University of California, Davis
Department of Computer Science

ECS-154A Computer Architecture

Professor Farrens

Winter 2025

Assessment 3

Cheating on exams is prohibited by the Academic Code of Conduct. I understand that if I am caught cheating on this exam my scores will be negated and I will receive no credit on the exam. Persons caught cheating will also face University disciplinary actions.

Name: Diego Tyner

Signature: Diego T

Student ID number: 921403964

For the written/short answer questions with a point value of 3 or less, the correct answer requires fewer than 8 words. If the point value of the question is 4 or more, you can use up to 16 words.

1] [3] What is the 3-term CPU time equation?

$$\text{CP.U}_{time} = \frac{\text{Instructs}}{\text{Program}} \times \frac{\overset{\text{Clock}}{\text{Cycles}}}{\text{Instruct}} \times \frac{\text{Time}}{\text{Clock cycles}}$$

2] [3] What is the Average Memory Access Time equation?

$$\text{AMAT} = \text{Hit Time} + (\text{Miss rate} \cdot \text{Miss Penalty})$$

3] [3] Caches, and in fact the entire memory hierarchy is possible because programs exhibit what principle? What are the two types of it?

Locality Principle. Spatial and Temporal

4] [3] When dealing with writes to a cache that hit, what are the two possible approaches? Which approach requires a dirty bit?

Write-through

Write-back. Needs dirty bit.

5] [3] What does the TLB do / how does it work?

TLB gives virtual physical translation. Has address mappings.

6] [2] What are the two main ways to define performance?

Throughput and responce time / latency.

7] [4] Which one is used more now when reporting the performance of a modern chip? Why?

Throughput, today programs are larger and have more instructions.

8] [4] You have written microcode for a 1-bus machine. If you are asked to rewrite the microcode for a 2-bus machine, is the number of cycles per instruction likely to go up, down, or remain the same? Why?

Down because the latency in waiting for the bus to open is reduced.

9] [4] What is a benchmark program?

A program that emulates the requirements of real usage. Used to measure performance.

10] [4] Do benchmark programs remain valid indefinitely? Why or why not?

No, computing requirements change over time.

11] [6] An important program spends 70% of its time doing Integer operations, and 30% of its time doing floating point arithmetic. By redesigning the hardware you can make the Integer unit 40% faster (take 60% as long), or you can make the Floating Point unit 90% faster (take 10% as long). Which should you do, and why? (Show your work).

SMALLER.

42?

Int .6

$100:\ 70 \longrightarrow 30+70(.6) = \boxed{72}$

Float

$30 \xrightarrow[.1]{} 70+30(.1) = 73$

$$\begin{array}{r} 70.0 \\ .6\ \times \\ \hline 42.00 \end{array}$$

Should use the integer optimizing redesign.

12] [7] Given a 2 Megabyte physical memory, a 36 bit Virtual address, and a page size of 4K bytes,

Show how the virtual address is partitioned.

← 36 → Page size

24	12

$1k = 2^{10}$

$4 = 2^{2}$ $= 2^{12}$

$1mb = 2^{20}$

$2MB = 2^{21}$

How many frames are there in memory? 9

How many entries are there in the page table? $\frac{\text{Physical Size}}{\text{Page size}} = \frac{2^{21}}{2^{12}} = \boxed{2^{9}}$

How wide is each entry? $\frac{2^{12}}{2^{9}} = \overset{3}{\cancel{8}} \longrightarrow \boxed{3}$

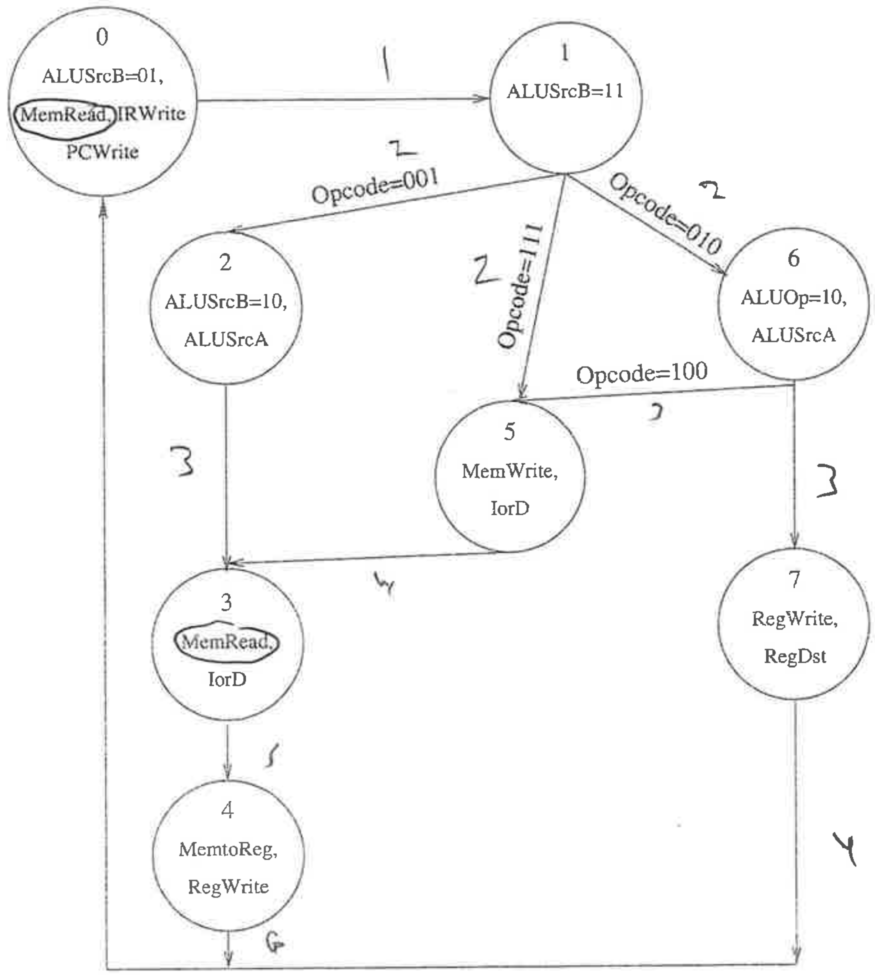
Is there a problem with this configuration? If so, show/explain how you can fix it.

No

13] [2] Assuming a 17-bit address and a 200-byte Fully Associative cache with a linesize=8, show how an address is partitioned/interpreted by the cache and label each partition.

$\underbrace{0\ 1\ 1\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 1\ 1}_{\text{TAG}}\ \big|\ \underbrace{0\ 0\ 1}_{\text{offset}}$

Here is the state diagram for a "random" machine:



14] [3] Assuming there are 3 state variables (Y2-Y0), that State0 = $\overline{Y2}*\overline{Y1}*\overline{Y0}$ (000) and State7 = Y2*Y1*Y0 (111), and that the 3 opcode bits are in I7-I5, write down the exact boolean equation for the **MemRead** signal.

000|011

$\overline{Y_2}\,\overline{Y_1}\,\overline{Y_0} + \overline{Y_2}Y_1Y_0 \implies \overline{Y_2}\,\overline{(Y_1 \oplus Y_0)}$

15] [3] Assuming the same situation as in the previous question, write down the exact boolean equation for **NextState2**.

$\overline{Y_2}\,\overline{Y_1}\,Y_0\,I_7\,\overline{I_6}\,\overline{I_5}$

16] [2] How many total cycles does the shortest instruction take to execute, start to finish? How many does the longest instruction take?

Shortest: 4

Longest: 6

In this question, we are going to wire up an 8-bit machine. This machine will use a 1-operand format, meaning that instructions are of the type ACC=ACC+R. So, for example, "Add R" is ACC=ACC+R.

The machine is byte-addressable. Immediates are sign-extended, and jumps are done by adding the sign-extended Immediate field to the PC.

The machine has 2 different instruction formats: A and B.

A-type:	Opcode 7-4	extra 3-0
B-type:	Opcode 7-4	Immediate 3-0

The ALU can perform 7 functions, written this way: OP [ALU2 ALU1 ALU0]

Operation	**ALU2**	**ALU1**	**ALU0**
Add	0	0	1
Sub	0	1	0
NOT B	0	1	1
AND	1	0	0
OR	1	0	1
Increment A	1	1	0
XOR	1	1	1

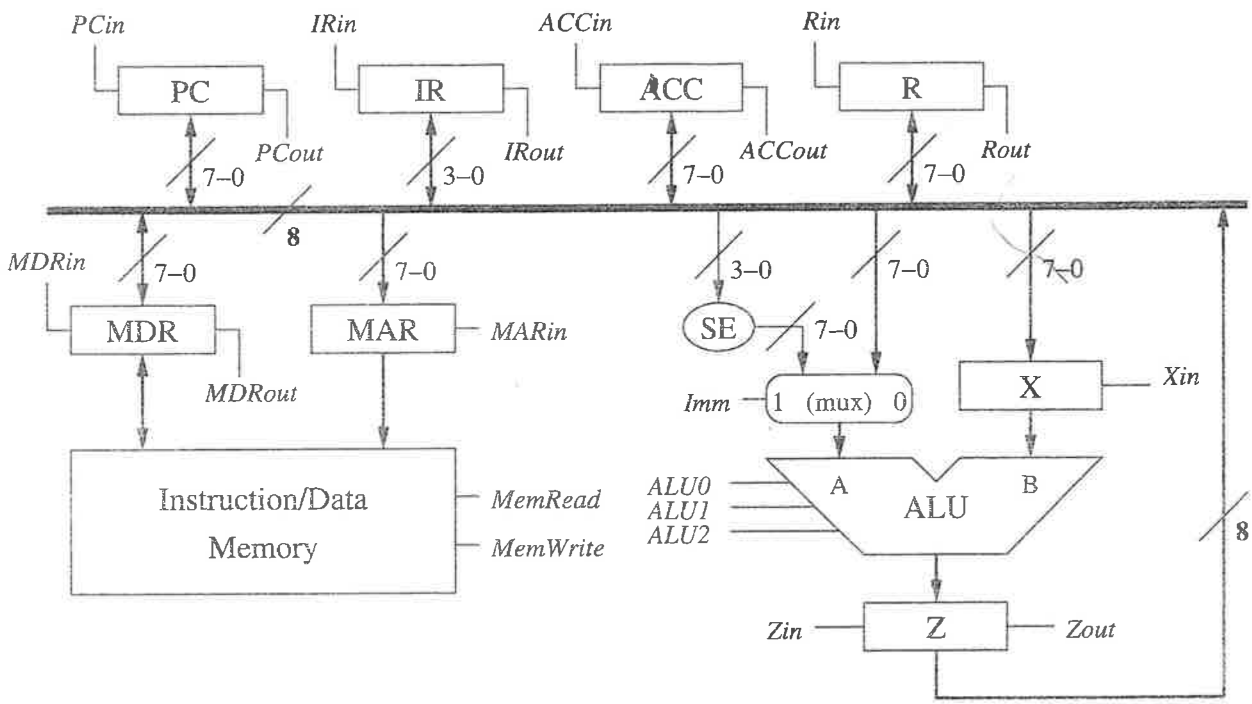
Here are a few of the instructions that have been defined:

Name	**Opcode**	**Operation**
ADD	0001	ACC=ACC+R
ADDM	0010	ACC=ACC+MEM[R]
ADDI	0011	ACC=ACC+Imm
COPYA2R	0100	R=ACC
COPYR2A	0101	ACC=R
LOADACC	0110	ACC=MEM[R]
JMP	1111	PC=PC+Imm

On the following page is a diagram of the machine. The control signals are in italics. The sign extend logic creates a 5-bit value which matches the contents of bit 3, so that the 8-bit value generated looks like 33333210 (instead of 76543210).

Here are the 20 control signals.

PCin	PCout	IRin	IRout	Rin	Rout	MARin
ACCin	ACCout	MDRin	MDRout	Zin	Zout	Xin
ALU0	ALU1	ALU2	MemRead	MemWrite	Imm	



17] [9] Fill in the steps necessary to perform an instruction fetch (incrementing the PC is part of the fetch process). Assume memory can respond to a READ request on the 2nd cycle after the MAR is written to (if MAR is loaded on cycle X, the MDR will be valid on cycle X+2).

Step	PCout	IRout	ACCout	Rout	Zout	MDRout	PCin	IRin	ACCin	Rin	Xin	Zin	MARin	MDRin	ALU2	ALU1	ALU0	Mread	Mwrite	Imm
0	1											1	1		1	1		1		
1		1			1													1		
2						1		1										1?		
3																				
4																				

Using Inc A unused

18] [8] Now that you have done the instruction fetch, fill in the microcode steps necessary to perform the following instruction: **ADDI 5**

Step	PCout	IRout	ACCout	Rout	Zout	MDRout	PCin	IRin	ACCin	Rin	Xin	Zin	MARin	MDRin	ALU2	ALU1	ALU0	Mread	Mwrite	Imm
0			1								1									
1		1										1					1			1
2					1				1											
3																				
4																				

Assume a 10-bit processor, with a 96-byte 3-way set associative cache and a linesize of 8. The contents of the cache are presented below (as always, there may be more information than you need):

Set 0

Tag	D(0)	D(1)	D(2)	D(3)	D(4)	D(5)	D(6)	D(7)
10101	3B	C1	43	86	3B	C1	43	86
11001	9D	90	B3	65	F4	EB	7A	BC
00110	9F	A8	28	54	FD	64	3F	13

Set 1 ✓

Tag	D(0)	D(1)	D(2)	D(3)	D(4)	D(5)	D(6)	D(7)
11110	72	9A	49	6F	84	CC	62	8D
11011	C4	25	C8	2E	75	E0	5A	F5
✓ 00111	69	83	1A	94	E4	FA	95	E8

Set 2 ✓

Tag	D(0)	D(1)	D(2)	D(3)	D(4)	D(5)	D(6)	D(7)
11110	E9	BA	C6	03	B8	AB	14	2B
✓ 11011	17	09	5A	8B	8F	0D	25	CD
00111	62	B2	04	B4	37	15	1C	3C

Set 3

Tag	D(0)	D(1)	D(2)	D(3)	D(4)	D(5)	D(6)	D(7)
11110	F8	E5	B4	AB	F4	50	6B	07
11011	EF	8C	99	9E	71	46	BF	0F
00111	10	E7	99	2E	A5	A2	A3	6B

Set 7

Tag	D(0)	D(1)	D(2)	D(3)	D(4)	D(5)	D(6)	D(7)
00010	03	47	05	45	E8	39	39	9D
10101	51	FE	CF	B5	5D	2A	DE	D8
11011	2A	F6	80	E0	E2	21	B7	B1

Set 6

Tag	D(0)	D(1)	D(2)	D(3)	D(4)	D(5)	D(6)	D(7)
11111	5A	70	5C	18	15	52	ED	1C
01011	CF	7E	2E	F9	25	8C	9C	38
00100	3C	D8	6F	CD	25	DB	32	1C

Set 5

Tag	D(0)	D(1)	D(2)	D(3)	D(4)	D(5)	D(6)	D(7)
11110	E0	C4	2C	B4	5D	AE	66	6E
11011	FF	D0	81	2E	4E	94	E5	20
00111	FB	5A	83	4B	E1	E9	43	D5

Set 4

Tag	D(0)	D(1)	D(2)	D(3)	D(4)	D(5)	D(6)	D(7)
11110	F7	8B	4E	E6	E7	94	B9	2D
11011	F9	09	23	A1	7A	C5	65	35
00111	7B	2F	9B	C8	DA	72	76	8C

19] [3] If the processor issues the address

1 6

0 0 1 1 1 | 0 1 | 1 1 0

Yes

Show how you have partitioned the address. Is this a hit in the cache? ([✓ YES] NO) If YES, you **must circle** the entry and the data value that is being referenced.

95

20] [3] If the processor issues the address

2 3

1 1 0 1 1 | 1 0 | 0 1 1

Yes

Show how you have partitioned the address. Is this a hit in the cache? ([✓ YES] NO) If YES, you **must circle** the entry and the data value that is being referenced.

8B

$\frac{\text{Mem}}{\text{line size}}$ = cachelines →

TAG | line no | offset

21] [6] This question is about a 32 byte Direct Mapped cache using a line size of 4 bytes. In the table below, fill in the cache's Tag values after each memory reference has been processed. If it is a miss, you should enter what the new tag should be in the appropriate slot. (X indicates the entry is invalid). If it is a hit, you should place an H in the correct location. There may be more Tag Array entries than you need.

	Contents of Tag Array after processing address (Time ->)			
Entry Number	Time 0 Initial Contents	Time 1 Address 0x547 (0 1 0 1 0|1 0 0|0 1 1 1)	Time 2 Address 0x456 (0 1 0 0 0|1 0 1|0 1 1 0)	Time 3 Address 0x546 (0 1 0 1 0|1 0 0|0 1 1 0)
0	X			
1	X			
2	X			
3	X			
100 4	X	01010		H
5	X		01000	
6	X			
7	X			

22] [2] What set of memory addresses are sent to memory on the first miss? (Your answer can be either in binary or in hex)

0x547

23] [2] On Figure 1 below show how an address flows from the CPU to Memory when the cache is physically addressed, and on figure 2 show how an address flows for a virtually addressed cache.

Figure 1

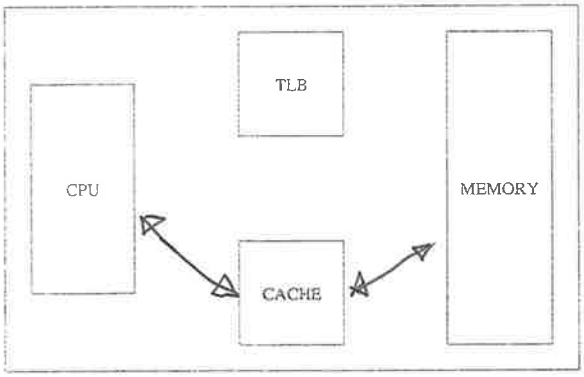
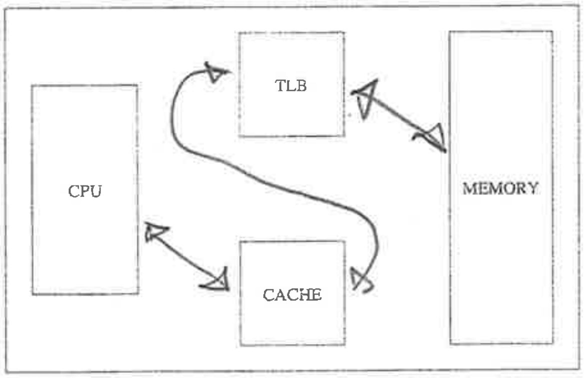


Figure 2



24] [4] Give 1 advantage for each approach (total of 20 words maximum).

Physical: Simpler

Virtual: More secure,

In this problem we have a machine that generates 12 bit Virtual Addresses, uses 256 byte pages, and has 2048 bytes of memory. The TLB has 5 entries, and is fully associative.

256 = 8

The first address the CPU issues is

1 1 0 1 | 0 0 0 1 1 0 0 1 (0xD19)

8

The requested page is not currently resident in the memory, so a page fault is generated and the Operating System is called in. After consulting its internal data structures, it decides to put the requested page in frame number 4.

Here are the contents of the page table and the TLB before the address is sent to memory:

Page Table		
Entry	Contents	Valid
0000		N
0001		N
0010		N
0011		N
0100		N
0101		N
0110		N
0111		N
1000		N
1001		N
1010		N
1011		N
1100		N
1101	100	~~N~~ Y
1110		N
1111		N

TLB		
Tag	Contents	Valid
	100	~~N~~ Y
		N
		N
		N
		N

25] [2] What is the physical address (in binary or hex) that gets sent to memory?

100 | 00011001

26] [2] Update the page table to show what it contains after the physical address is generated.

27] [3] Update the TLB to show what it contains after the physical address is generated.