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

EE 460N Fall 2018
Y. N. Patt, Instructor
Chirag Sakhuja, John MacKay, Aniket Deshmukh, Mohammad Behnia, TAs
Exam 2
November 19, 2018

Name: Chirag Sakhuja

Problem 1 (20 points):_______

Problem 2 (25 points):_______

Problem 3 (25 points):_______

Problem 4 (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!

$$\text{PA}\quad \boxed{\text{PFN}}\overbrace{\boxed{\text{offset}}}^{12}$$

$$\text{PA}\quad \boxed{\text{tag}}\underbrace{\boxed{\text{index}}}_{7}\underbrace{\boxed{\text{offset}}}_{6}$$

Name:______________________________________

Problem 1 (20 points): Answer the following questions.

Part a (5 points): A 16KB, 2-way set associative, physically-indexed, physically-tagged cache has a line size of 64B. We wish to use it with x86 ISA, which has a page size of 4KB. Assuming no help from the Operating System, can we design the cache such that the TLB, Tag Store, and Data Store accesses can all be made at the same time?

16 KB / 64 B = $2^{14}/2^6 = 2^8$ cache lines

2^8 lines / 2 = 2^7 sets

Yes/[✓ **No**] (Circle one).

Explain.

For this design to be possible, the index and offset fields must be a total of 13 bits, but only 12 bits are guaranteed not to change during translation.

Part b (5 points): A computer implements IEEE Floating Point, with the one exception that each data element is represented with 12 bits. Six bits are used for the exponent.

What is the smallest positive normalized number that can be represented exactly? Hint: Show result as power of 2.

$$\underbrace{\boxed{\text{S}}}_{1}\underbrace{\boxed{\text{Exp}}}_{6}\underbrace{\boxed{\text{Frac}}}_{5}$$

Bias is in the middle, so $2^6/2 - 1 = 31$

1.00000×2^{-30}

2^{-30}

What is the smallest positive number that can be represented exactly? Hint: Show result as power of 2.

0.00001×2^{-30}

2^{-35}

Part c (5 points): The microarchitecture of the VAX-11/780 has a 32-bit register containing the value in hex: 0x66666666. Could this register be of any use in performing BCD arithmetic. Yes/No. Explain.

This value can be used to adjust the BCD numbers before performing an ADD.

Part d (5 points): Interrupts and Exceptions both interrupt the normal execution of a program, put the machine in a consistent state, and go to a service routine for handling. There are, however, many differences between interrupts and exceptions, for example when they are carried out, their priority level, the context within which they operate, etc., mostly due to the fact that interrupts are caused by events that are

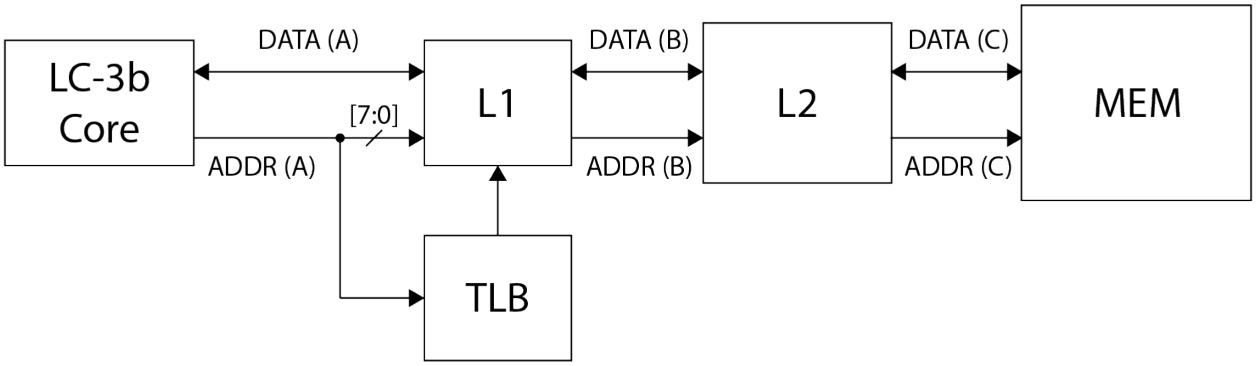
external

while exceptions are caused by events that are

internal

Name:___________________________________

Problem 2 (25 points): We have augmented the LC-3b with the memory hierarchy shown below.



Addr(A), Addr(B), and Addr(C) represent addresses that access respectively L1, L2, and Memory. Data(B) and Data(C) each transfer a full cache line.

The table below shows a sequence of five memory accesses from the LC-3b core. If ADDR(A) misses in L1, an access is required to L2. If ADDR(B) misses in L2, an access is required to Memory. Each of the five requests from the LC-3b core must complete before the next access from the LC-3b core is initiated.

ADDR(A)	ADDR(B)	ADDR(C)	Read/Write
0x3000			Read
	0x100		Read
		0x100	Read
0x3003			Write
0x3004			Read
	0x104		Read
0x3008			Read
	0x108		Read
		0x108	Read
0x8000			Read
	0x100		Write
	0x200		Read
		0x200	Read

You may make the following assumptions:

- Virtual addresses are 16 bits and the page size is 256B
- The TLB has 2 entries and is fully associative
- All accesses to the TLB are hits
- The L1 and L2 are both physically-indexed, physically-tagged
- The L1 contains 64 sets
- If a cache line is present in the L1, it will also be present in the L2 (although the contents of L2 may not be correct)
- The caches are initially empty

Name:______________________________________

Part a (3 points): How many bytes are in an L1 cache line?

4 Bytes

0x3000 and 0x3003 are hits, but 0x3004 is not

Part b (3 points): How many bytes are in an L2 cache line?

8 Bytes

0x100 and 0x104 are hits, but 0x108 is not

Part c (3 points): Fill in the VPNs and the PFNs of the two TLB entries.

VPN	PFN
x30	x1
x80	x2

x3000 translates to x100

x8000 translates to x200

Low 8 bits are page offset

Part d (4 points): Is the L1 cache write through or write back? (Circle one)

Write Through / [✓ Write Back]

Explain

The write to 0x3003 did not generate a write to the L2

Part e (4 points): Is the L2 cache write through or write back? (Circle one)

Write Through / [✓ Write Back]

Explain

The write to 0x100 did not generate a write to the memory

Part f (4 points): What is the associativity of the L1 cache?

1 Way(s)

Explain

When 0x8000 is brought into the L1, it evicts 0x3000, which is the only other location in the same set (if it was more than 1 way, it would not be necessary to evict 0x3000).

Part g (4 points): What is the minimum possible associativity of the L2 cache?

1 Way(s)

Explain

It may seem like the same concept as part f except with locations 0x100 and 0x200. However, we do not have enough information to know whether those addresses map to the same set or not, so the minimum possible associativity is still 1.

Problem 3 (25 points): Let us use one of the unused opcodes to add an instruction DOTPRODUCT (i.e., dot product) to the LC-3b ISA. Its format will be

15			12	11 9	8 6	5	4 0
1	**0**	**1**	**0**	**DR/SRA**	**SRB**	**0**	**vlen5**

The DOTPRODUCT of two vectors is computed as shown below:

$$\sum_{i=0}^{n-1} A[i] \times B[i]$$

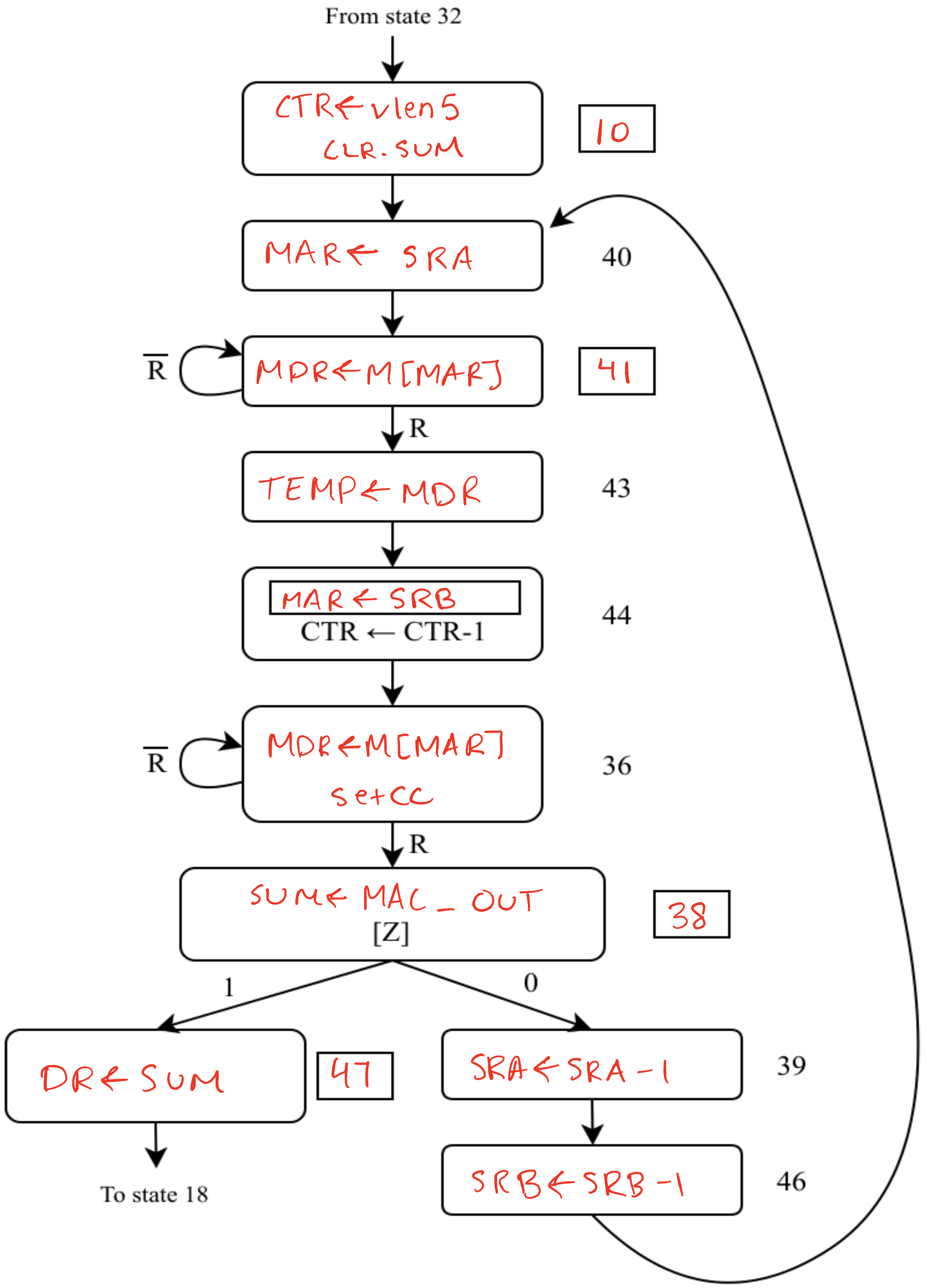
The two vectors are stored in memory. Their starting addresses are contained in SRA and SRB, and their length is specified as an immediate 5-bit value (vlen5). The instruction stores the result of the dot product in the register specified by DR. Assume vlen5 is not zero.

For this problem, you can assume no overflow will occur. Note: execution of this instruction will destroy the initial contents of SRA and SRB.

Your job: augment the LC-3b state machine, data path and microsequencer shown on the next three pages to add DOTPRODUCT to the LC-3b ISA.

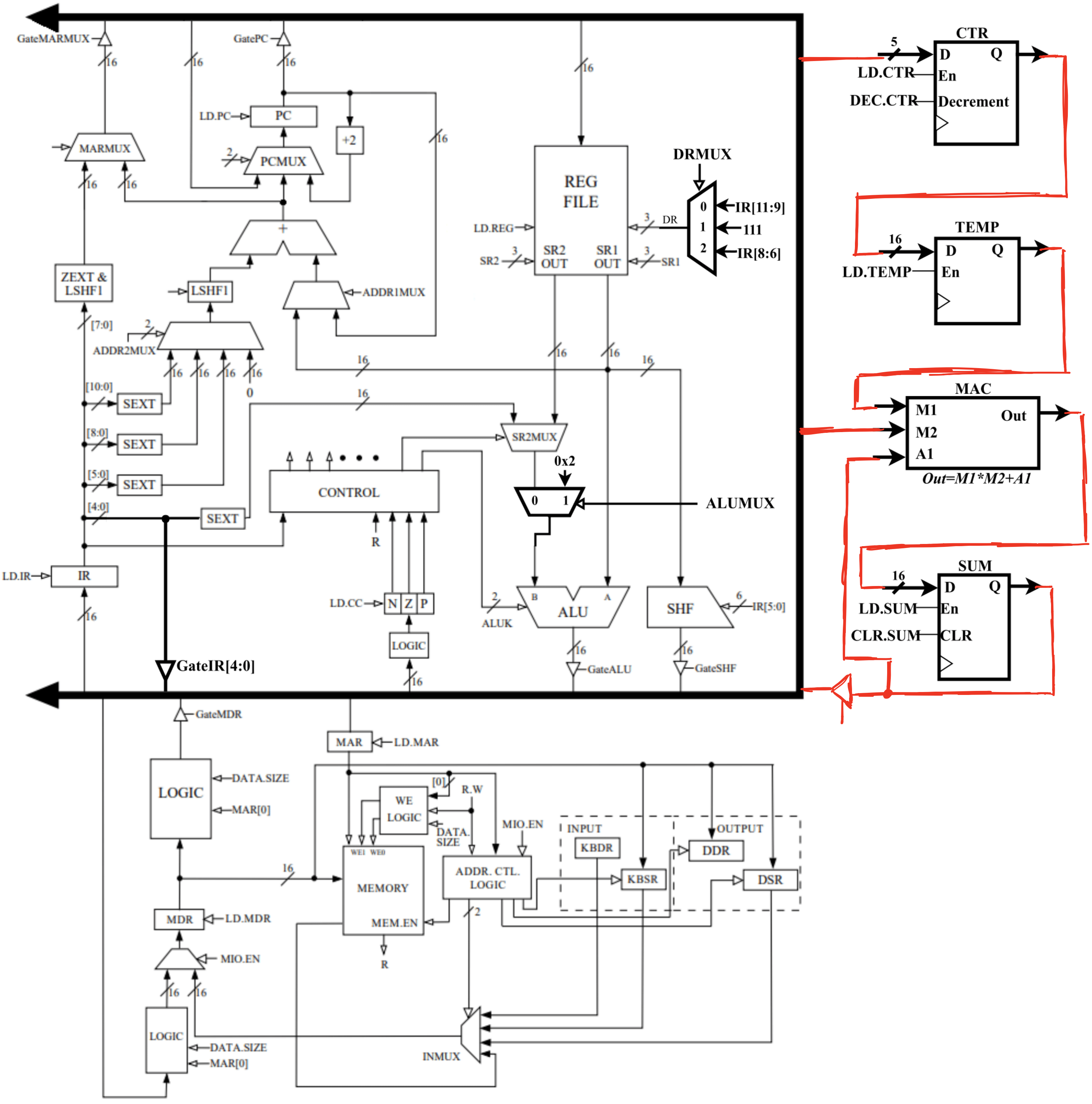
Name:____

Part a, The state machine (12 points): From decode (state 32), ten states are needed to complete the execution of DOTPRODUCT. One of the states (state 44) has been partially specified. Your job is to complete the specifications of all the states and add the missing state numbers.



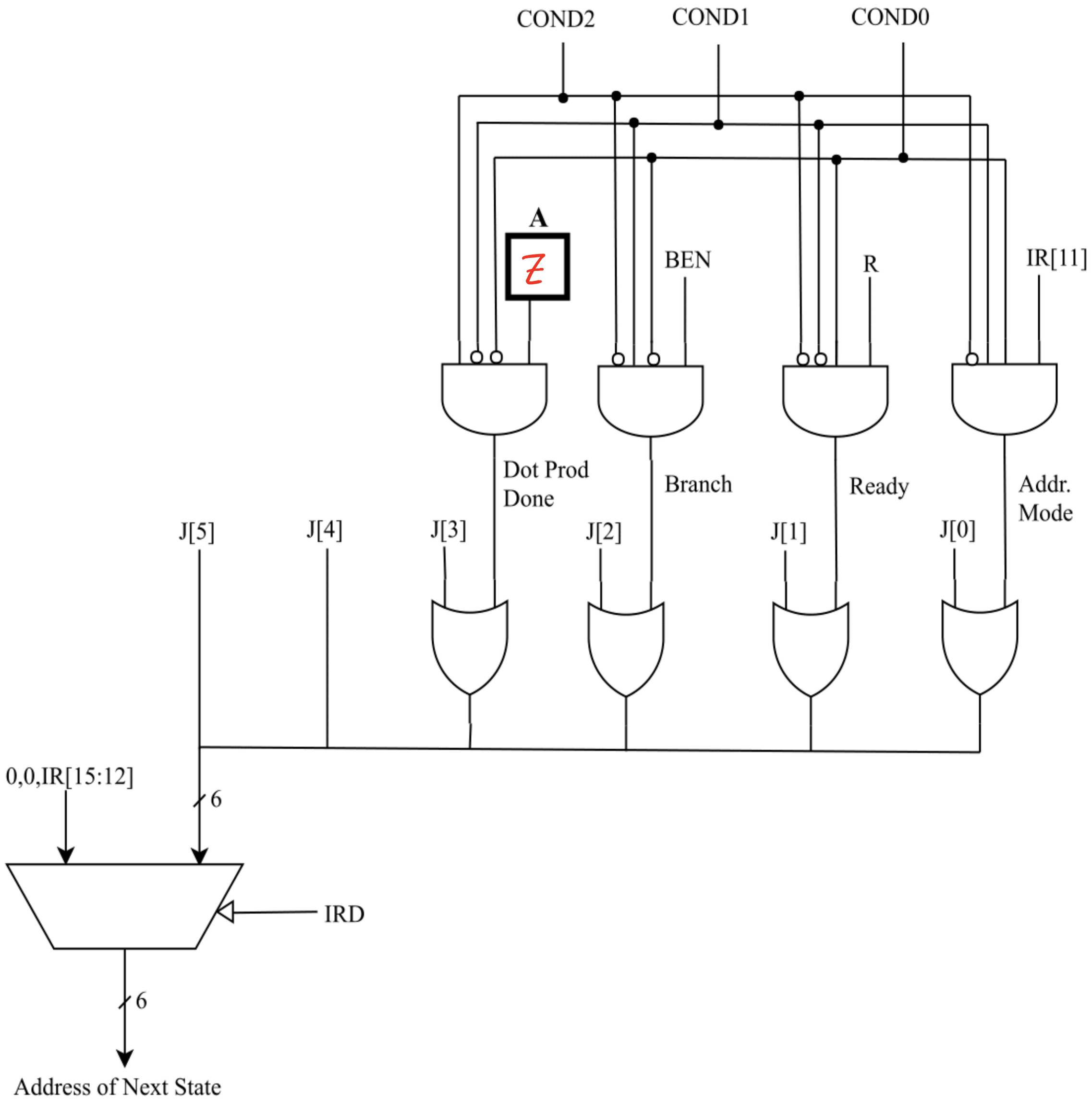
Name:_______________________________________

Part b, The data path (10 points): We have added GateIR[4:0], ALUMUX, made changes to DRMUX, provided registers for CTR (with built-in decrement functionality), TEMP, and SUM as well as a multiply-and-accumulate (MAC) unit. The MAC computes $M1 \times M2 + A1$. Your job is to implement the changes you made in Part a by connecting the necessary structures to the LC-3b datapath. You are free to add control signals and tri-state buffers as needed.



Name:___________________________________

Part c, The microsequencer (3 points): To make this work, we need to add a COND2 control signal to the micorsequencer. The only thing missing to complete the change to the microsequencer is the box labeled A. Your job: fill in the box labeled A.



DISCLAIMER: THIS QUESTION MAY LEGITIMATELY MAKE YOU CRY

Name:______________________________________

Problem 4 (30 points): Suppose the LC-3b ISA had a 12-bit, byte-addressable, virtual address space with two levels of virtual to physical translation, similar to the VAX.

A PTE is shown below:

V	M	ACC	0...0	PFN

It includes a Valid bit, a Modify bit, a 2 bit Access Control field, some number of unused bits (i.e., 0..0), and the PFN. The low bits of the PTE are used for the PFN.

The access control bits are defined as follows:
 00: none
 01: read-only
 10: read-write
 11: —

The virtual address space is divided evenly into two regions. The first half is user space, the second half is system space.

The user space page table starts at the beginning of a page. The system space page table starts at the beginning of a frame. We require 1/4 of physical memory to store the entire system page table.

A user program fetches and executes one LC-3b instruction, resulting in six accesses to physical memory, as shown by the following table:

VA	PA	Data
—	x00EA	x9...4
x08B0	x90	x9...2
x0100	x40	x2862
—	x00EE	x9...5
x08F6	xB6	x9...1
x0572	x32	x10A0

Note: Since the size of the PTE has not been given, entire PTEs are not shown in the above table.

Part a (17 points): Fill in the entries for the following:

		Physical Address Space	2^9 B
PTE Size	2	Page Size	32 B
SBR	x00E0	UBR	x8C6

x08F6 = VPN[x0572] * PTESIZE + UBR

x08B0 = VPN[x0100] * PTESIZE + UBR

x0046 = (VPN[x0572] - VPN[x0100]) * PTESIZE

0100 0110 = (VPN[0000 0101 0111 0010] - VPN[0000 0001 0000 0000]) * PTESIZE

0100 0110 = VPN[0000 0100 011|1 0010] << log(PTESIZE)

Division b/w VPN and offset must be here for math to work (and PTE SIZE of 2)

Name:______________________________________

Part b (3 points): After the instruction is executed, the register file is as shown:

x08F6 = x2A + VBR

VPN[0x0572]

$$\begin{array}{r} \overset{E,1}{x08\cancel{F}6} \\ -\ x2A \\ \hline x8C6 \end{array}$$

x00EE = xE + SBR

VPN[0x08F6]

2^6 virtual pages in system space, each with 2B PTEs = 2^7B

1/4 of PM

Register	Value
R0	x0550
R1	x0590
R2	x00A0
R3	x0200
R4	xFFA0
R5	x000C
R6	x0010
R7	x0100

What LC-3b instruction was executed?

x1E = 011110

-x1E = 100010

$\boxed{\text{LDB R4, R1, x22}}$

0010 100 001 100010

Part c (10 points): Complete the entries in the memory access table shown on the previous page.

0x0572 = 0101011|1 0010

110010

0x08F6 = 1000111|1 0110

1011 0110

0x0100 = 0001 000|0 0000

10 00000

0x08B0 = 1000 101|1 0000

100 10000