Lecture 6: Tomasulo Algorithm – register renaming and tag-based dependence check

Tomasulo design, big example

Scoreboarding Merit

Enable out-of-order execution and reduces stall cycles caused by RAW dependences

- Use register result status table to detect WAW and RAW dependence between a newly fetched instruction and pending instructions
- Record dependence into the FU status of scoreboard (in forms of Qj, Qk and Rj, Rk)
- Keep an instruction waiting if it has dependences with pending instructions
- When an inst writes result, wake up dependent instructions by matching the inst's FU index with Qj and Qk of *every* FU status entry

Good Dynamic Scheduling Needs More

Better handling of WAR and WAW dependences

- Scoreboarding: (1) Stalls issuing when WAW is detected; (2) Delay writing results when WAR is detected
- Is it necessary to *enforce* WAR and WAW dependences?

Better handling of structure hazards

- Why stall pipeline when two instructions go to the same FU? (Particular a problem for memory/integer instructions)

Better pipeline efficiency

- Two extra cycles between the EXs of two dependent instructions – Need data forwarding

More ILP beyond a basic block

- Need speculative execution, branch predictions, and dynamic memory disambiguation

Tomasulo Algorithm

History:

- ◈1966: scoreboarding in CDC6600, implementing limited dynamic scheduling
- ◈Three years later: Tomasulo in IBM 360/91, introducing register renaming and reservation station
- ◈Now appearing in todays Dec Alpha, SGI MIPS, SUN UltraSparc, Intel Pentium, IBM PowerPC and others in different forms

What Tomasulo Provides

Better handling of WAR and WAW dependences

- Use register renaming to remove WAR and WAW dependences – No stalls or delays anymore

Better handling of structure hazards

- Multiple reservation stations per FU – instruction is assigned to a reservation station

Better pipeline efficiency

- One extra (instead of two) between EXs of two dependent instructions

Dynamic memory disambiguation

- Enforce dependence between stores and loads

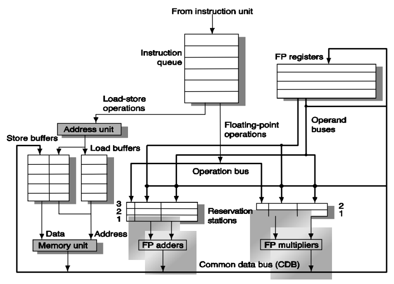
Distributed scheduling logic

- Register and RS status no longer centralized

Three Stages of Tomasulo Algorithm

1. Issue—get instruction from FP Op Queue
   - Condition: a free RS at the required FU
   - Actions: (1) decode the instruction; (2) allocate a RS; (3) do source register renaming; (4) do dest register renaming; (5) read register file; (6) dispatch the decoded and renamed instruction to the RS entry
2. Execution—operate on operands (EX)
   - Condition: At a given FU, At lease one instruction is ready
   - Action: select a ready instruction and send it to the FU
3. Write result—finish execution (WB)
   - Condition: At a given FU, some instruction finishes FU execution
   - Actions: (1) the FU writes the result onto the Common Data (store inst writes to memory instead); (2) the tag (RS index) is broadcast to all other RS to wake up potential waiting instructions; (3) update register status; (4) de-allocate the RS

Tomasulo Organization



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
- Note: No ready flags as in Scoreboard; Qj,Qk=0 => ready
- Store buffers only have Qi for RS producing result

A – to hold memory address for load and store instructoins
Busy—Indicates reservation station or FU is busy

How are dependences represented in reservation station?

Adapted from UCB CS252 S98B

Register Renaming in Tomasulo

Register Result Status Table: Indicates which RS will write each register, if one exists. Blank when no pending instructions that will write that register.
- Is very similar to the one in scoreboarding
- Performs register renaming to remove name (WAR and WAW) dependences

Modern term: register alias table (sometimes register renaming table)

Register Renaming in Tomasulo

Source Renaming: rename source register to mapped RS index (if renamed)
Destination renaming: record the renaming from the destination register to allocated RS index; used in future renaming
Example:
LD F2,45(R3) => LD (LD1),45(R3)
MULT F0, F2, F4 => MULT (MULT1), (LD1), F4

Key difference from that in scoreboarding: use RS index as tag of the operands

Renaming Implementation

0

F0	F2	F4	
-	-	-	...

LD F2,45(R3) => LD (LD1), 45(R3)

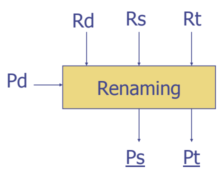
1

F0	F2	F4	
-	LD1	-	...

MULT F0, F2, F4 => MULT (MULT1), (LD1), F4

2

F0	F2	F4	
Mult1	LD1	-	...



For every inst:
1. Translate source registers if renamed
2. Record the renaming for dest register

Selection of Instruction for Execution

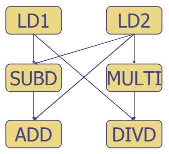
- Only "ready" instructions can join the competition
- There is a select logic to select instructions for FU execution
  - Some policy may be used, e.g. age based
- Non-ready instructions can be "waken up" during writeback of its parent inst

Writeback and 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 source index (tag)
  - Does the broadcast to every instruction in the fly
- Child instructions do tag matching and update their ready bits and value fields (if the tag matches theirs)
- Register update involves tag matching

Code Example

```
LD     F6,34(R2)
LD     F2,45(R3)
MULTI  F0,F2,F4
SUBD   F8,F6,F2
DIVD   F10,F0,F6
ADD    F6,F8,F2
```



Operation latencies: load/store 2 cycles,
Add/sub 2 cycles, Mult 10 cycles, divide 40 cycle

What to Observe

1. Whether some instructions can be issued
   => pay attention to (1) RS (or load/store buffer) allocation (decode, RS allocation, source register renaming, dispatch); (2) change to register status (dest renaming)
2. Whether some instruction can be selected for execution (for every FU)
   => Pay attention to instruction status change; the inst will finish in a given number of cycle
3. Whether some instruction is finishing execution
   => Pay attention to instruction status change; the inst may write its result the next cycle
4. Whether some instruction is writing result
   => Pay attention to (1) wakeup of the dependent instructions; (2) register status change; (3) RS de-allocation

Tomasulo Example Cycle 0

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
0	FU									

Tomasulo Example Cycle 1

Instruction status

Instruction		j	k	Issue	Execution complete	Write Result		Busy	Address
LD	F6	34+	R2	1			Load1	Yes	34+R2
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					
	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

Instruction		j	k	Issue	Execution complete	Write Result		Busy	Address
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

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk
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
2	FU		Load2		Load1					

Tomasulo Example Cycle 3

Instruction status					Execution	Write			
Instruction		j	k	Issue	complete	Result		Busy	Address
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				S1	S2	RS for j	RS for k
Time	Name	Busy	Op	Vj	Vk	Qj	Qk
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
- Load1 completing; what is waiting for Load1?

Tomasulo Example Cycle 4

Instruction status					Execution	Write			
Instruction		j	k	Issue	complete	Result		Busy	Address
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				S1	S2	RS for j	RS for k
Time	Name	Busy	Op	Vj	Vk	Qj	Qk
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		F0	F2	F4	F6	F8	F10	F12	...	F30
4	FU	Mult1	Load2		M(34+R2)	Add1				

- Load2 completing; what is waiting for it?

Tomasulo Example Cycle 5

Instruction status					Execution	Write			
Instruction		j	k	Issue	complete	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				S1	S2	RS for j	RS for k
Time	Name	Busy	Op	Vj	Vk	Qj	Qk
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					Execution	Write			
Instruction		j	k	Issue	complete	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				S1	S2	RS for j	RS for k
Time	Name	Busy	Op	Vj	Vk	Qj	Qk
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			

Tomasulo Example Cycle 7

Instruction status					Execution	Write			
Instruction		j	k	Issue	complete	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				
DIVD	F10	F0	F6	5					
ADDD	F6	F8	F2	6					

Reservation Stations				S1	S2	RS for j	RS for k
Time	Name	Busy	Op	Vj	Vk	Qj	Qk
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										
Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
7	FU	Mult1	M(45+R3)		Add2	Add1	Mult2			

- Add1 completing; what is waiting for it?

Tomasulo Example Cycle 8

Instruction status					Execution	Write			
Instruction		j	k	Issue	complete	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				S1	S2	RS for j	RS for k
Time	Name	Busy	Op	Vj	Vk	Qj	Qk
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		F0	F2	F4	F6	F8	F10	F12	...	F30
8	FU	Mult1	M(45+R3)		Add2	M()-M()	Mult2			

Tomasulo Example Cycle 9

Instruction status					Execution	Write						
Instruction		j	k	Issue	complete	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					S1	S2	RS for j	RS for k				
	Time	Name	Busy	Op	Vj	Vk	Qj	Qk				
	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				F0	F2	F4	F6	F8	F10	F12	...	F30
9			FU	Mult1	M(45+R3)		Add2	M()–M()	Mult2			

Tomasulo Example Cycle 10

Instruction status					Execution	Write						
Instruction		j	k	Issue	complete	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					S1	S2	RS for j	RS for k				
	Time	Name	Busy	Op	Vj	Vk	Qj	Qk				
	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					Execution	Write						
Instruction		j	k	Issue	complete	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					S1	S2	RS for j	RS for k				
	Time	Name	Busy	Op	Vj	Vk	Qj	Qk				
	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 statu												
Clock				F0	F2	F4	F6	F8	F10	F12	...	F30
11			FU	Mult1	M(45+R3)		(M-M)+M()	M()ŠM()	Mult2			

- Write result of ADDD here vs. scoreboard?

Tomasulo Example Cycle 12

Instruction status					Execution	Write						
Instruction		j	k	Issue	complete	Result			Busy	Address		
LD	F6	34+	R2	1	3	4		Load1	No			
LD	F2	45+	R3	2	4	5		Load2	No			
MUL	F0	F2	F4	3				Load3	No			
SUB	F8	F6	F2	4	7	8						
DIVD	F10	F0	F6	5								
ADD	F6	F8	F2	6	10	11						
Reservation Stations					S1	S2	RS for j	RS for k				
	Time	Name	Bus	Op	Vj	Vk	Qj	Qk				
	0	Add1	No									
	0	Add2	No									
	0	Add3	No									
	3	Mult1	Yes	MULT	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			

Tomasulo Example Cycle 13

Instruction status					Execution	Write						
Instruction		j	k	Issue	complete	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					S1	S2	RS for j	RS for k				
	Time	Name	Busy	Op	Vj	Vk	Qj	Qk				
	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					Execution	Write						
Instruction		j	k	Issue	complete	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					S1	S2	RS for j	RS for k				
	Time	Name	Busy	Op	Vj	Vk	Qj	Qk				
	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

Instruction		j	k	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			Load3	No	
SUBD	F8	F6	F2	4	7	8				
DIVD	F10	F0	F6	5						
ADDD	F6	F8	F2	6	10	11				

Reservation Stations

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk
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

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

Instruction		j	k	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	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk
0	Add1	No					
0	Add2	No					
	Add3	No					
0	Mult1	No					
40	Mult2	Yes	DIVD	M*F4	M(34+R2)		

Register result status

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
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		j	k	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	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk
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

Instruction		j	k	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	56					
ADDD	F6	F8	F2	6	10	11				

Reservation Stations

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk
0	Add1	No					
0	Add2	No					
	Add3	No					
0	Mult1	No					
0	Mult2	Yes	DIVD	M*F4	M(34+R2)		

Register result status

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

Instruction		j	k	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	56	57				
ADDD	F6	F8	F2	6	10	11				

Reservation Stations

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS for j Qj	RS for k Qk
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-oder issue,
  out-of-order execution, completion

The Use of Tag

In Tomasulo, RS index is used as tag (tag is a modern term).

- Tag is a unique identifier for a pending register result
- Tag decouples the register result from the architectural register specifier
- Tag removes WAR and WAW dependences without changing RAW dependences

It is not necessary to use RS index as tag!

Machine Correctness

E(D,P) = E(S,P) if

- E(D,P) and E(S,P) execute the same set of instructions
- For any inst *i*, *i* receives the outputs in E(D,P) of its parents in E(S,P)
- In E(D,P) any register or memory word receives the output of inst *j*, where *j* is the last instruction writes to the register or memory word in E(S,P)

RAW and WAW dependences are not necessary for a correct execution!

Tomasulo Summary

- Reservations stations:
  - Increases effective register number
  - Distributes scheduling logic
- Register renaming: Avoids WAR and WAW dependence
- Tag + Data broadcasting for waking up child instructions
- Pros: can be effectively combined with speculative execution
- Cons: CDB broadcasting adds one-cycle delay (addressed in modern instruction scheduling)