CS 2110 — Quiz 4, Group A
February 2018

Full name: ____________________
GT username: ____________________
This quiz is worth a total of **100 points**.

You are allowed to use one sheet of scrap paper. Feel free to request scrap paper from your Teaching Assistants. **Please make sure that all of your answers are contained within the answer boxes or the fill-in lines.** Do not write your work in the answer boxes, **keep all of your work on your scrap paper**. You will **NOT** be given credit for just showing work. Having anything except the answer inside the boxes or above the fill-in lines reduces autograder performance and might cause incorrect results. **Make sure to write your name, username, and answers legibly. You will not receive credit for illegible answers.**

Assembling Code

1. The left-side box contains the code for a valid assembly program. Suppose that the program is assembled and uploaded onto the memory of an LC-3 computer. **You should NOT execute / trace the program.** For lines **2, 3, 6, 9 and 11**, calculate what address in memory these instructions would end up being loaded to, and what binary value would be placed at these addresses. The bits are ordered from most significant to least significant, as usual. *Hint: the assembled values of the instructions.* 25

```
1   .orig x3000
2       BGN   LD R7, NUM
3             BRzp FIN
4
5       ABS   NOT R7, R7
6             ADD R7, R7, 1
7             ST R7, NUM
8
9       FIN   HALT
10
11      NUM   .fill #-1
12  .end
```

Line #	Address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
2	x3000	0	0	1	0	1	1	1	0	0	0	0	0	0	1	0	1
3	x3001	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	1
6	x3003	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	1
9	x3005	1	1	1	1	0	0	0	0	0	0	1	0	0	1	0	1
11	x3006	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Tracing a Program

2. Trace the left-side program and record the contents of the PC, R0, R1, and the CC in the right-side table. Assume the PC starts at 0x3000, R0 = 1, R1 = -1, and CC = z. Write HALT in the program counter when the HALT instruction is executed. You may or may not need all of the rows in the table. **Do NOT fill the rows after the execution of HALT.** 30

   Write the PC in hexadecimal and R0 and R1 in decimal. On rows you use, fill in all boxes regardless of whether or not their values are changed.

   Hint: ADD, AND and NOT change the condition codes.

Label	Address	Instruction
	x3000	ADD R0, R0, #-5
	x3001	ADD R0, R1, R0
	x3002	NOT R1, R1
	x3003	BRp DONE
	x3004	AND R1, R1, 0
DONE	x3005	HALT

Instruction #	PC	R0	R1	CC
INITIAL	x3000	1	-1	z
1	x3001	-4	-1	n
2	x3002	-5	-1	n
3	x3003	-5	0	z
4	x3004	-5	0	z
5	x3005	-5	0	z
6	HALT			
7				

Addressing Modes

3. The assembly program below uses different addressing modes to load data into registers. Suppose that the program is executed, and fill in the blanks in the right-side table according to the values of the registers after the program's execution. 28

```
.orig x3000
        LD R0, MEM1        ; x3000
        LEA R1, MEM2       ; x3001
        LDI R2, MEM3       ; x3002
        LDR R3, R1, 1      ; x3003

        HALT               ; x3004

   MEM1  .fill x3001       ; x3005
   MEM2  .fill xCAFE       ; x3006
   MEM3  .fill x3006       ; x3007
.end
```

Register Name	Final Hexadecimal Value
R0	x3001
R1	x3006
R2	xCAFE
R3	x3006

Pseudocode to Assembly

4. The code below is run on an LC-3 that has a positive number in its register $R1$ and a valid ASCII character in its register $R0$. Fill in the box with valid assembly code so that the completed program, when run, will print R1 copies (i.e. the number of copies should be equal to the number originally in R1) of the character in R0 and terminate. 17

```
.orig x3000
    ; Assume R1 is some positive number (R1 > 0)
    ; Assume R0 has some valid ASCII character value

    ; Convert the pseudocode below to assembly:
    ; while (R1 > 0) {
    ;     print(R0)
    ;     R1 = R1 - 1
    ; }

    ; START OF YOUR CODE
```

```
LOOP   OUT
       ADD R1, R1, #-1
       BRp LOOP

       HALT
```

```
    ; END OF YOUR CODE
.end
```