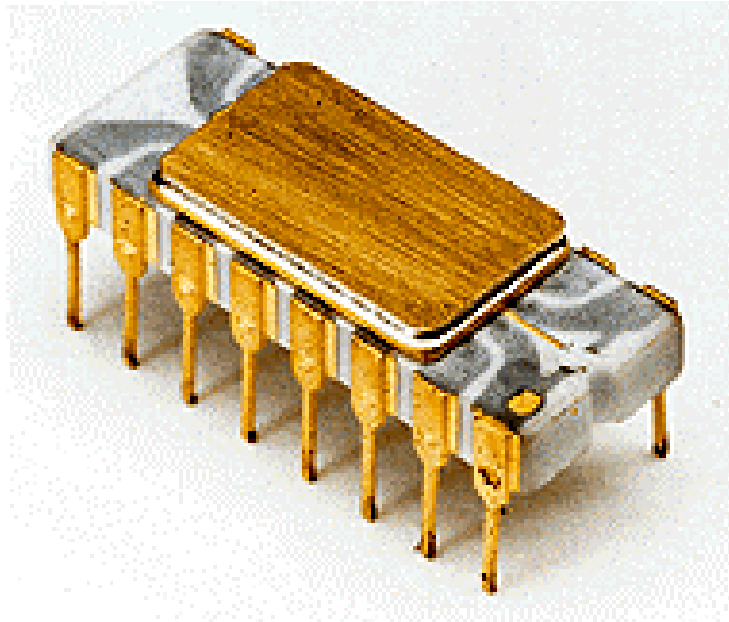


EECS 322 Computer Architecture

Pipeline Control, Data Hazards and Branch Hazards



Models

Single-cycle model (non-overlapping)

- Each instruction executes in a single cycle
- Every instruction and clock-cycle must be **stretched to the slowest instruction** (p.438)

Multi-cycle model (non-overlapping)

- Each instruction executes in variable number of cycles
- The clock-cycle must be **stretched to the slowest step**
- Ability to share functional units within the execution of a single instruction

Pipeline model (overlapping)

- Each instruction executes in several cycles
- The clock-cycle must be **stretched to the slowest step**
- Gains efficiency by overlapping the execution of multiple instructions, increasing hardware utilization. (p. 377)

Overhead

Single-cycle model

- 8 ns Clock (125 MHz), (non-overlapping)
- 1 ALU + 2 adders
- 0 Muxes
- 0 Datapath Register bits (Flip-Flops)

Multi-cycle model

- 2 ns Clock (500 MHz), (non-overlapping)
- 1 ALU + Controller
- 5 Muxes
- 160 Datapath Register bits (Flip-Flops)

Pipeline model

- 2 ns Clock (500 MHz), (overlapping)
- 2 ALU + Controller
- 4 Muxes
- 373 Datapath + 16 **Controlpath** Register bits (Flip-Flops)

Chip Area

Speed



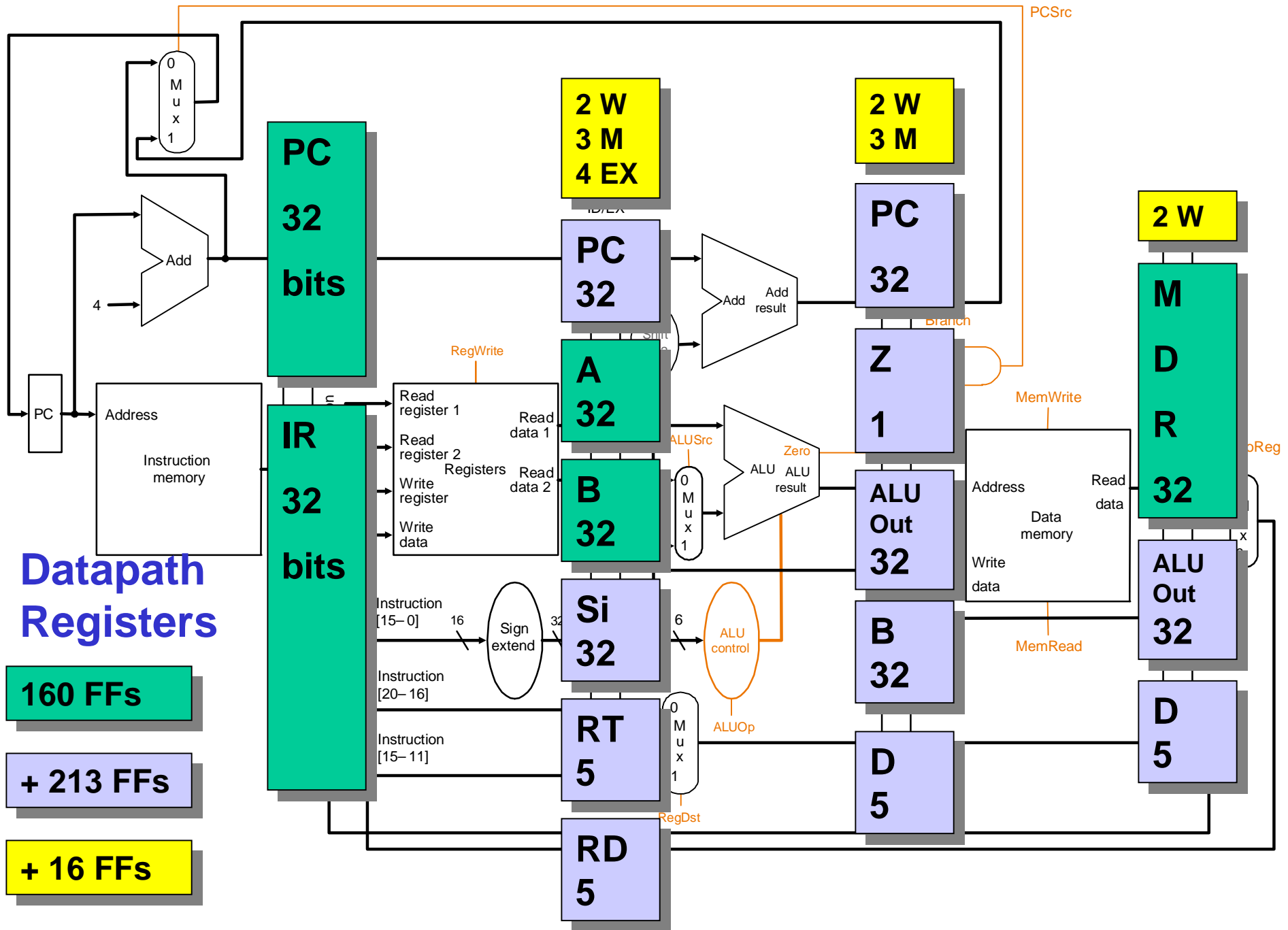


Figure 6.25

Pipeline Control: Controlpath Register bits

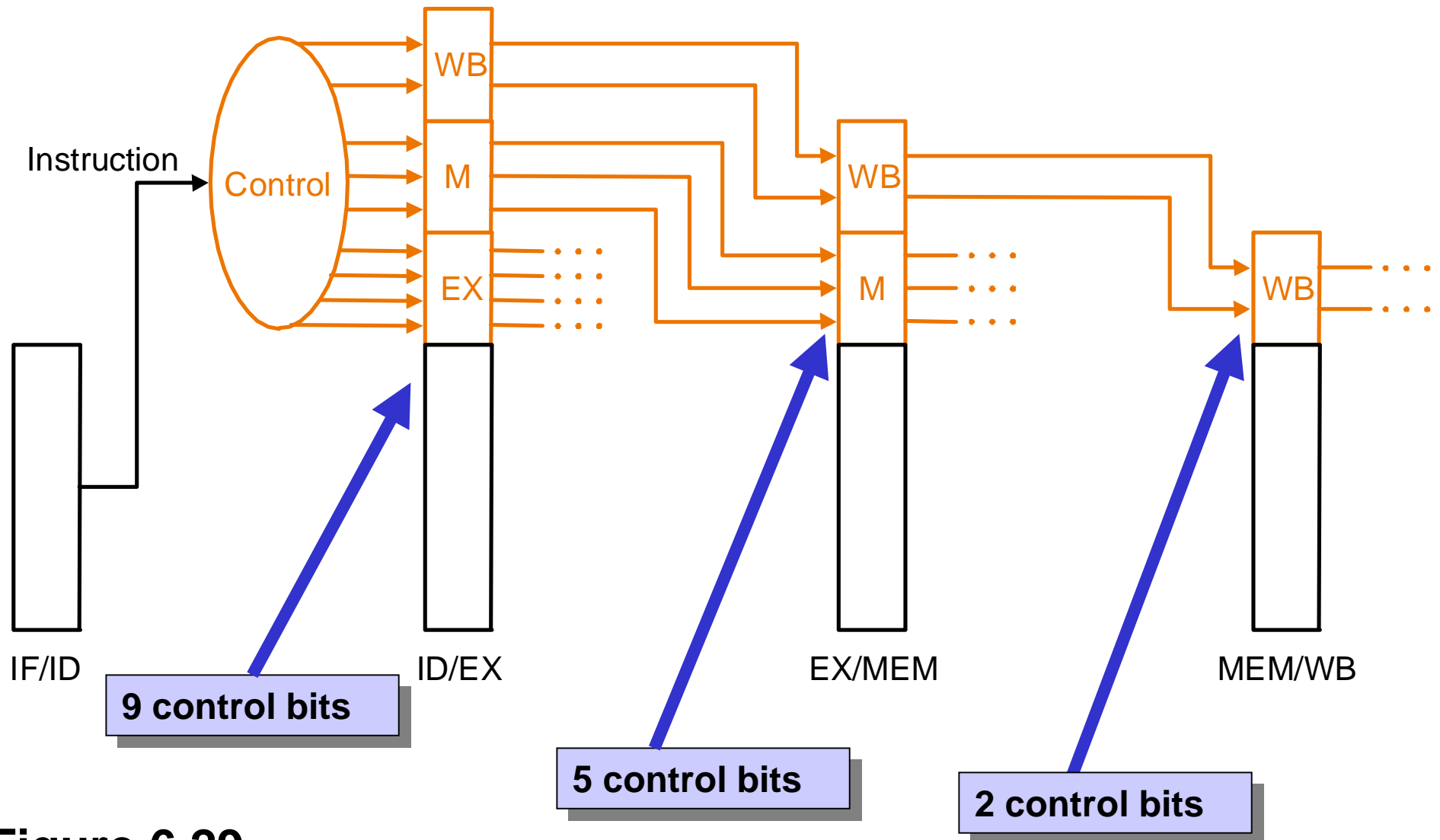


Figure 6.29

Pipeline Control: Controlpath Register bits

Figure 5.20, Single Cycle

Instruction	Reg Dst	ALU Src	Mem Reg	Reg Wrt	Mem Red	Mem Wrt	Bra-nch	ALU op1	ALU op0
R-format	1	0	0	1	0	0	0	1	0
lw	1	1	1	1	1	0	0	0	0
sw	X	1	X	0	0	1	0	0	0
beq	X	0	X	0	0	0	1	0	1

Figure 6.28

Instruction	ID / EX control lines				EX / MEM control lines			MEM / WB cntrl lines	
	Reg Dst	ALU Op1	ALU Op0	ALU Src	Bra-nch	Mem Red	Mem Wrt	Reg Wrt	Mem Reg
R-format	1	1	0	0	0	0	0	1	0
lw	1	0	0	1	0	1	0	1	1
sw	X	0	0	1	0	0	1	0	X
beq	X	0	1	0	1	0	0	0	X

Pipeline Hazards



Pipeline hazards

- **Solution #1 always works: stall, delay & procrastinate!**

Structural Hazards (i.e. fetching same memory bank)

- **Solution #2: partition architecture**

Control Hazards (i.e. branching)

- **Solution #1: stall! but decreases throughput**
- **Solution #2: guess and back-track**
- **Solution #3: delayed decision: delay branch & fill slot**

Data Hazards (i.e. register dependencies)

- **Worst case situation**
- **Solution #2: re-order instructions**
- **Solution #3: forwarding or bypassing: delayed load**

Pipeline Datapath and Controlpath

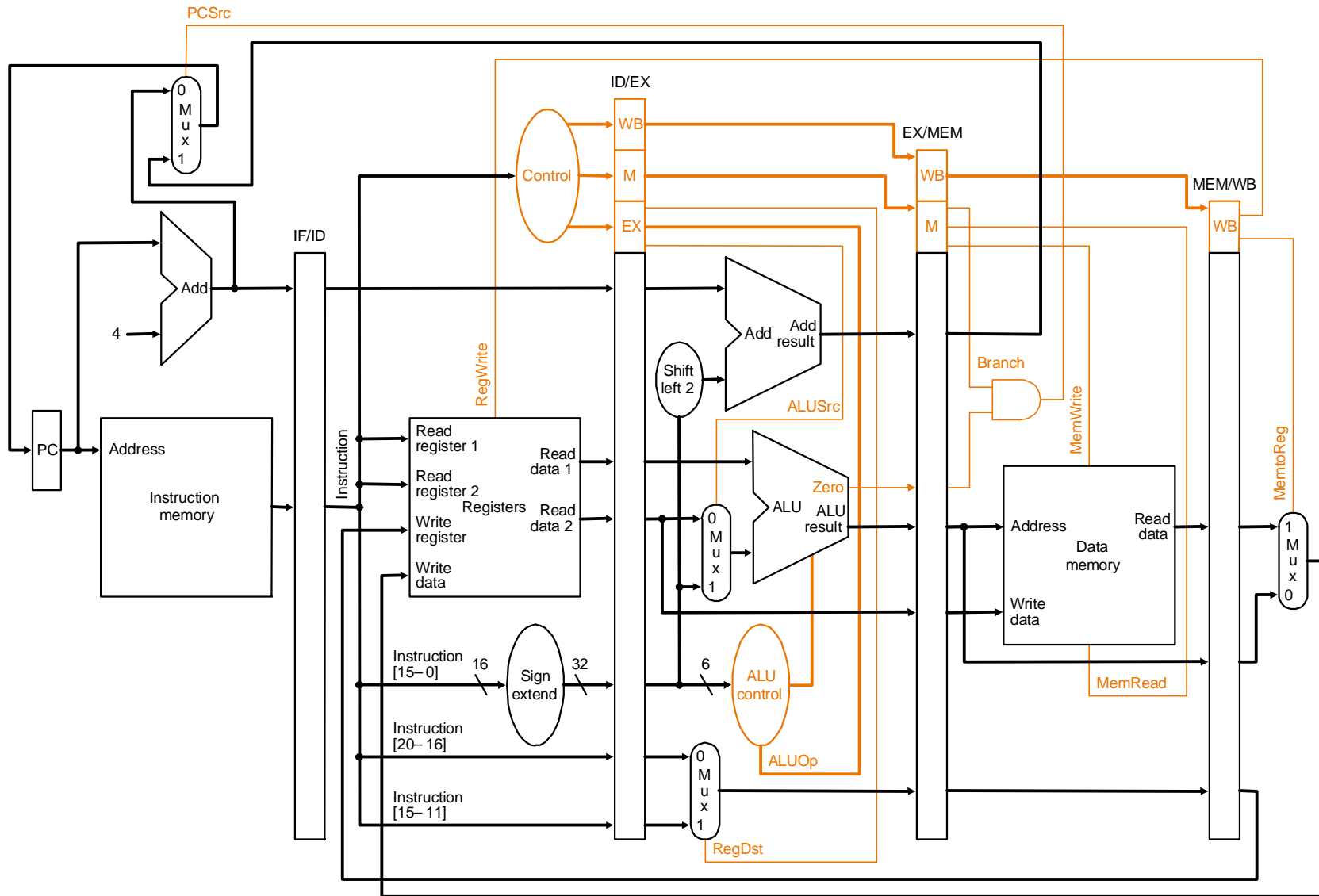


Figure 6.30

load inst.

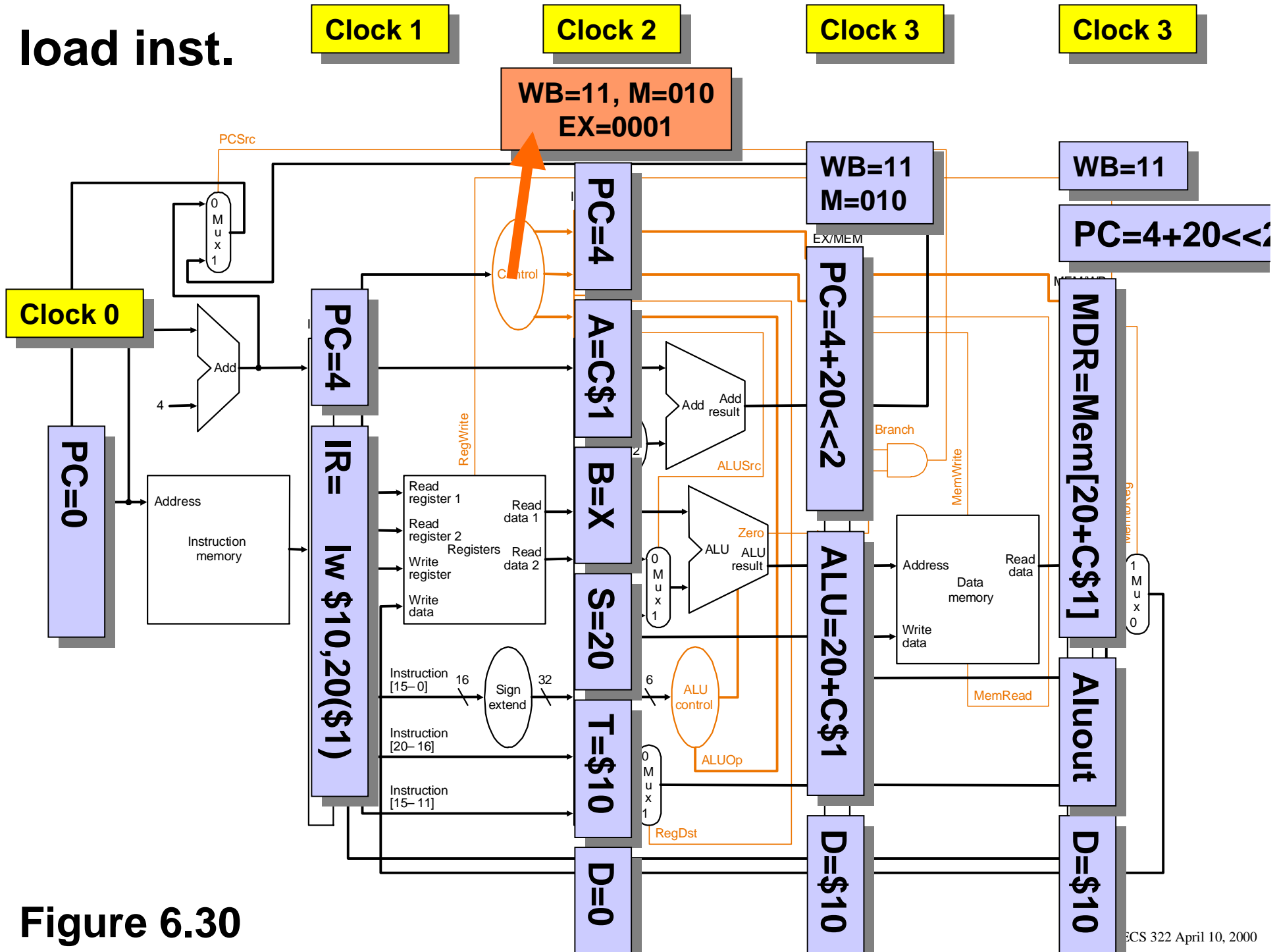


Figure 6.30

load inst.

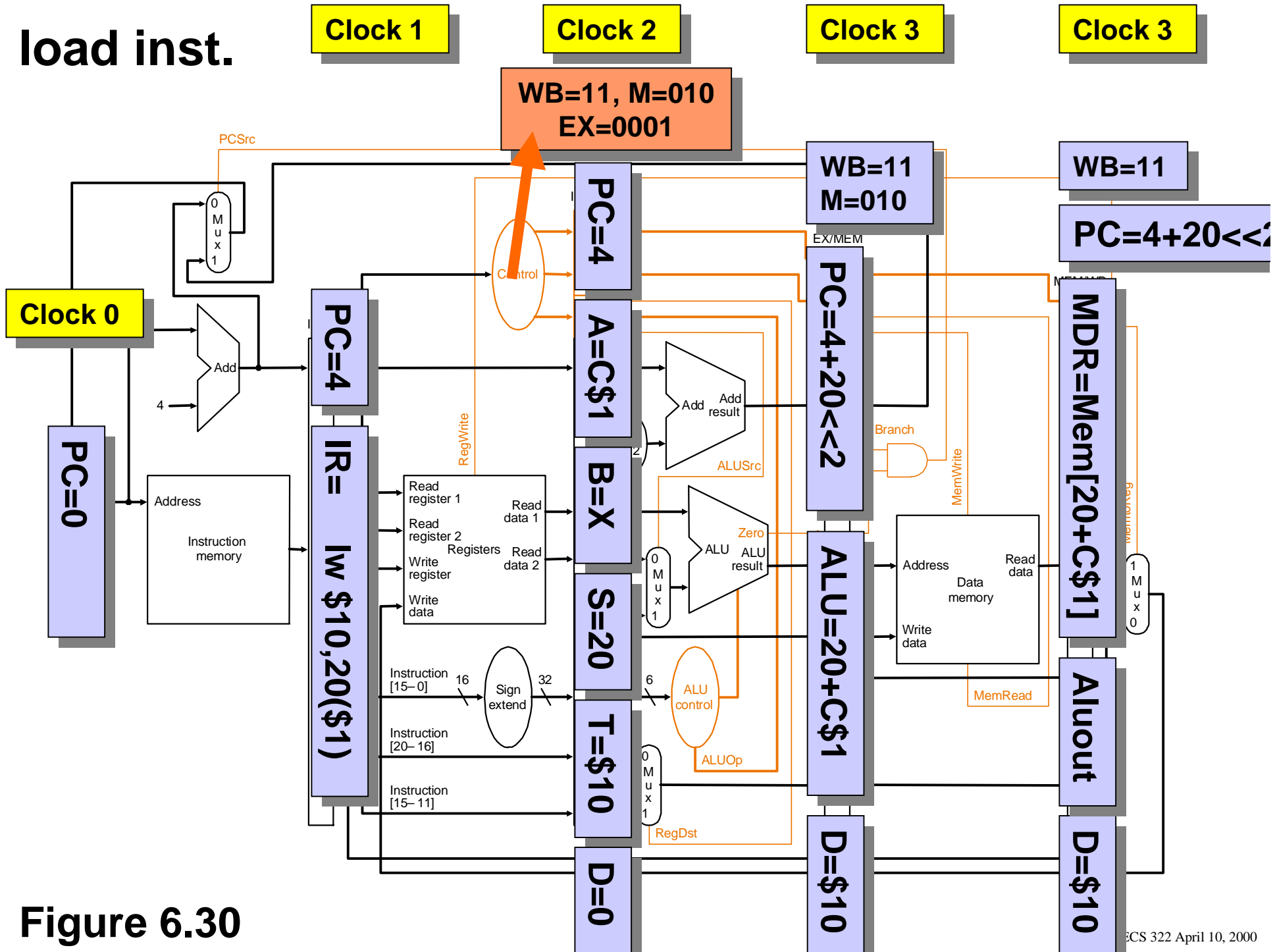


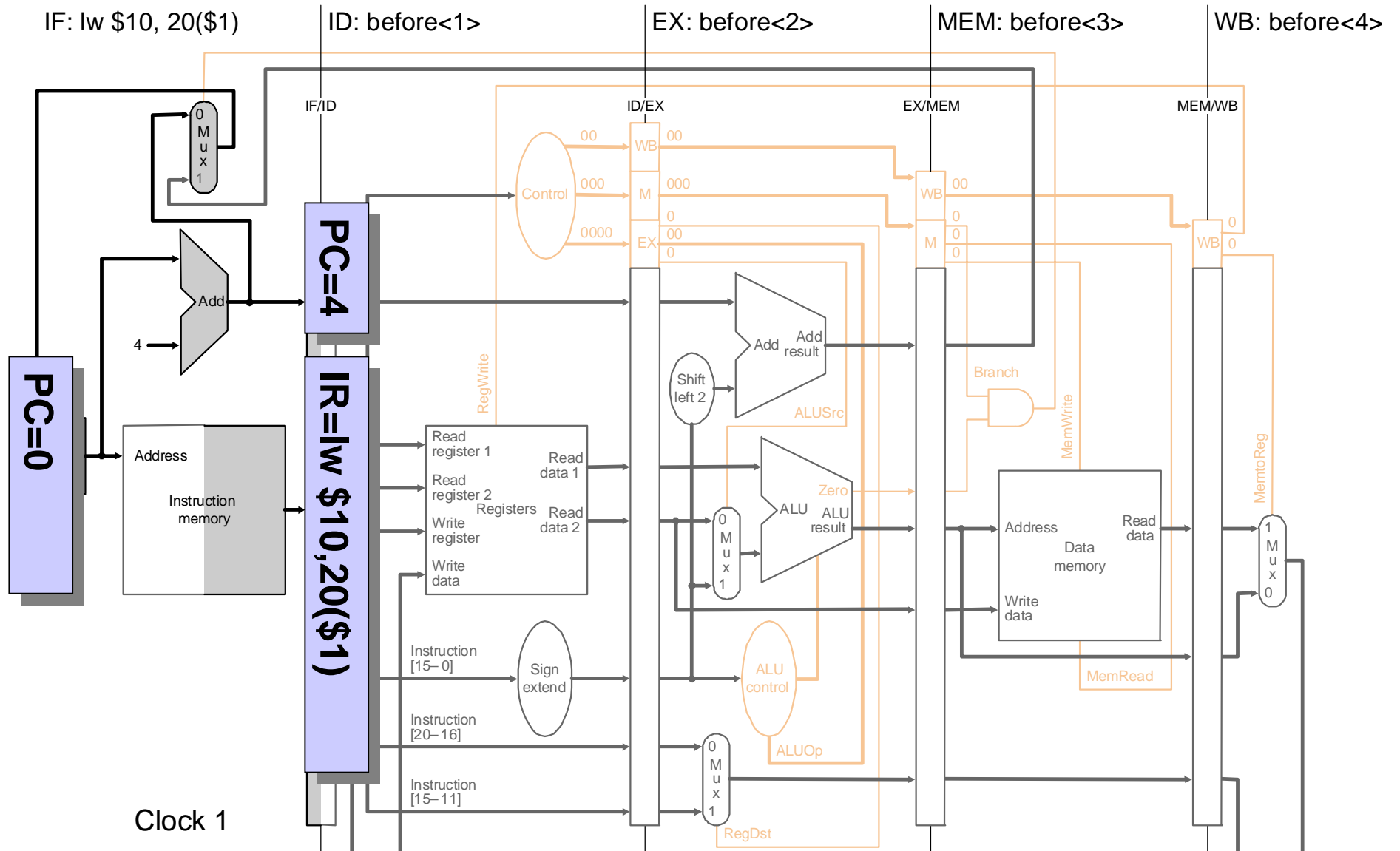
Figure 6.30

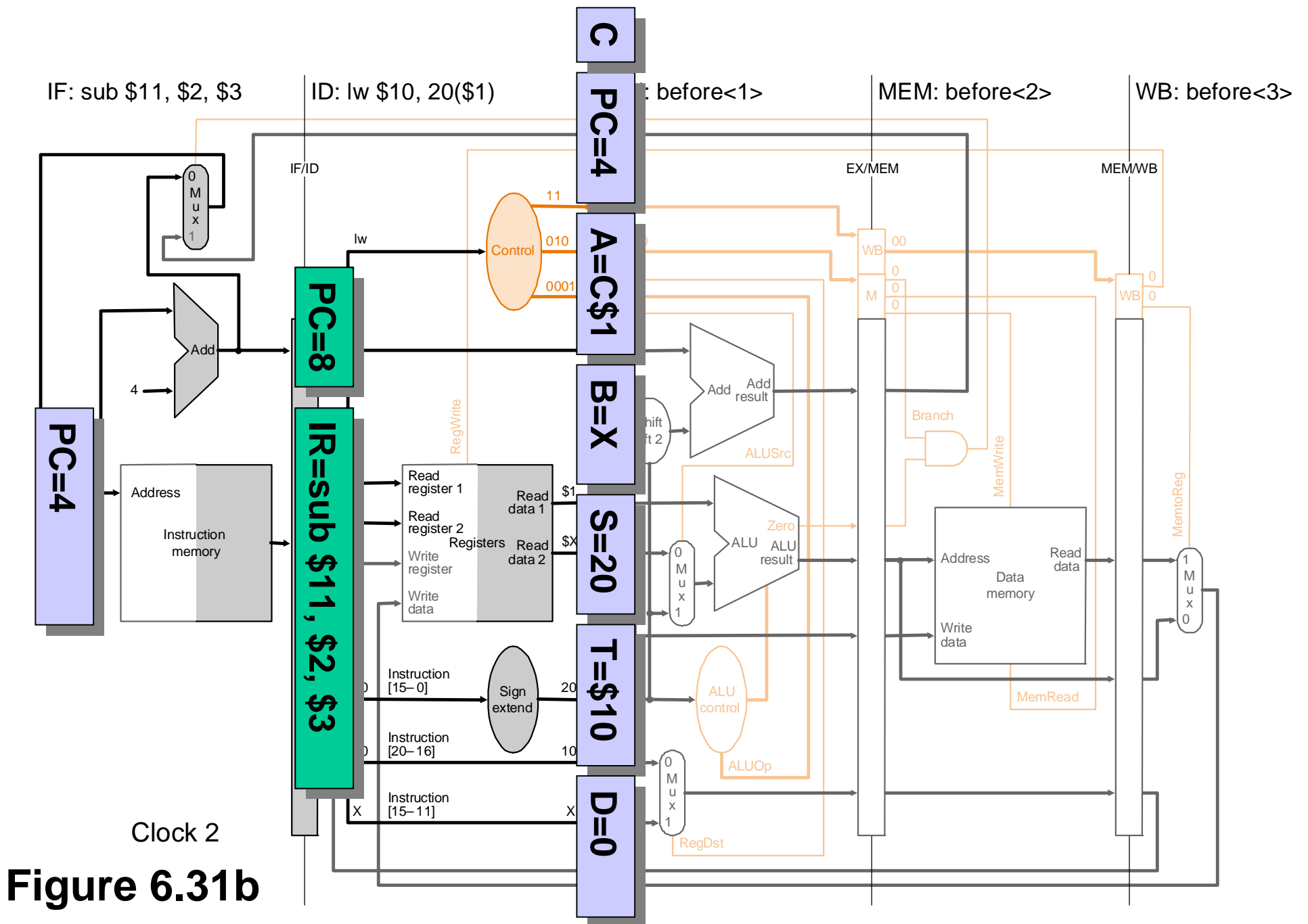
Pipeline Datapath and Controlpath

Contents of Register 1 = C\$1 = 3; C\$2=4; C\$3=4; C\$4=6; C\$5=7; C\$10=8; ... Memory[23]=9;
 Formats: add \$rd,\$rs=A,\$rt=B; lw \$rt=B,@(\$rs=A)

Clock	<IF/ID>	<ID/EX>	<EX/MEM>	<MEM/WB>
	<PC, IR>	<PC, A, B, S, Rt, Rd>	<PC, Z, ALU, B, R>	<MDR, ALU, R>
0	<0,?>	<?,?,?,?,??>	<?,?,?,?,??>	<?,?,?>
1	<4,lw \$10,20(\$1)>	<0?,?,?,?,??>	<?,?,?,?,??>	<?,?,?>
2	<8,sub \$11,\$2,\$3>	<4,C\$1→3,C\$10→8,20,\$10,0>	<0?,?,?,?,??>	<?,?,?>
3	<12,and \$12,\$4,\$5>	<8,C\$2→4,C\$3→4,X,\$3,\$11>	<4+20<2→84,0,20+3→23,8,\$10>	<?,?,?>
4	<16,or \$13,\$6,\$7>	<12,C\$4→6,C\$5→7,X,\$5,\$12>	<X,1,4-4=0,4,\$11>	<Mem[23]→9,23,\$10>
5	<20,add \$14,\$8,\$9>	<16,C\$6 ,C\$7,X,\$7,\$13>	<X,0,1,7,\$12>	<X,0,\$11>

Clock 1: Figure 6.31a





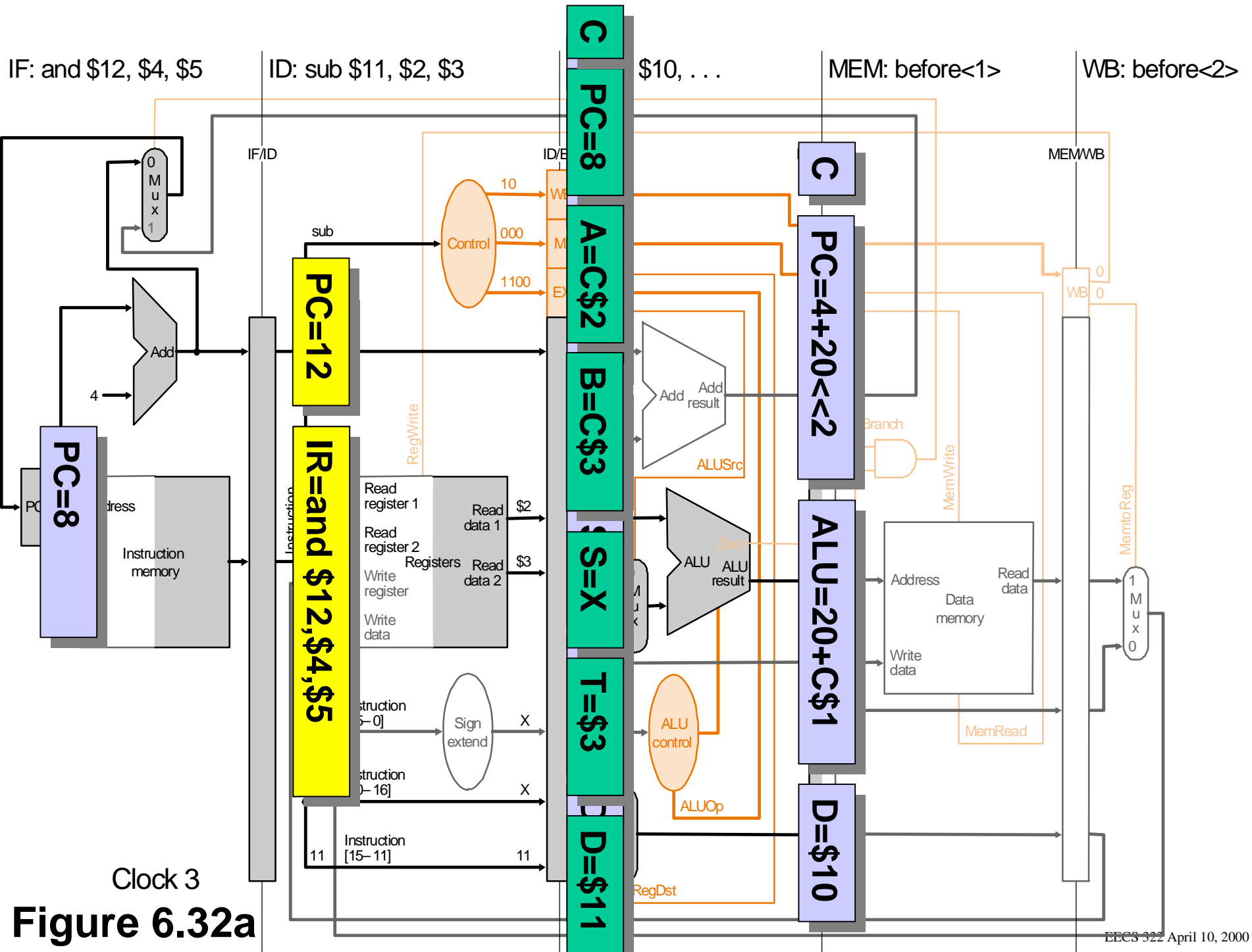
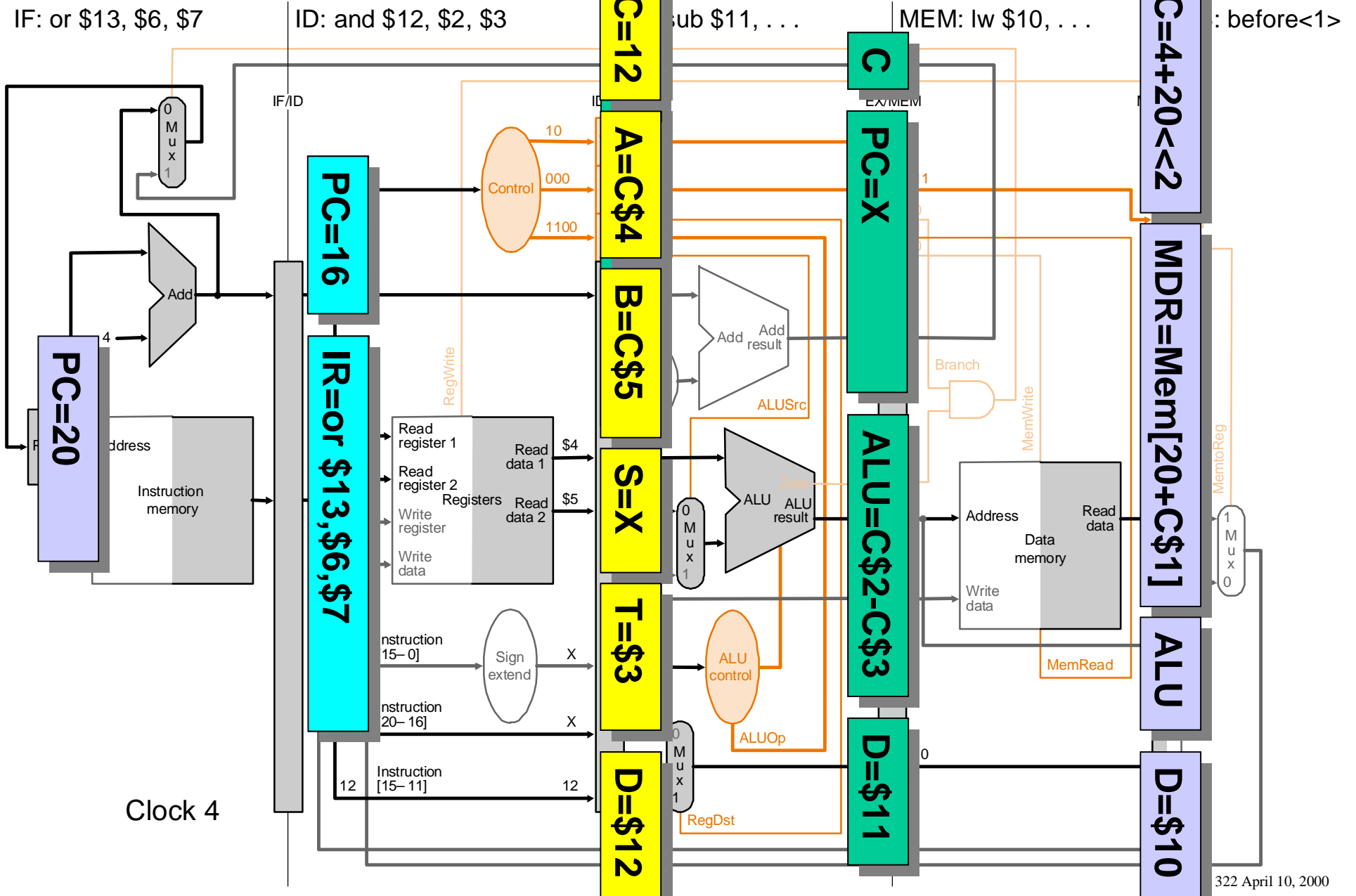


Figure 6.32a

Clock 4: Figure 6.32b



Data Dependencies: that can be resolved by forwarding

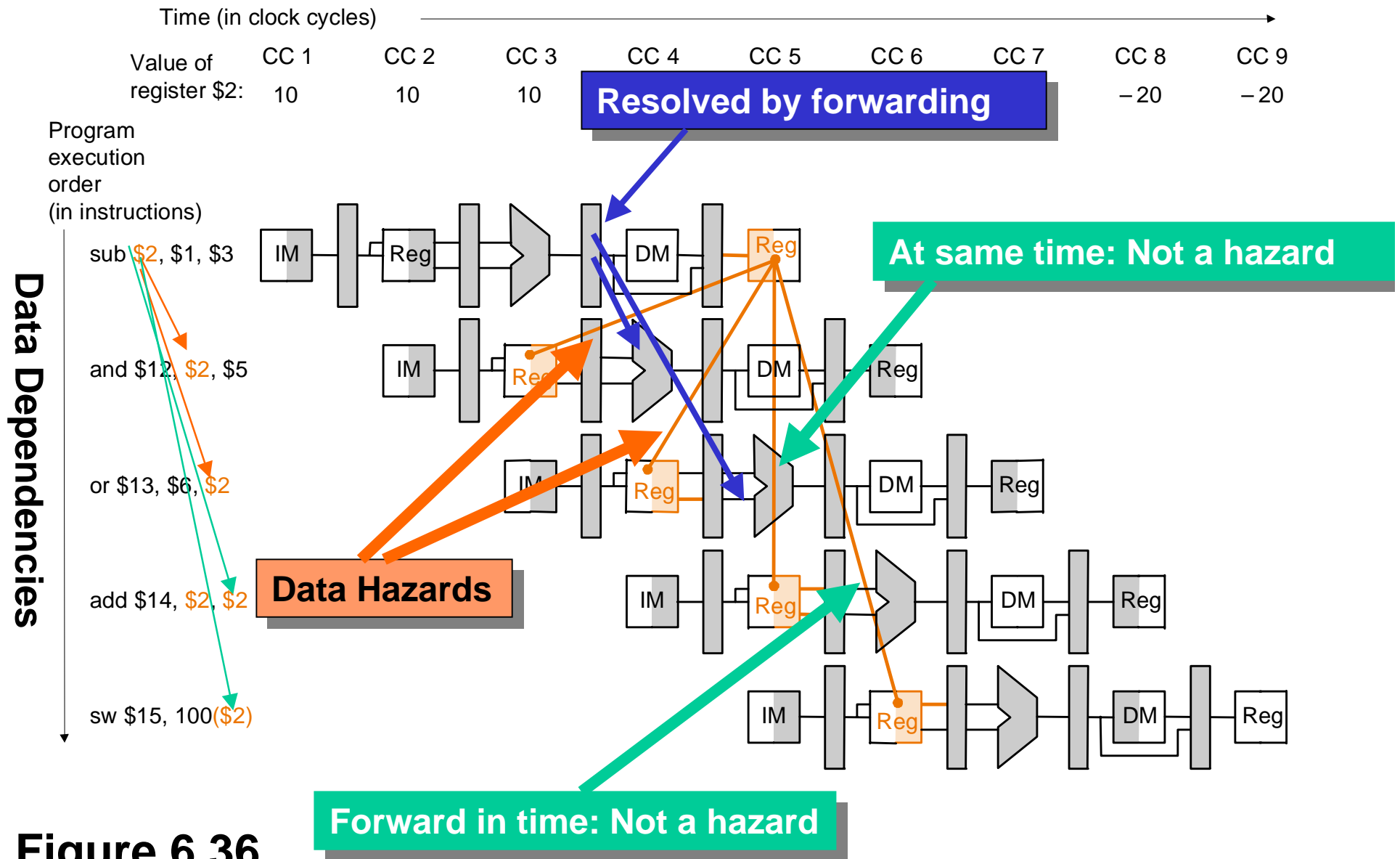


Figure 6.36

Data Hazards: arithmetic

	CC 1	CC 2	CC 3	CC 4	CC 5	CC 6	CC 7	CC 8	CC 9
Value of register \$2 :	10	10	10	10	10/-20	-20	-20	-20	-20
Value of EX/MEM :	X	X	X	-20	X	X	X	X	X
Value of MEM/WB :	X	X	X						

Forwards in time: Can be resolved

At same time: Not a hazard

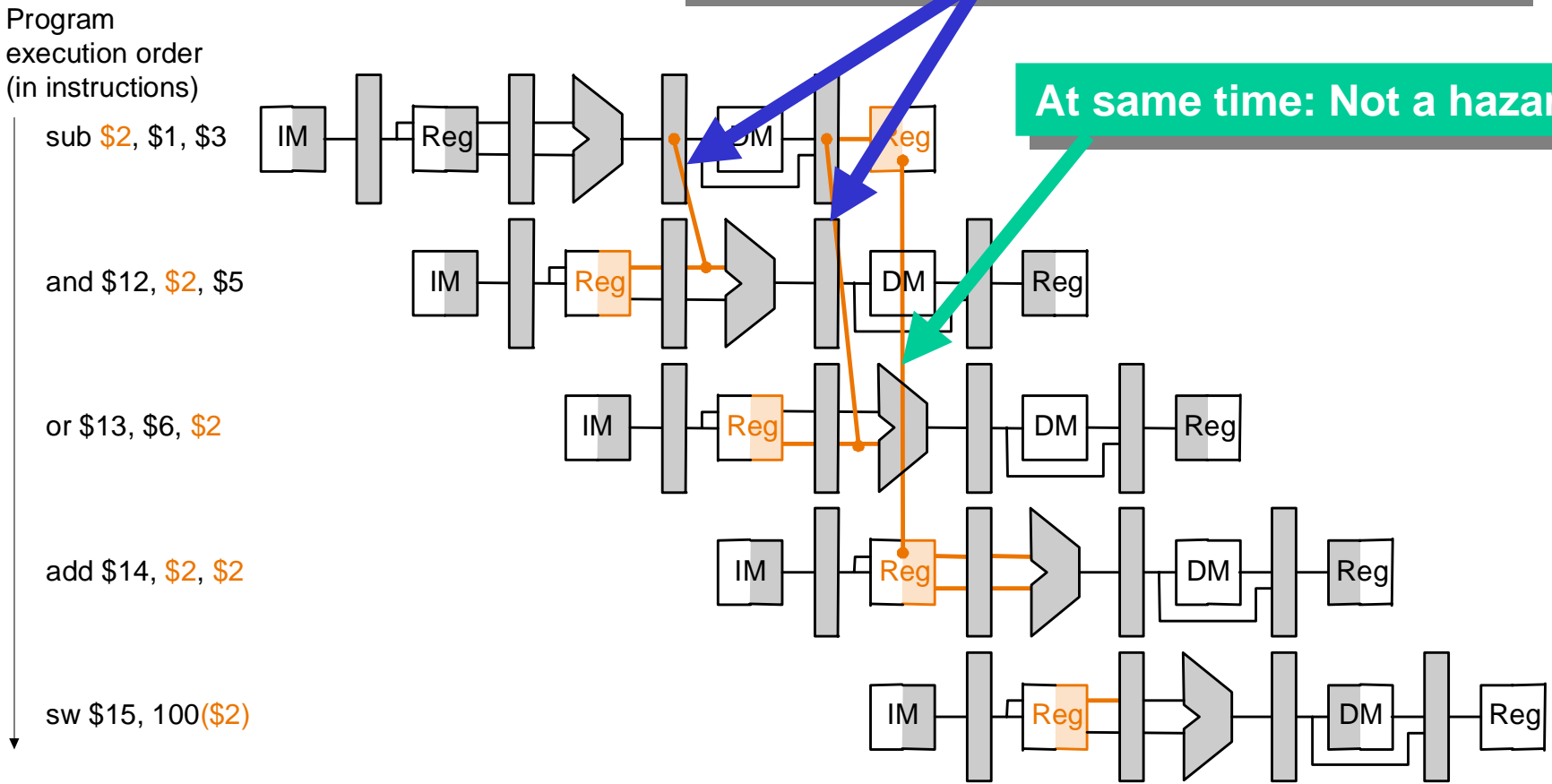
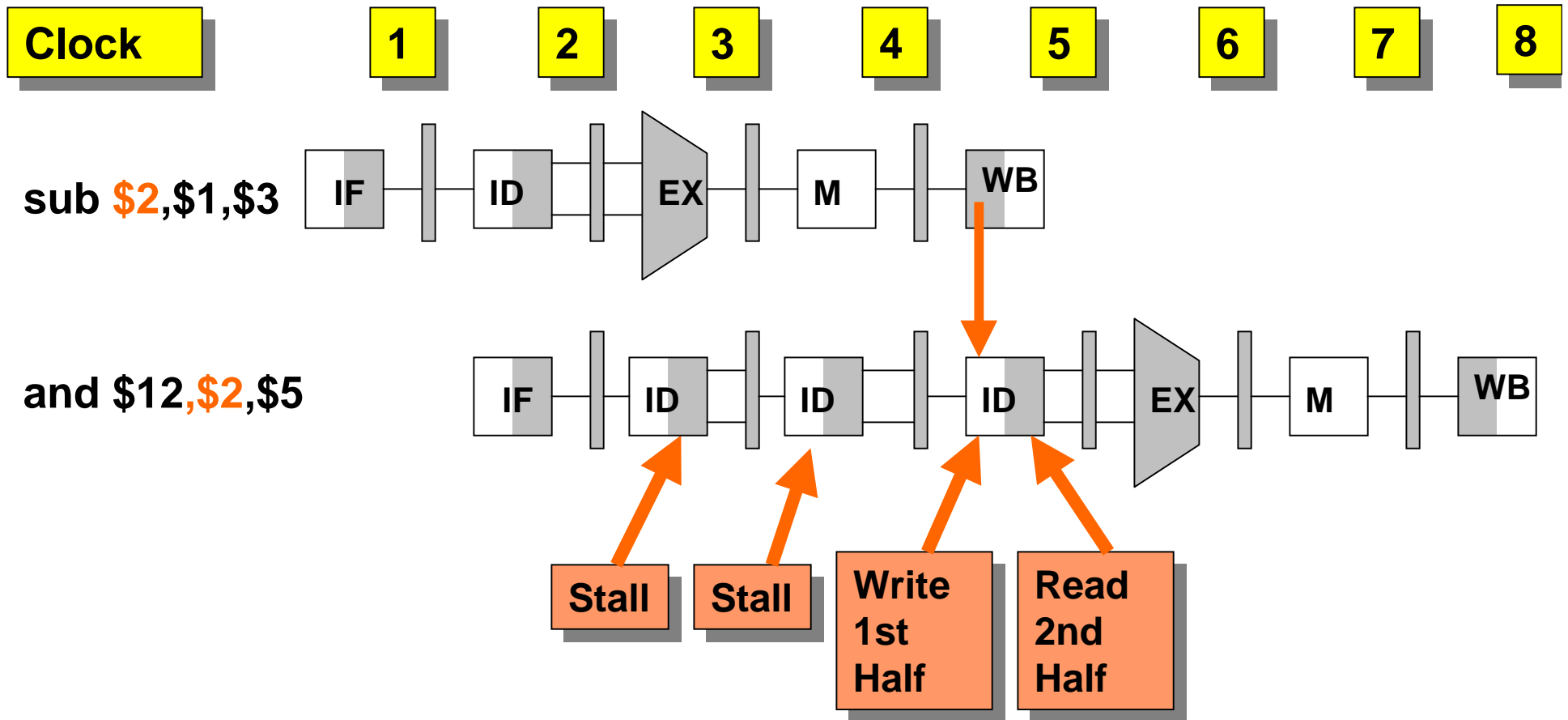


Figure 6.37

Data Dependencies: no forwarding



Suppose every instruction is dependant = 1 + 2 stalls = 3 clocks

$$\text{MIPS} = \frac{\text{Clock}}{\text{CPI}} = \frac{500 \text{ Mhz}}{3} = 167 \text{ MIPS}$$

Data Dependencies: no forwarding

A dependant instruction will take = 1 + 2 stalls = 3 clocks

An independent instruction will take = 1 + 0 stalls = 1 clocks

Suppose 10% of the time the instructions are dependant?

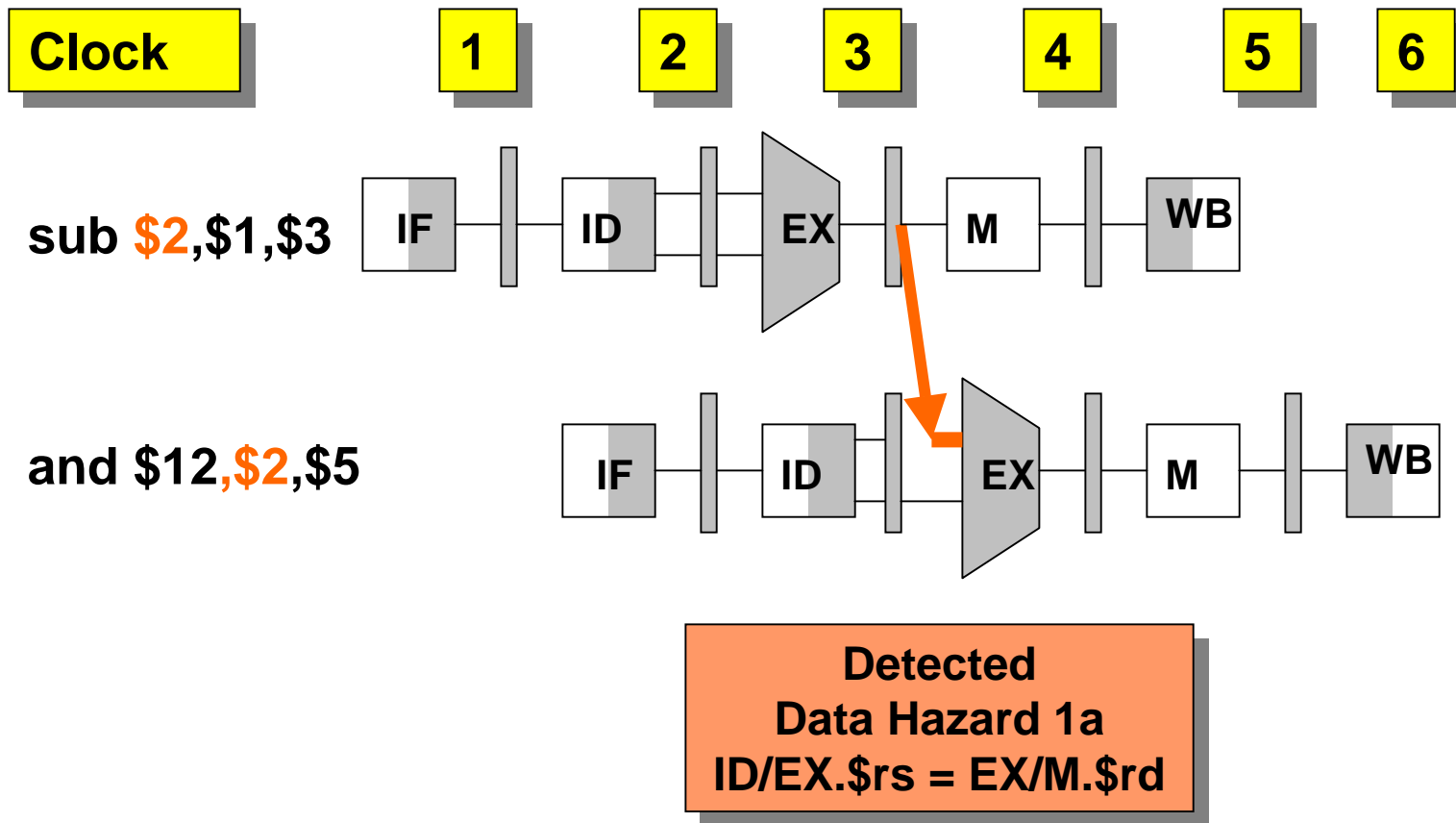
Average instruction time = $10\% \cdot 3 + 90\% \cdot 1 = 0.10 \cdot 3 + 0.90 \cdot 1 = 1.2$ clocks

$$\text{MIPS} = \frac{\text{Clock}}{\text{CPI}} = \frac{500 \text{ Mhz}}{1.2} = 417 \text{ MIPS (10\% dependency)}$$

$$\text{MIPS} = \frac{\text{Clock}}{\text{CPI}} = \frac{500 \text{ Mhz}}{3} = 167 \text{ MIPS (100\% dependency)}$$

$$\text{MIPS} = \frac{\text{Clock}}{\text{CPI}} = \frac{500 \text{ Mhz}}{1} = 500 \text{ MIPS (0\% dependency)}$$

Data Dependencies: with forwarding



Suppose every instruction is dependant = 1 + 0 stalls = 1 clock

$$\text{MIPS} = \frac{\text{Clock}}{\text{CPI}} = \frac{500 \text{ Mhz}}{1} = 500 \text{ MIPS}$$

Data Dependencies: Hazard Conditions

Data Hazard Condition

occurs whenever a data **source** needs a previous unavailable result due to a data **destination**.

Example

sub **\$2**, \$1, \$3
and \$12, **\$2**, \$5

sub **\$rd**, \$rs, \$rt
and \$rd, **\$rs**, \$rt

Data Hazard Detection

is always comparing a **destination** with a **source**.

<u>Destination</u>	<u>Source</u>	<u>Hazard Type</u>
EX/MEM.\$rdest =	ID/EX.\$rs	1a.
	ID/EX.\$rt	1b.
MEM/WB.\$rdest =	ID/EX.\$rs	2a.
	ID/EX.\$rt	2b.

Data Dependencies: Hazard Conditions

1a Data Hazard:

sub **\$2**, \$1, \$3
and \$12, **\$2**, \$5

1b Data Hazard:

sub **\$2**, \$1, \$3
and \$12, \$1, **\$2**

2a Data Hazard:

sub **\$2**, \$1, \$3
and \$12, \$1, \$5
or \$13, **\$2**, \$1

2b Data Hazard:

sub **\$2**, \$1, \$3
and \$12, \$1, \$5
or \$13, \$6, **\$2**

EX/MEM.\$rd = ID/EX.\$rs

sub **\$rd**, \$rs, \$rt
and \$rd, **\$rs**, \$rt

EX/MEM.\$rd = ID/EX.\$rt

sub **\$rd**, \$rs, \$rt
and \$rd, \$rs, **\$rt**

MEM/WB.\$rd = ID/EX.\$rs

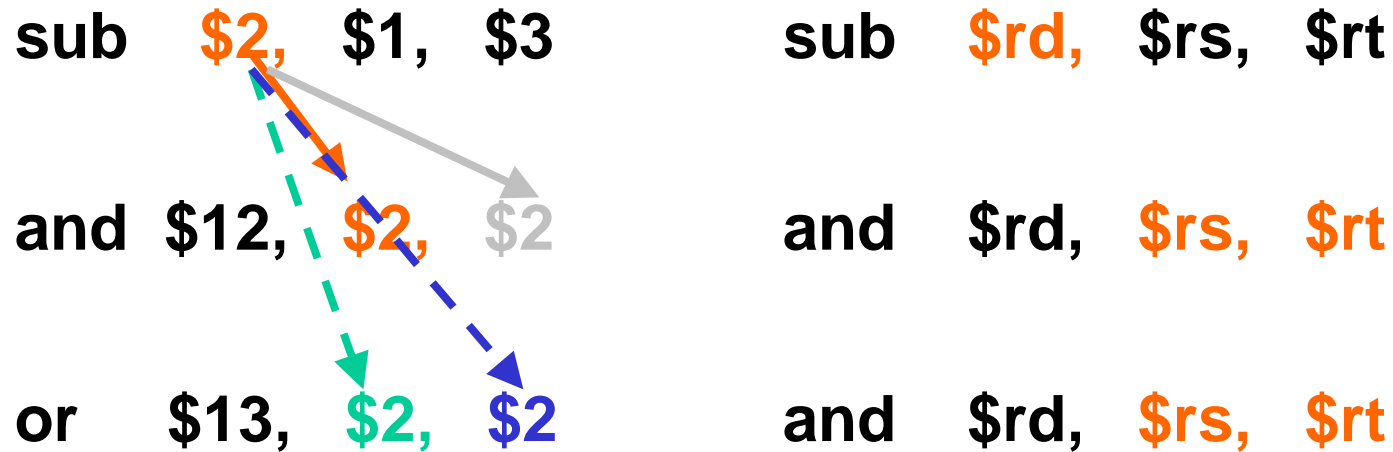
sub **\$rd**, \$rs, \$rt
sub \$rd, \$rs, \$rt
and \$rd, **\$rs**, \$rt

MEM/WB.\$rd = ID/EX.\$rt

sub **\$rd**, \$rs, \$rt
sub \$rd, \$rs, \$rt
and \$rd, \$rs, **\$rt**

Data Dependencies: Worst case

Data Hazard:



Data Hazard 1a: **EX/MEM.\$rd = ID/EX.\$rs**

Data Hazard 1b: EX/MEM.\$rd = ID/EX.\$rt

Data Hazard 2a: **MEM/WB.\$rd = ID/EX.\$rs**

Data Hazard 2b: **MEM/WB.\$rd = ID/EX.\$rt**

Data Dependencies: Hazard Conditions

Hazard

Type

Source

Destination

1a.

ID/EX.\$rs

1b.

ID/EX.\$rt

} = EX/MEM.\$rdest

2a.

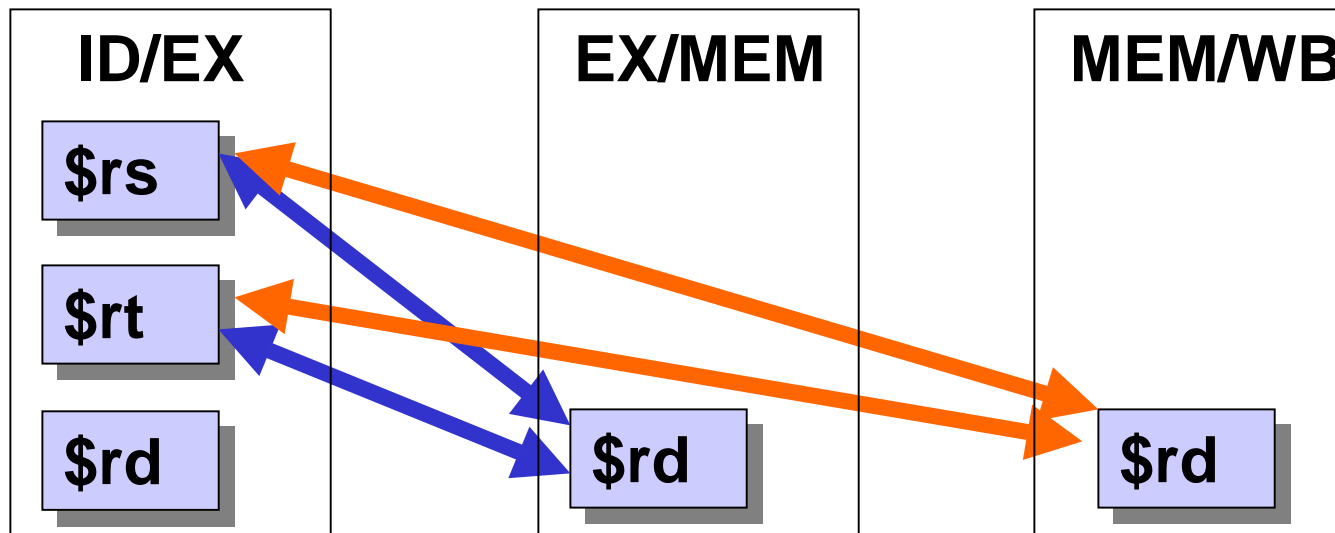
ID/EX.\$rs

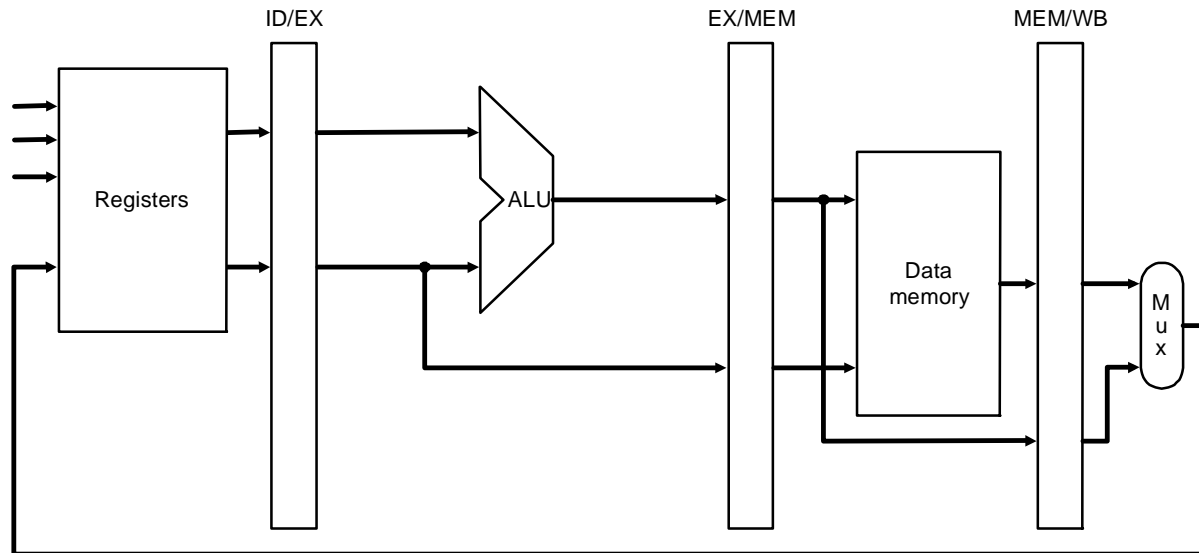
2b.

ID/EX.\$rt

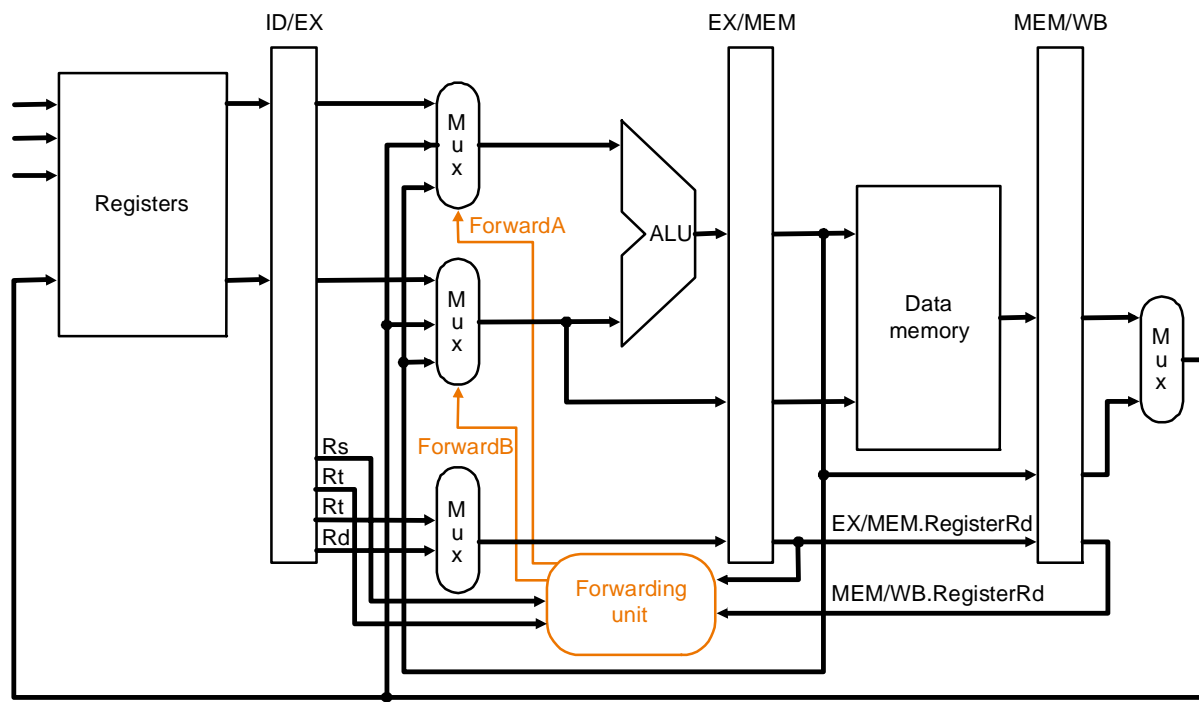
} = MEM/WB.\$rdest

Pipeline Registers





a. No forwarding



b. With forwarding

Figure 6.38

Data Hazards: Loads

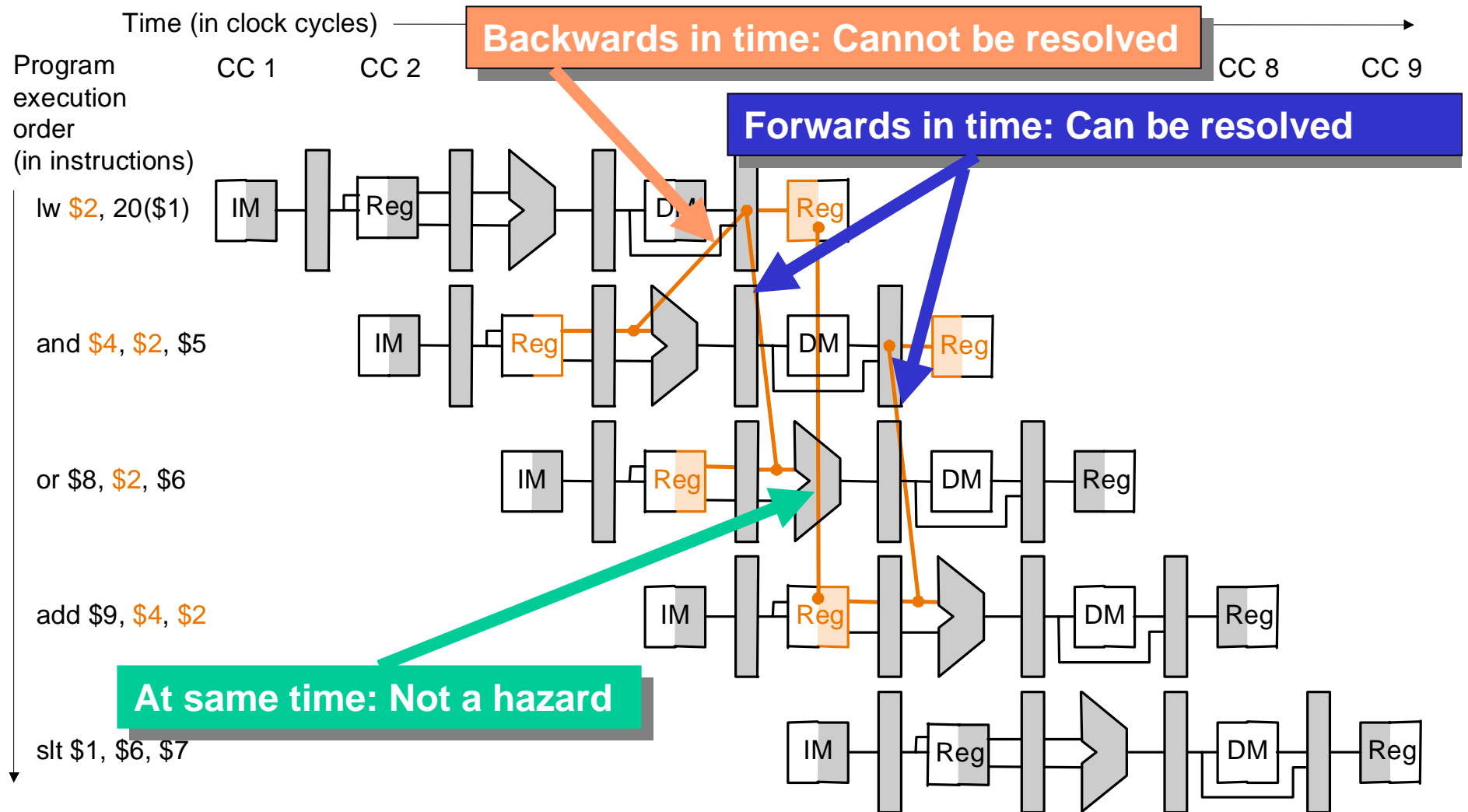


Figure 6.44

Data Hazards: load stalling

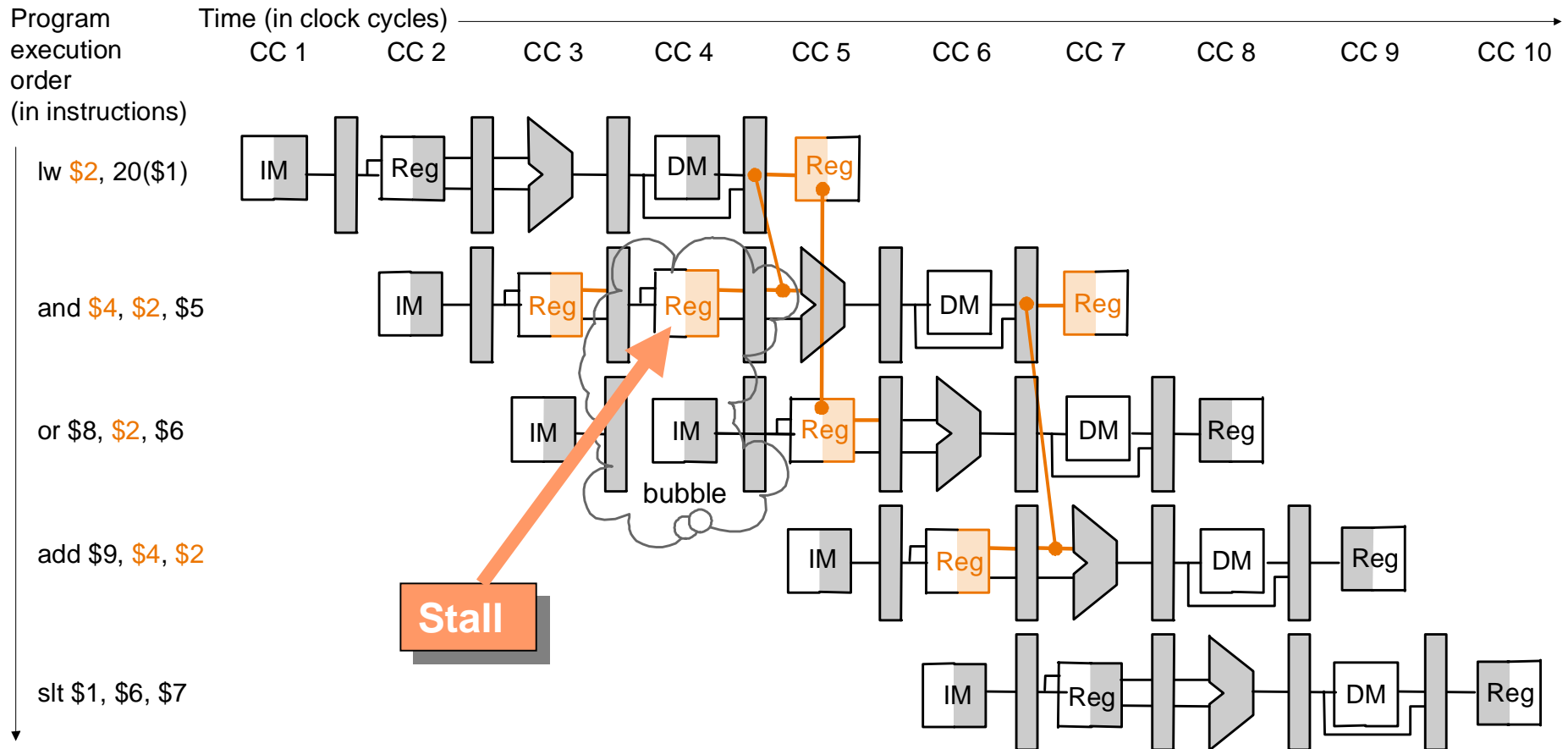


Figure 6.45

Data Hazards: Hazard detection unit (page 490)

Stall Condition

Source

IF/ID.\$rs

IF/ID.\$rt

Destination

} = ID/EX.\$rt \wedge ID/EX.MemRead=1

Stall Example

lw \$2, 20(\$1)

and \$4, \$2, \$5

lw \$rt, addr(\$rs)

and \$rd, \$rs, \$rt

No Stall Example: (only need to look at next instruction)

lw \$2, 20(\$1)

and \$4, \$1, \$5

or \$8, \$2, \$6

lw \$rt, addr(\$rs)

and \$rd, \$rs, \$rt

or \$rd, \$rs, \$rt

Data Hazards: Hazard detection unit (page 490)

No Stall Example: (only need to look at next instruction)

lw \$2, 20(\$1)
and \$4, \$1, \$5
or \$8, \$2, \$6

lw \$rt, addr(\$rs)
and \$rd, \$rs, \$rt
or \$rd, \$rs, \$rt

Example

load: assume half of the instructions are immediately followed by an instruction that uses it.

What is the average number of clocks for the load?

load instruction time: $50\% * (1 \text{ clock}) + 50\% * (2 \text{ clocks}) = 1.5$

Hazard Detection Unit: when to stall

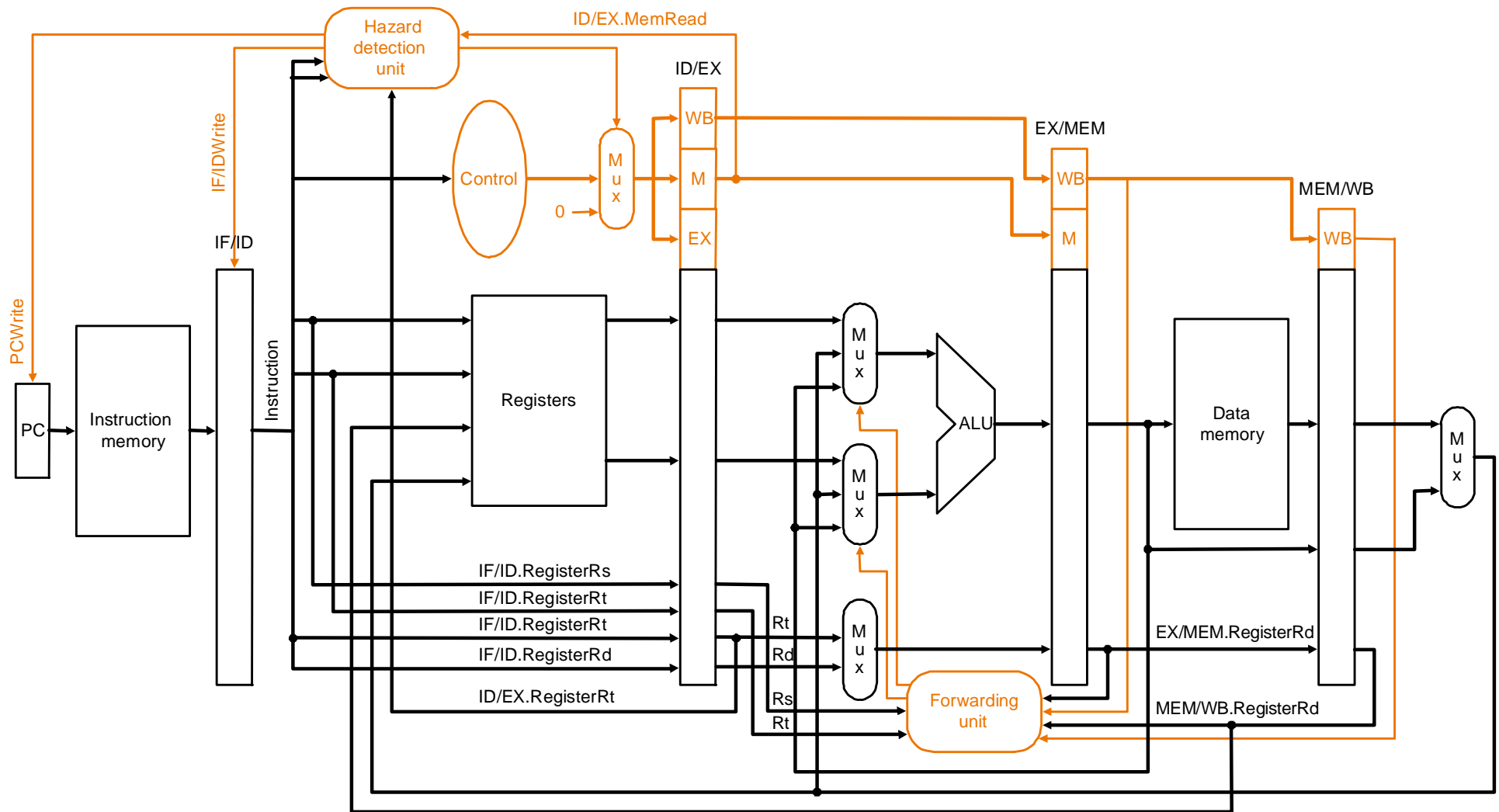


Figure 6.46

Data Dependency Units

Forwarding Condition

Source

ID/EX.\$rs

ID/EX.\$rt

ID/EX.\$rs

ID/EX.\$rt

Destination

} = EX/MEM.\$rd

} = MEM/WB.\$rd

Stall Condition

Source

IF/ID.\$rs

IF/ID.\$rt

Destination

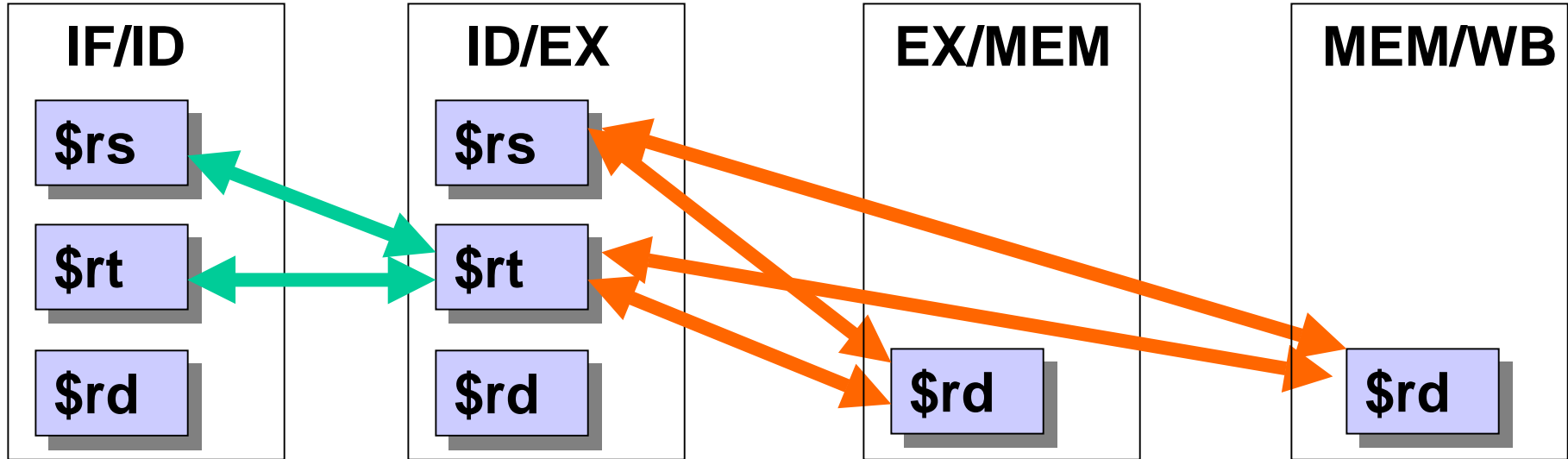
} = ID/EX.\$rt \wedge ID/EX.MemRead=1

Data Dependency Units

Pipeline Registers

Stalling Comparisons

Forwarding Comparisons



Stall Condition

Source

IF/ID.\$rs

IF/ID.\$rt

Destination

$$\left. \begin{array}{l} \text{IF/ID.\$rs} \\ \text{IF/ID.\$rt} \end{array} \right\} = \text{ID/EX.\$rt} \wedge \text{ID/EX.MemRead}=1$$

Branch Hazards: Soln #1, Stall until Decision made (fig. 6.4)

@3C: add \$4, \$5, \$6

@40: beq \$1, \$3, 7

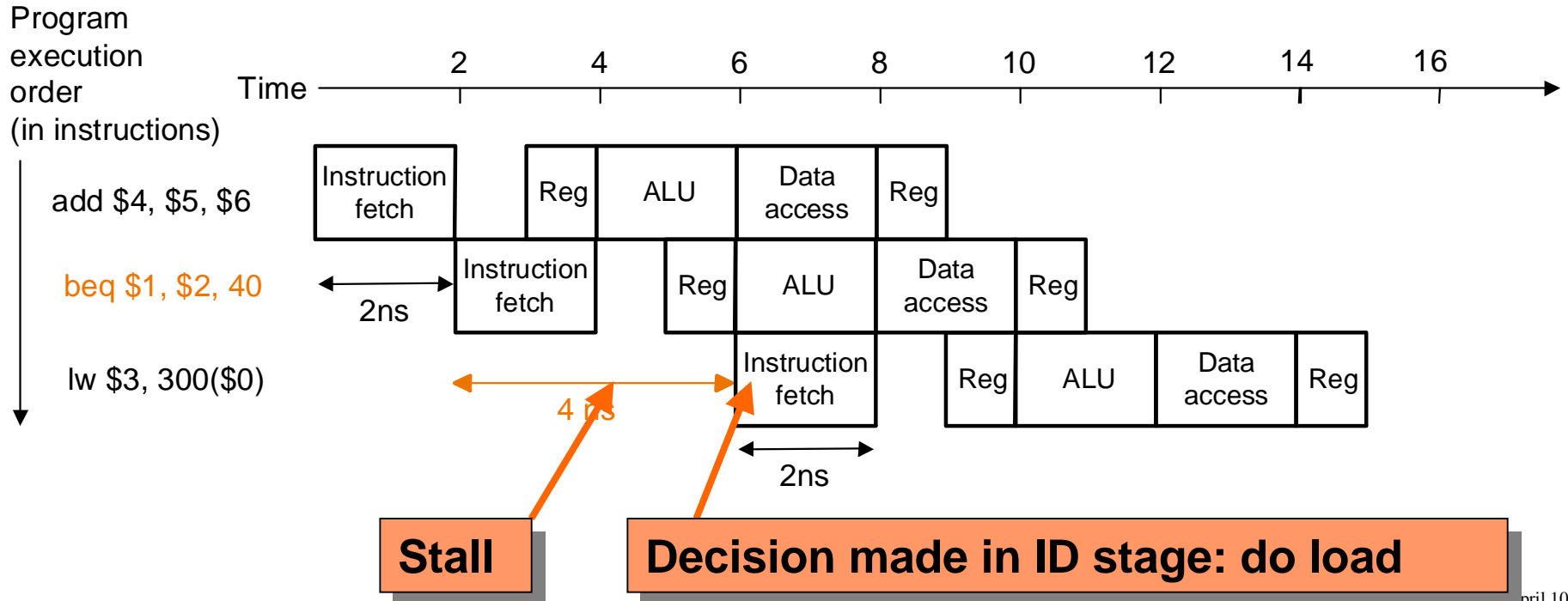
@44: and \$12, \$2, \$5

@48: or \$13, \$6, \$2

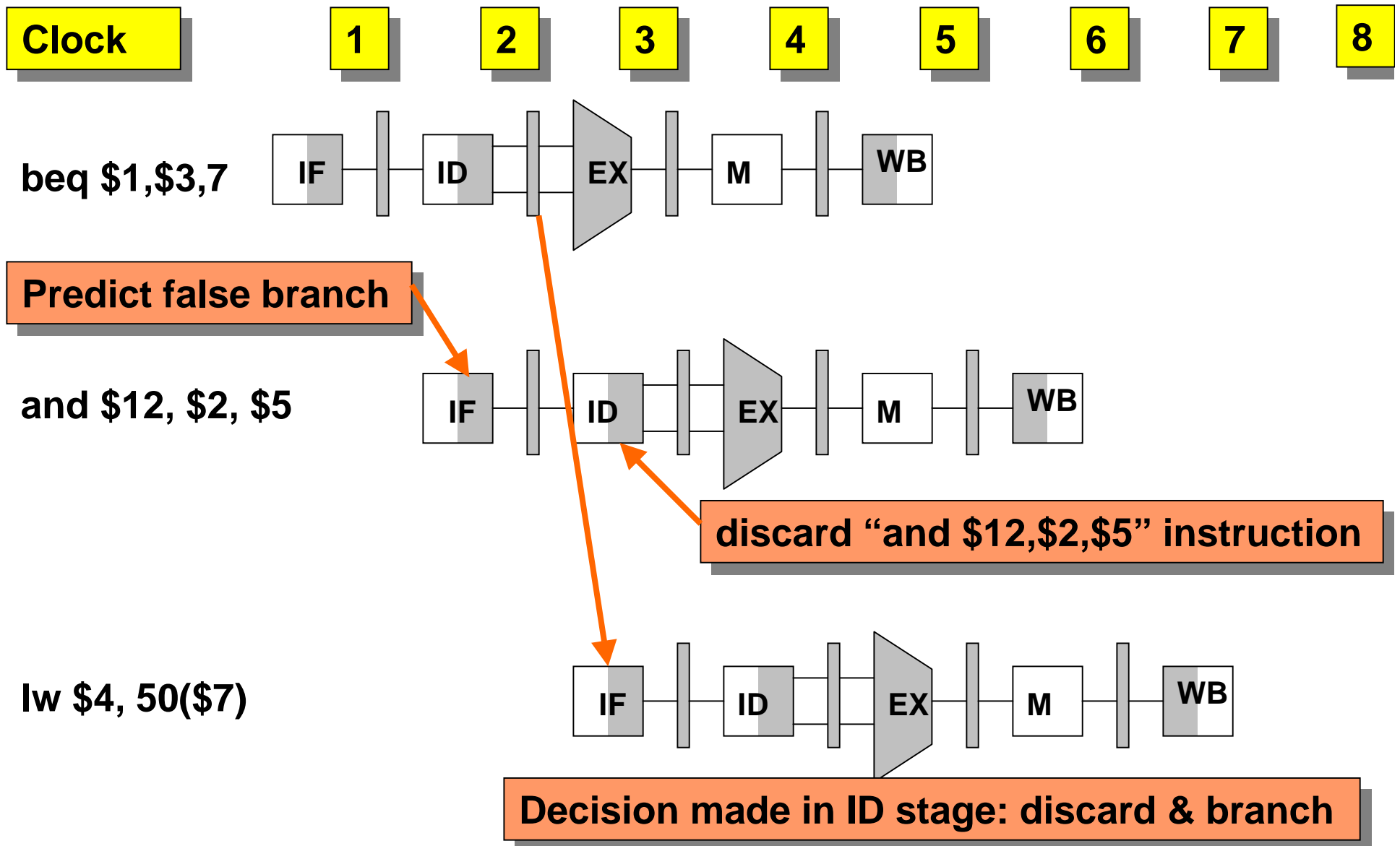
@4C: add \$14, \$2, \$2

@50: lw \$4, 50(\$7)

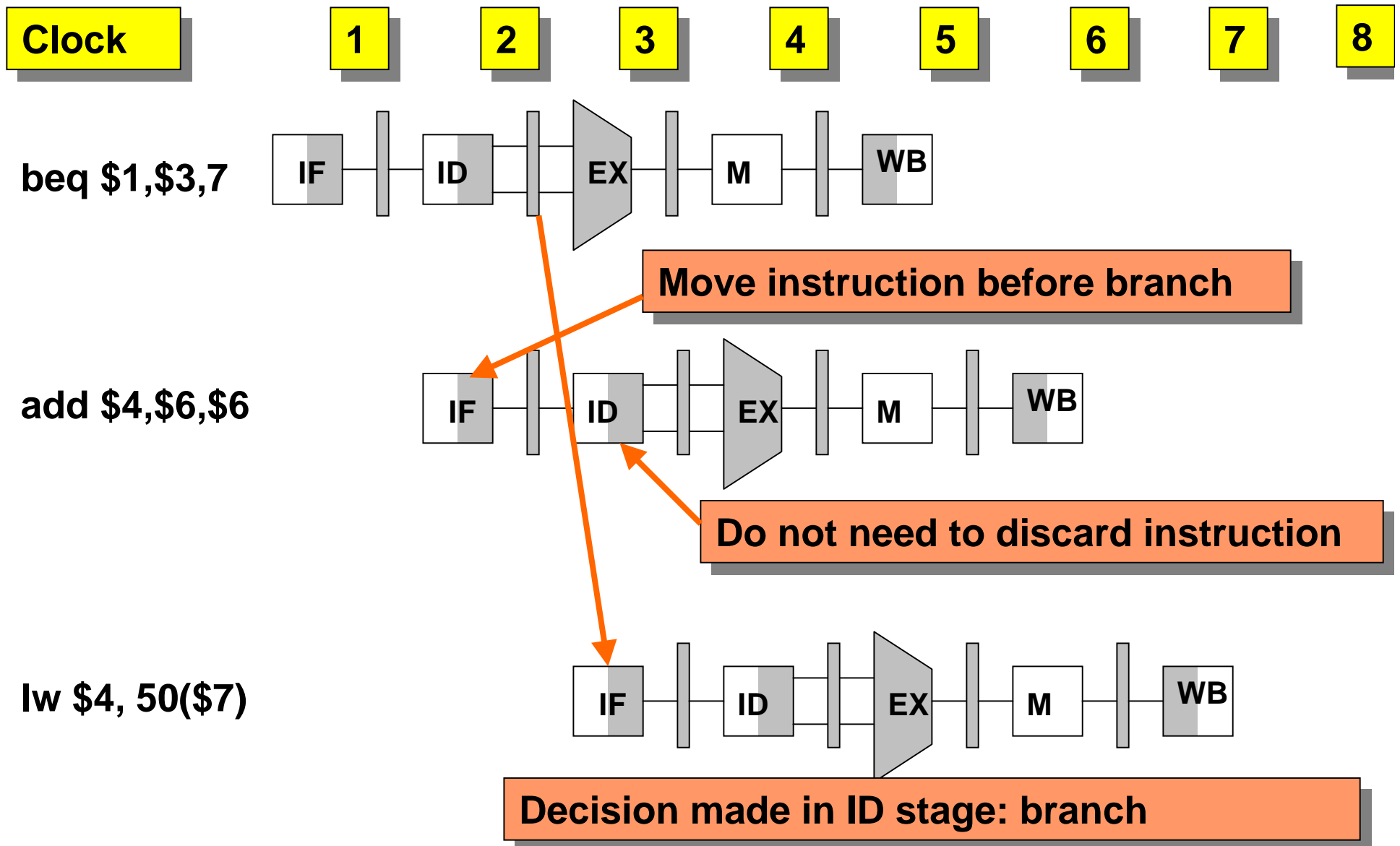
Soln #1: Stall until Decision is made



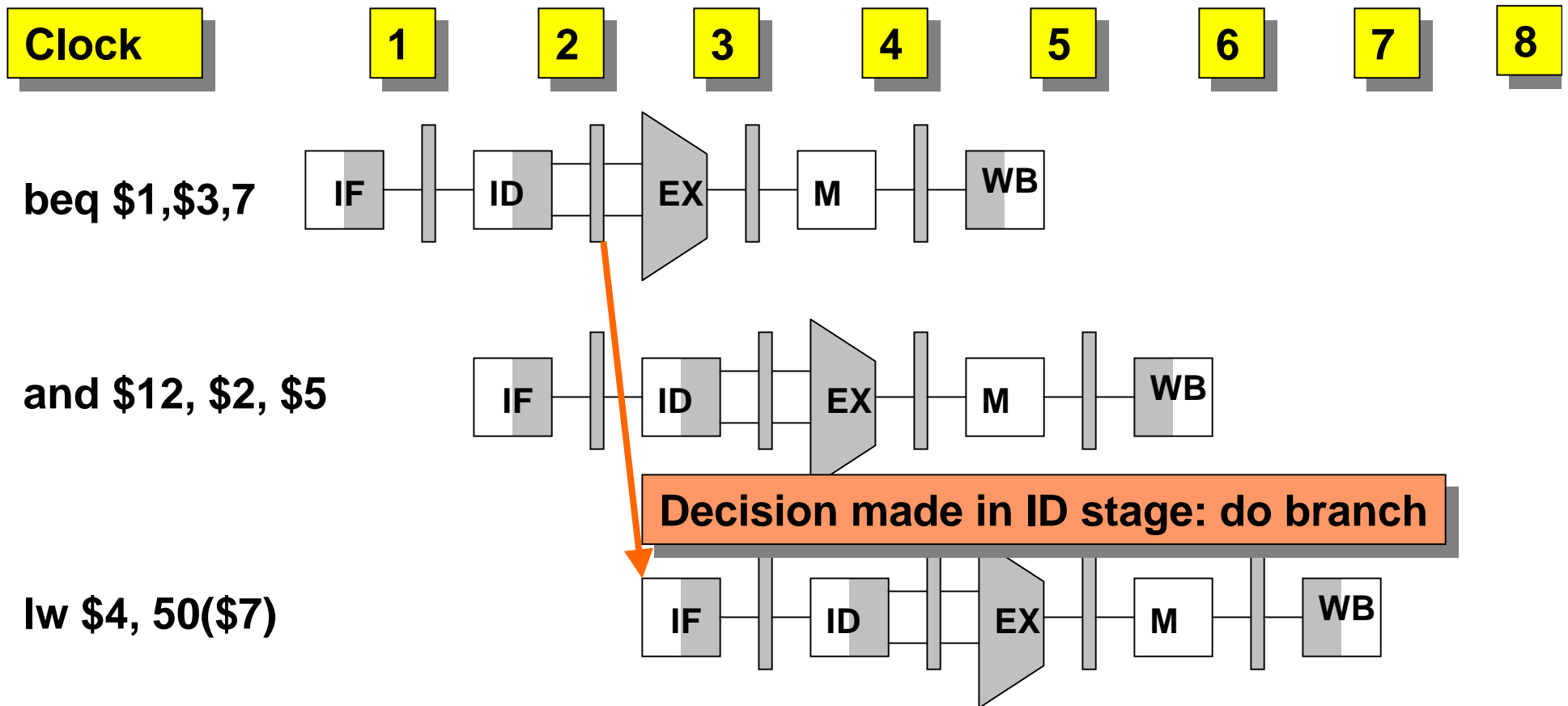
Branch Hazards: Soln #2, Predict until Decision made



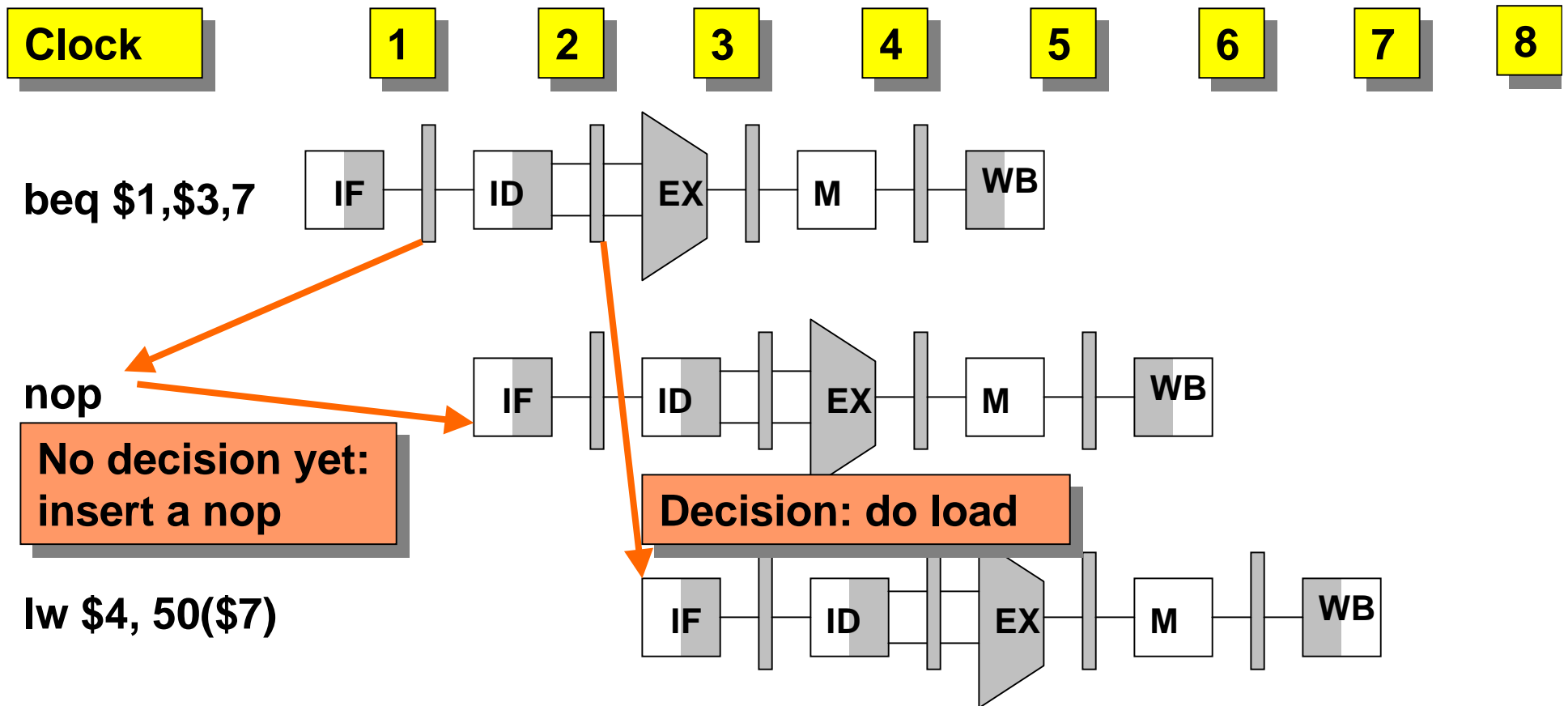
Branch Hazards: Soln #3, Delayed Decision



Branch Hazards: Soln #3, Delayed Decision



Branch Hazards: Decision made in the ID stage (figure 6.4)



Branch Hazards: Soln #2, Predict until Decision made

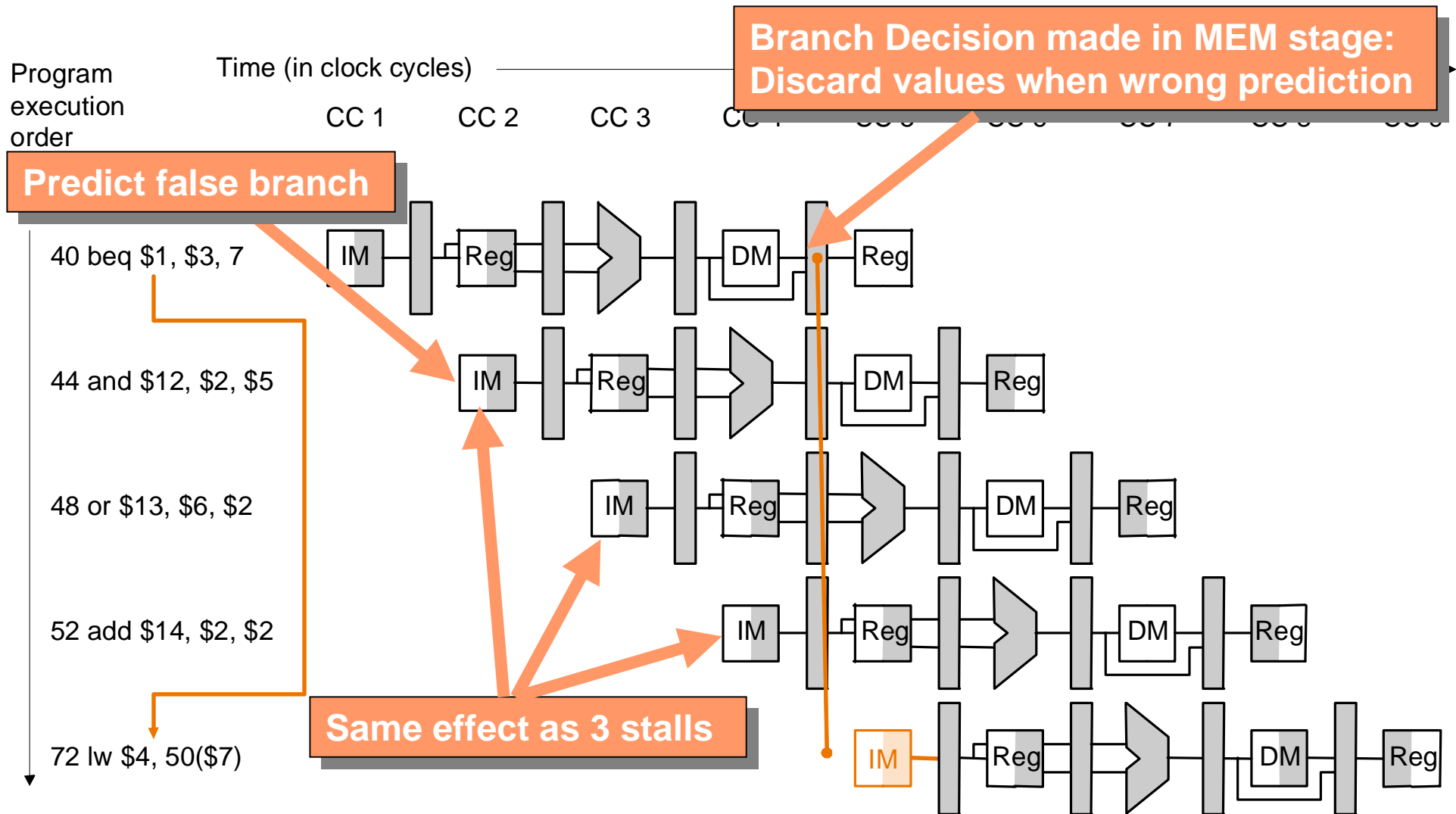
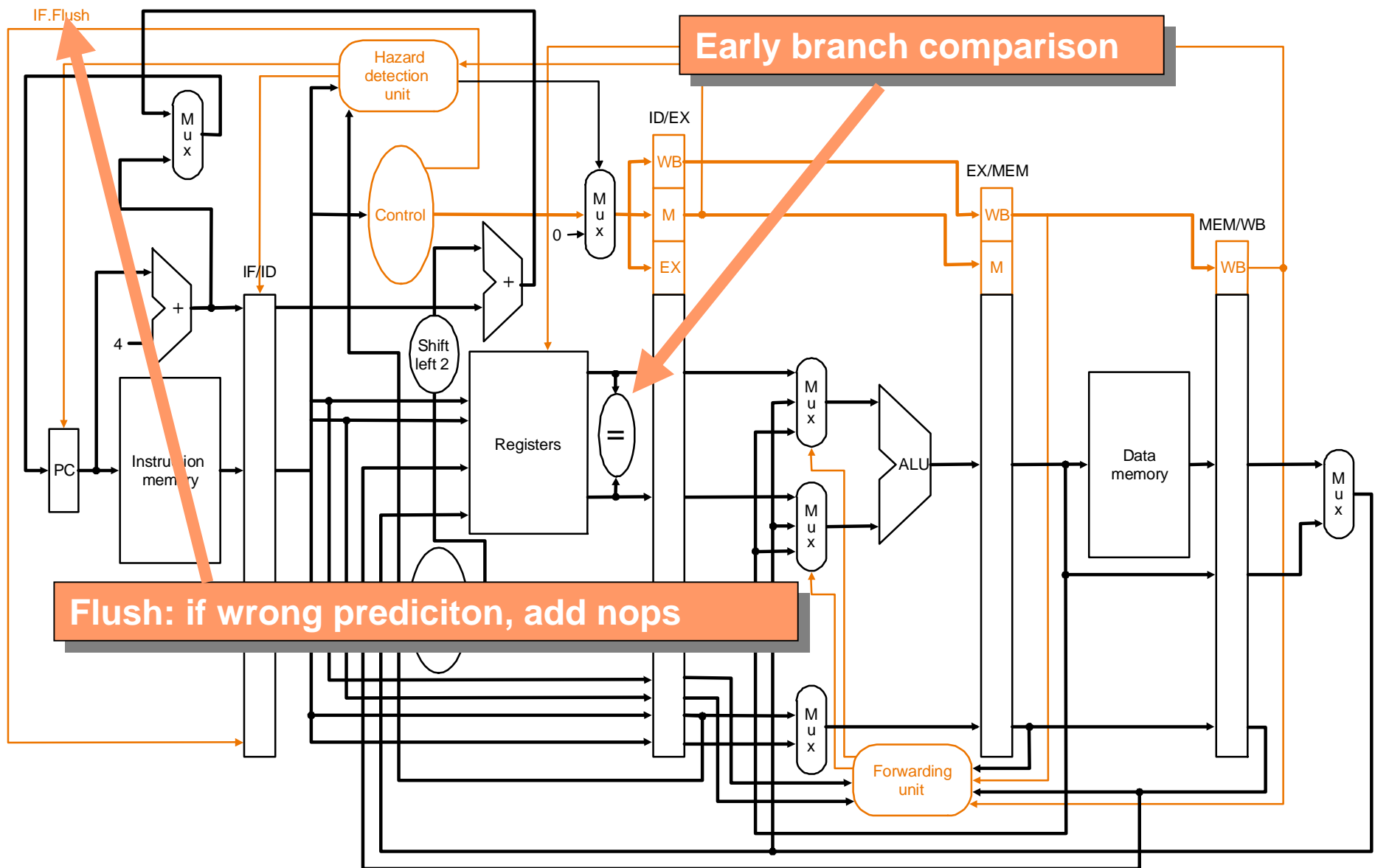


Figure 6.50

Figure 6.51



Performance

load: assume half of the instructions are immediately followed by an instruction that uses it (i.e. data dependency)

load instruction time = $50\% * (1 \text{ clock}) + 50\% * (2 \text{ clocks}) = 1.5$

Jump: assume that jumps always pay 1 full clock cycle delay (stall). Jump instruction time = 2

Branch: the branch delay of misprediction is 1 clock cycle that 25% of the branches are mispredicted.

branch time = $75\% * (1 \text{ clocks}) + 25\% * (2 \text{ clocks}) = 1.25$

Performance, page 504

Instruction	Single-Cycle	Multi-Cycle Clocks	Pipeline Cycles	Instruction Mix
loads	1	5	1.5	23%
stores	1	4	1	13%
arithmetic	1	4	1	43%
branches	1	3	1.25	19%
jumps	1	3	2	2%
Clock speed	125 Mhz 8 ns	500 Mhz 2 ns	500 Mhz 2 ns	
CPI	1	4.02	1.18	= Σ Cycles*Mix
MIPS	125 MIPS	125 MIPS	424 MIPS	= Clock/CPI

Homework for Wednesday April 5, 2000



- 1. Convert the RISCEE 1 Architecture into a pipeline Architecture (like Figure 6.30) (showing the number data and control bits).**
- 2. Build the control line table (like Figure 6.28) for the RISCEE3 pipeline architecture for RISCEE1 instructions: (addi, subi, load, store, beq, jmp, jal).**
- 3. Homework 6.23, p534**
- 4. Homework 6.24, p534**