CS2208b Assignment 5

Issued on: Thursday, March 24, 2016

Due by: 11:55 pm on Thursday, March 31, 2016

For this assignment, only an electronic submission (***attachments***) at owl.uwo.ca is required.

- Attachments must include:
  - ***ONE pdf*** file that has the flowchart, program documentations, and any related communications.
  - ***Text*** soft copy of the assembly program that you wrote for each question (*one program attachment per question*), i.e., ***ONE assembly*** source file in total.
- So, in total, you will submit 1 + 1 = 2 files.
- **Failure to follow the above format may cost you 10% of the total assignment mark.**

Late assignments are strongly discouraged

- 10% will be deducted from a late assignment (up to 24 hours after the due date/time)
- After 24 hours from the due date/time, late assignments will receive a zero grade.

In this assignment, you will use the *micro Vision ARM simulator* by *Keil*, which is an ***MS Windows*** based software, to develop the required programs in this assignment. The simulator (version 4) has been installed on all PCs at SSC-1032 and HSB-14 labs.

The *Keil micro Vision* simulator may also be installed on your Windows PC. You just need to download it from OWL and install it.

Programming style is very important in assembly language. It is expected to do the following in your programs:

- Using macros for the constants in your program to make it more readable.
- Applying neat spacing and code organization:
  - Assembly language source code should be arranged in three columns: *label*, *instruction*, and *comments*:
    - the *label* field starts at the beginning of the line,
    - the *instruction* field (opcodes + operands) starts at the next TAB stop, and
    - the *comments* are aligned in a column on the right.
- Using appropriate label names.
- Commenting each assembly line

Great Ways to Lose Marks

- Not grouping your lines into logical ideas
- Not using any whitespace at all
- Not bothering to comment
- Commenting the code by just stating what you're doing, instead of why, e.g.,

```
MOV r0, #5    ;move 5 into r0
```

- Not paying attention to the programming style (see the previous paragraph)
- Handing it in as soon as it assembles without testing and/or trying to break your code

QUESTION 1 (100 marks)

Recursion is a method where the solution to a problem depends on solutions to smaller instances of the same problem. A function is considered recursive if it calls itself.

The following function computes x^n recursively, where x is an integer number and n is a non-negative integer number.

```
int power(int x, unsigned int n)
{
  int y;

  if (n == 0)
    return 1;

  if (n & 1)
    return x * power(x, n - 1);
  else
  { y = power(x, n >> 1);
    return y * y;
  }
}
```

Draw ***a detailed flowchart*** and write an ARM assembly ***program*** to calculate x^n ***using the above recursive function***, where n is passed-by-value through the stack to the function and the returned value is stored in the stack just above the parameter. No other registers may be modified by the `power` function. Once the control is *completely* returned back from the function (i.e., after calculating x^n), the returned value must be stored in a local variable (called *result*) in the main function. ***Your code should be highly optimized, i.e., use as little number of instructions as possible.***

You should utilize a big enough stack so that you can calculate x^n for various n values.

How many stack frames are needed to calculate x^n, when n = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, and 12?