Lab #4 ECE271 Spring 2012 _____

NAME

This lab will give you more practice with: Assembler directives, Labels, CCR bits, Indexed addressing, Comments, and using TUTE's debug window. It will also introduce the use of breakpoints, using different comparison branches (other than BNE and BEQ), and compare accumulator instructions.

For Homework (do what you can before coming in)

You are to write an assembly language program that:

1. Uses an FCB assembler directive to put the following data bytes in memory starting at $0100
   $00, $45, $F4, $10, $54, $FC, $D9, $A0, $01, $C0, $01, $A0, $01, $90, $84, $8B,
   $01, $C0, $01, $A0, $01, $80, $94, $8B, $4C, $01, $C0, $01, $A0, $01, $90, $D9

NOTE: This data is in the file, LAB4data.txt, which has been posted on the class web page ("Lab section" on the assignments page), so you can copy it from there rather than typing it all in.

2. Ends with the number of bytes that are greater than $C0 in Accumulator A. Treat the numbers as being unsigned. (This code should start at $8800).
3. Terminates in the following statement: SELF BRA SELF
4. Uses labels throughout. There should be no numerical operands except in indexed addressing offsets and assembler directives like ORG, EQU, FCB, FDB, FCC, and RMB.

Has header and instruction comments including name, lab day, lab number, and date.

NOTES:

1. The program can be created using any text editor. However, in order to have TUTE be able to access it, save your program as a text only file with a .asm extension.
2. Reviewing the part of Section 2.4 dealing with Comparison Branch Instructions, pp. 93-96, might be helpful, also review the *compare accumulator to memory* instructions on pg. 90

In Lab (Turn in this sheet and **a listing** when you leave lab.)

1. Bring this lab sheet, your source program, your class notes, AND your TUTE instruction sheet
2. Connect an EVBU board OR use TUTE's simulator
3. Assemble and Load your program
4. Use TUTE's Debug window to display registers A, PC, and X. Also display CCR bits C, V, Z, and N. Display values of X and PC in hexadecimal and the value of A in decimal.
5. Set a breakpoint at the instruction in your program where you increment Accumulator A after you have found a byte that is greater than $C0. Then enter **Run→ Run** and record the register values each time you hit the breakpoint. Also record the value of the byte that got you to the breakpoint. You can use the value in X along with the table to determine this value.
6. Set a breakpoint at the BRA SELF and record the values of the registers and condition code bits when you first arrive at the SELF BRA SELF instruction.

NOTE: A values should be decimal and X and PC values hexadecimal

BrkPt Hit	Byte Tested	A	X	PC	C	V	Z	N
1st	_____	___	____	_____	__	__	__	__
2nd	_____	___	____	_____	__	__	__	__
3rd	_____	___	____	_____	__	__	__	__
4th	_____	___	____	_____	__	__	__	__
5th	_____	___	____	_____	__	__	__	__
SELF BRA SELF Values		___	____	_____	__	__	__	__

Do all the values in the above table make sense to you?