University of Algiers 1

Faculty of sciences
Computer science department

Resit exam L2-ADO

25-june-2025

Duration : 90 minutes

Surname: First name: Group: Student ID No:	/ 20

Exercise 1: (3.5 points)

Let a cache memory of M words organized in *k* lines. If the cache block size is doubled but the total number of data words in the cache is unchanged, how will the following cache parameter change? Circle the correct answer and provide justifications

1/ Number of offset bits:

a) Unchanged b) +1 c) -1 d) 2x e) 0.5x

Assuming **Address size**: 32 bits, **Cache size**: 1024 words, **Block size**: 4 words/block **(0.5pt. missing details = -0.5)**
so far, no. of lines 1024/4 = 256 >> offset = log2(4) = 2 bits, index bits = log2(256)=8 bits, tag = 32-8-2 = 22 bits.
If we double block size to be 8 words/block >> offset bits= log2(8)=3 bits >> **answer: +1**

2/ Number of cache lines

a) Unchanged b) +1 c) -1 d) 2x e) 0.5x

Based on the previous example: no. of lines = cache size/block size=1024/8 = 128 line **(0.5pt. missing details = -0.5)**
answer : 0.5x

3/ Number of tag bits

a) Unchanged b) +1 c) -1 d) 2x e) 0.5x

offset bits from question1 = 3, index bits = log2(128) = 7 bits **(0.5pt. missing details = -0.5)**
tag bits = 32 – 3- 7=22 bits >> **answer: unchanged**

4/ A particular program experiences an average memory access time AMAT of 1.3 cycles. The access time on a cache hit is 1 cycle, the miss penalty (additional access time) is 10 cycles. What is the hit ratio?

(1pt. missing details = no mark)

1.3 = 1+ (1- hit ratio)* 10
0.3 = (1- hit ratio)* 10 >>>> Hit ratio = 0.97

5/ Consider a machine with byte addressable memory of 2^{32} bytes divided into blocks of size 32 bytes. Assume a 2-set associative cache having 512 cache lines is used with this machine. The size of tag field in bits is _____

total address bits = 32 bits, offset bits: log2(32) = 5 bits. No. of cache lines are 512 **(1pt. missing details = no mark)**
divided into sets, each sets has 2 lines>> no. of sets = 512/2=256 sets >> dedicated bits=log2(256)=8 bits
tag field bits = 32 – 8 – 5 = 19 bits

Exercise 2: (6.5 points)

Consider a cache with the following specifications: 4-way set associative, holds 64 Kbytes of data and Data words are 32 bits each. Data words are not byte addressed, they are word addressed and the memory address is 40 bits. There are 16 words per cache block and A First in First out (FIFO) replacement policy is used for each set.

1/ Show how the main memory address is decomposed (i.e. give the widths of the T/I/O fields).

There are 16 (2^4) addressable entries / block, thus, we need 4 bits of offset **(1.5pt. missing details = no mark)**
For the index bits we have 2^{16} bytes x (1 word / 2^2 bytes) x (1 block / 2^4 words) x (1 set / 2^2 blocks) = 2^8 sets
Therefore, need 8 bits of index.
Or we have 64 Kbytes of data in the cache divided into block each of (16 words/block x 4 bytes/words) = 1k of lines further divided into sets (4 set associative) = 256 sets (8 bits for index)
The remaining (40 – 4 – 8 = 28) 28 bits form the tag

All cache entries are initially empty at startup, the following addresses (in hexadecimal) are supplied to this cache in the order shown below.

2/ Indicate (in hexadecimal) for each of the following memory access whether it is a cache hit or a cache miss **(4.5 points**)

Address	Tag bits	maps to which set	Cache status
1: 0x F F F B D 0 9 8 7 3	F F F B D 0 9	87	Compulsory miss
2: 0x A B C D E F 1 1 8 3	A B C D E F 1	18	Compulsory miss
3: 0x A B C D E F 2 1 8 3	A B C D E F 2	18	Compulsory miss
4: 0x A B C D E F 1 1 8 4	A B C D E F 1	18	Hit
5: 0x A B C D E F 1 1 8 4	A B C D E F 1	18	Hit
6: 0x F F F B D 0 9 8 7 4	F F F B D 0 9	87	Hit
7: 0x F F F B D 0 A 9 7 4	F F F B D 0 A	97	Compulsory miss
8: 0x F F F B D 0 A 8 7 8	F F F B D 0 A	87	Compulsory miss
9: 0x A B C D E F 2 1 8 3	A B C D E F 2	18	Hit

3/ What is the overall miss rate for this pattern of accesses? **(0.5 points)**

5/9

Exercise 3: (4 points)

1/ Write in hexadecimals the machine codes associated with the following MIPS instructions

```
slti $s0, $t7, -8
```

I type. Op = 0xA, rt = 16 ($s0), rs = 15 ($t7), imm = -8 (0xFFF8) => 0x 29F0 FFF8 **(1 pt. missing details = no mark)**
0010 10 01 111 1 0000 1111 1111 1111 1000
0.25pt … 0.5 pt … 0.25 pt (only if details are correct)

2/ Write the MIPS assembly language instruction represented by the following hexadecimal machine code

```
0x01600009
```

0000 00 (op) 01 011 (rs) 0 0000 (rt) 0000 0 (rd)000 00 (sh) 00 1001 (func) **(1 pt. missing details = no mark)**
R type. Op = 0, rs = 11 ($t3), rt = 0, rd = 0, sh = 0, func = 0x9 => `jalr $t3`
0.25pt … 0.5 pt … 0.25 pt (only if details are correct)

2/ Write the MIPS assembly language instruction represented by the following hexadecimal machine code

```
0x27a50004
```

0010 01 (op) 11 101 (rs) 0 0101 (rt) 0000 0000 0000 0100 **(1 pt. missing details = no mark)**
I type. Op = 9, rt = 9 ($a1), rs = 29 ($sp), imm = 0x4 => addiu $a1, $sp, 4
0.25pt … 0.5 pt … 0.25 pt (only if details are correct)

3/ Assume a **J-type MIPS instruction** that allows jumping to a target instruction located at address 0x8A00 0540. What is the value of the 26-bit address field in the original instruction? Express your answer in binary

Target address = pc_{31..28}$, Destination, 0b00 **(1 pt. missing details = no mark)**
= ~~1000~~ **1010 0000 0000 0000 0101 0100 00~~00~~**
Destination field: **1010 0000 0000 0000 0101 0100 00**

Exercise 4: (6 points)

Consider the following MIPS code:

```
1:    lw    $t1, 0 ($t0)
2:    lw    $t2, 4 ($t0)
3:    add   $t3, $t1, $t2
4 :   sw    $t3, 12 ($t0)
5 :   lw    $t4, 8 ($t0)
6 :   add   $t5, $t1, $t4
7 :   sw    $t5, 16 ($t0)
```

1/ List all possible data hazards in the code (i.e. affected instructions and registers)

(2 pt. missing details = no mark)

between inst. 1 and 3 (`1.rt = 3.rs = $t1`)
between inst. 2 and 3 (`2.rt = 3.rt = $t2`)
between inst. 3 and 4 (`3.rd = 4.rt = $t3`)
between inst. 5 and 6 (`5.rt = 6.rt = $t4`)
between inst. 6 and 7 (`6.rd = 7.rt = $t5`)

2/ Fill in the table below assuming a "five-stage" pipelined data path with a “data forwarding unit”, and “double-pumping”. Indicate, if applicable, “data forwarding” occurrences with arrows, “pipeline stalls” with a “nop” or “repeated stages”, and double-pumping usage with circle **(3 points)**

Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1 lw $t1, 0 ($t0)	IF	ID	EX	Mem	WB													
2 lw $t2, 4 ($t0)		IF	ID	EX	Mem	WB												
3 add $t3, $t1, $t2			IF	ID	ID	EX	~~MEM~~	WB										
4 sw $t3, 12 ($t0)				IF	IF	ID	EX	MEM	~~WB~~									
5 lw $t4, 8 ($t0)						IF	ID	EX	MEM	WB								
6 add $t5, $t1, $t4							IF	ID	nop	EX	~~MEM~~	WB						
7 sw $t5, 16 ($t0)								IF	IF	ID	EX	MEM	~~WB~~					

3/ Apply code rescheduling to reduce as much as possible the number of stalls

(1 pt. missing details = no mark)

Both stalls on: `$t1` between instructions 1 and 3 and on $t4 between inst. 5 and 6 can be eliminated if inst. 5 is scheduled before inst. 3 in the pipeline. This gives: 1 , 2 , **5** , 3 , 4 , 6 , 7.