

Computer Architecture

Fall, 2018

Week 12

2018.11.26

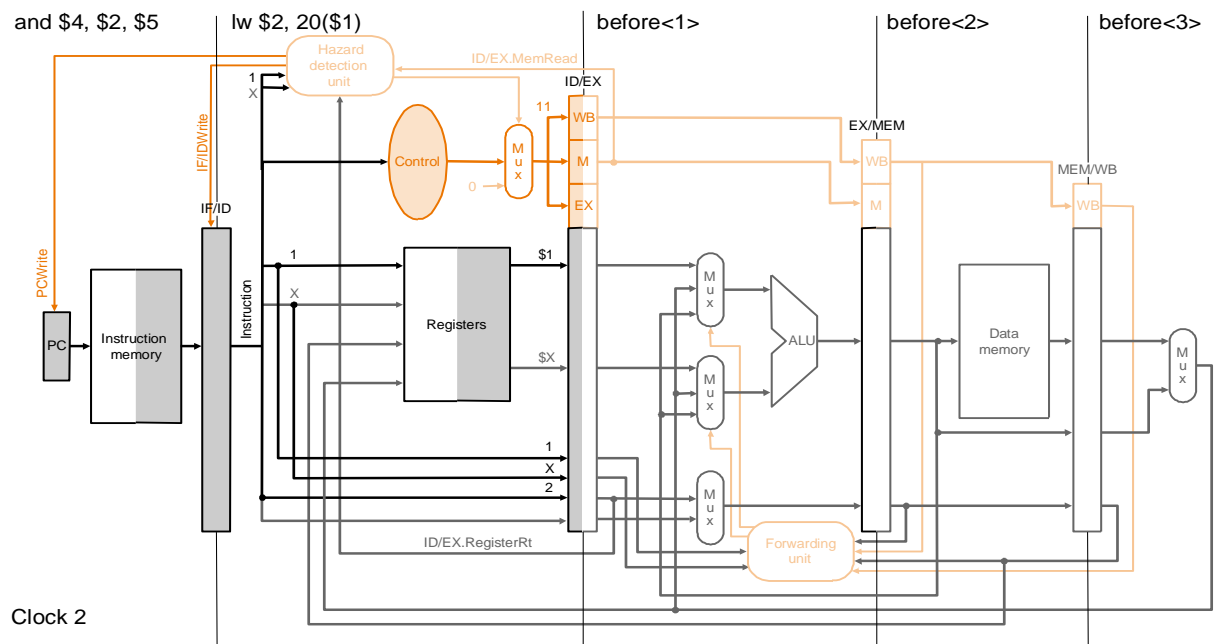
Group 1 (對抗賽)

1.

See following instructions:

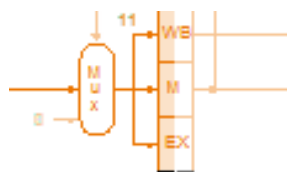
1. lw \$2,20(\$1)
2. and \$4,\$2,\$5
3. or \$8,\$2,\$6
4. add \$9,\$4,\$2
5. slt \$1,\$6,\$7

The following figure show that a pipeline execute these instructions in cycle2.



Questions

1. Which instructions will cause data hazard?
 2. See following mux, when will '0' be chose?
- A. cycle2 B. cycle3 C. cyce4 D. cyce5



Ans:

1.第 2 , 4 条指令会产生 data hazard , 期中第 2 条指令产生 load_use hazard,第 4 条指令产生 R_Type_use data hazard 。

2.B

Group 2 (對抗賽)

2.

For the following instruction,

1. Which dependencies are data hazards that can be resolved by forwarding?
2. Indicate how many pipeline stalls will occur and in which instruction?

add \$12, \$15, \$14

lw \$15, 100(\$12)

sub \$13, \$15, \$12

add \$12, \$15, \$13

ANS:

Instruction	CC1	CC2	CC3	CC4	CC5	CC6	CC7	CC8	
1)	IF	ID	EX	MEM	WB				
2)		IF	ID	EX	MEM	WB			
3)			IF	ID	stall	EX	MEM	WB	
4)				IF	stall	ID	EX	MEM	WB

Group 4

3.

Q : When handling Branch Hazard, if pipelines are deep, how can we solve the problem of significant branch penalty?

A : Dynamic prediction.

根據過去歷史推測 (branch prediction buffer, branch history table) , 如果過去為 taken , 就 predict taken , 如果過去為 not taken , 就 predict not taken 。當預測錯誤 , 則 flush pipeline and flip prediction 。

Group 8 (對抗賽)

4.

What is the different between “Exception” and “Interrupt”? Briefly explain

Ans:

一個 **unexpected event** 是發生在 CPU 裡頭 就叫 **Exception**

ex: **undefined opcode, overflow, sys call...**

如果是 I/O 發出的 **unexpected event** 就叫 **interrupt**

Group 7 (對抗賽)

5.

請列舉出至今所學過的 **hazards** 以及解決方式(**structural hazards**、**control hazards**、**data hazards**)

Ans :

Structural hazards :

硬體資源不夠多而導致同一時間內要執行的多個指令無法執行。

解決 : 合併多個功能或同段時間夾著做其他事

Control hazards :

當我們需要某個指令的結果來作一些決定，可是這個指令還在執行，無法馬上解決：

(a)**stall** : 如果知道是一個 **branch instruction**，則暫停直到正確的 **condition** 知道後才開始下一個 **instruction**。

(b)**predict** : 事先預測 **branch instruction** 不會發生，如果預測結果錯誤則放棄已經進入 **pipeline** 的指令而 **fetch branch target** 的 **instruction** 進入 **pipeline**，如果預測成功就可以快速順利的運作下去。

(c)**delayed branch** : 在 **branch instruction** 之後先執行下個合適的指令，但是必須確保這個指令是安全指令，不會影響 **program** 的執行結果才行。

Data hazards :

當一個 **instruction** 必須參考先前 **instruction** 的執行結果，但是先前的 **instruction** 卻還在 **pipeline** 中沒執行完，就會發生 **data hazard**。

解決：

Inserting **NOP** (**software**)

Forwarding, R-Type-use (**hardware**)

Stalls, load-use (**hardware**)

Group 7

6.

請簡單解釋 Superscalar、Dynamic pipeline scheduling 的意思

Ans :

Superscalar :

增加 functional units，使的一個 clock cycle 可以執行多個 instruction。

Dynamic pipeline scheduling :

利用一些額外的 hardware 來防止 hazards 產生。

(Allow the CPU to execute instructions out of order to avoid stalls. But commit result to registers in order)

Group 9 (對抗賽)

8.

Exception 發生在 pipeline 是怎麼處理的?

Answer:

- 1.不要讓這個指令完成
- 2.將前面的指令完成
- 3.把發生 exception 跟他之後的指令 flush
- 4.把 Cause 跟 EPC 設定好
- 5.跳去 OS

Group 11 (對抗賽)

8.

Given following code

a = b + e

c = b + f

Assuming all variables are in memory and are addressable as offset from register t0, we have MIPS instructions below:

L1 ld t1, 0(t0) // Load b

L2 ld t2, 8(t0) // Load e

L3 add t3,t1,t2//b+e

L4 sd t3, 24(t0) // Store a

L5 ld t4, 16(t0) // Load f

L6 add t5,t1,t4//b+f

L7 sd t5, 32(t0)

Please point out which kind of hazard would happen, where is(are) the hazard(s)?

How do we remove it by changing the order of instruction?

Ans:

Data-hazard (load-use data hazard)

L3 and L6 have hazard

Reorder as follow

L1 ld t1, 0(t0) // Load b

L2 ld t2, 8(t0) // Load e

L5 ld t4, 16(t0) // Load f

L3 add t3,t1,t2//b+e

L4 sd t3, 24(t0) // Store a

L6 add t5,t1,t4//b+f

L7 sd t5, 32(t0)