# Synchronous FIFO test-plan

| CHECKS | STATUS | NO. BUGS FOUND |
|---|---|---|
| a) **Asserting reset** and checking the output using **reset_test** | **DONE** | **FAILED (3)** |
| b) **Constraining data_in to reach MAX and MIN values if randomized data_in_c** | **DONE** | **PASSED** |
| c) **Create a queue *data_to_write_queue* to mirror the FIFO's pointers movements** which is used to compare the read values later on | **DONE** | **N/A** |
| d) **The mirrored pointers are used to determine the state of the FIFO's flags** | **DONE** | **N/A** |
| e) **Testing out the limit of the FIFO by writing until its overflowed** while monitoring The flags using **write_all_test/reset_write_read_all_test** | **DONE** | **FAILED (2)** |
| f) **Testing out the limit of the FIFO by reading until its underflowed** while monitoring the flags **reset_write_read_all_test** | **DONE** | **FAILED (2)** |
| g) **Randomizing write and read operations/tests while monitoring the flags write_read_rand_test** | **DONE** | **ALL OF THE ABOVE** |

## Coverage Groups

- **FLAGS_covgrp: Covering the write and read and reset operations.**
- **OPERATION_covgrp: Covering all the flags' data frames and data transitions.**

## BUG REPORT

| EXPECTED | DETECTED |
|---|---|
| When the rst_n is asserted:<br>• data_out = 0<br>• All the flags = 0 | When the Rst_n is asserted<br>• **Data_out = x;**<br>• All the Flags = 0 **except overflow & wr_ack** |
| **Almostfull not asserted when:**<br>• FIFO is written upto (FIFO_SIZE-2) | **Almostfull asserted when:**<br>• FIFO is written up to **(FIFO_SIZE-2)** |
| **Almostfull asserted when:**<br>• FIFO is written upto (FIFO_SIZE-1) | **Almostfull not asserted when:**<br>• FIFO is written up to **(FIFO_SIZE-1)** |
| **Underflow asserted when:**<br>• READ operation done AFTER FIFO was EMPTY | **Underflow asserted when**:<br>• On the same cycle that the FIFO just became EMPTY |

## Assertions REPORT

| Feature | Assertion |
|---|---|
| Whenever the full signal is high and wr_en is | @(posedge clk_sva) full_sva && wr_en_sva \|=> overflow_sva; |
| Whenever the empty signal is high and rd_en is asserted underflow should be high | @(posedge clk_sva) empty_sva && rd_en_sva \|=> underflow_sva |
| Whenever the full signal is low and wr_en is asserted wr_ack should be high | @(posedge clk_sva) (!full_sva && wr_en_sva \|=> wr_ack_sva); |
| Whenever the full signal is low wr_ack should be always low | @(posedge clk_sva) (full_sva \|=> !wr_ack_sva); |
| Whenever the almostfull signal is high and wr_en is asserted then FIFO is expected to be full next cycle | @(posedge clk_sva) (almostfull_sva && wr_en_sva \|=> full_sva); |
| Whenever the almostempty signal is high and rd_en is asserted then FIFO is expected to be empty next cycle | @(posedge clk_sva) (almostempty_sva && rd_en_sva \|=> empty_sva); |

**P.S. Check the coverage reports.**