

# **Computer Architecture**

Spring 2016

## **Lecture 12: Dynamic Scheduling: Tomasulo's Algorithm**

**Shuai Wang**

**Department of Computer Science and Technology**

**Nanjing University**

[Slides adapted from CS252, UC Berkeley and CS 246, Harvard University]

# Tomasulo's Algorithm

- Used in IBM 360/91 Machines (Late 60s)
- Similar to scoreboarding, but added **renaming in hardware**
- Key concept: **Reservation Stations (RS)**
- Can eliminate WAW and WAR hazards
  
- Very Important Topic
  - Scheduling ideas led to Alpha 21264, HP PA-8000, MIPS R10K, Pentium III, Pentium 4, PowerPC 604, etc...

MEMORANDUM

August 28, 1963

Memorandum To: Messrs. A. L. Williams  
T. V. Learson  
H. W. Miller, Jr.  
E. R. Piore  
O. M. Scott  
M. B. Smith  
A. K. Watson

Last week CDC had a press conference during which they officially announced their 6600 system. I understand that in the laboratory developing this system there are only 34 people, "including the janitor." Of these, 14 are engineers and 4 are programmers, and only one person has a Ph. D., a relatively junior programmer. To the outsider, the laboratory appeared to be cost conscious, hard working and highly motivated.

Contrasting this modest effort with our own vast development activities, I fail to understand why we have lost our industry leadership position by letting someone else offer the world's most powerful computer. At Jenny Lake, I think top priority should be given to a discussion as to what we are doing wrong and how we should go about changing it immediately.

TJW, Jr:jmc

T. J. Watson, Jr.

cc: Mr. W. E. McWhirter

[© IBM]

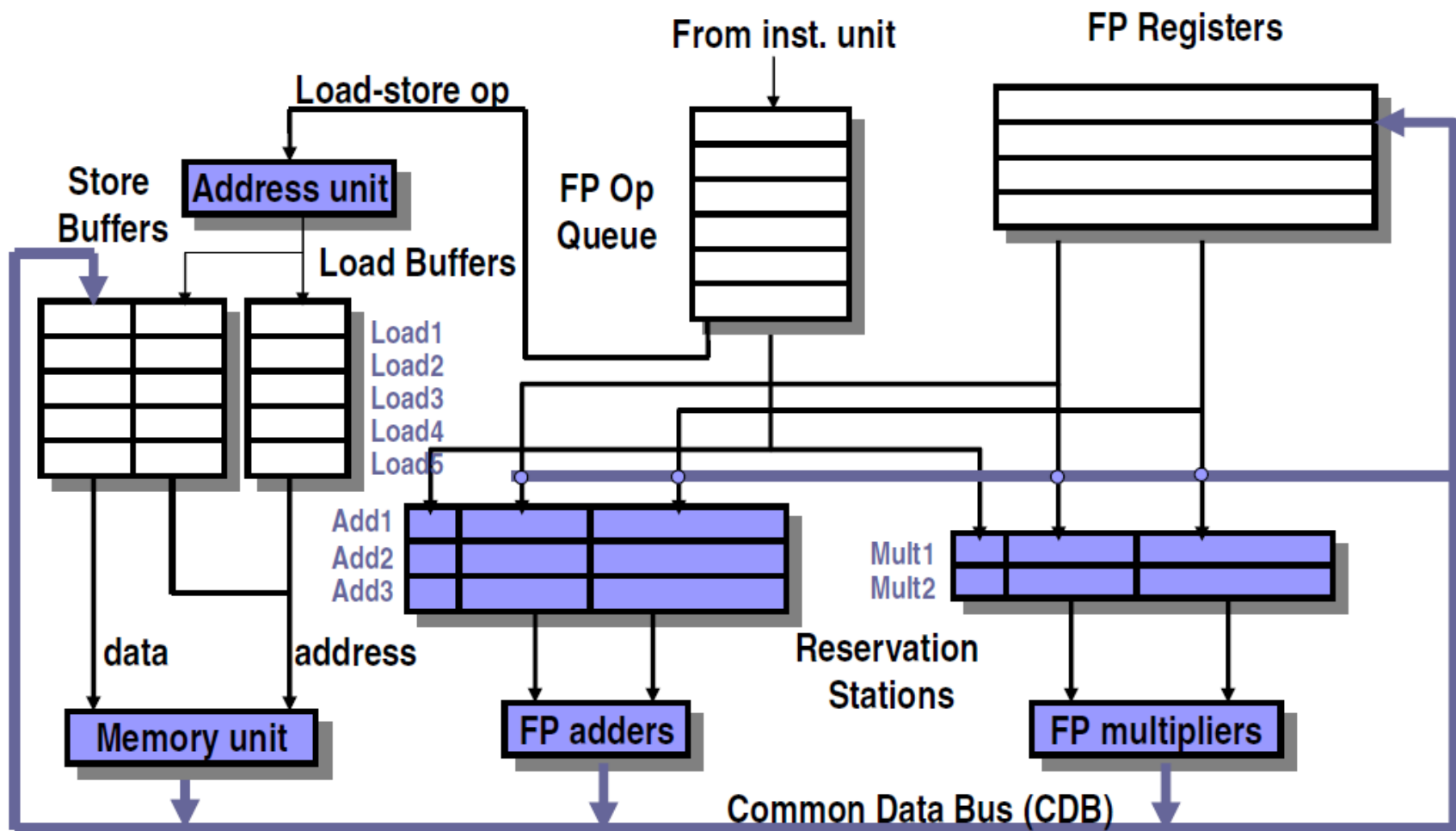
# Tomasulo's Algorithm

- **Distributed** (rather than centralized) control scheme
  - Bypassing is allowed via **Common Data Bus (CDB)** to RS
  - **Register Renaming** eliminates WAR/WAW hazards
- Scoreboard/Instruction Buffer => **Reservation Stations (RS)**
  - Fetch and Buffer **operands** as soon as available
    - Eliminates need to always get values from registers at execute
  - Pending instructions designate reservation stations that will provide their inputs
  - Successive writes to a register cause only the last one to update the register

# Register Renaming with Tomasulo

- At instruction issue:
  - Register specifiers for source operands are renamed to the names of the reservation stations
  - Values can exist in reservation station or register file
    - To eliminate WAR, register file values are *copied* to reservation stations at issue
    - Other methods example use *pointer-based* renaming (map-table)
- Technique used in Pentium III, Pentium M, PowerPC604...

# Tomasulo Organization



# Tomasulo Implementation

- **Reservation station has following fields**
  - **Op**: Operation to perform in the unit
  - **Vj, Vk**: Value of Source operands
    - Store buffers has V field, result to be stored
  - **Qj, Qk**: Reservation stations producing source registers (value to be written)
    - Note:  $Q_j, Q_k = 0 \Rightarrow$  ready
    - Store buffers only have  $Q_i$  for RS producing result
  - **Busy**: Indicates reservation station or FU is busy
- **Register result status**
  - **FU**: Indicates which functional unit will write this register. Blank when no active instructions that will write that register.

# Three Stages of Tomasulo's Algorithm

1. **Issue** - get instruction from FP Op. Queue
  - If reservation station free (no structural hazard), control issues instr & sends operands (renames registers).
2. **Execute** - operate on operands (EX)
  - When both operands ready then execute; if not ready, watch Common Data Bus for result
3. **Write Result** - finish execution (WB)
  - Write on Common Data Bus to all awaiting units; mark reservation station available

# Data Buses in Tomasulo Algorithm

- Normal data bus: data + destination (“go to” bus)
- **Common data bus**: data + source (“come from” bus)
  - 64 bits of data + 4 bits of Functional Unit **source** address
  - Write if matches expected Functional Unit (produces result)
  - Does the broadcast



# Tomasulo Example: Cycle 1

## Instruction status:

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

## 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					

# Tomasulo Example: Cycle 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<sub>j</sub></i>	S2 <i>V<sub>k</sub></i>	RS <i>Q<sub>j</sub></i>	RS <i>Q<sub>k</sub></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					

# Tomasulo Example: Cycle 3

## Instruction status:

Instruction	<i>j</i>	<i>k</i>	Issue	Exec	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	<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	Yes	MULTD		R(F4)	Load2	
	Mult2	No					

## Register result status:

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

Note: registers names are removed (“renamed”) in Reservation Stations; MULT issued vs. scoreboard

Load 1 is complete! What is waiting for it?

# Tomasulo Example: Cycle 4

## Instruction status:

Instruction	<i>j</i>	<i>k</i>	Issue	Exec	Write	Busy	Address
				Comp	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	<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	FU	Mult1	Load2		M(A1)	Add1			

**Load 2 is complete! What is waiting for it?**

# Tomasulo Example: Cycle 5

## Instruction status:

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

## 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>
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	<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	...	<i>F30</i>
5	Mult1	M(A2)		M(A1)	Add1	Mult2			

# Tomasulo Example: Cycle 6

## Instruction status:

Instruction	j	k	Exec Write			Busy	Address	
			Issue	Comp	Result			
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
				Vj	Vk	Qj	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	Mult1	M(A2)		Add2	Add1	Mult2			

Issue ADD here vs. scoreboard?

# Tomasulo Example: Cycle 7

## Instruction status:

Instruction	<i>j</i>	<i>k</i>	Issue	Exec	Write	Result	Load	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			
DIVD	F10	F0	F6	5				
ADDD	F6	F8	F2	6				

## 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>
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	Mult1	M(A2)		Add2	Add1	Mult2			

**Add1 completing; what is waiting for it?**

# Tomasulo Example: Cycle 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>V<sub>j</sub></i>	<i>V<sub>k</sub></i>	<i>Q<sub>j</sub></i>	<i>Q<sub>k</sub></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	Mult1	M(A2)		Add2	(M-M)	Mult2			

# Tomasulo Example: Cycle 9

## Instruction status:

Instruction	<i>j</i>	<i>k</i>	<i>Rj</i>	<i>Rk</i>	Exec Write		Busy	Address	
					Issue	Comp Result			
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	F0	F2	F4	F6	F8	F10	F12	...	F30
9	Mult1	M(A2)		Add2	(M-M)	Mult2			

# Tomasulo Example: Cycle 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:

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					
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	<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	...	<i>F30</i>
10	<i>FU</i>	Mult1	M(A2)		Add2	(M-M)	Mult2		

Add2 completing; what is waiting for it?

# Tomasulo Example: Cycle 11

*Instruction status:*

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

**Write result of ADDD here vs. scoreboard?**

**All quick instructions complete in this cycle!**

# Tomasulo Example: Cycle 12

## Instruction status:

Instruction	$j$	$k$	Exec Write			Busy	Address	
			Issue	Comp	Result			
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	$S1$	$S2$	$RS$	$RS$
				$V_j$	$V_k$	$Q_j$	$Q_k$
	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	Mult1	M(A2)		(M-M+M)	(M-M)	Mult2			

# Tomasulo Example: Cycle 13

## Instruction status:

Instruction	$j$	$k$	Issue	Exec Write		Busy	Address
				Comp	Result		
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	7	8	
DIVD	F10	F0	F6	5			
ADDD	F6	F8	F2	6	10	11	

## Reservation Stations:

Time	Name	Busy	Op	$S1$	$S2$	$RS$	$RS$
				$V_j$	$V_k$	$Q_j$	$Q_k$
	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	Mult1	M(A2)		(M-M+N)	(M-M)	Mult2			

# Tomasulo Example: Cycle 14

## Instruction status:

Instruction	j	k	Exec Write			Busy	Address	
			Issue	Comp	Result			
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	S1	S2	RS	RS
				Vj	Vk	Qj	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	Mult1	M(A2)		(M-M+M)	(M-M)	Mult2			

# Tomasulo Example: Cycle 15

## 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		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>V<sub>j</sub></i>	<i>V<sub>k</sub></i>	<i>Q<sub>j</sub></i>	<i>Q<sub>k</sub></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			

# Tomasulo Example: Cycle 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	F0	F2	F4	F6	F8	F10	F12	...	F30
16	M*F4	M(A2)		(M-M+N.(M-M)		Mult2			

# Tomasulo Example: Cycle 55 (Way Later!)

## 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:

<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>V<sub>j</sub></i>	<i>V<sub>k</sub></i>	<i>Q<sub>j</sub></i>	<i>Q<sub>k</sub></i>
	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	M*F4	M(A2)		(M-M+M)	(M-M)	Mult2			

# Tomasulo Example: Cycle 56

## Instruction status:

Instruction	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Exec</i>	<i>Write</i>	<i>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			
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>V<sub>j</sub></i>	<i>V<sub>k</sub></i>	<i>Q<sub>j</sub></i>	<i>Q<sub>k</sub></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+M)	(M-M)	Mult2			

**Mult2 is completing; what is waiting for it?**

# Tomasulo Example: Cycle 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	No					

## Register result status:

Clock	F0	F2	F4	F6	F8	F10	F12	...	F30
57	FU								

Once again: In-order issue, out-of-order execution and completion.

# Compare to Scoreboard: Cycle 62

*Instruction status:*

Instruction	<i>j</i>	<i>k</i>	<i>Read</i>	<i>Exec</i>	<i>Write</i>		<i>Exec</i>	<i>Write</i>		
			<i>Issue</i>	<i>Oper</i>	<i>Comp</i>	<i>Result</i>	<i>Issue</i>	<i>Comp</i>	<i>Result</i>	
LD	F6	34+	R2	1	2	3	4	1	3	4
LD	F2	45+	R3	5	6	7	8	2	4	5
MULTD	F0	F2	F4	6	9	19	20	3	15	16
SUBD	F8	F6	F2	7	9	11	12	4	7	8
DIVD	F10	F0	F6	8	21	61	62	5	56	57
ADDD	F6	F8	F2	13	14	16	22	6	10	11

**Why take longer on scoreboard/6600**

- Structural Hazards
- Lack of forwarding

# Advantages of Tomasulo

- The distribution of the hazard detection logic
  - distributed reservation stations and the CDB
  - If multiple instructions waiting on single result, & each instruction has other operand, then instructions can be released simultaneously by broadcast on CDB
  - If a centralized register file were used, the units would have to read their results from the registers when register buses are available.
- The elimination of stalls for WAW and WAR hazards

# Tomasulo vs. Scoreboarding

- No explicit checking for WAW or WAR hazards
  - Distribute RAW hazard detection
  - Renaming eliminates WAW hazards
  - Buffering values in Reservation Stations removes WAR hazards
- CDB broadcasts results rather than waiting on registers
- Loads/Store are treated like basic FUs
- Distributed vs. Centralized control

# Tomasulo Drawbacks

- Performance limited by Common Data Bus
  - Tag match in CDB requires many associative compares
  - Each CDB must go to multiple functional units
    - =>high capacitance, high wiring density
  - Number of functional units that can complete per cycle limited to one!
    - Multiple CDBs => more FU logic for parallel assoc stores
- Need Load/Store reordering
  - Load checks A field of all active stores
  - Store checks A field of earlier loads and stores
- Non-precise exceptions!
  - We will address in later lectures