C Programming Language Refresher Module for Operating Systems

December 28, 2019

Assignment 0.2 (Total points:)

Due date: January 4, 2019. Time: 23:59 Hrs. (Hard Deadline)

1 Combining C and Assembly Language Programs

This second exercise is aimed to serve two objectives – writing assembly language programs and secondly to help combine a C program with and assembly languge function.

You need to do the following:

1. Write a program that takes two integers as user inputs. Lets call it prog-add.c.
2. The program calls a function `add`, that is written in assembly languge (x86-64). The program takes two arguments from the stack – *viz.* the two integers that are passed from the `main()` function of `prog-add.c`.
3. The `add` function takes two integer arguments and add them. Thereafter it prints the sum of the two integers on the screen, using the assembly language instructions that calls the `write()` system call. The instruction mostly involves a software interrupt (trap) with arguments passed via registers that emulates the system call. You need to `write()` the sum to the descriptor corresponding to `stdout`.
4. After having printed the sum of the integer, the final step of the `add` function would be to terminate the program using the `exit()` system call. The `exit()` system call is called through the similar sequence of assembly instructions (and traps).

You would require to refer to the `manpages` for `gcc` and `nasm` for the same.

What To Submit

- Program source code with Makefile to compile and pause the compilation at each phase.

- Write-up describing the following:
  - Gcc command-line options that pauses the compilation of the program at each step, along with their descriptions.
  - Description the outcomes of each step involving the description of the output file.

Grading Rubric

- Successful compilation of the C and ASM program via the Makefile – 10 points.
- Correct output for the program – 20 points.
- Description of how the program works, *viz.* how the C program calls the ASM routine and how the routine takes the arguments and prints the output – 20 points.