SPI WRAPPER VERIFICATION

Under the supervision of Eng. Kareem Waseem

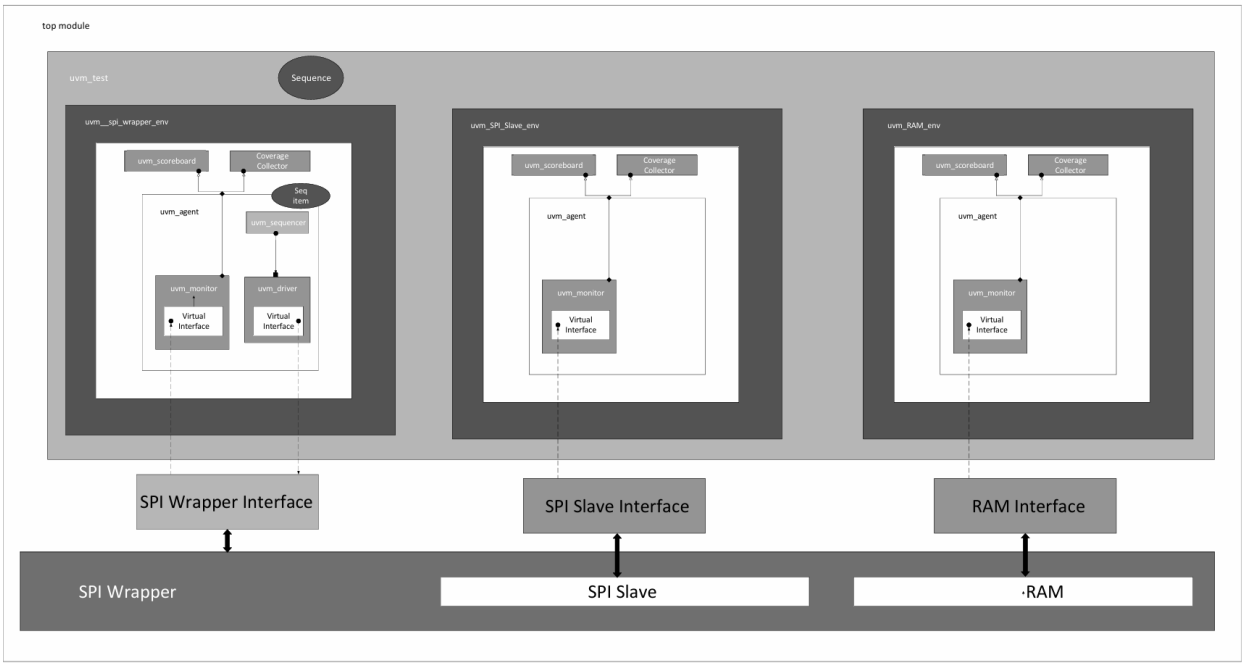
Prepared By:
Hazem Mohamed Ahmed Elsheshtawy
Saeed Maher Saeed
Sameh Mohammed Sameh

Table of Contents

Verification Plan

Label	Design Required Description	Stimulus Generation	Functional Coverage	Functionality Check	Part
RAM Reset Test	On reset, all memory outputs must be cleared.	Directed at simulation start (rst_n=0).	reset_sva cover property in ram_sva.	Assertion confirms RAM output remains 0 under reset and Comparison between DUT and golden RAM read outputs.	1
RAM Write Test	Verify correct data storage at all valid addresses.	Randomized write operations through the UVM sequence.	Address and data bins in functional coverage.	Assertions check data integrity between written and read values and Comparison between DUT and golden RAM read outputs.	1
RAM Read Test	Ensure correct data retrieval from stored locations.	Read operations after writes using randomized addressing.	Cross coverage between read addresses and data values.	Comparison between DUT and golden RAM read outputs.	1
SLAVE Reset Test	When reset is asserted, all internal outputs must be inactive.	Directed at start of simulation (rst_n=0).	reset_sva cover property in slave_sva.	Assertion checks output remains inactive during reset and Comparison between DUT and golden RAM read outputs.	2
SLAVE Communication Test	During SPI communication, MISO must be stable and MOSI transitions valid.	SPI clock and chip-select patterns driven by UVM sequence.	Coverage on SS_n and MOSI transitions cross (SS_n_with_MOSI).	Assertions ensure correct timing and data validity and Comparison between DUT and golden RAM read outputs.	2
Reset Test (Wrapper)	When reset is asserted, all outputs (e.g., MISO) must remain inactive.	Directed stimulus applied at simulation start by driving rst_n = 0.	reset_check cover property in SPI_wrapper_sva.	Assertion reset_check ensures MISO is low during reset and Comparison between DUT and golden RAM read outputs.	3
MISO Stability Test (Wrapper)	MISO should remain stable during communication periods excluding read operations.	Sequence generated by toggling SS_n to simulate communication.	MISO_stable_check cover property in SPI_wrapper_sva.	Assertion ensures $stable(MISO) when no read data is transmitted and Comparison between DUT and golden RAM read outputs.	3
Wrapper Integration Test	Ensure RAM and SLAVE receive same inputs as DUT wrapper.	assign statements used to replicate DUT inputs to interfaces.	Coverage collected on input consistency across interfaces.	Monitors confirm input synchronization between DUT and golden models.	3

UVM Testbench Structure



Part 1: UVM Environment for SPI-Slave

SPI Slave Interface

```
interface SLAVE_interface(clk);
    input clk;

    logic MOSI, rst_n, SS_n, tx_valid;
    logic [7:0] tx_data;
    logic [9:0] rx_data;
    logic rx_valid, MISO;
    logic [9:0] rx_data_golden;
    logic rx_valid_golden, MISO_golden;
endinterface
```

SPI Slave with Assertions

```
module SLAVE (MOSI,MISO,SS_n,clk,rst_n,rx_data,rx_valid,tx_data,tx_valid);

localparam IDLE      = 3'b000;
localparam WRITE     = 3'b001;
localparam CHK_CMD   = 3'b010;
localparam READ_ADD  = 3'b011;
localparam READ_DATA = 3'b100;

input MOSI, clk, rst_n, SS_n, tx_valid;
input [7:0] tx_data;
output reg [9:0] rx_data;
output reg rx_valid, MISO;

reg [3:0] counter;
reg received_address;

reg [2:0] cs, ns;

always @(posedge clk) begin
    if (~rst_n) begin
        cs <= IDLE;
    end
    else begin
        cs <= ns;
    end
end

always @(*) begin
```

```
    case (cs)
        IDLE : begin
            if (SS_n)
                ns = IDLE;
            else
                ns = CHK_CMD;
        end
        CHK_CMD : begin
            if (SS_n)
                ns = IDLE;
            else begin
                if (~MOSI)
                    ns = WRITE;
                else begin
                    if (!received_address)
                        ns = READ_ADD;
                    else
                        ns = READ_DATA;
                end
            end
        end
        WRITE : begin
            if (SS_n)
                ns = IDLE;
            else
                ns = WRITE;
        end
        READ_ADD : begin
            if (SS_n)
                ns = IDLE;
            else
                ns = READ_ADD;
        end
        READ_DATA : begin
            if (SS_n)
                ns = IDLE;
            else
                ns = READ_DATA;
        end
    endcase
end

always @(posedge clk) begin
    if (~rst_n) begin
        rx_data <= 0;
```

```
        rx_valid <= 0;
        received_address <= 0;
        MISO <= 0;
    end
    else begin
        case (cs)
            IDLE : begin
                rx_valid <= 0;
            end
            CHK_CMD : begin
                counter <= 10;
            end
            WRITE : begin
                if (counter > 0) begin
                    rx_data[counter-1] <= MOSI;
                    counter <= counter - 1;
                end
                else begin
                    rx_valid <= 1;
                end
            end
            READ_ADD : begin
                if (counter > 0) begin
                    rx_data[counter-1] <= MOSI;
                    counter <= counter - 1;
                end
                else begin
                    rx_valid <= 1;
                    received_address <= 1;
                end
            end
            READ_DATA : begin
                if (tx_valid) begin
                    rx_valid <= 0;
                    if (counter > 0) begin
                        MISO <= tx_data[counter-1];
                        counter <= counter - 1;
                    end
                    else begin
                        received_address <= 0;
                    end
                end
                else begin
                    if (counter > 0 && ~rx_valid) begin
                        rx_data[counter-1] <= MOSI;
```

```
                        counter <= counter - 1;
                    end
                    else begin
                        rx_valid <= 1;
                        counter <= 8;
                    end
                end
            end
            default: rx_valid <= 0;
        endcase
    end
end

//assertion

`ifdef SIM

property sync_reset;
    @(posedge clk)
    !rst_n |=> ((rx_data == 0) && (rx_valid == 0) && (MISO == 0)) ;
endproperty

assert property (sync_reset);
cover property (sync_reset);

sequence write_address;
    $fell(SS_n) ##1 (MOSI==0)[*3];
endsequence

sequence write_data;
    $fell(SS_n) ##1 (MOSI==0)[*2] ##1 (MOSI==1);
endsequence

sequence read_address;
    $fell(SS_n) ##1 (MOSI==1)[*2] ##1 (MOSI==0);
endsequence

sequence read_data;
    $fell(SS_n) ##1 (MOSI==1)[*3] ;
endsequence

sequence end_of_comm;
    rx_valid && SS_n;
endsequence
```

```
property write_address_comm;
    @(posedge clk) disable iff(!rst_n)
    write_address |-> ##10 end_of_comm;
endproperty

assert property (write_address_comm);
cover property (write_address_comm);

property write_data_comm;
    @(posedge clk) disable iff(!rst_n)
    write_data |-> ##10 end_of_comm;
endproperty

assert property (write_data_comm);
cover property (write_data_comm);

property read_address_comm;
    @(posedge clk) disable iff(!rst_n)
    read_address |-> ##10 end_of_comm;
endproperty

assert property (read_address_comm);
cover property (read_address_comm);

property read_data_comm;
    @(posedge clk) disable iff(!rst_n)
    read_data |-> ##10 rx_valid |-> ##10 SS_n;
endproperty

assert property (read_data_comm);
cover property (read_data_comm);

//last one



// 1. IDLE -> CHK_CMD
property IDLE_to_CHK_CMD;
    @(posedge clk) disable iff(!rst_n)
    (cs == IDLE) &&(!SS_n) |=> (cs == CHK_CMD)
endproperty

assert property (IDLE_to_CHK_CMD);
cover property (IDLE_to_CHK_CMD);
```

```
// 2. CHK_CMD -> WRITE or READ_ADD or READ_DATA
property CHK_CMD_to_WRITE;
    @(posedge clk) disable iff(!rst_n)
    (cs == CHK_CMD) && (!SS_n) && (!MOSI) |=> (cs == WRITE);
endproperty

assert property (CHK_CMD_to_WRITE);
cover property (CHK_CMD_to_WRITE);

property CHK_CMD_to_READ_ADD;
    @(posedge clk) disable iff(!rst_n)
    (cs == CHK_CMD) && (!SS_n) && (MOSI) && (!received_address) |=> (cs ==
READ_ADD);
endproperty

assert property (CHK_CMD_to_READ_ADD);
cover property (CHK_CMD_to_READ_ADD);

property CHK_CMD_to_READ_DATA;
    @(posedge clk) disable iff(!rst_n)
    (cs == CHK_CMD) && (!SS_n) && (MOSI) && (received_address) |=> (cs ==
READ_DATA);
endproperty

assert property (CHK_CMD_to_READ_DATA);
cover property (CHK_CMD_to_READ_DATA);


// 3. WRITE -> IDLE
property WRITE_to_IDLE;
    @(posedge clk) disable iff(!rst_n)
    (cs == WRITE) && (SS_n) |=> (cs == IDLE);
endproperty

assert property (WRITE_to_IDLE);
cover property (WRITE_to_IDLE);


// 4. READ_ADD -> IDLE
property READ_ADD_to_IDLE;
    @(posedge clk) disable iff(!rst_n)
    (cs == READ_ADD) && (SS_n) |=> (cs == IDLE);
endproperty

assert property (READ_ADD_to_IDLE);
```

```
cover property (READ_ADD_to_IDLE);


// 5. READ_DATA -> IDLE
property READ_DATA_to_IDLE;
    @(posedge clk) disable iff(!rst_n)
    (cs == READ_DATA) && (SS_n) |=> (cs == IDLE);
endproperty

assert property (READ_DATA_to_IDLE);
cover property (READ_DATA_to_IDLE);
`endif // SIM

endmodule
```

- We put all assertion in a conditional compilation to make the design synthesizable.

SPI Slave Golden Model

```
module
SPI_slave_golden(MOSI,SS_n,clk,rst_n,tx_valid,tx_data,MISO,rx_valid,rx_data);
  input MOSI,SS_n,clk,rst_n,tx_valid;
  input [7:0]tx_data;
  output reg MISO,rx_valid;
  output reg [9:0]rx_data;
  reg [2:0]cs,ns;
  reg address_enable;
  reg [3:0]count;
  localparam IDLE = 3'b000;
  localparam CHK_CMD = 3'b010;
  localparam WRITE = 3'b001;
  localparam READ_ADD = 3'b011;
  localparam READ_DATA = 3'b100;

  always @(posedge clk) begin
      if (~rst_n) begin
          cs<=IDLE;
      end
      else begin
          cs<=ns;
      end
  end

  always @(*) begin
      case(cs)
```

```
        IDLE: begin
          if(~SS_n) begin
            ns=CHK_CMD;
          end
        end

        CHK_CMD: begin
          if (SS_n) begin
            ns=IDLE;
          end
          else if (~MOSI) begin
            ns=WRITE;
          end
          else begin
            if (~address_enable) begin
                ns=READ_ADD;
            end else begin
                ns=READ_DATA;
            end
          end
        end

        WRITE: begin
          if(SS_n == 1) begin
            ns=IDLE;
          end
          else begin
            ns=WRITE;
          end
        end

        READ_ADD: begin
          if(SS_n == 1) begin
            ns=IDLE;
          end
          else begin
            ns=READ_ADD;
          end
        end

        READ_DATA: begin
          if(SS_n == 1) begin
            ns=IDLE;
          end
          else begin
```

```
            ns=READ_DATA;
          end
        end
    endcase
end

always @(posedge clk) begin
    if (~rst_n) begin
        MISO <= 0;
        rx_data <= 0;
        rx_valid <= 0;
        count <= 0;
        address_enable <= 0;
    end
    else begin
        case (cs)

          IDLE: rx_valid <= 0;

          CHK_CMD: count <= 0;

          WRITE: begin
              if (count < 10) begin
                rx_data[9 - count] <= MOSI;
                count <= count + 1;
              end
              else if (count == 10) begin
                  rx_valid <= 1;
              end
          end

          READ_ADD: begin
            if (count<10) begin
                rx_data[9 - count] <= MOSI;
                count <= count + 1;
            end
            else if (count == 10) begin
                  rx_valid <= 1;
                  address_enable <= 1;
                  count <= count + 1;
            end
          end

          READ_DATA: begin
            if (rx_valid && ~tx_valid) begin
```

```
                count <= 0;
              end
              else if (count<10 && ~tx_valid) begin
                  rx_data[9 - count] <= MOSI;
                  count <= count + 1;
              end
              else if (count == 10 && ~tx_valid) begin
                    rx_valid <= 1;
              end
              else if (tx_valid && count < 8) begin
                address_enable <= 0;
                rx_valid <= 0;
                MISO <= tx_data[7 - count];
                count <= count + 1;
              end
            end
          endcase
      end
  end
endmodule
```

SPI Shared Package

```
package shared_pkg;

    bit[5:0] count;
    int limit;
    logic [10:0] keep_arr;
    bit is_read;
    bit have_address_to_read;

endpackage
```

SPI Slave Sequence Item

```
package SPI_slave_seq_item_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import shared_pkg::*;


    class SPI_slave_seq_item extends uvm_sequence_item;
    `uvm_object_utils(SPI_slave_seq_item)

        //inputs and output
```

```
        rand logic rst_n, SS_n, tx_valid, MOSI;
        rand logic [7:0] tx_data;
        logic [9:0] rx_data;
        rand logic [10:0] arr_MOSI;
        logic rx_valid,MISO;
        logic rx_valid_golden,MISO_golden;
        logic [9:0] rx_data_golden;

        //constructor
        function new (string name = "SPI_slave_seq_item");
            super.new(name);
        endfunction

        //constraints

        //reset
        constraint reset_rate {
            rst_n dist {0:=1,1:=99};
        }

        constraint valid_MOSI_command {
            arr_MOSI[10:8] inside {3'b000, 3'b001, 3'b110, 3'b111};
            if(!have_address_to_read) {(arr_MOSI[10:8] != {3'b111});}
        }

        constraint ready_to_read {
            if(count >= 15) tx_valid ==1;
            else tx_valid == 0;
        }

        function void post_randomize();

            if(count == 0) keep_arr = arr_MOSI;

            is_read = (keep_arr[10:8] == 3'b111)? 1:0;
            limit = (is_read)? 23:13;

            SS_n = (count == limit)? 1:0;

            if(keep_arr[10:8] == 3'b110) have_address_to_read = 1'b1;
            if (is_read || (!rst_n)) have_address_to_read = 1'b0;

            //
            if((count > 0) && (count < 12)) begin
                MOSI = keep_arr [11-count];
```

```
            end

            //count
            if (!rst_n) begin
                count = 0;
            end
            else begin
                if (count == limit) count = 0;
                else count++;
            end

        endfunction
    endclass

endpackage
```

SPI Slave Reset Sequence

```
package SPI_slave_reset_seq_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import SPI_slave_seq_item_pkg::*;


    class SPI_slave_reset_seq extends uvm_sequence #(SPI_slave_seq_item);
    `uvm_object_utils(SPI_slave_reset_seq)

        //seq item
        SPI_slave_seq_item seq_item;

        //construcotr
        function new(string name = "SPI_slave_reset_seq");
            super.new(name);
        endfunction //new()

        //body
        task body();

            seq_item = SPI_slave_seq_item::type_id::create("seq_item");
            start_item(seq_item);
            seq_item.rst_n = 0;
            seq_item.MOSI = 1;
            seq_item.SS_n = 0;
            seq_item.tx_valid = 0;
            seq_item.tx_data = 4;
            finish_item(seq_item);
```

```
        endtask

    endclass //className extends superClass
endpackage
```

SPI Slave Main Sequence

```
package SPI_slave_main_seq_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import SPI_slave_seq_item_pkg::*;


    class SPI_slave_main_seq extends uvm_sequence #(SPI_slave_seq_item);
    `uvm_object_utils(SPI_slave_main_seq)

        //seq item
        SPI_slave_seq_item seq_item;

        //constructor
        function new(string name = "SPI_slave_main_seq");
            super.new(name);
        endfunction //new()

        //body
        task body();

            repeat(50000) begin
                seq_item = SPI_slave_seq_item::type_id::create("seq_item");
                start_item(seq_item);
                assert (seq_item.randomize);
                finish_item(seq_item);
            end
        endtask

    endclass //className extends superClass


endpackage
```

SPI Slave Sequencer

```
package SPI_slave_sqr_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import SPI_slave_seq_item_pkg::*;

    class SPI_slave_sqr extends uvm_sequencer #(SPI_slave_seq_item);
    `uvm_component_utils(SPI_slave_sqr)

        //constructor
        function new(string name = "SPI_slave_sqr", uvm_component parent = null);
            super.new(name,parent);
        endfunction //new()


    endclass //className extends superClass

endpackage
```

SPI Slave Configuration Object

```
package SPI_slave_config_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import uvm_pkg::*;

    class SPI_slave_config extends uvm_object;
    `uvm_object_utils(SPI_slave_config)

        virtual SLAVE_interface SPI_slave_vif;
        uvm_active_passive_enum is_active;

        //constructor
        function new(string name = "SPI_slave_config");
            super.new(name);
        endfunction //new()


    endclass //className extends superClass

endpackage
```

SPI Slave Driver

```
package SPI_slave_driver_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import SPI_slave_seq_item_pkg::*;
import shared_pkg::*;

    class SPI_slave_driver extends uvm_driver #(SPI_slave_seq_item);
    `uvm_component_utils (SPI_slave_driver)

        //virtual interface
        virtual SLAVE_interface SPI_slave_vif;

        //seq item
        SPI_slave_seq_item seq_item;

        //constructor
        function new(string name = "SPI_slave_driver", uvm_component parent =
null);
            super.new(name,parent);
        endfunction //new()

        //build phase
        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
        endfunction
        //run
        task run_phase(uvm_phase phase);
            super.run_phase(phase);
            forever begin
                seq_item = SPI_slave_seq_item::type_id::create("seq_item");
                seq_item_port.get_next_item(seq_item);
                SPI_slave_vif.rst_n = seq_item.rst_n;
                SPI_slave_vif.MOSI = seq_item.MOSI;
                SPI_slave_vif.SS_n = seq_item.SS_n;
                SPI_slave_vif.tx_valid = seq_item.tx_valid;
                SPI_slave_vif.tx_data = seq_item.tx_data;
                @(negedge SPI_slave_vif.clk);
                seq_item_port.item_done();
            end
        endtask
    endclass //className extends superClass
endpackage
```

SPI Slave Monitor

```
package SPI_slave_monitor_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import SPI_slave_seq_item_pkg::*;

    class SPI_slave_monitor extends uvm_monitor;
    `uvm_component_utils(SPI_slave_monitor)

        //virtual interface
        virtual SLAVE_interface SPI_slave_vif;

        //seq item
        SPI_slave_seq_item seq_item;

        //analysis port
        uvm_analysis_port #(SPI_slave_seq_item) mon_ap;

        //constructor
        function new(string name = "SPI_slave_monitor", uvm_component parent);
            super.new(name,parent);
        endfunction //new()

        //build
        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            mon_ap = new("mon_ap",this);
        endfunction

        //run
        task run_phase(uvm_phase phase);
            super.run_phase(phase);
            forever begin
                seq_item = SPI_slave_seq_item::type_id::create("seq_item");
                @(negedge SPI_slave_vif.clk);
                seq_item.rst_n = SPI_slave_vif.rst_n;
                seq_item.MOSI = SPI_slave_vif.MOSI;
                seq_item.SS_n = SPI_slave_vif.SS_n;
                seq_item.tx_data = SPI_slave_vif.tx_data;
                seq_item.tx_valid = SPI_slave_vif.tx_valid;
                seq_item.rx_valid = SPI_slave_vif.rx_valid;
                seq_item.rx_data = SPI_slave_vif.rx_data;
                seq_item.MISO = SPI_slave_vif.MISO;
                //golden outputs
```

```
                seq_item.rx_valid_golden = SPI_slave_vif.rx_valid_golden;
                seq_item.rx_data_golden = SPI_slave_vif.rx_data_golden;
                seq_item.MISO_golden = SPI_slave_vif.MISO_golden;
                //broadcast
                mon_ap.write(seq_item);
            end
        endtask

    endclass

endpackage
```

SPI Slave Agent

```
package SPI_slave_agent_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import SPI_slave_sqr_pkg::*;
import SPI_slave_driver_pkg::*;
import SPI_slave_monitor_pkg::*;
import SPI_slave_config_pkg::*;
import SPI_slave_seq_item_pkg::*;

    class SPI_slave_agent extends uvm_agent;
    `uvm_component_utils(SPI_slave_agent)

        //define handles
        SPI_slave_driver agent_driv;
        SPI_slave_sqr agent_sqr;
        SPI_slave_monitor agent_mon;
        SPI_slave_config agent_cfg;
        uvm_analysis_port #(SPI_slave_seq_item) agent_ap;

        //constructor
        function new(string name = "SPI_slave_agent", uvm_component parent =
null);
            super.new(name,parent);
        endfunction //new()

        //build phase
        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            //get config pointer to handler
            if(!(uvm_config_db
#(SPI_slave_config)::get(this,"","CFG_slave",agent_cfg))) begin
```

```
                `uvm_fatal("build_phase","can not get the CFG from DB in the
agent")
            end
            //build blocks
            if (agent_cfg.is_active == UVM_ACTIVE) begin
                agent_driv=SPI_slave_driver::type_id::create("agent_driv",this);
                agent_sqr=SPI_slave_sqr::type_id::create("agent_sqr",this);
            end
            agent_mon=SPI_slave_monitor::type_id::create("agent_mon",this);
            agent_ap=new("agent_ap",this);
        endfunction

        //connect phase
        function void connect_phase(uvm_phase phase);
            super.connect_phase(phase);
            if(agent_cfg.is_active == UVM_ACTIVE) begin
                agent_driv.seq_item_port.connect(agent_sqr.seq_item_export);
                agent_driv.SPI_slave_vif = agent_cfg.SPI_slave_vif;
            end
            agent_mon.SPI_slave_vif = agent_cfg.SPI_slave_vif;
            agent_mon.mon_ap.connect(agent_ap);
        endfunction

    endclass //className extends superClass

endpackage
```

SPI Slave Scoreboard

```
package SPI_slave_scoreboard_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import SPI_slave_seq_item_pkg::*;

    class SPI_slave_scoreboard extends uvm_scoreboard;
    `uvm_component_utils(SPI_slave_scoreboard)

        //counters
        int error_counter_slave = 0;
        int correct_counter_slave = 0;

        //seq item
        SPI_slave_seq_item seq_item;

        //ports
        uvm_analysis_export #(SPI_slave_seq_item) sb_export;
```

```
        uvm_tlm_analysis_fifo #(SPI_slave_seq_item) sb_fifo;

        //constructor
        function new(string name = "SPI_slave_scoreboard", uvm_component parent =
null);
            super.new(name,parent);
        endfunction //new()

        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            sb_export = new("sb_exprt",this);
            sb_fifo = new("sb_fifo",this);
        endfunction

        //connect
        function void connect_phase(uvm_phase phase);
            super.connect_phase(phase);
            sb_export.connect(sb_fifo.analysis_export);
        endfunction

        //run
        task run_phase(uvm_phase phase);
            super.run_phase(phase);
            forever begin
                sb_fifo.get(seq_item);
                if (seq_item.rx_data != seq_item.rx_data_golden ||
                    seq_item.rx_valid != seq_item.rx_valid_golden ||
                    seq_item.MISO != seq_item.MISO_golden ) begin
                    error_counter_slave++;
                end else begin
                    correct_counter_slave++;
                end
            end
        endtask
        //report
        function void report_phase(uvm_phase phase);
            super.report_phase(phase);
            `uvm_info("repo phase",$sformatf("Slave correct times:
%0d",correct_counter_slave),UVM_MEDIUM)
            `uvm_info("repo phase",$sformatf("Slave error times:
%0d",error_counter_slave),UVM_MEDIUM)
        endfunction
    endclass
endpackage
```

SPI Slave Coverage Collector

```
package SPI_slave_collector_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import SPI_slave_seq_item_pkg::*;
import shared_pkg::*;

    class SPI_slave_collector extends uvm_component;
    `uvm_component_utils(SPI_slave_collector)

        //seq item
        SPI_slave_seq_item seq_item_cvr;

        //ports
        uvm_analysis_export #(SPI_slave_seq_item) cvr_export;
        uvm_tlm_analysis_fifo #(SPI_slave_seq_item) cvr_fifo;

        //covergroups
        covergroup cvg_group;
            rx_data_CP: coverpoint seq_item_cvr.rx_data[9:8]
iff(seq_item_cvr.rst_n) {
                bins write_address = {2'b00};
                bins write_data    = {2'b01};
                bins read_address  = {2'b10};
                bins read_data     = {2'b11};
                bins trans1[]      = (2'b00 => 2'b01,2'b10);
                bins trans2[]      = (2'b01 => 2'b00,2'b11);
                bins trans3[]      = (2'b10 => 2'b00,2'b11);
                bins trans4[]      = (2'b11 => 2'b01,2'b10);
            }
            SS_n_CP: coverpoint seq_item_cvr.SS_n iff(seq_item_cvr.rst_n){
                bins full_normal_seq = (1 => 0[*13] => 1);
                bins full_read_seq = (1 => 0[*23] => 1);

                bins start_comm = (1 => 0[*4]);
            }
            MOSI_transitions_CP: coverpoint seq_item_cvr.MOSI iff((count <= 4) &&
(count >= 2)) {
                bins write_address = (0 => 0 => 0);
                bins write_data    = (0 => 0 => 1);
                bins read_address  = (1 => 1 => 0);
                bins read_data     = (1 => 1 => 1);
            }
```

```
            SS_n_with_MOSI: cross SS_n_CP,MOSI_transitions_CP
iff(seq_item_cvr.rst_n) {
                //as the full seq will be hit in the end of comm while the MISO
seq will be at the start of comm
                illegal_bins full_seq1 = binsof(SS_n_CP.full_normal_seq);
                illegal_bins full_seq2 = binsof(SS_n_CP.full_read_seq);
            }
        endgroup

        //constructor
        function new(string name = "SPI_slave_collector", uvm_component parent =
null);
            super.new(name,parent);
            cvg_group = new;
        endfunction //new()

        //build
        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            cvr_export = new("cvr_export",this);
            cvr_fifo = new ("cvr_fifo",this);
        endfunction

        //connect
        function void connect_phase(uvm_phase phase);
            super.connect_phase(phase);
            cvr_export.connect(cvr_fifo.analysis_export);
        endfunction

        //run
        task run_phase(uvm_phase phase);
            super.run_phase(phase);
            forever begin
                cvr_fifo.get(seq_item_cvr);
                cvg_group.sample();
            end
        endtask
    endclass //className extends superClass
endpackage
```

Crossing the full sequence bins of SS_n_CP with MOSI_transitions_CP would therefore be meaningless and temporally inconsistent, since the full sequences are detected at the end of communication, summarizing the entire SPI frame and the MOSI_transitions_CP transitions occur within the communication, while data bits are being transmitted.

SPI Slave Environment

```
package SPI_slave_env_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import SPI_slave_agent_pkg::*;
import SPI_slave_collector_pkg::*;
import SPI_slave_scoreboard_pkg::*;

    class SPI_slave_env extends uvm_env;
    `uvm_component_utils(SPI_slave_env)

        //hadles
        SPI_slave_agent ag;
        SPI_slave_collector cvr;
        SPI_slave_scoreboard sb;

        //constructor
        function new(string name = "SPI_slave_env", uvm_component parent = null);
            super.new(name,parent);
        endfunction

        //build
        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            ag = SPI_slave_agent::type_id::create("ag",this);
            cvr = SPI_slave_collector::type_id::create("cvr",this);
            sb = SPI_slave_scoreboard::type_id::create("sb",this);
        endfunction

        //connect
        function void connect_phase(uvm_phase phase);
            super.connect_phase(phase);
            ag.agent_ap.connect(cvr.cvr_export);
            ag.agent_ap.connect(sb.sb_export);
        endfunction
    endclass //className extends superClass
endpackage
```

SPI Slave Test

```
package SPI_slave_test_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import SPI_slave_env_pkg::*;
import SPI_slave_main_seq_pkg::*;
import SPI_slave_reset_seq_pkg::*;
import SPI_slave_config_pkg::*;

    class SPI_slave_test extends uvm_test;
    `uvm_component_utils(SPI_slave_test)

        //handles
        SPI_slave_env slave_env;
        SPI_slave_main_seq M_seq;
        SPI_slave_reset_seq R_seq;
        SPI_slave_config test_cfg;
        virtual SLAVE_interface SPI_slave_vif;

        //constructor
        function new(string name = "SPI_slave_test", uvm_component parent =
null);
             super.new(name,parent);
        endfunction

        //build
        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            slave_env = SPI_slave_env::type_id::create("slave_env",this);
            M_seq = SPI_slave_main_seq::type_id::create("M_seq");
            R_seq = SPI_slave_reset_seq::type_id::create("R_seq");
            test_cfg = SPI_slave_config::type_id::create("test_cfg");
            //get the virtual interface from db
            if(!(uvm_config_db #(virtual
SLAVE_interface)::get(this,"","SLAVE_IF",test_cfg.SPI_slave_vif))) begin
                `uvm_fatal("build phase","unable to get ALSU interface from DB in
test class");
            end
            //intialize is active var
            test_cfg.is_active = UVM_ACTIVE;
            //SET cfg to the db
            uvm_config_db
#(SPI_slave_config)::set(this,"slave_env*","CFG_slave",test_cfg);
        endfunction
```

```
        //run
        task run_phase(uvm_phase phase);
            super.run_phase(phase);
            phase.raise_objection(this);
            `uvm_info("run phase","reset asserted",UVM_LOW)
            R_seq.start(slave_env.ag.agent_sqr);
            `uvm_info("run phase","reset deasserted",UVM_LOW)
            `uvm_info("run phase","stimulas generation started of main
seq",UVM_LOW)
            M_seq.start(slave_env.ag.agent_sqr);
            `uvm_info("run phase","stimulas generation ended of main
seq",UVM_LOW)
            phase.drop_objection(this);
        endtask

    endclass

endpackage
```

SPI Slave Top Module

```
module SPI_slave_top();
import uvm_pkg::*;
`include "uvm_macros.svh"
import SPI_slave_test_pkg::*;
    //clock generation
    bit clk;
    always begin
        #10
        clk = ~clk;
    end
    //inst of if, design and golden module
     //if
     SLAVE_interface slave_if(clk);

     //design
     SLAVE DUT (
     .MOSI(slave_if.MOSI),
     .MISO(slave_if.MISO),
     .SS_n(slave_if.SS_n),
     .clk(clk),
     .rst_n(slave_if.rst_n),
     .rx_data(slave_if.rx_data),
```

```
     .rx_valid(slave_if.rx_valid),
     .tx_data(slave_if.tx_data),
     .tx_valid(slave_if.tx_valid)
     );

     //golden module
     SPI_slave_golden golden (
     .MOSI(slave_if.MOSI),
     .MISO(slave_if.MISO_golden),
     .SS_n(slave_if.SS_n),
     .clk(clk),
     .rst_n(slave_if.rst_n),
     .rx_data(slave_if.rx_data_golden),
     .rx_valid(slave_if.rx_valid_golden),
     .tx_data(slave_if.tx_data),
     .tx_valid(slave_if.tx_valid)
     );
     //virtual if to DB and run
     initial begin
        uvm_config_db #(virtual
SLAVE_interface)::set(null,"","SLAVE_IF",slave_if);
        run_test("SPI_slave_test");
     end
endmodule
```

SRC File

```
SPI_slav_interface.sv
+define+SIM
SPI_slave.sv
SPI_slave_golden.sv
shared_package.sv
SPI_slave_seq_item.sv
SPI_slave_R_seq.sv
SPI_slave_M_seq.sv
SPI_slave_sqr.sv
SPI_slave_config.sv
SPI_slave_driver.sv
SPI_slave_monitor.sv
SPI_slave_agent.sv
SPI_slave_scoreboard.sv
SPI_slave_collector.sv
SPI_slave_env.sv
SPI_slave_test.sv
SPI_slave_top.sv
```

Do File

```
vlib work
vlog -f src_files.list +cover -covercells
vsim -voptargs=+acc work.SPI_slave_top -classdebug -uvmcontrol=all  -cover
add wave /SPI_slave_top/slave_if/*
coverage save SLAVETB.ucdb -onexit
run -all
vcover report SLAVETB.ucdb -details -annotate -all -output coverage_rpt_slave.txt
```

- Excluding lines 37 and 38 as SS_n never goes high when we are in CHK_CMD so we don't enter the if statement of it.
- We add default case and Exclude it (line 127) as we don't enter the default case and we did that to avoid all false statement.

Coverage Report

```
Coverage Report by instance with details

================================================================================
Instance: /SPI_slave_top/slave_if
Design Unit: work.SLAVE_interface
================================================================================
Toggle Coverage:
   Enabled Coverage            Bins      Hits    Misses  Coverage
   ----------------            ----      ----    ------  --------
   Toggles                       74        74         0   100.00%

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

Toggle Coverage for instance /SPI_slave_top/slave_if --

                                    Node      1H->0L      0L->1H  "Coverage"
                                    ---------------------------------------
                                    MISO          1           1      100.00
                             MISO_golden          1           1      100.00
                                    MOSI          1           1      100.00
                                    SS_n          1           1      100.00
                                     clk          1           1      100.00
                                   rst_n          1           1      100.00
```

```
                    rx_data[9-0]          1          1      100.00
            rx_data_golden[9-0]          1          1      100.00
                    rx_valid          1          1      100.00
            rx_valid_golden          1          1      100.00
                    tx_data[7-0]          1          1      100.00
                    tx_valid          1          1      100.00

Total Node Count     =        37
Toggled Node Count   =        37
Untoggled Node Count =         0

Toggle Coverage      =     100.00% (74 of 74 bins)

================================================================================
Instance: /SPI_slave_top/DUT
Design Unit: work.SLAVE
================================================================================

Assertion Coverage:
   Assertions                  12       12        0   100.00%
--------------------------------------------------------------------
Name                File(Line)                Failure      Pass
                                               Count       Count
--------------------------------------------------------------------
/SPI_slave_top/DUT/assert__READ_DATA_to_IDLE
                SPI_slave.sv(262)              0        1
/SPI_slave_top/DUT/assert__READ_ADD_to_IDLE
                SPI_slave.sv(252)              0        1
/SPI_slave_top/DUT/assert__WRITE_to_IDLE
                SPI_slave.sv(242)              0        1
/SPI_slave_top/DUT/assert__CHK_CMD_to_READ_DATA
                SPI_slave.sv(232)              0        1
/SPI_slave_top/DUT/assert__CHK_CMD_to_READ_ADD
                SPI_slave.sv(224)              0        1
/SPI_slave_top/DUT/assert__CHK_CMD_to_WRITE
                SPI_slave.sv(216)              0        1
/SPI_slave_top/DUT/assert__IDLE_to_CHK_CMD
                SPI_slave.sv(206)              0        1
/SPI_slave_top/DUT/assert__read_data_comm
                SPI_slave.sv(193)              0        1
/SPI_slave_top/DUT/assert__read_address_comm
                SPI_slave.sv(185)              0        1
/SPI_slave_top/DUT/assert__write_data_comm
                SPI_slave.sv(177)              0        1
```

```
/SPI_slave_top/DUT/assert__write_address_comm
                SPI_slave.sv(169)              0        1
/SPI_slave_top/DUT/assert__sync_reset
                SPI_slave.sv(141)              0        1
Branch Coverage:
   Enabled Coverage            Bins      Hits    Misses  Coverage
   ----------------            ----      ----    ------  --------
   Branches                     38        38        0   100.00%

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

Branch Coverage for instance /SPI_slave_top/DUT

   Line        Item                    Count     Source
   ----        ----                    -----     ------
  File SPI_slave.sv
------------------------------------IF Branch------------------------------------
   20                              14087     Count coming in to IF
   20            1                   519        if (~rst_n) begin

   23            1                 13568        else begin

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------CASE Branch------------------------------------
   29                              37930     Count coming in to CASE
   30            1                  6897           IDLE : begin

   36            1                  5354           CHK_CMD : begin

   50            1                 13777            WRITE : begin

   56            1                  5102           READ_ADD : begin

   62            1                  6799           READ_DATA : begin

                                     1     All False Count
Branch totals: 6 hits of 6 branches = 100.00%

------------------------------------IF Branch------------------------------------
   31                               6897     Count coming in to IF
   31            1                  3085               if (SS_n)

   33            1                  3812               else
```

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

------------------------------------IF Branch------------------------------------
   37                              5354     Count coming in to IF
   39          1                    5354              else begin

Branch totals: 1 hit of 1 branch = 100.00%

------------------------------------IF Branch------------------------------------
   40                              5354     Count coming in to IF
   40          1                    2837                  if (~MOSI)

   42          1                    2517                  else begin

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
   43                              2517     Count coming in to IF
   43          1                    1328                      if (!received_address)

   45          1                    1189                      else

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
   51                             13777     Count coming in to IF
   51          1                    1858              if (SS_n)

   53          1                   11919               else

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
   57                              5102     Count coming in to IF
   57          1                     566              if (SS_n)

   59          1                    4536               else

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
   63                              6799     Count coming in to IF
   63          1                     661              if (SS_n)
```

```
   65          1                6138            else

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
   72                           50000    Count coming in to IF
   72          1                 520       if (~rst_n) begin

   78          1               49480       else begin

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------CASE Branch------------------------------------
   79                           49480    Count coming in to CASE
   80          1                3542            IDLE : begin

   83          1                3501            CHK_CMD : begin

   86          1               23303             WRITE : begin

   95          1                7202            READ_ADD : begin

   105          1               11932             READ_DATA : begin

Branch totals: 5 hits of 5 branches = 100.00%

------------------------------------IF Branch------------------------------------
   87                           23303    Count coming in to IF
   87          1               19608                if (counter > 0) begin

   91          1                3695                else begin

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
   96                            7202    Count coming in to IF
   96          1                6071                if (counter > 0) begin

   100          1                1131                else begin

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
```

```
   106                          11932    Count coming in to IF
   106          1                2893                if (tx_valid) begin

   116          1                9039                else begin

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
   108                           2893    Count coming in to IF
   108          1                2581                  if (counter > 0) begin

   112          1                 312                  else begin

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
   117                           9039    Count coming in to IF
   117          1                7338                  if (counter > 0 && ~rx_valid) begin

   121          1                1701                  else begin

Branch totals: 2 hits of 2 branches = 100.00%


Condition Coverage:
   Enabled Coverage            Bins   Covered    Misses  Coverage
   ----------------            ----      ----    ------  --------
   Conditions                     5         5         0   100.00%

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

Condition Coverage for instance /SPI_slave_top/DUT --

  File SPI_slave.sv
----------------Focused Condition View-------------------
Line       87 Item    1  (counter > 0)
Condition totals: 1 of 1 input term covered = 100.00%

    Input Term   Covered  Reason for no coverage   Hint
   -----------  --------  -----------------------  --------------
  (counter > 0)        Y

    Rows:       Hits  FEC Target           Non-masking condition(s)
 ---------  ---------  --------------------  -------------------------
```

```
  Row   1:         1  (counter > 0)_0      -
  Row   2:         1  (counter > 0)_1      -

----------------Focused Condition View-------------------
Line       96 Item    1  (counter > 0)
Condition totals: 1 of 1 input term covered = 100.00%

    Input Term   Covered  Reason for no coverage   Hint
   -----------  --------  -----------------------  --------------
  (counter > 0)        Y

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

----------------Focused Condition View-------------------
Line       108 Item    1  (counter > 0)
Condition totals: 1 of 1 input term covered = 100.00%

    Input Term   Covered  Reason for no coverage   Hint
   -----------  --------  -----------------------  --------------
  (counter > 0)        Y

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

----------------Focused Condition View-------------------
Line       117 Item    1  ((counter > 0) && ~rx_valid)
Condition totals: 2 of 2 input terms covered = 100.00%

    Input Term   Covered  Reason for no coverage   Hint
   -----------  --------  -----------------------  --------------
  (counter > 0)        Y
      rx_valid         Y

    Rows:       Hits  FEC Target           Non-masking condition(s)
 ---------  ---------  --------------------  -------------------------
  Row   1:         1  (counter > 0)_0      -
  Row   2:         1  (counter > 0)_1      ~rx_valid
  Row   3:         1  rx_valid_0          (counter > 0)
  Row   4:         1  rx_valid_1          (counter > 0)
```

```
Directive Coverage:
   Directives                 12       12        0   100.00%

DIRECTIVE COVERAGE:
---------------------------------------------------------------------------------------
Name                              Design Design   Lang File(Line)     Hits Status
                                  Unit   UnitType
---------------------------------------------------------------------------------------
/SPI_slave_top/DUT/cover__READ_DATA_to_IDLE
                                  SLAVE  Verilog  SVA  SPI_slave.sv(263)
                                                        648 Covered
/SPI_slave_top/DUT/cover__READ_ADD_to_IDLE
                                  SLAVE  Verilog  SVA  SPI_slave.sv(253)
                                                        559 Covered
/SPI_slave_top/DUT/cover__WRITE_to_IDLE  SLAVE  Verilog  SVA  SPI_slave.sv(243)
                                                       1818 Covered
/SPI_slave_top/DUT/cover__CHK_CMD_to_READ_DATA
                                  SLAVE  Verilog  SVA  SPI_slave.sv(233)
                                                        777 Covered
/SPI_slave_top/DUT/cover__CHK_CMD_to_READ_ADD
                                  SLAVE  Verilog  SVA  SPI_slave.sv(225)
                                                        637 Covered
/SPI_slave_top/DUT/cover__CHK_CMD_to_WRITE
                                  SLAVE  Verilog  SVA  SPI_slave.sv(217)
                                                       2054 Covered
/SPI_slave_top/DUT/cover__IDLE_to_CHK_CMD
                                  SLAVE  Verilog  SVA  SPI_slave.sv(207)
                                                       3501 Covered
/SPI_slave_top/DUT/cover__read_data_comm SLAVE  Verilog  SVA  SPI_slave.sv(194)
                                                        312 Covered
/SPI_slave_top/DUT/cover__read_address_comm
                                  SLAVE  Verilog  SVA  SPI_slave.sv(186)
                                                        771 Covered
/SPI_slave_top/DUT/cover__write_data_comm
                                  SLAVE  Verilog  SVA  SPI_slave.sv(178)
                                                        743 Covered
/SPI_slave_top/DUT/cover__write_address_comm
                                  SLAVE  Verilog  SVA  SPI_slave.sv(170)
                                                        801 Covered
/SPI_slave_top/DUT/cover__sync_reset    SLAVE  Verilog  SVA  SPI_slave.sv(142)
                                                        520 Covered
```

```
FSM Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    FSM States                     5         5        0   100.00%
    FSM Transitions                8         8        0   100.00%

================================FSM Details================================

FSM Coverage for instance /SPI_slave_top/DUT --

FSM_ID: cs
    Current State Object : cs
    ----------------------
    State Value MapInfo :
    ---------------------
Line        State Name            Value
----        ----------            -----
  30            IDLE                0
  36          CHK_CMD                2
  62         READ_DATA               4
  56          READ_ADD               3
  50            WRITE                1
    Covered States :
    ----------------
                State         Hit_count
                -----         ---------
                IDLE            3576
              CHK_CMD           3542
             READ_DATA          1560
              READ_ADD          1288
                WRITE           4121
    Covered Transitions :
    ---------------------
Line        Trans_ID        Hit_count        Transition
----        --------        ---------        ----------
  34            0             3542        IDLE -> CHK_CMD
  46            1              783        CHK_CMD -> READ_DATA
  44            2              651        CHK_CMD -> READ_ADD
  41            3             2067        CHK_CMD -> WRITE
  38            4               41        CHK_CMD -> IDLE
  64            5              782        READ_DATA -> IDLE
  58            6              651        READ_ADD -> IDLE
  52            7             2067        WRITE -> IDLE
```

```
   Summary                    Bins     Hits    Misses  Coverage
   -------                    ----     ----    ------  --------
      FSM States                5        5        0   100.00%
      FSM Transitions           8        8        0   100.00%
Statement Coverage:
   Enabled Coverage            Bins     Hits    Misses  Coverage
   ----------------            ----     ----    ------  --------
   Statements                  37       37        0   100.00%

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

Statement Coverage for instance /SPI_slave_top/DUT --

   Line        Item                    Count     Source
   ----        ----                    -----     ------
  File SPI_slave.sv
   1                                   module SLAVE
(MOSI,MISO,SS_n,clk,rst_n,rx_data,rx_valid,tx_data,tx_valid);

   2

   3                                   localparam IDLE      = 3'b000;

   4                                   localparam WRITE     = 3'b001;

   5                                   localparam CHK_CMD   = 3'b010;

   6                                   localparam READ_ADD  = 3'b011;

   7                                   localparam READ_DATA = 3'b100;

   8

   9                                   input MOSI, clk, rst_n, SS_n, tx_valid;

   10                                   input [7:0] tx_data;

   11                                   output reg [9:0] rx_data;

   12                                   output reg rx_valid, MISO;

   13
```

```
14                          reg [3:0] counter;
15                          reg received_address;
16
17                          reg [2:0] cs, ns;
18
19      1           14087   always @(posedge clk) begin
20                              if (~rst_n) begin
21      1             519           cs <= IDLE;
22                              end
23                              else begin
24      1           13568           cs <= ns;
25                              end
26                          end
27
28      1           37930   always @(*) begin
29                              case (cs)
30                                  IDLE : begin
31                                      if (SS_n)
32      1             3085                  ns = IDLE;
33                                      else
34      1             3812                  ns = CHK_CMD;
35                                  end
```

```
36                                  CHK_CMD : begin
37                                     if (SS_n)
38                                        ns = IDLE;
39                                     else begin
40                                        if (~MOSI)
41        1              2837                 ns = WRITE;
42                                        else begin
43                                           if (!received_address)
44        1              1328                    ns = READ_ADD;
45                                           else
46        1              1189                    ns = READ_DATA;
47                                        end
48                                     end
49                                  end
50                                  WRITE : begin
51                                     if (SS_n)
52        1              1858                ns = IDLE;
53                                     else
54        1              11919                ns = WRITE;
55                                  end
56                                  READ_ADD : begin
57                                     if (SS_n)
```

```
58      1          566          ns = IDLE;
59                        else
60      1          4536          ns = READ_ADD;
61                      end
62                      READ_DATA : begin
63                        if (SS_n)
64      1          661          ns = IDLE;
65                        else
66      1          6138          ns = READ_DATA;
67                      end
68                    endcase
69                  end
70
71      1          50000    always @(posedge clk) begin
72                    if (~rst_n) begin
73      1          520        rx_data <= 0;
74      1          520        rx_valid <= 0;
75      1          520        received_address <= 0;
76      1          520        MISO <= 0;
77                    end
78                    else begin
79                      case (cs)
```

```
80                                  IDLE : begin
81          1               3542            rx_valid <= 0;
82                                  end
83                                  CHK_CMD : begin
84          1               3501            counter <= 10;
85                                  end
86                                  WRITE : begin
87                                    if (counter > 0) begin
88          1              19608              rx_data[counter-1] <= MOSI;
89          1              19608              counter <= counter - 1;
90                                    end
91                                    else begin
92          1               3695              rx_valid <= 1;
93                                    end
94                                  end
95                                  READ_ADD : begin
96                                    if (counter > 0) begin
97          1               6071              rx_data[counter-1] <= MOSI;
98          1               6071              counter <= counter - 1;
99                                    end
100                                    else begin
101         1               1131              rx_valid <= 1;
```

```
102     1     1131          received_address <= 1;
103                       end
104                     end
105                     READ_DATA : begin
106                       if (tx_valid) begin
107     1     2893          rx_valid <= 0;
108                         if (counter > 0) begin
109     1     2581            MISO <= tx_data[counter-1];
110     1     2581            counter <= counter - 1;
111                         end
112                         else begin
113     1      312            received_address <= 0;
114                         end
115                       end
116                       else begin
117                         if (counter > 0 && ~rx_valid) begin
118     1     7338            rx_data[counter-1] <= MOSI;
119     1     7338            counter <= counter - 1;
120                         end
121                         else begin
122     1     1701            rx_valid <= 1;
123     1     1701            counter <= 8;
```

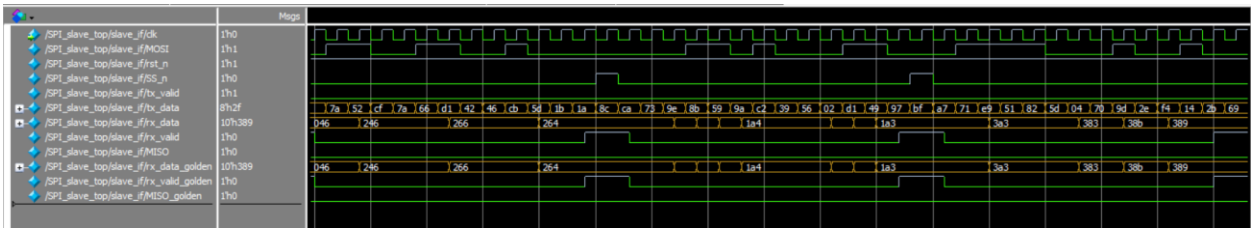
```
Toggle Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Toggles                         72        72         0   100.00%

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

Toggle Coverage for instance /SPI_slave_top/DUT --

                                  Node      1H->0L      0L->1H  "Coverage"
                                  ---------------------------------------
                                  MISO          1           1      100.00
                                  MOSI          1           1      100.00
                                  SS_n          1           1      100.00
                                   clk          1           1      100.00
                           counter[3-0]         1           1      100.00
                                cs[2-0]         1           1      100.00
                                ns[2-0]         1           1      100.00
                        received_address        1           1      100.00
                                 rst_n          1           1      100.00
                           rx_data[9-0]         1           1      100.00
                              rx_valid          1           1      100.00
                           tx_data[0-7]         1           1      100.00
                              tx_valid          1           1      100.00

Total Node Count     =        36
Toggled Node Count   =        36
Untoggled Node Count =         0

Toggle Coverage      =     100.00% (72 of 72 bins)

==============================================================================
```

Bug Report

- Change line 43 from if (received_address) to if (!received_address) to transition with the right flow.
- Change line 117 from if (counter > 0) to if (counter > 0 && ~rx_valid) to make sure we don't read as the ram would take some time to raise tx_valid.

QuestaSim Snippets

Waveform Snippet



Transcript Snippet

```
# ----------------------------------------------------------------
#
#   ************       IMPORTANT RELEASE NOTES         ************
#
#   You are using a version of the UVM library that has been compiled
#   with `UVM_NO_DEPRECATED undefined.
#   See http://www.eda.org/svdb/view.php?id=3313 for more details.
#
#   You are using a version of the UVM library that has been compiled
#   with `UVM_OBJECT_MUST_HAVE_CONSTRUCTOR undefined.
#   See http://www.eda.org/svdb/view.php?id=3770 for more details.
#
#       (Specify +UVM_NO_RELNOTES to turn off this notice)
#
# UVM_INFO verilog_src/questa_uvm_pkg-1.2/src/questa_uvm_pkg.sv(277) @ 0: reporter [Questa UVM] QUESTA_UVM-1.2.3
# UVM_INFO verilog_src/questa_uvm_pkg-1.2/src/questa_uvm_pkg.sv(278) @ 0: reporter [Questa UVM]  questa_uvm::init(all)
# UVM_INFO @ 0: reporter [RNTST] Running test SPI_slave_test...
# UVM_INFO SPI_slave_test.sv(45) @ 0: uvm_test_top [run phase] reset asserted
# ***********************************************************************
# * Questa UVM Transaction Recording Turned ON.                         *
# * recording_detail has been set.                                      *
# *  To turn off, set 'recording_detail' to off:                        *
# * uvm_config_db#(int)            ::set(null, "", "recording_detail", 0); *
# * uvm_config_db#(uvm_bitstream_t)::set(null, "", "recording_detail", 0); *
# ***********************************************************************
# UVM_INFO SPI_slave_test.sv(47) @ 20: uvm_test_top [run phase] reset deasserted
# UVM_INFO SPI_slave_test.sv(48) @ 20: uvm_test_top [run phase] stimulas generation started of main seq
# UVM_INFO SPI_slave_test.sv(50) @ 1000020: uvm_test_top [run phase] stimulas generation ended of main seq
# UVM_INFO verilog_src/uvm-1.1d/src/base/uvm_objection.svh(1267) @ 1000020: reporter [TEST_DONE] 'run' phase is ready to proceed to the 'extract' phase
# UVM_INFO SPI_slave_scoreboard.sv(55) @ 1000020: uvm_test_top.slave_env.sb [repo phase] Slave correct times: 50001
# UVM_INFO SPI_slave_scoreboard.sv(56) @ 1000020: uvm_test_top.slave_env.sb [repo phase] Slave error times: 0
#
# --- UVM Report Summary ---
#
# ** Report counts by severity
# UVM_INFO :   10
# UVM_WARNING :    0
# UVM_ERROR :    0
# UVM_FATAL :    0
# ** Report counts by id
# [Questa UVM]     2
# [RNTST]     1
# [TEST_DONE]     1
# [repo phase]     2
# [run phase]     4
# ** Note: $finish    : C:/questasim64_2021.1/win64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh(430)
#    Time: 1000020 ns  Iteration: 61  Instance: /SPI_slave_top
```

Code Coverage Snippets

Branch

Code Coverage Analysis

Branches - by instance (/SPI_slave_top/DUT)

```
20 if (~rst_n) begin
23 else begin
29 case (cs)
30 IDLE : begin
31 if (SS_n)
33 else
36 CHK_CMD : begin
37 if (SS_n)
39 else begin
40 if (~MOSI)
42 else begin
43 if (!received_address)
45 else
50 WRITE : begin
51 if (SS_n)
53 else
56 READ_ADD : begin
57 if (SS_n)
59 else
62 READ_DATA : begin
63 if (SS_n)
65 else
72 if (~rst_n) begin
78 else begin
80 IDLE : begin
83 CHK_CMD : begin
86 WRITE : begin
87 if (counter > 0) begin
91 else begin
95 READ_ADD : begin
96 if (counter > 0) begin
100 else begin
105 READ_DATA : begin
106 if (tx_valid) begin
108 if (counter > 0) begin
112 else begin
116 else begin
117 if (counter > 0 && ~rx_valid) begin
```

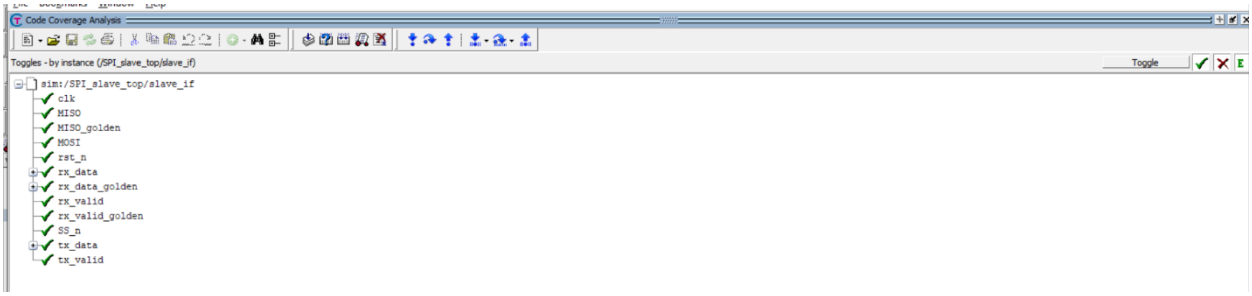
Statement

Code Coverage Analysis

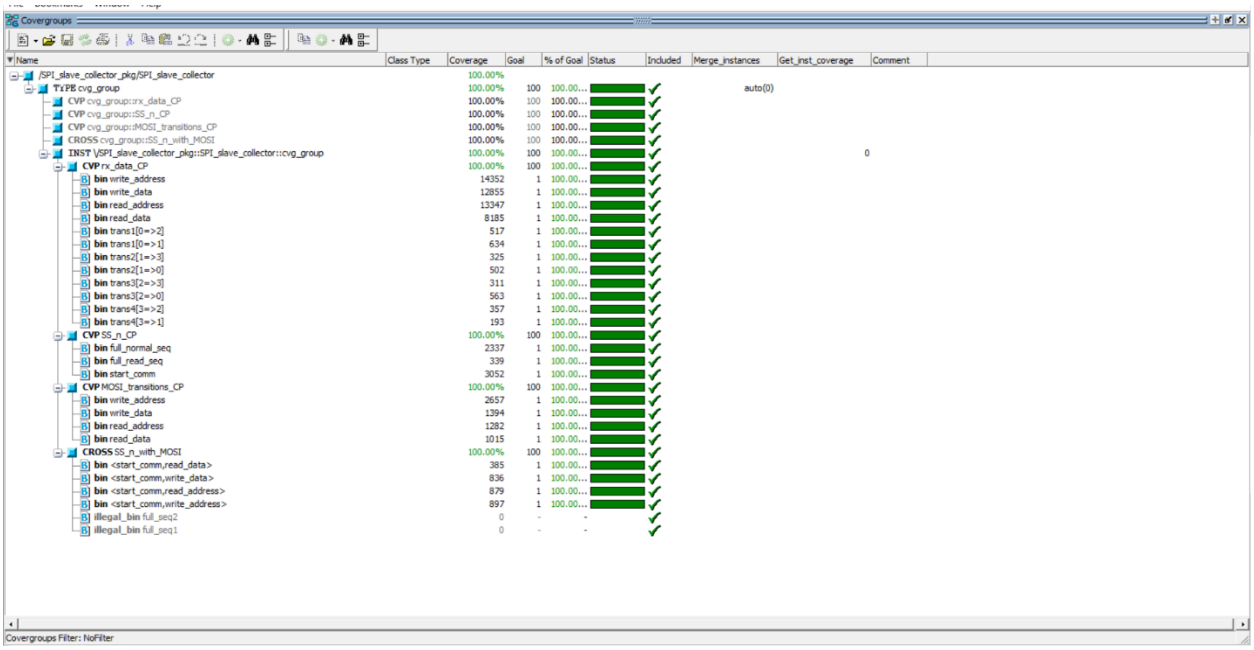
Statements - by instance (/SPI_slave_top/DUT)

```
21 cs <= IDLE;
24 cs <= ns;
28 always @(*) begin
32 ns = IDLE;
34 ns = CHK_CMD;
38 ns = IDLE;
41 ns = WRITE;
44 ns = READ_ADD;
46 ns = READ_DATA;
52 ns = IDLE;
54 ns = WRITE;
58 ns = IDLE;
60 ns = READ_ADD;
64 ns = IDLE;
66 ns = READ_DATA;
71 always @(posedge clk) begin
73 rx_data <= 0;
74 rx_valid <= 0;
75 received_address <= 0;
76 MISO <= 0;
81 rx_valid <= 0;
84 counter <= 10;
88 rx_data[counter-1] <= MOSI;
89 counter <= counter - 1;
92 rx_valid <= 1;
97 rx_data[counter-1] <= MOSI;
98 counter <= counter - 1;
101 rx_valid <= 1;
102 received_address <= 1;
107 rx_valid <= 0;
109 MISO <= tx_data[counter-1];
110 counter <= counter - 1;
113 received_address <= 0;
118 rx_data[counter-1] <= MOSI;
119 counter <= counter - 1;
122 rx_valid <= 1;
123 counter <= 0;
127 default: rx_valid <= 0;
```

Toggle

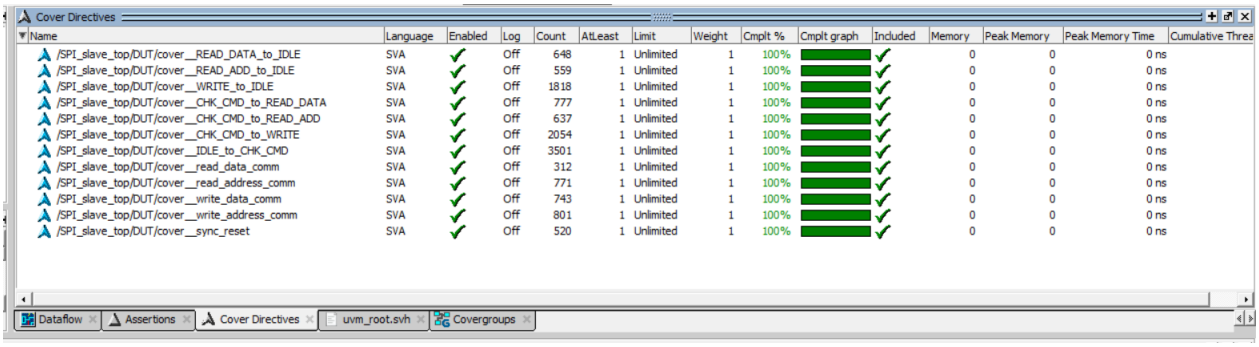


Functional Coverage Snippets

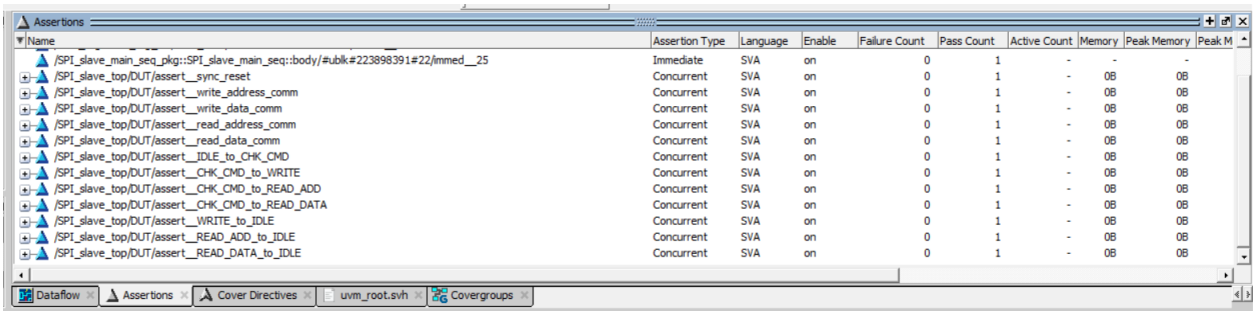


Sequential Domain Coverage (Assertion Coverage) Snippets

Cover directives



Assertions



Name	Assertion Type	Language	Enable	Failure Count	Pass Count	Active Count	Memory	Peak Memory	Peak M
/SPI_slave_main_seq_pkg::SPI_slave_main_seq::body/#ublk#223898391#22/immed__25	Immediate	SVA	on	0	1	-	-	-	
/SPI_slave_top/DUT/assert__sync_reset	Concurrent	SVA	on	0	1	-	0B	0B	
/SPI_slave_top/DUT/assert__write_address_comm	Concurrent	SVA	on	0	1	-	0B	0B	
/SPI_slave_top/DUT/assert__write_data_comm	Concurrent	SVA	on	0	1	-	0B	0B	
/SPI_slave_top/DUT/assert__read_address_comm	Concurrent	SVA	on	0	1	-	0B	0B	
/SPI_slave_top/DUT/assert__read_data_comm	Concurrent	SVA	on	0	1	-	0B	0B	
/SPI_slave_top/DUT/assert__IDLE_to_CHK_CMD	Concurrent	SVA	on	0	1	-	0B	0B	
/SPI_slave_top/DUT/assert__CHK_CMD_to_WRITE	Concurrent	SVA	on	0	1	-	0B	0B	
/SPI_slave_top/DUT/assert__CHK_CMD_to_READ_ADD	Concurrent	SVA	on	0	1	-	0B	0B	
/SPI_slave_top/DUT/assert__CHK_CMD_to_READ_DATA	Concurrent	SVA	on	0	1	-	0B	0B	
/SPI_slave_top/DUT/assert__WRITE_to_IDLE	Concurrent	SVA	on	0	1	-	0B	0B	
/SPI_slave_top/DUT/assert__READ_ADD_to_IDLE	Concurrent	SVA	on	0	1	-	0B	0B	
/SPI_slave_top/DUT/assert__READ_DATA_to_IDLE	Concurrent	SVA	on	0	1	-	0B	0B	

Assertions' Table

Feature	Assertion
Whenever rst_n is asserted, rx_data, rx_valid and MISO are low.	`@(posedge clk) !rst_n |=> ((rx_data == 0) && (rx_valid == 0) && (MISO == 0))`
Whenever write_add_seq (000) ((MOSI == 0) 3 times), then rx_valid and SS_n are high exactly after 10 clock cycles.	`@(posedge clk) disable iff(!rst_n) $fell(SS_n) ##1 (MOSI == 0)[*3] |-> ##10 rx_valid && SS_n;`
Whenever write_add_seq (001) ((MOSI == 0) 2 times then (MOSI == 1), then rx_valid and SS_n are high exactly after 10 clock cycles.	`@(posedge clk) disable iff(!rst_n) $fell(SS_n) ##1 (MOSI==0)[*2] ##1 (MOSI==1) |-> ##10 rx_valid && SS_n;`
Whenever write_add_seq (110) ((MOSI == 0) 3 times), then rx_valid and SS_n are high exactly after 10 clock cycles.	`@(posedge clk) disable iff(!rst_n) $fell(SS_n) ##1 (MOSI==1)[*2] ##1 (MOSI==0) |-> ##10 rx_valid && SS_n;`
Whenever write_add_seq (111) ((MOSI == 1) 3 times), then rx_valid and eventually SS_n are high exactly after 10 clock cycles.	`@(posedge clk) disable iff(!rst_n) $fell(SS_n) ##1 (MOSI==1)[*3] |-> ##10 rx_valid |-> ##10 SS_n;`
Whenever cs is IDLE then cs should be CHK_CMD next cycle.	`@(posedge clk) disable iff(!rst_n) (cs == IDLE) &&(!SS_n) |=> (cs == CHK_CMD);`

Whenever cs is CHK_CMD and MOSI is low then cs should be WRITE next cycle.	`@(posedge clk) disable iff(!rst_n) (cs == CHK_CMD) && (!SS_n) && (!MOSI) |=> (cs == WRITE);`
Whenever cs is CHK_CMD ,received_address is low and MOSI is high then cs should be READ_ADD next cycle.	`@(posedge clk) disable iff(!rst_n) (cs == CHK_CMD) && (!SS_n) && (MOSI) && (!received_address) |=> (cs == READ_ADD);`
Whenever cs is CHK_CMD ,received_address and MOSI are high then cs should be READ_ADD next cycle.	`@(posedge clk) disable iff(!rst_n) (cs == CHK_CMD) && (!SS_n) && (MOSI) && (received_address) |=> (cs == READ_DATA);`
Whenever cs is WRITE then cs should be IDLE next cycle.	`@(posedge clk) disable iff(!rst_n) (cs == WRITE) && (SS_n) |=> (cs == IDLE);`
Whenever cs is READ_ADD then cs should be IDLE next cycle.	`@(posedge clk) disable iff(!rst_n) (cs == READ_ADD) && (SS_n) |=> (cs == IDLE);`
Whenever cs is READ_DATA then cs should be IDLE next cycle.	`@(posedge clk) disable iff(!rst_n) (cs == READ_DATA) && (SS_n) |=> (cs == IDLE);`

Part 2: UVM Environment for Single-Port RAM

RAM Interface

```
interface RAM_interface (clk);

input clk;

logic [9:0] din;
logic rst_n, rx_valid;

logic [7:0] dout, dout_golden;
logic tx_valid, tx_valid_golden;

endinterface
```

RAM Assertions' File

```
module RAM_SVA(
    input [9:0] din,
    input clk,
    input rst_n,
    input rx_valid,
    input [7:0] dout,
    input tx_valid
);

property sync_reset;
    @(posedge clk)
    !rst_n |=> !tx_valid && (dout == 0);
endproperty

assert property (sync_reset);
cover property (sync_reset);

property tx_valid_off_seq_of_wrtie_add;
    @(posedge clk) disable iff(!rst_n)
    (din[9:8] == 2'b00) && (rx_valid == 1) |=> !tx_valid;
endproperty

assert property (tx_valid_off_seq_of_wrtie_add);
cover property (tx_valid_off_seq_of_wrtie_add);

property tx_valid_off_seq_of_wrtie_data;
    @(posedge clk) disable iff(!rst_n)
    (din[9:8] == 2'b01) && (rx_valid == 1) |=> !tx_valid;
endproperty

assert property (tx_valid_off_seq_of_wrtie_data);
cover property (tx_valid_off_seq_of_wrtie_data);

property tx_valid_off_seq_of_read_add;
    @(posedge clk) disable iff(!rst_n)
    (din[9:8] == 2'b10) && (rx_valid == 1) |=> !tx_valid;
endproperty

assert property (tx_valid_off_seq_of_read_add);
cover property (tx_valid_off_seq_of_read_add);
```

```
property tx_valid_on_seq;
    @(posedge clk) disable iff(!rst_n)
    (din[9:8] == 2'b11) && (rx_valid == 1) |=> tx_valid ##1 (!tx_valid)[->1];
endproperty

assert property (tx_valid_on_seq);
cover property (tx_valid_on_seq);

property write_data_eventually_after_address;
    @(posedge clk) disable iff(!rst_n)
    (din[9:8] == 2'b00) && (rx_valid == 1) |=> (din[9:8] == 2'b01)[->1];
endproperty

assert property (write_data_eventually_after_address);
cover property (write_data_eventually_after_address);

property read_data_eventually_after_address;
    @(posedge clk) disable iff(!rst_n)
    (din[9:8] == 2'b10) && (rx_valid == 1) |=> (din[9:8] == 2'b11)[->1];
endproperty

assert property (read_data_eventually_after_address);
cover property (read_data_eventually_after_address);

endmodule
```

RAM DUT

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

input [9:0] din;
input clk, rst_n, rx_valid;

output reg [7:0] dout;
output reg tx_valid;

reg [7:0] MEM [255:0];

reg [7:0] Rd_Addr, Wr_Addr;

always @(posedge clk) begin
    if (~rst_n) begin
        dout <= 0;
        tx_valid <= 0;
        Rd_Addr <= 0;
        Wr_Addr <= 0;
```

```
    end
    else begin
        if (rx_valid) begin
            case (din[9:8])
                2'b00 : Wr_Addr <= din[7:0];
                2'b01 : MEM[Wr_Addr] <= din[7:0];
                2'b10 : Rd_Addr <= din[7:0];
                2'b11 : dout <= MEM[Rd_Addr];
                default : dout <= 0;
            endcase
            tx_valid <= (din[9] && din[8])? 1'b1 : 1'b0;
        end
    end
end

endmodule
```

RAM Golden Model

```
module RAM_golden(clk,rst_n,rx_valid,din,tx_valid,dout);
parameter MEM_DEPTH = 256;
parameter ADDR_SIZE = 8;
input clk,rst_n,rx_valid;
input [9:0] din;
output reg tx_valid;
output reg [7:0] dout;
reg [ADDR_SIZE-1:0]ADD_read,ADD_write;
reg [7:0] mem [MEM_DEPTH-1:0];

always@(posedge clk)
begin
    if(!rst_n) begin
        tx_valid<=0;
        dout<=0;
        ADD_read <= 0;
        ADD_write <= 0;
    end
    else begin
        if(rx_valid) begin
            case (din[9:8])
                2'b00: begin
                    tx_valid<=0;
                    ADD_write<=din[7:0];
                end
```

```
                2'b01: begin
                    tx_valid<=0;
                    mem[ADD_write]<=din[7:0];
                end
                2'b10: begin
                    tx_valid<=0;
                    ADD_read<=din[7:0];
                end
                2'b11: begin
                    tx_valid<=1;
                    dout<=mem[ADD_read];
                end
            endcase
        end
    end
end
endmodule
```

RAM Shared Package

```
package shared_pkg;

    bit write_address_done;
    bit write_data_done;
    bit read_address_done;
    bit read_data_done;

endpackage
```

RAM Sequence Item

```
package RAM_seq_item_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import shared_pkg::*;


    class RAM_seq_item extends uvm_sequence_item;
    `uvm_object_utils(RAM_seq_item)

        //inputs and output
        rand logic rst_n, rx_valid;
        rand logic [9:0] din;
        logic [9:0] dout;
```

```
        logic tx_valid;
        logic [9:0] dout_golden;
        logic tx_valid_golden;

        //constructor
        function new (string name = "RAM_seq_item");
            super.new(name);
        endfunction

        //constraints

        //reset
        constraint reset_rate {
            rst_n dist {0:=1,1:=99};
        }

        constraint rx_rate {
            rx_valid dist {0:=5,1:=95};
        }

        constraint next_op {
            if (write_address_done) {
                soft din[9] == 1'b0;
            }
            else if (read_address_done) {
                soft din[9:8] == 2'b11;
            }
        }

        constraint next_read_data{
            if(read_data_done)
                din[9:8] == 2'b10;
        }


        constraint read_after_address {
            if(!read_address_done){
                din[9:8] != 2'b11;
            }
        }

        constraint after_write_data {
            if (write_data_done) {
                soft din[9:8] dist {2'b10 := 60 , 2'b00 := 40};
            }
```

```
            else if (read_data_done) {
                soft din[9:8] dist {2'b10 := 40 , 2'b00 := 60};
            }
        }

        function void post_randomize();

            if (din[9:8] == 2'b00) write_address_done = 1;
            else write_address_done = 0;

            if (din[9:8] == 2'b01) write_data_done = 1;
            else write_data_done = 0;


            if (din[9:8] == 2'b10) read_address_done = 1;
            else read_address_done = 0;

            if (din[9:8] == 2'b11) read_data_done = 1;
            else read_data_done = 0;

        endfunction

    endclass

endpackage
```

RAM Reset Sequence

```
package RAM_reset_seq_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import RAM_seq_item_pkg::*;

    class RAM_reset_seq extends uvm_sequence #(RAM_seq_item);
    `uvm_object_utils(RAM_reset_seq)

        //seq item
        RAM_seq_item seq_item;

        //construcotr
        function new(string name = "RAM_reset_seq");
            super.new(name);
        endfunction //new()

        //body
```

```
        task body();
            seq_item = RAM_seq_item::type_id::create("seq_item");
            start_item(seq_item);
            seq_item.rst_n = 0;
            seq_item.din = 5;
            seq_item.rx_valid = 0;
            finish_item(seq_item);
        endtask

    endclass //className extends superClass
endpackage
```

RAM Write Only Sequence

```
package RAM_write_only_seq_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import RAM_seq_item_pkg::*;

    class RAM_write_only_seq extends uvm_sequence #(RAM_seq_item);
    `uvm_object_utils(RAM_write_only_seq)

        //seq item
        RAM_seq_item seq_item;

        //construcotr
        function new(string name = "RAM_write_only_seq");
            super.new(name);
        endfunction //new()

        //body
        task body();
            seq_item = RAM_seq_item::type_id::create("seq_item");
            repeat(10000) begin
                start_item(seq_item);
                assert(seq_item.randomize() with {din[9] == 1'b0;});
                finish_item(seq_item);
            end
        endtask

    endclass //className extends superClass
endpackage
```

RAM Read Only Sequence

```
package RAM_read_only_seq_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import RAM_seq_item_pkg::*;

    class RAM_read_only_seq extends uvm_sequence #(RAM_seq_item);
    `uvm_object_utils(RAM_read_only_seq)

        //seq item
        RAM_seq_item seq_item;

        //construcotr
        function new(string name = "RAM_read_only_seq");
            super.new(name);
        endfunction //new()

        //body
        task body();
            seq_item = RAM_seq_item::type_id::create("seq_item");
            repeat(10000) begin
                start_item(seq_item);
                assert(seq_item.randomize() with {din[9] == 1'b1;});
                finish_item(seq_item);
            end
        endtask

    endclass //className extends superClass
endpackage
```

RAM Write Read Sequence

```
package RAM_write_read_seq_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import RAM_seq_item_pkg::*;

    class RAM_write_read_seq extends uvm_sequence #(RAM_seq_item);
    `uvm_object_utils(RAM_write_read_seq)

        //seq item
        RAM_seq_item seq_item;
```

```
        //construcotr
        function new(string name = "RAM_write_read_seq");
            super.new(name);
        endfunction //new()

        //body
        task body();
            seq_item = RAM_seq_item::type_id::create("seq_item");
            seq_item.next_read_data.constraint_mode(0);
            repeat(10000) begin
                start_item(seq_item);
                assert(seq_item.randomize());
                finish_item(seq_item);
            end
        endtask

    endclass //className extends superclass

endpackage
```

RAM Sequencer

```
package RAM_sqr_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import RAM_seq_item_pkg::*;

    class RAM_sqr extends uvm_sequencer #(RAM_seq_item);
    `uvm_component_utils(RAM_sqr)

        //constructor
        function new(string name = "RAM_sqr", uvm_component parent = null);
            super.new(name,parent);
        endfunction //new()


    endclass //className extends superClass

endpackage
```

RAM Configuration Object

```
package RAM_config_pkg;

import uvm_pkg::*;
`include "uvm_macros.svh"
import uvm_pkg::*;

    class RAM_config extends uvm_object;
    `uvm_object_utils(RAM_config)

        virtual RAM_interface RAM_vif;
        uvm_active_passive_enum is_active;

        //constructor
        function new(string name = "RAM_config");
            super.new(name);
        endfunction //new()


    endclass //className extends superClass

endpackage
```

RAM Driver

```
package RAM_driver_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import RAM_seq_item_pkg::*;
import shared_pkg::*;

    class RAM_driver extends uvm_driver #(RAM_seq_item);
    `uvm_component_utils (RAM_driver)

        //virtual interface
        virtual RAM_interface RAM_vif;

        //seq item
        RAM_seq_item seq_item;

        //constructor
        function new(string name = "RAM_driver", uvm_component parent = null);
            super.new(name,parent);
```

```
        endfunction //new()

        //build phase
        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
        endfunction

        //run
        task run_phase(uvm_phase phase);
            super.run_phase(phase);
            forever begin
                seq_item = RAM_seq_item::type_id::create("seq_item");
                seq_item_port.get_next_item(seq_item);
                RAM_vif.rst_n = seq_item.rst_n;
                RAM_vif.din = seq_item.din;
                RAM_vif.rx_valid = seq_item.rx_valid;
                @(negedge RAM_vif.clk);
                seq_item_port.item_done();
            end
        endtask
    endclass //className extends superClass

endpackage
```

RAM Monitor

```
package RAM_monitor_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import RAM_seq_item_pkg::*;

    class RAM_monitor extends uvm_monitor;
    `uvm_component_utils(RAM_monitor)

        //virtual interface
        virtual RAM_interface RAM_vif;

        //seq item
        RAM_seq_item seq_item;

        //analysis port
        uvm_analysis_port #(RAM_seq_item) mon_ap;

        //constructor
        function new(string name = "RAM_monitor", uvm_component parent);
            super.new(name,parent);
```

```
        endfunction //new()

        //build
        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            mon_ap = new("mon_ap",this);
        endfunction

        //run
        task run_phase(uvm_phase phase);
            super.run_phase(phase);
            forever begin
                seq_item = RAM_seq_item::type_id::create("seq_item");
                @(negedge RAM_vif.clk);
                seq_item.rst_n = RAM_vif.rst_n;
                seq_item.rx_valid = RAM_vif.rx_valid;
                seq_item.din = RAM_vif.din;
                seq_item.dout = RAM_vif.dout;
                seq_item.tx_valid = RAM_vif.tx_valid;
                //golden outputs
                seq_item.tx_valid_golden = RAM_vif.tx_valid_golden;
                seq_item.dout_golden = RAM_vif.dout_golden;
                //broadcast
                mon_ap.write(seq_item);
            end
        endtask

    endclass

endpackage
```

RAM Agent

```
package RAM_agent_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import RAM_sqr_pkg::*;
import RAM_driver_pkg::*;
import RAM_monitor_pkg::*;
import RAM_config_pkg::*;
import RAM_seq_item_pkg::*;

    class RAM_agent extends uvm_agent;
    `uvm_component_utils(RAM_agent)

        //define handles
```

```
        RAM_driver agent_driv;
        RAM_sqr agent_sqr;
        RAM_monitor agent_mon;
        RAM_config agent_cfg;
        uvm_analysis_port #(RAM_seq_item) agent_ap;

        //constructor
        function new(string name = "RAM_agent", uvm_component parent = null);
            super.new(name,parent);
        endfunction //new()

        //build phase
        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            //get config pointer to handler
            if(!(uvm_config_db #(RAM_config)::get(this,"","CFG_RAM",agent_cfg)))
begin
               `uvm_fatal("build_phase","can not get the CFG from DB in the
agent")
            end
            //build blocks
            if (agent_cfg.is_active == UVM_ACTIVE) begin
                agent_driv=RAM_driver::type_id::create("agent_driv",this);
                agent_sqr=RAM_sqr::type_id::create("agent_sqr",this);
            end
            agent_mon=RAM_monitor::type_id::create("agent_mon",this);
            agent_ap=new("agent_ap",this);
        endfunction

        //connect phase
        function void connect_phase(uvm_phase phase);
            super.connect_phase(phase);
            if(agent_cfg.is_active == UVM_ACTIVE) begin
                agent_driv.seq_item_port.connect(agent_sqr.seq_item_export);
                agent_driv.RAM_vif = agent_cfg.RAM_vif;
            end
            agent_mon.RAM_vif = agent_cfg.RAM_vif;
            agent_mon.mon_ap.connect(agent_ap);
        endfunction

    endclass //className extends superClass


endpackage
```

RAM Scoreboard

```
package RAM_scoreboard_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import RAM_seq_item_pkg::*;

    class RAM_scoreboard extends uvm_scoreboard;
    `uvm_component_utils(RAM_scoreboard)

        //counters
        int error_counter_ram = 0;
        int correct_counter_ram = 0;

        //seq item
        RAM_seq_item seq_item;

        //ports
        uvm_analysis_export #(RAM_seq_item) sb_export;
        uvm_tlm_analysis_fifo #(RAM_seq_item) sb_fifo;

        //constructor
        function new(string name = "RAM_scoreboard", uvm_component parent =
null);
            super.new(name,parent);
        endfunction //new()

        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            sb_export = new("sb_exprt",this);
            sb_fifo = new("sb_fifo",this);
        endfunction

        //connect
        function void connect_phase(uvm_phase phase);
            super.connect_phase(phase);
            sb_export.connect(sb_fifo.analysis_export);
        endfunction

        //run
        task run_phase(uvm_phase phase);
            super.run_phase(phase);
            forever begin
                sb_fifo.get(seq_item);
                if (seq_item.dout !== seq_item.dout_golden ||
```

```
                    seq_item.tx_valid !== seq_item.tx_valid_golden) begin
                    error_counter_ram++;
                end else begin
                    correct_counter_ram++;
                end
            end
        endtask

        //report
        function void report_phase(uvm_phase phase);
            super.report_phase(phase);
            `uvm_info("repo phase",$sformatf("RAM correct times:
%0d",correct_counter_ram),UVM_MEDIUM)
            `uvm_info("repo phase",$sformatf("RAM error times:
%0d",error_counter_ram),UVM_MEDIUM)
        endfunction
    endclass
endpackage
```

RAM Coverage Collector

```
package RAM_collector_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import RAM_seq_item_pkg::*;
import shared_pkg::*;

    class RAM_collector extends uvm_component;
    `uvm_component_utils(RAM_collector)

        //seq item
        RAM_seq_item seq_item_cvr;

        //ports
        uvm_analysis_export #(RAM_seq_item) cvr_export;
        uvm_tlm_analysis_fifo #(RAM_seq_item) cvr_fifo;

        //covergroups
        covergroup RAM_cvr;
            din_cp: coverpoint seq_item_cvr.din[9:8] iff(seq_item_cvr.rst_n) {
                bins write_address = {2'b00};
                bins write_data    = {2'b01};
                bins read_address  = {2'b10};
                bins read_data     = {2'b11};
                bins write_data_after_write_address = (2'b00 => 2'b01);
                bins read_data_after_read_address = (2'b10 => 2'b11);
```

```
                bins full_trans = (2'b00 => 2'b01 => 2'b10 => 2'b11);
            }
            rx_valid_CP: coverpoint seq_item_cvr.rx_valid iff(seq_item_cvr.rst_n)
{
                bins high = {1};
                bins low = {0};
            }
            tx_valid_CP: coverpoint seq_item_cvr.tx_valid iff(seq_item_cvr.rst_n)
{
                bins high = {1};
                bins low = {0};
            }

            din_with_rx: cross din_cp,rx_valid_CP {
                ignore_bins low_tx = binsof(rx_valid_CP.low);
            }

            din_read_with_tx: cross din_cp,tx_valid_CP {
                option.cross_auto_bin_max = 0;
                bins checked = binsof(din_cp.read_data) &&
binsof(tx_valid_CP.high);
            }
        endgroup

        //constructor
        function new(string name = "RAM_collector", uvm_component parent = null);
            super.new(name,parent);
            RAM_cvr = new;
        endfunction //new()

        //build
        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            cvr_export = new("cvr_export",this);
            cvr_fifo = new ("cvr_fifo",this);
        endfunction

        //connect
        function void connect_phase(uvm_phase phase);
            super.connect_phase(phase);
            cvr_export.connect(cvr_fifo.analysis_export);
        endfunction

        //run
        task run_phase(uvm_phase phase);
```

```
            super.run_phase(phase);
            forever begin
                cvr_fifo.get(seq_item_cvr);
                RAM_cvr.sample();
            end
        endtask
    endclass //className extends superClass
endpackage
```

- The case where rx_valid = 0 is ignored because, in this state, the RAM does not sample or process din. Only when rx_valid is high does a valid transaction occur, so ignoring the low case ensures coverage reflects active operations only.

RAM Environment

```
package RAM_env_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import RAM_agent_pkg::*;
import RAM_collector_pkg::*;
import RAM_scoreboard_pkg::*;

    class RAM_env extends uvm_env;
    `uvm_component_utils(RAM_env)

        //hadles
        RAM_agent ag;
        RAM_collector cvr;
        RAM_scoreboard sb;

        //constructor
        function new(string name = "RAM_env", uvm_component parent = null);
            super.new(name,parent);
        endfunction

        //build
        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            ag = RAM_agent::type_id::create("ag",this);
            cvr = RAM_collector::type_id::create("cvr",this);
            sb = RAM_scoreboard::type_id::create("sb",this);
        endfunction

        //connect
```

```
        function void connect_phase(uvm_phase phase);
            super.connect_phase(phase);
            ag.agent_ap.connect(cvr.cvr_export);
            ag.agent_ap.connect(sb.sb_export);
        endfunction

    endclass //className extends superClass

endpackage
```

RAM Test

```
package RAM_test_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"
import RAM_env_pkg::*;
import RAM_reset_seq_pkg::*;
import RAM_write_only_seq_pkg::*;
import RAM_read_only_seq_pkg::*;
import RAM_write_read_seq_pkg::*;
import RAM_config_pkg::*;

    class RAM_test extends uvm_test;
    `uvm_component_utils(RAM_test)

        //handles
        RAM_env RAM_ENV;
        RAM_reset_seq R_seq;
        RAM_write_only_seq Wr_seq;
        RAM_read_only_seq Rd_seq;
        RAM_write_read_seq WR_seq;
        RAM_config test_cfg;
        virtual RAM_interface RAM_vif;

        //constructor
        function new(string name = "RAM_test", uvm_component parent = null);
             super.new(name,parent);
        endfunction

        //build
        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            RAM_ENV = RAM_env::type_id::create("RAM_env",this);
            R_seq = RAM_reset_seq::type_id::create("R_seq");
            Wr_seq = RAM_write_only_seq::type_id::create("Wr_seq");
            Rd_seq = RAM_read_only_seq::type_id::create("Rd_seq");
```

```
            WR_seq = RAM_write_read_seq::type_id::create("WR_seq");
            test_cfg = RAM_config::type_id::create("test_cfg");
            //get the virtual interface from db
            if(!(uvm_config_db #(virtual
RAM_interface)::get(this,"","RAM_IF",test_cfg.RAM_vif))) begin
                `uvm_fatal("build phase","unable to get ALSU interface from DB in
test class");
            end
            //intialize is active var
            test_cfg.is_active = UVM_ACTIVE;
            //SET cfg to the db
            uvm_config_db #(RAM_config)::set(this,"RAM_env*","CFG_RAM",test_cfg);
        endfunction

        //run
        task run_phase(uvm_phase phase);
            super.run_phase(phase);
            phase.raise_objection(this);
            `uvm_info("run phase","reset asserted",UVM_LOW)
            R_seq.start(RAM_ENV.ag.agent_sqr);
            `uvm_info("run phase","reset deasserted",UVM_LOW)
            `uvm_info("run phase","stimulas generation started of write only
seq",UVM_LOW)
            Wr_seq.start(RAM_ENV.ag.agent_sqr);
            `uvm_info("run phase","stimulas generation ended of write only
seq",UVM_LOW)
            `uvm_info("run phase","stimulas generation started of read only
seq",UVM_LOW)
            Rd_seq.start(RAM_ENV.ag.agent_sqr);
            `uvm_info("run phase","stimulas generation ended of read only
seq",UVM_LOW)
            `uvm_info("run phase","stimulas generation started of read-write
seq",UVM_LOW)
            WR_seq.start(RAM_ENV.ag.agent_sqr);
            `uvm_info("run phase","stimulas generation ended of read-write
seq",UVM_LOW)
            phase.drop_objection(this);
        endtask

    endclass
endpackage
```

RAM Top Module

```
module RAM_top();
import uvm_pkg::*;
`include "uvm_macros.svh"
import RAM_test_pkg::*;

    //clock generation
    bit clk;
    always begin
        #10
        clk = ~clk;
    end

    //inst of if, design and golden module
     //if
     RAM_interface RAM_if(clk);

     //design
     RAM DUT (
     .din(RAM_if.din),
     .clk(clk),
     .rst_n(RAM_if.rst_n),
     .rx_valid(RAM_if.rx_valid),
     .dout(RAM_if.dout),
     .tx_valid(RAM_if.tx_valid)
     );

     //golden module
     RAM_golden golden (
     .din(RAM_if.din),
     .clk(clk),
     .rst_n(RAM_if.rst_n),
     .rx_valid(RAM_if.rx_valid),
     .dout(RAM_if.dout_golden),
     .tx_valid(RAM_if.tx_valid_golden)
     );

     //virtual if to DB and run
     initial begin
        uvm_config_db #(virtual RAM_interface)::set(null,"","RAM_IF",RAM_if);
        run_test("RAM_test");
     end
    //bind
    bind RAM RAM_SVA assertion_mod (
```

```
     .din(RAM_if.din),
     .clk(clk),
     .rst_n(RAM_if.rst_n),
     .rx_valid(RAM_if.rx_valid),
     .dout(RAM_if.dout),
     .tx_valid(RAM_if.tx_valid)
    );
endmodule
```

SRC file

```
RAM_interface.sv
RAM_assertions.sv
RAM.v
RAM_golden.sv
shared_package.sv
RAM_seq_item.sv
RAM_R_seq.sv
RAM_WO_seq.sv
RAM_RO_seq.sv
RAM_WR_seq.sv
RAM_sqr.sv
RAM_config.sv
RAM_driver.sv
RAM_monitor.sv
RAM_agent.sv
RAM_scorboard.sv
RAM_collector.sv
RAM_env.sv
RAM_test.sv
RAM_top.sv
```

Do File

```
vlib work
vlog -f src_files.list +cover -covercells
vsim -voptargs=+acc work.RAM_top -classdebug -uvmcontrol=all  -cover
add wave /RAM_top/RAM_if/*
coverage exclude -src RAM.v -line 27
coverage save RAMTB.ucdb -onexit
run -all
vcover report RAMTB.ucdb -details -annotate -all -output coverage_rpt_RAM.txt
```

- We exclude line 27 as there is a default case we won't enter.

Coverage Report

```
Coverage Report by instance with details

================================================================================
Instance: /RAM_top/RAM_if
Design Unit: work.RAM_interface
================================================================================
Toggle Coverage:
   Enabled Coverage            Bins      Hits    Misses  Coverage
   ----------------            ----      ----    ------  --------
   Toggles                       62        62         0   100.00%

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

Toggle Coverage for instance /RAM_top/RAM_if --

                                  Node      1H->0L      0L->1H  "Coverage"
                                  ----------------------------------------
                                   clk          1           1      100.00
                                din[9-0]        1           1      100.00
                                dout[7-0]       1           1      100.00
                           dout_golden[7-0]     1           1      100.00
                                 rst_n          1           1      100.00
                                rx_valid        1           1      100.00
                                tx_valid        1           1      100.00
                           tx_valid_golden      1           1      100.00

Total Node Count     =        31
Toggled Node Count   =        31
Untoggled Node Count =         0

Toggle Coverage      =     100.00% (62 of 62 bins)

================================================================================
Instance: /RAM_top/DUT/assertion_mod
Design Unit: work.RAM_SVA
================================================================================

Assertion Coverage:
   Assertions                    7         7         0   100.00%
--------------------------------------------------------------------
Name                File(Line)                Failure      Pass
                                              Count        Count
```

```
------------------------------------------------------------------
/RAM_top/DUT/assertion_mod/assert__read_data_eventually_after_address
                RAM_assertions.sv(63)           0        1
/RAM_top/DUT/assertion_mod/assert__write_data_eventually_after_address
                RAM_assertions.sv(55)           0        1
/RAM_top/DUT/assertion_mod/assert__tx_valid_on_seq
                RAM_assertions.sv(47)           0        1
/RAM_top/DUT/assertion_mod/assert__tx_valid_off_seq_of_read_add
                RAM_assertions.sv(39)           0        1
/RAM_top/DUT/assertion_mod/assert__tx_valid_off_seq_of_wrtie_data
                RAM_assertions.sv(31)           0        1
/RAM_top/DUT/assertion_mod/assert__tx_valid_off_seq_of_wrtie_add
                RAM_assertions.sv(23)           0        1
/RAM_top/DUT/assertion_mod/assert__sync_reset
                RAM_assertions.sv(15)           0        1

Directive Coverage:
   Directives                    7        7        0   100.00%

DIRECTIVE COVERAGE:
--------------------------------------------------------------------------------------------
Name                              Design Design   Lang File(Line)      Hits Status
                              Unit   UnitType
--------------------------------------------------------------------------------------------
/RAM_top/DUT/assertion_mod/cover__read_data_eventually_after_address
                              RAM_SVA Verilog  SVA  RAM_assertions.sv(64)
                                                          9106 Covered
/RAM_top/DUT/assertion_mod/cover__write_data_eventually_after_address
                              RAM_SVA Verilog  SVA  RAM_assertions.sv(56)
                                                          9125 Covered
/RAM_top/DUT/assertion_mod/cover__tx_valid_on_seq
                              RAM_SVA Verilog  SVA  RAM_assertions.sv(48)
                                                          4613 Covered
/RAM_top/DUT/assertion_mod/cover__tx_valid_off_seq_of_read_add
                              RAM_SVA Verilog  SVA  RAM_assertions.sv(40)
                                                          9200 Covered
/RAM_top/DUT/assertion_mod/cover__tx_valid_off_seq_of_wrtie_data
                              RAM_SVA Verilog  SVA  RAM_assertions.sv(32)
                                                          4741 Covered
/RAM_top/DUT/assertion_mod/cover__tx_valid_off_seq_of_wrtie_add
                              RAM_SVA Verilog  SVA  RAM_assertions.sv(24)
                                                          9230 Covered
/RAM_top/DUT/assertion_mod/cover__sync_reset
                              RAM_SVA Verilog  SVA  RAM_assertions.sv(16)
```

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

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

Toggle Coverage for instance /RAM_top/DUT/assertion_mod --

                                  Node      1H->0L      0L->1H  "Coverage"
                                  ---------------------------------------
                                   clk          1           1      100.00
                                din[0-9]          1           1      100.00
                               dout[0-7]          1           1      100.00
                                  rst_n          1           1      100.00
                                rx_valid         1           1      100.00
                                tx_valid         1           1      100.00

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

Toggle Coverage      =     100.00% (44 of 44 bins)

===============================================================================
Instance: /RAM_top/DUT
Design Unit: work.RAM
===============================================================================
Branch Coverage:
   Enabled Coverage            Bins      Hits    Misses  Coverage
   ----------------            ----      ----    ------  --------
   Branches                       8         8         0   100.00%

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

Branch Coverage for instance /RAM_top/DUT

   Line        Item                    Count     Source
   ----        ----                    -----     ------
  File RAM.v
------------------------------------IF Branch------------------------------------
   14                                 30001     Count coming in to IF
   14            1                      329        if (~rst_n) begin
```

```
   20          1                29672       else begin

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
   21                           29672    Count coming in to IF
   21          1                28165          if (rx_valid) begin

                                1507    All False Count
Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------CASE Branch------------------------------------
   22                           28165    Count coming in to CASE
   23          1                 9321                2'b00 : Wr_Addr <= din[7:0];

   24          1                 4797                2'b01 : MEM[Wr_Addr] <= din[7:0];

   25          1                 9314                2'b10 : Rd_Addr <= din[7:0];

   26          1                 4733                2'b11 : dout <= MEM[Rd_Addr];

Branch totals: 4 hits of 4 branches = 100.00%


Expression Coverage:
   Enabled Coverage            Bins   Covered    Misses  Coverage
   ----------------            ----     ----    ------  --------
   Expressions                  3        3        0   100.00%

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

Expression Coverage for instance /RAM_top/DUT --

  File RAM.v
----------------Focused Expression View-----------------
Line       30 Item    1  ((din[9] && din[8]) && rx_valid)
Expression totals: 3 of 3 input terms covered = 100.00%

  Input Term   Covered  Reason for no coverage   Hint
 -----------  --------  -----------------------  --------------
     din[9]        Y
     din[8]        Y
   rx_valid         Y
```

```
    Rows:       Hits  FEC Target           Non-masking condition(s)
 ---------  ---------  --------------------  -------------------------
  Row   1:         1  din[9]_0             -
  Row   2:         1  din[9]_1             (rx_valid && din[8])
  Row   3:         1  din[8]_0             din[9]
  Row   4:         1  din[8]_1             (rx_valid && din[9])
  Row   5:         1  rx_valid_0            (din[9] && din[8])
  Row   6:         1  rx_valid_1            (din[9] && din[8])


Statement Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------             ----      ----    ------  --------
    Statements                     10        10         0   100.00%

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

Statement Coverage for instance /RAM_top/DUT --

    Line        Item                    Count     Source
    ----        ----                   -----     ------
  File RAM.v
    1                                   module RAM (din,clk,rst_n,rx_valid,dout,tx_valid);

    2

    3                                   input [9:0] din;

    4                                   input clk, rst_n, rx_valid;

    5

    6                                   output reg [7:0] dout;

    7                                   output reg tx_valid;

    8

    9                                   reg [7:0] MEM [255:0];

    10

    11                                   reg [7:0] Rd_Addr, Wr_Addr;
```

```
  12

  13        1              30001    always @(posedge clk) begin

  14                                if (~rst_n) begin

  15        1               329          dout <= 0;

  16        1               329          tx_valid <= 0;

  17        1               329          Rd_Addr <= 0;

  18        1               329          Wr_Addr <= 0;

  19                                end

  20                                else begin

  21                                   if (rx_valid) begin

  22                                      case (din[9:8])

  23        1              9321                2'b00 : Wr_Addr <= din[7:0];

  24        1              4797                2'b01 : MEM[Wr_Addr] <= din[7:0];

  25        1              9314                2'b10 : Rd_Addr <= din[7:0];

  26        1              4733                2'b11 : dout <= MEM[Rd_Addr];

  27                                         default : dout <= 0;

  28                                      endcase

  29                                   end

  30        1              29672          tx_valid <= (din[9] && din[8] && rx_valid)? 1'b1 :
1'b0;


Toggle Coverage:
  Enabled Coverage            Bins     Hits    Misses  Coverage
  ----------------            ----     ----    ------  --------
```

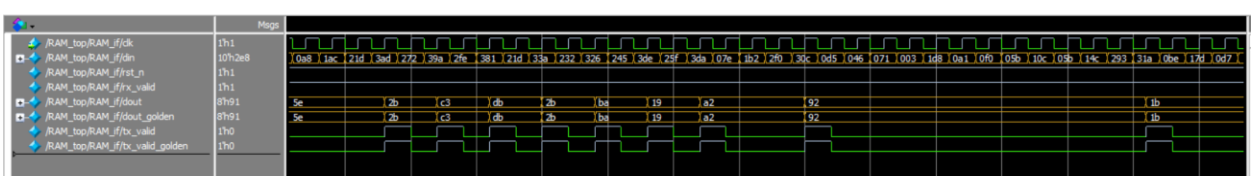
```
    Toggles                    76       76        0   100.00%

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

Toggle Coverage for instance /RAM_top/DUT --

                                  Node     1H->0L     0L->1H  "Coverage"
                                  ---------------------------------------
                            Rd_Addr[7-0]         1          1     100.00
                            Wr_Addr[7-0]         1          1     100.00
                                   clk          1          1     100.00
                              din[0-9]          1          1     100.00
                             dout[7-0]          1          1     100.00
                                 rst_n          1          1     100.00
                              rx_valid          1          1     100.00
                              tx_valid          1          1     100.00

Total Node Count     =        38
Toggled Node Count   =        38
Untoggled Node Count =         0

Toggle Coverage      =     100.00% (76 of 76 bins)

==============================================================================
```

Bug Report

- Change line 26 from 2'b11 : dout <= MEM[Wr_Addr]; to 2'b11 : dout <= MEM[Rd_Addr]; as when reading I access memory using read address.
- Change line 30 from tx_valid <= (din[9] && din[8] && rx_valid)? 1'b1 : 1'b0; to tx_valid <= (din[9] && din[8])? 1'b1 : 1'b0; and move it to line 29 (inside if statement) as we assume in case of reading tx_valid is always 1 also adding begin --- end to contain it.

QuestaSim Snippets

Waveform Snippet



Transcript Snippet

```
#   You are using a version of the UVM library that has been compiled
#   with `UVM_NO_DEPRECATED undefined.
#   See http://www.eda.org/svdb/view.php?id=3313 for more details.
#
#   You are using a version of the UVM library that has been compiled
#   with `UVM_OBJECT_MUST_HAVE_CONSTRUCTOR undefined.
#   See http://www.eda.org/svdb/view.php?id=3770 for more details.
#
#       (Specify +UVM_NO_RELNOTES to turn off this notice)
#
# UVM_INFO verilog_src/questa_uvm_pkg-1.2/src/questa_uvm_pkg.sv(277) @ 0: reporter [Questa UVM] QUESTA_UVM-1.2.3
# UVM_INFO verilog_src/questa_uvm_pkg-1.2/src/questa_uvm_pkg.sv(278) @ 0: reporter [Questa UVM]  questa_uvm::init(all)
# UVM_INFO @ 0: reporter [RNTST] Running test RAM_test...
# UVM_INFO RAM_test.sv(51) @ 0: uvm_test_top [run phase] reset asserted
# ************************************************************************
# * Questa UVM Transaction Recording Turned ON.                          *
# * recording_detail has been set.                                       *
# *  To turn off, set 'recording_detail' to off:                         *
# * uvm_config_db#(int)            ::set(null, "", "recording_detail", 0); *
# * uvm_config_db#(uvm_bitstream_t)::set(null, "", "recording_detail", 0); *
# ************************************************************************
# UVM_INFO RAM_test.sv(53) @ 20: uvm_test_top [run phase] reset deasserted
# UVM_INFO RAM_test.sv(54) @ 20: uvm_test_top [run phase] stimulas generation started of write only seq
# UVM_INFO RAM_test.sv(56) @ 200020: uvm_test_top [run phase] stimulas generation ended of write only seq
# UVM_INFO RAM_test.sv(57) @ 200020: uvm_test_top [run phase] stimulas generation started of read only seq
# UVM_INFO RAM_test.sv(59) @ 400020: uvm_test_top [run phase] stimulas generation ended of read only seq
# UVM_INFO RAM_test.sv(60) @ 400020: uvm_test_top [run phase] stimulas generation started of read-write seq
# UVM_INFO RAM_test.sv(62) @ 600020: uvm_test_top [run phase] stimulas generation ended of read-write seq
# UVM_INFO verilog_src/uvm-1.1d/src/base/uvm_objection.svh(1267) @ 600020: reporter [TEST_DONE] 'run' phase is ready to proceed to the 'extract' phase
# UVM_INFO RAM_scorboard.sv(54) @ 600020: uvm_test_top.RAM_env.sb [repo phase] RAM correct times: 30001
# UVM_INFO RAM_scorboard.sv(55) @ 600020: uvm_test_top.RAM_env.sb [repo phase] RAM error times: 0
#
# --- UVM Report Summary ---
#
# ** Report counts by severity
# UVM_INFO :   14
# UVM_WARNING :    0
# UVM_ERROR :    0
# UVM_FATAL :    0
# ** Report counts by id
# [Questa UVM]     2
# [RNTST]     1
# [TEST_DONE]     1
# [repo phase]     2
# [run phase]     8
# ** Note: $finish    : C:/questasim64_2021.1/win64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh(430)
#    Time: 600020 ns  Iteration: 61  Instance: /RAM_top
```

Code Coverage Report Snippets

Branch

```
Code Coverage Analysis
Branches - by instance (/RAM_top/DUT)                                   Branch
RAM.v
    14 if (~rst_n) begin
    20 else begin
    21 if (rx_valid) begin
    23 2'b00 : Wr_Addr <= din[7:0];
    24 2'b01 : MEM[Wr_Addr] <= din[7:0];
    25 2'b10 : Rd_Addr <= din[7:0];
    26 2'b11 : dout <= MEM[Rd_Addr];
E   27 default : dout <= 0;
RAM_top.sv
```

Statement

```
Code Coverage Analysis
Statements - by instance (/RAM_top/DUT)                                 Statement
RAM.v
    13 always @(posedge clk) begin
    15 dout <= 0;
    16 tx_valid <= 0;
    17 Rd_Addr <= 0;
    18 Wr_Addr <= 0;
    23 2'b00 : Wr_Addr <= din[7:0];
    24 2'b01 : MEM[Wr_Addr] <= din[7:0];
    25 2'b10 : Rd_Addr <= din[7:0];
    26 2'b11 : dout <= MEM[Rd_Addr];
E   27 default : dout <= 0;
    30 tx_valid <= (din[9] && din[8] && rx_valid)? 1'b1 : 1'b0;
RAM_top.sv
```

Toggle

Code Coverage Analysis

Toggles - by instance (/RAM_top/RAM_if)

Toggle	
sim:/RAM_top/RAM_if	
clk	✓
din	✓
dout	✓
dout_golden	✓
rst_n	✓
rx_valid	✓
tx_valid	✓
tx_valid_golden	✓

Functional Coverage Report Snippets

File Bookmarks Window Help

Covergroups

Name	Class Type	Coverage	Goal	% of Goal	Status	Included	Merge_instances	Get_inst_coverage	Comment
/RAM_collector_pkg/RAM_collector		100.00%							
TYPE RAM_cvr		100.00%	100	100.00...		✓	auto(0)		
CVP RAM_cvr::din_cp		100.00%	100	100.00...		✓			
CVP RAM_cvr::rx_valid_CP		100.00%	100	100.00...		✓			
CVP RAM_cvr::tx_valid_CP		100.00%	100	100.00...		✓			
CROSS RAM_cvr::din_with_rx		100.00%	100	100.00...		✓			
CROSS RAM_cvr::din_read_with_tx		100.00%	100	100.00...		✓			
INST \/RAM_collector_pkg::RAM_collector::RAM_cvr		100.00%	100	100.00...		✓		0	
CVP din_cp		100.00%	100	100.00...		✓			
bin write_address		9765	1	100.00...		✓			
bin write_data		5032	1	100.00...		✓			
bin read_address		9694	1	100.00...		✓			
bin read_data		4930	1	100.00...		✓			
bin write_data_after_write_address		5022	1	100.00...		✓			
bin read_data_after_read_address		4922	1	100.00...		✓			
bin full_trans		469	1	100.00...		✓			
CVP rx_valid_CP		100.00%	100	100.00...		✓			
bin high		27915	1	100.00...		✓			
bin low		1506	1	100.00...		✓			
CVP tx_valid_CP		100.00%	100	100.00...		✓			
bin high		4707	1	100.00...		✓			
bin low		24714	1	100.00...		✓			
CROSS din_with_rx		100.00%	100	100.00...		✓			
bin <full_trans,high>		452	1	100.00...		✓			
bin <read_data_after_read_address,high>		4699	1	100.00...		✓			
bin <write_data_after_write_address,high>		4772	1	100.00...		✓			
bin <read_data,high>		4707	1	100.00...		✓			
bin <write_data,high>		4778	1	100.00...		✓			
bin <read_address,high>		9187	1	100.00...		✓			
bin <write_address,high>		9243	1	100.00...		✓			
ignore_bin low_tx		150	-	-		✓			
CROSS din_read_with_tx		100.00%	100	100.00...		✓			
bin checked		4707	1	100.00...		✓			

Sequential Domain Coverage (Assertion Coverage) Snippets

Cover Directives

Cover Directives

Name	Language	Enabled	Log	Count	AtLeast	Limit	Weight	Cmplt %	Cmplt graph	Included	Memory	Peak Memory	Peak Memory
/RAM_top/DUT/assertion_mod/cover__read_data_eventually_af...	SVA	✓	Off	9106	1	Unlimited	1	100%		✓	0	0	
/RAM_top/DUT/assertion_mod/cover__write_data_eventually_af...	SVA	✓	Off	9125	1	Unlimited	1	100%		✓	0	0	
/RAM_top/DUT/assertion_mod/cover__tx_valid_on_seq	SVA	✓	Off	4613	1	Unlimited	1	100%		✓	0	0	
/RAM_top/DUT/assertion_mod/cover__tx_valid_off_seq_of_read...	SVA	✓	Off	9200	1	Unlimited	1	100%		✓	0	0	
/RAM_top/DUT/assertion_mod/cover__tx_valid_off_seq_of_wrti...	SVA	✓	Off	4741	1	Unlimited	1	100%		✓	0	0	
/RAM_top/DUT/assertion_mod/cover__tx_valid_off_seq_of_wrti...	SVA	✓	Off	9230	1	Unlimited	1	100%		✓	0	0	
/RAM_top/DUT/assertion_mod/cover__sync_reset	SVA	✓	Off	329	1	Unlimited	1	100%		✓	0	0	

Dataflow | Assertions | Cover Directives | RAM.v | uvm_root.svh

Assertions

Assertions

Name	Assertion Type	Language	Enable	Failure Count	Pass Count	Active Count	Memor
/RAM_write_read_seq_pkg::RAM_write_read_seq::body/#ublk#161771991#20/immed__22	Immediate	SVA	on	0	1	-	
/RAM_read_only_seq_pkg::RAM_read_only_seq::body/#ublk#244104311#20/immed__22	Immediate	SVA	on	0	1	-	
/RAM_write_only_seq_pkg::RAM_write_only_seq::body/#ublk#219035351#20/immed__22	Immediate	SVA	on	0	1	-	
/RAM_top/DUT/assertion_mod/assert__sync_reset	Concurrent	SVA	on	0	1	-	0
/RAM_top/DUT/assertion_mod/assert__tx_valid_off_seq_of_wrtie_add	Concurrent	SVA	on	0	1	-	0
/RAM_top/DUT/assertion_mod/assert__tx_valid_off_seq_of_wrtie_data	Concurrent	SVA	on	0	1	-	0
/RAM_top/DUT/assertion_mod/assert__tx_valid_off_seq_of_read_add	Concurrent	SVA	on	0	1	-	0
/RAM_top/DUT/assertion_mod/assert__tx_valid_on_seq	Concurrent	SVA	on	0	1	-	0
/RAM_top/DUT/assertion_mod/assert__write_data_eventually_after_address	Concurrent	SVA	on	0	1	-	0
/RAM_top/DUT/assertion_mod/assert__read_data_eventually_after_address	Concurrent	SVA	on	0	1	-	0

Dataflow | Assertions | Cover Directives | RAM.v | uvm_root.svh

Assertions' Table

Feature	Assertions
whenever reset is asserted, the output signals (tx_valid and dout) are low	`@(posedge clk) !rst_n |=> !tx_valid && (dout == 0);`
During address or data input phases (write_add_seq), the tx_valid signal must remain deasserted.	`@(posedge clk) disable iff(!rst_n) (din[9:8] == 2'b00) && (rx_valid == 1) |=> !tx_valid;`
During address or data input phases (write_data_seq), the tx_valid signal must remain deasserted.	`@(posedge clk) disable iff(!rst_n) (din[9:8] == 2'b01) && (rx_valid == 1) |=> !tx_valid;`
During address or data input phases (read_add_seq), the tx_valid signal must remain deasserted.	`@(posedge clk) disable iff(!rst_n) (din[9:8] == 2'b10) && (rx_valid == 1) |=> !tx_valid;`
Whenever a read_data_seq is asserted, the tx_valid signal must rise to indicate valid output and after it rises by one clock cycle, it should eventually fall.	`@(posedge clk) disable iff(!rst_n) (din[9:8] == 2'b11) && (rx_valid == 1) |=> tx_valid ##1 (!tx_valid)[->1];`
Every Write Address operation must be eventually followed by a Write Data operation.	`@(posedge clk) disable iff(!rst_n) (din[9:8] == 2'b00) && (rx_valid == 1) |=> (din[9:8] == 2'b01)[->1];`
Every Read Address operation must be eventually followed by a Read Data operation.	`@(posedge clk) disable iff(!rst_n) (din[9:8] == 2'b10) && (rx_valid == 1) |=> (din[9:8] == 2'b11)[->1];`

Part 3: UVM Environment for SPI Wrapper

SPI Wrapper

```
module WRAPPER (MOSI,MISO,SS_n,clk,rst_n);
input  MOSI, SS_n, clk, rst_n;
output MISO;

wire [9:0] rx_data_din;
wire       rx_valid;
wire       tx_valid;
wire [7:0] tx_data_dout;

RAM   RAM_instance   (rx_data_din,clk,rst_n,rx_valid,tx_data_dout,tx_valid);
SLAVE SLAVE_instance
(MOSI,MISO,SS_n,clk,rst_n,rx_data_din,rx_valid,tx_data_dout,tx_valid);

endmodule
```

SPI Wrapper Golden

```
module SPI_wrapper_golden (MOSI,MISO,SS_n,clk,rst_n);
    input MOSI, SS_n, clk, rst_n;
    output MISO;

    // Internal connection wires between SPI and RAM
    wire [9:0] rx_data;
    wire rx_valid;
    wire [7:0] tx_data;
    wire tx_valid;

    SPI_slave_golden SPI_GOLDEN (MOSI, SS_n, clk, rst_n, tx_valid, tx_data, MISO,
rx_valid, rx_data);

    RAM_golden RAM_GOLDEN (.clk(clk), .rst_n(rst_n), .rx_valid(rx_valid),
.din(rx_data), .tx_valid(tx_valid), .dout(tx_data));
endmodule : SPI_wrapper_golden
```

SPI Wrapper interface

```
interface SPI_wrapper_if (clk);
    input bit clk;

    logic MOSI, SS_n, rst_n;
    logic MISO, MISO_gold;
endinterface : SPI_wrapper_if
```

SPI Wrapper Assertions' File

```
module SPI_wrapper_sva (MOSI,MISO,SS_n,clk,rst_n);

    input bit MOSI,MISO,SS_n,clk,rst_n;

    property reset_check;
        @(posedge clk) (!rst_n) |=> (!MISO);
    endproperty

    property MISO_stable_check;
        @(posedge clk) disable iff (!rst_n) $fell(SS_n) |=> (SS_n == 1'b0 &&
$stable(MISO)) [*11];
    endproperty

    assert property (reset_check);
    assert property (MISO_stable_check);

    cover property (reset_check);
    cover property (MISO_stable_check);

endmodule : SPI_wrapper_sva
```

Shared Package

```
package shared_pkg;

    bit [5:0] count;
    bit is_read, have_address_to_read;
    int limit;
    logic [10:0] keep_arr;


    bit write_address_done, write_data_done, read_address_done, read_data_done;

endpackage
```

SPI_wrapper_seq_item.sv

```
package SPI_wrapper_seq_item_pkg;
    import uvm_pkg::*;
    import shared_pkg::*;

    `include "uvm_macros.svh"
```

```
class SPI_wrapper_seq_item extends uvm_sequence_item;
    `uvm_object_utils(SPI_wrapper_seq_item)

    rand logic MOSI, SS_n, rst_n;
    rand logic [10:0] arr_MOSI;

    logic MISO, MISO_gold;

    function new(string name = "SPI_wrapper_seq_item");
        super.new(name);
    endfunction : new

    constraint reset_c {rst_n dist {0 := 2 , 1 := 98};}

    constraint valid_MOSI_command {
        arr_MOSI [10:8] inside {3'b000, 3'b001, 3'b110, 3'b111};

        if (!have_address_to_read) {(arr_MOSI[10:8] != {3'b111});}
    }

    constraint next_op {
        if (write_address_done) {
            soft arr_MOSI[9] == 1'b0;
        }
        else if (read_address_done) {
            soft arr_MOSI[9:8] == 2'b11;
        }
    }

    constraint next_read_data {
        if (read_data_done) {
            soft arr_MOSI[9:8] == 2'b10;
        }
    }

    constraint read_after_address {
        if(!read_address_done){
            arr_MOSI[9:8] != 2'b11;
        }
    }

    constraint after_write_data {
        if (write_data_done) {
            soft arr_MOSI[9:8] dist {2'b10 := 60 , 2'b00 := 40};
        }
```

```
        else if (read_data_done) {
            soft arr_MOSI[9:8] dist {2'b10 := 40 , 2'b00 := 60};
        }
    }

    function void post_randomize();

        if (count == 0) keep_arr = arr_MOSI;

        is_read = (keep_arr [10:8] == 3'b111)? 1:0;
        limit = (is_read)? 23:13;

        SS_n = (count == limit)? 1:0;

        if (keep_arr [10:8] == 3'b110) have_address_to_read = 1'b1;
        if (is_read || (!rst_n)) have_address_to_read = 1'b0;

        //
        if((count > 0) && (count < 12)) begin
            MOSI = keep_arr [11-count];
        end else

        // ram
        if (keep_arr [9:8] == 2'b00) write_address_done = 1;
        else write_address_done = 0;

        if (keep_arr [9:8] == 2'b01) write_data_done = 1;
        else write_data_done = 0;


        if (keep_arr [9:8] == 2'b10) read_address_done = 1;
        else read_address_done = 0;

        if (keep_arr [9:8] == 2'b11) read_data_done = 1;
        else read_data_done = 0;

        // count
        if (!rst_n) begin
            count = 0;
        end else begin
            if (count == limit) count = 0;
            else count++;
        end
    endfunction : post_randomize
```

```
        function string convert2string();
            return $sformatf("%s MOSI = %0b, SS_n = %0b, rst_n = %0b, MISO =
%0b",super.convert2string(), MOSI, SS_n, rst_n, MISO);
        endfunction : convert2string

        function string convert2string_stim();
            return $sformatf("%s MOSI = %0b, SS_n = %0b, rst_n =
%0b",super.convert2string(), MOSI, SS_n, rst_n);
        endfunction : convert2string_stim

    endclass : SPI_wrapper_seq_item
endpackage : SPI_wrapper_seq_item_pkg
```

SPI Wrapper Reset Sequence

```
package SPI_wrapper_reset_seq_pkg;
    import uvm_pkg::*;
    import SPI_wrapper_seq_item_pkg::*;

    `include "uvm_macros.svh"

    class SPI_wrapper_reset_seq extends uvm_sequence #(SPI_wrapper_seq_item);
        `uvm_object_utils(SPI_wrapper_reset_seq)

        SPI_wrapper_seq_item seq_item;

        function new(string name = "SPI_wrapper_reset_seq");
            super.new(name);
        endfunction : new

        task body();
            seq_item = SPI_wrapper_seq_item::type_id::create("seq_item");

            start_item(seq_item);
            seq_item.rst_n = 0;
            finish_item(seq_item);

        endtask : body
    endclass : SPI_wrapper_reset_seq

endpackage : SPI_wrapper_reset_seq_pkg
```

SPI Wrapper Write Only Sequence

```
package SPI_wrapper_WO_seq_pkg;
    import uvm_pkg::*;
    import SPI_wrapper_seq_item_pkg::*;

    `include "uvm_macros.svh"

    class SPI_wrapper_WO_seq extends uvm_sequence #(SPI_wrapper_seq_item);
        `uvm_object_utils(SPI_wrapper_WO_seq)

        SPI_wrapper_seq_item seq_item;

        function new(string name = "SPI_wrapper_WO_seq");
            super.new(name);
        endfunction : new

        task body();
            seq_item = SPI_wrapper_seq_item::type_id::create("seq_item");

            repeat(10000) begin
                start_item(seq_item);
                assert(seq_item.randomize() with {arr_MOSI [9] == 1'b0;});
                finish_item(seq_item);
            end

        endtask : body
    endclass : SPI_wrapper_WO_seq

endpackage : SPI_wrapper_WO_seq_pkg
```

SPI Wrapper RO Sequence

```
package SPI_wrapper_RO_seq_pkg;
    import uvm_pkg::*;
    import SPI_wrapper_seq_item_pkg::*;

    `include "uvm_macros.svh"

    class SPI_wrapper_RO_seq extends uvm_sequence #(SPI_wrapper_seq_item);
        `uvm_object_utils(SPI_wrapper_RO_seq)

        SPI_wrapper_seq_item seq_item;

        function new(string name = "SPI_wrapper_RO_seq");
            super.new(name);
```

```
        endfunction : new

        task body();
            seq_item = SPI_wrapper_seq_item::type_id::create("seq_item");

            repeat(10000) begin
                start_item(seq_item);
                assert(seq_item.randomize() with {arr_MOSI [9] == 1'b1;});
                finish_item(seq_item);
            end

        endtask : body
    endclass : SPI_wrapper_RO_seq

endpackage : SPI_wrapper_RO_seq_pkg
```

SPI Wrapper Write Read Sequence

```
package SPI_wrapper_WR_seq_pkg;
    import uvm_pkg::*;
    import SPI_wrapper_seq_item_pkg::*;

    `include "uvm_macros.svh"

    class SPI_wrapper_WR_seq extends uvm_sequence #(SPI_wrapper_seq_item);
        `uvm_object_utils(SPI_wrapper_WR_seq)

        SPI_wrapper_seq_item seq_item;

        function new(string name = "SPI_wrapper_WR_seq");
            super.new(name);
        endfunction : new

        task body();
            seq_item = SPI_wrapper_seq_item::type_id::create("seq_item");

            seq_item.next_read_data.constraint_mode(0);
            repeat(10000) begin
                start_item(seq_item);
                assert(seq_item.randomize());
                finish_item(seq_item);
            end

        endtask : body
    endclass : SPI_wrapper_WR_seq
endpackage : SPI_wrapper_WR_seq_pkg
```

SPI Wrapper Sequencer

```
package SPI_wrapper_seqr_pkg;
    import uvm_pkg::*;
    import SPI_wrapper_seq_item_pkg::*;

    `include "uvm_macros.svh"

    class SPI_wrapper_seqr extends uvm_sequencer #(SPI_wrapper_seq_item);
        `uvm_component_utils(SPI_wrapper_seqr)

        function new(string name = "SPI_wrapper_seqr" , uvm_component parent =
null);
            super.new(name , parent);
        endfunction : new
    endclass : SPI_wrapper_seqr

endpackage : SPI_wrapper_seqr_pkg
```

SPI Wrapper Configuration Object

```
package SPI_wrapper_config_pkg;
    import uvm_pkg::*;
    `include "uvm_macros.svh"

    class SPI_wrapper_config extends uvm_object;
        `uvm_object_utils(SPI_wrapper_config)

        virtual SPI_wrapper_if wrapper_if;
        uvm_active_passive_enum is_active;

        function new(string name = "SPI_wrapper_config");
            super.new(name);
        endfunction : new

    endclass : SPI_wrapper_config

endpackage : SPI_wrapper_config_pkg
```

SPI Wrapper Driver

```
package SPI_wrapper_driver_pkg;
    import SPI_wrapper_seq_item_pkg::*;
    import uvm_pkg::*;

    `include "uvm_macros.svh"
```

```
    class SPI_wrapper_driver extends uvm_driver #(SPI_wrapper_seq_item);
        `uvm_component_utils(SPI_wrapper_driver)

        virtual SPI_wrapper_if wrapper_if;
        SPI_wrapper_seq_item seq_item;

        function new(string name = "SPI_wrapper_driver" , uvm_component parent =
null);
            super.new(name,parent);
        endfunction : new

        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
        endfunction : build_phase

        task run_phase(uvm_phase phase);
            super.run_phase(phase);

            forever begin
                seq_item = SPI_wrapper_seq_item::type_id::create("seq_item");
                seq_item_port.get_next_item(seq_item);

                wrapper_if.MOSI  = seq_item.MOSI;
                wrapper_if.SS_n  = seq_item.SS_n;
                wrapper_if.rst_n = seq_item.rst_n;

                @(negedge wrapper_if.clk);
                seq_item_port.item_done();
                `uvm_info("run_phase" , seq_item.convert2string_stim(), UVM_HIGH)
            end
        endtask : run_phase

    endclass : SPI_wrapper_driver
endpackage : SPI_wrapper_driver_pkg
```

SPI Wrapper Monitor

```
package SPI_wrapper_mon_pkg;
    import uvm_pkg::*;
    import SPI_wrapper_seq_item_pkg::*;

    `include "uvm_macros.svh"

    class SPI_wrapper_mon extends uvm_monitor;
        `uvm_component_utils(SPI_wrapper_mon)
```

```
        virtual SPI_wrapper_if wrapper_if;
        SPI_wrapper_seq_item seq_item;
        uvm_analysis_port #(SPI_wrapper_seq_item) mon_ap;

        function new(string name = "SPI_wrapper_mon" , uvm_component parent =
null);
            super.new(name , parent);
        endfunction : new

        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            mon_ap = new("mon_ap",this);
        endfunction : build_phase

        task run_phase(uvm_phase phase);
            super.run_phase(phase);

            forever begin
                seq_item = SPI_wrapper_seq_item::type_id::create("seq_item");
                @(negedge wrapper_if.clk);

                seq_item.rst_n     = wrapper_if.rst_n;
                seq_item.MOSI      = wrapper_if.MOSI;
                seq_item.SS_n      = wrapper_if.SS_n;
                seq_item.MISO      = wrapper_if.MISO;
                seq_item.MISO_gold = wrapper_if.MISO_gold;

                mon_ap.write(seq_item);
                `uvm_info("run_phase" , seq_item.convert2string(), UVM_HIGH)
            end
        endtask : run_phase

    endclass : SPI_wrapper_mon
endpackage : SPI_wrapper_mon_pkg
```

SPI Wrapper Agent

```
package SPI_wrapper_agent_pkg;
    import uvm_pkg::*;
    import SPI_wrapper_seqr_pkg::*;
    import SPI_wrapper_driver_pkg::*;
    import SPI_wrapper_mon_pkg::*;
    import SPI_wrapper_config_pkg::*;
    import SPI_wrapper_seq_item_pkg::*;
```

```
    `include "uvm_macros.svh"

    class SPI_wrapper_agent extends uvm_agent;
        `uvm_component_utils(SPI_wrapper_agent);

        SPI_wrapper_seqr seqr;
        SPI_wrapper_driver drv;
        SPI_wrapper_mon mon;
        SPI_wrapper_config cfg;
        uvm_analysis_port #(SPI_wrapper_seq_item) agt_ap;

        function new(string name = "SPI_wrapper_agent" , uvm_component parent =
null);
            super.new(name, parent);
        endfunction : new

        function void build_phase(uvm_phase phase);
            super.build_phase(phase);

            uvm_config_db#(SPI_wrapper_config)::get(this, "", "CFG", cfg);

            mon = SPI_wrapper_mon::type_id::create("mon",this);
            if (cfg.is_active == UVM_ACTIVE) begin
                seqr = SPI_wrapper_seqr::type_id::create("seqr",this);
                drv = SPI_wrapper_driver::type_id::create("drv",this);
            end

            agt_ap = new("agt_ap",this);
        endfunction : build_phase

        function void connect_phase(uvm_phase phase);
            super.connect_phase(phase);

            mon.wrapper_if = cfg.wrapper_if;
            mon.mon_ap.connect(agt_ap);

            if (cfg.is_active == UVM_ACTIVE) begin
                drv.wrapper_if = cfg.wrapper_if;
                drv.seq_item_port.connect(seqr.seq_item_export);
            end
        endfunction : connect_phase

    endclass : SPI_wrapper_agent
endpackage : SPI_wrapper_agent_pkg
```

SPI Wrapper Scoreboard

```
package SPI_wrapper_scoreboard_pkg;
    import uvm_pkg::*;
    import SPI_wrapper_seq_item_pkg::*;

    `include "uvm_macros.svh"

    class SPI_wrapper_scoreboard extends uvm_scoreboard;
        `uvm_component_utils(SPI_wrapper_scoreboard)
        uvm_analysis_export #(SPI_wrapper_seq_item) sb_exp;
        uvm_tlm_analysis_fifo #(SPI_wrapper_seq_item) sb_fifo;
        SPI_wrapper_seq_item seq_item;

        int correct_count = 0 , error_count = 0;

        function new(string name = "SPI_wrapper_scoreboard" , uvm_component
parent = null);
            super.new(name,parent);
        endfunction : new

        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            sb_exp = new("sb_exp",this);
            sb_fifo = new("sb_fifo",this);
        endfunction : build_phase

        function void connect_phase(uvm_phase phase);
            super.connect_phase(phase);
            sb_exp.connect(sb_fifo.analysis_export);
        endfunction : connect_phase

        task run_phase(uvm_phase phase);
            super.run_phase(phase);

            forever begin
                sb_fifo.get(seq_item);
                if(seq_item.MISO === seq_item.MISO_gold)
                    correct_count++;
                else begin
                    error_count++;
                    $display("time:%0t SS_n =%b dut out = %b , ref out =
%b",$time,seq_item.SS_n,seq_item.MISO,seq_item.MISO_gold);
                end
            end
```

```
        endtask : run_phase

        function void report_phase(uvm_phase phase);
           super.report_phase(phase);
           $display("correct_count =%0d , error_count
=%0d",correct_count,error_count);
        endfunction : report_phase

    endclass : SPI_wrapper_scoreboard
endpackage : SPI_wrapper_scoreboard_pkg
```

SPI Wrapper Coverage Collector

```
package SPI_wrapper_cover_collect_pkg;
    import uvm_pkg::*;
    import SPI_wrapper_seq_item_pkg::*;

    `include "uvm_macros.svh"

    class SPI_wrapper_cover_collect extends uvm_component;
        `uvm_component_utils(SPI_wrapper_cover_collect)
        uvm_analysis_export #(SPI_wrapper_seq_item) cov_exp;
        uvm_tlm_analysis_fifo #(SPI_wrapper_seq_item) cov_fifo;

        SPI_wrapper_seq_item seq_item;

        covergroup cg();
            // to be added
        endgroup : cg

        function new(string name = "SPI_wrapper_cover_collect" , uvm_component
parent = null);
            super.new(name,parent);
            cg = new();
        endfunction : new

        function void build_phase(uvm_phase phase);
            super.build_phase(phase);
            cov_exp = new("cov_exp",this);
            cov_fifo = new("cov_fifo",this);
        endfunction : build_phase

        function void connect_phase(uvm_phase phase);
            super.connect_phase(phase);
            cov_exp.connect(cov_fifo.analysis_export);
        endfunction : connect_phase
```

```
        task run_phase(uvm_phase phase);
            super.run_phase(phase);

            forever begin
                cov_fifo.get(seq_item);
                cg.sample();
            end
        endtask : run_phase
    endclass : SPI_wrapper_cover_collect
endpackage : SPI_wrapper_cover_collect_pkg
```

SPI Wrapper Environment

```
package SPI_wrapper_env_pkg;
import SPI_wrapper_cover_collect_pkg::*;
import SPI_wrapper_scoreboard_pkg::*;
import SPI_wrapper_agent_pkg::*;
import uvm_pkg::*;
`include "uvm_macros.svh"

class SPI_wrapper_env extends uvm_env;
  `uvm_component_utils(SPI_wrapper_env)

  SPI_wrapper_scoreboard sb;
  SPI_wrapper_cover_collect cov;
  SPI_wrapper_agent agt;

  function new(string name = "SPI_wrapper_env" , uvm_component parent = null);
    super.new(name,parent);
  endfunction : new

  function void build_phase(uvm_phase phase);
    super.build_phase(phase);
    sb = SPI_wrapper_scoreboard::type_id::create("sb",this);
    cov = SPI_wrapper_cover_collect::type_id::create("cov",this);
    agt = SPI_wrapper_agent::type_id::create("agt",this);
  endfunction : build_phase

  function void connect_phase(uvm_phase phase);
    super.connect_phase(phase);
    agt.agt_ap.connect(sb.sb_exp);
    agt.agt_ap.connect(cov.cov_exp);
  endfunction : connect_phase
endclass : SPI_wrapper_env
endpackage : SPI_wrapper_env_pkg
```

SPI Wrapper Test

```
package SPI_wrapper_test_pkg;

import SPI_wrapper_env_pkg::*;
import SPI_wrapper_config_pkg::*;
import SPI_wrapper_reset_seq_pkg::*;
import SPI_wrapper_WO_seq_pkg::*;
import SPI_wrapper_RO_seq_pkg::*;
import SPI_wrapper_WR_seq_pkg::*;

import SPI_slave_config_pkg::*;
import SPI_slave_env_pkg::*;

import RAM_config_pkg::*;
import RAM_env_pkg::*;

import uvm_pkg::*;
`include "uvm_macros.svh"


class SPI_wrapper_test extends uvm_test;
  `uvm_component_utils(SPI_wrapper_test)

  virtual SPI_wrapper_if wrapper_if;
  virtual RAM_interface ram_if;
  virtual SLAVE_interface spi_if;

  SPI_wrapper_env wrapper_env;
  SPI_wrapper_config wrapper_cfg;

  RAM_env ram_env;
  RAM_config ram_cfg;

  SPI_slave_env spi_env;
  SPI_slave_config spi_cfg;

  SPI_wrapper_reset_seq reset_seq;
  SPI_wrapper_WO_seq write_seq;
  SPI_wrapper_RO_seq read_seq;
  SPI_wrapper_WR_seq write_read_seq;

  function new(string name = "SPI_wrapper_test" , uvm_component parent = null);
    super.new(name,parent);
  endfunction : new
```

```
  function void build_phase(uvm_phase phase);
    super.build_phase(phase);
    wrapper_env = SPI_wrapper_env::type_id::create("wrapper_env",this);
    wrapper_cfg = SPI_wrapper_config::type_id::create("wrapper_cfg");

    ram_env = RAM_env::type_id::create("ram_env",this);
    ram_cfg = RAM_config::type_id::create("ram_cfg");

    spi_env = SPI_slave_env::type_id::create("spi_env",this);
    spi_cfg = SPI_slave_config::type_id::create("spi_cfg");

    write_seq = SPI_wrapper_WO_seq::type_id::create("write_seq");
    reset_seq = SPI_wrapper_reset_seq::type_id::create("reset_seq");
    read_seq = SPI_wrapper_RO_seq::type_id::create("read_seq");
    write_read_seq = SPI_wrapper_WR_seq::type_id::create("write_read_seq");

    if(!uvm_config_db#(virtual SPI_wrapper_if)::get(this, "", "inter",
wrapper_cfg.wrapper_if))
      `uvm_fatal("build_phase" , "error test");

    if(!uvm_config_db#(virtual RAM_interface)::get(this, "", "RAM_if",
ram_cfg.RAM_vif))
      `uvm_fatal("build_phase" , "error test");

    if(!uvm_config_db#(virtual SLAVE_interface)::get(this, "", "Slave_if",
spi_cfg.SPI_slave_vif))
      `uvm_fatal("build_phase" , "error test");

    wrapper_cfg.is_active = UVM_ACTIVE;
    ram_cfg.is_active     = UVM_PASSIVE;
    spi_cfg.is_active     = UVM_PASSIVE;

    uvm_config_db#(SPI_wrapper_config)::set(this, "*", "CFG", wrapper_cfg);
    uvm_config_db#(RAM_config)::set(this, "*", "CFG_RAM", ram_cfg);
    uvm_config_db#(SPI_slave_config)::set(this, "*", "CFG_slave", spi_cfg);
  endfunction : build_phase

  task run_phase(uvm_phase phase);
    super.run_phase(phase);
    phase.raise_objection(this);
    reset_seq.start(wrapper_env.agt.seqr);
    write_seq.start(wrapper_env.agt.seqr);
    read_seq.start(wrapper_env.agt.seqr);
    write_read_seq.start(wrapper_env.agt.seqr);
```

```
    phase.drop_objection(this);
  endtask : run_phase

endclass: SPI_wrapper_test

endpackage : SPI_wrapper_test_pkg
```

top.sv

```
import uvm_pkg::*;
import SPI_wrapper_test_pkg::*;

`include "uvm_macros.svh"

module top();
  bit clk;

  // Clock generation
  initial begin
    forever #10 clk = ~clk;
  end

  // Instantiate the interface and DUT
  SPI_wrapper_if wrapper_if (clk);
  RAM_interface ram_if (clk);
  SLAVE_interface spi_if (clk);

  WRAPPER            DUT    (wrapper_if.MOSI,
wrapper_if.MISO,      wrapper_if.SS_n, wrapper_if.clk, wrapper_if.rst_n);
  SPI_wrapper_golden GOLDEN (wrapper_if.MOSI, wrapper_if.MISO_gold,
wrapper_if.SS_n, wrapper_if.clk, wrapper_if.rst_n);

  bind WRAPPER RAM_SVA sva_RAM_inst (.din(DUT.rx_data_din), .clk(wrapper_if.clk),
.rst_n(wrapper_if.rst_n),
                                     .rx_valid(DUT.rx_valid),
.dout(DUT.tx_data_dout), .tx_valid(DUT.tx_valid));

  bind WRAPPER SPI_wrapper_sva sva_WRAPPER_inst (wrapper_if.MOSI,
wrapper_if.MISO, wrapper_if.SS_n, wrapper_if.clk, wrapper_if.rst_n);


  assign ram_if.rst_n           = DUT.rst_n;
  assign ram_if.din             = DUT.rx_data_din;
  assign ram_if.rx_valid        = DUT.rx_valid;
  assign ram_if.dout            = DUT.tx_data_dout;
  assign ram_if.tx_valid        = DUT.tx_valid;
```

```
  assign ram_if.dout_golden     = GOLDEN.tx_data;
  assign ram_if.tx_valid_golden = GOLDEN.tx_valid;

  assign spi_if.rst_n           = DUT.rst_n;
  assign spi_if.MOSI            = DUT.MOSI;
  assign spi_if.SS_n            = DUT.SS_n;
  assign spi_if.rx_valid        = DUT.rx_valid;
  assign spi_if.rx_data         = DUT.rx_data_din;
  assign spi_if.tx_valid        = DUT.tx_valid;
  assign spi_if.tx_data         = DUT.tx_data_dout;
  assign spi_if.MISO            = DUT.MISO;
  assign spi_if.MISO_golden     = GOLDEN.MISO;
  assign spi_if.rx_data_golden  = GOLDEN.rx_data;
  assign spi_if.rx_valid_golden = GOLDEN.rx_valid;


  initial begin
    // Set the virtual interface for the uvm test
    uvm_config_db#(virtual SPI_wrapper_if)::set(null, "uvm_test_top", "inter",
wrapper_if);
    uvm_config_db#(virtual RAM_interface)::set(null, "uvm_test_top", "RAM_if",
ram_if);
    uvm_config_db#(virtual SLAVE_interface)::set(null, "uvm_test_top",
"Slave_if", spi_if);
    run_test("SPI_wrapper_test");
  end
endmodule
```

SRC File

```
SPI_slave.sv
RAM.v
SPI_wrapper.v
SPI_slave_golden.sv
RAM_golden.sv
SPI_wrapper_golden.v
SPI_slave_interface.sv
RAM_interface.sv
SPI_wrapper_if.sv
RAM_assertions.sv
SPI_wrapper_sva.sv
shared_pkg.sv
SPI_slave_seq_item.sv
RAM_seq_item.sv
SPI_wrapper_seq_item.sv
SPI_slave_R_seq.sv
SPI_slave_M_seq.sv
RAM_R_seq.sv
RAM_WO_seq.sv
RAM_RO_seq.sv
RAM_WR_seq.sv
SPI_wrapper_reset_seq.sv
SPI_wrapper_WO_seq.sv
SPI_wrapper_RO_seq.sv
SPI_wrapper_WR_seq.sv
SPI_slave_sqr.sv
RAM_sqr.sv
SPI_wrapper_seqr.sv
SPI_slave_config.sv
RAM_config.sv
SPI_wrapper_config.sv
SPI_slave_driver.sv
RAM_driver.sv
SPI_wrapper_driver.sv
SPI_slave_monitor.sv
RAM_monitor.sv
SPI_wrapper_mon.sv
SPI_slave_agent.sv
RAM_agent.sv
SPI_wrapper_agent.sv
SPI_slave_scoreboard.sv
RAM_scorboard.sv
```

```
SPI_wrapper_scoreboard.sv
SPI_slave_collector.sv
SPI_wrapper_cover_collect.sv
RAM_collector.sv
SPI_slave_env.sv
RAM_env.sv
SPI_wrapper_env.sv
SPI_wrapper_test.sv
top.sv
```

Do File

```
vlib work
vlog -f src_files.list +define+SIM +cover -covercells
vsim -voptargs=+acc work.top -classdebug -uvmcontrol=all -cover -sv_seed random
add wave /top/wrapper_if/*
coverage exclude -src RAM.v -line 27
coverage exclude -src SPI_slave.sv -line 37
coverage exclude -src SPI_slave.sv -line 38
coverage exclude -src SPI_slave.sv -line 69
coverage exclude -src SPI_slave.sv -line 129
coverage save top.ucdb -onexit -du WRAPPER
run -all
vcover report top.ucdb -details -annotate -all -output coverage_rpt.txt
coverage report -detail -cvg -directive -comments -output {fcover_report.txt} {}
```

- We exclude line 27 as there is a default case we won't enter.
- Excluding lines 37 and 38 as SS_n never goes high when we are in CHK_CMD so we don't enter the if statement of it.
- We add default case and Exclude it (lines 69,129) as we don't enter the default case and we did that to avoid all false statement.

Coverage Report

Code Coverage Report

```
Coverage Report by instance with details

================================================================================
=== Instance: /\top#DUT /RAM_instance
=== Design Unit: work.RAM
================================================================================
Branch Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Branches                         8         8         0   100.00%

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

Branch Coverage for instance /\top#DUT /RAM_instance

    Line       Item                      Count     Source
    ----       ----                      -----     ------
  File RAM.v
------------------------------------IF Branch------------------------------------
    14                                 15069     Count coming in to IF
    14          1                        599        if (~rst_n) begin

    20          1                      14470        else begin

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
    21                                 14470     Count coming in to IF
    21          1                       3306           if (rx_valid) begin

                                       11164     All False Count
Branch totals: 2 hits of 2 branches = 100.00%

-----------------------------------CASE Branch-----------------------------------
    22                                  3306     Count coming in to CASE
    23          1                       1303                 2'b00 : Wr_Addr <= din[7:0];

    24          1                        662                 2'b01 : MEM[Wr_Addr] <= din[7:0];

    25          1                        771                 2'b10 : Rd_Addr <= din[7:0];

    26          1                        570                 2'b11 : dout <= MEM[Rd_Addr];
```

```
Branch totals: 4 hits of 4 branches = 100.00%


Expression Coverage:
   Enabled Coverage            Bins   Covered    Misses  Coverage
   ----------------            ----     ----    ------  --------
   Expressions                    2        2         0   100.00%

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

Expression Coverage for instance /\top#DUT /RAM_instance –

  File RAM.v
----------------Focused Expression View-----------------
Line       29 Item    1  (din[9] && din[8])
Expression totals: 2 of 2 input terms covered = 100.00%

  Input Term   Covered  Reason for no coverage   Hint
 -----------  --------  -----------------------  --------------
     din[9]        Y
     din[8]        Y

    Rows:       Hits  FEC Target           Non-masking condition(s)
 ---------  ---------  --------------------  -------------------------
  Row   1:         1  din[9]_0             -
  Row   2:         1  din[9]_1             din[8]
  Row   3:         1  din[8]_0             din[9]
  Row   4:         1  din[8]_1             din[9]


Statement Coverage:
   Enabled Coverage            Bins      Hits    Misses  Coverage
   ----------------            ----      ----    ------  --------
   Statements                    10        10         0   100.00%

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

Statement Coverage for instance /\top#DUT /RAM_instance –

   Line        Item                    Count     Source
   ----        ----                    -----     ------
```

File RAM.v

```
565                                module RAM (din,clk,rst_n,rx_valid,dout,tx_valid);
2
3                                  input [9:0] din;
4                                  input clk, rst_n, rx_valid;
5
6                                  output reg [7:0] dout;
7                                  output reg tx_valid;
8
9                                  reg [7:0] MEM [255:0];
10
11                                  reg [7:0] Rd_Addr, Wr_Addr;
12
13          1                15069     always @(posedge clk) begin
14                                        if (~rst_n) begin
15          1                  599            dout <= 0;
16          1                  599            tx_valid <= 0;
17          1                  599            Rd_Addr <= 0;
18          1                  599            Wr_Addr <= 0;
19                                        end
20                                        else begin
21                                           if (rx_valid) begin
22                                              case (din[9:8])
```

```
23        1        1303        2'b00 : Wr_Addr <= din[7:0];

24        1         662        2'b01 : MEM[Wr_Addr] <= din[7:0];

25        1         771        2'b10 : Rd_Addr <= din[7:0];

26        1         570        2'b11 : dout <= MEM[Rd_Addr];

27                             default : dout <= 0;

28                           endcase

29        1        3306        tx_valid <= (din[9] && din[8])? 1'b1 : 1'b0;


Toggle Coverage:
   Enabled Coverage          Bins     Hits    Misses  Coverage
   ----------------         ----     ----    ------  --------
   Toggles                    76       76        0   100.00%

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

Toggle Coverage for instance /\top#DUT /RAM_instance –

                          Node      1H->0L      0L->1H  "Coverage"
                          ---------------------------------------
                     Rd_Addr[7-0]          1          1      100.00
                     Wr_Addr[7-0]          1          1      100.00
                           clk          1          1      100.00
                       din[0-9]          1          1      100.00
                      dout[7-0]          1          1      100.00
                          rst_n          1          1      100.00
                       rx_valid          1          1      100.00
                       tx_valid          1          1      100.00

Total Node Count     =        38
Toggled Node Count   =        38
Untoggled Node Count =         0

Toggle Coverage      =     100.00% (76 of 76 bins)

================================================================================
=== Instance: /\top#DUT /SLAVE_instance
```

```
=== Design Unit: work.SLAVE
=================================================================

Assertion Coverage:
   Assertions                  12       12        0   100.00%
-----------------------------------------------------------------
Name              File(Line)                Failure     Pass
                                             Count       Count
-----------------------------------------------------------------
/\top#DUT /SLAVE_instance/assert__READ_DATA_to_IDLE
                SPI_slave.sv(264)              0         1
/\top#DUT /SLAVE_instance/assert__READ_ADD_to_IDLE
                SPI_slave.sv(254)              0         1
/\top#DUT /SLAVE_instance/assert__WRITE_to_IDLE
                SPI_slave.sv(244)              0         1
/\top#DUT /SLAVE_instance/assert__CHK_CMD_to_READ_DATA
                SPI_slave.sv(234)              0         1
/\top#DUT /SLAVE_instance/assert__CHK_CMD_to_READ_ADD
                SPI_slave.sv(226)              0         1
/\top#DUT /SLAVE_instance/assert__CHK_CMD_to_WRITE
                SPI_slave.sv(218)              0         1
/\top#DUT /SLAVE_instance/assert__IDLE_to_CHK_CMD
                SPI_slave.sv(208)              0         1
/\top#DUT /SLAVE_instance/assert__read_data_comm
                SPI_slave.sv(195)              0         1
/\top#DUT /SLAVE_instance/assert__read_address_comm
                SPI_slave.sv(187)              0         1
/\top#DUT /SLAVE_instance/assert__write_data_comm
                SPI_slave.sv(179)              0         1
/\top#DUT /SLAVE_instance/assert__write_address_comm
                SPI_slave.sv(171)              0         1
/\top#DUT /SLAVE_instance/assert__sync_reset
                SPI_slave.sv(143)              0         1
Branch Coverage:
   Enabled Coverage            Bins      Hits    Misses  Coverage
   ----------------            ----      ----    ------  --------
   Branches                     37        37        0   100.00%

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

Branch Coverage for instance /\top#DUT /SLAVE_instance

   Line        Item                    Count     Source
   ----        ----                    -----     ------
  File SPI_slave.sv
------------------------------------IF Branch------------------------------------
```

```
   20                          8602    Count coming in to IF
   20          1                600       if (~rst_n) begin

   23          1                8002       else begin

Branch totals: 2 hits of 2 branches = 100.00%

-----------------------------------CASE Branch-----------------------------------
   29                         22543    Count coming in to CASE
   30          1                4119          IDLE : begin

   36          1                3265          CHK_CMD : begin

   50          1                7770          WRITE : begin

   56          1                3791          READ_ADD : begin

   62          1                3598          READ_DATA : begin

Branch totals: 5 hits of 5 branches = 100.00%

-----------------------------------IF Branch-----------------------------------
   31                          4119    Count coming in to IF
   31          1                1656             if (SS_n)

   33          1                2463             else

Branch totals: 2 hits of 2 branches = 100.00%

-----------------------------------IF Branch-----------------------------------
   37                          3265    Count coming in to IF
   39          1                3265             else begin

Branch totals: 1 hit of 1 branch = 100.00%

-----------------------------------IF Branch-----------------------------------
   40                          3265    Count coming in to IF
   40          1                1756                if (~MOSI)

   42          1                1509                else begin

Branch totals: 2 hits of 2 branches = 100.00%

-----------------------------------IF Branch-----------------------------------
```

```
   43                          1509     Count coming in to IF
   43          1                1127                  if (!received_address)

   45          1                 382                 else

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
   51                          7770     Count coming in to IF
   51          1                1017            if (SS_n)

   53          1                6753            else

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
   57                          3791     Count coming in to IF
   57          1                 400           if (SS_n)

   59          1                3391            else

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
   63                          3598     Count coming in to IF
   63          1                 239           if (SS_n)

   65          1                3359            else

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
   74                         30001     Count coming in to IF
   74          1                 600       if (~rst_n) begin

   80          1               29401       else begin

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------CASE Branch------------------------------------
   81                         29401     Count coming in to CASE
   82          1                2169            IDLE : begin

   85          1                2128            CHK_CMD : begin
```

```
   88          1                13381            WRITE : begin

   97          1                 5277            READ_ADD : begin

   107         1                 6446            READ_DATA : begin

Branch totals: 5 hits of 5 branches = 100.00%

------------------------------------IF Branch------------------------------------
   89                           13381    Count coming in to IF
   89          1                11369                if (counter > 0) begin

   93          1                 2012                else begin

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
   98                            5277    Count coming in to IF
   98          1                 4487                if (counter > 0) begin

   102         1                  790                else begin

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
   108                           6446    Count coming in to IF
   108         1                 2552                if (tx_valid) begin

   118         1                 3894                else begin

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
   110                           2552    Count coming in to IF
   110         1                 2079                if (counter > 0) begin

   114         1                  473                else begin

Branch totals: 2 hits of 2 branches = 100.00%

------------------------------------IF Branch------------------------------------
   119                           3894    Count coming in to IF
   119         1                 3316                if (counter > 0 && ~rx_valid) begin
```

```
   123          1                  578                    else begin

Branch totals: 2 hits of 2 branches = 100.00%


Condition Coverage:
   Enabled Coverage            Bins   Covered    Misses  Coverage
   ----------------           ----     ----    ------  --------
   Conditions                    5        5        0   100.00%

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

Condition Coverage for instance /\top#DUT /SLAVE_instance –

  File SPI_slave.sv
----------------Focused Condition View-------------------
Line       89 Item    1  (counter > 0)
Condition totals: 1 of 1 input term covered = 100.00%

    Input Term   Covered  Reason for no coverage   Hint
   -----------  --------  -----------------------  --------------
  (counter > 0)        Y

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

----------------Focused Condition View-------------------
Line       98 Item    1  (counter > 0)
Condition totals: 1 of 1 input term covered = 100.00%

    Input Term   Covered  Reason for no coverage   Hint
   -----------  --------  -----------------------  --------------
  (counter > 0)        Y

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

----------------Focused Condition View-------------------
```

```
Line       110 Item    1  (counter > 0)
Condition totals: 1 of 1 input term covered = 100.00%

    Input Term   Covered  Reason for no coverage   Hint
   -----------  --------  -----------------------  --------------
  (counter > 0)        Y

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

----------------Focused Condition View-------------------
Line       119 Item    1  ((counter > 0) && ~rx_valid)
Condition totals: 2 of 2 input terms covered = 100.00%

    Input Term   Covered  Reason for no coverage   Hint
   -----------  --------  -----------------------  --------------
  (counter > 0)        Y
      rx_valid         Y

    Rows:       Hits  FEC Target           Non-masking condition(s)
 ---------  ---------  --------------------  -------------------------
  Row   1:          1  (counter > 0)_0       -
  Row   2:          1  (counter > 0)_1       ~rx_valid
  Row   3:          1  rx_valid_0           (counter > 0)
  Row   4:          1  rx_valid_1           (counter > 0)



Directive Coverage:
   Directives                  12       12        0   100.00%

DIRECTIVE COVERAGE:
Name                                Design Design   Lang File(Line)      Hits Status
                                    Unit   UnitType
/\top#DUT /SLAVE_instance/cover__READ_DATA_to_IDLE
                                    SLAVE  Verilog  SVA  SPI_slave.sv(265)
                                                                  227 Covered
/\top#DUT /SLAVE_instance/cover__READ_ADD_to_IDLE
                                    SLAVE  Verilog  SVA  SPI_slave.sv(255)
                                                                  382 Covered
/\top#DUT /SLAVE_instance/cover__WRITE_to_IDLE
```

```
                    SLAVE  Verilog  SVA  SPI_slave.sv(245)
                                          972 Covered
/\top#DUT /SLAVE_instance/cover__CHK_CMD_to_READ_DATA
                    SLAVE  Verilog  SVA  SPI_slave.sv(235)
                                          366 Covered
/\top#DUT /SLAVE_instance/cover__CHK_CMD_to_READ_ADD
                    SLAVE  Verilog  SVA  SPI_slave.sv(227)
                                          482 Covered
/\top#DUT /SLAVE_instance/cover__CHK_CMD_to_WRITE
                    SLAVE  Verilog  SVA  SPI_slave.sv(219)
                                         1238 Covered
/\top#DUT /SLAVE_instance/cover__IDLE_to_CHK_CMD
                    SLAVE  Verilog  SVA  SPI_slave.sv(209)
                                         2128 Covered
/\top#DUT /SLAVE_instance/cover__read_data_comm
                    SLAVE  Verilog  SVA  SPI_slave.sv(196)
                                          234 Covered
/\top#DUT /SLAVE_instance/cover__read_address_comm
                    SLAVE  Verilog  SVA  SPI_slave.sv(188)
                                          209 Covered
/\top#DUT /SLAVE_instance/cover__write_data_comm
                    SLAVE  Verilog  SVA  SPI_slave.sv(180)
                                          246 Covered
/\top#DUT /SLAVE_instance/cover__write_address_comm
                    SLAVE  Verilog  SVA  SPI_slave.sv(172)
                                          504 Covered
/\top#DUT /SLAVE_instance/cover__sync_reset
                    SLAVE  Verilog  SVA  SPI_slave.sv(144)
                                          600 Covered
FSM Coverage:
   Enabled Coverage          Bins     Hits    Misses  Coverage
   ----------------          ----     ----    ------  --------
   FSM States                   5        5        0   100.00%
   FSM Transitions              7        7        0   100.00%

================================FSM Details================================

FSM Coverage for instance /\top#DUT /SLAVE_instance –

FSM_ID: cs
   Current State Object : cs
   ----------------------
   State Value MapInfo :
   ---------------------
```

```
Line        State Name          Value
----        ----------          -----
  30            IDLE              0
  36          CHK_CMD               2
  62         READ_DATA              4
  56          READ_ADD              3
  50            WRITE               1
   Covered States :
   ----------------
              State        Hit_count
              -----        ---------
               IDLE           2219
             CHK_CMD            2169
            READ_DATA            743
            READ_ADD             971
              WRITE            2500
   Covered Transitions :
   ---------------------
Line        Trans_ID        Hit_count        Transition
----        --------        ---------        ----------
  34            0             2169        IDLE -> CHK_CMD
  46            1              377        CHK_CMD -> READ_DATA
  44            2              489        CHK_CMD -> READ_ADD
  41            3             1262        CHK_CMD -> WRITE
  64            5              377        READ_DATA -> IDLE
  58            6              489        READ_ADD -> IDLE
  52            7             1261        WRITE -> IDLE


   Summary                   Bins     Hits    Misses  Coverage
   -------                   ----     ----    ------  --------
      FSM States               5        5        0   100.00%
      FSM Transitions          7        7        0   100.00%
Statement Coverage:
   Enabled Coverage            Bins     Hits    Misses  Coverage
   ----------------            ----     ----    ------  --------
   Statements                  37       37        0   100.00%

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

Statement Coverage for instance /\top#DUT /SLAVE_instance –

   Line        Item                  Count     Source
```

```
----        ----                    -----     ------
File SPI_slave.sv
565                                 module SLAVE
                                    (MOSI,MISO,SS_n,clk,rst_n,rx_data,rx_valid,tx_data,tx_valid);

2

3                                   localparam IDLE      = 3'b000;

4                                   localparam WRITE     = 3'b001;

5                                   localparam CHK_CMD   = 3'b010;

6                                   localparam READ_ADD  = 3'b011;

7                                   localparam READ_DATA = 3'b100;

8

9                                   input MOSI, clk, rst_n, SS_n, tx_valid;

10                                   input [7:0] tx_data;

11                                   output reg [9:0] rx_data;

12                                   output reg rx_valid, MISO;

13

14                                   reg [3:0] counter;

15                                   reg received_address;

16

17                                   reg [2:0] cs, ns;

18

19          1                   8602     always @(posedge clk) begin

20                                      if (~rst_n) begin

21          1                    600           cs <= IDLE;
```

```
22                                  end
23                                  else begin
24          1                8002          cs <= ns;
25                                  end
26                              end
27
28          1               22543     always @(*) begin
29                                  case (cs)
30                                     IDLE : begin
31                                        if (SS_n)
32          1                1656                ns = IDLE;
33                                        else
34          1                2463                ns = CHK_CMD;
35                                     end
36                                     CHK_CMD : begin
37                                        if (SS_n)
38                                           ns = IDLE;
39                                        else begin
40                                           if (~MOSI)
41          1                1756                   ns = WRITE;
42                                           else begin
43                                              if (!received_address)
```

```
44      1       1127            ns = READ_ADD;
45                          else
46      1       382             ns = READ_DATA;
47                        end
48                      end
49                    end
50                    WRITE : begin
51                      if (SS_n)
52      1       1017            ns = IDLE;
53                      else
54      1       6753            ns = WRITE;
55                    end
56                    READ_ADD : begin
57                      if (SS_n)
58      1       400             ns = IDLE;
59                      else
60      1       3391            ns = READ_ADD;
61                    end
62                    READ_DATA : begin
63                      if (SS_n)
64      1       239             ns = IDLE;
65                      else
```

```
66      1          3359            ns = READ_DATA;
67                         end
68
69                         default : ns = IDLE;
70                     endcase
71                 end
72
73      1          30001    always @(posedge clk) begin
74                     if (~rst_n) begin
75      1           600         rx_data <= 0;
76      1           600         rx_valid <= 0;
77      1           600         received_address <= 0;
78      1           600         MISO <= 0;
79                     end
80                     else begin
81                         case (cs)
82                             IDLE : begin
83      1           2169            rx_valid <= 0;
84                             end
85                             CHK_CMD : begin
86      1           2128            counter <= 10;
87                             end
```

```
88                                  WRITE : begin
89                                     if (counter > 0) begin
90      1           11369                  rx_data[counter-1] <= MOSI;
91      1           11369                  counter <= counter – 1;
92                                     end
93                                     else begin
94      1            2012                  rx_valid <= 1;
95                                     end
96                                  end
97                                  READ_ADD : begin
98                                     if (counter > 0) begin
99      1            4487                  rx_data[counter-1] <= MOSI;
100     1            4487                  counter <= counter – 1;
101                                     end
102                                     else begin
103     1             790                  rx_valid <= 1;
104     1             790                  received_address <= 1;
105                                     end
106                                  end
107                                  READ_DATA : begin
108                                     if (tx_valid) begin
109     1            2552                  rx_valid <= 0;
```

```
110                                     if (counter > 0) begin
111        1            2079                 MISO <= tx_data[counter-1];
112        1            2079                 counter <= counter – 1;
113                                     end
114                                     else begin
115        1             473                 received_address <= 0;
116                                     end
117                                 end
118                                 else begin
119                                     if (counter > 0 && ~rx_valid) begin
120        1            3316                 rx_data[counter-1] <= MOSI;
121        1            3316                 counter <= counter – 1;
122                                     end
123                                     else begin
124        1             578                 rx_valid <= 1;
125        1             578                 counter <= 8;


Toggle Coverage:
   Enabled Coverage          Bins     Hits    Misses  Coverage
   ----------------          ----     ----    ------  --------
   Toggles                     72       72         0   100.00%

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

Toggle Coverage for instance /\top#DUT /SLAVE_instance –

                              Node      1H->0L      0L->1H  “Coverage”
```

```
                                        ----------------------------------------
                                        MISO          1          1      100.00
                                        MOSI          1          1      100.00
                                        SS_n          1          1      100.00
                                        clk           1          1      100.00
                                  counter[3-0]        1          1      100.00
                                      cs[2-0]         1          1      100.00
                                      ns[2-0]         1          1      100.00
                             received_address         1          1      100.00
                                       rst_n          1          1      100.00
                                  rx_data[9-0]        1          1      100.00
                                     rx_valid         1          1      100.00
                                  tx_data[0-7]        1          1      100.00
                                     tx_valid         1          1      100.00

Total Node Count     =        36
Toggled Node Count   =        36
Untoggled Node Count =         0

Toggle Coverage      =     100.00% (72 of 72 bins)

================================================================================
=== Instance: /\top#DUT /sva_WRAPPER_inst
=== Design Unit: work.SPI_wrapper_sva
================================================================================

Assertion Coverage:
   Assertions                    2        2        0   100.00%
--------------------------------------------------------------------------------
Name                File(Line)                 Failure      Pass
                                                 Count       Count
--------------------------------------------------------------------------------
/\top#DUT /sva_WRAPPER_inst/assert__MISO_stable_check
                 SPI_wrapper_sva.sv(14)            0         1
/\top#DUT /sva_WRAPPER_inst/assert__reset_check
                 SPI_wrapper_sva.sv(13)            0         1

Directive Coverage:
   Directives                    2        2        0   100.00%

DIRECTIVE COVERAGE:
--------------------------------------------------------------------------------
Name                                Design Design   Lang File(Line)      Hits Status
                                    Unit   UnitType
--------------------------------------------------------------------------------
/\top#DUT /sva_WRAPPER_inst/cover__MISO_stable_check
                                    SPI_wrapper_sva Verilog  SVA  SPI_wrapper_sva.sv(17)
                                                                      1304 Covered
/\top#DUT /sva_WRAPPER_inst/cover__reset_check
```

```
                    SPI_wrapper_sva Verilog  SVA  SPI_wrapper_sva.sv(16)
                                          600 Covered
Toggle Coverage:
   Enabled Coverage            Bins      Hits    Misses  Coverage
   ----------------           ----      ----    ------  --------
   Toggles                      10        10        0   100.00%

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

Toggle Coverage for instance /\top#DUT /sva_WRAPPER_inst –

                                Node      1H->0L      0L->1H  "Coverage"
                                ---------------------------------------
                                MISO          1           1      100.00
                                MOSI          1           1      100.00
                                SS_n          1           1      100.00
                                 clk          1           1      100.00
                                rst_n         1           1      100.00

Total Node Count     =         5
Toggled Node Count   =         5
Untoggled Node Count =         0

Toggle Coverage      =     100.00% (10 of 10 bins)

================================================================================
=== Instance: /\top#DUT /sva_RAM_inst
=== Design Unit: work.RAM_SVA
================================================================================

Assertion Coverage:
   Assertions                    7         7         0   100.00%
--------------------------------------------------------------------------------
Name                File(Line)                  Failure      Pass
                                                Count        Count
--------------------------------------------------------------------------------
/\top#DUT /sva_RAM_inst/assert__read_data_eventually_after_address
                 RAM_assertions.sv(63)              0         1
/\top#DUT /sva_RAM_inst/assert__write_data_eventually_after_address
                 RAM_assertions.sv(55)              0         1
/\top#DUT /sva_RAM_inst/assert__tx_valid_on_seq
                 RAM_assertions.sv(47)              0         1
/\top#DUT /sva_RAM_inst/assert__tx_valid_off_seq_of_read_add
                 RAM_assertions.sv(39)              0         1
/\top#DUT /sva_RAM_inst/assert__tx_valid_off_seq_of_wrtie_data
                 RAM_assertions.sv(31)              0         1
/\top#DUT /sva_RAM_inst/assert__tx_valid_off_seq_of_wrtie_add
```

```
                RAM_assertions.sv(23)            0        1
/\top#DUT /sva_RAM_inst/assert__sync_reset
                RAM_assertions.sv(15)            0        1

Directive Coverage:
   Directives                  7        7        0   100.00%

DIRECTIVE COVERAGE:
Name                              Design Design   Lang File(Line)      Hits Status
                                  Unit   UnitType
/\top#DUT /sva_RAM_inst/cover__read_data_eventually_after_address
                                  RAM_SVA Verilog  SVA  RAM_assertions.sv(64)
                                                        700 Covered
/\top#DUT /sva_RAM_inst/cover__write_data_eventually_after_address
                                  RAM_SVA Verilog  SVA  RAM_assertions.sv(56)
                                                        936 Covered
/\top#DUT /sva_RAM_inst/cover__tx_valid_on_seq
                                  RAM_SVA Verilog  SVA  RAM_assertions.sv(48)
                                                        340 Covered
/\top#DUT /sva_RAM_inst/cover__tx_valid_off_seq_of_read_add
                                  RAM_SVA Verilog  SVA  RAM_assertions.sv(40)
                                                        759 Covered
/\top#DUT /sva_RAM_inst/cover__tx_valid_off_seq_of_wrtie_data
                                  RAM_SVA Verilog  SVA  RAM_assertions.sv(32)
                                                        648 Covered
/\top#DUT /sva_RAM_inst/cover__tx_valid_off_seq_of_wrtie_add
                                  RAM_SVA Verilog  SVA  RAM_assertions.sv(24)
                                                       1275 Covered
/\top#DUT /sva_RAM_inst/cover__sync_reset
                                  RAM_SVA Verilog  SVA  RAM_assertions.sv(16)
                                                        600 Covered
Toggle Coverage:
   Enabled Coverage            Bins     Hits    Misses  Coverage
   ----------------            ----     ----    ------  --------
   Toggles                      44       44        0   100.00%

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

Toggle Coverage for instance /\top#DUT /sva_RAM_inst –

                                 Node      1H->0L      0L->1H  "Coverage"
                                 ----------------------------------------
                                  clk          1          1      100.00
```

```
                din[0-9]          1          1      100.00
               dout[0-7]          1          1      100.00
                  rst_n           1          1      100.00
               rx_valid           1          1      100.00
               tx_valid           1          1      100.00

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

Toggle Coverage      =     100.00% (44 of 44 bins)

================================================================================
=== Instance: /\top#DUT
=== Design Unit: work.WRAPPER
================================================================================
Toggle Coverage:
   Enabled Coverage            Bins      Hits    Misses  Coverage
   ----------------            ----      ----    ------  --------
   Toggles                      50        50         0   100.00%

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

Toggle Coverage for instance /\top#DUT  --

                     Node      1H->0L      0L->1H  "Coverage"
                     ---------------------------------------
                     MISO          1          1      100.00
                     MOSI          1          1      100.00
                     SS_n          1          1      100.00
                      clk          1          1      100.00
                    rst_n          1          1      100.00
          rx_data_din[0-9]         1          1      100.00
                 rx_valid          1          1      100.00
         tx_data_dout[0-7]         1          1      100.00
                 tx_valid          1          1      100.00

Total Node Count     =        25
Toggled Node Count   =        25
Untoggled Node Count =         0

Toggle Coverage      =     100.00% (50 of 50 bins)


DIRECTIVE COVERAGE:
```

```
Name                                Design Design   Lang File(Line)      Hits Status
                                    Unit   UnitType
/\top#DUT /SLAVE_instance/cover__READ_DATA_to_IDLE
                                    SLAVE  Verilog  SVA  SPI_slave.sv(265)
                                                         227 Covered
/\top#DUT /SLAVE_instance/cover__READ_ADD_to_IDLE
                                    SLAVE  Verilog  SVA  SPI_slave.sv(255)
                                                         382 Covered
/\top#DUT /SLAVE_instance/cover__WRITE_to_IDLE
                                    SLAVE  Verilog  SVA  SPI_slave.sv(245)
                                                         972 Covered
/\top#DUT /SLAVE_instance/cover__CHK_CMD_to_READ_DATA
                                    SLAVE  Verilog  SVA  SPI_slave.sv(235)
                                                         366 Covered
/\top#DUT /SLAVE_instance/cover__CHK_CMD_to_READ_ADD
                                    SLAVE  Verilog  SVA  SPI_slave.sv(227)
                                                         482 Covered
/\top#DUT /SLAVE_instance/cover__CHK_CMD_to_WRITE
                                    SLAVE  Verilog  SVA  SPI_slave.sv(219)
                                                        1238 Covered
/\top#DUT /SLAVE_instance/cover__IDLE_to_CHK_CMD
                                    SLAVE  Verilog  SVA  SPI_slave.sv(209)
                                                        2128 Covered
/\top#DUT /SLAVE_instance/cover__read_data_comm
                                    SLAVE  Verilog  SVA  SPI_slave.sv(196)
                                                         234 Covered
/\top#DUT /SLAVE_instance/cover__read_address_comm
                                    SLAVE  Verilog  SVA  SPI_slave.sv(188)
                                                         209 Covered
/\top#DUT /SLAVE_instance/cover__write_data_comm
                                    SLAVE  Verilog  SVA  SPI_slave.sv(180)
                                                         246 Covered
/\top#DUT /SLAVE_instance/cover__write_address_comm
                                    SLAVE  Verilog  SVA  SPI_slave.sv(172)
                                                         504 Covered
/\top#DUT /SLAVE_instance/cover__sync_reset
                                    SLAVE  Verilog  SVA  SPI_slave.sv(144)
                                                         600 Covered
/\top#DUT /sva_WRAPPER_inst/cover__MISO_stable_check
                                    SPI_wrapper_sva Verilog  SVA  SPI_wrapper_sva.sv(17)
                                                        1304 Covered
/\top#DUT /sva_WRAPPER_inst/cover__reset_check
                                    SPI_wrapper_sva Verilog  SVA  SPI_wrapper_sva.sv(16)
                                                         600 Covered
```

```
/\top#DUT /sva_RAM_inst/cover__read_data_eventually_after_address
                    RAM_SVA Verilog  SVA  RAM_assertions.sv(64)
                                          700 Covered
/\top#DUT /sva_RAM_inst/cover__write_data_eventually_after_address
                    RAM_SVA Verilog  SVA  RAM_assertions.sv(56)
                                          936 Covered
/\top#DUT /sva_RAM_inst/cover__tx_valid_on_seq
                    RAM_SVA Verilog  SVA  RAM_assertions.sv(48)
                                          340 Covered
/\top#DUT /sva_RAM_inst/cover__tx_valid_off_seq_of_read_add
                    RAM_SVA Verilog  SVA  RAM_assertions.sv(40)
                                          759 Covered
/\top#DUT /sva_RAM_inst/cover__tx_valid_off_seq_of_wrtie_data
                    RAM_SVA Verilog  SVA  RAM_assertions.sv(32)
                                          648 Covered
/\top#DUT /sva_RAM_inst/cover__tx_valid_off_seq_of_wrtie_add
                    RAM_SVA Verilog  SVA  RAM_assertions.sv(24)
                                         1275 Covered
/\top#DUT /sva_RAM_inst/cover__sync_reset
                    RAM_SVA Verilog  SVA  RAM_assertions.sv(16)
                                          600 Covered

TOTAL DIRECTIVE COVERAGE: 100.00%  COVERS: 21

ASSERTION RESULTS:
-----------------------------------------------------------------------
Name              File(Line)              Failure     Pass
                                   Count       Count
-----------------------------------------------------------------------
/\top#DUT /SLAVE_instance/assert__READ_DATA_to_IDLE
                SPI_slave.sv(264)              0        1
/\top#DUT /SLAVE_instance/assert__READ_ADD_to_IDLE
                SPI_slave.sv(254)              0        1
/\top#DUT /SLAVE_instance/assert__WRITE_to_IDLE
                SPI_slave.sv(244)              0        1
/\top#DUT /SLAVE_instance/assert__CHK_CMD_to_READ_DATA
                SPI_slave.sv(234)              0        1
/\top#DUT /SLAVE_instance/assert__CHK_CMD_to_READ_ADD
                SPI_slave.sv(226)              0        1
/\top#DUT /SLAVE_instance/assert__CHK_CMD_to_WRITE
                SPI_slave.sv(218)              0        1
/\top#DUT /SLAVE_instance/assert__IDLE_to_CHK_CMD
                SPI_slave.sv(208)              0        1
/\top#DUT /SLAVE_instance/assert__read_data_comm
                SPI_slave.sv(195)              0        1
```

```
/\top#DUT /SLAVE_instance/assert__read_address_comm
                SPI_slave.sv(187)              0        1
/\top#DUT /SLAVE_instance/assert__write_data_comm
                SPI_slave.sv(179)              0        1
/\top#DUT /SLAVE_instance/assert__write_address_comm
                SPI_slave.sv(171)              0        1
/\top#DUT /SLAVE_instance/assert__sync_reset
                SPI_slave.sv(143)              0        1
/\top#DUT /sva_WRAPPER_inst/assert__MISO_stable_check
                SPI_wrapper_sva.sv(14)          0        1
/\top#DUT /sva_WRAPPER_inst/assert__reset_check
                SPI_wrapper_sva.sv(13)          0        1
/\top#DUT /sva_RAM_inst/assert__read_data_eventually_after_address
                RAM_assertions.sv(63)           0        1
/\top#DUT /sva_RAM_inst/assert__write_data_eventually_after_address
                RAM_assertions.sv(55)           0        1
/\top#DUT /sva_RAM_inst/assert__tx_valid_on_seq
                RAM_assertions.sv(47)           0        1
/\top#DUT /sva_RAM_inst/assert__tx_valid_off_seq_of_read_add
                RAM_assertions.sv(39)           0        1
/\top#DUT /sva_RAM_inst/assert__tx_valid_off_seq_of_wrtie_data
                RAM_assertions.sv(31)           0        1
/\top#DUT /sva_RAM_inst/assert__tx_valid_off_seq_of_wrtie_add
                RAM_assertions.sv(23)           0        1
/\top#DUT /sva_RAM_inst/assert__sync_reset
                RAM_assertions.sv(15)           0        1

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

Functional Coverage Report

```
Coverage Report by instance with details

================================================================================
=== Instance: /top/DUT/SLAVE_instance
=== Design Unit: work.SLAVE
================================================================================

Directive Coverage:
   Directives                 12       12        0   100.00%

DIRECTIVE COVERAGE:
--------------------------------------------------------------------------------
Name                              Design Design   Lang File(Line)     Hits Status
                                  Unit   UnitType
--------------------------------------------------------------------------------
```

```
/top/DUT/SLAVE_instance/cover__READ_DATA_to_IDLE
                        SLAVE  Verilog  SVA  SPI_slave.sv(265)
                                                 264 Covered
/top/DUT/SLAVE_instance/cover__READ_ADD_to_IDLE
                        SLAVE  Verilog  SVA  SPI_slave.sv(255)
                                                 400 Covered
/top/DUT/SLAVE_instance/cover__WRITE_to_IDLE
                        SLAVE  Verilog  SVA  SPI_slave.sv(245)
                                                 931 Covered
/top/DUT/SLAVE_instance/cover__CHK_CMD_to_READ_DATA
                        SLAVE  Verilog  SVA  SPI_slave.sv(235)
                                                 381 Covered
/top/DUT/SLAVE_instance/cover__CHK_CMD_to_READ_ADD
                        SLAVE  Verilog  SVA  SPI_slave.sv(227)
                                                 497 Covered
/top/DUT/SLAVE_instance/cover__CHK_CMD_to_WRITE
                        SLAVE  Verilog  SVA  SPI_slave.sv(219)
                                                1186 Covered
/top/DUT/SLAVE_instance/cover__IDLE_to_CHK_CMD
                        SLAVE  Verilog  SVA  SPI_slave.sv(209)
                                                2102 Covered
/top/DUT/SLAVE_instance/cover__read_data_comm
                        SLAVE  Verilog  SVA  SPI_slave.sv(196)
                                                 266 Covered
/top/DUT/SLAVE_instance/cover__read_address_comm
                        SLAVE  Verilog  SVA  SPI_slave.sv(188)
                                                 246 Covered
/top/DUT/SLAVE_instance/cover__write_data_comm
                        SLAVE  Verilog  SVA  SPI_slave.sv(180)
                                                 243 Covered
/top/DUT/SLAVE_instance/cover__write_address_comm
                        SLAVE  Verilog  SVA  SPI_slave.sv(172)
                                                 481 Covered
/top/DUT/SLAVE_instance/cover__sync_reset
                        SLAVE  Verilog  SVA  SPI_slave.sv(144)
                                                 565 Covered

================================================================
=== Instance: /top/DUT/sva_WRAPPER_inst
=== Design Unit: work.SPI_wrapper_sva
================================================================

Directive Coverage:
   Directives                 2       2       0   100.00%
```

```
DIRECTIVE COVERAGE:
Name                          Design Design   Lang File(Line)      Hits Status
                              Unit   UnitType
/top/DUT/sva_WRAPPER_inst/cover__MISO_stable_check
                              SPI_wrapper_sva Verilog  SVA  SPI_wrapper_sva.sv(17)
                                                     1334 Covered
/top/DUT/sva_WRAPPER_inst/cover__reset_check
                              SPI_wrapper_sva Verilog  SVA  SPI_wrapper_sva.sv(16)
                                                      565 Covered

=================================================================================
=== Instance: /top/DUT/sva_RAM_inst
=== Design Unit: work.RAM_SVA
=================================================================================

Directive Coverage:
   Directives                 7        7        0   100.00%

DIRECTIVE COVERAGE:
Name                          Design Design   Lang File(Line)      Hits Status
                              Unit   UnitType
/top/DUT/sva_RAM_inst/cover__read_data_eventually_after_address
                              RAM_SVA Verilog  SVA  RAM_assertions.sv(64)
                                                      730 Covered
/top/DUT/sva_RAM_inst/cover__write_data_eventually_after_address
                              RAM_SVA Verilog  SVA  RAM_assertions.sv(56)
                                                      886 Covered
/top/DUT/sva_RAM_inst/cover__tx_valid_on_seq
                              RAM_SVA Verilog  SVA  RAM_assertions.sv(48)
                                                      428 Covered
/top/DUT/sva_RAM_inst/cover__tx_valid_off_seq_of_read_add
                              RAM_SVA Verilog  SVA  RAM_assertions.sv(40)
                                                      791 Covered
/top/DUT/sva_RAM_inst/cover__tx_valid_off_seq_of_wrtie_data
                              RAM_SVA Verilog  SVA  RAM_assertions.sv(32)
                                                      627 Covered
/top/DUT/sva_RAM_inst/cover__tx_valid_off_seq_of_wrtie_add
                              RAM_SVA Verilog  SVA  RAM_assertions.sv(24)
                                                     1219 Covered
/top/DUT/sva_RAM_inst/cover__sync_reset  RAM_SVA Verilog  SVA  RAM_assertions.sv(16)
                                                      565 Covered

=================================================================================
=== Instance: /top/SLAVE_instance
=== Design Unit: work.SLAVE
=================================================================================
```

```
Directive Coverage:
   Directives                 12       12        0   100.00%

DIRECTIVE COVERAGE:
Name                              Design Design   Lang File(Line)      Hits Status
                                  Unit   UnitType
/top/SLAVE_instance/cover__READ_DATA_to_IDLE
                                  SLAVE  Verilog  SVA  SPI_slave.sv(265)
                                                       264 Covered
/top/SLAVE_instance/cover__READ_ADD_to_IDLE
                                  SLAVE  Verilog  SVA  SPI_slave.sv(255)
                                                       400 Covered
/top/SLAVE_instance/cover__WRITE_to_IDLE SLAVE  Verilog  SVA  SPI_slave.sv(245)
                                                       931 Covered
/top/SLAVE_instance/cover__CHK_CMD_to_READ_DATA
                                  SLAVE  Verilog  SVA  SPI_slave.sv(235)
                                                       381 Covered
/top/SLAVE_instance/cover__CHK_CMD_to_READ_ADD
                                  SLAVE  Verilog  SVA  SPI_slave.sv(227)
                                                       497 Covered
/top/SLAVE_instance/cover__CHK_CMD_to_WRITE
                                  SLAVE  Verilog  SVA  SPI_slave.sv(219)
                                                      1186 Covered
/top/SLAVE_instance/cover__IDLE_to_CHK_CMD
                                  SLAVE  Verilog  SVA  SPI_slave.sv(209)
                                                      2102 Covered
/top/SLAVE_instance/cover__read_data_comm
                                  SLAVE  Verilog  SVA  SPI_slave.sv(196)
                                                       266 Covered
/top/SLAVE_instance/cover__read_address_comm
                                  SLAVE  Verilog  SVA  SPI_slave.sv(188)
                                                       246 Covered
/top/SLAVE_instance/cover__write_data_comm
                                  SLAVE  Verilog  SVA  SPI_slave.sv(180)
                                                       243 Covered
/top/SLAVE_instance/cover__write_address_comm
                                  SLAVE  Verilog  SVA  SPI_slave.sv(172)
                                                       481 Covered
/top/SLAVE_instance/cover__sync_reset    SLAVE  Verilog  SVA  SPI_slave.sv(144)
                                                       565 Covered

=================================================================================
=== Instance: /RAM_collector_pkg
```

```
=== Design Unit: work.RAM_collector_pkg
=========================================================================

Covergroup Coverage:
    Covergroups                  1        na        na   100.00%
        Coverpoints/Crosses      5        na        na        na
            Covergroup Bins     15        15         0   100.00%
-------------------------------------------------------------------------
Covergroup                                Metric      Goal      Bins    Status
                                                                
-------------------------------------------------------------------------
 TYPE /RAM_collector_pkg/RAM_collector/RAM_cvr       100.00%       100         -    Covered
    covered/total bins:                          15        15         -
    missing/total bins:                           0        15         -
    % Hit:                                 100.00%       100         -
    Coverpoint din_cp                      100.00%       100         -    Covered
        covered/total bins:                       6         6         -
        missing/total bins:                       0         6         -
        % Hit:                             100.00%       100         -
    Coverpoint rx_valid_CP                 100.00%       100         -    Covered
        covered/total bins:                       2         2         -
        missing/total bins:                       0         2         -
        % Hit:                             100.00%       100         -
    Coverpoint tx_valid_CP                 100.00%       100         -    Covered
        covered/total bins:                       2         2         -
        missing/total bins:                       0         2         -
        % Hit:                             100.00%       100         -
    Cross din_with_rx                      100.00%       100         -    Covered
        covered/total bins:                       4         4         -
        missing/total bins:                       0         4         -
        % Hit:                             100.00%       100         -
    Cross din_read_with_tx                 100.00%       100         -    Covered
        covered/total bins:                       1         1         -
        missing/total bins:                       0         1         -
        % Hit:                             100.00%       100         -
 Covergroup instance \/RAM_collector_pkg::RAM_collector::RAM_cvr
                                           100.00%       100         -    Covered
    covered/total bins:                          15        15         -
    missing/total bins:                           0        15         -
    % Hit:                                 100.00%       100         -
    Coverpoint din_cp                      100.00%       100         -    Covered
        covered/total bins:                       6         6         -
        missing/total bins:                       0         6         -
        % Hit:                             100.00%       100         -
        bin write_address                     9606         1         -    Covered
        bin write_data                        4866         1         -    Covered
```

```
    bin read_address                          6346        1        -    Covered
    bin read_data                             7434        1        -    Covered
    bin write_data_after_write_address         399        1        -    Covered
    bin read_data_after_read_address           373        1        -    Covered
  Coverpoint rx_valid_CP                    100.00%      100        -    Covered
    covered/total bins:                          2        2        -
    missing/total bins:                          0        2        -
    % Hit:                                 100.00%      100        -
    bin high                                  3368        1        -    Covered
    bin low                                  24884        1        -    Covered
  Coverpoint tx_valid_CP                    100.00%      100        -    Covered
    covered/total bins:                          2        2        -
    missing/total bins:                          0        2        -
    % Hit:                                 100.00%      100        -
    bin high                                  6348        1        -    Covered
    bin low                                  21904        1        -    Covered
  Cross din_with_rx                         100.00%      100        -    Covered
    covered/total bins:                          4        4        -
    missing/total bins:                          0        4        -
    % Hit:                                 100.00%      100        -
    Auto, Default and User Defined Bins:
      bin <read_data,high>                     627        1        -    Covered
      bin <write_data,high>                    657        1        -    Covered
      bin <read_address,high>                  827        1        -    Covered
      bin <write_address,high>                1257        1        -    Covered
    Illegal and Ignore Bins:
      ignore_bin trans_R                       373                 -    Occurred
      ignore_bin trans_W                       399                 -    Occurred
      ignore_bin low_tx                      24884                 -    Occurred
  Cross din_read_with_tx                    100.00%      100        -    Covered
    covered/total bins:                          1        1        -
    missing/total bins:                          0        1        -
    % Hit:                                 100.00%      100        -
    Auto, Default and User Defined Bins:
      bin checked                             3909        1        -    Covered

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

Covergroup Coverage:
  Covergroups                    1       na       na   100.00%
    Coverpoints/Crosses          4       na       na       na
      Covergroup Bins           23       23        0   100.00%
```

```
Covergroup                                  Metric      Goal      Bins    Status
                                                                
TYPE /SPI_slave_collector_pkg/SPI_slave_collector/cvg_group
                                        100.00%       100        -    Covered
  covered/total bins:                          23        23        -
  missing/total bins:                          0         23        -
  % Hit:                                100.00%       100        -
  Coverpoint rx_data_CP                        100.00%       100        -    Covered
    covered/total bins:                        12        12        -
    missing/total bins:                        0         12        -
    % Hit:                              100.00%       100        -
  Coverpoint SS_n_CP                           100.00%       100        -    Covered
    covered/total bins:                        3         3         -
    missing/total bins:                        0         3         -
    % Hit:                              100.00%       100        -
  Coverpoint MOSI_transitions_CP               100.00%       100        -    Covered
    covered/total bins:                        4         4         -
    missing/total bins:                        0         4         -
    % Hit:                              100.00%       100        -
  Cross SS_n_with_MOSI                         100.00%       100        -    Covered
    covered/total bins:                        4         4         -
    missing/total bins:                        0         4         -
    % Hit:                              100.00%       100        -
Covergroup instance \/SPI_slave_collector_pkg::SPI_slave_collector::cvg_group
                                        100.00%       100        -    Covered
  covered/total bins:                          23        23        -
  missing/total bins:                          0         23        -
  % Hit:                                100.00%       100        -
  Coverpoint rx_data_CP                        100.00%       100        -    Covered
    covered/total bins:                        12        12        -
    missing/total bins:                        0         12        -
    % Hit:                              100.00%       100        -
    bin write_address                          9475      1         -    Covered
    bin write_data                             4866      1         -    Covered
    bin read_address                           6346      1         -    Covered
    bin read_data                              7434      1         -    Covered
    bin trans1[0=>2]                           216       1         -    Covered
    bin trans1[0=>1]                           399       1         -    Covered
    bin trans2[1=>3]                           67        1         -    Covered
    bin trans2[1=>0]                           375       1         -    Covered
    bin trans3[2=>3]                           373       1         -    Covered
    bin trans3[2=>0]                           118       1         -    Covered
    bin trans4[3=>2]                           275       1         -    Covered
    bin trans4[3=>1]                           43        1         -    Covered
```

```
    Coverpoint SS_n_CP                    100.00%    100    -    Covered
      covered/total bins:                   3        3      -
      missing/total bins:                   0        3      -
      % Hit:                            100.00%    100      -
      bin full_normal_seq                 976        1      -    Covered
      bin full_read_seq                   280        1      -    Covered
      bin start_comm                     1605        1      -    Covered
    Coverpoint MOSI_transitions_CP            100.00%    100    -    Covered
      covered/total bins:                   4        4      -
      missing/total bins:                   0        4      -
      % Hit:                            100.00%    100      -
      bin write_address                  1930        1      -    Covered
      bin write_data                      407        1      -    Covered
      bin read_address                    557        1      -    Covered
      bin read_data                      1110        1      -    Covered
    Cross SS_n_with_MOSI                  100.00%    100    -    Covered
      covered/total bins:                   4        4      -
      missing/total bins:                   0        4      -
      % Hit:                            100.00%    100      -
      Auto, Default and User Defined Bins:
        bin <start_comm,read_data>          376        1      -    Covered
        bin <start_comm,write_data>         295        1      -    Covered
        bin <start_comm,read_address>       290        1      -    Covered
        bin <start_comm,write_address>      582        1      -    Covered
      Illegal and Ignore Bins:
        illegal_bin full_seq2               0                 -    ZERO
        illegal_bin full_seq1               0                 -    ZERO

COVERGROUP COVERAGE:
------------------------------------------------------------------------------
Covergroup                                Metric     Goal     Bins    Status
                                                                
------------------------------------------------------------------------------
TYPE /RAM_collector_pkg/RAM_collector/RAM_cvr     100.00%    100    -    Covered
    covered/total bins:                    15       15      -
    missing/total bins:                     0       15      -
    % Hit:                            100.00%    100      -
    Coverpoint din_cp                     100.00%    100    -    Covered
      covered/total bins:                   6        6      -
      missing/total bins:                   0        6      -
      % Hit:                            100.00%    100      -
    Coverpoint rx_valid_CP                100.00%    100    -    Covered
      covered/total bins:                   2        2      -
      missing/total bins:                   0        2      -
      % Hit:                            100.00%    100      -
```

```
  Coverpoint tx_valid_CP                    100.00%      100        -   Covered
    covered/total bins:                       2        2        -
    missing/total bins:                       0        2        -
    % Hit:                            100.00%      100        -
  Cross din_with_rx                       100.00%      100        -   Covered
    covered/total bins:                       4        4        -
    missing/total bins:                       0        4        -
    % Hit:                            100.00%      100        -
  Cross din_read_with_tx                    100.00%      100        -   Covered
    covered/total bins:                       1        1        -
    missing/total bins:                       0        1        -
    % Hit:                            100.00%      100        -
Covergroup instance \/RAM_collector_pkg::RAM_collector::RAM_cvr
                                      100.00%      100        -   Covered
  covered/total bins:                        15       15        -
  missing/total bins:                         0       15        -
  % Hit:                            100.00%      100        -
  Coverpoint din_cp                       100.00%      100        -   Covered
    covered/total bins:                       6        6        -
    missing/total bins:                       0        6        -
    % Hit:                            100.00%      100        -
    bin write_address                      9606        1        -   Covered
    bin write_data                         4866        1        -   Covered
    bin read_address                       6346        1        -   Covered
    bin read_data                          7434        1        -   Covered
    bin write_data_after_write_address      399        1        -   Covered
    bin read_data_after_read_address        373        1        -   Covered
  Coverpoint rx_valid_CP                    100.00%      100        -   Covered
    covered/total bins:                       2        2        -
    missing/total bins:                       0        2        -
    % Hit:                            100.00%      100        -
    bin high                               3368        1        -   Covered
    bin low                               24884        1        -   Covered
  Coverpoint tx_valid_CP                    100.00%      100        -   Covered
    covered/total bins:                       2        2        -
    missing/total bins:                       0        2        -
    % Hit:                            100.00%      100        -
    bin high                               6348        1        -   Covered
    bin low                               21904        1        -   Covered
  Cross din_with_rx                       100.00%      100        -   Covered
    covered/total bins:                       4        4        -
    missing/total bins:                       0        4        -
    % Hit:                            100.00%      100        -
    Auto, Default and User Defined Bins:
```

```
        bin <read_data,high>                627        1        -    Covered
        bin <write_data,high>               657        1        -    Covered
        bin <read_address,high>             827        1        -    Covered
        bin <write_address,high>            1257       1        -    Covered
      Illegal and Ignore Bins:
        ignore_bin trans_R                  373                 -    Occurred
        ignore_bin trans_W                  399                 -    Occurred
        ignore_bin low_tx                   24884               -    Occurred
    Cross din_read_with_tx                  100.00%    100      -    Covered
      covered/total bins:                   1          1        -
      missing/total bins:                   0          1        -
      % Hit:                                100.00%    100      -
      Auto, Default and User Defined Bins:
        bin checked                         3909       1        -    Covered
TYPE /SPI_slave_collector_pkg/SPI_slave_collector/cvg_group
                                            100.00%    100      -    Covered
    covered/total bins:                     23         23       -
    missing/total bins:                     0          23       -
    % Hit:                                  100.00%    100      -
    Coverpoint rx_data_CP                   100.00%    100      -    Covered
      covered/total bins:                   12         12       -
      missing/total bins:                   0          12       -
      % Hit:                                100.00%    100      -
    Coverpoint SS_n_CP                      100.00%    100      -    Covered
      covered/total bins:                   3          3        -
      missing/total bins:                   0          3        -
      % Hit:                                100.00%    100      -
    Coverpoint MOSI_transitions_CP          100.00%    100      -    Covered
      covered/total bins:                   4          4        -
      missing/total bins:                   0          4        -
      % Hit:                                100.00%    100      -
    Cross SS_n_with_MOSI                    100.00%    100      -    Covered
      covered/total bins:                   4          4        -
      missing/total bins:                   0          4        -
      % Hit:                                100.00%    100      -
Covergroup instance \/SPI_slave_collector_pkg::SPI_slave_collector::cvg_group
                                            100.00%    100      -    Covered
    covered/total bins:                     23         23       -
    missing/total bins:                     0          23       -
    % Hit:                                  100.00%    100      -
    Coverpoint rx_data_CP                   100.00%    100      -    Covered
      covered/total bins:                   12         12       -
      missing/total bins:                   0          12       -
      % Hit:                                100.00%    100      -
```

```
    bin write_address                    9475     1     -    Covered
    bin write_data                       4866     1     -    Covered
    bin read_address                     6346     1     -    Covered
    bin read_data                        7434     1     -    Covered
    bin trans1[0=>2]                     216      1     -    Covered
    bin trans1[0=>1]                     399      1     -    Covered
    bin trans2[1=>3]                     67       1     -    Covered
    bin trans2[1=>0]                     375      1     -    Covered
    bin trans3[2=>3]                     373      1     -    Covered
    bin trans3[2=>0]                     118      1     -    Covered
    bin trans4[3=>2]                     275      1     -    Covered
    bin trans4[3=>1]                     43       1     -    Covered
  Coverpoint SS_n_CP                     100.00%  100   -    Covered
    covered/total bins:                  3        3     -
    missing/total bins:                  0        3     -
    % Hit:                               100.00%  100   -
    bin full_normal_seq                  976      1     -    Covered
    bin full_read_seq                    280      1     -    Covered
    bin start_comm                       1605     1     -    Covered
  Coverpoint MOSI_transitions_CP         100.00%  100   -    Covered
    covered/total bins:                  4        4     -
    missing/total bins:                  0        4     -
    % Hit:                               100.00%  100   -
    bin write_address                    1930     1     -    Covered
    bin write_data                       407      1     -    Covered
    bin read_address                     557      1     -    Covered
    bin read_data                        1110     1     -    Covered
  Cross SS_n_with_MOSI                   100.00%  100   -    Covered
    covered/total bins:                  4        4     -
    missing/total bins:                  0        4     -
    % Hit:                               100.00%  100   -
    Auto, Default and User Defined Bins:
      bin <start_comm,read_data>         376      1     -    Covered
      bin <start_comm,write_data>        295      1     -    Covered
      bin <start_comm,read_address>      290      1     -    Covered
      bin <start_comm,write_address>     582      1     -    Covered
    Illegal and Ignore Bins:
      illegal_bin full_seq2              0        -     ZERO
      illegal_bin full_seq1              0        -     ZERO

TOTAL COVERGROUP COVERAGE: 100.00%  COVERGROUP TYPES: 2

DIRECTIVE COVERAGE:
Name                    Design Design   Lang File(Line)     Hits Status
```

```
                    Unit   UnitType
/top/DUT/SLAVE_instance/cover__READ_DATA_to_IDLE
                    SLAVE  Verilog  SVA  SPI_slave.sv(265)
                                         264 Covered
/top/DUT/SLAVE_instance/cover__READ_ADD_to_IDLE
                    SLAVE  Verilog  SVA  SPI_slave.sv(255)
                                         400 Covered
/top/DUT/SLAVE_instance/cover__WRITE_to_IDLE
                    SLAVE  Verilog  SVA  SPI_slave.sv(245)
                                         931 Covered
/top/DUT/SLAVE_instance/cover__CHK_CMD_to_READ_DATA
                    SLAVE  Verilog  SVA  SPI_slave.sv(235)
                                         381 Covered
/top/DUT/SLAVE_instance/cover__CHK_CMD_to_READ_ADD
                    SLAVE  Verilog  SVA  SPI_slave.sv(227)
                                         497 Covered
/top/DUT/SLAVE_instance/cover__CHK_CMD_to_WRITE
                    SLAVE  Verilog  SVA  SPI_slave.sv(219)
                                        1186 Covered
/top/DUT/SLAVE_instance/cover__IDLE_to_CHK_CMD
                    SLAVE  Verilog  SVA  SPI_slave.sv(209)
                                        2102 Covered
/top/DUT/SLAVE_instance/cover__read_data_comm
                    SLAVE  Verilog  SVA  SPI_slave.sv(196)
                                         266 Covered
/top/DUT/SLAVE_instance/cover__read_address_comm
                    SLAVE  Verilog  SVA  SPI_slave.sv(188)
                                         246 Covered
/top/DUT/SLAVE_instance/cover__write_data_comm
                    SLAVE  Verilog  SVA  SPI_slave.sv(180)
                                         243 Covered
/top/DUT/SLAVE_instance/cover__write_address_comm
                    SLAVE  Verilog  SVA  SPI_slave.sv(172)
                                         481 Covered
/top/DUT/SLAVE_instance/cover__sync_reset
                    SLAVE  Verilog  SVA  SPI_slave.sv(144)
                                         565 Covered
/top/DUT/sva_WRAPPER_inst/cover__MISO_stable_check
                    SPI_wrapper_sva Verilog  SVA  SPI_wrapper_sva.sv(17)
                                        1334 Covered
/top/DUT/sva_WRAPPER_inst/cover__reset_check
                    SPI_wrapper_sva Verilog  SVA  SPI_wrapper_sva.sv(16)
                                         565 Covered
/top/DUT/sva_RAM_inst/cover__read_data_eventually_after_address
```

```
                    RAM_SVA Verilog  SVA  RAM_assertions.sv(64)
                                        730 Covered
/top/DUT/sva_RAM_inst/cover__write_data_eventually_after_address
                    RAM_SVA Verilog  SVA  RAM_assertions.sv(56)
                                        886 Covered
/top/DUT/sva_RAM_inst/cover__tx_valid_on_seq
                    RAM_SVA Verilog  SVA  RAM_assertions.sv(48)
                                        428 Covered
/top/DUT/sva_RAM_inst/cover__tx_valid_off_seq_of_read_add
                    RAM_SVA Verilog  SVA  RAM_assertions.sv(40)
                                        791 Covered
/top/DUT/sva_RAM_inst/cover__tx_valid_off_seq_of_wrtie_data
                    RAM_SVA Verilog  SVA  RAM_assertions.sv(32)
                                        627 Covered
/top/DUT/sva_RAM_inst/cover__tx_valid_off_seq_of_wrtie_add
                    RAM_SVA Verilog  SVA  RAM_assertions.sv(24)
                                       1219 Covered
/top/DUT/sva_RAM_inst/cover__sync_reset  RAM_SVA Verilog  SVA  RAM_assertions.sv(16)
                                        565 Covered
/top/SLAVE_instance/cover__READ_DATA_to_IDLE
                    SLAVE  Verilog  SVA  SPI_slave.sv(265)
                                        264 Covered
/top/SLAVE_instance/cover__READ_ADD_to_IDLE
                    SLAVE  Verilog  SVA  SPI_slave.sv(255)
                                        400 Covered
/top/SLAVE_instance/cover__WRITE_to_IDLE SLAVE  Verilog  SVA  SPI_slave.sv(245)
                                        931 Covered
/top/SLAVE_instance/cover__CHK_CMD_to_READ_DATA
                    SLAVE  Verilog  SVA  SPI_slave.sv(235)
                                        381 Covered
/top/SLAVE_instance/cover__CHK_CMD_to_READ_ADD
                    SLAVE  Verilog  SVA  SPI_slave.sv(227)
                                        497 Covered
/top/SLAVE_instance/cover__CHK_CMD_to_WRITE
                    SLAVE  Verilog  SVA  SPI_slave.sv(219)
                                       1186 Covered
/top/SLAVE_instance/cover__IDLE_to_CHK_CMD
                    SLAVE  Verilog  SVA  SPI_slave.sv(209)
                                       2102 Covered
/top/SLAVE_instance/cover__read_data_comm
                    SLAVE  Verilog  SVA  SPI_slave.sv(196)
                                        266 Covered
/top/SLAVE_instance/cover__read_address_comm
                    SLAVE  Verilog  SVA  SPI_slave.sv(188)
```

```
                                                    246 Covered
/top/SLAVE_instance/cover__write_data_comm
                              SLAVE  Verilog  SVA  SPI_slave.sv(180)
                                                    243 Covered
/top/SLAVE_instance/cover__write_address_comm
                              SLAVE  Verilog  SVA  SPI_slave.sv(172)
                                                    481 Covered
/top/SLAVE_instance/cover__sync_reset   SLAVE  Verilog  SVA  SPI_slave.sv(144)
                                                    565 Covered

TOTAL DIRECTIVE COVERAGE: 100.00%  COVERS: 33

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

Note: In the functional coverage in this wrapper for RAM part when it was put to be part of the system seems to be not logical to happen so we removed them:

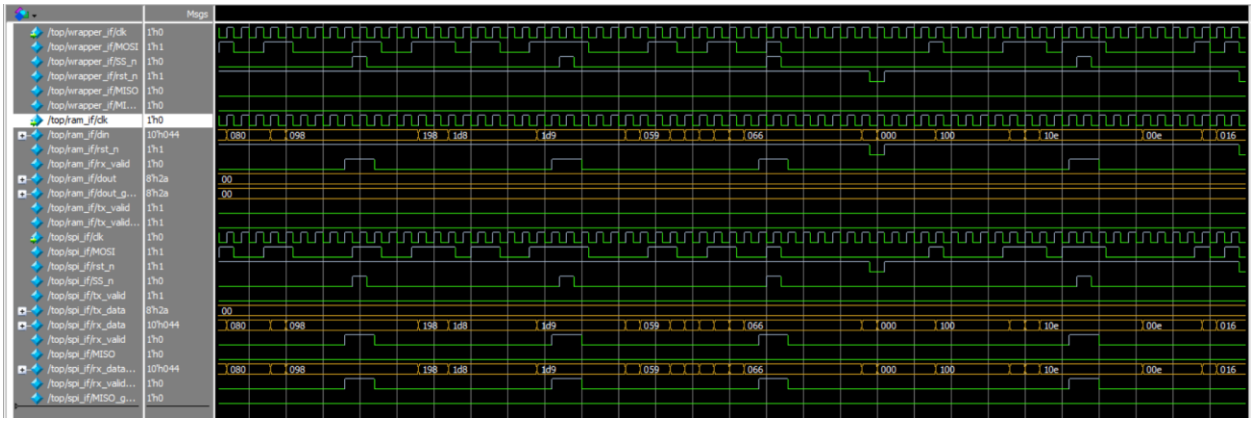
- Full trans: that checks the transation of the din[9:8] from write adress => write data => read address => read data "00 => 01 => 10 => 11" as the din now is the same as rx_data which is changing bit by bit so even when the MSB 2 bit are changed from 00 to 01 it keeps 01 for a while till the rx_data is full and then rise rx_valid so in the functional coverage we should say that these 2 bits keept for a while so there is no meaning to but it.
- cross: the transation with rx_valid is high is not hapeening as in the system the slave rise it when all bins of rx_data are ready to be sent so no rx_valid high with the change or transition the rx_data which is the same as din now in stable and not changed when the rx_valid is high.

Bug Report

- There were no bugs.

QuestaSim Snippets

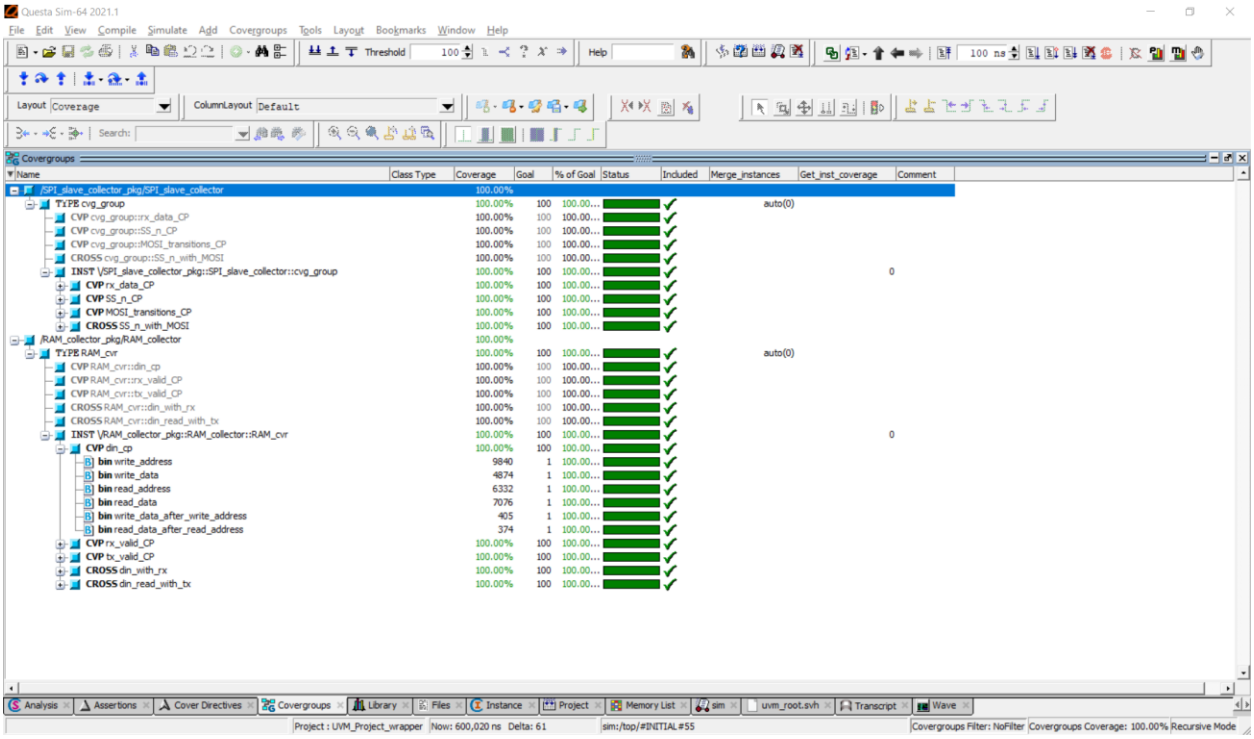
Waveform



Transcript

```
# UVM_INFO verilog_src/questa_uvm_pkg-1.2/src/questa_uvm_pkg.sv(277) @ 0: reporter [Questa UVM] QUESTA_UVM-1.2.3
# UVM_INFO verilog_src/questa_uvm_pkg-1.2/src/questa_uvm_pkg.sv(278) @ 0: reporter [Questa UVM]  questa_uvm::init(all)
# UVM_INFO @ 0: reporter [RNTST] Running test SPI_wrapper_test...
# ************************************************************************
# * Questa UVM Transaction Recording Turned ON.                          *
# * recording_detail has been set.                                       *
# *  To turn off, set 'recording_detail' to off:                         *
# * uvm_config_db#(int)            ::set(null, "", "recording_detail", 0); *
# * uvm_config_db#(uvm_bitstream_t)::set(null, "", "recording_detail", 0); *
# ************************************************************************
# ** Error: Assertion error.
#    Time: 294010 ns Started: 294010 ns  Scope: top.DUT.sva_WRAPPER_inst File: SPI_wrapper_sva.sv Line: 14
# UVM_INFO verilog_src/uvm-1.1d/src/base/uvm_objection.svh(1267) @ 600020: reporter [TEST_DONE] 'run' phase is ready to proceed to the 'extract' phase
# UVM_INFO RAM_scorboard.sv(54) @ 600020: uvm_test_top.ram_env.sb [repo phase] RAM correct times: 30001
# UVM_INFO RAM_scorboard.sv(55) @ 600020: uvm_test_top.ram_env.sb [repo phase] RAM error times: 0
# UVM_INFO SPI_slave_scoreboard.sv(55) @ 600020: uvm_test_top.spi_env.sb [repo phase] Slave correct times: 30001
# UVM_INFO SPI_slave_scoreboard.sv(56) @ 600020: uvm_test_top.spi_env.sb [repo phase] Slave error times: 0
# correct_count =30001 , error_count =0
#
# --- UVM Report Summary ---
#
# ** Report counts by severity
# UVM_INFO :    8
# UVM_WARNING :    0
# UVM_ERROR :    0
# UVM_FATAL :    0
# ** Report counts by id
# [Questa UVM]     2
# [RNTST]     1
# [TEST_DONE]     1
# [repo phase]     4
# ** Note: $finish    : D:/QuestaSim/win64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh(430)
#    Time: 600020 ns  Iteration: 61  Instance: /top
# Break in Task uvm_pkg/uvm_root::run_test at D:/QuestaSim/win64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh line 430
# QuestaSim-64 vcover 2021.1 Coverage Utility 2021.01 Jan 19 2021
# Start time: 11:06:26 on Oct 12,2025
# vcover report top.ucdb -details -annotate -all -output coverage_rpt.txt
# ** Error: (vcover-7) Failed to open UCDB file "top.ucdb" in read mode.
# ** Note: (vcover-17388) No matching coverage data found in file 'top.ucdb'.
# End time: 11:06:26 on Oct 12,2025, Elapsed time: 0:00:00
```

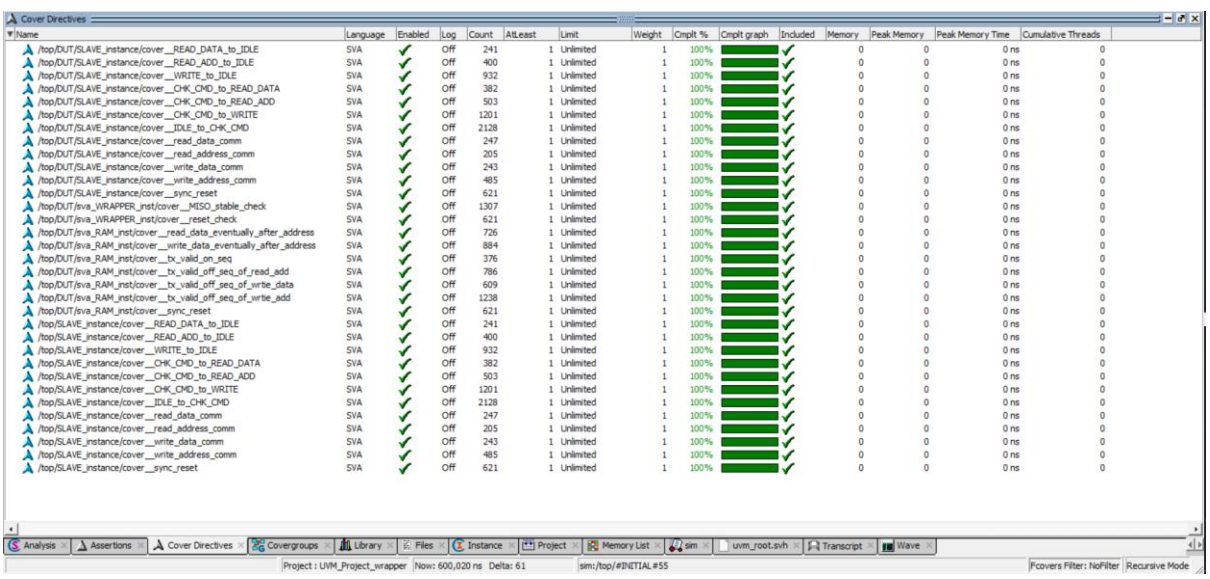
Functional Coverage



Name	Class Type	Coverage	Goal	% of Goal	Status	Included	Merge_instances	Get_inst_coverage	Comment
/SPI_slave_collector_pkg/SPI_slave_collector		100.00%							
TYPE cvg_group		100.00%	100	100.00...		✓	auto(0)		
CVP cvg_group::rx_data_CP		100.00%	100	100.00...		✓			
CVP cvg_group::SS_n_CP		100.00%	100	100.00...		✓			
CVP cvg_group::MOSI_transitions_CP		100.00%	100	100.00...		✓			
CROSS cvg_group::SS_n_with_MOSI		100.00%	100	100.00...		✓			
INST \/SPI_slave_collector_pkg::SPI_slave_collector::cvg_group		100.00%	100	100.00...		✓		0	
CVP rx_data_CP		100.00%	100	100.00...		✓			
CVP SS_n_CP		100.00%	100	100.00...		✓			
CVP MOSI_transitions_CP		100.00%	100	100.00...		✓			
CROSS SS_n_with_MOSI		100.00%	100	100.00...		✓			
/RAM_collector_pkg/RAM_collector		100.00%							
TYPE RAM_cvr		100.00%	100	100.00...		✓	auto(0)		
CVP RAM_cvr::din_cp		100.00%	100	100.00...		✓			
CVP RAM_cvr::rx_valid_CP		100.00%	100	100.00...		✓			
CVP RAM_cvr::tx_valid_CP		100.00%	100	100.00...		✓			
CROSS RAM_cvr::din_with_rx		100.00%	100	100.00...		✓			
CROSS RAM_cvr::din_read_with_tx		100.00%	100	100.00...		✓			
INST \/RAM_collector_pkg::RAM_collector::RAM_cvr		100.00%	100	100.00...		✓		0	
CVP din_cp		100.00%	100	100.00...		✓			
bin write_address		9840	1	100.00...		✓			
bin write_data		4874	1	100.00...		✓			
bin read_address		6332	1	100.00...		✓			
bin read_data		7076	1	100.00...		✓			
bin write_data_after_write_address		405	1	100.00...		✓			
bin read_data_after_read_address		374	1	100.00...		✓			
CVP rx_valid_CP		100.00%	100	100.00...		✓			
CVP tx_valid_CP		100.00%	100	100.00...		✓			
CROSS din_with_rx		100.00%	100	100.00...		✓			
CROSS din_read_with_tx		100.00%	100	100.00...		✓			

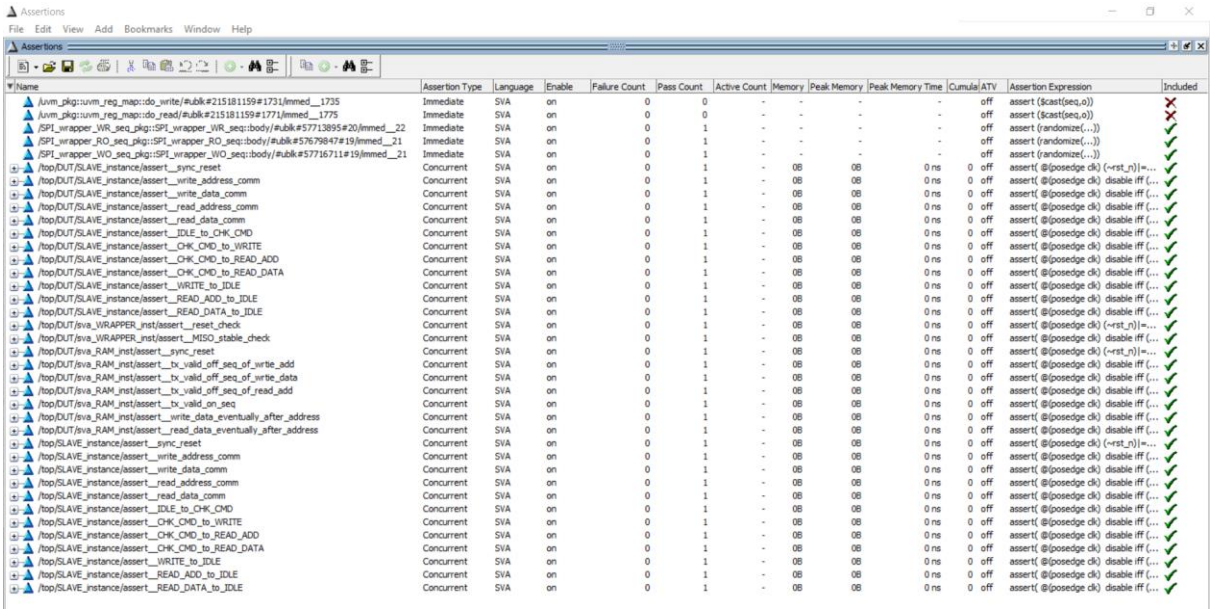
Sequential Domain Coverage (Assertion Coverage) Snippets

Cover Directives



Name	Language	Enabled	Log	Count	AtLeast	Limit	Weight	Cmplt %	Cmplt graph	Included	Memory	Peak Memory	Peak Memory Time	Cumulative Threads
/top/DUT/SLAVE_instance/cover__READ_DATA_to_IDLE	SVA	✓	Off	241	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/DUT/SLAVE_instance/cover__READ_ADD_to_IDLE	SVA	✓	Off	400	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/DUT/SLAVE_instance/cover__WRITE_to_IDLE	SVA	✓	Off	932	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/DUT/SLAVE_instance/cover__CHK_CMD_to_READ_DATA	SVA	✓	Off	382	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/DUT/SLAVE_instance/cover__CHK_CMD_to_READ_ADD	SVA	✓	Off	503	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/DUT/SLAVE_instance/cover__CHK_CMD_to_WRITE	SVA	✓	Off	1201	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/DUT/SLAVE_instance/cover__IDLE_to_CHK_CMD	SVA	✓	Off	2128	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/DUT/SLAVE_instance/cover__read_data_comm	SVA	✓	Off	247	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/DUT/SLAVE_instance/cover__read_address_comm	SVA	✓	Off	205	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/DUT/SLAVE_instance/cover__write_data_comm	SVA	✓	Off	243	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/DUT/SLAVE_instance/cover__write_address_comm	SVA	✓	Off	485	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/DUT/SLAVE_instance/cover__sync_reset	SVA	✓	Off	621	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/DUT/sva_WRAPPER_inst/cover__MISO_stable_check	SVA	✓	Off	1307	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/DUT/sva_WRAPPER_inst/cover__reset_check	SVA	✓	Off	621	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/DUT/sva_RAM_inst/cover__read_data_eventually_after_address	SVA	✓	Off	726	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/DUT/sva_RAM_inst/cover__write_data_eventually_after_address	SVA	✓	Off	884	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/DUT/sva_RAM_inst/cover__tx_valid_on_seq	SVA	✓	Off	376	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/DUT/sva_RAM_inst/cover__tx_valid_off_seq_of_read_add	SVA	✓	Off	786	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/DUT/sva_RAM_inst/cover__tx_valid_off_seq_of_write_data	SVA	✓	Off	609	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/DUT/sva_RAM_inst/cover__tx_valid_off_seq_of_write_add	SVA	✓	Off	1238	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/DUT/sva_RAM_inst/cover__sync_reset	SVA	✓	Off	621	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/SLAVE_instance/cover__READ_DATA_to_IDLE	SVA	✓	Off	241	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/SLAVE_instance/cover__READ_ADD_to_IDLE	SVA	✓	Off	400	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/SLAVE_instance/cover__WRITE_to_IDLE	SVA	✓	Off	932	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/SLAVE_instance/cover__CHK_CMD_to_READ_DATA	SVA	✓	Off	382	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/SLAVE_instance/cover__CHK_CMD_to_READ_ADD	SVA	✓	Off	503	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/SLAVE_instance/cover__CHK_CMD_to_WRITE	SVA	✓	Off	1201	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/SLAVE_instance/cover__IDLE_to_CHK_CMD	SVA	✓	Off	2128	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/SLAVE_instance/cover__read_data_comm	SVA	✓	Off	247	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/SLAVE_instance/cover__read_address_comm	SVA	✓	Off	205	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/SLAVE_instance/cover__write_data_comm	SVA	✓	Off	243	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/SLAVE_instance/cover__write_address_comm	SVA	✓	Off	485	1	Unlimited	1	100%		✓	0	0	0 ns	0
/top/SLAVE_instance/cover__sync_reset	SVA	✓	Off	621	1	Unlimited	1	100%		✓	0	0	0 ns	0

Assertions



Name	Assertion Type	Language	Enable	Failure Count	Pass Count	Active Count	Memory	Peak Memory	Peak Memory Time	Cumula	ATV	Assertion Expression	Included
/uvm_pkg::uvm_reg_map::do_write/#ublk#215181159#1731/immed__1735	Immediate	SVA	on	0	0	-	-	-	-		off	assert ($cast(seq,o))	✗
/uvm_pkg::uvm_reg_map::do_read/#ublk#215181159#1771/immed__1775	Immediate	SVA	on	0	0	-	-	-	-		off	assert ($cast(seq,o))	✗
/SPI_wrapper_WR_seq_pkg::SPI_wrapper_WR_seq::body/#ublk#57713895#20/immed__22	Immediate	SVA	on	0	1	-	-	-	-		off	assert (randomize(...))	✓
/SPI_wrapper_RO_seq_pkg::SPI_wrapper_RO_seq::body/#ublk#57679847#19/immed__21	Immediate	SVA	on	0	1	-	-	-	-		off	assert (randomize(...))	✓
/SPI_wrapper_WO_seq_pkg::SPI_wrapper_WO_seq::body/#ublk#57716711#19/immed__21	Immediate	SVA	on	0	1	-	-	-	-		off	assert (randomize(...))	✓
/top/DUT/SLAVE_instance/assert__sync_reset	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) (~rst_n)|=...	✓
/top/DUT/SLAVE_instance/assert__write_address_comm	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/DUT/SLAVE_instance/assert__write_data_comm	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/DUT/SLAVE_instance/assert__read_address_comm	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/DUT/SLAVE_instance/assert__read_data_comm	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/DUT/SLAVE_instance/assert__IDLE_to_CHK_CMD	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/DUT/SLAVE_instance/assert__CHK_CMD_to_WRITE	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/DUT/SLAVE_instance/assert__CHK_CMD_to_READ_ADD	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/DUT/SLAVE_instance/assert__CHK_CMD_to_READ_DATA	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/DUT/SLAVE_instance/assert__WRITE_to_IDLE	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/DUT/SLAVE_instance/assert__READ_ADD_to_IDLE	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/DUT/SLAVE_instance/assert__READ_DATA_to_IDLE	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/DUT/sva_WRAPPER_inst/assert__reset_check	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) (~rst_n)|=...	✓
/top/DUT/sva_WRAPPER_inst/assert__MISO_stable_check	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/DUT/sva_RAM_inst/assert__sync_reset	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) (~rst_n)|=...	✓
/top/DUT/sva_RAM_inst/assert__tx_valid_off_seq_of_write_add	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/DUT/sva_RAM_inst/assert__tx_valid_off_seq_of_write_data	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/DUT/sva_RAM_inst/assert__tx_valid_off_seq_of_read_add	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/DUT/sva_RAM_inst/assert__tx_valid_on_seq	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/DUT/sva_RAM_inst/assert__write_data_eventually_after_address	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/DUT/sva_RAM_inst/assert__read_data_eventually_after_address	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/SLAVE_instance/assert__sync_reset	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) (~rst_n)|=...	✓
/top/SLAVE_instance/assert__write_address_comm	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/SLAVE_instance/assert__write_data_comm	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/SLAVE_instance/assert__read_address_comm	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/SLAVE_instance/assert__read_data_comm	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/SLAVE_instance/assert__IDLE_to_CHK_CMD	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/SLAVE_instance/assert__CHK_CMD_to_WRITE	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/SLAVE_instance/assert__CHK_CMD_to_READ_ADD	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/SLAVE_instance/assert__CHK_CMD_to_READ_DATA	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/SLAVE_instance/assert__WRITE_to_IDLE	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/SLAVE_instance/assert__READ_ADD_to_IDLE	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓
/top/SLAVE_instance/assert__READ_DATA_to_IDLE	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	0	off	assert(@(posedge clk) disable iff (...	✓

Assertions' Table

Feature	Assertion
whenever reset is asserted, the output (MISO) is inactive.	`@(posedge clk) (!rst_n) |=> (!MISO);`
MISO remains with a stable value eventually as long as it is not a read data operation	`@(posedge clk) disable iff (!rst_n) $fell(SS_n) |=> (SS_n == 1'b0 && $stable(MISO)) [*11];`