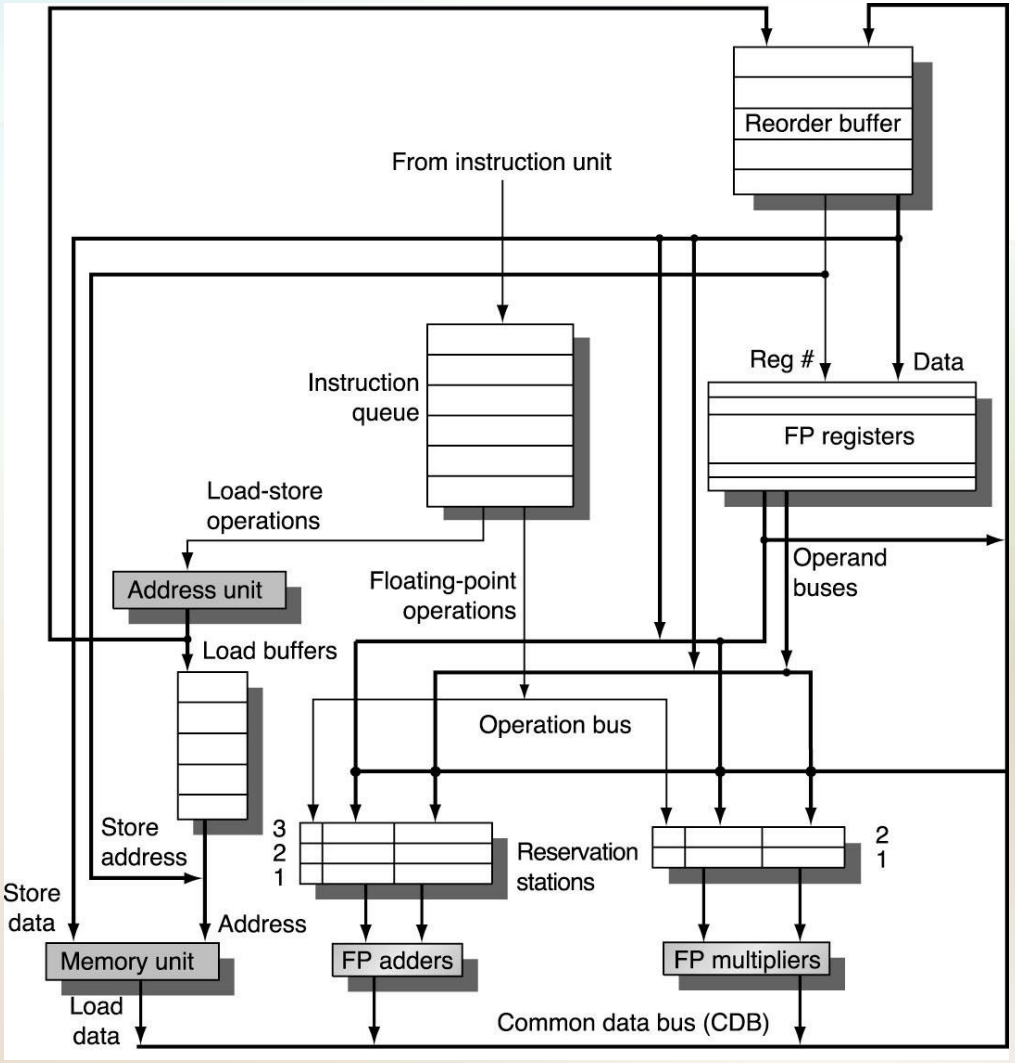
Instruction-level Parallelism

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 as "name" of result
  - 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**—get instruction from FP Op Queue

   If reservation station and reorder buffer slot free, issue instr & send operands & reorder buffer no. for destination. Operands may be read from register file or reorder buffer.

2. **Execution**—operate on operands (EX)

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

3. **Write result**—finish execution (WB)

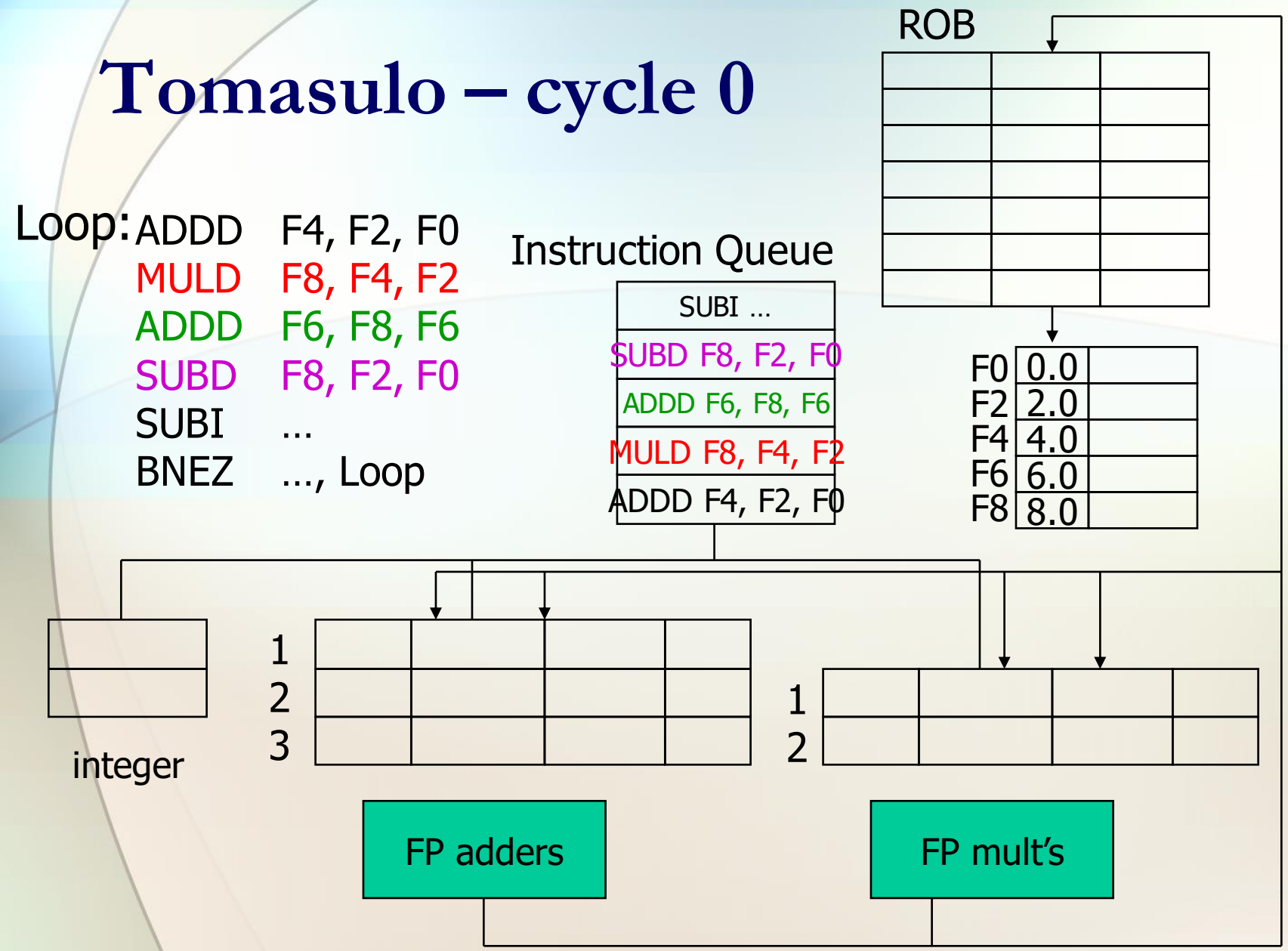
   Write on Common Data Bus to all awaiting FUs & reorder buffer; mark reservation station available.

4. **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.

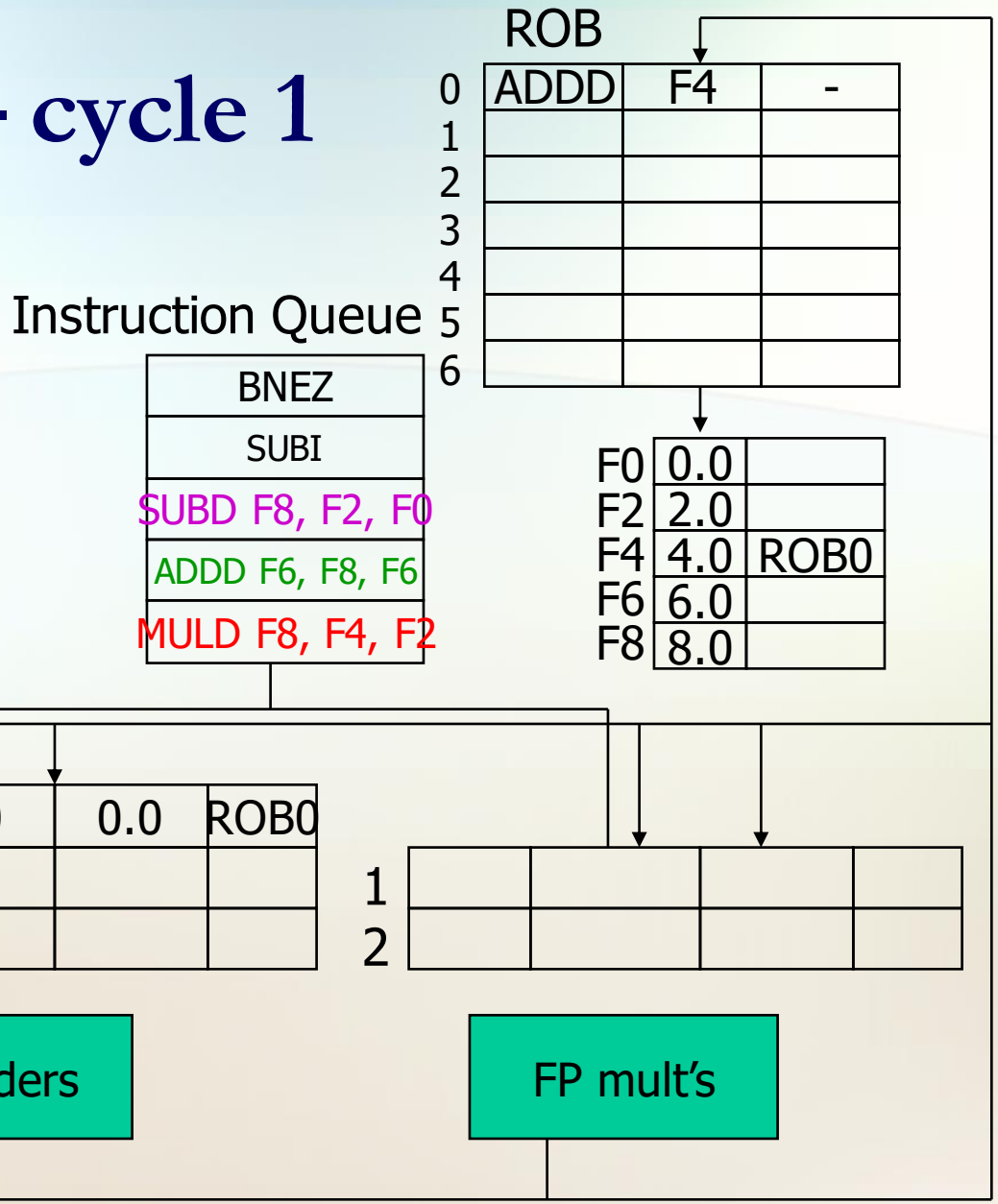
Tomasulo – cycle 0

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



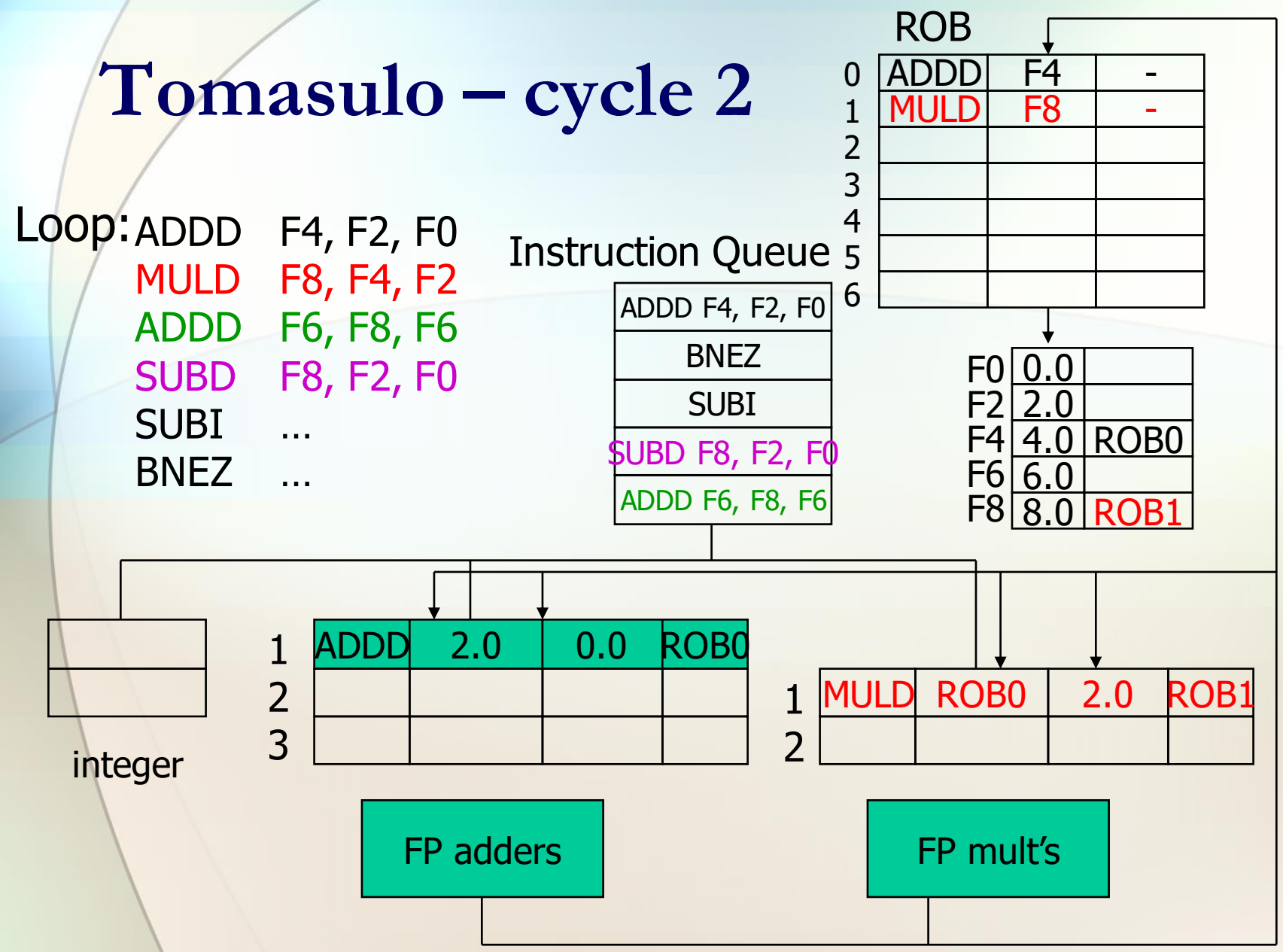
Tomasulo – cycle 1

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



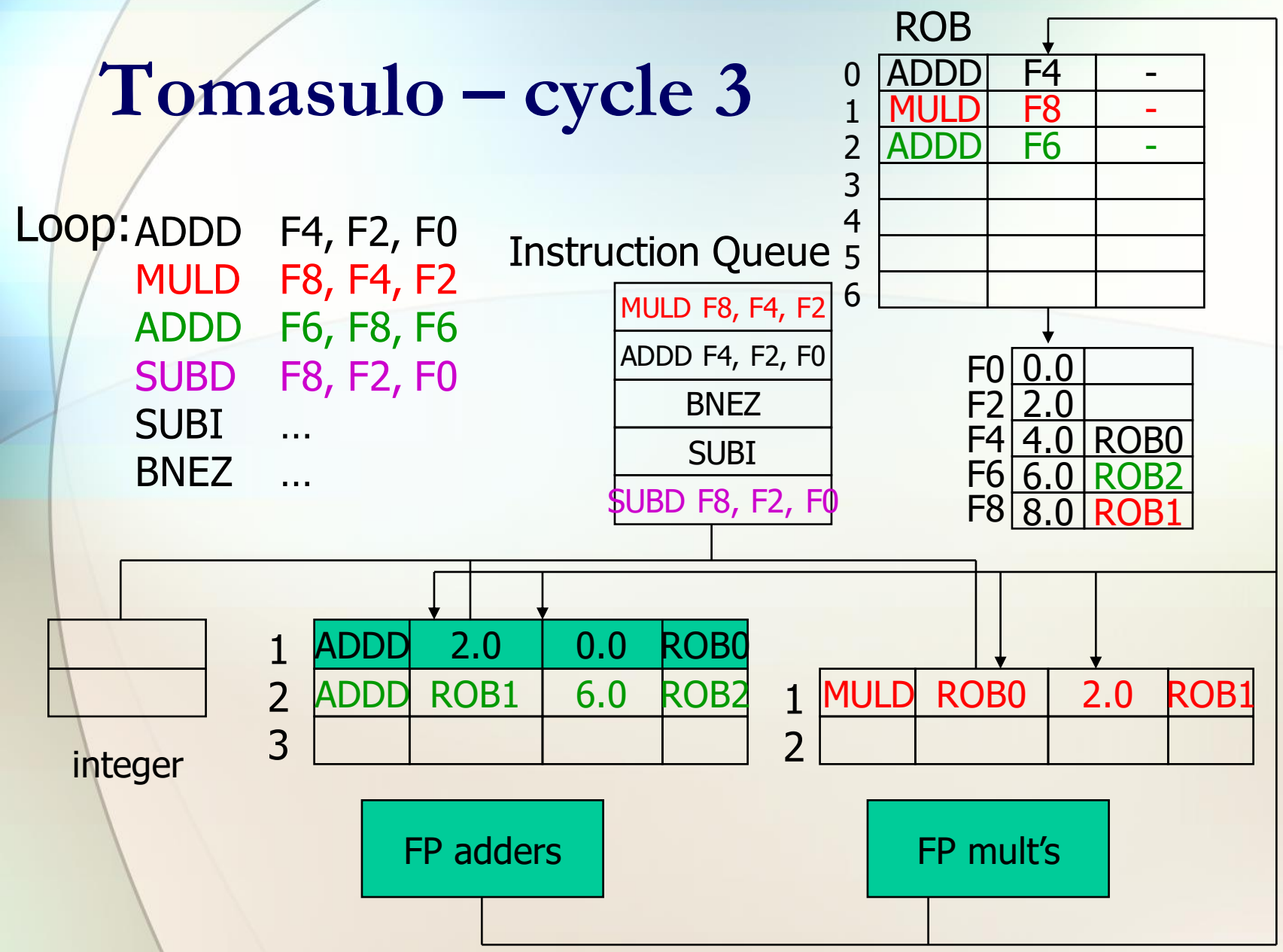
Tomasulo – cycle 2

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



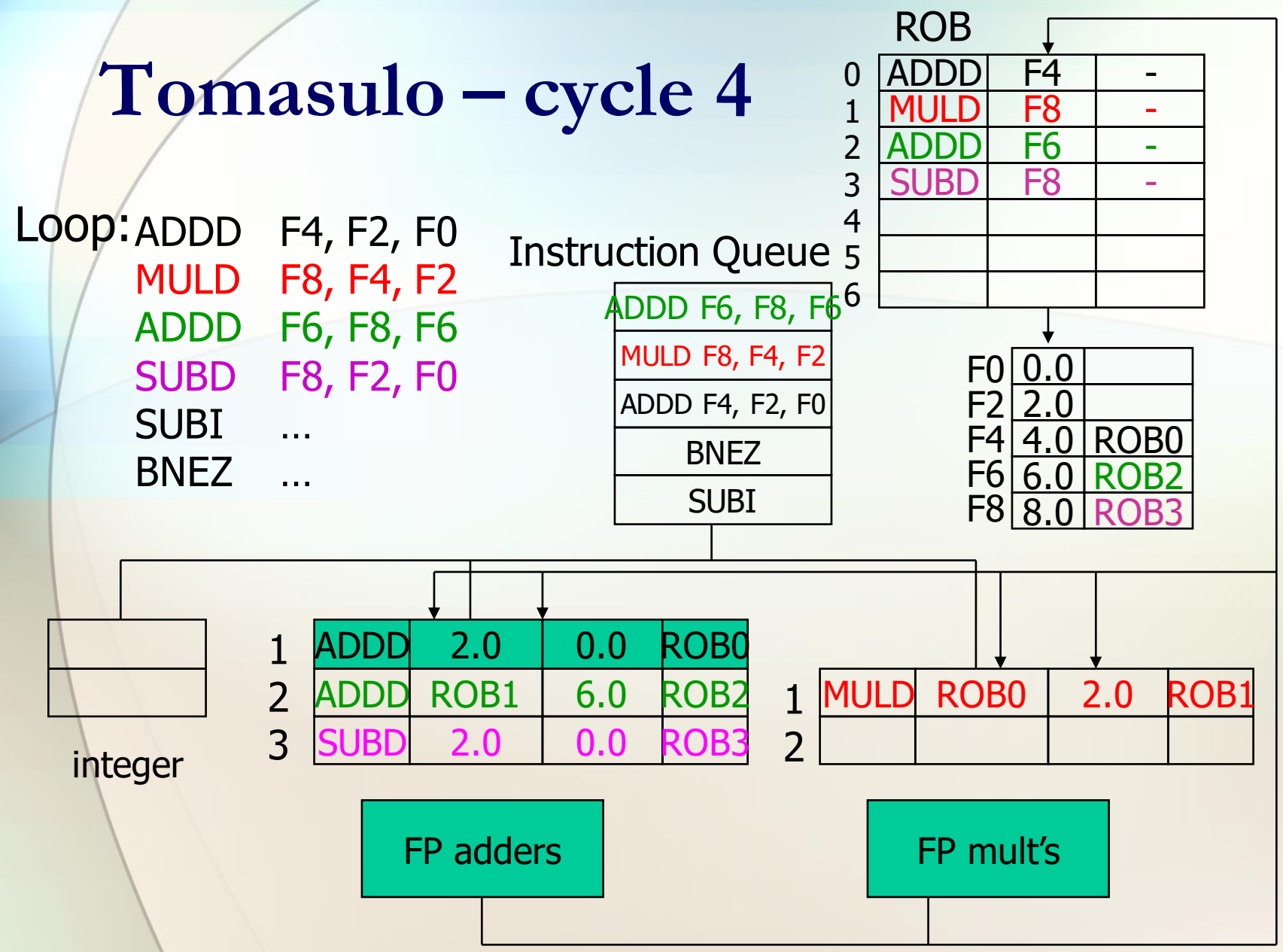
Tomasulo – cycle 3

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



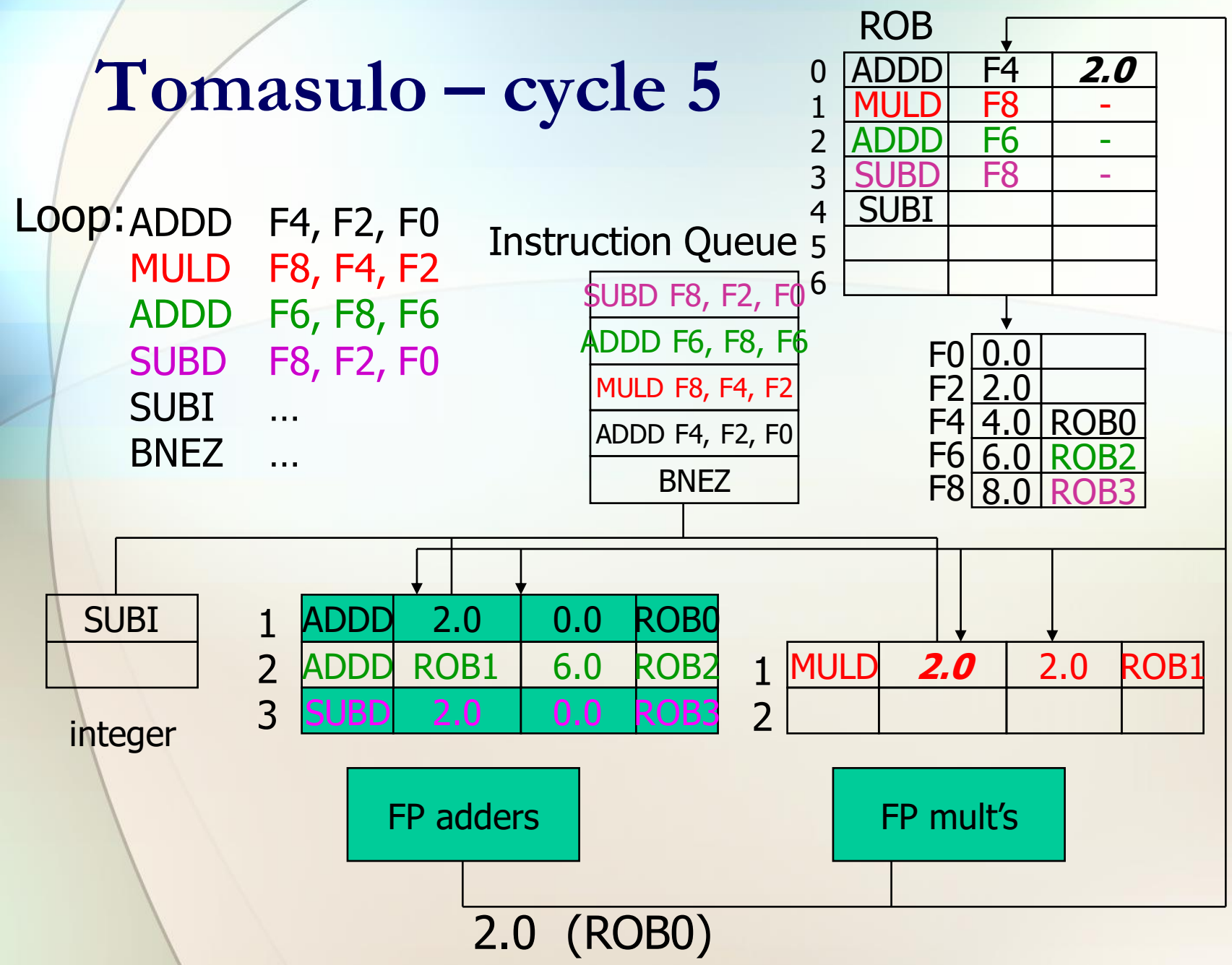
Tomasulo – cycle 4

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



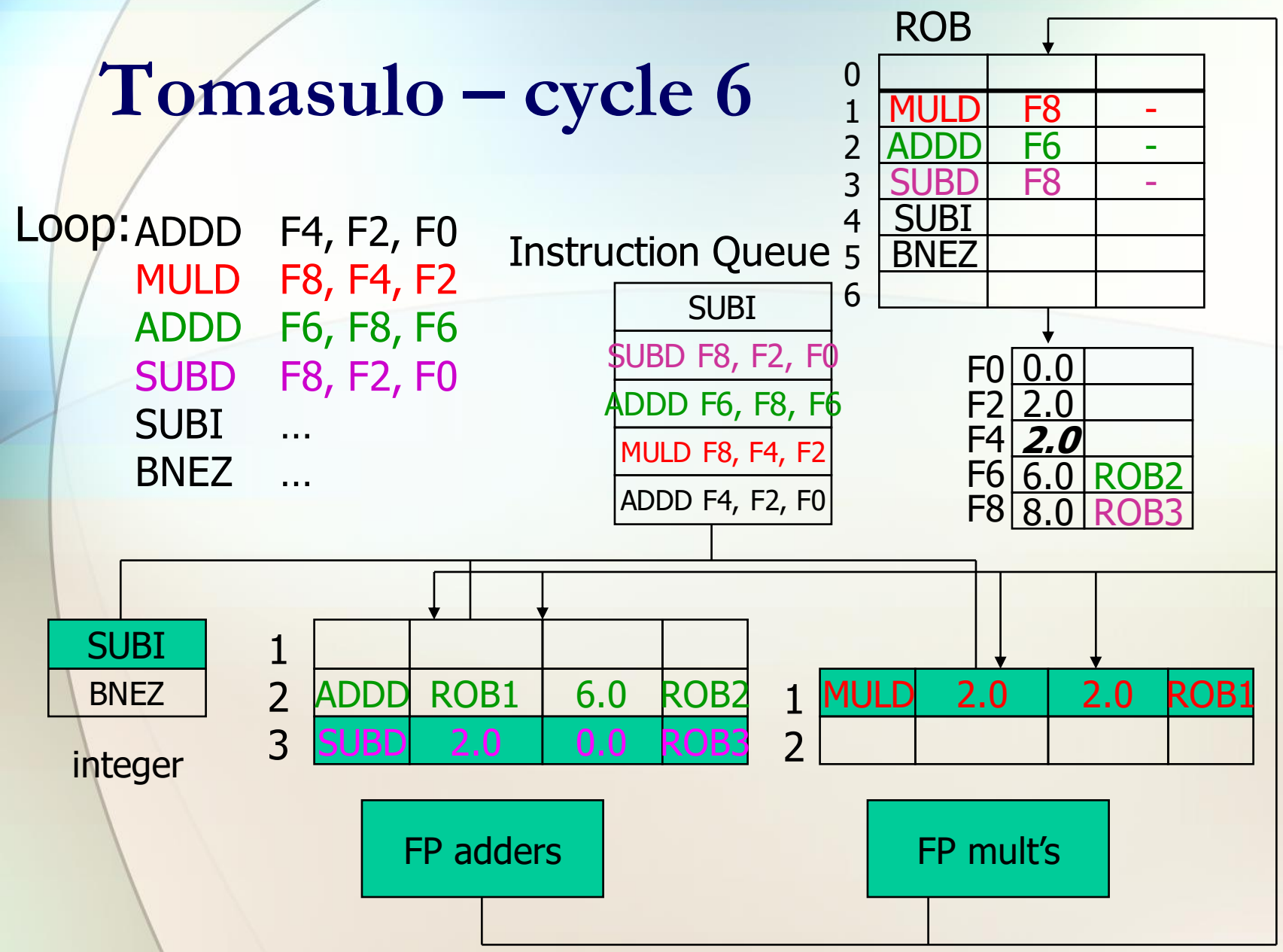
Tomasulo – cycle 5

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



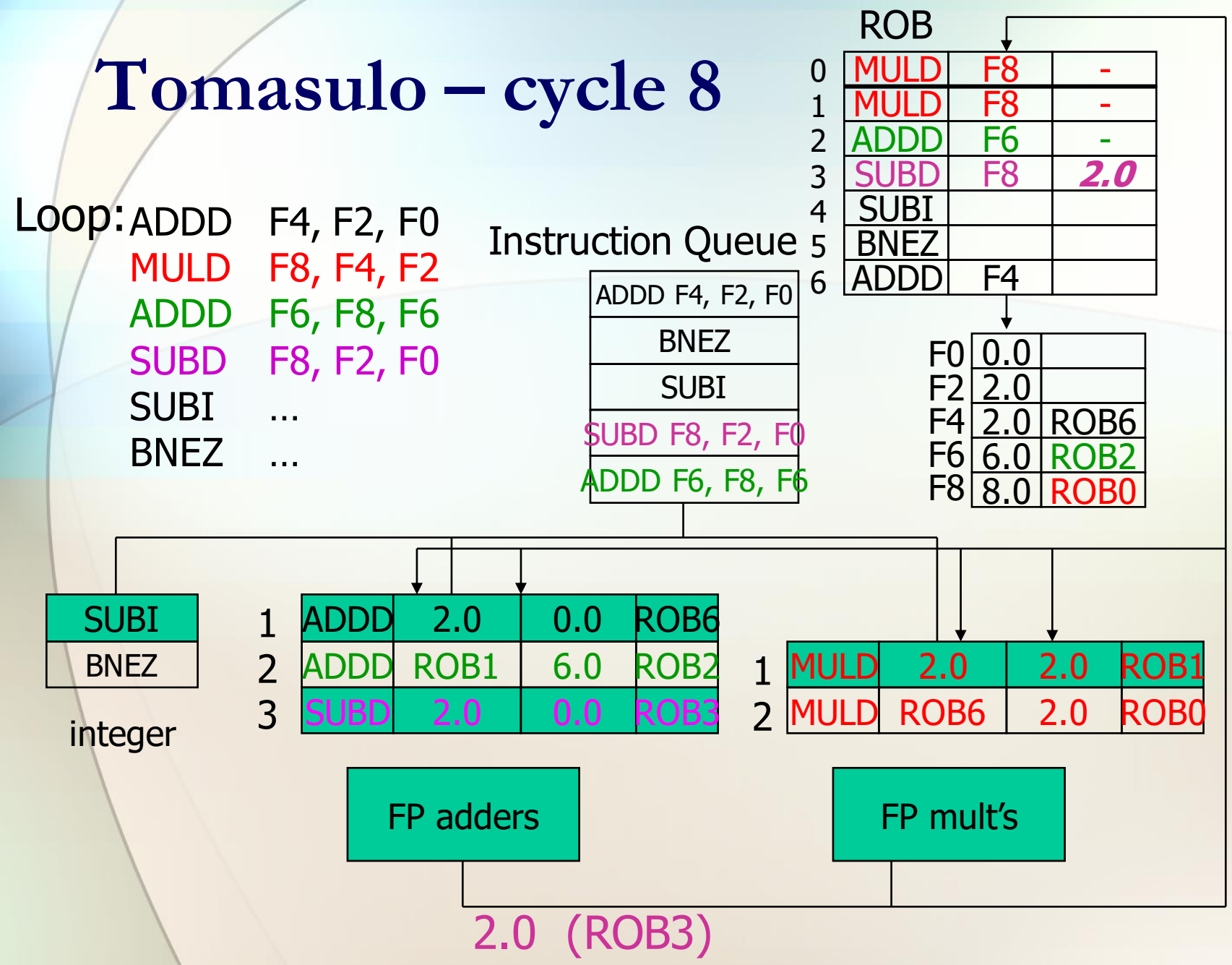
Tomasulo – cycle 6

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



Tomasulo – cycle 8

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



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

SUBI
BNEZ

integer

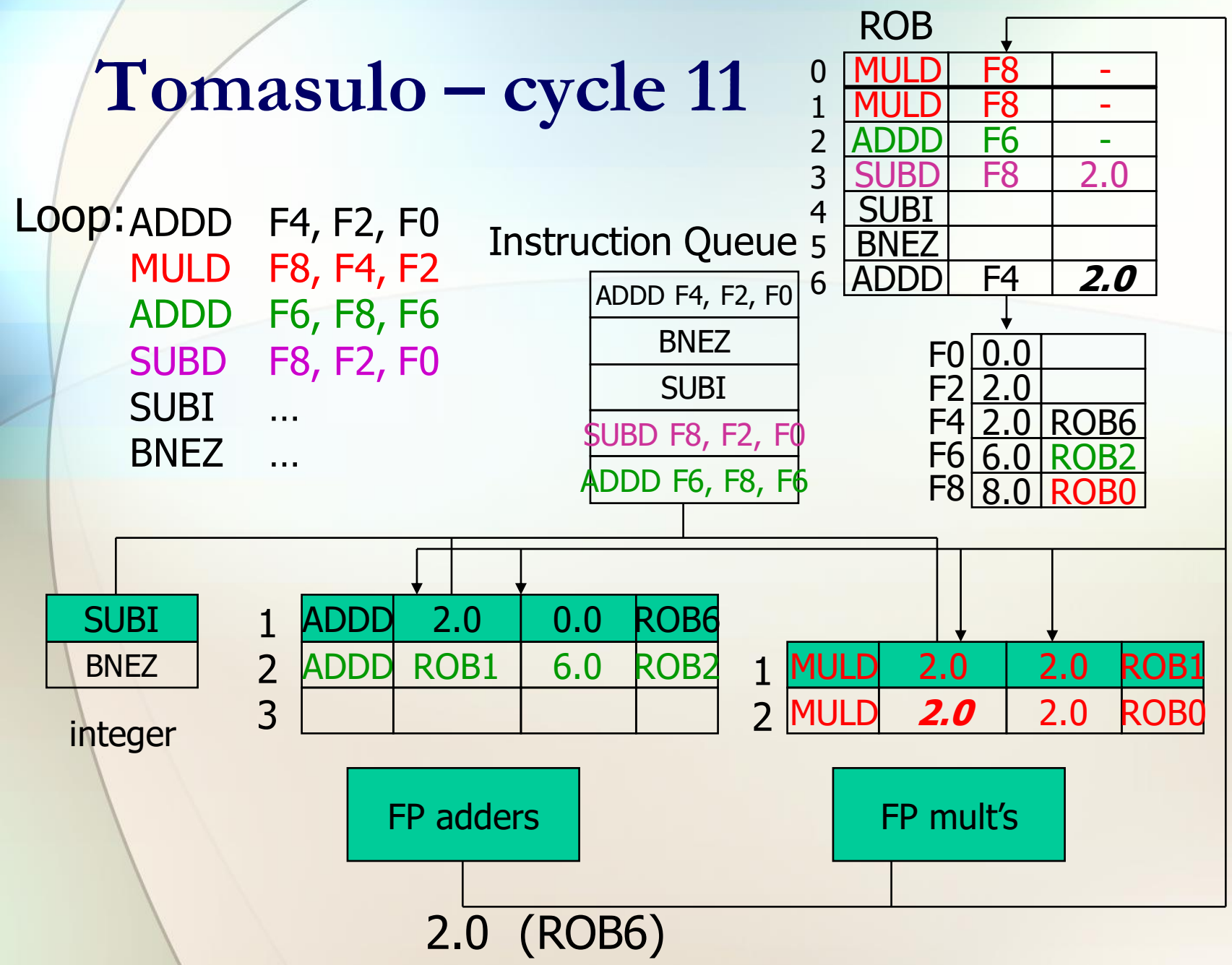
1	ADDD	2.0	0.0	ROB6
2	ADDD	ROB1	6.0	ROB2
3				

1	MULD	2.0	2.0	ROB1
2	MULD	ROB6	2.0	ROB0

FP adders

FP mult's

Tomasulo – cycle 11



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

ROB

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

BNEZ

integer

1				
2	ADDD	***4.0***	6.0	ROB2
3				

FP adders

1	MULD	2.0	2.0	ROB1
2	MULD	2.0	2.0	ROB0

FP mult's

4.0 (ROB1)

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

	BNEZ			
1	ADDD	ROB0	ROB2	ROB1
2	ADDD	4.0	6.0	ROB2
3				

integer

1				
2	MULD	2.0	2.0	ROB0

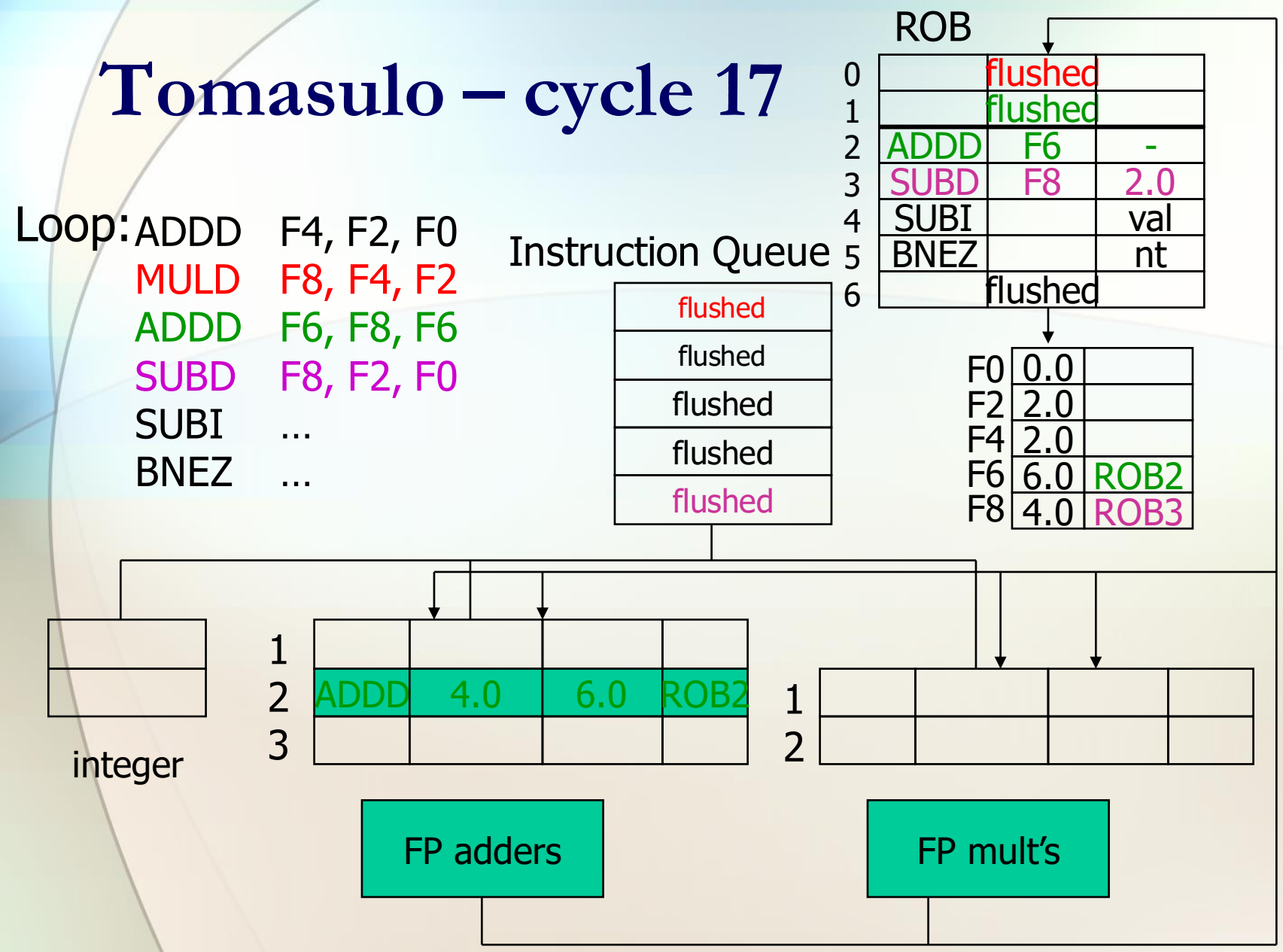
FP adders

FP mult's

Branch Mispredicted!

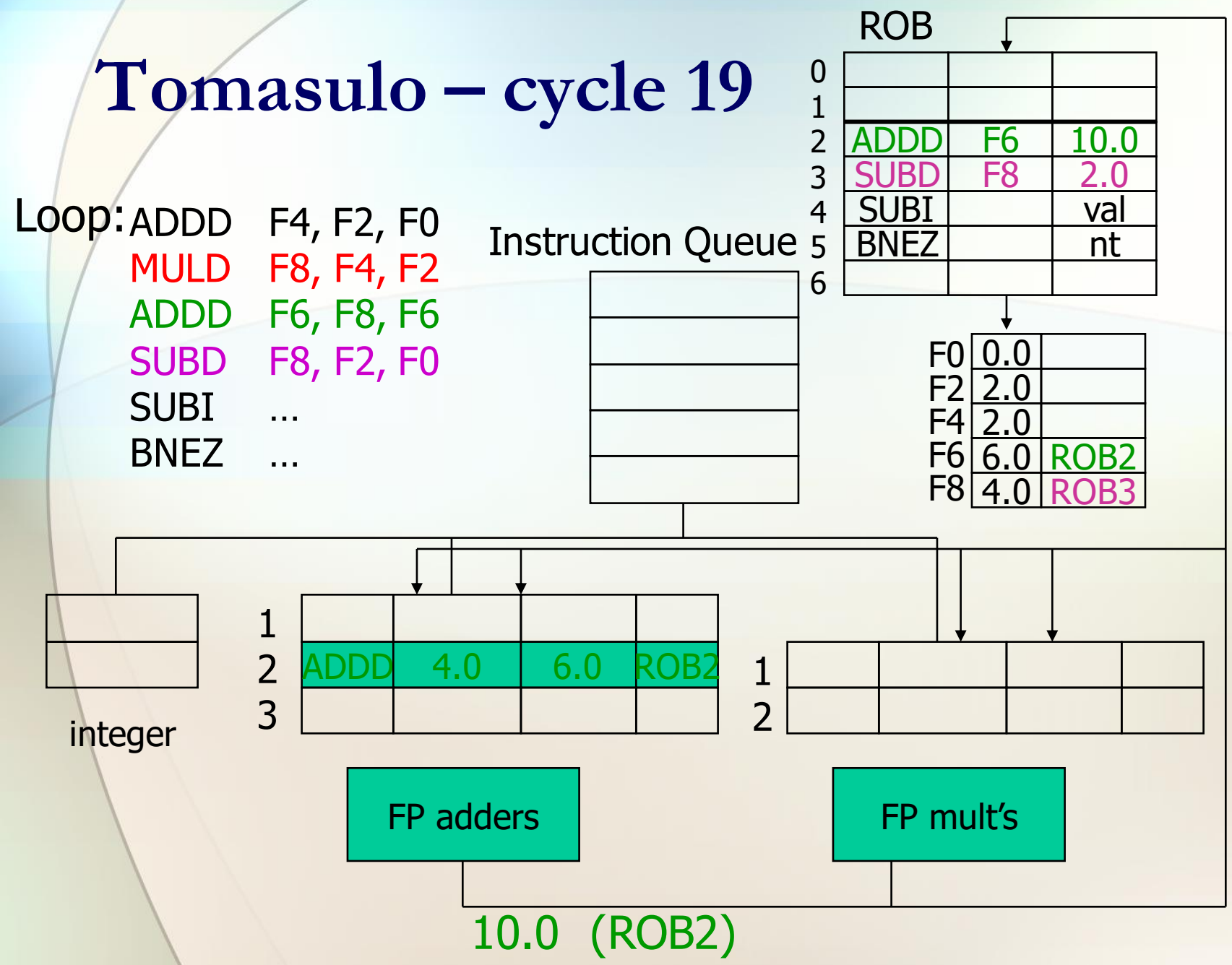
Tomasulo – cycle 17

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



Tomasulo – cycle 19

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



Tomasulo – cycle 20

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