cin.ufpe.br

Centro de Informática

U · F · P · E

Universidade Federal de Pernambuco

Processadores Superescalares

Melhorando ainda mais o desempenho....

Roteiro da Aula

- Pipeline e filosofia RISC
- Superpipeline
- Superscalar
- Dependências de dados em processadores superscalares
- Renomeamento de registradores
- VLIW
- Análise Comparativa

Pipeline

- Porque o pipeline é implementado de forma mais eficiente em processadores da classe do processador MIPS?

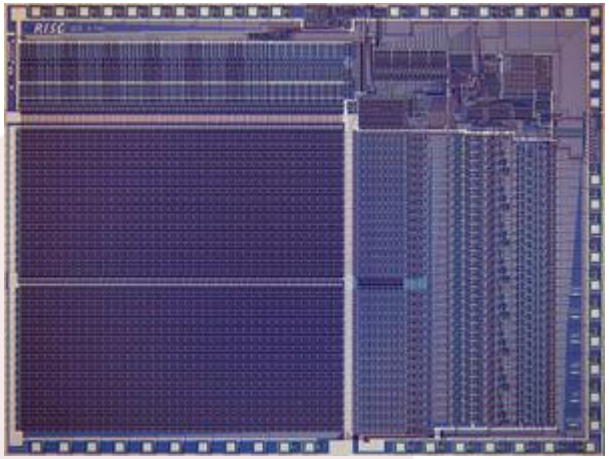
Filosofia RISC (Reduced Instruction Set Computer)

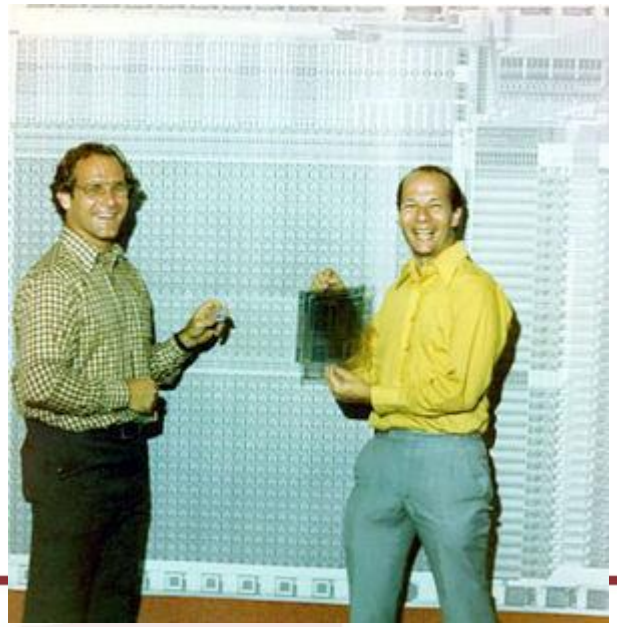
- Tornar as máquinas mais simples e portanto mais velozes:
  - poucas instruções, simples e com formato único
  - modos simples de endereçamento
  - implementação mais eficiente

Filosofia	**CISC**			**RISC**		
Máquinas	IBM370	VAX11	Xerox	IBM801	RISC I	RISC II
Ano	1973	1978	1978	1980	1981	1981
# Instr.	208	303	270	120	31	55
Tan. Instr.	2-6	2-57	1-3	4	4	4

Filosofia RISC (Reduced Instruction Set Computer)

- Tornar as máquinas mais simples e portanto mais velozes:
  - poucas instruções, simples e com formato único
  - modos simples de endereçamento
  - implementação mais eficiente

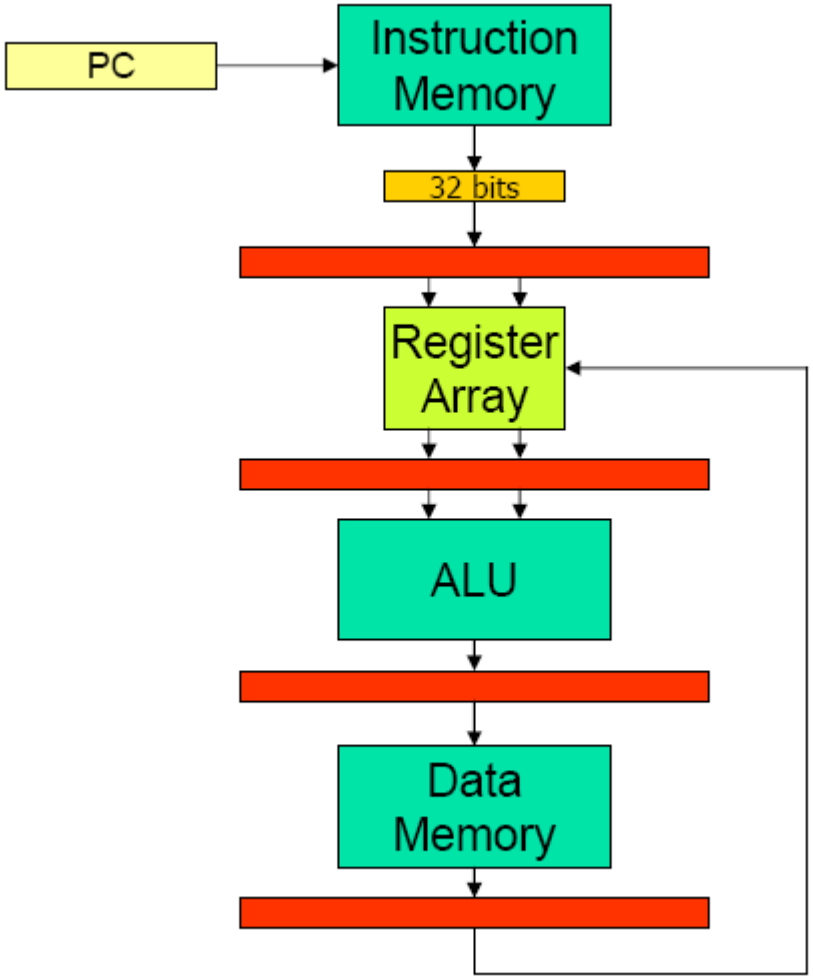


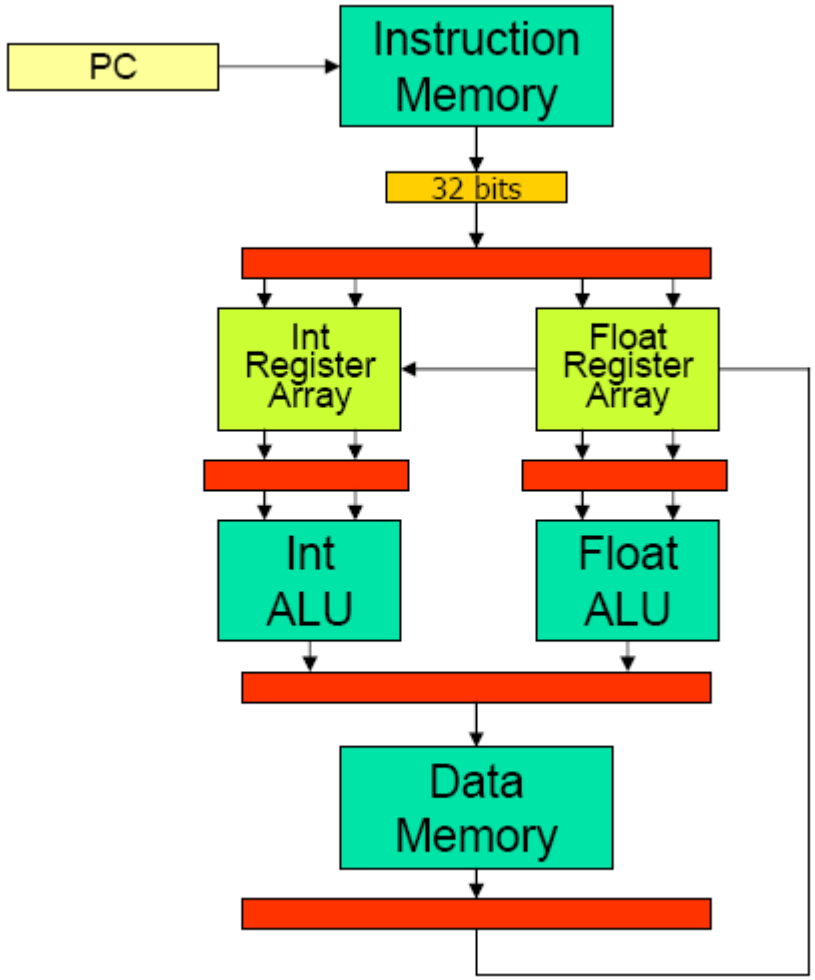


Características de algumas máquinas

	CISC			RISC	
Características	IBM 370/168	VAX 11/780	INTEL 80486	Motorola 88000	MIPS R4000
ano	73	78	89	88	91
número de instruções	208	303	235	51	94
tamanho das instruções	2-6	2-57	1-11	4	4
modos de endereçamento	4	22	11	3	1
número de registradores	16	16	8	32	32
memória de controle -Kbits	420	480	246	-	-
tamanho da cache-Kbytes	64	64	8	16	128

Pipeline


PC
Instruction Memory
32 bits
Register Array
ALU
Data Memory

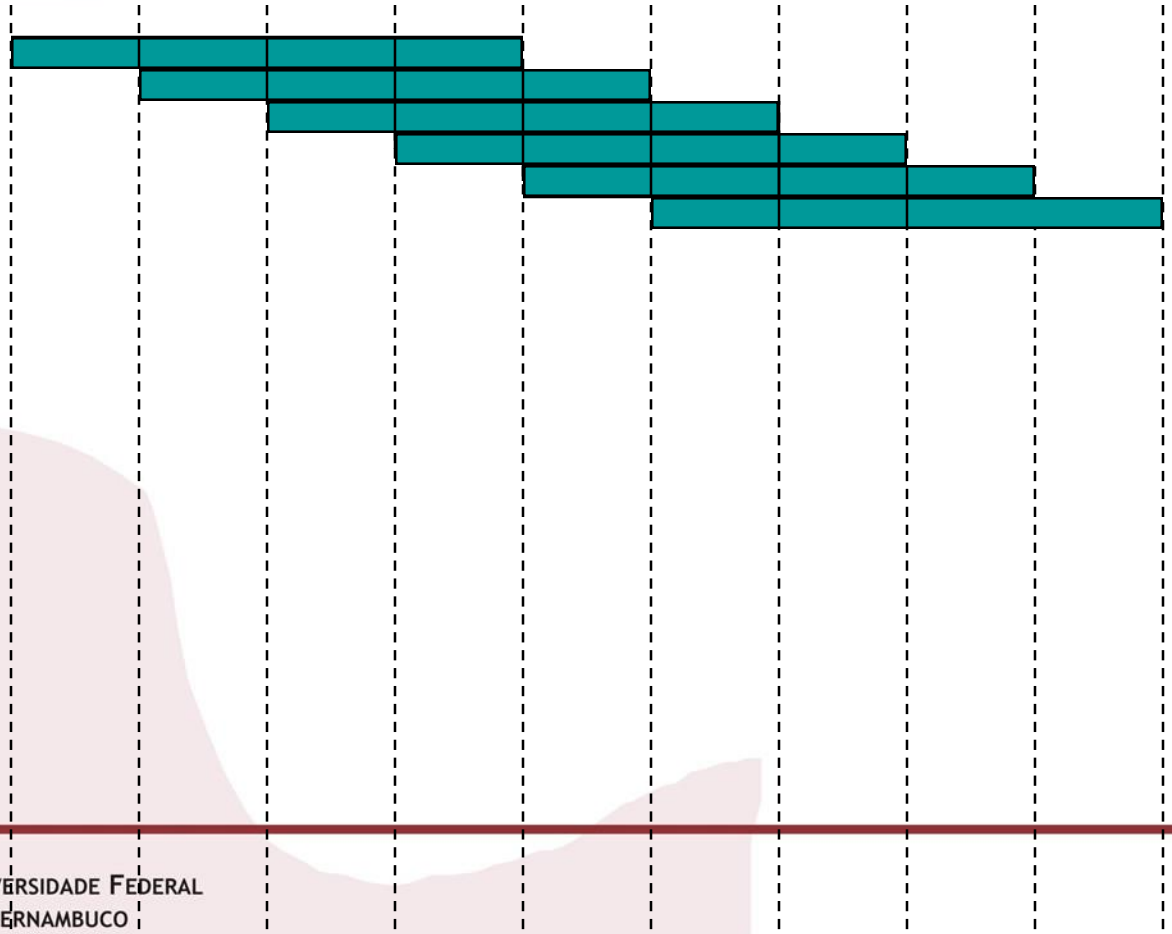

PC
Instruction Memory
32 bits
Int Register Array
Float Register Array
Int ALU
Float ALU
Data Memory

Pipeline

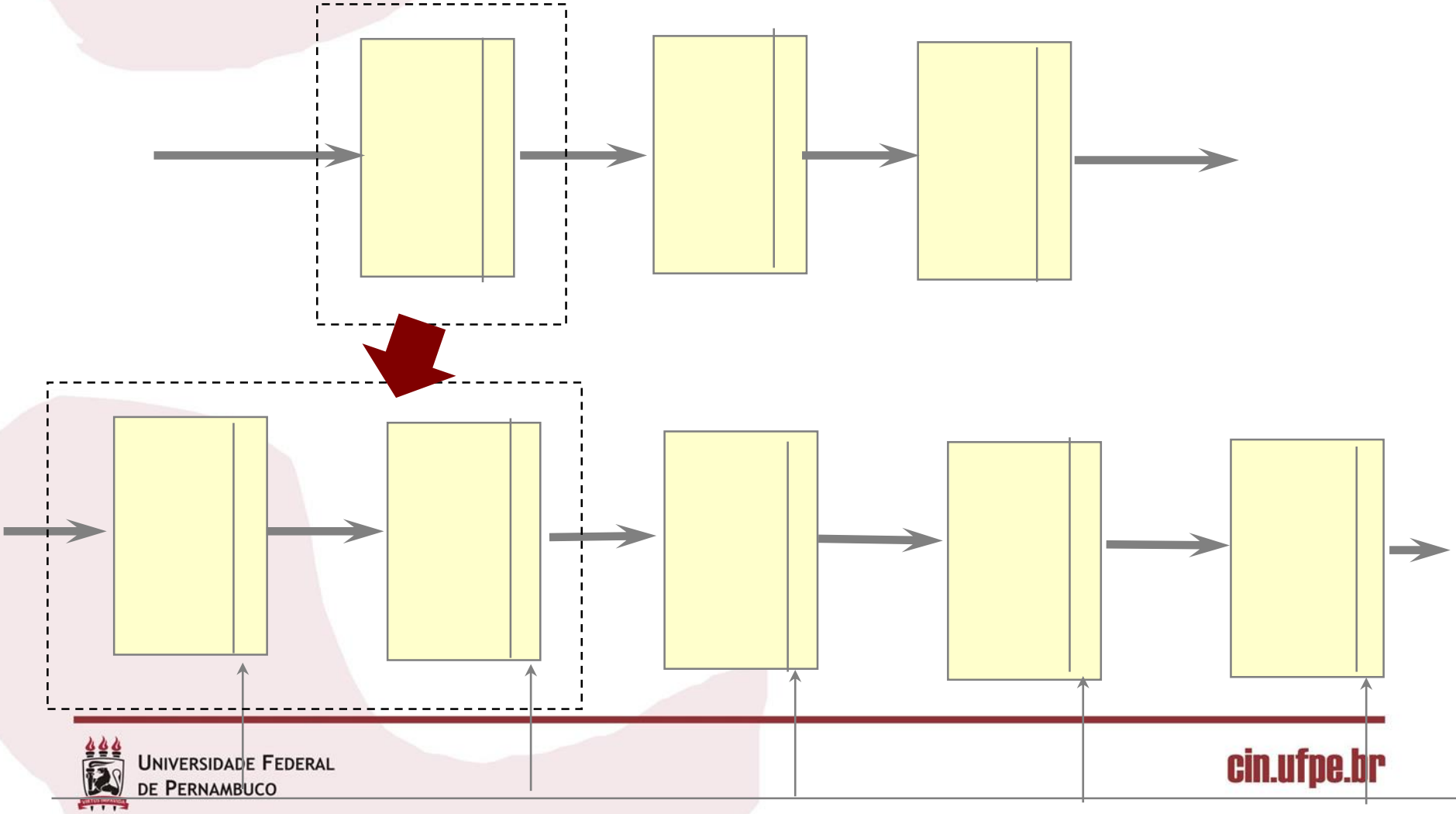
- Problema:
  - Como conseguir estágios com a mesma duração?
  - aumentar a frequência do clock
    - Superpipeline
- Como melhorar o desempenho do pipeline de processadores RISC e CISC?
  - executar instruções em paralelo
    - Superescalar

Melhorando o Desempenho

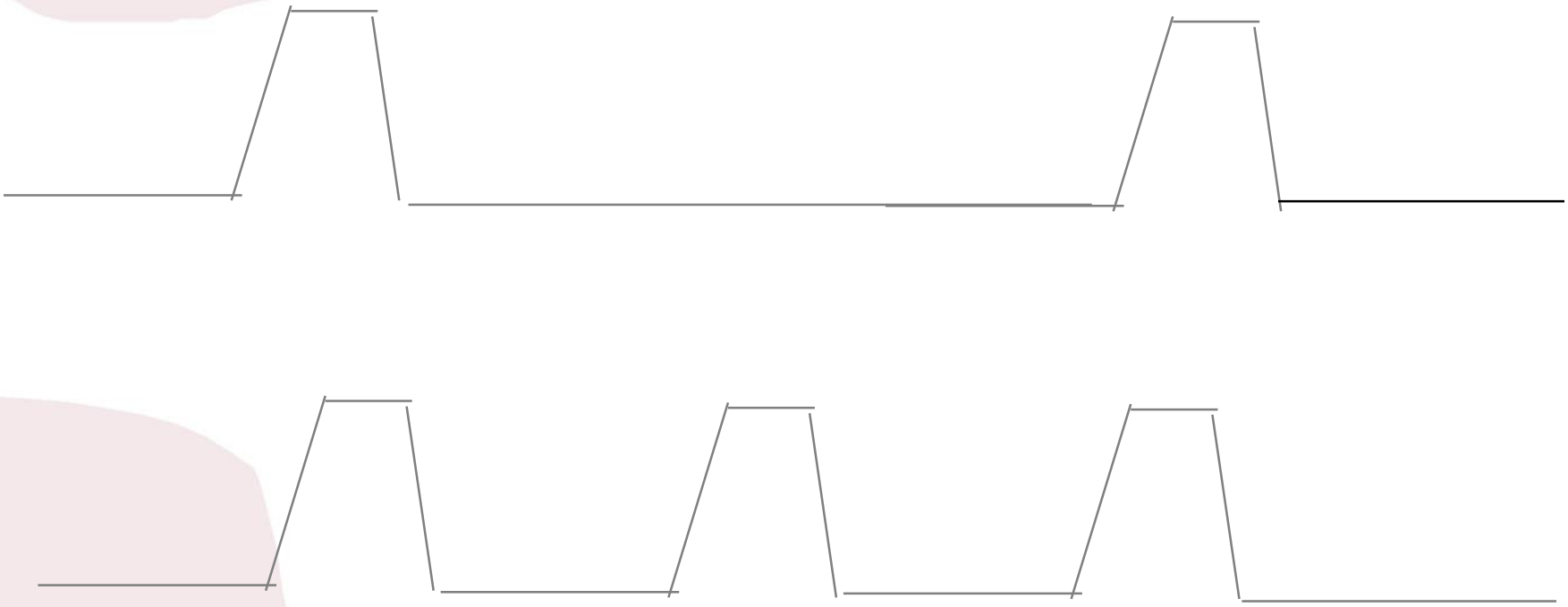
pipeline



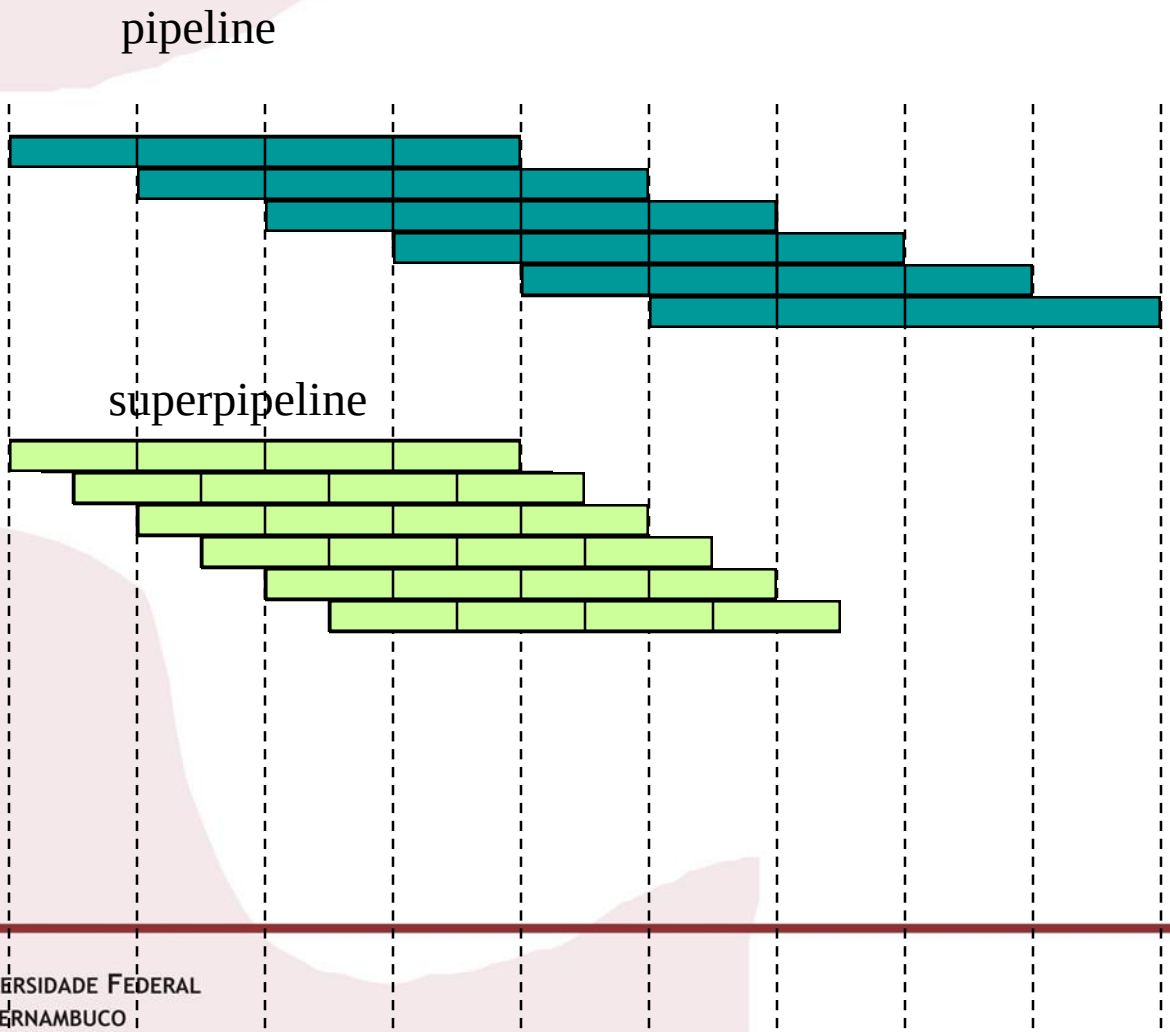
Superpipeline



Relógio



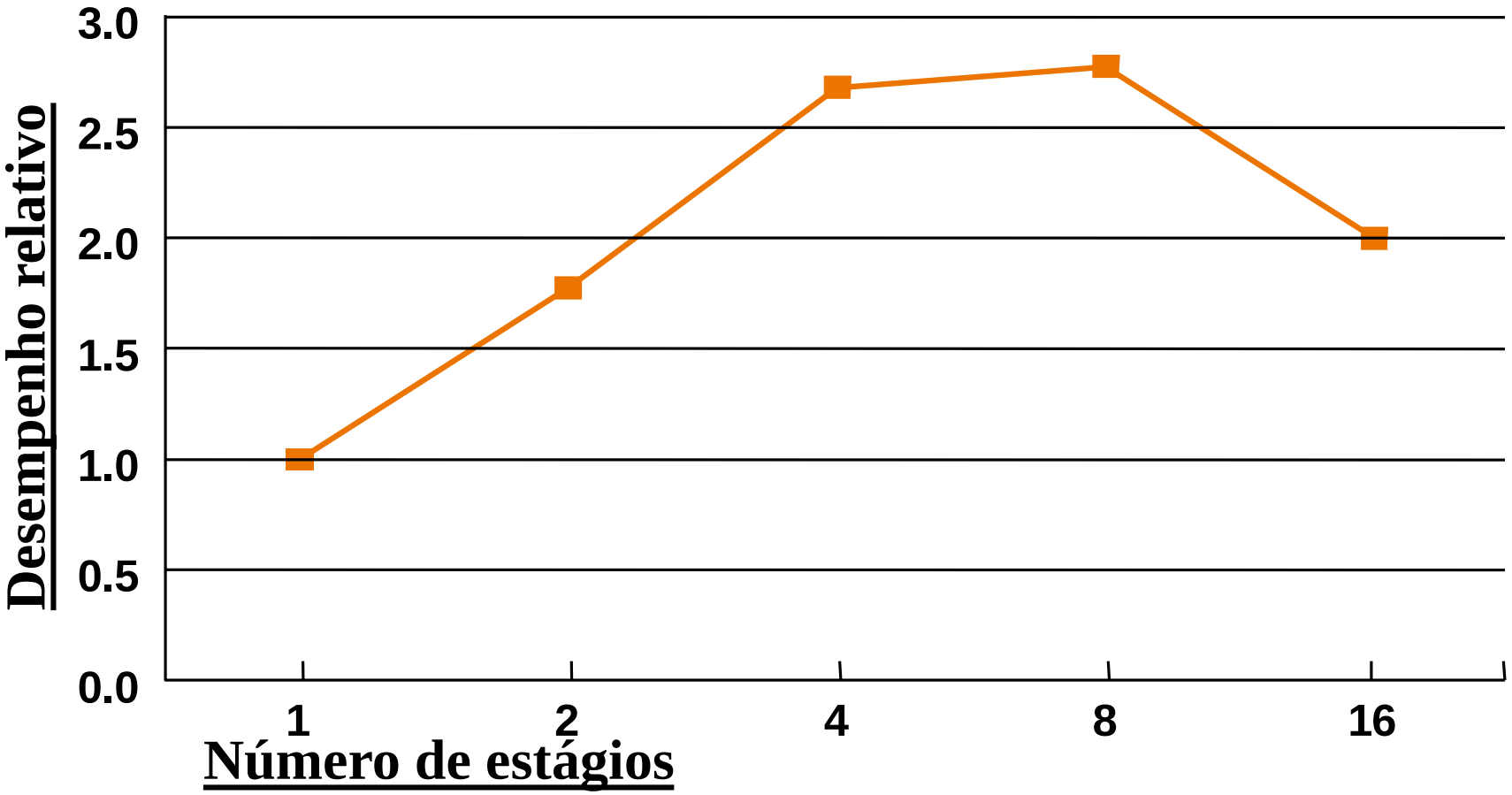
Melhorando o desempenho



Melhorando o desempenho Superpipeline

Centro de Informática U·F·P·E

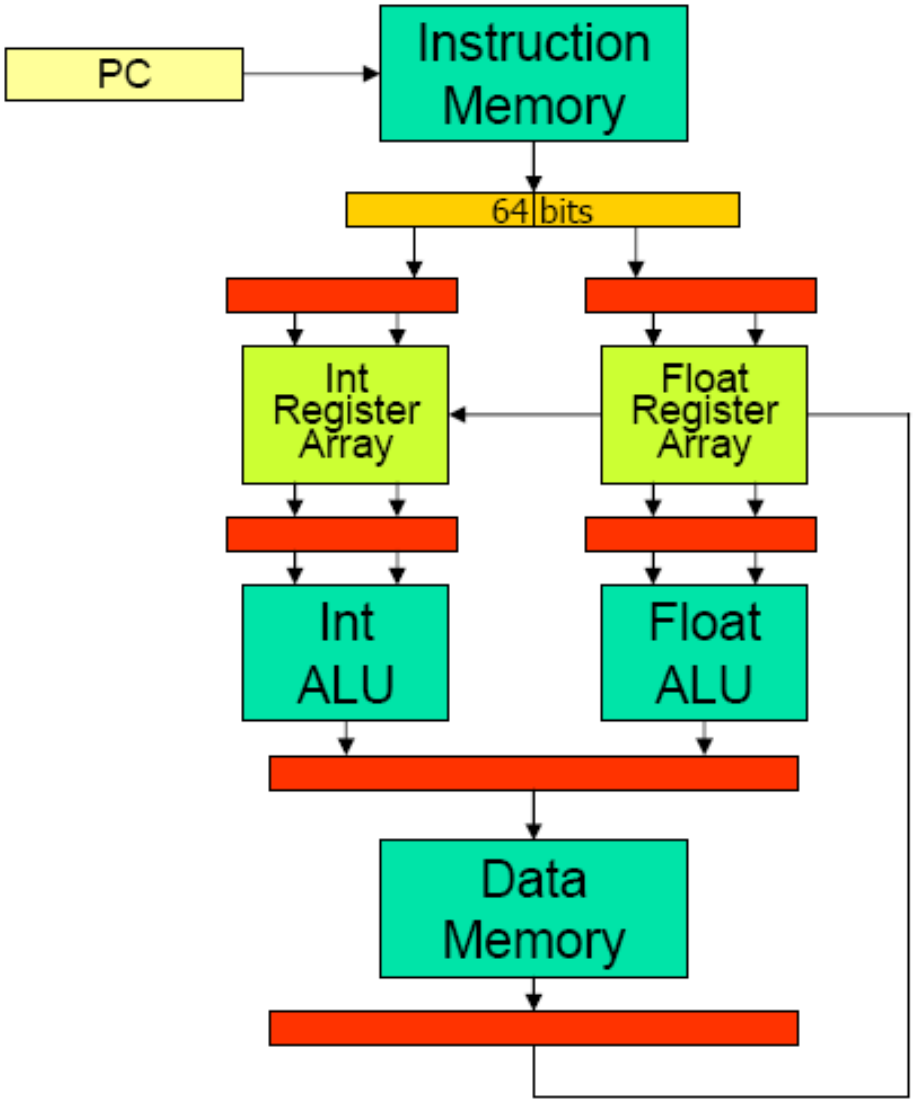
Número de Estágios X Aumento de desempenho



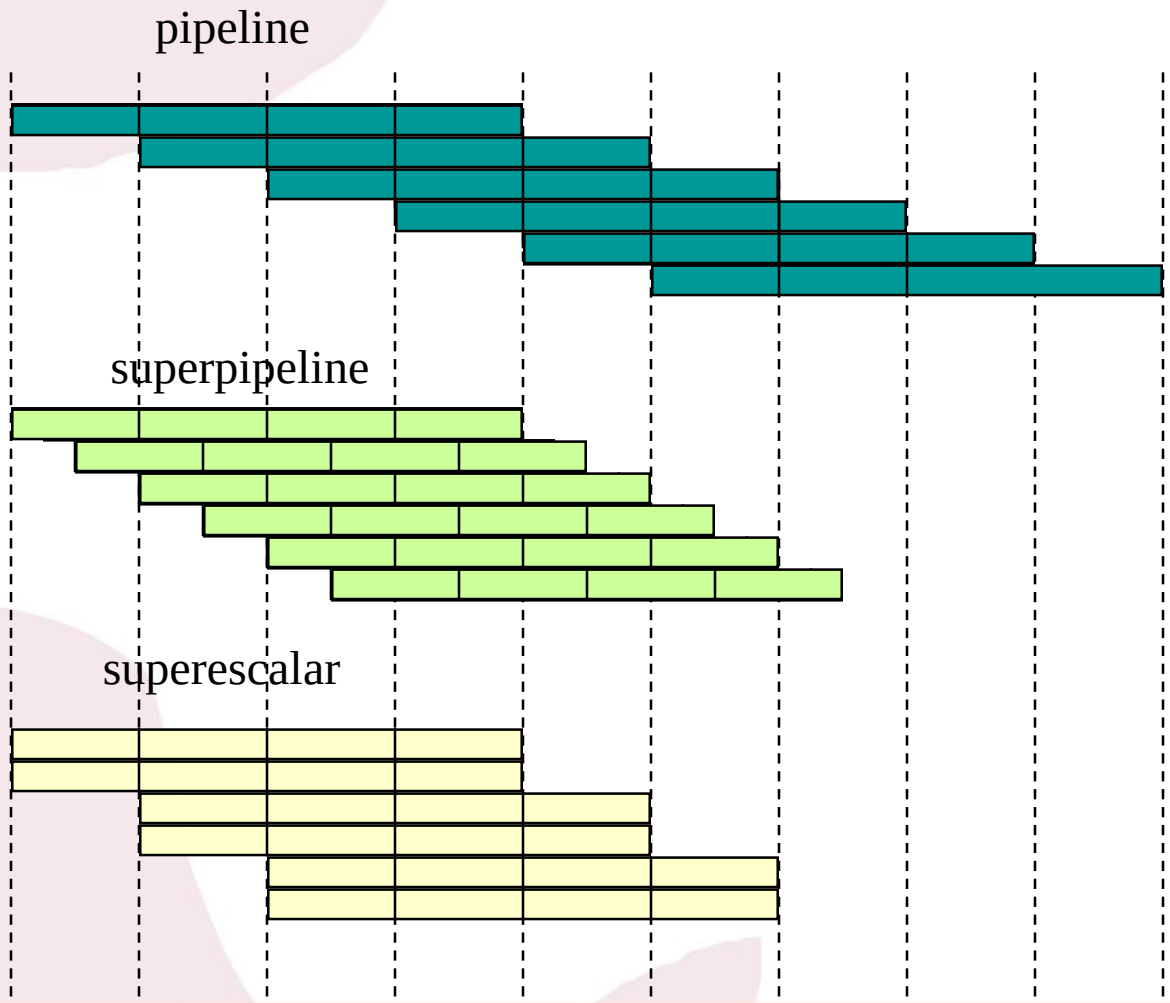
Processador superescalar

- Execução simultânea de instruções:
  - aritméticas, load, stores e desvios condicionais
- Aplicável a arquiteturas RISC e CISC
  - RISC:
    - implementação mais fácil
  - CISC:
    - implementação mais difícil

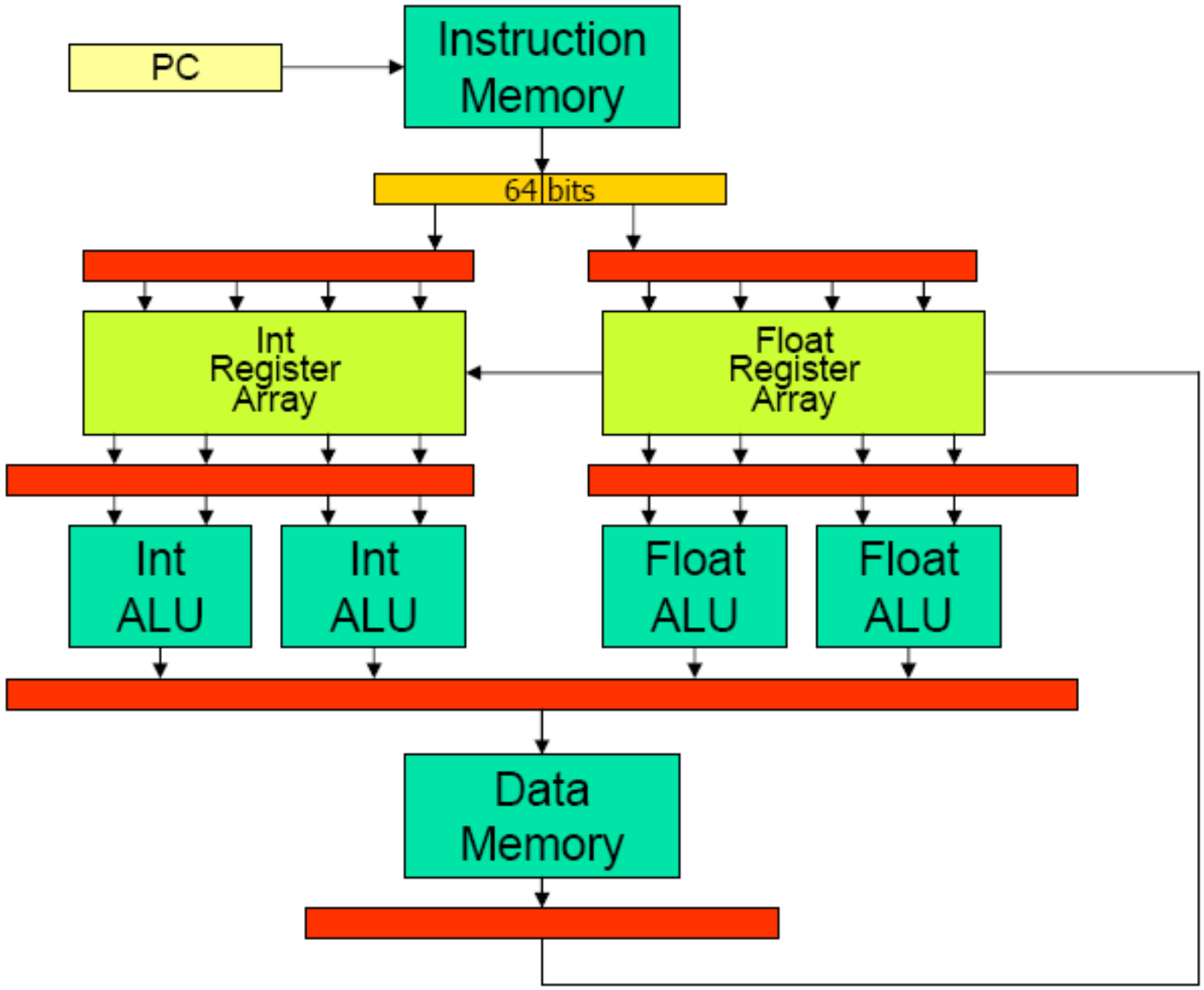
Processador Superescalar



Melhorando o desempenho



Processador Superescalar



Paralelismo no Nível de Instrução (ILP)

- Pipeline: execução de múltiplas instruções em paralelo
- Para aumentar ILP
  - Superpipeline (pipeline com mais estágios)
    - Menos trabalho por estágio $\Rightarrow$ menor período do clock
  - Despacho múltiplo– Execução de várias instruções por vez
    - Replicar estágios do pipeline $\Rightarrow$ multiplos pipelines
    - Inicia múltiplas instruções por ciclo de clock
    - CPI < 1, ou Instructions Per Cycle (IPC) >1
    - Ex., 4GHz 4-way multiple-issue
      - 16 BIPS, pico CPI = 0.25, pico IPC = 4
    - Mas, dependências reduzem o desempenho….

Despacho Múltiplo– Paralelismo ILP

- Despacho múltiplo – estático
  - Compilador agrupa instruções a serem executadas em conjunto
  - Pacotes são enviados para “issue slots”
  - Compilador detecta e resolve conflitos
- Despacho múltiplo – Dinâmico
  - CPU examina sequencia de instruções e escolhe instruções a serem executadas por ciclo
  - Compilador pode ajudar reordenando instruções
  - CPU resolve conflitos usando técnicas avançadas em tempo de execução

Especulação

- Faz previsão de desvio para definer fluxo das instruções que serão executadas
  - Começa a execução o mais rápido possível
  - Verifica se a previsão foi certa
    - Se sim, completa a instrução
    - Se não, volta para buscar as instruções corretas

Especulação – Software/Hardware

- Compilador reordena as instruções
  - Pode incluir instruções para adiar o desvio ou para correção de previsões erradas
- Hardware busca instruções seguintes a serem executadas
  - Armazena resultados em buffers até que eles “realmente” sejam necessários
  - Limpa buffers quando a especulação é incorreta

Especulações e Exceções

- E se uma exceção ocorre devido a uma instrução especulada?
  - e.g., load especulado antes de atualização de ponteiro nulo
  - Deve armazenar exceções (sem resolver) até que a execução da instrução seja terminada.

Despacho múltiplo – estático

- Compilador agrupa instruções em “issue packets”
  - Grupo de instruções que podem ser executadas em um ciclo
  - Determinadas pelos recursos do pipeline
- Um “issue packet” pode ser visto como uma instrução longa
  - Especifica instruções múltiplas concorrentes

  $\Rightarrow$ Very Long Instruction Word (VLIW)

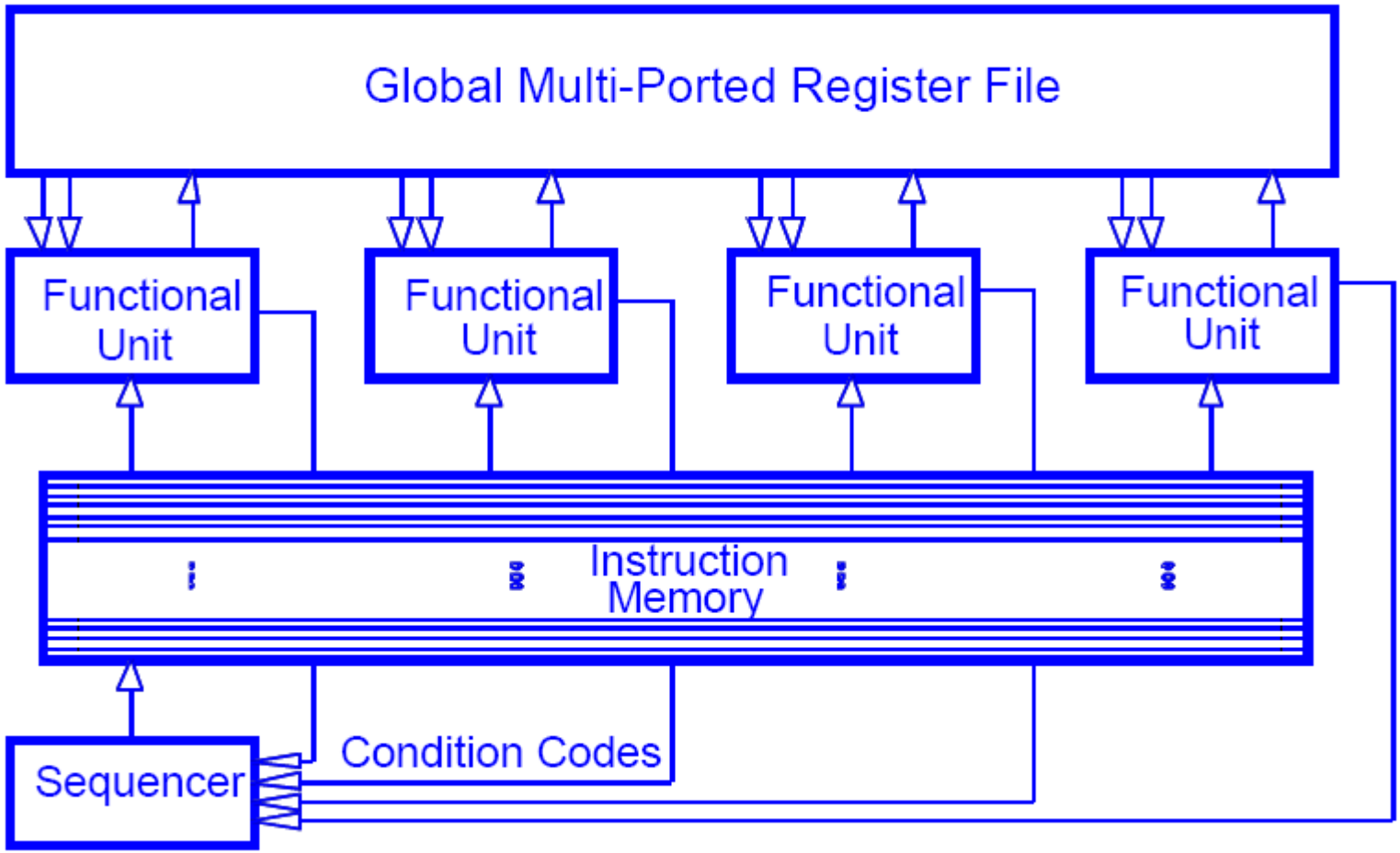
VLIW

Centro de Informática U·F·P·E

very long instruction word

O compilador descobre as instruções que podem ser executadas em paralelo e agrupa-as
formando uma longa instrução
que será despachada para a máquina

VLIW


Global Multi-Ported Register File
Functional Unit
Functional Unit
Functional Unit
Functional Unit
Instruction Memory
Condition Codes
Sequencer

VLIW

Simplifica o hardware transferindo para o compilador a lógica de detecção do paralelismo

circuitos mais simples
Clock com maior frequência

Escalonamento com Despacho múltiplo – estático

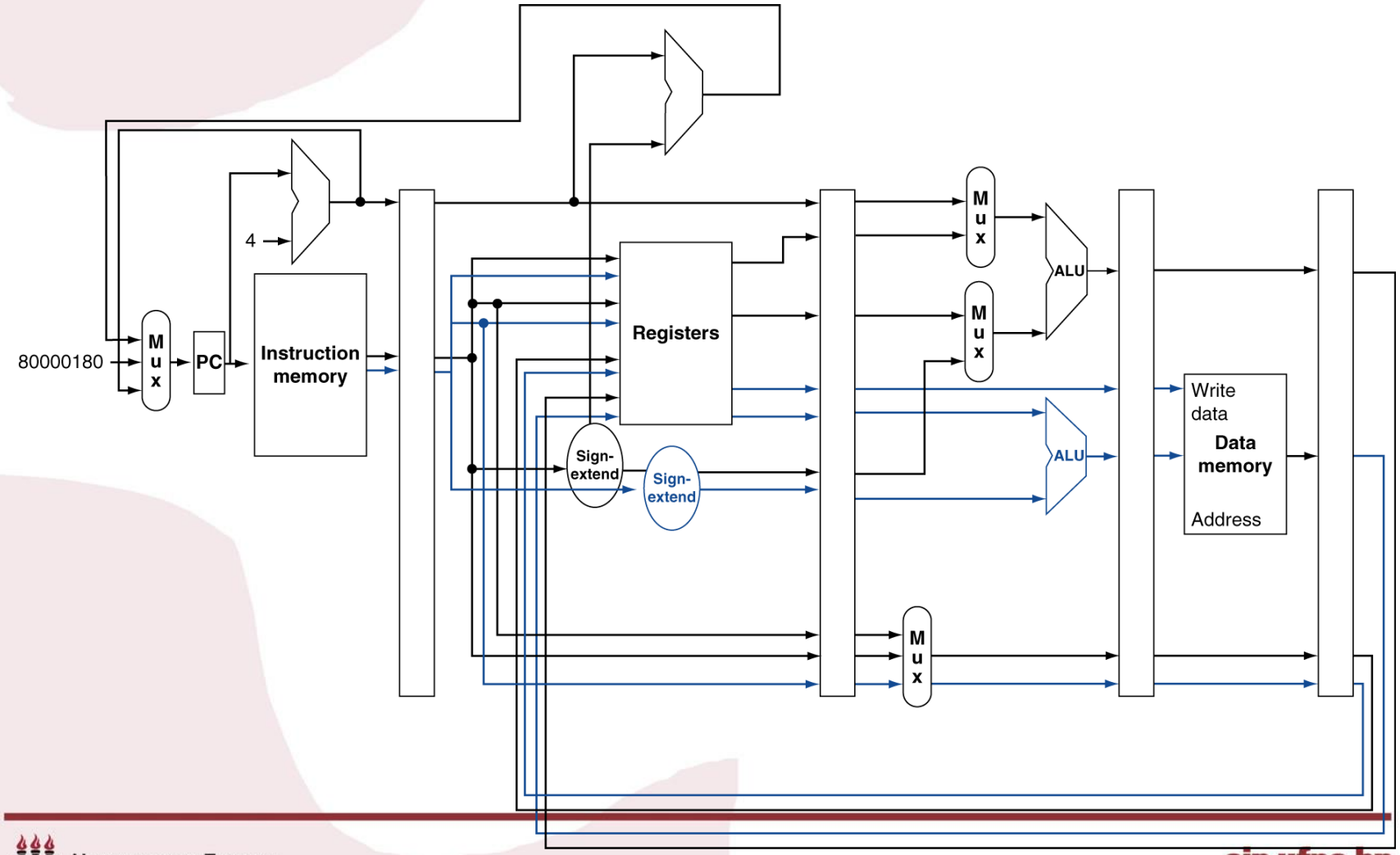
- Compilador deve remover todos conflitos
  - Reordenar instruções nos “issue packets”
  - Nenhuma dependência no packet
  - Preencher com “nops” se necessário

MIPS com Despacho múltiplo – estático

- Pacotes com duas instruções
  - Uma instrução ALU/branch
  - Uma instrução load/store
  - Pacote de 64-bits
    - ALU/branch ou load/store
    - NOP se não tiver combinação

Address	Instruction type	Pipeline Stages						
n	ALU/branch	IF	ID	EX	MEM	WB		
n + 4	Load/store	IF	ID	EX	MEM	WB		
n + 8	ALU/branch		IF	ID	EX	MEM	WB	
n + 12	Load/store		IF	ID	EX	MEM	WB	
n + 16	ALU/branch			IF	ID	EX	MEM	WB
n + 20	Load/store			IF	ID	EX	MEM	WB

MIPS com Despacho múltiplo – estático



Conflitos no MIPS com Despacho múltiplo estático

- Mais instruções executando em paralelo
- Conflito de dado
  - Forwarding evita conflitos no caso de “single-issue”
  - Não pode usar resultado da ALU em load/store no mesmo pacote
    - ```
      add  $t0, $s0, $s1
      load $s2, 0($t0)
      ```
    - Divide em dois pacotes: um retardo
- Conflito: Load-uso
  - Um ciclo de retardo mas com duas instruções
- Escalonamento mais agressivo é necessário
      ```

Exemplo de Escalonamento

- Escalonamento para dual-issue MIPS

```
Loop: lw   $t0, 0($s1)       # $t0=array element
      addu $t0, $t0, $s2     # add scalar in $s2
      sw   $t0, 0($s1)       # store result
      addi $s1, $s1,-4       # decrement pointer
      bne  $s1, $zero, Loop  # branch $s1!=0
```

	ALU/branch	Load/store	cycle
Loop:	nop	lw $t0, 0($s1)	1
	addi $s1, $s1,-4	nop	2
	addu $t0, $t0, $s2	nop	3
	bne $s1, $zero, Loop	sw $t0, 4($s1)	4

- IPC = 5/4 = 1.25 (c.f. peak IPC = 2)

Loop Unrolling

- Replica corpo do loop body para extrair mais paralelismo
  - Reduce overhead para controle do loop
- Usa registradores diferentes na replicação
  - Denominado “register renaming”
  - Evita “anti-dependencies” dependentes do loop
    - Store seguido por um load do mesmo registrador
    - Dependência de nome
      - Reuso do nome do registrador

Loop Unrolling Exemplo

```
lw   $t0, 0($s1)
addu $t0, $t0, $s2
sw   $t0, 0($s1)
lw   $t1, 4($s1)
addu $t1, $t1, $s2
sw   $t1, 4($s1)
lw   $t2, 8($s1)
addu $t2, $t2, $s2
sw   $t2, 8($s1)
lw   $t3,12($s1)
addu $t3, $t4, $s2
sw   $t3, 12($s1)
addi $s1, $s1,-16
bne  $s1, $zero, Loop
```

```
lw   $t0, 0($s1)
addi $s1, $s1,-16
lw   $t1, 12($s1)
lw   $t2, 8($s1)
lw   $t3, 4($s1)
addu $t0, $t0, $s2
addu $t1, $t1, $s2
addu $t2, $t2, $s2
addu $t3, $t4, $s2
sw   $t0, 16($s1)
sw   $t1, 12($s1)
sw   $t2, 8($s1)
sw   $t3, 4($s1)
bne  $s1, $zero, Loop
```

Desenrolar loop

Rearrumar o código

Loop Unrolling Exemplo

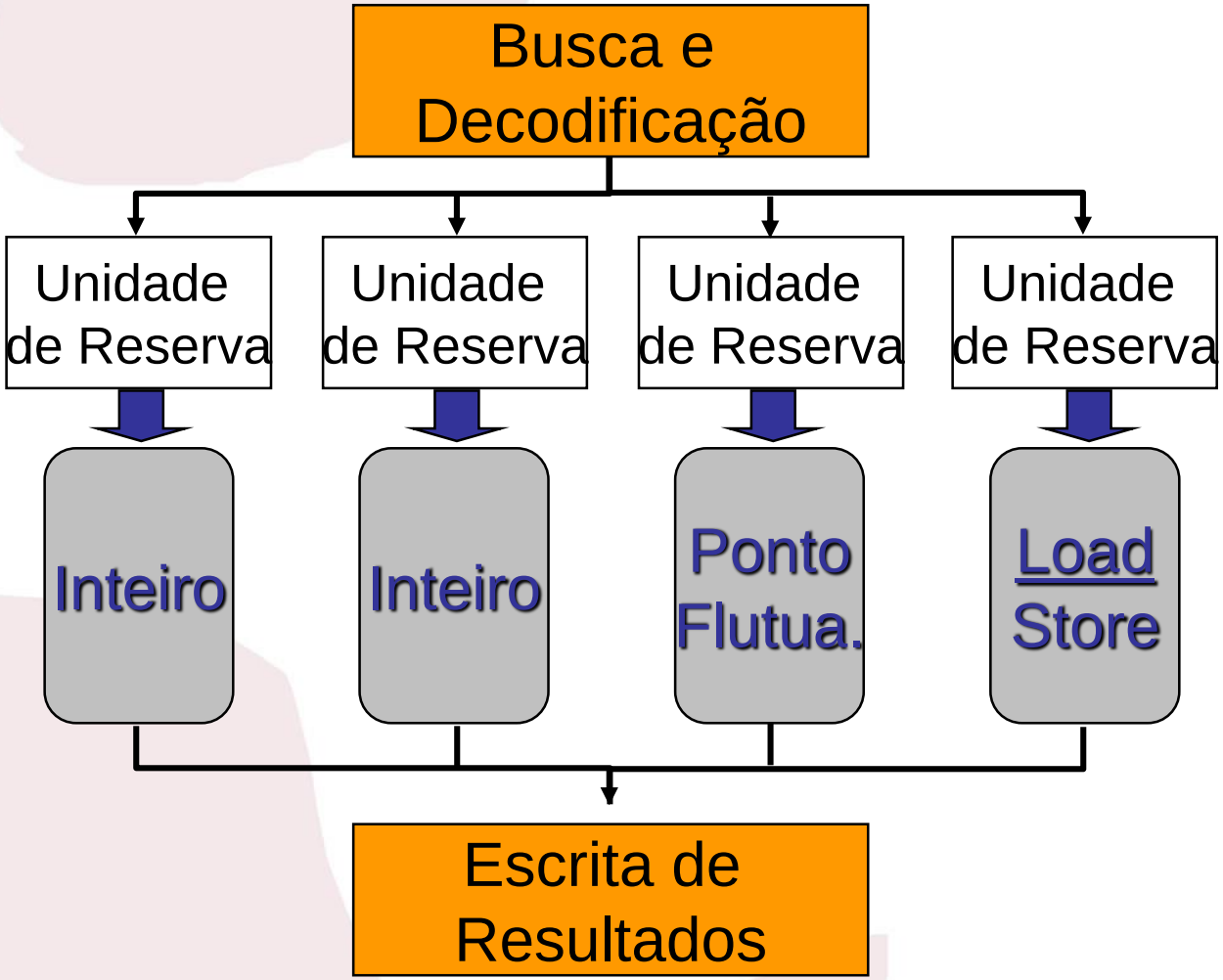
	ALU/branch	Load/store	cycle
Loop:	addi $s1, $s1,-16	lw $t0, 0($s1)	1
	nop	lw $t1, 12($s1)	2
	addu $t0, $t0, $s2	lw $t2, 8($s1)	3
	addu $t1, $t1, $s2	lw $t3, 4($s1)	4
	addu $t2, $t2, $s2	sw $t0, 16($s1)	5
	addu $t3, $t4, $s2	sw $t1, 12($s1)	6
	nop	sw $t2, 8($s1)	7
	bne $s1, $zero, Loop	sw $t3, 4($s1)	8

- IPC = 14/8 = 1.75
  - Perto de 2, mas com aumento de registradores e de código

Despacho múltiplo – Dinâmico

- Processadores “Superscalares”
- CPU decide se executa 0, 1, 2, … instruções a cada ciclo
  - Evita conflito estrutural e de dado
- Evita a necessidade de escalonamento pelo compilador
  - Apesar de otimizações serem bem-vindas

Processador Superescalar



Despacho:
Em ordem

Execução:
Fora de ordem

Escrita:
1) Em ordem
2) Fora de ordem

Superescalar:

Revisando Dependências de Dados

- Dependência Verdadeira:
  Read-after-Write (RAW)

- Dependência de Saída:
  Write-after-Write (WAW)

- Antidependência:
  Write-after-Read (WAR)

Superescalar:

Dependências WAR e WAW

Processadores com um pipeline não apresentam estas dependências porque apenas um estágio do pipeline altera o estado da máquina (os registradores) na ordem em que as instruções são iniciadas.

Superescalar:

Dependências de Dados

$r3 := r0\ op_1\ r5$ (I1)
$r4 := r3\ op_2\ r8$ (I2)
$r3 := r5\ op_3\ r9$ (I3)
$r7 := r3\ op_4\ r4$ (I4)

- Dependência Verdadeira: (RAW)
  I2 e I1, I4 e I3, I4 e I2
- Antidependência: (WAR)
  I3 não pode terminar antes de I2 iniciar
- Dependências de Saída: (WAW)
  I3 não pode terminar antes de I1

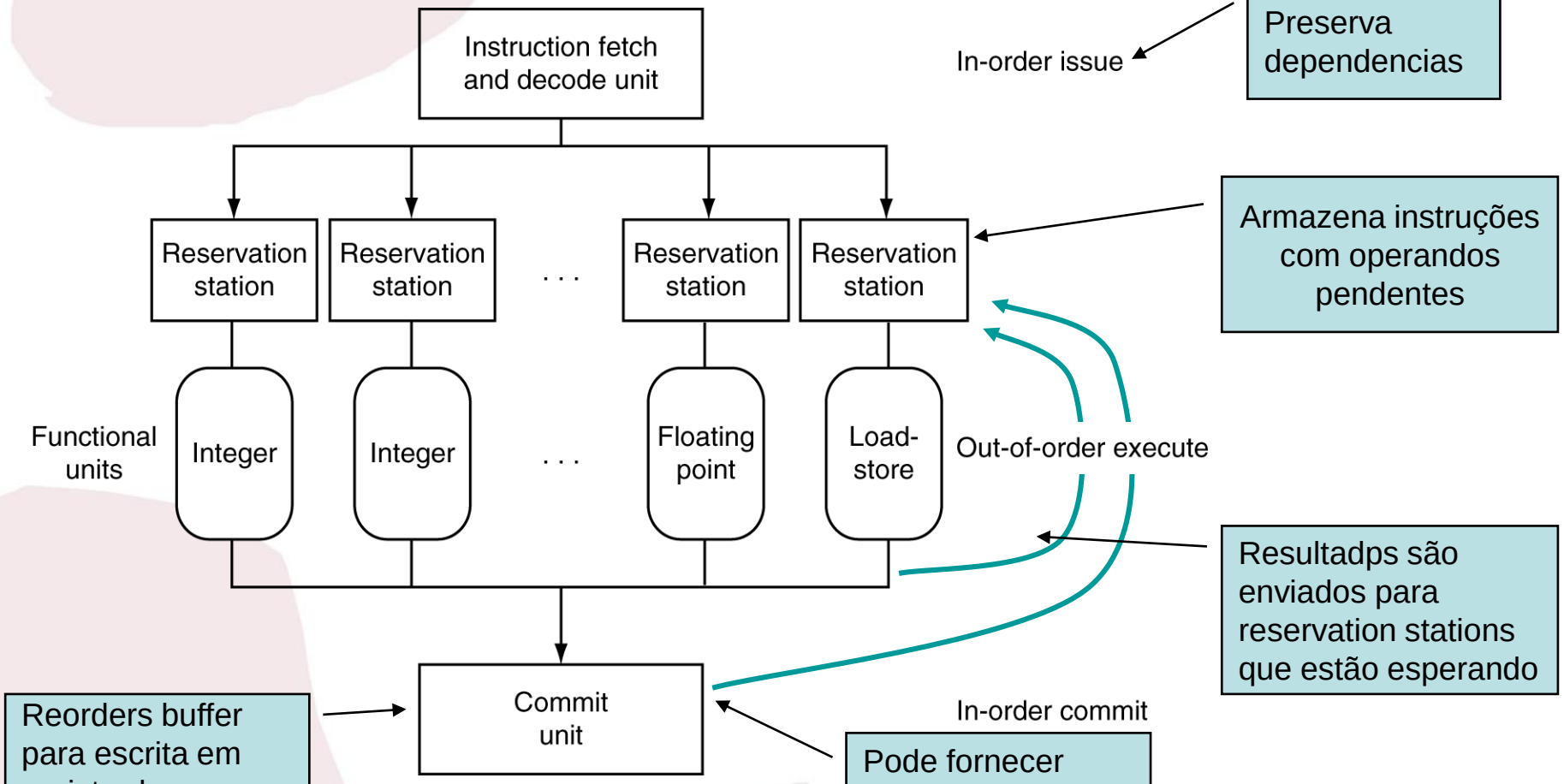
Dependências de Dados WAR e WAW: Como tratá-las

1) Inserir NOPs ou bolhas (igual RAW)

2) Inserir instruções independentes (igual RAW)

3) Usar armazenamento temporário

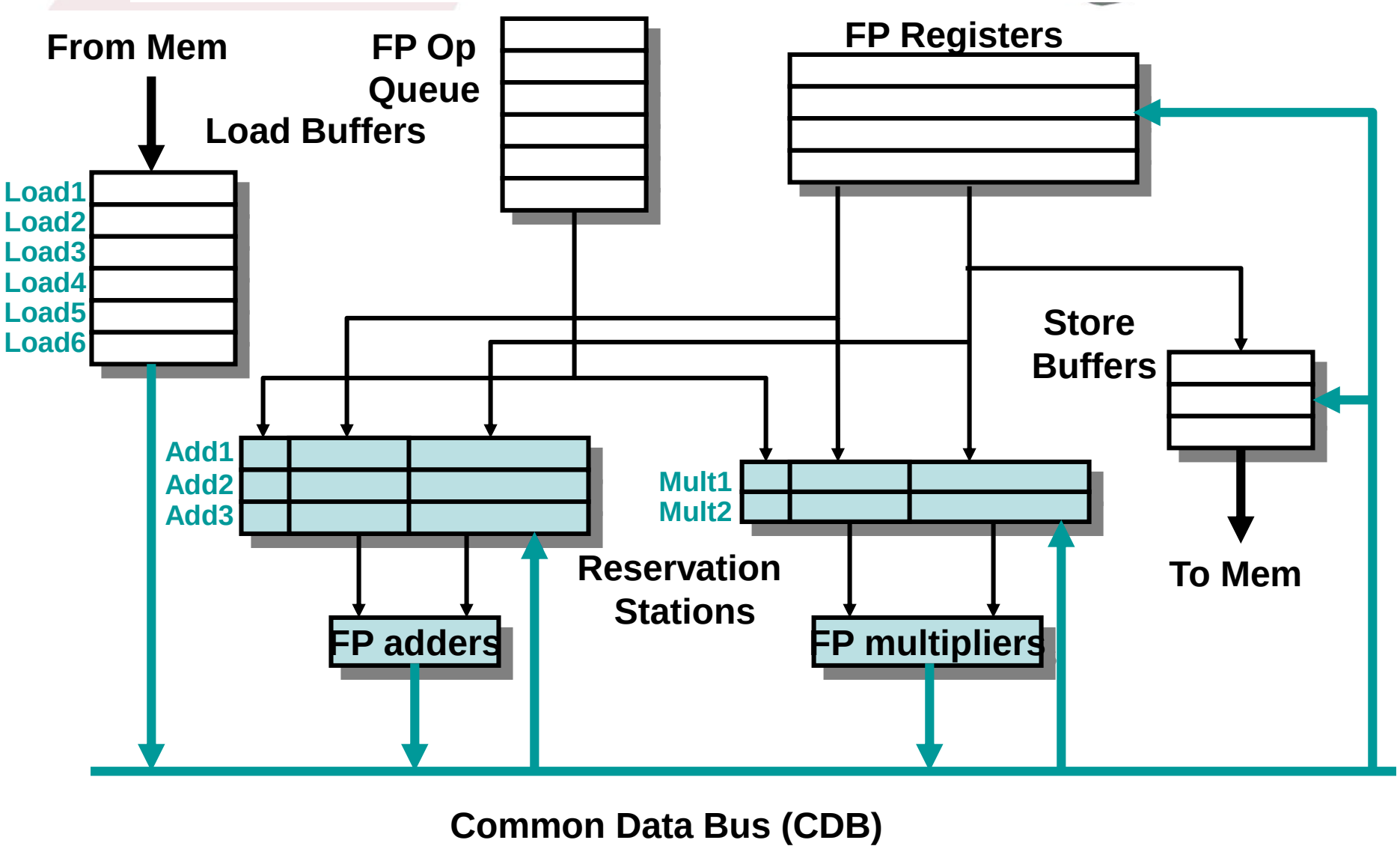
Pipeline com Escalonamento Dinâmico



Escalonamento Dinâmico – Despacho simples

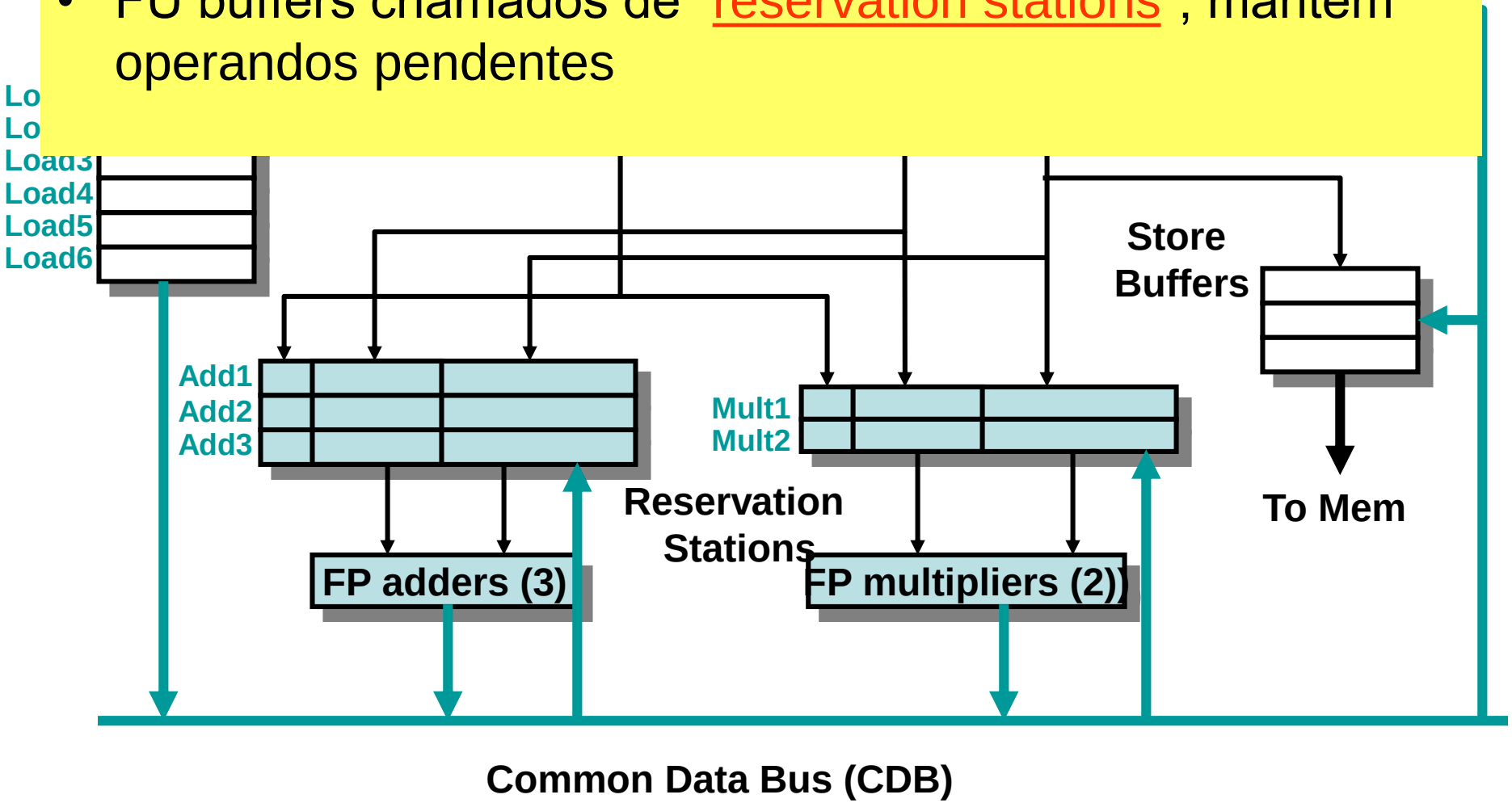
- Baseado no Técnica proposta por Roberto Tomasulo
  - IBM 360/91
  - não havia caches; tempo de acesso à memória grande
  - instruções de FP com grandes latências (delay))
- Como garantir alto desempenho sem compilador especial
  - Um pequeno número de registradores floating point (4 no 360) dificultava escalonamento das operações pelo compilador.
- Tomasulo: Como ter efetivamente mais registradores ? Como resolver os conflitos devido à dependência de dados
  - seguir quando os operandos estiverem prontos e renomeamento implementado no hardware!
- Usado nos processadores:
  - Alpha 21264, HP 8000, MIPS 10000, Pentium III, PowerPC 604, …

Estrutura Básica para Escalonamento Dinâmico



Estrutura Básica para Escalonamento Dinâmico

- Controle & buffers distribuídos na Function Units (FU)
- FU buffers chamados de "reservation stations"; mantém operandos pendentes



Reservation Station

Load/Store
Imediato/Endereço Efetivo:
Mantém informação sobre o end. de memória calculado para instruções de load ou store

0/1:Busy: Indica que a Reservation Station e sua FU estão ocupadas

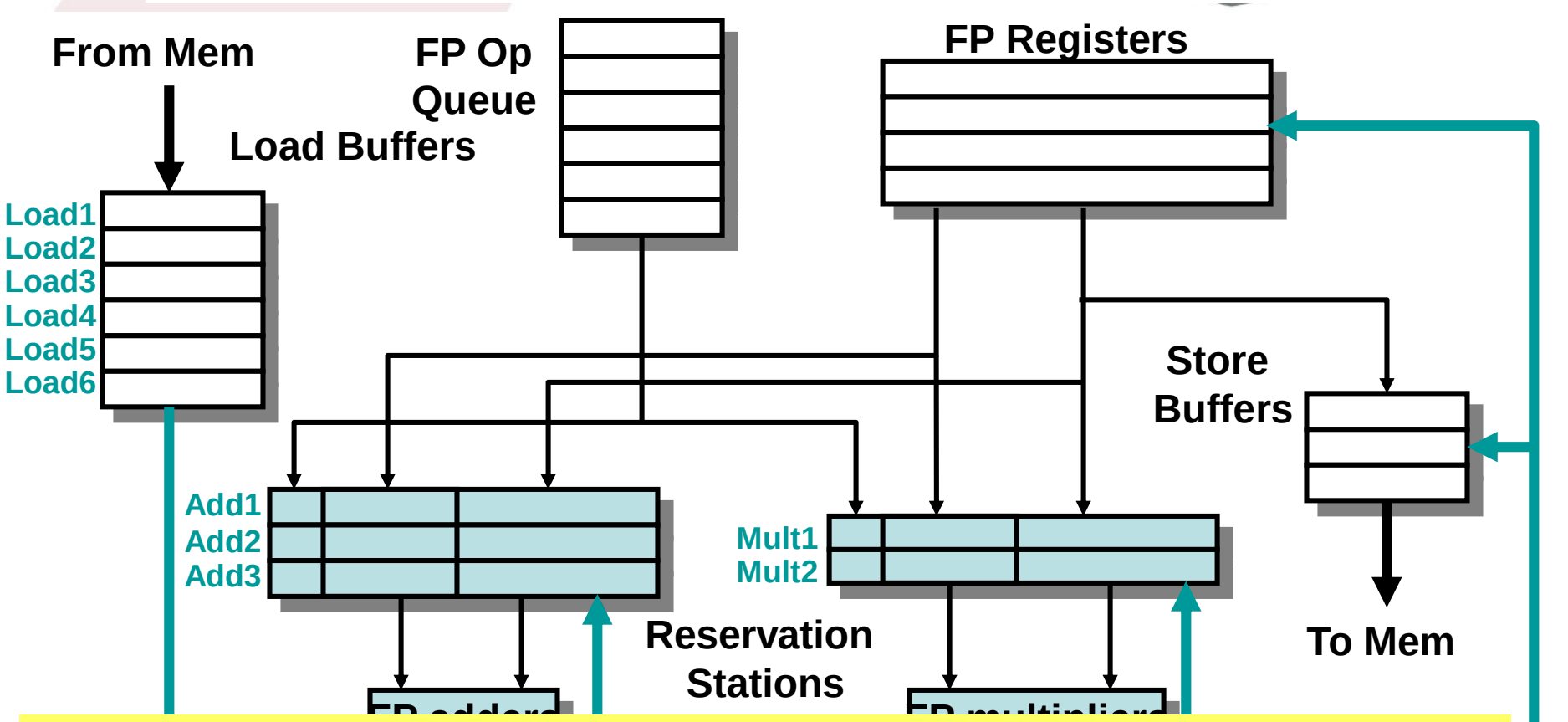
Operação: Operação a ser executada na unidade

número das Reservation Stations que produzirão os operandos correspondentes (valores a serem escritos)
- **Qj,Qk = 0 => ready**
- **Store buffers tem somente Qi para RS que produz resultado**

Vj, Vk: Valores dos operantos Fontes
Store buffers tem campos V, resultados devem ser armazenados

OBS.: Register File
Qi = N0. RS

Estrutura Básica para Escalonamento Dinâmico



Resultados enviados da Reservation Station para a FU, (sem usar os registradores) através de broadcast dos resultados para todas as FUs usando o Common Data Bus

Escalonamento Dinâmico

1. Issue— pega a instrução na “FP Op Queue” e decodifica. Se a reservation station da instrução está livre (não há conflito estrutural), despacha a instrução e envia operandos disponíveis.

2. Execute —executa a operação sobre os operandos (EX) se os dois operandos estão disponíveis. Se algum operando não estiver pronto, monitora o Common Data Bus (espera pelo cálculo do operando, essa espera resolve RAW). Quando um operando está pronto atualiza reservation table)

Escalonamento Dinâmico

3. Write result — termina a execução (WB)

Os resultados calculados pelas reservation units são enviados por Broadcast via Common Data Bus para todas unidades; marca a reservation station como disponível. Escreve no registrador.

Exemplo do Alg. Tomasulo

- Trecho de programa a ser executado:

```
1    L.D      F6,34(R2)
2    L.D      F2,45(R3)
3    MUL.D    F0,F2,F4
4    SUB.D    F8,F2,F6
5    DIV.D    F10,F0,F6
6    ADD.D    F6,F8,F2
```

RAW?: (1-4); (1-5); (2-3); (2-4); (2-6); ....

Exemplo do Alg. Tomasulo

- Assumir as seguintes latências:
  - Load: 1 ciclo
  - Add: 2 ciclos
  - Multiplicação: 10 ciclos
  - Divisão: 40 ciclos

- Load-Store:
  - Calcula o endereço efetivo (FU)
  - Instruções de Load ou Store usam buffers
  - Acesso à memória (somente load)
  - Escrita de resultado
    - Load: envia o valor para o registador e/ou reservation stations
    - Store: escreve o valor na memória
    - (escritas somente no estágio “WB” – simplifica o algoritmo de Tomasulo)

Exemplo do Alg. Tomasulo

Instruções do programa

3 estágios da execução

Instruction status:

Instruction		j	k	*Issue*	*Exec Comp*	*Write Result*
LD	F6	34+	R2			
LD	F2	45+	R3			
MULTD	F0	F2	F4			
SUBD	F8	F6	F2			
DIVD	F10	F0	F6			
ADDD	F6	F8	F2			

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

3 Load/Buffers

Reservation Stations:

Time	*Name*	*Busy*	*Op*	*S1 Vj*	*S2 Vk*	*RS Qj*	*RS Qk*
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	No					
	Mult2	No					

Latência (que falta) da FU

3 FP Adder R.S.
2 FP Mult R.S.

Register result status:

Clock		*F0*	*F2*	*F4*	*F6*	*F8*	*F10*	*F12*	...	*F30*
0	*FU*									

Clock cycle

Exemplo Tomasulo: Ciclo 1

Instruction status:

Instruction		j	k	Issue	Exec Comp	Write Result
LD	F6	34+	R2	1		
LD	F2	45+	R3			
MULTD	F0	F2	F4			
SUBD	F8	F6	F2			
DIVD	F10	F0	F6			
ADDD	F6	F8	F2			

	Busy	Address
Load1	Yes	34+R2
Load2	No	
Load3	No	

Reservation Stations:

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	No					
	Mult2	No					

Register result status:

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
1	FU				Load1					

Exemplo Tomasulo: Ciclo 2

Instruction status:

Instruction		j	k	Issue	Exec Comp	Write Result
LD	F6	34+	R2	1		
LD	F2	45+	R3	2		
MULTD	F0	F2	F4			
SUBD	F8	F6	F2			
DIVD	F10	F0	F6			
ADDD	F6	F8	F2			

	Busy	Address
Load1	Yes	34+R2
Load2	Yes	45+R3
Load3	No	

Reservation Stations:

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	No					
	Mult2	No					

Register result status:

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
2	FU		Load2		Load1					

Nota: pode haver múltiplos loads pendentes

Exemplo Tomasulo: Ciclo 3

Instruction status:

Instruction		j	k	Issue	Exec Comp	Write Result
LD	F6	34+	R2	1	3	
LD	F2	45+	R3	2		
MULTD	F0	F2	F4	3		
SUBD	F8	F6	F2			
DIVD	F10	F0	F6			
ADDD	F6	F8	F2			

	Busy	Address
Load1	Yes	34+R2
Load2	Yes	45+R3
Load3	No	

Reservation Stations:

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	Yes	MULTD		R(F4)	Load2	
	Mult2	No					

Register result status:

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
3	FU	Mult1	Load2		Load1					

- **Nota: nomes dos registradores são removidos (“renamed”) na Reservation Stations; MULT issued**
- **Load1 completa; alguém esperando por Load1?**

Exemplo Tomasulo: Ciclo 4

Instruction status:

Instruction		j	k	Issue	Exec Comp	Write Result
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	
MULTD	F0	F2	F4	3		
SUBD	F8	F6	F2	4		
DIVD	F10	F0	F6			
ADDD	F6	F8	F2			

	Busy	Address
Load1	No	
Load2	Yes	45+R3
Load3	No	

Reservation Stations:

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
	Add1	Yes	SUBD	M(A1)			Load2
	Add2	No					
	Add3	No					
	Mult1	Yes	MULTD		R(F4)	Load2	
	Mult2	No					

Register result status:

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
4	FU	Mult1	Load2		M(A1)	Add1				

- **Load2 completa; alguém esperando por Load2?**

Exemplo Tomasulo: Ciclo 5

Instruction status:

Instruction		j	k	Issue	Exec Comp	Write Result
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3		
SUBD	F8	F6	F2	4		
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2			

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reservation Stations:

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
2	Add1	Yes	SUBD	M(A1)	M(A2)		
	Add2	No					
	Add3	No					
10	Mult1	Yes	MULTD	M(A2)	R(F4)		
	Mult2	Yes	DIVD		M(A1)	Mult1	

Register result status:

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
5	FU	Mult1	M(A2)		M(A1)	Add1	Mult2			

- **Timer inicia a contagem regressiva para Add1, Mult1**

Exemplo Tomasulo: Ciclo 6

Instruction status:

Instruction		j	k	Issue	Exec Comp	Write Result
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3		
SUBD	F8	F6	F2	4		
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2	6		

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reservation Stations:

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
1	Add1	Yes	SUBD	M(A1)	M(A2)		
	Add2	Yes	ADDD		M(A2)	Add1	
	Add3	No					
9	Mult1	Yes	MULTD	M(A2)	R(F4)		
	Mult2	Yes	DIVD		M(A1)	Mult1	

Register result status:

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
6	FU	Mult1	M(A2)		Add2	Add1	Mult2			

- **Despacha ADDD, dependência de nome em F6?**

Exemplo Tomasulo: Ciclo 7

Instruction status:

Instruction		j	k	Issue	Exec Comp	Write Result
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3		
SUBD	F8	F6	F2	4	7	
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2	6		

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reservation Stations:

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
0	Add1	Yes	SUBD	M(A1)	M(A2)		
	Add2	Yes	ADDD		M(A2)	Add1	
	Add3	No					
8	Mult1	Yes	MULTD	M(A2)	R(F4)		
	Mult2	Yes	DIVD		M(A1)	Mult1	

Register result status:

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
7	FU	Mult1	M(A2)		Add2	Add1	Mult2			

- **Add1 (SUBD) completa; alguém esperando por add1?**

Exemplo Tomasulo: Ciclo 8

Instruction status:

Instruction		j	k	Issue	Exec Comp	Write Result
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3		
SUBD	F8	F6	F2	4	7	8
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2	6		

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reservation Stations:

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
	Add1	No					
2	Add2	Yes	ADDD	(M-M)	M(A2)		
	Add3	No					
7	Mult1	Yes	MULTD	M(A2)	R(F4)		
	Mult2	Yes	DIVD		M(A1)	Mult1	

Register result status:

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
8	*FU*	Mult1	M(A2)		Add2	(M-M)	Mult2			

Exemplo Tomasulo: Ciclo 9

Instruction status:

Instruction		j	k	*Issue*	*Exec Comp*	*Write Result*
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3		
SUBD	F8	F6	F2	4	7	8
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2	6		

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reservation Stations:

Time	*Name*	*Busy*	*Op*	*S1 Vj*	*S2 Vk*	*RS Qj*	*RS Qk*
	Add1	No					
1	Add2	Yes	ADDD	(M-M)	M(A2)		
	Add3	No					
6	Mult1	Yes	MULTD	M(A2)	R(F4)		
	Mult2	Yes	DIVD		M(A1)	Mult1	

Register result status:

Clock		*F0*	*F2*	*F4*	*F6*	*F8*	*F10*	*F12*	...	*F30*
9	*FU*	Mult1	M(A2)		Add2	(M-M)	Mult2			

Exemplo Tomasulo: Ciclo 10

Instruction status:

Instruction		j	k	Issue	Exec Comp	Write Result
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3		
SUBD	F8	F6	F2	4	7	8
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2	6	10	

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reservation Stations:

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
	Add1	No					
0	Add2	Yes	ADDD	(M-M)	M(A2)		
	Add3	No					
5	Mult1	Yes	MULTD	M(A2)	R(F4)		
	Mult2	Yes	DIVD		M(A1)	Mult1	

Register result status:

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
10	*FU*	Mult1	M(A2)		Add2	(M-M)	Mult2			

- **Add2 (ADDD) completa; alguém esperando por add2?**

Exemplo Tomasulo: Ciclo 11

Instruction status:

Instruction		j	k	Issue	Exec Comp	Write Result
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3		
SUBD	F8	F6	F2	4	7	8
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2	6	10	11

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reservation Stations:

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
	Add1	No					
	Add2	No					
	Add3	No					
4	Mult1	Yes	MULTD	M(A2)	R(F4)		
	Mult2	Yes	DIVD		M(A1)	Mult1	

Register result status:

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
11	FU	Mult1	M(A2)		(M-M+M	(M-M)	Mult2			

- **Resultado de ADDD é escrito!**
- **Todas as instruções mais rápidas terminam neste ciclo!**

Exemplo Tomasulo: Ciclo 12

Centro de Informática U·F·P·E

Instruction status:

Instruction		j	k	Issue	Exec Comp	Write Result
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3		
SUBD	F8	F6	F2	4	7	8
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2	6	10	11

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reservation Stations:

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
	Add1	No					
	Add2	No					
	Add3	No					
3	Mult1	Yes	MULTD	M(A2)	R(F4)		
	Mult2	Yes	DIVD		M(A1)	Mult1	

Register result status:

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
12	FU	Mult1	M(A2)		(M-M+M	(M-M)	Mult2			

Exemplo Tomasulo: Ciclo 13

Centro de Informática U·F·P·E

Instruction status:

Instruction		j	k	Issue	Exec Comp	Write Result
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3		
SUBD	F8	F6	F2	4	7	8
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2	6	10	11

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reservation Stations:

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
	Add1	No					
	Add2	No					
	Add3	No					
2	Mult1	Yes	MULTD	M(A2)	R(F4)		
	Mult2	Yes	DIVD		M(A1)	Mult1	

Register result status:

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
13	FU	Mult1	M(A2)		(M-M+M	(M-M)	Mult2			

Exemplo Tomasulo: Ciclo 14

Instruction status:

Instruction		j	k	Issue	Exec Comp	Write Result
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3		
SUBD	F8	F6	F2	4	7	8
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2	6	10	11

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reservation Stations:

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
	Add1	No					
	Add2	No					
	Add3	No					
1	Mult1	Yes	MULTD	M(A2)	R(F4)		
	Mult2	Yes	DIVD		M(A1)	Mult1	

Register result status:

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
14	FU	Mult1	M(A2)		(M-M+M	(M-M)	Mult2			

Exemplo Tomasulo: Ciclo 15

Instruction status:

Instruction		j	k	Issue	Exec Comp	Write Result
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3	15	
SUBD	F8	F6	F2	4	7	8
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2	6	10	11

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reservation Stations:

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
	Add1	No					
	Add2	No					
	Add3	No					
0	Mult1	Yes	MULTD	M(A2)	R(F4)		
	Mult2	Yes	DIVD		M(A1)	Mult1	

Register result status:

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
15	FU	Mult1	M(A2)		(M-M+M	(M-M)	Mult2			

- **Mult1 (MULTD) completa; alguém esperando por mult1?**

Exemplo Tomasulo: Ciclo 16

Instruction status:

Instruction		j	k	Issue	Exec Comp	Write Result
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3	15	16
SUBD	F8	F6	F2	4	7	8
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2	6	10	11

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reservation Stations:

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	No					
40	Mult2	Yes	DIVD	M*F4	M(A1)		

Register result status:

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
16	FU	M*F4	M(A2)		(M-M+M	(M-M)	Mult2			

- **Agora é só esperar que Mult2 (DIVD) complete**

Pulando alguns ciclos

Exemplo Tomasulo: Ciclo 55

Instruction status:

Instruction		j	k	Issue	Exec Comp	Write Result
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3	15	16
SUBD	F8	F6	F2	4	7	8
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2	6	10	11

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reservation Stations:

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	No					
1	Mult2	Yes	DIVD	M*F4	M(A1)		

Register result status:

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
55	FU	M*F4	M(A2)		(M-M+M	(M-M)	Mult2			

Exemplo Tomasulo: Ciclo 56

Instruction status:

Instruction		j	k	Issue	Exec Comp	Write Result
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3	15	16
SUBD	F8	F6	F2	4	7	8
DIVD	F10	F0	F6	5	56	
ADDD	F6	F8	F2	6	10	11

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reservation Stations:

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	No					
0	Mult2	Yes	DIVD	M*F4	M(A1)		

Register result status:

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
56	FU	M*F4	M(A2)		(M-M+M	(M-M)	Mult2			

- **Mult2 (DIVD) completa; alguém esperando por mult2?**

Exemplo Tomasulo: Ciclo 57

Instruction status:

Instruction		j	k	Issue	Exec Comp	Write Result
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3	15	16
SUBD	F8	F6	F2	4	7	8
DIVD	F10	F0	F6	5	56	57
ADDD	F6	F8	F2	6	10	11

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reservation Stations:

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	No					
	Mult2	Yes	DIVD	M*F4	M(A1)		

Register result status:

Clock **57**

	F0	F2	F4	F6	F8	F10	F12	...	F30
FU	M*F4	M(A2)		(M-M+M)	(M-M)	Result			

- **Despacho en ordem, execução fora de ordem, término fora de ordem**

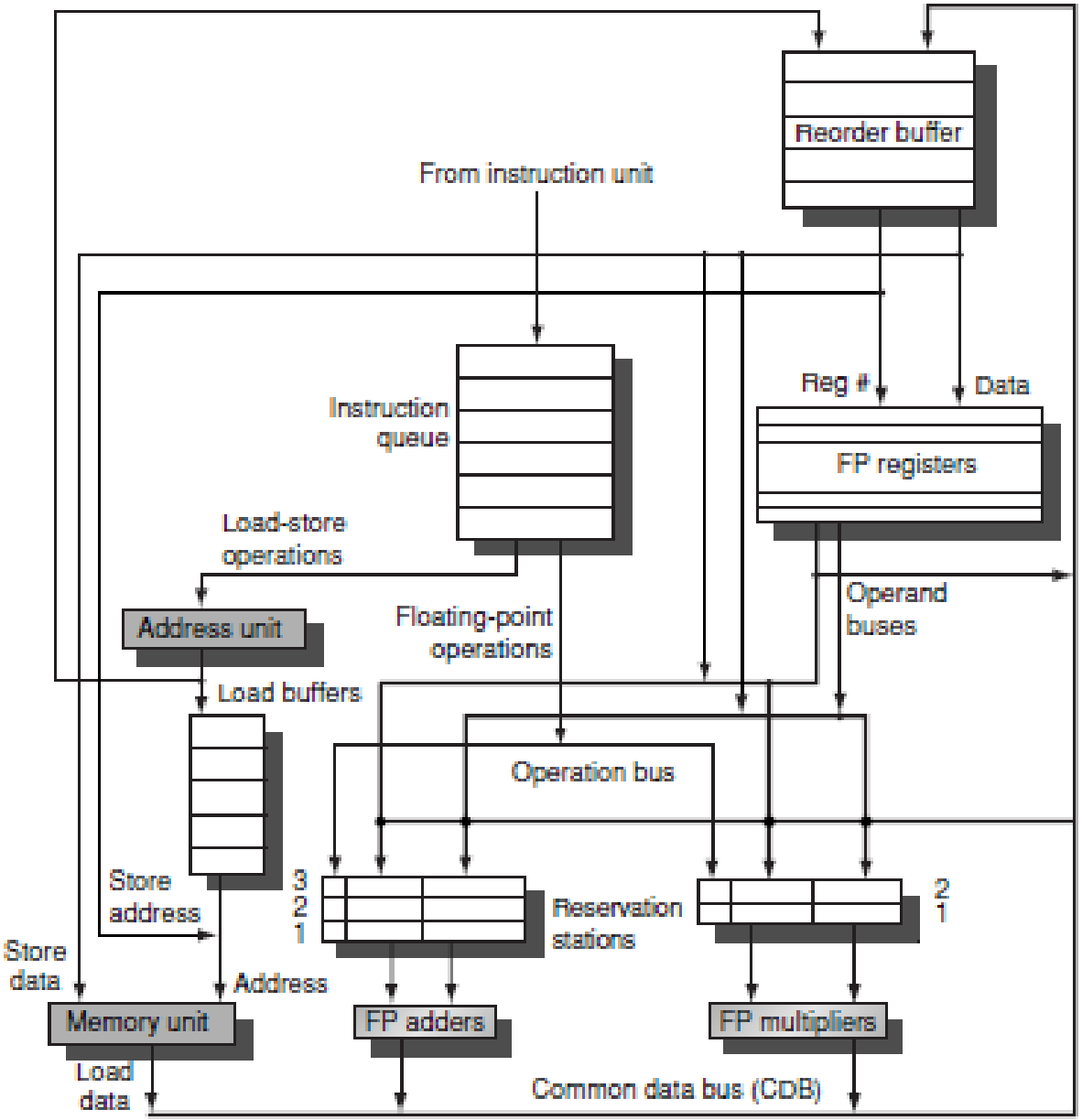
Como considerar previsão de desvio no escalonamento dinâmico das instruções

Escalonamento Dinâmico e Previsão de Desvio

- Especulação: Uso da previsão de desvio no Escalonamento Dinâmico
- Especulação melhora desempenho MAS…
- Importante ter um bom mecanismo de previsão de desvio
- Se houve especulação incorreta é necessário voltar e reiniciar a execução a partir do ponto em que foi feita a previsão incorreta :
  - O estado do processador tem que ser recuperado e as instruções corretas devem ser executadas

Escalonamento Dinâmico com Especulação

- Buffer para os resultados de instruções que não terminaram (uncommitted instructions): *reorder buffer*

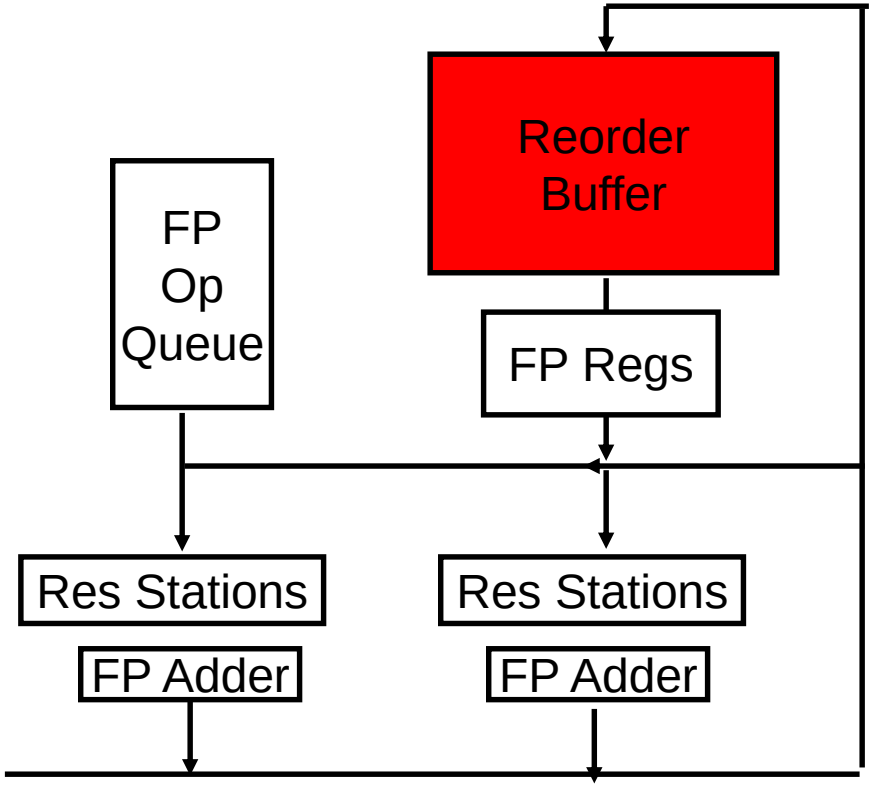


Escalonamento Dinâmico com Especulação

- Como garantir que a Especulação não vai introduzir erro?
- Técnica para especulação deve garantir *in-order completion* ou *commit*
- Término da instrução inclui duas etapas:
  - Término
  - Commit (escrita nos registradores)

Suporte de HW Especulação

- Buffer para os resultados de instruções que não terminaram que preserva a ordem (uncommitted instructions): *reorder buffer*
  - 3 campos: instr, reg. destino, valor
  - reorder buffer armazena valores intermediários
  - Depois que a instrução dá commit, o registrador é atualizado.
  - Permite desfazer instruções especuladas devido a um desvio previsto erradamente



Escalonamento Dinâmico com Especulação

1. Issue — pega a instrução da FP Op Queue

   Se há reservation station e reorder buffer slot livres: despacha instr & envia operandos & reorder buffer no. para o destino (este estágio é comumente chamado de "dispatch")

2. Execution — opera sobre os operandos (EX)

   Quando ambos os operandos estão prontos executa; se não monitora o CDB a espera do resultado; quando ambos estiverem na reservation station, executa; verifica se há RAW (comumente chamado de "issue")

3. Write result — termina a execução (WB)

   Escreve, usando o Common Data Bus, em todas FUs que estão esperando por esse valor e no reorder buffer; marca a reservation station como disponível.

4. Commit — atualiza o registrador com o resultado do reorder buffer

   Quando a instr. é a primeira no reorder buffer e o resultado está presente: atualiza o registrador com o resultado (ou store na memória) e remove a instr. do reorder buffer. Se foi um Mispredicted branch então flushes reorder buffer

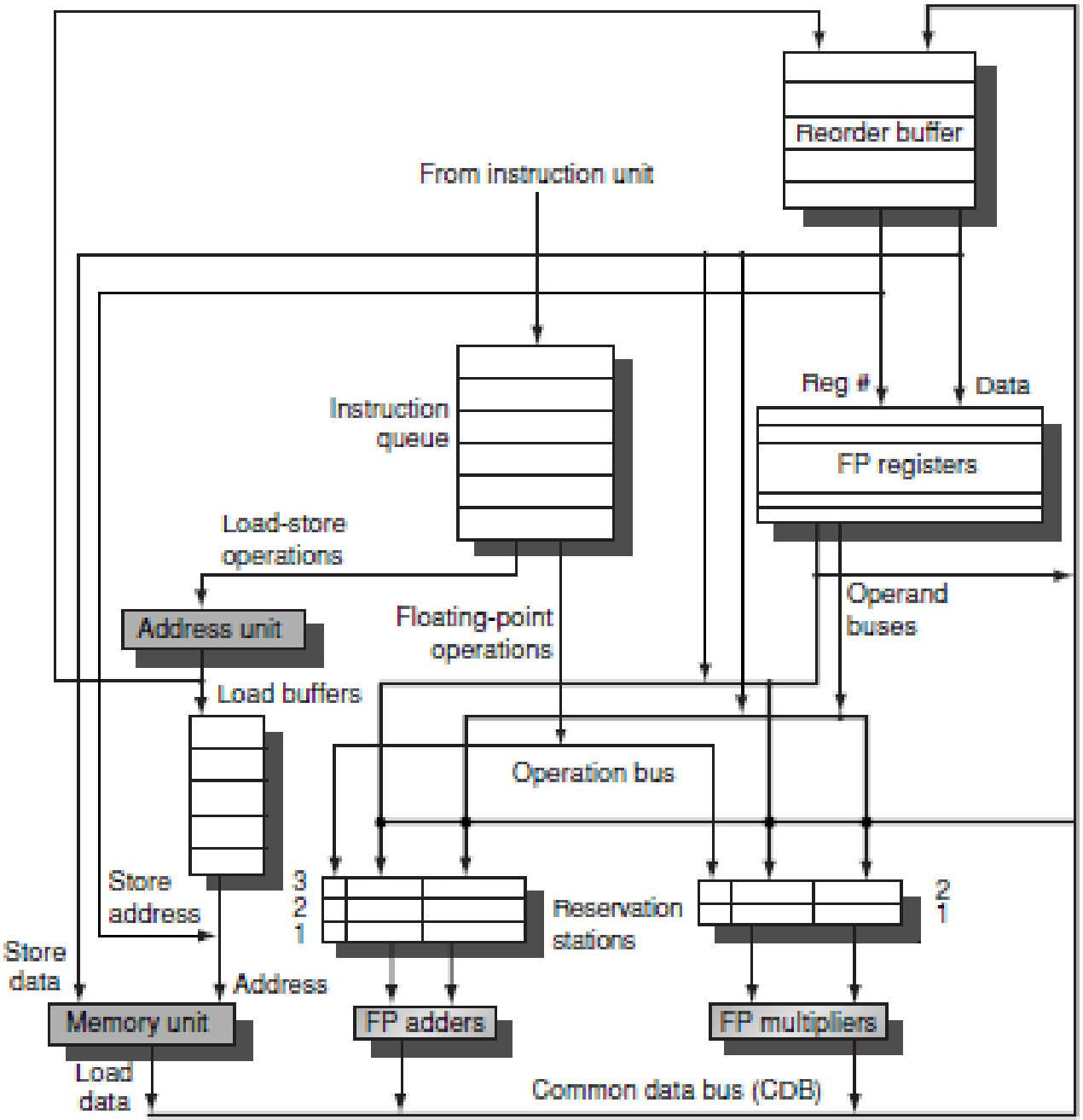
Escalonamento Dinâmico com Especulação

- Exemplo
  - Código:

```
L.D      F0, 0(R1)
MUL.D    F4, F0, F2
S.D      F4, 0(R1)
DADDIU   R1, R1, #-8
BNE      R1, R2, Loop
```

  - 2 iterações no loop
- Assumir
  - 2 ciclos para add
  - 6 ciclos para multiply
  - 2 adicionadores
  - 2 multiplicadores

Escalonamento Dinâmico com Especulação



Escalonamento Dinâmico com Especulação

Reorder buffer

	Op	Dest	Value	Ready
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				

Reservation stations:

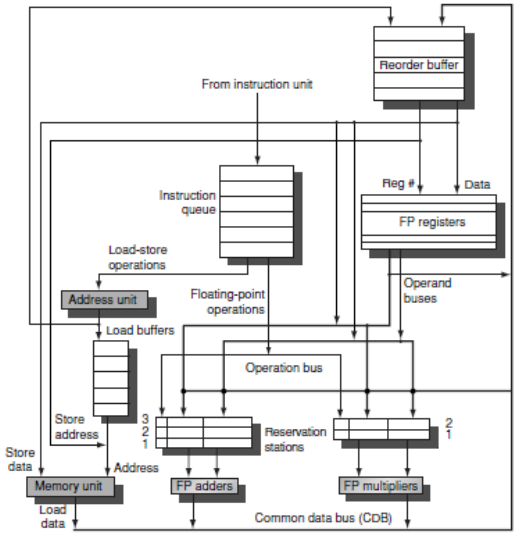
Name	Busy?	Op	Vj	Vk	Qj	Qk
Load1	N					
Load2	N					
Add1	N					
Add2	N					
Mult1	N					
Mult2	N					

Register result status table

F0	F2	F4	R1	R2

Pipeline diagram

Cycle	1	2	3	4	5
L.D F0,0(R1)					
MUL.D F4,F0,F2					
S.D F4,0(R1)					
DADDIU R1,R1,#-8					
BNE R1,R2,Loop					
L.D F0,0(R1)					
MUL.D F4,F0,F2					
S.D F4,0(R1)					
DADDIU R1,R1,#-8					
BNE R1,R2,Loop					



Ciclo 1-2 nento Dinâmico com Especulação

Reorder buffer

	Op	Dest	Value	Ready
1	L.D	F0		N
2				
3				
4				
5				
6				
7				
8				
9				
10				

Reservation stations:

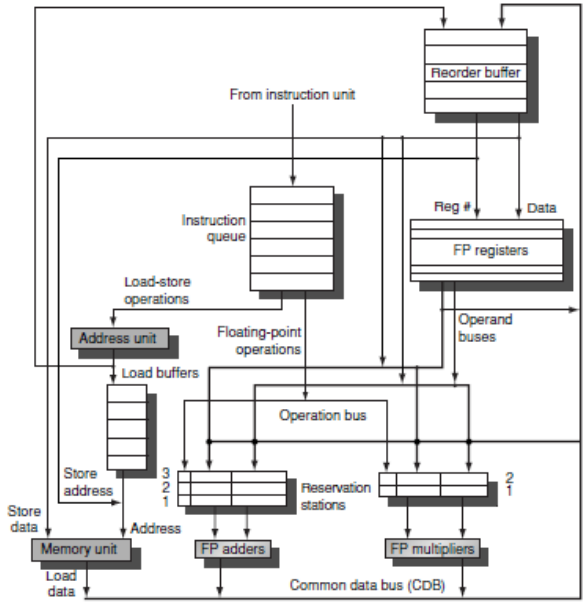
Name	Busy?	Op	Vj	Vk	Qj	Qk
Load1	Y	L.D	0	[R1]		
Load2	N					
Add1	N					
Add2	N					
Mult1	N					
Mult2	N					

Register result status table

F0	F2	F4	R1	R2
ROB1				

Pipeline diagram

Cycle	1	2	3	4	5
L.D F0,0(R1)	IF	IS			
MUL.D F4,F0,F2		IF			
S.D F4,0(R1)					
DADDIU R1,R1,#-8					
BNE R1,R2,Loop					
L.D F0,0(R1)					
MUL.D F4,F0,F2					
S.D F4,0(R1)					
DADDIU R1,R1,#-8					
BNE R1,R2,Loop					



Ciclo 3 amento Dinâmico com Especulação

Centro

ática
F · P · E

Reorder buffer

	Op	Dest	Value	Ready
1	L.D	F0		N
2	MUL.D	F4		N
3				
4				
5				
6				
7				
8				
9				
10				

Reservation stations:

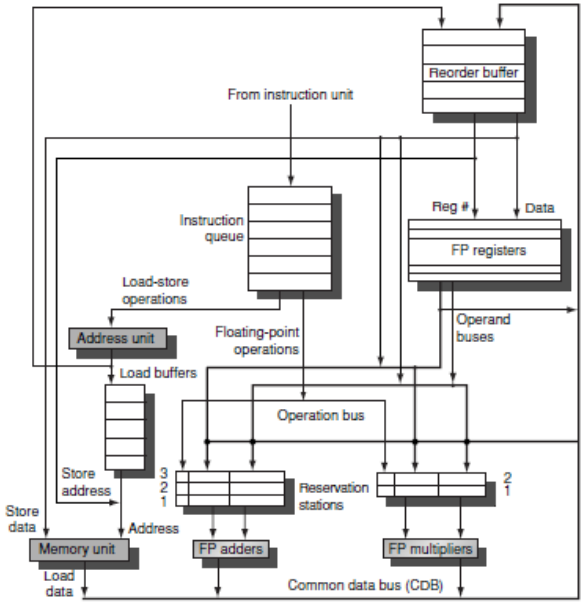
Name	Busy?	Op	Vj	Vk	Qj	Qk
Load1	Y	L.D	0	[R1]		
Load2	N					
Add1	N					
Add2	N					
Mult1	Y	MUL.D		[F2]	ROB1	
Mult2	N					

Register result status table

F0	F2	F4	R1	R2
ROB1		ROB2		

Pipeline diagram

Cycle	1	2	3	4	5
L.D F0,0(R1)	IF	IS	EX		
MUL.D F4,F0,F2		IF	IS		
S.D F4,0(R1)			IF		
DADDIU R1,R1,#-8					
BNE R1,R2,Loop					
L.D F0,0(R1)					
MUL.D F4,F0,F2					
S.D F4,0(R1)					
DADDIU R1,R1,#-8					
BNE R1,R2,Loop					



Ciclo 4 amento Dinâmico com Especulação

Reorder buffer

	Op	Dest	Value	Ready
1	L.D	F0		N
2	MUL.D	F4		N
3	S.D	0+R1	ROB2	N
4				
5				
6				
7				
8				
9				
10				

Reservation stations:

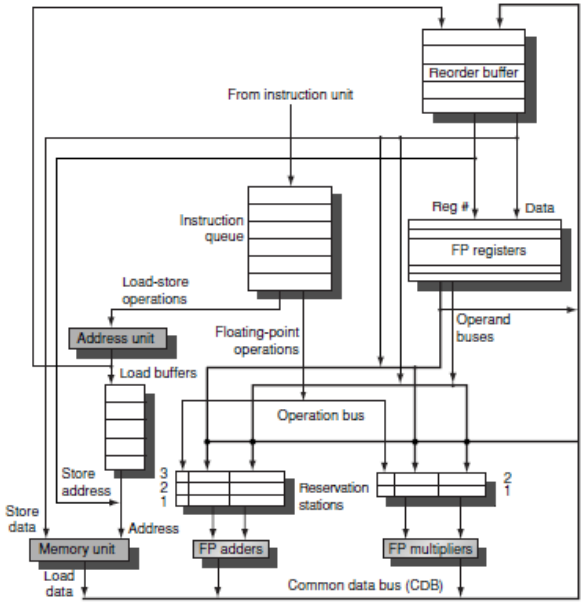
Name	Busy?	Op	Vj	Vk	Qj	Qk
Load1	Y	L.D	0	[R1]		
Load2	N					
Add1	N					
Add2	N					
Mult1	Y	MUL.D		[F2]	ROB1	
Mult2	N					

Register result status table

F0	F2	F4	R1	R2
ROB1		ROB2		

Pipeline diagram

Cycle	1	2	3	4	5
L.D F0,0(R1)	IF	IS	EX	M	
MUL.D F4,F0,F2		IF	IS	S	
S.D F4,0(R1)			IF	IS	
DADDIU R1,R1,#-8				IF	
BNE R1,R2,Loop					
L.D F0,0(R1)					
MUL.D F4,F0,F2					
S.D F4,0(R1)					
DADDIU R1,R1,#-8					
BNE R1,R2,Loop					



Ciclo 5 amento Dinâmico com Especulação

Centro

tica P · E

Reorder buffer

	Op	Dest	Value	Ready
1	L.D	F0	M[0+R1]	Y
2	MUL.D	F4		N
3	S.D	0+R1	ROB2	N
4	DADDIU	R1		N
5				
6				
7				
8				
9				
10				

Reservation stations:

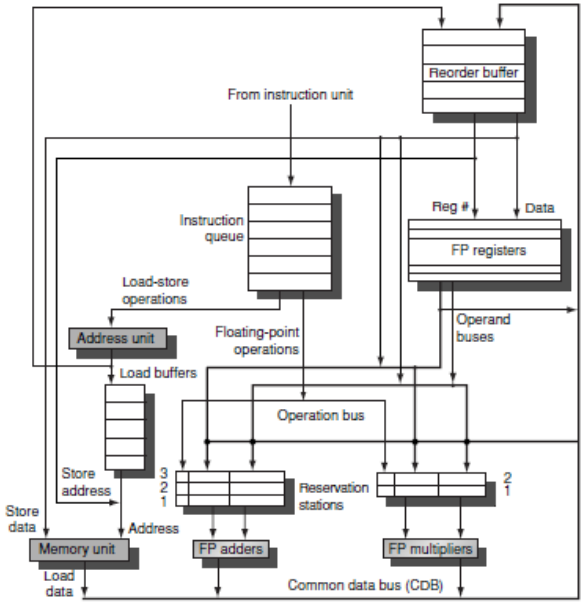
Name	Busy?	Op	Vj	Vk	Qj	Qk
Load1	N					
Load2	N					
Add1	Y	DADDIU	[R1]	-8		
Add2	N					
Mult1	Y	MUL.D	[F0]	[F2]		
Mult2	N					

Register result status table

F0	F2	F4	R1	R2
ROB1		ROB2	ROB4	

Pipeline diagram

Cycle	1	2	3	4	5
L.D F0,0(R1)	IF	IS	EX	M	WB
MUL.D F4,F0,F2		IF	IS	S	EX1
S.D F4,0(R1)			IF	IS	EX
DADDIU R1,R1,#-8				IF	IS
BNE R1,R2,Loop					IF
L.D F0,0(R1)					
MUL.D F4,F0,F2					
S.D F4,0(R1)					
DADDIU R1,R1,#-8					
BNE R1,R2,Loop					



Ciclo 6 amento Dinâmico com Especulação

Reorder buffer

	Op	Dest	Value	Ready
~~1~~	~~L.D~~	~~F0~~	~~M[0+R1]~~	~~Y~~
2	MUL.D	F4		N
3	S.D	0+R1	ROB2	N
4	DADDIU	R1		N
5	BNE	--		N
6				
7				
8				
9				
10				

Reservation stations:

Name	Busy?	Op	Vj	Vk	Qj	Qk
Load1	N					
Load2	N					
Add1	Y	DADDIU	[R1]	-8		
Add2	N					
Mult1	Y	MUL.D	[F0]	[F2]		
Mult2	N					

Register result status table

F0	F2	F4	R1	R2
		ROB2	ROB4	

Pipeline diagram

Cycle	1	2	3	4	5	6
L.D F0,0(R1)	IF	IS	EX	M	WB	C
MUL.D F4,F0,F2		IF	IS	S	EX1	EX2
S.D F4,0(R1)			IF	IS	EX	MEM
DADDIU R1,R1,#-8				IF	IS	EX1
BNE R1,R2,Loop					IF	IS
L.D F0,0(R1)						IF
MUL.D F4,F0,F2						
S.D F4,0(R1)						
DADDIU R1,R1,#-8						
BNE R1,R2,Loop						

Load commit

Ciclo 7 amento Dinâmico com Especulação

Centro nática · F · P · E

Reorder buffer

	Op	Dest	Value	Ready
1				
2	MUL.D	F4		N
3	S.D	0+R1	ROB2	N
4	DADDIU	R1		N
5	BNE	--		N
6	L.D	F0		N
7				
8				
9				
10				

Reservation stations:

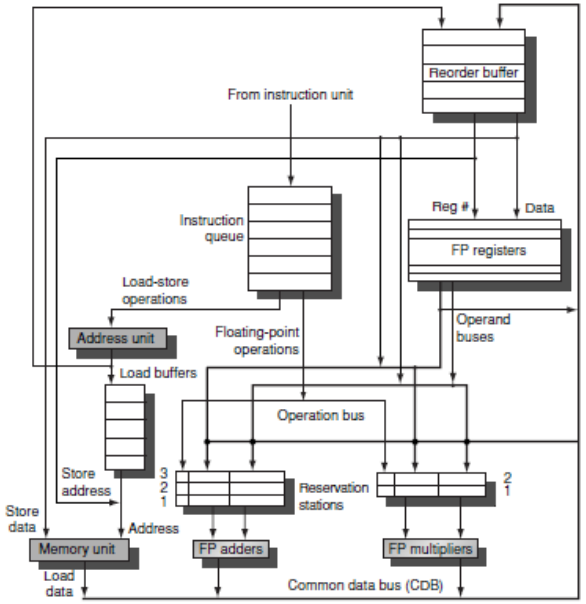
Name	Busy?	Op	Vj	Vk	Qj	Qk
Load1	Y	L.D	0			ROB4
Load2	N					
Add1	Y	DADDIU	[R1]	-8		
Add2	N					
Mult1	Y	MUL.D	[F0]	[F2]		
Mult2	N					

Register result status table

F0	F2	F4	R1	R2
ROB6		ROB2	ROB4	

Pipeline diagram

Cycle	1	2	3	4	5	6	7
L.D F0,0(R1)	IF	IS	EX	M	WB	C	
MUL.D F4,F0,F2		IF	IS	S	EX1	EX2	EX3
S.D F4,0(R1)			IF	IS	EX	MEM	S
DADDIU R1,R1,#-8				IF	IS	EX1	EX2
BNE R1,R2,Loop					IF	IS	S
L.D F0,0(R1)						IF	IS
MUL.D F4,F0,F2							IF
S.D F4,0(R1)							
DADDIU R1,R1,#-8							
BNE R1,R2,Loop							



Ciclo 8 amento Dinâmico com Especulação

Centro de Informática U·F·P·E

Reorder buffer

	Op	Dest	Value	Ready
1				
2	MUL.D	F4		N
3	S.D	0+R1	ROB2	N
4	DADDIU	R1	[R1]+(-8)	Y
5	BNE	--		N
6	L.D	F0		N
7	MUL.D	F4		N
8				
9				
10				

Reservation stations:

Name	Busy?	Op	Vj	Vk	Qj	Qk
Load1	Y	L.D	0	[R1]		
Load2	N					
Add1	N					
Add2	N					
Mult1	Y	MUL.D	[F0]	[F2]		
Mult2	Y	MUL.D		[F2]	ROB6	

Register result status table

F0	F2	F4	R1	R2
ROB6		ROB7	ROB4	

Pipeline diagram

Cycle	1	2	3	4	5	6	7	8
L.D F0,0(R1)	IF	IS	EX	M	WB	C		
MUL.D F4,F0,F2		IF	IS	S	EX1	EX2	EX3	EX4
S.D F4,0(R1)			IF	IS	EX	MEM	S	S
DADDIU R1,R1,#-8				IF	IS	EX1	EX2	WB
BNE R1,R2,Loop					IF	IS	S	EX
L.D F0,0(R1)						IF	IS	EX
MUL.D F4,F0,F2							IF	IS
S.D F4,0(R1)								IF
DADDIU R1,R1,#-8								
BNE R1,R2,Loop								

Ciclo 9 amento Dinâmico com Especulação

Centro de Informática U·F·P·E

Reorder buffer

	Op	Dest	Value	Ready
1				
2	MUL.D	F4		N
3	S.D	0+R1	ROB2	N
4	DADDIU	R1	[R1]+(-8)	Y
5	BNE	--		Y
6	L.D	F0		N
7	MUL.D	F4		N
8	S.D	0+R1	ROB7	N
9				
10				

Reservation stations:

Name	Busy?	Op	Vj	Vk	Qj	Qk
Load1	Y	L.D	0	[R1]		
Load2	N					
Add1	N					
Add2	N					
Mult1	Y	MUL.D	[F0]	[F2]		
Mult2	Y	MUL.D		[F2]	ROB6	

Register result status table

F0	F2	F4	R1	R2
ROB6		ROB7	ROB4	

Pipeline diagram

Cycle	1	2	3	4	5	6	7	8	9
L.D F0,0(R1)	IF	IS	EX	M	WB	C			
MUL.D F4,F0,F2		IF	IS	S	EX1	EX2	EX3	EX4	EX5
S.D F4,0(R1)			IF	IS	EX	MEM	S	S	S
DADDIU R1,R1,#-8				IF	IS	EX1	EX2	WB	--
BNE R1,R2,Loop					IF	IS	S	EX	--
L.D F0,0(R1)						IF	IS	EX	MEM
MUL.D F4,F0,F2							IF	IS	S
S.D F4,0(R1)								IF	IS
DADDIU R1,R1,#-8									IF
BNE R1,R2,Loop									

Ciclo 10 mento Dinâmico com Especulação

Reorder buffer

	Op	Dest	Value	Ready
1				
2	MUL.D	F4		N
3	S.D	0+R1	ROB2	N
4	DADDIU	R1	[R1]+(-8)	Y
5	BNE	--		Y
6	L.D	F0	M[0+R1]	Y
7	MUL.D	F4		N
8	S.D	0+R1	ROB7	N
9	DADDIU	R1		N
10				

Reservation stations:

Name	Busy?	Op	Vj	Vk	Qj	Qk
Load1	N					
Load2	N					
Add1	Y	DADDIU	[R1]+(-8)	-8		
Add2	N					
Mult1	Y	MUL.D	[F0]	[F2]		
Mult2	Y	MUL.D	[F0]	[F2]		

Register result status table

F0	F2	F4	R1	R2
ROB6		ROB7	ROB9	

Pipeline diagram

Cycle	1	2	3	4	5	6	7	8	9	10
L.D F0,0(R1)	IF	IS	EX	M	WB	C				
MUL.D F4,F0,F2		IF	IS	S	EX1	EX2	EX3	EX4	EX5	EX6
S.D F4,0(R1)			IF	IS	EX	MEM	S	S	S	S
DADDIU R1,R1,#-8				IF	IS	EX1	EX2	WB	--	--
BNE R1,R2,Loop					IF	IS	S	EX	--	--
L.D F0,0(R1)						IF	IS	EX	MEM	WB
MUL.D F4,F0,F2							IF	IS	S	EX1
S.D F4,0(R1)								IF	IS	EX
DADDIU R1,R1,#-8									IF	IS
BNE R1,R2,Loop										IF

Ciclo 11 mento Dinâmico com Especulação

Centro de Informática U·F·P·E

Reorder buffer

	Op	Dest	Value	Ready
1				
2	MUL.D	F4	[F0]*[F2]	Y
3	S.D	0+R1	[F0]*[F2]	Y
4	DADDIU	R1	[R1]+(-8)	Y
5	BNE	--		Y
6	L.D	F0	M[0+R1]	Y
7	MUL.D	F4		N
8	S.D	0+R1	ROB7	N
9	DADDIU	R1		N
10	BNE	--		N

Reservation stations:

Name	Busy?	Op	Vj	Vk	Qj	Qk
Load1	N					
Load2	N					
Add1	Y	DADDIU	[R1]+(-8)	-8		
Add2	N					
Mult1	N					
Mult2	Y	MUL.D	[F0]	[F2]		

Register result status table

F0	F2	F4	R1	R2
ROB6		ROB7	ROB9	

Pipeline diagram

Cycle	1	2	3	4	5	6	7	8	9	10	11
L.D F0,0(R1)	IF	IS	EX	M	WB	C					
MUL.D F4,F0,F2		IF	IS	S	EX1	EX2	EX3	EX4	EX5	EX6	WB
S.D F4,0(R1)			IF	IS	EX	MEM	S	S	S	S	--
DADDIU R1,R1,#-8				IF	IS	EX1	EX2	WB	--	--	--
BNE R1,R2,Loop					IF	IS	S	EX	--	--	--
L.D F0,0(R1)						IF	IS	EX	MEM	WB	--
MUL.D F4,F0,F2							IF	IS	S	EX1	EX2
S.D F4,0(R1)								IF	IS	EX	MEM
DADDIU R1,R1,#-8									IF	IS	EX1
BNE R1,R2,Loop										IF	IS

Ciclo 12 mento Dinâmico com Especulação

Centro de Informática U·F·P·E

Reorder buffer

	Op	Dest	Value	Ready
1				
~~2~~	~~MUL.D~~	~~F4~~	~~[F0]*[F2]~~	~~Y~~
3	S.D	0+R1	[F0]*[F2]	Y
4	DADDIU	R1	[R1]+(-8)	Y
5	BNE	--		Y
6	L.D	F0	M[0+R1]	Y
7	MUL.D	F4		N
8	S.D	0+R1	ROB7	N
9	DADDIU	R1		N
10	BNE	--		N

Reservation stations:

Name	Busy?	Op	Vj	Vk	Qj	Qk
Load1	N					
Load2	N					
Add1	Y	DADDIU	[R1]+(-8)	-8		
Add2	N					
Mult1	N					
Mult2	Y	MUL.D	[F0]	[F2]		

Register result status table

F0	F2	F4	R1	R2
ROB6		ROB7	ROB9	

Pipeline diagram

Cycle	1	2	3	4	5	6	7	8	9	10	11	12
L.D F0,0(R1)	IF	IS	EX	M	WB	C						
MUL.D F4,F0,F2		IF	IS	S	EX1	EX2	EX3	EX4	EX5	EX6	WB	C
S.D F4,0(R1)			IF	IS	EX	MEM	S	S	S	S	--	--
DADDIU R1,R1,#-8				IF	IS	EX1	EX2	WB	--	--	--	--
BNE R1,R2,Loop					IF	IS	S	EX	--	--	--	--
L.D F0,0(R1)						IF	IS	EX	MEM	WB	--	--
MUL.D F4,F0,F2							IF	IS	S	EX1	EX2	EX3
S.D F4,0(R1)								IF	IS	EX	MEM	S
DADDIU R1,R1,#-8									IF	IS	EX1	EX2
BNE R1,R2,Loop										IF	IS	S

Ciclo 13 mento Dinâmico com Especulação

Centro de Informática U·F·P·E

Reorder buffer

	Op	Dest	Value	Ready
1				
2				
~~3~~	~~S.D~~	~~0+R1~~	~~[F0]*[F2]~~	~~Y~~
4	DADDIU	R1	[R1]+(-8)	Y
5	BNE	--		Y
6	L.D	F0	M[0+R1]	Y
7	MUL.D	F4		N
8	S.D	0+R1	ROB7	N
9	DADDIU	R1	[R1]+(-16)	Y
10	BNE	--		N

Reservation stations:

Name	Busy?	Op	Vj	Vk	Qj	Qk
Load1	N					
Load2	N					
Add1	N					
Add2	N					
Mult1	N					
Mult2	Y	MUL.D	[F0]	[F2]		

Register result status table

F0	F2	F4	R1	R2
ROB6		ROB7	ROB9	

Pipeline diagram

Cycle	1	2	3	4	5	6	7	8	9	10	11	12	13
L.D F0,0(R1)	IF	IS	EX	M	WB	C							
MUL.D F4,F0,F2		IF	IS	S	EX1	EX2	EX3	EX4	EX5	EX6	WB	C	
S.D F4,0(R1)			IF	IS	EX	MEM	S	S	S	S	--	--	C
DADDIU R1,R1,#-8				IF	IS	EX1	EX2	WB	--	--	--	--	--
BNE R1,R2,Loop					IF	IS	S	EX	--	--	--	--	--
L.D F0,0(R1)						IF	IS	EX	MEM	WB	--	--	--
MUL.D F4,F0,F2							IF	IS	S	EX1	EX2	EX3	EX4
S.D F4,0(R1)								IF	IS	EX	MEM	S	S
DADDIU R1,R1,#-8									IF	IS	EX1	EX2	WB
BNE R1,R2,Loop										IF	IS	S	EX

Ciclo 14 mento Dinâmico com Especulação

Centro de Informática U·F·P·E

Reorder buffer

	Op	Dest	Value	Ready
1				
2				
3				
4	~~DADDIU~~	~~R1~~	~~[R1]+(-8)~~	~~Y~~
5	BNE	--		Y
6	L.D	F0	M[0+R1]	Y
7	MUL.D	F4		N
8	S.D	0+R1	ROB7	N
9	DADDIU	R1	[R1]+(-16)	Y
10	BNE	--		Y

Reservation stations:

Name	Busy?	Op	Vj	Vk	Qj	Qk
Load1	N					
Load2	N					
Add1	N					
Add2	N					
Mult1	N					
Mult2	Y	MUL.D	[F0]	[F2]		

Register result status table

F0	F2	F4	R1	R2
ROB6		ROB7	ROB9	

Pipeline diagram

Cycle	2	3	4	5	6	7	8	9	10	11	12	13	14
L.D F0,0(R1)	IS	EX	M	WB	C								
MUL.D F4,F0,F2	IF	IS	S	EX1	EX2	EX3	EX4	EX5	EX6	WB	C		
S.D F4,0(R1)		IF	IS	EX	MEM	S	S	S	S	--	--	C	
DADDIU R1,R1,#-8			IF	IS	EX1	EX2	WB	--	--	--	--	--	C
BNE R1,R2,Loop				IF	IS	S	EX	--	--	--	--	--	--
L.D F0,0(R1)					IF	IS	EX	MEM	WB	--	--	--	--
MUL.D F4,F0,F2						IF	IS	S	EX1	EX2	EX3	EX4	EX5
S.D F4,0(R1)							IF	IS	EX	MEM	S	S	S
DADDIU R1,R1,#-8								IF	IS	EX1	EX2	WB	--
BNE R1,R2,Loop									IF	IS	S	EX	--

Ciclo 15 mento Dinâmico com Especulação

Centro de Informática U·F·P·E

Reorder buffer

	Op	Dest	Value	Ready
1				
2				
3				
4				
~~5~~	~~BNE~~	~~--~~		~~Y~~
6	L.D	F0	M[0+R1]	Y
7	MUL.D	F4		N
8	S.D	0+R1	ROB7	N
9	DADDIU	R1	[R1]+(-16)	Y
10	BNE	--		Y

Reservation stations:

Name	Busy?	Op	Vj	Vk	Qj	Qk
Load1	N					
Load2	N					
Add1	N					
Add2	N					
Mult1	N					
Mult2	Y	MUL.D	[F0]	[F2]		

Register result status table

F0	F2	F4	R1	R2
ROB6		ROB7	ROB9	

Pipeline diagram

Cycle	3	4	5	6	7	8	9	10	11	12	13	14	15
L.D F0,0(R1)	EX	M	WB	C									
MUL.D F4,F0,F2	IS	S	EX1	EX2	EX3	EX4	EX5	EX6	WB	C			
S.D F4,0(R1)	IF	IS	EX	MEM	S	S	S	S	--	--	C		
DADDIU R1,R1,#-8		IF	IS	EX1	EX2	WB	--	--	--	--	--	C	
BNE R1,R2,Loop			IF	IS	S	EX	--	--	--	--	--	--	C
L.D F0,0(R1)				IF	IS	EX	MEM	WB	--	--	--	--	--
MUL.D F4,F0,F2					IF	IS	S	EX1	EX2	EX3	EX4	EX5	EX6
S.D F4,0(R1)						IF	IS	EX	MEM	S	S	S	S
DADDIU R1,R1,#-8							IF	IS	EX1	EX2	WB	--	--
BNE R1,R2,Loop								IF	IS	S	EX	--	--

Ciclo 16 mento Dinâmico com Especulação

Centro de Informática U·F·P·E

Reorder buffer

	Op	Dest	Value	Ready
1				
2				
3				
4				
5				
~~6~~	~~L.D~~	~~F0~~	~~M[0+R1]~~	~~Y~~
7	MUL.D	F4	[F0]*[F2]	Y
8	S.D	0+R1	[F0]*[F2]	Y
9	DADDIU	R1	[R1]+(-16)	Y
10	BNE	--		Y

Reservation stations:

Name	Busy?	Op	Vj	Vk	Qj	Qk
Load1	N					
Load2	N					
Add1	N					
Add2	N					
Mult1	N					
Mult2	N					

Register result status table

F0	F2	F4	R1	R2
		ROB7	ROB9	

Pipeline diagram

Cycle	4	5	6	7	8	9	10	11	12	13	14	15	16
L.D F0,0(R1)	M	WB	C										
MUL.D F4,F0,F2	S	EX1	EX2	EX3	EX4	EX5	EX6	WB	C				
S.D F4,0(R1)	IS	EX	MEM	S	S	S	S	--	--	C			
DADDIU R1,R1,#-8	IF	IS	EX1	EX2	WB	--	--	--	--	--	C		
BNE R1,R2,Loop		IF	IS	S	EX	--	--	--	--	--	--	C	
L.D F0,0(R1)			IF	IS	EX	MEM	WB	--	--	--	--	--	C
MUL.D F4,F0,F2				IF	IS	S	EX1	EX2	EX3	EX4	EX5	EX6	WB
S.D F4,0(R1)					IF	IS	EX	MEM	S	S	S	S	--
DADDIU R1,R1,#-8						IF	IS	EX1	EX2	WB	--	--	--
BNE R1,R2,Loop							IF	IS	S	EX	--	--	--

Ciclo 17 mento Dinâmico com Especulação

Centro de Informática U·F·P·E

Reorder buffer

	Op	Dest	Value	Ready
1				
2				
3				
4				
5				
6				
~~7~~	~~MUL.D~~	~~F4~~	~~[F0]*[F2]~~	~~Y~~
~~8~~	~~S.D~~	~~0+R1~~	~~[F0]*[F2]~~	~~Y~~
~~9~~	~~DADDIU~~	~~R1~~	~~[R1]+(-16)~~	~~Y~~
~~10~~	~~BNE~~	~~--~~		~~Y~~

Reservation stations:

Name	Busy?	Op	Vj	Vk	Qj	Qk
Load1	N					
Load2	N					
Add1	N					
Add2	N					
Mult1	N					
Mult2	N					

Register result status table

F0	F2	F4	R1	R2
		~~ROB7~~	~~ROB9~~	

Pipeline diagram

Cycle	8	9	10	11	12	13	14	15	16	17	18	19	20
L.D F0,0(R1)													
MUL.D F4,F0,F2	EX4	EX5	EX6	WB	C								
S.D F4,0(R1)	S	S	S	--	--	C							
DADDIU R1,R1,#-8	WB	--	--	--	--	--	C						
BNE R1,R2,Loop	EX	--	--	--	--	--	--	C					
L.D F0,0(R1)	EX	MEM	WB	--	--	--	--	--	C				
MUL.D F4,F0,F2	IS	S	EX1	EX2	EX3	EX4	EX5	EX6	WB	C			
S.D F4,0(R1)	IF	IS	EX	MEM	S	S	S	S	--	--	C		
DADDIU R1,R1,#-8		IF	IS	EX1	EX2	WB	--	--	--	--	--	C	
BNE R1,R2,Loop			IF	IS	S	EX	--	--	--	--	--	--	C

Escalonamento Dinâmico com Especulação

Cycle	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
L.D F0,0(R1)	IF	IS	EX	M	WB	C														
MUL.D F4,F0,F2		IF	IS	S	EX1	EX2	EX3	EX4	EX5	EX6	WB	C								
S.D F4,0(R1)			IF	IS	EX	MEM	S	S	S	S	--	--	C							
DADDIU R1,R1,#-8				IF	IS	EX1	EX2	WB	--	--	--	--	--	C						
BNE R1,R2,Loop					IF	IS	S	EX	--	--	--	--	--	--	C					
L.D F0,0(R1)						IF	IS	EX	MEM	WB	--	--	--	--	--	C				
MUL.D F4,F0,F2							IF	IS	S	EX1	EX2	EX3	EX4	EX5	EX6	WB	C			
S.D F4,0(R1)								IF	IS	EX	MEM	S	S	S	S	--	--	C		
DADDIU R1,R1,#-8									IF	IS	EX1	EX2	WB	--	--	--	--	--	C	
BNE R1,R2,Loop										IF	IS	S	EX	--	--	--	--	--	--	C

Renomeamento de Registradores

- “Reservation stations” e “reorder buffer” permitem armazenamento temporário
- Uma instrução vai para a “reservation station”
  - Se o operando está disponível no banco de registradores ou no “reorder buffer”
    - Copiada para “reservation station”
  - Se o operando não está disponível
    - Este será enviado para a “reservation station” e reorder buffer pela unidade funcional
- Armazenamento Temporário: Banco extra de registradores

Vantagens da Especulação Dinâmica

- Nem todos retardos são previsíveis
  - e.g., cache misses
- Nem sempre é possível desenrolar loops
  - Endereço de desvio é determinado em tempo de execução
- Implementações diferentes do mesmo ISA terá diferentes conflitos e retardos.

Multiple Issue funcioma?

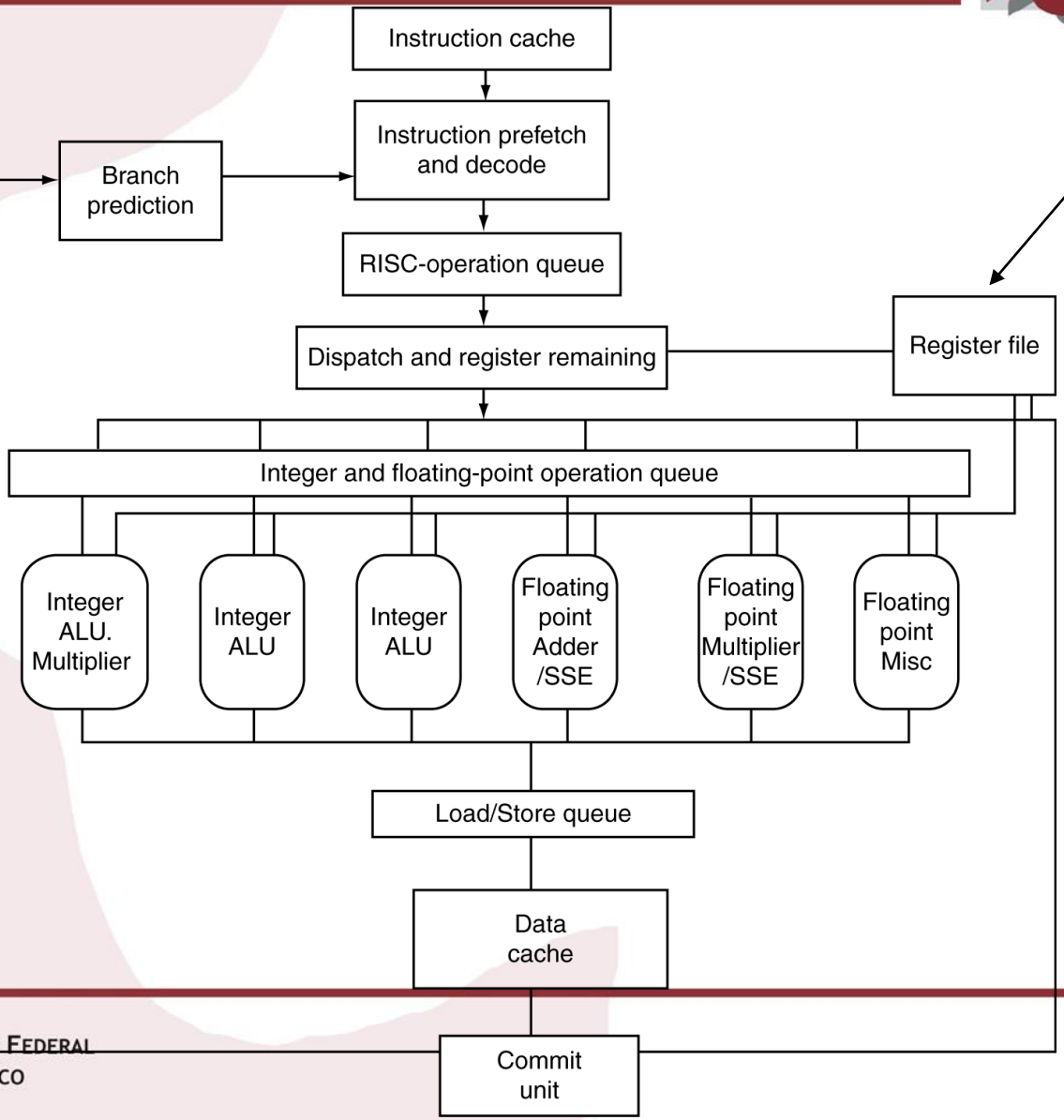
- Sim, mas não como se espera….
- Programas tem dependências que limitam ILP
- Algumas dependências são difíceis de se eliminar…
  - Dependencias com endereços de memória
- Dificuldade de descobrir paralelismo
  - Com o aumento do número de instruções a serem analizadas
- Tempo de acesso da memória
  - Difícil manter pipeline cheio
- Especulação pode ajudar se “bem feita”

Eficiência Energética

- Complexidade de escalonamento dinâmico e especulação eleva consumo de energia e dissipação de potência

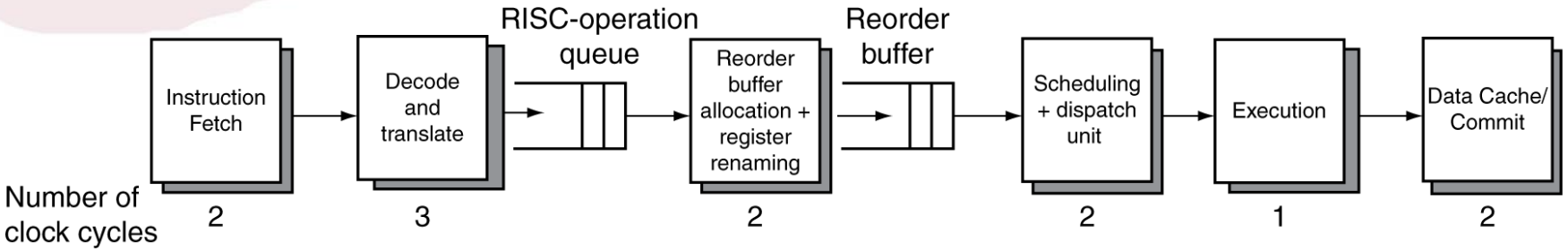
Microprocessor	Year	Clock Rate	Pipeline Stages	Issue width	Out-of-order/ Speculation	Cores	Power
i486	1989	25MHz	5	1	No	1	5W
Pentium	1993	66MHz	5	2	No	1	10W
Pentium Pro	1997	200MHz	10	3	Yes	1	29W
P4 Willamette	2001	2000MHz	22	3	Yes	1	75W
P4 Prescott	2004	3600MHz	31	3	Yes	1	103W
Core	2006	2930MHz	14	4	Yes	2	75W
UltraSparc III	2003	1950MHz	14	4	No	1	90W
UltraSparc T1	2005	1200MHz	6	1	No	8	70W

Microarquitetura do Opteron X4



Opteron X4 Fluxo do Pipeline

- Operações Inteiro



  - FP tem 5 estágios a mais
  - Até 106 RISC-ops em execução
- Bottlenecks
  - Instruções complexas com dependências
  - Erros na previsão de desvio
  - Tempo de acesso a memória

Centro de Informática U·F·P·E

Comparando as diversas arquiteturas...

Critérios de Otimização

Número médio de ciclos por instruções

SUPERESCALARES

Número de instruções por programa

VLIW

Tempo do ciclo do processador

SUPERPIPELINE

Várias Instruções por Ciclo

Detecção das Dependências de Controle

Superescalar

hardware

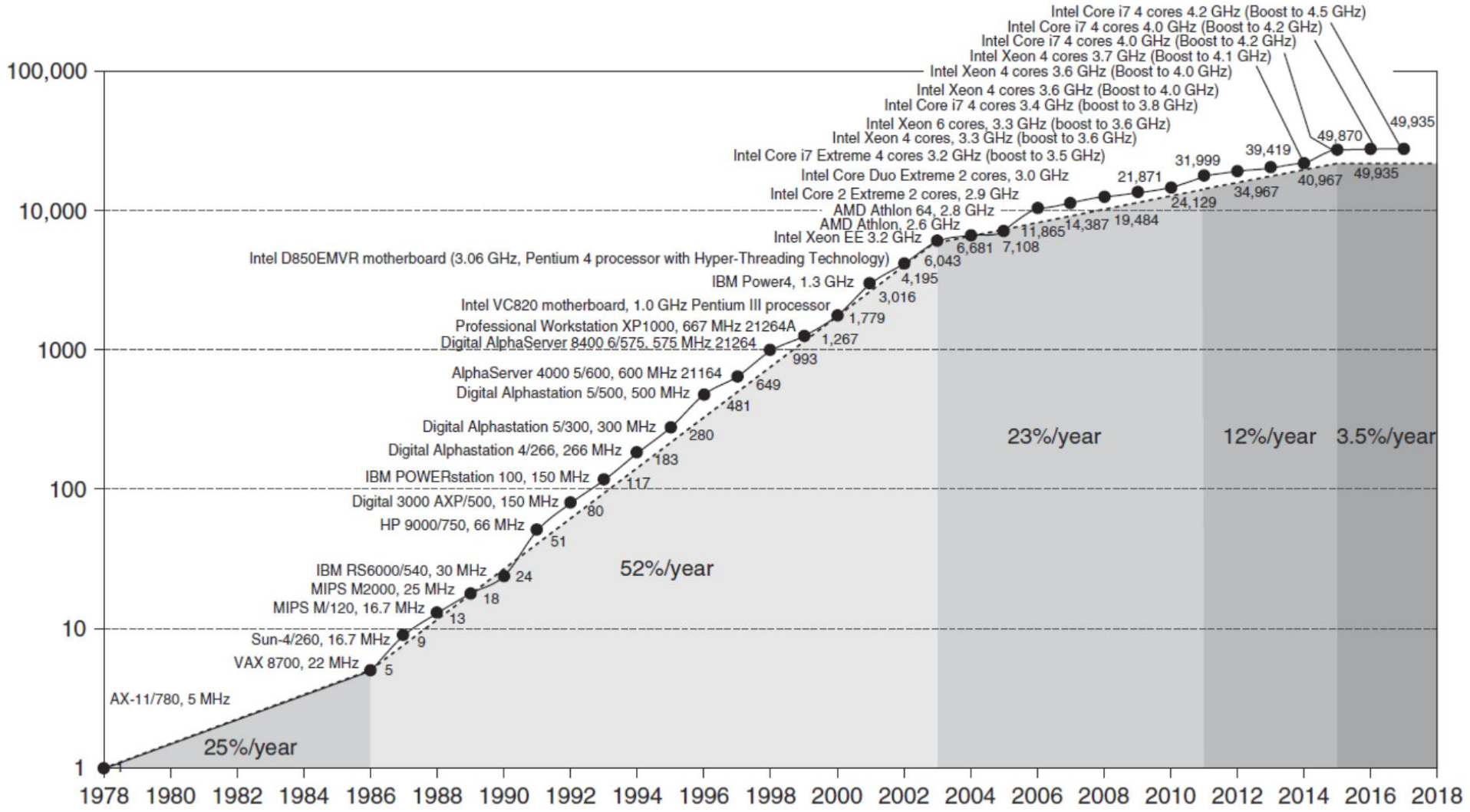
VLIW (very long instruction word)

compilador

Compatibilidade de Software

- **Importância:**
  - Garantir longevidade do software
  - Redução do “Time-to-market”
- **Código fonte**
  - Otimizações por Software
- **Código Objeto** (Binário)
  - Otimizações por Hardware

Desempenho dos Processadores



Conclusões

- ISA influencia no projeto da unidade de processamento e controle (e vice-versa)
- Pipelining melhora taxa de execução das instruções explorando paralelismo
  - Mais instruções completadas por segundo
  - Latencia de cada instrução não é reduzida
- Conflitos: estrutural, dado e controle
- Multiple issue e Escalonamento Dinamico (ILP)
  - Dependências limitam paralelismo de instrução
  - Complexidade levam ao limite de potência

Conclusões

- Necessidade e oportunidades para explorer paralelismo de thread.
  - Paralelismo explícito
- Vale a pena explorer paralelismo de thread em UM PROCESSADOR?
  - Aumento da complexidade e consume do potência
- Multiplos processadores com menor complexidade cada podem ser mais eficientes.