Computer Architecture

Fall, 2023
Week 11
2023.11.20

組別：__________ 簽名：____

[group8]

1.

a. What is the concept of a pipeline?
b. What is a structural hazard in pipelining?
c. Why is pipelining easily implemented in MIPS?

Ans :

a. Pipelining in computer architecture is a technique to increase the efficiency of instruction processing. It involves dividing the execution of an instruction into several stages, with each stage handled by different parts of the processor. This allows multiple instructions to be processed in overlapping phases, improving overall performance.
b. A structural hazard in pipelining occurs when multiple instructions compete for the same hardware resources at the same time. It's a limitation due to resource conflicts, such as when two instructions need the same part of the processor simultaneously
c. (1) All instructions are the same length
(2) Just few instruction formats
(3) Memory operands only in loads and stores

[group13]

2. Consider the following MIPS code:

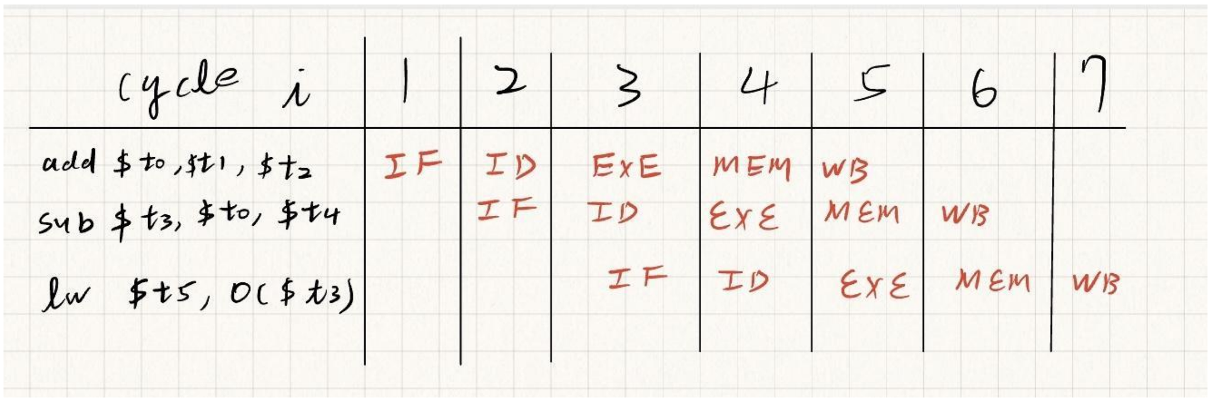
```
add   $t0, $t1, $t2      # Add contents of $t1 and $t2, store result in $t0
sub   $t3, $t0, $t4      # Subtract contents of $t4 from $t0, store result in $t3
lw    $t5, 0($t3)        # Load word from memory address contained in $t3 into $t5
```

Now we have a 5-stage MIPS pipelined processor. The 5 stages are instruction fetch(IF), instruction decode(ID), ALU Execution(EXE), Memory Fetch(MEM), and write back to register(WB). There is no optimization and hazard in the above MIPS code. Answer the following two questions.

(a) How many cycles will the 5-stage pipelined processor cost to execute the above MIPS code?
(b) If we can deal with the structural hazard by adding a NOP stage, consider the MIPS code with n instructions that may only have structural hazard and doesn't involve any other optimization technique. How many cycles will the 5-stage pipelined processor cost to execute the above MIPS code? Please write a formula associated with n.

Ans:

(a) There are 5 stages in total. The following timeline diagram shows the time sections the instructions will be executed in 5 stage pipeline processor:



cycle i	1	2	3	4	5	6	7
add \$t0, \$t1, \$t2	IF	ID	ExE	MEM	WB		
sub \$t3, \$t0, \$t4		IF	ID	ExE	MEM	WB	
lw \$t5, 0(\$t3)			IF	ID	ExE	MEM	WB

(b) The processor will cost n + 4 cycles for the scenario without any other hazards and optimization. The first instruction will be fetched in the first cycle and will be finished in the fifth cycle. The second instruction will be fetched in the second cycle and will be finished in the sixth cycle. The third instruction will be fetched in the third cycle and will be finished in the seventh cycle. By mathematical induction, we can conclude that nth in struction will be fetched at nth cycle and will be finished in the (n + 4) th cycle. As a conclusion, there will be n + 4 cycles to execute the codes in this scenario.

[group9]

3.

(a) Assign different actions to their corresponding stages.

Ex. PC = PC + 4 → IF

(1) ___ if (A == B) then PC = ALUOut

(2) ___ IR = Memory[PC]

(3) ___ ALUOut = PC + (sign-extend (IR[15-0]) << 2)

(4) ___ Reg[IR[15-11]] = ALUOut

Stages：

A. IF (instruction fetch)

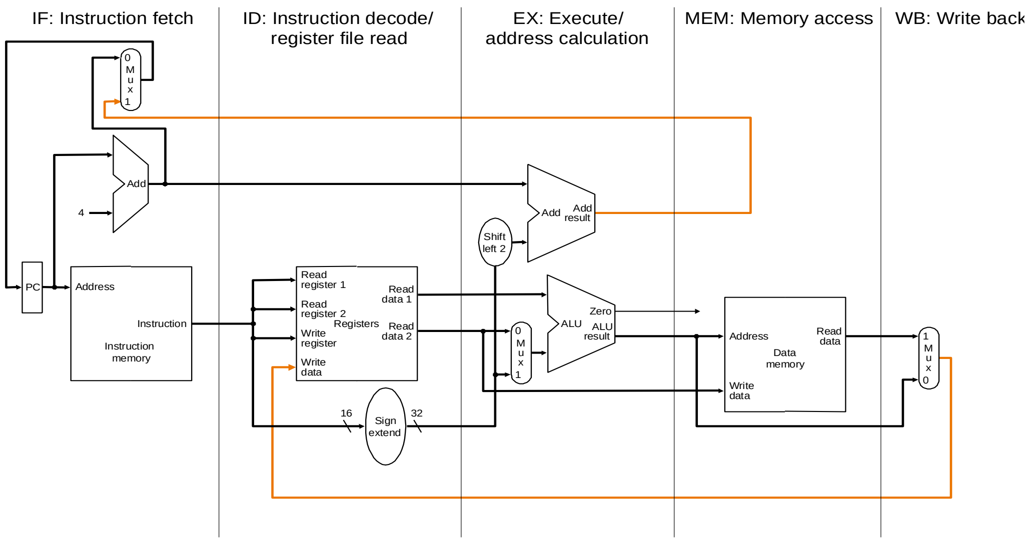
B. ID (instruction decode and register file read)

C. EX (execution or address calculation)

D. MEM (data memory access)

E. WB (write back)

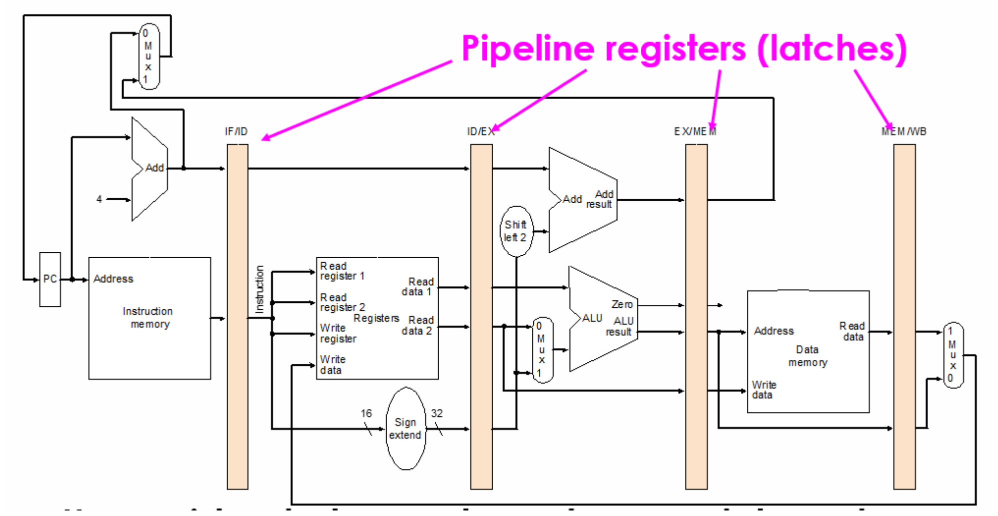
(b) Where to add the pipeline registers in the following diagram?



Ans:

(a) EX, IF, EX, WB

(b)



[group6]

4. Compare pipelining with single-cycle, choose the correct statement.

   (a) Pipelining is faster since the latency is shorter.

   (b) Compare to single-cycle, the speedup of pipelining = the number of pipes stage.

   (c) In pipelining , if there are 5 resources, then it may run at 4 instructions at some moment.

   (d) When designing a pipelining processor, we need to ensure flows do not conflict, or figure out how to resolve.

   Ans: (c)、(d)

   (a) latency 沒有變少

   (b) 會比 stage 的個數(n)再慢一點，因為每個 stage 的時間不一定一樣，且頭尾的時間並不會同時都有 n 個工作再做。

   (c)最前面或最後面可能不會每個 resource 都同時有在 work。

[group4]

5. Suggest you have a CPU with pipeline architecture, the same as the current teacher has taught. Answer True/False for the following questions:

(a) If the latency of all instructions in the program is 5 clock cycles, the average CPI should be 5.
(b) If the critical path of **IF, ID, EX, MEM, WB** takes 150ps, 200ps, 130ps, 450ps, 200ps, the latency of one instruction would be 1130ps.
(c) After totally executing "**addi $t0, $zero, 1**" and then "**add $t0, $t0, $t0**" two instructions, the value of $t0 is 2.

Answer:
(a) False, the execution time can overlap, so it may be smaller than 5.
(b) False, the time between stages should be determined by the longest stage, so it should be 2250ps.
(c) False, when $t0 is read in the second instruction, the result of the first instruction has not been written to the register yet.

[group5]

6. What is the order of the stages considering the instruction load and R-type, respectively? What is the problem when pipelining load and R-type?

(a) Data memory access
(b) Instruction fetch
(c) Write back
(d) Instruction decode
(e) Execution or address calculation

Ans:
load: (b) => (d) => (e) => (a) => (c)
R-type: (b) => (d) => (e) => (c)
Since load uses register file's write port during its 5th stage while R-type using it during the 4th stage , we have a structural hazard when pipelining them.

[group1]

7. There is a program with two types of instructions executed in the sequence: InstA -> InstB -> InstA -> InstA -> InstB.

   (a) How long does it take if we execute the program with one instruction per cycle?

(b) And how long does it take if we execute the program with pipeline?

(c) In this case, how much faster is the pipeline compared to one instruction per cycle? (Following are details for two instructions.)

InstA: Ifetch -> Reg/Dec -> Exec -> Wr

InstB: Ifetch -> Reg/Dec -> Exec -> Mem -> Wr

Task type	Time cost (ps)
Ifetch	250
Reg/Dec	100
Exec	220
Mem	250
Wr	130

Answer:

(a) Single cycle: $(250 + 100 + 220 + 250 + 130) \times 5 = 4750ps$

(b) Pipeline: $250 \times (4 + 5) = 2250ps$

(c) $\frac{4750}{2250} = \approx 2.11$

In this case, pipeline is 1.78 times faster as single cycle.

[group12]

8. Assuming that Instruction Fetch, Instruction Decode, Execute, Memory Access, Writing each takes 40ns, 30ns, 50ns, 20ns, 100ns respectively.

Calculate the total time and the number of cycles needed to process N load-word instructions using:

a. a single cycle processor
b. a processor that uses pipeline (assume that each clock cycle length is the length of the longest instruction delay)
c. Using the result from (a) and (b), at what value of N does the processor that uses pipeline become faster than a single cycle processor?

(Assume that no hazards could occur)

Ans:

(a)

A single cycle processor would process only one instruction at each cycle

So we have IF+ID+EX+MEM+WB = 40 + 30 + 50 + 20 + 100 = 240 ns for each instruction.

Thus it takes 240*N ns and N cycles to process N load word instructions.

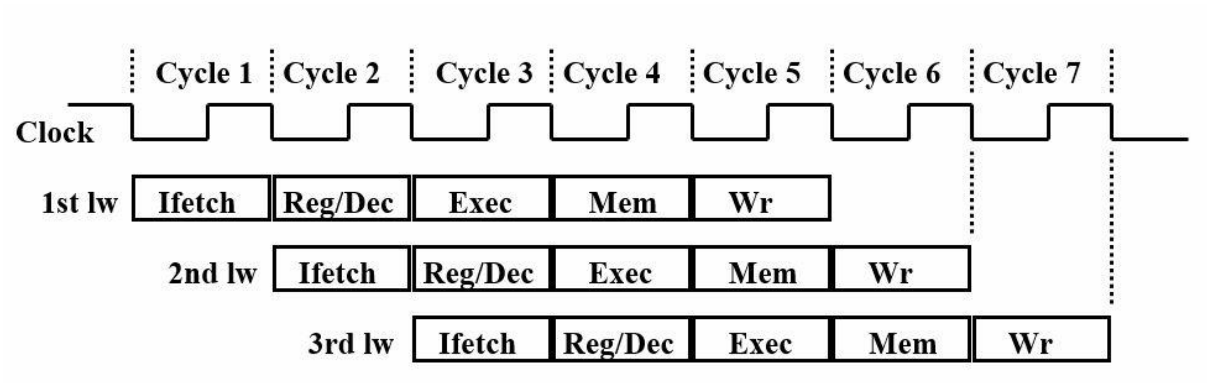
A processor that uses a pipeline can use the longest instruction delay time as the length of one clock cycle.

(b)

For executing a single instruction it would take 5 cycles to complete.

For executing two instructions it would take 6 cycles to complete.

For executing three instructions it would take 7 cycles to complete.



Thus, N instructions would take (4+N) * 100 ns and 4+N cycles to complete.

(c) We have to find the min N where (4+N) * 100 < 240*N,

=> 400 < 140N

=> 400/140 < N

=> ~2.85 < N

N = 3

So at only executing 3 instructions, the processor that uses pipeline is faster than the single cycle processor.