CS 222 Computer Organization & Architecture

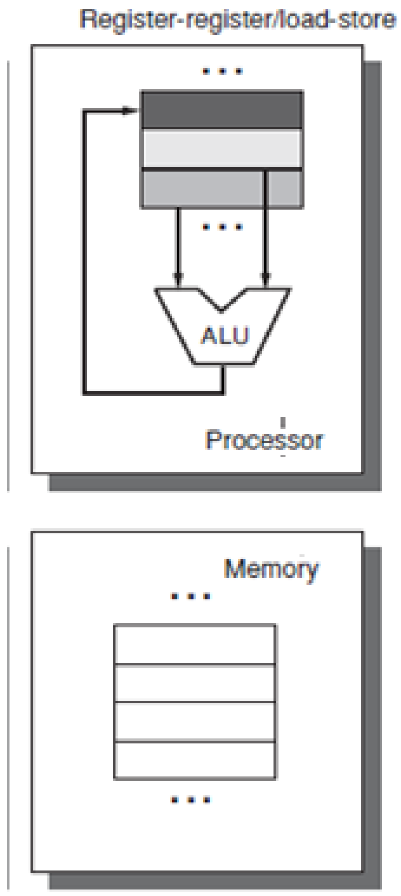
Lecture 22 [12.03.2019]

Introduction to RISC instruction pipeline

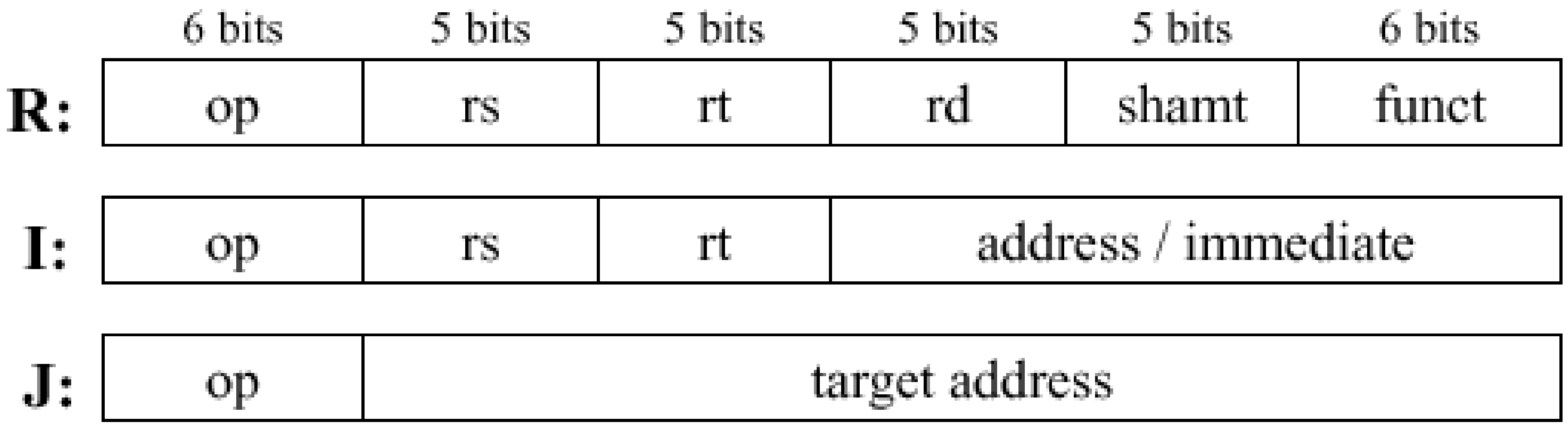
John Jose
Assistant Professor
Department of Computer Science & Engineering
Indian Institute of Technology Guwahati, Assam.

Introduction to MIPS

- Microprocessor without Interlocked Pipelined Stages
- 32 registers (32 bit each)
- Uniform length instructions
- RISC- Load store architecture



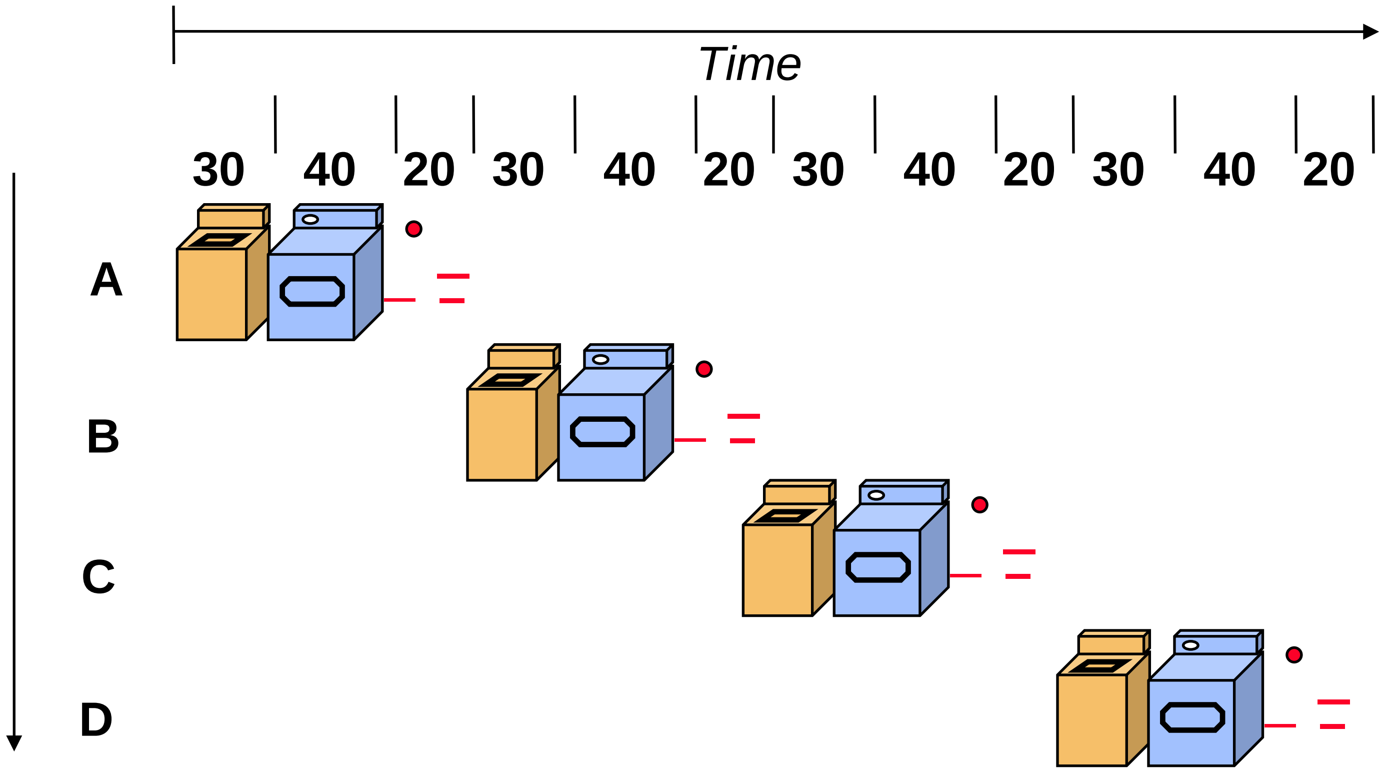
Introduction to MIPS



op: basic operation of the instruction (opcode)
rs: first source operand register
rt: second source operand register
rd: destination operand register
shamt: shift amount
funct: selects the specific variant of the opcode (function code)
address: offset for load/store instructions (+/-2^{15})
immediate: constants for immediate instructions

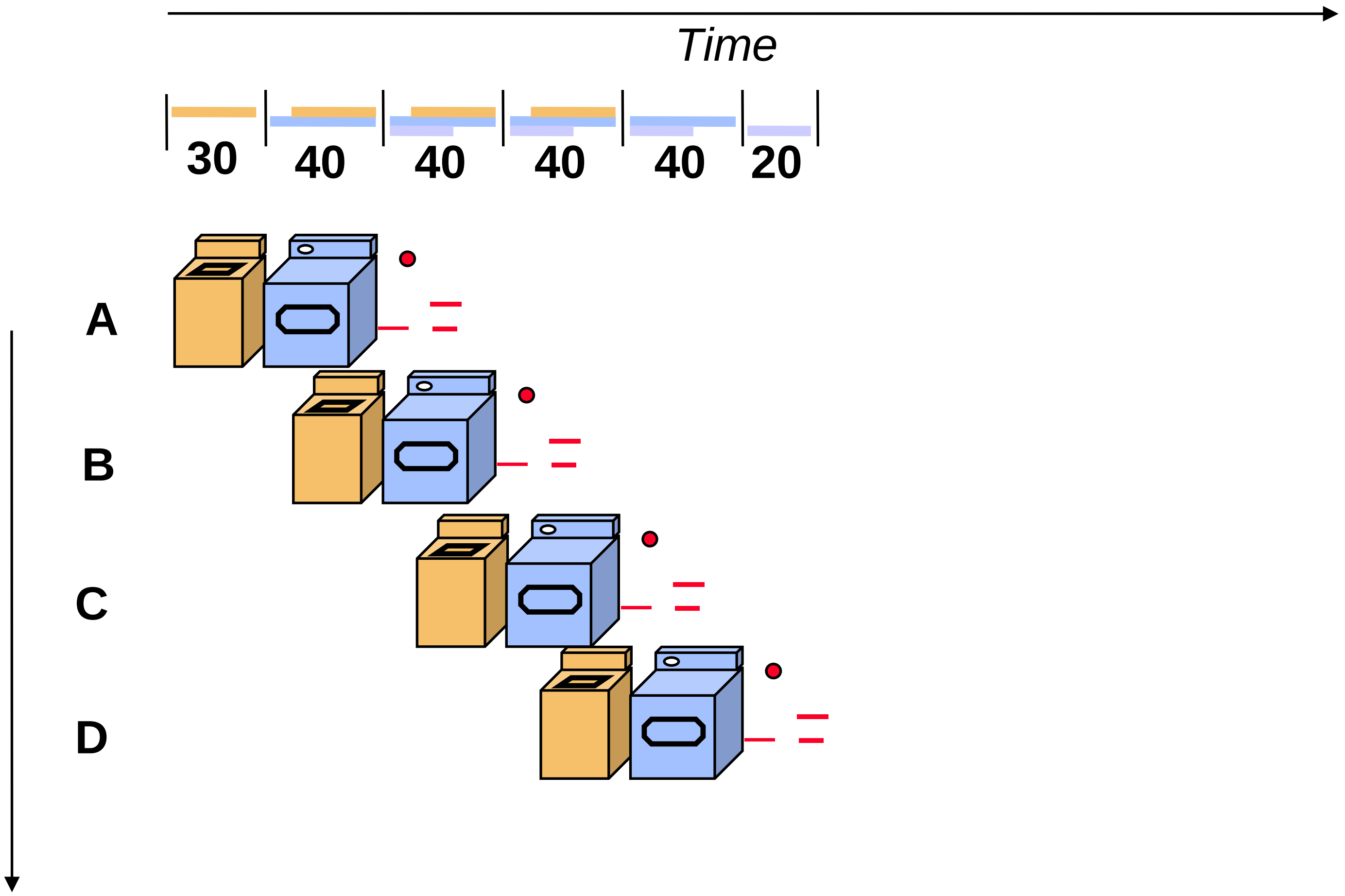
Unpipelined Work flow

- Start work when previous one is fully over
- Sequential laundry takes 6 hours for 4 loads



Pipelined Work flow

- ❖Start work as soon as possible
- ❖Pipelined laundry takes 3.5 hours for 4 loads

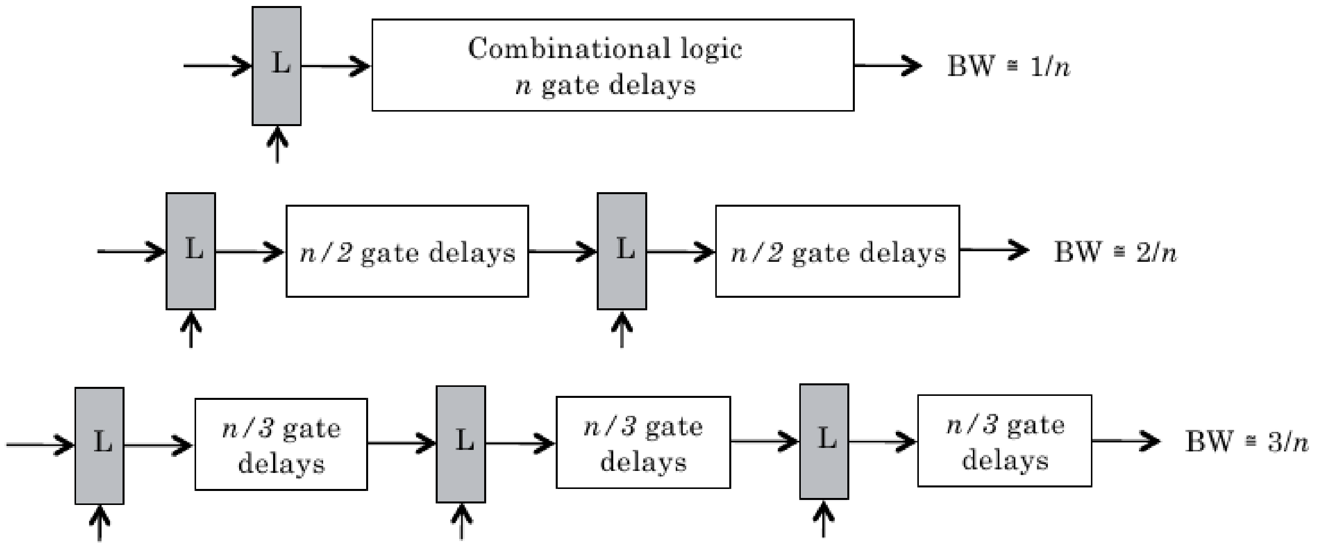


Pipelining Characteristics

- ❖Pipelining doesn’t reduce latency of single task, it improves throughput of entire workload
- ❖Pipeline rate limited by slowest pipeline stage
- ❖Potential speedup = Number of pipe stages
- ❖Unbalanced lengths of pipe stages reduces speedup
- ❖Time to “fill” pipeline and time to “drain” it reduces speedup

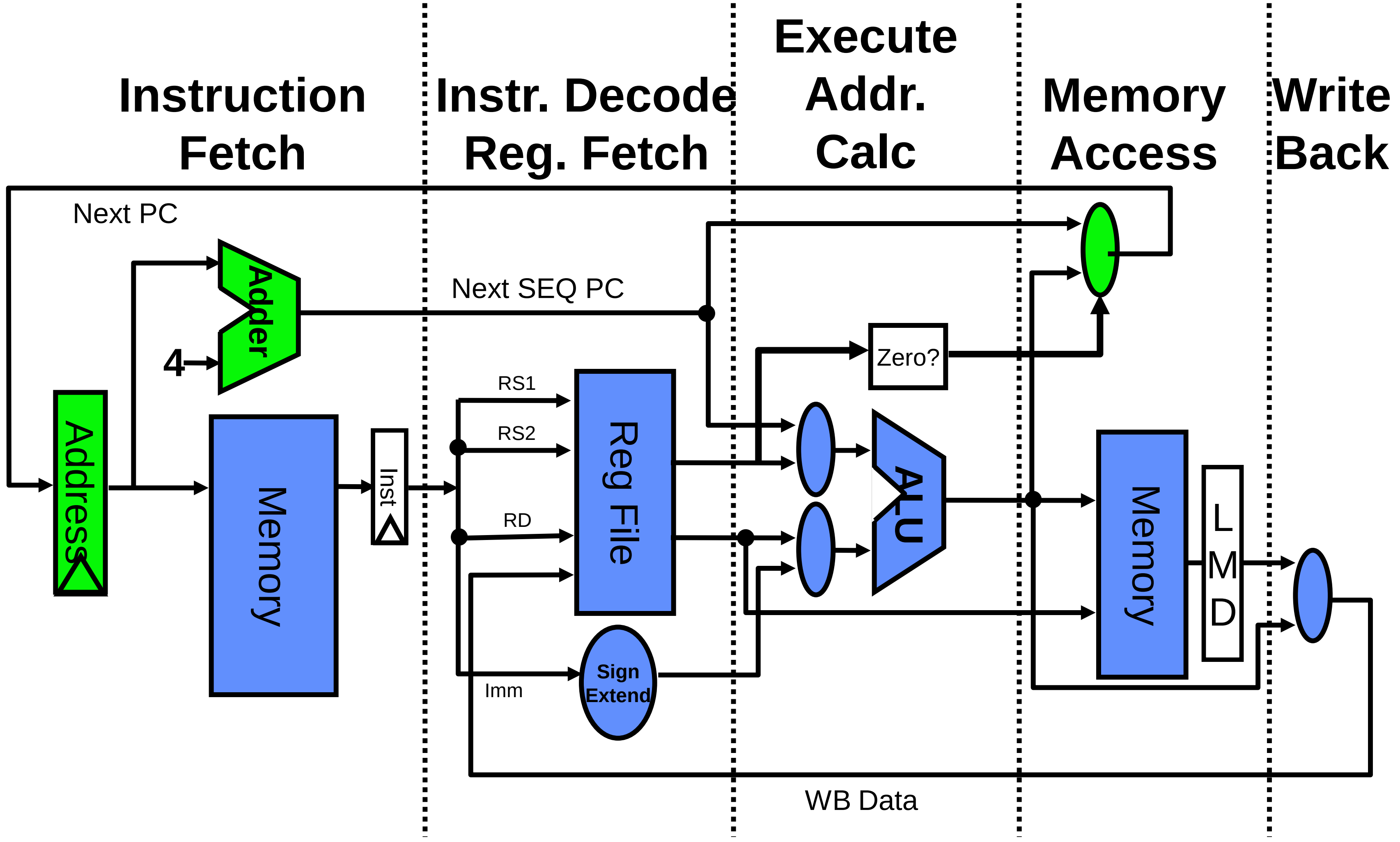
Pipelining in Circuits

- ❖ Pipelining partitions the system into multiple independent stages with added buffers between the stages.
- ❖ Pipelining can increase the throughout of a system.

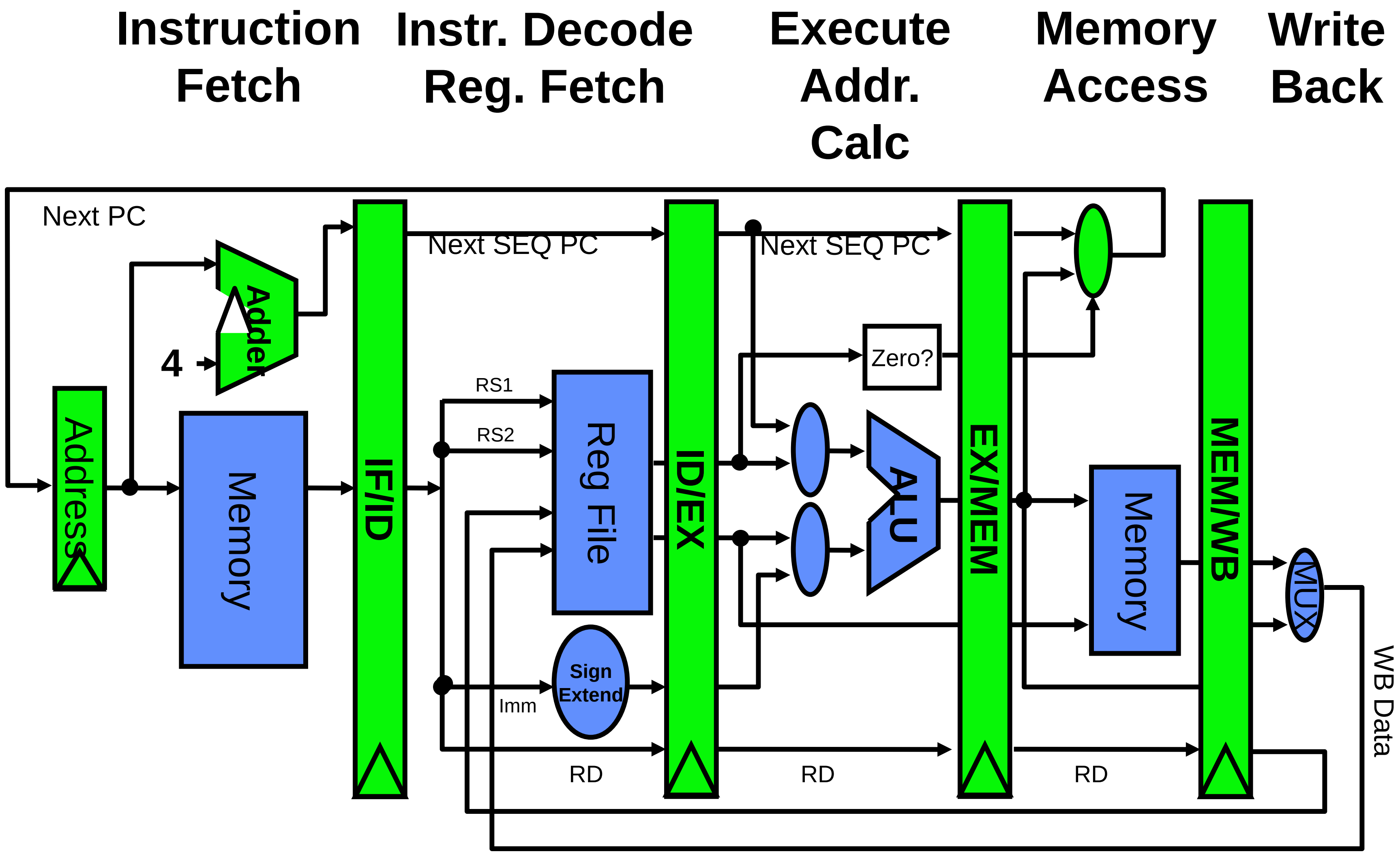


Potential k-fold increase of throughput in a k-stage pipelined system

Unpipelined RISC Data path

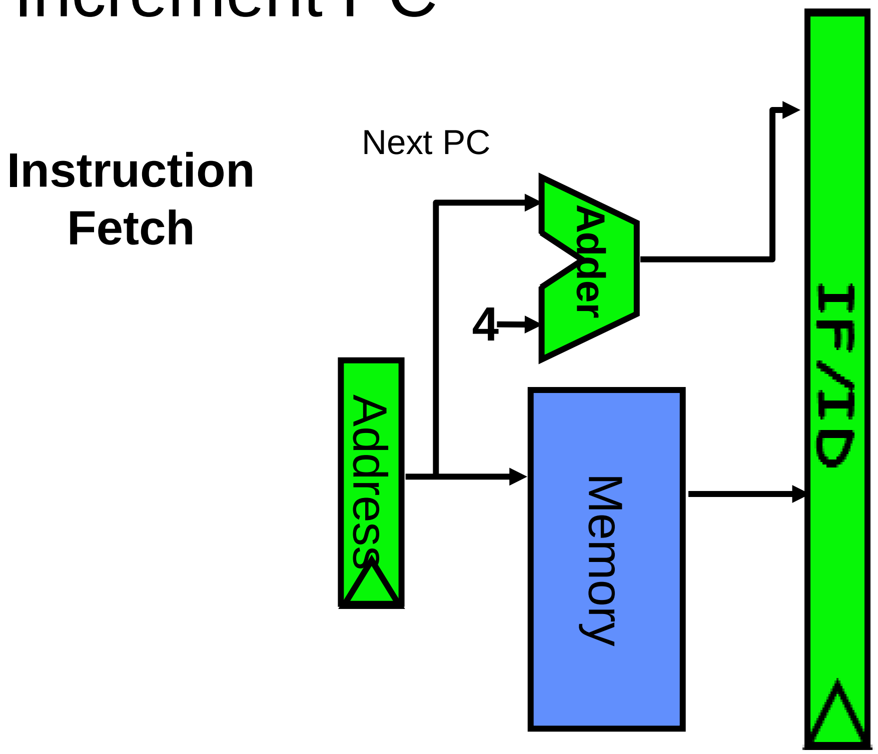


Pipelined RISC Data path



RISC MIPS Instruction Pipeline

- Each instruction can take at most 5 clock cycles
- **Instruction fetch cycle (IF)**
  - Based on PC, fetch the instruction from memory
  - Increment PC



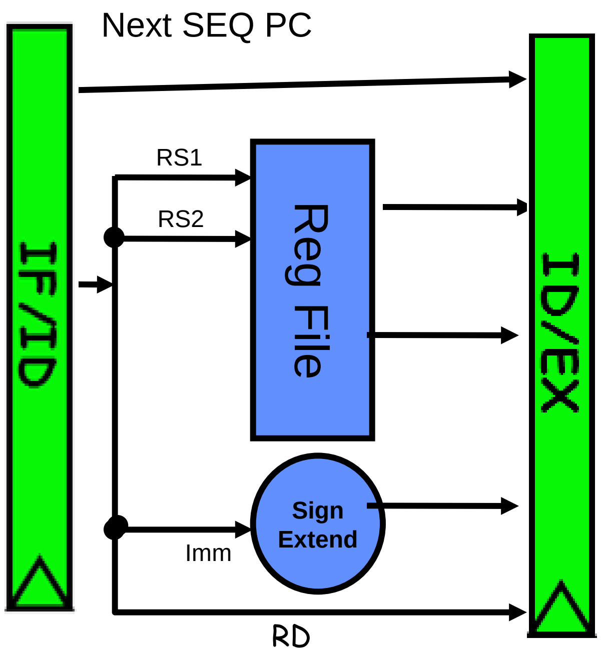
RISC MIPS Instruction Pipeline

- **Instruction decode/register fetch cycle (ID)**
- Decode the instruction + register read operation
- Fixed field decoding **[ADD R1,R2,R3] OR [LW R1,8(R2)]**

Ex: A3.01.02.03 : 10100011 00000001 00000010 00000011

Ex: 86.01.02.03 : 10000110 00000001 00001000 00000010

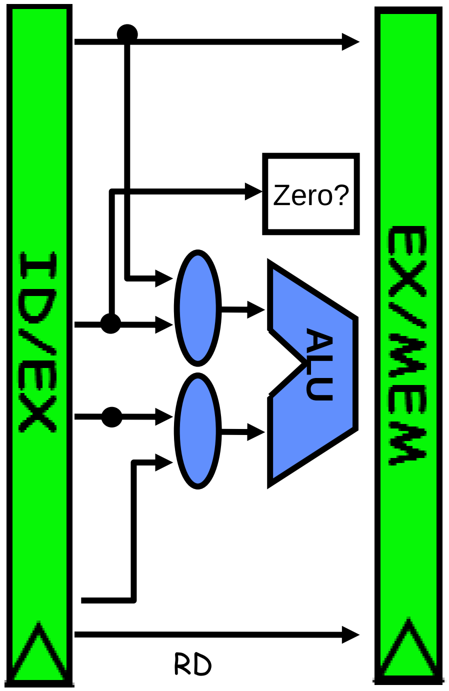
Instr. Decode
Reg. Fetch



RISC MIPS Instruction Pipeline

- **Execution/Effective address cycle (EX)**
- Memory reference: Calculate the effective address
- **[LW R1,8(R2)]** **EFF ADDR= [R2] +8**
- Register-register ALU instruction **[ADD R1,R2,R2]**

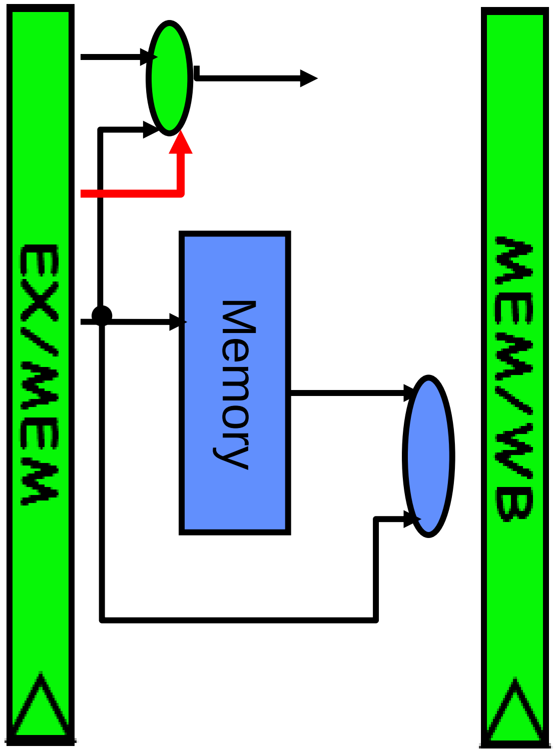
Execute Addr. Calc



RISC MIPS Instruction Pipeline

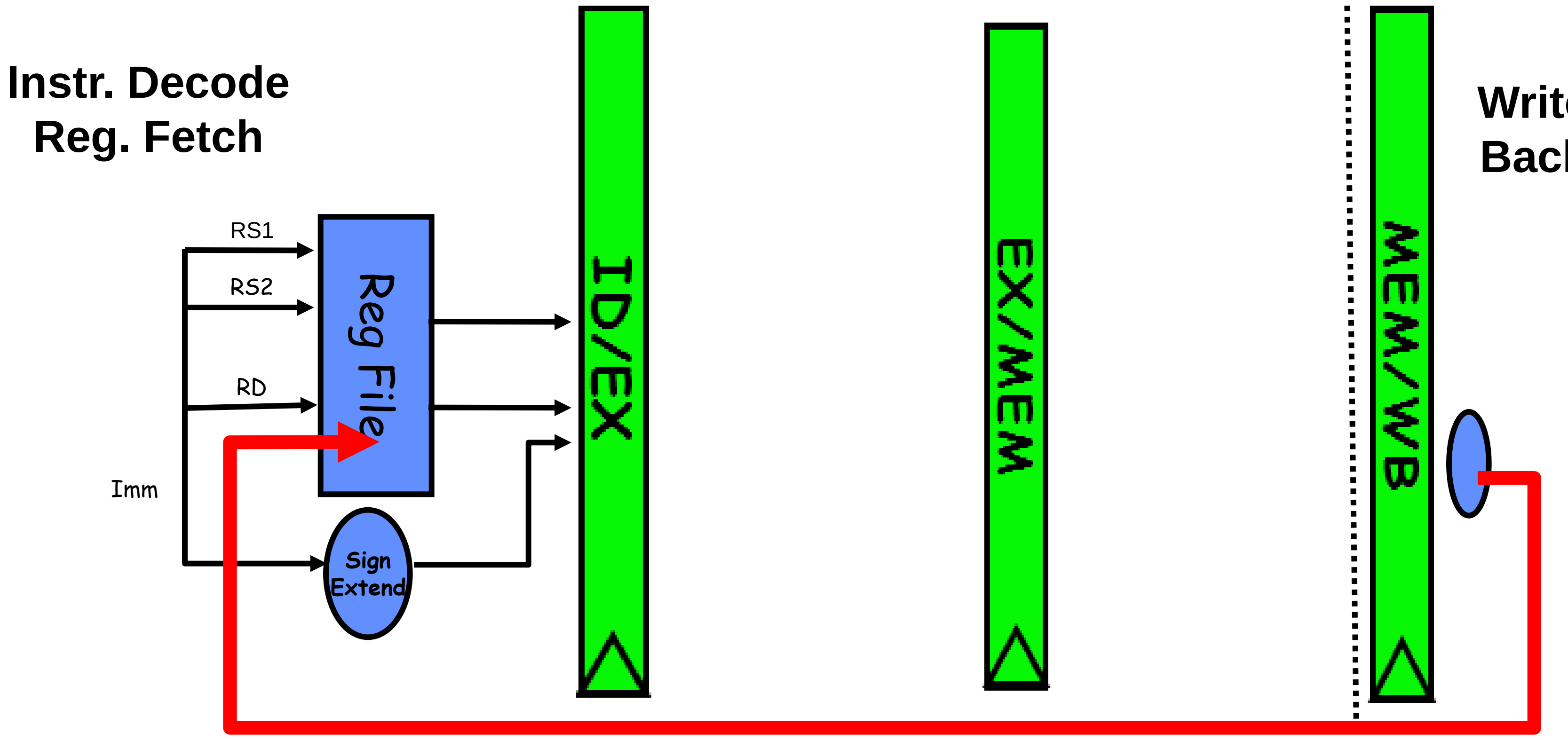
- **Memory access cycle (MEM)**
  - Load from memory and store in register **[LW R1,8(R2)]**
  - Store the data from the register to memory**[SW R3,16(R4)]**

Memory Access

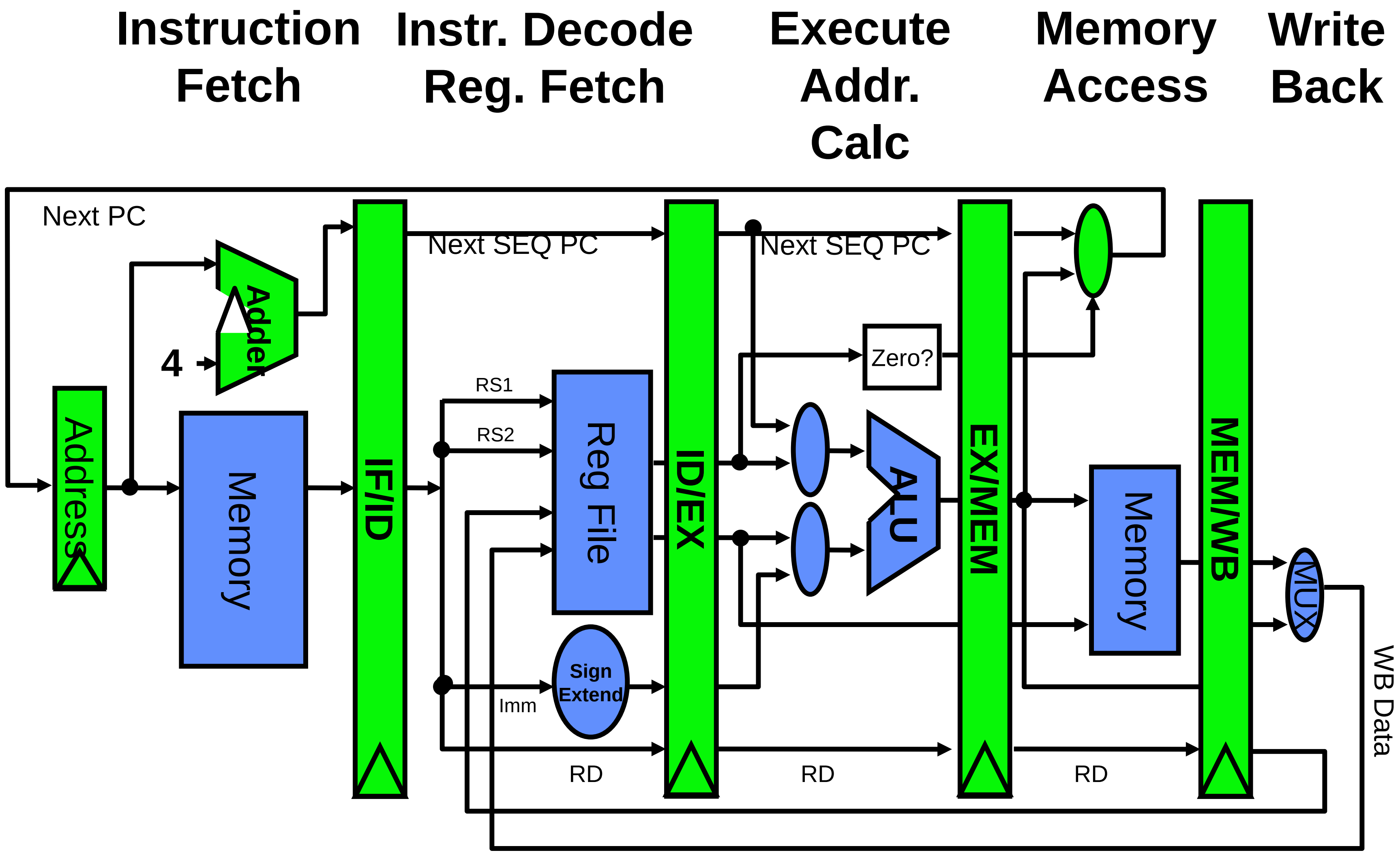


RISC Instruction Pipeline

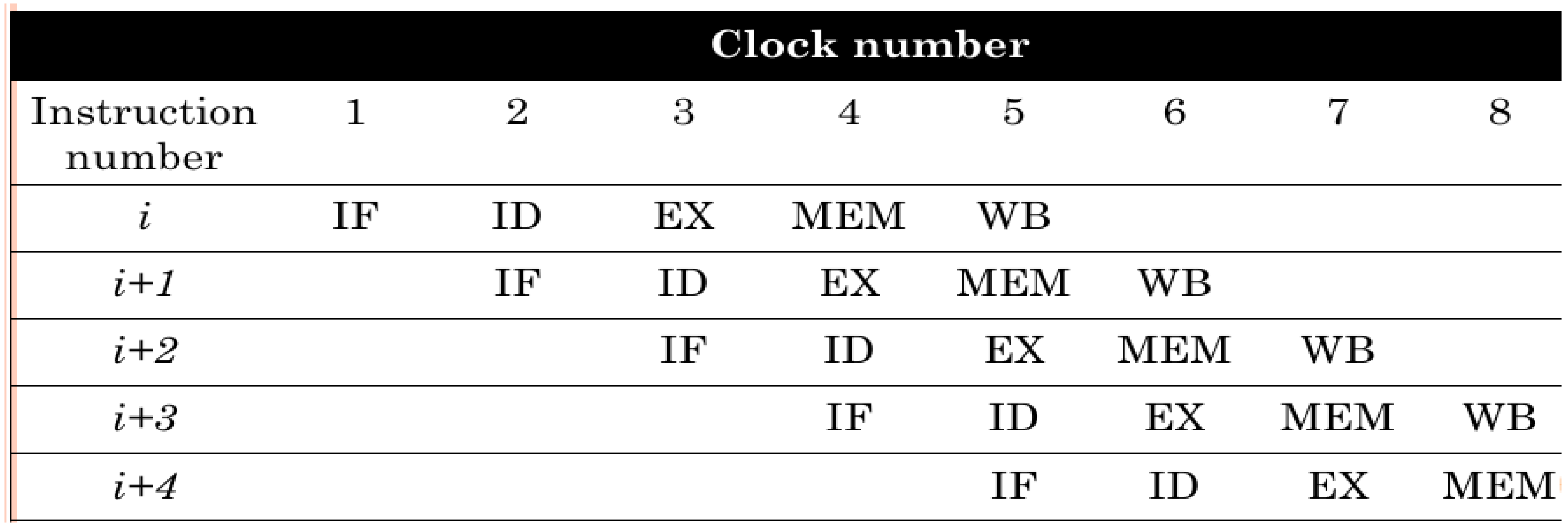
- **Write-back cycle (WB)**
  - Register-register ALU instruction or load instruction
  - Write to register file **[LW R1,8(R2)] , [ADD R1,R2,R3]**



Pipelined RISC Data path

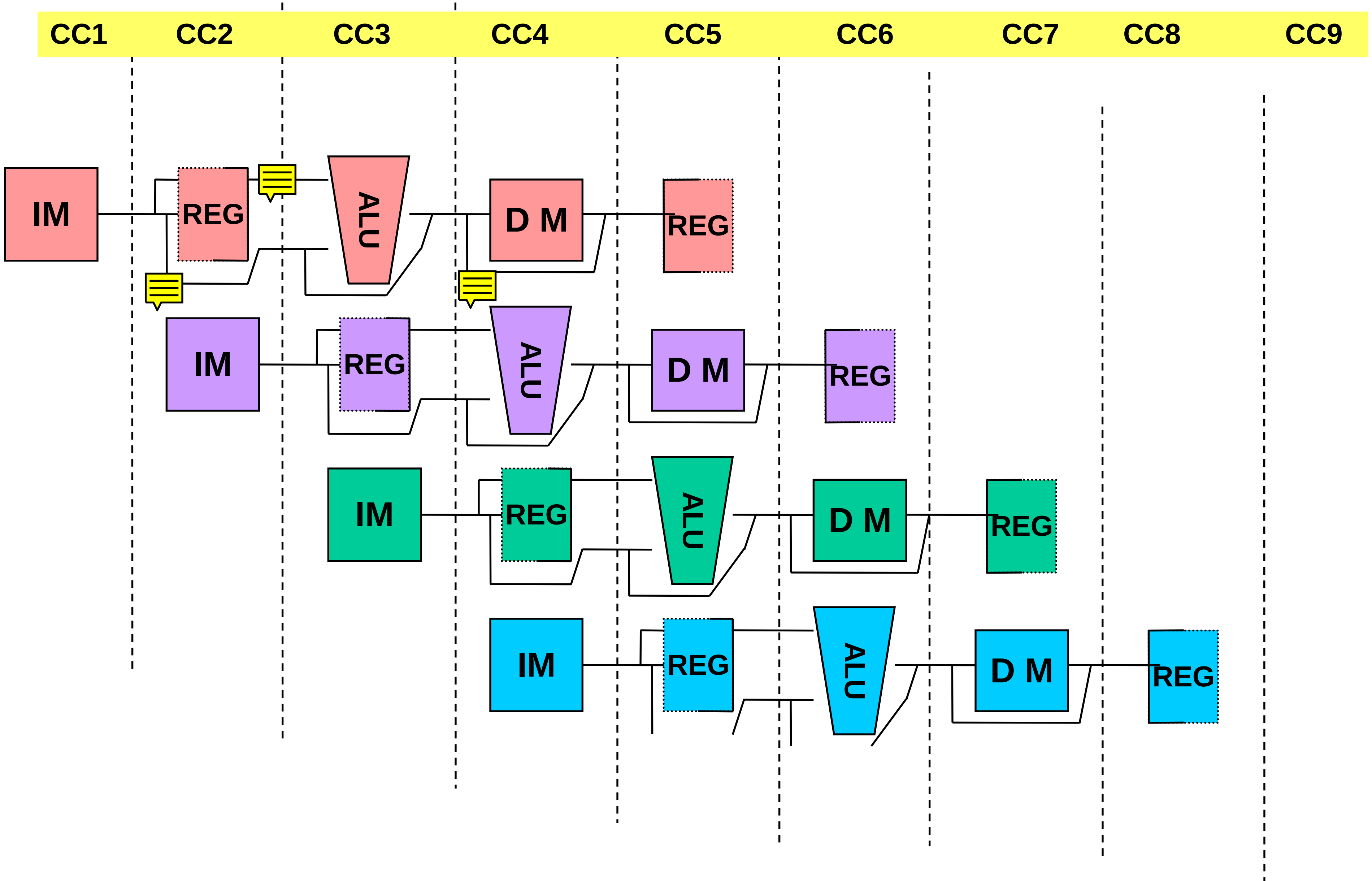


Visualizing Pipelining

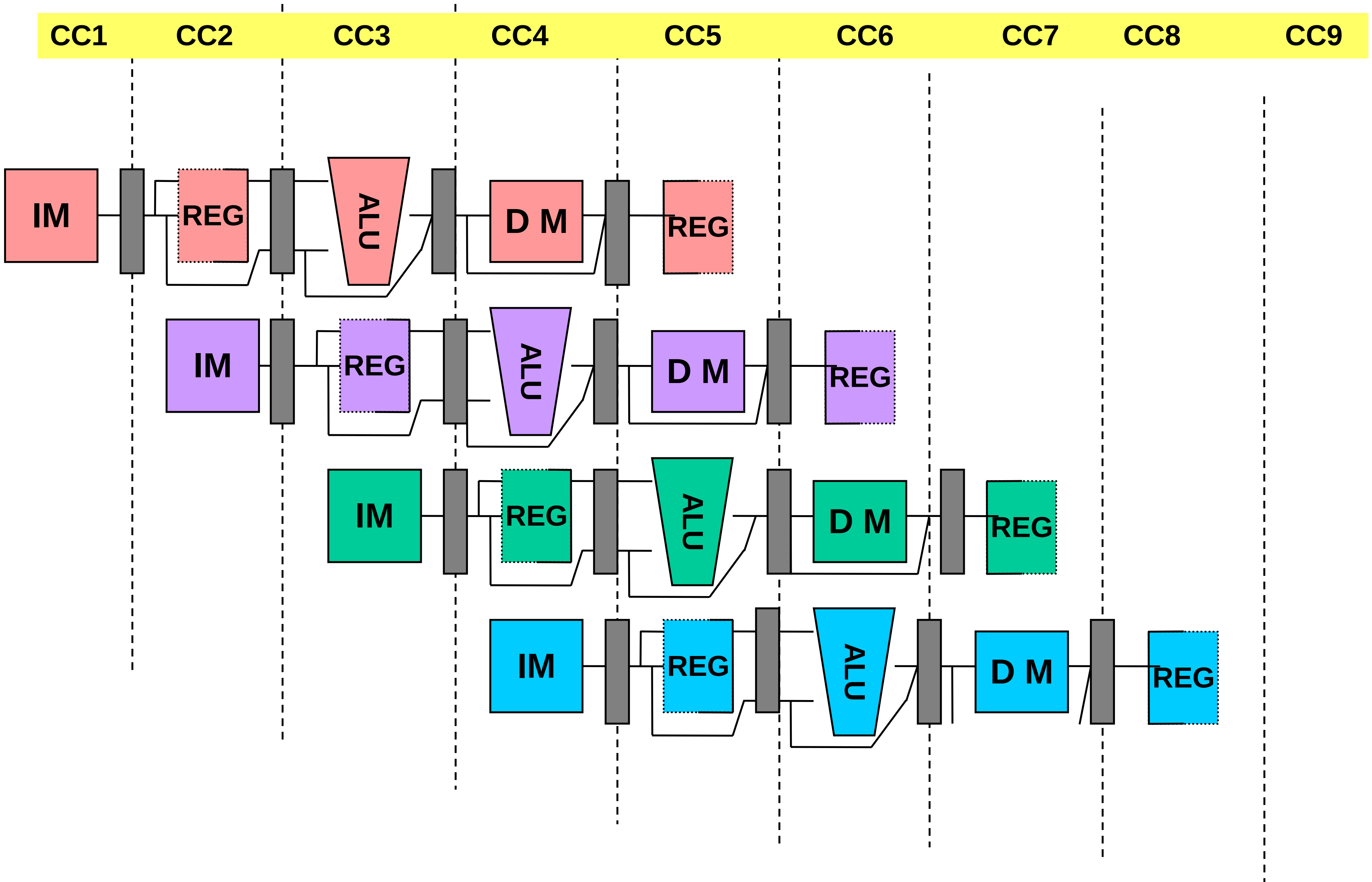


	Clock number							
Instruction number	1	2	3	4	5	6	7	8
i	IF	ID	EX	MEM	WB			
i+1		IF	ID	EX	MEM	WB		
i+2			IF	ID	EX	MEM	WB	
i+3				IF	ID	EX	MEM	WB
i+4					IF	ID	EX	MEM

Visualizing Pipelining



Visualizing Pipelining



johnjose@iitg.ac.in

http://www.iitg.ac.in/johnjose/