Department of Electrical and Computer Engineering
The University of Texas at Austin

ECE 306 Fall 2025
Instructor: Yale N. Patt
TAs: Luke Mason, Evan Lai, Madeleine Dreher
Exam 2
November 12, 2025

Name: Solution

Problem 1 (20 points): 20

Problem 2 (20 points): 20

Problem 3 (30 points): 30

Problem 4 (30 points): 30

Total (100 points): 100!

Note: Please be sure that your answers to all questions (and all supporting work that is required) are contained in the space provided. **Any solution we (Dr. Patt or the TAs) cannot read will receive 0 points.**

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

Please read the following sentence, and if you agree, sign where requested:
I have not given nor received any unauthorized help on this exam.

Signature:_____

GOOD LUCK!

Name: _____

Problem 1 (20 pts): Answer the following questions. An unanswered question will receive 1 point out of the 5.

Part A (8 pts): Every state in the state machine is defined by the values of the control signals that are specified for that state. The control signals specify both the processing done during its clock cycle, and the address (i.e., state number) of the state to be executed in the next clock cycle.

Your job: List the values of each control signal needed to specify the state whose address is #18. For example, if the state address is #30, the control signals are: LD_IR/Load, Gate_MDR/YES, For all other Gate_x control signals, Gate_x/NO, J/100000, COND/000, IRD/0.

LD.PC/Load J/011100 IRD/0
LD.MAR/Load For all other Gate.x signals, Gate.x/No
Gate.PC/Yes For all other Load.x signals, Load.x/No
PCmux/PC+1
COND/000

Part B (4 pts): The label KBDR corresponds to location xFE02 in LC-3 memory. An assembly language programmer includes in his assembly language program at x3000 the instruction LD R2, KBDR. What is the result of processing this instruction?

Assembler error (out of PC+offset9 range)

Part C (4 pts): What (everything) must be true about an assembly language program if state #20 is carried out during the execution of the program?

It must contain at least one JSRR instruction

Part D (4 pts): How many clock cycles does it take for the LC-3 to execute the following code segment if memory accesses take 4 clock cycles? Assume X is x3320 and Y is x3340.

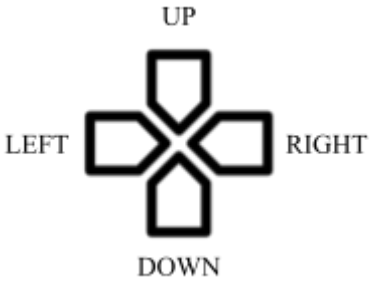
```
X   ADD R3, R3, #0 ; USE R3   8
    LEA  R3, Y      8
```

16 cycles

Name: _____

Problem 2 (20 pts):

You are writing a TRAP service routine to read input from a **directional pad (D-pad)** on a game controller, outputting which button was pressed. This is what a d-pad looks like:



The TRAP service routine continues to check if the d-pad has input ready, and then returns the ASCII code for the direction in R0 ('U', 'D', 'L', or 'R'). **All registers except R0 must be saved and restored.** Part of the service routine has been given to you on the next page.

We use memory-mapped I/O to access the two game controller registers as follows:

Game Controller Status Register (GC_SR) mapped to address xFEA0:

- Bit [15]: D-pad ready (1 = input available)
- All other bits: Other inputs from the game controller (buttons, joystick, etc…) are not used in this problem.

Game Controller Data Register (GC_DR) mapped to address xFEA2:

- Bits [3:0]: D-pad direction:

	3	2	1	0
Up	1	0	0	0
Down	0	1	0	0
Left	0	0	1	0
Right	0	0	0	1

- All other bits: Other inputs from the game controller not used in this problem

Part A (4 pts): You notice the service routine, in the worst case, takes around 200 cycles to finish after you make an input. If my LC-3 clock is running at 50 Hz (50 cycles/second), around how many seconds is that?

Seconds | 4

THE PROBLEM CONTINUES ON THE NEXT PAGE

Name: _____

Part B (16 points): Fill in the missing blanks in the code.

```
            .ORIG x2000
GET_DPAD    ST  R1 SAVE_R1
POLL        LDI R1 GCSR____
            BRzp POLL
            LDI R1 GCDR____
            LD  R0 UP_MASK
            AND R0 R0 R1
            BRnp IS_UP
            LD R1 DOWN-MASK
            AND R0 R0 R1
            BRnp IS_DOWN
            LD R1 LEFT_MASK
            AND R0 R0 R1
            BRnp IS_LEFT
            LD  R0 ASCII_R
            BRnzp DONE
IS_UP       LD  R0 ASCII_U
            BRnzp DONE
IS_DOWN     LD  R0 ASCII_D
            BRnzp DONE
IS_LEFT     LD  R0 ASCII_L
DONE        LD R1 SAVE_R1____
            RTI
SAVE_R1     .BLKW #1
GCSR        .FILL xFEA0
GCDR        .FILL xFEA2
UP_MASK     .FILL x0008
DOWN_MASK   .FILL x0004
LEFT_MASK   .FILL x0002
ASCII_U     .FILL x0055
ASCII_D     .FILL x0044
ASCII_L     .FILL x004C
ASCII_R     .FILL x005
            .END
```

Name: _____

Problem 3 (30 points):

Part A (3 pts): It takes N amount of time, on average, to search for a node in a linked list. If the size of the linked list is doubled, what happens to N?

It doubles

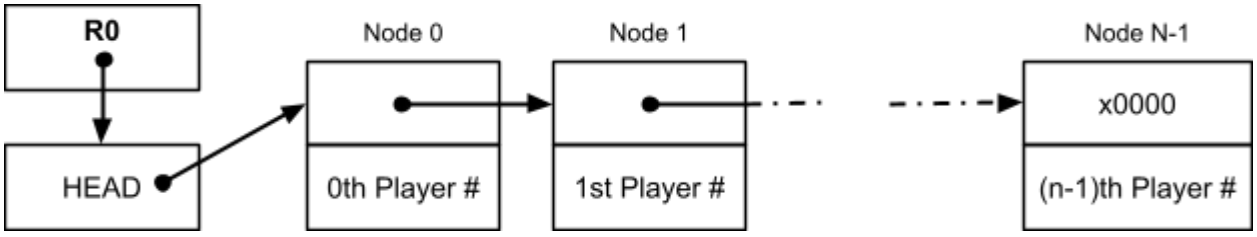
Part B (3 pts): Suppose R4 contains the location of node A in the linked list. It takes M amount of time, on average, to insert a node into the linked list after node A. If the size of the linked list is doubled, what happens to M?

It stays the same

Use the following code and explanation to answer part C, D, and E.

The head referee of the recent UT vs Vanderbilt football game has been forced to change careers and joined your ECE 306 class. Unfortunately, he has no idea how to complete his linked-list programming assignment. His goal is a subroutine to add football players (designated by their jersey number) into a linked list consisting of all players on the team. Each number is an unsigned 16 bit integer, and the players are sorted numerically from smallest number to largest number.

- Each player will be represented by a node consisting of two consecutive words.
    - The first word of each linked list node is a pointer to the next node.
    - The second word of a linked list node is the 16-bit player number.
- R0 initially contains the address of the list head which is itself a pointer to the first node.
- R1 initially contains the 16-bit unsigned integer value player number to insert.
- R6 initially contains the starting address of two available memory locations where a new node could be created.
- The final node's next pointer is x0000, the null terminator.



THE PROBLEM CONTINUES ON THE NEXT PAGE

Name: _____

The referee provided the SEARCHANDINSERT subroutine shown below. Assume registers do not need to be saved. Initially: R0 = address of list head, R1 = new player's number, R6 = address of available memory.

```
                .ORIG x3050
SEARCHANDINSERT NOT R5, R1
                ADD R5, R5, #1

LOOP1           ADD R4, R0, #0
                LDR R0, R0, #0
                BRz NEWNODE
                LDR R2, R0, #1
                ADD R3, R2, R5
                BRnz LOOP1

INSERT          STR R1, R0, #1
                ADD R1, R2, #0

LOOP2           ADD R4, R0, #0
                LDR R0, R0, #0
                BRz NEWNODE
                LDR R3, R0, #1
                STR R1, R0, #1
                ADD R1, R3, #0
                BR LOOP2

NEWNODE         STR R6, R4, #0
                AND R4, R4, #0
                STR R4, R6, #0
                STR R1, R6, #1
DONE                RET

                .END
```

Determines where the new player must be inserted

Insert the new player's number and shift other jersey numbers down the list accordingly.

Add new node at the end of the list

Name: _____

Part C (9 pts): The code successfully inserts the new node into the linked list in the correct location. However, there is an important performance aspect of linked lists that was completely ignored. Explain. *hint: performance relates to how long an operation takes to run.*

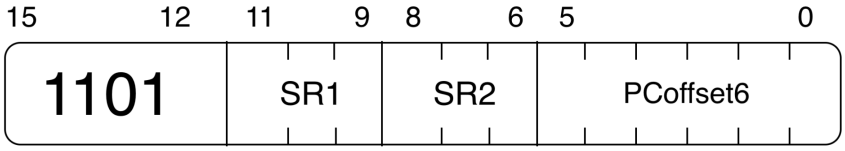
We traverse the list twice! Once to determine where to insert the new player, and once to adjust the jersey numbers of other players after we insert the new player. This second traversal can be avoided as we are using a linked list: we can insert the new player through manipulating "next node" pointers.

Part D (15 pts): In the code on the previous page, **cross out the sections** with code that you would need to replace to fix the performance problem. Then, write the correct code in the box below to replace the sections you crossed out (don't rewrite code that you didn't cross out).

```
STR R6, R4, #0
STR R1, R6, #1
STR R0, R6, #0
BR_nzp DONE
```

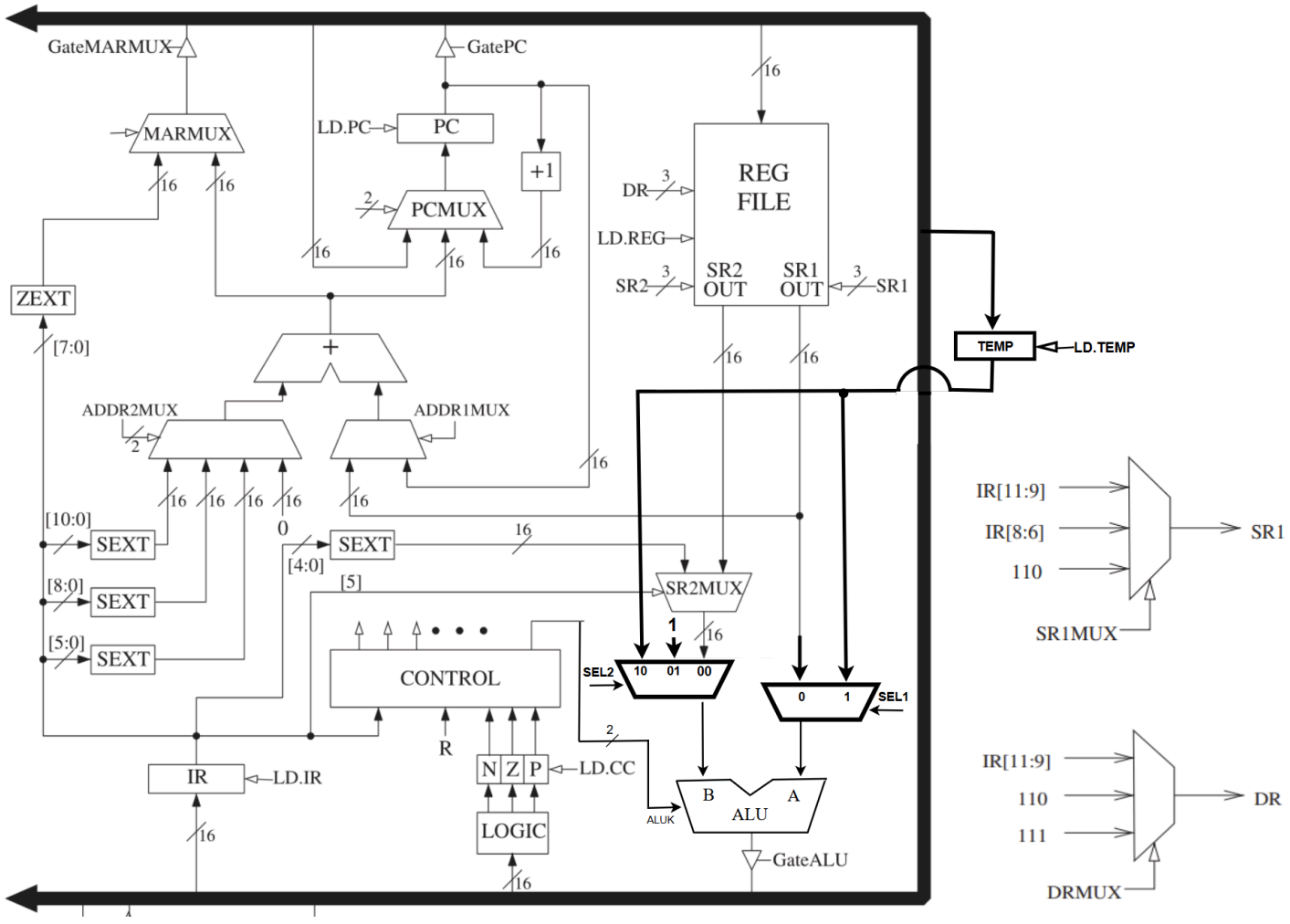
Name: _____

Problem 4 (30 points): We use the unused opcode (1101) to create a new LC3 instruction. The format of the instruction is shown below.



In order to support this instruction, we modified the datapath, the microsequencer, and the state machine.

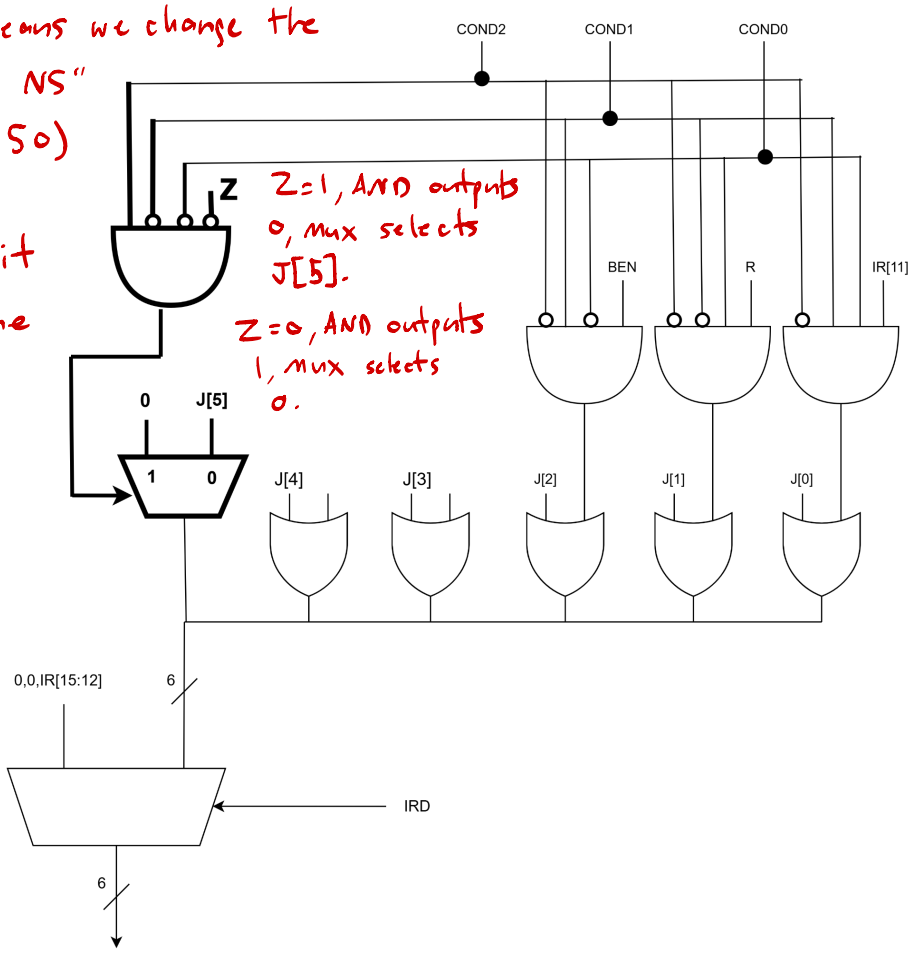
The changes to the datapath include a new register called **TEMP**, a mux between SR2MUX and ALU, and a mux between SR1OUT and ALU, all shown below in bold.



Name: _____

We also replaced the OR gate that had input J[5] in the microsequencer with an AND gate and a mux, also shown below in bold.

This means we change the "default NS" (which is 50) to 18 by clearing bit 5 of the NS.

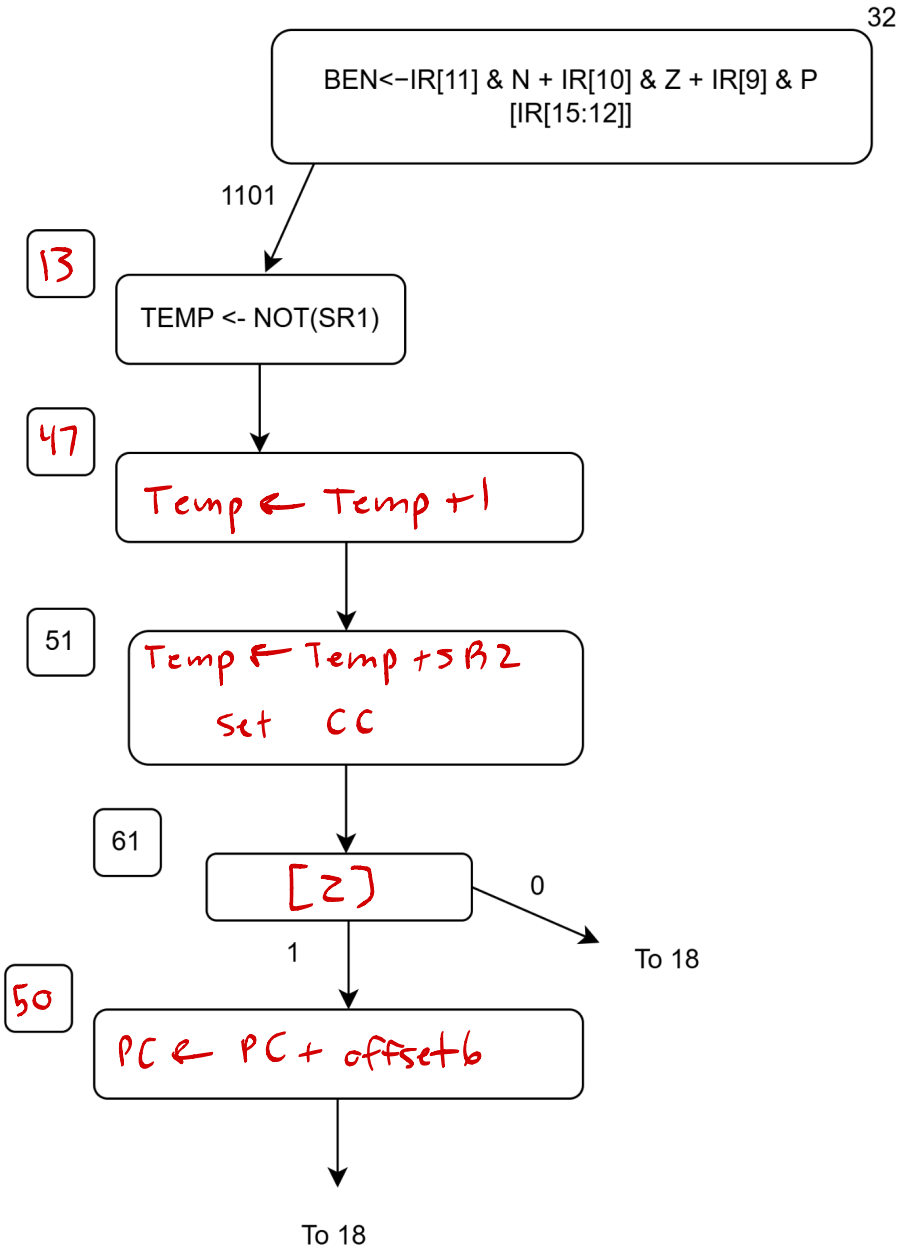


The state machine requires five new states from state #32 to the end of the instruction. We added five rows of control store below to implement those five states. Signals that are irrelevant in a given state are marked with an "X."

STATE NUMBER	COND	J	LD.PC	LD.CC	LD.TEMP	GATEALU	PCMUX	SEL2	SEL1	ADDR1MUX	ADDR2MUX	SR1MUX	ALUK
51	000	111101	0	1	1	1	X	TEMP	SR1OUT	X	X	IR[8:6]	ADD
13	000	101111	0	0	1	1	X	X	SR1OUT	X	X	IR[11:9]	NOT
50	000	010010	1	0	0	0	ADDER	X	X	PC	offset6	X	X
61	100	110010	0	0	0	0	X	X	X	X	X	X	X
47	000	110011	0	0	1	1	X	ONE	TEMP	X	X	X	ADD

Name: _____

Part A (20 points): Your job: use the above information to fill out **both** the state machine for the new instruction **and** the two missing state numbers in the control store (the bolded boxes).



Part B (10 points): In 20 words or less, describe what this instruction does.

Branch if SR1 = SR2

Name: _____

This page is left blank intentionally. Feel free to use it for scratch work.
You may tear the page off if you wish.
Nothing on this page will be considered for grading.