

# Lecture-9 (Dynamic Scheduling)

## CS422-Spring 2020

---

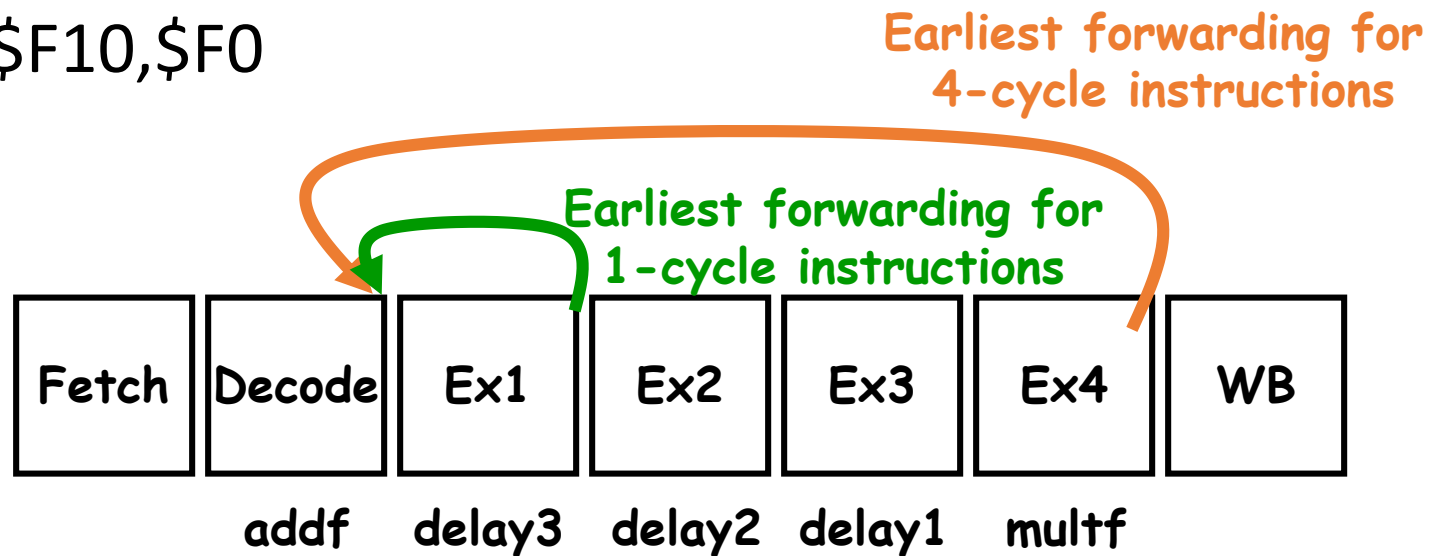
**Biswa@cse-IITK**



# How to Make CPI closer to One

- Let's assume full pipelining:
  - If we have a 4-cycle *latency*, then we need 3 instructions between a producing instruction and its use:

```
multf $F0,$F2,$F4  
delay-1  
delay-2  
delay-3  
addf $F6,$F10,$F0
```



# Out-of-order + Dynamic Scheduling ?

- Pipelining: Tries to achieve  $CPI = 1$
- Compiler scheduling minimizes the impacts of dependences.
- Hardware scheduling so far: In order execution  
Instructions after stall must wait even if independent.

Dynamic scheduling: Out of order execution

Hardware lookahead of blocked instructions

- Inorder, O3
- Inorder issue, O3 execute, Inorder completion

# Scoreboard

- Out-of-order execution divides ID stage:
  1. **Issue** - decode instructions, check for structural hazards
  2. **Read operands** - wait until no data hazards, then read operands (RAW)

And then

  2. **Execute** - Execute instruction and notify scoreboard when done
  3. **Write** - Wait until earlier instructions read operands before writing to register file (WAR)
- Scoreboards date to CDC6600 in 1963
- Instructions execute whenever not dependent on previous instructions and no hazards.

# Four Stages of Scoreboard Control - Details

- **Issue**—decode instructions & check for structural hazards (ID1)
  - Instructions issued in program order (for hazard checking)
  - Don't issue if **structural hazard**
  - Don't issue if instruction is **output dependent** on any previously issued but uncompleted instruction (no WAW hazards)
- **Read operands**—wait until no data hazards, then read operands (ID2)
  - All real dependencies (RAW hazards) resolved in this stage, since we wait for instructions to write back data.
  - **No forwarding of data** in this model!

# Four Stages of Scoreboard Control

- **Execution**—operate on operands (EX)
  - The functional unit begins execution upon receiving operands. When the result is ready, it notifies the scoreboard that it has completed execution.
- **Write result**—finish execution (WB)
  - Stall until no WAR hazards with previous instructions:

Example:

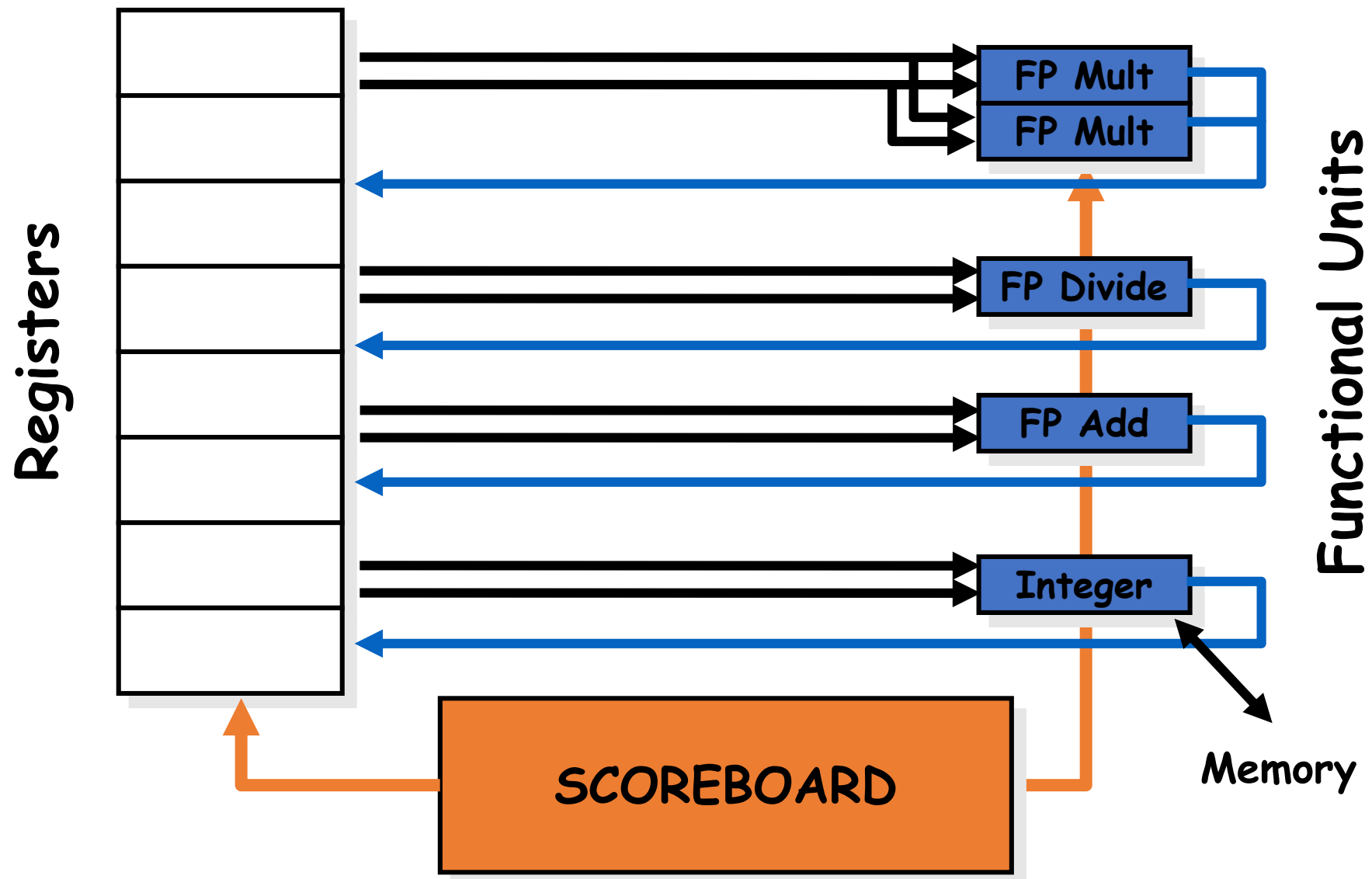
	DIVD	F0,F2,F4
	ADDD	F10,F0,F8
	SUBD	F8,F8,F14

CDC 6600 scoreboard would stall SUBD until ADDD reads operands

# Three Parts of the Scoreboard

- **Instruction status:**  
Which of 4 steps the instruction is in
- **Functional unit status:**—Indicates the state of the functional unit (FU). 9 fields for each functional unit
  - Busy:** Indicates whether the unit is busy or not
  - Op:** Operation to perform in the unit (e.g., + or -)
  - Fi:** Destination register
  - Fj,Fk:** Source-register numbers
  - Qj,Qk:** Functional units producing source registers Fj, Fk
  - Rj,Rk:** Flags indicating when Fj, Fk are ready
- **Register result status**—Indicates which functional unit will write each register, if one exists. Blank when no pending instructions will write that register

# Possible Architecture







# Cycle 1

## Instruction status:

Instruction	$j$	$k$	Issue	Read Oper	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			

## Functional unit status:

Time	Name	Busy	Op	dest $F_i$	$S1$ $F_j$	$S2$ $F_k$	$FU$ $Q_j$	$FU$ $Q_k$	$F_j?$ $R_j$	$F_k?$ $R_k$
	Integer	Yes	Load	F6		R2				Yes
	Mult1	No								
	Mult2	No								
	Add	No								
	Divide	No								

## Register result status:

Clock	$F0$	$F2$	$F4$	$F6$	$F8$	$F10$	$F12$	...	$F30$
1				Integer					

# Cycle 2

## Instruction status:

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

## Functional unit status:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>dest Fi</i>	<i>S1 Fj</i>	<i>S2 Fk</i>	<i>FU Qj</i>	<i>FU Qk</i>	<i>Fj? Rj</i>	<i>Fk? Rk</i>
	Integer	Yes	Load	F6		R2				Yes
	Mult1	No								
	Mult2	No								
	Add	No								
	Divide	No								

## 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>
2	<i>FU</i> Integer								

• Issue 2nd LD?

Can't since integer unit is busy.

# Cycle 3

## Instruction status:

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

## Functional unit status:

Time	Name	Busy	Op	dest <i>Fi</i>	<i>S1</i> <i>Fj</i>	<i>S2</i> <i>Fk</i>	<i>FU</i> <i>Qj</i>	<i>FU</i> <i>Qk</i>	<i>Fj?</i> <i>Rj</i>	<i>Fk?</i> <i>Rk</i>
	Integer	Yes	Load	F6		R2				No
	Mult1	No								
	Mult2	No								
	Add	No								
	Divide	No								

## 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>
3	<i>FU</i> Integer								

• Issue MULT?

• F2?

# Cycle 4

## Instruction status:

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

## Functional unit status:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>dest</i> <i>Fi</i>	<i>S1</i> <i>Fj</i>	<i>S2</i> <i>Fk</i>	<i>FU</i> <i>Qj</i>	<i>FU</i> <i>Qk</i>	<i>Fj?</i> <i>Rj</i>	<i>Fk?</i> <i>Rk</i>
	Integer	No								
	Mult1	No								
	Mult2	No								
	Add	No								
	Divide	No								

## 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>
4	FU Integer								

# Cycle 5

## Instruction status:

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

## Functional unit status:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>dest</i> <i>Fi</i>	<i>S1</i> <i>Fj</i>	<i>S2</i> <i>Fk</i>	<i>FU</i> <i>Qj</i>	<i>FU</i> <i>Qk</i>	<i>Fj?</i> <i>Rj</i>	<i>Fk?</i> <i>Rk</i>
	Integer	Yes	Load	F2		R3				Yes
	Mult1	No								
	Mult2	No								
	Add	No								
	Divide	No								

## 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	<i>FU</i>	Integer							

# Cycle 6

## Instruction status:

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

## Functional unit status:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>dest Fi</i>	<i>S1 Fj</i>	<i>S2 Fk</i>	<i>FU Qj</i>	<i>FU Qk</i>	<i>Fj? Rj</i>	<i>Fk? Rk</i>
	Integer	Yes	Load	F2		R3				Yes
	Mult1	Yes	Mult	F0	F2	F4	Integer		No	Yes
	Mult2	No								
	Add	No								
	Divide	No								

## 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	Integer							

# Cycle 7

## Instruction status:

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

## Functional unit status:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>dest Fi</i>	<i>S1 Fj</i>	<i>S2 Fk</i>	<i>FU Qj</i>	<i>FU Qk</i>	<i>Fj? Rj</i>	<i>Fk? Rk</i>
Integer		Yes	Load	F2		R3				No
Mult1		Yes	Mult	F0	F2	F4	Integer		No	Yes
Mult2		No								
Add		Yes	Sub	F8	F6	F2	Integer		Yes	No
Divide		No								

## 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	Integer			Add				

• Read multiply operands?

• LOAD is not done yet ☹️



# Cycle 8 (1<sup>st</sup> half)

## Instruction status:

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

## Functional unit status:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>dest Fi</i>	<i>S1 Fj</i>	<i>S2 Fk</i>	<i>FU Qj</i>	<i>FU Qk</i>	<i>Fj? Rj</i>	<i>Fk? Rk</i>
	Integer	Yes	Load	F2		R3				No
	Mult1	Yes	Mult	F0	F2	F4	Integer		No	Yes
	Mult2	No								
	Add	Yes	Sub	F8	F6	F2		Integer	Yes	No
	Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

## 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	Integer			Add	Divide			

**DIVD issues. MULT and SUBD. Both waiting for F2. LD #2 writes F2.**

# Cycle 8 (2<sup>nd</sup> Half)

## Instruction status:

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

## Functional unit status:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>dest Fi</i>	<i>S1 Fj</i>	<i>S2 Fk</i>	<i>FU Qj</i>	<i>FU Qk</i>	<i>Fj? Rj</i>	<i>Fk? Rk</i>
	Integer	No								
	Mult1	Yes	Mult	F0	F2	F4			Yes	Yes
	Mult2	No								
	Add	Yes	Sub	F8	F6	F2			Yes	Yes
	Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

## 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	<i>FU</i> Mult1 Add Divide								

# Cycle 9

## Instruction status:

Instruction	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Read Oper</i>	<i>Exec Comp</i>	<i>Write Result</i>
LD	F6	34+	R2	1	2	3 4
LD	F2	45+	R3	5	6	7 8
MULTD	F0	F2	F4	6	9	
SUBD	F8	F6	F2	7	9	
DIVD	F10	F0	F6	8		
ADDD	F6	F8	F2			

## Functional unit status:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>dest Fi</i>	<i>S1 Fj</i>	<i>S2 Fk</i>	<i>FU Qj</i>	<i>FU Qk</i>	<i>Fj? Rj</i>	<i>Fk? Rk</i>
	Integer	No								
10	Mult1	Yes	Mult	F0	F2	F4			Yes	Yes
	Mult2	No								
2	Add	Yes	Sub	F8	F6	F2			Yes	Yes
	Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

Note →  
Remaining

## 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	<i>FU</i> Mult1				Add	Divide			

- Read operands for MULT & SUB? Issue ADDD?

# Cycle 10

## Instruction status:

Instruction	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Read Oper</i>	<i>Exec Comp</i>	<i>Write Result</i>
LD	F6	34+ R2	1	2	3	4
LD	F2	45+ R3	5	6	7	8
MULTD	F0	F2 F4	6	9		
SUBD	F8	F6 F2	7	9		
DIVD	F10	F0 F6	8			
ADDD	F6	F8 F2				

## Functional unit status:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>dest Fi</i>	<i>S1 Fj</i>	<i>S2 Fk</i>	<i>FU Qj</i>	<i>FU Qk</i>	<i>Fj? Rj</i>	<i>Fk? Rk</i>
	Integer	No								
9	Mult1	Yes	Mult	F0	F2	F4			No	No
	Mult2	No								
1	Add	Yes	Sub	F8	F6	F2			No	No
	Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

## 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	FU Mult1 Add Divide								

# Cycle 11

## Instruction status:

Instruction	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Read Oper</i>	<i>Exec Comp</i>	<i>Write Result</i>
LD	F6	34+ R2	1	2	3	4
LD	F2	45+ R3	5	6	7	8
MULTD	F0	F2 F4	6	9		
SUBD	F8	F6 F2	7	9	11	
DIVD	F10	F0 F6	8			
ADDD	F6	F8 F2				

ADDD can't start because add unit is busy

## Functional unit status:

Time	Name	Busy	Op	dest <i>Fi</i>	<i>S1</i> <i>Fj</i>	<i>S2</i> <i>Fk</i>	<i>FU</i> <i>Qj</i>	<i>FU</i> <i>Qk</i>	<i>Fj?</i> <i>Rj</i>	<i>Fk?</i> <i>Rk</i>
	Integer	No								
8	Mult1	Yes	Mult	F0	F2	F4			No	No
	Mult2	No								
0	Add	Yes	Sub	F8	F6	F2			No	No
	Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

## 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	FU Mult1 Add Divide								

# Cycle 12

## Instruction status:

Instruction	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Read Oper</i>	<i>Exec Comp</i>	<i>Write Result</i>	
LD	F6	34+	R2	1	2	3	4
LD	F2	45+	R3	5	6	7	8
MULTD	F0	F2	F4	6	9		
SUBD	F8	F6	F2	7	9	11	12
DIVD	F10	F0	F6	8			
ADDD	F6	F8	F2				

## Functional unit status:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>dest</i> <i>Fi</i>	<i>S1</i> <i>Fj</i>	<i>S2</i> <i>Fk</i>	<i>FU</i> <i>Qj</i>	<i>FU</i> <i>Qk</i>	<i>Fj?</i> <i>Rj</i>	<i>Fk?</i> <i>Rk</i>
	Integer	No								
7	Mult1	Yes	Mult	F0	F2	F4			No	No
	Mult2	No								
	Add	No								
	Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

## 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					Divide			

- Read operands for DIVD?

# Cycle 13

## Instruction status:

Instruction	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Read Oper</i>	<i>Exec Comp</i>	<i>Write Result</i>
LD	F6	34+ R2	1	2	3	4
LD	F2	45+ R3	5	6	7	8
MULTD	F0	F2 F4	6	9		
SUBD	F8	F6 F2	7	9	11	12
DIVD	F10	F0 F6	8			
ADDD	F6	F8 F2	13			

## Functional unit status:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>dest Fi</i>	<i>S1 Fj</i>	<i>S2 Fk</i>	<i>FU Qj</i>	<i>FU Qk</i>	<i>Fj? Rj</i>	<i>Fk? Rk</i>
	Integer	No								
6	Mult1	Yes	Mult	F0	F2	F4			No	No
	Mult2	No								
	Add	Yes	Add	F6	F8	F2			Yes	Yes
	Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

## 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			Add		Divide			

# Cycle 14

## Instruction status:

Instruction	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Read Oper</i>	<i>Exec Comp</i>	<i>Write Result</i>
LD	F6	34+	R2	1	2	3 4
LD	F2	45+	R3	5	6	7 8
MULTD	F0	F2	F4	6	9	
SUBD	F8	F6	F2	7	9	11 12
DIVD	F10	F0	F6	8		
ADDD	F6	F8	F2	13	14	

## Functional unit status:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>dest Fi</i>	<i>S1 Fj</i>	<i>S2 Fk</i>	<i>FU Qj</i>	<i>FU Qk</i>	<i>Fj? Rj</i>	<i>Fk? Rk</i>
	Integer	No								
5	Mult1	Yes	Mult	F0	F2	F4			No	No
	Mult2	No								
2	Add	Yes	Add	F6	F8	F2			Yes	Yes
	Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

## 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			Add		Divide			



# Cycle 15

## Instruction status:

Instruction	<i>j</i>	<i>k</i>	Issue	Read Oper	Exec Comp	Write Result
LD	F6	34+	R2	1	2	3 4
LD	F2	45+	R3	5	6	7 8
MULTD	F0	F2	F4	6	9	
SUBD	F8	F6	F2	7	9	11 12
DIVD	F10	F0	F6	8		
ADDD	F6	F8	F2	13	14	

## Functional unit status:

Time	Name	Busy	Op	dest <i>Fi</i>	<i>S1</i> <i>Fj</i>	<i>S2</i> <i>Fk</i>	<i>FU</i> <i>Qj</i>	<i>FU</i> <i>Qk</i>	<i>Fj?</i> <i>Rj</i>	<i>Fk?</i> <i>Rk</i>
	Integer	No								
4	Mult1	Yes	Mult	F0	F2	F4			No	No
	Mult2	No								
1	Add	Yes	Add	F6	F8	F2			No	No
	Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

## 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>
15	Mult1			Add		Divide			

# Cycle 16

## Instruction status:

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

## Functional unit status:

<i>Time Name</i>	<i>Busy</i>	<i>Op</i>	<i>dest Fi</i>	<i>S1 Fj</i>	<i>S2 Fk</i>	<i>FU Qj</i>	<i>FU Qk</i>	<i>Fj? Rj</i>	<i>Fk? Rk</i>
Integer	No								
3 Mult1	Yes	Mult	F0	F2	F4			No	No
Mult2	No								
0 Add	Yes	Add	F6	F8	F2			No	No
Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

## 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 Mult1		Add		Divide				

# Cycle 17

## Instruction status:

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

WAR Hazard!

## Functional unit status:

Time	Name	Busy	Op	dest <i>Fi</i>	<i>S1</i> <i>Fj</i>	<i>S2</i> <i>Fk</i>	<i>FU</i> <i>Qj</i>	<i>FU</i> <i>Qk</i>	<i>Fj?</i> <i>Rj</i>	<i>Fk?</i> <i>Rk</i>
	Integer	No								
2	Mult1	Yes	Mult	F0	F2	F4			No	No
	Mult2	No								
	Add	Yes	Add	F6	F8	F2			No	No
	Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

## 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>
17	FU Mult1		Add		Divide				

- Why not write result of ADD???

# Cycle 18

## Instruction status:

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

## Functional unit status:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>dest Fi</i>	<i>S1 Fj</i>	<i>S2 Fk</i>	<i>FU Qj</i>	<i>FU Qk</i>	<i>Fj? Rj</i>	<i>Fk? Rk</i>
	Integer	No								
1	Mult1	Yes	Mult	F0	F2	F4			No	No
	Mult2	No								
	Add	Yes	Add	F6	F8	F2			No	No
	Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

## 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>
18	Mult1			Add		Divide			

# Cycle 19

## Instruction status:

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

## Functional unit status:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>dest Fi</i>	<i>S1 Fj</i>	<i>S2 Fk</i>	<i>FU Qj</i>	<i>FU Qk</i>	<i>Fj? Rj</i>	<i>Fk? Rk</i>
	Integer	No								
0	Mult1	Yes	Mult	F0	F2	F4			No	No
	Mult2	No								
	Add	Yes	Add	F6	F8	F2			No	No
	Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

## 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>
19	Mult1			Add		Divide			

# Cycle 20

## Instruction status:

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

## Functional unit status:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>dest Fi</i>	<i>S1 Fj</i>	<i>S2 Fk</i>	<i>FU Qj</i>	<i>FU Qk</i>	<i>Fj? Rj</i>	<i>Fk? Rk</i>
	Integer	No								
	Mult1	No								
	Mult2	No								
	Add	Yes	Add	F6	F8	F2			No	No
	Divide	Yes	Div	F10	F0	F6			Yes	Yes

## 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>
20				Add	Divide				

# Cycle 21

## Instruction status:

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

## Functional unit status:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>dest</i> <i>Fi</i>	<i>S1</i> <i>Fj</i>	<i>S2</i> <i>Fk</i>	<i>FU</i> <i>Qj</i>	<i>FU</i> <i>Qk</i>	<i>Fj?</i> <i>Rj</i>	<i>Fk?</i> <i>Rk</i>
	Integer	No								
	Mult1	No								
	Mult2	No								
	Add	Yes	Add	F6	F8	F2			No	No
	Divide	Yes	Div	F10	F0	F6			Yes	Yes

## 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>
21				Add		Divide			

- WAR Hazard is now gone...

# Cycle 22

## Instruction status:

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

## Functional unit status:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>dest</i>	<i>S1</i>	<i>S2</i>	<i>FU</i>	<i>FU</i>	<i>Fj?</i>	<i>Fk?</i>
				<i>Fi</i>	<i>Fj</i>	<i>Fk</i>	<i>Qj</i>	<i>Qk</i>	<i>Rj</i>	<i>Rk</i>
	Integer	No								
	Mult1	No								
	Mult2	No								
	Add	No								
39	Divide	Yes	Div	F10	F0	F6			No	No

## 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>
22	<i>FU</i> Divide								



# Cycle 61

## Instruction status:

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

## Functional unit status:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>dest Fi</i>	<i>S1 Fj</i>	<i>S2 Fk</i>	<i>FU Qj</i>	<i>FU Qk</i>	<i>Fj? Rj</i>	<i>Fk? Rk</i>
	Integer	No								
	Mult1	No								
	Mult2	No								
	Add	No								
0	Divide	Yes	Div	F10	F0	F6			No	No

## 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>
61	FU Divide								

# Cycle 62

## Instruction status:

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

## Functional unit status:

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>dest</i>		<i>S1</i>	<i>S2</i>	<i>FU</i>	<i>FU</i>	<i>Fj?</i>	<i>Fk?</i>
				<i>Fi</i>	<i>Fj</i>						
	Integer	No									
	Mult1	No									
	Mult2	No									
	Add	No									
	Divide	No									

## Register result status:

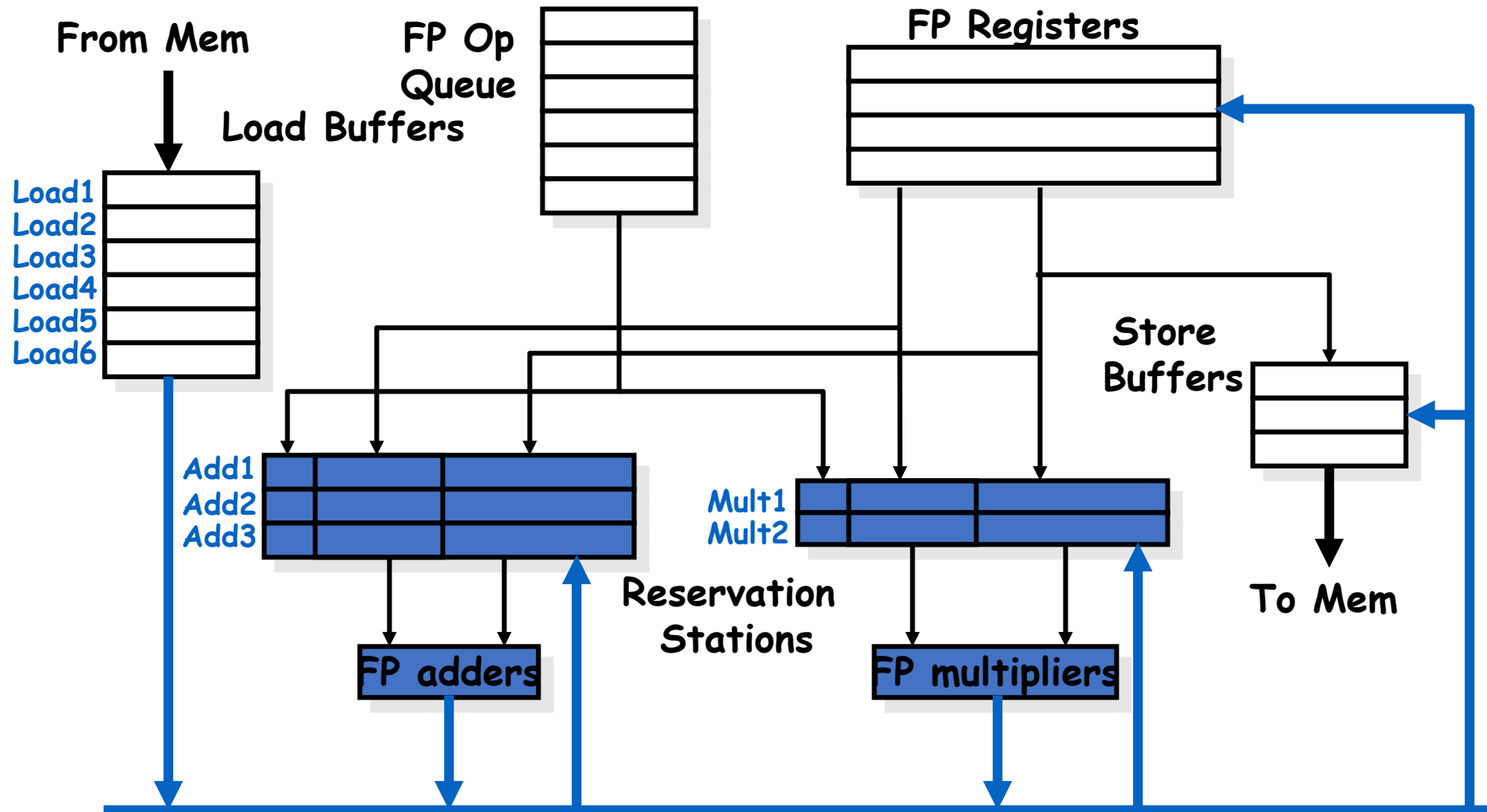
Clock	<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	...	<i>F30</i>
62	<i>FU</i>								

- In-order issue; out-of-order execute & commit

# Another Dynamic One: Tomasulo's Algorithm

- For IBM 360/91 about 3 years after CDC 6600 (1966)
- Goal: High Performance without special compilers
- Differences between IBM 360 & CDC 6600 ISA
  - IBM has only 2 register specifiers/instruction vs. 3 in CDC 6600
  - IBM has 4 FP registers vs. 8 in CDC 6600
  - IBM has memory-register ops
- Why Study? lead to Alpha 21264, HP 8000, MIPS 10000, Pentium II, PowerPC 604, ...

# Tomasulo's Organization



# Tomasulo vs Scoreboard

- Control & buffers distributed with Function Units (FU) vs. centralized in scoreboard;
  - 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, not through registers, over Common Data Bus that broadcasts results to all FUs
- Load and Stores treated as FUs with RSs as wells

# Reservation Station Components

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

**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: No ready flags as in Scoreboard;  $Q_j, Q_k = 0 \Rightarrow$  ready
- Store buffers only have  $Q_i$  for RS producing result

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

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

# Three Stages of Tomasulo Algorithm

## 1. Issue—get instruction from FP Op 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





# 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	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		Load2		Load1					

Note: Unlike 6600, 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	FU	Mult1	Load2			Load1			

- Note: registers names are removed ("renamed") in Reservation Stations; MULT issued vs. scoreboard
- Load1 completing; what is waiting for Load1?

# Cycle 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	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> <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 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	$S1$ $Vj$	$S2$ $Vk$	$RS$ $Qj$	$RS$ $Qk$
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	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 vs. scoreboard?

# 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:

<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					
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</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			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	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 vs. scoreboard?
- 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	<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	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	F0	F2	F4	F6	F8	F10	F12	...	F30
14	Mult1	M(A2)		(M-M+N)	(M-M)	Mult2			

# 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:

<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					
0	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>
15	FU	Mult1	M(A2)		(M-M+N)	(M-M)	Mult2		

# 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:

<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					
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			



# 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:

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					
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.

# Cycle 62

## *Instruction status:*

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

<i>Issue</i>	<i>Exec Comp</i>	<i>Write Result</i>
1	3	4
2	4	5
3	15	16
4	7	8
5	56	57
6	10	11

# Tomasulo vs Scoreboard

Pipelined Functional Units  
(6 load, 3 store, 3 +, 2 x/÷)  
window size:  $\leq 14$  instructions  
No issue on structural hazard  
WAR: renaming avoids  
WAW: renaming avoids  
Broadcast results from FU  
Control: reservation stations

Multiple Functional Units  
(1 load/store, 1 +, 2 x, 1 ÷)  
 $\leq 5$  instructions  
same  
stall completion  
stall issue  
Write/read registers  
central scoreboard