ECE 376 - Homework #6

LCD, Keypads, & Neopixels. Due Monday, October 9th

Design an embedded system with your PIC processor which includes

- An LCD display and
- A numeric keypad.

Complex RPN Calculator

1) Requirements.

Inputs:

- Numeric keypad
- 8 x Push Button on PORTB

Outputs

- LCD Display
- LED on RA1.

How they relate:

Display the stack as a complex number:

```
Y:    00000 + j00000
X:    00000 + j00000
```

Using the keypad, input a number from 00000 to 32000 (signed integers).

- RB7 changes whether you are editing the real (RA1=0) or complex part (RA1=1)

Once the numbers are input, you can manipulate them as follows:

- * Push
- # undo last numer (divide by 10)
- RB7 j. Switch between real and complex for data entry.
- RB3 multiply
- RB2 divide
- RB1 addition
- RB0 subtraction

When you press RB7, the LED on RA1 toggles.

- RA1=0 You're editing the real part of the number
- RA1=1 You're editing the complex part of the number

2) Resuting C Code along with an explanation for how it works.

```
Memory Summary:
    Program space        used  1280h (  4736) of 10000h bytes   (  7.2%)
    Data space           used    37h (    55) of   F80h bytes   (  1.4%)
    EEPROM space         used     0h (     0) of   400h bytes   (  0.0%)
    ID Location space    used     0h (     0) of     8h nibbles (  0.0%)
    Configuration bits   used     0h (     0) of     7h words   (  0.0%)
```

(main parts of the code)

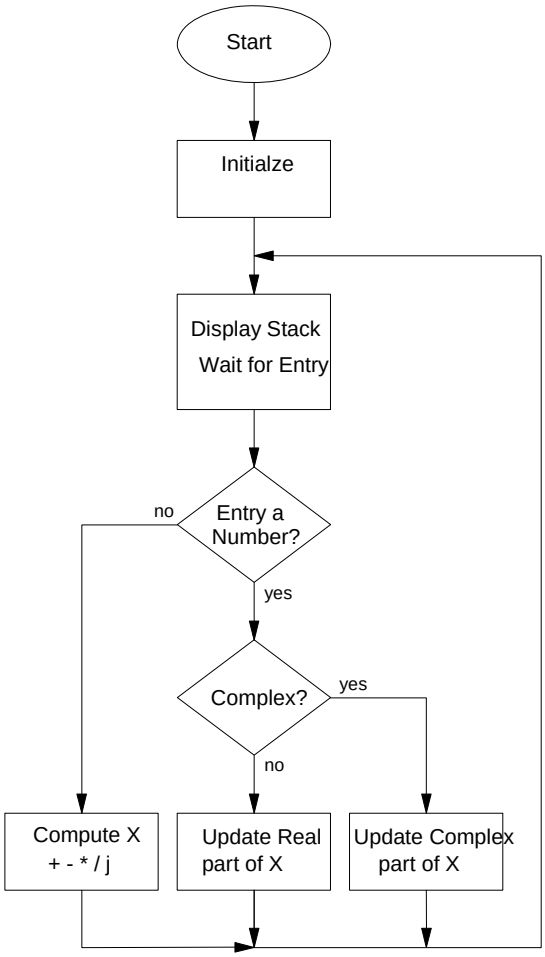
Output a complex number: Modified LCD_Out() routine

```
void LCD_Out2(int DATA, int iDATA)
{
   unsigned char A[5], i;

   if (DATA < 0) {
      LCD_Write('-');
      DATA = -DATA;
   }
   else LCD_Write(' ');

   for (i=0; i<5; i++) {
      A[i] = DATA % 10;
      DATA = DATA / 10;
   }
   for (i=5; i>0; i--) {
       LCD_Write(A[i-1] + '0');
   }

   if (iDATA < 0) {
      LCD_Write('-');
      iDATA = -iDATA;
   }
   else LCD_Write('+');
   LCD_Write('j');
   for (i=0; i<5; i++) {
      A[i] = iDATA % 10;
      iDATA = iDATA / 10;
   }
   for (i=5; i>0; i--) {
      LCD_Write(A[i-1] + '0');
   }
}
```

```
// Main Routine

void main(void)
{
   unsigned int i, j;
   int X, Y, Z, T, TEMP;
   int iX, iY, iZ, iT, iTEMP;
   int a, ia;

   TRISA = 0;
   TRISB = 0xFF;
   TRISC = 0xF8;
   TRISD = 0;
   TRISE = 0;
   TRISA = 0;
   ADCON1 = 15;

   LCD_Init();                // initialize the LCD

   LCD_Move(0,0);  for (i=0; i<20; i++) LCD_Write(MSG0[i]);
   Wait_ms(2000);
   LCD_Inst(1);

   X = 0;   iX = 0;
   Y = 0;   iY = 0;
   Z = 0;   iZ = 0;
   T = 0;   iT = 0;
   RA1 = 0;

   LCD_Move(0,0);  LCD_Write('Y');
   LCD_Move(1,0);  LCD_Write('X');
```

```
// Main Loop

   while(1) {

// Display teh stack
      LCD_Move(0,3);  LCD_Out2(Y, iY);
      LCD_Move(1,3);  LCD_Out2(X, iX);

      TEMP = ReadKey();

      if (TEMP < 10) {
         if(RA1) iX = (iX*10) + TEMP;
         else     X = (X*10) + TEMP;
         }

      if (TEMP == 10) {
         T = Z;    iT = iZ;
         Z = Y;    iZ = iY;
         Y = X;    iY = iX;
         X = 0;    iX = 0;
         RA1 = 0;
         }
      if (TEMP == 11) {
         if(RA1)  iX = iX/10;
         else      X = X / 10;
         }
      if (TEMP == 12) {   // Y - X
         X = Y - X;   iX = iY - iX;
         Y = Z;       iY = iZ;
         Z = T;       iZ = iT;
         RA1 = 0;
         }
      if (TEMP == 13) {   // Y + X
         X = X + Y;   iX = iX + iY;
         Y = Z;       iY = iZ;
         Z = T;       iZ = iT;
         RA1 = 0;
         }
      if (TEMP == 14) {   // Y / X
         a = (X*Y + iX*iY) / (X*X + iX*iX);
         ia = (X*iY - iX*Y) / (X*X + iX*iX);
         X = a;   iX = ia;
         Y = Z;   iY = iZ;
         Z = T;   iZ = iT;
         RA1 = 0;
         }
      if (TEMP == 15) {   // Y * X
         a = (X*Y - iX*iY);
         ia = X*iY + iX*Y;
         X = a;      iX = ia;
         Y = Z;      iY = iZ;
         Z = T;      iZ = iT;
         RA1 = 0;
         }
      if (TEMP == 19) {   // i
         RA1 = !RA1;
         }
      }
   }
```

3) Flow Chart


Start
Initialze
Display Stack
Wait for Entry
Entry a Number?
no
yes
Complex?
yes
no
Compute X + - * / j
Update Real part of X
Update Complex part of X

4) Validation: Collect data in lab to verify you met the requirements

Check each operation

Input a complex number:

```
       23 + j35        (enter)
+      17 + j22
------------------
```

Addition (RB1)

```
       23 + j35        (enter)
+      17 + j22
------------------
       40 + j57
```

Subtraction (RB0)

```
       23 + j35        (enter)
-      17 + j22
------------------
=       6 + j13
```

Multiplication

```
      23 + j35        (enter)
*     17 + j22
------------------
=   -379 + j1101
```

5) Demo