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

EE 460N Fall 2022
Instructor: Yale N. Patt
TAs: Kayvan Mansoorshahi, Michael Chen
Exam 2
November 16, 2022

Name: Solutions

Problem 1 (20 points):_______

Problem 2 (10 points):_______

Problem 3 (15 points):_______

Problem 4 (25 points):_______

Problem 5 (30 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 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 points): Answer the following questions.

Note: For each of the four answers below, if you leave the box empty, you will receive one point of the five.

Part a (5 points): Various storage structures are accessed in various ways. One way is the content addressable memory (CAM) access, in which part of the address is contained within the element being accessed. We have studied two storage structures that are accessed in this way. What are they?

TAG STORE OF CACHE	TLB

Part b (5 points): Consider a cache that is physically indexed, physically tagged. What must be true if the TLB can be accessed at the same time as the tag store? Use 15 words or fewer.

THE INDEX BITS MUST BE FROM THE UNMAPPED PAGE OFFSET BITS

Part c (5 points): A page fault often requires a page of virtual memory to be evicted from the frame of physical memory it is occupying in order to provide space for the page that needs to be brought into physical memory. What should be done with the evicted page. Please be complete but concise. No more than 20 words total.

IF PTE[M] = 1, WRITE PAGE BACK TO DISC.
SET PTE[V] = ∅

Part d (5 points): Page mode allows a DRAM access to take less time than not-page-mode.
What must be true for the access to be in page mode?

THE ROW BITS OF THE ADDRESS OF THIS ACCESS MUST BE IDENTICAL TO THE ROW BITS OF THE LAST ACCESS.

How does page mode save time?

WE DON'T HAVE TO LOAD THE ROW BUFFER

Name:____________________________

Problem 2 (10 points): An 8-bit floating point number, with its BIAS set in the spirit of the IEEE standard has a sign bit, 3 bits of fraction, and 4 bits of exponent.

Sign	Exponent	Fraction

0111 → BIAS
7

Answer the following questions for this floating point format.

Part a. (2 points): What's the smallest positive normalized number that can be represented exactly? Answer with a fraction.

0 0001 000 $= 1.000 \times 2^{1-BIAS}$
$= 1 \times 2^{1-7} = 2^{-6}$

1/64

Part b. (3 points): What is the smallest positive number that can be represented exactly? Answer with a fraction.

0 0000 001 $= 0.001 \times 2^{-6}$
$= 2^{-3} \times 2^{-6} = 2^{-9}$

1/512

There exists a number N such that every value greater than N will be represented by N or positive infinity.

Part c. (2 points): What is the value N (Express as a number).

0 1110 111 $= 1.111 \times 2^{14-BIAS}$
$= \frac{15}{8} \times 2^{14-7} = \frac{15}{8} \times 128$

240

Part d. (3 points): What determines whether values greater than N are represented by N or positive infinity. Explain in fewer than ten words.

THE ROUNDING MODE

Name:________________________________

Problem 3 (15 points):
Consider a 64B, physically addressed, write-back cache for a CPU with byte-addressable memory. Assume LRU replacement. Initially all cache lines are invalid.

Part a (5 points):
The processor makes the following consecutive memory accesses. Note the table shows physical addresses (i.e., after the translation has been made.)

	Physical Address in binary
1	1001 0000
2	0011 0000
3	1111 0000
4	1111 1000
5	1001 0010
6	0011 0100
7	0100 0000

Assume the cache is fully-associative. **Five** of the accesses require going to main memory. What is line size of the cache?

8B

Part b (5 points):
If we change the associativity from fully associative to direct-mapped, but leave the line size unchanged, what is the cache hit ratio for processing the seven accesses?

8-Sets. 3-bit index, 3-bit OFFSET

1/7

Part c (5 points):
Increasing associativity will usually reduce the number of conflict misses. If we now change the associativity from direct-mapped to set associative, but leave the line size unchanged, what is the minimum associativity that will minimize the number of conflict misses? Which accesses hit with this associativity?

Minimum associativity: 4-way

Accesses that hit: 5, 6

Name:________________________________

Problem 4 (25 points):
A memory system has the following specification:

- Byte addressable memory.
- 1 channel and 1 rank.
- 4 banks per chip.
- 64 rows per bank.
- 4 columns per row.
- 64-bit (8-byte) memory data bus.
- A page mode access takes 1 cycle. A non-page mode access takes 3 cycles.

Part a (1 points):

Given the above specification, how many bits is the physical address?

13- bits

Next step: we wish to determine the format of the memory address. The generic structure is shown below.

Other Bits	Row and Bank Bits	Byte on Bus

We do this by executing five load/store instructions in program order. Each takes one cycle to fetch, one cycle to decode, and at the start of the third cycle, the physical address is sent to the memory controller. There are no dependencies associated with any of the five instructions. The first instruction is fetched in cycle 1, decoded in cycle 2, accesses the memory controller in cycle 3, and completes the instruction at the end of cycle 5. The second instruction is fetched in cycle 2, decoded in cycle 3, accesses the memory controller in cycle 4, and if there is no bank conflict, completes the instruction at the end of cycle 6. The remaining instructions follow the same process. If there is a bank conflict, the subsequent access starts the cycle after the conflicting access completes.

The table below shows the cycle in which each load/store instruction completes execution.

Physical Address	Cycle Finished
0000_0011_0111_0101	5
0000_0011_0101_1111	6
0000_1000_0010_1011	9
0000_1000_1010_0011	10
0000_0101_1010_0001	13

No conflict

no bits

Part b (4 points):

Given the above accesses how many row buffer hits do we observe?

0

Part c (10 points):
Your job is to identify with the letters B (bank), C (column), R (row), BoB (Byte on Bus), etc. each bit of the memory address in the diagram below. We have provided you with 16 bits. Use as many as you need. Recall the restrictions in the format diagram above.

			C	C	R	R	R	B	R	R	R	B	Byte on Bus		

Part d (10 points):
Can this set of accesses be sped up by rearranging the bits of the physical address? If so, explain and write your new address format in the set of boxes below. If not, explain why not.

yes. we can rearrange bits to avoid Bank Conflicts.

C C R R B R R R R B B O B

Name:________________________________

Problem 5 (30 points):
We wish to add VAX-like virtual memory support to the LC-3b ISA with the following specifications:

- 16-bit virtual addresses.
- The memory management system uses the two-level page table scheme like VAX.
- Virtual memory is partitioned into two halves. User space starts at x0000 and system space starts at x8000.
- There is 16KB of physical memory.
- The page size is 256 bytes.
- The user page table starts at the beginning of the page. The system page table starts at the beginning of a frame.
- A PTE contains 16 bits and has the following form.

PTE:

Valid	Read Permission	Write Permission	0s	PFN
1 bit	1 bit	1 bit		

where a "1" in a permission bit means that user programs have permission to make that access.

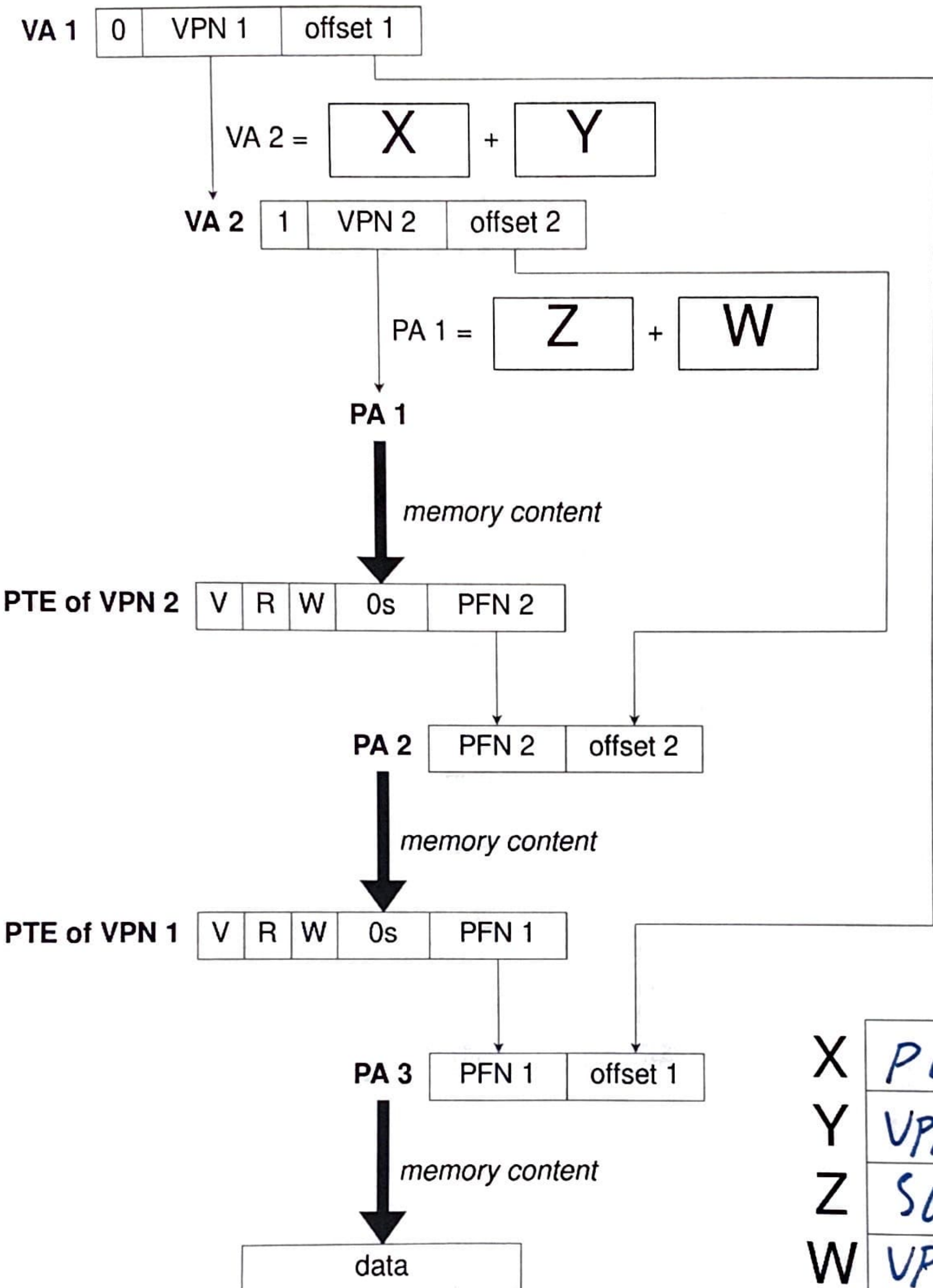
Part a (6 points):

How many pages does system space contain? **128**

How many pages does user space contain? **128**

How many frames does main memory contain? **64**

Part b (8 points):
Shown below is a VAX translation process diagram. Fill in the blanks in the table below with what X, Y, Z, and W represent (not actual values).



X	PBR
Y	VPN1 *2
Z	SBR
W	VPN2 *2

We wish to execute the instruction **LDW R1, R0, #0** with **R0 = x3FC8** initially.

Assume there is no TLB. Some of the physical memory values are provided below in ascending order. **Assume all physical memory accesses of the VAX address translation during the execution of the LDW instruction can be found below.** You do not need to worry about instruction fetch. No exceptions occur during the execution of this instruction.

Physical Address	Word Value
x053F	x802F
x057E	x802F
x07C8	x1126
x133F	x0019
x17A0	xC005
x19C8	x0324
x245C	xE03C
x2D7E	xC03A
x2FC8	x0422
x3168	x6013
x38B2	xE02D
x3C7E	x6007
x3AC8	x0928
x3C3F	xE007

Physical Memory

Part c (16 points):
Fill in the blanks in the VAX translation process diagram with actual values in hexadecimal. Recall: **The user page table starts at the beginning of the page. The system page table starts at the beginning of a frame.**