

Lab 6

HTAX TX/RX interface and SV Assertions

Due date

Oct 21, 2024 11:59 PM CST

Table of content

Academic Integrity.....	2
Introduction.....	2
Design Under Test.....	2
Environment Setup.....	2
To-do.....	7
Deliverables.....	7

Academic Integrity


The following actions are strictly prohibited and violate the honor code. The minimum penalty for plagiarism is a grade of zero and a report to the Aggie honor system office.

- Sharing your solutions with a classmate.
- Uploading assignments to external websites or tutoring websites
- Copying solutions from external websites or tutoring websites
- Copying code from a classmate or unauthorized sources and submitting it as your

Introduction

In this lab, you will write SystemVerilog Assertions for HyperTransport Advanced X-Bar and perform coverage analysis on some specifications.

Design Under Test

The DUT is the HyperTransport Advanced X-Bar, whose specifications are mentioned in this document:  HyperTransport Advanced X-Bar HTAX Specification.pdf .

Environment Setup

1. Accept the assignment's repository on GitHub Classroom:

https://classroom.github.com/a/o_GBC3t

2. Source the setup file.

```
source setupX.bash
```

3. In this lab we have partial UVM-TB for HTAX design. Below is the list of files in lab6/ directory with a brief description

- *test/test_lib.svh* – UVM Test library
- *test/base_test.sv* – Base test template
- *test/simple_random_test.sv* – Kicks of simple random sequence on port [1] sequencer
- *tb/htax_defines.sv* – Design related defines
- *tb/htax_pkg.sv* - Include UVM components and object-related files
- *tb/htax_top.sv* - TB Top module
- *tb/htax_packet_c.sv* - HTAX packet class
- *tb/htax_seqs.sv* - Base sequence and simple random sequence code
- *tb/htax_env.sv* - UVM Environment (only UVM agent instantiated)
- *tb/htax_tx_agent_c.sv* - TX Agent (sequencer and driver instantiated)
- *tb/htax_tx_driver_c.svp* - Encrypted TX Driver code (We'll write the code in the next lab)
- *tb/htax_sequencer_c.sv* - TX Sequencer code

- *tb/htax_tx_interface.sv* - TX Interface and SV Assertions
- *tb/htax_rx_interface.sv* - RX Interface and SV Assertions

Monitor, Scoreboard, Virtual Sequence, Virtual Sequencer will be included in future labs.

4. Simple random test: This test runs a simple random sequence (15 random packets) on port [1]. This is done using the code below in "test/simple_random_test.sv"

```
//UVM build phase
function void build_phase(uvm_phase phase);
uvm_config_wrapper::set(this,"tb.tx_port[1].sequencer.run_phase","default_sequence",simple_random_seq::type_id::get());
super.build_phase(phase);
endfunction : build_phase
```

Run this test using this command:

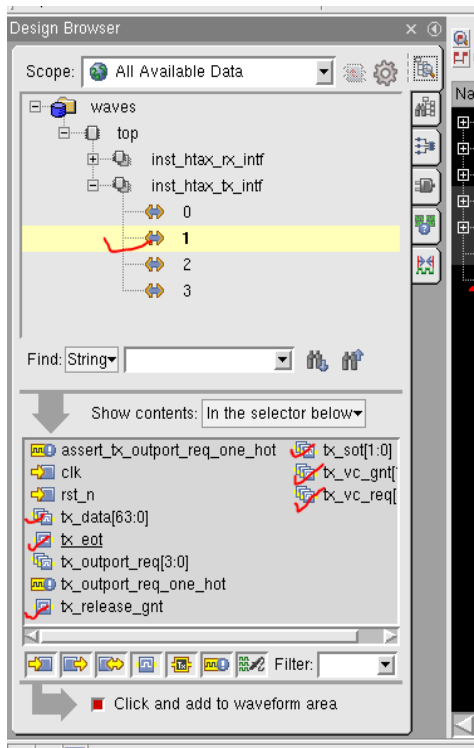
```
cd sim
xrun -f run.f +UVM_TESTNAME=simple_random_test
```

The test is simulated successfully if there are no UVM_FATAL or UVM_ERROR in the summary at the end of the simulation.

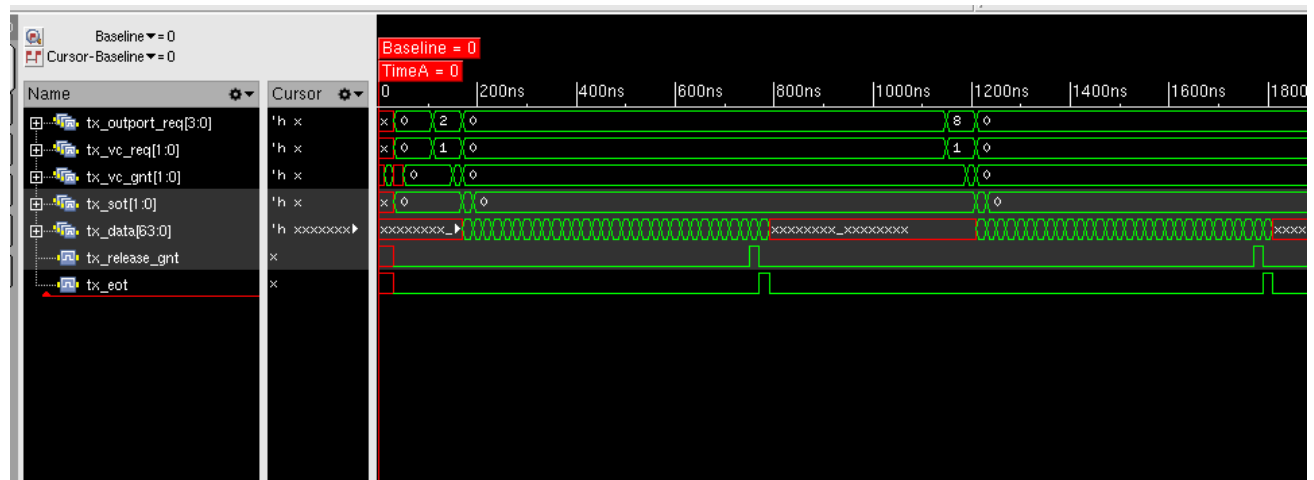
5. Open the waveform: open the waveform database using this command

```
simvision waves.shm/ &
```

Send all checked signals of htax_tx_intf (as shown in below figure) to the waveform window



You can see the waveform of all 15 TXN in the waveform window.



6. Open *tb/htax_tx_interface.sv* and *tb/htax_rx_interface.sv*.

The TX/RX interface signals are already provided to you in the above files.

One sample assertion is provided in *tb/htax_tx_interface.sv*.

Example: tx_outport_req is one-hot

```
property tx_outport_req_one_hot;
```

```

    @(posedge clk) disable iff(!rst_n)
    (|tx_outport_req) |-> $onehot(tx_outport_req);
endproperty

assert_tx_outport_req_one_hot : assert property(tx_outport_req_one_hot)
else
$error("HTAX_TX_INF ERROR : tx_outport request is not one hot encoded");

```

7. After coding assertions, simulate the simple random test. There are extra switches added in run.f which records coverage of each assertion.

```

-coverage A           // record "all" coverage
-covoverwrite         // overwrite existing coverage db
-covfile ./cov_conf.ccf // feed in coverage configuration file

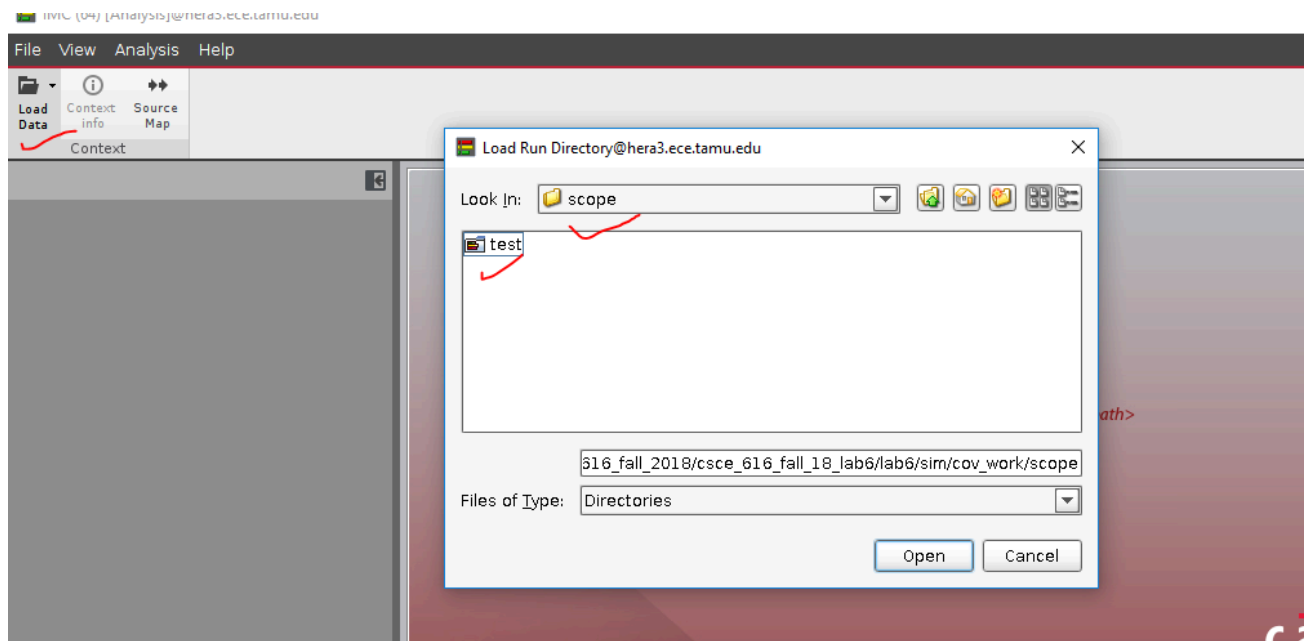
```

This will create a directory cov_work. Invoke IMC using below command and open the coverage database.

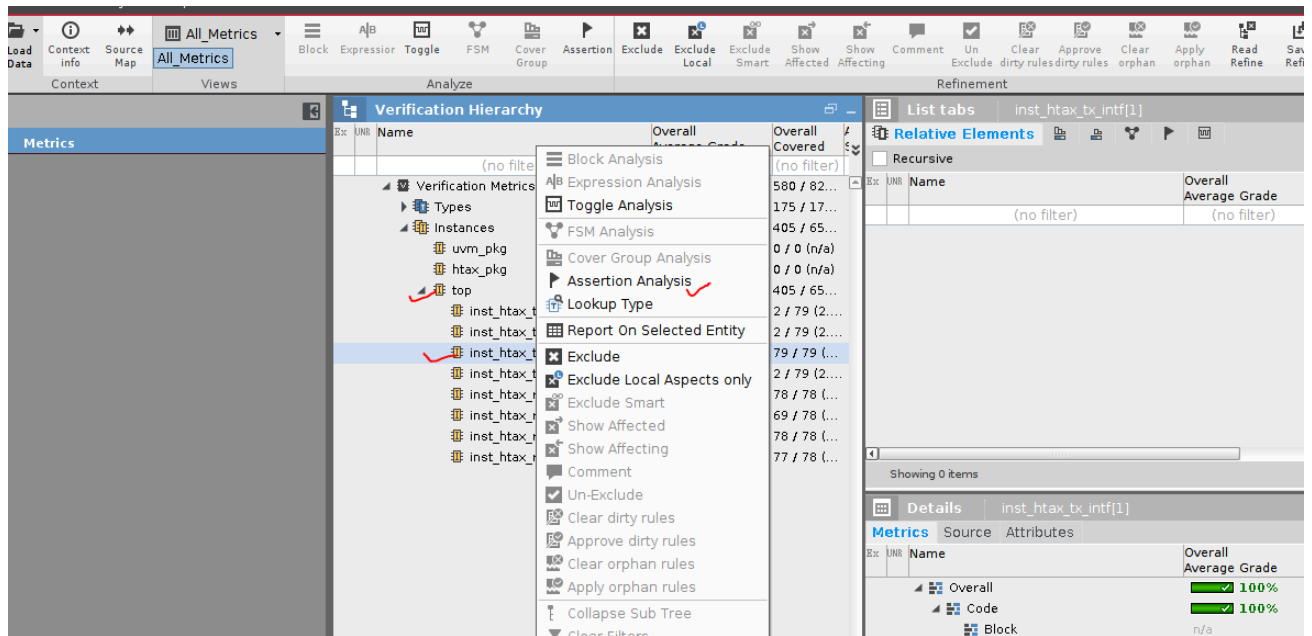
```
>> imc &
```

To open the coverage, follow the given steps:

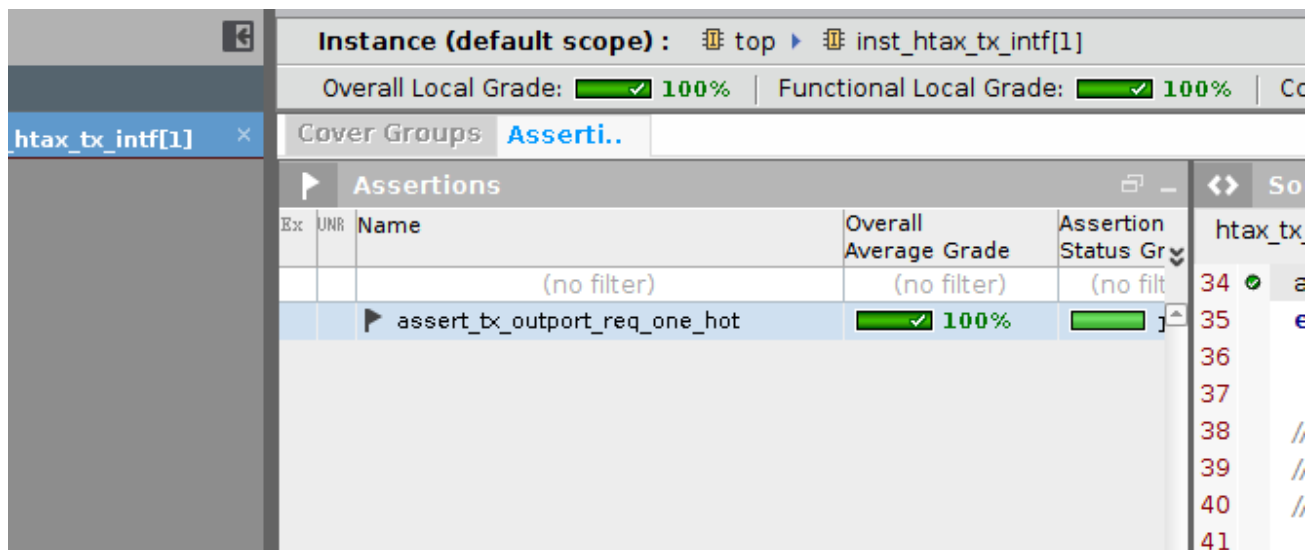
Load Data -> cov_work -> scope -> test -> open (refer to below figure)



Once the database is loaded, right click on top -> inst_htax_tx_intf[1] and select Assertion Analysis.



It will list all the assertions for the TX interface and show 100% (green) coverage for whichever assertion is exercised (as below)



8. Make sure that no assertions are failing at the end of simulation and after running the simulation, open IMC and check if all the assertions are covered or not. We will not tell you the reason why your assertions are not hitting. This is your part of the assignment to figure out.

To-do

1. Implement at least the following assertions in the TX interface (htax_tx_interface.sv)
 - 1.1. No tx_outport_req without tx_vc_req
 - 1.2. No tx_vc_req without tx_outport_req
 - 1.3. tx_vc_gnt is subset of vc_request
 - 1.4. No tx_sot without previous tx_vc_gnt
 - 1.5. No tx_eot without previous tx_vc_gnt
 - 1.6. tx_eot is asserted for a single clock cycle
 - 1.7. tx_release_gnt for pkt(t) one clock cycle or same clock cycle with tx_eot of pkt(t)
 - 1.8. No tx_sot of p(t+1) without tx_eot for p(t)
 - 1.9. Valid packet transfer – rise of tx_outport_req followed by a tx_vc_gnt followed by tx_sot followed by tx_release_gnt followed by tx_eot. Consider the right timings between each event.
2. Implement at least ONE assertion in the RX interface (from your v-plan) (htax_rx_interface.sv)
3. Take screenshots of your Assertion Analysis in IMC and demonstrate all your assertions show 100% (green) coverage. Paste your screenshots in a document and name it *coverage.pdf* inside your sim directory.

Deliverables

Commit and push all your changes to your remote repository.

Your repository must include the following:

- The test directory
- The sim directory containing coverage.pdf and updated files
- The tb directory containing updated files.
- The design directory

Important note: To get full credit, you must upload all the required files and directories and strictly name your files according to the requirements.