Name: Abdelrahman Ahmed

# Assignment ONE
# (Verification)

# First Task

## The code edited to make sure that coverage is 100%:

```
task assert_reset;
    // Toggle from 0 to 1
    reset = 0;
    @(negedge clk);
    reset = 1;  // This line explicitly creates a 0 -> 1 toggle
    @(negedge clk);

    // Check the reset functionality
    if (C != 5'b0) begin
        $display("Reset Error");
        ErrorCount++;
    end else
        CorrectCount++;

    // Toggle back to 0
    reset = 0;
endtask
```

commands.do

```
1  vlib work
2  vlog lab_1.v  lab_1_tb.sv +cover -covercells
3  vsim -voptargs=+acc work.adder_tb -cover
4  add wave *
5  coverage save adder_tb.ucdb -onexit -du work.adder
6  run -all
```

Coverage Report:

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

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

Toggle Coverage for instance /\adder_tb#a1  --

                                          Node      1H->0L      0L->1H  "Coverage"
                                          ----------------------------------------
                                        A[0-3]           1           1      100.00
                                        B[0-3]           1           1      100.00
                                        C[4-0]           1           1      100.00
                                           clk           1           1      100.00
                                         reset           1           1      100.00

Total Node Count     =         15
Toggled Node Count   =         15
Untoggled Node Count =          0

Toggle Coverage      =     100.00% (30 of 30 bins)


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

# Second Task

priority_enc.v > ...

```v
module priority_enc (
input  clk,
input  rst,
input  [3:0] D,
output reg  [1:0] Y,
output reg valid
);

always @(posedge clk) begin
  if (rst)
     Y <= 2'b0;
  else
    casex (D)
        4'b1000: Y <= 0;
        4'bX100: Y <= 1;
        4'bXX10: Y <= 2;
        4'bXXX1: Y <= 3;
    endcase
    valid <= (~|D)? 1'b0: 1'b1;
end
endmodule

```

## Test:

priority_tb.sv > priority_enc_tb

```sv
module priority_enc_tb();

reg clk, rst;
reg [3:0] D;
wire [1:0] Y;
wire valid;


priority_enc pe(.*);


integer ErrorCount = 0, CorrectCount = 0;

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

initial begin
    D = 4'b0;
    assert_reset;

    D = 4'b1000;
    check_outputs(2'b0, 1);

    D = 4'b0100;
    check_outputs(2'b01, 1);
    D = 4'b1100;
    check_outputs(2'b01, 1);

    D = 4'b0010;
    check_outputs(2'b10, 1);
    D = 4'b1100;
    check_outputs(2'b01, 1);

    D = 4'b0110;
    check_outputs(2'b10, 1);
    D = 4'b1010;
    check_outputs(2'b10, 1);
```

```
    check_outputs(2'b10, 1);
    D = 4'b1110;

    check_outputs(2'b10, 1);
    D = 4'b0001;
    check_outputs(2'b11, 1);
    D = 4'b0011;
    check_outputs(2'b11, 1);
    D = 4'b0101;

    check_outputs(2'b11, 1);
    D = 4'b1001;
    check_outputs(2'b11, 1);
    D = 4'b1101;

    check_outputs(2'b11, 1);
    D = 4'b0111;
    check_outputs(2'b11, 1);
    D = 4'b1111;
    check_outputs(2'b11, 1);

    D = 4'b1011;
    check_outputs(2'b11, 1);
    D = 4'b0;
    check_outputs(2'bXX, 0);

    $display("Number of -correct checks = %0d, - Number of Errors = %0d", CorrectCount, ErrorCount);
    $stop;
end

task check_outputs(input [3:0] Y_expected, input valid_expected);
    @(negedge clk);
    if (Y != Y_expected && valid_expected != valid)begin
        $display("Error in Y_expected = %b, Y = %b, valid = %b, valid_ex = %b", Y_expected, Y, valid, valid_expected);
        ErrorCount++;
    end
    else begin
        CorrectCount++;
    end
endtask
```

```
task assert_reset;
    rst = 0;
    @(negedge clk);
    rst = 1;
    @(negedge clk);
    if (Y != 2'b0 && valid != 0)begin
        $display("Error in rst activation");
        ErrorCount++;
    end
    rst = 0;
endtask

endmodule
```

Transcript:

```
# Loading work.priority_enc_tb(fast)
# Loading work.priority_enc(fast)
# Number of -correct checks = 16, - Number of Errors = 0
# ** Note: $stop    : priority_tb.sv(62)
#    Time: 72 ns  Iteration: 1  Instance: /priority_enc_tb
# Break in Module priority_enc_tb at priority_tb.sv line 62
```

Code Coverage Report:

```
================================================================================
Branch Coverage:
    Enabled Coverage             Bins      Hits    Misses  Coverage
    ----------------             ----      ----    ------  --------
    Branches                        7         7         0   100.00%

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

Branch Coverage for instance /\priority_enc_tb#pe

    Line         Item                    Count     Source
    ----         ----                    -----     ------
  File priority_enc.v
-----------------------------------IF Branch-----------------------------------
    10                                      19     Count coming in to IF
    10              1                        1       if (rst)
    12              1                       18       else
Branch totals: 2 hits of 2 branches = 100.00%

---------------------------------CASE Branch-----------------------------------
    13                                      18     Count coming in to CASE
    14              1                        1              4'b1000: Y <= 0;
    15              1                        3              4'bX100: Y <= 1;
    16              1                        4              4'bXX10: Y <= 2;
    17              1                        8              4'bXXX1: Y <= 3;
                                             2     All False Count
Branch totals: 5 hits of 5 branches = 100.00%

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

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

  Statement Coverage for instance /\priority_enc_tb#pe  --

      Line         Item                      Count     Source
      ----         ----                      -----     ------
    File priority_enc.v
      1                                                module priority_enc (
      2                                                input  clk,
      3                                                input  rst,
      4                                                input  [3:0] D,
      5                                                output reg  [1:0] Y,
      6                                                output reg valid
      7                                                );
      8
      9               1                          19    always @(posedge clk) begin
      10                                                 if (rst)
      11              1                           1        Y <= 2'b0;
      12                                                 else
      13                                                  casex (D)
      14              1                           1                4'b1000: Y <= 0;
      15              1                           3                4'bX100: Y <= 1;
      16              1                           4                4'bXX10: Y <= 2;
      17              1                           8                4'bXXX1: Y <= 3;
      18                                                  endcase
      19              1                          19        valid <= (~|D)? 1'b0: 1'b1;

  Toggle Coverage:
      Enabled Coverage              Bins      Hits    Misses  Coverage
      ----------------              ----      ----    ------  --------
      Toggles                         18        18         0   100.00%

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

  Toggle Coverage for instance /\priority_enc_tb#pe  --

                                        Node      1H->0L      0L->1H  "Coverage"
                                        ----------------------------------------
                                      D[0-3]           1           1      100.00
                                      Y[1-0]           1           1      100.00
                                         clk           1           1      100.00
                                         rst           1           1      100.00
                                       valid           1           1      100.00

  Total Node Count     =          9
  Toggled Node Count   =          9
  Untoggled Node Count =          0

  Toggle Coverage      =     100.00% (18 of 18 bins)


  Total Coverage By Instance (filtered view): 100.00%

  End time: 01:06:41 on Mar 07,2025, Elapsed time: 0:00:00
  Errors: 0, Warnings: 0
```

# Task 3

ALU.v

```v
module ALU_4_bit (
    input  clk,
    input  reset,
    input  [1:0] Opcode,   // The opcode
    input  signed [3:0] A, // Input data A in 2's complement
    input  signed [3:0] B, // Input data B in 2's complement

    output reg signed [4:0] C // ALU output in 2's complement

      );

    reg signed [4:0]      Alu_out; // ALU output in 2's complement

    localparam         Add             = 2'b00; // A + B
    localparam         Sub             = 2'b01; // A - B
    localparam         Not_A           = 2'b10; // ~A
    localparam         ReductionOR_B   = 2'b11; // |B

    // Do the operation
    always @* begin
       case (Opcode)
         Add:            Alu_out = A + B;
         Sub:            Alu_out = A - B;
         Not_A:          Alu_out = ~A;
         ReductionOR_B:  Alu_out = |B;
        default:  Alu_out = 5'b0;
       endcase
    end // always @ *

    // Register output C
    always @(posedge clk or posedge reset) begin
       if (reset)
         C <= 5'b0;
       else
         C<= Alu_out;
    end

endmodule

```

The Perfect Testbench ever:

ALU_tb.sv > ALU_4_bit_tb

```v
module ALU_4_bit_tb;
reg clk;
reg reset;
reg [1:0] Opcode;   // The opcode
reg signed [3:0] A; //data A in 2's complement
reg signed [3:0] B; // Input data B in 2's complement
wire signed [4:0] C; // ALU output in 2's complement

localparam MAXPOS = 7, ZERO = 0, MAXNEG = -8;

int ErrorCount = 0;
int CorrectCount = 0;

ALU_4_bit alu4(.*);

initial begin
    clk = 0;

    forever begin
        #10 clk = ~clk;
    end
end

initial begin
    A = 0;
    B = 0;
    Opcode = 0;
    assert_reset;
    Opcode_00;
    Opcode_01;
    Opcode_10;
    Opcode_11;

    /////// Just to make 100% Code coverage
    Opcode_00;
    ///////

    Opcode_xx;

    $display("corrects = %0d, Errors = %0d", CorrectCount, ErrorCount);
    $stop;
end
```

this is the main testbench program

You must guessed that Opcode_00,  Opcode_01 , Opcode_10, Opcode_11, and Opcode_xx
Are just **TASKS**
Here are the declaration of all tasks:

```
////////////////////////////////////////////////////
task assert_reset;
    // Toggle from 0 to 1
    reset = 0;
    @(negedge clk);
    reset = 1;  // This line explicitly creates a 0 -> 1 toggle
    @(negedge clk);

    // Check the reset functionality
    if (C != 5'b0) begin
        $display("Reset Error");
        ErrorCount++;
    end else
        CorrectCount++;

    // Toggle back to 0
    reset = 0;
endtask
////////////////////////////////////////////////////////
task check_result(input signed [4:0] value);
    @(negedge clk);
    if (C !== value)begin
        $display("Error A = %0d, b = %0d, c = %0d, value = %0d", A, B, C, value);
        ErrorCount++;
    end
    else begin
        CorrectCount++;
    end
    reset = 0;
endtask
////////////////////////////////////////////////////////
```

```
  ////////////////////////////////////////////////////////

  task Opcode_00;
      Opcode = 2'b00;
      A = MAXPOS; B = MAXPOS;
      check_result(14);
      A = MAXPOS; B = MAXNEG;
      check_result(-1);
      A = MAXNEG; B = MAXPOS;
      check_result(-1);
      A = MAXNEG; B = MAXNEG;
      check_result(-16);
      A = MAXPOS; B = ZERO;
      check_result(7);
      A = ZERO; B = MAXPOS;
      check_result(7);
      A = MAXNEG; B = ZERO;
      check_result(-8);
      A = ZERO; B = MAXNEG;
      check_result(-8);
      A = ZERO; B = ZERO;
      check_result(0);
  endtask
  ////////////////////////////////////////////////////////
```

```
////////////////////////////////////////////////////////
task Opcode_01;
    Opcode = 2'b01;
    A = MAXPOS; B = MAXPOS;
    check_result(0);
    A = MAXPOS; B = MAXNEG;
    check_result(15);
    A = MAXNEG; B = MAXPOS;
    check_result(-15);
    A = MAXNEG; B = MAXNEG;
    check_result(0);
    A = MAXPOS; B = ZERO;
    check_result(7);
    A = ZERO; B = MAXPOS;
    check_result(-7);
    A = MAXNEG; B = ZERO;
    check_result(-8);
    A = ZERO; B = MAXNEG;
    check_result(8);
    A = ZERO; B = ZERO;
    check_result(0);
endtask
////////////////////////////////////////////////////////
```

```
////////////////////////////////////////////////////////
task Opcode_10;
    Opcode = 2'b10;
    A = MAXPOS; B = MAXPOS;
    check_result(MAXNEG);
    A = MAXPOS; B = MAXNEG;
    check_result(MAXNEG);
    A = MAXNEG; B = MAXPOS;
    check_result(MAXPOS);
    A = MAXNEG; B = MAXNEG;
    check_result(MAXPOS);
    A = MAXPOS; B = ZERO;
    check_result(MAXNEG);
    A = ZERO; B = MAXPOS;
    check_result(-1);
    A = MAXNEG; B = ZERO;
    check_result(MAXPOS);
    A = ZERO; B = MAXNEG;
    check_result(-1);
    A = ZERO; B = ZERO;
    check_result(-1);
endtask
////////////////////////////////////////////////////////
```

```
////////////////////////////////////////////////////////
task Opcode_11;
    Opcode = 2'b11;
    A = MAXPOS; B = MAXPOS;
    check_result(1);
    A = MAXPOS; B = MAXNEG;
    check_result(1);
    A = MAXNEG; B = MAXPOS;
    check_result(1);
    A = MAXNEG; B = MAXNEG;
    check_result(1);
    A = MAXPOS; B = ZERO;
    check_result(0);
    A = ZERO; B = MAXPOS;
    check_result(1);
    A = MAXNEG; B = ZERO;
    check_result(0);
    A = ZERO; B = MAXNEG;
    check_result(1);
    A = ZERO; B = ZERO;
    check_result(0);
endtask
////////////////////////////////////////////////////////
```

```
////////////////////////////////////////////////////
task Opcode_xx;
    Opcode = 2'bXX;
    A = MAXPOS; B = MAXPOS;
    check_result(0);
    A = MAXPOS; B = MAXNEG;
    check_result(0);
    A = MAXNEG; B = MAXPOS;
    check_result(0);
    A = MAXNEG; B = MAXNEG;
    check_result(0);
    A = MAXPOS; B = ZERO;
    check_result(0);
    A = ZERO; B = MAXPOS;
    check_result(0);
    A = MAXNEG; B = ZERO;
    check_result(0);
    A = ZERO; B = MAXNEG;
    check_result(0);
    A = ZERO; B = ZERO;
    check_result(0);
endtask
```

```
# Loading work.ALU_4_bit(fast)
# corrects = 55, Errors = 0
# ** Note: $stop    : ALU_tb.sv(41)
#    Time: 1120 ns  Iteration: 1  Instance: /A
```

**100% code coverage**

```
Toggle Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Toggles                         44        44         0   100.00%

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

Toggle Coverage for instance /\ALU_4_bit_tb#alu4  --

                                          Node      1H->0L      0L->1H  "Coverage"
                                          ----------------------------------------
                                        A[0-3]           1           1      100.00
                                    Alu_out[4-0]         1           1      100.00
                                        B[0-3]           1           1      100.00
                                        C[4-0]           1           1      100.00
                                     Opcode[0-1]         1           1      100.00
                                           clk           1           1      100.00
                                         reset           1           1      100.00

Total Node Count     =         22
Toggled Node Count   =         22
Untoggled Node Count =          0

Toggle Coverage      =     100.00% (44 of 44 bins)


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

TASK 4

Sorry for not doing that….