Homework 6

Student

Aaryan Vinay Potdar

Total Points

100 / 100 pts

Question 1

Overview   **0** / 0 pts

✔ **+ 0 pts** Correct

+ 0 pts Incorrect

Question 2

Basic Assembly & Pseudo-Ops   **9** / 9 pts

2.1 ## True/False   **3** / 3 pts

✔ **+ 3 pts** Correct (True)

+ 0 pts Incorrect

2.2 ## Trap Vectors   **3** / 3 pts

✔ **+ 3 pts** Correct (OUT)

+ 0 pts Incorrect

2.3 ## True/False   **3** / 3 pts

✔ **+ 3 pts** Correct (False)

+ 0 pts Incorrect

Question 3

LC-3 Assembly - Coding   **17** / 17 pts

3.1 ⌐ **(no title)**   **5** / 5 pts

✔ **+ 5 pts** Correct: AND R3, R3, 0

+ 2.5 pts Partially correct

+ 0 pts Incorrect

3.2 ⌐ **(no title)**   **6** / 6 pts

✔ **+ 6 pts** Correct:
    ADD R2, R2, 10
    ADD R2, R2, 11
    (can use any Imm5 values < 16 that sum to 21)

+ 3 pts Partially Correct:
    ADD R2, R2, 21 (imm5 add outside of 5 bit range)

+ 0 pts Incorrect

3.3 ⌐ **(no title)**   **6** / 6 pts

✔ **+ 6 pts** Correct:
    ADD R1, R0, R0 ;MULTIPLY BY 2
    ADD R1, R1, R0
    (or any code that works)

+ 3 pts Partially Correct:
    (1) Does not place answer in R1
    (2) Multiplies another register by 3 instead of R0

+ 0 pts Incorrect

− 0.5 pts Syntax Error (e.g. has comma after instruction, no comma between arguments)

Question 4

LC-3 Assembly - Fill in the Blank   **10** / 10 pts

✔ **+ 10 pts** Correct:
    (1) BRn ENDWHILE
    (2) ST R0, ANSWER

+ 5 pts Partially Correct - 1 out of 2 correct

+ 0 pts Incorrect

Question 5

LC-3 Assembly - Tracing   **26** / 26 pts

5.1   (no title)   **6** / 6 pts

    + 0 pts Incorrect

    ✔ **+ 6 pts** Correct (0x300B)

5.2   (no title)   **6** / 6 pts

    + 0 pts Incorrect

    ✔ **+ 6 pts** Correct (12)

5.3   (no title)   **6** / 6 pts

    + 0 pts Incorrect

    ✔ **+ 6 pts** Correct (4)

5.4   (no title)   **4** / 4 pts

    ✔ **+ 4 pts** Correct (BRzp 1)

    + 3.5 pts Partially Correct: Forgets Space (BRzp1)

    + 0 pts Incorrect

5.5   (no title)   **4** / 4 pts

    + 0 pts Incorrect

    ✔ **+ 4 pts** Correct (0x3009)

Question 6

LC-3 Assembly - Long Coding   **23** / 23 pts

6.1 ⎢ **(no title)**   **15** / 15 pts

> ✔ **+ 15 pts** Completely correct

> **+ 3 pts** Correctly calculates effective address
> (should be at x3105 which is ARR + LENGTH - 1)

> **+ 2 pts** Correctly reads the last value of the array into a register (propagate error on calculated effective
> address)

> **+ 2 pts** Correctly sets the value to 0 if the last value is positive or zero (and branches over else block)

> **+ 2 pts** Correctly flips the sign if the last value is negative

> **+ 2 pts** Checks condition codes based on value

> **+ 2 pts** Partial credit - performs opposite based on value (ex. sets to 0 if value is negative)

> **+ 4 pts** Correctly stores updated value back into memory

> **+ 2 pts** Partial Credit- Only stores one value in memory correctly

> **+ 0 pts** Incorrect

6.2 ⎢ **(no title)**   **5** / 5 pts

> ✔ **+ 5 pts** Correct:
> LD RX, ARRAY
> LDR R0, RX, 3

> **+ 2.5 pts** Partially correct - does ONE of the following:
> (1) Used LDI/LEA instead of LD
> (2) Loads index off by one (ex. LDR R0, RX, 4)
> (3) Loads result into wrong register

> **+ 0 pts** Incorrect

> **+ 4 pts** Mostly correct -
> (1) Minor syntax error

6.3 ⎢ **(no title)**   **3** / 3 pts

> ✔ **+ 3 pts** Correct (LD R4, LENGTH) (still give points if commas/spaces are off)

> **+ 0 pts** Incorrect

Question 7

LC-3 Assembly - More Tracing   **15** / 15 pts

7.1 (no title)   **5** / 5 pts
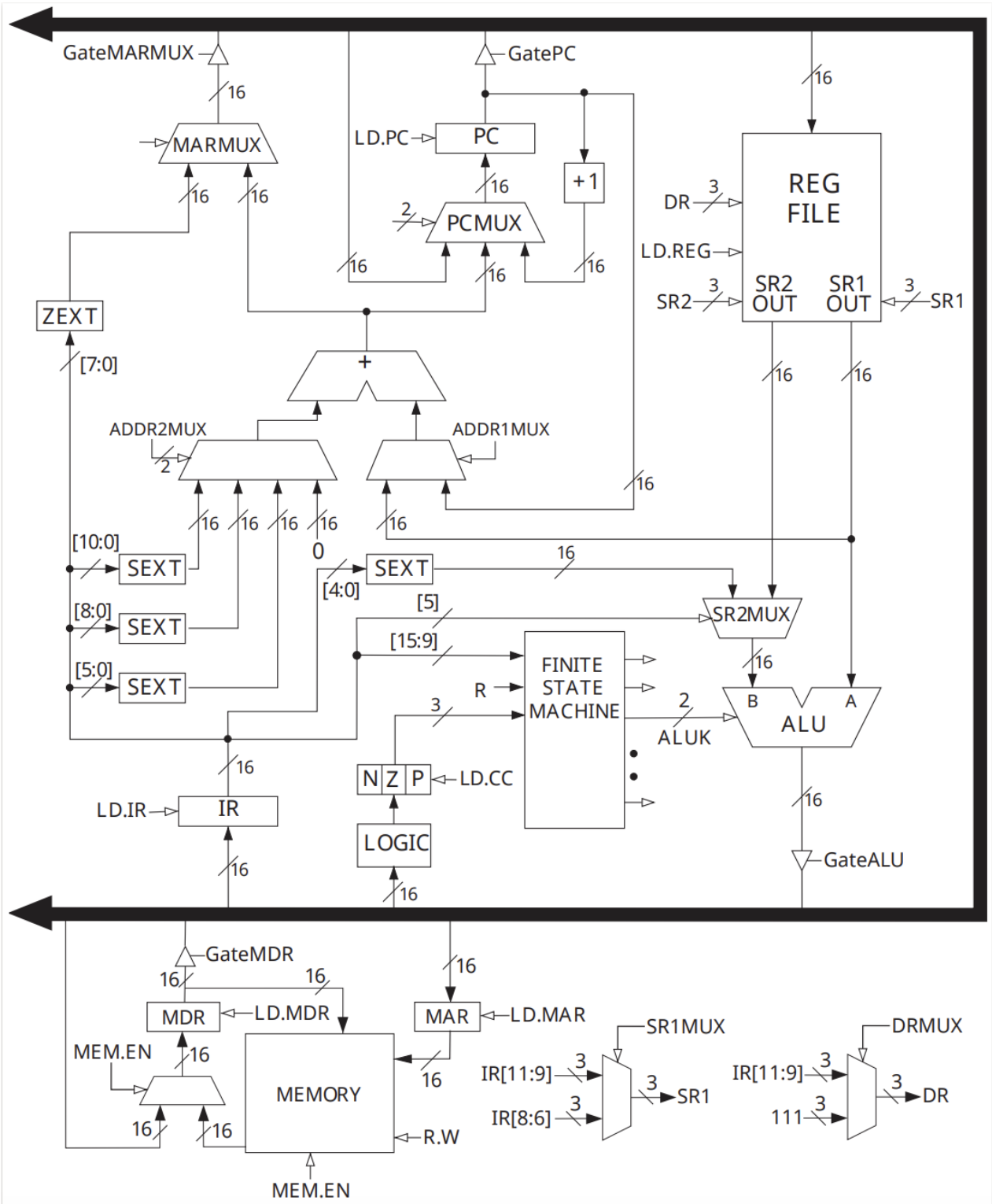
+ **0 pts** Incorrect

✔ + **5 pts** Correct (8)

7.2 (no title)   **5** / 5 pts

✔ + **5 pts** Correct (8)

+ **0 pts** Incorrect

7.3 (no title)   **5** / 5 pts

✔ + **5 pts** Correct (unknown)

+ **0 pts** Incorrect

Q1 Overview
0 Points

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADD		00	01			DR			SR1		0		00		SR2	

ADD

0001	DR	SR1	1	imm5

15 14 13 12	11 10 9	8 7 6	5	4 3	2 1 0

AND

0101	DR	SR1	0	00	SR2

AND

0101	DR	SR1	1	imm5

15 14 13 12	11 10 9	8 7 6	5 4 3 2 1 0

NOT

1001	DR	SR	111111

15 14 13 12	11 10 9	8 7 6 5 4 3 2 1 0

BR

0000	N	Z	P	PCoffset9

15 14 13 12	11	10	9	8 7 6 5 4 3 2 1 0

JMP

1100	000	BaseR	000000

JSR

0100	1	PCoffset11

JSRR

0100	0	00	BaseR	000000

15 14 13 12	11 10 9	8 7 6	5 4 3 2 1 0

LD

0010	DR	PCoffset9

LDI

1010	DR	PCoffset9

LDR

0110	DR	BaseR	offset6

LEA

1110	DR	PCoffset9

15 14 13 12	11 10 9	8 7 6	5 4 3 2 1 0

ST

0011	SR	PCoffset9

STI

1011	SR	PCoffset9

STR

0111	SR	BaseR	offset6

15 14 13 12	11 10 9	8	7 6 5 4 3 2 1 0

TRAP

1111	0000	trapvect8

This homework is worth a total of 100 points.

We have provided LC-3 datapath and instruction set here, but LC-3 reference materials can **also be found in Canvas > Files**.

This question (Q1) cannot be answered. It's used for formatting instructions. Do not worry about Gradescope saying you haven't answered one question. It's this one!

Please complete the following problems. The collaboration policy for the course still applies. Refer to the syllabus for details regarding this policy.

Q2 Basic Assembly & Pseudo-Ops
9 Points

Q2.1 True/False
3 Points

In LC-3 Assembly, HALT and TRAP x25 are the same thing, and can be used interchangeably.

- ◉ TRUE
- ○ FALSE

Q2.2 Trap Vectors
3 Points

Consider the following code:

```
.orig x3000
LD R0, CHAR
    ;;your code here!!
HALT


CHAR .fill 97 ;;stores character at label CHAR
.end
```

Which **TRAP** instruction would be useful to print the character from the label **CHAR** to the console?

- ○ HALT
- ○ GETC
- ○ PUTS
- ◉ OUT

Q2.3 True/False
3 Points

Consider the following LC-3 Code:
.orig x3500

The code above is an LC-3 **instruction**, which tells the processor to place the instructions in the program in memory starting at x3500.

○ TRUE

◉ FALSE

Q3 LC-3 Assembly - Coding
17 Points

For the following problems, write assembly instructions to complete the task described. These should each take 3 instructions or fewer.

Q3.1
5 Points

Set the value of Register 3 (R3) to 0.

```
AND R3, R3, 0
```

Q3.2
6 Points

Add 21 to the value already in Register 2 (R2).

```
LD R0, VAL
ADD R2, R2, R0
VAL .fill 21
```

Q3.3
6 Points

Multiply the value in R0 by 3 and place the answer in R1

```
ADD R1, R0, R0
ADD R1, R1, R0
```

Q4 LC-3 Assembly - Fill in the Blank
10 Points

Consider the LC-3 assembly code below:

```
;; ··Modulus(x):
;; ······int·x·=·17;
;; ······int·mod·=·5;
;; ······while·(x·>=·mod)·{
;; ·········x·-=·mod;
;; ······}
;; ······mem[ANSWER]·=·x;

.orig·x3000
    ····LD·R0,·X·
    ····LD·R1,·MOD·
    ····NOT·R2,·R1
    ····ADD·R2,·R2,·#1

    ····WHILE·ADD·R3,·R0,·R2·
    ·········;;·Blank·1:·Your·code·here!
    ·········ADD·R0,·R0,·R2
    ·········BR·WHILE
    ····ENDWHILE
    ····;;Blank·2:·Your·code·here!
    ····HALT

    ····X········.fill·17
    ····MOD·····.fill·5
    ····ANSWER·.blkw·1
.end
```

Given the pseudocode provided at the top of the image, fill in the indicated blanks to properly execute the modulus program.

Blank 1

```
BRn ENDWHILE
```

Blank 2

```
ST R0, ANSWER
```

Q5 LC-3 Assembly - Tracing
26 Points

Consider the following LC-3 Assembly code

```
                .orig x3000

                AND R0, R0, 0
                AND R1, R1, 0
HERE            ADD R0, R0, -1
                BRzp SKIP
                ADD R1, R1, 4
SKIP            LEA R2, A
                LD R3, B
                ADD R4, R3, R1
                BRnp END
                ST R4, A
END             HALT

A               .fill 12
B               .fill -3
                .end
```

Q5.1
6 Points

Assuming the code finishes running, what is the value in **R2**?

Please answer in hexadecimal and put 0x *in front of your answer.*

0x300B

Q5.2
6 Points

Assuming the code finishes running, what is the value at label **A**?

Please write your answer as a decimal number

> 12

Q5.3
6 Points

Assuming the code finishes running, what is the value in **R1**?

Please write your answer as a decimal number.

> 4

Q5.4
4 Points

Take the instruction `BRzp SKIP`. Rewrite this instruction using a decimal offset instead of a label, while preserving its functionality.

> BRzp 1

Q5.5
4 Points

Where in memory is the instruction `ST R4, A` located?

Please answer in hexadecimal and put `0x` in front of your answer.

> 0x3009

Q6 LC-3 Assembly - Long Coding
23 Points

Given the following LC-3 Assembly code, answer the questions below.

NOTE: each question is independent of the others (i.e. your answer for question 6.1 does not affect other questions)

```
.orig x3000

    ;YOUR CODE HERE
    HALT


LENGTH  .fill 6
ARRAY   .fill x3100
.end


.orig x3100
        .fill 4
        .fill -8
        .fill -3
        .fill 7
        .fill 2
        .fill -6

.end
```

Q6.1
15 Points

Using the assembly code above:

Write a program (replacing `;YOUR CODE HERE` above) that:

- Reads the last value of the array
- If that value is positive or zero, set the value **in memory** to 0
- Otherwise, flip the sign of the value **in memory** (ex. -4 becomes 4)

```
LD R0, ARRAY
ADD R1, R0, 5
LDR R2, R1, 0

BRzp SET
NOT R2, R2
ADD R2, R2, 1
BR FINISH


SET AND R2, R2, 0


FINISH STR R2, R1, 0
```

Q6.2
5 Points

Using the provided assembly code above:

Write **2 LC-3 assembly instructions** (replacing `; YOUR CODE HERE`) that loads the fourth value of the array (7) into **R0**.

```
LD R1, ARRAY
LDR R0, R1, 3
```

Q6.3
3 Points

Using the provided assembly code above:

Write a **single** LC-3 assembly instruction (replacing `; YOUR CODE HERE`) that loads the value **6** into **R4**.

```
LD R4, LENGTH
```

Q7 LC-3 Assembly - More Tracing
15 Points

Given the following LC-3 Assembly code, answer the questions below.

```
.orig x3000
        .fill x5020
        ADD R0, R0, 8
        .fill x923F
        ADD R1, R1, 1
        ADD R2, R1, 8
        ST R0, 3
        .fill xF025
        .fill x3401
        AND R3, R3, 0
.end
```

Q7.1
5 Points

What is the value in **R0** as a decimal number?

Note: if a value is unknown, write your answer as unknown .

8

Q7.2
5 Points

What is the value in **memory** at `0x3009` as a decimal number?

Note: if a value is unknown, write your answer as `unknown` .

> 8

Q7.3
5 Points

What is the value in **R3** after the program finishes running? Write your answer as a decimal number.

Note: if a value is unknown, write your answer as `unknown` .

> unknown