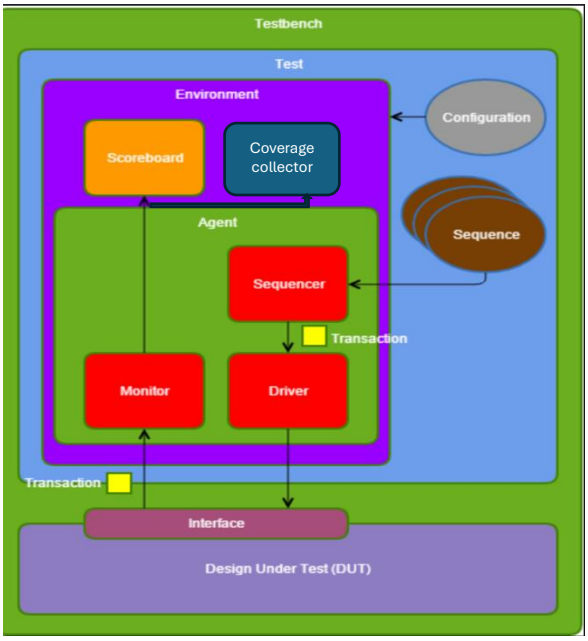
Test plane:

- I use UVM hierarchy to check FIFO design
- I do virtual interface to communicate with DUT as classes do not have ports
- I made sequences that are intended to be sent to the DUT
- Sequences delivering multiple sequence items through the sequencer to the driver
- Test start the sequence, build the env and passes the virtual interface
- Env build agent, scoreboard and coverage it also connects scoreboard and coverage with agent
- Agent builds sequencer, driver and monitor it also connect sequencer with driver and monitor with agent and virtual interface for driver and monitor
- Driver drives the interface signals
- Monitor monitors the activity of the interface and translate the pin level signals into sequence items to broadcast them to scoreboard and coverage
- Coverage checks on the functional coverage of the DUT
- Scoreboard checks for the DUT outputs



```
================================================================================
Branch Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Branches                        24        24         0   100.00%

================================Branch Details================================
```

```
Statement Coverage:
    Enabled Coverage              Bins      Hits    Misses  Coverage
    ----------------              ----      ----    ------  --------
    Statements                      25        25         0   100.00%

================================Statement Details================================
```

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

================================Toggle Details================================
```

```
TOTAL COVERGROUP COVERAGE: 98.43%  COVERGROUP TYPES: 1

DIRECTIVE COVERAGE:
```

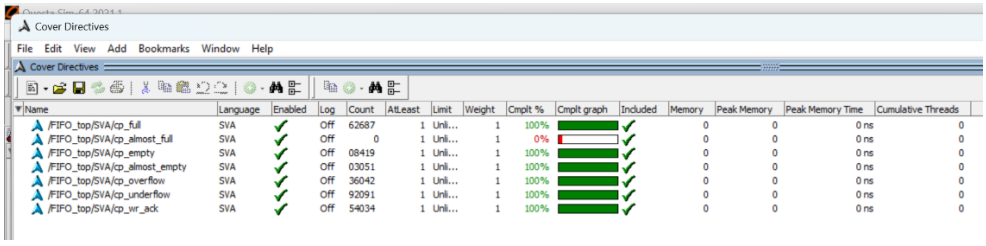
The coverage isn't 100% Bec. Cross coverage between wr_en , rd_en and full when wr is 0 or 1 and rd is 1 and full is 1 happen only at negedge of the clk

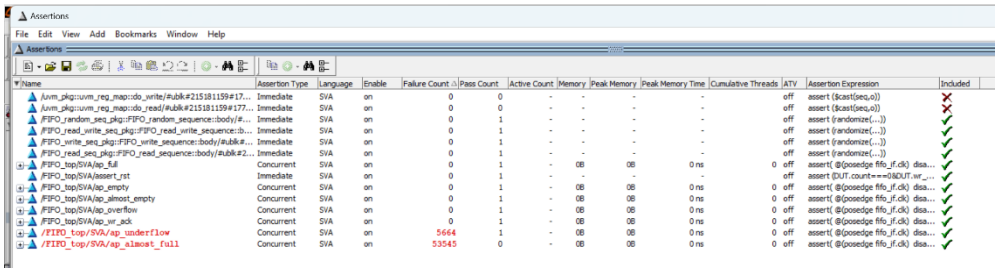
```
Assertion Coverage:
    Assertions                       8         8         0   100.00%
--------------------------------------------------------------------
Name                 File(Line)                   Failure      Pass
                                                  Count        Count
--------------------------------------------------------------------
/FIFO_top/DUT/inst/ap_full
                     FIFO_sva.sv(4)                     0          1
/FIFO_top/DUT/inst/ap_almost_full
                     FIFO_sva.sv(11)                    0          1
/FIFO_top/DUT/inst/ap_empty
                     FIFO_sva.sv(17)                    0          1
/FIFO_top/DUT/inst/ap_almost_empty
                     FIFO_sva.sv(23)                    0          1
/FIFO_top/DUT/inst/ap_overflow
                     FIFO_sva.sv(29)                    0          1
/FIFO_top/DUT/inst/ap_underflow
                     FIFO_sva.sv(35)                    0          1
/FIFO_top/DUT/inst/ap_wr_ack
                     FIFO_sva.sv(41)                    0          1
/FIFO_top/DUT/inst/assert_rst
                     FIFO_sva.sv(48)                    0          1

Directive Coverage:
    Directives                       7         7         0   100.00%

DIRECTIVE COVERAGE:
--------------------------------------------------------------------------------------------------------
Name                                     Design Design   Lang File(Line)      Hits Status
                                         Unit   UnitType
--------------------------------------------------------------------------------------------------------
/FIFO_top/DUT/inst/cp_full               FIFO_sva Verilog  SVA  FIFO_sva.sv(6)  122676 Covered
/FIFO_top/DUT/inst/cp_almost_full        FIFO_sva Verilog  SVA  FIFO_sva.sv(13) 27540 Covered
/FIFO_top/DUT/inst/cp_empty              FIFO_sva Verilog  SVA  FIFO_sva.sv(19) 105192 Covered
/FIFO_top/DUT/inst/cp_almost_empty       FIFO_sva Verilog  SVA  FIFO_sva.sv(25) 102566 Covered
/FIFO_top/DUT/inst/cp_overflow           FIFO_sva Verilog  SVA  FIFO_sva.sv(31) 108499 Covered
/FIFO_top/DUT/inst/cp_underflow          FIFO_sva Verilog  SVA  FIFO_sva.sv(37) 98864 Covered
/FIFO_top/DUT/inst/cp_wr_ack             FIFO_sva Verilog  SVA  FIFO_sva.sv(43) 133514 Covered
```

Before correcting the errors:

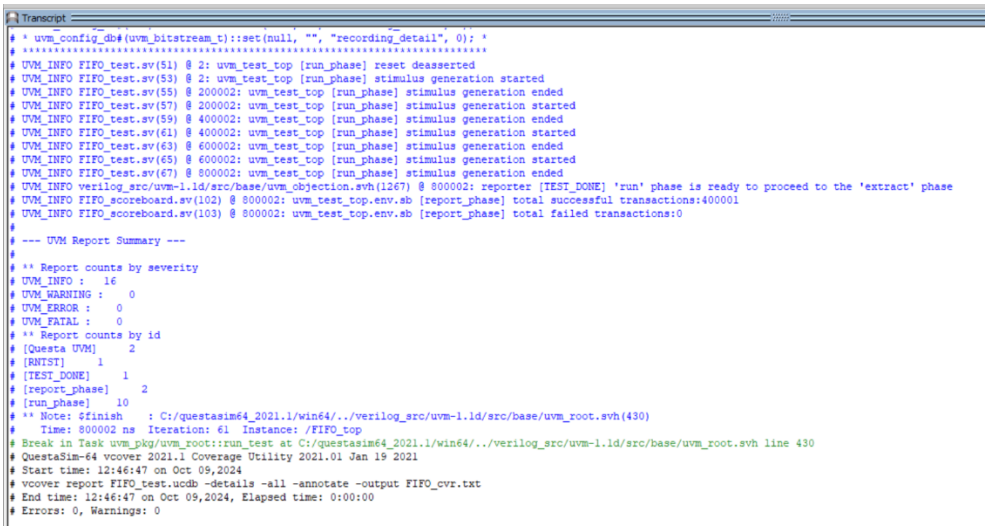


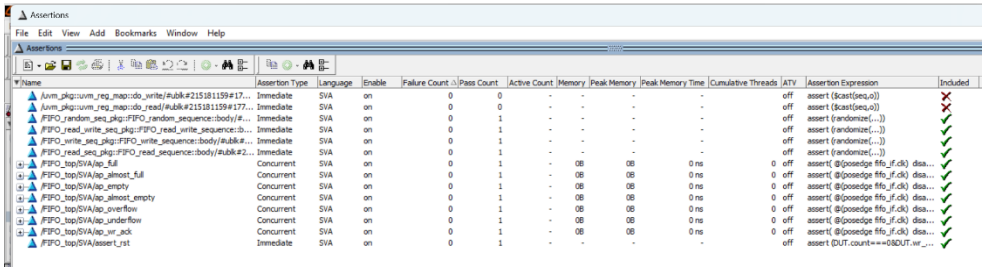


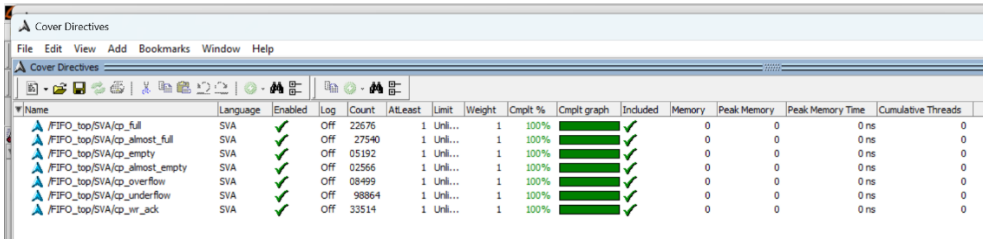
The errors was Bec. Of

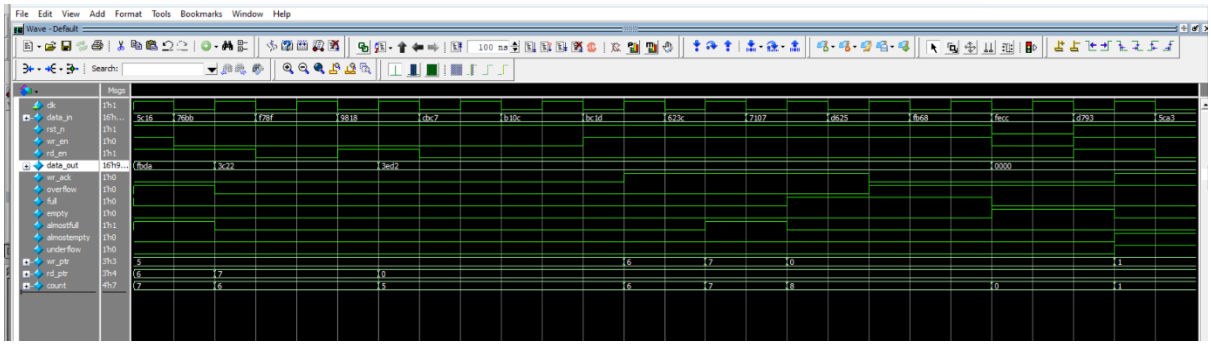
- Almost full signal was implemented logically in correct
- Underflow signal was combinational and the spec want it sequential output
- The data out when reset asserted doesn't take zero
- The count signal not handle all its case

After correcting the errors:









feature	assertion
When counter=fifo depth, full=1	(@(posedge fifo_if.clk) disable iff(!fifo_if.rst_n) DUT.count===fifo_if.FIFO_DEPTH |-> fifo_if.full);
When counter=fifo depth-1 ,almost full=1	(@(posedge fifo_if.clk) disable iff(!fifo_if.rst_n) DUT.count===(fifo_if.FIFO_DEPTH-1) |-> fifo_if.almostfull);
When counter=0, empty signal=1	(@(posedge fifo_if.clk) disable iff(!fifo_if.rst_n) DUT.count===0 |-> fifo_if.empty);
When counter=1, almost empty=1	(@(posedge fifo_if.clk) disable iff(!fifo_if.rst_n) DUT.count===1 |-> fifo_if.almostempty);
When full=1 and write enable =1, overflow=1	(@(posedge fifo_if.clk) disable iff(!fifo_if.rst_n) fifo_if.full&&fifo_if.wr_en |=> fifo_if.overflow);
When empty=1 and read enable=1, underflow=1	(@(posedge fifo_if.clk) disable iff(!fifo_if.rst_n) fifo_if.empty&&fifo_if.rd_en |=> fifo_if.underflow);
When write pointer value increased by one , wr_ack=1	(@(posedge fifo_if.clk) disable iff(!fifo_if.rst_n) DUT.wr_ptr===$past(DUT.wr_ptr)+1 |-> fifo_if.wr_ack);
When reset asserted then counter , write pointer and read pointer =0	assert_rst: assert final(DUT.count===0 && DUT.wr_ptr===0 && DUT.rd_ptr===0);