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
Final Exam
December 14, 2018

Name: Solution

Part A	**Part B**
Problem 1 (10 points):_____	Problem 6 (20 points):_____
Problem 2 (10 points):_____	Problem 7 (20 points):_____
Problem 3 (10 points):_____	Problem 8 (20 points):_____
Problem 4 (10 points):_____	Problem 9 (20 points):_____
Problem 5 (10 points):_____	
Part A Total (50 points):_____	Part B Total (80 points):_____

Total (130 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:_____

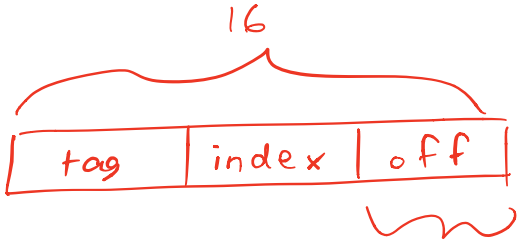
GOOD LUCK!

Name:_____

Problem 1 (10 points): An LC-3b system, with a 16-bit address space, has a 16 KB physically indexed, physically tagged cache. The cache is 4 way set associative, write-back, and uses a victim/next-victim replacement policy. A cache line is 64 B.

How many bits of storage are required for the tag store?

2^{11} bits

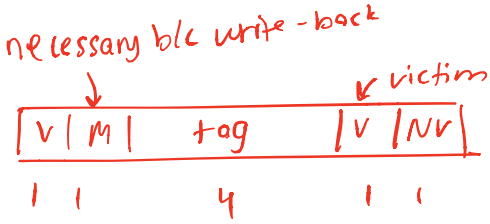


How many index bits?

$$\frac{2^{14}\text{ B cache}}{4\text{ ways} \cdot 64\text{B lines}} = 2^6 \text{ sets}$$

Must be 6 index bits, so 4 tag bits

tag store entry



8 bits per tag

$$\frac{2^{14}\text{B cache}}{64\text{B line}} = (2^8 \text{ lines})(8 \text{ bits})$$

Name:_____

Problem 2 (10 points): A given program takes time T to run on a processor. To improve performance, you consider using a newer processor that provides a **2x speedup** over the older one. You also have the option of using any number of older processors to create a multi-core system. After analyzing the program, you find that **50%** of the program can be parallelized.

Part a (5 points): Which option performs better? Explain.

The new processor because you would need an infinite number of old processors to achieve the same speedup

To further improve performance you decide to construct a system consisting of the new processor and four older processors together. Due to energy constraints, either only the new processor or the four-core system can run at any given time.

Part b (5 points): What is the best speedup over the original execution time (T), given this configuration?

$\frac{8}{3}$ (see below)

0.5T (the serial part) can be reduced to 0.25T on the new processor

0.5T (the parallel part) can be reduced to 0.125T on the 4 old processors

New time is 0.375T, so speedup is $\frac{1}{0.375} = \frac{8}{3}$

Name:_____

Problem 3 (10 points): For the LC-3b, an in-order pipeline was designed consisting of 5 stages: Fetch, Decode, Address Generation/Execute, Memory and Writeback. Each stage takes one cycle. Assume registers and condition codes are read in the Address Generation/Execute stage. Consider two different schemes:

1. No branch prediction or data forwarding. For a branch instruction, Fetch of the instruction following the branch occurs in the same cycle as when the branch reaches Writeback.
2. Perfect branch prediction, no data forwarding. Fetch of the next instruction following the branch occurs in the next cycle after the branch instruction is fetched.

The function FOO is executed on this processor.

```
FOO   AND R0, R0, #0
      ADD R0, R0, #-1
      BRZ SKIP
      ADD R6, R6, #1
SKIP  ADD R6, R6, #-1
      RET
```

The table below contains the instructions in FOO and the cycle in which each reaches the Writeback stage for the two schemes.

Your task: Fill in the missing entries.

Instruction	Scheme 1	Scheme 2
AND R0, R0, #0	5	5
ADD R0, R0, #-1	8	8
BRZ SKIP	11	11
ADD R6, R6, #1	15	12
ADD R6, R6, #-1	18	15
RET	19	16

We have provided on the next page four copies of a timing diagram that you can use for scratch work. Use as much or as little of it as you need.

See work below

w/o branch prediction

Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
AND R0, R0, #0	F	D	A	M	W															
ADD R0, R0, #-1		F	D	A	A	A	M	W												
BRZ SKIP			F	D	D	D	A	A	A	M	W									
ADD R6, R6, #1				F	F	F	F	F	F	F	F	D	A	M	W					
ADD R6, R6, #-1												F	D	A	A	A	M	W		
RET													F	D	D	D	A	M	W	

w/ branch prediction

Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
AND R0, R0, #0	F	D	A	M	W															
ADD R0, R0, #-1		F	D	A	A	A	M	W												
BRZ SKIP			F	D	D	D	A	A	A	M	W									
ADD R6, R6, #1				F	F	F	D	D	D	A	M	W								
ADD R6, R6, #-1							F	F	F	D	A	A	A	M	W					
RET										F	D	D	D	A	M	W				

Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
AND R0, R0, #0																				
ADD R0, R0, #-1																				
BRZ SKIP																				
ADD R6, R6, #1																				
ADD R6, R6, #-1																				
RET																				

Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
AND R0, R0, #0																				
ADD R0, R0, #-1																				
BRZ SKIP																				
ADD R6, R6, #1																				
ADD R6, R6, #-1																				
RET																				

Name:_____

Problem 4 (10 points): Consider an N-bit IEEE-style floating point representation. The BIAS is determined in exactly the same way as it is for 32-bit and 64-bit IEEE representations. For our N-bit representation, we are able to represent the value $12\frac{1}{8}$ exactly, but we cannot represent the value $34\frac{1}{4}$ exactly. The smallest positive normalized value we can represent is 2^{-62}.

Of our N bits, how many are fraction bits?

Number of Fraction Bits: 6

Of our N bits, how many are exponents bits?

Number of Exponent Bits: 7

What is N?

N: 14

sign bit!

What is the bias?

Bias: 63

$12\frac{1}{8} \Rightarrow 1100.001 = 1.100001 \times 2^3$

$34\frac{1}{4} \Rightarrow 100010.01 = 1.0001001 \times 2^5$

Must be 6 fraction bits

Minimum normalized exponent = 1 − BIAS

$-62 = 1 - BIAS$

$63 = BIAS$

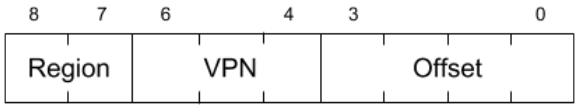
In IEEE style floating point, an exponent field of 0111…1 represents the actual exponent 0 (i.e. the bias is in the middle)

$63 \Rightarrow 0111111$

7 bits

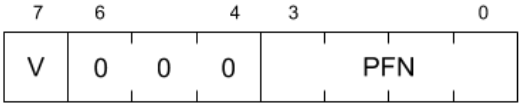
Name:_____

Problem 5 (10 points): Recall my handout on virtual addressing, showing the VAX process but reducing the sizes of everything to make the example less tedious. For example: VA is 9 bits, as follows:



Physical memory is 256 bytes.

Page table entries are 8 bits each and are broken down as follows.



The following snippets of physical memory are known.

Addr	Data
...	...
0x10	0x0F
0x11	0x88
0x12	0x83
0x13	0x89
...	...

Addr	Data
...	...
0x30	0x80
0x31	0x89
0x32	0x88
0x33	0x8A
...	...

Addr	Data
...	...
0x90	0x0F
0x91	0x88
0x92	0x83
0x93	0x89
...	...

Addr	Data
...	...
0xF0	0x03
0xF1	0x0E
0xF2	0x81
0xF3	0x80
...	...

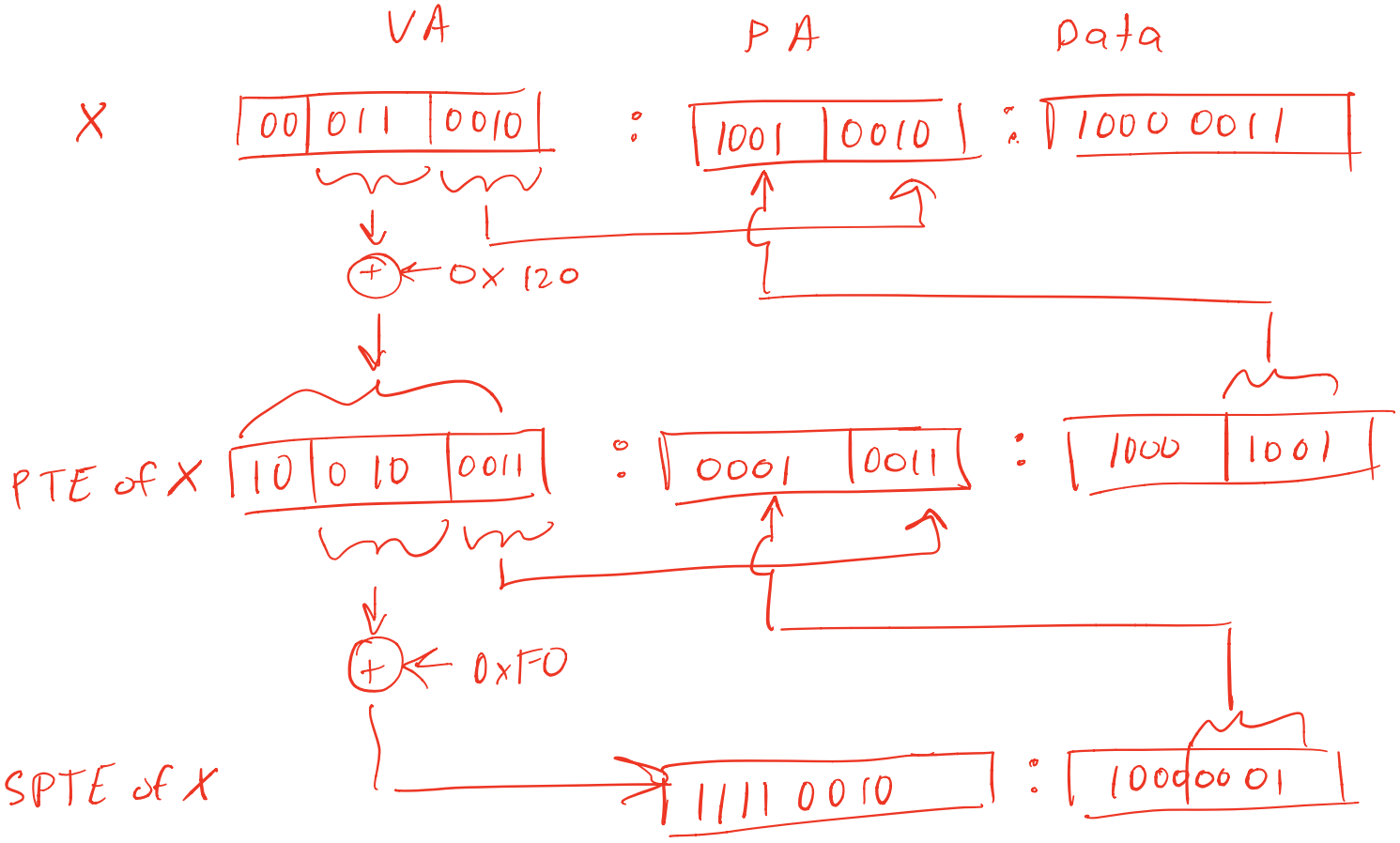
The instruction LOADBYTE R12,A is executed, where the virtual address A is 0x032.

After the instruction is fetched and decoded, three physical memory accesses are necessary to obtain the byte to be loaded into R12. Fill in the physical addresses and data that are accessed.

The P0BR and SBR are 0x120 and 0xF0, respectively.

see next page for work

	Addr	Data
Access 1	0x F2	0x81
Access 2	0x13	0x 89
Access 3	0x 92	0x 83


VA
PA
Data
X
00 011 0010
1001 0010
1000 0011
0x120
PTE of X
10 010 0011
0001 0011
1000 1001
0xF0
SPTE of X
1111 0010
10000001

Name:_____

Problem 6 (20 points): This problem is similar to the Tomasulo question from Exam 1, with some key differences. Consider an out-of-order processor which executes its instructions based on the Tomasulo algorithm ***with in-order retirement***. The ISA specifies 8 registers, R0 to R7. The execute stage of the pipeline contains one pipelined adder and one pipelined multiplier. Fetch and Decode take one cycle each. The ADD instruction takes additional 4 cycles to execute; the MUL instruction takes additional 5 cycles to execute. At the end of the last cycle of execution, each instruction will store its result into any reservation stations waiting for it and its ROB (Reorder Buffer) entry. Both ADD and MUL need one additional cycle to retire. Only a single instruction can retire at a time.

The adder and multiplier each have 2-entry reservation stations. The reservation stations and the ROB are initially empty and are filled from top to bottom. Each instruction remains in the reservation station until the end of the cycle in which it writes its result to its ROB entry.

In this example, the computer will execute a program fragment consisting of five instructions.

Part a (10 points): Determine the five instructions in the program fragment.

	Opcode	**DR**	**SR1**	**SR2**
I1	ADD	R4	R2	R3
I2	MUL	R3	R1	R4
I3	MUL	R1	R0	R1
I4	ADD	R3	R3	R3
I5	ADD	R4	R0	R4

Just need to look at values in RF and ROB to reconstruct this

To help you we have included the state of the register file before the first instruction is fetched, and after all five instructions have retired (the bolded values indicate the values that changed). We have also included a snapshot of the ROB (Reorder Buffer) at the beginning of cycle X.

Note that each ROB entry contains 3 bits of state information. The Valid (V) bit indicates if the entry is in use, i.e. V=0 indicates an empty slot in the Reorder Buffer. The Executed bit indicates that the entry corresponds to an instruction that has successfully executed. The Retirement bit indicates that the executed instruction has retired.

	V	Tag	Value
R0	1	-	4
R1	1	-	11
R2	1	-	-6
R3	1	-	3
R4	1	-	205
R5	1	-	49
R6	1	-	12
R7	1	-	25

Register File Before Cycle 1

	V	Tag	Value
R0	1	-	4
R1	1	-	**44**
R2	1	-	-6
R3	1	-	**-66**
R4	1	-	**1**
R5	1	-	49
R6	1	-	12
R7	1	-	25

Register File After Execution

V	Executed	Retired	DR	Value
0	1	1	R4	-3
0	1	1	R3	-33
1	1	0	R1	44
1	0	0	R3	-
1	1	0	R4	1
0	0	0	-	-

ROB at the beginning of cycle X

Part b (3 points): What is the value of X?

13

PROBLEM CONTINUES ON NEXT PAGE

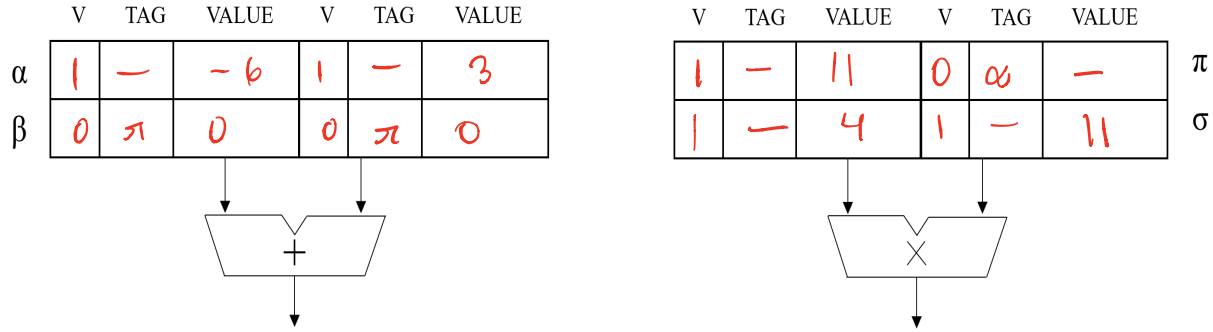
see work on next page

Name:_____

We have included a timing diagram template for your scratch work. Use as much of it as you deem useful.

Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
I1	F	D	E	E	E	E	W													
I2		F	D	–	–	–	E	E	E	E	E	W								
I3			F	D	E	E	E	E	E	–	–	–	W							
I4				F	D	–	–	–	–	–	–	E	E	E	E	W				
I5					F	D	E	E	E	E	–	–	–	–	–	–	W			

Part c (7 points): Complete the values of the reservation stations, ROB, and register file after cycle 5.

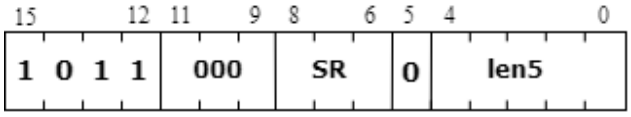


	V	Tag	Value
R0	1	–	4
R1	0	5	–
R2	1	–	-6
R3	0	J3	–
R4	0	α	–
R5	1	–	49
R6	1	–	12
R7	1	–	25

V	Executed	Retired	DR	Value
1	0	0	R4	–
1	0	0	R3	–
1	0	0	R1	–
1	0	0	R3	–
0	–	–	–	–
0	–	–	–	–

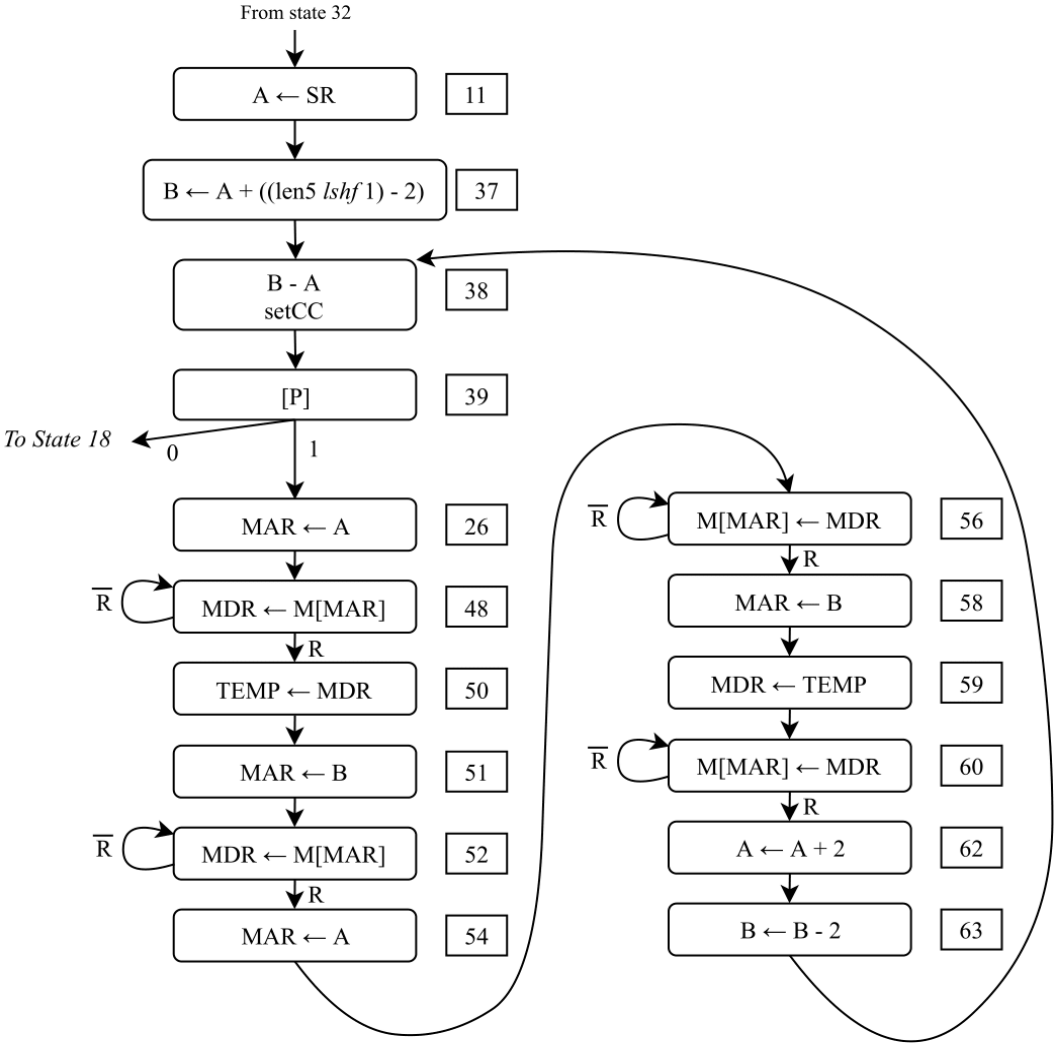
Name:_____

Problem 7 (20 points): The unused opcode '1011' in the LC-3b ISA is used for a new instruction. The new instruction has the following format.



The state machine for the new instruction is shown below.

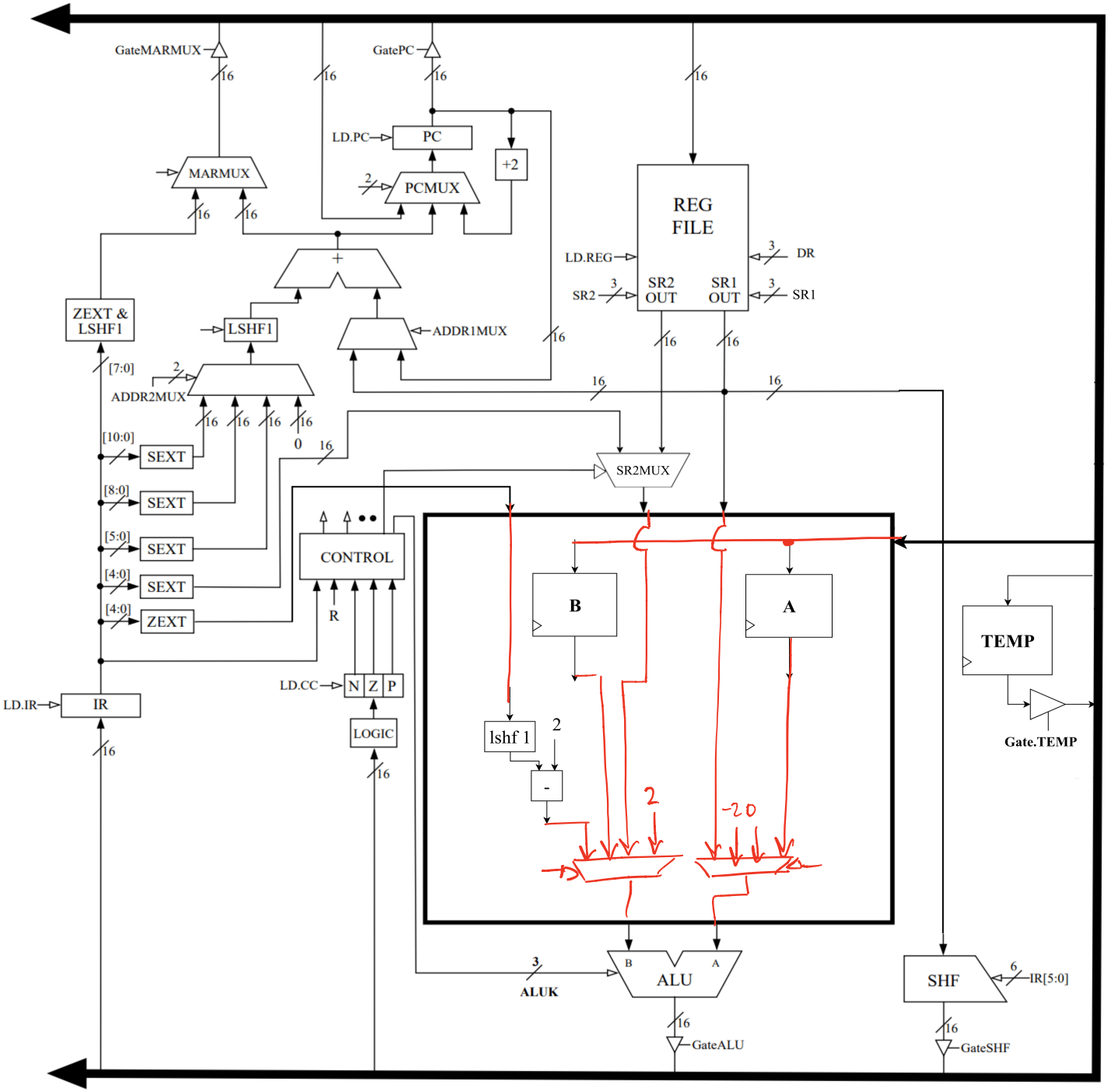
Your job: Augment the LC-3b data path and microsequencer shown on the next two pages to add the new instruction to the LC-3b ISA, and describe what the new instruction does.



Name:_____

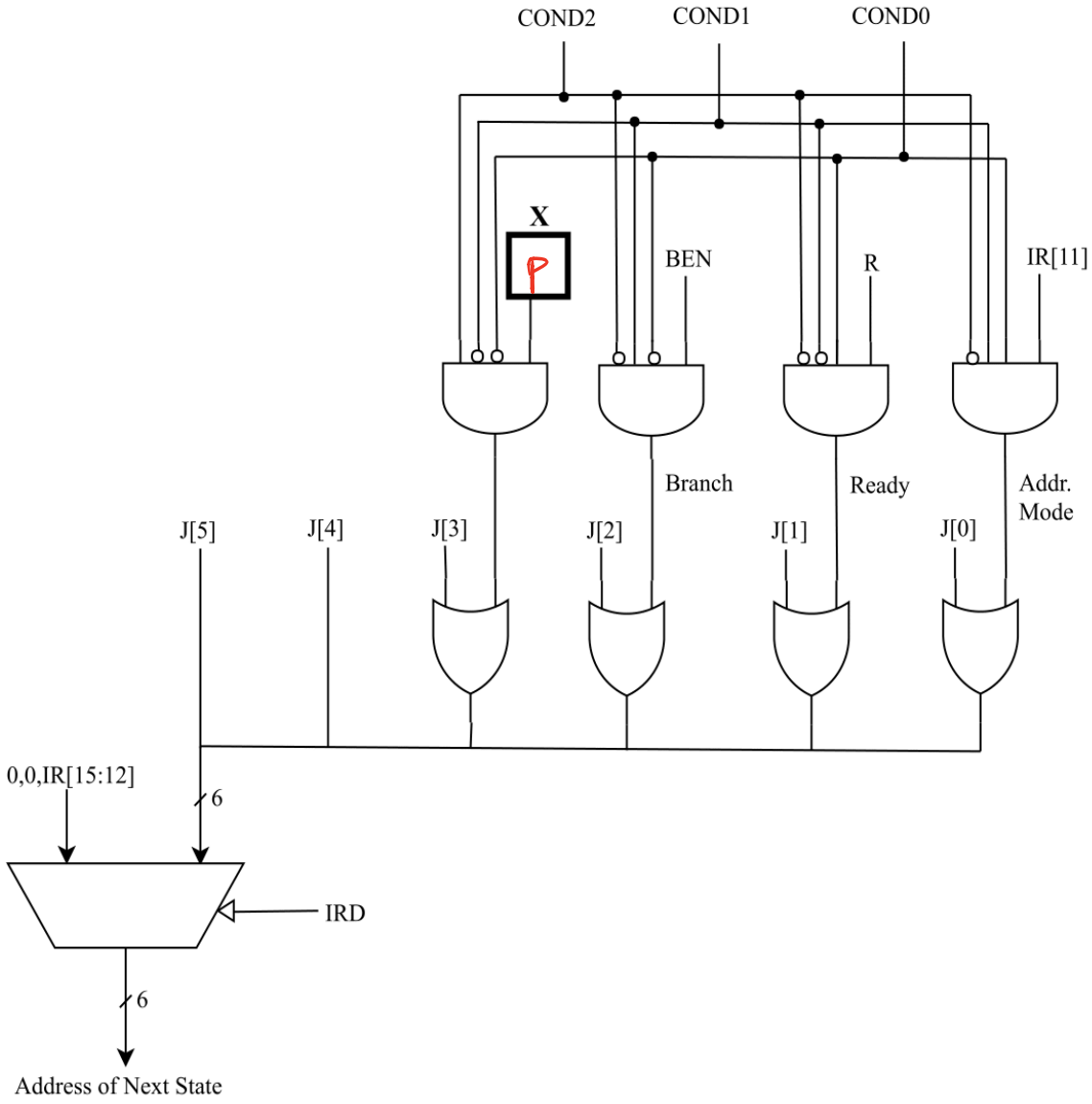
Part a, The data path (9 points): We have provided registers for A, B, and TEMP, as well as GateTEMP tri-state buffer, a left-shift-by-1 unit, and a subtractor. We also provide inputs from the bus and ZEXT IR[4:0] to the large box shown. Your job is to implement the changes needed to support the new instruction by adding the necessary logic to the box and connecting the necessary structures to the LC-3b datapath. *You may only add two 4:1 muxes, wires, and constant values.* You may not add additional inputs or outputs to the box.

Note: ALUK is a 3-bit code in order to provide the ALU with the ability to subtract. This is needed in state 38.



Name:_____

Part b, The microsequencer (3 points): To make this work, we need to add to the microsequencer a COND2 control signal and an input X. Fill in the box labeled X.



Part c (8 points): In 15 words or fewer, explain what the new instruction does. *(Hint: You really only need three words)*

Reverse an array

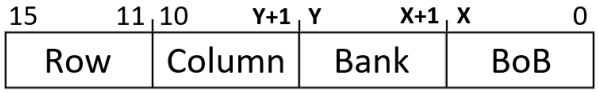
Name:_____

Problem 8 (20 points): A byte addressable processor with a 32-bit wide data bus executes the following function.

```
int check_palindrome (int * input) {
    int flag = 0;
    for(int i = 0; i < 16; i++) {
        if(input[i] != input[31-i]) flag = -1;
    }
    return flag;
}
```

This function determines whether an array of 32 integers is a palindrome. Integers are 32 bits. You may assume local variables are stored in registers and only array accesses go to memory. The array begins at address x8000.

Suppose we have an 8-way interleaved DRAM memory, with the following address scheme:



- A row access takes 6 cycles, and subsequent column accesses take 1 cycle.
- Memory address and data buses are independent.
- Memory requests arrive at the memory controller 1 per cycle.
- Memory requests must be issued in order.
- If a request comes to a busy bank, it (and future requests) must wait until the bank is free.

Part a (2 points): Fill in the values for X and Y.

2 bits for 32-bit BoB

3 bits for bank

X [1] Y [4]

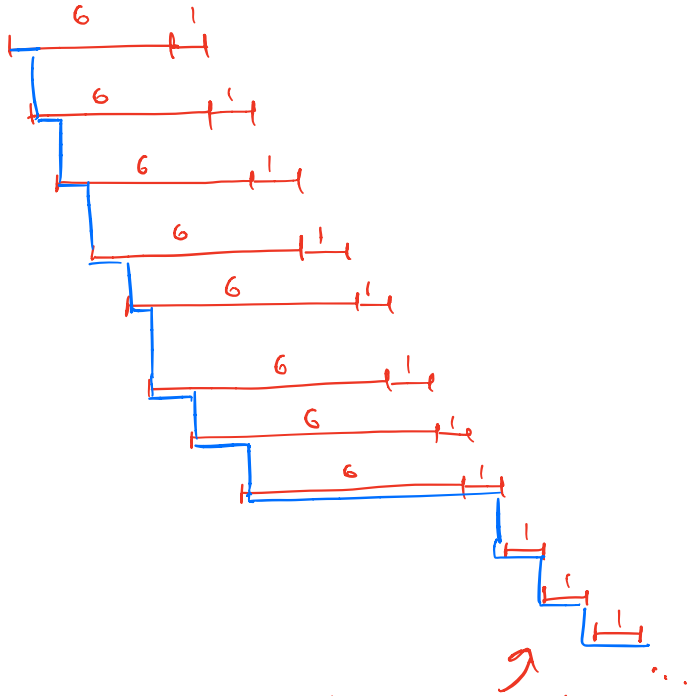
Part b (12 points): To execute this function, 32 memory locations must be accessed. What is the total number of clock cycles that these 32 memory accesses take?

[38]

Part c (6 points): Will moving to a 16-way interleaved memory system improve performance? Explain.

No because all accesses are still interleaved at some point and take 1 cycle each

0x8000	10000	000000	000	00
0x807C	10000	000011	111	00
0x8004	10000	000000	001	00
0x8078	10000	000011	110	00
0x8008	10000	000000	010	00
0x8074	10000	000011	101	00
0x800C	10000	000000	011	00
0x8070	10000	000011	100	00
0x8010	10000	000000	100	00



Since the row doesn't change, the remaining accesses all hit in the row buffers once each bank has opened it

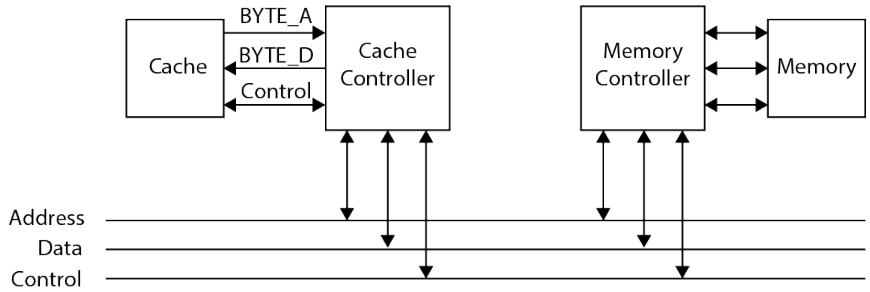
Blue line counts the number of total cycles

$7 + 7 + 24 = 38$

↑ remaining accesses after all banks are open

Name:_____

Problem 9 (20 points): The asynchronous bus discussed in class has two controllers relevant to this question, a cache controller and a memory controller. The bus can transfer one byte at a time.



Your job is to implement the state machine for a cache controller so it can load a cache line from memory in one bus cycle. A cache line consists of eight bytes.

The cache controller gets the following inputs from the cache:

- **REQ:** a request from the cache to get a cache line from memory
- **BYTE_A:** address of the next byte to be accessed (initially the address of the first byte in the cache line; updated in the cache and latched by the cache controller)
- **CACHE_I:** a control signal from the cache to the cache controller
- **DONE:** 1 if all of the bytes have been transferred; 0 if more bytes still need to be transferred

The cache controller has the following outputs to the cache:

- **BYTE_D:** byte of data
- **CACHE_O:** a control signal from the cache controller to the cache

The cache controller gets the following inputs from the bus:

- **BBSY_I:** 1 if the bus is busy; 0 if it is free
- **SSYN:** a control signal from the slave to the master
- **DATA:** the byte on the bus (produced by the memory controller and latched by the cache controller)

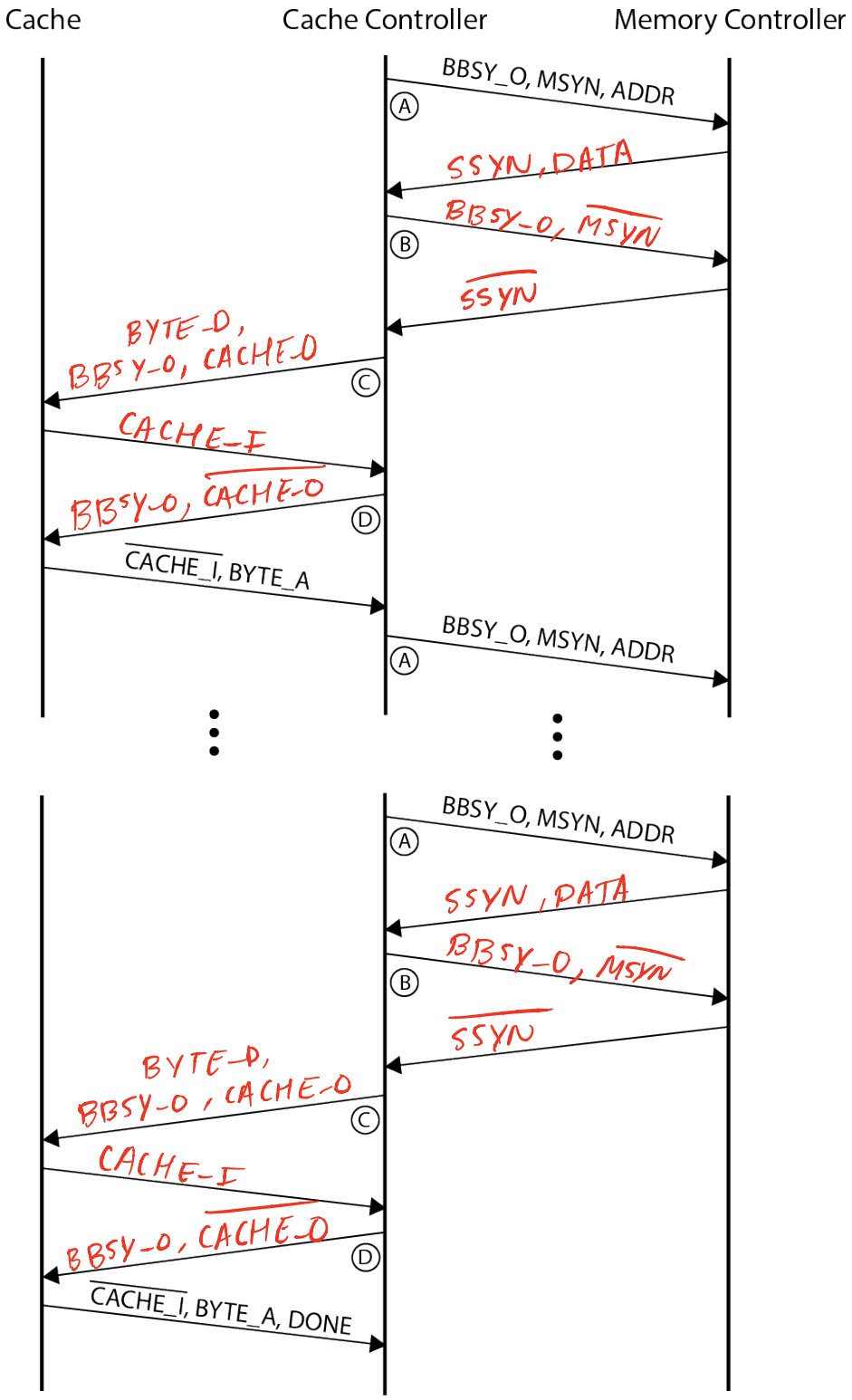
The cache controller has the following outputs to the bus:

- **BBSY_O:** 1 if the bus is busy; 0 if it is free
- **MSYN:** a control signal from the master to the slave
- **ADDR:** the address to access on the bus

PROBLEM CONTINUES ON NEXT PAGE

Name:_____

Part a (10 points): Complete the following asynchronous timing diagram by showing all the signals that are transmitted. Note that we have labeled the states A, B, C, and D of the cache controller at various points in time. Those states correspond to the states in the state machine on the next page.



Name:_____

Part b (10 points): Complete the state machine for the cache controller by filling in the bolded boxes showing how it loads a cache line from memory in one bus cycle.