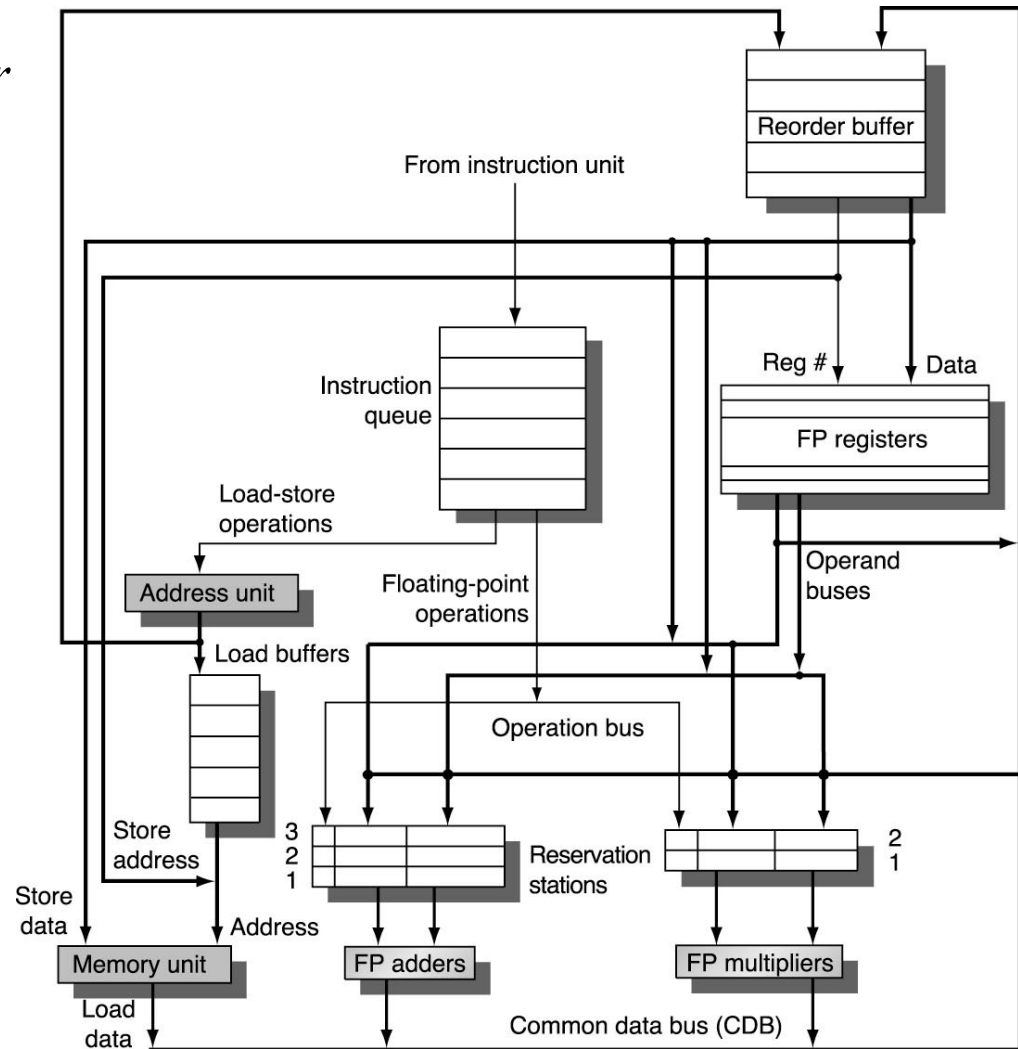


The Reorder Buffer

Hardware Speculative Execution

- Need HW buffer for results of uncommitted instructions: *reorder buffer*
 - Reorder buffer can be operand source
 - Once operand commits, result is found in register
 - 3 fields: instr. type, destination, value
 - Use reorder buffer number instead of reservation station
 - Instructions commit in order
 - As a result, its easy to undo speculated instructions on mispredicted branches or on exceptions



Hardware Speculative Execution

- Need HW buffer for results of uncommitted instructions:
reorder buffer
 - 3 fields:
 - instruction type
 - destination
 - value
 - once operand commits, result is found in register file
 - reservation station points to a ROB entry for pending source operand
 - flush ROB on a mispredicted branch
 - instructions are committed in-order from the ROB
 - handles precise exceptions

~~Four~~/Five Steps of Speculative ROB/Tomasulo Algorithm

1. Dispatch—get instruction from FP Op Queue

If reservation station and reorder buffer slot free, issue instr & send operands & reorder buffer no. for destination.

2. Issue—wait on operands

When both operands ready then execute; if not ready, watch CDB for result; when both in reservation station, proceed to execute

3. Execute —

4. Write result—finish execution (WB)

Write on Common Data Bus to all awaiting FUs & reorder buffer; mark reservation station available.

5. Commit—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.

HW Speculative Execution

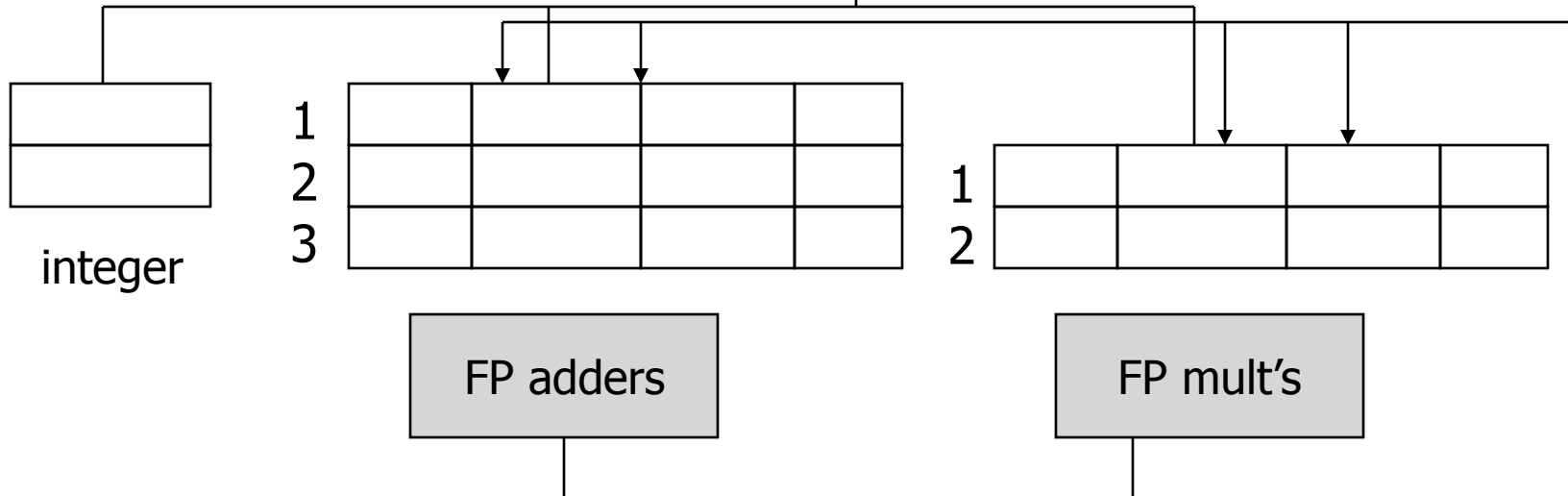
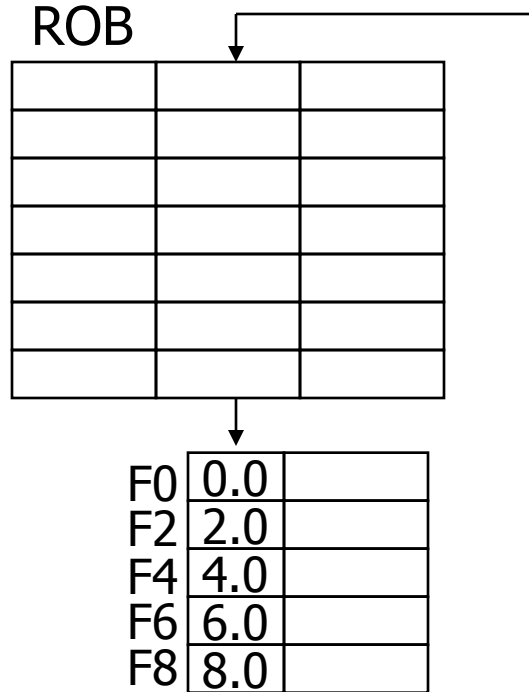
- The re-order buffer and in-order commit allow us to flush the speculative instructions from the machine when a misprediction is discovered.
- ROB is another possible source of operands
- ROB can provide precise exception in an out-of-order machine
- ROB allows us to ignore exceptions on speculative code.

ROB/Tomasulo – cycle 0

Loop:ADDD F4, F2, F0
 MULD F8, F4, F2
 ADDD F6, F8, F6
 SUBD F8, F2, F0
 SUBI ...
 BNEZ ..., Loop

Instruction Queue

SUBI ...
SUBD F8, F2, F0
ADDD F6, F8, F6
MULD F8, F4, F2
ADDD F4, F2, F0



ROB/Tomasulo – cycle 1

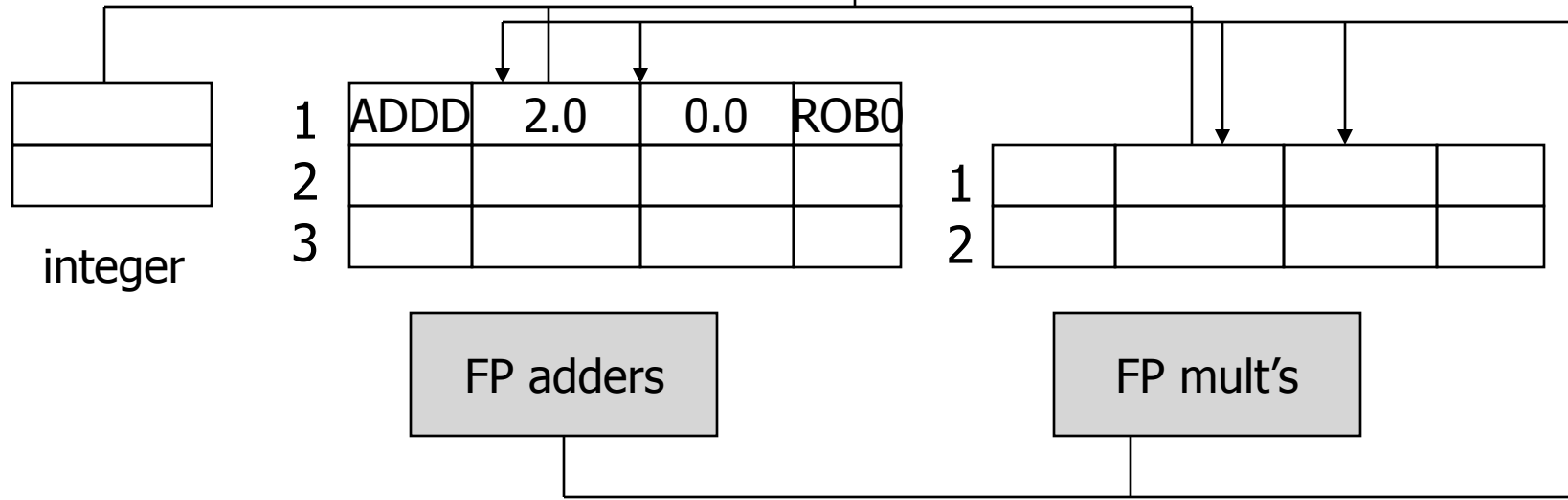
Loop: ADDD F4, F2, F0
 MULD F8, F4, F2
 ADDD F6, F8, F6
 SUBD F8, F2, F0
 SUBI ...
 BNEZ ...

Instruction Queue

BNEZ
SUBI
SUBD F8, F2, F0
ADDD F6, F8, F6
MULD F8, F4, F2

0	ADDD	F4	-	H
1				
2				
3				
4				
5				
6				

F0	0.0	
F2	2.0	
F4	4.0	ROB0
F6	6.0	
F8	8.0	



ROB/Tomasulo – cycle 2

Loop:

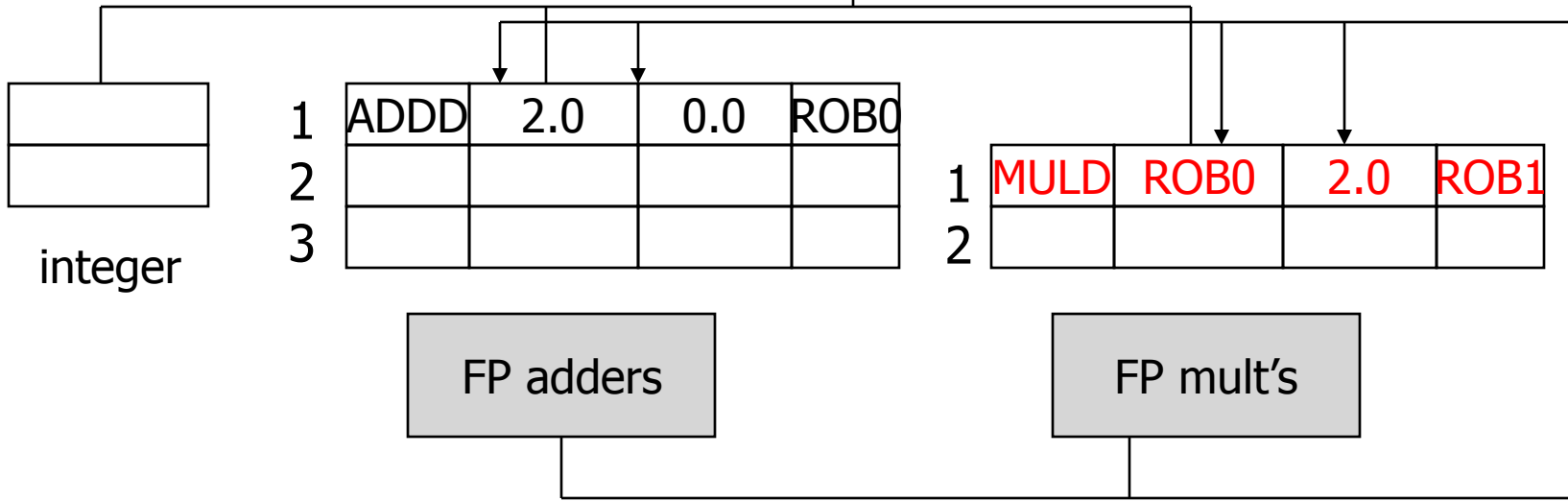
- ADDD F4, F2, F0
- MULD F8, F4, F2
- ADDD F6, F8, F6
- SUBD F8, F2, F0
- SUBI ...
- BNEZ ...

Instruction Queue

ADDD F4, F2, F0
BNEZ
SUBI
SUBD F8, F2, F0
ADDD F6, F8, F6

0	ADDD	F4	-	H
1	MULD	F8	-	
2				
3				
4				
5				
6				

F0	0.0	
F2	2.0	
F4	4.0	ROB0
F6	6.0	
F8	8.0	ROB1



ROB/Tomasulo – cycle 3

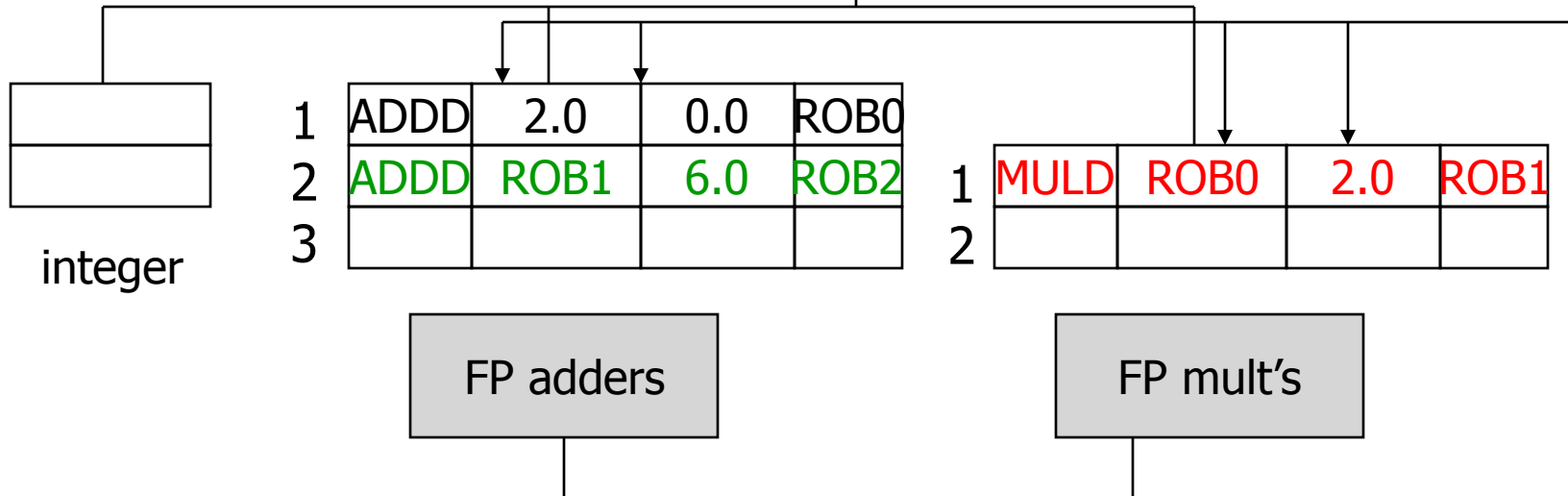
Loop: ADDD F4, F2, F0
 MULD F8, F4, F2
 ADDD F6, F8, F6
 SUBD F8, F2, F0
 SUBI ...
 BNEZ ...

Instruction Queue

MULD F8, F4, F2
ADDD F4, F2, F0
BNEZ
SUBI
SUBD F8, F2, F0

0	ADDD	F4	-	H
1	MULD	F8	-	
2	ADDD	F6	-	
3				
4				
5				
6				

F0	0.0	
F2	2.0	
F4	4.0	ROB0
F6	6.0	ROB2
F8	8.0	ROB1



ROB/Tomasulo – cycle 4

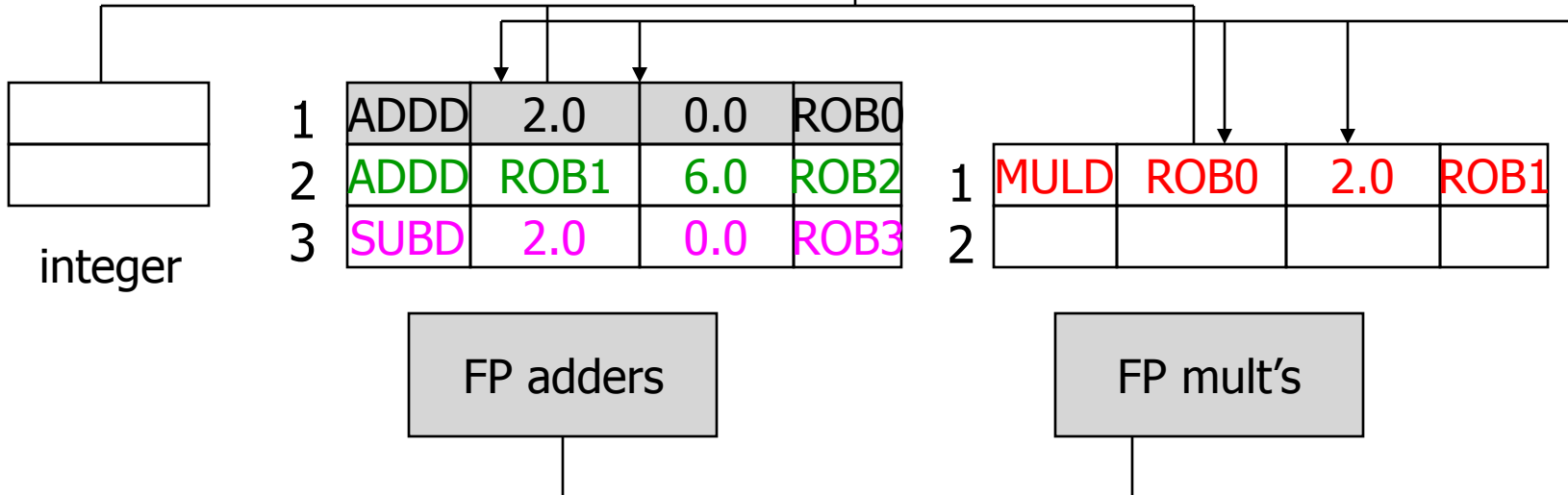
Loop: ADDD F4, F2, F0
 MULD F8, F4, F2
 ADDD F6, F8, F6
 SUBD F8, F2, F0
 SUBI ...
 BNEZ ...

Instruction Queue

ADDD F6, F8, F6
MULD F8, F4, F2
ADDD F4, F2, F0
BNEZ
SUBI

0	ADDD	F4	-	H
1	MULD	F8	-	
2	ADDD	F6	-	
3	SUBD	F8	-	
4				
5				
6				

F0	0.0	
F2	2.0	
F4	4.0	ROB0
F6	6.0	ROB2
F8	8.0	ROB3



ROB/Tomasulo – cycle 5

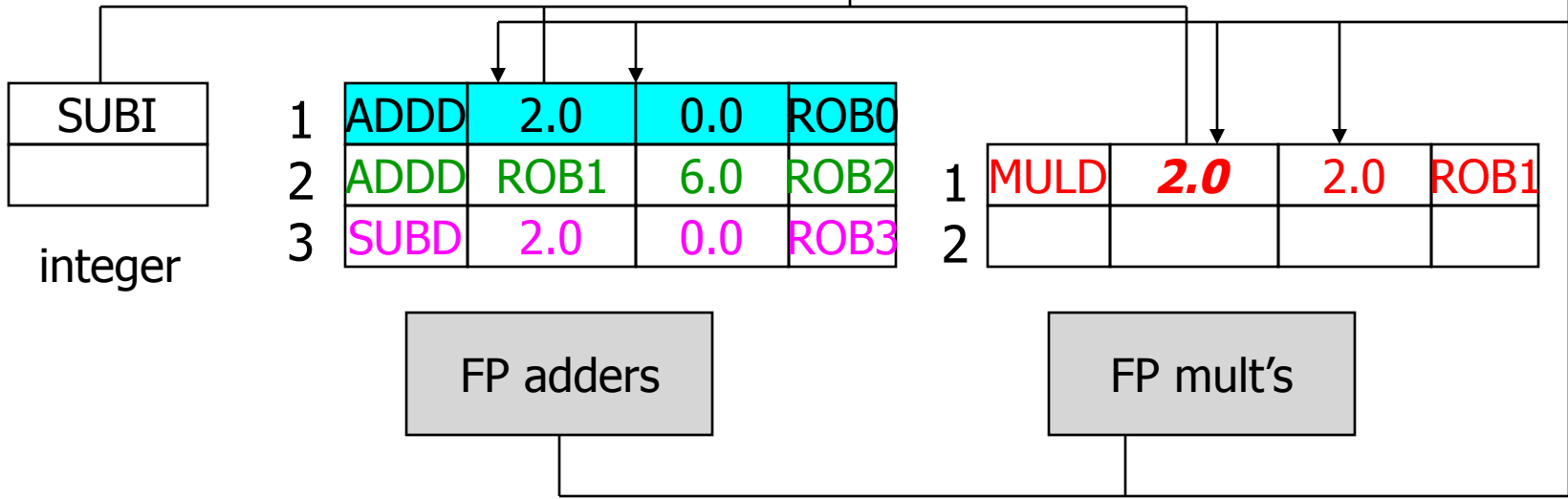
Loop: ADDD F4, F2, F0
 MULD F8, F4, F2
 ADDD F6, F8, F6
 SUBD F8, F2, F0
 SUBI ...
 BNEZ ...

Instruction Queue

SUBD F8, F2, F0
ADDD F6, F8, F6
MULD F8, F4, F2
ADDD F4, F2, F0
BNEZ

ROB		
0	ADDD	F4 2.0
1	MULD	F8 -
2	ADDD	F6 -
3	SUBD	F8 -
4	SUBI	
5		
6		

F0	0.0	
F2	2.0	
F4	4.0	ROB0
F6	6.0	ROB2
F8	8.0	ROB3



2.0 (ROB0)

ROB/Tomasulo – cycle 6

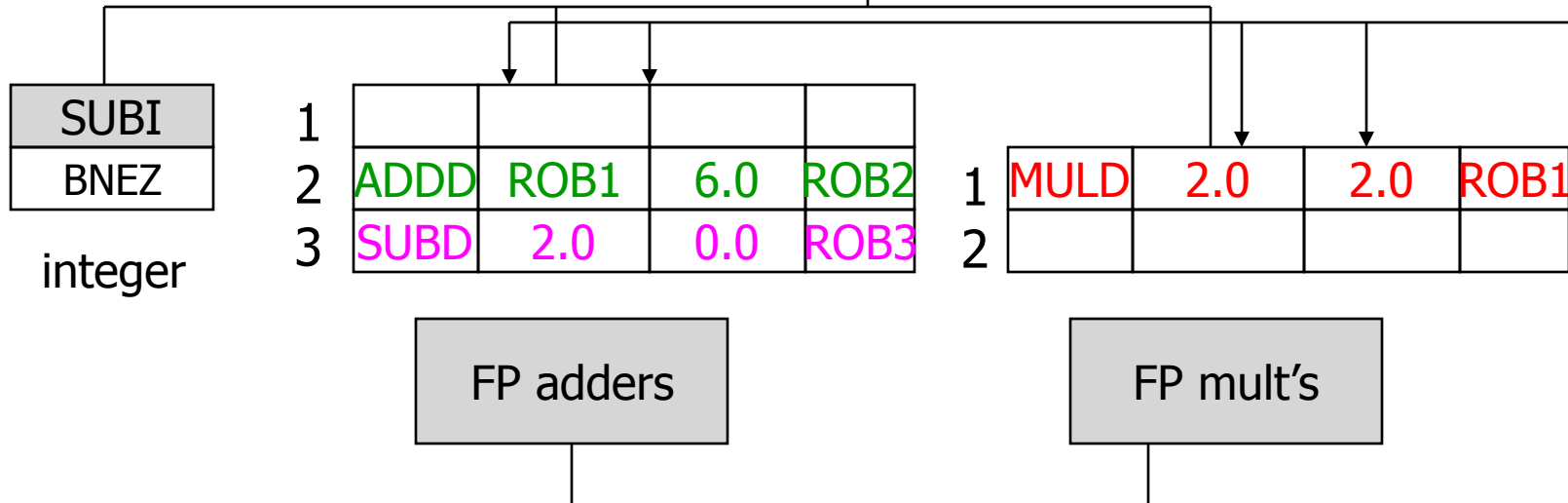
Loop: ADDD F4, F2, F0
 MULD F8, F4, F2
 ADDD F6, F8, F6
 SUBD F8, F2, F0
 SUBI ...
 BNEZ ...

Instruction Queue

SUBI
SUBD F8, F2, F0
ADDD F6, F8, F6
MULD F8, F4, F2
ADDD F4, F2, F0

0			
1	MULD	F8	-
2	ADDD	F6	-
3	SUBD	F8	-
4	SUBI		
5	BNEZ		
6			

F0	0.0	
F2	2.0	
F4	2.0	
F6	6.0	ROB2
F8	8.0	ROB3



ROB/Tomasulo – cycle 7

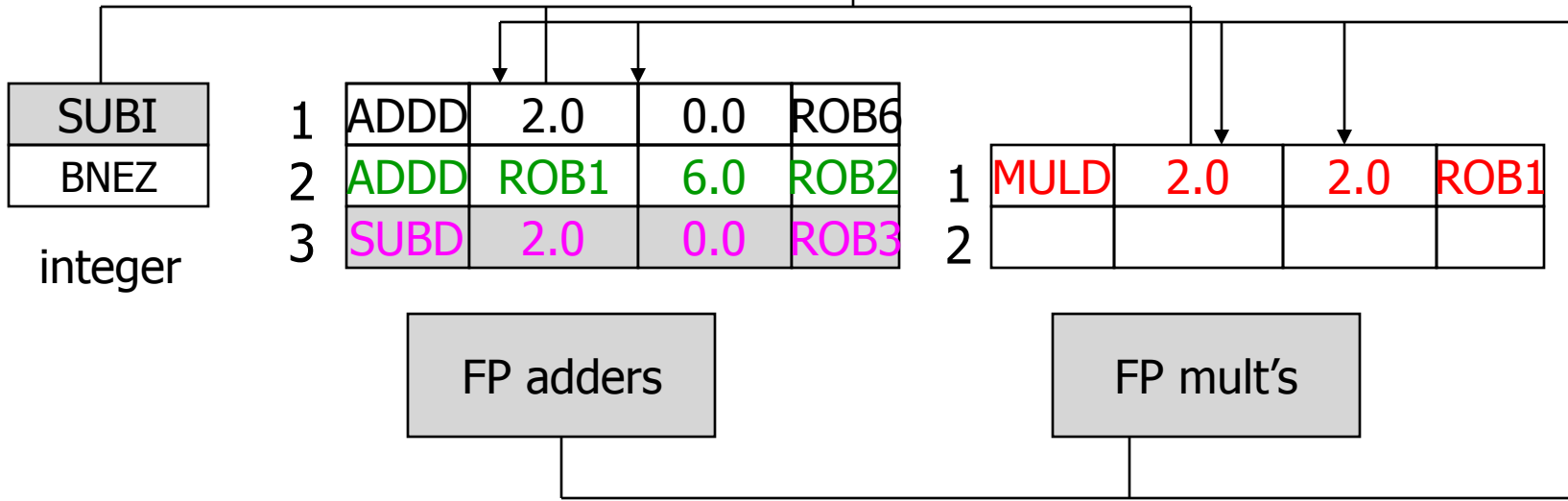
Loop: ADDD F4, F2, F0
 MULD F8, F4, F2
 ADDD F6, F8, F6
 SUBD F8, F2, F0
 SUBI ...
 BNEZ ...

Instruction Queue

BNEZ
SUBI
SUBD F8, F2, F0
ADDD F6, F8, F6
MULD F8, F4, F2

0			
1	MULD	F8	-
2	ADDD	F6	-
3	SUBD	F8	-
4	SUBI		
5	BNEZ		
6	ADDD	F4	

F0	0.0	
F2	2.0	
F4	2.0	ROB6
F6	6.0	ROB2
F8	8.0	ROB3



ROB/Tomasulo – cycle 8

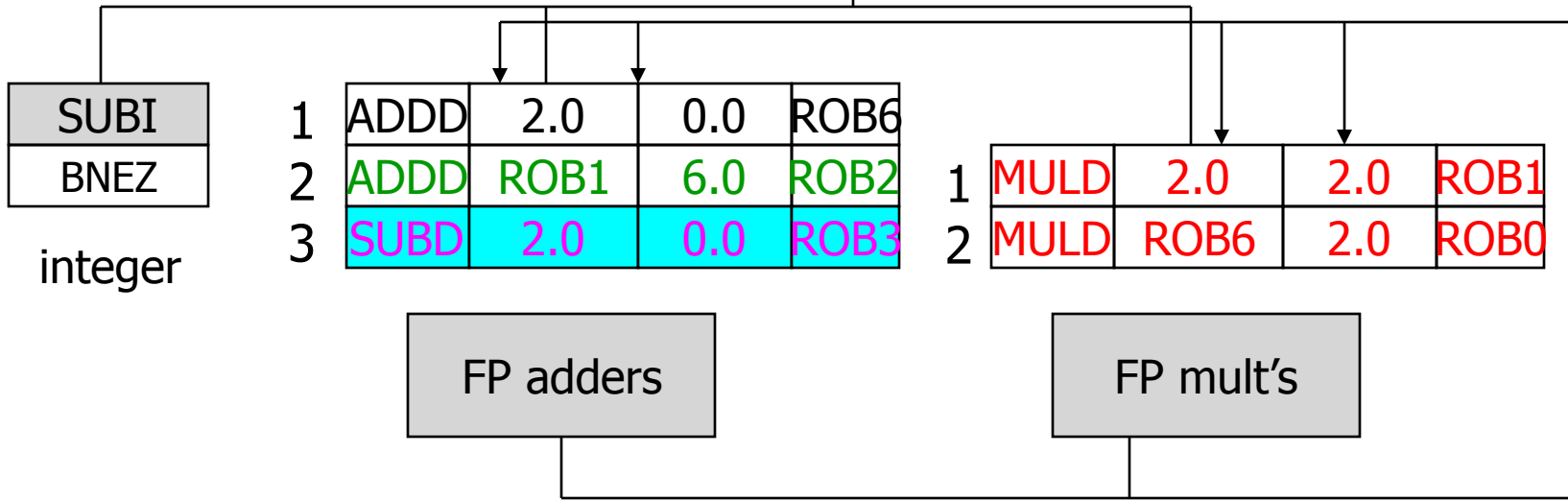
Loop: ADDD F4, F2, F0
 MULD F8, F4, F2
 ADDD F6, F8, F6
 SUBD F8, F2, F0
 SUBI ...
 BNEZ ...

Instruction Queue

ADDD F4, F2, F0
BNEZ
SUBI
SUBD F8, F2, F0
ADDD F6, F8, F6

0	MULD	F8	-
1	MULD	F8	-
2	ADDD	F6	-
3	SUBD	F8	2.0
4	SUBI		
5	BNEZ		
6	ADDD	F4	

F0	0.0	
F2	2.0	
F4	2.0	ROB6
F6	6.0	ROB2
F8	8.0	ROB0



2.0 (ROB3)

ROB/Tomasulo – cycle 9

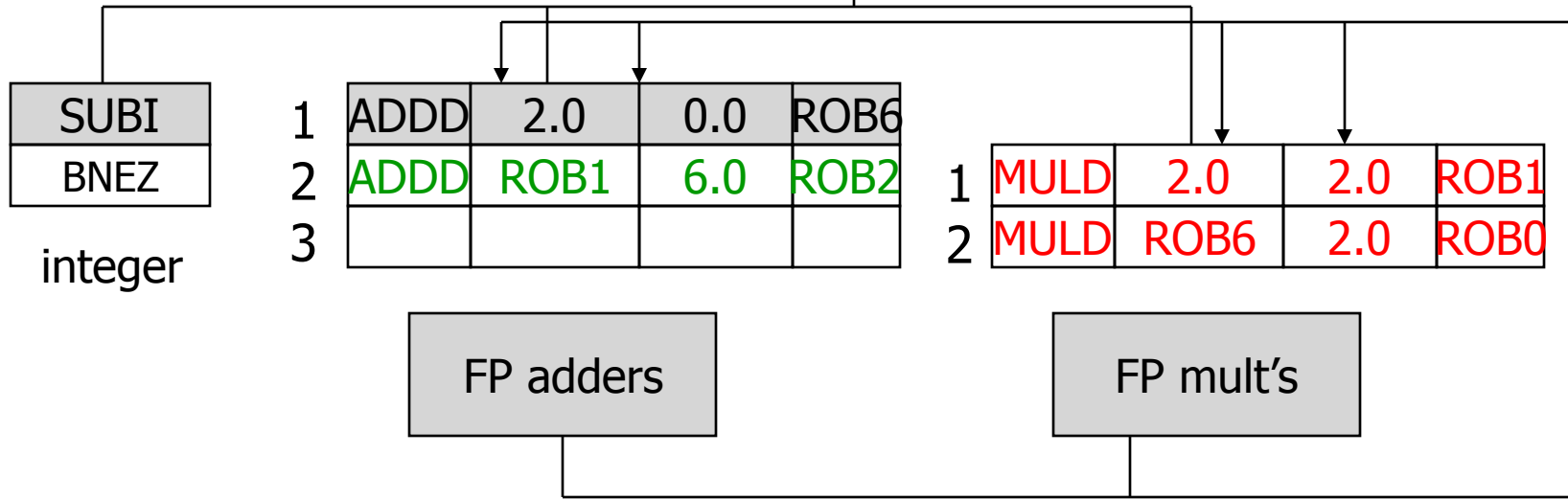
Loop: ADDD F4, F2, F0
 MULD F8, F4, F2
 ADDD F6, F8, F6
 SUBD F8, F2, F0
 SUBI ...
 BNEZ ...

Instruction Queue

ADDD F4, F2, F0
BNEZ
SUBI
SUBD F8, F2, F0
ADDD F6, F8, F6

ROB			
0	MULD	F8	-
1	MULD	F8	-
2	ADDD	F6	-
3	SUBD	F8	2.0
4	SUBI		
5	BNEZ		
6	ADDD	F4	

F0	0.0	
F2	2.0	
F4	2.0	ROB6
F6	6.0	ROB2
F8	8.0	ROB0



ROB/Tomasulo – cycle 11

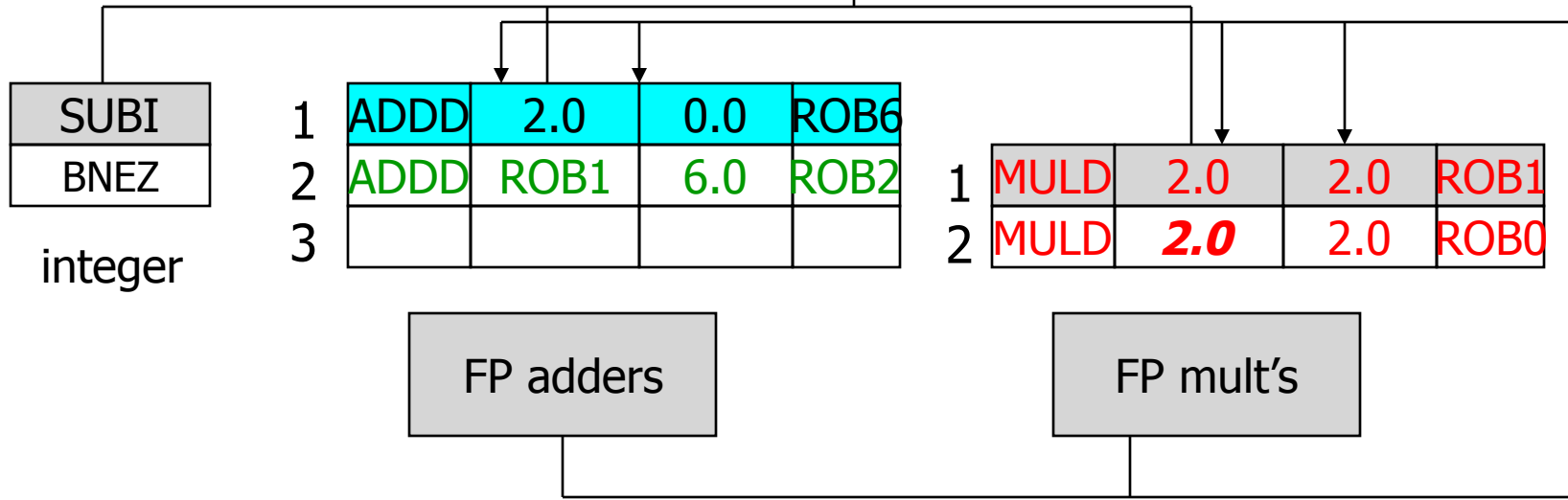
Loop: ADDD F4, F2, F0
 MULD F8, F4, F2
 ADDD F6, F8, F6
 SUBD F8, F2, F0
 SUBI ...
 BNEZ ...

Instruction Queue

ADDD F4, F2, F0
BNEZ
SUBI
SUBD F8, F2, F0
ADDD F6, F8, F6

ROB		
0	MULD	F8 -
1	MULD	F8 -
2	ADDD	F6 -
3	SUBD	F8 2.0
4	SUBI	
5	BNEZ	
6	ADDD	F4 2.0

F0	0.0	
F2	2.0	
F4	2.0	ROB6
F6	6.0	ROB2
F8	8.0	ROB0



2.0 (ROB6)

ROB/Tomasulo – cycle 15

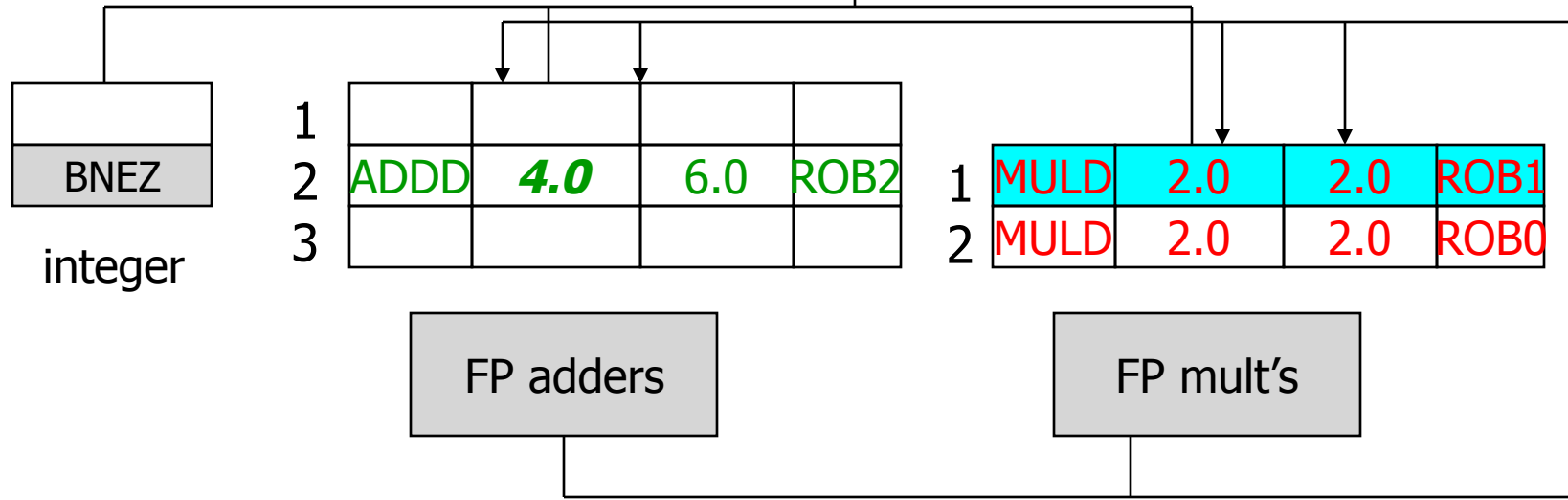
Loop: ADDD F4, F2, F0
 MULD F8, F4, F2
 ADDD F6, F8, F6
 SUBD F8, F2, F0
 SUBI ...
 BNEZ ...

Instruction Queue

ADDD F4, F2, F0
BNEZ
SUBI
SUBD F8, F2, F0
ADDD F6, F8, F6

0	MULD	F8	-
1	MULD	F8	4.0
2	ADDD	F6	-
3	SUBD	F8	2.0
4	SUBI		val
5	BNEZ		
6	ADDD	F4	2.0

F0	0.0	
F2	2.0	
F4	2.0	ROB6
F6	6.0	ROB2
F8	8.0	ROB0



4.0 (ROB1)

ROB/Tomasulo – cycle 16

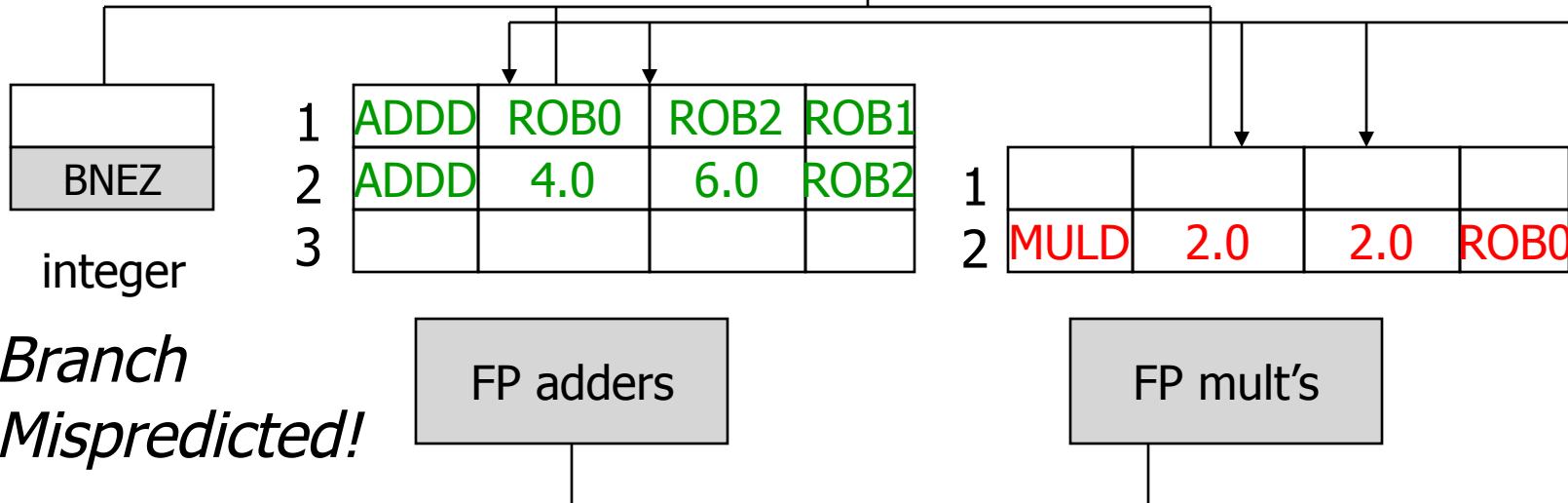
Loop: ADDD F4, F2, F0
 MULD F8, F4, F2
 ADDD F6, F8, F6
 SUBD F8, F2, F0
 SUBI ...
 BNEZ ...

Instruction Queue

MULD F8, F4, F2
ADDD F4, F2, F0
BNEZ
SUBI
SUBD F8, F2, F0

ROB		
0	MULD	F8 -
1	ADDD	F6 -
2	ADDD	F6 -
3	SUBD	F8 2.0
4	SUBI	val
5	BNEZ	
6	ADDD	F4 2.0

F0	0.0	
F2	2.0	
F4	2.0	ROB6
F6	6.0	ROB2
F8	4.0	ROB0



ROB/Tomasulo – cycle 17

Loop:

- ADDD F4, F2, F0
- MULD F8, F4, F2
- ADDD F6, F8, F6
- SUBD F8, F2, F0
- SUBI ...
- BNEZ ...

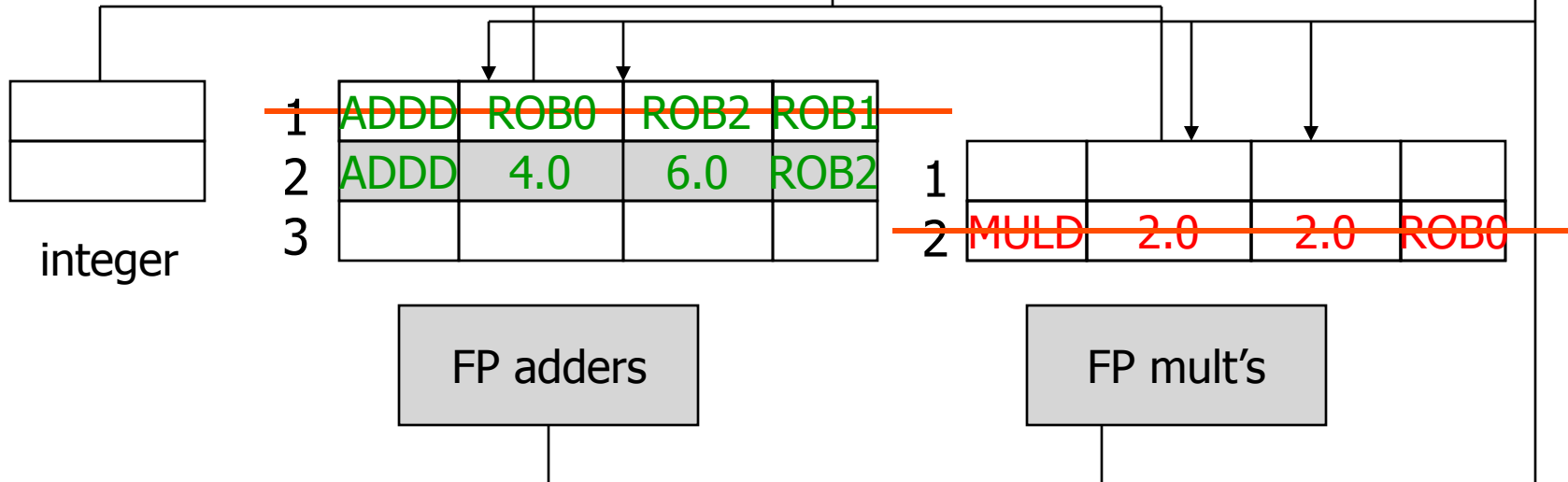
Instruction Queue

flushed
flushed
flushed
flushed
flushed

0	flushed	
1	flushed	
2	ADDD	F6 -
3	SUBD	F8 2.0
4	SUBI	val
5	BNEZ	nt
6	flushed	

F0	0.0	
F2	2.0	
F4	2.0	
F6	6.0	ROB2
F8	4.0	ROB3

RST entries from branch

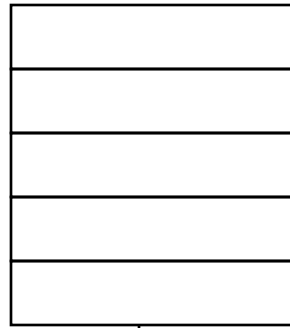


ROB/Tomasulo – cycle 19

Loop:

- ADDD F4, F2, F0
- MULD F8, F4, F2
- ADDD F6, F8, F6
- SUBD F8, F2, F0
- SUBI ...
- BNEZ ...

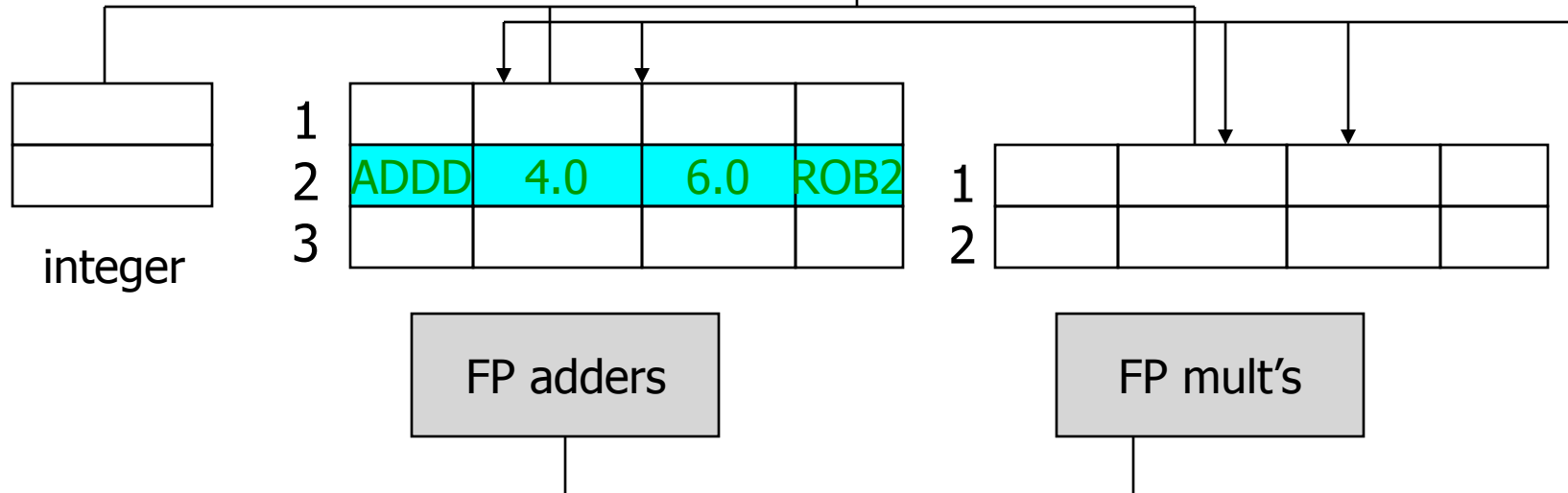
Instruction Queue



ROB		
0		
1		
2	ADDD	F6 10.0
3	SUBD	F8 2.0
4	SUBI	val
5	BNEZ	nt
6		

H

F0	0.0	
F2	2.0	
F4	2.0	
F6	6.0	ROB2
F8	4.0	ROB3

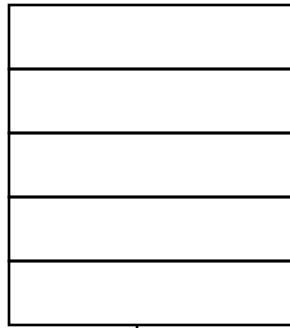


10.0 (ROB2)

ROB/Tomasulo – cycle 20

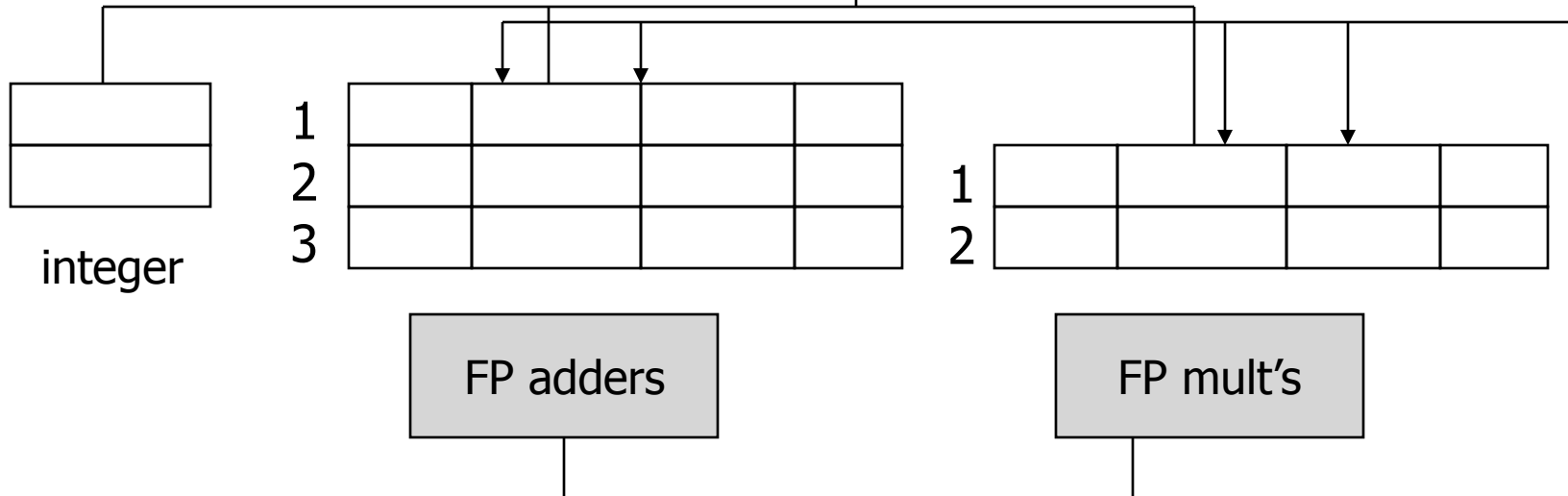
Loop:ADDD F4, F2, F0
 MULDD F8, F4, F2
 ADDD F6, F8, F6
 SUBD F8, F2, F0
 SUBI ...
 BNEZ ...

Instruction Queue



ROB		
0		
1		
2		
3	SUBD	F8 2.0
4	SUBI	val
5	BNEZ	nt
6		

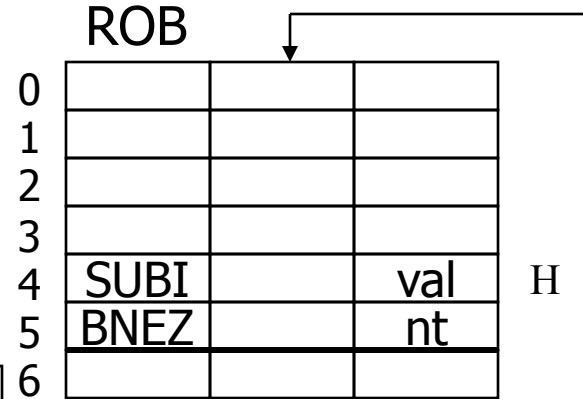
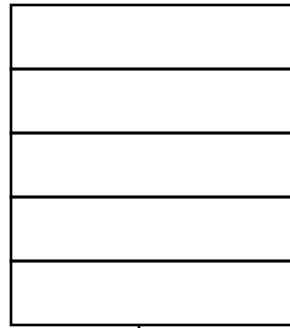
F0	0.0	
F2	2.0	
F4	2.0	
F6	10.0	
F8	4.0	ROB3



ROB/Tomasulo – cycle 21

Loop:ADDD F4, F2, F0
 MUL~~D~~ F8, F4, F2
 ADDD F6, F8, F6
 SUBD F8, F2, F0
 SUBI ...
 BNEZ ...

Instruction Queue



F0	0.0	
F2	2.0	
F4	2.0	
F6	10.0	
F8	2.0	

