CprE 381: Computer Organization and Assembly-Level Programming

Project Part 2 Report

Team Members: Varun Jain

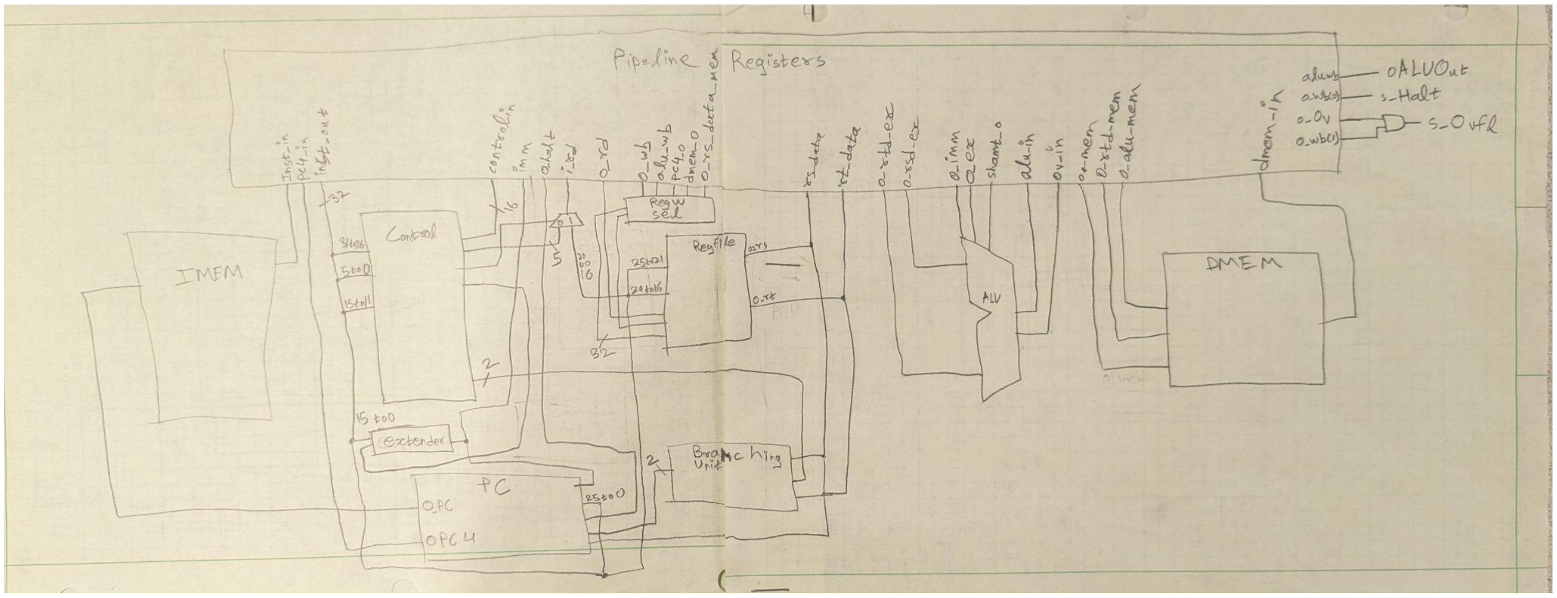
Project Teams Group #: Proj2_4_2

Refer to the highlighted language in the project 1 instruction for the context of the following questions.

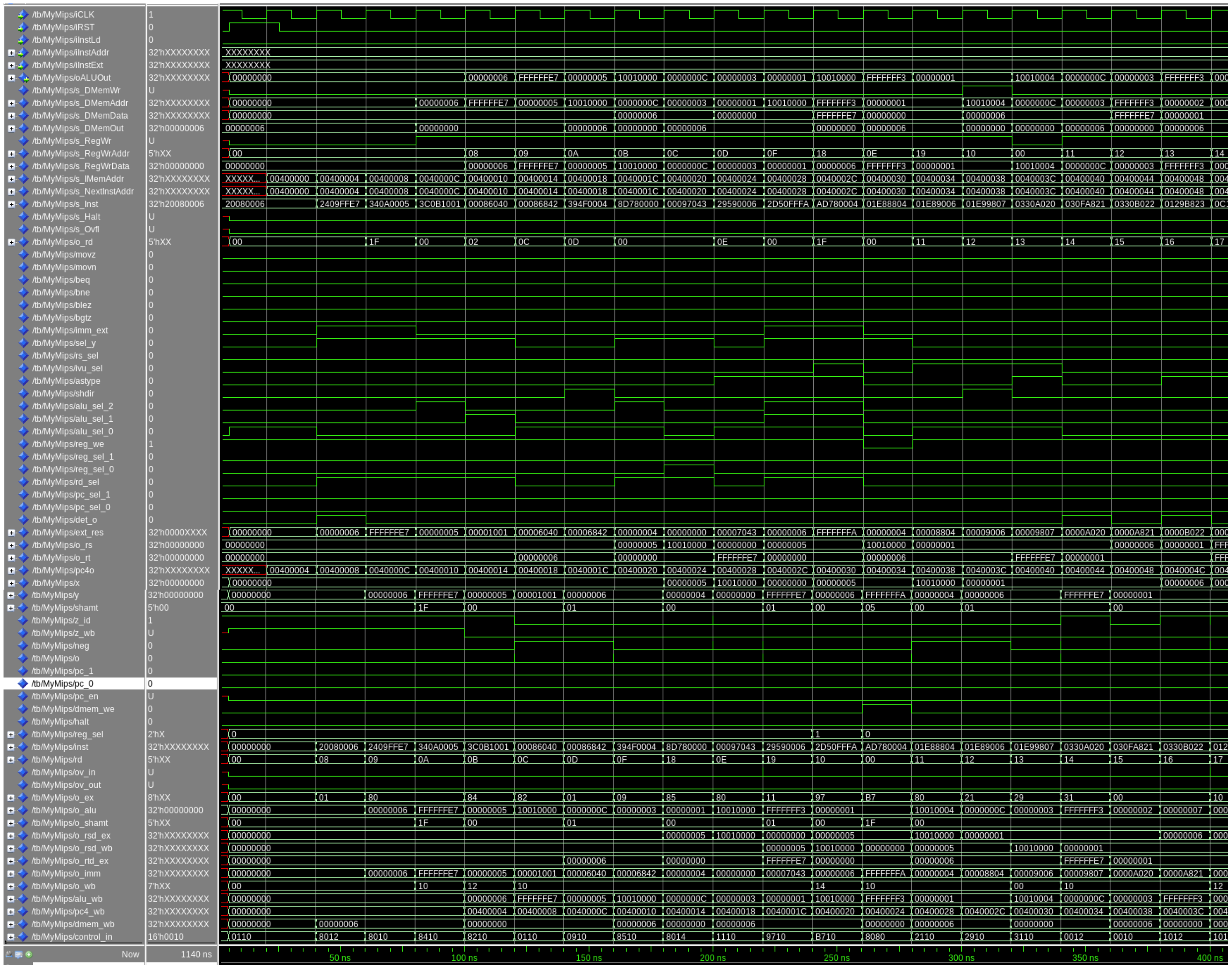
[1.a] Come up with a global list of the datapath values and control signals that are required during each pipeline stage.

IF/ID: IFIDwe as input, output instruction
ID/EX: WE, control and other values input. Control and other values output. Also has jump and branch evaluation
EX/MEM: WE, control and other values input. Control and other values output.
MEM/WB: WE, control and other values input. Control and other values output.

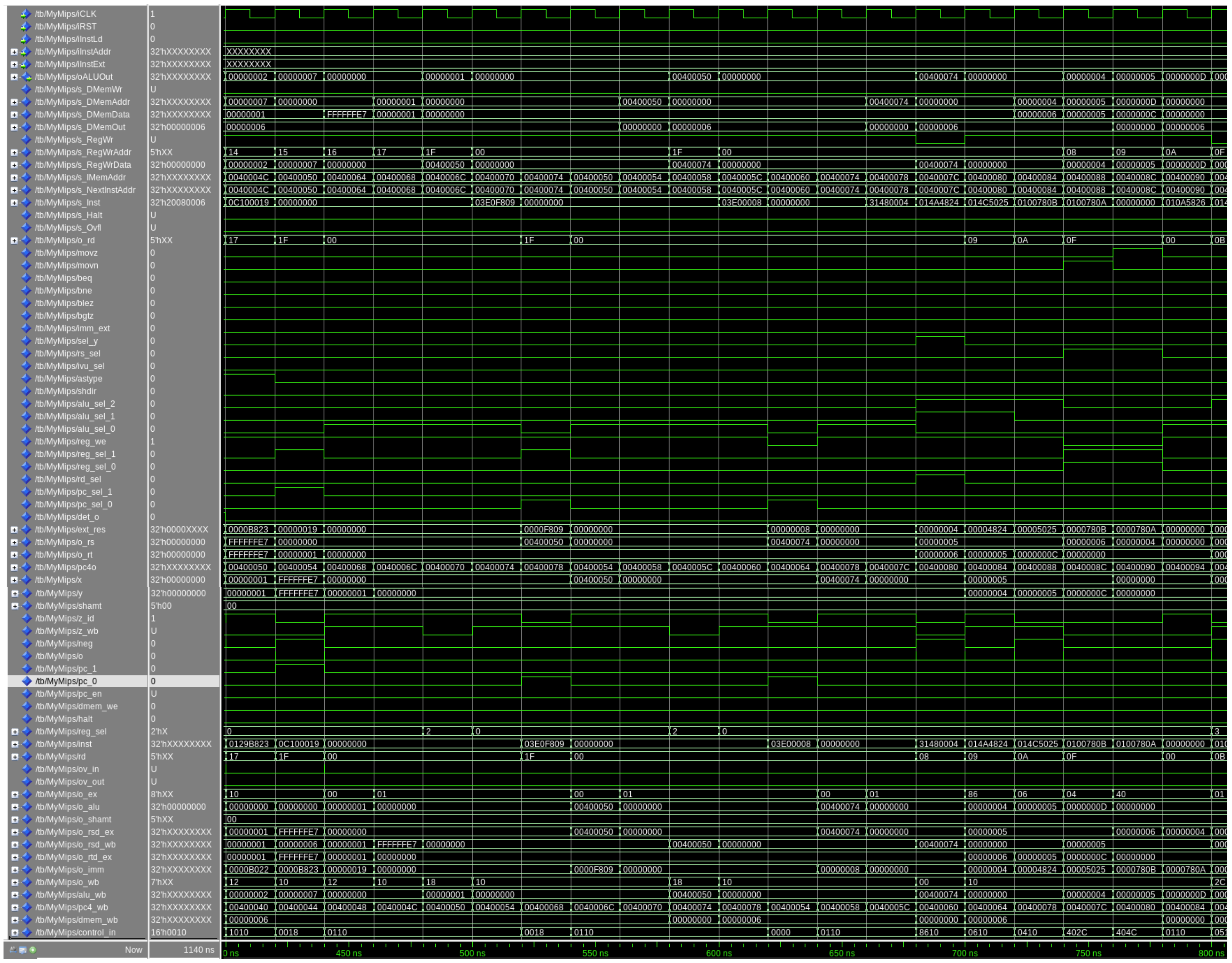
[1.b.ii] high-level schematic drawing of the interconnection between components.



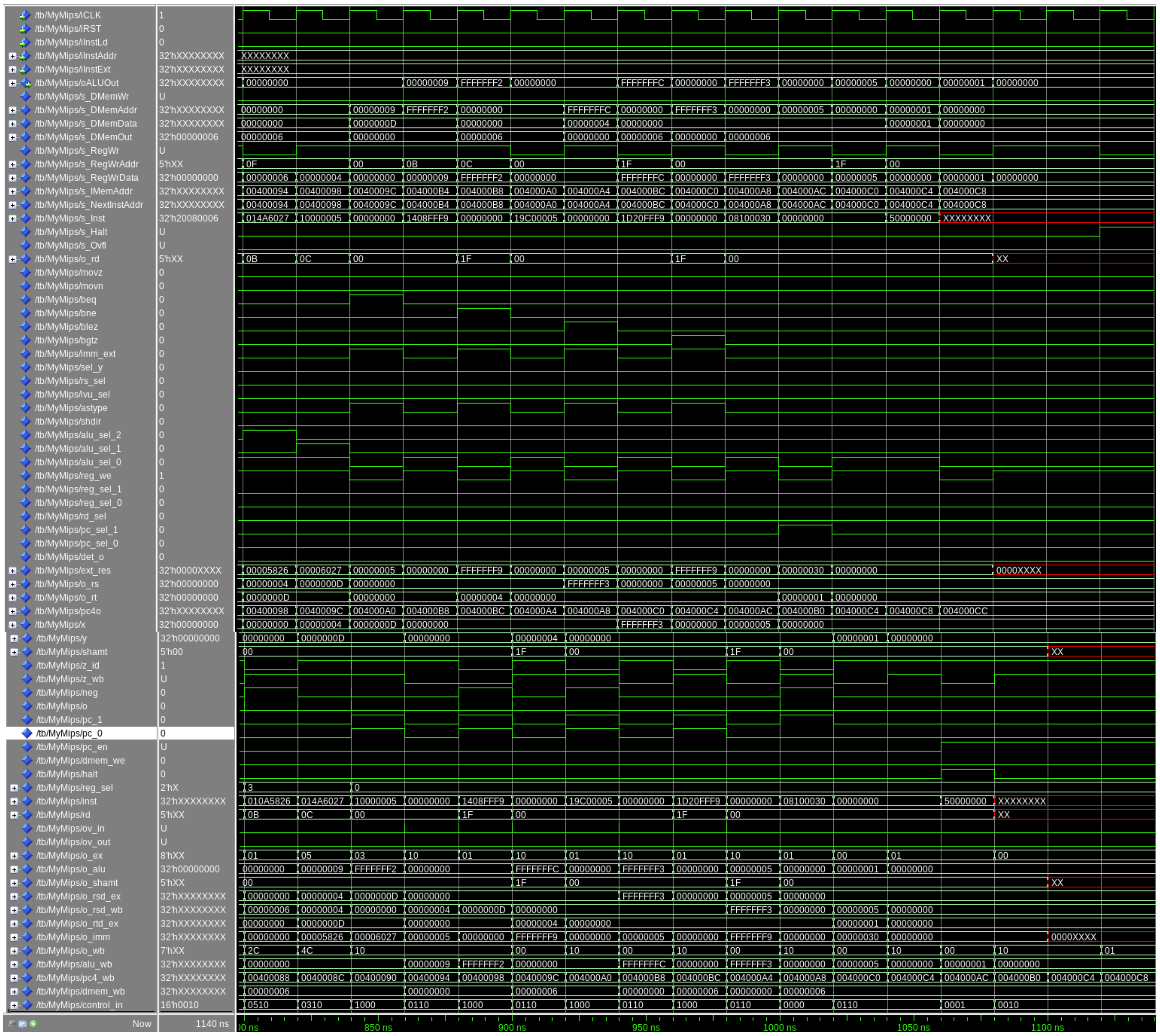
[1.c.i] include an annotated waveform in your writeup and provide a short discussion of result correctness.



/tb/MyMips/iCLK	1
/tb/MyMips/iRST	0
/tb/MyMips/iInstLd	0
/tb/MyMips/iInstAddr	32'hXXXXXXXX
/tb/MyMips/iInstExt	32'hXXXXXXXX
/tb/MyMips/oALUOut	32'hXXXXXXXX
/tb/MyMips/s_DMemWr	U
/tb/MyMips/s_DMemAddr	32'hXXXXXXXX
/tb/MyMips/s_DMemData	32'hXXXXXXXX
/tb/MyMips/s_DMemOut	32'h00000006
/tb/MyMips/s_RegWr	U
/tb/MyMips/s_RegWrAddr	5'hXX
/tb/MyMips/s_RegWrData	32'h00000000
/tb/MyMips/s_IMemAddr	32'hXXXXXXXX
/tb/MyMips/s_NextInstAddr	32'hXXXXXXXX
/tb/MyMips/s_Inst	32'h20080006
/tb/MyMips/s_Halt	U
/tb/MyMips/s_Ovfl	U
/tb/MyMips/o_rd	5'hXX
/tb/MyMips/movz	0
/tb/MyMips/movn	0
/tb/MyMips/beq	0
/tb/MyMips/bne	0
/tb/MyMips/blez	0
/tb/MyMips/bgtz	0
/tb/MyMips/imm_ext	0
/tb/MyMips/sel_y	0
/tb/MyMips/rs_sel	0
/tb/MyMips/ivu_sel	0
/tb/MyMips/astype	0
/tb/MyMips/shdir	0
/tb/MyMips/alu_sel_2	0
/tb/MyMips/alu_sel_1	0
/tb/MyMips/alu_sel_0	0
/tb/MyMips/reg_we	1
/tb/MyMips/reg_sel_1	0
/tb/MyMips/reg_sel_0	0
/tb/MyMips/rd_sel	0
/tb/MyMips/pc_sel_1	0
/tb/MyMips/pc_sel_0	0
/tb/MyMips/det_o	0
/tb/MyMips/ext_res	32'h0000XXXX
/tb/MyMips/o_rs	32'h00000000
/tb/MyMips/o_rt	32'h00000000
/tb/MyMips/pc4o	32'hXXXXXXXX
/tb/MyMips/x	32'h00000000
/tb/MyMips/y	32'h00000000
/tb/MyMips/shamt	5'h00
/tb/MyMips/z_id	1
/tb/MyMips/z_wb	U
/tb/MyMips/neg	0
/tb/MyMips/o	0
/tb/MyMips/pc_1	0
/tb/MyMips/pc_0	0
/tb/MyMips/pc_en	U
/tb/MyMips/dmem_we	0
/tb/MyMips/halt	0
/tb/MyMips/reg_sel	2'hX
/tb/MyMips/inst	32'hXXXXXXXX
/tb/MyMips/rd	5'hXX
/tb/MyMips/ov_in	U
/tb/MyMips/ov_out	U
/tb/MyMips/o_ex	8'hXX
/tb/MyMips/o_alu	32'h00000000
/tb/MyMips/o_shamt	5'hXX
/tb/MyMips/o_rsd_ex	32'hXXXXXXXX
/tb/MyMips/o_rsd_wb	32'hXXXXXXXX
/tb/MyMips/o_rtd_ex	32'hXXXXXXXX
/tb/MyMips/o_imm	32'hXXXXXXXX
/tb/MyMips/o_wb	7'hXX
/tb/MyMips/alu_wb	32'hXXXXXXXX
/tb/MyMips/pc4_wb	32'hXXXXXXXX
/tb/MyMips/dmem_wb	32'hXXXXXXXX
/tb/MyMips/control_in	16'h0010
Now	1140 ns



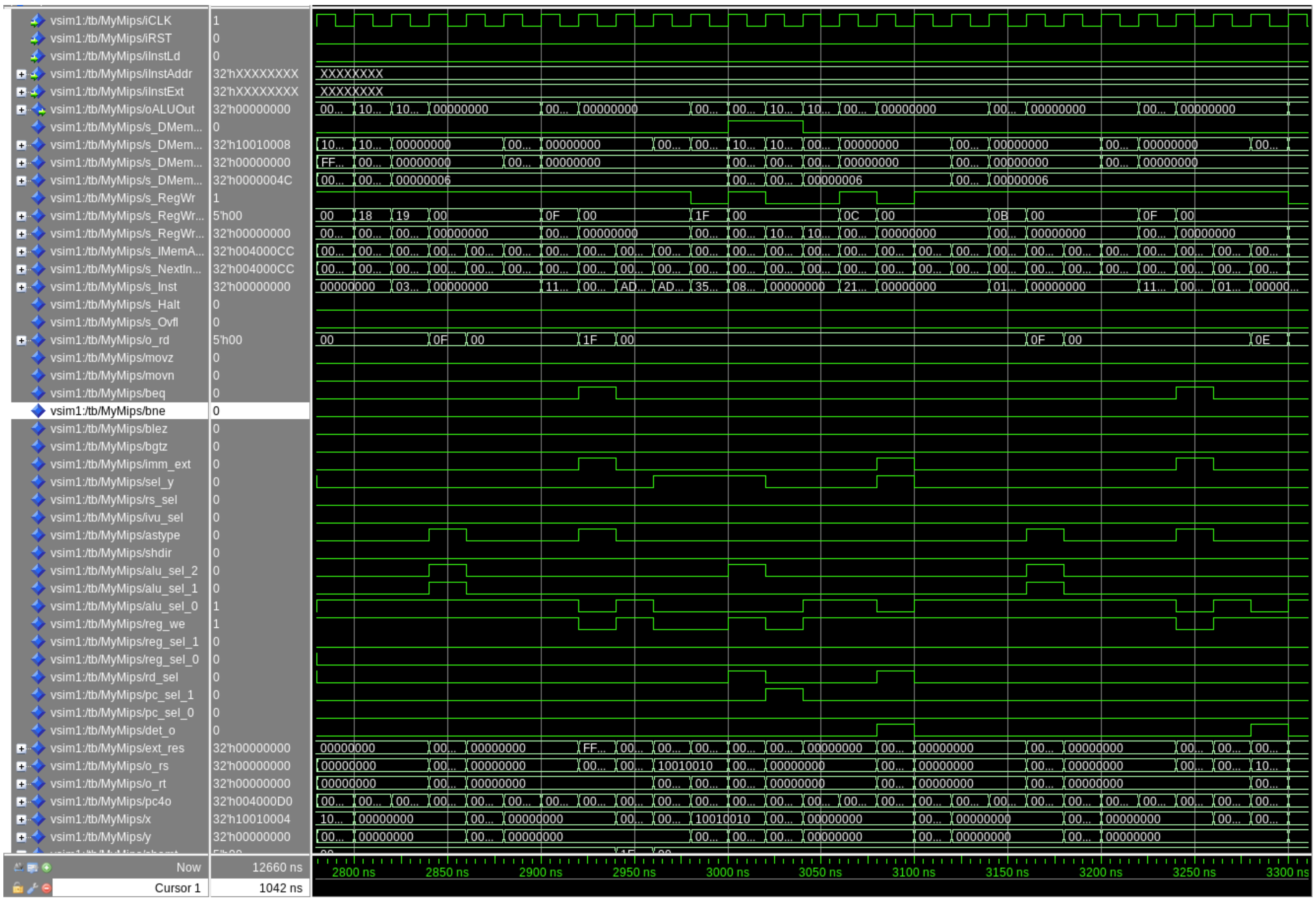
/tb/MyMips/iCLK	1
/tb/MyMips/iRST	0
/tb/MyMips/iInstLd	0
/tb/MyMips/iInstAddr	32'hXXXXXXXX
/tb/MyMips/iInstExt	32'hXXXXXXXX
/tb/MyMips/oALUOut	32'hXXXXXXXX
/tb/MyMips/s_DMemWr	U
/tb/MyMips/s_DMemAddr	32'hXXXXXXXX
/tb/MyMips/s_DMemData	32'hXXXXXXXX
/tb/MyMips/s_DMemOut	32'h00000006
/tb/MyMips/s_RegWr	U
/tb/MyMips/s_RegWrAddr	5'hXX
/tb/MyMips/s_RegWrData	32'h00000000
/tb/MyMips/s_IMemAddr	32'hXXXXXXXX
/tb/MyMips/s_NextInstAddr	32'hXXXXXXXX
/tb/MyMips/s_Inst	32'h20080006
/tb/MyMips/s_Halt	U
/tb/MyMips/s_Ovfl	U
/tb/MyMips/o_rd	5'hXX
/tb/MyMips/movz	0
/tb/MyMips/movn	0
/tb/MyMips/beq	0
/tb/MyMips/bne	0
/tb/MyMips/blez	0
/tb/MyMips/bgtz	0
/tb/MyMips/imm_ext	0
/tb/MyMips/sel_y	0
/tb/MyMips/rs_sel	0
/tb/MyMips/ivu_sel	0
/tb/MyMips/astype	0
/tb/MyMips/shdir	0
/tb/MyMips/alu_sel_2	0
/tb/MyMips/alu_sel_1	0
/tb/MyMips/alu_sel_0	0
/tb/MyMips/reg_we	1
/tb/MyMips/reg_sel_1	0
/tb/MyMips/reg_sel_0	0
/tb/MyMips/rd_sel	0
/tb/MyMips/pc_sel_1	0
/tb/MyMips/pc_sel_0	0
/tb/MyMips/det_o	0
/tb/MyMips/ext_res	32'h0000XXXX
/tb/MyMips/o_rs	32'h00000000
/tb/MyMips/o_rt	32'h00000000
/tb/MyMips/pc4o	32'hXXXXXXXX
/tb/MyMips/x	32'h00000000
/tb/MyMips/y	32'h00000000
/tb/MyMips/shamt	5'h00
/tb/MyMips/z_id	1
/tb/MyMips/z_wb	U
/tb/MyMips/neg	0
/tb/MyMips/o	0
/tb/MyMips/pc_1	0
/tb/MyMips/pc_0	0
/tb/MyMips/pc_en	U
/tb/MyMips/dmem_we	0
/tb/MyMips/halt	0
/tb/MyMips/reg_sel	2'hX
/tb/MyMips/inst	32'hXXXXXXXX
/tb/MyMips/rd	5'hXX
/tb/MyMips/ov_in	U
/tb/MyMips/ov_out	U
/tb/MyMips/o_ex	8'hXX
/tb/MyMips/o_alu	32'h00000000
/tb/MyMips/o_shamt	5'hXX
/tb/MyMips/o_rsd_ex	32'hXXXXXXXX
/tb/MyMips/o_rsd_wb	32'hXXXXXXXX
/tb/MyMips/o_rtd_ex	32'hXXXXXXXX
/tb/MyMips/o_imm	32'hXXXXXXXX
/tb/MyMips/o_wb	7'hXX
/tb/MyMips/alu_wb	32'hXXXXXXXX
/tb/MyMips/pc4_wb	32'hXXXXXXXX
/tb/MyMips/dmem_wb	32'hXXXXXXXX
/tb/MyMips/control_in	16'h0010
Now	1140 ns



/tb/MyMips/iCLK	1
/tb/MyMips/iRST	0
/tb/MyMips/iInstLd	0
/tb/MyMips/iInstAddr	32'hXXXXXXXX
/tb/MyMips/iInstExt	32'hXXXXXXXX
/tb/MyMips/oALUOut	32'hXXXXXXXX
/tb/MyMips/s_DMemWr	U
/tb/MyMips/s_DMemAddr	32'hXXXXXXXX
/tb/MyMips/s_DMemData	32'hXXXXXXXX
/tb/MyMips/s_DMemOut	32'h00000006
/tb/MyMips/s_RegWr	U
/tb/MyMips/s_RegWrAddr	5'hXX
/tb/MyMips/s_RegWrData	32'h00000000
/tb/MyMips/s_IMemAddr	32'hXXXXXXXX
/tb/MyMips/s_NextInstAddr	32'hXXXXXXXX
/tb/MyMips/s_Inst	32'h20080006
/tb/MyMips/s_Halt	U
/tb/MyMips/s_Ovfl	U
/tb/MyMips/o_rd	5'hXX
/tb/MyMips/movz	0
/tb/MyMips/movn	0
/tb/MyMips/beq	0
/tb/MyMips/bne	0
/tb/MyMips/blez	0
/tb/MyMips/bgtz	0
/tb/MyMips/imm_ext	0
/tb/MyMips/sel_y	0
/tb/MyMips/rs_sel	0
/tb/MyMips/ivu_sel	0
/tb/MyMips/astype	0
/tb/MyMips/shdir	0
/tb/MyMips/alu_sel_2	0
/tb/MyMips/alu_sel_1	0
/tb/MyMips/alu_sel_0	0
/tb/MyMips/reg_we	1
/tb/MyMips/reg_sel_1	0
/tb/MyMips/reg_sel_0	0
/tb/MyMips/rd_sel	0
/tb/MyMips/pc_sel_1	0
/tb/MyMips/pc_sel_0	0
/tb/MyMips/det_o	0
/tb/MyMips/ext_res	32'h0000XXXX
/tb/MyMips/o_rs	32'h00000000
/tb/MyMips/o_rt	32'h00000000
/tb/MyMips/pc4o	32'hXXXXXXXX
/tb/MyMips/x	32'h00000000
/tb/MyMips/y	32'h00000000
/tb/MyMips/shamt	5'h00
/tb/MyMips/z_id	1
/tb/MyMips/z_wb	U
/tb/MyMips/neg	0
/tb/MyMips/o	0
/tb/MyMips/pc_1	0
/tb/MyMips/pc_0	0
/tb/MyMips/pc_en	U
/tb/MyMips/dmem_we	0
/tb/MyMips/halt	0
/tb/MyMips/reg_sel	2'hX
/tb/MyMips/inst	32'hXXXXXXXX
/tb/MyMips/rd	5'hXX
/tb/MyMips/ov_in	U
/tb/MyMips/ov_out	U
/tb/MyMips/o_ex	8'hXX
/tb/MyMips/o_alu	32'h00000000
/tb/MyMips/o_shamt	5'hXX
/tb/MyMips/o_rsd_ex	32'hXXXXXXXX
/tb/MyMips/o_rsd_wb	32'hXXXXXXXX
/tb/MyMips/o_rtd_ex	32'hXXXXXXXX
/tb/MyMips/o_imm	32'hXXXXXXXX
/tb/MyMips/o_wb	7'hXX
/tb/MyMips/alu_wb	32'hXXXXXXXX
/tb/MyMips/pc4_wb	32'hXXXXXXXX
/tb/MyMips/dmem_wb	32'hXXXXXXXX
/tb/MyMips/control_in	16'h0010
Now	1140 ns

The waveform shows all instructions supported by the processor. All instructions produce the correct signals and are passed correctly through the pipeline. The result of the execution matches expected values and therefore is correct.

[1.c.ii] Include an annotated waveform in your writeup of two iterations or recursions of these programs executing correctly and provide a short discussion of result correctness. In your waveform and annotation, provide 3 different examples (at least one data-flow and one control-flow) of where you did not have to use the maximum number of NOPs.



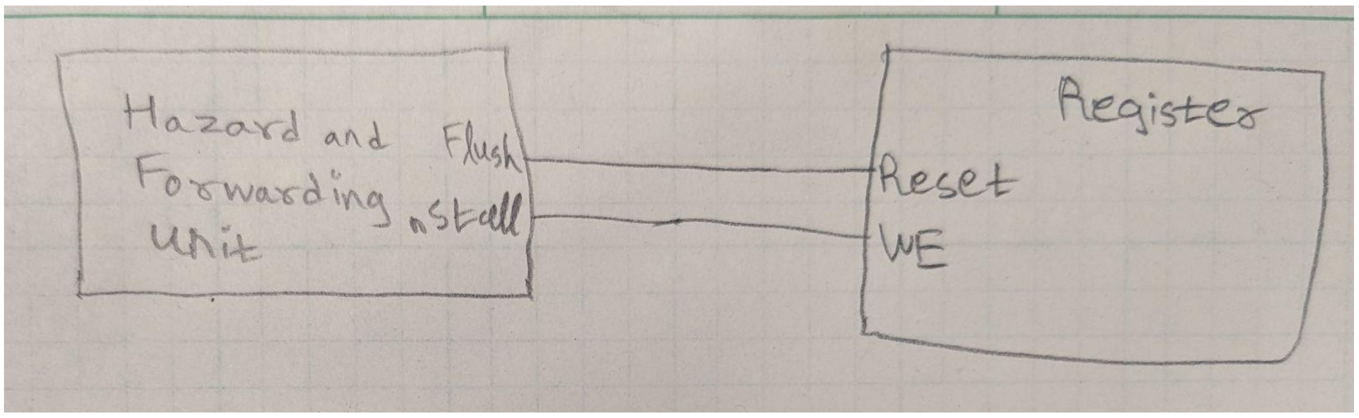
vsim1:/tb/MyMips/iCLK 1
vsim1:/tb/MyMips/iRST 0
vsim1:/tb/MyMips/iInstLd 0
vsim1:/tb/MyMips/iInstAddr 32'hXXXXXXXX
vsim1:/tb/MyMips/iInstExt 32'hXXXXXXXX
vsim1:/tb/MyMips/oALUOut 32'h00000000
vsim1:/tb/MyMips/s_DMem... 0
vsim1:/tb/MyMips/s_DMem... 32'h10010008
vsim1:/tb/MyMips/s_DMem... 32'h00000000
vsim1:/tb/MyMips/s_DMem... 32'h0000004C
vsim1:/tb/MyMips/s_RegWr 1
vsim1:/tb/MyMips/s_RegWr... 5'h00
vsim1:/tb/MyMips/s_RegWr... 32'h00000000
vsim1:/tb/MyMips/s_IMemA... 32'h004000CC
vsim1:/tb/MyMips/s_NextIn... 32'h004000CC
vsim1:/tb/MyMips/s_Inst 32'h00000000
vsim1:/tb/MyMips/s_Halt 0
vsim1:/tb/MyMips/s_Ovfl 0
vsim1:/tb/MyMips/o_rd 5'h00
vsim1:/tb/MyMips/movz 0
vsim1:/tb/MyMips/movn 0
vsim1:/tb/MyMips/beq 0
vsim1:/tb/MyMips/bne 0
vsim1:/tb/MyMips/blez 0
vsim1:/tb/MyMips/bgtz 0
vsim1:/tb/MyMips/imm_ext 0
vsim1:/tb/MyMips/sel_y 0
vsim1:/tb/MyMips/rs_sel 0
vsim1:/tb/MyMips/ivu_sel 0
vsim1:/tb/MyMips/astype 0
vsim1:/tb/MyMips/shdir 0
vsim1:/tb/MyMips/alu_sel_2 0
vsim1:/tb/MyMips/alu_sel_1 0
vsim1:/tb/MyMips/alu_sel_0 1
vsim1:/tb/MyMips/reg_we 1
vsim1:/tb/MyMips/reg_sel_1 0
vsim1:/tb/MyMips/reg_sel_0 0
vsim1:/tb/MyMips/rd_sel 0
vsim1:/tb/MyMips/pc_sel_1 0
vsim1:/tb/MyMips/pc_sel_0 0
vsim1:/tb/MyMips/det_o 0
vsim1:/tb/MyMips/ext_res 32'h00000000
vsim1:/tb/MyMips/o_rs 32'h00000000
vsim1:/tb/MyMips/o_rt 32'h00000000
vsim1:/tb/MyMips/pc4o 32'h004000D0
vsim1:/tb/MyMips/x 32'h10010004
vsim1:/tb/MyMips/y 32'h00000000
Now 12660 ns
Cursor 1 1042 ns
2800 ns
2850 ns
2900 ns
2950 ns
3000 ns
3050 ns
3100 ns
3150 ns
3200 ns
3250 ns
3300 ns

From 2840 to 2900 ns, and 3080 to 3140 ns, no-ops execute to avoid data-flow hazards. From 3020 to 3060, no-ops execute to avoid control flow hazards.

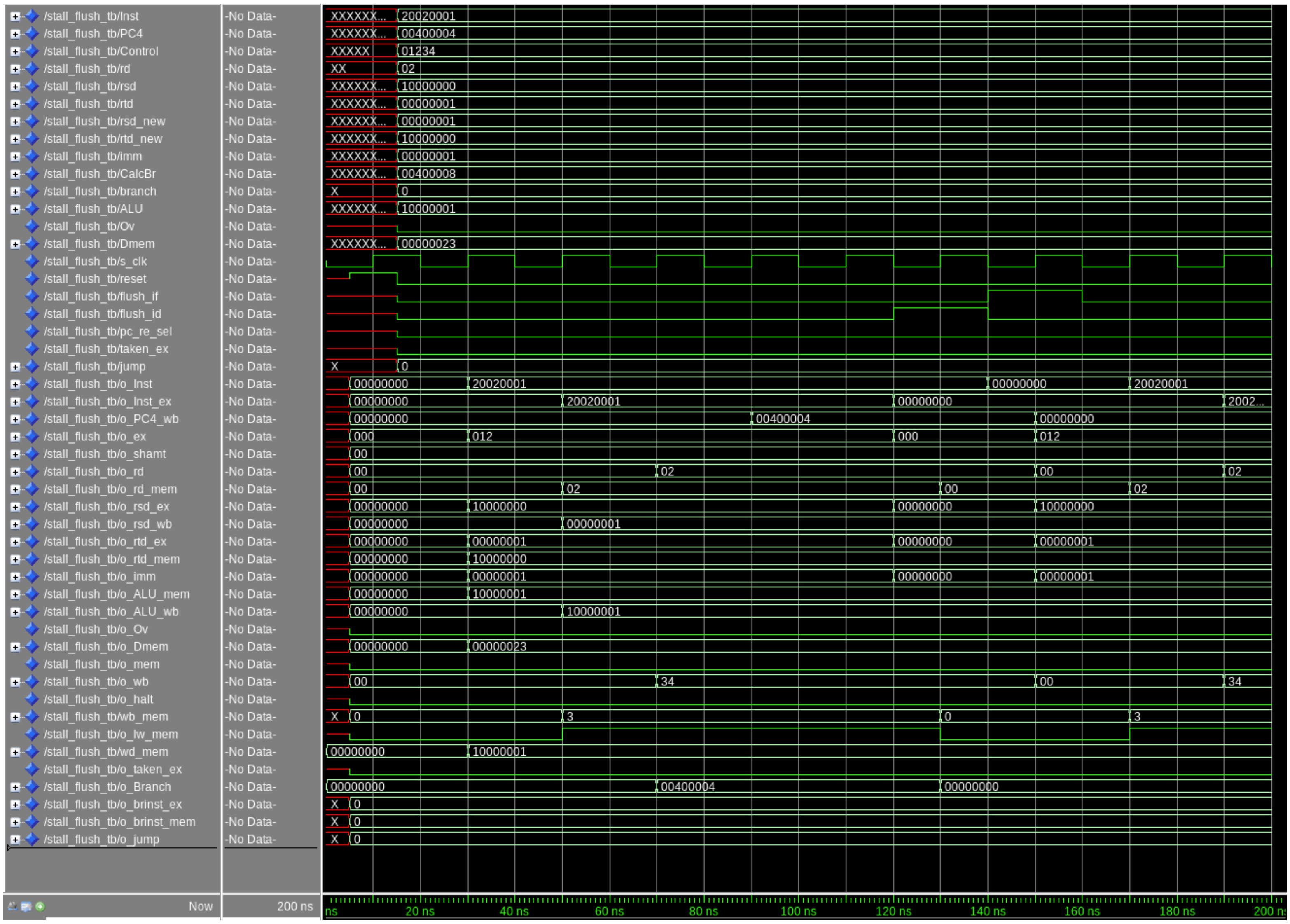
[1.d] report the maximum frequency your software-scheduled pipelined processor can run at and determine what your critical path is (specify each module/entity/component that this path goes through).

Max Frequency : 54.45 MHz
Critical Path : RegFile ⇒ Comparator ⇒ Branch Setting Unit ⇒ Program Counter

[2.a.ii] Draw a simple schematic showing how you could implement stalling and flushing operations given an ideal N-bit register.



[2.a.iii] Create a testbench that instantiates all four of the registers in a single design. Show that values that are stored in the initial IF/ID register are available as expected four cycles later, and that new values can be inserted into the pipeline every single cycle. Most importantly, this testbench should also test that each pipeline register can be individually stalled or flushed.



[2.b.i] list which instructions produce values, and what signals (i.e., bus names) in the pipeline these correspond to.

Instructions:
sll, srl, sra, sllv, srlv, srav, movz, movn, add, addu, sub, subu, and, or, xo, not, slt, sltu, addi, addiu, slti, sltiu, andi, or, xori, lui, jal, jalr, sw, lw

Signals:
s_DMemAddr, s_RegWrData, o_alu, alu_wb, pc4o, pc4_wb, s_DMemData, o_rt, o_rtd_ex, s_DMemOut, dmem_wb

[2.b.ii] List which of these same instructions consume values, and what signals in the pipeline these correspond to.

Instructions:
sll, srl, sra, sllv, srlv, srav, movz, movn, add, addu, sub, subu, and, or, xo, not, slt, sltu, addi, addiu, slti, sltiu, andi, or, xori, lui, jr, jalr, beq, bne, blez, bgtz, sw, lw

Signals:
o_rs, o_rt, o_rsd_ex, o_rtd_ex

[2.b.iii] generalized list of potential data dependencies. From this generalized list, select those dependencies that can be forwarded (write down the corresponding pipeline stages that will be forwarding and receiving the data), and those dependencies that will require hazard stalls.

Instruction in MEM or WB stage is writing to register used in the EX stage. For most instructions, data can be forwarded from both stages. If instruction in MEM is lw, flush IF/ID and ID/EX. If instruction in MEM or WB Stage is movn or movz, flush IF/ID and ID/EX.

[2.b.iv] global list of the datapath values and control signals that are required during each pipeline stage

flush_if, flush_id, pc_re, mem_wb, lw, o_Inst_ex, o_rd_mem, o_sel_rsd, o_sel_rtd, x_pre, y_pre

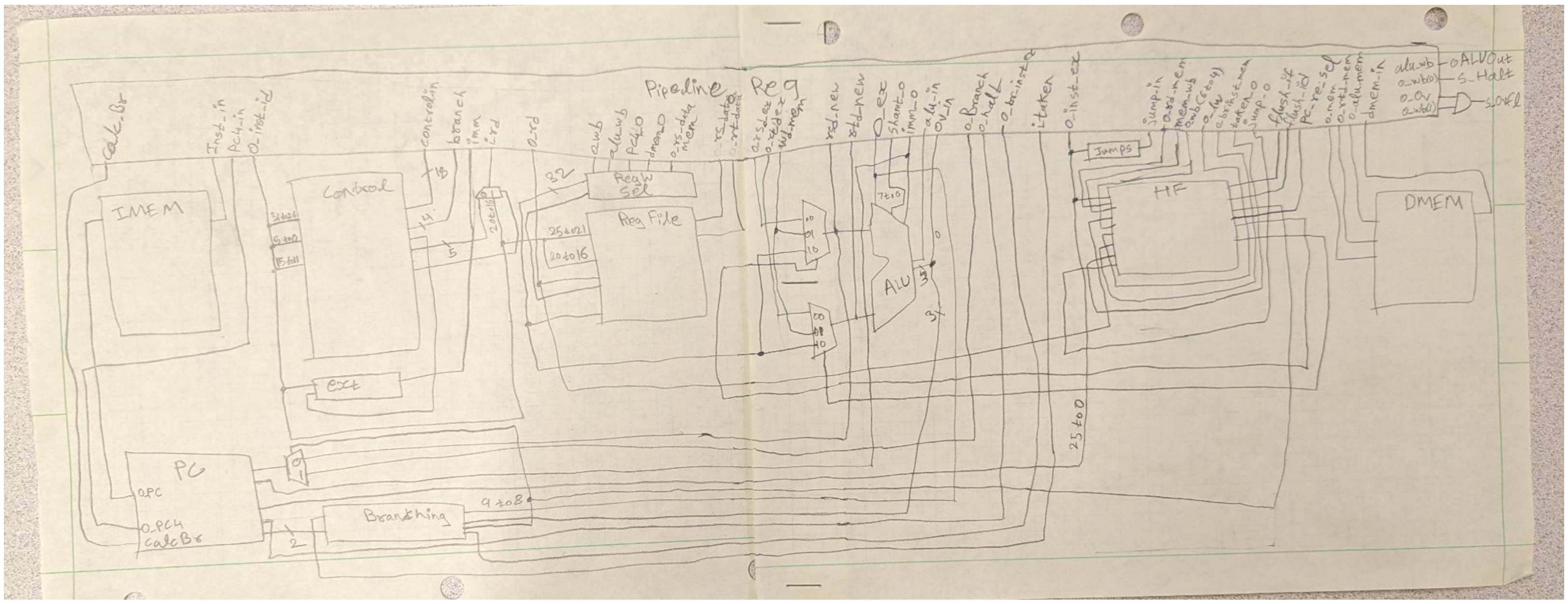
[2.c.i] list all instructions that may result in a non-sequential PC update and in which pipeline stage that update occurs.

j, jal, jr, jalr, beq, bne, blez, bgtz all update the PC in EX stage

[2.c.ii] For these instructions, list which stages need to be stalled and which stages need to be squashed/flushed relative to the stage each of these instructions is in.

Once the control flow instruction reach the MEM stage, the IF/ID and ID/EX pipeline registers must be flushed.

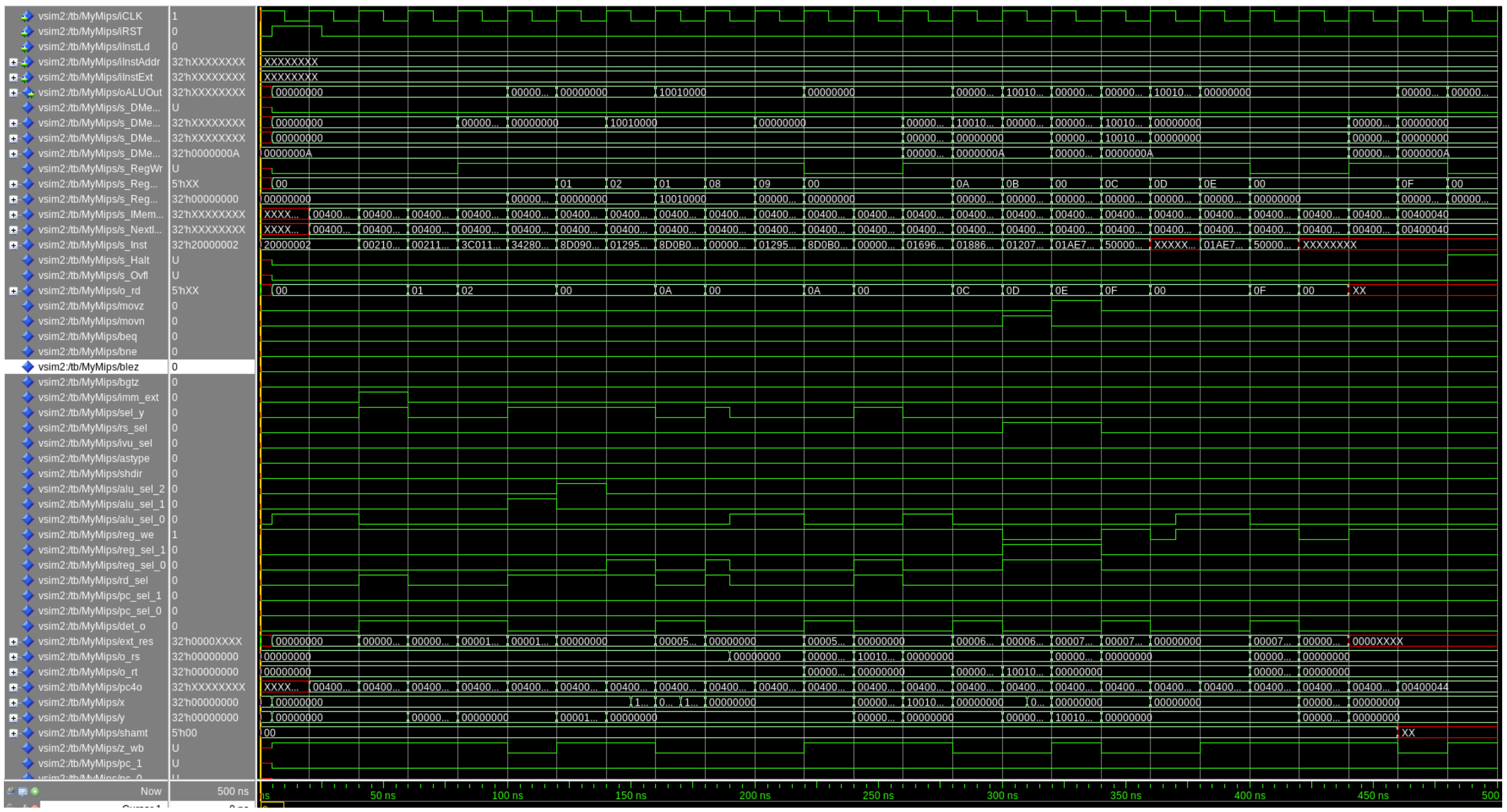
[2.d] implement the hardware-scheduled pipeline using only structural VHDL. As with the previous processors that you have implemented, start with a high-level schematic drawing of the interconnection between components.



[2.e.i] Create a spreadsheet to track these cases and justify the coverage of your testing approach. Include this spreadsheet in your report as a table.

ALU op requires data from prev instruction
MEM op requires data from prev instruction
Data hazard by instruction just after LW
Data hazard by instruction just after movn/movz

The first two cases test if data is forwarded under different conditions. The next two tests check if flushing occurs when a non-forwardable data hazard occurs.

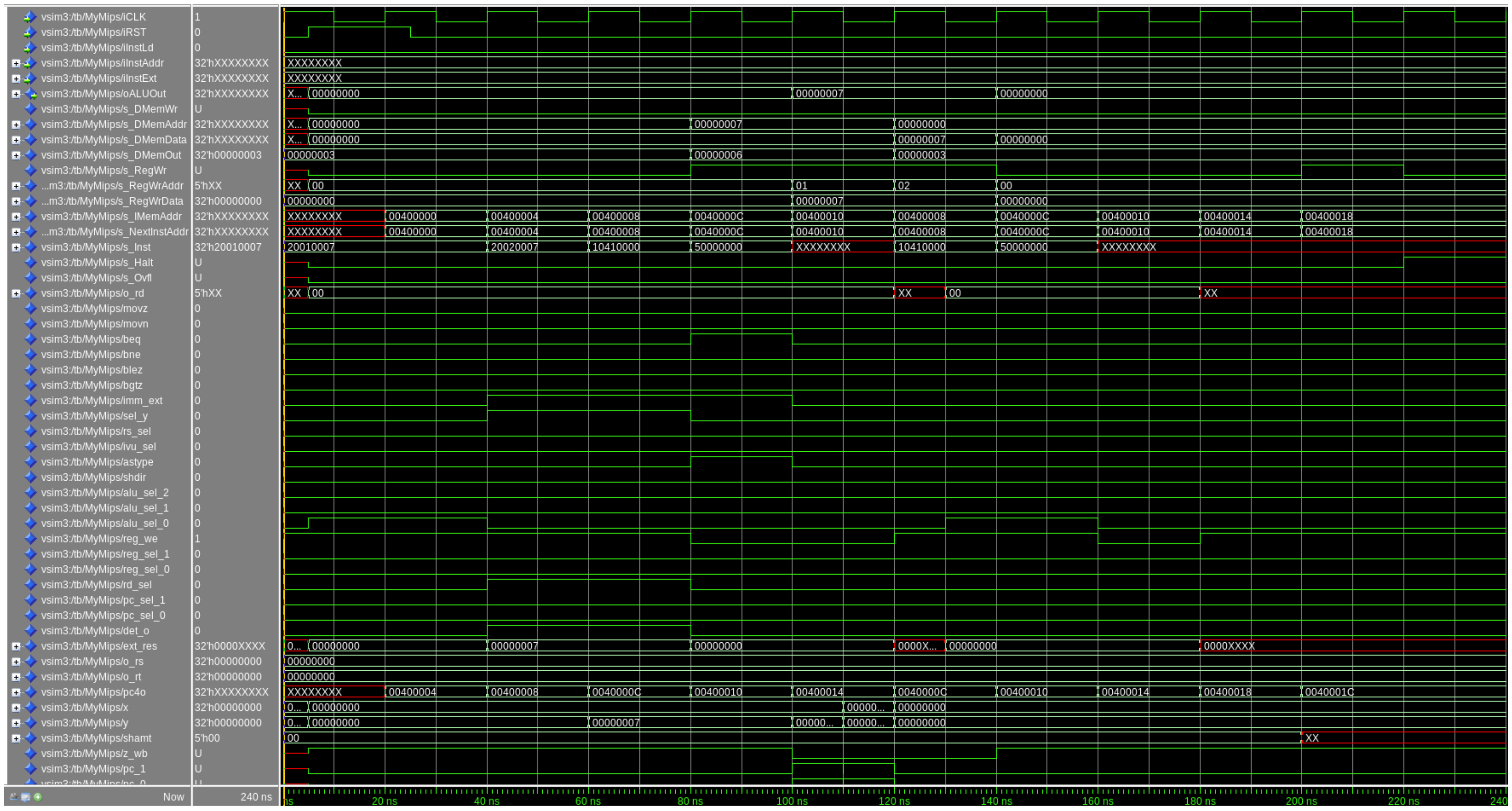

vsim2:/tb/MyMips/iCLK 1
vsim2:/tb/MyMips/iRST 0
vsim2:/tb/MyMips/iInstLd 0
vsim2:/tb/MyMips/iInstAddr 32'hXXXXXXXX
vsim2:/tb/MyMips/iInstExt 32'hXXXXXXXX
vsim2:/tb/MyMips/oALUOut 32'hXXXXXXXX
vsim2:/tb/MyMips/s_DMe... U
vsim2:/tb/MyMips/s_DMe... 32'hXXXXXXXX
vsim2:/tb/MyMips/s_DMe... 32'hXXXXXXXX
vsim2:/tb/MyMips/s_DMe... 32'h0000000A
vsim2:/tb/MyMips/s_RegWr U
vsim2:/tb/MyMips/s_Reg... 5'hXX
vsim2:/tb/MyMips/s_Reg... 32'h00000000
vsim2:/tb/MyMips/s_IMem... 32'hXXXXXXXX
vsim2:/tb/MyMips/s_NextI... 32'hXXXXXXXX
vsim2:/tb/MyMips/s_Inst 32'h20000002
vsim2:/tb/MyMips/s_Halt U
vsim2:/tb/MyMips/s_Ovfl U
vsim2:/tb/MyMips/o_rd 5'hXX
vsim2:/tb/MyMips/movz 0
vsim2:/tb/MyMips/movn 0
vsim2:/tb/MyMips/beq 0
vsim2:/tb/MyMips/bne 0
vsim2:/tb/MyMips/blez 0
vsim2:/tb/MyMips/bgtz 0
vsim2:/tb/MyMips/imm_ext 0
vsim2:/tb/MyMips/sel_y 0
vsim2:/tb/MyMips/rs_sel 0
vsim2:/tb/MyMips/ivu_sel 0
vsim2:/tb/MyMips/astype 0
vsim2:/tb/MyMips/shdir 0
vsim2:/tb/MyMips/alu_sel_2 0
vsim2:/tb/MyMips/alu_sel_1 0
vsim2:/tb/MyMips/alu_sel_0 0
vsim2:/tb/MyMips/reg_we 1
vsim2:/tb/MyMips/reg_sel_1 0
vsim2:/tb/MyMips/reg_sel_0 0
vsim2:/tb/MyMips/rd_sel 0
vsim2:/tb/MyMips/pc_sel_1 0
vsim2:/tb/MyMips/pc_sel_0 0
vsim2:/tb/MyMips/det_o 0
vsim2:/tb/MyMips/ext_res 32'h0000XXXX
vsim2:/tb/MyMips/o_rs 32'h00000000
vsim2:/tb/MyMips/o_rt 32'h00000000
vsim2:/tb/MyMips/pc4o 32'hXXXXXXXX
vsim2:/tb/MyMips/x 32'h00000000
vsim2:/tb/MyMips/y 32'h00000000
vsim2:/tb/MyMips/shamt 5'h00
vsim2:/tb/MyMips/z_wb U
vsim2:/tb/MyMips/pc_1 U
Now 500 ns
50 ns
100 ns
150 ns
200 ns
250 ns
300 ns
350 ns
400 ns
450 ns
500

Data forwarded when possible, else flushing occurs. Matches the expected results therefore correct.

[2.e.ii] Create a spreadsheet to track these cases and justify the coverage of your testing approach. Include this spreadsheet in your report as a table.

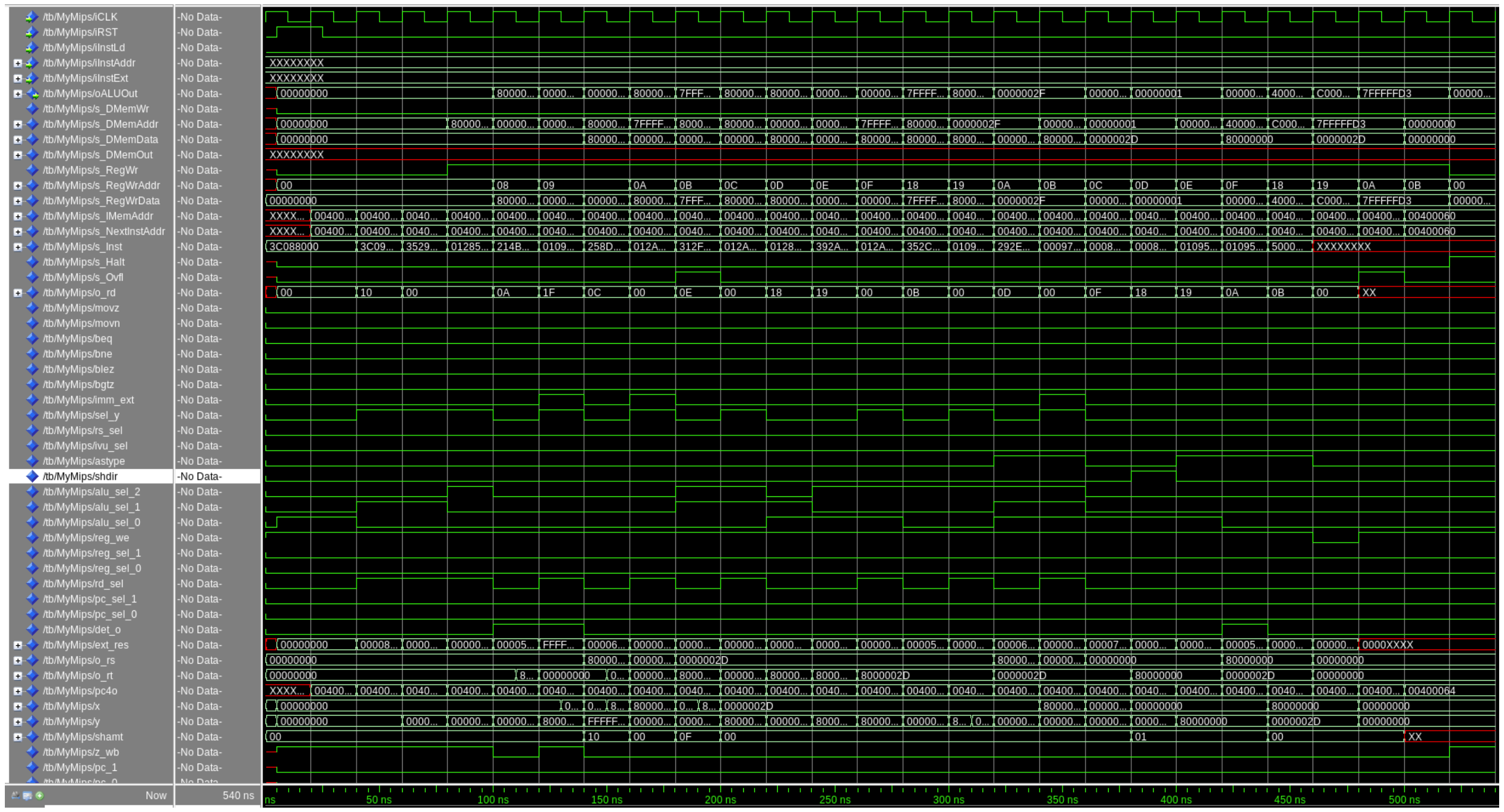
All branch and jump tests in base unit tests

These tests stress test the control flow instructions and sees if they execute and flush in the right manner.



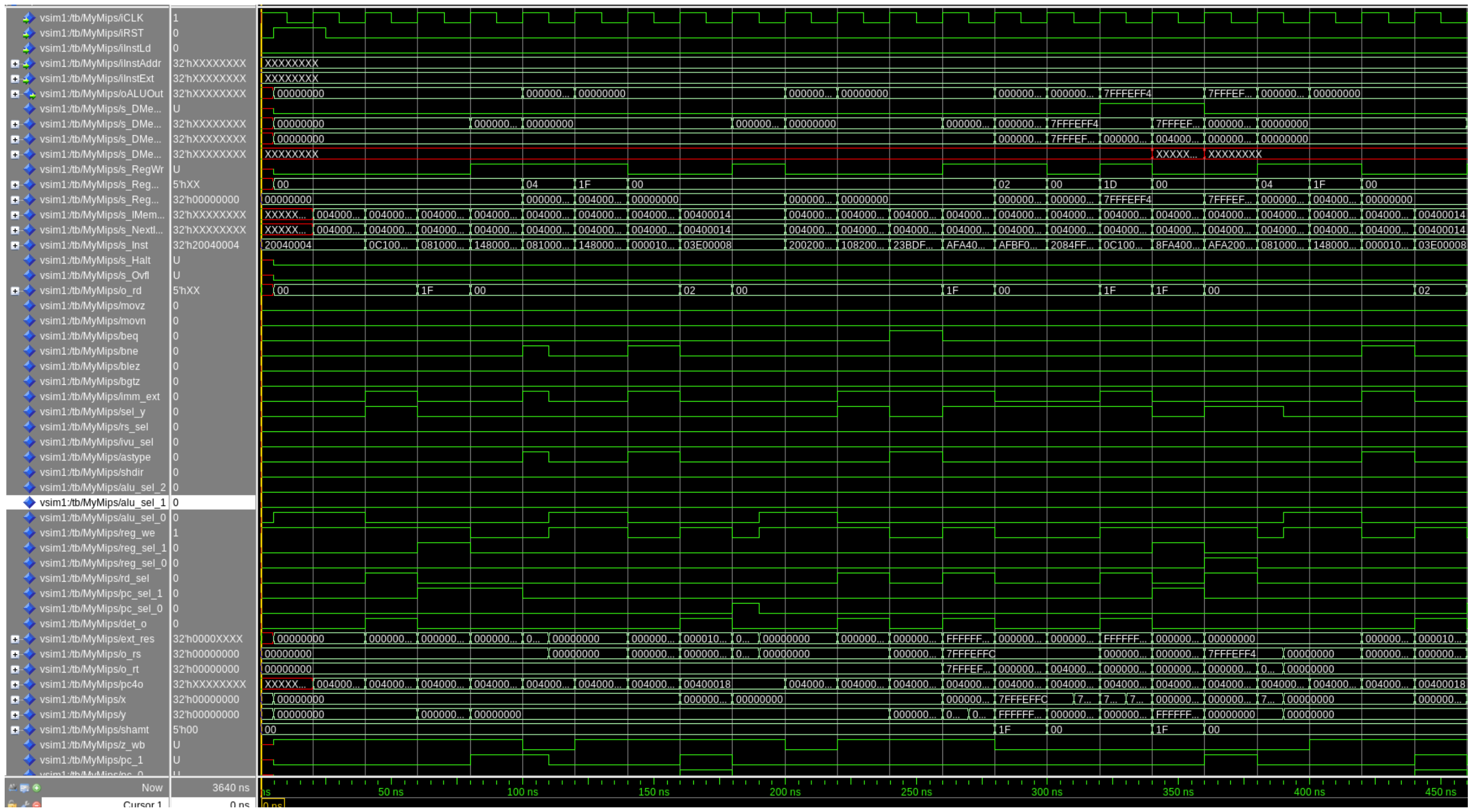
The Above waveform is from one of the many test cases. The pipeline is flushed at each control flow instruction and the right instructions are executed.

[2.e.iii] Proj1_base_test.s



Register/DMEM writes and other states equivalent to Project 1 results.

Proj1_cf_tests.s



```
vsim1:/tb/MyMips/iCLK          1
vsim1:/tb/MyMips/iRST          0
vsim1:/tb/MyMips/iInstLd       0
vsim1:/tb/MyMips/iInstAddr     32'hXXXXXXXX
vsim1:/tb/MyMips/iInstExt      32'hXXXXXXXX
vsim1:/tb/MyMips/oALUOut       32'hXXXXXXXX
vsim1:/tb/MyMips/s_DMe...      U
vsim1:/tb/MyMips/s_DMe...      32'hXXXXXXXX
vsim1:/tb/MyMips/s_DMe...      32'hXXXXXXXX
vsim1:/tb/MyMips/s_DMe...      32'hXXXXXXXX
vsim1:/tb/MyMips/s_RegWr       U
vsim1:/tb/MyMips/s_Reg...      5'hXX
vsim1:/tb/MyMips/s_Reg...      32'h00000000
vsim1:/tb/MyMips/s_IMem...     32'hXXXXXXXX
vsim1:/tb/MyMips/s_NextI...    32'hXXXXXXXX
vsim1:/tb/MyMips/s_Inst        32'h20040004
vsim1:/tb/MyMips/s_Halt        U
vsim1:/tb/MyMips/s_Ovfl        U
vsim1:/tb/MyMips/o_rd          5'hXX
vsim1:/tb/MyMips/movz          0
vsim1:/tb/MyMips/movn          0
vsim1:/tb/MyMips/beq           0
vsim1:/tb/MyMips/bne           0
vsim1:/tb/MyMips/blez          0
vsim1:/tb/MyMips/bgtz          0
vsim1:/tb/MyMips/imm_ext       0
vsim1:/tb/MyMips/sel_y         0
vsim1:/tb/MyMips/rs_sel        0
vsim1:/tb/MyMips/ivu_sel       0
vsim1:/tb/MyMips/astype        0
vsim1:/tb/MyMips/shdir         0
vsim1:/tb/MyMips/alu_sel_2     0
vsim1:/tb/MyMips/alu_sel_1     0
vsim1:/tb/MyMips/alu_sel_0     0
vsim1:/tb/MyMips/reg_we        1
vsim1:/tb/MyMips/reg_sel_1     0
vsim1:/tb/MyMips/reg_sel_0     0
vsim1:/tb/MyMips/rd_sel        0
vsim1:/tb/MyMips/pc_sel_1      0
vsim1:/tb/MyMips/pc_sel_0      0
vsim1:/tb/MyMips/det_o         0
vsim1:/tb/MyMips/ext_res       32'h0000XXXX
vsim1:/tb/MyMips/o_rs          32'h00000000
vsim1:/tb/MyMips/o_rt          32'h00000000
vsim1:/tb/MyMips/pc4o          32'hXXXXXXXX
vsim1:/tb/MyMips/x             32'h00000000
vsim1:/tb/MyMips/y             32'h00000000
vsim1:/tb/MyMips/shamt         5'h00
vsim1:/tb/MyMips/z_wb          U
vsim1:/tb/MyMips/pc_1          U
Now                            3640 ns
Cursor 1                       0 ns
```

Register/DMEM writes and other states equivalent to Project 1 results.

[2.f] report the maximum frequency your hardware-scheduled pipelined processor can run at and determine what your critical path is (specify each module/entity/component that this path goes through).

Max Frequency : 49.12 MHz
Critical Path : Hazard and Forwarding Unit ⇒ Mux ⇒ ALU (Adder) ⇒ Program Counter