Lecture 4

Speculative Instruction Execution

- **Dynamically scheduled pipelines (Ch. 3.4)**

→ ✓ Speculative execution (3.4.2)

→ ✓ Dynamic branch prediction (3.4.3)

Putting it Together:

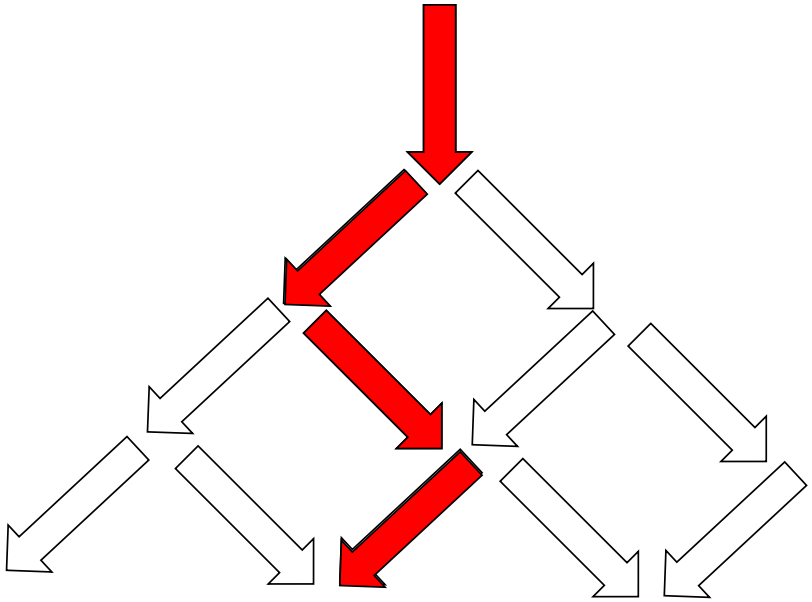
→ ✓ Tomasulo + Branch prediction + Speculation (3.4.4)

→ ✓ Dynamic memory disambiguation (3.4.5)

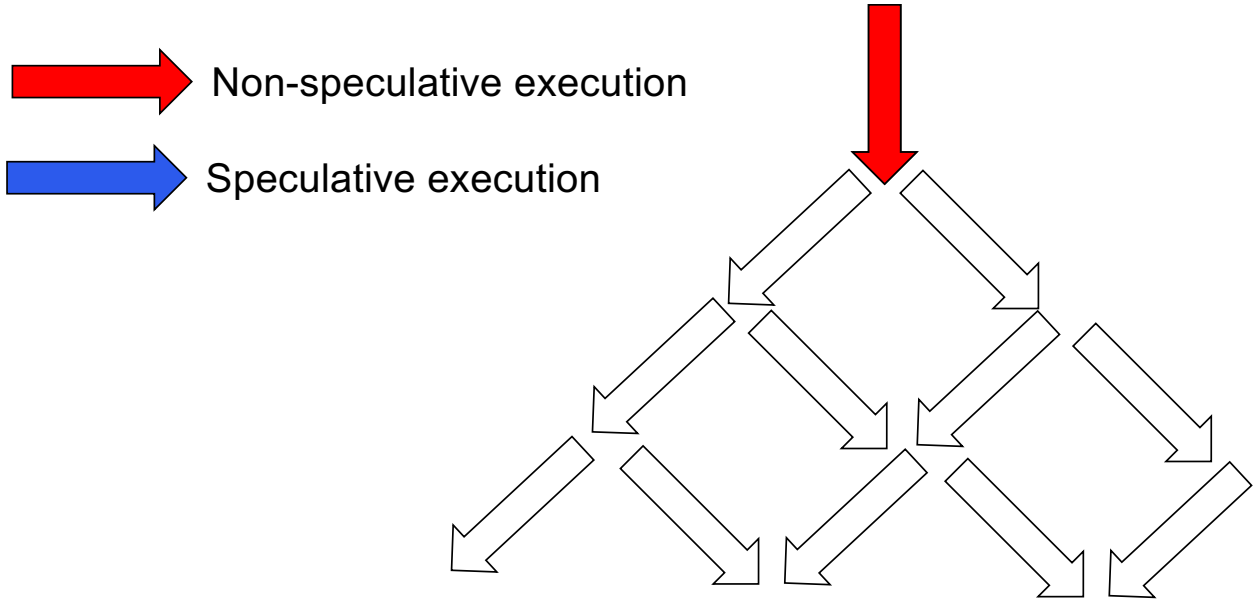
→ ✓ Register renaming techniques (3.4.6)

Speculative Execution

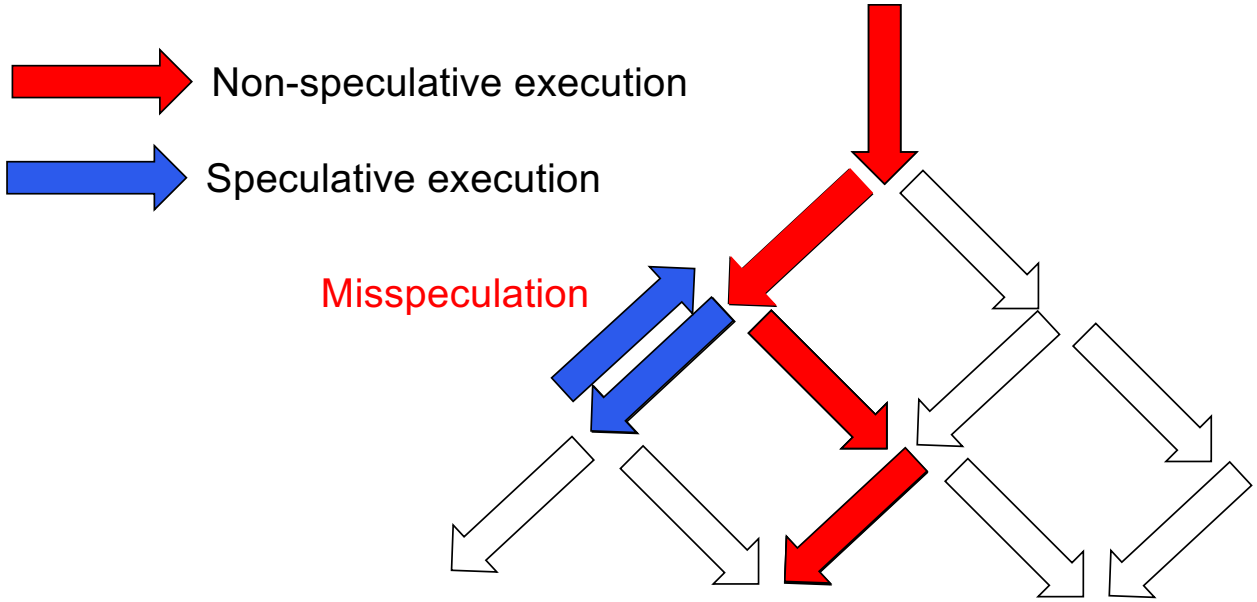
Non-speculative Execution



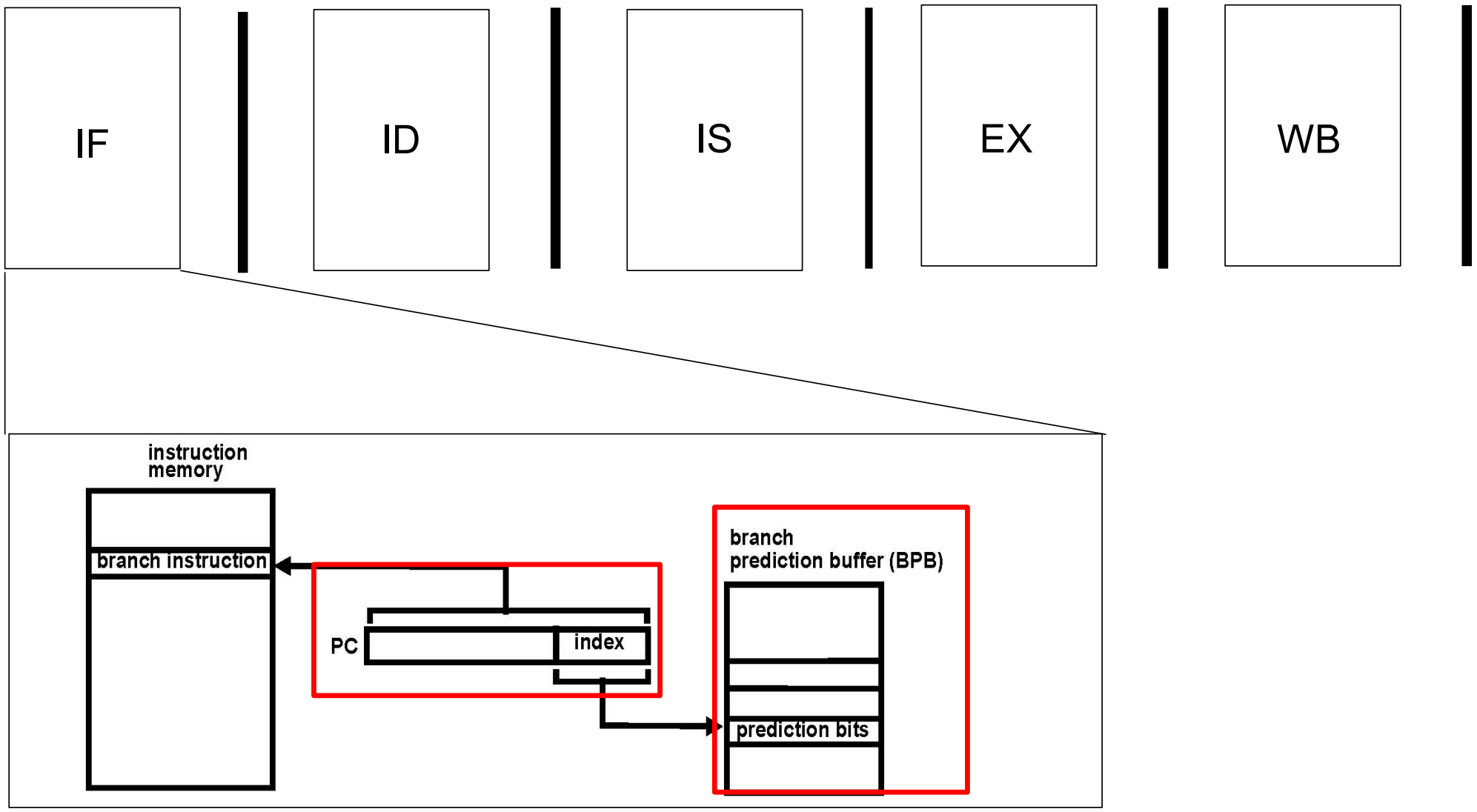
Speculative Execution



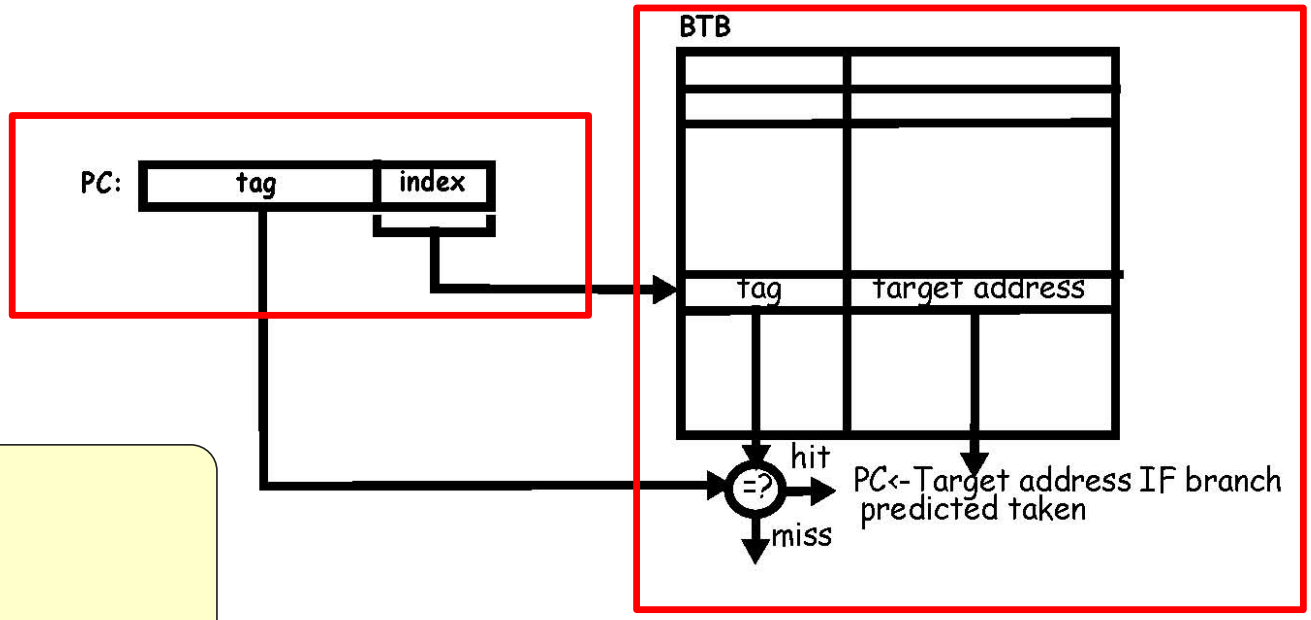
Speculative Execution



Dynamic Branch Prediction

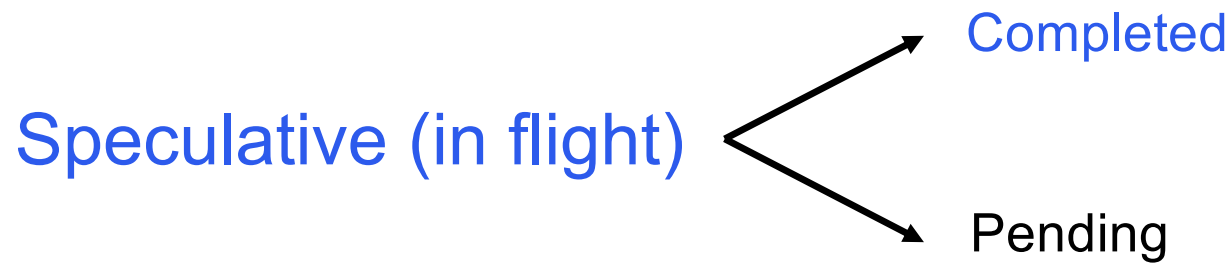

IF
ID
IS
EX
WB
instruction
memory
branch instruction
PC
index
branch
prediction buffer (BPB)
prediction bits

Branch Target Buffer (BTB)



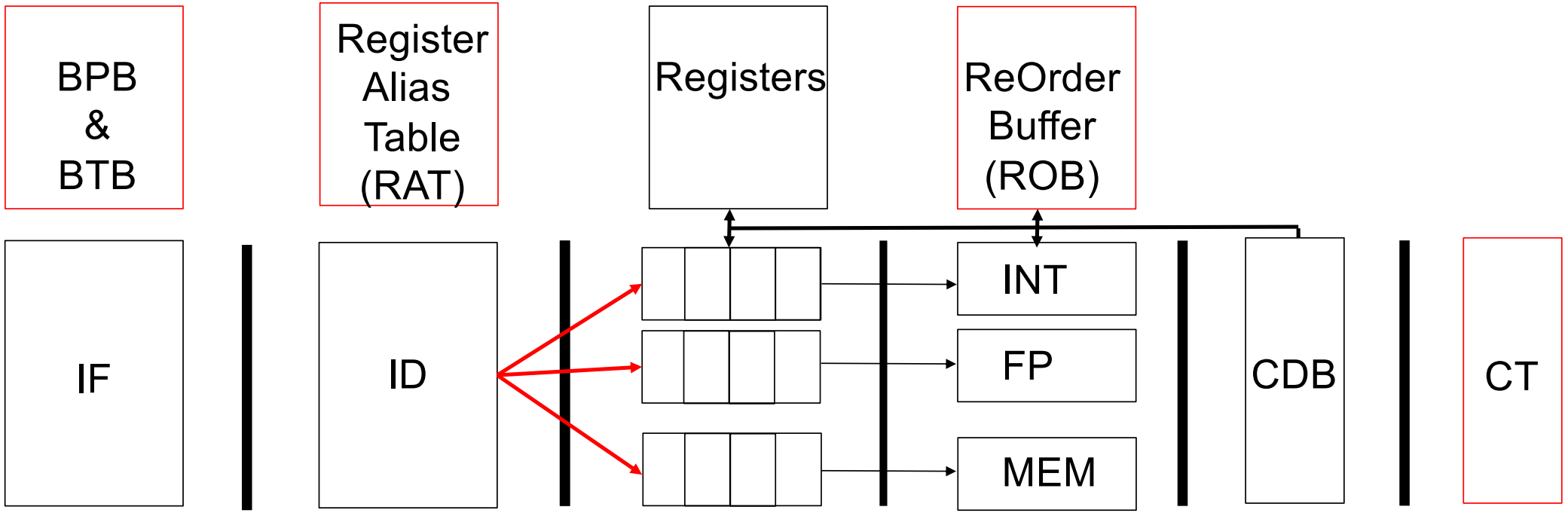
Discussion:
How is a miss handled?

Two Execution Modes



Committed (non-speculative)

Question:
What is the difference between a **completed** and a **committed** instruction?


BPB
&
BTB
Register
Alias
Table
(RAT)
Registers
ReOrder
Buffer
(ROB)
IF
ID
INT
FP
MEM
CDB
CT

Register Alias Table (RAT)

Register	Tag	Status
R0	--	Committed
R1	--	Committed
R2	ROB entry	Completed
…	…	…
R31	ROB entry	Completed
F0	ROB entry	Pending
F1	ROB entry	Pending
F2	--	Committed
...	...	…
F31	--	Committed

Register Alias Table (RAT)

Question:
Where can the values of registers R0, R1 and R2 be found given the content of the RAT?

Answer:
Registers R0 and R1 are committed so their values are found in the registerfile. Register R2 is completed, but not committed so the value is in ROB entry 0

Register	Tag	Status
R0	--	Committed
R1	--	Committed
R2	ROB 0	Completed
…	…	…
R31	ROB 1	Completed
F0	ROB 10	Pending
F1	ROB 11	Pending
F2	--	Committed
...	...	…
F31	--	Committed

ReOrder Buffer (ROB)

Entry	Instruction	Dest. reg	Value	Complete
15	BNEZ R3,Loop	--	--	YES
16	L.S F0,0(R1)	F0	1.2E-4	YES
17	L.S F1,0(R2)	F1	5.8E-3	YES
18	ADD.S F2,F1,F0	F2	2.0E-2	YES
19	S.S F2,0(R1)	--	--	NO
20	ADDI R1,R1,#4	R1	N/A	NO
21	ADDI R2,R2,#4	R2	N/A	NO
22	SUBI R3,R3,#1	R3	N/A	NO
23	BNEZ R3,Loop	--	--	--

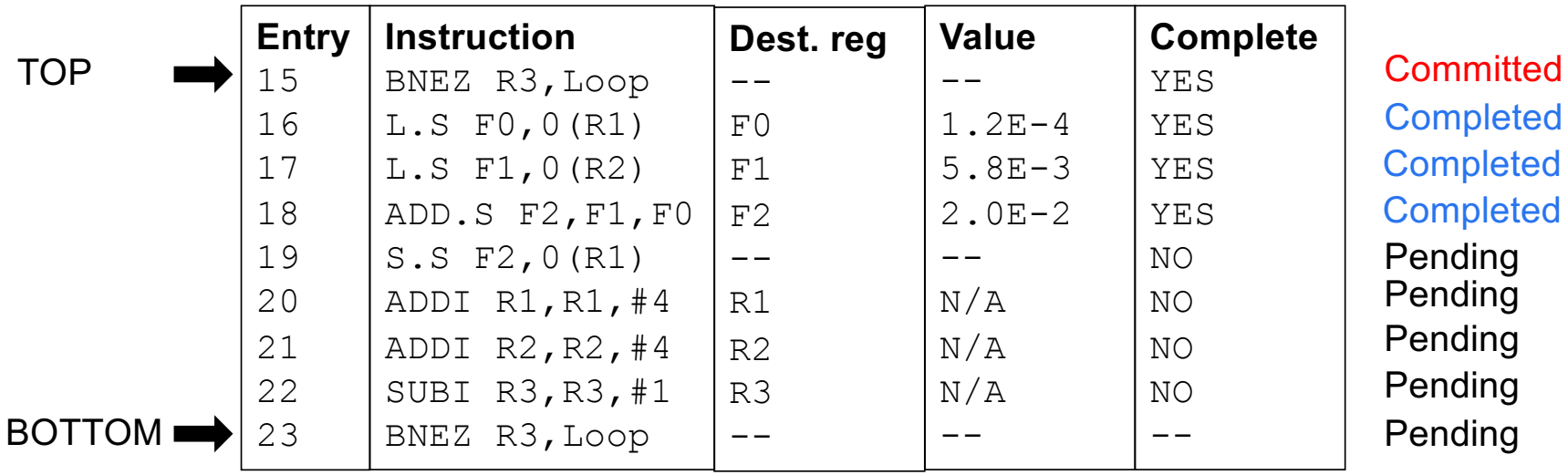
ReOrder Buffer (ROB)

Question:
Where can the values of F2 and R3 be found?

Answer:
F2 is in the reorder buffer but R3 is not available as an instruction is speculatively producing its value. This can be seen as the status is not completed.

Entry	Instruction	Dest. reg	Value	Complete
15	BNEZ R3,Loop	--	--	YES
16	L.S F0,0(R1)	F0	1.2E-4	YES
17	L.S F1,0(R2)	F1	5.8E-3	YES
18	ADD.S F2,F1,F0	F2	2.0E-2	YES
19	S.S F2,0(R1)	--	--	NO
20	ADDI R1,R1,#4	R1	N/A	NO
21	ADDI R2,R2,#4	R2	N/A	NO
22	SUBI R3,R3,#1	R3	N/A	NO
23	BNEZ R3,Loop	--	--	--

ReOrder Buffer (ROB)



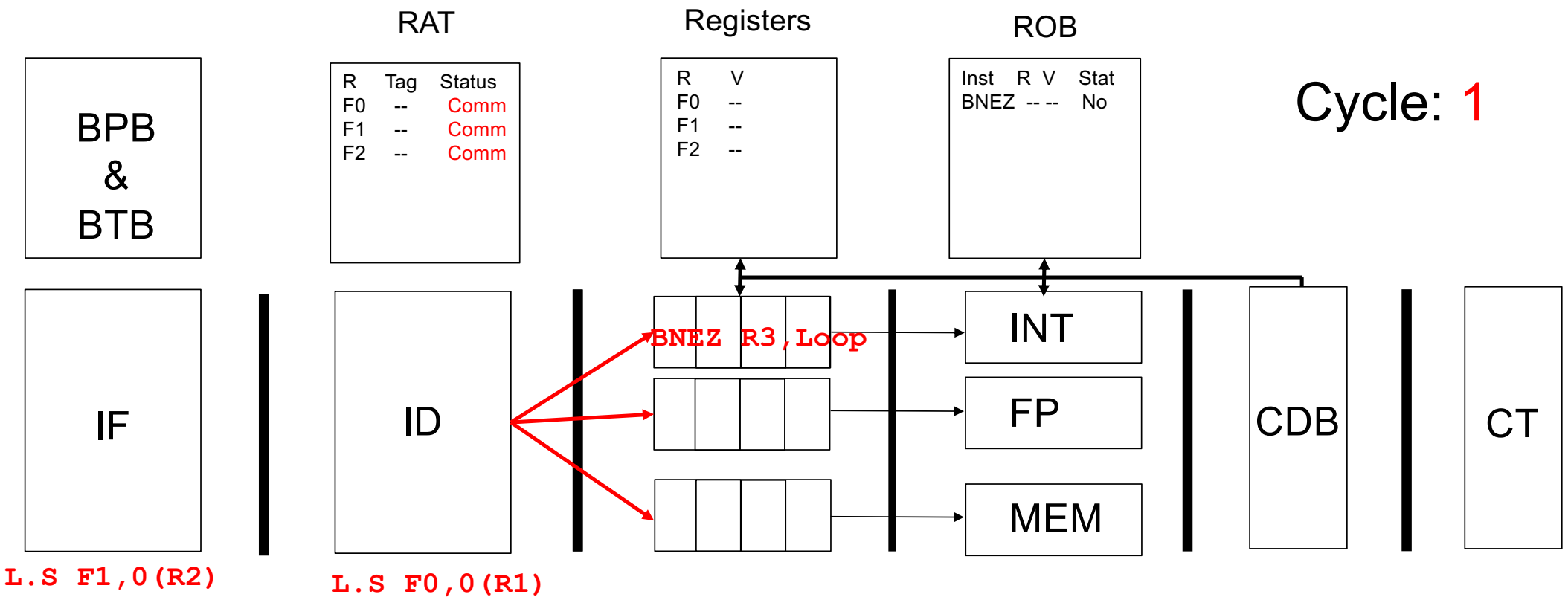
Entry	Instruction	Dest. reg	Value	Complete	
15	BNEZ R3,Loop	--	--	YES	Committed
16	L.S F0,0(R1)	F0	1.2E-4	YES	Completed
17	L.S F1,0(R2)	F1	5.8E-3	YES	Completed
18	ADD.S F2,F1,F0	F2	2.0E-2	YES	Completed
19	S.S F2,0(R1)	--	--	NO	Pending
20	ADDI R1,R1,#4	R1	N/A	NO	Pending
21	ADDI R2,R2,#4	R2	N/A	NO	Pending
22	SUBI R3,R3,#1	R3	N/A	NO	Pending
23	BNEZ R3,Loop	--	--	--	Pending

Tag = Reorder buffer entry

ReOrder Buffer (ROB)

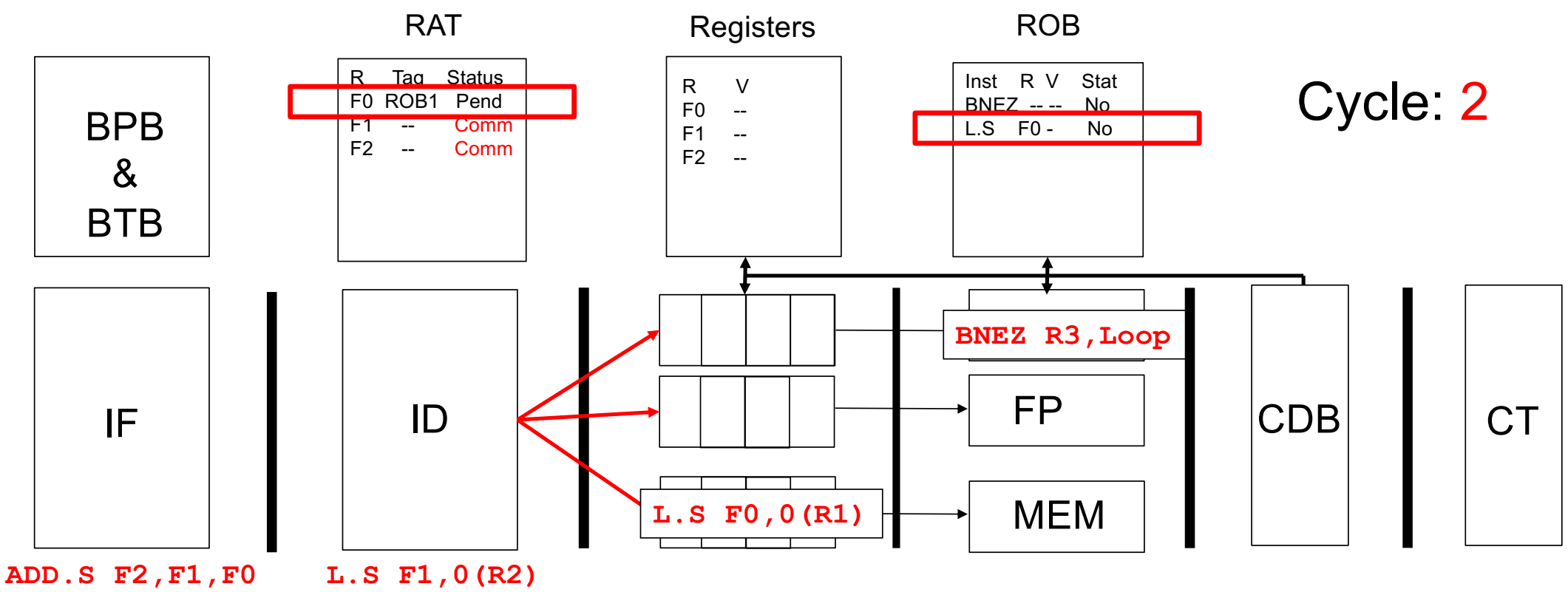
	Entry	Instruction	Dest. reg	Value	Complete	
TOP →	16	L.S F0,0(R1)	F0	1.2E-4	YES	Committed
	17	L.S F1,0(R2)	F1	5.8E-3	YES	Completed
	18	ADD.S F2,F1,F0	F2	2.0E-2	YES	Completed
	19	S.S F2,0(R1)	--	--	YES	Completed
	20	ADDI R1,R1,#4	R1	N/A	NO	Pending
	21	ADDI R2,R2,#4	R2	N/A	NO	Pending
	22	SUBI R3,R3,#1	R3	N/A	NO	Pending
	23	BNEZ R3,Loop	--	N/A	NO	Pending
BOTTOM →	24	L.S F0,0(R1)	F0	N/A	NO	Pending

Speculative Execution

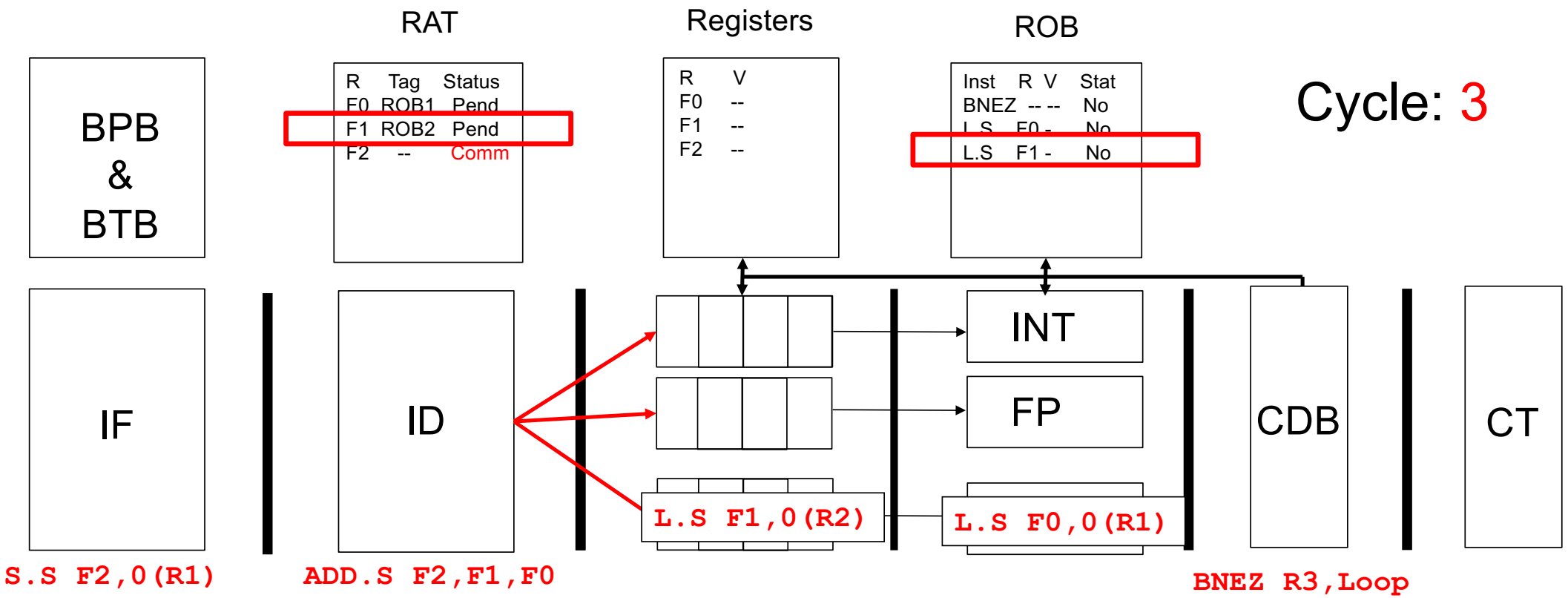


Here, the branch instruction is speculatively executed

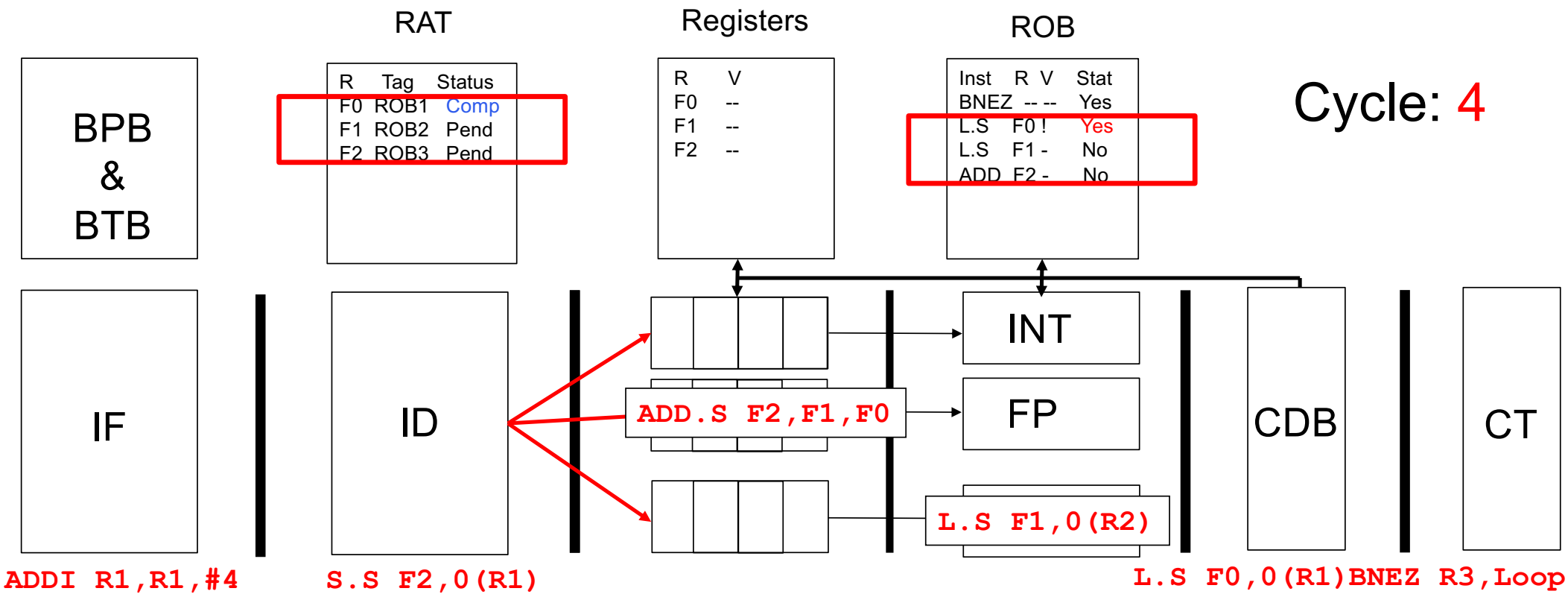
```
L.S F0,0(R1)
L.S F1,0(R2)
ADD.S F2,F1,F0
S.S F2,0(R1)
ADDI R1,R1,#4
ADDI R2,R2,#4
SUBI R3,R3,#1
BNEZ R3,Loop
```



L.S F0,0(R1)
L.S F1,0(R2)
ADD.S F2,F1,F0
S.S F2,0(R1)
ADDI R1,R1,#4
ADDI R2,R2,#4
SUBI R3,R3,#1
BNEZ R3,Loop

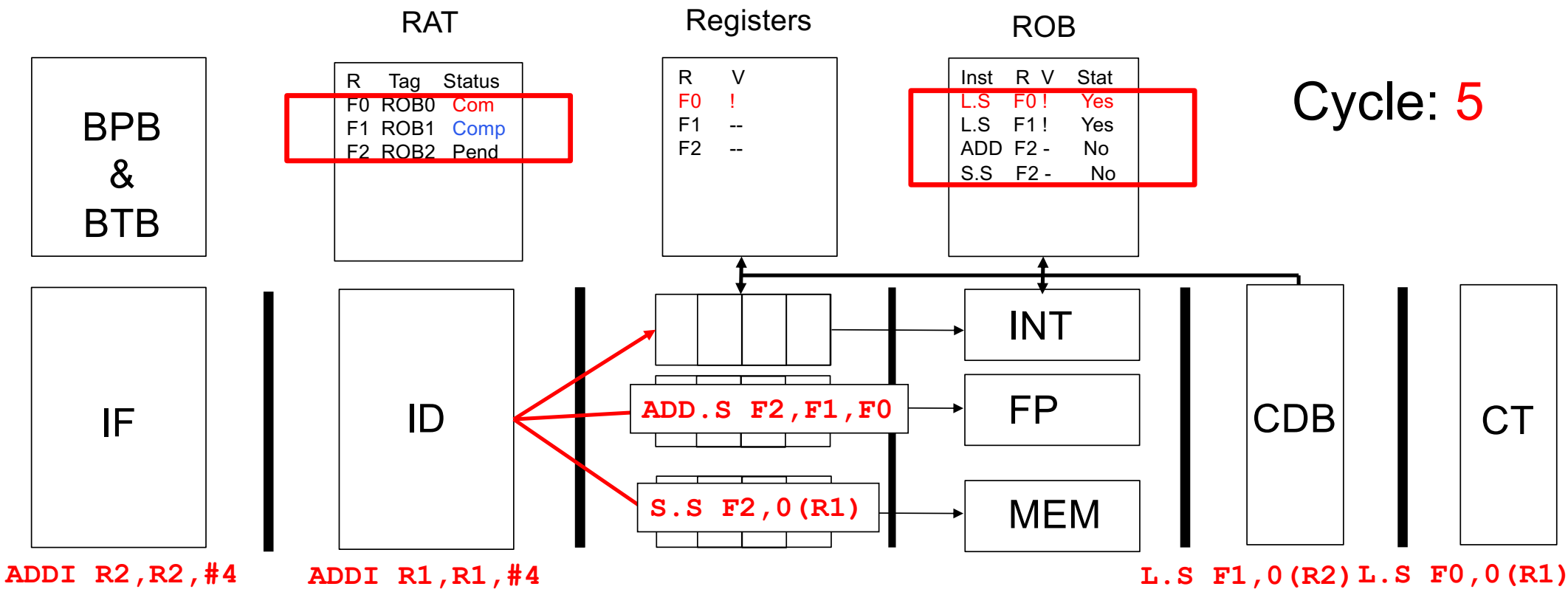



```
L.S F0,0(R1)
L.S F1,0(R2)
ADD.S F2,F1,F0
S.S F2,0(R1)
ADDI R1,R1,#4
ADDI R2,R2,#4
SUBI R3,R3,#1
BNEZ R3,Loop
```

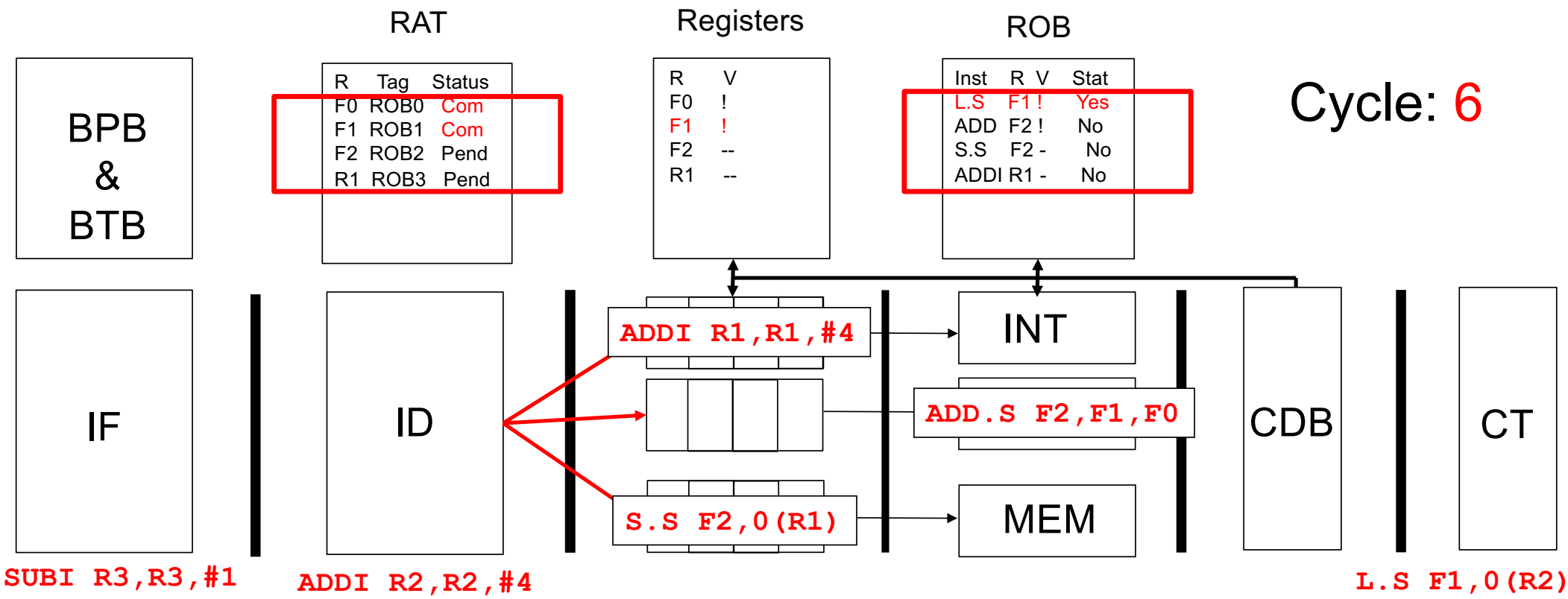


Here, the branch instruction commits

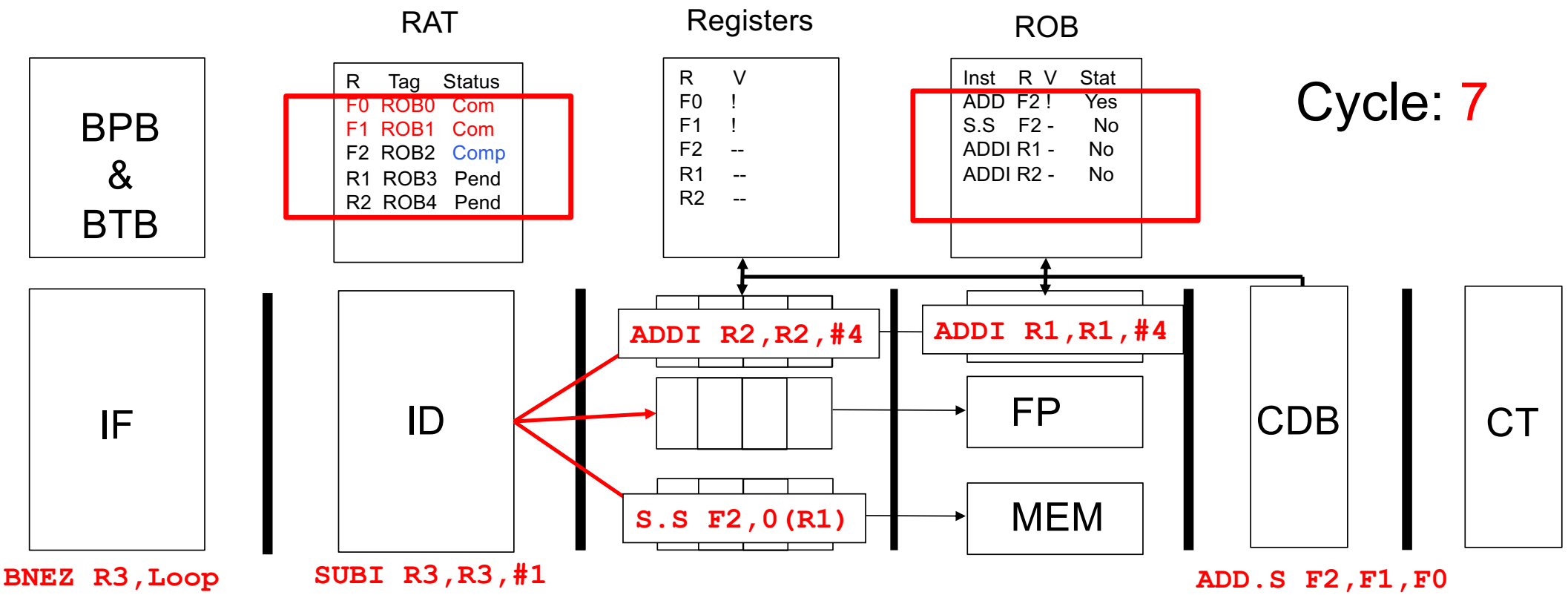
```
L.S  F0,0(R1)
L.S  F1,0(R2)
ADD.S F2,F1,F0
S.S  F2,0(R1)
ADDI R1,R1,#4
ADDI R2,R2,#4
SUBI R3,R3,#1
BNEZ R3,Loop
```



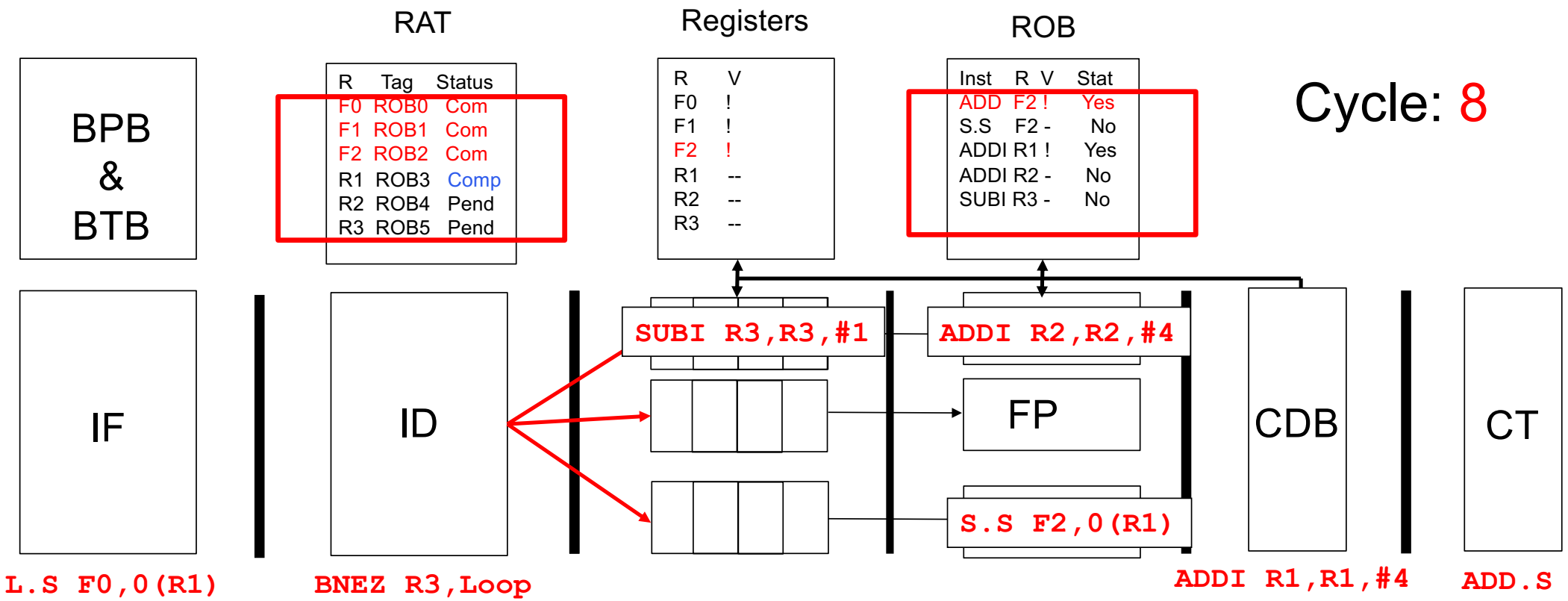

```
L.S F0,0(R1)
L.S F1,0(R2)
ADD.S F2,F1,F0
S.S F2,0(R1)
ADDI R1,R1,#4
ADDI R2,R2,#4
SUBI R3,R3,#1
BNEZ R3,Loop
```



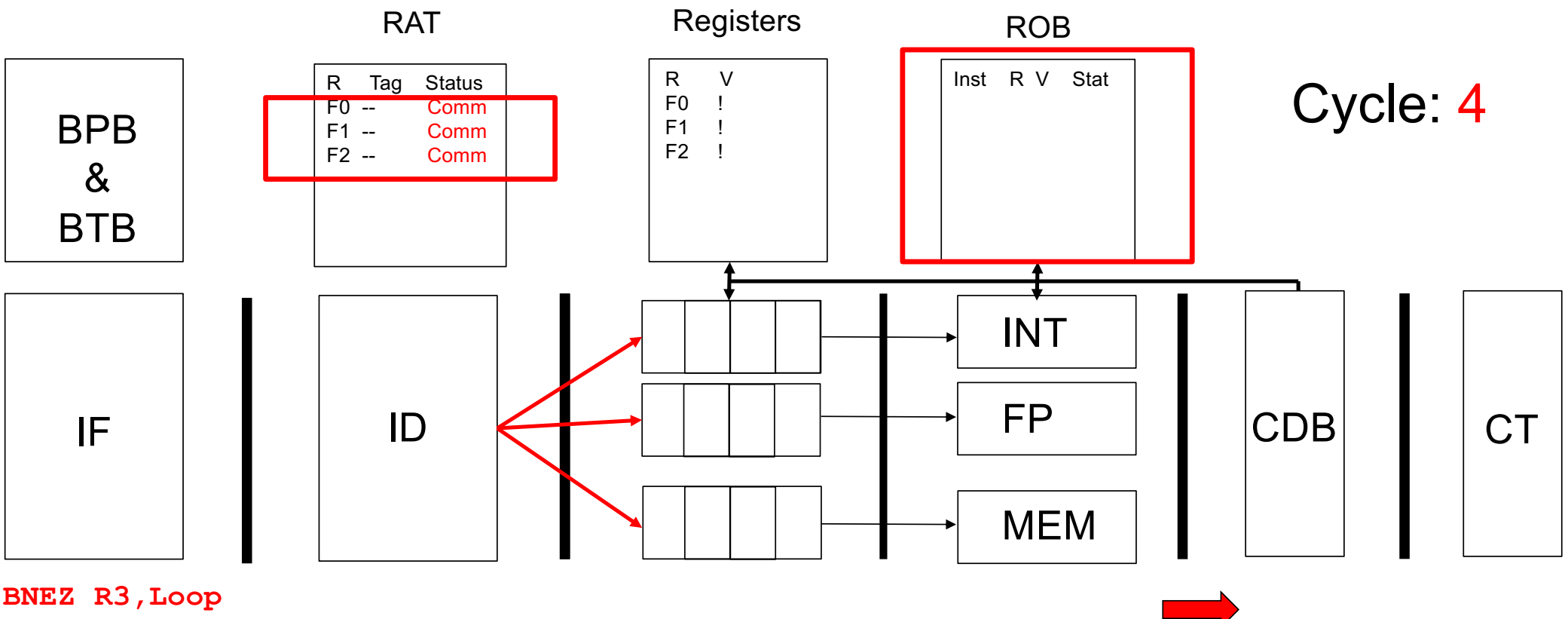

```
L.S F0,0(R1)
L.S F1,0(R2)
ADD.S F2,F1,F0
S.S F2,0(R1)
ADDI R1,R1,#4
ADDI R2,R2,#4
SUBI R3,R3,#1
BNEZ R3,Loop
```




```
L.S F0,0(R1)
L.S F1,0(R2)
ADD.S F2,F1,F0
S.S F2,0(R1)
ADDI R1,R1,#4
ADDI R2,R2,#4
SUBI R3,R3,#1
BNEZ R3,Loop
```




```
L.S F0,0(R1)
L.S F1,0(R2)
ADD.S F2,F1,F0
S.S F2,0(R1)
ADDI R1,R1,#4
ADDI R2,R2,#4
SUBI R3,R3,#1
BNEZ R3,Loop
```




```
L.S F0,0(R1)
L.S F1,0(R2)
ADD.S F2,F1,F0
S.S F2,0(R1)
ADDI R1,R1,#4
ADDI R2,R2,#4
SUBI R3,R3,#1
BNEZ R3,Loop
```

Mispredicted Branch

Question:
Consider the program below and determine how many cycles it takes to execute two iterations on the speculative pipeline assuming that the branch is correctly predicted.

```
Loop    L.S F0,0(R1)
        ADD.S F2,F1,F0
        S.S F2,0(R1)
        ADDI R1,R1,#4
        SUBI R3,R3,#1
        BNEZ R3,Loop
```

Answer:
18 cycles

```
Loop I1:L.S F0,0(R1)
     I2:ADD.S F2,F1,F0
     I3:S.S F2,0(R1)
     I4:ADDI R1,R1,#4
     I5:SUBI R3,R3,#1
     I6:BNEZ R3,Loop
```

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	C20	C21	C22	C23	C24
I1,1	IF	ID	IS	EX	WB																			
I2,1		IF	ID	IS	EX	EX	EX	EX	EX	WB														
I3,1			IF	ID	IS	IS	IS	IS	IS	EX	WB													
I4,1				IF	ID	IS	EX	WB																
I5,1					IF	ID	IS	EX	WB															
I6,1						IF	ID	IS	EX	WB	WB	WB												
I1,2							IF	ID	IS	EX	WB	WB	WB											
I2,2								IF	ID	IS	EX	EX	EX	EX	EX	WB								
I3,2									IF	ID	IS	IS	IS	IS	IS	EX	WB							
I4,2										IF	ID	IS	EX	WB										
I5,2											IF	ID	IS	EX	WB									
I6,2												IF	ID	IS	EX	WB	WB	WB						

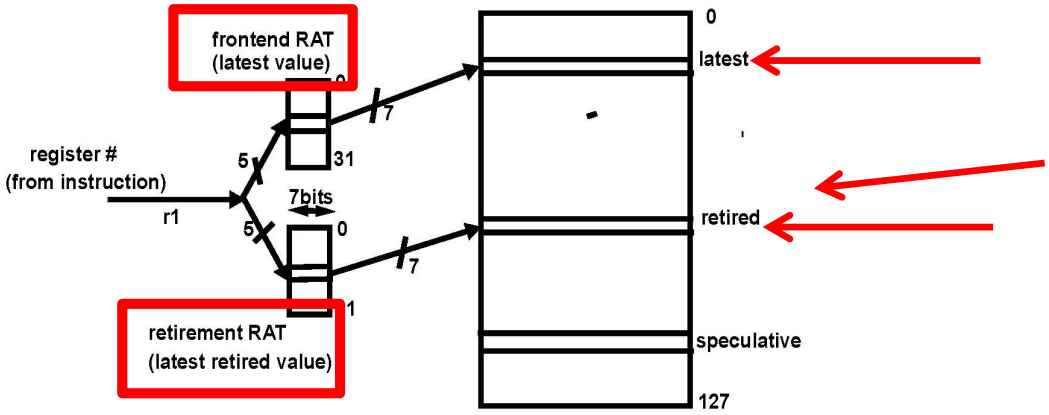
RAT

Register	Tag	Status
R1	--	--
R2	--	--
R3	--	--
F0	--	--
F1	--	--
F2	--	--
F3	--	--
F4	--	--

ROB

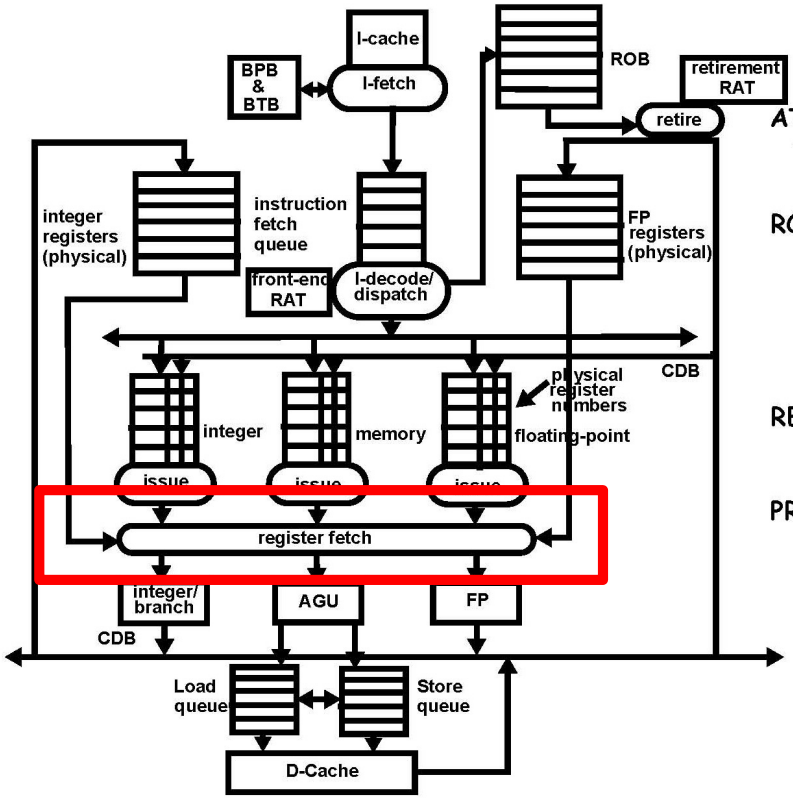
Entry	Instruction	Dest. reg	Value	Complete
16	L.S F0,0(R1)	F0	1.2E-4	YES
17	L.S F1,0(R2)	F1	5.8E-3	YES
18	ADD.S F2,F1,F0	F2	--	NO
19	ADD.S F1,F3,F4	F1	--	NO
20	SD F2,0(R1)	F2	N/A	NO
21	ADDI R1,R1,#4	R1	N/A	NO
22	ADDI R2,R2,#4	R2	N/A	NO
23	SUBI R3,R3,#1	R3	N/A	NO
24	BNEZ R3,Loop	F0	N/A	NO
25	L.S F0,0(R1)		N/A	NO

Register Renaming: Structures



- **Frontend RAT points to most recent value. Retirement RAT points to most recent retired value (could be the same)**
- **More speculative values of the same architectural register may be in the register file**
- **“Updating register” now means “Updating the pointer in the retirement RAT”**

Register Fetch After Issue



AT DISPATCH INPUT REGISTER # (NOT ITS VALUE) IS DISPATCHED WITH READY BIT

ROLE OF THE CDB:

- TRANSFER VALUES TO REGISTER
- AWAKE INSTRUCTIONS BY SETTING READY BITS IN ISSUE Qs

REGISTERS ARE FETCHED RIGHT BEFORE EXECUTION

PROBLEMS:

- EFFECTIVE LATENCY MUCH HIGHER
- SIMILAR PROBLEM IN TOMASULO

Memory disambiguation

- Loads and stores must execute in program order

Example)

SD.S F0, 0(R2)

LD.S F1, 0(R2)

If Mem[(R2)] = 0, and (F2) = 100 what will be in F1 after both instructions are executed?

SD.S F0, 0(R2)

LD.S F1, 0(R3)

What will be in F1 now? Could it be 100 under any condition?, what condition is that?

Exception handling

- Note: On a misspeculated branch instruction, all instructions before that branch will be fully executed and all instructions after will be squashed
- This is EXACTLY what is needed to handle PRECISE EXCEPTIONS