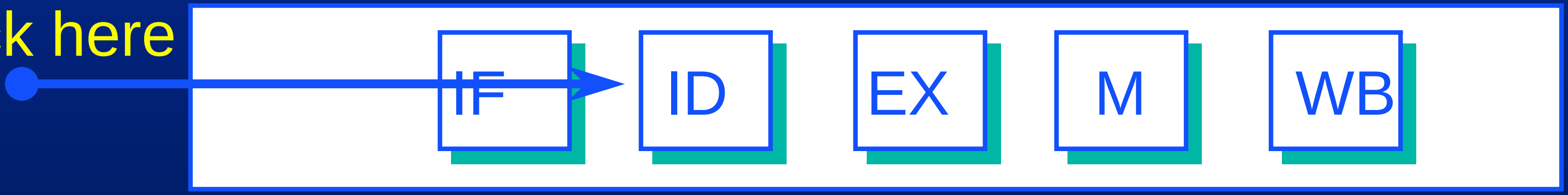
Dynamic instruction scheduling

Key idea: allow subsequent independent instructions to proceed

```
DIVD    F0,F2,F4        ; takes long time
ADDD    F10,F0,F8       ; stalls waiting for F0
SUBD    F12,F8,F13      ; Let this instr. bypass the ADDD
```

- Enables out-of-order execution => out-of-order completion

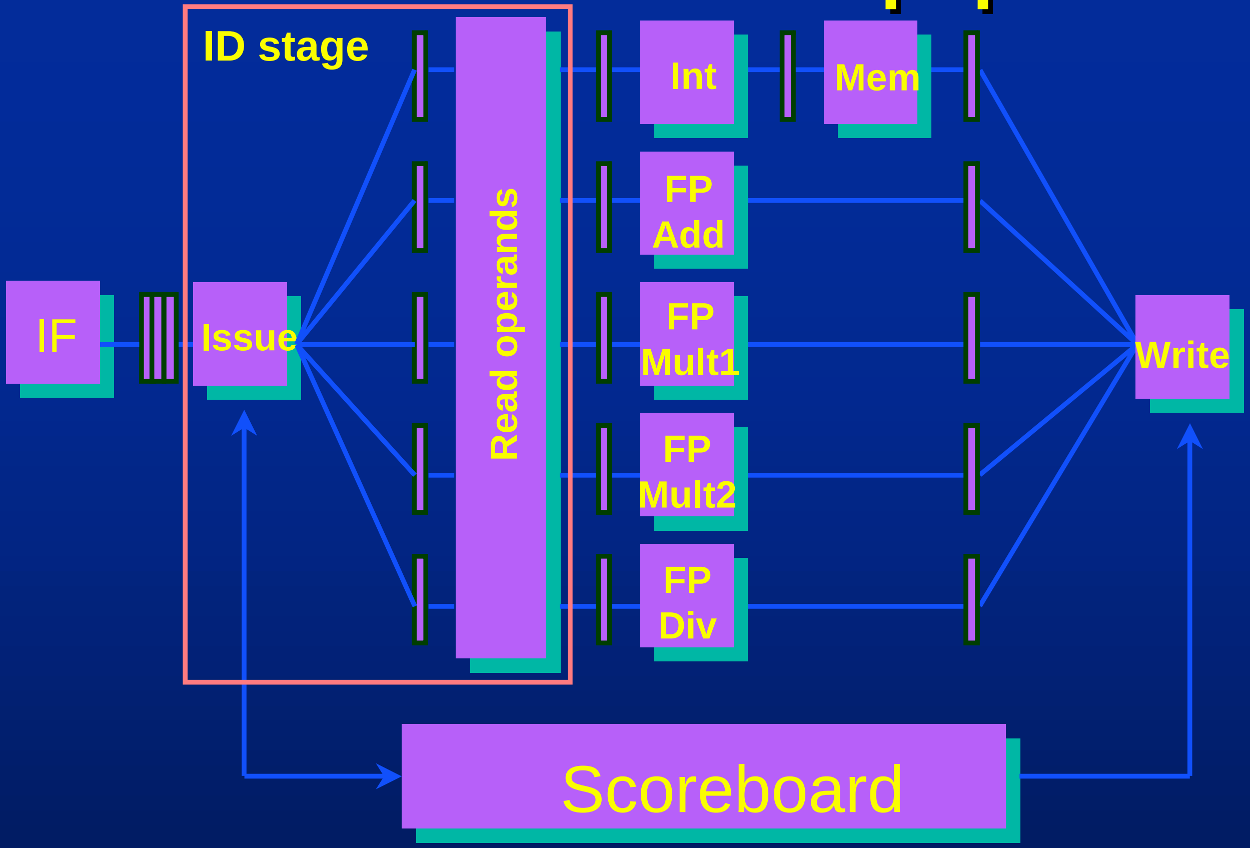
Instr. gets stuck here



Two historical schemes used in recent machines:

- Scoreboard dates back to CDC 6600 in 1963
- Tomasulo in IBM 360/91 in 1967

Scoreboard pipeline



- **Issue**: Decode and check for structural hazards
- **Read operands**: wait until no data hazard, then read operands
- All data hazards are handled by the scoreboard mechanism

Scoreboard complications

Out-of-order completion => WAR, WAW hazards

- WAR: instruction is stalled in the WB stage until a previous instruction has read the operand

- WAW: instruction is stalled in the Issue stage until a previous instruction has written its result

Scoreboard keeps track of dependencies and state of operations

Scoreboard functionality

Issue: Instruction is issued when:

- **No structural hazard for a functional unit**
- **No WAW with an instruction in execution**

Read: Instruction reads operands when they become available (RAW)

EX: normal execution

Write: Instruction writes when all previous instructions have read this operand

The scoreboard is updated when an instruction proceeds to a new stage

Data structures in the scoreboard

1. ***Instruction status***—keeps track of in which stage an instruction is.

2. ***Functional unit status***—Indicates the state of the functional unit (FU). 9 fields for each FU:
   - Busy: Indicates whether the unit is busy or not
   - Op: Operation to perform in the unit (e.g. add or sub)
   - Fi: Destination register name
   - Fj, Fk: Source register names
   - Qj, Qk: Name of functional unit producing regs Fj, Fk
   - Rj, Rk: Flags indicating when Fj and Fk are ready

3. ***Register result status***—Indicates which functional unit will write to each register, if any.

Scoreboard example

Instruction status

Instruction		j	k	***Issue***	***Read ops***	***Exec. compl.***	***Write result***
LD	F6	34+	R2				
LD	F2	45+	R3				
MULTD	F0	F2	F4				
SUBD	F8	F6	F2				
DIVD	F10	F0	F6				
ADDD	F6	F8	F2				

Functional unit status

				dest	src 1	src 2	FUsrc1	FUsrc2	Fj?	Fk?
Time	Name	***Busy***	***Op***	***Fi***	***Fj***	***Fk***	***Qj***	***Qk***	***Rj***	***Rk***
	Integer	No								
	Mult1	No								
	Mult2	No								
	Add	No								
	Divide	No								

Register result status

	F0	***F2***	***F4***	***F6***	***F8***	***F10***	***...***	***F30***
FU								

Clock: 0

Detailed Scoreboard Pipeline Control

Instruction status	*Wait until*	*Bookkeeping for FU*
Issue	Not busy(FU) & Not Result(\`D´)	Busy$\leftarrow$Yes; Fi$\leftarrow$\`D´; Fj$\leftarrow$\`Src1´; Fk$\leftarrow$\`Src2´; Qj$\leftarrow$Result(\`Src1´); Rj$\leftarrow$ not Qj Qk$\leftarrow$Result(\`Src2´); Rj$\leftarrow$ not Qj
Read operands	Rj = Yes & Rk = Yes	Rj $\leftarrow$ No; Rk $\leftarrow$ No;
Execution complete	Functional unit done	
Write result	For all functional units f$\neq$FU: (Fj(f) $\neq$ Fi(FU) or Rj(f) = No) AND (Fk(f) $\neq$ Fi(FU) or Rk(f) = No)	For all functional units f$\neq$FU: if Qj(f) = FU then Rj(f) $\leftarrow$ Yes; if Qk(f) = FU then Rk(f) $\leftarrow$ Yes; Result(Fi(FU)) $\leftarrow$ 0; Busy(FU) •$\leftarrow$ No;

Scoreboard example, cycle 1

Instruction status

Instruction		j	k	*Issue*	***Read ops***	***Exec. compl.***	***Write result***
LD	F6	34+	R2	1			
LD	F2	45+	R3				
MULTD	F0	F2	F4				
SUBD	F8	F6	F2				
DIVD	F10	F0	F6				
ADDD	F6	F8	F2				

Functional unit status

Time	Name			dest	src 1	src 2	FUsrc1	FUsrc2	Fj?	Fk?
		Busy	***Op***	***Fi***	***Fj***	***Fk***	***Qj***	***Qk***	***Rj***	***Rk***
	Integer	Yes	Load	F6		R2				Yes
	Mult1	No								
	Mult2	No								
	Add	No								
	Divide	No								

Register result status

	F0	***F2***	***F4***	***F6***	***F8***	***F10***	***...***	***F30***
FU				Integer				

Clock: 1

Scoreboard example, cycle 2

Instruction status

Instruction		j	k	Issue	Read ops	Exec. compl.	Write result
LD	F6	34+	R2	1	2		
LD	F2	45+	R3				
MULTD	F0	F2	F4				
SUBD	F8	F6	F2				
DIVD	F10	F0	F6				
ADDD	F6	F8	F2				

- Issue 2nd load?

Functional unit status

				dest	src 1	src 2	FUsrc1	FUsrc2	Fj?	Fk?
Time	Name	Busy	Op	Fi	Fj	Fk	Qj	Qk	Rj	Rk
	Integer	Yes	Load	F6		R2				No
	Mult1	No								
	Mult2	No								
	Add	No								
	Divide	No								

Register result status

	F0	F2	F4	F6	F8	F10	...	F30
FU				Integer				

Clock: 2

Scoreboard example, cycle 3

Instruction status

Instruction		j	k	*Issue*	*Read ops*	*Exec. compl.*	*Write result*
LD	F6	34+	R2	1	2	3	
LD	F2	45+	R3				
MULTD	F0	F2	F4				
SUBD	F8	F6	F2				
DIVD	F10	F0	F6				
ADDD	F6	F8	F2				

- Issue MULT?

Functional unit status

Time	Name	*Busy*	*Op*	dest *Fi*	src 1 *Fj*	src 2 *Fk*	FUsrc1 *Qj*	FUsrc2 *Qk*	Fj? *Rj*	Fk? *Rk*
	Integer	Yes	Load	F6		R2				No
	Mult1	No								
	Mult2	No								
	Add	No								
	Divide	No								

Register result status

	F0	*F2*	*F4*	*F6*	*F8*	*F10*	*...*	*F30*
FU				Integer				

Clock: 3

Scoreboard example, cycle 4

Instruction status

					Read	***Exec.***	***Write***
Instruction		j	k	***Issue***	***ops***	***compl.***	***result***
LD	F6	34+	R2	1	2	3	4
LD	F2	45+	R3				
MULTD	F0	F2	F4				
SUBD	F8	F6	F2				
DIVD	F10	F0	F6				
ADDD	F6	F8	F2				

Functional unit status

				dest	src 1	src 2	FUsrc1	FUsrc2	Fj?	Fk?
Time	Name	***Busy***	***Op***	***Fi***	***Fj***	***Fk***	***Qj***	***Qk***	***Rj***	***Rk***
	Integer	No								
	Mult1	No								
	Mult2	No								
	Add	No								
	Divide	No								

Register result status

	F0	***F2***	***F4***	***F6***	***F8***	***F10***	***...***	***F30***
FU				-				

Clock: 4

Scoreboard example, cycle 5

Instruction status

Instruction		j	k	*Issue*	*Read ops*	*Exec. compl.*	*Write result*
LD	F6	34+	R2	1	2	3	4
LD	F2	45+	R3	5			
MULTD	F0	F2	F4				
SUBD	F8	F6	F2				
DIVD	F10	F0	F6				
ADDD	F6	F8	F2				

Functional unit status

Time	Name	*Busy*	*Op*	dest *Fi*	src 1 *Fj*	src 2 *Fk*	FUsrc1 *Qj*	FUsrc2 *Qk*	Fj? *Rj*	Fk? *Rk*
	Integer	Yes	Load	F2		R3				Yes
	Mult1	No								
	Mult2	No								
	Add	No								
	Divide	No								

Register result status

	F0	*F2*	*F4*	*F6*	*F8*	*F10*	*...*	*F30*
FU		Integer						

Clock: 5

Scoreboard example, cycle 6

Instruction status

Instruction		j	k	Issue	Read ops	Exec. compl.	Write result
LD	F6	34+	R2	1	2	3	4
LD	F2	45+	R3	5	6		
MULTD	F0	F2	F4	6			
SUBD	F8	F6	F2				
DIVD	F10	F0	F6				
ADDD	F6	F8	F2				

Functional unit status

Time	Name	Busy	Op	dest Fi	src 1 Fj	src 2 Fk	FUsrc1 Qj	FUsrc2 Qk	Fj? Rj	Fk? Rk
	Integer	Yes	Load	F2		R3				No
	Mult1	Yes	Mult	F0	F2	F4	Integer		No	Yes
	Mult2	No								
	Add	No								
	Divide	No								

Register result status

	F0	F2	F4	F6	F8	F10	...	F30
FU	Mult1	Integer						

Clock: 6

Scoreboard example, cycle 7

Instruction status

Instruction		j	k	*Issue*	*Read ops*	*Exec. compl.*	*Write result*
LD	F6	34+	R2	1	2	3	4
LD	F2	45+	R3	5	6	7	
MULTD	F0	F2	F4	6			
SUBD	F8	F6	F2	7			
DIVD	F10	F0	F6				
ADDD	F6	F8	F2				

Functional unit status

				dest	src 1	src 2	FUsrc1	FUsrc2	Fj?	Fk?
Time	Name	*Busy*	*Op*	*Fi*	*Fj*	*Fk*	*Qj*	*Qk*	*Rj*	*Rk*
	Integer	Yes	Load	F2		R3				No
	Mult1	Yes	Mult	F0	F2	F4	Integer		No	Yes
	Mult2	No								
	Add	Yes	Sub	F8	F6	F2		Integer	Yes	No
	Divide	No								

Register result status

	F0	*F2*	*F4*	*F6*	*F8*	*F10*	*...*	*F30*
FU	Mult1	Integer			Add			

Clock: 7

Scoreboard example, cycle 8a

Instruction status

Instruction		j	k	*Issue*	*Read ops*	*Exec. compl.*	*Write result*
LD	F6	34+	R2	1	2	3	4
LD	F2	45+	R3	5	6	7	
MULTD	F0	F2	F4	6			
SUBD	F8	F6	F2	7			
DIVD	F10	F0	F6	8			
ADDD	F6	F8	F2				

Functional unit status

Time	Name	*Busy*	*Op*	dest *Fi*	src 1 *Fj*	src 2 *Fk*	FUsrc1 *Qj*	FUsrc2 *Qk*	Fj? *Rj*	Fk? *Rk*
	Integer	Yes	Load	F2		R3				No
	Mult1	Yes	Mult	F0	F2	F4	Integer		No	Yes
	Mult2	No								
	Add	Yes	Sub	F8	F6	F2		Integer	Yes	No
	Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

Register result status

	F0	*F2*	*F4*	*F6*	*F8*	*F10*	*...*	*F30*
FU	Mult1	Integer			Add	Divide		

Clock: 8

Scoreboard example, cycle 8

Instruction status

Instruction		j	k	Issue	Read ops	Exec. compl.	Write result
LD	F6	34+	R2	1	2	3	4
LD	F2	45+	R3	5	6	7	8
MULTD	F0	F2	F4	6			
SUBD	F8	F6	F2	7			
DIVD	F10	F0	F6	8			
ADDD	F6	F8	F2				

Functional unit status

				dest	src 1	src 2	FUsrc1	FUsrc2	Fj?	Fk?
Time	Name	***Busy***	***Op***	***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

	F0	***F2***	***F4***	***F6***	***F8***	***F10***	***...***	***F30***
FU	Mult1	-			Add	Divide		

Clock: 8

Scoreboard example, cycle 9

Instruction status

Instruction		j	k	*Issue*	*Read ops*	*Exec. compl.*	*Write result*
LD	F6	34+	R2	1	2	3	4
LD	F2	45+	R3	5	6	7	8
MULTD	F0	F2	F4	6	9		
SUBD	F8	F6	F2	7	9		
DIVD	F10	F0	F6	8			
ADDD	F6	F8	F2				

- Read operands for MULT & SUB
- Issue ADDD?

Functional unit status

				dest	src 1	src 2	FUsrc1	FUsrc2	Fj?	Fk?
Time	Name	*Busy*	*Op*	*Fi*	*Fj*	*Fk*	*Qj*	*Qk*	*Rj*	*Rk*
	Integer	No								
10	*Mult1*	Yes	Mult	F0	F2	F4			No	No
	Mult2	No								
2	*Add*	Yes	Sub	F8	F6	F2			No	No
	Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

Register result status

	F0	*F2*	*F4*	*F6*	*F8*	*F10*	*...*	*F30*
FU	Mult1				Add	Divide		

Clock: 9

Scoreboard example, cycle 11

Instruction status

Instruction		j	k	*Issue*	*Read ops*	*Exec. compl.*	*Write result*
LD	F6	34+	R2	1	2	3	4
LD	F2	45+	R3	5	6	7	8
MULTD	F0	F2	F4	6	9		
SUBD	F8	F6	F2	7	9	11	
DIVD	F10	F0	F6	8			
ADDD	F6	F8	F2				

- SUBD completes execution

Functional unit status

Time	Name	*Busy*	*Op*	dest *Fi*	src 1 *Fj*	src 2 *Fk*	FUsrc1 *Qj*	FUsrc2 *Qk*	Fj? *Rj*	Fk? *Rk*
	Integer	No								
8	*Mult1*	Yes	Mult	F0	F2	F4			No	No
	Mult2	No								
0	*Add*	Yes	Sub	F8	F6	F2			No	No
	Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

Register result status

	F0	*F2*	*F4*	*F6*	*F8*	*F10*	*...*	*F30*
FU	Mult1				Add	Divide		

Clock: 11

Scoreboard example, cycle 12

Instruction status

Instruction		j	k	*Issue*	*Read ops*	*Exec. compl.*	*Write result*
LD	F6	34+	R2	1	2	3	4
LD	F2	45+	R3	5	6	7	8
MULTD	F0	F2	F4	6	9		
SUBD	F8	F6	F2	7	9	11	12
DIVD	F10	F0	F6	8			
ADDD	F6	F8	F2				

- Read operands for DIVD?

Functional unit status

				dest	src 1	src 2	FUsrc1	FUsrc2	Fj?	Fk?
Time	Name	*Busy*	*Op*	*Fi*	*Fj*	*Fk*	*Qj*	*Qk*	*Rj*	*Rk*
	Integer	No								
7	*Mult1*	Yes	Mult	F0	F2	F4			No	No
	Mult2	No								
	Add	No							-	-
	Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

Register result status

	F0	*F2*	*F4*	*F6*	*F8*	*F10*	*...*	*F30*
FU	Mult1				-	Divide		

Clock: 12

Scoreboard example, cycle 13

- Issue ADDD

Instruction status

Instruction		j	k	Issue	Read ops	Exec. compl.	Write result
LD	F6	34+	R2	1	2	3	4
LD	F2	45+	R3	5	6	7	8
MULTD	F0	F2	F4	6	9		
SUBD	F8	F6	F2	7	9	11	12
DIVD	F10	F0	F6	8			
ADDD	F6	F8	F2	13			

Functional unit status

				dest	src 1	src 2	FUsrc1	FUsrc2	Fj?	Fk?
Time	Name	*Busy*	*Op*	*Fi*	*Fj*	*Fk*	*Qj*	*Qk*	*Rj*	*Rk*
	Integer	No								
6	*Mult1*	Yes	Mult	F0	F2	F4			No	No
	Mult2	No								
	Add	Yes	Add	F6	F8	F2			Yes	Yes
	Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

Register result status

	F0	*F2*	*F4*	*F6*	*F8*	*F10*	...	*F30*
FU	Mult1			Add		Divide		

Clock: 13

Scoreboard example, cycle 14

Instruction status

Instruction	j	k		*Issue*	*Read ops*	*Exec. compl.*	*Write result*
LD	F6	34+	R2	1	2	3	4
LD	F2	45+	R3	5	6	7	8
MULTD	F0	F2	F4	6	9		
SUBD	F8	F6	F2	7	9	11	12
DIVD	F10	F0	F6	8			
ADDD	F6	F8	F2	13	14		

Functional unit status

Time	Name	*Busy*	*Op*	dest *Fi*	src 1 *Fj*	src 2 *Fk*	FUsrc1 *Qj*	FUsrc2 *Qk*	Fj? *Rj*	Fk? *Rk*
	Integer	No								
5	*Mult1*	Yes	Mult	F0	F2	F4			No	No
	Mult2	No								
2	*Add*	Yes	Add	F6	F8	F2			No	No
	Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

Register result status

	F0	*F2*	*F4*	*F6*	*F8*	*F10*	...	*F30*
FU	Mult1			Add		Divide		

Clock: 14

Scoreboard example, cycle 16

Instruction status

Instruction	j	k		*Issue*	*Read ops*	*Exec. compl.*	*Write result*
LD	F6	34+	R2	1	2	3	4
LD	F2	45+	R3	5	6	7	8
MULTD	F0	F2	F4	6	9		
SUBD	F8	F6	F2	7	9	11	12
DIVD	F10	F0	F6	8			
ADDD	F6	F8	F2	13	14	16	

- Can ADDD write result?

Functional unit status

Time	Name	*Busy*	*Op*	dest *Fi*	src 1 *Fj*	src 2 *Fk*	FUsrc1 *Qj*	FUsrc2 *Qk*	Fj? *Rj*	Fk? *Rk*
	Integer	No								
3	*Mult1*	Yes	Mult	F0	F2	F4			No	No
	Mult2	No								
0	*Add*	Yes	Add	F6	F8	F2			No	No
	Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

Register result status

	F0	*F2*	*F4*	*F6*	*F8*	*F10*	*...*	*F30*
FU	Mult1			Add		Divide		

Clock: 16

Scoreboard example, cycle 17

Instruction status

Instruction	j	k		*Issue*	*Read ops*	*Exec. compl.*	*Write result*
LD	F6	34+	R2	1	2	3	4
LD	F2	45+	R3	5	6	7	8
MULTD	F0	F2	F4	6	9		
SUBD	F8	F6	F2	7	9	11	12
DIVD	F10	F0	F6	8			
ADDD	F6	F8	F2	13	14	16	

- ADDD stalls, waiting for DIVD to read F6
- Resolves a WAR hazard!

Functional unit status

				dest	src 1	src 2	FUsrc1	FUsrc2	Fj?	Fk?
Time	Name	*Busy*	*Op*	*Fi*	*Fj*	*Fk*	*Qj*	*Qk*	*Rj*	*Rk*
	Integer	No								
2	***Mult1***	Yes	Mult	F0	F2	F4			No	No
	Mult2	No								
	Add	Yes	Add	F6	F8	F2			No	No
	Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

Register result status

	F0	*F2*	*F4*	*F6*	*F8*	*F10*	*...*	*F30*
FU	Mult1			Add		Divide		

Clock: 17

Scoreboard example, cycle 19

Instruction status

Instruction		j	k	*Issue*	*Read ops*	*Exec. compl.*	*Write result*
LD	F6	34+	R2	1	2	3	4
LD	F2	45+	R3	5	6	7	8
MULTD	F0	F2	F4	6	9	19	
SUBD	F8	F6	F2	7	9	11	12
DIVD	F10	F0	F6	8			
ADDD	F6	F8	F2	13	14	16	

Functional unit status

				dest	src 1	src 2	FUsrc1	FUsrc2	Fj?	Fk?
Time	Name	*Busy*	*Op*	*Fi*	*Fj*	*Fk*	*Qj*	*Qk*	*Rj*	*Rk*
	Integer	No								
0	*Mult1*	Yes	Mult	F0	F2	F4			No	No
	Mult2	No								
	Add	Yes	Add	F6	F8	F2			No	No
	Divide	Yes	Div	F10	F0	F6	Mult1		No	Yes

Register result status

	F0	*F2*	*F4*	*F6*	*F8*	*F10*	*...*	*F30*
FU	Mult1			Add		Divide		

Clock: 19

Scoreboard example, cycle 20

Instruction status

Instruction	j	k		*Issue*	*Read ops*	*Exec. compl.*	*Write result*
LD	F6	34+	R2	1	2	3	4
LD	F2	45+	R3	5	6	7	8
MULTD	F0	F2	F4	6	9	19	20
SUBD	F8	F6	F2	7	9	11	12
DIVD	F10	F0	F6	8			
ADDD	F6	F8	F2	13	14	16	

Functional unit status

Time	Name	*Busy*	*Op*	dest *Fi*	src 1 *Fj*	src 2 *Fk*	FUsrc1 *Qj*	FUsrc2 *Qk*	Fj? *Rj*	Fk? *Rk*
	Integer	No								
	Mult1	No								
	Mult2	No								
	Add	Yes	Add	F6	F8	F2			No	No
	Divide	Yes	Div	F10	F0	F6	-		Yes	Yes

Register result status

	F0	*F2*	*F4*	*F6*	*F8*	*F10*	*...*	*F30*
FU	-			Add		Divide		

Clock: 20

Scoreboard example, cycle 21

Instruction status

Instruction		j	k	Issue	Read ops	Exec. compl.	Write result
LD	F6	34+	R2	1	2	3	4
LD	F2	45+	R3	5	6	7	8
MULTD	F0	F2	F4	6	9	19	20
SUBD	F8	F6	F2	7	9	11	12
DIVD	F10	F0	F6	8	21		
ADDD	F6	F8	F2	13	14	16	

Functional unit status

				dest	src 1	src 2	FUsrc1	FUsrc2	Fj?	Fk?
Time	Name	Busy	Op	Fi	Fj	Fk	Qj	Qk	Rj	Rk
	Integer	No								
	Mult1	No								
	Mult2	No								
	Add	Yes	Add	F6	F8	F2			No	No
	Divide	Yes	Div	F10	F0	F6			No	No

Register result status

	F0	F2	F4	F6	F8	F10	...	F30
FU				Add		Divide		

Clock: 21

Scoreboard example, cycle 22

Instruction status

Instruction		j	k	Issue	Read ops	Exec. compl.	Write result
LD	F6	34+	R2	1	2	3	4
LD	F2	45+	R3	5	6	7	8
MULTD	F0	F2	F4	6	9	19	20
SUBD	F8	F6	F2	7	9	11	12
DIVD	F10	F0	F6	8	21		
ADDD	F6	F8	F2	13	14	16	22

- Now ADDD can safely write its result in F6

Functional unit status

				dest	src 1	src 2	FUsrc1	FUsrc2	Fj?	Fk?
Time	Name	Busy	Op	Fi	Fj	Fk	Qj	Qk	Rj	Rk
	Integer	No								
	Mult1	No								
	Mult2	No								
	Add	No								
40	Divide	Yes	Div	F10	F0	F6			No	No

Register result status

	F0	F2	F4	F6	F8	F10	...	F30
FU				-		Divide		

Clock: 22

Scoreboard example, cycle 61

Instruction status

Instruction		j	k	Issue	Read ops	Exec. compl.	Write result
LD	F6	34+	R2	1	2	3	4
LD	F2	45+	R3	5	6	7	8
MULTD	F0	F2	F4	6	9	19	20
SUBD	F8	F6	F2	7	9	11	12
DIVD	F10	F0	F6	8	21	61	
ADDD	F6	F8	F2	13	14	16	22

Functional unit status

				dest	src 1	src 2	FUsrc1	FUsrc2	Fj?	Fk?
Time	Name	Busy	Op	Fi	Fj	Fk	Qj	Qk	Rj	Rk
	Integer	No								
	Mult1	No								
	Mult2	No								
	Add	No								
0	Divide	Yes	Div	F10	F0	F6			No	No

Register result status

	F0	F2	F4	F6	F8	F10	...	F30
FU						Divide		

Clock: 61

Scoreboard example, cycle 62

Instruction status

Instruction		j	k	Issue	Read ops	Exec. compl.	Write result
LD	F6	34+	R2	1	2	3	4
LD	F2	45+	R3	5	6	7	8
MULTD	F0	F2	F4	6	9	19	20
SUBD	F8	F6	F2	7	9	11	12
DIVD	F10	F0	F6	8	21	61	62
ADDD	F6	F8	F2	13	14	16	22

Functional unit status

Time	Name	Busy	Op	dest Fi	src 1 Fj	src 2 Fk	FUsrc1 Qj	FUsrc2 Qk	Fj? Rj	Fk? Rk
	Integer	No								
	Mult1	No								
	Mult2	No								
	Add	No								
	Divide	No								

Register result status

	F0	F2	F4	F6	F8	F10	...	F30
FU						-		

Clock: 62

Limitations with scoreboards

The scoreboard technique is limited by:

- Number of scoreboard entries (*window size*)
- Number and types of functional units
- Number of ports to the register bank
- Hazards caused by name dependencies

Tomasulo's algorithm addresses the last two limitations

Tomasulo’s Algorithm

In IBM 360/91, 4 years after the CDC 6600

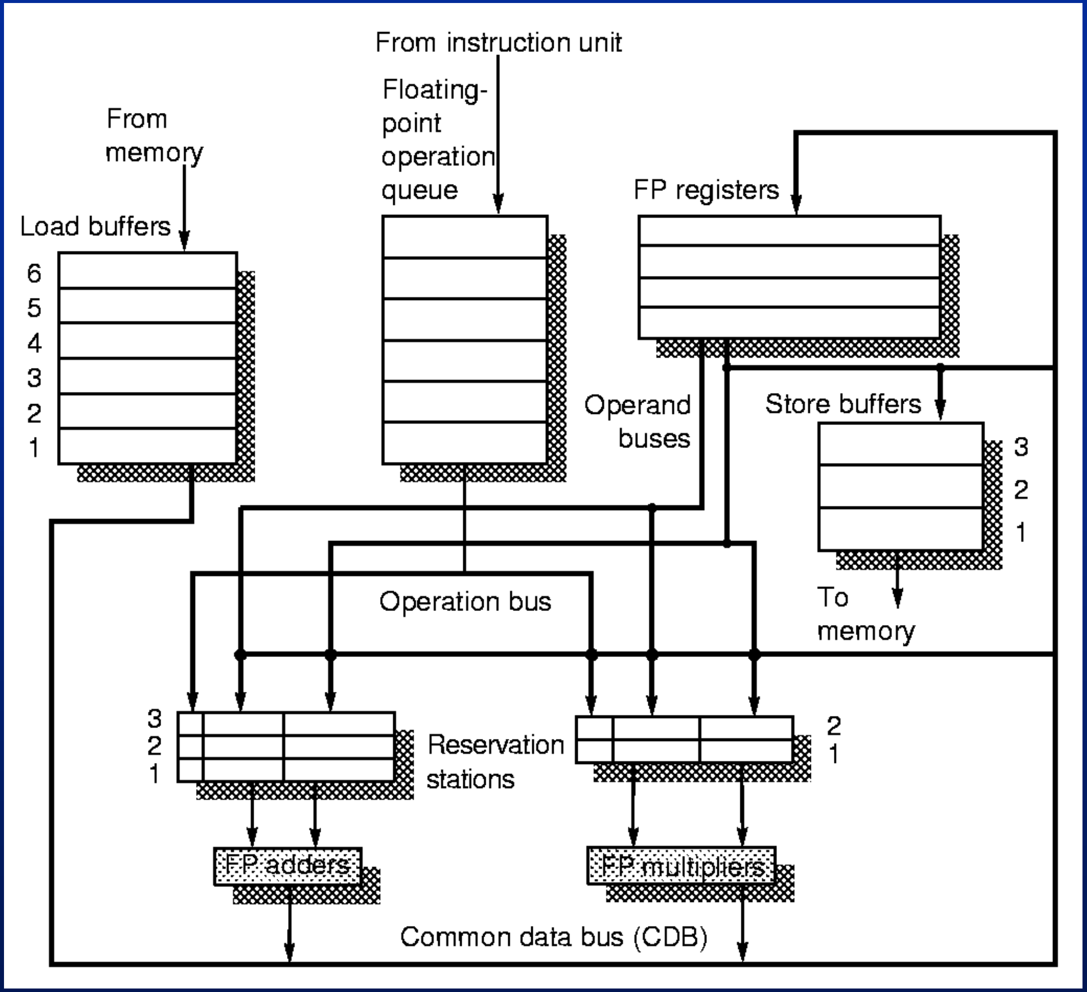
Goal: High performance without compiler support

Differences between Tomasulo & Scoreboard:

- Control & Buffers *distributed* with FUs (called ***reservation stations***) vs. centralised in Scoreboard
- Register names in instructions replaced by pointers to reservation station buffer (***HW register renaming***)
- Common Data Bus broadcasts results to all FUs
- Loads and Stores treated as FUs as well

This technique has been adopted in many recent machines (e.g. PowerPC)

Hardware Organization



Three stages of Tomasulo's Alg.

1. ***Issue***—get instruction from FP Op Queue
   - Issue if no structural hazard for a reservation station
2. ***Execution***—operate on operands (EX)
   - Execute when both operands are available;
     if not ready, watch Common Data Bus (CDB) for result
3. ***Write result***—finish execution (WB)
   - Write on CDB to all awaiting functional units;
     mark reservation station available

- Normal bus: data + destination
- Common Data Bus: data + ***source*** (snooping)

Tomasulo example, cycle 0

Instruction status

Instruction		j	k	*Issue*	*Exec. compl.*	*Write result*
LD	F6	34+	R2			
LD	F2	45+	R3			
MULTD	F0	F2	F4			
SUBD	F8	F6	F2			
DIVD	F10	F0	F6			
ADDD	F6	F8	F2			

Load buffers

	Busy	*Address*
Load1	No	
Load2	No	
Load3	No	

Functional unit status

				src 1	src 2	RS for j	RS for k
Time	Name	*Busy*	*Op*	*Vj*	*Vk*	*Qj*	*Qk*
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	No					
	Mult2	No					

Register result status

	F0	*F2*	*F4*	*F6*	*F8*	*F10*	*...*	*F30*
FU								

Clock: 0

Tomasulo example, cycle 1

Instruction status

Instruction		j	k	Issue	Exec. compl.	Write result
LD	F6	34+	R2	1		
LD	F2	45+	R3			
MULTD	F0	F2	F4			
SUBD	F8	F6	F2			
DIVD	F10	F0	F6			
ADDD	F6	F8	F2			

Load buffers

Time		Busy	Address
	Load1	Yes	R2+32
	Load2	No	
	Load3	No	

Functional unit status

				src 1	src 2	RS for j	RS for k
Time	Name	Busy	Op	Vj	Vk	Qj	Qk
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	No					
	Mult2	No					

Register result status

	F0	F2	F4	F6	F8	F10	...	F30
FU				Load1				

Clock: 1

Tomasulo example, cycle 2

Instruction status

Instruction		j	k	Issue	Exec. compl.	Write result
LD	F6	34+	R2	1		
LD	F2	45+	R3	2		
MULTD	F0	F2	F4			
SUBD	F8	F6	F2			
DIVD	F10	F0	F6			
ADDD	F6	F8	F2			

Load buffers

Time		Busy	Address
	Load1	Yes	R2+32
1	Load2	Yes	R3+45
	Load3	No	

Functional unit status

				src 1	src 2	RS for j	RS for k
Time	Name	Busy	Op	Vj	Vk	Qj	Qk
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	No					
	Mult2	No					

Register result status

	F0	F2	F4	F6	F8	F10	...	F30
FU		Load2		Load1				

Clock: 2

Tomasulo example, cycle 3

Instruction status

Instruction		j	k	Issue	Exec. compl.	Write result
LD	F6	34+	R2	1	3	
LD	F2	45+	R3	2		
MULTD	F0	F2	F4	3		
SUBD	F8	F6	F2			
DIVD	F10	F0	F6			
ADDD	F6	F8	F2			

Load buffers

Time		*Busy*	*Address*
0	***Load1***	Yes	R2+32
1	***Load2***	Yes	R3+45
	Load3	No	

Functional unit status

Time	Name	*Busy*	*Op*	src 1 *Vj*	src 2 *Vk*	RS for j *Qj*	RS for k *Qk*
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	Yes	Mult		F4	Load2	
	Mult2	No					

Register result status

	F0	*F2*	*F4*	*F6*	*F8*	*F10*	*...*	*F30*
FU	Mult1	Load2		Load1				

Clock: 3

Tomasulo example, cycle 4

Instruction status

Instruction		j	k	*Issue*	*Exec. compl.*	*Write result*
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	
MULTD	F0	F2	F4	3		
SUBD	F8	F6	F2	4		
DIVD	F10	F0	F6			
ADDD	F6	F8	F2			

Load buffers

Time		*Busy*	*Address*
	Load1	No	
0	***Load2***	Yes	R3+45
	Load3	No	

Functional unit status

				src 1	src 2	RS for j	RS for k
Time	Name	*Busy*	*Op*	*Vj*	*Vk*	*Qj*	*Qk*
	Add1	Yes	Sub	M(R2+34)			Load2
	Add2	No					
	Add3	No					
	Mult1	Yes	Mult		F4	Load2	
	Mult2	No					

Register result status

	F0	*F2*	*F4*	*F6*	*F8*	*F10*	*...*	*F30*
FU	Mult1	Load2		-	Add1			

Clock: 4

Tomasulo example, cycle 5

Instruction status

Instruction		j	k	Issue	Exec. compl.	Write result
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3		
SUBD	F8	F6	F2	4		
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2			

Load buffers

Time		Busy	Address
	Load1	No	
	Load2	No	
	Load3	No	

Functional unit status

Time	Name	Busy	Op	src 1 Vj	src 2 Vk	RS for j Qj	RS for k Qk
2	Add1	Yes	Sub	M(R2+34)	M(R3+45)		-
	Add2	No					
	Add3	No					
10	Mult1	Yes	Mult	M(R3+45)	F4	-	
	Mult2	Yes	Div		F6	Mult1	

Register result status

	F0	F2	F4	F6	F8	F10	...	F30
FU	Mult1	-			Add1	Mult2		

Clock: 5

Tomasulo example, cycle 6

Instruction status

Instruction		j	k	Issue	Exec. compl.	Write result
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3		
SUBD	F8	F6	F2	4		
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2	6		

Load buffers

Time		Busy	Address
	Load1	No	
	Load2	No	
	Load3	No	

Functional unit status

				src 1	src 2	RS for j	RS for k
Time	Name	Busy	Op	Vj	Vk	Qj	Qk
1	Add1	Yes	Sub	M(R2+34)	M(R3+45)		
	Add2	Yes	Add		F2	Add1	
	Add3	No					
9	Mult1	Yes	Mult	M(R3+45)	F4		
	Mult2	Yes	Div		F6	Mult1	

Register result status

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

Clock: 6

Tomasulo example, cycle 7

Instruction status

Instruction		j	k	*Issue*	*Exec. compl.*	*Write result*
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3		
SUBD	F8	F6	F2	4	7	
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2	6		

Load buffers

Time		*Busy*	*Address*
	Load1	No	
	Load2	No	
	Load3	No	

Functional unit status

				src 1	src 2	RS for j	RS for k
Time	Name	***Busy***	***Op***	***Vj***	***Vk***	***Qj***	***Qk***
0	***Add1***	Yes	Sub	M(R2+34)	M(R3+45)		
	Add2	Yes	Add		F2	Add1	
	Add3	No					
8	***Mult1***	Yes	Mult	M(R3+45)	F4		
	Mult2	Yes	Div		F6	Mult1	

Register result status

	F0	***F2***	***F4***	***F6***	***F8***	***F10***	***...***	***F30***
FU	Mult1			Add2	Add1	Mult2		

Clock: 7

Tomasulo example, cycle 8

Instruction status

Instruction		j	k	*Issue*	*Exec. compl.*	*Write result*
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3		
SUBD	F8	F6	F2	4	7	8
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2	6		

Load buffers

Time		*Busy*	*Address*
	Load1	No	
	Load2	No	
	Load3	No	

Functional unit status

Time	Name	*Busy*	*Op*	src 1 *Vj*	src 2 *Vk*	RS for j *Qj*	RS for k *Qk*
	Add1	No					
2	***Add2***	Yes	Add	F6-F2	F2	-	
	Add3	No					
7	***Mult1***	Yes	Mult	M(R3+45)	F4		
	Mult2	Yes	Div		F6	Mult1	

Register result status

	F0	*F2*	*F4*	*F6*	*F8*	*F10*	*...*	*F30*
FU	Mult1			Add2	-	Mult2		

Clock: 8

Tomasulo example, cycle 10

Instruction status

Instruction		j	k	Issue	Exec. compl.	Write result
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3		
SUBD	F8	F6	F2	4	7	8
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2	6	10	

Load buffers

Time		Busy	Address
	Load1	No	
	Load2	No	
	Load3	No	

Functional unit status

Time	Name	Busy	Op	src 1 Vj	src 2 Vk	RS for j Qj	RS for k Qk
	Add1	No					
0	Add2	Yes	Add	F6-F2	F2		
	Add3	No					
5	Mult1	Yes	Mult	M(R3+45)	F4		
	Mult2	Yes	Div		F6	Mult1	

Register result status

	F0	F2	F4	F6	F8	F10	...	F30
FU	Mult1			Add2		Mult2		

Clock: 10

Tomasulo example, cycle 11

Instruction status

Instruction		j	k	*Issue*	*Exec. compl.*	*Write result*
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3		
SUBD	F8	F6	F2	4	7	8
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2	6	10	11

Load buffers

Time		*Busy*	*Address*
	Load1	No	
	Load2	No	
	Load3	No	

Functional unit status

Time	Name	*Busy*	*Op*	src 1 *Vj*	src 2 *Vk*	RS for j *Qj*	RS for k *Qk*
	Add1	No					
	Add2	No					
	Add3	No					
4	***Mult1***	Yes	Mult	M(R3+45)	F4		
	Mult2	Yes	Div		F6	Mult1	

Register result status

	F0	*F2*	*F4*	*F6*	*F8*	*F10*	*...*	*F30*
FU	Mult1			-		Mult2		

Clock: 11

Tomasulo example, cycle 15

Instruction status

Instruction		j	k	*Issue*	*Exec. compl.*	*Write result*
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3	15	
SUBD	F8	F6	F2	4	7	8
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2	6	10	11

Load buffers

Time		*Busy*	*Address*
	Load1	No	
	Load2	No	
	Load3	No	

Functional unit status

				src 1	src 2	RS for j	RS for k
Time	Name	*Busy*	*Op*	*Vj*	*Vk*	*Qj*	*Qk*
	Add1	No					
	Add2	No					
	Add3	No					
0	***Mult1***	Yes	Mult	M(R3+45)	F4		
	Mult2	Yes	Div		F6	Mult1	

Register result status

	F0	*F2*	*F4*	*F6*	*F8*	*F10*	*...*	*F30*
FU	Mult1					Mult2		

Clock: 15

Tomasulo example, cycle 16

Instruction status

Instruction		j	k	Issue	Exec. compl.	Write result
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3	15	16
SUBD	F8	F6	F2	4	7	8
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2	6	10	11

Load buffers

Time		Busy	Address
	Load1	No	
	Load2	No	
	Load3	No	

Functional unit status

Time	Name	Busy	Op	src 1 Vj	src 2 Vk	RS for j Qj	RS for k Qk
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	No					
40	Mult2	Yes	Div	F0	F6	-	

Register result status

	F0	F2	F4	F6	F8	F10	...	F30
FU	-					Mult2		

Clock: 15

Tomasulo example, cycle 56

Instruction status

Instruction		j	k	Issue	Exec. compl.	Write result
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3	15	16
SUBD	F8	F6	F2	4	7	8
DIVD	F10	F0	F6	5	56	
ADDD	F6	F8	F2	6	10	11

Load buffers

Time		Busy	Address
	Load1	No	
	Load2	No	
	Load3	No	

Functional unit status

Time	Name	Busy	Op	src 1 Vj	src 2 Vk	RS for j Qj	RS for k Qk
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	No					
0	Mult2	Yes	Div	F0	F6		

Register result status

	F0	F2	F4	F6	F8	F10	...	F30
FU						Mult2		

Clock: 56

Tomasulo example, cycle 57

Instruction status

Instruction		j	k	Issue	Exec. compl.	Write result
LD	F6	34+	R2	1	3	4
LD	F2	45+	R3	2	4	5
MULTD	F0	F2	F4	3	15	16
SUBD	F8	F6	F2	4	7	8
DIVD	F10	F0	F6	5	56	57
ADDD	F6	F8	F2	6	10	11

Load buffers

Time		Busy	Address
	Load1	No	
	Load2	No	
	Load3	No	

Functional unit status

Time	Name	Busy	Op	src 1 Vj	src 2 Vk	RS for j Qj	RS for k Qk
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	No					
	Mult2	No					

Register result status

	F0	F2	F4	F6	F8	F10	...	F30
FU						-		

Clock: 57

Example of WAR hazards in Tomasulo’s Algorithm

Example:

```
LF      F6, 34(R2)
DIVF    F10, F6, F0
ADDF    F6, F8, F2
```

ADDF can safely finish before DIVF has read register F6 because:

- DIVF has ***renamed*** register F6 to point at LFs functional unit
- LF ***broadcasts*** its result on the Common Data Bus

Register renaming can thus be done:

- statically by the compiler
- dynamically by the hardware