Fundamentals of Pre-Silicon Validation - Winter -2024

Implementation and Verification of Asynchronous FIFO

Team -1
Bhargav Chunduri - bhargavc@pdx.edu
Dhushyanth Dharmavarapu – dharmava@pdx.edu
Venkata Krishna Kumar vedantam – vedantam@pdx.edu

VERIFICATION TEST PLAN

Introduction

Our objective of the verification:

The primary goal of this verification plan is to comprehensively validate the design's functionality and accuracy using various verification techniques, while also implementing error detection and correction mechanisms.

Division of tasks:

Team members and tasks allocation:

• **Bhargav Chunduri** : He will concentrate on implementing and testing the core functionality of the asynchronous FIFO design. This involves fine-tuning the write operation with two idle cycles as per specifications, designing and testing the FIFO memory, and ensuring the proper operation of the generator and monitor modules. Furthermore, he will assist in drafting the verification plan document and provide support in other areas as required.

• **Dhushyanth Dharmavarapu:** He will be responsible for implementing and verifying the read functionality of the asynchronous FIFO design, ensuring one read idle cycle as specified. He will conduct thorough testing of the design using various test cases, write the Design Specification document, and develop the driver and scoreboard functionalities. Additionally, he will contribute to testing other modules and provide assistance as needed.

• **Venkata Krishna Kumar Vedantam**: He will lead the integration efforts by implementing the design through interfaces and ensuring proper connectivity with the top module. Additionally, he will calculate and document the FIFO depth, write the scoreboard, and thoroughly verify the percentage. He will also focus on creating the overall environment, tests, and the testbench top module. He will collaborate with other team members to ensure seamless integration and functionality.

VERIFICATION TEST PLAN:

Functional Verification:

- The functional verification process for an asynchronous FIFO involves thorough testing of key features like write/read operations, overflow/underflow conditions, and proper control signal handling.
- Special attention is given to ensuring correct synchronization and addressing potential metastability issues due to the asynchronous nature between clock domains. Test cases are designed to verify performance and robustness, including boundary conditions and stress testing.
- The FIFO's correct operation is validated, and verification environments are established with the help of simulation tools.

Edge Case Testing:

Test Case Scenarios:

Test case for Fifo Full:

This test case focuses on writing data to every available location in the FIFO until it reaches full capacity. The verification process includes monitoring the FIFO's full flag or status to ensure it correctly indicates when the FIFO is full. Additionally, the test observes the FIFO's behavior when attempting to write data while it is full, ensuring it either blocks further writes, generates an error, or follows another predefined response.

Test case for Full Empty:

In this test case, the FIFO starts off empty, and data is read from it until it is empty once more. Similar to the "Test Full" case, the empty flag or status of the FIFO must be monitored to verify it accurately reflects when the FIFO is empty. Additionally, the FIFO's behavior when attempting to read from it while empty should be examined, ensuring it either blocks reads, generates an error, or follows a specific predefined response.

Test case for Full Error:

This test case checks the FIFO's behavior when attempting to write data while it is already full. The expected outcome is that the FIFO should either raise an error, assert a full flag, or use another mechanism to signal that the write operation cannot be performed because the FIFO is full.

Test case for Empty Error:

Similar to the "Test Full Error" case, this test case verifies the behavior of the FIFO when attempting to read data from it while it is empty. The expected behavior is that the FIFO should raise an error, assert an empty flag, or use some other mechanism to indicate that a read operation cannot be performed due to the FIFO being empty.

Test case for Concurrent Write-Read:

This test case tests the simultaneous writing and reading of data to/from the FIFO. It verifies the proper concurrent functioning of the read and write ports, ensuring there are no race conditions or synchronization issues. The FIFO's behavior during these simultaneous operations is monitored to ensure it operates correctly.

Coverage Metrics:

- The three key coverage metrics for an Asynchronous FIFO are **functional coverage**, **code coverage**, and **assertion coverage**.
- **Functional coverage** tracks how effectively the test scenarios cover all necessary behaviors of the FIFO. **Code coverage** measures the percentage of code lines, branches, and conditions tested by the verification process, ensuring comprehensive design testing.
- **Assertion coverage** evaluates how well the assertions identify design properties and violations. These metrics are used to assess the thoroughness of the verification effort, pinpoint areas requiring more testing, and gauge confidence in the FIFO's accuracy and robustness.

Verification Environment:

We create a UVM (Universal Verification Methodology) based verification environment for an Asynchronous FIFO involves several steps to ensure thorough testing and validation of the design.

Git Hub Link for Project respositry:

- https://github.com/bhargavchun/Fifo_Pre_si.git

REFERENCES:

- **https://github.com/teekamkhandelwal/asynchronous_fifo/blob/main/r_pointer_epty.v**

- **http://www.sunburst-design.com/papers/CummingsSNUG2002SJ_FIFO1.pdf**