

Lecture 17

Data Hazards II

School of Computer Science and Engineering
Soongsil University

4. The Processor

- 4.1 Introduction
- 4.2 Logic Design Conventions
- 4.3 Building a Datapath
- 4.4 A Simple Implementation Scheme
- 4.5 An Overview of Pipelining
- 4.6 Pipelined Datapath and Control
- 4.7 Data Hazards: Forwarding versus Stalling
- 4.8 Control Hazards
- 4.9 Exceptions
- 4.10 Parallelism via Instructions
- 4.11 Real Stuff: The ARM Cortex-A8 and Intel Core i7 Pipelines
- 4.12 Going Faster: Instruction-Level Parallelism and Matrix Multiply

New Hazard Conditions

- 1a. EX/MEM.RegWrite and (EX/MEM.RegisterRd \neq 0)
and (EX/MEM.RegisterRd = ID/EX.RegisterRs)
- 1b. EX/MEM.RegWrite and (EX/MEM.RegisterRd \neq 0)
and (EX/MEM.RegisterRd = ID/EX.RegisterRt)
- 2a. MEM/WB.RegWrite and (MEM/WB.RegisterRd \neq 0)
and (MEM/WB.RegisterRd = ID/EX.RegisterRs)
- 2b. MEM/WB.RegWrite and (MEM/WB.RegisterRd \neq 0)
and (MEM/WB.RegisterRd = ID/EX.RegisterRt)

Forwarding

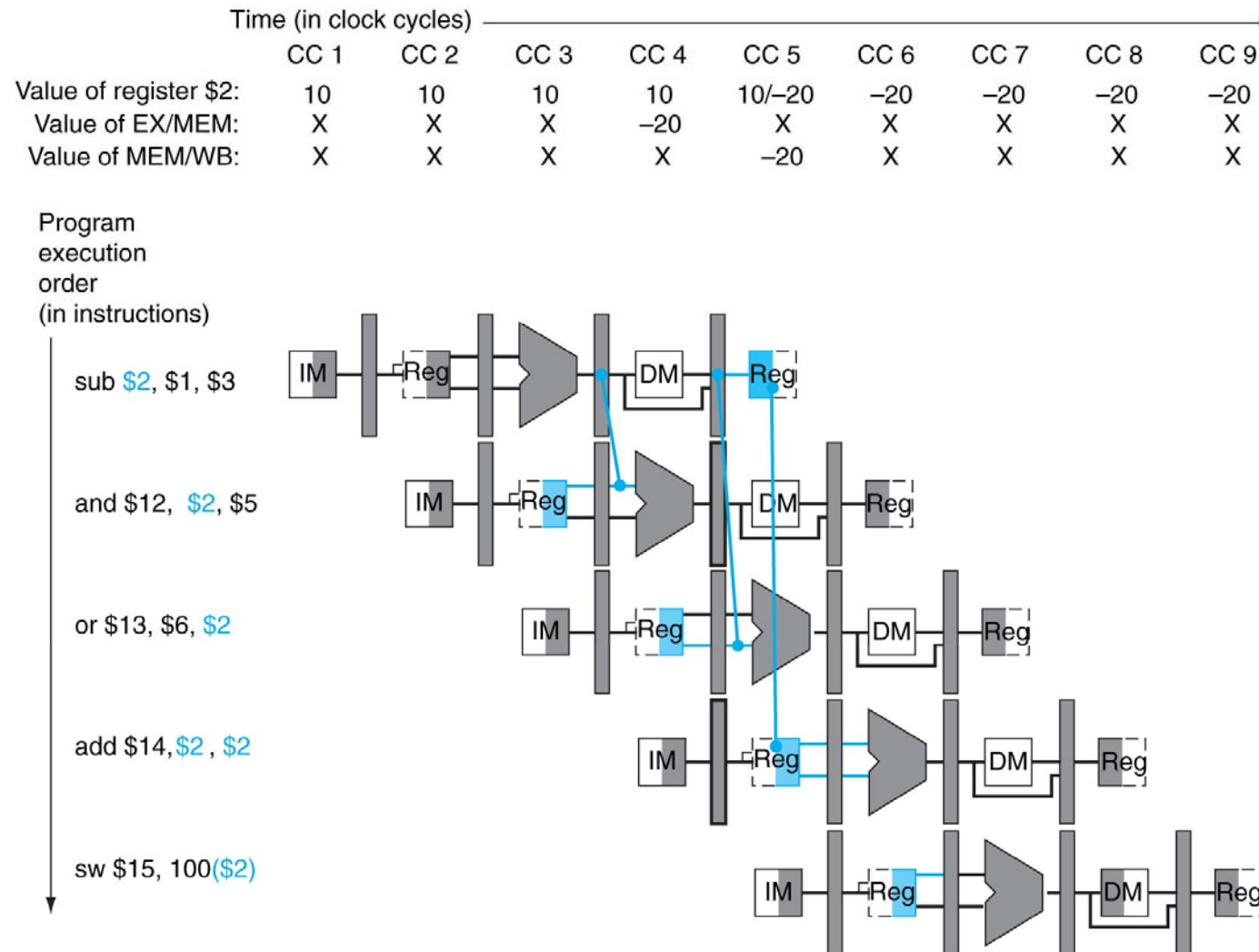
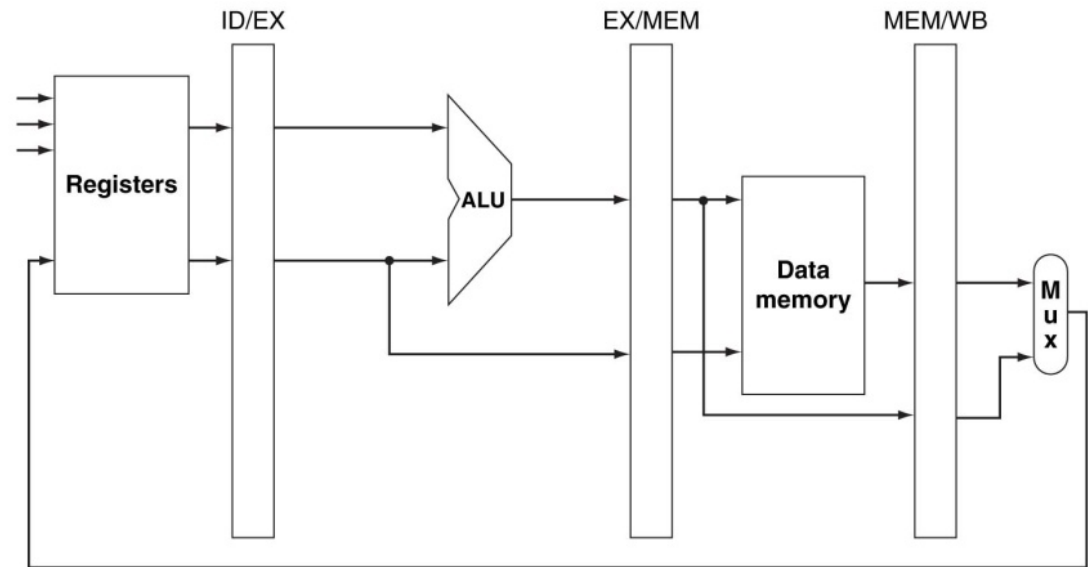
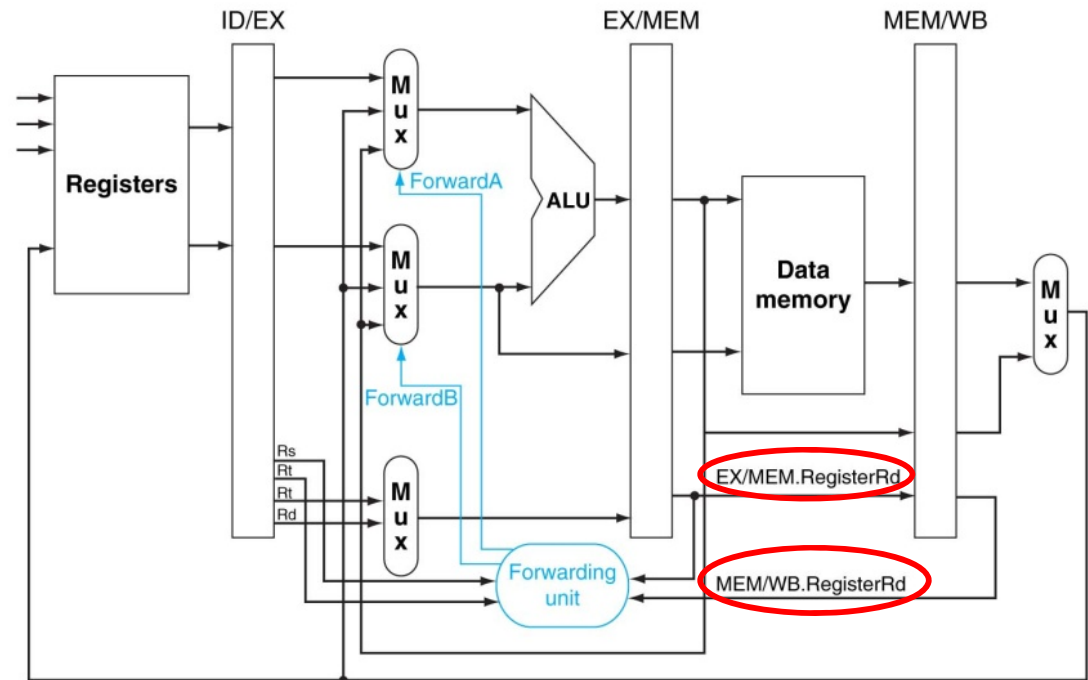


Figure 4.53

Forwarding Unit



a. No forwarding



b. With forwarding

Figure 4.54

Control Signals for the Forwarding Multiplexors

MUX control	Source	Description
ForwardA=00	ID/EX	1st ALU operand comes from the register file.
ForwardA=10	EX/MEM	1st ALU operand is forwarded from the prior ALU result. (1a hazard)
ForwardA=01	MEM/WB	1st ALU operand is forwarded from data memory or an earlier ALU result. (2a hazard)
ForwardB=00	ID/EX	2nd ALU operand comes from the register file.
ForwardB=10	EX/MEM	2nd ALU operand is forwarded from the prior ALU result. (1b hazard)
ForwardB=01	MEM/WB	2nd ALU operand is forwarded from data memory or an earlier ALU result. (2b hazard)

Figure 4.55

Two Conditions for Forwarding (1)

1. EX hazard (= 1a & 1b hazards)

If (EX/MEM.RegWrite
and (EX/MEM.RegisterRd \neq 0)
and (EX/MEM.RegisterRd = ID/EX.RegisterRs)) **ForwardA = 10**

If (EX/MEM.RegWrite
and (EX/MEM.RegisterRd \neq 0)
and (EX/MEM.RegisterRd = ID/EX.RegisterRt)) **ForwardB = 10**

♣ **Forwards the results from the previous instruction to either input of the ALU.**

Two Conditions for Forwarding (2)

2. MEM hazard (= 2a & 2b hazards)

If (MEM/WB.RegWrite
and(MEM/WB.RegisterRd != 0)
and (MEM/WB.RegisterRd = ID/EX.RegisterRs)) **ForwardA = 01**

If (MEM/WB.RegWrite
and(MEM/WB.RegisterRd !=0)
and (MEM/WB.RegisterRd = ID/EX.RegisterRt)) **ForwardB = 01**

♣ **Forwards the results from the instruction in the WB stage to either input of the ALU.**

Double Data Hazard

```
add  $1, $1, $2  
add  $1, $1, $3  
add  $1, $1, $4
```

- Simultaneous type 1 and type 2 hazards for the same ALU input
 - type 1 hazard
 - ❖ Want to use the most recent result
 - ❖ The result is forwarded from the MEM stage.

Revised Control for Type 2 Hazard

- Forward only when Type 1 hazard condition isn't true
 - ❖ If (MEM/WB.RegWrite
and (MEM/WB.RegisterRd != 0)
and not (EX/MEM.RegWrite and EX/MEM.RegisterRd != 0)
and (EX/MEM.RegisterRd != ID/EX.RegisterRs)
and (MEM/WB.RegisterRd = ID/EX.RegisterRs))

ForwardA = 01

- ❖ If (MEM/WB.RegWrite
and (MEM/WB.RegisterRd != 0)
and not (EX/MEM.RegWrite and EX/MEM.RegisterRd != 0)
and (EX/MEM.RegisterRd != ID/EX.RegisterRt)
and (MEM/WB.RegisterRd = ID/EX.RegisterRt))

ForwardB = 01

Final Control for ForwardA

```
If (EX/MEM.RegWrite
    and (EX/MEM.RegisterRd != 0)
    and (EX/MEM.RegisterRd = ID/EX.RegisterRs)) ForwardA = 10
else if (MEM/WB.RegWrite
    and (MEM/WB.RegisterRd != 0)
    /* and not (EX/MEM.RegWrite and EX/MEM.RegisterRd != 0)
       and (EX/MEM.RegisterRd != ID/EX.RegisterRs) */
    and (MEM/WB.RegisterRd = ID/EX.RegisterRs)) ForwardA = 01
else ForwardA = 00
```

Datapath with Forwarding Unit

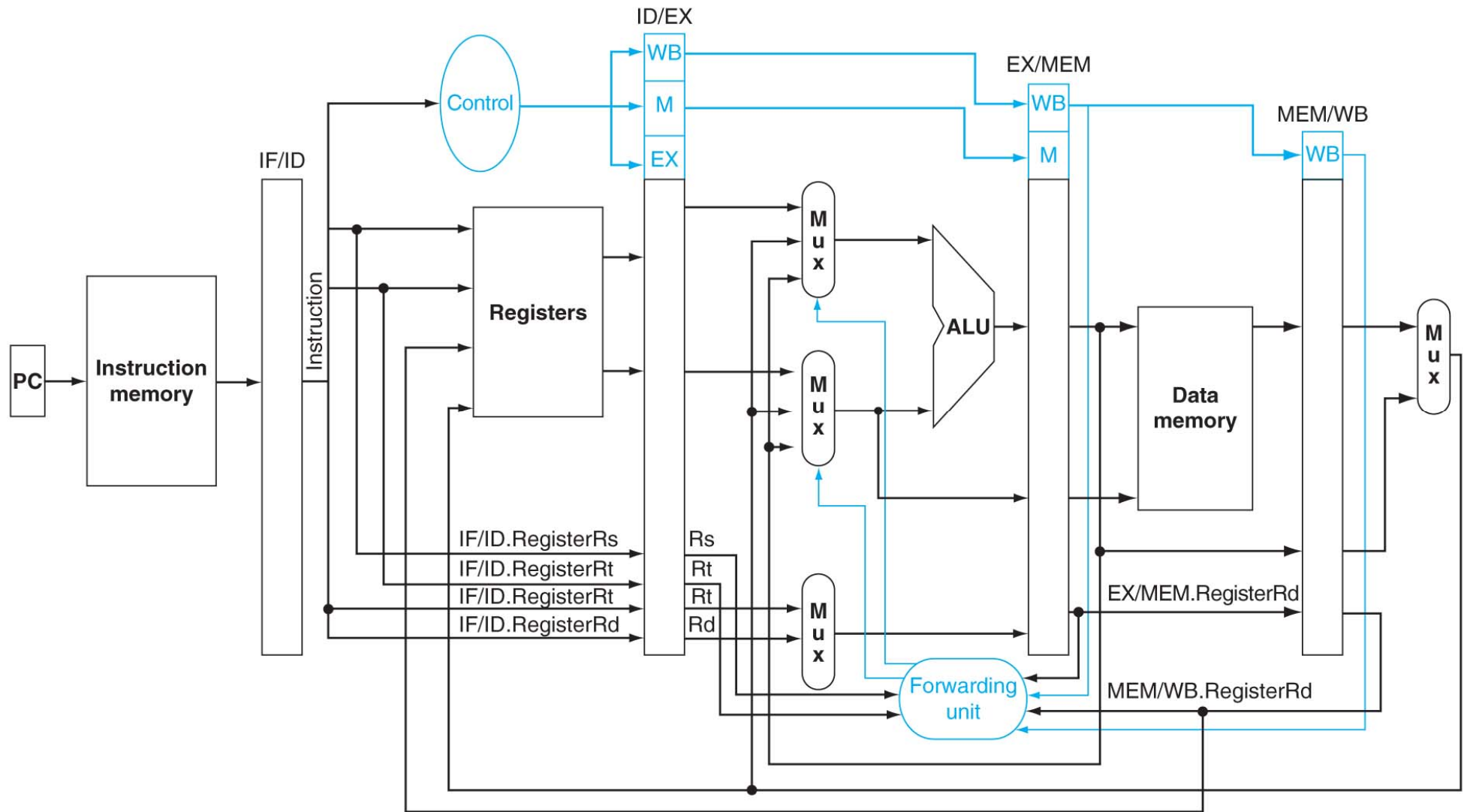


Figure 4.56

A Close-up of the Datapath in Figure 4.54

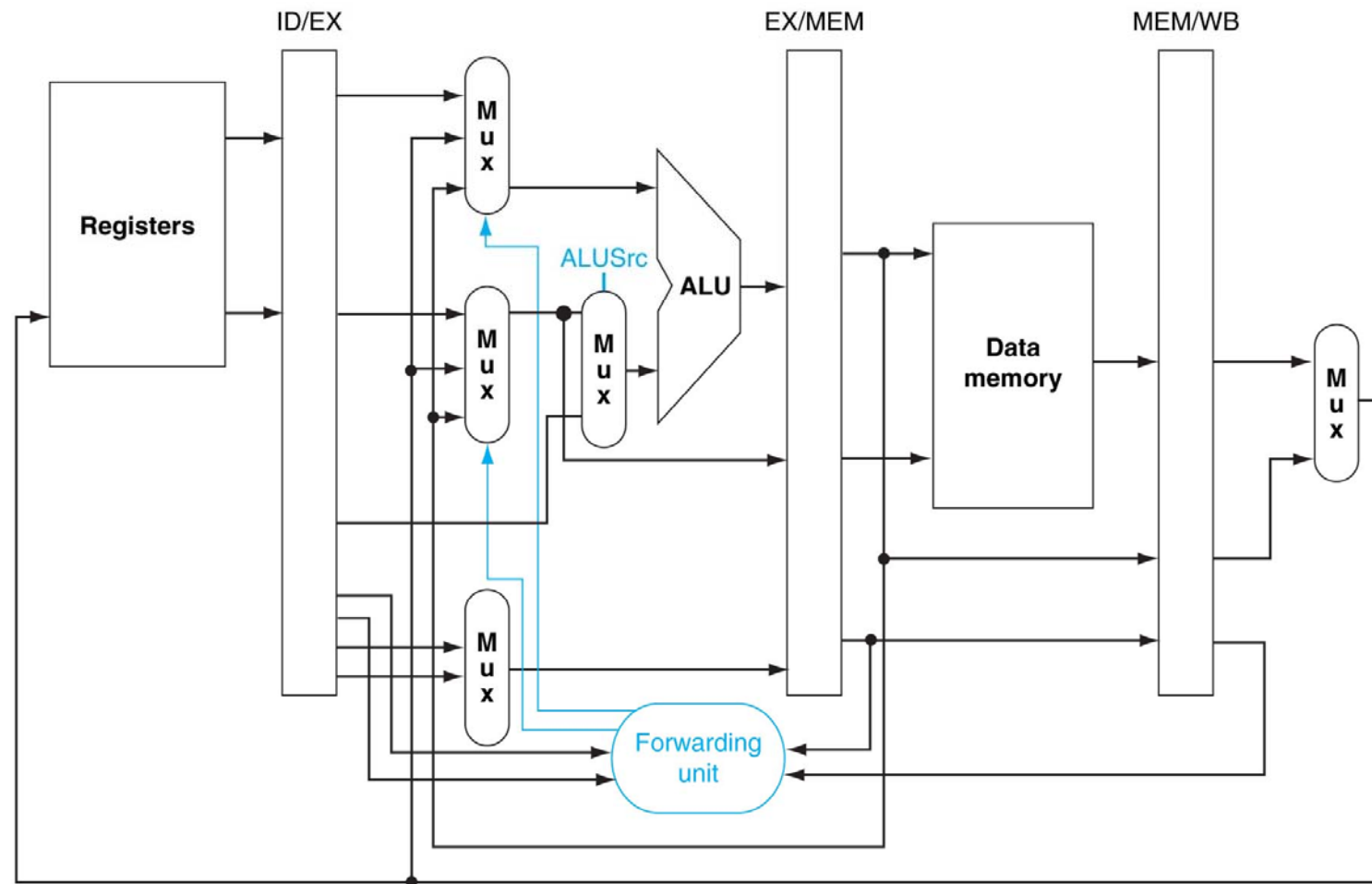


Figure 4.57

Data Hazards and Stalls

▪ Load-use data hazard

- ❖ A load instruction followed by an instruction that reads its result.

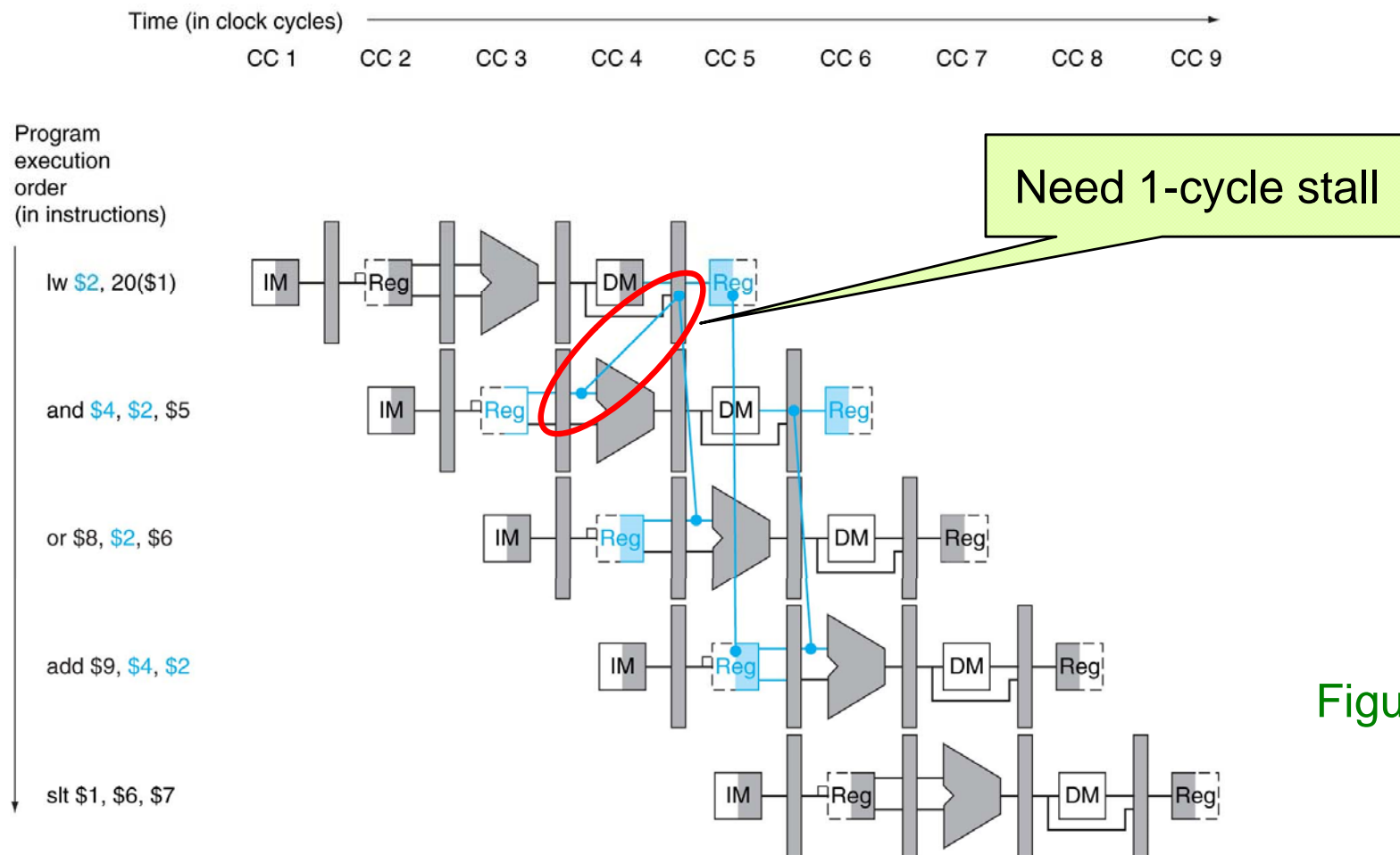


Figure 4.58

Hazard Detection Unit

- ID stage
- Insert the stall between the load and its use
- if (ID/EX.MemRead and
((ID/EX.RegisterRt = IF/ID.RegisterRs) or
(ID/EX.RegisterRt = IF/ID.RegisterRt)))
then stall the pipeline

	IF	ID	EX	MEM	WB
clock 1	or \$8,\$2,\$6	and \$4,\$2,\$5	lw \$2,20(\$1)		
clock 2	or \$8,\$2,\$6	and \$4,\$2,\$5	bubble	lw \$2,20(\$1)	
clock 3	add \$9,\$4,\$2	or \$8,\$2,\$6	and \$4,\$2,\$5	bubble	lw \$2,20(\$1)

Hazard Detection Unit and Forwarding Unit

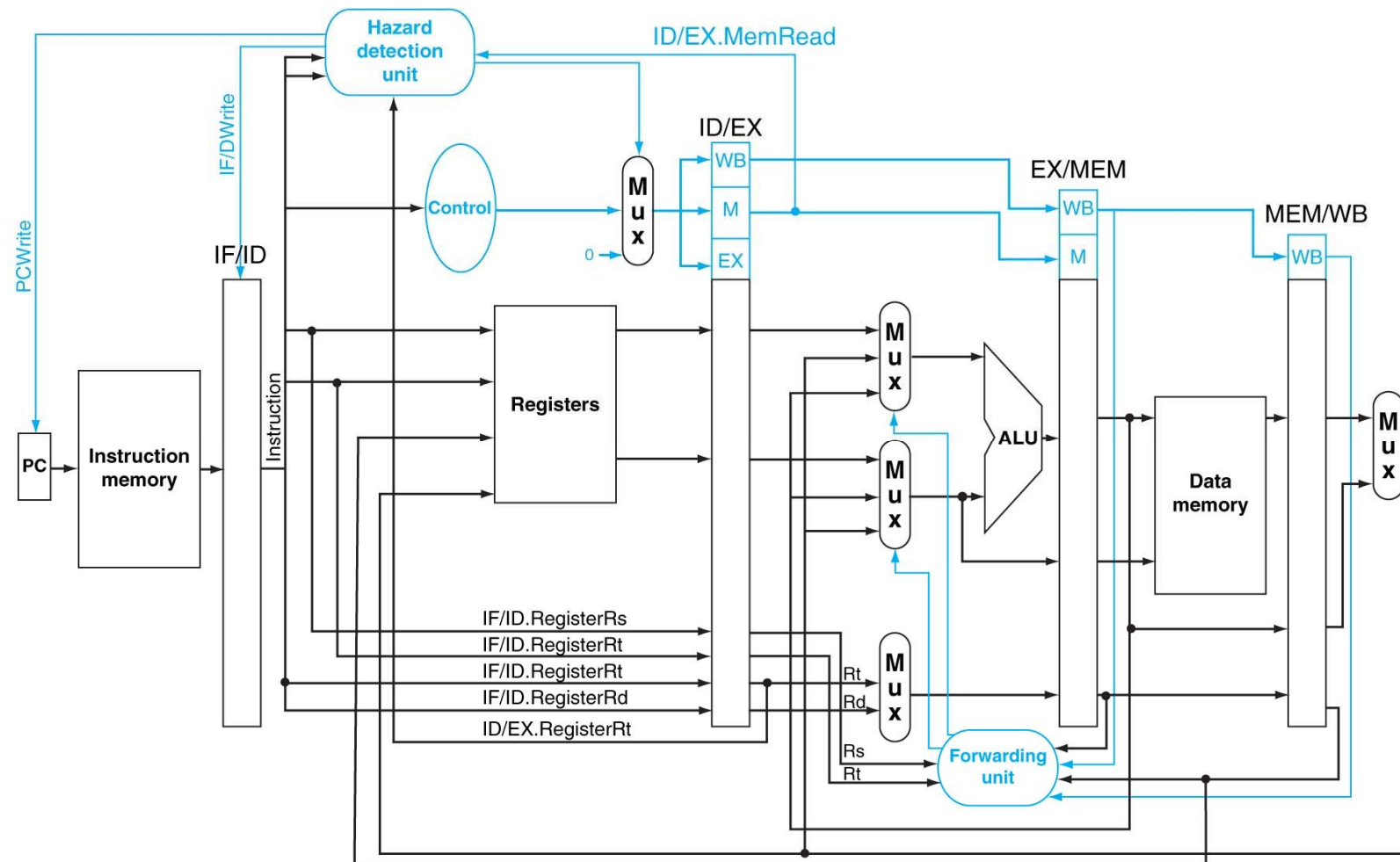


Figure 4.60

Inserting Bubbles

(1) Stalling instructions in ID and IF stages

Changing neither PC nor IF/ID register

-> fetch and decode the same instructions during stalls

(2) Simulating nop instruction

Deasserting all 9 control signals in the EX, MEM and WB stages

- ◆ Only RegWrite and MemWrite need be 0.
- ◆ Others can be don't cares.

Delayed Load

- Requiring compiler to follow the load with an instruction independent of that load
 - ❖ in the worst case, inserting nop instruction
 - Delaying the use of the data loaded from memory
- [ref] Mano

The BIG Picture
















- ❖ The compiler must understand the pipeline to achieve the best performance.
- Otherwise, unexpected stalls will reduce the performance.

Supplement

Example: Prob. 1 of 2011-1 Terminal Exam

- Without forwarding unit

```
add $1,$2,$3
sub $2,$3,$4
and $5,$1,$6
or  $7,$2,$5
sw  $1,100($7)
```

	IF	ID	EX	MEM	WB
CC0	add				
CC1	sub	add			
CC2	and	sub	add		
CC3	or	and	sub	add	
CC4	or	and		sub	add
CC5	sw	or	and		sub
CC6	sw	or		and	
CC7	sw	or			and
CC8		sw	or		
CC9		sw		or	
CC10		sw			or
CC11			sw		
CC12				sw	
CC13					sw

Example: Prob. 1 of 2011-1 Terminal Exam

- With forwarding unit

```
add $1,$2,$3
sub $2,$3,$4
and $5,$1,$6
or  $7,$2,$5
sw  $1,100($7)
```


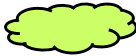




	IF	ID	EX	MEM	WB
CC0	add				
CC1	sub	add			
CC2	and	sub	add		
CC3	or	and	sub	add	
CC4	sw	or	and	sub	add
CC5		sw	or	and	sub
CC6			sw	or	and
CC7				sw	or
CC8					sw

Example: Load-Use Data Hazard

- Without forwarding unit

```

add $3,$1,$2
lw  $1,100($7)
and $5,$1,$6
or  $7,$2,$4
sw  $1,100($8)
    
```




	IF	ID	EX	MEM	WB
CC0	add				
CC1	lw	add			
CC2	and	lw	add		
CC3	or	and	lw	add	
CC4	or	and		lw	add
CC5	or	and			lw
CC6	sw	or	and		
CC7		sw	or	and	
CC8			sw	or	and
CC9				sw	or
CC10					sw

Example: Load-Use Data Hazard

- With forwarding unit

```

add $3,$1,$2
lw  $1,100($7)
and $5,$1,$6
or  $7,$2,$4
sw  $1,100($8)
    
```

	IF	ID	EX	MEM	WB
CC0	add				
CC1	lw	add			
CC2	and	lw	add		
CC3	or	and	lw	add	
CC4	or	and		lw	add
CC5	sw	or	and		lw
CC6		sw	or	and	
CC7			sw	or	and
CC8				sw	or
CC9					sw
CC10					