Assignment 4

Digital Design Verification

Contents

1 Q1: ALSU

1.1 1. Testbench code

```
`timescale 1ns/1ps

import ALSU_pkg::*;

module ALSU_tb;

  // ----------------------------------
  // Testbench signals
  // ----------------------------------
  logic clk;
  logic rst;
  logic cin;
  logic red_op_A;
  logic red_op_B;
  logic bypass_A;
  logic bypass_B;
  logic direction;
  logic serial_in;
  logic signed [2:0] A;
  logic signed [2:0] B;
  logic [2:0]        opcode;
  wire [15:0]        leds;
  wire signed [5:0]  out;

  // ----------------------------------
  // DUT instantiation
  // ----------------------------------
  ALSU #(
    .INPUT_PRIORITY("A"),
    .FULL_ADDER("ON")
  ) dut (
    .clk        (clk),
    .rst        (rst),
    .cin        (cin),
    .red_op_A   (red_op_A),
    .red_op_B   (red_op_B),
    .bypass_A   (bypass_A),
    .bypass_B   (bypass_B),
    .direction  (direction),
    .serial_in  (serial_in),
    .A          (A),
    .B          (B),
    .opcode     (opcode),
    .leds       (leds),
    .out        (out)
  );

  // ----------------------------------
  // Clock & reset generation
  // ----------------------------------
  initial begin
        clk = 0;
        forever begin
           #5 clk = ~clk;
        end
   end


  // --------------------------------------
  // Create an object for random stimulus
  // --------------------------------------
  alsu_rand_class rand_stim;

// ------------
  // reset task
  // ------------
  task do_reset();
    rst = 1;
    #10;
    // Check result against a golden model
      golden_model(
        rst, A, B, cin, serial_in, red_op_A, red_op_B, bypass_A, bypass_B,
        direction, opcode
      );
    #10;
    rst = 0;
  endtask

  // ----------------------------------
  // Golden model for reference
  // ----------------------------------
task golden_model(
    input logic rst,
    input logic signed [2:0] A, B,
    input logic cin, serial_in, red_op_A, red_op_B,
    input logic bypass_A, bypass_B, direction,
    input logic [2:0] opcode
);

    logic signed [5:0] expected_out;
    logic [15:0] expected_leds;

    logic invalid_red_op, invalid_opcode, invalid;

    // Invalid condition handling
    invalid_red_op = (red_op_A | red_op_B) & (opcode[1] | opcode[2]);
    invalid_opcode = opcode[1] & opcode[2];
    invalid = invalid_red_op | invalid_opcode;

    if(rst) begin
       expected_out = 0;
       expected_leds = 0;
    end else begin
      if (invalid)
        expected_leds = ~expected_leds;
      else
        expected_leds = 0;
    end


    if (bypass_A && bypass_B)
    expected_out = ("A" == "A") ? A : B;  // INPUT_PRIORITY is "A"
    else if (bypass_A)
    expected_out = A;
    else if (bypass_B)
    expected_out = B;
    else if (invalid)
    expected_out = 0;
    else begin
    case (opcode)
        3'h0: begin // OR or Reduction OR
            if (red_op_A && red_op_B)
                expected_out = ("A" == "A") ? |A : |B;
            else if (red_op_A)
                expected_out = |A;
            else if (red_op_B)
                expected_out = |B;
            else
                expected_out = A | B;
        end
        3'h1: begin // XOR or Reduction XOR
            if (red_op_A && red_op_B)
                expected_out = ("A" == "A") ? ^A : ^B;
            else if (red_op_A)
                expected_out = ^A;
            else if (red_op_B)
                expected_out = ^B;
            else
                expected_out = A ^ B;
        end
        3'h2: expected_out = A + B; // ADD
        3'h3: expected_out = A * B; // MUL
        3'h4: begin // SHIFT
            if (direction)
                expected_out = {expected_out[4:0], serial_in};
            else
                expected_out = {serial_in, expected_out[5:1]};
        end
        3'h5: begin // ROTATE
            if (direction)
                expected_out = {expected_out[4:0], expected_out[5]};
            else
                expected_out = {expected_out[0], expected_out[5:1]};
        end
        default: expected_out = 0;
    endcase
    end

    // Wait another clock so the output is stable
    @(posedge clk);
    #1;

    if ( (out != expected_out) && (leds != expected_leds)) begin
        $error("[ALSU]␣Mismatch␣with␣golden␣model:␣opcode=%0b.␣␣out=%0d,␣expected_out=%0d",
                opcode, out, expected_out);
    end
endtask

  // Variables for simulation control
  integer i, j;

  // Testbench stimulus generation
  initial begin
    // Initialize all inputs
    rst       = 1;
    cin       = 0;
    red_op_A  = 0;
    red_op_B  = 0;
    bypass_A  = 0;
    bypass_B  = 0;
    direction = 0;
    serial_in = 0;
    opcode    = 0;
    A         = 0;
    B         = 0;

    // Instantiate the random stimulus object
    rand_stim = new();

    // Hold reset for a few clock cycles
    rand_stim.stop();

    do_reset();  // start in reset

    rand_stim.start();
    // ============================================================
    // Phase 1: Full Constrained Randomization (Constraints 1-7 enabled)
    // Disable constraint 8 here
    // ============================================================
    rand_stim.disable_constrain_8();
    $display("Phase 1: Full constrained randomization with constraints 1-7");
    for (i = 0; i < 500; i = i + 1) begin

      if (!rand_stim.randomize()) begin
           $error("Randomization failed in phase 1 at iteration %0d", i);
           $finish;
      end

      @(negedge clk);

      // Drive DUT with randomized values
      cin       = rand_stim.cin;
      red_op_A  = rand_stim.red_op_A;
      red_op_B  = rand_stim.red_op_B;
      bypass_A  = rand_stim.bypass_A;
      bypass_B  = rand_stim.bypass_B;
      direction = rand_stim.direction;
      serial_in = rand_stim.serial_in;
      opcode    = rand_stim.opcode;
      A         = rand_stim.A;
      B         = rand_stim.B;

      // Wait a clock for inputs to be sampled
      @(posedge clk);


     // Check result against a golden model
     golden_model(
        rst, A, B, cin, serial_in, red_op_A, red_op_B, bypass_A, bypass_B,
        direction, opcode
     );

      // Sample only when not in reset or bypass mode
      if (!(rst || bypass_A || bypass_B)) begin
        rand_stim.sample();
      end

    end

    $display("ALSU Phase 1 test completed");

    // ============================================================
    // Transition to Phase 2: Opcode Verification
    // ============================================================
    $display("Transitioning to Phase 2: Opcode verification");
    // Force key signals to 0 as required
    rst      = 0;
    bypass_A = 0;
    bypass_B = 0;
    red_op_A = 0;
    red_op_B = 0;

    // Disable all constraints for the random stimulus object
    // Enable constraint 8 (for unique opcode generation)
    rand_stim.disabled_all_constrains_expect_8();

    // Randomize other inputs once (without constraints)
    if (!rand_stim.randomize()) begin
      $error("Randomization without constraints failed in phase 2");
      $finish;
    end
    // Drive constant signals from the randomized object
    cin       = rand_stim.cin;
    direction = rand_stim.direction;
    serial_in = rand_stim.serial_in;
    A         = rand_stim.A;
    B         = rand_stim.B;

      // Sample only when not in reset or bypass mode
      if (!(rst || bypass_A || bypass_B)) begin
        rand_stim.sample();
      end


    // ============================================================
    // Phase 2: Nested Loop for Opcode Verification
    // ============================================================
    for (j = 0; j < 6; j = j + 1) begin

      if (!rand_stim.randomize()) begin
        $error("Randomization without constraints failed in phase 2");
        $finish;
      end

      @(negedge clk);
      opcode = rand_stim.opcode_array[j];
      rand_stim.opcode = rand_stim.opcode_array[j];

      // Drive the DUT with the current opcode while other inputs remain constant
        // Wait a clock for inputs to be sampled
        @(posedge clk);

        // Check result against a golden model
        golden_model(
           rst, A, B, cin, serial_in, red_op_A, red_op_B, bypass_A, bypass_B,
            direction, opcode
        );

      // Sample only when not in reset or bypass mode
      if (!(rst || bypass_A || bypass_B)) begin
        rand_stim.sample();
      end

    end

    // ============================================================
    // End simulation and dump coverage data if needed
    // ============================================================
    $display("Testbench␣completed.");
    // direct rst value for 0 -> 1
    @(posedge clk);
    rst       = 1;
    #20;
    $finish;
  end

endmodule
```

1.2 2. Package code

```
package ALSU_pkg;
    // ------------------------------------------
    // 1) Define the opcodes (including invalids)
    // ------------------------------------------
    typedef enum logic [2:0] {
        OR_0       = 3'h0,  // 000
        XOR_1      = 3'h1,  // 001
        ADD_2      = 3'h2,  // 010
        MUL_3      = 3'h3,  // 011
        SHIFT_4    = 3'h4,  // 100
        ROTATE_5   = 3'h5,  // 101
        INVALID_6  = 3'h6,  // 110
        INVALID_7  = 3'h7   // 111
    } opcode_e;

    // ------------------------------------------
    // 2) 3-bit signed range is -4 .. +3
    // ------------------------------------------
    localparam logic signed [2:0] MAXNEG = -4;  // 3'b100
    localparam logic signed [2:0] ZERO   =  0;
    localparam logic signed [2:0] MAXPOS =  3;  // 3'b011

    class alsu_rand_class;
        // ------------------------------
        // Randomizable DUT inputs
        // ------------------------------
        bit                     clk;
        rand bit                rst;
        rand bit                cin;
        rand bit                red_op_A;
        rand bit                red_op_B;
        rand bit                bypass_A;
        rand bit                bypass_B;
        rand bit                direction;
        rand bit                serial_in;
        rand opcode_e           opcode;
        rand logic signed [2:0] A;
        rand logic signed [2:0] B;
        rand opcode_e           opcode_array[6];

        bit disable_all = 0;
        bit disable_8 = 0;


        // ------------------------------
        // Constraints from specification
        // ------------------------------

        // (a) Make RESET happen with a low probability
        constraint c_reset_low_prob {
            if (!disable_all) {
                rst dist { 0 := 95, 1 := 5 };
            }
        }

        // (b) For ADD or MUL, pick corner values of A,B more often
        //     (MAXNEG, ZERO, MAXPOS) than the other possibilities.
        //     Weighted distribution is used here.
        constraint c_adder_mult_corner {
            if (!disable_all) {
                if (opcode inside {ADD_2, MUL_3}) {
                  A dist { MAXNEG := 3, ZERO := 3, MAXPOS := 3, [-3:-1] := 1, [1:2] := 1 };
                  B dist { MAXNEG := 3, ZERO := 3, MAXPOS := 3, [-3:-1] := 1, [1:2] := 1 };
                }
            }
        }


        // (c) If opcode=OR or XOR and red_op_A=1, then A has exactly one bit set
        //     and B is 0 .
        constraint c_red_opA_onebit {
            if (!disable_all) {
                if ((opcode==OR_0 || opcode==XOR_1) && red_op_A==1) {
                  // Force B to be 0 or near 0
                  B == 0;
                  // A has exactly 1 bit set in its 3 bits:
                  A dist {1:=3, 2:=3, MAXNEG:=3 , MAXPOS := 1 , [-3:0] := 1};
                }
            }
        }

        // (d) Similarly, if opcode=OR or XOR and red_op_B=1, then B has exactly one bit set
        //     and A is 0.
        constraint c_red_opB_onebit {
            if (!disable_all) {
                if ((opcode==OR_0 || opcode==XOR_1) && red_op_B==1) {
                  A == 0;
                  B dist {1:=3, 2:=3, MAXNEG:=3 , MAXPOS := 1 , [-3:0]:= 1 };
                }
            }
        }

        // (e) Invalid cases (opcode=6 or 7, or red_op_X=1 for non-OR/XOR)
        //     should occur *less* frequently.
        //     Weighted distribution on opcode:
        constraint c_opcode_distribution {
            if (!disable_all) {
                opcode dist {
                  INVALID_6  := 1,
                  INVALID_7  := 1,
                  OR_0       := 5,
                  XOR_1      := 5,
                  ADD_2      := 5,
                  MUL_3      := 5,
                  SHIFT_4    := 5,
                  ROTATE_5   := 5
                };
            }
        }

        // (f) For red_op_A/B, require them to be 0 if opcode in {ADD_2, MUL_3, SHIFT_4, ROTATE_5}
        // except for a small chance to produce the invalid scenario:
        constraint c_red_op_non_orxor {
            if (!disable_all) {
                if (opcode inside {ADD_2, MUL_3, SHIFT_4, ROTATE_5}) {
                  (red_op_A == 0) dist {0:=95, 1:=5};
                  (red_op_B == 0) dist {0:=95, 1:=5};
                }
            }
        }

        constraint c_red_op_orxor {
            if (!disable_all) {
                if (opcode inside {XOR_1, OR_0}) {
                       {red_op_A, red_op_B} dist {2'b00 :/ 5, 2'b01 :/ 10, 2'b10 :/ 10 ,2'b11 :/ 75};
                }
            }
        }

        // (g) bypass_A and bypass_B should be disabled most of the time
        constraint c_bypass_dist {
            if (!disable_all) {
              bypass_A dist {0:=3,1:=1};
              bypass_B dist {0:=3,1:=1};
            }
        }

        // (h) If SHIFT or ROTATE, do not constrain A,B.
        //     (No explicit constraint needed => they can be anything.)


        // -------------------------------
        // Constraint 8
        // -------------------------------
        constraint c_opcode_unique {
            if (!disable_8) {
              foreach (opcode_array[i]) {
                opcode_array[i] == i;
              }
            }
        }

        // -----------------------------------
        // Functional Coverage
        // -----------------------------------
        covergroup cg;
          coverpoint rst;
          coverpoint cin;
          red_op_A_cp: coverpoint red_op_A{
            bins red_op_A_0 = {0};
            bins red_op_A_1 = {1};
            bins red_op_A_default = default;
          }
          red_op_B_cp: coverpoint red_op_B{
            bins red_op_B_0 = {0};
            bins red_op_B_1 = {1};
            bins red_op_B_default = default;
          }
          coverpoint bypass_A;
          coverpoint bypass_B;
          coverpoint direction;
          coverpoint serial_in;
          A_cp: coverpoint A {
            bins A_data_0 = {0};
            bins A_data_max = {MAXPOS};
            bins A_data_min = {MAXNEG};
            bins A_data_default = default;
            bins A_data_walkingones[] = {3'b001, 3'b010, 3'b100};
          }
          B_cp: coverpoint B {
            bins B_data_0 = {0};
            bins B_data_max = {MAXPOS};
            bins B_data_min = {MAXNEG};
            bins B_data_default = default;
            bins B_data_walkingones[] = {3'b001, 3'b010, 3'b100};
          }
          A_cp_mod: coverpoint A {
            bins A_data_0 = {0};
            bins A_data_max = {MAXPOS};
            bins A_data_min = {MAXNEG};
          }
          B_cp_mod: coverpoint B {
            bins B_data_0 = {0};
            bins B_data_max = {MAXPOS};
            bins B_data_min = {MAXNEG};
          }
          ALU_cp: coverpoint opcode {
            bins Bins_shift[] = {SHIFT_4, ROTATE_5};
            bins Bins_arith[] = {ADD_2, MUL_3};
            bins Bins_bitwise[] = {OR_0, XOR_1};
            bins Bins_invalid = {INVALID_6, INVALID_7};
            bins Bins_trans = (OR_0 => XOR_1 => ADD_2 => MUL_3 => SHIFT_4 => ROTATE_5);
          }

          cross A_cp , red_op_A_cp{
           ignore_bins assert_red_op_A = binsof(A_cp.A_data_walkingones) && binsof(red_op_A_cp.red_op_A_1);
          }

          cross B_cp , red_op_A_cp , red_op_B_cp{
           ignore_bins assert_red_op_B = binsof(B_cp.B_data_walkingones) && binsof(red_op_A_cp.red_op_A_0) &&
              binsof(red_op_B_cp.red_op_B_1) intersect {1};
          }

          cross A_cp_mod, B_cp_mod, ALU_cp {
            bins arith_permutations = binsof(ALU_cp.Bins_arith) && binsof(A_cp_mod) && binsof(B_cp_mod);
            option.cross_auto_bin_max=0;
          }

          cross cin, ALU_cp {
            bins addition_cin = binsof(ALU_cp.Bins_arith) intersect {ADD_2} && binsof(cin);
            option.cross_auto_bin_max=0;
          }

          cross direction, ALU_cp {
            bins shift_rotate_direction = binsof(ALU_cp.Bins_shift) && binsof(direction);
            option.cross_auto_bin_max=0;
          }

          cross serial_in, ALU_cp {
            bins shift_serial_in = binsof(ALU_cp.Bins_shift)intersect {SHIFT_4} && binsof(serial_in);
            option.cross_auto_bin_max=0;
          }

          cross A_cp, red_op_A_cp, B_cp, ALU_cp {
            bins or_xor_red_op_A = binsof(ALU_cp.Bins_bitwise) && binsof(red_op_A_cp.red_op_A_1) && binsof(A_cp.A_data_walkingones) &&
               binsof(B_cp.B_data_0);
            option.cross_auto_bin_max=0;
          }

          cross B_cp, red_op_B_cp, A_cp, ALU_cp {
            bins or_xor_red_op_B = binsof(ALU_cp.Bins_bitwise) && binsof(red_op_B_cp.red_op_B_1) && binsof(B_cp.B_data_walkingones) &&
               binsof(A_cp.A_data_0);
            option.cross_auto_bin_max=0;
          }

          cross ALU_cp, red_op_A_cp, red_op_B_cp {
            ignore_bins invalid_reduction = binsof(ALU_cp.Bins_bitwise) && (binsof(red_op_A_cp.red_op_A_1) ||
               binsof(red_op_B_cp.red_op_B_1));
          }


        endgroup

        // constructor
        function new();
          cg = new();
        endfunction

        function void stop();
            cg.stop();
        endfunction

        function void start();
           cg.start();
        endfunction

        function void sample();
            cg.sample();
        endfunction

        // disabled all constrains expect 8
        function void disabled_all_constrains_expect_8();
          disable_all = 1;
          disable_8 = 0 ;
        endfunction
```

```
        // disable constrain 8
        function void disable_constrain_8();
          disable_all = 0;
          disable_8 = 1 ;
        endfunction

        // enable all constrains
        function void enable_all_constrains();
          disable_all = 0;
          disable_8 = 0 ;
        endfunction


    endclass


endpackage
```

1.3 3. Design code

```
module ALSU(A, B, cin, serial_in, red_op_A, red_op_B, opcode, bypass_A, bypass_B, clk, rst, direction, leds, out);
parameter INPUT_PRIORITY = "A";
parameter FULL_ADDER = "ON";
input clk, cin, rst, red_op_A, red_op_B, bypass_A, bypass_B, direction, serial_in;
input [2:0] opcode;
input signed [2:0] A, B;
output reg [15:0] leds;
output reg signed [5:0] out;

reg red_op_A_reg, red_op_B_reg, bypass_A_reg, bypass_B_reg, direction_reg, serial_in_reg;
reg signed cin_reg;
reg [2:0] opcode_reg;
reg signed [2:0] A_reg, B_reg;

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 clk or posedge rst) begin
  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 <= cin;
     red_op_B_reg <= red_op_B;
     red_op_A_reg <= red_op_A;
     bypass_B_reg <= bypass_B;
     bypass_A_reg <= bypass_A;
     direction_reg <= direction;
     serial_in_reg <= serial_in;
     opcode_reg <= opcode;
     A_reg <= A;
     B_reg <= B;
  end
end

//leds output blinking
always @(posedge clk or posedge rst) begin
  if(rst) begin
     leds <= 0;
  end else begin
      if (invalid)
        leds <= ~leds;
      else
        leds <= 0;
  end
end

//ALSU output processing
always @(posedge clk or posedge rst) begin
  if(rst) begin
    out <= 0;
  end
  else begin
    if (bypass_A_reg && bypass_B_reg)
      out <= (INPUT_PRIORITY == "A")? A_reg: B_reg;
    else if (bypass_A_reg)
      out <= A_reg;
    else if (bypass_B_reg)
      out <= B_reg;
    else if (invalid)
        out <= 0;
    else begin
        case (opcode)
          3'h0: begin
            if (red_op_A_reg && red_op_B_reg)
              out <= (INPUT_PRIORITY == "A")? |A_reg: |B_reg;
            else if (red_op_A_reg)
              out <= |A_reg;
            else if (red_op_B_reg)
              out <= |B_reg;
            else
              out <= A_reg | B_reg;
          end
          3'h1: begin
            if (red_op_A_reg && red_op_B_reg)
              out <= (INPUT_PRIORITY == "A")? ^A_reg: ^B_reg;
            else if (red_op_A_reg)
              out <= ^A_reg;
            else if (red_op_B_reg)
              out <= ^B_reg;
            else
              out <= A_reg ^ B_reg;
          end
          3'h2: out <= A_reg + B_reg;
          3'h3: out <= A_reg * B_reg;
          3'h4: begin
            if (direction_reg)
              out <= {out[4:0], serial_in_reg};
            else
              out <= {serial_in_reg, out[5:1]};
          end
          3'h5: begin
            if (direction_reg)
              out <= {out[4:0], out[5]};
            else
              out <= {out[0], out[5:1]};
          end
        endcase
    end
  end
end

endmodule
```

1.4 4. Bug Fixes

```
no bugs except cin_reg is one bit not two bits
```

1.5 5. Verification Plan

Label	Description	Stimulus Generation	Functional Coverage	Functionality Check
ALSU_1	When the reset is asserted, the outputs should be low.	• Directed reset is applied at simulation start via the `do_reset()` task. • Afterwards, the reset signal is randomized with the constraint `c_reset_low_prob` (5% chance of `rst`=1).	• A covergroup in `alsu_rand_class` monitors transitions on `rst`. • Coverage bins ensure that enough reset assertions are observed during randomization.	• The golden model is invoked during reset to verify that both `out` and `leds` are 0. • Any deviation (non-zero outputs when `rst` is high) flags an error.
ALSU_2	In the absence of invalid conditions, when the opcode is ADD, the output should perform addition on ports A and B, incorporating `cin` if `FULL_ADDER` is enabled.	• The random stimulus is generated with the weighted constraint `c_opcode_distribution` to ensure frequent selection of ADD (3'h2). • Inputs A and B are randomized using `c_adder_mult_corner` to emphasize corner cases (values `MAXNEG`, `ZERO`, `MAXPOS`). • Reduction control signals (`red_op_A` and `red_op_B`) are mostly deasserted to avoid invalid conditions.	• The covergroup within `alsu_rand_class` collects data on opcode, A, B, and `cin` along with other control signals. • Specific bins track the occurrence of corner-case operand values and the frequency of the ADD opcode.	• After each randomized transaction, the `golden_model()` task computes the expected output (i.e. `A+B` plus `cin` when relevant). • The testbench compares the DUT output against the golden model; any mismatch in the computed sum flags an error.

Table 1: Verification Plan

1.6 6. Do File

```
vlib work
vlog ALSU.sv ALSU_tb.sv ALSU_pkg.sv +cover -covercells
vsim -voptargs=+acc work.ALSU_tb -cover
add wave *
coverage save ALSU_tb.ucdb -onexit
run -all

# to run do file
#-- do run.txt
#to execute coverage report
#--vcover report ALSU_tb.ucdb -details -annotate -all -output code_coverage_rpt.txt -du=ALSU
#--vcover report -details -cvg -output functional_coverage_report.txt ALSU_tb.ucdb
```

1.7 7. functional Coverage Report

```
Coverage Report by instance with details
```

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

Covergroup Coverage:
    Covergroups                      1        na        na    99.12%
        Coverpoints/Crosses         22        na        na        na
            Covergroup Bins         98        93         5    94.89%
----------------------------------------------------------------------------------------------------
Covergroup                                            Metric       Goal       Bins    Status

----------------------------------------------------------------------------------------------------
 TYPE /ALSU_pkg/alsu_rand_class/cg                    99.12%        100          -    Uncovered
    covered/total bins:                                   93         98          -
    missing/total bins:                                    5         98          -
    % Hit:                                            94.89%        100          -
    Coverpoint rst                                   100.00%        100          -    Covered
        covered/total bins:                                2          2          -
        missing/total bins:                                0          2          -
        % Hit:                                       100.00%        100          -
    Coverpoint cin                                   100.00%        100          -    Covered
        covered/total bins:                                2          2          -
        missing/total bins:                                0          2          -
        % Hit:                                       100.00%        100          -
    Coverpoint red_op_A_cp                           100.00%        100          -    Covered
        covered/total bins:                                2          2          -
        missing/total bins:                                0          2          -
        % Hit:                                       100.00%        100          -
    Coverpoint red_op_B_cp                           100.00%        100          -    Covered
        covered/total bins:                                2          2          -
        missing/total bins:                                0          2          -
        % Hit:                                       100.00%        100          -
    Coverpoint bypass_A                              100.00%        100          -    Covered
        covered/total bins:                                2          2          -
        missing/total bins:                                0          2          -
        % Hit:                                       100.00%        100          -
    Coverpoint bypass_B                              100.00%        100          -    Covered
        covered/total bins:                                2          2          -
        missing/total bins:                                0          2          -
        % Hit:                                       100.00%        100          -
    Coverpoint direction                             100.00%        100          -    Covered
        covered/total bins:                                2          2          -
        missing/total bins:                                0          2          -
        % Hit:                                       100.00%        100          -
    Coverpoint serial_in                             100.00%        100          -    Covered
        covered/total bins:                                2          2          -
        missing/total bins:                                0          2          -
        % Hit:                                       100.00%        100          -
    Coverpoint A_cp                                  100.00%        100          -    Covered
        covered/total bins:                                5          5          -
        missing/total bins:                                0          5          -
        % Hit:                                       100.00%        100          -
    Coverpoint B_cp                                  100.00%        100          -    Covered
        covered/total bins:                                5          5          -
        missing/total bins:                                0          5          -
        % Hit:                                       100.00%        100          -
    Coverpoint A_cp_mod                              100.00%        100          -    Covered
        covered/total bins:                                3          3          -
        missing/total bins:                                0          3          -
        % Hit:                                       100.00%        100          -
    Coverpoint B_cp_mod                              100.00%        100          -    Covered
        covered/total bins:                                3          3          -
        missing/total bins:                                0          3          -
        % Hit:                                       100.00%        100          -
    Coverpoint ALU_cp                                100.00%        100          -    Covered
        covered/total bins:                                8          8          -
        missing/total bins:                                0          8          -
        % Hit:                                       100.00%        100          -
    Cross #cross__0#                                 100.00%        100          -    Covered
        covered/total bins:                                8          8          -
        missing/total bins:                                0          8          -
        % Hit:                                       100.00%        100          -
    Cross #cross__1#                                 100.00%        100          -    Covered
        covered/total bins:                               18         18          -
        missing/total bins:                                0         18          -
        % Hit:                                       100.00%        100          -
    Cross #cross__2#                                 100.00%        100          -    Covered
        covered/total bins:                                1          1          -
        missing/total bins:                                0          1          -
        % Hit:                                       100.00%        100          -
    Cross #cross__3#                                 100.00%        100          -    Covered
        covered/total bins:                                1          1          -
        missing/total bins:                                0          1          -
        % 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          -
    Cross #cross__7#                                     100.00%        100          -     Covered
        covered/total bins:                                    1          1          -
        missing/total bins:                                    0          1          -
        % Hit:                                           100.00%        100          -
    Cross #cross__8#                                      80.76%        100          -     Uncovered
        covered/total bins:                                   21         26          -
        missing/total bins:                                    5         26          -
        % Hit:                                            80.76%        100          -
Covergroup instance \/ALSU_pkg::alsu_rand_class::cg
                                                          99.12%        100          -     Uncovered
    covered/total bins:                                       93         98          -
    missing/total bins:                                        5         98          -
    % Hit:                                                94.89%        100          -
    Coverpoint rst                                       100.00%        100          -     Covered
        covered/total bins:                                    2          2          -
        missing/total bins:                                    0          2          -
        % Hit:                                           100.00%        100          -
        bin auto[0]                                          268          1          -     Covered
        bin auto[1]                                           18          1          -     Covered
    Coverpoint cin                                       100.00%        100          -     Covered
        covered/total bins:                                    2          2          -
        missing/total bins:                                    0          2          -
        % Hit:                                           100.00%        100          -
        bin auto[0]                                          142          1          -     Covered
        bin auto[1]                                          144          1          -     Covered
    Coverpoint red_op_A_cp                               100.00%        100          -     Covered
        covered/total bins:                                    2          2          -
        missing/total bins:                                    0          2          -
        % Hit:                                           100.00%        100          -
        bin red_op_A_0                                        55          1          -     Covered
        bin red_op_A_1                                       231          1          -     Covered
        default bin red_op_A_default                           0                     -     ZERO
    Coverpoint red_op_B_cp                               100.00%        100          -     Covered
        covered/total bins:                                    2          2          -
        missing/total bins:                                    0          2          -
        % Hit:                                           100.00%        100          -
        bin red_op_B_0                                        47          1          -     Covered
        bin red_op_B_1                                       239          1          -     Covered
        default bin red_op_B_default                           0                     -     ZERO
    Coverpoint bypass_A                                  100.00%        100          -     Covered
        covered/total bins:                                    2          2          -
        missing/total bins:                                    0          2          -
        % Hit:                                           100.00%        100          -
        bin auto[0]                                          284          1          -     Covered
        bin auto[1]                                            2          1          -     Covered
    Coverpoint bypass_B                                  100.00%        100          -     Covered
        covered/total bins:                                    2          2          -
        missing/total bins:                                    0          2          -
        % Hit:                                           100.00%        100          -
        bin auto[0]                                          281          1          -     Covered
        bin auto[1]                                            5          1          -     Covered
    Coverpoint direction                                 100.00%        100          -     Covered
        covered/total bins:                                    2          2          -
        missing/total bins:                                    0          2          -
        % Hit:                                           100.00%        100          -
        bin auto[0]                                          146          1          -     Covered
        bin auto[1]                                          140          1          -     Covered
    Coverpoint serial_in                                 100.00%        100          -     Covered
        covered/total bins:                                    2          2          -
        missing/total bins:                                    0          2          -
        % Hit:                                           100.00%        100          -
        bin auto[0]                                          149          1          -     Covered
        bin auto[1]                                          137          1          -     Covered
    Coverpoint A_cp                                      100.00%        100          -     Covered
        covered/total bins:                                    5          5          -
        missing/total bins:                                    0          5          -
        % Hit:                                           100.00%        100          -
        bin A_data_0                                          74          1          -     Covered
        bin A_data_max                                        37          1          -     Covered
        bin A_data_min                                        37          1          -     Covered
        bin A_data_walkingones[1]                             31          1          -     Covered
        bin A_data_walkingones[2]                             32          1          -     Covered
        default bin A_data_default                            75                     -     Occurred
    Coverpoint B_cp                                      100.00%        100          -     Covered
        covered/total bins:                                    5          5          -
        missing/total bins:                                    0          5          -
        % Hit:                                           100.00%        100          -
        bin B_data_0                                          53          1          -     Covered
        bin B_data_max                                        41          1          -     Covered
        bin B_data_min                                        44          1          -     Covered
        bin B_data_walkingones[1]                             38          1          -     Covered
        bin B_data_walkingones[2]                             33          1          -     Covered
        default bin B_data_default                            77                     -     Occurred
    Coverpoint A_cp_mod                                  100.00%        100          -     Covered
        covered/total bins:                                    3          3          -
        missing/total bins:                                    0          3          -
        % Hit:                                           100.00%        100          -
        bin A_data_0                                          74          1          –     Covered
        bin A_data_max                                        37          1          –     Covered
        bin A_data_min                                        37          1          –     Covered
    Coverpoint B_cp_mod                                  100.00%        100          –     Covered
        covered/total bins:                                    3          3          –
        missing/total bins:                                    0          3          –
        % Hit:                                           100.00%        100          –
        bin B_data_0                                          53          1          –     Covered
        bin B_data_max                                        41          1          –     Covered
        bin B_data_min                                        44          1          –     Covered
    Coverpoint ALU_cp                                    100.00%        100          –     Covered
        covered/total bins:                                    8          8          –
        missing/total bins:                                    0          8          –
        % Hit:                                           100.00%        100          –
        bin Bins_shift[SHIFT_4]                               47          1          –     Covered
        bin Bins_shift[ROTATE_5]                              61          1          –     Covered
        bin Bins_arith[ADD_2]                                 49          1          –     Covered
        bin Bins_arith[MUL_3]                                 53          1          –     Covered
        bin Bins_bitwise[OR_0]                                29          1          –     Covered
        bin Bins_bitwise[XOR_1]                               33          1          –     Covered
        bin Bins_invalid                                      14          1          –     Covered
        bin Bins_trans                                         1          1          –     Covered
    Cross #cross__0#                                     100.00%        100          –     Covered
        covered/total bins:                                    8          8          –
        missing/total bins:                                    0          8          –
        % Hit:                                           100.00%        100          –
        Auto, Default and User Defined Bins:
            bin <A_data_walkingones[2],red_op_A_0>             4          1          –     Covered
            bin <A_data_walkingones[1],red_op_A_0>             3          1          –     Covered
            bin <A_data_min,red_op_A_1>                       31          1          –     Covered
            bin <A_data_min,red_op_A_0>                        6          1          –     Covered
            bin <A_data_max,red_op_A_1>                       34          1          –     Covered
            bin <A_data_0,red_op_A_1>                         45          1          –     Covered
            bin <A_data_max,red_op_A_0>                        3          1          –     Covered
            bin <A_data_0,red_op_A_0>                         29          1          –     Covered
        Illegal and Ignore Bins:
            ignore_bin assert_red_op_A                        56                     –     Occurred
    Cross #cross__1#                                     100.00%        100          –     Covered
        covered/total bins:                                   18         18          –
        missing/total bins:                                    0         18          –
        % Hit:                                           100.00%        100          –
        Auto, Default and User Defined Bins:
            bin <B_data_walkingones[2],red_op_A_1,red_op_B_1>
                                                              22          1          –     Covered
            bin <B_data_walkingones[2],red_op_A_1,red_op_B_0>
                                                               2          1          –     Covered
            bin <B_data_walkingones[1],red_op_A_1,red_op_B_1>
                                                              25          1          –     Covered
            bin <B_data_walkingones[1],red_op_A_1,red_op_B_0>
                                                               2          1          –     Covered
            bin <B_data_min,red_op_A_1,red_op_B_1>            31          1          –     Covered
            bin <B_data_min,red_op_A_1,red_op_B_0>             2          1          –     Covered
            bin <B_data_max,red_op_A_1,red_op_B_1>            37          1          –     Covered
            bin <B_data_0,red_op_A_1,red_op_B_1>              35          1          –     Covered
            bin <B_data_max,red_op_A_1,red_op_B_0>             1          1          –     Covered
            bin <B_data_0,red_op_A_1,red_op_B_0>              14          1          –     Covered
            bin <B_data_walkingones[2],red_op_A_0,red_op_B_0>
                                                               2          1          –     Covered
            bin <B_data_walkingones[1],red_op_A_0,red_op_B_0>
                                                               3          1          –     Covered
            bin <B_data_min,red_op_A_0,red_op_B_1>             8          1          –     Covered
            bin <B_data_min,red_op_A_0,red_op_B_0>             3          1          –     Covered
            bin <B_data_max,red_op_A_0,red_op_B_1>             2          1          –     Covered
            bin <B_data_0,red_op_A_0,red_op_B_1>               2          1          –     Covered
            bin <B_data_max,red_op_A_0,red_op_B_0>             1          1          –     Covered
            bin <B_data_0,red_op_A_0,red_op_B_0>               2          1          –     Covered
        Illegal and Ignore Bins:
            ignore_bin assert_red_op_B                        15                     –     Occurred
    Cross #cross__2#                                     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 arith_permutations                            29          1          –     Covered
    Cross #cross__3#                                     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 addition_cin                                  49          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 shift_rotate_direction                       108          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 shift_serial_in                               47          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 or_xor_red_op_A                         5         1         -     Covered
        Cross #cross__7#                              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_red_op_B                        11         1         -     Covered
        Cross #cross__8#                               80.76%       100         -     Uncovered
            covered/total bins:                            21        26         -
            missing/total bins:                             5        26         -
            % Hit:                                     80.76%       100         -
            Auto, Default and User Defined Bins:
                bin <Bins_trans,red_op_A_1,red_op_B_0>      1         1         -     Covered
                bin <Bins_invalid,red_op_A_1,red_op_B_1>
                                                            3         1         -     Covered
                bin <Bins_invalid,red_op_A_1,red_op_B_0>
                                                            4         1         -     Covered
                bin <Bins_invalid,red_op_A_0,red_op_B_1>
                                                            4         1         -     Covered
                bin <Bins_invalid,red_op_A_0,red_op_B_0>
                                                            3         1         -     Covered
                bin <Bins_bitwise[XOR_1],red_op_A_0,red_op_B_0>
                                                            7         1         -     Covered
                bin <Bins_bitwise[OR_0],red_op_A_0,red_op_B_0>
                                                            8         1         -     Covered
                bin <Bins_arith[MUL_3],red_op_A_1,red_op_B_1>
                                                           47         1         -     Covered
                bin <Bins_shift[ROTATE_5],red_op_A_1,red_op_B_1>
                                                           56         1         -     Covered
                bin <Bins_arith[MUL_3],red_op_A_1,red_op_B_0>
                                                            4         1         -     Covered
                bin <Bins_shift[ROTATE_5],red_op_A_1,red_op_B_0>
                                                            4         1         -     Covered
                bin <Bins_arith[ADD_2],red_op_A_1,red_op_B_1>
                                                           44         1         -     Covered
                bin <Bins_shift[SHIFT_4],red_op_A_1,red_op_B_1>
                                                           43         1         -     Covered
                bin <Bins_arith[ADD_2],red_op_A_1,red_op_B_0>
                                                            1         1         -     Covered
                bin <Bins_shift[SHIFT_4],red_op_A_1,red_op_B_0>
                                                            2         1         -     Covered
                bin <Bins_arith[MUL_3],red_op_A_0,red_op_B_1>
                                                            1         1         -     Covered
                bin <Bins_shift[ROTATE_5],red_op_A_0,red_op_B_1>
                                                            1         1         -     Covered
                bin <Bins_arith[MUL_3],red_op_A_0,red_op_B_0>
                                                            1         1         -     Covered
                bin <Bins_arith[ADD_2],red_op_A_0,red_op_B_1>
                                                            4         1         -     Covered
                bin <Bins_shift[SHIFT_4],red_op_A_0,red_op_B_1>
                                                            1         1         -     Covered
                bin <Bins_shift[SHIFT_4],red_op_A_0,red_op_B_0>
                                                            1         1         -     Covered
                bin <Bins_trans,red_op_A_0,*>               0         1         2     ZERO
                bin <Bins_trans,*,red_op_B_1>               0         1         2     ZERO
                bin <Bins_shift[ROTATE_5],red_op_A_0,red_op_B_0>
                                                            0         1         1     ZERO
                bin <Bins_arith[ADD_2],red_op_A_0,red_op_B_0>
                                                            0         1         1     ZERO
            Illegal and Ignore Bins:
                ignore_bin invalid_reduction               47                   -     Occurred

    COVERGROUP COVERAGE:
    ----------------------------------------------------------------------------------------------
    Covergroup                                         Metric      Goal      Bins     Status

    ----------------------------------------------------------------------------------------------
     TYPE /ALSU_pkg/alsu_rand_class/cg                 99.12%       100         -     Uncovered
        covered/total bins:                                93        98         -
        missing/total bins:                                 5        98         -
        % Hit:                                         94.89%       100         -
        Coverpoint rst                                100.00%       100         -     Covered
            covered/total bins:                             2         2         -
            missing/total bins:                             0         2         -
            % Hit:                                    100.00%       100         -
        Coverpoint cin                                100.00%       100         -     Covered
            covered/total bins:                             2         2         -
            missing/total bins:                             0         2         -
            % Hit:                                    100.00%       100         -
        Coverpoint red_op_A_cp                        100.00%       100         -     Covered
            covered/total bins:                             2         2         -
            missing/total bins:                             0         2         -
            % Hit:                                    100.00%       100         -
        Coverpoint red_op_B_cp                        100.00%       100         -     Covered
            covered/total bins:                             2         2         -
            missing/total bins:                             0         2         -
            % Hit:                                    100.00%       100         -
        Coverpoint bypass_A                              100.00%        100         -    Covered
            covered/total bins:                                2          2         -
            missing/total bins:                                0          2         -
            % Hit:                                       100.00%        100         -
        Coverpoint bypass_B                              100.00%        100         -    Covered
            covered/total bins:                                2          2         -
            missing/total bins:                                0          2         -
            % Hit:                                       100.00%        100         -
        Coverpoint direction                             100.00%        100         -    Covered
            covered/total bins:                                2          2         -
            missing/total bins:                                0          2         -
            % Hit:                                       100.00%        100         -
        Coverpoint serial_in                             100.00%        100         -    Covered
            covered/total bins:                                2          2         -
            missing/total bins:                                0          2         -
            % Hit:                                       100.00%        100         -
        Coverpoint A_cp                                  100.00%        100         -    Covered
            covered/total bins:                                5          5         -
            missing/total bins:                                0          5         -
            % Hit:                                       100.00%        100         -
        Coverpoint B_cp                                  100.00%        100         -    Covered
            covered/total bins:                                5          5         -
            missing/total bins:                                0          5         -
            % Hit:                                       100.00%        100         -
        Coverpoint A_cp_mod                              100.00%        100         -    Covered
            covered/total bins:                                3          3         -
            missing/total bins:                                0          3         -
            % Hit:                                       100.00%        100         -
        Coverpoint B_cp_mod                              100.00%        100         -    Covered
            covered/total bins:                                3          3         -
            missing/total bins:                                0          3         -
            % Hit:                                       100.00%        100         -
        Coverpoint ALU_cp                                100.00%        100         -    Covered
            covered/total bins:                                8          8         -
            missing/total bins:                                0          8         -
            % Hit:                                       100.00%        100         -
        Cross #cross__0#                                 100.00%        100         -    Covered
            covered/total bins:                                8          8         -
            missing/total bins:                                0          8         -
            % Hit:                                       100.00%        100         -
        Cross #cross__1#                                 100.00%        100         -    Covered
            covered/total bins:                               18         18         -
            missing/total bins:                                0         18         -
            % Hit:                                       100.00%        100         -
        Cross #cross__2#                                 100.00%        100         -    Covered
            covered/total bins:                                1          1         -
            missing/total bins:                                0          1         -
            % Hit:                                       100.00%        100         -
        Cross #cross__3#                                 100.00%        100         -    Covered
            covered/total bins:                                1          1         -
            missing/total bins:                                0          1         -
            % 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         -
        Cross #cross__7#                                 100.00%        100         -    Covered
            covered/total bins:                                1          1         -
            missing/total bins:                                0          1         -
            % Hit:                                       100.00%        100         -
        Cross #cross__8#                                  80.76%        100         -    Uncovered
            covered/total bins:                               21         26         -
            missing/total bins:                                5         26         -
            % Hit:                                        80.76%        100         -
     Covergroup instance \/ALSU_pkg::alsu_rand_class::cg
                                                          99.12%        100         -    Uncovered
        covered/total bins:                                   93         98         -
        missing/total bins:                                    5         98         -
        % Hit:                                            94.89%        100         -
        Coverpoint rst                                   100.00%        100         -    Covered
            covered/total bins:                                2          2         -
            missing/total bins:                                0          2         -
            % Hit:                                       100.00%        100         -
            bin auto[0]                                      268          1         -    Covered
            bin auto[1]                                       18          1         -    Covered
        Coverpoint cin                                   100.00%        100         -    Covered
            covered/total bins:                                2          2         -
            missing/total bins:                                0          2         -
            % Hit:                                       100.00%        100         -
            bin auto[0]                                      142          1         -    Covered
            bin auto[1]                                      144          1         -    Covered
        Coverpoint red_op_A_cp                           100.00%        100         -    Covered
            covered/total bins:                                2          2         -
            missing/total bins:                                0          2         -
        % Hit:                                       100.00%        100         -
        bin red_op_A_0                                    55          1         -    Covered
        bin red_op_A_1                                   231          1         -    Covered
        default bin red_op_A_default                       0                    -    ZERO
    Coverpoint red_op_B_cp                           100.00%        100         -    Covered
        covered/total bins:                                2          2         -
        missing/total bins:                                0          2         -
        % Hit:                                       100.00%        100         -
        bin red_op_B_0                                    47          1         -    Covered
        bin red_op_B_1                                   239          1         -    Covered
        default bin red_op_B_default                       0                    -    ZERO
    Coverpoint bypass_A                              100.00%        100         -    Covered
        covered/total bins:                                2          2         -
        missing/total bins:                                0          2         -
        % Hit:                                       100.00%        100         -
        bin auto[0]                                      284          1         -    Covered
        bin auto[1]                                        2          1         -    Covered
    Coverpoint bypass_B                              100.00%        100         -    Covered
        covered/total bins:                                2          2         -
        missing/total bins:                                0          2         -
        % Hit:                                       100.00%        100         -
        bin auto[0]                                      281          1         -    Covered
        bin auto[1]                                        5          1         -    Covered
    Coverpoint direction                             100.00%        100         -    Covered
        covered/total bins:                                2          2         -
        missing/total bins:                                0          2         -
        % Hit:                                       100.00%        100         -
        bin auto[0]                                      146          1         -    Covered
        bin auto[1]                                      140          1         -    Covered
    Coverpoint serial_in                             100.00%        100         -    Covered
        covered/total bins:                                2          2         -
        missing/total bins:                                0          2         -
        % Hit:                                       100.00%        100         -
        bin auto[0]                                      149          1         -    Covered
        bin auto[1]                                      137          1         -    Covered
    Coverpoint A_cp                                  100.00%        100         -    Covered
        covered/total bins:                                5          5         -
        missing/total bins:                                0          5         -
        % Hit:                                       100.00%        100         -
        bin A_data_0                                      74          1         -    Covered
        bin A_data_max                                    37          1         -    Covered
        bin A_data_min                                    37          1         -    Covered
        bin A_data_walkingones[1]                         31          1         -    Covered
        bin A_data_walkingones[2]                         32          1         -    Covered
        default bin A_data_default                        75                    -    Occurred
    Coverpoint B_cp                                  100.00%        100         -    Covered
        covered/total bins:                                5          5         -
        missing/total bins:                                0          5         -
        % Hit:                                       100.00%        100         -
        bin B_data_0                                      53          1         -    Covered
        bin B_data_max                                    41          1         -    Covered
        bin B_data_min                                    44          1         -    Covered
        bin B_data_walkingones[1]                         38          1         -    Covered
        bin B_data_walkingones[2]                         33          1         -    Covered
        default bin B_data_default                        77                    -    Occurred
    Coverpoint A_cp_mod                              100.00%        100         -    Covered
        covered/total bins:                                3          3         -
        missing/total bins:                                0          3         -
        % Hit:                                       100.00%        100         -
        bin A_data_0                                      74          1         -    Covered
        bin A_data_max                                    37          1         -    Covered
        bin A_data_min                                    37          1         -    Covered
    Coverpoint B_cp_mod                              100.00%        100         -    Covered
        covered/total bins:                                3          3         -
        missing/total bins:                                0          3         -
        % Hit:                                       100.00%        100         -
        bin B_data_0                                      53          1         -    Covered
        bin B_data_max                                    41          1         -    Covered
        bin B_data_min                                    44          1         -    Covered
    Coverpoint ALU_cp                                100.00%        100         -    Covered
        covered/total bins:                                8          8         -
        missing/total bins:                                0          8         -
        % Hit:                                       100.00%        100         -
        bin Bins_shift[SHIFT_4]                           47          1         -    Covered
        bin Bins_shift[ROTATE_5]                          61          1         -    Covered
        bin Bins_arith[ADD_2]                             49          1         -    Covered
        bin Bins_arith[MUL_3]                             53          1         -    Covered
        bin Bins_bitwise[OR_0]                            29          1         -    Covered
        bin Bins_bitwise[XOR_1]                           33          1         -    Covered
        bin Bins_invalid                                  14          1         -    Covered
        bin Bins_trans                                     1          1         -    Covered
    Cross #cross__0#                                 100.00%        100         -    Covered
        covered/total bins:                                8          8         -
        missing/total bins:                                0          8         -
        % Hit:                                       100.00%        100         -
        Auto, Default and User Defined Bins:
            bin <A_data_walkingones[2],red_op_A_0>         4          1         -    Covered
            bin <A_data_walkingones[1],red_op_A_0>         3          1         -    Covered
            bin <A_data_min,red_op_A_1>                   31          1         -    Covered
            bin <A_data_min,red_op_A_0>                    6          1         -    Covered
            bin <A_data_max,red_op_A_1>                   34          1         -    Covered
            bin <A_data_0,red_op_A_1>                     45          1         -    Covered
```

```
            bin <A_data_max,red_op_A_0>                    3            1            -     Covered
            bin <A_data_0,red_op_A_0>                     29            1            -     Covered
        Illegal and Ignore Bins:
            ignore_bin assert_red_op_A                    56                         -     Occurred
    Cross #cross__1#                                 100.00%          100            -     Covered
        covered/total bins:                               18           18            -
        missing/total bins:                                0           18            -
        % Hit:                                       100.00%          100            -
        Auto, Default and User Defined Bins:
            bin <B_data_walkingones[2],red_op_A_1,red_op_B_1>
                                                          22            1            -     Covered
            bin <B_data_walkingones[2],red_op_A_1,red_op_B_0>
                                                           2            1            -     Covered
            bin <B_data_walkingones[1],red_op_A_1,red_op_B_1>
                                                          25            1            -     Covered
            bin <B_data_walkingones[1],red_op_A_1,red_op_B_0>
                                                           2            1            -     Covered
            bin <B_data_min,red_op_A_1,red_op_B_1>        31            1            -     Covered
            bin <B_data_min,red_op_A_1,red_op_B_0>         2            1            -     Covered
            bin <B_data_max,red_op_A_1,red_op_B_1>        37            1            -     Covered
            bin <B_data_0,red_op_A_1,red_op_B_1>          35            1            -     Covered
            bin <B_data_max,red_op_A_1,red_op_B_0>         1            1            -     Covered
            bin <B_data_0,red_op_A_1,red_op_B_0>          14            1            -     Covered
            bin <B_data_walkingones[2],red_op_A_0,red_op_B_0>
                                                           2            1            -     Covered
            bin <B_data_walkingones[1],red_op_A_0,red_op_B_0>
                                                           3            1            -     Covered
            bin <B_data_min,red_op_A_0,red_op_B_1>         8            1            -     Covered
            bin <B_data_min,red_op_A_0,red_op_B_0>         3            1            -     Covered
            bin <B_data_max,red_op_A_0,red_op_B_1>         2            1            -     Covered
            bin <B_data_0,red_op_A_0,red_op_B_1>           2            1            -     Covered
            bin <B_data_max,red_op_A_0,red_op_B_0>         1            1            -     Covered
            bin <B_data_0,red_op_A_0,red_op_B_0>           2            1            -     Covered
        Illegal and Ignore Bins:
            ignore_bin assert_red_op_B                    15                         -     Occurred
    Cross #cross__2#                                 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 arith_permutations                        29            1            -     Covered
    Cross #cross__3#                                 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 addition_cin                              49            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 shift_rotate_direction                   108            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 shift_serial_in                           47            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 or_xor_red_op_A                            5            1            -     Covered
    Cross #cross__7#                                 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_red_op_B                           11            1            -     Covered
    Cross #cross__8#                                  80.76%          100            -     Uncovered
        covered/total bins:                               21           26            -
        missing/total bins:                                5           26            -
        % Hit:                                        80.76%          100            -
        Auto, Default and User Defined Bins:
            bin <Bins_trans,red_op_A_1,red_op_B_0>         1            1            -     Covered
            bin <Bins_invalid,red_op_A_1,red_op_B_1>
                                                           3            1            -     Covered
            bin <Bins_invalid,red_op_A_1,red_op_B_0>
                                                           4            1            -     Covered
            bin <Bins_invalid,red_op_A_0,red_op_B_1>
                                                           4            1            -     Covered
            bin <Bins_invalid,red_op_A_0,red_op_B_0>
                                                           3            1            -     Covered
            bin <Bins_bitwise[XOR_1],red_op_A_0,red_op_B_0>
                                                           7            1            -     Covered
            bin <Bins_bitwise[OR_0],red_op_A_0,red_op_B_0>
                                                           8            1            -     Covered
            bin <Bins_arith[MUL_3],red_op_A_1,red_op_B_1>
                                                          47            1            -     Covered
            bin <Bins_shift[ROTATE_5],red_op_A_1,red_op_B_1>
```

```
                                                            56          1           -      Covered
          bin <Bins_arith[MUL_3],red_op_A_1,red_op_B_0>
                                                            4           1           -      Covered
          bin <Bins_shift[ROTATE_5],red_op_A_1,red_op_B_0>
                                                            4           1           -      Covered
          bin <Bins_arith[ADD_2],red_op_A_1,red_op_B_1>
                                                            44          1           -      Covered
          bin <Bins_shift[SHIFT_4],red_op_A_1,red_op_B_1>
                                                            43          1           -      Covered
          bin <Bins_arith[ADD_2],red_op_A_1,red_op_B_0>
                                                            1           1           -      Covered
          bin <Bins_shift[SHIFT_4],red_op_A_1,red_op_B_0>
                                                            2           1           -      Covered
          bin <Bins_arith[MUL_3],red_op_A_0,red_op_B_1>
                                                            1           1           -      Covered
          bin <Bins_shift[ROTATE_5],red_op_A_0,red_op_B_1>
                                                            1           1           -      Covered
          bin <Bins_arith[MUL_3],red_op_A_0,red_op_B_0>
                                                            1           1           -      Covered
          bin <Bins_arith[ADD_2],red_op_A_0,red_op_B_1>
                                                            4           1           -      Covered
          bin <Bins_shift[SHIFT_4],red_op_A_0,red_op_B_1>
                                                            1           1           -      Covered
          bin <Bins_shift[SHIFT_4],red_op_A_0,red_op_B_0>
                                                            1           1           -      Covered
          bin <Bins_trans,red_op_A_0,*>                     0           1           2      ZERO
          bin <Bins_trans,*,red_op_B_1>                     0           1           2      ZERO
          bin <Bins_shift[ROTATE_5],red_op_A_0,red_op_B_0>
                                                            0           1           1      ZERO
          bin <Bins_arith[ADD_2],red_op_A_0,red_op_B_0>
                                                            0           1           1      ZERO
      Illegal and Ignore Bins:
          ignore_bin invalid_reduction                      47                      -      Occurred

TOTAL COVERGROUP COVERAGE: 99.12%  COVERGROUP TYPES: 1

Total Coverage By Instance (filtered view): 99.12%
```

1.8 8. code Coverage Report

```
Coverage Report by DU with details

=================================================================================
=== Design Unit: work.ALSU
=================================================================================
Branch Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Branches                        32        32         0   100.00%

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

Branch Coverage for Design Unit work.ALSU

    Line         Item                      Count     Source
    ----         ----                      -----     ------
  File ALSU.sv
------------------------------------IF Branch------------------------------------
    24                                      1016     Count coming in to IF
    24              1                          4       if(rst) begin
    35              1                       1012       end else begin
Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
    51                                      1019     Count coming in to IF
    51              1                          6       if(rst) begin
    53              1                       1013       end else begin
Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
    54                                      1013     Count coming in to IF
    54              1                        784           if (invalid)
    56              1                        229           else
Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
    63                                       984     Count coming in to IF
    63              1                          4       if(rst) begin
    66              1                        980       else begin
Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
    67                                       980     Count coming in to IF
    67              1                         51         if (bypass_A_reg && bypass_B_reg)
    69              1                        200         else if (bypass_A_reg)
    71              1                        182         else if (bypass_B_reg)
    73              1                        418         else if (invalid)
    75              1                        129         else begin
Branch totals: 5 hits of 5 branches = 100.00%

------------------------------------CASE Branch------------------------------------
```

```
    76                                         129      Count coming in to CASE
    77              1                           35               3'h0: begin
    87              1                           38               3'h1: begin
    97              1                           10               3'h2: out <= A_reg + B_reg;
    98              1                           18               3'h3: out <= A_reg * B_reg;
    99              1                           15               3'h4: begin
    105             1                           11               3'h5: begin
                                                 2      All False Count
Branch totals: 7 hits of 7 branches = 100.00%

------------------------------------IF Branch------------------------------------
    78                                          35      Count coming in to IF
    78              1                            5               if (red_op_A_reg && red_op_B_reg)
    80              1                            7               else if (red_op_A_reg)
    82              1                           13               else if (red_op_B_reg)
    84              1                           10               else
Branch totals: 4 hits of 4 branches = 100.00%

------------------------------------IF Branch------------------------------------
    88                                          38      Count coming in to IF
    88              1                            6               if (red_op_A_reg && red_op_B_reg)
    90              1                            8               else if (red_op_A_reg)
    92              1                           14               else if (red_op_B_reg)
    94              1                           10               else
Branch totals: 4 hits of 4 branches = 100.00%

------------------------------------IF Branch------------------------------------
    100                                         15      Count coming in to IF
    100             1                           11               if (direction_reg)
    102             1                            4               else
Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
    106                                         11      Count coming in to IF
    106             1                            6               if (direction_reg)
    108             1                            5               else
Branch totals: 2 hits of 2 branches = 100.00%


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

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

Condition Coverage for Design Unit work.ALSU --

  File ALSU.sv
----------------Focused Condition View-------------------
Line       67 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       78 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       88 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:
    Enabled Coverage              Bins    Covered    Misses  Coverage
    ----------------              ----    -------    ------  --------
    Expressions                      8          8         0   100.00%

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

Expression Coverage for Design Unit work.ALSU --

  File ALSU.sv
----------------Focused Expression View-----------------
Line       18 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       19 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       20 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:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Statements                      48        48         0   100.00%

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

Statement Coverage for Design Unit work.ALSU --

    Line         Item                        Count     Source
    ----         ----                        -----     ------
  File ALSU.sv
    1                                                  module ALSU(A, B, cin, serial_in, red_op_A, red_op_B, opcode, bypass_A, bypass_B, c
    2                                                  parameter INPUT_PRIORITY = "A";
    3                                                  parameter FULL_ADDER = "ON";
    4                                                  input clk, cin, rst, red_op_A, red_op_B, bypass_A, bypass_B, direction, serial_in;
    5                                                  input [2:0] opcode;
    6                                                  input signed [2:0] A, B;
    7                                                  output reg [15:0] leds;
    8                                                  output reg signed [5:0] out;
    9
```

```
10                                        reg red_op_A_reg, red_op_B_reg, bypass_A_reg, bypass_B_reg, direction_reg, serial_ir
11                                        reg signed cin_reg;
12                                        reg [2:0] opcode_reg;
13                                        reg signed [2:0] A_reg, B_reg;
14
15                                        wire invalid_red_op, invalid_opcode, invalid;
16
17                                        //Invalid handling
18            1                 463       assign invalid_red_op = (red_op_A_reg | red_op_B_reg) & (opcode_reg[1] | opcode_reg
19            1                 444       assign invalid_opcode = opcode_reg[1] & opcode_reg[2];
20            1                 228       assign invalid = invalid_red_op | invalid_opcode;
21
22                                        //Registering input signals
23            1                1016       always @(posedge clk or posedge rst) begin
24                                          if(rst) begin
25            1                   4            cin_reg <= 0;
26            1                   4            red_op_B_reg <= 0;
27            1                   4            red_op_A_reg <= 0;
28            1                   4            bypass_B_reg <= 0;
29            1                   4            bypass_A_reg <= 0;
30            1                   4            direction_reg <= 0;
31            1                   4            serial_in_reg <= 0;
32            1                   4            opcode_reg <= 0;
33            1                   4            A_reg <= 0;
34            1                   4            B_reg <= 0;
35                                          end else begin
36            1                1012            cin_reg <= cin;
37            1                1012            red_op_B_reg <= red_op_B;
38            1                1012            red_op_A_reg <= red_op_A;
39            1                1012            bypass_B_reg <= bypass_B;
40            1                1012            bypass_A_reg <= bypass_A;
41            1                1012            direction_reg <= direction;
42            1                1012            serial_in_reg <= serial_in;
43            1                1012            opcode_reg <= opcode;
44            1                1012            A_reg <= A;
45            1                1012            B_reg <= B;
46                                          end
47                                        end
48
49                                        //leds output blinking
50            1                1019       always @(posedge clk or posedge rst) begin
51                                          if(rst) begin
52            1                   6            leds <= 0;
53                                          end else begin
54                                              if (invalid)
55            1                 784              leds <= ~leds;
56                                              else
57            1                 229              leds <= 0;
58                                          end
59                                        end
60
61                                        //ALSU output processing
62            1                 984       always @(posedge clk or posedge rst) begin
63                                          if(rst) begin
64            1                   4           out <= 0;
65                                          end
66                                          else begin
67                                            if (bypass_A_reg && bypass_B_reg)
68            1                  51             out <= (INPUT_PRIORITY == "A")? A_reg: B_reg;
69                                            else if (bypass_A_reg)
70            1                 200             out <= A_reg;
71                                            else if (bypass_B_reg)
72            1                 182             out <= B_reg;
73                                            else if (invalid)
74            1                 418               out <= 0;
75                                            else begin
76                                                case (opcode)
77                                                  3'h0: begin
78                                                    if (red_op_A_reg && red_op_B_reg)
79            1                   5                     out <= (INPUT_PRIORITY == "A")? |A_reg: |B_reg;
80                                                    else if (red_op_A_reg)
81            1                   7                     out <= |A_reg;
82                                                    else if (red_op_B_reg)
83            1                  13                     out <= |B_reg;
84                                                    else
85            1                  10                     out <= A_reg | B_reg;
86                                                  end
87                                                  3'h1: begin
88                                                    if (red_op_A_reg && red_op_B_reg)
89            1                   6                     out <= (INPUT_PRIORITY == "A")? ^A_reg: ^B_reg;
90                                                    else if (red_op_A_reg)
91            1                   8                     out <= ^A_reg;
92                                                    else if (red_op_B_reg)
93            1                  14                     out <= ^B_reg;
94                                                    else
95            1                  10                     out <= A_reg ^ B_reg;
96                                                  end
97            1                  10               3'h2: out <= A_reg + B_reg;
98            1                  18               3'h3: out <= A_reg * B_reg;
99                                                  3'h4: begin
100                                                   if (direction_reg)
101           1                  11                     out <= {out[4:0], serial_in_reg};
```

```
102                                                        else
103             1                        4                   out <= {serial_in_reg , out[5:1]};
104                                                      end
105                                                      3'h5: begin
106                                                        if (direction_reg)
107             1                        6                   out <= {out[4:0], out[5]};
108                                                        else
109             1                        5                   out <= {out[0], out[5:1]};

Toggle Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Toggles                        118       118         0   100.00%

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

Toggle Coverage for Design Unit work.ALSU

                                              Node      1H->0L      0L->1H  "Coverage"
                                              ---------------------------------------
                                            A[0-2]           1           1      100.00
                                        A_reg[0-2]           1           1      100.00
                                            B[0-2]           1           1      100.00
                                        B_reg[0-2]           1           1      100.00
                                          bypass_A           1           1      100.00
                                      bypass_A_reg           1           1      100.00
                                          bypass_B           1           1      100.00
                                      bypass_B_reg           1           1      100.00
                                               cin           1           1      100.00
                                           cin_reg           1           1      100.00
                                               clk           1           1      100.00
                                         direction           1           1      100.00
                                     direction_reg           1           1      100.00
                                           invalid           1           1      100.00
                                    invalid_opcode           1           1      100.00
                                    invalid_red_op           1           1      100.00
                                         leds[0-15]          1           1      100.00
                                       opcode[0-2]           1           1      100.00
                                   opcode_reg[0-2]           1           1      100.00
                                          out[0-5]           1           1      100.00
                                          red_op_A           1           1      100.00
                                      red_op_A_reg           1           1      100.00
                                          red_op_B           1           1      100.00
                                      red_op_B_reg           1           1      100.00
                                               rst           1           1      100.00
                                         serial_in           1           1      100.00
                                     serial_in_reg           1           1      100.00

Total Node Count     =         59
Toggled Node Count   =         59
Untoggled Node Count =          0

Toggle Coverage      =     100.00% (118 of 118 bins)


Total Coverage By Design Unit (filtered view): 100.00%
```

1.9 9.Waveform

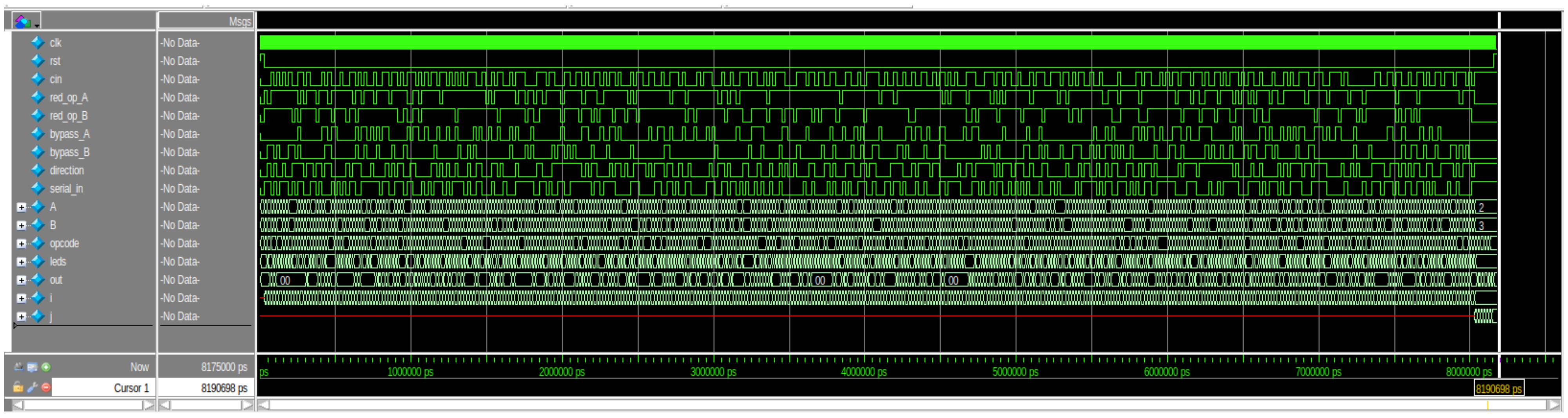
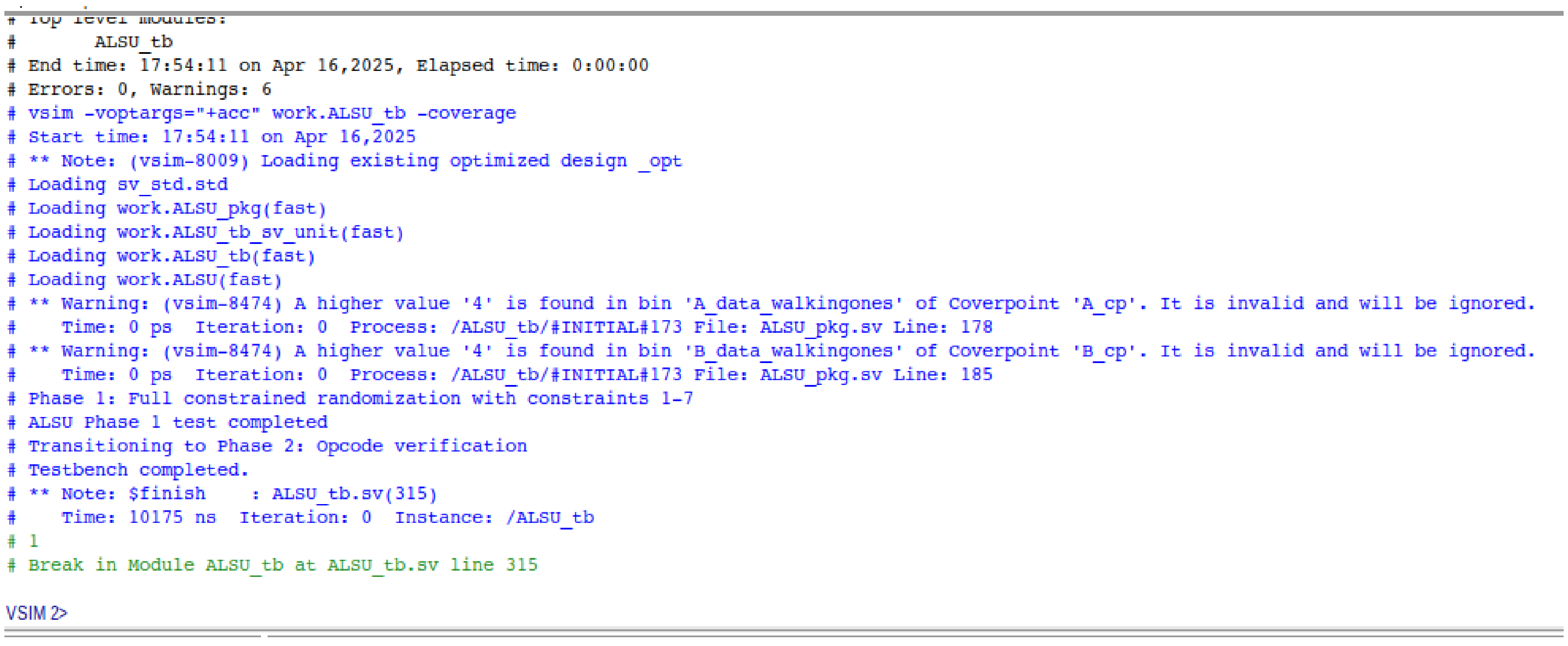


Figure 1: simulation waveform



```
# Top level modules:
#       ALSU_tb
# End time: 17:54:11 on Apr 16,2025, Elapsed time: 0:00:00
# Errors: 0, Warnings: 6
# vsim -voptargs="+acc" work.ALSU_tb -coverage
# Start time: 17:54:11 on Apr 16,2025
# ** Note: (vsim-8009) Loading existing optimized design _opt
# Loading sv_std.std
# Loading work.ALSU_pkg(fast)
# Loading work.ALSU_tb_sv_unit(fast)
# Loading work.ALSU_tb(fast)
# Loading work.ALSU(fast)
# ** Warning: (vsim-8474) A higher value '4' is found in bin 'A_data_walkingones' of Coverpoint 'A_cp'. It is invalid and will be ignored.
#    Time: 0 ps  Iteration: 0  Process: /ALSU_tb/#INITIAL#173 File: ALSU_pkg.sv Line: 178
# ** Warning: (vsim-8474) A higher value '4' is found in bin 'B_data_walkingones' of Coverpoint 'B_cp'. It is invalid and will be ignored.
#    Time: 0 ps  Iteration: 0  Process: /ALSU_tb/#INITIAL#173 File: ALSU_pkg.sv Line: 185
# Phase 1: Full constrained randomization with constraints 1-7
# ALSU Phase 1 test completed
# Transitioning to Phase 2: Opcode verification
# Testbench completed.
# ** Note: $finish    : ALSU_tb.sv(315)
#    Time: 10175 ns  Iteration: 0  Instance: /ALSU_tb
# 1
# Break in Module ALSU_tb at ALSU_tb.sv line 315

VSIM 2>
```

Figure 2: Transcript : all test cases passed

2 Q2: SVA

2.1 1. Code

```
//Q1
property p_1;
 @(posedge clk) a |-> ##2 b;
endproperty

assert property (p_1);

//Q2
property p_2;
 @(posedge clk) (a & b) |-> ##[1:3] c;
endproperty

assert property (p_2);

//Q3
sequence s11b;
  @(posedge clk) b ##2 (!b);
endsequence

//Q4-a
property p_3;
  @(poseedge clk) $onehot(Y);
endproperty

assert property (p_3);

//Q4-b
property p_4;
  @(posedge clk) (D == 0) |-> ##1 (!valid);
endproperty

assert property (p_4);
```

3 Q3: Counter

3.1 1. Testbench code

```
import counter_pkg::*;

module counter_tb(counter_intf.TEST count_if_tb);

    //================================================================
    // Create an instance of the random config class
    //================================================================
    counter_cfg cfg;

    //================================================================
    // Direct assertion: asynchronous reset assertion
    //================================================================
    always_comb begin
        if(!count_if_tb.rst_n) begin
            a_reset: assert final(count_if_tb.count_out == 0);
        end
    end

    //================================================================
    // SVA replaced golden model
    //================================================================
    // When the load control signal is active, then the dout has the value of the din
    property p_load;
        @(posedge count_if_tb.clk) disable iff (!count_if_tb.rst_n) (!count_if_tb.load_n) |=> (count_if_tb.count_out ===
           $past(count_if_tb.data_load));
    endproperty
    p1:assert property (p_load) else
         $error("SVA p_load failed: when load asserted, count_out != data_load");
    cover property (p_load);

    // When the load control signal is not active, and the enable is off then the dout does not change
    property p_load_en;
        @(posedge count_if_tb.clk) disable iff (!count_if_tb.rst_n) (count_if_tb.load_n&&!count_if_tb.ce) |=> (count_if_tb.count_out ===
            $past(count_if_tb.count_out));
    endproperty
    p2:assert property (p_load_en) else
         $error("SVA␣p_load_en␣failed:␣when␣load␣not␣asserted␣and␣enable␣off,␣count_out␣!=␣past(count_out)");

    cover property (p_load_en);

    // When the load control signal is not active and the enable is active, and the up_down is high then the dout is incremented.
    property p_inc;
        @(posedge count_if_tb.clk) disable iff (!count_if_tb.rst_n) (count_if_tb.load_n && count_if_tb.ce && count_if_tb.up_down)
                                |=> (count_if_tb.count_out === $past(count_if_tb.count_out) + 1'b1);
    endproperty
    p3:assert property (p_inc) else
         $error("SVA␣p_inc␣failed:␣increment␣behavior");

    cover property (p_inc);

    // When the load control signal is not active and the enable is active, and the up_down is low then the dout is decremented.
    property p_dec;
        @(posedge count_if_tb.clk) disable iff (!count_if_tb.rst_n) (count_if_tb.load_n && count_if_tb.ce && !count_if_tb.up_down)
                                |=> (count_if_tb.count_out === $past(count_if_tb.count_out) - 1'b1);
    endproperty
    p4:assert property (p_dec) else
         $error("SVA␣p_dec␣failed:␣decrement␣behavior");

    cover property (p_dec);

    // max_count output is high when the counter output is maximum.
    always_comb begin
        if(count_if_tb.rst_n && count_if_tb.count_out === {count_if_tb.WIDTH{1'b1}}) begin
            p_max_flag: assert final(count_if_tb.max_count === 1'b1) else
                                $error("SVA␣p_max_flag␣failed:␣max_count␣mismatch");
        end
    end

    //zero output is high when the counter output is zero.
    always_comb begin
        if(count_if_tb.rst_n && count_if_tb.count_out === {count_if_tb.WIDTH{1'b0}}) begin
            p_zero_flag: assert final(count_if_tb.zero === 1'b1) else
                                $error("SVA␣p_zero_flag␣failed:␣zero␣flag␣mismatch");
        end
    end

  //===============================================================
  // Main verification process: Drives the testbench flow
  //===============================================================
  initial begin
    // Create config object for randomization
    cfg = new();

    // 2) Run random tests
    for (int i = 0; i < 500; i++) begin
      if (!cfg.randomize()) begin
        $error("Randomization␣failed!");
        $finish;
      end

      @(negedge count_if_tb.clk); // Wait for clock edge

      // Drive signals from random config
      count_if_tb.rst_n     = cfg.rst_n;
      count_if_tb.load_n    = cfg.load_n;
      count_if_tb.up_down   = cfg.up_down;
      count_if_tb.ce        = cfg.ce;
      count_if_tb.data_load= cfg.data_load;

      cfg.count_out = count_if_tb.count_out;

      cfg.sample();
    end

    $display("All␣done.␣End␣of␣simulation.");
    $finish; // End simulation
  end

endmodule
```

3.2 2. Package code

```
package counter_pkg;

    //===========================
    // 1) Declare the parameter
    //===========================
    parameter int WIDTH = 4;

    //====================================================
    // 2) Create a class for constrained-random stimulus
    //====================================================
    class counter_cfg;

        //==================================
        // DUT signals we want to randomize
        //==================================
        bit             clk;            // clock signal
        rand bit        rst_n;          // Active-low reset
        rand bit        load_n;         // Active-low load
        rand bit        up_down;        // 1 => increment, 0 => decrement
        rand bit        ce;             // clock enable
        rand logic [WIDTH-1:0] data_load;
        logic [WIDTH-1:0] count_out;

        //====================
        // Coverage group
        //====================
        covergroup cg ;
          cp1: coverpoint rst_n;
          cp2: coverpoint load_n;
          cp3: coverpoint up_down;
          cp4: coverpoint ce;
          cp5: coverpoint data_load;
          cp6: coverpoint count_out;
        endgroup

        //===================
        // Constructor
        //===================
        function new();
          cg = new();
        endfunction

        function void sample();
          cg.sample();
        endfunction

        //===============================================
        // 3) Constraints to meet the 70%/30% guidelines
        // use distribution for probability
        //===============================================
        constraint reset_deactivated_most {
          // Reset low 30% of time, high 70%
          rst_n dist { 1 := 70, 0 := 30 };
        }

        constraint load_active_70 {
          // load_n=0 is "active" => 70% of time
          load_n dist { 0 := 70, 1 := 30 };
        }

        constraint enable_active_70 {
          // ce=1 => 70% of time
          ce dist { 1 := 70, 0 := 30 };
        }

        constraint up_down_dist {
          // Distribution constraint: 50% chance of 0, 50% chance of 1
          up_down dist { 0 := 50, 1 := 50 };
        }

        constraint up_down_data_load_c {
              if (up_down) {
            // up_down=1 => pick data_load mostly in lower half
            data_load dist {
              [0 : (1<<(WIDTH-1))-1] := 80,
              [(1<<(WIDTH-1)) : (1<<WIDTH)-1] := 20
            };
              } else {
            // up_down=0 => pick data_load mostly in upper half
            data_load dist {
              [0 : (1<<(WIDTH-1))-1] := 20,
              [(1<<(WIDTH-1)) : (1<<WIDTH)-1] := 80
            };
              }
        }

    endclass

endpackage
```

3.3 3. Design code

```
////////////////////////////////////////////////////////////////////////////////
// Author: Kareem Waseem
// Course: Digital Verification using SV & UVM
//
// Description: Counter Design
//
////////////////////////////////////////////////////////////////////////////////
module counter (counter_intf.DUT count_if_dut);

always @(posedge count_if_dut.clk or negedge count_if_dut.rst_n) begin
    if (!count_if_dut.rst_n)
        count_if_dut.count_out <= 0;
    else if (!count_if_dut.load_n)
        count_if_dut.count_out <= count_if_dut.data_load;
    else if (count_if_dut.ce) begin
        if (count_if_dut.up_down)
            count_if_dut.count_out <= count_if_dut.count_out + 1;
        else
            count_if_dut.count_out <= count_if_dut.count_out - 1;
    end
end

assign count_if_dut.max_count = (count_if_dut.count_out == {count_if_dut.WIDTH{1'b1}})? 1:0;
assign count_if_dut.zero = (count_if_dut.count_out == 0)? 1:0;
```

```
endmodule
```

3.4 4. Interface code

```
interface counter_intf (
    input bit clk
);

//================================================================
// Local parameters: Override the default WIDTH if needed
//================================================================
localparam WIDTH = 4;


//================================================================
// Declare all signals used to interface with DUT or top
//================================================================
bit rst_n;
bit load_n;
bit up_down;
bit ce;
bit [WIDTH-1:0] data_load;
bit [WIDTH-1:0] count_out;
bit max_count;
bit zero;

modport DUT ( input clk,rst_n,load_n,up_down,ce,data_load,
              output count_out,max_count,zero);

modport TEST ( output clk,rst_n,load_n,up_down,ce,data_load,
               input count_out,max_count,zero);

endinterface
```

3.5 5. Bug Fixes

```
missing (begin and end) for "else if (ce)"
```

3.6 6. Verification Plan

Label	Design Requirement Description	Stimulus Generation	Functional Coverage	Functionality Check
COUNTER_1	When reset (`rst_n`) is asserted, the output `count_out` should be zero.	Directed at the start of the simulation, followed by randomization with a constraint to keep reset active for 30% of the time.	Coverage point to track how many times reset is asserted and confirm `count_out == 0`.	A checker in the testbench ensures `count_out` is 0 when `rst_n == 0`.
COUNTER_2	When load (`load_n`) is asserted, `count_out` should take the value of `data_load`.	Randomization of `load_n` with a 70% probability of being active (low) and randomized `data_load` values.	Coverage point to track how many times `load_n` is asserted and the range of `data_load` values.	A checker in the testbench verifies `count_out == data_load` when `load_n == 0`.
COUNTER_3	When `ce` is enabled, the counter should increment or decrement based on `up_down`.	Randomization with 70% chance for `ce` being high, and a 50-50 distribution for `up_down`.	Coverage point to capture the toggling of `up_down` and transitions of `count_out`.	A checker compares `count_out` with the expected increment or decrement, verified against the golden model.
COUNTER_4	`max_count` should be asserted when `count_out` reaches its maximum value.	Random tests allowing the counter to reach its maximum possible value ({`WIDTH{1'b1}`}).	Coverage point to check how often `max_count` is triggered.	A checker validates `max_count == 1` when `count_out == max value`.
COUNTER_5	`zero` should be asserted when `count_out` is zero.	Directed reset tests and decrement tests pushing `count_out` to zero.	Coverage point for how often `zero` is asserted.	A checker verifies `zero == 1` only when `count_out == 0`.

Table 2: Verification Plan for Counter Design

3.7 7. Do File

```
vlib work
vlog counter_pkg.sv counter.sv counter_intf.sv counter_tb.sv top_module.sv +cover -covercells
vsim -voptargs=+acc work.top -cover
do wave.do
coverage save top.ucdb -onexit
run -all
do coverage.do

# to run do file
#-- do run.do
# to execute coverage report (one for code coverage and other fuctional coverage)
#-- vcover report top.ucdb -details -annotate -all -output code_coverage_rpt.txt -du=counter
#-- vcover report -details -cvg -output functional_coverage_rpt.txt top.ucdb
```

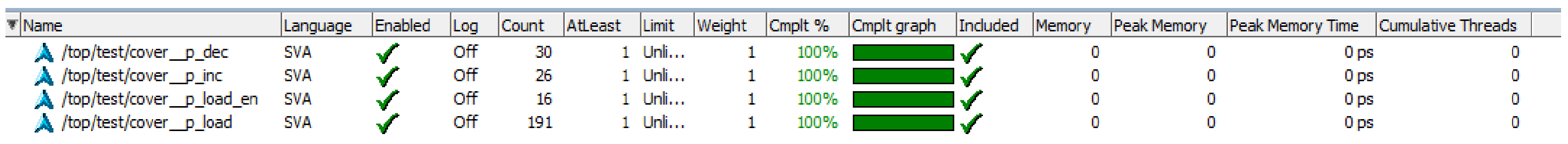
3.8 8. code Coverage Report

```
Coverage Report by DU with details

=================================================================================
=== Design Unit: work.counter
=================================================================================
Branch Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Branches                        10        10         0   100.00%

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

Branch Coverage for Design Unit work.counter

    Line         Item                      Count     Source
    ----         ----                      -----     ------
  File counter.sv
------------------------------------IF Branch------------------------------------
    11                                       207     Count coming in to IF
    11              1                         81         if (!count_if_dut.rst_n)
    13              1                         98         else if (!count_if_dut.load_n)
    15              1                         22         else if (count_if_dut.ce) begin
                                               6     All False Count
Branch totals: 4 hits of 4 branches = 100.00%

------------------------------------IF Branch------------------------------------
    16                                        22     Count coming in to IF
    16              1                         13             if (count_if_dut.up_down)
    18              1                          9             else
Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
    23                                       145     Count coming in to IF
    23              1                         10     assign count_if_dut.max_count = (count_if_dut.count_out == {count_if_dut.WIDTH{1'b1
    23              2                        135     assign count_if_dut.max_count = (count_if_dut.count_out == {count_if_dut.WIDTH{1'b1
Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
    24                                       145     Count coming in to IF
    24              1                         34     assign count_if_dut.zero = (count_if_dut.count_out == 0)? 1:0;
    24              2                        111     assign count_if_dut.zero = (count_if_dut.count_out == 0)? 1:0;
Branch totals: 2 hits of 2 branches = 100.00%


Condition Coverage:
    Enabled Coverage              Bins   Covered    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Conditions                       2         2         0   100.00%

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

Condition Coverage for Design Unit work.counter --

  File counter.sv
----------------Focused Condition View-------------------
Line       23 Item    1  (count_if_dut.count_out == {count_if_dut.WIDTH{1}})
Condition totals: 1 of 1 input term covered = 100.00%

                                              Input Term   Covered  Reason for no coverage   Hint
                                             -----------  --------  -----------------------  --------------
  (count_if_dut.count_out == {count_if_dut.WIDTH{1}})         Y

     Rows:       Hits  FEC Target                                              Non-masking condition(s)
 ---------  ---------  --------------------                                    ------------------------
  Row   1:          1  (count_if_dut.count_out == {count_if_dut.WIDTH{1}})_0   -
  Row   2:          1  (count_if_dut.count_out == {count_if_dut.WIDTH{1}})_1   -

----------------Focused Condition View-------------------
Line       24 Item    1  (count_if_dut.count_out == 0)
Condition totals: 1 of 1 input term covered = 100.00%

                    Input Term   Covered  Reason for no coverage   Hint
                   -----------  --------  -----------------------  --------------
  (count_if_dut.count_out == 0)         Y

     Rows:       Hits  FEC Target                     Non-masking condition(s)
 ---------  ---------  --------------------           ------------------------
  Row   1:          1  (count_if_dut.count_out == 0)_0  -
  Row   2:          1  (count_if_dut.count_out == 0)_1  -


Statement Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Statements                       7         7         0   100.00%

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

Statement Coverage for Design Unit work.counter --

    Line         Item                      Count     Source
    ----         ----                      -----     ------
  File counter.sv
    8                                                module counter (counter_intf.DUT count_if_dut);
    9
    10              1                        207     always @(posedge count_if_dut.clk or negedge count_if_dut.rst_n) begin
    11                                                   if (!count_if_dut.rst_n)
    12              1                         81             count_if_dut.count_out <= 0;
    13                                                      else if (!count_if_dut.load_n)
    14              1                        98                 count_if_dut.count_out <= count_if_dut.data_load;
    15                                                      else if (count_if_dut.ce) begin
    16                                                          if (count_if_dut.up_down)
    17              1                        13                     count_if_dut.count_out <= count_if_dut.count_out + 1;
    18                                                          else
    19              1                         9                     count_if_dut.count_out <= count_if_dut.count_out - 1;
    20                                                      end
    21                                                  end
    22
    23              1                       145         assign count_if_dut.max_count = (count_if_dut.count_out == {count_if_dut.WIDTH{1'b1
    24              1                       145         assign count_if_dut.zero = (count_if_dut.count_out == 0)? 1:0;


Total Coverage By Design Unit (filtered view): 100.00%
```

3.9 9. functional coverage report

```
Coverage Report by instance with details

=================================================================================
=== Instance: /counter_pkg
=== Design Unit: work.counter_pkg
=================================================================================

Covergroup Coverage:
    Covergroups                      1        na        na    100.00%
        Coverpoints/Crosses          6        na        na         na
            Covergroup Bins         40        40         0    100.00%
----------------------------------------------------------------------------------------------------------------
Covergroup                                             Metric       Goal       Bins    Status

----------------------------------------------------------------------------------------------------------------
 TYPE /counter_pkg/counter_cfg/cg                      100.00%        100          -    Covered
    covered/total bins:                                     40         40          -
    missing/total bins:                                      0         40          -
    % Hit:                                             100.00%        100          -
    Coverpoint cp1                                     100.00%        100          -    Covered
        covered/total bins:                                  2          2          -
        missing/total bins:                                  0          2          -
        % Hit:                                         100.00%        100          -
        bin auto[0]                                         46          1          -    Covered
        bin auto[1]                                        126          1          -    Covered
    Coverpoint cp2                                     100.00%        100          -    Covered
        covered/total bins:                                  2          2          -
        missing/total bins:                                  0          2          -
        % Hit:                                         100.00%        100          -
        bin auto[0]                                        129          1          -    Covered
        bin auto[1]                                         43          1          -    Covered
    Coverpoint cp3                                     100.00%        100          -    Covered
        covered/total bins:                                  2          2          -
        missing/total bins:                                  0          2          -
        % Hit:                                         100.00%        100          -
        bin auto[0]                                         80          1          -    Covered
        bin auto[1]                                         92          1          -    Covered
    Coverpoint cp4                                     100.00%        100          -    Covered
        covered/total bins:                                  2          2          -
        missing/total bins:                                  0          2          -
        % Hit:                                         100.00%        100          -
        bin auto[0]                                         51          1          -    Covered
        bin auto[1]                                        121          1          -    Covered
    Coverpoint cp5                                     100.00%        100          -    Covered
        covered/total bins:                                 16         16          -
        missing/total bins:                                  0         16          -
        % Hit:                                         100.00%        100          -
        bin auto[0]                                          7          1          -    Covered
        bin auto[1]                                         10          1          -    Covered
        bin auto[2]                                         10          1          -    Covered
        bin auto[3]                                         12          1          -    Covered
        bin auto[4]                                         10          1          -    Covered
        bin auto[5]                                         15          1          -    Covered
        bin auto[6]                                          9          1          -    Covered
        bin auto[7]                                         13          1          -    Covered
        bin auto[8]                                         12          1          -    Covered
        bin auto[9]                                         10          1          -    Covered
        bin auto[10]                                        12          1          -    Covered
        bin auto[11]                                        13          1          -    Covered
        bin auto[12]                                         9          1          -    Covered
        bin auto[13]                                        13          1          -    Covered
        bin auto[14]                                        11          1          -    Covered
        bin auto[15]                                         6          1          -    Covered
    Coverpoint cp6                                     100.00%        100          -    Covered
        covered/total bins:                                 16         16          -
        missing/total bins:                                  0         16          -
        % Hit:                                         100.00%        100          -
        bin auto[0]                                         57          1          -    Covered
        bin auto[1]                                         10          1          -    Covered
        bin auto[2]                                          6          1          -    Covered
        bin auto[3]                                          6          1          -    Covered
        bin auto[4]                                          3          1          -    Covered
        bin auto[5]                                         11          1          -    Covered
        bin auto[6]                                      5          1          -      Covered
        bin auto[7]                                      7          1          -      Covered
        bin auto[8]                                      9          1          -      Covered
        bin auto[9]                                     10          1          -      Covered
        bin auto[10]                                     9          1          -      Covered
        bin auto[11]                                     8          1          -      Covered
        bin auto[12]                                     7          1          -      Covered
        bin auto[13]                                     7          1          -      Covered
        bin auto[14]                                     7          1          -      Covered
        bin auto[15]                                    10          1          -      Covered

COVERGROUP COVERAGE:
------------------------------------------------------------------------------------------------
Covergroup                                          Metric       Goal       Bins      Status

------------------------------------------------------------------------------------------------
 TYPE /counter_pkg/counter_cfg/cg                  100.00%        100          -      Covered
    covered/total bins:                                 40         40          -
    missing/total bins:                                  0         40          -
    % Hit:                                         100.00%        100          -
    Coverpoint cp1                                 100.00%        100          -      Covered
        covered/total bins:                              2          2          -
        missing/total bins:                              0          2          -
        % Hit:                                     100.00%        100          -
        bin auto[0]                                     46          1          -      Covered
        bin auto[1]                                    126          1          -      Covered
    Coverpoint cp2                                 100.00%        100          -      Covered
        covered/total bins:                              2          2          -
        missing/total bins:                              0          2          -
        % Hit:                                     100.00%        100          -
        bin auto[0]                                    129          1          -      Covered
        bin auto[1]                                     43          1          -      Covered
    Coverpoint cp3                                 100.00%        100          -      Covered
        covered/total bins:                              2          2          -
        missing/total bins:                              0          2          -
        % Hit:                                     100.00%        100          -
        bin auto[0]                                     80          1          -      Covered
        bin auto[1]                                     92          1          -      Covered
    Coverpoint cp4                                 100.00%        100          -      Covered
        covered/total bins:                              2          2          -
        missing/total bins:                              0          2          -
        % Hit:                                     100.00%        100          -
        bin auto[0]                                     51          1          -      Covered
        bin auto[1]                                    121          1          -      Covered
    Coverpoint cp5                                 100.00%        100          -      Covered
        covered/total bins:                             16         16          -
        missing/total bins:                              0         16          -
        % Hit:                                     100.00%        100          -
        bin auto[0]                                      7          1          -      Covered
        bin auto[1]                                     10          1          -      Covered
        bin auto[2]                                     10          1          -      Covered
        bin auto[3]                                     12          1          -      Covered
        bin auto[4]                                     10          1          -      Covered
        bin auto[5]                                     15          1          -      Covered
        bin auto[6]                                      9          1          -      Covered
        bin auto[7]                                     13          1          -      Covered
        bin auto[8]                                     12          1          -      Covered
        bin auto[9]                                     10          1          -      Covered
        bin auto[10]                                    12          1          -      Covered
        bin auto[11]                                    13          1          -      Covered
        bin auto[12]                                     9          1          -      Covered
        bin auto[13]                                    13          1          -      Covered
        bin auto[14]                                    11          1          -      Covered
        bin auto[15]                                     6          1          -      Covered
    Coverpoint cp6                                 100.00%        100          -      Covered
        covered/total bins:                             16         16          -
        missing/total bins:                              0         16          -
        % Hit:                                     100.00%        100          -
        bin auto[0]                                     57          1          -      Covered
        bin auto[1]                                     10          1          -      Covered
        bin auto[2]                                      6          1          -      Covered
        bin auto[3]                                      6          1          -      Covered
        bin auto[4]                                      3          1          -      Covered
        bin auto[5]                                     11          1          -      Covered
        bin auto[6]                                      5          1          -      Covered
        bin auto[7]                                      7          1          -      Covered
        bin auto[8]                                      9          1          -      Covered
        bin auto[9]                                     10          1          -      Covered
        bin auto[10]                                     9          1          -      Covered
        bin auto[11]                                     8          1          -      Covered
        bin auto[12]                                     7          1          -      Covered
        bin auto[13]                                     7          1          -      Covered
        bin auto[14]                                     7          1          -      Covered
        bin auto[15]                                    10          1          -      Covered

TOTAL COVERGROUP COVERAGE: 100.00%  COVERGROUP TYPES: 1

Total Coverage By Instance (filtered view): 100.00%
```

3.10 10.Waveform



Name	Language	Enabled	Log	Count	AtLeast	Limit	Weight	Cmplt %	Cmplt graph	Included	Memory	Peak Memory	Peak Memory Time	Cumulative Threads
/top/test/cover__p_dec	SVA	✓	Off	30	1	Unli...	1	100%	🟩	✓	0	0	0 ps	0
/top/test/cover__p_inc	SVA	✓	Off	26	1	Unli...	1	100%	🟩	✓	0	0	0 ps	0
/top/test/cover__p_load_en	SVA	✓	Off	16	1	Unli...	1	100%	🟩	✓	0	0	0 ps	0
/top/test/cover__p_load	SVA	✓	Off	191	1	Unli...	1	100%	🟩	✓	0	0	0 ps	0

Figure 3: cover property

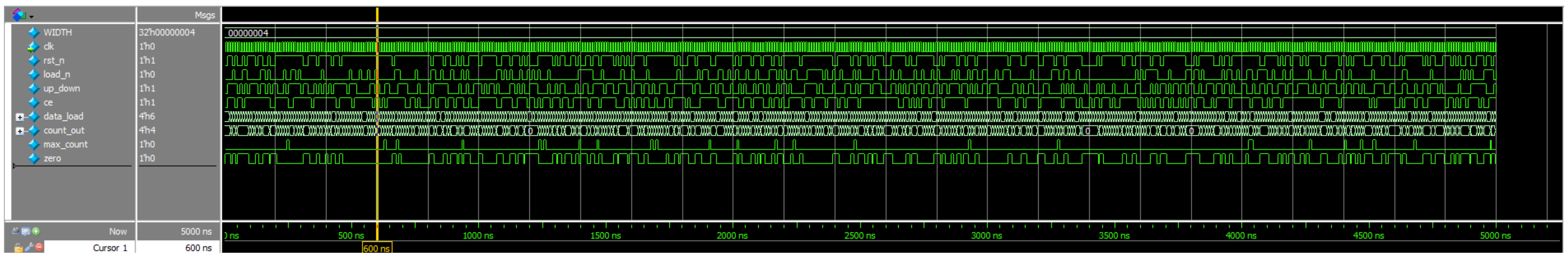
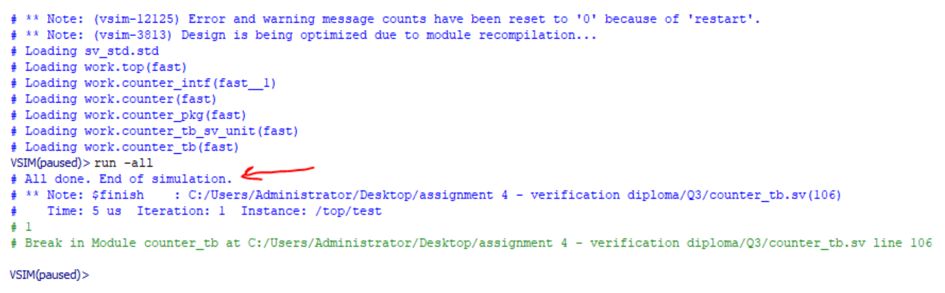


Figure 4: simulation waveform



```
# ** Note: (vsim-12125) Error and warning message counts have been reset to '0' because of 'restart'.
# ** Note: (vsim-3813) Design is being optimized due to module recompilation...
# Loading sv_std.std
# Loading work.top(fast)
# Loading work.counter_intf(fast__1)
# Loading work.counter(fast)
# Loading work.counter_pkg(fast)
# Loading work.counter_tb_sv_unit(fast)
# Loading work.counter_tb(fast)
VSIM(paused)> run -all
# All done. End of simulation.
# ** Note: $finish    : C:/Users/Administrator/Desktop/assignment 4 - verification diploma/Q3/counter_tb.sv(106)
#    Time: 5 us  Iteration: 1  Instance: /top/test
# 1
# Break in Module counter_tb at C:/Users/Administrator/Desktop/assignment 4 - verification diploma/Q3/counter_tb.sv line 106

VSIM(paused)>
```

Figure 5: Transcript : all test cases passed

4 Q4: configuration register

4.1 1. Testbench code

```
module config_reg_tb;

    //---------------------------------------------
    // Enumerated type for refister addresses
    //---------------------------------------------
    typedef enum logic [2:0] {
        ADC0_REG = 3'd0,
        ADC1_REG = 3'd1,
        TEMP_SENSOR0_REG = 3'd2,
        TEMP_SENSOR1_REG = 3'd3,
        ANALOG_TEST = 3'd4,
        DIGITAL_TEST = 3'd5,
        AMP_GAIN = 3'd6,
        DIGITAL_CONFIG = 3'd7
    } reg_addr_e;

    //--------------------------------------------------
    // Golden model for reset values of register
    // using assosiative array with type string key
    //--------------------------------------------------
    logic [15:0] reset_assoc [string];

    //--------------------------------------
    // instantiate configuration register
    //--------------------------------------
    logic clk;
    logic reset;
    logic write;
    logic [15:0] data_in;
    reg_addr_e   address;
    logic [15:0] data_out;

    config_reg dut (
        .clk(clk),
        .reset(reset),
        .write(write),
        .data_in(data_in),
        .address(address),
        .data_out(data_out)
    );

    //------------------
    // Clock generator
    //------------------
    initial begin
        clk = 1'b0;
        forever #5 clk = ~clk;
    end

    //-------------------------------------
    // Task to check dut during reseting
    //-------------------------------------
    task check_rest();
    begin
        reg_addr_e addr;

        reset = 1'b1;
        #10;
        reset = 1'b0;

        $display("check␣reset␣task");

        for (int i = 0; i < addr.num(); i++) begin
            if(i==0) begin
                addr = addr.first();
                address = addr;
            end else if (i==(addr.num()-1))begin
                addr = addr.last();
                address = addr;
            end else begin
                addr = addr.next();
                address = addr;
            end
            #10;
            //compare registers content after resting with associative array that hold reset values
            if (data_out !== reset_assoc[addr.name()]) begin
                $display("Mismatch␣at␣%s:␣expected␣%h,␣got␣%h", addr.name(), reset_assoc[addr.name()], data_out);
            end
        end

    end
    endtask

    //-----------------------------------
    // Task to check register values
    //-----------------------------------
    task check_registers();
        reg_addr_e addr;
        logic [15:0] expected_output;
        // corner cases

        $display("check␣register␣values␣all␣zero␣write");
        write = 1'b1;
        data_in = 16'b0;

        for (int i = 0; i < addr.num(); i++) begin
            if(i==0) begin
                addr = addr.first();
                address = addr;
            end else if (i==(addr.num()-1))begin
                addr = addr.last();
                address = addr;
            end else begin
                addr = addr.next();
                address = addr;
            end
            #10;
            if (data_out !== 16'b0) begin
                $display("Mismatch␣at␣%s:␣expected␣%h,␣got␣%h", addr.name(), 16'b0, data_out);
            end
        end

        $display("check␣register␣values␣all␣ones␣write");
        write = 1'b1;
        data_in = 16'hFFFF;

        for (int i = 0; i < addr.num(); i++) begin
            if(i==0) begin
                addr = addr.first();
                address = addr;
            end else if (i==(addr.num()-1))begin
                addr = addr.last();
                address = addr;
            end else begin
                addr = addr.next();
                address = addr;
            end
            #10;
            if (data_out !== 16'hFFFF) begin
                $display("Mismatch␣at␣%s:␣expected␣%h,␣got␣%h", addr.name(), 16'hFFFF, data_out);
            end
        end

        $display("check␣register␣values␣random␣write");
        write = 1'b1;
        expected_output = $urandom_range(0,16'hFFFF);
        data_in = expected_output;

        for (int i = 0; i < addr.num(); i++) begin
            if(i==0) begin
                addr = addr.first();
                address = addr;
            end else if (i==(addr.num()-1))begin
                addr = addr.last();
                address = addr;
            end else begin
                addr = addr.next();
                address = addr;
            end
            #10;
            if (data_out !== expected_output) begin
                $display("Mismatch␣at␣%s:␣expected␣%h,␣got␣%h", addr.name(), expected_output, data_out);
            end
        end

    endtask

    //--------------------
    // simulation steps
    //-------------------
    initial begin
        //initialize inputs
        reset = 1'b0;  //un activated reset
        write = 1'b0;
        data_in = 15'b0;
        address = ADC0_REG;

        // set reset values in associative array to hold it to the end of simulation
        reset_assoc["ADC0_REG"]=16'hFFFF;
        reset_assoc["ADC1_REG"]=16'h0;
        reset_assoc["TEMP_SENSOR0_REG"]=16'h0;
        reset_assoc["TEMP_SENSOR1_REG"]=16'h0;
        reset_assoc["ANALOG_TEST"]=16'hABCD;
        reset_assoc["DIGITAL_TEST"]=16'h0;
        reset_assoc["AMP_GAIN"]=16'h0;
        reset_assoc["DIGITAL_CONFIG"]=16'h1;

        // reset dut
        @(posedge clk);
        check_rest();

        @(posedge clk);
        check_registers();

        $finish;
    end

endmodule
```

4.2 2. Design code

```
  module config_reg(
        input  logic clk,
        input  logic reset,
        input  logic write,
        input  logic [15:0] data_in,
        input  logic [2:0] address,
        output logic [15:0] data_out
);

`protected

    MTI!#DU;Wt7W\E}vp$.h,RJ@F"cok2*:fU|]?KQ7"kDn2YBkJW1x7+D][a92\T\Xo+#4'mT1<VEU
    Y>-jEo-mnC;onUwzW]kB*s~TBlVA61@Ba}+7?|,5XoUXY[@7?]AEK#5u2[,vCY?5WACBAkE-5Q<A
    \1das*pl7+WpVrAz@eu$WA{ebXUv5TrV[v$A1+IB~*kJK~TXv^VW@]DQ]uzjv(G,i{LY]DZ$4^O,
    {cQ#=]/_xJpe_kwG?wE4qLAR5TU$3sI(^E#*,iTK7'RVBKXe\YOiZwxmlmAjGo2]1_1uC2}##l#I
    P#a7n^|e1lLPZ*l[3rTr=iEkb@T>J-pmux,$Z.\vQ2[-lJajw\sjirkh-'$[Ko7r*\?+$WWAB_~,
    ?e_<@'aB.a=Yi7aXRznB;\7ou~=!OV-p2^mDDu'sRiAn!m[J{Ho#-e1=Y7VDVT[G?v=$nL[:Kaw!
    EZTH}xUz5KaliVG][KZ2evwm~|tZIk3xeGs2o[GG]p[BAw!w>eQaTz*R1#TxX{n>^}!EC$ekE#Z7
    ,B5sWuxN^H\2w>3Y=2El3}xW2xWQk5[#Qviz*w1@s+^\UYY=-_nUw5s@-V2B-CjTp]Rw)e<UawXJ
    3+H*vpW@U{x{$R@W^$>,]E$v#BR{a+,uWuA}WxQ}X'JX[O5xkbl#B#/Dxo[*wV,FqZXmAy=I-\^n
    Br*@5+[Jzoh,$11KG>^9J*e~Hr?'TCNH}Bw~1[u@e$u>Q23#BR'$Wxl_]IpI*{J5?OKDrHkA$GsG
    a<j^D{*'Q^XzeaXL$Bsow<3*5lXBz*W*Z$CXMPK^zCvpDU5DT1aTJ=_Das=>o-TCs?aDE1}U5irC
    <A,~vQi}QOuo;!VY@]C-+ROh'?\Ku'pxvE!2(.Ss3JD'Q'eD?Esjp^[6]Z+u3n1]1urmi}@$eiDk
    l2\+73l-ExuR,n<*I!oW^;1BE;*DBxTpVWXBhq][<}=KoTyU>2jsY}R!U[sD+O>_{1U^Ev!27T\Y
    s}1q*$5T6kv!!hLEIwsx'CXe#GeQ*Y'[OvRkpr\a1oxGHpoW'lj}xRpjG!oTem?Q,mIp'7^11G]p
    A*mTLj;;\jrV[V*ps_T*jdrBX]tKE?X#n@<W[kp5XH]FNWoTBC]]Ud;5D[?][5D15}Tt_{]o#-Q*
    ^ep=(IH<oPD}QBJr<1p-'{sm5z@HBG,#rCiEQY}_1r[>;rr]u,wCKpGu}*w>lvvBr#j>s3Bz^I2x
    <\jOW1@Ii!H^[;!R=[}7iu=Z[am[jT>B>IkYmQVyGO,VAQ\E\[Y2*RR=w}o_O$BwT$I3G_OI[HT'
    3v@DZH1kp'@HO~EoI"'Kj3Wo'3*={jE$m*bo_}xBVs]E[psIrjABY+s]usrsi!\L#_#J/Vm\HVCe
␣␣␣␣<o}K]@5,Yo!+Re77K>G-'^#si7;KBo[a@=i
`endprotected


endmodule␣//␣test
```

4.3 3. Bug report

```
 Bug 1
    a) Design Input for Bug to Appear:
        Read ABCD from register ANALOG_TEST after reset.

    b) Expected Behavior:
        Register ANALOG_TEST should be readable as ABCD.

    c) Observed Behavior:
        Register ANALOG_TEST reads as ABCC instead of ABCD.

Bug 2
    a) Design Input for Bug to Appear:
        Write 0000 to register ADC0_REG.

    b) Expected Behavior:
        Register ADC0_REG should be writable as 0000.

    c) Observed Behavior:
        Register ADC0_REG is written as FFFF instead of 0000.

Bug 3
    a) Design Input for Bug to Appear:
        Write FFFF to register TEMP_SENSOR0_REG.

    b) Expected Behavior:
        Register TEMP_SENSOR0_REG should be writable as FFFF.

    c) Observed Behavior:
        Register TEMP_SENSOR0_REG is written as FFFE instead of FFFF.

Bug 4
    a) Design Input for Bug to Appear:
        Write FFFF to register DIGITAL_TEST.

    b) Expected Behavior:
        Register DIGITAL_TEST should be writable as FFFF.

    c) Observed Behavior:
        Register DIGITAL_TEST is written as 0000 instead of FFFF.

Bug 5
    a) Design Input for Bug to Appear:
        Write FFFF to register DIGITAL_CONFIG.

    b) Expected Behavior:
        Register DIGITAL_CONFIG should be writable as FFFF.

    c) Observed Behavior:
        Register DIGITAL_CONFIG is written as 7FFF instead of FFFF.

Bug 6
    a) Design Input for Bug to Appear:
        Write 37E7 to register ADC0_REG.

    b) Expected Behavior:
        Register ADC0_REG should be writable as 37E7.

    c) Observed Behavior:
        Register ADC0_REG is written as B7E7 instead of 37E7.

Bug 7
    a) Design Input for Bug to Appear:
        Write 37E7 to register ADC1_REG.

    b) Expected Behavior:
        Register ADC1_REG should be writable as 37E7.

    c) Observed Behavior:
        Register ADC1_REG is written as E737 instead of 37E7.

Bug 8
    a) Design Input for Bug to Appear:
        Write 37E7 to register TEMP_SENSOR0_REG.

    b) Expected Behavior:
        Register TEMP_SENSOR0_REG should be writable as 37E7.

    c) Observed Behavior:
        Register TEMP_SENSOR0_REG is written as 6FCE instead of 37E7.

Bug 9
    a) Design Input for Bug to Appear:
        Write 37E7 to register DIGITAL_TEST.

    b) Expected Behavior:
        Register DIGITAL_TEST should be writable as 37E7.

    c) Observed Behavior:
        Register DIGITAL_TEST is written as FFFF instead of 37E7.
```

4.4 4. Verification Plan

Label	Design Requirement Description	Stimulus Generation	Functional Coverage	Functionality Check
CONFIG_1	Upon reset, all registers should be set to their predefined reset values.	Directed reset at the start of the simulation, followed by random operations.	-	A checker in the testbench compares each register's output with the expected reset value from the associative array.
CONFIG_2	Writing to a register should update its value to the input data.	Randomized write operations with varying data inputs across all registers.	-	A checker verifies that the register's output matches the input data after a write operation.
CONFIG_3	Registers should maintain their values when no write operation is performed.	Randomized operations with a 50% chance of write being disabled.	-	A checker ensures that register values remain unchanged when write is not active.
CONFIG_4	The module should correctly handle simultaneous reset and write operations, prioritizing reset.	Directed tests where reset and write are asserted simultaneously.	-	A checker confirms that reset values take precedence over write operations when both are active.
CONFIG_5	The module should correctly handle edge cases, such as maximum and minimum data values.	Directed tests with data inputs set to all zeros, all ones, and random values.	-	A checker verifies correct register behavior for edge case data inputs.

Table 3: Verification Plan for Configuration Register Design

4.5 5. Waveform

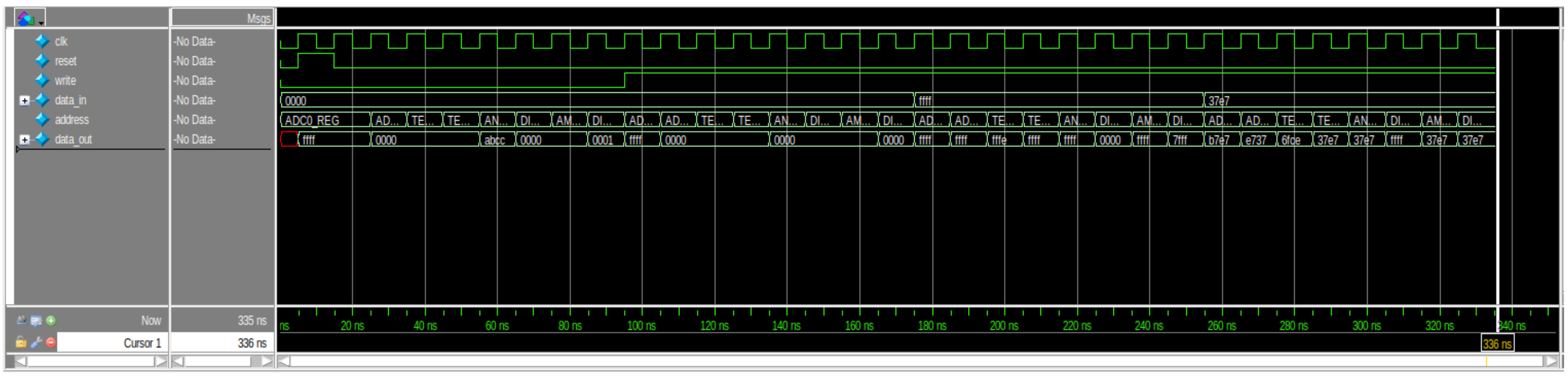
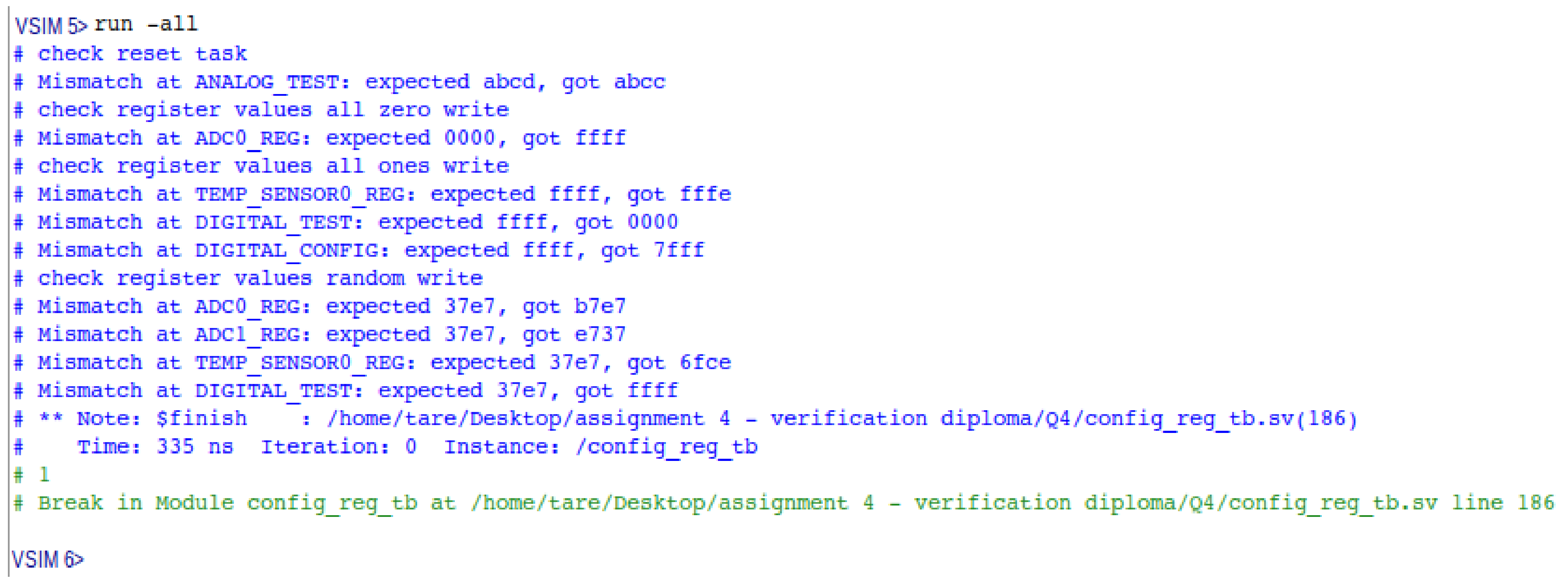


Figure 6: simulation waveform



```
VSIM 5> run -all
# check reset task
# Mismatch at ANALOG_TEST: expected abcd, got abcc
# check register values all zero write
# Mismatch at ADC0_REG: expected 0000, got ffff
# check register values all ones write
# Mismatch at TEMP_SENSOR0_REG: expected ffff, got fffe
# Mismatch at DIGITAL_TEST: expected ffff, got 0000
# Mismatch at DIGITAL_CONFIG: expected ffff, got 7fff
# check register values random write
# Mismatch at ADC0_REG: expected 37e7, got b7e7
# Mismatch at ADC1_REG: expected 37e7, got e737
# Mismatch at TEMP_SENSOR0_REG: expected 37e7, got 6fce
# Mismatch at DIGITAL_TEST: expected 37e7, got ffff
# ** Note: $finish    : /home/tare/Desktop/assignment 4 - verification diploma/Q4/config_reg_tb.sv(186)
#    Time: 335 ns  Iteration: 0  Instance: /config_reg_tb
# 1
# Break in Module config_reg_tb at /home/tare/Desktop/assignment 4 - verification diploma/Q4/config_reg_tb.sv line 186

VSIM 6>
```

Figure 7: Transcript : all test cases passed