RAM

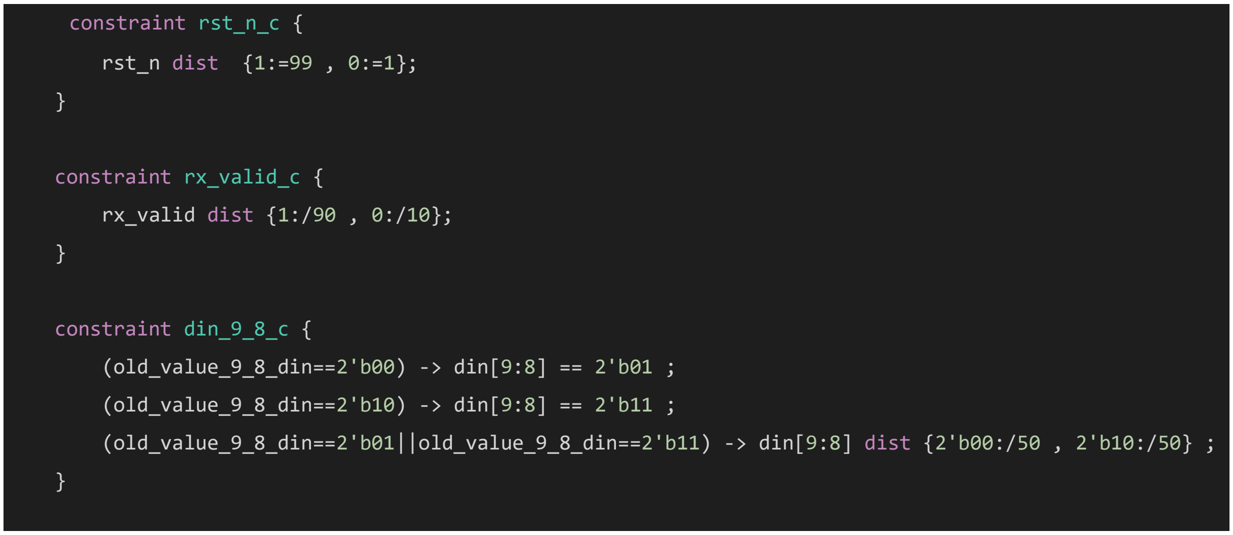
Test plan:

Function	How to test
Reset	Make assertions on values after reset
Write address	Assert if the din [9:8]==00 the next will be write data which is din[9:8]==01
Read address	Similar constrain input and cover point then make assertions
Write data	Similar constrain input and cover point then make assertions
Read data	Similar constrain input and cover point then make assertions

Run.do file:

```
vlib work
vlog -f src_files.txt
vsim -voptargs=+acc work.top -classdebug -uvmcontrol=all
add wave -position insertpoint sim:/top/ram_regif/*
coverage save ram.ucdb -onexit
run -all
```

Constrains randomization to input:

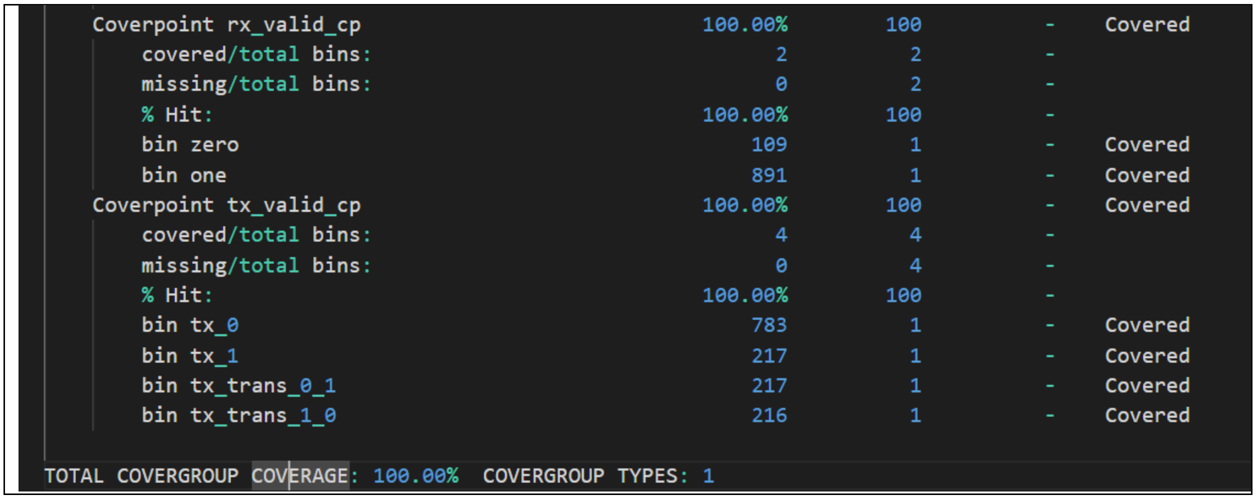


```
     constraint rst_n_c {
        rst_n dist  {1:=99 , 0:=1};
    }

    constraint rx_valid_c {
        rx_valid dist {1:/90 , 0:/10};
    }

    constraint din_9_8_c {
        (old_value_9_8_din==2'b00) -> din[9:8] == 2'b01 ;
        (old_value_9_8_din==2'b10) -> din[9:8] == 2'b11 ;
        (old_value_9_8_din==2'b01||old_value_9_8_din==2'b11) -> din[9:8] dist {2'b00:/50 , 2'b10:/50} ;
    }
```

Constrain din [9:8] if the previous randomization is 00 next to be 10. (Write flow)

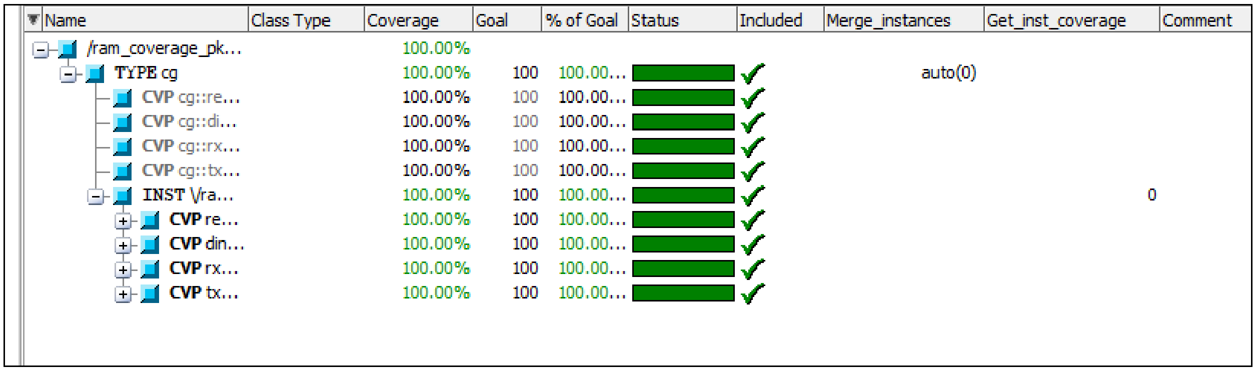
Constrain din [9:8] if the previous randomization is 10 next to be 11. (Read flow)

Code coverage:



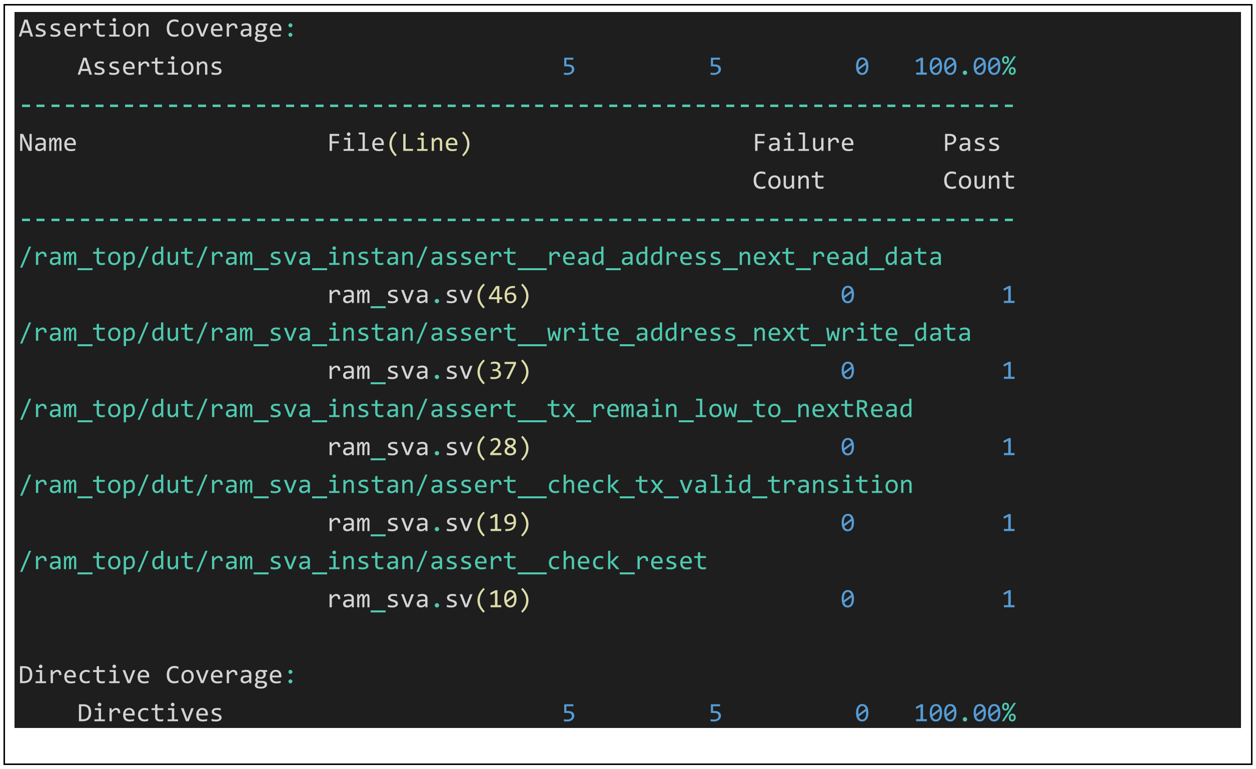
```
    Coverpoint rx_valid_cp                          100.00%       100           -     Covered
        covered/total bins:                               2         2           -
        missing/total bins:                               0         2           -
        % Hit:                                      100.00%       100           -
        bin zero                                        109         1           -     Covered
        bin one                                         891         1           -     Covered
    Coverpoint tx_valid_cp                          100.00%       100           -     Covered
        covered/total bins:                               4         4           -
        missing/total bins:                               0         4           -
        % Hit:                                      100.00%       100           -
        bin tx_0                                        783         1           -     Covered
        bin tx_1                                        217         1           -     Covered
        bin tx_trans_0_1                                217         1           -     Covered
        bin tx_trans_1_0                                216         1           -     Covered

TOTAL COVERGROUP COVERAGE: 100.00%  COVERGROUP TYPES: 1
```

Cover groups form Questa:



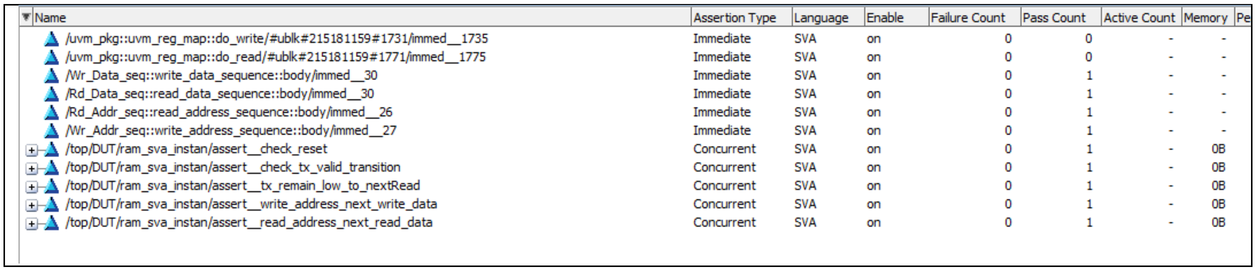
Name	Class Type	Coverage	Goal	% of Goal	Status	Included	Merge_instances	Get_inst_coverage	Comment
/ram_coverage_pk...		100.00%							
TYPE cg		100.00%	100	100.00...	🟩	✓	auto(0)		
CVP cg::re...		100.00%	100	100.00...	🟩	✓			
CVP cg::di...		100.00%	100	100.00...	🟩	✓			
CVP cg::rx...		100.00%	100	100.00...	🟩	✓			
CVP cg::tx...		100.00%	100	100.00...	🟩	✓			
INST \/ra...		100.00%	100	100.00...	🟩	✓		0	
CVP re...		100.00%	100	100.00...	🟩	✓			
CVP din...		100.00%	100	100.00...	🟩	✓			
CVP rx...		100.00%	100	100.00...	🟩	✓			
CVP tx...		100.00%	100	100.00...	🟩	✓			

Assertion’s coverage:



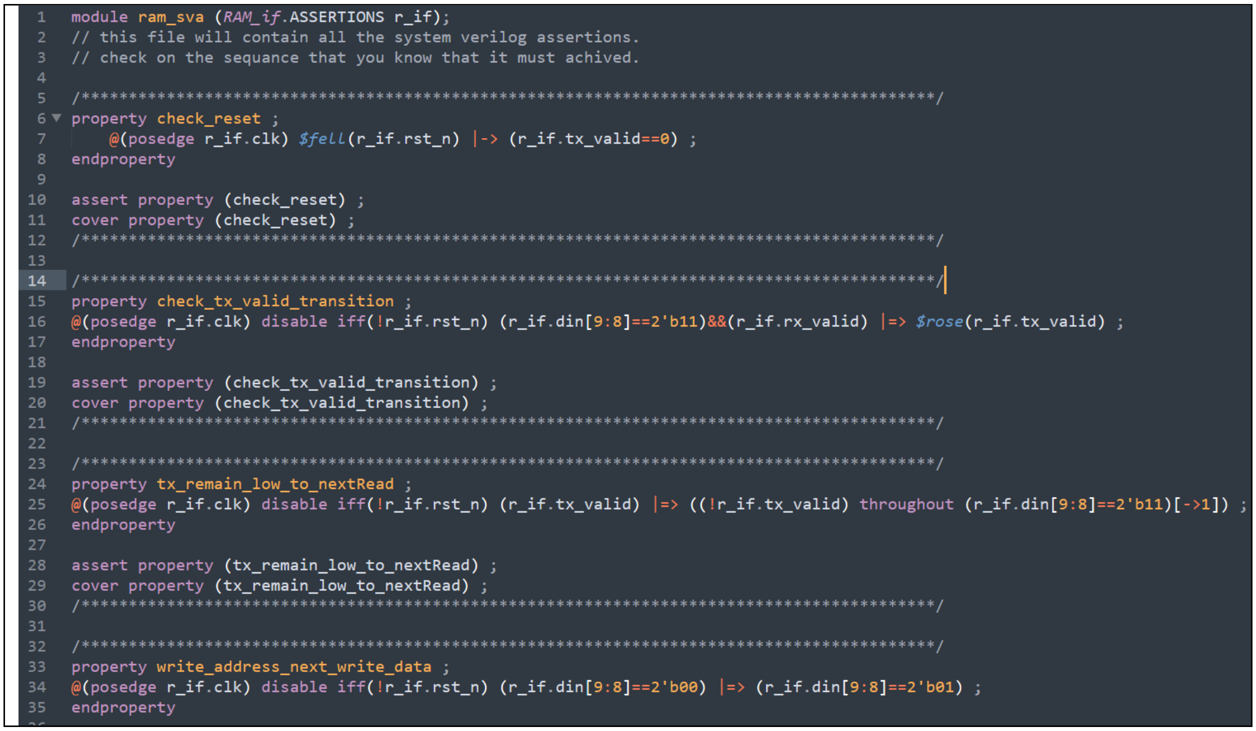
```
Assertion Coverage:
    Assertions                     5         5         0    100.00%
--------------------------------------------------------------------
Name                    File(Line)                   Failure      Pass
                                                     Count        Count
--------------------------------------------------------------------
/ram_top/dut/ram_sva_instan/assert__read_address_next_read_data
                     ram_sva.sv(46)                      0           1
/ram_top/dut/ram_sva_instan/assert__write_address_next_write_data
                     ram_sva.sv(37)                      0           1
/ram_top/dut/ram_sva_instan/assert__tx_remain_low_to_nextRead
                     ram_sva.sv(28)                      0           1
/ram_top/dut/ram_sva_instan/assert__check_tx_valid_transition
                     ram_sva.sv(19)                      0           1
/ram_top/dut/ram_sva_instan/assert__check_reset
                     ram_sva.sv(10)                      0           1

Directive Coverage:
    Directives                     5         5         0    100.00%
```

Assertions from Questa sim:



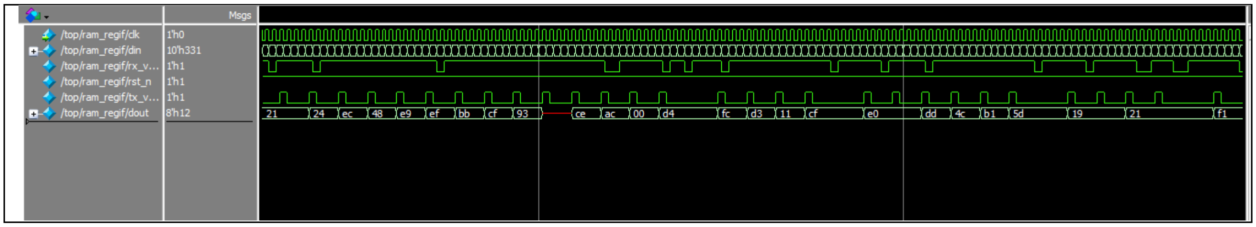
Name	Assertion Type	Language	Enable	Failure Count	Pass Count	Active Count	Memory	Pe
/uvm_pkg::uvm_reg_map::do_write/#ublk#215181159#1731/immed__1735	Immediate	SVA	on	0	0	-	-	
/uvm_pkg::uvm_reg_map::do_read/#ublk#215181159#1771/immed__1775	Immediate	SVA	on	0	0	-	-	
/Wr_Data_seq::write_data_sequence::body/immed__30	Immediate	SVA	on	0	1	-	-	
/Rd_Data_seq::read_data_sequence::body/immed__30	Immediate	SVA	on	0	1	-	-	
/Rd_Addr_seq::read_address_sequence::body/immed__26	Immediate	SVA	on	0	1	-	-	
/Wr_Addr_seq::write_address_sequence::body/immed__27	Immediate	SVA	on	0	1	-	-	
/top/DUT/ram_sva_instan/assert__check_reset	Concurrent	SVA	on	0	1	-	0B	
/top/DUT/ram_sva_instan/assert__check_tx_valid_transition	Concurrent	SVA	on	0	1	-	0B	
/top/DUT/ram_sva_instan/assert__tx_remain_low_to_nextRead	Concurrent	SVA	on	0	1	-	0B	
/top/DUT/ram_sva_instan/assert__write_address_next_write_data	Concurrent	SVA	on	0	1	-	0B	
/top/DUT/ram_sva_instan/assert__read_address_next_read_data	Concurrent	SVA	on	0	1	-	0B	

Assertions in code:

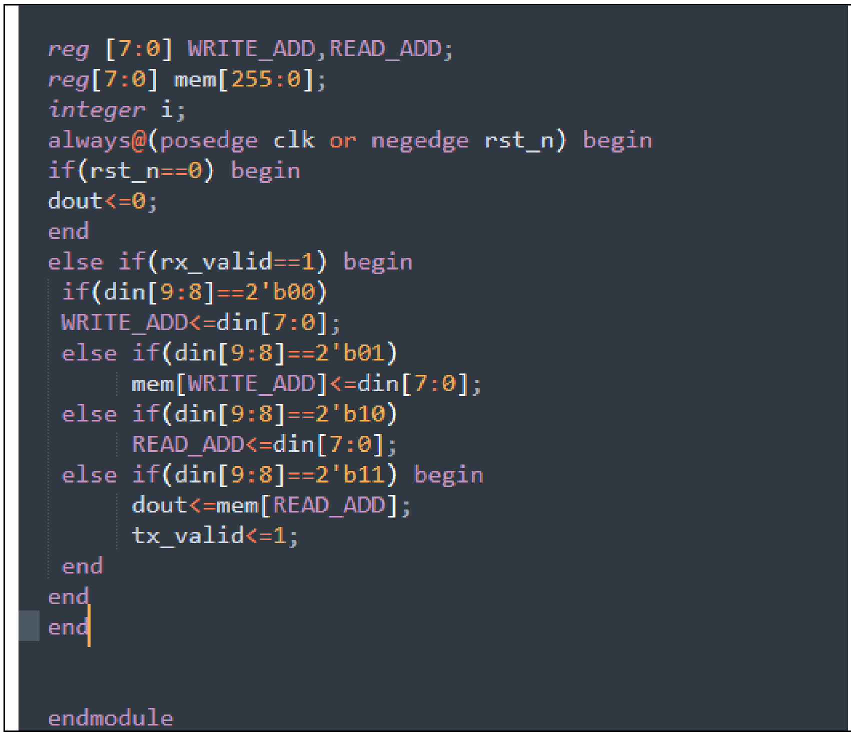


```
module ram_sva (RAM_if.ASSERTIONS r_if);
// this file will contain all the system verilog assertions.
// check on the sequance that you know that it must achived.

/*********************************************************************************/
property check_reset ;
    @(posedge r_if.clk) $fell(r_if.rst_n) |-> (r_if.tx_valid==0) ;
endproperty

assert property (check_reset) ;
cover property (check_reset) ;
/*********************************************************************************/

/*********************************************************************************/
property check_tx_valid_transition ;
@(posedge r_if.clk) disable iff(!r_if.rst_n) (r_if.din[9:8]==2'b11)&&(r_if.rx_valid) |=> $rose(r_if.tx_valid) ;
endproperty

assert property (check_tx_valid_transition) ;
cover property (check_tx_valid_transition) ;
/*********************************************************************************/

/*********************************************************************************/
property tx_remain_low_to_nextRead ;
@(posedge r_if.clk) disable iff(!r_if.rst_n) (r_if.tx_valid) |=> ((!r_if.tx_valid) throughout (r_if.din[9:8]==2'b11)[->1]) ;
endproperty

assert property (tx_remain_low_to_nextRead) ;
cover property (tx_remain_low_to_nextRead) ;
/*********************************************************************************/

/*********************************************************************************/
property write_address_next_write_data ;
@(posedge r_if.clk) disable iff(!r_if.rst_n) (r_if.din[9:8]==2'b00) |=> (r_if.din[9:8]==2'b01) ;
endproperty
```

Results on wave form:


Msgs
/top/ram_regif/clk 1'h0
/top/ram_regif/din 10'h331
/top/ram_regif/rx_v... 1'h1
/top/ram_regif/rst_n 1'h1
/top/ram_regif/tx_v... 1'h1
/top/ram_regif/dout 8'h12
21 24 ec 48 e9 ef bb cf 93 ce ac 00 d4 fc d3 11 cf e0 dd 4c b1 5d 19 21 f1

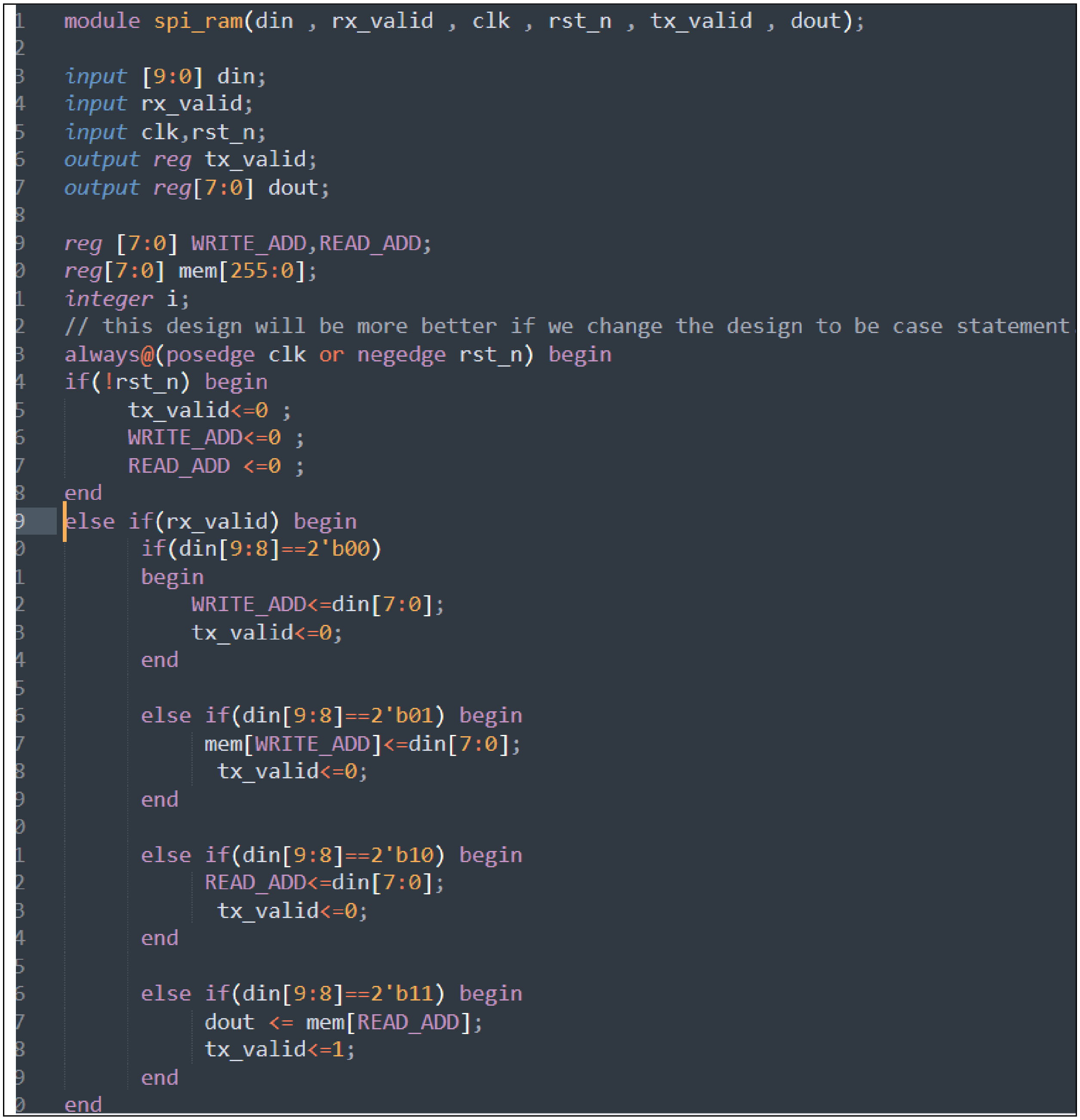
Bug report:



```
reg [7:0] WRITE_ADD,READ_ADD;
reg[7:0] mem[255:0];
integer i;
always@(posedge clk or negedge rst_n) begin
if(rst_n==0) begin
dout<=0;
end
else if(rx_valid==1) begin
 if(din[9:8]==2'b00)
 WRITE_ADD<=din[7:0];
 else if(din[9:8]==2'b01)
      mem[WRITE_ADD]<=din[7:0];
 else if(din[9:8]==2'b10)
      READ_ADD<=din[7:0];
 else if(din[9:8]==2'b11) begin
      dout<=mem[READ_ADD];
      tx_valid<=1;
 end
end
end


endmodule
```

1- **tx_valid must be low for all cases except read data .**
2- **on reset WRITE_ADD<=0 ; READ_ADD <=0 ; should be zero .**
3- **also we should handle the case if rst_n = 1 and rx_valid not equal =1 ;**

code after correctness:



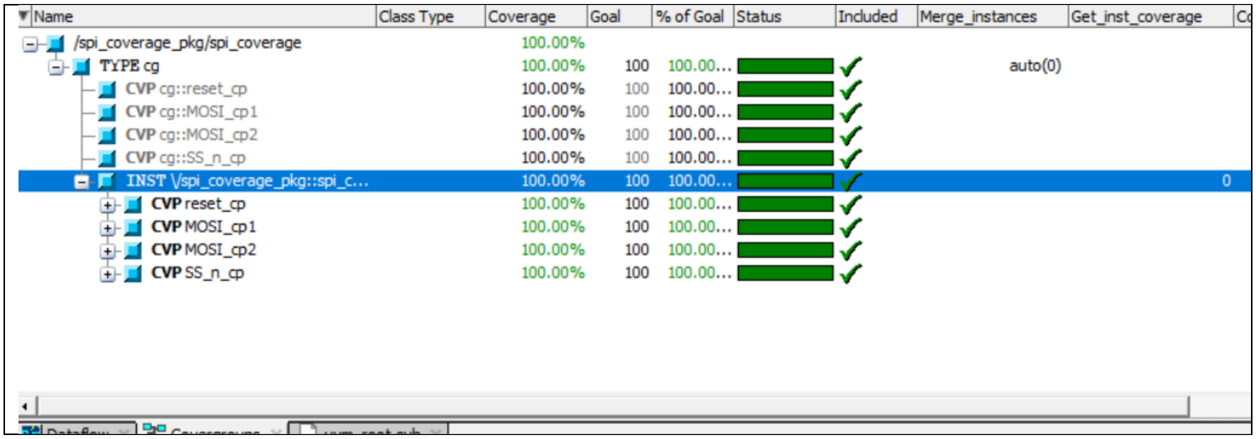
```
module spi_ram(din , rx_valid , clk , rst_n , tx_valid , dout);

input [9:0] din;
input rx_valid;
input clk,rst_n;
output reg tx_valid;
output reg[7:0] dout;

reg [7:0] WRITE_ADD,READ_ADD;
reg[7:0] mem[255:0];
integer i;
// this design will be more better if we change the design to be case statement
always@(posedge clk or negedge rst_n) begin
if(!rst_n) begin
     tx_valid<=0 ;
     WRITE_ADD<=0 ;
     READ_ADD <=0 ;
end
else if(rx_valid) begin
      if(din[9:8]==2'b00)
      begin
          WRITE_ADD<=din[7:0];
          tx_valid<=0;
      end

      else if(din[9:8]==2'b01) begin
           mem[WRITE_ADD]<=din[7:0];
            tx_valid<=0;
      end

      else if(din[9:8]==2'b10) begin
           READ_ADD<=din[7:0];
            tx_valid<=0;
      end

      else if(din[9:8]==2'b11) begin
           dout <= mem[READ_ADD];
           tx_valid<=1;
      end
end
```

SPI

Run.do file:

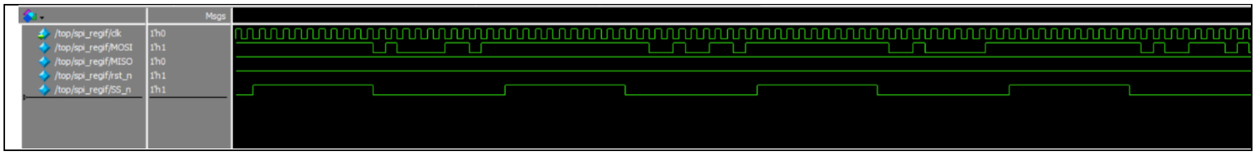
```
vlib work
vlog -f src_files.txt
vsim -voptargs=+acc work.top -classdebug -uvmcontrol=all
add wave -position insertpoint sim:/top/ram_regif/*
coverage save spi.ucdb -onexit
run -all
```

Cover groups form Questa:



Name	Class Type	Coverage	Goal	% of Goal	Status	Included	Merge_instances	Get_inst_coverage	Cc
/spi_coverage_pkg/spi_coverage		100.00%							
TYPE cg		100.00%	100	100.00...	🟩	✓	auto(0)		
CVP cg::reset_cp		100.00%	100	100.00...	🟩	✓			
CVP cg::MOSI_cp1		100.00%	100	100.00...	🟩	✓			
CVP cg::MOSI_cp2		100.00%	100	100.00...	🟩	✓			
CVP cg::SS_n_cp		100.00%	100	100.00...	🟩	✓			
INST \/spi_coverage_pkg::spi_c...		100.00%	100	100.00...	🟩	✓		0	
CVP reset_cp		100.00%	100	100.00...	🟩	✓			
CVP MOSI_cp1		100.00%	100	100.00...	🟩	✓			
CVP MOSI_cp2		100.00%	100	100.00...	🟩	✓			
CVP SS_n_cp		100.00%	100	100.00...	🟩	✓			

Results on wave form:



	Msgs
/top/spi_regif/clk	1'h0
/top/spi_regif/MOSI	1'h1
/top/spi_regif/MISO	1'h0
/top/spi_regif/rst_n	1'h1
/top/spi_regif/SS_n	1'h1

Bug report:

there wasn’t any bugs