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.

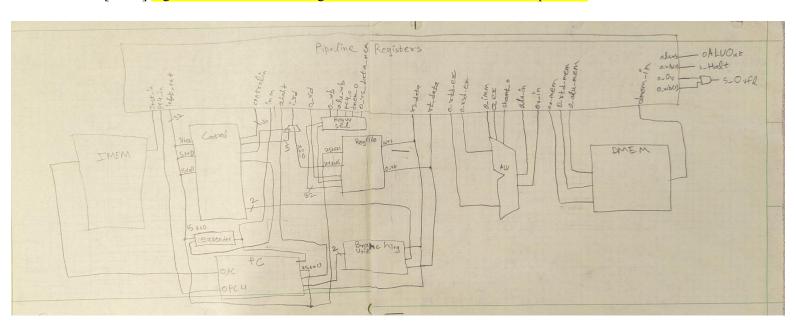
[1.a] Come up with a global list of the datapath values and control signals that are required during each pipeline stage.

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.

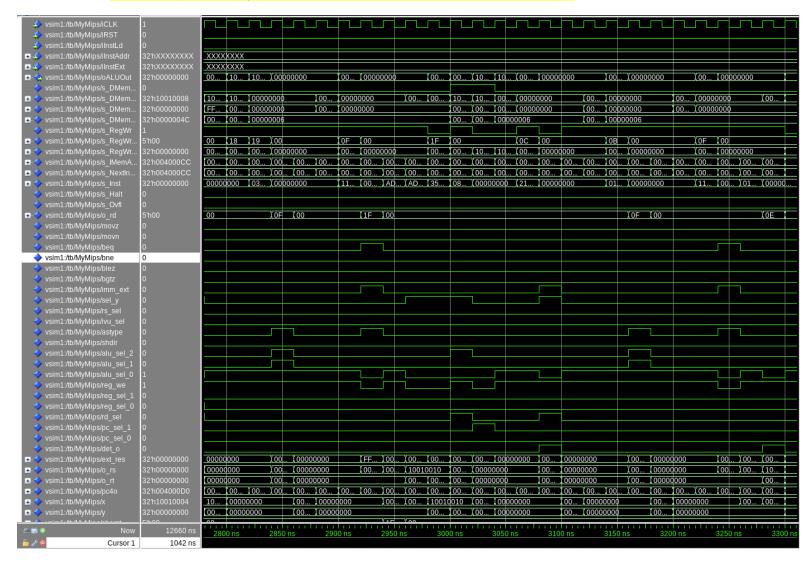
♣ /tb/MyMips/iCLK	1																				
/b/MyMips/iRST	0																				
/tb/MyMips/iInstLd	0																				
■ ❖ /tb/MyMips/iInstAddr	32'hXXXXXXXX	XXXXXXXX																			
	32'hXXXXXXXX 32'hXXXXXXXX	XXXXXXXX (00000000					00000006		00000005	10010000	00000000	00000000	00000001	10010000	Tererere 2	00000001		10010004	100000000	100000000	FFFFFFF3 1000
/b/MyMips/s_DMemWr	U	(00000000					.00000000	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	00000000	10010000	00000000	100000003	100000001	110010000	FFFFFF	100000001		10010004	00000000	100000003	177777773 .000
Ttb/MyMips/s_DMemAddr	32'hXXXXXXXX	00000000				00000006	FFFFFE7	00000005	10010000	0000000C	00000003	00000001	10010000	FFFFFFF3	00000001		10010004	0000000C	00000003	FFFFFFF3	00000002 000
*/ /tb/MyMips/s_DMemData	32'hXXXXXXXX	(00000000								00000006		00000000		FFFFFE7			00000006				00000001
/tb/MyMips/s_DMemOut	32'h00000006	00000006				00000000			00000006	00000000	00000006			00000000	100000006		100000000	100000000	100000006	100000000	00000006
→ /tb/MyMips/s_RegWr ▼ → /tb/MyMips/s_RegWrAddr	5'hXX	(00					108	09	0A	I OB	0C	IOD	OF	118	IOF	I 19	110	100	111	112	113 114
■	32'h00000000	00000000								10010000											FFFFFFF3 000
	32'hXXXXXXXX																				00400048 004
*/ /tb/MyMips/s_InernAddr */ /tb/MyMips/s_NextInstAddr */ /tb/MyMips/s_Inst	32'hXXXXXXXX	20080006																			00400048 004 0129B823 0C
/tb/MyMips/s_irist	J	20080000	1	2409FFE1	340A0005	SCOPIOOI	100080040	100000042	394F0004	. 00/00000	00097043	29390006	ZUSUFFFA	AD760004	U1E000U4	01599000	101599001	10330A020	USUFA621	U330B022	01298023 100
/tb/MyMips/s_Ovfl	Ŭ																				
• /tb/MyMips/o_rd	5'nXX	(00			1F	00	02	0C	0D	00		0E	00	1F	00	11	12	13	14	15	16 17
tb/MyMips/movz	0																				
/tb/MyMips/movn	0																				
/tb/MyMips/bne	0																				
/tb/MyMips/blez	0																				
/tb/MyMips/bgtz /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 ♦ /th/MyMips/alu_sel_2	0														-						
/tb/MyMips/alu_sel_1	0																				
nonkylinjssina sel 2 hohkylinjssina sel 2 hohkylinjssina sel 0 hohkylinjssina sel 1 hohkylinjssina sel 1 hohkylinjssina sel 1	0																				
/tb/MyMips/reg_we	1																				
/tb/MyMips/reg_set_1	0																				
/tb/MyMips/rd_sel	0																				
/tb/MyMips/pc_sel_1	0																				
→ /tb/MyMips/pc_sel_0 → /tb/MyMips/det_o	0				-															-	
* / /tb/MyMips/ext res	32'h0000XXXX	(00000000		00000006	FFFFFFE7	00000005	00001001	00006040	00006842	00000004	00000000	00007043	00000006	FFFFFFA	00000004	00008804	00009006	00009807	0000A020	0000A821	0000B022 000
/tb/MyMips/o_rs	32'h00000000	00000000								00000005					10010000						00000001 FFF
	32'h00000000	00000000						00000006		00000000		FFFFFFE7			00000006			FFFFFFE7			FFI
→ /tb/MyMips/pc4o → /tb/MyMips/x	32'hXXXXXXXX 32'h00000000	XXXXX U		00400008	0040000C	00400010	00400014	00400018	0040001C			100400028					00000001		00400044		0040004C 1004 00000006 1000
■ /tb/MyMips/y	32'h00000000	00000000			00000006	FFFFFFE7	00000005	00001001	00000006			00000000							FFFFFFE7	00000001	
	5'h00	00				1F	00		01		00		01	00	05	00	01			00	
/tb/MyMips/z_id	1																				
/tb/MyMips/z_wb /tb/MyMips/neg	0																				
/tb/MyMips/o	0																				
→ /เม/wiywiips/pc_1	0																				
	0																				
/tb/MyMips/pc_en /tb/MyMips/dmem we	0																				
/b/MyMips/halt	0																				
/tb/MyMips/reg_sel	2'hX	(0									0070000				0	0450000	0150000	0450000		0005100	
→ /tb/MyMips/inst → /tb/MyMips/rd	32'hXXXXXXXX 5'hXX	(00000000				340A0005														030FA821 115	0330B022 012
→ /tb/MyMips/ov in	U		ì	00		OA.	. 35					, oc			, 00						1/
/tb/MyMips/ov_out	U																				
/tb/MyMips/o_ex	8'hXX	(00			80															00	10
→ /tb/MyMips/o_alu → /tb/MyMips/o_shamt	32'h00000000 5'hXX	(00					100000005		0000000C	00000003	00000001					10010004	.0000000C	.00000003	1111111113	00000002	00000007 000
→ /tb/MyMips/o_rsd_ex	32'hXXXXXXXX	(00000000					.00		.01			10010000					00000001				00000006 000
■ /tb/MyMips/o_rsd_wb	32'hXXXXXXXX	00000000											00000005	10010000	00000000	00000005		10010000			
	32'hXXXXXXXX	00000000							00000006		00000000		FFFFFFE7	00000000		00000006				00000001	
→ /tb/MyMips/o_imm → /tb/MyMips/o_wb	32'hXXXXXXXX 7'hXX	(00000000						10	00006040	00006842	00000004				I FFFFFFFA	00000004			100009807	0000A020	0000A821 000
	32'hXXXXXXXX	(00000000				10			00000005	10010000	0000000C					00000001				00000003	FFFFFFF3 000
/tb/MyMips/pc4_wb	32'hXXXXXXXX	00000000					00400004	00400008	0040000C	00400010	00400014	00400018	0040001C	00400020	00400024	00400028	0040002C	00400030	00400034	00400038	0040003C 004
/tb/MyMips/pc4_wb /tb/MyMips/dmem_wb /tb/MyMips/control_in	32'hXXXXXXXX	00000000		00000006			00000000			00000006					00000000			00000000			00000000 000
±	16'h0010	(0110	, , , , , , , , , , , , , , , , , , ,		8010	8410	8210	0110		8510	8014	.1110	19/10	B/10	18080			3110	0012	0010	1012 1101
	1140 ns			50 ns		10	0 ns		150 ns		200	0 ns		250 ns		30	0 ns		350 ns		400 ns

CLK RST	0																				<u> </u>
InstAddr	u 32'hXXXXXXXX	XXXXXXXX																			
instExt ALUOut	32'hXXXXXXXX 32'hXXXXXXXX	XXXXXXXX		100000000		00000001	00000000				00400050	00000000				00400074	100000000		00000004	00000000	10000
MemWr	U	00000002	00000007	100000000		100000001	00000000				.00400050	00000000				00400074	100000000		00000004	00000005	10000
ddr ata	32'hXXXXXXXX		00000000			00000000				00400050	00000000				00400074	00000000		00000004			
ı	32'hXXXXXXXX 32'h00000006	00000001		IFFFFFFE7	00000001	100000000				00000000	00000006				00000000	00000006		00000006		00000000	
ı	υ																				
	5'hXX 32'h00000000			16	17	1F 100400050	100000000				1F 00400074	00				00400074	00000000		08	09	1000C
	32'hXXXXXXXX	0040004C	00400050	00400064	00400068	0040006C	00400070	00400074		00400054	00400058	0040005C			00400078	0040007C	00400080	00400084	00400088	0040008C	0040
ı	32'hXXXXXXXXX 32'h20080006	I0040004C I0C100019		00400064	00400068	10040006C		00400074		00400054			00400060					00400084 0100780B			
	J	00100013	00000000				OSEOI GOS	.00000000				05200000	00000000		31400004	024/14024	101403023	01007000	01007007		010
	,	117	1F	100				Î 1F	100								109	I OA	0F		100
	^	1) IF	100				115	.00								109	I UA	UF		100
							 														
	0																				
ı	0																				
	0																				
	0																				
																		-			
	!)																				
				┼																	
C) 2'h0000XXXX		00000010	00000000				00005000	*********					00000000				00005025	00007000		1000
	h00000000		00000000					0000F809 00400050					00000008			00000004			000007808		
	h00000000			00000000														000000C			
	hXXXXXXXX h00000000			00400068		100400070	100400074			00000000	0040005C	00400060			0040007C		00400084			00400090	
	h00000000			00000001														00000005			
	5'h00	00																			
																	 				
	0																				
	0																				
0									1												
	0																				
į	2hX 32hXXXXXXXX	0 10129B823	0C100019	100000000		2	0	03E0F809	100000000		Z	0	03E00008	00000000		31480004	014A4824	014C5025	0100780B	0100780A	000
				00					00								09		0F		00
U																					
8		10			01				00	01					01					40	
	32'h00000000 5'hXX	00000000	00000000	00000001	00000000				00400050	00000000				00400074	00000000		00000004	00000005	0000000D	00000000	
	32'hXXXXXXXXX	00000001	FFFFFFE7	00000000					00400050	00000000				00400074	00000000		00000005			00000006	000
3	2'hXXXXXXXX	00000001	00000006	00000001	FFFFFFE7						00400050	00000000					00000000		00000005		
3	2'hXXXXXXXX 2'hXXXXXXXX			00000001					0000F809	00000000				80000000	00000000			00000005			
		12	10	12	10	18	10					10				00	10				
	32'hXXXXXXXX 32'hXXXXXXXX			00000000		00000001			00400060		00400050		00400054	00400058			00000000		00000004		
	32'hXXXXXXXX	00000006			20400040	3045555					00000000	00000006				00000000	00000006				000
	16'h0010	1010	10019	0110				0018	0110				0000	0110		8610	0610	I 0410	402C	404C	0110

/tb/MyMips/iCLK /tb/MyMips/iRST	0																	
tb/MyMips/ilnstLd	0	VVVVVVVV																
tb/MyMips/iInstAddr tb/MyMips/iInstExt	32'hXXXXXXXX 32'hXXXXXXXXX	XXXXXXXX																
tb/MyMips/oALUOut		00000000			00000009	FFFFFFF2	00000000		FFFFFFC	00000000	FFFFFFF3	00000000	00000005	00000000	00000001	00000000		i
tb/MyMips/s_DMemWr	U																	
tb/MyMips/s_DMemAddr	32'hXXXXXXXX	00000000			FFFFFFF2						00000000	00000005	00000000	00000001				
tb/MyMips/s_DMemData	32'hXXXXXXXX 32'h00000006	00000000		0000000D		00000000		00000004			00000006			00000001	00000000			
tb/MyMips/s_DMemOut/ tb/MyMips/s_RegWr	U	00000000		,00000000		00000000		.00000000	00000000	,00000000	,00000000							i
/tb/MyMips/s_RegWrAddr	5'hXX	0F		00	ОВ	OC.	00		1F	00			1F	00				
/tb/MyMips/s_RegWrData	32'h00000000			00000000										00000000		00000000		
tb/MyMips/s_IMemAddr	32'hXXXXXXXX													004000C4				
tb/MyMips/s_NextInstAddr tb/MyMips/s_Inst	32'hXXXXXXXX 32'h20080006	014A6027											1004000C0	004000C4	XXXXXXXX			
tb/MyMips/s_Halt	U	014/1002/	10000000	,00000000	14001113	. 00000000	13000003		10201113	,00000000	00100030	, 00000000		30000000	70000000			ı
tb/MyMips/s_Ovfl	U																	
b/MyMips/o_rd	5'hXX	0B	0C	00		1F	00			1F	00					XX		H
b/MyMips/movz	0	+																H
:b/MyMips/movn :b/MyMips/beq	0	+																ı
b/MyMips/bne	0																	ı
b/MyMips/blez	0																	ı
b/MyMips/bgtz	0																	H
tb/MyMips/imm_ext tb/MyMips/sel_y	0																	ı
:b/MyMips/rs sel	o																	ī
b/MyMips/ivu_sel	o																	
b/MyMips/astype	0																	
b/MyMips/shdir	0																	H
b/MyMips/alu_sel_2 b/MyMips/alu_sel_1	0																	H
b/MyMips/alu_sei_1 b/MyMips/alu_sel_0	0			-		-				-		-						i
b/MyMips/reg_we	1	+																
b/MyMips/reg_sel_1	0																	
b/MyMips/reg_sel_0	0																	H
b/MyMips/rd_sel	0	+																H
b/MyMips/pc_sel_1 b/MyMips/pc_sel_0	0	+																i
b/MyMips/det_o	o																	Ī
tb/MyMips/ext_res	32'h0000XXXX					FFFFFFF9		00000005				00000030	00000000			0000XXXX		
tb/MyMips/o_rs	32'h00000000			00000000				FFFFFFF3	00000000	00000005								#
tb/MyMips/o_rt tb/MyMips/pc4o	32'h00000000 32'hXXXXXXXX	0000000D		00000000			00000000		00400000	00400004		00000001		004000C8	100400000			
tb/MyMips/x	32'h00000000			0000000D			00400074				00000005			00400008	100400000			i
tb/MyMips/y	32'h00000000	00000000			00000000		00000004	100000000						00000000				i
tb/MyMips/shamt	5'h00	00					IF	00			1F	00					XX	
tb/MyMips/z_id	1																	
tb/MyMips/z_wb tb/MyMips/neg	0	H	-															
tb/MyMips/o	0	H				1		1										i
tb/MyMips/pc_1	0																	I
tb/MyMips/pc_0	0																	4
tb/MyMips/pc_en	U																	J
tb/MyMips/dmem_we tb/MyMips/halt	0																	i
tb/MyMips/reg sel	2'hX	3		0														i
b/MyMips/inst	32'hXXXXXXXX	010A5826	014A6027	10000005	00000000	1408FFF9	00000000	19C00005	00000000	1D20FFF9	00000000	08100030	00000000		50000000	XXXXXXX		j
b/MyMips/rd	5'hXX	OB	OC.	00		1F	00			1F	00					XX		
b/MyMips/ov_in	0																	H
:b/MyMips/ov_out :b/MyMips/o_ex	8'hXX	01	05	03	110	101	10	01	10	01	10	01	100	01		00		i
b/MyMips/o_alu		00000000												00000000				i
b/MyMips/o_shamt	5'hXX	00						00			1F	00					XX	j
b/MyMips/o_rsd_ex	32'hXXXXXXXX	00000000							FFFFFFF3	00000000	00000005							4
tb/MyMips/o_rsd_wb		00000006									HEFFEFF3	,00000000		00000000				#
tb/MyMips/o_rtd_ex tb/MyMips/o_imm	32'hXXXXXXXXX 32'hXXXXXXXXX	00000000			00000000			00000000	00000005	00000000	FFFFFFF	00000000		00000000			0000XXXX	j
tb/MyMips/o_wb	7'hXX			10	30000000	3000000							00000030			10		1
tb/MyMips/alu_wb	32'hXXXXXXXX	00000000					00000000		FFFFFFC	00000000	FFFFFFF3	00000000	00000005	00000000	00000001	00000000		
tb/MyMips/pc4_wb	32'hXXXXXXXX	00400088		00400090										004000C4	004000AC	004000B0	004000C4	10
/tb/MyMips/dmem_wb	32'hXXXXXXXX	00000006		1000	0110	1000	00000006 0110				00000000		0110		0001	0010		#
/tb/MyMips/control_in	16'h0010	0510																

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.



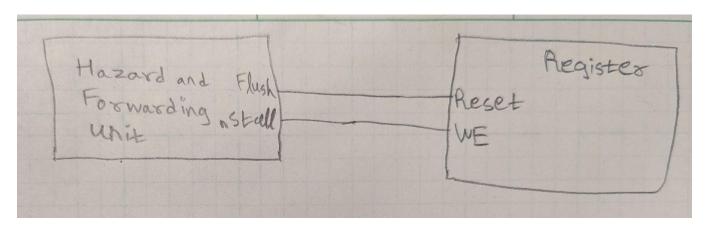
From 2840 to 2900 ns, and 3080 to 3140 ns, no-ops execute to avoid data-flow hazards. From 3020 to 3060, no-ops execute to avoid control flow hazards.

[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).

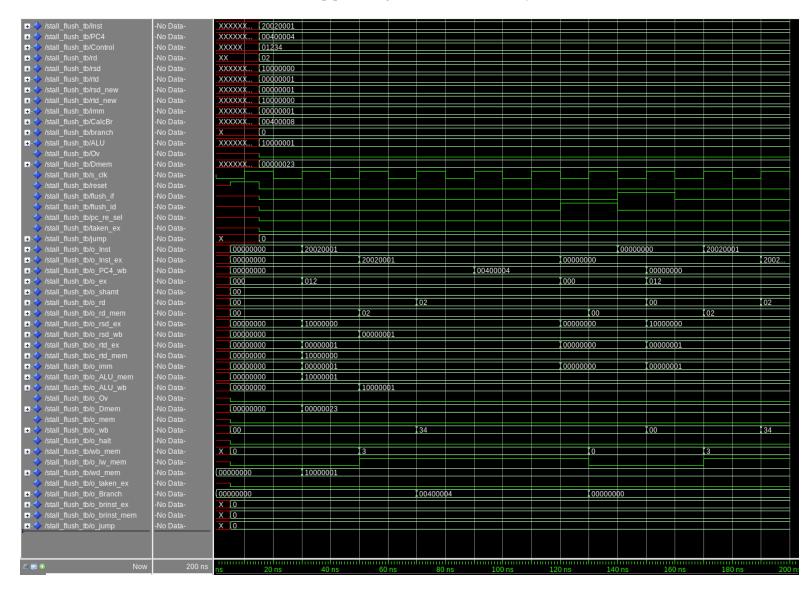
Max Frequency: 54.45 MHz

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

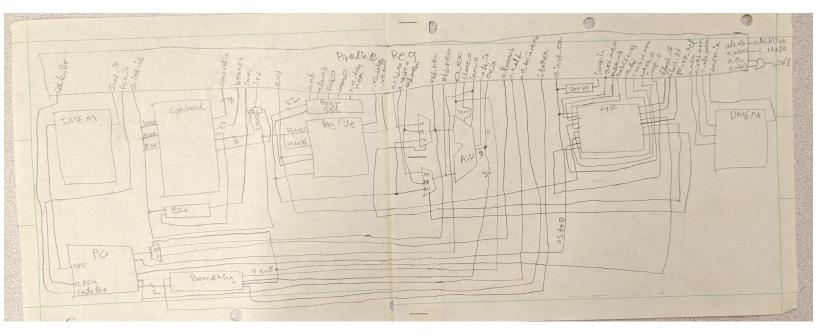
[2.c.i] list all instructions that may result in a non-sequential PC update and in which pipeline stage that update occurs.

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.

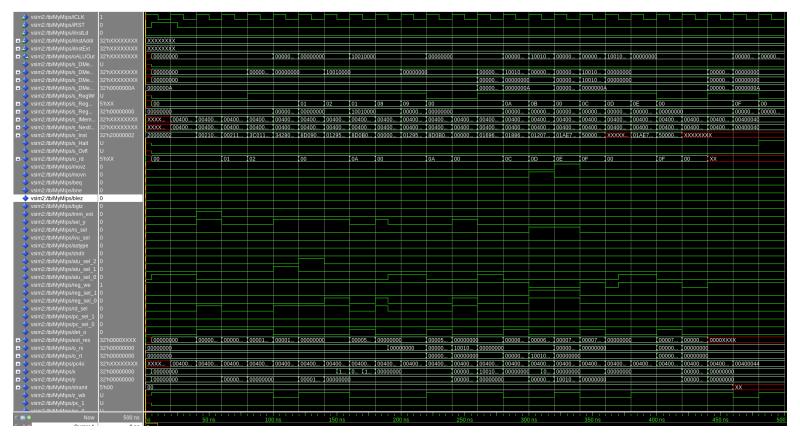
ALU op requires data from prev instruction

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.

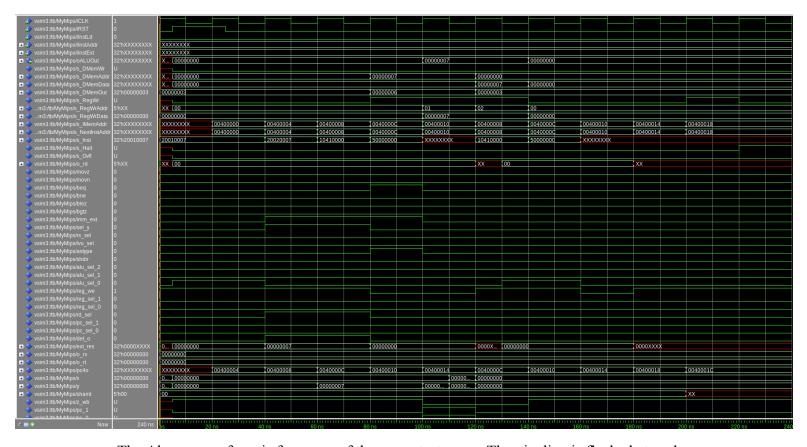


Data forwarded when possible, else flushing occurs. Matches the expected results therefore correct.

[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.

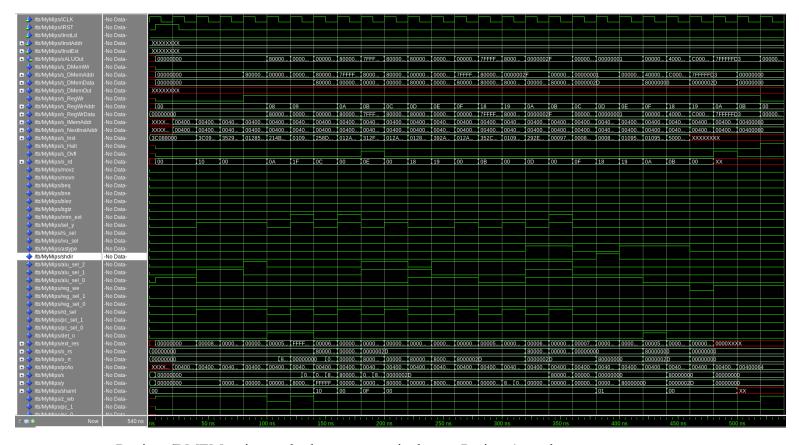
All branch and jump tests in base unit tests

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



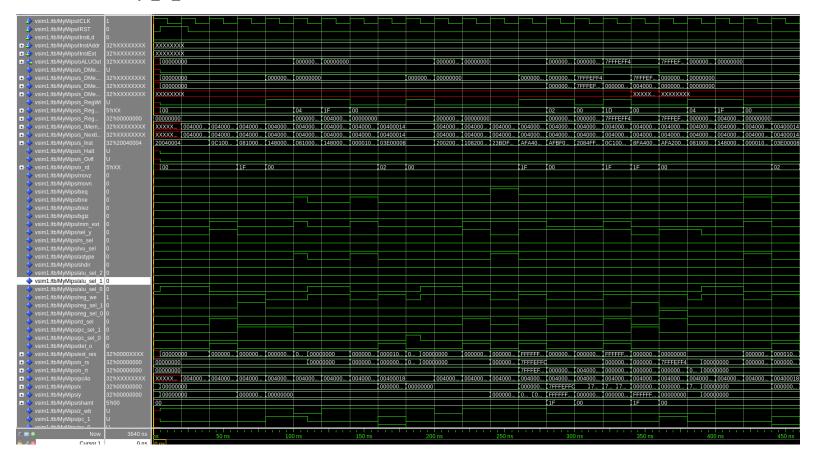
The Above waveform is from one of the many test cases. The pipeline is flushed at each control flow instruction and the right instructions are executed.

[2.e.iii] Proj1_base_test.s



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

Proj1_cf_tests.s



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 \Rightarrow Mux \Rightarrow ALU (Adder) \Rightarrow Program

Counter