ECE 376 - Homework #2

Assembler, Flow Charts. Due Monday, January 25th

Please make the subject "ECE 376 HW#2" if submitting homework electronically to Jacob_Glower@yahoo.com (or on blackboard)

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

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

       movlw    4

       addwf    B,W
       addwf    B,W

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

       movwf    A
```

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

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

A      equ      0
B      equ      2
C      equ      4

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

       movlw    4
       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
```

3) Convert the following C code to assembler (traffic light controller: output green, yellow, red)

```
; unsigned char A, B;

A       equ       0
B       equ       1

; A = A + 1;

        incf      A,F

; if(A > 2) A = 0;

        movlw     2
        cpfsgt    A
        goto      L1
        clrf      A

L1:
; if(A == 0) B = 1;

        movlw     0
        cpfseq    A
        goto      L2
        movlw     1
        movwf     B

L2:
; else if(A == 1) B = 2;
        movlw     1
        cpfseq    A
        goto      L3
        movlw     2
        movwf     B
        goto      L4

; else B = 4;
L3:
        movlw     4
        movwf     B
L5:
        nop
```

4) Convert the following C code in to assembler

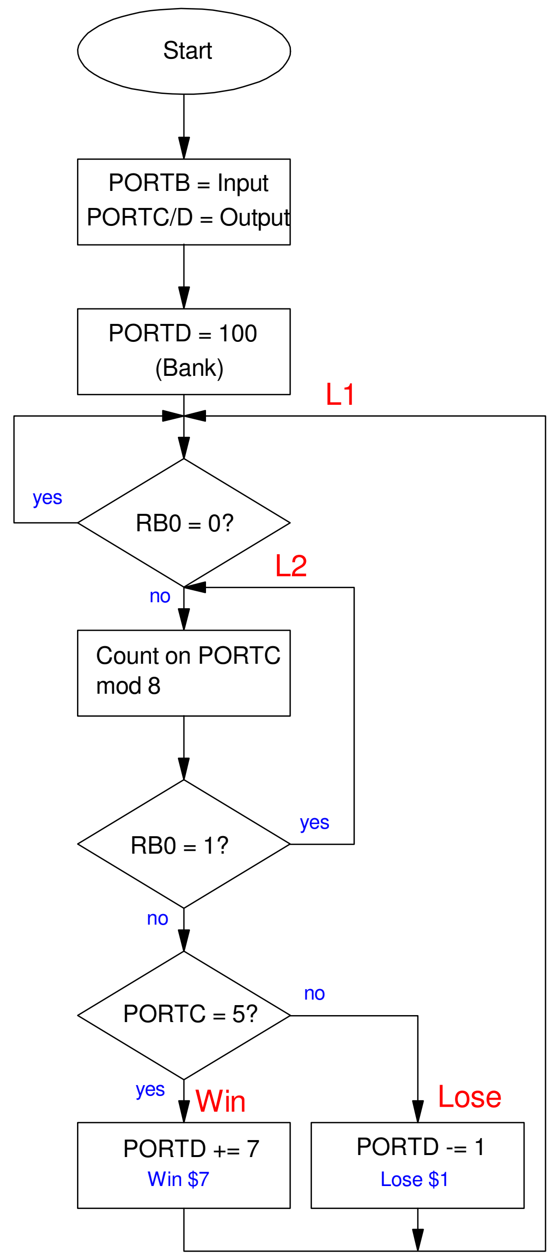
```
; unsigned char A, B, C;

A      equ      0
B      equ      1
C      equ      2

; A = 0;

       clrf     A

;while(A < 10) {
L1:
       movlw    10
       cpfslt   A
       goto     L2

;   B = B + C;

       movf     C,W
       addwf    B,F

;   A = A + 1;

       incf     A,F

;   }

       goto     L1
L2:
       nop
```

5) The flow chart below turns your PIC into an electornic slot machine:

- Press RB0 to play *RB0 is PORTB pin 0 (RB0 is the name for that pin in C code)*
- If the number 5 comes up (1 in 8 chance), you win \$7. Otherwise you lose \$1

Write the corresponding assembler code.

```
        org     0x800

        movlw   0xFF
        movwf   TRISB
        clrf    TRISC
        clrf    TRISD

        movlw   0x0F
        movwf   ADCON1

        movlw   100
        movwf   PORTD
L1:

        btfss   PORTB,0
        goto    L1

L2:
        incf    PORTC,W
        andlw   0x07
        movwf   PORTC

        btfsc   PORTB,0
        goto    L2

        movlw   5
        cpfseq  PORTC
        goto    Lose
Win:
        movlw   7
        addwf   PORTD,F
        goto    L1

Lose:
        decf    PORTD,F
        goto    L1
```



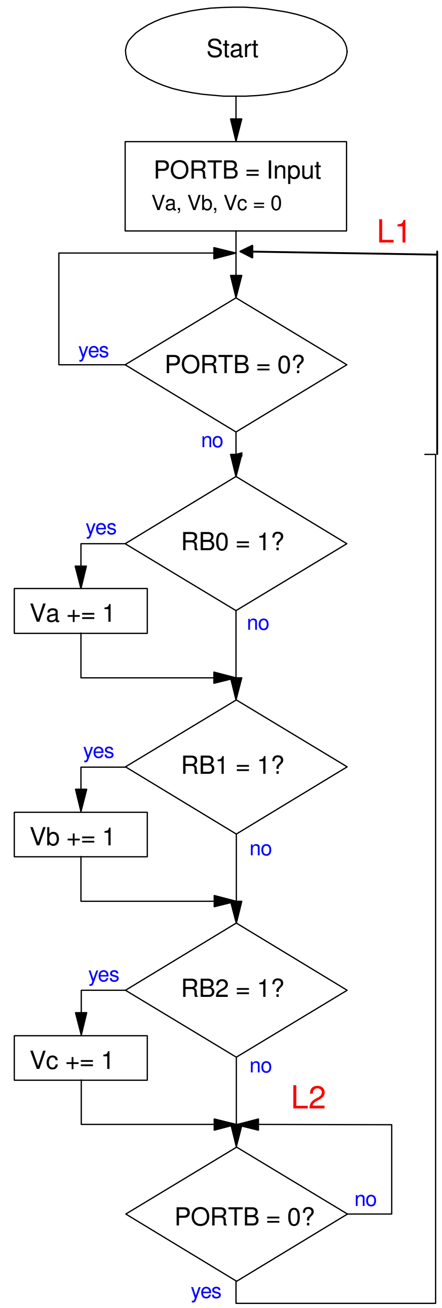
Problem #5
Slot Machine

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

- On reset, all votes are set to zero (Va = Vb = Vc = 0)
- When RB0 is pressed, one vote is counted for candidate A
- When RB1 is pressed, one vote is counted for candidate B
- When RB2 is pressed, one vote is counted for candidate C

Write the corresponding assembler code

```
Va      equu      0
Vb      equ       1
Vc      equ       2

        org       0x800

        movlw     0xFF
        movwf     TRISB
        clrf      Va
        clrf      Vb
        clrf      Vc
        movlw     0x0F
        movwf     ADCON1

L1:
        movlw     0
        cpfsgt    PORTB
        goto      L1

        btfsc     PORTB,0
        incf      Va,F

        btfsc     PORTB,1
        incf      Vb,F

        btfsc     PORTB,2
        incf      Vc,F

L2:
        movlw     0
        cpfseq    PORTB
        goto      L2

        goto      L1
```



Problem 6: Voting Machine