Computer Architecture
CS 325 - ON40
Department of Physics and Computer Science
Medgar Evers College
Exam 3

Direction: Submit your typed work in the Exams directory of your github repository and/or as an attachment on Google classroom under the Exam03 assessment. All submissions should have their appropriate extensions.

Problem	Maximum Points	Points Earned
1	5	
2	5	
3	5	
4	5	
Total	20	

A CSM computer has 20-bit words and instructions in the format

`[opcode|operandX|operandY]`

where opcode is 4 bits and each operand is a memory reference that is 8 bits. The instruction commands list for the computer are

Opcode	Description
0	Halts the program
1	Adds M(X) and M(Y) and puts the result in X
2	Subtracts M(Y) from M(X) and puts the result in X
3	Multiples M(X) and M(Y) and puts the least significant bits of the result in X and most significant bits in AC
4	Divides M(X) by M(Y) and stores the quotient in X and the remainder in AC
5	Transfers M(X) to AC
6	Transfers contents from AC to X
7	Transfers M(X) to Y
8	Makes next instruction X
9	If contents of AC $\geq$ M(X), makes next instruction Y
A	If contents of AC $\leq$ M(X), makes next instruction Y
B	If contents of AC $\geq$ 0, makes next instruction X; otherwise, makes next instruction Y
C	Increments M(X) by 1
D	Decrements M(X) by 1
E	Transfers M(M(X)) to Y

where instructions that work with a single operand sets the second operand to 00, and the function M(X) implies the contents of memory address X; hence opcode E implies transfer the contents of the memory address that is the contents of memory address X.

1. Write a program that stores the maximum value of the memory addresses 64 through 6E in the memory address 32.

   Hint: Start by storing the contents of 6E in 32 and loop.

2. Given that the contents of memory location `R` is `(R) = 1011011011000100`, perform the following logic operations

   a. 4 bit arithmetic left shift of `R`

   b. 6 bit logical right shift of `R`

   c. 8 bit cyclic left shift of `R`

   d. 5 bit logical left shift of `R`

   e. 7 bit arithmetic right shift of `R`

3. For each of the following numbers, write it and its negation in 8-bit sign-magnitude and twos-complement formats.

   a. 20

   b. -71

   c. -68

   d. 91

   e. 114

4. For each of the following pair of operands perform both twos complement addition and subtraction [the second operand must be subtracted from the first operand], and state if there was overflow. You must show work to receive full credit.

   a. `(R1) = 0111010101000100; (R2) = 00100000`

   b. `(R1) = 1111010110010000; (R2) = 001011100100`

5. **Extra Credit**

   In the provided cpp file, write the function `Addition()` whose header is

   ```
   bool Addition(bool a[],bool b[],bool r[],int n)
   ```

   Given that n is the size of all the array parameters, the function assigns r the twos complement sum of a and b; and then, returns true if overflow occurs or false if it does not occur. The sign bit and the most significant bit are indices 0 and 1 respectively. For instance, if a `= [0,1,1,0]`(6) and b `= [0,0,1,1]`(3), the function call will return true and r `= [1,0,0,1]`. Furthermore, the definition can only use bool variables and arrays except for int variables used to traverse the arrays. And, it cannot call any functions. Including any additional libraries to the cpp file and not following the constraints will disqualify this extra credit.

   Hint: Use bitwise operations.