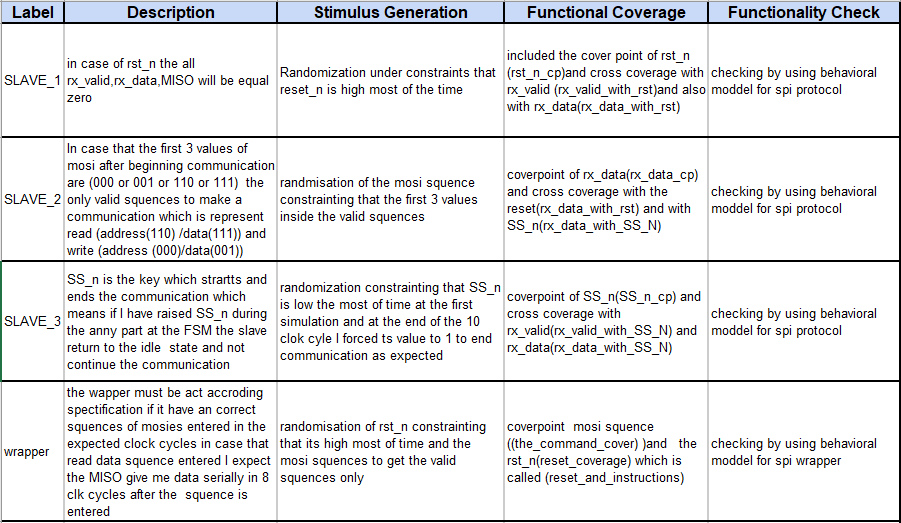
Final Project

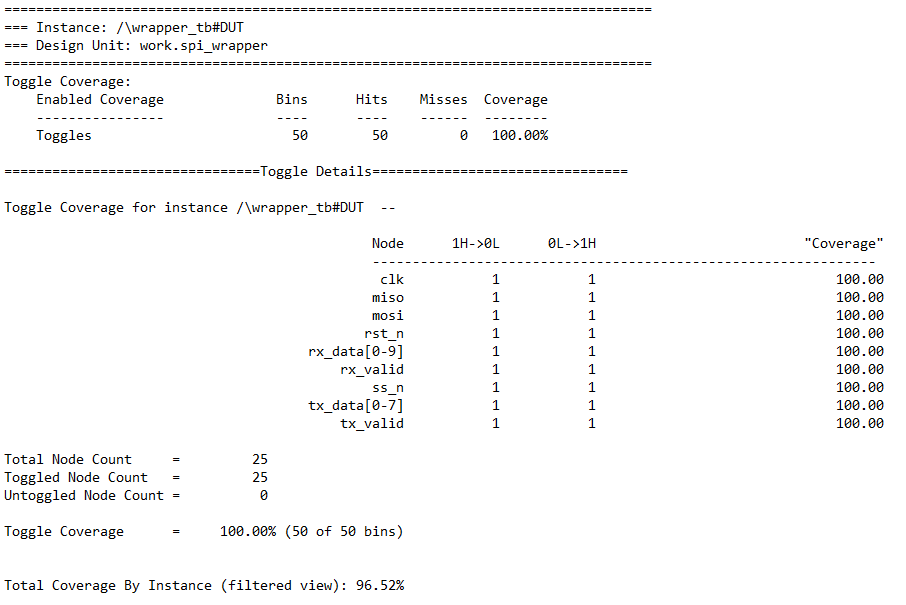
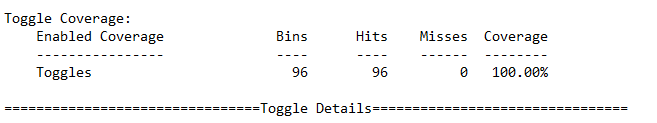
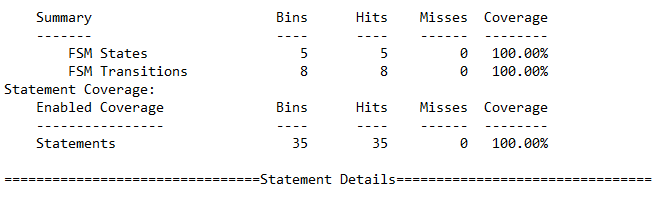
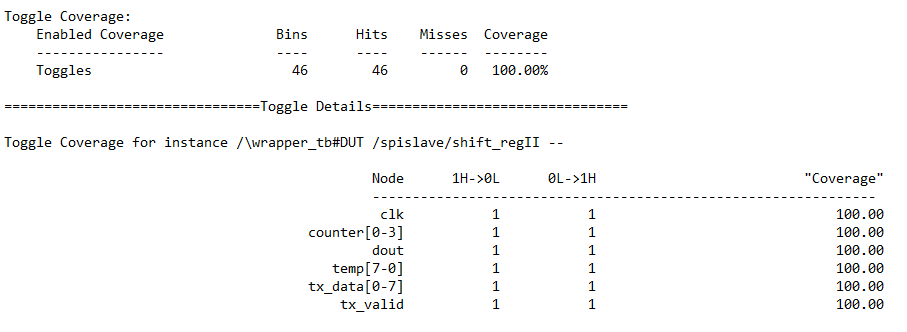
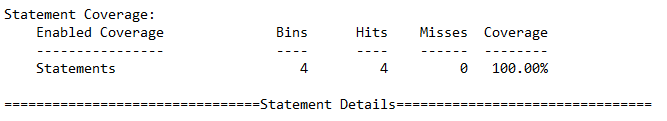
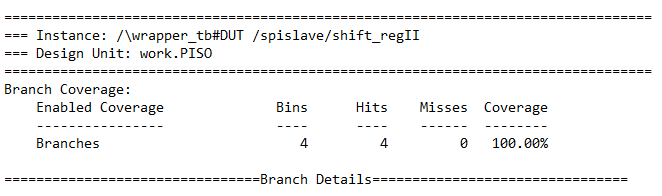
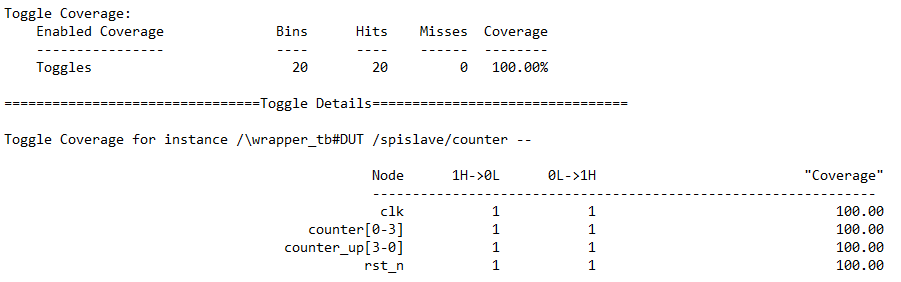
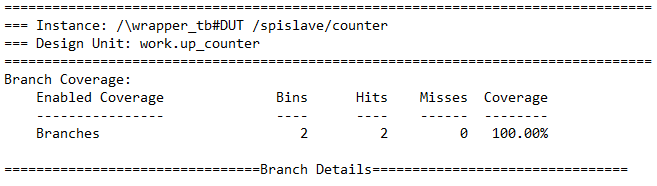
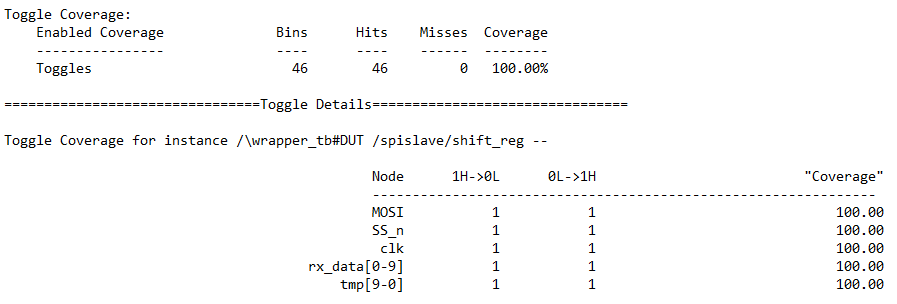
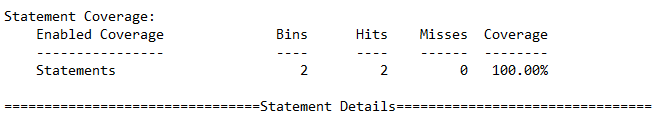
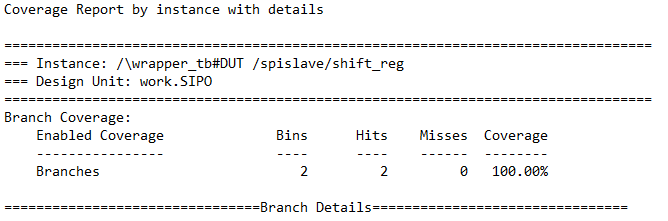
Project (3)-SPI-Wrapper:

1. Verification plan
2. Reset functionality should clear the flag, counter, rx\_valid and MISO.
3. Check Starting communication phase when SS\_n gets low.
4. Testing next state choice after current state is check command according to flag & MOSI.
5. Verify the serial to parallel conversion operation after 10 clock cycles after its start.
6. Checking the address is updated after the rx\_valid is asserted in WRITE and READ\_ADD cases.
7. Verifying the Read\_data process.
8. Check End communication transition when master makes SS\_n high.

**Note:**

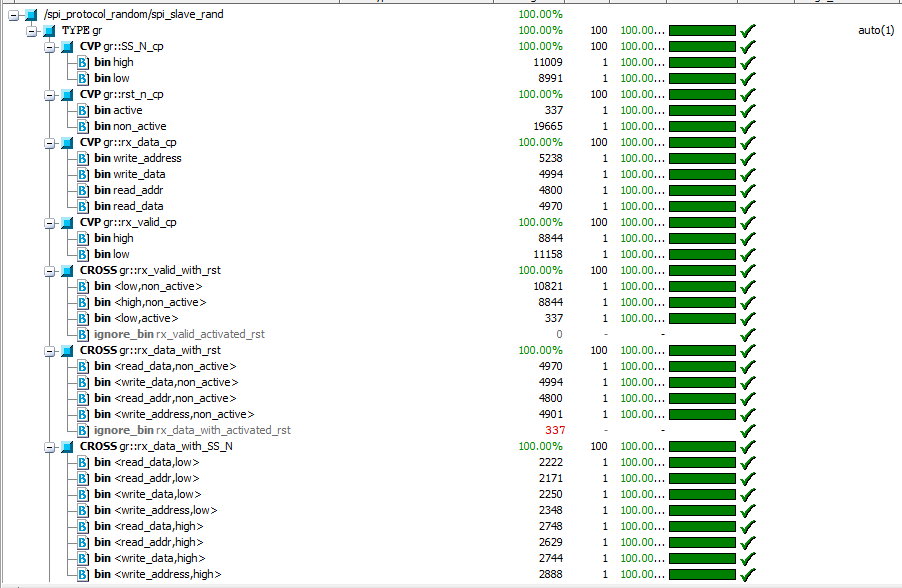
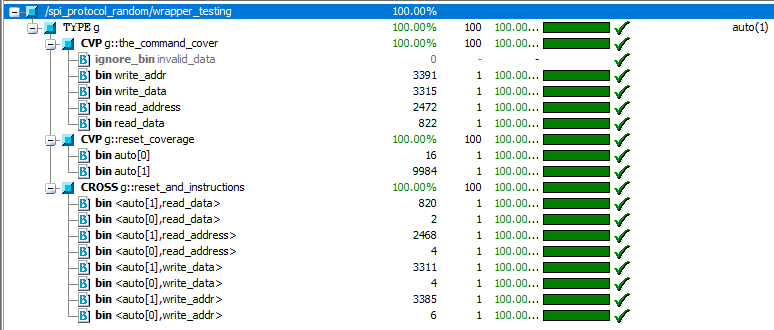
* In Constraining and debugging the possibility of the master delaying the end communication signal (SS\_n) was not taken in consideration for example if the master is writing in the RAM the high SS\_n signal will come right after 11 clock cycles from starting the communication.
* In this DUT: dividing the inter tasks (counter, PISO, SIPO) in the SPI module resulted in inevitable delays that are not specified in the specs to correct these delays we would have to tear down the whole design. So, we just test for basic functionality neglecting timing specially in READ\_DATA case as it clearly has a problem.
* We adjusted the testbench to match the design delays in order to check for the delayed output each clock cycle.

1. Verification Document
2. Coverage reports:



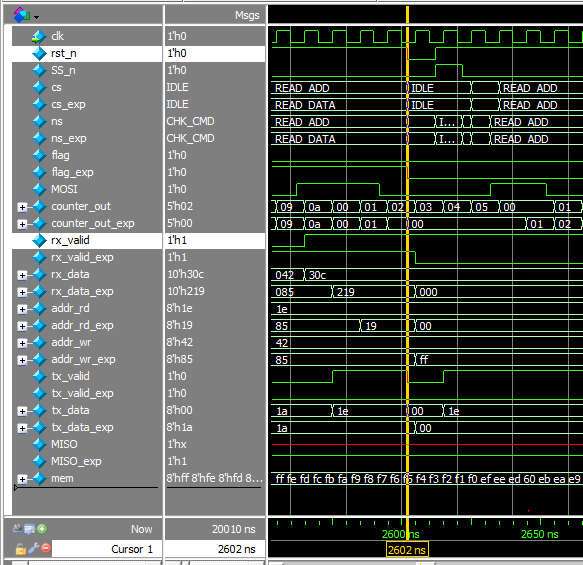
Note: Coverage didn’t reach 100% because of conditional coverage the all-false term.

* Function Coverage:



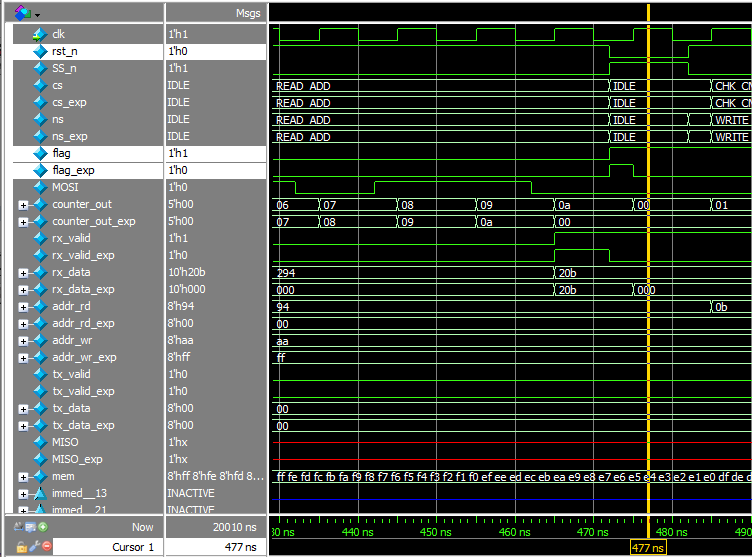
1. Bugs report:
2. Timing delays in the SPI wrapper operations specially in READ\_DATA case.
3. In case of active reset rx\_valid does not change (remains the same).

Snippet shows the stability of the rx\_valid flag even during active reset



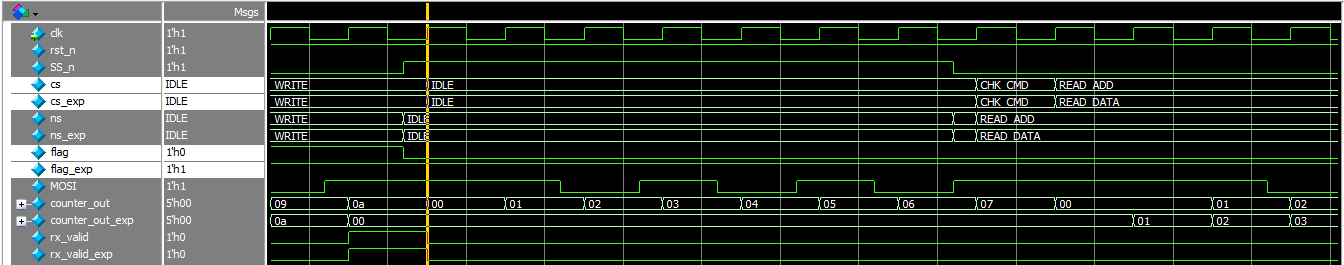
Snippet shows the false positive of the almost flag

1. During asserted reset the flag which indicates that an address was sent does not get cleared.



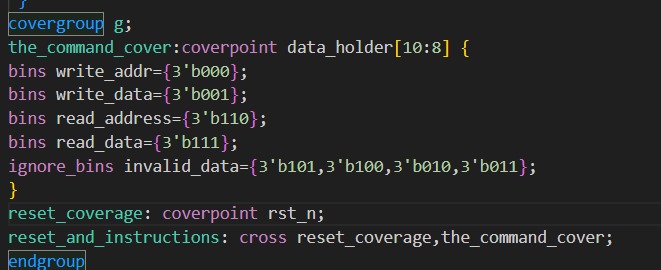
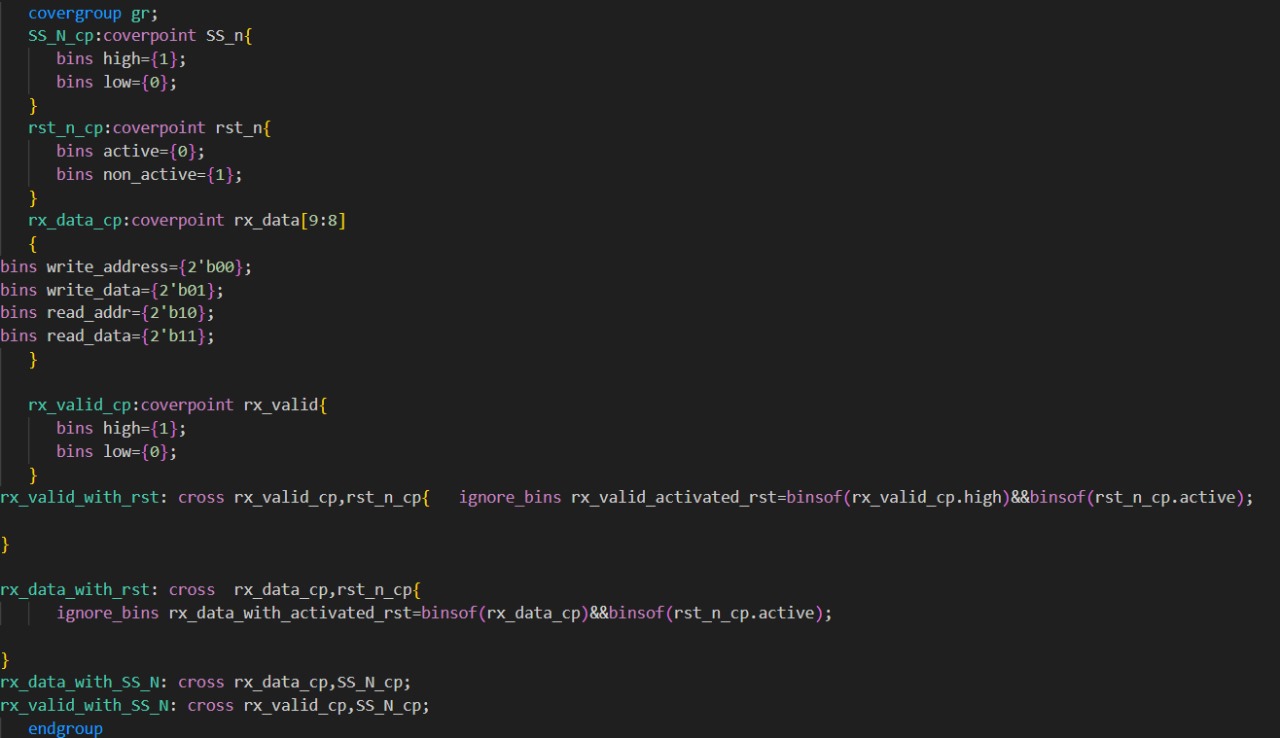
Snippet shows the stability of the address flag even during active reset

1. If the current state is WRITE and master ends the communication the flag is cleared. Which causes a problem in the next READ operation induces wrong READ\_ADD operation.



Snippet shows the false clear of the flag after WRITE operation

1. Checking to clear the counter in the design takes place on the current state which makes the counter clears but the next clock cycle instead it should check for the next state if it is equal CHK\_CMD to clear the counter right away and be able to assert rx\_valid and access memory before the communication ends.
2. Code snippets
3. Coverage & cover points:



1. Assertions file:

