Assignment 2

Q1 : Write SystemVerilog code to :

1. Create a fixed array my_array of type integer and size 10 with default value 5 and display
2. Fill the array by 0, 2, 4, ….. and so on, using for-each loop and display
3. Replace elements from index 1 to 5 by 7 (like this: {0,7,7,7,7,7,12,…..}) and display
4. Create a dynamic array dyn_arr of type int, insert the last 6 elements of my_array in it then display
5. Print the sum of elements more than 3 and the product of elements more than 4
6. Print the max , min , unique, and unique indexes of dyn_arr.

Q2 : Write SystemVerilog code to :

1. Create a fixed array arr1 with 16 packed and 8 unpacked dimensions
2. Create a fixed array arr2 with 8 unpacked dimension and integer data type
3. Fill arr1 by index (arr1[i] = i) and arr2 by double of index (arr2[i] = 2i) and display
4. How do these 2 arrays take place in memory, Draw.
5. Create a dynamic array dyn and copy data from arr1 to it , show which dimension should be dynamic, and display

Q3: Your task is to develop a SystemVerilog testbench that verifies the RAM module using Package and class to randomize all inputs and test all cases. The verification should include:

1. Package (RAM_pkg):
   - Containing a class called (RAM_class):
     a) Randomize all inputs (addr_wr, addr_rd, din, wr_en, rd_en, rst_n).
     b) Constraint on rst_n to be active high most of time.
     c) Constraint on wr_en to be active 60%.
     d) Constraint on rd_en to be active 40%.
     e) Constraint on rd_en and wr_en should not be high in the same time.

f) addr_wr and addr_rd should be within the valid address range.

2. Testbench (RAM_tb):

   - Import package (RAM_pkg).
   - Then create a class object and construct it by new function.
   - Then randomize class object using ($randomize) in for loop for many times to test all cases in read and write cases

3. Coverage Goals:

   - Ensure all key functionalities are exercised.
   - Generate a basic report summarizing the test results.

   Note: You must use tasks in your testbench to improve modularity and code readability. Tasks should be implemented for common operations like writing to and reading from memory, ensuring better structure and reusability in the verification process.

Submission Requirements

One PDF file having same requirements as last assignment

Good luck!