Habib University

EE-371/CS-330/CE-321 Computer Architecture – Spring 2023
Instructors: Dr. Tariq Kamal, Dr. Farhan Khan, Dr. Munzir Zafar, Dr. Sorath

Time = 40 minutes **Quiz 03** **Max Points: 20**

Instructions:

i. ***Smart watches, laptops, and similar electronics are strictly NOT allowed.***
ii. ***Answer sheets should contain all steps, working, explanations, and assumptions.***
iii. Attempt the quiz on clean papers with black/blue ink.
iv. Print your name and HU ID on all sheets.
v. Turn in your question paper along with your answer sheets.
vi. You are not allowed to ask/share your method or answer with your peers. The work submitted by you is solely your own work. Any violation of this will be the violation of HU Honor code and proper action will be taken as per university policy if found to be involved in such an activity.

CLO Assessment:

This assignment assesses students for the following course learning outcomes.

Course Learning Outcomes		CLO Assessed
CLO 1	***Explain*** the role of ISA in modern processors and instruction encodings and assembly language programming	
CLO 2	***Explain*** the architecture and working of a single cycle processor	
CLO 3	***Design*** the architecture to mitigate issues of a pipelined processor	✓
CLO 4	***Analyze the*** performance of cache operations	

Question 1 [4 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 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 registers x13, x14, and x15 be? Assume the register file is written at the beginning of the cycle and read at the end of a cycle.

```
addi    x11, x12, 5
add     x13, x11, x12
addi    x14, x11, 15
add     x15, x11, x11
```

x11 = 11
x12 = 22

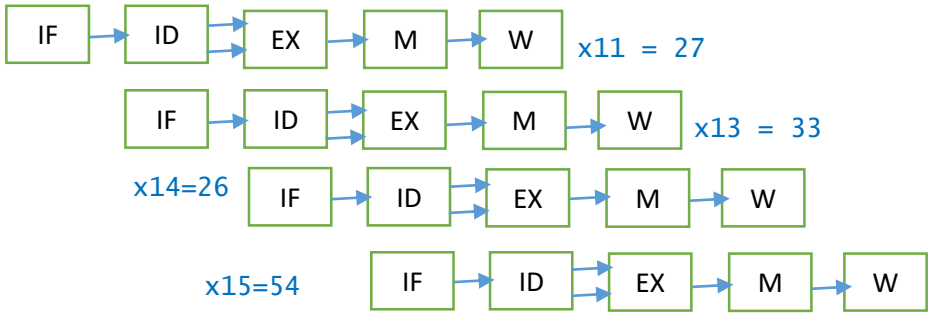
x

addi x11, x12, 5

add x13, x11, x12

addi x14, x11, 15

add x15, x11, x11



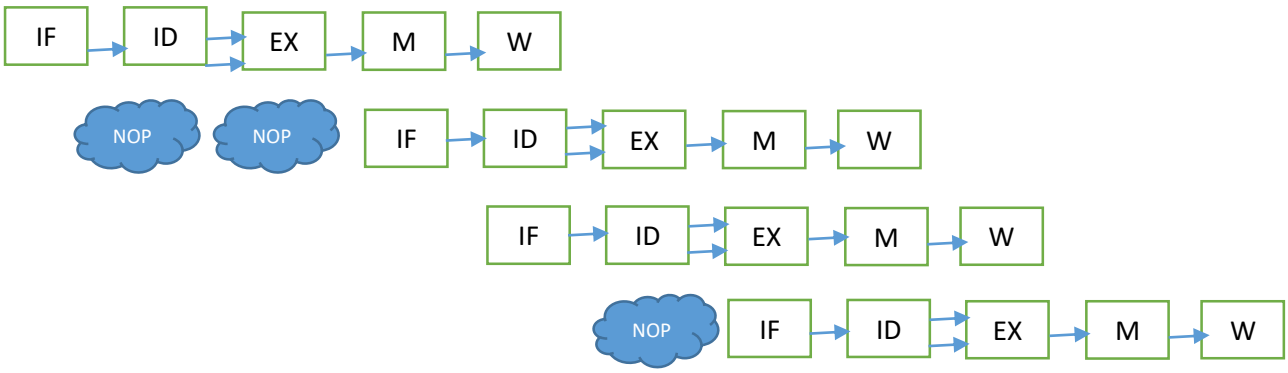
Final values are:
x13=33,
x14=26, and
x15=54

Question 2 [4 points]:

Add NOP instructions to the code below so that it will run correctly on a pipeline that does not handle data hazards.

```
addi    x11, x12, 5
add     x13, x11, x12
addi    x14, x11, 15
add     x15, x13, x12
```

```
addi    x11, x12, 5
NOP
NOP
add     x13, x11, x12
addi    x14, x11, 15
NOP
add     x15, x13, x12
```



Question 3 [8 points]:

Consider the fragment of RISC-V assembly below:

```
LOOP:   ld   x10, 0(x13)
        ld   x11, 8(x13)
        add  x12, x10, x11
        subi x13, x13, 16
        bnez x12, LOOP
```

Assume that perfect branch prediction is used (no stalls due to control hazards), that there are no delay slots, that the pipeline has full forwarding support, and that branches are resolved in the EX (as opposed to the ID) stage.

a) [4 points] Show a pipeline execution diagram for the first two iterations of this loop.

```
1 ld   x10, 0(x13)
2 ld   x11, 8(x13)
3 add  x12, x10, x11
4 subi x13, x13, 16
5 bnez x12, LOOP
```

I.No.	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16
1	IF	ID	EX	M	W											
2		IF	ID	EX	M	W										
-			S	S	S	S	S									
3				IF	ID	EX	M	W								
4					IF	ID	EX	M	W							
5						IF	ID	EX	M	W						
1							IF	ID	EX	M	W					
2								IF	ID	EX	M	W				
-									S	S	S	S	S			
3										IF	ID	EX	M	W		
4											IF	ID	EX	M	W	
5												IF	ID	EX	M	W

b) [4 points] Mark pipeline stages that do not perform useful work. How often while the pipeline is full do we have a cycle in which all five pipeline stages are doing useful work? (Begin with the cycle during which the subi is in the IF stage. End with the cycle during which the bnez is in the IF stage.)

I.No.	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16
1	IF	ID	EX	M	W											
2		IF	ID	EX	M	W										
-			S	S	S	S	S									
3				IF	ID	EX	M!	W								
4					IF	ID	EX	M!	W							
5						IF	ID	EX	M!	W!						
1							IF	ID	EX	M	W					
2								IF	ID	EX	M	W				
-									S	S	S	S	S			
3										IF	ID	EX	M!	W		
4											IF	ID	EX	M!	W	
5												IF	ID	EX	M!	W!

In a clock cycle, a pipeline stage is not performing useful work if it is being stalled or if an instruction at a particular stage is not doing any useful work.

As the table above shows, there are not any cycles during which every pipeline stage is doing useful work.

Question 4 [4 points]:

Which of the two pipeline diagrams below better describes the operation of the pipeline's hazard detection unit? Why?

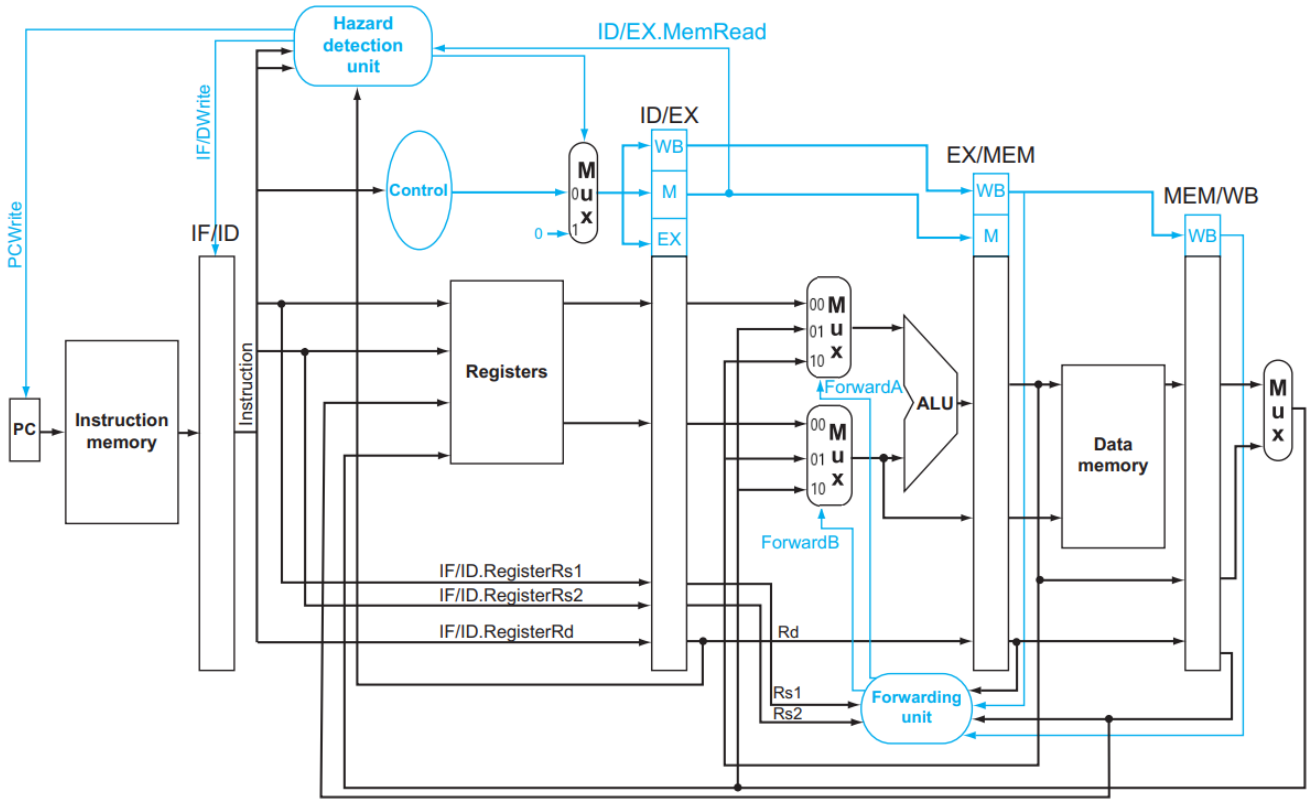
Choice 1:

```
ld x11, 0(x12):     IF ID EX ME WB
add x13, x11, x14:    IF ID EX..ME WB
or x15, x16, x17:      IF ID..EX ME WB
```

Choice 2:

```
ld x11, 0(x12):    IF ID EX ME WB
add x13, x11, x14:  IF ID..EX ME WB
or x15, x16, x17:    IF..ID EX ME WB
```

Choice 2. A careful examination of the code shows that the need for a stall is detected during the ID stage. It is this stage that prevents the fetch of a new instruction, effectively causing the add to repeat its ID stage.


Hazard detection unit
ID/EX.MemRead
IF/DWrite
PCWrite
Control
ID/EX
EX/MEM
MEM/WB
WB
M
EX
IF/ID
Registers
PC
Instruction memory
Instruction
ALU
Data memory
ForwardA
ForwardB
IF/ID.RegisterRs1
IF/ID.RegisterRs2
IF/ID.RegisterRd
Rd
Rs1
Rs2
Forwarding unit
Mux