CS 251, Spring 2023, Assignment 0

Due Friday, May 19, 10:00 PM
No lates accepted

You are required to read, complete and sign, and submit (as well as follow) the following statement of Academic Integrity.

Statement of Academic Integrity for CS 251 Spring 2023, **Assignment 0**
I declare the following statements to be true:

- I have not used any unauthorized aids.
- I recognize that while I can discuss the questions in this assignment on Piazza and other forums with the instructors and with other students in the class, the write up that I am submitting is my own.
- I am aware that misconduct related to course work can result in significant penalties, including failing the course and suspension (this is covered in Policy 71:)

  `https://uwaterloo.ca/secretariat/policies-procedures-guidelines/policy-71`

Student Name:

UW ID#:

Signature:

Date:

The purpose of this assignment is to explore the ARM instructions presented in class. We provide a subset of nine (9) core instructions, which themselves are subsets of the simplified LEGv8 introduced in the textbook.

Coverage material for this assignment can be found on the the course website as:

- ARM overview,
- ARM reference, and
- Introduction to ARM in Lecture Notes,

Future assignments will focus on the implementation of this core set of instructions. For this assignment, make note of the following details:

- Use only the following ARM instructions: `ADD, SUB, ADDI, SUBI, B, CBZ, CBNZ, LDUR, STUR`.
- All instruction addresses are in decimal.
- Registers **may not** be assumed to be 0 initially(except for XZR or X31).
- Your solutions should use minimal number of ARM assembly instructions.
- You may use any other registers as temporary registers, but be sure to state their purpose at the bottom of your solution
- You may not need all the lines that are provided in the answer space. On the other hand, if you need additional lines, please add them.

Read the A0 Official Post on piazza. In this post, we will include all A0 assignment related information such as

- updates/corrections made to the assignment
- remark request due date
- FAQs

1. (5 pts) Write an ARM program that assigns 12, 43 and -2 to registers X0, X1 and X2, respectively. It should then assign the sum of the three numbers to register X4.

   Your code starts at PC=100.

100
104
108
112
116
120
124

2. (6 pts) Consider the following high-level pseudo-code, where `x2` and `x3` are variables:

```
if (x2 == x3)
   x3 = x3 + x2
else
    x3 = x3 - x2
x4 = x3
```

Write the corresponding ARM program starting at instruction address 36 and assume registers X2, X3 and X4 hold the value of the variables x2, x3 and x4, respectively.

36
40
44
48
52
56
60
64

3. (5 pts) Consider the following high-level pseudo-code, where `a` and `b` are integer arrays of 100 elements:

```
b[8] = a[0]
```

Write the corresponding ARM program starting at instruction address 40. Assume

- Registers X4 and X5 hold the value of the base address of arrays `a` and `b`, respectively.
- You may assume that both X4 and X5 are multiples of 8.
- All the elements of the arrays are in memory and updated elements of any array should be written back to memory.

40	
44	
48	
52	

4. In this question, we will compare the performance of two ARM programs that achieve the same functionality. Consider the following high-level pseudo-code:

```
sum = 0;
for k = 0 to 9
    sum = sum + a[k]
end for
```

Assume the following:

- Program starts at instruction address 40
- Registers X2 and X3 will hold the values for `sum` and `k`, respectively.
- Register X4 holds the value of the base address of the integer array `a`.
- Array `a` has length of 100 elements.
- Make sure that the base address for array `a` is unchanged (X4 should be preserved).
- All the elements of the array are in memory and updated elements of the array, if any, should be written back to memory.

(a) (4 pts) Write the corresponding ARM program.

40	
44	
48	
52	
56	
60	
64	
68	
72	
76	
80	

(b) (4 pts) The high-level pseudo-code was rewritten as follows:

```
sum = 0;
for k = 9 to 0
    sum = sum + a[k]
    write sum to MEM     //after the last element of a
end for
```

Write the corresponding ARM program starting at instruction address 36 keeping the same assumptions as above and including the assumption that `sum` will be written to memory (MEM) **immediately after the last element of the array** `a`.

36	
40	
44	
48	
52	
56	
60	
64	
68	
72	
76	

(c) (3 pts) Give the execution times below, compute the total clock cycles required to execute the program in part a and in part b. Which program requires less clock cycles? You may assume that I-format instructions are considered as R-format(arithmetic) instructions in this question.

Instruction Type	Clock Cycles (cc)
R-format (arithmetic) or I-format (immediate)	2
D-format (load/store)	6
B- or CB- format (branch)	3

5. In this question, you will explore how to write ARM assembly programs using **only** the core instructions introduced in class and explicitly allowed in this assignment.

   (a) (8 pts) Write an ARM assembly program that will multiply two numbers by using repeated addition. For example, to achieve $4 \times 2 = 8$, the program should execute $4 + 4 = 8$ or $2 + 2 + 2 + 2 = 8$. Your program should write the result of the multiplication to memory (MEM) at byte address `80`. You can make the following assumptions:

   - The first and second operands of the multiplication operator are in registers X2 and X3, respectively.
   - PC will start at address 0
   - Your program should work for all multipliers and multiplicands that are zero or a positive integer.

   You may use any other registers as temporary registers, but be sure to state their purpose at the bottom of your solution.

0	
4	
8	
12	
16	
20	
24	
28	
32	
36	

Purpose of temporary registers used if any:

(b) (10 pts) Write an ARM assembly program to raise a positive integer a to a given power of b. That is, your program should compute a^b, where the base, a, is an integer $a > 0$ and the power, b, is such that $b \geq 0$. You may find it useful to use repeated addition to achieve the desired multiplication. Your program should write the result of exponentiation to memory (MEM) at byte address `80`. You can make the following assumptions:

- The base and power are in registers X10 and X15, respectively.
- PC will start at address 0

You may use any other registers as temporary registers, but be sure to state their purpose at the bottom of your solution.

0	
4	
8	
12	
16	
20	
24	
28	
32	
36	
40	
44	
48	

Purpose of temporary registers used if any: