Computer Architecture
Fall, 2018
Week 11
2018.11.19

[group11]

1. (超過本週範圍)

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  lw  t1, 0(t0)  // Load b
L2  lw  t2, 8(t0)  // Load e
L3  add t3, t1, t2 // b+e
L4  sw  t3, 24(t0) // Store a
L5  lw  t4, 16(t0) // Load f
L6  add t5, t1, t4 // b+f
L7  sw  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   lw   t1, 0(t0)   // Load b
L2   lw   t2, 8(t0)   // Load e
L5   lw   t4, 16(t0)        // Load f
L3   add  t3, t1, t2  // b+e
L4   sw   t3, 24(t0)        // Store a
L6   add  t5, t1, t4  // b+f
L7   sw   t5, 32(t0)
```

[group7]

2.

為了實現 datapath forwarding，需要再 ALU 前面加上一個 3-1MUX。請問 3-1 分別接上甚麼訊號?為甚麼?

Ans: 三個信號及其用途為:

i. 原始的 input: 本來就要傳的東西

ii. MEM stage 的信號: 如果當前指令的 rs 或 rt 跟前一指令的 rd 相同時使用，避免還沒寫進 register 就拿來用

iii. WB stage 的信號: 如果當前指令的 rs 或 rt 跟兩指令前的 rd 相同，且與前一指令的 rd 不同時使用，避免資料還沒讀進 register 就拿來用

[group9]

3.

Identify all of the data dependencies in the following code and show which dependencies are data hazards.

```
add $2, $5, $4
add $4, $2, $5
sw  $5, 90($2)
add $3, $2, $4
```

A1.

	Data dependency	Data hazard
$2	(1,2) (1,3) (1,4)	(1,2) (1,3)
$4	(2,4)	(2,4)

[group6]

4.

Consider the following program in MIPS code. Please count how many forwarding happened here?

```
sub     $t0, $s1, $t0
and     $s0, $t0, $v0
addi    $t0, $s0, 0
sw      $t8, 0($s0)
sw      $t9, 0($s0)
```

Answer: 3 (line2, line3, line4)

[group1]

5.

There are three types of data hazard in pipelining.

(1) Please explain the reason of the occurrence of RAW, WAR and WAW.

(2) Which of them don't happen in MIPS 5-stage pipeline and why?

Ans:

(1)

RAW : I2 tries to read operands before I1 writes it.

WAR : I2 tries to write operands before I1 reads it.

WAW : I2 tries to write operands before I1 writes it.

(2)

WAR&WAW. Because all instructions in MIPS 5-stage take 5 stages, and reads are always in stage 2, and writes are always in stage 5.

[group12]

6.

Solve all the hazards in this instruction sequence for a five-stage pipeline with forwarding and without forwarding.

```
lw  $1, 40($2)
add $2, $3, $3
add $1, $1, $2
sw  $1, 20($2)
```

Ans.

Without forwarding :

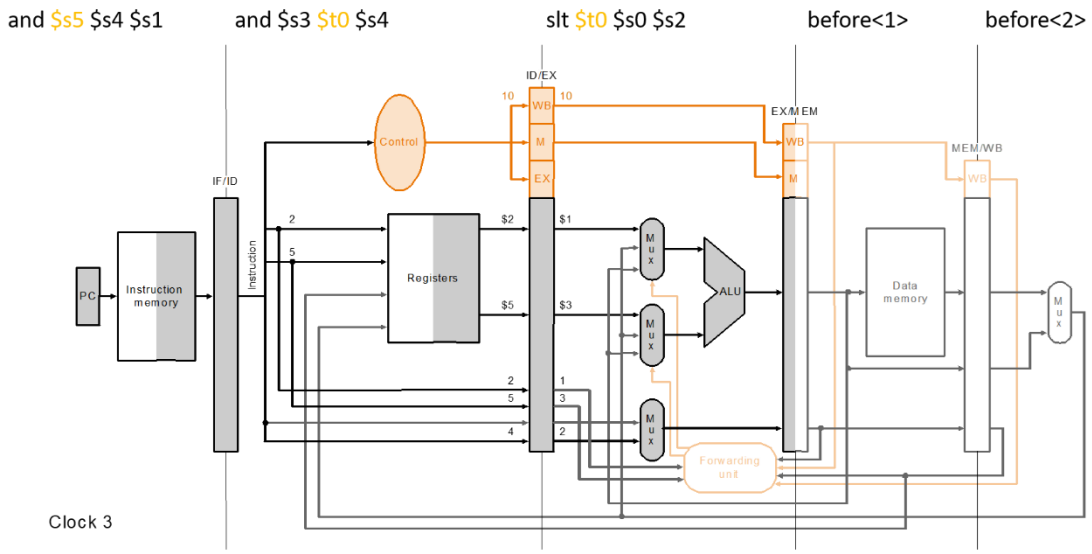
Instructions	Pipeline stage
lw $1, 40($6)	IF ID EX MEM WB
add $2, $3, $3	IF ID EX MEM WB
nop	IF ID EX MEM WB
nop	IF ID EX MEM WB
add $1, $1, $2	IF ID EX MEM WB
nop	IF ID EX MEM WB
nop	IF ID EX MEM WB
sw $1, 20($2)	IF ID EX MEM WB

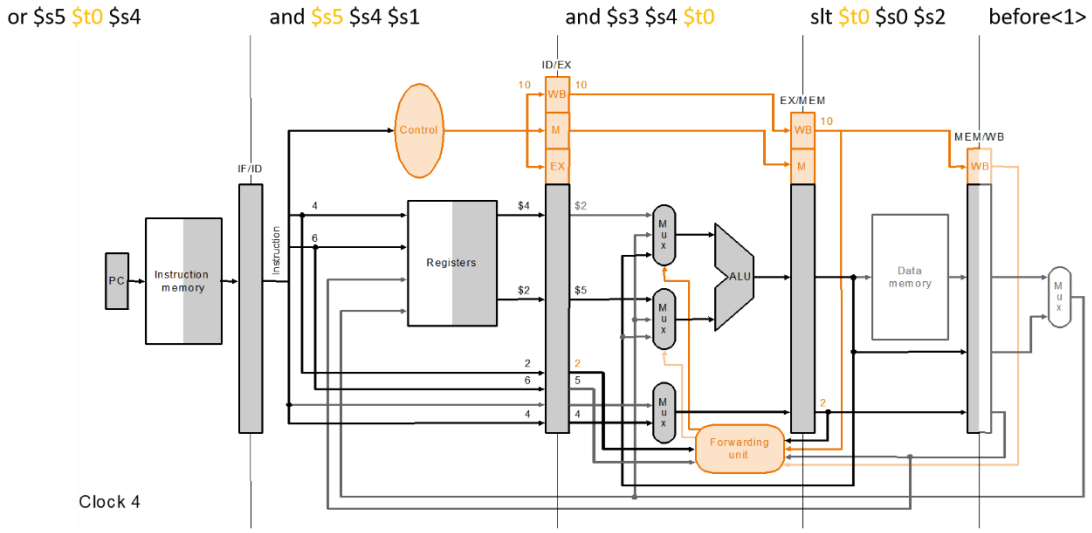
With forwarding :

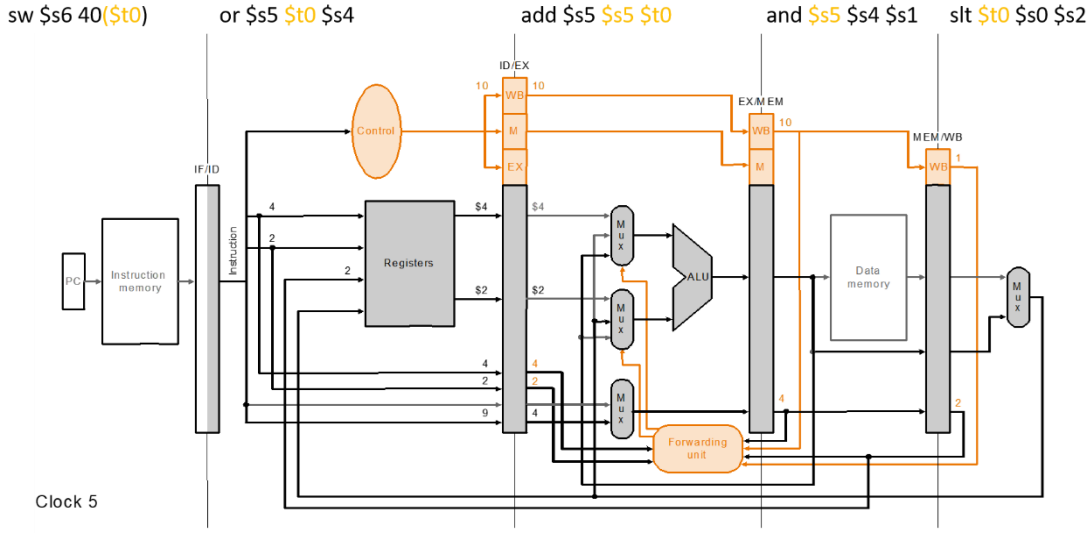
Instructions	Pipeline stage
lw $1, 40($6)	IF ID EX MEM WB
add $2, $3, $3	IF ID EX MEM WB
add $1, $1, $2	IF ID EX MEM WB
sw $1, 20($2)	IF ID EX MEM WB

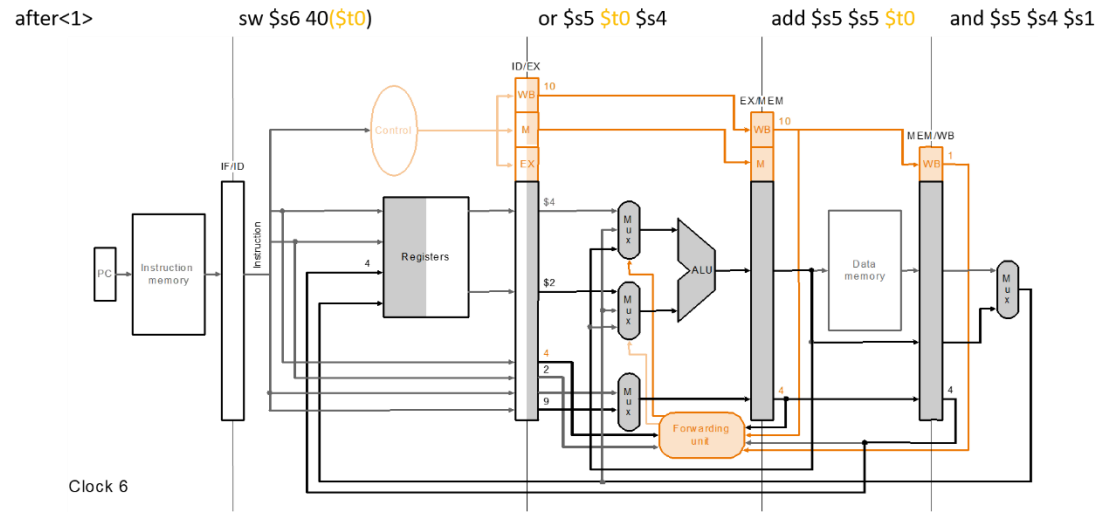
[group10]

7.

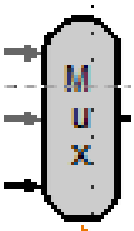

and $s5 $s4 $s1
and $s3 $t0 $s4
slt $t0 $s0 $s2
before<1>
before<2>
ID/EX
EX/MEM
MEM/WB
Control
IF/ID
Instruction memory
Registers
ALU
Data memory
Forwarding unit
Clock 3


or $s5 $t0 $s4
and $s5 $s4 $s1
and $s3 $s4 $t0
slt $t0 $s0 $s2
before<1>
ID/EX
EX/MEM
MEM/WB
Control
IF/ID
Instruction memory
Registers
ALU
Data memory
Forwarding unit
Clock 4


sw $s6 40($t0)
or $s5 $t0 $s4
add $s5 $s5 $t0
and $s5 $s4 $s1
slt $t0 $s0 $s2
ID/EX
EX/MEM
MEM/WB
Control
IF/ID
Instruction memory
Registers
ALU
Data memory
Forwarding unit
Clock 5



忽略圖片上 $2 $4 等符號，依照下列指令分別寫出 四張圖 ALU 兩 MUX 的



control signal (由上至下為 00 01 10)

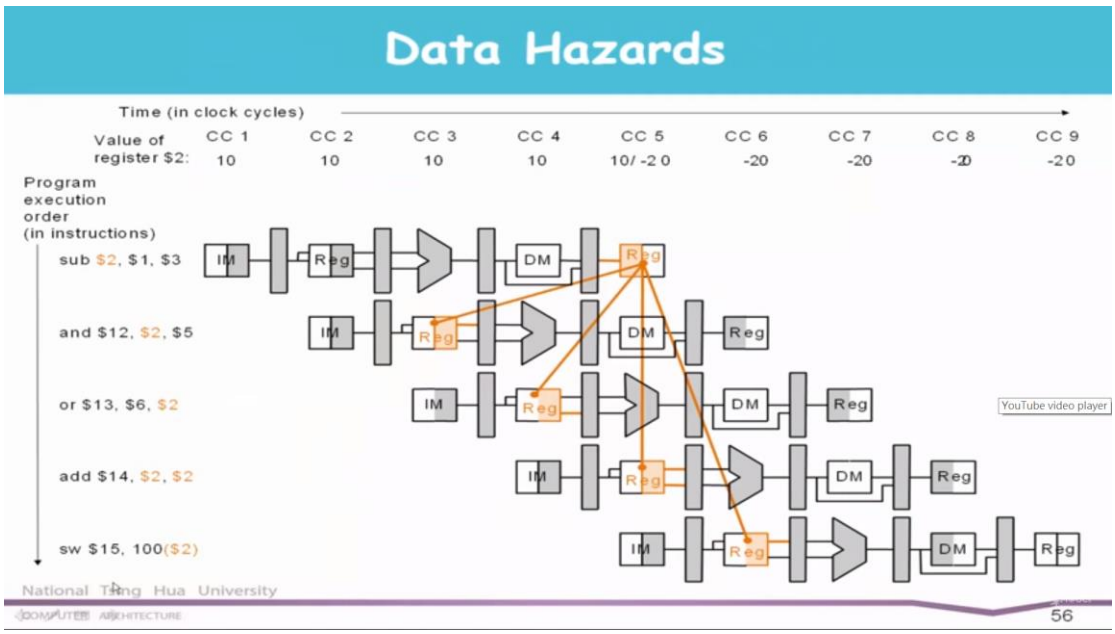
```
slt     $t0 $s0 $s2
and     $s3 $t0 $s4
addi    $s4 $t0 -5
or      $s5 $s4 $t0
sw      $s6 40($t0)
```

Ans:

cycle	上 MUX	下 MUX
3	10	00
4	01	00
5	10	00
6	00	00

[group2]

8.



Refer to the above diagram:

1.

Without using NOP and forwarding method ,

please explain why data hazard would happen? What type of data hazard is it?

2.

Which instruction(s) use(s) the wrong data value? Which instruction(s) use(s) the wright one?

Ans:

1

因為其中幾個 instruction 要讀 data 時，要被讀的 register 中放的不是正確的值。發生 read after write(RAW) hazard。

2.

第 5 個 clock cycle (cc5) 時，sub 才會把資料寫進$2，但 and 跟 or 分別在 cc2 跟 cc3 時就要讀$2 裡面的值，所以 and 跟 or 會讀到錯誤的值。

而 add 在 cc5 要讀，跟 sub 寫進$2 是同一個 clock cycle，但 register file 內有 internal forwarding，所以會讀到正確的值。

sw 的讀是在 cc6，是在 sub 把正確值寫進$2 的後面的 clock cycle，所以是用到正確的值。