CS2208b Lab No. 4
Introduction to Computer Organization and Architecture

Monday March 7, 2016 (section 3 @ ***SSC-1032*** from 2:30 pm to 3:30 pm)
Tuesday March 8, 2016 (section 4 @ ***SSC-1032*** from 1:30 pm to 2:30 pm)
Tuesday March 8, 2016 (section 8 @ ***HSB-14*** from 3:30 pm to 4:30 pm)
Thursday March 10, 2016 (section 5 @ ***SSC-1032*** from 2:30 pm to 3:30 pm)
Friday March 11, 2016 (section 6 @ ***SSC-1032*** from 1:30 pm to 2:30 pm)
Friday March 11, 2016 (section 7 @ ***SSC-1032*** from 3:30 pm to 4:30 pm)

The objective of this lab is:

- To practice ARM assembly programming

If you would like to leave, and at least 30 minutes have passed, raise your hand and wait for the TA.
Show the TA what you did. If, and only if, you did a reasonable effort during the lab, he/she will give you the lab mark.

==========

PROBLEM SET

Before you start practicing this lab, you need to review and fully understand how to use *the Keil ARM Simulator*, as well as tutorial 6 *(Tutorial_06_ARM_Addressing_Modes.pdf).*

1. Translate the following tasks into a single ARM instruction:

   a. Add 32 times of the content of registers r0 and the content of r1 only if N is clear. Store the result in register r2

   b. Subtract the content of register r3 from 0x990, put the results in register r4 only if C is set and Z is clear.

   c. Clear the 2nd least significant byte of the content of register r5, i.e., store `(00000000)`$_2$ in it, and put the results in register r6. The result of the instruction must affect the value of the *Current Program Status Register* (CPSR).

   Test your answers by putting them in a program, which starts by assigning values to r0 and r1 and then comparing them together to set/clear the flags in such a way to test both cases of (a), (b), and (c). Note that, you will need two sets of r0 and r1 values for each case.

   Hint: you may want to consider Table 3.2 in the textbook.

2. Encode *by hand* the instructions that you suggested in Question 1, i.e., generate the 4 byte machine language of each ARM instruction. Verify your answers using Keil's simulator.
   Hint 1: you may want to consider Figure 3.26 in the textbook.
   Hint 2: the Op-Code of the `ADD` instruction is `(0100)`$_2$, the Op-Code of the `SUB` instruction is `(0010)`$_2$, the Op-Code of the `RSB` instruction is `(0011)`$_2$, the Op-Code of the `AND` instruction is `(0000)`$_2$, and the Op-Code of the `BIC` instruction is `(1110)`$_2$.
   Hint 3: the code for the `LSL` shift type is `(00)`$_2$

3. Convert the GCD algorithm given in this flowchart into

   a. ***Traditional*** assembly, where only branches can be conditional, i.e., do not utilize the ARM conditional execution feature.

   b. ARM assembly, where all instructions are conditional, thus improving code density.

   PS: The only instructions you need are `CMP`, `B` and `SUB`.

   Test your code by assigning various values to `r0` and `r1`.