**Deadline: December 11, 2023 at 23:59**

---

**Note:** You are required to use the LaTeX template provided on Brightspace.

# Question 1 (10 points)

A program using LC-3 machine code has been written and stored in memory at locations `x3000` to `x3008` as depicted in Table 1.

| Address | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| x3000 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| x3001 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| x3002 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| x3003 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| x3004 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| x3005 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| x3006 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |
| x3007 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| x3008 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |

Table 1

a) What is the functionality of this program?

b) What are the contents of the registers in Table 2 in hexadecimal notation (base-16) after the instruction at location `x3004` is executed for the first time, and at the end of the program?

| | PC | R1 | R2 | R3 | R4 | R5 |
|---|---|---|---|---|---|---|
| x3004 is executed the 1st time | ? | ? | ? | ? | ? | ? |
| At the end of the program | ? | ? | ? | ? | ? | ? |

Table 2

# Question 2 (5 points)

The memory addressability is 64 bits. What does this tell you about the size of the MAR and MDR?

# Question 3 (15 points)

Assume that we have a 32-bit computer architecture. The Instruction Register (`IR`) in this architecture is a 32-bit and its format as follows:

| OPCODE | DR | SR1 | SR2 | UNUSED |
|---|---|---|---|---|

where DR is the Destination Register, SR1 is Source Register 1, and SR2 is Source Register 2. Assuming that there are 300 opcodes, answer the following questions.

a) What is the minimum number of bits required to represent the OPCODE?

b) What is the maximum number of registers in this architecture? Motivate your answer.

c) If we use the maximum number of registers for DR, SR1 and SR2 fields in the IR format, what is the number of UNUSED bits in this architecture? Clarify your answer.

## Question 4 (20 points)

a) How might one use a single LC-3 instruction to move the value in R2 into R3?

b) The LC-3 has no subtract instruction. How could one perform the operation R1 ← R2 - R3 using only three LC-3 instructions?

c) Using only one LC-3 instruction and without changing the contents of the register, how might one set the condition codes based on the value that resides in R1?

d) Is there a sequence on LC-3 instructions that will cause the condition codes at the end of the sequence to be both N=1, Z=1 and P=0? Explain.

e) Write the LC-3 instruction that clears the contents of R4.

## Question 5 (5 points)

In Figure 1, the content of some of the memory locations is provided. The content of the relevant LC-3 registers are also shown. What are the contents of register R1 and the NZP flags after the instruction pointed by the PC is executed?

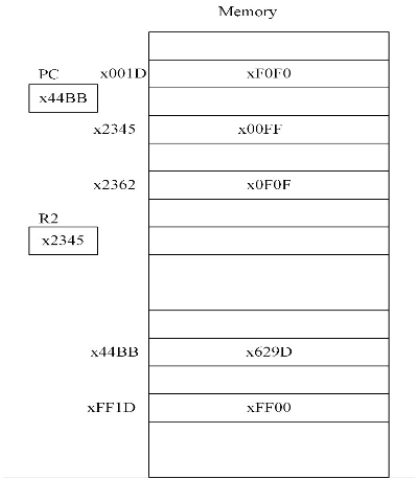


Figure 1

## Question 6 (10 points)

What is the functionality of the program shown in Table 3?

| | | |
|---|---|---|
| 1001 | 100 | 011 111111 |
| 0001 | 100 | 100 1 00001 |
| 0001 | 001 | 100 0 00 010 |
| 0000 | 010 | 000000101 |
| 0000 | 100 | 000000001 |
| 0000 | 001 | 000000010 |
| 0001 | 001 | 011 1 00000 |
| 0000 | 111 | 000000001 |
| 0101 | 001 | 0101 00000 |
| 1111 | 000 | 00010 0101 |

Table 3

## Assembly Programming (35 points)

The purpose of this part of the assignment is to familiarize you with the basics of LC-3 assembly language programming, the LC3Tools simulator and rudimentary debugging.

### Preparation

Make sure you have LC3Tools installed. If you have not done this yet, or do not know how to do this, consult the LC3 guide on Nestor.

As you might have noticed, there is another file on Brightspace called `ca2324_hw3_skeleton.asm`. This file will serve as a template for this assignment. Open the file in LC3Tools.

## Description

In this assignment you will closely monitor the contents of all registers `R0` to `R7` while running the code we provided below. At this point, the provided code template should be opened in LC3Tools. We highly recommend you to type over the code into the template, rather than copy pasting it.

You can test if you have written the code correctly by submitting the .asm file to Themis (at LC-3: Basic Multiplication) and check if it passes.

Once you are done writing the code, press the "Assemble" button at the top right. As explained in Tutorial 2, you can now run the code line-by-line using the "step in" button, or run it normally using the "run" button.

In the solution template of this assignment, you will find a table. This table should be filled in and contain the following rows:

- One row for the state of the registers before the loop starts
- One row for the state of the registers **after each iteration** of the loop

**Before you run the code, try to answer this question:** how many rows do you think you will end up with in your table?

After you are done running the code, make sure that you fill in the table in the template (obviously reporting the **actual** values from **ALL** eight registers, for **ALL** iterations of the loop)!

```
            .ORIG x3000          ;Program begins here
;-------------
;Instructions
;-------------
            AND R3, R3, #0       ;R3 <- 0
            LD  R1, DEC_6        ;R1 <- 6
            LD  R2, DEC_12       ;R2 <- 12

DO_WHILE    ADD R3, R3, R2       ;R3 <- R3 + R2
            ADD R1, R1, #-1      ;R1 <- R1 - 1
            BRp DO_WHILE         ;if (LMR > 0) goto DO_WHILE

            HALT                 ;Terminate the program
;-----
;Data
;-----
DEC_6       .FILL   #6  ;Put the value 6 into memory here
DEC_12      .FILL   #12 ;Put the value 12 into memory here

            .END
```

Code 1: This program `R3` ← `12 * 6`.