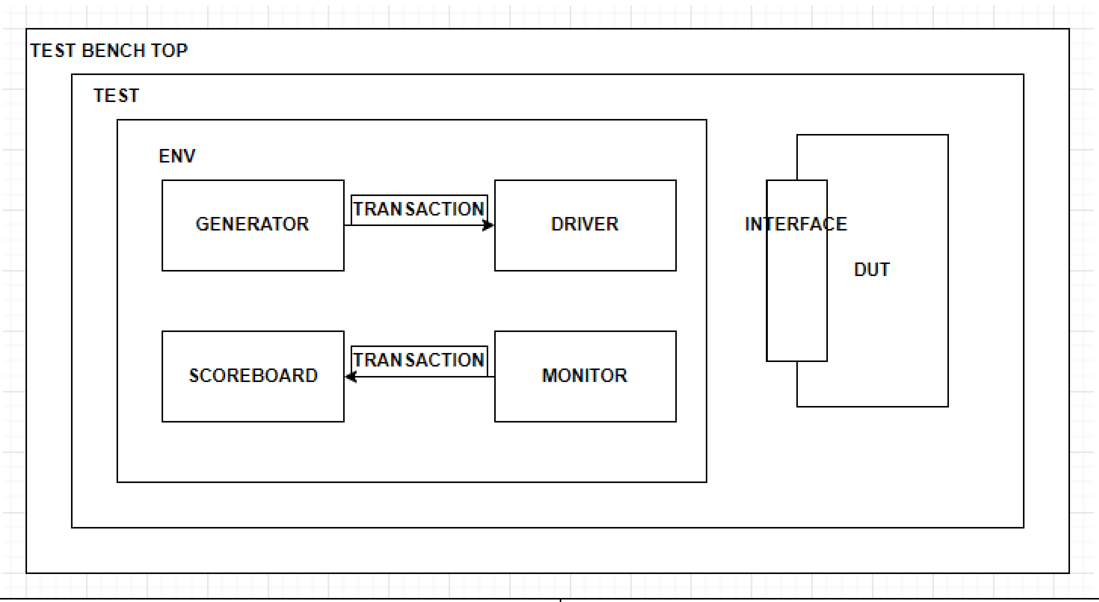
Class Based Testbench Hierarchy Overview

Verification Strategy:

Functional Verification of Asynchronous FIFO is a tedious task as two different clock domains are involved namely the write clock and the read clock. This milestone and the next milestone deal with us as a team coming with a system Verilog based OOP verification environment and provide deterministic testcases. We have come up with the idea of using Gray box verification, gray box verification typically means input signals are triggered accordingly to check the functional output of DUT and use it for comparison with the reference model.

Components of System Verilog Test Environment:



COMPONENT	DESCRIPTION
Generator	Generates different input stimulus to be drive to DUT (Design Under Test).
Interface	Contains Design Signals that can be driven or monitored
Driver	Drives the generated stimulus to the design.
Monitor	Monitor the design input-output ports to capture design activity.
Scoreboard	Checks output from the design with expected behavior.
Environment	Contains all the verification components mentioned below.
Test	Contains the environment that can be tweaked with different configuration settings.

Table:1 Test Bench Environment and Components (Source: Chip Verify)

- **Interface** – If the design contains hundred of port signals it would be difficult to connect, maintain and re-use those signals. Interface contains all design input-output ports into a container and the design can be driven with values through this interface.
- **Driver –** Driver as name suggests drives the values to the DUT through a pre-defined task in the driver class. This is the level of abstraction required to make test benches more flexible and scalable.
- **Scoreboard –** Scoreboard can have a reference model that behaves the same way as DUT, this model reflects the expected behavior of the DUT. Input sent to the DUT is also sent to the reference model.
- **Environment –** It makes the verification more flexible and scalable because more components can be plugged into the same environment for a future project. Environment class encapsulates major components of testbench – generator, driver, scoreboard, monitor.

Test Top: The test will instantiate an object of the environment and configure the way the test wants to. Testbench top module orchestrates the test environment, manages clock generation (using clocking blocks), reset generation (through driver class), interface connectivity. All the above-mentioned components collectively manage data-exchange, randomize input generations using randomize methods on the inputs which have been declared as rand or randc (here in Asynchronous FIFO, input data is randomized), transfer data and result analysis.

Required Tools: Siemens EDA Questa tool is used for OOP System Verilog simulation environment and debugging of Asynchronous FIFO, another tool used by our team is edaplayground.

Functions and Test Case Suites:

TEST CASES	DESCRIPTIONS
FIFO Reset	Reset task in driver class checks the reset condition to see if the FIFO has arrived to a known state.
FIFO Full	Drive the FIFO to completely fill the depth and check if the full flag is triggered or not.
FIFO Empty	Check to see if the empty flag is triggered when there is no data being driven to the FIFO and check if the empty flag is triggered when whole of the FIFO is being read.

Resources:

- Professor Venkatesh Patil slides
- Chipverify (https://www.chipverify.com/systemverilog/systemverilog-simple-testbench)
- Verification Academy (https://verificationacademy.com/forums/t/verification-asynchronous-fifo-cummings/41199)

Currently, out System Verilog Verification environment works fine for 7 write and read transactions and failing for transaction 8 and above, this environment creation is a ongoing process for milestone3.

Transcript1 – Shows the transactions for 5 reads and writes

Transcript2 – shows transdactions for 8 reads and writes.