**Branch Prediction**

**Increment I**

1- LDM R2,0A

2- LDM R0,0

3- LDM R1,50

4- LDM R3,20

5- LDM R4,2

6- JMP R3 --Hazard1

.ORG 20

7- SUB R0,R2,R5

8- JZ R1

9- ADD R4,R4,R4 -- Hazard 2

10- OUT R4 -- Hazard 3

11- INC R0

12- JMP R3

.ORG 50

13- LDM R0,0

14- LDM R2,8

15- LDM R3,60

16- LDM R4,3

17- JMP R3 -- Hazard4

.ORG 60

18- ADD R4,R4,R4 -- Hazard5

19- OUT R4 -- Hazard6

20- INC R0

21- AND R0,R2,R5 -- Hazard7

22- JZ R3

23- INC R4 -- Hazard8

24- OUT R4 -- Hazard9

Hazard1:

Control Hazard, R3 value has not been written yet, although LDM instruction takes 2 cycles, jmp instruction is executed in fetch stage, so PC will hold a false value and false code will be execuetd

Hazard2:

Control Hazard, if branch is taken and predicted not taken it would be falsely executed.

Hazard3:

Also a control hazard, same as 2, in addition to data hazard on R4, needs 2 Nops.

Hazard4:

Exactly the Same as hazard1

Hazard5:

Data Hazard on R4, needs 1 Nop instruction.

Hazard6:

Data Hazard on R4, needs 2 Nop instructions.

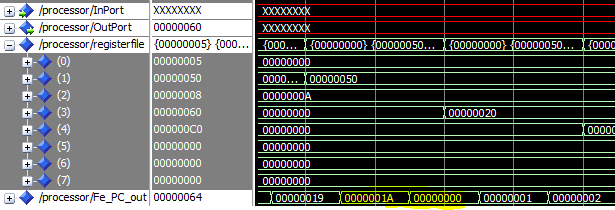
Hazard7:

Data Hazard on R0, needs 2 Nop instructions.

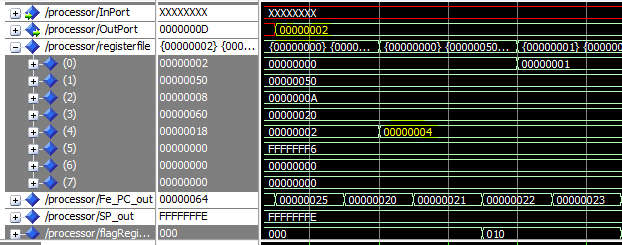
Hazards 8, 9:

Exactly as hazards 2 and 3

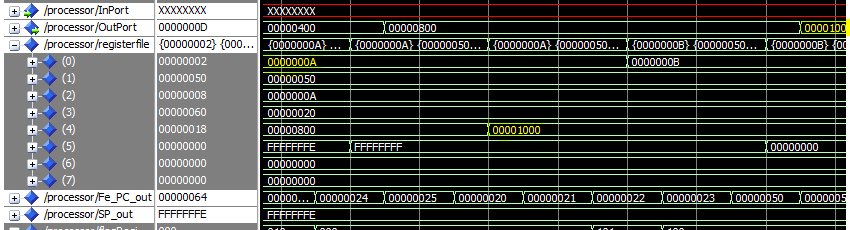
**Screenshots:**

****

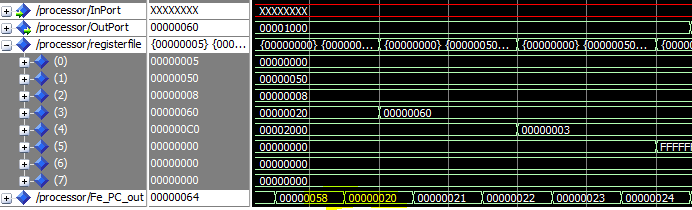
Branching was done to 0 instead of 20 (R3 value), for it has not been written then yet.



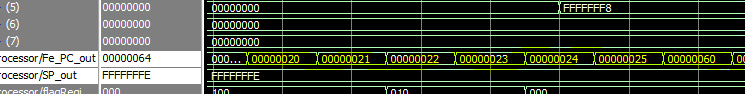
In the 1st iteration of loop 1, false value (2) of R4 was written in the output port, for the new value has not been written in R4 (Data hazard)



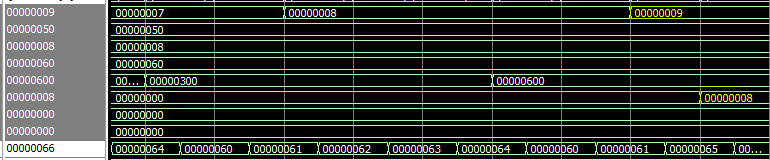
When the loop finishes(R0 = 0A), the 2 instructions, ADD&OUT below the branch instruction were falsely executed (control hazard).

****

Jmp instruction, executed falsely to address 20 instead of 60, for the value of R3 weren't written back, so 2nd loop didn't start as expected, instead, control went to loop 1 for an iteration then directed to loop2

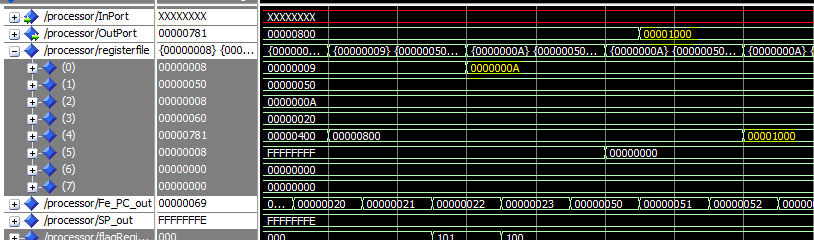


Loop 2 executed an extra iteration, for INC R0 instruction writes back after R0 is used to check the terminating condition controlled by R5, yielded 9 instead of 8 iterations as shown

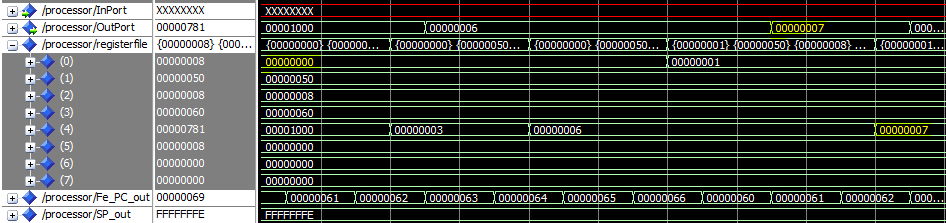


**Increment II**All data hazars are supposedly solved by the forwarding unit, most of the control ones, only hazards still exist are:

Hazard 2&3, similarly Hazard 8&9

****

When the 1st loop finishes, Add & Out instructions were executed falsely, for the flushing unit is disabled.



In the 1st iteration, branch was predicted falsely not taken, the 2 instructions below were executed falsely, yielded R4 incremented to 7 and written to the output port.

**Increment III**

The same as increment II, no hazards are to be solved by hazard detection unit

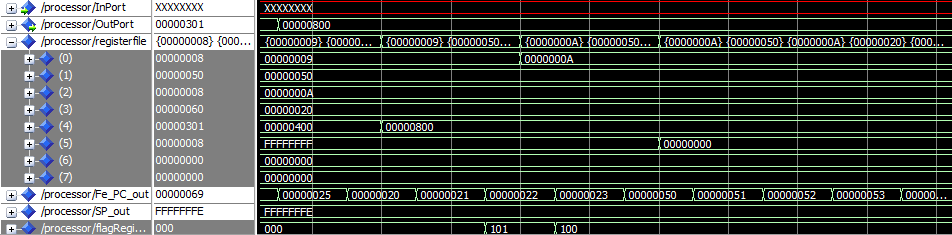
**Increment IV**

**Screenshots:**

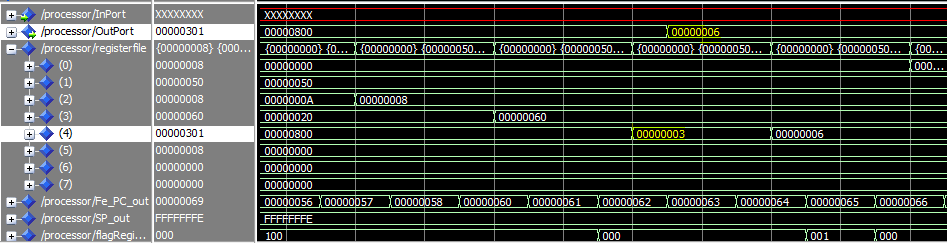
After 1st loop finished (for R0 has reached 0A), neither a new value was written in R4, nor it appeared in the output port.

Also correct values were written in R4 and in the output port before the end of the loop (400, 800).

Also Jmp instruction executed properly to Address 20.

****

At the beginning of the 2nd loop, R4 was read correctly (3), so the result of R4 \* 2 was written in the output port correctly (6).



In the last iteration of 2nd loop, when R0 became 8, correct value was directly forwarded in the AND instruction, yielded 8 in R5 so the loop broke, R4 was incremented and put in the output port as shown.

