CSE 2312 Programming Assignment 1 – Fall 2021

Programming Lab Policies:

- Labs that fail to compile, or do not terminate correctly, will receive a zero.
- Labs that fail to compile, or do not terminate correctly, may not be resubmitted for a grade. This includes instances where students did not upload the correct file for grading.
- All labs must be submitted via the upload link on Canvas. Labs are not accepted through e-mail or Canvas Messenger.
- **Students must make a credible attempt to pass all programming labs to receive a passing grade in the course.**

Write ARM assembly implementations for all of the functions below. The functions must be present in a single .s file. Your function/procedure names must be identical to that presented below, as your implementations will be tested with generic C code used by the TAs.

All of your functions must return a value such that the program will run to completion with no segmentation faults. If a function cannot be successfully implemented, it still must return a valid value: **no function may be omitted**. Attempting to omit a function will result in a compile error.

A sample driver is included with this lab for student convenience. It is the student's responsibility to test their program with a sufficient range of values to ensure their code functions correctly. Programs will be tested with different values than are used in the sample driver.

Submit your assignment via the submission link on Canvas. The name of this file should be **lab#_lastname_loginID.s**. Example: If your name is John Doe and your login ID is jxd1234, your submission file name must be "lab#_Doe_jxd1234.s". Labs uploaded as a zip file, or labs that require a custom makefile, will receive a zero.

All questions worth nine points, unless otherwise noted.

1. uint64_t add64 (uint32_t x, uint32_t y) // returns x + y; worth 10 points

2. uint64_t sub64 (uint64_t x, uint64_t y) // returns x - y

3. int16_t minS16(int16_t x, int16_t y) // returns the minimum of x, y

4. uint8_t minU8(uint8_t x, uint8_t y)// returns the minimum of x, y

5. bool isLessThanU32(uint32_t x, uint32_t y) // returns 1 if x < y, 0 else

6. bool isLessThanS8(int8_t x, int8_t y) // returns 1 if x < y, 0 else

7. uint16_t shiftLeftU16 (uint16_t x, uint16_t p) // returns x << p = x * 2^p for p = 0 .. 8

8. uint16_t shiftU16(uint16_t x, int16_t p) // return x * 2^p for p = -16 .. 16

9. int8_t shiftS8 (int8_t x, int8_t p) // return x * 2^p for p = -8 .. 8

10. bool isEqualS16(int16_t x, int16_t y) // returns 1 if x = y, 0 if x != y

11. bool isEqualU32(uint32_t x, uint32_t y) // returns 1 if x= y, 0 if x != y

All available conditionals are on page 1-19 (39) of the ARM 7 Reference Manual. Keep in mind that signed and unsigned integers may require different conditionals.