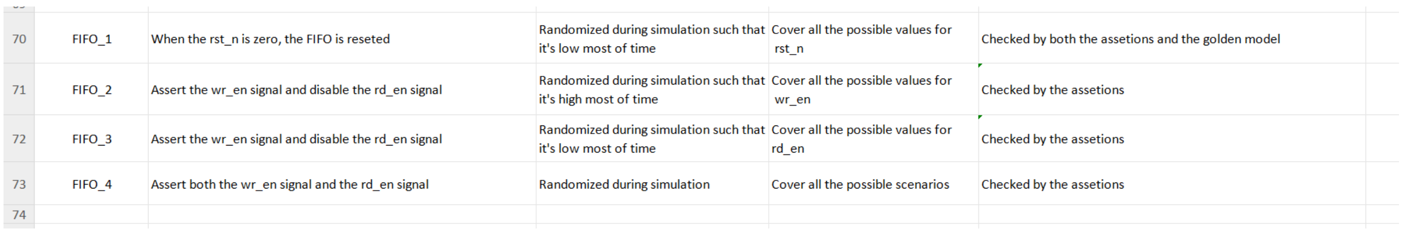
# FIFO

## UVM verification

**Name: Ali Mohamed Mohamed Saied**

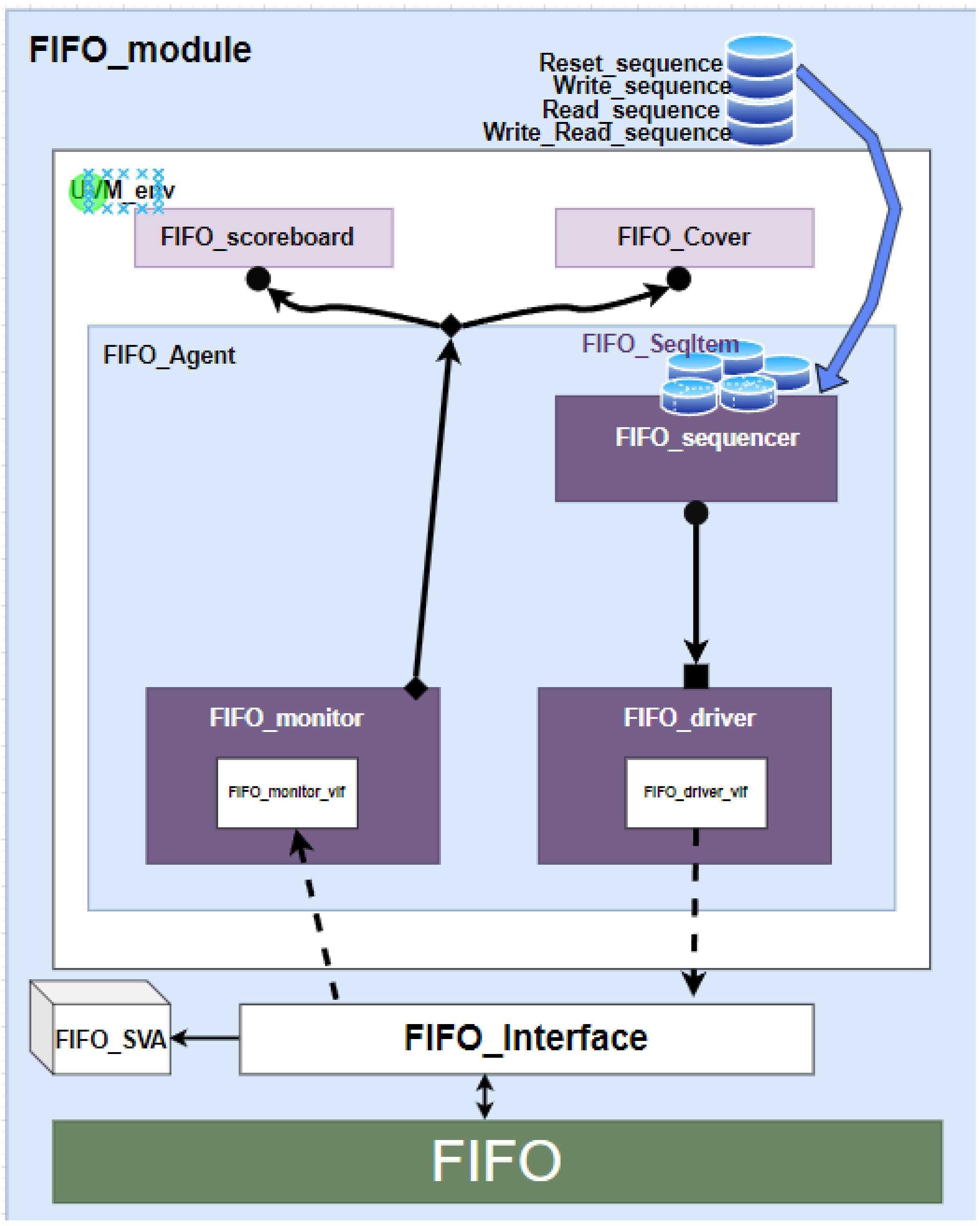
**Under supervision of : Eng. Kareem Wassem**

## 1:Verification plan: (Overview)



| | | | | | |
|---|---|---|---|---|---|
| 70 | FIFO_1 | When the rst_n is zero, the FIFO is reseted | Randomized during simulation such that it's low most of time | Cover all the possible values for rst_n | Checked by both the assetions and the golden model |
| 71 | FIFO_2 | Assert the wr_en signal and disable the rd_en signal | Randomized during simulation such that it's high most of time | Cover all the possible values for wr_en | Checked by the assetions |
| 72 | FIFO_3 | Assert the wr_en signal and disable the rd_en signal | Randomized during simulation such that it's low most of time | Cover all the possible values for rd_en | Checked by the assetions |
| 73 | FIFO_4 | Assert both the wr_en signal and the rd_en signal | Randomized during simulation | Cover all the possible scenarios | Checked by the assetions |
| 74 | | | | | |

## 2: UVM structure:



# <u>3:UVM testbench flow sequence:</u>

**<u>A: Top module (FIFO_top):</u>**

This is the module which is responsible for generating the system clock signal (clk), instantiate the interface module passing to it the generated clock signal, instantiate the design (DUT) passing to it the just-instantiated interface instance.

**<u>B: Interface (FIFO_Interface):</u>**

Inside this part, the system input and output signals are created and sent to the “modport” which determines which of this signal will be input and/or output depending on the “modport” type.

**<u>C: Design(FIFO.v):</u>**

It’s the module to be tested. It holds the main operation of the system.

**<u>D: Assertions module(FIFO_SVA):</u>**

Inside this module we place sequences which we need to test their occurrence such as the assertion of certain signal once another signal is asserted. This module is reponsible for doing this job.

**<u>E: UVM_test (FIFO_test):</u>**

It’s the package where the testing strategy is stated. The sequence of the testing process is mentioned there, such that by reading it you can realize which signal to be asserted first and which one second,….etc.
It does that by instantiating the uvm_sequence(mentioned nextly).

**<u>F: UVM_env(FIFO_env):</u>**

Inside this package we include the testing structure. We have various testing elements which are:
1-uvm_scoreboard. 2-Coverage collector. 3-uvm_agent.
These structures are classes each of them has a testing rule. Objects of these classes are decalared here in uvm_env.

**G: UVM_scoreboard(FIFO_Scoreboard):**

It's a class all its role is to check if the system output is correct depending on an available golden reference.

**H: Coverage collector(FIFO_Cover):**

It's the class responsible for making sure that all the required bits have experienced toggling at least one time(coverpoint), and also make sure that the required cross-coverage between various signals occurs.

**I: uvm_agent(FIFO_Agent):**

It's a class responsible for including both the stimulus holder for the system (uvm_driver), and the result holder for the checker classes(uvm_scoreboard and coverage-collector).

**I: uvm_sequence_item(FIFO_SeqItem):**

It's the class in which the values for the system input signals are created, some of these signals values may have some depend on some given "constraint" .

**K: uvm_sequence(FIFO_Write_seq, FIFO_Read_Seq,…..etc.):**

A class responsible for starting the process of generation of the system input signals, so that it can force some of these signals to certain values directly, while other signals are "randomized".

**L: uvm_driver(FIFO_driver):**

It's class responsible for holding the generated input signals values from where they are generated (uvm_sequence_item) to the design so that it can operate.

**M: uvm_sequencer(FIFO_Sequencer):**

This class work as arbiter between the uvm_sequence and the uvm_driver.

**N: uvm_monitor(FIFO_Monitor):**

A class responsible for holding the system output signals values from the interface to the place where they will be tested if they are correct or not (uvm_scoreboard and coverage-collector).

# 4: Testing results:

## I)Bug report:

1) Put brackets around the conditions in any if statement containing more than one condition gathered together to prevent false execution.
2) Adding this code snippet to the always block specified for "count":

```
else if(({wr_en, rd_en} == 2'b11)) begin
            if(count < FIFO_DEPTH)
                count <= count + 1;
            if(count != 0)
                count <= count - 1;
        end
```

3) Adding this code snippet to the always block speceified for "rd_en" and delete the assignment statement of "underflow":

```
else begin
        if (empty & rd_en)
            underflow <= 1;
        else
            underflow <= 0;
    end
```

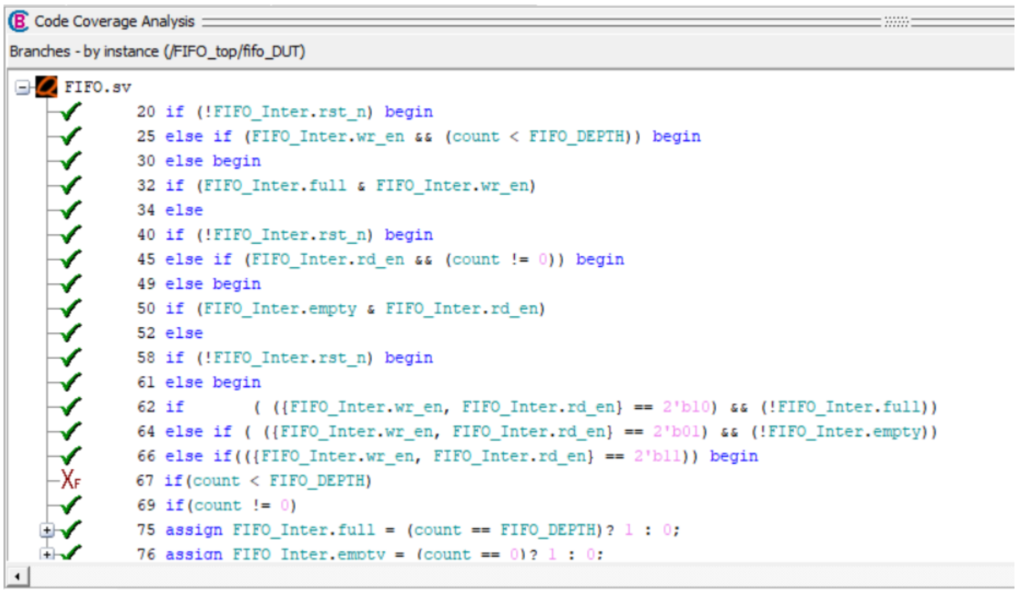
4) Modify the assignment statement of "almostempty" to be:

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

5) Reset signals overflow, wr_ack and underflow .

6)The data_out wasn't reset-ted when the rst_n signal is low.

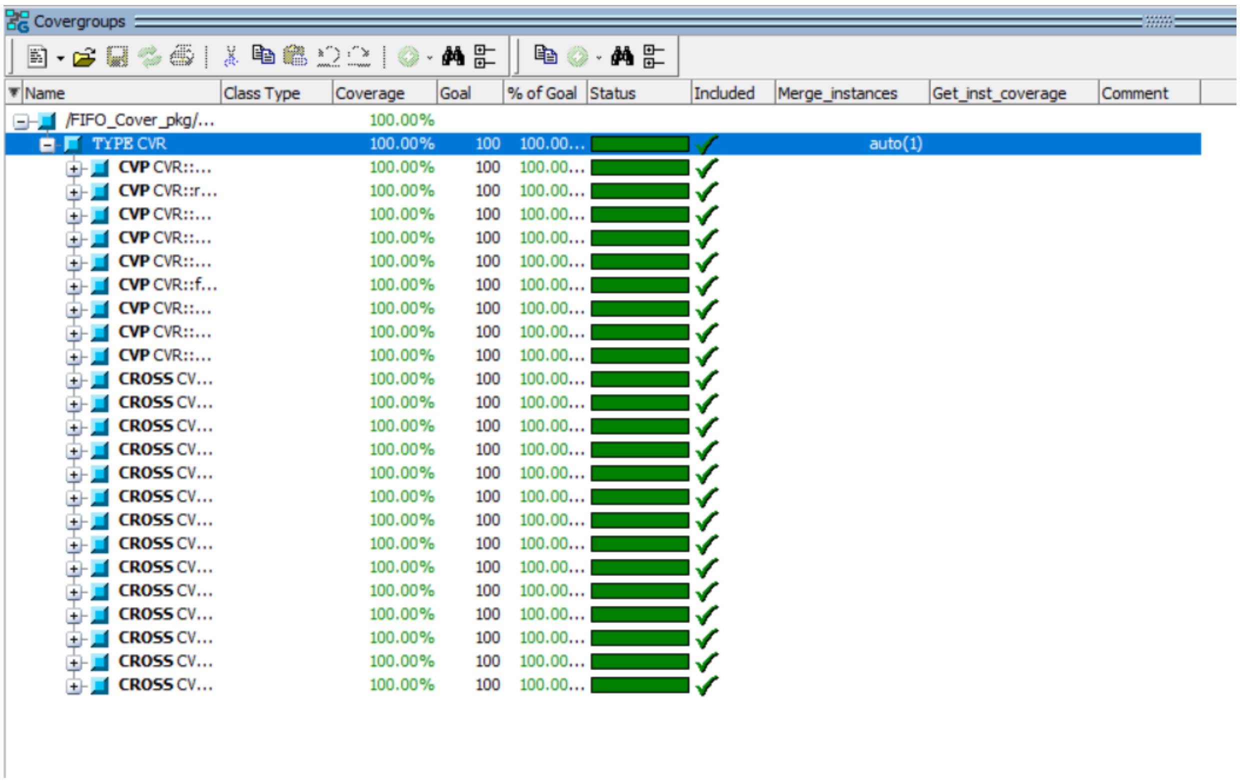
## II):Code coverage report, Functional coverage report, and Sequential domain coverage:



Code Coverage Analysis

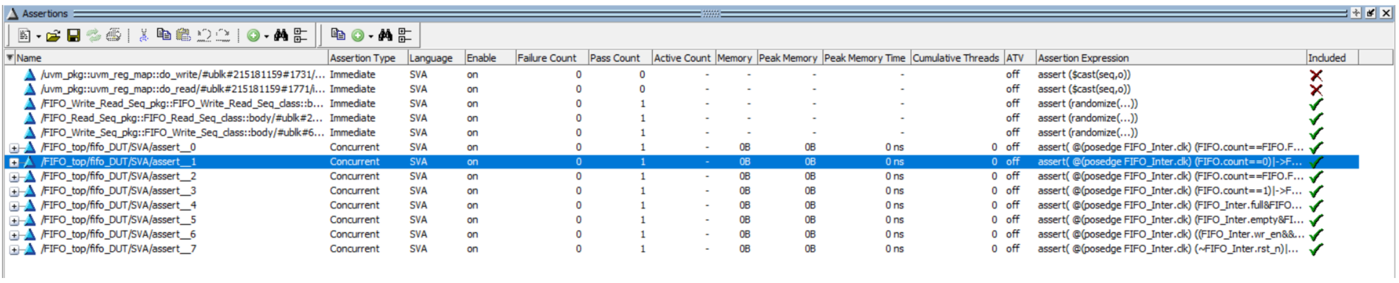
Branches - by instance (/FIFO_top/fifo_DUT)

```
FIFO.sv
✓    20 if (!FIFO_Inter.rst_n) begin
✓    25 else if (FIFO_Inter.wr_en && (count < FIFO_DEPTH)) begin
✓    30 else begin
✓    32 if (FIFO_Inter.full & FIFO_Inter.wr_en)
✓    34 else
✓    40 if (!FIFO_Inter.rst_n) begin
✓    45 else if (FIFO_Inter.rd_en && (count != 0)) begin
✓    49 else begin
✓    50 if (FIFO_Inter.empty & FIFO_Inter.rd_en)
✓    52 else
✓    58 if (!FIFO_Inter.rst_n) begin
✓    61 else begin
✓    62 if       ( ({FIFO_Inter.wr_en, FIFO_Inter.rd_en} == 2'b10) && (!FIFO_Inter.full))
✓    64 else if ( ({FIFO_Inter.wr_en, FIFO_Inter.rd_en} == 2'b01) && (!FIFO_Inter.empty))
✓    66 else if(({FIFO_Inter.wr_en, FIFO_Inter.rd_en} == 2'b11)) begin
XF   67 if(count < FIFO_DEPTH)
✓    69 if(count != 0)
✓    75 assign FIFO_Inter.full = (count == FIFO_DEPTH)? 1 : 0;
✓    76 assign FIFO_Inter.empty = (count == 0)? 1 : 0;
```



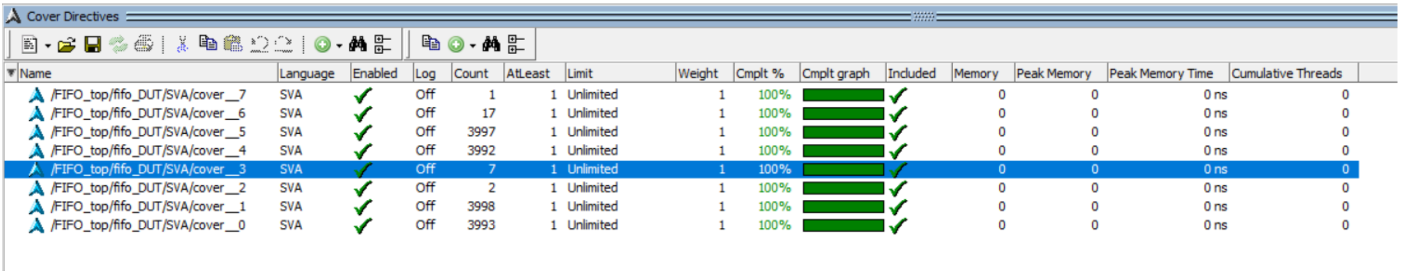
Covergroups

| Name | Class Type | Coverage | Goal | % of Goal | Status | Included | Merge_instances | Get_inst_coverage | Comment |
|---|---|---|---|---|---|---|---|---|---|
| /FIFO_Cover_pkg/... | | 100.00% | | | | | | | |
| TYPE CVR | | 100.00% | 100 | 100.00... | 🟩 | ✓ | auto(1) | | |
| CVP CVR::... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CVP CVR::r... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CVP CVR::... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CVP CVR::... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CVP CVR::... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CVP CVR::f... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CVP CVR::... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CVP CVR::... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CVP CVR::... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CROSS CV... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CROSS CV... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CROSS CV... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CROSS CV... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CROSS CV... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CROSS CV... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CROSS CV... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CROSS CV... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CROSS CV... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CROSS CV... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CROSS CV... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CROSS CV... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CROSS CV... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |
| CROSS CV... | | 100.00% | 100 | 100.00... | 🟩 | ✓ | | | |



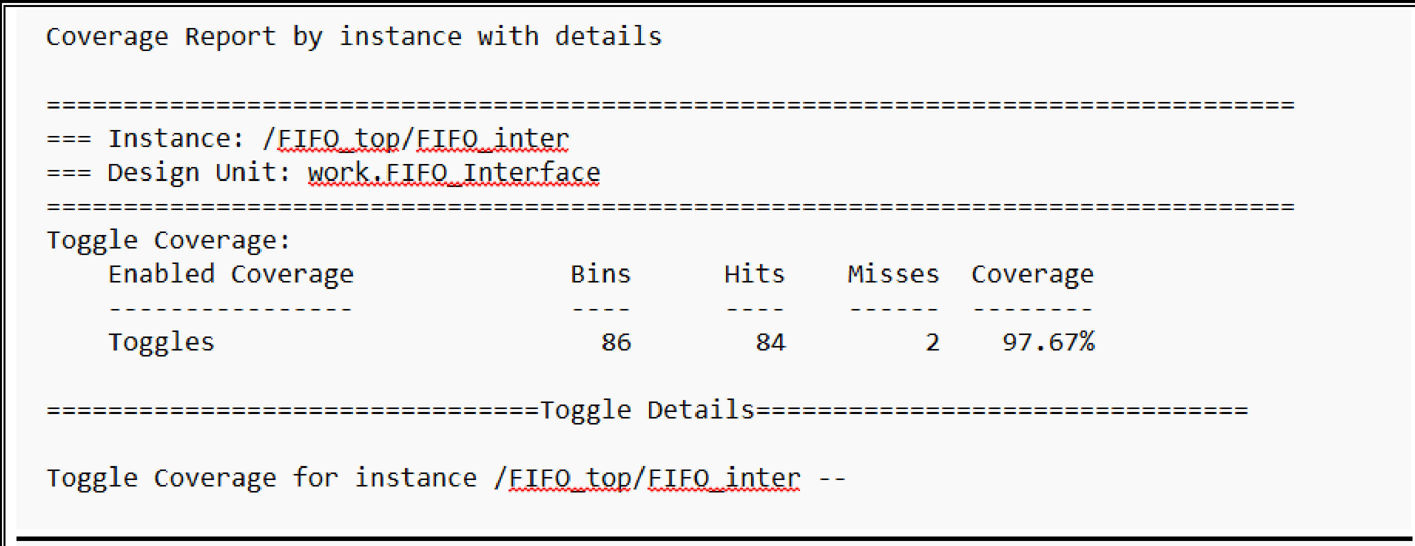
Assertions

| Name | Assertion Type | Language | Enable | Failure Count | Pass Count | Active Count | Memory | Peak Memory | Peak Memory Time | Cumulative Threads | ATV | Assertion Expression | Included |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| /uvm_pkg::uvm_reg_map::do_write/#ublk#215181159#1731/... | Immediate | SVA | on | 0 | 0 | - | - | - | - | | off | assert ($cast(seq,o)) | ✗ |
| /uvm_pkg::uvm_reg_map::do_read/#ublk#215181159#1771/i... | Immediate | SVA | on | 0 | 0 | - | - | - | - | | off | assert ($cast(seq,o)) | ✗ |
| /FIFO_Write_Read_Seq_pkg::FIFO_Write_Read_Seq_class::b... | Immediate | SVA | on | 0 | 1 | - | - | - | - | | off | assert (randomize(...)) | ✓ |
| /FIFO_Read_Seq_pkg::FIFO_Read_Seq_class::body/#ublk#2... | Immediate | SVA | on | 0 | 1 | - | - | - | - | | off | assert (randomize(...)) | ✓ |
| /FIFO_Write_Seq_pkg::FIFO_Write_Seq_class::body/#ublk#6... | Immediate | SVA | on | 0 | 1 | - | - | - | - | | off | assert (randomize(...)) | ✓ |
| /FIFO_top/fifo_DUT/SVA/assert__0 | Concurrent | SVA | on | 0 | 1 | - | 0B | 0B | 0 ns | 0 | off | assert( @(posedge FIFO_Inter.clk) (FIFO.count==FIFO.F... | ✓ |
| /FIFO_top/fifo_DUT/SVA/assert__1 | Concurrent | SVA | on | 0 | 1 | - | 0B | 0B | 0 ns | 0 | off | assert( @(posedge FIFO_Inter.clk) (FIFO.count==0)\|->F... | ✓ |
| /FIFO_top/fifo_DUT/SVA/assert__2 | Concurrent | SVA | on | 0 | 1 | - | 0B | 0B | 0 ns | 0 | off | assert( @(posedge FIFO_Inter.clk) (FIFO.count==FIFO.F... | ✓ |
| /FIFO_top/fifo_DUT/SVA/assert__3 | Concurrent | SVA | on | 0 | 1 | - | 0B | 0B | 0 ns | 0 | off | assert( @(posedge FIFO_Inter.clk) (FIFO.count==1)\|->F... | ✓ |
| /FIFO_top/fifo_DUT/SVA/assert__4 | Concurrent | SVA | on | 0 | 1 | - | 0B | 0B | 0 ns | 0 | off | assert( @(posedge FIFO_Inter.clk) (FIFO_Inter.full&FIFO... | ✓ |
| /FIFO_top/fifo_DUT/SVA/assert__5 | Concurrent | SVA | on | 0 | 1 | - | 0B | 0B | 0 ns | 0 | off | assert( @(posedge FIFO_Inter.clk) (FIFO_Inter.empty&FI... | ✓ |
| /FIFO_top/fifo_DUT/SVA/assert__6 | Concurrent | SVA | on | 0 | 1 | - | 0B | 0B | 0 ns | 0 | off | assert( @(posedge FIFO_Inter.clk) ((FIFO_Inter.wr_en&&... | ✓ |
| /FIFO_top/fifo_DUT/SVA/assert__7 | Concurrent | SVA | on | 0 | 1 | - | 0B | 0B | 0 ns | 0 | off | assert( @(posedge FIFO_Inter.clk) (~FIFO_Inter.rst_n)\|... | ✓ |

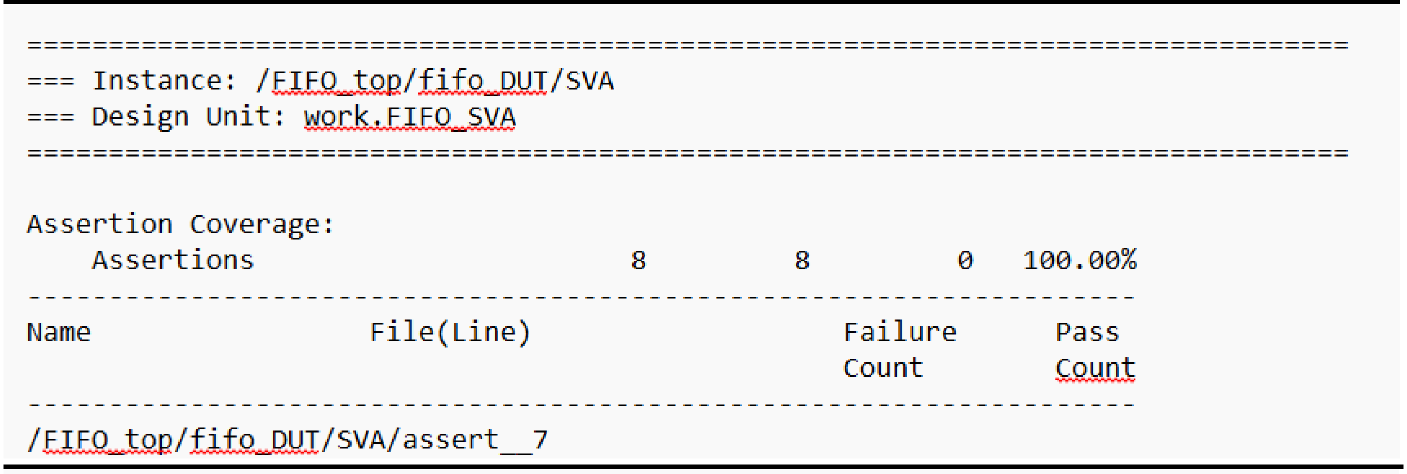


Cover Directives

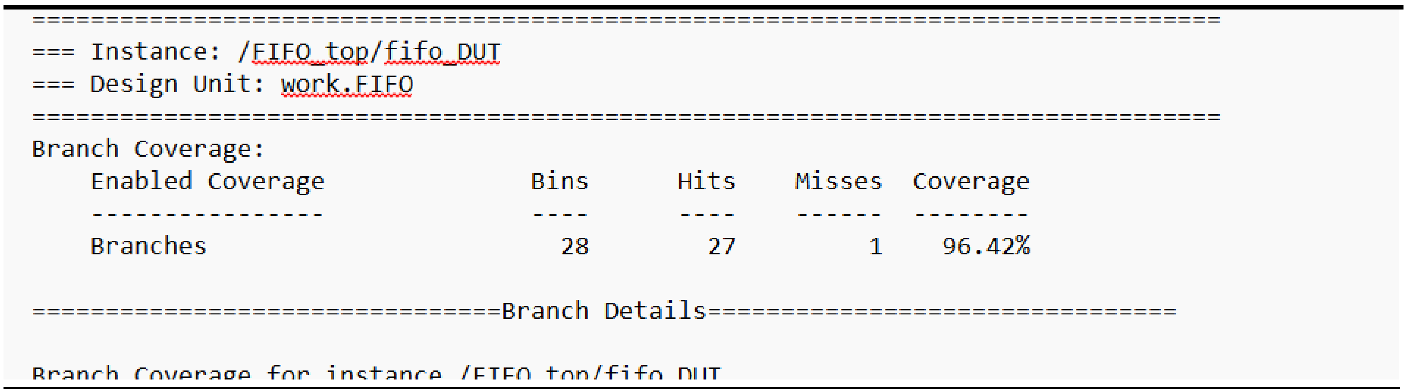
| Name | Language | Enabled | Log | Count | AtLeast | Limit | Weight | Cmplt % | Cmplt graph | Included | Memory | Peak Memory | Peak Memory Time | Cumulative Threads |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| /FIFO_top/fifo_DUT/SVA/cover__7 | SVA | ✓ | Off | 1 | 1 | Unlimited | 1 | 100% | 🟩 | ✓ | 0 | 0 | 0 ns | 0 |
| /FIFO_top/fifo_DUT/SVA/cover__6 | SVA | ✓ | Off | 17 | 1 | Unlimited | 1 | 100% | 🟩 | ✓ | 0 | 0 | 0 ns | 0 |
| /FIFO_top/fifo_DUT/SVA/cover__5 | SVA | ✓ | Off | 3997 | 1 | Unlimited | 1 | 100% | 🟩 | ✓ | 0 | 0 | 0 ns | 0 |
| /FIFO_top/fifo_DUT/SVA/cover__4 | SVA | ✓ | Off | 3992 | 1 | Unlimited | 1 | 100% | 🟩 | ✓ | 0 | 0 | 0 ns | 0 |
| /FIFO_top/fifo_DUT/SVA/cover__3 | SVA | ✓ | Off | 7 | 1 | Unlimited | 1 | 100% | 🟩 | ✓ | 0 | 0 | 0 ns | 0 |
| /FIFO_top/fifo_DUT/SVA/cover__2 | SVA | ✓ | Off | 2 | 1 | Unlimited | 1 | 100% | 🟩 | ✓ | 0 | 0 | 0 ns | 0 |
| /FIFO_top/fifo_DUT/SVA/cover__1 | SVA | ✓ | Off | 3998 | 1 | Unlimited | 1 | 100% | 🟩 | ✓ | 0 | 0 | 0 ns | 0 |
| /FIFO_top/fifo_DUT/SVA/cover__0 | SVA | ✓ | Off | 3993 | 1 | Unlimited | 1 | 100% | 🟩 | ✓ | 0 | 0 | 0 ns | 0 |



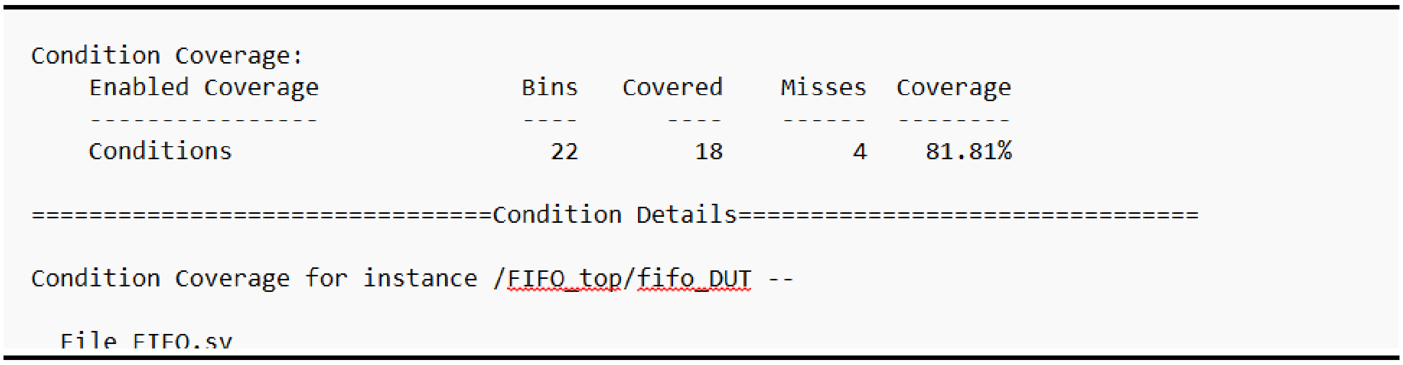
```
Coverage Report by instance with details

=================================================================================
=== Instance: /FIFO_top/FIFO_inter
=== Design Unit: work.FIFO_Interface
=================================================================================
Toggle Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Toggles                         86        84         2    97.67%

================================Toggle Details================================

Toggle Coverage for instance /FIFO_top/FIFO_inter --
```



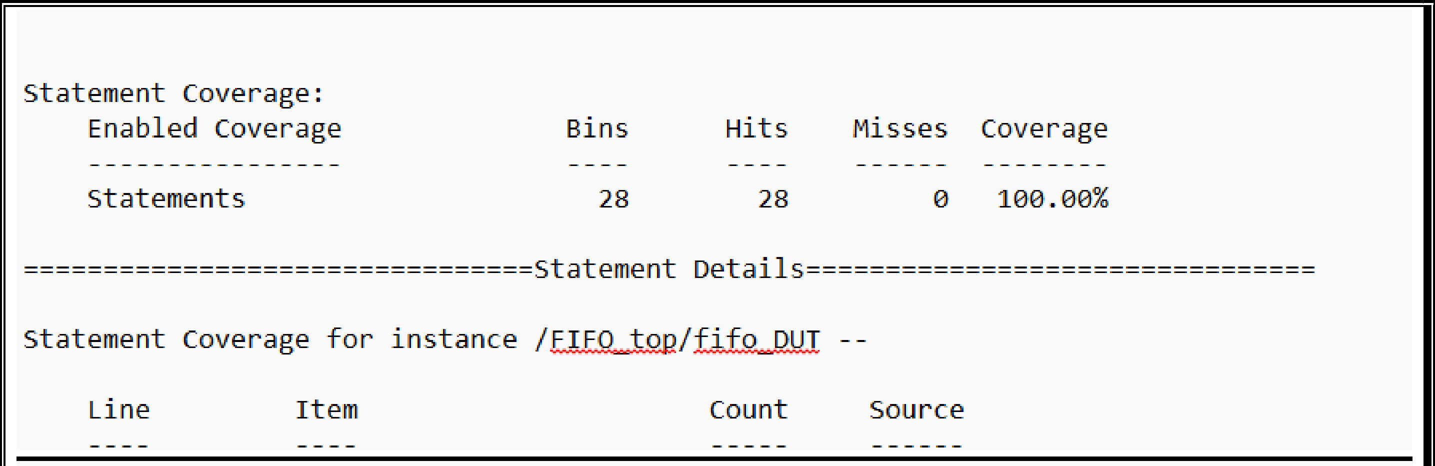
```
=================================================================================
=== Instance: /FIFO_top/fifo_DUT/SVA
=== Design Unit: work.FIFO_SVA
=================================================================================

Assertion Coverage:
    Assertions                       8         8         0   100.00%
--------------------------------------------------------------------
Name                     File(Line)                    Failure      Pass
                                                       Count        Count
--------------------------------------------------------------------
/FIFO_top/fifo_DUT/SVA/assert__7
```



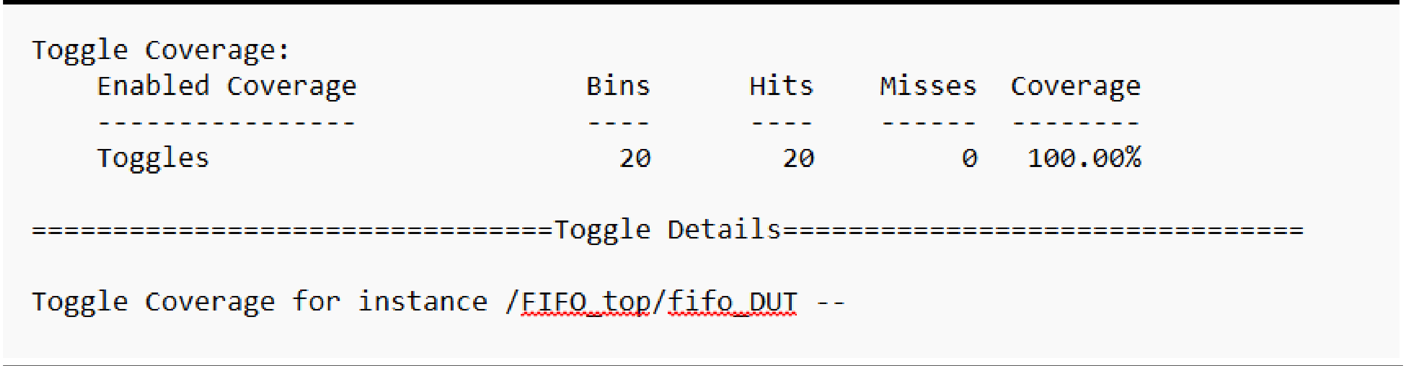
```
=================================================================================
=== Instance: /FIFO_top/fifo_DUT
=== Design Unit: work.FIFO
=================================================================================
Branch Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Branches                        28        27         1    96.42%

================================Branch Details================================

Branch Coverage for instance /FIFO_top/fifo_DUT
```



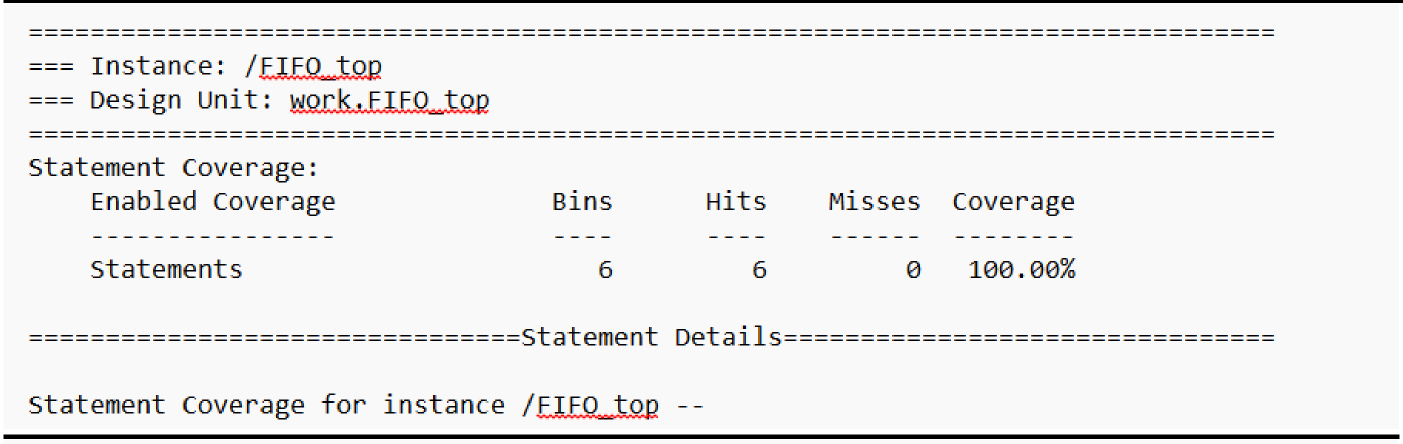
```
Condition Coverage:
    Enabled Coverage              Bins   Covered    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Conditions                      22        18         4    81.81%

================================Condition Details================================

Condition Coverage for instance /FIFO_top/fifo_DUT --

  File FIFO.sv
```



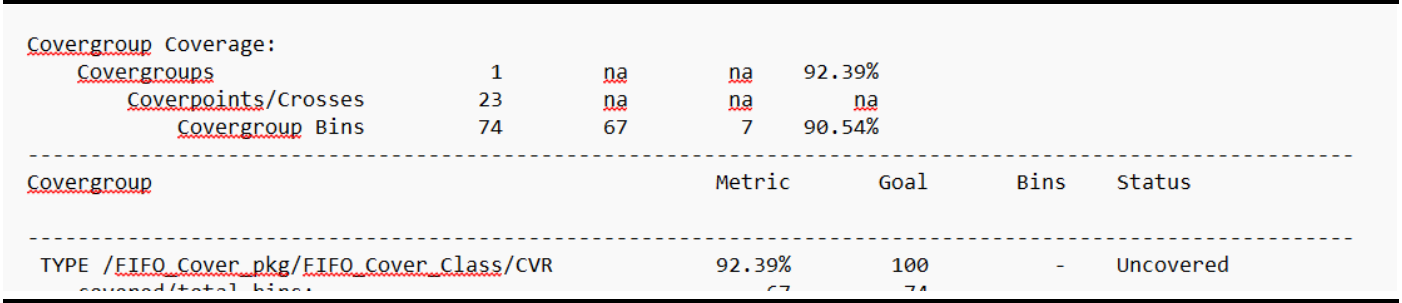
```
Statement Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Statements                      28        28         0   100.00%

================================Statement Details================================

Statement Coverage for instance /FIFO_top/fifo_DUT --

    Line         Item                        Count     Source
    ----         ----                        -----     ------
```



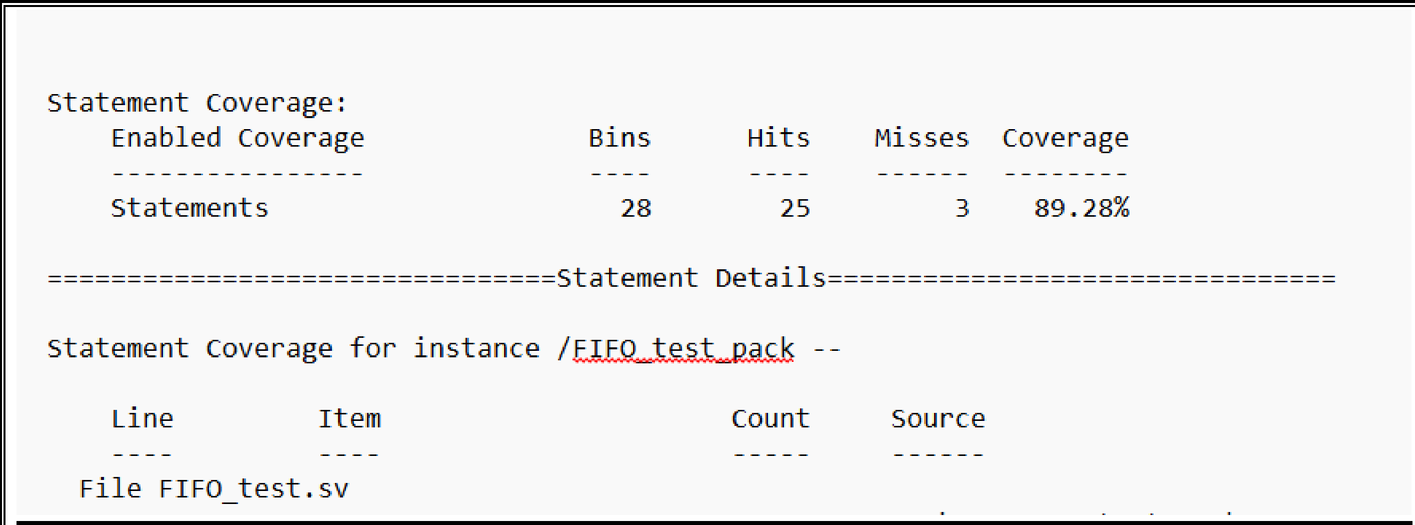
```
Toggle Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Toggles                         20        20         0   100.00%

================================Toggle Details================================

Toggle Coverage for instance /FIFO_top/fifo_DUT --
```



```
=================================================================================
=== Instance: /FIFO_top
=== Design Unit: work.FIFO_top
=================================================================================
Statement Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Statements                       6         6         0   100.00%

================================Statement Details================================

Statement Coverage for instance /FIFO_top --
```



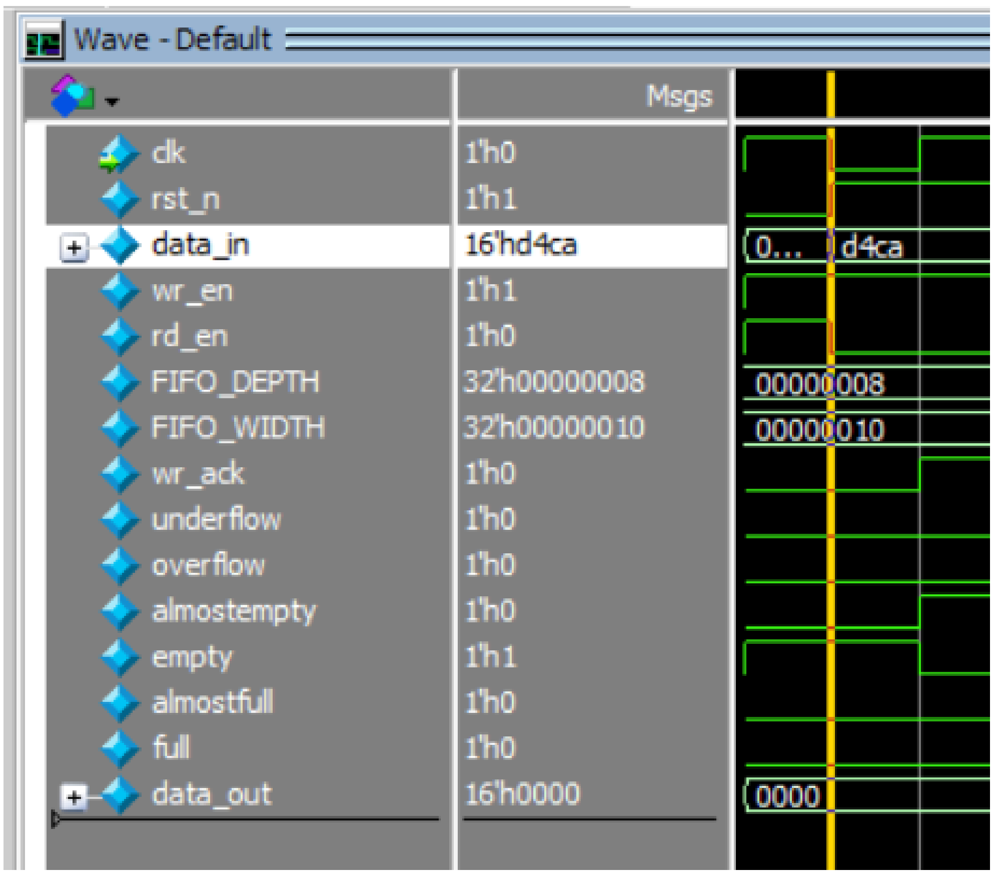
```
Covergroup Coverage:
    Covergroups                      1        na        na    92.39%
        Coverpoints/Crosses         23        na        na        na
            Covergroup Bins         74        67         7    90.54%
----------------------------------------------------------------------------------------------
Covergroup                                             Metric       Goal       Bins    Status

----------------------------------------------------------------------------------------------
 TYPE /FIFO_Cover_pkg/FIFO_Cover_Class/CVR              92.39%        100          -    Uncovered
    covered/total bins:                                    67         74
```



```
Statement Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Statements                      28        25         3    89.28%

================================Statement Details================================

Statement Coverage for instance /FIFO_test_pack --

    Line         Item                       Count     Source
    ----         ----                       -----     ------
  File FIFO_test.sv
```

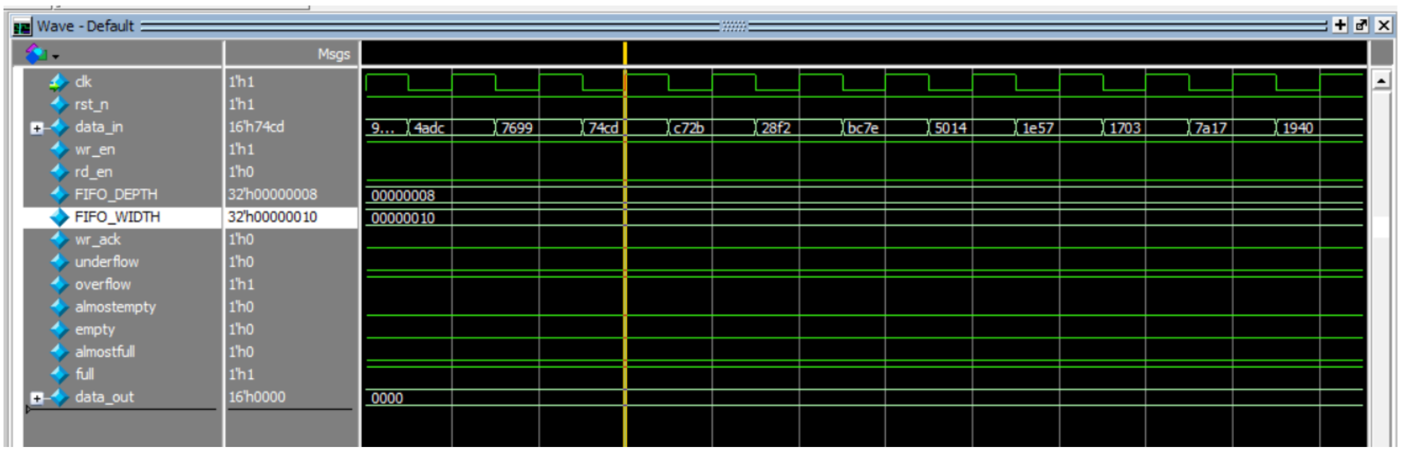
## III):QuestaSim snippets:

### 1-Reset process:

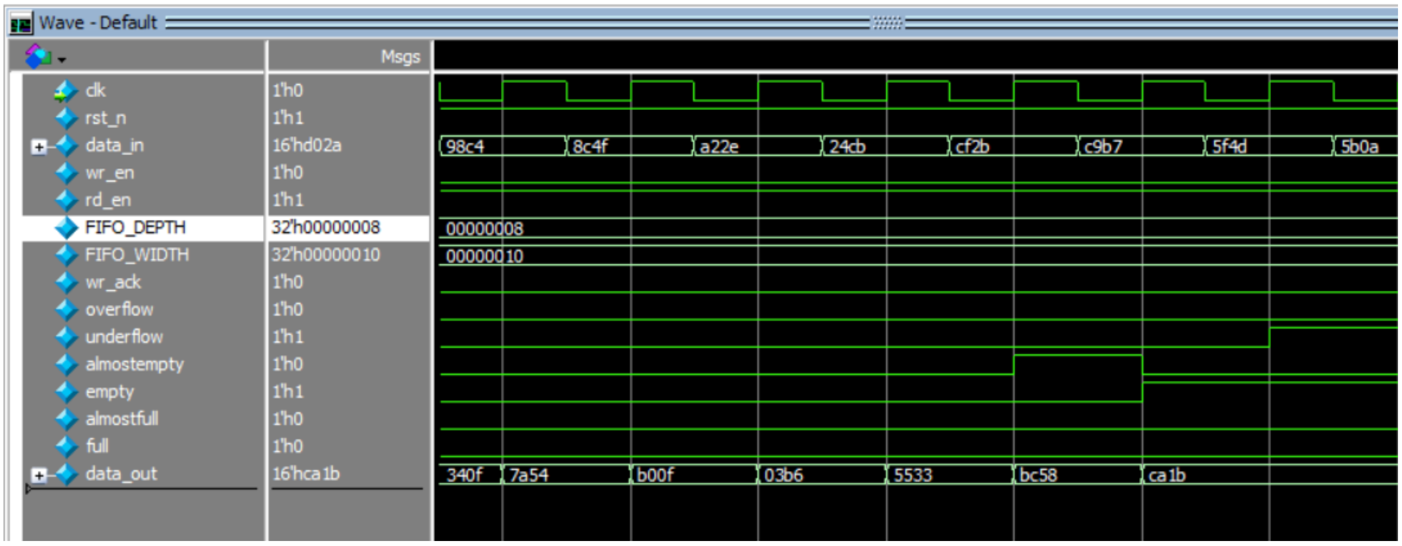


```
Wave - Default
                      Msgs
clk                   1'h0
rst_n                 1'h1
data_in               16'hd4ca         0...  d4ca
wr_en                 1'h1
rd_en                 1'h0
FIFO_DEPTH            32'h00000008     00000008
FIFO_WIDTH            32'h00000010     00000010
wr_ack                1'h0
underflow             1'h0
overflow              1'h0
almostempty           1'h0
empty                 1'h1
almostfull            1'h0
full                  1'h0
data_out              16'h0000         0000
```

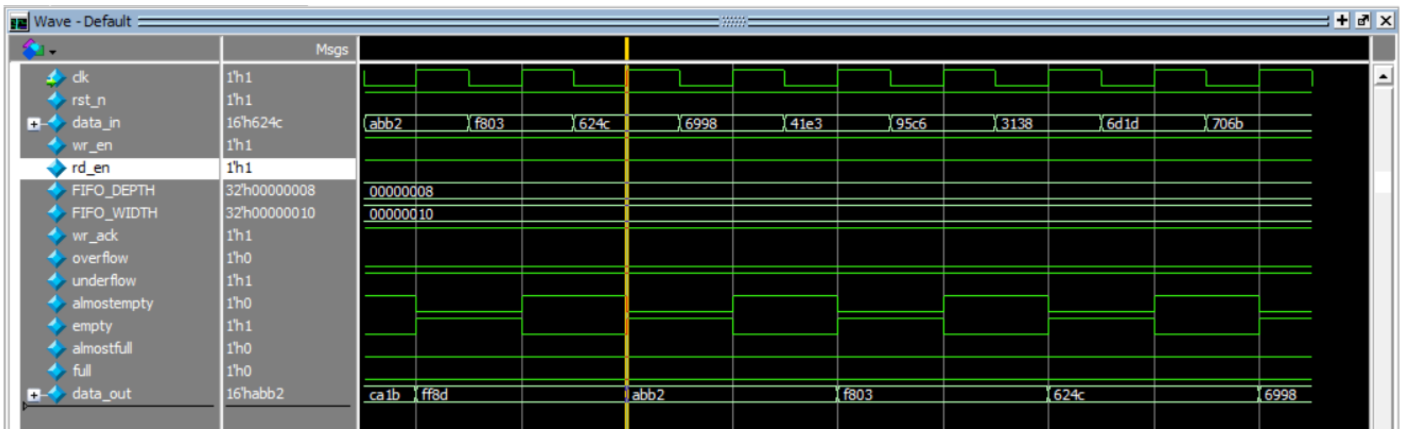
## 2-Write process:



## 3-Read process:



## 4-Write_Read_Process:



## IV)Assertions table:

| Feature | Assertion |
|---|---|
| When the FIFO is full, the full flag must be asserted. | `assert property (@(posedge FIFO_Inter.clk) ((FIFO.count == FIFO.FIFO_DEPTH) \|-> FIFO_Inter.full));`<br>`cover property (@(posedge FIFO_Inter.clk) ((FIFO.count == FIFO.FIFO_DEPTH) \|-> FIFO_Inter.full));` |
| When the FIFO is empty, the empty flag must be asserted. | `assert property (@(posedge FIFO_Inter.clk) ((FIFO.count == 0) \|-> FIFO_Inter.empty));`<br>`cover property (@(posedge FIFO_Inter.clk) ((FIFO.count == 0) \|-> FIFO_Inter.empty));` |
| When the FIFO is almost full, the almostfull flag must be asserted. | `assert property (@(posedge FIFO_Inter.clk) ((FIFO.count == FIFO.FIFO_DEPTH-1) \|-> FIFO_Inter.almostfull));`<br>`cover property (@(posedge FIFO_Inter.clk) ((FIFO.count == FIFO.FIFO_DEPTH-1) \|-> FIFO_Inter.almostfull));` |
| When the FIFO is almost empty, the almostempty flag must be asserted. | `assert property (@(posedge FIFO_Inter.clk) ((FIFO.count == 1) \|-> FIFO_Inter.almostempty));`<br>`cover property (@(posedge FIFO_Inter.clk) ((FIFO.count == 1) \|-> FIFO_Inter.almostempty));` |
| When a write request is introduced while the full flag is asserted, the overflow flag must be asserted. | `assert property (@(posedge FIFO_Inter.clk) ((FIFO_Inter.full & FIFO_Inter.wr_en) \|=> FIFO_Inter.overflow));`<br>`cover property (@(posedge FIFO_Inter.clk) ((FIFO_Inter.full & FIFO_Inter.wr_en) \|=> FIFO_Inter.overflow));` |
| When a read request is introduced while the empty flag is asserted, the underflow flag must be asserted. | `assert property (@(posedge FIFO_Inter.clk) ((FIFO_Inter.empty & FIFO_Inter.rd_en) \|=> FIFO_Inter.underflow));`<br>`cover property (@(posedge FIFO_Inter.clk) ((FIFO_Inter.empty & FIFO_Inter.rd_en) \|=> FIFO_Inter.underflow));` |
| When a write request is introduced while there is empty spaces in the FIFO, the wr_ack flag must be asserted. | `assert property (@(posedge FIFO_Inter.clk) ((FIFO_Inter.wr_en && (FIFO.count < FIFO.FIFO_DEPTH)) \|=> FIFO_Inter.wr_ack));`<br>`cover property (@(posedge FIFO_Inter.clk) ((FIFO_Inter.wr_en && (FIFO.count < FIFO.FIFO_DEPTH)) \|=> FIFO_Inter.wr_ack));` |
| When reset is asserted, the counters: wr_ptr, rd_ptr, and count, must be cleared. | `assert property (@(posedge FIFO_Inter.clk) ((!FIFO_Inter.rst_n) \|=> ((!FIFO.wr_ptr) & (!FIFO.rd_ptr) & (!FIFO.count))));`<br>`cover property (@(posedge FIFO_Inter.clk) ((!FIFO_Inter.rst_n) \|=> ((!FIFO.wr_ptr) & (!FIFO.rd_ptr) & (!FIFO.count))));` |