UNIVERSITY OF CALGARY
DEPARTMENT OF COMPUTER SCIENCE
Winter 2021
CPSC 355: Computing Machinery I

Assignment 02

Weight: 8.33% of the final grade
Due: Feb 14th (11:59 PM)

Basic Assembly Language Programming

Create an ARMv8 A64 assembly language program that finds the maximum of $y = -5x^3 - 21x^2 + 4x + 16$ in the range $-5 \leq x \leq 6$, by stepping through the range one by one in a loop and testing.

Use only long integers for *x*, and do not factor the expression. Use the `printf()` function to display to the screen the values of *x*, *y* and the current *maximum* on each iteration of your loop.

You are to create 2 versions of your program:

1. Write the program without macros (i.e., do not use *m4*), and use only the `mul`, `add`, and `mov` instructions to do your calculations. Use a pre-test loop, where the test is at the top of the loop.
2. Optimize the above program by putting the loop test at the bottom of the loop (make sure it is still a pre-test loop), and by making use of the `madd` instruction. Also, add macros to the above program to make it more readable (use *m4*). In particular, provide macros for heavily used registers.

Running Your Program

To verify that your assembly language program works, run both versions under *gdb*, capturing output from each session using the *script* UNIX command.

For version 1, single step through the program (use `ni`) for at least one iteration of your loop, displaying the instruction being executed (use `display/i $pc`). Also print out the contents of particular registers (use `p`) at key points in your program to show that it is working as expected.

For version 2, set a breakpoint just after the place where the final result is calculated, and then print out the *maximum*. Do not single step through this version.

Other Requirements

Make sure your code is properly formatted into columns, is readable and fully documented, and includes identifying information at the top of each file. You must comment each line of assembly code. Your code should also be well designed: make sure it is well organized, clear, and concise.

New Skills Needed for this Assignment:

- Ability to work with basic arithmetic, loops, and if-else constructs in assembly.
- Ability to print to standard output using the `printf()` function.
- Ability to optimize assembly code by rearranging loops and using alternate instructions.
- Ability to use macros in assembly code.
- Ability to assemble programs using *gcc* and use *m4* to process macros.
- Ability to use *gdb* to debug and display assembly language programs.

Submission

- Name your programs ***assign2a.s*** and ***assign2b.asm***, and your scripts ***script1.txt*** and ***script2.txt***.
- Submit your assembly source code files for **both** programs and **two** scripts to the appropriate Dropbox on D2L.
- Your TA will assemble and run your programs to test them.

Late Submission Policy

The Late policy for assignments is as follows:

- 20% grade penalty for 1 calendar day of lateness.
- 40% grade penalty for 2 calendar days of lateness.
- A grade of zero for more than 2 days of lateness.

Academic Misconduct

This assignment is to be done by individual students: your final submission must be your own original work. Teamwork is not allowed. Any similarities between submissions will be further investigated for academic misconduct. While you are encouraged to discuss the assignment with your colleagues, this must be limited to conceptual and design decisions. Code sharing by any means is prohibited, including *looking* at someone else's paper or screen. The submission of the compiler-generated assembly code is absolutely prohibited. Any re-used code of excess of 5 lines in C and 10 lines in assembly (10 assembly language instructions) must be cited and have its source acknowledged. Failure to credit the source will also result in a misconduct investigation.

D2L Marks

Marks posted on D2L are subject to change (up or down).

Marking Criteria

Functionality (Version 1)		
Equation calculation	4	_____
Test for maximum	4	_____
Display to screen using printf()	2	_____
Loop	4	_____
Optimization (Version 2)	3	_____
Use of Macros (Version 2)	3	_____
2 Scripts showing use of *gdb*	4	_____
Complete documentation and commenting	4	_____
Design quality	2	_____
Total	**30**	**_____**