CSE-3666 – Introduction to Computer Architecture

Spring 2023

Instructor	Office	Contact Details	Office Hours
Hasan Baig	305C	Email: hasan.baig@uconn.edu	MoWed: 11am – 12pm[1, 2] Thurs: 4.00pm – 4.45pm[1]
Sumaira Zaman (TA) Rauni Machado (TA)	Virtual	Email: sumaira.zaman@uconn.edu Email: rauni.machado-filho@uconn.edu	See Discord for details.

[1] *By appointment only via https://calendly.com/hb_officehours/15min*
[2] *Expect Discord's post response within 24 hours and emails within 48 hours.*

Assignment - 2

Release Date: April 01, 2023	**Due by:** **April 21, 2023**
Total points: 100	**Points obtained:**
	Scaled marks (out of 7.5):

Students' Name(s):	**Students' ID(s):**

Purpose:
The purpose of this assignment is to help you strengthen your understandings about the topics related to RISC-V instructions encoding, processor designing, pipelining and hazards, and performance of a computer system.

Instructions:
1. This assignment should be submitted in a group of up to two students.
2. Questions should be answered in the space provided and the scanned copies be uploaded on GradeScope under **Assignment 2.**

Grading Criteria:
1. Your assignments will be checked by instructor/TA followed by a Viva (if needed).
2. During the viva, you will be judged whether you understood the question yourself or not. If you are unable to answer correctly to the question you have attempted right, you may lose your points. No excuses/justifications will be entertained.
3. Zero will be given if the assignment is found to be plagiarized.
4. Untidy work will result in reduction of your points.

Late submission penalty:
- 1-day late submission – 20% deduction; 2-days late submission – 40% deduction.
- No submission will be accepted after two days of the original deadline.

CLO Assessment:
This assignment assesses students for some portion of the following course learning outcomes.

Course Learning Outcomes		CLO Assessed
CLO 1	Represent data (e.g., integer/floating-point numbers, instructions.) with bits.	
CLO 2	Program with RISC-V instructions (computing with instructions).	
CLO 3	Design an instruction set architecture (ISA). For example, describe types of instructions, encode/decode instructions, and identify and apply good design principles.	✓
CLO 4	Design digital circuit (e.g., arithmetic-logic unit) with logic gates.	
CLO 5	Design a processor that supports a specified ISA.	✓
CLO 6	Design a processor pipeline and handle data and control hazards.	✓
CLO 7	Design memory cache in a computer system.	
CLO 8	Evaluate the performance of a computer system.	✓

Questions

1. *(10 points)* Consider two processors P1 and P2 that have the same ISA but different implementations. The ISA has four classes of instructions: class A, B, C, and D. The clock rate of two processors and the number of clock cycles required for each class on the processors are listed in the following table.

Processor	Clock Rate	Class A	Class B	Class C	Class D
P1	2 GHz	1	2	3	3
P2	3 GHz	1	1	4	5

Suppose the breakdown of instructions executed in a program is as follows:

10% class A, 20% class B, 50% class C, and 20% class D.

Round answers to the nearest hundredth if necessary. For example, 1/2 to 0.5, 2/3 to 0.67.

a. What is the overall CPI of the program on P1?
b. What is the overall CPI of the program on P2?
c. How many times faster is the program on P2 than on P1? Note that the clock rate is different on P1 and P2.
d. Suppose a compiler can optimize the program, replacing all class D instructions with class A instructions. Each class D instruction requires two class A instructions. What is the average CPI of the program on P2 after the optimization?
e. What is the speedup the compiler in d) can achieve on processor P2?

Solution

a.

0.1 * 1 + 0.2 * 2 + 0.5 * 3 + 0.2 * 3 = 2.6

b.

0.1 * 1 + 0.2 * 1 + 0.5 * 4 + 0.2 * 5 = 3.3

c.

$$\frac{\text{CPUTime}_1}{\text{CPUTime}_2} = \frac{\text{IC} \times \text{CPI}_1 \times \text{CycleTime}_1}{\text{IC} \times \text{CPI}_2 \times \text{CycleTime}_2} = \frac{2.6 \times \frac{1}{2}}{3.3 \times \frac{1}{3}} = 1.18$$

Note that P2 is faster although its CPI is higher.

d.

If the instruction count (IC) is 1 before the optimization, it becomes 1 + 0.2 = 1.2 after the optimization. IC for each class after the optimization is listed below.

	Class A	**Class B**	**Class C**
Cycle	1	1	4
IC for each class	0.1 + 0.2 * 2 = 0.5	0.2	0.5

The new average CPI is = (0.5 * 1 + 0.2 * 1 + 0.5 * 4) / 1.2 = 2.25

We could calculate the new percentages of each class:

Class A: 0.5 / 1.2, Class B: 0.2/1.2, and Class C: 0.5/1.2.

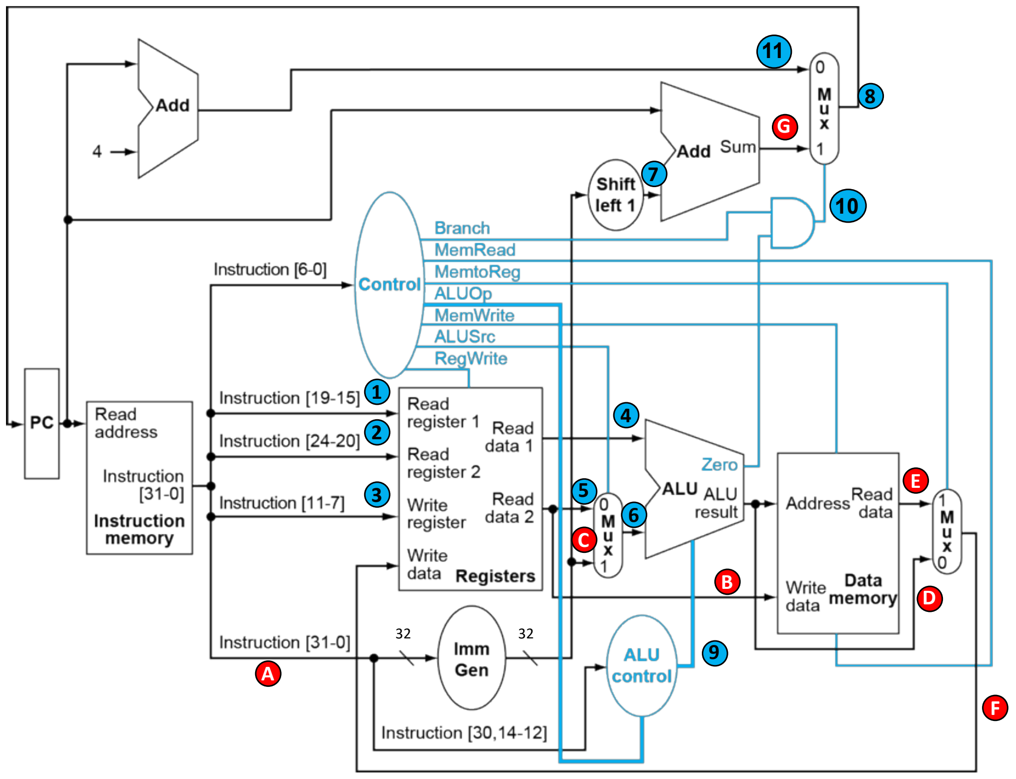
e.

The speedup achieved by the compiler is (3.3 * 1) / (2.25 * 1.2) = 1.22.

Write down every factor in CPU time. The clock cycle time is canceled out.

Questions

2. *(20 points)* The Single-cycle RISC-V processor, its register contents and the data memory contents are shown in the figures below.



Registers	
23	x31
345	x30
765	x29
21	x28
14543	x27
123	x26
145	x25
43	x24
23	x23
213	x22
435	x21
193	x20
232	x19
12	x18
341	x17
53	x16
54	x15
3423	x14
23	x13
32	x12
65	x11
1545	x10
2	x9
Frame pointer	x8
43	x7
56	x6
98	x5
Thread pointer	x4
Global pointer	x3
Stack pointer	x2
Return address	x1
Constant value 0	x0

Registers

Data Memory	
12	
42	
234	
345	
342	824
⋮	
5656	
12324	
1232	
3434	
123	304
⋮	
545	80
12	76
232	72
644	68
34	64
92	60
⋮	
576	16
123	12
123	8
532	4
12353	0

Data Memory

For the following instruction:

```
lw x26, 52(x18)
```

a. Assume that the current value of PC = 8, write down the values (in decimal) of the following points labelled in the figure above.

Probe points	Values	Score Points
1.	18	1
2.	20	1
3.	26	1
4.	12	1
5.	193	1
6.	52	1
7.	104	2
8.	12 (same as the value of 11)	0.5
9.	0010	1
10.	0	1
11.	8 + 4 = 12	1
B.	193 (same as the value of 5)	0.5
D.	64	1
E.	34	1
F.	34	1
G.	104 + 8 = 112	1

b. Indicate the values of control signals in the table below.

Probe points	Values	Score Points
Branch	0	0.5
MemRead	1	0.5
MemtoReg	1	0.5
ALUOp[1:0]	00	1
MemWrite	0	0.5
ALUSrc	1	0.5
RegWrite	1	0.5

Questions

3. *(10 points)* Consider the following instruction mix:

R-type	I-type (non-lw)	Load	Store	Branch
24%	28%	25%	10%	11%

a. What fraction of all instructions use data memory? (2.5)

25+10=35%. Only Load and Store use Data memory.

b. What fraction of all instructions use instruction memory? (2.5)

100% Every instruction must be fetched from instruction memory before it can be executed.

c. What fraction of all instructions use the sign extend? (2.5)

28 + 25 + 10 + 11 = 74%. Only R-type instructions do not use the Sign extender.

d. What is the sign extend doing during cycles in which its output is not needed? (2.5)

The sign extend produces an output during every cycle. If its output is not needed, it is simply ignored or not used.

Questions

4. *(10 points)* Assume that individual stages of the datapath have the following latencies:

IF	ID	EX	MEM	WB
250ps	350ps	150ps	300 ps	200ps

Also, assume that instructions executed by the processor are broken down as follows:

ALU/Logic	Jump/Branch	Load	Store
45%	20%	20%	15%

a. What is the clock cycle time in a pipelined and non-pipelined processor? (2)

Pipelined: 350; non-pipelined: 1250

b. What is the total latency of an lw instruction in a pipelined and non-pipelined processor? (2)

Pipelined: 1750; non-pipelined: 1250

c. If we can split one stage of the pipelined datapath into two new stages, each with half the latency of the original stage, which stage would you split and what is the new clock cycle time of the processor? (2)

Split the ID stage. This reduces the clock-cycle time to 300ps.

d. Assuming there are no stalls or hazards, what is the utilization of the data memory? (2)

35%

e. Assuming there are no stalls or hazards, what is the utilization of the write-register port of the "Registers" unit? (2)

65%

Questions

5. *(10 points)* Assume that x11 is initialized to 11 and x12 is initialized to 22. Suppose you executed the code below on a version of the pipeline from Section 4.6 that does not handle data hazards (i.e., the programmer is responsible for addressing data hazards by inserting NOP instructions where necessary). What would the final values of register x15 be? Assume the register file is written at the beginning of the cycle and read at the end of a cycle. Therefore, an ID stage will return the results of a WB state occurring during the same cycle. In the book, see Section 4.8 and Figure 4.68 for details.

```
addi x11, x12, 5

add x13, x11, x12

addi x14, x11, 15

add x15, x11, x11
```

(7pts for working and 3 pts for answer)

Solution

x15 = 54 (The code will run correctly because the result of the first instruction is written back to the register file at the beginning of the 5th cycle, whereas the final instruction reads the updated value of x1 during the second half of this cycle.)

Questions

6. *(10 points)* Consider the fragment of RISC-V assembly below:

```
sw x29, 12(x16)

lw x29, 8(x16)

sub x17, x15, x14

beq x17, x0 label

add x15, x11, x14

sub x15, x30, x14
```

Suppose we modify the pipeline so that it has only one memory (that handles both instructions and data). In this case, there will be a structural hazard every time a program needs to fetch an instruction during the same cycle in which another instruction accesses data.

a. Draw a pipeline diagram to show where the code above will stall (Use ** to indicate stall). (3)

```
sw x29, 12(x16)      IF ID EX ME WB
lw x29, 8(x16)          IF ID EX ME WB
sub x17, x15, x14          IF ID EX ME WB
                              **
                                 **
beq x17, x0 label                   IF ID EX ME WB
add x15, x11, x14                      IF ID EX ME WB
sub x15, x30, x14                         IF ID EX ME WB
```

b. In general, is it possible to reduce the number of stalls/NOPs resulting from this structural hazard by reordering code? (2)

Reordering code won't help. Every instruction must be fetched; thus, every data access causes a stall. Reordering code will just change the pair of instructions that are in conflict.

c. We have seen that data hazards can be eliminated by adding NOPs to the code. Can you do the same with this structural hazard? If so, explain how. If not, explain why not. (3)

   You can't solve this structural hazard with NOPs, because even the NOPs must be fetched from instruction memory.

d. Approximately how many stalls would you expect this structural hazard to generate in a typical program? (Use the instruction mix from question 4 above) (2)

   35%. Every data access will cause a stall.

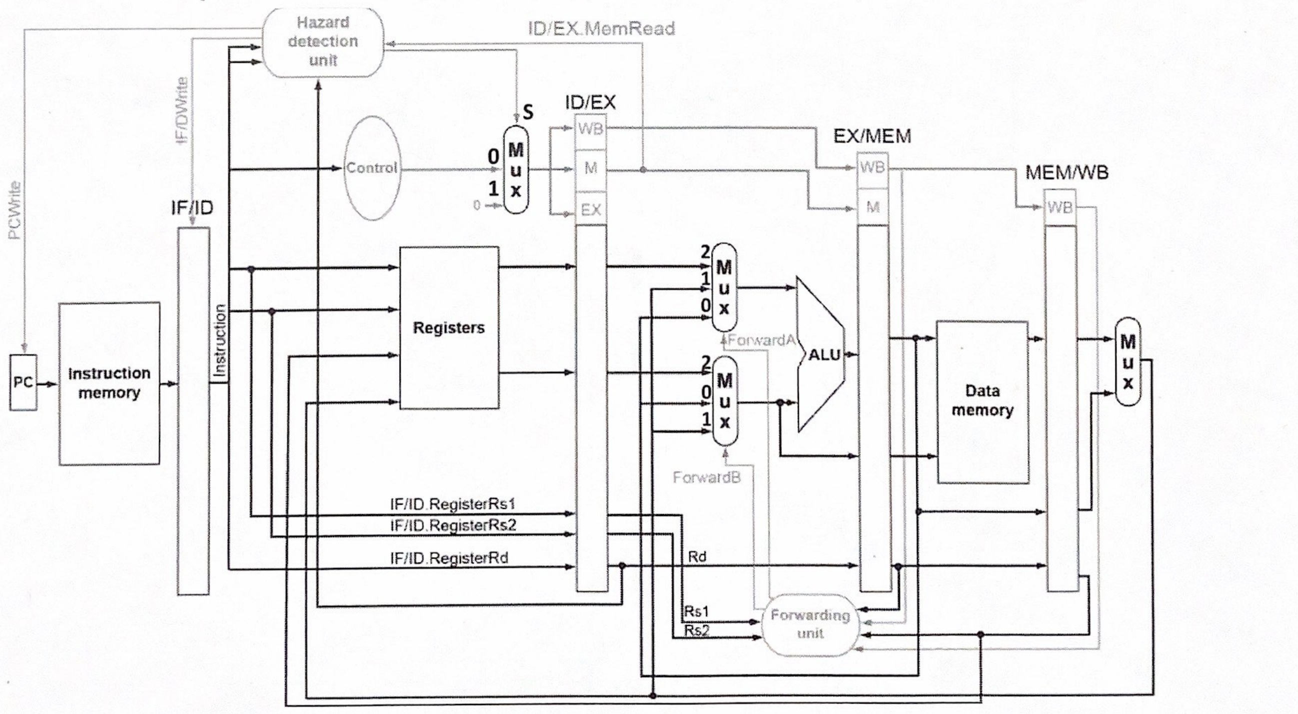
Questions
7. *(20 points)* For the following set of instructions, fill up the sheet of single-cycle diagram for pipeline execution shown on the next page.

```
sw x1, 8 (x2)
lw x3, 8 (x1)
add x4, x3, x6
```

Note: This sheet tends to change which will be updated, if needed, later. So, try to attempt it only after the relevant discussion takes place in the class.

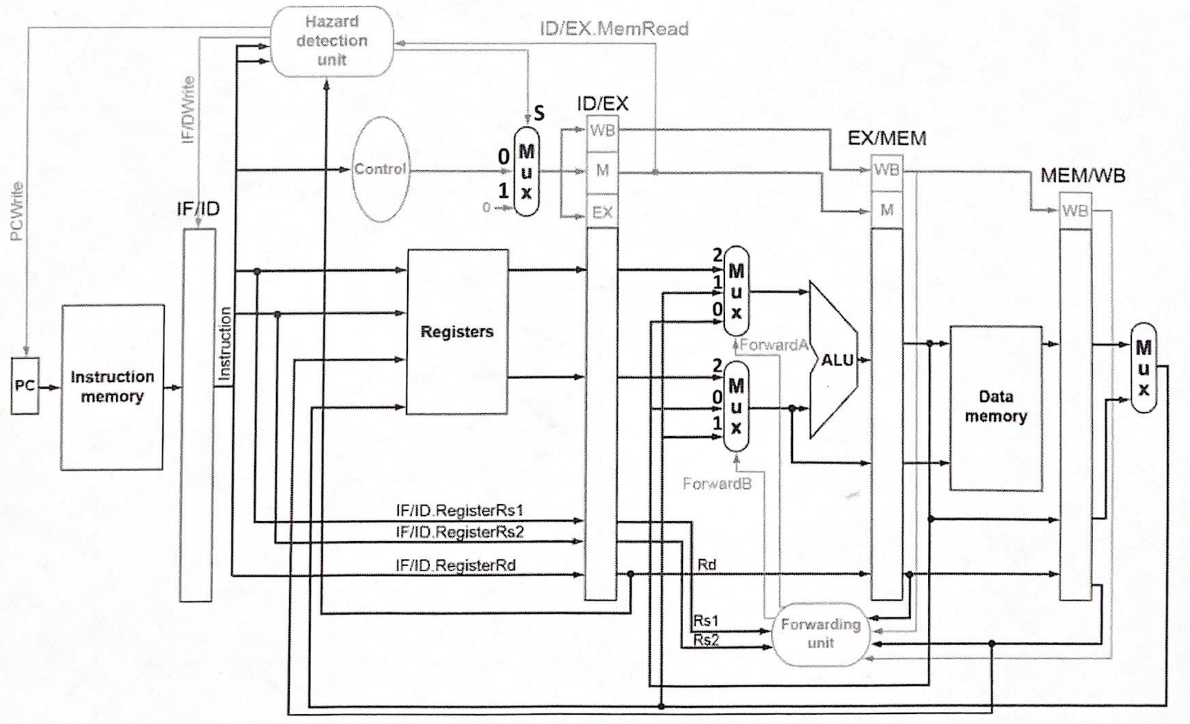
Pipeline with single-cycle diagram practice sheet

	Instruction Fetch	Instruction Decode	Execution	Memory	WriteBack
Cycle 1	In: sw x1,8(x2)	N/A			
Cycle 2	In: lw x3,8(x1)	In: sw x1, 8(x2) IF/ID. Rs1=x2, Rs2=x1, Rd=- S=0, ID/Ex.MemRead=~~0~~-	N/A		
Cycle 3	In: add x4,x3,x6	In: lw x3, 8(x1) IF/ID. Rs1=x1, Rs2=-, Rd=x3 S=0, ID/Ex.MemRead=0	In: sw x1,8(x2) ID/Ex. Rs1=x2 Rs2=x1 Rd= - ForwardA: 2 ForwardB: 2	N/A	
Cycle 4	In: N/A	In: add x4,x3,x6 IF/ID. Rs1=x3, Rs2=x6, Rd=x4 S=1 ID/Ex.MemRead=1	In: lw x3,8(x1) ID/Ex. Rs1=x1 Rs2= - Rd= x3 ForwardA: 2 ForwardB: 2	In: sw x1,8(x2) Ex/Mem.Rs1=x2 Rs2=x1, Rd= -	N/A
Cycle 5	In:	In: add x4,x3,x6 IF/ID. Rs1=x3, Rs2=x6, Rd=x4 S=0, ID/Ex.MemRead=0	In: NOP ID/Ex. Rs1=x3 Rs2=x6 Rd= x4 ForwardA: 0 ForwardB: ~~0~~2	In: lw x3,8(x1) Ex/Mem.Rs1=x1 Rs2= - , Rd= x3	In: N/A MEM/WB.Rd= -



Designed by Hasan Baig, UConn, USA

	Instruction Fetch	Instruction Decode	Execution	Memory	WriteBack
Cycle 6	In:	In: ______ IF/ID. Rs1=___, Rs2=___ , Rd=___ S=___, ID/Ex.MemRead=___	In: add x4,x3,x6 ID/Ex. Rs1= x3 Rs2= x6 Rd= x4 ForwardA: 1 ForwardB: 2	In: NOP Ex/Mem.Rs1= x3 Rs2= x6 , Rd= x4	In: lw x3,8(x1) MEM/WB.Rd= x3
Cycle 7	In:	In: ______ IF/ID. Rs1=___, Rs2=___ , Rd=___ S=___, ID/Ex.MemRead=___	In: ______ ID/Ex. Rs1= __Rs2= __ Rd= __ ForwardA:____ ForwardB:____	In: add x4,x3,x6 Ex/Mem.Rs1= x3 Rs2= x6 , Rd= x4	In: NOP MEM/WB.Rd= x4
Cycle 8	In:	In: ______ IF/ID. Rs1=___, Rs2=___ , Rd=___ S=___, ID/Ex.MemRead=___	In: ______ ID/Ex. Rs1= __Rs2= __ Rd= __ ForwardA:____ ForwardB:____	In: ______ Ex/Mem.Rs1=___ Rs2=___ , Rd=___	In: add x4,x3,x6 MEM/WB.Rd= x4
Cycle 9	In:	In: ______ IF/ID. Rs1=___, Rs2=___ , Rd=___ S=___, ID/Ex.MemRead=___	In: ______ ID/Ex. Rs1= __Rs2= __ Rd= __ ForwardA:____ ForwardB:____	In: ______ Ex/Mem.Rs1=___ Rs2=___ , Rd=___	In: ______ MEM/WB.Rd=__
Cycle 10	In:	In: ______ IF/ID. Rs1=___, Rs2=___ , Rd=___ S=___, ID/Ex.MemRead=___	In: ______ ID/Ex. Rs1= __Rs2= __ Rd= __ ForwardA:____ ForwardB:____	In: ______ Ex/Mem.Rs1=___ Rs2=___ , Rd=___	In: ______ MEM/WB.Rd=__



Designed by Hasan Baig, UConn, USA

Questions

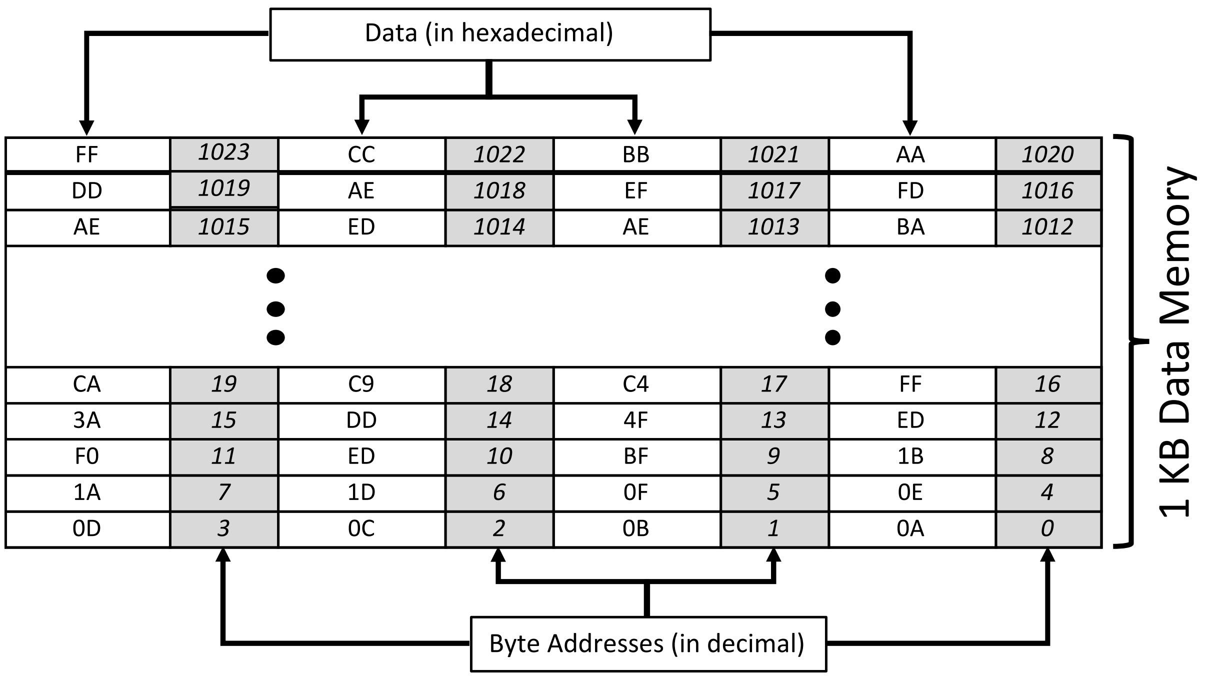
8. *(10 points)* Assume a direct mapped cache with 4 cache blocks, each of which contains four bytes. The cache is initially empty. The series of address references, which refers to bytes in a memory, issued by a processor are shown in Table I below. Assume that a 1 KB memory is used in the system as shown in Figure 1. For each memory reference labelled in Table I below, write if accessing these memory locations resulted in a cache-hit (H) or cache-miss (M). Also show the final contents of the cache in Table II. Cut the previous contents and write the new values if the data must be overwritten.

Table I

Requested Word at address	1	0	8	1	2	5	10	6	16	9	5	13	1019	17	1016
Status (H/M)	M	H	M	H	H	M	H	H	M	H	H	M	M	H	H

Table II

	Byte indexes			
Cache Index	11	10	01	00
3	3A	DD	4F	ED
2	~~F0~~ DD	~~ED~~ AE	~~BF~~ EF	~~1B~~ FD
1	1A	1D	0F	0E
0	~~0D~~ CA	~~0C~~ C9	~~0B~~ C4	~~0A~~ FF



FF	1023	CC	1022	BB	1021	AA	1020
DD	1019	AE	1018	EF	1017	FD	1016
AE	1015	ED	1014	AE	1013	BA	1012
⋮						⋮	
CA	19	C9	18	C4	17	FF	16
3A	15	DD	14	4F	13	ED	12
F0	11	ED	10	BF	9	1B	8
1A	7	1D	6	0F	5	0E	4
0D	3	0C	2	0B	1	0A	0

Figure 1. 1 KB Data Memory. The memory addresses are shown as decimal values and the memory contents/data are shown as hexadecimal values.