SV Randomization & Functional Coverage Assignment - Extra

Q1. Write a module to test queue data type and its predefined methods. Run Questasim to make sure the display statements are working as expected.

- Declare int j and a queue q of type int
- initialize int j as 1 and queue q as (0, 2, 5)
- insert int j at index 1 in queue q and display q
- delete index 1 element from queue q and display q
- push an element (7) in the front in queue q and display q
- push an element (9) at the back in queue q and display q
- pop an element from back of queue q into j, display q, and j
- pop an element from front of queue q into j, display q, and j
- reverse, sort, reverse sort and shuffle the queue and display q after using each method

Q2. You are asked to work on the adder module done in the first session. Adjust the testbench using the appropriate data type to the following:

1. Create a package that have a user defined enum that takes the value MAXPOS, ZERO, and MAXNEG.
2. Create a class to randomize the design inputs under the following constraints
    - Reset to be asserted with a low probability that you decide
    - Constrain the adder inputs to (A, B) have the MAXPOS, ZERO and MAXNEG values more often than the remaining values
3. Functional coverage model in the class
    - 2 identical Covergroups for ports A and B (Covgrp_A and Convgrp_B)
    - Each covergroup has 2 coverpoints
    - First coverpoint will cover the following bins

Bins	Values
data_0	0
data_max	MAXPOS
data_min	MAXNEG
data_default	Remaining values

    - Second coverpoint will cover the following bins

Bins	Values
data_0max	Transition from 0 to MAXPOS
data_maxmin	Transition from MAXPOS to MAXNEG
data_minmax	Transition from MAXNEG to MAXPOS

Kareem Waseem

- Use sample task to sample A and B values. Don't sample the input values when the reset is asserted

Check the code coverage and functional coverage reports and modify the testbench as necessary to achieve 100% code coverage and functional coverage.

Use a do file to compile the package, design and testbench then simulate and save the coverage. Finally generate the code coverage report.

Q3.

Add the following functional coverage inside of the class you created in Q3 of assignment 2 extra

- coverpoint that checks for the transition of x from 0 to 1 to 0 to make sure that your testbench is hitting the required pattern frequently.

Generate a functional and code coverage report. Make sure that the functional coverage as well as **statements, branch, toggle and FSM coverage** are 100%. Provide justifications in case you were not able to reach the 100% coverage.

Deliverables:

One PDF file having the following

1. Testbench code
2. Package code
3. Design code
4. Report any bugs detected in the design and fix them
5. Snippet to your verification plan document
6. Do file
7. Code Coverage report snippets
8. Functional Coverage report snippets
9. **Clear and neat** QuestaSim waveform snippets showing the functionality of the design
10. Your PDF file must have this format <your_name>_Assignment3_extra for example Kareem_Waseem_Assignment3_extra

Kareem Waseem