Laboratory 2 (100 Points)

Assignment 1 (20 Points)

Implement a logic circuit for the two-bit comparator. This comparator has two two-bit inputs A ($A = A_1A_0$) and B ($B = B_1B_0$) and produces one output. The output should be 1 if A is greater than B, otherwise the output should be 0.

a. Construct the truth table for the two-bit comparator
b. Write the sum of product form of the function from the truth table.
c. Simplify the function as much as possible
d. Design a logic diagram for the simplified function
e. Run the simulation of the logic diagram using Simulation Waveform Editor tool.
f. Write a Verilog structural description (gate-entry modeling) for the simplified function
g. Run the simulation of the Verilog code using Simulation Waveform Editor tool.
h. Write a Verilog testbench for the Verilog code to test the model.

a. Construct the truth table for the two-bit comparator

A1	A0	B1	B0	A_greater_than_B	
0	0	0	0	0	
0	0	0	1	0	
0	0	1	0	0	
0	0	1	1	0	
0	1	0	0	1	A > B (A = 1, B = 0)
0	1	0	1	0	
0	1	1	0	0	
0	1	1	1	0	
1	0	0	0	1	A > B (A = 2, B = 0)
1	0	0	1	1	A > B (A = 2, B = 1)
1	0	1	0	0	
1	0	1	1	0	
1	1	0	0	1	A > B (A = 3, B = 0)
1	1	0	1	1	A > B (A = 3, B = 1)
1	1	1	0	1	A > B (A = 3, B = 2)
1	1	1	1	0	

b. Write the sum of product form of the function from the truth table.

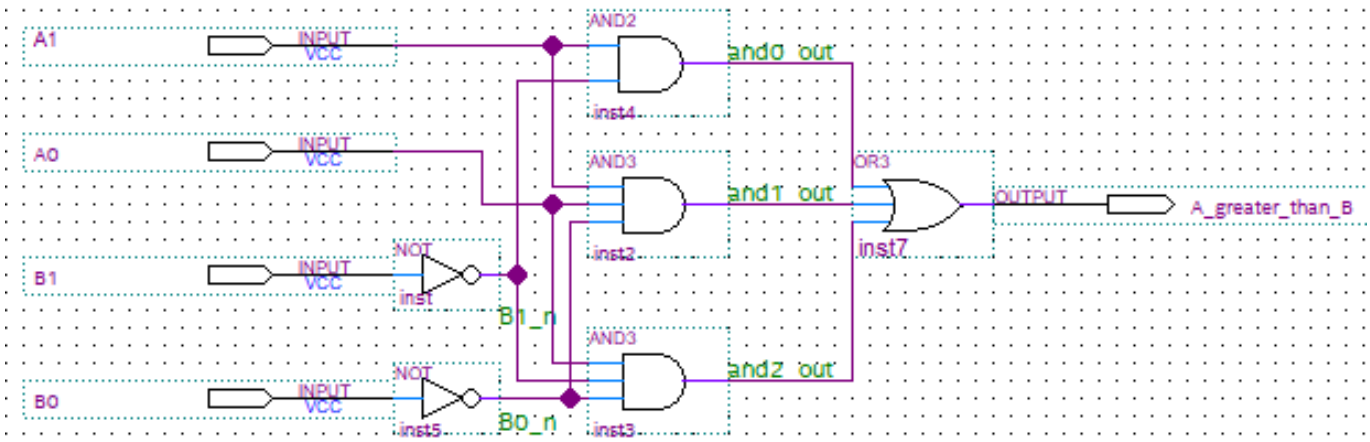
A_greater_than_B = A1’A0B1’B0’ + A1A0’B1’B0’ + A1A0’B1’B0 + A1A0B1’B0’ + A1A0B1’B0 + A1A0B1B0’

c. Simplify the function as much as possible

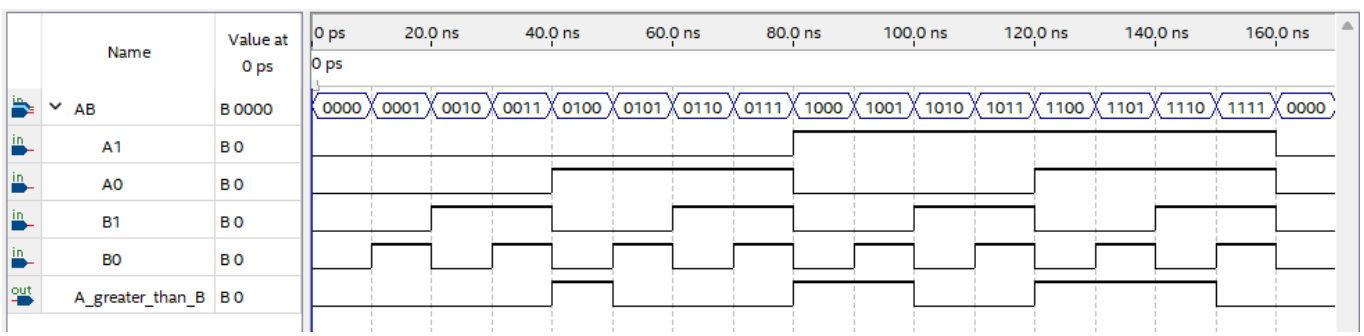
		B1B0			
		00	01	11	10
A1A0	00				
	01	1			
	11	1	1		1
	10	1	1		

A_greater_than_B = A0B1'B0' + A1A0B0'+A1B1'

d. Design a logic diagram for the simplified function



e. Run the simulation of the logic diagram using Simulation Waveform Editor tool



f. Write a Verilog structural description (gate-entry modeling) for the simplified function

comparator_greater_than_structural.v

```
1
2   module comparator_greater_than_structural(A, B, A_greater_than_B);
3      input [1:0] A, B;
4      output A_greater_than_B;
5      wire B0_n, B1_n, and0_out, and1_out, and2_out;
6
7      not inv0(B0_n, B[0]);
8      not inv1(B1_n, B[1]);
9      and and0(and0_out, A[1], B1_n);
10     and and1(and1_out, A[1], A[0], B0_n);
11     and and2(and2_out, A[0], B1_n, B0_n);
12     or or0(A_greater_than_B, and0_out, and1_out, and2_out);
13  endmodule
14
```

- **Set the Verilog file as Top-Level Entity**

Project Navigator | Hierarchy

Entity:Instance

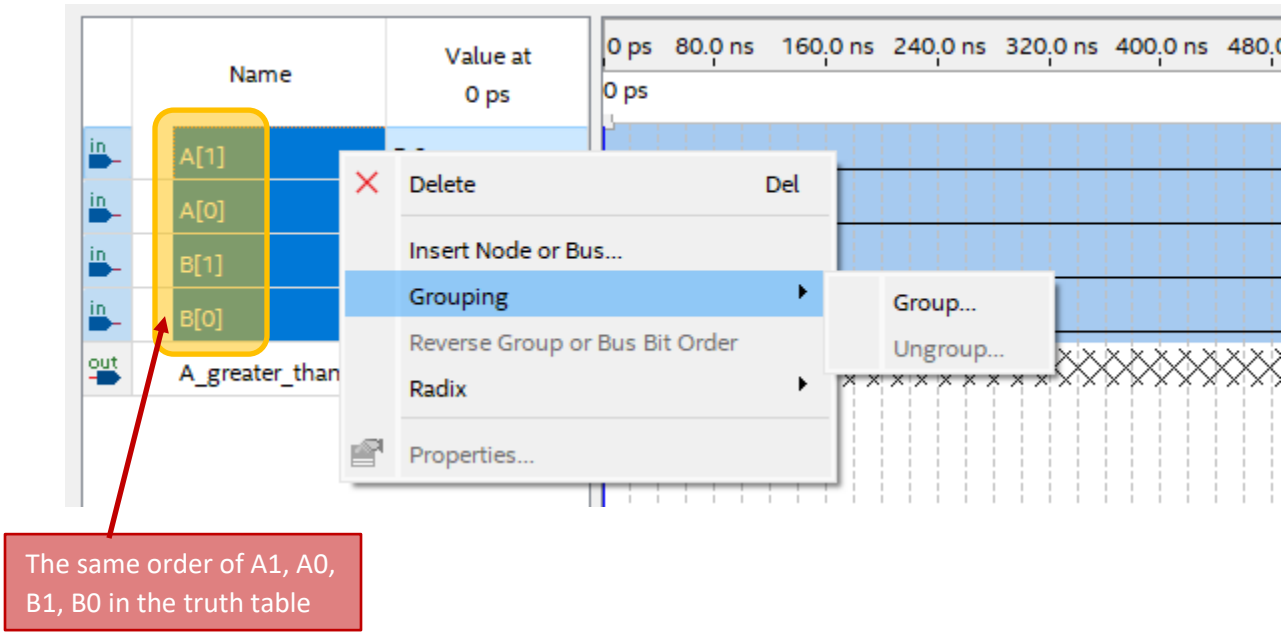
MAX 10: 10M02SCU169C8G

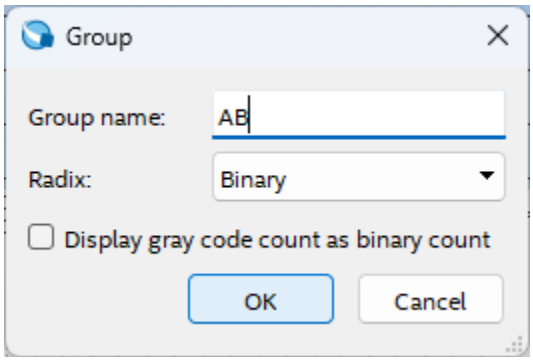
comparator_greater_than_structural

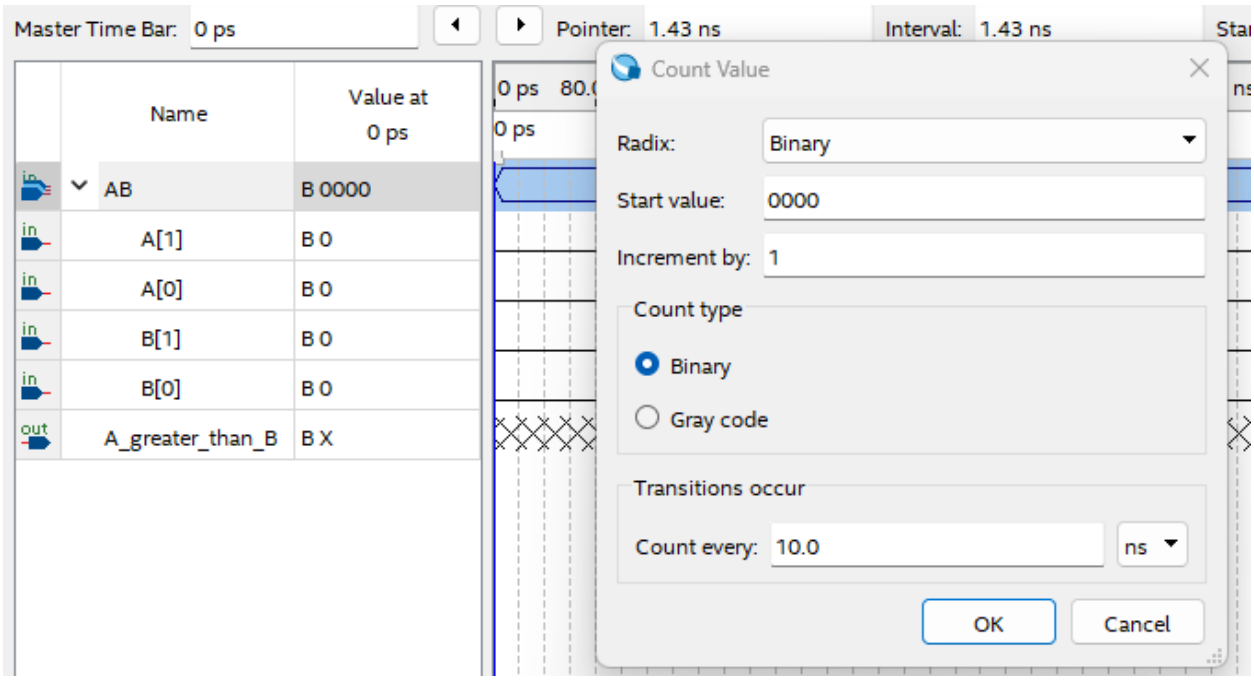
comparator_greater_than_structural.v

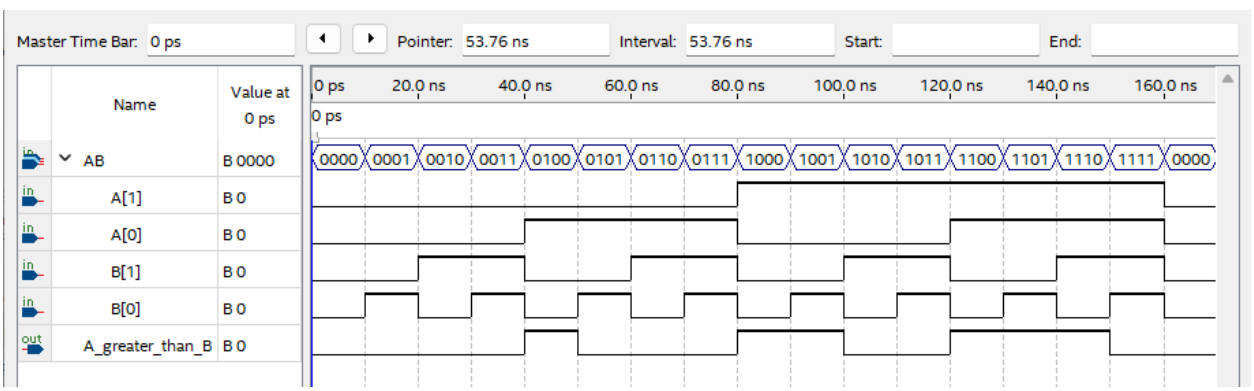
```
1
2   module comparator_greater_than_structural(A,
3      input [1:0] A, B;
4      output A_greater_than_B;
5      wire B0_n, B1_n, and0_out, and1_out, and2
6
7      not inv0(B0_n, B[0]);
8      not inv1(B1_n, B[1]);
9      and and0(and0_out, A[1], B1_n);
10     and and1(and1_out, A[1], A[0], B0_n);
11     and and2(and2_out, A[0], B1_n, B0_n);
12     or or0(A_greater_than_B, and0_out, and1_c
13  endmodule
14
```

g. Run the simulation of the Verilog code using Simulation Waveform Editor tool





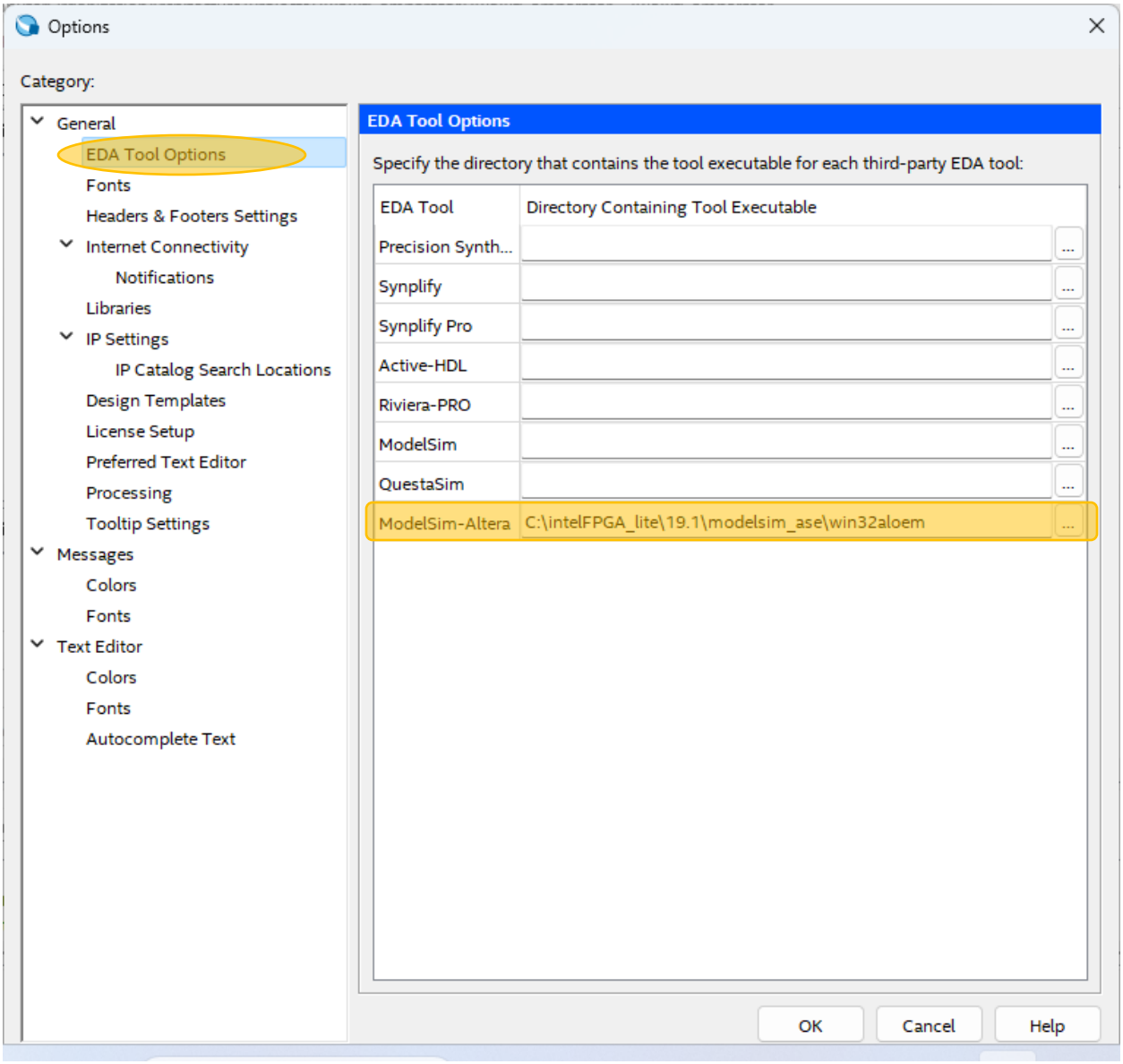




g. Write a Verilog testbench for the Verilog code to test the model

- When you installed the Quartus Prime Lite software, ModelSim - Intel FPGA Starter edition was installed. We need to configure the Quartus Prime Lite to use ModelSim.

- Go to **Tools\Options**:



- Write the Verilog testbench to test the module **comparator_greater_than_structural**:

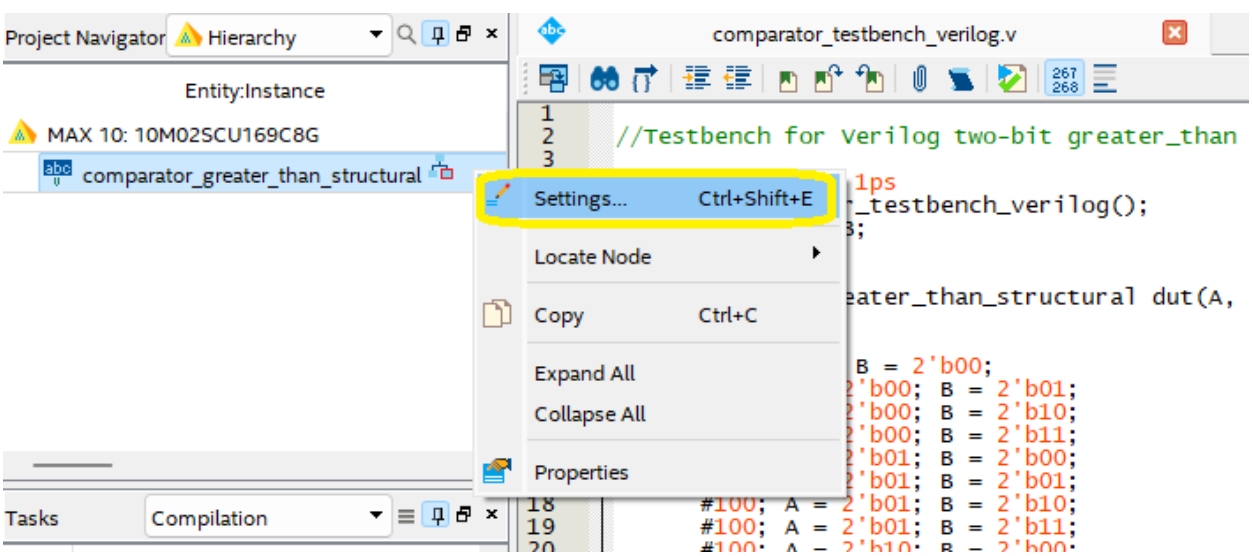
comparator_testbench_verilog.v

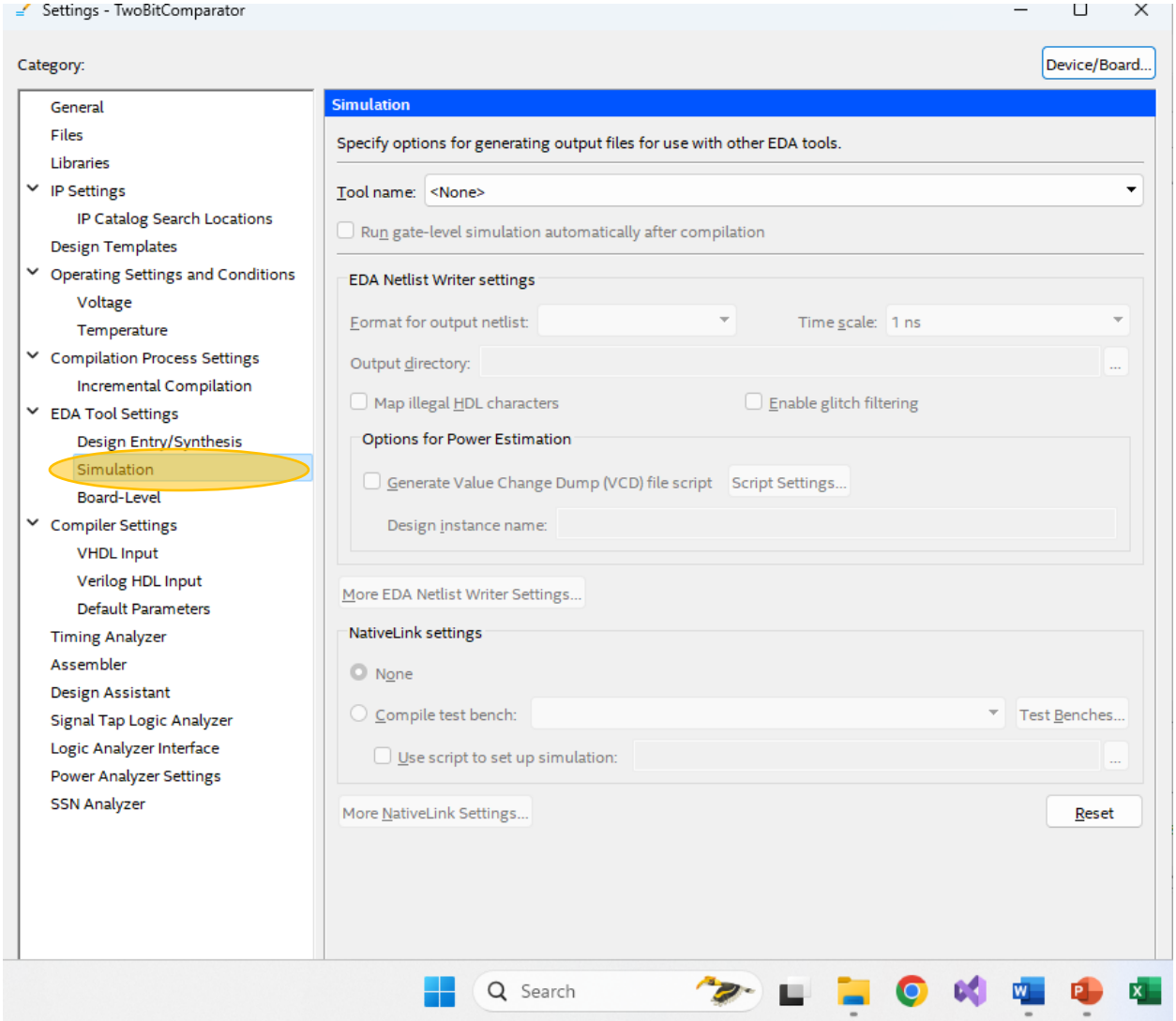
```verilog
//Testbench for Verilog two-bit greater_than comparator

`timescale 1ns / 1ps
module comparator_testbench_verilog();
   reg [1:0] A, B;
   wire dut_out;

   comparator_greater_than_structural dut(A, B, dut_out);
   initial
   begin
      $monitor("monitor: Value of A = %b, B = %b", A, B);
      $display("display: Starting ..... A = %b, B = %b", A, B);
      A = 2'b00; B = 2'b00;
      #100; A = 2'b00; B = 2'b01;
      #100; A = 2'b00; B = 2'b10;
      #100; A = 2'b00; B = 2'b11;
      #100; A = 2'b01; B = 2'b00;
      #100; A = 2'b01; B = 2'b01;
      #100; A = 2'b01; B = 2'b10;
      #100; A = 2'b01; B = 2'b11;
      #100; A = 2'b10; B = 2'b00;
      #100; A = 2'b10; B = 2'b01;
      #100; A = 2'b10; B = 2'b10;
      #100; A = 2'b10; B = 2'b11;
      #100; A = 2'b11; B = 2'b00;
      #100; A = 2'b11; B = 2'b01;
      #100; A = 2'b11; B = 2'b10;
      #100; A = 2'b11; B = 2'b11;
      $display("display: Ending ..... A = %b, B = %b", A, B);
   end
endmodule
```

- Configure EDA Tool Settings for simulation




Settings - TwoBitComparator
Category:
Device/Board...
General
Files
Libraries
IP Settings
IP Catalog Search Locations
Design Templates
Operating Settings and Conditions
Voltage
Temperature
Compilation Process Settings
Incremental Compilation
EDA Tool Settings
Design Entry/Synthesis
Simulation
Board-Level
Compiler Settings
VHDL Input
Verilog HDL Input
Default Parameters
Timing Analyzer
Assembler
Design Assistant
Signal Tap Logic Analyzer
Logic Analyzer Interface
Power Analyzer Settings
SSN Analyzer
Simulation
Specify options for generating output files for use with other EDA tools.
Tool name: <None>
Run gate-level simulation automatically after compilation
EDA Netlist Writer settings
Format for output netlist:
Time scale: 1 ns
Output directory:
Map illegal HDL characters
Enable glitch filtering
Options for Power Estimation
Generate Value Change Dump (VCD) file script
Script Settings...
Design instance name:
More EDA Netlist Writer Settings...
NativeLink settings
None
Compile test bench:
Test Benches...
Use script to set up simulation:
More NativeLink Settings...
Reset
Search

Settings - TwoBitComparator

Category:

Device/Board...

- General
- Files
- Libraries
- IP Settings
  - IP Catalog Search Locations
- Design Templates
- Operating Settings and Conditions
  - Voltage
  - Temperature
- Compilation Process Settings
  - Incremental Compilation
- EDA Tool Settings
  - Design Entry/Synthesis
  - Simulation
  - Board-Level
- Compiler Settings
  - VHDL Input
  - Verilog HDL Input
  - Default Parameters
- Timing Analyzer
- Assembler
- Design Assistant
- Signal Tap Logic Analyzer
- Logic Analyzer Interface
- Power Analyzer Settings
- SSN Analyzer

Simulation

Specify options for generating output files for use with other EDA tools.

Tool name: ModelSim-Altera

☐ Run gate-level simulation automatically after compilation

EDA Netlist Writer settings

Format for output netlist: Verilog HDL

Time scale: 1 ns

Output directory: simulation/modelsim

☐ Map illegal HDL characters

☐ Enable glitch filtering

Options for Power Estimation

☐ Generate Value Change Dump (VCD) file script

Script Settings...

Design instance name:

More EDA Netlist Writer Settings...

NativeLink settings

○ None

◉ Compile test bench:

Test Benches...

☐ Use script to set up simulation:

○ Script to compile test bench:

More NativeLink Settings...

Reset

Search

Test Benches

Specify settings for each test bench.

Existing test bench settings:

New...

Name	Top Level Module	Design Instance	Run For	Test Bench File(s)

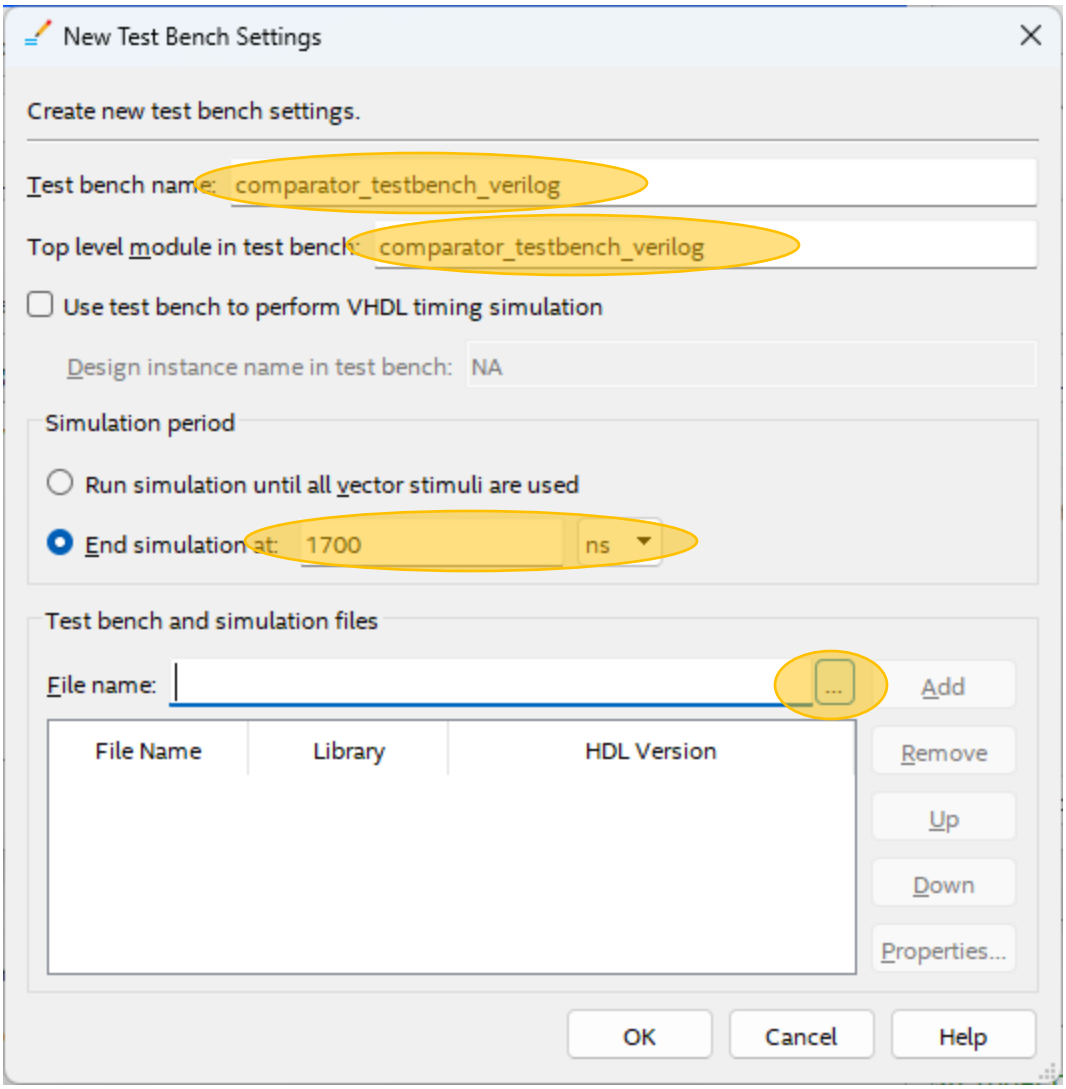
Edit...

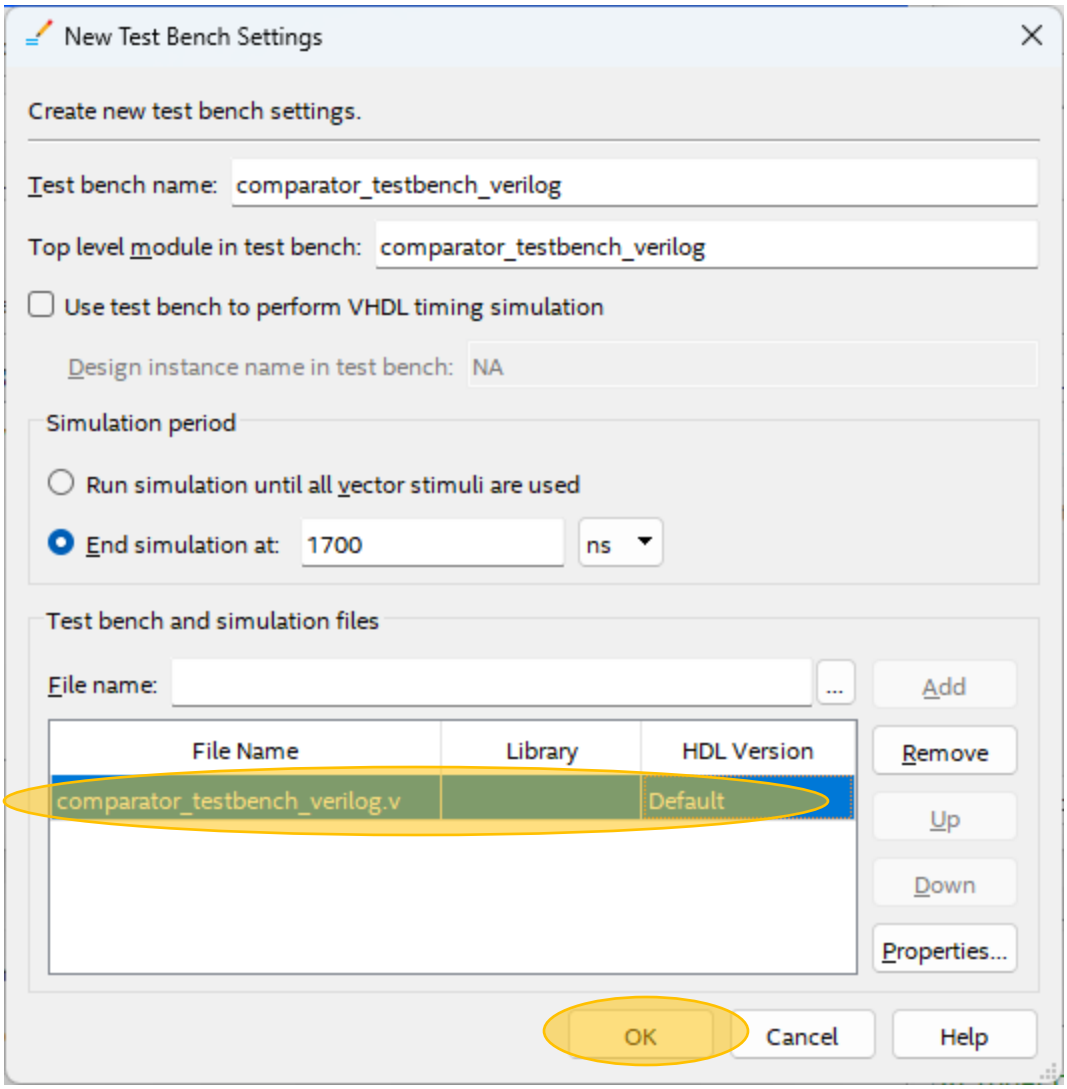
Delete

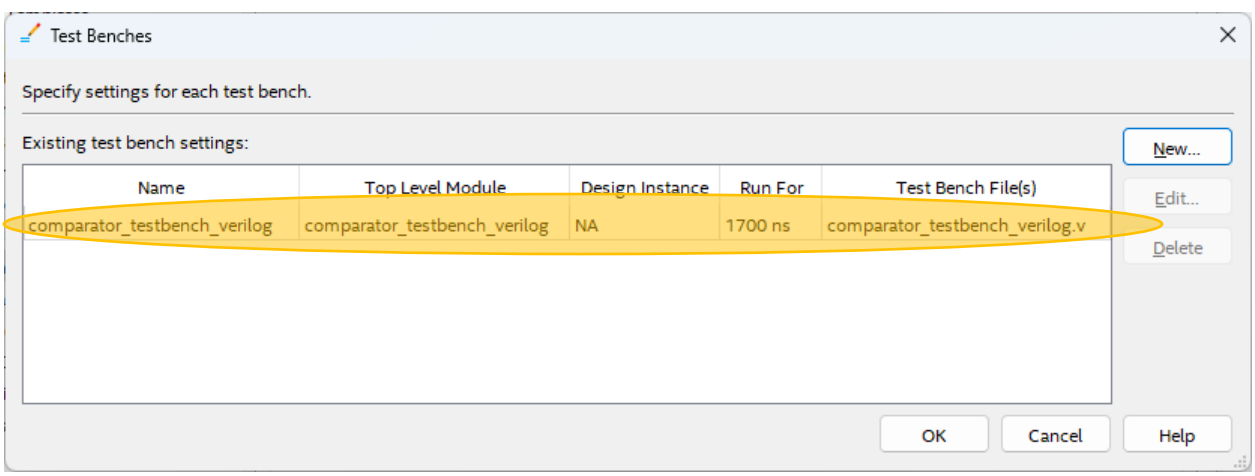
OK

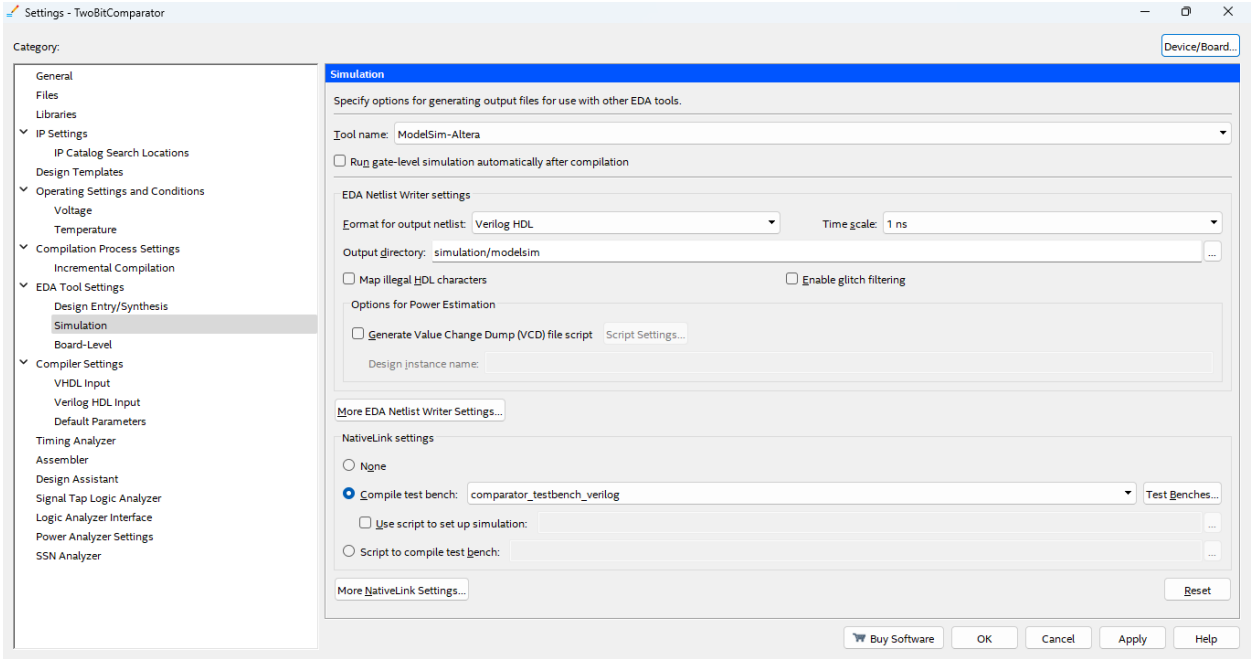
Cancel

Help

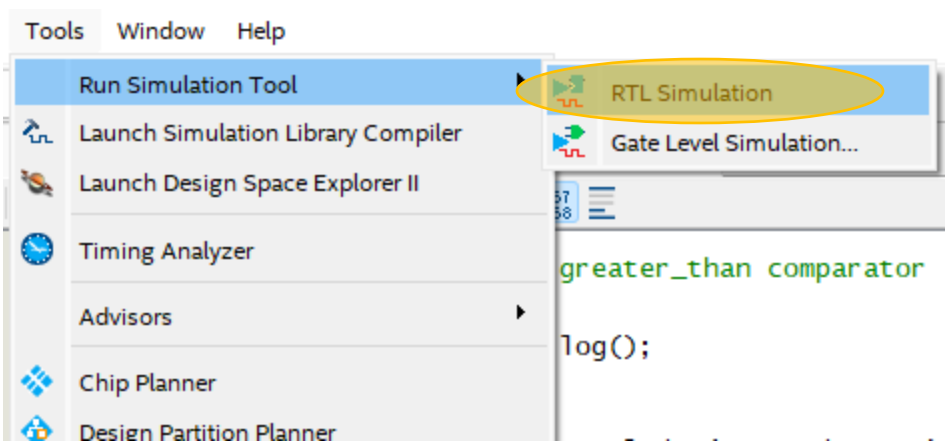

New Test Bench Settings
Create new test bench settings.
Test bench name: comparator_testbench_verilog
Top level module in test bench: comparator_testbench_verilog
Use test bench to perform VHDL timing simulation
Design instance name in test bench: NA
Simulation period
Run simulation until all vector stimuli are used
End simulation at: 1700 ns
Test bench and simulation files
File name:
...
Add
File Name
Library
HDL Version
Remove
Up
Down
Properties...
OK
Cancel
Help

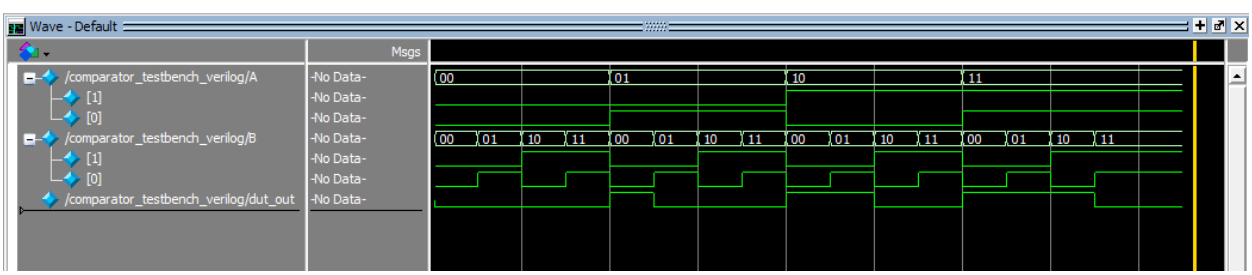

New Test Bench Settings
Create new test bench settings.
Test bench name: comparator_testbench_verilog
Top level module in test bench: comparator_testbench_verilog
Use test bench to perform VHDL timing simulation
Design instance name in test bench: NA
Simulation period
Run simulation until all vector stimuli are used
End simulation at: 1700
ns
Test bench and simulation files
File name:
...
Add
File Name
Library
HDL Version
comparator_testbench_verilog.v
Default
Remove
Up
Down
Properties...
OK
Cancel
Help


Test Benches
Specify settings for each test bench.
Existing test bench settings:
New...
Edit...
Delete
Name
Top Level Module
Design Instance
Run For
Test Bench File(s)
comparator_testbench_verilog
comparator_testbench_verilog
NA
1700 ns
comparator_testbench_verilog.v
OK
Cancel
Help



- Compile and run simulation using ModelSim





Assignment 2 (20 Points)

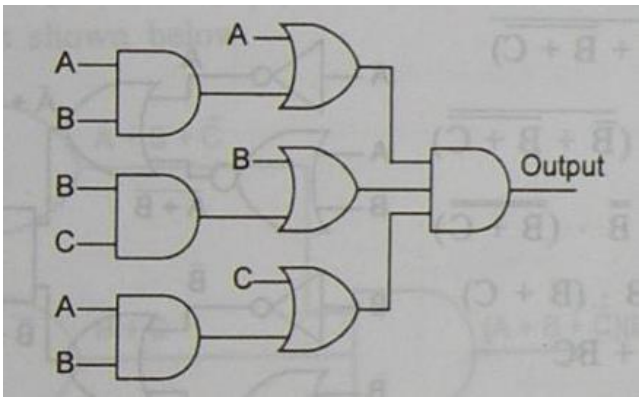
Design a logic diagram for the following Boolean expressions and write a Verilog program using gate-level modeling and verify the waveform with its simplified Boolean expression (as much as possible) truth table.

a) AB(C + D) + AB(C + D)'

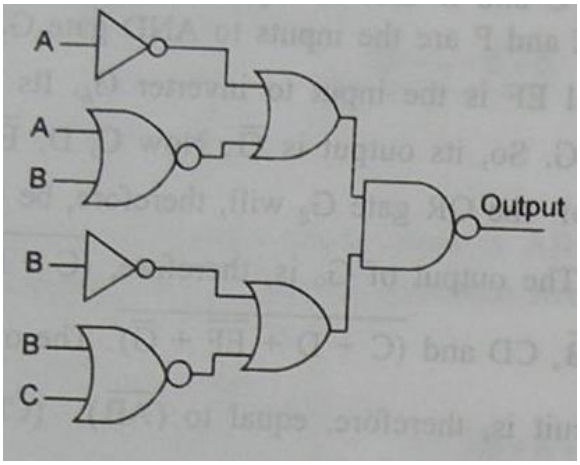
b) AB'C + B + BD' + ABD' + A'C

Assignment 3 (20 Points)

Write the Boolean expressions for the following a logic diagram and write a Verilog program using gate-level modeling for the expression. Then verify the waveform with its simplified Boolean expression (as much as possible) truth table.



(a)



(b)

Assignment 4 (20 Points)

Reduce the following expression in both SOP and POS forms using 3 variable K-map and design the logic diagram. Now write a Verilog program using gate-level modeling for these simplified Boolean expressions and verify them with their truth tables and waveforms.

a) $\Sigma\, m(0, 2, 3, 4, 5, 6)$

b) $\Pi\, M(0, 1, 2, 3 , 4, 7)$

Assignment 5 (20 Points)

Reduce the following expression in both SOP and POS forms using 4 variable K-map and design the logic diagram. Now write a Verilog program using gate-level modeling for these simplified Boolean expressions and verify them with their truth tables and waveforms.

a) $\Sigma\, m(0, 1, 2, 3, 5, 7, 8, 9, 10, 12, 13)$

b) $\Pi\, M(2, 8, 9, 10, 11, 12, 14)$