Name (First Last): | **Username (UCR netID):**

CS 061-001 – Computer Org. & Assembly Language — Winter - 2020

Final exam Monday 3/16 — total 10 points

Time: 3 hours

- **You may use your text book and your lecture notes during the exam - *but no other resources***

- **Write your name in the space provided above** – *this is the only place it's needed.*

- **You MUST write your answers in the boxes** – *click inside the box to get focus.*
  Do NOT change the size of the boxes – *they are mapped to the grading template.*
  If you need more space, you're doing something wrong 😊

- **Some questions may require the use of logisim**:
  If you are working on a Windows machine, and have logisim installed locally, just open logisim, build the circuit, export the image to png *(uncheck "Printer view")*, then embed that png in your Word doc, inside the question box.

  If instead you are working on a Mac or Linux machine *(and thus do not have logisim installed locally)*, I suggest that you remote into sledge.cs.ucr.edu via X2Go, open Piazza in a Firefox or Chrome browser on sledge, and use the Google doc link for the exam. Then follow the same instructions for drawing & embedding the circuit.

  Alternatively, you may use any other electronic drawing canvas (MS Canvas, Google Draw, etc.) so long as you save the image as png/jpeg for embedding.

  As a last resort, you may even draw the circuit freehand and scan it in – again, so long as you save it as png/jpg. It must be drawn clearly and meticulously, or it will not be graded.

- **When you are ready, print your exam to pdf, and submit to Gradescope.**
  (you do not have to map your answers as with the homework – the answer boxes are pre-mapped)

- **The Gradescope folder closes 5 minutes after the end of the exam.**
  At, say, 15 minutes before that time, no matter how much you may have done, you should stop, pdf the whole exam, and submit to Gradescope - then go back & finish if necessary.
  You can do this as often as you like – we will only grade the final submission.

- **You may ask questions on Piazza during the exam *(Private to Instructors ONLY!)***
  - but we will not answer questions that look like they are fishing for hints.

Section I: Short Answer (2 lines max) – 2 points each (40 points total)

1. What is the main purpose of the first pass of a two-pass assembler?

1. What is the decimal equivalent of the IEEE-754 floating point number: xC1180000

2. Convert the decimal number **-1,789** to a 16-bit two's complement binary number, and represent the result as hexadecimal

3. Simplify the Boolean expression:
   a.b'.c.d' + a.b.c'.d + a.b'.c'.d' + a'.b.c'.d

4. Given that the ASCII codes for 'A' through 'Z' are x41 through x5A; and the codes for 'a' through 'z' are x61 through x7A:
   What are the ***bit-mask*** and the ***logic operation*** (NOT the LC3 instruction) that would force a character stored in R0 to lower case - i.e. convert an upper-case character stored in R0 into lower case, and leave a lower case unchanged (so 'A' would become 'a', and 'a' would remain 'a').
   Note: the LC-3 stores ASCII characters in the lower ("right-hand") byte of a 16-bit word, with the upper byte set to zero

The next four questions refer to an ISA with a byte addressable memory; addressing is 32 bits; word size is 64 bits. An instruction is a single word.

5. What is the total capacity of memory, in bytes?

6. How many bits are stored in each location in memory?

7. What are the sizes of the PC and IR registers?

8. What are the sizes of the MAR and MDR registers?

2. What is the advantage of the ripple carry adder, and what is the advantage of the carry lookahead?

3. Using logisim *(or equivalent app as per instructions on front page)*, construct the circuit for an **RS Latch**, using only AND, OR, NOT gates *(the NOT gates may be depicted as "open circle" inputs*

The next four questions refer to the following (simplified!) scenario for a Finite State Machine controlling a garage door opener:

- The user controls the door via a one-button clicker; the door's downward motion can be halted when a sensor is tripped; and sensors report when the door is fully up, or fully down.
- The FSM controls the door motor by turning it off or on, in up or down direction.
- The door can be: Moving up, Moving down, Stopped up, Stopped down, Stopped "in between".

9. How many bits need to be stored in the storage component of the FSM?

10. How many bits of control signal need to be output by the combinational logic component of the FSM? (i.e. signals controlling functions of the gate).

11. How many "external inputs" are there to the combinational logic component of the FSM? (i.e. signals from sensors on the gate?)

12. What transitions are possible when the door is in the "Moving down" state?

The next two questions relate to the following LC-3 code fragment *(the hex values in the first column give the address at which the corresponding instruction is stored)***:**

```
        xA109       start       ADD     R1, R1, R2
                                        ……
        xA1C8                   BRzp    start
```

Given: the BR opcode is 0000

13. Assemble the BRzp instruction at xA1CD (i.e. translate the instruction into Machine Language)

14. Which of the condition code registers will be <u>set</u> by the BRzp instruction in the previous question?

15. What specific detail of an ISA determines the word size of a processor? *(e.g. the LC3 is a 16-bit processor)*

16. Given the structure of the LC3 TRAP instruction, how many Trap Service Routines could the LC3 possibly manage?

17. What does the technique of "Memory Mapping" actually map? *(i.e. from what, to what? 2 lines max)*

Section II: Free response - 15 points each

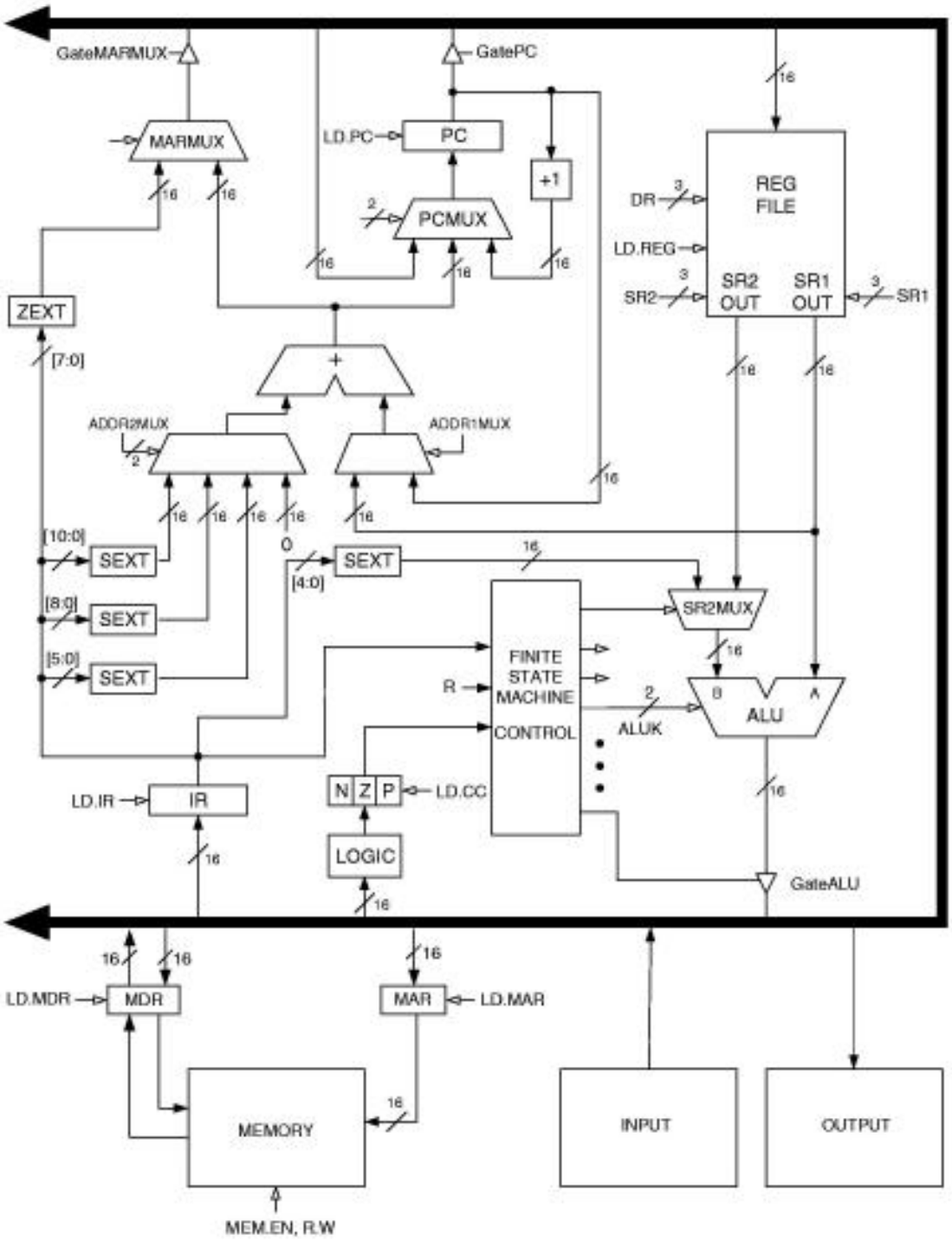
4. Given the data path of the LC-3, give a complete description of the Store Indirect instruction

```
STI R5, result      ; where "result" is a label whose PCoffset9 is x1A0
```

a) (1 point) Using the op-code table provided, assemble the instruction to ML

b) (2 points) Give the RT (Register Transfer) specification of the instruction.

c) (12 points) List, in the correct sequence, every control signal set by the FSM to implement this instruction.
Use the attached schematic as a guide - but remember that even though some tristate devices or MUXes may not be depicted on it, you must still specify all relevant control signals *(compose your own names for any control signals that may not be shown on the schematic)*.
For control signals that act as selectors, indicate the data or process selected – e.g. if the control selects the RegisterX as the input to MUX123, you would write: **MUX123 selects RegisterX**
Points will be awarded only for control signals listed in the correct order.


GateMARMUX
GatePC
MARMUX
LD.PC
PC
+1
PCMUX
REG FILE
DR
LD.REG
SR2
SR2 OUT
SR1 OUT
SR1
ZEXT
[7:0]
ADDR2MUX
ADDR1MUX
[10:0]
[8:0]
[5:0]
SEXT
SEXT
SEXT
0
[4:0]
SR2MUX
FINITE STATE MACHINE
CONTROL
R
ALUK
ALU
LD.IR
IR
N Z P
LD.CC
LOGIC
GateALU
LD.MDR
MDR
MAR
LD.MAR
MEMORY
INPUT
OUTPUT
MEM.EN, R.W

5.

(a) 10 points Write a Trap Service Routine (TSR) that will read an ascii character from the keyboard and store and return it in **R0**; but echo to the display the **'*'** character (ascii x2A) *(this is to hide a password as it is being typed)*.
Your TSR must handle:
- polling the keyboard status register (memory mapped to xFE00), and reading from the keyboard data register (xFE02)
- polling the display status register (xFE04), and writing to the display data register (xFE06).

*This TSR may **not** itself invoke any TRAPs - if it does, you will lose all 10 points.*
You may use only **R1** for temporary storage *(i.e. save & restore only R1)*

```
;    GET_PW_CHAR
; A TSR for reading a password character from the keyboard:
; The character is stored in R0 for return to the calling routine.
; A '*' character (ascii x2A) is echoed to the display.

              .ORIG x0650
              ST R1, SAVE_R1 ; This is the only Register save needed
```

(b) 5 points In 1 or 2 lines: how would this BIOS routine be associated with trap x30?
i.e. what would have to be set up so that this TSR would be invoked by the instruction TRAP x30

6.

A certain ISA provides for 60 instructions and 32 general purpose registers; It uses 36 bits for addressing a byte addressable memory; its word size is 32 bits; and instructions are single words. Several of its instructions take the following format:

OPCODE	Destination Register	Operand specifier A	Operand specifier B

The two operand specifier fields can be of variable size depending on how they are used.

Express all results in terms of 'k' = 1024, 'M' = $(1024)^2$, 'G' = $(1024)^3$ as appropriate;
two's complement ranges may be expressed to the nearest power of 2 - e. g. a 4-bit two's complement number corresponds to the range approx +/- 8

a. (4 points) For some opcodes, Operand specifier A is a register, and Operand specifier B can be either a register or an immediate value (imm) (like the ADD instruction in the LC3): assuming 1 bit must be used as an addressing mode flag, and assuming 2's complement imm values, **what range of values can be represented by imm?**

b. (4 points) For other opcodes, Operand specifier fields A and B are merged to form a single field for use as a signed PC offset to a target address, similar to the LD and LDI instructions (and others) in the LC-3. What is the range of addresses that an instruction may access? *(assume 2's complement values)*

An instruction can access an address within memory locations

c. (4 points) For yet other opcodes, such as jump to subroutine (JSR), there is no need for a destination register, so all three fields indicated above can be combined to form a single address generation field. Assuming the same PC-offset mechanism as above, and assuming 1 bit must be used as an addressing mode flag, **what is the furthest a subroutine may be located from the JSR?** *(assume 2's complement values)*

memory locations

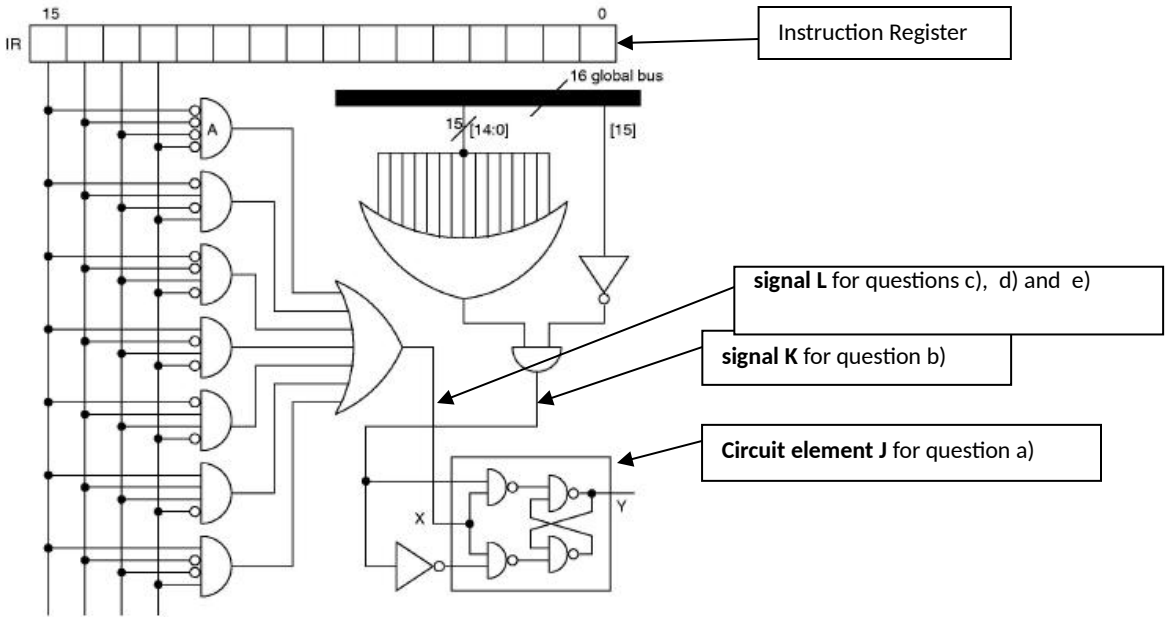
d. (1 point) What is the address space of this ISA –

e. (1 point) what is the maximum capacity of the memory (in bytes) –

f. (1 point) How would the maximum capacity of the memory (as in part e) be affected if the word size were instead 64 bits?

7.

The circuit below shows a part of the implementation of the LC-3 architecture:



a) (2 points) What is circuit element **J**? *(as in multiplexer? adder? …)*

b) (3 points) Signal **K** is one of the inputs to circuit element **J.**
Describe how it is generated *(i.e. the digital logic)*, and what information it provides about the current value on the bus *(i.e. what does it "mean"?)*

c) (2 points) Signal **L** is the other input to circuit element J.
Study the LC-3 overall schematic on page 6 of this exam: Signal L is shown in the schematic; what is it called there?

d) (3 points) Give a ***brief high level*** description of the "meaning" of signal L *(i.e. WITHOUT referring to gates!)*
L is 1 if …

e) (5 points) From your understanding of this whole component of the LC-3 architecture, it should be clear that there is an error in the depicted generation of signal L.
Identify the errorand explain why it is wrong and how to fix it.

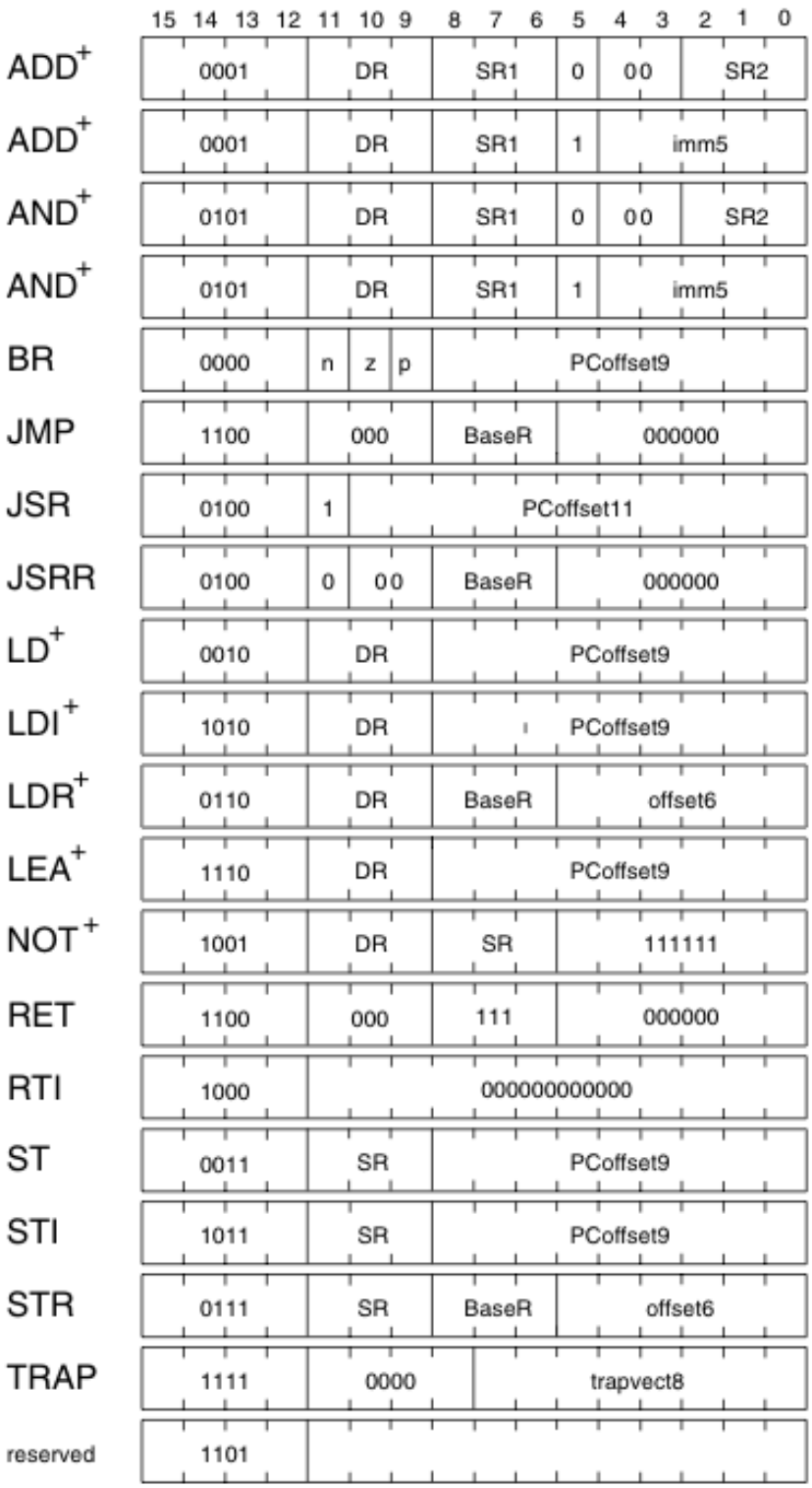


Figure A.2 Format of the entire LC-3 instruction set. Note: + indicates instructions that modify condition codes