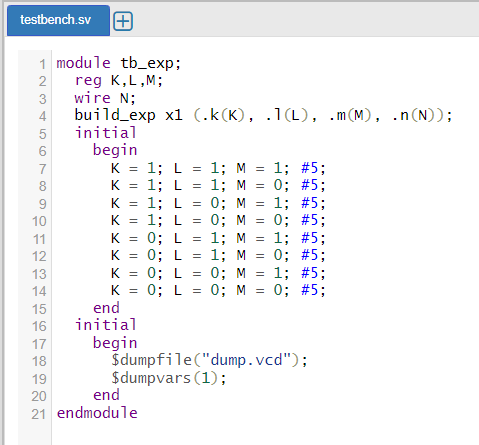
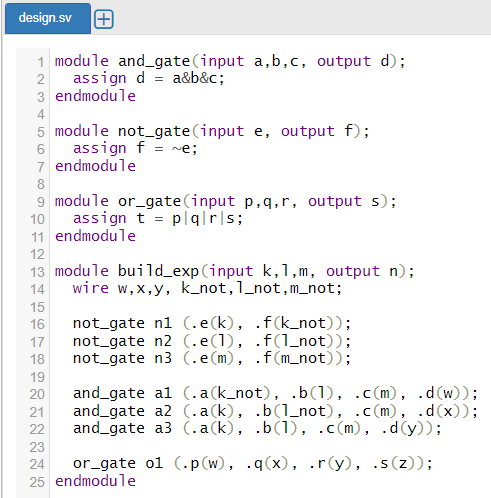
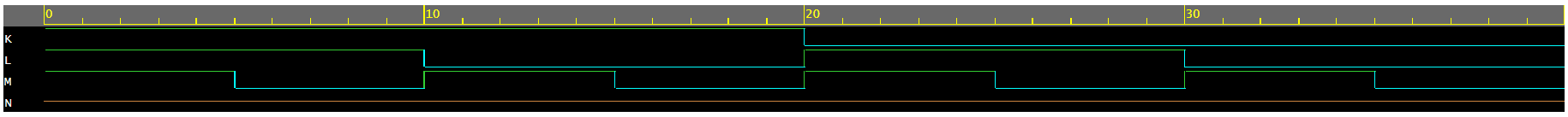
**Lab 3: Implementation of Structural & Behavioral Verilog Code**

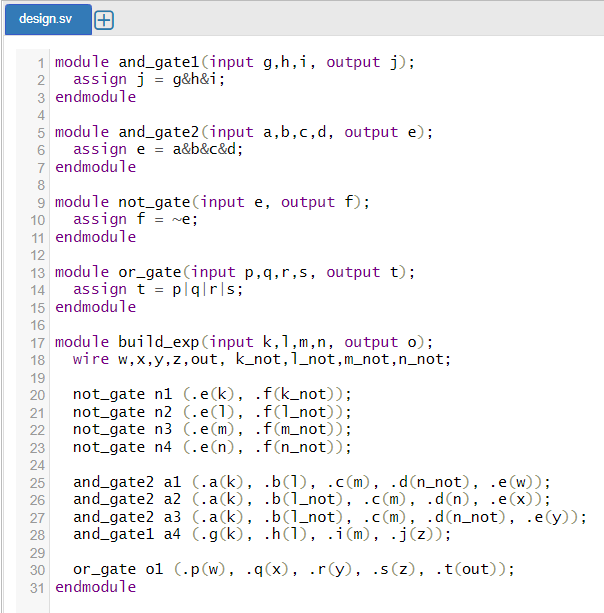
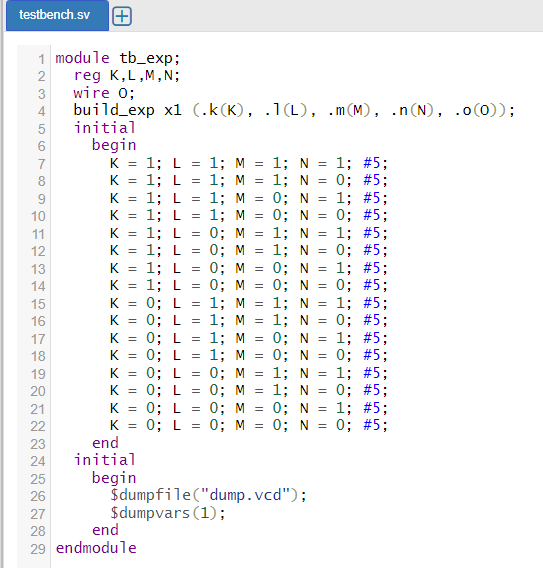
**Question 1: Implement the following expression using the Verilog HDL. Moreover, Verify your circuit against the waveform.**

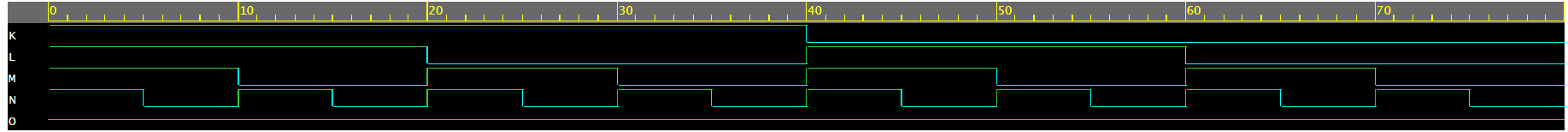
1. **F(A, B, C)= A’BC + AB’C + ABC**

** **

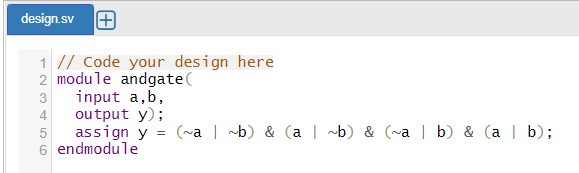
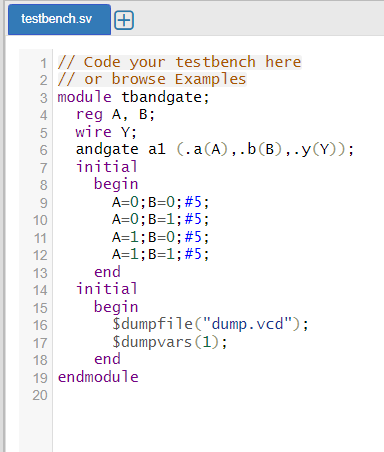
****

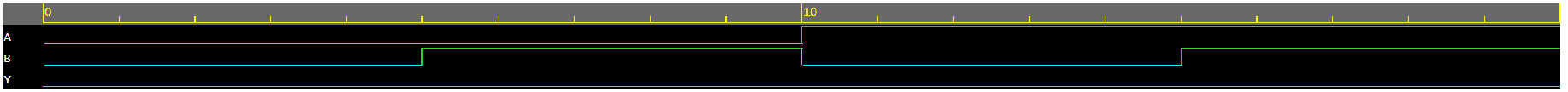
1. **F(A, B, C, D)=ABCD’ + A’BC D+ AB’CD’ + ABC**

****

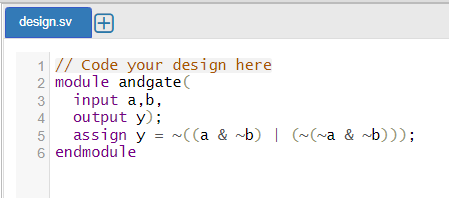
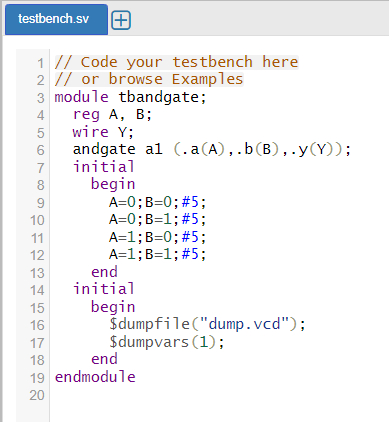
****

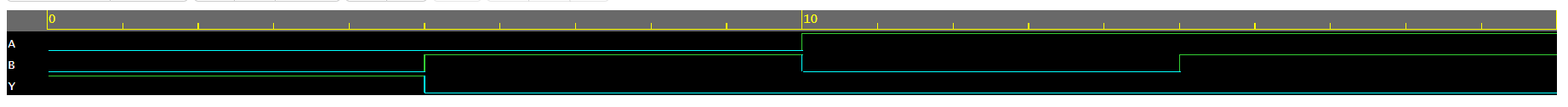
1. **F(A, B) = (A’+B’) (A+B’) (A’+B) (A+B)**

****

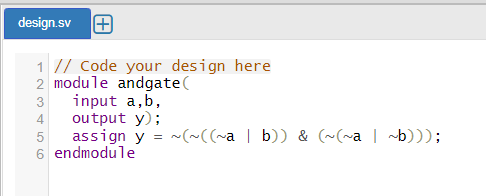
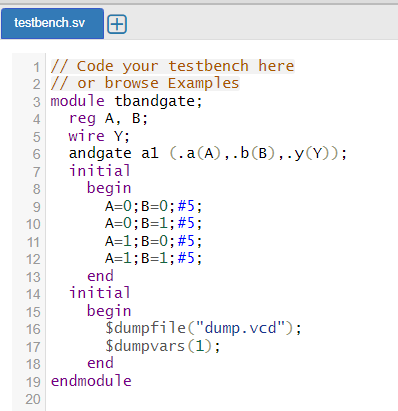
****

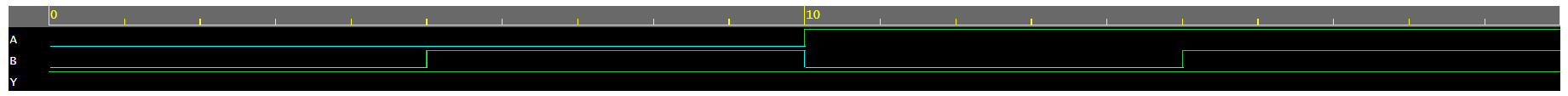
1. **(A, B) = ((A.B’)’+((A)’(B)’)’)’**



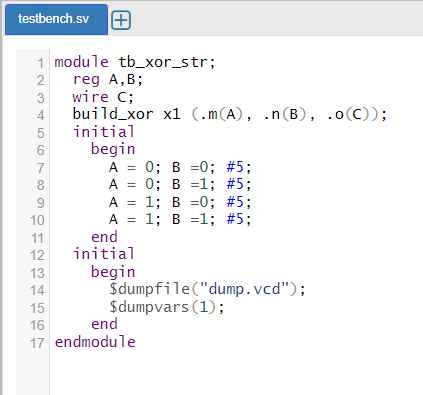
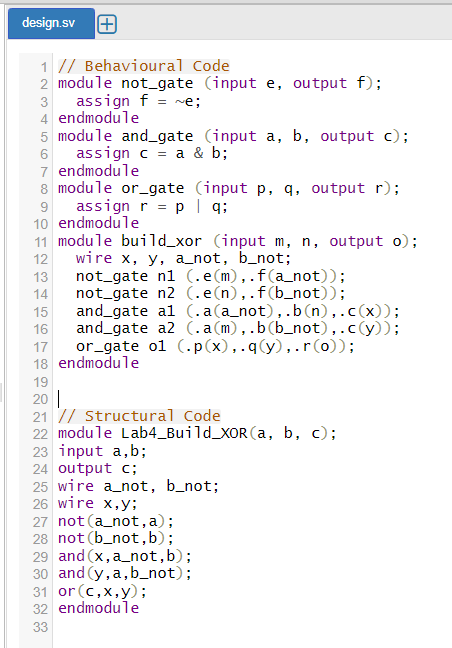


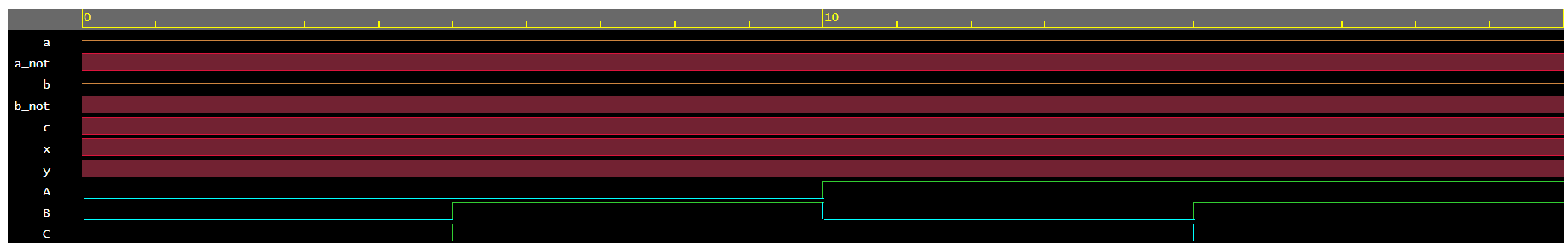
1. **(A, B) = (((A)’+B)’. ((A)’+(B)’)’)’**

****

****

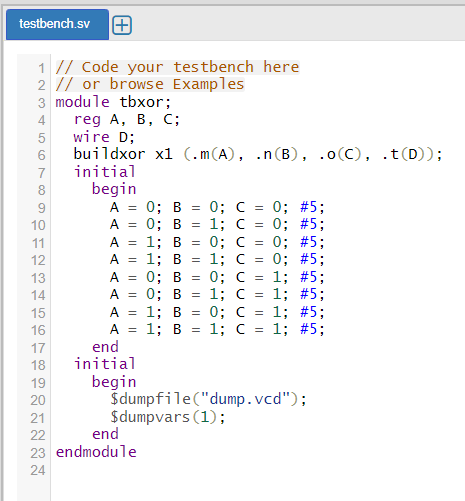
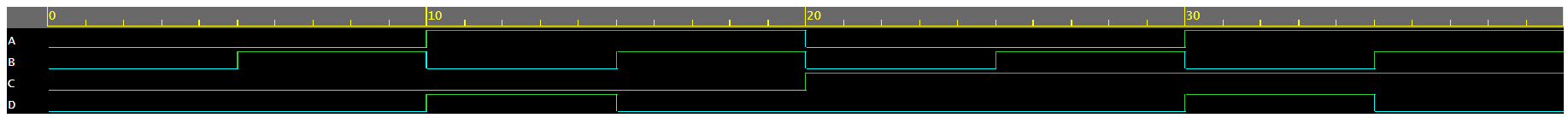
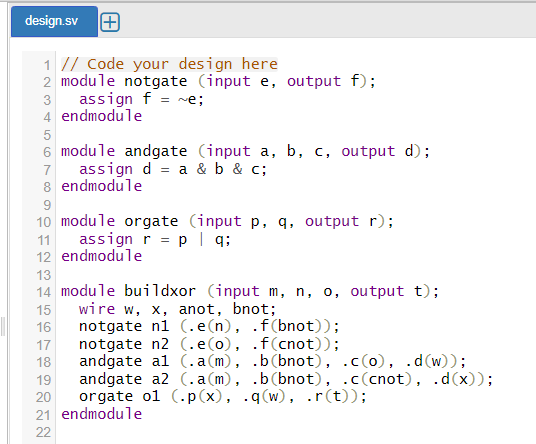
**Question 2: Implement the XOR gate using Behavioral and Structural code of Verilog Hardware Description Language.**

** **

****

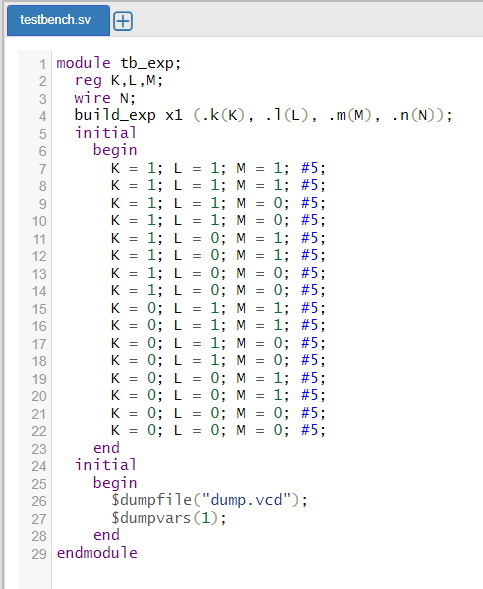
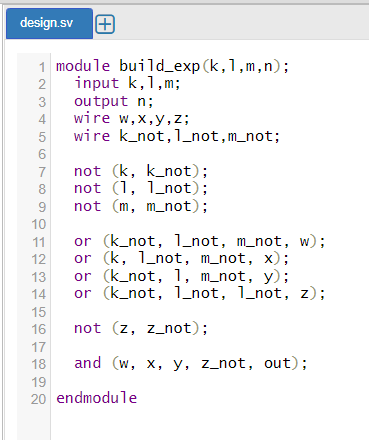
**Question 3: Implement the following expression using the Verilog Hardware Description Language (HDL) (Structural Coding).**

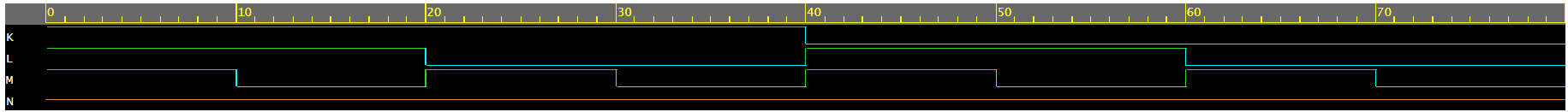
1. **F (A, B, C) = (A’+B’+C’)(A+B’+C’)(A’+B+C’)(A’+B’)`**

** **

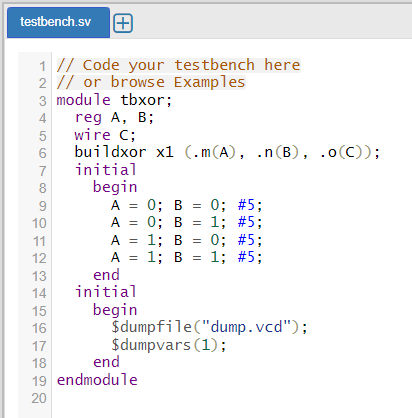
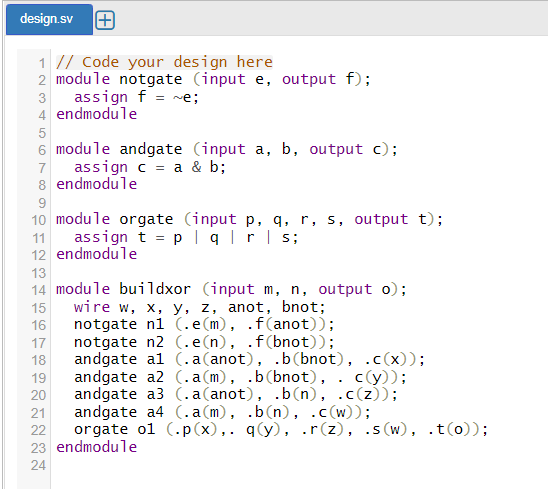
**Question 3: Implement the following expression using universal NAND and NOR gate. Write down Verilog Structural and Behavioral code for that expression.**

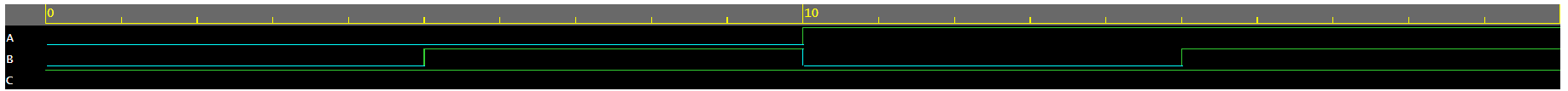
1. **F (A, B, C) = (AB’C) + (AB’C’)**

** **

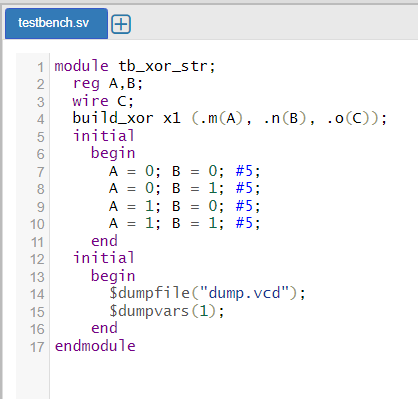
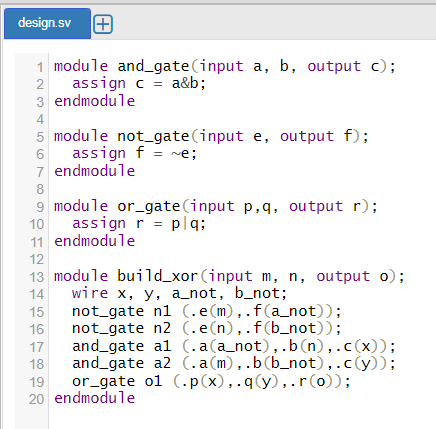
****

1. **F (A, B) = A’B’ + AB’ + A’B + AB**

 ****

****

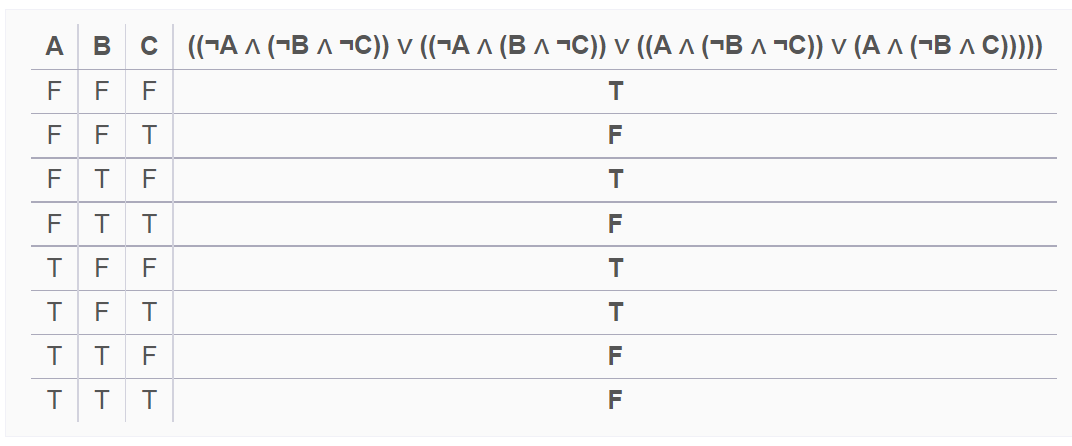
**Question 4 Write down three modules in a single Verilog code to design AND, OR and NOT gate. Now use those modules to design the XOR gate. Use AND, OR and NOT gate as the instances to implement the XOR gate. Write the corresponding Testbench code for the verification of your XOR gate.**

** **

****

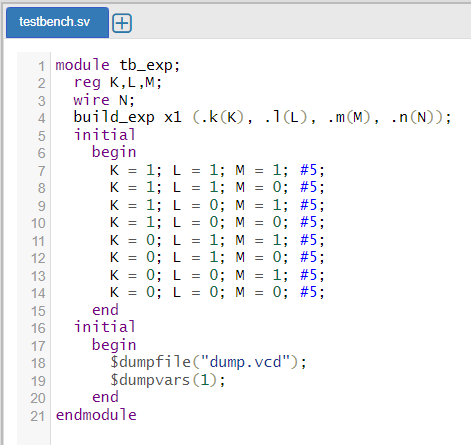
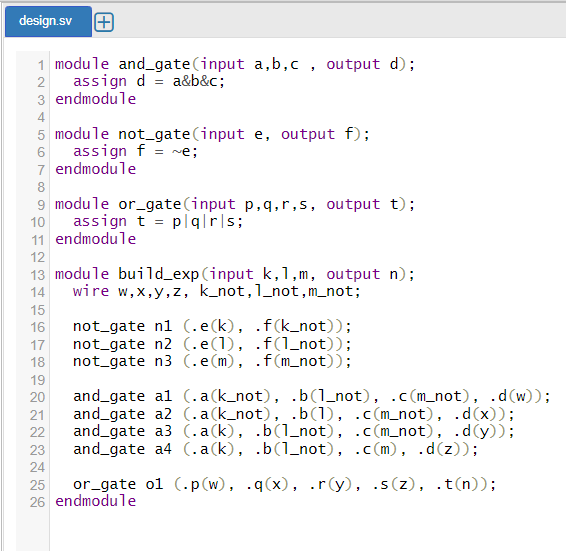
**Question 5 Consider the following expression: Y = A’.B’.C’ + A’.B.C’ + A.B’.C’ + A.B’.C**

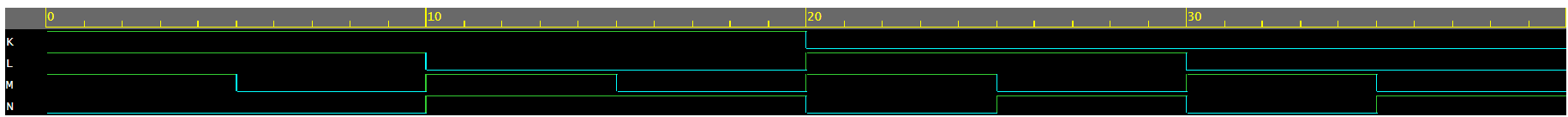
**Design one three input ‘And’ gate module and one four input ‘OR’ gate module using Verilog. Instantiate those two modules to design the above-mentioned expression. Design the corresponding Testbench code for the verification purpose.**

****

**Y**

**Reduced Exression:** A’.B’.C’

** **

****