Name: Duc Anh Nguyen

Course: EE126

Date: 10.24.2022

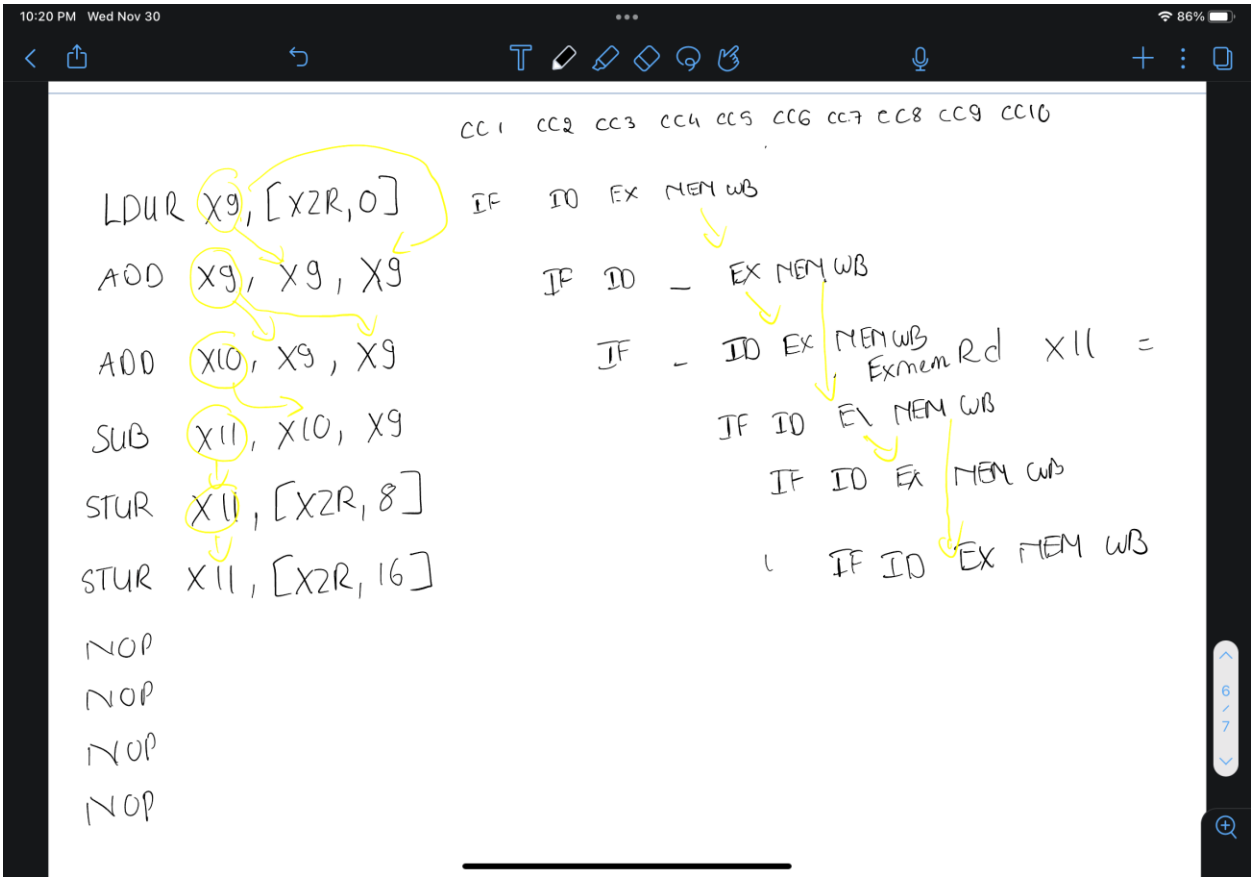
Lab Report 5

I. Introduction:

- The goal of this lab is to overcome data hazard using forwarding and stalling. The main implementation is to install the control unit and the hazard detection unit.

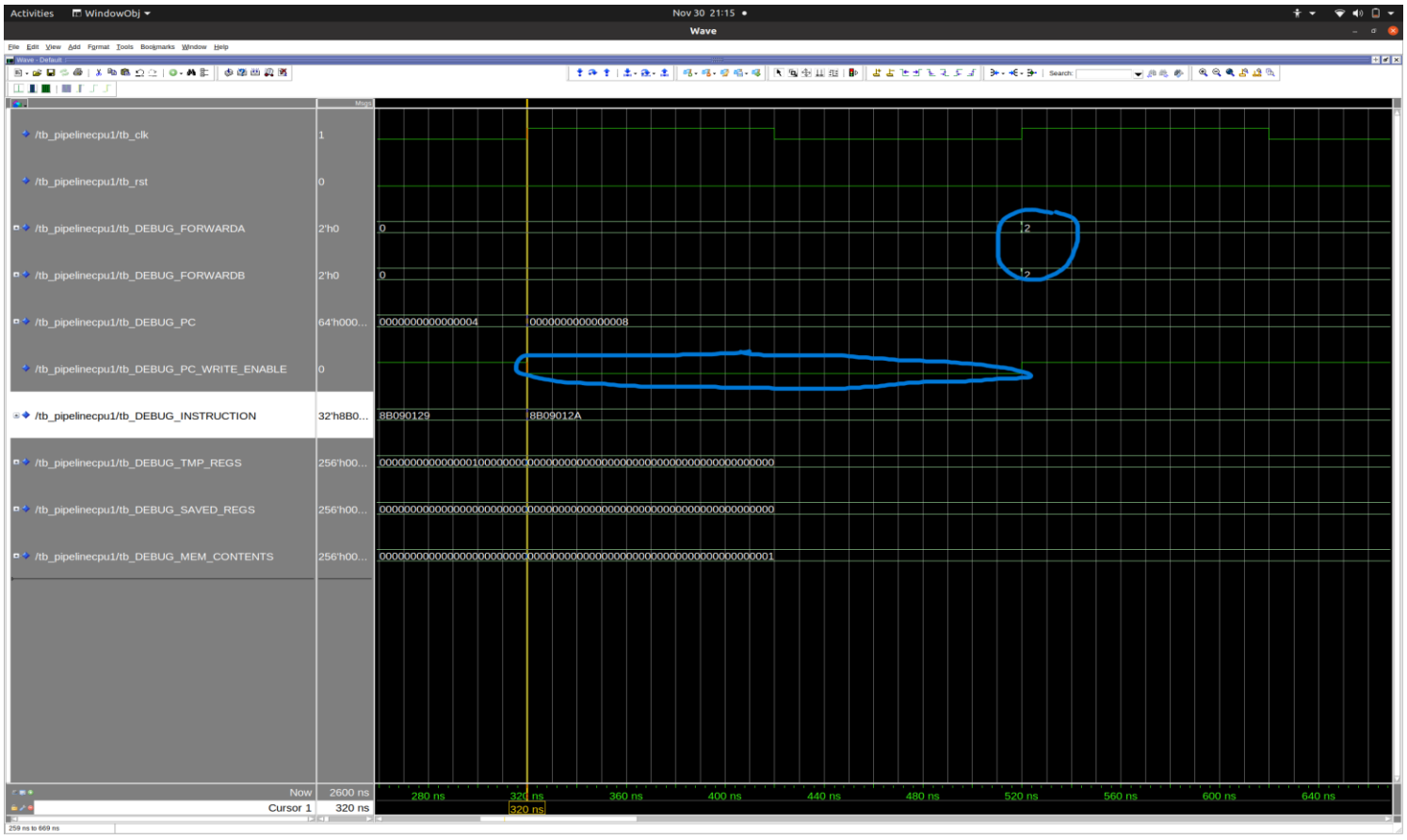
II. Procedure:

Figure 1:



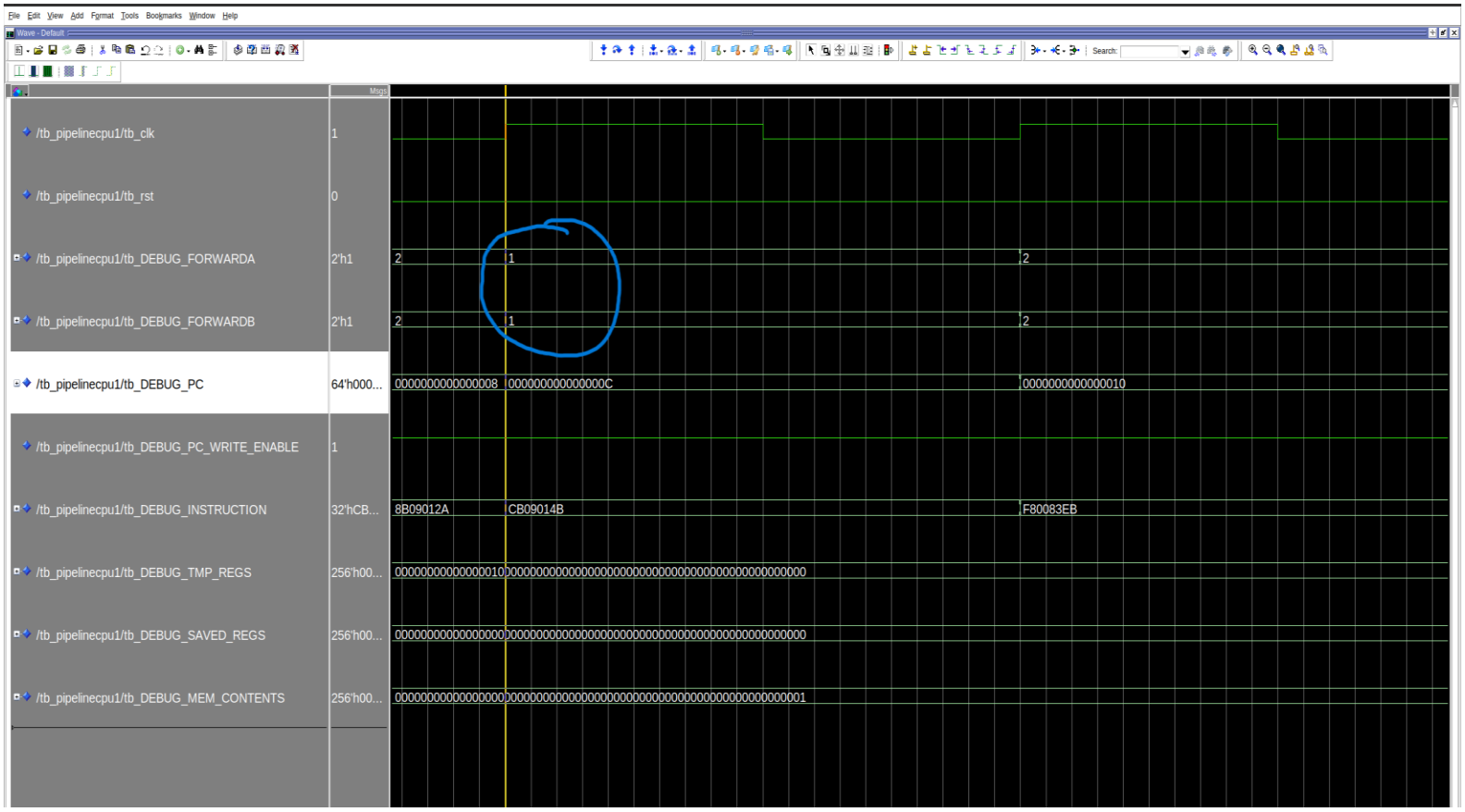
- Figure 1 is a hand-written figure of how the pipeline should look through each stage
- The first data hazard would be instruction 1 and 2 with the dependencies on register X9
- To overcome this, we would need to implement stalling and forwarding at the same time
- In figure 1, we can see that the pipeline is stall by 1 cycle with a bubble add to other pipeline register.

Figure 2:



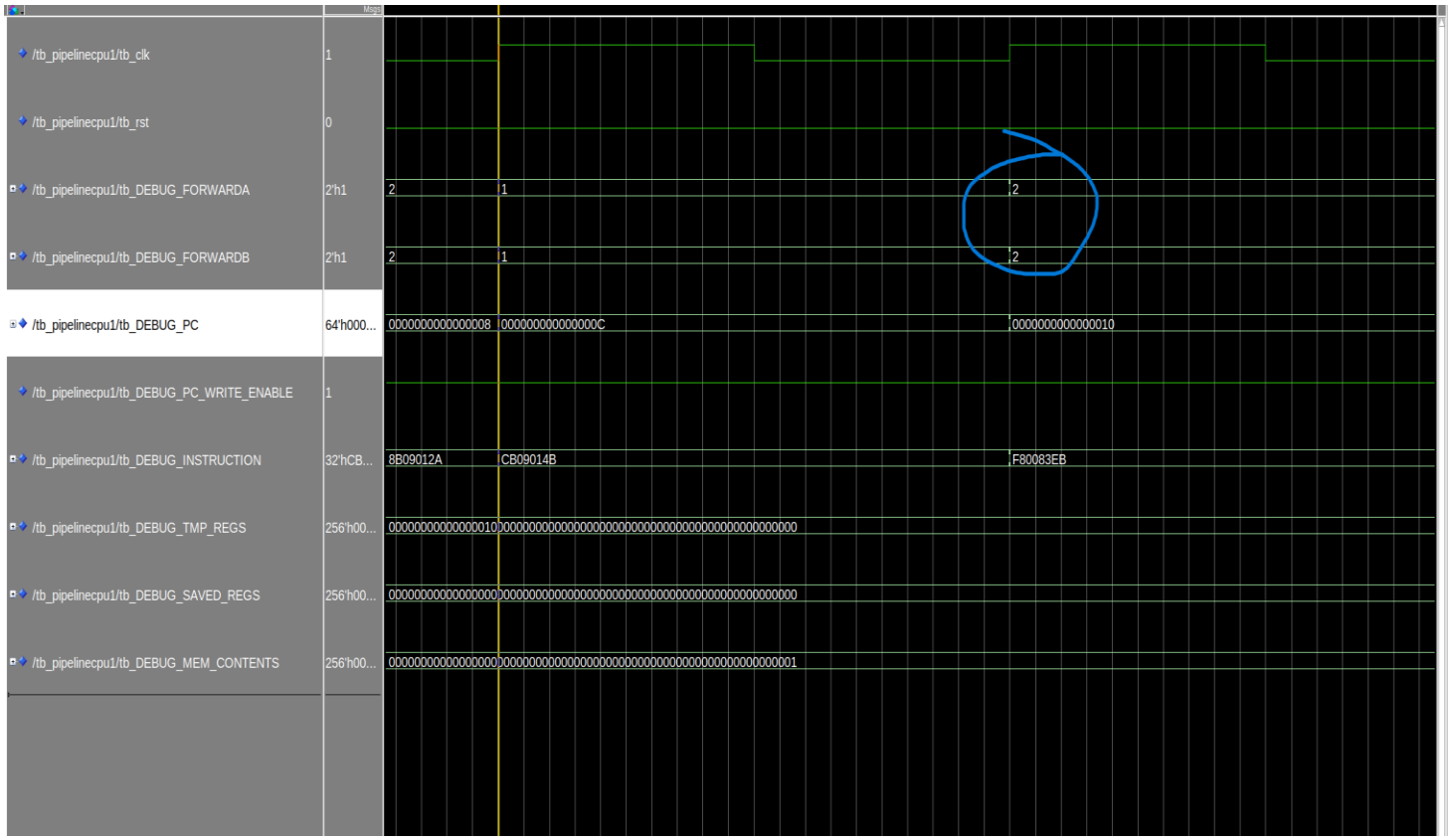
- In Figure 2: the large blue markup is at cycle 4, and it send a signal to the PC and IFID pipeline register to stop writing on clock cycle and send nop instruction to another pipeline. That's why it is low
- In the small markup, after stalling for one cycle, the forwarding unit start forwarding both Rn and Rm of the ALU for instruction 2.

Figure 3:



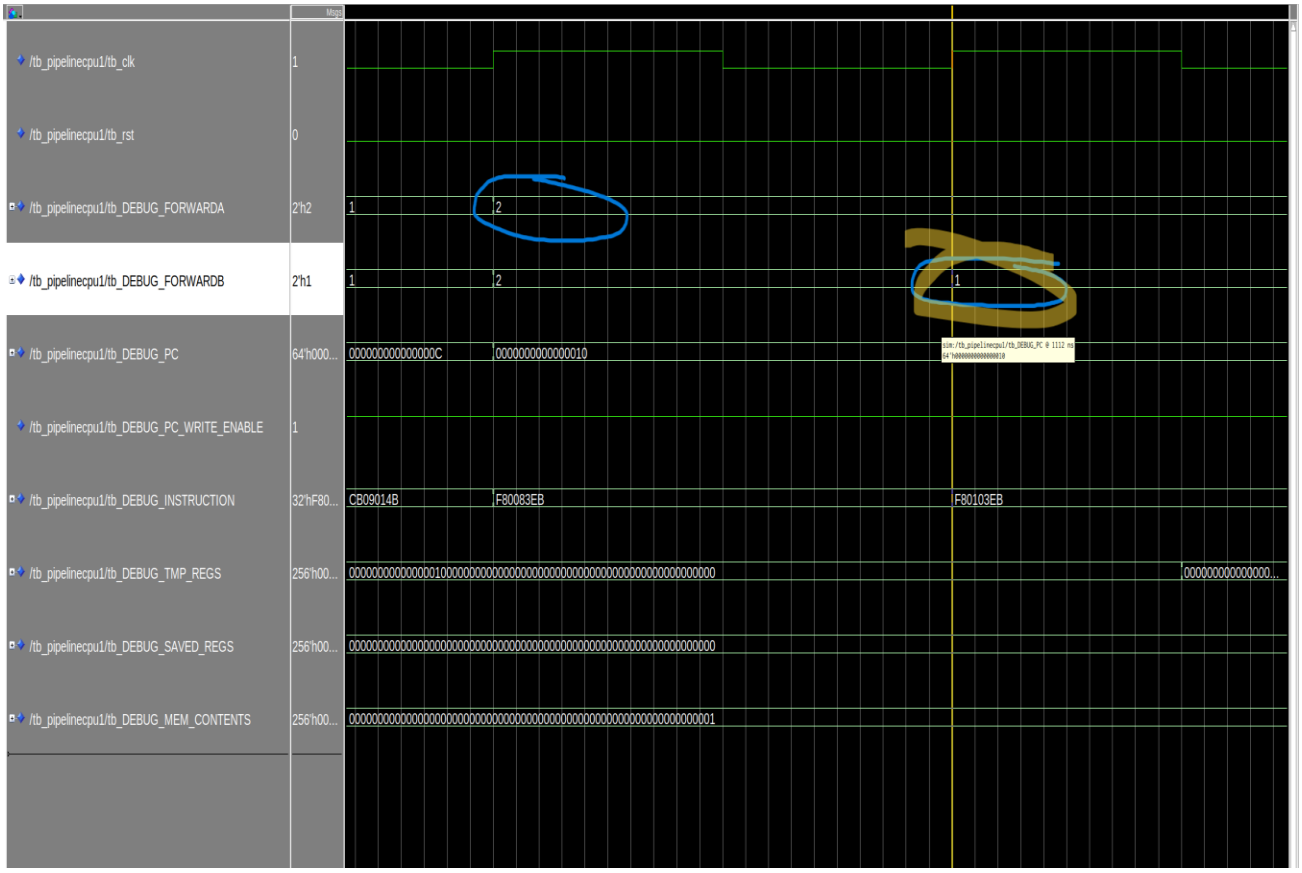
- In cycle 5, the new data of X9 have to be moved from the MEMWB register, that's why the value of ForwardA and ForwardB are 1.

Figure 4:



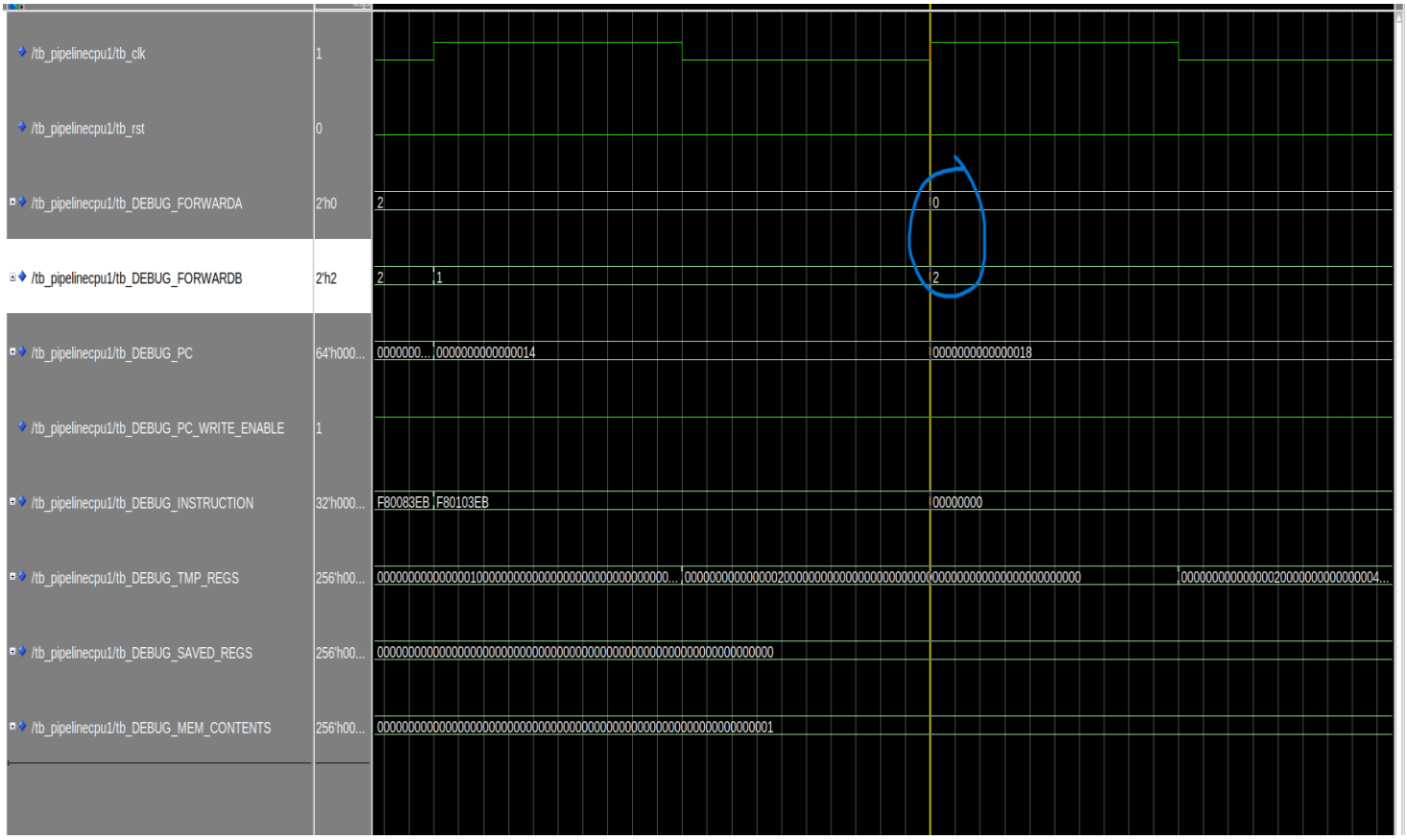
- Similarly, in cycle 5, the data will be transferred from the EXMEM register back to the ALU

Figure 5:



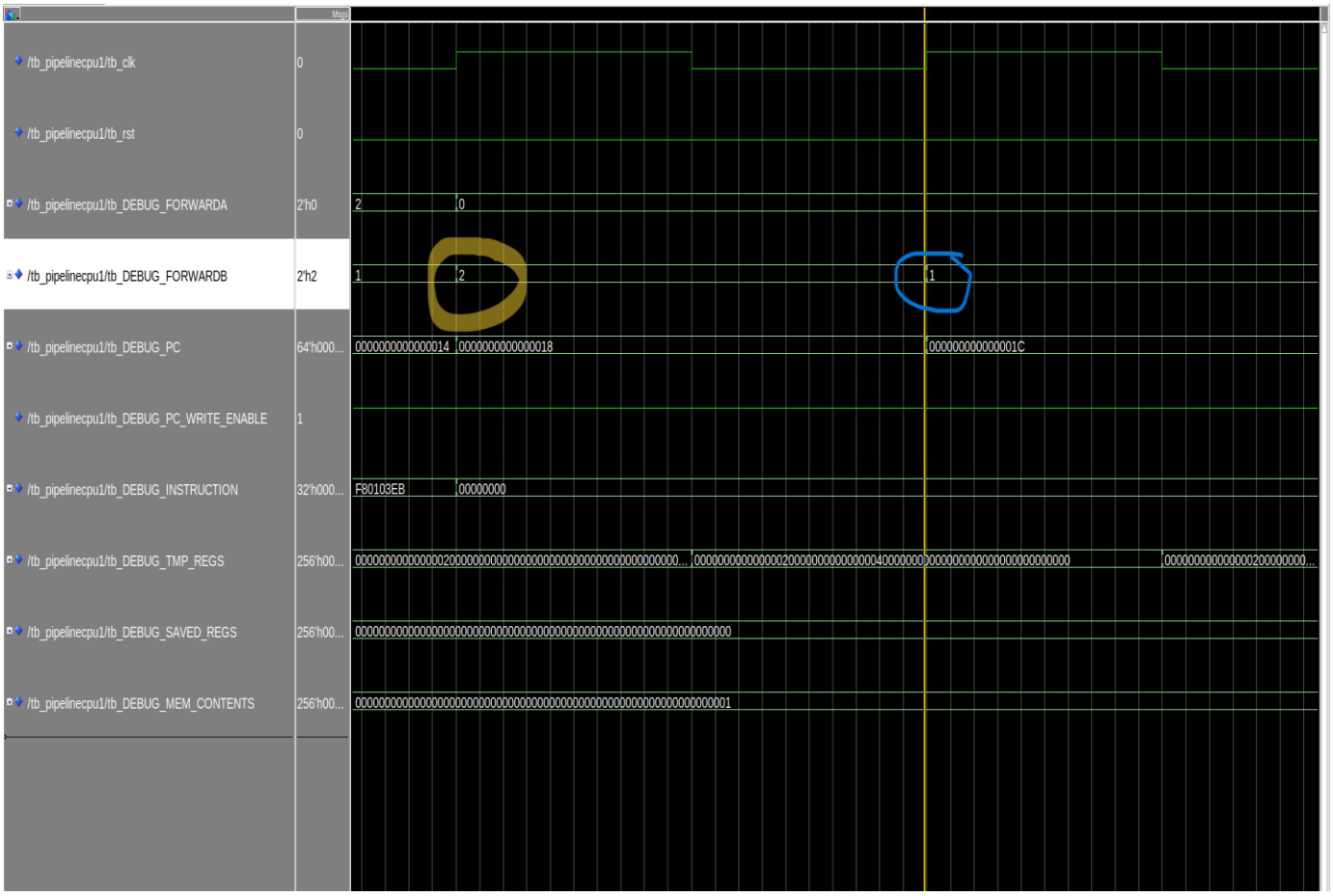
- In cycle 7, because the data of Rm has to be transferred from the MEM register to the ALU, the forward value (Yellow markup) is 1, whereas data of Rn can be transferred from the EX stage to the ALU, so the forward value (Blue markup) is 2.

Figure 6:



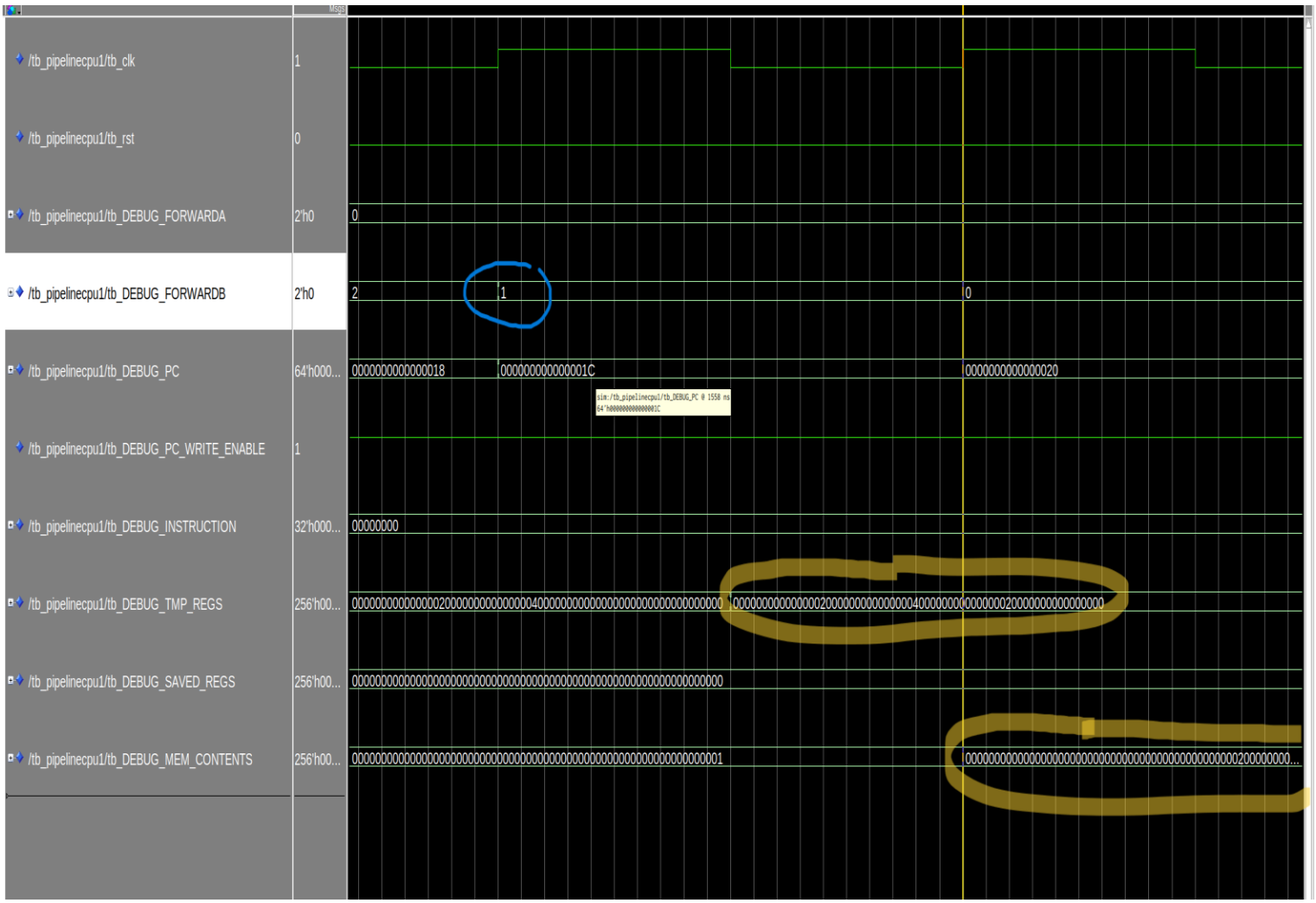
- For the STUR instruction, Rn does not have to forward, that's why ForwardA is 0. However, Rm has to be forwarded from the EX stage.
- You can also see below that the value of X9 has been updated to 2

Figure 7:



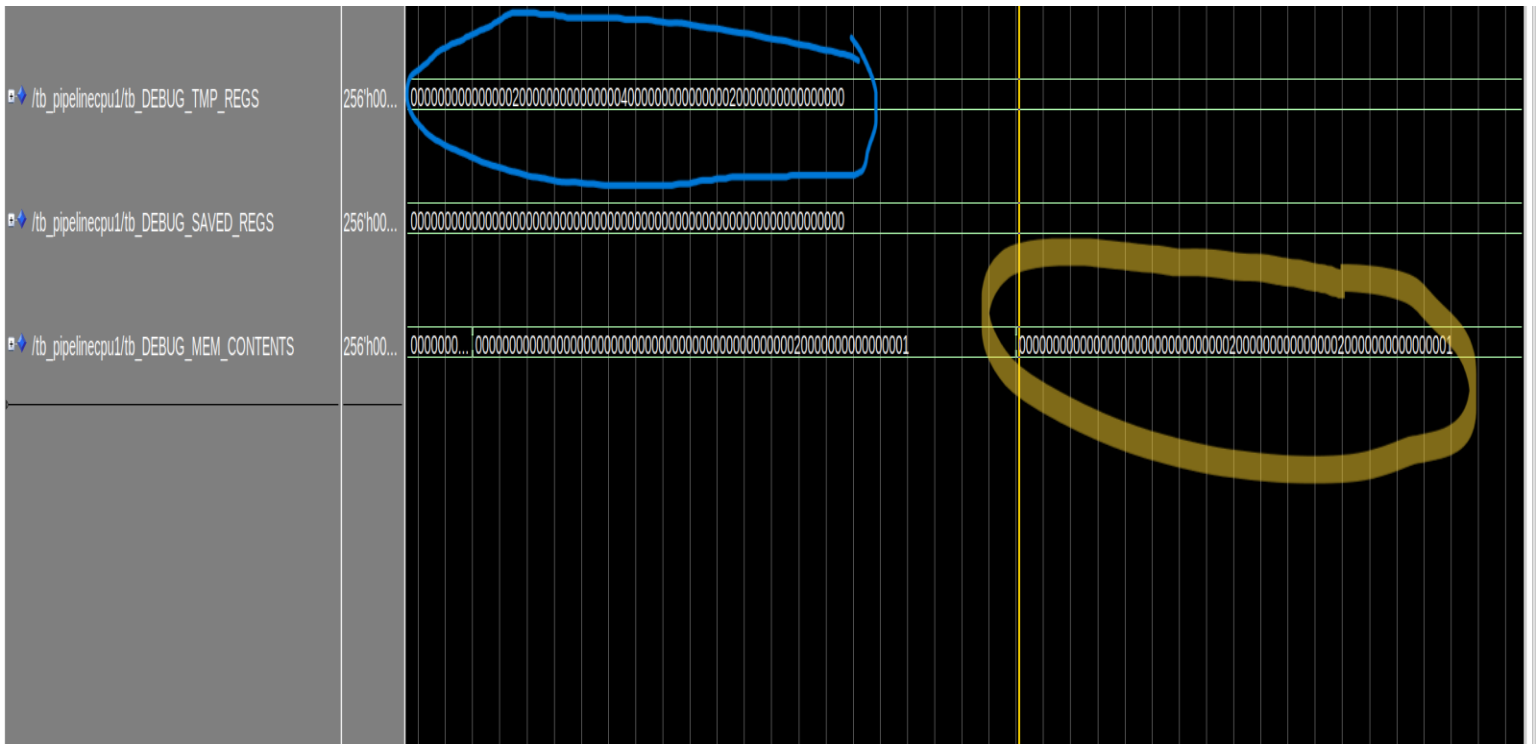
- For the last STUR instruction, Rm has to be moved from the MEM staged, so forward has to be 1.

Figure 8:



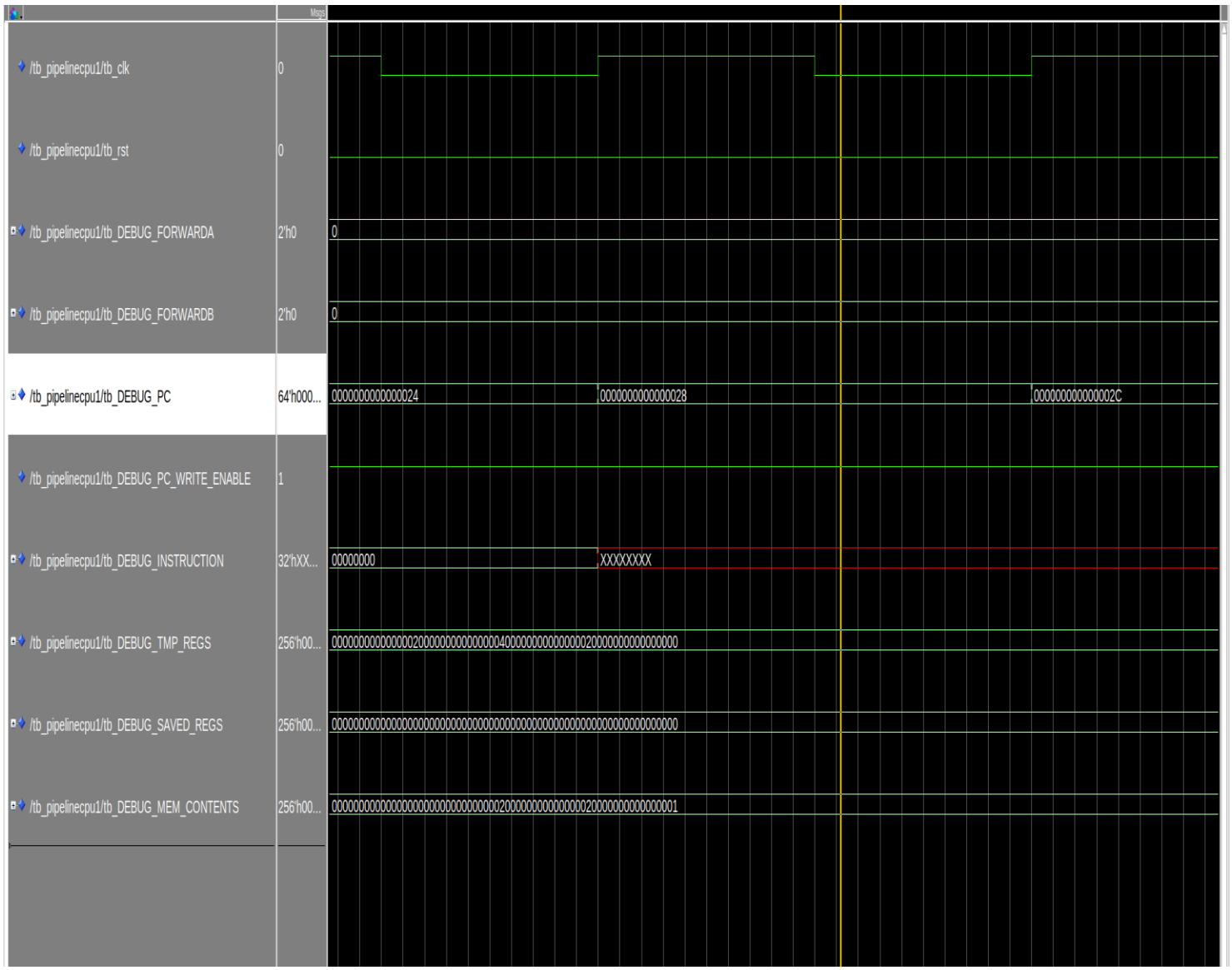
- After cycle 10, the forwarding unit would receive a NOP instruction, so that's why both ForwardA and ForwardB has become 0.
- The yellow markup is the final value of all the temporary registers.

Figure 9:



- This is the final value of Memory and Registers during the cycle 10,
- The rest should be NOP with no instruction in figure 10.

Figure 10:


/tb_pipelinecpu1/tb_clk
0
/tb_pipelinecpu1/tb_rst
0
/tb_pipelinecpu1/tb_DEBUG_FORWARDA
2'h0
0
/tb_pipelinecpu1/tb_DEBUG_FORWARDB
2'h0
0
/tb_pipelinecpu1/tb_DEBUG_PC
64'h000...
0000000000000024
0000000000000028
000000000000002C
/tb_pipelinecpu1/tb_DEBUG_PC_WRITE_ENABLE
1
/tb_pipelinecpu1/tb_DEBUG_INSTRUCTION
32'hXX...
00000000
XXXXXXXX
/tb_pipelinecpu1/tb_DEBUG_TMP_REGS
256'h00...
0000000000000002000000000000000400000000000000020000000000000000
/tb_pipelinecpu1/tb_DEBUG_SAVED_REGS
256'h00...
0000000000000000000000000000000000000000000000000000000000000000
/tb_pipelinecpu1/tb_DEBUG_MEM_CONTENTS
256'h00...
0000000000000000000000000000000200000000000000020000000000000001