

MC542

Organização de Computadores Teoria e Prática

2007

Prof. Paulo Cesar Centoducatte

ducatte@ic.unicamp.br

www.ic.unicamp.br/~ducatte

MO401

Arquitectura de Computadores

Algoritmo de Tomasulo

**"Computer Architecture: A Quantitative
Approach" - (Capítulo 3)**

Vantagens de Scheduling Dinâmico

- Trata de casos que não são conhecidos em tempo de compilação
 - Casos que envolvem referências à memória
- Simplifica o compilador
- Permite que um código compilado para um pipeline execute de forma eficiente em um pipeline diferente
- Hardware speculation - técnica com bom desempenho que usa scheduling dinâmico como base

HW : Paralelismo de Instruções

- Idéia Principal: permitir que instruções após a que está em "stall" prosigam

```
DIVD  F0, F2, F4
ADDD  F10, F0, F8
SUBD  F12, F8, F14
```

- Habilitar **out-of-order execution** e permitir **out-of-order completion**
- Diferenciar quando uma instrução *inicia a execução* e quando ela *completa a execução* em 2 tempos, em ambos ela está *em execução*
- Em um pipeline com schedule dinâmico todas as instruções passam pelo estágio **issue** (**decodificação**, **hazard estrutural?**) em ordem (**in-order issue**)

HW : Paralelismo de Instruções

- **out-of-order execution:** possibilita a ocorrência de hazards WAR e WAW

div.d	f0, f2, f4
add.d	f6, f0, f8
sub.d	f8, f10, f14
mul.d	f6, f10, f8

add.d e sub.d => (f8) WAR

- add.d espera por div.d

add.d e mul.d => (f6) WAW

HW : Paralelismo de Instruções

- **out-of-order completion:** problemas com execuções
- **out-of-order completion** deve preservar o comportamento sob execuções como se fosse executado em uma máquina **in-order**
- **Processadores com scheduling dinâmico** preservam o comportamento sob exceções garantindo que as instruções não possam gerar exceções até que o processador saiba que a instrução que gerou a exceção está sendo completada.

HW : Paralelismo de Instruções

- **Processadores com scheduling dinâmico** podem gerar exceções imprecisas: uma exceção é dita imprecisa se o estado do processador quando ela foi gerada não corresponde exatamente ao estado que ela ocorreria se a instrução fosse executada seqüencialmente.

Exemplo:

- Uma instrução fora de ordem já foi completada e uma instrução anterior a ela gera a exceção
- Uma instrução fora de ordem ainda não foi completada e uma instrução posterior a ela gera a exceção

Scheduling Dinâmico Implementação - MIPS

- Pipeline simples tem 1 estágio que verifica se há hazard estrutural e de dados: **Instruction Decode (ID)**, também chamado de Instruction Issue
- Quebrar o estágio **ID** do pipeline de 5 estágios em dois estágios:
 - **Issue** — Decodificação das instruções, verificação de hazards estruturais
 - **Read operands** — *Espera até não haver data hazards, então lê os operandos*

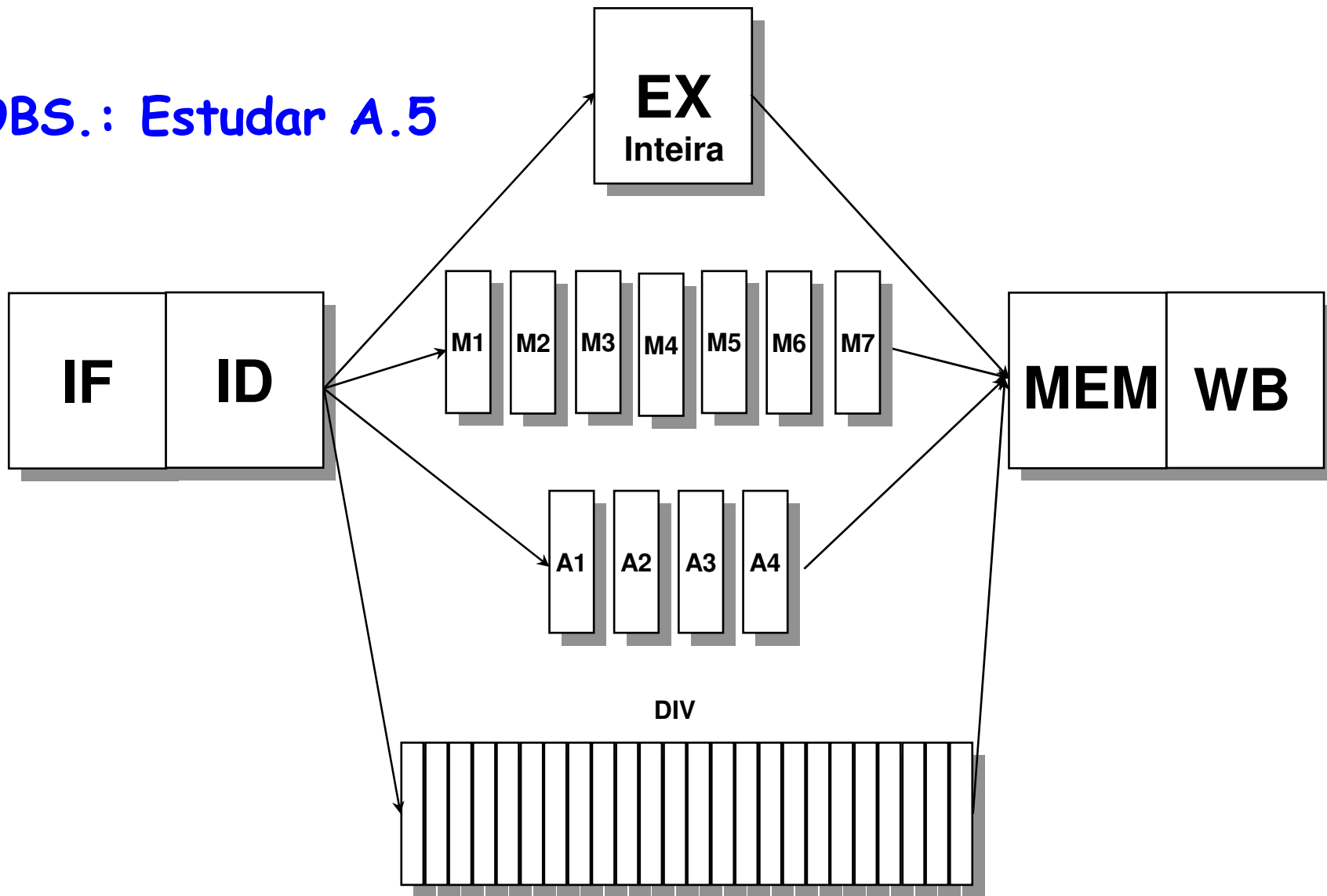
Scheduling Dinâmico

Implementação - MIPS

- Estágio **EX** segue o de leitura de operandos como no pipeline simples.
- **OBS.:**
 - A execução pode levar múltiplos ciclos, dependendo da instrução
 - O pipeline permite múltiplas instruções em EX, tem múltiplas unidades funcionais (FUs)

Scheduling Dinâmico Implementação - MIPS

OBS.: Estudar A.5



Algoritmo Dinâmico : Algoritmo de Tomasulo

- IBM 360/91 (1967 - não havia caches; tempo de acesso à memória grande e instruções de FP com grandes latências (delay))
- Idéia: Alto desempenho sem compilador especial
- Um pequeno número de registradores floating point (4 no 360) evita um bom scheduling das operações pelo compilador.
 - Tomasulo: Como ter efetivamente mais registradores ? Como resolver os hazards RAW, WAW e RAW?
 - seguir quando os operandos estiverem prontos e renaming implementado no hardware!
- Descendentes:
 - Alpha 21264, HP 8000, MIPS 10000, Pentium III, PowerPC 604, ...

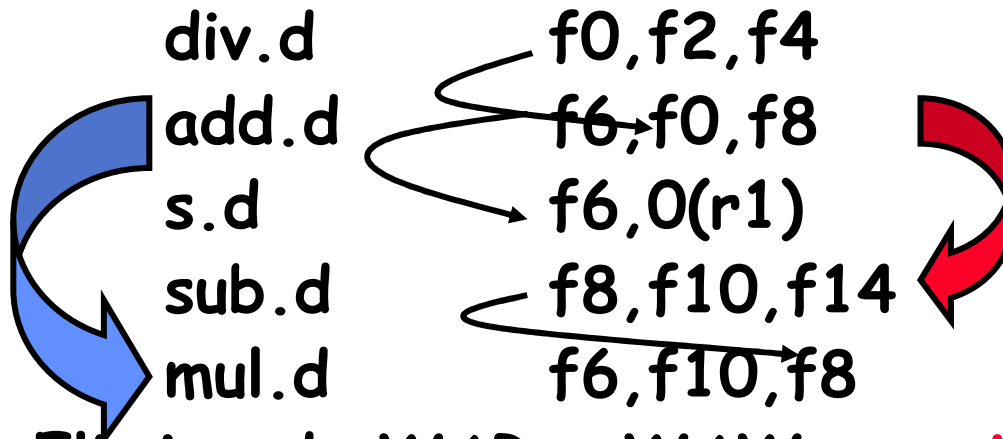
Algoritmo de Tomasulo

- Controle & buffers distribuídos na Function Units (FU)
 - FU buffers chamado de "reservation stations"; mantém operandos pendentos
- Substituição dos Registradores nas instruções por valores ou apontadores para a reservation stations (RS): denominado register renaming ;
 - Evita os hazards WAR e WAW
 - Se existe mais reservation stations que registradores, então pode-se fazer otimizações não realizadas pelos compiladores
- Resultados da RS para a FU, (sem usar os registradores), broadcasts dos resultados para todas as FUs usando o Common Data Bus
- Load e Stores tratados como FUs com RSs

Algoritmo de Tomasulo

Register Rename

- **WAR(f8); WAW(f6)** e RAW(f0, f6 e f8)



- Eliminando WAR e WAW - **register rename**

- Suponha dois registradores temporários S e T

div.d f0, f2, f4
 add.d **S**, f0, f8
 s.d **S**, 0(r1)
 sub.d **T**, f10, f14
 mul.d f6, f10, **T**

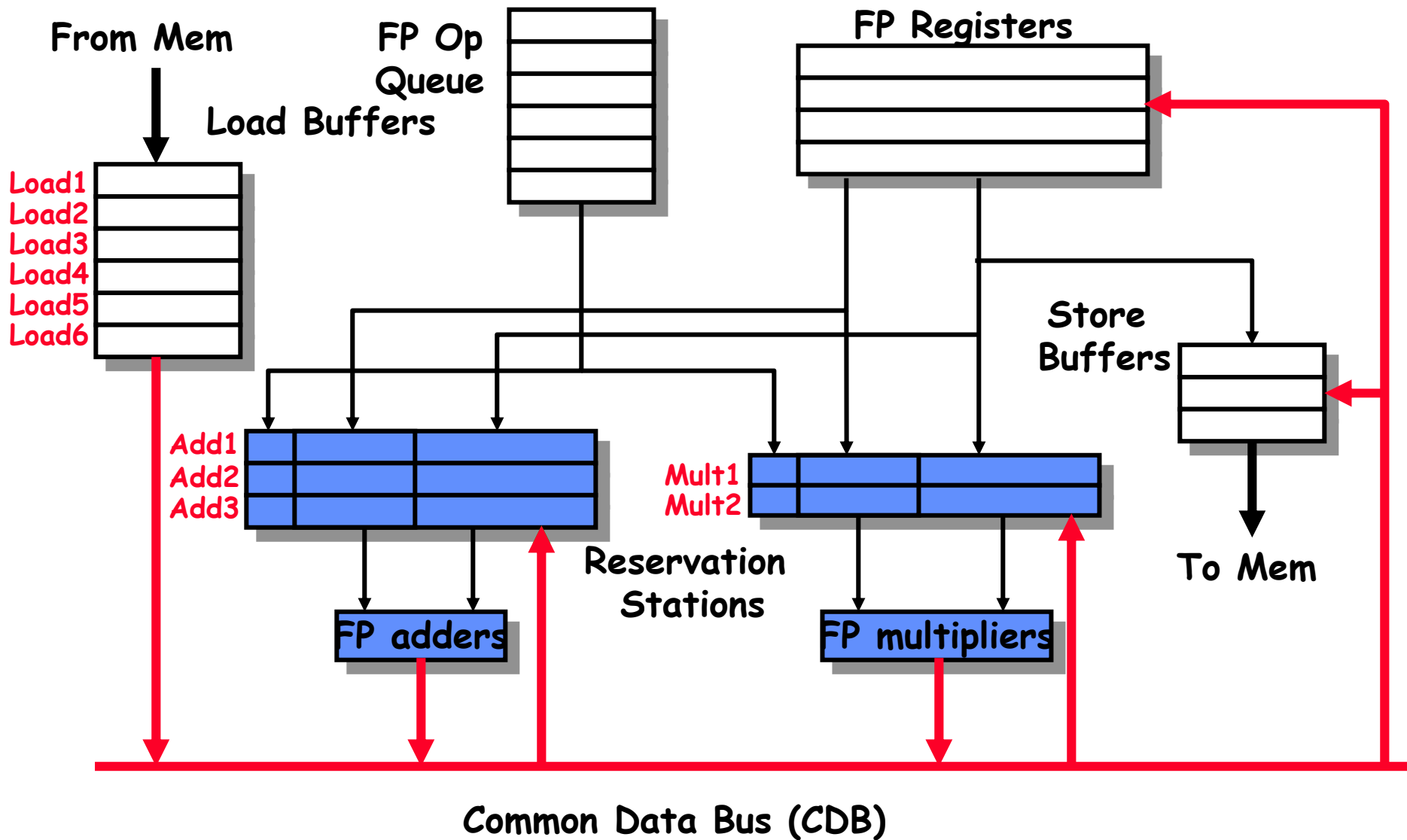
- 1) F8 deve ser substituído por T no resto do código - requer análise mais sofisticada (branches, ...)
- 2) Neste exemplo o **register rename** pode ser realizado pelo compilador (análise estática)

Algoritmo de Tomasulo

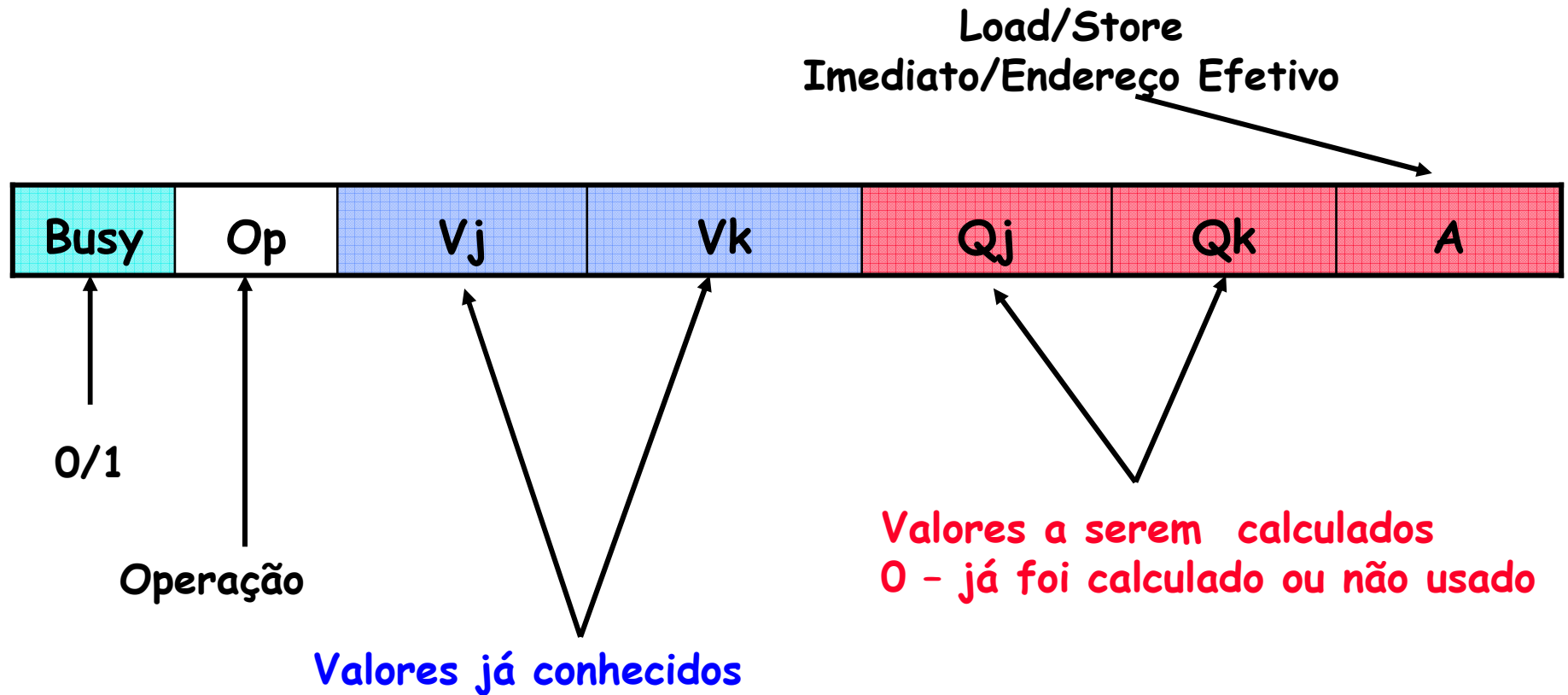
Exemplo

- Foco: Unidades de ponto-flutuante e load-store
- Cada estágio pode ter um número arbitrário de ciclos
- Múltiplas unidades funcionais
- Diferentes instruções possuem tempos diferentes no estágio EX
- Unidades disponíveis: **load-store**; **mult** e **adder**

Estrutura Básica de uma Implementação do Algoritmo de Tomasulo (para o MIPS)



Reservation Station



OBS.: Register File
Qi = No. RS

OBS.: Terminologia do Sreboard
Do CDC (ver apêndice A)

Reservation Station

Op: Operação a ser executada na unidade (e.g., + or -)

Vj, Vk: **Valores** dos operandos Fontes

- Store buffers tem campos V, resultados devem ser armazenados

Qj, Qk: Reservation Stations produzirá os operandos correspondentes (valores a serem escritos)

- $Q_j, Q_k = 0 \Rightarrow$ ready
- Store buffers tem somente Q_i para RS producing result

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

A: Mantém informação sobre o end. de memória calculado para load ou store

Register result status (campo Q_i no register file) — Indica para cada registrador a unidade funcional (reservation station) que irá escreve-lo. Em branco se não há instruções pendentes que escreve no registrador.

3 estágios do algoritmo de Tomasulo

1. Issue— pega a instrução na “FP Op Queue”

Se a **reservation station** está livre (não há hazard estrutural), issues instr & envia operandos (**renames registers**)

2. Execute —executa a operação sobre os operandos (EX)

Se os dois operandos estão prontos executa a operação;

Se não, monitora o **Common Data Bus** (espera pelo cálculo do operando, essa espera resolve RAW)

(quando um operando está pronto -> **reservation table**)

3. Write result — termina a execução (WB)

Broadcast via **Common Data Bus** o resultados para todas unidades; marca a **reservation station** como disponível

3 estágios do algoritmo de Tomasulo

- data bus normal: dado + destino (“go to” bus)
- Common data bus: dado + source (“come from” bus)
 - 64 bits de dados + 4 bits para endereço da Functional Unit
 - Escreve se há casamento com a Functional Unit (produz resultado)
 - broadcast

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);

WAW?: (1-6)

WAR?: (5-6)

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)
 - Load ou Store buffers
 - Acesso à memória (somente load)
 - Write Result
 - » Load: envia o valor para o registrador 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	<i>j</i>	<i>k</i>	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:

Latência (que falta) da FU

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

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

Register result status:

Clock
0
Clock cycle

	<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	...	<i>F30</i>
FU									

Exemplo Tomasulo: Ciclo 1

Instruction status:

Instruction	<i>j</i>	<i>k</i>	Issue	Exec	Write	Comp	Result	Busy	Address
LD	F6	34+	R2	1				Load1	Yes 34+R2
LD	F2	45+	R3					Load2	No
MULTD	F0	F2	F4					Load3	No
SUBD	F8	F6	F2						
DIVD	F10	F0	F6						
ADDD	F6	F8	F2						

Reservation Stations:

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

Register result status:

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

Exemplo Tomasulo: Ciclo 2

Instruction status:

Instruction	<i>j</i>	<i>k</i>	Issue	Exec	Write	Comp	Result	Busy	Address
LD	F6	34+	R2	1				Load1	Yes 34+R2
LD	F2	45+	R3	2				Load2	Yes 45+R3
MULTD	F0	F2	F4					Load3	No
SUBD	F8	F6	F2						
DIVD	F10	F0	F6						
ADDD	F6	F8	F2						

Reservation Stations:

Time	Name	Busy	Op	S1 <i>V_j</i>	S2 <i>V_k</i>	RS <i>Q_j</i>	RS <i>Q_k</i>
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	No					
	Mult2	No					

Register result status:

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

Nota: pode haver múltiplos loads pendentes

Exemplo Tomasulo: Ciclo 3

Instruction status:

Instruction	<i>j</i>	<i>k</i>	Issue	Exec Comp	Write Result	Busy	Address
LD	F6	34+	R2	1	3	Load1	Yes 34+R2
LD	F2	45+	R3	2		Load2	Yes 45+R3
MULTD	F0	F2	F4	3		Load3	No
SUBD	F8	F6	F2				
DIVD	F10	F0	F6				
ADDD	F6	F8	F2				

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	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	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Exec Comp</i>	<i>Write Result</i>	Busy	Address	
LD	F6	34+	R2	1	2	4	Load1	No
LD	F2	45+	R3	2	4		Load2	Yes 45+R3
MULTD	F0	F2	F4	3			Load3	No
SUBD	F8	F6	F2	4				
DIVD	F10	F0	F6					
ADDD	F6	F8	F2					

Reservation Stations:

Time	Name	Busy	Op	<i>S1</i> <i>Vj</i>	<i>S2</i> <i>Vk</i>	<i>RS</i> <i>Qj</i>	<i>RS</i> <i>Qk</i>
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	Mult1	Load2		M(A1)	Add1				

- Load2 completa; alguém esperando por Load2?

Exemplo Tomasulo: Ciclo 5

Instruction status:

Instruction	<i>j</i>	<i>k</i>	Issue	Exec Comp	Write Result	Busy	Address
LD	F6	34+	R2	1	3	4	Load1
LD	F2	45+	R3	2	4	5	Load2
MULTD	F0	F2	F4	3			Load3
SUBD	F8	F6	F2	4			
DIVD	F10	F0	F6	5			
ADDD	F6	F8	F2				

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	<i>Exec Write</i>			Busy	Address	
			<i>Issue</i>	<i>Comp</i>	<i>Result</i>			
LD	F6	34+	R2	1	3	4	Load1	No
LD	F2	45+	R3	2	4	5	Load2	No
MULTD	F0	F2	F4	3			Load3	No
SUBD	F8	F6	F2	4				
DIVD	F10	F0	F6	5				
ADDD	F6	F8	F2	6				

Reservation Stations:

Time	Name	Busy	Op	$S1$	$S2$	RS	RS
				V_j	V_k	Q_j	Q_k
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				Mult1	M(A2)	Add2	Add1	Mult2	

- Issue ADDD, dependência de nome em F6?

Exemplo Tomasulo: Ciclo 7

Instruction status:

Instruction	<i>j</i>	<i>k</i>	<i>Exec Write</i>			Busy	Address	
			<i>Issue</i>	<i>Comp</i>	<i>Result</i>			
LD	F6	34+	R2	1	3	4	Load1	No
LD	F2	45+	R3	2	4	5	Load2	No
MULTD	F0	F2	F4	3			Load3	No
SUBD	F8	F6	F2	4	7			
DIVD	F10	F0	F6	5				
ADDD	F6	F8	F2	6				

Reservation Stations:

Time	Name	Busy	Op	<i>S1</i>	<i>S2</i>	<i>RS</i>	<i>RS</i>
				<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>
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	<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	...	<i>F30</i>
7	<i>FU</i>	Mult1	M(A2)		Add2	Add1	Mult2		

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

Exemplo Tomasulo: Ciclo 8

Instruction status:

Instruction	<i>j</i>	<i>k</i>	<i>Exec Write</i>			Busy	Address	
			<i>Issue</i>	<i>Comp</i>	<i>Result</i>			
LD	F6	34+	R2	1	3	4	Load1	No
LD	F2	45+	R3	2	4	5	Load2	No
MULTD	F0	F2	F4	3			Load3	No
SUBD	F8	F6	F2	4	7	8		
DIVD	F10	F0	F6	5				
ADDD	F6	F8	F2	6				

Reservation Stations:

Time	Name	Busy	Op	<i>S1</i>	<i>S2</i>	<i>RS</i>	<i>RS</i>
				<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>
	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	<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	...	<i>F30</i>
8					(M-M)	Mult2			
	Mult1	M(A2)		Add2					

Exemplo Tomasulo: Ciclo 9

Instruction status:

Instruction	<i>j</i>	<i>k</i>	<i>Exec Write</i>			Busy	Address	
			<i>Issue</i>	<i>Comp</i>	<i>Result</i>			
LD	F6	34+	R2	1	3	4	Load1	No
LD	F2	45+	R3	2	4	5	Load2	No
MULTD	F0	F2	F4	3			Load3	No
SUBD	F8	F6	F2	4	7	8		
DIVD	F10	F0	F6	5				
ADDD	F6	F8	F2	6				

Reservation Stations:

Time	Name	Busy	Op	<i>S1</i>	<i>S2</i>	<i>RS</i>	<i>RS</i>
				<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>
	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	<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	...	<i>F30</i>
9	FU								
	Mult1	M(A2)		Add2	(M-M)	Mult2			

Exemplo Tomasulo: Ciclo 10

Instruction status:

Instruction	<i>j</i>	<i>k</i>	<i>Exec Write</i>			Busy	Address	
			<i>Issue</i>	<i>Comp</i>	<i>Result</i>			
LD	F6	34+	R2	1	3	4	Load1	No
LD	F2	45+	R3	2	4	5	Load2	No
MULTD	F0	F2	F4	3			Load3	No
SUBD	F8	F6	F2	4	7	8		
DIVD	F10	F0	F6	5				
ADDD	F6	F8	F2	6	10			

Reservation Stations:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>S1</i>	<i>S2</i>	<i>RS</i>	<i>RS</i>
				<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>
	Add1	No					
0	Add2	Yes	ADDD	M(A2)	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	<i>j</i>	<i>k</i>	Issue	Exec Comp	Write Result	Busy	Address	
LD	F6	34+	R2	1	3	4	Load1	No
LD	F2	45+	R3	2	4	5	Load2	No
MULTD	F0	F2	F4	3			Load3	No
SUBD	F8	F6	F2	4	7	8		
DIVD	F10	F0	F6	5				
ADDD	F6	F8	F2	6	10	11		

Reservation Stations:

Time	Name	Busy	Op	<i>S1</i> <i>Vj</i>	<i>S2</i> <i>Vk</i>	<i>RS</i> <i>Qj</i>	<i>RS</i> <i>Qk</i>
	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	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

Instruction status:

Instruction	<i>j</i>	<i>k</i>	<i>Exec Write</i>			Busy	Address	
			<i>Issue</i>	<i>Comp</i>	<i>Result</i>			
LD	F6	34+	R2	1	3	4	Load1	No
LD	F2	45+	R3	2	4	5	Load2	No
MULTD	F0	F2	F4	3			Load3	No
SUBD	F8	F6	F2	4	7	8		
DIVD	F10	F0	F6	5				
ADDD	F6	F8	F2	6	10	11		

Reservation Stations:

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

Register result status:

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

Exemplo Tomasulo: Ciclo 13

Instruction status:

Instruction	<i>j</i>	<i>k</i>	<i>Exec Write</i>			Busy	Address	
			<i>Issue</i>	<i>Comp</i>	<i>Result</i>			
LD	F6	34+	R2	1	3	4	Load1	No
LD	F2	45+	R3	2	4	5	Load2	No
MULTD	F0	F2	F4	3			Load3	No
SUBD	F8	F6	F2	4	7	8		
DIVD	F10	F0	F6	5				
ADDD	F6	F8	F2	6	10	11		

Reservation Stations:

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

Register result status:

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

Exemplo Tomasulo: Ciclo 14

Instruction status:

Instruction	<i>j</i>	<i>k</i>	<i>Exec Write</i>			Busy	Address	
			<i>Issue</i>	<i>Comp</i>	<i>Result</i>			
LD	F6	34+	R2	1	3	4	Load1	No
LD	F2	45+	R3	2	4	5	Load2	No
MULTD	F0	F2	F4	3			Load3	No
SUBD	F8	F6	F2	4	7	8		
DIVD	F10	F0	F6	5				
ADDD	F6	F8	F2	6	10	11		

Reservation Stations:

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

Register result status:

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

Exemplo Tomasulo: Ciclo 15

Instruction status:

Instruction	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Exec Comp</i>	<i>Write Result</i>	Load	Busy	Address
LD	F6	34+	R2	1	3	4	Load1	No
LD	F2	45+	R3	2	4	5	Load2	No
MULTD	F0	F2	F4	3	15		Load3	No
SUBD	F8	F6	F2	4	7	8		
DIVD	F10	F0	F6	5				
ADDD	F6	F8	F2	6	10	11		

Reservation Stations:

Time	Name	Busy	Op	<i>S1 Vj</i>	<i>S2 Vk</i>	<i>RS Qj</i>	<i>RS Qk</i>
	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	Mult1	M(A2)		(M-M+N)	(M-M)	Mult2			

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

Exemplo Tomasulo: Ciclo 16

Instruction status:

Instruction	<i>j</i>	<i>k</i>	<i>Exec Write</i>			Busy	Address	
			<i>Issue</i>	<i>Comp</i>	<i>Result</i>			
LD	F6	34+	R2	1	3	4	Load1	No
LD	F2	45+	R3	2	4	5	Load2	No
MULTD	F0	F2	F4	3	15	16	Load3	No
SUBD	F8	F6	F2	4	7	8		
DIVD	F10	F0	F6	5				
ADDD	F6	F8	F2	6	10	11		

Reservation Stations:

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

Register result status:

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

- Agora é só esperar que Mult2 (DIVD) complete

**Pulando alguns ciclos
(façam como exercício os ciclos faltantes?)**

Exemplo Tomasulo: Ciclo 55

Instruction status:

Instruction	<i>j</i>	<i>k</i>	<i>Exec Write</i>			Busy	Address	
			<i>Issue</i>	<i>Comp</i>	<i>Result</i>			
LD	F6	34+	R2	1	3	4	Load1	No
LD	F2	45+	R3	2	4	5	Load2	No
MULTD	F0	F2	F4	3	15	16	Load3	No
SUBD	F8	F6	F2	4	7	8		
DIVD	F10	F0	F6	5				
ADDD	F6	F8	F2	6	10	11		

Reservation Stations:

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

Register result status:

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

Exemplo Tomasulo: Ciclo 56

Instruction status:

Instruction	<i>j</i>	<i>k</i>	<i>Exec Write</i>			Busy	Address	
			<i>Issue</i>	<i>Comp</i>	<i>Result</i>			
LD	F6	34+	R2	1	3	4	Load1	No
LD	F2	45+	R3	2	4	5	Load2	No
MULTD	F0	F2	F4	3	15	16	Load3	No
SUBD	F8	F6	F2	4	7	8		
DIVD	F10	F0	F6	5	56			
ADDD	F6	F8	F2	6	10	11		

Reservation Stations:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>S1</i>	<i>S2</i>	<i>RS</i>	<i>RS</i>
				<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>
	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	M*F4	M(A2)		(M-M+N	(M-M)	Mult2			

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

Exemplo Tomasulo: Ciclo 57

Instruction status:

Instruction	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Exec Comp</i>	<i>Write Result</i>	Busy	Address	
LD	F6	34+	R2	1	3	4	Load1	No
LD	F2	45+	R3	2	4	5	Load2	No
MULTD	F0	F2	F4	3	15	16	Load3	No
SUBD	F8	F6	F2	4	7	8		
DIVD	F10	F0	F6	5	56	57		
ADDD	F6	F8	F2	6	10	11		

Reservation Stations:

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

Register result status:

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

- In-order issue, out-of-order execution e out-of-order completion.