

UNIVERSITY OF MASSACHUSETTS  
Dept. of Electrical & Computer Engineering

Advanced Computer Architecture  
ECE 568

Part 6

Dynamic Scheduling - Tomasulo's Algorithm

Israel Koren  
Fall 2009

ECE568/Koren Part.6 .1

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## HW Dynamic Scheduling

- ◆ Enables **out-of-order execution** and allows **out-of-order completion**
- ◆ All instructions pass through issue stage in order (**in-order issue**)
- ◆ Split the ID pipe stage of pipeline into 2 stages:
- ◆ **Issue**—Decode instructions, check for structural hazards
- ◆ **Read operands**—Wait until no data hazards, then read operands

ECE568/Koren Part.6 .2

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

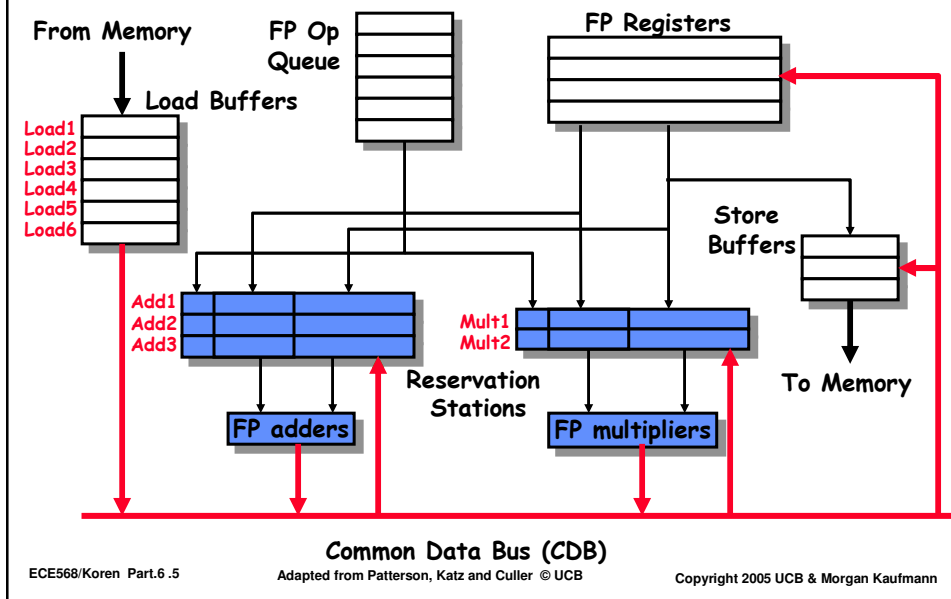
## Dynamic Scheduling: Tomasulo's Algorithm

- ◆ For IBM 360/91
- ◆ Goal: High Performance without special compilers
- ◆ Small number of floating point registers (4 in 360) prevented interesting compiler scheduling of operations
  - This led Tomasulo to try to figure out how to get more effective registers — **renaming in hardware!**
- ◆ Why Study 1966 Computer?
- ◆ The descendants of this have flourished!
  - Alpha 21264, HP 8000, MIPS 10000, Pentium III, PowerPC 604, ...

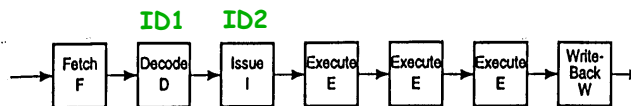
## Tomasulo Algorithm

- ◆ 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):
  - form of **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 well

## Tomasulo's Organization



## Example: $X = Y + Z$ & $A = B * C$



In-order instruction issuing	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	
$R_1 \leftarrow (Y)$	F	D	I	E	E	E	W																
$R_2 \leftarrow (Z)$		F	D	I	E	E	E	W															
$R_3 \leftarrow (R_1) + (R_2)$			F	D	*	*	I	E	E	E	W												
$X \leftarrow (R_3)$				F	*	*	D	*	*	I	E	E	E	W									
$R_4 \leftarrow (B)$						F	*	*	D	I	E	E	E	W									
$R_5 \leftarrow (C)$									F	D	I	E	E	E	W								
$R_6 \leftarrow (R_4) \times (R_5)$										F	D	*	*	I	E	E	E	W					
$A \leftarrow (R_6)$											F	*	*	D	*	*	I	E	E	E	W		

Software static scheduling and scoreboarding reduced the # of cycles to 16 or 18 - using six registers

## Dynamic Scheduling - Tomasulo's Algorithm

Assume: Two LOADs can overlap

Minimum register code	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8
$R_1 \leftarrow (Y)$	F	D	I	E	E	W												
$R_2 \leftarrow (Z)$		F	D	I	E	E	W											
$R_3 \leftarrow (R_1) + (R_2)$			F	D	I	*	*	E	E	W								
$X \leftarrow (R_3)$				F	D	I	*	*	*	*	E	E	W					
$R_1 \leftarrow (B)$					F	D	I	E	E	W								
$R_2 \leftarrow (C)$						F	D	I	E	E	W							
$R_3 \leftarrow (R_1) \times (R_2)$							F	D	I	*	*	E	E	W				
$A \leftarrow (R_3)$								F	D	I	*	*	*	*	E	E	W	

- 18 vs. 22 cycles even with only 3 registers
- No structural hazards, only data hazards

Source: J. Smith, IEEE Computer, July 1989.

ECE568/Koren Part.6.7

## Reservation Station Components

**Op:** Operation to perform in the unit (e.g., ADD)

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

- Store buffers has V<sub>j</sub> field, result to be stored

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

- Note: Q<sub>j</sub>, Q<sub>k</sub>=0 => ready
- Store buffers only have Q<sub>j</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.

ECE568/Koren Part.6.8

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Three Phases of Tomasulo's Algorithm

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

If reservation station free (no structural hazard), control issues instruction & gets operands (renames registers)

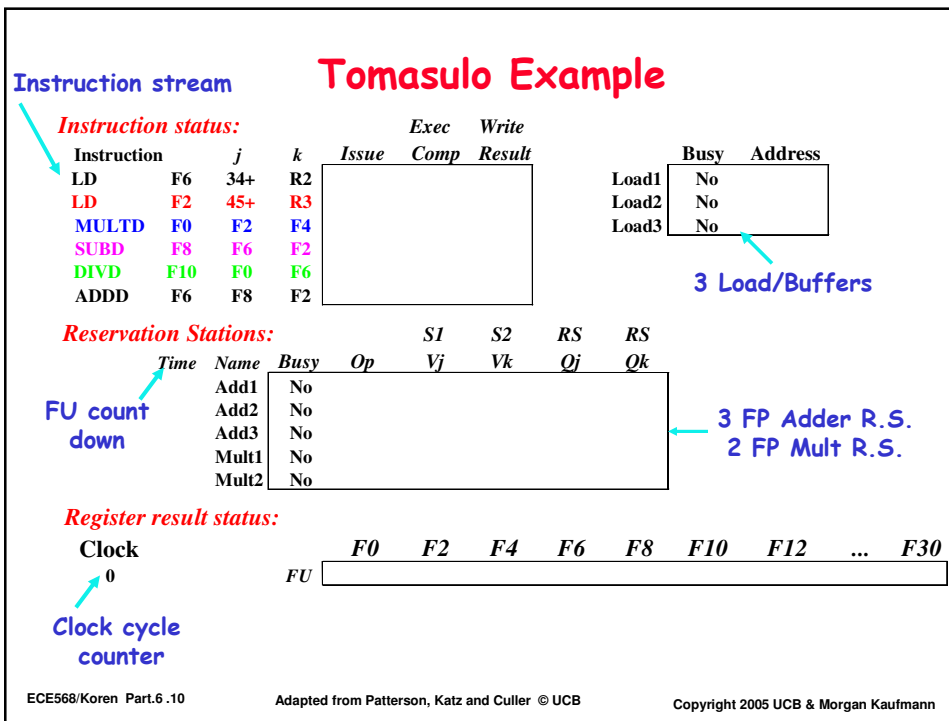
### 2. Execute—operate on operands (EX)

When both operands ready then execute; if not ready, watch Common Data Bus for result

### 3. Write result—finish execution (WB)

Write on Common Data Bus to all awaiting units; mark reservation station available

- ◆ 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
  - Does the broadcast
  - Write if matches expected Functional Unit (produces result)
- ◆ Example - speed of operations:  
2 clocks for FI.pt.ADD,SUB and LOAD; 10 for MULT; 40 clocks for DIV



## Tomasulo Example Cycle 1

**Instruction status:**

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

**Reservation Stations:**

Time	Name	Busy	Op	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					

ECE568/Koren Part.6 .11

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Tomasulo Example Cycle 2

**Instruction status:**

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

**Reservation Stations:**

Time	Name	Busy	Op	S1 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: Can have multiple loads outstanding**

ECE568/Koren Part.6 .12

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Tomasulo Example Cycle 3

**Instruction status:**

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

**Reservation Stations:**

Time	Name	Busy	Op	S1	S2	RS	RS
				Vj	Vk	Qj	Qk
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	Yes	MULTD		R(F4)	Load2	
	Mult2	No					

**Register result status:**

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

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

ECE568/Koren Part.6.13

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Tomasulo Example Cycle 4

**Instruction status:**

Instruction	j	k	Exec		Write	Busy	Address
			Issue	Comp	Result		
LD	F6	34+	R2	1	3	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	S1	S2	RS	RS
				Vj	Vk	Qj	Qk
	Add1	Yes	SUBD	M(A1)		Load2	
	Add2	No					
	Add3	No					
	Mult1	Yes	MULTD		R(F4)	Load2	
	Mult2	No					

**Register result status:**

Clock	F0	F2	F4	F6	F8	F10	F12	...	F30
4	FU	Mult1	Load2		M(A1)	Add1			

- Load2 completing; what is waiting for Load2?

ECE568/Koren Part.6.14

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Tomasulo Example Cycle 5

**Instruction status:**

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

**Reservation Stations:**

Time	Name	Busy	Op	S1	S2	RS	RS
				Vj	Vk	Qj	Qk
2	Add1	Yes	SUBD	M(A1)	M(A2)		
	Add2	No					
	Add3	No					
10	Mult1	Yes	MULTD	M(A2)	R(F4)		
	Mult2	Yes	DIVD	R(F6)	Mult1		

**Register result status:**

Clock	F0	F2	F4	F6	F8	F10	F12	...	F30
5	FU	Mult1	M(A2)		Add1	Mult2			

- Timer starts down for Add1, Mult1

ECE568/Koren Part.6 .15

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Tomasulo Example Cycle 6

**Instruction status:**

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

**Reservation Stations:**

Time	Name	Busy	Op	S1	S2	RS	RS
				Vj	Vk	Qj	Qk
1	Add1	Yes	SUBD	M(A1)	M(A2)		
	Add2	Yes	ADDD		R(F2)	Add1	
	Add3	No					
9	Mult1	Yes	MULTD	M(A2)	R(F4)		
	Mult2	Yes	DIVD	R(F6)	Mult1		

**Register result status:**

Clock	F0	F2	F4	F6	F8	F10	F12	...	F30
6	FU	Mult1		Add2	Add1	Mult2			

- Issue ADDD here despite name dependency on F6?

ECE568/Koren Part.6 .16

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Tomasulo Example Cycle 7

**Instruction status:**

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

**Reservation Stations:**

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
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		R(F6)	Mult1	

**Register result status:**

Clock	F0	F2	F4	F6	F8	F10	F12	...	F30
7	FU								
	Mult1			Add2	Add1	Mult2			

- Add1 (SUBD) completing; what is waiting for it?

## Tomasulo Example Cycle 8

**Instruction status:**

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

**Reservation Stations:**

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
	Add1	No					
2	Add2	Yes	ADDD	(M-M)	M(A2)		
	Add3	No					
7	Mult1	Yes	MULTD	M(A2)	R(F4)		
	Mult2	Yes	DIVD		R(F6)	Mult1	

**Register result status:**

Clock	F0	F2	F4	F6	F8	F10	F12	...	F30
8	FU								
	Mult1			Add2	(M-M)	Mult2			

## Tomasulo Example Cycle 11

**Instruction status:**

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

**Reservation Stations:**

Time	Name	Busy	Op	Vj	Vk	Qj	Qk
	Add1	No					
	Add2	No					
	Add3	No					
4	Mult1	Yes	MULTD	M(A2)	R(F4)		
	Mult2	Yes	DIVD		R(F6)	Mult1	

**Register result status:**

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

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

ECE568/Koren Part.6 .19

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Tomasulo Example Cycle 15

**Instruction status:**

Instruction	j	k	Exec			Write	Busy	Address
			Issue	Comp	Result			
LD	F6	34+	R2	1	3	4	Load1	No
LD	F2	45+	R3	2	4	5	Load2	No
MULTD	F0	F2	F4	3	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	Vj	Vk	Qj	Qk
	Add1	No					
	Add2	No					
	Add3	No					
0	Mult1	Yes	MULTD	M(A2)	R(F4)		
	Mult2	Yes	DIVD		R(F6)	Mult1	

**Register result status:**

Clock	F0	F2	F4	F6	F8	F10	F12	...	F30
15				Mult1			Mult2		

- Mult1 (MULTD) completing; what is waiting for it?

ECE568/Koren Part.6 .20

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Tomasulo Example Cycle 16

**Instruction status:**

Instruction	j	k	Exec			Write	Busy	Address
			Issue	Comp	Result			
LD	F6	34+	R2	1	3	4	Load1	No
LD	F2	45+	R3	2	4	5	Load2	No
MULTD	F0	F2	F4	3	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	Vj	Vk	Qj	Qk
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	No					
40	Mult2	Yes	DIVD	M*F4	R(F6)		

**Register result status:**

Clock	F0	F2	F4	F6	F8	F10	F12	...	F30
16	FU	M*F4				Mult2			

- Just waiting for Mult2 (DIVD) to complete

## Tomasulo Example Cycle 56

**Instruction status:**

Instruction	j	k	Exec			Write	Busy	Address
			Issue	Comp	Result			
LD	F6	34+	R2	1	3	4	Load1	No
LD	F2	45+	R3	2	4	5	Load2	No
MULTD	F0	F2	F4	3	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	Vj	Vk	Qj	Qk
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	No					
0	Mult2	Yes	DIVD	M*F4	R(F6)		

**Register result status:**

Clock	F0	F2	F4	F6	F8	F10	F12	...	F30
56	FU					Mult2			

- Mult2 (DIVD) is completing; what is waiting for it?

## Tomasulo Example Cycle 57

**Instruction status:**

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

**Reservation Stations:**

Time	Name	Busy	Op	S1		S2		RS		RS	
				Vj	Vk	Qj	Qk				
Add1		No									
Add2		No									
Add3		No									
Mult1		No									
Mult2		Yes	DIVD	M*F4	R(F6)						

**Register result status:**

Clock	F0	F2	F4	F6	F8	F10	F12	...	F30
56	FU <span style="border: 1px solid black; padding: 2px;">Result</span>								

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

## Tomasulo Drawbacks

- ◆ High Complexity
- ◆ Many associative stores (CDB) at high speed resulting limit on performance by Common Data Bus
  - Each CDB must go to multiple functional units
    - ⇒ high wiring density, high capacitance
  - Number of functional units that can complete per cycle limited to one!
    - » Multiple CDBs ⇒ more FU logic for parallel stores
- ◆ Non-precise interrupts!
  - We will address this later

## Tomasulo Loop Example

```

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

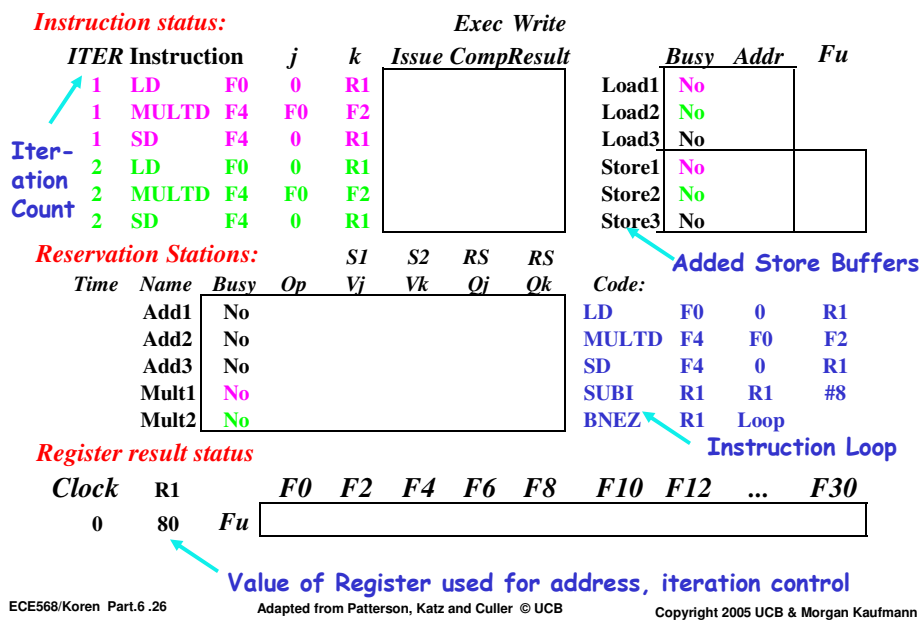
- ◆ This time assume Multiply takes 4 clocks
- ◆ Assume 1st load takes 8 clocks (L1 cache miss), 2nd load takes 4 clocks (hit)
- ◆ To be clear, will show clocks for SUBI, BNEZ
  - Reality: integer instructions ahead of FI. Point Instructions
- ◆ Show 2 iterations of loop

ECE568/Koren Part.6 .25

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Loop Example



ECE568/Koren Part.6 .26

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Loop Example Cycle 1

**Instruction status:**

						Exec Write			
ITER	Instruction	j	k	Issue	CompResult	Busy	Addr	Fu	
1	LD	F0	0	R1	1	Load1	Yes	80	
						Load2	No		
						Load3	No		
						Store1	No		
						Store2	No		
						Store3	No		

**Reservation Stations:**

Time	Name	Busy	Op	S1	S2	RS	RS	Code:
				Vj	Vk	Oj	Ok	
Add1	No							LD F0 0 R1 ←
Add2	No							MULTD F4 F0 F2
Add3	No							SD F4 0 R1
Mult1	No							SUBI R1 R1 #8
Mult2	No							BNEZ R1 Loop

**Register result status**

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

ECE568/Koren Part.6 .27

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Loop Example Cycle 2

**Instruction status:**

						Exec Write			
ITER	Instruction	j	k	Issue	CompResult	Busy	Addr	Fu	
1	LD	F0	0	R1	1	Load1	Yes	80	
1	MULTD	F4	F0	F2	2	Load2	No		
						Load3	No		
						Store1	No		
						Store2	No		
						Store3	No		

**Reservation Stations:**

Time	Name	Busy	Op	S1	S2	RS	RS	Code:
				Vj	Vk	Oj	Ok	
Add1	No							LD F0 0 R1 ←
Add2	No							MULTD F4 F0 F2
Add3	No							SD F4 0 R1
Mult1	Yes	Multd		R(F2)	Load1			SUBI R1 R1 #8
Mult2	No							BNEZ R1 Loop

**Register result status**

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

ECE568/Koren Part.6 .28

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Loop Example Cycle 3

**Instruction status:**

ITER	Instruction	j	k	Issue	Comp	Result	Busy	Addr	Fu
1	LD	F0	0	R1	1		Load1	Yes	80
1	MULTD	F4	F0	F2	2		Load2	No	
1	SD	F4	0	R1	3		Load3	No	
							Store1	Yes	80
							Store2	No	
							Store3	No	

**Reservation Stations:**

Time	Name	Busy	Op	S1	S2	RS	RS	Code:
				Vj	Vk	Oj	Ok	
	Add1	No						LD F0 0 R1
	Add2	No						MULTD F4 F0 F2
	Add3	No						SD F4 0 R1
	Mult1	Yes	Multd			R(F2)	Load1	SUBI R1 R1 #8
	Mult2	No						BNEZ R1 Loop

**Register result status**

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

◆ **Implicit renaming sets up data flow graph**

ECE568/Koren Part.6 .29

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Loop Example Cycle 4

**Instruction status:**

ITER	Instruction	j	k	Issue	Comp	Result	Busy	Addr	Fu
1	LD	F0	0	R1	1		Load1	Yes	80
1	MULTD	F4	F0	F2	2		Load2	No	
1	SD	F4	0	R1	3		Load3	No	
							Store1	Yes	80
							Store2	No	
							Store3	No	

**Reservation Stations:**

Time	Name	Busy	Op	S1	S2	RS	RS	Code:
				Vj	Vk	Oj	Ok	
	Add1	No						LD F0 0 R1
	Add2	No						MULTD F4 F0 F2
	Add3	No						SD F4 0 R1
	Mult1	Yes	Multd			R(F2)	Load1	SUBI R1 R1 #8
	Mult2	No						BNEZ R1 Loop

**Register result status**

Clock	R1	F0	F2	F4	F6	F8	F10	F12	...	F30
4	80	Fu	Load1	Mult1						

◆ **Dispatching SUBI Instruction (not in FP queue)**

ECE568/Koren Part.6 .30

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Loop Example Cycle 5

**Instruction status:**

ITER Instruction				j			k			Issue CompResult			Exec Write		
Issue	Op	Rj	Rk	Issue	Comp	Result	Busy	Addr	Fu						
1	LD	F0	0	R1	1		Load1	Yes	80						
1	MULTD	F4	F0	F2	2		Load2	No							
1	SD	F4	0	R1	3		Load3	No							
							Store1	Yes	80	Mult1					
							Store2	No							
							Store3	No							

**Reservation Stations:**

Time	Name	Busy	Op	S1	S2	RS	RS	Code:
				Vj	Vk	Oj	Ok	
	Add1	No						LD F0 0 R1
	Add2	No						MULTD F4 F0 F2
	Add3	No						SD F4 0 R1
	Mult1	Yes	Multd			R(F2)	Load1	SUBI R1 R1 #8
	Mult2	No						BNEZ R1 Loop

**Register result status**

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

◆ **And, BNEZ instruction (not in FP queue)**

ECE568/Koren Part.6 .31

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Loop Example Cycle 6

**Instruction status:**

ITER Instruction				j			k			Issue CompResult			Exec Write		
Issue	Op	Rj	Rk	Issue	Comp	Result	Busy	Addr	Fu						
1	LD	F0	0	R1	1		Load1	Yes	80						
1	MULTD	F4	F0	F2	2		Load2	Yes	72						
1	SD	F4	0	R1	3		Load3	No							
2	LD	F0	0	R1	6		Store1	Yes	80	Mult1					
							Store2	No							
							Store3	No							

**Reservation Stations:**

Time	Name	Busy	Op	S1	S2	RS	RS	Code:
				Vj	Vk	Oj	Ok	
	Add1	No						LD F0 0 R1
	Add2	No						MULTD F4 F0 F2
	Add3	No						SD F4 0 R1
	Mult1	Yes	Multd			R(F2)	Load1	SUBI R1 R1 #8
	Mult2	No						BNEZ R1 Loop

**Register result status**

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

◆ **F0 never sees Load from location 80; no WAW**

ECE568/Koren Part.6 .32

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Loop Example Cycle 7

**Instruction status:**

				Exec Write				
ITER	Instruction	j	k	Issue	CompResult	Busy	Addr	Fu
1	LD	F0	0	R1	1	Load1	Yes	80
1	MULTD	F4	F0	F2	2	Load2	Yes	72
1	SD	F4	0	R1	3	Load3	No	
2	LD	F0	0	R1	6	Store1	Yes	80
2	MULTD	F4	F0	F2	7	Store2	No	
						Store3	No	

**Reservation Stations:**

Time	Name	Busy	Op	Vj	Vk	Oj	Ok	Code:
Add1	No							LD F0 0 R1
Add2	No							MULTD F4 F0 F2 ←
Add3	No							SD F4 0 R1
Mult1	Yes	Multd				R(F2)	Load1	SUBI R1 R1 #8
Mult2	Yes	Multd				R(F2)	Load2	BNEZ R1 Loop

**Register result status**

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

- ◆ Register file completely detached from computation
- ◆ First and Second iteration completely overlapped

ECE568/Koren Part.6 .33

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Loop Example Cycle 8

**Instruction status:**

				Exec Write				
ITER	Instruction	j	k	Issue	CompResult	Busy	Addr	Fu
1	LD	F0	0	R1	1	Load1	Yes	80
1	MULTD	F4	F0	F2	2	Load2	Yes	72
1	SD	F4	0	R1	3	Load3	No	
2	LD	F0	0	R1	6	Store1	Yes	80
2	MULTD	F4	F0	F2	7	Store2	Yes	72
2	SD	F4	0	R1	8	Store3	No	

**Reservation Stations:**

Time	Name	Busy	Op	Vj	Vk	Oj	Ok	Code:
Add1	No							LD F0 0 R1
Add2	No							MULTD F4 F0 F2 ←
Add3	No							SD F4 0 R1
Mult1	Yes	Multd				R(F2)	Load1	SUBI R1 R1 #8
Mult2	Yes	Multd				R(F2)	Load2	BNEZ R1 Loop

**Register result status**

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

ECE568/Koren Part.6 .34

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Loop Example Cycle 9

**Instruction status:**

ITER	Instruction	j	k	Issue	Comp	Result	Busy	Addr	Fu
1	LD	F0	0	R1	1	9	Load1	Yes 80	
1	MULTD	F4	F0	F2	2		Load2	Yes 72	
1	SD	F4	0	R1	3		Load3	No	
2	LD	F0	0	R1	6		Store1	Yes 80	Mult1
2	MULTD	F4	F0	F2	7		Store2	Yes 72	Mult2
2	SD	F4	0	R1	8		Store3	No	

**Reservation Stations:**

Time	Name	Busy	Op	Vj	Vk	Oj	Ok	Code:
Add1	No							LD F0 0 R1
Add2	No							MULTD F4 F0 F2
Add3	No							SD F4 0 R1
Mult1	Yes	Multd		R(F2)	Load1			SUBI R1 R1 #8
Mult2	Yes	Multd		R(F2)	Load2			BNEZ R1 Loop

**Register result status**

Clock	R1	F0	F2	F4	F6	F8	F10	F12	...	F30
9	72	Fu	Load2	Mult2						

◆ Load1 completing: who is waiting? Note: Dispatching SUBI

ECE568/Koren Part.6 .35

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Loop Example Cycle 10

**Instruction status:**

ITER	Instruction	j	k	Issue	Comp	Result	Busy	Addr	Fu
1	LD	F0	0	R1	1	9	Load1	No	
1	MULTD	F4	F0	F2	2		Load2	Yes 72	
1	SD	F4	0	R1	3		Load3	No	
2	LD	F0	0	R1	6	10	Store1	Yes 80	Mult1
2	MULTD	F4	F0	F2	7		Store2	Yes 72	Mult2
2	SD	F4	0	R1	8		Store3	No	

**Reservation Stations:**

Time	Name	Busy	Op	Vj	Vk	Oj	Ok	Code:
Add1	No							LD F0 0 R1
Add2	No							MULTD F4 F0 F2
Add3	No							SD F4 0 R1
4	Mult1	Yes	Multd	M[80]	R(F2)			SUBI R1 R1 #8
	Mult2	Yes	Multd		R(F2)	Load2		BNEZ R1 Loop

**Register result status**

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

◆ Load2 completing: who is waiting? Note: Dispatching BNEZ

ECE568/Koren Part.6 .36

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Loop Example Cycle 11

**Instruction status:**

ITER	Instruction	j	k	Issue	Comp	Result	Busy	Addr	Fu
1	LD	F0	0	R1	1	9	10		
1	MULTD	F4	F0	F2	2				
1	SD	F4	0	R1	3				
2	LD	F0	0	R1	6	10	11		
2	MULTD	F4	F0	F2	7				
2	SD	F4	0	R1	8				

	Load1	Load2	Load3	Store1	Store2	Store3
10	No	No	No	No	No	No
11	No	No	Yes	Yes	Yes	No
12	No	No	No	No	No	No

**Reservation Stations:**

Time	Name	Busy	Op	Vj	Vk	Oj	Ok	Code:
10	Add1	No						LD F0 0 R1
10	Add2	No						MULTD F4 F0 F2
10	Add3	No						SD F4 0 R1
11	Mult1	Yes	Multd	M[80]	R(F2)			SUBI R1 R1 #8
11	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	Fu	Load3		Mult2					

◆ Next load in sequence

ECE568/Koren Part.6 .37

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Loop Example Cycle 12

**Instruction status:**

ITER	Instruction	j	k	Issue	Comp	Result	Busy	Addr	Fu
1	LD	F0	0	R1	1	9	10		
1	MULTD	F4	F0	F2	2				
1	SD	F4	0	R1	3				
2	LD	F0	0	R1	6	10	11		
2	MULTD	F4	F0	F2	7				
2	SD	F4	0	R1	8				

	Load1	Load2	Load3	Store1	Store2	Store3
10	No	No	No	No	No	No
11	No	No	Yes	Yes	Yes	No
12	No	No	No	No	No	No

**Reservation Stations:**

Time	Name	Busy	Op	Vj	Vk	Oj	Ok	Code:
11	Add1	No						LD F0 0 R1
11	Add2	No						MULTD F4 F0 F2
11	Add3	No						SD F4 0 R1
12	Mult1	Yes	Multd	M[80]	R(F2)			SUBI R1 R1 #8
12	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	Fu	Load3		Mult2					

◆ Why not issue third multiply?

ECE568/Koren Part.6 .38

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Loop Example Cycle 13

**Instruction status:**

				Exec Write					
ITER	Instruction	j	k	Issue	Comp	Result	Busy	Addr	Fu
1	LD	F0	0	R1	1	9	10	Load1	No
1	MULTD	F4	F0	F2	2			Load2	No
1	SD	F4	0	R1	3			Load3	Yes 64
2	LD	F0	0	R1	6	10	11	Store1	Yes 80 Mult1
2	MULTD	F4	F0	F2	7			Store2	Yes 72 Mult2
2	SD	F4	0	R1	8			Store3	No

**Reservation Stations:**

Time	Name	Busy	Op	Vj	Vk	Oj	Ok	Code:
	Add1	No						LD F0 0 R1
	Add2	No						MULTD F4 F0 F2 ←
	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	Fu	Load3	Mult2						

♦ Why not issue third store?

ECE568/Koren Part.6 .39

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Loop Example Cycle 14

**Instruction status:**

				Exec Write					
ITER	Instruction	j	k	Issue	Comp	Result	Busy	Addr	Fu
1	LD	F0	0	R1	1	9	10	Load1	No
1	MULTD	F4	F0	F2	2	14		Load2	No
1	SD	F4	0	R1	3			Load3	Yes 64
2	LD	F0	0	R1	6	10	11	Store1	Yes 80 Mult1
2	MULTD	F4	F0	F2	7			Store2	Yes 72 Mult2
2	SD	F4	0	R1	8			Store3	No

**Reservation Stations:**

Time	Name	Busy	Op	Vj	Vk	Oj	Ok	Code:
	Add1	No						LD F0 0 R1
	Add2	No						MULTD F4 F0 F2 ←
	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	Fu	Load3	Mult2						

♦ Mult1 completing. Who is waiting?

ECE568/Koren Part.6 .40

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Loop Example Cycle 15

**Instruction status:**

				Exec Write					
ITER	Instruction	j	k	Issue	Comp	Result	Busy	Addr	Fu
1	LD	F0	0	R1	1	9	10	Load1	No
1	MULTD	F4	F0	F2	2	14	15	Load2	No
1	SD	F4	0	R1	3			Load3	Yes 64
2	LD	F0	0	R1	6	10	11	Store1	Yes 80 (80)*R1
2	MULTD	F4	F0	F2	7	15		Store2	Yes 72 Mult2
2	SD	F4	0	R1	8			Store3	No

**Reservation Stations:**

Time	Name	Busy	Op	Vj	Vk	Oj	Qk	Code:
	Add1	No						LD F0 0 R1
	Add2	No						MULTD F4 F0 F2 ←
	Add3	No						SD F4 0 R1
	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	Fu	Load3	Mult2						

♦ **Mult2 completing. Who is waiting?**

ECE568/Koren Part.6 .41

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Loop Example Cycle 16

**Instruction status:**

				Exec Write					
ITER	Instruction	j	k	Issue	Comp	Result	Busy	Addr	Fu
1	LD	F0	0	R1	1	9	10	Load1	No
1	MULTD	F4	F0	F2	2	14	15	Load2	No
1	SD	F4	0	R1	3			Load3	Yes 64
2	LD	F0	0	R1	6	10	11	Store1	Yes 80 (80)*R1
2	MULTD	F4	F0	F2	7	15	16	Store2	Yes 72 (72)*R2
2	SD	F4	0	R1	8			Store3	No

**Reservation Stations:**

Time	Name	Busy	Op	Vj	Vk	Oj	Qk	Code:
	Add1	No						LD F0 0 R1
	Add2	No						MULTD F4 F0 F2 ←
	Add3	No						SD F4 0 R1
4	Mult1	Yes	Multd		R(F2)	Load3		SUBI R1 R1 #8
	Mult2	No						BNEZ R1 Loop

**Register result status**

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

ECE568/Koren Part.6 .42

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Loop Example Cycle 17

**Instruction status:**

				Exec Write					
ITER	Instruction	j	k	Issue	Comp	Result	Busy	Addr	Fu
1	LD	F0	0	R1	1	9	10	Load1	No
1	MULTD	F4	F0	F2	2	14	15	Load2	No
1	SD	F4	0	R1	3			Load3	Yes 64
2	LD	F0	0	R1	6	10	11	Store1	Yes 80 [80]*R2
2	MULTD	F4	F0	F2	7	15	16	Store2	Yes 72 [72]*R2
2	SD	F4	0	R1	8			Store3	Yes 64 <b>Multi1</b>

**Reservation Stations:**

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Oj	RS Ok	Code:
Add1	No							LD F0 0 R1
Add2	No							MULTD F4 F0 F2
Add3	No							SD F4 0 R1 ←
Multi1	Yes	Multi			R(F2)	Load3		SUBI R1 R1 #8
Multi2	No							BNEZ R1 Loop

**Register result status**

Clock	R1	F0	F2	F4	F6	F8	F10	F12	...	F30
17	64	Fu	Load3		Multi1					

ECE568/Koren Part.6 .43

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Loop Example Cycle 18

**Instruction status:**

				Exec Write					
ITER	Instruction	j	k	Issue	Comp	Result	Busy	Addr	Fu
1	LD	F0	0	R1	1	9	10	Load1	No
1	MULTD	F4	F0	F2	2	14	15	Load2	No
1	SD	F4	0	R1	3	<b>18</b>		Load3	Yes 64
2	LD	F0	0	R1	6	10	11	Store1	Yes 80 [80]*R2
2	MULTD	F4	F0	F2	7	15	16	Store2	Yes 72 [72]*R2
2	SD	F4	0	R1	8			Store3	Yes 64 Multi1

**Reservation Stations:**

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Oj	RS Ok	Code:
Add1	No							LD F0 0 R1
Add2	No							MULTD F4 F0 F2
Add3	No							SD F4 0 R1 ←
Multi1	Yes	Multi			R(F2)	Load3		SUBI R1 R1 #8
Multi2	No							BNEZ R1 Loop

**Register result status**

Clock	R1	F0	F2	F4	F6	F8	F10	F12	...	F30
18	64	Fu	Load3		Multi1					

ECE568/Koren Part.6 .44

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Loop Example Cycle 19

**Instruction status:**

ITER	Instruction	j	k	Issue	Comp	Result	Busy	Addr	Fu
1	LD	F0	0	R1	1	9	10	Load1	No
1	MULTD	F4	F0	F2	2	14	15	Load2	No
1	SD	F4	0	R1	3	18	19	Load3	Yes 64
2	LD	F0	0	R1	6	10	11	Store1	No
2	MULTD	F4	F0	F2	7	15	16	Store2	Yes 72 [72]*R2
2	SD	F4	0	R1	8	19		Store3	Yes 64 Mult1

**Reservation Stations:**

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

**Register result status**

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

ECE568/Koren Part.6 .45

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Loop Example Cycle 20

**Instruction status:**

ITER	Instruction	j	k	Issue	Comp	Result	Busy	Addr	Fu
1	LD	F0	0	R1	1	9	10	Load1	Yes 56
1	MULTD	F4	F0	F2	2	14	15	Load2	No
1	SD	F4	0	R1	3	18	19	Load3	Yes 64
2	LD	F0	0	R1	6	10	11	Store1	No
2	MULTD	F4	F0	F2	7	15	16	Store2	No
2	SD	F4	0	R1	8	19	20	Store3	Yes 64 Mult1

**Reservation Stations:**

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

**Register result status**

Clock	R1	F0	F2	F4	F6	F8	F10	F12	...	F30
20	56	Fu	Load1			Mult1				

- In-order issue, out-of-order execution & completion across loop iterations

ECE568/Koren Part.6 .46

Adapted from Patterson, Katz and Culler © UCB

Copyright 2005 UCB & Morgan Kaufmann

## Why can Tomasulo overlap iterations of loops?

- ◆ **Register renaming**
  - Multiple iterations use different physical destinations for registers (dynamic loop unrolling)
- ◆ **Reservation stations**
  - Permit instruction issue to advance past integer control flow operations
  - Also buffer old values of registers - totally avoiding the WAR stall that we saw in the scoreboard
- ◆ **Other perspective: Tomasulo building data flow dependency graph on the fly**

## Summary: Tomasulo's scheme offers two major advantages

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