Class Based Testbench Explanation

Team 5 Asynchronous FIFO

May 2, 2025

ECE 593
Dr. Venkatesh Patil

Team 5 Members:
Kai Roy, Pavan Gaddam, SuryaTeja Purma, Gene Hu

Generator

The generator generates two transaction signals for write operations and read operations. The number of transactions generated for write and read operations are set individually in the testbench. We decided to implement the write and read operations separately to allow us to simulate asynchronous access to the FIFO module. The generator communicates with the driver via two mailboxes, one for each operation.

Driver

The driver is responsible for converting the data received from the generator module via the mailbox system, and it will drive the input data to the DUT every clock edge. This module is also responsible for resetting the DUT's input signals when reset is received from the testbench. As previously mentioned, since this is an asynchronous FIFO, our driver implementation is able to drive reads and writes separately.

Monitor

The monitor observes the interface connecting our environment to the DUT. Two concurrent monitor operations occur observing any changes to relevant write signals and read signals respectively. This information is only sent to the scoreboard if the write enable signal or the read enable signal is set high. This means that the data on the interface is valid and should be stored in the DUT and will need to be checked by the scoreboard.

Scoreboard

The scoreboard keeps track of all the write operations and stores it in its own FIFO buffer. Only the write operations are stored because any successful read operation can only occur when there is data stored in the DUT. This means that successful read operations can only occur after a successful write operation. This allows us to bypass the FIFO for any read transactions sent by the monitor and the results can be immediately compared with what is in the write transaction buffer.

Status of the Testbench

At the time of writing this report, our class-based testbench is not functional. All of the components compile together, but the testbench sees a fatal error during simulation that occurs in the scoreboard and has something to do with the mailbox. Currently the addition of an if statement that prevents the scoreboard from attempting to pull data from the mailbox when it is empty is being developed and tested and the results are promising, but the testbench is still not functioning as expected and seems to be stalling, indicating that the there is still an issue with the testing environment.