

Lab 7 Post lab – Simplified DLX

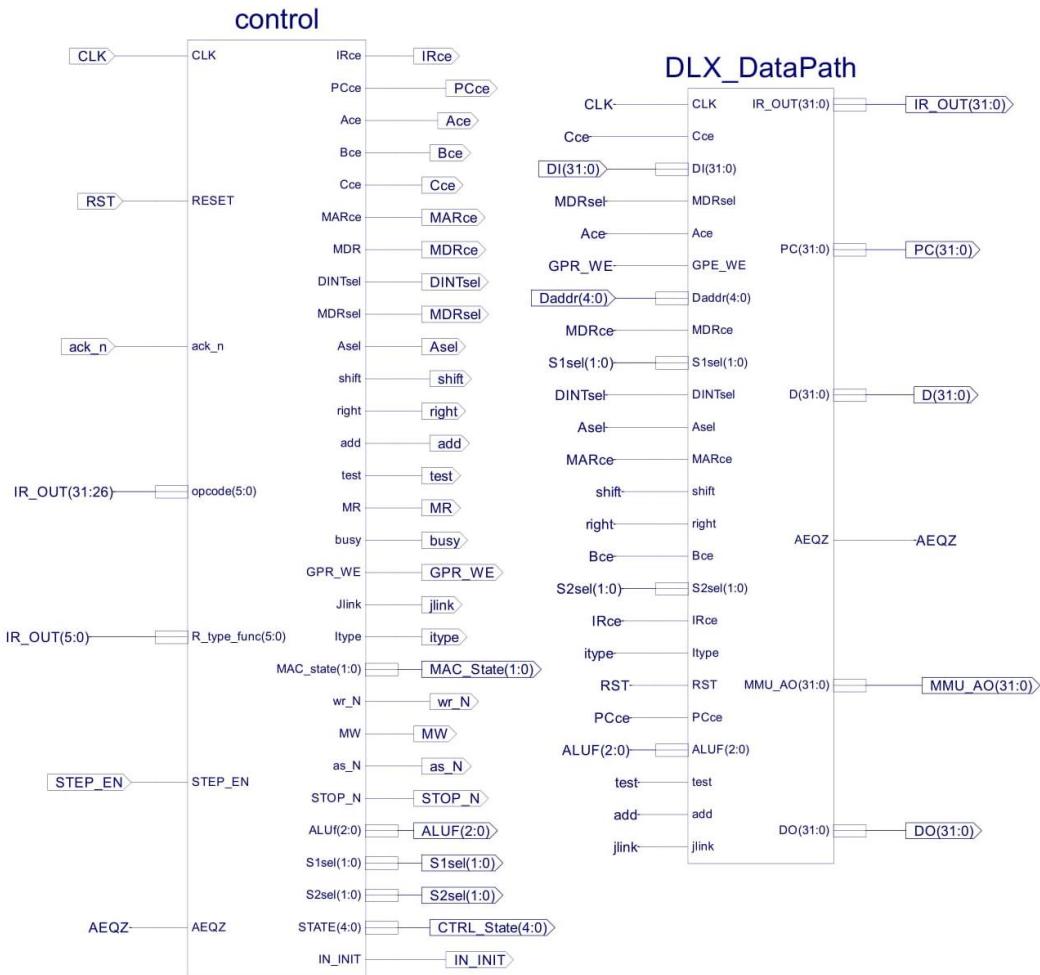
Yohai Shiloh

Yarin Koren

Group: B2

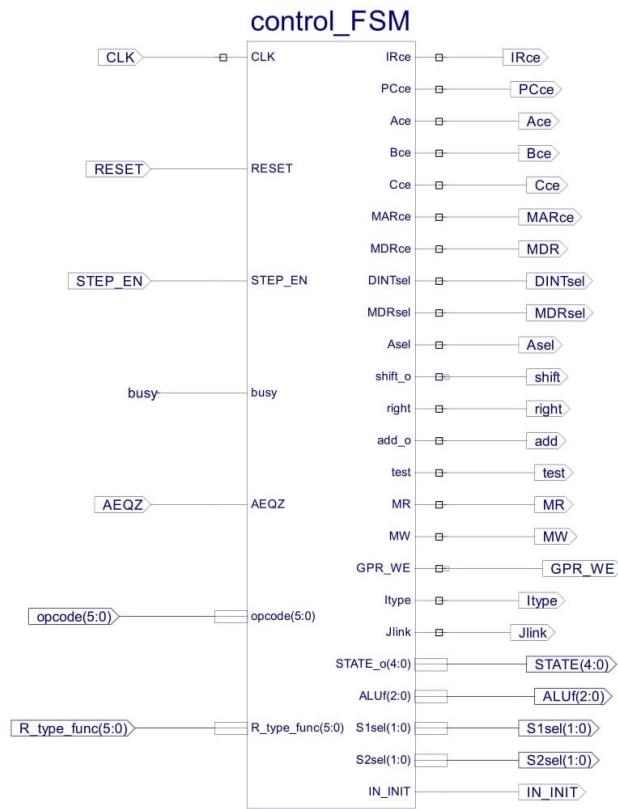
DLX – Top Level

The scheme of the connections between the data path of the processor and the control unit(including the MAC state machine as part of its symbol) attached below:



The Control

Control Top level schematic:



Note: The MAC was implemented the same as in the previous lab and therefore will not be detailed again.

Finite State Machine Verilog code:

The Finite State Machine of the control was written in Verilog(code attached below). The FSM code is built of four main parts – parameters definitions, states transition definition, sequential part and combinational part. The first defines toe parameters we would use in the code to make it more readable, and includes the states, paths, instructions etc. The second includes the conditions for transitions from one state to another, the third take care of resets and updating the states according to the clock and the fourth define the signals the FSM would have to generate to control the MAC, the data-path and ensure its right functionality.

```
1 `timescale 1ns / 1ps
2 ///////////////////////////////////////////////////////////////////
3 // Company:
4 // Engineer:
5 //
6 // Create Date: 23:39:45 07/06/2024
7 // Design Name:
8 // Module Name: control_FSM
9 // Project Name:
10 // Target Devices:
11 // Tool versions:
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 ///////////////////////////////////////////////////////////////////
21 module control_FSM(
22     input CLK,
23     input RESET,
24     input STEP_EN,
25     input busy,
26     input [5:0] opcode,
27     input [5:0] R_type_func, // Instruction's 6 LSB
28     input AEQZ,
29     output IN_INIT,
30     output [4:0] STATE_o,
31     output [2:0] ALUF,
32     output IRce,
33     output PCce,
34     output Ace,
35     output Bce,
36     output Cce,
37     output MARce,
38     output MDRce,
39     output [1:0] S1sel,
40     output [1:0] S2sel,
41     output DINTsel,
42     output MDRsel,
43     output Asel,
44     output shift_o,
45     output right,
46     output add_o,
47     output test,
48     output MR,
49     output MW,
50     output GPR_WE,
51     output Itype,
52     output Jlink
53 );
54
55 // Supporting registers
56 reg [4:0] next_state;
57 reg [4:0] STATE;
```

```
58     reg hold; // Small FSM latch indicator. Equal to one only if
59                     // it's in MEM transaction state for one cycle or more
60     wire bt; // 'Branch taken', determines either will branch or not
61
62 // states parameters definitions:
63 parameter INIT = 5'h00;
64 parameter fetch = 5'h01;
65 parameter decode = 5'h02;
66 parameter halt = 5'h1f;
67 parameter ALU = 5'h03;
68 parameter shift_state = 5'h04;
69 parameter WBR = 5'h05;
70 parameter ALUI= 5'h06;
71 parameter testI = 5'h07;
72 parameter WBI = 5'h08;
73 parameter addressCMP = 5'h09;
74 parameter load = 5'h0a;
75 parameter copyMDR2C = 5'h0b;
76 parameter copyGPR2MDR = 5'h0c;
77 parameter store = 5'h0d;
78 parameter JR = 5'h0e;
79 parameter savePC = 5'h0f;
80 parameter JALR = 5'h10;
81 parameter branch = 5'h11;
82 parameter Btaken = 5'h12;
83
84 // R type instructions' Function code
85 parameter R_inst_opcode = 6'b000000;
86 parameter slli = 3'b000;
87 parameter srli = 3'b010;
88 parameter add_func= 3'b011;
89 parameter sub = 3'b010;
90 parameter and_logic = 3'b110;
91 parameter or_logic = 3'b101;
92 parameter xor_logic = 3'b100;
93
94 // I type instructions' Opcode
95 parameter D1 = 3'b110; // special nop - go to INIT/fetch
96 parameter D5 = 3'b001;
97 parameter D6 = 3'b011;
98 parameter D7 = 2'b10; // lw and sw instruction flow
99 parameter D8 = 6'b010110;
100 parameter D9 = 6'b010111;
101 parameter D12 = 5'b00010;
102 parameter lw = 4'b0011;
103 parameter sw = 4'b1011;
104 parameter addi = 6'b001011;
105
106 always @(STATE or busy or STEP_EN) begin // State Machine flow
107     if(~STEP_EN) next_state = INIT;
108     case(STATE)
109         INIT:
110             if(STEP_EN) next_state = fetch;
111             else next_state = INIT;
112         fetch:
113             if(~busy & hold) next_state = decode;
114             else next_state = fetch;
```

```

115     decode:
116         if (opcode == R_inst_opcode) // R type instruction was received
117             if(R_type_func[5]==1) next_state = ALU;
118             else if(R_type_func[5] == 0) next_state = shift_state;
119             else next_state = halt; // Undefined R type instruction was detected
120     else // I type instruction
121         if(opcode[5:3] == D1) // special nop
122             next_state = (STEP_EN) ? fetch : INIT;
123             else if(opcode == addi) next_state = ALUI;
124             else if(opcode[5:3] == D6) next_state = testI;
125             else if(opcode[5:4]== D7) next_state = addressCMP;
126             else if(opcode == D8) next_state = JR;
127             else if(opcode == D9) next_state = savePC;
128             else if(opcode[5:1] == D12) next_state = branch;
129             else next_state = halt;
130     ALU, shift_state:
131         next_state = WBR;
132     ALUI, testI, copyMDR2C:
133         next_state = WBI;
134     WBI, WBR, JR, JALR, Btaken:
135         next_state = (STEP_EN) ? fetch : INIT;
136     addressCMP:
137         next_state = (opcode[3]) ? copyGPR2MDR : load;
138     copyGPR2MDR:
139         next_state = store;
140     store:
141         if(~busy & hold) next_state = INIT;
142         else next_state = store;
143     load:
144         if(~busy & hold) next_state = copyMDR2C;
145         else next_state = load;
146     savePC:
147         next_state = JALR;
148     branch:
149         if(bt) next_state = Btaken; // Branch at the next step
150         else next_state = (STEP_EN) ? fetch : INIT; // Don't branch and read the
next instruction
151     halt:
152         next_state = halt;
153
154     default:
155         next_state = INIT;
156
157     endcase
158 end
159
160 always @(posedge CLK) begin // Sequential part: checks reset and pass STATE forward
161     if (RESET)
162         STATE <= INIT;
163     else begin
164         STATE <= #1 next_state;
165         hold <= ((STATE == fetch)|(STATE == store)|(STATE == load)) ? 1 : 0;
166     end
167 end
168
169 // Combinational part - compute output signals and conditions for the next state
170 assign IN_INIT = ((STATE == INIT) || (STATE == halt)) ? 1 : 0;
171 assign STATE_o = STATE;
172 assign ALUF = ((opcode[5:3] == 3'b001)|| (opcode[5:3] == 3'b011)) ? opcode[2:0] : (
opcode == R_inst_opcode)&(R_type_func[5:3] == 3'b100) ? R_type_func[2:0] : 3'b000;
173 assign IRce = (STATE == fetch) ? 1:0;
174 assign PCce = (STATE == decode)|(STATE == Btaken)|(STATE == JR)|(STATE == JALR) ? 1:0;
175 assign Ace = (STATE == decode) ? 1:0;
176 assign Bce = Ace;
177 assign Cce = (STATE == ALU)|(STATE == testI)|(STATE == ALUI)|(STATE == shift_state)|(STATE == copyMDR2C)|(STATE == savePC) ? 1:0;
178 assign MARce = (STATE == addressCMP) ? 1:0;
179 assign MDRce = (STATE == load)|(STATE == copyGPR2MDR) ? 1:0;
180 assign S1sel[1] = (STATE == copyGPR2MDR)|(STATE == copyMDR2C) ? 1:0;
181 assign S1sel[0] = ((Cce)&(STATE != savePC))|(STATE == addressCMP)|(STATE == JR)|(STATE == JALR) ? 1:0;
182 assign S2sel = (STATE == decode) ? 2'b11 : ((opcode[5:3] == 3'b010)|(STATE == copyGPR2MDR)|(STATE==copyMDR2C)) ? 2'b10 : ((STATE == addressCMP)||((~GPR_WE)&Itype))||(STATE == Btaken)) ? 2'b01 : 2'b0;
183 assign DINTsel = (shift_o)|(STATE == copyGPR2MDR)|(STATE == copyMDR2C) ? 1:0;
184 assign MDRsel = (STATE == load) ? 1 : 0;
185 assign Asel = (STATE == load)|(STATE == store) ? 1 : 0;
186 assign shift_o = (STATE == shift_state) ? 1 : 0;
187 assign right = (shift_o & R_type_func[1]);
188 assign add_o = (STATE ==decode)|(STATE == ALUI)|(STATE == addressCMP)|(STATE == Btaken )| (STATE ==JR)|(STATE == savePC)|(STATE == JALR) ? 1 : 0;
189 assign test = (STATE == testI) ? 1 : 0;
190 assign MR = (STATE == fetch)|(STATE == load) ? 1 : 0;
191 assign MW = (STATE == store) ? 1 : 0;
192 assign GPR_WE = (STATE ==WBI)|(STATE == WBR)|(STATE == JALR) ? 1 : 0;
193 assign Itype = (STATE == ALUI)|(STATE == testI)|(STATE == WBI) ? 1 : 0;
194 assign Jlink = (STATE == JALR) ? 1 : 0;
195 assign bt = (AEQZ ^ opcode[0]);
196
197 endmodule

```

Test Vectors and Control Validation

Test vectors

To improve the readability of the test vectors, we marked the different groups of signals in separated colors (light blue for inputs, orange for MAC related signals, purple for the select signals of the muxes and so on). In addition, the state of the control is written in hexadecimal for similar reasons. Table of states' numbering is attached below:

State	Number(Hex)	State	Number(Hex)
INIT	0	load	a
Fetch	1	copyMDR2C	b
Decode	2	copyGPR2MDR	c
ALU	3	store	d
Shift	4	JR	e
WBR	5	savePC	f
ALUI	6	JALR	10
testI	7	branch	11
WBI	8	Btaken	12
addressCAMP	9	halt	1f

ALU:

	Clock	0	1	2	3	4	5	6	7	8	9
Input	STEP_EN	0	0	1	0	0	0	0	0	0	0
Input	Reset	1	0	0	0	0	0	0	0	0	0
Input	Busy	0	0	0	1	1	0	0	0	1	0
Output	STATE Ctrl	0	0	1	1	1	1	2	3	5	0
Output	IN_INIT	1	1	0	0	0	0	0	0	0	1
Output	S1sel	00	00	00	00	00	00	00	01	00	00
Output	S2sel	00	00	00	00	00	00	11	00	00	00
Output	Asel	0	0	0	0	0	0	0	0	0	0
Output	DINTsel	0	0	0	0	0	0	0	0	0	0
Output	MDRsel	0	0	0	0	0	0	0	0	0	0
Output	IRce	0	0	1	1	1	1	0	0	0	0
Output	PCce	0	0	0	0	0	0	1	0	0	0
Output	Ace	0	0	0	0	0	0	1	0	0	0
Output	Bce	0	0	0	0	0	0	0	1	0	0
Output	Cce	0	0	0	0	0	0	0	1	0	0
Output	MARce	0	0	0	0	0	0	0	0	0	0
Output	MDRce	0	0	0	0	0	0	0	0	0	0
Output	GPR_WE	0	0	0	1	0	0	0	0	1	0
Output	Itype	0	0	0	0	0	0	0	0	0	0
Output	Jlink	0	0	0	0	0	0	0	0	0	0
Output	test	0	0	0	0	0	0	0	0	0	0
Output	add	0	0	0	0	0	0	1	0	0	0
Output	shift	0	0	0	0	0	0	0	0	0	0
Output	right	0	0	0	0	0	0	0	0	0	0
Output	MR	0	0	1	1	1	1	0	0	0	0
Output	MW	0	0	0	0	0	0	0	0	0	0
Output	STATE MAC	00	00	00	01	01	10	00	00	00	00
Input(MAC)	ACK_N	1	1	1	1	1	0	1	1	1	1

TESTI:

Clock		0	1	2	3	4	5	6	7	8	9
Input	STEP_EN	0	0	1	0	0	0	0	0	0	0
Input	Reset	1	0	0	0	0	0	0	0	0	0
Input	Busy	0	0	0	1	1	0	0	0	0	0
Output	STATE Ctrl	0	0	1	1	1	1	2	7	8	0
Output	IN_INIT	1	1	0	0	0	0	0	0	0	1
Output	S1sel	00	00	00	00	00	00	00	01	00	0
Output	S2sel	00	00	00	00	00	00	11	01	00	0
Output	Asel	0	0	0	0	0	0	0	0	0	0
Output	DINTsel	0	0	0	0	0	0	0	0	0	0
Output	MDRsel	0	0	0	0	0	0	0	0	0	0
Output	IRce	0	0	1	1	1	1	0	0	0	0
Output	PCce	0	0	0	0	0	0	1	0	0	0
Output	Ace	0	0	0	0	0	0	1	0	0	0
Output	Bce	0	0	0	0	0	0	1	0	0	0
Output	Cce	0	0	0	0	0	0	0	1	0	0
Output	MARce	0	0	0	0	0	0	0	0	0	0
Output	MDRce	0	0	0	0	0	0	0	0	0	0
Output	GPR_WE	0	0	0	1	0	0	0	0	1	0
Output	Itype	0	0	0	0	0	0	0	1	1	0
Output	Jlink	0	0	0	0	0	0	0	0	0	0
Output	test	0	0	0	0	0	0	0	1	0	0
Output	add	0	0	0	0	0	0	1	0	0	0
Output	shift	0	0	0	0	0	0	0	0	0	0
Output	right	0	0	0	0	0	0	0	0	0	0
Output	MR	0	0	1	1	1	1	0	0	0	0
Output	MW	0	0	0	0	0	0	0	0	0	0
Output	STATE MAC	00	00	00	01	01	10	00	00	00	0
Input(MAC)	ACK_N	1	1	1	1	1	0	1	1	1	1

LOAD:

STORE:

JALR:

	Clock	0	1	2	3	4	5	6	7	8	9
Input	STEP_EN	0	0	1	0	0	0	0	0	0	0
Input	Reset	1	0	0	0	0	0	0	0	0	0
Input	Busy	0	0	0	1	1	0	0	0	0	0
Output	STATE Ctrl	0	0	1	1	1	1	2	f	10	0
Output	IN_INIT	1	1	0	0	0	0	0	0	0	1
Output	S1sel	00	00	00	00	00	00	00	00	01	0
Output	S2sel	00	00	00	00	00	00	11	10	10	0
Output	Asel	0	0	0	0	0	0	0	0	0	0
Output	DINTsel	0	0	0	0	0	0	0	0	0	0
Output	MDRsel	0	0	0	0	0	0	0	0	0	0
Output	IRce	0	0	1	1	1	1	0	0	0	0
Output	PCce	0	0	0	0	0	0	1	0	1	0
Output	Ace	0	0	0	0	0	0	1	0	0	0
Output	Bce	0	0	0	0	0	0	1	0	0	0
Output	Cce	0	0	0	0	0	0	0	1	0	0
Output	MARce	0	0	0	0	0	0	0	0	0	0
Output	MDRce	0	0	0	0	0	0	0	0	0	0
Output	GPR_WE	0	0	0	1	0	0	0	0	1	0
Output	Itype	0	0	0	0	0	0	0	0	0	0
Output	Jlink	0	0	0	0	0	0	0	0	1	0
Output	test	0	0	0	0	0	0	0	0	0	0
Output	add	0	0	0	0	0	0	1	1	1	0
Output	shift	0	0	0	0	0	0	0	0	0	0
Output	right	0	0	0	0	0	0	0	0	0	0
Output	MR	0	0	1	1	1	1	0	0	0	0
Output	MW	0	0	0	0	0	0	0	0	0	0
Output	STATE MAC	00	00	00	01	01	10	00	00	00	0
Input(MAC)	ACK_N	1	1	1	1	1	0	1	1	1	1

BTAKEN(both for taking the branch and skipping it):

Output	Jlink	0	0	0	0	0	0	0	0	0	0
Output	test	0	0	0	0	0	0	0	0	0	0
Output	add	0	0	0	0	0	1	0	1	0	0
Output	shift	0	0	0	0	0	0	0	0	0	0
Output	right	0	0	0	0	0	0	0	0	0	0
Output	MR	0	0	1	1	1	0	0	0	0	0
Output	MW	0	0	0	0	0	0	0	0	0	0
Output	STATE MAC	0	0	0	1	1	2	0	0	0	0
Input(MAC)	ACK_N	1	1	1	1	1	0	1	1	1	1

Control test-bench

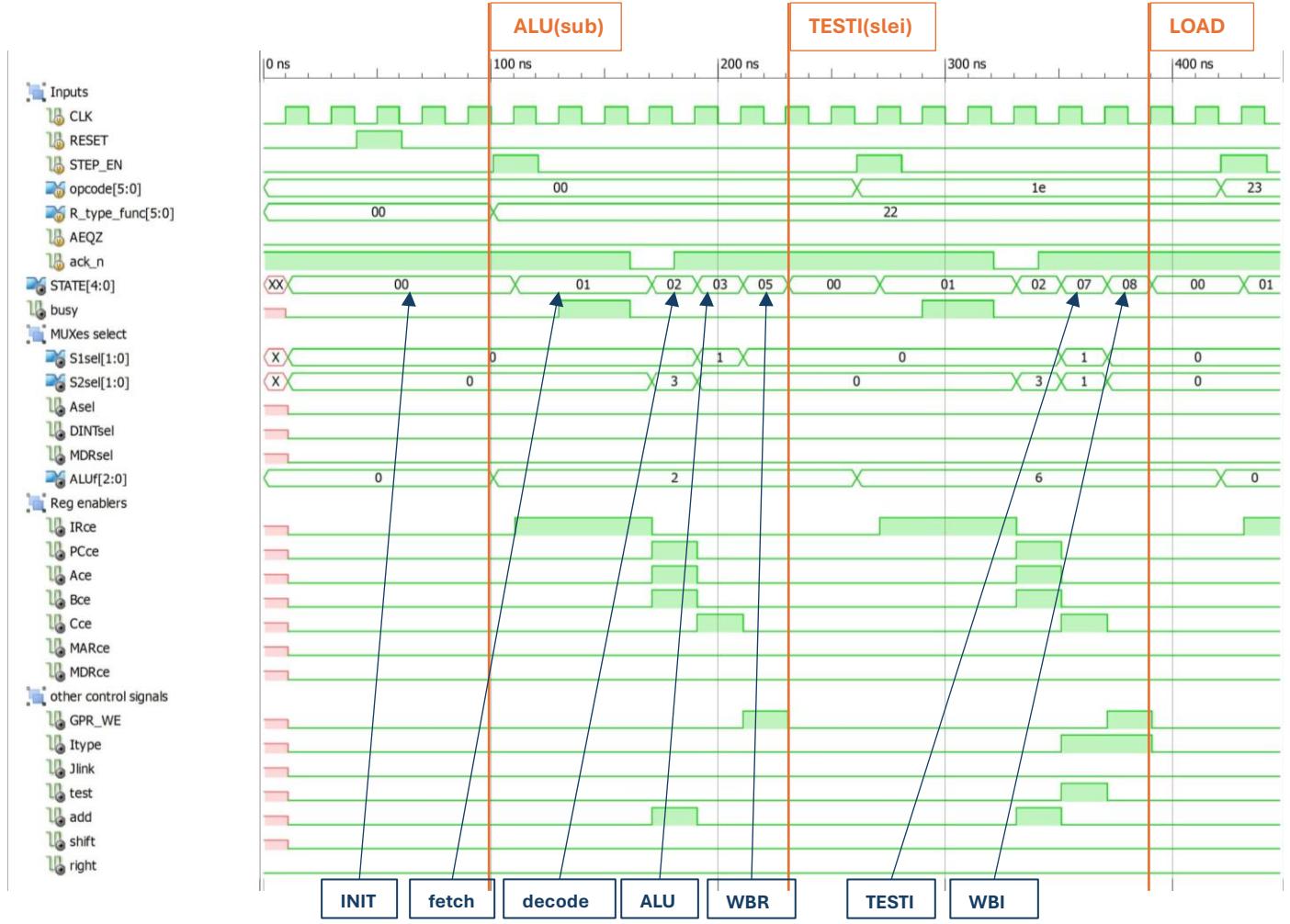
After designing the control unit, we used the next test-bench code to validate its functionality :

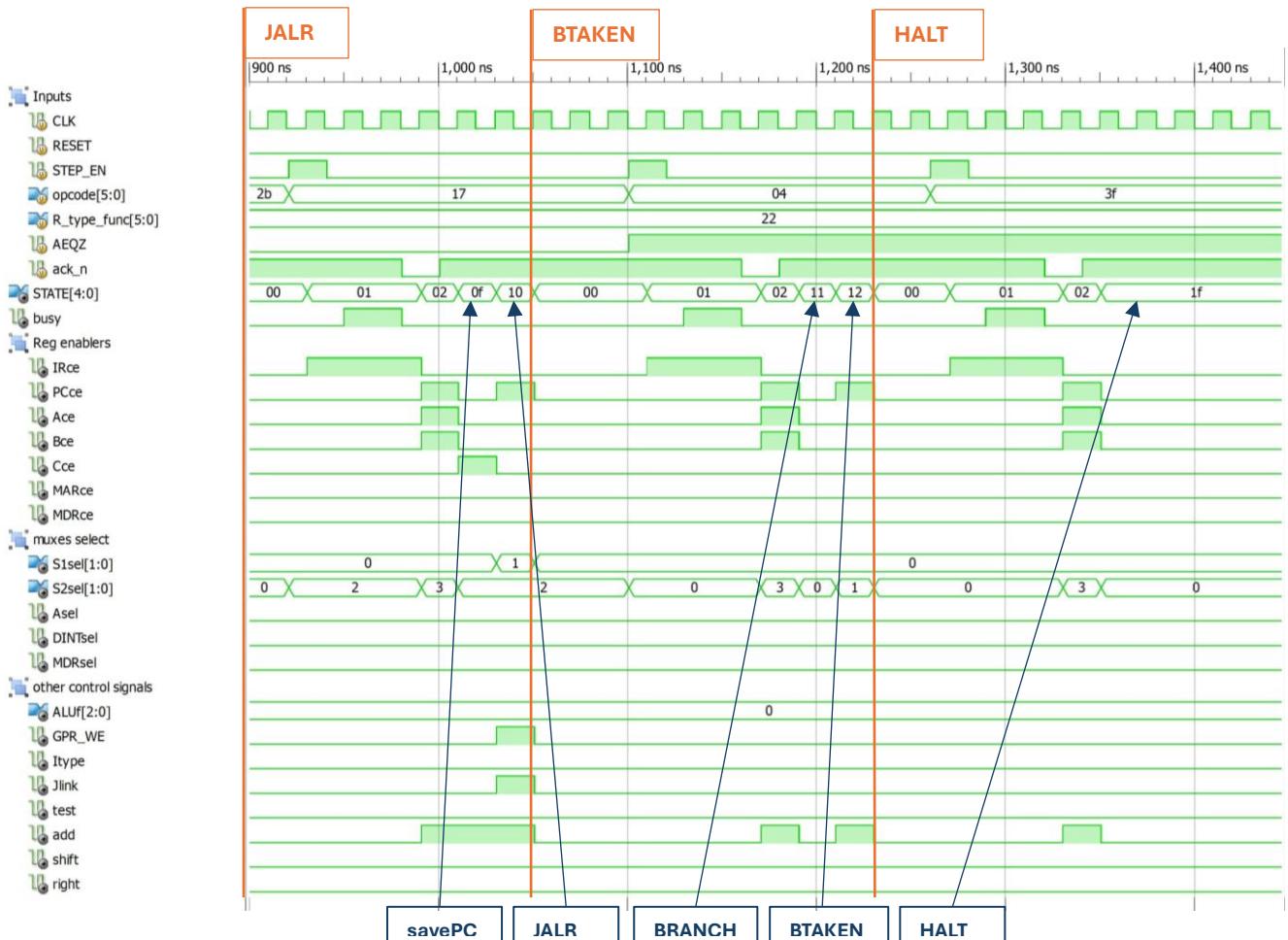
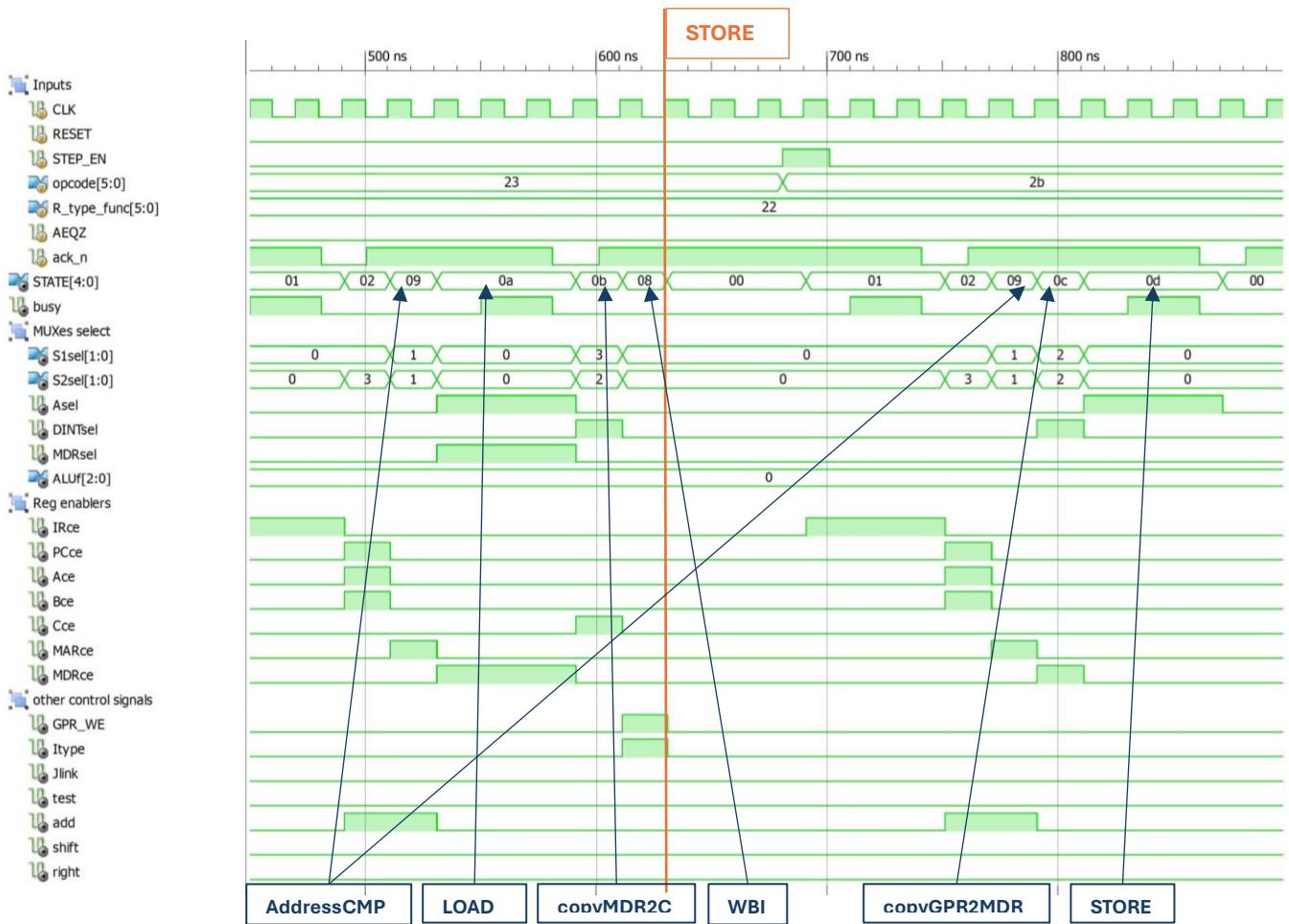
```

3 `timescale 1ns / 1ps
4
5 module control_control_sch_tb();
6
7 // Inputs
8 reg CLK;
9 reg RESET;
10 reg STEP_EN;
11 reg [5:0] opcode;
12 reg [5:0] R_type_func;
13 reg ack_n;
14 reg AEQZ;
15
16 // Output
17 wire IRce;
18 wire PCce;
19 wire Ace;
20 wire Bce;
21 wire Cce;
22 wire MARce;
23 wire MDR;
24 wire DINTsel;
25 wire MDRsel;
26 wire Asel;
27 wire shift;
28 wire right;
29 wire add;
30 wire test;
31 wire MR;
32 wire busy;
33 wire GPR_WE;
34 wire [2:0] ALUF;
35 wire Jlink;
36 wire Itype;
37 wire [1:0] S1sel;
38 wire [1:0] S2sel;
39 wire [1:0] MAC_state;
40 wire wr_N;
41 wire MW;
42 wire as_N;
43 wire STOP_N;
44 wire [4:0] STATE;
45
46 // Bidirs
47
48 // Instantiate the UUT
49 control UUT (
50     .CLK(CLK),
51     .IRce(IRce),
52     .PCce(PCce),
53     .Ace(Ace),
54     .Bce(Bce),
55     .Cce(Cce),
56     .RESET(RESET),
57     .MARce(MARce),
58     .MDR(MDR),
59     .DINTsel(DINTsel),
60     .MDRsel(MDRsel),
61     .Asel(Asel),
62     .shift(shift),
63     .right(right),
64     .add(add),
65     .test(test),
66     .MR(MR),
67     .busy(busy),
68     .STEP_EN(STEP_EN),
69     .GPR_WE(GPR_WE),
70     .opcode(opcode),
71     .R_type_func(R_type_func),
72     .ALUF(ALUF),
73     .Jlink(Jlink),
74     .Itype(Itype),
75     .S1sel(S1sel),
76     .S2sel(S2sel),
77     .MAC_state(MAC_state),
78     .wr_N(wr_N),
79     .ack_n(ack_n),
80     .MW(MW),
81     .as_N(as_N),
82     .STOP_N(STOP_N),
83     .STATE(STATE),
84     .AEQZ(AEQZ)
85 );
86
87 initial
88     CLK = 0;
89     always #10 CLK = ~CLK;
90
91 // Initialize Inputs
92 initial begin
93     STEP_EN = 0;
94     CLK = 0;
95     RESET = 0;
96     opcode = 6'b000000;
97     ack_n = 1;
98     R_type_func = 6'b000000;
99     AEQZ = 0;
100
101 //wating 2 clock cycles for reset
102 #41;
103 RESET = 1;
104 #20;
105 RESET = 0;
106 #40;
107
108 // start ALU(sub) operation:
109 initial
110     opcode = 6'b000000;
111     R_type_func = 6'b100010;
112     STEP_EN = 1;
113     #20;
114     STEP_EN = 0;
115     #40;
116     ack_n = 0;
117     #20;
118     ack_n = 1;
119     #80;
120
121 // start TESTI(slei) operation:
122 initial
123     opcode = 6'b011110;
124     STEP_EN = 1;
125     #20;
126     STEP_EN = 0;
127     #40;
128     ack_n = 0;
129     #20;
130     ack_n = 1;
131     #80;
132
133 // start load operation:
134 initial
135     opcode = 6'b100011;
136     STEP_EN = 1;
137     #20;
138     STEP_EN = 0;
139     #40;
140     ack_n = 0;
141     #20;
142     ack_n = 1;
143     #80;
144     ack_n = 0;
145     #20;
146     ack_n = 1;
147     #80;
148 // start store operation:
149     opcode = 6'b101011;
150     STEP_EN = 1;
151     #20;
152     STEP_EN = 0;
153     #40;
154     ack_n = 0;
155     #20;
156     ack_n = 1;
157     #100;
158     ack_n = 0;
159     #20;
160     ack_n = 1;
161     #40;
162
163 // start JALR operation:
164     opcode = 6'b010111;
165     STEP_EN = 1;
166     #20;
167     STEP_EN = 0;
168     #40;
169     ack_n = 0;
170     #20;
171     ack_n = 1;
172     #100;
173
174 // start BTAKEN(beqz) operation:
175     opcode = 6'b000100;
176     AEQZ = 1'b1;
177     STEP_EN = 1;
178     #20;
179     STEP_EN = 0;
180     #40;
181     ack_n = 0;
182     #20;
183     ack_n = 1;
184     #80;
185
186 // start halt operatoin
187     opcode = 6'b111111;
188     STEP_EN = 1;
189     #20;
190     STEP_EN = 0;
191     #40;
192     ack_n = 0;
193     #20;
194     ack_n = 1;
195     #80;
196
197 end
198
199 endmodule

```

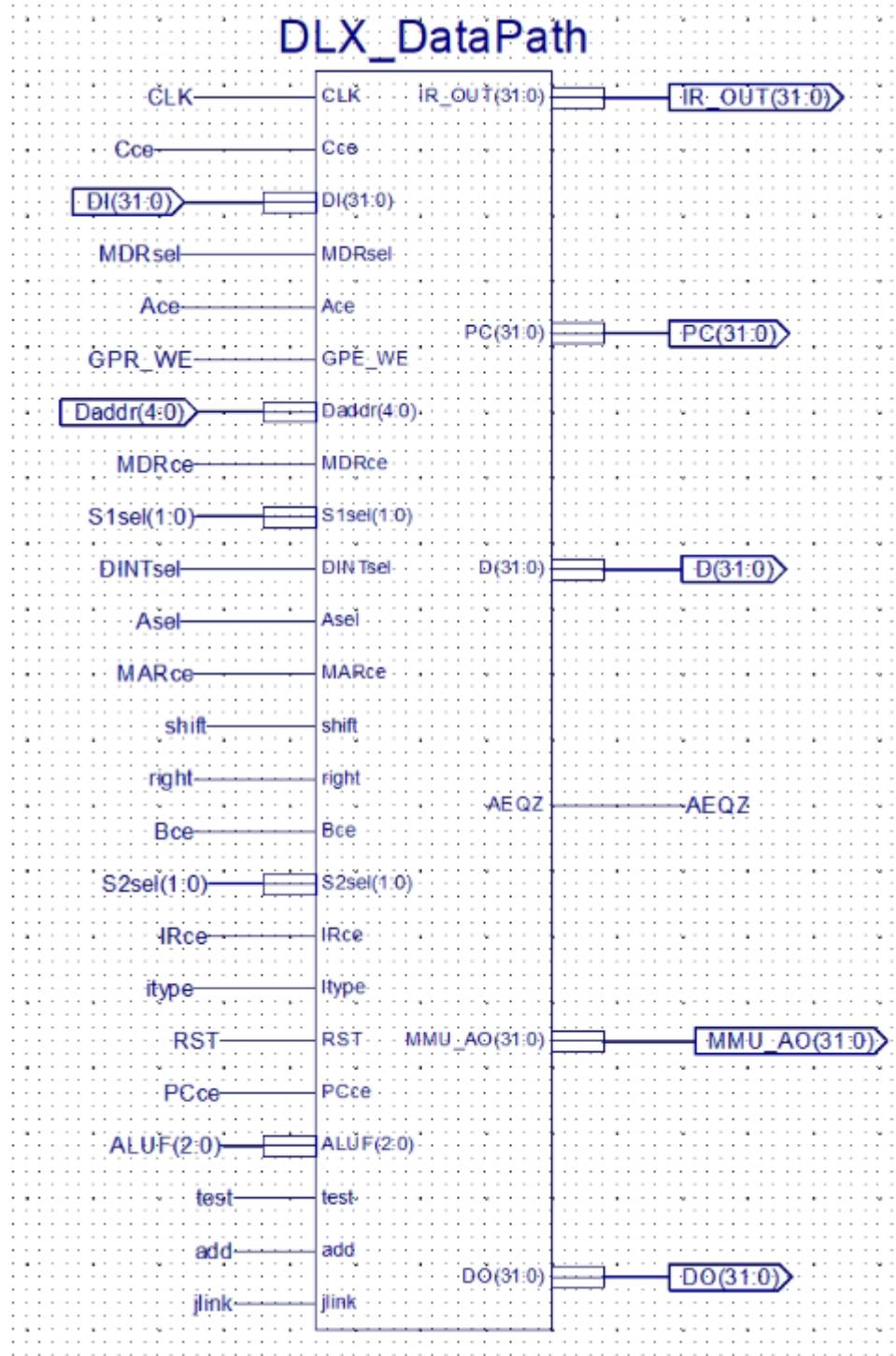
After running the code and simulating the control, we received the following waveforms, which matching the test vectors:





The Datapath

Datapath schematics:

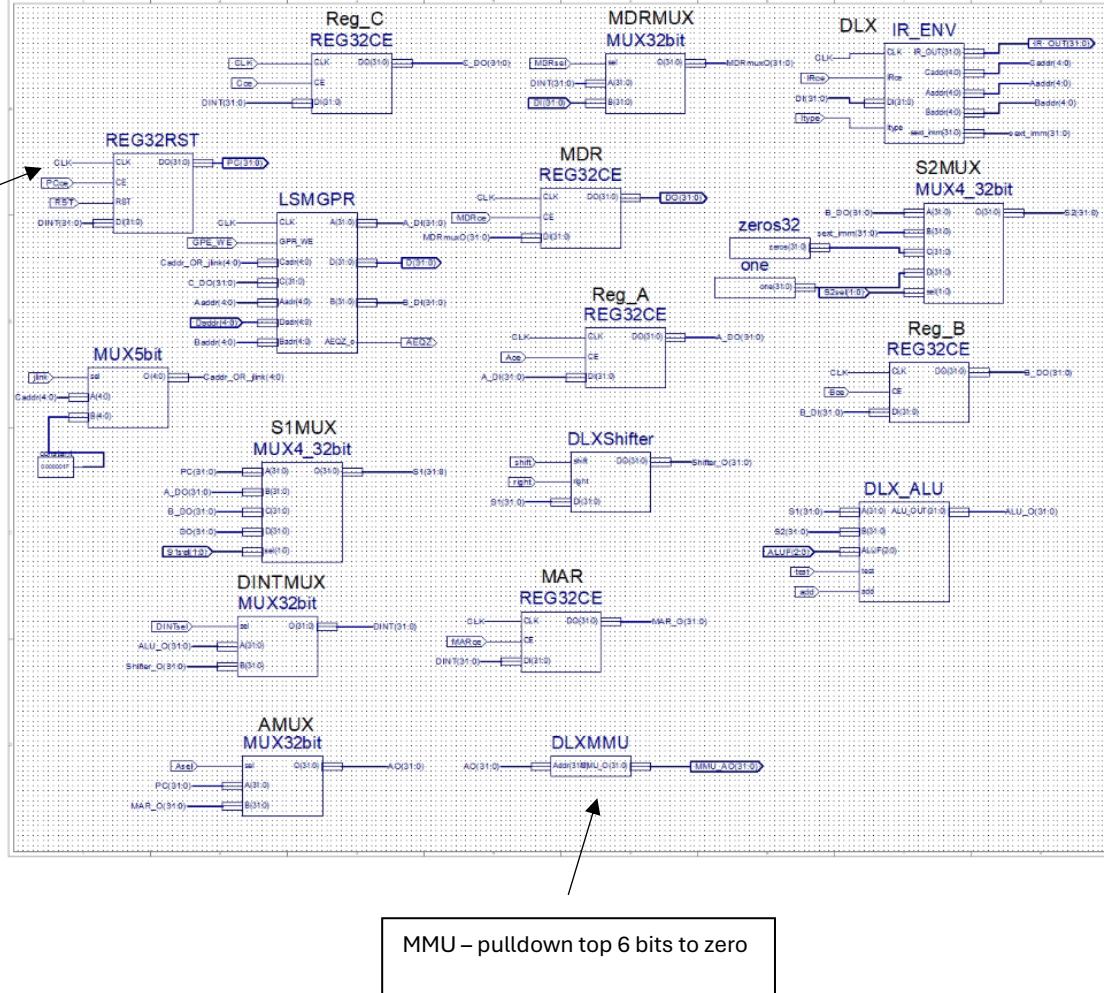


The datapath contains the following units:

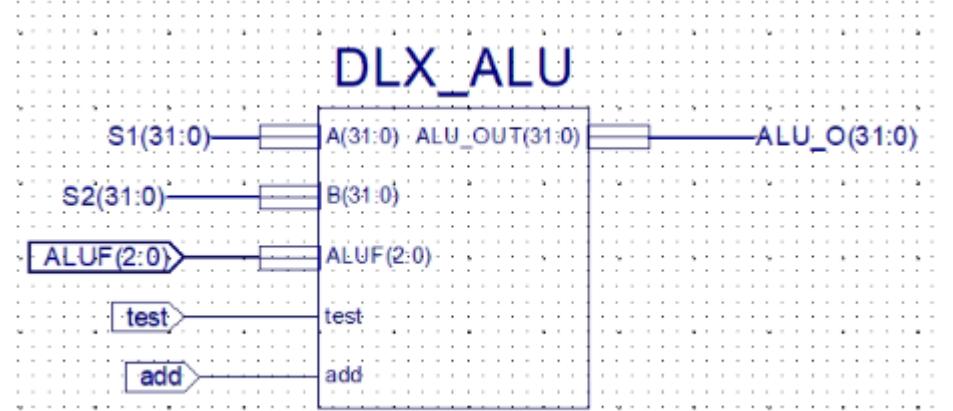
Registers: A, B, C, PC (register with reset), MDR, MAR

Muxes: MDR, DINT, S1, S2, A, jlink mux (chooses between Caddr to R31 in case of jlink).

Shifter, GPR (from lab 6), ALU, MMU, IR environment.

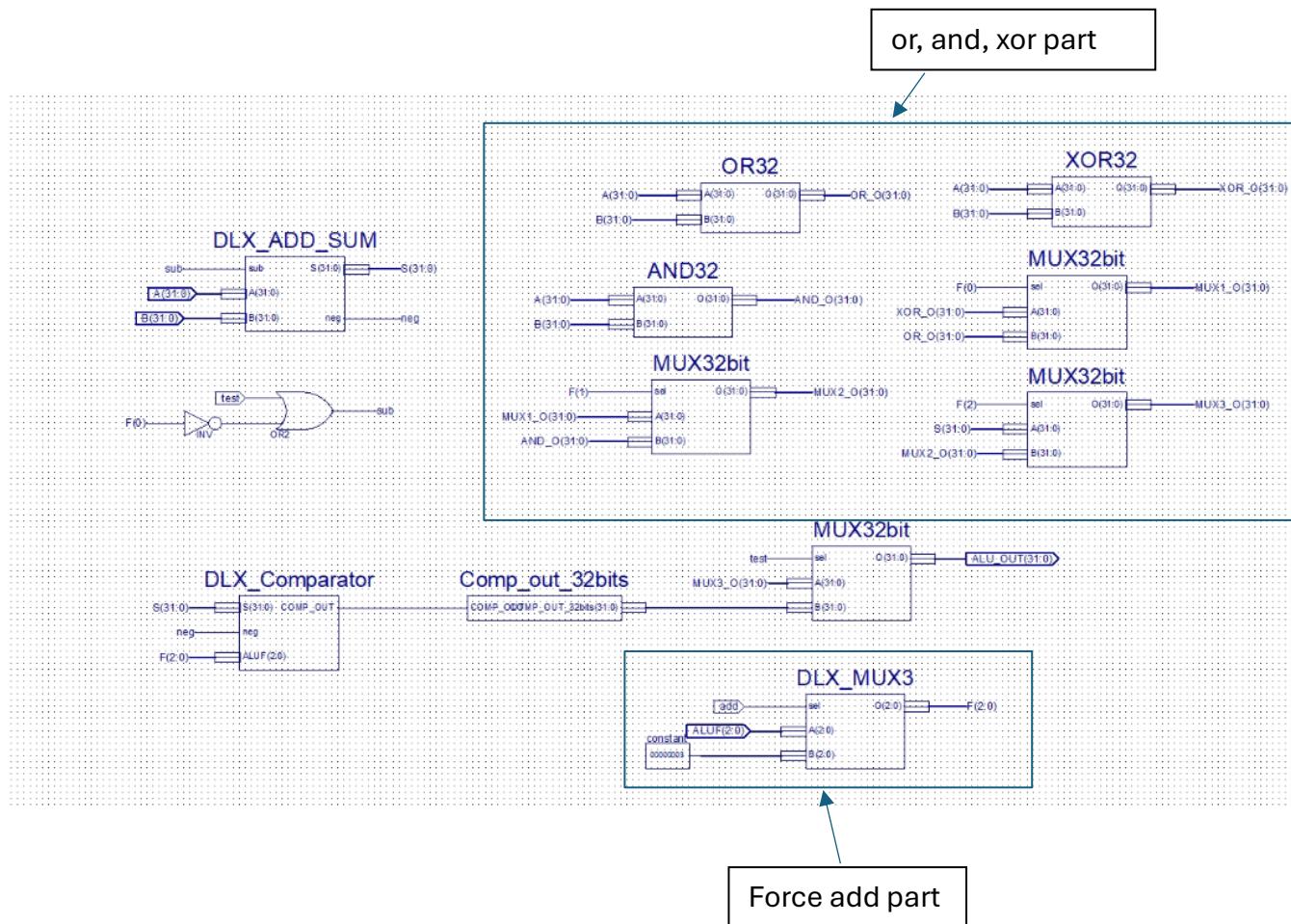


The ALU:



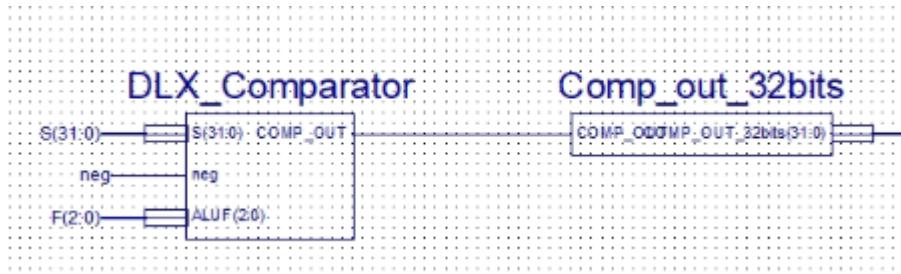
The ALU responsible of arithmetic and comparation operation

Inside the ALU:

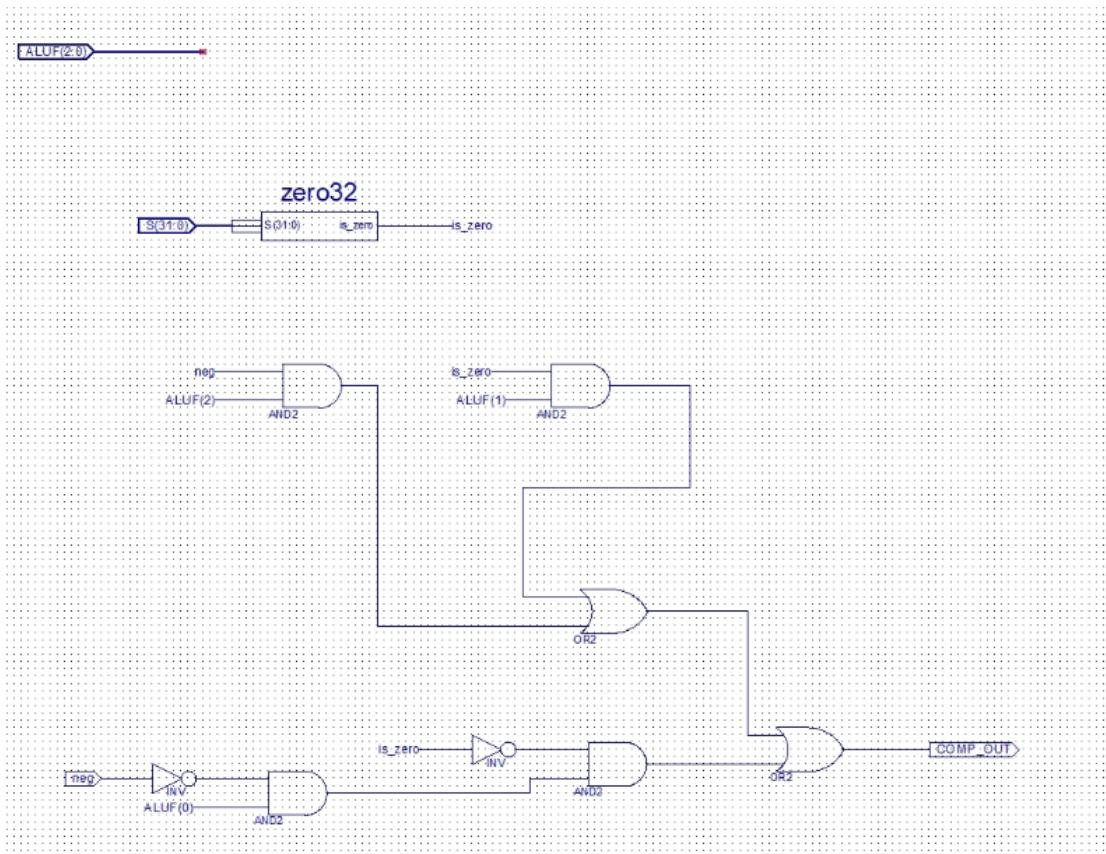


The comparator:

inside the ALU we have a comparator unit which responsible of the comparation operation

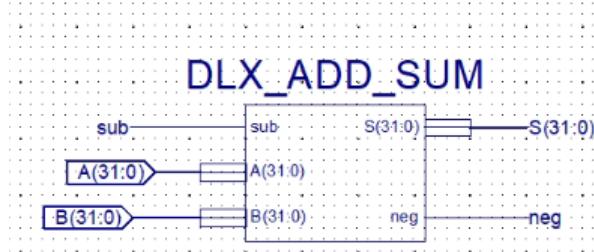


inside the comparator:

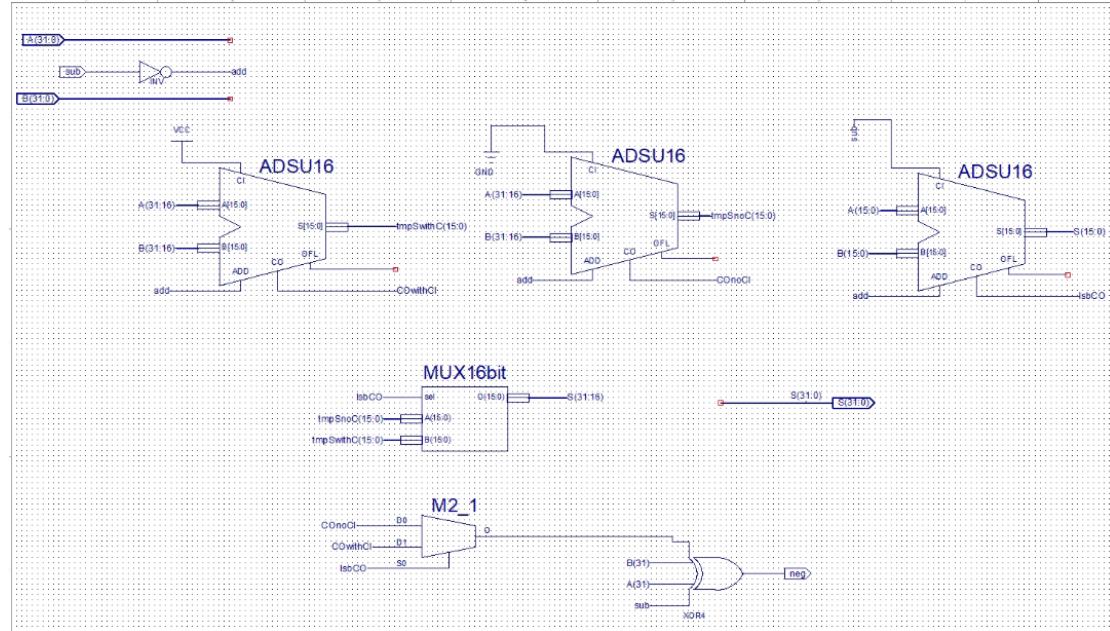


The ADD-SUB unit:

inside the ALU we have ADD-SUB unit which responsible for add and subtract operations

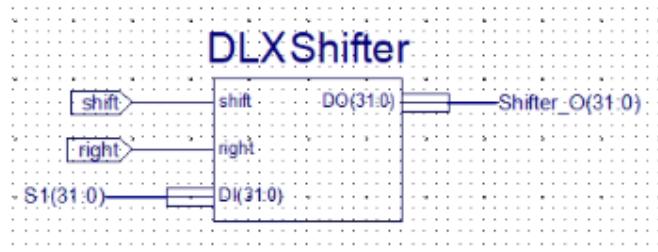


inside the ADD-SUB:



The shifter:

the shifter responsible of shifting operations



Inside the shifter:

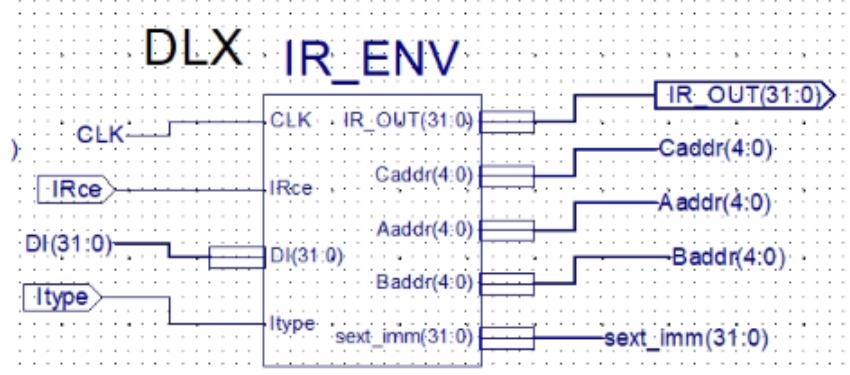
```

1  // 
2  module DLXShifter(
3      input [31:0] DI,
4      input shift,
5      input right,
6      output [31:0] DO
7  );
8
9
10 assign DO = (shift && right) ? (DI>>1) : (shift) ? (DI<<1) : DI;
11
12 endmodule
13

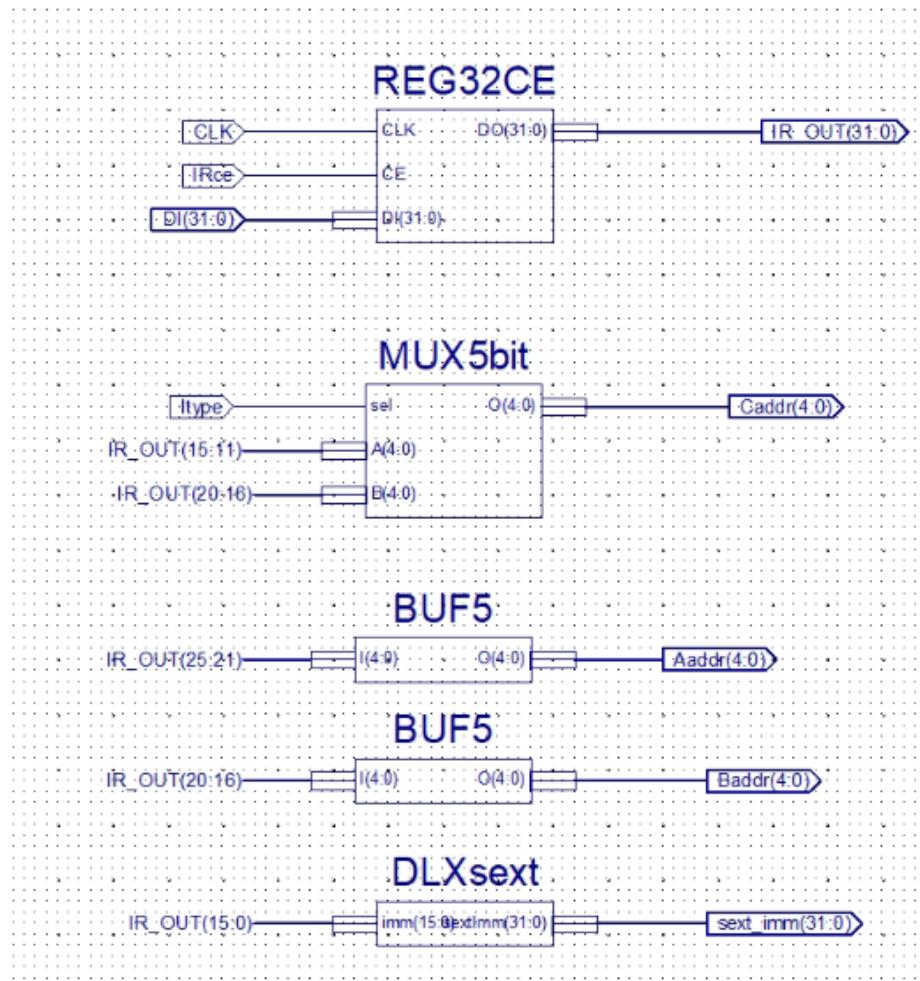
```

The IR environment:

The IR environment responsible for breaking the instruction into it's RD RS1 RS2 IMM parts depends on the type of it.



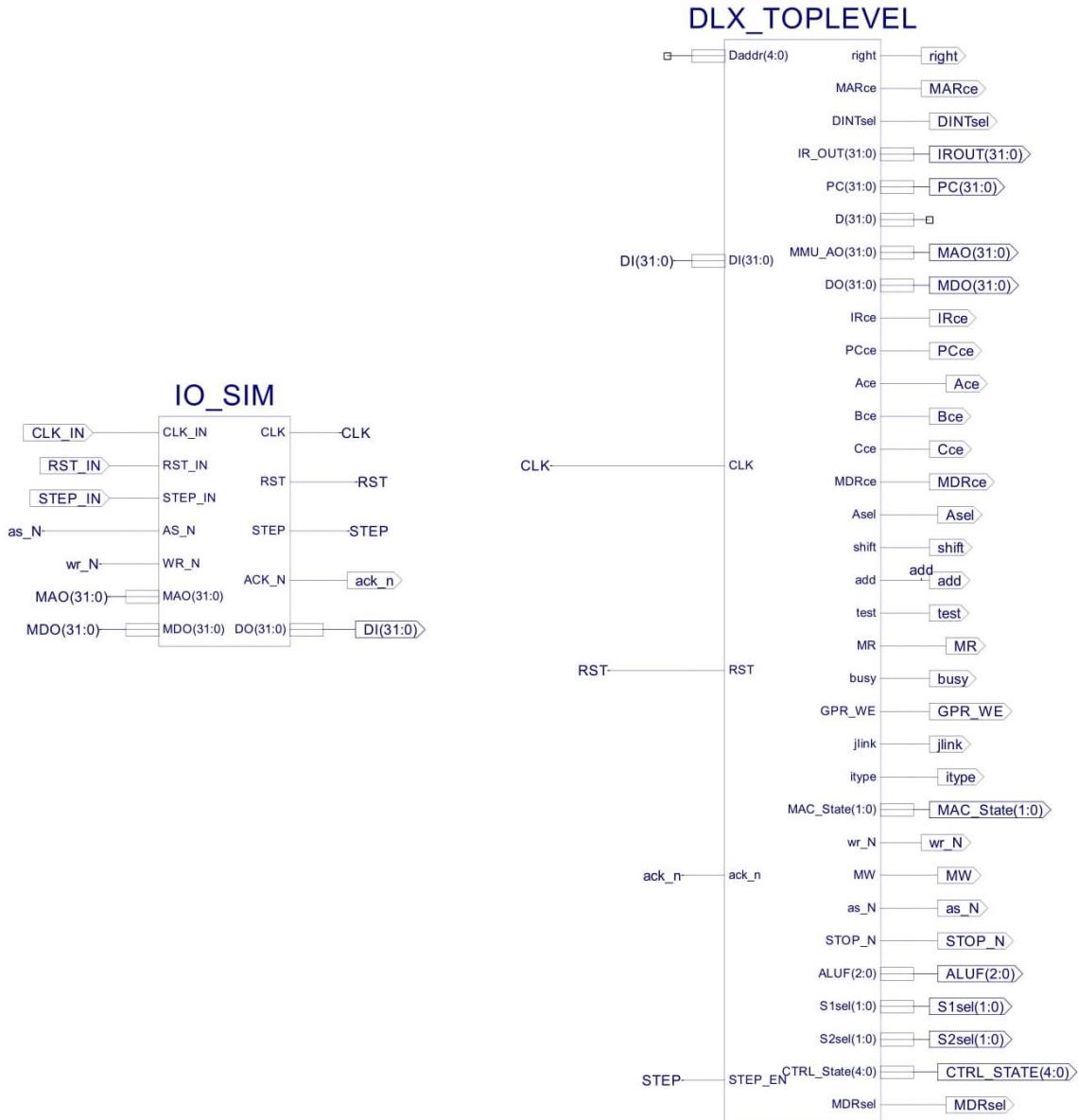
Inside the IR environment:



DLX Simulation and The IOsimul

Scheme of the connections to the DLX:

In order to simulate the entire DLX processor functionality and behavior, after finishing the design we connected the DLX to the IOsimul unit the way it shown in the scheme below.



After connecting the two systems, we built a test-bench and simulated the execution of DLX program and we received its waveforms, which all attached in the next pages.

DLX program:

The code that was loaded into the IOsimul, as it was given by the course stuff, can be seen below, with the traversed states for each instruction detailed right to it:

```
pc=0x0

lw R1 R0 data1      INIT → fetch → decode → AddressCMP → LOAD → copyMDR2C → WBI → INIT/fetch
sw R1 R0 adr2      INIT → fetch → decode → AddressCMP → copyGPR2MDR → STORE → INIT/fetch
addi R2 R1 0x1       INIT → fetch → decode → ALUI → WBI → INIT/fetch
add R3 R1 R2       INIT → fetch → decode → ALU → WBR → INIT/fetch
srlti R5 R2        INIT → fetch → decode → SHIFT → WBR → INIT/fetch
slli R4 R2         INIT → fetch → decode → SHIFT → WBR → INIT/fetch
slti R6 R2 3       INIT → fetch → decode → TESTI → WBI → INIT/fetch
beqz R6 jump1      INIT → fetch → decode → BRANCH (no branch) → INIT/fetch
sw R4 R1 adr1      INIT → fetch → decode → AddressCMP → copyGPR2MDR → STORE → INIT/fetch

jump1: lw R7 R1 adr1      INIT → fetch → decode → AddressCMP → LOAD → copyMDR2C → WBI → INIT/fetch
sgei R8 R4 0        INIT → fetch → decode → TESTI → WBI → INIT/fetch
bnez R8 jump2      INIT → fetch → decode → BRANCH → Btaken → INIT/fetch
or R4 R2 R4        INIT → fetch → decode → ALU → WBR → INIT/fetch
jump2: special-nop INIT → fetch → decode → INIT/fetch
addi R3 R0 jump3    INIT → fetch → decode → ALUI → WBI → INIT/fetch
jalr R3            INIT → fetch → decode → savePC → JALR → INIT/fetch
sub R3 R5 R1       INIT → fetch → decode → ALU → WBR → INIT/fetch
beqz R3 jump4      INIT → fetch → decode → BRANCH → Btaken → INIT/fetch
jump3: jr R31      INIT → fetch → decode → JR → INIT/fetch
jump4: seqi R3 R0 -1 INIT → fetch → decode → TESTI → WBI → INIT/fetch

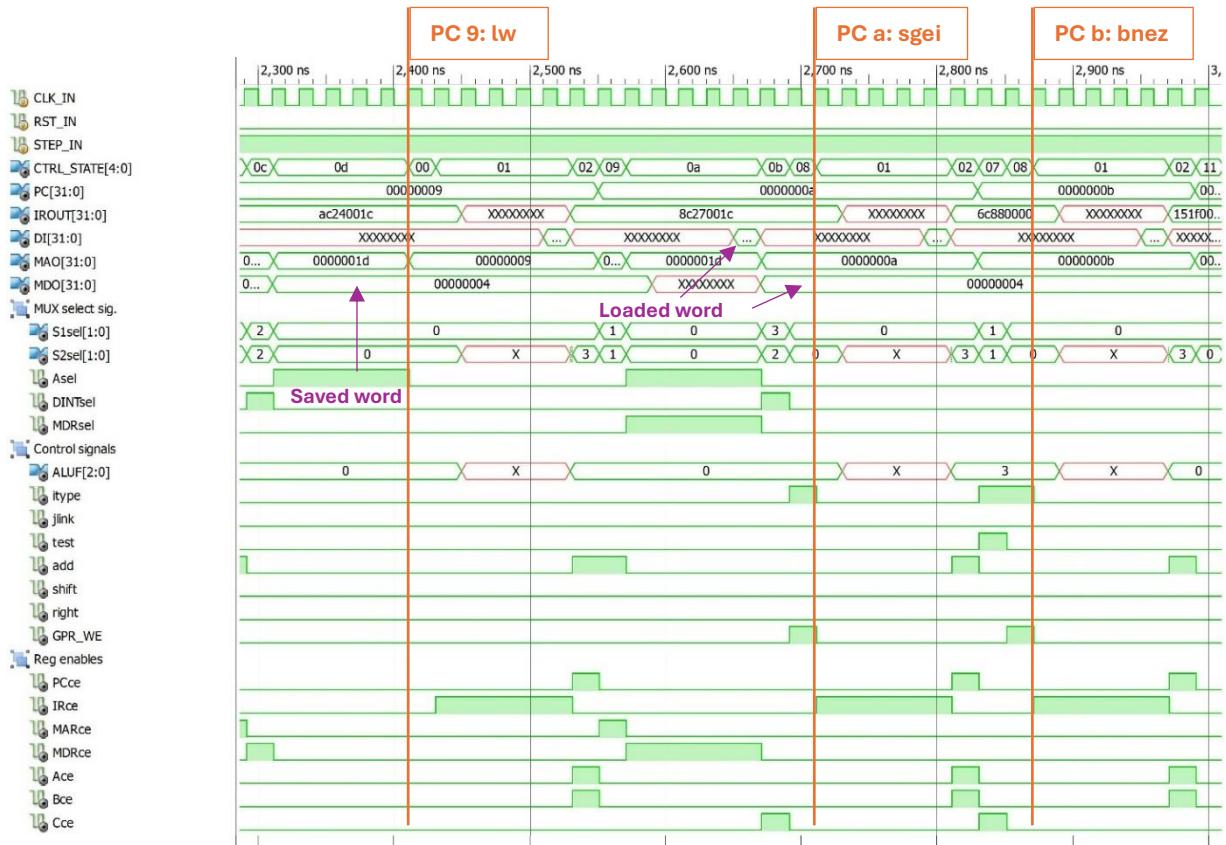
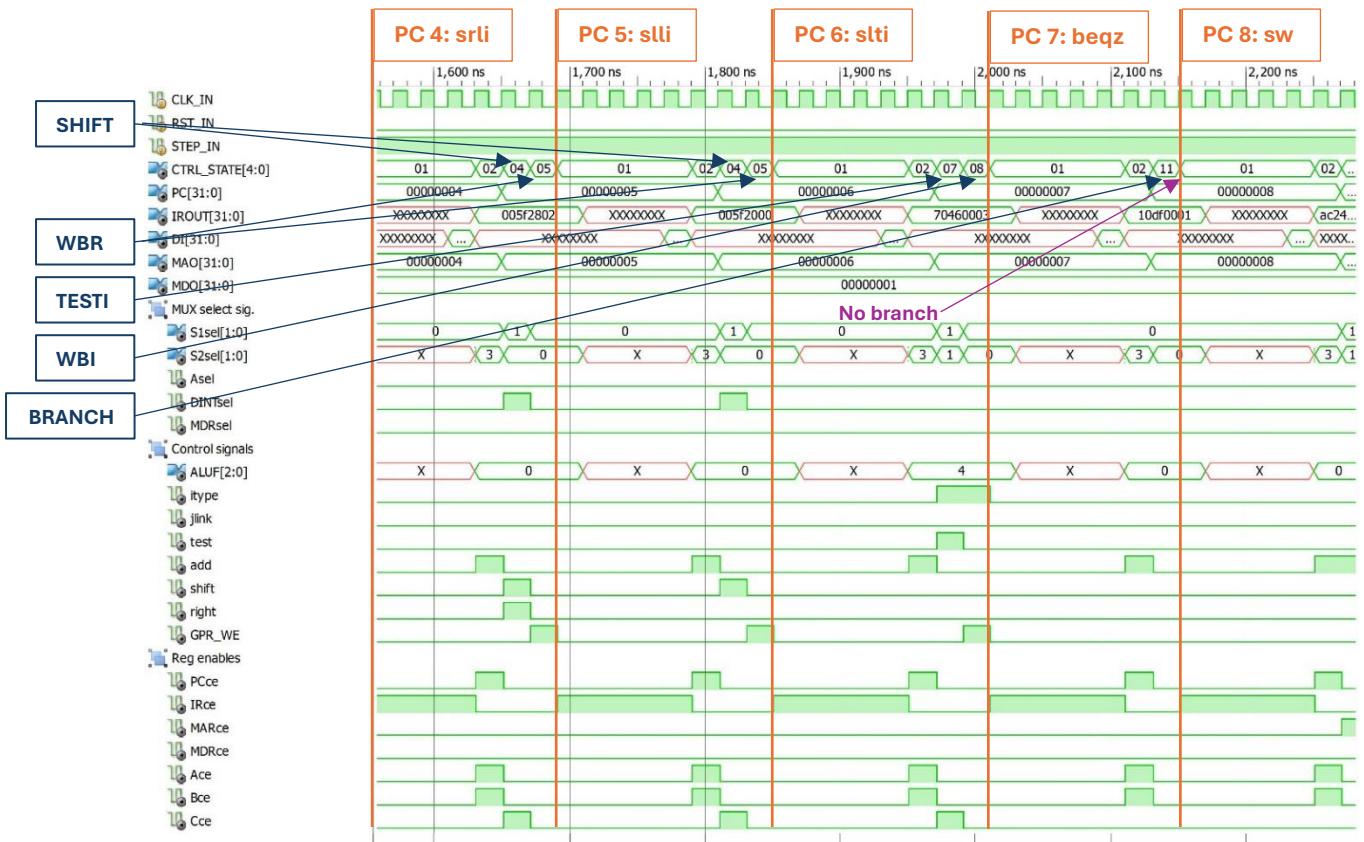
lw R2 R0 data2      INIT → fetch → decode → AddressCMP → LOAD → copyMDR2C → WBI → INIT/fetch
snei R5 R2 0        INIT → fetch → decode → TESTI → WBI → INIT/fetch
slei R6 R4 6        INIT → fetch → decode → TESTI → WBI → INIT/fetch
xor R6 R4 R0       INIT → fetch → decode → ALU → WBR → INIT/fetch
or R2 R6 R4        INIT → fetch → decode → ALU → WBR → INIT/fetch
halt                INIT → fetch → HALT

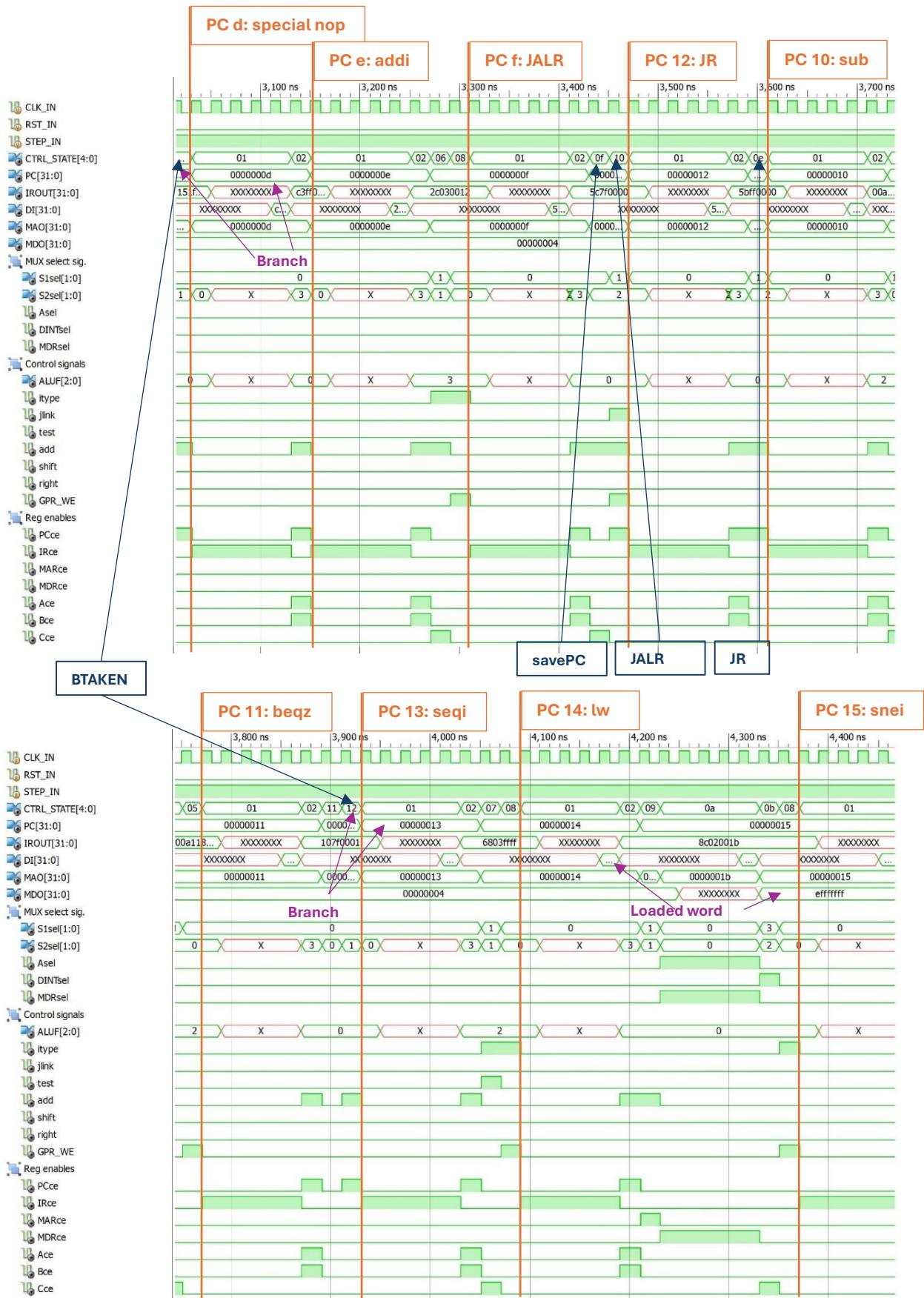
data1: dc 0x1
data2: dc 0xFFFFFFFF
adr1: dc 0x0
adr2: dc 0x0
```

DLX test-bench with the IOsimul:

Simulation waveform results:









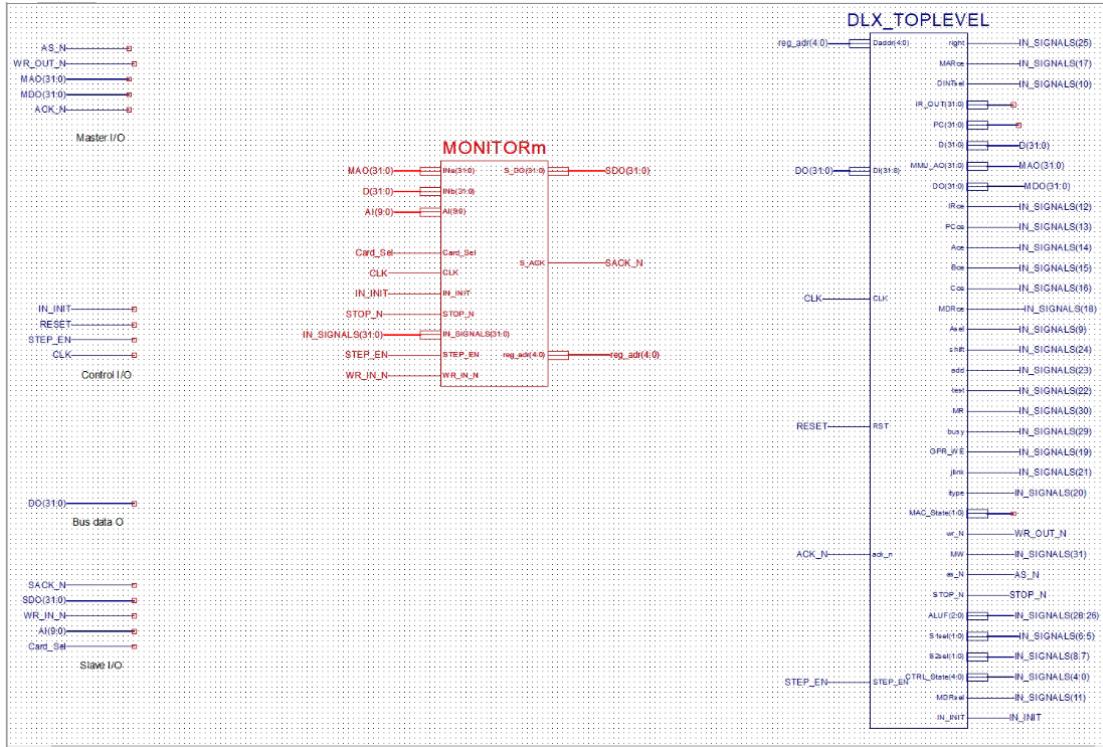
7.2)

The RESA

After the simulation of the DLX machine with the i/o simul we connect the DLX to the i/o logic to upload it to the fpga and simulate it with the RESA program.

We used the same assembly program as we used for the i/o simul simulation.

DLX to i/o logic connection schematic:



After the connection of the DLX to i/o logic we uploaded it to the fpga using the following label configuration:

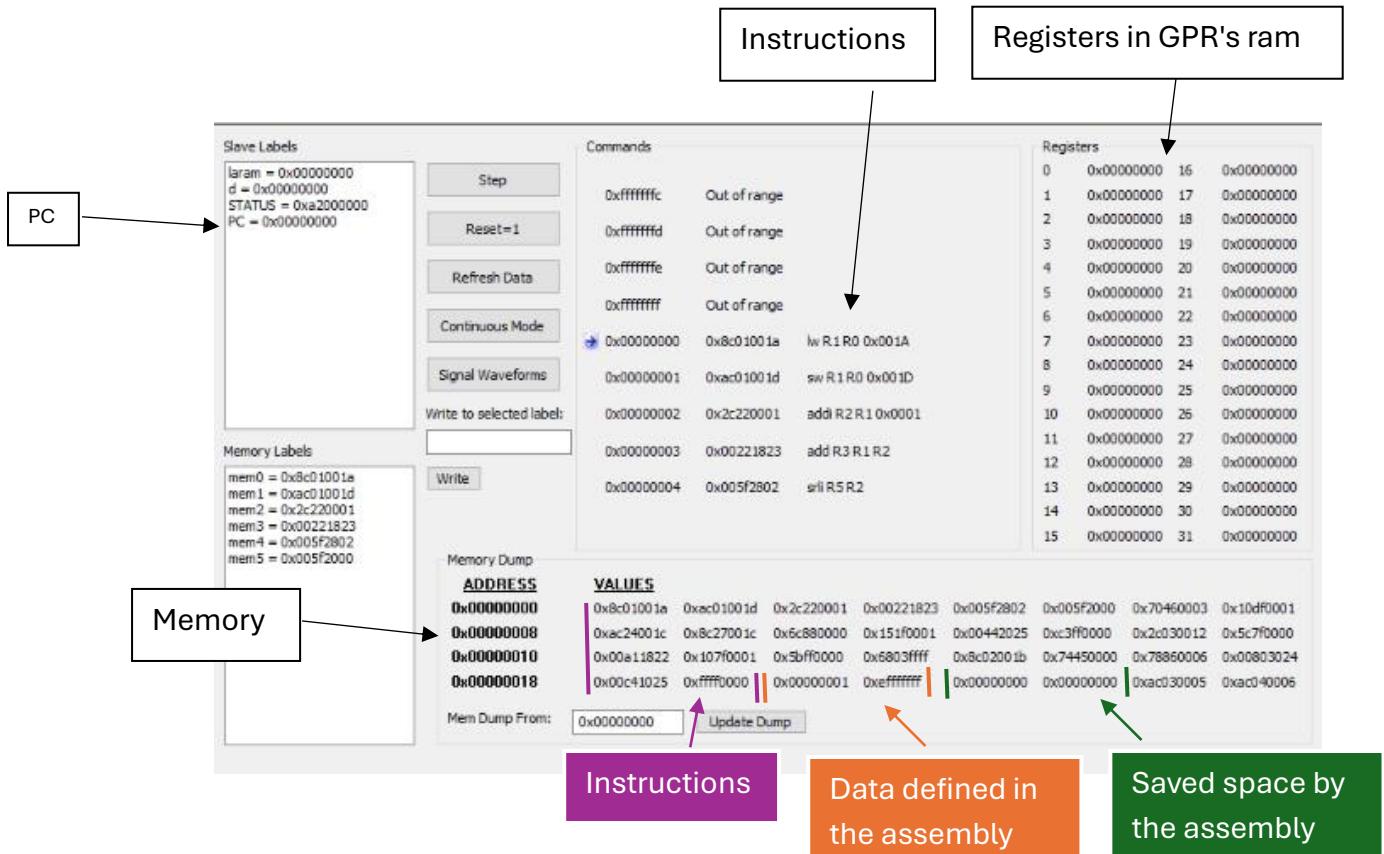
Labels Configuration		
Slave Labels	Graphic labels (From Monitor RAM)	Memory Labels
label@0x1c0 d@0x1a0 STATUS@0xe0 PC@0x180	state@0x0:0 state@0x1 state@0x2 state@0x3 state@0x4 state@0x5 state@0x6 state@0x7 state@0x8 state@0x9 state@0xa state@0xb state@0xc state@0xd state@0xe state@0xf	mem@0x0:0 mem@0x1 mem@0x2 mem@0x3 mem@0x4 mem@0x5
<input type="button" value="Delete Selected Label"/>	<input type="button" value="Delete Selected Label"/>	<input type="button" value="Delete Selected Label"/>
Label name: <input type="text"/> Address: <input type="text"/> <input type="radio"/> Slave <input type="radio"/> Graphic <input type="radio"/> Memory <input type="button" value="Insert/Edit Label"/> <input type="button" value="Clear All Labels"/>	Label Type: <input checked="" type="radio"/> Slave <input type="radio"/> Graphic <input type="radio"/> Memory CPU RAM start address: <input type="text"/> <input type="button" value="Load Labels from file..."/> <input type="button" value="Save Labels to file..."/>	<input type="button" value="Load Labels from file..."/> <input type="button" value="Save Labels to file..."/>
<input type="button" value="Next >"/> <input type="button" value="Cancel"/>		

Labels Configuration		
Slave Labels	Graphic labels (From Monitor RAM)	Memory Labels
label@0x1c0 d@0x1a0 STATUS@0xe0 PC@0x180	roe@0x0 d@0x1e0 ace@0x0 bce@0x1 cce@0x2 mce@0x3 mcce@0x4 men@0x5	mem@0x0:0 mem@0x1 mem@0x2 mem@0x3 mem@0x4 mem@0x5
<input type="button" value="Delete Selected Label"/>	<input type="button" value="Delete Selected Label"/>	<input type="button" value="Delete Selected Label"/>
Label name: <input type="text"/> Address: <input type="text"/> <input type="radio"/> Slave <input type="radio"/> Graphic <input type="radio"/> Memory <input type="button" value="Insert/Edit Label"/> <input type="button" value="Clear All Labels"/>	Label Type: <input checked="" type="radio"/> Slave <input type="radio"/> Graphic <input type="radio"/> Memory CPU RAM start address: <input type="text"/> <input type="button" value="Load Labels from file..."/> <input type="button" value="Save Labels to file..."/>	<input type="button" value="Load Labels from file..."/> <input type="button" value="Save Labels to file..."/>
<input type="button" value="Back <"/> <input type="button" value="Next >"/> <input type="button" value="Cancel"/>		

Labels Configuration		
Slave Labels	Graphic labels (From Monitor RAM)	Memory Labels
label@0x1c0 d@0x1a0 STATUS@0xe0 PC@0x180	type@0x14 j@0x15 test@0x16 add@0x17 shift@0x18 right@0x19 alu@0x1a alu1@0x1b alu2@0x1c bus@0x1d mr@0x1e mw@0x1f	mem@0x0:0 mem@0x1 mem@0x2 mem@0x3 mem@0x4 mem@0x5
<input type="button" value="Delete Selected Label"/>	<input type="button" value="Delete Selected Label"/>	<input type="button" value="Delete Selected Label"/>
Label name: <input type="text"/> Address: <input type="text"/> <input type="radio"/> Slave <input type="radio"/> Graphic <input type="radio"/> Memory <input type="button" value="Insert/Edit Label"/> <input type="button" value="Clear All Labels"/>	Label Type: <input checked="" type="radio"/> Slave <input type="radio"/> Graphic <input type="radio"/> Memory CPU RAM start address: <input type="text"/> <input type="button" value="Load Labels from file..."/> <input type="button" value="Save Labels to file..."/>	<input type="button" value="Load Labels from file..."/> <input type="button" value="Save Labels to file..."/>
<input type="button" value="Back <"/> <input type="button" value="Next >"/> <input type="button" value="Cancel"/>		

In the Graphic labels window, the IN_SIGNALS connections are detailed.

Next, we loaded the relevant bit file and .cod file. The following screen shows the initial values of the fpga memory registers and the gpr's ram memory registers before running and instruction:



Now for each instruction (disregarding repeating instructions) we will show the formula, the values of the fpga and GPR memory registers before and after the running of the instruction and the signal waveforms after executing the instruction and compare them to the test vectors.

PC = 0x0

Instruction: `lw R1 R0 0x001a`

Formula: $R1 = \text{Mem}[0x001a] = 1$

PC = PC + 1 = 1

*The before values the same as the initial values above.

After step enable:

R1 =1

PC = 1

Slave Labels		Commands		Registers	
loram = 0x00000000	d = 0x00000000	Step	0xfffffff0	0	0x00000000
STATUS = 0xa200000c	PC = 0x00000001	Reset=1	0xffffffe0	1	0x00000001
		Refresh Data	0xffffffff	2	0x00000000
		Continuous Mode	0x00000000	3	0x00000000
		Signal Waveforms	0x00000001 0xb01001a	4	0x00000000
		Write to selected label:	lw R1 R0 0x001A	5	0x00000000
Memory Labels			0x00000002 0xac01001d	6	0x00000000
mem0 = 0xb01001a	mem1 = 0xac01001d	Write	sw R1 R0 0x001D	7	0x00000000
mem2 = 0xc220001	mem3 = 0x00221823		0x00000003 0x0221823	8	0x00000000
mem4 = 0x005f2802	mem5 = 0x005f2000		addi R2 R1 0x0001	9	0x00000000
			0x00000004 0x005f2802	10	0x00000000
			srl R5 R2	11	0x00000000
			0x00000005 0x005f2000	12	0x00000000
			slli R4 R2	13	0x00000000
				14	0x00000000
				15	0x00000000

Memory Dump

ADDRESS VALUES

0x00000000	0xb01001a	0xac01001d	0x2c220001	0x00221823	0x005f2802	0x005f2000	0x70460003	0x10df0001
0x00000008	0xac24001c	0xb01001c	0x6c80000	0x151f0001	0x00442025	0xc3f0000	0x2c030012	0x5cf0000
0x00000010	0x00a11822	0x107f0001	0x5bff0000	0x6803ffff	0xbcd2001b	0x74450000	0x78860006	0x00803024
0x00000018	0x00c41025	0xffff0000	0x00000001	0xffffffff	0x00000000	0x00000000	0xac040006	0xffff00b9

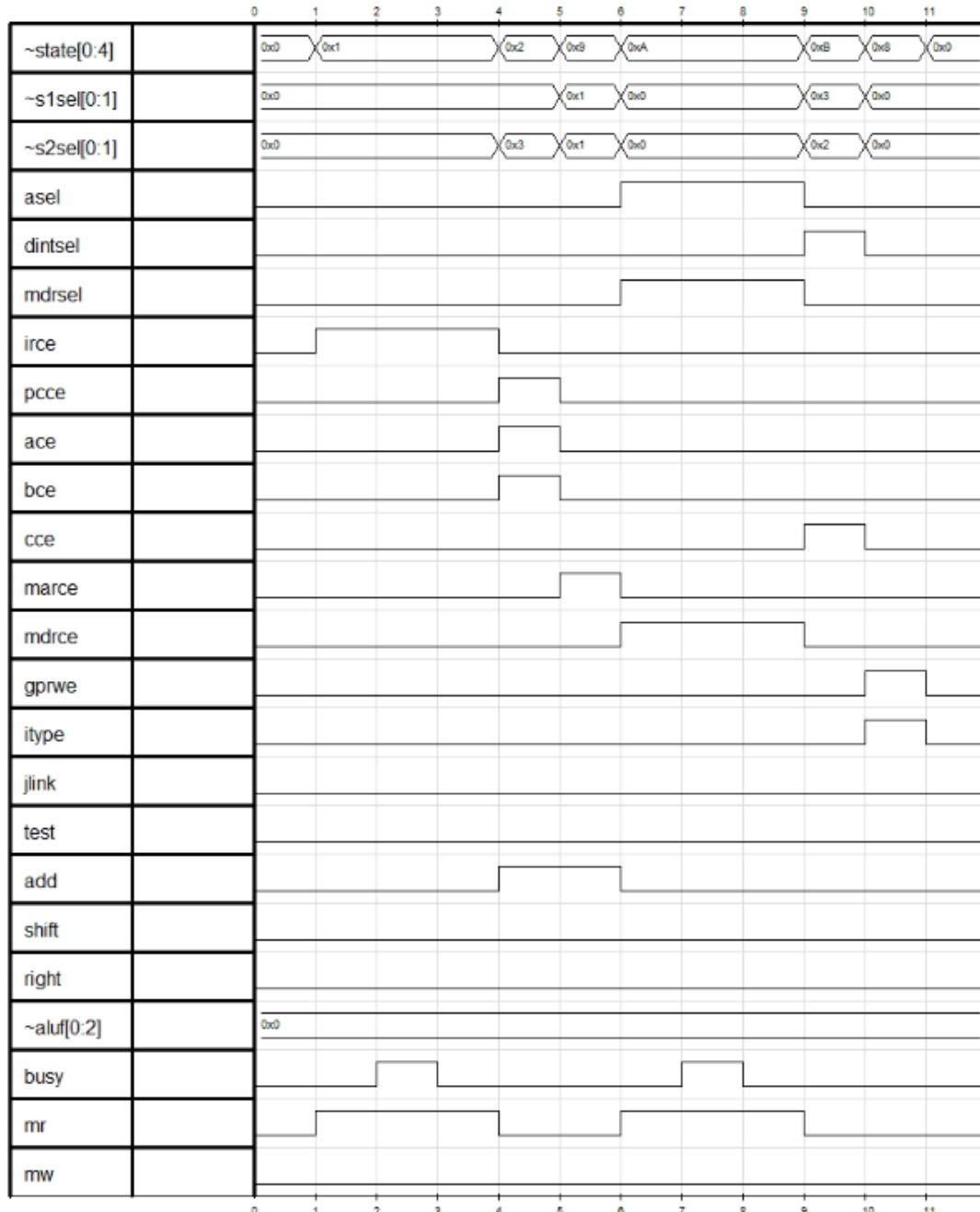
Mem Dump From:

0x00000000

Update Dump

Memory 0x001a

Waveforms:



PC = 0x1

Instruction: sw R1 R0 0x001d

Formula: Mem[0x001d] = R1 = 1

PC = PC + 1 = 2

Before step enable:

Slave Labels

```
taram = 0x00000000
d = 0x00000000
STATUS = 0xa200000c
PC = 0x00000001
```

Commands

- Step
- Reset=1
- Refresh Data
- Continuous Mode
- Signal Waveforms

Write to selected label:

Registers

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0x00000000	16	0x00000000													
1	0x00000001	17	0x00000000													
2	0x00000000	18	0x00000000													
3	0x00000000	19	0x00000000													
4	0x00000000	20	0x00000000													
5	0x00000000	21	0x00000000													
6	0x00000000	22	0x00000000													
7	0x00000000	23	0x00000000													
8	0x00000000	24	0x00000000													
9	0x00000000	25	0x00000000													
10	0x00000000	26	0x00000000													
11	0x00000000	27	0x00000000													
12	0x00000000	28	0x00000000													
13	0x00000000	29	0x00000000													
14	0x00000000	30	0x00000000													
15	0x00000000	31	0x00000000													

Memory Labels

```
mem0 = 0x8c01001a
mem1 = 0xac01001d
mem2 = 0x2c220001
mem3 = 0x00221823
mem4 = 0x005f2802
mem5 = 0x005f2000
```

Write

Registers

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0x00000000	16	0x00000000													
1	0x00000001	17	0x00000000													
2	0x00000000	18	0x00000000													
3	0x00000000	19	0x00000000													
4	0x00000000	20	0x00000000													
5	0x00000000	21	0x00000000													
6	0x00000000	22	0x00000000													
7	0x00000000	23	0x00000000													
8	0x00000000	24	0x00000000													
9	0x00000000	25	0x00000000													
10	0x00000000	26	0x00000000													
11	0x00000000	27	0x00000000													
12	0x00000000	28	0x00000000													
13	0x00000000	29	0x00000000													
14	0x00000000	30	0x00000000													
15	0x00000000	31	0x00000000													

Memory Dump

ADDRESS	VALUES
0x8c01001a	0xac01001d 0x2c220001 0x00221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0xac24001c	0x8c27001c 0x6c880000 0x151f0001 0x00442025 0xc3ff0000 0x2c030012 0x5c7f0000
0x00a11822	0x107f0001 0x5bff0000 0x6803ffff 0x8c02001b 0x74450000 0x78860006 0x00803024
0x00c41025	0xfffff0000 0x00000001 0xffffffff 0x00000000 0x00000001 0xac040006 0xfffff00b9

Mem Dump From: 0x00000000 Update Dump

R1 = 1

Memory 0x001d = 0

After step enable:

Slave Labels

```
taram = 0x00000000
d = 0x00000000
STATUS = 0xa200000b
PC = 0x00000002
```

Commands

- Step
- Reset=1
- Refresh Data
- Continuous Mode
- Signal Waveforms

Write to selected label:

Registers

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0x00000000	16	0x00000000													
1	0x00000001	17	0x00000000													
2	0x00000000	18	0x00000000													
3	0x00000000	19	0x00000000													
4	0x00000000	20	0x00000000													
5	0x00000000	21	0x00000000													
6	0x00000000	22	0x00000000													
7	0x00000000	23	0x00000000													
8	0x00000000	24	0x00000000													
9	0x00000000	25	0x00000000													
10	0x00000000	26	0x00000000													
11	0x00000000	27	0x00000000													
12	0x00000000	28	0x00000000													
13	0x00000000	29	0x00000000													
14	0x00000000	30	0x00000000													
15	0x00000000	31	0x00000000													

Memory Labels

```
mem0 = 0x8c01001a
mem1 = 0xac01001d
mem2 = 0x2c220001
mem3 = 0x00221823
mem4 = 0x005f2802
mem5 = 0x005f2000
```

Write

Registers

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0x00000000	16	0x00000000													
1	0x00000001	17	0x00000000													
2	0x00000000	18	0x00000000													
3	0x00000000	19	0x00000000													
4	0x00000000	20	0x00000000													
5	0x00000000	21	0x00000000													
6	0x00000000	22	0x00000000													
7	0x00000000	23	0x00000000													
8	0x00000000	24	0x00000000													
9	0x00000000	25	0x00000000													
10	0x00000000	26	0x00000000													
11	0x00000000	27	0x00000000													
12	0x00000000	28	0x00000000													
13	0x00000000	29	0x00000000													
14	0x00000000	30	0x00000000													
15	0x00000000	31	0x00000000													

Memory Dump

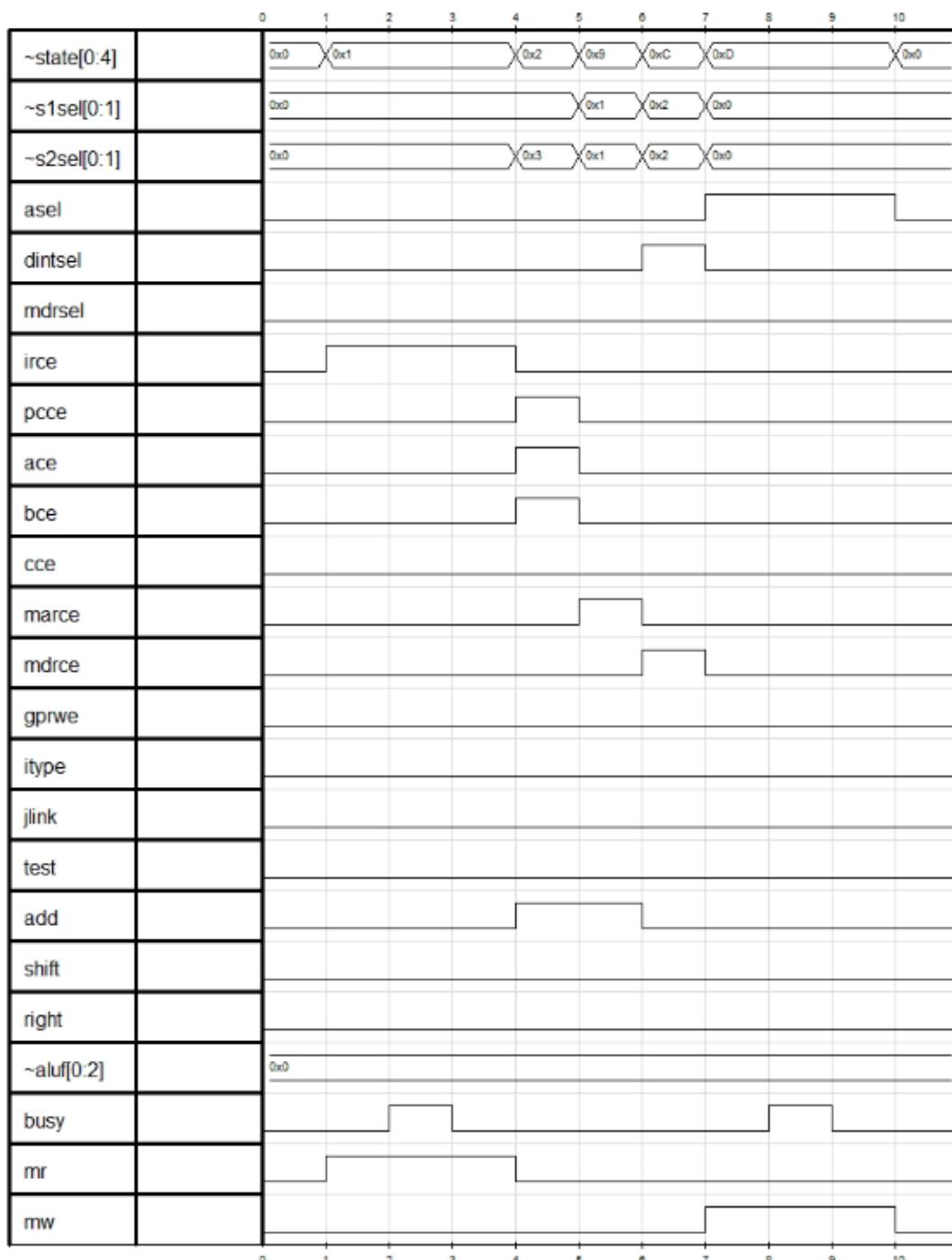
ADDRESS	VALUES
0x8c01001a	0xac01001d 0x2c220001 0x00221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0xac24001c	0x8c27001c 0x6c880000 0x151f0001 0x00442025 0xc3ff0000 0x2c030012 0x5c7f0000
0x00a11822	0x107f0001 0x5bff0000 0x6803ffff 0x8c02001b 0x74450000 0x78860006 0x00803024
0x00c41025	0xfffff0000 0x00000001 0xffffffff 0x00000000 0x00000001 0xac040006 0xfffff00b9

Mem Dump From: 0x00000000 Update Dump

R1 = 1

Memory 0x001d = 1

Waveforms:



PC = 0x2

Instruction: addi R2 R1 0x00001

Formula: R2 = R1 + 1 = 2

PC = PC + 1 = 3

Before step enable:

The screenshot shows a debugger interface with the following details:

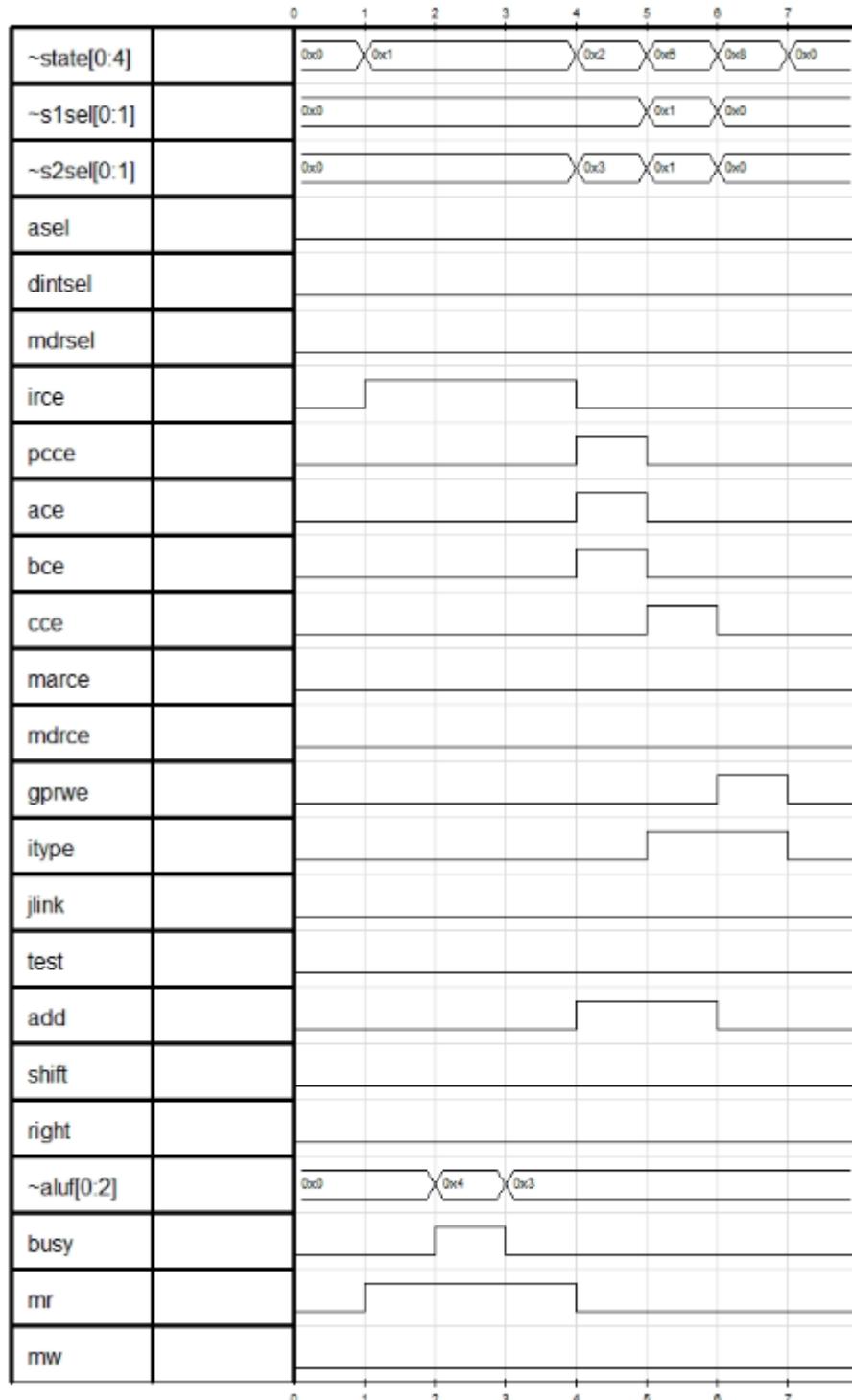
- Registers:** R2 = 0, R1 = 1.
- Slave Labels:** laram = 0x00000000, d = 0x00000000, STATUS = 0xa200000b, PC = 0x00000002.
- Memory Labels:** mem0 = 0x8c01001a, mem1 = 0xac01001d, mem2 = 0x2c220001, mem3 = 0x00221823, mem4 = 0x005f2802, mem5 = 0x005f2000.
- Commands:** Step (highlighted), Reset=1, Refresh Data, Continuous Mode, Signal Waveforms, Write to selected label: [].
- Registers Table:** Shows 16 registers from 0 to 31. Register 1 contains the value 1.
- Memory Dump:** Shows memory dump from address 0x00000000 to 0x00000015. The dump includes values for addresses 0x8c01001a, 0xac24001c, 0x00000008, 0x00000010, and 0x00000018.
- Buttons:** Mem Dump From: 0x00000000, Update Dump.

After step enable:

The screenshot shows the debugger interface after the step command was enabled:

- Registers:** R2 = 2, R1 = 1.
- Slave Labels:** laram = 0x00000000, d = 0x00000000, STATUS = 0xa2000008, PC = 0x00000003.
- Memory Labels:** mem0 = 0x8c01001a, mem1 = 0xac01001d, mem2 = 0x2c220001, mem3 = 0x00221823, mem4 = 0x005f2802, mem5 = 0x005f2000.
- Commands:** Step (highlighted), Reset=1, Refresh Data, Continuous Mode, Signal Waveforms, Write to selected label: [].
- Registers Table:** Shows 16 registers from 0 to 31. Register 1 now contains the value 2.
- Memory Dump:** Shows memory dump from address 0x00000000 to 0x00000015. The dump includes values for addresses 0x8c01001a, 0xac24001c, 0x00000008, 0x00000010, and 0x00000018.
- Buttons:** Mem Dump From: 0x00000000, Update Dump.

Waveforms:



PC = 0x3

Instruction: add R3 R1 R2

Formula: R3 = R1 + R2 = 3

PC = PC + 1 = 4

Before step enable:

Registers

0	0x00000000	16	0x00000000
1	0x00000001	17	0x00000000
2	0x00000002	18	0x00000000
3	0x00000000	19	0x00000000
4	0x00000000	20	0x00000000
5	0x00000000	21	0x00000000
6	0x00000000	22	0x00000000
7	0x00000000	23	0x00000000
8	0x00000000	24	0x00000000
9	0x00000000	25	0x00000000
10	0x00000000	26	0x00000000
11	0x00000000	27	0x00000000
12	0x00000000	28	0x00000000
13	0x00000000	29	0x00000000
14	0x00000000	30	0x00000000
15	0x00000000	31	0x00000000

Slave Labels

laram = 0x00000000
d = 0x00000000
STATUS = 0xa2000008
PC = 0x00000003

Commands

Step
Reset=1
Refresh Data
Continuous Mode
Signal Waveforms
Write to selected label:
Write

Memory Labels

mem0 = 0x8c01001a
mem1 = 0xac01001d
mem2 = 0x2c220001
mem3 = 0x00221823
mem4 = 0x005f2802
mem5 = 0x005f2000

Memory Dump

ADDRESS	VALUES
0x00000000	0x8c01001a 0xac01001d 0x2c220001 0x00221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0x00000008	0xac24001c 0x8c27001c 0x6c880000 0x151f0001 0x00442025 0xc3ff0000 0x2c030012 0x5c7f0000
0x00000010	0x00a11822 0x107f0001 0x5bfff0000 0x6803ffff 0x8c02001b 0x74450000 0x78860006 0x00803024
0x00000018	0x00c41025 0xffff0000 0x00000001 0xffffffff 0x00000000 0x00000001 0xac040006 0xffff7400

Mem Dump From: 0x00000000 Update Dump

After step enable:

Registers

0	0x00000000	16	0x00000000
1	0x00000001	17	0x00000000
2	0x00000002	18	0x00000000
3	0x00000003	19	0x00000000
4	0x00000000	20	0x00000000
5	0x00000000	21	0x00000000
6	0x00000000	22	0x00000000
7	0x00000000	23	0x00000000
8	0x00000000	24	0x00000000
9	0x00000000	25	0x00000000
10	0x00000000	26	0x00000000
11	0x00000000	27	0x00000000
12	0x00000000	28	0x00000000
13	0x00000000	29	0x00000000
14	0x00000000	30	0x00000000
15	0x00000000	31	0x00000000

Slave Labels

laram = 0x0c010000
d = 0x00000000
STATUS = 0xa2000008
PC = 0x00000004

Commands

Step
Reset=1
Refresh Data
Continuous Mode
Signal Waveforms
Write to selected label:
Write

Memory Labels

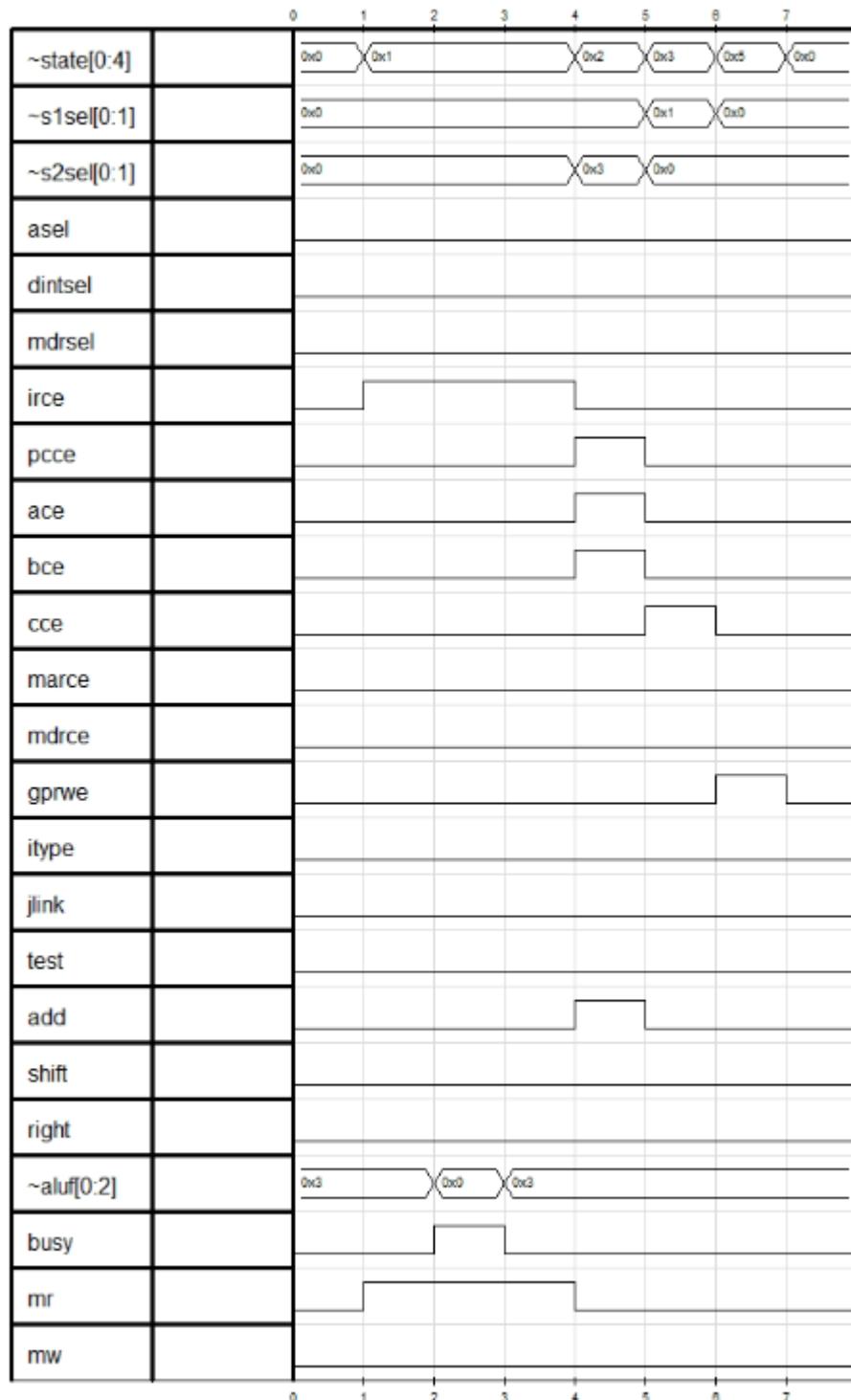
mem0 = 0x8c01001a
mem1 = 0xac01001d
mem2 = 0x2c220001
mem3 = 0x00221823
mem4 = 0x005f2802
mem5 = 0x005f2000

Memory Dump

ADDRESS	VALUES
0x00000000	0x8c01001a 0xac01001d 0x2c220001 0x00221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0x00000008	0xac24001c 0x8c27001c 0x6c880000 0x151f0001 0x00442025 0xc3ff0000 0x2c030012 0x5c7f0000
0x00000010	0x00a11822 0x107f0001 0x5bfff0000 0x6803ffff 0x8c02001b 0x74450000 0x78860006 0x00803024
0x00000018	0x00c41025 0xffff0000 0x00000001 0xffffffff 0x00000000 0x00000001 0xac040006 0xffff7400

Mem Dump From: 0x00000000 Update Dump

Waveforms:



PC = 0x4

Instruction: srl R5 R2

Formula: R5 = R2 >> 1 = 1

PC = PC + 1 = 5

Before step enable:

R5 = 0 R2 = 2

The screenshot shows a debugger interface with the following details:

- Registers:** Shows 16 registers (R0-R15) all set to 0x00000000.
- Commands:** Step button is highlighted. Other buttons include Reset=1, Refresh Data, Continuous Mode, Signal Waveforms, Write to selected label:, Write, and Memory Dump.
- Memory Labels:** mem0 = 0xb01001a, mem1 = 0xac01001d, mem2 = 0x2c220001, mem3 = 0x00221823, mem4 = 0x005f2802, mem5 = 0x005f2000.
- Registers:** Shows the state of the registers before the instruction is executed.
- Memory Dump:** Displays memory dump from address 0x00000000 to 0x00000018.

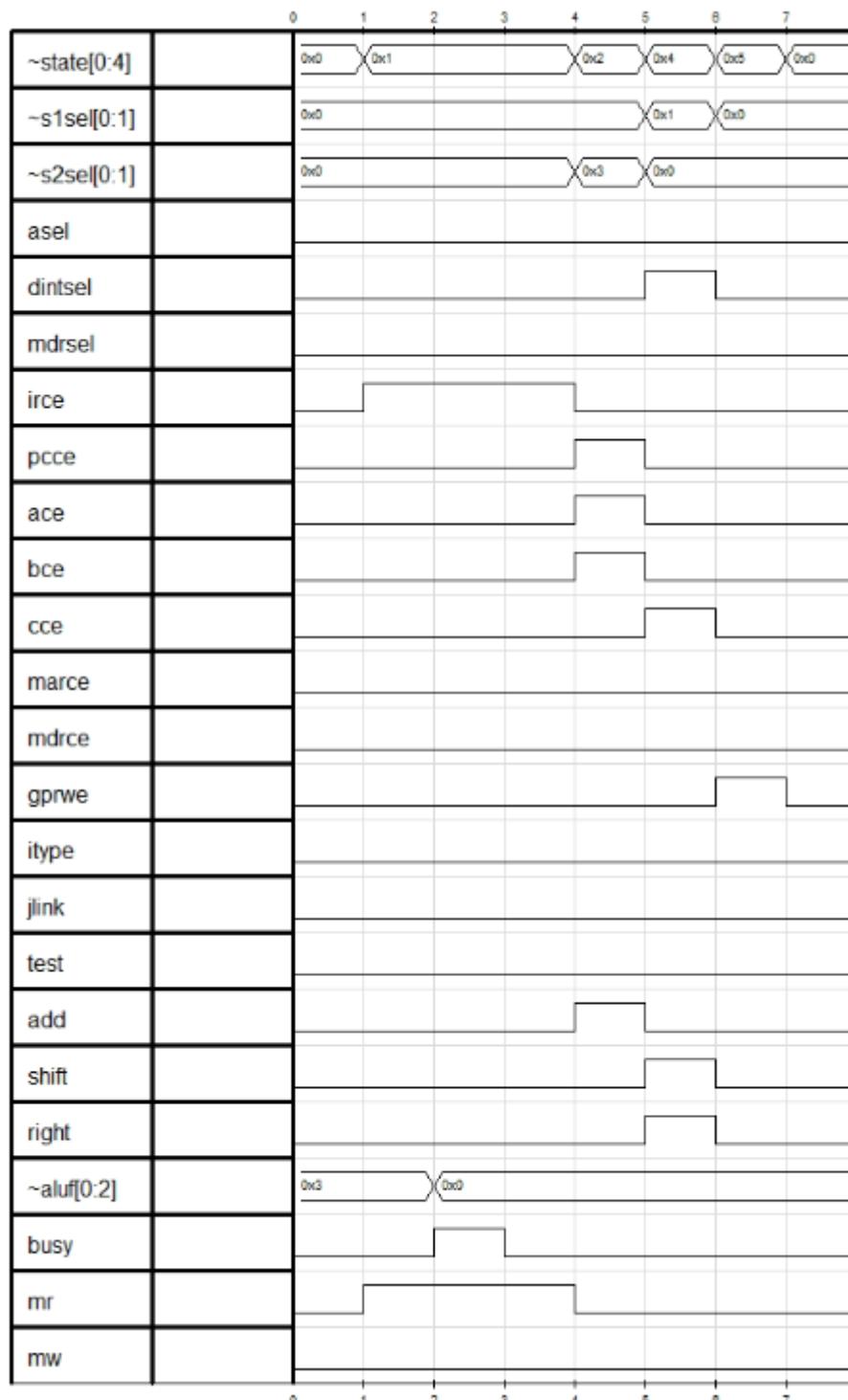
After step enable:

R5 = 1

The screenshot shows the debugger interface after the instruction has been executed:

- Registers:** Register R5 is now 0x00000001, while others remain at 0x00000000.
- Commands:** Step button is highlighted. Other buttons include Reset=1, Refresh Data, Continuous Mode, Signal Waveforms, Write to selected label:, Write, and Memory Dump.
- Memory Labels:** mem0 = 0xb01001a, mem1 = 0xac01001d, mem2 = 0x2c220001, mem3 = 0x00221823, mem4 = 0x005f2802, mem5 = 0x005f2000.
- Registers:** Shows the state of the registers after the instruction was executed.
- Memory Dump:** Displays memory dump from address 0x00000000 to 0x00000018.

Waveforms:



PC = 0x5

Instruction: slli R4 R2

Formula: R4 = R2 << 1 = 4

PC = PC + 1 = 6

Before step enable:

R4 = 0 R2 = 2

The screenshot shows a debugger interface with the following sections:

- Registers:** Shows 16-bit register values from 0 to 15. Register 0 is 0x00000000, Register 1 is 0x00000001, Register 2 is 0x00000002, Register 3 is 0x00000003, Register 4 is 0x00000004, Register 5 is 0x00000001, Register 6 is 0x00000000, Register 7 is 0x00000000, Register 8 is 0x00000000, Register 9 is 0x00000000, Register 10 is 0x00000000, Register 11 is 0x00000000, Register 12 is 0x00000000, Register 13 is 0x00000000, Register 14 is 0x00000000, Register 15 is 0x00000000.
- Commands:** A list of assembly instructions. The 5th instruction is highlighted with a blue arrow and the value 0x005f2000. The instruction is `slli R4 R2`.
- Memory Dump:** A table showing memory values at addresses 0x00000000, 0x00000008, 0x00000010, and 0x00000018.
- Slave Labels:** A list of labels: laram = 0x0c000000, d = 0x00000000, STATUS = 0xa2000008, PC = 0x00000005.
- Memory Labels:** A list of labels: mem0 = 0xb0c01001a, mem1 = 0xac0c1001d, mem2 = 0xc2c20001, mem3 = 0x00221823, mem4 = 0x005f2802, mem5 = 0x005f2000.
- Buttons:** Step, Reset=1, Refresh Data, Continuous Mode, Signal Waveforms, Write.

A box labeled "PC = 5" is shown on the left side of the interface.

After step enable:

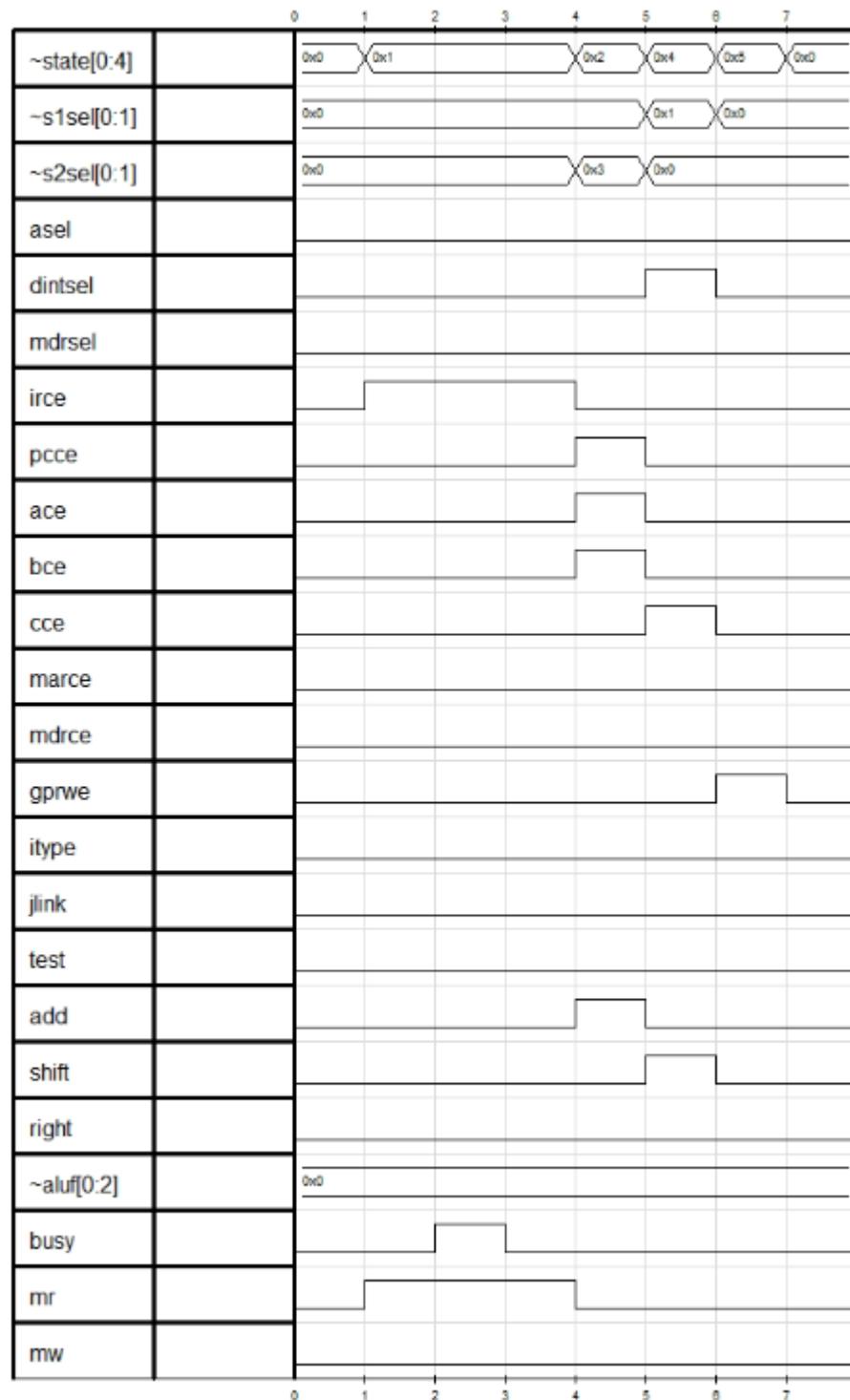
R4 = 4

The screenshot shows the debugger interface after the step operation. The state has changed as follows:

- Registers:** Register 1 (R4) now contains the value 0x00000004.
- Commands:** The 5th instruction is now highlighted with a blue arrow and the value 0x70460003. The instruction is `slli R6 R2`.
- Memory Dump:** The memory dump table remains the same as before.
- Slave Labels:** The PC value has changed to 0x00000006.
- Memory Labels:** The PC value has changed to 0x00000006.
- Buttons:** Step, Reset=1, Refresh Data, Continuous Mode, Signal Waveforms, Write.

A box labeled "PC = 6" is shown on the left side of the interface.

Waveforms:



PC = 0x6

Instruction: slti R6 R2 0x003

Formula: R6 = (R2 < 3) ? 1:0 => R6 = 1

PC = PC + 1 = 7

Before step enable:

R6 = 0 R2 = 2

Slave Labels

```
laram = 0x00000000
d = 0x00000000
STATUS = 0xa2000008
PC = 0x00000006
```

Commands

0x00000002	0x2c220001	addi R2 R1 0x0001
0x00000003	0x00221823	add R3 R1 R2
0x00000004	0x005f2802	srl R5 R2
0x00000005	0x005f2000	sll R4 R2
0x00000006	0x70460003	slti R6 R2 0x0003
0x00000007	0x10df0001	beqz R6 0x0009
0x00000008	0xac24001c	sw R4 R1 0x001C
0x00000009	0x8c27001c	lw R7 R1 0x001C
0x0000000a	0x6c880000	sgei R8 R4 0x0000
0x0000000b	0x151f0001	bnez R8 0x000D

Registers

0	0x00000000	16	0x00000000
1	0x00000001	17	0x00000000
2	0x00000002	18	0x00000000
3	0x00000003	19	0x00000000
4	0x00000004	20	0x00000000
5	0x00000001	21	0x00000000
6	0x00000000	22	0x00000000
7	0x00000000	23	0x00000000
8	0x00000000	24	0x00000000
9	0x00000000	25	0x00000000
10	0x00000000	26	0x00000000
11	0x00000000	27	0x00000000
12	0x00000000	28	0x00000000
13	0x00000000	29	0x00000000
14	0x00000000	30	0x00000000
15	0x00000000	31	0x00000000

Memory Labels

```
mem0 = 0x8c01001a
mem1 = 0xac01001d
mem2 = 0x2c220001
mem3 = 0x00221823
mem4 = 0x005f2802
mem5 = 0x005f2000
```

Write to selected label:

Write

Memory Dump

ADDRESS	VALUES
0x00000000	0x8c01001a 0xac01001d 0x2c220001 0x00221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0x00000008	0xac24001c 0x8c27001c 0x6c880000 0x151f0001 0x00442025 0xc3ff0000 0x2c030012 0x5c7f0000
0x00000010	0x00a11822 0x107f0001 0xbffff000 0x6803ffff 0x8c02001b 0x74450000 0x78860006 0x00803024
0x00000018	0x00c41025 0xfffff0000 0x00000001 0xffffffff 0x00000000 0x00000001 0xac040006 0xfffff7400

Mem Dump From: 0x00000000 Update Dump

After step enable:

R6 = 1

Slave Labels

```
laram = 0x00000000
d = 0x00000000
STATUS = 0xa2000008
PC = 0x00000007
```

Commands

0x00000003	0x00221823	add R3 R1 R2
0x00000004	0x005f2802	srl R5 R2
0x00000005	0x005f2000	sll R4 R2
0x00000006	0x70460003	slti R6 R2 0x0003
0x00000007	0x10df0001	beqz R6 0x0009
0x00000008	0xac24001c	sw R4 R1 0x001C
0x00000009	0x8c27001c	lw R7 R1 0x001C
0x0000000a	0x6c880000	sgei R8 R4 0x0000
0x0000000b	0x151f0001	bnez R8 0x000D

Registers

0	0x00000000	16	0x00000000
1	0x00000001	17	0x00000000
2	0x00000002	18	0x00000000
3	0x00000003	19	0x00000000
4	0x00000004	20	0x00000000
5	0x00000001	21	0x00000000
6	0x00000001	22	0x00000000
7	0x00000000	23	0x00000000
8	0x00000000	24	0x00000000
9	0x00000000	25	0x00000000
10	0x00000000	26	0x00000000
11	0x00000000	27	0x00000000
12	0x00000000	28	0x00000000
13	0x00000000	29	0x00000000
14	0x00000000	30	0x00000000
15	0x00000000	31	0x00000000

Memory Labels

```
mem0 = 0x8c01001a
mem1 = 0xac01001d
mem2 = 0x2c220001
mem3 = 0x00221823
mem4 = 0x005f2802
mem5 = 0x005f2000
```

Write to selected label:

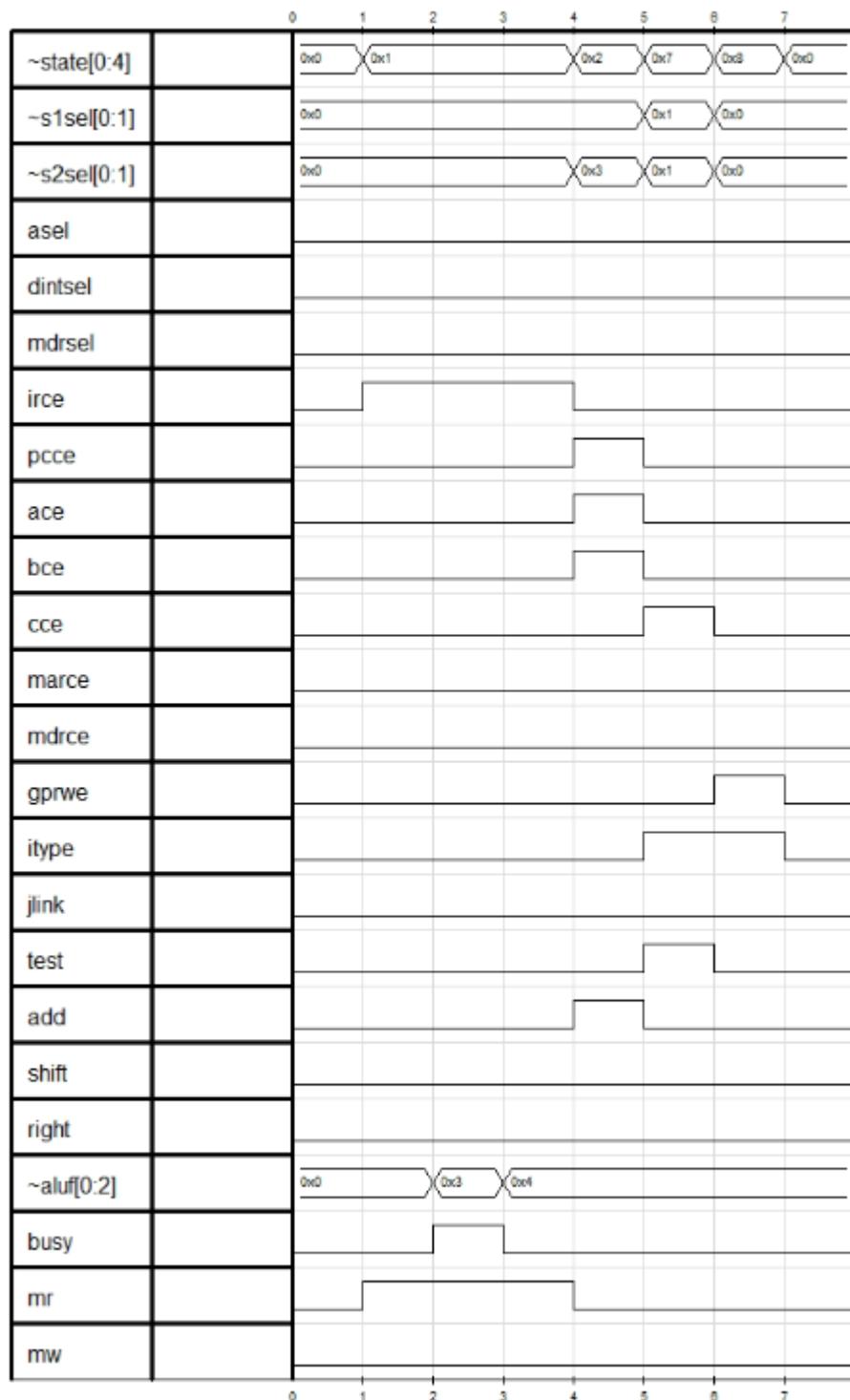
Write

Memory Dump

ADDRESS	VALUES
0x00000000	0x8c01001a 0xac01001d 0x2c220001 0x00221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0x00000008	0xac24001c 0x8c27001c 0x6c880000 0x151f0001 0x00442025 0xc3ff0000 0x2c030012 0x5c7f0000
0x00000010	0x00a11822 0x107f0001 0xbffff000 0x6803ffff 0x8c02001b 0x74450000 0x78860006 0x00803024
0x00000018	0x00c41025 0xfffff0000 0x00000001 0xffffffff 0x00000000 0x00000001 0xac040006 0xfffff7400

Mem Dump From: 0x00000000 Update Dump

Waveforms:



PC = 0x7

Instruction: beqz R6 R2 0x009

Formula: PC = (!R6) ? 0x009 : (pc+1)

PC = PC + 1 = 8

Before step enable:

PC = 7

R6 = 1

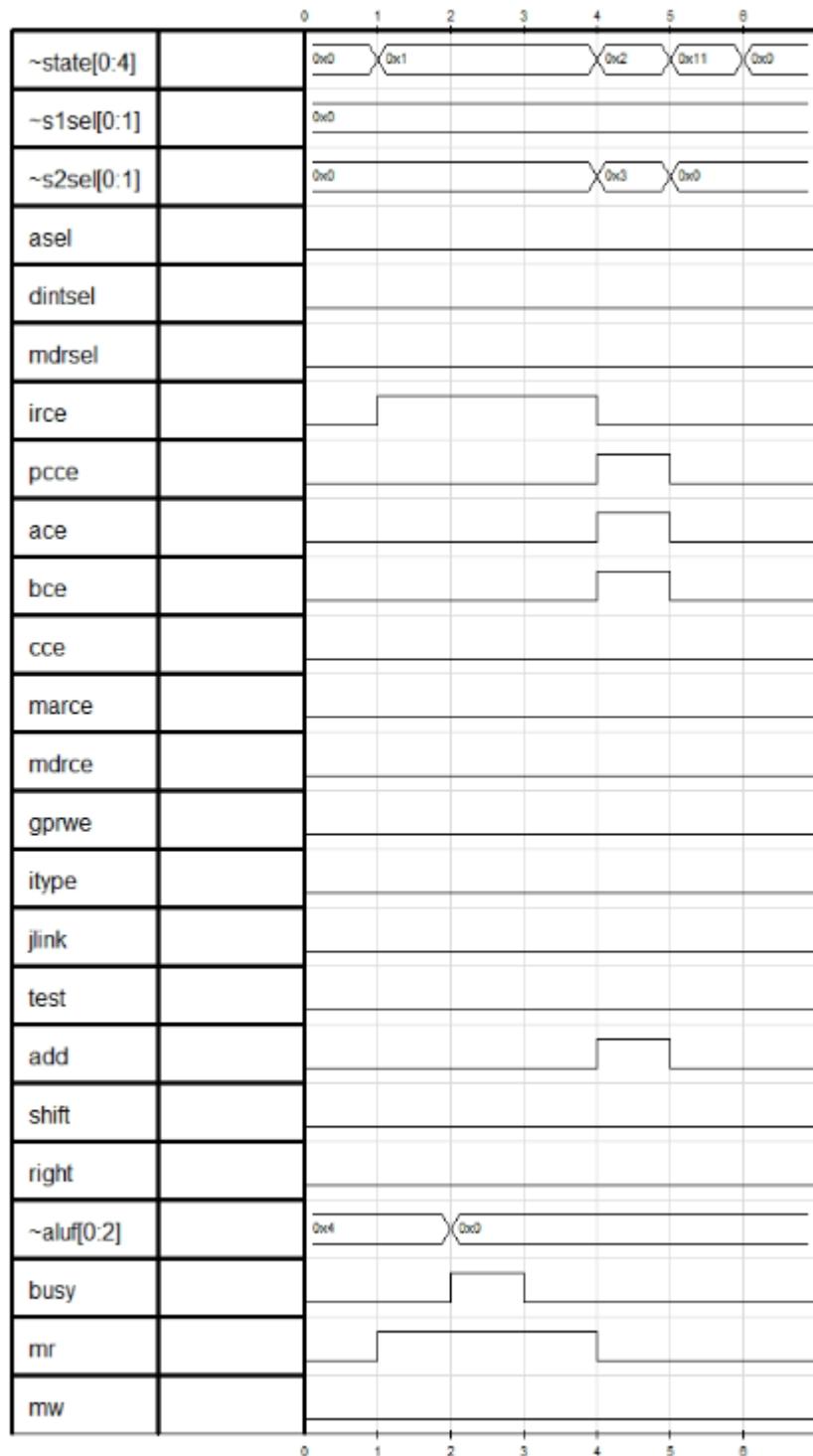
	Registers
0	0x00000000 16 0x00000000
1	0x00000001 17 0x00000000
2	0x00000002 18 0x00000000
3	0x00000003 19 0x00000000
4	0x00000004 20 0x00000000
5	0x00000001 21 0x00000000
6	0x00000001 22 0x00000000
7	0x00000000 23 0x00000000
8	0x00000000 24 0x00000000
9	0x00000000 25 0x00000000
10	0x00000000 26 0x00000000
11	0x00000000 27 0x00000000
12	0x00000000 28 0x00000000
13	0x00000000 29 0x00000000
14	0x00000000 30 0x00000000
15	0x00000000 31 0x00000000

After step enable:

PC = 8

	Registers
0	0x00000000 16 0x00000000
1	0x00000001 17 0x00000000
2	0x00000002 18 0x00000000
3	0x00000003 19 0x00000000
4	0x00000004 20 0x00000000
5	0x00000001 21 0x00000000
6	0x00000001 22 0x00000000
7	0x00000000 23 0x00000000
8	0x00000000 24 0x00000000
9	0x00000000 25 0x00000000
10	0x00000000 26 0x00000000
11	0x00000000 27 0x00000000
12	0x00000000 28 0x00000000
13	0x00000000 29 0x00000000
14	0x00000000 30 0x00000000
15	0x00000000 31 0x00000000

Waveforms:



PC = 0xa

Instruction: sgei R8 R4 0x000

Formula: R8 = (R4 >= 0) ? 1 : 0 => R8 = 1

PC = PC + 1 = b

Before step enable:

PC = a

R8 = 0

R4 = 4

The screenshot shows a debugger interface with the following sections:

- Slave Labels:** laram = 0x00000000, d = 0x00000000, STATUS = 0xa2000000, PC = 0x0000000a.
- Commands:** Step, Reset=1, Refresh Data, Continuous Mode, Signal Waveforms, Write to selected label:, Write.
- Registers:** A table showing 16 registers (0-15) with their values. Register 4 has a value of 4.
- Memory Labels:** mem0 = 0xb0c01001a, mem1 = 0xac0c1001d, mem2 = 0x2c220001, mem3 = 0x00221823, mem4 = 0x005f2802, mem5 = 0x005f2000.
- Memory Dump:** ADDRESS, VALUES. It shows memory starting at address 0x00000000 with various values.
- Mem Dump From:** 0x00000000, Update Dump.

Annotations with arrows point from the text labels "PC = a", "R8 = 0", and "R4 = 4" to the corresponding fields in the debugger interface.

After step enable:

PC = b

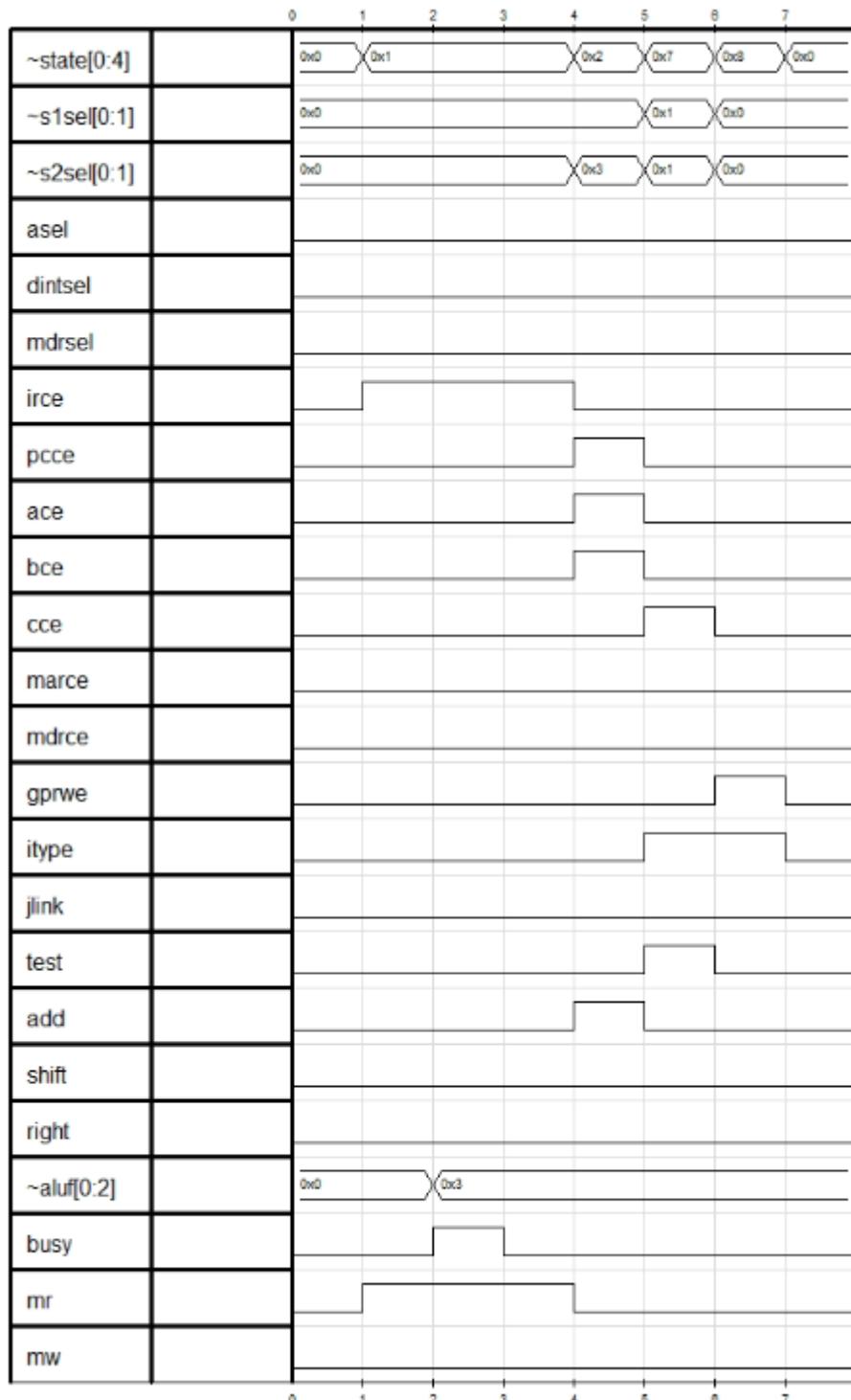
R8 = 1

The screenshot shows the debugger interface after the step command was executed. The state has changed as follows:

- Slave Labels:** laram = 0x00000000, d = 0x00000000, STATUS = 0xa2000008, PC = 0x0000000b.
- Registers:** A table showing 16 registers (0-15). Register 4 has been updated to 1.
- Memory Labels:** mem0 = 0xb0c01001a, mem1 = 0xac0c1001d, mem2 = 0x2c220001, mem3 = 0x00221823, mem4 = 0x005f2802, mem5 = 0x005f2000.
- Memory Dump:** ADDRESS, VALUES. It shows memory starting at address 0x00000000 with various values.
- Mem Dump From:** 0x00000000, Update Dump.

Annotations with arrows point from the text label "PC = b" and the register value "R8 = 1" to the corresponding fields in the debugger interface.

Waveforms:



PC = 0xb

Instruction: bnez R8 0x000d

Formula: PC = (R8) ? 0x000d : (pc+1)

PC = 0x000d

Before step enable:

R8 = 1

PC = b

Slave Labels

- laram = 0x00000000
- d = 0x00000000
- STATUS = 0xa2000008
- PC = 0x0000000b

Commands

Address	Value	Comment
0x00000007	0x10df0001	beqz R6 0x0009
0x00000008	0xac24001c	sw R4 R1 0x001C
0x00000009	0x8c27001c	lw R7 R1 0x001C
0x0000000a	0x6c880000	sgel R8 R4 0x0000
0x0000000b	0x151f0001	bnez R8 0x000D
0x0000000c	0x0442025	or R4 R2 R4
0x0000000d	0xc3ff0000	special-nop
0x0000000e	0x2c030012	addi R3 R0 0x0012
0x0000000f	0x5c7f0000	jalr R3

Registers

Register	Value	Label
0	0x00000000	16
1	0x00000001	17
2	0x00000002	18
3	0x00000003	19
4	0x00000004	20
5	0x00000001	21
6	0x00000001	22
7	0x00000004	23
8	0x00000001	24
9	0x00000000	25
10	0x00000000	26
11	0x00000000	27
12	0x00000000	28
13	0x00000000	29
14	0x00000000	30
15	0x00000000	31

Memory Labels

- mem0 = 0x8c01001a
- mem1 = 0xac01001d
- mem2 = 0x2c220001
- mem3 = 0x00221823
- mem4 = 0x005f2802
- mem5 = 0x005f2000

Memory Dump

Address	Values
0x00000000	0x8c01001a 0xac01001d 0x2c220001 0x00221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0x00000008	0xac24001c 0x8c27001c 0x6c880000 0x151f0001 0x0442025 0xc3ff0000 0x2c030012 0x5c7f0000
0x00000010	0x00a11822 0x107f0001 0x5bff0000 0x6803ffff 0x8c02001b 0x74450000 0x78860006 0x00803024
0x00000018	0x00c41025 0xfffff0000 0x00000001 0xffffffff 0x00000000 0x00000004 0xac040006 0xffff7400

Mem Dump From: 0x00000000 Update Dump

After step enable:

PC = d

Slave Labels

- laram = 0x00000000
- d = 0x00000000
- STATUS = 0xa2000008
- PC = 0x0000000d

Commands

Address	Value	Comment
0x00000009	0x8c27001c	lw R7 R1 0x001C
0x0000000a	0x6c880000	sgel R8 R4 0x0000
0x0000000b	0x151f0001	bnez R8 0x000D
0x0000000c	0x0442025	or R4 R2 R4
0x0000000d	0xc3ff0000	special-nop
0x0000000e	0x2c030012	addi R3 R0 0x0012
0x0000000f	0x5c7f0000	jalr R3
0x00000010	0x00a11822	sub R3 R5 R1
0x00000011	0x107f0001	beqz R3 0x0013

Registers

Register	Value	Label
0	0x00000000	16
1	0x00000001	17
2	0x00000002	18
3	0x00000003	19
4	0x00000004	20
5	0x00000001	21
6	0x00000001	22
7	0x00000004	23
8	0x00000001	24
9	0x00000000	25
10	0x00000000	26
11	0x00000000	27
12	0x00000000	28
13	0x00000000	29
14	0x00000000	30
15	0x00000000	31

Memory Labels

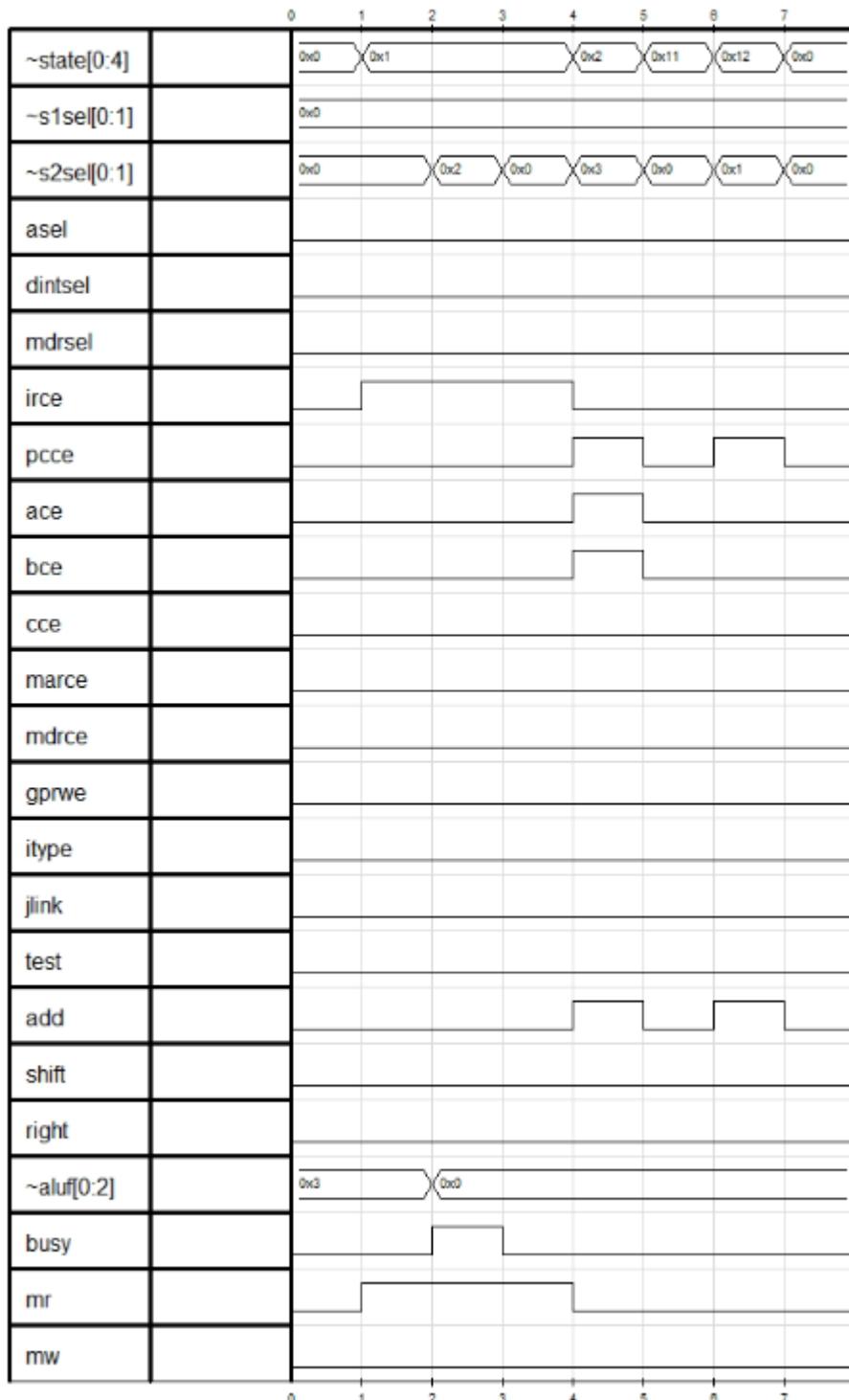
- mem0 = 0x8c01001a
- mem1 = 0xac01001d
- mem2 = 0x2c220001
- mem3 = 0x00221823
- mem4 = 0x005f2802
- mem5 = 0x005f2000

Memory Dump

Address	Values
0x00000000	0x8c01001a 0xac01001d 0x2c220001 0x00221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0x00000008	0xac24001c 0x8c27001c 0x6c880000 0x151f0001 0x0442025 0xc3ff0000 0x2c030012 0x5c7f0000
0x00000010	0x00a11822 0x107f0001 0x5bff0000 0x6803ffff 0x8c02001b 0x74450000 0x78860006 0x00803024
0x00000018	0x00c41025 0xfffff0000 0x00000001 0xffffffff 0x00000000 0x00000004 0xac040006 0xffff7400

Mem Dump From: 0x00000000 Update Dump

Waveforms:



PC = 0xd

Instruction: special-nop

Formula: -

PC = PC + 1 = e

Before step enable:

Slave Labels

```
laram = 0x00000000
d = 0x00000000
STATUS = 0xa2000008
PC = 0x0000000d
```

Commands

Address	Value	Comment
0x00000009	0x8c27001c	lw R7 R1 0x001C
0x0000000a	0x6c880000	sgei R8 R4 0x0000
0x0000000b	0x151f0001	bnez R8 R0x000D
0x0000000c	0x00442025	or R4 R2 R4
0x0000000d	0xc3ff0000	special-nop
0x0000000e	0x2c30012	addi R3 R0 0x0012
0x0000000f	0x5c7f0000	jalr R3
0x00000010	0x00a11822	sub R3 R5 R1
0x00000011	0x107f0001	beqz R3 0x0013

Registers

Register	Value	Comment
0	0x00000000	16
1	0x00000001	17
2	0x00000002	18
3	0x00000003	19
4	0x00000004	20
5	0x00000001	21
6	0x00000001	22
7	0x00000004	23
8	0x00000001	24
9	0x00000000	25
10	0x00000000	26
11	0x00000000	27
12	0x00000000	28
13	0x00000000	29
14	0x00000000	30
15	0x00000000	31

Memory Labels

```
mem0 = 0x8c01001a
mem1 = 0xac01001d
mem2 = 0x2c220001
mem3 = 0x00221823
mem4 = 0x005f2802
mem5 = 0x005f2000
```

Memory Dump

Address	Value
0x00000000	0x8c01001a
0x00000008	0xac24001c
0x00000010	0x00a11822
0x00000018	0x00c41025

Mem Dump From: 0x00000000 Update Dump

After step enable:

Slave Labels

```
laram = 0x00000000
d = 0x00000000
STATUS = 0xa2000006
PC = 0x0000000e
```

Commands

Address	Value	Comment
0x0000000a	0x6c880000	sgei R8 R4 0x0000
0x0000000b	0x151f0001	bnez R8 R0x000D
0x0000000c	0x00442025	or R4 R2 R4
0x0000000d	0xc3ff0000	special-nop
0x0000000e	0x2c30012	addi R3 R0 0x0012
0x0000000f	0x5c7f0000	jalr R3
0x00000010	0x00a11822	sub R3 R5 R1
0x00000011	0x107f0001	beqz R3 0x0013
0x00000012	0x5bffff00	jr R31

Registers

Register	Value	Comment
0	0x00000000	16
1	0x00000001	17
2	0x00000002	18
3	0x00000003	19
4	0x00000004	20
5	0x00000001	21
6	0x00000001	22
7	0x00000004	23
8	0x00000001	24
9	0x00000000	25
10	0x00000000	26
11	0x00000000	27
12	0x00000000	28
13	0x00000000	29
14	0x00000000	30
15	0x00000000	31

Memory Labels

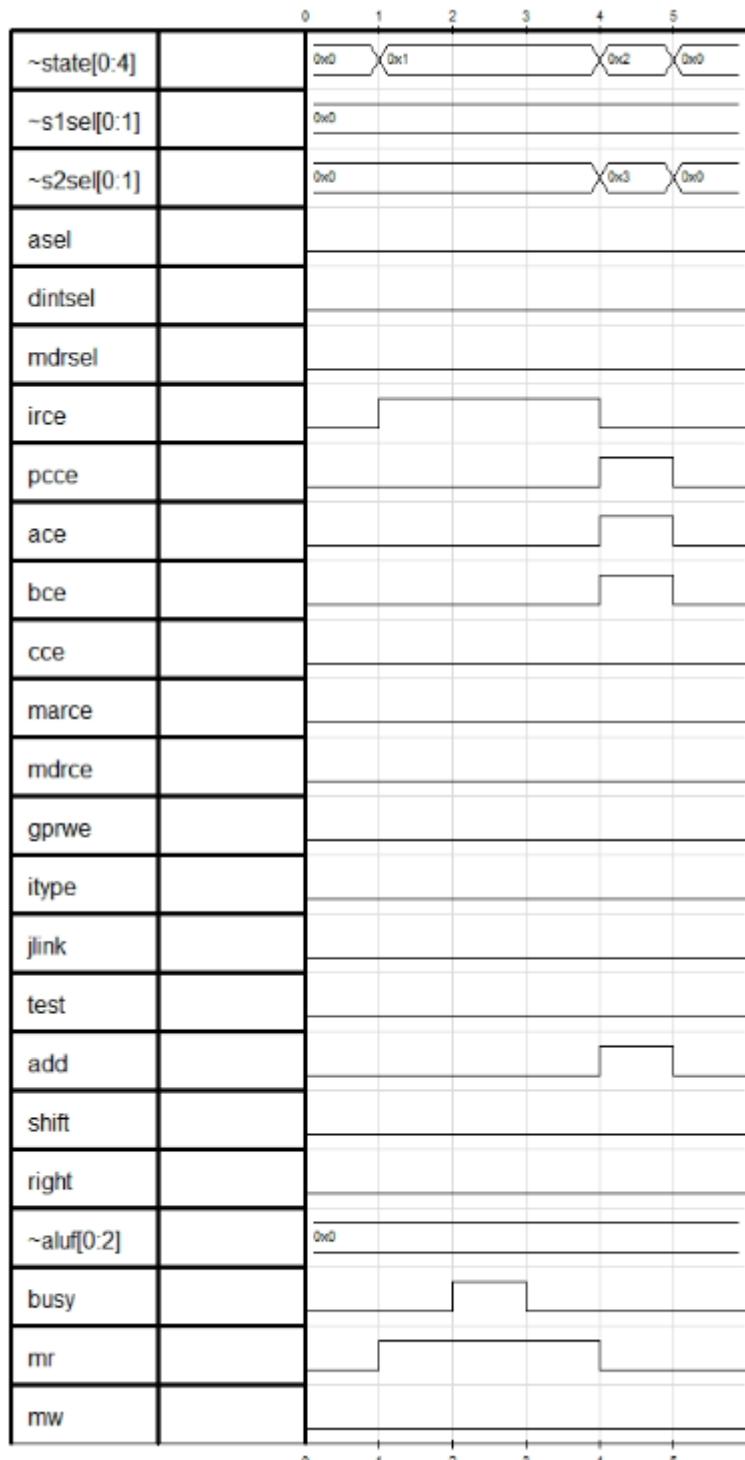
```
mem0 = 0x8c01001a
mem1 = 0xac01001d
mem2 = 0x2c220001
mem3 = 0x00221823
mem4 = 0x005f2802
mem5 = 0x005f2000
```

Memory Dump

Address	Value
0x00000000	0x8c01001a
0x00000008	0xac24001c
0x00000010	0x00a11822
0x00000018	0x00c41025

Mem Dump From: 0x00000000 Update Dump

Waveforms:



PC = 0xf

Instruction: jalr R3

Formula: R31 = PC + 1 = 0x10

PC = R3 = 0x12

Before step enable:

PC = f

R3 = 0x 12

R31 = 0

Registers	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0x00000000	16	0x00000000													
1	0x00000001	17	0x00000000													
2	0x00000002	18	0x00000000													
3	0x00000012	19	0x00000000													
4	0x00000004	20	0x00000000													
5	0x00000001	21	0x00000000													
6	0x00000001	22	0x00000000													
7	0x00000004	23	0x00000000													
8	0x00000001	24	0x00000000													
9	0x00000000	25	0x00000000													
10	0x00000000	26	0x00000000													
11	0x00000000	27	0x00000000													
12	0x00000000	28	0x00000000													
13	0x00000000	29	0x00000000													
14	0x00000000	30	0x00000000													
15	0x00000000	31	0x00000000													

Slave Labels

laram = 0x00000000
d = 0x00000000
STATUS = 0xa2000008
PC = 0x0000000f

Memory Labels

mem0 = 0x8c01001a
mem1 = 0xac01001d
mem2 = 0x2c220001
mem3 = 0x0221823
mem4 = 0x005f2802
mem5 = 0x005f2000

Commands

Step

Reset=1

Refresh Data

Continuous Mode

Signal Waveforms

Write to selected label:

Write

Memory Dump

ADDRESS VALUES

0x00000000 0x8c01001a 0xac01001d 0x2c220001 0x0221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0x00000008 0xac24001c 0x8c27001c 0x6c880000 0x151f0001 0x00442025 0xc3ff0000 0x2c030012 0x5c7f0000
0x00000010 0x00a11822 0x107f0001 0x5bff0000 0x6803ffff 0xb8c02001b 0x74450000 0x78860006 0x00803024
0x00000018 0x00c41025 0xfffff0000 0x00000001 0xefffffff 0x00000000 0x00000004 0xac040006 0xffff7400

Mem Dump From: 0x00000000 Update Dump

After step enable:

PC = 0x12

R31 = 0x10

Registers	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0x00000000	16	0x00000000													
1	0x00000001	17	0x00000000													
2	0x00000002	18	0x00000000													
3	0x00000012	19	0x00000000													
4	0x00000004	20	0x00000000													
5	0x00000001	21	0x00000000													
6	0x00000001	22	0x00000000													
7	0x00000004	23	0x00000000													
8	0x00000001	24	0x00000000													
9	0x00000000	25	0x00000000													
10	0x00000000	26	0x00000000													
11	0x00000000	27	0x00000000													
12	0x00000000	28	0x00000000													
13	0x00000000	29	0x00000000													
14	0x00000000	30	0x00000000													
15	0x00000000	31	0x00000010													

Slave Labels

laram = 0x00000000
d = 0x00000000
STATUS = 0xa2000008
PC = 0x00000012

Memory Labels

mem0 = 0x8c01001a
mem1 = 0xac01001d
mem2 = 0x2c220001
mem3 = 0x0221823
mem4 = 0x005f2802
mem5 = 0x005f2000

Commands

Step

Reset=1

Refresh Data

Continuous Mode

Signal Waveforms

Write to selected label:

Write

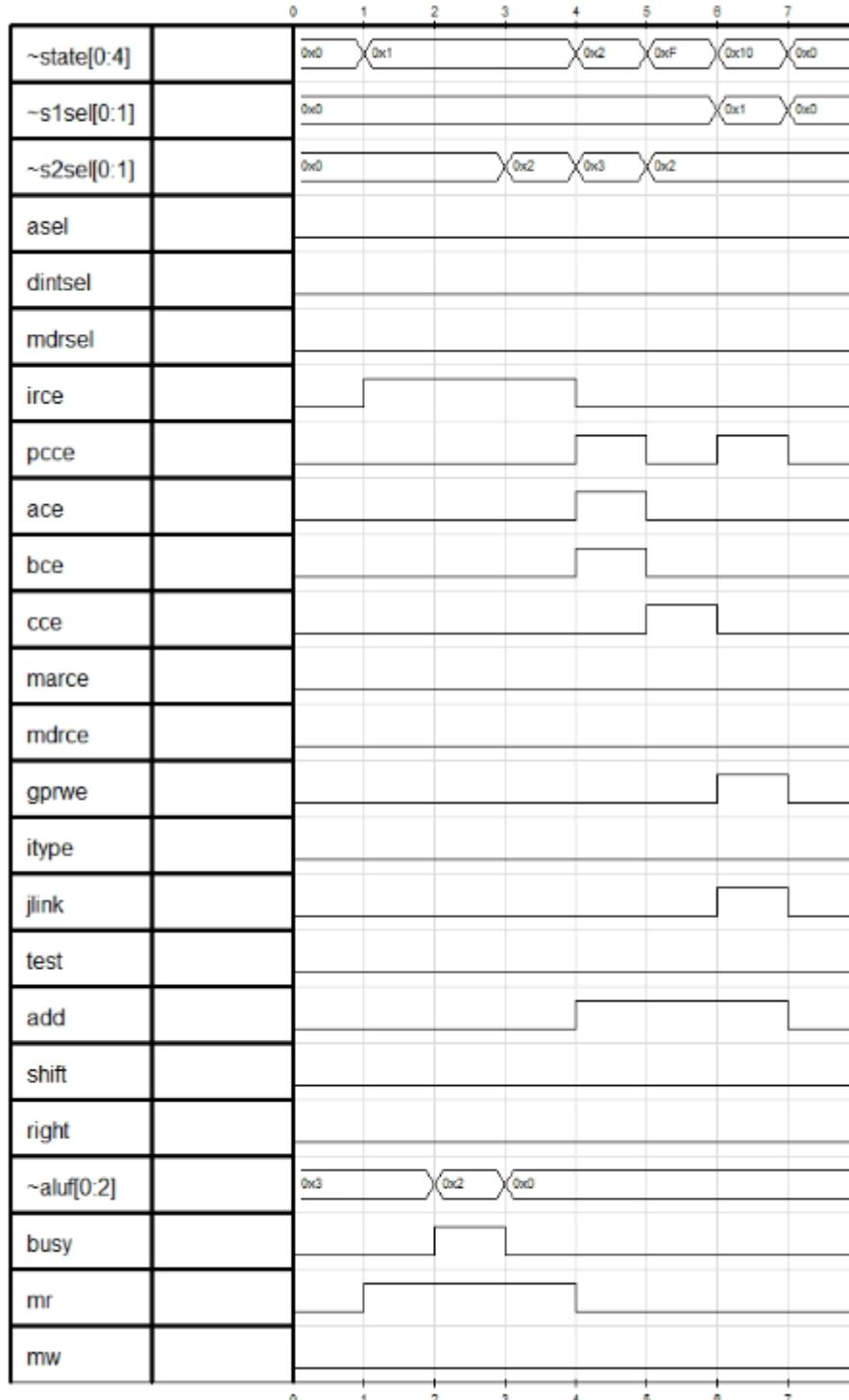
Memory Dump

ADDRESS VALUES

0x00000000 0x8c01001a 0xac01001d 0x2c220001 0x0221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0x00000008 0xac24001c 0x8c27001c 0x6c880000 0x151f0001 0x00442025 0xc3ff0000 0x2c030012 0x5c7f0000
0x00000010 0x00a11822 0x107f0001 0x5bff0000 0x6803ffff 0xb8c02001b 0x74450000 0x78860006 0x00803024
0x00000018 0x00c41025 0xfffff0000 0x00000001 0xefffffff 0x00000000 0x00000004 0xac040006 0xffff7400

Mem Dump From: 0x00000000 Update Dump

Waveforms:



PC = 0x12

Instruction: jr R31

Formula: -

PC = R31 = 0x10

Before step enable:

The screenshot shows a debugger interface with the following details:

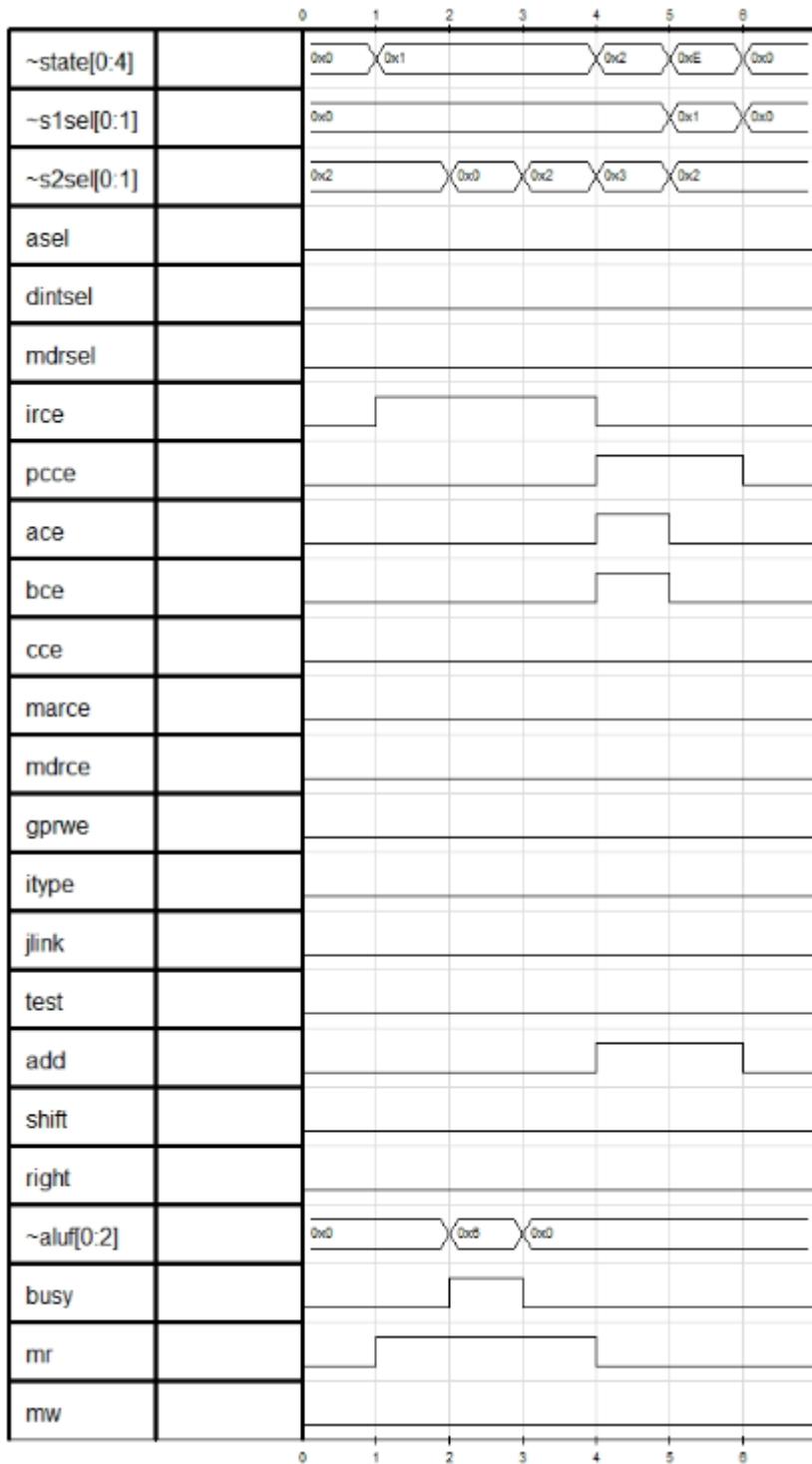
- Registers:** Shows 16 registers from 0 to 15. Register 15 contains 0x00000010.
- Commands:** A list of assembly instructions:
 - 0x000000e 0x2c030012 addi R3 R0 0x0012
 - 0x000000f 0x5c7f0000 jalr R3
 - 0x0000010 0x00a11822 sub R3 R5 R1
 - 0x0000011 0x107f0001 beqz R3 0x0013
 - 0x0000012 0x5bff0000 jr R31** (highlighted)
 - 0x0000013 0x6803ffff seqj R3 R0 0xFFFF
 - 0x0000014 0x8c02001b lw R2 R0 0x001B
 - 0x0000015 0x74450000 srl R5 R2 R0x0000
 - 0x0000016 0x78860006 slei R6 R4 0x0006
- Memory Dump:** A table showing memory values at addresses 0x00000000, 0x00000008, 0x00000010, and 0x00000018.
- Slave Labels:** Shows laram = 0x00000000, d = 0x00000000, STATUS = 0xa2000008, and PC = 0x00000012.
- Memory Labels:** Shows mem0 = 0xb01001a, mem1 = 0xac01001d, mem2 = 0xc220001, mem3 = 0x00221823, mem4 = 0x005f2802, and mem5 = 0x005f2000.
- Buttons:** Step, Reset=1, Refresh Data, Continuous Mode, Signal Waveforms, Write, and Mem Dump From: 0x00000000, Update Dump.

After step enable:

The screenshot shows the debugger interface after the step operation has been executed. The state has changed as follows:

- Registers:** Register 15 now contains 0x00000010.
- Commands:** The instruction at address 0x0000012 has been executed, and the next instruction at 0x0000013 is highlighted.
- Memory Dump:** The dump table remains the same as in the previous state.
- Slave Labels:** The PC value has been updated to 0x00000010.
- Memory Labels:** The PC value has been updated to 0x00000010.
- Buttons:** Step, Reset=1, Refresh Data, Continuous Mode, Signal Waveforms, Write, and Mem Dump From: 0x00000000, Update Dump.

Waveforms:



PC = 0x10

Instruction: sub R3 R5 R1

Formula: R3 = R5 - R1 = 0

PC = PC + 1 = 0x11

Before step enable:

PC = 0x10

R5 = 0x1

R1 = 0x1

R3 = 0x12

Registers	Value	Index	Label
0	0x00000000	16	0x00000000
1	0x00000001	17	0x00000000
2	0x00000002	18	0x00000000
3	0x00000012	19	0x00000000
4	0x00000004	20	0x00000000
5	0x00000001	21	0x00000000
6	0x00000001	22	0x00000000
7	0x00000004	23	0x00000000
8	0x00000001	24	0x00000000
9	0x00000000	25	0x00000000
10	0x00000000	26	0x00000000
11	0x00000000	27	0x00000000
12	0x00000000	28	0x00000000
13	0x00000000	29	0x00000000
14	0x00000000	30	0x00000000
15	0x00000000	31	0x00000010

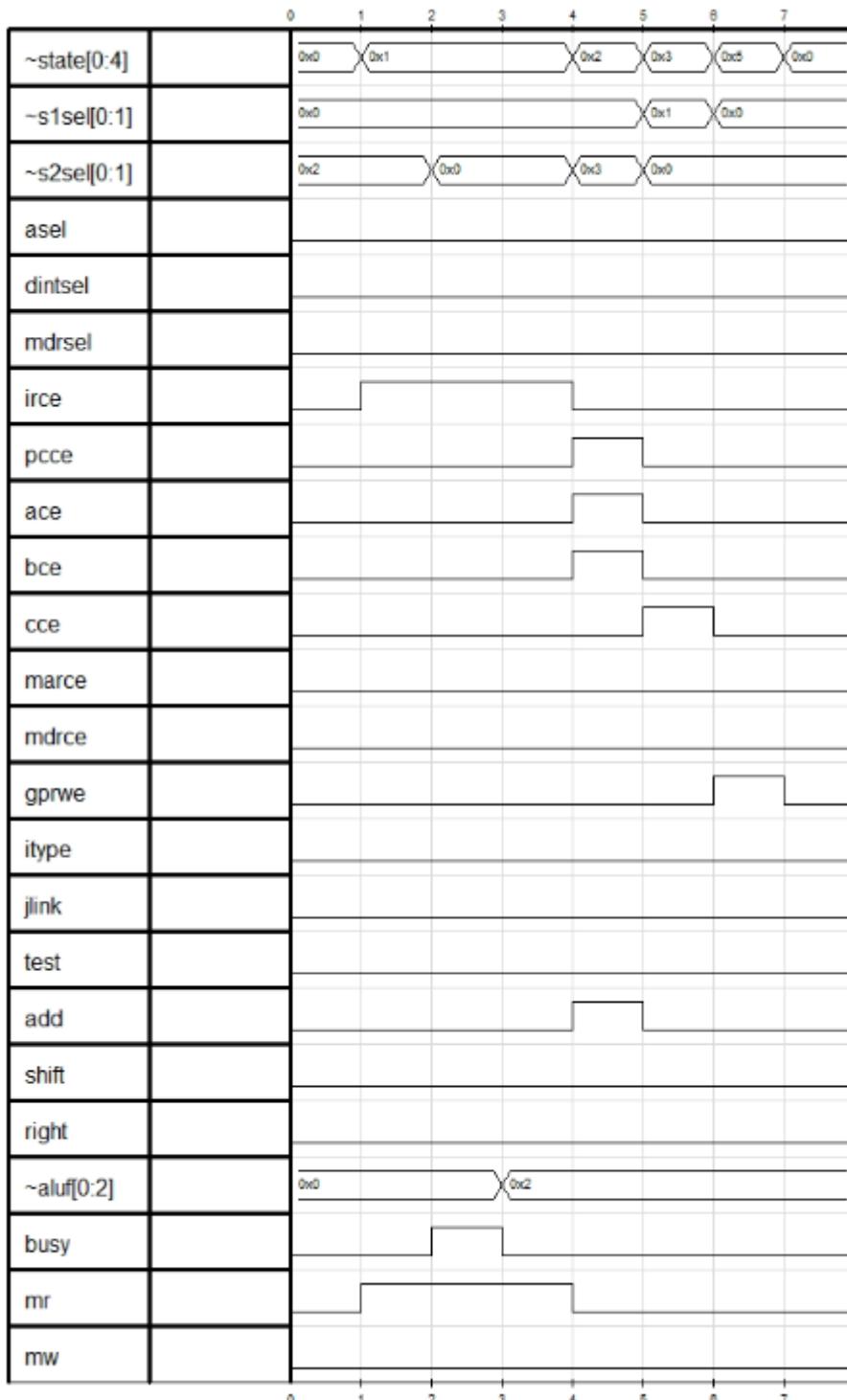
After step enable:

PC = 0x11

R3 = 0

Registers	Value	Index	Label
0	0x00000000	16	0x00000000
1	0x00000001	17	0x00000000
2	0x00000002	18	0x00000000
3	0x00000000	19	0x00000000
4	0x00000004	20	0x00000000
5	0x00000001	21	0x00000000
6	0x00000001	22	0x00000000
7	0x00000004	23	0x00000000
8	0x00000001	24	0x00000000
9	0x00000000	25	0x00000000
10	0x00000000	26	0x00000000
11	0x00000000	27	0x00000000
12	0x00000000	28	0x00000000
13	0x00000000	29	0x00000000
14	0x00000000	30	0x00000000
15	0x00000000	31	0x00000010

Waveforms:



PC = 0x11

Instruction: beqz R3 0x0013

Formula: PC = (!R3) ? 0x0013 : (PC + 1)

PC = 0x0013

Before step enable:

PC = 0x11

R3 = 0

Registers	Value	Registers	Value
0	0x00000000	16	0x00000000
1	0x00000001	17	0x00000000
2	0x00000002	18	0x00000000
3	0x00000000	19	0x00000000
4	0x00000004	20	0x00000000
5	0x00000001	21	0x00000000
6	0x00000001	22	0x00000000
7	0x00000004	23	0x00000000
8	0x00000001	24	0x00000000
9	0x00000000	25	0x00000000
10	0x00000000	26	0x00000000
11	0x00000000	27	0x00000000
12	0x00000000	28	0x00000000
13	0x00000000	29	0x00000000
14	0x00000000	30	0x00000000
15	0x00000000	31	0x00000000

Slave Labels

```
laram = 0x00000100
d = 0x00000000
STATUS = 0xa2000008
PC = 0x00000011
```

Memory Labels

```
mem0 = 0xb0c01001a
mem1 = 0xac01001d
mem2 = 0x2c220001
mem3 = 0x00221823
mem4 = 0x005f2802
mem5 = 0x005f2000
```

Commands

Step

Reset=1

Refresh Data

Continuous Mode

Signal Waveforms

Write to selected label:

Write

Registers

Memory Dump

ADDRESS	VALUES
0x00000000	0x8c01001a 0xac01001d 0x2c220001 0x00221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0x00000008	0xac24001c 0x8c27001c 0x6c880000 0x151f0001 0x0442025 0xc3ff0000 0x2c030012 0x5c7f0000
0x00000010	0x00a11822 0x107f0001 0x5bff0000 0x6803ffff 0xb0c02001b 0x74450000 0x78860006 0x00803024
0x00000018	0x00c41025 0xfffff0000 0x00000001 0xffffffff 0x00000000 0x00000004 0xac040006 0xfffff7400

Mem Dump From: 0x00000000 Update Dump

After step enable:

PC = 0x13

R3 = 1

Registers	Value	Registers	Value
0	0x00000000	16	0x00000000
1	0x00000001	17	0x00000000
2	0x00000002	18	0x00000000
3	0x00000000	19	0x00000000
4	0x00000004	20	0x00000000
5	0x00000001	21	0x00000000
6	0x00000001	22	0x00000000
7	0x00000004	23	0x00000000
8	0x00000001	24	0x00000000
9	0x00000000	25	0x00000000
10	0x00000000	26	0x00000000
11	0x00000000	27	0x00000000
12	0x00000000	28	0x00000000
13	0x00000000	29	0x00000000
14	0x00000000	30	0x00000000
15	0x00000000	31	0x00000010

Slave Labels

```
laram = 0x08000000
d = 0x00000000
STATUS = 0xa2000008
PC = 0x00000013
```

Memory Labels

```
mem0 = 0xb0c01001a
mem1 = 0xac01001d
mem2 = 0x2c220001
mem3 = 0x00221823
mem4 = 0x005f2802
mem5 = 0x005f2000
```

Commands

Step

Reset=1

Refresh Data

Continuous Mode

Signal Waveforms

Write to selected label:

Write

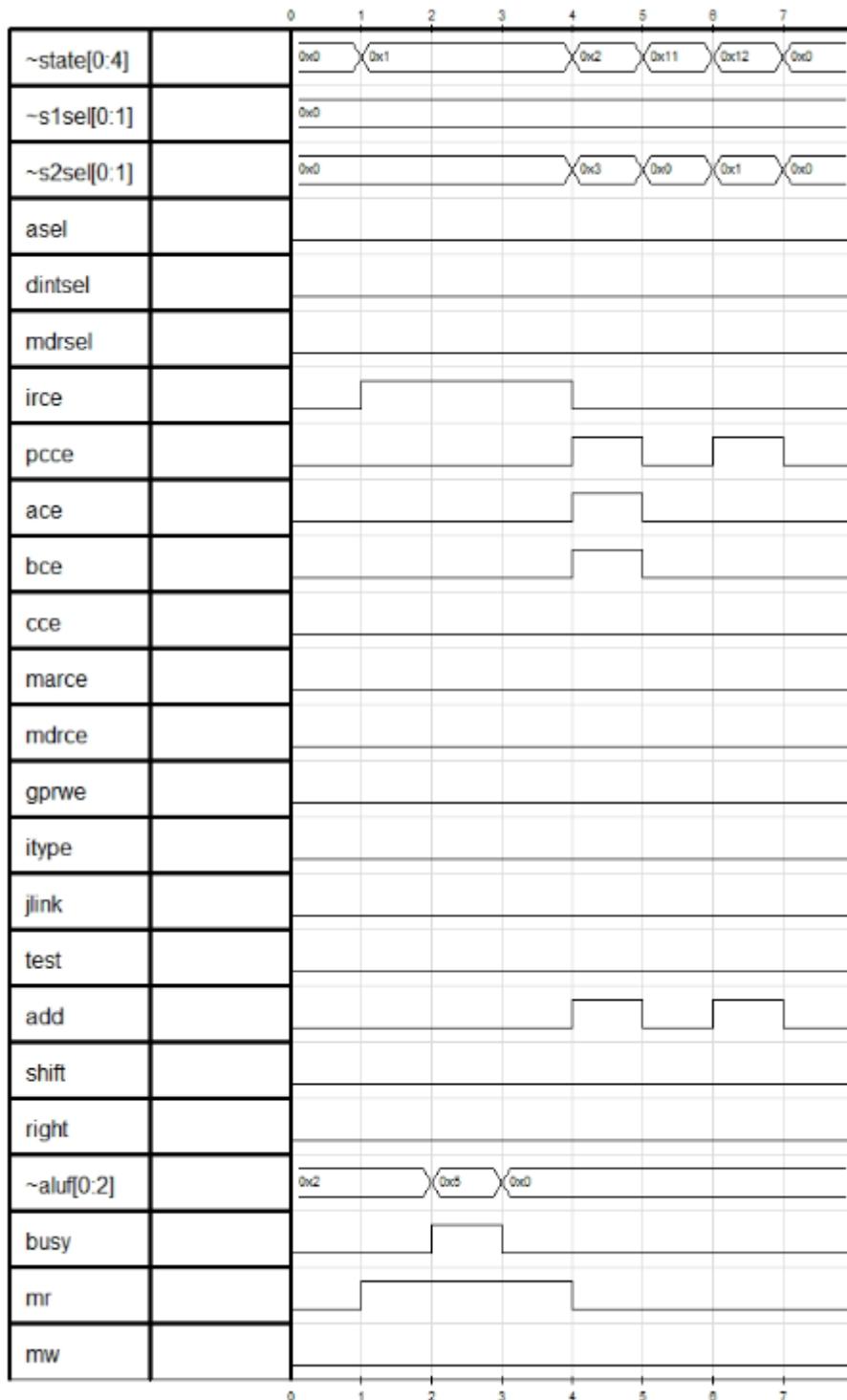
Registers

Memory Dump

ADDRESS	VALUES
0x00000000	0x8c01001a 0xac01001d 0x2c220001 0x00221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0x00000008	0xac24001c 0x8c27001c 0x6c880000 0x151f0001 0x0442025 0xc3ff0000 0x2c030012 0x5c7f0000
0x00000010	0x00a11822 0x107f0001 0x5bff0000 0x6803ffff 0xb0c02001b 0x74450000 0x78860006 0x00803024
0x00000018	0x00c41025 0xfffff0000 0x00000001 0xffffffff 0x00000000 0x00000004 0xac040006 0xfffff7400

Mem Dump From: 0x00000000 Update Dump

Waveforms:



PC = 0x13

Instruction: seqi R3 R0 0xFFFF

Formula: R3 = (R0 == 0xFFFF) ? 1 : 0 => R3 = 0

PC = PC + 1 = 0x14

Before step enable:

PC = 0x13

R3 = 0

Registers	Value	Register
0	0x00000000	16
1	0x00000001	17
2	0x00000002	18
3	0x00000000	19
4	0x00000004	20
5	0x00000001	21
6	0x00000001	22
7	0x00000004	23
8	0x00000001	24
9	0x00000000	25
10	0x00000000	26
11	0x00000000	27
12	0x00000000	28
13	0x00000000	29
14	0x00000000	30
15	0x00000000	31

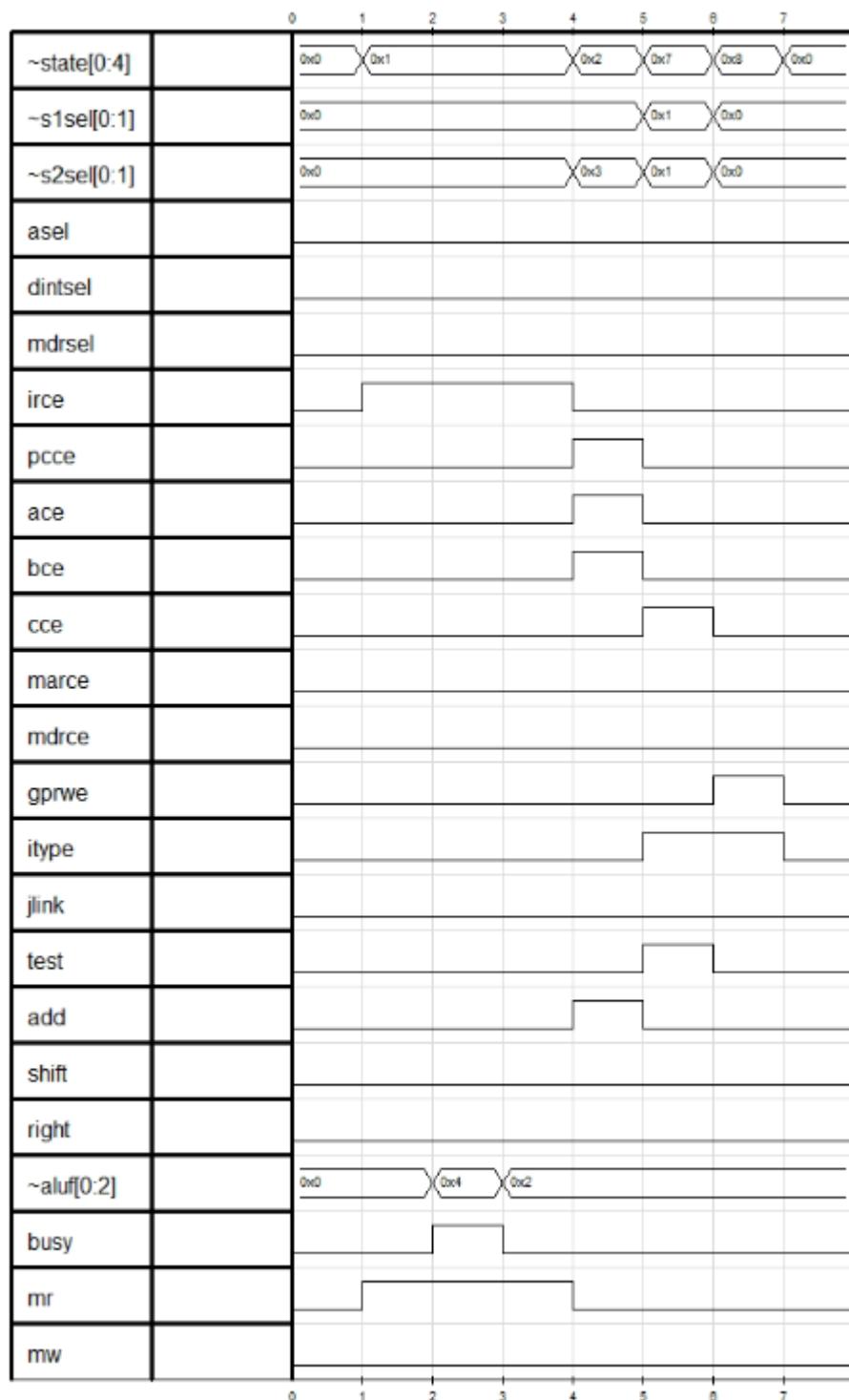
After step enable:

PC = 0x14

R3 = 0

Registers	Value	Register
0	0x00000000	16
1	0x00000001	17
2	0x00000002	18
3	0x00000000	19
4	0x00000004	20
5	0x00000001	21
6	0x00000001	22
7	0x00000004	23
8	0x00000001	24
9	0x00000000	25
10	0x00000000	26
11	0x00000000	27
12	0x00000000	28
13	0x00000000	29
14	0x00000000	30
15	0x00000000	31

Waveforms:



PC = 0x15

Instruction: snei R5 R2 0x0000

Formula: R5 = (R2 != 0x0000) ? 1 : 0 => R5 = 1

PC = PC + 1 = 0x16

Before step enable:

The screenshot shows the debugger interface with the following details:

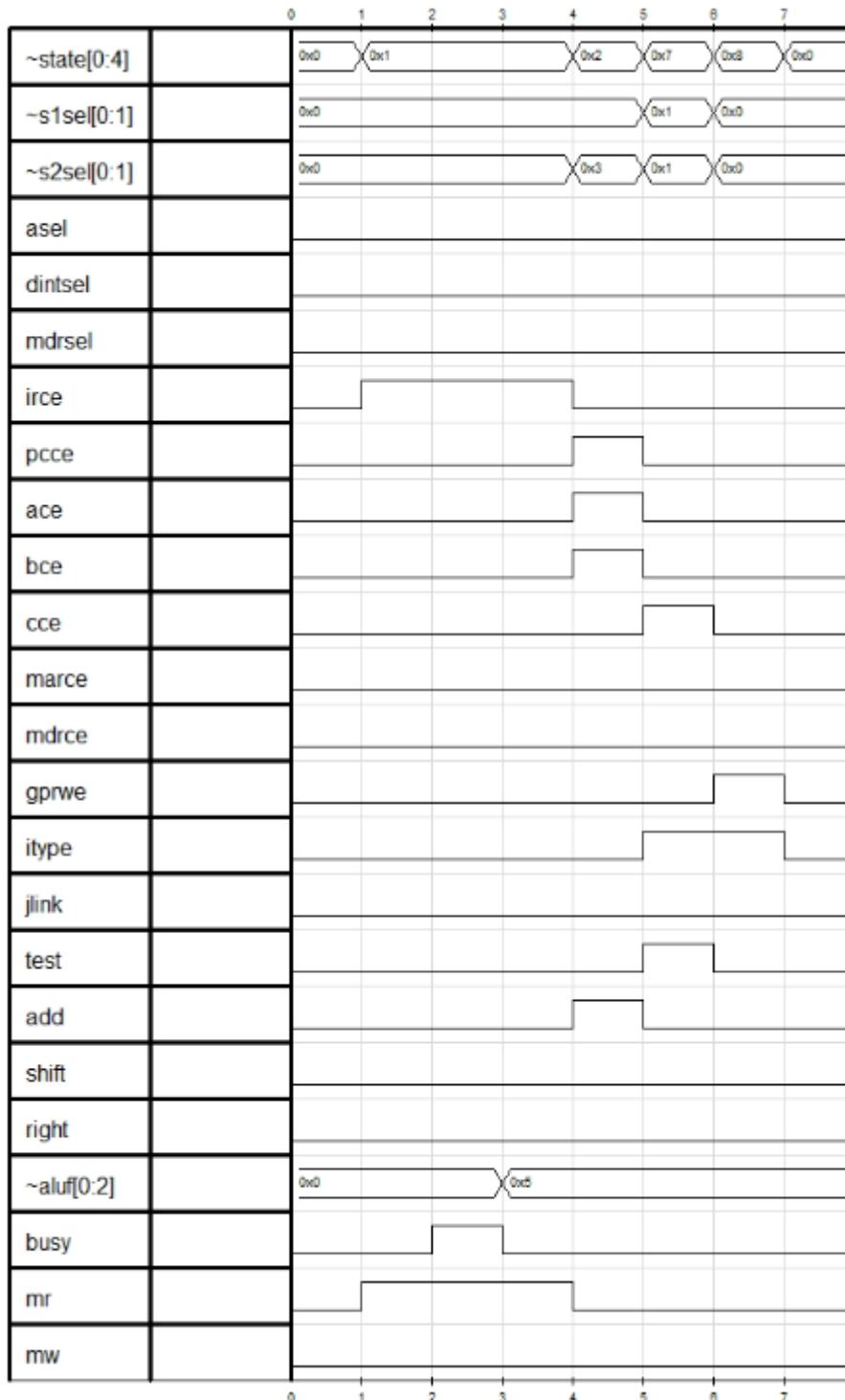
- Registers:** R5 is set to 1, and R2 is set to 0xffffffff.
- Commands:** The assembly code includes instructions like beqz R3 0x0013, jr R31, seq R3 R0 0xFFFF, lw R2 R0 0x001B, snei R5 R2 0x0000, slei R6 R4 0x0006, xor R6 R4 R0, or R2 R6 R4, and halt.
- Memory Dump:** A table showing memory values at addresses 0x00000000, 0x00000008, 0x00000010, and 0x00000018.
- Slave Labels:** laram = 0x08000000, d = 0x00000000, STATUS = 0xa200000c, PC = 0x00000015.
- Memory Labels:** mem0 = 0xbfc01001a, mem1 = 0xac01001d, mem2 = 0x2c220001, mem3 = 0x00221823, mem4 = 0x005f2802, mem5 = 0x005f2000.

After step enable:

The screenshot shows the debugger interface after the instruction has been executed:

- Registers:** R5 is still set to 1, and R2 is now set to 0x00000001.
- Commands:** The assembly code remains the same, but the result of the snei instruction is reflected in the register values.
- Memory Dump:** A table showing memory values at addresses 0x00000000, 0x00000008, 0x00000010, and 0x00000018.
- Slave Labels:** laram = 0x00000000, d = 0x00000000, STATUS = 0xa2000008, PC = 0x00000016.
- Memory Labels:** mem0 = 0xbfc01001a, mem1 = 0xac01001d, mem2 = 0x2c220001, mem3 = 0x00221823, mem4 = 0x005f2802, mem5 = 0x005f2000.

Waveforms:



PC = 0x16

Instruction: slei R6 R4 0x0006

Formula: R6 = (R4 <= 0x0006) ? 1 : 0 => R6 = 1

PC = PC + 1 = 0x17

Before step enable:

R4 = 4 R6 = 1

Slave Labels

laram = 0x00000000
d = 0x00000000
STATUS = 0xa2000008
PC = 0x00000016

Commands

Step Reset=1 Refresh Data Continuous Mode Signal Waveforms

Write to selected label:

Registers

0	0x00000000	16	0x00000000
1	0x00000001	17	0x00000000
2	0xffffffff	18	0x00000000
3	0x00000000	19	0x00000000
4	0x00000004	20	0x00000000
5	0x00000001	21	0x00000000
6	0x00000001	22	0x00000000
7	0x00000004	23	0x00000000
8	0x00000001	24	0x00000000
9	0x00000000	25	0x00000000
10	0x00000000	26	0x00000000
11	0x00000000	27	0x00000000
12	0x00000000	28	0x00000000
13	0x00000000	29	0x00000000
14	0x00000000	30	0x00000000
15	0x00000000	31	0x00000010

Memory Labels

mem0 = 0x8c01001a
mem1 = 0xac01001d
mem2 = 0x2c220001
mem3 = 0x00221823
mem4 = 0x005f2802
mem5 = 0x005f2000

Memory Dump

ADDRESS	VALUES
0x00000000	0x8c01001a 0xac01001d 0x2c220001 0x00221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0x00000008	0xac24001c 0x8c27001c 0x6c880000 0x151f0001 0x00442025 0xc3ff0000 0x2c030012 0x5c7f0000
0x00000010	0x00a11822 0x107f0001 0x5bf0000 0x6803ffff 0x8c02001b 0x74450000 0x78860006 0x00803024
0x00000018	0x00c41025 0xffff0000 0x00000001 0xefffffff 0x00000000 0x00000004 0xac040006 0xffff7400

Mem Dump From: 0x00000000 Update Dump

After step enable:

R6 = 1

Slave Labels

laram = 0x14000000
d = 0x00000000
STATUS = 0xa2000008
PC = 0x00000017

Commands

Step Reset=1 Refresh Data Continuous Mode Signal Waveforms

Write to selected label:

Registers

0	0x00000000	16	0x00000000
1	0x00000001	17	0x00000000
2	0xffffffff	18	0x00000000
3	0x00000000	19	0x00000000
4	0x00000004	20	0x00000000
5	0x00000001	21	0x00000000
6	0x00000001	22	0x00000000
7	0x00000004	23	0x00000000
8	0x00000001	24	0x00000000
9	0x00000000	25	0x00000000
10	0x00000000	26	0x00000000
11	0x00000000	27	0x00000000
12	0x00000000	28	0x00000000
13	0x00000000	29	0x00000000
14	0x00000000	30	0x00000000
15	0x00000000	31	0x00000010

Memory Labels

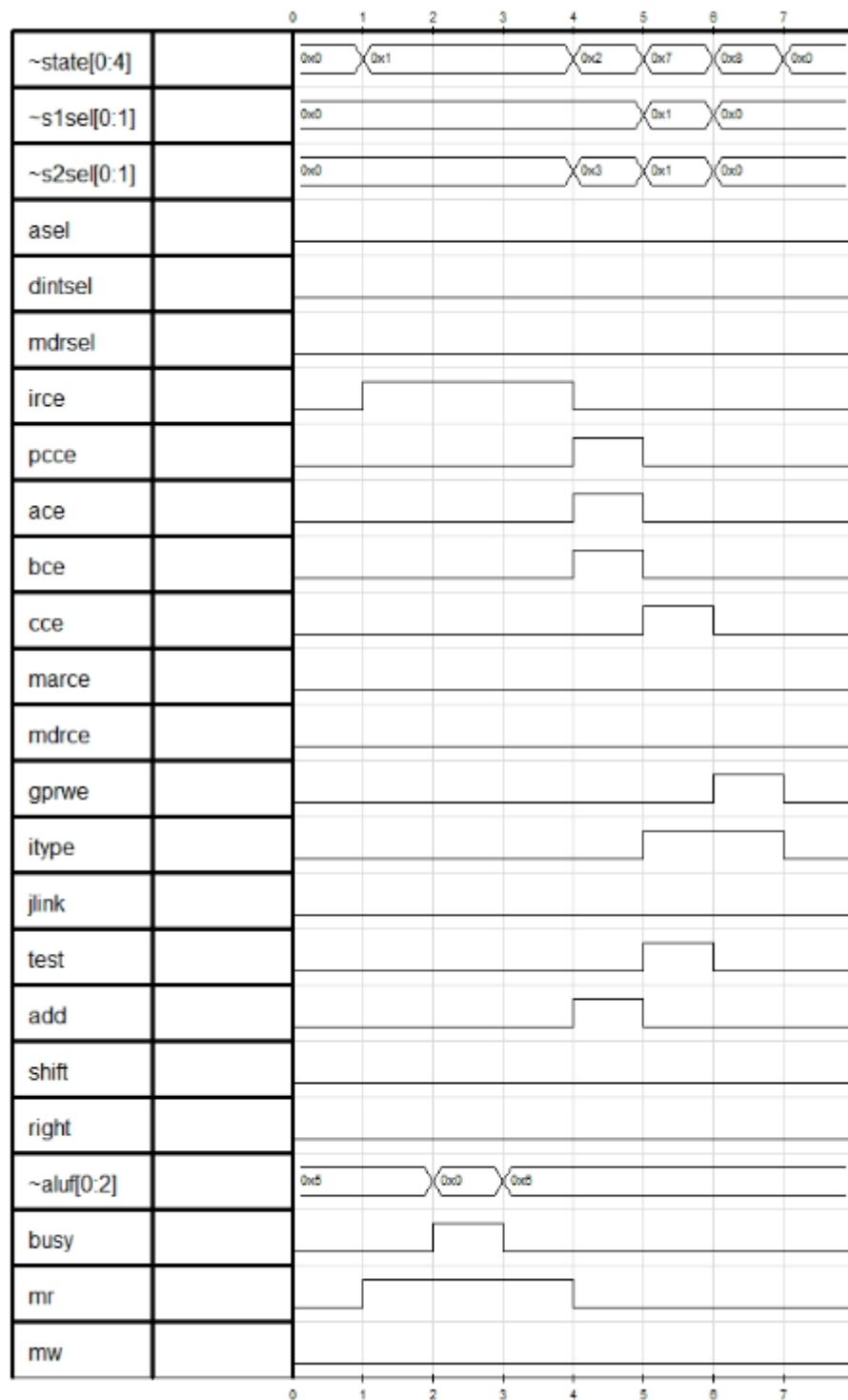
mem0 = 0x8c01001a
mem1 = 0xac01001d
mem2 = 0x2c220001
mem3 = 0x00221823
mem4 = 0x005f2802
mem5 = 0x005f2000

Memory Dump

ADDRESS	VALUES
0x00000000	0x8c01001a 0xac01001d 0x2c220001 0x00221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0x00000008	0xac24001c 0x8c27001c 0x6c880000 0x151f0001 0x00442025 0xc3ff0000 0x2c030012 0x5c7f0000
0x00000010	0x00a11822 0x107f0001 0x5bf0000 0x6803ffff 0x8c02001b 0x74450000 0x78860006 0x00803024
0x00000018	0x00c41025 0xffff0000 0x00000001 0xefffffff 0x00000000 0x00000004 0xac040006 0xffff7400

Mem Dump From: 0x00000000 Update Dump

Waveforms:



PC = 0x17

Instruction: xor R6 R4 R0

Formula: R6 = R4 xor R0 = R4 = 4

PC = PC + 1 = 0x18

Before step enable:

Registers

0	0x00000000	16	0x00000000
1	0x00000001	17	0x00000000
2	0xffffffff	18	0x00000000
3	0x00000000	19	0x00000000
4	0x00000004	20	0x00000000
5	0x00000001	21	0x00000000
6	0x00000001	22	0x00000000
7	0x00000004	23	0x00000000
8	0x00000001	24	0x00000000
9	0x00000000	25	0x00000000
10	0x00000000	26	0x00000000
11	0x00000000	27	0x00000000
12	0x00000000	28	0x00000000
13	0x00000000	29	0x00000000
14	0x00000000	30	0x00000000
15	0x00000000	31	0x00000010

Commands

0x00000013	0x6803ffff	seqj R3 R0 0xFFFF
0x00000014	0x8c02001b	lw R2 R0 0x001B
0x00000015	0x74450000	snei R5 R2 0x0000
0x00000016	0x78860006	slel R6 R4 0x0006
0x00000017	0x00803024	xor R6 R4 R0
0x00000018	0x00c41025	or R2 R6 R4
0x00000019	0xffff0000	halt
0x0000001a	0x00000001	no disassembly
0x0000001b	0xefffffff	no disassembly

Slave Labels

laram = 0x14000000
d = 0x00000000
STATUS = 0xa2000008
PC = 0x00000017

Memory Labels

mem0 = 0xb01001a
mem1 = 0xac01001d
mem2 = 0xc220001
mem3 = 0x00221823
mem4 = 0x005f2802
mem5 = 0x005f2000

Step

Reset=1

Refresh Data

Continuous Mode

Signal Waveforms

Write to selected label:

Write

Memory Dump

ADDRESS	VALUES
0x00000000	0xac01001d 0x2c220001 0x00221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0x00000008	0xac24001c 0x8c27001c 0x6c880000 0x151f0001 0x00442025 0xc3ff0000 0x2c030012 0x5cf0000
0x00000010	0x00a11822 0x107f0001 0x5bff0000 0x6803ffff 0x8c02001b 0x74450000 0x78860006 0x00803024
0x00000018	0x00c41025 0xffff0000 0x00000001 0xffffffff 0x00000000 0x00000004 0xac040006 0xffff7400

Mem Dump From: 0x00000000 Update Dump

After step enable:

Registers

0	0x00000000	16	0x00000000
1	0x00000001	17	0x00000000
2	0xffffffff	18	0x00000000
3	0x00000000	19	0x00000000
4	0x00000004	20	0x00000000
5	0x00000001	21	0x00000000
6	0x00000004	22	0x00000000
7	0x00000004	23	0x00000000
8	0x00000001	24	0x00000000
9	0x00000000	25	0x00000000
10	0x00000000	26	0x00000000
11	0x00000000	27	0x00000000
12	0x00000000	28	0x00000000
13	0x00000000	29	0x00000000
14	0x00000000	30	0x00000000
15	0x00000000	31	0x00000010

Commands

0x00000014	0x8c02001b	lw R2 R0 0x001B
0x00000015	0x74450000	snei R5 R2 0x0000
0x00000016	0x78860006	slel R6 R4 0x0006
0x00000017	0x00803024	xor R6 R4 R0
0x00000018	0x00c41025	or R2 R6 R4
0x00000019	0xffff0000	halt
0x0000001a	0x00000001	no disassembly
0x0000001b	0xffffffff	no disassembly
0x0000001c	0x00000000	slli R0 R0

Slave Labels

laram = 0x18000000
d = 0x00000000
STATUS = 0xa2000008
PC = 0x00000018

Memory Labels

mem0 = 0xb01001a
mem1 = 0xac01001d
mem2 = 0xc220001
mem3 = 0x00221823
mem4 = 0x005f2802
mem5 = 0x005f2000

Step

Reset=1

Refresh Data

Continuous Mode

Signal Waveforms

Write to selected label:

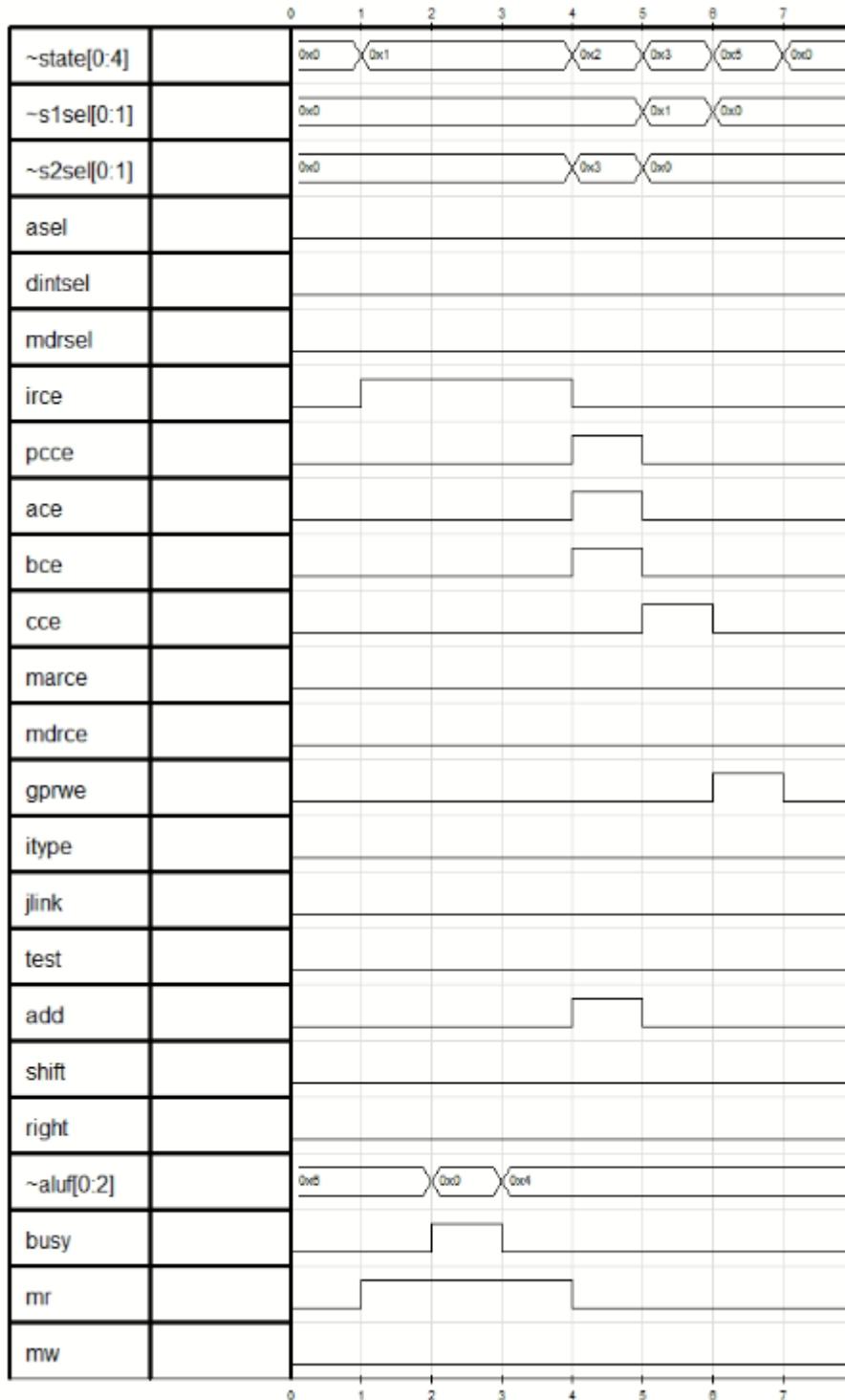
Write

Memory Dump

ADDRESS	VALUES
0x00000000	0xac01001d 0x2c220001 0x00221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0x00000008	0xac24001c 0x8c27001c 0x6c880000 0x151f0001 0x00442025 0xc3ff0000 0x2c030012 0x5cf0000
0x00000010	0x00a11822 0x107f0001 0x5bff0000 0x6803ffff 0x8c02001b 0x74450000 0x78860006 0x00803024
0x00000018	0x00c41025 0xffff0000 0x00000001 0xffffffff 0x00000000 0x00000004 0xac040006 0xffff7400

Mem Dump From: 0x00000000 Update Dump

Waveforms:



PC = 0x18

Instruction: or R2 R6 R4

Formula: R2 = R6 or R4 = 4

PC = PC + 1 = 0x19

Before step enable:

PC = 0x18

Slave Labels		Commands		Registers		
laram = 0x18000000	d = 0x00000000	0x00000014	0x8c02001b	lw R2 R0 0x001B	0	0x00000000 16 0x00000000
STATUS = 0xa2000008	PC = 0x00000018	0x00000015	0x74450000	snei R5 R2 0x0000	1	0x00000001 17 0x00000000
		0x00000016	0x78860006	slei R6 R4 0x0006	2	0xffffffff 18 0x00000000
		0x00000017	0x00803024	xor R6 R4 R0	3	0x00000000 19 0x00000000
		0x00000018	0x00c41025	or R2 R6 R4	4	0x00000004 20 0x00000000
		0x00000019	0xfffff000	halt	5	0x00000001 21 0x00000000
		0x0000001a	0x00000001	no disassembly	6	0x00000004 22 0x00000000
		0x0000001b	0xffffffff	no disassembly	7	0x00000004 23 0x00000000
		0x0000001c	0x00000000	sli R0 R0	8	0x00000001 24 0x00000000
					9	0x00000000 25 0x00000000
					10	0x00000000 26 0x00000000
					11	0x00000000 27 0x00000000
					12	0x00000000 28 0x00000000
					13	0x00000000 29 0x00000000
					14	0x00000000 30 0x00000000
					15	0x00000000 31 0x00000010

Memory Labels

ADDRESS	VALUES
0x00000000	0x8c01001a 0xac01001d 0x2c220001 0x00221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0x00000008	0xac24001c 0x8c27001c 0x6c880000 0x151f0001 0x00442025 0xc3ff0000 0x2c030012 0x5c7f0000
0x00000010	0x00a11822 0x107f0001 0xbffff000 0x6803ffff 0x8c02001b 0x74450000 0x78860006 0x0803024
0x00000018	0x00c41025 0xfffff000 0x00000001 0xffffffff 0x00000000 0x00000004 0xac040006 0xffff7400

Write to selected label: Write

Memory Dump

ADDRESS	VALUES
0x00000000	0x8c01001a 0xac01001d 0x2c220001 0x00221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0x00000008	0xac24001c 0x8c27001c 0x6c880000 0x151f0001 0x00442025 0xc3ff0000 0x2c030012 0x5c7f0000
0x00000010	0x00a11822 0x107f0001 0xbffff000 0x6803ffff 0x8c02001b 0x74450000 0x78860006 0x0803024
0x00000018	0x00c41025 0xfffff000 0x00000001 0xffffffff 0x00000000 0x00000004 0xac040006 0xffff7400

Mem Dump From: 0x00000000 Update Dump

R2 = 0xffffffff

R4 = 4

R6 = 4

After step enable:

PC = 0x19

Slave Labels		Commands		Registers		
laram = 0x10000000	d = 0x00000000	0x00000015	0x74450000	snei R5 R2 0x0000	0	0x00000000 16 0x00000000
STATUS = 0xa2000008	PC = 0x00000019	0x00000016	0x78860006	slei R6 R4 0x0006	1	0x00000001 17 0x00000000
		0x00000017	0x00803024	xor R6 R4 R0	2	0x00000004 18 0x00000000
		0x00000018	0x00c41025	or R2 R6 R4	3	0x00000000 19 0x00000000
		0x00000019	0xfffff000	halt	4	0x00000004 20 0x00000000
		0x0000001a	0x00000001	no disassembly	5	0x00000001 21 0x00000000
		0x0000001b	0xffffffff	no disassembly	6	0x00000004 22 0x00000000
		0x0000001c	0x00000000	sli R0 R0	7	0x00000004 23 0x00000000
		0x0000001d	0x00000004	no disassembly	8	0x00000001 24 0x00000000
					9	0x00000000 25 0x00000000
					10	0x00000000 26 0x00000000
					11	0x00000000 27 0x00000000
					12	0x00000000 28 0x00000000
					13	0x00000000 29 0x00000000
					14	0x00000000 30 0x00000000
					15	0x00000000 31 0x00000010

Memory Labels

ADDRESS	VALUES
0x00000000	0x8c01001a 0xac01001d 0x2c220001 0x00221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0x00000008	0xac24001c 0x8c27001c 0x6c880000 0x151f0001 0x00442025 0xc3ff0000 0x2c030012 0x5c7f0000
0x00000010	0x00a11822 0x107f0001 0xbffff000 0x6803ffff 0x8c02001b 0x74450000 0x78860006 0x0803024
0x00000018	0x00c41025 0xfffff000 0x00000001 0xffffffff 0x00000000 0x00000004 0xac040006 0xffff7400

Write to selected label: Write

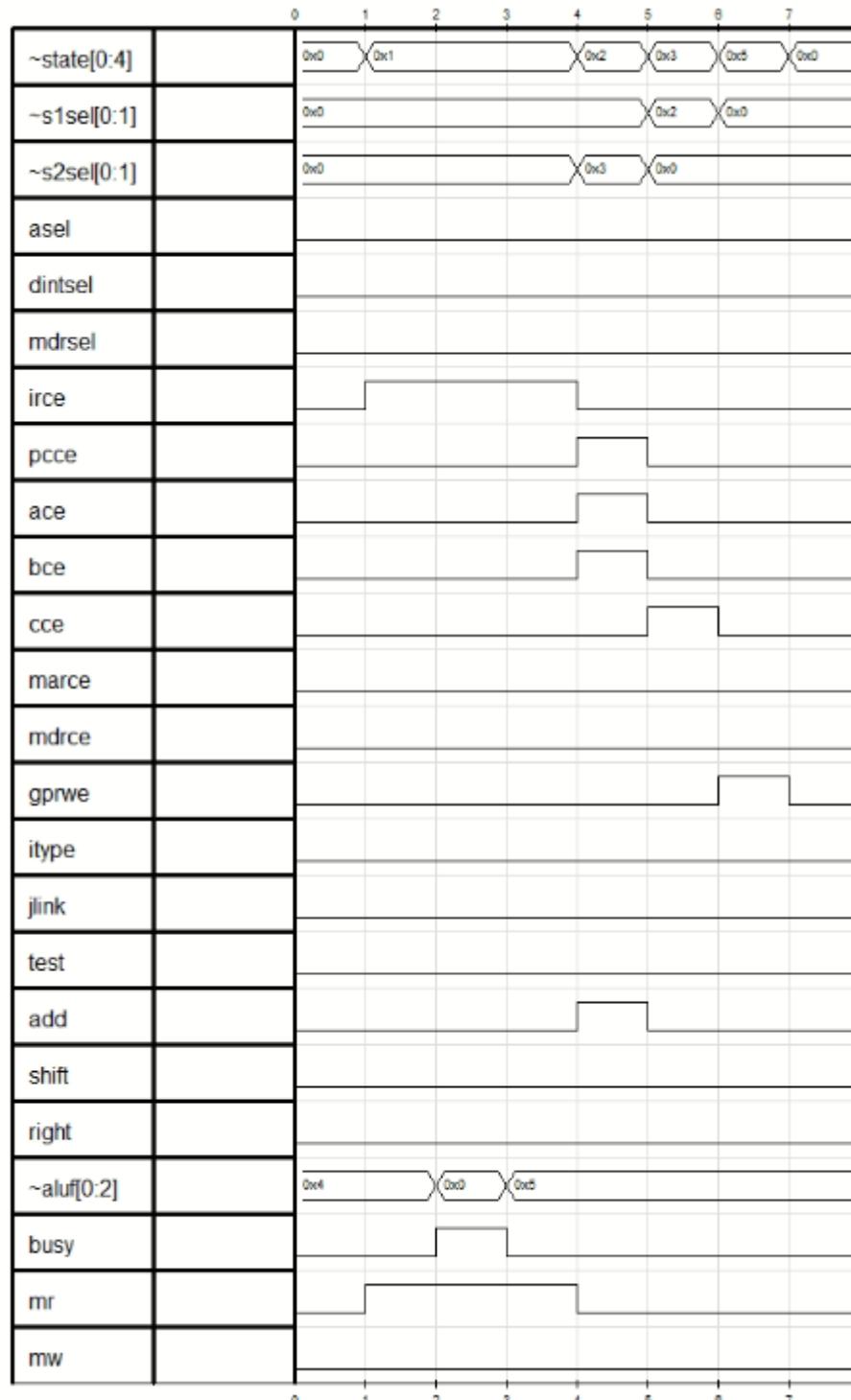
Memory Dump

ADDRESS	VALUES
0x00000000	0x8c01001a 0xac01001d 0x2c220001 0x00221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0x00000008	0xac24001c 0x8c27001c 0x6c880000 0x151f0001 0x00442025 0xc3ff0000 0x2c030012 0x5c7f0000
0x00000010	0x00a11822 0x107f0001 0xbffff000 0x6803ffff 0x8c02001b 0x74450000 0x78860006 0x0803024
0x00000018	0x00c41025 0xfffff000 0x00000001 0xffffffff 0x00000000 0x00000004 0xac040006 0xffff7400

Mem Dump From: 0x00000000 Update Dump

R2 = 4

Waveforms:



PC = 0x19

Instruction: halt

Formula: -

PC = PC + 1 = 0x1a

Before step enable:

Slave Labels

```
laram = 0x10000000
d = 0x00000000
STATUS = 0xa2000008
PC = 0x00000019
```

Commands

Registers

0	0x00000000	16	0x00000000
1	0x00000001	17	0x00000000
2	0x00000004	18	0x00000000
3	0x00000000	19	0x00000000
4	0x00000004	20	0x00000000
5	0x00000001	21	0x00000000
6	0x00000004	22	0x00000000
7	0x00000004	23	0x00000000
8	0x00000001	24	0x00000000
9	0x00000000	25	0x00000000
10	0x00000000	26	0x00000000
11	0x00000000	27	0x00000000
12	0x00000000	28	0x00000000
13	0x00000000	29	0x00000000
14	0x00000000	30	0x00000000
15	0x00000000	31	0x00000000

Memory Labels

```
mem0 = 0xb0c01001a
mem1 = 0xac01001d
mem2 = 0x2c220001
mem3 = 0x00221823
mem4 = 0x005f2802
mem5 = 0x005f2000
```

Write

Memory Dump

ADDRESS	VALUES
0x00000000	0xb0c01001a 0xac01001d 0x2c220001 0x00221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0x00000008	0xac24001c 0x8c27001c 0x6c880000 0x151f0001 0x00442025 0xc3ff0000 0x2c030012 0x5c7f0000
0x00000010	0x00a11822 0x107f0001 0x5bff0000 0x6803ffff 0x8c02001b 0x74450000 0x78860006 0x00803024
0x00000018	0x00c41025 0xffff0000 0x00000001 0xefffffff 0x00000000 0x00000004 0xac040006 0xffff7400

Mem Dump From: 0x00000000 Update Dump

After step enable:

Slave Labels

```
laram = 0x14000000
d = 0x00000000
STATUS = 0xa2000006
PC = 0x0000001a
```

Commands

Registers

0	0x00000000	16	0x00000000
1	0x00000001	17	0x00000000
2	0x00000004	18	0x00000000
3	0x00000000	19	0x00000000
4	0x00000004	20	0x00000000
5	0x00000001	21	0x00000000
6	0x00000004	22	0x00000000
7	0x00000004	23	0x00000000
8	0x00000001	24	0x00000000
9	0x00000000	25	0x00000000
10	0x00000000	26	0x00000000
11	0x00000000	27	0x00000000
12	0x00000000	28	0x00000000
13	0x00000000	29	0x00000000
14	0x00000000	30	0x00000000
15	0x00000000	31	0x00000000

Memory Labels

```
mem0 = 0xb0c01001a
mem1 = 0xac01001d
mem2 = 0x2c220001
mem3 = 0x00221823
mem4 = 0x005f2802
mem5 = 0x005f2000
```

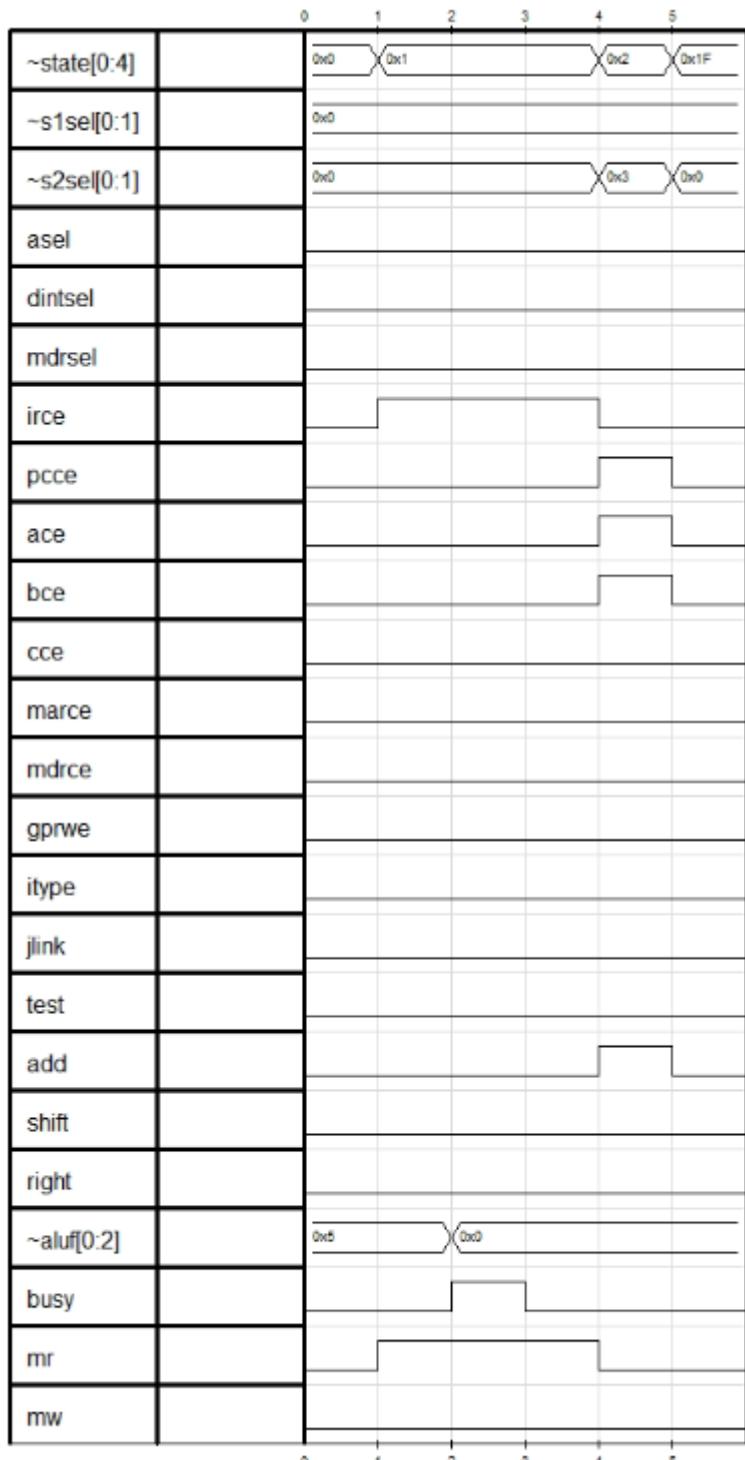
Write

Memory Dump

ADDRESS	VALUES
0x00000000	0xb0c01001a 0xac01001d 0x2c220001 0x00221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0x00000008	0xac24001c 0x8c27001c 0x6c880000 0x151f0001 0x00442025 0xc3ff0000 0x2c030012 0x5c7f0000
0x00000010	0x00a11822 0x107f0001 0x5bff0000 0x6803ffff 0x8c02001b 0x74450000 0x78860006 0x00803024
0x00000018	0x00c41025 0xffff0000 0x00000001 0xefffffff 0x00000000 0x00000004 0xac040006 0xffff7400

Mem Dump From: 0x00000000 Update Dump

Waveforms:



*Now the program is stuck until we press reset and pc will set to 0 if we try to press step we will receive an error.

After step enable:



After reset:

The screenshot displays a debugger interface with several panes. On the left, 'Slave Labels' and 'Memory Labels' are listed. In the center, 'Commands' and 'Registers' are shown. The 'Registers' pane lists 16 registers from 0 to 15 with their corresponding values. Below it, a 'Memory Dump' table shows memory addresses and their values. A box highlights the 'PC = 0' value in the 'Registers' pane. A line connects this box to the 'PC' value in the 'Registers' pane.

Registers	Value	Registers	Value
0	0x00000000	16	0x00000000
1	0x00000001	17	0x00000000
2	0x00000004	18	0x00000000
3	0x00000000	19	0x00000000
4	0x00000004	20	0x00000000
5	0x00000001	21	0x00000000
6	0x00000004	22	0x00000000
7	0x00000004	23	0x00000000
8	0x00000001	24	0x00000000
9	0x00000000	25	0x00000000
10	0x00000000	26	0x00000000
11	0x00000000	27	0x00000000
12	0x00000000	28	0x00000000
13	0x00000000	29	0x00000000
14	0x00000000	30	0x00000000
15	0x00000000	31	0x00000000

Memory Dump	ADDRESS	VALUES
0x00000000	0x8c01001a	0xac01001d 0x2c220001 0x00221823 0x005f2802 0x005f2000 0x70460003 0x10df0001
0x00000008	0xac24001c	0x8c270001 0x6c880000 0x151f0001 0x0442025 0xc3ff0000 0x2c030012 0x5c7f0000
0x00000010	0x00a11822	0x107f0001 0x5bff0000 0x6803ffff 0x8c02001b 0x74450000 0x78860006 0x00803024
0x00000018	0x00c41025	0xffff0000 0x00000001 0xffffffff 0x00000000 0x00000004 0xac040006 0xffff7400

Conclusion:

All the instructions works as expected in addition all the waveforms are as expected comparing to the test vector we can see that the states travers as expected and the outputs in each state are as expected as well therefore we can conclude our DLX machine works as defined.