P6

P6: Start with Tomasulo's algorithm… add ROB

- Separate ROB and RS

Simple-P6

- Our old RS organization: 1 ALU, 1 load, 1 store, 2 3-cycle FP

P6 Data Structures

Reservation Stations are same as before

ROB

- **head**, **tail**: pointers maintain sequential order
- **R**: insn output register, **V**: insn output value

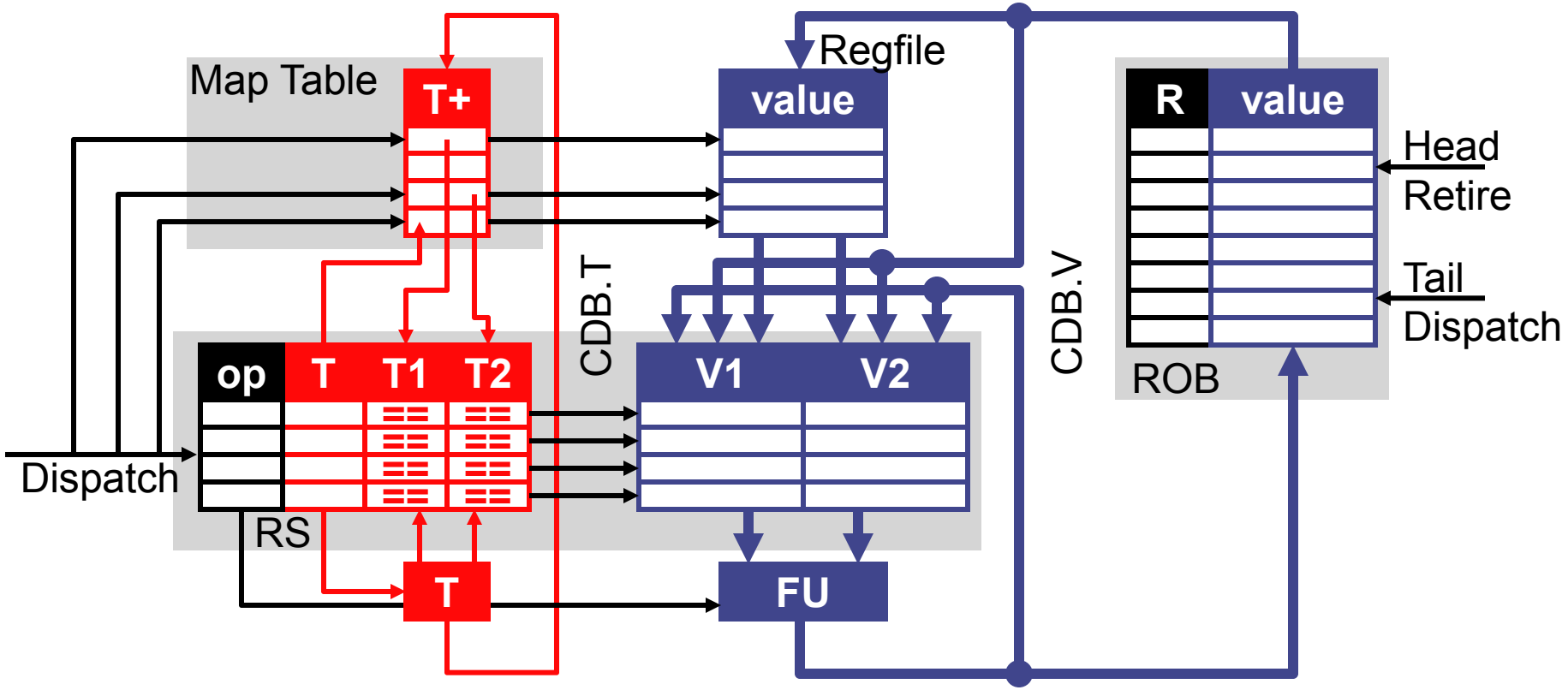
Tags are different

- Tomasulo: RS# → P6: ROB#

Map Table is different

- **T+**: tag + "ready-in-ROB" bit
- T==0 → Value is ready in regfile
- T!=0 → Value is not ready
- T!=0+ → Value is ready in the ROB

P6 Data Structures



- Insn fields and status bits
- Tags
- Values

P6 Data Structures

ROB							
ht	#	Insn	R	V	S	X	C
	1	`ldf X(r1),f1`					
	2	`mulf f0,f1,f2`					
	3	`stf f2,Z(r1)`					
	4	`addi r1,4,r1`					
	5	`ldf X(r1),f1`					
	6	`mulf f0,f1,f2`					
	7	`stf f2,Z(r1)`					

Map Table	
Reg	T+
f0	
f1	
f2	
r1	

CDB	
T	V

Reservation Stations								
#	FU	busy	op	T	T1	T2	V1	V2
1	ALU	no						
2	LD	no						
3	ST	no						
4	FP1	no						
5	FP2	no						

P6 Pipeline

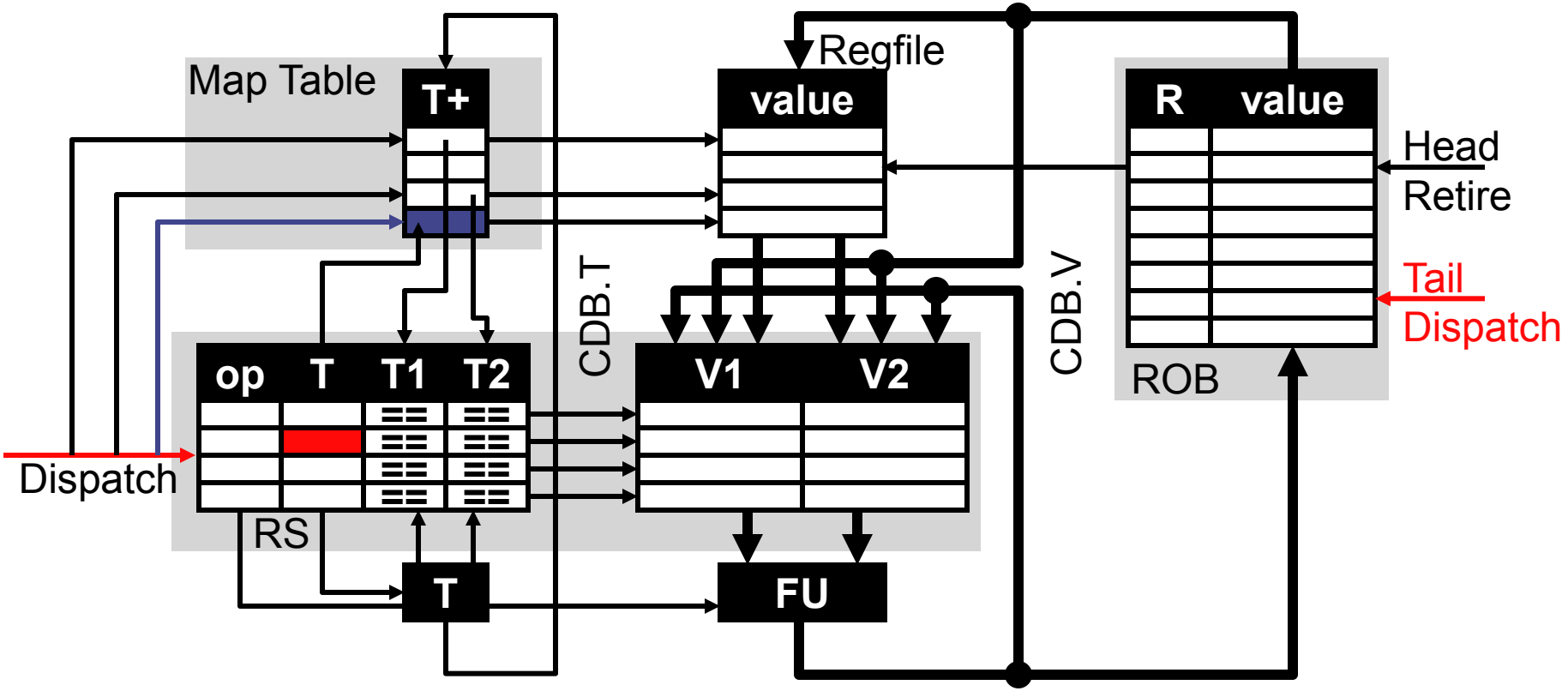
New pipeline structure: F, **D**, S, **X**, **C**, **R**

- **D (dispatch)**
  - Structural hazard (ROB/LSQ/RS) ? **Stall**
  - Allocate ROB/LSQ/RS
  - Set RS tag to ROB#
  - Set Map Table entry to ROB# and clear "ready-in-ROB" bit
  - Read ready registers into RS (from either ROB or Regfile)
- **X (execute)**
  - Free RS entry
  - Use to be at W, can be earlier because RS# are not tags

P6 Pipeline

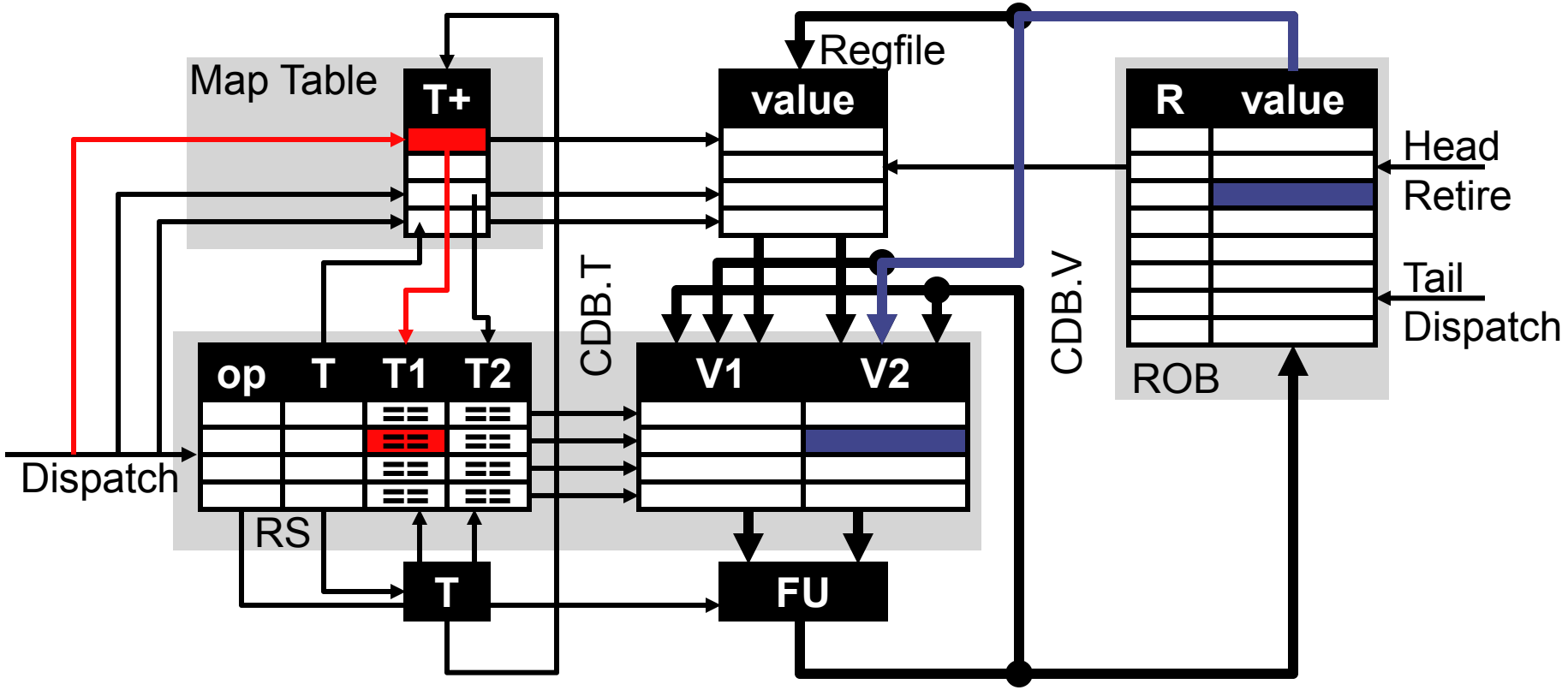
- **C (complete)**
  - Structural hazard (CDB)? **wait**
  - Write value into ROB entry indicated by RS tag
  - Mark ROB entry as complete
  - If not overwritten, mark Map Table entry "ready-in-ROB" bit (+)
- **R (retire)**
  - Insn at ROB head not complete ? **stall**
  - Handle any exceptions
  - Write ROB head value to register file
  - If store, write LSQ head to D$
  - Free ROB/LSQ entries

P6 Dispatch (D): Part I



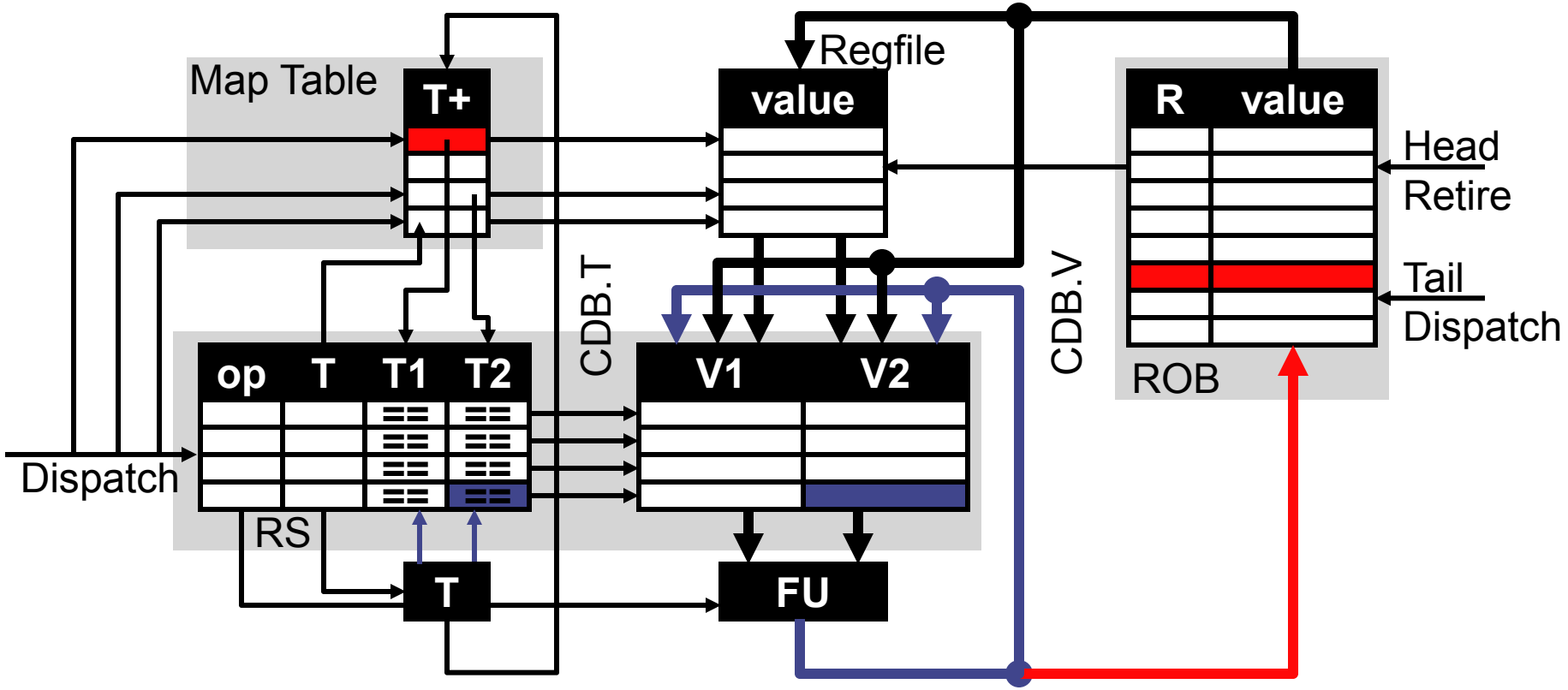
- RS/ROB full ? stall
- Allocate RS/ROB entries, assign ROB# to RS output tag
- Set output register Map Table entry to ROB#, clear “ready-in-ROB”

P6 Dispatch (D): Part II



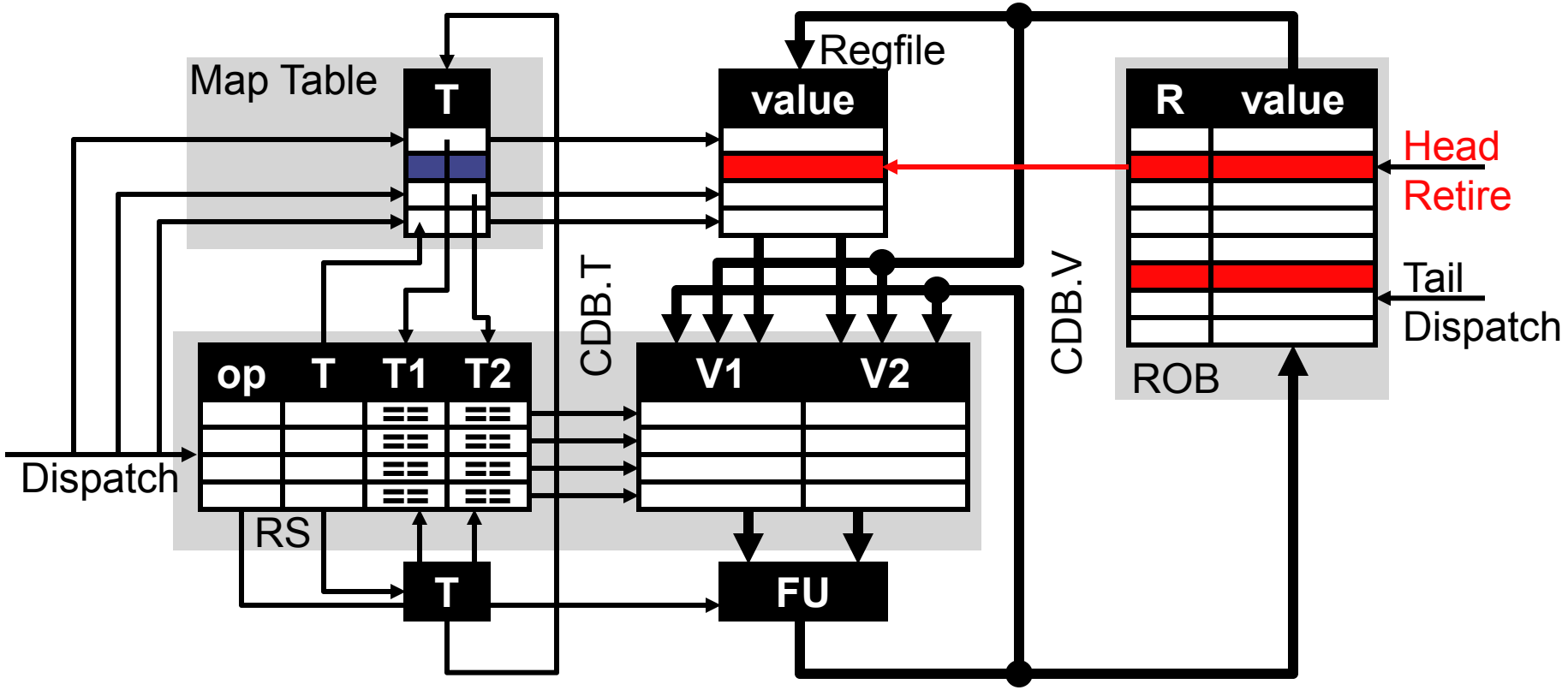
- Read tags for register inputs from Map Table
  - Tag==0 → copy value from Regfile (not shown)
  - Tag!=0 → copy Map Table tag to RS
  - Tag!=0+ → copy value from ROB

P6 Complete (C)



- Structural hazard (CDB) ? Stall : broadcast <value,tag> on CDB
- Write result into ROB, if still valid clear MapTable "ready-in-ROB" bit
- Match tags, write CDB.V into RS slots of dependent insns

P6 Retire (R)



- ROB head not complete ? stall : free ROB entry
- Write ROB head result to Regfile
- If still valid, clear Map Table entry

P6: Cycle 1

ROB

ht	#	Insn	R	V	S	X	C
ht	1	ldf X(r1),f1	f1				
	2	mulf f0,f1,f2					
	3	stf f2,Z(r1)					
	4	addi r1,4,r1					
	5	ldf X(r1),f1					
	6	mulf f0,f1,f2					
	7	stf f2,Z(r1)					

Map Table

Reg	T+
f0	
f1	ROB#1
f2	
r1	

CDB

T	V

Reservation Stations

#	FU	busy	op	T	T1	T2	V1	V2
1	ALU	no						
2	LD	yes	ldf	ROB#1				[r1]
3	ST	no						
4	FP1	no						
5	FP2	no						

set ROB# tag

allocate

P6: Cycle 2

ROB

ht	#	Insn	R	V	S	X	C
h	1	ldf X(r1),f1	f1		c2		
t	2	mulf f0,f1,f2	f2				
	3	stf f2,Z(r1)					
	4	addi r1,4,r1					
	5	ldf X(r1),f1					
	6	mulf f0,f1,f2					
	7	stf f2,Z(r1)					

Map Table

Reg	T+
f0	
f1	ROB#1
f2	ROB#2
r1	

CDB

T	V

Reservation Stations

#	FU	busy	op	T	T1	T2	V1	V2
1	ALU	no						
2	LD	yes	ldf	ROB#1				[r1]
3	ST	no						
4	FP1	yes	mulf	ROB#2		ROB#1	[f0]	
5	FP2	no						

set ROB# tag

allocate

P6: Cycle 3

ROB

ht	#	Insn	R	V	S	X	C
h	1	ldf X(r1),f1	f1		c2	c3	
	2	mulf f0,f1,f2	f2				
t	3	stf f2,Z(r1)					
	4	addi r1,4,r1					
	5	ldf X(r1),f1					
	6	mulf f0,f1,f2					
	7	stf f2,Z(r1)					

Map Table

Reg	T+
f0	
f1	ROB#1
f2	ROB#2
r1	

CDB

T	V

Reservation Stations

#	FU	busy	op	T	T1	T2	V1	V2
1	ALU	no						
2	LD	no						
3	ST	yes	stf	ROB#3	ROB#2			[r1]
4	FP1	yes	mulf	ROB#2		ROB#1	[f0]	
5	FP2	no						

free
allocate

P6: Cycle 4

ROB

ht	#	Insn	R	V	S	X	C
h	1	ldf X(r1),f1	f1	[f1]	c2	c3	c4
	2	mulf f0,f1,f2	f2		c4		
	3	stf f2,Z(r1)					
t	4	addi r1,4,r1	r1				
	5	ldf X(r1),f1					
	6	mulf f0,f1,f2					
	7	stf f2,Z(r1)					

Map Table

Reg	T+
f0	
f1	ROB#1+
f2	ROB#2
r1	ROB#4

CDB

T	V
ROB#1	[f1]

ldf finished

1. set "ready-in-ROB" bit
2. write result to ROB
3. CDB broadcast

Reservation Stations

#	FU	busy	op	T	T1	T2	V1	V2
1	ALU	yes	add	ROB#4			[r1]	
2	LD	no						
3	ST	yes	stf	ROB#3	ROB#2			[r1]
4	FP1	yes	mulf	ROB#2		ROB#1	[f0]	CDB.V
5	FP2	no						

allocate

ROB#1 ready
grab CDB.V

P6: Cycle 5

ROB

ht	#	Insn	R	V	S	X	C
	1	ldf X(r1),f1	f1	[f1]	c2	c3	c4
h	2	mulf f0,f1,f2	f2		c4	c5	
	3	stf f2,Z(r1)					
	4	addi r1,4,r1	r1		c5		
t	5	ldf X(r1),f1	f1				
	6	mulf f0,f1,f2					
	7	stf f2,Z(r1)					

Map Table

Reg	T+
f0	
f1	ROB#5
f2	ROB#2
r1	ROB#4

CDB

T	V

ldf retires

1. write ROB result to regfile

Reservation Stations

#	FU	busy	op	T	T1	T2	V1	V2
1	ALU	yes	add	ROB#4			[r1]	
2	LD	yes	ldf	ROB#5		ROB#4		
3	ST	yes	stf	ROB#3	ROB#2			[r1]
4	FP1	no						
5	FP2	no						

allocate

free

P6: Cycle 6

ROB

ht	#	Insn	R	V	S	X	C
	1	ldf X(r1),f1	f1	[f1]	c2	c3	c4
h	2	mulf f0,f1,f2	f2		c4	c5+	
	3	stf f2,Z(r1)					
	4	addi r1,4,r1	r1		c5	c6	
	5	ldf X(r1),f1	f1				
t	6	mulf f0,f1,f2	f2				
	7	stf f2,Z(r1)					

Map Table

Reg	T+
f0	
f1	ROB#5
f2	ROB#6
r1	ROB#4

CDB

T	V

Reservation Stations

#	FU	busy	op	T	T1	T2	V1	V2	
1	ALU	no							free
2	LD	yes	ldf	ROB#5		ROB#4			
3	ST	yes	stf	ROB#3	ROB#2			[r1]	
4	FP1	yes	mulf	ROB#6		ROB#5	[f0]		allocate
5	FP2	no							

P6: Cycle 7

ROB

ht	#	Insn	R	V	S	X	C
	1	ldf X(r1),f1	f1	[f1]	c2	c3	c4
h	2	mulf f0,f1,f2	f2		c4	c5+	
	3	stf f2,Z(r1)					
	4	addi r1,4,r1	r1	[r1]	c5	c6	c7
	5	ldf X(r1),f1	f1		c7		
t	6	mulf f0,f1,f2	f2				
	7	stf f2,Z(r1)					

Map Table

Reg	T+
f0	
f1	ROB#5
f2	ROB#6
r1	ROB#4+

CDB

T	V
ROB#4	[r1]

stall D (no free ST RS)

Reservation Stations

#	FU	busy	op	T	T1	T2	V1	V2
1	ALU	no						
2	LD	yes	ldf	ROB#5		ROB#4		CDB.V
3	ST	yes	stf	ROB#3	ROB#2			[r1]
4	FP1	yes	mulf	ROB#6		ROB#5	[f0]	
5	FP2	no						

ROB#4 ready
grab CDB.V

P6: Cycle 8

ROB

ht	#	Insn	R	V	S	X	C
	1	ldf X(r1),f1	f1	[f1]	c2	c3	c4
h	2	mulf f0,f1,f2	f2	[f2]	c4	c5+	c8
	3	stf f2,Z(r1)			c8		
	4	addi r1,4,r1	r1	[r1]	c5	c6	c7
	5	ldf X(r1),f1	f1		c7	c8	
t	6	mulf f0,f1,f2	f2				
	7	stf f2,Z(r1)					

Map Table

Reg	T+
f0	
f1	ROB#5
f2	ROB#6
r1	ROB#4+

CDB

T	V
ROB#2	[f2]

stall R for addi (in-order)

ROB#2 invalid in MapTable
don't set "ready-in-ROB"

Reservation Stations

#	FU	busy	op	T	T1	T2	V1	V2
1	ALU	no						
2	LD	no						
3	ST	yes	stf	ROB#3	ROB#2		[f2]	[r1]
4	FP1	yes	mulf	ROB#6		ROB#5	[f0]	
5	FP2	no						

ROB#2 ready
grab CDB.V

P6: Cycle 9

ROB

ht	#	Insn	R	V	S	X	C
	1	ldf X(r1),f1	f1	[f1]	c2	c3	c4
	2	mulf f0,f1,f2	f2	[f2]	c4	c5+	c8
h	3	stf f2,Z(r1)			c8	c9	
	4	addi r1,4,r1	r1	[r1]	c5	c6	c7
	5	ldf X(r1),f1	f1	[f1]	c7	c8	c9
	6	mulf f0,f1,f2	f2		c9		
t	7	stf f2,Z(r1)					

Map Table

Reg	T+
f0	
f1	ROB#5+
f2	ROB#6
r1	ROB#4+

CDB

T	V
ROB#5	[f1]

retire mulf

all pipe stages active at once!

Reservation Stations

#	FU	busy	op	T	T1	T2	V1	V2
1	ALU	no						
2	LD	no						
3	ST	yes	stf	ROB#7	ROB#6			ROB#4.V
4	FP1	yes	mulf	ROB#6		ROB#5	[f0]	CDB.V
5	FP2	no						

free, re-allocate

ROB#5 ready

grab CDB.V

P6: Cycle 10

ROB

ht	#	Insn	R	V	S	X	C
	1	ldf X(r1),f1	f1	[f1]	c2	c3	c4
	2	mulf f0,f1,f2	f2	[f2]	c4	c5+	c8
h	3	stf f2,Z(r1)			c8	c9	c10
	4	addi r1,4,r1	r1	[r1]	c5	c6	c7
	5	ldf X(r1),f1	f1	[f1]	c7	c8	c9
	6	mulf f0,f1,f2	f2		c9	c10	
t	7	stf f2,Z(r1)					

Map Table

Reg	T+
f0	
f1	ROB#5+
f2	ROB#6
r1	ROB#4+

CDB

T	V

Reservation Stations

#	FU	busy	op	T	T1	T2	V1	V2	
1	ALU	no							
2	LD	no							
3	ST	yes	stf	ROB#7	ROB#6			ROB#4.V	
4	FP1	no							free
5	FP2	no							

P6: Cycle 11

ROB

ht	#	Insn	R	V	S	X	C
	1	ldf X(r1),f1	f1	[f1]	c2	c3	c4
	2	mulf f0,f1,f2	f2	[f2]	c4	c5	c8
	3	stf f2,Z(r1)			c8	c9	c10
h	4	addi r1,4,r1	r1	[r1]	c5	c6	c7
	5	ldf X(r1),f1	f1	[f1]	c7	c8	c9
	6	mulf f0,f1,f2	f2		c9	c10	
t	7	stf f2,Z(r1)					

Map Table

Reg	T+
f0	
f1	ROB#5+
f2	ROB#6
r1	ROB#4+

CDB

T	V

retire stf

Reservation Stations

#	FU	busy	op	T	T1	T2	V1	V2
1	ALU	no						
2	LD	no						
3	ST	yes	stf	ROB#7	ROB#6			ROB#4.V
4	FP1	no						
5	FP2	no						

Precise State in P6

Point of ROB is maintaining **precise state**

- How does that work?
- Easy as 1,2,3
  1. Wait until last good insn retires, first bad insn at ROB head
  2. Clear contents of ROB, RS, and Map Table
  3. Start over
- Works because zero (0) means the right thing…
  - 0 in ROB/RS → entry is empty
  - Tag == 0 in Map Table → register is in regfile
- …and because regfile and D$ writes take place at R
- Example: page fault in first **`stf`**

P6: Cycle 9 (with precise state)

ROB

ht	#	Insn	R	V	S	X	C
	1	ldf X(r1),f1	f1	[f1]	c2	c3	c4
	2	mulf f0,f1,f2	f2	[f2]	c4	c5+	c8
h	3	stf f2,Z(r1)			c8	c9	
	4	addi r1,4,r1	r1	[r1]	c5	c6	c7
	5	ldf X(r1),f1	f1	[f1]	c7	c8	c9
	6	mulf f0,f1,f2	f2		c9		
t	7	stf f2,Z(r1)					

Map Table

Reg	T+
f0	
f1	ROB#5+
f2	ROB#6
r1	ROB#4+

CDB

T	V
ROB#5	[f1]

PAGE FAULT

Reservation Stations

#	FU	busy	op	T	T1	T2	V1	V2
1	ALU	no						
2	LD	no						
3	ST	yes	stf	ROB#7	ROB#6			ROB#4.V
4	FP1	yes	mulf	ROB#6		ROB#5	[f0]	CDB.V
5	FP2	no						

P6: Cycle 10 (with precise state)

ROB

ht	#	Insn	R	V	S	X	C
	1	ldf X(r1),f1	f1	[f1]	c2	c3	c4
	2	mulf f0,f1,f2	f2	[f2]	c4	c5+	c8
	3	stf f2,Z(r1)					
	4	addi r1,4,r1					
	5	ldf X(r1),f1					
	6	mulf f0,f1,f2					
	7	stf f2,Z(r1)					

Map Table

Reg	T+
f0	
f1	
f2	
r1	

CDB

T	V

faulting insn at ROB head?
CLEAR EVERYTHING

Reservation Stations

#	FU	busy	op	T	T1	T2	V1	V2
1	ALU	no						
2	LD	no						
3	ST	no						
4	FP1	no						
5	FP2	no						

P6: Cycle 11 (with precise state)

ROB

ht	#	Insn	R	V	S	X	C
	1	ldf X(r1),f1	f1	[f1]	c2	c3	c4
	2	mulf f0,f1,f2	f2	[f2]	c4	c5+	c8
ht	3	stf f2,Z(r1)					
	4	addi r1,4,r1					
	5	ldf X(r1),f1					
	6	mulf f0,f1,f2					
	7	stf f2,Z(r1)					

Map Table

Reg	T+
f0	
f1	
f2	
r1	

CDB

T	V

START OVER
(after OS fixes page fault)

Reservation Stations

#	FU	busy	op	T	T1	T2	V1	V2
1	ALU	no						
2	LD	no						
3	ST	yes	stf	ROB#3			[f4]	[r1]
4	FP1	no						
5	FP2	no						

P6: Cycle 12 (with precise state)

ROB

ht	#	Insn	R	V	S	X	C
	1	ldf X(r1),f1	f1	[f1]	c2	c3	c4
	2	mulf f0,f1,f2	f2	[f2]	c4	c5+	c8
h	3	stf f2,Z(r1)			c12		
t	4	addi r1,4,r1	r1				
	5	ldf X(r1),f1					
	6	mulf f0,f1,f2					
	7	stf f2,Z(r1)					

Map Table

Reg	T+
f0	
f1	
f2	
r1	ROB#4

CDB

T	V

Reservation Stations

#	FU	busy	op	T	T1	T2	V1	V2
1	ALU	yes	addi	ROB#4			[r1]	
2	LD	no						
3	ST	yes	stf	ROB#3			[f4]	[r1]
4	FP1	no						
5	FP2	no						

P6 Performance

In other words: what is the cost of precise state?

+ In general: same performance as "plain" Tomasulo
  - ROB is not a performance device
  - Maybe a little better (RS freed earlier → fewer struct hazards)
- Unless ROB is too small
  - In which case ROB struct hazards become a problem
- Rules of thumb for ROB size
  - At least N (width) * number of pipe stages between D and R
  - At least N * $t_{hit\text{-}L2}$
  - Can add a factor of 2 to both if you want
  - What is the rationale behind these?

P6 (Tomasulo+ROB) Redux

Popular design for a while

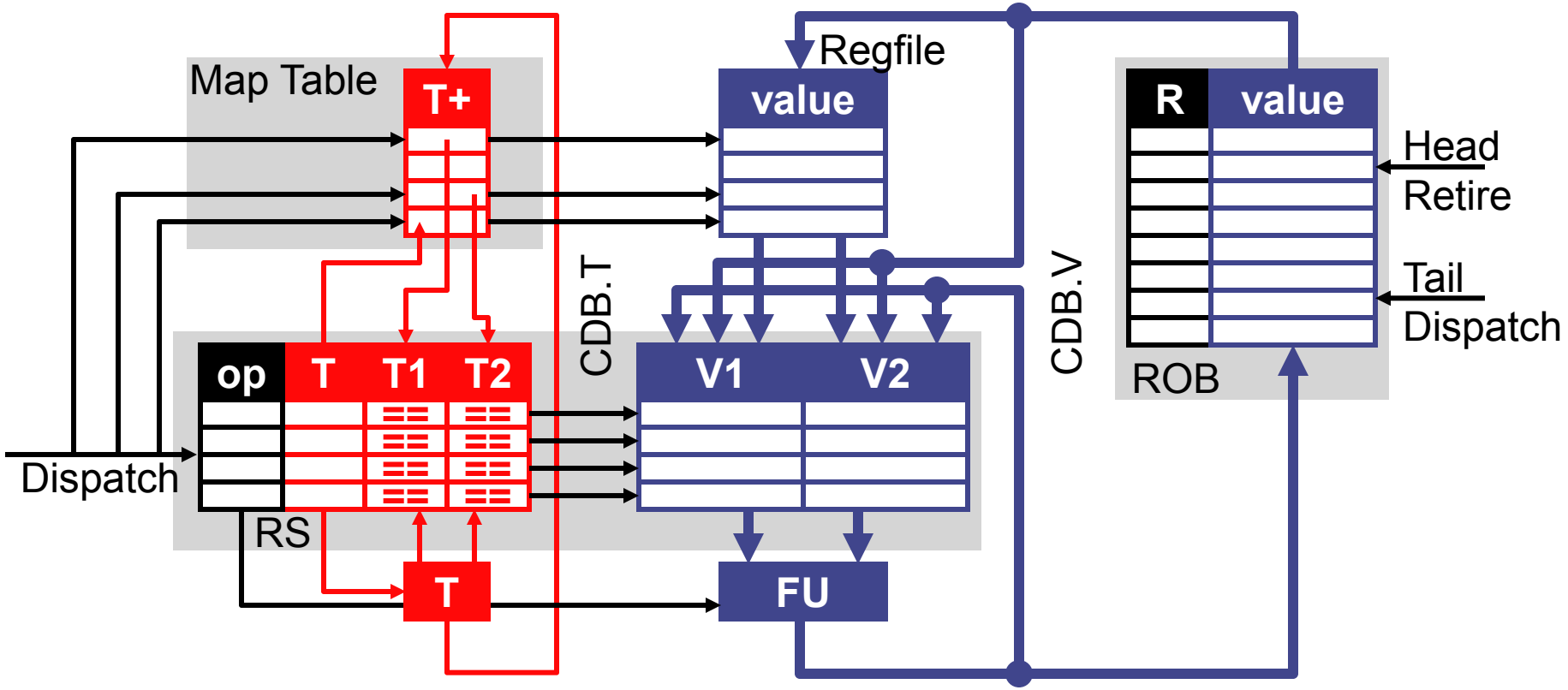
- (Relatively) easy to implement correctly
  - Anything goes wrong (mispredicted branch, fault, interrupt)?
  - Just clear everything and start again
- Examples: Intel PentiumPro, IBM/Motorola PowerPC, AMD K6

Actually making a comeback…

- Examples: Intel PentiumM

But went away for a while, why?

The Problem with P6



Problem for high performance implementations

- Too much **value movement** (regfile/ROB→RS→ROB→regfile)
- Multi-input muxes, long buses complicate routing and slow clock