# ECE 376 - Homework #2

Assembler, Flow Charts, Binary Inputs.

## Assembler Programming

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

| Command | W | A | B |
|---|---|---|---|
| ; Start | 13 | 11 | 9 |
| movf A, W | **11** | 11 | 9 |
| andlw 26 | 11 = b0000 1011<br>26 = b0001 1010<br>---<br>**11 = b0000 1010**<br>**ans = 10** | 11 | 9 |
| addwf B,F | 10 | 11 | **19** |
| comf A,W | b0000 1011<br>---<br>b1111 0100<br>**ans = 244 = –12** | 11 | 20 |
| negf B,F | 244 = –12 | 11 | +244 = –20 |
| subwf A,F | –12 | **11 – (–12) = 23**<br>**ans = 23** | 32 |

## Assembler Coding

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

```
% unsigned char A, B, C;

A       equ       0
B       equ       1
C       equ       2

%  C = 2*A + 3*B + 4;

        movf      A,W
        addwf     A,W     %  2A
        addwf     B,W
        addwf     B,W
        addwf     B,W     % + 3B
        addlw     4       % + 4
        movwf     C
```

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

One method: 22 insructions

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

A       equ 0
B       equ 2
C       equ 4

% C = 2*A + 3*B + 4;

        movff     A,C
        movff     A+1,C+1

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

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

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

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

        movlw     4
        addwf     C,F
        movlw     0
        addwfc    C+1,F
```

Another solution uses the MUL command (15 instructions)

- note: MUL results in a 16-bit result (PRODH : PRODL)

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

A      equ 0
B      equ 2
C      equ 4

% C = 2*A + 3*B + 4;

       movlw     4
       movwf     C
       clrf      C+1

       movf      A,W
       mullw     2
       movf      PRODL
       addwf     C,F
       movf      PRODH
       addwfc    C+1,F

       movf      B,W
       mullw     3
       movf      PRODL
       addwf     C,F
       movf      PRODH
       addwfc    C+1,F
```

3) Convert the following C code to assembler

```
% unsigned char A, B, C;

A      equ       0
B      equ       1
C      equ       2

% if( B > 10 )
%   C = A + 2;
%else
%   C = A + 5;

       movl      10
       cpfsgt    B
       goto      If
       goto      Else
If:    movf      A,W
       addlw     2
       movwf     C
       goto      End
Else:  movlw     5
       addwf     A,W
       movwf     C
End:   nop
```

4) Convert the following C code in to assembler

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

%while( B > 0) {
%   if(B > 10) {
%      C = A + 2;
%   else
%      C = A + 5;
%   }

While:
       movlw     0
       cpfsgt    B
       goto      End

       movlw     10
       cpfsgt    B
       goto      Else
If:
       movf      A,W
       addlw     2
       movwf     C
       goto      While
Else:
       movf      A,W
       addlw     5
       movwf     C
       goto      While
End:
       nop
```

## Flow Charts & Counters

5) The flow chart below turns your PIC into a rigged voting machine

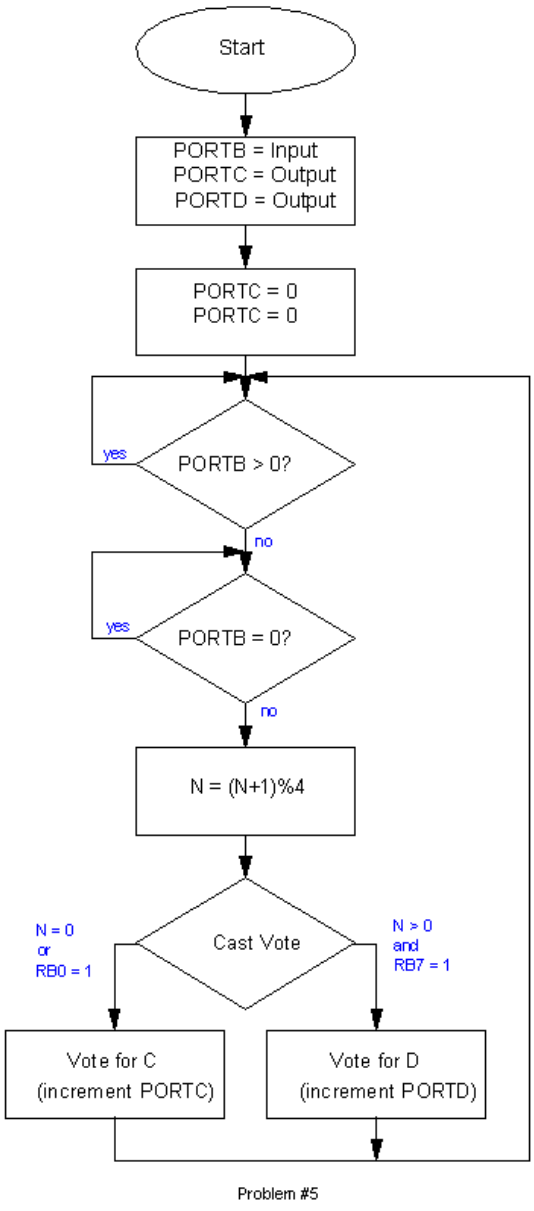
- When you press RB0, one vote is cast for Candidate C
- When you press RB7, one vote is cast for Candiadate D
- Every 4th vote always goes to Candidate C

Write the corresponding assembler code.

```
        org     0x800

        movlw   0xFF
        movwf   TRISB
        clrf    TRISC
        clrf    TRISD
        movlw   0x0F
        movwf   ADCON1

        clrf    PORTC
        clrf    PORTD
L1:
        movlw   0
        cpfsgt  PORTB
        goto    L2
        goto    L1
L2:
        movlw   0
        cpfseq  PORTB
        goto    L3
        goto    L2
L3:
        incf    N,W
        andlw   0x03
        movwf   N
L4:
        movlw   0
        cpfseq  N
        goto    Test2
        goto    VoteC
        btfsc   PORTB,0
        goto    VoteC
VoteD
        incf    PORTD,F
        goto    L1
VoteC
        incf    PORTC,F
        goto    L1
```




Problem #5

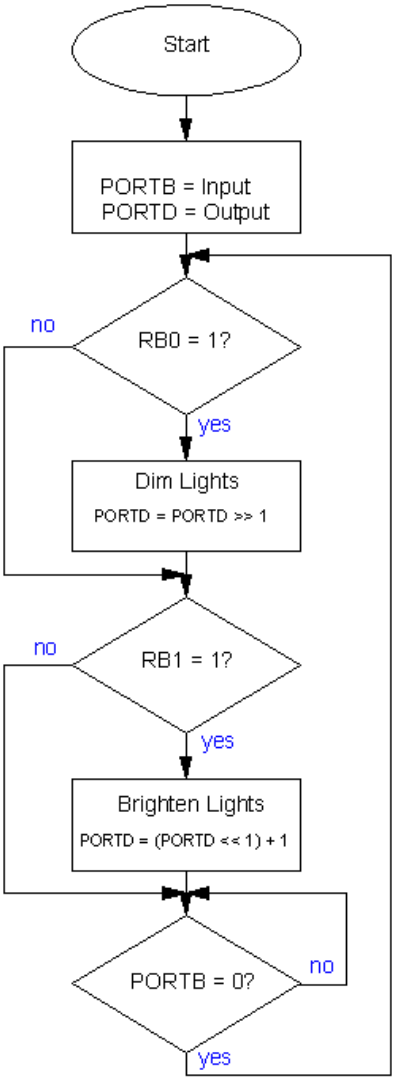
6) The flow chart below turns your PIC into an electronic flashlight

- RB0: Make the light dimmer
- RB1: Make the light brighter

Write the corresponding assembler code

```
        org     0x800

        movlw   0xFF
        movwf   TRISB
        clrf    TRISD
        movlw   0x0F
        movwf   ADCON1
L1:
        btfss   PORTB,0
        goto    L2
        rrncf   PORTD,W
        andlw   0x7F
        movwf   PORTD
L2:
        btfss   PORTB,1
        goto    L3
        rlncf   PORTD,W
        iorlw   1
        movwf   PORTD
L3:
        movlw   0
        cpfseq  PORTB
        goto    L3
        goto    L1
```




Problem #6