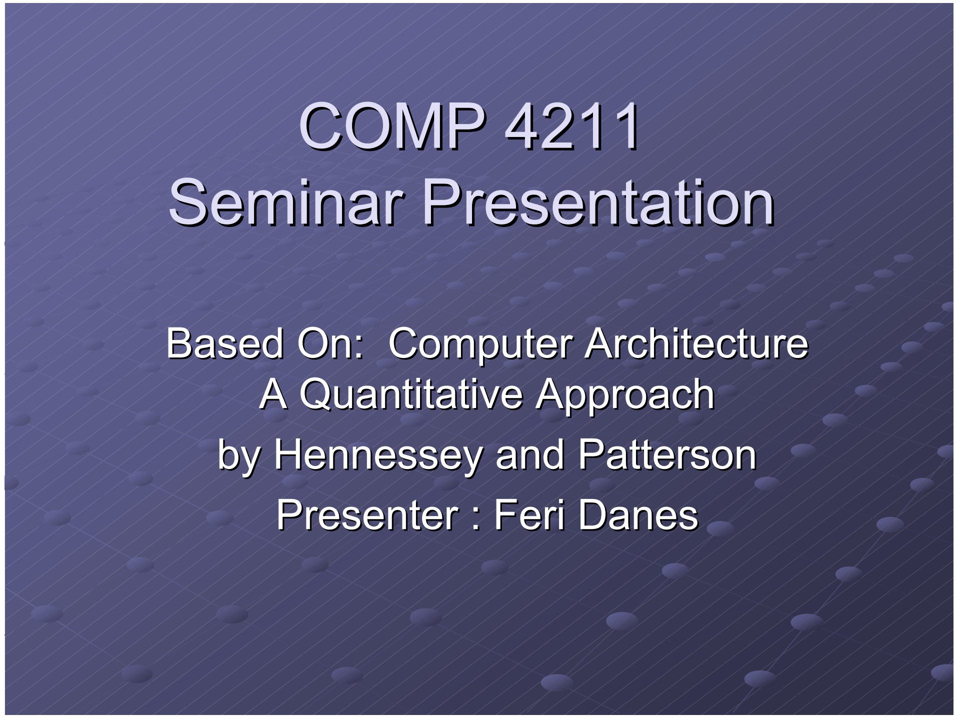
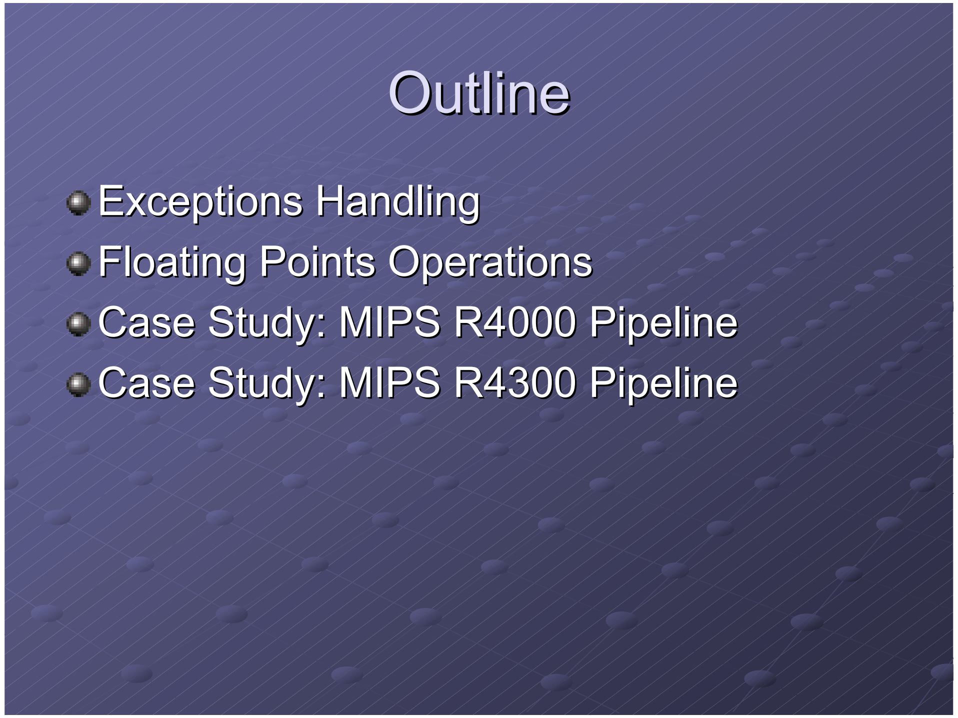


COMP 4211
Seminar Presentation

Based On: Computer Architecture
A Quantitative Approach

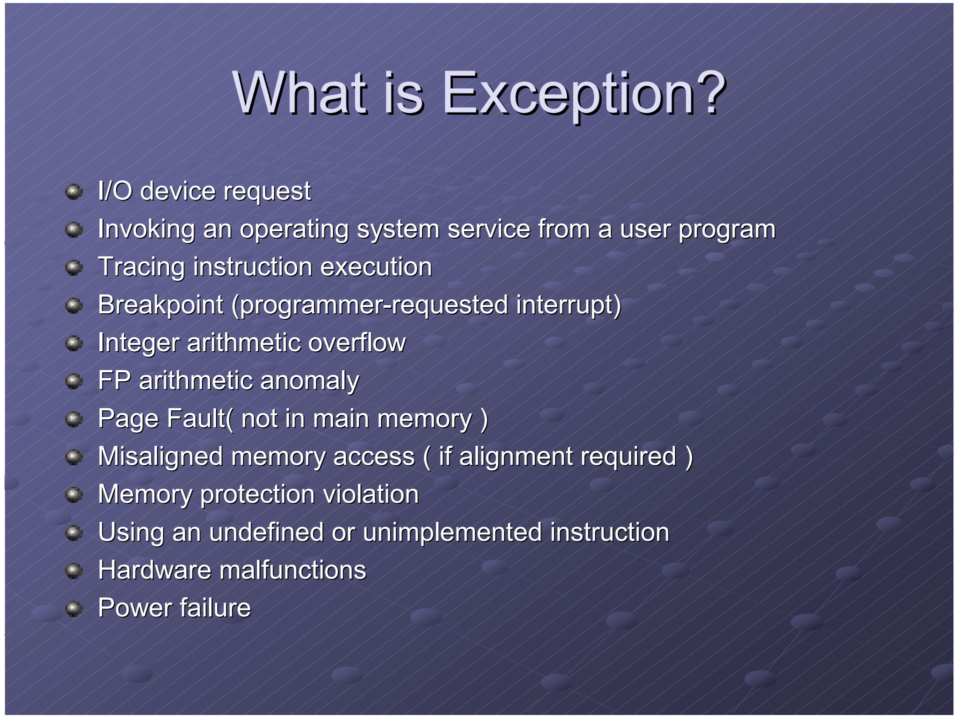
by Hennessey and Patterson

Presenter : Feri Danes



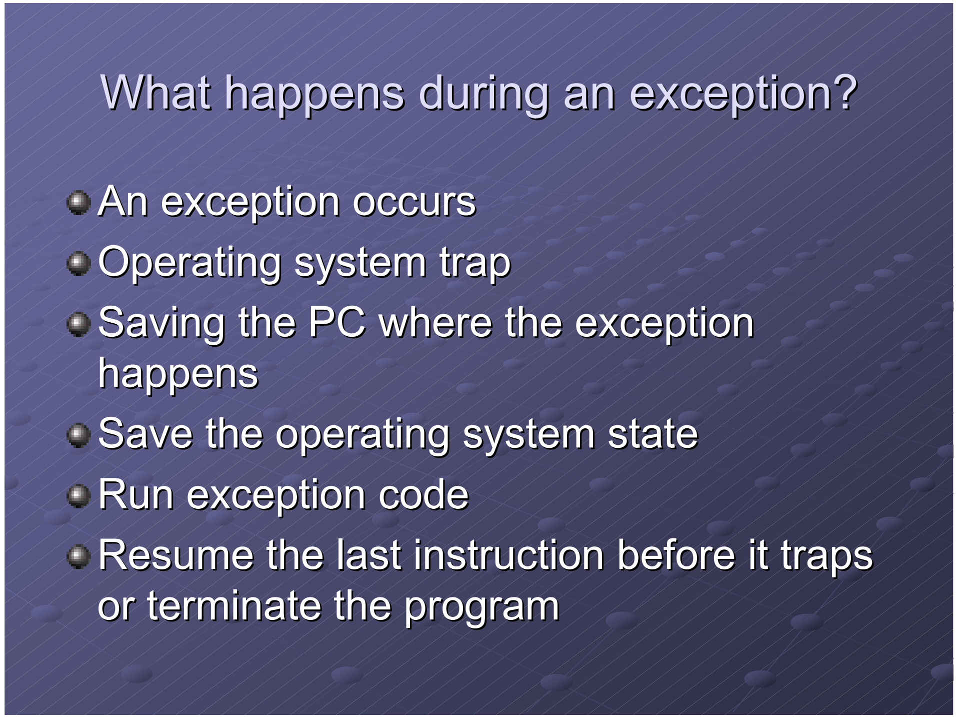
Outline

- Exceptions Handling
- Floating Points Operations
- Case Study: MIPS R4000 Pipeline
- Case Study: MIPS R4300 Pipeline



What is Exception?

- I/O device request
- Invoking an operating system service from a user program
- Tracing instruction execution
- Breakpoint (programmer-requested interrupt)
- Integer arithmetic overflow
- FP arithmetic anomaly
- Page Fault(not in main memory)
- Misaligned memory access (if alignment required)
- Memory protection violation
- Using an undefined or unimplemented instruction
- Hardware malfunctions
- Power failure



What happens during an exception?

- An exception occurs
- Operating system trap
- Saving the PC where the exception happens
- Save the operating system state
- Run exception code
- Resume the last instruction before it traps or terminate the program

How does this influence the hardware?

- Unpipelined implementations
  - Occur within instructions
  - Restartable
- Pipelined implementations
  - Preserves the CPU state by stalling the instruction following the exception source

Exception in Pipelined architecture

- Force a trap instruction into the pipeline on the next IF
- Flush the pipeline for the faulting instruction and all instructions that follows
- After exception handling routine finishes restore the PC of the saved PC and delay branches if exsists

Precise Exceptions

ìif the pipeline can be stopped so that the instructions just the faulting instruction are completed and the faulting instruction can be restarted from scratchî

Exceptions in MIPS

Pipeline stage	Problem exceptions occurring
IF	Page fault on IF, misaligned memory access, memory protection violation
ID	Undefined or illegal opcode
EX	Arithmetic exception
MEM	Page fault on data fetch, misaligned memory access, memory violation protection
WB	None

Exceptions in MIPS

- Precise exceptions
- Exception status vector
- eg:
  - Data fault on LD
  - Page fault on DADD

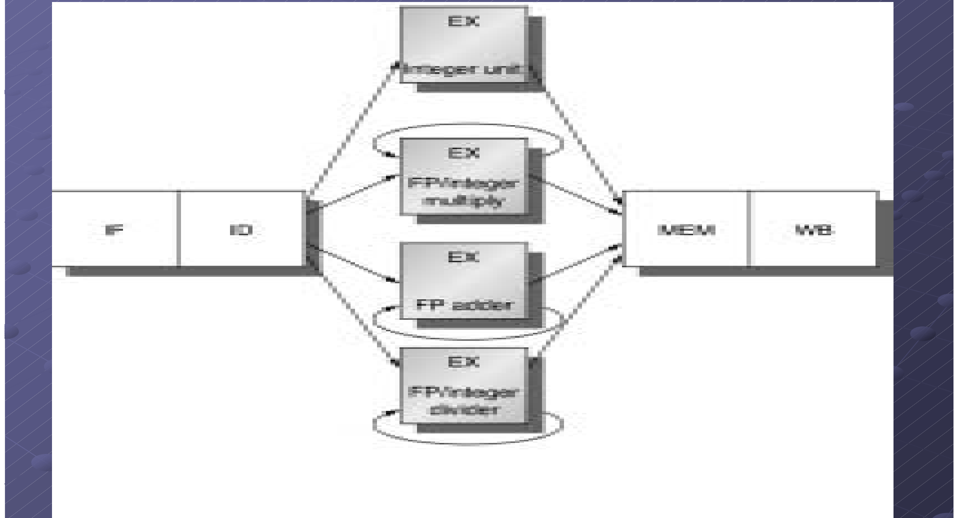
LD	IF	ID	EX	MEM	WB	
ADD		IF	ID	EX	MEM	WB

Floating Points Operation

There are four separate functional units that we can assume to support Function Point operations:

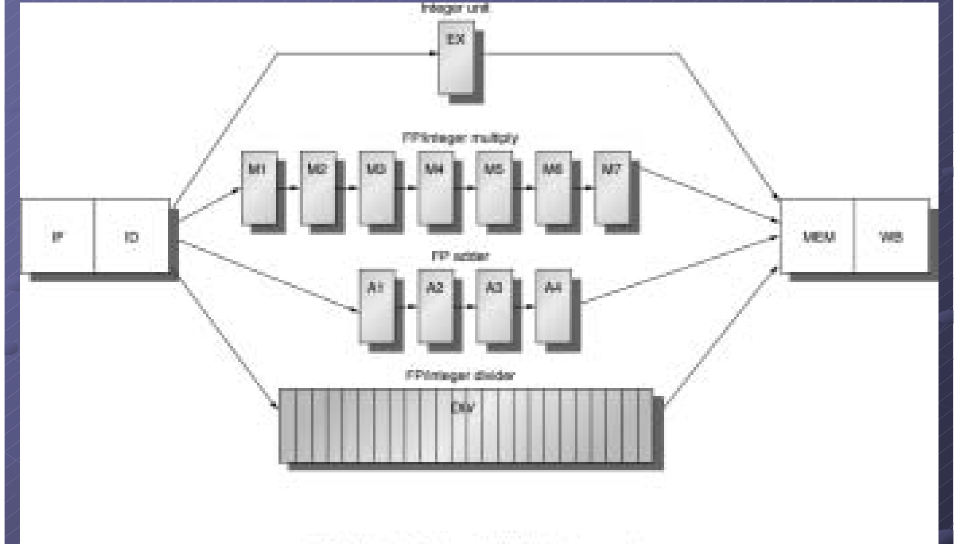
- Integer Unit
- FP and integer multiplier
- FP adder
- FP and integer divider

Floating Points Operation



© 2003 Elsevier Science (USA). All rights reserved.

Function Point Operations



© 2003 Elsevier Science (USA). All rights reserved.

Hazards

- Structural hazards on the FP and integer divider
- Structural hazards on WB
- WAW because of out of order completion
- Precise exception is harder to maintain
- Increase the number of RAW hazards

Structural Hazard

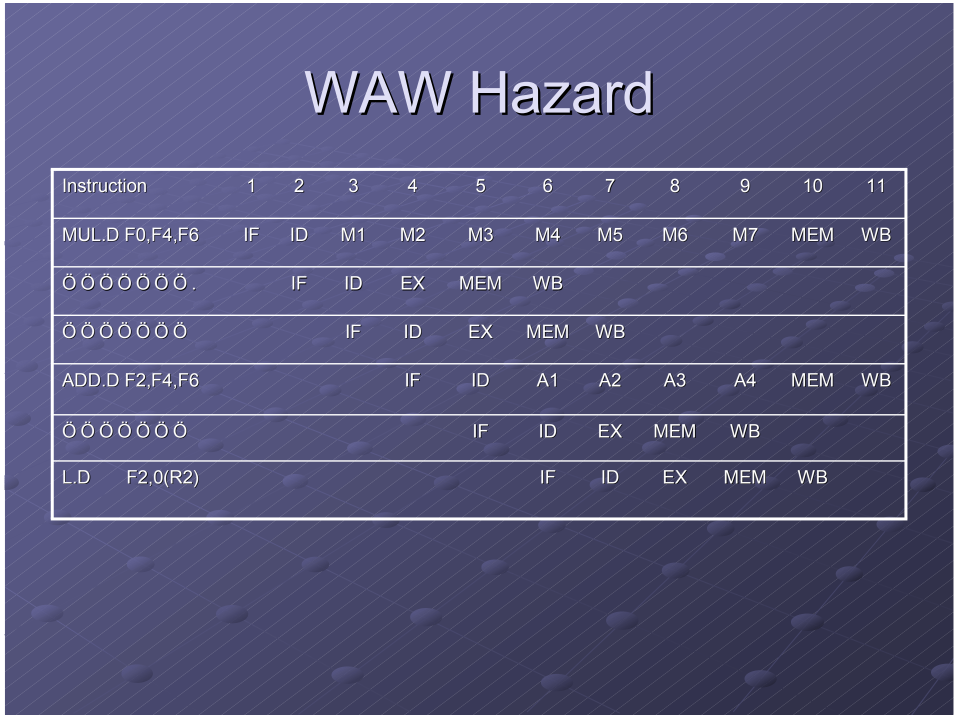
Instruction	1	2	3	4	5	6	7	8	9	10	11
MUL.D F0,F4,F6	IF	ID	M1	M2	M3	M4	M5	M6	M7	MEM	WB
ÖÖÖÖÖÖÖÖ.		IF	ID	EX	MEM	WB					
ÖÖÖÖÖÖÖÖ.			IF	ID	EX	MEM	WB				
ADD.D F2,F4,F6				IF	ID	A1	A2	A3	A4	MEM	
ÖÖÖÖÖÖÖÖ.					IF	ID	EX	MEM	WB		
ÖÖÖÖÖÖÖÖ.						IF	ID	EX	MEM	WB	
L.D F2,0(R2)							IF	ID	EX	MEM	WB

Structural Hazard

- Stall the instruction that can cause structural hazard in the ID stage
- Stall the instruction when it enters the MEM or WB stage

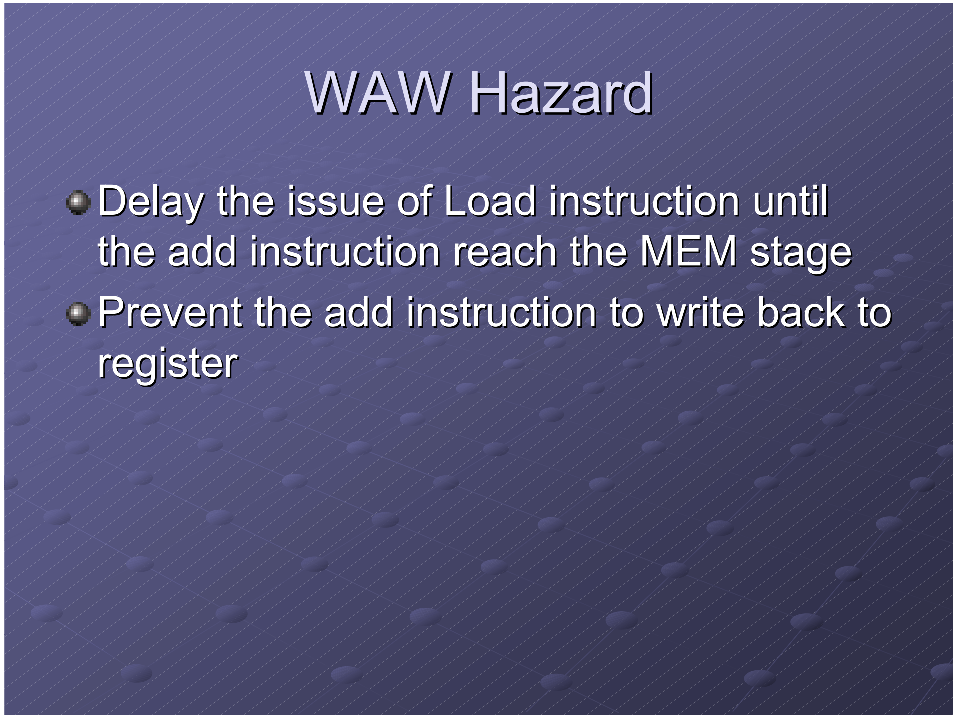
RAW Hazard

Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
L.D F0F4,F6	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	
S.D F2,0(R2)					IF	stall	Stall	stall	stall	Stall	Stall	ID	EX	stall	stall	stall	MEM



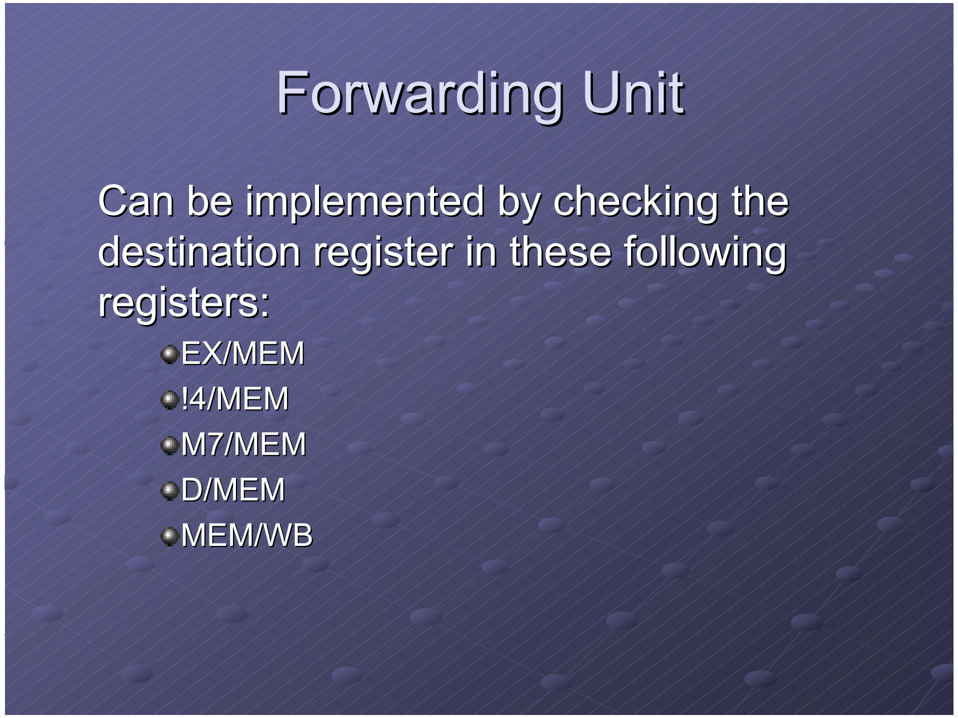
WAW Hazard

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
Ö Ö Ö Ö Ö Ö Ö .		IF	ID	EX	MEM	WB					
Ö Ö Ö Ö Ö Ö Ö			IF	ID	EX	MEM	WB				
ADD.D F2,F4,F6				IF	ID	A1	A2	A3	A4	MEM	WB
Ö Ö Ö Ö Ö Ö Ö					IF	ID	EX	MEM	WB		
L.D F2,0(R2)						IF	ID	EX	MEM	WB	



WAW Hazard

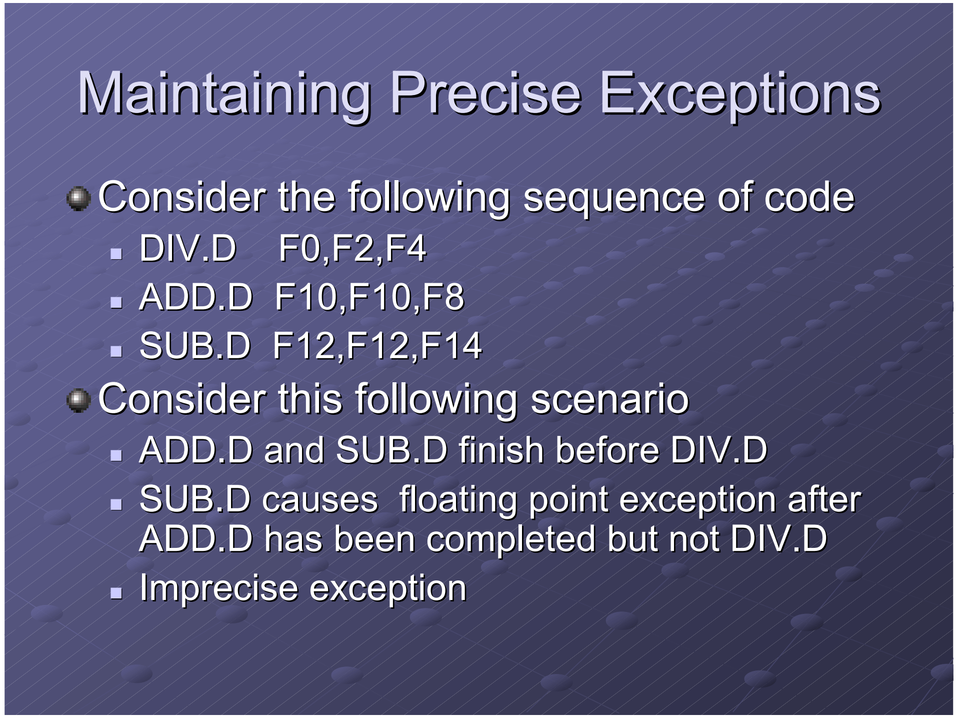
- Delay the issue of Load instruction until the add instruction reach the MEM stage
- Prevent the add instruction to write back to register



Forwarding Unit

Can be implemented by checking the destination register in these following registers:

- EX/MEM
- !4/MEM
- M7/MEM
- D/MEM
- MEM/WB



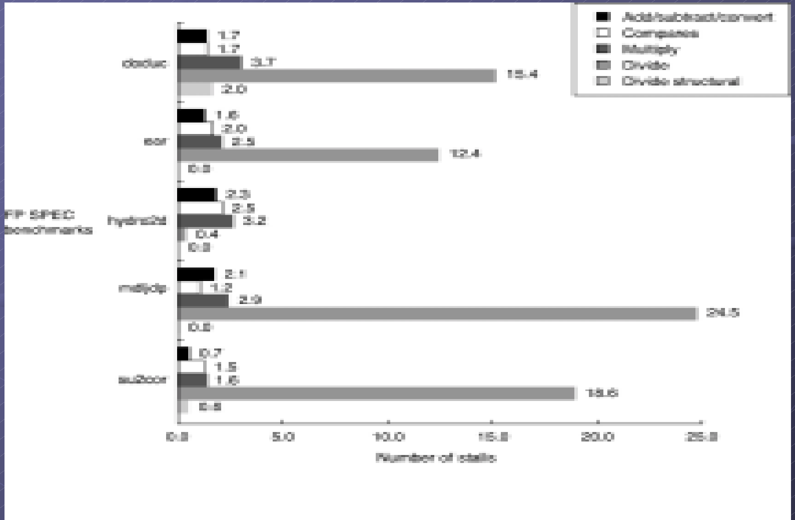
Maintaining Precise Exceptions

- Consider the following sequence of code
  - DIV.D F0,F2,F4
  - ADD.D F10,F10,F8
  - SUB.D F12,F12,F14
- Consider this following scenario
  - ADD.D and SUB.D finish before DIV.D
  - SUB.D causes floating point exception after ADD.D has been completed but not DIV.D
  - Imprecise exception

Maintaining Precise Exception

- Ignore
- buffer the result of an operation until all operation that were issued earlier are complete
  - History file
  - future file
- Let exceptions to be imprecise but with state information
  - save the instruction that precede the completed instruction and run those instructions before restarting the execution
- Guarantee that no instruction that precede the issuing instruction can be completed without exception before issuing that instruction

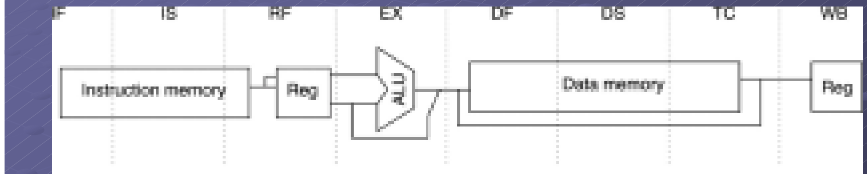
MIPS FP Performance



Case Study MIPS R4000

- Eight stages pipeline
- Additional pipeline is allocated for cache access
- Often called as *superpipelining*

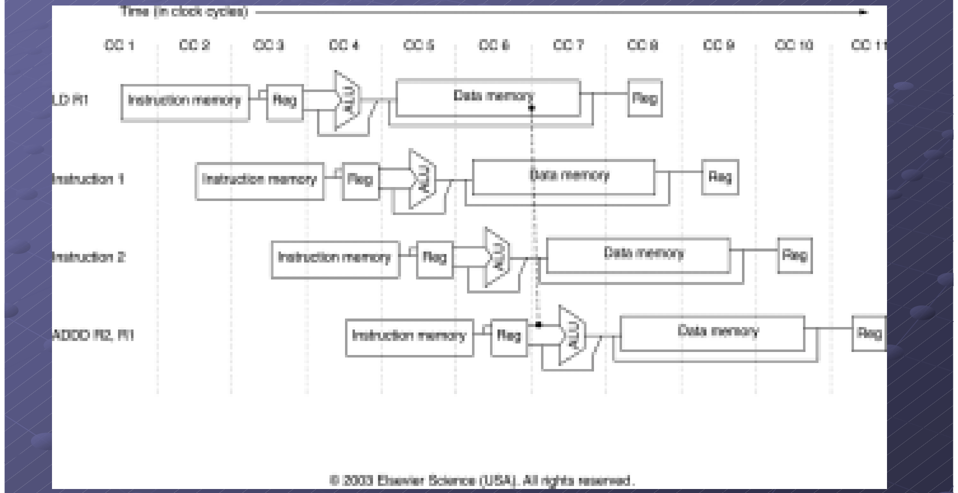
R4000 Pipeline



R4000 Pipeline

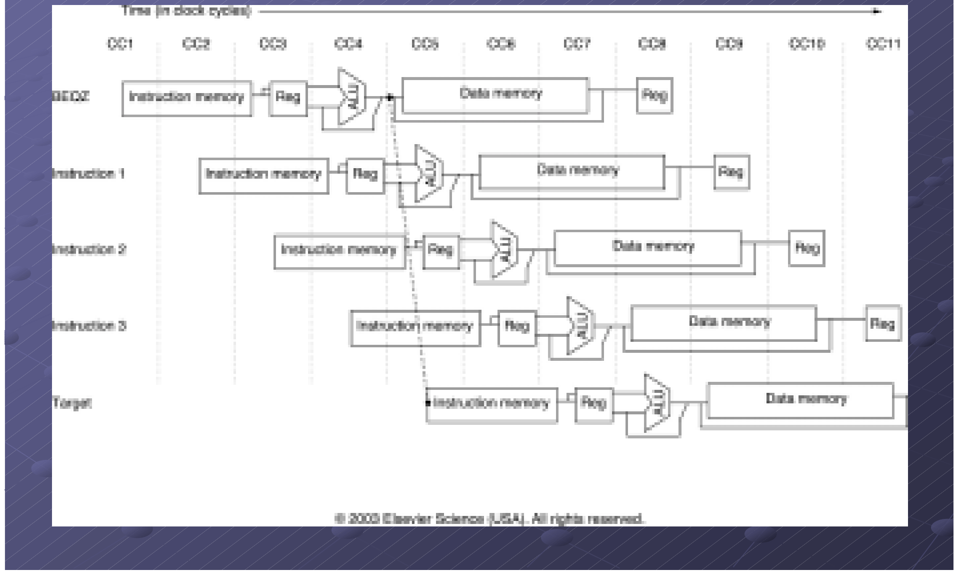
- IF--First half of instruction fetch, PC selection and initiation of instruction cache access
- IS--Second half instruction fetch, complete instruction cache access
- RF--Instruction decode and register fetch and instruction cache hit detection
- EX--Execution includes effective address calculation, ALU operation, branch target calculation and branch condition calculation
- DF--Data fetch first half
- DS--Second half data fetch completion of data cache access
- TC--Tag check, determine whether the data cache access hit
- WB--Write Back for loads and register-register operation

RAW Hazard



© 2003 Elsevier Science (USA). All rights reserved.

Branch Delay

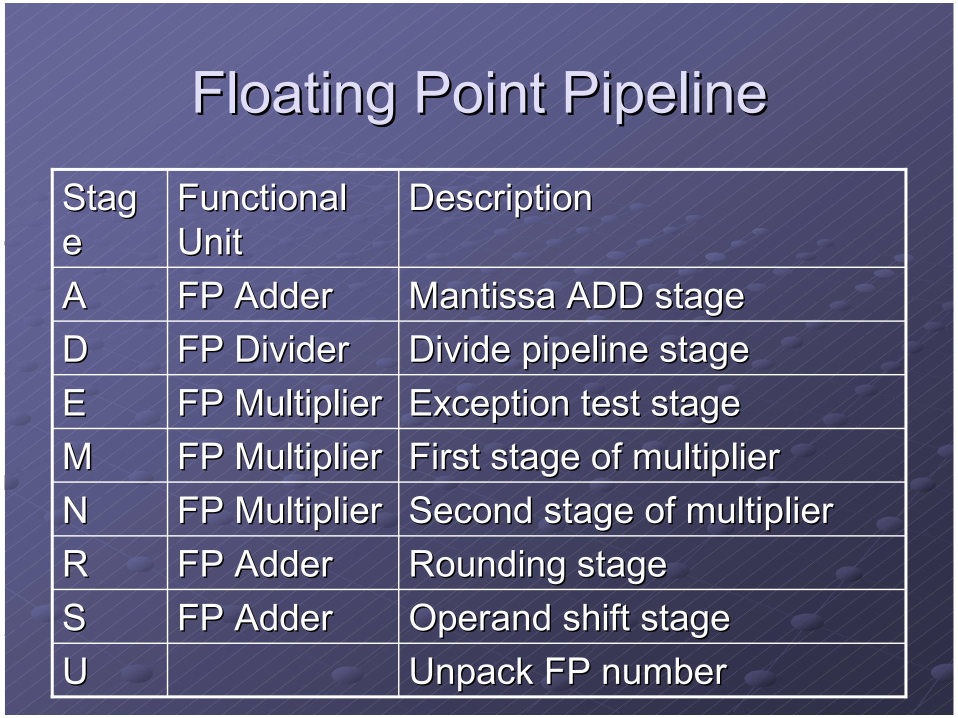


© 2003 Elsevier Science (USA). All rights reserved.

Branch Delay

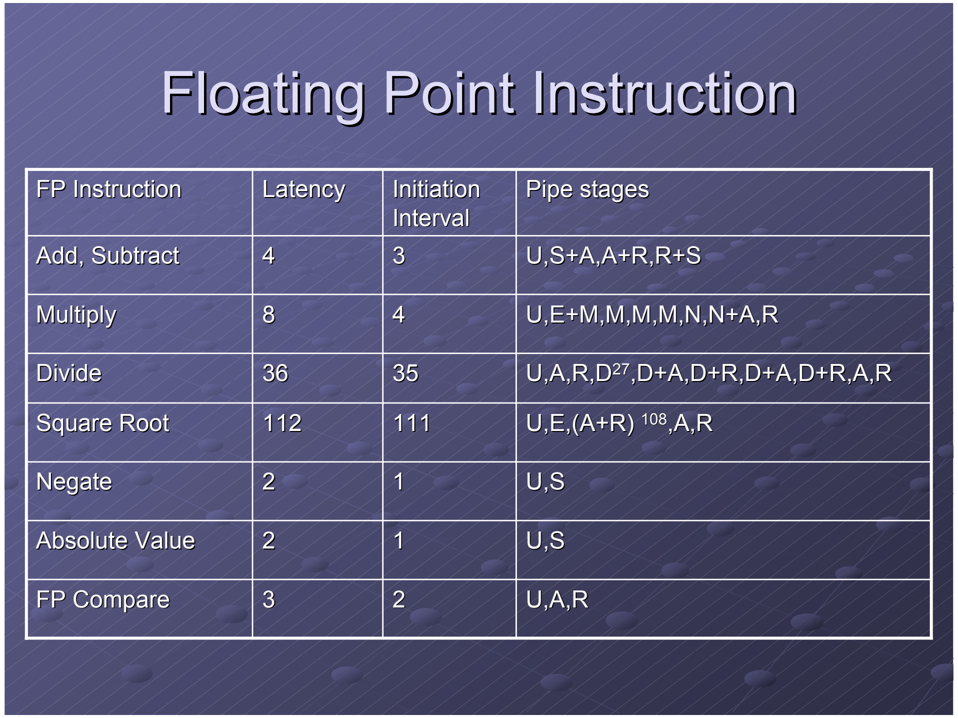
- Branch taken
  - 1 Delay Slot + 2 stall
- Brach not taken
  - 1 Delay Slot

Note : Branch predicted not taken policy is used for the 2 out 3 cycles of branch delay



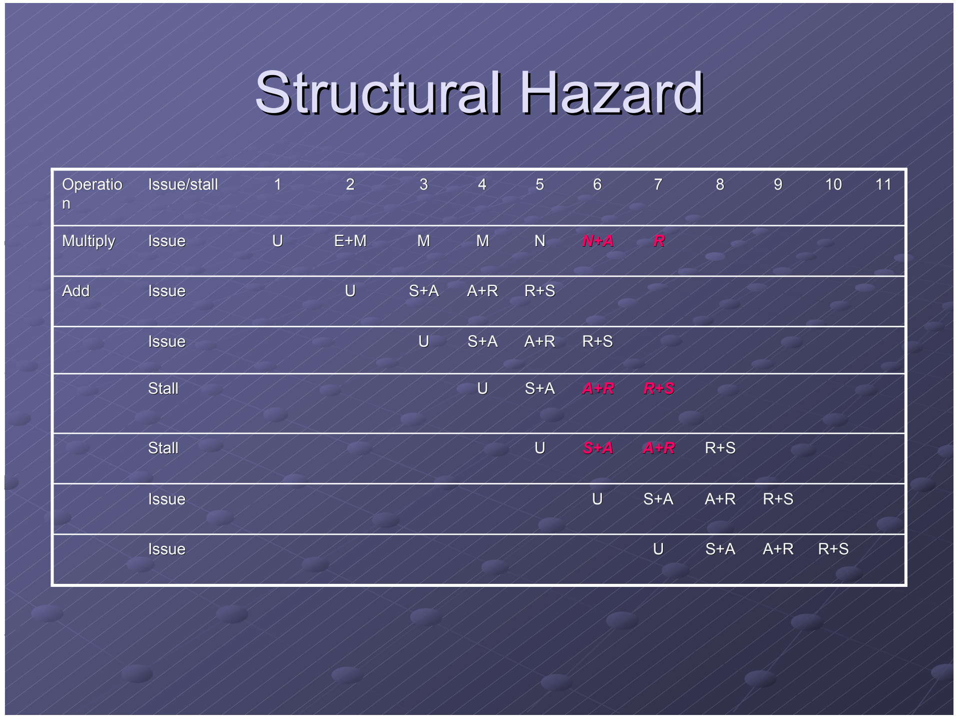
Floating Point Pipeline

Stage	Functional Unit	Description
A	FP Adder	Mantissa ADD stage
D	FP Divider	Divide pipeline stage
E	FP Multiplier	Exception test stage
M	FP Multiplier	First stage of multiplier
N	FP Multiplier	Second stage of multiplier
R	FP Adder	Rounding stage
S	FP Adder	Operand shift stage
U		Unpack FP number



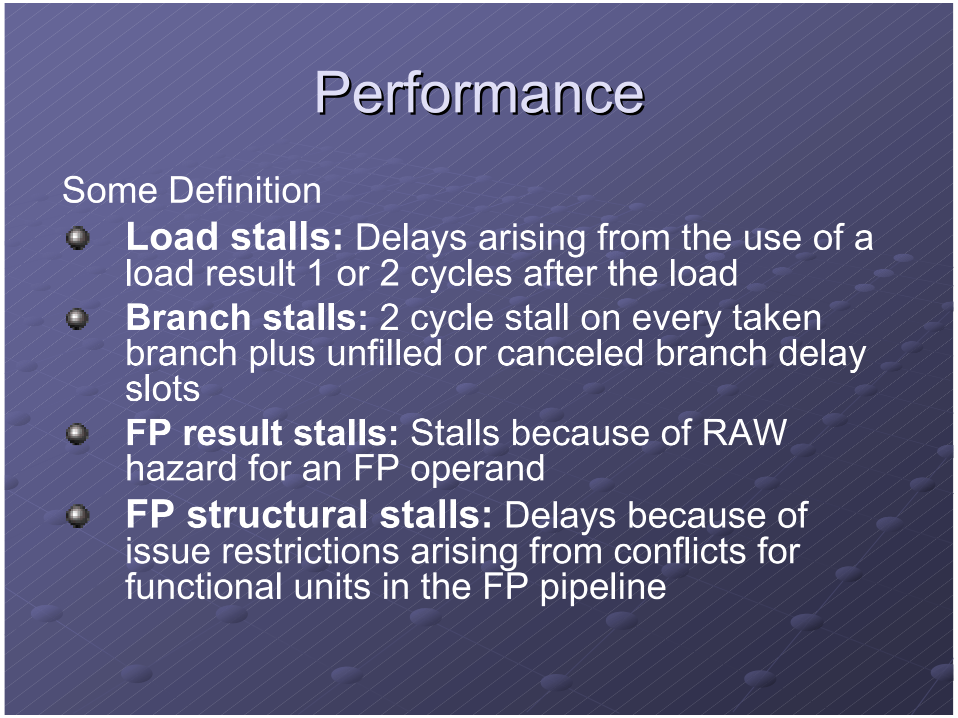
Floating Point Instruction

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



Structural Hazard

Operation	Issue/stall	1	2	3	4	5	6	7	8	9	10	11
Multiply	Issue	U	E+M	M	M	N	***N+A***	***R***				
Add	Issue		U	S+A	A+R	R+S						
	Issue			U	S+A	A+R	R+S					
	Stall				U	S+A	***A+R***	***R+S***				
	Stall					U	***S+A***	***A+R***	R+S			
	Issue						U	S+A	A+R	R+S		
	Issue							U	S+A	A+R	R+S	

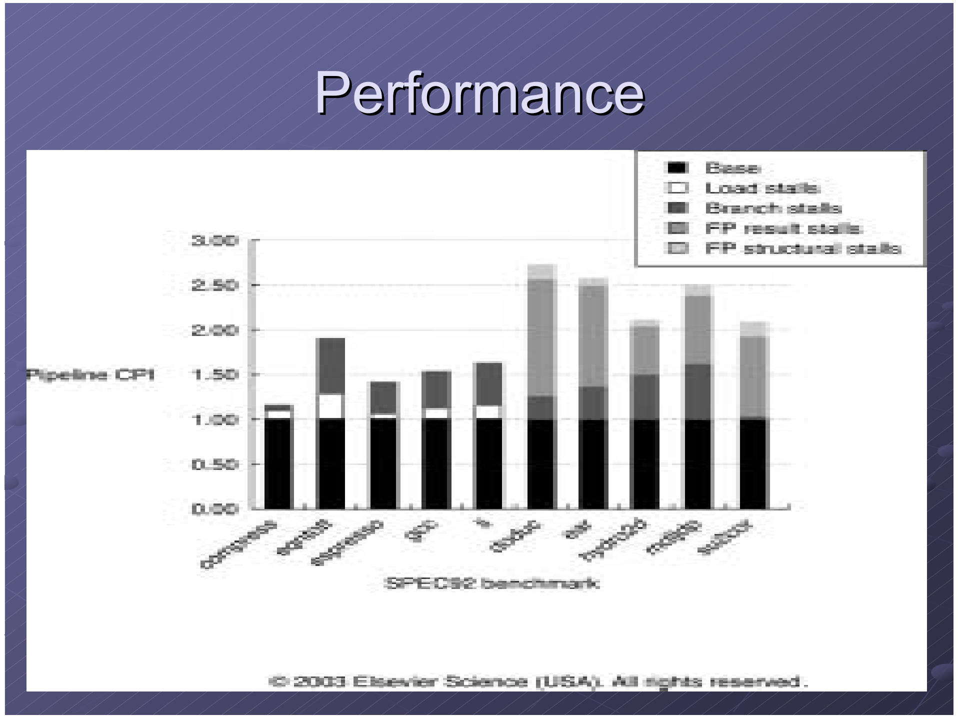


Performance

Some Definition

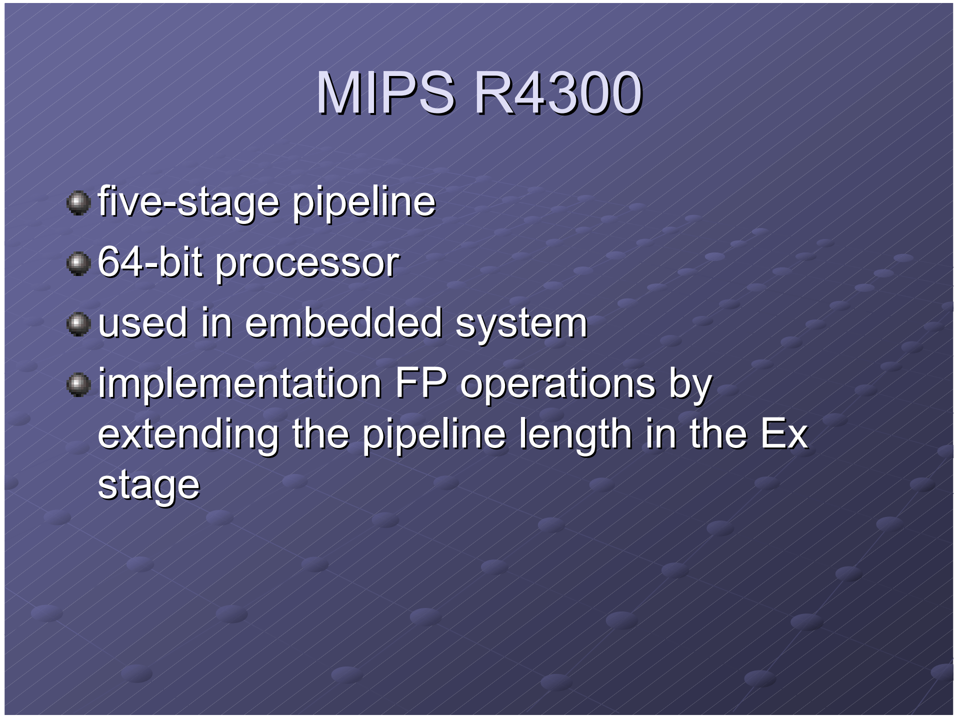
- **Load stalls:** Delays arising from the use of a load result 1 or 2 cycles after the load
- **Branch stalls:** 2 cycle stall on every taken branch plus unfilled or canceled branch delay slots
- **FP result stalls:** Stalls because of RAW hazard for an FP operand
- **FP structural stalls:** Delays because of issue restrictions arising from conflicts for functional units in the FP pipeline

Performance



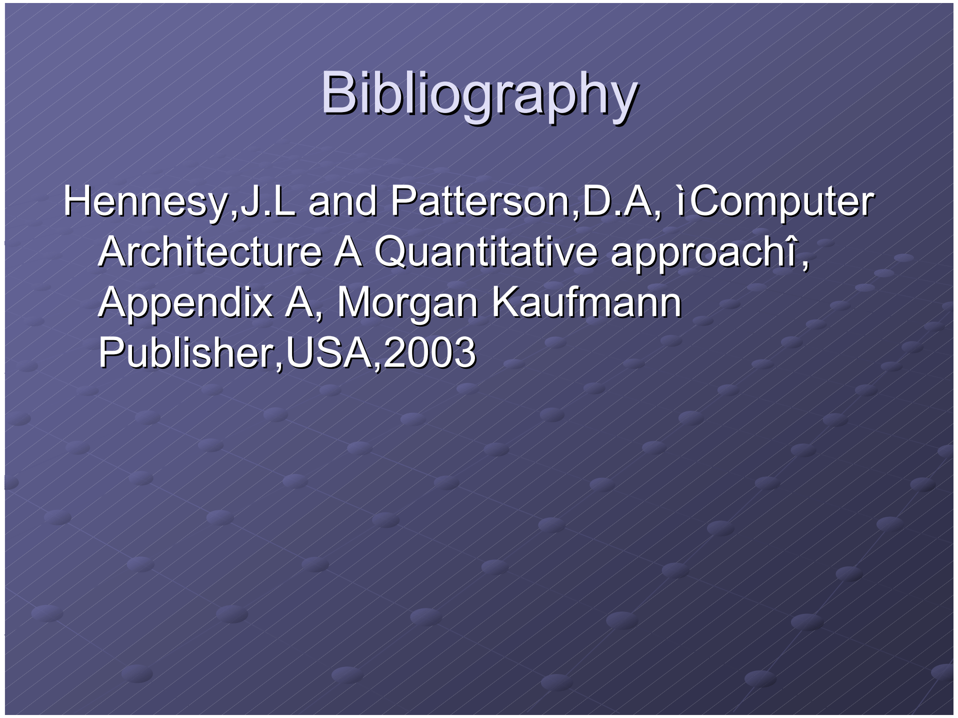
© 2003 Elsevier Science (USA). All rights reserved.

MIPS R4300



- five-stage pipeline
- 64-bit processor
- used in embedded system
- implementation FP operations by extending the pipeline length in the Ex stage

Bibliography



Hennesy,J.L and Patterson,D.A, ìComputer Architecture A Quantitative approachî, Appendix A, Morgan Kaufmann Publisher,USA,2003