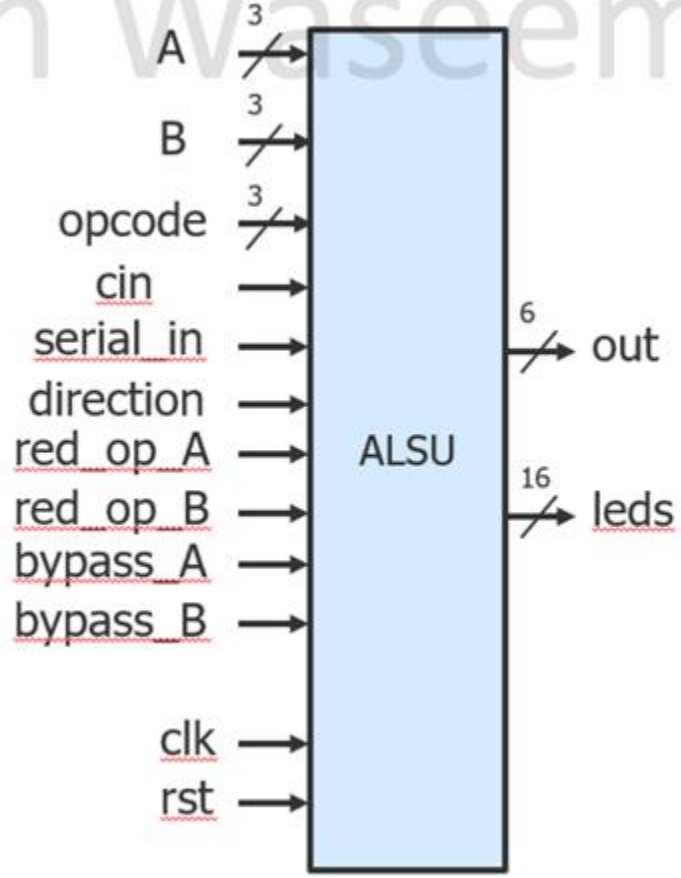
ALSU

Arithmetic Logic Shift Unit



By: Khaled Ahmed Hamed

Under supervision of eng. Kareem Waseem

For more details, [Github Repo]

Design Specs:

Inputs

Each input bit except for the clk and rst will have a DFF in front of its port. Any processing will take place from the DFF output.

Input	Width	Description
clk	1	Input clock
rst	1	Active high asynchronous reset
A	3	Input port A
B	3	Input port B
cin	1	Carry in bit, only valid to be used if the parameter FULL_ADDER is "ON"
serial_in	1	Serial in bit, used in shift operations only
red_op_A	1	When set to high, this indicates that reduction operation would be executed on A rather than bitwise operations on A and B when the opcode indicates AND and XOR operations
red_op_B	1	When set to high, this indicates that reduction operation would be executed on B rather than bitwise operations on A and B when the opcode indicates AND and XOR operations
opcode	3	Opcode has a separate table to describe the different operations executed
bypass_A	1	When set to high, this indicates that port A will be registered to the output ignoring the opcode operation
bypass_B	1	When set to high, this indicates that port B will be registered to the output ignoring the opcode operation
direction	1	The direction of the shift or rotation operation is left when this input is set to high; otherwise, it is right.

Outputs and parameters

Output	Width	Description
leds	16	When an invalid operation occurs, all bits blink (bits turn on and then off with each clock cycle). Blinking serves as a warning; otherwise, if a valid operation occurs, it is set to low.
out	6	Output of the ALSU

Parameter	Default value	Description
INPUT_PRIORITY	A	Priority is given to the port set by this parameter whenever there is a conflict. Conflicts can occur in two scenarios, red_op_A and red_op_B are both set to high or bypass_A and bypass_B are both set to high. Legal values for this parameter are A and B
FULL_ADDER	ON	When this parameter has value "ON" then cin input must be considered in the addition operation between A and B. Legal values for this parameter are ON and OFF

Opcodes & Handling invalid cases

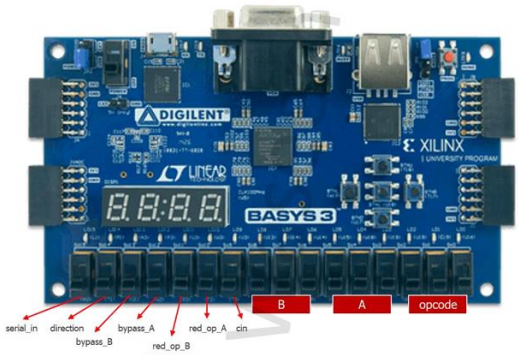
Opcode	Operation
000	AND
001	XOR
010	Addition
011	Multiplication
100	Shift output by 1 bit
101	Rotate output by 1 bit
110	Invalid opcode
111	Invalid opcode

<u>Invalid cases</u>

1. Opcode bits are set to 110 or 111
2. red_op_A or red_op_B are set to high and the opcode is not AND or XOR operation

<u>Output when invalid cases occurs</u>

1. leds are blinking
2. out bits are set to low, but if the bypass_A or bypass_B are high then the output will take the value of A or B.



- "clk" is connected to W5 pin as suggested in the board's reference manual with frequency 100 MHz
- "rst" is connected to button U18
- "leds" are connected to the LEDs on the board

Design Flow

Design Code Snippet:

```
module ALSU (A,B,opcode,cin,serial_in,direction,red_op_A,red_op_B,bypass_A,bypass_B,clk,rst,out,leds);
input clk,rst,cin,serial_in,red_op_A,red_op_B,bypass_A,bypass_B,direction;
reg cin_FF,serial_in_FF,red_op_A_FF,red_op_B_FF,bypass_A_FF,bypass_B_FF,direction_FF;
input [2:0] A,B,opcode;
reg [2:0] A_FF,B_FF,opcode_FF;
output reg [5:0] out;
output reg [15:0] leds;
parameter INPUT_PRIORITY = "A";
parameter FULL_ADDER = "ON";
wire invalid_red_op, invalid_opcode, invalid;
assign invalid_red_op = (red_op_A_FF | red_op_B_FF) && (opcode_FF[1] | opcode_FF[2]);
assign invalid_opcode = opcode_FF[1] & opcode_FF[2];
assign invalid = invalid_red_op | invalid_opcode;
//First Register Level
always @(posedge clk or posedge rst) begin
    if (rst) begin
        A_FF <= 0;B_FF <= 0;opcode_FF <= 0;cin_FF<=0 ; serial_in_FF<=0;
        red_op_A_FF<=0;red_op_B_FF<=0;bypass_A_FF<=0;bypass_B_FF<=0;direction_FF<=0;
    end
    else begin
        A_FF <= A;B_FF <= B;opcode_FF <= opcode;cin_FF<=cin ;serial_in_FF<=serial_in;red_op_A_FF<=red_op_A;
        red_op_B_FF<=red_op_B;bypass_A_FF<=bypass_A;bypass_B_FF<=bypass_B;direction_FF<=direction;
    end
end
//ALSU Functionality
always @(posedge clk or posedge rst) begin
    if (rst) begin
        out<=0;
        leds <= 0;
    end
    else if (invalid) begin
            out<=0;
            leds <= ~leds;
    end
    else if ( bypass_A && bypass_B) begin
        if (INPUT_PRIORITY=="A") begin
            out<=A_FF;
        end
        else if (INPUT_PRIORITY=="B") begin
            out<=B_FF;
        end
    end
    else if (bypass_A) begin
        out<=A_FF;
    end
    else if (bypass_B) begin
        out<=B_FF;
    end
    else begin //Valid Arithmetic Operations
        case (opcode_FF)
            3'b000 : begin
                if (red_op_A && red_op_B) begin
                    if (INPUT_PRIORITY=="A") begin
                        out<= &A_FF ;
                    end
```

```
                end
                else if (INPUT_PRIORITY=="B") begin
                    out<= &B_FF;
                end
            end
            else if(red_op_A) begin
                out<= &A_FF;
            end
            else if (red_op_B) begin
                out<= &B_FF;
            end
            else begin
                out<= A & B;
            end
        end
        3'b001 : begin
            if (red_op_A && red_op_B) begin
                if (INPUT_PRIORITY=="A") begin
                    out<= ^A_FF;
                end
                else if (INPUT_PRIORITY=="B") begin
                    out<=^B_FF;
                end
            end
            else if(red_op_A) begin
                out<=^A_FF;
            end
            else if (red_op_B) begin
                out<=^B_FF;
            end
            else begin
                out<= A_FF ^ B_FF;
            end
        end
        3'b010 : begin
            if (FULL_ADDER=="ON") begin
                out<=A_FF+B_FF+cin_FF;
            end
            else begin
                out<=A_FF+B_FF;
            end
        end
        3'b011 : begin
            out<= A_FF * B_FF;
        end
        3'b100 : begin
            if (direction_FF) begin
                out<={out[4:0],serial_in_FF};
            end
            else begin
                out<={serial_in_FF,out[5:1]};
            end
        end
        3'b101 : begin
            if (direction_FF) begin
```

```
        3'b101 : begin
            if (direction_FF) begin
                out<={out[4:0],out[5]};
            end
            else begin
                out<={out[0],out[5:1]};
            end
        end
        end
    endcase
  end
end
endmodule
```

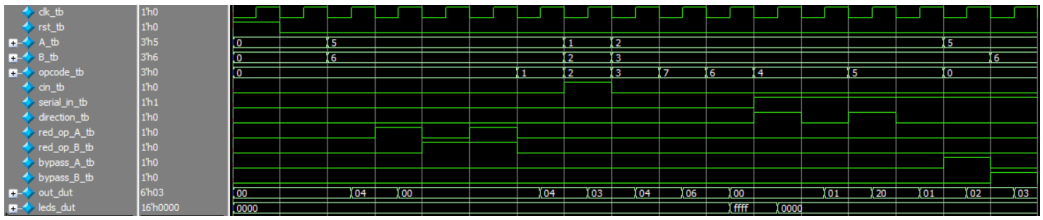
Testbench Code Snippet:

```
`timescale 1ns/1ps

module ALSU_tb ();
// Signals Declaration
reg [2:0] A_tb, B_tb;
reg [2:0] opcode_tb;
reg cin_tb, serial_in_tb, direction_tb, red_op_A_tb, red_op_B_tb, bypass_A_tb, bypass_B_tb;
reg clk_tb, rst_tb;
wire [5:0] out_dut;
wire [15:0] leds_dut;

// Clock Generation
initial begin
    clk_tb = 0;
    forever
    #5 clk_tb = ~clk_tb;
end

// DUT instantiation
ALSU DUT (
    .A(A_tb), .B(B_tb), .opcode(opcode_tb), .cin(cin_tb), .serial_in(serial_in_tb),
    .direction(direction_tb), .red_op_A(red_op_A_tb), .red_op_B(red_op_B_tb),
    .bypass_A(bypass_A_tb), .bypass_B(bypass_B_tb), .clk(clk_tb), .rst(rst_tb),
    .out(out_dut), .leds(leds_dut)
    );

// Test Stimulus Generator
initial begin
    // Initialize all inputs
    rst_tb = 0;
    cin_tb = 0;
    serial_in_tb = 0;
    direction_tb = 0;
    red_op_A_tb = 0;
    red_op_B_tb = 0;
    bypass_A_tb = 0;
    bypass_B_tb = 0;
    A_tb = 0;
    B_tb = 0;
    opcode_tb = 0;

    // Test reset
    rst_tb = 1;
    @(negedge clk_tb);
    rst_tb = 0;
    @(negedge clk_tb);

    // Test case: AND Operation (no reduction, no bypass)
    opcode_tb = 3'b000;
    A_tb = 3'b101;
    B_tb = 3'b110;
    @(negedge clk_tb);
    $display("AND Operation: out_dut = %b", out_dut);
```

```
    // Test case: AND Operation (reduction A)
    red_op_A_tb = 1;
    @(negedge clk_tb);
    $display("AND Operation (reduction A): out_dut = %b", out_dut);
    red_op_A_tb = 0;

    // Test case: AND Operation (reduction B)
    red_op_B_tb = 1;
    @(negedge clk_tb);
    $display("AND Operation (reduction B): out_dut = %b", out_dut);
    red_op_B_tb = 0;

    // Test case: AND Operation (reduction A and B)
    red_op_A_tb = 1;
    red_op_B_tb = 1;
    @(negedge clk_tb);
    $display("AND Operation (reduction A and B): out_dut = %b", out_dut);
    red_op_A_tb = 0;
    red_op_B_tb = 0;

    // Test case: XOR Operation
    opcode_tb = 3'b001;
    A_tb = 3'b101;
    B_tb = 3'b110;
    @(negedge clk_tb);
    $display("XOR Operation: out_dut = %b", out_dut);

    // Test case: ADD Operation
    opcode_tb = 3'b010;
    A_tb = 3'b001;
    B_tb = 3'b010;
    cin_tb = 1;
    @(negedge clk_tb);
    $display("ADD Operation: out_dut = %b", out_dut);
    cin_tb = 0;

    // Test case: Multiplication Operation
    opcode_tb = 3'b011;
    A_tb = 3'b010;
    B_tb = 3'b011;
    @(negedge clk_tb);
    $display("Multiplication Operation: out_dut = %b", out_dut);

    // Test case: Invalid Operation
    opcode_tb = 3'b111; // Invalid opcode
    @(negedge clk_tb);
    $display("Invalid Operation: out_dut = %b, leds_dut = %b", out_dut, leds_dut);

    // Test case: Invalid Operation
    opcode_tb = 3'b110; // Invalid opcode
    @(negedge clk_tb);
    $display("Invalid Operation: out_dut = %b, leds_dut = %b", out_dut, leds_dut);
```

```
    // Test case: Shift Operation (left)
    opcode_tb = 3'b100;
    serial_in_tb = 1;
    direction_tb = 1;
    @(negedge clk_tb);
    $display("Shift Operation (left): out_dut = %b", out_dut);

    // Test case: Shift Operation (right)
    direction_tb = 0;
    @(negedge clk_tb);
    $display("Shift Operation (right): out_dut = %b", out_dut);

    // Test case: Rotate Operation (left)
    opcode_tb = 3'b101;
    direction_tb = 1;
    @(negedge clk_tb);
    $display("Rotate Operation (left): out_dut = %b", out_dut);

    // Test case: Rotate Operation (right)
    direction_tb = 0;
    @(negedge clk_tb);
    $display("Rotate Operation (right): out_dut = %b", out_dut);

    // Test case: Bypass A
    bypass_A_tb = 1;
    A_tb = 3'b101;
    opcode_tb = 3'b000;
    @(negedge clk_tb);
    $display("Bypass A: out_dut = %b", out_dut);
    bypass_A_tb = 0;

    // Test case: Bypass B
    bypass_B_tb = 1;
    B_tb = 3'b110;
    @(negedge clk_tb);
    $display("Bypass B: out_dut = %b", out_dut);
    bypass_B_tb = 0;

    $stop;
end

//Test Monitor & Results
initial begin
    $monitor("clk_tb = %b ,rst_tb = %b , A_tb = %b , B_tb = %b , opcode_tb = %b , cin_tb = %b , serial_in_tb = %b , direction_tb = %b , red_op_A_tb = %b , red_op_B_tb = %b ,bypass_A_tb = %b ,
        bypass_B_tb = %b , out_dut = %b , leds_dut = %b",
            clk_tb, rst_tb, A_tb, B_tb, opcode_tb, cin_tb, serial_in_tb, direction_tb, red_op_A_tb, red_op_B_tb, bypass_A_tb, bypass_B_tb, out_dut, leds_dut);
end

endmodule
```

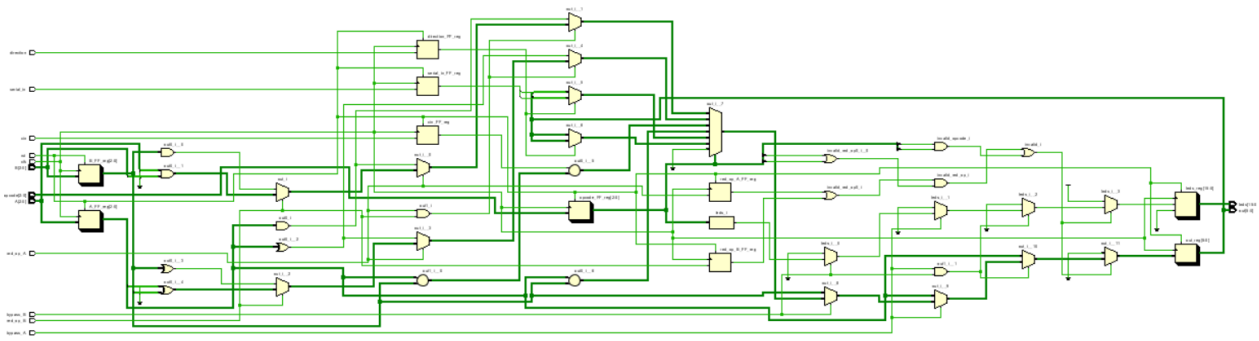
Testbench Simulation:



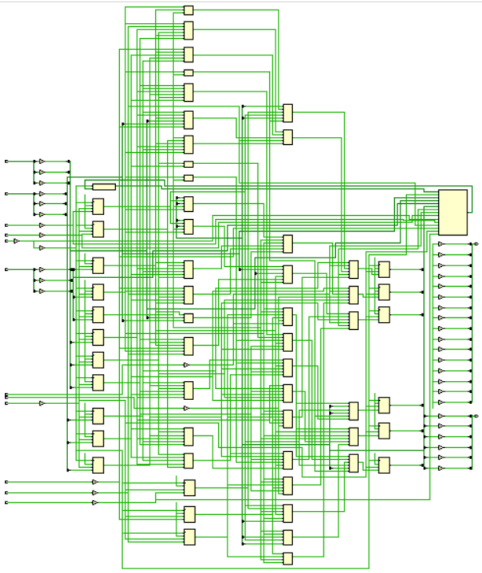
Do file Snippet:

```
vlib work
vlog ALSU.v ALSU_tb.v
vsim -voptargs=+acc work.ALSU_tb
add wave *
run -all
#quit -sim
```

Schematic Using Vivado After Elaboration:



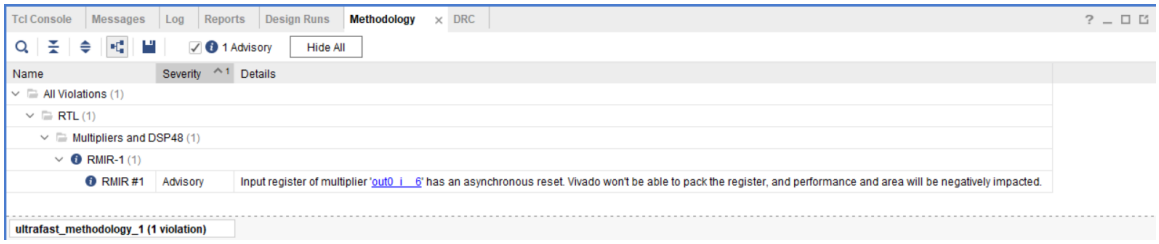
Schematic Using Vivado After Synthesis:



Snippet from the utilization & timing report & after the synthesis and

implementation:

Elaboration:



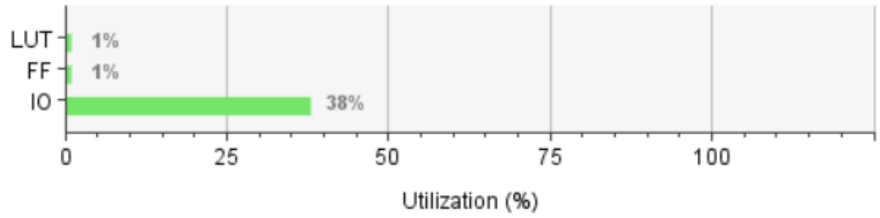
Synthesis :

Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	6.481 ns	Worst Hold Slack (WHS):	0.294 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	6	Total Number of Endpoints:	6	Total Number of Endpoints:	21

Summary

Resource	Utilization	Available	Utilization %
LUT	34	20800	0.16
FF	20	41600	0.05
IO	40	106	37.74



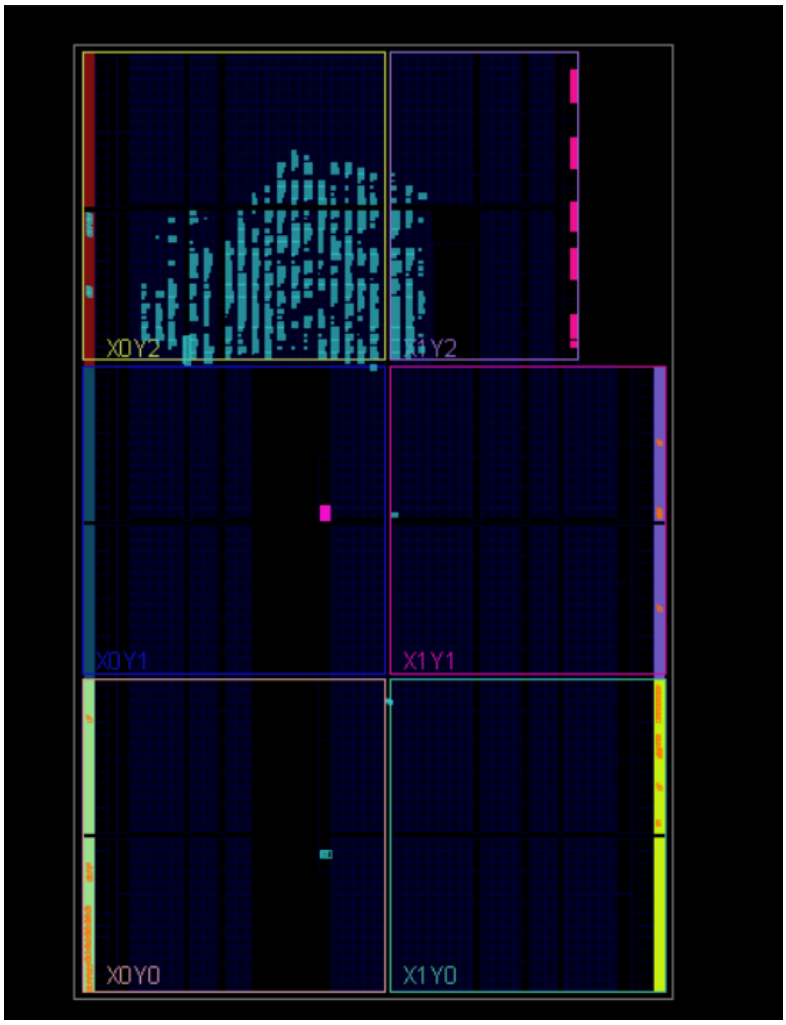
Synthesis message

- Synthesized Design (6 infos, 2 status messages)
  - General Messages (6 infos, 2 status messages)
    - [Netlist 29-17] Analyzing 18 Unisim elements for replacement
    - [Netlist 29-28] Unisim Transformation completed in 0 CPU seconds
    - [Project 1-479] Netlist was created with Vivado 2018.2
    - [Project 1-570] Preparing netlist for logic optimization
    - Parsing XDC File [Constraints_basys4.xdc] (1 more like this)
    - [Opt 31-138] Pushed 0 inverter(s) to 0 load pin(s).
    - [Project 1-111] Unisim Transformation Summary:
      No Unisim elements were transformed.

Implementation message

- Implemented Design (1 warning, 9 infos, 4 status messages)
  - General Messages (1 warning, 9 infos, 4 status messages)
    - [Netlist 29-17] Analyzing 189 Unisim elements for replacement
    - [Netlist 29-28] Unisim Transformation completed in 1 CPU seconds
    - [Project 1-479] Netlist was created with Vivado 2018.2
    - [Project 1-570] Preparing netlist for logic optimization
    - [Timing 38-478] Restoring timing data from binary archive.
    - [Timing 38-479] Binary timing data restore complete.
    - [Project 1-856] Restoring constraints from binary archive.
    - [Project 1-853] Binary constraint restore complete.
    - Reading XDEF placement. (3 more like this)
    - [Project 1-111] Unisim Transformation Summary:
      A total of 98 instances were transformed.
      CFGLUT5 => CFGLUT5 (SRLC32E, SRL16E): 92 instances
      RAM32M => RAM32M (RAMD32, RAMD32, RAMD32, RAMD32, RAMD32, RAMD32, RAMS32, RAMS32): 6 instances

Implementation:

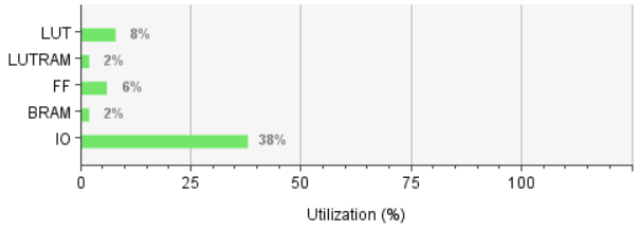


Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	2.003 ns	Worst Hold Slack (WHS):	0.043 ns	Worst Pulse Width Slack (WPWS):	3.750 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	4644	Total Number of Endpoints:	4628	Total Number of Endpoints:	2699

Summary

Resource	Utilization	Available	Utilization %
LUT	1591	20800	7.65
LUTRAM	160	9600	1.67
FF	2415	41600	5.81
BRAM	1	50	2.00
IO	40	106	37.74



Constraints File:

```
## Clock signal
set_property -dict {PACKAGE_PIN W5 IOSTANDARD LVCMOS33} [get_ports clk]
create_clock -period 10.000 -name sys_clk_pin -waveform {0.000 5.000} -add [get_ports clk]


## Switches , Inputs
set_property -dict {PACKAGE_PIN V17 IOSTANDARD LVCMOS33} [get_ports {opcode[0]}]
set_property -dict {PACKAGE_PIN V16 IOSTANDARD LVCMOS33} [get_ports {opcode[1]}]
set_property -dict {PACKAGE_PIN W16 IOSTANDARD LVCMOS33} [get_ports {opcode[2]}]
set_property -dict {PACKAGE_PIN W17 IOSTANDARD LVCMOS33} [get_ports {A[0]}]
set_property -dict {PACKAGE_PIN W15 IOSTANDARD LVCMOS33} [get_ports {A[1]}]
set_property -dict {PACKAGE_PIN V15 IOSTANDARD LVCMOS33} [get_ports {A[2]}]
set_property -dict {PACKAGE_PIN W14 IOSTANDARD LVCMOS33} [get_ports {B[0]}]
set_property -dict {PACKAGE_PIN W13 IOSTANDARD LVCMOS33} [get_ports {B[1]}]
set_property -dict {PACKAGE_PIN V2 IOSTANDARD LVCMOS33} [get_ports {B[2]}]
set_property -dict {PACKAGE_PIN T3 IOSTANDARD LVCMOS33} [get_ports cin]
set_property -dict {PACKAGE_PIN T2 IOSTANDARD LVCMOS33} [get_ports red_op_A]
set_property -dict {PACKAGE_PIN R3 IOSTANDARD LVCMOS33} [get_ports red_op_B]
set_property -dict {PACKAGE_PIN W2 IOSTANDARD LVCMOS33} [get_ports bypass_A]
set_property -dict {PACKAGE_PIN U1 IOSTANDARD LVCMOS33} [get_ports bypass_B]
set_property -dict {PACKAGE_PIN T1 IOSTANDARD LVCMOS33} [get_ports direction]
set_property -dict {PACKAGE_PIN R2 IOSTANDARD LVCMOS33} [get_ports serial_in]


## LEDs , Output
set_property -dict {PACKAGE_PIN U16 IOSTANDARD LVCMOS33} [get_ports {leds[0]}]
set_property -dict {PACKAGE_PIN E19 IOSTANDARD LVCMOS33} [get_ports {leds[1]}]
set_property -dict {PACKAGE_PIN U19 IOSTANDARD LVCMOS33} [get_ports {leds[2]}]
set_property -dict {PACKAGE_PIN V19 IOSTANDARD LVCMOS33} [get_ports {leds[3]}]
set_property -dict {PACKAGE_PIN W18 IOSTANDARD LVCMOS33} [get_ports {leds[4]}]
set_property -dict {PACKAGE_PIN U15 IOSTANDARD LVCMOS33} [get_ports {leds[5]}]
set_property -dict {PACKAGE_PIN U14 IOSTANDARD LVCMOS33} [get_ports {leds[6]}]
set_property -dict {PACKAGE_PIN V14 IOSTANDARD LVCMOS33} [get_ports {leds[7]}]
set_property -dict {PACKAGE_PIN V13 IOSTANDARD LVCMOS33} [get_ports {leds[8]}]
set_property -dict {PACKAGE_PIN V3 IOSTANDARD LVCMOS33} [get_ports {leds[9]}]
set_property -dict {PACKAGE_PIN W3 IOSTANDARD LVCMOS33} [get_ports {leds[10]}]
set_property -dict {PACKAGE_PIN U3 IOSTANDARD LVCMOS33} [get_ports {leds[11]}]
set_property -dict {PACKAGE_PIN P3 IOSTANDARD LVCMOS33} [get_ports {leds[12]}]
set_property -dict {PACKAGE_PIN N3 IOSTANDARD LVCMOS33} [get_ports {leds[13]}]
set_property -dict {PACKAGE_PIN P1 IOSTANDARD LVCMOS33} [get_ports {leds[14]}]
set_property -dict {PACKAGE_PIN L1 IOSTANDARD LVCMOS33} [get_ports {leds[15]}]
```

```
##Buttons , Reset
set_property -dict {PACKAGE_PIN U18 IOSTANDARD LVCMOS33} [get_ports rst]
#set_property -dict { PACKAGE_PIN T18   IOSTANDARD LVCMOS33 } [get_ports btnU]
#set_property -dict { PACKAGE_PIN W19   IOSTANDARD LVCMOS33 } [get_ports btnL]
#set_property -dict { PACKAGE_PIN T17   IOSTANDARD LVCMOS33 } [get_ports btnR]
#set_property -dict { PACKAGE_PIN U17   IOSTANDARD LVCMOS33 } [get_ports btnD]
```

```
## Configuration options, can be used for all designs
set_property CONFIG_VOLTAGE 3.3 [current_design]
set_property CFGBVS VCCO [current_design]

## SPI configuration mode options for QSPI boot, can be used for all designs
set_property BITSTREAM.GENERAL.COMPRESS TRUE [current_design]
set_property BITSTREAM.CONFIG.CONFIGRATE 33 [current_design]
set_property CONFIG_MODE SPIx4 [current_design]


create_debug_core u_ila_0 ila
set_property ALL_PROBE_SAME_MU true [get_debug_cores u_ila_0]
set_property ALL_PROBE_SAME_MU_CNT 1 [get_debug_cores u_ila_0]
set_property C_ADV_TRIGGER false [get_debug_cores u_ila_0]
set_property C_DATA_DEPTH 1024 [get_debug_cores u_ila_0]
set_property C_EN_STRG_QUAL false [get_debug_cores u_ila_0]
set_property C_INPUT_PIPE_STAGES 0 [get_debug_cores u_ila_0]
set_property C_TRIGIN_EN false [get_debug_cores u_ila_0]
set_property C_TRIGOUT_EN false [get_debug_cores u_ila_0]
set_property port_width 1 [get_debug_ports u_ila_0/clk]
connect_debug_port u_ila_0/clk [get_nets [list clk_IBUF_BUFG]]
set_property PROBE_TYPE DATA_AND_TRIGGER [get_debug_ports u_ila_0/probe0]
set_property port_width 6 [get_debug_ports u_ila_0/probe0]
connect_debug_port u_ila_0/probe0 [get_nets [list {out_OBUF[0]} {out_OBUF[1]} {out_OBUF[2]} {out_OBUF[3]} {out_OBUF[
create_debug_port u_ila_0 probe
set_property PROBE_TYPE DATA_AND_TRIGGER [get_debug_ports u_ila_0/probe1]
set_property port_width 3 [get_debug_ports u_ila_0/probe1]
connect_debug_port u_ila_0/probe1 [get_nets [list {B_IBUF[0]} {B_IBUF[1]} {B_IBUF[2]}]]
create_debug_port u_ila_0 probe
set_property PROBE_TYPE DATA_AND_TRIGGER [get_debug_ports u_ila_0/probe2]
set_property port_width 3 [get_debug_ports u_ila_0/probe2]
connect_debug_port u_ila_0/probe2 [get_nets [list {opcode_IBUF[0]} {opcode_IBUF[1]} {opcode_IBUF[2]}]]
create_debug_port u_ila_0 probe
set_property PROBE_TYPE DATA_AND_TRIGGER [get_debug_ports u_ila_0/probe3]
set_property port_width 3 [get_debug_ports u_ila_0/probe3]
connect_debug_port u_ila_0/probe3 [get_nets [list {A_IBUF[0]} {A_IBUF[1]} {A_IBUF[2]}]]
create_debug_port u_ila_0 probe
set_property PROBE_TYPE DATA_AND_TRIGGER [get_debug_ports u_ila_0/probe4]
set_property port_width 1 [get_debug_ports u_ila_0/probe4]
connect_debug_port u_ila_0/probe4 [get_nets [list bypass_A_IBUF]]
create_debug_port u_ila_0 probe
set_property PROBE_TYPE DATA_AND_TRIGGER [get_debug_ports u_ila_0/probe5]
set_property port_width 1 [get_debug_ports u_ila_0/probe5]
connect_debug_port u_ila_0/probe5 [get_nets [list bypass_B_IBUF]]
create_debug_port u_ila_0 probe
set_property PROBE_TYPE DATA_AND_TRIGGER [get_debug_ports u_ila_0/probe6]
set_property port_width 1 [get_debug_ports u_ila_0/probe6]
connect_debug_port u_ila_0/probe6 [get_nets [list cin_IBUF]]
create_debug_port u_ila_0 probe
set_property PROBE_TYPE DATA_AND_TRIGGER [get_debug_ports u_ila_0/probe7]
set_property port_width 1 [get_debug_ports u_ila_0/probe7]
connect_debug_port u_ila_0/probe7 [get_nets [list clk_IBUF]]
```

```
set_property port_width 1 [get_debug_ports u_ila_0/probe7]
connect_debug_port u_ila_0/probe7 [get_nets [list clk_IBUF]]
create_debug_port u_ila_0 probe
set_property PROBE_TYPE DATA_AND_TRIGGER [get_debug_ports u_ila_0/probe8]
set_property port_width 1 [get_debug_ports u_ila_0/probe8]
connect_debug_port u_ila_0/probe8 [get_nets [list direction_IBUF]]
create_debug_port u_ila_0 probe
set_property PROBE_TYPE DATA_AND_TRIGGER [get_debug_ports u_ila_0/probe9]
set_property port_width 1 [get_debug_ports u_ila_0/probe9]
connect_debug_port u_ila_0/probe9 [get_nets [list red_op_A_IBUF]]
create_debug_port u_ila_0 probe
set_property PROBE_TYPE DATA_AND_TRIGGER [get_debug_ports u_ila_0/probe10]
set_property port_width 1 [get_debug_ports u_ila_0/probe10]
connect_debug_port u_ila_0/probe10 [get_nets [list red_op_B_IBUF]]
create_debug_port u_ila_0 probe
set_property PROBE_TYPE DATA_AND_TRIGGER [get_debug_ports u_ila_0/probe11]
set_property port_width 1 [get_debug_ports u_ila_0/probe11]
connect_debug_port u_ila_0/probe11 [get_nets [list rst_IBUF]]
create_debug_port u_ila_0 probe
set_property PROBE_TYPE DATA_AND_TRIGGER [get_debug_ports u_ila_0/probe12]
set_property port_width 1 [get_debug_ports u_ila_0/probe12]
connect_debug_port u_ila_0/probe12 [get_nets [list serial_in_IBUF]]
set_property C_CLK_INPUT_FREQ_HZ 300000000 [get_debug_cores dbg_hub]
set_property C_ENABLE_CLK_DIVIDER false [get_debug_cores dbg_hub]
set_property C_USER_SCAN_CHAIN 1 [get_debug_cores dbg_hub]
connect_debug_port dbg_hub/clk [get_nets clk_IBUF_BUFG]
```

Verification using UVM

Full Environment, Assertions & Golden Model Reference

Including shift register inside

Adding the shift register to the ALSU:

Design code snippets after adjustments:

```
module ALSU(ALSU_interface.DUT ALSU_if);
// Parameters & Internal signals
parameter INPUT_PRIORITY = "A";
parameter FULL_ADDER = "ON";
reg red_op_A_reg, red_op_B_reg, bypass_A_reg, bypass_B_reg, direction_reg, serial_in_reg;
reg signed [1:0] cin_reg;//Make cin_reg signed 2bits to perform correct signed addition
reg [2:0] opcode_reg;
reg signed [2:0] A_reg, B_reg;//A & B registers are signed as well as A & B
wire invalid_red_op, invalid_opcode, invalid;

//Invalid handling
assign invalid_red_op = (red_op_A_reg | red_op_B_reg) & (opcode_reg[1] | opcode_reg[2]);
assign invalid_opcode = opcode_reg[1] & opcode_reg[2];
assign invalid = invalid_red_op | invalid_opcode;
//Registering input signals
always @(posedge ALSU_if.clk or posedge ALSU_if.rst) begin
  if(ALSU_if.rst) begin
     cin_reg <= 0;
     red_op_B_reg <= 0;
     red_op_A_reg <= 0;
     bypass_B_reg <= 0;
     bypass_A_reg <= 0;
     direction_reg <= 0;
     serial_in_reg <= 0;
     opcode_reg <= 0;
     A_reg <= 0;
     B_reg <= 0;
  end else begin
     cin_reg <= ALSU_if.cin;
     red_op_B_reg <= ALSU_if.red_op_B;
     red_op_A_reg <= ALSU_if.red_op_A;
     bypass_B_reg <= ALSU_if.bypass_B;
     bypass_A_reg <= ALSU_if.bypass_A;
     direction_reg <= ALSU_if.direction;
     serial_in_reg <= ALSU_if.serial_in;
     opcode_reg <= ALSU_if.opcode;
     A_reg <= ALSU_if.A;
     B_reg <= ALSU_if.B;
  end
end
//leds output blinking
always @(posedge ALSU_if.clk or posedge ALSU_if.rst) begin
  if(ALSU_if.rst) begin
     ALSU_if.leds <= 0;
  end else begin
      if (invalid)
        ALSU_if.leds <= ~ALSU_if.leds;
      else
        ALSU_if.leds <= 0;
  end
end
//ALSU output processing
always @(posedge ALSU_if.clk or posedge ALSU_if.rst) begin
  if(ALSU_if.rst) begin
    ALSU_if.out <= 0;
```

```
    end
    else begin
      if (invalid)
          ALSU_if.out <= 0;
      else if (bypass_A_reg && bypass_B_reg)
        ALSU_if.out <= (INPUT_PRIORITY == "A")? A_reg: B_reg;
      else if (bypass_A_reg)
        ALSU_if.out <= A_reg;
      else if (bypass_B_reg)
        ALSU_if.out <= B_reg;
      else begin
          case (opcode_reg)//Bug: opcode --> opcode_reg
            3'h0: begin
              //Third bug --> AND replaced with OR
              if (red_op_A_reg && red_op_B_reg)
                ALSU_if.out = (INPUT_PRIORITY == "A")? |A_reg: |B_reg;
              else if (red_op_A_reg)
                ALSU_if.out <= |A_reg;
              else if (red_op_B_reg)
                ALSU_if.out <= |B_reg;
              else
                ALSU_if.out <= A_reg | B_reg;
            end
            3'h1: begin
              //Fourth bug --> OR replaced with XOR
              if (red_op_A_reg && red_op_B_reg)
                ALSU_if.out <= (INPUT_PRIORITY == "A")? ^A_reg: ^B_reg;
              else if (red_op_A_reg)
                ALSU_if.out <= ^A_reg;
              else if (red_op_B_reg)
                ALSU_if.out <= ^B_reg;
              else
                ALSU_if.out <= A_reg ^ B_reg;
            end
            //Fifth bug --> Added logic condition for FULL_ADDER operation
            3'h2: if(FULL_ADDER=="ON") begin
              ALSU_if.out <= A_reg + B_reg + cin_reg;
            end
            else if (FULL_ADDER=="OFF") begin
               ALSU_if.out <= A_reg + B_reg ;
            end
            3'h3: ALSU_if.out <= A_reg * B_reg;
            3'h4: begin
              if (direction_reg)
                ALSU_if.out <= ALSU_if.out_shift_reg;
              else
                ALSU_if.out <= ALSU_if.out_shift_reg;
            end
            3'h5: begin
              if (direction_reg)
                ALSU_if.out <= ALSU_if.out_shift_reg;
              else
                ALSU_if.out <= ALSU_if.out_shift_reg;
            end
          endcase
```

```
          endcase
      end
    end
  end
  endmodule
```

Golden Model code snippets:

```
module ALSU_gm (ALSU_interface.GM ALSU_if);
    // Parameters for configurability
    parameter INPUT_PRIORITY = "A";
    parameter FULL_ADDER = "ON";
    // Internal registers to store inputs after clock one cycle
    reg serial_in_FF, red_op_A_FF, red_op_B_FF, bypass_A_FF, bypass_B_FF, direction_FF;
    reg signed [1:0] cin_FF;
    reg signed [2:0] A_FF, B_FF;
    reg [2:0] opcode_FF;
    // Wires for invalid operation detection
    wire invalid_red_op, invalid_opcode, invalid;
    // Invalid operation detection logic
    assign invalid_red_op = (red_op_A_FF | red_op_B_FF) & (opcode_FF[1] | opcode_FF[2]);
    assign invalid_opcode = opcode_FF[1] & opcode_FF[2];
    assign invalid = invalid_red_op | invalid_opcode;
    // First Register Level - Capturing inputs on clock edge
    always @(posedge ALSU_if.clk or posedge ALSU_if.rst) begin
        if (ALSU_if.rst) begin
            A_FF <= 0; B_FF <= 0; opcode_FF <= 0; cin_FF <= 0; serial_in_FF <= 0; red_op_A_FF <= 0;
            red_op_B_FF <= 0; bypass_A_FF <= 0; bypass_B_FF <= 0; direction_FF <= 0;
        end
    else begin
            A_FF <= ALSU_if.A; B_FF <= ALSU_if.B; opcode_FF <= ALSU_if.opcode; cin_FF <= ALSU_if.cin; serial_in_FF <= ALSU_if.serial_in; red_op_A_FF <= ALSU_if.red_op_A;
            red_op_B_FF <= ALSU_if.red_op_B; bypass_A_FF <= ALSU_if.bypass_A; bypass_B_FF <= ALSU_if.bypass_B; direction_FF <= ALSU_if.direction;
        end
    end
    // LED Functionality - Blinking when invalid operation detected
    always @(posedge ALSU_if.clk or posedge ALSU_if.rst) begin
        if (ALSU_if.rst) begin
            ALSU_if.leds_ref <= 0;
        end else begin
            if (invalid) begin
                ALSU_if.leds_ref <= ~ALSU_if.leds_ref;
            end else begin
                ALSU_if.leds_ref <= 0;
            end
        end
    end
    // ALSU Functionality - Core processing logic
    always @(posedge ALSU_if.clk or posedge ALSU_if.rst) begin
        if (ALSU_if.rst) begin
            ALSU_if.out_ref <= 0;
        end else begin
            if (invalid) begin
                ALSU_if.out_ref <= 0;
            end else if (bypass_A_FF && bypass_B_FF) begin
                // Handling the bypass logic for both inputs
                ALSU_if.out_ref <= (INPUT_PRIORITY == "A") ? A_FF : B_FF;
            end else if (bypass_A_FF) begin
                ALSU_if.out_ref <= A_FF;
            end else if (bypass_B_FF) begin
                ALSU_if.out_ref <= B_FF;
            end else begin
                // Valid Arithmetic Operations
```

```
                case (opcode_FF)
                    3'b000: begin
                        // AND operation with reduction if enabled
                        if (red_op_A_FF && red_op_B_FF) begin
                            ALSU_if.out_ref <= (INPUT_PRIORITY == "A") ? |A_FF : |B_FF;
                        end else if (red_op_A_FF) begin
                            ALSU_if.out_ref <= |A_FF;
                        end else if (red_op_B_FF) begin
                            ALSU_if.out_ref <= |B_FF;
                        end else begin
                            ALSU_if.out_ref <= A_FF | B_FF;
                        end
                    end
                    3'b001: begin
                        // XOR operation with reduction if enabled
                        if (red_op_A_FF && red_op_B_FF) begin
                            ALSU_if.out_ref <= (INPUT_PRIORITY == "A") ? ^A_FF : ^B_FF;
                        end else if (red_op_A_FF) begin
                            ALSU_if.out_ref <= ^A_FF;
                        end else if (red_op_B_FF) begin
                            ALSU_if.out_ref <= ^B_FF;
                        end else begin
                            ALSU_if.out_ref <= A_FF ^ B_FF;
                        end
                    end
                    3'b010: begin
                        // ADD operation with optional carry-in based on FULL_ADDER parameter
                        if (FULL_ADDER == "ON") begin
                            ALSU_if.out_ref <= A_FF + B_FF + cin_FF;
                        end else begin
                            ALSU_if.out_ref <= A_FF + B_FF;
                        end
                    end
                    3'b011: begin
                        // Multiplication operation
                        ALSU_if.out_ref <= A_FF * B_FF;
                    end
                    3'b100: begin
                        // Shift operation with direction and serial input
                        if (direction_FF) begin
                            ALSU_if.out_ref <= {ALSU_if.out_ref[4:0], serial_in_FF};
                        end else begin
                            ALSU_if.out_ref <= {serial_in_FF, ALSU_if.out_ref[5:1]};
                        end
                    end
                    3'b101: begin
                        // Rotate operation with direction
                        if (direction_FF) begin
                            ALSU_if.out_ref <= {ALSU_if.out_ref[4:0], ALSU_if.out_ref[5]};
                        end else begin
                            ALSU_if.out_ref <= {ALSU_if.out_ref[0], ALSU_if.out_ref[5:1]};
                        end
                    end
                endcase
```

```
                            end
                        end
                    endcase
                end
            end
        end
    endmodule
```

SVA code snippets:

```
module ALSU_SVA (ALSU_interface.DUT ALSU_if);

    //Make cin signed 2 bits to perform correct signed addition
    logic signed [1:0] cin_signed;
    assign cin_signed = ALSU_if.cin;

    always_comb begin
        if (ALSU_if.rst) begin
            reset_assertion: assert final (ALSU_if.out==0 && ALSU_if.leds==0);
            reset_cover: cover (ALSU_if.out==0 && ALSU_if.leds==0);
        end
    end

    property invalid_property;
        @(posedge ALSU_if.clk) disable iff (ALSU_if.rst)
        (((ALSU_if.red_op_A | ALSU_if.red_op_B) & (ALSU_if.opcode[1] | ALSU_if.opcode[2])) | (ALSU_if.opcode[1] & ALSU_if.opcode[2])) |-> ##2 ((ALSU_if.out == 0) && (ALSU_if.leds===$past(ALSU_if.leds)));
    endproperty

    property bypass_A_property;
        @(posedge ALSU_if.clk) disable iff (ALSU_if.rst)
        ((ALSU_if.bypass_A) && !(((ALSU_if.red_op_A | ALSU_if.red_op_B) & (ALSU_if.opcode[1] | ALSU_if.opcode[2])) | (ALSU_if.opcode[1] & ALSU_if.opcode[2]))) |-> ##2 (ALSU_if.out == $past(ALSU_if.A,2));
    endproperty

    property bypass_B_property;
        @(posedge ALSU_if.clk) disable iff (ALSU_if.rst)
        ((!ALSU_if.bypass_A && ALSU_if.bypass_B) && !(((ALSU_if.red_op_A | ALSU_if.red_op_B) & (ALSU_if.opcode[1] | ALSU_if.opcode[2])) | (ALSU_if.opcode[1] & ALSU_if.opcode[2])))
         |-> ##2 (ALSU_if.out == $past(ALSU_if.B,2));
    endproperty

    property red_OR_A;
        @(posedge ALSU_if.clk) disable iff (ALSU_if.rst)
        ((ALSU_if.opcode==1'h0 && ALSU_if.red_op_A) && !ALSU_if.bypass_A && !ALSU_if.bypass_B) |-> ##2 (ALSU_if.out == |$past(ALSU_if.A,2));
    endproperty

    property red_OR_B;
        @(posedge ALSU_if.clk) disable iff (ALSU_if.rst)
        ((ALSU_if.opcode==1'h0 && ALSU_if.red_op_B && !ALSU_if.red_op_A) && !ALSU_if.bypass_A && !ALSU_if.bypass_B) |-> ##2 (ALSU_if.out == |$past(ALSU_if.B,2));
    endproperty

    property red_OR_A_B;
        @(posedge ALSU_if.clk) disable iff (ALSU_if.rst)
        ((ALSU_if.opcode==1'h0 && !ALSU_if.red_op_B && !ALSU_if.red_op_A) && !ALSU_if.bypass_A && !ALSU_if.bypass_B) |-> ##2 (ALSU_if.out == $past(ALSU_if.A,2) | $past(ALSU_if.B,2));
    endproperty

    property red_XOR_A;
        @(posedge ALSU_if.clk) disable iff (ALSU_if.rst)
        ((ALSU_if.opcode==1'h1 && ALSU_if.red_op_A) && !ALSU_if.bypass_A && !ALSU_if.bypass_B) |-> ##2 (ALSU_if.out == ^$past(ALSU_if.A,2));
    endproperty

    property red_XOR_B;
        @(posedge ALSU_if.clk) disable iff (ALSU_if.rst)
        ((ALSU_if.opcode==1'h1 && ALSU_if.red_op_B && !ALSU_if.red_op_A) && !ALSU_if.bypass_A && !ALSU_if.bypass_B) |-> ##2 (ALSU_if.out == ^$past(ALSU_if.B,2));
    endproperty
```

```
    property red_XOR_A_B;
        @(posedge ALSU_if.clk) disable iff (ALSU_if.rst)
        ((ALSU_if.opcode==1'h1 && !ALSU_if.red_op_B && !ALSU_if.red_op_A) && !ALSU_if.bypass_A && !ALSU_if.bypass_B) |-> ##2 (ALSU_if.out == $past(ALSU_if.A,2) ^ $past(ALSU_if.B,2));
    endproperty

    property Add;
        @(posedge ALSU_if.clk) disable iff (ALSU_if.rst)
        ((ALSU_if.opcode==3'h2 && !ALSU_if.red_op_B && !ALSU_if.red_op_A && !ALSU_if.bypass_A && !ALSU_if.bypass_B))
         |-> ##2 (ALSU_if.out == ($past(ALSU_if.A,2) + $past(ALSU_if.B,2) + $past(cin_signed,2)));
    endproperty

    property Mult;
        @(posedge ALSU_if.clk) disable iff (ALSU_if.rst)
        ((ALSU_if.opcode==3'h3 && !ALSU_if.red_op_B && !ALSU_if.red_op_A) && !ALSU_if.bypass_A && !ALSU_if.bypass_B)
         |-> ##2 (ALSU_if.out == $past(ALSU_if.A,2) * $past(ALSU_if.B,2));
    endproperty

    property shift_left;
        @(posedge ALSU_if.clk) disable iff (ALSU_if.rst)
        ((ALSU_if.opcode==3'h4 && ALSU_if.direction && !ALSU_if.red_op_B && !ALSU_if.red_op_A) && !ALSU_if.bypass_A && !ALSU_if.bypass_B)
         |-> ##2 (ALSU_if.out == {$past(ALSU_if.out[4:0]),$past(ALSU_if.serial_in,2) });
    endproperty

    property shift_right;
        @(posedge ALSU_if.clk) disable iff (ALSU_if.rst)
        ((ALSU_if.opcode==3'h4 && !ALSU_if.direction && !ALSU_if.red_op_B && !ALSU_if.red_op_A) && !ALSU_if.bypass_A && !ALSU_if.bypass_B)
         |-> ##2 (ALSU_if.out == {$past(ALSU_if.serial_in,2),$past(ALSU_if.out[5:1]) });
    endproperty

    property rotate_left;
        @(posedge ALSU_if.clk) disable iff (ALSU_if.rst)
        ((ALSU_if.opcode==3'h5 && ALSU_if.direction && !ALSU_if.red_op_B && !ALSU_if.red_op_A) && !ALSU_if.bypass_A && !ALSU_if.bypass_B)
         |-> ##2 (ALSU_if.out == {$past(ALSU_if.out[4:0]),$past(ALSU_if.out[5]) });
    endproperty

    property rotate_right;
        @(posedge ALSU_if.clk) disable iff (ALSU_if.rst)
        ((ALSU_if.opcode==3'h5 && !ALSU_if.direction && !ALSU_if.red_op_B && !ALSU_if.red_op_A) && !ALSU_if.bypass_A && !ALSU_if.bypass_B)
         |-> ##2 (ALSU_if.out == {$past(ALSU_if.out[0]),$past(ALSU_if.out[5:1]) });
    endproperty

    invalid_property_assertion: assert property (invalid_property);
    invalid_property_cover: cover property (invalid_property);

    bypass_A_property_assertion: assert property (bypass_A_property);
    bypass_A_property_cover: cover property (bypass_A_property);

    bypass_B_property_assertion: assert property (bypass_B_property);
    bypass_B_property_cover: cover property (bypass_B_property);

    red_OR_A_property_assertion: assert property (red_OR_A);
    red_OR_A_property_cover: cover property (red_OR_A);
```

```
    red_OR_B_property_assertion: assert property (red_OR_B);
    red_OR_B_property_cover: cover property (red_OR_B);

    red_OR_A_B_property_assertion: assert property (red_OR_A_B);
    red_OR_A_B_property_cover: cover property (red_OR_A_B);

    red_XOR_A_property_assertion: assert property (red_XOR_A);
    red_XOR_A_property_cover: cover property (red_XOR_A);

    red_XOR_B_property_assertion: assert property (red_XOR_B);
    red_XOR_B_property_cover: cover property (red_XOR_B);

    red_XOR_A_B_property_assertion: assert property (red_XOR_A_B);
    red_XOR_A_B_property_cover: cover property (red_XOR_A_B);

    Add1_assertion: assert property (Add);
    Add1_cover: cover property (Add);

    Mult_assertion: assert property (Mult);
    Mult_cover: cover property (Mult);

    shift_left_assertion: assert property (shift_left);
    shift_left_cover: cover property (shift_left);

    shift_right_assertion: assert property (shift_right);
    shift_right_cover: cover property (shift_right);

    rotate_left_assertion: assert property (rotate_left);
    rotate_left_cover: cover property (rotate_left);

    rotate_right_assertion: assert property (rotate_right);
    rotate_right_cover: cover property (rotate_right);

endmodule : ALSU_SVA
```

Interface code snippet:

```
import ALSU_shared_pkg::*;

interface ALSU_interface (input bit clk);

// Signals
bit rst, cin, red_op_A, red_op_B, bypass_A, bypass_B, direction, serial_in;
opcode_e opcode;
bit signed [2:0] A, B;
logic [15:0] leds;
logic signed [5:0] out,out_shift_reg;
logic [15:0] leds_ref;
logic signed [5:0] out_ref;

// Modports
modport DUT (input clk ,rst, cin, red_op_A, red_op_B, bypass_A, bypass_B, direction, serial_in, opcode, A, B,
             output leds, out, out_shift_reg);
modport GM (input clk ,rst, cin, red_op_A, red_op_B, bypass_A, bypass_B, direction, serial_in, opcode, A, B,
             output leds_ref, out_ref, out_shift_reg);
endinterface : ALSU_interface
```

Top code snippet:

```
import uvm_pkg::*;
import ALSU_test_pkg::*;
import shared_pkg::*;

`include "uvm_macros.svh"

module ALSU_top ();

 bit clk;

 initial begin
    clk=0;
    forever
        #1 clk=~clk;
 end

ALSU_interface ALSU_if (clk);

ALSU ALSU_DUT (ALSU_if);

ALSU_gm ALSU_ref (ALSU_if);

bind ALSU ALSU_SVA SVA (ALSU_if);

shift_reg_if shift_regif ();

shift_reg SHIFT_REG_DUT (shift_regif);

assign shift_regif.datain =ALSU_if.out;
assign ALSU_if.out_shift_reg = shift_regif.dataout ;
assign shift_regif.serial_in = ALSU_DUT.serial_in_reg;
assign shift_regif.direction = direction_e'(ALSU_DUT.direction_reg);
assign shift_regif.mode = mode_e'(ALSU_DUT.opcode_reg);

initial begin
    uvm_config_db#(virtual ALSU_interface)::set(null, "uvm_test_top", "ALSU_IF", ALSU_if);
    uvm_config_db#(virtual shift_reg_if)::set(null, "uvm_test_top", "shift_reg_Vif", shift_regif);
    run_test("ALSU_test");
end

endmodule : ALSU_top
```

Shared package code snippet:

```
package ALSU_shared_pkg;
    typedef enum bit [2:0] {OR,XOR,ADD,MULT,SHIFT,ROTATE,INVALID_6,INVALID_7} opcode_e;
    parameter MAXPOS = 3;
    parameter MAXNEG = -4;
    parameter ZERO = 0 ;
endpackage : ALSU_shared_pkg
```

Environment code snippet:

```
package ALSU_env_pkg;
    import uvm_pkg::*;
    `include "uvm_macros.svh"
    import ALSU_agent_pkg::*;
    import ALSU_scoreboard_pkg::*;
    import ALSU_coverage_pkg::*;

    class ALSU_env extends uvm_env;
        `uvm_component_utils(ALSU_env)

        ALSU_agent agt;
        ALSU_scoreboard sb;
        ALSU_coverage cov;

        function new(string name ="ALSU_env", uvm_component parent = null);
            super.new(name,parent);
        endfunction : new

        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            agt=ALSU_agent::type_id::create("agt",this);
            sb=ALSU_scoreboard::type_id::create("sb",this);
            cov=ALSU_coverage::type_id::create("cov",this);
        endfunction : build_phase

        function void connect_phase(uvm_phase phase);
            super.connect_phase(phase);
            agt.agt_ap.connect(sb.sb_export);
            agt.agt_ap.connect(cov.cov_export);
        endfunction : connect_phase
    endclass : ALSU_env

endpackage : ALSU_env_pkg
```

Agent code snippet:

```
package ALSU_agent_pkg;
    import ALSU_shared_pkg::*;
    import ALSU_sequencer_pkg::*;
    import ALSU_sequence_item_pkg::*;
    import ALSU_config_obj_pkg::*;
    import ALSU_monitor_pkg::*;
    import ALSU_driver_pkg::*;
    import uvm_pkg::*;
    `include "uvm_macros.svh"
    class ALSU_agent extends uvm_agent;
        `uvm_component_utils(ALSU_agent)

        ALSU_sequencer sqr;
        ALSU_monitor mon;
        ALSU_driver drv;
        ALSU_config_obj ALSU_cfg;
        uvm_analysis_port #(ALSU_sequence_item) agt_ap;

        function new(string name = "ALSU_agent", uvm_component parent = null);
            super.new(name,parent);
        endfunction

        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            if (!uvm_config_db #(ALSU_config_obj)::get(this,"","CFG",ALSU_cfg)) begin
                `uvm_fatal("build_phase","Driver - unable to get configuration object");
             end
            if (ALSU_cfg.is_active == UVM_ACTIVE) begin
                drv = ALSU_driver::type_id::create("drv",this);
                sqr = ALSU_sequencer::type_id::create("sqr",this);
            end
            mon = ALSU_monitor::type_id::create("mon",this);
            agt_ap =new("agt_ap",this);
        endfunction : build_phase

        function void connect_phase(uvm_phase phase);
            super.connect_phase(phase);
            if (ALSU_cfg.is_active == UVM_ACTIVE) begin
                drv.seq_item_port.connect(sqr.seq_item_export);
                drv.ALSU_vif=ALSU_cfg.ALSU_config_vif;
            end
            mon.ALSU_vif=ALSU_cfg.ALSU_config_vif;
            mon.mon_ap.connect(agt_ap);
        endfunction : connect_phase
    endclass : ALSU_agent
endpackage : ALSU_agent_pkg
```

Driver code snippet:

```
package ALSU_driver_pkg;
    `include "uvm_macros.svh"
    import uvm_pkg::*;
    import ALSU_shared_pkg::*;
    import ALSU_sequence_item_pkg::*;
    import ALSU_main_sequence_pkg::*;
    import ALSU_reset_sequence_pkg::*;

    class ALSU_driver extends uvm_driver #(ALSU_sequence_item);
        `uvm_component_utils(ALSU_driver)

        virtual ALSU_interface ALSU_vif;
        ALSU_sequence_item stim_seq_item;

        function new(string name ="ALSU_driver", uvm_component parent = null);
            super.new(name,parent);
        endfunction : new

        task run_phase(uvm_phase phase);
            super.run_phase(phase);
            forever begin
                stim_seq_item=ALSU_sequence_item::type_id::create("stim_seq_item");
                seq_item_port.get_next_item(stim_seq_item);
                ALSU_vif.serial_in=stim_seq_item.serial_in;
                ALSU_vif.direction=stim_seq_item.direction;
                ALSU_vif.A=stim_seq_item.A;
                ALSU_vif.B=stim_seq_item.B;
                ALSU_vif.cin=stim_seq_item.cin;
                ALSU_vif.opcode=stim_seq_item.opcode;
                ALSU_vif.rst=stim_seq_item.rst;
                ALSU_vif.bypass_A=stim_seq_item.bypass_A;
                ALSU_vif.bypass_B=stim_seq_item.bypass_B;
                ALSU_vif.red_op_A=stim_seq_item.red_op_A;
                ALSU_vif.red_op_B=stim_seq_item.red_op_B;
                @(negedge ALSU_vif.clk);
                seq_item_port.item_done();
                `uvm_info("run_phase",stim_seq_item.convert2string_stimulus(), UVM_HIGH)
            end
        endtask : run_phase

    endclass : ALSU_driver
endpackage : ALSU_driver_pkg
```

Monitor code snippet:

```
package ALSU_monitor_pkg;
    `include "uvm_macros.svh"
    import uvm_pkg::*;
    import ALSU_shared_pkg::*;
    import ALSU_sequence_item_pkg::*;

    class ALSU_monitor extends uvm_monitor;
        `uvm_component_utils(ALSU_monitor)
        virtual ALSU_interface ALSU_vif;
        ALSU_sequence_item rsp_seq_item;
        uvm_analysis_port #(ALSU_sequence_item) mon_ap;

        function new(string name = "ALSU_monitor", uvm_component parent = null);
            super.new(name,parent);
        endfunction

        function void build_phase (uvm_phase phase);
            super.build_phase(phase);
            mon_ap = new("mon_ap",this);
        endfunction

        task run_phase(uvm_phase phase);
            super.run_phase (phase);
            forever begin
                rsp_seq_item = ALSU_sequence_item::type_id::create("rsp_seq_item");
                @(negedge ALSU_vif.clk);
                rsp_seq_item.serial_in=ALSU_vif.serial_in;
                rsp_seq_item.direction=ALSU_vif.direction;
                rsp_seq_item.A=ALSU_vif.A;
                rsp_seq_item.B=ALSU_vif.B;
                rsp_seq_item.rst=ALSU_vif.rst;
                rsp_seq_item.cin=ALSU_vif.cin;
                rsp_seq_item.opcode=ALSU_vif.opcode;
                rsp_seq_item.bypass_A=ALSU_vif.bypass_A;
                rsp_seq_item.bypass_B=ALSU_vif.bypass_B;
                rsp_seq_item.red_op_A=ALSU_vif.red_op_A;
                rsp_seq_item.red_op_B=ALSU_vif.red_op_B;
                mon_ap.write(rsp_seq_item);
                `uvm_info("run_phase",rsp_seq_item.convert2string_stimulus(), UVM_HIGH)
            end
        endtask
    endclass : ALSU_monitor
endpackage : ALSU_monitor_pkg
```

Sequencer code snippet:

```
package ALSU_sequencer_pkg;

import ALSU_sequence_item_pkg::*;
import ALSU_shared_pkg::*;
import uvm_pkg::*;
`include "uvm_macros.svh"

class ALSU_sequencer extends uvm_sequencer #(ALSU_sequence_item);
    `uvm_component_utils(ALSU_sequencer)

    function new(string name = "ALSU_sequencer", uvm_component parent = null);
        super.new(name,parent);
    endfunction

endclass : ALSU_sequencer

endpackage : ALSU_sequencer_pkg
```

Sequence Item code snippets:

```
package ALSU_sequence_item_pkg;
    import uvm_pkg::*;
    import ALSU_shared_pkg::*;
    `include "uvm_macros.svh"
    class ALSU_sequence_item extends uvm_sequence_item;
        `uvm_object_utils(ALSU_sequence_item)

        function new(string name = "ALSU_sequence_item");
            super.new(name);
        endfunction

        // Signals
        rand bit rst, cin, red_op_A, red_op_B, bypass_A, bypass_B, direction, serial_in;
        rand opcode_e opcode;
        rand bit signed [2:0] A, B;
        rand logic [15:0] leds;
        rand logic signed [5:0] out;
        logic [15:0] leds_ref;
        logic signed [5:0] out_ref;
        rand opcode_e opcode_array[5:0];
        bit [2:0] EXTREEMS [2:0] = '{MAXNEG,ZERO,MAXPOS};
        bit [2:0] INTERNALS [] = {3'b101, 3'b110, 3'b111, 3'b000, 3'b011} ;//!=EXTREEMS
        bit [2:0] WALKING_ONES [2:0] = '{3'b001,3'b010,3'b100};
        bit [2:0] SLEEPING_ONES [] = {3'b000,3'b011,3'b101,3'b110,3'b111};//!= WALKING_ONES

        // Constraint blocks
        constraint rst_constraint {
            rst dist {0:/95,1:/5};//Reset to be asserted with a low probability , 95% off & 5% on
        }
        constraint opcode_e_constraint {
            opcode dist {[OR:ROTATE]:/90,[INVALID_6:INVALID_7]:/10};//Invalid cases should occur less frequent(10%) than the valid cases(90%)
        }
        constraint A_B_constraints {
            if (opcode == ADD || opcode == MULT )
            {   //A & B to take the values (MAXPOS, ZERO and MAXNEG) more often than the other values
                A dist { EXTREEMS:/75 ,INTERNALS:/25};
                B dist { EXTREEMS:/75 ,INTERNALS:/25};
            }
            else if ((opcode == OR || opcode == XOR))
            {
                if (red_op_A){///(red_op_A_package==1)
                        //A most of the time to have one bit high in its 3 bits while constraining the B to be low
                        A dist  { WALKING_ONES:/80 ,SLEEPING_ONES:/20 };
                        B == 0;
                    }
                else if(red_op_B && !red_op_A ){///(red_op_B_package==1 && red_op_A_package==0)
                        //B most of the time to have one bit high in its 3 bits while constraining the A to be low
                        B dist { WALKING_ONES:/80 ,SLEEPING_ONES:/20 };
                        A == 0;
                    }
            }
            else {
                //Do not constraint the inputs A or B when the operation is shift or rotate
                A inside {[MAXNEG:MAXPOS]};
```

```
            else {
                //Do not constraint the inputs A or B when the operation is shift or rotate
                A inside {[MAXNEG:MAXPOS]};
                B inside {[MAXNEG:MAXPOS]};
                }
        }
        constraint bypass_constraint {
            //bypass_A and bypass_B should be disabled most of the time
            bypass_A dist {0:/90,1:/10};//90% off & 10% on
            bypass_B dist {0:/90,1:/10};//90% off & 10% on
        }
        constraint unique_opcodes {
            foreach (opcode_array[i]) {
            // Ensure elements that are not manually overridden are unique
                opcode_array[i] inside {[OR:ROTATE]};
                foreach (opcode_array[j]) {
                    if (i != j) { // Exclude overridden elements
                        opcode_array[i] != opcode_array[j];
                    }
                }
            }
        }

        // Convert to string functions
        function string convert2string();
            return $sformatf(
            "%s rst = %0b, cin = %0b, red_op_A = %0b, red_op_B = %0b, bypass_A = %0b, bypass_B = %0b, direction = %0b, serial_in = %0b, opcode = %s, A = %0d, B = %0d, leds = 0x%0h, out = %0d",
            super.convert2string(), rst, cin, red_op_A, red_op_B, bypass_A, bypass_B, direction, serial_in, opcode.name(), A, B, leds, out);
            endfunction : convert2string


        function string convert2string_stimulus();
            return $sformatf(
            "rst = %0b, cin = %0b, red_op_A = %0b, red_op_B = %0b, bypass_A = %0b, bypass_B = %0b, direction = %0b, serial_in = %0b, opcode = %s, A = %0d, B = %0d, leds = 0x%0h, out = %0d",
            rst, cin, red_op_A, red_op_B, bypass_A, bypass_B, direction, serial_in, opcode.name(), A, B, leds, out);
            endfunction : convert2string_stimulus

    endclass : ALSU_sequence_item
endpackage : ALSU_sequence_item_pkg
```

Main Sequence code snippet:

```
package ALSU_main_sequence_pkg;
    import ALSU_shared_pkg::*;
    import uvm_pkg::*;
    import ALSU_sequence_item_pkg::*;
    `include "uvm_macros.svh"

    class ALSU_main_sequence extends uvm_sequence #(ALSU_sequence_item);
        `uvm_object_utils(ALSU_main_sequence)

        ALSU_sequence_item seq_item;

        function new(string name = "ALSU_main_sequence");
            super.new(name);
        endfunction

        task body;
            repeat(10_000) begin
                seq_item=ALSU_sequence_item::type_id::create("seq_item");
                start_item(seq_item);
                assert(seq_item.randomize());
                finish_item(seq_item);
            end
        endtask : body

    endclass : ALSU_main_sequence

endpackage : ALSU_main_sequence_pkg
```

Reset Sequence code snippet:

```
package ALSU_reset_sequence_pkg;
    import ALSU_shared_pkg::*;
    import uvm_pkg::*;
    import ALSU_sequence_item_pkg::*;
    `include "uvm_macros.svh"
    class ALSU_reset_sequence extends uvm_sequence #(ALSU_sequence_item);
        `uvm_object_utils(ALSU_reset_sequence)

        ALSU_sequence_item seq_item;

        function new(string name = "ALSU_reset_sequence");
            super.new(name);
        endfunction

        task body;
            seq_item=ALSU_sequence_item::type_id::create("seq_item");
            start_item(seq_item);
            seq_item.rst=1;
            seq_item.serial_in=0;
            seq_item.direction=0;
            seq_item.A=0;
            seq_item.B=0;
            seq_item.cin=0;
            seq_item.opcode=opcode_e'(0);
            seq_item.bypass_A=0;
            seq_item.bypass_B=0;
            seq_item.red_op_A=0;
            seq_item.red_op_B=0;
            finish_item(seq_item);
        endtask : body
    endclass : ALSU_reset_sequence
endpackage : ALSU_reset_sequence_pkg
```

Valid Sequence code snippet:

```
package ALSU_valid_sequence_pkg;
    import ALSU_shared_pkg::*;
    import uvm_pkg::*;
    import ALSU_sequence_item_pkg::*;
    `include "uvm_macros.svh"
    class ALSU_valid_sequence extends uvm_sequence #(ALSU_sequence_item);
        `uvm_object_utils(ALSU_valid_sequence)

        int i =-1;

        ALSU_sequence_item seq_item;

        function new(string name = "ALSU_valid_sequence");
            super.new(name);
        endfunction

        task body;
            repeat (10) begin
            i++;
            seq_item=ALSU_sequence_item::type_id::create("seq_item");
            start_item(seq_item);
            assert(seq_item.randomize() with {opcode == i;});
            finish_item(seq_item);
            end
        endtask : body
    endclass : ALSU_valid_sequence
endpackage : ALSU_valid_sequence_pkg
```

Scoreboard code snippet:

```
package ALSU_scoreboard_pkg;
    import uvm_pkg::*;
    import ALSU_shared_pkg::*;
    import ALSU_sequence_item_pkg::*;
    `include "uvm_macros.svh"
    class ALSU_scoreboard extends uvm_scoreboard;
        `uvm_component_utils(ALSU_scoreboard)
        uvm_analysis_export #(ALSU_sequence_item) sb_export;
        uvm_tlm_analysis_fifo #(ALSU_sequence_item) sb_fifo;
        ALSU_sequence_item seq_item_sb;

        integer correct_counter=0;
        integer error_counter=0;

        function new(string name = "ALSU_scoreboard", uvm_component parent = null);
            super.new(name,parent);
        endfunction

        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            sb_export =new("sb_export",this);
            sb_fifo =new("sb_fifo",this);
        endfunction : build_phase

        function void connect_phase(uvm_phase phase);
            super.connect_phase(phase);
            sb_export.connect(sb_fifo.analysis_export);
        endfunction : connect_phase

        task run_phase(uvm_phase phase);
            super.run_phase(phase);
            forever begin
                sb_fifo.get(seq_item_sb);
                if (seq_item_sb.out_ref!= seq_item_sb.out) begin
                    `uvm_error("run_phase", $sformatf("There is an error!!, Transacton received by the DUT is: %s whilethe refernce output is: 0b%0b",
                     seq_item_sb.convert2string(), seq_item_sb.out_ref));
                    error_counter++;
                end
                else begin
                    `uvm_info("run_phase", $sformatf("Correct Transacton received, Output is: %s",
                                seq_item_sb.convert2string()),UVM_HIGH)
                    correct_counter++;
                end
            end
        endtask : run_phase

        function void report_phase(uvm_phase phase);
            super.report_phase(phase);
            `uvm_info("report_phase", $sformatf("Total coorect counts is: 0d%0d",correct_counter),UVM_LOW)
            `uvm_info("report_phase", $sformatf("Total error counts is: 0d%0d",error_counter),UVM_LOW)
        endfunction : report_phase
    endclass : ALSU_scoreboard
endpackage : ALSU_scoreboard_pkg
```

Coverage code snippet:

```
package ALSU_coverage_pkg;
    import uvm_pkg::*;
    import ALSU_shared_pkg::*;
    import ALSU_sequence_item_pkg::*;
    `include "uvm_macros.svh"
    class ALSU_coverage extends uvm_component;
        `uvm_component_utils(ALSU_coverage)
        uvm_analysis_export #(ALSU_sequence_item) cov_export;
        uvm_tlm_analysis_fifo #(ALSU_sequence_item) cov_fifo;
        ALSU_sequence_item seq_item_cov;

        // Covergroups
        bit [2:0] WALKING_ONES [2:0] = '{3'b001,3'b010,3'b100};
        covergroup cg;
            option.per_instance = 1;  // Enable per-instance options
            A_cp: coverpoint seq_item_cov.A {
                bins A_data_0 = {0};
                bins A_data_max = {MAXPOS};
                bins A_data_min = {MAXNEG};
                bins A_data_default = default ;
            }
            A_cp_walking_ones: coverpoint seq_item_cov.A iff (seq_item_cov.red_op_A) {
                bins A_data_walkingones[] = WALKING_ONES;
            }
            B_cp: coverpoint seq_item_cov.B {
                bins B_data_0 = {0};
                bins B_data_max = {MAXPOS};
                bins B_data_min = {MAXNEG};
                bins B_data_default = default ;
            }
            B_cp_walking_ones: coverpoint seq_item_cov.B iff ((seq_item_cov.red_op_B) && !(seq_item_cov.red_op_A)) {
                bins B_data_walkingones[] = WALKING_ONES;
            }
            ALU_cp: coverpoint seq_item_cov.opcode {
                bins Bins_shift[] = {SHIFT,ROTATE};
                bins Bins_arith[] = {ADD , MULT};
                bins Bins_bitwise[] ={OR,XOR};
                illegal_bins Bins_invalid ={INVALID_6,INVALID_7};
                bins Bins_trans ={0=>1=>2=>3=>4=>5};
            }
            opcode_cp: coverpoint seq_item_cov.opcode {option.weight = 0;}
            cin_cp: coverpoint seq_item_cov.cin {option.weight = 0;}
            serialin_cp: coverpoint seq_item_cov.serial_in {option.weight = 0;}
            direction_cp: coverpoint seq_item_cov.direction {option.weight = 0;}
            red_op_A_cp: coverpoint seq_item_cov.red_op_A {option.weight = 0;}
            red_op_B_cp: coverpoint seq_item_cov.red_op_B {option.weight = 0;}
            cross ALU_cp, A_cp, B_cp {
            // When the ALU is addition or multiplication, A and B should have taken all permutations of maxpos, maxneg and zero.
            bins ADD_MULT_EXTREEMS = binsof(ALU_cp.Bins_arith) && (binsof(A_cp) && binsof(B_cp));//Default bin won't be considered by default
            option.cross_auto_bin_max=0;
            }
            // When the ALU is addition, c_in should have taken 0 or 1
            cross opcode_cp, cin_cp {
                bins ADD_CIN0 = binsof(opcode_cp) intersect {ADD} && binsof(cin_cp) intersect {0};
                bins ADD_CIN1 = binsof(opcode_cp) intersect {ADD} && binsof(cin_cp) intersect {1};
```

```
            // When the ALU is addition, c_in should have taken 0 or 1
            cross opcode_cp, cin_cp {
                bins ADD_CIN0 = binsof(opcode_cp) intersect {ADD} && binsof(cin_cp) intersect {0};
                bins ADD_CIN1 = binsof(opcode_cp) intersect {ADD} && binsof(cin_cp) intersect {1};
                option.cross_auto_bin_max=0;
            }
            // When the ALU is shifting, then shift_in must take 0 or 1
            cross opcode_cp, serialin_cp {
                bins SHIFT_SERIALIN0 = binsof(opcode_cp) intersect {SHIFT} && binsof(serialin_cp) intersect {0};
                bins SHIFT_SERIALIN1 = binsof(opcode_cp) intersect {SHIFT} && binsof(serialin_cp) intersect {1};
                option.cross_auto_bin_max=0;
            }
            // When the ALU is shifting or rotating, then direction must take 0 or 1
            cross ALU_cp, direction_cp {
                bins SHIFT_ROTATE_DIRECTION0 = binsof(ALU_cp.Bins_shift) && binsof(direction_cp) intersect {0};
                bins SHIFT_ROTATE_DIRECTION1 = binsof(ALU_cp.Bins_shift) && binsof(direction_cp) intersect {1};
                option.cross_auto_bin_max=0;
            }
            // When the ALU is OR or XOR and red_op_A is asserted, then A took all walking one patterns (001, 010, and 100) while B is taking the value 0
            cross A_cp_walking_ones, B_cp iff((seq_item_cov.opcode==OR || seq_item_cov.opcode==XOR ) && seq_item_cov.red_op_A) {
                bins OR_XOR_REDA = binsof(A_cp_walking_ones.A_data_walkingones) && binsof(B_cp.B_data_0);
                option.cross_auto_bin_max=0;
            }
            //When the ALU is OR or XOR and red_op_B is asserted, then B took all walking one patterns (001, 010, and 100) while A is taking the value 0
            cross B_cp_walking_ones, A_cp iff((seq_item_cov.opcode==OR || seq_item_cov.opcode==XOR ) && seq_item_cov.red_op_B) {
                bins OR_XOR_REDB = binsof(B_cp_walking_ones.B_data_walkingones) && binsof(A_cp.A_data_0);
                option.cross_auto_bin_max=0;
            }
            //Covering the invalid case: reduction operation is activated while the opcode is not OR or XOR
            cross red_op_A_cp, red_op_B_cp, opcode_cp {
                bins INVALID_REDUCTION = (binsof(red_op_A_cp) intersect {1} || binsof(red_op_B_cp) intersect {1}) &&(binsof(opcode_cp) intersect{[ADD:INVALID_7]});
                option.cross_auto_bin_max=0;
            }
        endgroup

        function new(string name = "ALSU_coverage", uvm_component parent = null);
            super.new(name,parent);
            cg=new();
        endfunction

        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            cov_export =new("cov_export",this);
            cov_fifo =new("cov_fifo",this);
        endfunction : build_phase

        function void connect_phase(uvm_phase phase);
            super.connect_phase(phase);
            cov_export.connect(cov_fifo.analysis_export);
        endfunction : connect_phase

        task run_phase(uvm_phase phase);
            super.run_phase(phase);
            forever begin
                cov_fifo.get(seq_item_cov);
```

```
            task run_phase(uvm_phase phase);
                super.run_phase(phase);
                forever begin
                    cov_fifo.get(seq_item_cov);
                    cg.sample();
                end
            endtask : run_phase
        endclass : ALSU_coverage
    endpackage : ALSU_coverage_pkg
```

Test code snippet:

```
package ALSU_test_pkg;
    `include "uvm_macros.svh"
    import uvm_pkg::*;
    import ALSU_env_pkg::*;
    import package_shift_reg_env_pkg::*;
    import ALSU_config_obj_pkg::*;
    import shift_reg_config_pkg::*;
    import ALSU_agent_pkg::*;
    import ALSU_main_sequence_pkg::*;
    import ALSU_reset_sequence_pkg::*;
    import ALSU_valid_sequence_pkg::*;
    import shigt_reg_main_sequence_pkg::*;
    import shift_reg_agent_pkg::*;
    import ALSU_sequence_item_pkg::*; // Correct package import

    class ALSU_test extends uvm_test ;
        `uvm_component_utils(ALSU_test)

        ALSU_config_obj ALSU_config_obj_test;
        shift_reg_config shift_reg_cfg;
        ALSU_env env;
        shift_reg_env env2;
        virtual ALSU_interface ALSU_test_vif;
        ALSU_reset_sequence reset_seq;
        ALSU_main_sequence main_seq;
        ALSU_valid_sequence valid_seq;

        function new(string name ="ALSU_test", uvm_component parent = null);
            super.new(name,parent);
        endfunction : new

        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            env = ALSU_env::type_id::create("env",this);
            env2 = shift_reg_env::type_id::create("env2",this);
            ALSU_config_obj_test = ALSU_config_obj::type_id::create("ALSU_config_obj_test");
            shift_reg_cfg = shift_reg_config::type_id::create("shift_reg_cfg");
            reset_seq = ALSU_reset_sequence::type_id::create("reset_seq");
            main_seq = ALSU_main_sequence::type_id::create("main_seq");
            valid_seq = ALSU_valid_sequence::type_id::create("valid_seq");

            if (!uvm_config_db#(virtual ALSU_interface)::get(this, "", "ALSU_IF", ALSU_config_obj_test.ALSU_config_vif)) begin
                `uvm_fatal("build_phase","Test - unable to get the virtual interface");
             end

             if (!uvm_config_db#(virtual shift_reg_if)::get(this,"","shift_reg_Vif",shift_reg_cfg.shift_reg_Vif)) begin
                `uvm_fatal("build_phase","Test - unable to get the virtual interface");
             end

            ALSU_config_obj_test.is_active=UVM_ACTIVE;
            shift_reg_cfg.is_active=UVM_PASSIVE;

            uvm_config_db #(shift_reg_config)::set(this,"*","CFG",shift_reg_cfg);
            uvm_config_db#(ALSU_config_obj)::set(this, "*", "CFG", ALSU_config_obj_test);
```

```
            // Correct class references
            set_type_override_by_type(ALSU_sequence_item::get_type(), ALSU_sequence_item_valid_invalid::get_type());

        endfunction : build_phase

        task run_phase(uvm_phase phase);
            super.run_phase(phase);
            phase.raise_objection(this);

            `uvm_info("run_phase","Reset asserted", UVM_LOW)
            reset_seq.start(env.agt.sqr);

            `uvm_info("run_phase","Stimulus generation started", UVM_LOW)
            main_seq.start(env.agt.sqr);

            `uvm_info("run_phase","Valid opcodes sequence started", UVM_LOW)
            valid_seq.start(env.agt.sqr);

            phase.drop_objection(this);
        endtask

    endclass : ALSU_test
endpackage : ALSU_test_pkg
```

Configuration object code snippet:

```
package ALSU_config_obj_pkg;
    `include "uvm_macros.svh"
    import uvm_pkg::*;

    class ALSU_config_obj extends uvm_object;
        `uvm_object_utils(ALSU_config_obj)

        virtual ALSU_interface ALSU_config_vif;

        uvm_active_passive_enum is_active;

        function new(string name ="ALSU_config_obj");
            super.new(name);
        endfunction : new

    endclass : ALSU_config_obj
endpackage : ALSU_config_obj_pkg
```

Shift Register:

Design code snippets:

```
module shift_reg (shift_reg_if.SHIFT_REG_DUT_MP shift_regif);

always @(*) begin
   if (shift_regif.mode) // rotate
      if (shift_regif.direction) // left
         shift_regif.dataout <= {shift_regif.datain[4:0], shift_regif.datain[5]};
      else// Right
         shift_regif.dataout <= {shift_regif.datain[0], shift_regif.datain[5:1]};
   else  // shift
      if (shift_regif.direction) // left
         shift_regif.dataout <= {shift_regif.datain[4:0], shift_regif.serial_in};
      else // Right
         shift_regif.dataout <= {shift_regif.serial_in, shift_regif.datain[5:1]};
end
endmodule
```

Interface code snippet:

```
import shared_pkg::*;

interface shift_reg_if ();

  logic serial_in;
  direction_e direction;
  mode_e mode;
  logic [5:0] datain, dataout;

  // Modport
  modport SHIFT_REG_DUT_MP (input serial_in, direction, mode,datain,output dataout);
endinterface : shift_reg_if
```

Shared package code snippet:

```
package ALSU_shared_pkg;
    typedef enum bit [2:0] {OR,XOR,ADD,MULT,SHIFT,ROTATE,INVALID_6,INVALID_7} opcode_e;
    parameter MAXPOS = 3;
    parameter MAXNEG = -4;
    parameter ZERO = 0 ;
endpackage : ALSU_shared_pkg
```

Environment code snippet:

```
package package_shift_reg_env_pkg;
    import shift_reg_agent_pkg::*;
    import shift_reg_scoreboard_pkg::*;
    import shift_reg_coverage_pkg::*;


    import uvm_pkg::*;
    `include "uvm_macros.svh"

    class shift_reg_env extends uvm_env ;
        `uvm_component_utils(shift_reg_env)
        shift_reg_agent agt;
        shift_reg_scoreboard sb;
        shift_reg_coverage cov;

        function new(string name = "shift_reg_env", uvm_component parent = null);
            super.new(name,parent);
        endfunction

        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            agt=shift_reg_agent::type_id::create("agt",this);
            sb=shift_reg_scoreboard::type_id::create("sb",this);
            cov=shift_reg_coverage::type_id::create("cov",this);
        endfunction

        function void connect_phase(uvm_phase phase);
            super.connect_phase(phase);
            agt.agt_ap.connect(sb.sb_export);
            agt.agt_ap.connect(cov.cov_export);
        endfunction : connect_phase

    endclass
endpackage
```

Agent code snippet:

```
package shift_reg_agent_pkg;
    import shift_reg_driver_pkg::*;
    import MySequencer_pkg::*;
    import shift_reg_config_pkg::*;
    import shift_reg_monitor_pkg::*;
    import shigt_reg_sequence_item_pkg::*;

    import uvm_pkg::*;
    `include "uvm_macros.svh"
    class shift_reg_agent extends uvm_agent ;
        `uvm_component_utils(shift_reg_agent)
        MySequencer sqr;
        shift_reg_driver drv;
        shift_reg_monitor mon;
        shift_reg_config shift_reg_cfg;
        uvm_analysis_port #(shigt_reg_sequence_item) agt_ap;

        function new(string name = "shift_reg_agent", uvm_component parent = null);
            super.new(name,parent);
        endfunction

        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            if (!uvm_config_db #(shift_reg_config)::get(this,"","CFG",shift_reg_cfg)) begin
                `uvm_fatal("build_phase","Driver - unable to get configuration object");
             end
            if (shift_reg_cfg.is_active == UVM_ACTIVE) begin
                drv = shift_reg_driver::type_id::create("drv",this);
                sqr = MySequencer::type_id::create("sqr",this);
            end
            mon = shift_reg_monitor::type_id::create("mon",this);
            agt_ap =new("agt_ap",this);

        endfunction : build_phase

        function void connect_phase(uvm_phase phase);
            super.connect_phase(phase);
            if (shift_reg_cfg.is_active == UVM_ACTIVE) begin
                drv.seq_item_port.connect(sqr.seq_item_export);
                drv.shift_reg_Vif=shift_reg_cfg.shift_reg_Vif;
            end
            mon.shift_reg_Vif=shift_reg_cfg.shift_reg_Vif;
            mon.mon_ap.connect(agt_ap);
        endfunction : connect_phase
    endclass : shift_reg_agent
endpackage : shift_reg_agent_pkg
```

Driver code snippet:

```
package shift_reg_driver_pkg;
    import shift_reg_config_pkg::*;
    import shigt_reg_sequence_item_pkg::*;
    import shigt_reg_main_sequence_pkg::*;

    import uvm_pkg::*;
    `include "uvm_macros.svh"
    class shift_reg_driver extends uvm_driver #(shigt_reg_sequence_item);
        `uvm_component_utils(shift_reg_driver);
        virtual shift_reg_if shift_reg_Vif;
        shigt_reg_sequence_item stim_seq_item;

        function new(string name = "shift_reg_driver", uvm_component parent = null);
            super.new(name,parent);
        endfunction

        task run_phase(uvm_phase phase);
            super.run_phase (phase);
            forever begin
                stim_seq_item=shigt_reg_sequence_item::type_id::create("stim_seq_item");
                seq_item_port.get_next_item(stim_seq_item);
                shift_reg_Vif.direction=stim_seq_item.direction;
                shift_reg_Vif.serial_in=stim_seq_item.serial_in;
                shift_reg_Vif.datain=stim_seq_item.datain;
                shift_reg_Vif.mode=stim_seq_item.mode;
                #2;
                seq_item_port.item_done();
                `uvm_info("run_phase",stim_seq_item.convert2string_stimulus(), UVM_HIGH)
            end
        endtask
    endclass
endpackage
```

Monitor code snippet:

```
package shift_reg_monitor_pkg;
    import shift_reg_config_pkg::*;
    import shigt_reg_sequence_item_pkg::*;

    import uvm_pkg::*;
    `include "uvm_macros.svh"
    class shift_reg_monitor extends uvm_monitor ;
        `uvm_component_utils(shift_reg_monitor);
        virtual shift_reg_if shift_reg_Vif;
        shigt_reg_sequence_item rsp_seq_item;
        uvm_analysis_port #(shigt_reg_sequence_item) mon_ap;

        function new(string name = "shift_reg_monitor", uvm_component parent = null);
            super.new(name,parent);
        endfunction

        function void build_phase (uvm_phase phase);
            super.build_phase(phase);
            mon_ap = new("mon_ap",this);
        endfunction

        task run_phase(uvm_phase phase);
            super.run_phase (phase);
            forever begin
                rsp_seq_item = shigt_reg_sequence_item::type_id::create("rsp_seq_item");
                #2;
                rsp_seq_item.direction=shift_reg_Vif.direction;
                rsp_seq_item.serial_in=shift_reg_Vif.serial_in;
                rsp_seq_item.datain=shift_reg_Vif.datain;
                rsp_seq_item.mode=shift_reg_Vif.mode;
                mon_ap.write(rsp_seq_item);
                `uvm_info("run_phase",rsp_seq_item.convert2string_stimulus(), UVM_HIGH)
            end
        endtask

    endclass
endpackage
```

Sequencer code snippet:

```
package MySequencer_pkg;
import shigt_reg_sequence_item_pkg::*;

import shared_pkg::*;
import uvm_pkg::*;
`include "uvm_macros.svh"
class MySequencer extends uvm_sequencer #(shigt_reg_sequence_item);
    `uvm_component_utils(MySequencer)

    function new(string name = "MySequencer", uvm_component parent = null);
        super.new(name,parent);
    endfunction

endclass : MySequencer
endpackage : MySequencer_pkg
```

Sequence Item code snippet:

```
package shigt_reg_sequence_item_pkg;
    import shared_pkg::*;
    import uvm_pkg::*;
    `include "uvm_macros.svh"
    class shigt_reg_sequence_item extends uvm_sequence_item ;
        `uvm_object_utils(shigt_reg_sequence_item)

        rand bit reset;
        rand bit serial_in;
        rand mode_e mode;
        rand direction_e direction;
        rand logic [5:0] datain;
        logic [5:0] dataout;

        function new(string name = "shigt_reg_sequence_item");
            super.new(name);
        endfunction

        constraint reset_constraint {
            reset dist {0:/98, 1:/2};
        }
        constraint datain_constraint {
            datain dist {[63:0]:=1};
        }

        function string convert2string();
            return $sformatf("%s reset = 0b%0b, serial_in = 0b%0b, mode = %s, direction = %s, data_in = 0b%0b",
             super.convert2string(), reset, serial_in, mode, direction, datain);
        endfunction : convert2string

        function string convert2string_stimulus();
            return $sformatf("reset = 0b%0b, serial_in = 0b%0b, mode = %s, direction = %s, data_in = 0b%0b"
                             , reset, serial_in, mode, direction, datain);
        endfunction : convert2string_stimulus
    endclass : shigt_reg_sequence_item


endpackage : shigt_reg_sequence_item_pkg
```

Main Sequence code snippet:

```
package shigt_reg_main_sequence_pkg;
    import shared_pkg::*;
    import uvm_pkg::*;
    import shigt_reg_sequence_item_pkg::*;
    `include "uvm_macros.svh"
    class shigt_reg_main_sequence extends uvm_sequence #(shigt_reg_sequence_item) ;
        `uvm_object_utils(shigt_reg_main_sequence)

        shigt_reg_sequence_item seq_item;

        function new(string name = "shigt_reg_main_sequence");
            super.new(name);
        endfunction

        task body;
            repeat(10_000) begin
                seq_item=shigt_reg_sequence_item::type_id::create("seq_item");
                start_item(seq_item);
                assert(seq_item.randomize());
                finish_item(seq_item);
            end
        endtask : body
    endclass : shigt_reg_main_sequence


endpackage : shigt_reg_main_sequence_pkg
```

Reset Sequence code snippet:

```
package shigt_reg_reset_sequence_pkg;
    import shared_pkg::*;
    import uvm_pkg::*;
    import shigt_reg_sequence_item_pkg::*;

    `include "uvm_macros.svh"
    class shigt_reg_reset_sequence extends uvm_sequence #(shigt_reg_sequence_item) ;
        `uvm_object_utils(shigt_reg_reset_sequence)

        shigt_reg_sequence_item seq_item;

        function new(string name = "shigt_reg_reset_sequence");
            super.new(name);
        endfunction

        task body;
            seq_item = shigt_reg_sequence_item::type_id::create("seq_item");
            start_item(seq_item);
            seq_item.reset=1;
            seq_item.serial_in=0;
            seq_item.datain=0;
            seq_item.direction= direction_e'(0) ;
            seq_item.mode=mode_e'(0);
            finish_item(seq_item);
        endtask : body
    endclass : shigt_reg_reset_sequence


endpackage : shigt_reg_reset_sequence_pkg
```

Scoreboard code snippet:

```
package shift_reg_scoreboard_pkg;
    import uvm_pkg::*;
    import shared_pkg::*;

    import shigt_reg_sequence_item_pkg::*;
    `include "uvm_macros.svh"
    class shift_reg_scoreboard extends uvm_scoreboard;
        `uvm_component_utils(shift_reg_scoreboard)
        uvm_analysis_export #(shigt_reg_sequence_item) sb_export;
        uvm_tlm_analysis_fifo #(shigt_reg_sequence_item) sb_fifo;
        shigt_reg_sequence_item seq_item_sb;
        logic [5:0] dataout_ref;

        integer correct_counter=0;
        integer error_counter=0;

        function new(string name = "shift_reg_scoreboard", uvm_component parent = null);
            super.new(name,parent);
        endfunction

        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            sb_export =new("sb_export",this);
            sb_fifo =new("sb_fifo",this);
        endfunction : build_phase

        function void connect_phase(uvm_phase phase);
            super.connect_phase(phase);
            sb_export.connect(sb_fifo.analysis_export);
        endfunction : connect_phase

        task run_phase(uvm_phase phase);
            super.run_phase(phase);
            forever begin
                sb_fifo.get(seq_item_sb);
                ref_model(seq_item_sb);
                if (seq_item_sb.dataout != dataout_ref) begin
                    `uvm_error("run_phase", $sformatf("There is an error!!, Transacton received by the DUT is: %s whilethe refernce output is: 0b%0b",
                     seq_item_sb.convert2string(), dataout_ref));
                    error_counter++;
                end
                else begin
                    `uvm_info("run_phase", $sformatf("Correct Transacton received, Output is: %s",
                                seq_item_sb.convert2string()),UVM_HIGH)
                    correct_counter++;
                end
            end
        endtask : run_phase

        task ref_model(shigt_reg_sequence_item seq_item_chk);
            if (seq_item_chk.reset) begin
                dataout_ref=0;
            end
            else begin
                if (seq_item_chk.mode) // rotate
```

```
        task ref_model(shigt_reg_sequence_item seq_item_chk);
            if (seq_item_chk.reset) begin
                dataout_ref=0;
            end
            else begin
                if (seq_item_chk.mode) // rotate
                    if (seq_item_chk.direction) // left
                        dataout_ref = {seq_item_chk.datain[4:0], seq_item_chk.datain[5]};
                    else
                        dataout_ref = {seq_item_chk.datain[0], seq_item_chk.datain[5:1]};
                else // shift
                    if (seq_item_chk.direction) // left
                        dataout_ref = {seq_item_chk.datain[4:0], seq_item_chk.serial_in};
                    else
                        dataout_ref = {seq_item_chk.serial_in, seq_item_chk.datain[5:1]};
            end
        endtask : ref_model

        function void report_phase(uvm_phase phase);
            super.report_phase(phase);
            `uvm_info("report_phase", $sformatf("Total coorect counts is: 0d%0d",correct_counter),UVM_LOW)
            `uvm_info("report_phase", $sformatf("Total error counts is: 0d%0d",error_counter),UVM_LOW)
        endfunction : report_phase
    endclass : shift_reg_scoreboard
endpackage : shift_reg_scoreboard_pkg
```

Coverage code snippet:

```
package shift_reg_coverage_pkg;
    import uvm_pkg::*;
    import shared_pkg::*;
    import shigt_reg_sequence_item_pkg::*;
    `include "uvm_macros.svh"
    class shift_reg_coverage extends uvm_component;
        `uvm_component_utils(shift_reg_coverage)
        uvm_analysis_export #(shigt_reg_sequence_item) cov_export;
        uvm_tlm_analysis_fifo #(shigt_reg_sequence_item) cov_fifo;
        shigt_reg_sequence_item seq_item_cov;

        // Covergroups
        covergroup cg;
            data_in_cp: coverpoint seq_item_cov.datain;
            direction_cp: coverpoint seq_item_cov.direction;
            mode_cp: coverpoint seq_item_cov.mode;
            serial_in_cp: coverpoint seq_item_cov.serial_in;
        endgroup : cg

        function new(string name = "shift_reg_scoreboard", uvm_component parent = null);
            super.new(name,parent);
            cg=new();
        endfunction

        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            cov_export =new("cov_export",this);
            cov_fifo =new("cov_fifo",this);
        endfunction : build_phase

        function void connect_phase(uvm_phase phase);
            super.connect_phase(phase);
            cov_export.connect(cov_fifo.analysis_export);
        endfunction : connect_phase

        task run_phase(uvm_phase phase);
            super.run_phase(phase);
            forever begin
                cov_fifo.get(seq_item_cov);
                cg.sample();
            end
        endtask : run_phase


    endclass : shift_reg_coverage
endpackage : shift_reg_coverage_pkg
```

Configuration object code snippet:

```
package shift_reg_config_pkg;
    import uvm_pkg::*;
    `include "uvm_macros.svh"
    class shift_reg_config extends uvm_object;
        `uvm_object_utils(shift_reg_config);

        virtual shift_reg_if shift_reg_Vif;

        uvm_active_passive_enum is_active;

        function new(string name = "shift_reg_config");
            super.new(name);
        endfunction
    endclass
endpackage

```

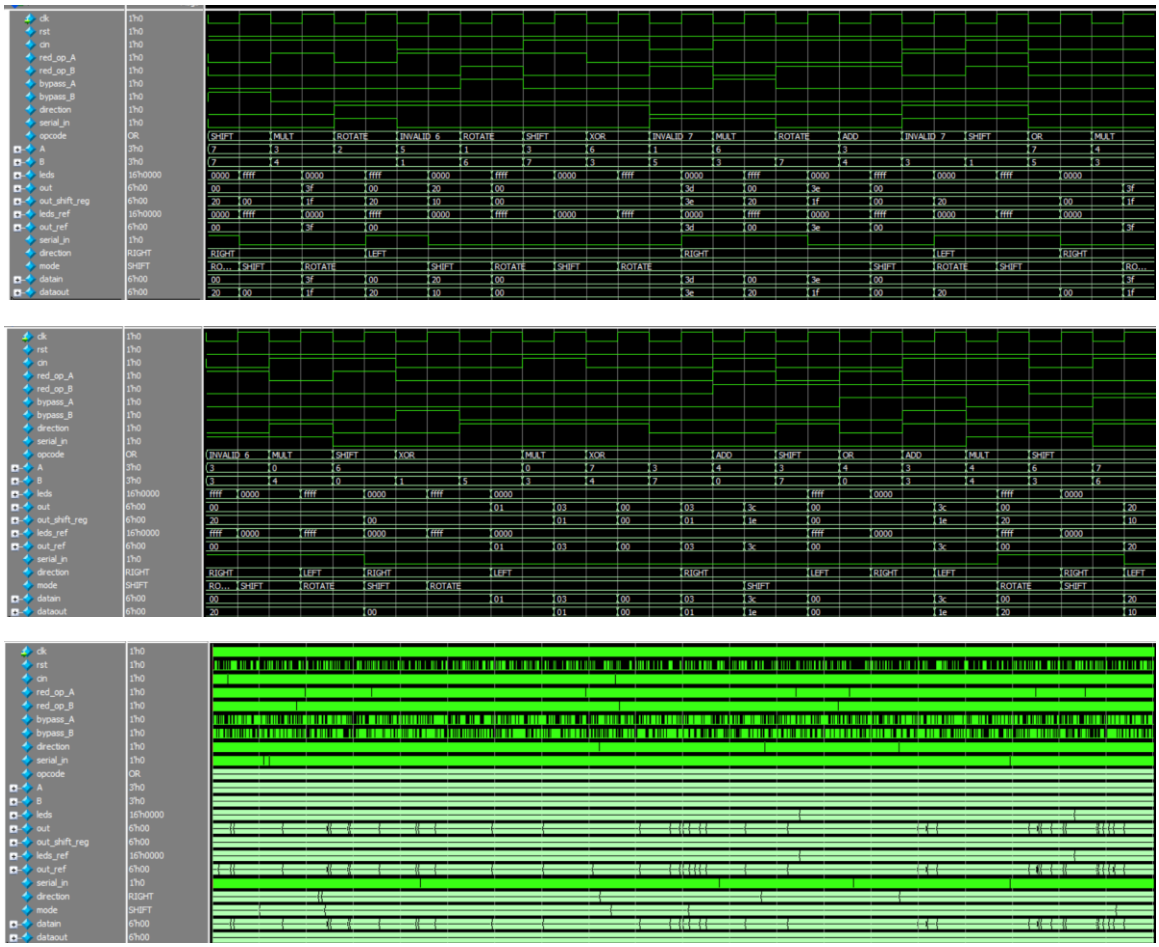
DO File:

```
vlib work
vlog -f ALSU.list  +cover -covercells
vsim -voptargs=+acc work.ALSU_top -cover -sv_seed random -l sim.ALSU_log
add wave /ALSU_top/ALSU_if/*
coverage save ALSU.ucdb -onexit
run -all
quit -sim
vcover report ALSU.ucdb -details -annotate -all -output coverage_ALSU_rpt.txt
```

ALSU & Shift Register list of files:

```
ALSU.sv
ALSU_SVA.sv
ALSU_config_obj_pkg.sv
ALSU_shared_pkg.sv
ALSU_interface.sv
ALSU_valid_sequence_pkg.sv
ALSU_main_sequence_pkg.sv
ALSU_reset_sequence_pkg.sv
ALSU_sequence_item_pkg.sv
ALSU_sequencer_pkg.sv
ALSU_env_pkg.sv
ALSU_test_pkg.sv
ALSU_driver_pkg.sv
ALSU_agent_pkg.sv
ALSU_scoreboard_pkg.sv
ALSU_monitor_pkg.sv
ALSU_coverage_pkg.sv
shift_reg.sv
shift_reg_config_pkg.sv
shared_pkg.sv
shift_reg_if.sv
shigt_reg_main_sequence_pkg.sv
shigt_reg_reset_sequence_pkg.sv
shigt_reg_sequence_item_pkg.sv
MySequencer_pkg.sv
package_shift_reg_env_pkg.sv
shift_reg_driver_pkg.sv
shift_reg_agent_pkg.sv
shift_reg_scoreboard_pkg.sv
shift_reg_monitor_pkg.sv
shift_reg_coverage_pkg.sv
ALSU_top.sv
```

Waveform snippets:



Transcript snippet:

```
# UVM_INFO ALSU_scoreboard_pkg.sv(49) @ 20022: uvm_test_top.env.sb [report_phase] Total coorect counts is: 0d10011
# UVM_INFO ALSU_scoreboard_pkg.sv(50) @ 20022: uvm_test_top.env.sb [report_phase] Total error counts is: 0d0
# UVM_INFO shift_reg_scoreboard_pkg.sv(70) @ 20022: uvm_test_top.env2.sb [report_phase] Total coorect counts is: 0d10011
# UVM_INFO shift_reg_scoreboard_pkg.sv(71) @ 20022: uvm_test_top.env2.sb [report_phase] Total error counts is: 0d0
#
# --- UVM Report Summary ---
#
# ** Report counts by severity
# UVM_INFO :   11
# UVM_WARNING :    0
# UVM_ERROR :    0
# UVM_FATAL :    0
# ** Report counts by id
# [Questa UVM]     2
# [RNTST]     1
# [TEST_DONE]     1
# [report_phase]     4
# [run_phase]     3
# ** Note: $finish    : E:/DIGITAL/Programs/QuestaSim/win64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh(430)
#    Time: 20022 ns  Iteration: 61  Instance: /ALSU_top
```

Code coverage report text file:

ALSU:

Branch coverage:

Can’t reach 100% branch coverage, Because there is a missing default case in case statement which couldn’t be hit, I had a discussion with Eng. Kareem and shared thoughts that it’s a tool bug so it’s ok not to reach 100% code coverage because of it.

```
=== Instance: /ALSU_top/ALSU_DUT
=== Design Unit: work.ALSU
================================================================================
Branch Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Branches                        32        31         1    96.87%

================================Branch Details================================

Branch Coverage for instance /ALSU_top/ALSU_DUT

    Line         Item                      Count     Source
    ----         ----                      -----     ------
  File ALSU.sv
------------------------------------IF Branch------------------------------------
    18                                     10507     Count coming in to IF
    18              1                       1023       if(ALSU_if.rst) begin
    29              1                       9484       end else begin
Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
    44                                     10507     Count coming in to IF
    44              1                       1023       if(ALSU_if.rst) begin
    46              1                       9484       end else begin
Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
    47                                      9484     Count coming in to IF
    47              1                       4886           if (invalid)
    49              1                       4598           else
Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
    55                                     10507     Count coming in to IF
    55              1                       1023       if(ALSU_if.rst) begin
    58              1                       9484       else begin
Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
    59                                      9484     Count coming in to IF
    59              1                       4886         if (invalid)
    61              1                         35         else if (bypass_A_reg && bypass_B_reg)
    63              1                        383         else if (bypass_A_reg)
    65              1                        348         else if (bypass_B_reg)
    67              1                       3832         else begin
Branch totals: 5 hits of 5 branches = 100.00%

-----------------------------------CASE Branch-----------------------------------
    68                                      3832     Count coming in to CASE
    69              1                       1612               3'h0: begin
    80              1                       1145               3'h1: begin
    92              1                        229               3'h2: if(FULL_ADDER=="ON") begin
    98              1                        265               3'h3: ALSU_if.out <= A_reg * B_reg;
    99              1                        286               3'h4: begin
    105             1                        295               3'h5: begin
                                           ***0***     All False Count
Branch totals: 6 hits of 7 branches = 85.71%

------------------------------------IF Branch------------------------------------
    71                                      1612     Count coming in to IF
    71              1                        131                if (red_op_A_reg && red_op_B_reg)
    73              1                        100                else if (red_op_A_reg)
    75              1                        101                else if (red_op_B_reg)
    77              1                       1280                else
Branch totals: 4 hits of 4 branches = 100.00%

------------------------------------IF Branch------------------------------------
    82                                      1145     Count coming in to IF
    82              1                        144                if (red_op_A_reg && red_op_B_reg)
    84              1                        113                else if (red_op_A_reg)
    86              1                        101                else if (red_op_B_reg)
    88              1                        787                else
Branch totals: 4 hits of 4 branches = 100.00%

------------------------------------IF Branch------------------------------------
    100                                      286     Count coming in to IF
    100             1                        153                if (direction_reg)
    102             1                        133                else
Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
    106                                      295     Count coming in to IF
    106             1                        164                if (direction_reg)
    108             1                        131                else
Branch totals: 2 hits of 2 branches = 100.00%
```

Condition coverage:

```
Condition Coverage:
    Enabled Coverage              Bins   Covered    Misses  Coverage
    ----------------              ----   -------    ------  --------
    Conditions                       6         6         0   100.00%

================================Condition Details================================

Condition Coverage for instance /ALSU_top/ALSU_DUT --

  File ALSU.sv
----------------Focused Condition View-------------------
Line       62 Item    1  (bypass_A_reg && bypass_B_reg)
Condition totals: 2 of 2 input terms covered = 100.00%

    Input Term   Covered  Reason for no coverage   Hint
   -----------  --------  -----------------------  --------------
  bypass_A_reg         Y
  bypass_B_reg         Y

     Rows:       Hits  FEC Target            Non-masking condition(s)
 ---------  ---------  --------------------  -------------------------
  Row   1:          1  bypass_A_reg_0        -
  Row   2:          1  bypass_A_reg_1        bypass_B_reg
  Row   3:          1  bypass_B_reg_0        bypass_A_reg
  Row   4:          1  bypass_B_reg_1        bypass_A_reg

----------------Focused Condition View-------------------
Line       72 Item    1  (red_op_A_reg && red_op_B_reg)
Condition totals: 2 of 2 input terms covered = 100.00%

    Input Term   Covered  Reason for no coverage   Hint
   -----------  --------  -----------------------  --------------
  red_op_A_reg         Y
  red_op_B_reg         Y

     Rows:       Hits  FEC Target            Non-masking condition(s)
 ---------  ---------  --------------------  -------------------------
  Row   1:          1  red_op_A_reg_0        -
  Row   2:          1  red_op_A_reg_1        red_op_B_reg
  Row   3:          1  red_op_B_reg_0        red_op_A_reg
  Row   4:          1  red_op_B_reg_1        red_op_A_reg

----------------Focused Condition View-------------------
Line       83 Item    1  (red_op_A_reg && red_op_B_reg)
Condition totals: 2 of 2 input terms covered = 100.00%

    Input Term   Covered  Reason for no coverage   Hint
   -----------  --------  -----------------------  --------------
  red_op_A_reg         Y
  red_op_B_reg         Y

     Rows:       Hits  FEC Target            Non-masking condition(s)
 ---------  ---------  --------------------  -------------------------
  Row   1:          1  red_op_A_reg_0        -
  Row   2:          1  red_op_A_reg_1        red_op_B_reg
  Row   3:          1  red_op_B_reg_0        red_op_A_reg
  Row   4:          1  red_op_B_reg_1        red_op_A_reg
```

Expression coverage:

```
Expression Coverage:
    Enabled Coverage              Bins   Covered    Misses  Coverage
    ----------------              ----   -------    ------  --------
    Expressions                      8         8         0   100.00%

================================Expression Details================================

Expression Coverage for instance /ALSU_top/ALSU_DUT --

  File ALSU.sv
----------------Focused Expression View-----------------
Line       14 Item    1  ((red_op_A_reg | red_op_B_reg) & (opcode_reg[1] | opcode_reg[2]))
Expression totals: 4 of 4 input terms covered = 100.00%

    Input Term   Covered  Reason for no coverage   Hint
   -----------  --------  -----------------------  --------------
  red_op_A_reg         Y
  red_op_B_reg         Y
  opcode_reg[1]        Y
  opcode_reg[2]        Y

     Rows:       Hits  FEC Target            Non-masking condition(s)
 ---------  ---------  --------------------  -------------------------
  Row   1:          1  red_op_A_reg_0        ((opcode_reg[1] | opcode_reg[2]) && ~red_op_B_reg)
  Row   2:          1  red_op_A_reg_1        ((opcode_reg[1] | opcode_reg[2]) && ~red_op_B_reg)
  Row   3:          1  red_op_B_reg_0        ((opcode_reg[1] | opcode_reg[2]) && ~red_op_A_reg)
  Row   4:          1  red_op_B_reg_1        ((opcode_reg[1] | opcode_reg[2]) && ~red_op_A_reg)
  Row   5:          1  opcode_reg[1]_0       ((red_op_A_reg | red_op_B_reg) && ~opcode_reg[2])
  Row   6:          1  opcode_reg[1]_1       ((red_op_A_reg | red_op_B_reg) && ~opcode_reg[2])
  Row   7:          1  opcode_reg[2]_0       ((red_op_A_reg | red_op_B_reg) && ~opcode_reg[1])
  Row   8:          1  opcode_reg[2]_1       ((red_op_A_reg | red_op_B_reg) && ~opcode_reg[1])

----------------Focused Expression View-----------------
Line       15 Item    1  (opcode_reg[1] & opcode_reg[2])
Expression totals: 2 of 2 input terms covered = 100.00%

    Input Term   Covered  Reason for no coverage   Hint
   -----------  --------  -----------------------  --------------
  opcode_reg[1]        Y
  opcode_reg[2]        Y

     Rows:       Hits  FEC Target            Non-masking condition(s)
 ---------  ---------  --------------------  -------------------------
  Row   1:          1  opcode_reg[1]_0       opcode_reg[2]
  Row   2:          1  opcode_reg[1]_1       opcode_reg[2]
  Row   3:          1  opcode_reg[2]_0       opcode_reg[1]
  Row   4:          1  opcode_reg[2]_1       opcode_reg[1]

----------------Focused Expression View-----------------
Line       16 Item    1  (invalid_red_op | invalid_opcode)
Expression totals: 2 of 2 input terms covered = 100.00%

     Input Term   Covered  Reason for no coverage   Hint
    -----------  --------  -----------------------  --------------
  invalid_red_op         Y
  invalid_opcode         Y

     Rows:       Hits  FEC Target            Non-masking condition(s)
 ---------  ---------  --------------------  -------------------------
  Row   1:          1  invalid_red_op_0      ~invalid_opcode
  Row   2:          1  invalid_red_op_1      ~invalid_opcode
  Row   3:          1  invalid_opcode_0      ~invalid_red_op
  Row   4:          1  invalid_opcode_1      ~invalid_red_op
```

Statement coverage:

```
Statement Coverage:
    Enabled Coverage             Bins      Hits    Misses  Coverage
    ----------------             ----      ----    ------  --------
    Statements                     48        48         0   100.00%

================================Statement Details================================

Statement Coverage for instance /ALSU_top/ALSU_DUT --

    Line         Item                     Count     Source
    ----         ----                     -----     ------
  File ALSU.sv
    2                                              module ALSU(ALSU_interface.DUT ALSU_if);
    3
    4                                              // Parameters & Internal signals
    5                                              parameter INPUT_PRIORITY = "A";
    6                                              parameter FULL_ADDER = "ON";
    7                                              reg red_op_A_reg, red_op_B_reg, bypass_A_reg, bypass_B_reg, direction_reg, serial_in_reg;
    8                                              reg signed [1:0] cin_reg;//Make cin_reg signed 2bits to perform correct signed addition
    9                                              reg [2:0] opcode_reg;
    10                                             reg signed [2:0] A_reg, B_reg;//A & B registers are signed as well as A & B
    11                                             wire invalid_red_op, invalid_opcode, invalid;
    12
    13                                             //Invalid handling
    14             1                        9551   assign invalid_red_op = (red_op_A_reg | red_op_B_reg) & (opcode_reg[1] | opcode_reg[2]);
    15             1                        8649   assign invalid_opcode = opcode_reg[1] & opcode_reg[2];
    16             1                        5848   assign invalid = invalid_red_op | invalid_opcode;
    17                                             //Registering input signals
    18             1                       10451   always @(posedge ALSU_if.clk or posedge ALSU_if.rst) begin
    19                                               if(ALSU_if.rst) begin
    20             1                         902                  cin_reg <= 0;
    21             1                         902                  red_op_B_reg <= 0;
    22             1                         902                  red_op_A_reg <= 0;
    23             1                         902                  bypass_B_reg <= 0;
    24             1                         902                  bypass_A_reg <= 0;
    25             1                         902                  direction_reg <= 0;
    26             1                         902                  serial_in_reg <= 0;
    27             1                         902                  opcode_reg <= 0;
    28             1                         902                  A_reg <= 0;
    29             1                         902                  B_reg <= 0;
    30                                               end else begin
    31             1                        9549                  cin_reg <= ALSU_if.cin;
    32             1                        9549                  red_op_B_reg <= ALSU_if.red_op_B;
    33             1                        9549                  red_op_A_reg <= ALSU_if.red_op_A;
    34             1                        9549                  bypass_B_reg <= ALSU_if.bypass_B;
    35             1                        9549                  bypass_A_reg <= ALSU_if.bypass_A;
    36             1                        9549                  direction_reg <= ALSU_if.direction;
    37             1                        9549                  serial_in_reg <= ALSU_if.serial_in;
    38             1                        9549                  opcode_reg <= ALSU_if.opcode;
    39             1                        9549                  A_reg <= ALSU_if.A;
    40             1                        9549                  B_reg <= ALSU_if.B;
    41                                               end
    42                                             end
    43                                             //leds output blinking
    44             1                       10451   always @(posedge ALSU_if.clk or posedge ALSU_if.rst) begin
    45                                               if(ALSU_if.rst) begin
    46             1                         902                  ALSU_if.leds <= 0;
    47                                               end else begin
    48                                                   if (invalid)
    49             1                        5038                     ALSU_if.leds <= ~ALSU_if.leds;
    50                                                   else
    51             1                        4511                     ALSU_if.leds <= 0;
    52                                               end
    53                                             end
    54                                             //ALSU output processing
    55             1                       10451   always @(posedge ALSU_if.clk or posedge ALSU_if.rst) begin
    56                                               if(ALSU_if.rst) begin
    57             1                         902                  ALSU_if.out <= 0;
    58                                               end
    59                                               else begin
    60                                                 if (invalid)
    61             1                        5038                     ALSU_if.out <= 0;
    62                                                 else if (bypass_A_reg && bypass_B_reg)
    63             1                          38                       ALSU_if.out <= (INPUT_PRIORITY == "A")? A_reg: B_reg;
```

```
    61             1                        5038                     ALSU_if.out <= 0;
    62                                                 else if (bypass_A_reg && bypass_B_reg)
    63             1                          38                       ALSU_if.out <= (INPUT_PRIORITY == "A")? A_reg: B_reg;
    64                                                 else if (bypass_A_reg)
    65             1                         348                       ALSU_if.out <= A_reg;
    66                                                 else if (bypass_B_reg)
    67             1                         396                       ALSU_if.out <= B_reg;
    68                                                 else begin
    69                                                     case (opcode_reg)//Bug: opcode --> opcode_reg
    70                                                       3'h0: begin
    71                                                         //Third bug --> AND replaced with OR
    72                                                         if (red_op_A_reg && red_op_B_reg)
    73             1                         161                       ALSU_if.out = (INPUT_PRIORITY == "A")? |A_reg: |B_reg;
    74                                                         else if (red_op_A_reg)
    75             1                         112                       ALSU_if.out <= |A_reg;
    76                                                         else if (red_op_B_reg)
    77             1                         107                       ALSU_if.out <= |B_reg;
    78                                                         else
    79             1                        1151                       ALSU_if.out <= A_reg | B_reg;
    80                                                       end
    81                                                       3'h1: begin
    82                                                         //Fourth bug --> OR replaced with XOR
    83                                                         if (red_op_A_reg && red_op_B_reg)
    84             1                         168                       ALSU_if.out <= (INPUT_PRIORITY == "A")? ^A_reg: ^B_reg;
    85                                                         else if (red_op_A_reg)
    86             1                         124                       ALSU_if.out <= ^A_reg;
    87                                                         else if (red_op_B_reg)
    88             1                         101                       ALSU_if.out <= ^B_reg;
    89                                                         else
    90             1                         717                       ALSU_if.out <= A_reg ^ B_reg;
    91                                                       end
    92                                                       //Fifth bug --> Added logic condition for FULL_ADDER operation
    93                                                       3'h2: if(FULL_ADDER=="ON") begin
    94             1                         273                       ALSU_if.out <= A_reg + B_reg + cin_reg;
    95                                                       end
    96                                                       else if (FULL_ADDER=="OFF") begin
    97                                                         ALSU_if.out <= A_reg + B_reg ;
    98                                                       end
    99             1                         286                     3'h3: ALSU_if.out <= A_reg * B_reg;
    100                                                      3'h4: begin
    101                                                        if (direction_reg)
    102            1                         159                       ALSU_if.out <= ALSU_if.out_shift_reg;
    103                                                        else
    104            1                         123                       ALSU_if.out <= ALSU_if.out_shift_reg;
    105                                                      end
    106                                                      3'h5: begin
    107                                                        if (direction_reg)
    108            1                         137                       ALSU_if.out <= ALSU_if.out_shift_reg;
    109                                                        else
    110            1                         110                       ALSU_if.out <= ALSU_if.out_shift_reg;
```

Toggle coverage:

```
Toggle Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Toggles                        138       138         0   100.00%

================================Toggle Details================================

Toggle Coverage for instance /ALSU_top/ALSU_if --

                                          Node      1H->0L      0L->1H                          "Coverage"
                                          ----------------------------------------------------------------
                                        A[2-0]           1           1                              100.00
                                        B[2-0]           1           1                              100.00
                                     bypass_A            1           1                              100.00
                                     bypass_B            1           1                              100.00
                                          cin            1           1                              100.00
                                          clk            1           1                              100.00
                                    direction            1           1                              100.00
                                    leds[15-0]           1           1                              100.00
                                leds_ref[15-0]           1           1                              100.00
                                       opcode                ENUM type       Value       Count
                                                                   OR           6      100.00
                                                                  XOR           6      100.00
                                                                  ADD           2      100.00
                                                                 MULT           4      100.00
                                                                SHIFT           4      100.00
                                                               ROTATE           3      100.00
                                                            INVALID_6           3      100.00
                                                            INVALID_7           1      100.00
                                      out[5-0]           1           1                              100.00
                                  out_ref[5-0]           1           1                              100.00
                             out_shift_reg[5-0]          1           1                              100.00
                                     red_op_A            1           1                              100.00
                                     red_op_B            1           1                              100.00
                                          rst            1           1                              100.00
                                    serial_in            1           1                              100.00

Total Node Count     =          73
Toggled Node Count   =          73
Untoggled Node Count =           0

Toggle Coverage      =     100.00% (138 of 138 bins)
```

Functional coverage:

Report text file:

```
=== Instance: /ALSU_coverage_pkg
=== Design Unit: work.ALSU_coverage_pkg
================================================================================

Covergroup Coverage:
    Covergroups                    1        na        na   100.00%
        Coverpoints/Crosses       18        na        na        na
            Covergroup Bins       45        45         0   100.00%
--------------------------------------------------------------------------------
Covergroup                                          Metric       Goal       Bins    Status
--------------------------------------------------------------------------------
 TYPE /ALSU_coverage_pkg/ALSU_coverage/cg           100.00%        100          -    Covered
    covered/total bins:                                  45         45          -
    missing/total bins:                                   0         45          -
    % Hit:                                           100.00%        100          -
    Coverpoint A_cp                                  100.00%        100          -    Covered
        covered/total bins:                                3          3          -
        missing/total bins:                                0          3          -
        % Hit:                                       100.00%        100          -
    Coverpoint A_cp_walking_ones                     100.00%        100          -    Covered
        covered/total bins:                                2          2          -
        missing/total bins:                                0          2          -
        % Hit:                                       100.00%        100          -
    Coverpoint B_cp                                  100.00%        100          -    Covered
        covered/total bins:                                3          3          -
        missing/total bins:                                0          3          -
        % Hit:                                       100.00%        100          -
    Coverpoint B_cp_walking_ones                     100.00%        100          -    Covered
        covered/total bins:                                2          2          -
        missing/total bins:                                0          2          -
        % Hit:                                       100.00%        100          -
    Coverpoint ALU_cp                                100.00%        100          -    Covered
        covered/total bins:                                7          7          -
        missing/total bins:                                0          7          -
        % Hit:                                       100.00%        100          -
    Coverpoint opcode_cp                               0.00%        100          -    ZERO
        covered/total bins:                                0          8          -
        missing/total bins:                                8          8          -
        % Hit:                                         0.00%        100          -
    Coverpoint cin_cp                                  0.00%        100          -    ZERO
        covered/total bins:                                0          2          -
        missing/total bins:                                2          2          -
        % Hit:                                         0.00%        100          -
    Coverpoint serialin_cp                             0.00%        100          -    ZERO
        covered/total bins:                                0          2          -
        missing/total bins:                                2          2          -
        % Hit:                                         0.00%        100          -
    Coverpoint direction_cp                            0.00%        100          -    ZERO
        covered/total bins:                                0          2          -
        missing/total bins:                                2          2          -
        % Hit:                                         0.00%        100          -
    Coverpoint red_op_A_cp                             0.00%        100          -    ZERO
        covered/total bins:                                0          2          -
        missing/total bins:                                2          2          -
        % Hit:                                         0.00%        100          -
    Coverpoint red_op_B_cp                             0.00%        100          -    ZERO
        covered/total bins:                                0          2          -
        missing/total bins:                                2          2          -
        % Hit:                                         0.00%        100          -
    Cross #cross__0#                                 100.00%        100          -    Covered
        covered/total bins:                                1          1          -
        missing/total bins:                                0          1          -
        % Hit:                                       100.00%        100          -
    Cross #cross__1#                                 100.00%        100          -    Covered
        covered/total bins:                                2          2          -
        missing/total bins:                                0          2          -
        % Hit:                                       100.00%        100          -
    Cross #cross__2#                                 100.00%        100          -    Covered
        covered/total bins:                                2          2          -
        missing/total bins:                                0          2          -
        % Hit:                                       100.00%        100          -
    Cross #cross__3#                                 100.00%        100          -    Covered    |
        covered/total bins:                                2          2          -
```

```
    % Hit:                                          100.00%        100         -
Cross #cross__3#                                    100.00%        100         -    Covered
    covered/total bins:                                   2          2         -
    missing/total bins:                                   0          2         -
    % Hit:                                          100.00%        100         -
Cross #cross__4#                                    100.00%        100         -    Covered
    covered/total bins:                                   1          1         -
    missing/total bins:                                   0          1         -
    % Hit:                                          100.00%        100         -
Cross #cross__5#                                    100.00%        100         -    Covered
    covered/total bins:                                   1          1         -
    missing/total bins:                                   0          1         -
    % Hit:                                          100.00%        100         -
Cross #cross__6#                                    100.00%        100         -    Covered
    covered/total bins:                                   1          1         -
    missing/total bins:                                   0          1         -
    % Hit:                                          100.00%        100         -
Covergroup instance \/ALSU_coverage_pkg::ALSU_coverage::cg
                                                    100.00%        100         -    Covered
covered/total bins:                                      45         45         -
missing/total bins:                                       0         45         -
% Hit:                                              100.00%        100         -
Coverpoint A_cp                                     100.00%        100         -    Covered
    covered/total bins:                                   3          3         -
    missing/total bins:                                   0          3         -
    % Hit:                                          100.00%        100         -
    bin A_data_0                                       1921          1         -    Covered
    bin A_data_max                                     1715          1         -    Covered
    bin A_data_min                                     1589          1         -    Covered
    default bin A_data_default                         4786                    -    Occurred
Coverpoint A_cp_walking_ones                        100.00%        100         -    Covered
    covered/total bins:                                   2          2         -
    missing/total bins:                                   0          2         -
    % Hit:                                          100.00%        100         -
    bin A_data_walkingones[1]                           422          1         -    Covered
    bin A_data_walkingones[2]                           432          1         -    Covered
Coverpoint B_cp                                     100.00%        100         -    Covered
    covered/total bins:                                   3          3         -
    missing/total bins:                                   0          3         -
    % Hit:                                          100.00%        100         -
    bin B_data_0                                       2299          1         -    Covered
    bin B_data_max                                     1666          1         -    Covered
    bin B_data_min                                     1545          1         -    Covered
    default bin B_data_default                         4501                    -    Occurred
Coverpoint B_cp_walking_ones                        100.00%        100         -    Covered
    covered/total bins:                                   2          2         -
    missing/total bins:                                   0          2         -
    % Hit:                                          100.00%        100         -
    bin B_data_walkingones[1]                           221          1         -    Covered
    bin B_data_walkingones[2]                           194          1         -    Covered
Coverpoint ALU_cp                                   100.00%        100         -    Covered
    covered/total bins:                                   7          7         -
    missing/total bins:                                   0          7         -
    % Hit:                                          100.00%        100         -
    illegal_bin Bins_invalid                           1035                    -    Occurred
    bin Bins_shift[SHIFT]                              1462          1         -    Covered
    bin Bins_shift[ROTATE]                             1544          1         -    Covered
    bin Bins_arith[ADD]                                1404          1         -    Covered
    bin Bins_arith[MULT]                               1467          1         -    Covered
    bin Bins_bitwise[OR]                               1527          1         -    Covered
    bin Bins_bitwise[XOR]                              1572          1         -    Covered
    bin Bins_trans                                        1          1         -    Covered
Coverpoint opcode_cp [1]                            100.00%        100         -    Covered
    covered/total bins:                                   8          8         -
    missing/total bins:                                   0          8         -
    % Hit:                                          100.00%        100         -
    bin auto[OR]                                       1527          1         -    Covered
    bin auto[XOR]                                      1572          1         -    Covered
    bin auto[ADD]                                      1404          1         -    Covered
    bin auto[MULT]                                     1467          1         -    Covered
    bin auto[SHIFT]                                    1462          1         -    Covered
    bin auto[ROTATE]                                   1544          1         -    Covered
    bin auto[INVALID_6]                                 504          1         -    Covered
    bin auto[INVALID_7]                                 531          1         -    Covered
```

```
    bin auto[INVALID_6]                                 504          1         -    Covered
    bin auto[INVALID_7]                                 531          1         -    Covered
Coverpoint cin_cp [1]                               100.00%        100         -    Covered
    covered/total bins:                                   2          2         -
    missing/total bins:                                   0          2         -
    % Hit:                                          100.00%        100         -
    bin auto[0]                                        5015          1         -    Covered
    bin auto[1]                                        4996          1         -    Covered
Coverpoint serialin_cp [1]                          100.00%        100         -    Covered
    covered/total bins:                                   2          2         -
    missing/total bins:                                   0          2         -
    % Hit:                                          100.00%        100         -
    bin auto[0]                                        4984          1         -    Covered
    bin auto[1]                                        5027          1         -    Covered
Coverpoint direction_cp [1]                         100.00%        100         -    Covered
    covered/total bins:                                   2          2         -
    missing/total bins:                                   0          2         -
    % Hit:                                          100.00%        100         -
    bin auto[0]                                        4963          1         -    Covered
    bin auto[1]                                        5048          1         -    Covered
Coverpoint red_op_A_cp [1]                          100.00%        100         -    Covered
    covered/total bins:                                   2          2         -
    missing/total bins:                                   0          2         -
    % Hit:                                          100.00%        100         -
    bin auto[0]                                        5896          1         -    Covered
    bin auto[1]                                        4115          1         -    Covered
Coverpoint red_op_B_cp [1]                          100.00%        100         -    Covered
    covered/total bins:                                   2          2         -
    missing/total bins:                                   0          2         -
    % Hit:                                          100.00%        100         -
    bin auto[0]                                        5844          1         -    Covered
    bin auto[1]                                        4167          1         -    Covered
Cross #cross__0#                                    100.00%        100         -    Covered
    covered/total bins:                                   1          1         -
    missing/total bins:                                   0          1         -
    % Hit:                                          100.00%        100         -
    Auto, Default and User Defined Bins:
        bin ADD_MULT_EXTREEMS                          2044          1         -    Covered
Cross #cross__1#                                    100.00%        100         -    Covered
    covered/total bins:                                   2          2         -
    missing/total bins:                                   0          2         -
    % Hit:                                          100.00%        100         -
    Auto, Default and User Defined Bins:
        bin ADD_CIN0                                   716          1         -    Covered
        bin ADD_CIN1                                   688          1         -    Covered
Cross #cross__2#                                    100.00%        100         -    Covered
    covered/total bins:                                   2          2         -
    missing/total bins:                                   0          2         -
    % Hit:                                          100.00%        100         -
    Auto, Default and User Defined Bins:
        bin SHIFT_SERIALIN0                            738          1         -    Covered
        bin SHIFT_SERIALIN1                            724          1         -    Covered
Cross #cross__3#                                    100.00%        100         -    Covered
    covered/total bins:                                   2          2         -
    missing/total bins:                                   0          2         -
    % Hit:                                          100.00%        100         -
    Auto, Default and User Defined Bins:
        bin SHIFT_ROTATE_DIRECTION0                   1491          1         -    Covered
        bin SHIFT_ROTATE_DIRECTION1                   1515          1         -    Covered
Cross #cross__4#                                    100.00%        100         -    Covered
    covered/total bins:                                   1          1         -
    missing/total bins:                                   0          1         -
    % Hit:                                          100.00%        100         -
    Auto, Default and User Defined Bins:
        bin OR_XOR_REDA                                371          1         -    Covered
Cross #cross__5#                                    100.00%        100         -    Covered
    covered/total bins:                                   1          1         -
    missing/total bins:                                   0          1         -
    % Hit:                                          100.00%        100         -
    Auto, Default and User Defined Bins:
        bin OR_XOR_REDB                                148          1         -    Covered
Cross #cross__6#                                    100.00%        100         -    Covered
    covered/total bins:                                   1          1         -
    missing/total bins:                                   0          1         -
    % Hit:                                          100.00%        100         -
    Auto, Default and User Defined Bins:
        bin INVALID_REDUCTION                          5182          1         -    Covered
```

Assertions coverage:

```
Assertion Coverage:
    Assertions                      16        16         0   100.00%
--------------------------------------------------------------------
Name                 File(Line)                   Failure      Pass
                                                  Count        Count
--------------------------------------------------------------------
/ALSU_top/ALSU_DUT/SVA/reset_assertion
                     ALSU_SVA.sv(9)                     0          1
/ALSU_top/ALSU_DUT/SVA/invalid_property_assertion
                     ALSU_SVA.sv(104)                   0          1
/ALSU_top/ALSU_DUT/SVA/bypass_A_property_assertion
                     ALSU_SVA.sv(107)                   0          1
/ALSU_top/ALSU_DUT/SVA/bypass_B_property_assertion
                     ALSU_SVA.sv(110)                   0          1
/ALSU_top/ALSU_DUT/SVA/red_OR_A_property_assertion
                     ALSU_SVA.sv(113)                   0          1
/ALSU_top/ALSU_DUT/SVA/red_OR_B_property_assertion
                     ALSU_SVA.sv(116)                   0          1
/ALSU_top/ALSU_DUT/SVA/red_OR_A_B_property_assertion
                     ALSU_SVA.sv(119)                   0          1
/ALSU_top/ALSU_DUT/SVA/red_XOR_A_property_assertion
                     ALSU_SVA.sv(122)                   0          1
/ALSU_top/ALSU_DUT/SVA/red_XOR_B_property_assertion
                     ALSU_SVA.sv(125)                   0          1
/ALSU_top/ALSU_DUT/SVA/red_XOR_A_B_property_assertion
                     ALSU_SVA.sv(128)                   0          1
/ALSU_top/ALSU_DUT/SVA/Add1_assertion
                     ALSU_SVA.sv(131)                   0          1
/ALSU_top/ALSU_DUT/SVA/Mult_assertion
                     ALSU_SVA.sv(134)                   0          1
/ALSU_top/ALSU_DUT/SVA/shift_left_assertion
                     ALSU_SVA.sv(137)                   0          1
/ALSU_top/ALSU_DUT/SVA/shift_right_assertion
                     ALSU_SVA.sv(140)                   0          1
/ALSU_top/ALSU_DUT/SVA/rotate_left_assertion
                     ALSU_SVA.sv(143)                   0          1
/ALSU_top/ALSU_DUT/SVA/rotate_right_assertion
                     ALSU_SVA.sv(146)                   0          1
```

Directive coverage:

```
Directive Coverage:
    Directives                      16        16         0   100.00%

DIRECTIVE COVERAGE:
-------------------------------------------------------------------------------------------------
Name                                    Design Design   Lang File(Line)      Hits Status
                                        Unit   UnitType
-------------------------------------------------------------------------------------------------
/ALSU_top/ALSU_DUT/SVA/reset_cover      ALSU_SVA Verilog  SVA  ALSU_SVA.sv(10)  518 Covered
/ALSU_top/ALSU_DUT/SVA/invalid_property_cover
                                        ALSU_SVA Verilog  SVA  ALSU_SVA.sv(105)4691 Covered
/ALSU_top/ALSU_DUT/SVA/bypass_A_property_cover
                                        ALSU_SVA Verilog  SVA  ALSU_SVA.sv(108) 366 Covered
/ALSU_top/ALSU_DUT/SVA/bypass_B_property_cover
                                        ALSU_SVA Verilog  SVA  ALSU_SVA.sv(111) 332 Covered
/ALSU_top/ALSU_DUT/SVA/red_OR_A_property_cover
                                        ALSU_SVA Verilog  SVA  ALSU_SVA.sv(114) 228 Covered
/ALSU_top/ALSU_DUT/SVA/red_OR_B_property_cover
                                        ALSU_SVA Verilog  SVA  ALSU_SVA.sv(117) 116 Covered
/ALSU_top/ALSU_DUT/SVA/red_OR_A_B_property_cover
                                        ALSU_SVA Verilog  SVA  ALSU_SVA.sv(120) 682 Covered
/ALSU_top/ALSU_DUT/SVA/red_XOR_A_property_cover
                                        ALSU_SVA Verilog  SVA  ALSU_SVA.sv(123) 261 Covered
/ALSU_top/ALSU_DUT/SVA/red_XOR_B_property_cover
                                        ALSU_SVA Verilog  SVA  ALSU_SVA.sv(126) 115 Covered
/ALSU_top/ALSU_DUT/SVA/red_XOR_A_B_property_cover
                                        ALSU_SVA Verilog  SVA  ALSU_SVA.sv(129) 671 Covered
/ALSU_top/ALSU_DUT/SVA/Add1_cover       ALSU_SVA Verilog  SVA  ALSU_SVA.sv(132) 245 Covered
/ALSU_top/ALSU_DUT/SVA/Mult_cover       ALSU_SVA Verilog  SVA  ALSU_SVA.sv(135) 252 Covered
/ALSU_top/ALSU_DUT/SVA/shift_left_cover ALSU_SVA Verilog  SVA  ALSU_SVA.sv(138) 112 Covered
/ALSU_top/ALSU_DUT/SVA/shift_right_cover ALSU_SVA Verilog  SVA  ALSU_SVA.sv(141) 142 Covered
/ALSU_top/ALSU_DUT/SVA/rotate_left_cover ALSU_SVA Verilog  SVA  ALSU_SVA.sv(144) 117 Covered
/ALSU_top/ALSU_DUT/SVA/rotate_right_cover
                                        ALSU_SVA Verilog  SVA  ALSU_SVA.sv(147) 131 Covered
```

Code coverage report text file:

Shift Register:

Branch coverage:

```
Branch Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Branches                         5         5         0   100.00%

================================Branch Details================================

Branch Coverage for instance /ALSU_top/SHIFT_REG_DUT

    Line         Item                      Count     Source
    ----         ----                      -----     ------
  File shift_reg.sv
------------------------------------IF Branch------------------------------------
    11                                      9490     Count coming in to IF
    11              1                       4457        if (shift_regif.mode) // rotate
    17              1                       2299           if (shift_regif.direction) // left
    19              1                       2734           else // Right
Branch totals: 3 hits of 3 branches = 100.00%

------------------------------------IF Branch------------------------------------
    12                                      4457     Count coming in to IF
    12              1                       2220           if (shift_regif.direction) // left
    14              1                       2237           else// Right
Branch totals: 2 hits of 2 branches = 100.00%
```

Condition coverage:

```
Statement Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Statements                       5         5         0   100.00%

================================Statement Details================================

Statement Coverage for instance /ALSU_top/SHIFT_REG_DUT --

    Line         Item                      Count     Source
    ----         ----                      -----     ------
  File shift_reg.sv
    8                                                 module shift_reg (shift_reg_if.SHIFT_REG_DUT_MP shift_regif);
    9
    10              1                            1    always @(*) begin
    11                                                  if (shift_regif.mode) // rotate
    12                                                     if (shift_regif.direction) // left
    13              1                         2220           shift_regif.dataout <= {shift_regif.datain[4:0], shift_regif.datain[5]};
    14                                                     else// Right
    15              1                         2237           shift_regif.dataout <= {shift_regif.datain[0], shift_regif.datain[5:1]};
    16                                                  else  // shift
    17                                                     if (shift_regif.direction) // left
    18              1                         2299           shift_regif.dataout <= {shift_regif.datain[4:0], shift_regif.serial_in};
    19                                                     else // Right
    20              1                         2734           shift_regif.dataout <= {shift_regif.serial_in, shift_regif.datain[5:1]};
```

Statement coverage:

```
Statement Coverage:
    Enabled Coverage             Bins      Hits    Misses  Coverage
    ----------------             ----      ----    ------  --------
    Statements                     14        14         0   100%

================================Statement Details================================

Statement Coverage for instance /shift_reg_coverage_pkg --

    Line         Item                    Count     Source
    ----         ----                    -----     ------
  File shift_reg_coverage_pkg.sv
    1                                              package shift_reg_coverage_pkg;
    2                                                 import uvm_pkg::*;
    3                                                 import shared_pkg::*;
    4                                                 import shigt_reg_sequence_item_pkg::*;
    5                                                 `include "uvm_macros.svh"
    6                                                 class shift_reg_coverage extends uvm_component;
    7               1                                         `uvm_component_utils(shift_reg_coverage)
    7               2
    7               3
    8                                                         uvm_analysis_export #(shigt_reg_sequence_item) cov_export;
    9                                                         uvm_tlm_analysis_fifo #(shigt_reg_sequence_item) cov_fifo;
    10                                                        shigt_reg_sequence_item seq_item_cov;
    11
    12                                                        // Covergroups
    13                                                        covergroup cg;
    14                                                                data_in_cp: coverpoint seq_item_cov.datain;
    15                                                                direction_cp: coverpoint seq_item_cov.direction;
    16                                                                mode_cp: coverpoint seq_item_cov.mode;
    17                                                                serial_in_cp: coverpoint seq_item_cov.serial_in;
    18                                                        endgroup : cg
    19
    20                                                        function new(string name = "shift_reg_scoreboard", uvm_component parent = null);
    21              1                        1                        super.new(name,parent);
    22              1                        1                        cg=new();
    23                                                        endfunction
    24
    25                                                        function void build_phase(uvm_phase phase);
    26              1                        1                        super.build_phase(phase);
    27              1                        1                        cov_export =new("cov_export",this);
    28              1                        1                        cov_fifo =new("cov_fifo",this);
    29                                                        endfunction : build_phase
    30
    31                                                        function void connect_phase(uvm_phase phase);
    32              1                        1                        super.connect_phase(phase);
    33              1                        1                        cov_export.connect(cov_fifo.analysis_export);
    34                                                        endfunction : connect_phase
    35
    36                                                        task run_phase(uvm_phase phase);
    37              1                        1                        super.run_phase(phase);
    38              1                        1                        forever begin
    39              1                    10012                                cov_fifo.get(seq_item_cov);
    40              1                    10011                                cg.sample();
```

Toggle coverage:

```
Toggle Coverage:
    Enabled Coverage             Bins      Hits    Misses  Coverage
    ----------------             ----      ----    ------  --------
    Toggles                        30        30         0   100.00%

================================Toggle Details================================

Toggle Coverage for instance /ALSU_top/shift_regif --

                                        Node      1H->0L      0L->1H                          "Coverage"
                                        ----------------------------------------------------------------
                                   datain[5-0]           1           1                             100.00
                                  dataout[5-0]           1           1                             100.00
                                     direction                ENUM type       Value       Count
                                                                  RIGHT           1      100.00
                                                                   LEFT           1      100.00
                                        mode                ENUM type       Value       Count
                                                                  SHIFT           1      100.00
                                                                 ROTATE           1      100.00
                                   serial_in           1           1                             100.00

Total Node Count     =         17
Toggled Node Count   =         17
Untoggled Node Count =          0

Toggle Coverage      =     100.00% (30 of 30 bins)
```

Functional coverage:

Report text file:

```
Covergroup Coverage:
    Covergroups                    1        na        na    100%
        Coverpoints/Crosses        4        na        na     na
            Covergroup bins       78        78         0    100%
----------------------------------------------------------------------------------------------------------------
Covergroup                                          Metric       Goal       Bins    Status
----------------------------------------------------------------------------------------------------------------
 TYPE /shift_reg_coverage_pkg/shift_reg_coverage/cg
                                                    100%          100          -    Covered
    covered/total bins:                                78           78          -
    missing/total bins:                                 0           78          -
    % Hit:                                         100%          100          -
    Coverpoint data_in_cp                          100%          100          -    Uncovered
        covered/total bins:                            64           64          -
        missing/total bins:                             0           64          -
        % Hit:                                     100%          100          -
        bin auto[0]                                 6841            1          -    Covered
        bin auto[1]                                  942            1          -    Covered
        bin auto[2]                                  214            1          -    Covered
        bin auto[3]                                  336            1          -    Covered
        bin auto[4]                                   44            1          -    Covered
        bin auto[5]                                    1            1          -    Covered
        bin auto[6]                                   20            1          -    Covered
        bin auto[7]                                    9            1          -    Covered
        bin auto[8]                                   11            1          -    Covered
        bin auto[9]                                   32            1          -    Covered
        bin auto[10]                                   0            1          -    Covered
        bin auto[11]                                   0            1          -    Covered
        bin auto[12]                                  10            1          -    Covered
        bin auto[13]                                   4            1          -    Covered
        bin auto[14]                                   2            1          -    Covered
        bin auto[15]                                   6            1          -    Covered
        bin auto[16]                                  23            1          -    Covered
        bin auto[17]                                  16            1          -    Covered
        bin auto[18]                                   3            1          -    Covered
        bin auto[19]                                   1            1          -    Covered
        bin auto[20]                                   8            1          -    Covered
        bin auto[21]                                   6            1          -    Covered
        bin auto[22]                                   2            1          -    Covered
        bin auto[23]                                   1            1          -    Covered
        bin auto[24]                                   8            1          -    Covered
        bin auto[25]                                   7            1          -    Covered
        bin auto[26]                                   1            1          -    Covered
        bin auto[27]                                   4            1          -    Covered
        bin auto[28]                                  12            1          -    Covered
        bin auto[29]                                   9            1          -    Covered
        bin auto[30]                                  10            1          -    Covered
        bin auto[31]                                  10            1          -    Covered
        bin auto[32]                                  75            1          -    Covered
        bin auto[33]                                   8            1          -    Covered
        bin auto[34]                                   1            1          -    Covered
        bin auto[35]                                   6            1          -    Covered
        bin auto[36]                                   2            1          -    Covered
        bin auto[37]                                  18            1          -    Covered
        bin auto[38]                                   5            1          -    Covered
        bin auto[39]                                   1            1          -    Covered
        bin auto[40]                                   1            1          -    Covered
        bin auto[41]                                   1            1          -    Covered
        bin auto[42]                                  19            1          -    Covered
        bin auto[43]                                   8            1          -    Covered
        bin auto[44]                                   1            1          -    Covered
        bin auto[45]                                   2            1          -    Covered
        bin auto[46]                                  19            1          -    Covered
        bin auto[47]                                   1            1          -    Covered
        bin auto[48]                                   1            1          -    Covered
        bin auto[49]                                  18            1          -    Covered
        bin auto[50]                                  12            1          -    Covered
        bin auto[51]                                   2            1          -    Covered
        bin auto[52]                                  32            1          -    Covered
        bin auto[53]                                  40            1          -    Covered
        bin auto[54]                                   7            1          -    Covered
        bin auto[55]                                   8            1          -    Covered
        bin auto[56]                                   4            1          -    Covered
        bin auto[57]                                  14            1          -    Covered
        bin auto[58]                                  20            1          -    Covered
        bin auto[59]                                  21            1          -    Covered
        bin auto[60]                                 230            1          -    Covered
        bin auto[61]                                 300            1          -    Covered
        bin auto[62]                                 265            1          -    Covered
        bin auto[63]                                 512            1          -    Covered
    Coverpoint direction_cp                    100.00%          100          -    Covered
        covered/total bins:                            2            2          -
        missing/total bins:                            0            2          -
        % Hit:                                 100.00%          100          -
        bin auto[RIGHT]                             5198            1          -    Covered
        bin auto[LEFT]                              4713            1          -    Covered
    Coverpoint mode_cp                         100.00%          100          -    Covered
        covered/total bins:                            2            2          -
        missing/total bins:                            0            2          -
        % Hit:                                 100.00%          100          -
        bin auto[SHIFT]                             5195            1          -    Covered
        bin auto[ROTATE]                            4716            1          -    Covered
    Coverpoint serial_in_cp                    100.00%          100          -    Covered
        covered/total bins:                            2            2          -
        missing/total bins:                            0            2          -
        % Hit:                                 100.00%          100          -
        bin auto[0]                                 5246            1          -    Covered
        bin auto[1]                                 4765            1          -    Covered
```

```
Covergroup Coverage:
    Covergroups                 1        na        na   100.00%
        Coverpoints/Crosses    18        na        na        na
            Covergroup Bins    45        45         0   100.00%
----------------------------------------------------------------------------------------------------
Covergroup                                          Metric       Goal       Bins    Status

----------------------------------------------------------------------------------------------------
 TYPE /ALSU_coverage_pkg/ALSU_coverage/cg          100.00%        100          -    Covered
    covered/total bins:                                 45         45          -
    missing/total bins:                                  0         45          -
    % Hit:                                         100.00%        100          -
    Coverpoint A_cp                                100.00%        100          -    Covered
        covered/total bins:                              3          3          -
        missing/total bins:                              0          3          -
        % Hit:                                     100.00%        100          -
    Coverpoint A_cp_walking_ones                   100.00%        100          -    Covered
        covered/total bins:                              2          2          -
        missing/total bins:                              0          2          -
        % Hit:                                     100.00%        100          -
    Coverpoint B_cp                                100.00%        100          -    Covered
        covered/total bins:                              3          3          -
        missing/total bins:                              0          3          -
        % Hit:                                     100.00%        100          -
    Coverpoint B_cp_walking_ones                   100.00%        100          -    Covered
        covered/total bins:                              2          2          -
        missing/total bins:                              0          2          -
        % Hit:                                     100.00%        100          -
    Coverpoint ALU_cp                              100.00%        100          -    Covered
        covered/total bins:                              7          7          -
        missing/total bins:                              0          7          -
        % Hit:                                     100.00%        100          -
    Coverpoint opcode_cp                             0.00%        100          -    ZERO
        covered/total bins:                              8          8          -
        missing/total bins:                              0          8          -
        % Hit:                                     100.00%        100          -
    Coverpoint cin_cp                                0.00%        100          -    ZERO
        covered/total bins:                              2          2          -
        missing/total bins:                              0          2          -
        % Hit:                                     100.00%        100          -
    Coverpoint serialin_cp                           0.00%        100          -    ZERO
        covered/total bins:                              2          2          -
        missing/total bins:                              0          2          -
        % Hit:                                     100.00%        100          -
    Coverpoint direction_cp                          0.00%        100          -    ZERO
        covered/total bins:                              2          2          -
        missing/total bins:                              0          2          -
        % Hit:                                     100.00%        100          -
    Coverpoint red_op_A_cp                           0.00%        100          -    ZERO
        covered/total bins:                              2          2          -
        missing/total bins:                              0          2          -
        % Hit:                                     100.00%        100          -
    Coverpoint red_op_B_cp                           0.00%        100          -    ZERO
        covered/total bins:                              2          2          -
        missing/total bins:                              0          2          -
        % Hit:                                     100.00%        100          -
    Cross #cross__0#                               100.00%        100          -    Covered
        covered/total bins:                              1          1          -
        missing/total bins:                              0          1          -
        % Hit:                                     100.00%        100          -
    Cross #cross__1#                               100.00%        100          -    Covered
        covered/total bins:                              2          2          -
        missing/total bins:                              0          2          -
        % Hit:                                     100.00%        100          -
    Cross #cross__2#                               100.00%        100          -    Covered
        covered/total bins:                              2          2          -
        missing/total bins:                              0          2          -
        % Hit:                                     100.00%        100          -
    Cross #cross__3#                               100.00%        100          -    Covered
        covered/total bins:                              2          2          -
```

```
        covered/total bins:                              2          2          -
        missing/total bins:                              0          2          -
        % Hit:                                     100.00%        100          -
    Cross #cross__4#                               100.00%        100          -    Covered
        covered/total bins:                              1          1          -
        missing/total bins:                              0          1          -
        % Hit:                                     100.00%        100          -
    Cross #cross__5#                               100.00%        100          -    Covered
        covered/total bins:                              1          1          -
        missing/total bins:                              0          1          -
        % Hit:                                     100.00%        100          -
    Cross #cross__6#                               100.00%        100          -    Covered
        covered/total bins:                              1          1          -
        missing/total bins:                              0          1          -
        % Hit:                                     100.00%        100          -
Covergroup instance \/ALSU_coverage_pkg::ALSU_coverage::cg
                                                   100.00%        100          -    Covered
    covered/total bins:                                 45         45          -
    missing/total bins:                                  0         45          -
    % Hit:                                         100.00%        100          -
    Coverpoint A_cp                                100.00%        100          -    Covered
        covered/total bins:                              3          3          -
        missing/total bins:                              0          3          -
        % Hit:                                     100.00%        100          -
        bin A_data_0                                  2034          1          -    Covered
        bin A_data_max                                1645          1          -    Covered
        bin A_data_min                                1714          1          -    Covered
        default bin A_data_default                    4618                     -    Occurred
    Coverpoint A_cp_walking_ones                   100.00%        100          -    Covered
        covered/total bins:                              2          2          -
        missing/total bins:                              0          2          -
        % Hit:                                     100.00%        100          -
        bin A_data_walkingones[1]                      432          1          -    Covered
        bin A_data_walkingones[2]                      449          1          -    Covered
    Coverpoint B_cp                                100.00%        100          -    Covered
        covered/total bins:                              3          3          -
        missing/total bins:                              0          3          -
        % Hit:                                     100.00%        100          -
        bin B_data_0                                  2300          1          -    Covered
        bin B_data_max                                1770          1          -    Covered
        bin B_data_min                                1610          1          -    Covered
        default bin B_data_default                    4331                     -    Occurred
    Coverpoint B_cp_walking_ones                   100.00%        100          -    Covered
        covered/total bins:                              2          2          -
        missing/total bins:                              0          2          -
        % Hit:                                     100.00%        100          -
        bin B_data_walkingones[1]                      192          1          -    Covered
        bin B_data_walkingones[2]                      190          1          -    Covered
    Coverpoint ALU_cp                              100.00%        100          -    Covered
        covered/total bins:                              7          7          -
        missing/total bins:                              0          7          -
        % Hit:                                     100.00%        100          -
        illegal_bin Bins_invalid                       953                     -    Occurred
        bin Bins_shift[SHIFT]                         1524          1          -    Covered
        bin Bins_shift[ROTATE]                        1496          1          -    Covered
        bin Bins_arith[ADD]                           1549          1          -    Covered
        bin Bins_arith[MULT]                          1483          1          -    Covered
        bin Bins_bitwise[OR]                          1497          1          -    Covered
        bin Bins_bitwise[XOR]                         1509          1          -    Covered
        bin Bins_trans                                   1          1          -    Covered
    Coverpoint opcode_cp [1]                       100.00%        100          -    Covered
        covered/total bins:                              8          8          -
        missing/total bins:                              0          8          -
        % Hit:                                     100.00%        100          -
        bin auto[OR]                                  1497          1          -    Covered
        bin auto[XOR]                                 1509          1          -    Covered
        bin auto[ADD]                                 1549          1          -    Covered
        bin auto[MULT]                                1483          1          -    Covered
        bin auto[SHIFT]                               1524          1          -    Covered
        bin auto[ROTATE]                              1496          1          -    Covered
        bin auto[INVALID_6]                            454          1          -    Covered
        bin auto[INVALID_7]                            499          1          -    Covered
    Coverpoint cin_cp [1]                          100.00%        100          -    Covered
        covered/total bins:                              2          2          -
        missing/total bins:                              0          2          -
        % Hit:                                     100.00%        100          -
        bin auto[0]                                   5066          1          -    Covered
        bin auto[1]                                   4945          1          -    Covered
    Coverpoint serialin_cp [1]                     100.00%        100          -    Covered
        covered/total bins:                              2          2          -
        missing/total bins:                              0          2          -
        % Hit:                                     100.00%        100          -
        bin auto[0]                                   4957          1          -    Covered
        bin auto[1]                                   5054          1          -    Covered
    Coverpoint direction_cp [1]                    100.00%        100          -    Covered
        covered/total bins:                              2          2          -
        missing/total bins:                              0          2          -
        % Hit:                                     100.00%        100          -
        bin auto[0]                                   5002          1          -    Covered
        bin auto[1]                                   5009          1          -    Covered
```

```
        % Hit:                                  100.00%        100          -
        bin auto[0]                               5002          1          -    Covered
        bin auto[1]                               5009          1          -    Covered
    Coverpoint red_op_A_cp [1]                  100.00%        100          -    Covered
        covered/total bins:                           2          2          -
        missing/total bins:                           0          2          -
        % Hit:                                  100.00%        100          -
        bin auto[0]                               5695          1          -    Covered
        bin auto[1]                               4316          1          -    Covered
    Coverpoint red_op_B_cp [1]                  100.00%        100          -    Covered
        covered/total bins:                           2          2          -
        missing/total bins:                           0          2          -
        % Hit:                                  100.00%        100          -
        bin auto[0]                               5772          1          -    Covered
        bin auto[1]                               4239          1          -    Covered
    Cross #cross__0#                            100.00%        100          -    Covered
        covered/total bins:                           1          1          -
        missing/total bins:                           0          1          -
        % Hit:                                  100.00%        100          -
        Auto, Default and User Defined Bins:
            bin ADD_MULT_EXTREEMS                 2211          1          -    Covered
    Cross #cross__1#                            100.00%        100          -    Covered
        covered/total bins:                           2          2          -
        missing/total bins:                           0          2          -
        % Hit:                                  100.00%        100          -
        Auto, Default and User Defined Bins:
            bin ADD_CIN0                           796          1          -    Covered
            bin ADD_CIN1                           753          1          -    Covered
    Cross #cross__2#                            100.00%        100          -    Covered
        covered/total bins:                           2          2          -
        missing/total bins:                           0          2          -
        % Hit:                                  100.00%        100          -
        Auto, Default and User Defined Bins:
            bin SHIFT_SERIALIN0                    805          1          -    Covered
            bin SHIFT_SERIALIN1                    719          1          -    Covered
    Cross #cross__3#                            100.00%        100          -    Covered
        covered/total bins:                           2          2          -
        missing/total bins:                           0          2          -
        % Hit:                                  100.00%        100          -
        Auto, Default and User Defined Bins:
            bin SHIFT_ROTATE_DIRECTION0           1523          1          -    Covered
            bin SHIFT_ROTATE_DIRECTION1           1487          1          -    Covered
    Cross #cross__4#                            100.00%        100          -    Covered
        covered/total bins:                           1          1          -
        missing/total bins:                           0          1          -
        % Hit:                                  100.00%        100          -
        Auto, Default and User Defined Bins:
            bin OR_XOR_REDA                        376          1          -    Covered
    Cross #cross__5#                            100.00%        100          -    Covered
        covered/total bins:                           1          1          -
        missing/total bins:                           0          1          -
        % Hit:                                  100.00%        100          -
        Auto, Default and User Defined Bins:
            bin OR_XOR_REDB                        158          1          -    Covered
    Cross #cross__6#                            100.00%        100          -    Covered
        covered/total bins:                           1          1          -
        missing/total bins:                           0          1          -
        % Hit:                                  100.00%        100          -
        Auto, Default and User Defined Bins:
            bin INVALID_REDUCTION                 5313          1          -    Covered
```

Practicing UVM Factory:

Sequence item code snippets:

```
package ALSU_sequence_item_pkg;
    import uvm_pkg::*;
    import ALSU_shared_pkg::*;
    `include "uvm_macros.svh"
    class ALSU_sequence_item extends uvm_sequence_item;
        `uvm_object_utils(ALSU_sequence_item)

        function new(string name = "ALSU_sequence_item");
            super.new(name);
        endfunction

        // Signals
        rand bit rst, cin, red_op_A, red_op_B, bypass_A, bypass_B, direction, serial_in;
        rand opcode_e opcode;
        rand bit signed [2:0] A, B;
        rand logic [15:0] leds;
        rand logic signed [5:0] out;
        logic [15:0] leds_ref;
        logic signed [5:0] out_ref;
        rand opcode_e opcode_array[5:0];
        bit [2:0] EXTREEMS [2:0] = '{MAXNEG,ZERO,MAXPOS};
        bit [2:0] INTERNALS [] = {3'b101, 3'b110, 3'b111, 3'b000, 3'b011} ;//!=EXTREEMS
        bit [2:0] WALKING_ONES [2:0] = '{3'b001,3'b010,3'b100};
        bit [2:0] SLEEPING_ONES [] = {3'b000,3'b011,3'b101,3'b110,3'b111};//!= WALKING_ONES

        // Constraint blocks
        constraint rst_constraint {
            rst dist {0:/95,1:/5};//Reset to be asserted with a low probability , 95% off & 5% on
        }
        constraint opcode_e_constraint {
            opcode dist {[OR:ROTATE]:/90,[INVALID_6:INVALID_7]:/10};//Invalid cases should occur less frequent(10%) than the valid cases(90%)
        }
        constraint A_B_constraints {
            if (opcode == ADD || opcode == MULT )
            {   //A & B to take the values (MAXPOS, ZERO and MAXNEG) more often than the other values
                A dist { EXTREEMS:/75 ,INTERNALS:/25};
                B dist { EXTREEMS:/75 ,INTERNALS:/25};
            }
            else if ((opcode == OR || opcode == XOR))
            {
                if (red_op_A){///(red_op_A_package==1)
                        //A most of the time to have one bit high in its 3 bits while constraining the B to be low
                        A dist  { WALKING_ONES:/80 ,SLEEPING_ONES:/20 };
                        B == 0;
                    }
                else if(red_op_B && !red_op_A ){///(red_op_B_package==1 && red_op_A_package==0)
                        //B most of the time to have one bit high in its 3 bits while constraining the A to be low
                        B dist { WALKING_ONES:/80 ,SLEEPING_ONES:/20 };
                        A == 0;
                    }
            }
            else {
                //Do not constraint the inputs A or B when the operation is shift or rotate
                A inside {[MAXNEG:MAXPOS]};
                B inside {[MAXNEG:MAXPOS]};
```

```
        }
        constraint bypass_constraint {
            //bypass_A and bypass_B should be disabled most of the time
            bypass_A dist {0:/90,1:/10};//90% off & 10% on
            bypass_B dist {0:/90,1:/10};//90% off & 10% on
        }
        constraint unique_opcodes {
            foreach (opcode_array[i]) {
            // Ensure elements that are not manually overridden are unique
                opcode_array[i] inside {[OR:ROTATE]};
                    foreach (opcode_array[j]) {
                        if (i != j) { // Exclude overridden elements
                            opcode_array[i] != opcode_array[j];
                        }
                    }
                }
            }

        // Convert to string functions
        function string convert2string();
            return $sformatf(
            "%s rst = %0b, cin = %0b, red_op_A = %0b, red_op_B = %0b, bypass_A = %0b, bypass_B = %0b, direction = %0b, serial_in = %0b, opcode = %s, A = %0d, B = %0d, leds = 0x%0h, out = %0d",
            super.convert2string(), rst, cin, red_op_A, red_op_B, bypass_A, bypass_B, direction, serial_in, opcode.name(), A, B, leds, out);
            endfunction : convert2string


        function string convert2string_stimulus();
            return $sformatf(
            "rst = %0b, cin = %0b, red_op_A = %0b, red_op_B = %0b, bypass_A = %0b, bypass_B = %0b, direction = %0b, serial_in = %0b, opcode = %s, A = %0d, B = %0d, leds = 0x%0h, out = %0d",
            rst, cin, red_op_A, red_op_B, bypass_A, bypass_B, direction, serial_in, opcode.name(), A, B, leds, out);
            endfunction : convert2string_stimulus

    endclass : ALSU_sequence_item
```

```
    class ALSU_sequence_item_valid_invalid extends ALSU_sequence_item;
        `uvm_object_utils(ALSU_sequence_item_valid_invalid)

        function new(string name = "ALSU_sequence_item_valid_invalid");
            super.new(name);
        endfunction

        constraint opcode_e_constraint {
            opcode dist {[OR:ROTATE]}; // Valid cases only
        }
    endclass : ALSU_sequence_item_valid_invalid

endpackage : ALSU_sequence_item_pkg
```

Test code snippets:

```
package ALSU_test_pkg;
    `include "uvm_macros.svh"
    import uvm_pkg::*;
    import ALSU_env_pkg::*;
    import package_shift_reg_env_pkg::*;
    import ALSU_config_obj_pkg::*;
    import shift_reg_config_pkg::*;
    import ALSU_agent_pkg::*;
    import ALSU_main_sequence_pkg::*;
    import ALSU_reset_sequence_pkg::*;
    import ALSU_valid_sequence_pkg::*;
    import shigt_reg_main_sequence_pkg::*;
    import shift_reg_agent_pkg::*;
    import ALSU_sequence_item_pkg::*; // Correct package import

    class ALSU_test extends uvm_test ;
        `uvm_component_utils(ALSU_test)

        ALSU_config_obj ALSU_config_obj_test;
        shift_reg_config shift_reg_cfg;
        ALSU_env env;
        shift_reg_env env2;
        virtual ALSU_interface ALSU_test_vif;
        ALSU_reset_sequence reset_seq;
        ALSU_main_sequence main_seq;
        ALSU_valid_sequence valid_seq;

        function new(string name ="ALSU_test", uvm_component parent = null);
            super.new(name,parent);
        endfunction : new

        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            env = ALSU_env::type_id::create("env",this);
            env2 = shift_reg_env::type_id::create("env2",this);
            ALSU_config_obj_test = ALSU_config_obj::type_id::create("ALSU_config_obj_test");
            shift_reg_cfg = shift_reg_config::type_id::create("shift_reg_cfg");
            reset_seq = ALSU_reset_sequence::type_id::create("reset_seq");
            main_seq = ALSU_main_sequence::type_id::create("main_seq");
            valid_seq = ALSU_valid_sequence::type_id::create("valid_seq");

            if (!uvm_config_db#(virtual ALSU_interface)::get(this, "", "ALSU_IF", ALSU_config_obj_test.ALSU_config_vif)) begin
                `uvm_fatal("build_phase","Test - unable to get the virtual interface");
             end

             if (!uvm_config_db#(virtual shift_reg_if)::get(this,"","shift_reg_Vif",shift_reg_cfg.shift_reg_Vif)) begin
                `uvm_fatal("build_phase","Test - unable to get the virtual interface");
             end

            ALSU_config_obj_test.is_active=UVM_ACTIVE;
            shift_reg_cfg.is_active=UVM_PASSIVE;

            uvm_config_db #(shift_reg_config)::set(this,"*","CFG",shift_reg_cfg);
            uvm_config_db#(ALSU_config_obj)::set(this, "*", "CFG", ALSU_config_obj_test);
```

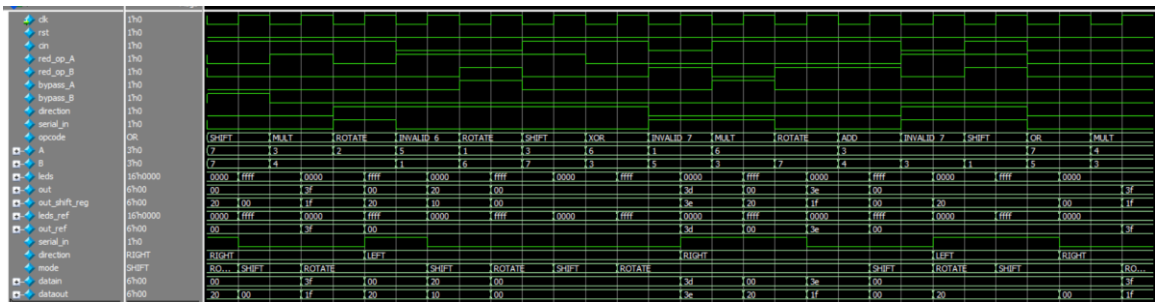
```
            set_type_override_by_type(ALSU_sequence_item::get_type(), ALSU_sequence_item_valid_invalid::get_type());

        endfunction : build_phase

        task run_phase(uvm_phase phase);
            super.run_phase(phase);
            phase.raise_objection(this);

            `uvm_info("run_phase","Reset asserted", UVM_LOW)
            reset_seq.start(env.agt.sqr);

            `uvm_info("run_phase","Stimulus generation started", UVM_LOW)
            main_seq.start(env.agt.sqr);

            `uvm_info("run_phase","Valid opcodes sequence started", UVM_LOW)
            valid_seq.start(env.agt.sqr);

            phase.drop_objection(this);
        endtask

    endclass : ALSU_test
endpackage : ALSU_test_pkg
```

Transcript snippet:

```
# UVM_INFO ALSU_scoreboard_pkg.sv(49) @ 20022: uvm_test_top.env.sb [report_phase] Total coorect counts is: 0d10011
# UVM_INFO ALSU_scoreboard_pkg.sv(50) @ 20022: uvm_test_top.env.sb [report_phase] Total error counts is: 0d0
# UVM_INFO shift_reg_scoreboard_pkg.sv(70) @ 20022: uvm_test_top.env2.sb [report_phase] Total coorect counts is: 0d10011
# UVM_INFO shift_reg_scoreboard_pkg.sv(71) @ 20022: uvm_test_top.env2.sb [report_phase] Total error counts is: 0d0
#
# --- UVM Report Summary ---
#
# ** Report counts by severity
# UVM_INFO :   11
# UVM_WARNING :    0
# UVM_ERROR :    0
# UVM_FATAL :    0
# ** Report counts by id
# [Questa UVM]     2
# [RNTST]     1
# [TEST_DONE]     1
# [report_phase]     4
# [run_phase]     3
# ** Note: $finish    : E:/DIGITAL/Programs/QuestaSim/win64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh(430)
#    Time: 20022 ns  Iteration: 61  Instance: /ALSU_top
```

Waveform snippets:

Before (Valid & Invalid cases):



After (Valid cases only):