RISC Architecture: Pipelining

Virendra Singh

Associate Professor
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

Computer Organization & Architecture

Lecture 16 (29 April 2013)

CADSL

ILP: Instruction Level Parallelism

- Single-cycle and multi-cycle datapaths execute one instruction at a time.

- How can we get better performance?

- Answer: Execute multiple instruction at a time:

  - Pipelining – Enhance a multi-cycle datapath to fetch one instruction every cycle.

  - Parallelism – Fetch multiple instructions every cycle.

CADSL

Pipelining in a Computer

➢ Divide datapath into nearly equal tasks, to be performed serially and requiring non-overlapping resources.

➢ Insert registers at task boundaries in the datapath; registers pass the output data from one task as input data to the next task.

➢ Synchronize tasks with a clock having a cycle time that just exceeds the time required by the longest task.

➢ Break each instruction down into a fixed number of tasks so that instructions can be executed in a staggered fashion.

CADSL

Single-Cycle Datapath

Instruction class	Instr. fetch (IF)	Instr. Decode (also reg. file read) (ID)	Execution (ALU Operation) (EX)	Data access (MEM)	Write Back (Reg. file write) (WB)	Total time
lw	2ns	1ns	2ns	2ns	1ns	8ns
sw	2ns	1ns	2ns	2ns		8ns
R-format add, sub, and, or, slt	2ns	1ns	2ns		1ns	8ns
B-format, beq	2ns	1ns	2ns			8ns

No operation on data; idle time equalizes instruction length to a fixed clock period.

CADSL

Pipelined Datapath

Instruction class	Instr. fetch (IF)	Instr. Decode (also reg. file read) (ID)	Execu-tion (ALU Opera-tion) (EX)	Data access (MEM)	Write Back (Reg. file write) (WB)	Total time
lw	2ns	~~1ns~~ 2ns	2ns	2ns	~~1ns~~ 2ns	10ns
sw	2ns	~~1ns~~ 2ns	2ns	2ns	~~1ns~~ 2ns	10ns
R-format: add, sub, and, or, slt	2ns	~~1ns~~ 2ns	2ns	2ns	~~1ns~~ 2ns	10ns
B-format: beq	2ns	~~1ns~~ 2ns	2ns	2ns	~~1ns~~ 2ns	10ns

No operation on data; idle time inserted to equalize instruction lengths.

CADSL

Single-Cycle Datapath

CADSL
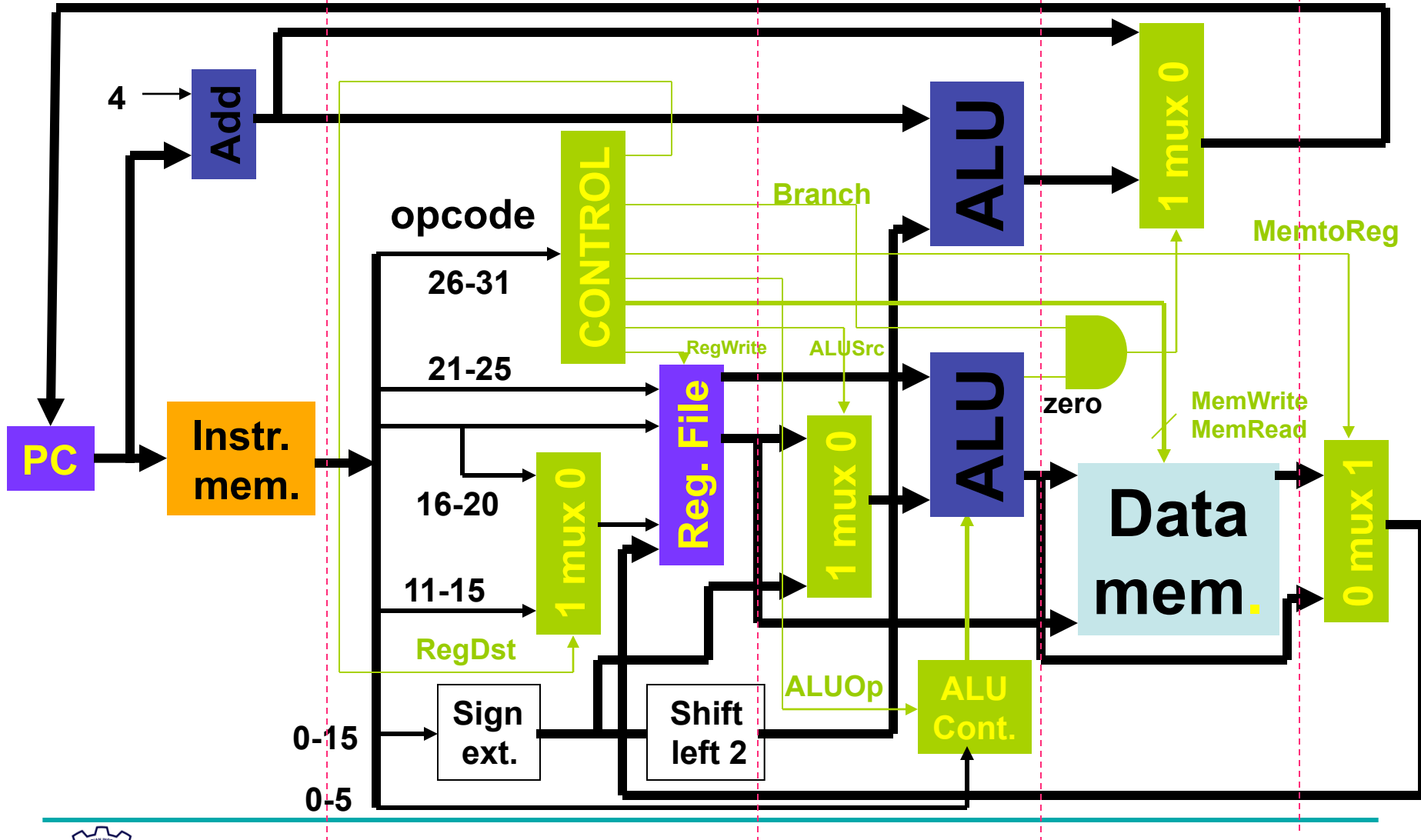
Pipelining of RISC Instructions

Fetch Instruction	→	Examine Opcode	→	Fetch Operands	→	Perform Operation	→	Store Result
IF		ID		EX		MEM		WB
Instruction Fetch		Instruction Decode and Fetch operands		Execute		Memory Operation		Write Back to Reg file
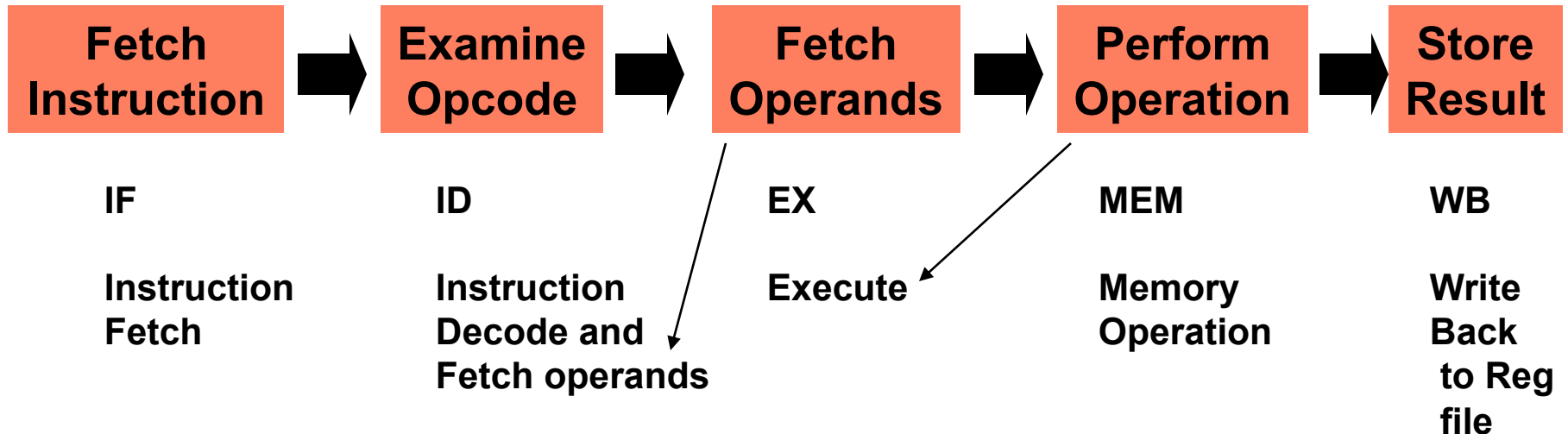
Although an instruction takes five clock cycles, one instruction is completed every cycle.

CADSL

Pipeline Registers

CADSL
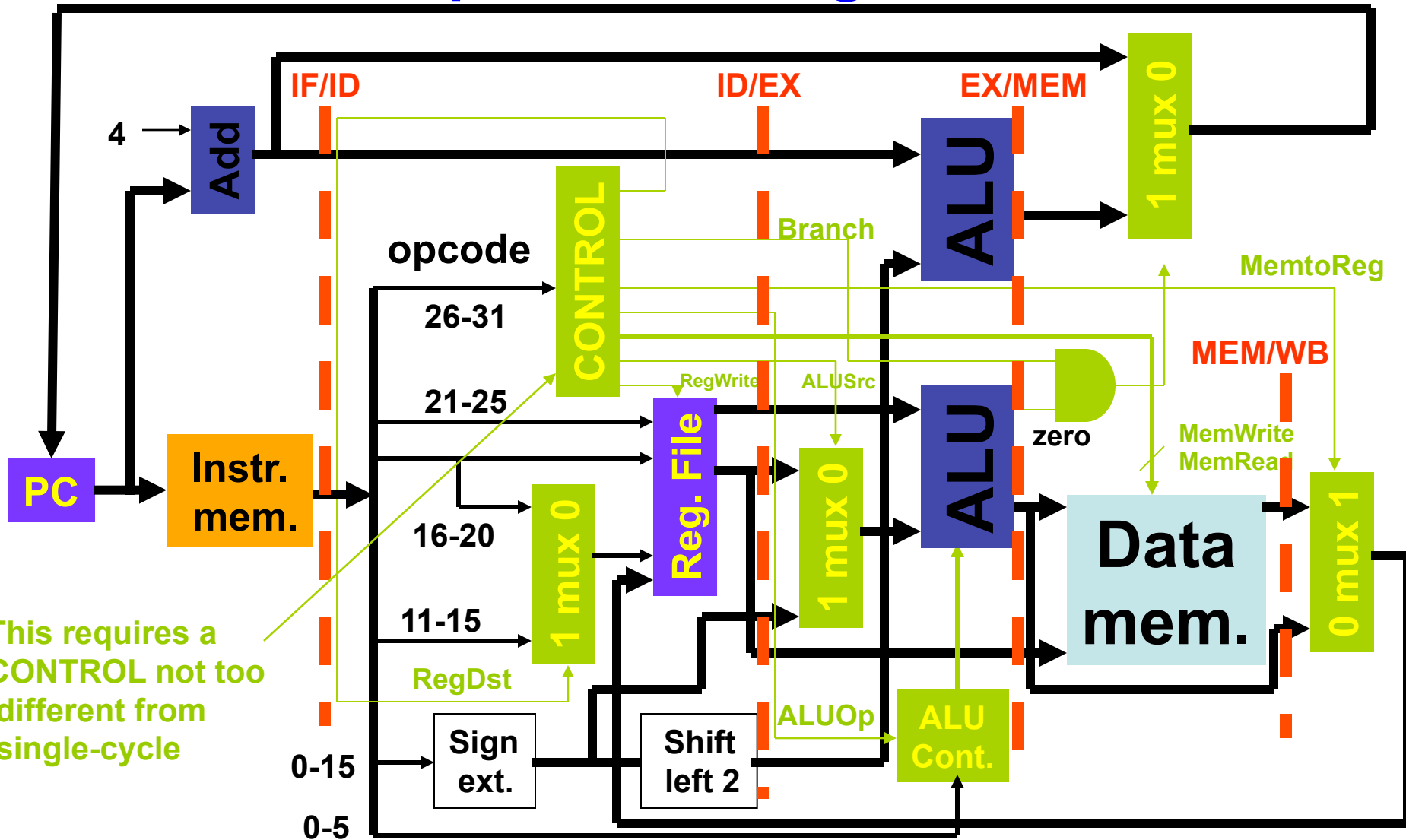
Pipeline Register Functions

- Four pipeline registers are added:

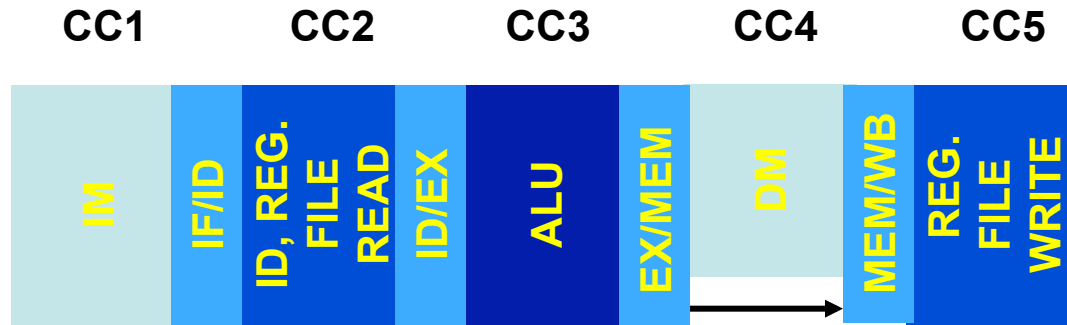| Register name	Data held
IF/ID	PC+4, Instruction word (IW)
ID/EX	PC+4, R1, R2, IW(0-15) sign ext., IW(11-15)
EX/MEM	PC+4, zero, ALUResult, R2, IW(11-15) or IW(16-20)
MEM/WB	M[ALUResult], ALUResult, IW(11-15) or IW(16-20)

CADSL

Pipelined Datapath

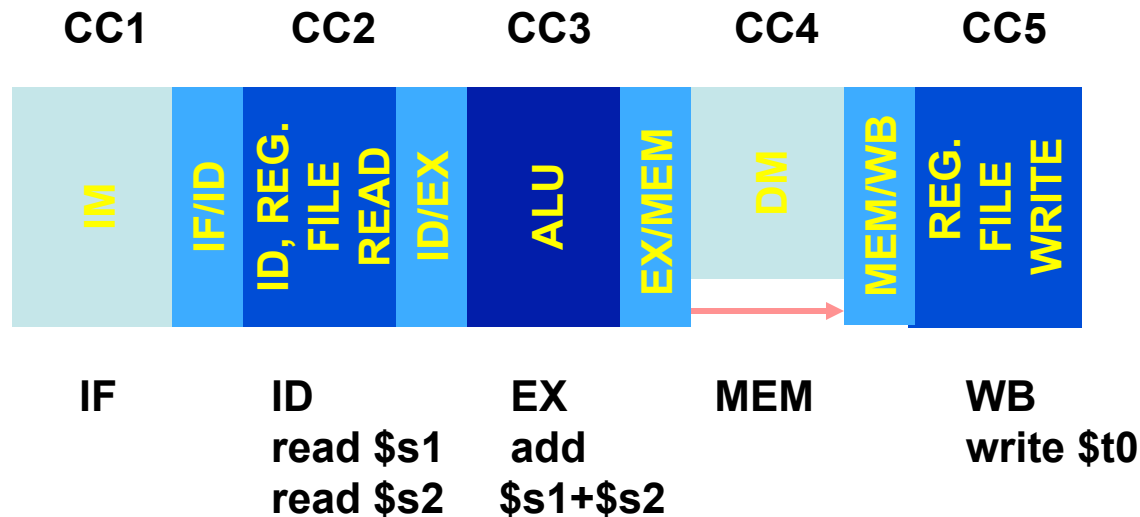CADSL
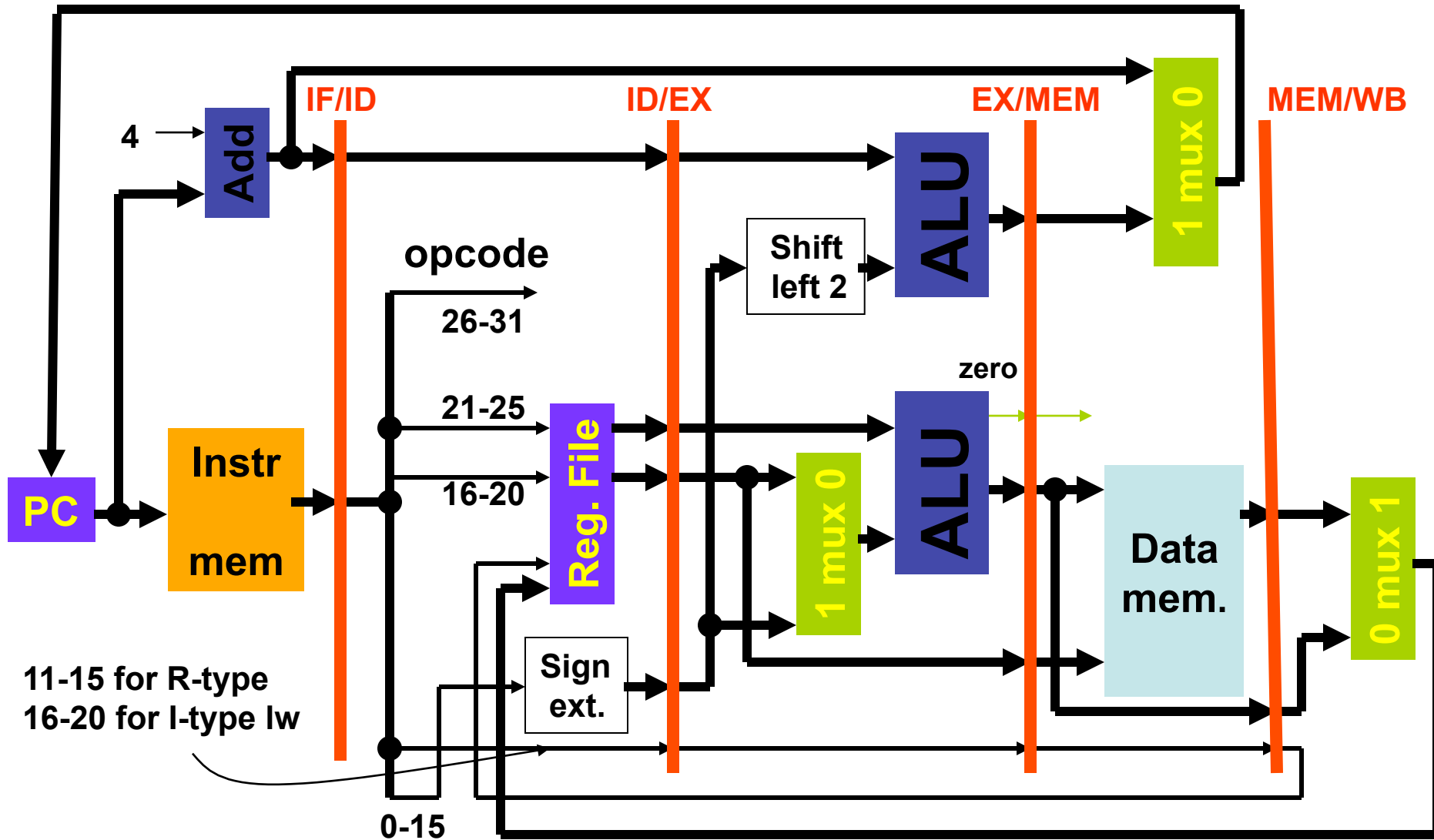
Five-Cycle Pipeline

CADSL

Add Instruction

- add $t0, $s1, $s2

  Machine instruction word

  000000 10001 10010 01000 00000 100000

  opcode $s1 $s2 $t0   function

CC1	CC2	CC3	CC4	CC5
IM	IF/ID — ID, REG. FILE READ — ID/EX	ALU — EX/MEM	DM	MEM/WB — REG. FILE WRITE

IF	ID	EX	MEM	WB
	read $s1	add		write $t0
	read $s2	$s1+$s2		

CADSL

Pipelined Datapath Executing add

CADSL
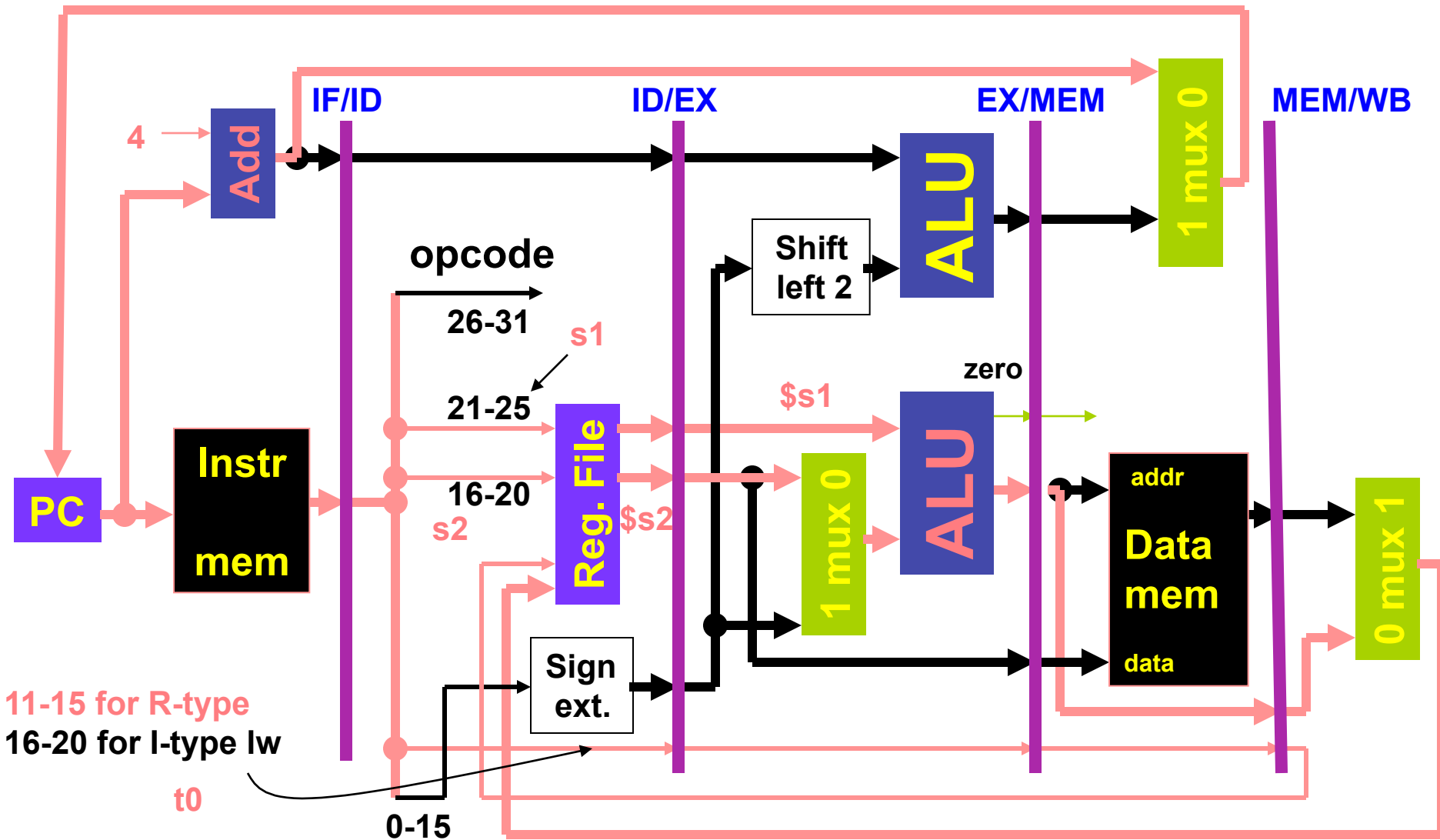
Load Instruction

- lw $t0, 1200 ($t1)

  100011 01001 01000 0000 0100 1000 0000

  opcode $t1 $t0   1200

CC1	CC2	CC3	CC4	CC5

IF

ID
read $t1
sign ext
1200

EX
add
$t1+1200

MEM
read
M[addr]

WB
write $t0

CADSL
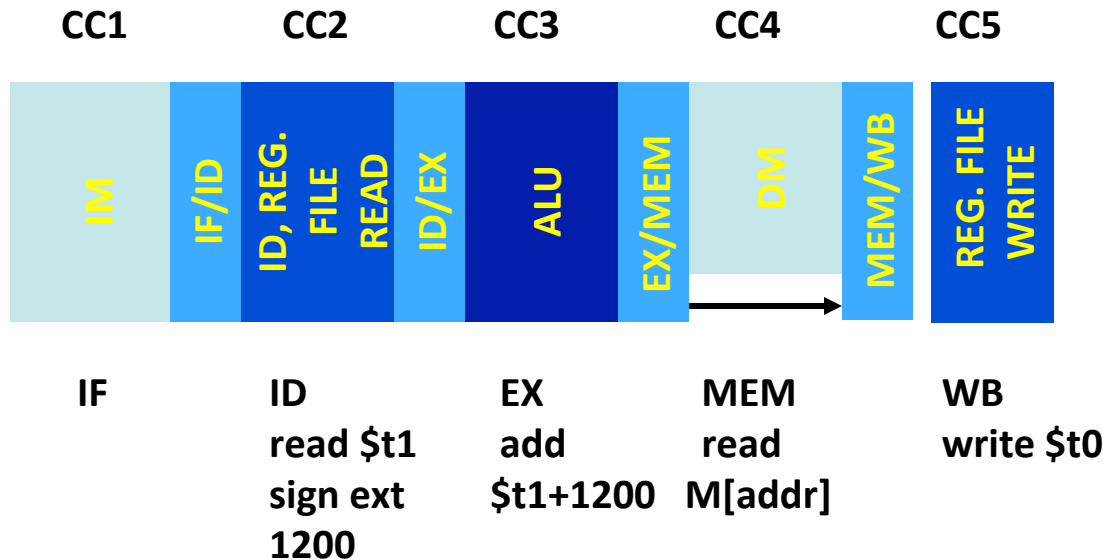
Pipelined Datapath Executing lw

IF/ID **ID/EX** **EX/MEM** **MEM/WB**

4

Add

opcode

26-31

t1

21-25

Instr mem

PC

$t1

zero

Shift left 2

ALU

ALU

1 mux 0

1 mux 0

16-20

Reg. File

ALU

Data mem

addr

data

0 mux 1

Sign ext.
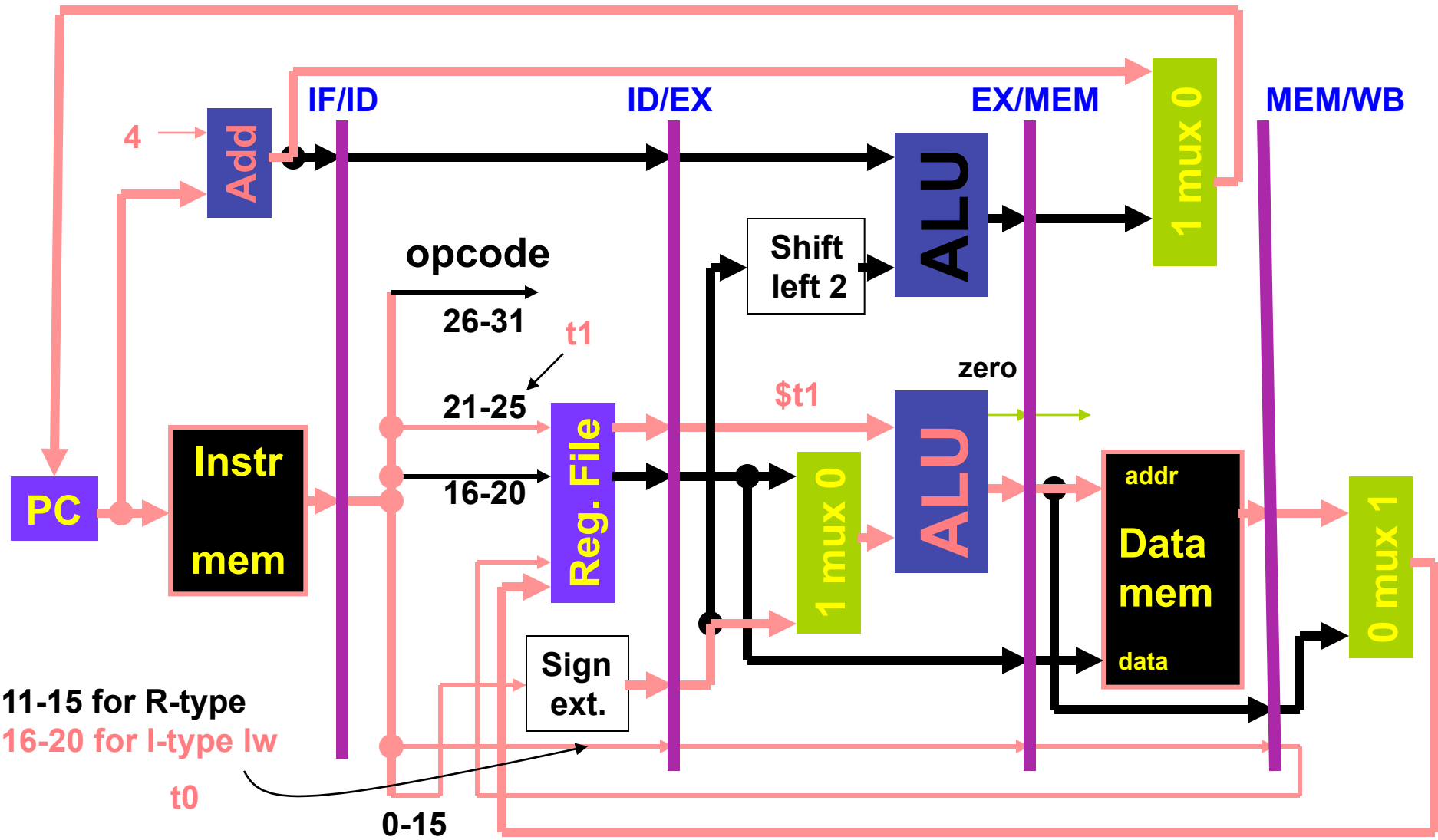
11-15 for R-type
16-20 for I-type lw

t0

0-15

1200

CADSL

Store Instruction

- sw $t0, 1200 ($t1)

101011 01001 01000   0000 0100 1000 0000

opcode $t1 $t0   1200

CC1	CC2	CC3	CC4	CC5

IF

ID
read $t1
sign ext
1200

EX
add
$t1+1200
(addr)

MEM
write
M[addr]
← $t0

WB

CADSL
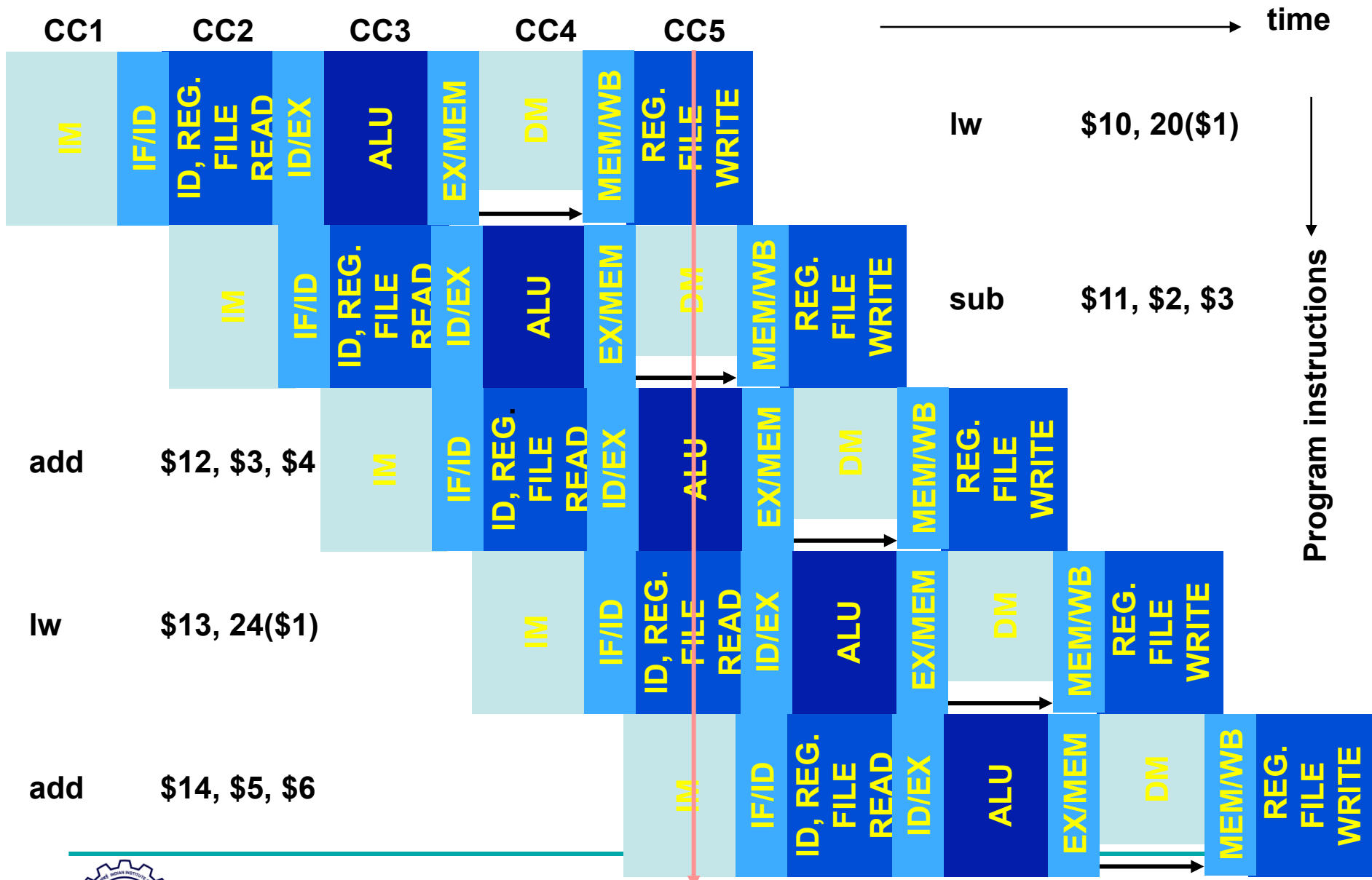
Pipelined Datapath Executing sw

CADSL

Executing a Program

Consider a five-instruction segment:

```
lw    $10, 20($1)
sub   $11, $2, $3
add   $12, $3, $4
lw    $13, 24($1)
add   $14, $5, $6
```
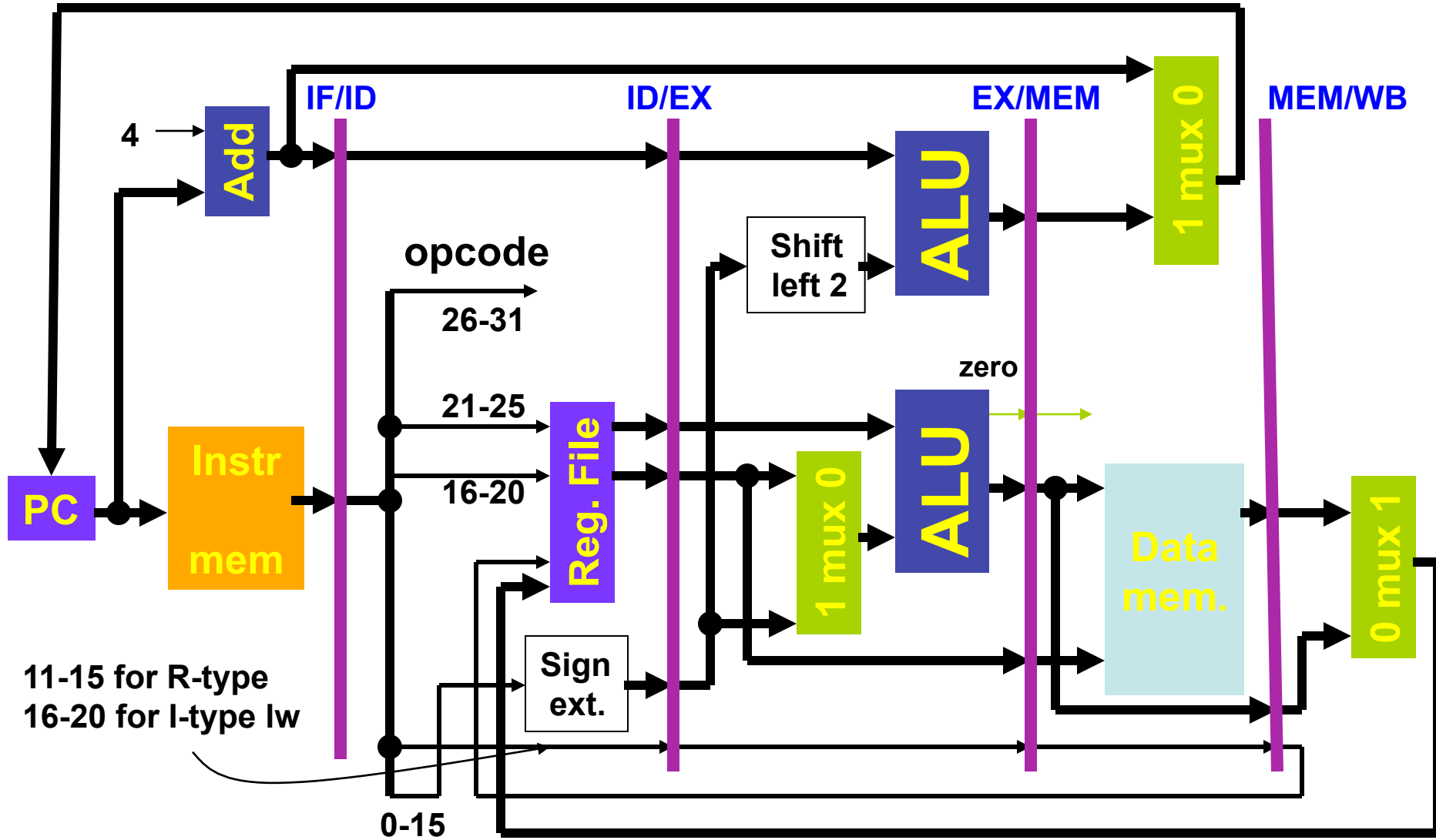
CADSL

Program Execution

CC1   CC2   CC3   CC4   CC5   time

| | IM | IF/ID | ID, REG. FILE READ | ID/EX | ALU | EX/MEM | DM | MEM/WB | REG. FILE WRITE | | lw | $10, 20($1) |

| | | IM | IF/ID | ID, REG. FILE READ | ID/EX | ALU | EX/MEM | DM | MEM/WB | REG. FILE WRITE | sub | $11, $2, $3 |

add $12, $3, $4   IM   IF/ID   ID, REG. FILE READ   ID/EX   ALU   EX/MEM   DM   MEM/WB   REG. FILE WRITE

lw $13, 24($1)   IM   IF/ID   ID, REG. FILE READ   ID/EX   ALU   EX/MEM   DM   MEM/WB   REG. FILE WRITE

add $14, $5, $6   IM   IF/ID   ID, REG. FILE READ   ID/EX   ALU   EX/MEM   DM   MEM/WB   REG. FILE WRITE

Program instructions

CADSL

CC5

IF: add $14, $5, $6   ID: lw $13, 24($1)   EX: add $12, $3, $4   MEM: sub $11, $2, $3   WB: lw $10, 20($1)

CADSL

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 . . .

CADSL

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

George Bernard Shaw

CADSL

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

CADSL

Thank You

CADSL