Computer Organization

Pipelined CPU

Bogyeong Park (arch23-ta@hpcs.snu.ac.kr)
High Performance Computer System (HPCS) Lab

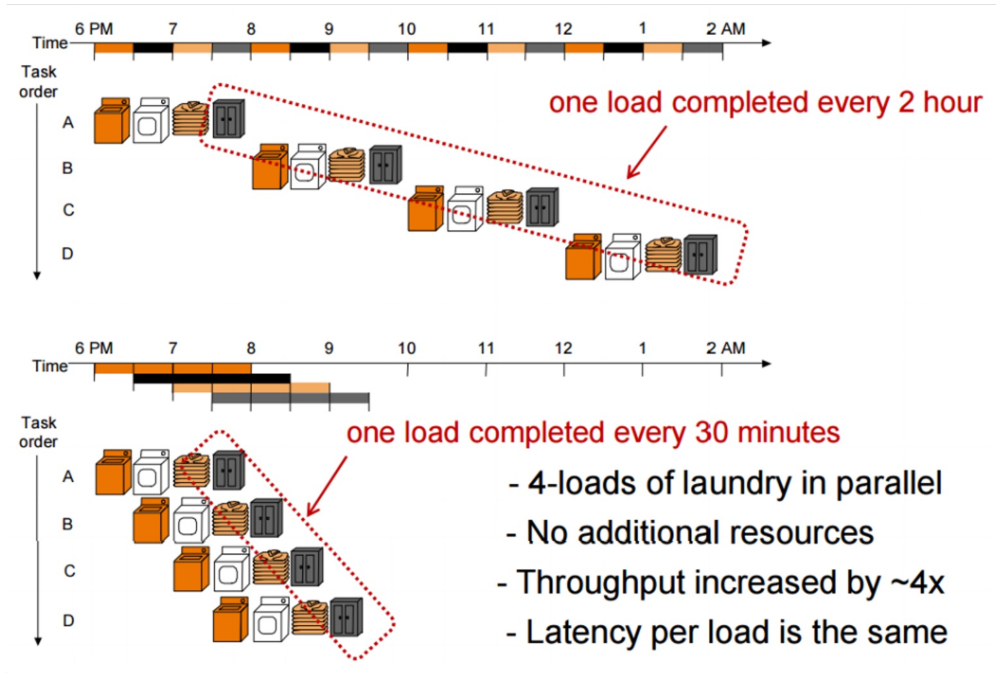
April 24th, 2023

Objectives

- Understand why pipelined CPUs show better performance than single- and multi-cycle CPUs.
- Understand what are the control/data hazards and how to resolve them.
- Design and implement a pipelined CPU.
    - Refer to the textbook (Section 4.6 - 4.6) and lecture slides.

Why pipelining? - Analogy

- Multiple instructions are overlapped in execution.
  - Task = a basket of laundry
  - Resource = Machines / Furniture


6 PM
7
8
9
10
11
12
1
2 AM
Time
Task order
A
B
C
D
one load completed every 2 hour
one load completed every 30 minutes
- 4-loads of laundry in parallel
- No additional resources
- Throughput increased by ~4x
- Latency per load is the same

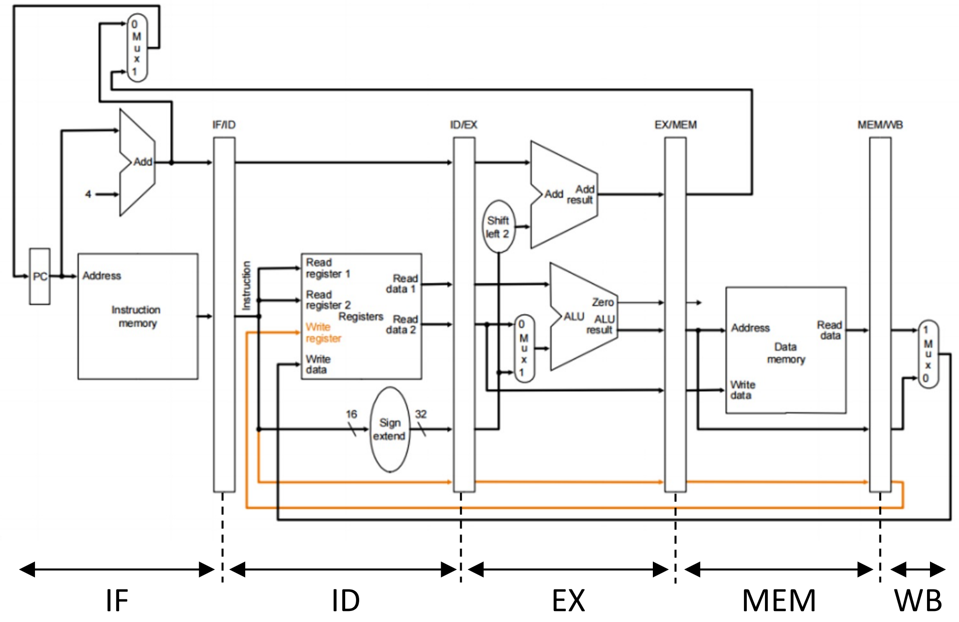
	Laundry
Task	A basket of laundry
Resource	Machines/Furniture

⬇

	CPU
Task	Instruction
Resource	Pipeline Stage (e.g., IF)

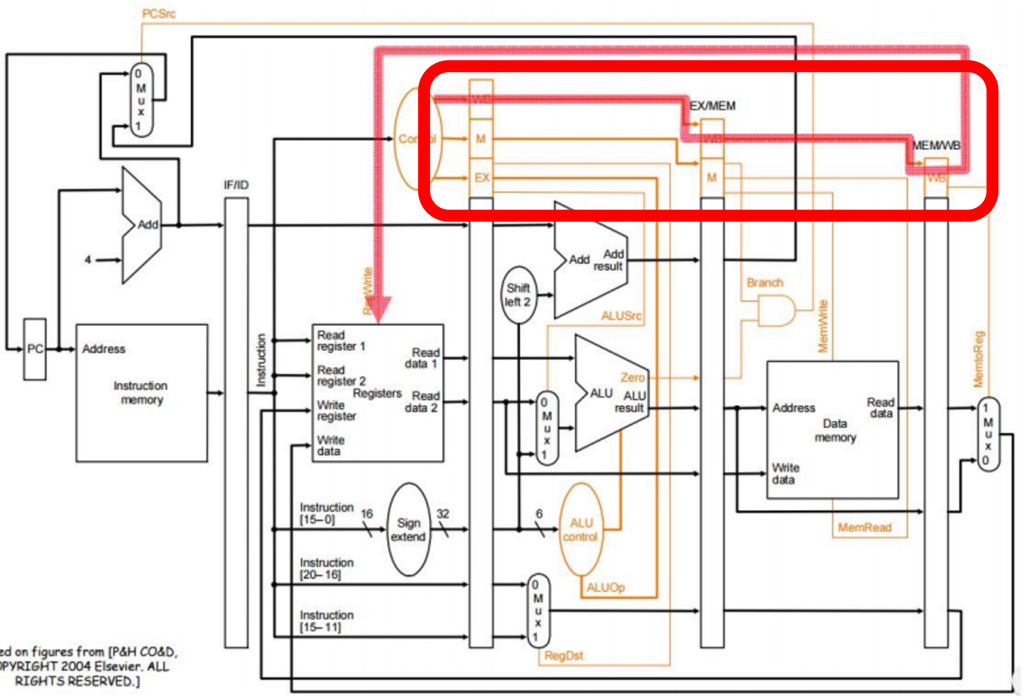
Pipelining – Datapath-only

- The transformed data of an instruction flows through the pipeline stages and pipeline latches.
- IF -> (IF/ID latch) –[next cycle] -> ID -> (ID/EX latch) –[next cycle] -> …



Pipelining – With Control

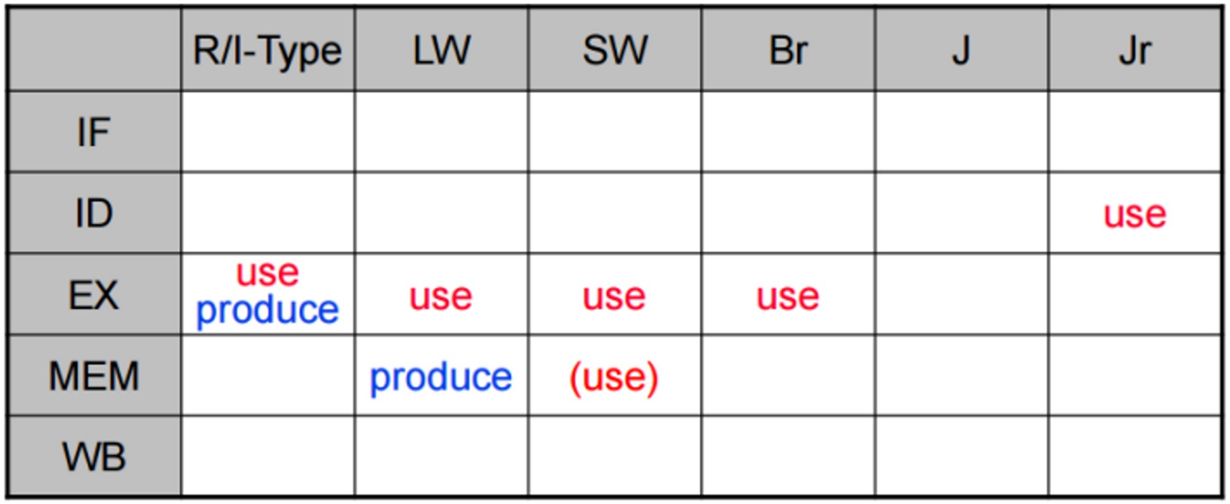
- Each stage requires different set of control signals for different instructions in the pipeline.
- Thus, control signals also flow through the latches



ased on figures from [P&H CO&D, COPYRIGHT 2004 Elsevier. ALL RIGHTS RESERVED.]

How to resolve data hazard?

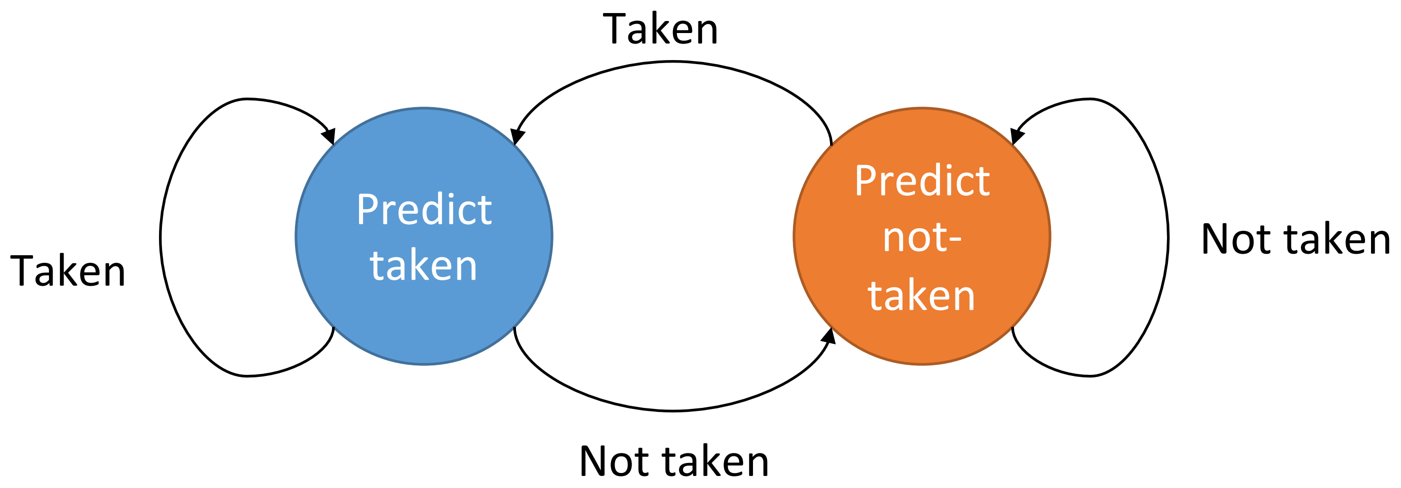
- Solution 1: Pipeline stall (i.e., insert the bubbles)
  - Stop advancing PC and disable latching
- Solution 2: Data forwarding
  - Deliver the dependent data right after it is produced
  - C.f.) Data use/produce table for data forwarding



	R/I-Type	LW	SW	Br	J	Jr
IF						
ID						use
EX	use produce	use	use	use		
MEM		produce	(use)			
WB						

How to resolve control hazard?

- How to resolve?
  - Solution 1: Pipeline stall
    - Stop pipeline advancing until the control hazard is resolved
  - Solution 2: Branch prediction (+ flush on miss)
    - Always-taken, always-not-taken, or other advanced prediction techniques



Assignment

- Implement a pipelined CPU (baseline)
    - <u>Your CPU module must pass all provided test cases</u>
    - It is **pipeline stall**(data hazard) + **always-taken** pipelined CPU
- Compare the performance between multi-cycle CPU and pipelined CPU
    - <u>Your report must include the comparison on # of clock cycles to execute all test cases</u>

Extra credits

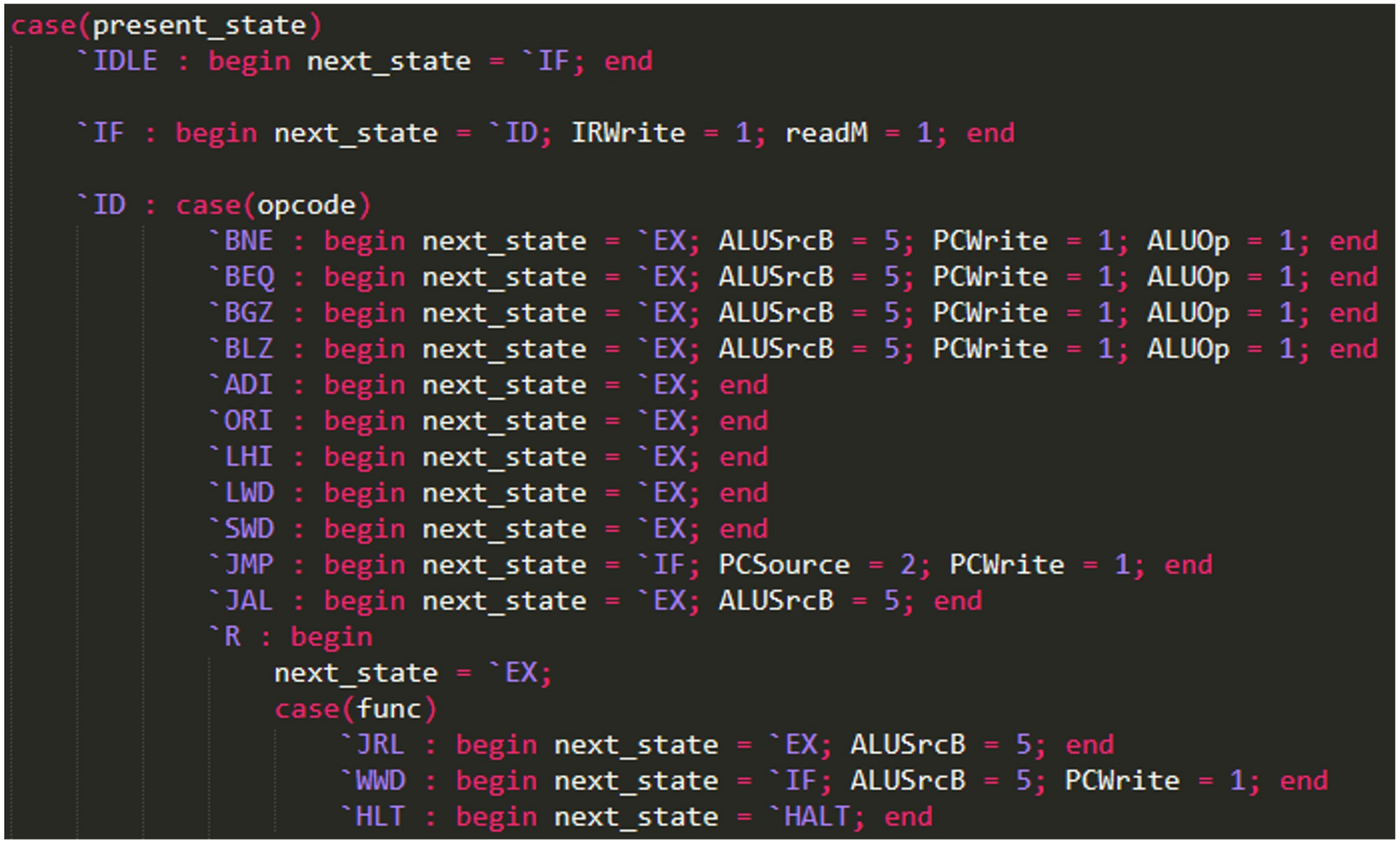
[1] Performance of the **data forwarding**

- Compares the performances between (1) with data forwarding and (2) without data forwarding (baseline)

[2] Branch predictor

- You need to implement a branch predictor and compare its performance to [1]
- Provide the accuracy of your branch predictor
- Simple predictor is **NOT** allowed
  (at least 2-bit up/down counter based predictor)

Tip 1, Use macro!

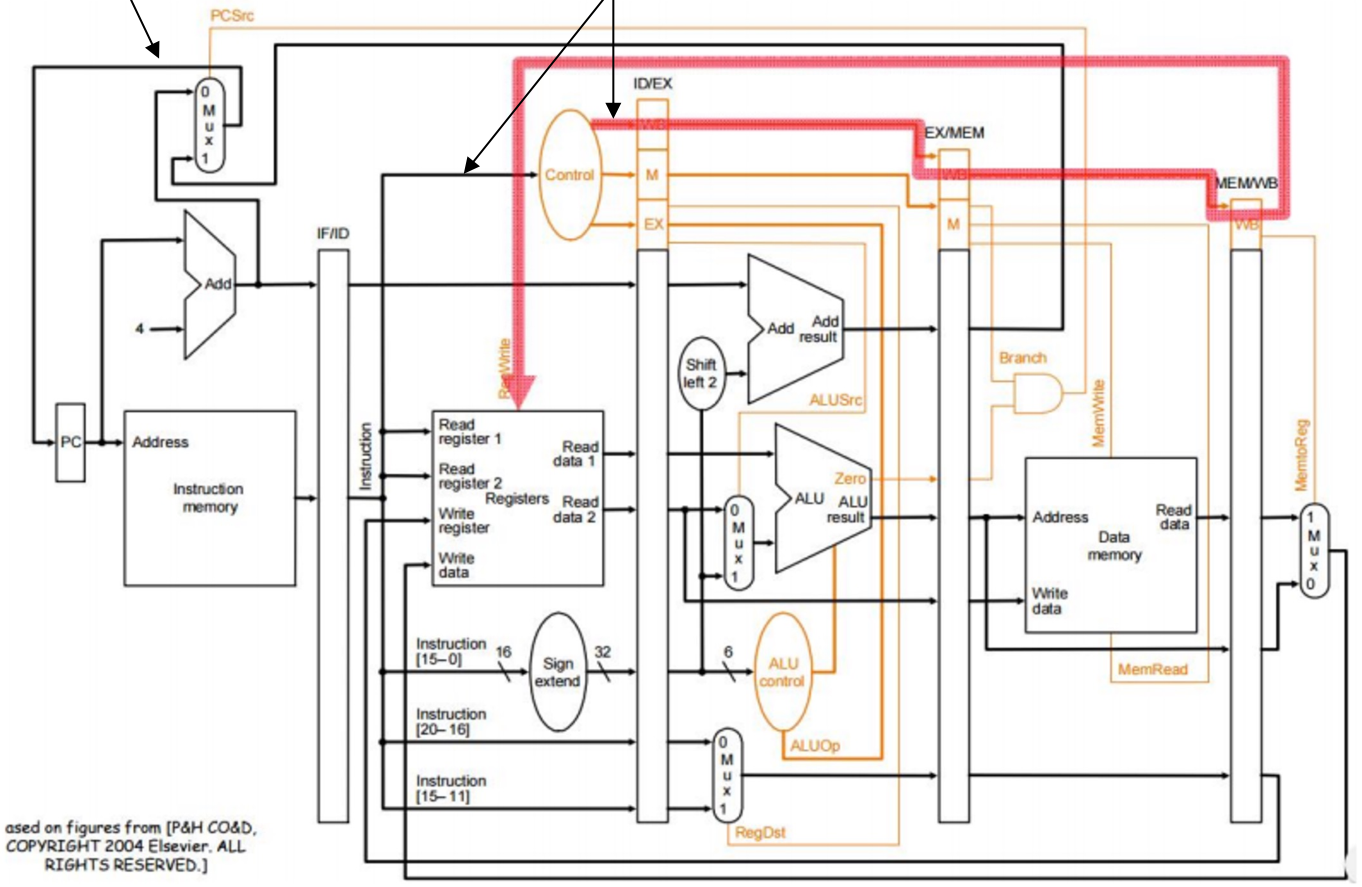


```
case(present_state)
    `IDLE : begin next_state = `IF; end

    `IF : begin next_state = `ID; IRWrite = 1; readM = 1; end

    `ID : case(opcode)
            `BNE : begin next_state = `EX; ALUSrcB = 5; PCWrite = 1; ALUOp = 1; end
            `BEQ : begin next_state = `EX; ALUSrcB = 5; PCWrite = 1; ALUOp = 1; end
            `BGZ : begin next_state = `EX; ALUSrcB = 5; PCWrite = 1; ALUOp = 1; end
            `BLZ : begin next_state = `EX; ALUSrcB = 5; PCWrite = 1; ALUOp = 1; end
            `ADI : begin next_state = `EX; end
            `ORI : begin next_state = `EX; end
            `LHI : begin next_state = `EX; end
            `LWD : begin next_state = `EX; end
            `SWD : begin next_state = `EX; end
            `JMP : begin next_state = `IF; PCSource = 2; PCWrite = 1; end
            `JAL : begin next_state = `EX; ALUSrcB = 5; end
            `R : begin
                next_state = `EX;
                case(func)
                    `JRL : begin next_state = `EX; ALUSrcB = 5; end
                    `WWD : begin next_state = `IF; ALUSrcB = 5; PCWrite = 1; end
                    `HLT : begin next_state = `HALT; end
```

Tip 2, Redraw the slide's figure!

Name all the wires!

What signals are there in those bundles?



ased on figures from [P&H CO&D, COPYRIGHT 2004 Elsevier. ALL RIGHTS RESERVED.]