UVM 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. The primary objective of this milestone is creating a UVM based Test environment. The verification plan aims to thoroughly validate the asynchronous data transfer, storage, and retrieval processes within the FIFO, while also confirming the correct handling of potential corner cases and error scenarios.

Testbench Architecture; Component used (list and describe Drivers, Monitors, scoreboards, checkers etc.)

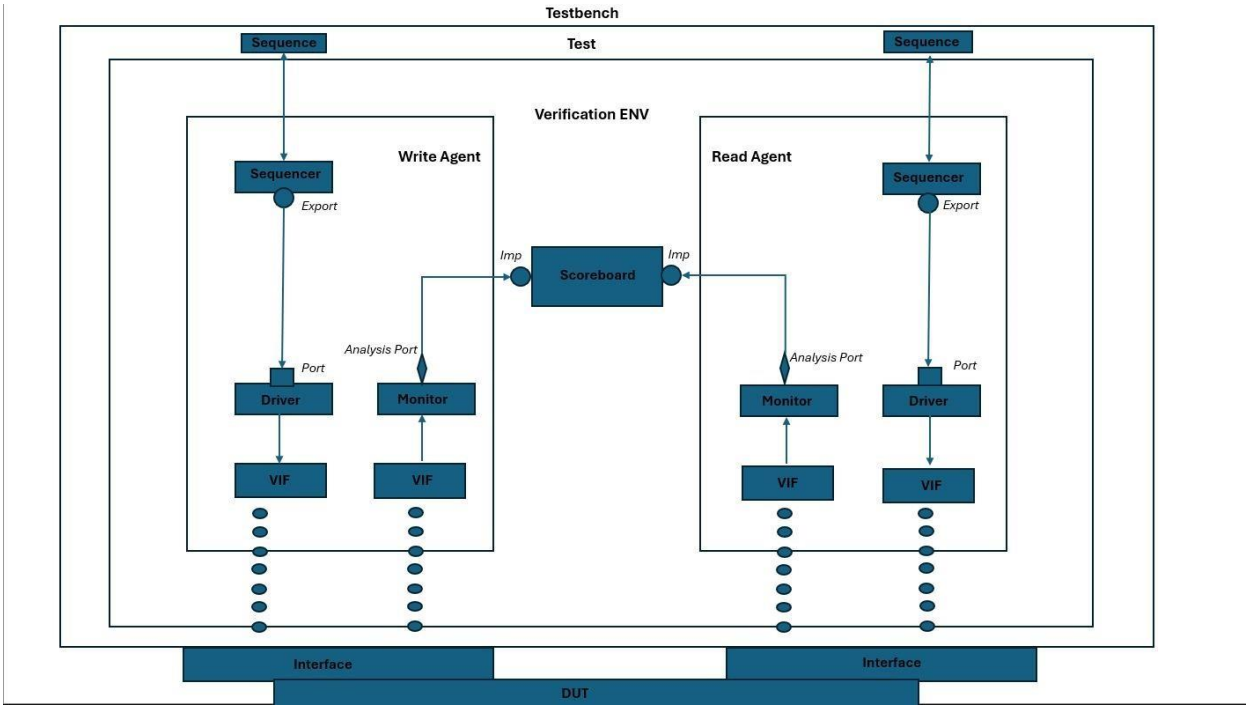

Figure : Testbench components and flow

Testbench architecture used will include:

- Testbench Top
  - Highest level of the testbench Hierarchy
  - Contains Environment instantiation and configuration
  - instantiates the DUT and the test sequence
- Environment
  - Contains the agents and components for driving stimulus and checking DUT responses
  - Instantiates the agents
- Agents (write and read)

    - Responsible for interfacing with DUT interface
    - Contains components to drive stimulus and collect responses
        - Agent Sequencer
            - Generates sequence of transactions
        - Agent Driver
            - Drives the stimulus to the DUT
        - Agent Monitor
            - Monitors signals on the interface of the DUT
            - Collects data for analysis and scoreboard
- Scoreboard
    - Compares expected results with actual results from DUT
    - Raises error flags on data mismatch
- Sequences
    - Contains sequence of transactions that represent specific test scenarios
    - Sequence Item represents a single traction (Data packet)
        - Contains information for driving and checking
        - Each test contains separate sequence

Implementation

- Expected Data Flow
    - TB Top Initiates test sequence
    - Environment instantiates and configures the agent
    - Agent
        - Generates sequences
        - Drives transactions to the DUT interface
        - Monitors the DUT behavior (outputs)
    - Scoreboard compares input/outputs and flags discrepancies

- Agents used
    - Write Data Agent
    - Read Data Agent

Test Scenarios:

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.

Transcript:

```
#       (Specify +UVM_NO_RELNOTES to turn off this notice)
#
# UVM_INFO verilog_src/questa_uvm_pkg-1.2/src/questa_uvm_pkg.sv(277) @ 0: reporter [Questa UVM] QUESTA_UVM-1.2.3
# UVM_INFO verilog_src/questa_uvm_pkg-1.2/src/questa_uvm_pkg.sv(278) @ 0: reporter [Questa UVM]  questa_uvm::init(+struct)
# UVM_INFO @ 0: reporter [RNTST] Running test fifo_test...
# UVM_INFO @ 0: reporter [UVMTOP] UVM testbench topology:
# ----------------------------------------------------------------------
# Name                        Type                              Size  Value
# ----------------------------------------------------------------------
# uvm_test_top                fifo_test                         -     @471
#   env                       fifo_env                          -     @478
#     rd_agnt                 read_agent                        -     @492
#       rd_drv                read_driver                       -     @507
#         rsp_port            uvm_analysis_port                 -     @522
#         seq_item_port       uvm_seq_item_pull_port            -     @514
#       rd_mon                read_monitor                      -     @639
#         rd_monitor_port     uvm_analysis_port                 -     @647
#       rd_seqr               read_sequencer                    -     @530
#         rsp_export          uvm_analysis_export               -     @537
#         seq_item_export     uvm_seq_item_pull_imp             -     @631
#         arbitration_queue   array                             0     -
#         lock_queue          array                             0     -
#         num_last_reqs       integral                          32    'd1
#         num_last_rsps       integral                          32    'd1
#     scb                     fifo_scoreboard                   -     @499
#       scoreboard_read_port  uvm_analysis_imp_rd_monitor_port  -     @669
#       scoreboard_write_port uvm_analysis_imp_monitor_port     -     @661
#     wr_agnt                 write_agent                       -     @485
#       wr_drv                write_driver                      -     @678
#         rsp_port            uvm_analysis_port                 -     @693
#         seq_item_port       uvm_seq_item_pull_port            -     @685
#       wr_mon                write_monitor                     -     @810
#         monitor_port        uvm_analysis_port                 -     @818
#       wr_seqr               write_sequencer                   -     @701
#         rsp_export          uvm_analysis_export               -     @708
#         seq_item_export     uvm_seq_item_pull_imp             -     @802
#         arbitration_queue   array                             0     -
#         lock_queue          array                             0     -
#         num_last_reqs       integral                          32    'd1
#         num_last_rsps       integral                          32    'd1
# ----------------------------------------------------------------------
```

```
#
# UVM_INFO uvm_scoreboard.sv(69) @ 770: uvm_test_top.env.scb [SCOREBOARD] MATCH Expected Data: 7 --- DUT Read Data: 7
# UVM_INFO uvm_scoreboard.sv(69) @ 980: uvm_test_top.env.scb [SCOREBOARD] MATCH Expected Data: 16 --- DUT Read Data: 16
# UVM_INFO uvm_scoreboard.sv(69) @ 1050: uvm_test_top.env.scb [SCOREBOARD] MATCH Expected Data: 17 --- DUT Read Data: 17
# UVM_INFO uvm_scoreboard.sv(69) @ 1120: uvm_test_top.env.scb [SCOREBOARD] MATCH Expected Data: 28 --- DUT Read Data: 28
# UVM_INFO uvm_scoreboard.sv(69) @ 1190: uvm_test_top.env.scb [SCOREBOARD] MATCH Expected Data: 58 --- DUT Read Data: 58
# UVM_INFO uvm_scoreboard.sv(69) @ 1260: uvm_test_top.env.scb [SCOREBOARD] MATCH Expected Data: 4c --- DUT Read Data: 4c
# UVM_INFO uvm_scoreboard.sv(69) @ 1330: uvm_test_top.env.scb [SCOREBOARD] MATCH Expected Data: 31 --- DUT Read Data: 31
# UVM_INFO uvm_scoreboard.sv(69) @ 1400: uvm_test_top.env.scb [SCOREBOARD] MATCH Expected Data: a --- DUT Read Data: a
# UVM_INFO uvm_scoreboard.sv(69) @ 1470: uvm_test_top.env.scb [SCOREBOARD] MATCH Expected Data: 48 --- DUT Read Data: 48
# UVM_INFO uvm_scoreboard.sv(69) @ 1540: uvm_test_top.env.scb [SCOREBOARD] MATCH Expected Data: 5b --- DUT Read Data: 5b
# UVM_INFO uvm_scoreboard.sv(69) @ 1610: uvm_test_top.env.scb [SCOREBOARD] MATCH Expected Data: 32 --- DUT Read Data: 32
# UVM_INFO uvm_scoreboard.sv(69) @ 1820: uvm_test_top.env.scb [SCOREBOARD] MATCH Expected Data: 4a --- DUT Read Data: 4a
```

```
#
# Total Coverage By Instance (filtered view): 100.00%
#
#
```

Transcript attached in zip file.

References:

1. Professor examples from slides
2. Verification academy