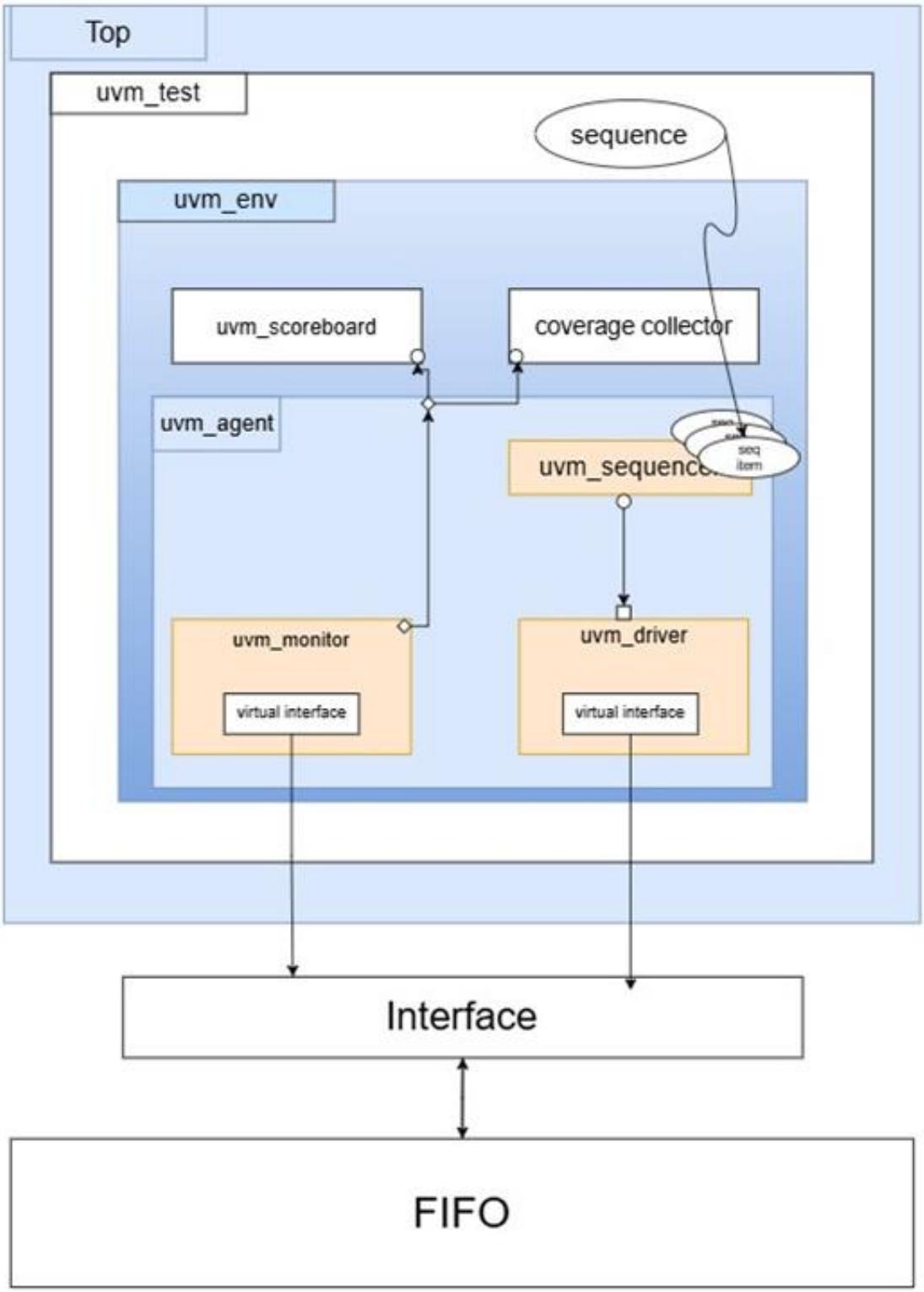
UVM PROJECT
SYNCHRONOUS FIFO

ABDELRAHMAN AHMED SAYED

❖ Contents:

UVM structure



🔧 How UVM Structure Works?

First, the top module init the clock and passes it to the interface. Then, the interface is passed to the FIFO (DUT). Additionally, the interface is placed into the UVM configuration database, but how?

To put the interface into the database, we must use a **virtual interface**.

During the test execution, the test retrieves this virtual interface and stores it in a variable within an object of the config class. This config object is then saved in the configuration database.

In the run phase, the test passes the sequence to the sequencer (details of which will be discussed later).

Within the test, we initialize the **environment (env)**. Inside the environment, we have:

- **Scoreboard**
- **Coverage**
- **Agent**

The agent itself contains:

- **Monitor**
- **Driver**
- **Sequencer**

Once the test passes the sequence to the sequencer, the sequencer forwards it to the driver using an internal mechanism. The driver waits until it hits get_next_item(), and then sends the seq_items.

These seq_items are passed using the previously stored configuration object. The monitor observes these transactions, captures both the sequence items and the DUT outputs, and forwards them to the **scoreboard** and **coverage collector** for checking and analysis.

Assertions can also be bound to the DUT to check protocol correctness and enhance verification quality.

Verification plan

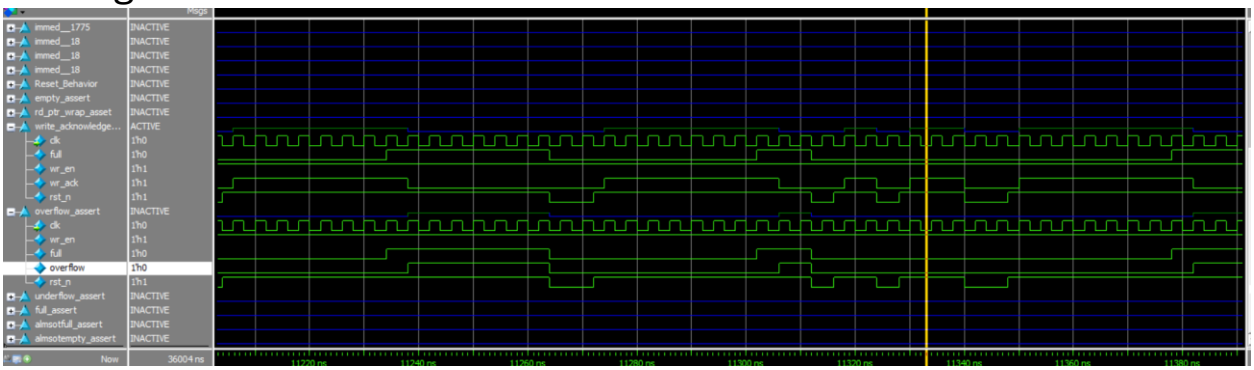
	A	B	C	D	E
1	**Label**	**Description**	**Stimulus Generation**	**Functional Coverage**	**Functionality Check**
2	FIFO_1	set rst_n to be 0 then 1 data_out should be 0 acting like nothing was there	direct in the Reset Seqeunce	----------	in check_data function in golden model
3	FIFO_2	receiving write_acknowledge after wr_en when not full	Randomization under constraints for write enable	coverpoint for the wr_en signal & for full signal	write_acknowledge_assert will check for the functionality
4	FIFO_3	receiving overflow hight when wr_en is high and full is high	Randomization under constraints for wr_en	covered by Cross_rd_wr_overflow	overflow_assert will check for the functionality
5	FIFO_4	receiving underflow when rd_en is high and empty is high	Randomization under constraints for rd_en	covered by Cross_rd_wr_empty	empty_assert will check for the functionality
6	FIFO_5	receiving full when internal siganl count is equal FIFO_DEPTH	Randomization under constraints for wr_en	covered by Cross_rd_wr_full	full_assert will check for the functionality
7	FIFO_6	receiving almsotfull when the internal signal count is equal to FIFO_DEPTH - 1	Randomization under constraints for wr_en	covered by Cross_rd_wr_almostf	almsotfull_assert will check for the functionality
8	FIFO_7	receiving almsotempty when the internal signal count is equal 1	Randomization under constraints for rd_en	covered by Cross_rd_wr_almostem	almsotempty_assert will check for the functionality
9	FIFO_8	receiving underflow when rd_en is high and empty is high	Randomization under constraints for rd_en	covered by Cross_rd_wr_empty	empty_assert will check for the functionality
10					

link

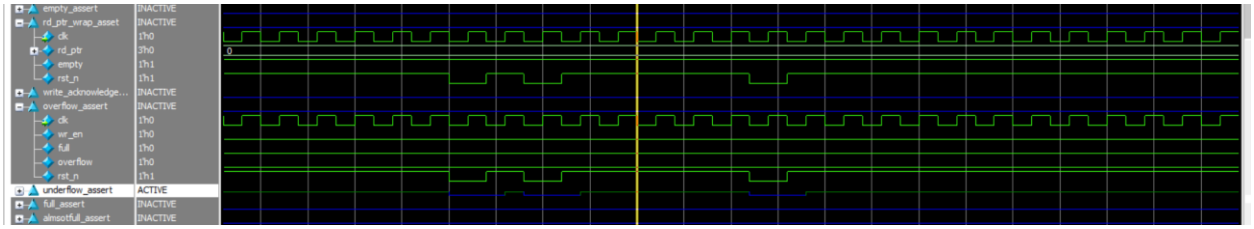
Questa Sim snippets:

```
# --- UVM Report Summary ---
#
# ** Report counts by severity
# UVM_INFO :27017
# UVM_WARNING :    0
# UVM_ERROR :    0
# UVM_FATAL :    0
# ** Report counts by id
# [Questa UVM]     2
# [RNTST]     1
# [TEST_DONE]     1
# [report_phase]     2
# [run_phase] 27011
# ** Note: $finish    : C:/questasim64_2021.1/win64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh(430)
#    Time: 36004 ns  Iteration: 61  Instance: /top
# Break in Task uvm_pkg/uvm_root::run_test at C:/questasim64_2021.1/win64/../verilog_src/uvm-1.1d/src/bas
```

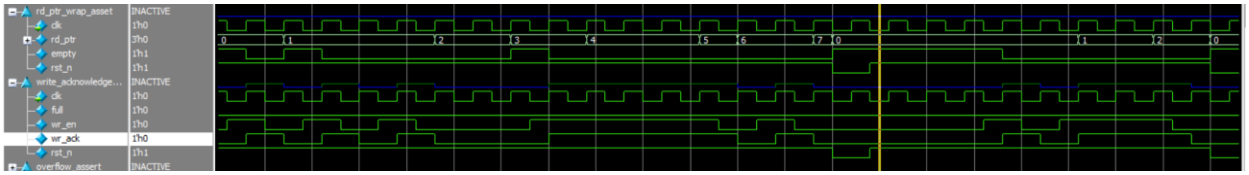
Writing:



Reading:



Writing and Reading:




```
stempty = x, underflow = x
# UVM_INFO fifo_mon.sv(46) @ 30680: uvm_test_top.env.agt.fifo_monitor [run_phase] rst_n = 1, wr_en = 1, rd_en = 0, data_in = 10231, data_out = 44432, wr_ack = 1, overflow = 0, full = 0, empty = 0, almostfull = 0, al
mostempty = 1, underflow = 0
# HERE in pushing 1
# UVM_INFO fifo_scoreboard.sv(53) @ 30680: uvm_test_top.env.fifo_scoreboard [run_phase] Correct output: rst_n = 1, wr_en = 1, rd_en = 0, data_in = 10231, data_out = 44432, wr_ack = 1, overflow = 0, full = 0, empty =
 0, almostfull = 0, almostempty = 1, underflow = 0
# UVM_INFO fifo_driver.sv(32) @ 30684: uvm_test_top.env.agt.fifo_driver [run_phase] rst_n = 1, wr_en = 0, rd_en = 0, data_in = 50665, data_out = x, wr_ack = x, overflow = x, full = x, empty = x, almostfull = x, almo
stempty = x, underflow = x
# UVM_INFO fifo_mon.sv(46) @ 30684: uvm_test_top.env.agt.fifo_monitor [run_phase] rst_n = 1, wr_en = 0, rd_en = 0, data_in = 50665, data_out = 44432, wr_ack = 0, overflow = 0, full = 0, empty = 0, almostfull = 0, al
mostempty = 1, underflow = 0
# UVM_INFO fifo_scoreboard.sv(53) @ 30684: uvm_test_top.env.fifo_scoreboard [run_phase] Correct output: rst_n = 1, wr_en = 0, rd_en = 0, data_in = 50665, data_out = 44432, wr_ack = 0, overflow = 0, full = 0, empty =
 0, almostfull = 0, almostempty = 1, underflow = 0
# UVM_INFO fifo_driver.sv(32) @ 30688: uvm_test_top.env.agt.fifo_driver [run_phase] rst_n = 1, wr_en = 1, rd_en = 1, data_in = 978, data_out = x, wr_ack = x, overflow = x, full = x, empty = x, almostfull = x, almost
empty = x, underflow = x
# UVM_INFO fifo_mon.sv(46) @ 30688: uvm_test_top.env.agt.fifo_monitor [run_phase] rst_n = 1, wr_en = 1, rd_en = 1, data_in = 978, data_out = 10231, wr_ack = 1, overflow = 0, full = 0, empty = 0, almostfull = 0, almo
stempty = 1, underflow = 0
# HERE in pushing popping
# UVM_INFO fifo_scoreboard.sv(53) @ 30688: uvm_test_top.env.fifo_scoreboard [run_phase] Correct output: rst_n = 1, wr_en = 1, rd_en = 1, data_in = 978, data_out = 10231, wr_ack = 1, overflow = 0, full = 0, empty = 0
, almostfull = 0, almostempty = 1, underflow = 0
# UVM_INFO fifo_driver.sv(32) @ 30692: uvm_test_top.env.agt.fifo_driver [run_phase] rst_n = 1, wr_en = 0, rd_en = 1, data_in = 23420, data_out = x, wr_ack = x, overflow = x, full = x, empty = x, almostfull = x, almo
stempty = x, underflow = x
# UVM_INFO fifo_mon.sv(46) @ 30692: uvm_test_top.env.agt.fifo_monitor [run_phase] rst_n = 1, wr_en = 0, rd_en = 1, data_in = 23420, data_out = 978, wr_ack = 0, overflow = 0, full = 0, empty = 1, almostfull = 0, almo
stempty = 0, underflow = 0
# HERE in popping 1
# UVM_INFO fifo_scoreboard.sv(53) @ 30692: uvm_test_top.env.fifo_scoreboard [run_phase] Correct output: rst_n = 1, wr_en = 0, rd_en = 1, data_in = 23420, data_out = 978, wr_ack = 0, overflow = 0, full = 0, empty = 1
, almostfull = 0, almostempty = 0, underflow = 0
# UVM_INFO fifo_driver.sv(32) @ 30696: uvm_test_top.env.agt.fifo_driver [run_phase] rst_n = 1, wr_en = 0, rd_en = 1, data_in = 16037, data_out = x, wr_ack = x, overflow = x, full = x, empty = x, almostfull = x, almo
stempty = x, underflow = x
# UVM_INFO fifo_mon.sv(46) @ 30696: uvm_test_top.env.agt.fifo_monitor [run_phase] rst_n = 1, wr_en = 0, rd_en = 1, data_in = 16037, data_out = 978, wr_ack = 0, overflow = 0, full = 0, empty = 1, almostfull = 0, almo
stempty = 0, underflow = 1
# EMpty
# UVM_INFO fifo_scoreboard.sv(53) @ 30696: uvm_test_top.env.fifo_scoreboard [run_phase] Correct output: rst_n = 1, wr_en = 0, rd_en = 1, data_in = 16037, data_out = 978, wr_ack = 0, overflow = 0, full = 0, empty = 1
, almostfull = 0, almostempty = 0, underflow = 1
# UVM_INFO fifo_driver.sv(32) @ 30700: uvm_test_top.env.agt.fifo_driver [run_phase] rst_n = 1, wr_en = 1, rd_en = 1, data_in = 17080, data_out = x, wr_ack = x, overflow = x, full = x, empty = x, almostfull = x, almo
```

```
mostempty = 1, underflow = 0
# HERE in popping 1
# UVM_INFO fifo_scoreboard.sv(53) @ 30728: uvm_test_top.env.fifo_scoreboard [run_phase] Correct output: rst_n = 1, wr_en = 0, rd_en = 1, data_in = 46875, data_out = 25932, wr_ack = 0, overflow = 0, full = 0, empty =
 0, almostfull = 0, almostempty = 1, underflow = 0
# UVM_INFO fifo_driver.sv(32) @ 30732: uvm_test_top.env.agt.fifo_driver [run_phase] rst_n = 1, wr_en = 1, rd_en = 0, data_in = 20312, data_out = x, wr_ack = x, overflow = x, full = x, empty = x, almostfull = x, almo
stempty = x, underflow = x
# UVM_INFO fifo_mon.sv(46) @ 30732: uvm_test_top.env.agt.fifo_monitor [run_phase] rst_n = 1, wr_en = 1, rd_en = 0, data_in = 20312, data_out = 25932, wr_ack = 1, overflow = 0, full = 0, empty = 0, almostfull = 0, al
mostempty = 0, underflow = 0
# HERE in pushing 1
# UVM_INFO fifo_scoreboard.sv(53) @ 30732: uvm_test_top.env.fifo_scoreboard [run_phase] Correct output: rst_n = 1, wr_en = 1, rd_en = 0, data_in = 20312, data_out = 25932, wr_ack = 1, overflow = 0, full = 0, empty =
 0, almostfull = 0, almostempty = 0, underflow = 0
# UVM_INFO fifo_driver.sv(32) @ 30736: uvm_test_top.env.agt.fifo_driver [run_phase] rst_n = 1, wr_en = 0, rd_en = 0, data_in = 47471, data_out = x, wr_ack = x, overflow = x, full = x, empty = x, almostfull = x, almo
stempty = x, underflow = x
# UVM_INFO fifo_mon.sv(46) @ 30736: uvm_test_top.env.agt.fifo_monitor [run_phase] rst_n = 1, wr_en = 0, rd_en = 0, data_in = 47471, data_out = 25932, wr_ack = 0, overflow = 0, full = 0, empty = 0, almostfull = 0, al
mostempty = 0, underflow = 0
# UVM_INFO fifo_scoreboard.sv(53) @ 30736: uvm_test_top.env.fifo_scoreboard [run_phase] Correct output: rst_n = 1, wr_en = 0, rd_en = 0, data_in = 47471, data_out = 25932, wr_ack = 0, overflow = 0, full = 0, empty =
 0, almostfull = 0, almostempty = 0, underflow = 0
# UVM_INFO fifo_driver.sv(32) @ 30740: uvm_test_top.env.agt.fifo_driver [run_phase] rst_n = 1, wr_en = 0, rd_en = 0, data_in = 51569, data_out = x, wr_ack = x, overflow = x, full = x, empty = x, almostfull = x, almo
stempty = x, underflow = x
# UVM_INFO fifo_mon.sv(46) @ 30740: uvm_test_top.env.agt.fifo_monitor [run_phase] rst_n = 1, wr_en = 0, rd_en = 0, data_in = 51569, data_out = 25932, wr_ack = 0, overflow = 0, full = 0, empty = 0, almostfull = 0, al
mostempty = 0, underflow = 0
# UVM_INFO fifo_scoreboard.sv(53) @ 30740: uvm_test_top.env.fifo_scoreboard [run_phase] Correct output: rst_n = 1, wr_en = 0, rd_en = 0, data_in = 51569, data_out = 25932, wr_ack = 0, overflow = 0, full = 0, empty =
 0, almostfull = 0, almostempty = 0, underflow = 0
# UVM_INFO fifo_driver.sv(32) @ 30744: uvm_test_top.env.agt.fifo_driver [run_phase] rst_n = 1, wr_en = 1, rd_en = 0, data_in = 21745, data_out = x, wr_ack = x, overflow = x, full = x, empty = x, almostfull = x, almo
stempty = x, underflow = x
# UVM_INFO fifo_mon.sv(46) @ 30744: uvm_test_top.env.agt.fifo_monitor [run_phase] rst_n = 1, wr_en = 1, rd_en = 0, data_in = 21745, data_out = 25932, wr_ack = 1, overflow = 0, full = 0, empty = 0, almostfull = 0, al
mostempty = 0, underflow = 0
# HERE in pushing 1
# UVM_INFO fifo_scoreboard.sv(53) @ 30744: uvm_test_top.env.fifo_scoreboard [run_phase] Correct output: rst_n = 1, wr_en = 1, rd_en = 0, data_in = 21745, data_out = 25932, wr_ack = 1, overflow = 0, full = 0, empty =
 0, almostfull = 0, almostempty = 0, underflow = 0
# UVM_INFO fifo_driver.sv(32) @ 30748: uvm_test_top.env.agt.fifo_driver [run_phase] rst_n = 1, wr_en = 0, rd_en = 0, data_in = 16462, data_out = x, wr_ack = x, overflow = x, full = x, empty = x, almostfull = x, almo
stempty = x, underflow = x
# UVM_INFO fifo_mon.sv(46) @ 30748: uvm_test_top.env.agt.fifo_monitor [run_phase] rst_n = 1, wr_en = 0, rd_en = 0, data_in = 16462, data_out = 25932, wr_ack = 0, overflow = 0, full = 0, empty = 0, almostfull = 0, al
```

HARD EXPLAING:

When writing the data_in is queuing in the queue :

```
# (12) 0
# ------------------------------------------------------
# UVM_INFO fifo_scoreboard.sv(53) @ 12: uvm_test_top.env.fifo_scoreboard [run_phase] Correct output: rst_n = 1, wr_en = 1, rd_en = 0, data_in = 100
stfull = 0, almostempty = 0, underflow = 0
# UVM_INFO fifo_driver.sv(32) @ 16: uvm_test_top.env.agt.fifo_driver [run_phase] rst_n = 1, wr_en = 1, rd_en = 0, data_in = 28851, data_out = x, wr_
mpty = x, underflow = x
# UVM_INFO fifo_mon.sv(46) @ 16: uvm_test_top.env.agt.fifo_monitor [run_phase] rst_n = 1, wr_en = 1, rd_en = 0, data_in = 28851, data_out = x, wr_a
ty = 0, underflow = 0
# HERE in pushing 1
# ------------------INSIDE GOLDEN MODEL--------------
# When rst_n: 1, wr_en: 1, rd_en: 0, data_in: 28851
# size of queue: 3
# (1) 63824
# (2) 1003
# (3) 28851
# (4) 0
# (5) 0
# (6) 0
# (7) 0
# (8) 0
# (9) 0
# (10) 0
# (11) 0
```

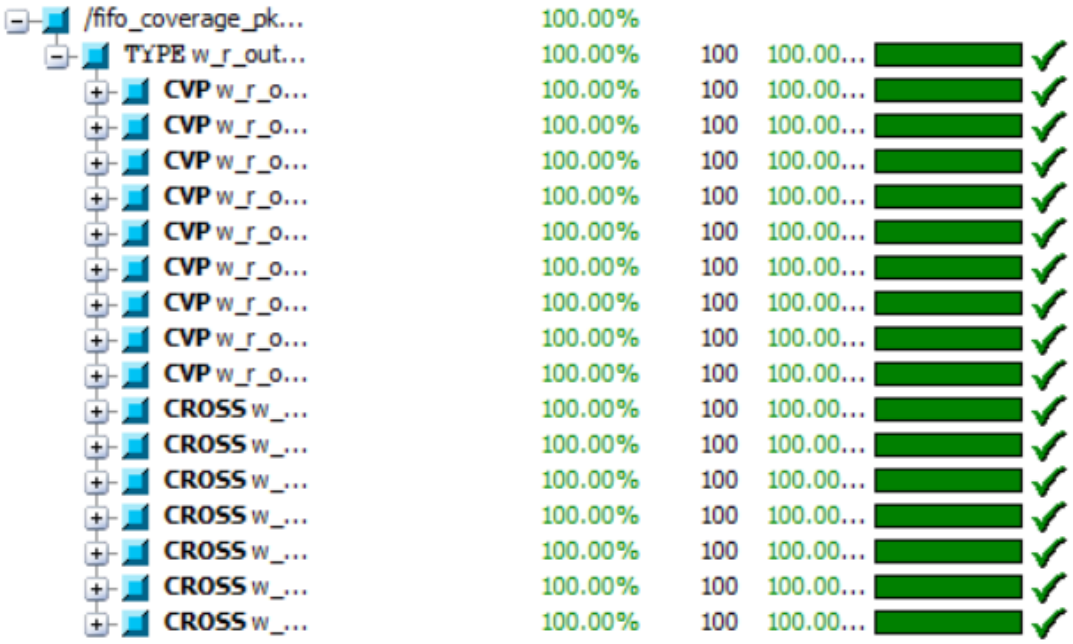
When reading, the first in the queue is the last out :

```
# ------------------INSIDE GOLDEN MODEL--------------
# When rst_n: 1, wr_en: 0, rd_en: 1, data_in: 19497
# size of queue: 7
# (1)1003
# (2)28851
# (3)59564
# (4)31353
# (5)47039
# (6)14784
# (7)22582
# (8)0
# (9)0
# (10)0
# (11)0
# (12)0
```

out

```
.fifo_scoreboard [run_phase] Correct output: rst_n = 1, wr_en = 0, rd_en = 1, data_in = 15650, data_out = 1003, wr_ack = 0, overflow = 0, full = 0, empty =
```

FCoverage:



Assertions coverage :

coverage_all.txt

```
/\top#dut /sva_f/rd_ptr_wrap_cover       fifo_sva Verilog  SVA  fifo_sva.sv(22)    3 Covered
/\top#dut /sva_f/Write_Acknowledge_cover fifo_sva Verilog  SVA  fifo_sva.sv(29) 2588 Covered
/\top#dut /sva_f/overflow_cover          fifo_sva Verilog  SVA  fifo_sva.sv(38) 1015 Covered
/\top#dut /sva_f/underflow_cover         fifo_sva Verilog  SVA  fifo_sva.sv(47) 2801 Covered
/\top#dut /sva_f/full_cover              fifo_sva Verilog  SVA  fifo_sva.sv(55) 1139 Covered
/\top#dut /sva_f/almsotfull_cover        fifo_sva Verilog  SVA  fifo_sva.sv(64)  167 Covered
/\top#dut /sva_f/almsotempty_cover       fifo_sva Verilog  SVA  fifo_sva.sv(73) 1107 Covered

TOTAL DIRECTIVE COVERAGE: 100.00%  COVERS: 8

ASSERTION RESULTS:
--------------------------------------------------------------------
Name                File(Line)                   Failure      Pass
                                                 Count        Count
--------------------------------------------------------------------
/\top#dut /sva_f/Reset_Behavior
                    fifo_sva.sv(5)                     0          1
/\top#dut /sva_f/empty_assert
                    fifo_sva.sv(12)                    0          1
/\top#dut /sva_f/rd_ptr_wrap_asset
                    fifo_sva.sv(21)                    0          1
/\top#dut /sva_f/write_acknowledge_assert
                    fifo_sva.sv(28)                    0          1
/\top#dut /sva_f/overflow_assert
                    fifo_sva.sv(37)                    0          1
/\top#dut /sva_f/underflow_assert
                    fifo_sva.sv(46)                    0          1
/\top#dut /sva_f/full_assert
                    fifo_sva.sv(54)                    0          1
/\top#dut /sva_f/almsotfull_assert
                    fifo_sva.sv(63)                    0          1
/\top#dut /sva_f/almsotempty_assert
                    fifo_sva.sv(72)                    0          1

Total Coverage By Instance (filtered view): 100.00%

End time: 22:37:08 on May 06,2025, Elapsed time: 0:00:00
Errors: 0, Warnings: 0
```

Do file:

run.do

```
vlib work
vlog -f src_files.list +define+DEBUG
#  +define+DEBUG
vsim -voptargs=+acc work.top -classdebug -uvmcontrol=all
add wave /top/fifo_interface_ins/*
add wave /uvm_pkg::uvm_reg_map::do_write/#ublk#215181159#1731/immed__1735 /uvm_pkg::uvm_reg_map::do_read/#ublk#215181159#1771/immed__1775 /f
coverage save FIFO_cov.ucdb -onexit -du work.FIFO
run -all
```

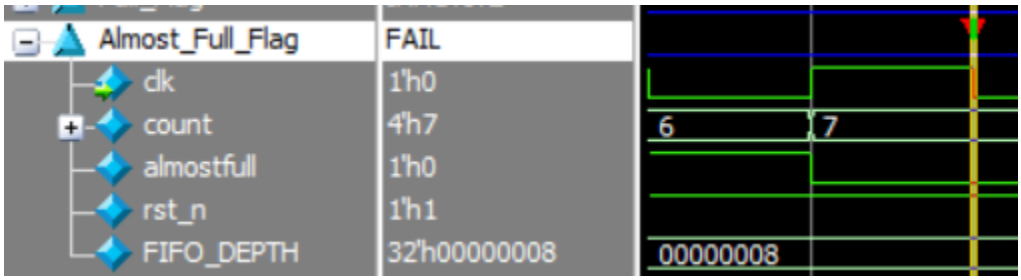
BUGS

- **Bug 1(almostfull)**

In the design:

```
62
63   assign fifo_if.almostfull = (count == FIFO_DEPTH-2)? 1 : 0; /
64
```

In assertions:



It should be:

```
assign fifo_if.almostfull = (count == FIFO_DEPTH-1)? 1 : 0; /
```

Name	Language	Enable	Failure Count	Assertion Type
/FIFO_top/dut/Reset_Behavior	SVA	on	0	Immediate
/FIFO_top/dut/Write_Acknowledge	SVA	on	0	Concurrent
/FIFO_top/dut/Overflow_Detection	SVA	on	7	Concurrent
/FIFO_top/dut/Underflow_Detection	SVA	on	0	Concurrent
/FIFO_top/dut/Empty_Flag	SVA	on	0	Concurrent
/FIFO_top/dut/Full_Flag	SVA	on	0	Concurrent
/FIFO_top/dut/Almost_Full_Flag	SVA	on	0	Concurrent
/FIFO_top/dut/Almost_Empty_Flag	SVA	on	0	Concurrent
/FIFO_top/fifo_tb_inst/#ublk#182146786#12/immed__15	SVA	on	0	Immediate

- **BUG 2** (wr_ack)

In design:

missing this part:

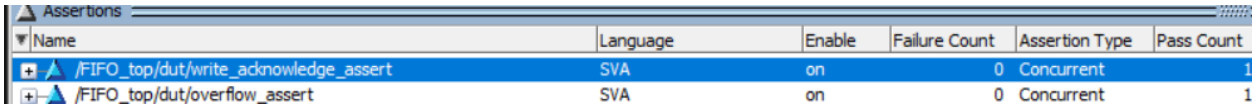
```
reg [max_fifo_addr-1:0] wr_ptr, rd_ptr;
reg [max_fifo_addr:0] count;

always @(posedge fifo_if.clk or negedge fifo_if.rst_n) begin
    if (!fifo_if.rst_n) begin
        wr_ptr <= 0;
        fifo_if.wr_ack <= 0; /// should make it 0 again if rst_n active
    end
    else if (fifo_if.wr_en && count < FIFO_DEPTH) begin
        mem[wr_ptr] <= fifo_if.data_in;
```

Before it:

Assertions

Name	Language	Enable	Failure Count	Assertion Type
/FIFO_top/dut/write_acknowledge_assert	SVA	on	386	Concurrent
/FIFO_top/dut/overflow_assert	SVA	on	0	Concurrent

After it:

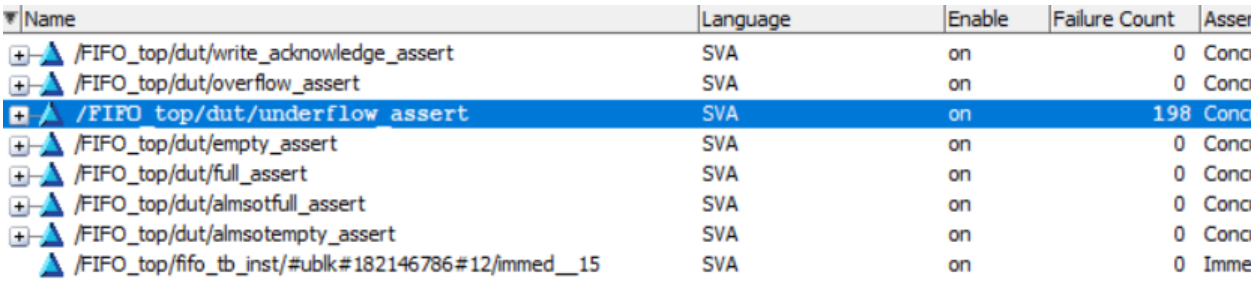


- **BUG 3(underflow)**

In the design:

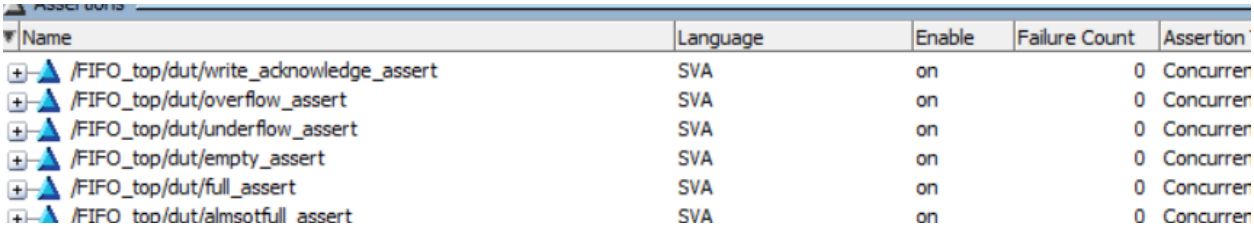
```
assign fifo_if.empty = (count == 0)? 1 : 0;
// assign fifo_if.underflow = (fifo_if.empty && fifo_if.rd_en)? 1 : 0;  ////// SHOULD BE SEQ not
assign fifo_if.almostfull = (count == FIFO_DEPTH-1)? 1 : 0; /////// FIXING
```

Underflow should be sequential



Should be like this:

```
always @(posedge fifo_if.clk or negedge fifo_if.rst_n) begin
    if (!fifo_if.rst_n) begin
        rd_ptr <= 0;
        fifo_if.underflow <= 0;
    end
    else if (fifo_if.rd_en && count != 0) begin
        fifo_if.data_out <= mem[rd_ptr];
        rd_ptr <= rd_ptr + 1;
    end
    else begin
        if(count == 0 && fifo_if.rd_en)begin
            fifo_if.underflow <= 1;
        end
        else begin
            fifo_if.underflow <= 0;
        end
    end
```



Name	Language	Enable	Failure Count	Assertion
/FIFO_top/dut/write_acknowledge_assert	SVA	on	0	Concurren
/FIFO_top/dut/overflow_assert	SVA	on	0	Concurren
/FIFO_top/dut/underflow_assert	SVA	on	0	Concurren
/FIFO_top/dut/empty_assert	SVA	on	0	Concurren
/FIFO_top/dut/full_assert	SVA	on	0	Concurren
/FIFO_top/dut/almsotfull_assert	SVA	on	0	Concurren

- **BUG 4** **(missing when write and read high)**

```
    else if ( ({fifo_if.wr_en, fifo_if.rd_en} == 2'b11) && fifo_if.empty)
        count <= count + 1;
    else if ( ({fifo_if.wr_en, fifo_if.rd_en} == 2'b11) && fifo_if.full)
        count <= count - 1;
```

Assertions table

A	B
Feature	Assertion
Whenever the FIFO is full, `wr_ack` is always = 0	@(posedge clk) (full |-> !wr_ack)`
When FIFO count is 0, `empty` must be high	@(posedge clk) disable iff(!rst_n) (FIFO.count == 0) |-> fifo_interface_ins.empty
If not full and `wr_en` is high, `wr_ack` must be high in next cycle	@(posedge clk) disable iff(!rst_n) (!full && wr_en) |=> wr_ack
If full and `wr_en` is high, `overflow` must be high in next cycle	@(posedge clk) disable iff(!rst_n) (full && wr_en) |=> overflow
If empty and `rd_en` is high, `underflow` must be high in next cycle	@(posedge clk) disable iff(!rst_n) (empty && rd_en) |=> underflow
When FIFO count reaches depth, `full` must be high	@(posedge clk) disable iff(!rst_n) (FIFO.count == FIFO.FIFO_DEPTH) |-> full
When FIFO count is one less than depth, `almostfull` must be high	@(posedge clk) disable iff(!rst_n) (FIFO.count == FIFO.FIFO_DEPTH - 1) |-> almostfull
When FIFO count is 1, `almostempty` must be high	@(posedge clk) disable iff(!rst_n) (FIFO.count == 1) |-> almostempty

THANK YOU