MEGHNAD SAHA INSTITUTE OF TECHNOLOGY

LABORATORY MANUAL

DEPARTMENT : Computer Science and Engineering

SESSION : Even

SUBJECT : Computer Architecture Lab

SUBJECT CODE : PCC-CS492

SEMESTER : 4th

DEPARTMENT OFCOMPUTER SCIENCE AND ENGINEERING

Vision

- **To attain a global platform in academics, research and innovation by preparing competent computer engineers to cater for the needs of industry and society at large.**

Mission

- **To address the dynamic & growing needs of the software industry by creating quality professionals with a strong focus on principles of Computer Science and Engineering.**

- **To provide state-of-the-art infrastructure to facilitate the research work for enhancing the knowledge in emerging technologies including machine learning models, data science, cybersecurity, and IoT etc.**

- **To strengthen the industry-academic relationship through collaboration with global IT organizations, healthcare units and relevant institutions for data sharing and developing technology.**

- **To nurture the students by inculcating the spirit of ethical and social values through creating strong foundation in ethical coding and computational design paradigms.**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PROGRAMME EDUCATIONAL OBJECTIVES(PEOs)

PEO	PROGRAMME EDUCATIONAL OBJECTIVES
PEO1	To prepare graduates who will be successful software professionals, academicians, researchers related to computational frameworks, and entrepreneurial pursuit.
PEO2	To prepare graduates who will achieve peer recognition and adapt to the new technological environment as an individual or in a team demonstrating good computational, analytical, model designing, and software implementation skills.
PEO3	To prepare graduates who will thrive to pursue life-long learning for keeping pace with dynamic technological changes, through the development of an intuitive computational learning paradigm.
PEO4	To foster the values of professional and societal ethics for generating responsible citizens with proven expertise in computational domain.

PO	PROGRAM OUTCOME DETAILS
PO 1	**Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the Solution of complex engineering problems.
PO 2	**Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO 3	**Design / development of solutions**: Design solutions for complex engineering problems and design system components or processes thatmeetthespecifiedneedswithappropriateconsiderationforthepublichealthand safety, and the cultural, societal, and environmental considerations.
PO 4	**Conduct investigations of complex problems: Use** research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
PO 5	**Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
PO 6	**The engineer and society:** Apply reasoning in formed by the contextual knowledge to assessorial l, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
PO 7	**Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
PO 8	**Ethics:** Apply ethical principles and commit to professional ethics and Responsibilities and norms of the engineering practice.
PO 9	**Individual and team work:** Function effectively as an individual, and as a Member or leader in diverse teams, and in multidisciplinary settings.

PO 10	**Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
PO 11	**Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage Projects and in multidisciplinary environments.
PO 12	**Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest Context of technological change.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PROGRAM SPECIFIC OUTCOMES(PSOs)

After the completion of the course, B.Tech Computer Science and Engineering, the graduates will Have the following Program Specific Outcomes:

PSO	PROGRAMSPECIFICOUTCOMESDETAILS
PSO1	**Apply the skills of basic science, discrete mathematics, probability, principles of electrical and electronics, and programming aptitude to develop the self-learning capabilities for the modelling and designing of computing systems in a way of solving engineering problems.**
PSO2	**Evaluate the solution framework of complex engineering problems by applying algorithms, tools, and techniques related to computer science to identify the optimal solutions for enriching computational research and development.**
PSO3	**Strengthen the industry-institute partnership and socio-economic framework by enhancing the emerging areas of computer science & engineering for the growing needs of society with computational solutions.**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GENERAL LABORATORY INSTRUCTIONS

1. Students are advised to come to the laboratory at least 5 minutes before (to the starting time), those who come after 5minutes will not be allowed in the lab.

2. Plan your task properly much before to the commencement, come prepared to the lab with the synopsis/program/experiment details.

3. Students should enter in the laboratory with:
    a. Laboratory observation notes with all the details (Problem statement, Aim, Algorithm procedure, Program Expected Output, etc.,) filled in for the lab session.
    b. Laboratory Record updated up to the last session experiments and other utensils (if any) needed in the lab.
    c. Proper Dress code and Identity card.

4. Sign in the laboratory login register, write the TIME-IN, and occupy the computer system allotted to you by the faculty.

5. Execute your task in the laboratory, and record the results/output in the lab Observation notebook, and get certified by the concerned faculty.

6. All the students should be polite and cooperative with the laboratory staff must maintain the discipline and decency in the laboratory.

7. Students/Faculty must keep the mobile phones in SWITCHED OFF mode During the lab sessions. Misuse of the equipment, misbehavior with the staff and systems etc. will act severe punishment.

8. Students must take the permission of the faculty in case of any urgency to go out, if anybody found outside the lab / class without permission during working hours will be treated seriously and punished appropriately.

9. Students should LOGOFF/ SHUT DOWN the computer system before he/she leaves the lab after completing the task (experiment) in all aspects.

Head of the Department **Principal**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DO's

1. Follow the rules and guidelines set by the department for computer lab usage.
2. Follow the instructions provided by the Technical Assistant and Faculty members.
3. Use the equipment and software appropriately for educational purposes only.
4. Respect the equipment and take care of it. Do not tamper with or damage the computers, keyboards, mouse, and other equipment.
5. Keep the computer and the lab clean and organized.
6. Be respectful of other users and their privacy while inside the lab.
7. Log out of your computer account and shut down the computer properly before leaving the lab.
8. Ask for help if you need it. Don't be afraid to seek assistance from Technical Assistants or Faculty.
9. Report any technical issues to the technical staff or System Administrator.

DON'Ts

1. Don't use the lab for non-educational purposes such as playing games or browsing social media.
2. Don't enter the lab with your Bag & wearing shoes (only mobile and money purse allowed)
3. Don't download or install any unauthorized software or files on the computer.
4. Don't alter or modify the computer hardware or software in any way.
5. Don't eat or drink inside the computer lab to avoid spilling or damaging equipment.
6. Don't leave your personal belongings unattended in the lab.
7. Don't modify, delete or copy files that belong to other students or the lab.
8. Do not engage in any form of cyberbullying or harassment.
9. Do not use your personal devices in the lab without permission from the lab administrator or instructor.
10. Don't disturb other students by making excessive noise or talking loudly.
11. Do not share or copy copyrighted material.
12. Don't insert any kind of external device onto the computer.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

University Syllabus

Maulana Abul Kalam Azad University of Technology, West Bengal

(Formerly West Bengal University of Technology)

Syllabus for B.Tech in Computer Science &Engineering

(Applicable from the academic session 2020-2021)

Course Name: Computer Architecture Lab	
Code: PCC-CS492	Credit Points: 2
Contacts: 4 hrs./week	
Teaching Scheme: Continuous Assessment	
Full Marks: 100	
Internal Assessment: 40	External Assessment :60

Laboratory Experiments:

1. HDL introduction.
2. Basic digital logic base programming with HDL
3. 8-bit Addition, Multiplication, Division
4. 8-bit Register design
5. Memory unit design and perform memory operations.
6. 8-bit simple ALU design
7. 8-bit simple CPU design
8. Interfacing of CPU and Memory.

RECOMMENDED SYSTEM / SOFTWARE REQUIREMENTS:

1. CPU: Intel i5, RAM: 8 GB, HDD: 160 GB System.
2. Windows 10 & Xilinx 14.7

USEFULTEXTBOOKS/REFERECES:

1. B.Ram – "Computer Organization & Architecture", Newage Publications.
2. N. Mathivanan, "Microprocessors, PC Hardware and Interfacing", Prentice Hall, 2004.
3. Rajaraman – "Computer Organization & Architecture", PHI.
4. Verilog HDL: A Guide to Digital Design and Synthesis by **Samir Palnitkar**, 2nd Edition.
5. Verilog Digital System Design, by **Zainalabedin Navabi.**
6. Verilog: Frequently Asked Questions,**Shivakumar Chonnad.**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Assessment Methods of University Lab Examination

Internal Assessment	**40Marks**
Practical Assessment	**60marks**
Total	**100Marks**

Internal Assessment(40Marks)

PCA1	PCA2
Lab Report and Viva (25)	**Lab Report and Viva (25)**
Lab Conduct Session (15)	**Lab Conduct Session (15)**

Practical Assessment(60Marks)

Viva	**20Marks**
Experiment and Result	**40Marks**

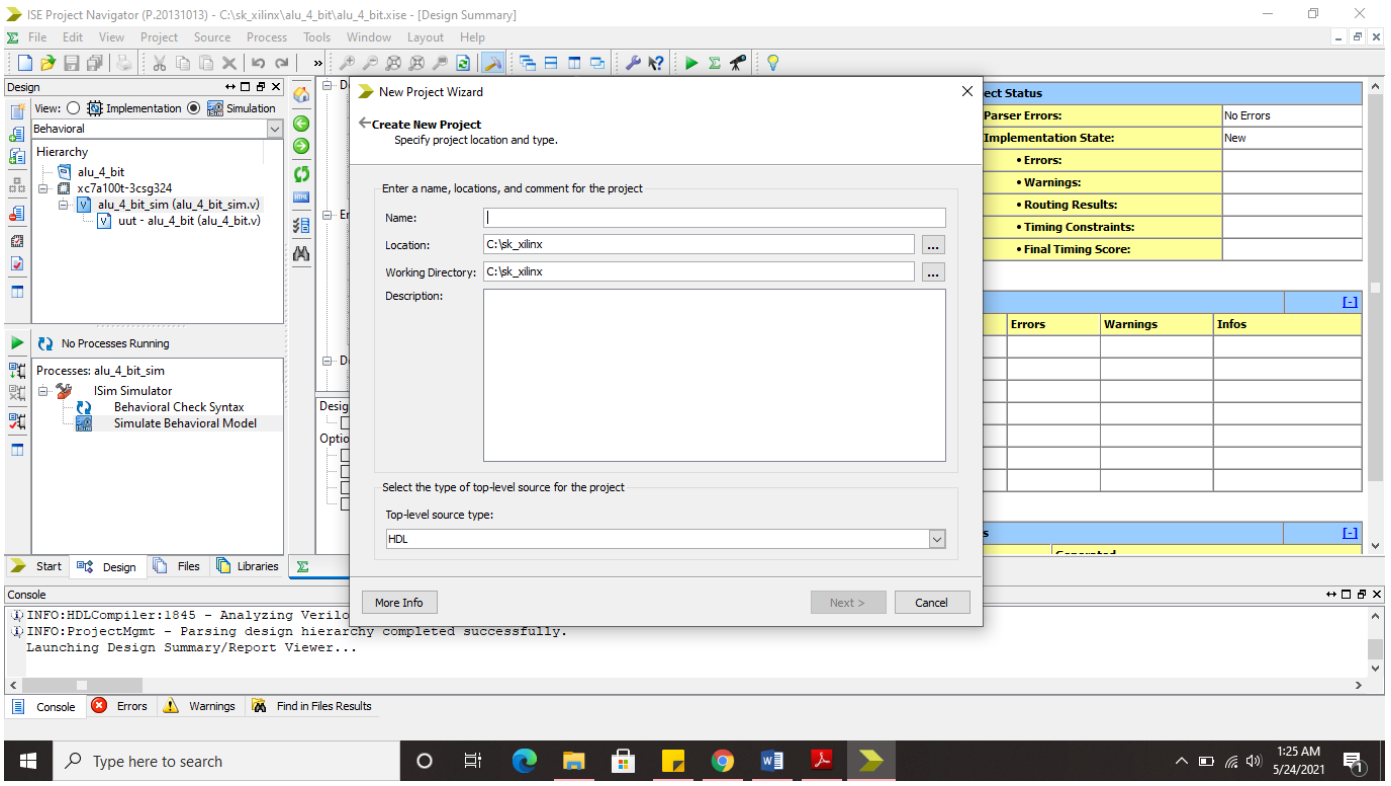
ASSIGNMENT LIST

Procedure for performing each experiment:

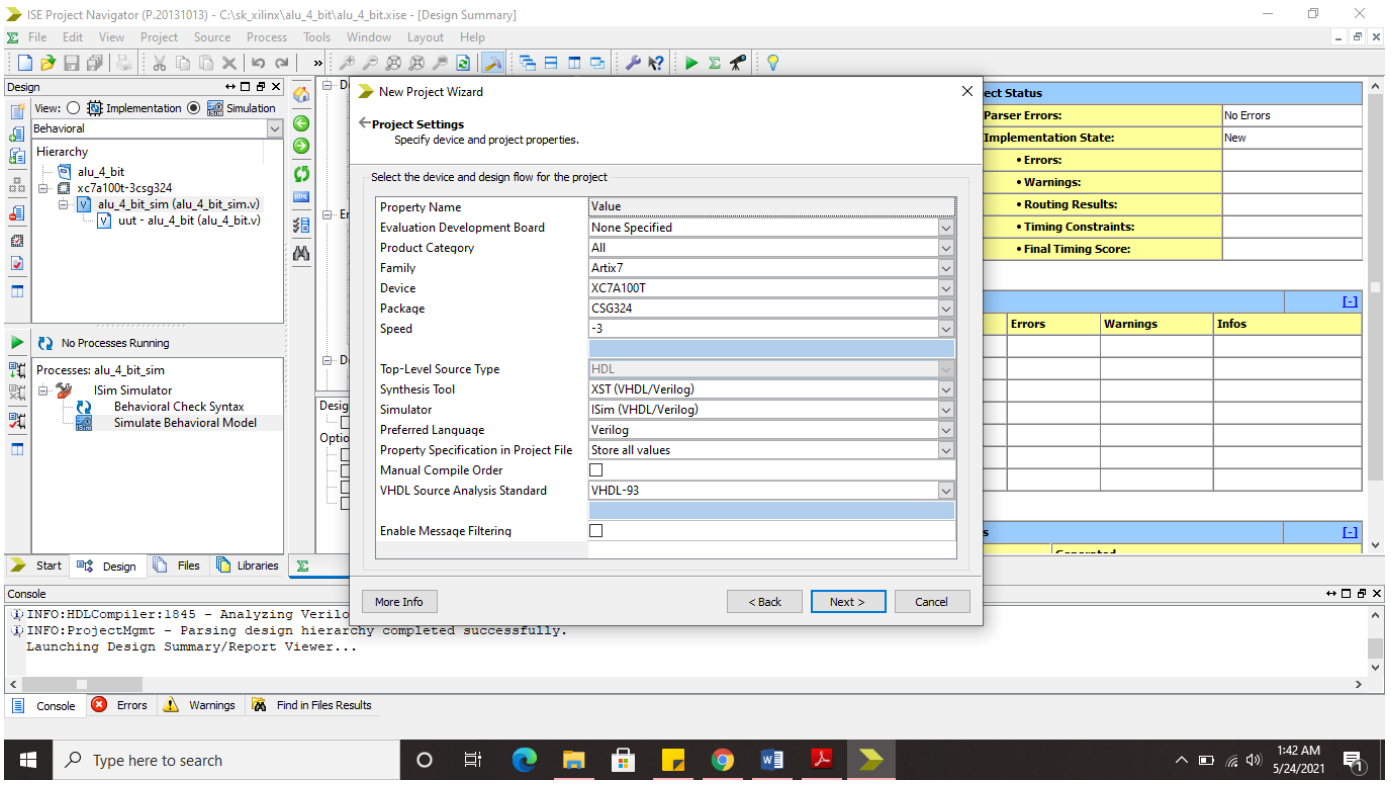
Students are hereby advised to follow the following steps to perform the experiments:

Create Implementation File

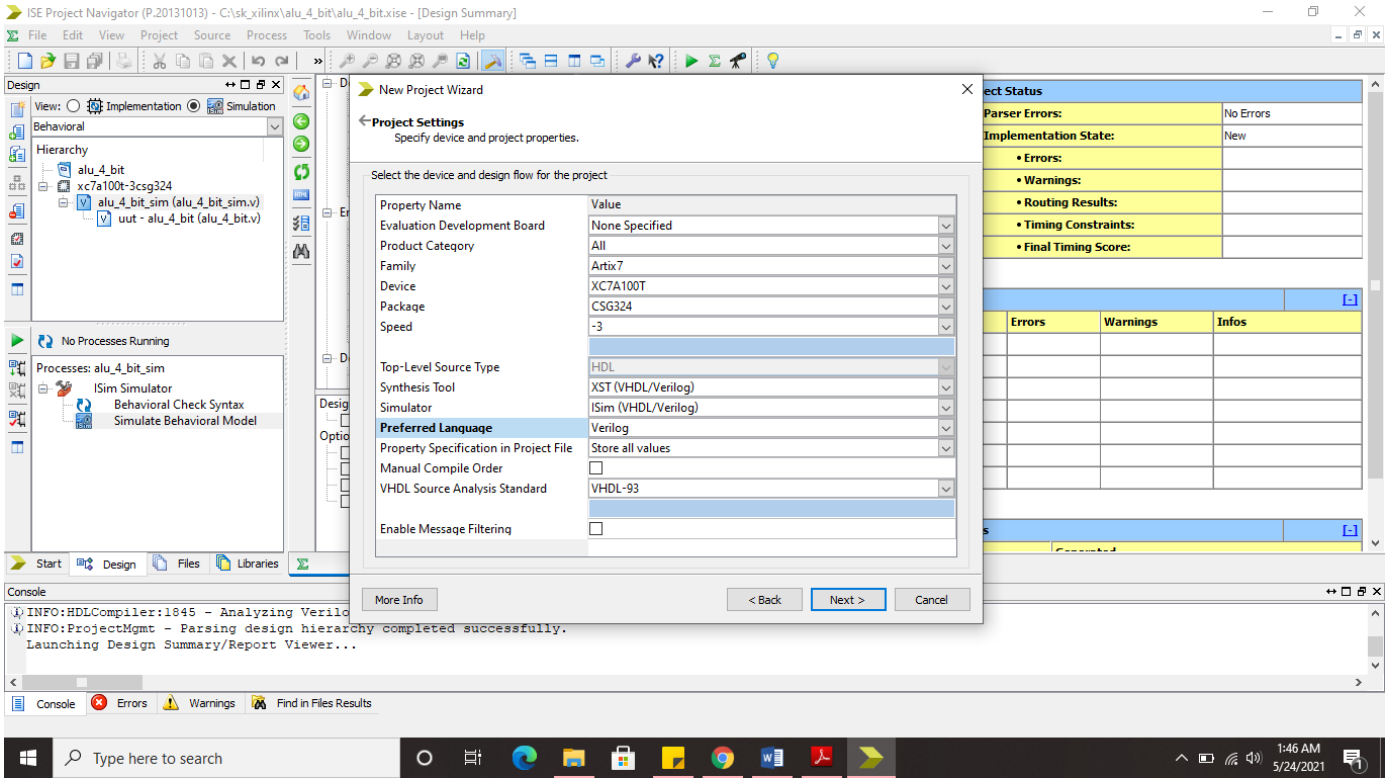
Step 1. Open/Initiate the Xilinx ISE. Then click File ⟶ New Project.



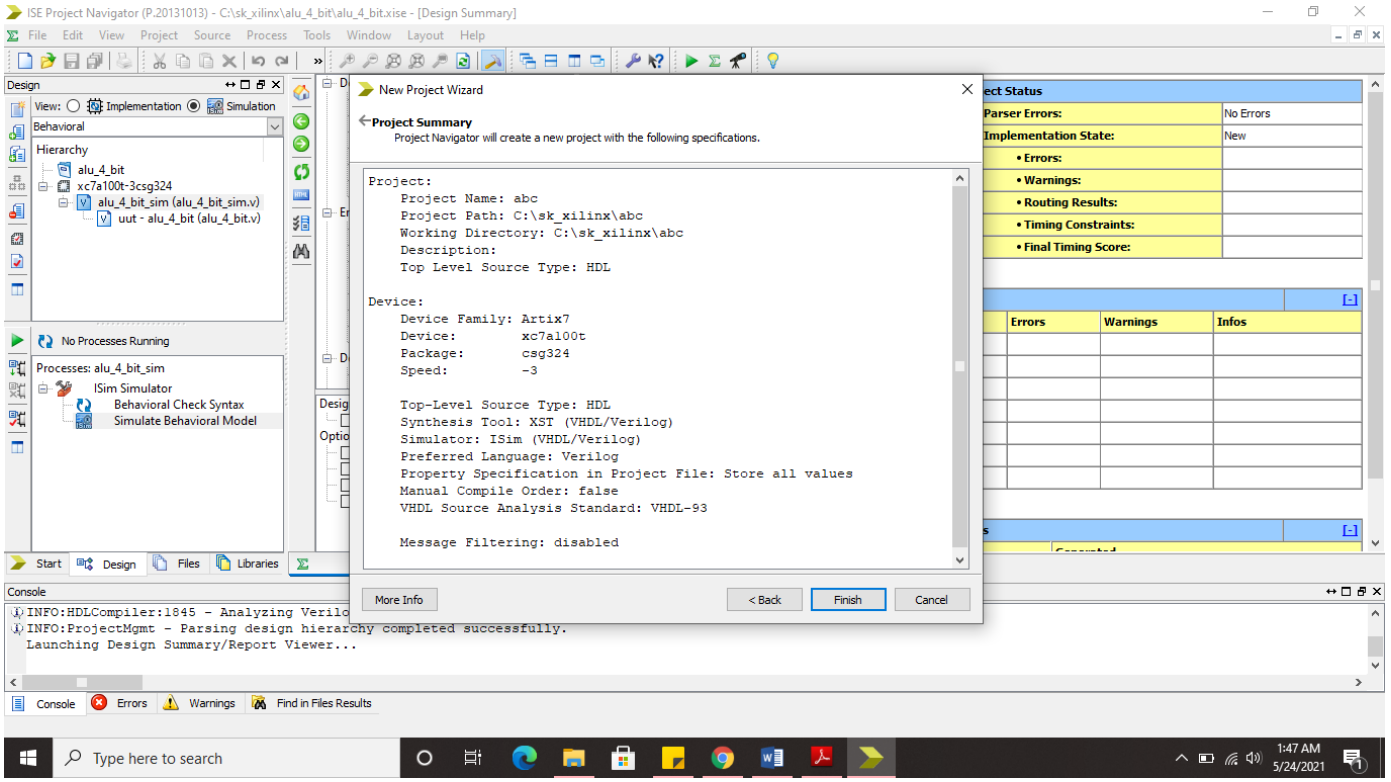
Give File Name -> Next

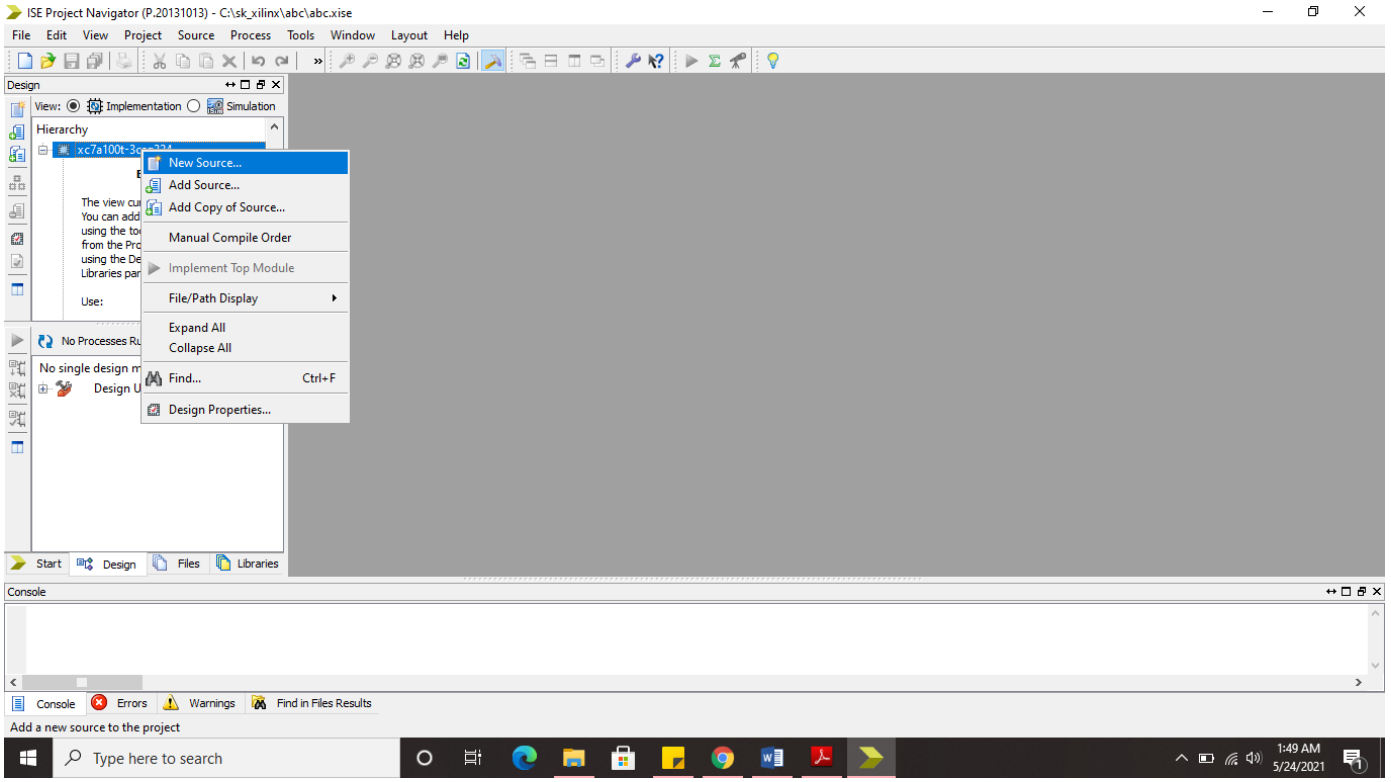


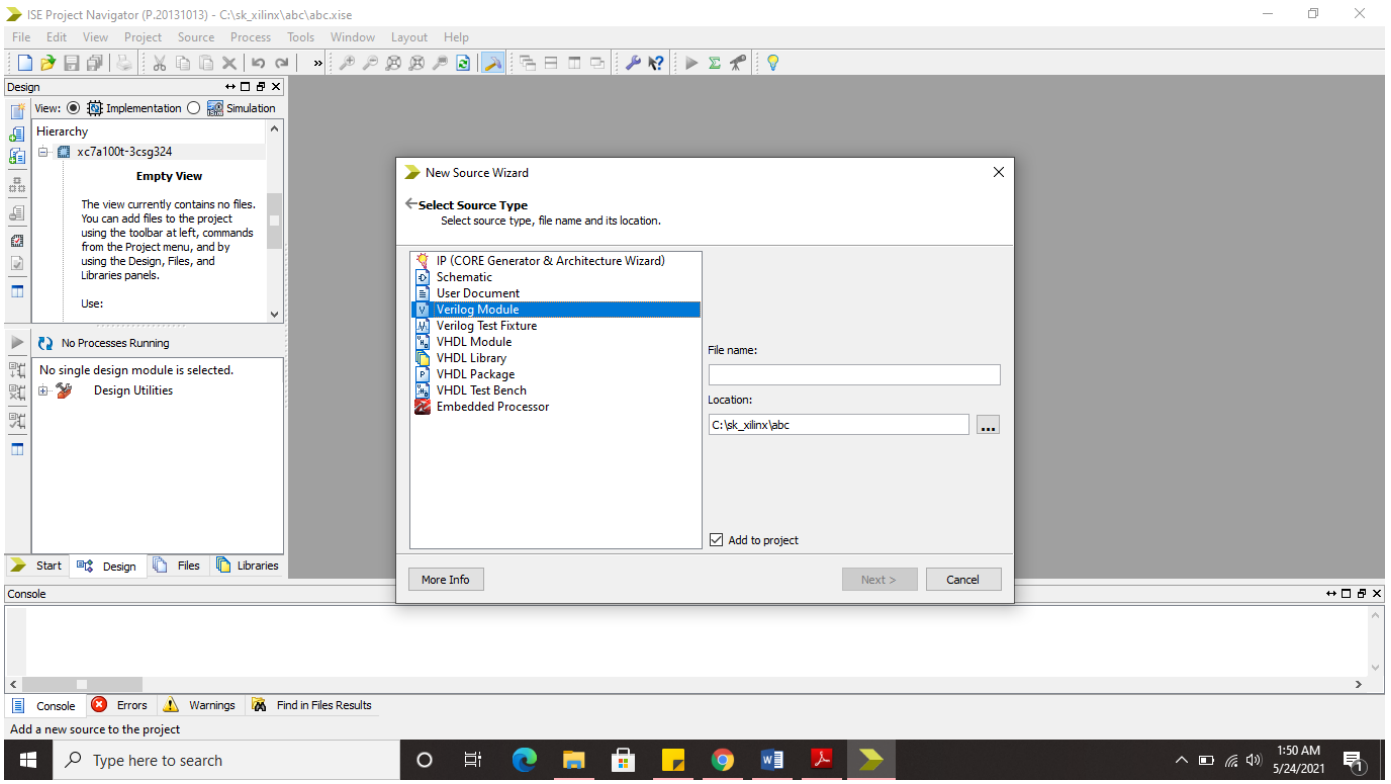
Select preferred language Verilog & next

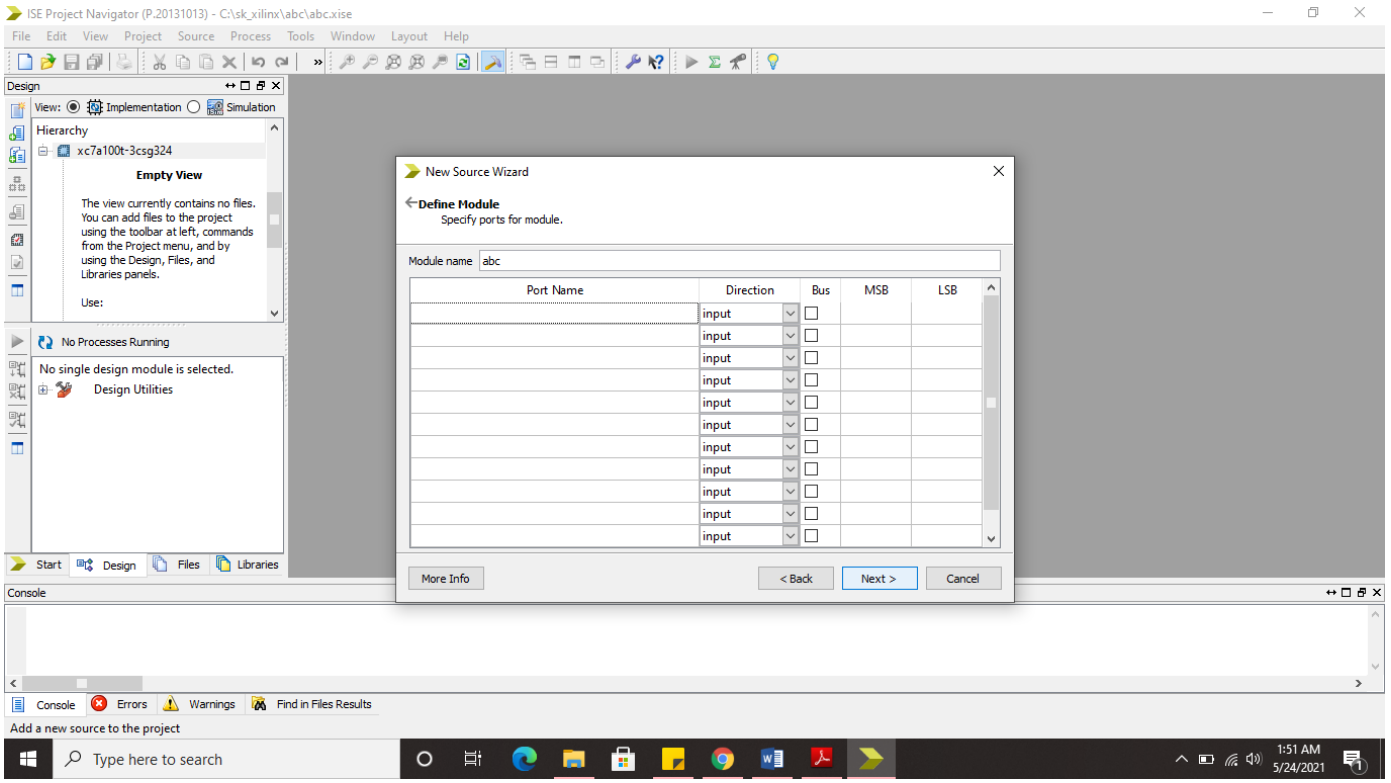


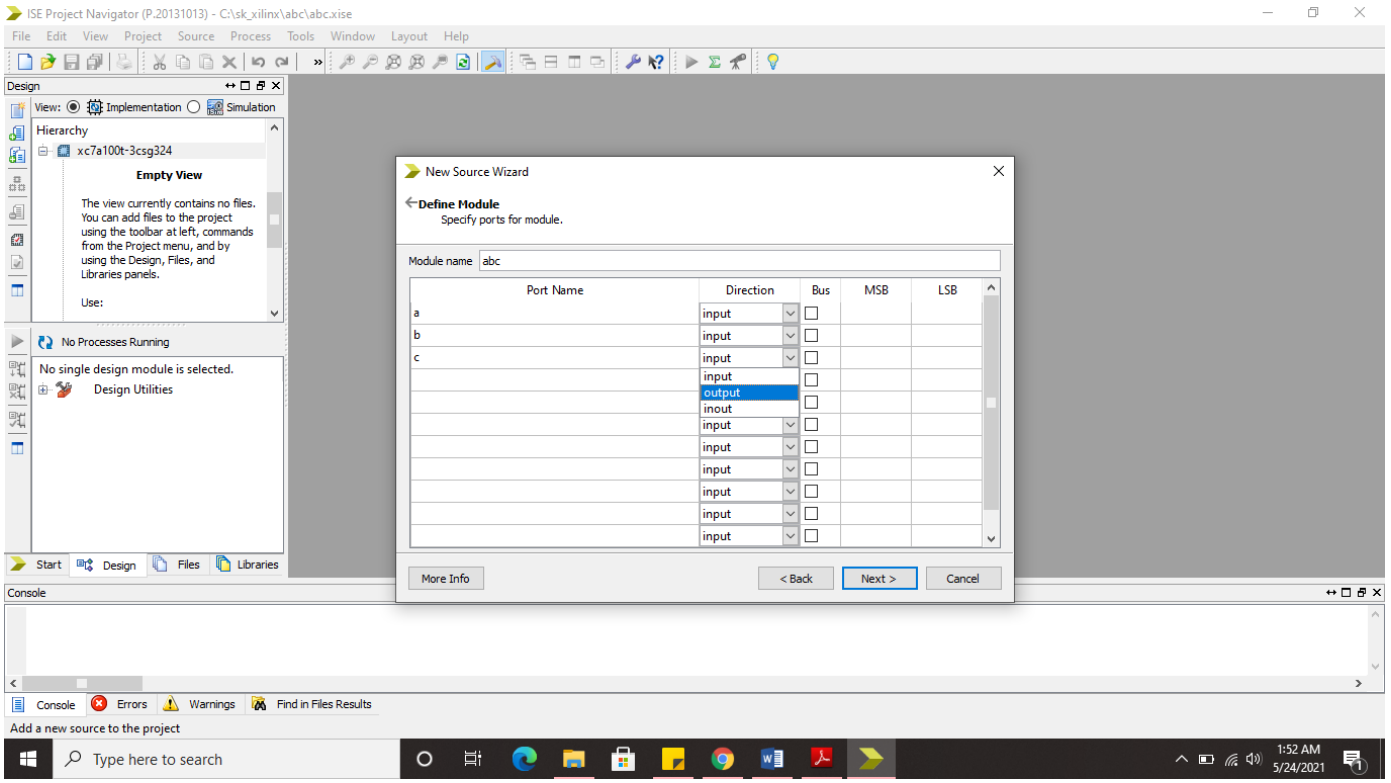
Click Finish.

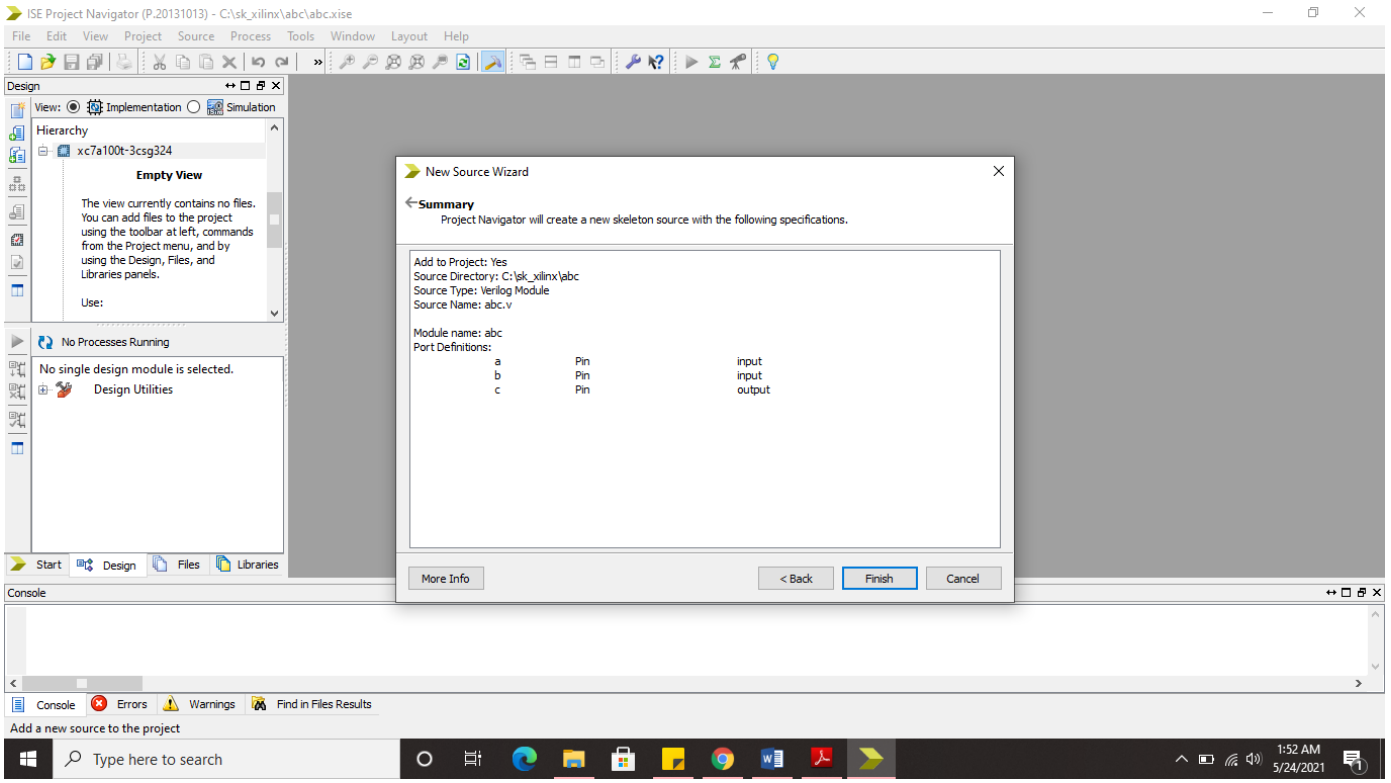


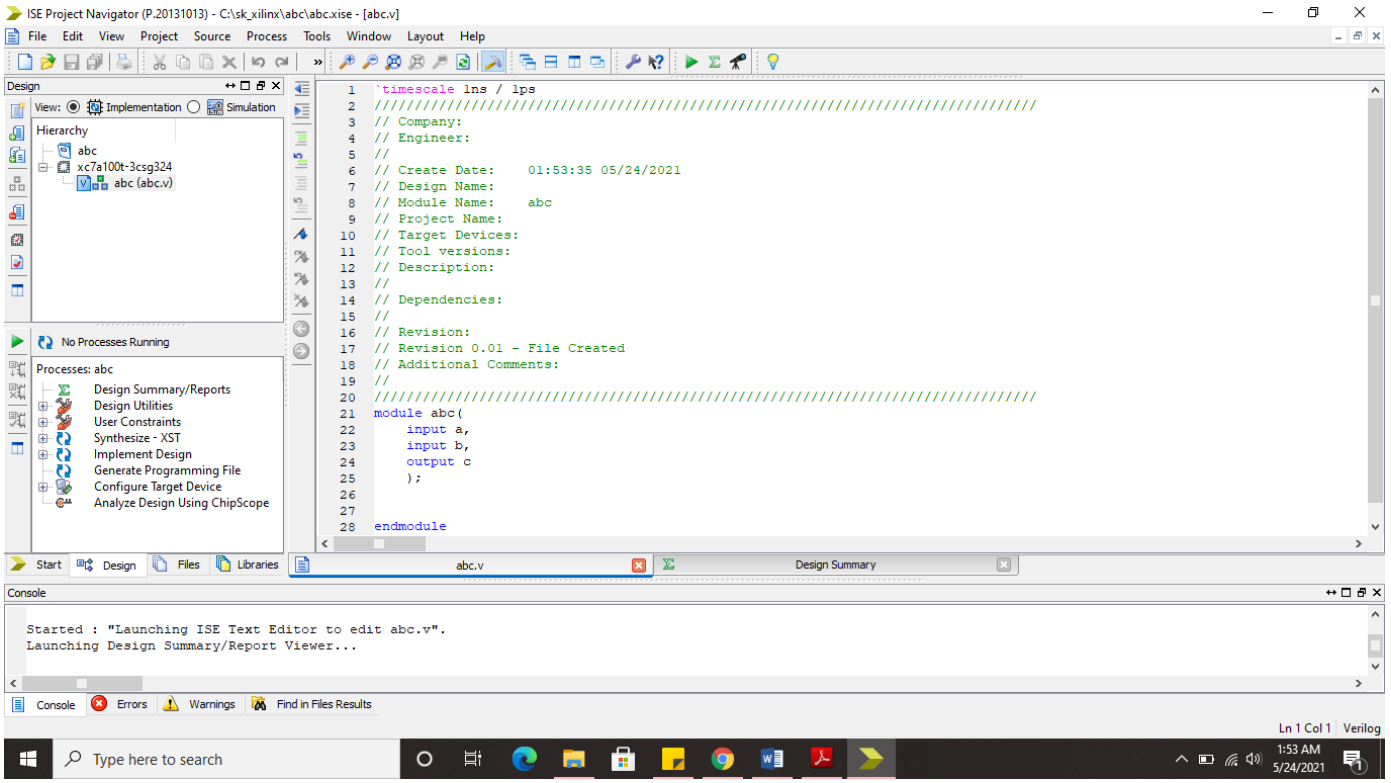

ISE Project Navigator (P.20131013) - C:\sk_xilinx\abc\abc.xise
File Edit View Project Source Process Tools Window Layout Help
Design
View: Implementation Simulation
Hierarchy
New Source...
Add Source...
Add Copy of Source...
Manual Compile Order
Implement Top Module
File/Path Display
Expand All
Collapse All
Find... Ctrl+F
Design Properties...
No Processes Running
No single design module is selected.
Design Utilities
Start Design Files Libraries
Console
Console Errors Warnings Find in Files Results
Add a new source to the project
Type here to search
1:49 AM
5/24/2021


ISE Project Navigator (P.20131013) - C:\sk_xilinx\abc\abc.xise
File Edit View Project Source Process Tools Window Layout Help
Design
View: Implementation Simulation
Hierarchy
xc7a100t-3csg324
Empty View
The view currently contains no files. You can add files to the project using the toolbar at left, commands from the Project menu, and by using the Design, Files, and Libraries panels.
Use:
No Processes Running
No single design module is selected.
Design Utilities
Start Design Files Libraries
New Source Wizard
Select Source Type
Select source type, file name and its location.
IP (CORE Generator & Architecture Wizard)
Schematic
User Document
Verilog Module
Verilog Test Fixture
VHDL Module
VHDL Library
VHDL Package
VHDL Test Bench
Embedded Processor
File name:
Location:
C:\sk_xilinx\abc
Add to project
More Info
Next >
Cancel
Console
Console Errors Warnings Find in Files Results
Add a new source to the project
Type here to search
1:50 AM
5/24/2021

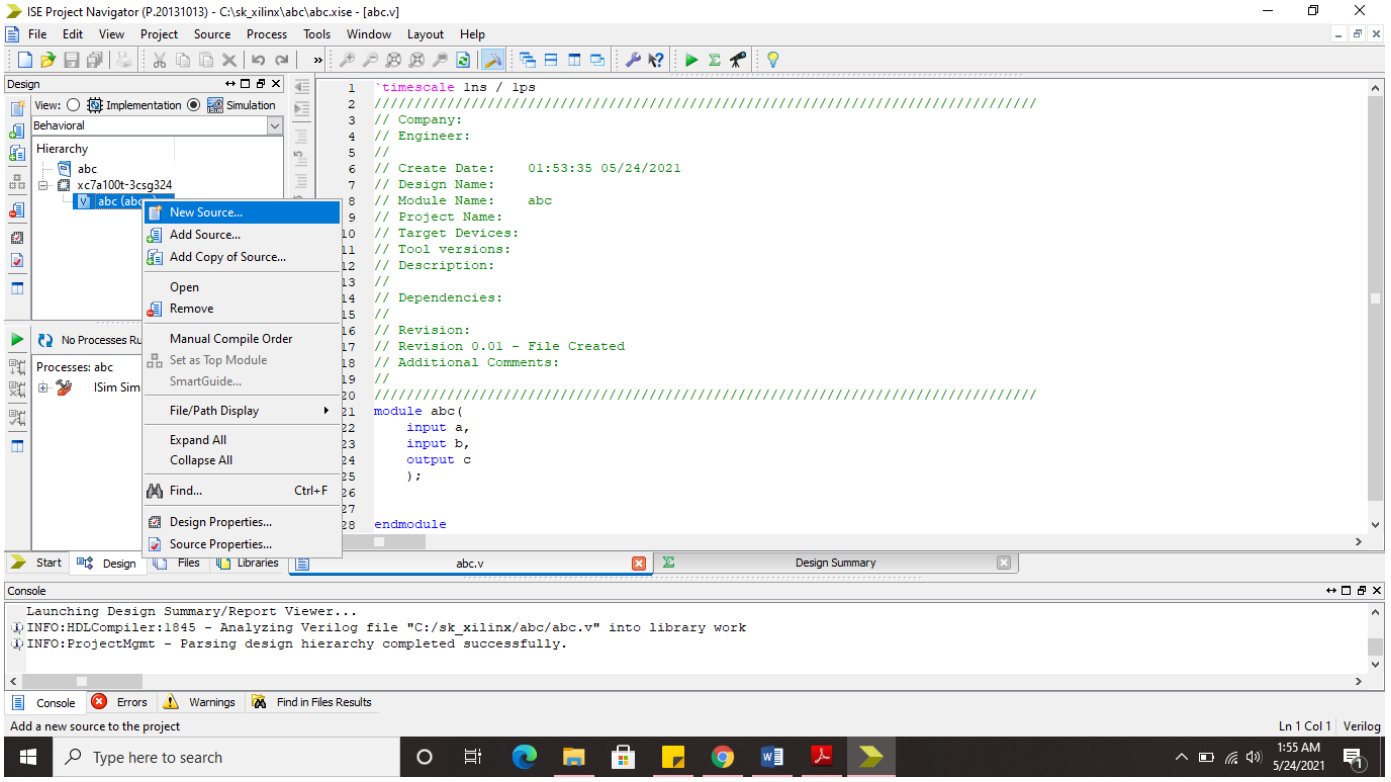

ISE Project Navigator (P.20131013) - C:\sk_xilinx\abc\abc.xise
File Edit View Project Source Process Tools Window Layout Help
Design
View: Implementation Simulation
Hierarchy
xc7a100t-3csg324
Empty View
The view currently contains no files. You can add files to the project using the toolbar at left, commands from the Project menu, and by using the Design, Files, and Libraries panels.
Use:
No Processes Running
No single design module is selected.
Design Utilities
Start Design Files Libraries
New Source Wizard
Define Module
Specify ports for module.
Module name abc
Port Name Direction Bus MSB LSB
input
More Info
< Back Next > Cancel
Console
Console Errors Warnings Find in Files Results
Add a new source to the project
Type here to search
1:51 AM
5/24/2021

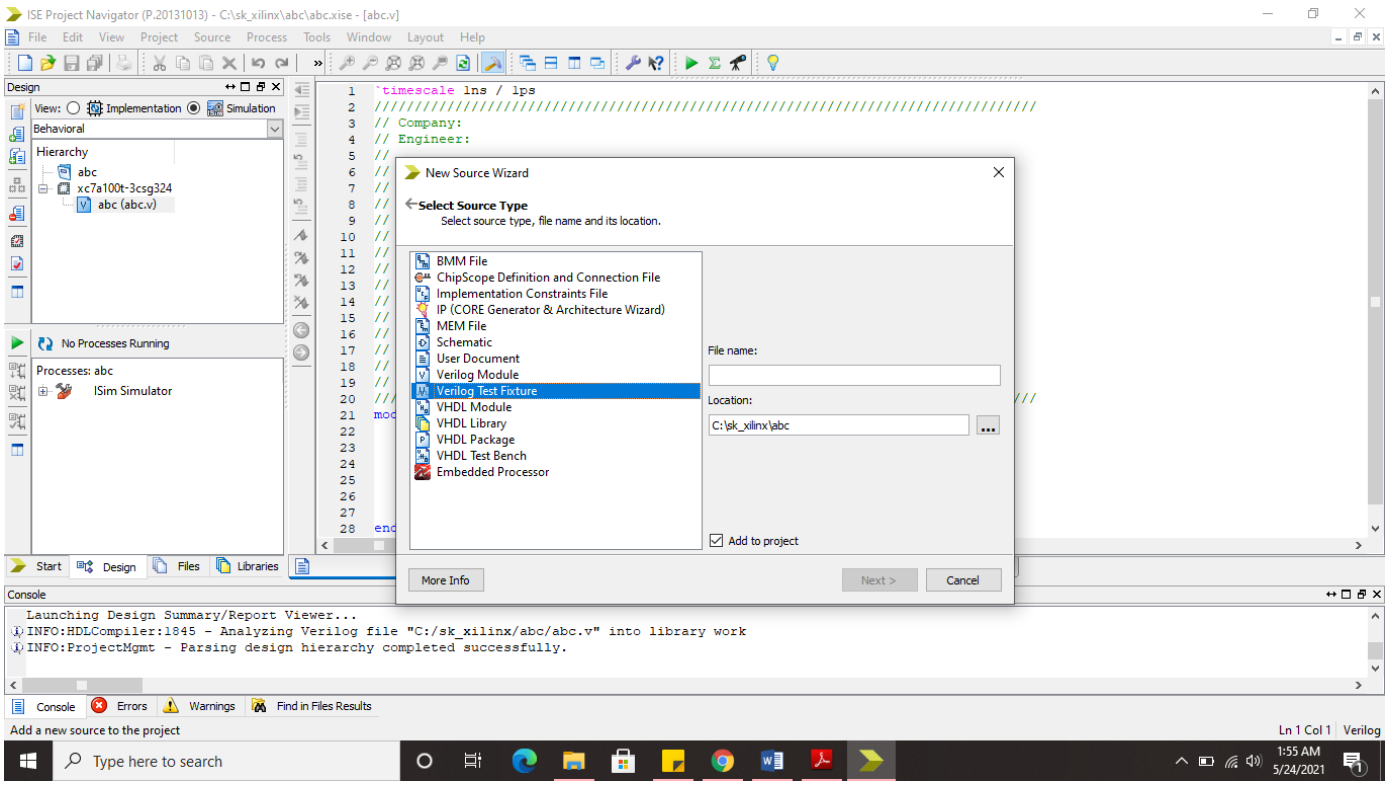

ISE Project Navigator (P.20131013) - C:\sk_xilinx\abc\abc.xise
File Edit View Project Source Process Tools Window Layout Help
Design
View: Implementation Simulation
Hierarchy
xc7a100t-3csg324
Empty View
The view currently contains no files. You can add files to the project using the toolbar at left, commands from the Project menu, and by using the Design, Files, and Libraries panels.
Use:
No Processes Running
No single design module is selected.
Design Utilities
Start Design Files Libraries
New Source Wizard
Define Module
Specify ports for module.
Module name abc
Port Name Direction Bus MSB LSB
a input
b input
c input
input
output
inout
More Info
< Back Next > Cancel
Console
Console Errors Warnings Find in Files Results
Add a new source to the project
Type here to search
1:52 AM
5/24/2021

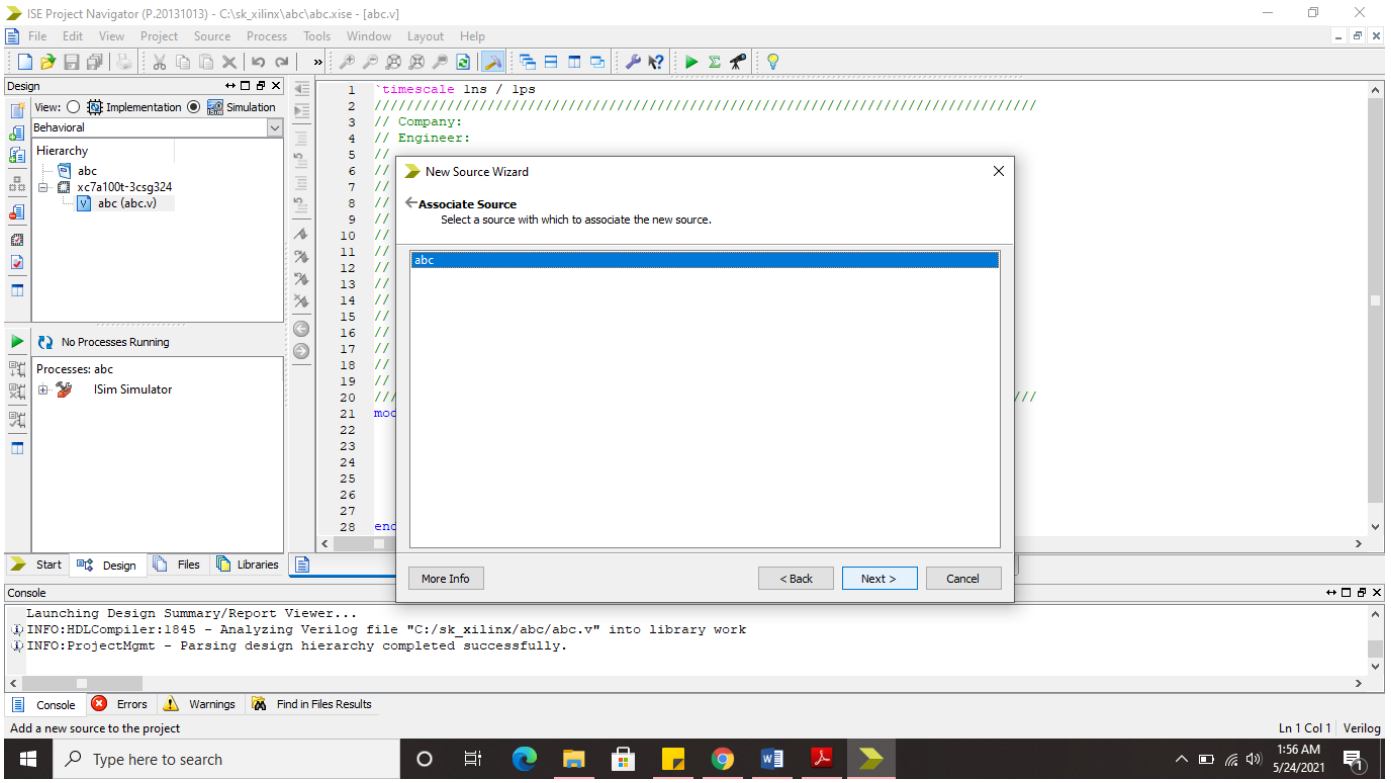

ISE Project Navigator (P.20131013) - C:\sk_xilinx\abc\abc.xise
File Edit View Project Source Process Tools Window Layout Help
Design
View: Implementation Simulation
Hierarchy
xc7a100t-3csg324
Empty View
The view currently contains no files. You can add files to the project using the toolbar at left, commands from the Project menu, and by using the Design, Files, and Libraries panels.
Use:
No Processes Running
No single design module is selected.
Design Utilities
Start Design Files Libraries
New Source Wizard
Summary
Project Navigator will create a new skeleton source with the following specifications.
Add to Project: Yes
Source Directory: C:\sk_xilinx\abc
Source Type: Verilog Module
Source Name: abc.v
Module name: abc
Port Definitions:
a Pin input
b Pin input
c Pin output
More Info
< Back
Finish
Cancel
Console
Console Errors Warnings Find in Files Results
Add a new source to the project
Type here to search
1:52 AM
5/24/2021

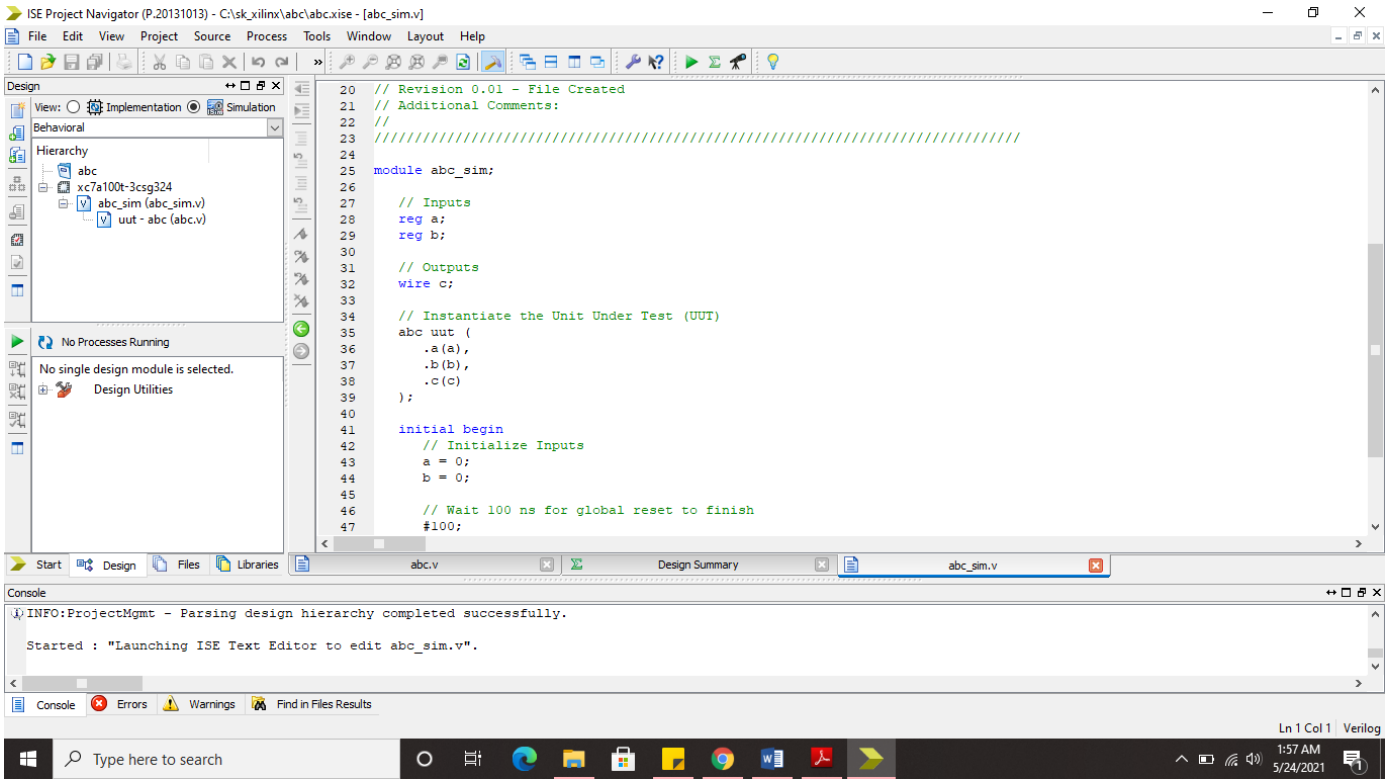

ISE Project Navigator (P.20131013) - C:\sk_xilinx\abc\abc.xise - [abc.v]
File Edit View Project Source Process Tools Window Layout Help
Design
View: Implementation Simulation
Hierarchy
abc
xc7a100t-3csg324
abc (abc.v)
No Processes Running
Processes: abc
Design Summary/Reports
Design Utilities
User Constraints
Synthesize - XST
Implement Design
Generate Programming File
Configure Target Device
Analyze Design Using ChipScope
Start Design Files Libraries
abc.v
Design Summary
`timescale 1ns / 1ps
//
// Company:
// Engineer:
//
// Create Date: 01:53:35 05/24/2021
// Design Name:
// Module Name: abc
// Project Name:
// Target Devices:
// Tool versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//
module abc(
input a,
input b,
output c
);
endmodule
Console
Started : "Launching ISE Text Editor to edit abc.v".
Launching Design Summary/Report Viewer...
Console Errors Warnings Find in Files Results
Ln 1 Col 1 Verilog
Type here to search
1:53 AM
5/24/2021

Create simulation File

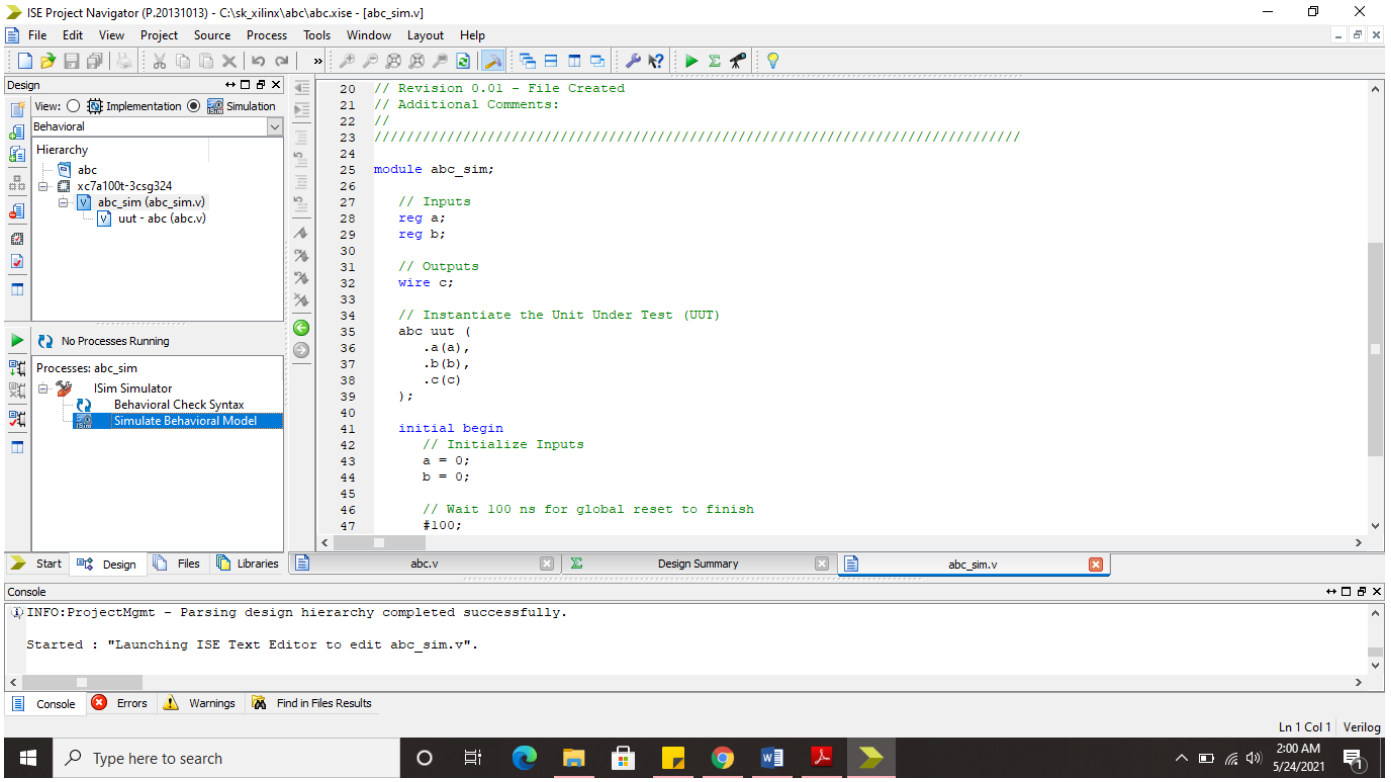




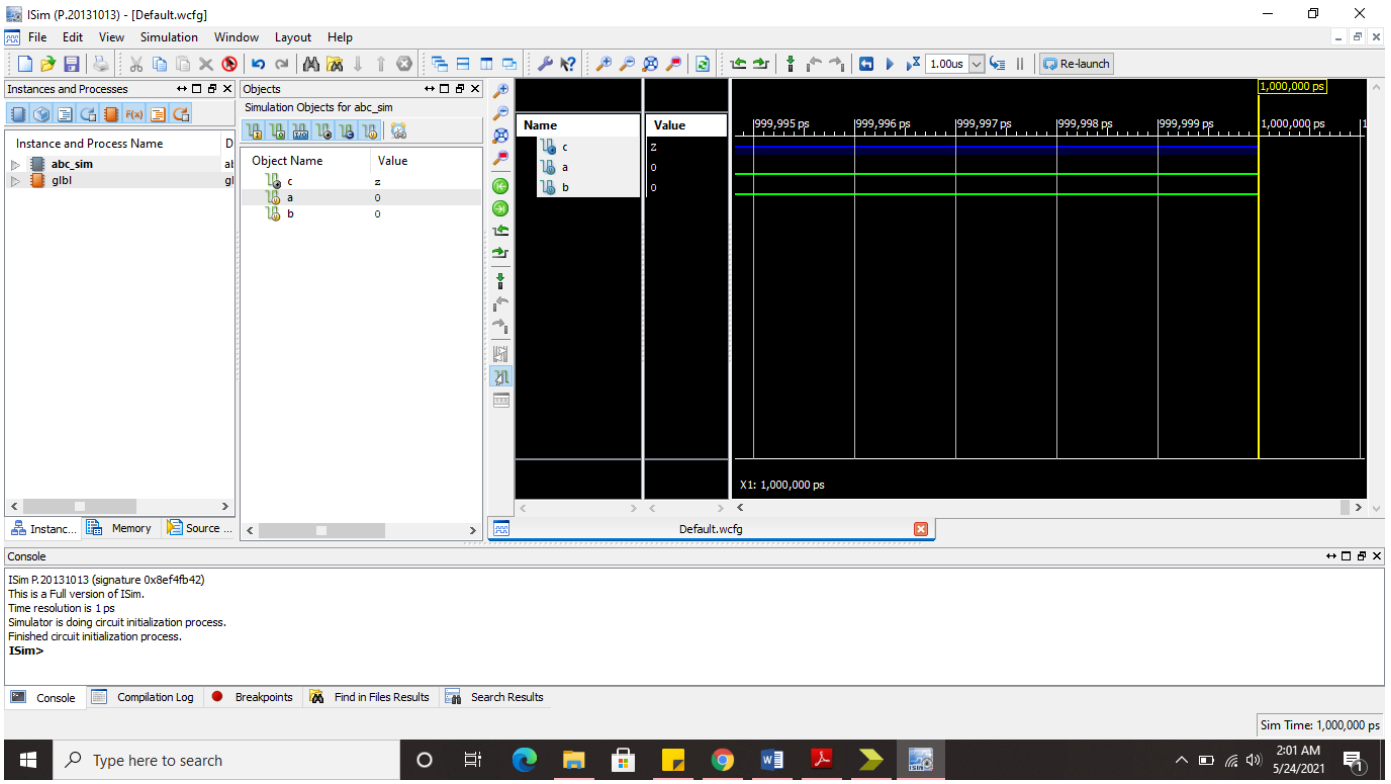

ISE Project Navigator (P.20131013) - C:\sk_xilinx\abc\abc.xise - [abc.v]
File Edit View Project Source Process Tools Window Layout Help
Design
View: Implementation Simulation
Behavioral
Hierarchy
abc
xc7a100t-3csg324
abc (abc.v)
No Processes Running
Processes: abc
ISim Simulator
Start Design Files Libraries
New Source Wizard
Associate Source
Select a source with which to associate the new source.
abc
More Info
< Back
Next >
Cancel
1 `timescale 1ns / 1ps
2 //
3 // Company:
4 // Engineer:
5 //
6 //
7 //
8 //
9 //
10 //
11 //
12 //
13 //
14 //
15 //
16 //
17 //
18 //
19 //
20 ///
21 mod
22
23
24
25
26
27
28 end
Console
Launching Design Summary/Report Viewer...
INFO:HDLCompiler:1845 - Analyzing Verilog file "C:/sk_xilinx/abc/abc.v" into library work
INFO:ProjectMgmt - Parsing design hierarchy completed successfully.
Console Errors Warnings Find in Files Results
Add a new source to the project
Ln 1 Col 1 Verilog
Type here to search
1:56 AM
5/24/2021


ISE Project Navigator (P.20131013) - C:\sk_xilinx\abc\abc.xise - [abc_sim.v]
File Edit View Project Source Process Tools Window Layout Help
Design
View: Implementation Simulation
Behavioral
Hierarchy
abc
xc7a100t-3csg324
abc_sim (abc_sim.v)
uut - abc (abc.v)
No Processes Running
No single design module is selected.
Design Utilities
Start Design Files Libraries
abc.v
Design Summary
abc_sim.v
20 // Revision 0.01 - File Created
21 // Additional Comments:
22 //
23 //
24
25 module abc_sim;
26
27 // Inputs
28 reg a;
29 reg b;
30
31 // Outputs
32 wire c;
33
34 // Instantiate the Unit Under Test (UUT)
35 abc uut (
36 .a(a),
37 .b(b),
38 .c(c)
39);
40
41 initial begin
42 // Initialize Inputs
43 a = 0;
44 b = 0;
45
46 // Wait 100 ns for global reset to finish
47 #100;
Console
INFO:ProjectMgmt - Parsing design hierarchy completed successfully.
Started : "Launching ISE Text Editor to edit abc_sim.v".
Console Errors Warnings Find in Files Results
Ln 1 Col 1 Verilog
Type here to search
1:57 AM
5/24/2021

Run the Simulation File



Output :



Experiment No.1

Familiarization with fundamental logic gates such as AND, OR, NOT, NAND, NOR,XOR using Verilog.

TWO INPUT AND GATE:-

Implementation Code:-

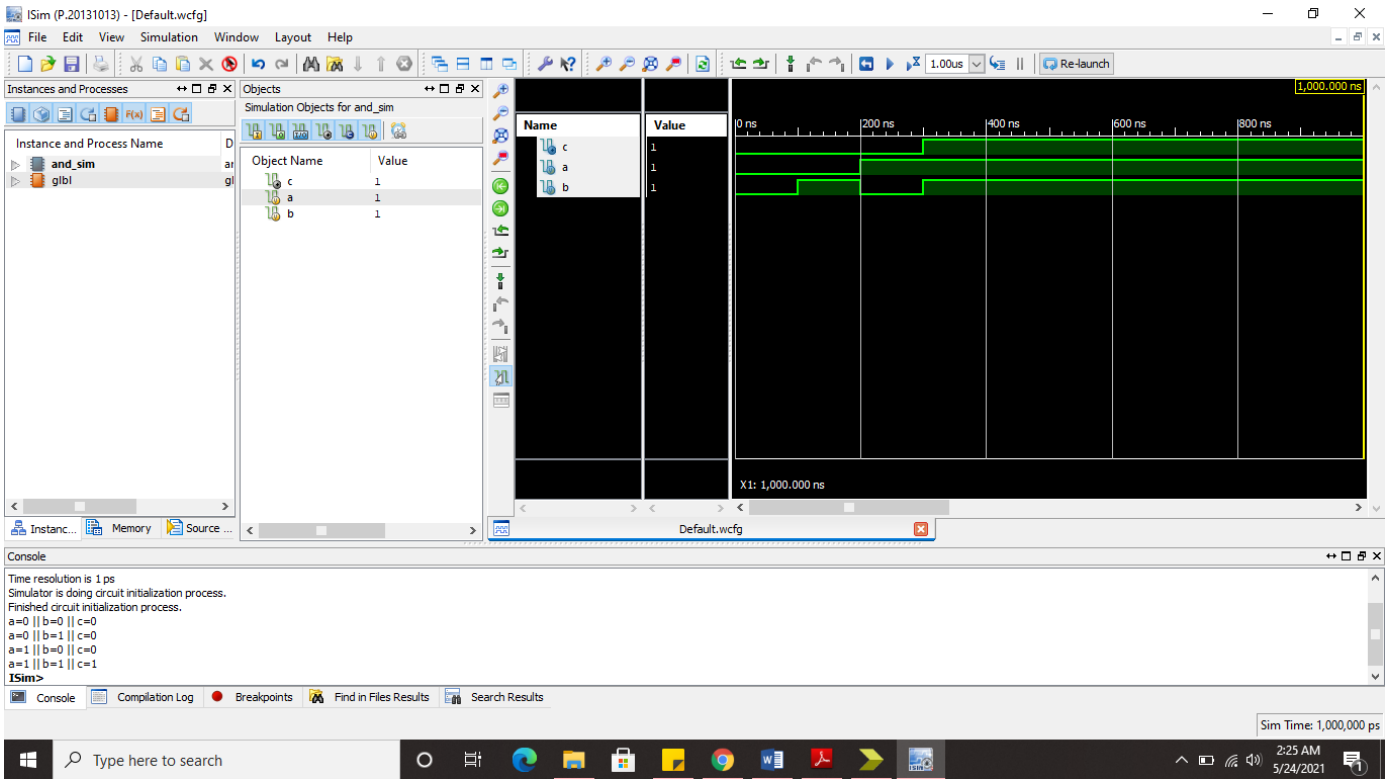
```
module and_gate(
   input a,
   input b,
   output c
   );
and(c,a,b);

endmodule
```

Simulation Code:-

```
module and_sim;

	// Inputs
	reg a;
	reg b;

	// Outputs
	wire c;

	// Instantiate the Unit Under Test (UUT)
	and_gate uut (
		.a(a),
		.b(b),
		.c(c)
	);

	initial begin
		// Initialize Inputs
		a = 0;
		b = 0;

		// Wait 100 ns for global reset to finish
		#100;
		b=1;
		#100;
		a=1;
		b=0;
		#100;
		b=1;
```

```
        // Add stimulus here

    end
    initial
    $monitor("a=%d || b=%d || c=%d",a,b,c);

endmodule
```

Output:-



TWO INPUT OR GATE:-

Implementation Code:-

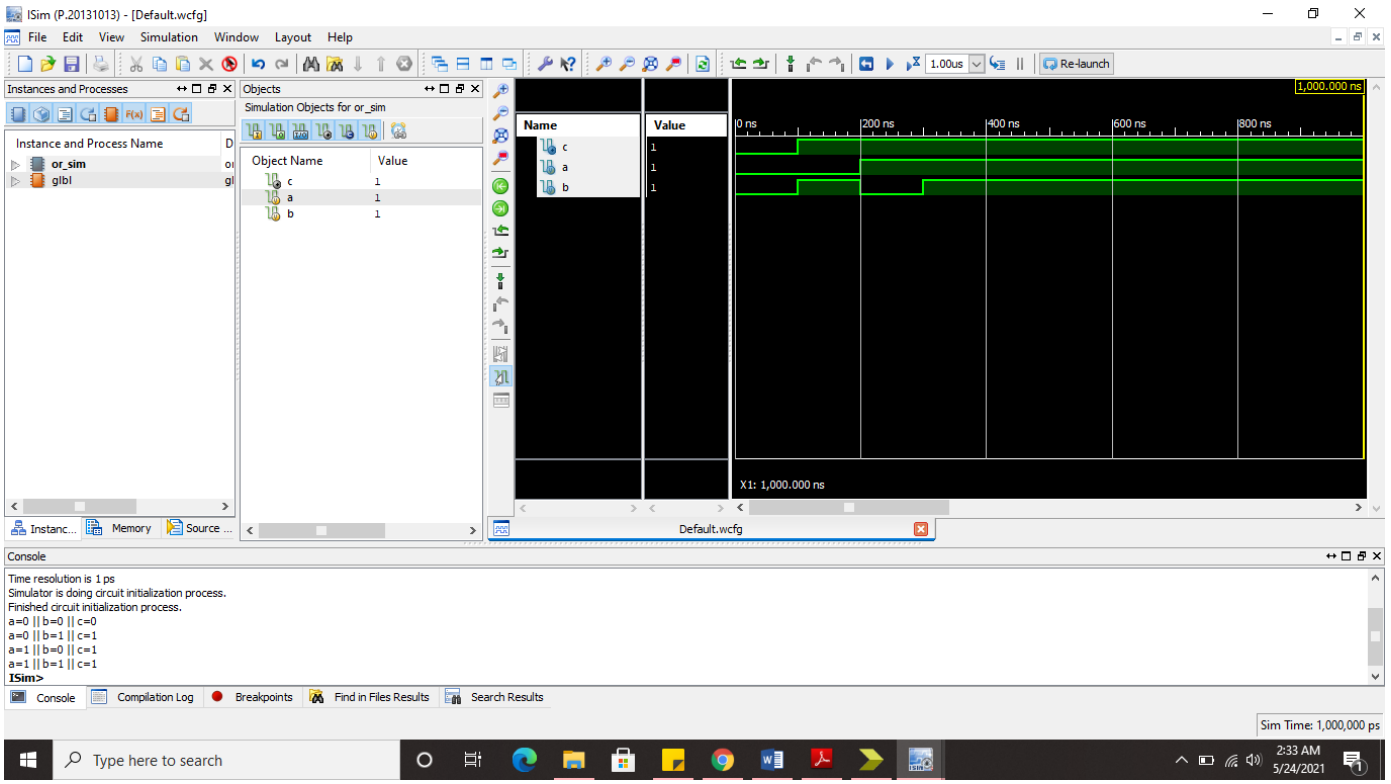
```
module or_gate(
   input a,
   input b,
   output c
   );
or(c,a,b);

endmodule
```

Simulation Code:-

```
module or_sim;

	// Inputs
	reg a;
	reg b;

	// Outputs
	wire c;

	// Instantiate the Unit Under Test (UUT)
	or_gate uut (
		.a(a),
		.b(b),
		.c(c)
	);

	initial begin
		// Initialize Inputs
		a = 0;
		b = 0;

		// Wait 100 ns for global reset to finish
		#100;
		b=1;
		#100;
		a=1;
		b=0;
		#100;
		b=1;

		// Add stimulus here

	end
	initial
	$monitor("a=%d || b=%d || c=%d",a,b,c);
```

Endmodule

Output:-



NOT GATE:-

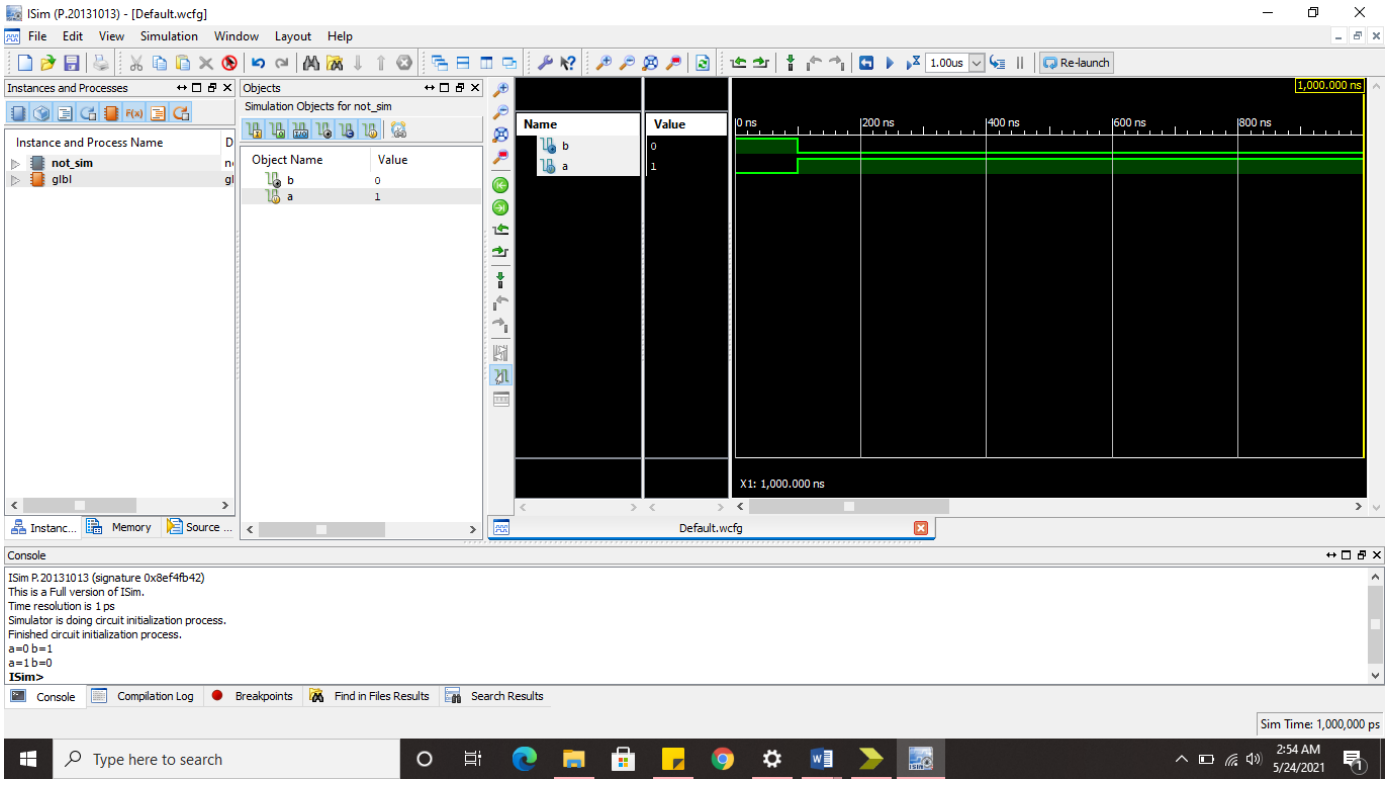
Implementation Code:-

```
module not_gate(
   input a,
   output b
   );
not(b,a);

endmodule
```

Simulation Code:-

```
module not_sim;

	// Inputs
	reg a;

	// Outputs
	wire b;

	// Instantiate the Unit Under Test (UUT)
	not_gate uut (
		.a(a),
		.b(b)
	);

	initial begin
		// Initialize Inputs
		a = 0;

		// Wait 100 ns for global reset to finish
		#100;
		a=1;

		// Add stimulus here

	end
    initial
		$monitor("a=%d b=%d",a,b);
endmodule
```

Output:-


ISim (P.20131013) - [Default.wcfg]
File Edit View Simulation Window Layout Help
Re-launch
Instances and Processes
Instance and Process Name
not_sim
glbl
Objects
Simulation Objects for not_sim
Object Name
Value
b
0
a
1
Name
Value
0 ns
200 ns
400 ns
600 ns
800 ns
1,000.000 ns
X1: 1,000.000 ns
Default.wcfg
Console
ISim P.20131013 (signature 0x8ef4fb42)
This is a Full version of ISim.
Time resolution is 1 ps
Simulator is doing circuit initialization process.
Finished circuit initialization process.
a=0 b=1
a=1 b=0
ISim>
Console
Compilation Log
Breakpoints
Find in Files Results
Search Results
Sim Time: 1,000,000 ps
Type here to search
2:54 AM
5/24/2021

TWO INPUT NAND GATE:-

Implementation Code:-

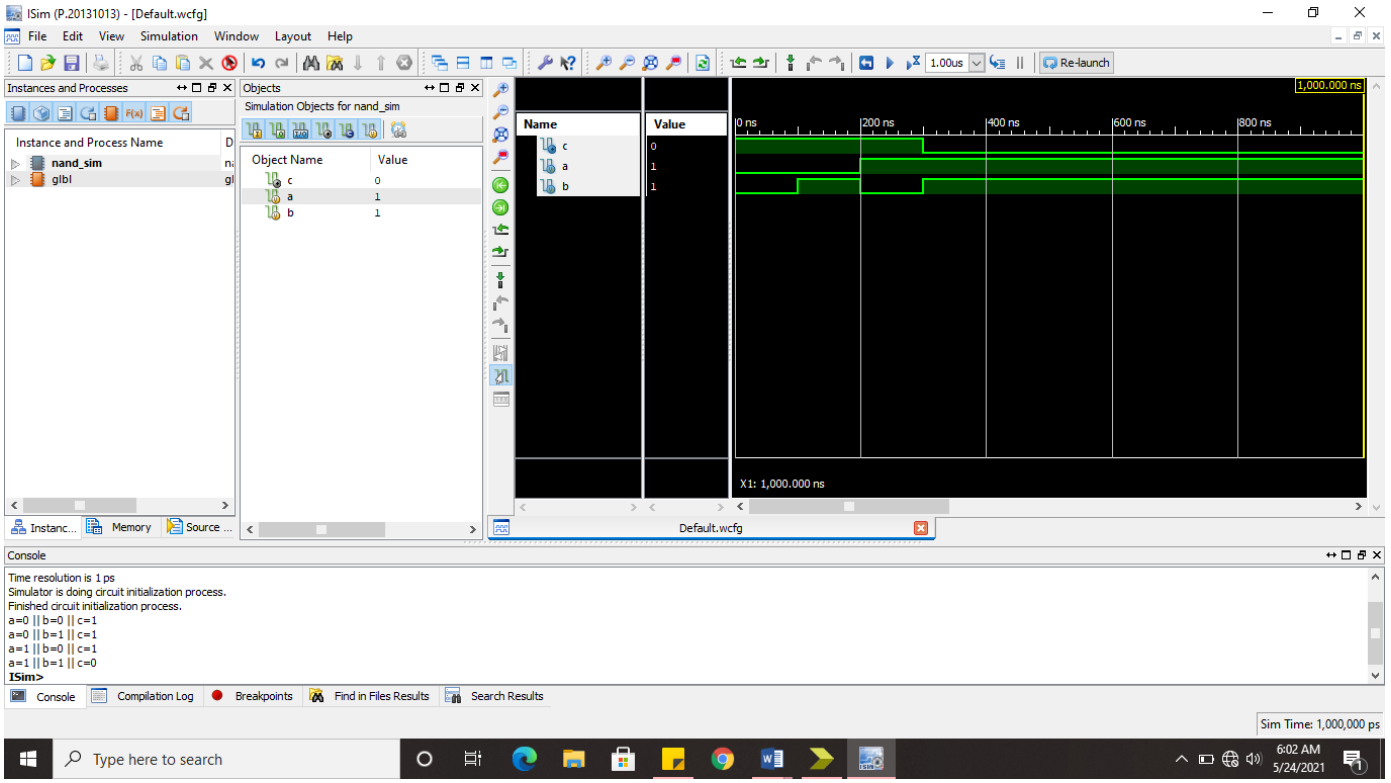
```
module nand_gate(
   input a,
   input b,
   output c
   );
nand(c,a,b);

endmodule
```

Simulation Code:-

```
module nand_sim;

	// Inputs
	reg a;
	reg b;

	// Outputs
	wire c;

	// Instantiate the Unit Under Test (UUT)
	nand_gate uut (
		.a(a),
		.b(b),
		.c(c)
	);

	initial begin
		// Initialize Inputs
```

```
        a = 0;
        b = 0;

        // Wait 100 ns for global reset to finish
        #100;
        b=1;
        #100;
        a=1;
        b=0;
        #100;
        b=1;

        // Add stimulus here

    end
    initial
    $monitor("a=%d || b=%d || c=%d",a,b,c);

endmodule
```

Output:-


ISim (P.20131013) - [Default.wcfg]
File Edit View Simulation Window Layout Help
Re-launch
Instances and Processes
Instance and Process Name
nand_sim
glbl
Objects
Simulation Objects for nand_sim
Object Name
Value
c 0
a 1
b 1
Name
Value
0 ns
200 ns
400 ns
600 ns
800 ns
1,000.000 ns
X1: 1,000.000 ns
Default.wcfg
Console
Time resolution is 1 ps
Simulator is doing circuit initialization process.
Finished circuit initialization process.
a=0 || b=0 || c=1
a=0 || b=1 || c=1
a=1 || b=0 || c=1
a=1 || b=1 || c=0
ISim>
Console
Compilation Log
Breakpoints
Find in Files Results
Search Results
Sim Time: 1,000,000 ps
Type here to search
6:02 AM
5/24/2021

TWO INPUT NOR GATE:-

Implementation Code:-

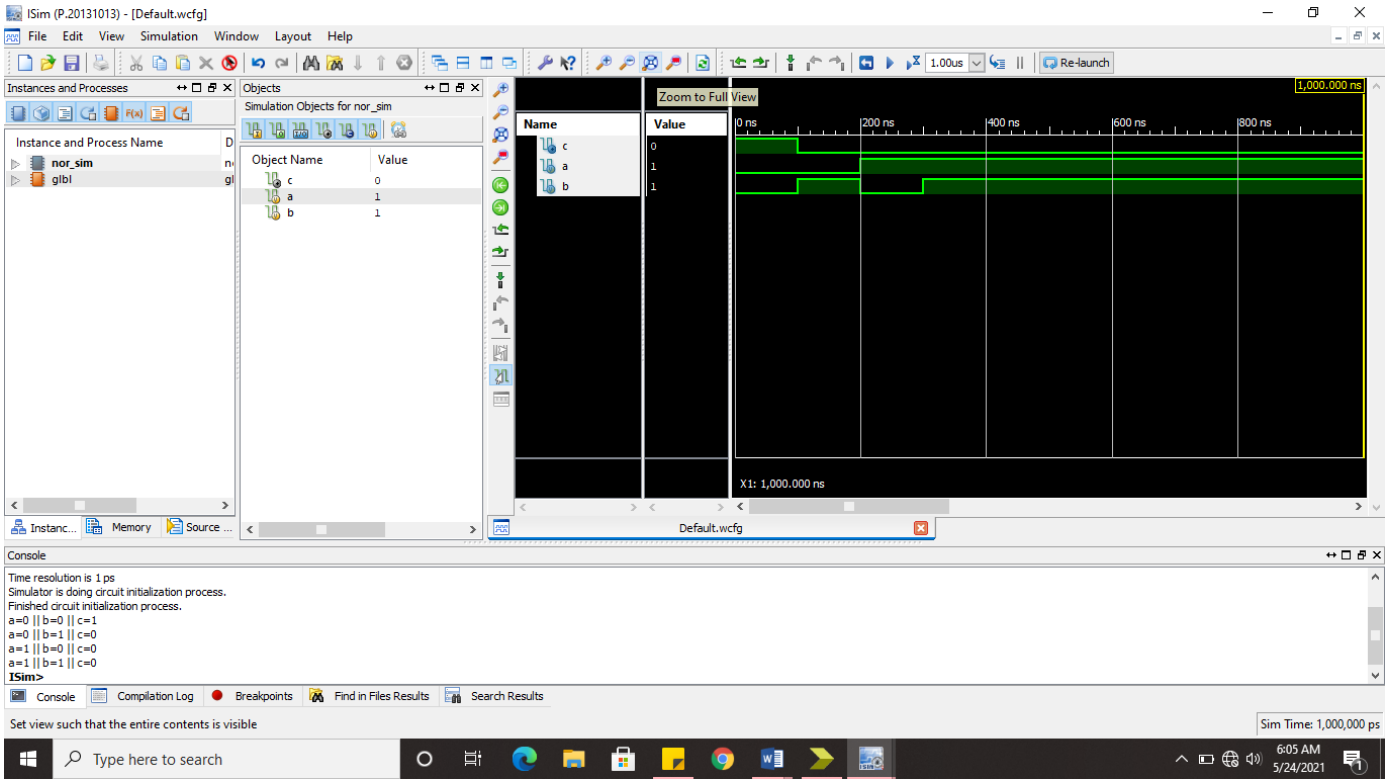
```
module nor_gate(
   input a,
   input b,
   output c
   );
nor(c,a,b);

endmodule
```

Simulation Code:-

```
module nor_sim;

	// Inputs
	reg a;
	reg b;

	// Outputs
	wire c;

	// Instantiate the Unit Under Test (UUT)
	nor_gate uut (
		.a(a),
		.b(b),
		.c(c)
	);

	initial begin
		// Initialize Inputs
```

```
        a = 0;
        b = 0;

        // Wait 100 ns for global reset to finish
        #100;
        b=1;
        #100;
        a=1;
        b=0;
        #100;
        b=1;

        // Add stimulus here

    end
    initial
    $monitor("a=%d || b=%d || c=%d",a,b,c);

endmodule
```

Output:-


ISim (P.20131013) - [Default.wcfg]
File Edit View Simulation Window Layout Help
Re-launch
Instances and Processes
Instance and Process Name
nor_sim
glbl
Objects
Simulation Objects for nor_sim
Object Name
Value
c 0
a 1
b 1
Zoom to Full View
Name
Value
0 ns
200 ns
400 ns
600 ns
800 ns
1,000.000 ns
X1: 1,000.000 ns
Default.wcfg
Console
Time resolution is 1 ps
Simulator is doing circuit initialization process.
Finished circuit initialization process.
a=0 || b=0 || c=1
a=0 || b=1 || c=0
a=1 || b=0 || c=0
a=1 || b=1 || c=0
ISim>
Console
Compilation Log
Breakpoints
Find in Files Results
Search Results
Set view such that the entire contents is visible
Sim Time: 1,000,000 ps
Type here to search
6:05 AM
5/24/2021

TWO INPUT X-OR GATE:-

Implementation Code:-

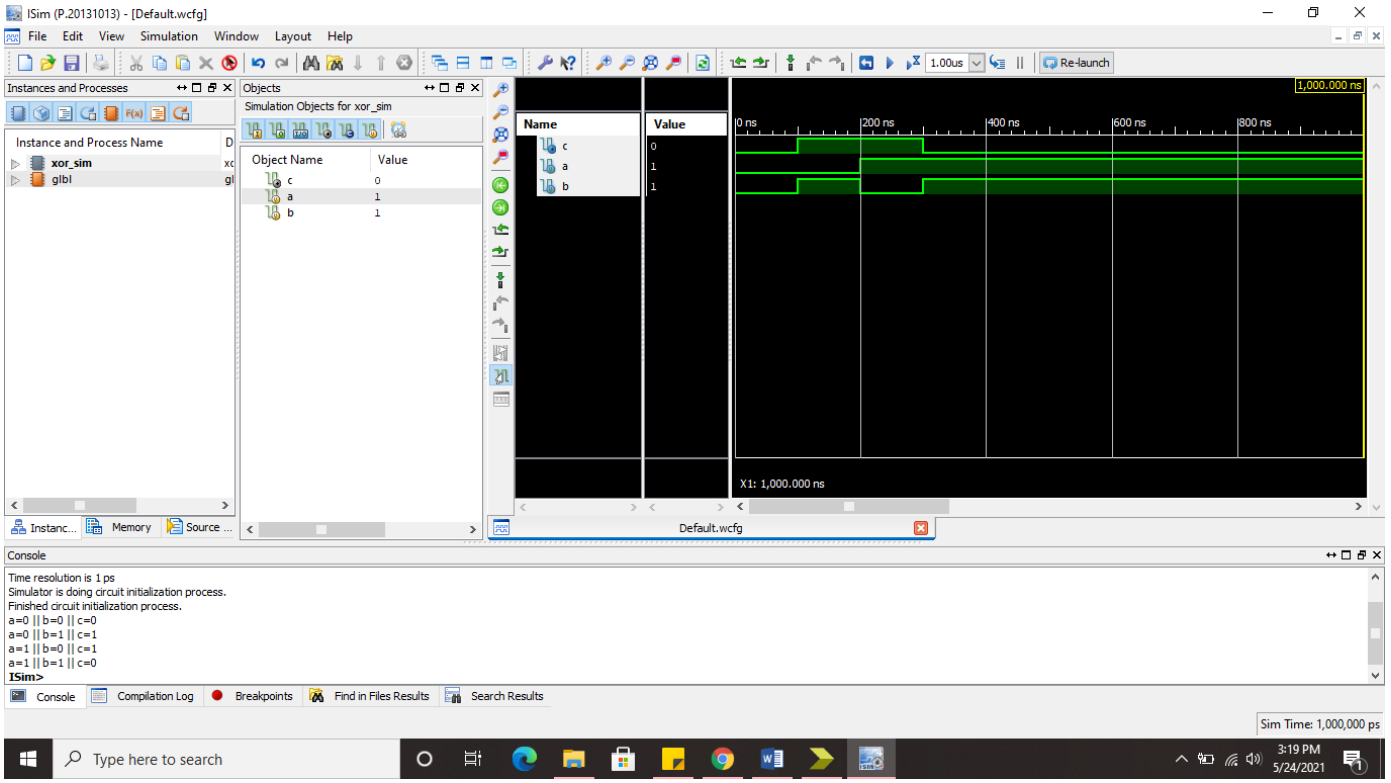
```
module xor_gate(
   input a,
   input b,
   output c
   );
xor(c,a,b);

endmodule
```

Simulation Code:-

```
module xor_sim;

	// Inputs
	reg a;
	reg b;

	// Outputs
	wire c;

	// Instantiate the Unit Under Test (UUT)
	xor_gate uut (
		.a(a),
		.b(b),
		.c(c)
	);

	initial begin
```

```
        // Initialize Inputs
        a = 0;
        b = 0;

        // Wait 100 ns for global reset to finish
        #100;
        b=1;
        #100;
        a=1;
        b=0;
        #100;
        b=1;

        // Add stimulus here

    end
    initial
    $monitor("a=%d || b=%d || c=%d",a,b,c);

endmodule
```

Output:-


ISim (P.20131013) - [Default.wcfg]
File Edit View Simulation Window Layout Help
Re-launch
Instances and Processes
Instance and Process Name
xor_sim
glbl
Objects
Simulation Objects for xor_sim
Object Name
Value
c
a
b
Name
Value
0 ns
200 ns
400 ns
600 ns
800 ns
1,000.000 ns
X1: 1,000.000 ns
Default.wcfg
Console
Time resolution is 1 ps
Simulator is doing circuit initialization process.
Finished circuit initialization process.
a=0 || b=0 || c=0
a=0 || b=1 || c=1
a=1 || b=0 || c=1
a=1 || b=1 || c=0
ISim>
Console
Compilation Log
Breakpoints
Find in Files Results
Search Results
Sim Time: 1,000,000 ps
Type here to search
3:19 PM
5/24/2021

Experiment No.2

Design of Half Adder & Full Adder circuit using Verilog.

HALF ADDER

Implementation Code:-

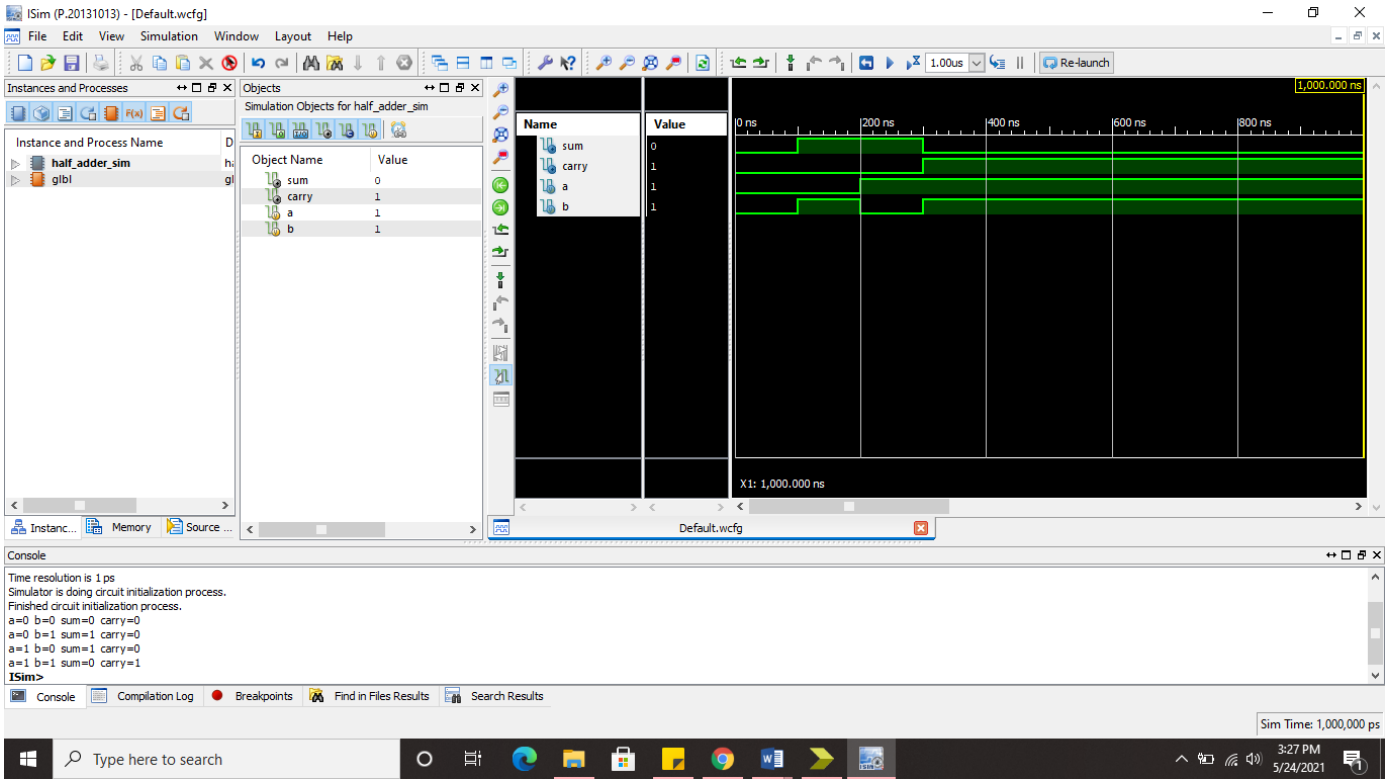
```
module half_adder(
   input a,
   input b,
   output sum,
   output carry
   );
xor(sum,a,b);
and(carry,a,b);

endmodule
```

Simulation Code:-

```
module half_adder_sim;

        // Inputs
        reg a;
        reg b;

        // Outputs
        wire sum;
        wire carry;

        // Instantiate the Unit Under Test (UUT)
        half_adder uut (
                .a(a),
```

```
		.b(b),
		.sum(sum),
		.carry(carry)
	);

	initial begin
		// Initialize Inputs
		a = 0;
		b = 0;

		// Wait 100 ns for global reset to finish
		#100;
		b=1;
		#100;
		a=1;
		b=0;
		#100;
		b=1;

		// Add stimulus here

	end
	initial
	$monitor("a=%d  b=%d  sum=%d  carry=%d",a,b,sum,carry);

endmodule
```

Output:-


ISim (P.20131013) - [Default.wcfg]
File Edit View Simulation Window Layout Help
Re-launch
Instances and Processes
Instance and Process Name
half_adder_sim
glbl
Objects
Simulation Objects for half_adder_sim
Object Name
Value
sum 0
carry 1
a 1
b 1
Name
Value
0 ns
200 ns
400 ns
600 ns
800 ns
1,000.000 ns
X1: 1,000.000 ns
Default.wcfg
Console
Time resolution is 1 ps
Simulator is doing circuit initialization process.
Finished circuit initialization process.
a=0 b=0 sum=0 carry=0
a=0 b=1 sum=1 carry=0
a=1 b=0 sum=1 carry=0
a=1 b=1 sum=0 carry=1
ISim>
Console
Compilation Log
Breakpoints
Find in Files Results
Search Results
Sim Time: 1,000,000 ps
Type here to search
3:27 PM
5/24/2021

FULL ADDER:-

Implementation Code:-

```
module full_adder(
   input a,
   input b,
   input c,
   inout d,
   output sum,
   inout f,
   inout g,
   output carry
   );
xor(d,a,b);
xor(sum,c,d);
and(f,a,b);
and(g,c,d);
or(carry,f,g);

endmodule
```

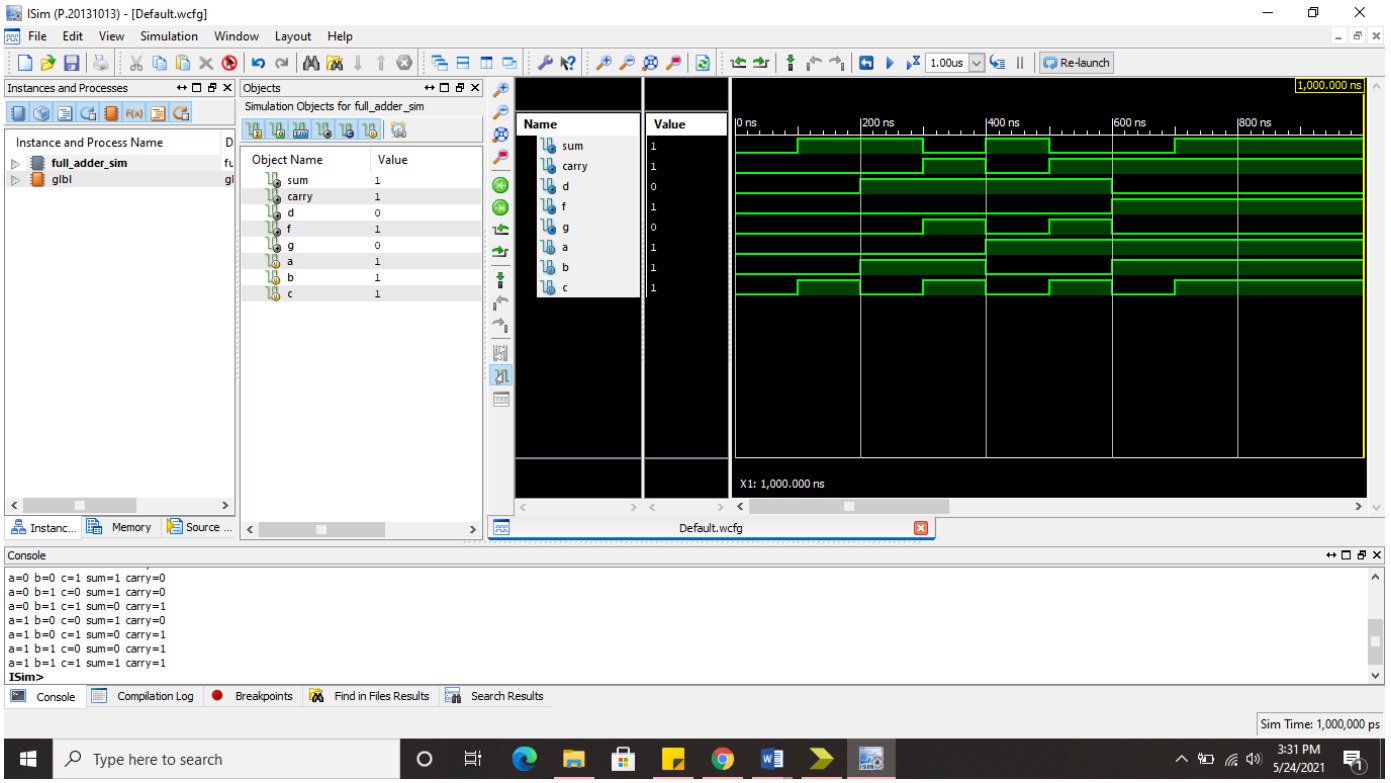
Simulation Code:-

```
module full_adder_sim;

        // Inputs
        reg a;
        reg b;
        reg c;

        // Outputs
        wire sum;
```

```
	wire carry;

	// Bidirs
	wire d;
	wire f;
	wire g;

	// Instantiate the Unit Under Test (UUT)
	full_adder uut (
		.a(a),
		.b(b),
		.c(c),
		.d(d),
		.sum(sum),
		.f(f),
		.g(g),
		.carry(carry)
	);

	initial begin
		// Initialize Inputs
		a = 0;
		b = 0;
		c = 0;

		// Wait 100 ns for global reset to finish
		#100;
		c=1;
		#100;
		b=1;
```

```
                c=0;
                #100;
                c=1;
                #100;
                a=1;
                b=0;
                c=0;
                #100;
                c=1;
                #100;
                b=1;
                c=0;
    #100;
                c=1;
                // Add stimulus here

        end
        initial
        $monitor("a=%d  b=%d  c=%d  sum=%d  carry=%d",a,b,c,sum,carry);

endmodule
```

Output:-


ISim (P.20131013) - [Default.wcfg]
File Edit View Simulation Window Layout Help
Re-launch
Instances and Processes
Instance and Process Name
full_adder_sim
glbl
Objects
Simulation Objects for full_adder_sim
Object Name Value
sum 1
carry 1
d 0
f 1
g 0
a 1
b 1
c 1
Name
Value
0 ns
200 ns
400 ns
600 ns
800 ns
1,000.000 ns
X1: 1,000.000 ns
Default.wcfg
Console
a=0 b=0 c=1 sum=1 carry=0
a=0 b=1 c=0 sum=1 carry=0
a=0 b=1 c=1 sum=0 carry=1
a=1 b=0 c=0 sum=1 carry=0
a=1 b=0 c=1 sum=0 carry=1
a=1 b=1 c=0 sum=0 carry=1
a=1 b=1 c=1 sum=1 carry=1
ISim>
Console
Compilation Log
Breakpoints
Find in Files Results
Search Results
Sim Time: 1,000,000 ps
Type here to search
3:31 PM
5/24/2021

Experiment No.3

Design of Half Subtractor circuit using Verilog.

HALF SUBTRACTOR

Implementation Code:-

```
module half_subtractor(
   input a,
   input b,
   output diff,
   output borr,
   inout c
   );
xor(diff,a,b);
not(c,a);
and(borr,b,c);


endmodule
```

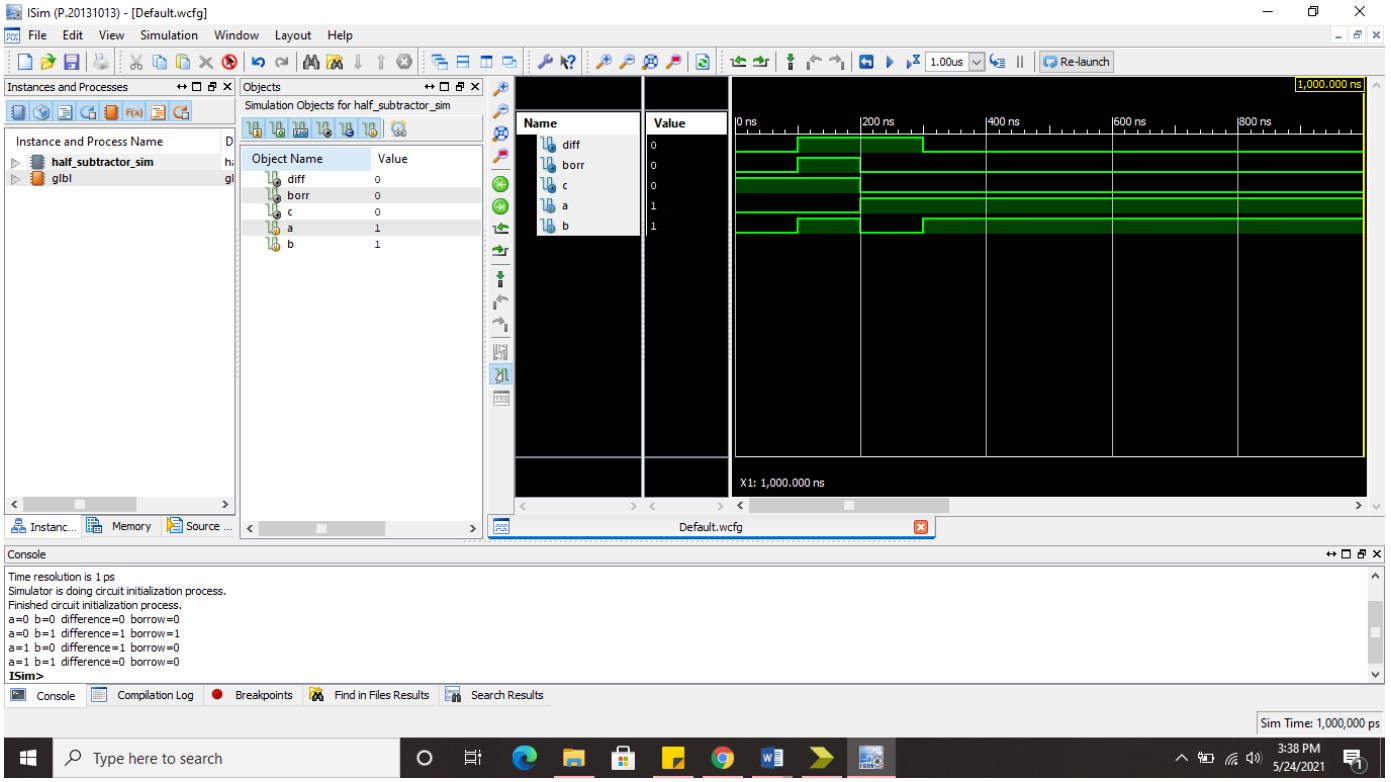
Simulation Code:-

```
module half_subtractor_sim;


        // Inputs
        reg a;
        reg b;


        // Outputs
        wire diff;
        wire borr;


        // Bidirs
```

```
wire c;


// Instantiate the Unit Under Test (UUT)
half_subtractor uut (
        .a(a),
        .b(b),
        .diff(diff),
        .borr(borr),
        .c(c)
);


initial begin
        // Initialize Inputs
        a = 0;
        b = 0;


        // Wait 100 ns for global reset to finish
        #100;
        b=1;
        #100;
        a=1;
        b=0;
        #100;
        b=1;


        // Add stimulus here


end
initial
$monitor("a=%d  b=%d  difference=%d  borrow=%d",a,b,diff,borr);
```

endmodule

Output:-



Experiment No.4

Solve De-Morgan's law using Verilog.

1ST LAW *Not* (**A** *and* **B**) is the same as *Not* **A** *or Not* **B**.

Implementation Code:-

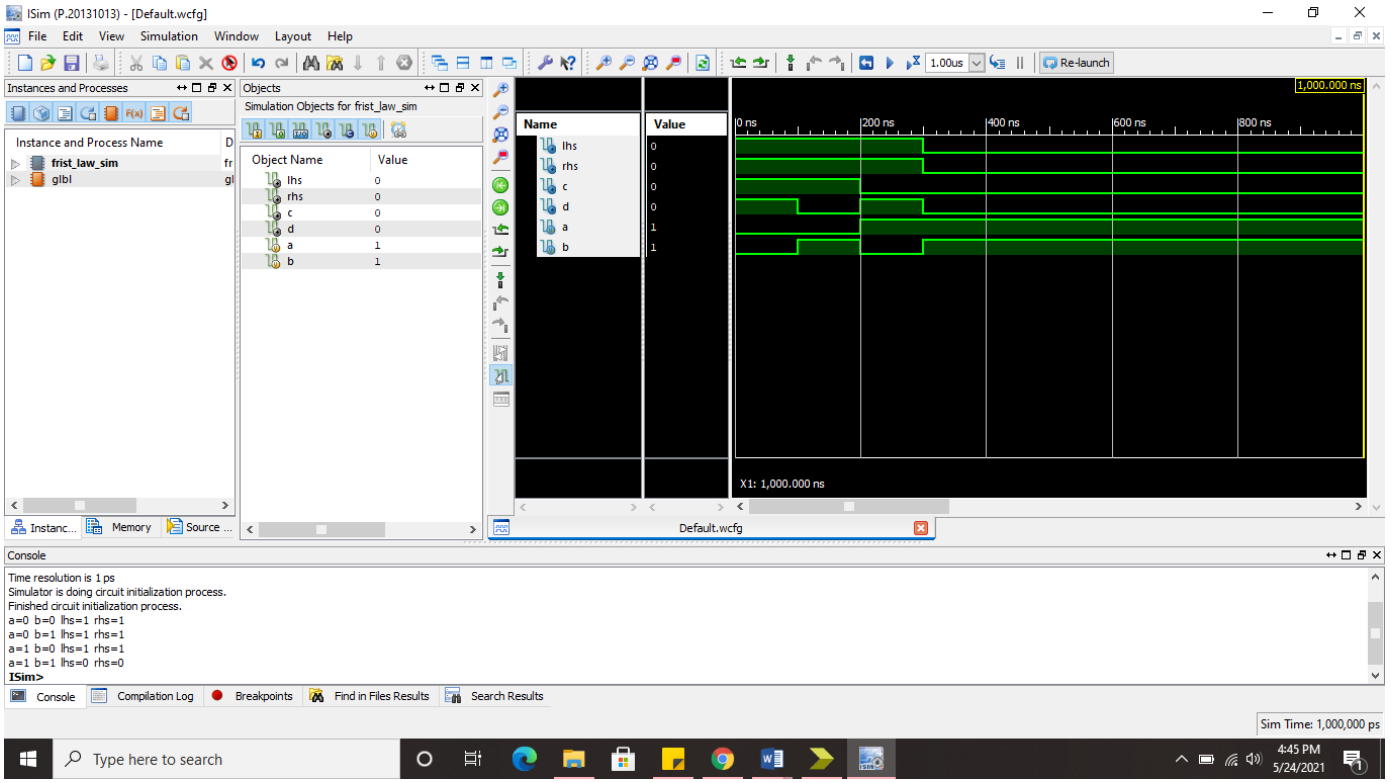
```
module first_law(
   input a,
   input b,
   inout c,
   inout d,
   output lhs,
   output rhs
   );
nand(lhs,a,b);
not(c,a);
not(d,b);
or(rhs,c,d);

endmodule
```

Simulation Code:-

```
module frist_law_sim;


        // Inputs

        reg a;

        reg b;


        // Outputs

        wire lhs;

        wire rhs;


        // Bidirs

        wire c;

        wire d;


        // Instantiate the Unit Under Test (UUT)
```

```
	first_law uut (
		.a(a),
		.b(b),
		.c(c),
		.d(d),
		.lhs(lhs),
		.rhs(rhs)
	);

	initial begin
		// Initialize Inputs
		a = 0;
		b = 0;

		// Wait 100 ns for global reset to finish
		#100;
		b=1;
		#100;
		a=1;
		b=0;
		#100;
		b=1;

		// Add stimulus here

	end
	initial
	$monitor("a=%d  b=%d  lhs=%d  rhs=%d",a,b,lhs,rhs);
endmodule
```

Output:-



2ND LAW *Not* (**A** *or* **B**) is the same as *Not* **A** *and Not* **B**

Implementation Code:-

```
module second_law(
   input a,
   input b,
   inout c,
   inout d,
   output lhs,
   output rhs
   );
nor(lhs,a,b);
not(c,a);
not(d,b);
```

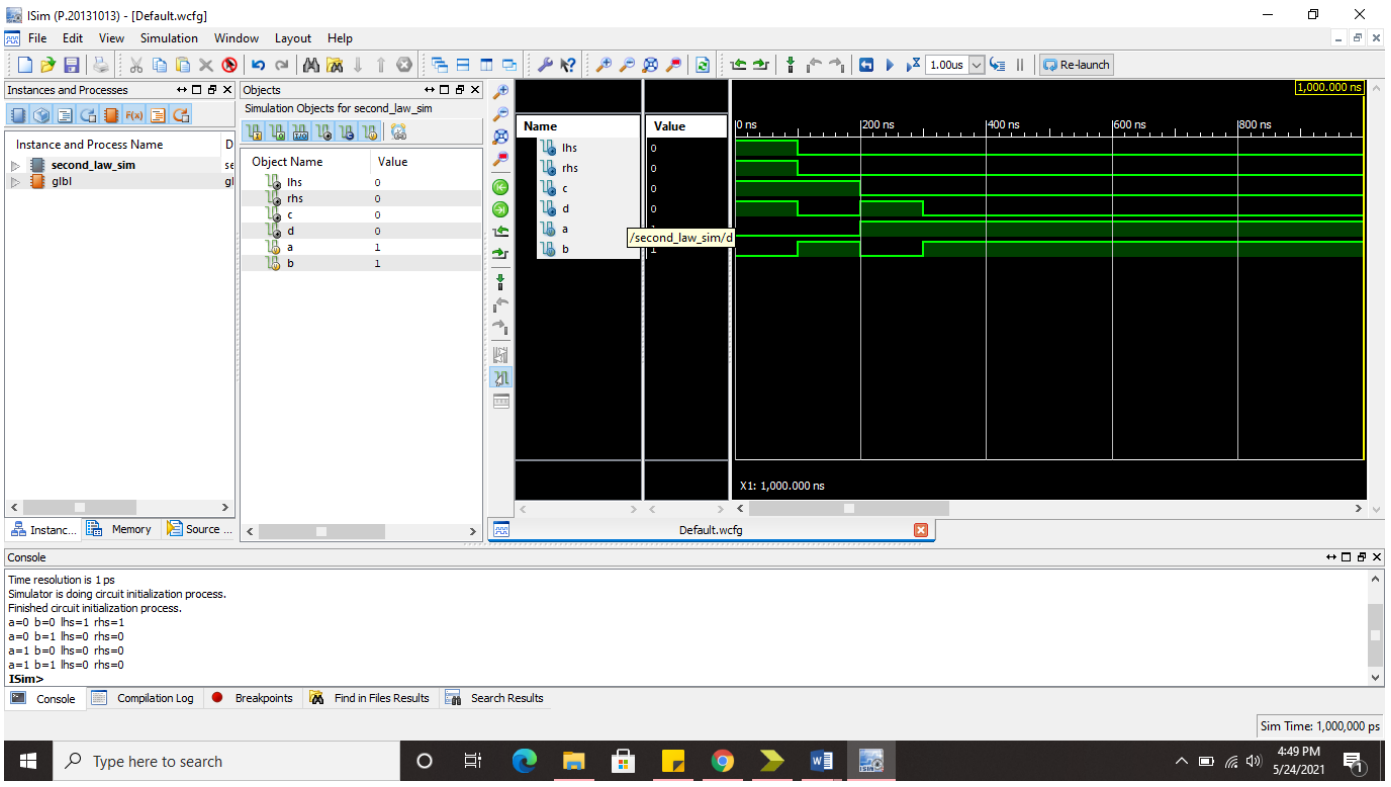
```
and(rhs,c,d);
endmodule
```

Simulation Code:-

```
module second_law_sim;

	// Inputs
	reg a;
	reg b;

	// Outputs
	wire lhs;
	wire rhs;

	// Bidirs
	wire c;
	wire d;

	// Instantiate the Unit Under Test (UUT)
	second_law uut (
		.a(a),
		.b(b),
		.c(c),
		.d(d),
		.lhs(lhs),
		.rhs(rhs)
	);

	initial begin
		// Initialize Inputs
```

```
        a = 0;
        b = 0;
        // Wait 100 ns for global reset to finish
        #100;
        b=1;
        #100;
        a=1;
        b=0;
        #100;
        b=1;
        // Add stimulus here
    end
    initial
    $monitor("a=%d  b=%d  lhs=%d  rhs=%d",a,b,lhs,rhs);
endmodule
```

Output:-



Experiment No.5

Simulation and verification of 8 bit adder circuit using Verilog.

8 BIT ADDER:-

Implementation Code:-

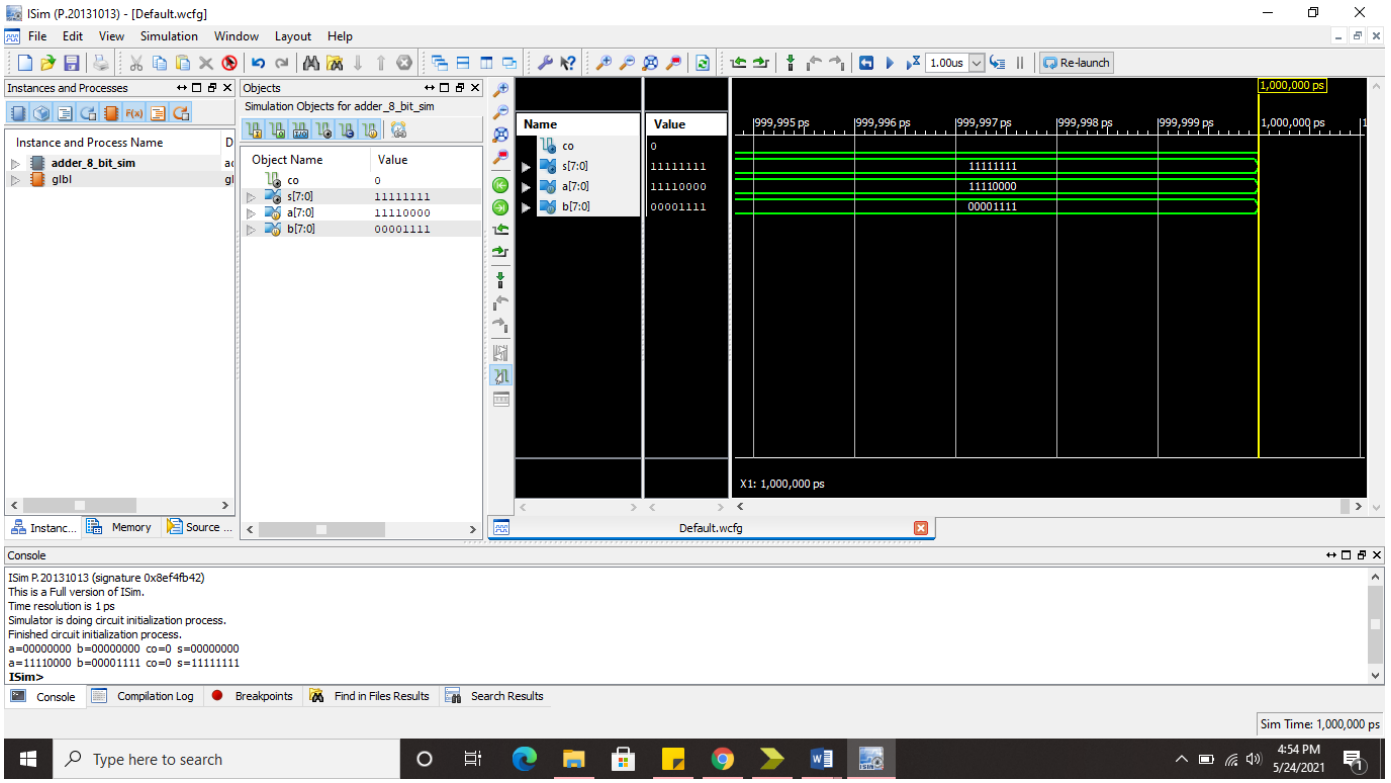
```
module adder_8_bit(
   input [7:0] a,
   input [7:0] b,
   output co,
   output [7:0] s
   );
            assign{co,s}=a+b;
endmodule
```

Simulation Code:-

```
module adder_8_bit_sim;

            // Inputs
            reg [7:0] a;
            reg [7:0] b;

            // Outputs
            wire co;
            wire [7:0] s;

            // Instantiate the Unit Under Test (UUT)
            adder_8_bit uut (
                        .a(a),
                        .b(b),
```

```
		.co(co),
		.s(s)
	);

	initial begin
		// Initialize Inputs
		a = 0;
		b = 0;

		// Wait 100 ns for global reset to finish
		#100;
    a=8'b11110000;
		  b=8'b00001111;
		// Add stimulus here

	end
	initial
	$monitor("a=%b  b=%b  co=%b  s=%b",a,b,co,s);

endmodule
```

Output:-


ISim (P.20131013) - [Default.wcfg]
File Edit View Simulation Window Layout Help
Re-launch
Instances and Processes
Instance and Process Name
adder_8_bit_sim
glbl
Objects
Simulation Objects for adder_8_bit_sim
Object Name
Value
co 0
s[7:0] 11111111
a[7:0] 11110000
b[7:0] 00001111
Name
Value
999,995 ps
999,996 ps
999,997 ps
999,998 ps
999,999 ps
1,000,000 ps
X1: 1,000,000 ps
Default.wcfg
Console
ISim P.20131013 (signature 0x8ef4fb42)
This is a Full version of ISim.
Time resolution is 1 ps
Simulator is doing circuit initialization process.
Finished circuit initialization process.
a=00000000 b=00000000 co=0 s=00000000
a=11110000 b=00001111 co=0 s=11111111
ISim>
Console
Compilation Log
Breakpoints
Find in Files Results
Search Results
Sim Time: 1,000,000 ps
Type here to search
4:54 PM
5/24/2021

Experiment No.6

Simulation and verification of 8 to 3 Decoder using Verilog.

8 TO 3 DECODER:-

Implementation Code:-

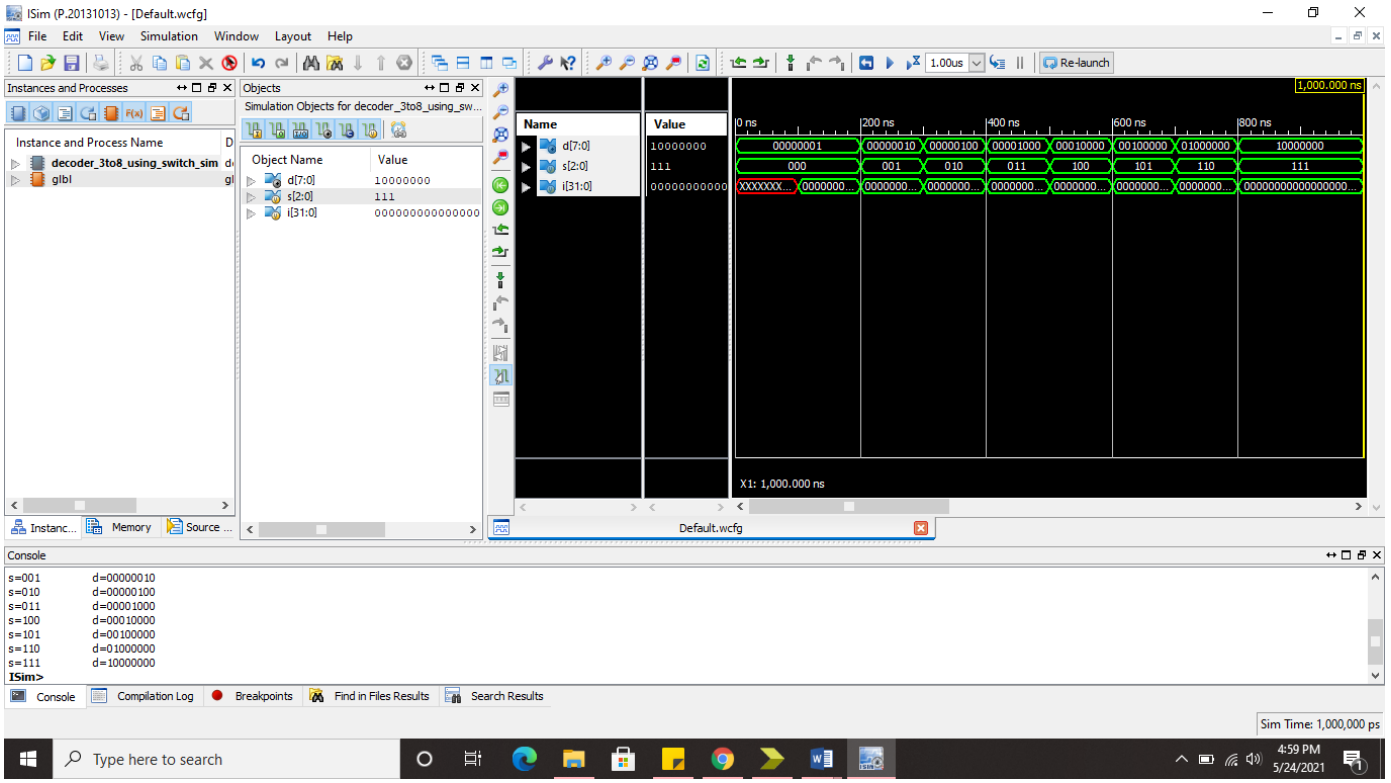
```
module decoder_3to8_using_switch(
   input [2:0] s,
   output reg [7:0] d
   );
            always@(s,d)
            case(s)
            3'b000 : d = 8'b00000001;
            3'b001 : d = 8'b00000010;
            3'b010 : d = 8'b00000100;
            3'b011 : d = 8'b00001000;
            3'b100 : d = 8'b00010000;
            3'b101 : d = 8'b00100000;
            3'b110 : d = 8'b01000000;
            3'b111 : d = 8'b10000000;
            default:$monitor("Wrong Display");
            endcase
endmodule
```

Simulation Code:-

```
module decoder_3to8_using_switch_sim;

            // Inputs
            reg [2:0] s;

            // Outputs
```

```
	wire [7:0] d;


	// Instantiate the Unit Under Test (UUT)
	decoder_3to8_using_switch uut (
		.s(s),
		.d(d)
	);
	integer i;
	initial begin
		// Initialize Inputs
		s = 0;


		// Wait 100 ns for global reset to finish
		#100;
		for(i=1;i<8;i=i+1)
		#100
		s=i;


		// Add stimulus here
	end
initial $monitor("s=%b\t d=%b",s,d);
endmodule
```

Output:-


ISim (P.20131013) - [Default.wcfg]
File Edit View Simulation Window Layout Help
Re-launch
Instances and Processes
Objects
Simulation Objects for decoder_3to8_using_sw...
Instance and Process Name
decoder_3to8_using_switch_sim
glbl
Object Name
Value
d[7:0]
10000000
s[2:0]
111
i[31:0]
00000000000000
Name
Value
1,000.000 ns
0 ns
200 ns
400 ns
600 ns
800 ns
00000001
00000010
00000100
00001000
00010000
00100000
01000000
10000000
000
001
010
011
100
101
110
111
X1: 1,000.000 ns
Default.wcfg
Console
s=001 d=00000010
s=010 d=00000100
s=011 d=00001000
s=100 d=00010000
s=101 d=00100000
s=110 d=01000000
s=111 d=10000000
ISim>
Console
Compilation Log
Breakpoints
Find in Files Results
Search Results
Sim Time: 1,000,000 ps
Type here to search
4:59 PM
5/24/2021

Experiment No.7

Simulation and verification of 4 to 1 Multiplexer using Verilog.

4 TO 1 MULTIPLEXER

Implementation Code:-

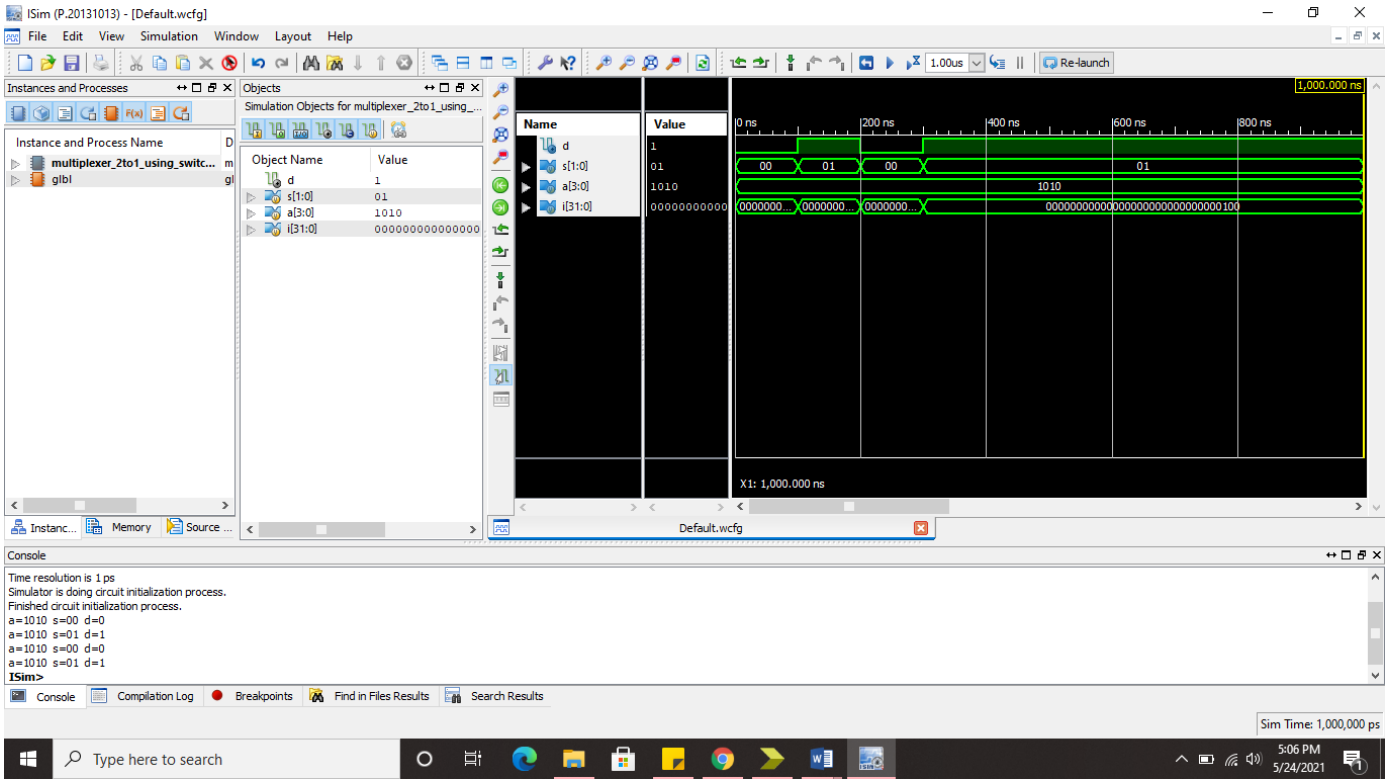
```
module multiplexer_2t01_using_switch(
   input [1:0] s,
   input [3:0] a,
   output reg d
   );
            always@(s)
            case(s)
            0:d=a[0];
            1:d=a[1];
            2:d=a[2];
            3:d=a[3];
            endcase

endmodule
```

Simulation Code:-

```
module multiplexer_2to1_using_switch_sim;

            // Inputs
            reg [1:0] s;
            reg [3:0] a;

            // Outputs
            wire d;
```

```
	// Instantiate the Unit Under Test (UUT)
	multiplexer_2t01_using_switch uut (
		.s(s),
		.a(a),
		.d(d)
	);
integer i;
	initial begin
		// Initialize Inputs
		s = 0;
		a = 4'b1010;
		for(i=1;i<4;i=i+1)
		#100 s=a[i];

		// Wait 100 ns for global reset to finish


		// Add stimulus here

	end
	initial
	$monitor("a=%b  s=%b  d=%b",a,s,d);

endmodule
```

Output:-


ISim (P.20131013) - [Default.wcfg]
File Edit View Simulation Window Layout Help
Re-launch
Instances and Processes
Instance and Process Name
multiplexer_2to1_using_switc...
glbl
Objects
Simulation Objects for multiplexer_2to1_using_...
Object Name
Value
d 1
s[1:0] 01
a[3:0] 1010
i[31:0] 00000000000000
Name
Value
0 ns
200 ns
400 ns
600 ns
800 ns
1,000.000 ns
X1: 1,000.000 ns
Default.wcfg
Console
Time resolution is 1 ps
Simulator is doing circuit initialization process.
Finished circuit initialization process.
a=1010 s=00 d=0
a=1010 s=01 d=1
a=1010 s=00 d=0
a=1010 s=01 d=1
ISim>
Console
Compilation Log
Breakpoints
Find in Files Results
Search Results
Sim Time: 1,000,000 ps
Type here to search
5:06 PM
5/24/2021

Experiment No.8

Test 4 bit ALU using Verilog.

4 BIT ALU

Implementation Code:-

```
module alu_4_bit(
   input [3:0] a,
   input [3:0] b,
   input [1:0] f,
   output reg [3:0] y
   );
always@(a or b or f)
begin
case(f)
2'b00:y=a+b;
2'b01:y=a-b;
2'b10:y=a&b;
2'b11:y=a^b;
default:y=4'b0000;
endcase
end
endmodule
```

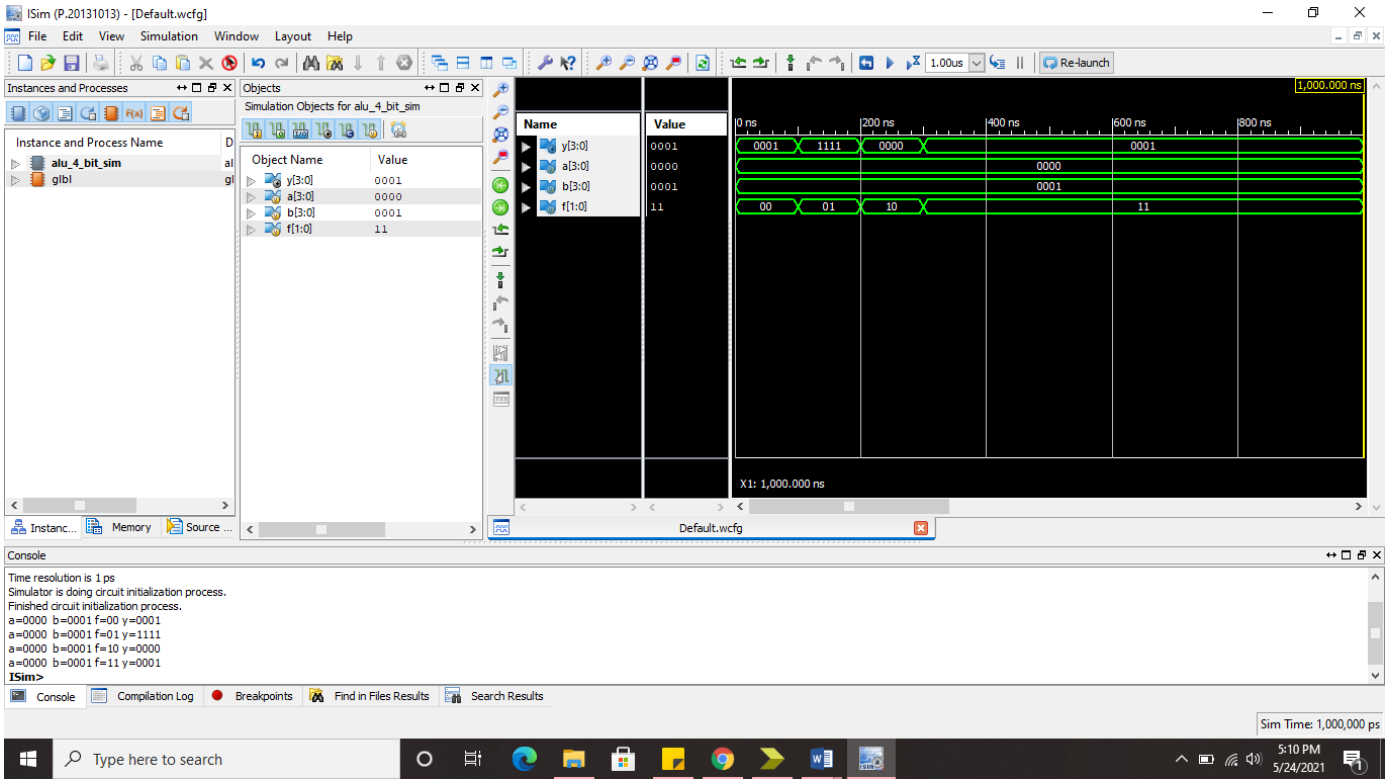
Simulation Code:-

```
module alu_4_bit_sim;

        // Inputs
        reg [3:0] a;
        reg [3:0] b;
```

```
	reg [1:0] f;

	// Outputs
	wire [3:0] y;

	// Instantiate the Unit Under Test (UUT)
	alu_4_bit uut (
		.a(a),
		.b(b),
		.f(f),
		.y(y)
	);

	initial begin
		// Initialize Inputs
		a = 2'b00;
		b = 2'b01;
		f = 2'b00;

		// Wait 100 ns for global reset to finish
		#100;
		f = 2'b01;
		#100
		f = 2'b10;
		#100
		f = 2'b11;

		// Add stimulus here

	end
```

```
        initial

        $monitor("a=%b  b=%b f=%b y=%b",a,b,f,y);


endmodule
```

Output:-



Experiment No.9

Explain the functioning of RAM and ROM.

4 BIT RAM

Implementation Code:-

```
module ram_4_bit(
   input clk,
   input en,
   input wr,
   input [3:0] di,
   input [4:0] addr,
   output reg [3:0] dout
   );
reg[3:0] RAM[31:0];
always@(clk)begin
        if(en)begin
                if(wr)begin
                        RAM[addr]<=di;
                        end
                        dout<=RAM[addr];
                end
        end
endmodule
```

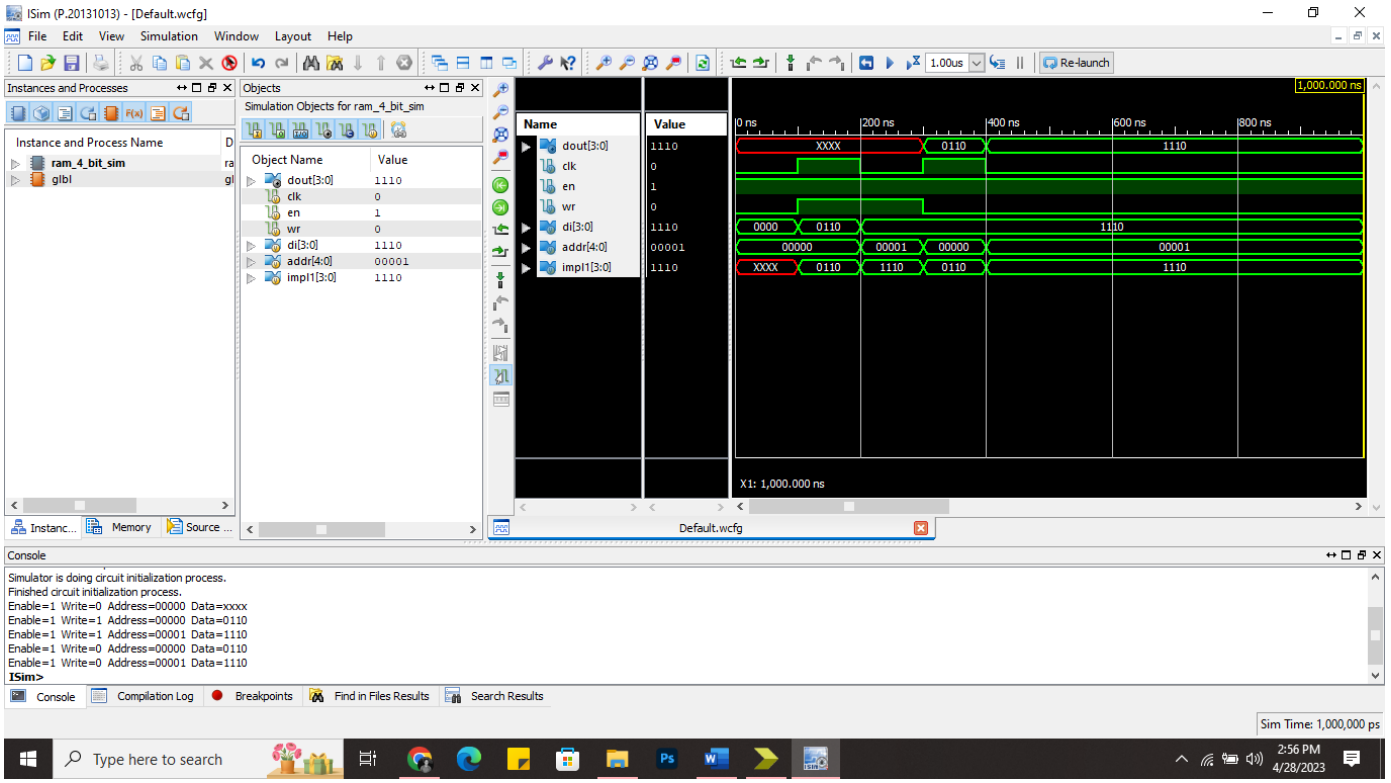
Simulation Code:-

```
module ram_4_bit_sim;
```

```
	// Inputs
	reg clk;
	reg en;
	reg wr;
	reg [3:0] di;
	reg [4:0] addr;

	// Outputs
	wire [3:0] dout;

	// Instantiate the Unit Under Test (UUT)
	ram_4_bit uut (
		.clk(clk),
		.en(en),
		.wr(wr),
		.di(di),
		.addr(addr),
		.dout(dout)
	);

	initial begin
		// Initialize Inputs
		clk = 0;
		en = 1;
		wr = 0;
		di = 0;
		addr = 4'b0000;

		// Wait 100 ns for global reset to finish
		#100;
```

```
                clk = 1;
                wr = 1;
                di = 4'b0110;
                #100;
                clk = 0;
                di = 4'b1110;
                addr = 4'b0001;
                #100;
                clk = 1;
                wr = 0;
                addr = 4'b0000;
                #100;
                clk = 0;
                addr = 4'b0001;

        end
        initial
        $monitor("Enable=%b  Write=%b  Address=%b  Data=%b",en,wr,addr,(wr)?di:dout);


Endmodule
```

Output:-



4 BIT ROM

Implementation Code:-

```
module rom_4_bit(
    input clk,
    input en,
    input [3:0] addr,
    output reg [3:0] data
    );
always@(clk)begin
            if(en)begin
```

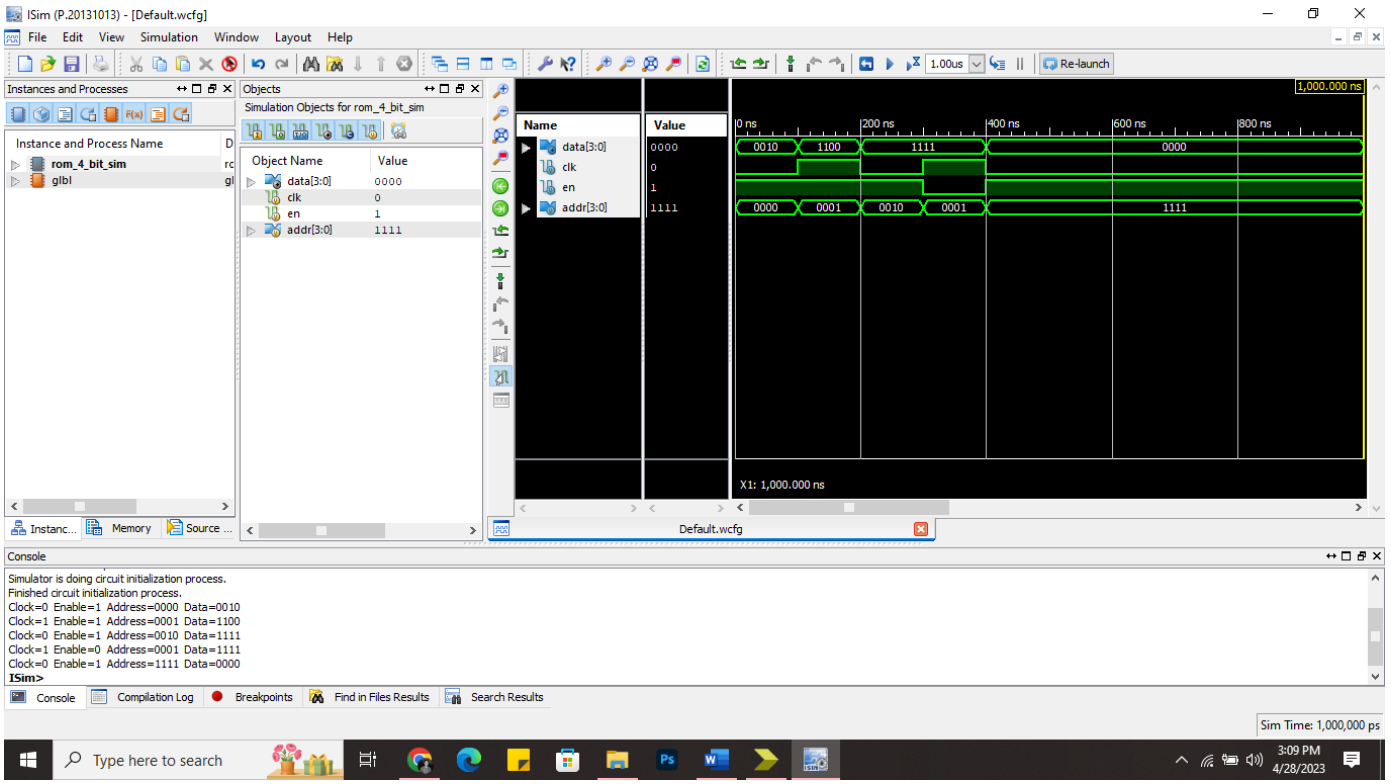
```
		case(addr)
			4'b0000:data<=4'b0010;
			4'b0001:data<=4'b1100;
			4'b0010:data<=4'b1111;
			default:data<=4'b0000;
		endcase
	end
end
endmodule
```

Simulation Code:-

```
module rom_4_bit_sim;

	// Inputs
	reg clk;
	reg en;
	reg [3:0] addr;

	// Outputs
	wire [3:0] data;

	// Instantiate the Unit Under Test (UUT)
	rom_4_bit uut (
		.clk(clk),
		.en(en),
		.addr(addr),
		.data(data)
	);
```

```
        initial begin
                // Initialize Inputs
                clk = 0;
                en = 1;
                addr = 4'b0000;

                // Wait 100 ns for global reset to finish
                #100;
                clk = 1;
                addr = 4'b0001;
    #100;
                clk = 0;
                addr = 4'b0010;
                #100;
                clk = 1;
                en = 0;
                addr = 4'b0001;
                #100;
                clk = 0;
                en = 1;
                addr = 4'b1111;

        end
        initial
        $monitor("Clock=%b  Enable=%b  Address=%b  Data=%b",clk,en,addr,data);

endmodule
```

Output:-


ISim (P.20131013) - [Default.wcfg]
File Edit View Simulation Window Layout Help
Re-launch
Instances and Processes
Instance and Process Name
rom_4_bit_sim
glbl
Objects
Simulation Objects for rom_4_bit_sim
Object Name Value
data[3:0] 0000
clk 0
en 1
addr[3:0] 1111
Name
Value
1,000.000 ns
0 ns
200 ns
400 ns
600 ns
800 ns
0010 1100 1111 0000
0000 0001 0010 0001 1111
X1: 1,000.000 ns
Default.wcfg
Console
Simulator is doing circuit initialization process.
Finished circuit initialization process.
Clock=0 Enable=1 Address=0000 Data=0010
Clock=1 Enable=1 Address=0001 Data=1100
Clock=0 Enable=1 Address=0010 Data=1111
Clock=1 Enable=0 Address=0001 Data=1111
Clock=0 Enable=1 Address=1111 Data=0000
ISim>
Console
Compilation Log
Breakpoints
Find in Files Results
Search Results
Sim Time: 1,000,000 ps
Type here to search
3:09 PM
4/28/2023