P1)

1. Design code

```
module queue_operations () ;
    int j  ;
    int q [$] ;
    initial begin
        j = 1 ;
        q = {0,2,5} ;
        q.insert (1 , j) ;
        $display("Queue elements afer inserting j = %p", q);
        q.delete (1) ;
        $display("Queue elements afer deleting= %p", q);
        q.push_front(7) ;
        $display("Queue elements afer pushing front 7= %p", q);
        q.push_back(9) ;
        $display("Queue elements afer pushing back 9 = %p", q);
        j = q.pop_back() ;
        $display ("the j variable has the value %d" , j) ;
        $display("Queue contents after pop back = %p", q);
        j = q.pop_front() ;
        $display ("the j variable has the value %d" , j) ;
        $display("Queue contents after pop front = %p", q);
        q.reverse() ;
        $display("reversed Queue contents = %p", q);
        q.sort() ;
        $display("sorted Queue contents = %p", q);
        q.rsort() ;
        $display("reversed sorted Queue contents = %p", q);
        q.shuffle() ;
        $display("shuffled Queue contents = %p", q);
    end
endmodule
```

2. Transcript output

```
# Queue elements afer inserting j = '{0, 1, 2, 5}
# Queue elements afer deleting= '{0, 2, 5}
# Queue elements afer pushing front 7= '{7, 0, 2, 5}
# Queue elements afer pushing back 9 = '{7, 0, 2, 5, 9}
# the j variable has the value           9
# Queue contents after pop back = '{7, 0, 2, 5}
# the j variable has the value           7
# Queue contents after pop front = '{0, 2, 5}
# reversed Queue contents = '{5, 2, 0}
# sorted Queue contents = '{0, 2, 5}
# reversed sorted Queue contents = '{5, 2, 0}
# shuffled Queue contents = '{2, 0, 5}
```

P2)

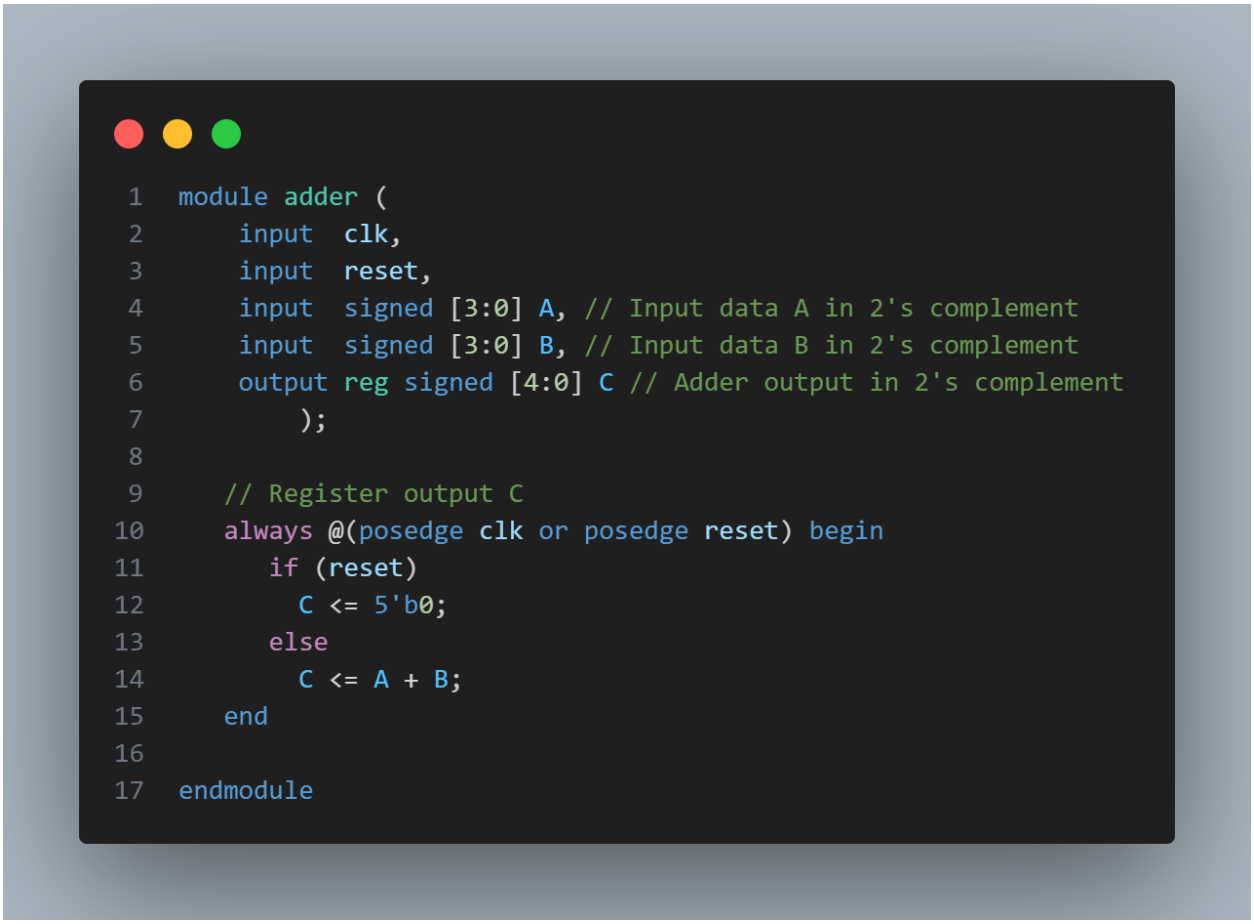
1. Testbench code

```
import adder_package::*;
module adder_tb();
    reg  clk;
    reg  reset;
    reg  signed [3:0] A;
    reg  signed [3:0] B;
    wire signed [4:0] C;
    reg  signed [4:0] expected_out;
    adder_class obj = new ;

    integer error_count , correct_count ;

    localparam MAXPOS = 7 ;
    localparam MAXNEG = -8 ;

    adder DUT (clk,reset,A,B,C) ;

    initial begin
        clk = 0;
        forever begin
            #2 clk = ~clk;
            obj.clk_cv = clk ;
        end
    end

    initial begin
        A = 0 ;
        B = 4 ;
        reset = 0 ;
        error_count = 0 ;
        correct_count = 0 ;
        // TEST_1
        assert_reset () ;
        // TEST_2
        repeat(1000) begin
            assert(obj.randomize()) ;
            A = obj.A_cv ;
            B = obj.B_cv ;
            reset = obj.reset_cv ;
            obj.sampling() ;
            golden_model () ;
            check_result () ;
        end
        // TEST_18
        assert_reset () ;

        $display ("correct counter = %d , error counter = %d " , correct_count , error_count ) ;
        $stop ;
    end

    task golden_model();
        if (reset) expected_out = 0 ;
        else expected_out = A + B ;
    endtask

    task check_result ();
        @ (negedge clk) ;
        if (C == expected_out) begin
            $display ("test passes") ;
            correct_count = correct_count + 1 ;
        end
        else begin
           $display ("test fail ") ;
            error_count = error_count + 1 ;
        end
    endtask

    task assert_reset ();
        reset = 1 ;
        expected_out = 0 ;
        check_result () ;
        reset = 0 ;
    endtask

endmodule
```

2. Package code

```
package adder_package;
    typedef enum int signed { MAXPOS = 7 , ZERO = 0  , MAXNEG = -8 } values_t;
    class adder_class;
        rand logic signed  [3:0] A_cv ;
        rand logic signed  [3:0] B_cv ;
        rand values_t values;
        bit reset_cv ;
        bit clk_cv ;


        constraint reset_constraint {
            reset_cv dist {0:/97 , 1:/3} ;
        }

        constraint A_values {
            A_cv dist { MAXPOS:/20, ZERO:/20, MAXNEG:/20,
                        [-7:-1]:/20, [1:6]:/20 };
        }

        constraint B_values {
            B_cv dist { MAXPOS:/20, ZERO:/20, MAXNEG:/20,
                        [-7:-1]:/20, [1:6]:/20 };
        }

        covergroup covgrp_A @(posedge clk_cv) ;
            cvp1: coverpoint A_cv {
                bins data_0       = {ZERO} ;
                bins data_max     = {MAXPOS};
                bins data_min     = {MAXNEG};
                bins data_default = default ;
            }
            cvp2: coverpoint A_cv {
                bins data_0max   = (ZERO=>MAXPOS) ;
                bins data_maxmin = (MAXPOS=>MAXNEG);
                bins data_minmax = (MAXNEG=>MAXPOS);
            }
        endgroup

        covergroup covgrp_B @(posedge clk_cv) ;
            cvp1: coverpoint B_cv {
                bins data_0       = {ZERO} ;
                bins data_max     = {MAXPOS};
                bins data_min     = {MAXNEG};
                bins data_default = default ;
            }
            cvp2: coverpoint B_cv {
                bins data_0max   = (ZERO=>MAXPOS) ;
                bins data_maxmin = (MAXPOS=>MAXNEG);
                bins data_minmax = (MAXNEG=>MAXPOS);
            }
        endgroup

        function new();
            covgrp_A = new() ;
            covgrp_B = new() ;
        endfunction

        task sampling ();
            if (!reset_cv) begin
                covgrp_A.sample() ;
                covgrp_B.sample() ;
            end
        endtask

    endclass
endpackage
```

3. Design code



```
module adder (
    input  clk,
    input  reset,
    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 // Adder output in 2's complement
        );

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

endmodule
```

4. Snippet to your verification plan document

Label	Design Requirement Description	Stimulus Generation	Functional Coverage	Functionality Check
TEST_1	When the reset is asserted by task **assert_reset**, the output value should be zero	Directed at the start of the simulation, then randomized with constraint that drive the reset to be most of the simulation off		A checker in the testbench to make sure the output is correct
TEST_2	when the **reset** is deasserted , the output should be the summation of **operand A** and **operand B**	Randomized under constraint on operand A & operand B to take value (-8) 20% of simulation , take value (0) 20% of simulation , take value (7) 20% of simulation and take the rest of values 40% of simulation	cover all values of **operand A** and **operand B** input.	A checker in the testbench to make sure the output is correct
TEST_3	When the reset is asserted, the output value should be low	Directed at the end of the simulation		A checker in the testbench to make sure the output is correct

5. Do file

```
vlib work
vlog adder.v adder_tb.sv package.svh  +cover -covercells
vsim -voptargs=+acc work.adder_tb -cover
add wave *
coverage save adder.ucdb -onexit -du adder
run -all
```

6. Code Coverage report snippets

```
----------------------------------------------------------------------------

Covergroup Coverage:
    Covergroups                 2        na        na   100.00%
        Coverpoints/Crosses     4        na        na        na
            Covergroup Bins    12        12         0   100.00%
----------------------------------------------------------------------------------------------------------
Covergroup                                          Metric       Goal       Bins    Status

----------------------------------------------------------------------------------------------------------
 TYPE /adder_package/adder_class/covgrp_A          100.00%        100          -    Covered
    covered/total bins:                                  6          6          -
    missing/total bins:                                  0          6          -
    % Hit:                                         100.00%        100          -
    Coverpoint cvp1                                100.00%        100          -    Covered
        covered/total bins:                              3          3          -
        missing/total bins:                              0          3          -
        % Hit:                                     100.00%        100          -
    Coverpoint cvp2                                100.00%        100          -    Covered
        covered/total bins:                              3          3          -
        missing/total bins:                              0          3          -
        % Hit:                                     100.00%        100          -
 Covergroup instance \/adder_package::adder_class::covgrp_A
                                                   100.00%        100          -    Covered
    covered/total bins:                                  6          6          -
    missing/total bins:                                  0          6          -
    % Hit:                                         100.00%        100          -
    Coverpoint cvp1                                100.00%        100          -    Covered
        covered/total bins:                              3          3          -
        missing/total bins:                              0          3          -
        % Hit:                                     100.00%        100          -
        bin data_0                                     206          1          -    Covered
        bin data_max                                   201          1          -    Covered
        bin data_min                                   187          1          -    Covered
        default bin data_default                       408                     -    Occurred
    Coverpoint cvp2                                100.00%        100          -    Covered
        covered/total bins:                              3          3          -
        missing/total bins:                              0          3          -
        % Hit:                                     100.00%        100          -
        bin data_0max                                   39          1          -    Covered
        bin data_maxmin                                 42          1          -    Covered
        bin data_minmax                                 40          1          -    Covered
```

```
TYPE /adder_package/adder_class/covgrp_B            100.00%        100          -    Covered
   covered/total bins:                                    6          6          -
   missing/total bins:                                    0          6          -
   % Hit:                                           100.00%        100          -
   Coverpoint cvp1                                  100.00%        100          -    Covered
       covered/total bins:                                3          3          -
       missing/total bins:                                0          3          -
       % Hit:                                       100.00%        100          -
   Coverpoint cvp2                                  100.00%        100          -    Covered
       covered/total bins:                                3          3          -
       missing/total bins:                                0          3          -
       % Hit:                                       100.00%        100          -
Covergroup instance \/adder_package::adder_class::covgrp_B
                                                    100.00%        100          -    Covered
   covered/total bins:                                    6          6          -
   missing/total bins:                                    0          6          -
   % Hit:                                           100.00%        100          -
   Coverpoint cvp1                                  100.00%        100          -    Covered
       covered/total bins:                                3          3          -
       missing/total bins:                                0          3          -
       % Hit:                                       100.00%        100          -
       bin data_0                                       211          1          -    Covered
       bin data_max                                     193          1          -    Covered
       bin data_min                                     179          1          -    Covered
       default bin data_default                         419                     -    Occurred
   Coverpoint cvp2                                  100.00%        100          -    Covered
       covered/total bins:                                3          3          -
       missing/total bins:                                0          3          -
       % Hit:                                       100.00%        100          -
       bin data_0max                                     48          1          -    Covered
       bin data_maxmin                                   33          1          -    Covered
       bin data_minmax                                   39          1          -    Covered
```

```
TOTAL COVERGROUP COVERAGE: 100.00%  COVERGROUP TYPES: 2

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

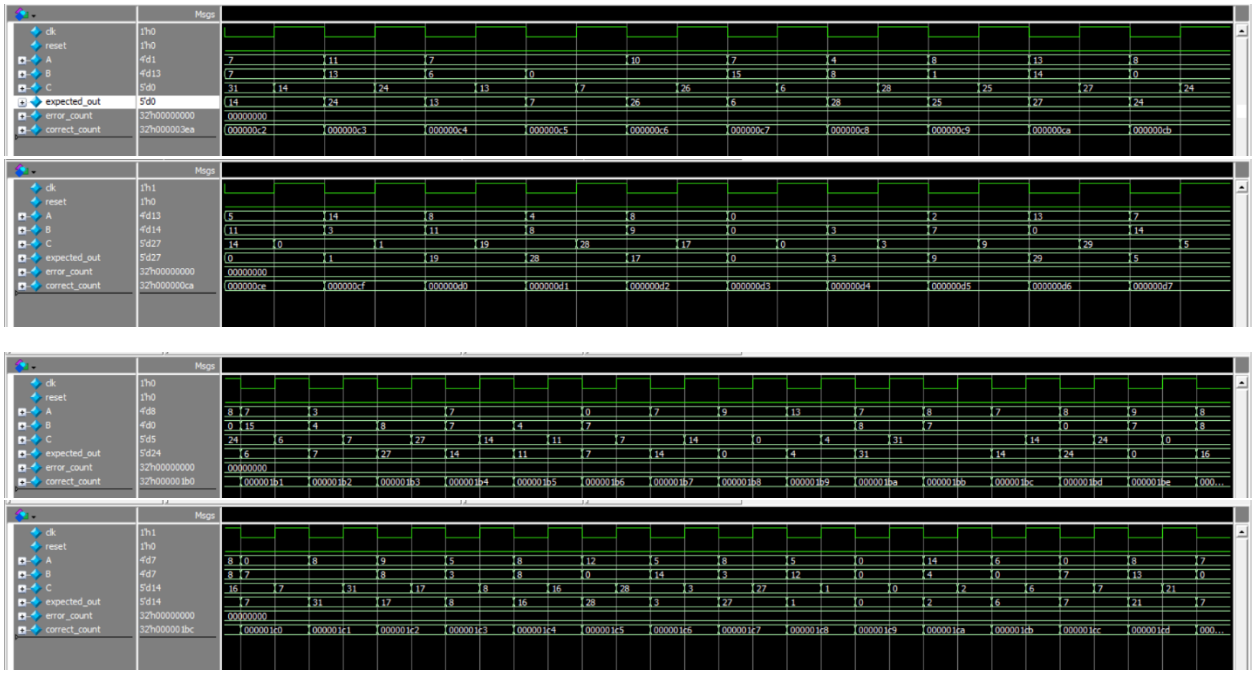
7. Functional Coverage report snippets

```
# ================================================================================
# Branch Coverage:
#     Enabled Coverage              Bins      Hits    Misses  Coverage
#     ----------------              ----      ----    ------  --------
#     Branches                         2         2         0   100.00%
#
# ================================Branch Details================================
#
# Branch Coverage for instance /\adder_tb#DUT
#
#     Line         Item                      Count     Source
#     ----         ----                      -----     ------
#   File adder.v
# ------------------------------------IF Branch------------------------------------
#     11                                      1003     Count coming in to IF
#     11              1                          4          if (reset)
#
#     13              1                        999          else
#
# Branch totals: 2 hits of 2 branches = 100.00%
#
#
# Statement Coverage:
#     Enabled Coverage              Bins      Hits    Misses  Coverage
#     ----------------              ----      ----    ------  --------
#     Statements                       3         3         0   100.00%
#
```

```
#
# Statement Coverage:
#     Enabled Coverage              Bins      Hits    Misses  Coverage
#     ----------------              ----      ----    ------  --------
#     Statements                       3         3         0   100.00%
#
# ================================Statement Details================================
#
# Statement Coverage for instance /\adder_tb#DUT  --
#
#     Line         Item                     Count     Source
#     ----         ----                     -----     ------
#   File adder.v
#     1                                               module adder (
#
#     2                                                   input  clk,
#
#     3                                                   input  reset,
#
#     4                                                   input  signed [3:0] A, // Input data A in 2's complement
#
#     5                                                   input  signed [3:0] B, // Input data B in 2's complement
#
#     6                                                   output reg signed [4:0] C // Adder output in 2's complement
#
#     7                                                   );
#
#     8
#
#     9                                                  // Register output C
#
#     10             1                      1003        always @(posedge clk or posedge reset) begin
#
#     11                                                   if (reset)
#
#     12             1                         4          C <= 5'b0;
#
#     13                                                   else
#
#     14             1                       999          C <= A + B;
#
#
```

```
#
# Toggle Coverage:
#     Enabled Coverage               Bins      Hits    Misses  Coverage
#     ----------------               ----      ----    ------  --------
#     Toggles                          30        30         0   100.00%
#
# ================================Toggle Details================================
#
# Toggle Coverage for instance /\adder_tb#DUT  --
#
#                                              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%
#
```

8. Clear and neat QuestaSim waveform snippets showing the functionality of the design




```
test passes
test passes
test passes
test passes
correct counter =        1002 , error counter =           0
** Note: $stop    : adder_tb.sv(48)
```

P3)

1. Testbench code

```
import FSM_package::*;
module FSM_tb();
    reg X_tb ;
    reg clk_tb ;
    reg rst_tb ;
    wire Y_tb ;
    wire [9:0] count_tb ;

    reg Y_expected ;
    reg [9:0] count_expected ;
    state_e cs_tb , ns_tb ;
    reg Y_passed , count_passed ;
            fsm_transaction obj = new ;

    FSM_010 DUT (clk_tb, rst_tb, X_tb, Y_tb, count_tb) ;

    integer error_count , correct_count ;

    initial begin
        clk_tb = 0;
        forever begin
            #2 clk_tb = ~clk_tb;
            obj.clk_cv= clk_tb ;
        end
    end

    initial begin
        rst_tb = 0 ;
        correct_count = 0 ;
        error_count = 0 ;

        assert_reset() ;

        repeat (1000) begin
            assert( obj.randomize() ) else $fatal("Randomization failed");
            X_tb = obj.X ;
            rst_tb = obj.rst ;
            check_result() ;
        end

        assert_reset() ;

        $display("correct counter = %d , error counter = %d " , correct_count , error_count ) ;
        $stop ;
    end

    task assert_reset () ;
        rst_tb = 1 ;
        Y_expected = 0 ;
        count_expected = 0 ;
        ns_tb = IDLE ;
        cs_tb = IDLE ;
        @(negedge clk_tb);
        if (Y_expected == Y_tb && count_expected == count_tb) begin
            $display("Test passes") ;
            correct_count = correct_count + 1 ;
        end
        else begin
            $display("Test fails") ;
            error_count = error_count + 1 ;
        end
        rst_tb = 0 ;
    endtask

    task golden_model () ;
        if (rst_tb) begin
            Y_expected = 0 ;
            count_expected = 0 ;
            ns_tb = IDLE ;
        end
        else begin
            case (cs_tb)
                IDLE: begin
                    if (X_tb) ns_tb = IDLE ;
                    else ns_tb = ZERO ;
                end
                ZERO: begin
                    if (X_tb) ns_tb = ONE ;
                    else ns_tb = ZERO ;
                end
                ONE: begin
                    if (X_tb) ns_tb = IDLE ;
                    else ns_tb = STORE ;
                end
                STORE: begin
                    count_expected = count_expected + 1 ;
                    if (X_tb) ns_tb = IDLE ;
                    else ns_tb = ZERO ;
                end
            endcase
        end
    endtask

    task check_result();
        if (cs_tb == STORE) Y_expected = 1 ;
        else Y_expected = 0 ;
        if (Y_expected == Y_tb) Y_passed = 1 ;
        else Y_passed = 0 ;
        golden_model() ;
        @(posedge clk_tb);
        cs_tb = ns_tb ;
        @(negedge clk_tb);
        if (count_expected == count_tb) count_passed = 1 ;
        else count_passed = 0 ;
        if (Y_passed && count_passed) begin
            $display("Test passes") ;
            correct_count = correct_count + 1 ;
        end
        else begin
            $display("Test fails") ;
            error_count = error_count + 1 ;
        end
    endtask
endmodule
```

2. Package code

```
package FSM_package;
    typedef enum logic [1:0] { IDLE , ZERO , ONE , STORE } state_e ;
    class fsm_transaction;
        rand logic X ;
        rand logic rst ;
        // state_e FSM_state ;
        bit clk_cv ;

        constraint RESET {
            rst dist {0:/97 , 1:/3} ;
        }

        constraint X_rand {
            X dist {0:/67 , 1:/33} ;
        }

        covergroup covgrp @(posedge clk_cv) ;
            cv1: coverpoint X {
                bins seq_transition = (0 => 1 => 0) ;
            }
        endgroup

        function new();
            covgrp = new() ;
        endfunction

    endclass
endpackage
```

3. Design code

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

4. Snippet to your verification plan document

A	B	C	D	E
Label	**Design Requirement Description**	**Stimulus Generation**	**Functional Coverage**	**Functionality Check**
TEST_1	When the **reset** is asserted, the output **Y** & **user_count** should be 0	Directed at the start of the simulation then randomized to be off most of the simulation		A checker in the testbench to make sure the output is correct
TEST_2	When the input signal **X** has a pattern **010** the output **Y** should be **0** & **user_count incremented**	Randomized inout **X** under constraint to be low 67% of simulation & to be high 33% of simulation	check all the values of inpuut **X**	A checker in the testbench to make sure the output is correct
TEST_3	When the **reset** is asserted, the output value should be low	Directed at the end of the simulation		A checker in the testbench to make sure the output is correct

5. Do file

```
1  vlib work
2  vlog FSm_010.v TESTBENCH.svh package.svh  +cover -covercells
3  vsim -voptargs=+acc work.FSM_tb -cover
4  add wave *
5  coverage exclude -src FSm_010.v -line 42 -code s
6  coverage save FSM_010.ucdb -onexit -du FSM_010
7  run -all
```

6. Code Coverage report snippets

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

----------------------------------------------------------------------------------------------------------
 TYPE /FSM_package/fsm_transaction/covgrp             100.00%        100          -    Covered
    covered/total bins:                                     1          1          -
    missing/total bins:                                     0          1          -
    % Hit:                                           100.00%        100          -
    Coverpoint cvl                                   100.00%        100          -    Covered
        covered/total bins:                                 1          1          -
        missing/total bins:                                 0          1          -
        % Hit:                                       100.00%        100          -
 Covergroup instance \/FSM_package::fsm_transaction::covgrp
                                                     100.00%        100          -    Covered
    covered/total bins:                                     1          1          -
    missing/total bins:                                     0          1          -
    % Hit:                                           100.00%        100          -
    Coverpoint cvl                                   100.00%        100          -    Covered
        covered/total bins:                                 1          1          -
        missing/total bins:                                 0          1          -
        % Hit:                                       100.00%        100          -
        bin seq_transition                               130          1          -    Covered

TOTAL COVERGROUP COVERAGE: 100.00%  COVERGROUP TYPES: 1

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

7. Functional Coverage report snippets

```
# ================================================================================
# === Instance: /\FSM_tb#DUT
# === Design Unit: work.FSM_010
# ================================================================================
# Branch Coverage:
#     Enabled Coverage              Bins      Hits    Misses  Coverage
#     ----------------              ----      ----    ------  --------
#     Branches                        20        20         0   100.00%
#
# ================================Branch Details================================
#
# Branch Coverage for instance /\FSM_tb#DUT
#
#     Line         Item                      Count     Source
#     ----         ----                      -----     ------
#   File FSm_010.v
# ------------------------------------CASE Branch------------------------------------
#     21                                     1002     Count coming in to CASE
#     22              1                       232        IDLE:
#
#     27              1                       359        ZERO:
#
#     32              1                       278        ONE:
#
#     37              1                       133        STORE:
#
# Branch totals: 4 hits of 4 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     23                                     232     Count coming in to IF
#     23              1                      107         if (x)
#
#     25              1                      125         else
#
# Branch totals: 2 hits of 2 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     28                                     359     Count coming in to IF
#     28              1                      174         if (x)
#
#     30              1                      185         else
#
# Branch totals: 2 hits of 2 branches = 100.00%
#
```

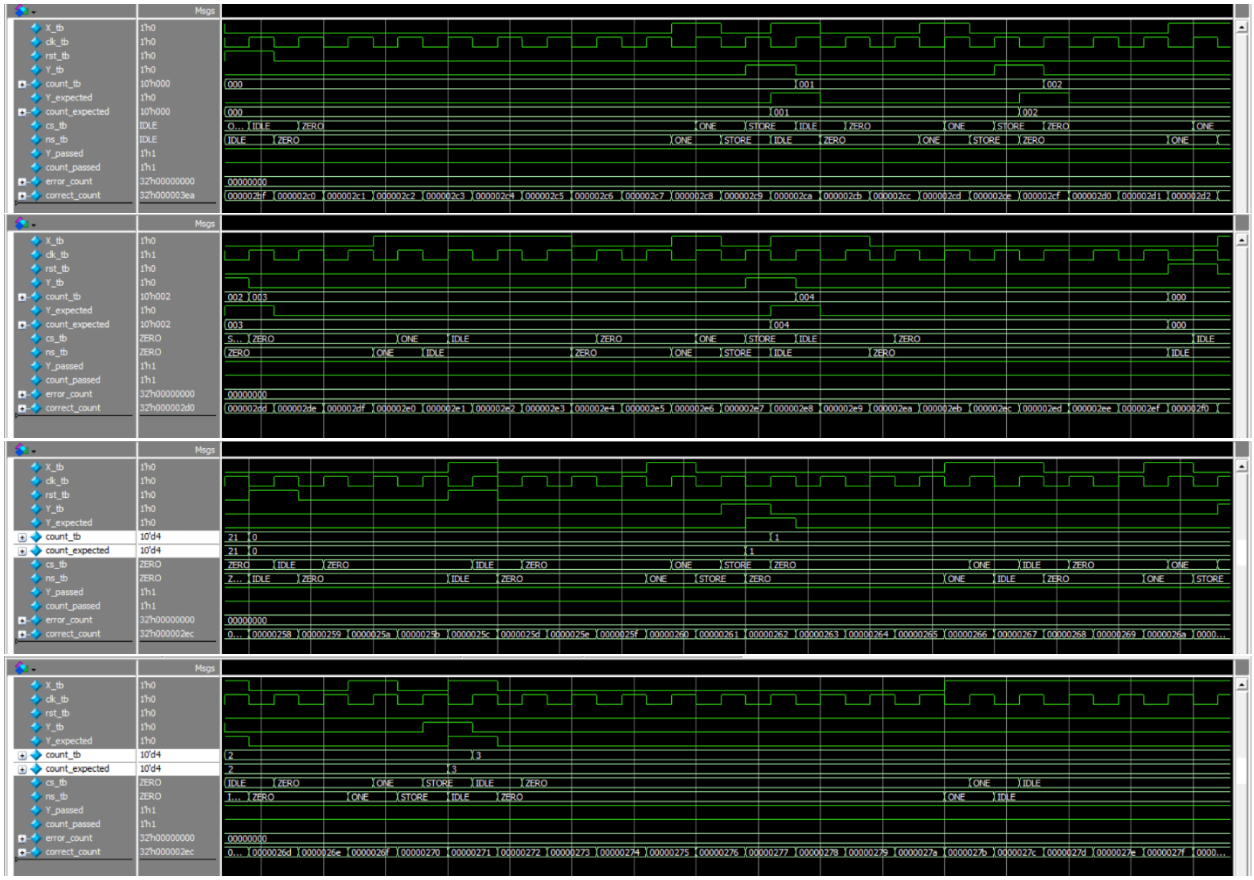
```
# ------------------------------------IF Branch------------------------------------
#     33                                     278     Count coming in to IF
#     33              1                      173         if (x)
#
#     35              1                      105         else
#
# Branch totals: 2 hits of 2 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     38                                     133     Count coming in to IF
#     38              1                       31         if (x)
#
#     40              1                      102         else
#
# Branch totals: 2 hits of 2 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     47                                     776     Count coming in to IF
#     47              1                       44       if(rst) begin
#
#     50              1                      732       else begin
#
# Branch totals: 2 hits of 2 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     56                                     614     Count coming in to IF
#     56              1                       44       if(rst) begin
#
#     59              1                      570       else begin
#
# Branch totals: 2 hits of 2 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     60                                     570     Count coming in to IF
#     60              1                      100         if (cs == STORE)
#
#                                            470     All False Count
# Branch totals: 2 hits of 2 branches = 100.00%
#
# ------------------------------------IF Branch------------------------------------
#     65                                     577     Count coming in to IF
#     65              1                      102      assign y = (cs == STORE)? 1:0;
#
#     65              2                      475      assign y = (cs == STORE)? 1:0;
#
# Branch totals: 2 hits of 2 branches = 100.00%
```

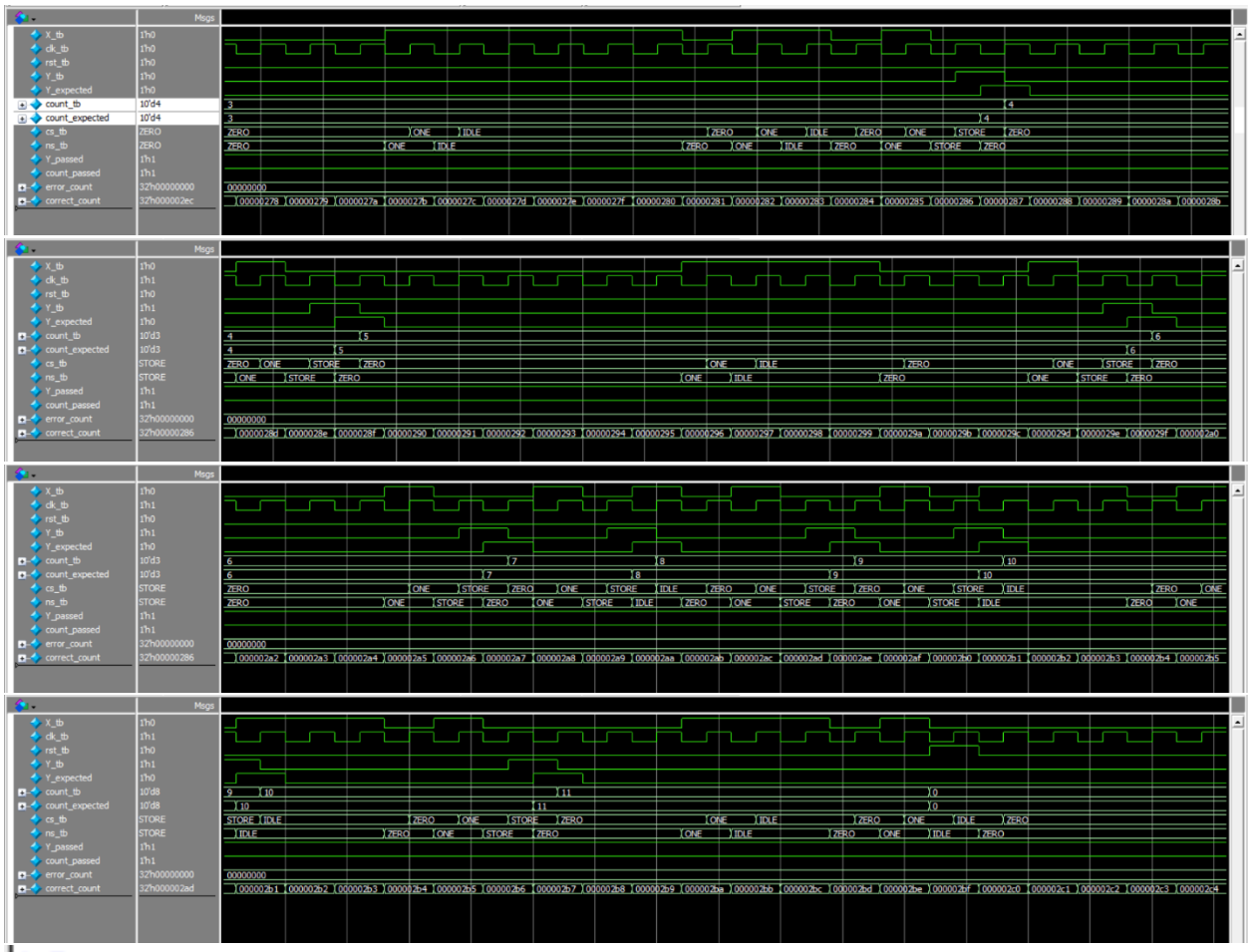
```
#
# Condition Coverage:
#     Enabled Coverage              Bins   Covered    Misses  Coverage
#     ----------------              ----      ----    ------  --------
#     Conditions                       2         2         0   100.00%
#
# ================================Condition Details================================
#
# Condition Coverage for instance /\FSM_tb#DUT  --
#
#   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 Details================================
#
# FSM Coverage for instance /\FSM_tb#DUT  --
#
# 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                 186
#                     ZERO                 315
#                      ONE                 173
#                    STORE                 102
#     Covered Transitions :
#     ---------------------
# Line            Trans_ID           Hit_count          Transition
# ----            --------           ---------          ----------
#   26                   0                 116          IDLE -> ZERO
#   29                   1                 173          ZERO -> ONE
#   48                   2                  12          ZERO -> IDLE
#   36                   3                 102          ONE -> STORE
#   34                   4                  71          ONE -> IDLE
#   41                   5                  69          STORE -> ZERO
#   39                   6                  33          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                     16        16         0   100.00%
#
```

```
# Toggle Coverage:
#     Enabled Coverage              Bins      Hits    Misses  Coverage
#     ----------------              ----      ----    ------  --------
#     Toggles                         26        26         0   100.00%
#
# ================================Toggle Details================================
#
# Toggle Coverage for instance /\FSM_tb#DUT  --
#
#                                          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[4-0]       1           1                                100.00
#                                             x           1           1                                100.00
#                                             y           1           1                                100.00
#
# Total Node Count     =          13
# Toggled Node Count   =          13
# Untoggled Node Count =           0
#
# Toggle Coverage      =     100.00% (26 of 26 bins)
#
#
# Total Coverage By Instance (filtered view): 100.00%
#
# End time: 19:29:55 on Sep 11,2025, Elapsed time: 0:00:00
# Errors: 0, Warnings: 0
```

8. Clear and neat QuestaSim waveform snippets showing the functionality of the design





```
# Test passes
# Test passes
# Test passes
# Test passes
# correct counter =        1002 , error counter =           0
# ** Note: $stop    : TESTBENCH.svh(44)
#    Time: 4008 ns  Iteration: 1  Instance: /FSM_tb
# Break in Module FSM_tb at TESTBENCH.svh line 44
```

P4)

1. Design code

```
module processor ();
    logic [24] memory [int] ;
    localparam first_address = 400 ;

    initial begin
        memory [0] = 24'hA50400;
        memory [first_address]     = 24'h123456;
        memory [first_address+1]   = 24'h789ABC;
        memory [first_address+220] = 24'h0F1E2D;

        $display("the number of elements in the array is %d" , memory.num);
        foreach (memory[i]) begin
            $display("the elemnt in the array of index %d is %h" , i , memory[i]);
        end
    end
endmodule
```

2. Clear and neat QuestaSim waveform snippets showing the functionality of the design

```
# the number of elements in the array is           4
# the elemnt in the array of index           0 is a50400
# the elemnt in the array of index         400 is 123456
# the elemnt in the array of index         401 is 789abc
# the elemnt in the array of index         620 is 0f1e2d
```

P5)

1. Design code

```
module Q5 ();
    typedef logic [6:0] section ;
    typedef struct  {
        section header ;
        section cmd ;
        section data;
        section crc ;
    } packet;

    initial begin
        packet my_packet ;
        my_packet.header = 7'h5A;
        $display("my packet header is %h" , my_packet.header);
        my_packet.cmd  = 7'h3C ;
        $display("my packet cmd is %h" , my_packet.cmd);
        my_packet.data = 7'h6E ;
        $display("my packet data is %h" , my_packet.data);
        my_packet.crc  = 7'h15 ;
        $display("my packet crc is %h" , my_packet.crc);
        $display("my packet is %p" , my_packet);
    end
endmodule
```

2. Clear and neat QuestaSim waveform snippets showing the functionality of the design

```
# my packet header is 5a
# my packet cmd is 3c
# my packet data is 6e
# my packet crc is 15
# my packet is '{header:90, cmd:60, data:110, crc:21}
```