



# CS305: Computer Architecture

## Dynamic Instruction Scheduling

<https://www.cse.iitb.ac.in/~biswa/courses/CS305/main.html>

<https://www.cse.iitb.ac.in/~biswa/>

# Recap

---

Remember out-of-order processor

---

Inorder fetch, out-of-order execute, inorder commit

---

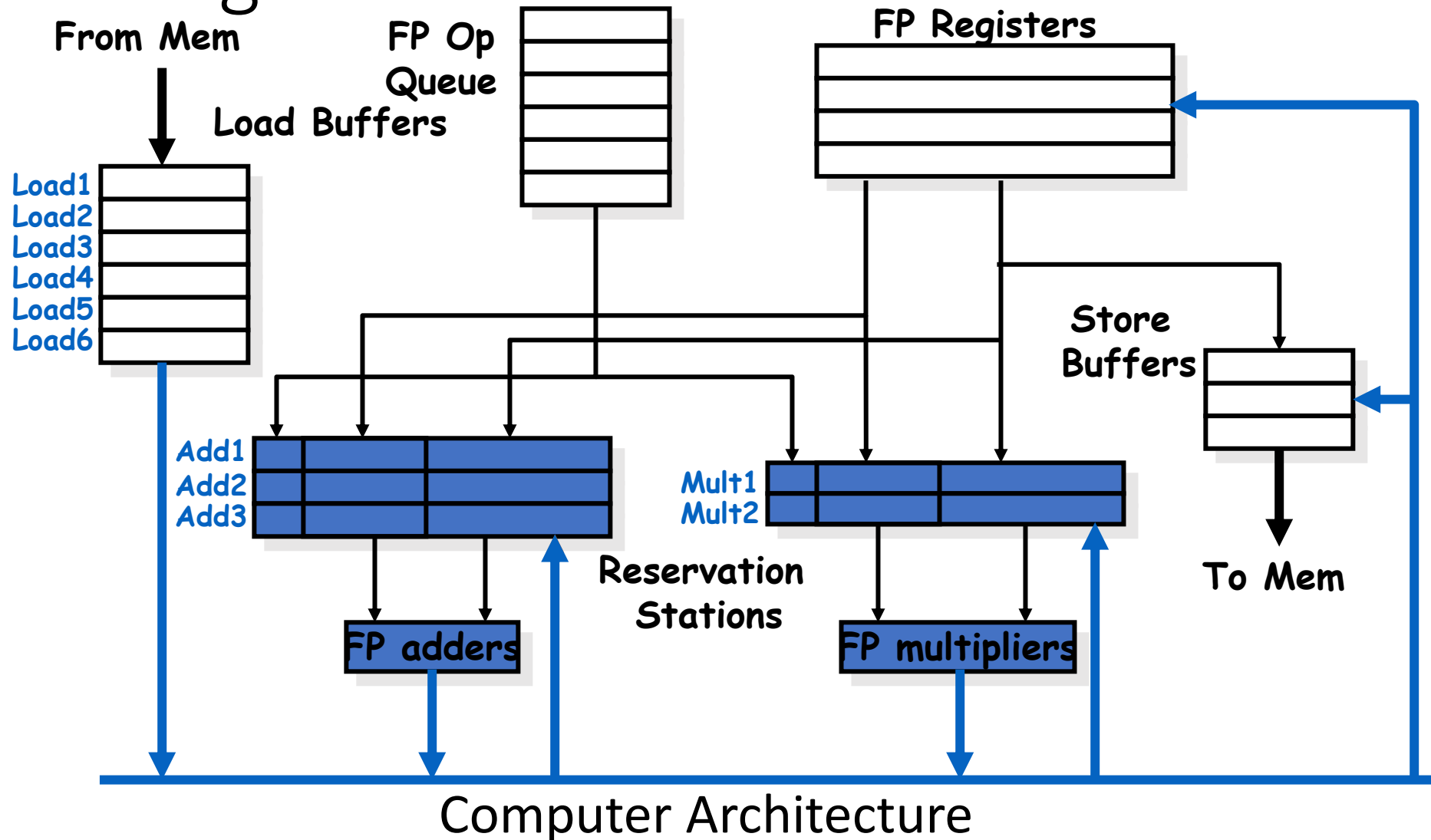
Now, we will discuss in details

---

Static scheduling: Compiler can do; loop unrolling etc.

---

# Tomasulo's Organization for dynamic scheduling



# Points to remember

- Control & buffers distributed with Function Units (FU)
  - FU buffers called “reservation stations”; have pending operands
- Registers in instructions replaced by values or pointers to reservation stations(RS); called register renaming ;
  - avoids WAR, WAW hazards
  - More reservation stations than registers, so can do optimizations compilers can't
- Results to FU from RS over Common Data Bus that broadcasts results to all FUs
- Load and Stores treated as FUs with RSs as wells
- Decode stage of the pipeline: becomes two stages:

**Issue:** Decode instructions, check structural hazards

**Read operands:** Wait until no data hazards, then read operands.

Some processors use the term **dispatch** and **issue**.

# Reservation Station Components

**Op:** Operation to perform in the unit (e.g., + or −)

**V<sub>j</sub>, V<sub>k</sub>:** Value of Source operands

- Store buffers has V field, result to be stored

**Q<sub>j</sub>, Q<sub>k</sub>:** Reservation stations producing source registers (value to be written)

- Q<sub>j</sub>, Q<sub>k</sub>=0 => ready
- Store buffers only have Q<sub>i</sub> for RS producing result

**Busy:** Indicates reservation station or FU is busy

**Register result status**—Indicates which functional unit will write each register, if one exists. Blank when no pending instructions that will write that register.

# Three Stages with Tomasulo

## 1. Issue—get instruction from FP Op/instruction Queue

If reservation station free (no structural hazard), control issues instr & sends operands (renames registers).

## 2. Execution—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

- 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

# The New Pipeline

**Inorder** Instruction fetch

Fetches instructions and enqueues them into a Q called **Instruction Q (IQ)**.

**Inorder** instruction issue from the IQ

**Outoforder** execution

New concept of **register renaming** through reservation stations that eliminates WAR and WAW hazards

# An Example

## Instruction status:

Instruction	$j$	$k$	$Issue$	$Exec$	$Write$
				$Comp$	$Result$
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	

Think about memory hierarchy 😊

## Reservation Stations:

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

## Register result status:

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



# Cycle 1

## Instruction status:

Instruction	$j$	$k$	Issue	Exec	Write
			Comp	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	

Load: 2 cycle  
 FP add: 2 cycles  
 FP multiply: 10 cycles  
 FP divide: 40 cycles

## 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				Load1					

# Cycle 2

## Instruction status:

Instruction	<i>j</i>	<i>k</i>	Issue	Exec	Write	Busy	Address
LD	F6	34+	R2	1		Yes	34+R2
LD	F2	45+	R3	2		Yes	45+R3
MULTD	F0	F2	F4			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
2		Load2		Load1					

can have multiple loads outstanding

# Cycle 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	<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

# Cycle 4

## 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		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> 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>
	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 completing; what is waiting for Load2?

# Cycle 5

## Instruction status:

Instruction	$j$	$k$	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				
DIVD	F10	F0	F6	5				
ADDD	F6	F8	F2					

## Reservation Stations:

Time	Name	Busy	Op	$S1$ $V_j$	$S2$ $V_k$	$RS$ $Q_j$	$RS$ $Q_k$
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	Mult1	M(A2)		M(A1)	Add1	Mult2			

# Cycle 6

## 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			Load3	No
SUBD	F8	F6	F2	4				
DIVD	F10	F0	F6	5				
ADDD	F6	F8	F2	6				

## Reservation Stations:

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

- Issue ADDD here

# Cycle 7

## 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			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	<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	FU	Mult1	M(A2)	Add2	Add1	Mult2			

- Add1 completing; what is waiting for it?

# Cycle 8

## 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			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>Vj</i>	<i>S2</i> <i>Vk</i>	<i>RS</i> <i>Qj</i>	<i>RS</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	Mult1	M(A2)		Add2	(M-M)	Mult2			



# Cycle 9

## 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			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>Vj</i>	<i>S2</i> <i>Vk</i>	<i>RS</i> <i>Qj</i>	<i>RS</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	Mult1	M(A2)		Add2	(M-M)	Mult2			

# 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:

<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-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?

# Cycle 11

## 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	<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					
4	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>
11	Mult1	M(A2)		(M-M+M)	(M-M)	Mult2			

- Write result of ADDD here.
- All quick instructions complete in this cycle!

# Cycle 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	FU	Mult1	M(A2)		(M-M+N)	(M-M)	Mult2		

# Cycle 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	<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					
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			

# Cycle 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	<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					
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	Mult1	M(A2)		(M-M+N)	(M-M)	Mult2			

# Cycle 15

## 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		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					
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			

# Cycle 16

## 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				
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					
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+N	(M-M)	Mult2		



# Cycle 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:

<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					
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			

# Cycle 56

## 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			
ADDD	F6	F8	F2	6	10	11		

## Reservation Stations:

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

- Mult2 is completing; what is waiting for it?

# Cycle 57

## 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	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
56	FU	M*F4	M(A2)		(M-M+M	(M-M)	Result		

- Once again: In-order issue, out-of-order execution and completion. Please note we have not introduced branch prediction, speculation, exception into this scheduling yet. So out of order completion is ok so far.

# Register Renaming

- Tomasulo provides *Implicit Register Renaming*
  - User registers renamed to reservation station tags
- Explicit Register Renaming:
  - Use *physical* register file that is larger than number of registers specified by ISA
- Keep a translation table:
  - ISA register => physical register mapping
  - Physical register becomes free when not being used by any instructions in progress. More later after ROB.

# Explicit Register Renaming

- Rapid access to a table of translations
- A physical register file that has more registers than specified by the ISA
- Ability to figure out which physical registers are free.
  - No free registers  $\Rightarrow$  stall on issue
- Thus, register renaming doesn't require reservation stations.  
However:
  - Many modern architectures use explicit register renaming + Tomasulo-like reservation stations to control execution.

ευχαριστώ