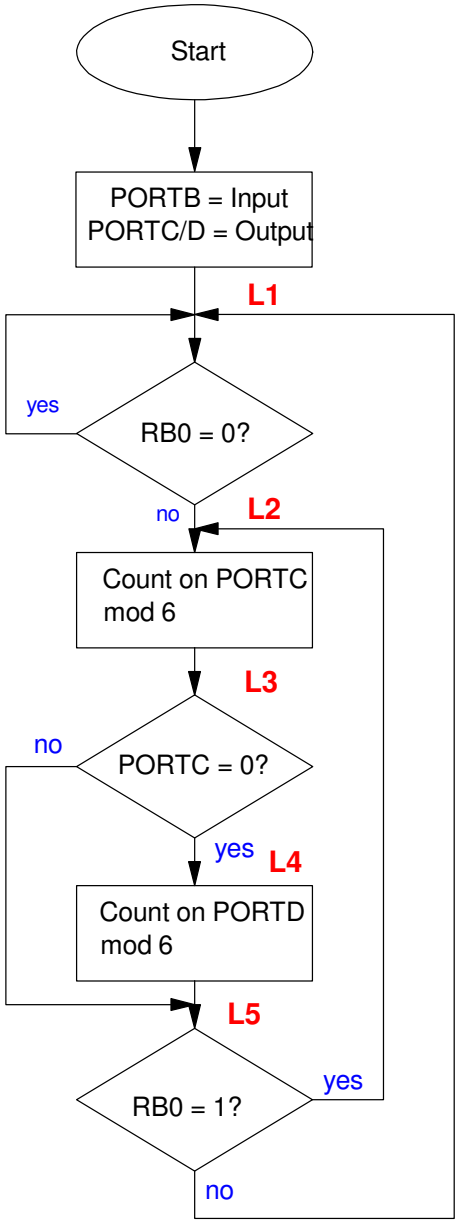
5) (20 points) The flow chart below rolls two six-sided dice

- Press RB0 to roll the dice (count really fast)
- Release RB0 to see the result on PORTC and PORTD

Write the corresponding assembler code.

```
org     0x800
        movlw     0x0F
        movwf     ADCON1

        movlw     0xFF
        movwf     TRISB
        clrf      TRISC
        clrf      TRISD
L1:
        btfss     PORTB,0
        goto      L1
        goto      L2          ; not needed
L2:
        incf      PORTC,F
        movlw     6
        cpfseq    PORTC
        goto      L3
        clrf      PORTC
L3:
        movlw     0
        cpfseq    PORTC
        goto      L5
        goto      L4          ; not needed
L4:
        incf      PORTD,F
        movlw     6
        cpfseq    PORTD
        goto      L5
        clrf      PORTD
L5:
        btfss     PORTB,0
        goto      L1
        goto      L2
```



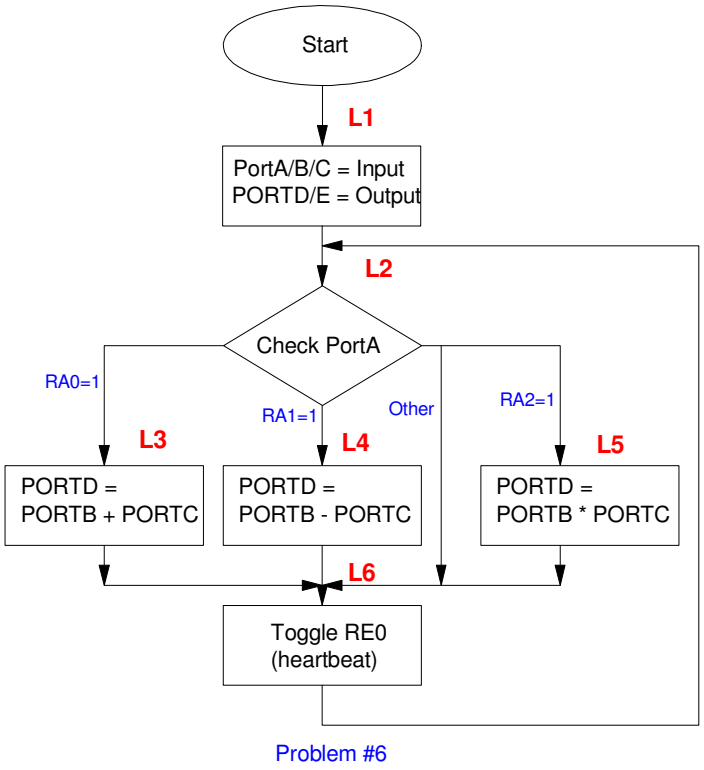
Problem 5: Roll 2d6

6) (20 points) The flow chart below turns your PIC into an 8-bit calculator that adds, subrtacts, and multiplies. Write the corresponding assembly code

```
org     0x800
        movlw     0x0F
        movwf     ADCON1

L1:
        movlw     0xFF
        movwf     TRISA
        movwf     TRISB
        movwf     TRISC
        clrf      TRISD
        clrf      TRISE

L2:
        btfsc     PORTA,0
        goto      L3
        btfsc     PORTA,1
        goto      L4
        btfsc     PORTA,2
        goto      L5
        goto      L6
L3:
        movf      PORTB,W
        addwf     PORTC,W
        movwf     PORTD
        goto      L6
L4:
        movf      PORTC,W
        subwf     PORTB,W
        movwf     PORTD
        goto      L6
L5:
        movf      PORTB
        mulwf     PORTC
        movff     PRODL, PORTD
L6:
        btg       PORTE,0
        goto      L2
```



Problem #6