Microprocessor & Computer Architecture (µpCA)

UE19CS252

Dr. D. C. Kiran

Department of
Computer Science and Engineering

Microprocessor & Computer Architecture (µpCA)

Pipeline Processor- Hazards

Dr. D. C. Kiran
Department of Computer Science and Engineering

Microprocessor & Computer Architecture (µpCA)

Syllabus

~~Unit 1: **Basic Processor Architecture and Design**~~

Unit 2: Pipelined Processor and Design

- ~~3-Stage ARM Processor~~
- ~~5-Stage Pipeline Processor~~
- ~~Introduction to Pipeline Processor~~
- ~~Understanding the Pipeline Execution~~
- What May Go Wrong
  - Introduction to Hazards
  - Attacking Hazards
  - Performance with Stalls
  - Structural Hazards

Unit 3: **Memory Design**

Unit 4: **Input/Output Device Design**

Unit 5: **Advanced Architecture**

Microprocessor & Computer Architecture (µpCA)

Text 1: "Computer Organization and Design", Patterson, Hennessey, 5th Edition, Morgan Kaufmann, 2014.

Reference 1:"Computer Architecture: A Quantitative Approach", Hennessey, Patterson, 5th Edition, Morgan Kaufmann, 2011.

Appendix C **Pipelining: Basic and Intermediate Concepts**

Facts About Pipeline Processor

- Pipelining increases the CPU instruction throughput. ***(Pipeline Throughput)*** Ideally, CPI = 1.
- Pipelining does not reduce the execution time of an instruction. ***(Pipeline Latency)***
- Infact, it slightly increases the execution time due to the increased **control overhead** of the pipeline stage register delays.
- All instructions that share a pipeline must have the same stages in the same order.
  - ✓ **Add** does nothing during Mem stage
  - ✓ **sw** does nothing during WB stage
  - ✓ **b{cond}** does nothing in EX, MEM & WB stage

Microprocessor & Computer Architecture (µpCA)

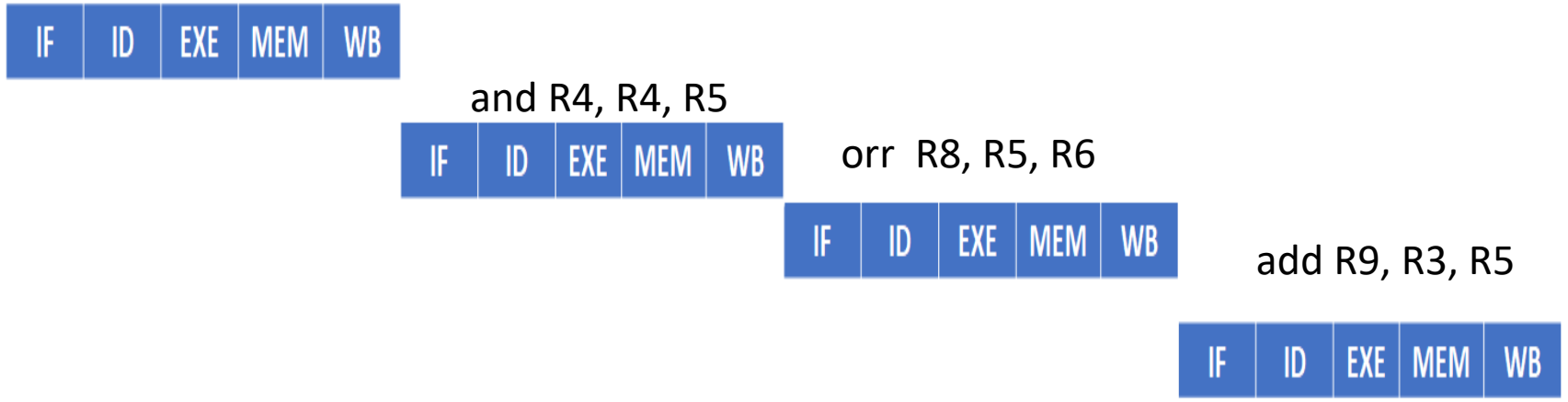
Non-Pipeline Execution

sub R2, R1, R3

and R4, R4, R5

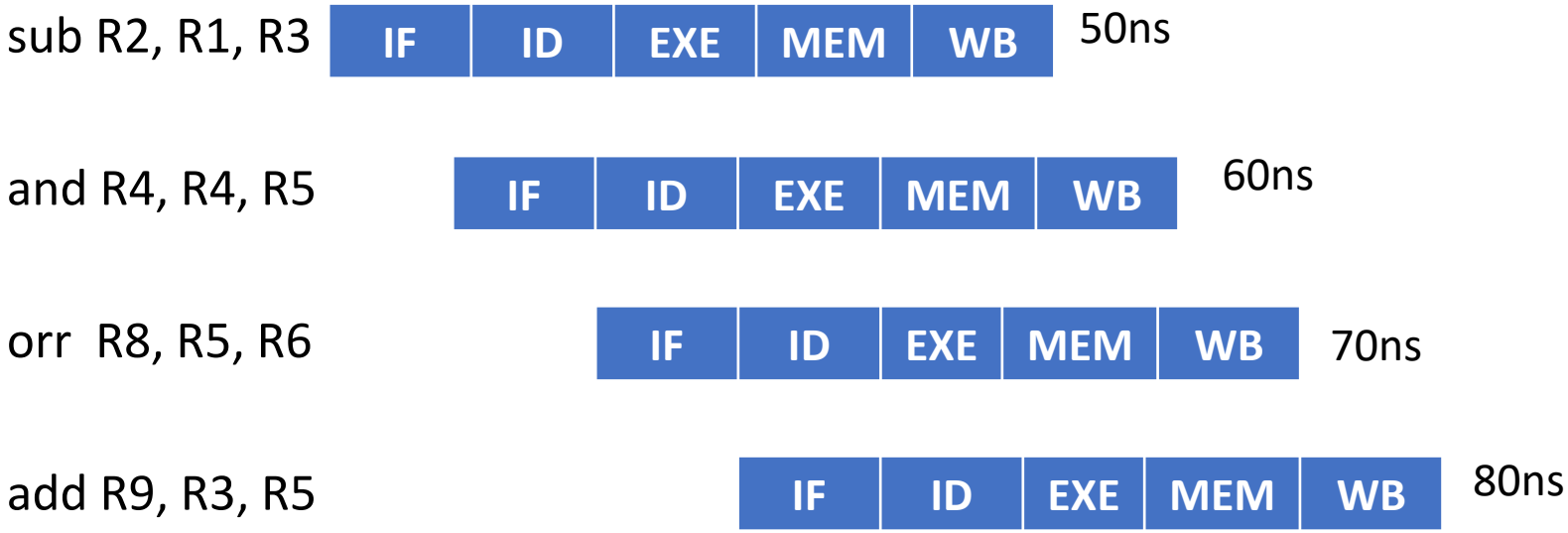
orr R8, R5, R6

add R9, R3, R5



If Each stage takes 10ns, the latency will be 50 ns and the throughput is 200 ns

Pipelining is Good



If Each stage take 10ns, the latency will be 50 ns

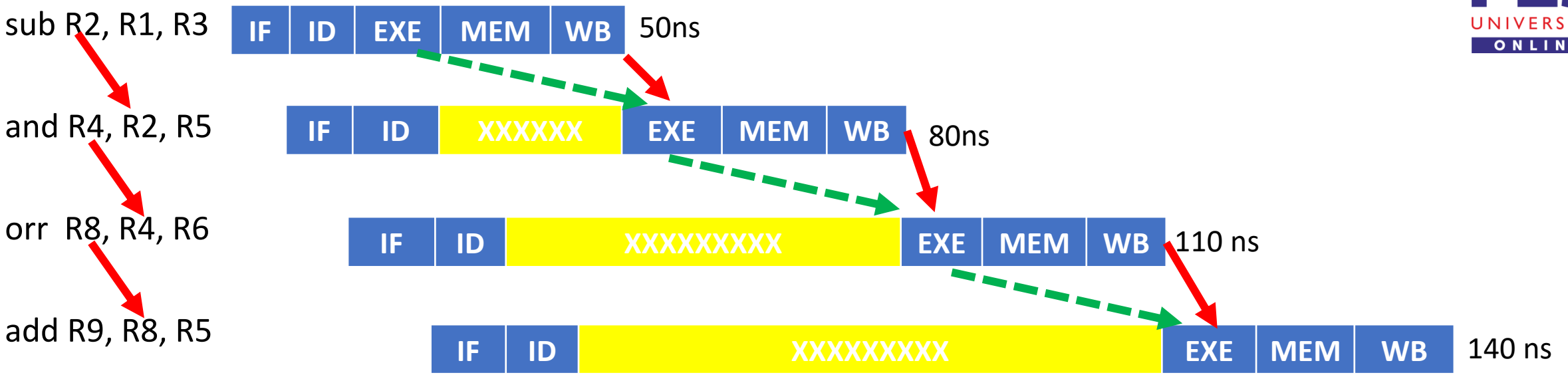
If Each stage take 10ns, the throughput is 80 ns instead of 200 ns

Pipelining, What May Go Wrong?

- Is Good, When every instruction is independent of the other and can execute in the pipeline without any constraint
- But in application programs, most of the statements are dependent on each other
  - C = A + B;
  - E = C * D;
- This causes constraint on the pipeline to execute these dependent instructions in given 5 cycles.
  - Increases Latency

Microprocessor & Computer Architecture (µpCA)

Pipelining, What May Go Wrong?



If Each stage take 10ns, the latency will be 50 ns

If Each stage take 10ns, the throughput is 80 ns instead of 200 ns

With gap between stages, the throughput is 140 ns instead of 80ns

Pipelining, What May Go Wrong?→ Hazards

- Situations that prevent the next instruction in the instruction stream from executing during its designated cycle
- Condition that prevents an instruction in the pipe from executing its next scheduled pipe stage
- Where one instruction cannot **immediately** follow another

- **Hazards reduce the performance from the ideal speedup gained by pipelining**

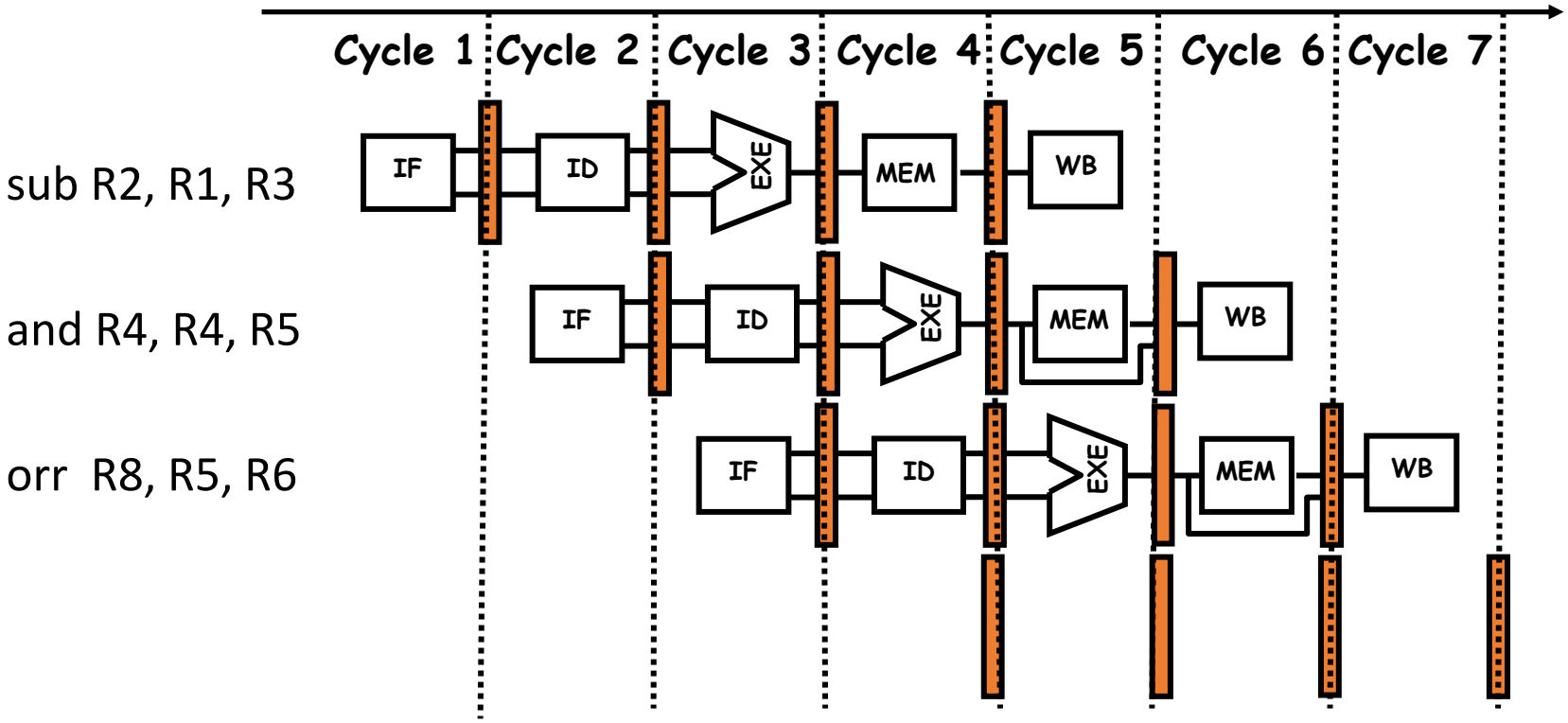
Hazards: Limits to Pipelining

- ***Structural Hazard:***
  - Due to structure of the pipeline
  - Hardware cannot support combination of instructions in the pipeline needing the same resources
- ***Data Hazard:***
  - Due to data dependencies between instructions
  - Instruction depends on the result of prior instruction still in the pipeline
- ***Control Hazard:***
  - Due to control instructions
  - Pipelining of branches & other instructions that change the PC

Stalls

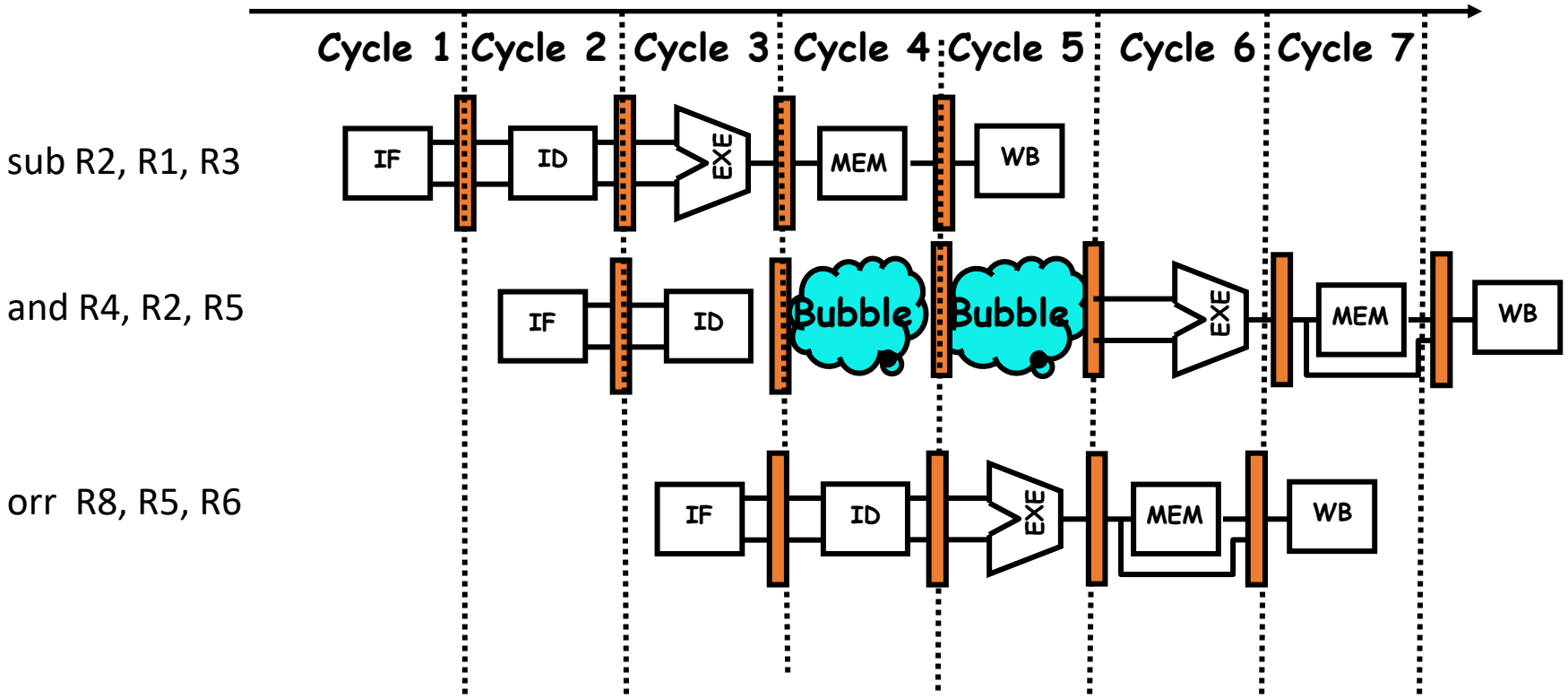
- Common solution for any type of hazard is to **stall** the pipeline until the hazard is resolved (Stall Cycles)
- This is achieved by inserting one or more **"bubbles"** in the pipeline
- To do this, hardware or software must **detect** that a hazard has occurred (Hazard Detection)

Stalls


Cycle 1
Cycle 2
Cycle 3
Cycle 4
Cycle 5
Cycle 6
Cycle 7
sub R2, R1, R3
and R4, R4, R5
orr R8, R5, R6
IF
ID
EXE
MEM
WB

Microprocessor & Computer Architecture (µpCA)

Stalls



Stalls

Instruction Number	Clock number									
	1	2	3	4	5	6	7	8	9	10
sub R2, R1, R3	IF	ID	EX	MEM	WB					
and R4, R2, R5		IF	ID	**stall**	**stall**	EX	MEM	WB		
orr R8, R5, R6			IF	ID	EX	MEM	WB			

Disadvantage

- Stall cycle is a waste cycle
  - Opportunity wasted to execute one instruction
- Adds to pipeline delay and hence increases CPI
  - (CPI > 1)
  - Ex: If 30% are load/store, with only one memory; CPI ≥ 1.3
- Not a preferred solution
- Affects performance of pipeline architecture

Microprocessor & Computer Architecture (µpCA)

Performance of Pipelines with Stalls

A stall causes the pipeline performance to degrade the ideal performance.

$$\text{Speedup from pipelining} = \frac{\text{Average instruction time unpipelined}}{\text{Average instruction time pipelined}}$$

$$= \frac{\text{CPI}_{\text{unpipelined}} * \text{Clock Cycle Time}_{\text{unpipelined}}}{\text{CPI}_{\text{pipelined}} * \text{Clock Cycle Time}_{\text{pipelined}}}$$

$$= \frac{\text{CPI}_{\text{unpipelined}}}{1 + \text{Pipeline stall cycles per instruction}}$$

Note 1: Ignoring the cycle time overhead of pipelining and assume the stages are all perfectly balanced, then the cycle time of the two machines are equal

Note 2: Ideal CPI of pipeline processor is almost always =1

$\text{CPI}_{\text{pipelined}}$ = Ideal CPI + Pipeline stall clock cycles per instruction

= 1 + Pipeline stall clock cycles per instruction

Performance of Pipelines with Stalls

If all instructions take the same number of cycles, which must also equal the number of pipeline stages (the depth of the pipeline) then unpipelined CPI is equal to the depth of the pipeline

$$\text{Speedup} = \frac{\text{Pipeline depth}}{1 + \text{Pipeline stall cycles per instruction}}$$

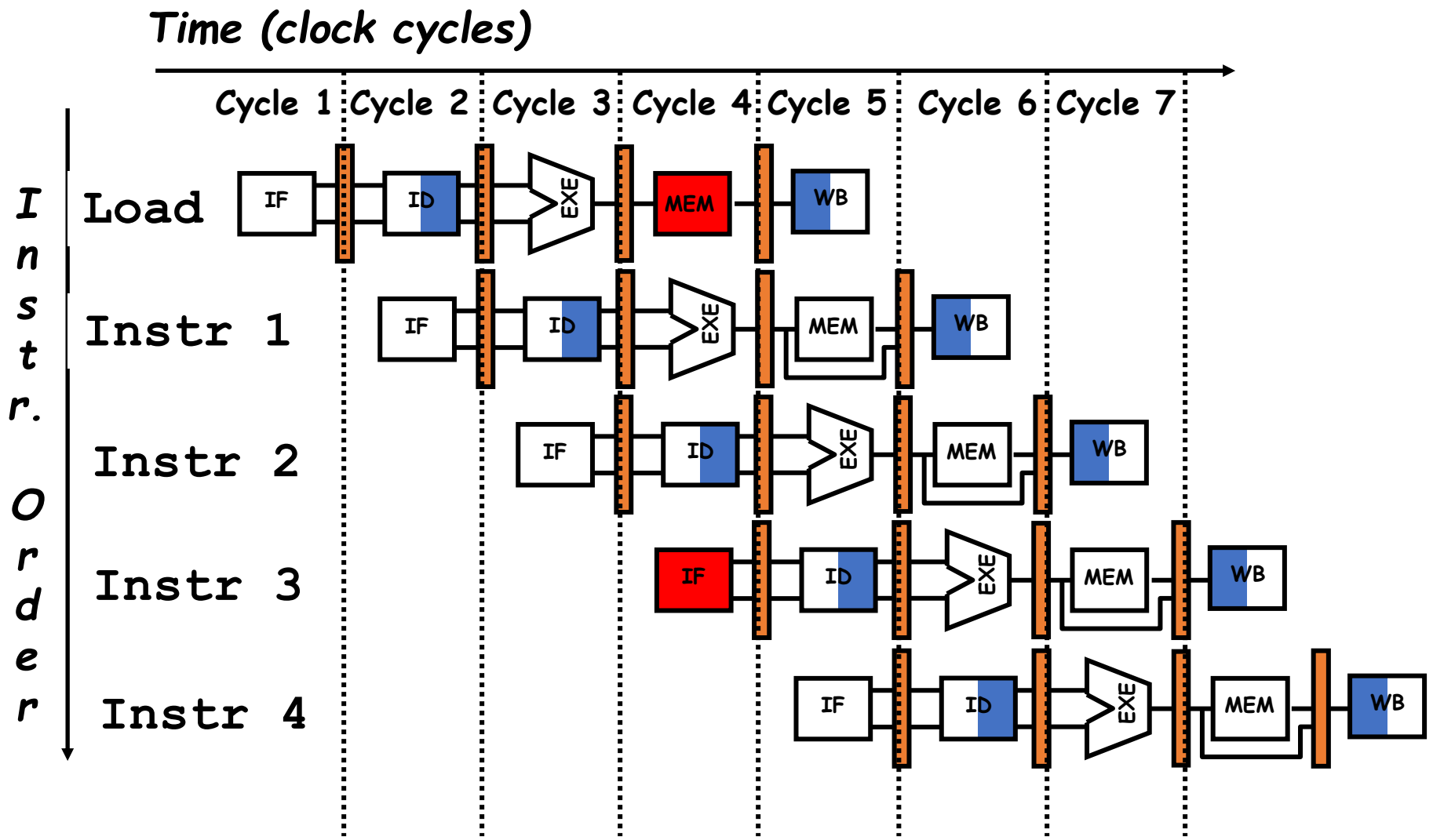
If there are no pipeline stalls, this leads to the intuitive result that pipelining can improve performance by the depth of pipeline.

Structural Hazard

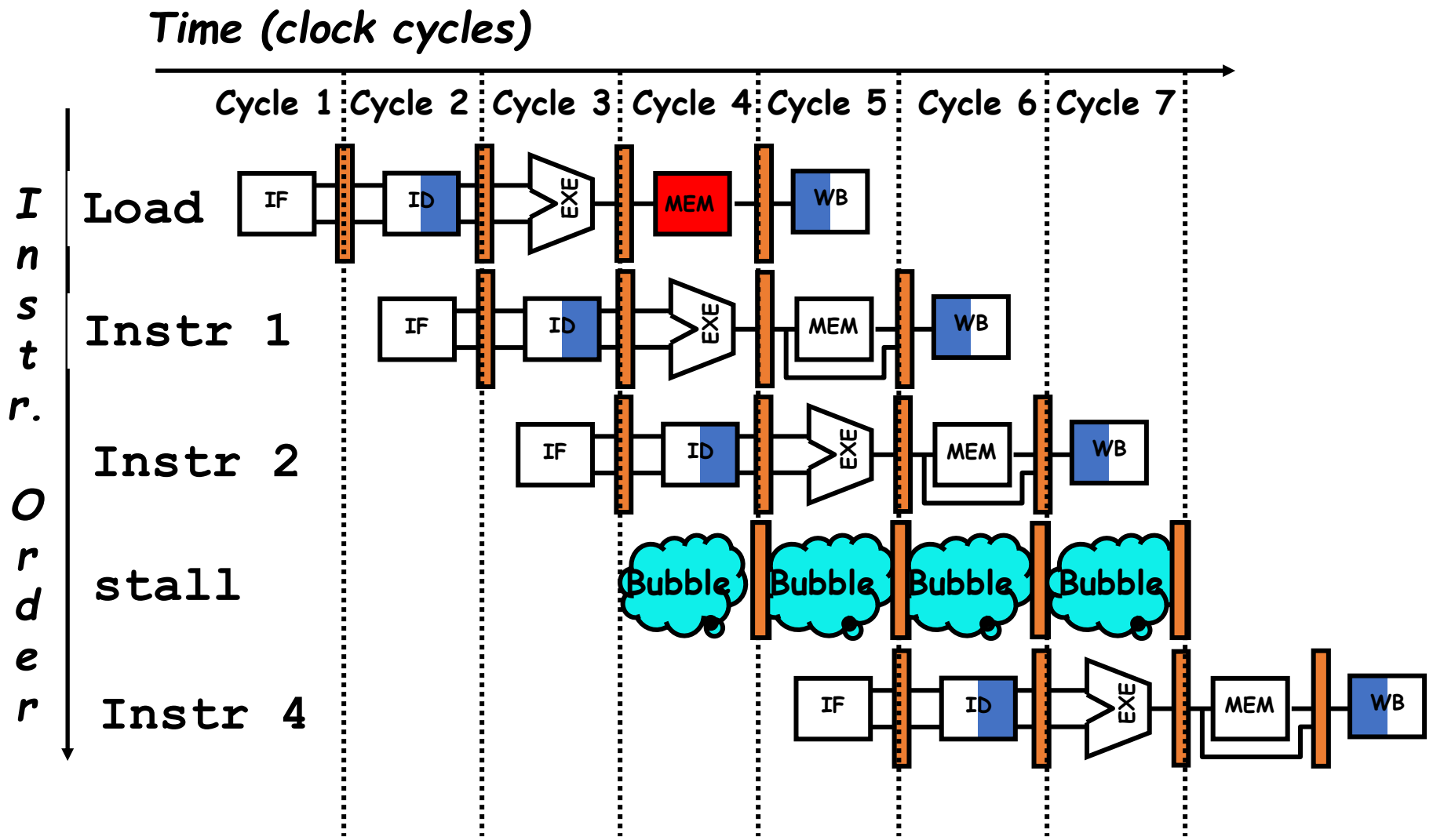
- If a resource conflict arises due to a hardware resource being required by more than one instruction in a single cycle, and one or more such instructions cannot be accommodated, then a structural hazard occurs, for example:
  - **when a pipelined machine has a shared single-memory for data and instructions.**
    - **Unified Memory**
  - **when a machine has only one register file write port**
    - **Skip MEM stage for add/sub..**
  - **when a machine has overlapping of read and write**
    - **Overlap ID & WB**

Microprocessor & Computer Architecture (µpCA)

Structural Hazard: IF VS MEM:



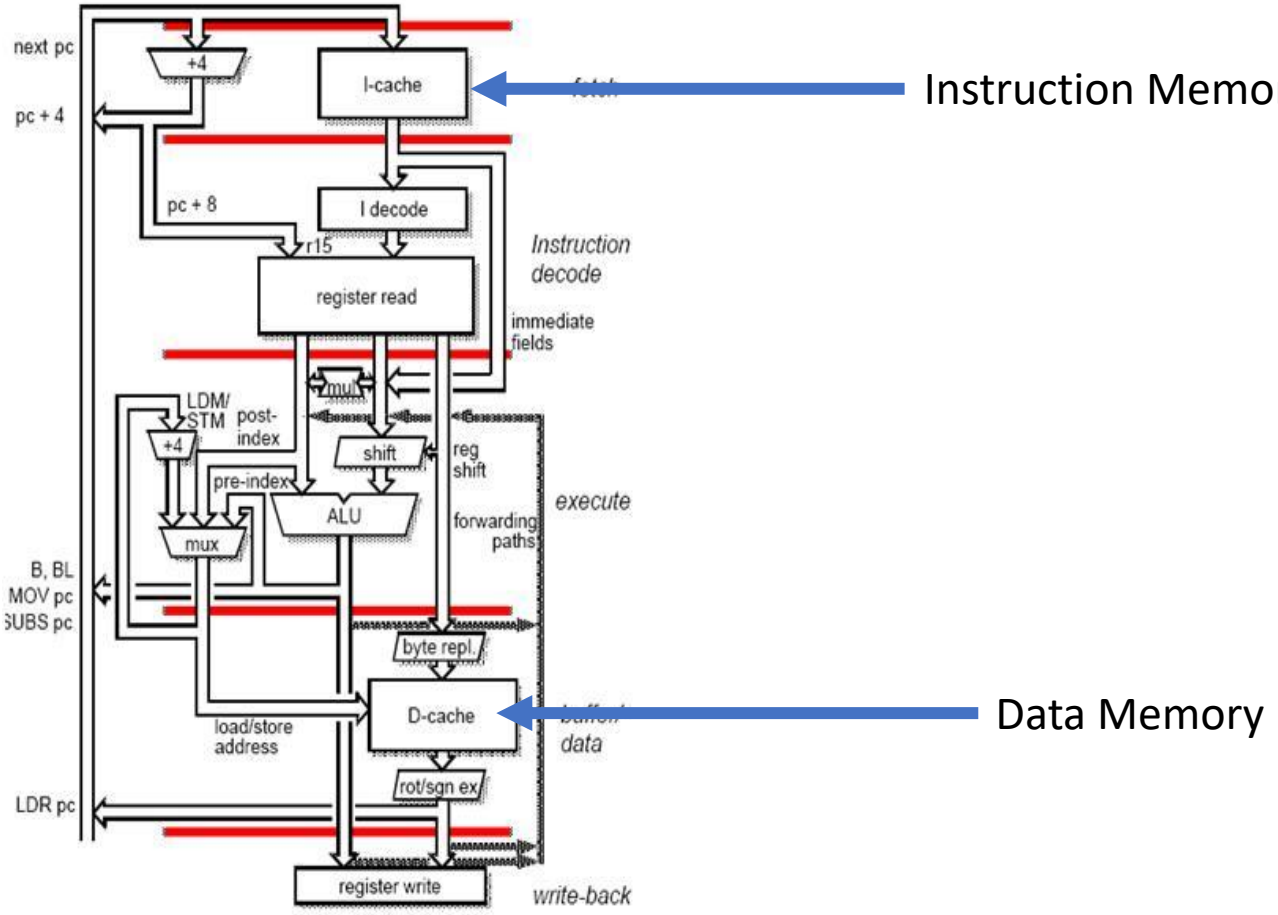
Structural Hazard: IF VS MEM:



Microprocessor & Computer Architecture (µpCA)

Structural Hazard: IF VS MEM: Solution

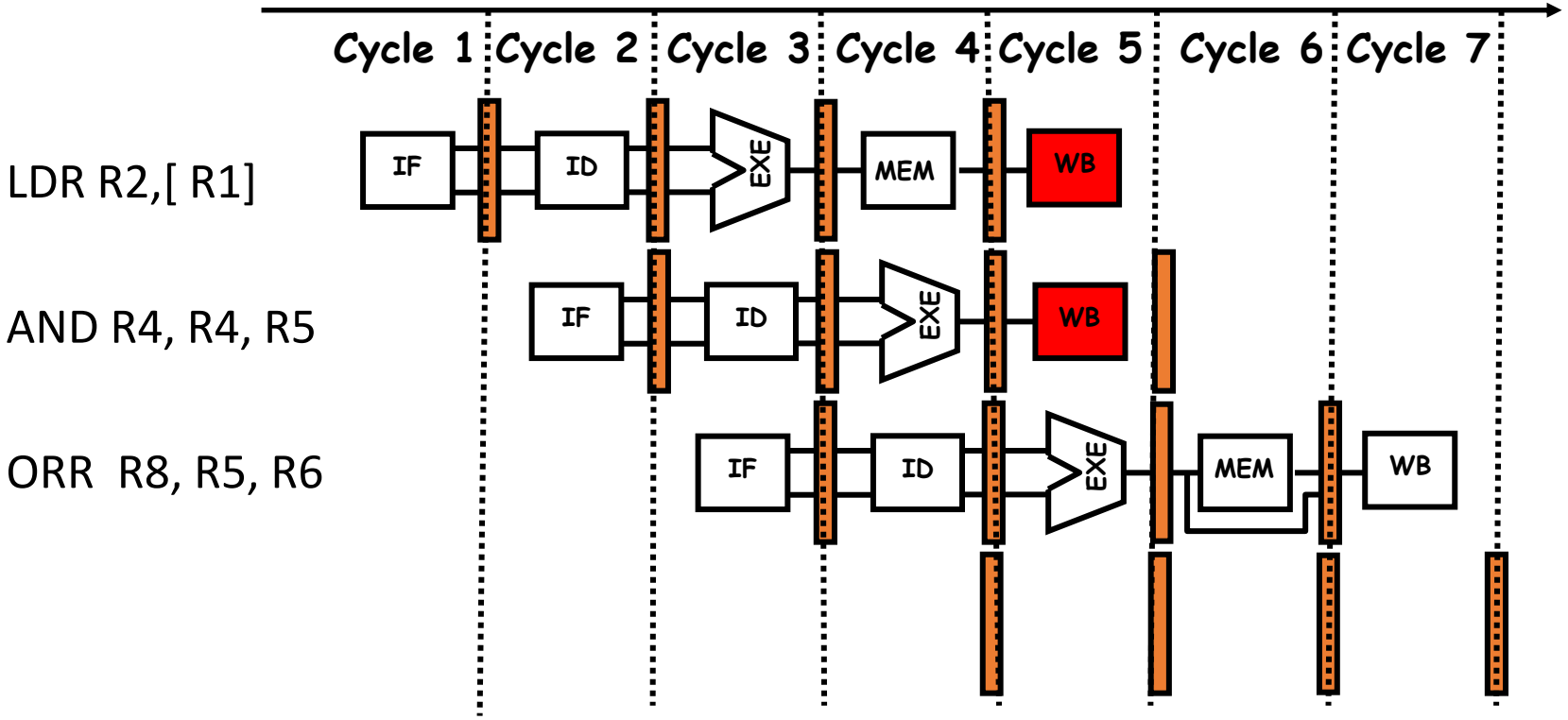
- **Replication of resources**
  - Split Memory (Instruction Memory & Data Memory)



Instruction Memory

Data Memory

Structural Hazard: WB Vs WB:

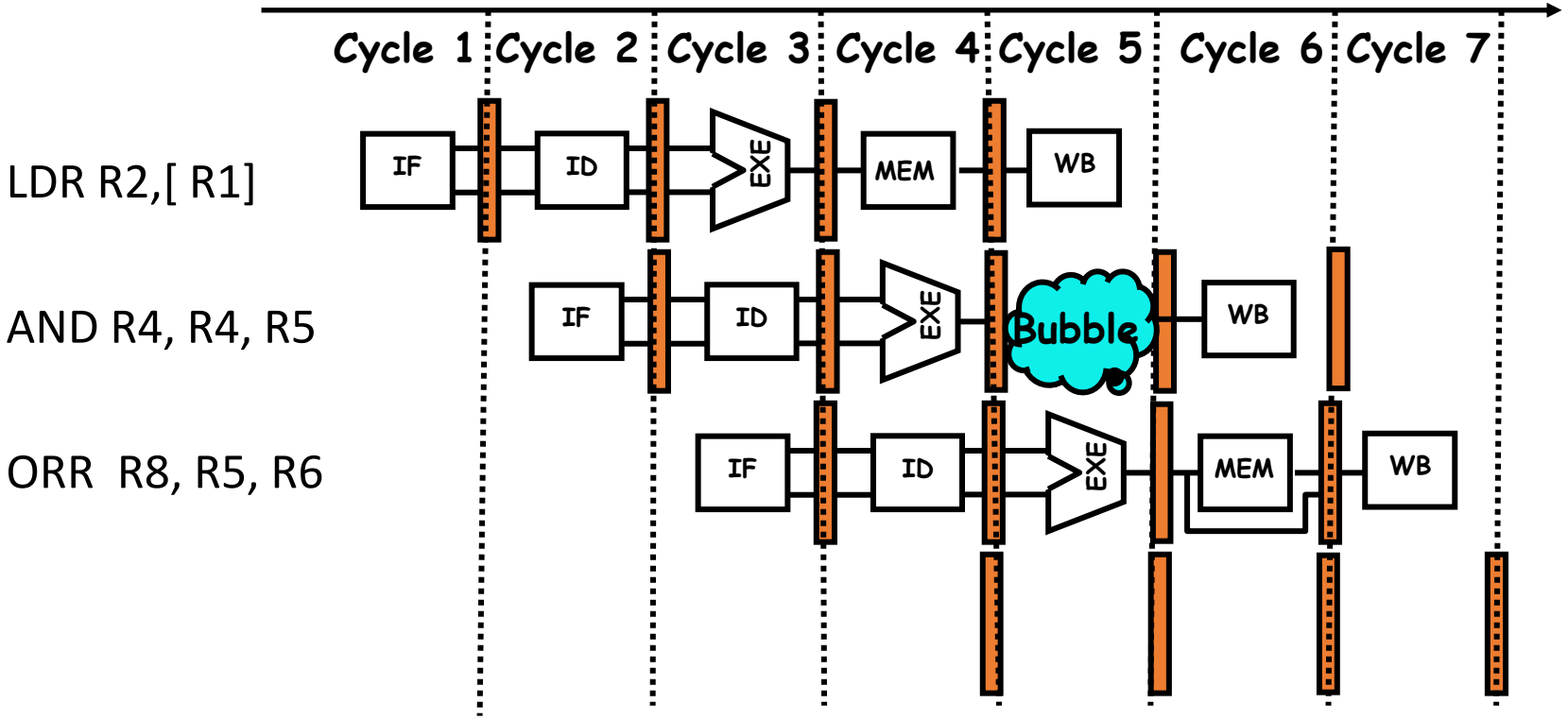


2 ways to solve this pipeline hazard.

- **Stall the stage**
- **Let all the instruction follow all stages, AND may take longer than usual**

Microprocessor & Computer Architecture (µpCA)

Structural Hazard: WB Vs WB: → Solution 1

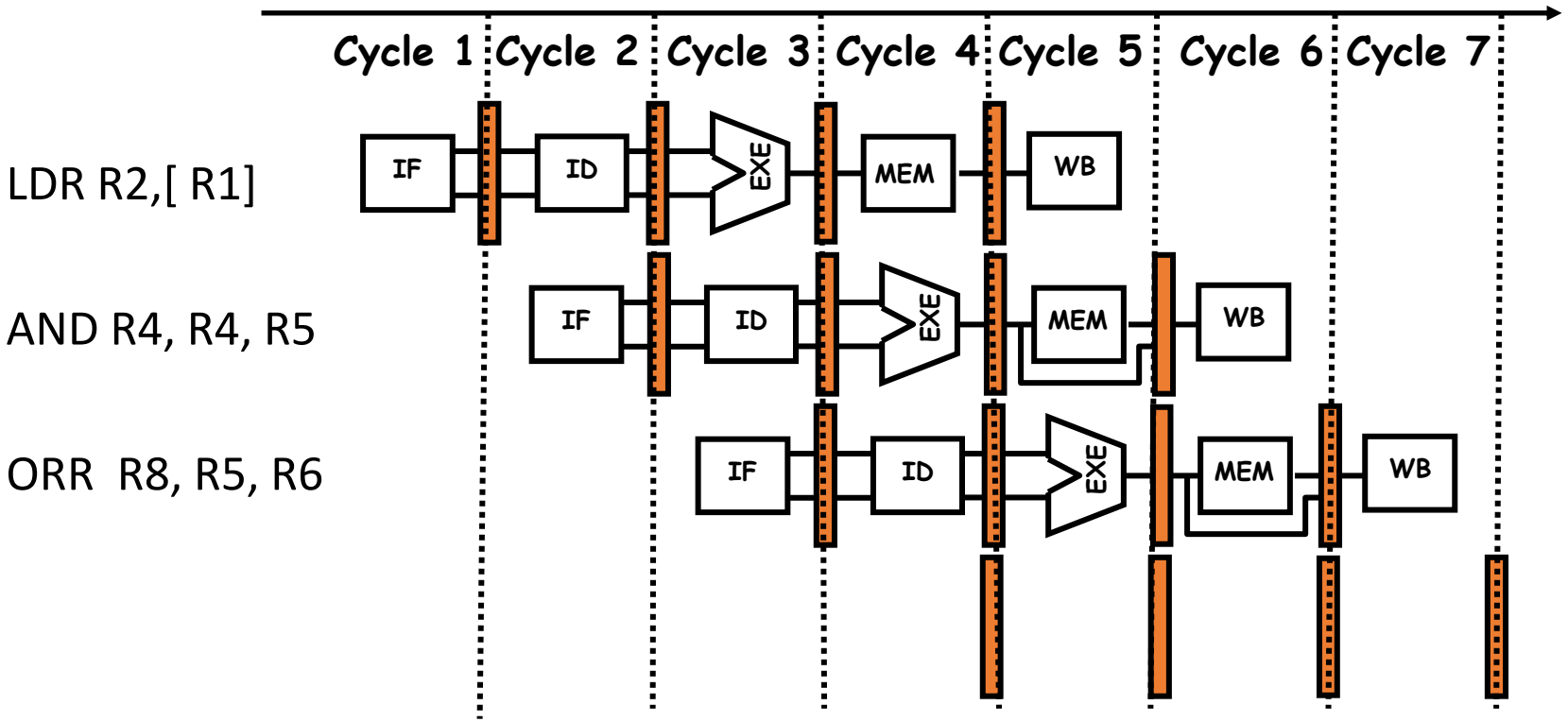


2 ways to solve this pipeline hazard.

- **Stall the stage**
- **Let all the instruction follow all stages, AND may take longer than usual**

Microprocessor & Computer Architecture (µpCA)

Structural Hazard: WB Vs WB: → Solution 2

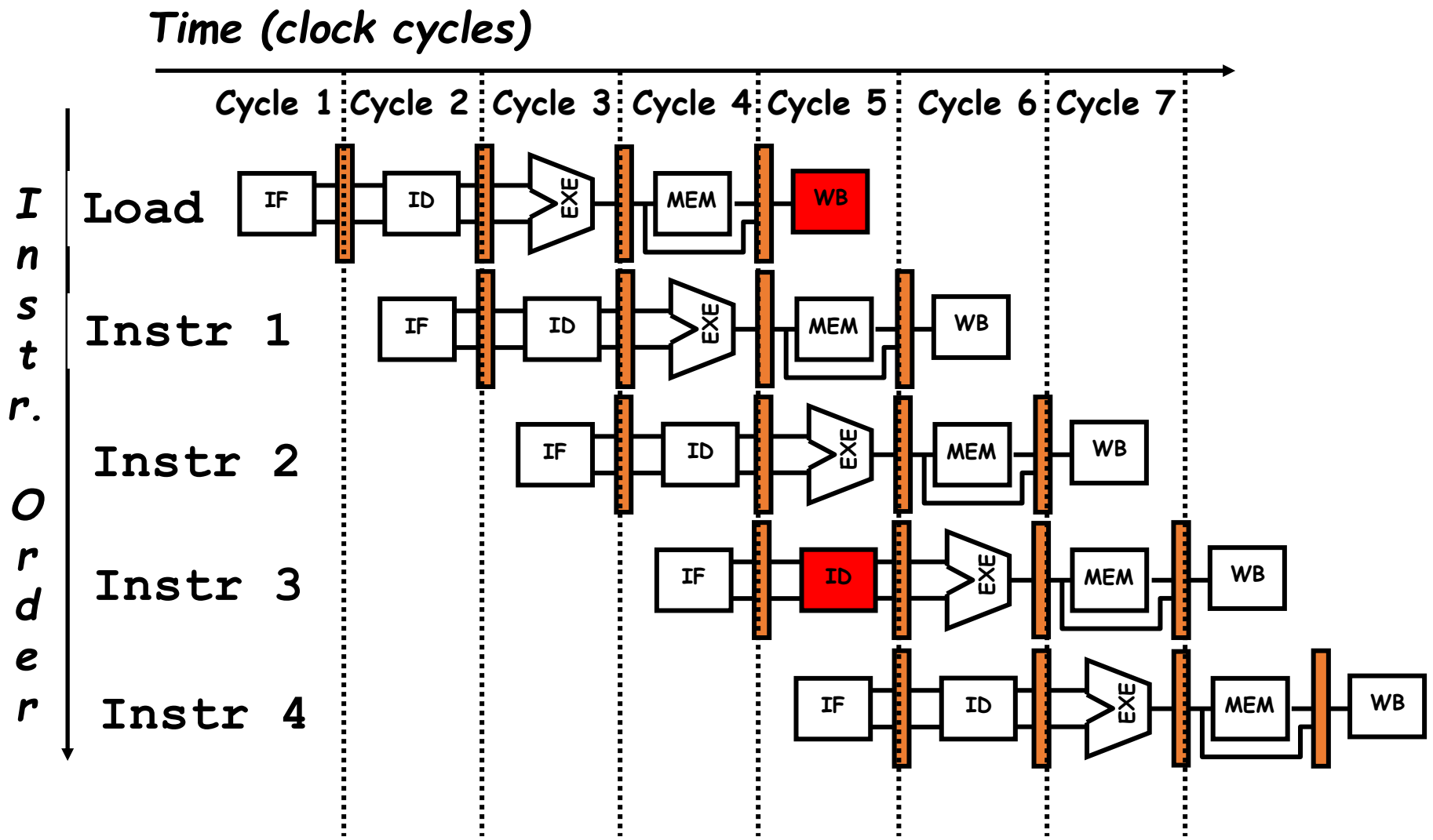


2 ways to solve this pipeline hazard.

- **Stall the stage**
- **Let all the instruction follow all stages, AND may take longer than usual**

Microprocessor & Computer Architecture (µpCA)

Structural Hazard: ID VS WB



Structural Hazard: ID VS WB

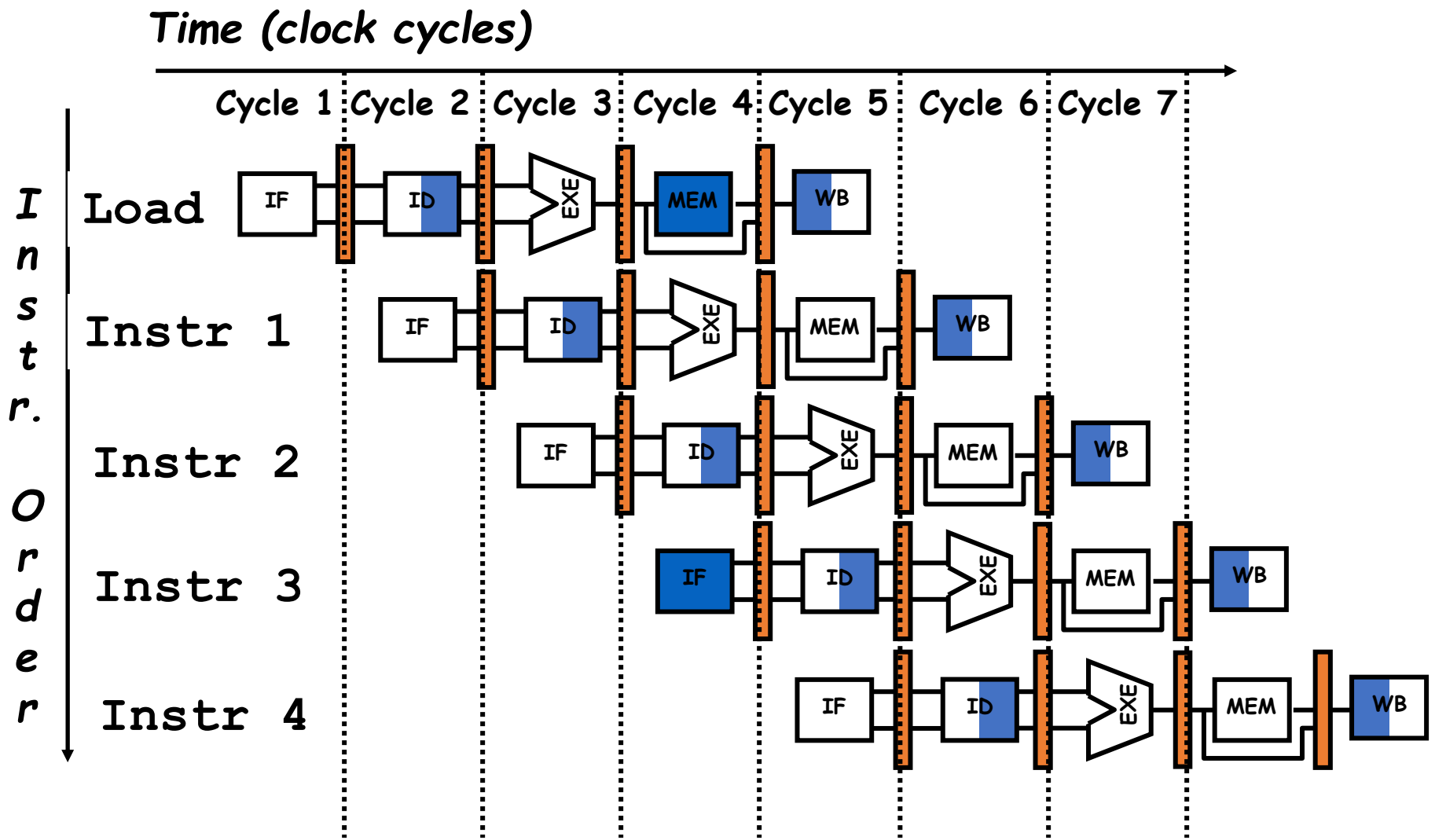
Partitioning/Overlapping

1st half of the cycle *Reg Write*

2nd half of the same cycle *Reg Read*

Microprocessor & Computer Architecture (µpCA)

Partitioning/Overlapping



Performance (with stalls)

- **n >> k; Speedup = k; no. of stages (also known as Depth of Pipeline)**
- *For 5-stage pipeline with stalls:*
- $Speed\ up = \frac{Depth\ of\ Pipeline}{CPI}$ = 5
- $Speed\ up = \frac{Depth\ of\ Pipeline}{1 + Pipeline\ Stall\ cycles\ per\ instruction}$
- If stalls are 0, then Speedup = k = 5
- If say, 30% are Load/Store in a unified memory pipeline, then every load/store overlap with IF will introduce 3 stalls. Thus: CPI = 1+(3x0.3) = 1.9
- $Speed\ up = \frac{5}{1 + (3x0.3)}$ = 2.63

Data Hazards

THANK YOU

Dr. D. C. Kiran
Department of Computer Science and Engineering

dckiran@pes.edu

9829935135