

CMSC 611: Advanced Computer Architecture

Scoreboard & Tomasulo

HW Schemes: Instruction Parallelism

- Why in HW at run time?
 - Works when can't know real dependence at compile time
 - Compiler simpler
 - Code for one machine runs well on another
- Key idea: Allow instructions behind stall to proceed
 - DIVD F0,F2,F4
 - ADDD F10,F0,F8
 - SUBD F12,F8,F14
 - Enables out-of-order execution => out-of-order completion
 - ID stage checks for structural and data hazards

Out of Order Execution

- 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
- Scoreboards allow instruction to execute whenever 1 & 2 hold, not waiting for prior instructions
- CDC 6600: In order issue, out of order execution, out of order commit/completion

Scoreboard Implications

- Out-of-order completion → WAR, WAW hazards

Example: DIVID F0, F2, F4

ADDD F10, F0, F8

SUBD F8, F8, F8

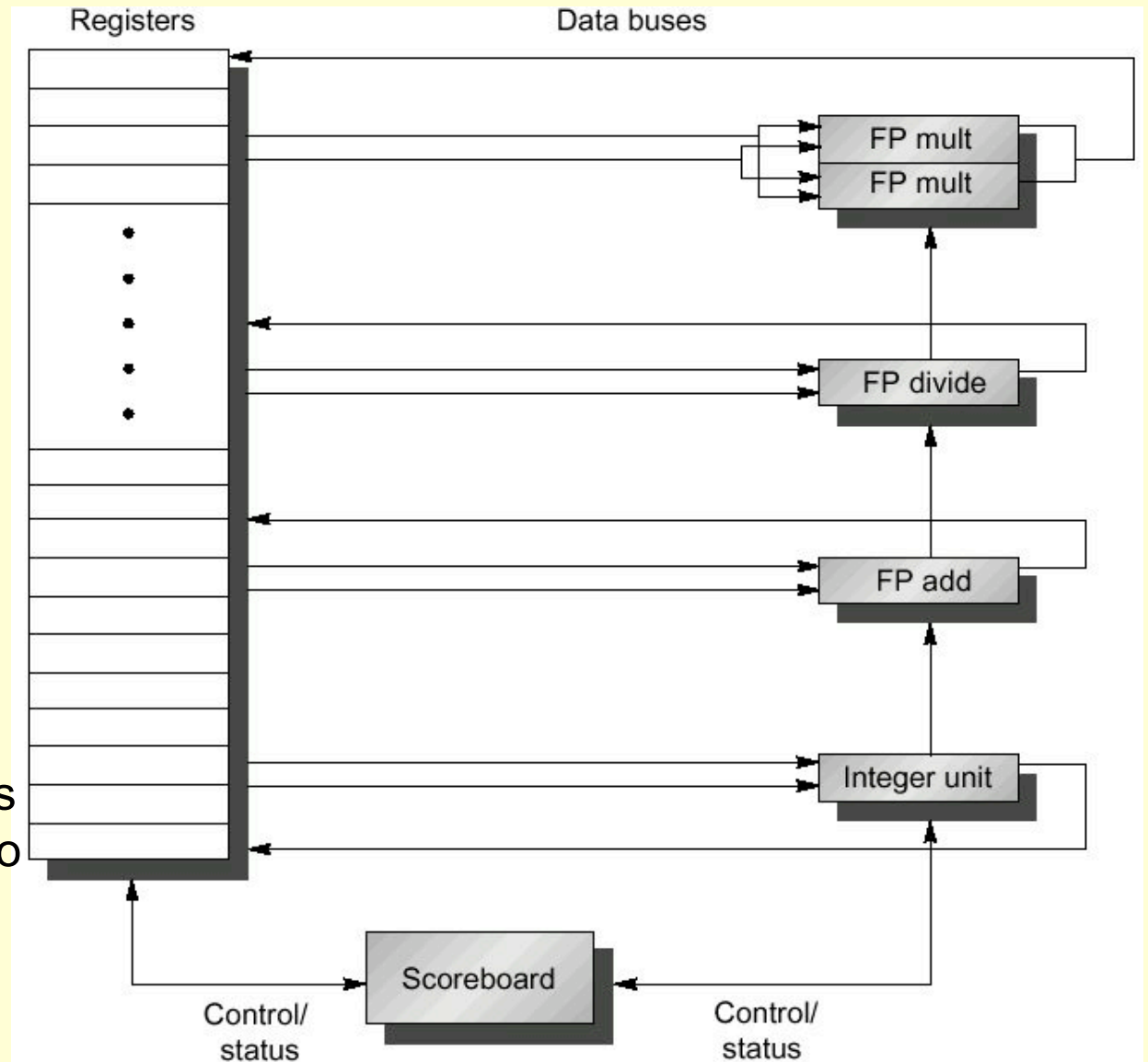
- Solutions for WAR
 - Queue both the operation and copies of its operands
 - Read registers only during Read Operands stage
- For WAW, must detect hazard: stall until other completes
- Scoreboard keeps track of dependencies, state or operations
 - Replace ID, EX, WB with 4 stages

Four Stages of Scoreboard

1. Issue—decode instructions & check for structural hazards (ID1).
 - If a functional unit for the instruction is free and no other active instruction has the same destination register (WAW), the scoreboard issues the instruction to the functional unit and updates its internal data structure.
 - If a structural or WAW hazard exists, then the instruction issue stalls, and no further instructions will issue until these hazards are cleared.
2. Read operands—wait until no data hazards, then read operands (ID2).
 - A source operand is available if no earlier issued active instruction is going to write it, or if the register containing the operand is being written by a currently active functional unit.
 - When the source operands are available, the scoreboard tells the functional unit to proceed to read the operands from the registers and begin execution.
 - The scoreboard resolves RAW hazards dynamically in this step, and instructions may be sent into execution out of order.
3. 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.
4. Write result—finish execution (WB)
 - Once the scoreboard is aware that the functional unit has completed execution, the scoreboard checks for WAR hazards. If none, it writes results, otherwise it stalls

MIPS Processor with Scoreboard

- Given the small latency of integer operations, it is not worth the scoreboard complexity
- 2 Multiplier, 1 divider, 1 adder and one integer unit
- Major cost driven by data buses
- The scoreboard control function units
- The scoreboard enables out-of-order execution to maximize parallelism



Three Parts of the Scoreboard

1. Instruction status—which of 4 steps for instruction
2. 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
3. Register result status—Indicates which functional unit will write each register, if any. Blank when no pending instructions will write that register

CDC Scoreboard

- Speedup 1.7 from compiler; 2.5 by hand
BUT slow memory (no cache)
- Limitations of 6600 scoreboard:
 - No forwarding hardware
 - Limited to instructions in basic block (small window)
 - Small number of functional units (causes structural hazards)
 - Do not issue on structural hazards
 - Wait for WAR hazards and prevent WAW hazards

Scoreboard Example Cycle 1

Instruction status

Instruction	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Read operand</i>	<i>Executic complet</i>	<i>Write Result</i>
LD F6	34+	R2	1			
LD F2	45+	R3				
MULT 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 for j Qj</i>	<i>FU for k 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>...</i>	<i>F30</i>
1									
	FU Integer								

Scoreboard Example Cycle 2

Instruction status

Instruction	<i>j</i>	<i>k</i>	Issue	Read operand	Executic complet	Write Result
LD	F6	34+	R2	1	2	
LD	F2	45+	R3			
MULT	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 for j</i> <i>Qj</i>	<i>FU for k</i> <i>Qk</i>	<i>Fj?</i> <i>Rj</i>	<i>Fk?</i> <i>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?

Scoreboard Example Cycle 3

Instruction status

Instruction	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Read operand</i>	<i>Executic complet</i>	<i>Write Result</i>
LD F6	34+	R2	1	2	3	
LD F2	45+	R3				
MULT 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 for j Qj</i>	<i>FU for k 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>...</i>	<i>F30</i>
3									
	FU Integer								

Scoreboard Example Cycle 4

Instruction status

Instruction	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Read operand</i>	<i>Executic complet</i>	<i>Write Result</i>
LD F6	34+	R2	1	2	3	4
LD F2	45+	R3				
MULT 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 for j Qj</i>	<i>FU for k 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>...</i>	<i>F30</i>
4									
	<i>FU</i> Integer								

Scoreboard Example Cycle 5

Instruction status

Instruction	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Read operand</i>	<i>Executic complet</i>	<i>Write Result</i>
LD F6	34+	R2	1	2	3	4
LD F2	45+	R3	5			
MULT 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 for j Qj</i>	<i>FU for k Qk</i>	<i>Fj? Rj</i>	<i>Fk? Rk</i>
	Integer	Yes	Load	F2		R3				Yes
	Mult1	No								
	Mult2	No								
	Add	No								
	Divide	No								

Register result status

Clock	F0	F2	F4	F6	F8	F10	F12	...	F30
5		Integer							

FU

Scoreboard Example Cycle 6

Instruction status

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

Functional unit status

Time	Name	Busy	Op	dest	S1	S2	FU for j	FU for k	Fj?	Fk?
				Fi	Fj	Fk	Qj	Qk	Rj	Rk
	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	F0	F2	F4	F6	F8	F10	F12	...	F30
6		Mult1	Integer						

Scoreboard Example Cycle 7

Instruction status

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

Functional unit status

Time	Name	Busy	Op	dest <i>Fi</i>	S1 <i>Fj</i>	S2 <i>Fk</i>	FU for <i>j</i> <i>Qj</i>	FU for <i>k</i> <i>Qk</i>	<i>Fj?</i> <i>Rj</i>	<i>Fk?</i> <i>Rk</i>
	Integer	Yes	Load	F2		R3				Yes
	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		Mult1	Integer		Add				

- Read multiply operands?

Scoreboard Example Cycle 8a

Instruction status

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

Functional unit status

Time	Name	Busy	Op	dest	S1	S2	FU for <i>j</i>	FU for <i>k</i>	F _{<i>j</i>} ?	F _{<i>k</i>} ?
				<i>F_i</i>	<i>F_j</i>	<i>F_k</i>	<i>Q_j</i>	<i>Q_k</i>	<i>R_j</i>	<i>R_k</i>
	Integer	Yes	Load	F2		R3				Yes
	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	F0	F2	F4	F6	F8	F10	F12	...	F30
8	Mult1	Integer			Add	Divide			

Scoreboard Example Cycle 8b

Instruction status

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

Functional unit status

Time	Name	Busy	Op	dest	S1	S2	FU for j	FU for k	Fj?	Fk?
				Fi	Fj	Fk	Qj	Qk	Rj	Rk
	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	F0	F2	F4	F6	F8	F10	F12	...	F30
8	Mult1				Add	Divide			

Scoreboard Example Cycle 11

Instruction status

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

Functional unit status

Time	Name	Busy	Op	dest <i>Fi</i>	S1 <i>Fj</i>	S2 <i>Fk</i>	FU for <i>j</i> <i>Qj</i>	FU for <i>k</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			Yes	Yes
	Mult2	No								
0	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>
11	Mult1				Add	Divide			

FU

Scoreboard Example Cycle 13

Instruction status

Instruction	<i>j</i>	<i>k</i>	Issue	Read operand	Executic complet	Write Result
LD F6	34+	R2	1	2	3	4
LD F2	45+	R3	5	6	7	8
MULT 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

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

Register result status

Clock	F0	F2	F4	F6	F8	F10	F12	...	F30
13	Mult1			Add		Divide			

FU

Scoreboard Example Cycle 14

Instruction status

Instruction	<i>j</i>	<i>k</i>	Issue	Read operand	Executic complet	Write Result
LD F6	34+	R2	1	2	3	4
LD F2	45+	R3	5	6	7	8
MULT 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>	S1 <i>Fj</i>	S2 <i>Fk</i>	FU for <i>j</i> <i>Qj</i>	FU for <i>k</i> <i>Qk</i>	<i>Fj?</i> <i>Rj</i>	<i>Fk?</i> <i>Rk</i>
	Integer	No								
5	Mult1	Yes	Mult	F0	F2	F4			Yes	Yes
	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			

FU

Scoreboard Example Cycle 15

Instruction status

Instruction	<i>j</i>	<i>k</i>	Issue	Read operand	Executic complet	Write Result
LD F6	34+	R2	1	2	3	4
LD F2	45+	R3	5	6	7	8
MULT 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>	S1 <i>Fj</i>	S2 <i>Fk</i>	FU for <i>j</i> <i>Qj</i>	FU for <i>k</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			Yes	Yes
	Mult2	No								
1	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>
15	Mult1			Add		Divide			

FU

Scoreboard Example Cycle 16

Instruction status

Instruction	<i>j</i>	<i>k</i>	Issue	Read operand	Executic complet	Write Result
LD F6	34+	R2	1	2	3	4
LD F2	45+	R3	5	6	7	8
MULT 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

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

FU

Scoreboard Example Cycle 17

Instruction status

Instruction	<i>j</i>	<i>k</i>	Issue	Read operand	Executic complet	Write Result
LD F6	34+	R2	1	2	3	4
LD F2	45+	R3	5	6	7	8
MULT 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

Time	Name	Busy	Op	dest <i>Fi</i>	S1 <i>Fj</i>	S2 <i>Fk</i>	FU for <i>j</i> <i>Qj</i>	FU for <i>k</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			Yes	Yes
	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>
17	Mult1			Add		Divide			

- Write result of ADDD?

Scoreboard Example Cycle 18

Instruction status

Instruction	<i>j</i>	<i>k</i>	Issue	Read operand	Executic complet	Write Result
LD F6	34+	R2	1	2	3	4
LD F2	45+	R3	5	6	7	8
MULT 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

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

Register result status

Clock	F0	F2	F4	F6	F8	F10	F12	...	F30
18	Mult1			Add		Divide			

FU

Scoreboard Example Cycle 19

Instruction status

Instruction	<i>j</i>	<i>k</i>	Issue	Read operand	Executic complet	Write Result
LD F6	34+	R2	1	2	3	4
LD F2	45+	R3	5	6	7	8
MULT 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

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

Register result status

Clock	F0	F2	F4	F6	F8	F10	F12	...	F30
19	Mult1			Add		Divide			

FU

Scoreboard Example Cycle 20

Instruction status

Instruction	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Read operand</i>	<i>Executic complet</i>	<i>Write Result</i>
LD F6	34+	R2	1	2	3	4
LD F2	45+	R3	5	6	7	8
MULT 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 for j Qj</i>	<i>FU for k Qk</i>	<i>Fj? Rj</i>	<i>Fk? Rk</i>
	Integer	No								
	Mult1	No								
	Mult2	No								
	Add	Yes	Add	F6	F8	F2			Yes	Yes
	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>...</i>	<i>F30</i>
20				Add		Divide			

FU

Scoreboard Example Cycle 21

Instruction status

Instruction	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Read operand</i>	<i>Executic complet</i>	<i>Write Result</i>
LD F6	34+	R2	1	2	3	4
LD F2	45+	R3	5	6	7	8
MULT 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 Fi</i>	<i>S1 Fj</i>	<i>S2 Fk</i>	<i>FU for j Qj</i>	<i>FU for k Qk</i>	<i>Fj? Rj</i>	<i>Fk? Rk</i>
	Integer	No								
	Mult1	No								
	Mult2	No								
	Add	Yes	Add	F6	F8	F2			Yes	Yes
	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			

FU

Scoreboard Example Cycle 22

Instruction status

Instruction	<i>j</i>	<i>k</i>	Issue	Read operand	Executic complet	Write Result
LD F6	34+	R2	1	2	3	4
LD F2	45+	R3	5	6	7	8
MULT 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

Time	Name	Busy	Op	dest <i>Fi</i>	S1 <i>Fj</i>	S2 <i>Fk</i>	FU for <i>j</i> <i>Qj</i>	FU for <i>k</i> <i>Qk</i>	<i>Fj?</i> <i>Rj</i>	<i>Fk?</i> <i>Rk</i>
	Integer	No								
	Mult1	No								
	Mult2	No								
	Add	No								
40	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>
22							Divide		

FU

Scoreboard Example Cycle 61

Instruction status

Instruction	<i>j</i>	<i>k</i>	Issue	Read operand	Executic complet	Write Result
LD F6	34+	R2	1	2	3	4
LD F2	45+	R3	5	6	7	8
MULT 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

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

Register result status

Clock	F0	F2	F4	F6	F8	F10	F12	...	F30
61						Divide			

FU

Scoreboard Example Cycle 62

Instruction status

Instruction	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Read operand</i>	<i>Executic complet</i>	<i>Write Result</i>
LD F6	34+	R2	1	2	3	4
LD F2	45+	R3	5	6	7	8
MULT 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 Fi</i>	<i>S1 Fj</i>	<i>S2 Fk</i>	<i>FU for j Qj</i>	<i>FU for k Qk</i>	<i>Fj? Rj</i>	<i>Fk? Rk</i>
	Integer	No								
	Mult1	No								
	Mult2	No								
	Add	No								
	0 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>...</i>	<i>F30</i>
62									

FU

Scoreboard Summary

- Why in HW at run time?
 - Works when can't know real dependence at compile time
 - Compiler simpler
 - Code for one machine runs well on another
- Key idea: Allow instructions behind stall to proceed
 - Enables out-of-order execution / out-of-order completion
 - ID stage checked both for structural and data hazards
- Out-of-order execution divides ID stage:
 - Issue—decode instructions, check for structural hazards
 - Read operands—wait until no data hazards, then read
- CDC 6600
 - Speedup 1.7 from compiler; 2.5 by hand BUT slow memory (no cache)
 - No forwarding (First write register then read it)
 - Limited to instructions in basic block (small window)
 - Number of functional units(structural hazards)
 - Wait for WAR hazards

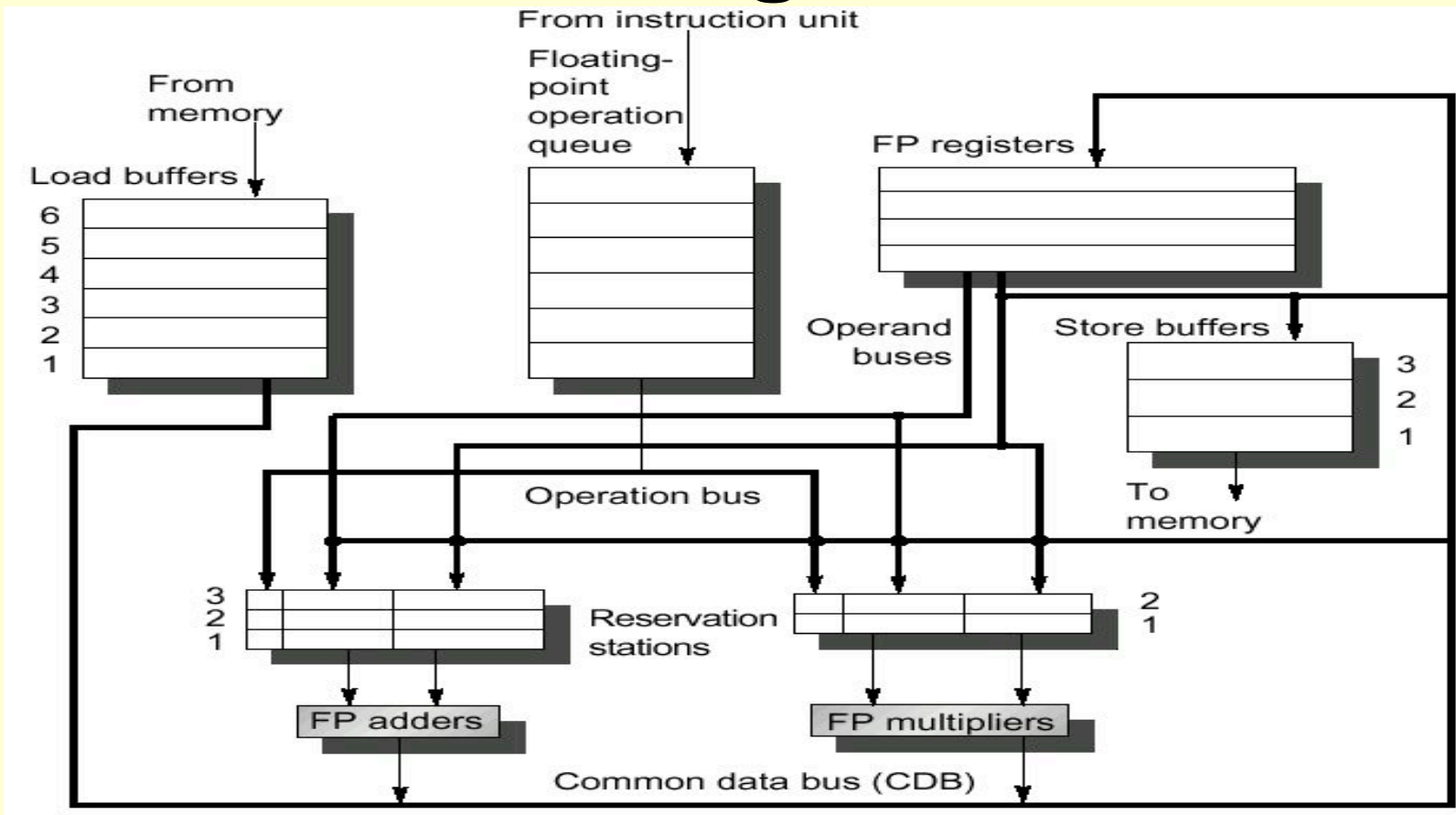
Tomasulo's Algorithm vs. Scoreboard

- Tomasulo's algorithm lead to Alpha 21264, HP 8000, MIPS 10000, Pentium II, PowerPC 604
- Many differences from scoreboard
 - Key: register renaming to avoid WAR and WAW
- Control & buffers distributed with Function Units (FU) vs. centralized in scoreboard;
 - FU buffers called “reservation stations”;
have pending operands

Tomasulo “Register Renaming”

- Registers in instructions replaced by values or pointers to reservation stations(RS)
 - avoids WAR, WAW hazards
 - More reservation stations than registers
 - Can do optimizations compilers cannot perform
- Operands to FU from RS, not registers
- Results broadcast over Common Data Bus
- Load and Stores treated as FU w/ RS
- Integer instructions can go past branches, allowing FP operations beyond basic block in FP queue

Tomasulo Organization



- The reservation stations hold instructions that had been issued and are awaiting execution at a functional unit
- All results from FP FU and loads are broadcasted on the CDB

Reservation Station Components

- Functional Unit Reservation Station
 - Op—Operation to perform in the unit (e.g., + or –)
 - V_j, V_k —Value of Source operands
 - Store buffers has V field, result to be stored
 - Q_j, Q_k —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 Reservation Station
 - 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 vs. Common Data Bus

- 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)
 - Broadcast: one to many

Tomasulo Example

Instruction status

Instruction	j	k	Issue	Execution complete	Write Result	Busy	Address
LD	F6	34+	R2			Load1	No
LD	F2	45+	R3		Load2	No	
MULT	F0	F2	F4		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 for j Qj	RS for k Qk
0	Add1	No					
0	Add2	No					
0	Add3	No					
0	Mult1	No					
0	Mult2	No					

Register result status

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

Tomasulo Example Cycle 1

Instruction status

Instruction	<i>j</i>	<i>k</i>	Issue	Execution complete	Write Result	Busy	Address
LD F6	34+	R2	1			Load1	34+R2
LD F2	45+	R3				Load2	
MULT F0	F2	F4				Load3	
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 for j</i> <i>Qj</i>	<i>RS for k</i> <i>Qk</i>
0	Add1	No					
0	Add2	No					
	Add3	No					
0	Mult1	No					
0	Mult2	No					

Register result status

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

Tomasulo Example Cycle 2

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

Reservation Stations			S1	S2	RS for <i>j</i>	RS for <i>k</i>
Time	Name	Busy Op	V _j	V _k	Q _j	Q _k
0	Add1	No				
0	Add2	No				
	Add3	No				
0	Mult1	No				
0	Mult2	No				

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

Note: Unlike 6600, can have multiple loads outstanding

Tomasulo Example Cycle 3

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

Reservation Stations			<i>S1</i>	<i>S2</i>	<i>RS for j</i>	<i>RS for k</i>
Time	Name	Busy Op	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>
0	Add1	No				
0	Add2	No				
	Add3	No				
0	Mult1	Yes MULTD		R(F4)	Load2	
0	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?

Tomasulo Example Cycle 4

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

Reservation Stations			<i>S1</i>	<i>S2</i>	<i>RS for j</i>	<i>RS for k</i>
Time	Name	Busy Op	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>
0	Add1	Yes SUBD	M(34+R2)			Load2
0	Add2	No				
	Add3	No				
0	Mult1	Yes MULTD		R(F4)	Load2	
0	Mult2	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	Mult1	Load2	M(34+R2)	Add1				

- Load2 completing; what is waiting for it?

Tomasulo Example Cycle 5

Instruction status

Instruction	<i>j</i>	<i>k</i>	Issue	Execution complete	Write Result	Busy	Address
LD F6	34+	R2	1	3	4	Load1	No
LD F2	45+	R3	2	4	5	Load2	No
MULT 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	<i>S1</i> <i>Vj</i>	<i>S2</i> <i>Vk</i>	<i>RS for j</i> <i>Qj</i>	<i>RS for k</i> <i>Qk</i>
2	Add1	Yes	SUBD	M(34+R2)	M(45+R3)		
0	Add2	No					
	Add3	No					
10	Mult1	Yes	MULTD	M(45+R3)	R(F4)		
0	Mult2	Yes	DIVD		M(34+R2)	Mult1	

Register result status

Clock	F0	F2	F4	F6	F8	F10	F12	...	F30
5	FU	Mult1	M(45+R3)		M(34+R2)	Add1	Mult2		

Tomasulo Example Cycle 6

Instruction status

Instruction	<i>j</i>	<i>k</i>	Issue	Execution complete	Write Result	Busy	Address
LD F6	34+	R2	1	3	4	Load1	No
LD F2	45+	R3	2	4	5	Load2	No
MULT 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 <i>V_j</i>	S2 <i>V_k</i>	RS for <i>j</i> <i>Q_j</i>	RS for <i>k</i> <i>Q_k</i>
1	Add1	Yes	SUBD	M(34+R2)	M(45+R3)		
0	Add2	Yes	ADDD		M(45+R3)	Add1	
	Add3	No					
9	Mult1	Yes	MULTD	M(45+R3)	R(F4)		
0	Mult2	Yes	DIVD		M(34+R2)	Mult1	

Register result status

Clock	F0	F2	F4	F6	F8	F10	F12	...	F30
6	FU	Mult1	M(45+R3)		Add2	Add1	Mult2		

- Issue ADDD here vs. scoreboard?

Tomasulo Example Cycle 7

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

Reservation Stations			<i>S1</i>	<i>S2</i>	<i>RS for j</i>	<i>RS for k</i>
Time	Name	Busy Op	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>
0	Add1	Yes SUBD	M(34+R2)	M(45+R3)		
0	Add2	Yes ADDD		M(45+R3)	Add1	
	Add3	No				
8	Mult1	Yes MULTD	M(45+R3)	R(F4)		
0	Mult2	Yes DIVD		M(34+R2)	Mult1	

Register result status		F0	F2	F4	F6	F8	F10	F12	...	F30
Clock										
7	FU	Mult1	M(45+R3)		Add2	Add1	Mult2			

- Add1 completing; what is waiting for it?

Tomasulo Example Cycle 8

Instruction status

Instruction	<i>j</i>	<i>k</i>	Issue	Execution complete	Write Result	Busy	Address
LD F6	34+	R2	1	3	4	Load1	No
LD F2	45+	R3	2	4	5	Load2	No
MULT 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 for j</i> <i>Qj</i>	<i>RS for k</i> <i>Qk</i>
0	Add1	No					
2	Add2	Yes	ADDD	M()-M()	M(45+R3)		
0	Add3	No					
7	Mult1	Yes	MULTD	M(45+R3)	R(F4)		
0	Mult2	Yes	DIVD		M(34+R2)	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(45+R3)		Add2	M()-M()	Mult2			

Tomasulo Example Cycle 9

Instruction status

Instruction	<i>j</i>	<i>k</i>	Issue	Execution complete	Write Result	Busy	Address
LD F6	34+	R2	1	3	4	Load1	No
LD F2	45+	R3	2	4	5	Load2	No
MULT 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 for j</i> <i>Qj</i>	<i>RS for k</i> <i>Qk</i>
0	Add1	No					
1	Add2	Yes	ADDD	M()-M()	M(45+R3)		
0	Add3	No					
6	Mult1	Yes	MULTD	M(45+R3)	R(F4)		
0	Mult2	Yes	DIVD		M(34+R2)	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(45+R3)		Add2	M()-M()	Mult2			

Tomasulo Example Cycle 10

Instruction status

Instruction	<i>j</i>	<i>k</i>	Issue	Execution complete	Write Result	Busy	Address
LD F6	34+	R2	1	3	4	Load1	No
LD F2	45+	R3	2	4	5	Load2	No
MULT 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>Vj</i>	<i>S2</i> <i>Vk</i>	<i>RS for j</i> <i>Qj</i>	<i>RS for k</i> <i>Qk</i>
0	Add1	No					
0	Add2	Yes	ADDD	$M()-M()$	$M(45+R3)$		
0	Add3	No					
5	Mult1	Yes	MULTD	$M(45+R3)$	$R(F4)$		
0	Mult2	Yes	DIVD		$M(34+R2)$	Mult1	

Register result status

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
10	FU	Mult1	$M(45+R3)$		Add2	$M()-M()$	Mult2			

- Add2 completing; what is waiting for it?

Tomasulo Example Cycle 11

Instruction status

Instruction	<i>j</i>	<i>k</i>	Issue	Execution complete	Write Result	Busy	Address
LD F6	34+	R2	1	3	4	Load1	No
LD F2	45+	R3	2	4	5	Load2	No
MULT 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 for j</i> <i>Qj</i>	<i>RS for k</i> <i>Qk</i>
0	Add1	No					
0	Add2	No					
0	Add3	No					
4	Mult1	Yes	MULTD	M(45+R3)	R(F4)		
0	Mult2	Yes	DIVD		M(34+R2)	Mult1	

Register result status

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
11	FU	Mult1	M(45+R3)		(M-M)+M()	M()-M()	Mult2			

- Write result of ADDD here vs. scoreboard?

Tomasulo Example Cycle 12

Instruction status

Instruction	<i>j</i>	<i>k</i>	Issue	Execution complete	Write Result	Busy	Address
LD F6	34+	R2	1	3	4	Load1	No
LD F2	45+	R3	2	4	5	Load2	No
MULT 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 for j</i> <i>Qj</i>	<i>RS for k</i> <i>Qk</i>
0	Add1	No					
0	Add2	No					
0	Add3	No					
3	Mult1	Yes	MULTD	M(45+R3)	R(F4)		
0	Mult2	Yes	DIVD		M(34+R2)	Mult1	

Register result status

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
12	FU	Mult1	M(45+R3)		(M-M)+M()	M()-M()	Mult2			

- Note: all quick instructions complete already

Tomasulo Example Cycle 13

Instruction status

Instruction	<i>j</i>	<i>k</i>	Issue	Execution complete	Write Result	Busy	Address
LD F6	34+	R2	1	3	4	Load1	No
LD F2	45+	R3	2	4	5	Load2	No
MULT 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 for j</i> <i>Qj</i>	<i>RS for k</i> <i>Qk</i>
0	Add1	No					
0	Add2	No					
	Add3	No					
2	Mult1	Yes	MULTD	M(45+R3)	R(F4)		
0	Mult2	Yes	DIVD		M(34+R2)	Mult1	

Register result status

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
13	FU	Mult1	M(45+R3)		(M-M)+M()	M()-M()	Mult2			

Tomasulo Example Cycle 14

Instruction status

Instruction	<i>j</i>	<i>k</i>	Issue	Execution complete	Write Result	Busy	Address
LD F6	34+	R2	1	3	4	Load1	No
LD F2	45+	R3	2	4	5	Load2	No
MULT 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 for j</i> <i>Qj</i>	<i>RS for k</i> <i>Qk</i>
0	Add1	No					
0	Add2	No					
0	Add3	No					
1	Mult1	Yes	MULTD	M(45+R3)	R(F4)		
0	Mult2	Yes	DIVD		M(34+R2)	Mult1	

Register result status

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
14	FU	Mult1	M(45+R3)		(M-M)+M()	M()-M()	Mult2			

Tomasulo Example Cycle 15

Instruction status				Execution	Write		Busy	Address
Instruction	<i>j</i>	<i>k</i>	Issue	complete	Result			
LD	F6	34+	R2	1	3	4	Load1	No
LD	F2	45+	R3	2	4	5	Load2	No
MULT	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			S1	S2	RS for <i>j</i>	RS for <i>k</i>
Time	Name	Busy Op	V _j	V _k	Q _j	Q _k
0	Add1	No				
0	Add2	No				
	Add3	No				
0	Mult1	Yes	MULTD	M(45+R3)	R(F4)	
0	Mult2	Yes	DIVD		M(34+R2)	Mult1

Register result status		F0	F2	F4	F6	F8	F10	F12	...	F30
Clock										
15	FU	Mult1	M(45+R3)		(M-M)+M()	M()-M()	Mult2			

- Mult1 completing; what is waiting for it?

Tomasulo Example Cycle 16

Instruction status				Execution	Write		Busy	Address
Instruction	<i>j</i>	<i>k</i>	Issue	complete	Result			
LD	F6	34+	R2	1	3	4	Load1	No
LD	F2	45+	R3	2	4	5	Load2	No
MULT	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>S1</i>	<i>S2</i>	<i>RS for j</i>	<i>RS for k</i>
Time	Name	Busy Op	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>
0	Add1	No				
0	Add2	No				
	Add3	No				
0	Mult1	No				
40	Mult2	Yes	DIVD	M*F4		M(34+R2)

Register result status		<i>F0</i>	<i>F2</i>	<i>F4</i>	<i>F6</i>	<i>F8</i>	<i>F10</i>	<i>F12</i>	...	<i>F30</i>
Clock										
16	FU	M*F4	M(45+R3)		(M-M)+M()	M()-M()	Mult2			

- Note: Just waiting for divide

Tomasulo Example Cycle 55

Instruction status

Instruction	<i>j</i>	<i>k</i>	Issue	Execution complete	Write Result	Busy	Address
LD F6	34+	R2	1	3	4	Load1	No
LD F2	45+	R3	2	4	5	Load2	No
MULT 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 for j</i> <i>Qj</i>	<i>RS for k</i> <i>Qk</i>
0	Add1	No					
0	Add2	No					
	Add3	No					
0	Mult1	No					
1	Mult2	Yes	DIVD	M*F4	M(34+R2)		

Register result status

Clock	F0	F2	F4	F6	F8	F10	F12	...	F30
55	FU	M*F4	M(45+R3)		(M-M)+M()	M()-M()	Mult2		

Tomasulo Example Cycle 56

Instruction status				Execution	Write		Busy	Address
Instruction	<i>j</i>	<i>k</i>	Issue	complete	Result			
LD	F6	34+	R2	1	3	4	Load1	No
LD	F2	45+	R3	2	4	5	Load2	No
MULT	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>S1</i>	<i>S2</i>	<i>RS for j</i>	<i>RS for k</i>
Time	Name	Busy Op	<i>Vj</i>	<i>Vk</i>	<i>Qj</i>	<i>Qk</i>
0	Add1	No				
0	Add2	No				
	Add3	No				
0	Mult1	No				
0	Mult2	Yes	DIVD	M*F4		M(34+R2)

Register result status		F0	F2	F4	F6	F8	F10	F12	...	F30
Clock										
56	FU	M*F4	M(45+R3)		(M-M)+M()	M()-M()	Mult2			

- Mult 2 completing; what is waiting for it?

Tomasulo Example Cycle 57

Instruction status				Execution	Write	Busy	Address	
Instruction	<i>j</i>	<i>k</i>	Issue	complete	Result			
LD	F6	34+	R2	1	3	4	Load1	No
LD	F2	45+	R3	2	4	5	Load2	No
MULT	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			S1	S2	RS for <i>j</i>	RS for <i>k</i>
Time	Name	Busy Op	V _j	V _k	Q _j	Q _k
0	Add1	No				
0	Add2	No				
	Add3	No				
0	Mult1	No				
0	Mult2	No				

Register result status

Clock	F0	F2	F4	F6	F8	F10	F12	...	F30
57	FU	M*F4	M(45+R3)		(M-M)+M()	M()-M()	M*F4/M		

- Again, in-order issue, out-of-order execution, completion

Compare to Scoreboard

Instruction status

Instruction	<i>j</i>	<i>k</i>	<i>Issue</i>	<i>Read operand</i>	<i>Executic complet</i>	<i>Write Result</i>
LD F6	34+	R2	1	2	3	4
LD F2	45+	R3	5	6	7	8
MULT 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

Time Name

Busy Op dest S1 S2 FU for j FU for k Fj? Fk?
Busy Op Fi Fj Fk Qj Qk Rj Rk

Integer

No

Mult1

No

Mult2

No

Add

No

0 Divide

No

Register result status

Clock

F0 F2 F4 F6 F8 F10 F12 ... F30

62

FU

- Why longer on Scoreboard/6600?

Tomasulo vs. Scoreboard

	Tomasulo (IBM 360/91)	Scoreboard (CDC 6600)
Functional Units	Pipelined (6 load, 3 store, 3 +, 2 x/÷)	Multiple (1 load/store, 1 +, 2 x, 1 ÷)
Window size	14 instructions	5 instructions
Structural hazard	No issue	No issue
WAR	Renaming avoids	Stall completion
WAW	Renaming avoids	Stall completion
Operands	Broadcast results from FU	Write/read registers
Control	Reservation stations	Central scoreboard

- Tomasulo Drawbacks
 - Circuit complexity
 - Many associative stores (CDB) at high speed
 - Performance limited by Common Data Bus
 - Multiple CDBs → more FU logic for parallel associative stores

Tomasulo Loop Example

Loop:LD	F0	0	R1
MULTD	F4	F0	F2
SD	F4	0	R1
SUBI	R1	R1	#8
BNEZ		R1	Loop

- Assume Multiply takes 4 clocks
- Assume first load takes 8 clocks (cache miss), second load takes 4 clocks (hit)
- To be clear, will show clocks for SUBI, BNEZ
- Reality, integer instructions ahead

Loop Example Cycle 0

Instruction status

Instruction	j	k	iteration	Issue	Execution complete	Write Result	Busy	Address
LD F0	0	R1	1		Load1	No		Qi
MULT F4	F0	F2	1		Load2	No		
SD F4	0	R1	1		Load3	No		
LD F0	0	R1	2		Store1	No		
MULT F4	F0	F2	2		Store2	No		
SD F4	0	R1	2		Store3	No		

Reservation Stations

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk	Code:
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
0	Mult1	No						SUBI R1 R1 #8
0	Mult2	No						BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
0	80	Qi							

Loop Example Cycle 1

Instruction status

Instruction	j	k	iteration	Issue	Execution Write complete	Result	Busy	Address
LD F0	0	R1	1	1			Yes	80
MULT F4	F0	F2	1				No	
SD F4	0	R1	1				No	Qi
LD F0	0	R1	2				No	
MULT F4	F0	F2	2				No	
SD F4	0	R1	2				No	

Reservation Stations

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk	Code:
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
0	Mult1	No						SUBI R1 R1 #8
0	Mult2	No						BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
1	80	Qi	Load1						

Loop Example Cycle 2

Instruction status

Instruction	j	k	iteration	Issue	Execution complete	Write Result	Busy	Address
LD F0	0	R1	1	1		Load1	Yes	80
MULT F4	F0	F2	1	2		Load2	No	
SD F4	0	R1	1			Load3	No	Qi
LD F0	0	R1	2			Store1	No	
MULT F4	F0	F2	2			Store2	No	
SD F4	0	R1	2			Store3	No	

Reservation Stations

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk	Code:
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
0	Mult1	Yes	MULTD		R(F2)	Load1		SUBI R1 R1 #8
0	Mult2	No						BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
2	80	Qi	Load1	Mult1					

Loop Example Cycle 3

Instruction status				Execution Write			Busy	Address	
Instruction	<i>j</i>	<i>k</i>	iteration	Issue	complete	Result			
LD	F0	0	R1	1	<div style="border: 1px solid black; width: 100%; height: 100%;"></div>	Load1	Yes	80	
MULT	F4	F0	F2	1		Load2	No		
SD	F4	0	R1	1		Load3	No	Qi	
LD	F0	0	R1	2		Store1	Yes	80	Mult1
MULT	F4	F0	F2	2		Store2	No		
SD	F4	0	R1	2		Store3	No		

Reservation Stations

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk	Code:
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
0	Mult1	Yes	MULTD		R(F2)	Load1		SUBI R1 R1 #8
0	Mult2	No						BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
3	80	Qi	Load1	Mult1					

- Note: MULT1 has no registers names in RS

Loop Example Cycle 4

Instruction status

Instruction	j	k	iteration	Issue	Execution complete	Write Result	Busy	Address
LD F0	0	R1	1	1		Load1	Yes	80
MULT F4	F0	F2	1	2		Load2	No	
SD F4	0	R1	1	3		Load3	No	Qi
LD F0	0	R1	2			Store1	Yes	80
MULT F4	F0	F2	2			Store2	No	
SD F4	0	R1	2			Store3	No	

Reservation Stations

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk	Code:
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
0	Mult1	Yes	MULTD		R(F2)	Load1		SUBI R1 R1 #8
0	Mult2	No						BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
4	72	Qi	Load1	Mult1					

- Issue SUBI

Loop Example Cycle 5

Instruction status

Instruction	j	k	iteration	Issue	Execution complete	Write Result	Busy	Address
LD F0	0	R1	1	1		Load1	Yes	80
MULT F4	F0	F2	1	2		Load2	No	
SD F4	0	R1	1	3		Load3	No	Qi
LD F0	0	R1	2			Store1	Yes	80
MULT F4	F0	F2	2			Store2	No	
SD F4	0	R1	2			Store3	No	

Reservation Stations

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk	Code:
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
0	Mult1	Yes	MULTD		R(F2)	Load1		SUBI R1 R1 #8
0	Mult2	No						BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
5	72	Qi	Load1	Mult1					

- Issue BNEZ

Loop Example Cycle 6

Instruction status				Execution Write			Busy	Address
Instruction	<i>j</i>	<i>k</i>	iteration	Issue	complete	Result		
LD F0	0	R1	1	1		Load1	Yes	80
MULT F4	F0	F2	1	2		Load2	Yes	72
SD F4	0	R1	1	3		Load3	No	Qi
LD F0	0	R1	2	6		Store1	Yes	80
MULT F4	F0	F2	2			Store2	No	
SD F4	0	R1	2			Store3	No	

Reservation Stations

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk	Code:
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
0	Mult1	Yes	MULTD		R(F2)	Load1		SUBI R1 R1 #8
0	Mult2	No						BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
6	72	Qi	Load2	Mult1					

- Note: F0 never sees Load1 result

Loop Example Cycle 7

Instruction status

Instruction	j	k	iteration	Issue	Execution complete	Write Result	Busy	Address
LD F0	0	R1	1	1		Load1	Yes	80
MULT F4	F0	F2	1	2		Load2	Yes	72
SD F4	0	R1	1	3		Load3	No	Qi
LD F0	0	R1	2	6		Store1	Yes	80
MULT F4	F0	F2	2	7		Store2	No	
SD F4	0	R1	2			Store3	No	

Reservation Stations

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk	Code:
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
0	Mult1	Yes	MULTD		R(F2)	Load1		SUBI R1 R1 #8
0	Mult2	Yes	MULTD		R(F2)	Load2		BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
7	72	Qi	Load2	Mult2					

- Note: MULT2 has no registers names in RS

Loop Example Cycle 8

Instruction status

Instruction	j	k	iteration	Issue	Execution complete	Write Result	Busy	Address
LD F0	0	R1	1	1		Load1	Yes	80
MULT F4	F0	F2	1	2		Load2	Yes	72
SD F4	0	R1	1	3		Load3	No	Qi
LD F0	0	R1	2	6		Store1	Yes	80
MULT F4	F0	F2	2	7		Store2	Yes	72
SD F4	0	R1	2	8		Store3	No	

Reservation Stations

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk	Code:
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
0	Mult1	Yes	MULTD		R(F2)	Load1		SUBI R1 R1 #8
0	Mult2	Yes	MULTD		R(F2)	Load2		BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
8	72	Qi	Load2	Mult2					

Loop Example Cycle 9

Instruction status				Execution Write			Busy	Address
Instruction	<i>j</i>	<i>k</i>	iteration	Issue	complete	Result		
LD	F0	0	R1	1	9	Load1	Yes	80
MULT	F4	F0	F2	1	2	Load2	Yes	72
SD	F4	0	R1	1	3	Load3	No	Qi
LD	F0	0	R1	2	6	Store1	Yes	80
MULT	F4	F0	F2	2	7	Store2	Yes	72
SD	F4	0	R1	2	8	Store3	No	

Reservation Stations

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk	Code:
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
0	Mult1	Yes	MULTD		R(F2)	Load1		SUBI R1 R1 #8
0	Mult2	Yes	MULTD		R(F2)	Load2		BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
9	64	Qi	Load2	Mult2					

- Load1 completing; what is waiting for it?

Loop Example Cycle 10

Instruction status				Execution Write			Busy	Address
Instruction	<i>j</i>	<i>k</i>	<i>iteration</i>	<i>Issue</i>	<i>complete</i>	<i>Result</i>		
LD F0	0	R1	1	1	9	10	Load1	No
MULT F4	F0	F2	1	2			Load2	Yes 72
SD F4	0	R1	1	3			Load3	No Qi
LD F0	0	R1	2	6	10		Store1	Yes 80 Mult1
MULT F4	F0	F2	2	7			Store2	Yes 72 Mult2
SD F4	0	R1	2	8			Store3	No

Reservation Stations

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk	Code:
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
4	Mult1	Yes	MULTD	M(80)	R(F2)			SUBI R1 R1 #8
0	Mult2	Yes	MULTD		R(F2)	Load2		BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
10	64 Qi	Load2		Mult2					

- Load2 completing; what is waiting for it?

Loop Example Cycle 11

Instruction status

Instruction	j	k	iteration	Issue	Execution complete	Write Result	Busy	Address
LD F0	0	R1	1	1	9	10	No	
MULT F4	F0	F2	1	2			No	
SD F4	0	R1	1	3			Yes	64 Qi
LD F0	0	R1	2	6	10	11	Yes	80 Mult1
MULT F4	F0	F2	2	7			Yes	72 Mult2
SD F4	0	R1	2	8			No	

Reservation Stations

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk	Code:
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
3	Mult1	Yes	MULTD	M(80)	R(F2)			SUBI R1 R1 #8
4	Mult2	Yes	MULTD	M(72)	R(F2)			BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
11	64 Qi	Load3		Mult2					

Loop Example Cycle 12

Instruction status

Instruction	j	k	iteration	Issue	Execution complete	Write Result	Busy	Address
LD F0	0	R1	1	1	9	10	No	
MULT F4	F0	F2	1	2			No	
SD F4	0	R1	1	3			Yes	64 Qi
LD F0	0	R1	2	6	10	11	Yes	80 Mult1
MULT F4	F0	F2	2	7			Yes	72 Mult2
SD F4	0	R1	2	8			No	

Reservation Stations

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk	Code:
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
2	Mult1	Yes	MULTD	M(80)	R(F2)			SUBI R1 R1 #8
3	Mult2	Yes	MULTD	M(72)	R(F2)			BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
12	64 Qi	Load3		Mult2					

Loop Example Cycle 13

Instruction status

Instruction	j	k	iteration	Issue	Execution complete	Write Result	Busy	Address
LD F0	0	R1	1	1	9	10	No	
MULT F4	F0	F2	1	2			No	
SD F4	0	R1	1	3			Yes	64 Qi
LD F0	0	R1	2	6	10	11	Yes	80 Mult1
MULT F4	F0	F2	2	7			Yes	72 Mult2
SD F4	0	R1	2	8			No	

Reservation Stations

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk	Code:
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
1	Mult1	Yes	MULTD	M(80)	R(F2)			SUBI R1 R1 #8
2	Mult2	Yes	MULTD	M(72)	R(F2)			BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
13	64 Qi	Load3		Mult2					

Loop Example Cycle 14

Instruction status				Execution Write			Busy	Address	
Instruction	<i>j</i>	<i>k</i>	<i>iteration</i>	<i>Issue</i>	<i>complete</i>	<i>Result</i>			
LD	F0	0	R1	1	9	10	Load1	No	
MULT	F4	F0	F2	1	14		Load2	No	
SD	F4	0	R1	1			Load3	Yes	64 Qi
LD	F0	0	R1	2	10	11	Store1	Yes	80 Mult1
MULT	F4	F0	F2	2	7		Store2	Yes	72 Mult2
SD	F4	0	R1	2	8		Store3	No	

Reservation Stations

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for <i>j</i> Qj	RS for <i>k</i> Qk	Code:
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
0	Mult1	Yes	MULTD	M(80)	R(F2)			SUBI R1 R1 #8
1	Mult2	Yes	MULTD	M(72)	R(F2)			BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
14	64 Qi	Load3		Mult2					

- Mult1 completing; what is waiting for it?

Loop Example Cycle 15

Instruction status

Instruction	j	k	iteration	Issue	Execution complete	Write Result	Busy	Address
LD F0	0	R1	1	1	9	10	No	
MULT F4	F0	F2	1	2	14	15	No	
SD F4	0	R1	1	3			Yes	64 Qi
LD F0	0	R1	2	6	10	11	Yes	80 $M(80)*R(F1)$
MULT F4	F0	F2	2	7	15		Yes	72 Mult2
SD F4	0	R1	2	8			No	

Reservation Stations

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk	Code:
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
0	Mult1	No						SUBI R1 R1 #8
0	Mult2	Yes	MULTD	M(72)	R(F2)			BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
15	64 Qi	Load3		Mult2					

- Mult2 completing; what is waiting for it?

Loop Example Cycle 16

Instruction status

Instruction	j	k	iteration	Issue	Execution complete	Write Result	Busy	Address
LD F0	0	R1	1	1	9	10	Load1	No
MULT F4	F0	F2	1	2	14	15	Load2	No
SD F4	0	R1	1	3			Load3	Yes 64 Qi
LD F0	0	R1	2	6	10	11	Store1	Yes 80 $M(80)*R(F_0)$
MULT F4	F0	F2	2	7	15	16	Store2	Yes 72 $M(72)*R(F_0)$
SD F4	0	R1	2	8			Store3	No

Reservation Stations

Time	Name	Busy	Op	$S1$ Vj	$S2$ Vk	RS for j Qj	RS for k Qk	Code:
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
0	Mult1	Yes	MULTD		R(F2)	Load3		SUBI R1 R1 #8
0	Mult2	No						BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
16	64 Qi	Load3		Mult1					

Loop Example Cycle 17

Instruction status

Instruction	j	k	iteration	Issue	Execution complete	Write Result	Busy	Address
LD F0	0	R1	1	1	9	10	Load1	No
MULT F4	F0	F2	1	2	14	15	Load2	No
SD F4	0	R1	1	3			Load3	Yes 64 Qi
LD F0	0	R1	2	6	10	11	Store1	Yes 80 $M(80)*R(F_0)$
MULT F4	F0	F2	2	7	15	16	Store2	Yes 72 $M(72)*R(F_0)$
SD F4	0	R1	2	8			Store3	Yes 64 Mult1

Reservation Stations

Time	Name	Busy	Op	S1 V_j	S2 V_k	RS for j Q_j	RS for k Q_k	Code:
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
0	Mult1	Yes	MULTD		R(F2)	Load3		SUBI R1 R1 #8
0	Mult2	No						BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
17	64 Qi	Load3		Mult1					

Loop Example Cycle 18

Instruction status

Instruction	j	k	iteration	Issue	Execution complete	Write Result	Busy	Address
LD F0	0	R1	1	1	9	10	Load1	No
MULT F4	F0	F2	1	2	14	15	Load2	No
SD F4	0	R1	1	3	18		Load3	Yes 64 Qi
LD F0	0	R1	2	6	10	11	Store1	Yes 80 $M(80)*R(F_0)$
MULT F4	F0	F2	2	7	15	16	Store2	Yes 72 $M(72)*R(F_0)$
SD F4	0	R1	2	8			Store3	Yes 64 Mult1

Reservation Stations

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk	Code:
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
0	Mult1	Yes	MULTD		R(F2)	Load3		SUBI R1 R1 #8
0	Mult2	No						BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
18	56	Qi	Load3						Mult1

Loop Example Cycle 19

Instruction status

Instruction	<i>j</i>	<i>k</i>	<i>iteration</i>	<i>Issue</i>	<i>Execution complete</i>	<i>Write Result</i>		Busy	Address
LD F0	0	R1	1	1	9	10	Load1	No	
MULT F4	F0	F2	1	2	14	15	Load2	No	
SD F4	0	R1	1	3	18	19	Load3	Yes	64 Qi
LD F0	0	R1	2	6	10	11	Store1	No	
MULT F4	F0	F2	2	7	15	16	Store2	Yes	72 M(72)*R(72)
SD F4	0	R1	2	8			Store3	Yes	64 Mult1

Reservation Stations

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>S1</i> <i>Vj</i>	<i>S2</i> <i>Vk</i>	<i>RS for j</i> <i>Qj</i>	<i>RS for k</i> <i>Qk</i>	<i>Code:</i>
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
0	Mult1	Yes	MULTD		R(F2)	Load3		SUBI R1 R1 #8
0	Mult2	No						BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
19	56 Qi	Load3		Mult1					

Loop Example Cycle 20

Instruction status

Instruction	<i>j</i>	<i>k</i>	<i>iteration</i>	<i>Issue</i>	<i>Execution complete</i>	<i>Write Result</i>		Busy	Address
LD F0	0	R1	1	1	9	10	Load1	No	
MULT F4	F0	F2	1	2	14	15	Load2	No	
SD F4	0	R1	1	3	18	19	Load3	Yes	64 Qi
LD F0	0	R1	2	6	10	11	Store1	No	
MULT F4	F0	F2	2	7	15	16	Store2	Yes	72 M(72)*R(72)
SD F4	0	R1	2	8	20		Store3	Yes	64 Mult1

Reservation Stations

<i>Time</i>	<i>Name</i>	<i>Busy</i>	<i>Op</i>	<i>S1</i> <i>Vj</i>	<i>S2</i> <i>Vk</i>	<i>RS for j</i> <i>Qj</i>	<i>RS for k</i> <i>Qk</i>	<i>Code:</i>
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
0	Mult1	Yes	MULTD		R(F2)	Load3		SUBI R1 R1 #8
0	Mult2	No						BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
20	56 Qi	Load3		Mult1					

Loop Example Cycle 21

Instruction status

Instruction	j	k	iteration	Issue	Execution complete	Write Result	Busy	Address
LD F0	0	R1	1	1	9	10	Load1	No
MULT F4	F0	F2	1	2	14	15	Load2	No
SD F4	0	R1	1	3	18	19	Load3	Yes 64 Qi
LD F0	0	R1	2	6	10	11	Store1	No
MULT F4	F0	F2	2	7	15	16	Store2	No
SD F4	0	R1	2	8	20	21	Store3	Yes 64 Mult1

Reservation Stations

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk	Code:
0	Add1	No						LD F0 0 R1
0	Add2	No						MULT F4 F0 F2
0	Add3	No						SD F4 0 R1
0	Mult1	Yes	MULTD		R(F2)	Load3		SUBI R1 R1 #8
0	Mult2	No						BNEZ R1 Loop

Register result status

Clock	R1	F0	F2	F4	F6	F8	F10	F12...	F30
21	56	Qi	Load3			Mult1			