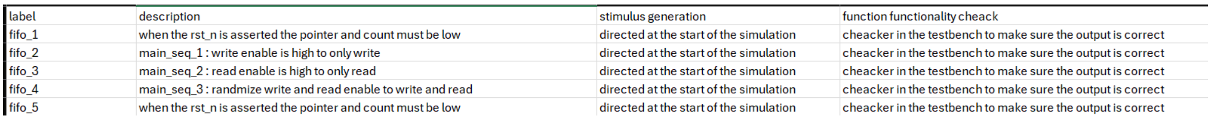
Abdalrahman Taha Mahmed

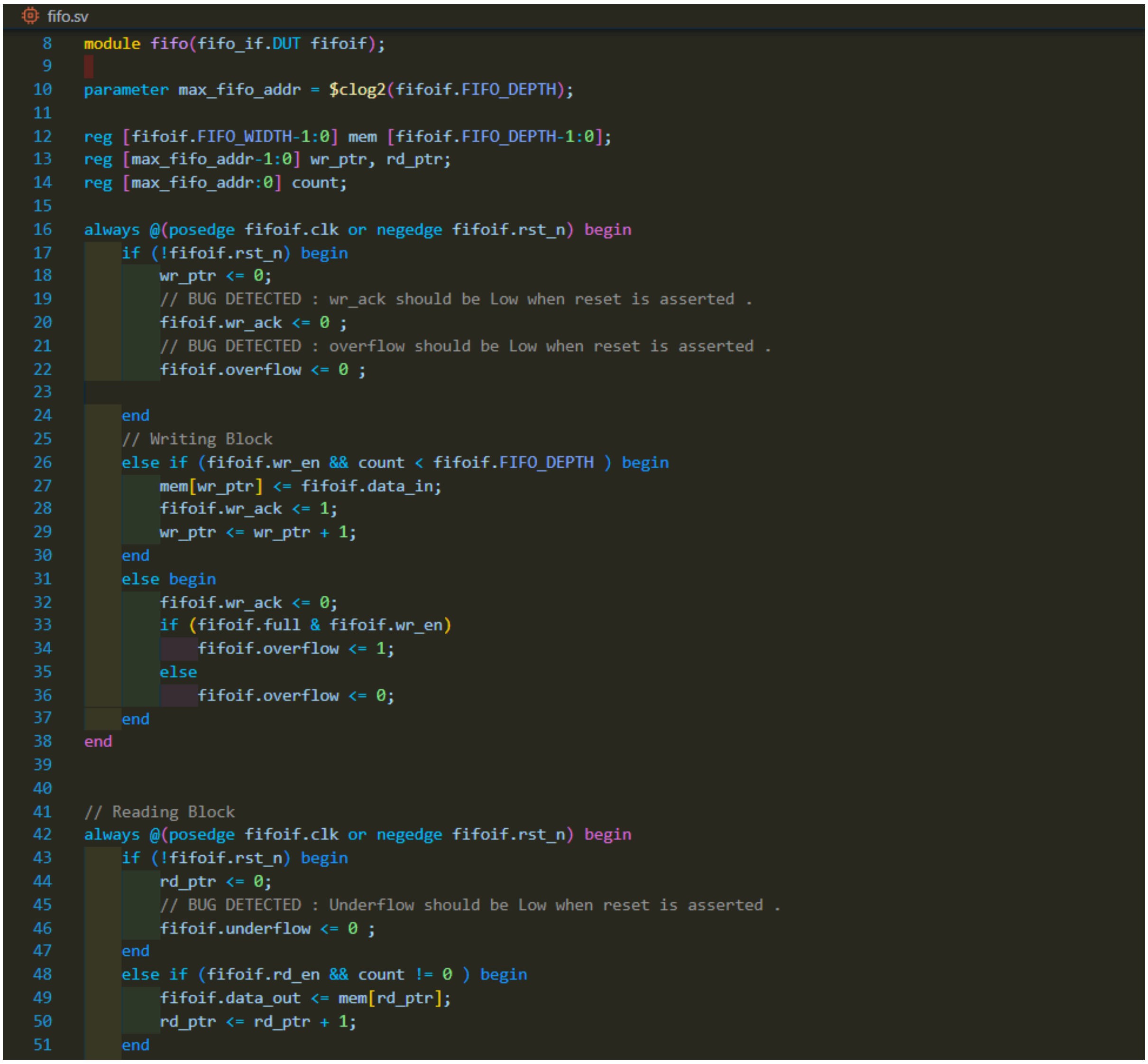
FIFO Using UVM

Verification Plan :

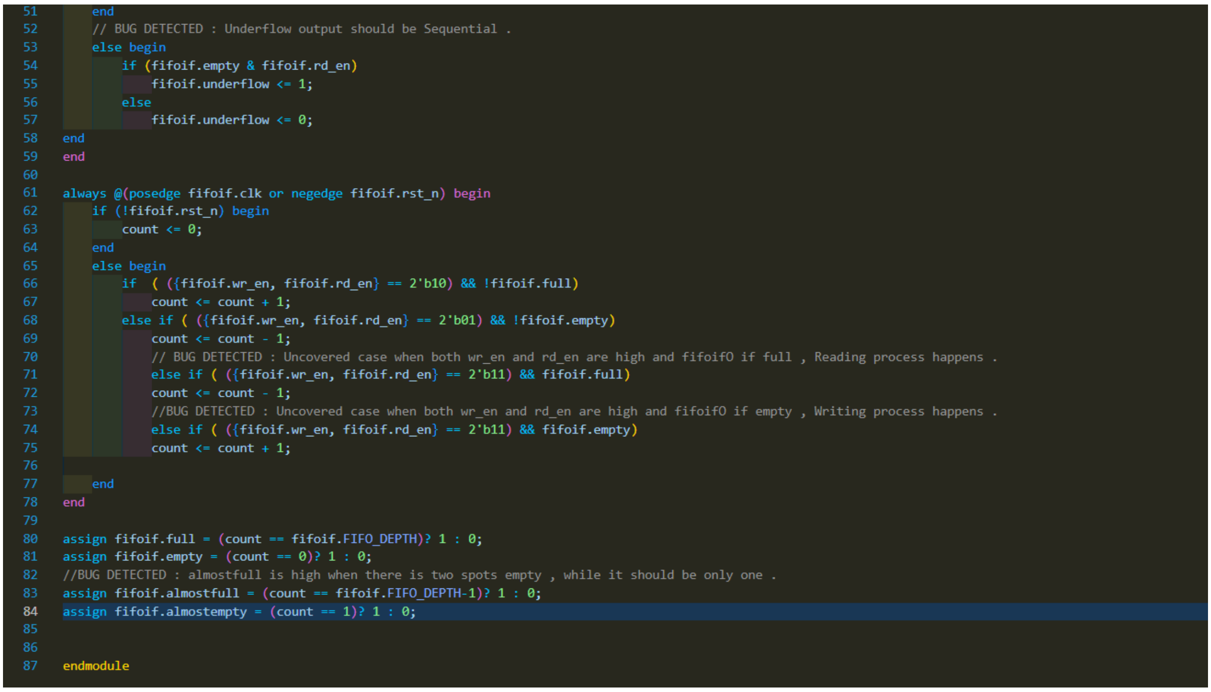


label	description	stimulus generation	function functionality cheack
fifo_1	when the rst_n is asserted the pointer and count must be low	directed at the start of the simulation	cheacker in the testbench to make sure the output is correct
fifo_2	main_seq_1 : write enable is high to only write	directed at the start of the simulation	cheacker in the testbench to make sure the output is correct
fifo_3	main_seq_2 : read enable is high to only read	directed at the start of the simulation	cheacker in the testbench to make sure the output is correct
fifo_4	main_seq_3 : randmize write and read enable to write and read	directed at the start of the simulation	cheacker in the testbench to make sure the output is correct
fifo_5	when the rst_n is asserted the pointer and count must be low	directed at the start of the simulation	cheacker in the testbench to make sure the output is correct

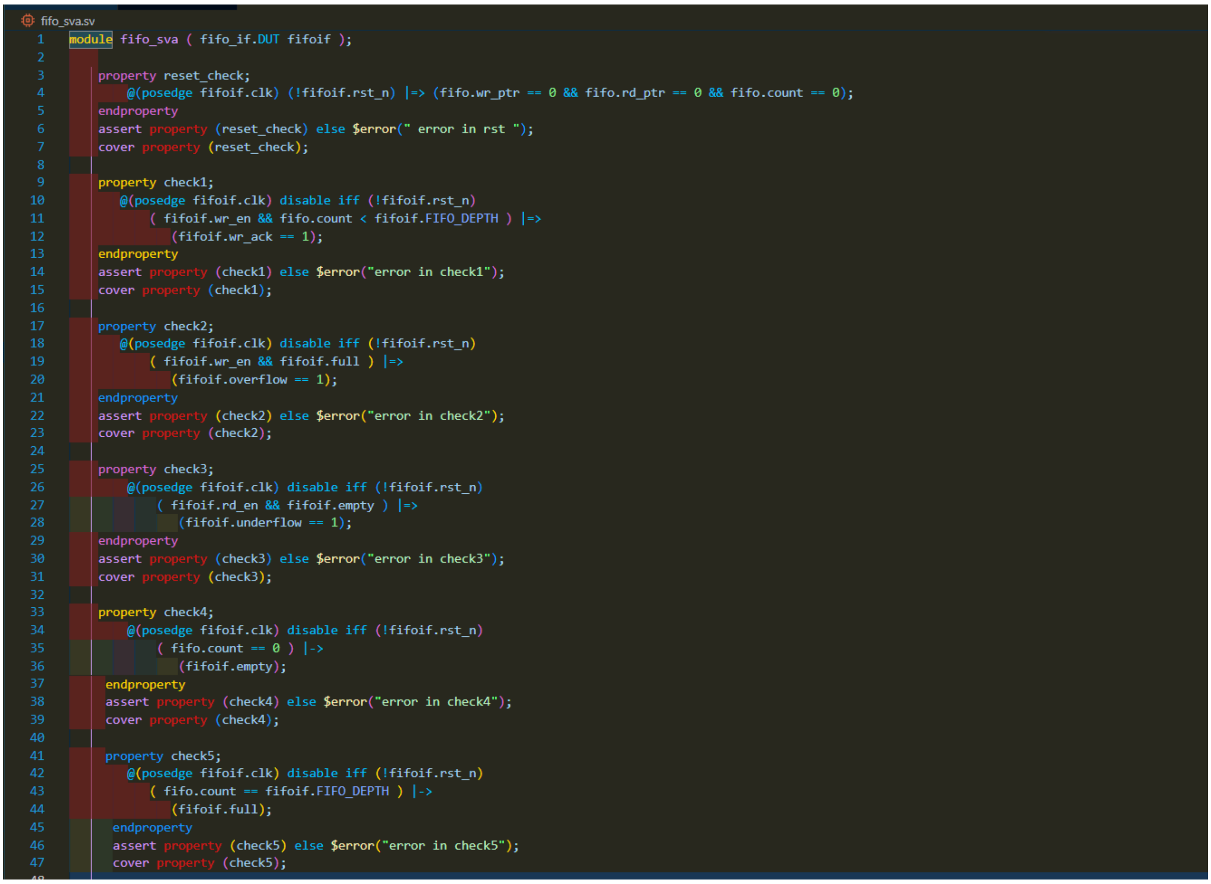
Codes:



```
module fifo(fifo_if.DUT fifoif);


parameter max_fifo_addr = $clog2(fifoif.FIFO_DEPTH);

reg [fifoif.FIFO_WIDTH-1:0] mem [fifoif.FIFO_DEPTH-1:0];
reg [max_fifo_addr-1:0] wr_ptr, rd_ptr;
reg [max_fifo_addr:0] count;

always @(posedge fifoif.clk or negedge fifoif.rst_n) begin
    if (!fifoif.rst_n) begin
        wr_ptr <= 0;
        // BUG DETECTED : wr_ack should be Low when reset is asserted .
        fifoif.wr_ack <= 0 ;
        // BUG DETECTED : overflow should be Low when reset is asserted .
        fifoif.overflow <= 0 ;

    end
    // Writing Block
    else if (fifoif.wr_en && count < fifoif.FIFO_DEPTH ) begin
        mem[wr_ptr] <= fifoif.data_in;
        fifoif.wr_ack <= 1;
        wr_ptr <= wr_ptr + 1;
    end
    else begin
        fifoif.wr_ack <= 0;
        if (fifoif.full & fifoif.wr_en)
            fifoif.overflow <= 1;
        else
            fifoif.overflow <= 0;
    end
end


// Reading Block
always @(posedge fifoif.clk or negedge fifoif.rst_n) begin
    if (!fifoif.rst_n) begin
        rd_ptr <= 0;
        // BUG DETECTED : Underflow should be Low when reset is asserted .
        fifoif.underflow <= 0 ;
    end
    else if (fifoif.rd_en && count != 0 ) begin
        fifoif.data_out <= mem[rd_ptr];
        rd_ptr <= rd_ptr + 1;
    end
```

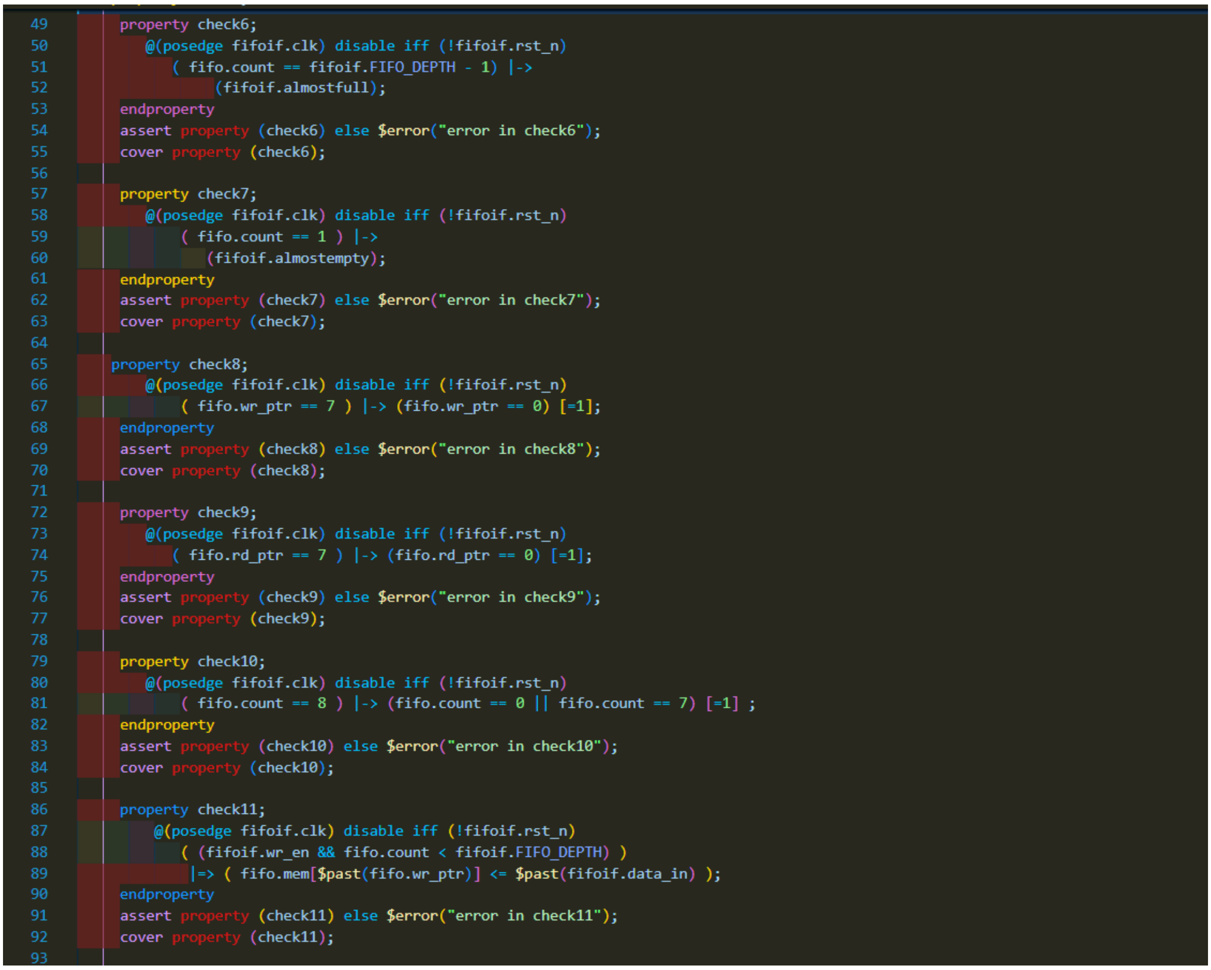


```
    end
    // BUG DETECTED : Underflow output should be Sequential .
    else begin
        if (fifoif.empty & fifoif.rd_en)
            fifoif.underflow <= 1;
        else
            fifoif.underflow <= 0;
end
end

always @(posedge fifoif.clk or negedge fifoif.rst_n) begin
    if (!fifoif.rst_n) begin
        count <= 0;
    end
    else begin
        if  ( ({fifoif.wr_en, fifoif.rd_en} == 2'b10) && !fifoif.full)
            count <= count + 1;
        else if ( ({fifoif.wr_en, fifoif.rd_en} == 2'b01) && !fifoif.empty)
            count <= count - 1;
            // BUG DETECTED : Uncovered case when both wr_en and rd_en are high and fifoifO if full , Reading process happens .
            else if ( ({fifoif.wr_en, fifoif.rd_en} == 2'b11) && fifoif.full)
            count <= count - 1;
            //BUG DETECTED : Uncovered case when both wr_en and rd_en are high and fifoifO if empty , Writing process happens .
            else if ( ({fifoif.wr_en, fifoif.rd_en} == 2'b11) && fifoif.empty)
            count <= count + 1;

    end
end

assign fifoif.full = (count == fifoif.FIFO_DEPTH)? 1 : 0;
assign fifoif.empty = (count == 0)? 1 : 0;
//BUG DETECTED : almostfull is high when there is two spots empty , while it should be only one .
assign fifoif.almostfull = (count == fifoif.FIFO_DEPTH-1)? 1 : 0;
assign fifoif.almostempty = (count == 1)? 1 : 0;


endmodule
```

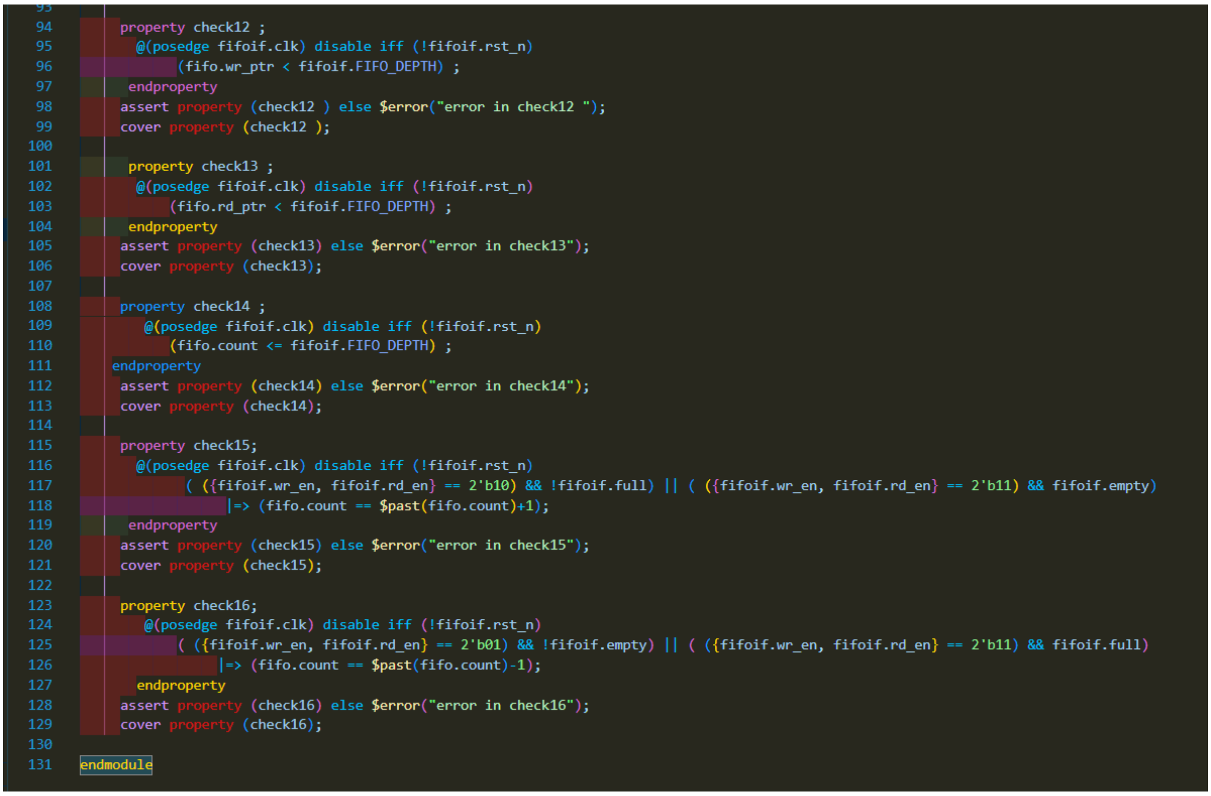


fifo_sva.sv

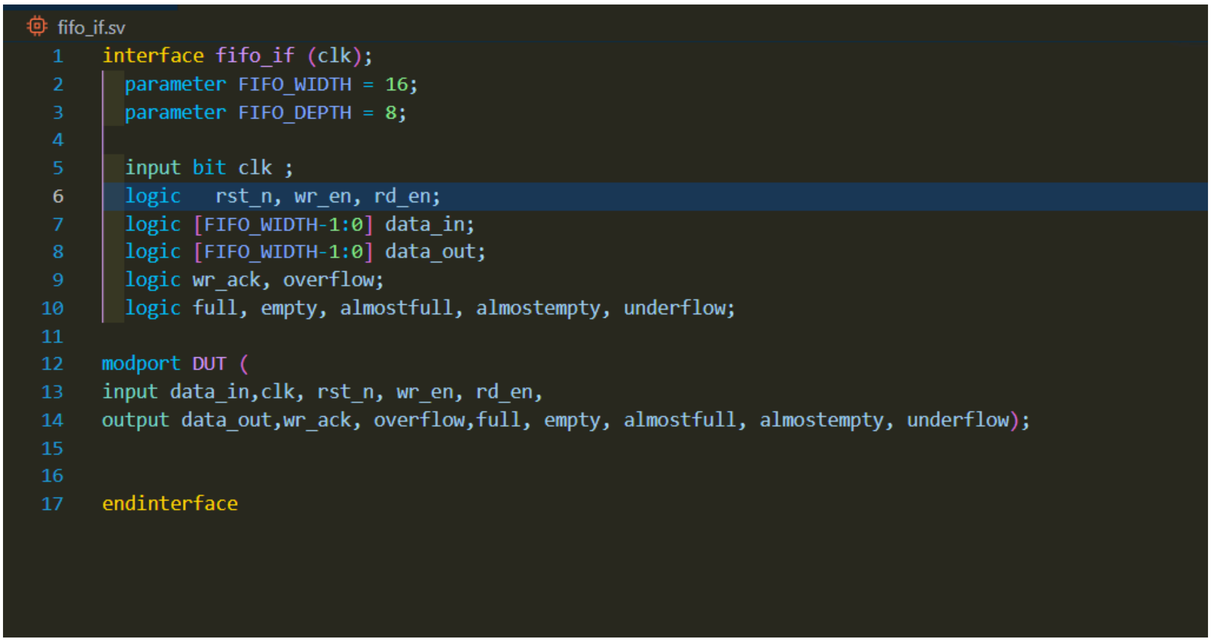
```
module fifo_sva ( fifo_if.DUT fifoif );

    property reset_check;
        @(posedge fifoif.clk) (!fifoif.rst_n) |=> (fifo.wr_ptr == 0 && fifo.rd_ptr == 0 && fifo.count == 0);
    endproperty
    assert property (reset_check) else $error(" error in rst ");
    cover property (reset_check);

    property check1;
       @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
           ( fifoif.wr_en && fifo.count < fifoif.FIFO_DEPTH ) |=>
              (fifoif.wr_ack == 1);
    endproperty
    assert property (check1) else $error("error in check1");
    cover property (check1);

    property check2;
       @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
           ( fifoif.wr_en && fifoif.full ) |=>
              (fifoif.overflow == 1);
    endproperty
    assert property (check2) else $error("error in check2");
    cover property (check2);

    property check3;
        @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
            ( fifoif.rd_en && fifoif.empty ) |=>
               (fifoif.underflow == 1);
    endproperty
    assert property (check3) else $error("error in check3");
    cover property (check3);

    property check4;
        @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
            ( fifo.count == 0 ) |->
               (fifoif.empty);
     endproperty
     assert property (check4) else $error("error in check4");
     cover property (check4);

     property check5;
        @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
           ( fifo.count == fifoif.FIFO_DEPTH ) |->
              (fifoif.full);
      endproperty
      assert property (check5) else $error("error in check5");
      cover property (check5);
```



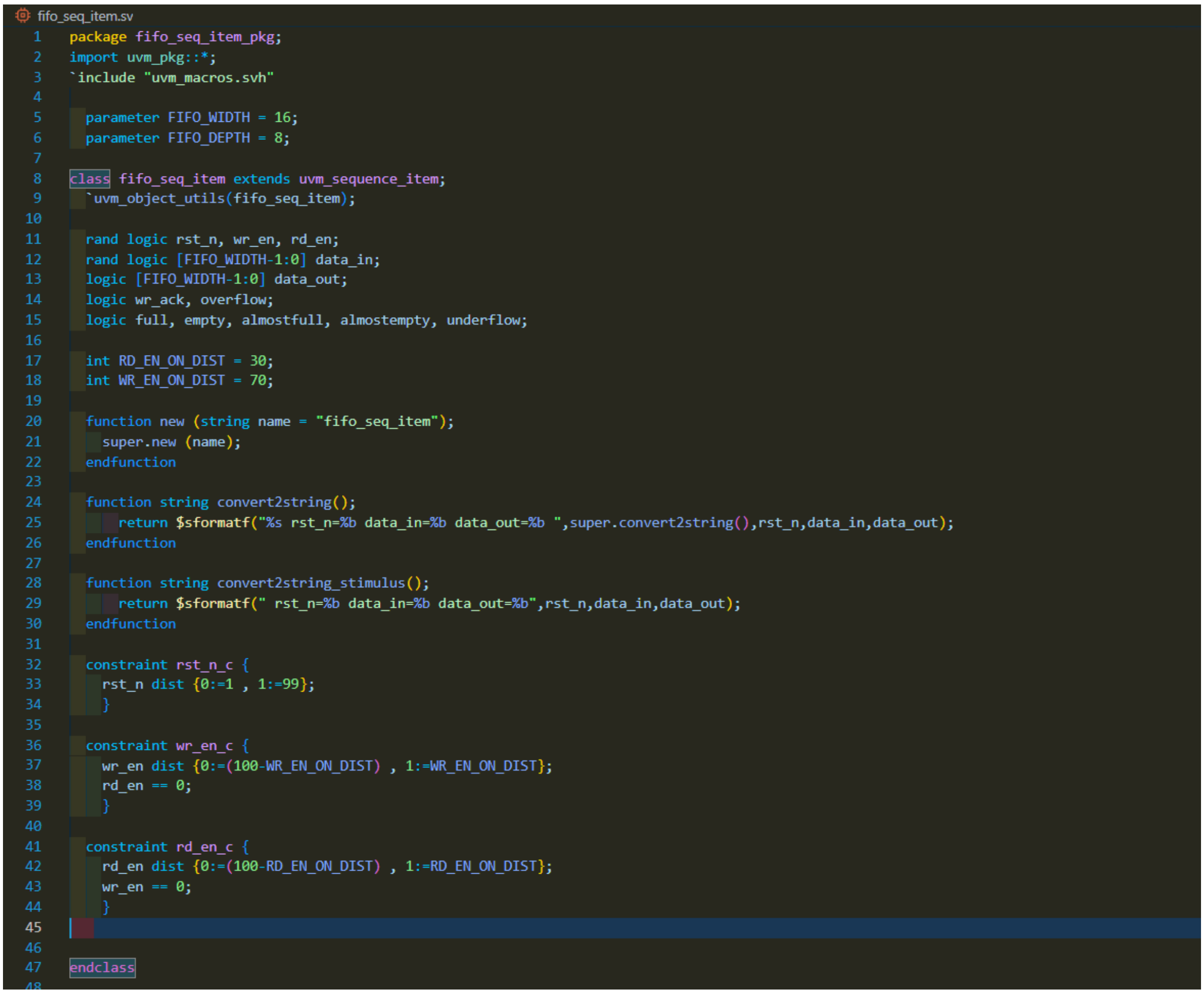
```
property check6;
    @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
        ( fifo.count == fifoif.FIFO_DEPTH - 1) |->
              (fifoif.almostfull);
endproperty
assert property (check6) else $error("error in check6");
cover property (check6);

property check7;
    @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
         ( fifo.count == 1 ) |->
             (fifoif.almostempty);
endproperty
assert property (check7) else $error("error in check7");
cover property (check7);

property check8;
    @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
         ( fifo.wr_ptr == 7 ) |-> (fifo.wr_ptr == 0) [=1];
endproperty
assert property (check8) else $error("error in check8");
cover property (check8);

property check9;
    @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
        ( fifo.rd_ptr == 7 ) |-> (fifo.rd_ptr == 0) [=1];
endproperty
assert property (check9) else $error("error in check9");
cover property (check9);

property check10;
    @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
         ( fifo.count == 8 ) |-> (fifo.count == 0 || fifo.count == 7) [=1] ;
endproperty
assert property (check10) else $error("error in check10");
cover property (check10);

property check11;
     @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
         ( (fifoif.wr_en && fifo.count < fifoif.FIFO_DEPTH) )
           |=> ( fifo.mem[$past(fifo.wr_ptr)] <= $past(fifoif.data_in) );
endproperty
assert property (check11) else $error("error in check11");
cover property (check11);
```



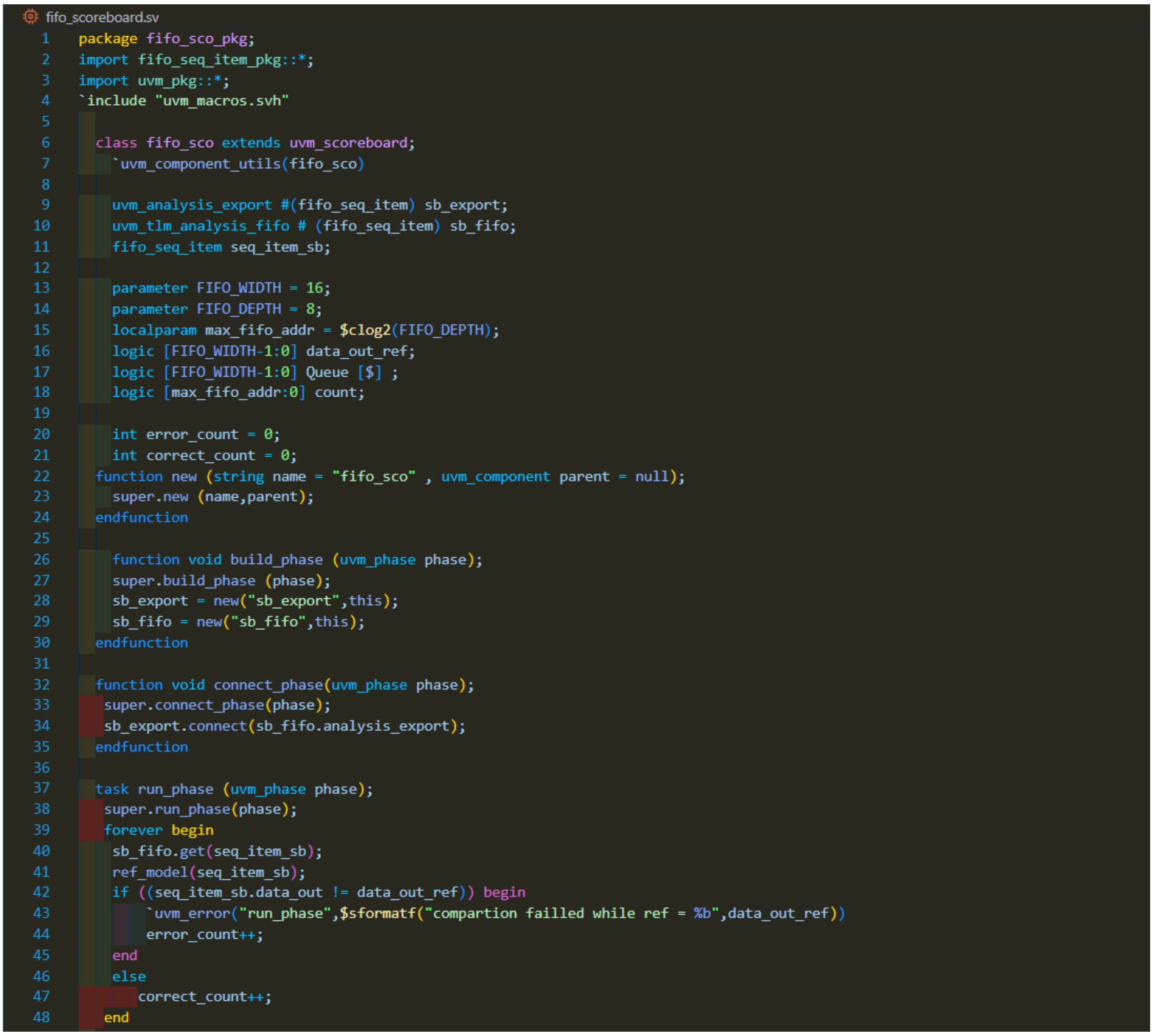
```
property check12 ;
  @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
        (fifo.wr_ptr < fifoif.FIFO_DEPTH) ;
 endproperty
assert property (check12 ) else $error("error in check12 ");
cover property (check12 );

 property check13 ;
  @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
       (fifo.rd_ptr < fifoif.FIFO_DEPTH) ;
 endproperty
assert property (check13) else $error("error in check13");
cover property (check13);

property check14 ;
   @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
       (fifo.count <= fifoif.FIFO_DEPTH) ;
endproperty
assert property (check14) else $error("error in check14");
cover property (check14);

property check15;
  @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
         ( ({fifoif.wr_en, fifoif.rd_en} == 2'b10) && !fifoif.full) || ( ({fifoif.wr_en, fifoif.rd_en} == 2'b11) && fifoif.empty)
              |=> (fifo.count == $past(fifo.count)+1);
 endproperty
assert property (check15) else $error("error in check15");
cover property (check15);

property check16;
   @(posedge fifoif.clk) disable iff (!fifoif.rst_n)
        ( ({fifoif.wr_en, fifoif.rd_en} == 2'b01) && !fifoif.empty) || ( ({fifoif.wr_en, fifoif.rd_en} == 2'b11) && fifoif.full)
             |=> (fifo.count == $past(fifo.count)-1);
  endproperty
assert property (check16) else $error("error in check16");
cover property (check16);

endmodule
```


fifo_if.sv

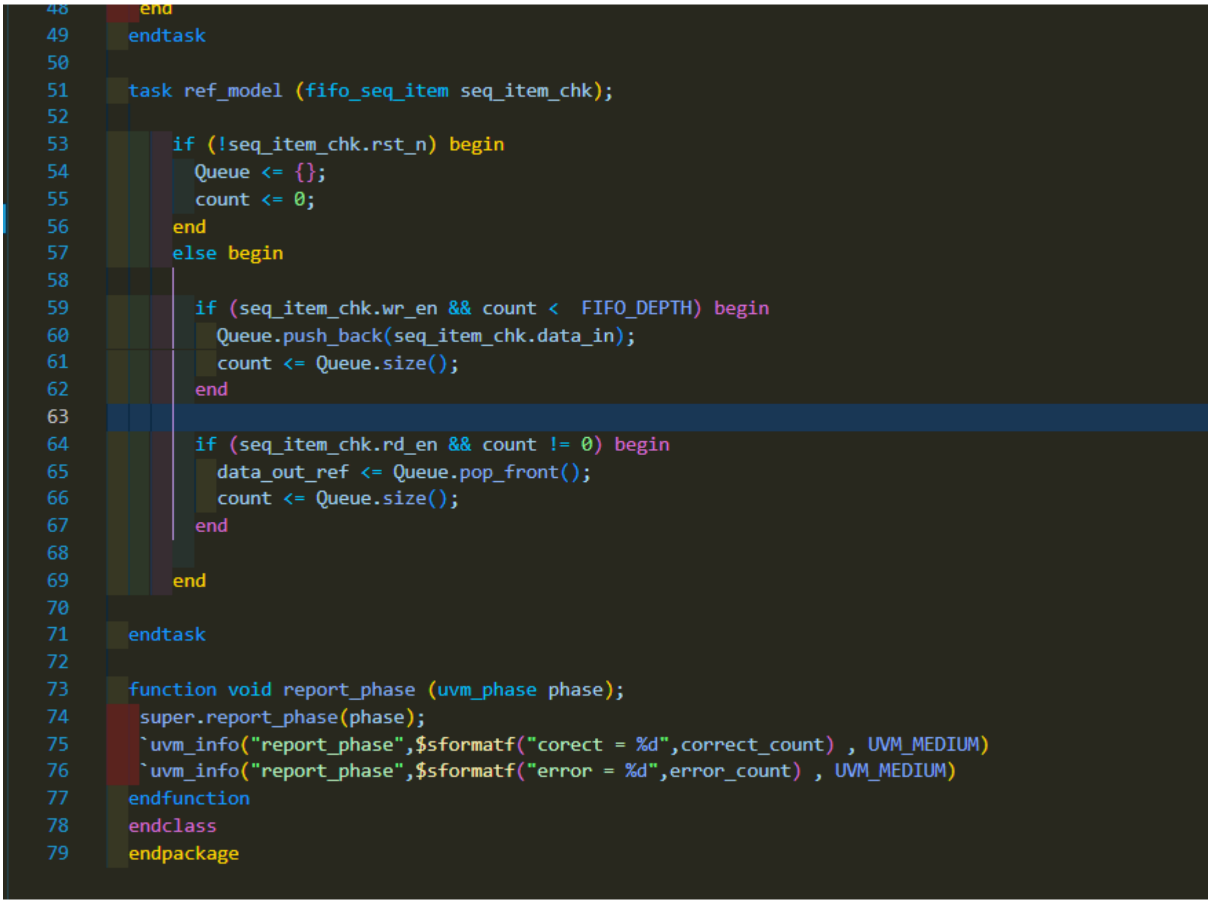
```
interface fifo_if (clk);
  parameter FIFO_WIDTH = 16;
  parameter FIFO_DEPTH = 8;

  input bit clk ;
  logic   rst_n, wr_en, rd_en;
  logic [FIFO_WIDTH-1:0] data_in;
  logic [FIFO_WIDTH-1:0] data_out;
  logic wr_ack, overflow;
  logic full, empty, almostfull, almostempty, underflow;

modport DUT (
input data_in,clk, rst_n, wr_en, rd_en,
output data_out,wr_ack, overflow,full, empty, almostfull, almostempty, underflow);


endinterface
```


fifo_seq_item.sv

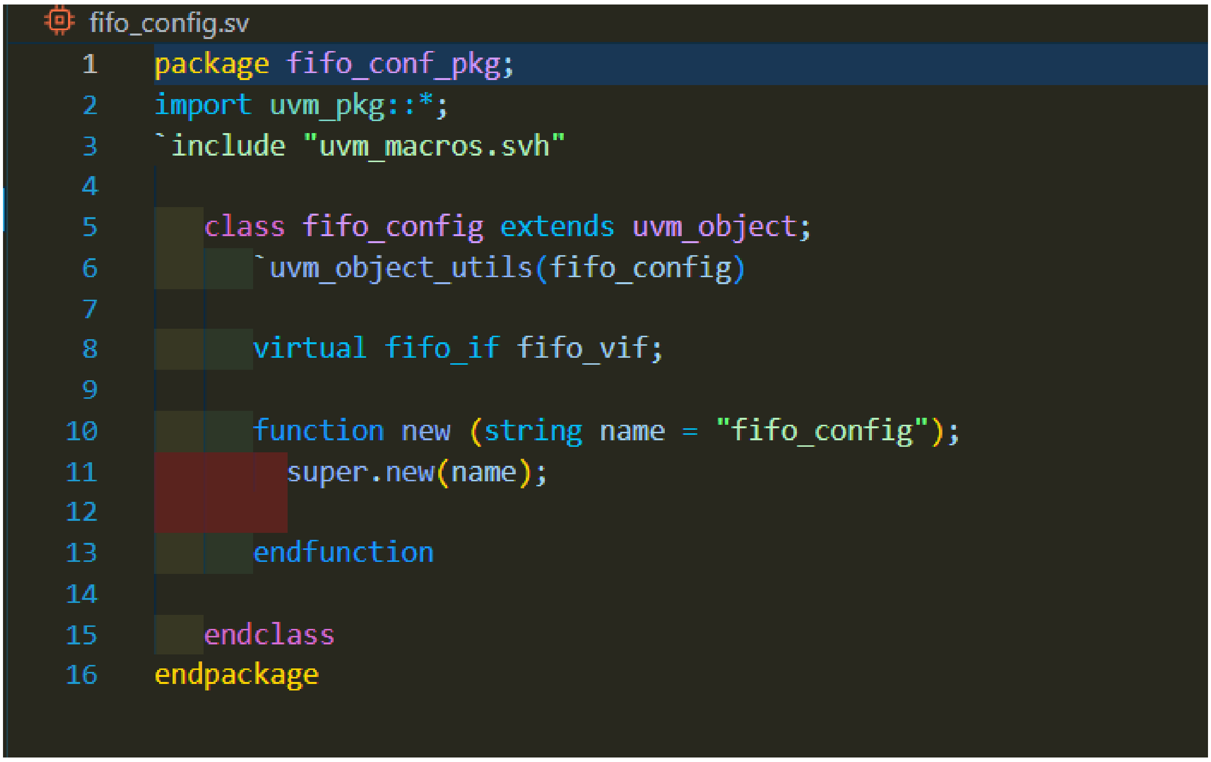
```
package fifo_seq_item_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"

  parameter FIFO_WIDTH = 16;
  parameter FIFO_DEPTH = 8;

class fifo_seq_item extends uvm_sequence_item;
  `uvm_object_utils(fifo_seq_item);

  rand logic rst_n, wr_en, rd_en;
  rand logic [FIFO_WIDTH-1:0] data_in;
  logic [FIFO_WIDTH-1:0] data_out;
  logic wr_ack, overflow;
  logic full, empty, almostfull, almostempty, underflow;

  int RD_EN_ON_DIST = 30;
  int WR_EN_ON_DIST = 70;

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

  function string convert2string();
      return $sformatf("%s rst_n=%b data_in=%b data_out=%b ",super.convert2string(),rst_n,data_in,data_out);
  endfunction

  function string convert2string_stimulus();
      return $sformatf(" rst_n=%b data_in=%b data_out=%b",rst_n,data_in,data_out);
  endfunction

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

  constraint wr_en_c {
    wr_en dist {0:=(100-WR_EN_ON_DIST) , 1:=WR_EN_ON_DIST};
    rd_en == 0;
    }

  constraint rd_en_c {
    rd_en dist {0:=(100-RD_EN_ON_DIST) , 1:=RD_EN_ON_DIST};
    wr_en == 0;
    }


endclass
```


fifo_scoreboard.sv

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

  class fifo_sco extends uvm_scoreboard;
    `uvm_component_utils(fifo_sco)

    uvm_analysis_export #(fifo_seq_item) sb_export;
    uvm_tlm_analysis_fifo # (fifo_seq_item) sb_fifo;
    fifo_seq_item seq_item_sb;

    parameter FIFO_WIDTH = 16;
    parameter FIFO_DEPTH = 8;
    localparam max_fifo_addr = $clog2(FIFO_DEPTH);
    logic [FIFO_WIDTH-1:0] data_out_ref;
    logic [FIFO_WIDTH-1:0] Queue [$] ;
    logic [max_fifo_addr:0] count;

    int error_count = 0;
    int correct_count = 0;
  function new (string name = "fifo_sco" , uvm_component parent = null);
    super.new (name,parent);
  endfunction

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

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

  task run_phase (uvm_phase phase);
   super.run_phase(phase);
   forever begin
    sb_fifo.get(seq_item_sb);
    ref_model(seq_item_sb);
    if ((seq_item_sb.data_out != data_out_ref)) begin
        `uvm_error("run_phase",$sformatf("compartion failled while ref = %b",data_out_ref))
        error_count++;
    end
    else
       correct_count++;
   end
```

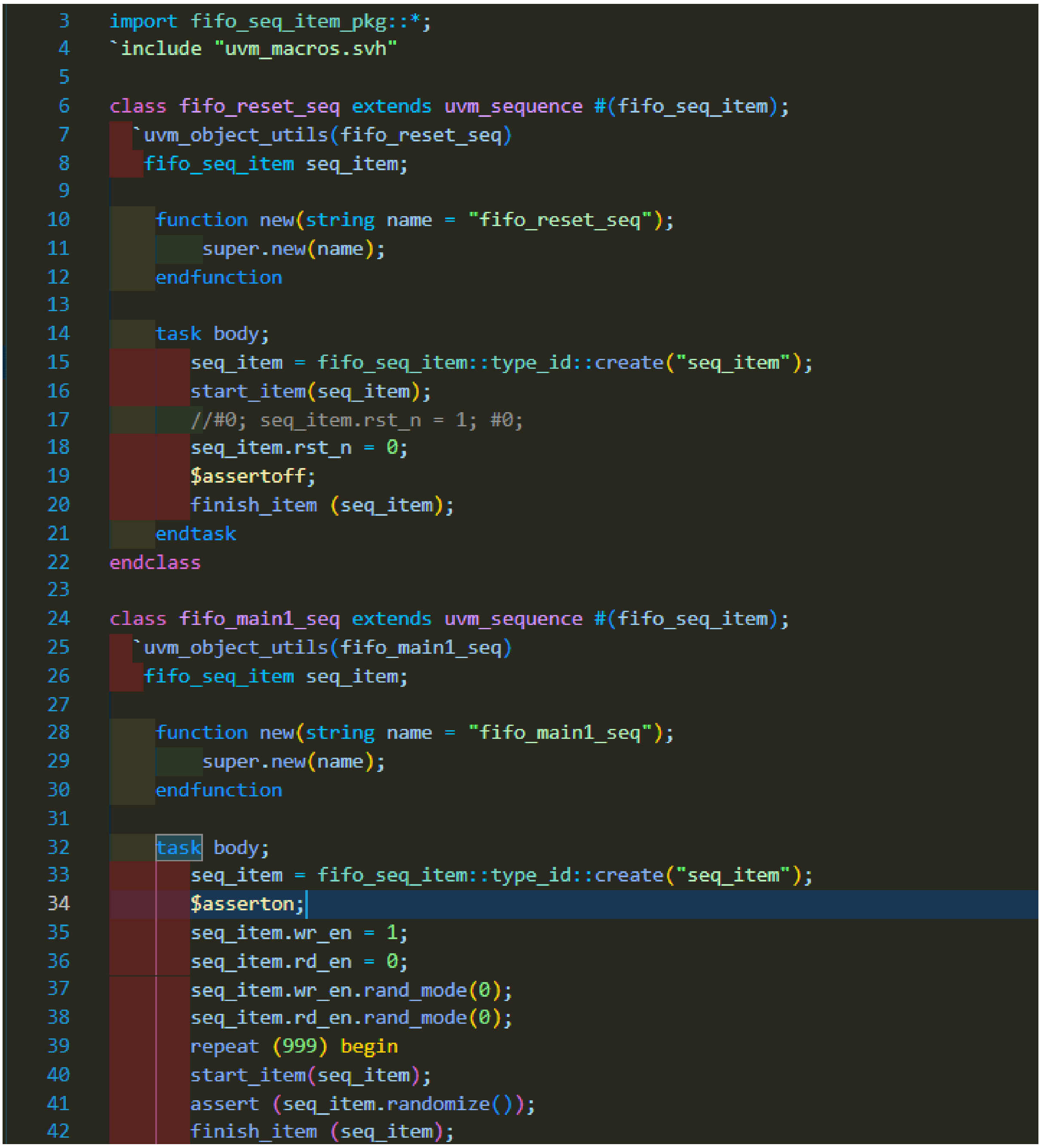


```
  end
endtask

task ref_model (fifo_seq_item seq_item_chk);

    if (!seq_item_chk.rst_n) begin
      Queue <= {};
      count <= 0;
    end
    else begin

      if (seq_item_chk.wr_en && count <  FIFO_DEPTH) begin
        Queue.push_back(seq_item_chk.data_in);
        count <= Queue.size();
      end

      if (seq_item_chk.rd_en && count != 0) begin
        data_out_ref <= Queue.pop_front();
        count <= Queue.size();
      end

    end

endtask

function void report_phase (uvm_phase phase);
 super.report_phase(phase);
 `uvm_info("report_phase",$sformatf("corect = %d",correct_count) , UVM_MEDIUM)
 `uvm_info("report_phase",$sformatf("error = %d",error_count) , UVM_MEDIUM)
endfunction
endclass
endpackage
```

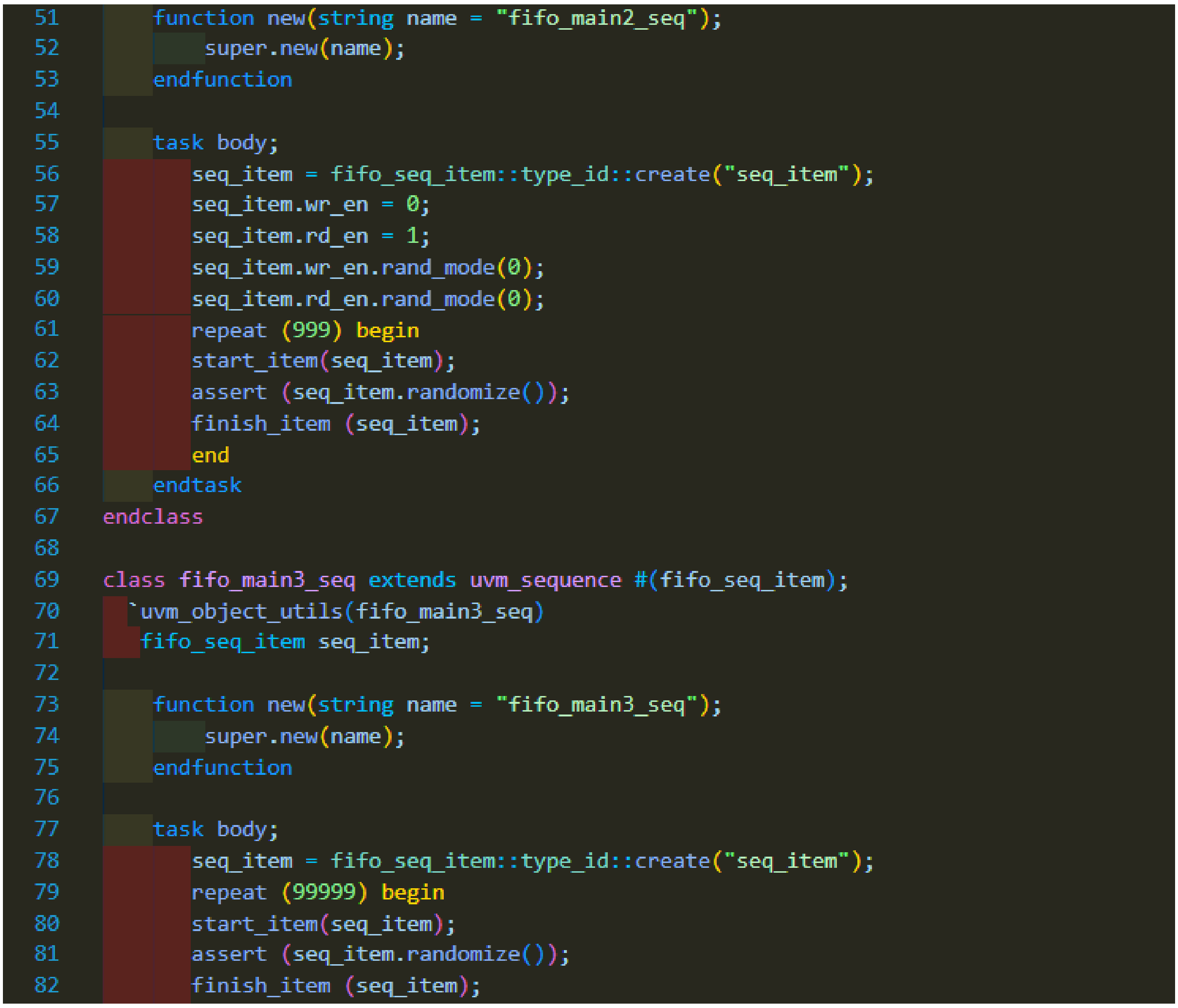


fifo_config.sv

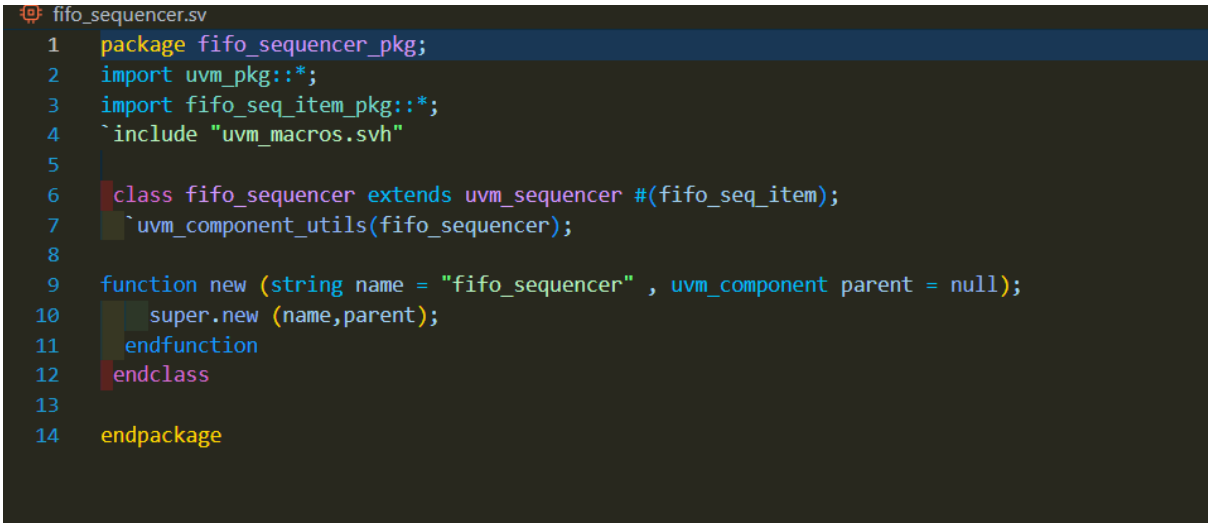
```
package fifo_conf_pkg;
import uvm_pkg::*;
`include "uvm_macros.svh"

  class fifo_config extends uvm_object;
    `uvm_object_utils(fifo_config)

    virtual fifo_if fifo_vif;

    function new (string name = "fifo_config");
      super.new(name);

    endfunction

  endclass
endpackage
```



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

class fifo_reset_seq extends uvm_sequence #(fifo_seq_item);
  `uvm_object_utils(fifo_reset_seq)
   fifo_seq_item seq_item;

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

    task body;
       seq_item = fifo_seq_item::type_id::create("seq_item");
       start_item(seq_item);
       //#0; seq_item.rst_n = 1; #0;
       seq_item.rst_n = 0;
       $assertoff;
       finish_item (seq_item);
    endtask
endclass

class fifo_main1_seq extends uvm_sequence #(fifo_seq_item);
  `uvm_object_utils(fifo_main1_seq)
   fifo_seq_item seq_item;

    function new(string name = "fifo_main1_seq");
        super.new(name);
    endfunction

    task body;
       seq_item = fifo_seq_item::type_id::create("seq_item");
       $asserton;
       seq_item.wr_en = 1;
       seq_item.rd_en = 0;
       seq_item.wr_en.rand_mode(0);
       seq_item.rd_en.rand_mode(0);
       repeat (999) begin
       start_item(seq_item);
       assert (seq_item.randomize());
       finish_item (seq_item);
```

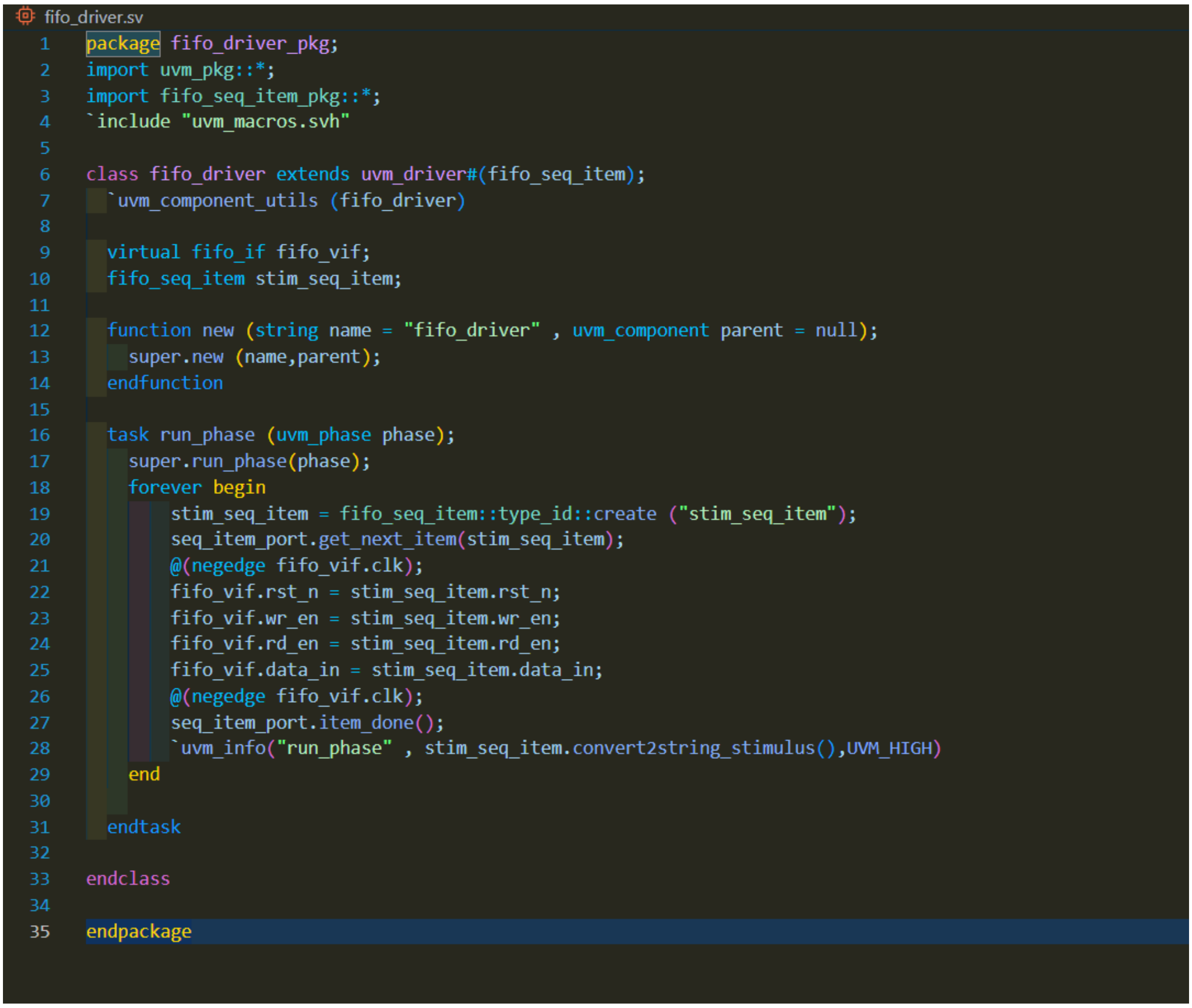


```systemverilog
    function new(string name = "fifo_main2_seq");
        super.new(name);
    endfunction

    task body;
       seq_item = fifo_seq_item::type_id::create("seq_item");
       seq_item.wr_en = 0;
       seq_item.rd_en = 1;
       seq_item.wr_en.rand_mode(0);
       seq_item.rd_en.rand_mode(0);
       repeat (999) begin
       start_item(seq_item);
       assert (seq_item.randomize());
       finish_item (seq_item);
       end
    endtask
endclass

class fifo_main3_seq extends uvm_sequence #(fifo_seq_item);
  `uvm_object_utils(fifo_main3_seq)
   fifo_seq_item seq_item;

    function new(string name = "fifo_main3_seq");
        super.new(name);
    endfunction

    task body;
       seq_item = fifo_seq_item::type_id::create("seq_item");
       repeat (99999) begin
       start_item(seq_item);
       assert (seq_item.randomize());
       finish_item (seq_item);
```

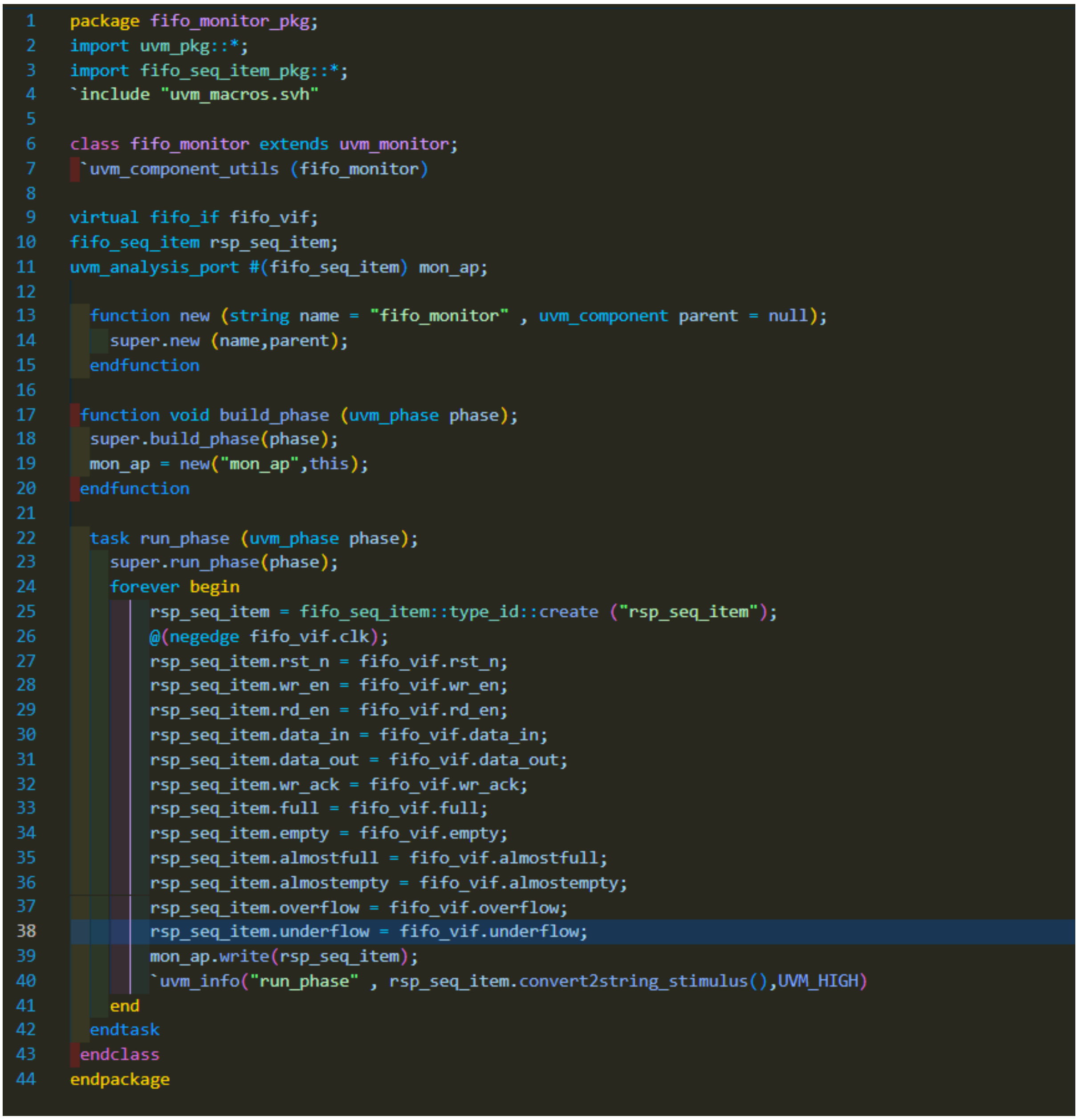


fifo_sequencer.sv

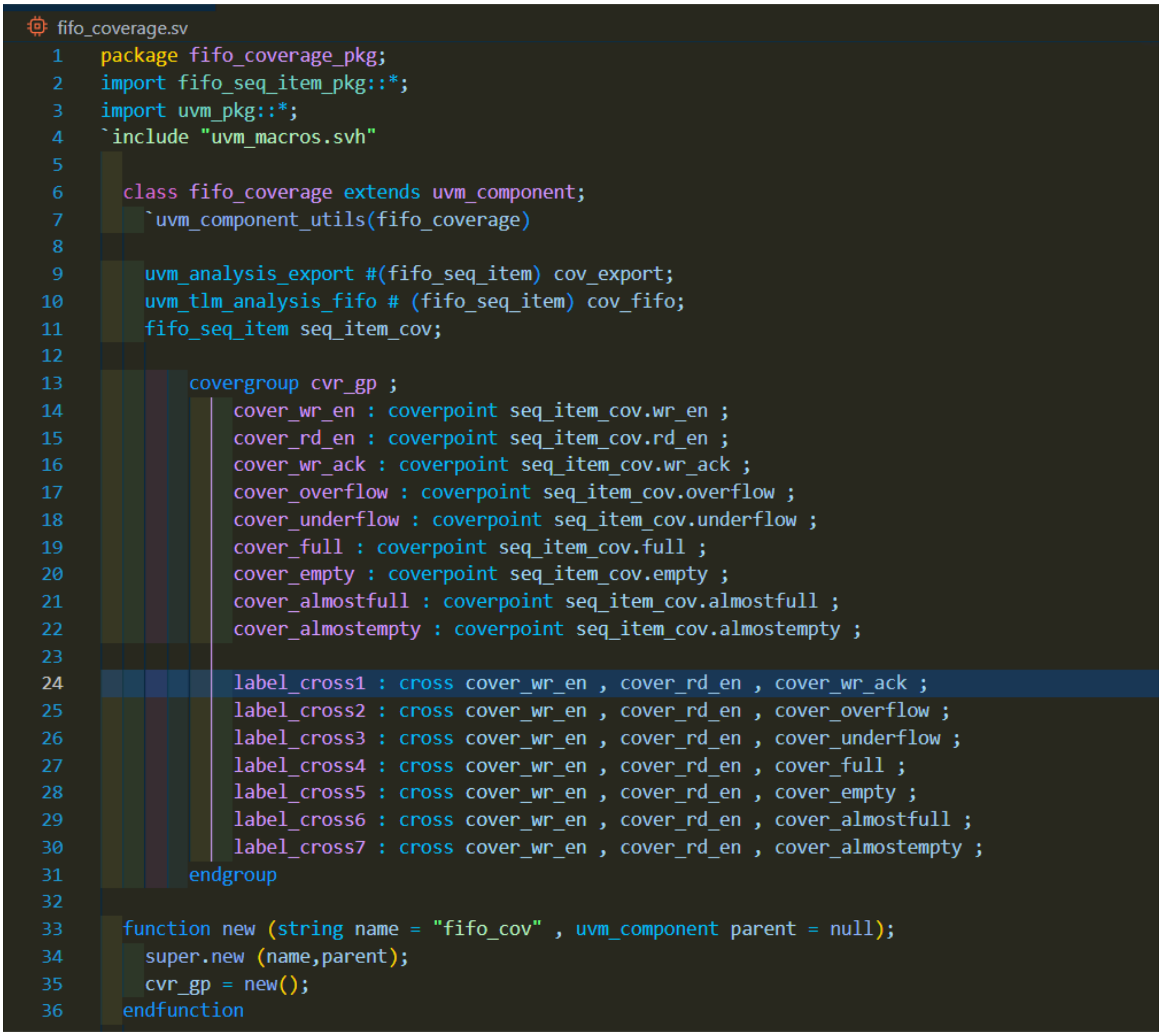
```systemverilog
package fifo_sequencer_pkg;
import uvm_pkg::*;
import fifo_seq_item_pkg::*;
`include "uvm_macros.svh"

 class fifo_sequencer extends uvm_sequencer #(fifo_seq_item);
  `uvm_component_utils(fifo_sequencer);

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

endpackage
```



```systemverilog
package fifo_driver_pkg;
import uvm_pkg::*;
import fifo_seq_item_pkg::*;
`include "uvm_macros.svh"

class fifo_driver extends uvm_driver#(fifo_seq_item);
  `uvm_component_utils (fifo_driver)

  virtual fifo_if fifo_vif;
  fifo_seq_item stim_seq_item;

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

  task run_phase (uvm_phase phase);
    super.run_phase(phase);
    forever begin
        stim_seq_item = fifo_seq_item::type_id::create ("stim_seq_item");
        seq_item_port.get_next_item(stim_seq_item);
        @(negedge fifo_vif.clk);
        fifo_vif.rst_n = stim_seq_item.rst_n;
        fifo_vif.wr_en = stim_seq_item.wr_en;
        fifo_vif.rd_en = stim_seq_item.rd_en;
        fifo_vif.data_in = stim_seq_item.data_in;
        @(negedge fifo_vif.clk);
        seq_item_port.item_done();
        `uvm_info("run_phase" , stim_seq_item.convert2string_stimulus(),UVM_HIGH)
    end

  endtask

endclass

endpackage
```

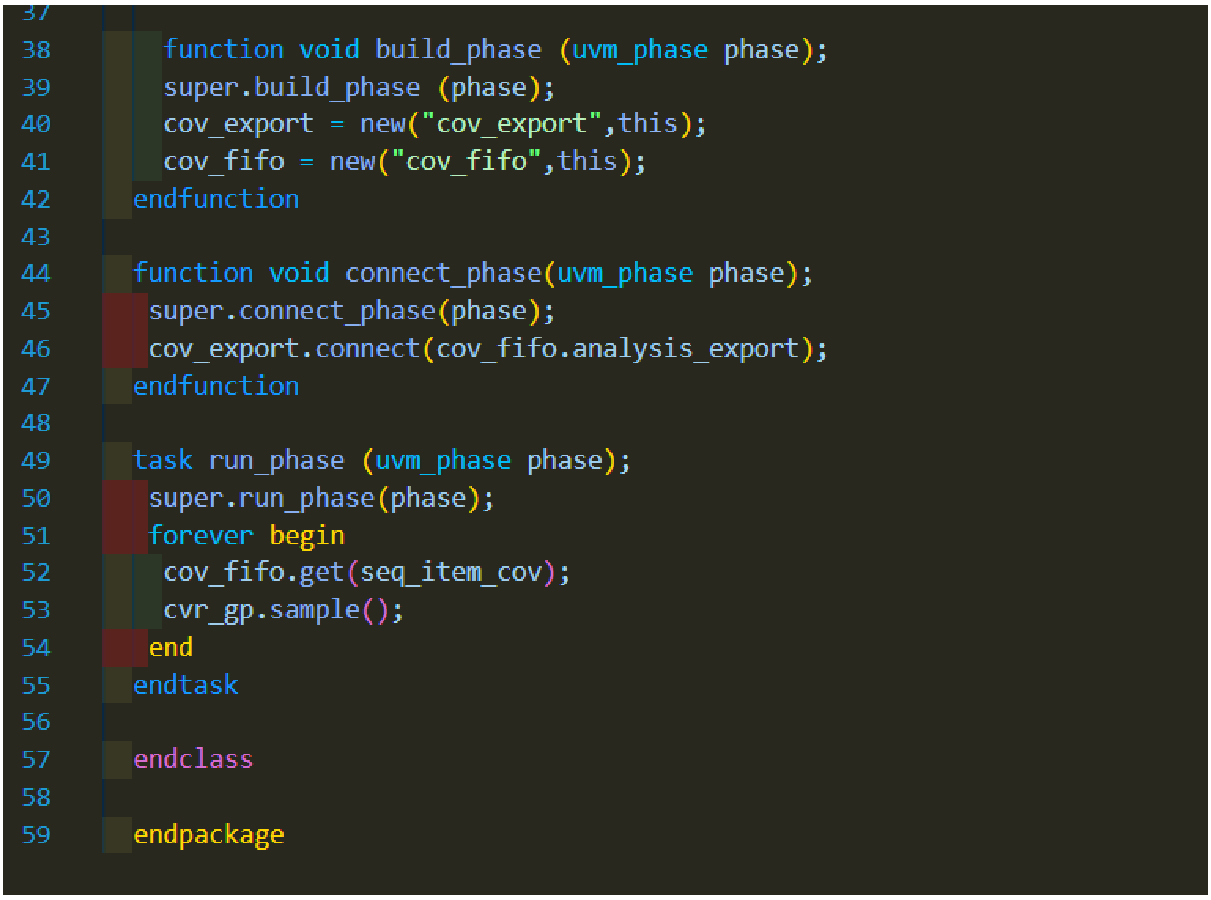


```
package fifo_monitor_pkg;
import uvm_pkg::*;
import fifo_seq_item_pkg::*;
`include "uvm_macros.svh"

class fifo_monitor extends uvm_monitor;
 `uvm_component_utils (fifo_monitor)

virtual fifo_if fifo_vif;
fifo_seq_item rsp_seq_item;
uvm_analysis_port #(fifo_seq_item) mon_ap;

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

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

  task run_phase (uvm_phase phase);
    super.run_phase(phase);
    forever begin
        rsp_seq_item = fifo_seq_item::type_id::create ("rsp_seq_item");
        @(negedge fifo_vif.clk);
        rsp_seq_item.rst_n = fifo_vif.rst_n;
        rsp_seq_item.wr_en = fifo_vif.wr_en;
        rsp_seq_item.rd_en = fifo_vif.rd_en;
        rsp_seq_item.data_in = fifo_vif.data_in;
        rsp_seq_item.data_out = fifo_vif.data_out;
        rsp_seq_item.wr_ack = fifo_vif.wr_ack;
        rsp_seq_item.full = fifo_vif.full;
        rsp_seq_item.empty = fifo_vif.empty;
        rsp_seq_item.almostfull = fifo_vif.almostfull;
        rsp_seq_item.almostempty = fifo_vif.almostempty;
        rsp_seq_item.overflow = fifo_vif.overflow;
        rsp_seq_item.underflow = fifo_vif.underflow;
        mon_ap.write(rsp_seq_item);
        `uvm_info("run_phase" , rsp_seq_item.convert2string_stimulus(),UVM_HIGH)
    end
  endtask
 endclass
endpackage
```

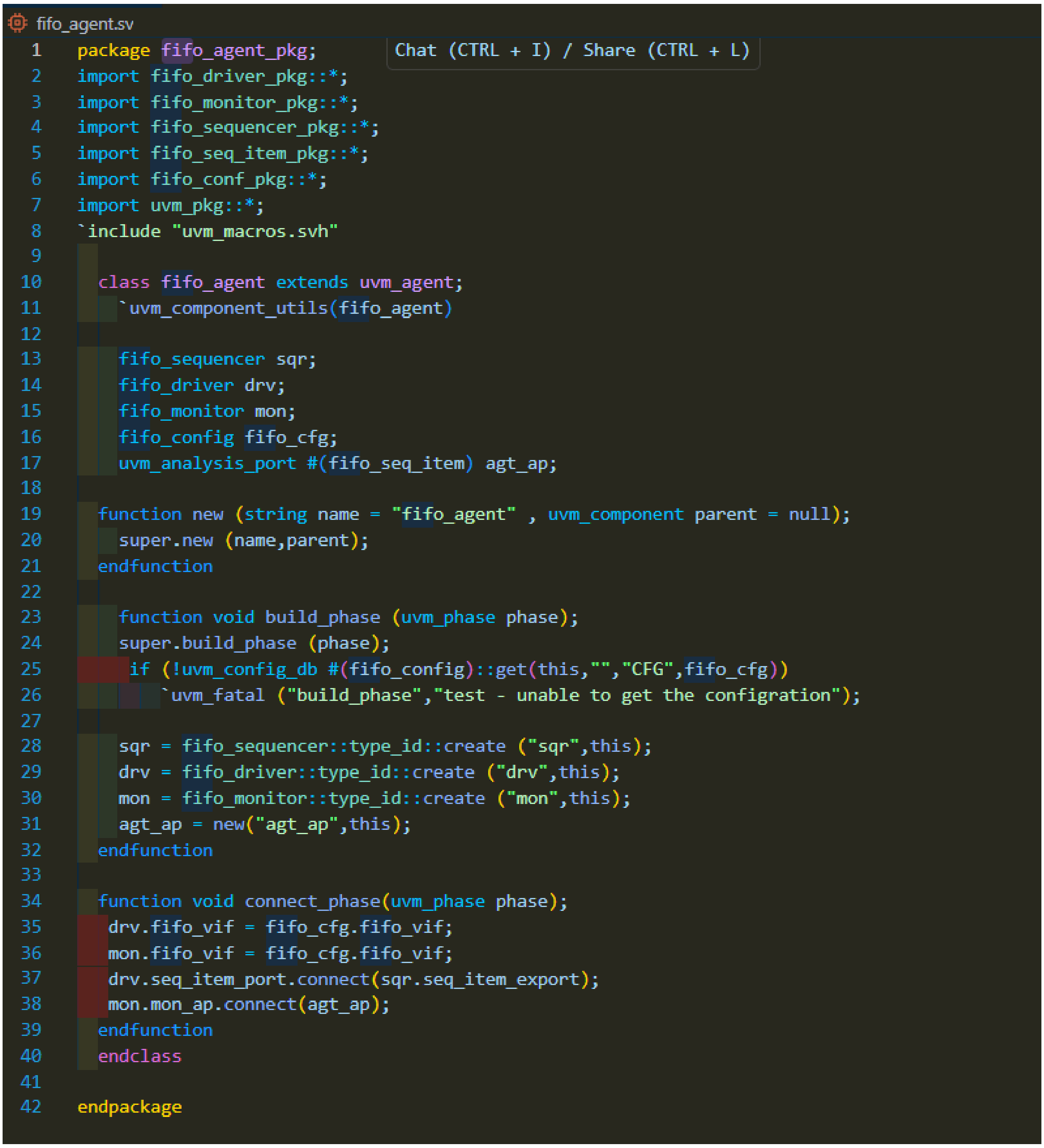


fifo_coverage.sv

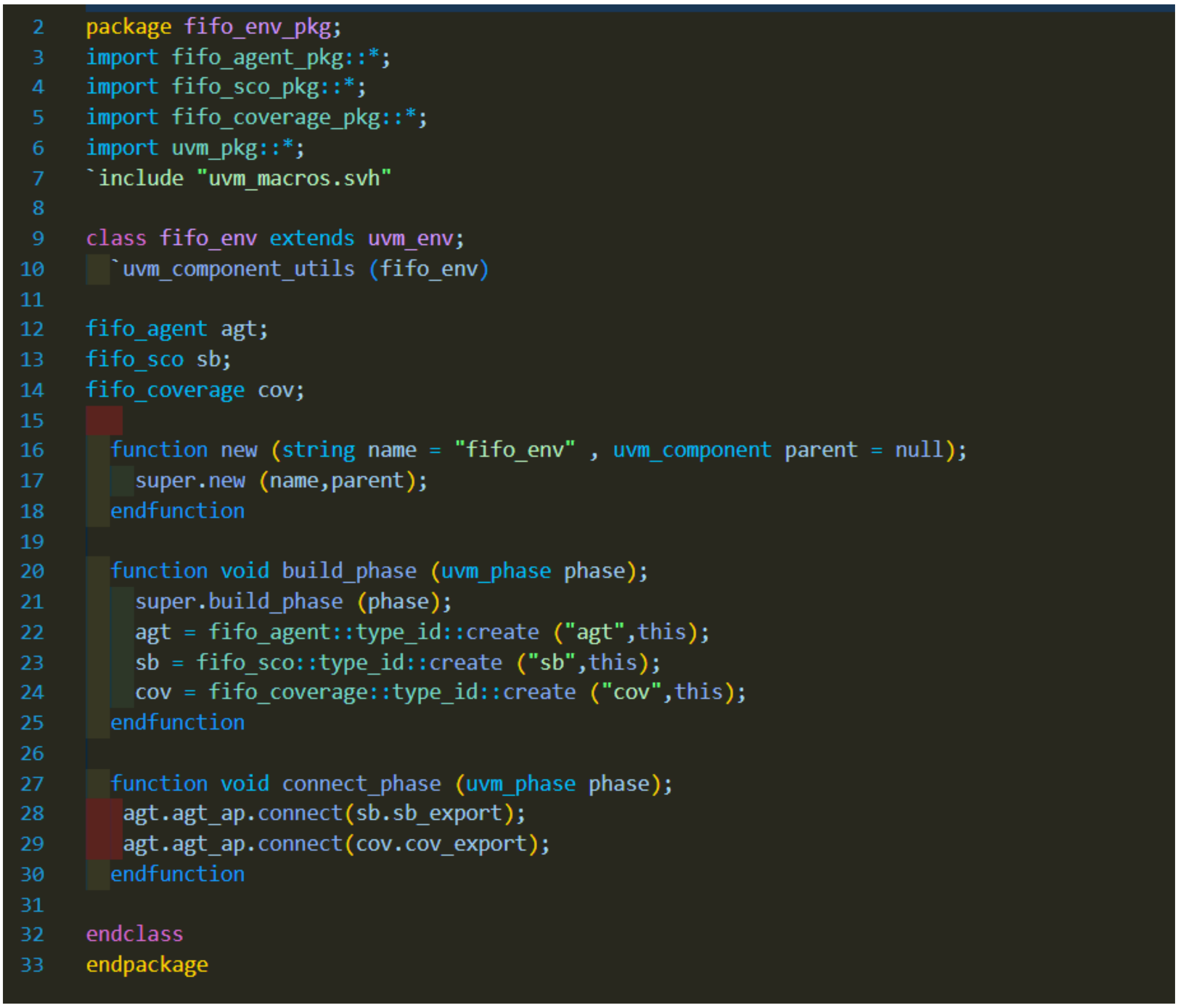
```
package fifo_coverage_pkg;
import fifo_seq_item_pkg::*;
import uvm_pkg::*;
`include "uvm_macros.svh"

  class fifo_coverage extends uvm_component;
    `uvm_component_utils(fifo_coverage)

    uvm_analysis_export #(fifo_seq_item) cov_export;
    uvm_tlm_analysis_fifo # (fifo_seq_item) cov_fifo;
    fifo_seq_item seq_item_cov;

        covergroup cvr_gp ;
            cover_wr_en : coverpoint seq_item_cov.wr_en ;
            cover_rd_en : coverpoint seq_item_cov.rd_en ;
            cover_wr_ack : coverpoint seq_item_cov.wr_ack ;
            cover_overflow : coverpoint seq_item_cov.overflow ;
            cover_underflow : coverpoint seq_item_cov.underflow ;
            cover_full : coverpoint seq_item_cov.full ;
            cover_empty : coverpoint seq_item_cov.empty ;
            cover_almostfull : coverpoint seq_item_cov.almostfull ;
            cover_almostempty : coverpoint seq_item_cov.almostempty ;

            label_cross1 : cross cover_wr_en , cover_rd_en , cover_wr_ack ;
            label_cross2 : cross cover_wr_en , cover_rd_en , cover_overflow ;
            label_cross3 : cross cover_wr_en , cover_rd_en , cover_underflow ;
            label_cross4 : cross cover_wr_en , cover_rd_en , cover_full ;
            label_cross5 : cross cover_wr_en , cover_rd_en , cover_empty ;
            label_cross6 : cross cover_wr_en , cover_rd_en , cover_almostfull ;
            label_cross7 : cross cover_wr_en , cover_rd_en , cover_almostempty ;
        endgroup

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



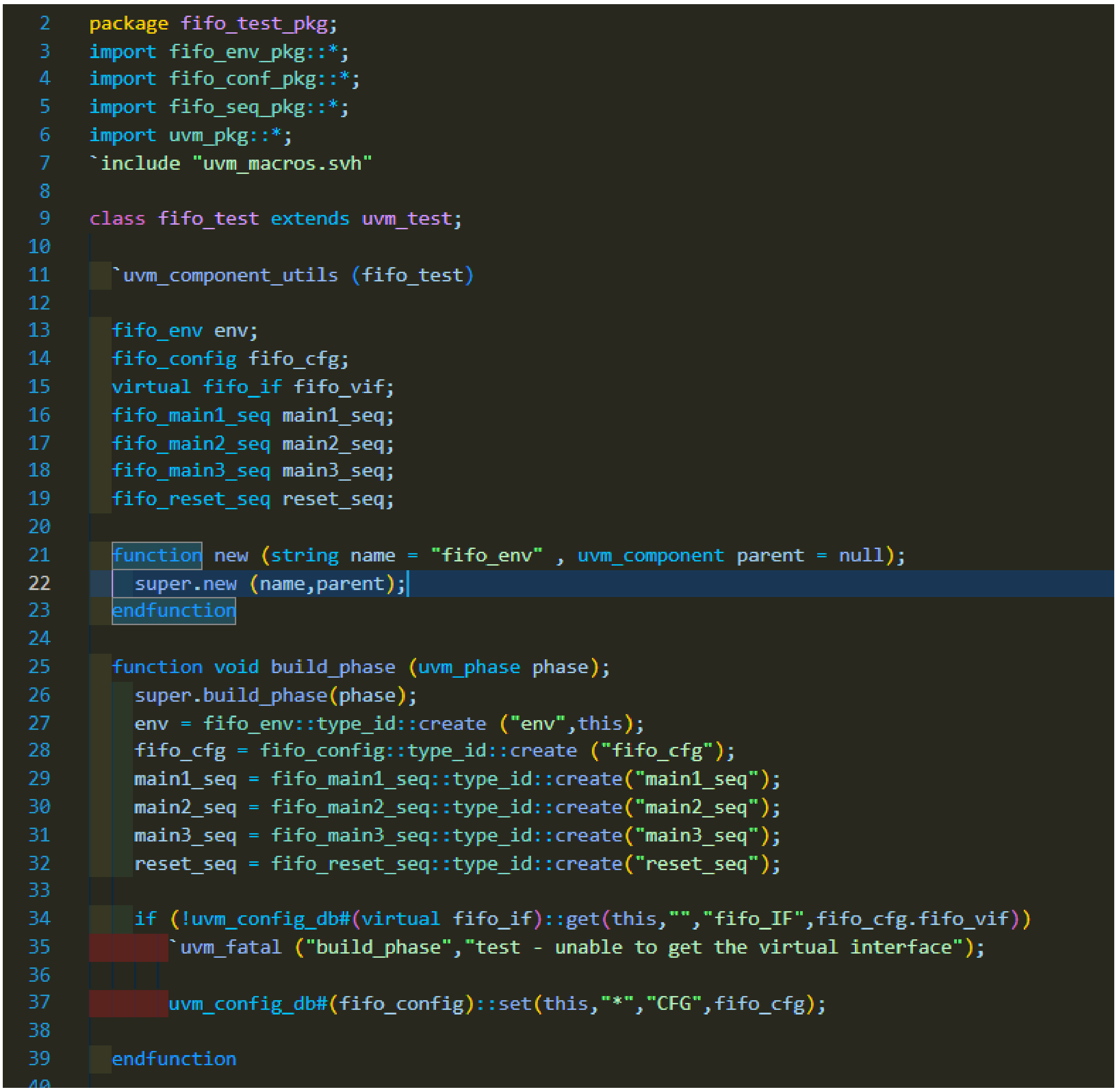
```
  function void build_phase (uvm_phase phase);
  super.build_phase (phase);
  cov_export = new("cov_export",this);
  cov_fifo = new("cov_fifo",this);
endfunction

function void connect_phase(uvm_phase phase);
 super.connect_phase(phase);
 cov_export.connect(cov_fifo.analysis_export);
endfunction

task run_phase (uvm_phase phase);
 super.run_phase(phase);
 forever begin
  cov_fifo.get(seq_item_cov);
  cvr_gp.sample();
 end
endtask

endclass

endpackage
```



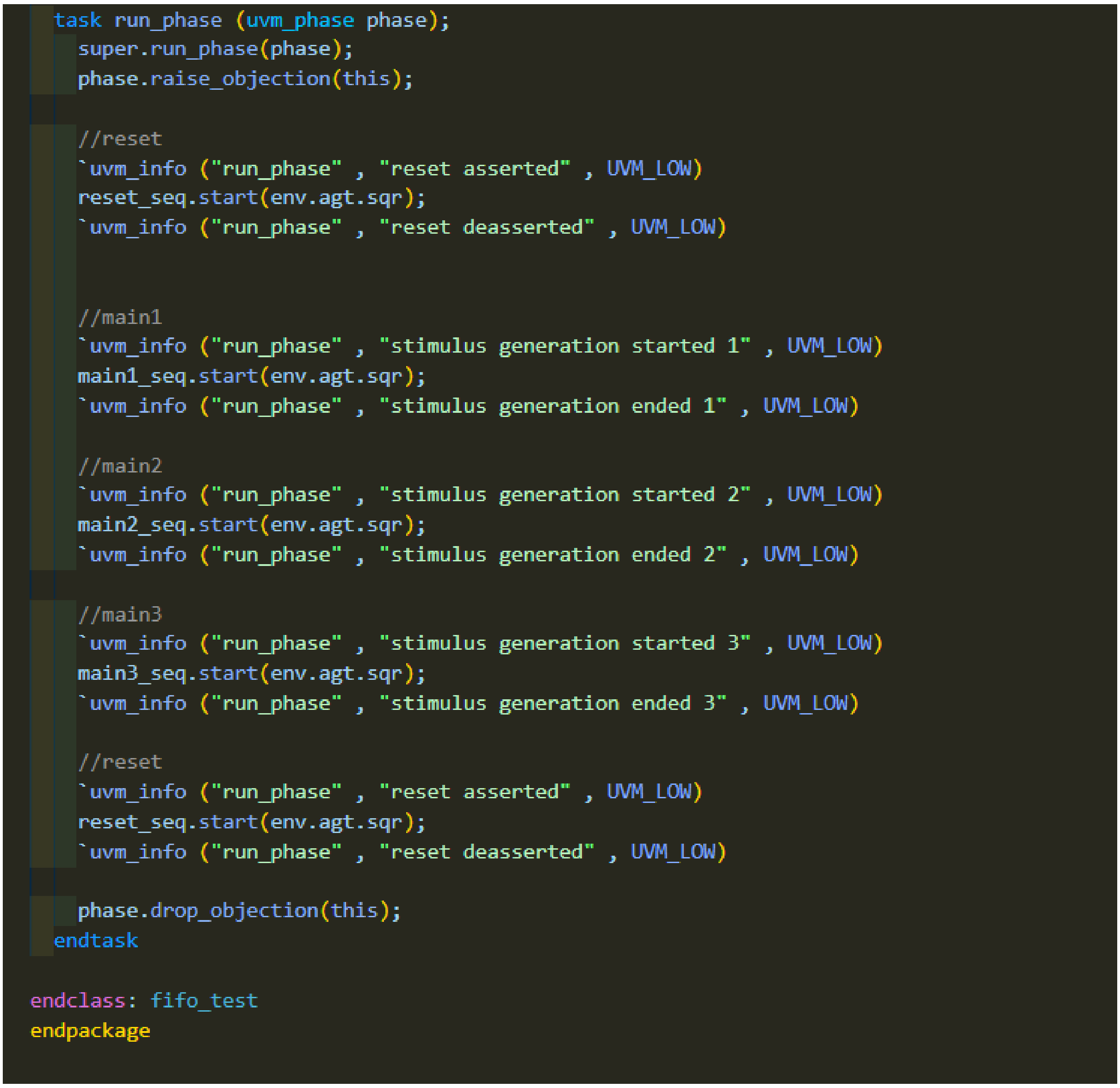
```
package fifo_agent_pkg;
import fifo_driver_pkg::*;
import fifo_monitor_pkg::*;
import fifo_sequencer_pkg::*;
import fifo_seq_item_pkg::*;
import fifo_conf_pkg::*;
import uvm_pkg::*;
`include "uvm_macros.svh"

  class fifo_agent extends uvm_agent;
    `uvm_component_utils(fifo_agent)

    fifo_sequencer sqr;
    fifo_driver drv;
    fifo_monitor mon;
    fifo_config fifo_cfg;
    uvm_analysis_port #(fifo_seq_item) agt_ap;

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

    function void build_phase (uvm_phase phase);
    super.build_phase (phase);
     if (!uvm_config_db #(fifo_config)::get(this,"","CFG",fifo_cfg))
        `uvm_fatal ("build_phase","test - unable to get the configration");

    sqr = fifo_sequencer::type_id::create ("sqr",this);
    drv = fifo_driver::type_id::create ("drv",this);
    mon = fifo_monitor::type_id::create ("mon",this);
    agt_ap = new("agt_ap",this);
  endfunction

  function void connect_phase(uvm_phase phase);
   drv.fifo_vif = fifo_cfg.fifo_vif;
   mon.fifo_vif = fifo_cfg.fifo_vif;
   drv.seq_item_port.connect(sqr.seq_item_export);
   mon.mon_ap.connect(agt_ap);
  endfunction
  endclass

endpackage
```



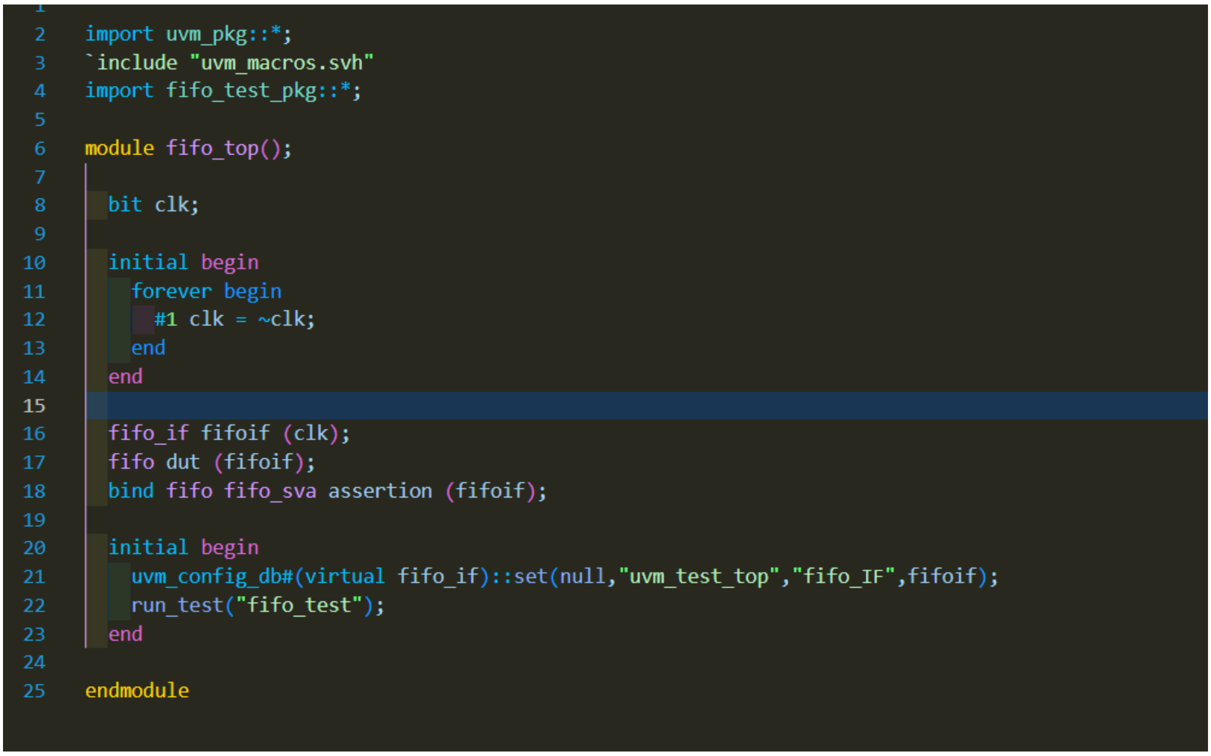
```systemverilog
package fifo_env_pkg;
import fifo_agent_pkg::*;
import fifo_sco_pkg::*;
import fifo_coverage_pkg::*;
import uvm_pkg::*;
`include "uvm_macros.svh"

class fifo_env extends uvm_env;
  `uvm_component_utils (fifo_env)

fifo_agent agt;
fifo_sco sb;
fifo_coverage cov;

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

  function void build_phase (uvm_phase phase);
    super.build_phase (phase);
    agt = fifo_agent::type_id::create ("agt",this);
    sb = fifo_sco::type_id::create ("sb",this);
    cov = fifo_coverage::type_id::create ("cov",this);
  endfunction

  function void connect_phase (uvm_phase phase);
   agt.agt_ap.connect(sb.sb_export);
   agt.agt_ap.connect(cov.cov_export);
  endfunction

endclass
endpackage
```



```
package fifo_test_pkg;
import fifo_env_pkg::*;
import fifo_conf_pkg::*;
import fifo_seq_pkg::*;
import uvm_pkg::*;
`include "uvm_macros.svh"

class fifo_test extends uvm_test;

  `uvm_component_utils (fifo_test)

  fifo_env env;
  fifo_config fifo_cfg;
  virtual fifo_if fifo_vif;
  fifo_main1_seq main1_seq;
  fifo_main2_seq main2_seq;
  fifo_main3_seq main3_seq;
  fifo_reset_seq reset_seq;

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

  function void build_phase (uvm_phase phase);
    super.build_phase(phase);
    env = fifo_env::type_id::create ("env",this);
    fifo_cfg = fifo_config::type_id::create ("fifo_cfg");
    main1_seq = fifo_main1_seq::type_id::create("main1_seq");
    main2_seq = fifo_main2_seq::type_id::create("main2_seq");
    main3_seq = fifo_main3_seq::type_id::create("main3_seq");
    reset_seq = fifo_reset_seq::type_id::create("reset_seq");

    if (!uvm_config_db#(virtual fifo_if)::get(this,"","fifo_IF",fifo_cfg.fifo_vif))
       `uvm_fatal ("build_phase","test - unable to get the virtual interface");

       uvm_config_db#(fifo_config)::set(this,"*","CFG",fifo_cfg);

  endfunction
```

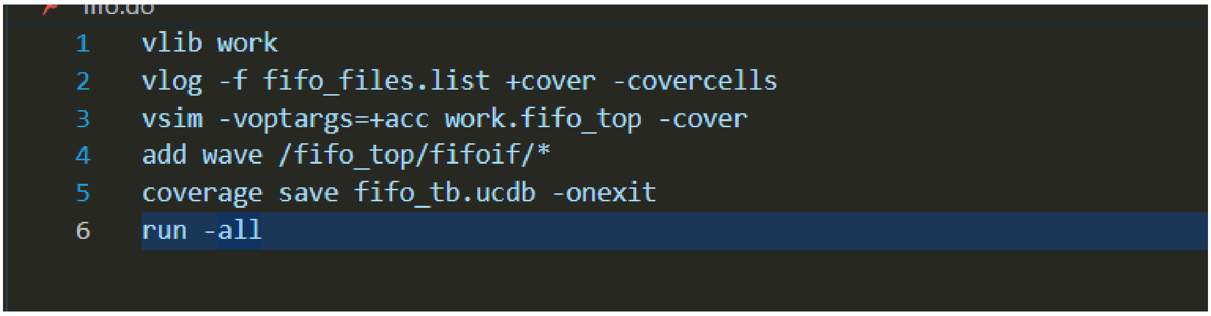


```
  task run_phase (uvm_phase phase);
    super.run_phase(phase);
    phase.raise_objection(this);

    //reset
    `uvm_info ("run_phase" , "reset asserted" , UVM_LOW)
    reset_seq.start(env.agt.sqr);
    `uvm_info ("run_phase" , "reset deasserted" , UVM_LOW)


    //main1
    `uvm_info ("run_phase" , "stimulus generation started 1" , UVM_LOW)
    main1_seq.start(env.agt.sqr);
    `uvm_info ("run_phase" , "stimulus generation ended 1" , UVM_LOW)

    //main2
    `uvm_info ("run_phase" , "stimulus generation started 2" , UVM_LOW)
    main2_seq.start(env.agt.sqr);
    `uvm_info ("run_phase" , "stimulus generation ended 2" , UVM_LOW)

    //main3
    `uvm_info ("run_phase" , "stimulus generation started 3" , UVM_LOW)
    main3_seq.start(env.agt.sqr);
    `uvm_info ("run_phase" , "stimulus generation ended 3" , UVM_LOW)

    //reset
    `uvm_info ("run_phase" , "reset asserted" , UVM_LOW)
    reset_seq.start(env.agt.sqr);
    `uvm_info ("run_phase" , "reset deasserted" , UVM_LOW)

    phase.drop_objection(this);
  endtask

endclass: fifo_test
endpackage
```



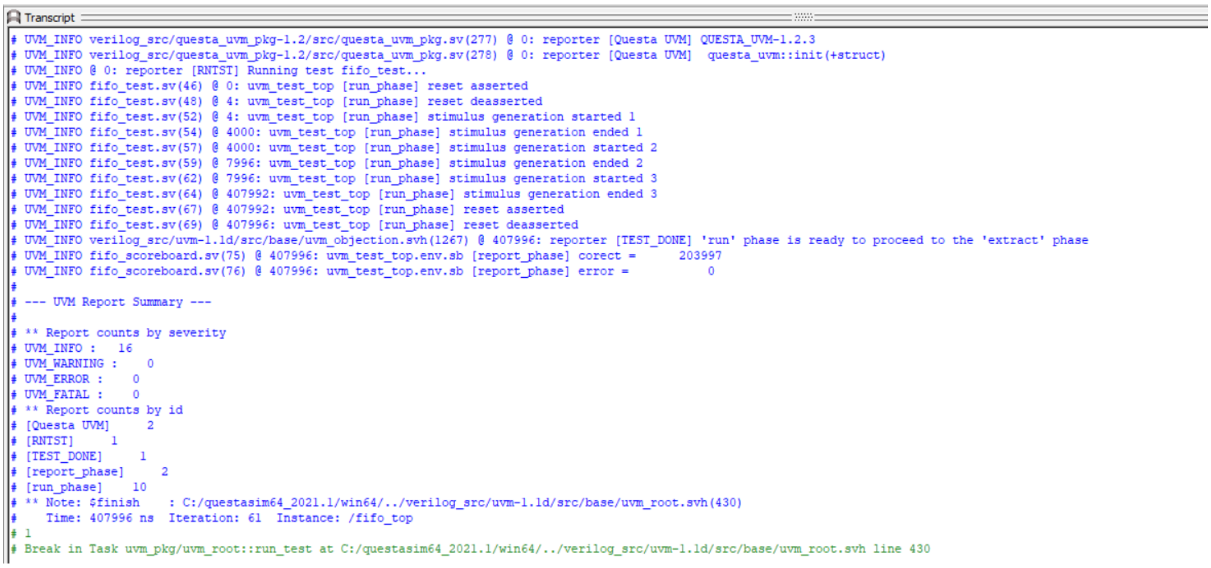
```systemverilog

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

module fifo_top();

  bit clk;

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

  fifo_if fifoif (clk);
  fifo dut (fifoif);
  bind fifo fifo_sva assertion (fifoif);

  initial begin
    uvm_config_db#(virtual fifo_if)::set(null,"uvm_test_top","fifo_IF",fifoif);
    run_test("fifo_test");
  end

endmodule
```

Do File :



```
vlib work
vlog -f fifo_files.list +cover -covercells
vsim -voptargs=+acc work.fifo_top -cover
add wave /fifo_top/fifoif/*
coverage save fifo_tb.ucdb -onexit
run -all
```

Display :



```
# 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(+struct)
# UVM_INFO @ 0: reporter [RNTST] Running test fifo_test...
# UVM_INFO fifo_test.sv(46) @ 0: uvm_test_top [run_phase] reset asserted
# UVM_INFO fifo_test.sv(48) @ 4: uvm_test_top [run_phase] reset deasserted
# UVM_INFO fifo_test.sv(52) @ 4: uvm_test_top [run_phase] stimulus generation started 1
# UVM_INFO fifo_test.sv(54) @ 4000: uvm_test_top [run_phase] stimulus generation ended 1
# UVM_INFO fifo_test.sv(57) @ 4000: uvm_test_top [run_phase] stimulus generation started 2
# UVM_INFO fifo_test.sv(59) @ 7996: uvm_test_top [run_phase] stimulus generation ended 2
# UVM_INFO fifo_test.sv(62) @ 7996: uvm_test_top [run_phase] stimulus generation started 3
# UVM_INFO fifo_test.sv(64) @ 407992: uvm_test_top [run_phase] stimulus generation ended 3
# UVM_INFO fifo_test.sv(67) @ 407992: uvm_test_top [run_phase] reset asserted
# UVM_INFO fifo_test.sv(69) @ 407996: uvm_test_top [run_phase] reset deasserted
# UVM_INFO verilog_src/uvm-1.1d/src/base/uvm_objection.svh(1267) @ 407996: reporter [TEST_DONE] 'run' phase is ready to proceed to the 'extract' phase
# UVM_INFO fifo_scoreboard.sv(75) @ 407996: uvm_test_top.env.sb [report_phase] corect =      203997
# UVM_INFO fifo_scoreboard.sv(76) @ 407996: uvm_test_top.env.sb [report_phase] error =           0
#
# --- UVM Report Summary ---
#
# ** Report counts by severity
# UVM_INFO :   16
# UVM_WARNING :    0
# UVM_ERROR :    0
# UVM_FATAL :    0
# ** Report counts by id
# [Questa UVM]     2
# [RNTST]     1
# [TEST_DONE]     1
# [report_phase]     2
# [run_phase]    10
# ** Note: $finish    : C:/questasim64_2021.1/win64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh(430)
#    Time: 407996 ns  Iteration: 61  Instance: /fifo_top
# 1
# Break in Task uvm_pkg/uvm_root::run_test at C:/questasim64_2021.1/win64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh line 430
```

Assertion :

Feature_1 : when rst_n is asserted , pointer and count = 0

Assertion :

```
(!rst_n) |=>
(wr_ptr == 0 && rd_ptr == 0 && count == 0);
```

Feature_2 : when write enable asserted and count less than fifo_depth (8) , wr_ack = 1

Assertion :

```
(wr_en && count < fifo_depth) |=>
(wr_ack == 1);
```

Feature_3 : when write enable asserted and full is high , overflow = 1

Assertion :

```
(wr_en && full) |=>
(overflow == 1);
```

Feature_4 : when read enable is asserted and empty is high , underflow = 1

Assertion :

```
(wr_en && empty) |=>
(underflow == 1);
```

Feature_5 : when count = 0 , empty = 1

Assertion :

```
(count == 0) |->
(empty == 1);
```

Feature_6 : when count = fifo_depth , full = 1

Assertion :

```
(count == fifo_depth) |->
(full == 1);
```

Feature_7 : when count = fifo_depth - 1 , almostfull = 1

Assertion :

```
(count == fifo_depth-1) |->
(almostfull == 1);
```

Feature_8 : when count = 1 , almostempty = 1

Assertion :

```
(count == 1) |->
(almostempty == 1);
```

Feature_9 : when wr_ptr = 7 , the next change must be 0

Assertion :

```
(wr_ptr == 7) |->
(wr_ptr == 0) [=1];
```

Feature_10 : when rd_ptr = 7 , the next change must be 0

Assertion :

```
(rd_ptr == 7) |->
(wr_ptr == 0) [=1];
```

Feature_11 : when count = 8 , the next change must be 0 or 7

Assertion :

```
(count == 8) |->
((count == 7 || count == 0)) [=1];
```

Feature_12 : when wr_en is asserted and count less than fifo_depth , mem take the value in data_in

Assertion :

```
(wr_en && count < fifo_depth) |=>
(mem[wr_ptr] = $past(data_in));
```

Feature_13 : wr_ptr should be less than fifo_depth

Assertion :

```
(wr_ptr < fifo_depth)
```

Feature_14 : rd_ptr should be less than fifo_depth

Assertion :

```
(rd_ptr < fifo_depth)
```

Feature_15 : count should be less or equal than fifo_depth

Assertion :

```
(count <= fifo_depth )
```

Feature_16 : when wr_en is high and rd_en is low and full is low or wr_en is high and rd_en is high and empty is high , count increment

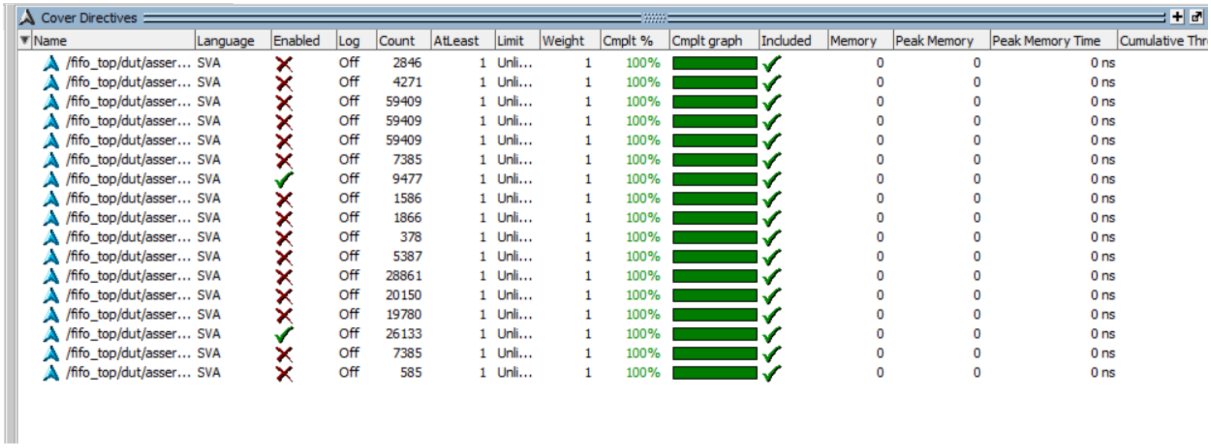
Assertion :

```
(({wr_rn,rd_en}==2’b10 && !full) || ({wr_rn,rd_en}==2’b11 &&
empty)) |=>
(count == $past(count) + 1);
```

Feature_17 : when rd_en is high and wr_en is low and empty is low or rd_en is high and wr_en is high and full is high , count decrement

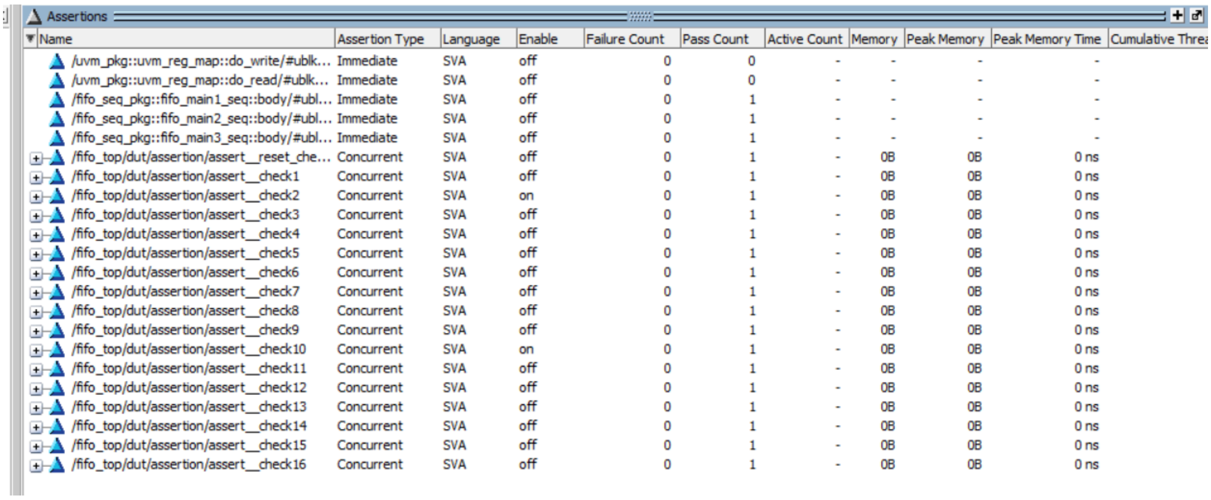
Assertion :

```
(({wr_rn,rd_en}==2’b01 && !empty) || ({wr_rn,rd_en}==2’b11 &&
full)) |=>
(count == $past(count) - 1);
```



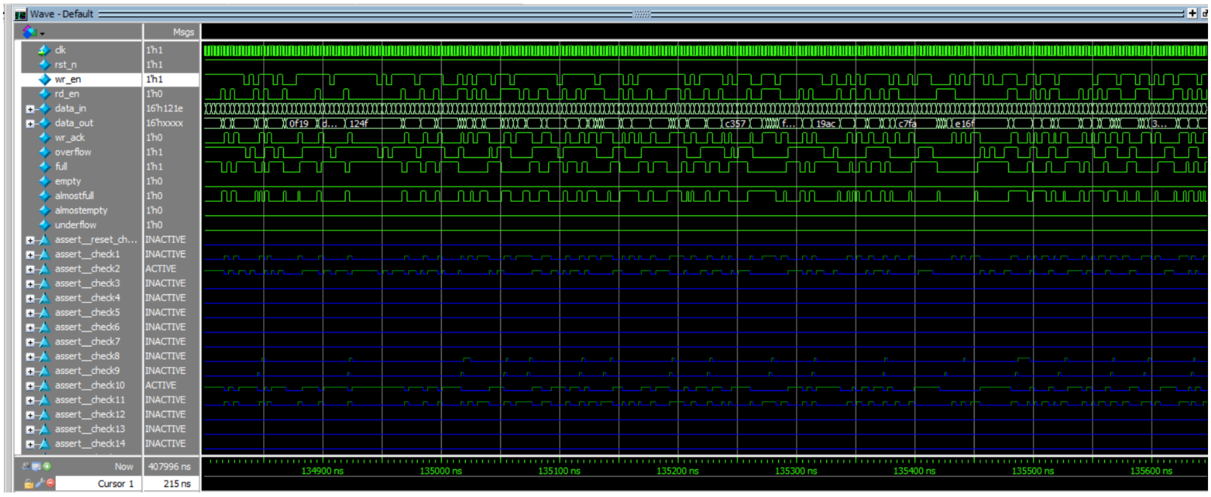
Cover Directives

Name	Language	Enabled	Log	Count	AtLeast	Limit	Weight	Cmplt %	Cmplt graph	Included	Memory	Peak Memory	Peak Memory Time	Cumulative Thr
/fifo_top/dut/asser...	SVA	✗	Off	2846	1	Unli...	1	100%	🟩	✓	0	0	0 ns	
/fifo_top/dut/asser...	SVA	✗	Off	4271	1	Unli...	1	100%	🟩	✓	0	0	0 ns	
/fifo_top/dut/asser...	SVA	✗	Off	59409	1	Unli...	1	100%	🟩	✓	0	0	0 ns	
/fifo_top/dut/asser...	SVA	✗	Off	59409	1	Unli...	1	100%	🟩	✓	0	0	0 ns	
/fifo_top/dut/asser...	SVA	✗	Off	59409	1	Unli...	1	100%	🟩	✓	0	0	0 ns	
/fifo_top/dut/asser...	SVA	✗	Off	7385	1	Unli...	1	100%	🟩	✓	0	0	0 ns	
/fifo_top/dut/asser...	SVA	✓	Off	9477	1	Unli...	1	100%	🟩	✓	0	0	0 ns	
/fifo_top/dut/asser...	SVA	✗	Off	1586	1	Unli...	1	100%	🟩	✓	0	0	0 ns	
/fifo_top/dut/asser...	SVA	✗	Off	1866	1	Unli...	1	100%	🟩	✓	0	0	0 ns	
/fifo_top/dut/asser...	SVA	✗	Off	378	1	Unli...	1	100%	🟩	✓	0	0	0 ns	
/fifo_top/dut/asser...	SVA	✗	Off	5387	1	Unli...	1	100%	🟩	✓	0	0	0 ns	
/fifo_top/dut/asser...	SVA	✗	Off	28861	1	Unli...	1	100%	🟩	✓	0	0	0 ns	
/fifo_top/dut/asser...	SVA	✗	Off	20150	1	Unli...	1	100%	🟩	✓	0	0	0 ns	
/fifo_top/dut/asser...	SVA	✗	Off	19780	1	Unli...	1	100%	🟩	✓	0	0	0 ns	
/fifo_top/dut/asser...	SVA	✓	Off	26133	1	Unli...	1	100%	🟩	✓	0	0	0 ns	
/fifo_top/dut/asser...	SVA	✗	Off	7385	1	Unli...	1	100%	🟩	✓	0	0	0 ns	
/fifo_top/dut/asser...	SVA	✗	Off	585	1	Unli...	1	100%	🟩	✓	0	0	0 ns	



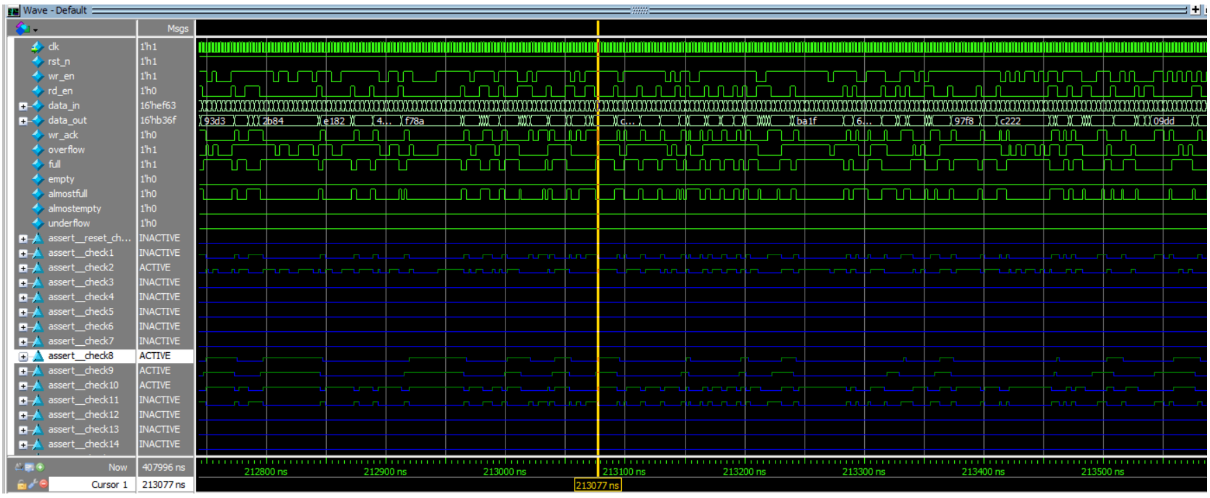
Assertions

Name	Assertion Type	Language	Enable	Failure Count	Pass Count	Active Count	Memory	Peak Memory	Peak Memory Time	Cumulative Threa
/uvm_pkg::uvm_reg_map::do_write/#ublk...	Immediate	SVA	off	0	0	-	-	-	-	
/uvm_pkg::uvm_reg_map::do_read/#ublk...	Immediate	SVA	off	0	0	-	-	-	-	
/fifo_seq_pkg::fifo_main1_seq::body/#ubl...	Immediate	SVA	off	0	1	-	-	-	-	
/fifo_seq_pkg::fifo_main2_seq::body/#ubl...	Immediate	SVA	off	0	1	-	-	-	-	
/fifo_seq_pkg::fifo_main3_seq::body/#ubl...	Immediate	SVA	off	0	1	-	-	-	-	
/fifo_top/dut/assertion/assert__reset_che...	Concurrent	SVA	off	0	1	-	0B	0B	0 ns	
/fifo_top/dut/assertion/assert__check1	Concurrent	SVA	off	0	1	-	0B	0B	0 ns	
/fifo_top/dut/assertion/assert__check2	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	
/fifo_top/dut/assertion/assert__check3	Concurrent	SVA	off	0	1	-	0B	0B	0 ns	
/fifo_top/dut/assertion/assert__check4	Concurrent	SVA	off	0	1	-	0B	0B	0 ns	
/fifo_top/dut/assertion/assert__check5	Concurrent	SVA	off	0	1	-	0B	0B	0 ns	
/fifo_top/dut/assertion/assert__check6	Concurrent	SVA	off	0	1	-	0B	0B	0 ns	
/fifo_top/dut/assertion/assert__check7	Concurrent	SVA	off	0	1	-	0B	0B	0 ns	
/fifo_top/dut/assertion/assert__check8	Concurrent	SVA	off	0	1	-	0B	0B	0 ns	
/fifo_top/dut/assertion/assert__check9	Concurrent	SVA	off	0	1	-	0B	0B	0 ns	
/fifo_top/dut/assertion/assert__check10	Concurrent	SVA	on	0	1	-	0B	0B	0 ns	
/fifo_top/dut/assertion/assert__check11	Concurrent	SVA	off	0	1	-	0B	0B	0 ns	
/fifo_top/dut/assertion/assert__check12	Concurrent	SVA	off	0	1	-	0B	0B	0 ns	
/fifo_top/dut/assertion/assert__check13	Concurrent	SVA	off	0	1	-	0B	0B	0 ns	
/fifo_top/dut/assertion/assert__check14	Concurrent	SVA	off	0	1	-	0B	0B	0 ns	
/fifo_top/dut/assertion/assert__check15	Concurrent	SVA	off	0	1	-	0B	0B	0 ns	
/fifo_top/dut/assertion/assert__check16	Concurrent	SVA	off	0	1	-	0B	0B	0 ns	



Wave - Default

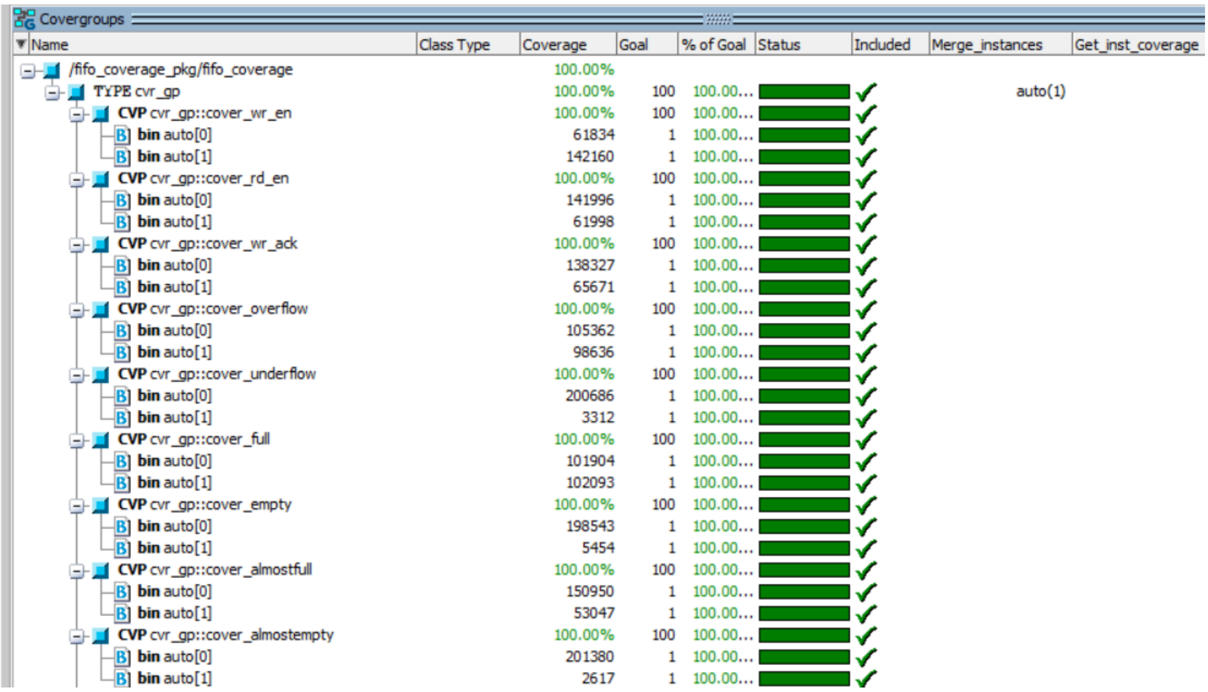
	Msgs
clk	1'h1
rst_n	1'h1
wr_en	1'h1
rd_en	1'h0
data_in	16'h121e
data_out	16'hxxxx
wr_ack	1'h0
overflow	1'h1
full	1'h1
empty	1'h0
almostfull	1'h0
almostempty	1'h0
underflow	1'h0
assert__reset_ch...	INACTIVE
assert__check1	INACTIVE
assert__check2	ACTIVE
assert__check3	INACTIVE
assert__check4	INACTIVE
assert__check5	INACTIVE
assert__check6	INACTIVE
assert__check7	INACTIVE
assert__check8	INACTIVE
assert__check9	INACTIVE
assert__check10	ACTIVE
assert__check11	INACTIVE
assert__check12	INACTIVE
assert__check13	INACTIVE
assert__check14	INACTIVE
Now	407996 ns
Cursor 1	215 ns



Wave - Default

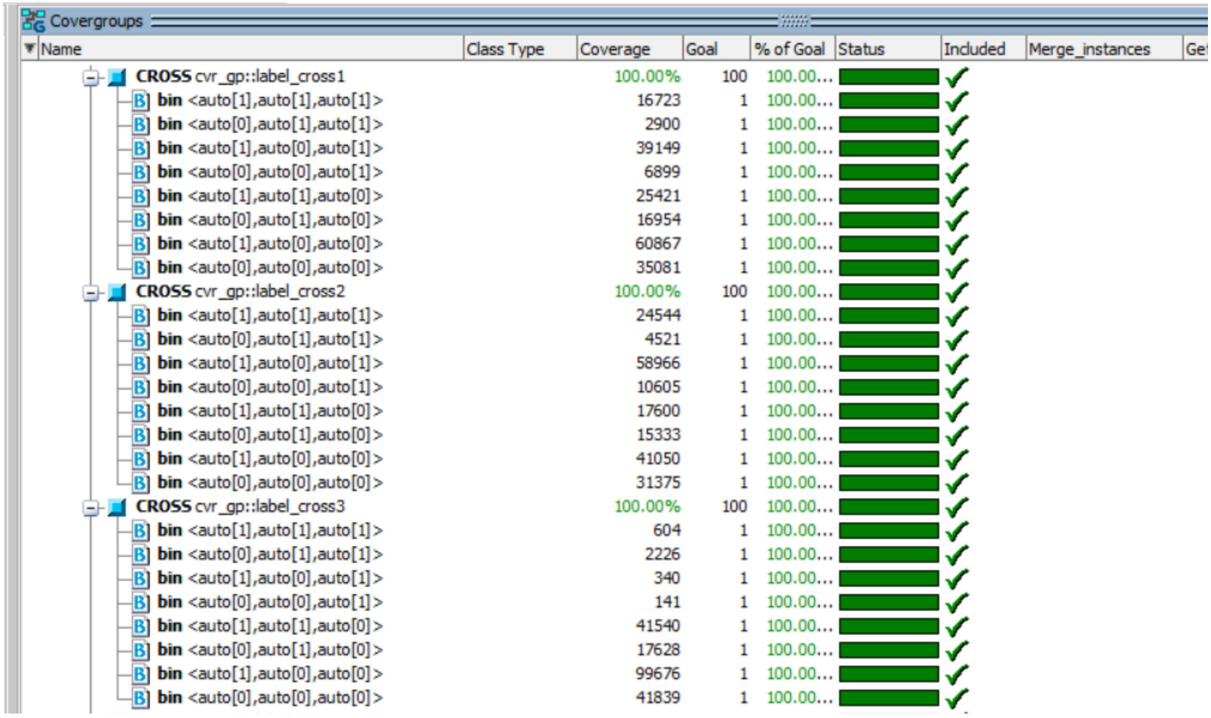
	Msgs
clk	1'h1
rst_n	1'h1
wr_en	1'h1
rd_en	1'h0
data_in	16'hef63
data_out	16'hb36f
wr_ack	1'h0
overflow	1'h1
full	1'h1
empty	1'h0
almostfull	1'h0
almostempty	1'h0
underflow	1'h0
assert__reset_ch...	INACTIVE
assert__check1	INACTIVE
assert__check2	ACTIVE
assert__check3	INACTIVE
assert__check4	INACTIVE
assert__check5	INACTIVE
assert__check6	INACTIVE
assert__check7	INACTIVE
assert__check8	ACTIVE
assert__check9	ACTIVE
assert__check10	ACTIVE
assert__check11	INACTIVE
assert__check12	INACTIVE
assert__check13	INACTIVE
assert__check14	INACTIVE
Now	407996 ns
Cursor 1	213077 ns

Covergroup :



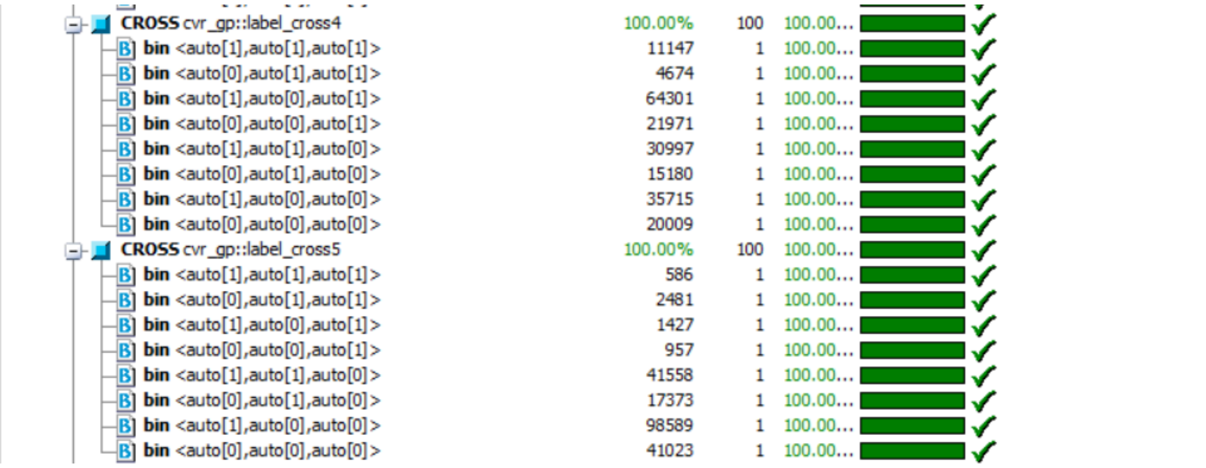
Covergroups

Name	Class Type	Coverage	Goal	% of Goal	Status	Included	Merge_instances	Get_inst_coverage
/fifo_coverage_pkg/fifo_coverage		100.00%						
TYPE cvr_gp		100.00%	100	100.00...	🟩	✓	auto(1)	
CVP cvr_gp::cover_wr_en		100.00%	100	100.00...	🟩	✓		
bin auto[0]		61834	1	100.00...	🟩	✓		
bin auto[1]		142160	1	100.00...	🟩	✓		
CVP cvr_gp::cover_rd_en		100.00%	100	100.00...	🟩	✓		
bin auto[0]		141996	1	100.00...	🟩	✓		
bin auto[1]		61998	1	100.00...	🟩	✓		
CVP cvr_gp::cover_wr_ack		100.00%	100	100.00...	🟩	✓		
bin auto[0]		138327	1	100.00...	🟩	✓		
bin auto[1]		65671	1	100.00...	🟩	✓		
CVP cvr_gp::cover_overflow		100.00%	100	100.00...	🟩	✓		
bin auto[0]		105362	1	100.00...	🟩	✓		
bin auto[1]		98636	1	100.00...	🟩	✓		
CVP cvr_gp::cover_underflow		100.00%	100	100.00...	🟩	✓		
bin auto[0]		200686	1	100.00...	🟩	✓		
bin auto[1]		3312	1	100.00...	🟩	✓		
CVP cvr_gp::cover_full		100.00%	100	100.00...	🟩	✓		
bin auto[0]		101904	1	100.00...	🟩	✓		
bin auto[1]		102093	1	100.00...	🟩	✓		
CVP cvr_gp::cover_empty		100.00%	100	100.00...	🟩	✓		
bin auto[0]		198543	1	100.00...	🟩	✓		
bin auto[1]		5454	1	100.00...	🟩	✓		
CVP cvr_gp::cover_almostfull		100.00%	100	100.00...	🟩	✓		
bin auto[0]		150950	1	100.00...	🟩	✓		
bin auto[1]		53047	1	100.00...	🟩	✓		
CVP cvr_gp::cover_almostempty		100.00%	100	100.00...	🟩	✓		
bin auto[0]		201380	1	100.00...	🟩	✓		
bin auto[1]		2617	1	100.00...	🟩	✓		

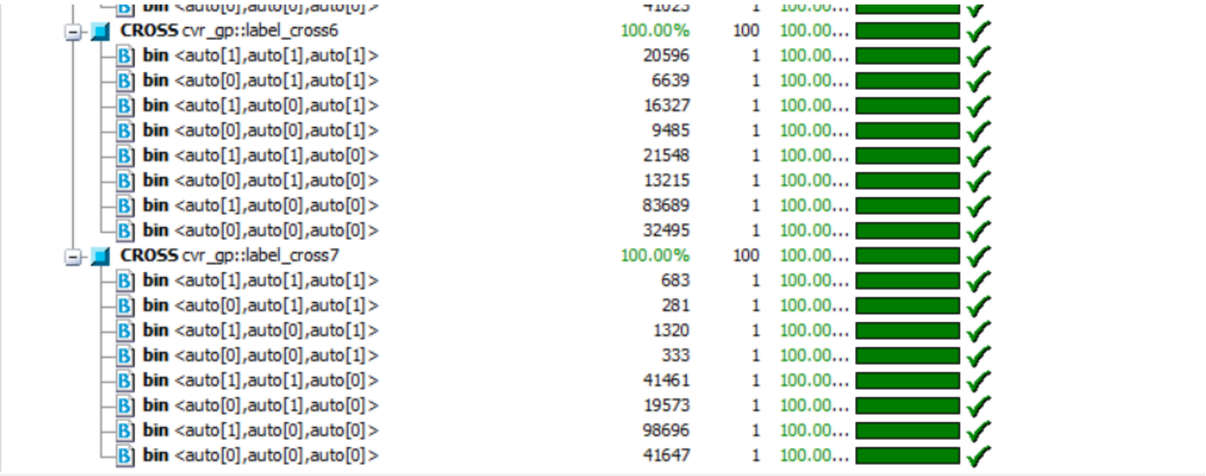


Covergroups

Name	Class Type	Coverage	Goal	% of Goal	Status	Included	Merge_instances	Ge
CROSS cvr_gp::label_cross1		100.00%	100	100.00...	🟩	✓		
bin <auto[1],auto[1],auto[1]>		16723	1	100.00...	🟩	✓		
bin <auto[0],auto[1],auto[1]>		2900	1	100.00...	🟩	✓		
bin <auto[1],auto[0],auto[1]>		39149	1	100.00...	🟩	✓		
bin <auto[0],auto[0],auto[1]>		6899	1	100.00...	🟩	✓		
bin <auto[1],auto[1],auto[0]>		25421	1	100.00...	🟩	✓		
bin <auto[0],auto[1],auto[0]>		16954	1	100.00...	🟩	✓		
bin <auto[1],auto[0],auto[0]>		60867	1	100.00...	🟩	✓		
bin <auto[0],auto[0],auto[0]>		35081	1	100.00...	🟩	✓		
CROSS cvr_gp::label_cross2		100.00%	100	100.00...	🟩	✓		
bin <auto[1],auto[1],auto[1]>		24544	1	100.00...	🟩	✓		
bin <auto[0],auto[1],auto[1]>		4521	1	100.00...	🟩	✓		
bin <auto[1],auto[0],auto[1]>		58966	1	100.00...	🟩	✓		
bin <auto[0],auto[0],auto[1]>		10605	1	100.00...	🟩	✓		
bin <auto[1],auto[1],auto[0]>		17600	1	100.00...	🟩	✓		
bin <auto[0],auto[1],auto[0]>		15333	1	100.00...	🟩	✓		
bin <auto[1],auto[0],auto[0]>		41050	1	100.00...	🟩	✓		
bin <auto[0],auto[0],auto[0]>		31375	1	100.00...	🟩	✓		
CROSS cvr_gp::label_cross3		100.00%	100	100.00...	🟩	✓		
bin <auto[1],auto[1],auto[1]>		604	1	100.00...	🟩	✓		
bin <auto[0],auto[1],auto[1]>		2226	1	100.00...	🟩	✓		
bin <auto[1],auto[0],auto[1]>		340	1	100.00...	🟩	✓		
bin <auto[0],auto[0],auto[1]>		141	1	100.00...	🟩	✓		
bin <auto[1],auto[1],auto[0]>		41540	1	100.00...	🟩	✓		
bin <auto[0],auto[1],auto[0]>		17628	1	100.00...	🟩	✓		
bin <auto[1],auto[0],auto[0]>		99676	1	100.00...	🟩	✓		
bin <auto[0],auto[0],auto[0]>		41839	1	100.00...	🟩	✓		

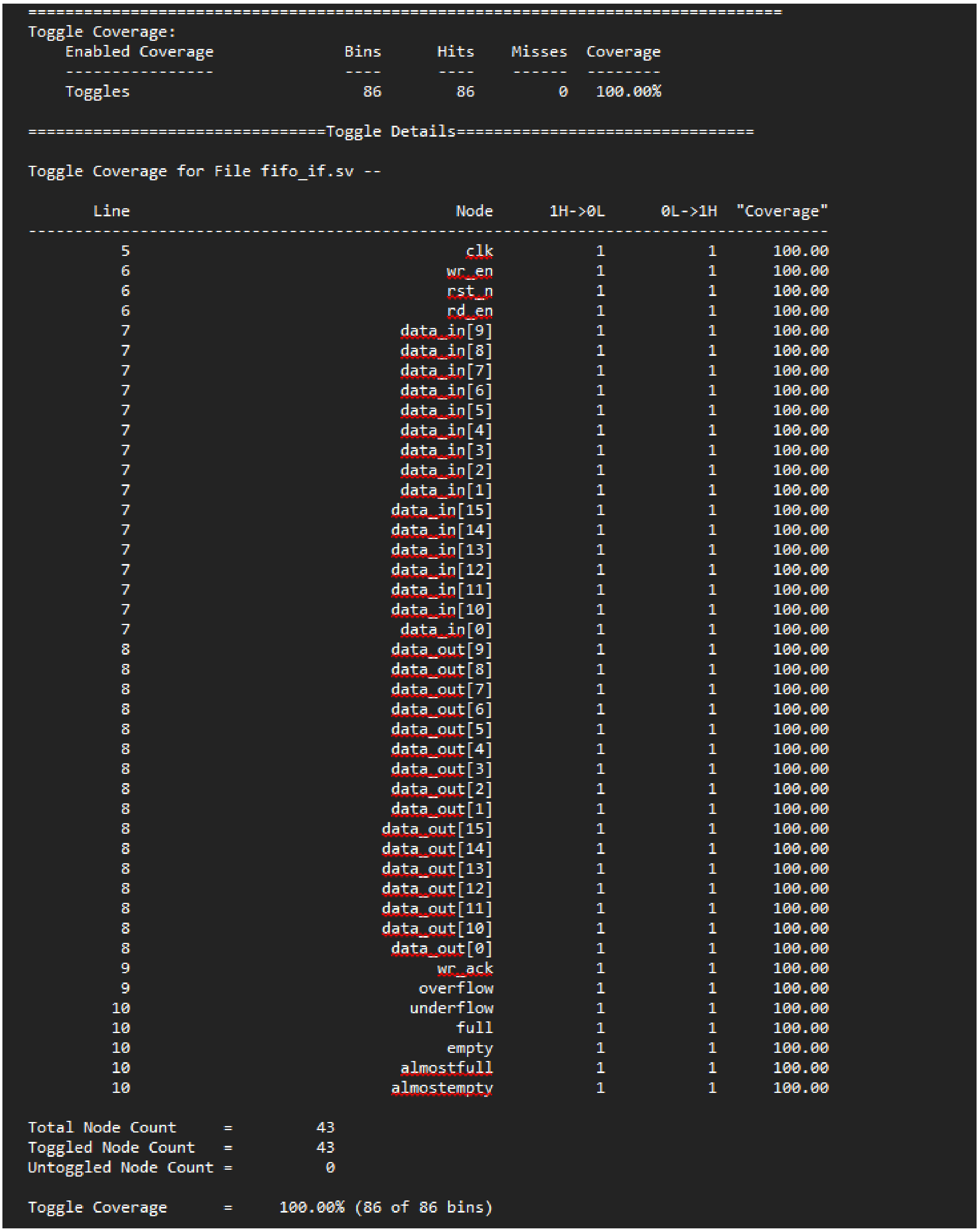


```
CROSS cvr_gp::label_cross4                100.00%   100  100.00...  ✓
   bin <auto[1],auto[1],auto[1]>            11147     1  100.00...  ✓
   bin <auto[0],auto[1],auto[1]>             4674     1  100.00...  ✓
   bin <auto[1],auto[0],auto[1]>            64301     1  100.00...  ✓
   bin <auto[0],auto[0],auto[1]>            21971     1  100.00...  ✓
   bin <auto[1],auto[1],auto[0]>            30997     1  100.00...  ✓
   bin <auto[0],auto[1],auto[0]>            15180     1  100.00...  ✓
   bin <auto[1],auto[0],auto[0]>            35715     1  100.00...  ✓
   bin <auto[0],auto[0],auto[0]>            20009     1  100.00...  ✓
CROSS cvr_gp::label_cross5                100.00%   100  100.00...  ✓
   bin <auto[1],auto[1],auto[1]>              586     1  100.00...  ✓
   bin <auto[0],auto[1],auto[1]>             2481     1  100.00...  ✓
   bin <auto[1],auto[0],auto[1]>             1427     1  100.00...  ✓
   bin <auto[0],auto[0],auto[1]>              957     1  100.00...  ✓
   bin <auto[1],auto[1],auto[0]>            41558     1  100.00...  ✓
   bin <auto[0],auto[1],auto[0]>            17373     1  100.00...  ✓
   bin <auto[1],auto[0],auto[0]>            98589     1  100.00...  ✓
   bin <auto[0],auto[0],auto[0]>            41023     1  100.00...  ✓
```

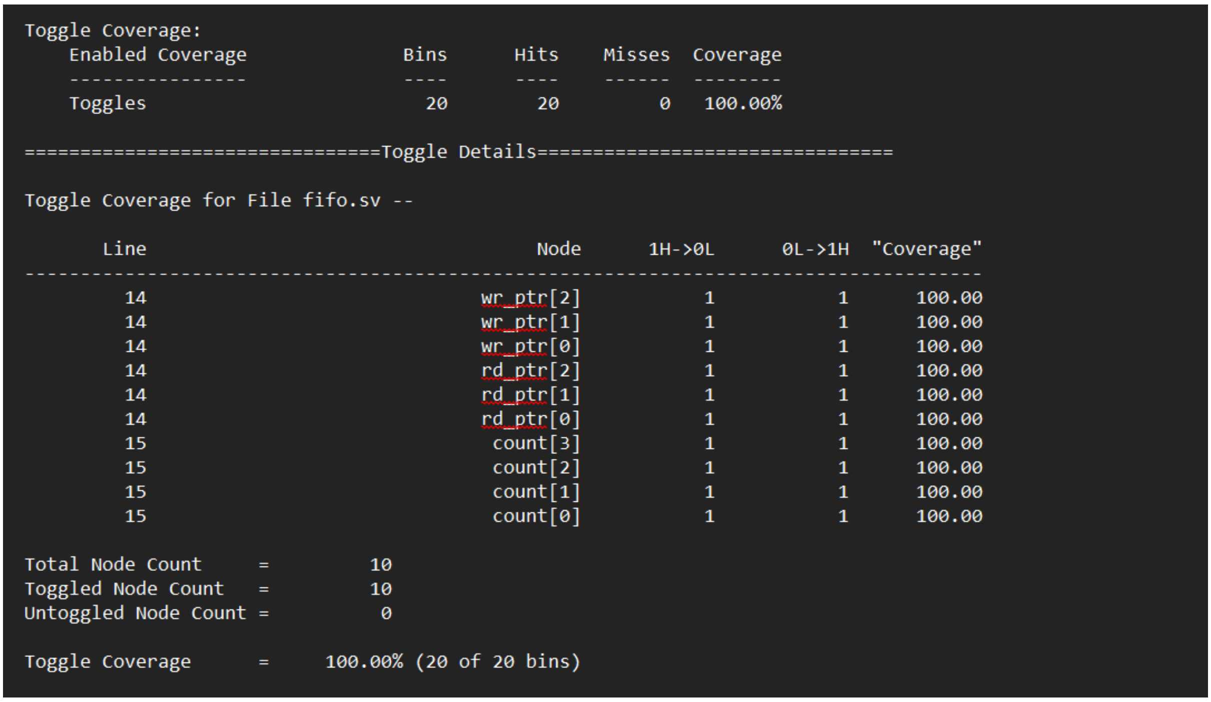


```
CROSS cvr_gp::label_cross6                100.00%   100  100.00...  ✓
   bin <auto[1],auto[1],auto[1]>            20596     1  100.00...  ✓
   bin <auto[0],auto[1],auto[1]>             6639     1  100.00...  ✓
   bin <auto[1],auto[0],auto[1]>            16327     1  100.00...  ✓
   bin <auto[0],auto[0],auto[1]>             9485     1  100.00...  ✓
   bin <auto[1],auto[1],auto[0]>            21548     1  100.00...  ✓
   bin <auto[0],auto[1],auto[0]>            13215     1  100.00...  ✓
   bin <auto[1],auto[0],auto[0]>            83689     1  100.00...  ✓
   bin <auto[0],auto[0],auto[0]>            32495     1  100.00...  ✓
CROSS cvr_gp::label_cross7                100.00%   100  100.00...  ✓
   bin <auto[1],auto[1],auto[1]>              683     1  100.00...  ✓
   bin <auto[0],auto[1],auto[1]>              281     1  100.00...  ✓
   bin <auto[1],auto[0],auto[1]>             1320     1  100.00...  ✓
   bin <auto[0],auto[0],auto[1]>              333     1  100.00...  ✓
   bin <auto[1],auto[1],auto[0]>            41461     1  100.00...  ✓
   bin <auto[0],auto[1],auto[0]>            19573     1  100.00...  ✓
   bin <auto[1],auto[0],auto[0]>            98696     1  100.00...  ✓
   bin <auto[0],auto[0],auto[0]>            41647     1  100.00...  ✓
```

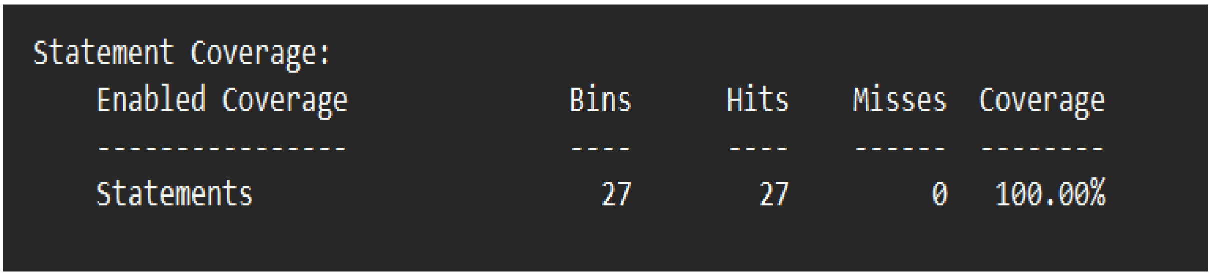
Codecoverage :



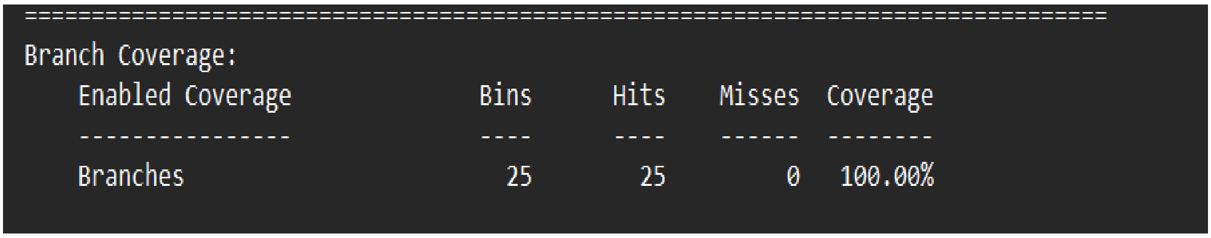
```
=================================================================================
Toggle Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Toggles                         86        86         0   100.00%

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

Toggle Coverage for File fifo_if.sv --

       Line                                   Node      1H->0L      0L->1H  "Coverage"
--------------------------------------------------------------------------------------
          5                                    clk           1           1      100.00
          6                                  wr_en           1           1      100.00
          6                                  rst_n           1           1      100.00
          6                                  rd_en           1           1      100.00
          7                             data_in[9]           1           1      100.00
          7                             data_in[8]           1           1      100.00
          7                             data_in[7]           1           1      100.00
          7                             data_in[6]           1           1      100.00
          7                             data_in[5]           1           1      100.00
          7                             data_in[4]           1           1      100.00
          7                             data_in[3]           1           1      100.00
          7                             data_in[2]           1           1      100.00
          7                             data_in[1]           1           1      100.00
          7                            data_in[15]           1           1      100.00
          7                            data_in[14]           1           1      100.00
          7                            data_in[13]           1           1      100.00
          7                            data_in[12]           1           1      100.00
          7                            data_in[11]           1           1      100.00
          7                            data_in[10]           1           1      100.00
          7                             data_in[0]           1           1      100.00
          8                            data_out[9]           1           1      100.00
          8                            data_out[8]           1           1      100.00
          8                            data_out[7]           1           1      100.00
          8                            data_out[6]           1           1      100.00
          8                            data_out[5]           1           1      100.00
          8                            data_out[4]           1           1      100.00
          8                            data_out[3]           1           1      100.00
          8                            data_out[2]           1           1      100.00
          8                            data_out[1]           1           1      100.00
          8                           data_out[15]           1           1      100.00
          8                           data_out[14]           1           1      100.00
          8                           data_out[13]           1           1      100.00
          8                           data_out[12]           1           1      100.00
          8                           data_out[11]           1           1      100.00
          8                           data_out[10]           1           1      100.00
          8                            data_out[0]           1           1      100.00
          9                                 wr_ack           1           1      100.00
          9                               overflow           1           1      100.00
         10                              underflow           1           1      100.00
         10                                   full           1           1      100.00
         10                                  empty           1           1      100.00
         10                             almostfull           1           1      100.00
         10                            almostempty           1           1      100.00

Total Node Count     =         43
Toggled Node Count   =         43
Untoggled Node Count =          0

Toggle Coverage      =     100.00% (86 of 86 bins)
```



```
Toggle Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Toggles                         20        20         0   100.00%

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

Toggle Coverage for File fifo.sv --

       Line                                   Node      1H->0L      0L->1H  "Coverage"
---------------------------------------------------------------------------------------
         14                              wr_ptr[2]           1           1      100.00
         14                              wr_ptr[1]           1           1      100.00
         14                              wr_ptr[0]           1           1      100.00
         14                              rd_ptr[2]           1           1      100.00
         14                              rd_ptr[1]           1           1      100.00
         14                              rd_ptr[0]           1           1      100.00
         15                               count[3]           1           1      100.00
         15                               count[2]           1           1      100.00
         15                               count[1]           1           1      100.00
         15                               count[0]           1           1      100.00

Total Node Count     =         10
Toggled Node Count   =         10
Untoggled Node Count =          0

Toggle Coverage      =     100.00% (20 of 20 bins)
```

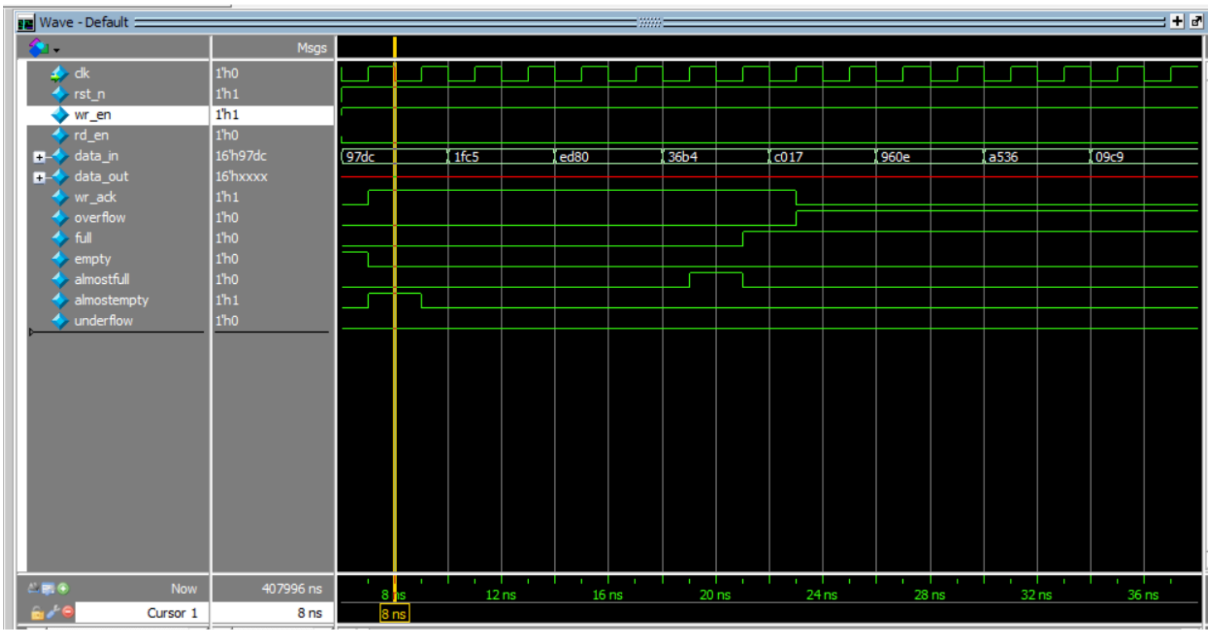


```
Statement Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Statements                      27        27         0   100.00%
```



```
=================================================================================
Branch Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Branches                        25        25         0   100.00%
```

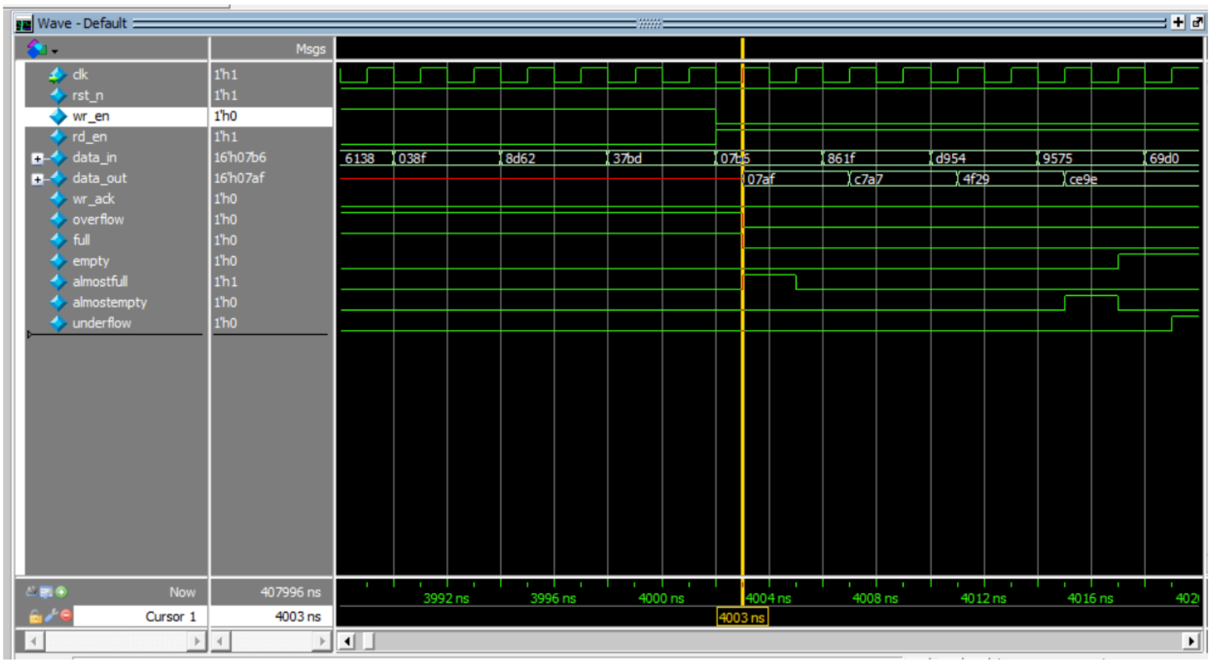
Sequence_1 write_only :



Wave - Default

```
                 Msgs
clk              1'h0
rst_n            1'h1
wr_en            1'h1
rd_en            1'h0
data_in          16'h97dc
data_out         16'hxxxx
wr_ack           1'h1
overflow         1'h0
full             1'h0
empty            1'h0
almostfull       1'h0
almostempty      1'h1
underflow        1'h0

Now              407996 ns
Cursor 1         8 ns
```

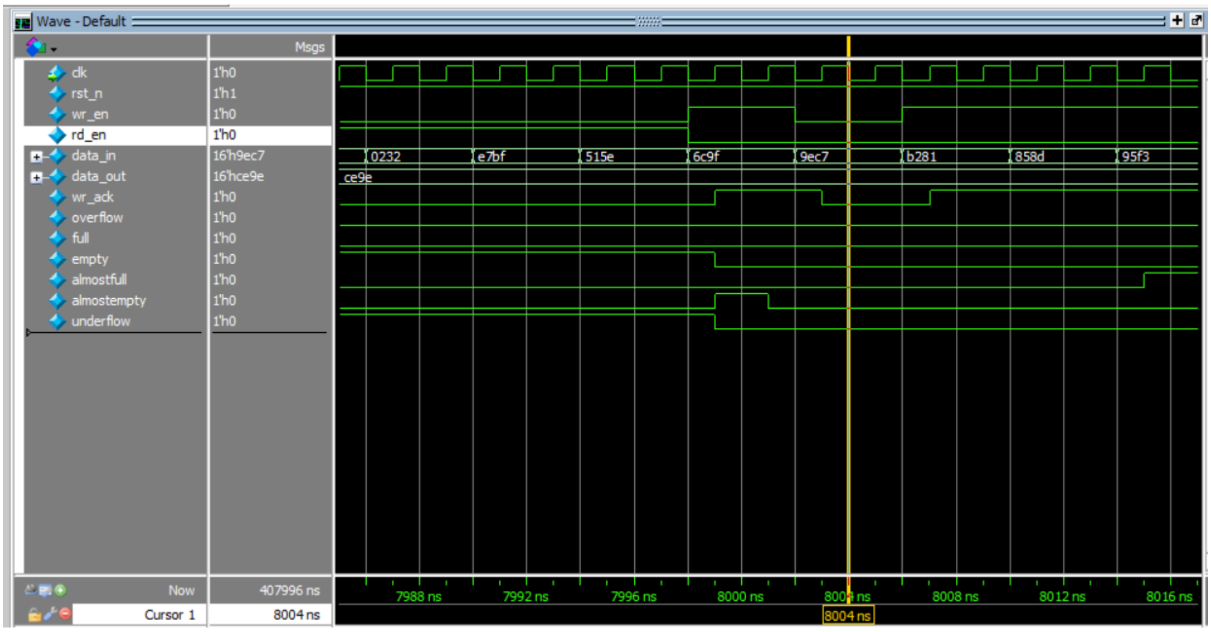
Sequence_2 read only :



Wave - Default

```
                 Msgs
clk              1'h1
rst_n            1'h1
wr_en            1'h0
rd_en            1'h1
data_in          16'h07b6
data_out         16'h07af
wr_ack           1'h0
overflow         1'h0
full             1'h0
empty            1'h0
almostfull       1'h1
almostempty      1'h0
underflow        1'h0

Now              407996 ns
Cursor 1         4003 ns
```

Sequence_3 randmize write and read :

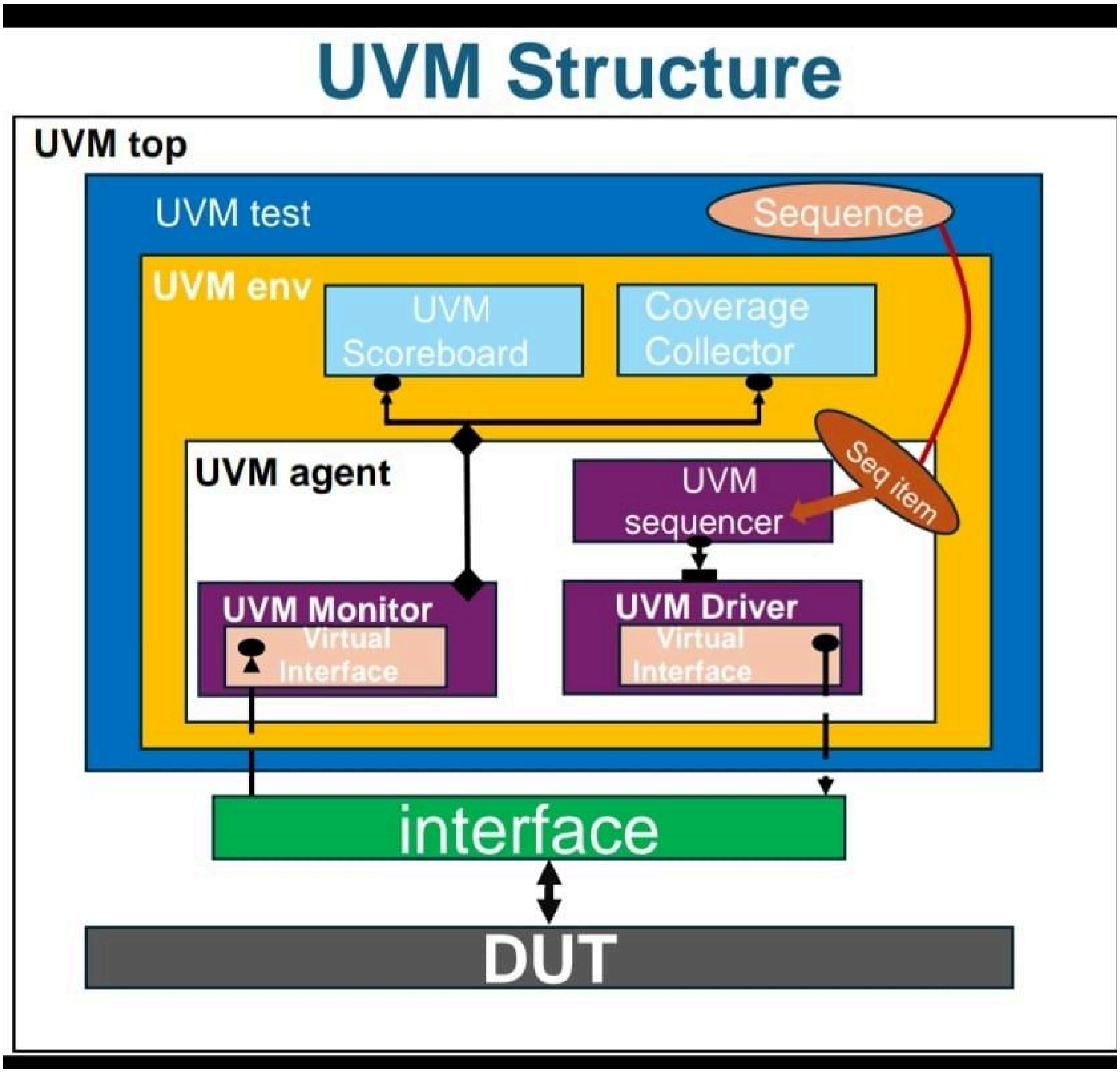


Wave - Default

	Msgs
clk	1'h0
rst_n	1'h1
wr_en	1'h0
rd_en	1'h0
data_in	16'h9ec7
data_out	16'hce9e
wr_ack	1'h0
overflow	1'h0
full	1'h0
empty	1'h0
almostfull	1'h0
almostempty	1'h0
underflow	1'h0

Now	407996 ns
Cursor 1	8004 ns

Structure :



1. Top Module (Testbench Top)

- Instantiates the **DUT (Design Under Test)**
- Instantiates the **Interface**
- Connects the interface to both the DUT and the testbench
- Runs the UVM testbench

2. Interface

The interface groups DUT signals together ,making them easier to pass into the UVM testbench via a **virtual interface**.

3. Driver

The **driver** is a component that converts **transaction-level objects** (e.g., packets, data structures) into **pin-level activity** — i.e., toggling interface signals to stimulate the DUT.

- It inherits from `uvm_driver`.
- It receives transactions from the **sequencer**.
- It uses the **virtual interface** to drive the DUT.

4. Monitor

The **monitor** passively observes the DUT signals via the interface and **converts signal-level activity back into transactions**.

- It does **not drive signals**
- It collects transactions and forwards them to a **scoreboard or coverage collector**

5. Sequencer and Sequence

- **The sequencer sends sequences of transactions to the driver.**
- **The sequence is like a script that creates and sends test data**

6. Scoreboard (Analysis and Checking)

- **Receives transactions from the monitor**
- **Compares actual output with expected output**
- **Reports PASS/FAIL or logs discrepancies**

7. Environment and Test

- **The Environment instantiates and connects all components (driver, monitor, scoreboard).**
- **The Test starts sequences and drives the simulation.**

8. Summary Flow

[Top Module (tb_top)]

↓

[Interface] ↔ [DUT]

↓

[UVM Testbench]

├─ Driver ← Sequences from Sequencer

├─ Monitor → Sends data to Scoreboard

└─ Scoreboard → Compares expected vs actual