Derek Chen
Lucy Hu
Calvin Lee

Final Project Report

EDA Playground link: https://edaplayground.com/x/LQdF

Objective

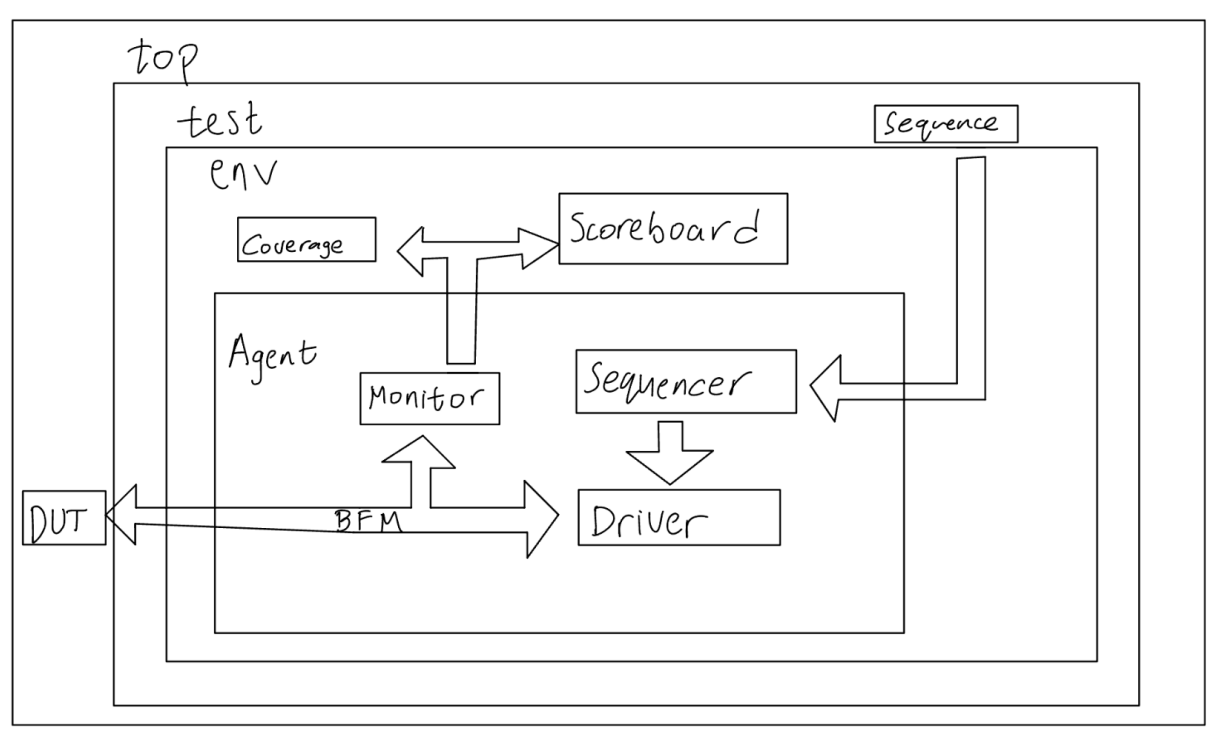
Our team developed a UVM testing environment to encapsulate our Hamming Decoder source code. The primary goal was to establish a highly comprehensive tester capable of examining our design under a wide range of scenarios. This approach aimed to maximize test coverage, ensuring thorough evaluation of all possible cases within the constraints of our testing framework.

Apparatus

In order to accomplish our objective, our team designed a UVM environment that comprehensively tested various cases for our target design. The test cases were based on 11-bit messages with 5 parity bits. Our testing strategy encompassed three error scenarios: no errors, one error, and two errors.For the "no errors" scenario, we expected the retrieval of the correct message, as errors were intentionally absent. In the case of "one error," our objective was to detect and correct the error, thereby obtaining the correct message. As for "two errors," our aim was to detect the errors without necessarily correcting them, prioritizing error detection over message retrieval.

We leveraged EDA Playground, incorporating UVM 1.2, along with popular simulators such as Mentor Questa and Aldec Riviera Pro, to execute our testing environment. Both simulators produced consistent and accurate results, which we will discuss in a subsequent section. Our testing structure drew inspiration from Ray Salemi's UVM Primer book, specifically the final section, which emphasized the utilization of coverage, scoreboard, monitors, drivers, agents, and sequencers.

Procedure



In the construction of our testbench, we utilized Ray Salemi's UVM Primer book as a valuable reference, incorporating a comprehensive set of UVM components. These components played integral roles in optimizing our utilization of the UVM environment. Here's a breakdown of the key components and their functions within our testbench:

- Sequencer: We employed a sequencer, prioritizing it over a tester, to efficiently deliver testing cases to our design-under-test (DUT). The sequencer acts as the central coordinator of test sequences, ensuring the orderly and controlled execution of stimuli.
- Sequence: The sequence components include an encoder in order to produce the parity bits and the actual 16 bit hamming encoded messages. We use these to ultimately send to our DUT, the hamming decoder. This information is also passed to ultimately if the result is correct in the scoreboard.
- Scoreboard: The scoreboard checks whether results are correct or incorrect based on the information given to it. The information that it receives are the recovered message and the original 16 bit encoded message. Based on this, the scoreboard will decide whether the decoded result is correct or not.
- Coverage Reporter: With the inclusion of a coverage reporter, we were able to maintain a thorough record of the various test cases handled throughout our testing process. The coverage component allowed us to monitor the extent of test coverage achieved and identify any potential areas that required additional testing.

- Result Monitor: Our testbench employed a result monitor to continuously track and validate the results obtained from the DUT. This monitoring mechanism served a dual purpose: verifying the correctness of the received results and providing inputs for coverage analysis.
- Agent: The agent component assumed a crucial role in establishing the connection and monitoring between our DUT and the testbench components. It facilitated seamless communication and synchronization between these entities, enabling effective data exchange.
- Driver: The driver played a pivotal role in driving values into the DUT. It received instructions from the sequencer and effectively conveyed the corresponding stimuli to the Bus Functional Model (BFM), ensuring the precise injection of desired inputs into the DUT.
- BFM and Environment (uvm_env): The BFM served as the bridge connecting our DUT to the testbench. It facilitated the seamless exchange of data and signals between these components. The testbench environment, specifically the uvm_env, orchestrated the configuration and operation of the BFM, ensuring the compatibility and synchronization of values between the DUT and the testbench.

Collectively, these components formed a robust and cohesive testbench infrastructure, optimizing the utilization of the UVM environment. Through the coordination of the sequencer, agent, driver, BFM, and monitors, we achieved efficient stimulus generation, accurate result verification, comprehensive coverage tracking, and seamless integration between the DUT and the testbench.

Results

Our test results demonstrate highly favorable outcomes, reflecting the effectiveness of our verification process. The figures below illustrate the substantial coverage achieved across all three error test cases, surpassing the threshold of 95%. Our coverage results are determined by the number of 11-bit messages and associated parity bits that were thoroughly examined during testing. These impressive coverage percentages attest to the comprehensive nature of our testing approach, ensuring a robust evaluation of the design's performance and error handling capabilities.

```
TWO ERROR COVERAGE: 95%

ONE ERROR COVERAGE: 98%

NO ERROR COVERAGE: 98%
```

For two error testing, we tested 4110 cases and received all of them to correctly detect errors.

```
incorrect =           0 correct =        4110
```

For one error testing, we tested 3210 cases and received all of them to correctly detect errors.

```
incorrect =           0 correct =        3210
```

For no error testing,we tested 3210 cases and received all of them to correctly detect errors.

```
incorrect =           0 correct =        3210
```

Conclusions

In conclusion, the implementation of the Hamming Decoder module within a UVM testing environment proved highly effective in achieving a comprehensive evaluation of our design across various scenarios. The coverage achieved during testing was deemed sufficient, and the overall results were positive, instilling confidence in the functionality of our module. While we encountered some initial challenges pertaining to BFM accessibility throughout the testbench, we successfully navigated through them, resulting in a relatively smooth testing process.

In hindsight, to further enhance the project, we acknowledge the potential benefits of expanding our test cases beyond isolated scenarios of two errors, one error, or no errors. By incorporating combinations of these scenarios, we could explore the system's behavior under more diverse conditions. Our current testing environment focuses on individually testing each case in bulk, but future improvements could involve introducing a mixed test scenario that encompasses all possible error configurations. Expanding our testing scope in this manner would allow us to gather more comprehensive insights into the performance and robustness of the Hamming Decoder module, providing a more thorough evaluation of its capabilities.