Habib University

EE-371/CS-330/CE-321 Computer Architecture and Organization – 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 [20 points]:

Assume that the following sequence of instructions is executed on a five-stage pipelined RISC-V processor:

```
ld  x29, 20(x10)
and x18, x29, x16
sub x14, x29, x15
or  x29, x14, x29
and x10, x16, x17
```

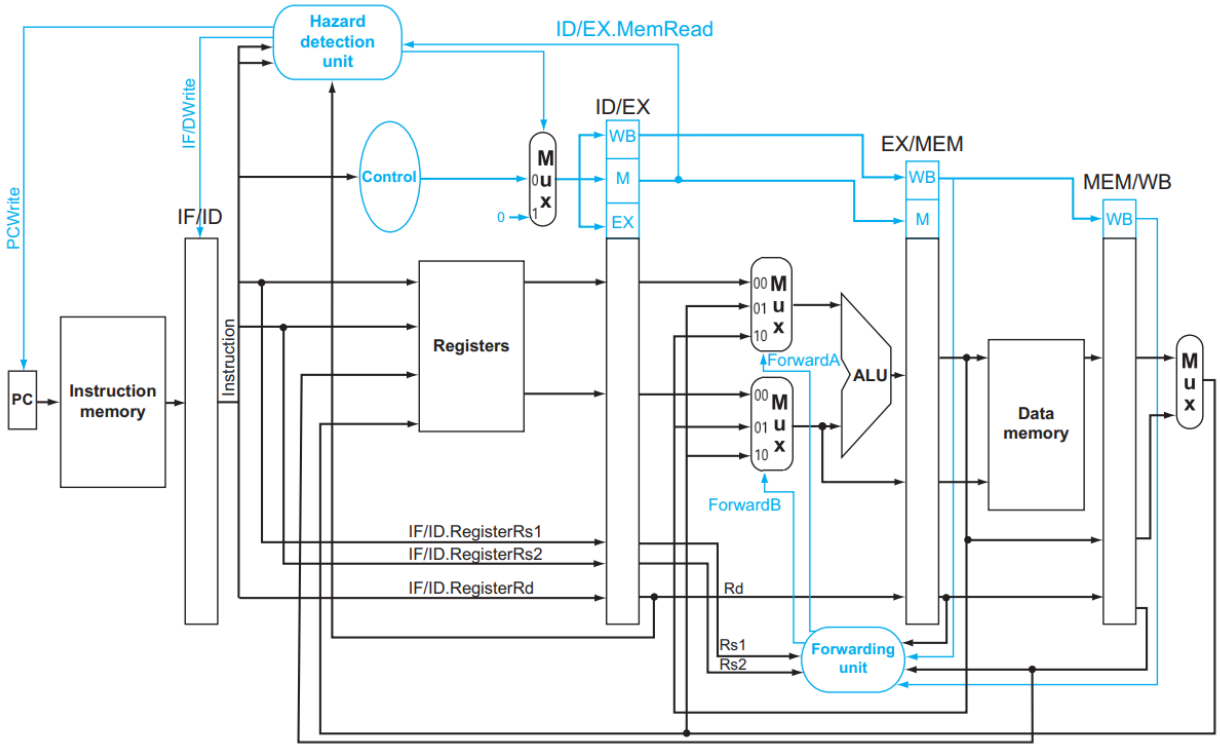
a) [4 marks] If there is no forwarding or hazard detection, insert NOPs to ensure correct execution.

b) [4 marks] Now, change and/or rearrange the original code to minimize the number of NOPs needed.

c) [4 marks] If a processor has the forwarding unit implemented, but not the hazard detection unit. What happens when the original code executes on such a processor?

d) [4 marks] The figure below shows the pipelined processor with forwarding and hazard detection units. For the first seven cycles during the execution of this code, specify which control signals are asserted in each cycle by hazard detection and forwarding units.

e) [4 marks] If there is no forwarding, what new input and output signals do we need for the hazard detection unit so that the original code executes correctly? Using this instruction sequence as an example, explain why each signal is needed.



a) If there is no forwarding or hazard detection, insert NOPs to ensure correct execution.

Solution:

Pipeline Diagram:
("**. .**" indicates a stall).

```
ld  x29, 20(x10)     IF  ID  EX  MEM WB
and x18, x29, x16        ..  ..  IF  ID  EX  MEM WB
sub x14, x29, x15                IF  ID  EX  MEM WB
or  x29, x14, x29                    ..  ..  IF  ID  EX  MEM WB
and x10, x16, x17                            IF  ID  EX  MEM WB
```

Modified Code:

```
ld  x29, 20(x10)
NOP
NOP
and x18, x29, x16
sub x14, x29, x15
NOP
NOP
or  x29, x14, x29
and x10, x16, x17
```

b) Now, change and/or rearrange the original code to minimize the number of NOPs needed.

Solution:

```
ld  x29, 20(x10)
and x10, x16, x17
NOP
sub x14, x29, x15
and x18, x29, x16
NOP
or  x29, x14, x29
```

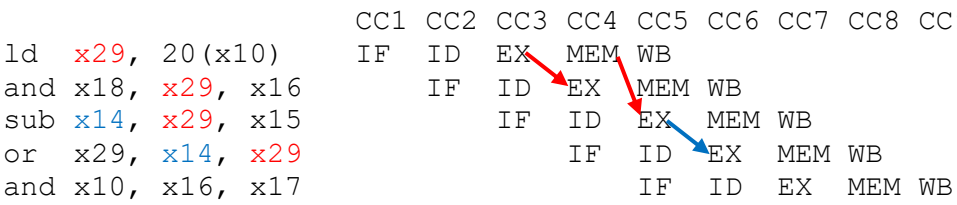
c) If a processor has the forwarding unit implemented, but not the hazard detection unit. What happens when the original code executes on such a processor?

Solution:

The intended logic of the code is as follows:

```
ld  x29, 20(x10)  // x29 = Mem[x10+20]
and x18, x29, x16 // x18 = x29 & x16 = Mem[x10+20] & x16
sub x14, x29, x15 // x14 = x29 – x15 = Mem[x10+20] - x15
or  x29, x14, x29 // x29 = x14 | x29 = (Mem[x10+20]-x15) | Mem[x10+20]
and x10, x16, x17 // x10 = x16 & x17
```

Here is multi-cycle diagram, which also shows the data forwarding that will be performed by the forwarding unit:



So, the code will mostly execute as per the intended logic, thanks to the forwarding unit.

An exception is the `and` instruction. We know that the value to be written to the destination register by the `ld` instruction is available only after the MEM stage, which is why a hazard detection unit is needed so that it inserts a stall when a load-use data hazard occurs. In this case, if a hazard detection unit was implemented, it would have ensured that that the EX stage of the `and` instruction is delayed till the `ld` instruction completes its MEM stage.

In the absence of a hazard detection unit, an error will occur during the EX stage of the `and` instruction (i.e., in CC4): As the 1st input argument of the ALU, the forwarding unit will mistakenly supply the output of the EX stage of the `ld` instruction i.e., the ALU output in the EX stage of the `ld` instruction i.e., `x10 + 20` i.e., the address. However, what the original code intended was the value at this address i.e., `Mem[x10+20]`.

The resulting logic that ends up being implemented is as follows:

```
ld  x29, 20(x10)  // x29 = Mem[x10+20]
and x18, x29, x16 // x18 = x29 & x16 = (x10+20) & x16
sub x14, x29, x15 // x14 = x29 – x15 = Mem[x10+20] - x15
or  x29, x14, x29 // x29 = x14 | x29 = (Mem[x10+20]-x15) | Mem[x10+20]
and x10, x16, x17 // x10 = x16 & x17
```

Note that an unintended value is stored in register `x18`. Instead of getting the intended value of (`Mem[x10+20] & x16`) , the value it ends up getting is (`(x10+20) & x16`).

d) The figure below shows the pipelined processor with forwarding and hazard detection units. For the first seven cycles during the execution of this code, specify which control signals are asserted in each cycle by hazard detection and forwarding units.

Solution:

	CC1	CC2	CC3	CC4	CC5	CC6	CC7	CC8	CC9	CC10
ld x29, 20(x10)	IF	ID	EX	MEM	WB					
and becomes nop		IF	ID	..	..	..				
and x18, x29, x16			IF	ID	EX	MEM	WB			
sub x14, x29, x15				IF	ID	EX	MEM	WB		
or x29, x14, x29					IF	ID	EX	MEM	WB	
and x10, x16, x17						IF	ID	EX	MEM	WB
ForwardA	X	X	00	X	**01**	00	**10**			
ForwardB	X	X	00	X	00	00	00			
PCWrite	1	1	**0**	1	1	1	1			
IF/IDWrite	1	1	**0**	1	1	1	1			
ControlMux	0	0	**1**	0	0	0	0			

Only in clock-cycle 3 does the hazard detection unit need to insert a stall, so PCWrite, IF/ID.Write, and ControlMux values change in only this cycle. In clock-cycle 5, the first arg of ALU comes from the MEM/WB register so ForwardA=01. In clock-cycle 7, the first arg of ALU comes from the EX/MEM register, so ForwardA=10.

e) If there is no forwarding, what new input and output signals do we need for the hazard detection unit so that the original code executes correctly? Using this instruction sequence as an example, explain why each signal is needed.

Solution:

Since there is no forwarding, the hazard detection unit needs to:

1. Detect a data hazard (for which the unit needs inputs)
2. Stall the pipeline when data hazard is detected (for which the unit needs output signals)

Now:

1. A data hazard occurs when either of the source registers of an instruction depends on the destination of either the last or the 2nd last instruction. So, to detect a data hazard, the hazard detection unit needs access to:
   a. rs1, rs2 of the current instruction
   b. rd of the last instruction
   c. rd of the 2nd last instruction

   The hazard detection unit already has access to:
   IF/ID.RegisterRs1, IF/ID.RegisterRs2, (source register values of the instruction in the ID stage)
   ID/EX.RegisterRd (destination register value of the instruction in EX stage)
   The only value missing is the rd value of the 2nd last instruction i.e. the instruction in the MEM stage. So we only need to provide the value of EX/MEM.RegisterRd.
2. To stall the pipeline, no additional output is needed. The hazard detection unit is already fully equipped to insert a stall

In order to properly execute the original code given in the question, the hazard detection unit will be inserting the four nops that we inserted in part (a), by detecting the data hazards between `ld` and `and` instructions, and by detecting the data hazard between `sub` and `or` instructions, using the inputs we have mentioned above.