Out-of-Order Execution: Reorder Buffer

Computer Architecture

Wei Wang

Text Book Chapters

- “Computer Architecture” Hennessy and Patterson, Chapter 3.6 “Hardware-based Speculation”.

Road Map

- Reorder Buffer
- An Example with Reorder Buffer
- Summary
- Acknowledgment

Putting Things Together

- We have learned many CPU implementation/optimization techniques so far...
  - Pipelining
    - Partition instruction execution into overlap-able stages
    - Enables more complex designs
  - Branch prediction and speculative execution
    - Achieve continuous flow of instructions
  - Scheduling and Hazard elimination
    - Dynamic scheduling (handles RAW hazards)
    - Register renaming (handles WAR and WAW hazards)
  - Multiple function units, register file ports
    - Enable multiple issue (superscalar) CPUs
    - Potentially reduce CPI<1
- All these techniques are incorporated into one CPU today.

Complexities from Complex Optimizations

- However, these optimization techniques do make other things worse...
  - Out-of-order completion
    - May cause Inconsistent data
    - Post-interrupt/mispredict write-backs change states
  - Incorrect speculations
    - CPU states cannot be updated by incorrect speculations
    - Need a place to hold results from speculative executions.
  - Precise Interrupts
    - All instructions before interrupt must complete
    - All instructions after interrupt must seem to never start

Example: Inconsistent Data

- Original code is written as,

```
1 product = x
2 ...
3 product_is_ready = True
```

  - When *product_is_ready* is *True*, *product* is always set.

- Out-of-Order completion may switch these two statements,

```
1 product_is_ready = True
2 ...
3 product = x
```

  - When an external observer see *product_is_ready* is *True*, it tries to read *product*. However, *product* may not be set yet.

Example: Precise Interrupts

- Recall the example 1 in the Tomasulo's Algorithm
  - After cycle 17, instruction 5 (DIV) starts execution with M_1 as the divisor.
  - If this DIV instruction causes a divide-by-zero exception, and the programmer tries to read the registers,
  - The programmer will see a Val_2 in R6, instead of M_1, and be confused.

Solving Three Problems with One Solution

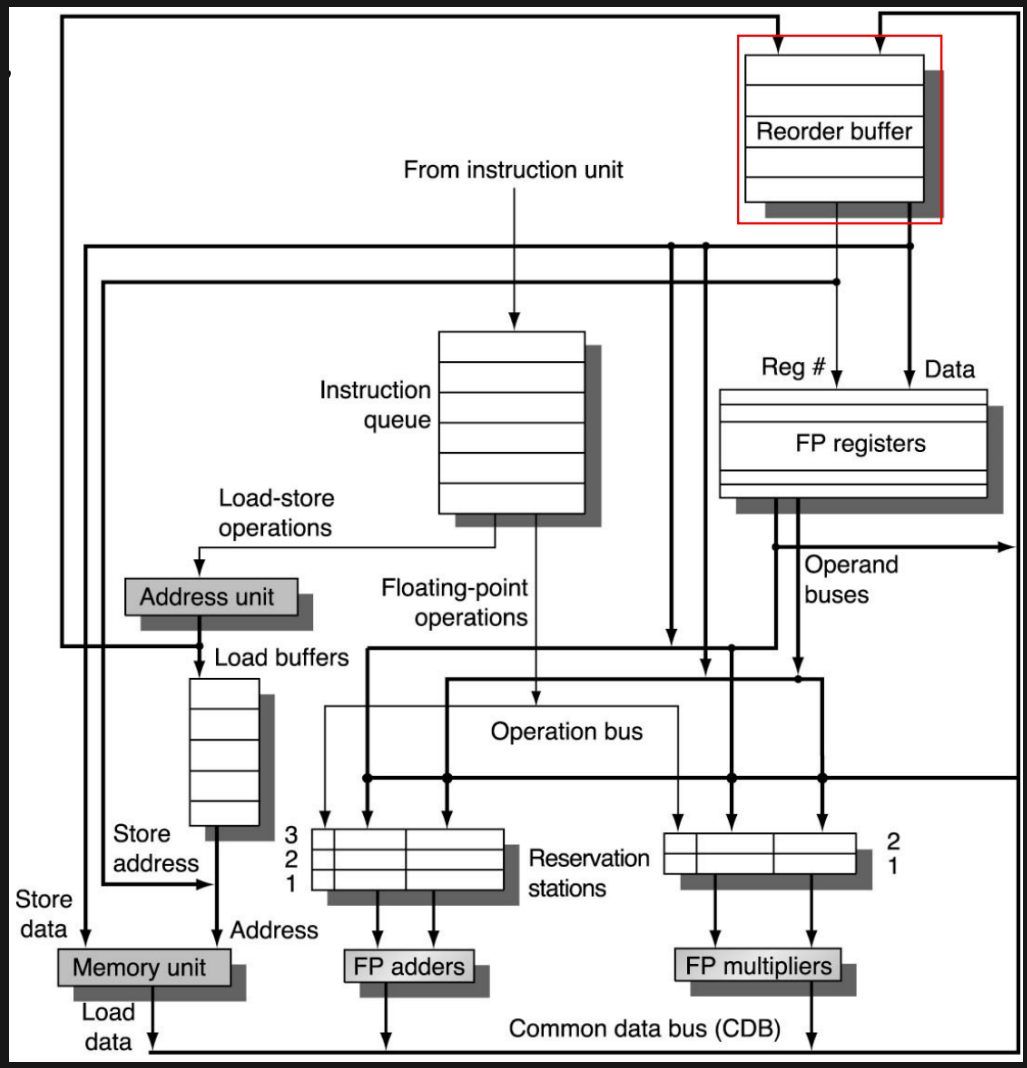
- Force in-order completion at WB and memory writes (stall when necessary)
  - Writes memory in-order, avoiding inconsistent data
  - Writes registers in-order, providing precise interrupts
  - Completion in-order, preventing speculative execution from completion until the branch is resolved.

The Commit Stage

- Allow out of order write-back
  - But values are not written-back to register file or memory
  - Only writes to a buffer
- Commit Stage actually writes to register file and memory.
  - Commit must be in-order
- Collect pre-commit instructions
  - In a Reorder Buffer (ROB)
    - holds completed but not committed instruction
  - ROB effectively contains a set of physical registers
    - similar to a reservation station
    - and becomes a bypass (forwarding) source

Reorder Buffer: HW Buffer for the Results of Uncommitted Instruction

- 3 fields: instr. type, destination, value
- Reorder buffer can be operand source => more registers like RS
- Use reorder buffer number instead of reservation station when execution completes
- Supplies operands between execution complete & commit
- Once operand commits, result is put into register
- Instructions commit in order
- As a result, its easy to undo speculated instructions on mispredicted branches or on exceptions



Four Steps of Speculative Tomasulo Algorithm

1. Issue (IS): get instruction from FP Op Queue
   - If reservation station and reorder buffer slot free, issue instr & send operands & get reorder buffer no. for destination (this stage sometimes called “dispatch”)
2. Execution (EX): operate on operands
   - When both operands ready then execute; if not ready, watch CDB for result; when both in reservation station, execute; checks RAW (sometimes called “issue”)
3. Write result (WB): finish execution
   - Write on Common Data Bus to all awaiting FUs & reorder buffer; mark reservation station available. (tags are now ROB ids not RS ids)
4. Commit (CM): update register with reorder result
   - When instr. at head of reorder buffer & result present, update register with result (or store to memory) and remove instr from reorder buffer. Mis-predicted branch flushes reorder buffer (sometimes called “graduation”)

Road Map

Reorder Buffer

An Example with Reorder Buffer

Summary

Acknowledgment

Reorder Buffer Example: Cycle 0

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
	Mul2	No						

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
Head, Tail	1					
	2					
	3					
	4					
	5					
	6					
	7					

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #							...
Busy							...
Value							

Reorder Buffer Example: Cycle 1

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
	Mul2	No						

Load Buffers:

	Busy	Address
Load1	Yes	34+R11
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
Head → Tail	1	Yes	LD R6, 34+R11	Issue	R6	
	2					
	3					
	4					
	5					
	6					
	7					

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #				#1			...
Busy				Yes			...
Value							

Reorder Buffer Example: Cycle 1

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
	Mul2	No						

Load Buffers:

	Busy	Address
Load1	Yes	34+R11
Load2	No	
Load3		

Instruction 1 occupies Load1.

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
Head →	1	Yes	LD R6, 34+R11	Issue	R6	
Tail →	2					
	3					
	4					
	5					
	6					
	7					

Issue instruction 1. Allocate ROB #1 for it.

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #				#1			...
Busy				Yes			...
Value							

Value of R6 will be freom ROB #1.

Reorder Buffer Example: Cycle 2

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
	Mul2	No						

Load Buffers:

	Busy	Address
Load1	Yes	34+R11
Load2	Yes	45+R13
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
Head →	1	Yes	LD R6, 34+R11	Ex1	R6	
Tail →	2	Yes	LD R2, 45+R13	Issue	R2	
	3					
	4					
	5					
	6					
	7					

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #		#2		#1			...
Busy		Yes		Yes			...
Value							

Reorder Buffer Example: Cycle 2

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
	Mul2	No						

Load Buffers:

	Busy	Address
Load1	Yes	34+R11
Load2	Yes	45+R13
Load3	No	

Instruction 2 occupies Load2.

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
Head →	1	Yes	LD R6, 34+R11	Ex1	R6	
Tail →	2	Yes	LD R2, 45+R13	Issue	R2	
	3					
	4					
	5					
	6					
	7					

Tail moves to the new ROB entry #2.

Instruction 2 issue at ROB #2

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #		#2		#1			...
Busy		Yes		Yes			...
Value							

Value of R2 will be freom ROB #2.

Reorder Buffer Example: Cycle 2

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
	Mul2	No						

Load Buffers:

	Busy	Address
Load1	Yes	34+R11
Load2	Yes	45+R13
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
Head →	1	Yes	LD R6, 34+R11	Ex1	R6	
Tail →	2	Yes	LD R2, 45+R13	Issue	R2	
	3					
	4					
	5					
	6					
	7					

Instruction 1 starts executing. Takes 2 cycles to finish.

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #		#2		#1			...
Busy		Yes		Yes			...
Value							

Reorder Buffer Example: Cycle 3

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	Yes	MUL		V(R4)	#2		#3
	Mul2	No						

Load Buffers:

	Busy	Address
Load1	Yes	34+R11
Load2	Yes	45+R13
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
Head →	1	Yes	LD R6, 34+R11	EX2	R6	
	2	Yes	LD R2, 45+R13	Ex1	R2	
Tail →	3	Yes	MUL R0, R2, R4	Issue	R0	
	4					
	5					
	6					
	7					

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3	#2		#1			...
Busy	Yes	Yes		Yes			...
Value							

Reorder Buffer Example: Cycle 3

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	Yes	MUL		V(R4)	#2		#3
	Mul2	No						

Load Buffers:

	Busy	Address
Load1	Yes	34+R11
Load2	Yes	45+R13
Load3	No	

Instruction 3 occupies Mul1. Src1 is from ROB #2, Src2's value (R4) is copied.

	Entry	Busy	Instruction	Stage	Dest	Value
Head →	1	Yes	LD R6, 34+R11	EX2	R6	
	2	Yes	LD R2, 45+R13	Ex1	R2	
Tail →	3	Yes	MUL R0, R2, R4	Issue	R0	
	4					
	5					

Tail moves to the new ROB entry #3.

Instruction 3 issued at ROB #3

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3	#2		#1			...
Busy	Yes	Yes		Yes			...
Value							

Value of R2 will be freom ROB #2.

Reorder Buffer Example: Cycle 3

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	Yes	MUL		V(R4)	#2		#3
	Mul2	No						

Load Buffers:

	Busy	Address
Load1	Yes	34+R11
Load2	Yes	45+R13
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
Head →	1	Yes	LD R6, 34+R11	EX2	R6	
	2	Yes	LD R2, 45+R13	Ex1	R2	
Tail →	3	Yes	MUL R0, R2, R4	Issue	R0	
	4					
	5					
	6					
	7					

Instruction 1 continues executing.

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3	#2		#1			...
Busy	Yes	Yes		Yes			...
Value							

Reorder Buffer Example: Cycle 3

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	Yes	MUL		V(R4)	#2		#3
	Mul2	No						

Load Buffers:

	Busy	Address
Load1	Yes	34+R11
Load2	Yes	45+R13
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
Head →	1	Yes	LD R6, 34+R11	EX2	R6	
	2	Yes	LD R2, 45+R13	Ex1	R2	
Tail →	3	Yes	MUL R0, R2, R4	Issue	R0	
	4					
	5					
	6					
	7					

Instruction 2 starts executing.

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3	#2		#1			...
Busy	Yes	Yes		Yes			...
Value							

Reorder Buffer Example: Cycle 4

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	Yes	SUB	M_1			#2	#4
	Add2	No						
	Add3	No						
	Mul1	Yes	MUL		V(R4)	#2		#3
	Mul2	No						

Load Buffers:

	Busy	Address
Load1	No	
Load2	Yes	45+R13
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
Head →	1	Yes	LD R6, 34+R11	WB	R6	M_1
	2	Yes	LD R2, 45+R13	EX2	R2	
	3	Yes	MUL R0, R2, R4	Issue	R0	
Tail →	4	Yes	SUB R8, R6, R2	Issue	R8	
	5					
	6					
	7					

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3	#2		#1	#4		...
Busy	Yes	Yes		Yes	Yes		...
Value							

Reorder Buffer Example: Cycle 4

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	Yes	SUB	M_1			#2	#4
	Add2	No						

Instruction 4 occupies Add11. Src1 is from ROB #1 (R6) with value of M_1, Src2 is from ROB #2.

Load Buffers:

	Busy	Address
Load1	No	
Load2	Yes	45+R13
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
Head →	1	Yes	LD R6, 34+R11	WB	R6	M_1
	2	Yes	LD R2, 45+R13	EX2	R2	
	3	Yes	MUL R0, R2, R4	Issue	R0	
Tail →	4	Yes	SUB R8, R6, R2	Issue	R8	
	5					
	6					

Tail moves to the new ROB entry #4.

Instruction 4 issued at ROB #4

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3	#2		#1	#4		...
Busy	Yes	Yes		Yes	Yes		...
Value							

Value of R8 will be from ROB #4.

Reorder Buffer Example: Cycle 4

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	Yes	SUB	M_1			#2	#4
	Add2	No						
	Add3	No						
	Mul1	Yes	MUL		V(R4)	#2		#3
	Mul2	No						

Load Buffers:

	Busy	Address
Load1	No	
Load2	Yes	45+R13
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
Head →	1	Yes	LD R6, 34+R11	WB	R6	M_1
	2	Yes	LD R2, 45+R13	EX2	R2	
	3	Yes	MUL R0, R2, R4	Issue	R0	
Tail →	4	Yes	SUB R8, R6, R2	Issu	R8	
	5					
	6					
	7					

Instruction 2 continues executing.

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3	#2		#1	#4		...
Busy	Yes	Yes		Yes	Yes		...
Value							

Reorder Buffer Example: Cycle 4

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	Yes	SUB	M_1			#2	#4
	Add2	No						
	Add3	No						
	Mul1	Yes	MUL		V(R4)	#2		#3
	Mul2	No						

Load Buffers:

	Busy	Address
Load1	No	
Load2	Yes	45+R13
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
Head →	1	Yes	LD R6, 34+R11	WB	R6	M_1
	2	Yes	LD R2, 45+R13	EX2	R2	
	3	Yes	MUL R0, R2, R4	Issue	R0	
Tail →	4	Yes	SUB R8, R6, R2			
	5					
	6					
	7					

Instruction 1 writes back. Value M_1 and id "#1" are posted to CDB. ROB #1 picks up value M_1.

Register File:

	R0	R2	R4	R6	R8		...
Reorder #	#3	#2		#1	#4		...
Busy	Yes	Yes		Yes	Yes		...
Value							

Reorder Buffer Example: Cycle 5

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	Yes	SUB	M_1	M_2			#4
	Add2	No						
	Add3	No						
	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
Head →	2	Yes	LD R2, 45+R13	WB	R2	M_2
	3	Yes	MUL R0, R2, R4	Issue	R0	
	4	Yes	SUB R8, R6, R2	Issue	R8	
Tail →	5	Yes	DIV R10, R0, R6	Issue	R10	
	6					
	7					

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3	#2			#4	#5	...
Busy	Yes	Yes		No	Yes	Yes	...
Value				M_1			

Reorder Buffer Example: Cycle 5

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	Yes	SUB	M_1	M_2			#4
	Add2	No						
	Add3	No						
	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Instruction 5 occupies Add1. Src2 is from ROB #1 (R6) with value of M_1, Src1 is from ROB #3.

					t	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
Head →	2	Yes	LD R2, 45+R13	WB	R2	M_2
	3	Yes	MUL R0, R2, R4	Issue	R0	
	4	Yes	SUB R8, R6, R2	Issue	R8	
Tail →	5	Yes	DIV R10, R0, R6	Issue	R10	
	6					
	7					

Tail moves to the new ROB entry #5.

Instruction 5 issued at ROB #5

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3	#2			#4	#5	...
Busy	Yes	Yes		No	Yes	Yes	...
Value				M_1			

Value of R10 will be from ROB #5

Reorder Buffer Example: Cycle 5

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	Yes	SUB	M_1	M_2			#4
	Add2	No						
	Add3	No						
	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Mul1 check CDB id "#2", which matches its src1, so it picks up the value M_2.

Re

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
Head →	2	Yes	LD R2, 45+R13	WB	R2	M_2
	3	Yes	MUL R0, R2, R4	Issu	R0	
	4	Yes	SUB R8, R6, R2	Iss	R8	
Tail →	5	Yes	DIV R10, R0, R6		10	
	6					
	7					

Instruction 2 writes back. Value M_2 and id "#2" are posted to CDB. ROB #2 picks up value M_2.

Register File:

	R0	R2	R4	R6	R8		...
Reorder #	#3	#2			#4		
Busy	Yes	Yes		No	Yes	Yes	
Value				M_1			

Reorder Buffer Example: Cycle 5

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	Yes	SUB	M_1	M_2			#4
	Add2	No						
	Add3	No						
	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

Head moves down.

	Busy	Instruction	Stage	Dest	Value
1	Yes	LD R6, 34+R11	Commit	R6	M_1
Head → 2	Yes	LD R2, 45+R13	WB	R2	M_2
3	Yes	MUL R0, R2, R4	Issue	R0	
4	Yes	SUB R8, R6, R2			
Tail → 5	Yes	DIV R10, R0, R6			
6					
7					

Instruction 1 commits.

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3	#2			#4	#5	...
Busy	Yes	Yes		No	Yes	Yes	...
Value				M_1			

R6 now has value M_1 from ROB #1. R6 status sets to Not Busy.

Reorder Buffer Example: Cycle 6

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
1	Add1	Yes	SUB	M_1	M_2			#4
	Add2	Yes	ADD		M_2	#4		#6
	Add3	No						
9	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	Ex1	R0	
	4	Yes	SUB R8, R6, R2	Ex1	R8	
	5	Yes	DIV R10, R0, R6	Issue	R10	
Tail →	6	Yes	ADD R6, R8, R2	Issue	R6	
	7					

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3			#6	#4	#5	...
Busy	Yes	No		Yes	Yes	Yes	...
Value		M_2		M_1			

Reorder Buffer Example: Cycle 6

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	Dest
1	Add1	Yes	SUB	M_1	M_2			#4
	Add2	Yes	ADD		M_2	#4		#6
	Add3	No						

Instruction 6 occupies Add2. Src1 is from #4, Src2 is from ROB #2 with value of M_2.

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	Ex1	R0	
	4	Yes	SUB R8, R6, R2	Ex1	R8	
	5	Yes	DIV R10, R0, R6	Issue	R10	
Tail →	6	Yes	ADD R6, R8, R2	Issue	R6	
	7					

Tail moves to the new ROB entry #6.

Instruction 6 issued at ROB #6

	R0	R2	R4		R10	...
	#3			#6	#4	#5	...
Busy	Yes	No		Yes	Yes	Yes	...
Value		M_2		M_1			

Value of R2 will be from ROB #6

Reorder Buffer Example: Cycle 6

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
1	Add1	Yes	SUB	M_1	M_2			#4
	Add2	Yes	ADD		M_2	#4		#6
	Add3	No						
9	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	Ex1	R0	
	4	Yes	SUB R8, R6, R2	Ex1	R8	
	5	Yes	DIV R10, R0, R6	Issu	R10	
Tail →	6	Yes	ADD R6, R8, R2		6	
	7					

Instruction 3 starts execut-ing. 9 cycles remain.

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3			#6	#4	#5	...
Busy	Yes	No		Yes	Yes	Yes	...
Value		M_2		M_1			

Reorder Buffer Example: Cycle 6

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
1	Add1	Yes	SUB	M_1	M_2			#4
	Add2	Yes	ADD		M_2	#4		#6
	Add3	No						
9	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	Ex1	R0	
	4	Yes	SUB R8, R6, R2	Ex1	R8	
	5	Yes	DIV R10, R0, R6	Issu	R10	
Tail →	6	Yes	ADD R6, R8, R2	Issu	R6	
	7					

Instruction 4 starts executing. 1 cycle remains.

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3			#6	#4	#5	...
Busy	Yes	No		Yes	Yes	Yes	...
Value		M_2		M_1			

Reorder Buffer Example: Cycle 6

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
1	Add1	Yes	SUB	M_1	M_2			#4
	Add2	Yes	ADD		M_2	#4		#6
	Add3	No						
9	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

Head moves down.

	Busy	Instruction	Stage	Dest	Value
1	Yes	LD R6, 34+R11	Commit	R6	M_1
2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head → 3	Yes	MUL R0, R2, R4	Ex1	R0	
4	Yes	SUB R8, R6, R2			
5	Yes	DIV R10, R0, R6			
Tail → 6	Yes	ADD R6, R8, R2			
7					

Instruction 2 commits.

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3			#6	#4	#5	...
Busy	Yes	No		Yes	Yes	Yes	...
Value		M_2					

R2 now has value M_2 from ROB #2. R2 status sets to Not Busy.

Reorder Buffer Example: Cycle 7

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
0	Add1	Yes	SUB	M_1	M_2			#4
	Add2	Yes	ADD		M_2	#4		#6
	Add3	No						
8	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	Ex2	R0	
	4	Yes	SUB R8, R6, R2	Ex2	R8	
	5	Yes	DIV R10, R0, R6	Issue	R10	
Tail →	6	Yes	ADD R6, R8, R2	Issue	R6	
	7					

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3			#6	#4	#5	...
Busy	Yes	No		Yes	Yes	Yes	...
Value		M_2		M_1			

Reorder Buffer Example: Cycle 7

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
0	Add1	Yes	SUB	M_1	M_2			#4
	Add2	Yes	ADD		M_2	#4		#6
	Add3	No						
8	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	Ex2	R0	
	4	Yes	SUB R8, R6, R2	Ex2	R8	
	5	Yes	DIV R10, R0, R6	Issu	R10	
Tail →	6	Yes	ADD R6, R8, R2			
	7					

Instruction 3 continues executing.

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3			#6	#4	#5	...
Busy	Yes	No		Yes	Yes	Yes	...
Value		M_2		M_1			

Reorder Buffer Example: Cycle 7

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
0	Add1	Yes	SUB	M_1	M_2			#4
	Add2	Yes	ADD		M_2	#4		#6
	Add3	No						
8	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	Ex2	R0	
	4	Yes	SUB R8, R6, R2	Ex2	R8	
	5	Yes	DIV R10, R0, R6	Issu	R10	
Tail →	6	Yes	ADD R6, R8, R2	Issu	R6	
	7					

Instruction 4 finishes executing.

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3			#6	#4	#5	...
Busy	Yes	No		Yes	Yes	Yes	...
Value		M_2		M_1			

Reorder Buffer Example: Cycle 8

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
0	Add2	Yes	ADD	Val_1	M_2			#6
	Add3	No						
7	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	EX3	R0	
	4	Yes	SUB R8, R6, R2	WB	R8	Val_1
	5	Yes	DIV R10, R0, R6	Issue	R10	
Tail →	6	Yes	ADD R6, R8, R2	Issue	R6	
	7					

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3			#6	#4	#5	...
Busy	Yes	No		Yes	Yes	Yes	...
Value		M_2		M_1			

Reorder Buffer Example: Cycle 8

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
0	Add2	Yes	ADD	Val_1	M_2			#6
	Add3	No						
7	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	EX3	R0	
	4	Yes	SUB R8, R6, R2	WB	R8	Val_1
	5	Yes	DIV R10, R0, R6	Issue	R10	
Tail →	6	Yes	ADD R6, R8, R2			
	7					

Instruction 3 continues executing.

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3			#6	#4	#5	...
Busy	Yes	No		Yes	Yes	Yes	...
Value		M_2		M_1			

Reorder Buffer Example: Cycle 8

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	Dest
	Add1	No						
0	Add2	Yes	ADD	Val_1	M_2			#6
	Add3	No						
7								

Add2 check CDB id "#4", which matches its src1, so it picks up the value Val_1.

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	EX3	R0	
	4	Yes	SUB R8, R6, R2	WB	R8	Val_1
	5	Yes	DIV R10, R0, R6	Issue	R10	
Tail →	6	Yes	ADD R6, R8, R2	Issue	R6	
	7					

Instruction 4 writes back. Value Val_1 and id "#4" are posted to CDB. ROB #4 picks up value Val_1.

Register File:

	R0	R2	R4	R6	R8
Reorder #	#3			#6	#4
Busy	Yes	No		Yes	Yes
Value		M_2		M_1	

Reorder Buffer Example: Cycle 9

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
1	Add2	Yes	ADD	Val_1	M_2			#6
	Add3	No						
6	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	EX4	R0	
	4	Yes	SUB R8, R6, R2	WB	R8	Val_1
	5	Yes	DIV R10, R0, R6	Issue	R10	
Tail →	6	Yes	ADD R6, R8, R2	Ex1	R6	
	7					

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3			#6	#4	#5	...
Busy	Yes	No		Yes	Yes	Yes	...
Value		M_2		M_1			

Reorder Buffer Example: Cycle 9

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
1	Add2	Yes	ADD	Val_1	M_2			#6
	Add3	No						
6	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	EX4	R0	
	4	Yes	SUB R8, R6, R2	WB	R8	Val_1
	5	Yes	DIV R10, R0, R6	Issue	R10	
Tail →	6	Yes	ADD R6, R8, R2			
	7					

Instruction 3 continues executing.

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3			#6	#4	#5	...
Busy	Yes	No		Yes	Yes	Yes	...
Value		M_2		M_1			

Reorder Buffer Example: Cycle 9

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
1	Add2	Yes	ADD	Val_1	M_2			#6
	Add3	No						
6	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	EX4	R0	
	4	Yes	SUB R8, R6, R2	WB	R8	Val_1
	5	Yes	DIV R10, R0, R6	Issue	R10	
Tail →	6	Yes	ADD R6, R8, R2	Ex1	R6	
	7					

Instruction 6 starts executing.

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3			#6	#4		
Busy	Yes	No		Yes	Yes	Yes	...
Value		M_2		M_1			

Reorder Buffer Example: Cycle 10

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
0	Add2	Yes	ADD	Val_1	M_2			#6
	Add3	No						
5	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	EX5	R0	
	4	Yes	SUB R8, R6, R2	WB	R8	Val_1
	5	Yes	DIV R10, R0, R6	Issue	R10	
Tail →	6	Yes	ADD R6, R8, R2	Ex2	R6	
	7					

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3			#6	#4	#5	...
Busy	Yes	No		Yes	Yes	Yes	...
Value		M_2		M_1			

Reorder Buffer Example: Cycle 10

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
0	Add2	Yes	ADD	Val_1	M_2			#6
	Add3	No						
5	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	EX5	R0	
	4	Yes	SUB R8, R6, R2	WB	R8	Val_1
	5	Yes	DIV R10, R0, R6	Issue	R10	
Tail →	6	Yes	ADD R6, R8, R2			
	7					

Instruction 3 continues executing.

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3			#6	#4	#5	...
Busy	Yes	No		Yes	Yes	Yes	...
Value		M_2		M_1			

Reorder Buffer Example: Cycle 10

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	Dest
	Add1	No						
0	Add2	Yes	ADD	Val_1	M_2			#6
	Add3	No						
5	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	EX5	R0	
	4	Yes	SUB R8, R6, R2	WB	R8	Val_1
	5	Yes	DIV R10, R0, R6	Issue	R10	
Tail →	6	Yes	ADD R6, R8, R2	Ex2	R6	
	7					

Instruction 6 finishes executing.

Register File:

	R0	R2	R4	R6	R8		
Reorder #	#3			#6	#4		
Busy	Yes	No		Yes	Yes	Yes	...
Value		M_2		M_1			

Reorder Buffer Example: Cycle 11

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
4	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	EX6	R0	
	4	Yes	SUB R8, R6, R2	WB	R8	Val_1
	5	Yes	DIV R10, R0, R6	Issue	R10	
Tail →	6	Yes	ADD R6, R8, R2	WB	R6	Val_2
	7					

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3			#6	#4	#5	...
Busy	Yes	No		Yes	Yes	Yes	...
Value		M_2		M_1			

Reorder Buffer Example: Cycle 11

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
4	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	EX6	R0	
	4	Yes	SUB R8, R6, R2	WB	R8	Val_1
	5	Yes	DIV R10, R0, R6	Issu	R10	
Tail →	6	Yes	ADD R6, R8, R2			Val_2
	7					

Instruction 3 continues executing.

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3			#6	#4	#5	...
Busy	Yes	No		Yes	Yes	Yes	...
Value		M_2		M_1			

Reorder Buffer Example: Cycle 11

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
4	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	EX6	R0	
	4	Yes	SUB R8, R6, R2	WB	R8	Val_1
	5	Yes	DIV R10, R0, R6	Issue	R10	
Tail →	6	Yes	ADD R6, R8, R2	WB	R6	Val_2
	7					

Register File:

	R0	R2	R4	R6	R8
Reorder #	#3			#6	#4
Busy	Yes	No		Yes	Yes
Value		M_2		M_1	

Instruction 6 writes back. Value Val_2 and id "#6" are posted to CDB. ROB #6 picks up value Val_2.

Reorder Buffer Example: Cycle 15

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
0	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	EX10	R0	
	4	Yes	SUB R8, R6, R2	WB	R8	Val_1
	5	Yes	DIV R10, R0, R6	Issue	R10	
Tail →	6	Yes	ADD R6, R8, R2	WB	R6	Val_2
	7					

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3			#6	#4	#5	...
Busy	Yes	No		Yes	Yes	Yes	...
Value		M_2		M_1			

Reorder Buffer Example: Cycle 15

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
0	Mul1	Yes	MUL	M_2	V(R4)			#3
	Mul2	Yes	DIV		M_1	#3		#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	EX10	R0	
	4	Yes	SUB R8, R6, R2	WB	R8	Val_1
	5	Yes	DIV R10, R0, R6	Issue	R10	
Tail →	6	Yes	ADD R6, R8, R2			Val_2
	7					

Instruction 3 finishes executing.

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3			#6	#4	#5	...
Busy	Yes	No		Yes	Yes	Yes	...
Value		M_2		M_1			

Reorder Buffer Example: Cycle 16

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
	Mul2	Yes	DIV	Val_3	M_1			#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	WB	R0	Val_3
	4	Yes	SUB R8, R6, R2	WB	R8	Val_1
	5	Yes	DIV R10, R0, R6	Issue	R10	
Tail →	6	Yes	ADD R6, R8, R2	WB	R6	Val_2
	7					

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #	#3			#6	#4	#5	...
Busy	Yes	No		Yes	Yes	Yes	...
Value		M_2		M_1			

Reorder Buffer Example: Cycle 16

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
	Mul2	Yes	DIV	Val_3	M_1			#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Mul2 check CDB id "#3", which matches its src1, so it picks up the value Val_3.

						Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
Head →	3	Yes	MUL R0, R2, R4	WB	R0	Val_3
	4	Yes	SUB R8, R6, R2	WB	R8	Val_1
	5	Yes	DIV R10, R0, R6	Issued	R10	
Tail →	6	Yes	ADD R6, R8, R2	Issued	R6	Val_2
	7					

Instruction 3 writes back. Value Val_3 and id "#3" are posted to CDB. ROB #6 picks up value Val_3.

Register File:

	R0	R2	R4	R6	R8
Reorder #	#3			#6	#4
Busy	Yes	No		Yes	Yes
Value		M_2		M_1	

Reorder Buffer Example: Cycle 17

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
39	Mul2	Yes	DIV	Val_3	M_1			#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
	3	Yes	MUL R0, R2, R4	Commit	R0	Val_3
Head →	4	Yes	SUB R8, R6, R2	WB	R8	Val_1
	5	Yes	DIV R10, R0, R6	Ex1	R10	
Tail →	6	Yes	ADD R6, R8, R2	WB	R6	Val_2
	7					

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #				#6	#4	#5	...
Busy	No	No		Yes	Yes	Yes	...
Value	Val_3	M_2		M_1			

Reorder Buffer Example: Cycle 17

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
39	Mul2	Yes	DIV	Val_3	M_1			#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
	3	Yes	MUL R0, R2, R4	Commit	R0	Val_3
Head →	4	Yes	SUB R8, R6, R2	WB	R8	Val_1
	5	Yes	DIV R10, R0, R6	Ex1	R10	
Tail →	6	Yes	ADD R6, R8, R2	WB	R6	Val_2
	7					

Instruction 5 starts executing.

Register File:

	R0	R2	R4	R6	R		
Reorder #				#6	#4	#5	...
Busy	No	No		Yes	Yes	Yes	...
Value	Val_3	M_2		M_1			

Reorder Buffer Example: Cycle 17

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
39	Mul2	Yes	DIV	Val_3	M_1			#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

Entry	Busy	Instruction	Stage	Dest	Value
1	Yes	LD R6, 34+R11	Commit	R6	M_1
2	Yes	LD R2, 45+R13	Commit	R2	M_2
3	Yes	MUL R0, R2, R4	Commit	R0	Val_3
Head → 4	Yes	SUB R8, R6, R2	WB	R8	Val_1
5	Yes	DIV R10, R0, R6	Ex1	R10	
Tail → 6	Yes	ADD R6, R8, R2			Val_2
7					

Head moves down.

Instruction 3 commits.

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #				#6	#4	#5	...
Busy	No	No		Yes	Yes	Yes	...
Value	Val_3	M_2					

R0 now has value Val_3 from ROB #3. R0 status sets to Not Busy.

Reorder Buffer Example: Cycle 18

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
38	Mul2	Yes	DIV	Val_3	M_1			#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
	3	Yes	MUL R0, R2, R4	Commit	R0	Val_3
	4	Yes	SUB R8, R6, R2	Commit	R8	Val_1
Head →	5	Yes	DIV R10, R0, R6	Ex2	R10	
Tail →	6	Yes	ADD R6, R8, R2	WB	R6	Val_2
	7					

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #				#6		#5	...
Busy	No	No		Yes	No	Yes	...
Value	Val_3	M_2		M_1	Val_1		

Reorder Buffer Example: Cycle 18

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
38	Mul2	Yes	DIV	Val_3	M_1			#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
	3	Yes	MUL R0, R2, R4	Commit	R0	Val_3
	4	Yes	SUB R8, R6, R2	Commit	R8	Val_1
Head →	5	Yes	DIV R10, R0, R6	Ex2	R10	
Tail →	6	Yes	ADD R6, R8, R2	WB	R6	Val_2
	7					

Instruction 5 continues executing.

Register File:

	R0	R2	R4	R6	R		
Reorder #				#6		#5	...
Busy	No	No		Yes	No	Yes	...
Value	Val_3	M_2		M_1	Val_1		

Reorder Buffer Example: Cycle 18

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
38	Mul2	Yes	DIV	Val_3	M_1			#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

Entry	Busy	Instruction	Stage	Dest	Value
1	Yes	LD R6, 34+R11	Commit	R6	M_1
2	Yes	LD R2, 45+R13	Commit	R2	M_2
3	Yes	MUL R0, R2, R4	Commit	R0	Val_3
4	Yes	SUB R8, R6, R2	Commit	R8	Val_1
Head→5	Yes	DIV R10, R0, R6	Ex2	R10	
Tail→6	Yes	ADD R6, R8, R2	WB	R6	Val_2
7					

Head moves down.

Instruction 4 commits.

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #				#6		#5	...
Busy	No	No		Yes	No	Yes	...
Value	Val_3	M_2		M_1	Val_1		

R8 now has value Val_1 from ROB #4. R8 status sets to Not Busy.

Reorder Buffer Example: Cycle 56

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
0	Mul2	Yes	DIV	Val_3	M_1			#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
	3	Yes	MUL R0, R2, R4	Commit	R0	Val_3
	4	Yes	SUB R8, R6, R2	Commit	R8	Val_1
Head →	5	Yes	DIV R10, R0, R6	Ex40	R10	
Tail →	6	Yes	ADD R6, R8, R2	WB	R6	Val_2
	7					

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #				#6		#5	...
Busy	No	No		Yes	No	Yes	...
Value	Val_3	M_2		M_1	Val_1		

Reorder Buffer Example: Cycle 56

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
0	Mul2	Yes	DIV	Val_3	M_1			#5

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
	3	Yes	MUL R0, R2, R4	Commit	R0	Val_3
	4	Yes	SUB R8, R6, R2	Commit	R8	Val_1
Head →	5	Yes	DIV R10, R0, R6	Ex40	R10	
Tail →	6	Yes	ADD R6, R8, R2	WB	R6	Val_2
	7					

Instruction 5 finishes executing.

Register File:

	R0	R2	R4	R6	R8		
Reorder #				#6		#5	...
Busy	No	No		Yes	No	Yes	...
Value	Val_3	M_2		M_1	Val_1		

Reorder Buffer Example: Cycle 57

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
	Mul2	No						

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
	3	Yes	MUL R0, R2, R4	Commit	R0	Val_3
	4	Yes	SUB R8, R6, R2	Commit	R8	Val_1
Head →	5	Yes	DIV R10, R0, R6	WB	R10	Val_4
Tail →	6	Yes	ADD R6, R8, R2	WB	R6	Val_2
	7					

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #				#6		#5	...
Busy	No	No		Yes	No	Yes	...
Value	Val_3	M_2		M_1	Val_1		

Reorder Buffer Example: Cycle 57

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
	Mul2	No						

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
	3	Yes	MUL R0, R2, R4	Commit	R0	Val_3
	4	Yes	SUB R8, R6, R2	Commit	R8	Val_1
Head →	5	Yes	DIV R10, R0, R6	WB	R10	Val_4
Tail →	6	Yes	ADD R6, R8, R2	WB	R6	Val_2
	7					

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #				#6		#5	...
Busy	No	No		Yes	No	Yes	...
Value	Val_3	M_2		M_1	Val_1		

Reorder Buffer Example: Cycle 57

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
	Mul2	No						

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

	Entry	Busy	Instruction	Stage	Dest	Value
	1	Yes	LD R6, 34+R11	Commit	R6	M_1
	2	Yes	LD R2, 45+R13	Commit	R2	M_2
	3	Yes	MUL R0, R2, R4	Commit	R0	Val_3
	4	Yes	SUB R8, R6, R2	Commit	R8	Val_1
Head →	5	Yes	DIV R10, R0, R6	WB	R10	Val_4
Tail →	6	Yes	ADD R6, R8, R2	WB	R6	Val_2
	7					

Register File:

	R0	R2	R4	R6	R
Reorder #				#6	
Busy	No	No		Yes	N
Value	Val_3	M_2		M_1	Va

Instruction 5 writes back. Mul2 posts Value Val_4 and id "#4" to CDB. ROB #4 picks up value Val_4.

Reorder Buffer Example: Cycle 58

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
	Mul2	No						

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

Entry	Busy	Instruction	Stage	Dest	Value
1	Yes	LD R6, 34+R11	Commit	R6	M_1
2	Yes	LD R2, 45+R13	Commit	R2	M_2
3	Yes	MUL R0, R2, R4	Commit	R0	Val_3
4	Yes	SUB R8, R6, R2	Commit	R8	Val_1
5	Yes	DIV R10, R0, R6	Commit	R10	Val_4
6	Yes	ADD R6, R8, R2	WB	R6	Val_2
7					

Head → 6

Tail → 7

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #				#6			...
Busy	No	No		Yes	No	No	...
Value	Val_3	M_2		M_1	Val_1	Val_4	

Reorder Buffer Example: Cycle 58

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
	Mul2	No						

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

Entry	Busy	Instruction	Stage	Dest	Value
1	Yes	LD R6, 34+R11	Commit	R6	M_1
2	Yes	LD R2, 45+R13	Commit	R2	M_2
3	Yes	MUL R0, R2, R4	Commit	R0	Val_3
4	Yes	SUB R8, R6, R2	Commit	R8	Val_1
5	Yes	DIV R10, R0, R6	Commit	R10	Val_4
6	Yes	ADD R6, R8, R2	WB	R6	Val_2
7					

Head moves down.

Head → 6

Tail → 7

Instruction 5 commits.

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #				#6			...
Busy	No	No		Yes	No	No	...
Value	Val_3	M_2		M_1	Val_1	Val_4	

R10 now has value Val_4 from ROB #5. R10 status sets to Not Busy.

Reorder Buffer Example: Cycle 59

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
	Mul2	No						

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

Entry	Busy	Instruction	Stage	Dest	Value
1	Yes	LD R6, 34+R11	Commit	R6	M_1
2	Yes	LD R2, 45+R13	Commit	R2	M_2
3	Yes	MUL R0, R2, R4	Commit	R0	Val_3
4	Yes	SUB R8, R6, R2	Commit	R8	Val_1
5	Yes	DIV R10, R0, R6	Commit	R10	Val_4
6	Yes	ADD R6, R8, R2	Commit	R6	Val_2
7					

Head → entry 6; Tail → entry 7

Register File:

	R0	R2	R4	R6	R8	R10	...
Reorder #							...
Busy	No	No		No	No	No	...
Value	Val_3	M_2		Val_2	Val_1	Val_4	

Reorder Buffer Example: Cycle 59

Reservation Stations for the Adder and Multiplier:

Time	Name	Busy	Op	V_j	V_k	Q_j	Q_k	*Dest*
	Add1	No						
	Add2	No						
	Add3	No						
	Mul1	No						
	Mul2	No						

Load Buffers:

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reorder Buffer:

Entry	Busy	Instruction	Stage	Dest	Value
1	Yes	LD R6, 34+R11	Commit	R6	M_1
2	Yes	LD R2, 45+R13	Commit	R2	M_2
3	Yes	MUL R0, R2, R4	Commit	R0	Val_3
4	Yes	SUB R8, R6, R2	Commit	R8	Val_1
5	Yes	DIV R10, R0, R6	Commit	R10	Val_4
6	Yes	ADD R6, R8, R2	Commit	R6	Val_2
7					

Head moves down.

Head

Tail

Instruction 6 commits.

Register File:

	R0	R2	R4	R6	R8		...
Reorder #							...
Busy	No	No		No	No	No	...
Value	Val_3	M_2		Val_2			

R6 now has value Val_2 from ROB #6. R6 status sets to Not Busy.

Reorder Buffer Example: Summary

Instruction Status:

Instructions			
Op	Dest *i*	Src1 *j*	Src2 *k*
LD	R6	34+	R11
LD	R2	45+	R13
MUL	R0	R2	R4
SUB	R8	R6	R2
DIV	R10	R0	R6
ADD	R6	R8	R2

Stages			
IS	EX	WB	Commit
1	2-3	4	5
2	3-4	5	6
3	6-15	16	17
4	6-7	8	18
5	17-56	57	58
6	9-10	11	59

In-of-order completion (commit).

In-order issue.

Out-of-order execution/write-back.

Road Map

Reorder Buffer

An Example with Reorder Buffer

Summary

Acknowledgment

Steps for Tomasulo+ROB Algortihm

- For each cycle,
  1. Check if next instruction can be issued.
     - Issue next instruction if possible.
     - Allocate ROB entry, allocate RS (or load buffer or store buffer) entry, and update register file states.
     - Move tail pointer to the new ROB entry.
  2. Check if previously issued instructions can execute (ready and have non-busy functional units).
     - Start/continue executing all instructions that are ready. Update timer.
  3. Check if any instruction can write-back.
     - Pick the first write-back-able instruction to write-back.
     - Update corresponding values in RS and ROB.
  4. Check if any instructions can commit.
     - Commit the first commit-able instruction.
     - Update the values in the register file or memory.
     - Move head pointer down by one entry.

Precise State with ROB

- ROB maintains precise state and allows speculation
  - Waits until precise condition reaches retire/commit stage
  - (Or until branch is noted mis-predicted)
  - Clear ROB, RS, and register status table (Flush)
  - Service exception/Restart from True Branch target
- Need to do similar things with memory ops
  - Called Memory Ordering Buffer (MOB)
    - Completed stores write to MOB then complete (write to memory) in-order (when they read head of buffer)

Precise Exception Example

- If at cycle 18, there is a divide-by-zero exception due to instruction 5
  - Simply flush the execution of instruction 6 from reorder buffer
  - When programmer inspects R6, value M_1 will be seen, allowing investigating the cause.

Tomasulo + ROB Summary

- Many implementations are very similar
  - Pentium III, PowerPC, etc
- Some limitations
  - Too many value copy operations
    - Register file => RS => ROB => Register File
  - Too many muxes/busses (CDB)
    - Values are coming from everywhere to everywhere else!
  - Reservation Stations mix values (data) and tags (control)
    - Slows down the max clock frequency

Road Map

Reorder Buffer

An Example with Reorder Buffer

Summary

Acknowledgment

Acknowledgment

This lecture is partially based on the slides from Dr. David Brooks. The first Tomasulo's example was based ib Dr. Broderson's example, for CS152 at UCB (Copyright (C) 2001 UCB).