- We feed the machine the following program:

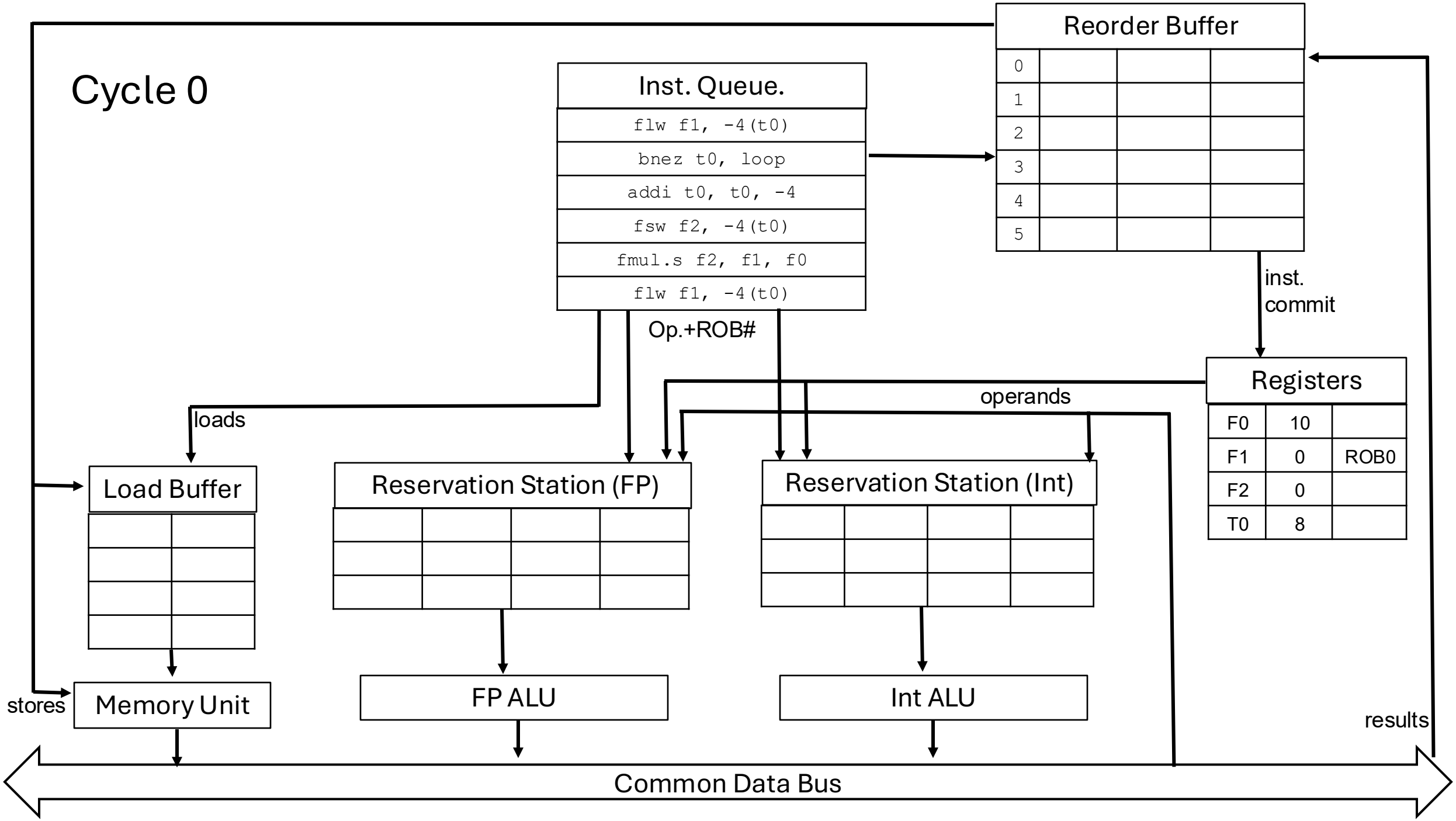
```
loop:
  flw f1, -4(t0)
  fmul.s f2, f1, f0
  fsw f2, -4(t0)
  addi t0, t0, -4
  bnez t0, loop

fmul.s f3,f2,f4   #just filling
fsub.s f5,f1,f2
fdiv.s f0,f3,f1
fadd.s f1,f5,f2
```

- loads and stores take 2 cycles
- float muls/divs take 4 cycles
- float adds/subs take 3 cycles
- integer operations take 2 cycles

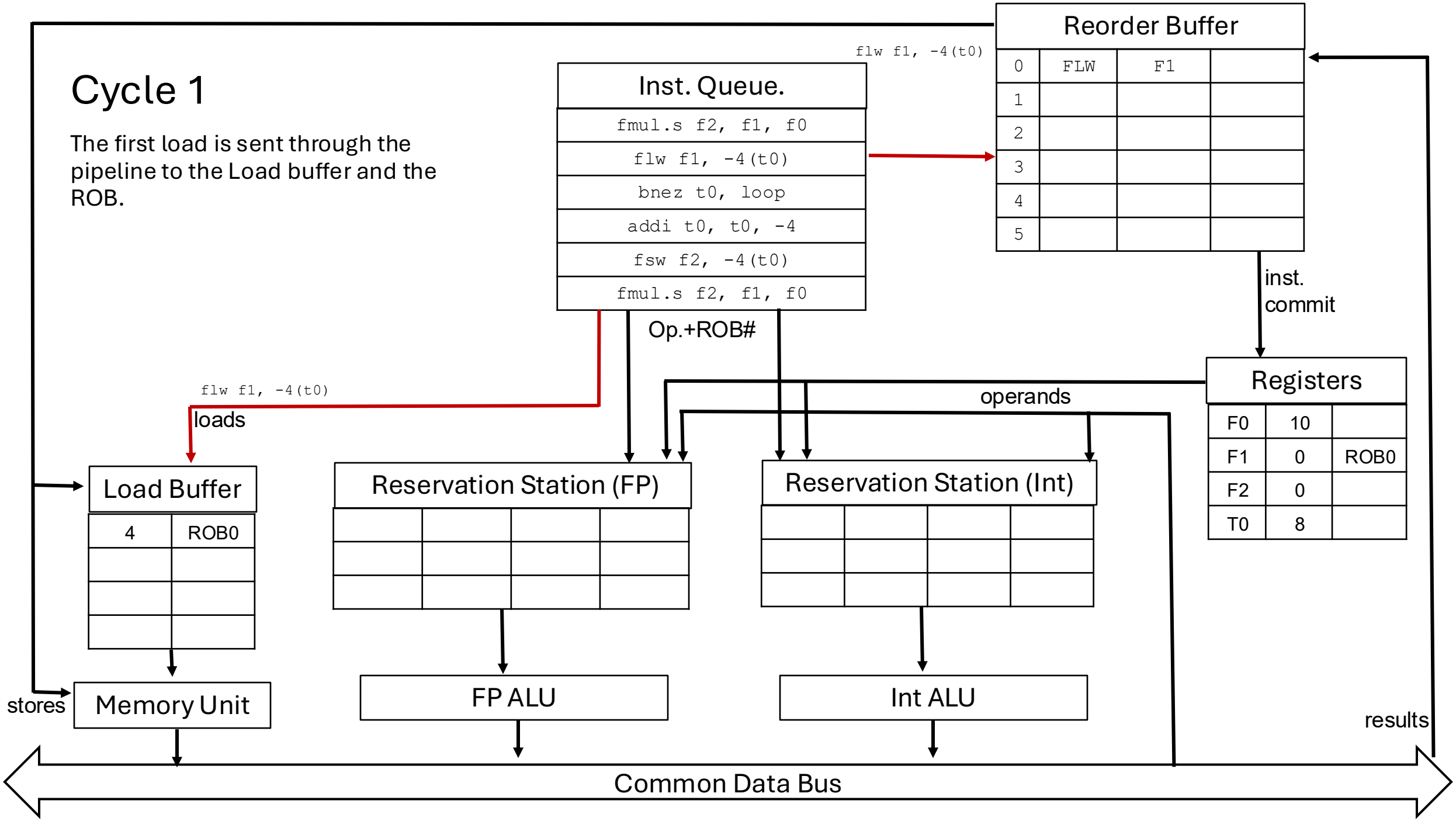
- The branch predictor is correct at the first `bnez` instruction and takes the loop.
- The second time it finds the `bnez`, it predicts jump and is wrong, the ROB needs to be flushed.

The loop performs a float vector multiplication by a scalar in `F0`


Cycle 0
Reorder Buffer
0
1
2
3
4
5
Inst. Queue.
flw f1, -4(t0)
bnez t0, loop
addi t0, t0, -4
fsw f2, -4(t0)
fmul.s f2, f1, f0
flw f1, -4(t0)
Op.+ROB#
inst.
commit
Registers
F0 10
F1 0 ROB0
F2 0
T0 8
operands
loads
Load Buffer
Reservation Station (FP)
Reservation Station (Int)
stores
Memory Unit
FP ALU
Int ALU
results
Common Data Bus

Cycle 1

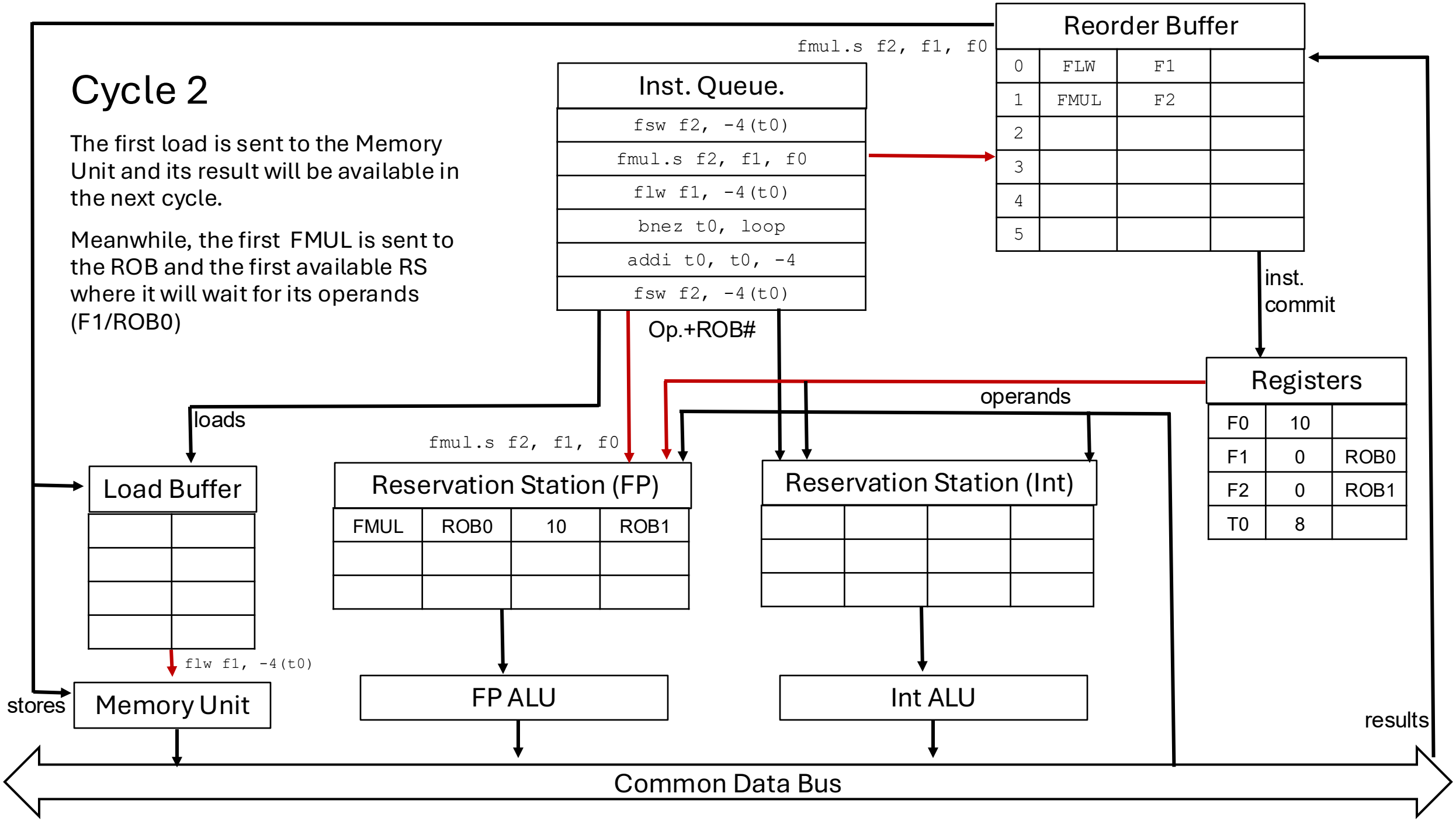
The first load is sent through the pipeline to the Load buffer and the ROB.



Cycle 2

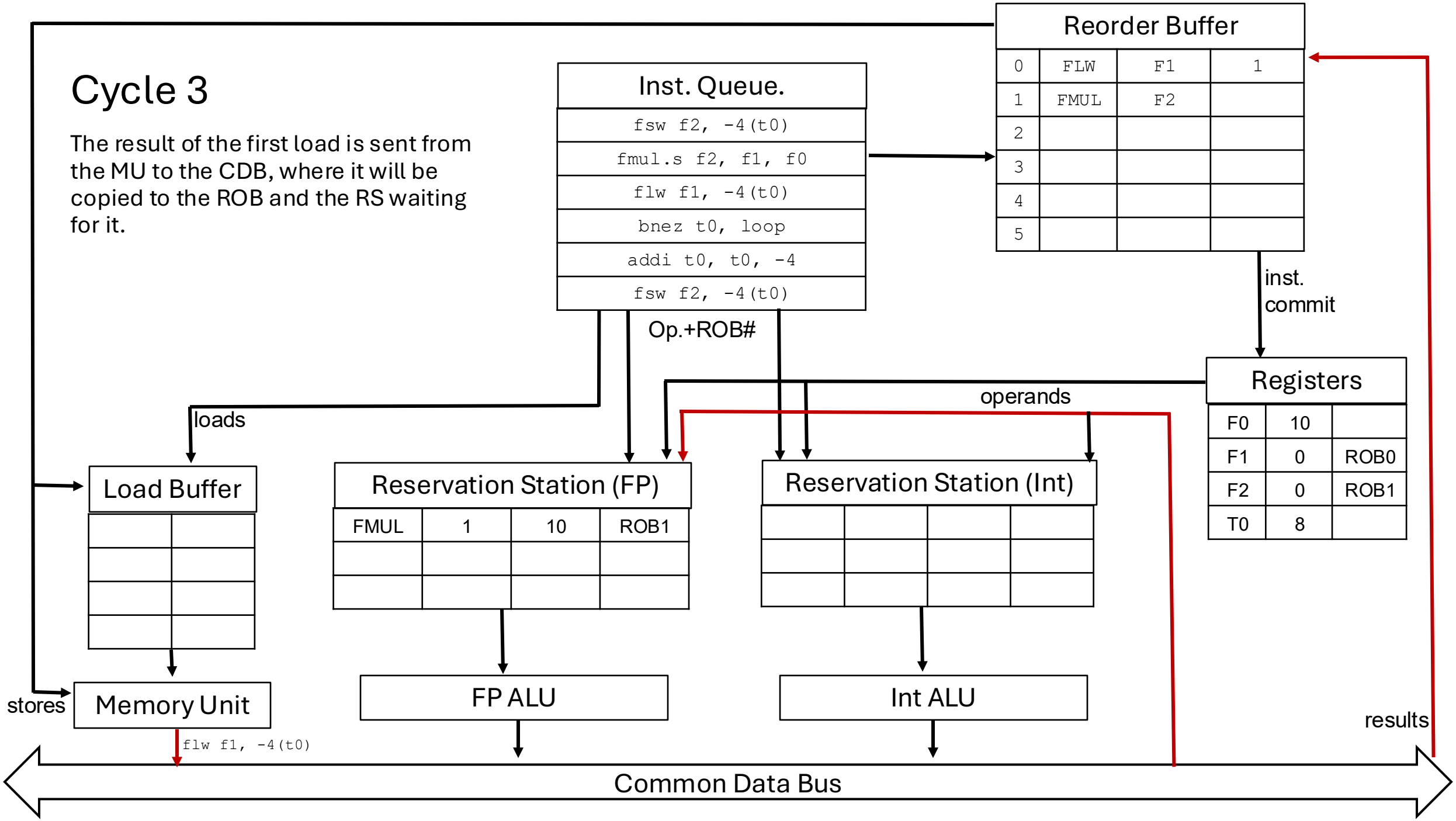
The first load is sent to the Memory Unit and its result will be available in the next cycle.

Meanwhile, the first FMUL is sent to the ROB and the first available RS where it will wait for its operands (F1/ROB0)



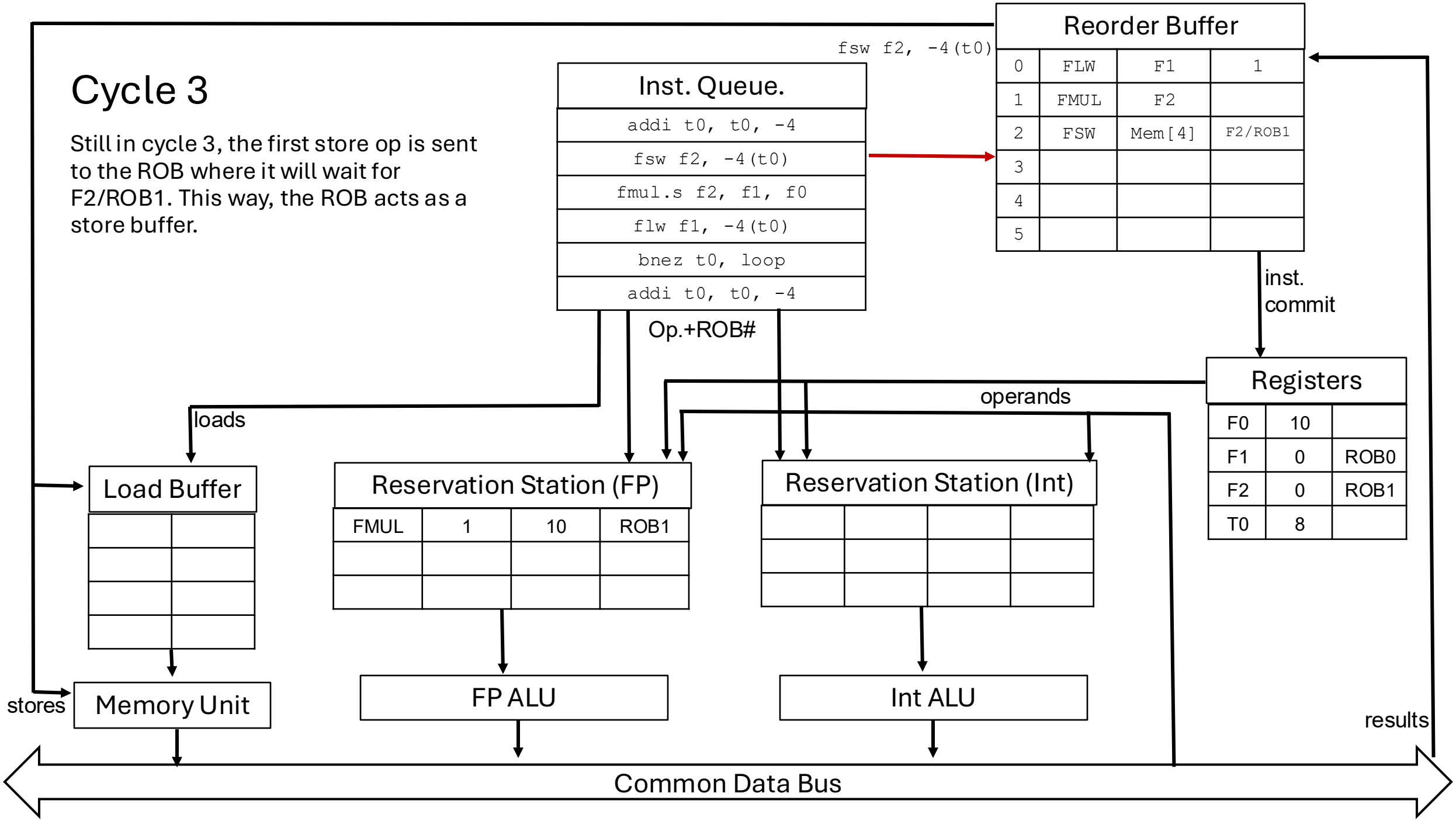
Cycle 3

The result of the first load is sent from the MU to the CDB, where it will be copied to the ROB and the RS waiting for it.



Cycle 3

Still in cycle 3, the first store op is sent to the ROB where it will wait for F2/ROB1. This way, the ROB acts as a store buffer.

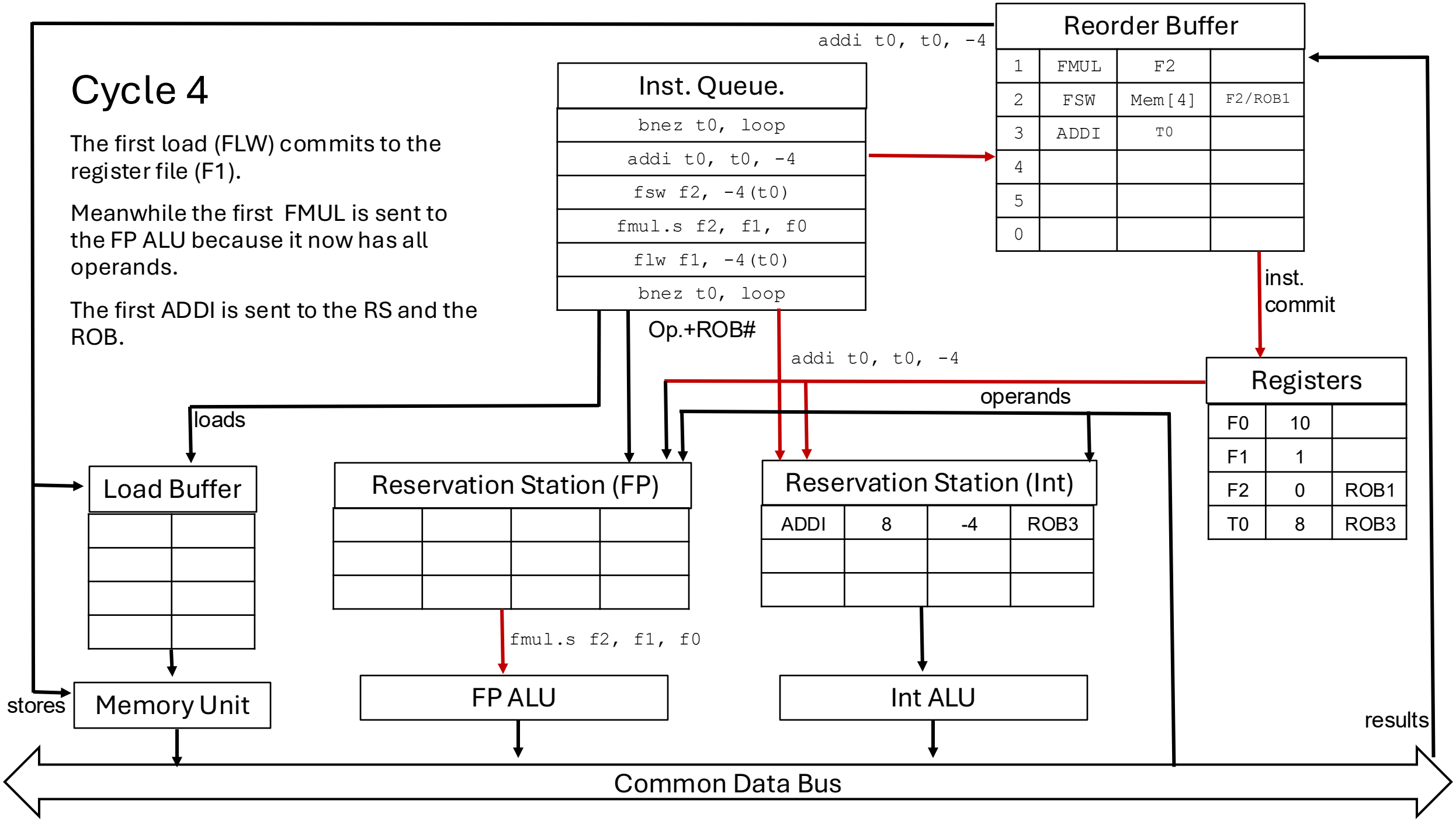


Cycle 4

The first load (FLW) commits to the register file (F1).

Meanwhile the first FMUL is sent to the FP ALU because it now has all operands.

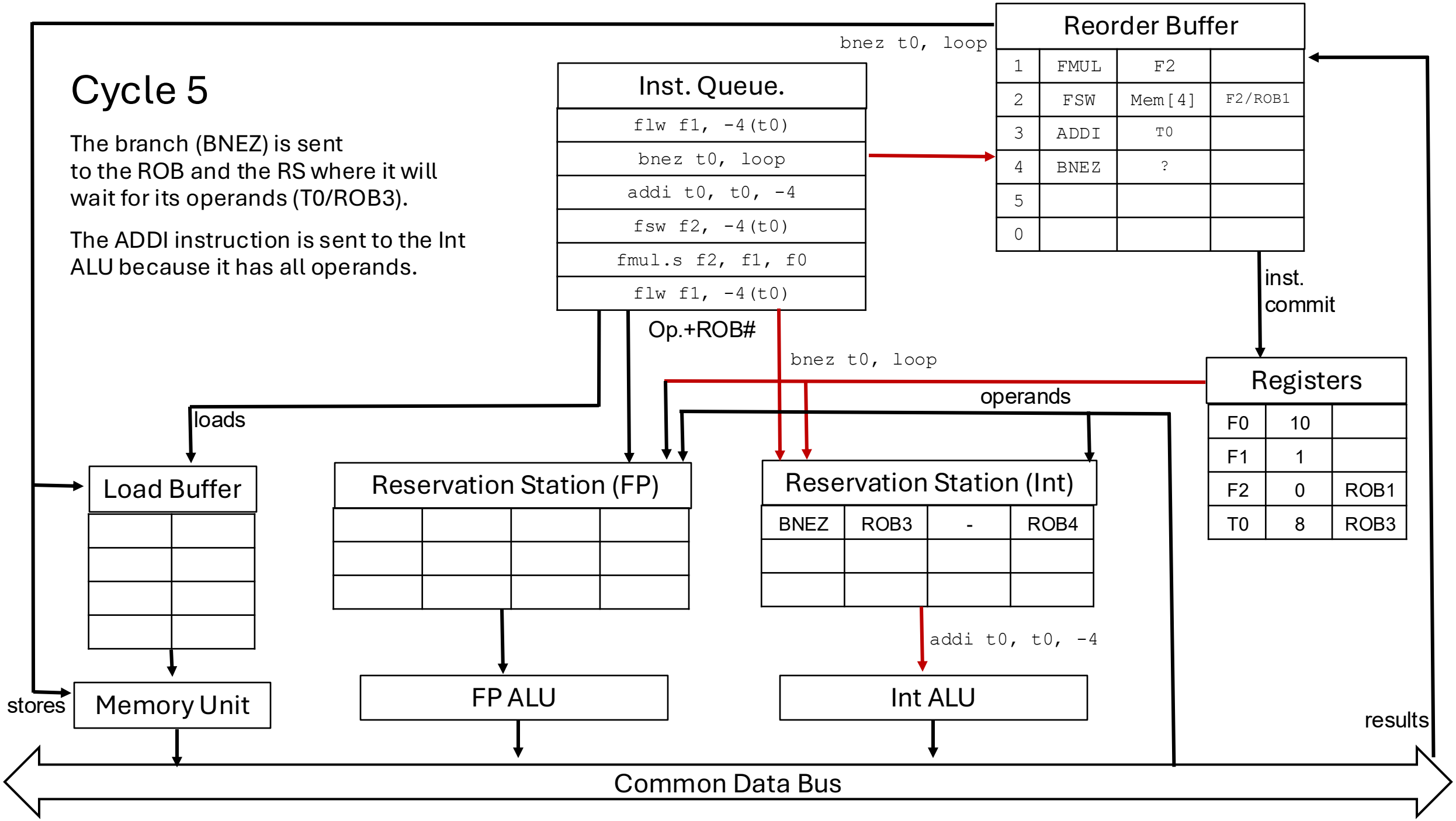
The first ADDI is sent to the RS and the ROB.



Cycle 5

The branch (BNEZ) is sent to the ROB and the RS where it will wait for its operands (T0/ROB3).

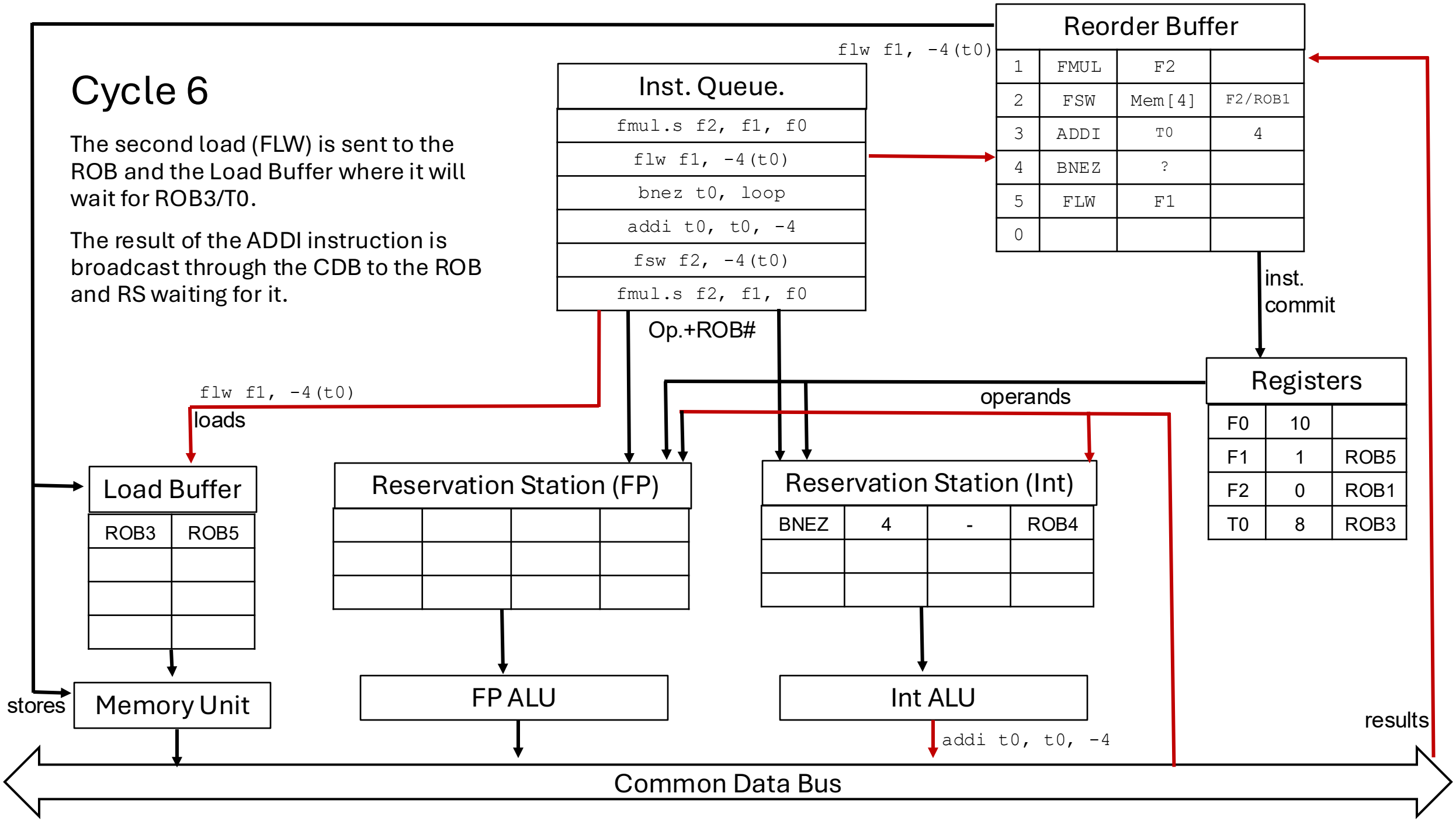
The ADDI instruction is sent to the Int ALU because it has all operands.



Cycle 6

The second load (FLW) is sent to the ROB and the Load Buffer where it will wait for ROB3/T0.

The result of the ADDI instruction is broadcast through the CDB to the ROB and RS waiting for it.

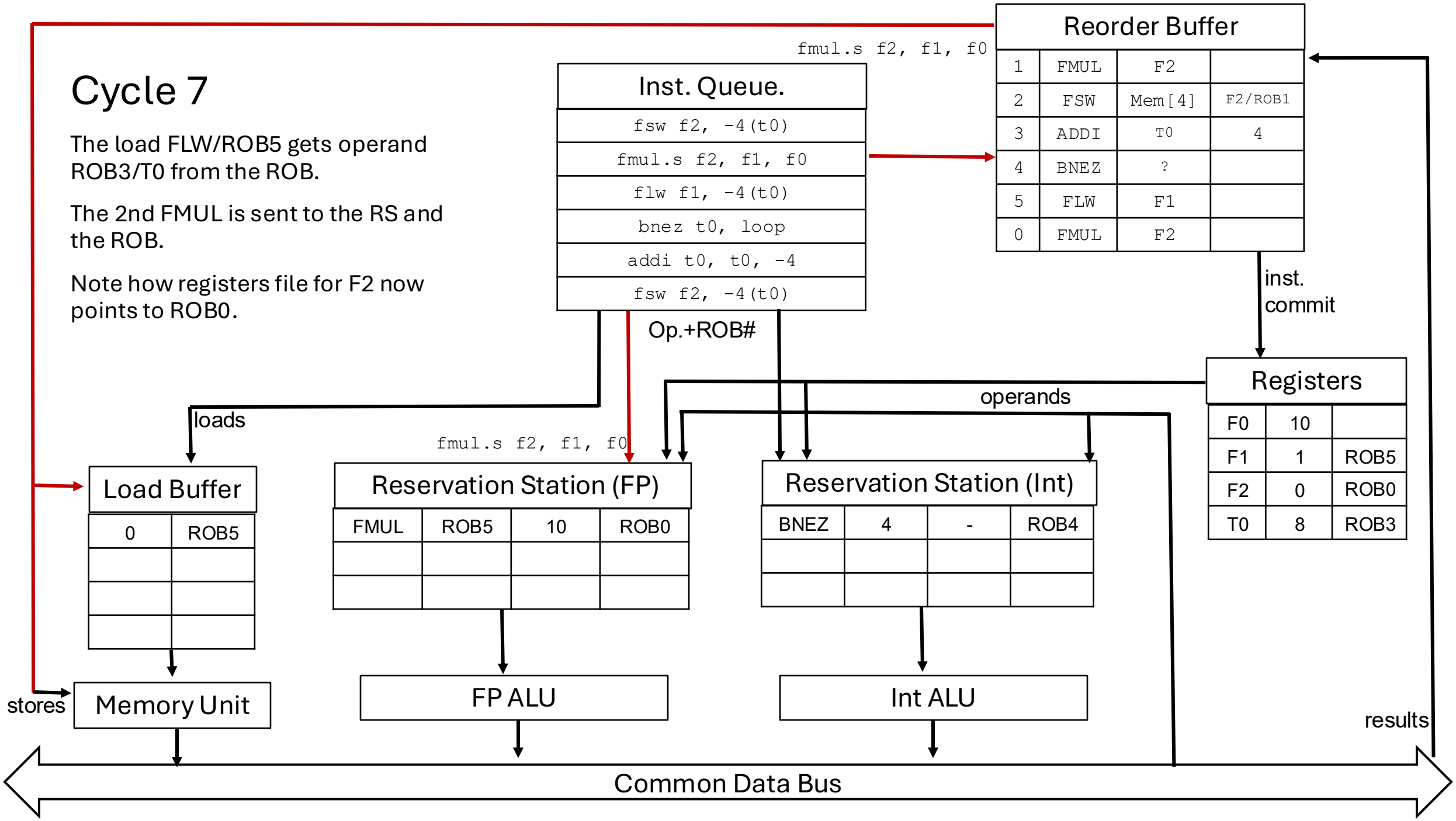


Cycle 7

The load FLW/ROB5 gets operand ROB3/T0 from the ROB.

The 2nd FMUL is sent to the RS and the ROB.

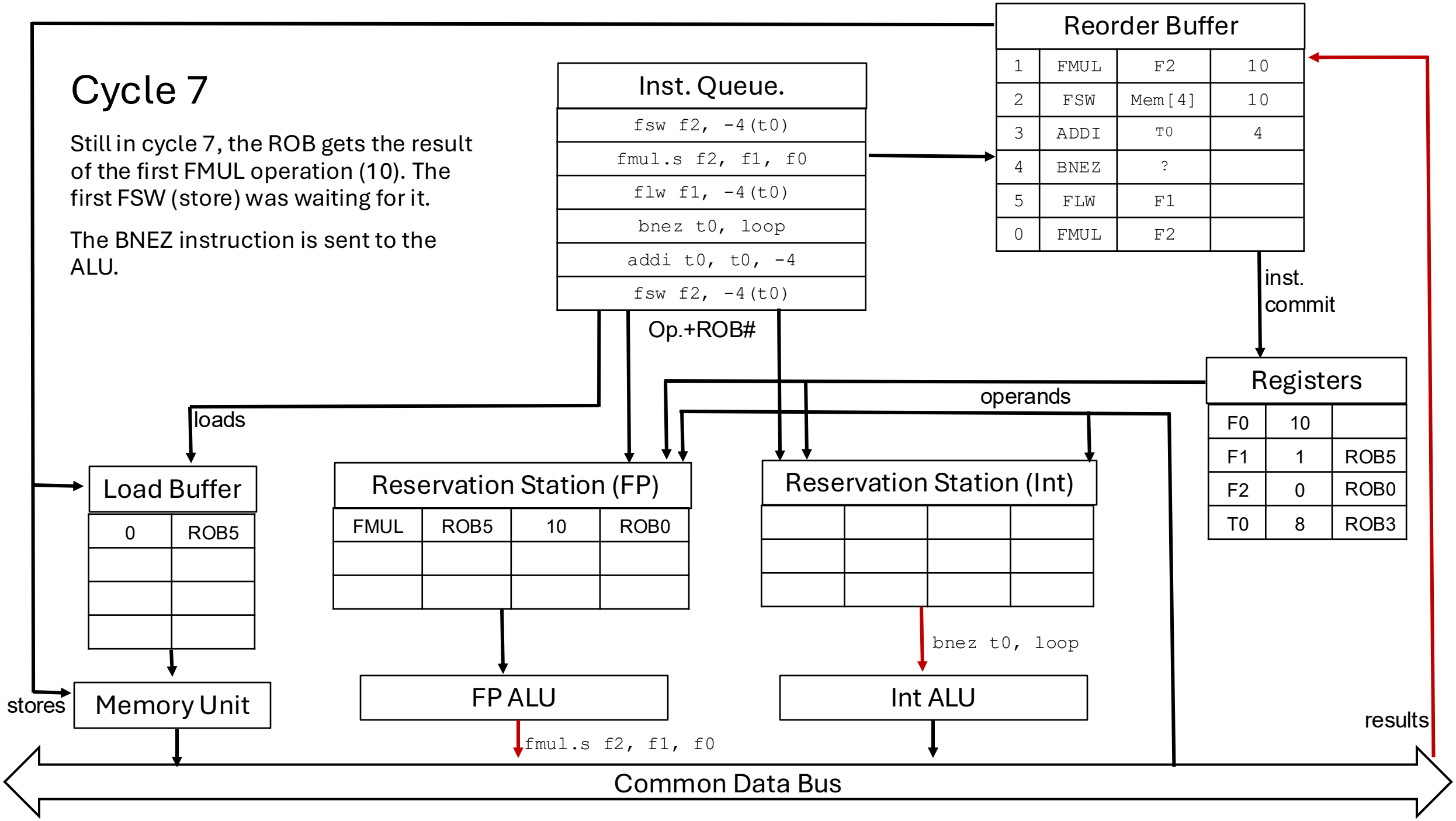
Note how registers file for F2 now points to ROB0.



Cycle 7

Still in cycle 7, the ROB gets the result of the first FMUL operation (10). The first FSW (store) was waiting for it.

The BNEZ instruction is sent to the ALU.



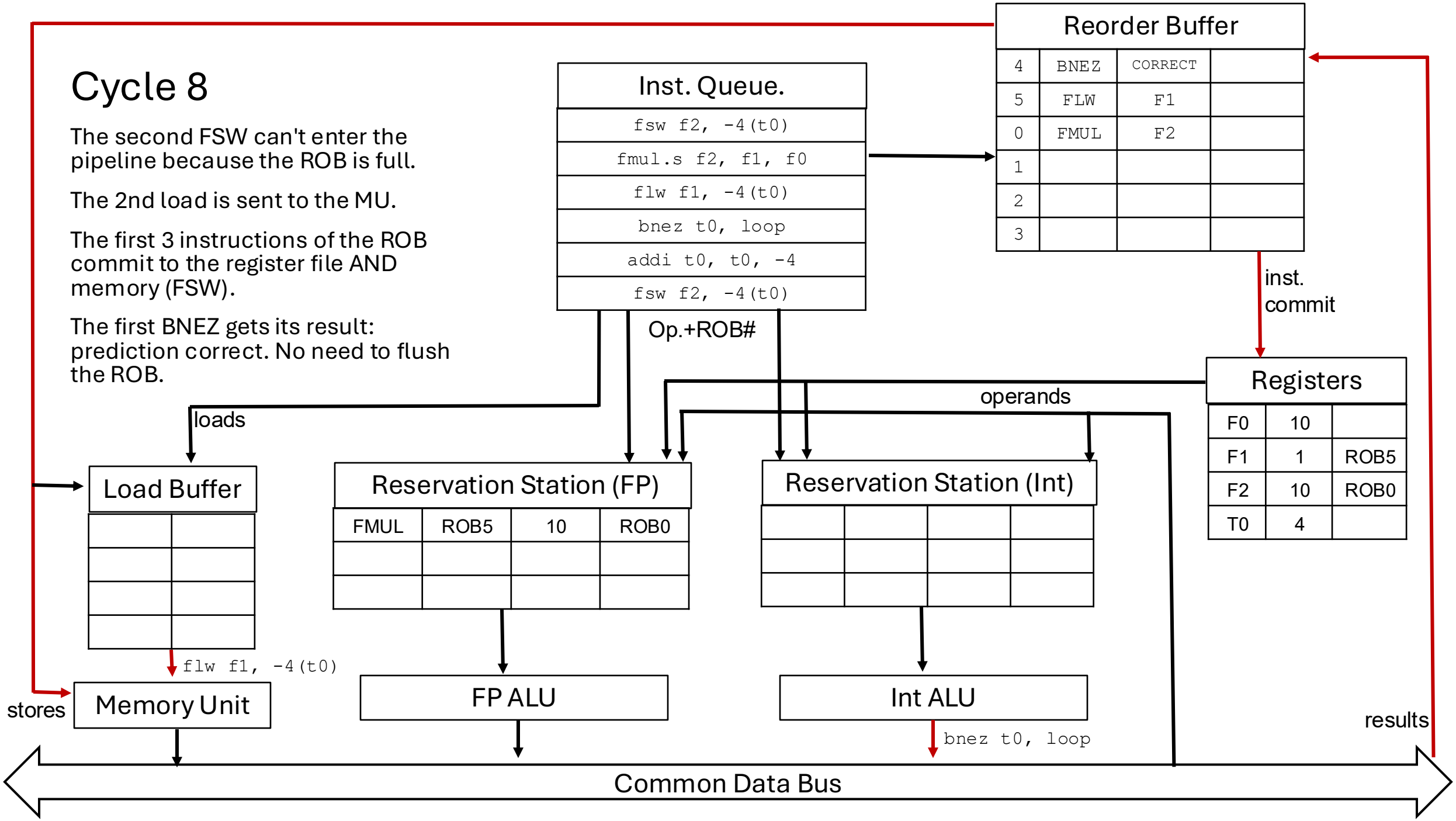
Cycle 8

The second FSW can't enter the pipeline because the ROB is full.

The 2nd load is sent to the MU.

The first 3 instructions of the ROB commit to the register file AND memory (FSW).

The first BNEZ gets its result: prediction correct. No need to flush the ROB.

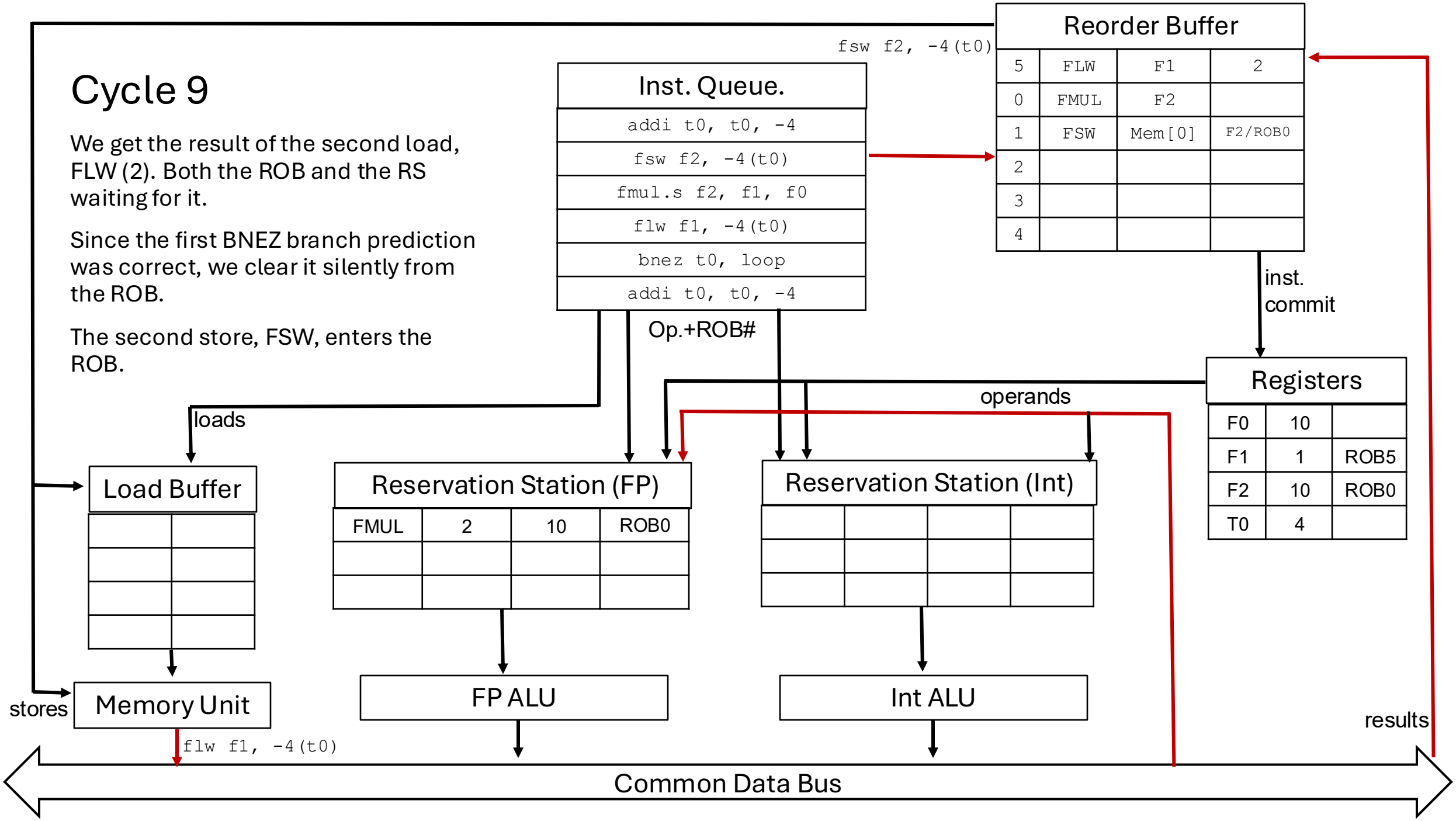


Cycle 9

We get the result of the second load, FLW (2). Both the ROB and the RS waiting for it.

Since the first BNEZ branch prediction was correct, we clear it silently from the ROB.

The second store, FSW, enters the ROB.

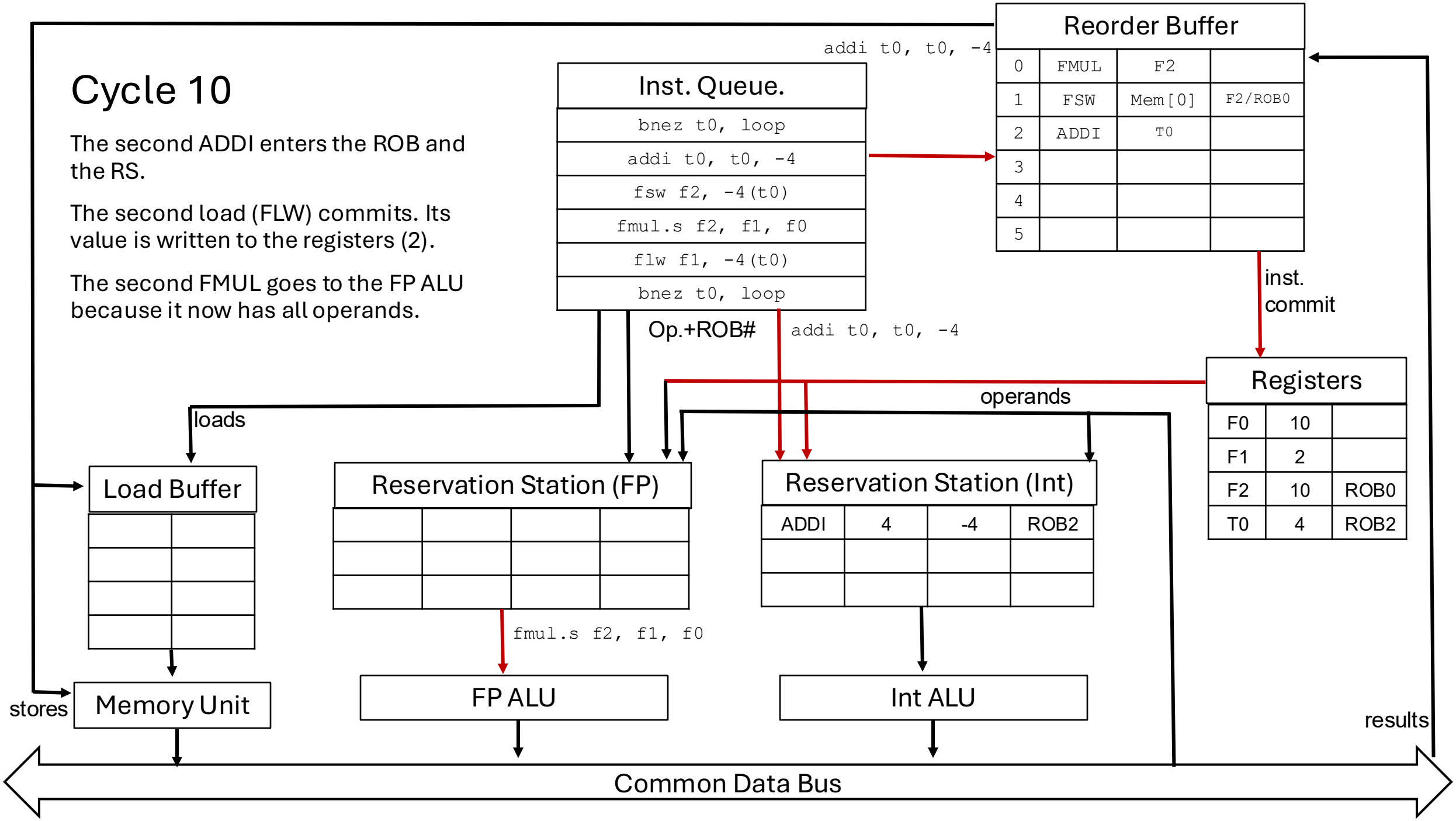


Cycle 10

The second ADDI enters the ROB and the RS.

The second load (FLW) commits. Its value is written to the registers (2).

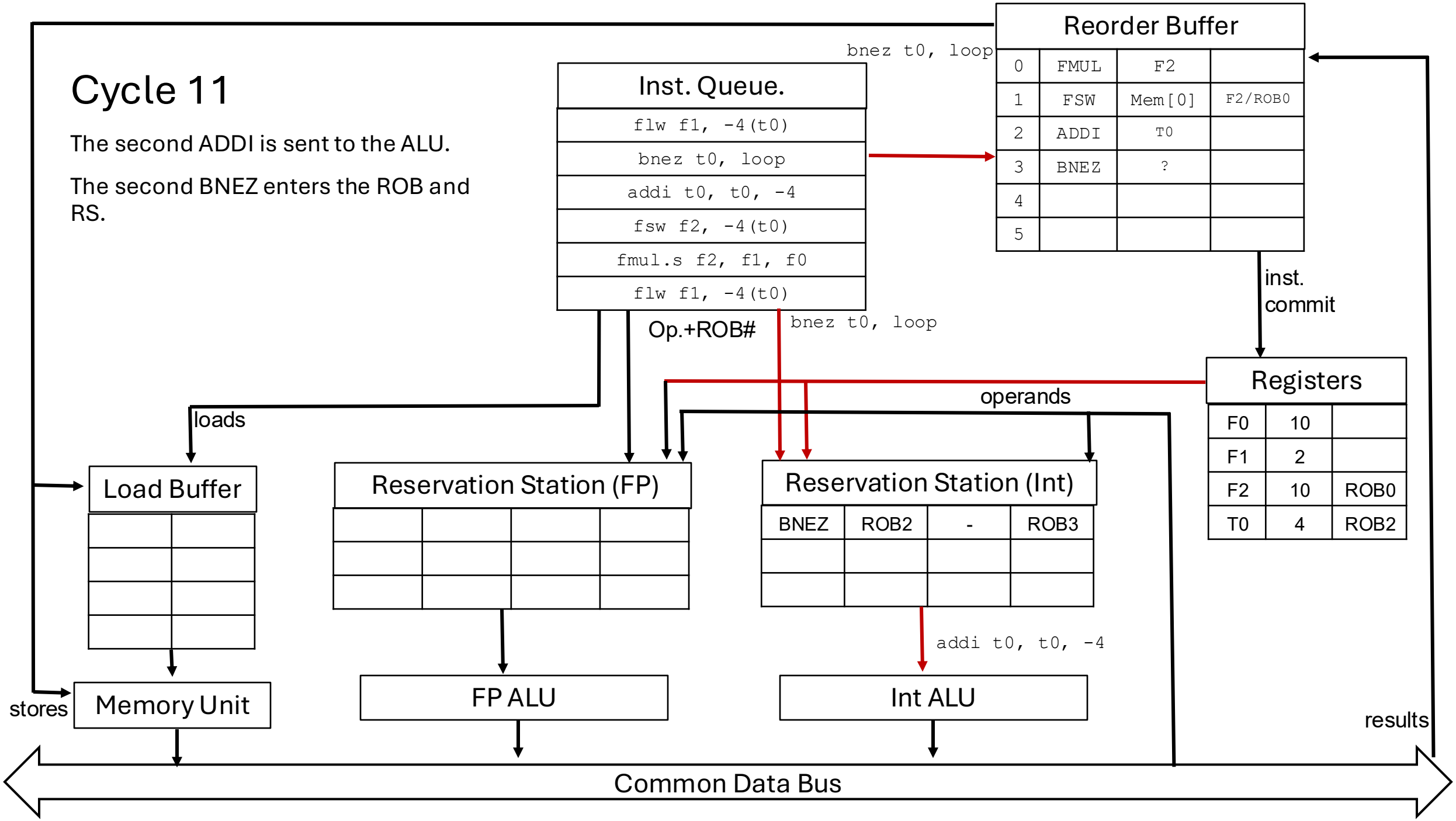
The second FMUL goes to the FP ALU because it now has all operands.



Cycle 11

The second ADDI is sent to the ALU.

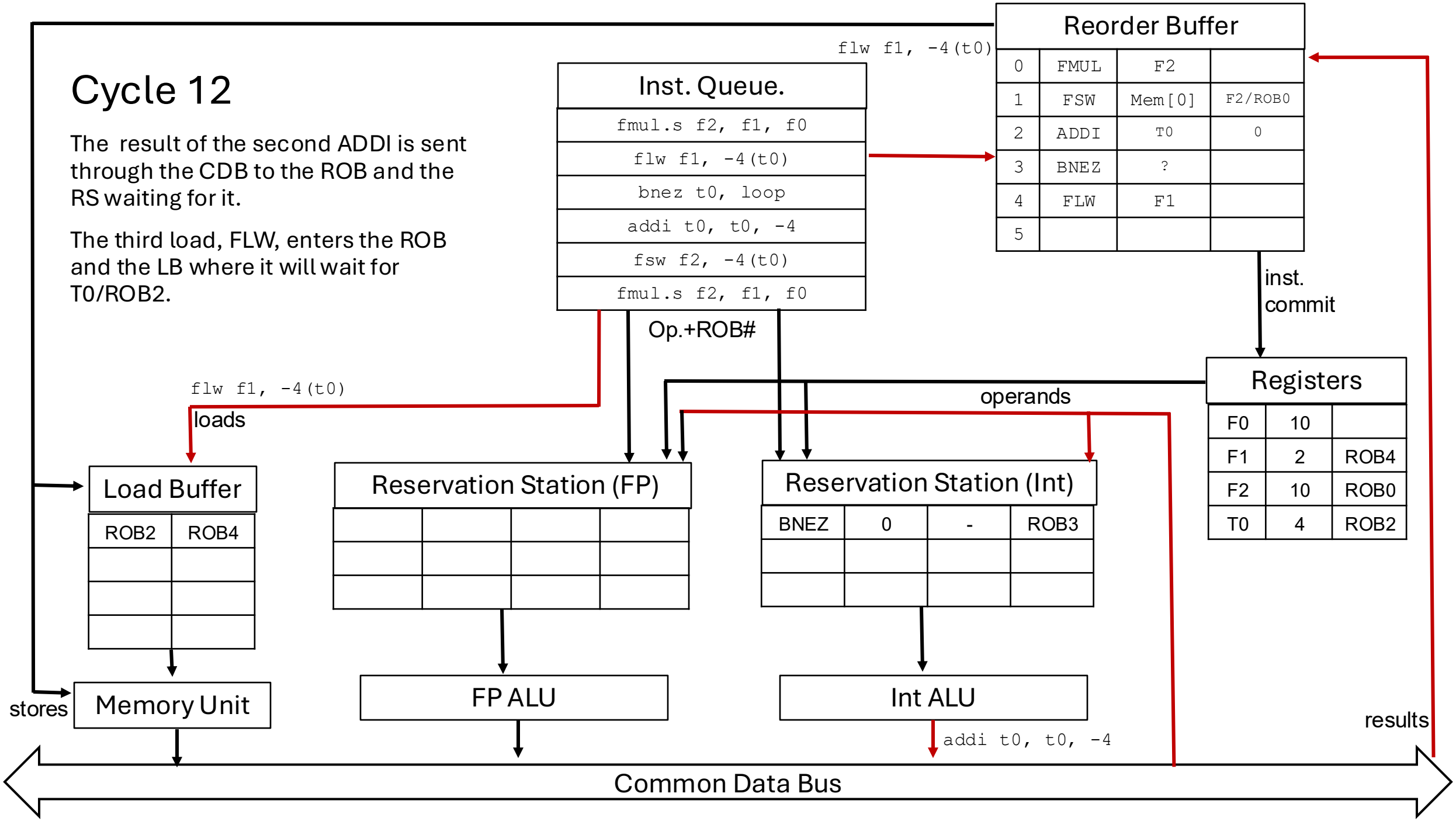
The second BNEZ enters the ROB and RS.



Cycle 12

The result of the second ADDI is sent through the CDB to the ROB and the RS waiting for it.

The third load, FLW, enters the ROB and the LB where it will wait for T0/ROB2.

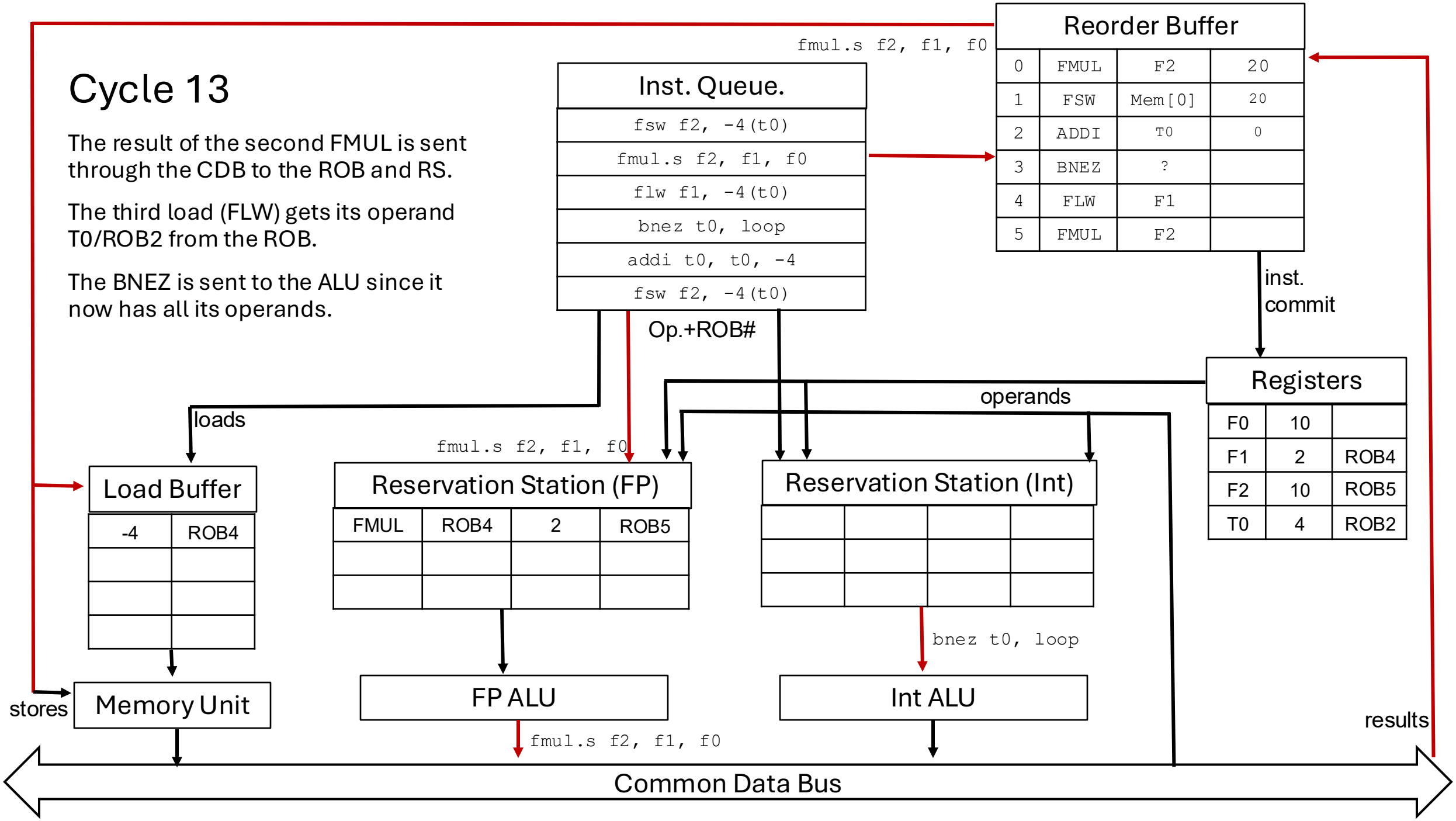


Cycle 13

The result of the second FMUL is sent through the CDB to the ROB and RS.

The third load (FLW) gets its operand T0/ROB2 from the ROB.

The BNEZ is sent to the ALU since it now has all its operands.



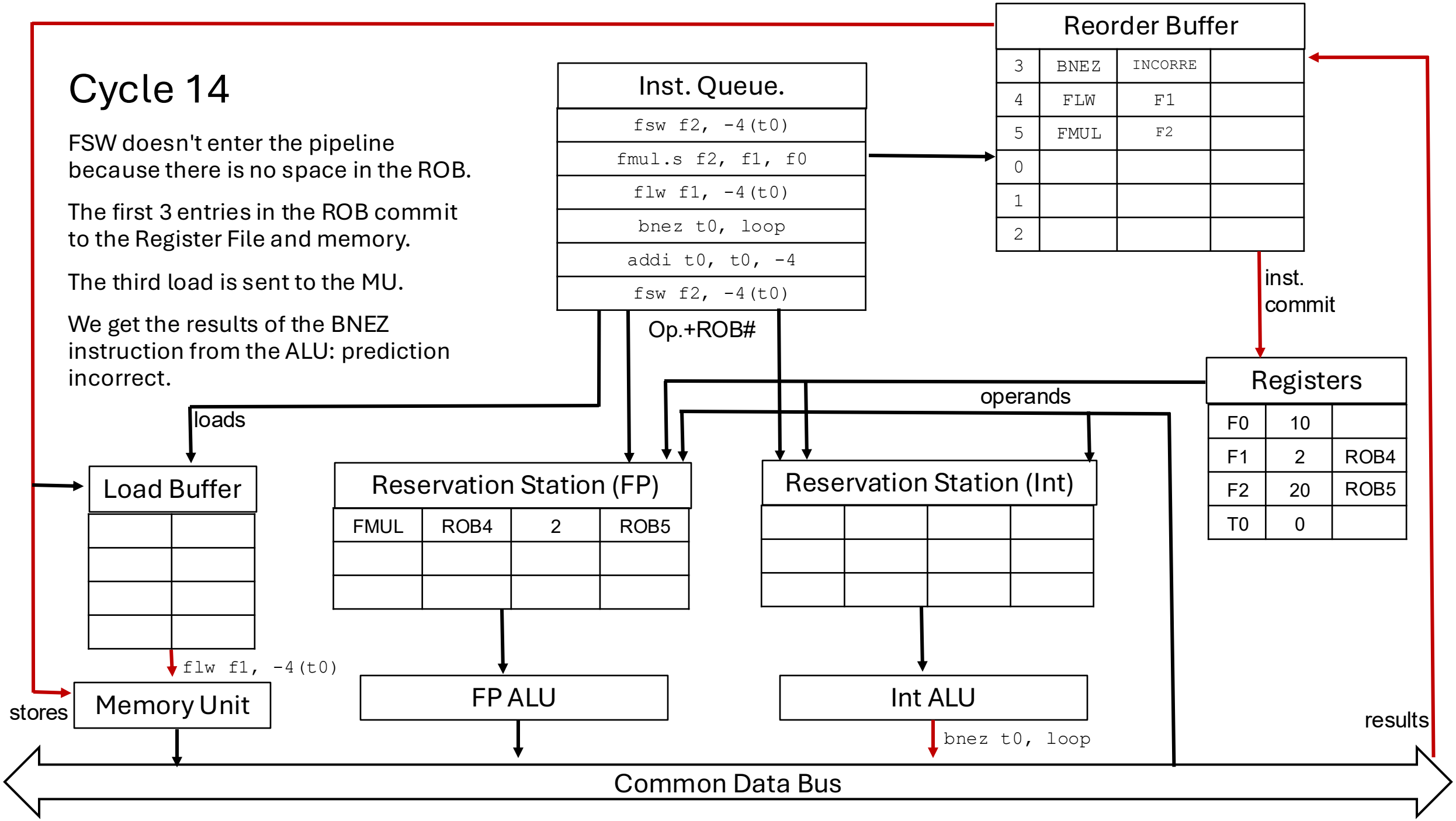
Cycle 14

FSW doesn't enter the pipeline because there is no space in the ROB.

The first 3 entries in the ROB commit to the Register File and memory.

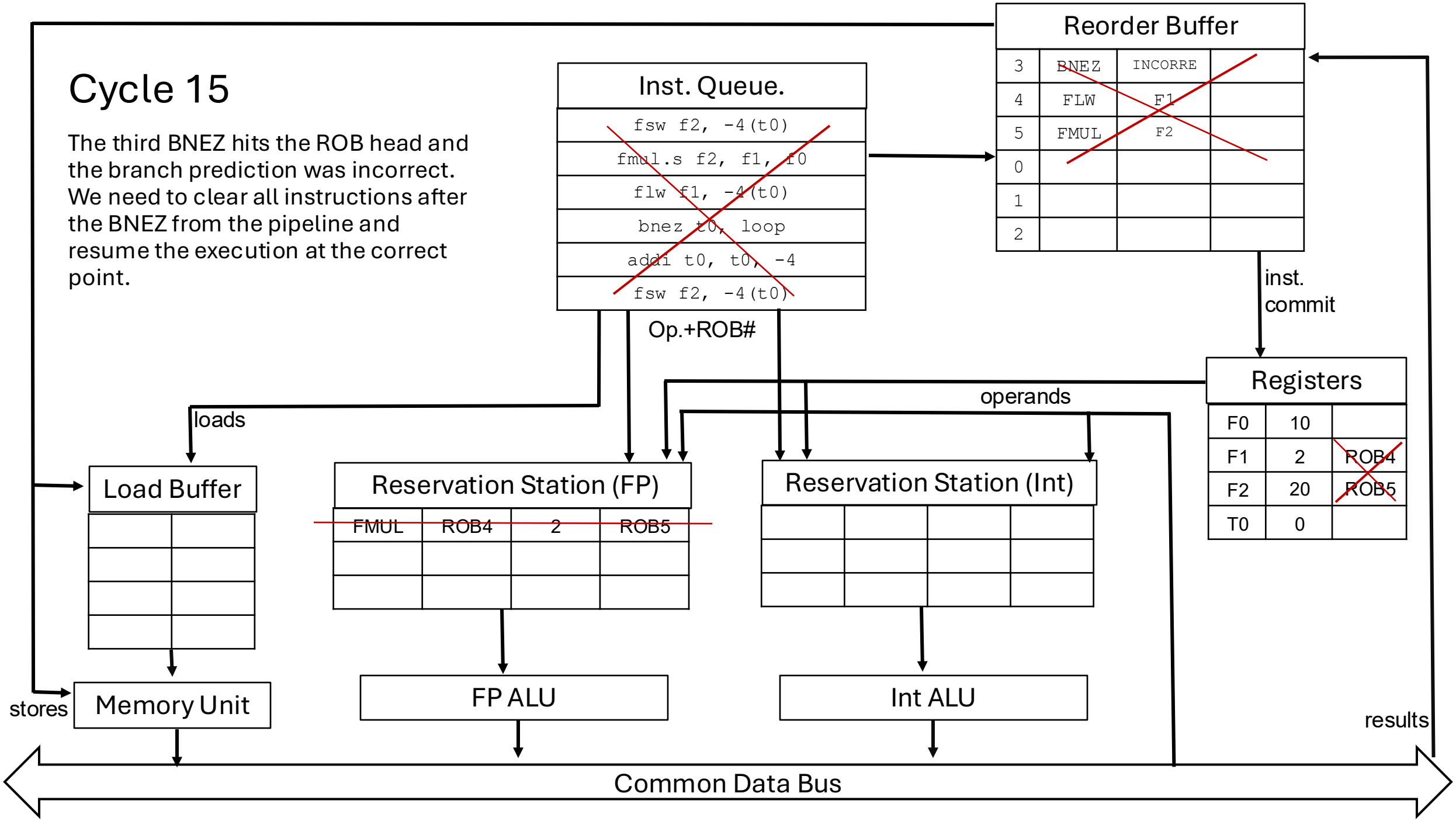
The third load is sent to the MU.

We get the results of the BNEZ instruction from the ALU: prediction incorrect.



Cycle 15

The third BNEZ hits the ROB head and the branch prediction was incorrect. We need to clear all instructions after the BNEZ from the pipeline and resume the execution at the correct point.



Cycle 16

We resume the execution at the correct point.