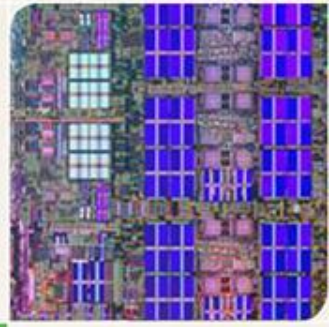
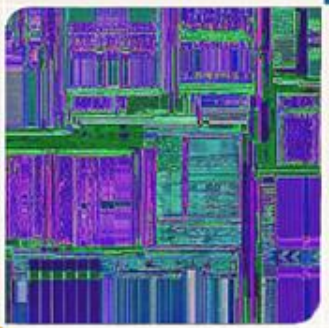
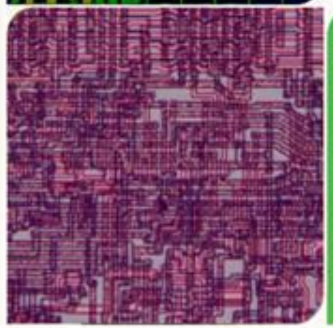



CS4290/CS6290

Fall 2011

Prof. Hyesoon Kim

Georgia Tech College of Computing

Thanks to Prof. Prvulovic and Prof. Loh

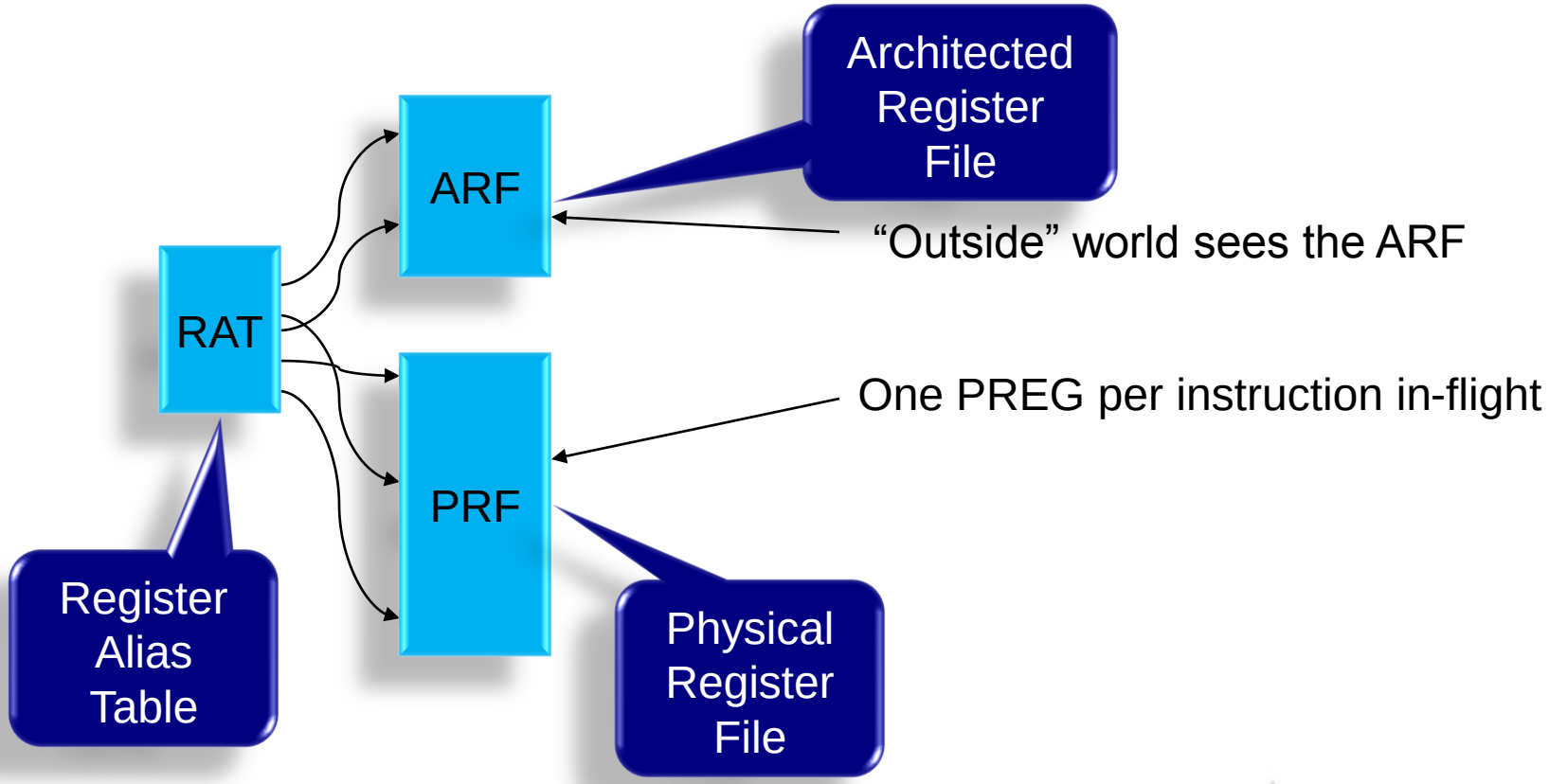


Review

- Control Hazard
- Data Hazard
  - Data dependences
    - RAW/WAR/WAW
    - True/Anti/Output
  - Memory dependences
    - Memory disambiguation problem
- Architecture vs. microarchitecture

Register File Organization

- We need some physical structure to store the register values

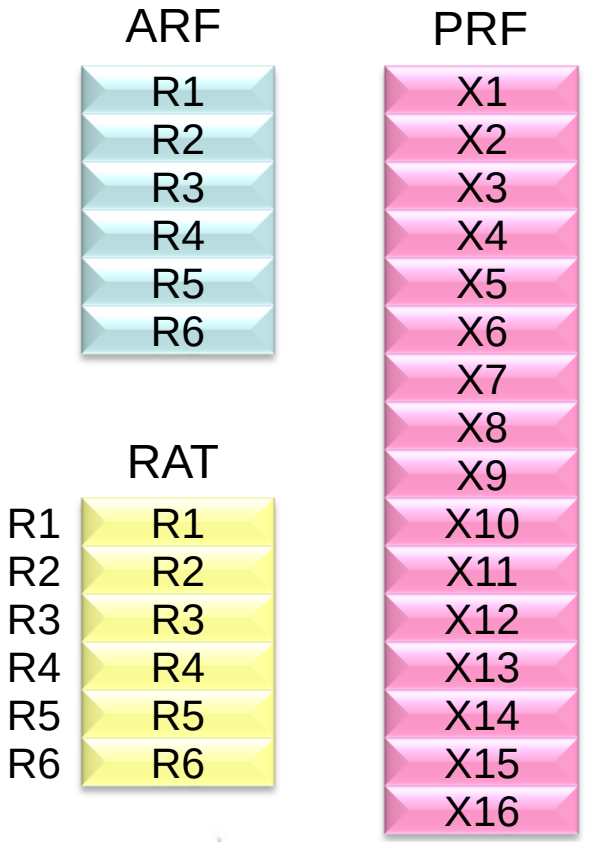


Putting it all Together

```
top:
    R1 = R2 + R3
    R2 = R4 – R1
    R1 = R3 * R6
    R2 = R1 + R2
    R3 = R1 >> 1
    BNEZ R3, top
```

Free pool:

X9, X11, X7, X2, X13, X4, X8, X12, X3, X5…



1. List all dependencies for 2 iterations
2. Show graph when all deps enforced
3. Show graph when only RAW enforced
4. Rename the registers for 2 iterations

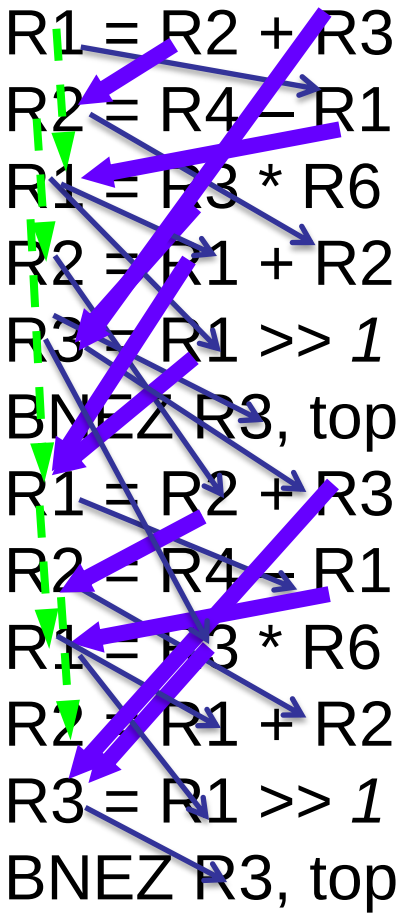


Putting it all Together

```
R1 = R2 + R3
R2 = R4 – R1
R1 = R3 * R6
R2 = R1 + R2
R3 = R1 >> 1
BNEZ R3, top
R1 = R2 + R3
R2 = R4 – R1
R1 = R3 * R6
R2 = R1 + R2
R3 = R1 >> 1
BNEZ R3, top
```

1. List all dependencies for 2 iterations

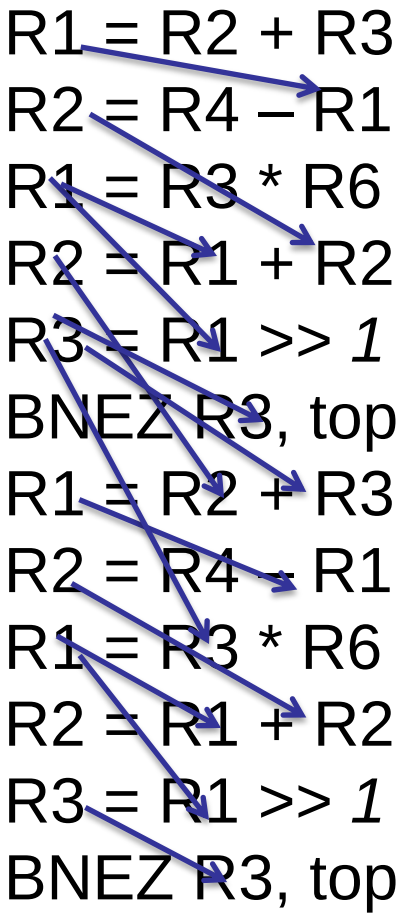
Putting it all Together



2. Show graph when all deps enforced

RAW
WAR
WAW

Putting it all Together



3. Show graph when only RAW enforced

Putting it all Together



RAW
WAR
WAW

```
R1 = R2 + R3      X9 = R2 + R3
R2 = R4 – R1      X11 = R4 – X9
R1 = R3 * R6      X7 = R3 * R6
R2 = R1 + R2      X2= X7 + X11
R3 = R1 >> 1      X13 = X7 >> 1
BNEZ R3, top      BNEZ X13, top
R1 = R2 + R3      X14= X2 + X13
R2 = R4 – R1      X8 = R4 – X14
R1 = R3 * R6      X12 = X13 * R6
R2 = R1 + R2      X3 = X12 + X8
R3 = R1 >> 1      X5 = X12 >> 1
BNEZ R3, top      BNEZ X5, top
```

Free pool:

X9, X11, X7, X2, X13, X14, X8, X12, X3, X5…

ARF

R1
R2
R3
R4
R5
R6

PRF

X1
X2
X3
X4
X5
X6
X7
X8
X9
X10
X11
X12
X13
X14
X15
X16

RAT

R1	X12
R2	X3
R3	X5
R4	R4
R5	R5
R6	R6

4. Rename the registers for 2 iterations

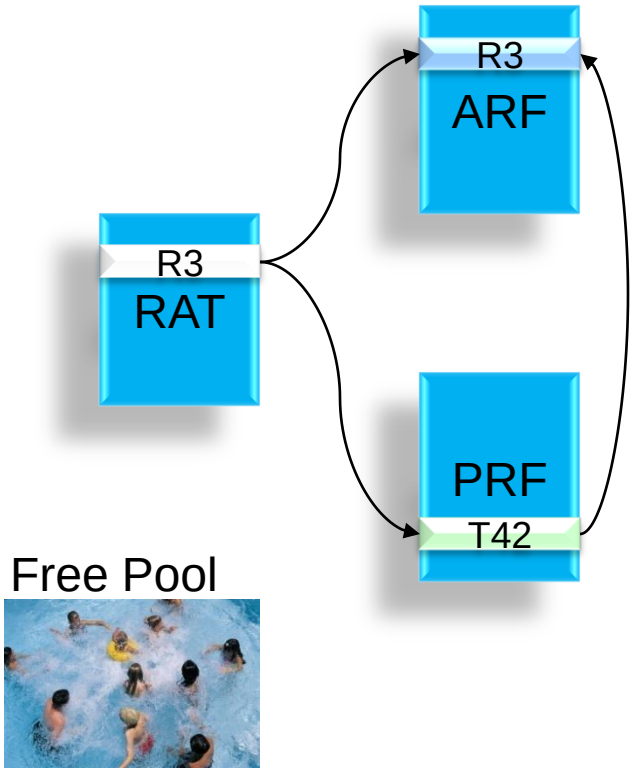


Even Physical Registers are Limited

- We keep using new physical registers
  - What happens when we run out?
- There must be a way to “recycle”

- When can we recycle?
  - When we have given its value to all instructions that use it as a source operand!
  - This is not as easy as it sounds

Instruction Commit (leaving the pipe)



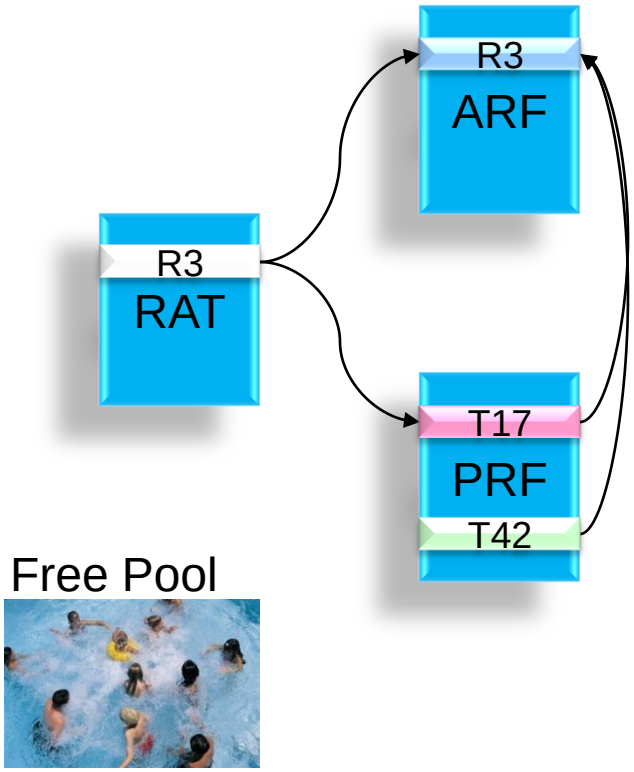
Architected register file contains the “official” processor state

When an instruction leaves the pipeline, it makes its result “official” by updating the ARF

The ARF now contains the correct value; update the RAT

T42 is no longer needed, return to the physical register free pool

Careful with the RAT Update!



Update ARF as usual

Deallocate physical register

Don't touch the RAT!
(Someone else is the most recent writer to R3)

At some point in the future, the newer writer of R3 exits

This instruction was the most recent writer, *now* update the RAT

Deallocate physical register



How?

- Scoreboard
- Tomasulo's algorithm
- ROB
- History buffer
- Check point (HPS)

Scoreboard

- Before writing, it sets busy
- After finishing it sets ready
- Keep track of registers are ready or not
- Keep track of functional unit states

Register ID	valid	

Limitations of Scoreboard

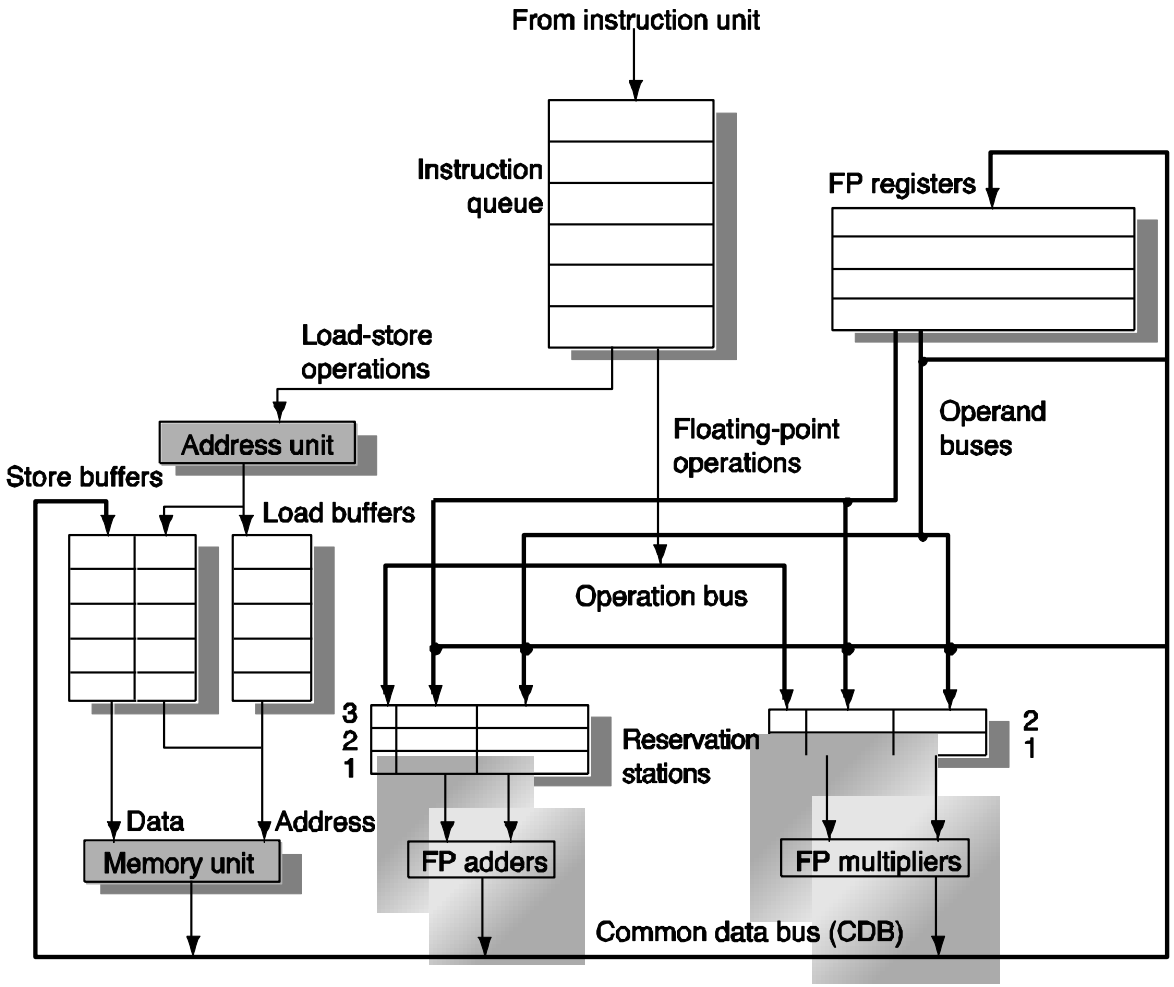
- No register renaming
- Detect WAR and WAW but it cannot not eliminate them.
  - Solutions for WAR:
    - Stall writeback until registers have been read
    - Read registers only during Read Operands stage
  - Solution for WAW:
    - Detect hazard and stall issue of new instruction until other instruction completes



Implementing Dynamic Scheduling

- Tomasulo's Algorithm
  - Used in IBM 360/91 (in the 60s)
  - Tracks when operands are available to satisfy data dependences
  - Removes name dependences through register renaming
  - Very similar to what is used today
    - Almost all modern high-performance processors use a derivative of Tomasulo's… much of the terminology survives to today.

Tomasulo's Algorithm: The Picture



Three Stages of Tomasulo Algorithm

1. Issue—get instruction from Instruction Queue

   If reservation station free (no structural hazard),
   control issues instr & sends operands (renames registers).

2. Execution—operate on operands (EX)

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

3. Write result—finish execution (WB)

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

- Normal data bus: data + destination ("go to" bus)
- Common data bus: data + source ("come from" bus)
  - 64 bits of data + 4 bits of Functional Unit source address
  - Write if matches expected Functional Unit (produces result)
  - Does the broadcast



Issue (1)

- Get next instruction from instruction queue.
- Find a free *reservation station* for it
  (if none are free, stall until one is)
- Read operands that are in the registers
- If the operand is not in the register,
  find which reservation station will produce it
- In effect, this step renames registers
  (reservation station IDs are "temporary" names)

Issue (2)

Instruction Buffers

0.	F2 = F4 + F1
1.	F1 = F2 / F3
2.	F4 = F1 – F2
3.	F1 = F2 + F3

To-Do list (from last slide):
Get next inst from IB's
Find free reservation station
Read operands from RF
Record source of other operands
Update source mapping (RAT)

Reg File

F1	3.141593
F2	-1.00000
F3	2.718282
F4	0.707107

RAT

F1	0
F2	α
F3	0
F4	0

A1 (α)	F2=F4+F1	0.7071	3.14
A2 (β)			
A3 (χ)			

C1 (δ)	F1 = F2/F3	α(A1)	2.718282
C2 (ε)			

Reservation stations

Adder

FP-Cmplx



Execute (1)

- Monitor results as they are produced
- Put a result into all reservation stations waiting for it (missing source operand)
- When *all operands available* for an instruction, it is ready (we can actually execute it)
- Several ready instrs for one functional unit?
  - Pick one.
  - Except for load/store
    Load/Store must be done in
    the proper order to avoid hazards through memory
    (more loads/stores this in a later lecture)

Execute (2)

1.	F1 = F2 / F3
2.	F4 = F1 – F2
3.	F1 = F2 + F3

F2=F4+F1

(α) 3.8487

To-Do list (from last slide):
Monitor results from ALUs
Capture matching operands
Compete for ALUs

A1 (α)			
A2 (β)	F4=F1-F2	(δ)	~~(α)~~ 3.8487
A3 (χ)	F1=F2+F3	~~(α)~~ 3.8487	2.718

Adder

C1 (δ)	F1=F2/F3	~~(α)~~ 3.8487	2.718
C2 (ε)			

FP-Cmplx

Execute (3)
More than one ready inst for the same unit

Common heuristic: oldest first

You can do whatever: it only affects performance, not correctness

F2=F4+F1
(α) 3.8487

3.	F1 = F2 + F3
2.	F4 = F3 – F2
1.	F1 = F2 / F3
0.	F2 = F4 + F1

A1 (α)			
A2 (β)	F4=F3-F2	2.718	3.8487
A3 (χ)	F1=F2+F3	3.8487	2.718

Adder

C1 (δ)	F1=F2/F3	3.8487	2.718
C2 (ε)			

FP-Cmplx



Write Result (1)

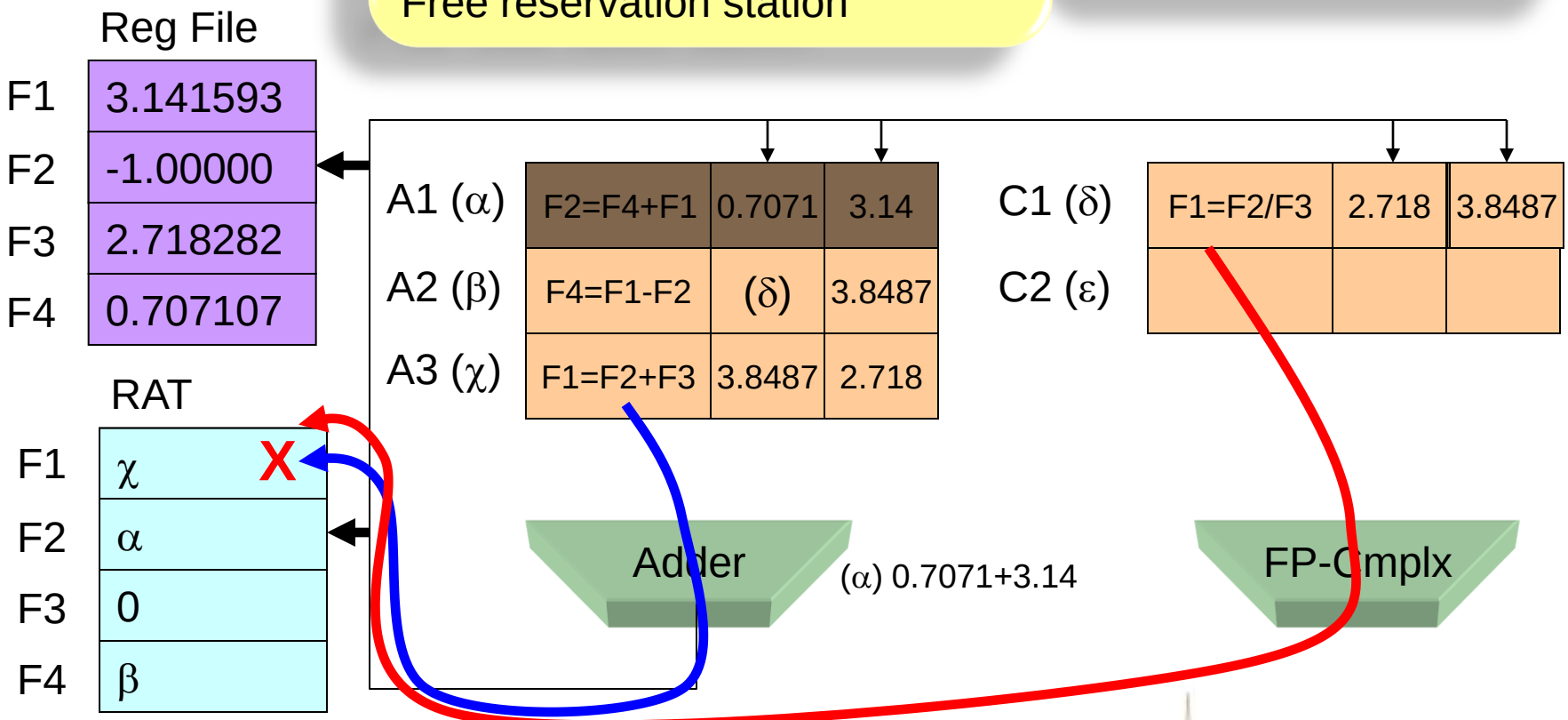
- When result is computed, make it available on the "common data bus" (CDB), where waiting reservation stations can pick it up
- Stores write to memory
- Result stored in the register file
- This step frees the reservation station
- For our register renaming, this recycles the temporary name (future instructions can again find the value in the actual register, until it is renamed again)

Write Result (2)

0. F2 = F4 + F1
1. F1 = F2 / F3
2. F4 = F1 – F2
3. F1 = F2 + F3

To-Do list (from last slide):
Broadcast on CDB
Writeback to RF
Update Mapping
Free reservation station

Only update RAT (and RF) if RAT still contains your mapping!





Tomasulo's Algorithm: Load/Store

- The reservation stations take care of dependences through registers.
- Dependences also possible through memory
  - Loads and stores not reordered in original IBM 360
  - We'll talk about how to do load-store reordering later



Reservation Station Components

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

Vj, Vk: Value of Source operands

- Store buffers has V field, result to be stored

Qj, Qk: Reservation stations producing source registers (value to be written)

- 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.

A : Memory address calculation for a load or store

Detailed Example

Load: 2 cycles
Add: 2 cycles
Mult: 10 cycles
Divide: 40 cycles

Assume

R2 is 100

R3 is 200

F4 is 2.5

		Is	Ex	W
1. L.D	F6, 34(R2)			
2. L.D	F2, 45(R3)			
3. MUL.D	F0, F2, F4			
4. SUB.D	F8, F2, F6			
5. DIV.D	F10,F0,F6			
6. ADD.D	F6, F8, F2			

Reservation Stations

	Busy	Op	Vj	Vk	Qj	Qk	A
LD1							
LD2							
AD1							
AD2							
AD3							
ML1							
ML2							

Cycle:

Register Status:

F0	F2	F4	F6	F8	F10	F12	…

Detailed Example

Load: 2 cycles
Add: 2 cycles
Mult: 10 cycles
Divide: 40 cycles

Assume

R2 is 100

R3 is 200

F4 is 2.5

		Is	Ex	W
1. L.D	F6, 34(R2)	1		
2. L.D	F2, 45(R3)			
3. MUL.D	F0, F2, F4			
4. SUB.D	F8, F2, F6			
5. DIV.D	F10,F0,F6			
6. ADD.D	F6, F8, F2			

Reservation Stations

	Busy	Op	Vj	Vk	Qj	Qk	A
LD1	1	L.D					134
LD2							
AD1							
AD2							
AD3							
ML1							
ML2							

	F0	F2	F4	F6	F8	F10	F12
Register Status:				LD1			…

Cycle: 1

Detailed Example

Load: 2 cycles
Add: 2 cycles
Mult: 10 cycles
Divide: 40 cycles

Assume

R2 is 100

R3 is 200

F4 is 2.5

		Is	Ex	W
1. L.D	F6, 34(R2)	1	2	
2. L.D	F2, 45(R3)	2		
3. MUL.D	F0, F2, F4			
4. SUB.D	F8, F2, F6			
5. DIV.D	F10,F0,F6			
6. ADD.D	F6, F8, F2			

Reservation Stations

	Busy	Op	Vj	Vk	Qj	Qk	A
LD1	1	L.D					134
LD2	1	L.D					245
AD1							
AD2							
AD3							
ML1							
ML2							

Cycle: 2

Register Status:

F0	F2	F4	F6	F8	F10	F12	
	LD2		LD1				…

Detailed Example

Load: 2 cycles
Add: 2 cycles
Mult: 10 cycles
Divide: 40 cycles

Assume

R2 is 100

R3 is 200

F4 is 2.5

		Is	Ex	W
1. L.D	F6, 34(R2)	1	2	
2. L.D	F2, 45(R3)	2	3	
3. MUL.D	F0, F2, F4	3		
4. SUB.D	F8, F2, F6			
5. DIV.D	F10,F0,F6			
6. ADD.D	F6, F8, F2			

Reservation Stations

	Busy	Op	Vj	Vk	Qj	Qk	A
LD1	1	L.D					134
LD2	1	L.D					245
AD1							
AD2							
AD3							
ML1	1	MUL.D		2.5	LD2		
ML2							

Cycle: 3

Register Status:

F0	F2	F4	F6	F8	F10	F12	
ML1	LD2		LD1				…

Detailed Example

Load: 2 cycles
Add: 2 cycles
Mult: 10 cycles
Divide: 40 cycles

Assume

R2 is 100

R3 is 200

F4 is 2.5

F6 is 0.5

		Is	Ex	W
1. L.D	F6, 34(R2)	1	2	4
2. L.D	F2, 45(R3)	2	3	
3. MUL.D	F0, F2, F4	3		
4. SUB.D	F8, F2, F6	4		
5. DIV.D	F10,F0,F6			
6. ADD.D	F6, F8, F2			

Reservation Stations

	Busy	Op	Vj	Vk	Qj	Qk	A
LD1	0						
LD2	1	L.D					245
AD1	1	SUB.D		0.5	LD2		
AD2							
AD3							
ML1	1	MUL.D		2.5	LD2		
ML2							

	F0	F2	F4	F6	F8	F10	F12
Register Status:	ML1	LD2		LD1	AD1		…

Cycle: 4

Detailed Example

Load: 2 cycles
Add: 2 cycles
Mult: 10 cycles
Divide: 40 cycles

Assume

R2 is 100

R3 is 200

F4 is 2.5

F6 is 0.5

		Is	Ex	W
1. L.D	F6, 34(R2)	1	2	4
2. L.D	F2, 45(R3)	2	3	5
3. MUL.D	F0, F2, F4	3		
4. SUB.D	F8, F2, F6	4		
5. DIV.D	F10,F0,F6	5		
6. ADD.D	F6, F8, F2			

Reservation Stations

	Busy	Op	Vj	Vk	Qj	Qk	A
LD1	0						
LD2	0						
AD1	1	SUB.D	1.5	0.5			
AD2							
AD3							
ML1	1	MUL.D	1.5	2.5			
ML2	1	DIV.D		0.5	ML1		

	F0	F2	F4	F6	F8	F10	F12	
Register Status:	ML1				AD1	ML2		…

Cycle: 5

Detailed Example

Load: 2 cycles
Add: 2 cycles
Mult: 10 cycles
Divide: 40 cycles

Assume

R2 is 100

R3 is 200

F4 is 2.5

		Is	Ex	W
1. L.D	F6, 34(R2)	1	2	4
2. L.D	F2, 45(R3)	2	3	5
3. MUL.D	F0, F2, F4	3	6	
4. SUB.D	F8, F2, F6	4	6	
5. DIV.D	F10,F0,F6	5		
6. ADD.D	F6, F8, F2	6		

Reservation Stations

	Busy	Op	Vj	Vk	Qj	Qk	A
LD1	0						
LD2	0						
AD1	1	SUB.D	1.5	0.5			
AD2	1	ADD.D		2.5	AD1		
AD3							
ML1	1	MUL.D	1.5	2.5			
ML2	1	DIV.D		0.5	ML1		

Cycle: 6

Register Status:

F0	F2	F4	F6	F8	F10	F12	
ML1			AD2	AD1	ML2		…

Detailed Example

Load: 2 cycles
Add: 2 cycles
Mult: 10 cycles
Divide: 40 cycles

Assume

R2 is 100

R3 is 200

F4 is 2.5

F6 is 0.5

F8 is 1.0

		Is	Ex	W
1. L.D	F6, 34(R2)	1	2	4
2. L.D	F2, 45(R3)	2	3	5
3. MUL.D	F0, F2, F4	3	6	
4. SUB.D	F8, F2, F6	4	6	8
5. DIV.D	F10,F0,F6	5		
6. ADD.D	F6, F8, F2	6		

Reservation Stations

	Busy	Op	Vj	Vk	Qj	Qk	A
LD1	0						
LD2	0						
AD1	0						
AD2	1	ADD.D	1.0	2.5			
AD3							
ML1	1	MUL.D	1.5	2.5			
ML2	1	DIV.D		0.5	ML1		

Cycle: 8

Register Status:

F0	F2	F4	F6	F8	F10	F12	
ML1			AD2		ML2		…

Detailed Example

Load: 2 cycles
Add: 2 cycles
Mult: 10 cycles
Divide: 40 cycles

Assume

R2 is 100

R3 is 200

F4 is 2.5

F6 is 0.5

F8 is 1.0

		Is	Ex	W
1. L.D	F6, 34(R2)	1	2	4
2. L.D	F2, 45(R3)	2	3	5
3. MUL.D	F0, F2, F4	3	6	
4. SUB.D	F8, F2, F6	4	6	8
5. DIV.D	F10,F0,F6	5		
6. ADD.D	F6, F8, F2	6	9	

Reservation Stations

	Busy	Op	Vj	Vk	Qj	Qk	A
LD1	0						
LD2	0						
AD1	0						
AD2	1	ADD.D	1.0	2.5			
AD3							
ML1	1	MUL.D	1.5	2.5			
ML2	1	DIV.D		0.5	ML1		

Cycle: 9

Register Status:

F0	F2	F4	F6	F8	F10	F12	
ML1			AD2		ML2		…

Detailed Example

Load: 2 cycles
Add: 2 cycles
Mult: 10 cycles
Divide: 40 cycles

Assume

R2 is 100

R3 is 200

F4 is 2.5

F6 is 0.5

F8 is 1.0

		Is	Ex	W
1. L.D	F6, 34(R2)	1	2	4
2. L.D	F2, 45(R3)	2	3	5
3. MUL.D	F0, F2, F4	3	6	
4. SUB.D	F8, F2, F6	4	6	8
5. DIV.D	F10,F0,F6	5		
6. ADD.D	F6, F8, F2	6	9	11

Reservation Stations

	Busy	Op	Vj	Vk	Qj	Qk	A
LD1	0						
LD2	0						
AD1	0						
AD2	0						
AD3							
ML1	1	MUL.D	1.5	2.5			
ML2	1	DIV.D		0.5	ML1		

	F0	F2	F4	F6	F8	F10	F12	
Register Status:	ML1					ML2		…

Cycle: 11

Detailed Example

Load: 2 cycles
Add: 2 cycles
Mult: 10 cycles
Divide: 40 cycles

Assume

R2 is 100

R3 is 200

F4 is 2.5

F6 is 0.5

F8 is 1.0

		Is	Ex	W
1. L.D	F6, 34(R2)	1	2	4
2. L.D	F2, 45(R3)	2	3	5
3. MUL.D	F0, F2, F4	3	6	16
4. SUB.D	F8, F2, F6	4	6	8
5. DIV.D	F10,F0,F6	5		
6. ADD.D	F6, F8, F2	6	9	11

Reservation Stations

	Busy	Op	Vj	Vk	Qj	Qk	A
LD1	0						
LD2	0						
AD1	0						
AD2	0						
AD3							
ML1	0						
ML2	1	DIV.D	3.75	0.5			

	F0	F2	F4	F6	F8	F10	F12	
Register Status:						ML2		…

Cycle: 16

Detailed Example

Load: 2 cycles
Add: 2 cycles
Mult: 10 cycles
Divide: 40 cycles

Assume

R2 is 100

R3 is 200

F4 is 2.5

F6 is 0.5

F8 is 1.0

		Is	Ex	W
1. L.D	F6, 34(R2)	1	2	4
2. L.D	F2, 45(R3)	2	3	5
3. MUL.D	F0, F2, F4	3	6	16
4. SUB.D	F8, F2, F6	4	6	8
5. DIV.D	F10,F0,F6	5	17	
6. ADD.D	F6, F8, F2	6	9	11

Reservation Stations

	Busy	Op	Vj	Vk	Qj	Qk	A
LD1	0						
LD2	0						
AD1	0						
AD2	0						
AD3							
ML1	0						
ML2	1	DIV.D	3.75	0.5			

	F0	F2	F4	F6	F8	F10	F12	
Register Status:						ML2		…

Cycle: 17

Detailed Example

Load: 2 cycles
Add: 2 cycles
Mult: 10 cycles
Divide: 40 cycles

Assume

R2 is 100

R3 is 200

F4 is 2.5

F6 is 0.5

F8 is 1.0

		Is	Ex	W
1. L.D	F6, 34(R2)	1	2	4
2. L.D	F2, 45(R3)	2	3	5
3. MUL.D	F0, F2, F4	3	6	16
4. SUB.D	F8, F2, F6	4	6	8
5. DIV.D	F10,F0,F6	5	17	
6. ADD.D	F6, F8, F2	6	9	11

Reservation Stations

	Busy	Op	Vj	Vk	Qj	Qk	A
LD1	0						
LD2	0						
AD1	0						
AD2	0						
AD3							
ML1	0						
ML2	1	DIV.D	3.75	0.5			

	F0	F2	F4	F6	F8	F10	F12	
Register Status:						ML2		…

Cycle: 18

Detailed Example

Load: 2 cycles
Add: 2 cycles
Mult: 10 cycles
Divide: 40 cycles

Assume

R2 is 100

R3 is 200

F4 is 2.5

F6 is 0.5

F8 is 1.0

		Is	Ex	W
1. L.D	F6, 34(R2)	1	2	4
2. L.D	F2, 45(R3)	2	3	5
3. MUL.D	F0, F2, F4	3	6	16
4. SUB.D	F8, F2, F6	4	6	8
5. DIV.D	F10,F0,F6	5	17	57
6. ADD.D	F6, F8, F2	6	9	11

Reservation Stations

	Busy	Op	Vj	Vk	Qj	Qk	A
LD1	0						
LD2	0						
AD1	0						
AD2	0						
AD3							
ML1	0						
ML2	0	DIV.D	3.75	0.5			

	F0	F2	F4	F6	F8	F10	F12	
Register Status:						ML2		…

Cycle: 57