MIPS Processor Report

Design Principles:

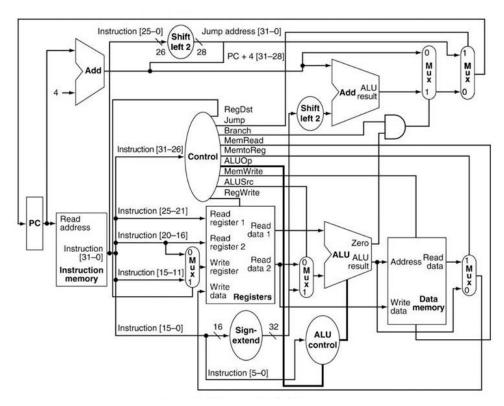


Figure 1, Single-cycle MIPS Processor

Our processor design is based upon that shown in Fig. 1. Each visible component is implemented as a VHDL component. Because there are so few components, generics are only utilized for the mux. Each ALU and left-shifter has its own unique code. The data memory, program counter, and registers are the only components whose processes are driven by the clock. All other components' processes are driven by signals output from other components.

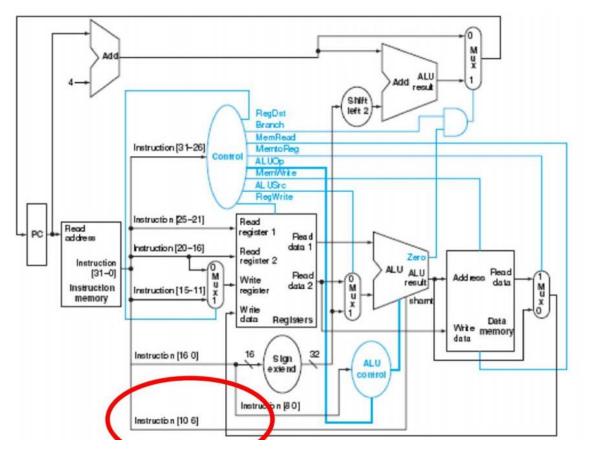


Figure 2, Single-cycle MIPS processor with sll implemented.

Modifications to the design were needed to implement *bne* and *sll*. To implement *bne*, an additional control signal (branch_s) is sent to the branching and-gate. If this signal is high, *beq* is used. If this signal is low, *bne* is used. To implement *sll*, instructions [10-6] are sent to the main ALU. An additional operation was added to the ALU to shift by *shamt*. This implementation can be seen in Fig. 2.

Challenges Faced:

Connecting the components was the most difficult part of the processor design. Once the components were connected, it became more difficult to debug. For complex issues, unit testing was used on each .vhd file to determine which component(s) was responsible.

Before implementing *sll*, incrementing i within the while-loop presented an issue. Because i must be multiplied by four to be used for addressing, *sll* seems like the natural solution. Without an implementation, we first thought to add i to the base address four times. This was inelegant and consumed too much of the instruction memory. Instead, we elected to use a human optimization, incrementing i by four on each iteration of the loop.

Part C MIPS Code:

```
add $s0, $zero, $zero; A = 0x0
addi $s1, $zero, 48 ; B = 0x30
addi $t0, $zero, 10; x = 10;
addi $t1, $zero, 20; y = 20;
addi $t2, $zero, $zero; i = 0;
addi $t3, $zero, 1 ; store 1 as a comparator and operand
addi $t4, $zero, 3; store 3 as operand
LOOP: ; while (y >= x)
slt $t5, $t1, $t0
beg $t3, $t5, END
add $t6, $s0, $t2 ; A[i]
lw $t6, 0($t6)
add $t6, $t6, $t0; A[i] + x
add $t6, $t6, $t1; (A[i] + x) + y
add $t7, $s1, $t2; B[i]
sw $t6, 0($t7) ; B[i] = A[i] + x + y
sub $t0, $t0, $t3; x -= 1
sub $t1, $t1, $t4; y -= 3
addi $t2, $t2, 4 ; i += 1;
j LOOP ; re-evaluate
END:
```

Figure 3, MIPS code for the given C program. Plain text available in Appendix A.

Waveform Screenshots:

] ▼	Msgs							
∮pc/ck	1							
- (pc/instr_31_0	32'h00000000	32'h8D	. 32'h8D110004	32h12110002	32'h02959820	32'h08000006	32'hAD130008	32'h00000000
	26'h0000000	26'h11	. 26'h1110004	26'h2110002	, 26'h2959820	26'h0000006	26'h1130008	26'h0000000
├-⟨> /pc/instr_31_26	6'h00	6h23		6'h04	(6'h00	6'h02	6'h2B	(6'h00
⊢ ♦ /pc/instr_25_21	5'h00	5'h08		5'h10	5'h14	5'h00	5'h08	5'h00
├-⟨> /pc/instr_20_16	5'h00	(5'h10	5h11		5'h15	5'h00	5'h13	5'h00
├ - /pc/instr_15_11	5'h00	(5'h00			5h13	5'h00		
├-⟨> /pc/instr_10_6	5'h00	(5'h00						
├-⟨> /pc/instr_15_0	16'h0000	16'h0000	16'h0004	16'h0002	16'h9820	16'h0006	16'h0008	16'h0000
├-⟨> /pc/instr_5_0	6'h00	(6'h00	6'h04	6'h02	(6'h20	6'h06	(6'h08	(6'h00
<pre>/pc/regdst_c</pre>	1							
<pre>/pc/jump_c</pre>	0							
/pc/branch_c	0							
<pre>/pc/branch_s_c</pre>	1							
<pre>/pc/memread_c</pre>	0							
/pc/memtoreg_c	0							
<pre>/pc/memwrite_c</pre>	0							
√ /pc/alusrc_c	0							
<pre>/pc/regwrite_c</pre>	1							
/pc/aluop_c	2'h2	(2'h0		(2'h1	(2'h2	2'h0		2'h2
	5'h00	(5'h10	5h11		5'h13	5'h00	[5h00	
/pc/read_data1	32'h00000000	(32'h0000		32'h00000005	32'h0000000E	32'h00000000		
/pc/read_data2	32'h00000000	(32'h00	. 132'h00000000	32'h00000004	32'h00000005	32'h00000000	32'h00000013	32'h00000000
/pc/signext_val	32'h00000000	(32'h00	. 32'h00000004	32'h00000002	32'h00009820	32'h00000006	32'h00000008	32'h00000000
/pc/alusrc2	32'h00000000		. 32'h00000004		32'h00000005	32'h00000000	32'h00000008	32'h00000000
/pc/aluctrl	4'h3	4h2		14'h6	, 4'h2	4'h2		4'h3
/pc/alures	32'h00000000		. 132'h00000004	32'h00000001	32'h00000013	32'h00000000	32'h00000008	32'h00000000
♦ /pc/aluzero	1							
/pc/read_data_m	32'h00000004	32'h00	. 132'h00000004					
/pc/memtoreg_val	32'h00000000		. 132'h00000004	32'h00000001	32'h00000013	32'h00000000	32'h00000008	32'h00000000
/pc/pc_inc_val	32'h00000020		. 132'h00000008	32'h0000000C	32'h00000010	32'h00000014	32'h0000001C	32'h00000020
	4'h0	4'h0						
/pc/jump shift val	32'h00000000		. 【32'h04440010	132'h08440008	132h0A566080	1 32'h00000018	32'h044C0020	32'h00000000
/pc/signext_shift_val	32'h00000000		. 132'h00000010	32'h00000008	32'h00026080	32'h00000018	32'h00000020	32'h00000000
/pc/jump_alu_val	32'h00000020		. 32'h00000018	32'h00000014	32'h00026090	32'h0000002C	32'h0000003C	32'h00000020
/pc/branch_sel	0	DE HOUT		32110000001	JEHOUGEOD30	- DENIOUS DEC	- DETROGOGOS C	021100000020
/pc/branch mux val	32'h00000020	32'h00	. 132'h00000008	(32'h000000C	(32'h00000010	32'h00000014	32'h0000001C	32'h00000020
/pc/next addr	32'h00000020		. 132h00000008	32'h000000C	32'h00000010	32'h00000014	32'h0000001C	32'h00000020
/pc/addr	32'h0000001C		. 32'h00000004	32'h00000008	1 32'h00000010	32'h00000010	32'h00000016	32h00000020
/pc/instr_mem/read	32'h0000001C		. 132h0000004	32'h0000008	32'h0000000C	32'h00000010	32'h00000018	32'h0000001C
/pc/instr_mem/instr	32'h00000000		. 132h0000004	32h12110002	32'h02959820	32'h08000006	32hAD130008	32h0000001C
⊢/pc/instr_mem/data	{32'h8D1000000}			004} {32h12110002} {3				
+-<> (0)	32'h8D 1000007	32'h8D1		0.111000270	211020702070211000	000007 021102303022	. 19211AD 1300007 32	
+ (1)	32'h8D110004	32'h8D1						
+- (2)	32'h12110002	32'h121						
(2) +-(3)	32'h02959820	32'h029						
+- (4)	32'h08000006	32h0800						
- • · · ·		32110500						
Now Now	1000 ns	ns	100 ns	200 ns	300 ns	400 ns	500 ns	600 ns
Cursor 1	634 ns							

Figure 4, Screen capture of waveforms for part A.

≟ ▼	Msgs												
	32'h00000000	(32'h00	32'h00000	0004			32'h00000	0005	32'h00000	000	32'h00000	8000	32'h00
 ∳ /pc/alu_c/input0	32'h00000000	(32'h00000	000		32'h00000	005	32'h00000	00E	32'h00000	000			
🛶 /pc/alu_c/input1	32'h00000000	32'h00	32'h00000	0004			32'h00000	005	32'h00000	000	32'h00000	8000	32'h00
 - ♦ /pc/alu_c/alu_ctrl	4'h3	(4'h2			(4'h6		4'h2		4h2				4h3
├�� /pc/alu_c/shamt	5'h00	5'h00											
📥 /pc/alu_c/output	32'h00000000	32'h00	32'h00000	0004	32'h00000	001	32'h00000	013	32'h00000	000	32'h00000	8000	32'h00
/pc/alu_c/zero	1												l —
├�� /pc/main_mem/addr	32'h00000000	32'h00	32'h00000	0004	32'h00000	001	32'h00000	013	32'h00000	000	32'h00000	8000	32'h00
├�� /pc/main_mem/writ	32'h00000000	32'h00	32'h00000	0000	32'h00000	004	32'h00000	005	32'h00000	000	32'h00000	013	32'h00
<pre>/pc/main_mem/Mem</pre>	0												
<pre>/pc/main_mem/Mem</pre>	0												
<pre>/pc/main_mem/ck</pre>	0												
	32'h00000004	32'h00	32'h00000	0004									
	{32'h00000005}	{32'h0000	0005} {32	n00000004	} {32'h00000	000} {32¦	100000000}	{32'h00000	000} {32'h0	{0000000}	{32'h000	{32'h0000	00005} {
<u>+</u> > (0)	32'h00000005	32'h00000	005										
<u>+</u> - (1)	32'h00000004	32'h00000	004										
<u>+</u> (2)	32'h00000013	32'h00000	000									32'h0000	0013
<u>+</u> (3)	32'h00000000	32'h00000	000										
	32'h00000000	32'h00000	000										
<u>+</u> - \diamondsuit (5)	32'h00000000	32'h00000	000										
<u>+</u> > (6)	32'h00000000	32'h00000	000										
<u>+</u> -🔷 (7)	32'h00000000	32'h00000	000										
<u>+</u> > (8)	32'h00000000	32'h00000	000										
<u>+</u> > (9)	32'h00000000	32'h00000	000										
<u>+</u> (10)	32'h00000000	32'h00000	000										
<u>+</u> (11)	32'h00000000	32'h00000	000										
<u>+</u> (12)	32'h00000000	32'h00000	000										
<u>+</u> - \diamondsuit (13)	32'h00000000	32'h00000	000										
<u>+</u> (14)	32'h00000000	32'h00000	000										
<u>+</u> (15)	32'h00000000	32'h00000	000										
<u>+</u> (16)	32'h00000000	32'h00000	000										
<u>+</u> > (17)	32'h00000000	32'h00000	000										
<u>+</u> (18)	32'h00000000	32'h00000	000										
<u>+</u> (19)	32'h00000000	32'h00000	000										
<u>+</u> > (20)	32'h00000000	32'h00000	000										
<u>+</u> -🔷 (21)	32'h00000000	32'h00000	000										
<u>+</u> > (22)	32'h00000000	32'h00000	000										
<u>+</u> (23)	32'h00000000	32'h00000	000										
+ - (24)	32'h00000000	32'h00000	000										
<u>+</u> > (25)	32'h00000000	32'h00000	000										
<u>+</u> (26)	32'h00000000	32'h00000	000										
<u>+</u> > (27)	32'h00000000	32'h00000	000										
<u>+</u> > (28)	32'h00000000	32'h00000	000										
+ - - (29)	32'h00000000	32'h00000	000										
<u>+</u> > (30)	32'h00000000	32'h00000	000										
<u>+</u> > (31)	32'h00000000	32'h00000	000										
🛶 /pc/mem_mux/input0	32'h00000000	32'h00	32'h00000	0004	32'h00000	001	32'h00000	013	32'h00000	000	32'h00000	8000	32'h00
Now	1000 ns	111111111		0 ns	200	ns ns		0 ns		ns		0 ns	duuu
		15	10	0.115	200	115	30	U-115		7.115	50	0.115	

Figure 5, Screen captures of waveforms for memory in part A.

- /pc/regs/read_data2	32'h00000000	(32'h00 (32'h00000000	32'h00000004	32'h00000005	32'h00000000	32'h00000013	32'h0000000
/pc/regs/data_mem	{32'h000000000}	{32'h0					
<u>+</u> > (0)	32'h00000000	32'h00000000					
+- (1)	32'h00000000	32'h00000000					
+ (2)	32'h00000000	32'h00000000					
+ (3)	32'h00000000	32'h00000000					
+-4 (4)	32'h00000000	32'h00000000					
+ (5)	32'h00000000	32'h00000000					
±- ♦ (6)	32'h00000000	32'h00000000					
+-4 (7)	32'h00000000	32'h00000000					
+ (8)	32'h00000000	32'h00000000					
+ (9)	32'h00000000	32'h00000000					
+ (10)	32'h00000000	32'h00000000					
[⊥] +- ♦ (11)	32'h00000000	32'h00000000					
±- → (12)	32'h00000000	32'h00000000					
±- ♦ (13)	32'h00000000	32'h00000000					
[⊤] - → (14)	32'h00000000	32'h00000000					
±- ♦ (15)	32'h00000000	32'h00000000					
±- ♦ (16)	32'h00000005	32'h00 (32'h00000005					
+ (17)	32'h00000004	32'h00000000	32'h00000004				
+ (18)	32'h00000000	32'h00000000					
+ (19)	32'h00000013	32'h00000000			32'h00000013		
+ (20)	32'h0000000E	32'h0000000E					
+ (21)	32'h00000005	32'h00000005					
+ (22)	32'h00000000	32'h00000000					
+ (23)	32'h00000000	32'h00000000					
+ (24)	32'h00000000	32'h00000000					
+ (25)	32'h00000000	32'h00000000					
<u>+</u> > (26)	32'h00000000	32'h00000000					
+ - (27)	32'h00000000	32'h00000000					
<u>+</u> - \checkmark (28)	32'h00000000	32'h00000000					
<u>+</u> - (29)	32'h00000000	32'h00000000					
<u>+</u> - \checkmark (30)	32'h00000000	32'h00000000					
+ - - (31)	32'h00000000	32'h00000000					
-💠 /pc/pc_add/cur_addr	32'h0000001C	32'h00 , 32'h00000004	32'h00000008	32'h0000000C	32'h00000010	32'h00000018	32'h0000001
-💠 /pc/pc_add/next_addr	32'h00000020	(32'h00 , 32'h00000008	32'h0000000C	32'h00000010	32'h00000014	32'h0000001C	32'h0000002
√pc/sign_e/im	16'h0000	16'h0000 16'h0004	16'h0002	16'h9820	16'h0006	16'h0008	16'h0000
-💠 /pc/sign_e/im_ext	32'h00000000	(32'h00 32'h00000004	32'h00000002	32'h00009820	32'h00000006	32'h00000008	32'h0000000
🥠 /pc/sign_e_s/sign_e	32'h00000000	(32'h00 (32'h00000004	32'h00000002	32'h00009820	32'h00000006	32'h00000008	32'h0000000
-👍 /pc/sign_e_s/sign_e	32'h00000000	(32'h00 (32'h00000010	32'h00000008	32'h00026080	32'h00000018	32'h00000020	32'h0000000
/pc/j_addr/sign_ext	32'h00000020	(32'h00 32'h00000008	32'h0000000C	32'h00000010	32'h00000014	32'h0000001C	32'h0000002
√pc/j_addr/pc	32'h00000000	(32'h00 (32'h00000010	32'h00000008	32'h00026080	32'h00000018	32'h00000020	32'h0000000
-🔙 /pc/j_addr/branch	32'h00000020	(32'h00 , 32'h00000018	32'h00000014	32'h00026090	32'h0000002C	32'h0000003C	32'h0000002
√pc/b_mux/input0	32'h00000020	(32'h00 (32'h00000008	32'h0000000C	32'h00000010	32'h00000014	32'h0000001C	32'h000000
/pc/b_mux/input1	32'h00000020	(32'h00 (32'h00000018	32'h00000014	32'h00026090	32'h0000002C	32'h0000003C	32'h000000
<pre>/pc/b_mux/sel</pre>	0						
-👍 /pc/b_mux/output	32'h00000020	(32'h00 , 32'h00000008	32'h000000C	32'h00000010	32'h00000014	32'h0000001C	32'h000000
	1000 ns		damandaman	la a a a a a a a a a a a a a a a a a a	400 ns	500 ns	600 r

Figure 6, Screen captures for waveforms of registers in part A.

· ·	Msgs										
/pc/ck	0										
- √ /pc/instr_31_0	32'h00000000	32'h8D	32'h8D110	0004	32h1211000	2 32'h	02959822	32'hAD13	8000	32'h00000000	
- √ /pc/instr_25_0	26'h0000000	26'h11	26'h11100	04	26'h2110002	, 26'h	2959822	26'h1130	008	26'h0000000	
	6'h00	6'h23			6'h04	(6'h0	0	(6'h2B		(6'h00	
⊢ ♦ /pc/instr_25_21	5'h00	(5'h08			5h10	[5h1	4	(5'h08		5'h00	
- ♦ /pc/instr 20 16	5'h00		5h11			5'h1		5h13		5'h00	
	5'h00	(5'h00				5'h1		[5'h00			
- ♦ /pc/instr_10_6	5'h00	(5'h00									
- ♦ /pc/instr_15_0	16'h0000		16'h0004		16'h0002	16'h	9822	16'h0008		16'h0000	
/pc/instr_5_0	6'h00	_	6'h04		6'h02	6'h2		(6'h08		6'h00	
♦ /pc/regdst_c	1										
√pc/jump_c	0										
♦ /pc/branch c	0										
♦ /pc/branch s c	1				1						
/pc/memread_c	0				-						
✓ /pc/memtoreg_c	0										
✓ /pc/memwrite_c	0										
✓ /pc/memwrite_c ♦ /pc/alusrc_c	0										
✓ /pc/ausic_c ✓ /pc/regwrite_c	1										
→ /pc/regwrite_c → /pc/aluop c	1 2'h2	(2'h0			2h1	2'h2		2'h0		2'h2	
	5'h00		5'h11		, ZIII	. 2n2		1 5'h00		/2112	
X 11 1T. T.	32'h00000000	(32'h00000			32'h0000000		0000000E	32'h0000	0000		
	32'h00000000			000						2250000000	
	32'h00000000		32'h00000		32'h0000000		00000005	32'h0000		32'h00000000	
/pc/signext_val			32'h00000		32'h0000000		00009822	32'h0000		32'h00000000	
/pc/alusrc2	32'h00000000 4'h3		32'h00000	004		. 32n	00000005	32'h0000	0008	32'h00000000	
/pc/aluctrl		(4h2			4'h6			4'h2		4h3	
/pc/alures	32'h00000000	32nuu	32'h00000	0004	32'h0000000	0 32h	00000009	32'h0000	0008	32'h00000000	
/pc/aluzero	1 32'h00000004	/!!									
/pc/read_data_m		(32'h00000						V 11		V	
/pc/memtoreg_val	32'h00000000	(32'h00000			32'h0000000		00000009	32'h0000		32'h00000000	
/pc/pc_inc_val	32'h00000024		32'h00000	8000	32'h0000000	C 32h	00000018	32'h0000	001C	(32'h00000020	32'h000
⊢ ♦ /pc/pc_31_28	4'h0	(4'h0									
	32'h00000000		32'h04440		32'h0844000		0A566088	32'h044C		32'h00000000	
/pc/signext_shift_val	32'h00000000		32'h00000		32'h0000000		00026088	32'h0000		32'h00000000	
⊢ ♦ /pc/jump_alu_val	32'h00000024	32'h00	32'h00000	018	32'h0000001	4 32'h	000260A0	32'h0000	003C	32'h00000020	32'h000
<pre>/pc/branch_sel</pre>	0										
⊢ ♦ /pc/branch_mux_val	32'h00000024		32'h00000		32'h0000001		00000018	32'h0000		32'h00000020	32'h000
	32'h00000024		32'h00000		32'h0000001		00000018	32'h0000		32'h00000020	32'h000
⊢ ♦ /pc/addr	32'h00000020		32'h00000		32'h0000000		00000014	32'h0000		32'h0000001C	32'h000
-🍫 /pc/instr_mem/read	32'h00000020		32'h00000		32'h0000000		00000014	32'h0000		32'h0000001C	32'h000
-🔷 /pc/instr_mem/instr	32'h00000000		32h8D110		32'h1211000		02959822	32'hAD13		32'h00000000	
⊢🔷 /pc/instr_mem/data	{32'h8D100000}	{32'h8D10	00000} {321	8D110004	f} {32'h1211000)2} {32'h02959	9820} {32'h08	3000006} {32'h	02959822	} {32'hAD130008} {32	?h00000000}
+ - (0)	32'h8D 100000	32'h8D100	0000								
+ - (1)	32'h8D110004	32'h8D110	0004								
<u>+</u> -🔷 (2)	32'h12110002	32'h12110	0002								
<u>+</u> -🔷 (3)	32'h02959820	32'h02959	9820								
<u>+</u> -🔷 (4)	32'h08000006	32'h08000	0006								
Now	1000 ns			haman		mondon	nundunu		duum		
				0 ns	200 ns		300 ns	40	10 ns	500 ns	

Figure 7, Screen captures for waveforms in part B.

<u> </u>	Ms	sgs (
+-4> /pc/alu_c/alu_ctrl	-No Data-	(4'h2	(4'h6		ľ 4h2	14h3
+-4 /pc/alu_c/shamt	-No Data-	(5'h00	1110		1112	1113
+- /pc/alu_c/output	-No Data-	(32'h00 (32'h00000004	32'h00000000	32'h00000009	32'h00000008	32'h00000000
/pc/alu_c/zero	-No Data-			52.13333333	525555555	
/pc/main_mem/addr	-No Data-	(32'h00 (32'h00000004	. 32'h00000000	32'h00000009	32'h00000008	32'h00000000
+-4 /pc/main_mem/writ	-No Data-	(32'h00 32'h00000000	. 32'h00000004	32'h0000005	32'h00000009	1 32'h00000000
/pc/main_mem/Mem	-No Data-					
/pc/main_mem/Mem	-No Data-]	
<pre>/pc/main_mem/ck</pre>	-No Data-					
pc/main_mem/read	-No Data-	(32'h00000004				
/pc/main_mem/data	-No Data-	{32'h00000004} {32'h000000	04} {32'h00000000} {32'h	00000000} {32'h00000	000} {32 {32'h000	000004} {32'h00000004} {3
+ - (0)	-No Data-	32'h00000004				
<u>+</u> - \checkmark (1)	-No Data-	32'h00000004				
+ - (2)	-No Data-	32'h00000000			32'h0000	00009
±- (3)	-No Data-	32'h00000000				
±- (4)	-No Data-	32'h00000000				
<u>+</u> - ♦ (5)	-No Data-	32'h00000000				
<u>+</u> - (6)	-No Data-	32'h00000000				
± - ♦ (7)	-No Data-	32'h00000000				
± - → (8)	-No Data-	32'h00000000				
. (9)	-No Data-	32'h00000000				
.	-No Data-	32'h00000000				
<u>+</u> > (11)	-No Data-	32'h00000000				
± - → (12)	-No Data-	32'h00000000				
± - → (13)	-No Data-	32'h00000000				
± - → (14)	-No Data-	32'h00000000				
‡ - ♦ (15)	-No Data-	32'h00000000				
□ - ◇ (16)	-No Data-	32'h00000000				
□ - ◇ (17)	-No Data-	32'h00000000				
 - ♦ (18)	-No Data-	32'h00000000				
<u>+</u> - (19)	-No Data-	32'h00000000				
<u>+</u> -4 (20)	-No Data-	32'h00000000				
<u>+</u> -4 (21)	-No Data-	32'h00000000				
<u>+</u> (22)	-No Data-	32'h00000000				
<u>+</u> (23)	-No Data-	32'h00000000				
<u>+</u> (24)	-No Data-	32'h00000000				
<u>+</u> - \(\) (25)	-No Data-	32'h00000000				
(26)	-No Data-	32'h00000000				
± (27)	-No Data-	32'h00000000				
(28)	-No Data-	32'h00000000				
(29)	-No Data-	32'h00000000				
±- → (30)	-No Data-	32'h00000000				
(31)	-No Data-	32'h00000000	Vooli oooooo	V acil account	V apli pagazza	Vasil occoord
/pc/mem_mux/input0	-No Data-	(32'h00) 32'h00000004	, 32'h00000000	32'h00000009	32'h00000008	, 32'h00000000
/pc/mem_mux/input1	-No Data-	(32'h00000004				
/pc/mem_mux/sel	-No Data-	22"-000000-4	7221-0000000	22/200000000	225-0000000	225-0000000
II /pc/mem_mux/output	-No Data-	(32'h00000004	32'h00000000	32h00000009	32'h00000008	32'h00000000
△ 🗊 💿 Now	1000 n	ns 100 ns	200 ns	300 ns	400 ns	500 ns
6 ✓ Cursor 1	1204 n					
	-					

Figure 8, Screen capture for waveforms of memory in part B.

<u> </u>	l Msgs	;						
pc/regs/reg_reg2 pc/regs/reg_reg2	-No Data-	(5'h10	. 5'h11		I 5'h15	I5h13	I 5'h00	
+-4 /pc/regs/write_reg	-No Data-	(5h10	5h11		5h13	15h00	131100	
+-4 /pc/regs/write_data	-No Data-	(32'h0000		X 32'h00000000	32'h00000009	32'h00000008	32'h00000000	
+- /pc/regs/read_data1	-No Data-	(32'h0000		32'h00000004	32'h0000000E	32'h00000000	, 32110000000	
+- /pc/regs/read_data2	-No Data-		1 32'h00000000	132h00000004	1 32'h00000005	X 32h00000009	I 32'h00000000	
	-No Data-		{32'h000000000} {					00000} {32'b000000
±- (0)	-No Data-	32'h0000		(1521100000000) [52	1000000007 [521100111	, 15211000000001 152	1000000007 02110000	35007 (5211050550
T (1)	-No Data-	32'h0000						
+- (2)	-No Data-	32'h0000						
±- → (3)	-No Data-	32'h0000						
±- → (4)	-No Data-	32'h0000						
+- (5)	-No Data-	32'h0000						
+ - - (6)	-No Data-	32'h0000						
T (7)	-No Data-	32'h0000						
+- (8)	-No Data-	32'h0000						
÷- (9)	-No Data-	32'h0000						
+- (10)	-No Data-	32'h0000						
+- (11)	-No Data-	32'h0000						
+- (12)	-No Data-	32'h0000						
+- (13)	-No Data-	32'h0000						
+- (14)	-No Data-	32'h0000						
+- (15)	-No Data-	32'h0000						
+- (16)	-No Data-		32'h0000004					
+- (17)	-No Data-	32'h0000		32'h00000004				
+- (18)	-No Data-	32'h0000		, 32110000000 1				
+- (19)	-No Data-	32'h0000				(32'h00000009		
+- (20)	-No Data-	32'h0000				, 52110000000		
±- (21)	-No Data-	32'h0000						
+- (22)	-No Data-	32'h0000						
+- (23)	-No Data-	32'h0000						
+- (24)	-No Data-	32'h0000						
±- (25)	-No Data-	32'h0000						
+- (26)	-No Data-	32'h0000						
±- - - (27)	-No Data-	32'h0000						
±- (28)	-No Data-	32'h0000						
T- (29)	-No Data-	32'h0000						
T-4 (30)	-No Data-	32'h0000						
T-4 (31)	-No Data-	32'h0000						
	-No Data-		. X 32'h00000004	32'h00000008	32'h00000014	132h00000018	32'h0000001C	32'h00000020
	-No Data-		. 32'h00000008	1 32'h0000000C	32'h00000018	32'h0000001C	I 32'h00000020	32'h00000024
	-No Data-) 16'h0004	16'h0002	16'h9822	16'h0008	16'h0000	
+- /pc/sign_e/im_ext	-No Data-		32'h00000004	32'h00000002	32'h00009822	32'h00000008	32'h00000000	
/pc/sign_e_s/sign_e	-No Data-		32'h00000004	32'h00000002	32'h00009822	32'h00000008	32'h00000000	
#_	-No Data-		32'h00000010	32'h00000008	32'h00026088	32'h00000020	32'h00000000	
+ / /pc/j_addr/sign_ext	-No Data-		32'h00000008	32'h0000000C	32'h00000018	32'h0000001C	32'h00000020	32'h00000024
#/pc/j_addr/pc	-No Data-		32'h00000010	32'h00000008	32'h00026088	32'h00000020	32'h00000000	
+-4 /pc/j_addr/branch	-No Data-		32'h00000018	32'h00000014	32'h000260A0	32'h0000003C	32'h00000020	32'h00000024
			1 1111111111111111111111111111111111111			danaan laan a		
△ P • Now	1000 ns	ns	100 ns	200 ns	300 ns	400 ns	500 ns	600 ns
6 ✓ Cursor 1	1162 ns							

Figure 9, Screen capture of waveforms of registers in part B.

		2000
Wave - Default	Msgs	
*	Misgs	
/pc/alu_c/zero	0	
■ → /pc/main_mem/address	32'h00000001)\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
II → /pc/main_mem/write_data	32'h00000004	_ DCDCCCD[CCCD]CCCD[DCCCD[DCCCD[DCCCD]DCCCD]DCCCD[DCCCD[DCCCD]DCCCD[DCCCD[DCCCD]DCDCD]DCDCD[DCCCD]DCCCDDDDDDDD
<pre>/pc/main_mem/MemWrite</pre>	0	
/pc/main_mem/MemRead	0	
<pre>/pc/main_mem/ck</pre>	0	ռումիտումիտումիտումիտումիտումիտումիտումիտ
II — (/pc/main_mem/read_data	32'h00000002	3 (32h0000b001) (32h00000001) (32h0000b001) (32h0b000002) (32h00000002) (32h00000002
	{32'h00000001}	32h0000) (32h00000001)) (32h0000001)) (32h00000001)) (32h00000001)) (32h00000001))
± - ◆ (0)	32'h00000001	32/h00000001
★ - ◇ (1)	32'h00000002	32'h00000002
± - → (2)	32'h00000003	32'h00000003
± - ♦ (3)	32'h00000004	32h00000004
± - ◇ (4)	32'h00000005	32'h00000005
± - ◇ (5)	32'h00000006	32/100/200006
± - ◇ (6)	32'h00000007	32/h00000007
±- → (7)	32'h00000000	32'h0000000
±- → (8)	32'h00000000	32'h0000000
± - ◇ (9)	32'h00000000	32'h00000000
± - ◇ (10)	32'h00000000	32'h00000000
<u>+</u> - → (11)	32'h00000000	32'h0000000
±- → (12)	32'h0000001F	32'h00000), 32'h0000001F
+ - (13)	32'h0000001C	32'h00000000
± - ♦ (14)	32'h00000019	32/h0000000
<u>+</u> - ((15)	32'h00000016	32'h0000000
<u>+</u> (16)	32'h00000013	32'h0000000 (32'h00000013
<u>+</u> -🔷 (17)	32'h00000010	32'h00000000 (32'h00000010
<u>+</u> -4 (18)	32'h00000000	32'h0000000
<u>+</u> - (19)	32'h00000000	32'h00000000
±- → (20)	32'h00000000	32'h0000000
<u>+</u> - - (21)	32'h00000000	32'h00000000
– - → (22)	32'h00000000	32'h00000000
– - ♦ (23)	32'h00000000	32'h0000000
<u>+</u> -4 (24)	32'h00000000	32/h00000000
<u>+</u> - (25)	32'h00000000	32h0000000
±- 4 (26)	32'h00000000	32h0000000
<u>+</u> - 4 (27)	32'h00000000	32h0000000
<u>+</u> - (28)	32'h00000000	32h0000000
±- (29)	32'h00000000	32h0000000
±- (30)	32'h00000000	32h0000000
+ (31)	32'h00000000	32'h00000000

Figure 10, Screen capture for waveforms of memory in part C.

▼	M:	lsgs
<pre>/pc/regs/write_reg</pre>	-No Data-	- I saaxik aadiaaatiaa xibaaatiaaat saadiaaaataa xibaaatiaa kabaaatiaa baadiaaati xaadiaaati.
/pc/regs/write_data	-No Data-	and a substantial and the contraction and the
👍 /pc/regs/read_data1	-No Data-	
/pc/regs/read_data2	-No Data-	
<pre>/pc/regs/data_mem</pre>	-No Data-	\(\) \(\) \(\) \(\) \(\) \(\) \(\) \(\)
<u>+</u> -◆ (0)	-No Data-	32'h00000000
± - ♦ (1)	-No Data-	32'h0000000
±- (2)	-No Data-	32'h00000000
±- (3)	-No Data-	32'h00000000
<u>+</u> - \checkmark (4)	-No Data-	32h00000000
±- (5)	-No Data-	32'h00000000
<u>+</u> - \spadesuit (6)	-No Data-	32h00000000
±- → (7)	-No Data-	32'h00000000
<u>+</u> -◆ (8)	-No Data-	32h0000000A (32h0000009 (32h00000008 (32h00000007 (32h00000006 (32h00000005 (32h00000004
±- (9)	-No Data-	32h00000014 (32h00000011 (32h0000000E (32h0000000B (32h0000000B (32h00000005 (32h00000002
<u>+</u> - (10)	-No Data-	32h0000000
±- (11)	-No Data-	32h00000001
<u>+</u> - ♦ (12)	-No Data-	X32hoboooo3
<u>+</u> -🔷 (13)	-No Data-	32'h00000000 (32'h00000000
<u>+</u> - ♦ (14)	-No Data-	32h \ \ \ \ 32\\00000001\frac{1}{2}\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \
<u>+</u> - (15)	-No Data-	32h0000 (32h00000030 (32h00000034 (32h00000038 (32h0000003C (32h0000040 (32h000000444
<u>+</u> - ♦ (16)	-No Data-	32'h00000000
<u>+</u> - (17)	-No Data-	32'h00000030
<u>+</u> - \langle (18)	-No Data-	32'h00000000
<u>+</u> - (19)	-No Data-	32'h00000000
<u>+</u> - ♦ (20)	-No Data-	32'h00000000
<u>+</u> - (21)	-No Data-	32'h00000000
<u>+</u> - → (22)	-No Data-	32/100/00000
<u>+</u> - (23)	-No Data-	32/100/00000
<u>+</u> - ♦ (24)	-No Data-	32h00000000
<u>+</u> - (25)	-No Data-	32/100/00000
<u>+</u> - \checkmark (26)	-No Data-	32h00000000
<u>+</u> - \(\) (27)	-No Data-	32h00000000
<u>+</u> - \checkmark (28)	-No Data-	32h00000000
<u>+</u> - \checkmark (29)	-No Data-	32h00000000
+- (30)	-No Data-	32'h00000000

Figure 11, Screen capture for waveforms of registers in part C.

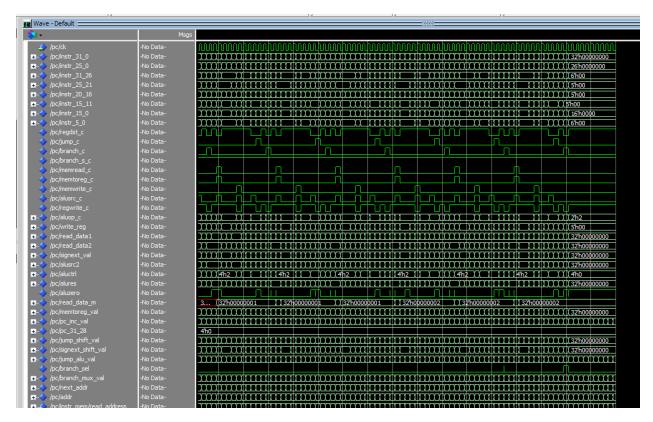


Figure 12, Screen capture of signal waveforms in part C.

Question Answers:

Part A:

Final Values:

\$t0 = 0

\$s0 = 5

\$s1 = 4

\$s3 = 19

\$s4 = 14

\$s5 = 5

0x00000000: 5

0x00000004: 4

0x00000008: 19

Part B:

Final Values:

\$t0 = 0

\$s0 = 4

\$s1 = 4

\$s3 = 9

\$s4 = 14

\$s5 = 5

0x00000000: 4

0x00000004: 4

0x00000008: 9

Part C: Final Register Values: x = 4 (in register \$t0) y = 2 (in register \$t1) i = 24 (in register \$t2) Final Data Memory Values: 0x00000000: 1 0x000000004: 2 0x000000008: 3 0x000000001: 5 0x000000014: 6 0x000000018: 7

0x00000034: 28

0x00000038: 25

0x0000003C: 22

0x00000040: 19

0x00000044: 16

81 cycles were executed to complete part C.

```
Appendix A:
add $s0, $zero, $zero; A = 0x0
addi $s1, $zero, 48; B = 0x30
addi $t0, $zero, 10; x = 10;
addi $t1, $zero, 20; y = 20;
addi $t2, $zero, $zero; i = 0;
addi $t3, $zero, 1; store 1 as a comparator and operand
addi $t4, $zero, 3; store 3 as operand
LOOP: ; while (y \ge x)
slt $t5, $t1, $t0
beq $t3, $t5, END
add $t6, $s0, $t2; A[i]
lw $t6, 0($t6)
add $t6, $t6, $t0; A[i] + x
add $t6, $t6, $t1; (A[i] + x) + y
add $t7, $s1, $t2; B[i]
sw $t6, 0($t7); B[i] = A[i] + x + y
sub $t0, $t0, $t3; x -= 1
sub $t1, $t1, $t4; y -= 3
addi $t2, $t2, 4; i += 1;
j LOOP ; re-evaluate
```

END: