System Verilog

Randomization & Functional Coverage

Assignment 3 Extra

By: Magdy Ahmed Abbas Abdelhamid

Question 1

COUNTER

SV Randomization & Functional Coverage – Extra

Q1. Verify the functionality of the following counter:

- **Parameters:**
    1. WIDTH: width of the data_load and count_out ports (Valid values: 4, 6, 8, default: 4)
- **Inputs**:
    1. clk
    2. rst_n (active low sync rst)
    3. load_n (active low load)
    4. up_down (When the input is high then increment counter, else decrement the counter)
    5. ce (enable signal to increment or decrement the counter depending on the up_down)
    6. data_load (load data to count_out output when the load signal is asserted)
- **Outputs**:
    1. count_out (counter output)
    2. max_count (When the counter reaches the maximum value, this signal is high, else low)
    3. zero (When the counter reaches the minimum value, this signal is high, else low)

Requirements:

1. List your verification plan items.
2. Create a verification requirement document based on your verification plan items to support your verification planning, an example of the document can be found in the link here. Please copy this document to have your own version and fill the document with the design requirements.
3. Constrained Randomization
    a. Constraint the reset to be deactivated most of the time
    b. Constraint the load signal to be active 70% of the time
    c. Constraint the enable signal to be active 70% of the time
4. Functional Coverage
    a. Coverpoint for load data when load is asserted
    b. Coverpoint for count out if the reset is deasserted, enable is active and up_down is high
        i. Autogenerate bins for all values
    c. Coverpoint for count out if the reset is deasserted, enable is active and up_down is high
        i. Transition bin to check when overflow occurs (maximum value to zero)
    d. Coverpoint for count out if the reset is deasserted, enable is active and up_down is low
        i. Autogenerate bins for all values
    e. Coverpoint for count out if the reset is deasserted, enable is active and up_down is low
        i. Transition bin to check when underflow occurs (zero to maximum value)

Verification Plan

Test Plan:

- **In case of the activation of the reset (reset) the output must be zero until positive edge of the clock arrival as it is synchronous reset.**
  - we don't care about the randomization of all input signals,
    but we care about the constraint randomization for the reset
    (as it must be deactivated most of time)
  - There is no cover point for reset signal in our verification design.
  - Output Checked against golden mode.

- **In case of deactivation for the reset and activate the load signal where in this case we expect that the output will take the data_load value.**
  - Randomization under constraints on the load_n signal to be activated 70% of the time simulation.
  - we create cover point for data_load values as in the default configuration there will be auto 16 bins have to be hit.
  - Output Checked against golden model.

- **In case of counting up or down we have to deactivate reset and load and activate the enable signal to inform the simulator that we shall count not load the output with the data.**
  - Randomization under constraints for the enable signal to be activated most of time (70% of time simulation).
  - there is no cover point for enable signal in our verification design.
  - Output Checked against golden mode.

- **for the output count_out, it will depend on the input values.**
  - Randomization for all input signals.
  - Included in a coverpoint for count_out where we create four cover points for it.
  - Output Checked against golden model.

Technique for Testing Feature:

- **Randomize Testing.**
- **Self - Checking.**
- **Verification IP, Block Level.**

Counter Design

```
////////////////////////////////////////////////////////////////////////////
// Author: Kareem Waseem
// Course: Digital Verification using SV & UVM
//
// Description: Counter Design
//
////////////////////////////////////////////////////////////////////////////

module counter (clk , rst_n, load_n, up_down, ce, data_load, count_out, max_count, zero);

parameter WIDTH = 4;
input clk;
input rst_n;
input load_n;
input up_down;
input ce;
input [WIDTH − 1: 0] data_load;

output reg [WIDTH − 1: 0] count_out;
output max_count;
output zero;

always @(posedge clk) begin
   if (! rst_n)
      count_out <= 0;
   else if (! load_n)
      count_out <= data_load;
   else if (ce)
      if (up_down)
         count_out <= count_out + 1;
      else
         count_out <= count_out − 1;
end

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

endmodule
```

Counter Package

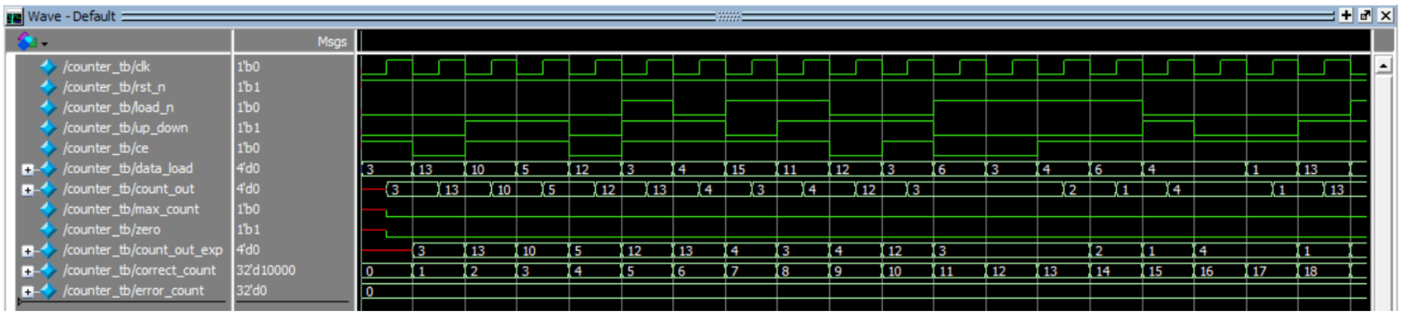
```
package pkg ;

// parameters declaration
parameter SIZE  =  4 ;
localparam MAX_COUNT  =  {SIZE{1'b1}} ;
localparam ZERO  =  0 ;

class test ;

rand bit rst_n_c , load_n_c , ce_c , up_down_c ;
rand bit [SIZE − 1: 0] data_load_c;
logic [SIZE − 1: 0] count_out_c ;


constraint c_rst { rst_n_c dist {0: = 5 , 1: = 95}; }

constraint c_load { load_n_c dist {0:/70 , 1:/30}; }

constraint c_enable { ce_c dist {0:/30 , 1:/70}; }

covergroup cg ;
   load_cp :  coverpoint data_load_c iff(! load_n_c);
   count_up_cp1 :  coverpoint count_out_c iff(rst_n_c && ce_c && up_down_c);
   count_up_cp2 :  coverpoint count_out_c iff(rst_n_c && ce_c && up_down_c)
   {
      bins Bins_trans_overflow  =  (MAX_COUNT  =>  ZERO) ;
   }
   count_down_cp1 :  coverpoint count_out_c iff(rst_n_c && ce_c && ! up_down_c);
   count_down_cp2 :  coverpoint count_out_c iff(rst_n_c && ce_c && ! up_down_c)
   {
      bins Bins_trans_underflow  =  (ZERO  =>  MAX_COUNT) ;
   }
endgroup

function new ;
   cg  =  new();
endfunction

endclass
endpackage
```

Counter Testbench

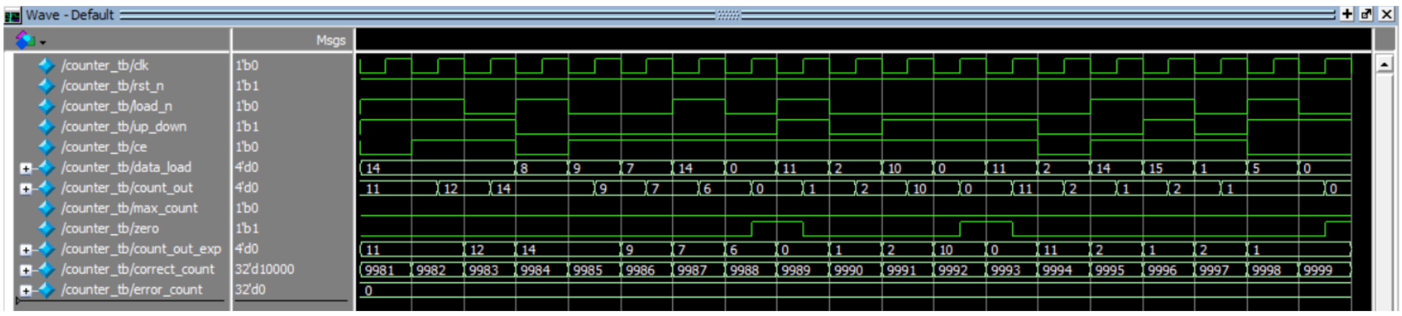
```
import pkg ::* ;
module counter_tb ;
// input and output declaration
logic clk , rst_n , load_n , up_down , ce ;
logic [SIZE - 1: 0] data_load;
logic [SIZE - 1: 0] count_out;
logic max_count , zero;
logic [SIZE - 1: 0] count_out_exp;  // expected value for output count
// counters declaration
int correct_count , error_count ;
counter #(. WIDTH(SIZE)) DUT (.*);
// clock declaration
initial begin
  clk  =  0 ;
  forever
  #25 clk  =  ~clk ;   // clock period  =  50 ns
end
test obj1 ;  // create object from the class
initial begin
  obj1  =  new();
  correct_count  =  0 ;
  error_count  =  0 ;
  for(int i  = 0; i < 10000; i + +) begin
    assert(obj1. randomize());
    rst_n  =  obj1. rst_n_c;
    load_n  =  obj1. load_n_c;
    up_down  =  obj1. up_down_c;
    ce  =  obj1. ce_c;
    data_load  =  obj1. data_load_c;
    @(negedge clk);       // this delay is to sample the correct output
    obj1. count_out_c  =  count_out;
    obj1. cg. sample();         // we sample the data after each randomized iteration
    self_check();
  end
  $display("correct counter  =  %0d , error counter  =  %0d",
                                                         correct_count, error_count);
  $stop;
end
```

```
task self_check();
  if(! rst_n)
    count_out_exp  =  0;
  else if(! load_n)
    count_out_exp  =  data_load ;
  else if(ce) begin
    if(up_down)
      count_out_exp  =  count_out_exp  +  1;
    else
      count_out_exp  =  count_out_exp  –  1;
  end
  if(count_out ! ==  count_out_exp) begin
    $display("the counter doesn't operate correctly");
    error_count + +;
  end   else
    correct_count + +;
endtask
endmodule
```

Waveform



```
Wave - Default
                              Msgs
/counter_tb/clk               1'b0
/counter_tb/rst_n             1'b1
/counter_tb/load_n            1'b0
/counter_tb/up_down           1'b1
/counter_tb/ce                1'b0
/counter_tb/data_load         4'd0
/counter_tb/count_out         4'd0
/counter_tb/max_count         1'b0
/counter_tb/zero              1'b1
/counter_tb/count_out_exp     4'd0
/counter_tb/correct_count     32'd10000
/counter_tb/error_count       32'd0
```



```
Wave - Default
                              Msgs
/counter_tb/clk               1'b0
/counter_tb/rst_n             1'b1
/counter_tb/load_n            1'b0
/counter_tb/up_down           1'b1
/counter_tb/ce                1'b0
/counter_tb/data_load         4'd0
/counter_tb/count_out         4'd0
/counter_tb/max_count         1'b0
/counter_tb/zero              1'b1
/counter_tb/count_out_exp     4'd0
/counter_tb/correct_count     32'd10000
/counter_tb/error_count       32'd0
```

Transcript

```
# correct counter = 10000 , error counter = 0
```

Do File

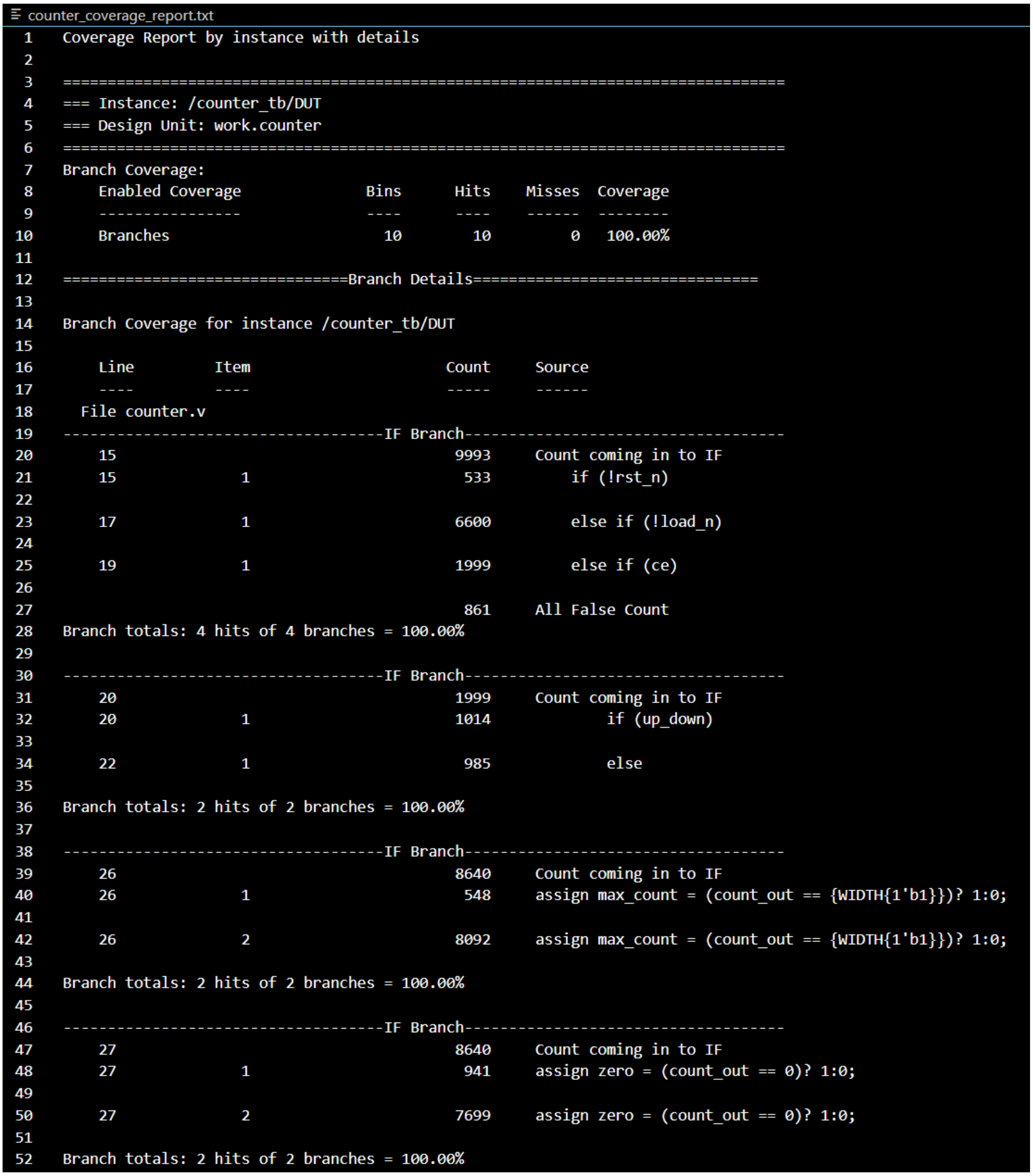
Do File for (Counter) Question 1

```
vlib work
vlog counter. v package. sv counter_tb. sv  + cover  − covercells
vsim  − voptargs = +acc work. counter_tb  − cover
add wave  ∗
coverage save counter_tb. ucdb  − onexit
run  − all
```

Counter Verification Requirements

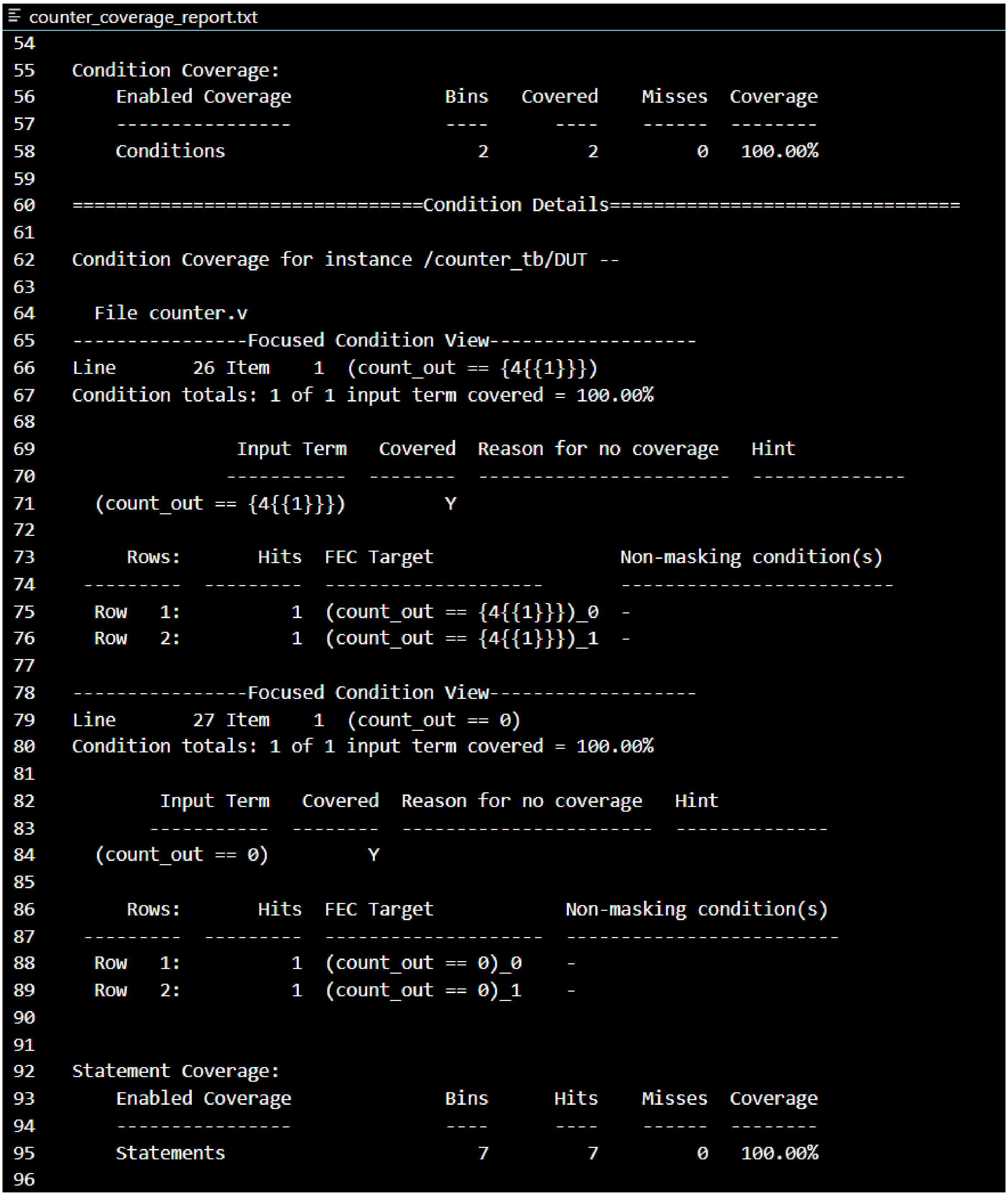
Label	Description	Stimulus Generation	Functional Coverage	Functionality Check
counter_1	Incase of the activation of the reset (reset) the output must be zero untill positive edge of the clock arrival as it is synchronous reset	we don't care about the randomization of all input siganls but we care about the constraint randomization for the reset (as it must be deactivated most of time)	there is no cover point for reset signal in our verification design	Output Checked against golden model
counter_2	Incase of deactivation for the reset and activate the load signal where in this case we expect that the output will take the data_load value	Randomization under constraints on the load_n signal to be activated 70% of the time simulation	we create cover point for data_load values as in the default configuration there will be auto 16 bins have to be hit	Output Checked against golden model
counter_3	In case of counting up or down we have to deactivate reset and load and activate the enable signal to inform the simulator that we shall count not load the output with the data	Randomization under constraints for the enable signal to be activated most of time (70% of time simulation)	there is no cover point for enable signal in our verification design	Output Checked against golden model
counter_4	for the output count_out , it will depend on the input values	Randomization for all input signals	Included in a coverpoint for count_out where we create four cover points for it	Output Checked against golden model

CODE COVERAGE

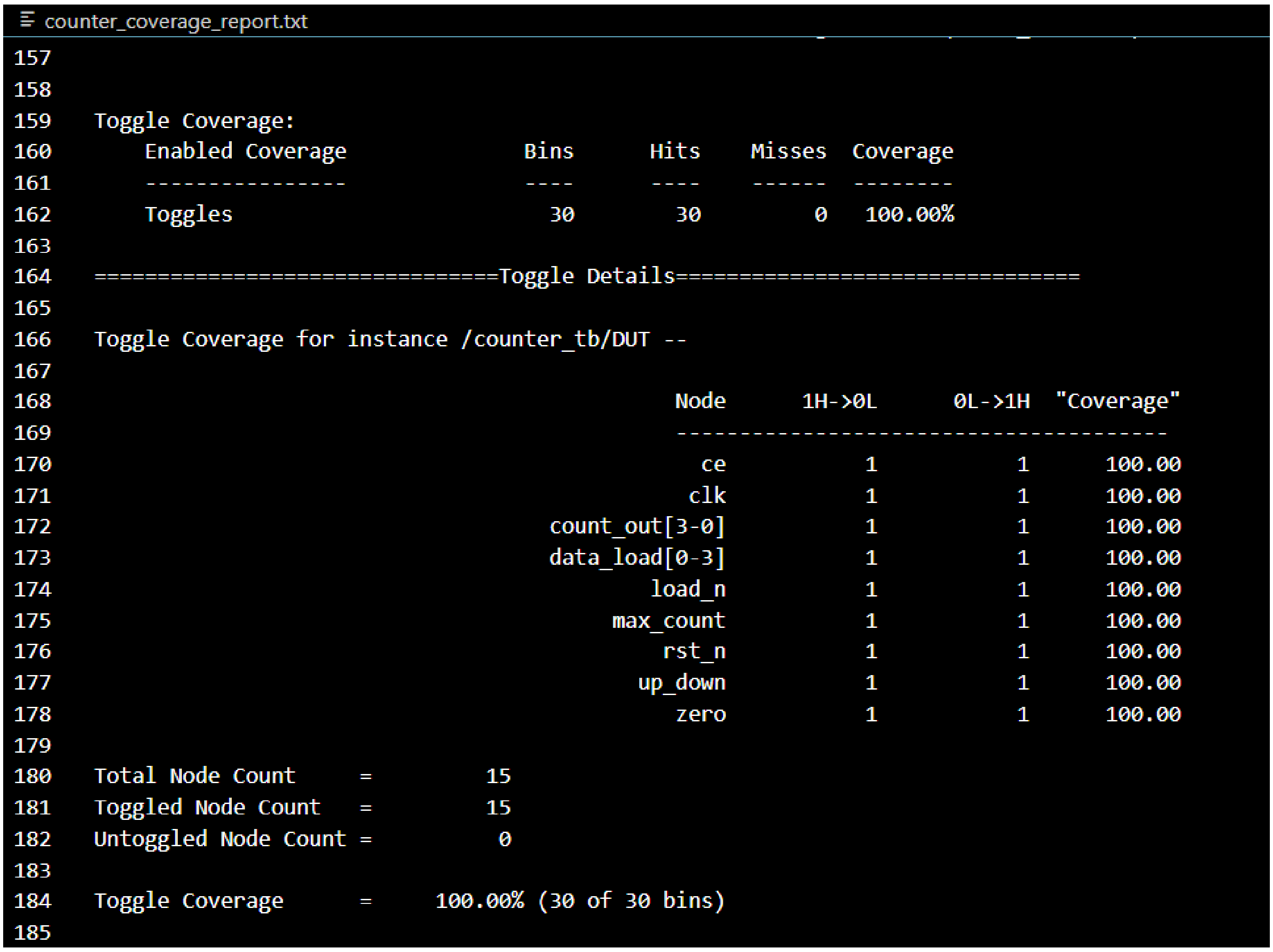


counter_coverage_report.txt

```
Coverage Report by instance with details

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

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

Branch Coverage for instance /counter_tb/DUT

    Line         Item                      Count     Source
    ----         ----                      -----     ------
  File counter.v
------------------------------------IF Branch------------------------------------
    15                                      9993     Count coming in to IF
    15              1                        533         if (!rst_n)

    17              1                       6600         else if (!load_n)

    19              1                       1999         else if (ce)

                                             861     All False Count
Branch totals: 4 hits of 4 branches = 100.00%

------------------------------------IF Branch------------------------------------
    20                                      1999     Count coming in to IF
    20              1                       1014             if (up_down)

    22              1                        985             else

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
    26                                      8640     Count coming in to IF
    26              1                        548     assign max_count = (count_out == {WIDTH{1'b1}})? 1:0;

    26              2                       8092     assign max_count = (count_out == {WIDTH{1'b1}})? 1:0;

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
    27                                      8640     Count coming in to IF
    27              1                        941     assign zero = (count_out == 0)? 1:0;

    27              2                       7699     assign zero = (count_out == 0)? 1:0;

Branch totals: 2 hits of 2 branches = 100.00%
```


counter_coverage_report.txt

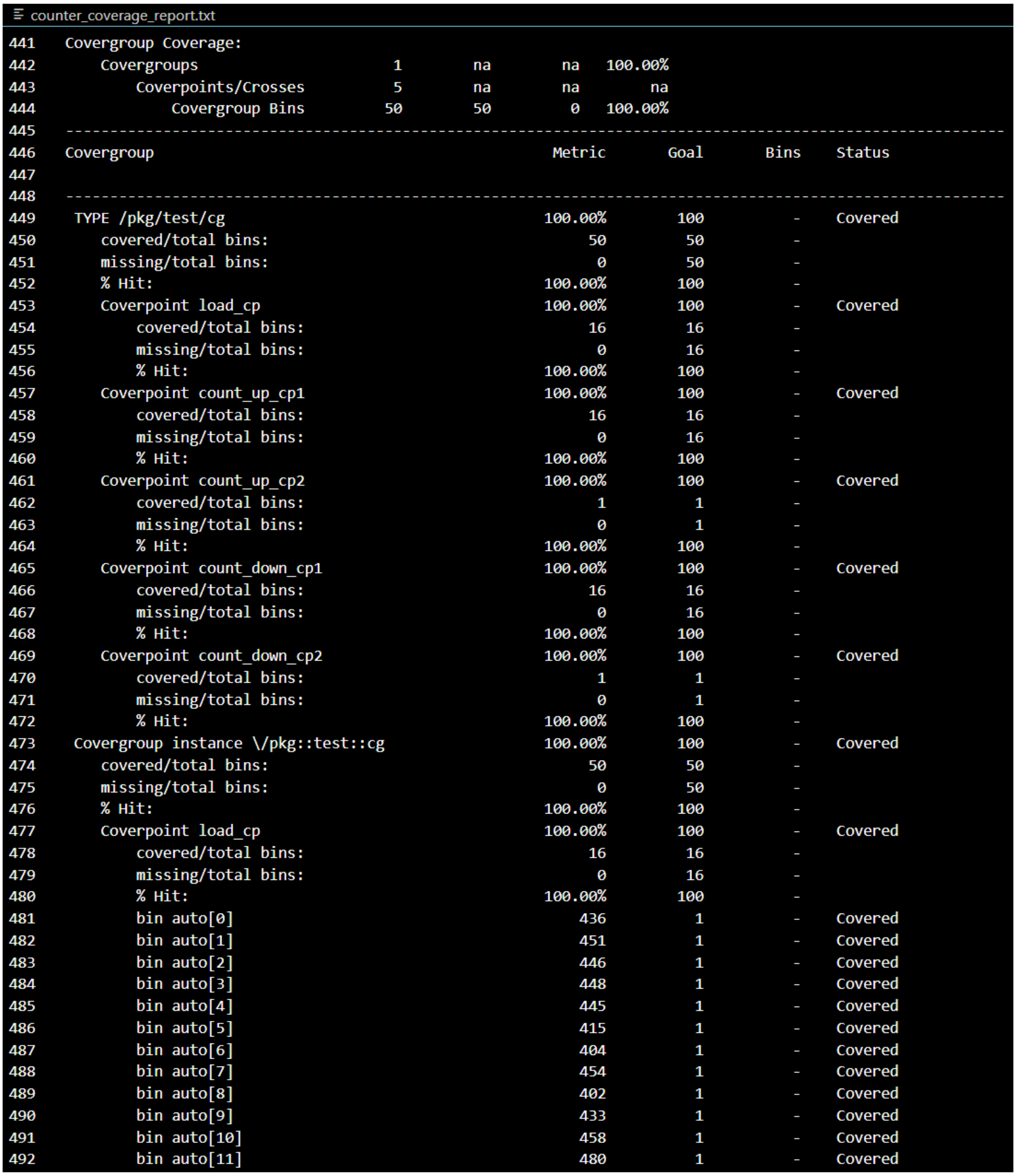
```

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

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

Condition Coverage for instance /counter_tb/DUT --

  File counter.v
----------------Focused Condition View-------------------
Line       26 Item    1  (count_out == {4{{1}}})
Condition totals: 1 of 1 input term covered = 100.00%

               Input Term   Covered  Reason for no coverage   Hint
              -----------  --------  ----------------------  --------------
  (count_out == {4{{1}}})         Y

     Rows:       Hits  FEC Target                Non-masking condition(s)
 ---------  ---------  --------------------      -------------------------
  Row   1:          1  (count_out == {4{{1}}})_0  -
  Row   2:          1  (count_out == {4{{1}}})_1  -

----------------Focused Condition View-------------------
Line       27 Item    1  (count_out == 0)
Condition totals: 1 of 1 input term covered = 100.00%

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

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


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

```



counter_coverage_report.txt

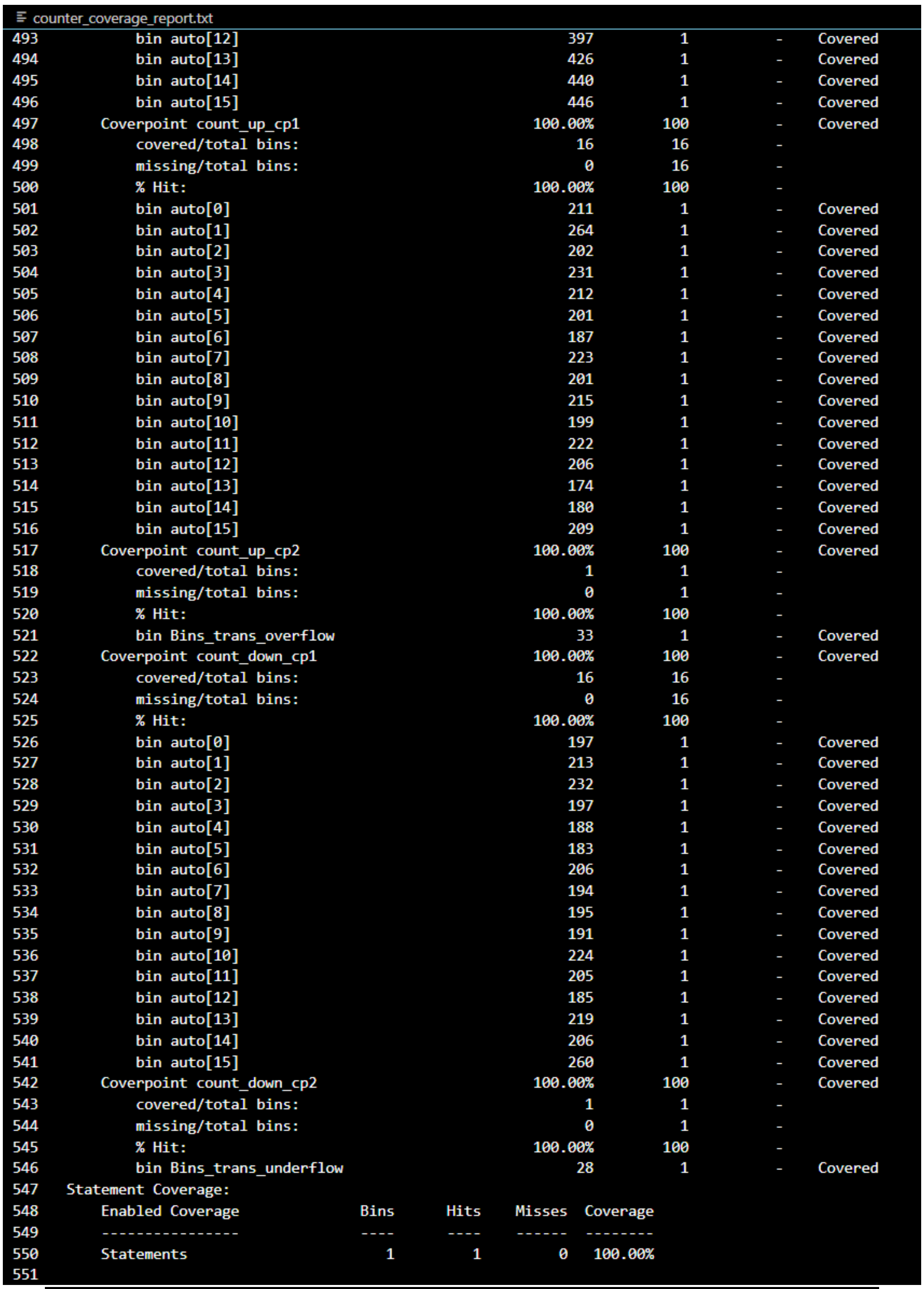
```


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

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

Toggle Coverage for instance /counter_tb/DUT --

                                              Node      1H->0L      0L->1H  "Coverage"
                                              ---------------------------------------
                                                ce           1           1      100.00
                                               clk           1           1      100.00
                                     count_out[3-0]          1           1      100.00
                                     data_load[0-3]          1           1      100.00
                                            load_n           1           1      100.00
                                         max_count           1           1      100.00
                                             rst_n           1           1      100.00
                                           up_down           1           1      100.00
                                              zero           1           1      100.00

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

Toggle Coverage      =     100.00% (30 of 30 bins)

```

Functional Coverage

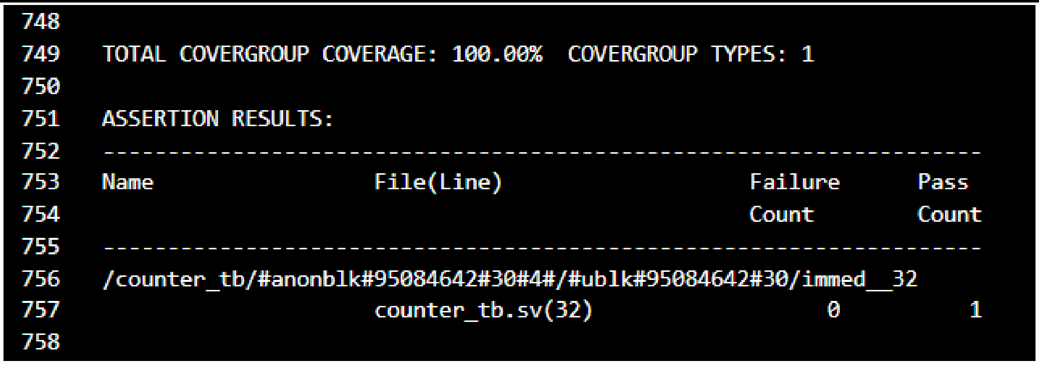


counter_coverage_report.txt

```
Covergroup Coverage:
    Covergroups                      1        na        na   100.00%
        Coverpoints/Crosses          5        na        na        na
            Covergroup Bins         50        50         0   100.00%

----------------------------------------------------------------------------------------------------------
Covergroup                                             Metric       Goal       Bins    Status


----------------------------------------------------------------------------------------------------------
 TYPE /pkg/test/cg                                    100.00%        100          -    Covered
    covered/total bins:                                    50         50          -
    missing/total bins:                                     0         50          -
    % Hit:                                            100.00%        100          -
    Coverpoint load_cp                                100.00%        100          -    Covered
        covered/total bins:                                16         16          -
        missing/total bins:                                 0         16          -
        % Hit:                                        100.00%        100          -
    Coverpoint count_up_cp1                           100.00%        100          -    Covered
        covered/total bins:                                16         16          -
        missing/total bins:                                 0         16          -
        % Hit:                                        100.00%        100          -
    Coverpoint count_up_cp2                           100.00%        100          -    Covered
        covered/total bins:                                 1          1          -
        missing/total bins:                                 0          1          -
        % Hit:                                        100.00%        100          -
    Coverpoint count_down_cp1                         100.00%        100          -    Covered
        covered/total bins:                                16         16          -
        missing/total bins:                                 0         16          -
        % Hit:                                        100.00%        100          -
    Coverpoint count_down_cp2                         100.00%        100          -    Covered
        covered/total bins:                                 1          1          -
        missing/total bins:                                 0          1          -
        % Hit:                                        100.00%        100          -
 Covergroup instance \/pkg::test::cg                  100.00%        100          -    Covered
    covered/total bins:                                    50         50          -
    missing/total bins:                                     0         50          -
    % Hit:                                            100.00%        100          -
    Coverpoint load_cp                                100.00%        100          -    Covered
        covered/total bins:                                16         16          -
        missing/total bins:                                 0         16          -
        % Hit:                                        100.00%        100          -
        bin auto[0]                                       436          1          -    Covered
        bin auto[1]                                       451          1          -    Covered
        bin auto[2]                                       446          1          -    Covered
        bin auto[3]                                       448          1          -    Covered
        bin auto[4]                                       445          1          -    Covered
        bin auto[5]                                       415          1          -    Covered
        bin auto[6]                                       404          1          -    Covered
        bin auto[7]                                       454          1          -    Covered
        bin auto[8]                                       402          1          -    Covered
        bin auto[9]                                       433          1          -    Covered
        bin auto[10]                                      458          1          -    Covered
        bin auto[11]                                      480          1          -    Covered
```


counter_coverage_report.txt

```
            bin auto[12]                                     397          1           -   Covered
            bin auto[13]                                     426          1           -   Covered
            bin auto[14]                                     440          1           -   Covered
            bin auto[15]                                     446          1           -   Covered
        Coverpoint count_up_cp1                          100.00%        100           -   Covered
            covered/total bins:                               16         16           -
            missing/total bins:                                0         16           -
            % Hit:                                       100.00%        100           -
            bin auto[0]                                      211          1           -   Covered
            bin auto[1]                                      264          1           -   Covered
            bin auto[2]                                      202          1           -   Covered
            bin auto[3]                                      231          1           -   Covered
            bin auto[4]                                      212          1           -   Covered
            bin auto[5]                                      201          1           -   Covered
            bin auto[6]                                      187          1           -   Covered
            bin auto[7]                                      223          1           -   Covered
            bin auto[8]                                      201          1           -   Covered
            bin auto[9]                                      215          1           -   Covered
            bin auto[10]                                     199          1           -   Covered
            bin auto[11]                                     222          1           -   Covered
            bin auto[12]                                     206          1           -   Covered
            bin auto[13]                                     174          1           -   Covered
            bin auto[14]                                     180          1           -   Covered
            bin auto[15]                                     209          1           -   Covered
        Coverpoint count_up_cp2                          100.00%        100           -   Covered
            covered/total bins:                                1          1           -
            missing/total bins:                                0          1           -
            % Hit:                                       100.00%        100           -
            bin Bins_trans_overflow                           33          1           -   Covered
        Coverpoint count_down_cp1                        100.00%        100           -   Covered
            covered/total bins:                               16         16           -
            missing/total bins:                                0         16           -
            % Hit:                                       100.00%        100           -
            bin auto[0]                                      197          1           -   Covered
            bin auto[1]                                      213          1           -   Covered
            bin auto[2]                                      232          1           -   Covered
            bin auto[3]                                      197          1           -   Covered
            bin auto[4]                                      188          1           -   Covered
            bin auto[5]                                      183          1           -   Covered
            bin auto[6]                                      206          1           -   Covered
            bin auto[7]                                      194          1           -   Covered
            bin auto[8]                                      195          1           -   Covered
            bin auto[9]                                      191          1           -   Covered
            bin auto[10]                                     224          1           -   Covered
            bin auto[11]                                     205          1           -   Covered
            bin auto[12]                                     185          1           -   Covered
            bin auto[13]                                     219          1           -   Covered
            bin auto[14]                                     206          1           -   Covered
            bin auto[15]                                     260          1           -   Covered
        Coverpoint count_down_cp2                        100.00%        100           -   Covered
            covered/total bins:                                1          1           -
            missing/total bins:                                0          1           -
            % Hit:                                       100.00%        100           -
            bin Bins_trans_underflow                          28          1           -   Covered
    Statement Coverage:
        Enabled Coverage              Bins      Hits    Misses  Coverage
        ----------------              ----      ----    ------  --------
        Statements                       1         1         0   100.00%

```



```

TOTAL COVERGROUP COVERAGE: 100.00%  COVERGROUP TYPES: 1

ASSERTION RESULTS:
---------------------------------------------------------------------
Name                 File(Line)                    Failure      Pass
                                                   Count        Count
---------------------------------------------------------------------
/counter_tb/#anonblk#95084642#30#4#/#ublk#95084642#30/immed__32
                     counter_tb.sv(32)                   0          1

```

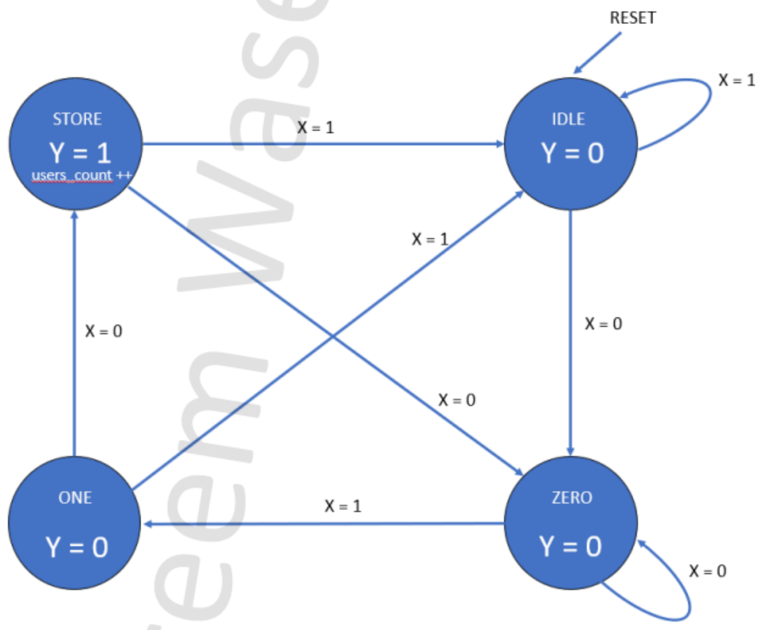
Question 2

FSM

Q2. Verify the functionality of the following Moore FSM that detects “010” non-overlapped pattern.

Ports:

Name	Type	Size	Description
x	Input	1 bit	Input sequence
clk			Clock
rst			Active high asynchronous reset
y	Output	1 bit	Output that is HIGH when the sequence 010 is detected
count		10 bits	Outputs the number of time the pattern was detected



Requirements:

1. Create a package with a typedef enum for the states named state_e
2. Create a class inside of the pacakge named fsm_transaction
    1. Variables
        - x, rst, y_exp (1 bit)
        - user_count_exp (10 bits)
    2. Constraint the reset to be deactivated most of the time
    3. Constraint the x to have value ‘0’ 67% of the time

2. In your testbench, import the package and randomize the object created from the above class
    a. After randomization, send the object to a task named check_result
    b. Inside of the check_result, you will send the object to another task named golden_model to evaluate the values of the y_exp and user_count_exp of the object. Golden_model task should have inside of it 2 variables declared as cs and ns of datatype state_e. Those will be used to model the FSM.
    c. After returning from the golden_model task, the check_result task will compare the values of the y_exp and user_counts_exp of the object with the y and user_counts ports of the DUT.
3. Generate a code coverage report and make sure that the **statements, branch, toggle and FSM coverage** are 100%

FSM RTL Design

```
//////////////////////////////////////////////////////////////////////
// Author: Kareem Waseem
// Course: Digital Verification using SV & UVM
//
// Description: 010 − sequence − detector Design
//
//////////////////////////////////////////////////////////////////////
module FSM_010(clk, rst, x, y, users_count);
  parameter IDLE   =  2′b00;
  parameter ZERO   =  2′b01;
  parameter ONE    =  2′b10;
  parameter STORE  =  2′b11;
  input clk, rst, x;
  output y;
  output reg [9: 0] users_count;
  reg [1: 0] cs, ns;
  always @(∗) begin
    case (cs)
      IDLE:
        if (x)
          ns  =  IDLE;
        else
          ns  =  ZERO;
      ZERO:
        if (x)
          ns  =  ONE;
        else
          ns  =  ZERO;
      ONE:
        if (x)
          ns  =  IDLE;
        else
          ns  =  STORE;
      STORE:
        if (x)
          ns  =  IDLE;
        else
          ns  =  ZERO;
      default:   ns  =  IDLE;
    endcase
  end
```

```
    always @(posedge clk or posedge rst) begin
        if(rst) begin
            cs <= IDLE;
        end
        else begin
            cs <= ns;
        end
    end

    always @(posedge clk or posedge rst) begin
        if(rst) begin
            users_count <= 0;
        end
        else begin
            if (cs == STORE)
                users_count <= users_count + 1;
        end
    end

    assign y = (cs == STORE)? 1: 0;

endmodule
```

Golden Model

```
module FSM_golden_model(x, clk, rst, y_exp, count_exp);

parameter STORE = 2'b00;
parameter IDLE = 2'b01;
parameter ZERO = 2'b10;
parameter ONE = 2'b11;

input x, clk, rst;
output reg y_exp;
output reg [9:0] count_exp;

reg [1:0] cs, ns;

// Next state Logic
always @(cs or x) begin
   case (cs)
      STORE :
         if (x)
            ns = IDLE;
         else
            ns = ZERO;
      IDLE :
         if (x)
            ns = IDLE;
         else
            ns = ZERO;
      ZERO :
         if (x)
            ns = ONE;
         else
            ns = ZERO;
      ONE :
         if (x)
            ns = IDLE;
         else
            ns = STORE;
      default : ns = IDLE;
   endcase
end
```

```
// State Memory
always @(posedge clk or posedge rst) begin
   if (rst) begin
      cs  <=  IDLE;
      count_exp  <=  0;
   end
   else
      cs  <=  ns;
end

// Output Logic
always @(cs) begin
   case (cs)
      STORE :  begin
               y_exp  =  1;
               count_exp  =  count_exp  +  1;
            end
      IDLE, ZERO, ONE :  y_exp  =  0;
   endcase
end

endmodule
```

Package

```
package my_pkg ;

typedef enum logic [1: 0] {IDLE , ZERO , ONE , STORE} state_e;

class fsm_transaction ;

rand bit rst_c , x_c ;
bit y_exp ;
bit [9: 0] user_count_exp ;

constraint C_inputs { rst_c dist {0: = 90 , 1: = 10};
                  x_c dist {0:/67 , 1:/33};    }

endclass
endpackage
```

FSM Testbench

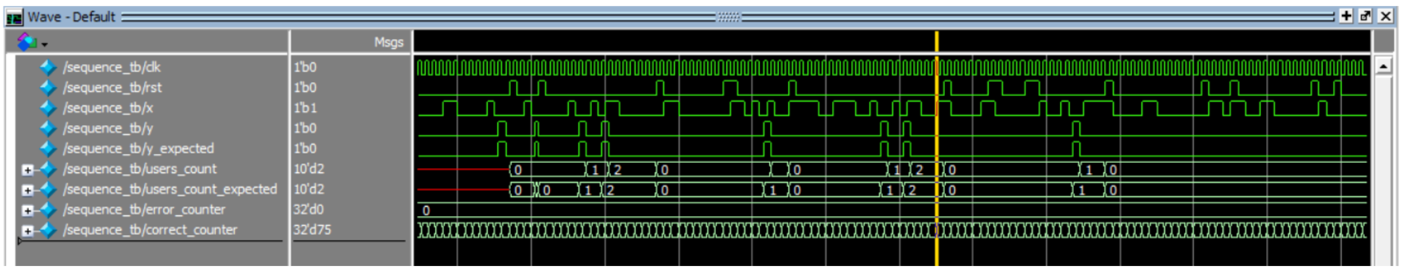
```
import my_pkg ::*;

module sequence_tb;

parameter IDLE   =  2'b00;
parameter ZERO   =  2'b01;
parameter ONE    =  2'b10;
parameter STORE  =  2'b11;

logic clk, rst, x;
logic y, y_expected;
logic [9: 0] users_count, users_count_expected;

fsm_transaction my_fsm;      // create object from the class

FSM_010 #(. IDLE(IDLE), . ZERO(ZERO), . ONE(ONE), . STORE(STORE))
                                        DUT_Design (clk, rst, x, y, users_count);
FSM_golden_model #(. IDLE(IDLE), . ZERO(ZERO), . ONE(ONE), . STORE(STORE))
                          DUT_GOLD (x, clk, rst, y_expected, users_count_expected);
int error_counter, correct_counter;

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

initial begin
  error_counter  =  0;
  correct_counter  =  0;
  my_fsm  =  new();
  for(int i  =  0 ; i  <  10000 ; i + +) begin
    assert(my_fsm. randomize());
    rst  =  my_fsm. rst_c;
    x  =  my_fsm. x_c;
    my_fsm. y_exp  =  y_expected;
    my_fsm. user_count_exp  =  users_count_expected;
    check_result();
  end
  // Directed test to get the toggle cover to 100
  rst = 1;
  #10;
  rst = 0;
  for (int i = 0 ; i < 10000 ; i + +) begin
    x = 0;
    #2;
    x = 1;
    #2;
    x = 0;
    #2;
  end
  $display("Error_Counter = %0d , Correct_Counter = %0d",
                                                  error_counter, correct_counter);
  $stop;
end
// Check Result Task
task check_result();
  @(negedge clk);
  if ((y ! == y_expected) && (users_count ! == users_count_expected)) begin
    $display("The FSM Design is Wrong! ");
    error_counter + +;
  end
  else
    correct_counter + +;
endtask

endmodule
```

Do File

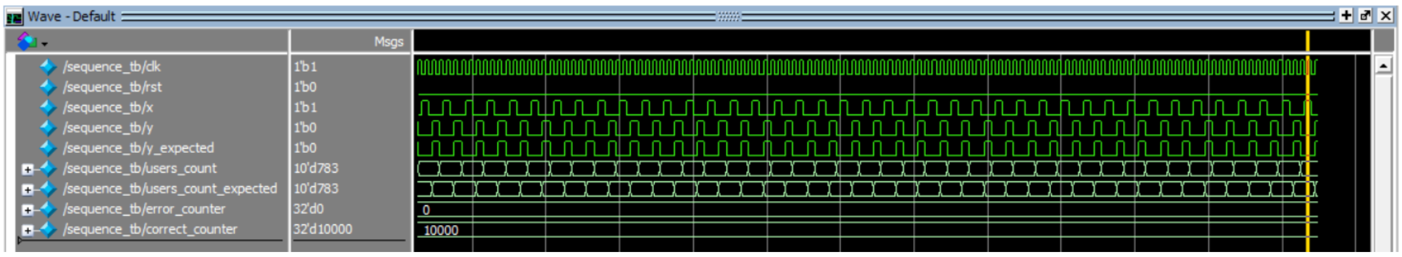
Do File for (FSM_010) Question 2

```
vlib work
vlog FSM_010. v FSM_package. sv FSM_golden_model. sv FSM_testbench. sv + cover − covercells
vsim − voptargs = +acc work. sequence_tb − cover
add wave *
coverage save FSM_testbench. ucdb − onexit − du work. FSM_010
run − all
# quit − sim
# vcover report FSM_testbench. ucdb − details − annotate − all − output FSM_Coverage_Rpt. txt
```

Waveform



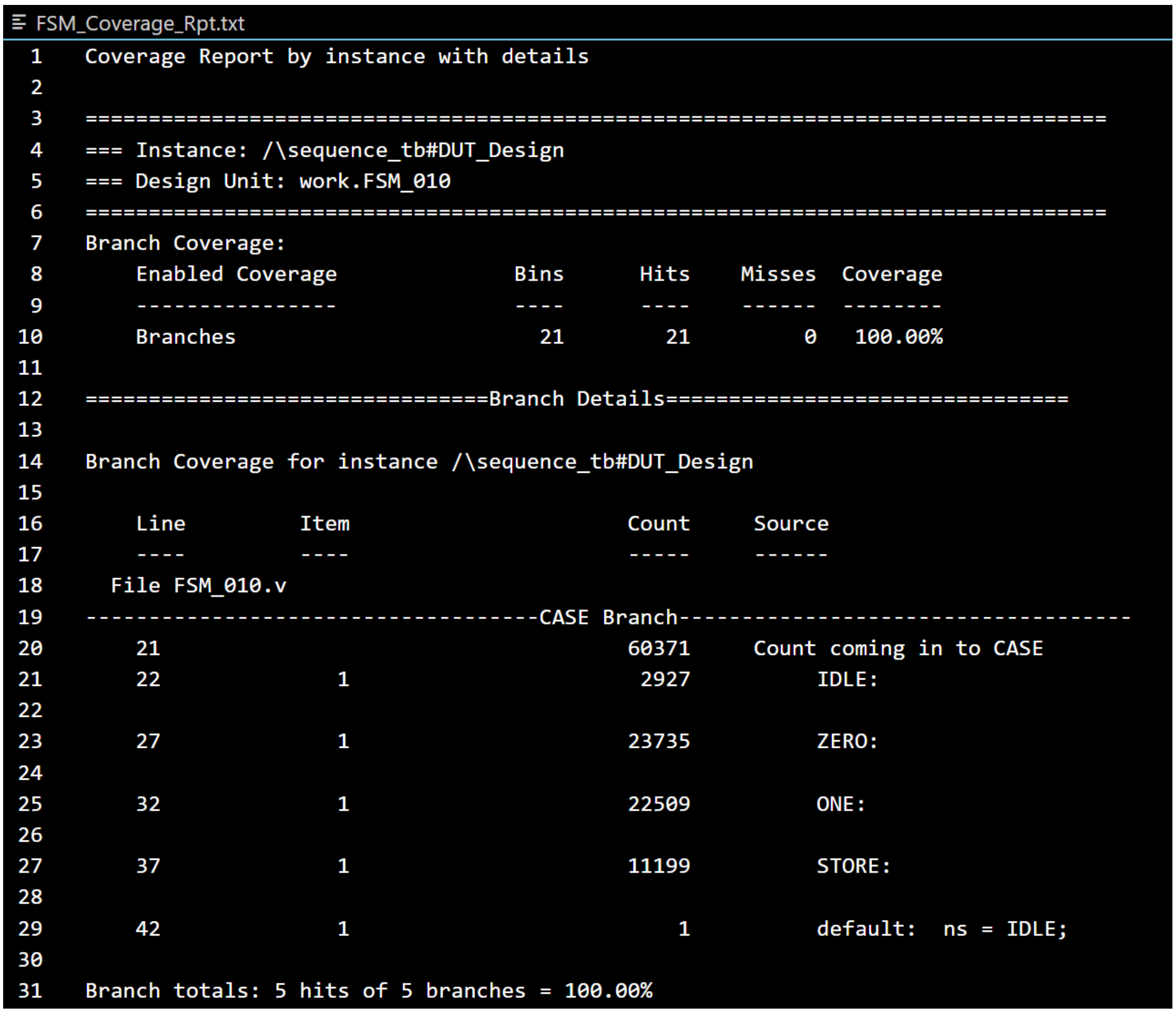
```
Wave - Default
                                   Msgs
/sequence_tb/clk                   1'b0
/sequence_tb/rst                   1'b0
/sequence_tb/x                     1'b1
/sequence_tb/y                     1'b0
/sequence_tb/y_expected            1'b0
/sequence_tb/users_count           10'd2
/sequence_tb/users_count_expected  10'd2
/sequence_tb/error_counter         32'd0
/sequence_tb/correct_counter       32'd75
```



```
Wave - Default
                                   Msgs
/sequence_tb/clk                   1'b1
/sequence_tb/rst                   1'b0
/sequence_tb/x                     1'b1
/sequence_tb/y                     1'b0
/sequence_tb/y_expected            1'b0
/sequence_tb/users_count           10'd783
/sequence_tb/users_count_expected  10'd783
/sequence_tb/error_counter         32'd0
/sequence_tb/correct_counter       32'd10000
```

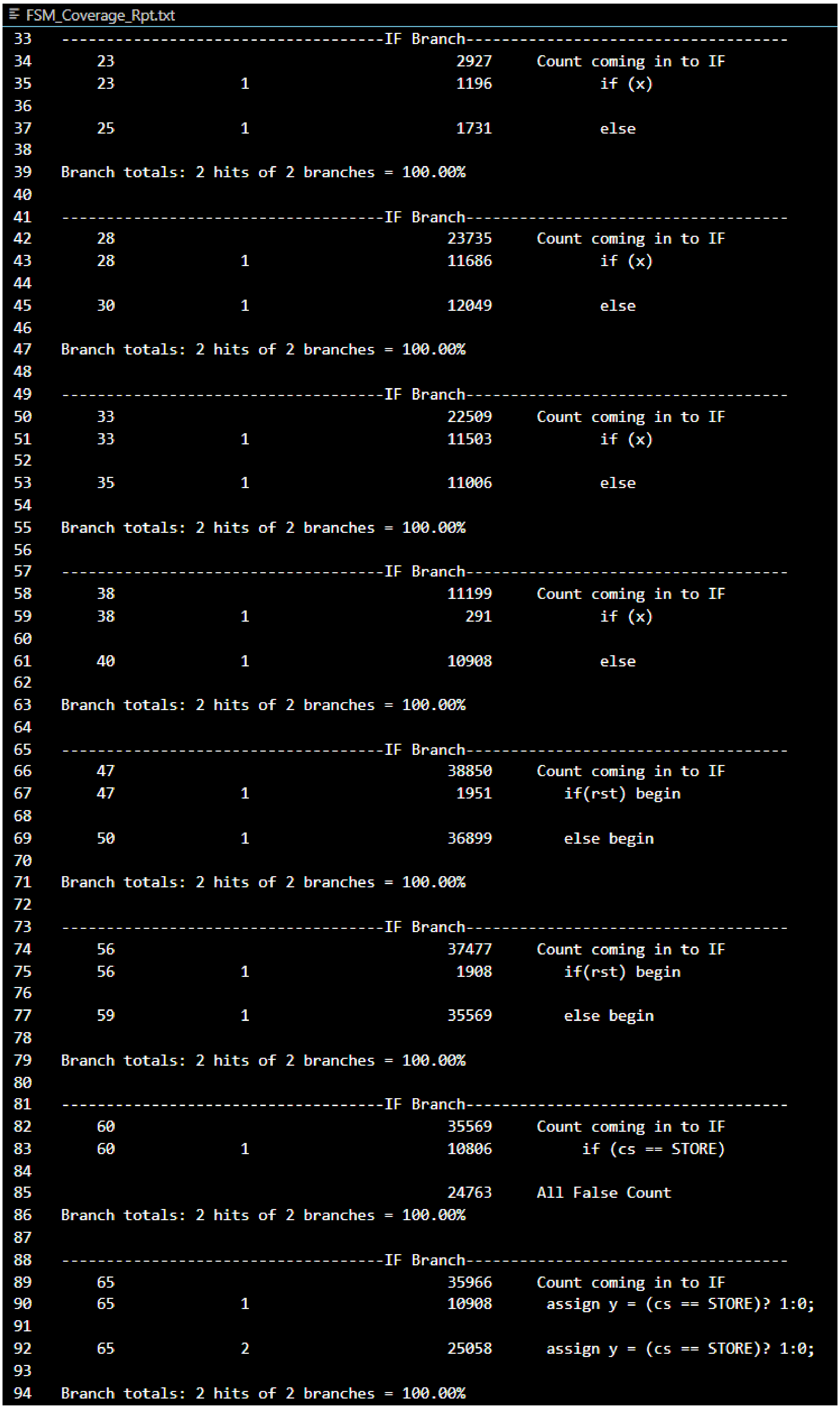
```
# Error_Counter = 0 , Correct_Counter = 10000
```

Code Coverage

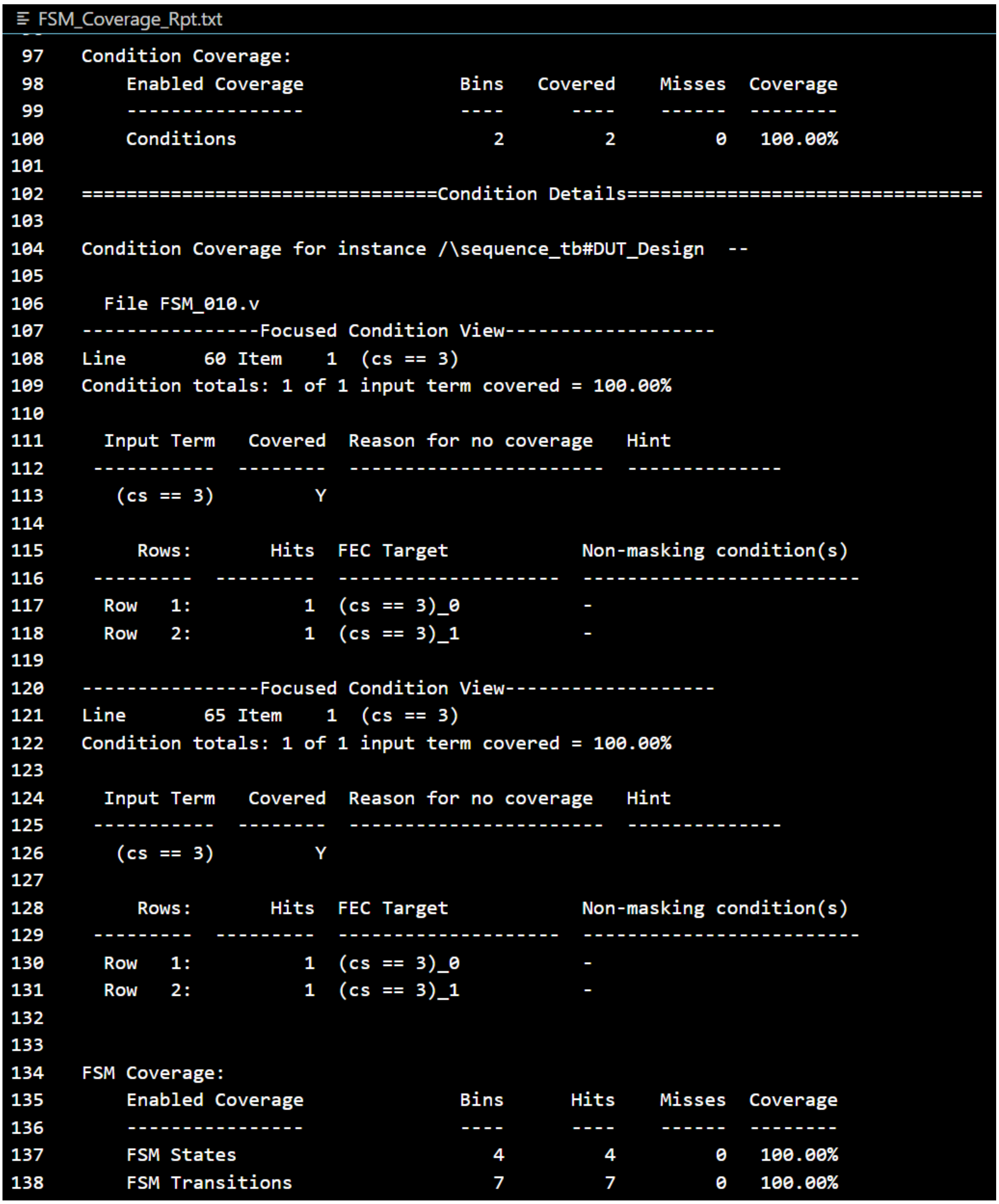


FSM_Coverage_Rpt.txt

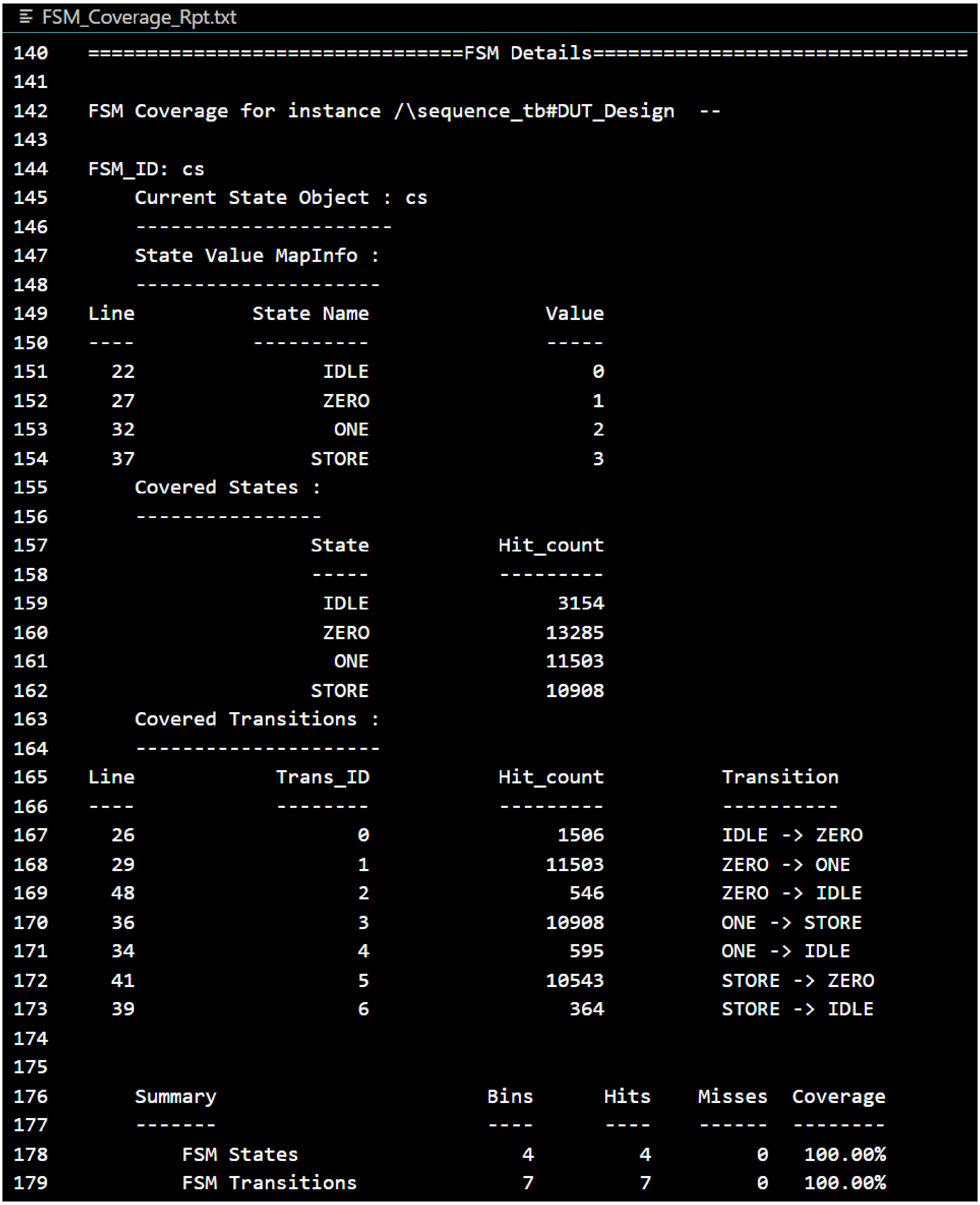
```
Coverage Report by instance with details

=================================================================================
=== Instance: /\sequence_tb#DUT_Design
=== Design Unit: work.FSM_010
=================================================================================
Branch Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Branches                        21        21         0   100.00%

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

Branch Coverage for instance /\sequence_tb#DUT_Design

    Line         Item                      Count     Source
    ----         ----                      -----     ------
  File FSM_010.v
------------------------------------CASE Branch------------------------------------
    21                                     60371     Count coming in to CASE
    22            1                         2927          IDLE:

    27            1                        23735          ZERO:

    32            1                        22509          ONE:

    37            1                        11199          STORE:

    42            1                            1          default:  ns = IDLE;

Branch totals: 5 hits of 5 branches = 100.00%
```


FSM_Coverage_Rpt.txt

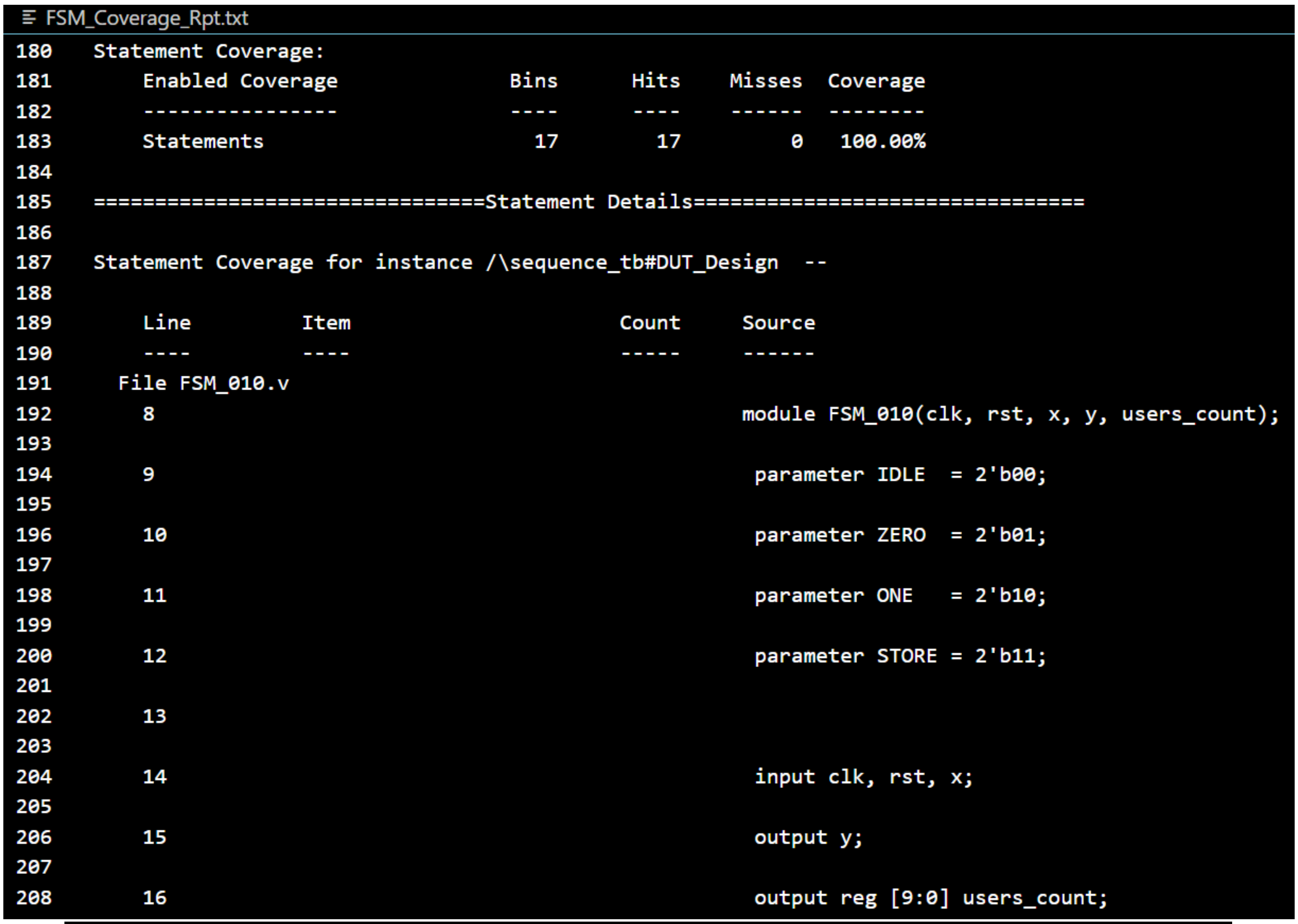
```
-----------------------------------IF Branch----------------------------------
    23                                  2927    Count coming in to IF
    23               1                  1196           if (x)

    25               1                  1731           else

Branch totals: 2 hits of 2 branches = 100.00%

-----------------------------------IF Branch----------------------------------
    28                                 23735    Count coming in to IF
    28               1                 11686           if (x)

    30               1                 12049           else

Branch totals: 2 hits of 2 branches = 100.00%

-----------------------------------IF Branch----------------------------------
    33                                 22509    Count coming in to IF
    33               1                 11503           if (x)

    35               1                 11006           else

Branch totals: 2 hits of 2 branches = 100.00%

-----------------------------------IF Branch----------------------------------
    38                                 11199    Count coming in to IF
    38               1                   291           if (x)

    40               1                 10908           else

Branch totals: 2 hits of 2 branches = 100.00%

-----------------------------------IF Branch----------------------------------
    47                                 38850    Count coming in to IF
    47               1                  1951       if(rst) begin

    50               1                 36899       else begin

Branch totals: 2 hits of 2 branches = 100.00%

-----------------------------------IF Branch----------------------------------
    56                                 37477    Count coming in to IF
    56               1                  1908       if(rst) begin

    59               1                 35569       else begin

Branch totals: 2 hits of 2 branches = 100.00%

-----------------------------------IF Branch----------------------------------
    60                                 35569    Count coming in to IF
    60               1                 10806          if (cs == STORE)

                                       24763    All False Count
Branch totals: 2 hits of 2 branches = 100.00%

-----------------------------------IF Branch----------------------------------
    65                                 35966    Count coming in to IF
    65               1                 10908     assign y = (cs == STORE)? 1:0;

    65               2                 25058     assign y = (cs == STORE)? 1:0;

Branch totals: 2 hits of 2 branches = 100.00%
```



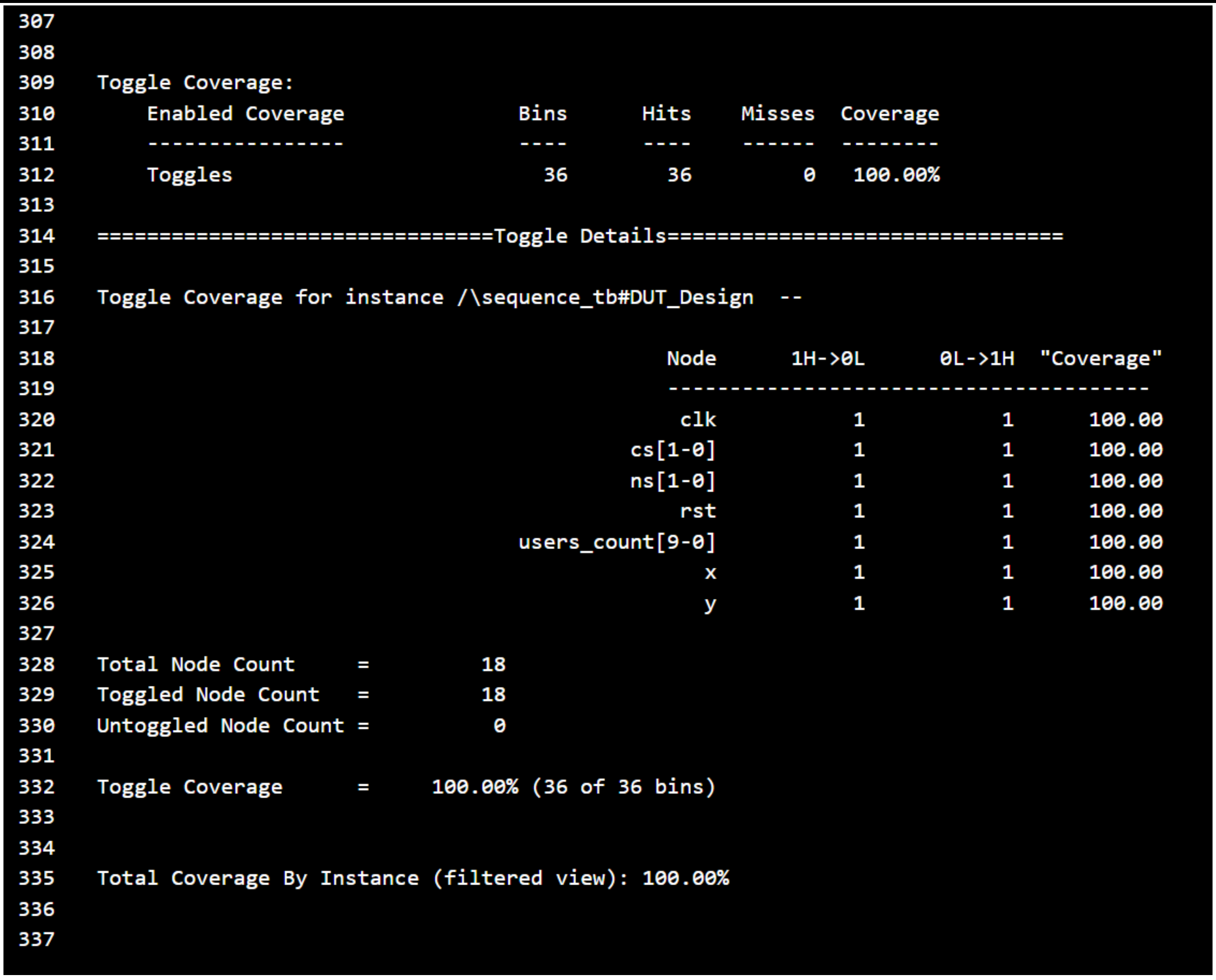
```
Condition Coverage:
    Enabled Coverage              Bins    Covered    Misses  Coverage
    ----------------              ----       ----    ------  --------
    Conditions                       2          2         0   100.00%

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

Condition Coverage for instance /\sequence_tb#DUT_Design  --

  File FSM_010.v
----------------Focused Condition View-------------------
Line       60 Item    1  (cs == 3)
Condition totals: 1 of 1 input term covered = 100.00%

  Input Term   Covered  Reason for no coverage   Hint
 -----------  --------  -----------------------  --------------
   (cs == 3)         Y

     Rows:       Hits  FEC Target            Non-masking condition(s)
 ---------  ---------  --------------------  -------------------------
  Row   1:          1  (cs == 3)_0           -
  Row   2:          1  (cs == 3)_1           -

----------------Focused Condition View-------------------
Line       65 Item    1  (cs == 3)
Condition totals: 1 of 1 input term covered = 100.00%

  Input Term   Covered  Reason for no coverage   Hint
 -----------  --------  -----------------------  --------------
   (cs == 3)         Y

     Rows:       Hits  FEC Target            Non-masking condition(s)
 ---------  ---------  --------------------  -------------------------
  Row   1:          1  (cs == 3)_0           -
  Row   2:          1  (cs == 3)_1           -


FSM Coverage:
    Enabled Coverage              Bins       Hits    Misses  Coverage
    ----------------              ----       ----    ------  --------
    FSM States                       4          4         0   100.00%
    FSM Transitions                  7          7         0   100.00%
```


FSM_Coverage_Rpt.txt

```
================================FSM Details================================

FSM Coverage for instance /\sequence_tb#DUT_Design  --

FSM_ID: cs
    Current State Object : cs
    ----------------------
    State Value MapInfo :
    ---------------------
Line          State Name               Value
----          ----------               -----
  22                IDLE                   0
  27                ZERO                   1
  32                 ONE                   2
  37               STORE                   3
    Covered States :
    ----------------
                   State           Hit_count
                   -----           ---------
                    IDLE                3154
                    ZERO               13285
                     ONE               11503
                   STORE               10908
    Covered Transitions :
    ---------------------
Line            Trans_ID           Hit_count          Transition
----            --------           ---------          ----------
  26                   0                1506          IDLE -> ZERO
  29                   1               11503          ZERO -> ONE
  48                   2                 546          ZERO -> IDLE
  36                   3               10908          ONE -> STORE
  34                   4                 595          ONE -> IDLE
  41                   5               10543          STORE -> ZERO
  39                   6                 364          STORE -> IDLE


    Summary                       Bins      Hits    Misses  Coverage
    -------                       ----      ----    ------  --------
        FSM States                   4         4         0   100.00%
        FSM Transitions              7         7         0   100.00%
```



```
Statement Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Statements                      17        17         0   100.00%

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

Statement Coverage for instance /\sequence_tb#DUT_Design  --

    Line         Item                      Count     Source
    ----         ----                      -----     ------
  File FSM_010.v
    8                                                module FSM_010(clk, rst, x, y, users_count);

    9                                                 parameter IDLE  = 2'b00;

    10                                                parameter ZERO  = 2'b01;

    11                                                parameter ONE   = 2'b10;

    12                                                parameter STORE = 2'b11;

    13

    14                                                input clk, rst, x;

    15                                                output y;

    16                                                output reg [9:0] users_count;
```



```


Toggle Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Toggles                         36        36         0   100.00%

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

Toggle Coverage for instance /\sequence_tb#DUT_Design  --

                                              Node      1H->0L      0L->1H  "Coverage"
                                              ---------------------------------------
                                               clk           1           1      100.00
                                           cs[1-0]           1           1      100.00
                                           ns[1-0]           1           1      100.00
                                               rst           1           1      100.00
                                  users_count[9-0]           1           1      100.00
                                                 x           1           1      100.00
                                                 y           1           1      100.00

Total Node Count     =         18
Toggled Node Count   =         18
Untoggled Node Count =          0

Toggle Coverage      =     100.00% (36 of 36 bins)


Total Coverage By Instance (filtered view): 100.00%


```