Name: Sushan Patel
UIN: 334003274

Lab 08 Report

1. Rx_Monitor Code:

```
//////////////////////////////////////////////////////////////////////////
// Texas A&M University
// CSCE 616 Hardware Design Verification
// Created by  : Prof. Quinn and Saumil Gogri
//////////////////////////////////////////////////////////////////////////

class htax_rx_monitor_c extends uvm_monitor;

	`uvm_component_utils(htax_rx_monitor_c)

	//Analysis port to communicate with Scoreboard
	uvm_analysis_port #(htax_rx_mon_packet_c) rx_collect_port;

	virtual interface htax_rx_interface htax_rx_intf;
	htax_rx_mon_packet_c rx_mon_packet;
	int pkt_len;

	function new (string name, uvm_component parent);
		super.new(name, parent);
		rx_collect_port = new ("rx_collect_port", this);
		rx_mon_packet		= new();
	endfunction : new

	function void build_phase (uvm_phase phase);
		super.build_phase(phase);
		if(!uvm_config_db#(virtual htax_rx_interface)::get(this,"","rx_vif",htax_rx_intf))
			`uvm_fatal("RX_VIF",{"Virtual Interface needs to be set for
",get_full_name(),".rx_vif"})
	endfunction : build_phase

	//TO DO : Complete the run_phase method
	task run_phase(uvm_phase phase);
		forever begin
			pkt_len=0;
			//TO DO : Wait for rising edge of htax_rx_intf.rx_sot
			@(posedge htax_rx_intf.clk)
			wait(htax_rx_intf.rx_sot);

			//TO DO : On consecutive clock cycles append htax_rx_intf.rx_data to
rx_mon_packet.data[] till htax_rx_intf.rx_eot pulse
```

```
                        while(~htax_rx_intf.rx_eot)  begin  //TO  DO  :  Replace  XXX  with  appropriate
condition in while loop
                                @(posedge htax_rx_intf.clk)
                                rx_mon_packet.data = new[++pkt_len] (rx_mon_packet.data);
                                rx_mon_packet.data[pkt_len-1]=htax_rx_intf.rx_data;
                            end
                        //TO DO : Write the rx_mon_packet on anaylysis port
                        rx_collect_port.write(rx_mon_packet);


                end
        endtask : run_phase
endclass : htax_rx_monitor_c
```

2. Scoreboard Code:

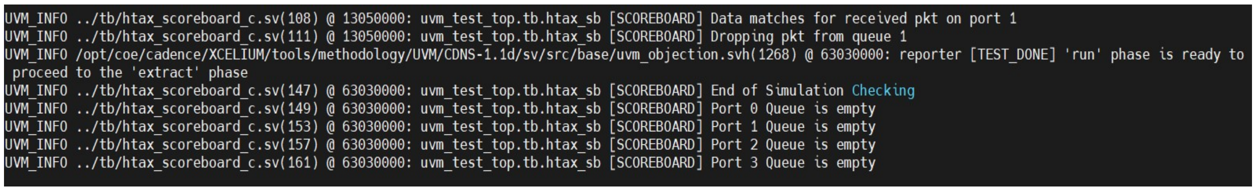
```
///////////////////////////////////////////////////////////////////////
// Texas A&M University
// CSCE 616 Hardware Design Verification
// Created by  : Prof. Quinn and Saumil Gogri
///////////////////////////////////////////////////////////////////////

        //Declasing analysis TX port
        `uvm_analysis_imp_decl(_tx0_export)
        `uvm_analysis_imp_decl(_tx1_export)
        `uvm_analysis_imp_decl(_tx2_export)
        `uvm_analysis_imp_decl(_tx3_export)

        //Declasing analysis RX port
        `uvm_analysis_imp_decl(_rx0_export)
        `uvm_analysis_imp_decl(_rx1_export)
        `uvm_analysis_imp_decl(_rx2_export)
        `uvm_analysis_imp_decl(_rx3_export)

class htax_scoreboard_c extends uvm_scoreboard;

        `uvm_component_utils(htax_scoreboard_c)

        //Registering with Factory
        uvm_analysis_imp_tx0_export #(htax_tx_mon_packet_c, htax_scoreboard_c) tx0_export;
        uvm_analysis_imp_tx1_export #(htax_tx_mon_packet_c, htax_scoreboard_c) tx1_export;
        uvm_analysis_imp_tx2_export #(htax_tx_mon_packet_c, htax_scoreboard_c) tx2_export;
        uvm_analysis_imp_tx3_export #(htax_tx_mon_packet_c, htax_scoreboard_c) tx3_export;

        uvm_analysis_imp_rx0_export #(htax_rx_mon_packet_c, htax_scoreboard_c) rx0_export;
        uvm_analysis_imp_rx1_export #(htax_rx_mon_packet_c, htax_scoreboard_c) rx1_export;
        uvm_analysis_imp_rx2_export #(htax_rx_mon_packet_c, htax_scoreboard_c) rx2_export;
```

```
	uvm_analysis_imp_rx3_export #(htax_rx_mon_packet_c, htax_scoreboard_c) rx3_export;

	//Creating queue to store the incoming TX transactions
	htax_tx_mon_packet_c port0_queue[$], port1_queue[$], port2_queue[$], port3_queue[$],
cmp_pkt[4];

	function new (string name, uvm_component parent);
		super.new(name,parent);
		tx0_export=new("tx0_export",this);
		tx1_export=new("tx1_export",this);
		tx2_export=new("tx2_export",this);
		tx3_export=new("tx3_export",this);

		rx0_export=new("rx0_export",this);
		rx1_export=new("rx1_export",this);
		rx2_export=new("rx2_export",this);
		rx3_export=new("rx3_export",this);
	endfunction : new

	//Write Method - TX[0] Monitor
	function void write_tx0_export(htax_tx_mon_packet_c tx_mon_packet);
		tx_mon_packet.print();
		push_to_queue(tx_mon_packet);
	endfunction : write_tx0_export

	//Write Method - TX[1] Monitor
	function void write_tx1_export(htax_tx_mon_packet_c tx_mon_packet);
		tx_mon_packet.print();
		push_to_queue(tx_mon_packet);
	endfunction : write_tx1_export

	//Write Method - TX[2] Monitor
	function void write_tx2_export(htax_tx_mon_packet_c tx_mon_packet);
		tx_mon_packet.print();
		push_to_queue(tx_mon_packet);
	endfunction : write_tx2_export

	//Write Method - TX[3] Monitor
	function void write_tx3_export(htax_tx_mon_packet_c tx_mon_packet);
		tx_mon_packet.print();
		push_to_queue(tx_mon_packet);
	endfunction : write_tx3_export

	//TO DO : Complete the code for push_to_queue function -- preferably use the SV-Case
statement
	//Add incoming TX Monitor packet to corresponding queue from mon_pkt.dest_port
	function void push_to_queue(htax_tx_mon_packet_c mon_pkt);
		case(mon_pkt.dest_port)
```

```
                    'b00 : port0_queue.push_front(mon_pkt);
                    'b01 : port1_queue.push_front(mon_pkt);
                    'b10 : port2_queue.push_front(mon_pkt);
                    'b11 : port3_queue.push_front(mon_pkt);
                endcase
        endfunction : push_to_queue

        //Write Method - RX[0] Monitor
        function void write_rx0_export(htax_rx_mon_packet_c rx_mon_packet);
                `uvm_info("SCOREBOARD",$sformatf("Received Data Packet on Port0:"), UVM_NONE)
                rx_mon_packet.print();
                cmp_pkt[0] = new ();
                cmp_pkt[0] = port0_queue.pop_back();
                if(cmp_pkt[0].data==rx_mon_packet.data)
                        `uvm_info("SCOREBOARD","Data matches for received pkt on port 0",
UVM_NONE)
                else
                        `uvm_fatal("SCOREBOARD","Data mismatch for received pkt on port 0")
                `uvm_info("SCOREBOARD","Dropping pkt from queue 0", UVM_NONE)
        endfunction : write_rx0_export

        //TO DO : Write Method - RX[1] Monitor
        function void write_rx1_export(htax_rx_mon_packet_c rx_mon_packet);
                //TO DO : Create a new cmp_pkt[i]
                //                  Pop the last element from queue i assign it to cmp_pkt[i]
                //                  Compare the data field of cmp_pkt[i] and
rx_mon_packet; Mismatch results into fatal error
                `uvm_info("SCOREBOARD",$sformatf("Received Data Packet on Port1:"), UVM_NONE)
                rx_mon_packet.print();
                cmp_pkt[1] = new ();
                cmp_pkt[1] = port1_queue.pop_back();
                if(cmp_pkt[1].data==rx_mon_packet.data)
                        `uvm_info("SCOREBOARD","Data matches for received pkt on port 1",
UVM_NONE)
                else
                        `uvm_fatal("SCOREBOARD","Data mismatch for received pkt on port 1")
                `uvm_info("SCOREBOARD","Dropping pkt from queue 1", UVM_NONE)
        endfunction : write_rx1_export

        //TO DO : Write Method - RX[2] Monitor
        function void write_rx2_export(htax_rx_mon_packet_c rx_mon_packet);
                //TO DO : Create a new cmp_pkt[i]
                //                  Pop the last element from queue i assign it to cmp_pkt[i]
                //                  Compare the data field of cmp_pkt[i] and
rx_mon_packet; Mismatch results into fatal error
                `uvm_info("SCOREBOARD",$sformatf("Received Data Packet on Port2:"), UVM_NONE)
                rx_mon_packet.print();
                cmp_pkt[2] = new ();
```

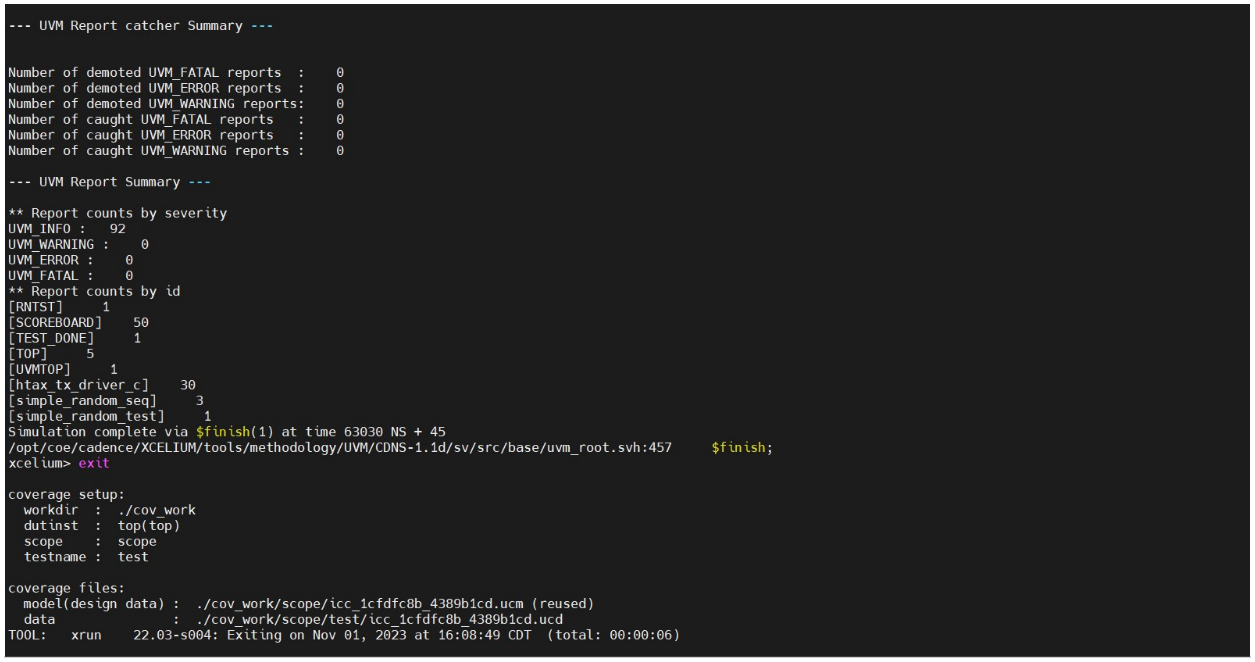
```
			cmp_pkt[2] = port2_queue.pop_back();
			if(cmp_pkt[2].data==rx_mon_packet.data)
				`uvm_info("SCOREBOARD","Data matches for received pkt on port 2", 
UVM_NONE)
			else
				`uvm_fatal("SCOREBOARD","Data mismatch for received pkt on port 2")
			`uvm_info("SCOREBOARD","Dropping pkt from queue 2", UVM_NONE)
		endfunction : write_rx2_export

		//TO DO : Write Method - RX[3] Monitor
		function void write_rx3_export(htax_rx_mon_packet_c rx_mon_packet);
			//TO DO : Create a new cmp_pkt[i]
			//				Pop the last element from queue i assign it to cmp_pkt[i]
			//				Compare the data field of cmp_pkt[i] and 
rx_mon_packet; Mismatch results into fatal error
			`uvm_info("SCOREBOARD",$sformatf("Received Data Packet on Port3:"), UVM_NONE)
			rx_mon_packet.print();
			cmp_pkt[3] = new ();
			cmp_pkt[3] = port3_queue.pop_back();
			if(cmp_pkt[3].data==rx_mon_packet.data)
				`uvm_info("SCOREBOARD","Data matches for received pkt on port 3", 
UVM_NONE)
			else
				`uvm_fatal("SCOREBOARD","Data mismatch for received pkt on port 3")
			`uvm_info("SCOREBOARD","Dropping pkt from queue 3", UVM_NONE)
		endfunction : write_rx3_export

		function void check();
			`uvm_info("SCOREBOARD", "End of Simulation Checking", UVM_NONE)
			if(port0_queue.size()==0)
    `uvm_info("SCOREBOARD","Port 0 Queue is empty", UVM_NONE)
   else
    `uvm_fatal("SCOREBOARD","Port 0 Queue is non-empty at end of simulation")
			if(port1_queue.size()==0)
    `uvm_info("SCOREBOARD","Port 1 Queue is empty", UVM_NONE)
   else
    `uvm_fatal("SCOREBOARD","Port 1 Queue is non-empty at end of simulation")
			if(port2_queue.size()==0)
    `uvm_info("SCOREBOARD","Port 2 Queue is empty", UVM_NONE)
   else
    `uvm_fatal("SCOREBOARD","Port 2 Queue is non-empty at end of simulation")
			if(port3_queue.size()==0)
    `uvm_info("SCOREBOARD","Port 3 Queue is empty", UVM_NONE)
   else
    `uvm_fatal("SCOREBOARD","Port 3 Queue is non-empty at end of simulation")
		endfunction : check

endclass : htax_scoreboard_c
```

3. UVM Summary:



```
UVM_INFO ../tb/htax_scoreboard_c.sv(108) @ 13050000: uvm_test_top.tb.htax_sb [SCOREBOARD] Data matches for received pkt on port 1
UVM_INFO ../tb/htax_scoreboard_c.sv(111) @ 13050000: uvm_test_top.tb.htax_sb [SCOREBOARD] Dropping pkt from queue 1
UVM_INFO /opt/coe/cadence/XCELIUM/tools/methodology/UVM/CDNS-1.1d/sv/src/base/uvm_objection.svh(1268) @ 63030000: reporter [TEST_DONE] 'run' phase is ready to
 proceed to the 'extract' phase
UVM_INFO ../tb/htax_scoreboard_c.sv(147) @ 63030000: uvm_test_top.tb.htax_sb [SCOREBOARD] End of Simulation Checking
UVM_INFO ../tb/htax_scoreboard_c.sv(149) @ 63030000: uvm_test_top.tb.htax_sb [SCOREBOARD] Port 0 Queue is empty
UVM_INFO ../tb/htax_scoreboard_c.sv(153) @ 63030000: uvm_test_top.tb.htax_sb [SCOREBOARD] Port 1 Queue is empty
UVM_INFO ../tb/htax_scoreboard_c.sv(157) @ 63030000: uvm_test_top.tb.htax_sb [SCOREBOARD] Port 2 Queue is empty
UVM_INFO ../tb/htax_scoreboard_c.sv(161) @ 63030000: uvm_test_top.tb.htax_sb [SCOREBOARD] Port 3 Queue is empty
```

4. Simulation Output:



```
--- UVM Report catcher Summary ---


Number of demoted UVM_FATAL reports  :    0
Number of demoted UVM_ERROR reports  :    0
Number of demoted UVM_WARNING reports:    0
Number of caught UVM_FATAL reports   :    0
Number of caught UVM_ERROR reports   :    0
Number of caught UVM_WARNING reports :    0

--- UVM Report Summary ---

** Report counts by severity
UVM_INFO :   92
UVM_WARNING :    0
UVM_ERROR :    0
UVM_FATAL :    0
** Report counts by id
[RNTST]     1
[SCOREBOARD]    50
[TEST_DONE]     1
[TOP]     5
[UVMTOP]     1
[htax_tx_driver_c]    30
[simple_random_seq]     3
[simple_random_test]     1
Simulation complete via $finish(1) at time 63030 NS + 45
/opt/coe/cadence/XCELIUM/tools/methodology/UVM/CDNS-1.1d/sv/src/base/uvm_root.svh:457     $finish;
xcelium> exit

coverage setup:
  workdir  :  ./cov_work
  dutinst  :  top(top)
  scope    :  scope
  testname :  test

coverage files:
  model(design data) :  ./cov_work/scope/icc_1cfdfc8b_4389b1cd.ucm (reused)
  data               :  ./cov_work/scope/test/icc_1cfdfc8b_4389b1cd.ucd
TOOL:   xrun    22.03-s004: Exiting on Nov 01, 2023 at 16:08:49 CDT  (total: 00:00:06)
```