Lab 8

HTAX RX-Monitor/Scoreboard

Due date

Nov 11, 2024 11:59 PM

Table of content

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 learn to write UVM-Monitor and Scoreboard logic for the HyperTransport Advanced X-Bar design.

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/vo_xSs6l

2. Source the setup file.

```
source setupX.bash
cd work
```

3. We have now a full UVM-TB for HTAX design in place. Below is list of files in lab8/ directory with 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_tx_mon_packet_c.sv* - HTAX TX Monitor packet class

- *tb/htax_rx_mon_packet_c.sv* - HTAX RX Monitor packet class [NEW]
- *tb/htax_seqs.sv* - Base sequence and simple random sequence code
- *tb/htax_env.sv* - UVM Environment (Included all the TX/RX agents)
- *tb/htax_tx_agent_c.sv* - TX Agent (ACTIVE)
- *tb/htax_rx_agent_c.sv* - RX Agent (PASSIVE) [NEW]
- *tb/htax_tx_driver_c.sv* - TX Driver Code
- *tb/htax_tx_monitor_c.sv* - TX Monitor Code
- *tb/htax_rx_monitor_c.sv* - RX Monitor Code [NEW] (Complete the code in this lab)
- *tb/htax_scoreboard_c.sv* - Scoreboard Code [NEW] (Complete the code in this 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

4. **RX Monitor Code**: Open *tb/htax_rx_monitor_c.sv*. You need to complete the **TO DOs** in the task `run_phase`.

Note: RX monitor packet has only one field – data (dynamic array for 64-bits).

5. **HTAX Scoreboard Code**: Open *tb/htax_scoreboard_c.sv*.

There are 4 queues defined in scoreboard, one for each port. Any incoming TX monitor packet will be pushed in to corresponding to 'tx_mon_packet.dest_port' queue. For incoming RX monitor packet, we'll pop the packet from 'received port' queue and compare the data field. Ideally they should match.

6. **Simple random test**: Run this test using below command:

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

The test is simulated successfully if there are no UVM_FATAL, UVM_ERROR and no Assertion failures with signature "ncsim: *E,ASRTST" in the summary at the end of simulation.

7. **Check**: In tb/htax_scoreboard_c.sv we have defined a check() function. This function is executed at the end of simulation. Ideally all the queues should be empty at the end i.e. every packet entering the DUT from one of the TX ports is received out at the RX port based on the pkt.dest_port. Look for below statements at the end of log just before UVM summary.

```
UVM_INFO ../tb/htax_scoreboard_c.sv(150) @ 65130000:
uvm_test_top.tb.htax_sb [SCOREBOARD] End of Simulation Checking
UVM_INFO ../tb/htax_scoreboard_c.sv(152) @ 65130000:
uvm_test_top.tb.htax_sb [SCOREBOARD] Port 0 Queue is empty
UVM_INFO ../tb/htax_scoreboard_c.sv(156) @ 65130000:
uvm_test_top.tb.htax_sb [SCOREBOARD] Port 1 Queue is empty
UVM_INFO ../tb/htax_scoreboard_c.sv(160) @ 65130000:
uvm_test_top.tb.htax_sb [SCOREBOARD] Port 2 Queue is empty
UVM_INFO ../tb/htax_scoreboard_c.sv(164) @ 65130000:
uvm_test_top.tb.htax_sb [SCOREBOARD] Port 3 Queue is empty
```

To-do

1. In the **RX Monitor**, complete the following tasks
   1.1. Wait for rising edge of htax_rx_intf.rx_sot
   1.2. On consecutive clock cycles append htax_rx_intf.rx_data to rx_mon_packet.data[] till htax_rx_intf.rx_eot pulse (similar to previous lab)
   1.3. Replace XXX with appropriate condition in while loop
   1.4. Write the rx_mon_packet on analysis port

2. In the **HTAX Scoreboard**, complete the following tasks

   2.1. Complete the code for push_to_queue function (preferably using the SV-Case statement . push_to_queue adds incoming TX Monitor packet to corresponding queue from mon_pkt.dest_port)

   2.2. Complete the write method for RX[1] Monitor, RX[2] Monitor, RX[3] Monitor. For each you need to do the following.

      2.2.1. Create a new cmp_pkt[i]

      2.2.2. Pop the last element from queue i assign it to cmp_pkt[i]

      2.2.3. Compare the data field of cmp_pkt[i] and rx_mon_packet; Mismatch results into fatal error

3. Make a lab report that includes your **RX Monitor** and **HTAX Scoreboard** code, UVM summary (shown in step 7) and simulation output that shows that there are no UVM_FATAL/UVM_ERROR/Assertion failures and name it lab_report.pdf.

Deliverables

Commit and push all your changes to your remote repository.

Your repository must include the following:

- The test directory
- The sim directory containing lab_report.pdf
- The tb directory containing updated files.

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