PIPELINE DESIGN CONSIDERATIONS

COMPUTATIONAL BANDWIDTH*

PRECISE EXCEPTIONS

MINIMUM LATENCY

*** MOST CURRENT RISC MACHINES RETIRE ONE INTEGER OPERATION (ADD/SUBTRACT) PER CYCLE TO AND FROM REGISTERS.**

Hypothetical Non-Pipelined CPU

Instruction	Pipeline Stages									
1	IF	ID	EX	MEM	WB					
2						IF	ID	EX	MEM	WB

Simple machine:

Each instruction takes 5 clock ticks
- IF   Instruction fetch from cache
- ID   Instruction Decode
- EX   Execute (arithmetic, sht, logic, address calculation)
- MEM   Memory operation (lod/store from/to cache)
- WB   Write Back (changes to register, if any)

Cycles-per-instruction = 5 (or more!)

NOBODY BUILDS SUCH THINGS

Single-Issue, 5-Stage RISC Pipeline

Instruction	Pipeline Stages								
1	IF	ID	EX	MEM	WB				
2		IF	ID	EX	MEM	WB			
3			IF	ID	EX	MEM	WB		
4				IF	ID	EX	MEM	WB	
5					IF	ID	EX	MEM	WB

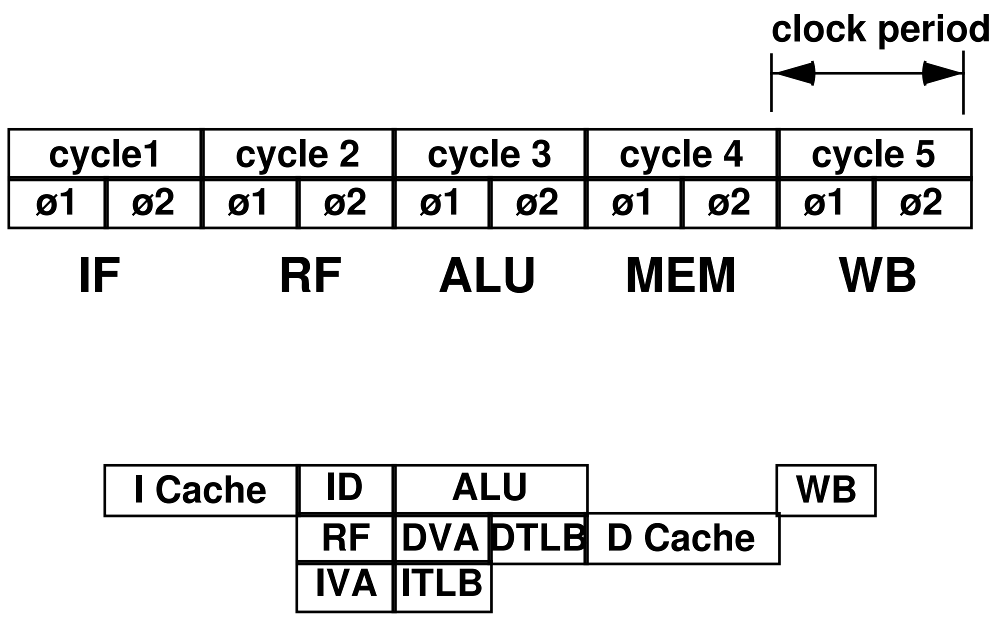
5 (or more) instructions in progress at once (R4000)

Typical of early & current RISCs; some used 4 stages

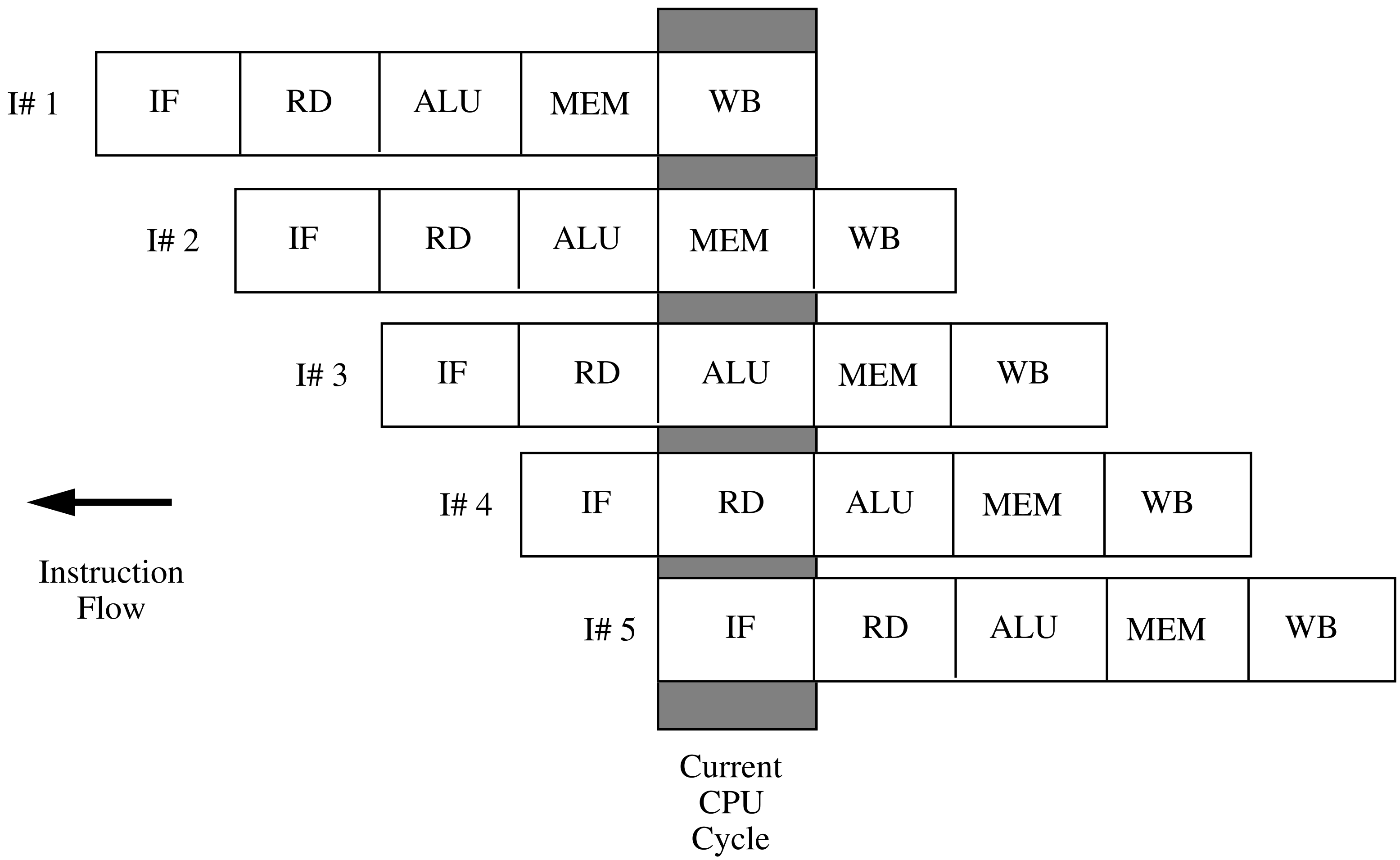
Attempt to issue (start) 1 instruction per cycle
Current & new systems: start 2 or more, sometimes!

Cycles/instruction = 1 (at best), 1.2-1.6 more typical

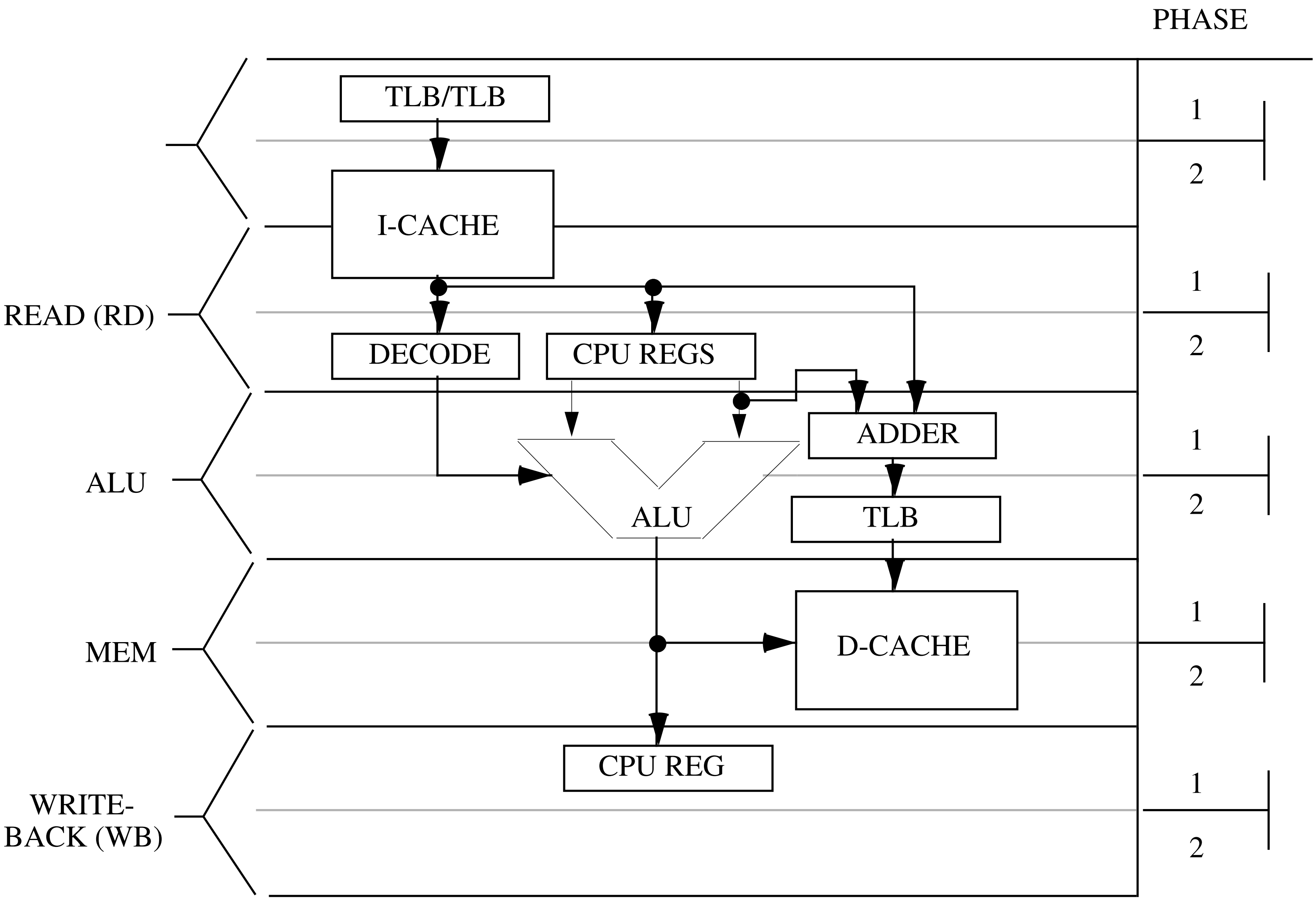
FIVE STAGE PIPELINE



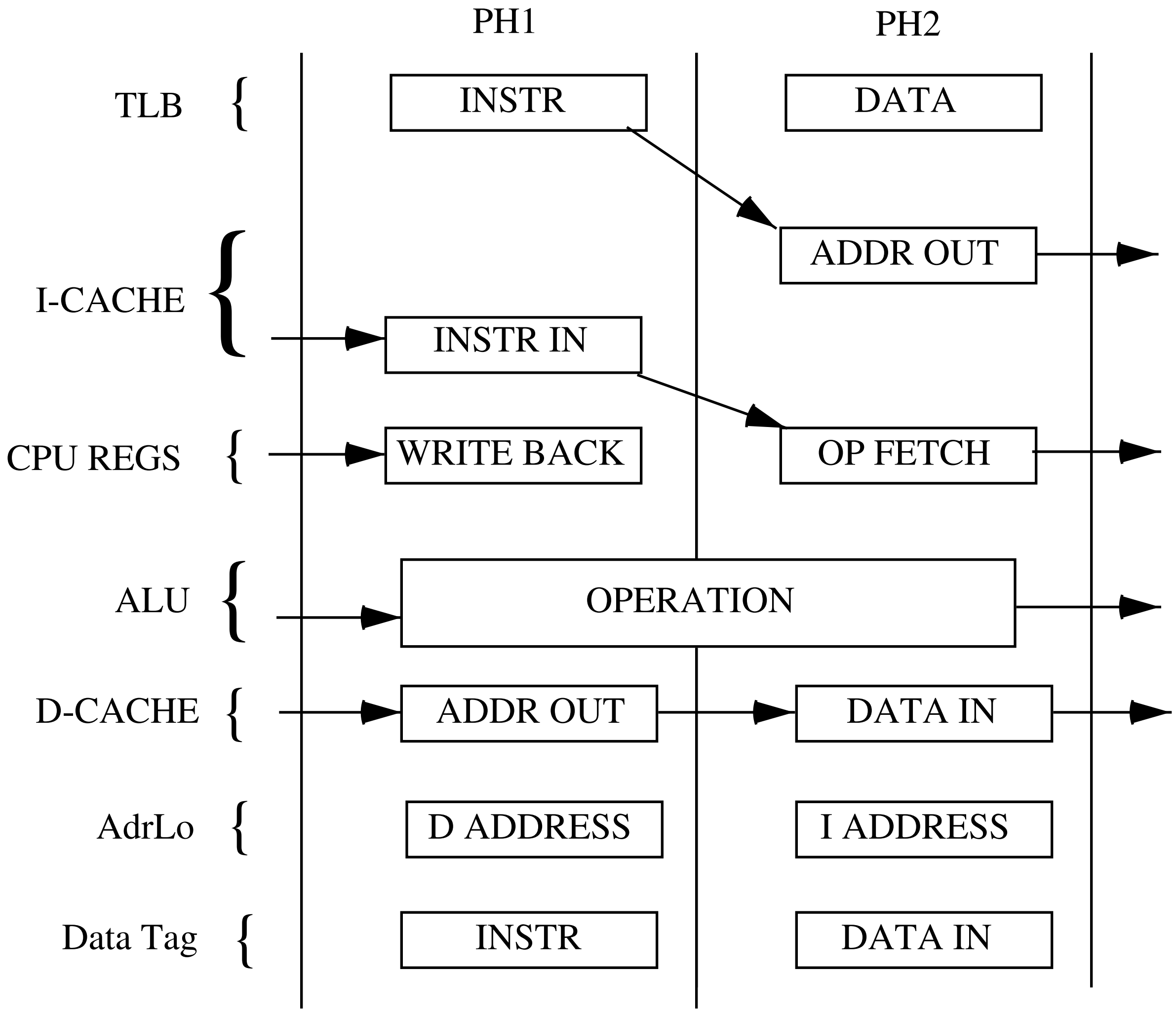
LR3000 CPU PIPELINE



LR3000 PIPELINE (SIMPLIFIED)



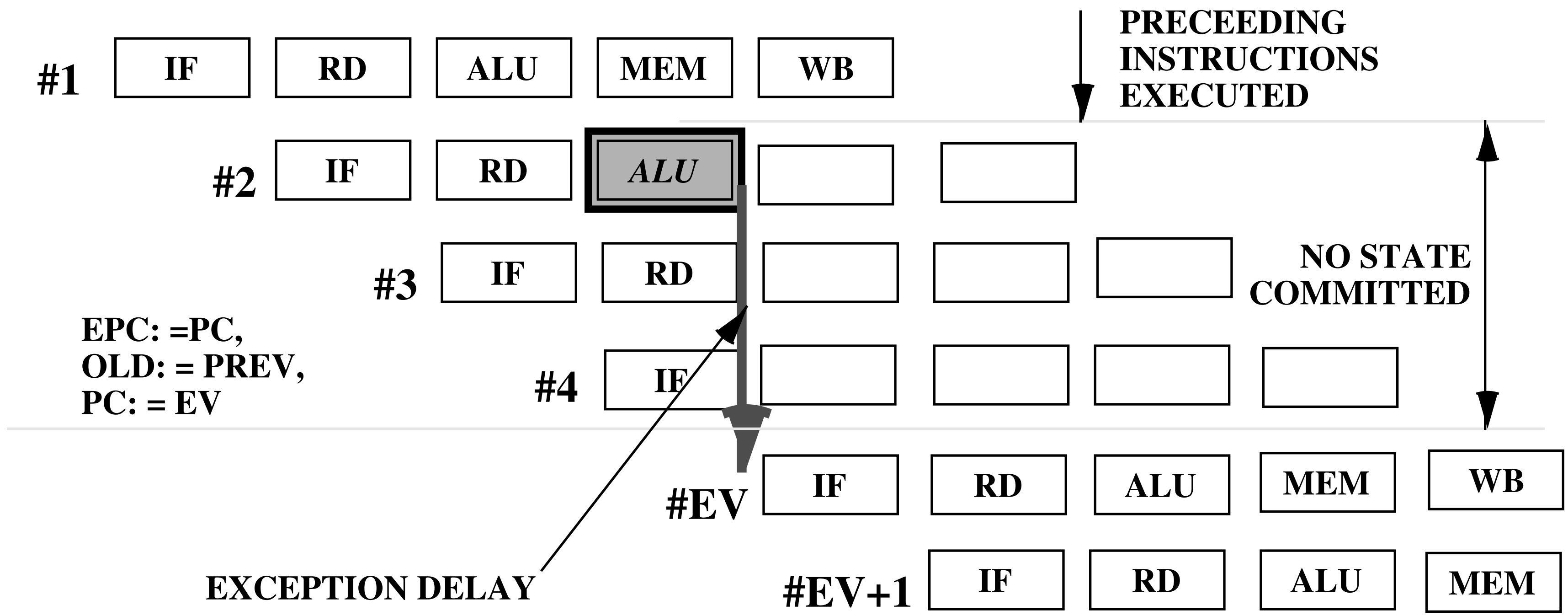
CPU RESOURCE UTILIZATION



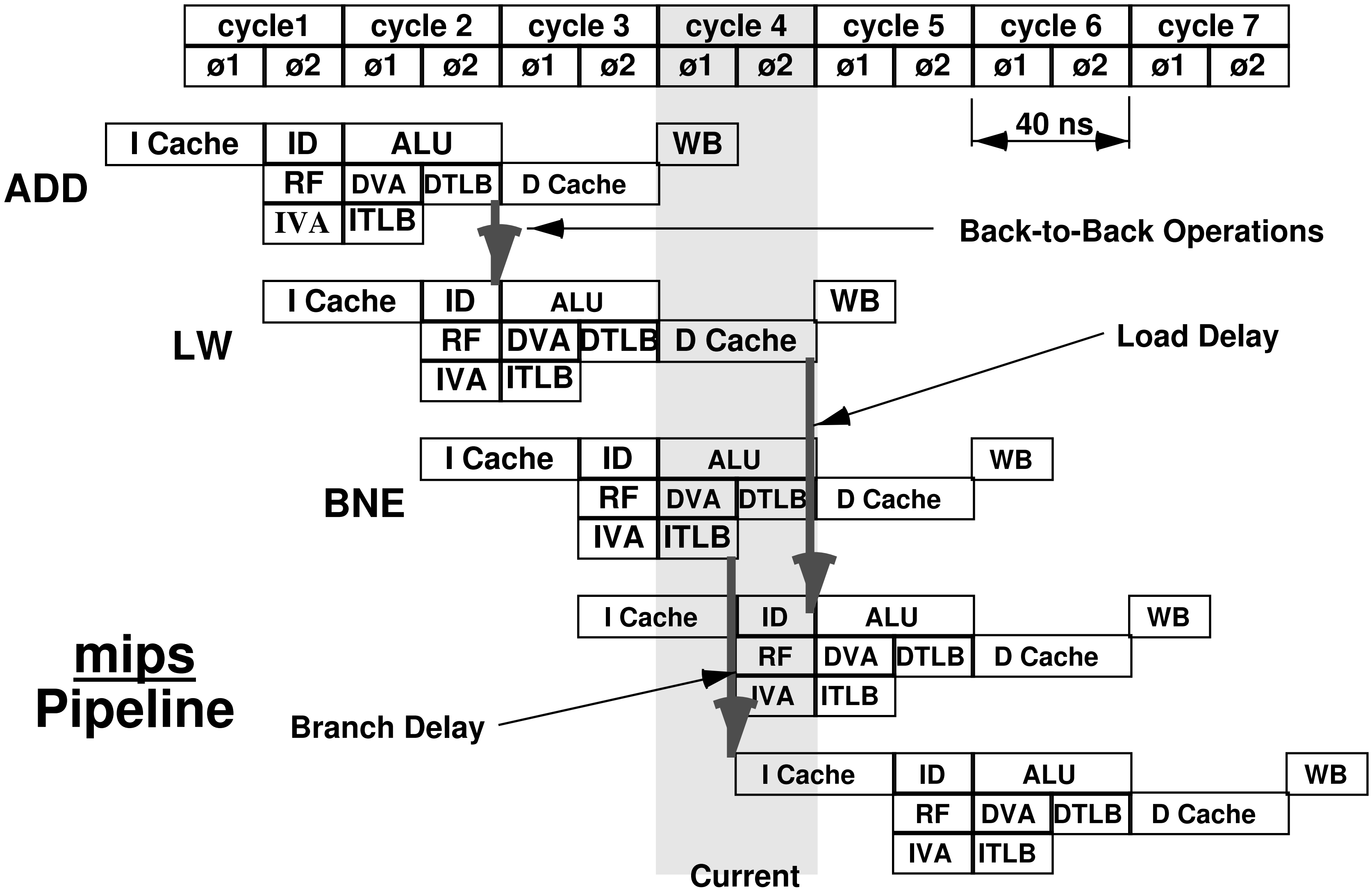
PIPELINE STALLS

IF	RD	ALU	MEM	WB					
	IF	RD	ALU	*MEM*	MEM	MEM	MEM	WB	
		IF	RD	ALU	stall	stall	stall	MEM	WB
			IF	RD	stall	stall	stall	ALU	MEM
				IF	stall	stall	stall	RD	ALU
								IF	RD
									IF

PRECISE EXCEPTION HANDLING



CRITICAL PIPELINE & SYSTEM DESIGN FACTOR
HARDWARE MAKES ALL EXCEPTIONS PRECISE



mips
Pipeline

Instruction Scheduling Example

a = b + 1; if (c == 0) d = 0;

Straightforward code

```
lw      r2, b            # r2 = b
nop                      # load slot
add     r2, 1            # r2 = b + 1
sw      r2, a            # a = b + 1
lw      r3, c            # r3 = c
nop                      # load slot
bne     r3, zero, 1f     # if (c != 0) skip
nop                      # branch slot
sw      zero, d          # if (c == 0) d = 0
1: ...
```

Scheduled Code

```
lw      r2, b            # r2 = b
lw      r3, c            # r3 = c
add     r2, 1            # r2 = b + 1
bne     r3, zero 1f      # if ( c != 0) skip
sw      r2, a            # a = b + 1
sw      zero, d          # if (c == 0) d = 0
1: ...
```

LATENCY COMPARISONS

*** ESTIMATED CPI PRIOR TO SYSTEMS AVAILABLE FOR MEASUREMENT**

****BEST CASE ASSUMING MIPS COMPILERS; 70% FILL 1st DELAY SLOT, 30% 2nd DELAY SLOT**

***** SPARC HAS A 2 CYCLE LOAD, BUT IF THE DATA IS USED, ONE MORE CYCLE IS NEEDED**

TYPE	FREQUENCY	MIPS		SPARC		MOTOROLA	
		cycles	cpi	cycles	cpi	cycles	cpi
alu	.55	1.0	.55	1.0	.55	1.0	.55
load	.20	1.3	.26	2.3***	.46	1.8**	.36
store	.10	1.0	.10	3.0	.30	1.0	.10
branch	.15	1.5	.22	2.5	.37	1.5	.22
IDEAL CYCLES/RISC INST.			1.13		1.68		1.23
ESTIMATED MHz/VUP			1.25		2.0		1.5*