Multicycle operations

E. Sanchez

Politecnico di Torino
Dipartimento di Automatica e Informatica

FLOATING-POINT OPERATIONS

Floating point units perform more complex operations than integer ones.

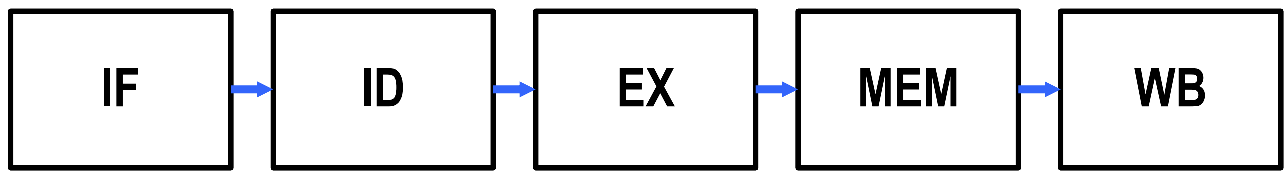
Therefore, in order to force them to perform their job in a single clock cycle, the designer should

- either use a very slow clock, or
- make these units very complex.

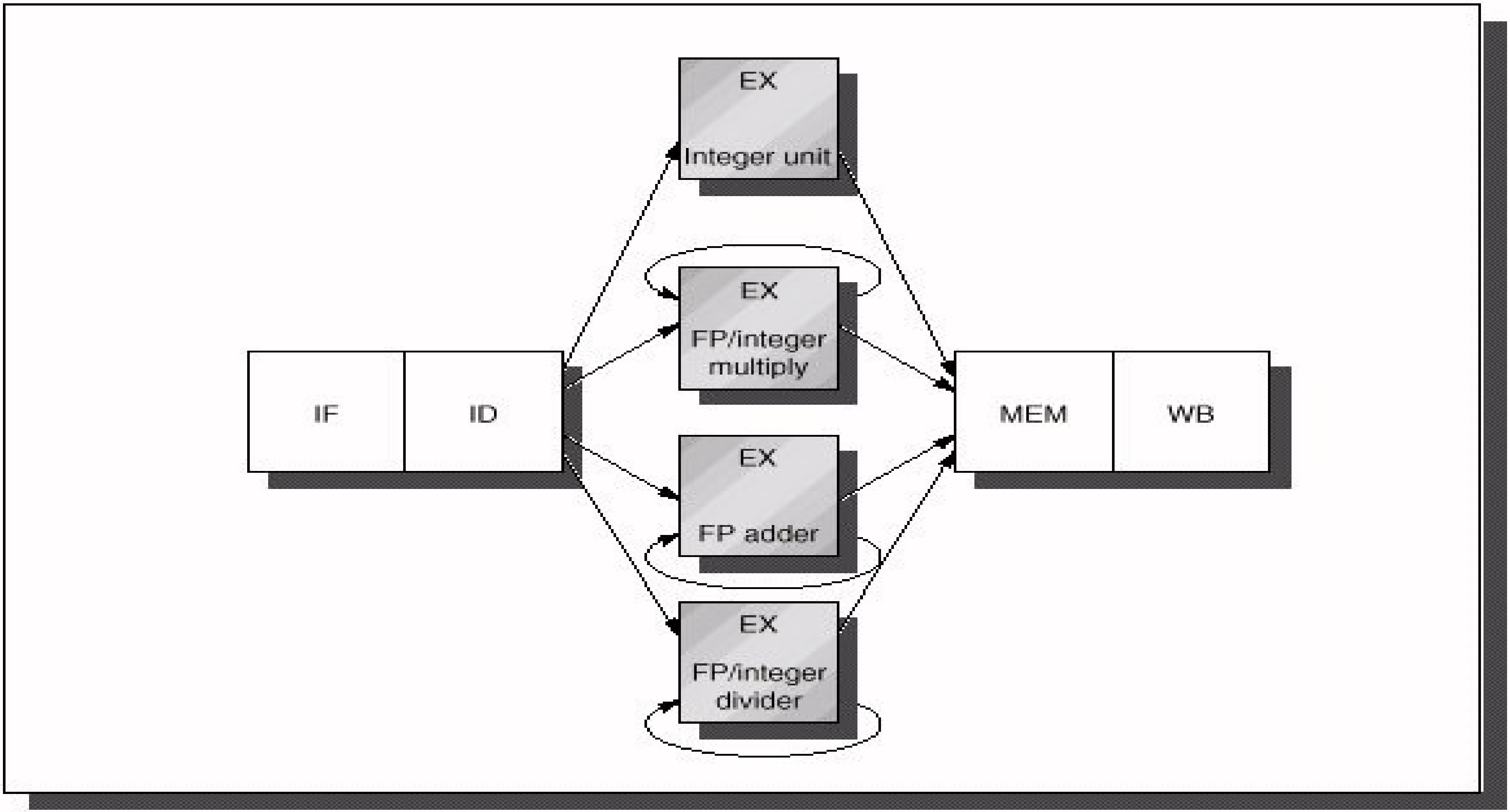
As a popular alternative, floating point units generally require more than one clock cycle to complete.

The EX stage is composed of different functional units, and is repeated as many times, as the instruction requires.

Integer Pipeline



Extension for FP



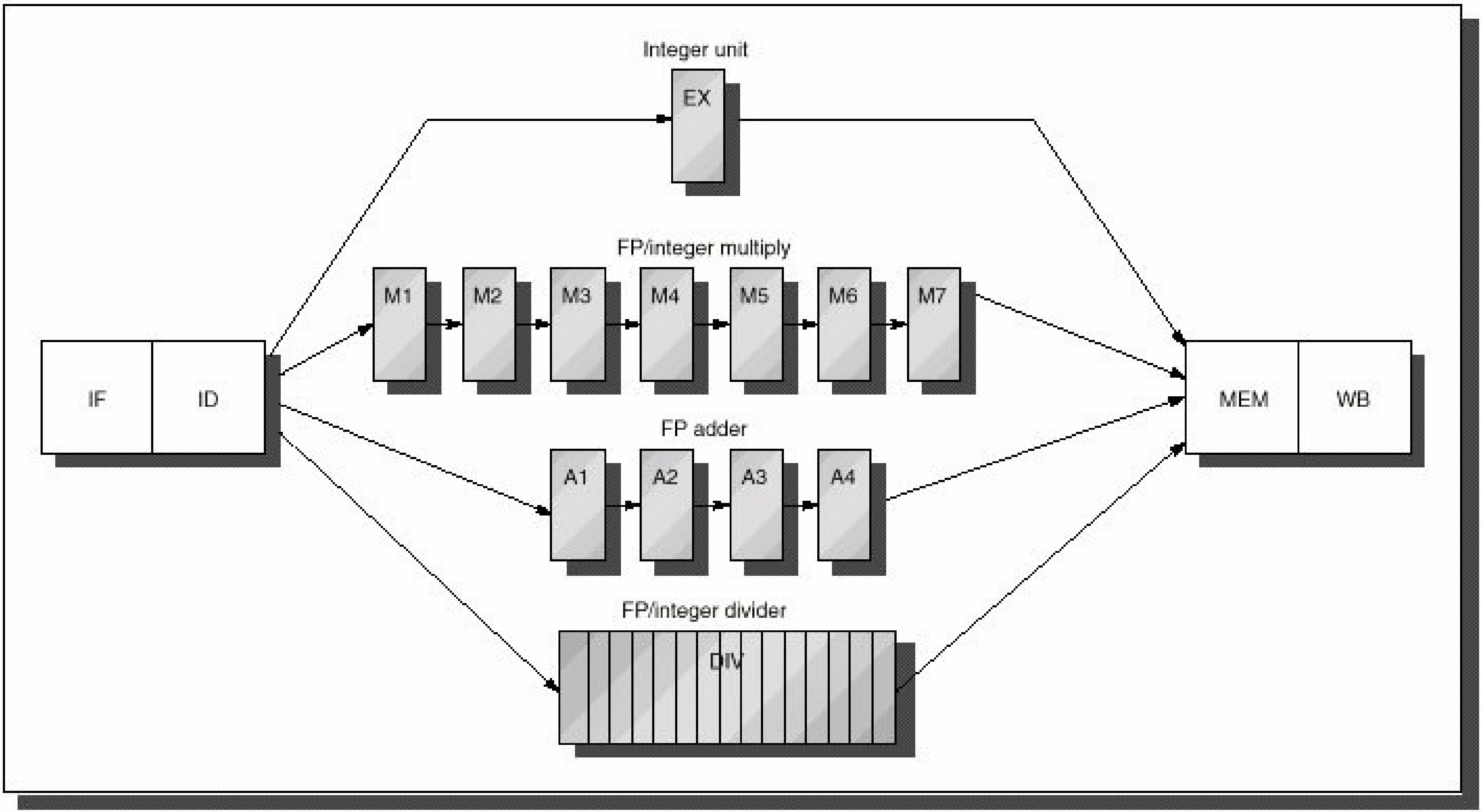
Latency and Initiation Interval

- ***Latency***
  - **It is the number of cycles that should last between an instruction that produces a result and an instruction that uses the same result.**
- ***Initiation interval***
  - **It is the number of cycles that must elapse between issuing two operations of the same type to the same unit.**

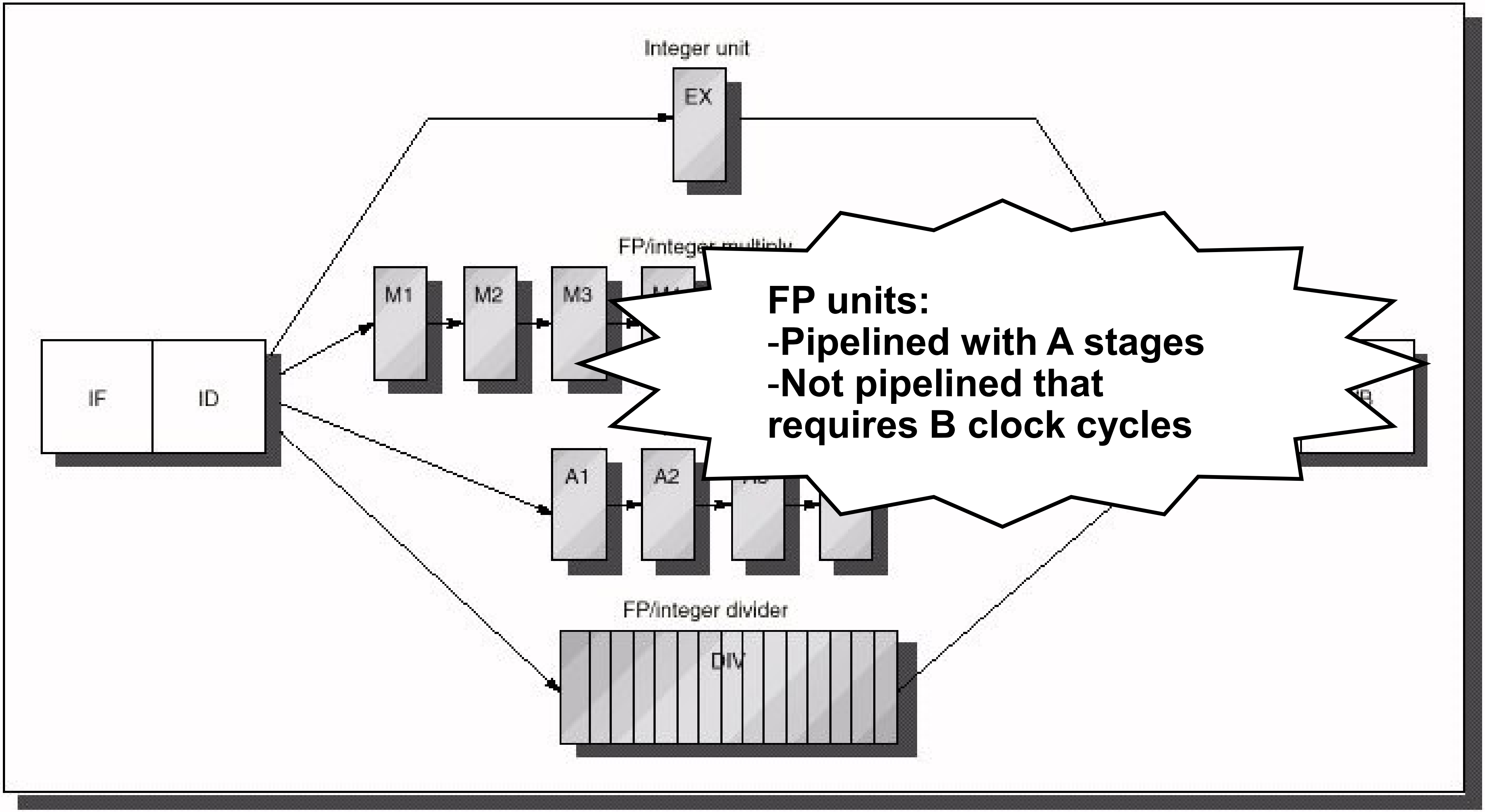
Example

Functional Unit	*Latency*	*Initiation Interval*
Integer ALU	**0**	**1**
Data Memory	**1**	**1**
FP add	**3**	**1**
FP/integer multiply	**6**	**1**
FP/integer divide	**24**	**24**

Pipelined FP units



Pipelined FP units



Hazards

Due to the different structure of the EX stage, hazards may become more frequent.

Example

`MULTD`	IF	ID	*M1*	M2	M3	M4	M5	M6	M7	MEM	WB
`ADDD`		IF	ID	*A1*	A2	A3	A4	MEM	WB		
`LD`			IF	ID	*EX*	MEM	WB				
`SD`				IF	ID	*EX*	MEM	WB			

Stages requiring a data available

Stages where a data is made available

Structural hazards

Structural hazards can occur:

- **because of the unpipelined divide unit, several instructions could need it at the same time**
- **because the instructions have different running times, the number of register writes required in a cycle can be larger than 1.**

Contemporary register writes

Instruction	Clock cycle number 1	2	3	4	5	6	7	8	9	10	11
MUL.D F0,F4,F6	IF	ID	M1	M2	M3	M4	M5	M6	M7	MEM	WB
...		IF	ID	EX	MEM	WB					
...			IF	ID	EX	MEM	WB				
ADD.D F2,F4,F6				IF	ID	A1	A2	A3	A4	MEM	WB
...					IF	ID	EX	MEM	WB		
...						IF	ID	EX	MEM	WB	
L.D F2,0(R2)							IF	ID	EX	MEM	WB

Solutions

- Adding other write ports (normally too expensive)
- Forcing a structural hazard:
  - instructions are stalled in the ID stage, or
  - instructions are stalled before entering the MEM or WB stage.

More frequent data hazards

Because of longer latency of operations, stalls for data hazards may stall the pipeline for longer periods.

Example

Instruction	Clock cycle number																
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
L.D F4,0(R2)	IF	ID	EX	MEM	WB												
MUL.D F0,F4,F6		IF	ID	stall	M1	M2	M3	M4	M5	M6	M7	MEM	WB				
ADD.D F2,F0,F8			IF	stall	ID	stall	stall	stall	stall	stall	stall	A1	A2	A3	A4	MEM	WB
S.D F2,0(R2)					IF	stall	stall	stall	stall	stall	stall	ID	EX	stall	stall	stall	MEM

More frequent data hazards

Because of longer latency of operati... ...ata hazards may stall the pipeline for longe...

Read After Write (RAW) hazard

<u>Example</u>

Instruction	Clock cycle number 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
L.D F4,0(R2)	IF	ID	EX	MEM	WB												
MUL.D F0,F4,F6		IF	ID	stall	M1	M2	M3	M4	M5	M6	M7	MEM	WB				
ADD.D F2,F0,F8			IF	stall	ID	stall	stall	stall	stall	stall	stall	A1	A2	A3	A4	MEM	WB
S.D F2,0(R2)					IF	stall	stall	stall	stall	stall	stall	ID	EX	stall	stall	stall	MEM

New data hazards

Instructions no longer reach WB in order: therefore, new kinds of data hazards are now possible.

Example

	Clock cycle number										
Instruction	1	2	3	4	5	6	7	8	9	10	11
MUL.D F0,F4,F6	IF	ID	M1	M2	M3	M4	M5	M6	M7	MEM	WB
...		IF	ID	EX	MEM	WB					
...			IF	ID	EX	MEM	WB				
ADD.D F2,F4,F6				IF	ID	A1	A2	A3	A4	MEM	WB
...					IF	ID	EX	MEM	WB		
...						IF	ID	EX	MEM	WB	
L.D F2,0(R2)							IF	ID	EX	MEM	WB

New data h

Instructions no longer reach WB
kinds of data hazards are now po

L.D could write in F2 before ADD.D

Example

	Clock cycle number										
Instruction	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**		**11**
`MUL.D F0,F4,F6`	IF	ID	M1	M2	M3	M4	M5	M6	M7	M	WB
...		IF	ID	EX	MEM	WB					
...			IF	ID	EX	MEM	WB				
`ADD.D F2,F4,F6`				IF	ID	A1	A2	A3	A4	MEM	WB
...					IF	ID	EX	MEM	WB		
...						IF	ID	EX	MEM	WB	
`L.D F2,0(R2)`							IF	ID	EX	MEM	WB

New data hazards

Instructions no longer reach WB in or... new kinds of data hazards are now possible

Write After Write (WAW) hazard

Example

	Clock cycle number										
Instruction	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
`MUL.D F0,F4,F6`	IF	ID	M1	M2	M3	M4	M5	M6	M7	MEM	WB
`...`		IF	ID	EX	MEM	WB					
`...`			IF	ID	EX	MEM	WB				
`ADD.D F2,F4,F6`				IF	ID	A1	A2	A3	A4	MEM	WB
`...`					IF	ID	EX	MEM	WB		
`...`						IF	ID	EX	MEM	WB	
`L.D F2,0(R2)`							IF	ID	EX	MEM	WB

Solution

Before issuing an instruction to the EX stage, check whether it is going to write on the same register of an instruction still in the EX stage.

In this case, stall the new instruction during the ID stage.

Summary

If hazard detection is always performed in the ID stage, three checks have to be performed:

- ***structural hazards*** **(involving the divide unit and the write port)**
- ***RAW data hazards*****: check whether some source register is listed among the destination registers of pending instructions, and whether this register will not be available at the right moment**
- ***WAW data hazards*****: check whether the instruction currently in ID has the same destination register of any instruction in A1,…,A4, D, M1, …, M7.**

THE MIPS R4000 PIPELINE

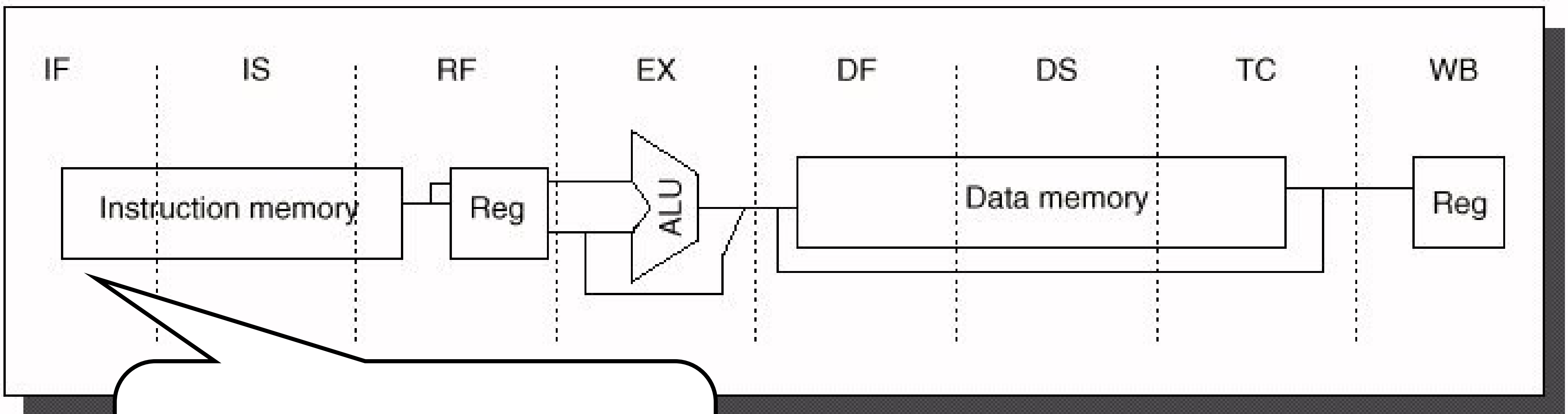
The MIPS R-4000 processor is a 64-bit microprocessor introduced in 1991, whose instruction set is similar to the MIPS64 one.

The R-4000 uses a deeper pipeline (8 stages) to account for slower cache access and higher clock frequency:

- memory accesses are decomposed in several stages.

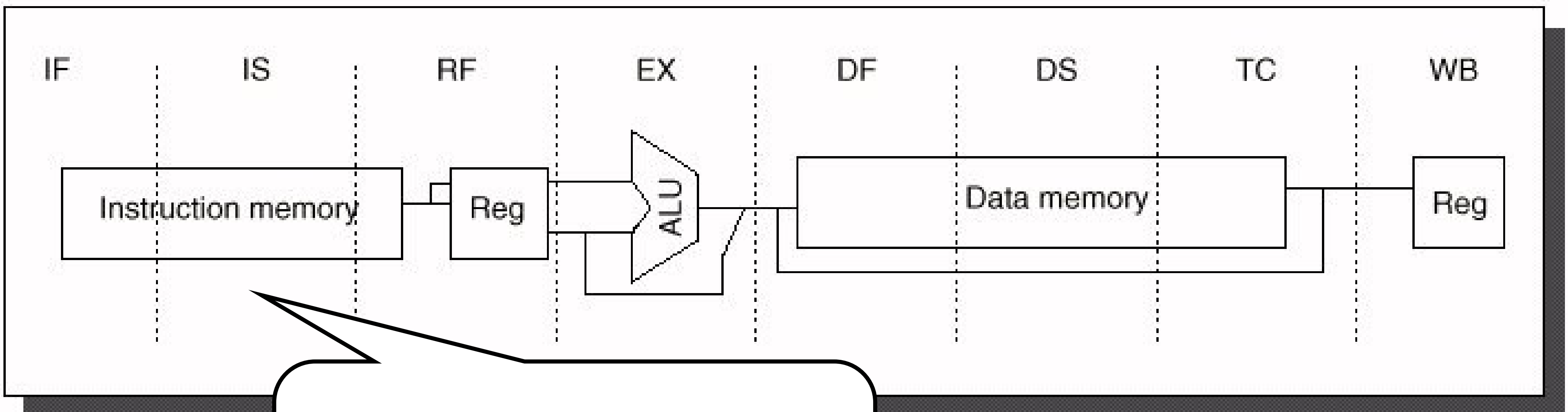
Long pipelines sometimes take the name of *superpipelines*.

The 8-stage pipeline



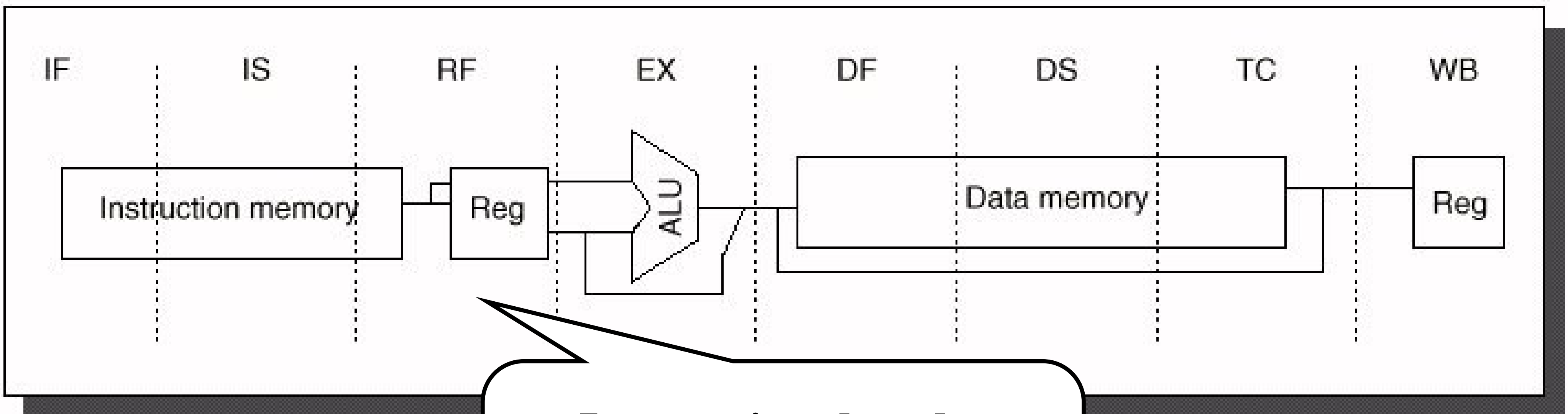
First part of instruction fetch

The 8-stage pipeline



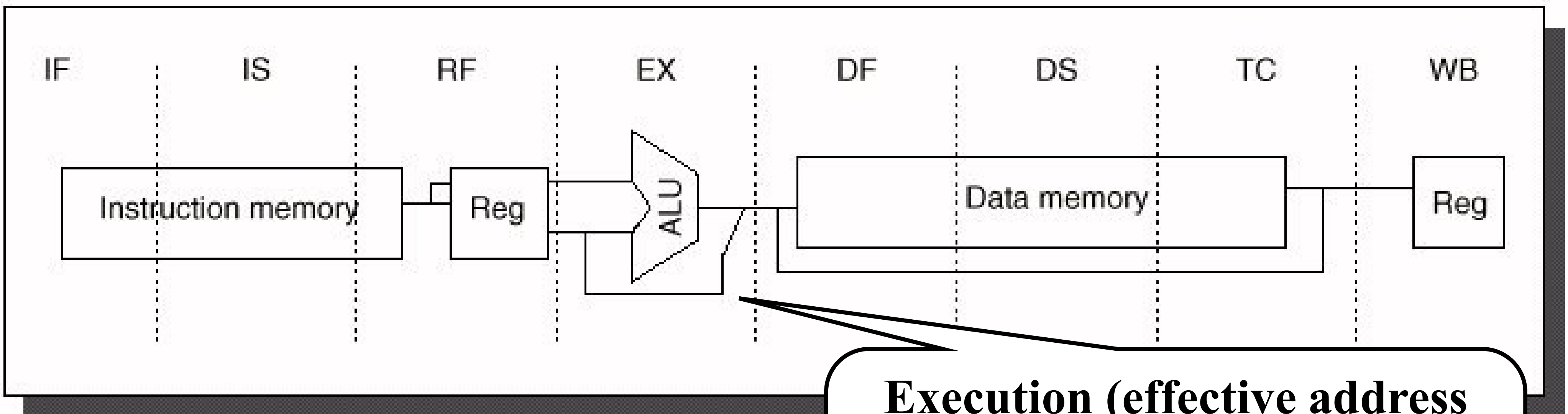
Second part of instruction fetch

The 8-stage pipeline



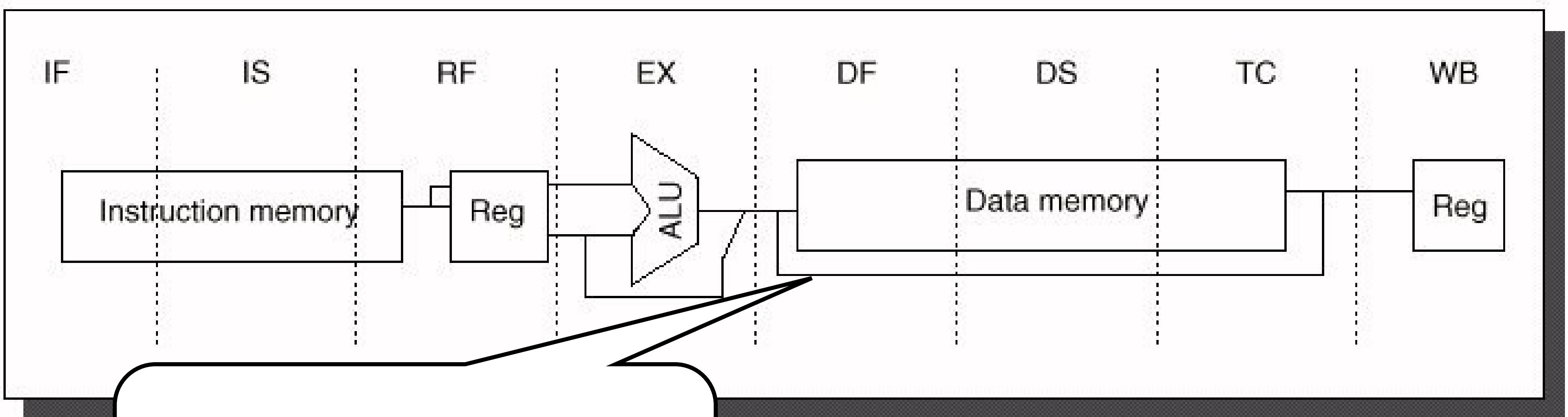
Instruction decode; register fetch; hazard checking

The 8-stage pipeline



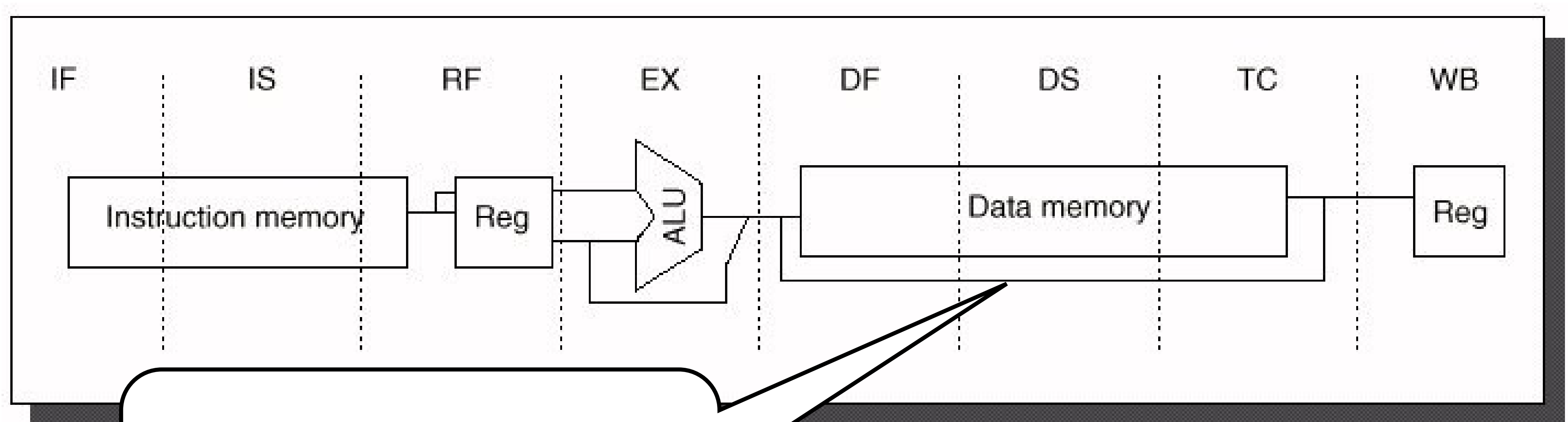
Execution (effective address calculation, ALU operation, branch target computation, condition evaluation)

The 8-stage pipeline



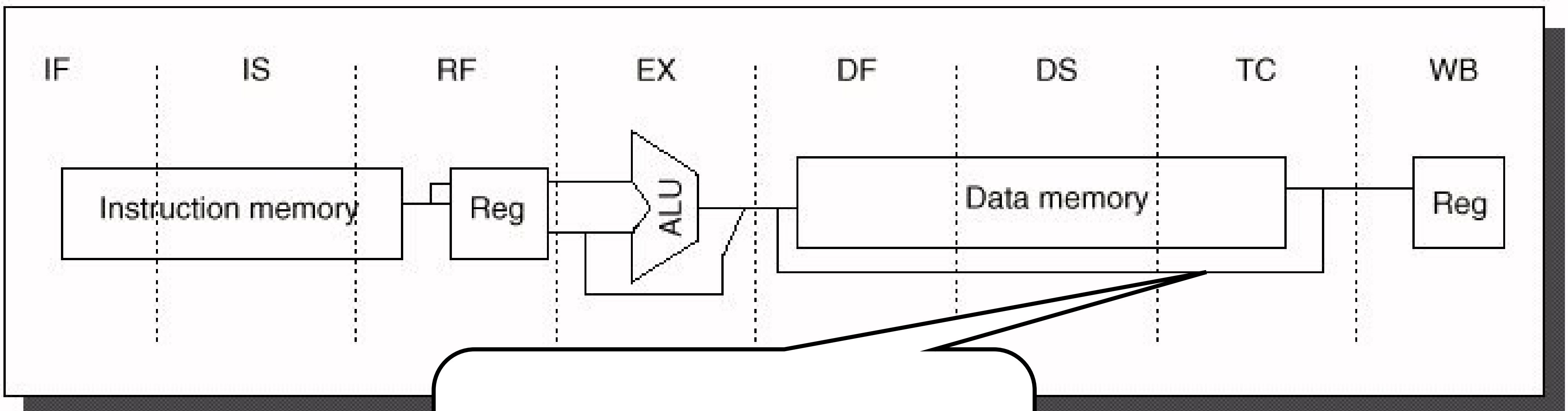
First part of data fetch

The 8-stage pipeline



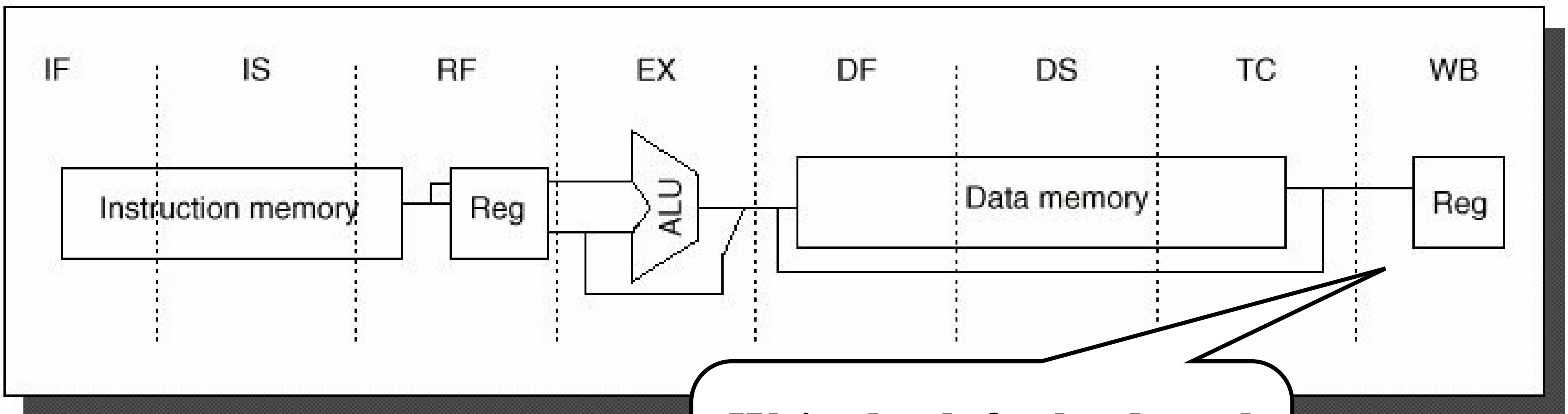
Second part of data fetch

The 8-stage pipeline



Last part of data fetch

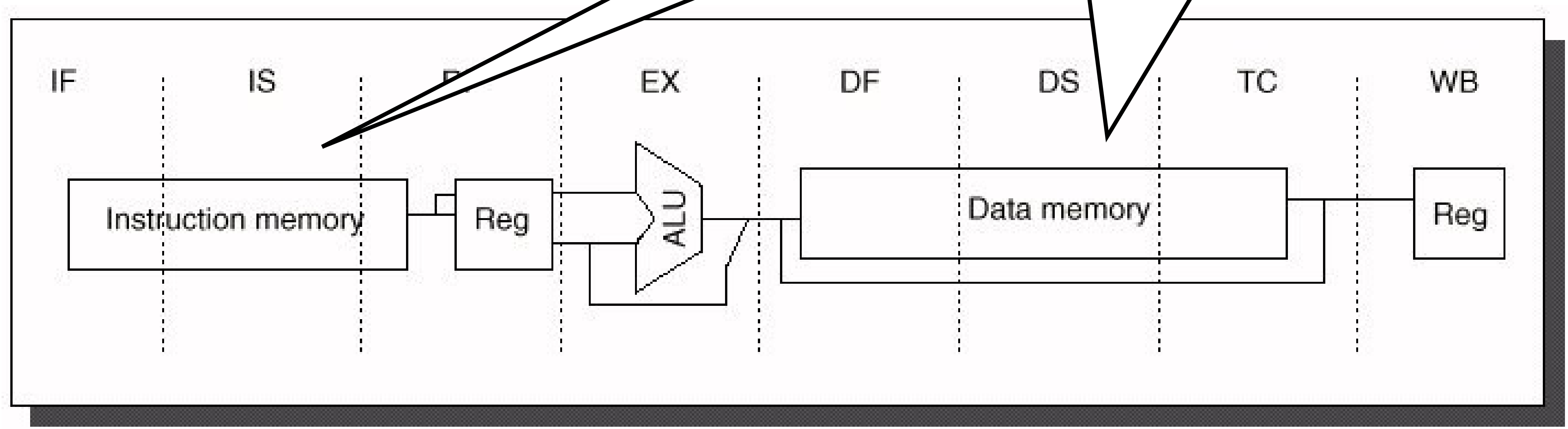
The 8-stage pipeline



Write-back for loads and register-register operations

The 8-sta

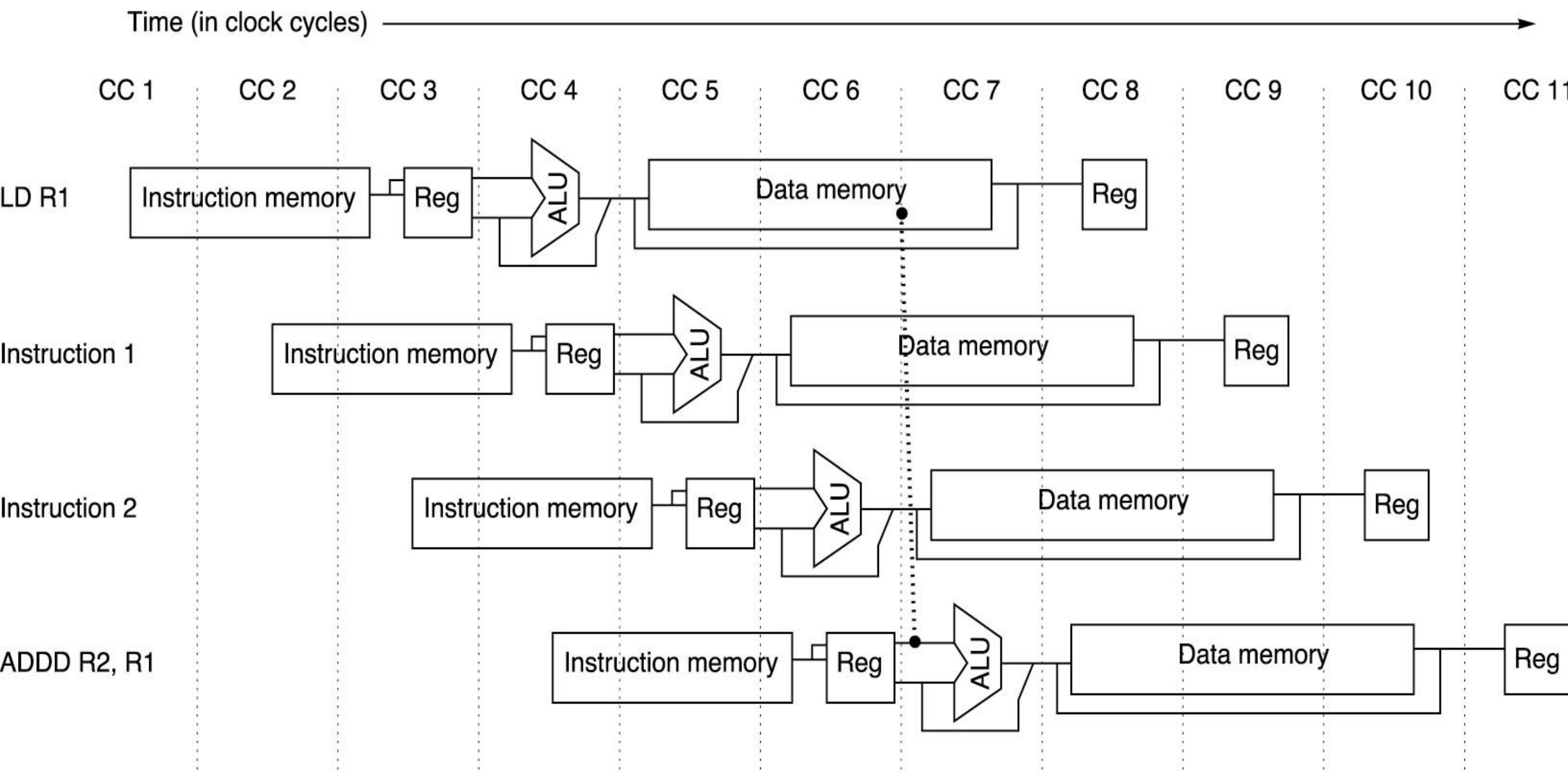
Both instruction and data memory accesses are pipelined: a new instruction can start on every clock cycle.



Characteristics

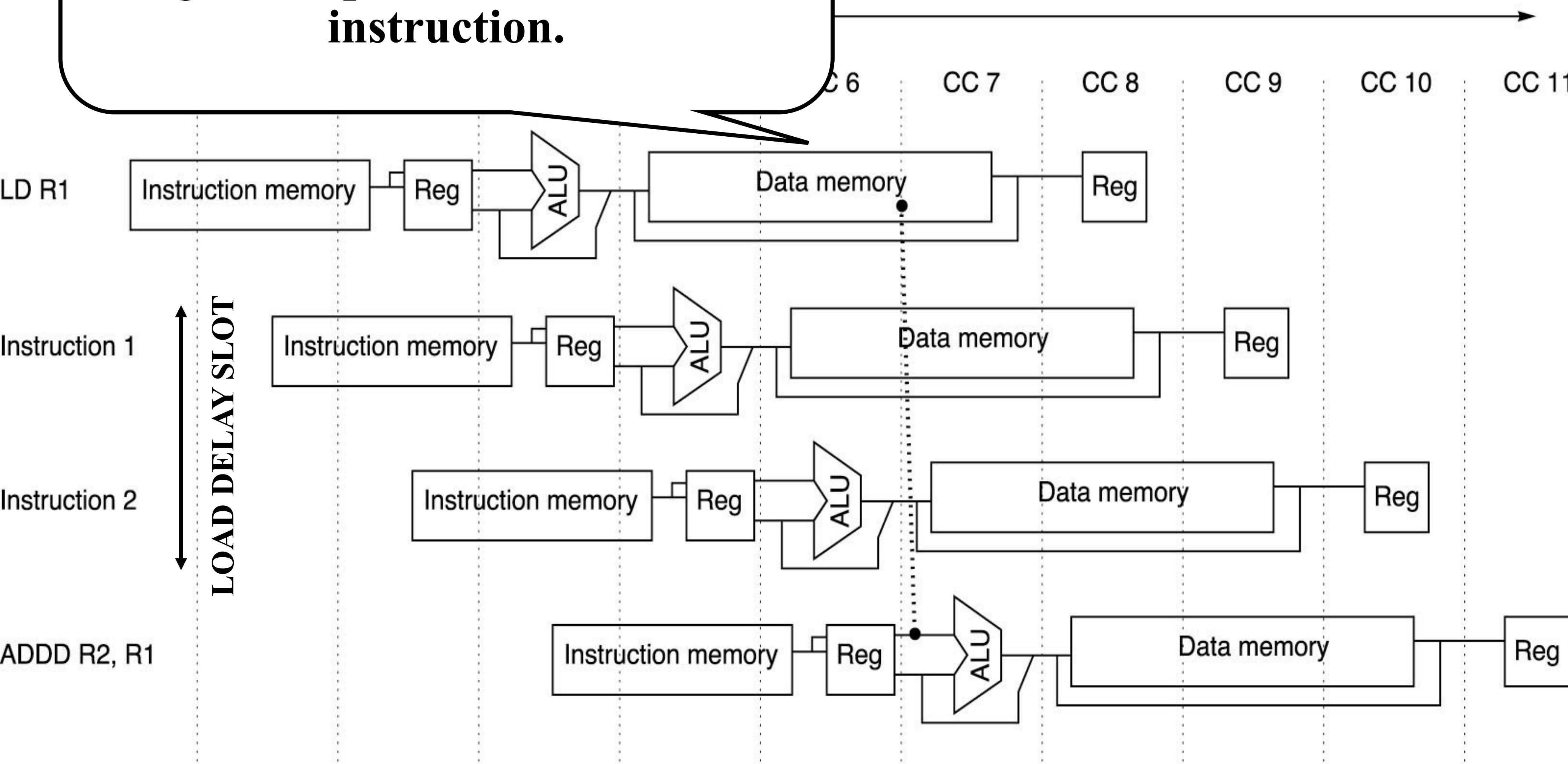
- **More forwarding is required**
- **Increased load delay slot (2 cycles)**
- **Increased branch delay slot (3 cycles).**

Load delay slot

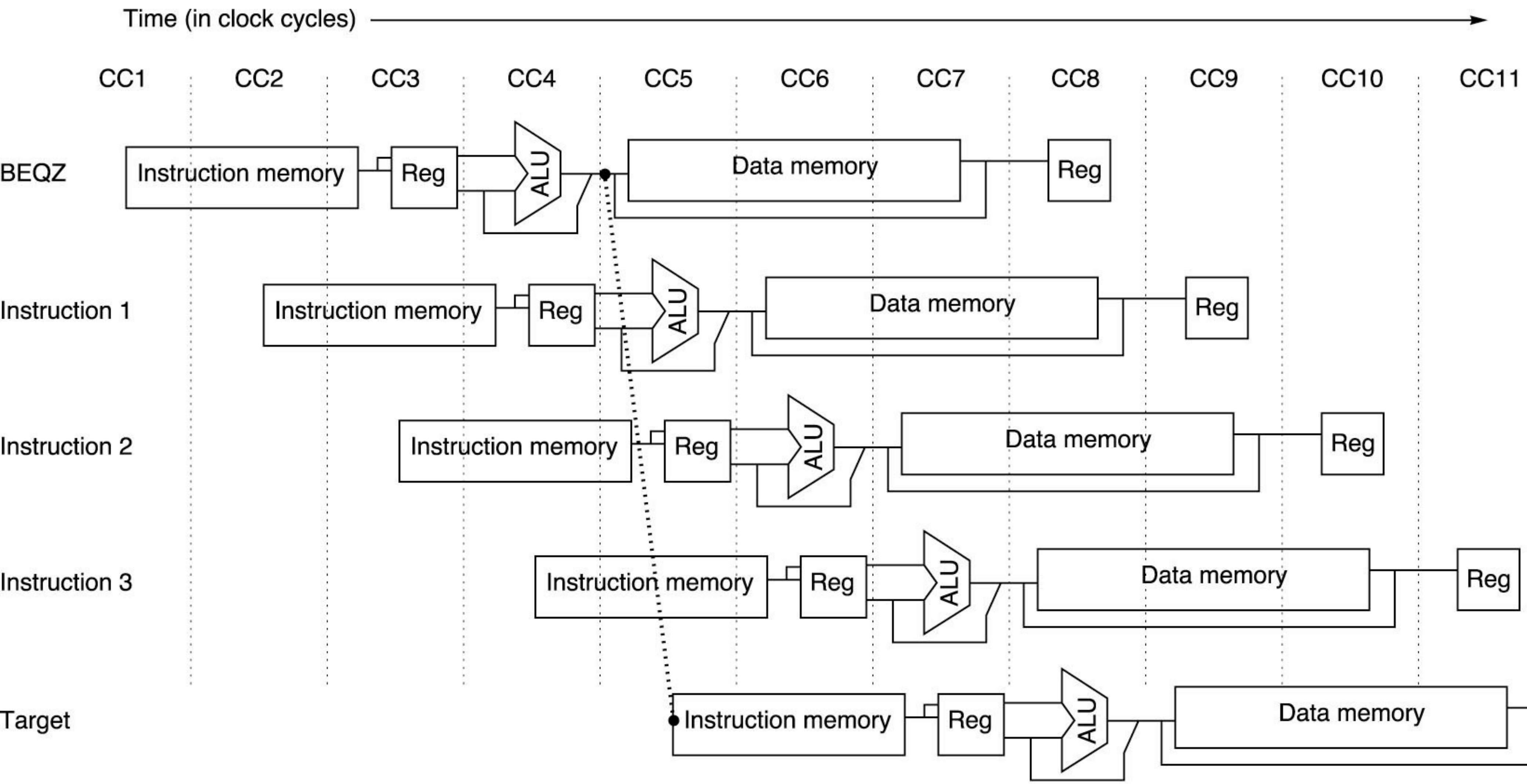


ay slot

The data is available at the end of DS. Activating the forwarding logic, it is passed to the ADDD instruction.

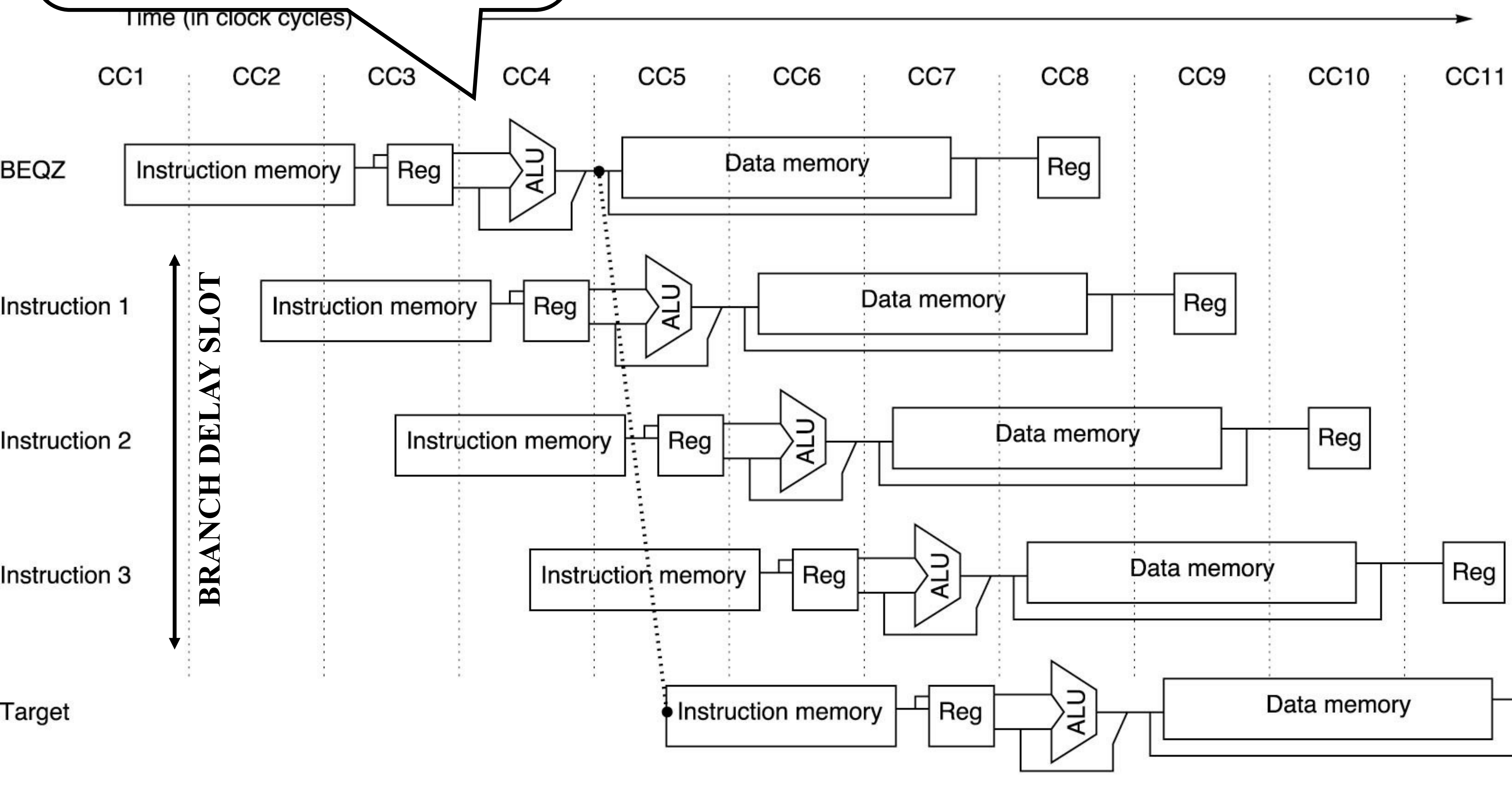


Branch delay slot



Condition evaluation is performed during EX.

nch delay slot



FP pipeline

- The FP unit is composed of three functional units: divider, multiplier, adder
- the FP unit can be thought as composed of 8 different stages:

stage	*functional unit*	*description*
A	adder	Mantissa ADD stage
D	divider	divide
E	multiplier	exception test
M	multiplier	multiplier I
N	multiplier	multiplier II
R	adder	rounding
S	adder	operand shift
U		unpack numbers

FP operations

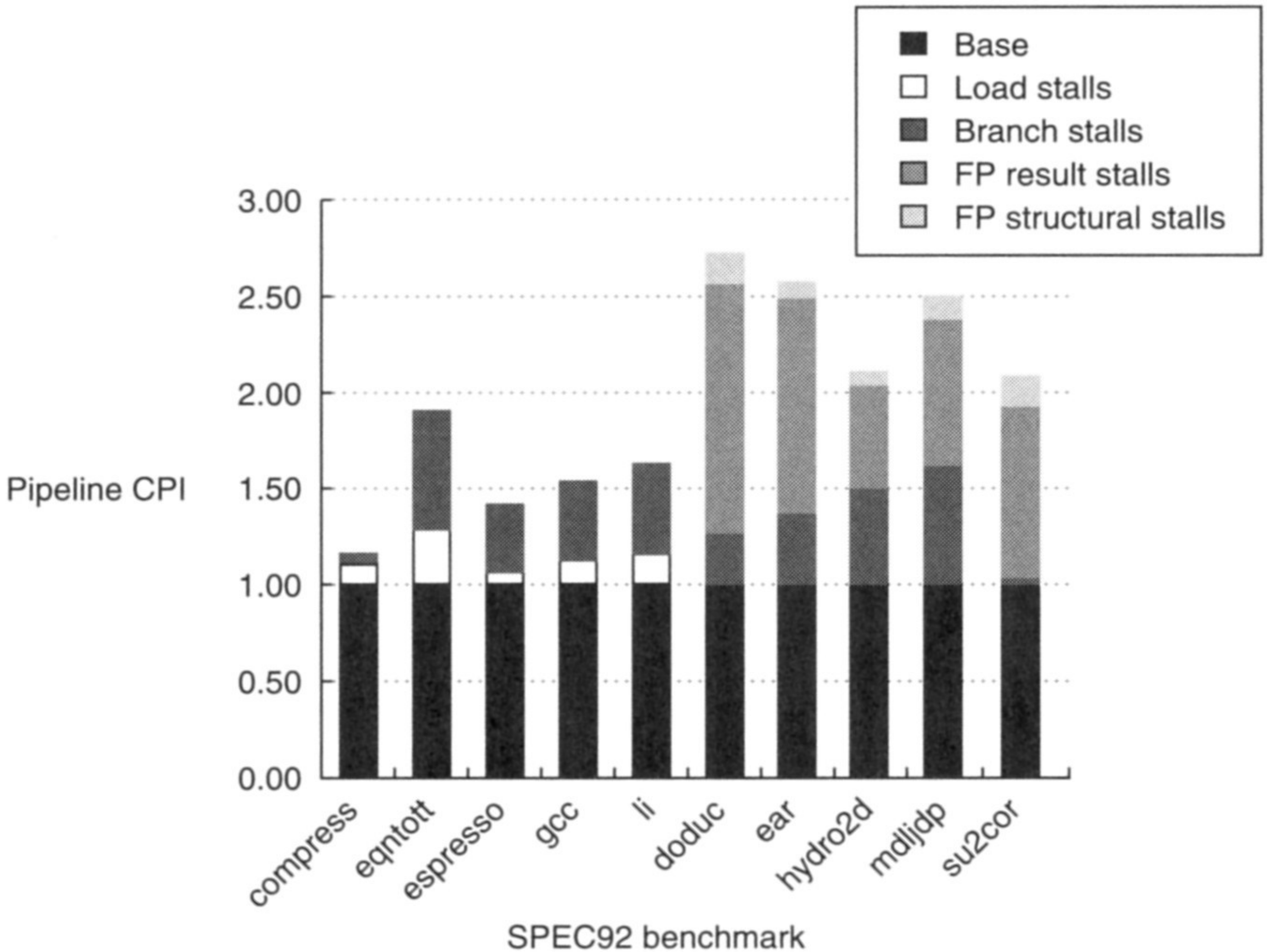
FP instruction	Latency	Initiation interval	Pipe stages
Add, subtract	4	3	U, S + A, A + R, R + S
Multiply	8	4	U, E + M, M, M, M, N, N + A, R
Divide	36	35	U, A, R, D^{27}, D + A, D + R, D + A, D + R, A, R
Square root	112	111	U, E, $(A+R)^{108}$, A, R
Negate	2	1	U, S
Absolute value	2	1	U, S
FP compare	3	2	U, A, R

Latency is reduced by 1 if the destination is a store instruction.

operations

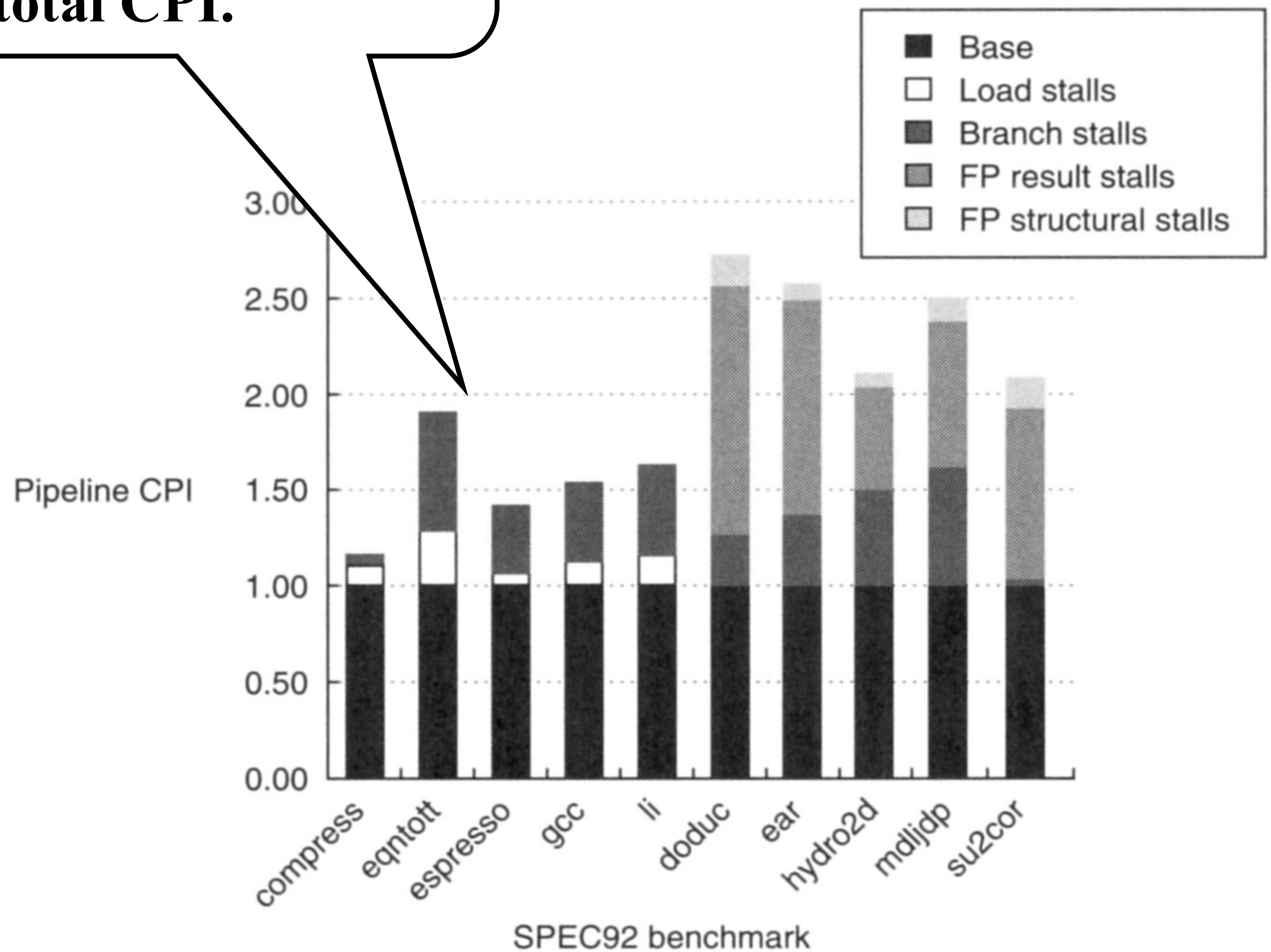
FP instruction	Latency	Initiation interval	Pipe stages
Add, subtract	4	3	U, S + A, A + R, R + S
Multiply	8	4	U, E + M, M, M, M, N, N + A, R
Divide	36	35	U, A, R, D^{27}, D + A, D + R, D + A, D + R, A, R
Square root	112	111	U, E, $(A+R)^{108}$, A, R
Negate	2	1	U, S
Absolute value	2	1	U, S
FP compare	3	2	U, A, R

Performance



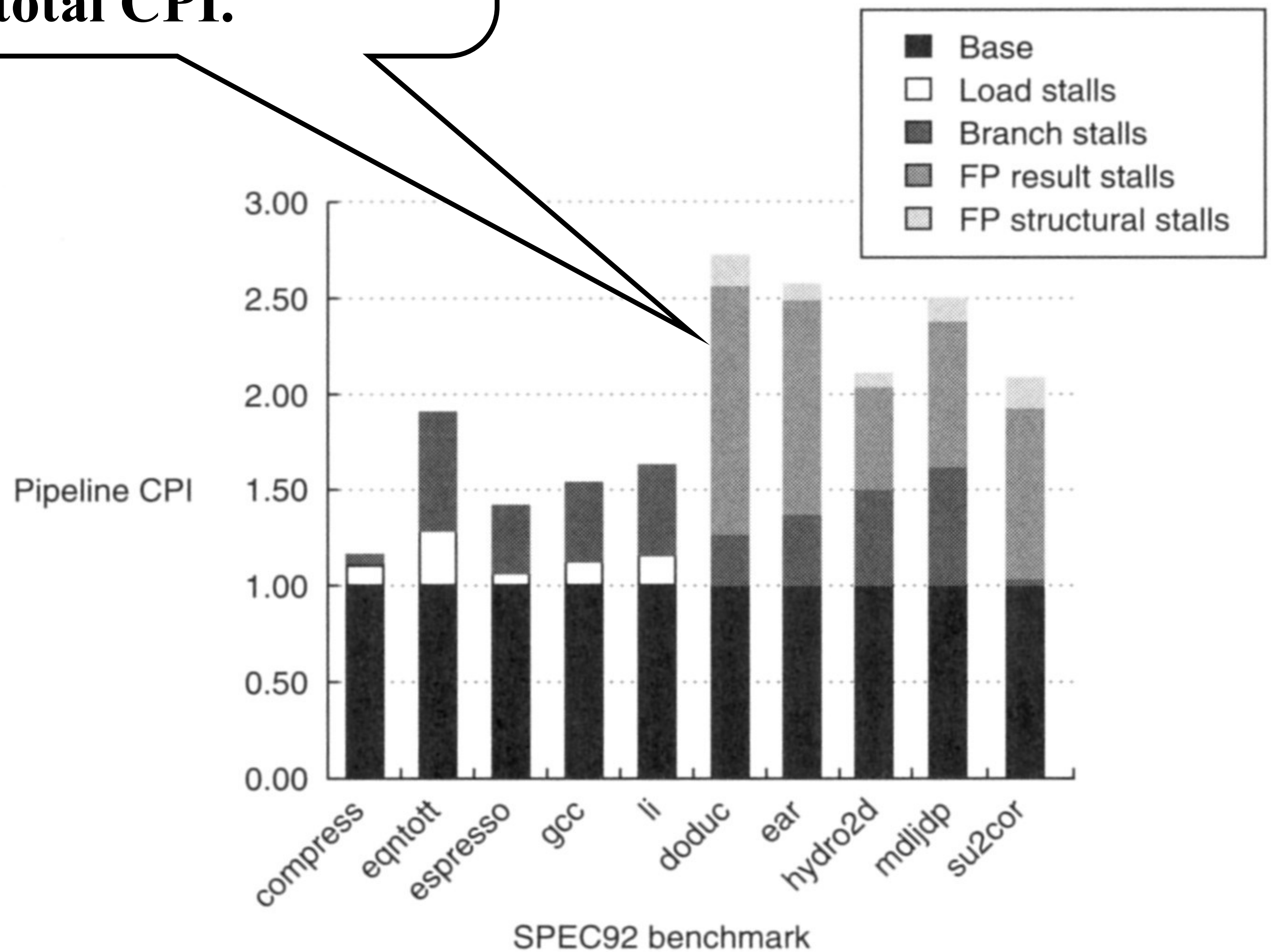
For integer programs branch delays are the most important contributors to total CPI.

formance



For FP programs FP result stalls are the most important contributors to total CPI.

...formance



...formance

Total CPI varies between 1.2 and 2.8, depending on the program.