ELEC 5260/6260 Problem Set #3
Due Wednesday, 2/6/2013

<u>PROGRAM 1:</u>

To practice writing and using *subroutines*, write a subroutine that calculates the expression: Z = (a0 * x0) – (a1 * x1).

The <u>values</u> of the four variables a0, a1, x0, x1 are to be passed to the subroutine in registers. The <u>address</u> of variable Z is to be passed to the subroutine in a register.

The “main” program is to call the subroutine two times, once for each of the following sets of data. You may not use “immediate” data – all values are to be read from memory and results are to be stored in memory.

x0 and x1 are to be 32-bit integers.
a0, a1, and z are to be stored using Q24.8 format, i.e. 24 integer bits and 8 fraction bits, stored in a 32-bit word as follows: $n_{23}n_{22}n_{21\ldots}\ n_1n_0\ .\ n_{-1}n_{-2}\ \ldots\ n_{-7}n_{-8}$

First call: x0=200, x1=100, a0=5.25, a1=6.75.
Store the answer at variable z1.

Second call: x0=300, x1=200, a0=3.5, a1=4.125.
Store the answer at variable z2.

For convenience, define the arguments to be passed to the subroutine at the end of your code section, and the variables to be written in a data section. There should be eight “variables” in the code section – two sets of four arguments. There should be two variables in the data section.

You can test this with the simulator, but version to be submitted must be executed in RAM on the Discovery board. Submit a printout of your source program, and a screen image of the debug window, with the results highlighted (circled) in a memory window.

PROGRAM 2:

Given below is a C program that includes several function calls.

a. Write the equivalent routines in ARM assembly language and test the program in the Keil debugger. Submit two screen captures of the debug window, showing the final result returned for each of the two functions called by the main program. The main program and the three "functions" should be implemented as separate routines/subroutines as listed (i.e. do not "merge" them to optimize the program.)

b. Enter and compile the C program, as given, and compare the "listing file" to your hand-written assembly language program. Alternatively, you may view the disassembled code in the debugger window to see the C and equivalent assembly language instructions. Turn off any optimizations that would normally be performed by the C compiler.

This program can be done in the simulator or on the Discovery board.

```
int k;  //global variable

int f1(int x1, int x2) {
   return x1 + x2;
}

int f2(int x1) {
   return x1 + 1;
}

void f3(int r) {
   int j;
   for (j = 0; j < 2; j++)
        k = k + f1(r + j, 5);
}

void main () {
   int a;
   k = 0;
   f3(3);
   a = f2(2);
}
```