Lab 4: UVM Introduction

Sri Sai Harshith Yellanki
Shamit Savant

Part 1 – Completing the UVM Testbench and Achieving 100% Coverage

Objective

The goal of this part was to complete a partially provided Universal Verification Methodology (UVM) testbench for a packet filter DUT that operates using the AXI4-Stream protocol. We were required to fix disconnected components within the environment, update the randomized sequence generation, and modify testbench parameters as necessary to achieve **100% functional and code coverage**.

What We Worked On

We focused on four key files in the testbench. Here's what each one does and what changes we made to it:

`filter_env.svh` – This file sets up the environment for testing

We added the missing output agent (called `agent_out`) that listens to the DUT's output.

We connected this agent's driver and monitor to the output interface (`out_vif`).

We hooked up both the input and output agents to the scoreboard so the system could compare expected vs. actual outputs.

We also made sure the data from both agents was sent to the coverage collectors.

Finally, we turned on packet-level checking for the scoreboard.

`filter_sequence.svh` – This is where test packets are generated

We limited the message type to valid ones (`0x0`, `0xA`, `0x5`, `0x3`) so the DUT would forward them.

We set the length of packets to vary between 1 and 16 bytes to get more diverse inputs.

We made sure the data and payload arrays were sized correctly based on that length.

`filter_tb.sv` – The main testbench module

We kept the number of packets high (`NUM_PACKETS = 1000`) to explore a wide range of test cases.

We ensured that the ready signal (`TOGGLE_READY`) was enabled to introduce some timing randomness and simulate real-world behavior.

Fixing the Testbench Environment

When we first ran the test using `make gui`, the testbench didn't work properly. There were no test results, no coverage data, and no packet forwarding. This happened because the output agent and scoreboard weren't fully connected yet.
Here's what we did to fix it:
Created and wired up the output agent (`agent_out`)

Connected the output driver and monitor to the correct interface

Hooked up both input and output agents to the scoreboard

Sent data from the agents to the coverage modules

Enabled packet-level operation for accurate comparison

Getting to 100% Coverage

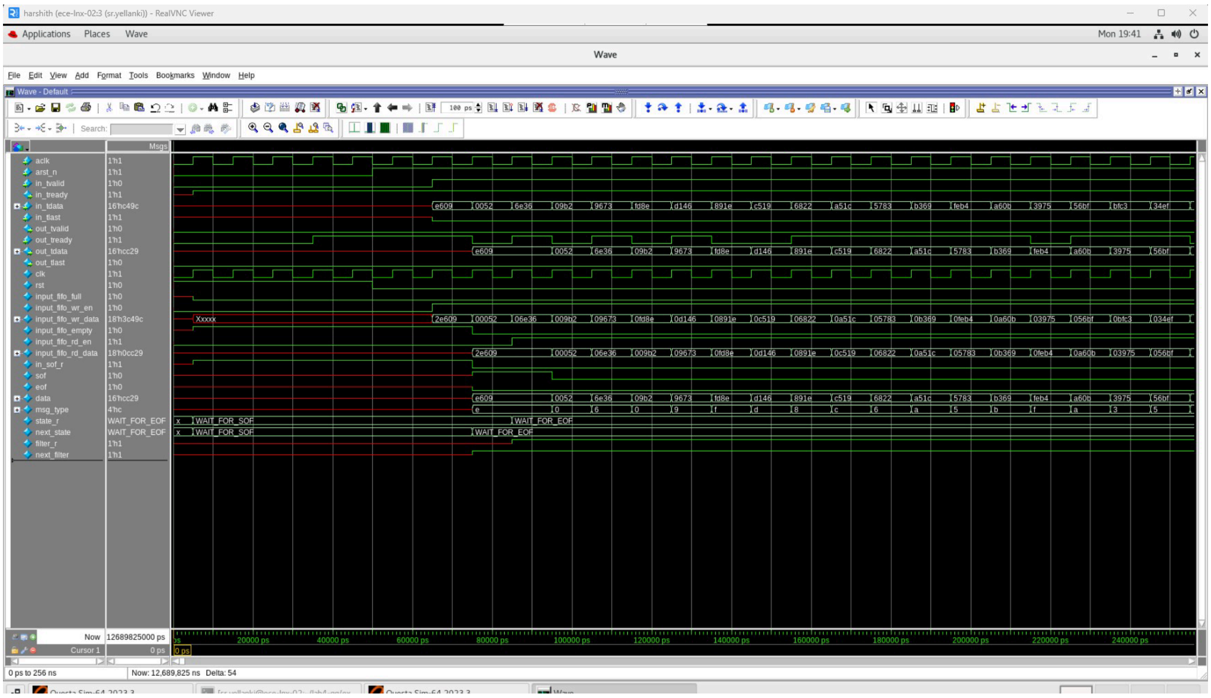
Once the test was running, our next goal was to increase coverage. Initially, coverage was below 100%, so we made a few changes to improve it:
We made sure the packets had **all message types from 0 to 15** that the DUT would filter.

We used different **packet lengths** (from 1 to 4095 bytes) to test how the DUT handled short and long messages.

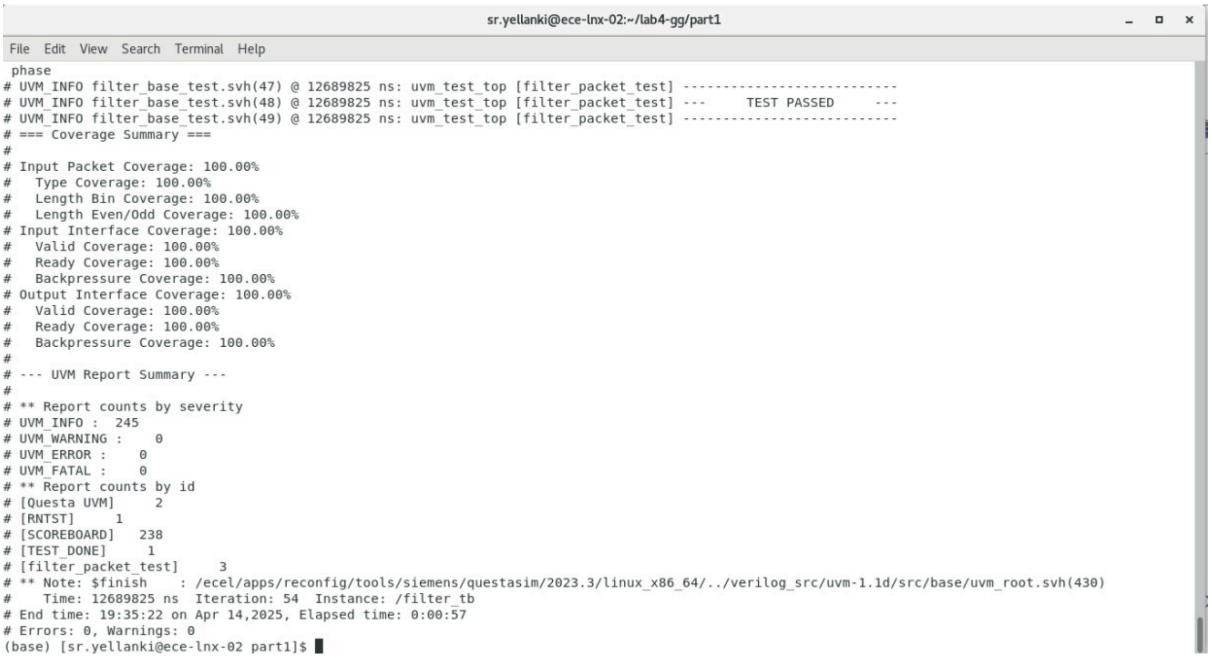
We verified that the structure of the packet data matched the expected format.

We kept the test size at 1000 packets to ensure we had enough variation.

Screenshots



	Msgs
aclk	1'h1
arst_n	1'h1
in_tvalid	1'h0
in_tready	1'h1
in_tdata	16'hc49c
in_tlast	1'h1
out_tvalid	1'h0
out_tready	1'h1
out_tdata	16'hcc29
out_tlast	1'h0
clk	1'h1
rst	1'h0
input_fifo_full	1'h0
input_fifo_wr_en	1'h0
input_fifo_wr_data	18'h3c49c
input_fifo_empty	1'h0
input_fifo_rd_en	1'h1
input_fifo_rd_data	18'h0cc29
in_sof_r	1'h1
sof	1'h0
eof	1'h0
data	16'hcc29
msg_type	4'hc
state_r	WAIT_FOR_EOF
next_state	WAIT_FOR_EOF
filter_r	1'h1
next_filter	1'h1
Now	12689825000 ps
Cursor 1	0 ps


sr.yellanki@ece-lnx-02:~/lab4-gg/part1
File Edit View Search Terminal Help

```
 phase
# UVM_INFO filter_base_test.svh(47) @ 12689825 ns: uvm_test_top [filter_packet_test] ---------------------------
# UVM_INFO filter_base_test.svh(48) @ 12689825 ns: uvm_test_top [filter_packet_test] ---     TEST PASSED     ---
# UVM_INFO filter_base_test.svh(49) @ 12689825 ns: uvm_test_top [filter_packet_test] ---------------------------
# === Coverage Summary ===
#
# Input Packet Coverage: 100.00%
#   Type Coverage: 100.00%
#   Length Bin Coverage: 100.00%
#   Length Even/Odd Coverage: 100.00%
# Input Interface Coverage: 100.00%
#   Valid Coverage: 100.00%
#   Ready Coverage: 100.00%
#   Backpressure Coverage: 100.00%
# Output Interface Coverage: 100.00%
#   Valid Coverage: 100.00%
#   Ready Coverage: 100.00%
#   Backpressure Coverage: 100.00%
#
# --- UVM Report Summary ---
#
# ** Report counts by severity
# UVM_INFO :  245
# UVM_WARNING :    0
# UVM_ERROR :    0
# UVM_FATAL :    0
# ** Report counts by id
# [Questa UVM]     2
# [RNTST]     1
# [SCOREBOARD]   238
# [TEST_DONE]     1
# [filter_packet_test]     3
# ** Note: $finish    : /ecel/apps/reconfig/tools/siemens/questasim/2023.3/linux_x86_64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh(430)
#    Time: 12689825 ns  Iteration: 54  Instance: /filter_tb
# End time: 19:35:22 on Apr 14,2025, Elapsed time: 0:00:57
# Errors: 0, Warnings: 0
(base) [sr.yellanki@ece-lnx-02 part1]$ 
```