Data Hazards:-

Data Hazards :-

Occurs due to data dependencies between instructions that are various stages of execution in the pipeline.

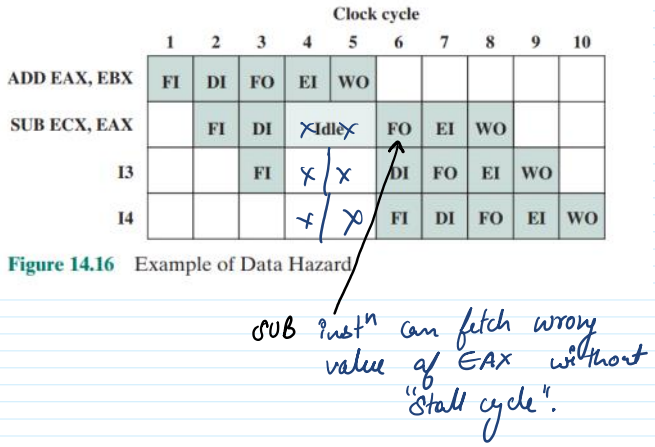


Figure 14.16 Example of Data Hazard

SUB inst^n can fetch wrong value of EAX without "Stall cycle".

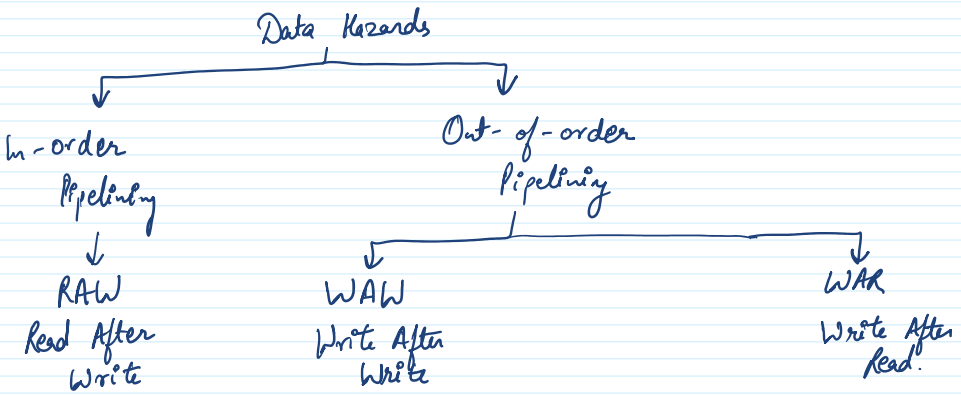
In-order Pipeline :-

A preceding inst^n is always ahead of a succeeding inst^n.

Out-of-order Pipeline :-

(Modern Processors)

It is possible for later instructions to execute before earlier instructions.



[1]: add r1, r2, r3
[2]: sub r3, r1, r4
[3]: mull r5, r8, r9

[1]: add r1, r2, r3
[2]: sub r1, r4, r3

Inst^n [2] cannot write the value of r1 before inst^n [1] writes to it.

[1]: add r1, r2, r3
[2]: add r2, r5, r6

Inst^n [2] cannot write the value of r2 before inst^n [1] read it.

Solutions :-

① Data Forwarding :-
/ bypassing

By using additional hadware, the data required can be forwarded as soon as they are computed, instead of waiting for the result to be written into the register

Instⁿ	1	2	3	4	5	6
ADD R2, R5, R8	FI	DI	FO	EI	WO	
SUB R9, R2, R6		FI	DI	FO	EI	WO

R2 written here

R2 read here

A naive solution:-

Instⁿ	1	2	3	4	5	6	7	8
ADD R2, R5, R8	FI	DI	FO	EI	WO			
SUB R9, R2, R6		FI	DIx	STALL	STALL	DI	FO	EI
Next Instⁿ			FIx	STALL	STALL	FI	DI	EX
Next Instⁿ				STALL	STALL		FI	DI
Next Instⁿ				STALL	STALL			FI

"3 clock cycles wasted"

Data forwading :- using additional H/W forward the data from ALU as soon as it is computing
/bypassing

Instⁿ	1	2	3	4	5	6	7	8	9
① ADD R4, R5, R6	FI	DI	FO	EI	WO				
② SUB R3, R4, R8		FI	DI	FO	EI	WO			
③ ADD R7, R2, R4			FI	DI	FO	EI	WO		
④ AND R9, R4, R10				FI	DI	FO	EI	WO	
⑤ OR R11, R4, R5					FI	DI	FO	EI	WO

Data flow

Concurrent register Access

Solution:-
Data Forwarding
Using additional hadware forward the result directly from the output of the ALU to the input registers of the next instruction.

Solution:-
Concurrent Register Access
Splitting a cycle into two halves
(register write in first half clock cycle, register read in second half clock cycle).

Control Hazard :-

Arises because of branch instructions being executed in a pipeline.
→ Can cause greater performance loss than data hazards.

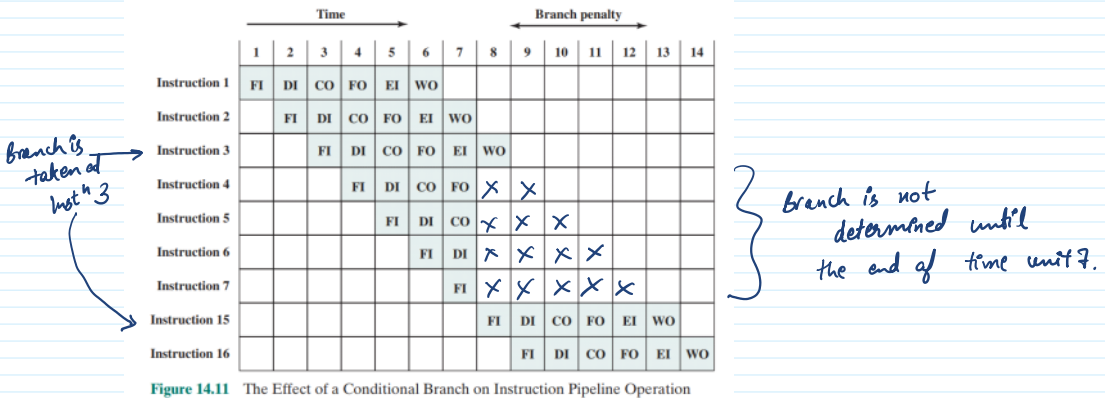


Figure 14.11 The Effect of a Conditional Branch on Instruction Pipeline Operation

Dealing with branches:-

1. Multiple streams
2. Prefetch Branch Target
3. Loop Buffer
4. Branch Prediction
5. Delayed Branch

Reading exercise :- Page 509-515

Computer Organization and Architecture, William Stallings (10th Edition)