Lab #2b **ECE 271** **Simple Code 2**

You will use TUTE to: (1) create a source program with three separate routines, (2) assemble it, (3) load the resulting machine code into the EVBU's ram, and (4) execute each of the routines. These routines will make use of the LEDs and switches of the interface board. When you are sure that all your routines are working, print out a copy of your assembled program (this is called a listing) and have a monitor initial it. At the end of the lab period, hand in the initialed listing. **NOTE:** This will be the only program you will be asked to write where we will not require labels and comments!

1. Bring this lab sheet, your class notes, AND your TUTE instruction sheet to lab.
2. Open a serial communication path to the EVBU
3. Select the Comm Tab and then expand the Comm window so you can see what is happening in this window.
4. In the blue area, write your source program being sure to have three separate ORGs.
5. Save it as Lab 2.asm The .asm extension is important for TUTE to color code the text
6. Assemble your source program and, if you have no errors,
7. Send the machine code to the EVBU with **Run -> Load**
8. Set the start of execution to the first statement of Routine #1 by putting the cursor at this statement and then selecting the menu items **Run -> Set Execution Point**. This statement should then be bounded by a blue dotted box. If it isn't, repeat the previous steps of 6. again.
9. Execute your program by **Run -> Run**. This transfers total control to the 68HC11.
10. Once you are sure the routine is working the way you think it should, get control back by pressing the Reset button on the EVBU board.
11. Repeat steps 6 – 8 for Routines 2 and 3.
12. Record your measured time for 1 complete count cycle on the LEDs (all lights off to all lights off again).
13. Print out a copy of your assembled program.
14. Demonstrate your routines to a monitor.

Routine #1. This very simple code should read the switches ($1003) and write the contents to the LED's ($1004). After doing those operations, the program should branch back and do the same thing again. This code will form an endless loop which when started will terminate the BUFFALO communication channel. To get back to BUFFALO, you will need to press the reset button on the EVBU. Use an ORG statement to place the first instruction of this code at memory location $100. While the program is running, check the behavior of the LED's when you slide the switches, SW8 – SW1. Make sure you understand why they behave as they do.

Routine #2. In this program, have SW8-SW1 control the LED's PB7-PB0 as in Program #1 except, change the code so that when a switch is down, the LED it controls will turn on. Of course, when all switches are up (towards LEDs), then all LEDs will be off. Use an ORG statement to place the first instruction of this code at memory location $110.

Routine #3. Write code, ORG'd at $120, that will create a 16-bit countdown timer using the A and B registers arranged as a pair of nested counters. Every time the timer expires, it should increment $1004. When you get your code working, it should cause the LED's PB7 to PB0 to act as an 8 bit counter. Use a stop watch to measure one full count cycle of the LEDs, and for homework, use this time along with the number of clock cycles in your code, to calculate the frequency at which the microprocessor is executing instructions.