Name: ____________________________

Problem 1 (15 points): _________
Problem 2 (15 points): _________
Problem 3 (15 points): _________
Problem 4 (10 points): _________
Problem 5 (15 points): _________
Problem 6 (10 points): _________
Problem 7 (20 points): _________
Total (100 points): ________________

Note: Please be sure that your answers to all questions (and all supporting work that is required) are contained in the space provided.

Note: Please be sure your name is written legibly on each sheet of the exam.

I will not cheat on this exam.

______________________________
Signature

GOOD LUCK!
Problem 1 (15 points): Yale’s short answer

Part a (3 points): As you know the memory address space of the LC-3 is 16 bits. If the MAR is loaded with the value xFE00, how does the hardware know to access the Keyboard Status Register or memory location xFE00.

(in 15 words or fewer, please)

Answer:

Part b (4 points): The following assembly program is assembled, and run on the LC-3 Simulator.

```
.ORIG x3000
LD R0, ASCII0
TRAP x21
TRAP x21
TRAP x21
TRAP x21
TRAP x21
TRAP x21
ASCII0 .FILL x30
.END
```

Before it is executed, you set a breakpoint at x3003. What happens? (in 15 words or fewer, please.)

Answer:

Part c (4 points): At the end of a LD instruction midway through the execution of a LC-3 program, the contents of the condition codes are set as follows N=1, Z=1, P=0. What can you infer? (in 15 words or fewer, please)

Answer:

Part d (4 points): There are three addressing modes available to the assembly language programmer who wishes to load a value from memory into R5. If the load instruction is in a loop and each time through the loop, the next consecutive memory location is loaded into R5, which addressing mode is most appropriate to use. Explain why. (in 15 words or fewer, please).

Answer:
Problem 2 (15 points):

What is the output of the following program?

```
.ORIG x3000
LD R1, LETA
LEA R0, BUFFER
STR R1, R0, #0
ADD R1, R1, #1
STR R1, R0, #1
ADD R1, R1, #1
STR R1, R0, #2
ADD R1, R1, #1
STR R1, R0, #3
TRAP x22
LD R0, LF
TRAP x21
LEA R0, STRING
TRAP x22
TRAP x25
LF .FILL x000A
LETA .FILL x0041
BUFFER .BLKW #4
STRING .STRINGZ "EFGH"
.END
```

Please write your answer in the box below in 15 words or fewer:

**Answer:** 
3
Problem 3 (15 points):

**Part a** (10 points): Reverse-assemble the binary program (convert the binary program into an assembly language program). Most of the instructions have already been reverse-assembled for you, so your job is to complete the task.

<table>
<thead>
<tr>
<th>Label</th>
<th>Assembly Language</th>
<th>Machine Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>.ORIG x3000</td>
<td>0011000000000000</td>
<td>0011000000000000</td>
</tr>
<tr>
<td>AND R0, R0, x0</td>
<td>0101000000100000</td>
<td>0101000000100000</td>
</tr>
<tr>
<td>AND R1, R1, x0</td>
<td>0101001001100000</td>
<td>0101001001100000</td>
</tr>
<tr>
<td>ADD R1, R1, x9</td>
<td>0001001001101001</td>
<td>0001001001101001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0001000000000100</td>
</tr>
<tr>
<td>LD R2, FF</td>
<td>0010010000001000</td>
<td>0010010000001000</td>
</tr>
<tr>
<td>LEA R3, FF</td>
<td>1110011000000111</td>
<td>1110011000000111</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0111001011000010</td>
</tr>
<tr>
<td>LEA R7, DD</td>
<td>1110111000000011</td>
<td>1110111000000011</td>
</tr>
<tr>
<td>EE</td>
<td>NOT R5, R5</td>
<td>1001101101111111</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1001101101111111</td>
</tr>
<tr>
<td>BRnz DD</td>
<td>0000110000000001</td>
<td>0000110000000001</td>
</tr>
<tr>
<td>NOT R4, R3</td>
<td>1001100011111111</td>
<td>1001100011111111</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0110110010000001</td>
</tr>
<tr>
<td>TRAP x25</td>
<td>1111000000100101</td>
<td>1111000000100101</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1101000000000000</td>
</tr>
<tr>
<td>.FILL xFF00</td>
<td>1111111100000000</td>
<td>1111111100000000</td>
</tr>
<tr>
<td>.FILL xFAFA</td>
<td>1111110111111010</td>
<td>1111110111111010</td>
</tr>
</tbody>
</table>

**Part b** (5 points): Generate the symbol table that an LC-3 assembler would create while assembling this program. You may not need all of the spaces provided.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Problem 4 (10 points):

The input to the following logic circuit is the 16 bits of the MAR. What information does the output provide?

Answer: 5
Problem 5 (10 points):

The following program has been assembled and loaded into the LC-3 simulator. A breakpoint has been set on the TRAP x25 instruction. Your job is to trace the execution of the program until the breakpoint is reached. By trace we mean record the value of the PC at the beginning of each instruction in the order that they are executed. The first few have been done for you. You may not need all of the spaces provided.

Note: The instruction labeled A is at location x0100 in memory, and the instruction labeled START is at location x3000 in memory.

```
.ORIG x0000
.FILL x0100
.FILL x0101
.FILL x0102
.FILL x0103
.FILL x0104
.FILL x0105
.FILL x0106
.BLKW x00F9

A RET ; LOCATION x0100
RET
RET
RET
RET
RET

.BLKW x2EF9

START AND R0, R0, #0 ; LOCATION x3000
BRz L1
LD R0, DATA
L1 NOT R0, R0
BRn L2
TRAP x05
L2 TRAP x06
AND R0, R0, #0
TRAP x25
DATA .FILL x4040
.END
```

<table>
<thead>
<tr>
<th>PC Trace</th>
</tr>
</thead>
<tbody>
<tr>
<td>x3000</td>
</tr>
<tr>
<td>x3001</td>
</tr>
<tr>
<td>x3003</td>
</tr>
</tbody>
</table>

**Name:**

---
Problem 6 (10 points):

What does the following program do?

```
.ORIG x3000
AND R5, R5, #0
AND R3, R3, #0
ADD R3, R3, #8
LEA R0, BB
LDR R1, R0, #1
LDR R1, R1, #0
ADD R2, R1, #0
AGAIN ADD R2, R2, R2
ADD R3, R3, #-1
BRp AGAIN
LDR R4, R0, #0
AND R1, R1, R4
NOT R1, R1
ADD R1, R1, #1
ADD R2, R2, R1
BRnp NO
ADD R5, R5, #1
NO TRAP x25
BB .FILL xFF00
.FILL x4000
.END
```

Please write your answer in the box below in 25 words or fewer:

**Answer:** 7
Problem 7 (20 points):

The table below shows a snapshot of the Program Counter, the 8 registers, and the condition code (CC) of the LC-3 at six different times during the execution of a program: before the program executes, after execution of instruction 1, after execution of instruction 2, after execution of instruction 3, after execution of instruction 4, and after execution of instruction 5. Fill in the missing values in the table as well as the missing parts of instructions 1, 4 and 5.

<table>
<thead>
<tr>
<th></th>
<th>Initial Value</th>
<th>After 1st Instruction</th>
<th>After 2nd Instruction</th>
<th>After 3rd Instruction</th>
<th>After 4th Instruction</th>
<th>After 5th Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>CC</td>
<td>Z</td>
<td>N</td>
<td>P</td>
<td>N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R0</td>
<td>x0000</td>
<td></td>
<td>x0000</td>
<td>x0000</td>
<td>x0000</td>
<td></td>
</tr>
<tr>
<td>R1</td>
<td>x1111</td>
<td>x1111</td>
<td>x1111</td>
<td>x1111</td>
<td>x1111</td>
<td>x1111</td>
</tr>
<tr>
<td>R2</td>
<td>x2222</td>
<td>x2222</td>
<td>x2222</td>
<td>x2222</td>
<td>x2222</td>
<td>x2222</td>
</tr>
<tr>
<td>R3</td>
<td>x3333</td>
<td>x3333</td>
<td>x3308</td>
<td>x3308</td>
<td>x3308</td>
<td>x3308</td>
</tr>
<tr>
<td>R4</td>
<td>x4444</td>
<td>x4444</td>
<td>x4444</td>
<td>x9FFF</td>
<td>x9FFF</td>
<td></td>
</tr>
<tr>
<td>R5</td>
<td>x5555</td>
<td>x5555</td>
<td>x5555</td>
<td>x5555</td>
<td>x5555</td>
<td>x5555</td>
</tr>
<tr>
<td>R6</td>
<td>x6666</td>
<td>x6666</td>
<td>x6666</td>
<td>x6666</td>
<td>x6666</td>
<td>x6666</td>
</tr>
<tr>
<td>R7</td>
<td>xFEFE</td>
<td>xFEF0</td>
<td>xFEF0</td>
<td>xFEF0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Instruction 1: AND

Instruction 2: LEA R3 , #6

Instruction 3: LEA R4 , #0

Instruction 4: LD __________ , #0

Instruction 5: