RISC Design

Pipeline Hazards

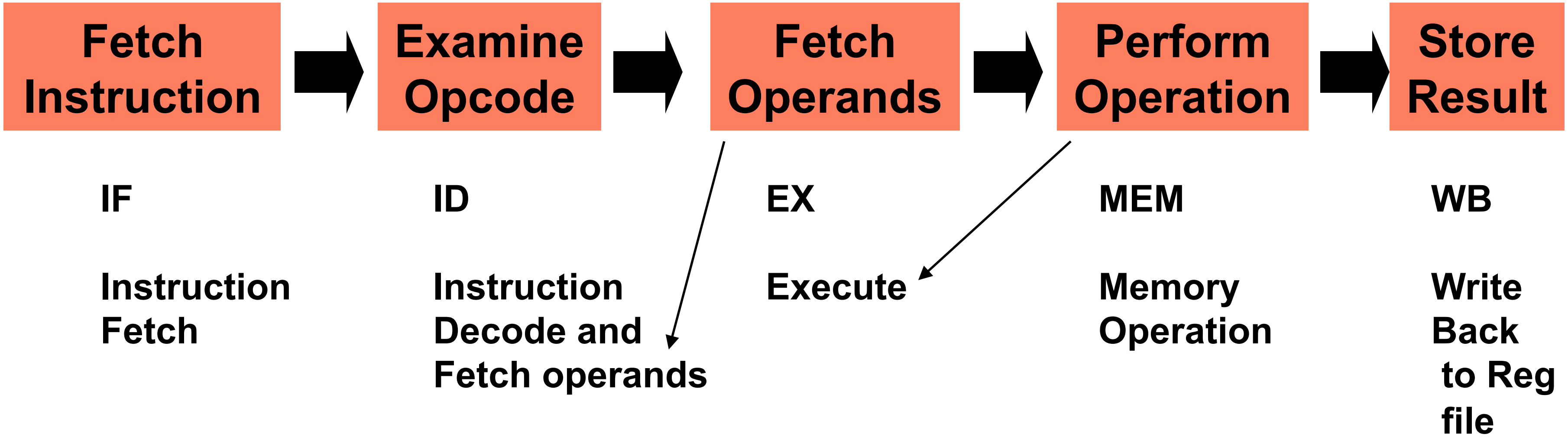
Virendra Singh
Computer Architecture and Dependable Systems Lab
Department of Electrical Engineering
Indian Institute of Technology Bombay
http://www.ee.iitb.ac.in/~viren/
E-mail: viren@ee.iitb.ac.in

EE-309: Microprocessors

Lecture 36 (19 Oct 2015)

CADSL

Pipelining of RISC Instructions



Fetch Instruction → Examine Opcode → Fetch Operands → Perform Operation → Store Result

IF

Instruction Fetch

ID

Instruction Decode and Fetch operands

EX

Execute

MEM

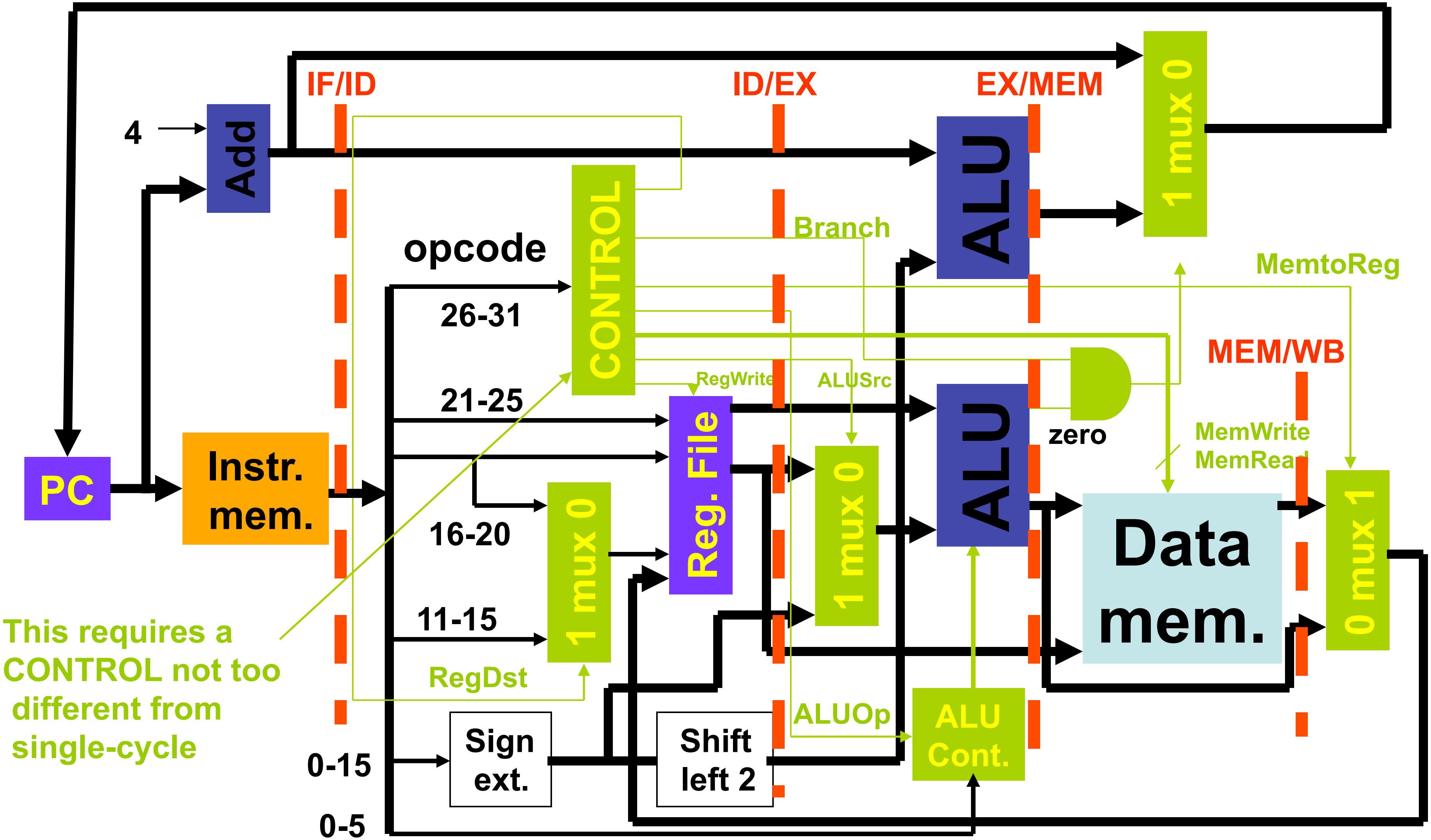
Memory Operation

WB

Write Back to Reg file

Although an instruction takes five clock cycles, one instruction is completed every cycle.

Pipeline Registers



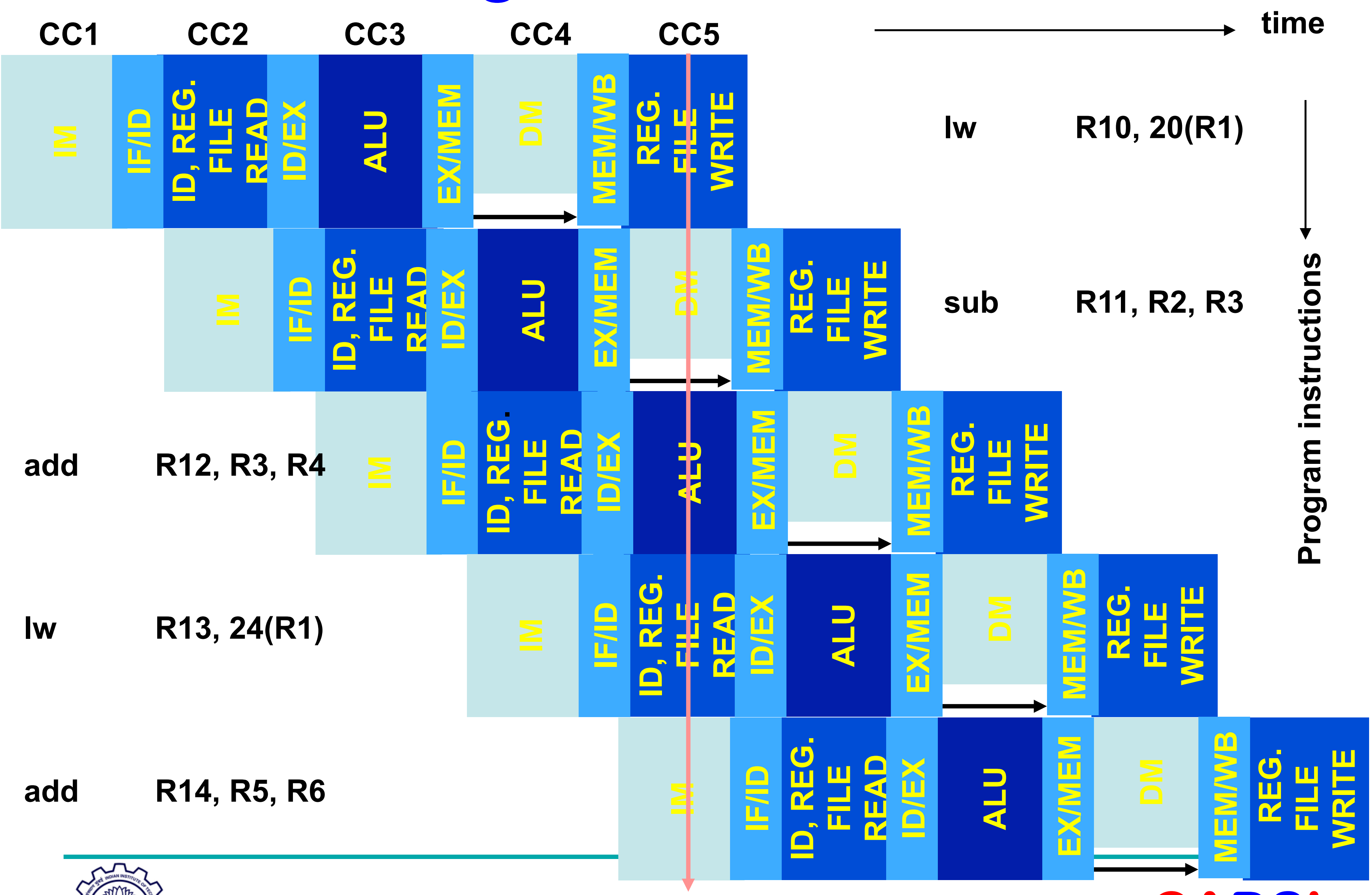
This requires a CONTROL not too different from single-cycle

Pipeline Register Functions

- Four pipeline registers are added:

Register name	Data held
IF/ID	PC+4, Instruction word (IW)
ID/EX	PC+4, R1, R2, IW(0-15) sign ext., IW(11-15)
EX/MEM	PC+4, zero, ALUResult, R2, IW(11-15) or IW(16-20)
MEM/WB	M[ALUResult], ALUResult, IW(11-15) or IW(16-20)

Program Execution



lw R10, 20(R1)

sub R11, R2, R3

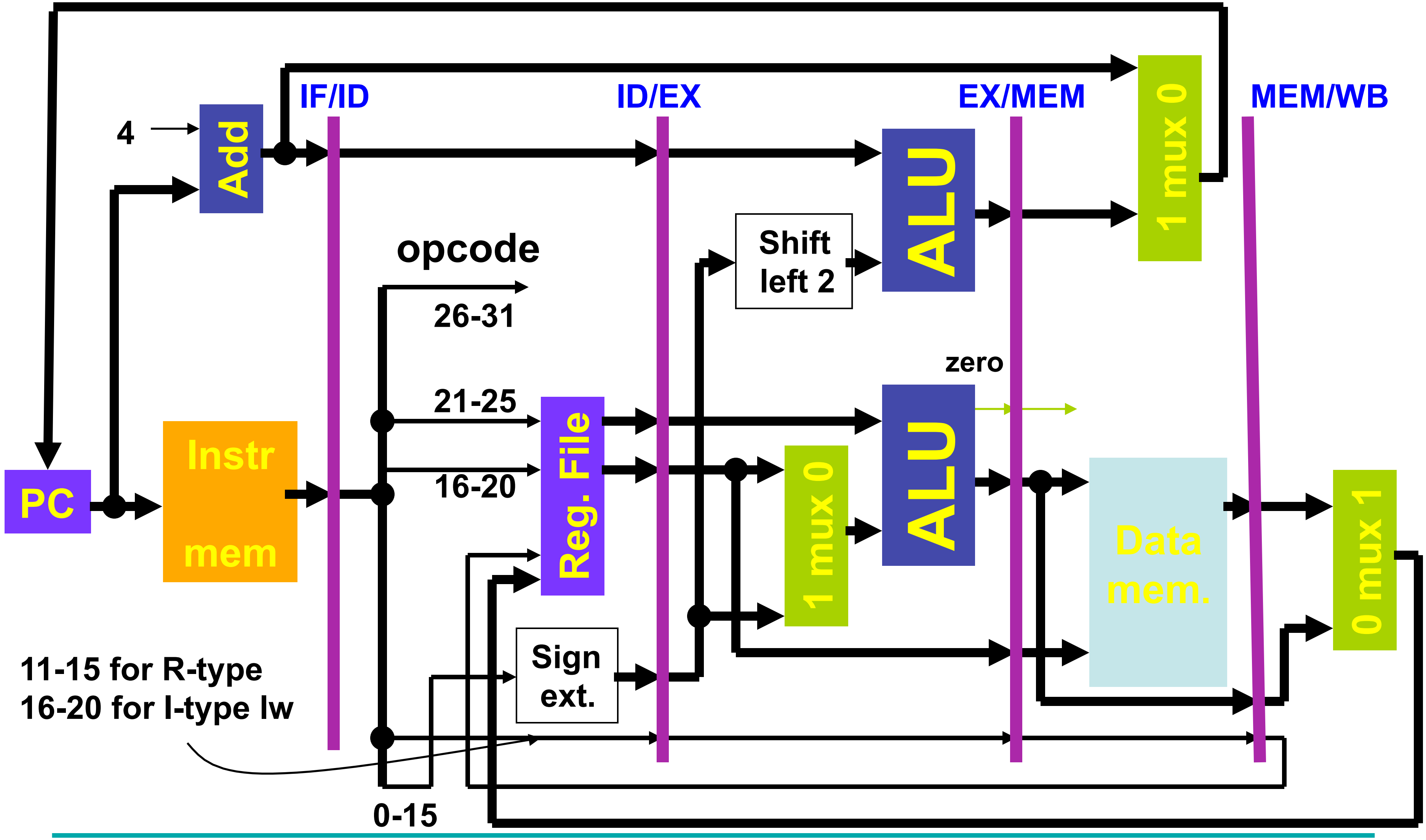
add R12, R3, R4

lw R13, 24(R1)

add R14, R5, R6

CC5

IF: add R14, R5, R6 ID: lw R13, 24(R1) EX: add R12, R3, R4 MEM: sub R11, R2, R3 WB: lw R10, 20(R1)



Single Lane Traffic

Advantages of Pipeline

- After the fifth cycle (CC5), one instruction is completed each cycle; CPI ≈ 1, neglecting the initial pipeline latency of 5 cycles.
  - *Pipeline latency is defined as the number of stages in the pipeline, or*
  - *The number of clock cycles after which the first instruction is completed.*
- The clock cycle time is about four times shorter than that of single-cycle datapath and about the same as that of multicycle datapath.
- For multicycle datapath, CPI = 3. ….
- So, pipelined execution is faster, but . . .

Science is always wrong. It never solves a problem without creating ten more.

George Bernard Shaw

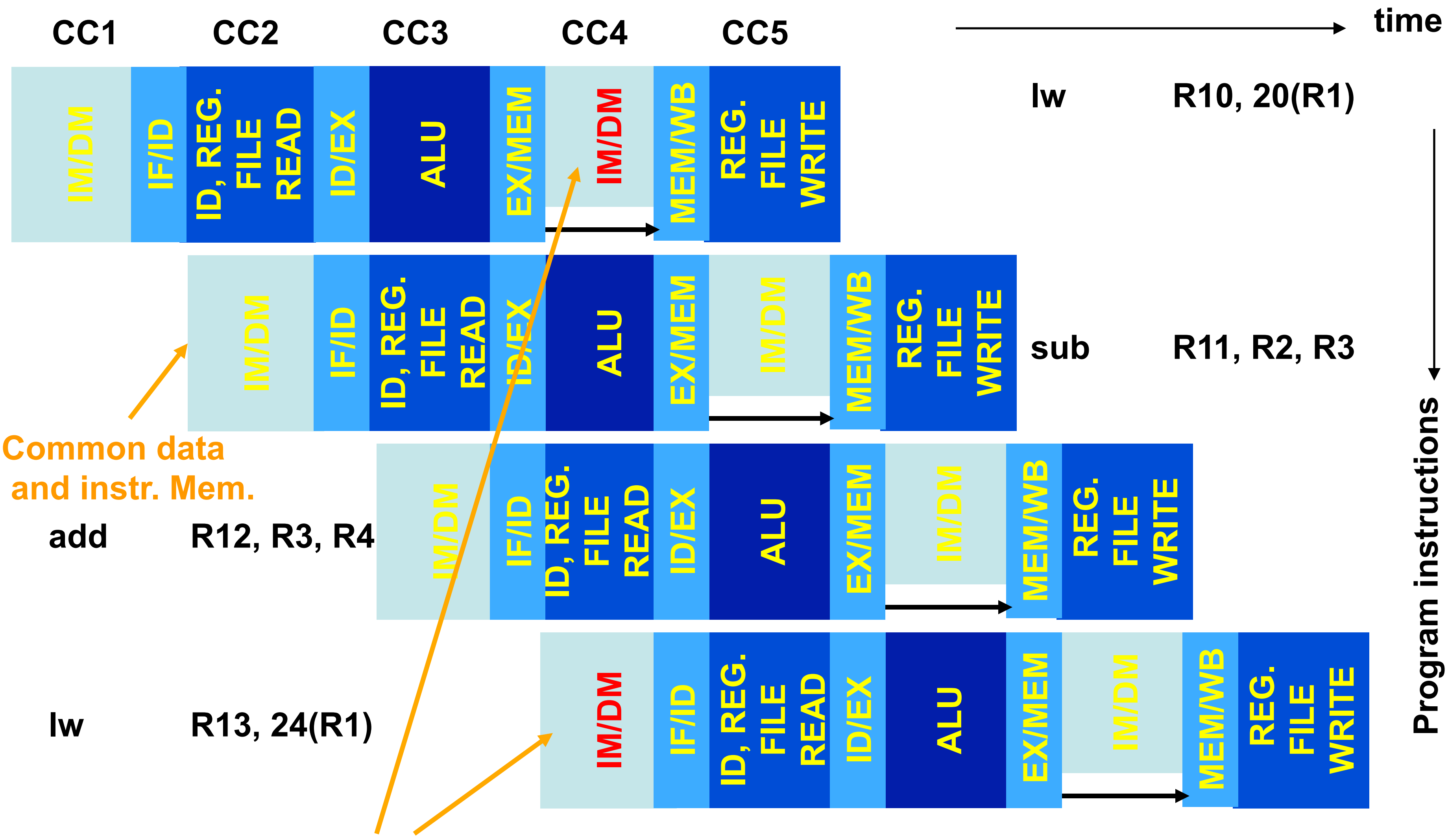
Pipeline Hazards

- Definition: *Hazard in a pipeline is a situation in which the next instruction cannot complete execution one clock cycle after completion of the present instruction.*
- Three types of hazards:
  - Structural hazard (resource conflict)
  - Data hazard
  - Control hazard

Structural Hazard

- Two instructions cannot execute due to a resource conflict.
- Example: Consider a computer with a common data and instruction memory. The fourth cycle of a *lw* instruction requires memory access (memory read) and at the same time the first cycle of the fourth instruction requires instruction fetch (memory read). This will cause a memory resource conflict.

Example of Structural Hazard



lw R10, 20(R1)

sub R11, R2, R3

add R12, R3, R4

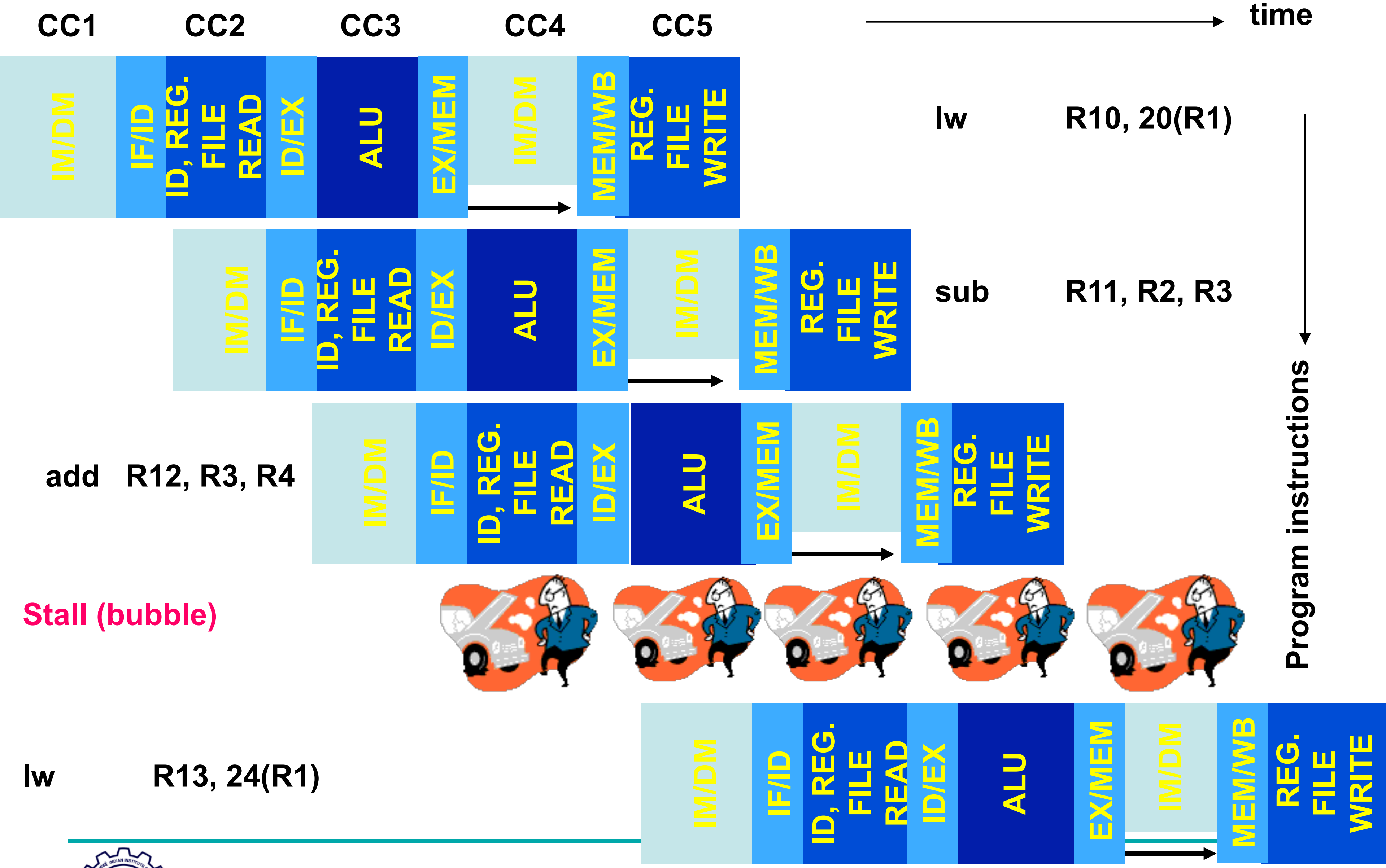
lw R13, 24(R1)

Nedded by two instructions

Possible Remedies for Structural Hazards

- Provide duplicate hardware resources in datapath.
- Control unit or compiler can insert delays (no-op cycles) between instructions. This is known as pipeline *stall* or *bubble*.

Stall (Bubble) for Structural Hazard



lw R10, 20(R1)

sub R11, R2, R3

add R12, R3, R4

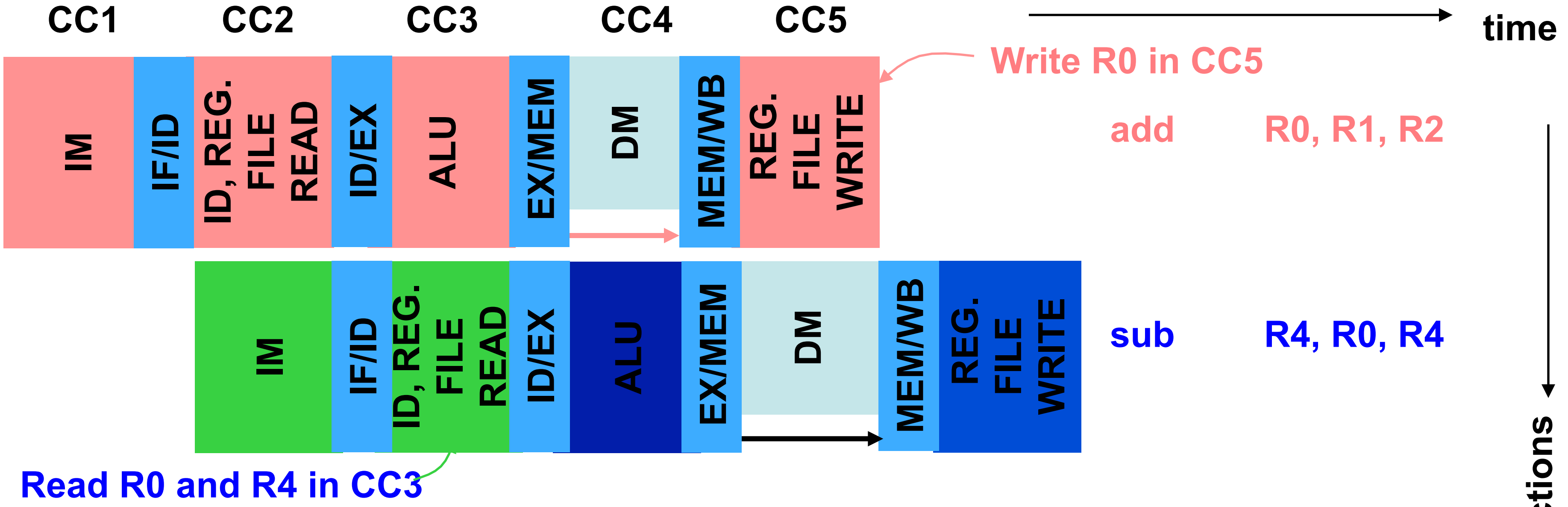
Stall (bubble)

lw R13, 24(R1)

Data Hazard

- Data hazard means that an instruction cannot be completed because the needed data, to be generated by another instruction in the pipeline, is not available.
- Example: consider two instructions:
  - add R0, R1, R2
  - sub R3, R0, R4 # needs R0

Example of Data Hazard



We need to read R0 from reg file in cycle 3

But R0 will not be written in reg file until cycle 5

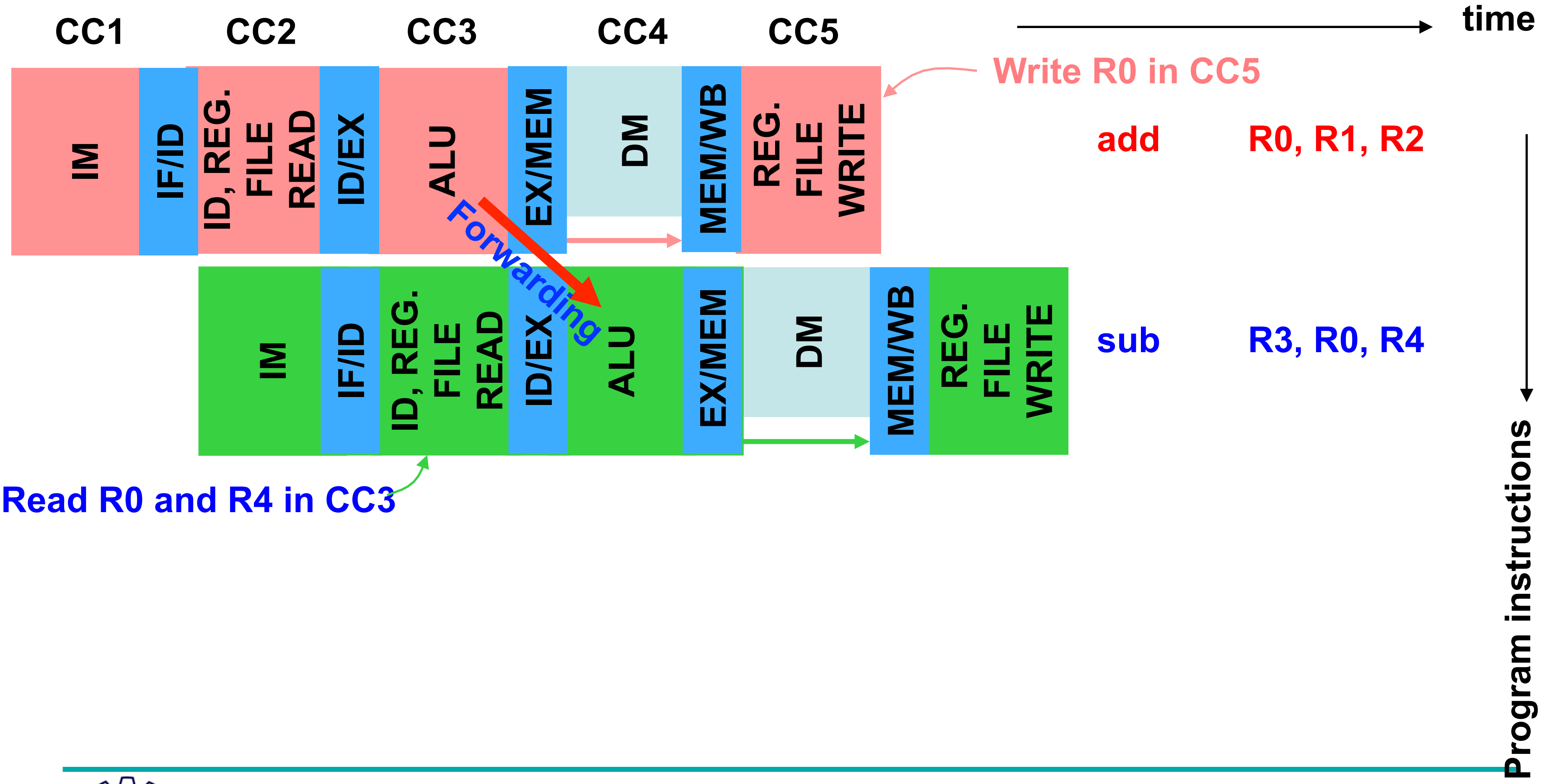
However, R0 will only be used in cycle 4

And it is available at the end of cycle 3

Forwarding or Bypassing

- Output of a resource used by an instruction is forwarded to the input of some resource being used by another instruction.
- Forwarding can eliminate some, but not all, data hazards.

Forwarding for Data Hazard



CC1 CC2 CC3 CC4 CC5

time

IM | IF/ID | ID, REG. FILE READ | ID/EX | ALU | EX/MEM | DM | MEM/WB | REG. FILE WRITE

Write R0 in CC5

add R0, R1, R2

IM | IF/ID | ID, REG. FILE READ | ID/EX | ALU | EX/MEM | DM | MEM/WB | REG. FILE WRITE

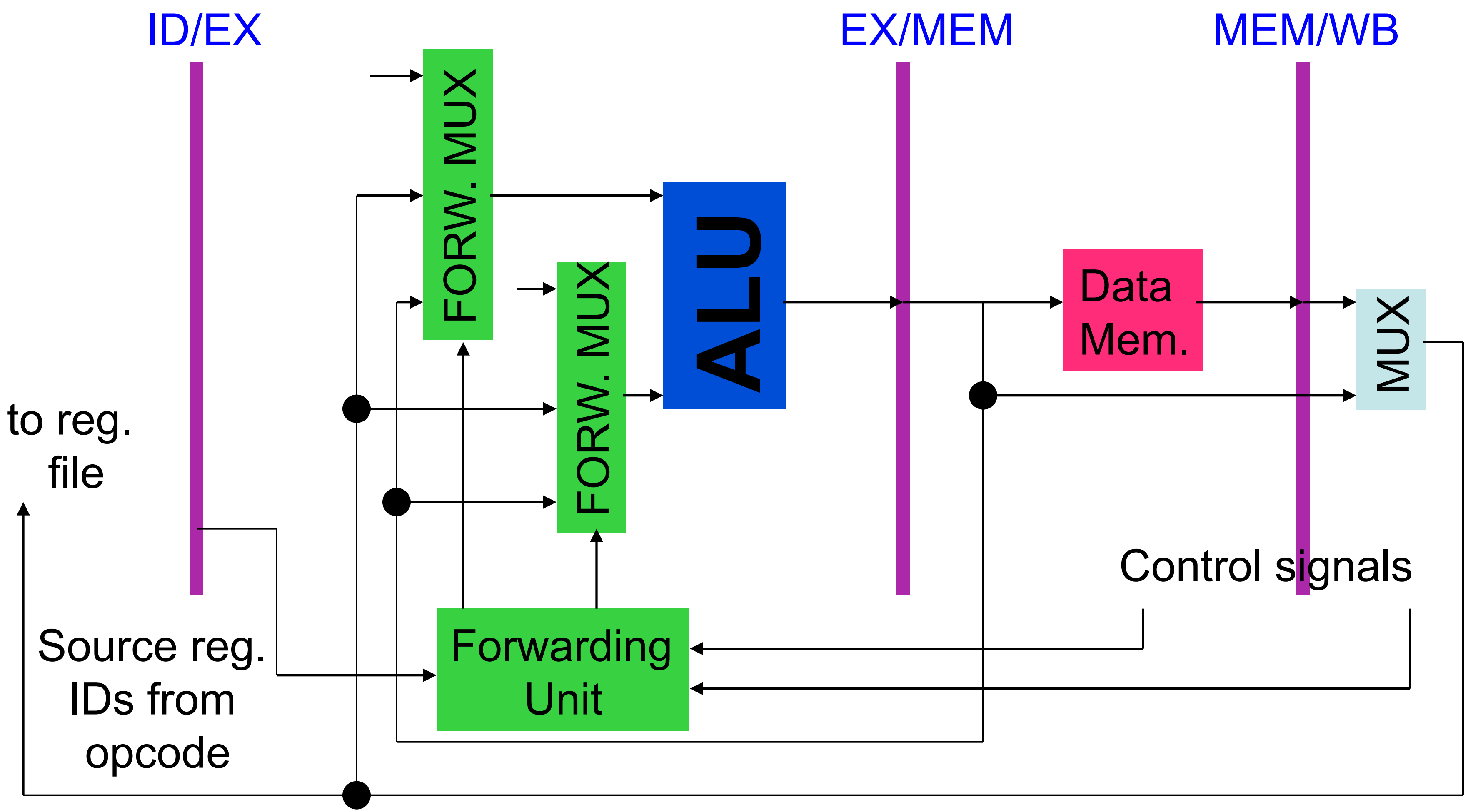
Forwarding

sub R3, R0, R4

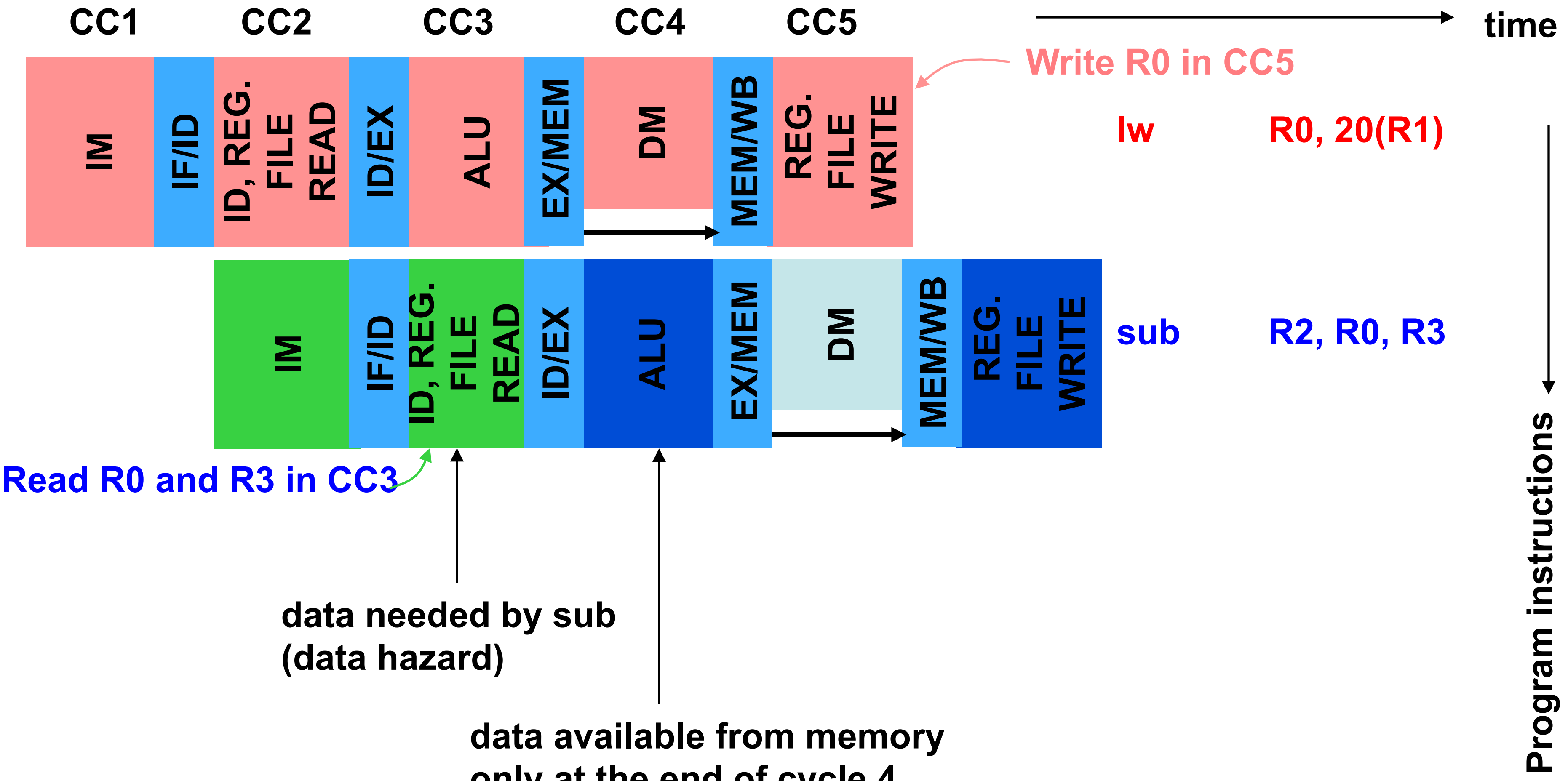
Read R0 and R4 in CC3

Program instructions

Forwarding Unit Hardware



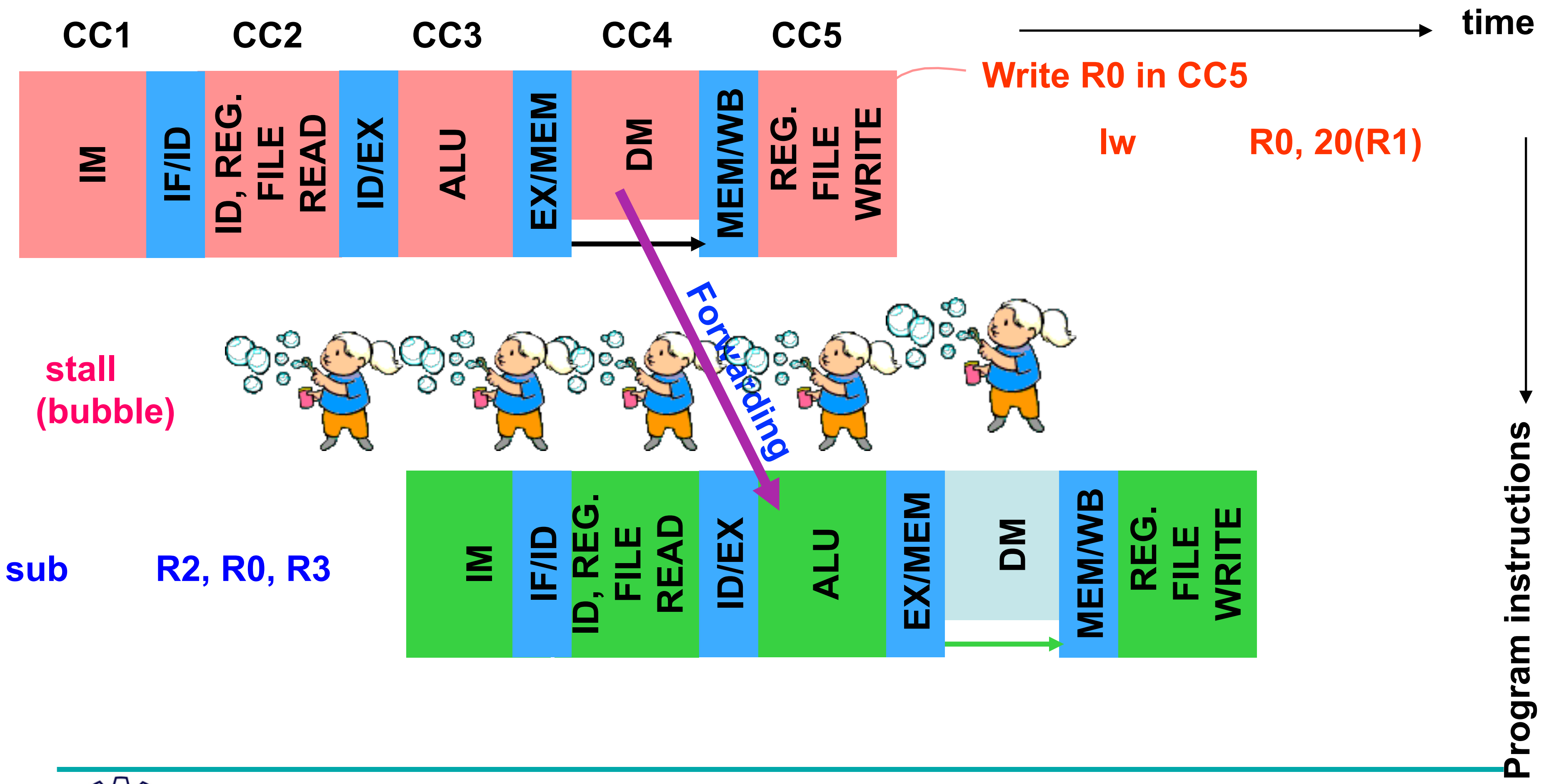
Forwarding Alone May Not Work



data needed by sub
(data hazard)

data available from memory
only at the end of cycle 4

Use Bubble and Forwarding



CC1 CC2 CC3 CC4 CC5

time

Write R0 in CC5

lw R0, 20(R1)

IM | IF/ID | ID, REG. FILE READ | ID/EX | ALU | EX/MEM | DM | MEM/WB | REG. FILE WRITE

Forwarding

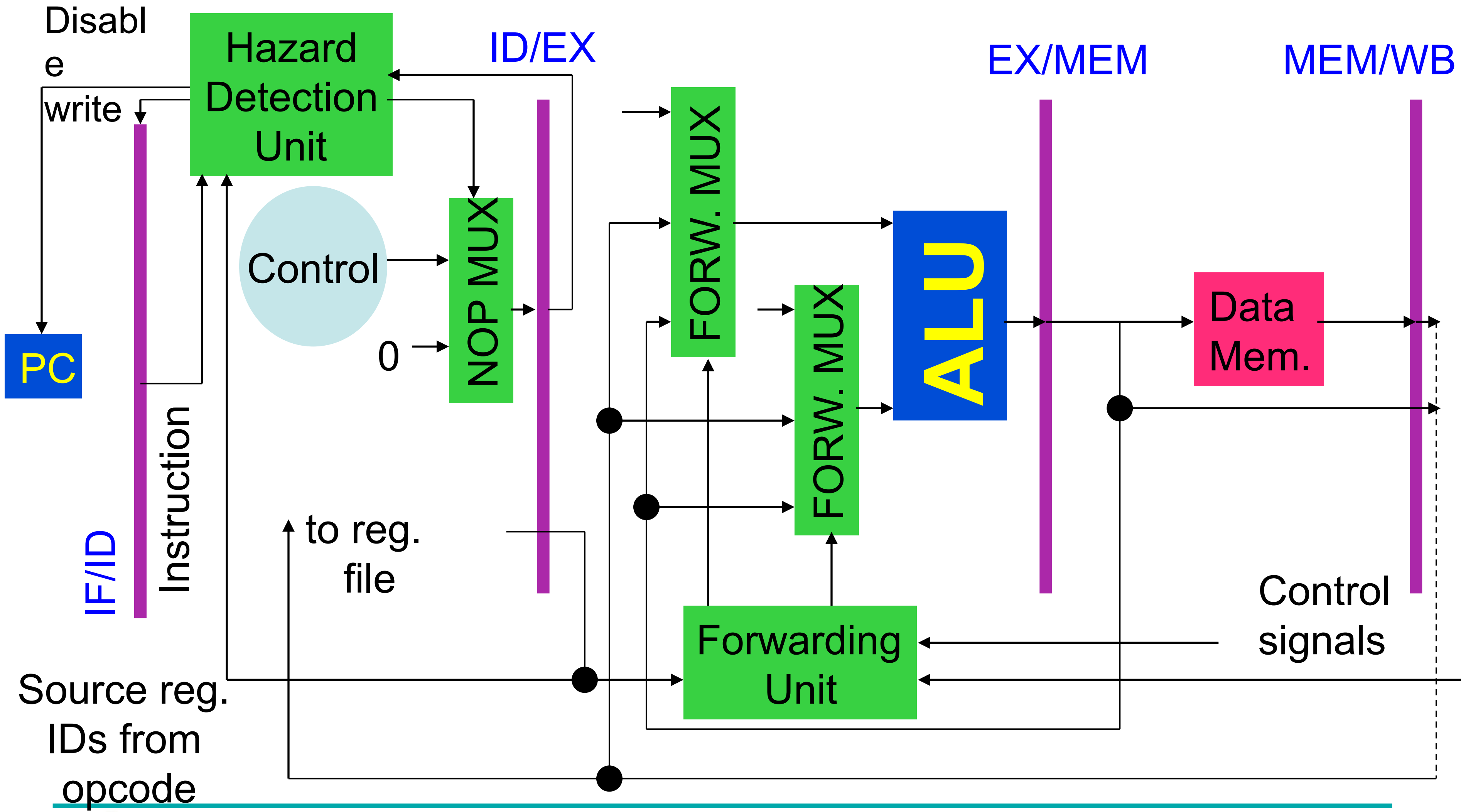
stall (bubble)

sub R2, R0, R3

IM | IF/ID | ID, REG. FILE READ | ID/EX | ALU | EX/MEM | DM | MEM/WB | REG. FILE WRITE

Program instructions

Hazard Detection Unit Hardware



Thank You