

Xilinx Vivado 2018.2 Design Tool Guide (Simulation)

Logic and Digital System Design – CS 303

Sabanci University
Fall 2021

Software & Hardware

- For Lab assignments and Term Project, you will use **Xilinx Vivado 2018.2 Design Tool** to make design.
- During a design process, you will
 1. make your design on Xilinx Vivado 2018.2 Design Tool.
 2. simulate your design on Xilinx Vivado 2018.2 Design Tool.
 3. export your design to FPGA.

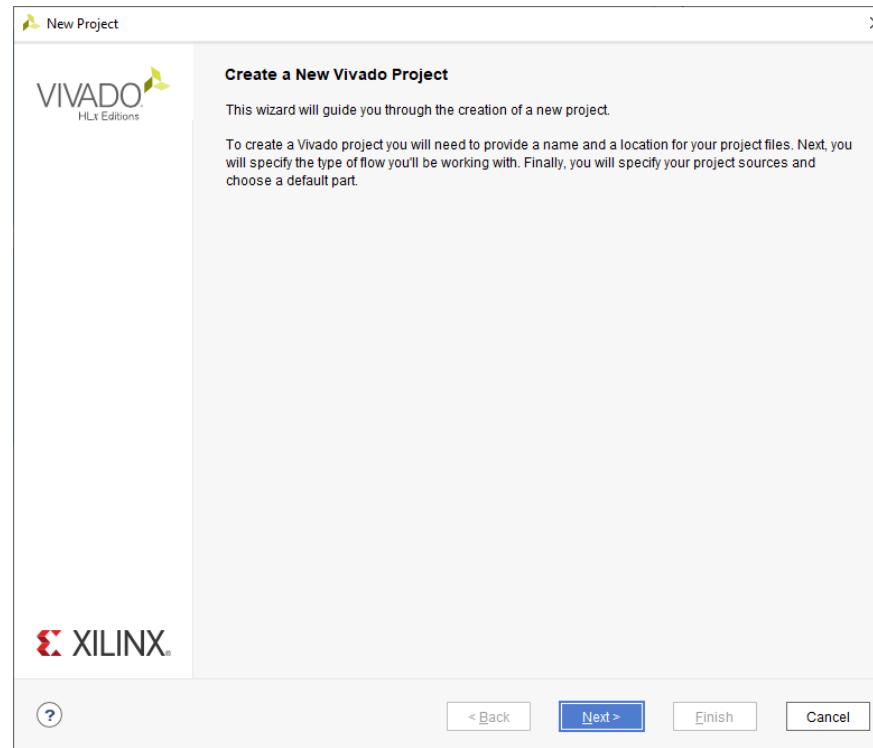
Creating a New Project

- After you start Xilinx Vivado 2018.2, you may use *Create Project* button to create a new project.



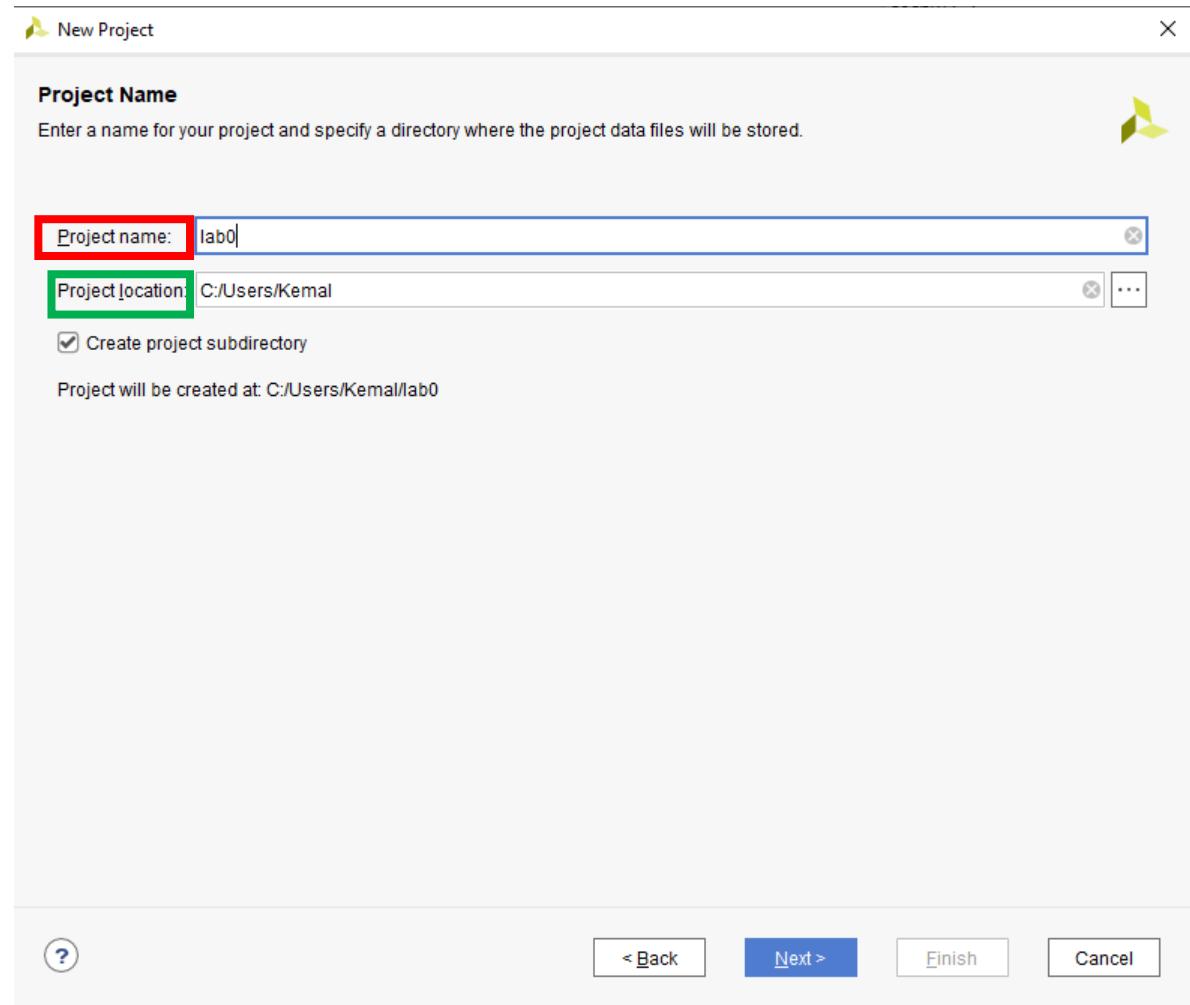
Creating a New Project

- Click on *Next* button.



Creating a New Project

- Specify project name
- Specify project location
- Click on *Next* button

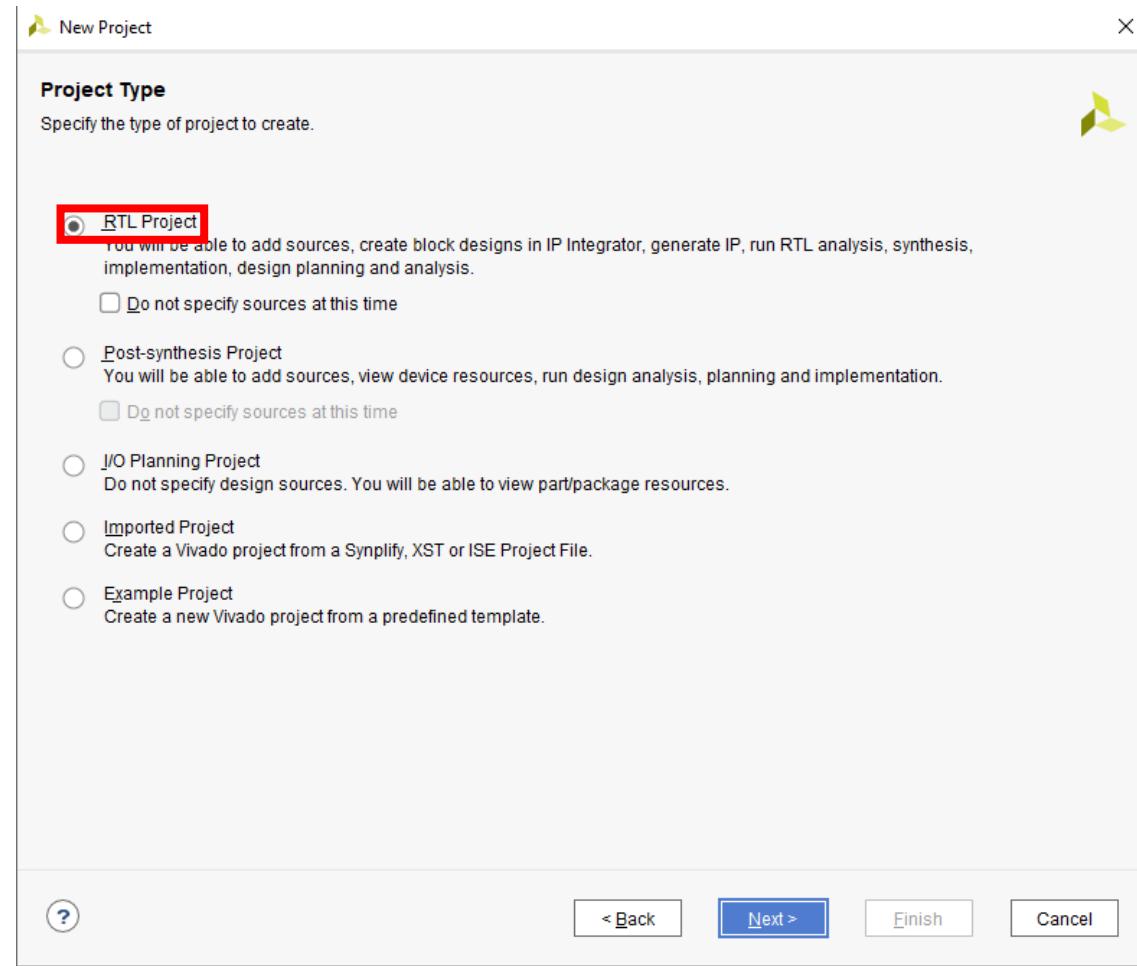


Creating a New Project

- Your project name should not include
 - Turkish character
 - Space
- Directory names in your project path should not include
 - Turkish character
 - Space
 - e.g. D:\Xilinx\Yeni Klasör => **Invalid project path**
 - e.g. D:\Xilinx\lab1 => **Valid project path**

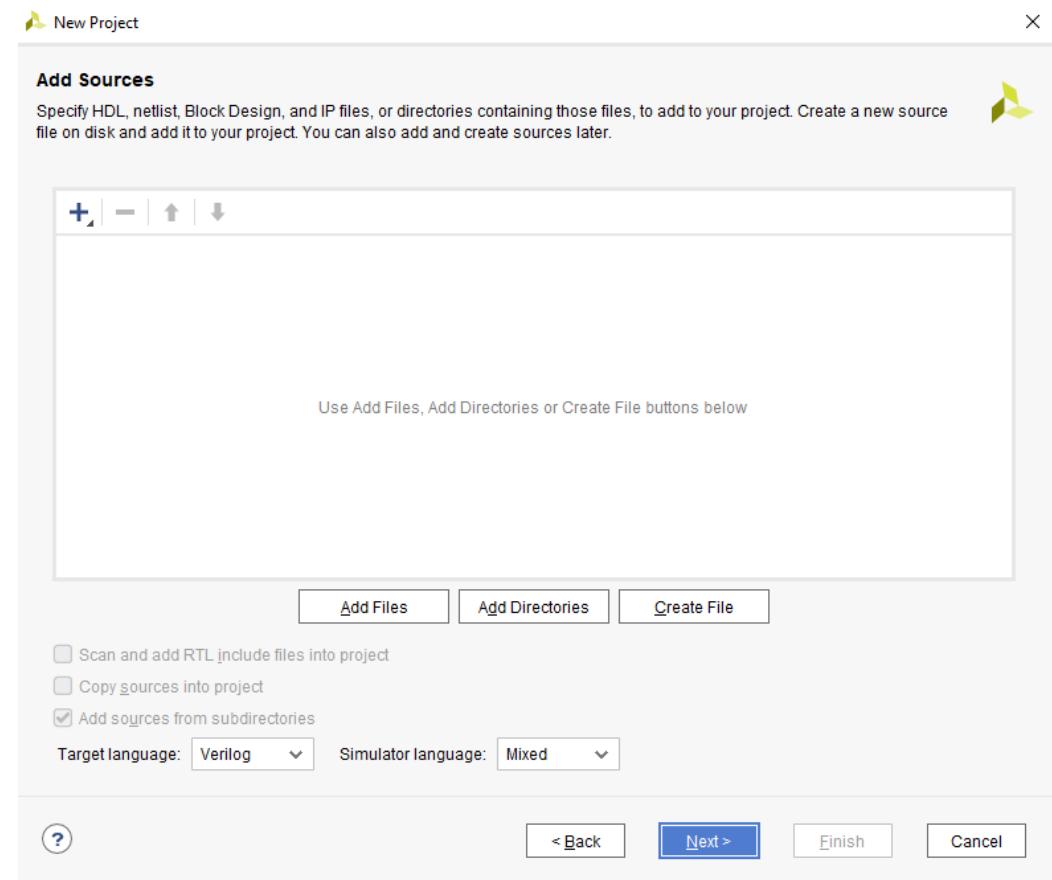
Creating a New Project

- Click on **RTL project**
- Click on **Next** button



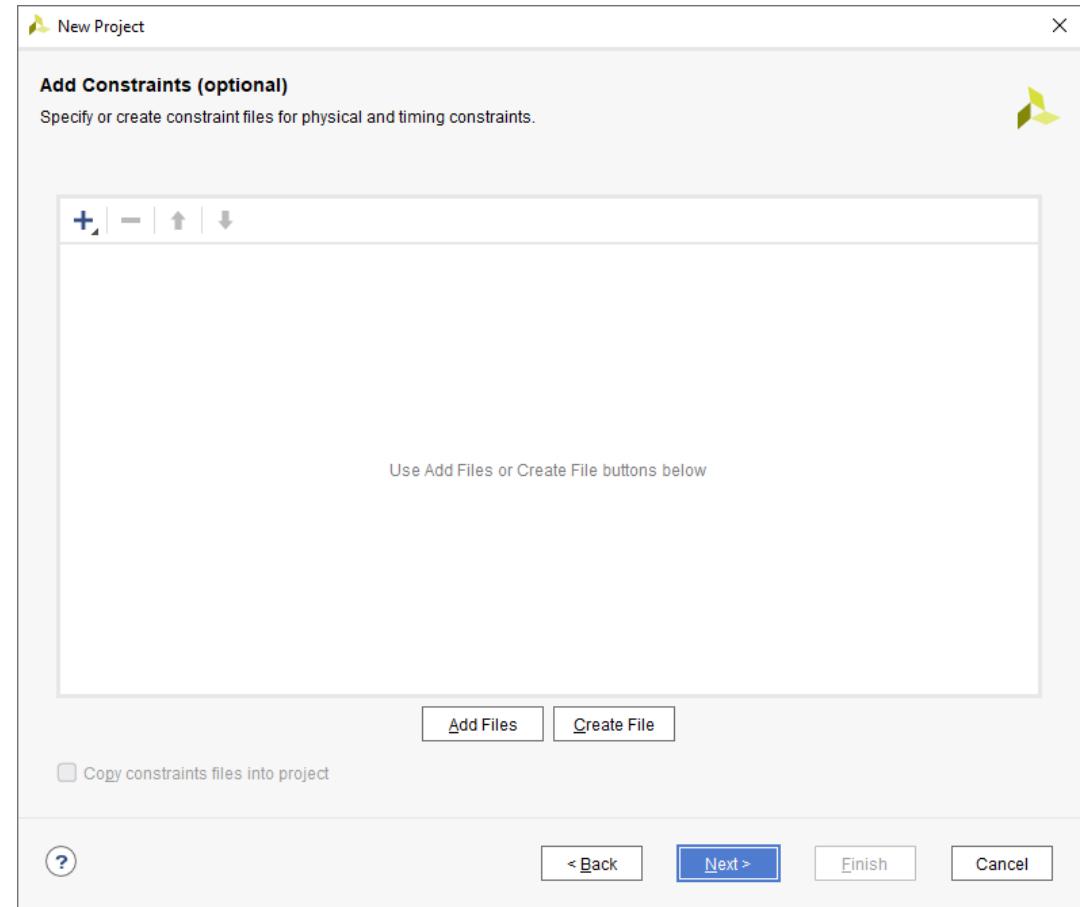
Creating a New Project

- Click on *Next* button



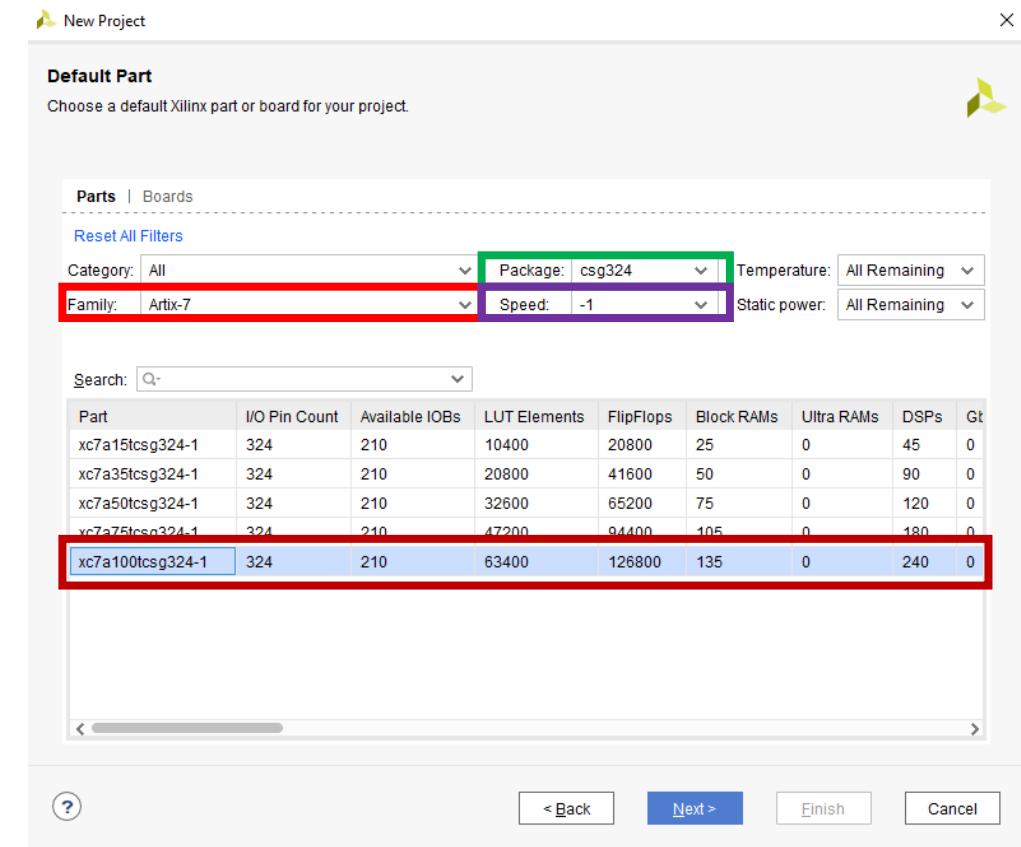
Creating a New Project

- Click on *Next* button



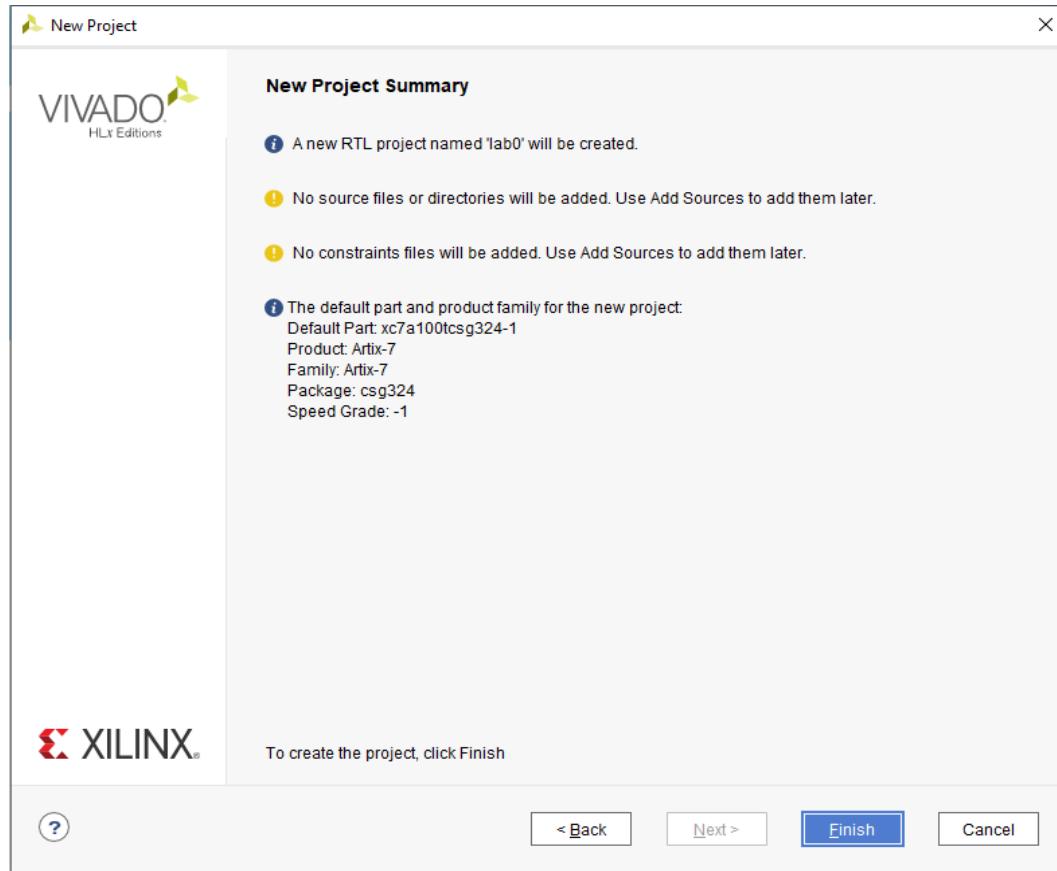
Creating a New Project

- You need to specify device properties.
- Family is **Artix-7**
- Package is **csg324**
- Speed is **-1**
- You need to select **xc7a100tcs3g324-1**
- Click on **Next** button



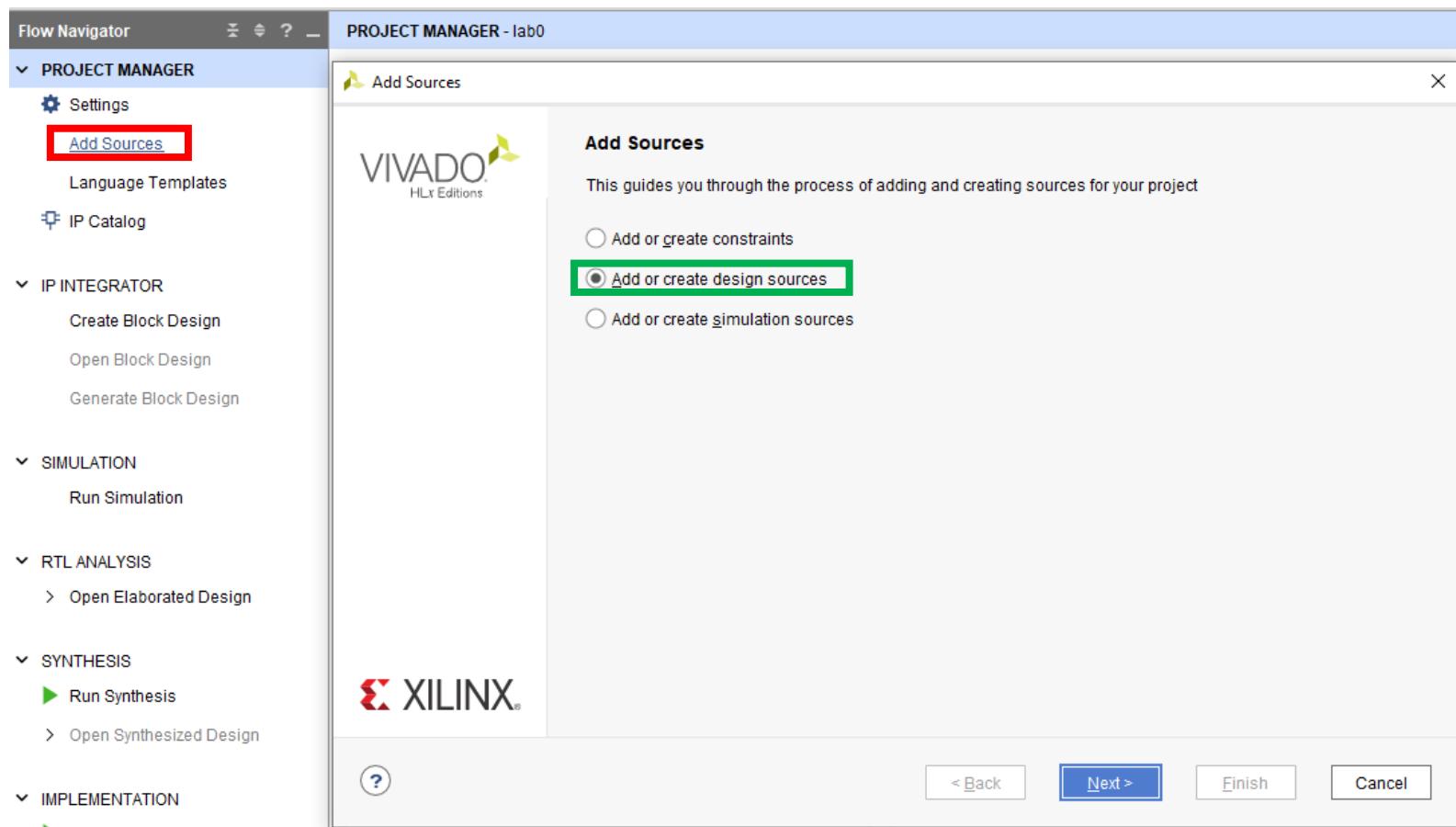
Creating a New Project

- Click on *Finish* button



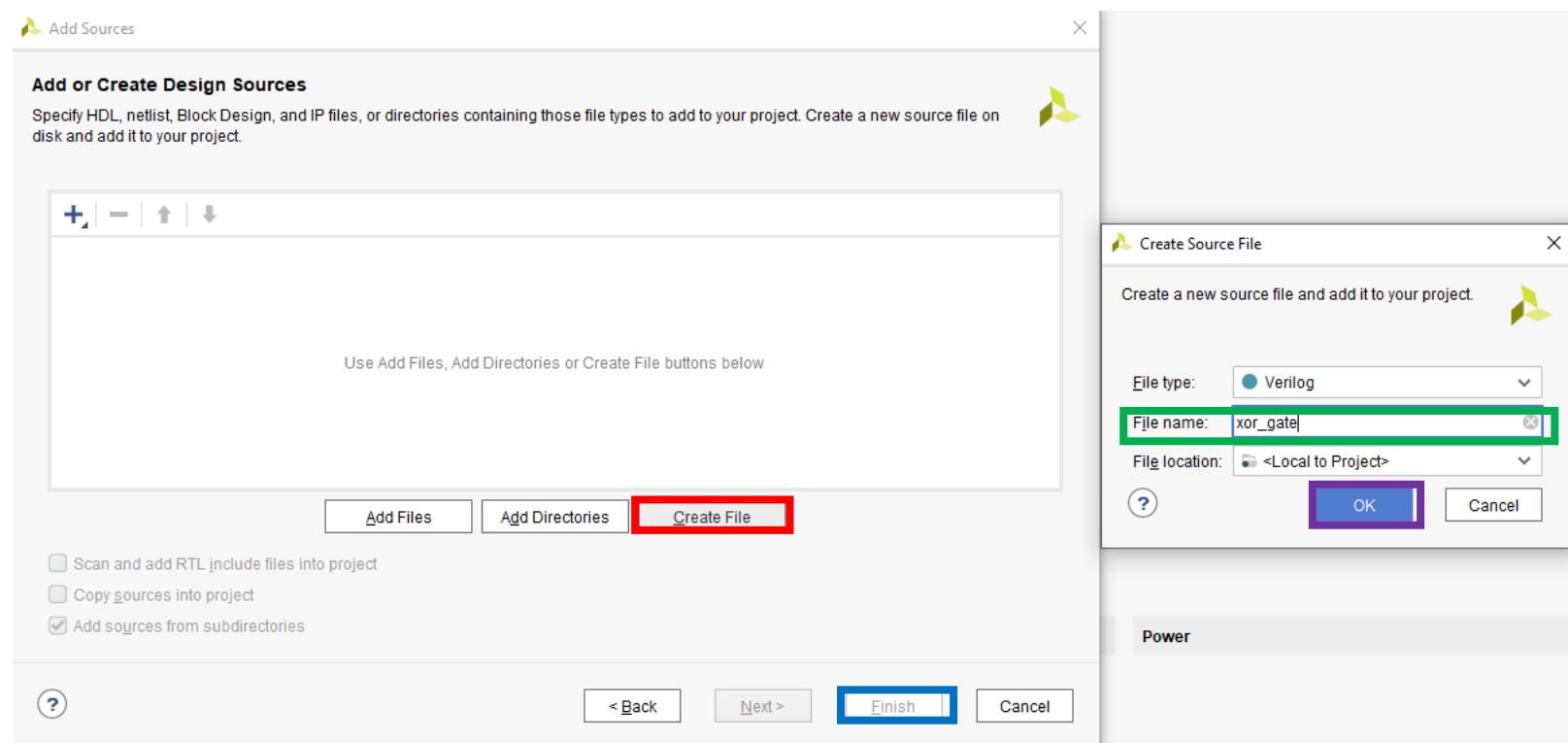
Creating a Source File

- Click on **Add Source** button
- Choose **Add or create design sources**
- Click on **Next** button



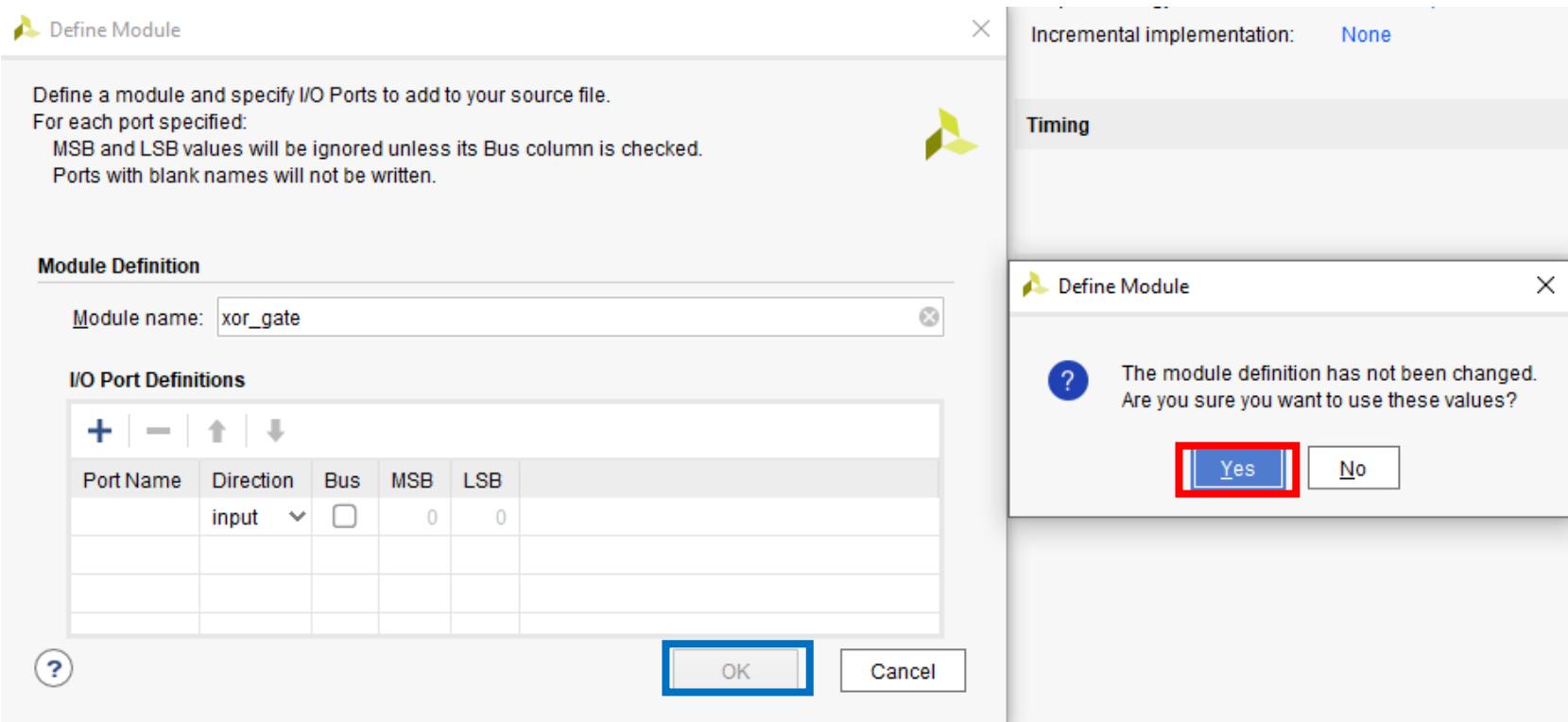
Creating a Source File

- Click on **Create File**
- Give a name to source file
- Click on **OK**
- Click on **Finish** button



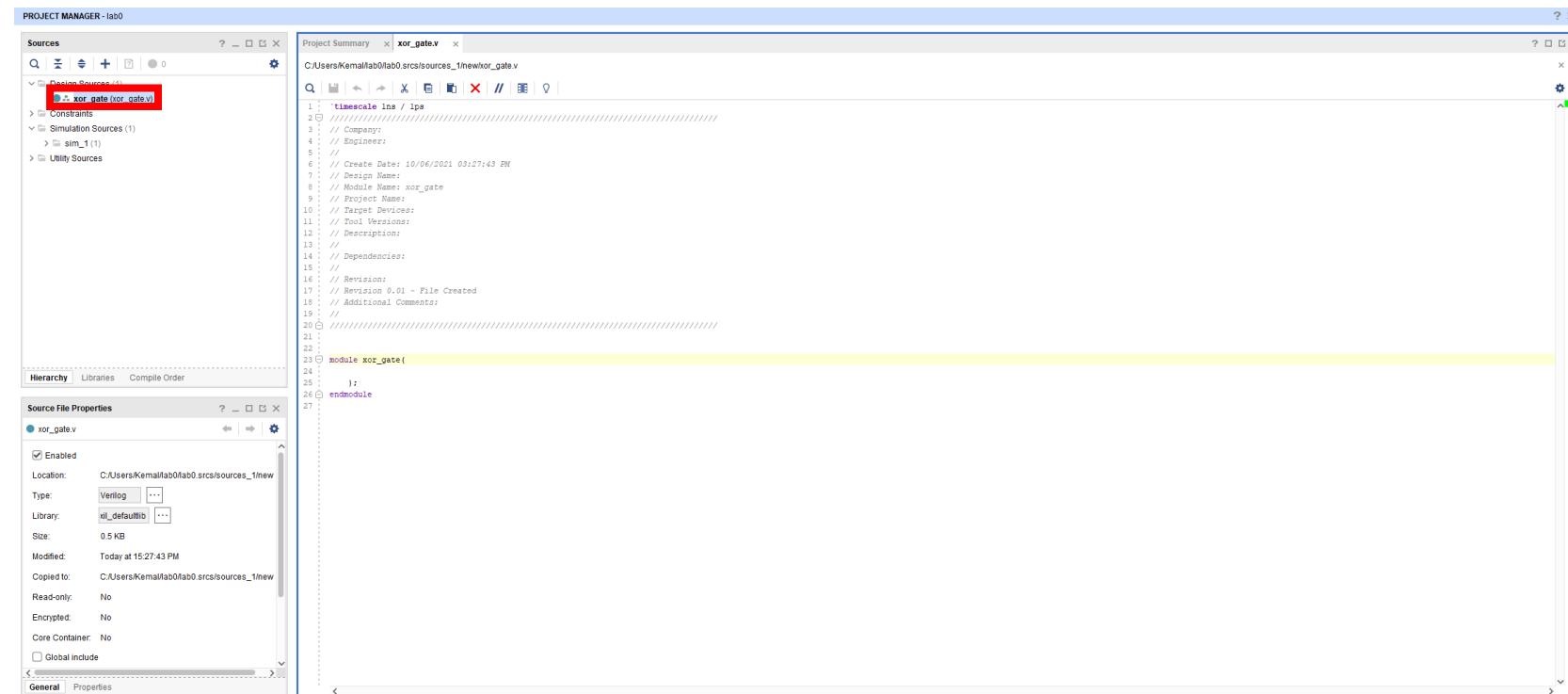
Creating a Source File

- Click on **OK** button
- Click on **Yes** button



Making a Sample Design

- Double click on your **source file**
- The editor will open it



The screenshot shows a software interface for a Verilog project named "lab0". On the left, the "PROJECT MANAGER - lab0" window displays the project structure with a selected "Design Sources (1)" item containing "xor_gate.v". Below it, the "Source File Properties" window for "xor_gate.v" shows details like type "Verilog", library "id_defaultlib", and size "0.5 KB". The main workspace on the right contains the Verilog code for the "xor_gate" module:

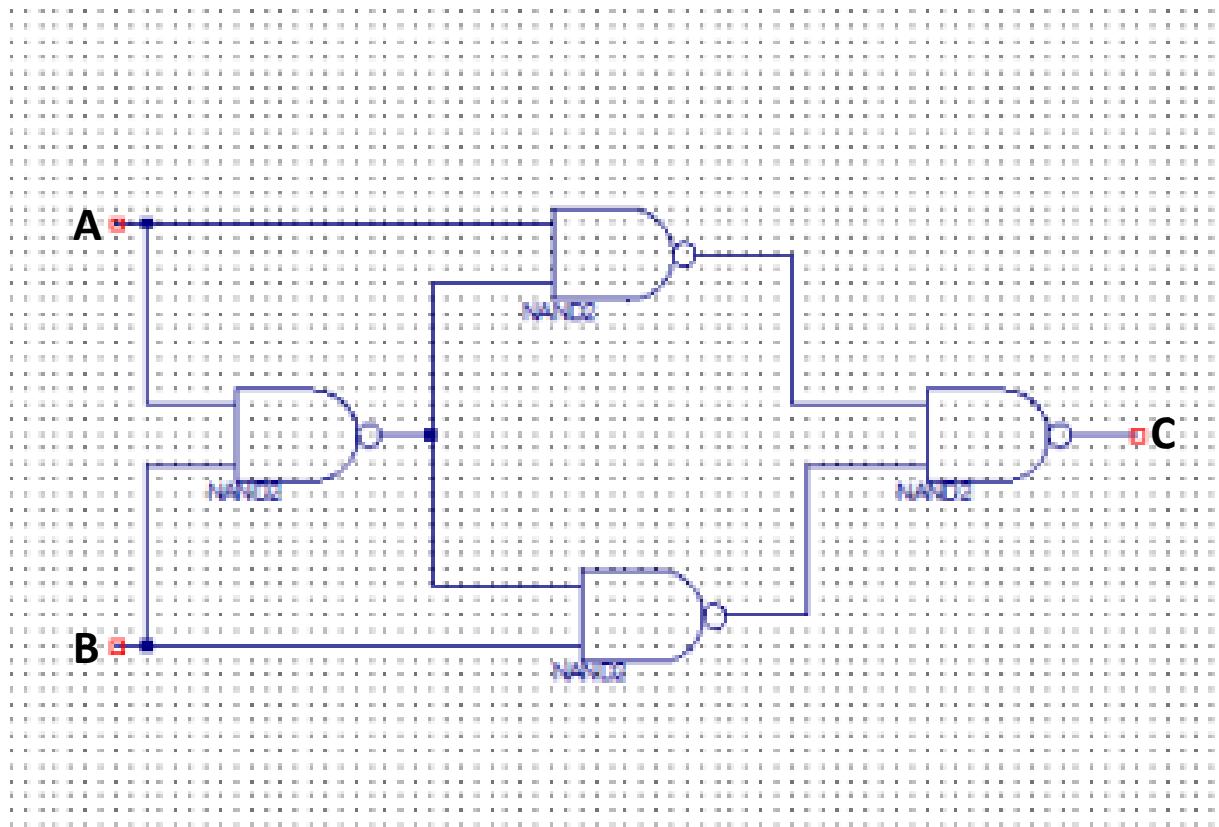
```
timescale 1ns / 1ps
// Company:
// Engineer:
// Create Date: 10/06/2021 03:27:43 PM
// Module Name: xor_gate
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
// Dependencies:
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
// 
module xor_gate(
);
endmodule
```

Making a Sample Design

- In this tutorial, we will implement a simple XOR gate using four NAND gates (do not worry about the details.)
- We are going to use *Verilog*
- Verilog is a Hardware Description Language (HDL).
- It used for describing digital circuits.
- You are describing connections between circuit elements.

A Sample Design – XOR gate

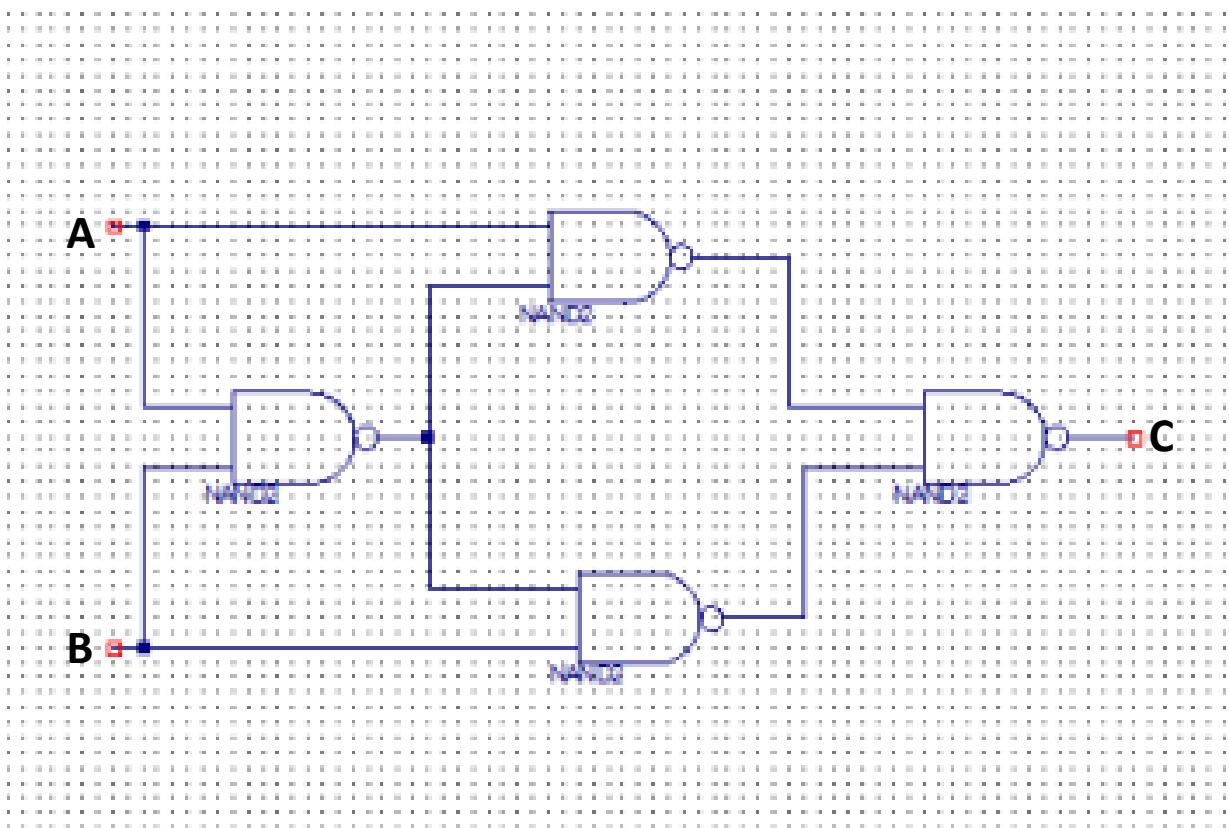
- A XOR gate
 - 2 inputs
 - A
 - B
 - 1 output
 - C



A Sample Design – XOR gate

- **module/endmodule** is used to define the design
- **A unique name** must be given to each design in a project. It has the same name as the source file.

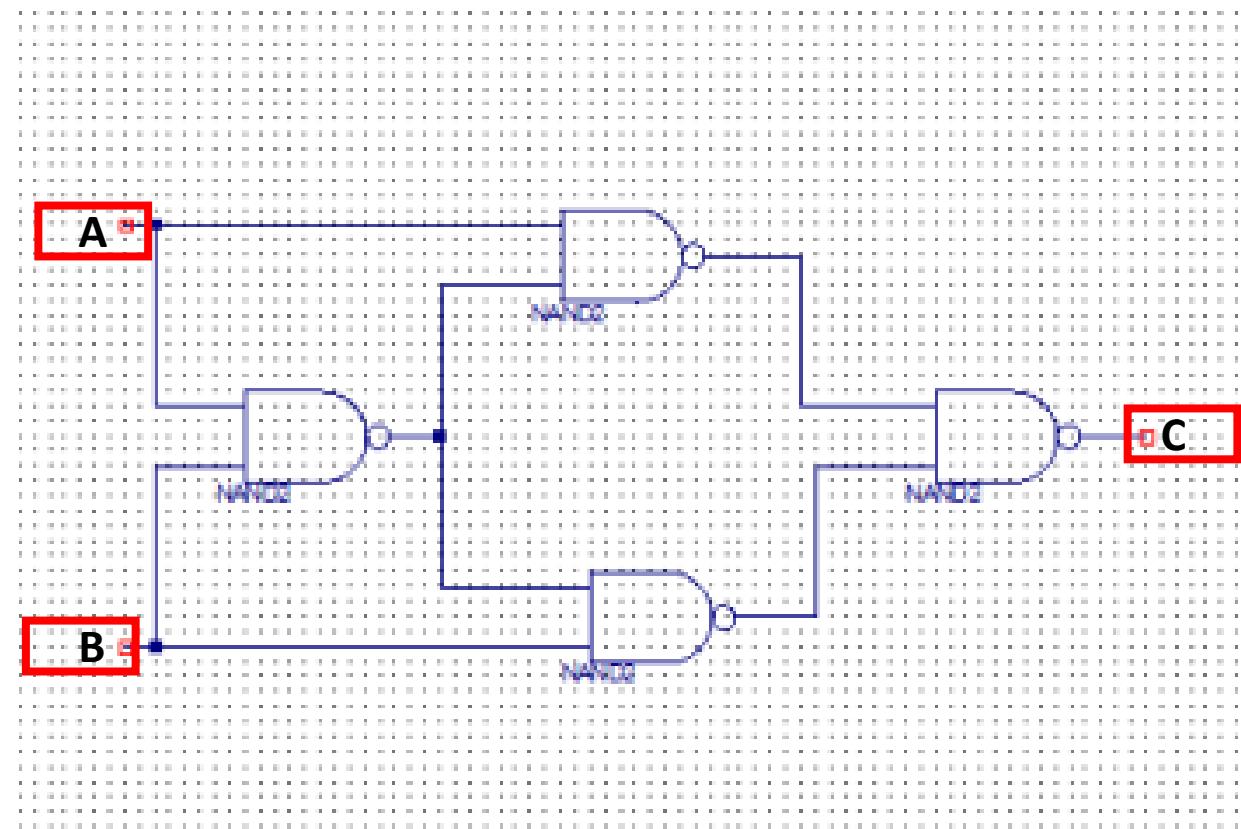
```
21  
22  
23 module xor_gate(  
24  
25 );  
26 endmodule  
27
```



A Sample Design – XOR gate

- All I/Os must be defined in argument list
- Order of the list is not important.

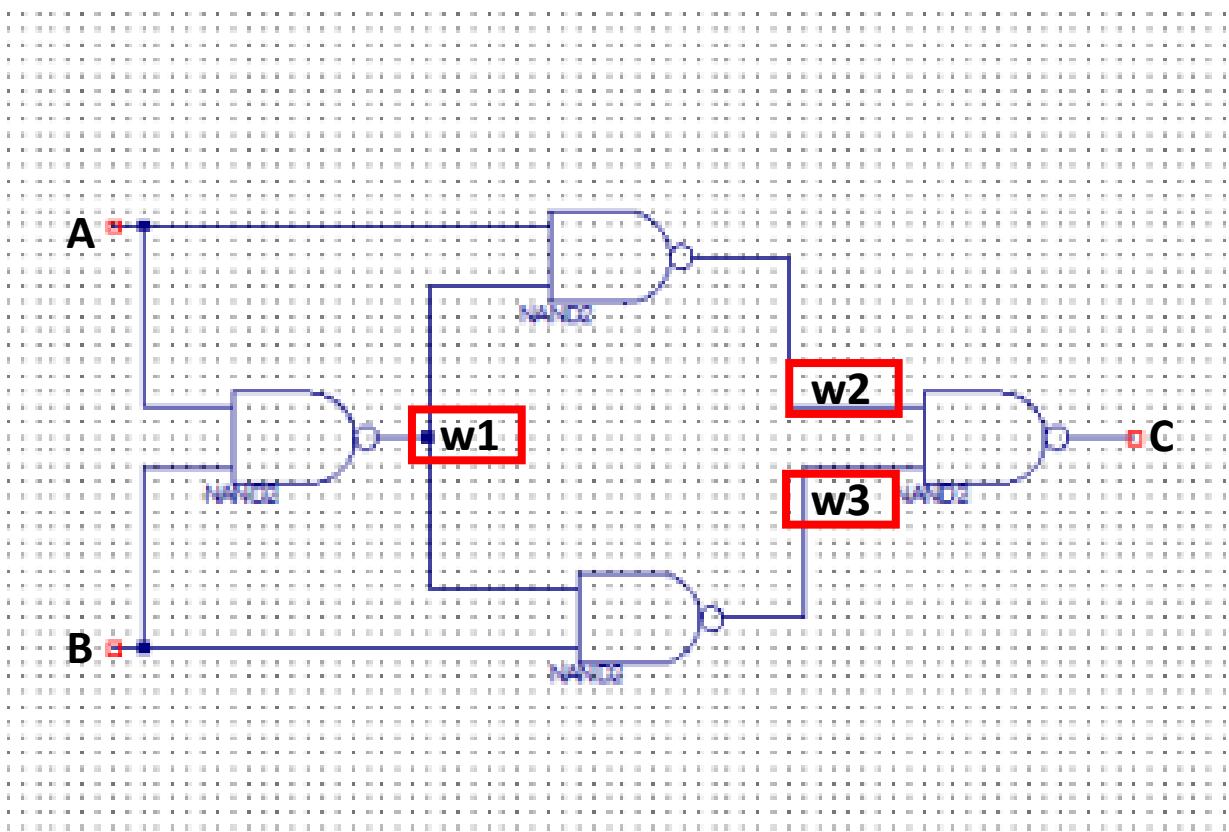
```
22
23 module xor_gate(input A,
24   input B,
25   output C
26
27 );
28 endmodule
29
```



A Sample Design – XOR gate

- There may be some interconnections between gates
- Gates are connected with nets.
- Nets are defined as **wire**.

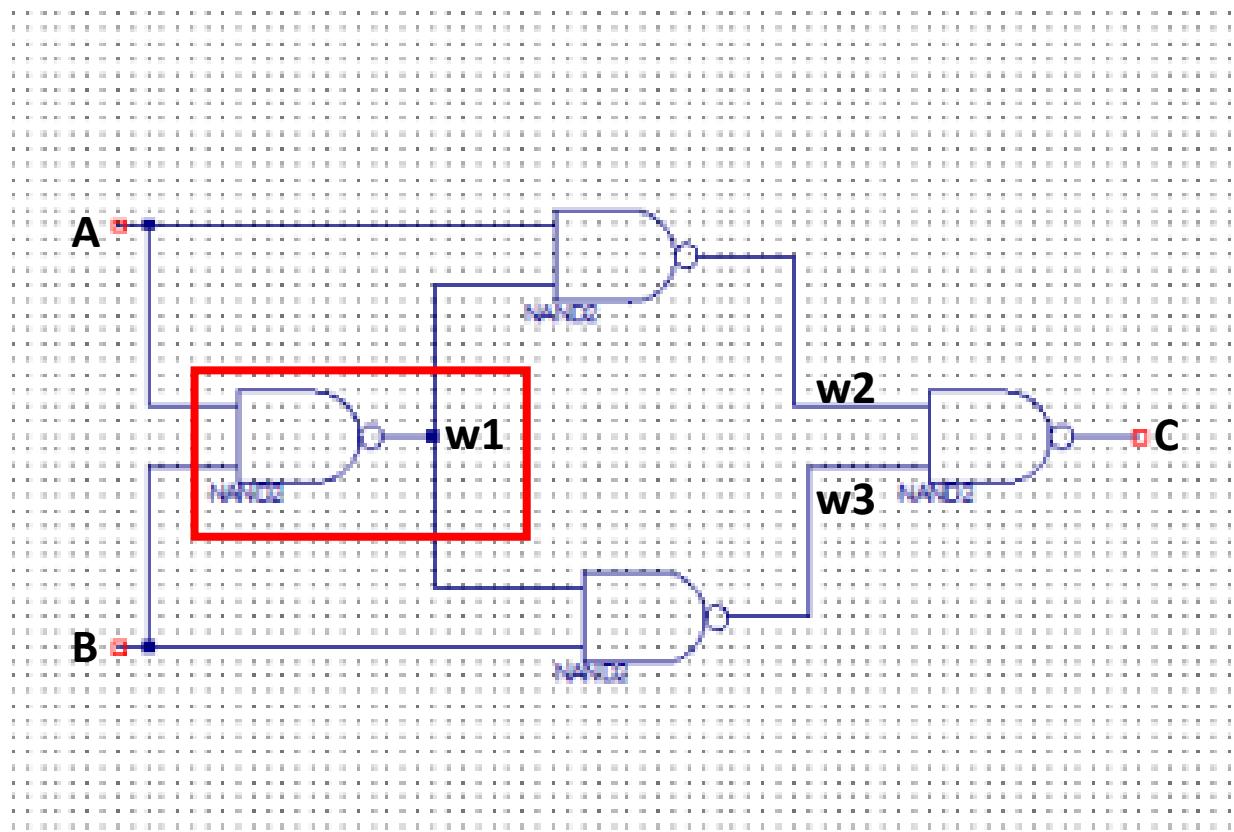
```
21  
22  
23 module xor_gate(input A,  
24           input B,  
25           output C  
26  
27 );  
28  
29   wire w1, w2, w3;  
30  
31 endmodule  
32
```



A Sample Design – XOR gate

- `assign` is used for defining wires.
- `&` is AND operation
- `~` is NOT operation
- Not + AND = NAND

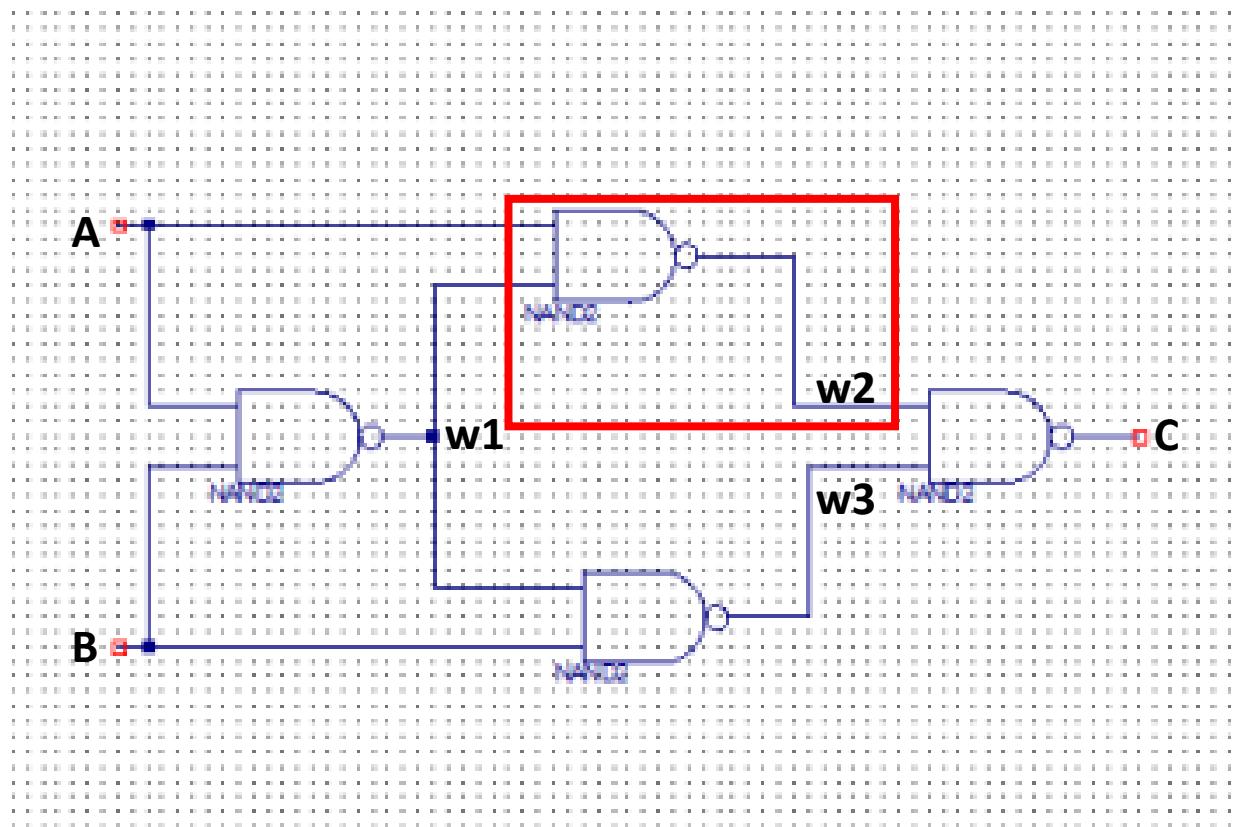
```
22
23 module xor_gate(input A,
24          input B,
25          output C
26
27 );
28
29     wire w1, w2, w3;
30
31     assign w1 = ~(A & B);
32
33 endmodule
34
```



A Sample Design – XOR gate

- `assign` is used for defining wires.
- `&` is AND operation
- `~` is NOT operation
- Not + AND = NAND

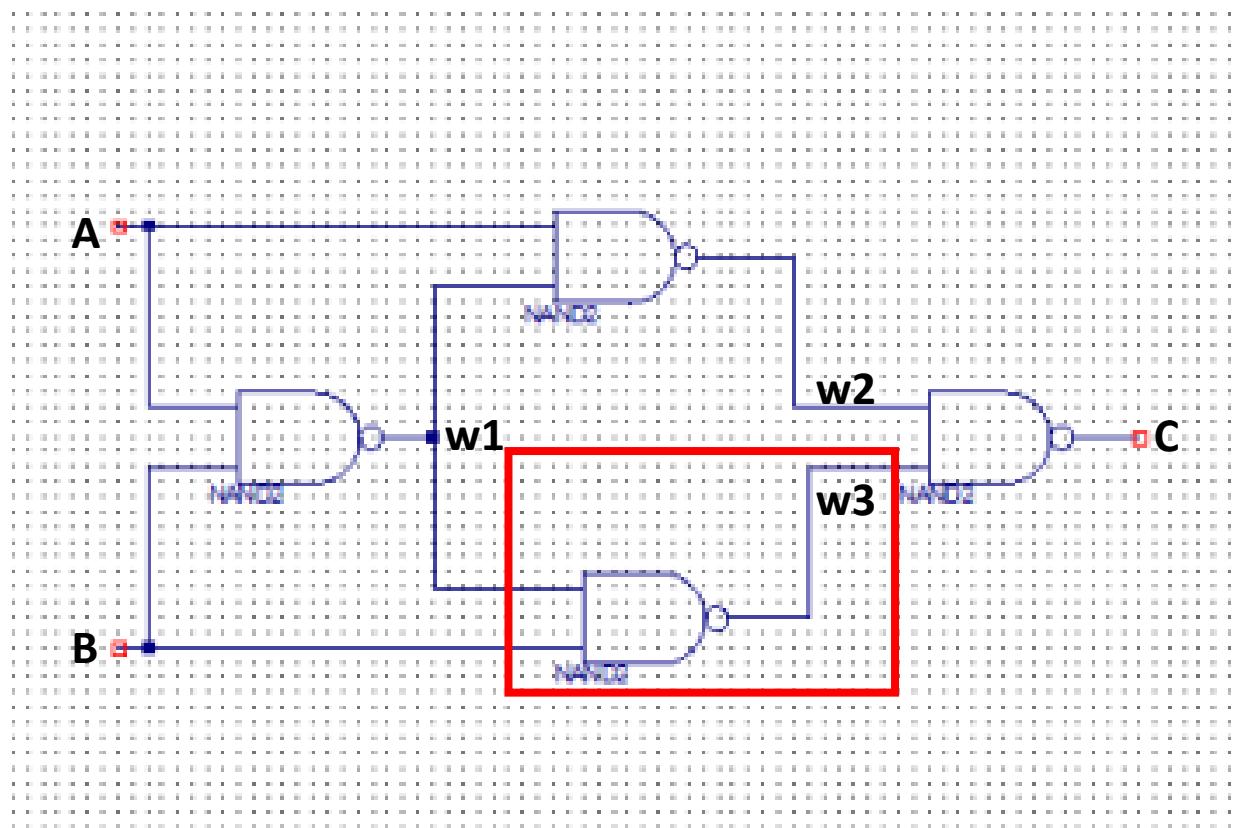
```
22
23 module xor_gate(input A,
24                  input B,
25                  output C
26
27 );
28
29     wire w1, w2, w3;
30
31     assign w1 = ~(A & B);
32     assign w2 = ~(A & w1);
33
34 endmodule
35
```



A Sample Design – XOR gate

- `assign` is used for defining wires.
- `&` is AND operation
- `~` is NOT operation
- Not + AND = NAND

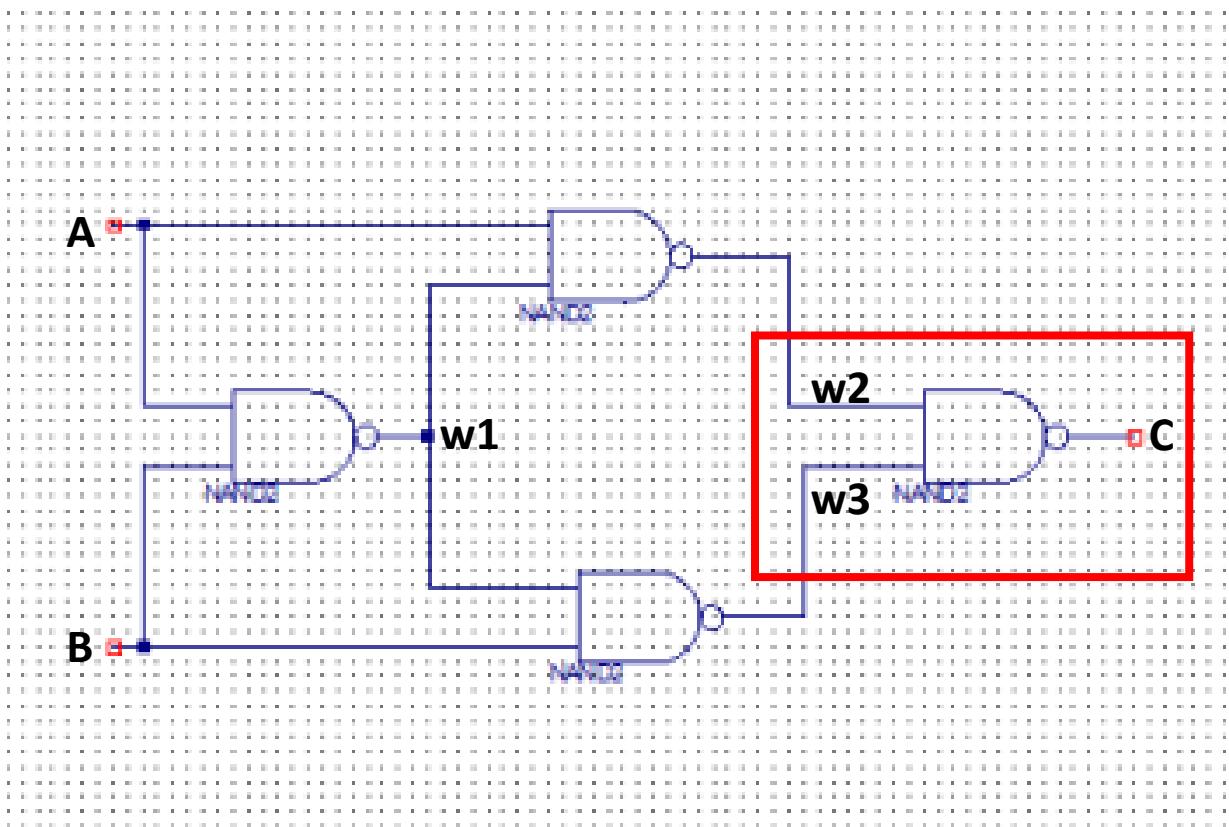
```
22
23 module xor_gate(input A,
24                  input B,
25                  output C
26
27 );
28
29 wire w1, w2, w3;
30
31 assign w1 = ~(A & B);
32 assign w2 = ~(A & w1);
33 assign w3 = ~(B & w1);
34
35 endmodule
36
```



A Sample Design – XOR gate

- `assign` is used for defining wires.
- `&` is AND operation
- `~` is NOT operation
- Not + AND = NAND

```
22
23 module xor_gate(input A,
24         input B,
25         output C
26
27 );
28
29 wire w1, w2, w3;
30
31 assign w1 = ~(A & B);
32 assign w2 = ~(A & w1);
33 assign w3 = ~(B & w1);
34 assign C = ~(w2 & w3);
35
36 endmodule
37
```



A Sample Design – XOR gate

- When you finish your design, click on **Save file** icon.
- If there is no error in your design, you will not see any warnings in **the message section**.

The screenshot shows a CAD software interface with the following components:

- PROJECT MANAGER - lab0**: Shows a tree view of sources: Design Sources (1) containing `xor_gate (xor_gate.v)`, Constraints, Simulation Sources (1), and Utility Sources.
- Source File Properties**: A toolbar with icons for search, filter, add, and save (highlighted with a red box).
- Project Summary**: A tab showing the path `C:/Users/Kamal/lab0/lab0.scs/sources_1/new/xor_gate.v`.
- Code Editor**: Displays Verilog code for an XOR gate:

```
1 //timescale 1ns / 1ps
2 //
3 // Company:
4 // Engineer:
5 //
6 // Create Date: 10/06/2021 03:27:43 PM
7 // Design Name:
8 // Module Name: xor_gate
9 // Project Name:
10 // Target Devices:
11 // Tool Versions:
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 //
21 //
22 //
23 module xor_gate(input A,
24                   input B,
25                   output C
26 );
27
28   wire w1, w2, w3;
29
30   assign w1 = ~ (A & B);
31   assign w2 = ~ (A & w1);
32   assign w3 = ~ (B & w1);
33   assign C = ~ (w2 & w3);
34
35 endmodule
36
37
```
- Messages Panel**: A green-bordered panel at the bottom labeled "Messages" which is currently empty.

A Sample Design – XOR gate

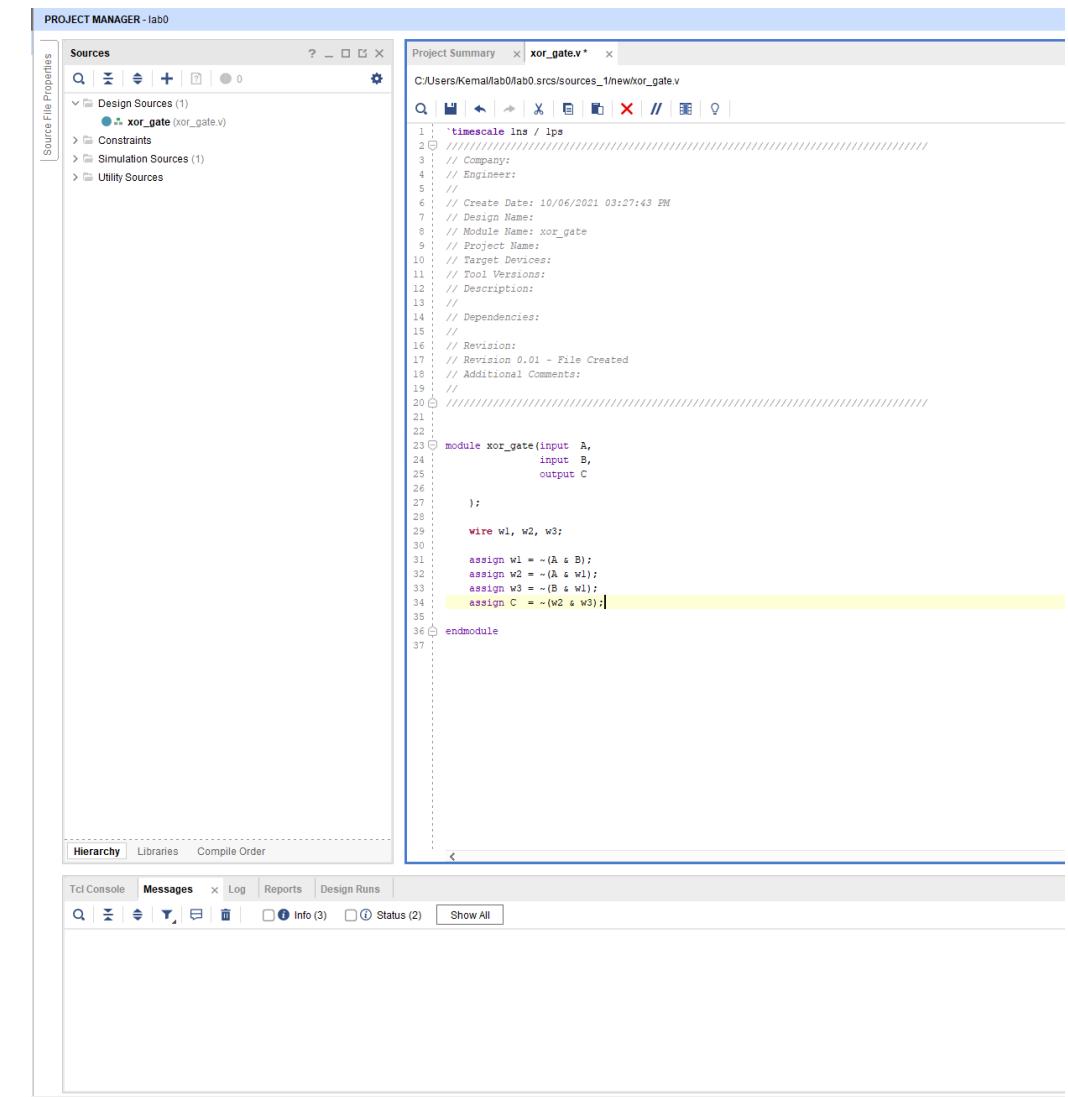
- When you finish your design, click on **Save file** icon.
- If there is an error in your design, you will see warnings in the message section.

The screenshot shows a CAD tool interface with the following components:

- PROJECT MANAGER - lab0**: Shows a tree view of sources: Design Sources (2), Constraints, Simulation Sources (2), and Utility Sources. A red box highlights the "Sources" tab.
- Source File Properties**: A sidebar with various icons.
- Project Summary**: A panel showing the path C:/Users/Kemal/lab0/srsc/sources_1/new/xor_gate.v. A red box highlights the save icon (disk).
- Code Editor**: The file xor_gate.v is open. It contains Verilog code for an XOR gate. A yellow box highlights the "Additional Comments" section at the bottom of the code.
- Tcl Console / Messages**: A panel showing analysis results. A green box highlights the entire panel. The results include:
 - Analysis Results** (2 critical warnings)
 - sources_1** (1 critical warning)
 - [HDL 9-806] Syntax error near "endmodule". [xor_gate.v:36]
 - sim_1** (1 critical warning)
 - [HDL 9-806] Syntax error near "endmodule". [xor_gate.v:36]

A Sample Design – XOR gate

- Design process is finished.
- You need to simulate your design.
- Do not forget to save it.



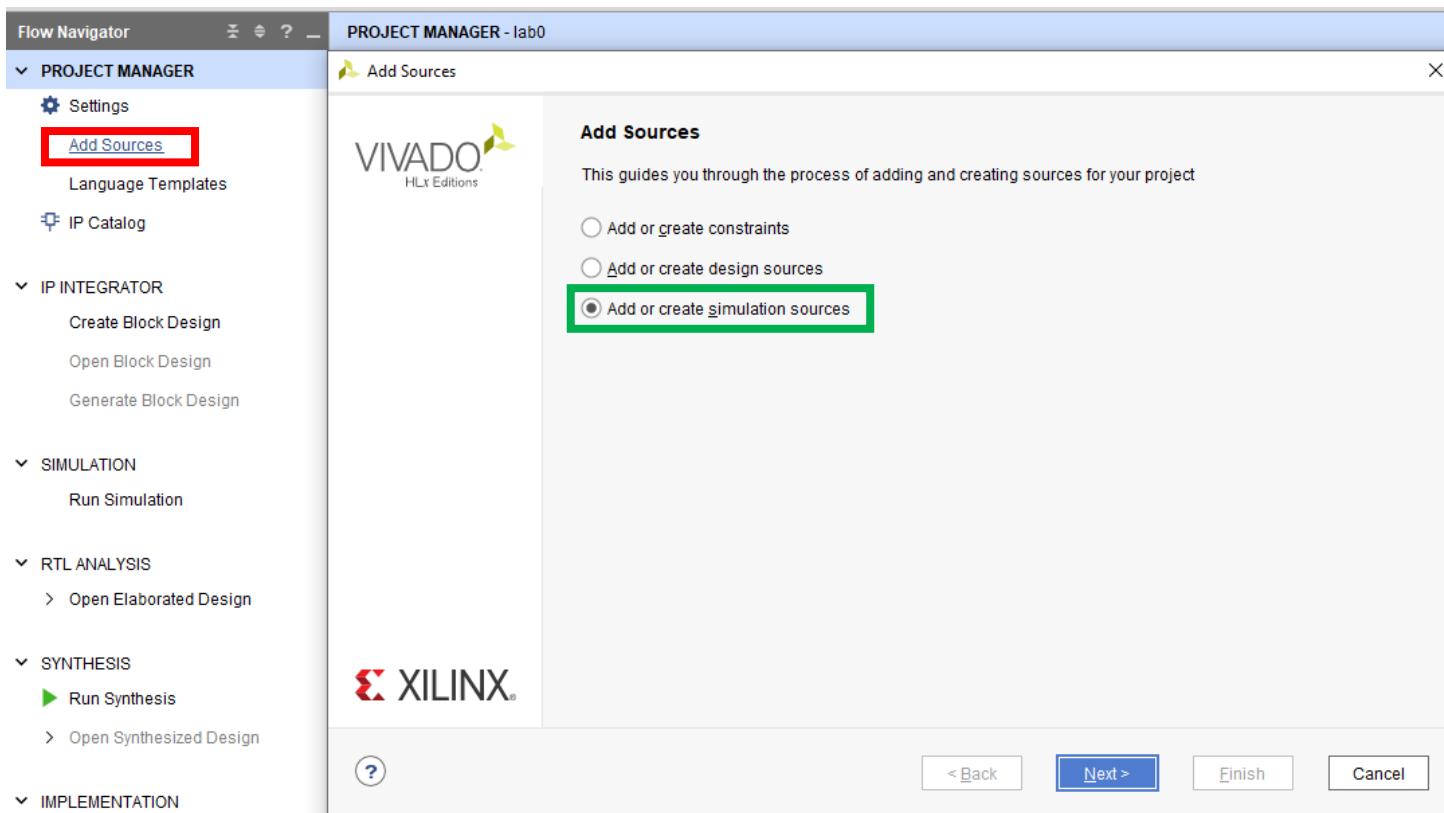
The screenshot shows a CAD software interface with the following components:

- PROJECT MANAGER - lab0**: A sidebar showing the project structure with "Sources" expanded, displaying "Design Sources (1)" containing "xor_gate (xor_gate.v)".
- Code Editor**: A main window titled "Project Summary" with the file "xor_gate.v" open. The code is a Verilog module definition for an XOR gate. The assignment statement "assign C = -(W2 & W3);" is highlighted with a yellow background.
- Tcl Console**: A bottom panel showing the command line interface with tabs for "Messages", "Log", "Reports", and "Design Runs".

```
1: timescale 1ns / 1ps
2: // Company:
3: // Engineer:
4: //
5: // Create Date: 10/06/2021 03:27:43 PM
6: // Design Name:
7: // Module Name: xor_gate
8: // Project Name:
9: // Target Devices:
10: // Tool Versions:
11: // Description:
12: // Dependencies:
13: //
14: // Revision:
15: // Revision 0.01 - File Created
16: // Additional Comments:
17: //
18: ///////////////////////////////////////////////////////////////////
19: //
20: //
21: //
22: module xor_gate(input A,
23:                   input B,
24:                   output C
25: );
26:   wire W1, W2, W3;
27:
28:   assign W1 = -(A & B);
29:   assign W2 = -(A & W1);
30:   assign W3 = -(B & W1);
31:   assign C = -(W2 & W3);
32:
33: endmodule
34:
35: //
36: //
37: //
```

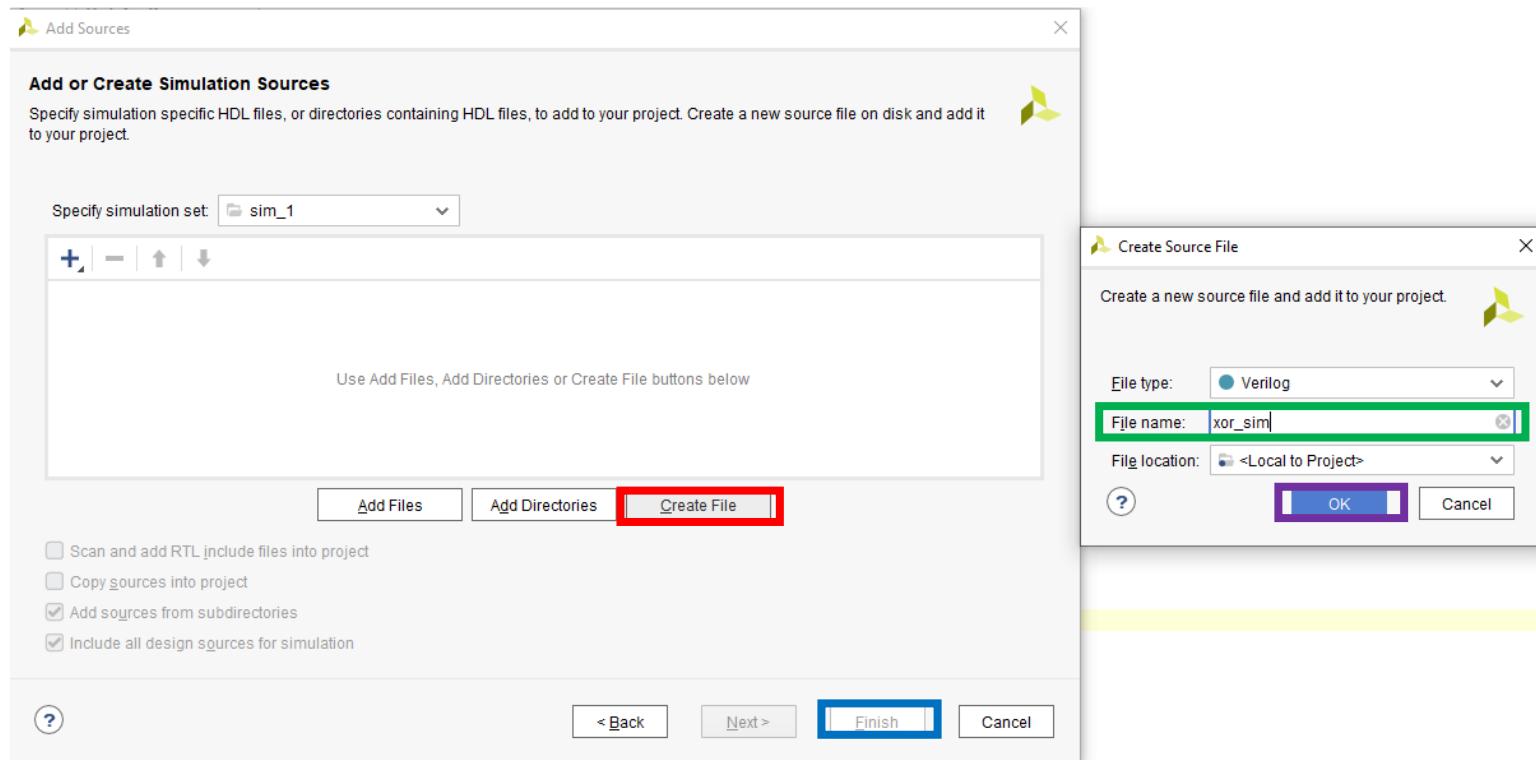
Simulation

- Click on **Add Source** button
- Choose **Add or create simulation sources**
- Click on **Next** button



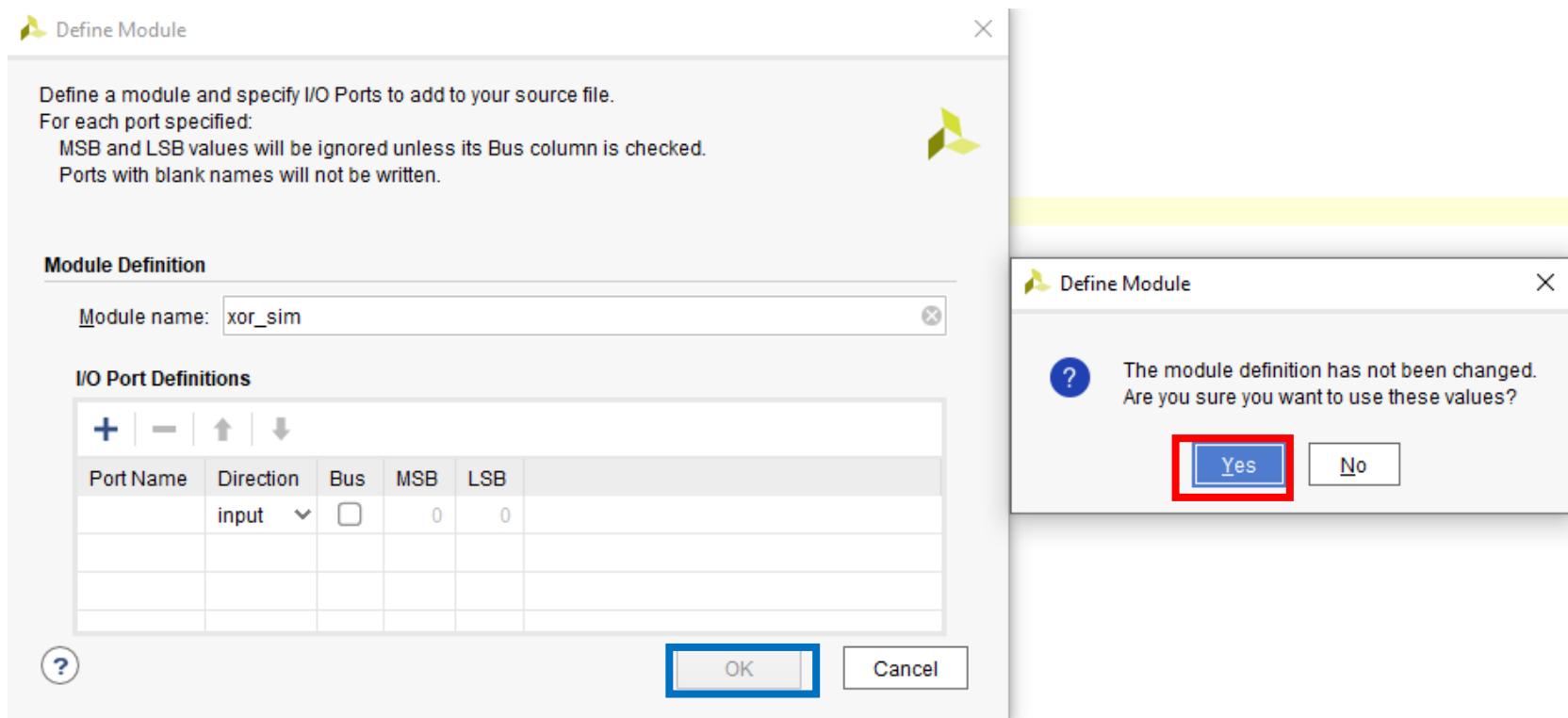
Simulation

- Click on **Create File**
- Give **a name to source file**
- Click on **OK**
- Click on ***Finish*** button



Simulation

- Click on **OK** button
- Click on **Yes** button



Simulation

- Click on **Simulation Sources** button
- Double click on *sim file*
- The editor will open it

The screenshot shows a CAD software interface with two main windows. On the left is the 'PROJECT MANAGER - lab0' window, which displays a tree view of project files under 'Sources'. A red box highlights the 'Simulation Sources (2)' section, and a blue box highlights the 'xor_sim.v' file within it. At the bottom of this window are tabs for 'Hierarchy', 'Libraries', and 'Compile Order', with 'Hierarchy' currently selected. On the right is a code editor window titled 'Project Summary' with tabs for 'xor_gate.v' and 'xor_sim.v'. The code editor shows Verilog simulation code for an XOR gate. The 'xor_sim.v' tab is active, displaying the following code:

```
1 `timescale 1ns / 1ps
2 //////////////////////////////////////////////////////////////////
3 // Company:
4 // Engineer:
5 //
6 // Create Date: 10/06/2021 09:36:39 PM
7 // Design Name:
8 // Module Name: xor_sim
9 // Project Name:
10 // Target Devices:
11 // Tool Versions:
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 //////////////////////////////////////////////////////////////////
21
22
23 module xor_sim(
24
25 );
26 endmodule
27
```

Simulation

- Test file needs to be created manually.
- Inputs
- Outputs
- Instantiation
- Scenario

```
20 //////////////////////////////////////////////////////////////////
21
22
23 module xor_sim(
24
25 );
26
27 //Inputs
28 reg A;
29 reg B;
30
31 //Outputs
32 wire C;
33
34 //Instantiate the UUT
35 xor_gate UUT(
36 .A(A),
37 .B(B),
38 .C(C)
39 );
40
41 //Initialize Inputs
42 initial begin
43     A = 0;
44     B = 0;
45 end
46 endmodule
47
```

Simulation

- You will write your test scenario between initial begin and end
- A = 1; is used to apply Logic-1 to input A.
- #10; puts 10ns delay

```
21  
22  
23 module xor_sim(  
24 );  
25 //Inputs  
26 reg A;  
27 reg B;  
28  
29 //Outputs  
30 wire C;  
31  
32 //Instantiate the UUT  
33 xor_gate UUT(  
34 .A(A),  
35 .B(B),  
36 .C(C)  
37 );  
38  
39  
40  
41 //Initialize Inputs  
42 initial begin  
43     A = 0;  
44     B = 0;  
45     #10;  
46     A = 1;  
47     #10;  
48     A = 0;  
49     B = 1;  
50     #10;  
51     A = 1;  
52     #10;  
53 end  
54 endmodule  
55
```

Simulation

- When you finish your design, click on **Save file icon**.
- When you save it, you will see that **sim file** is set as Top module in the simulation sources.

The screenshot shows a CAD software interface with two main windows. On the left is the 'PROJECT MANAGER - lab0' window, which displays a 'Sources' tree view. Under 'Design Sources', there is a file named 'xor_gate (xor_gate.v)'. Under 'Simulation Sources', there is a folder 'sim_1 (1)' containing two files: 'xor_sim (xor_sim.v)' and 'UUT: xor_gate (xor_gate.v)'. The 'xor_sim (xor_sim.v)' file is highlighted with a green border. On the right is a 'Project Summary' window showing the contents of the 'xor_sim.v' file. The code defines a module 'xor_sim' with inputs 'A' and 'B' and output 'C'. It instantiates a UUT 'xor_gate' and initializes inputs 'A' and 'B' with values 0, 1, and 0 again. A red box highlights the save icon in the toolbar of the Project Summary window. The code is as follows:

```
// Create Date: 10/06/2021 09:36:39 PM
// Design Name:
// Module Name: xor_sim
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
// -----
module xor_sim(
);

//Inputs
reg A;
reg B;

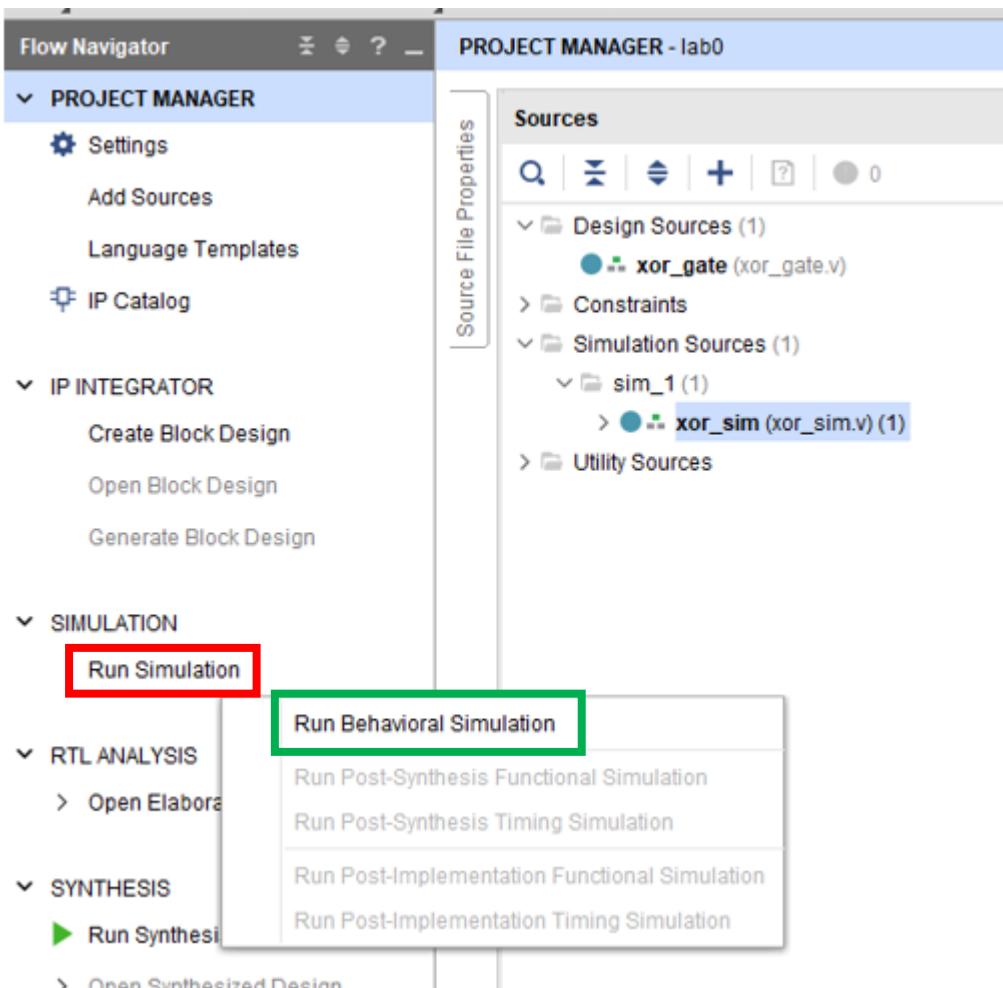
//Outputs
wire C;

//Instantiate the UUT
xor_gate UUT(
.A(A),
.B(B),
.C(C)
);

//Initialize Inputs
initial begin
    A = 0;
    B = 0;
    #10;
    A = 1;
    #10;
    A = 0;
    B = 1;
    #10;
    A = 1;
    #10;
end
endmodule
```

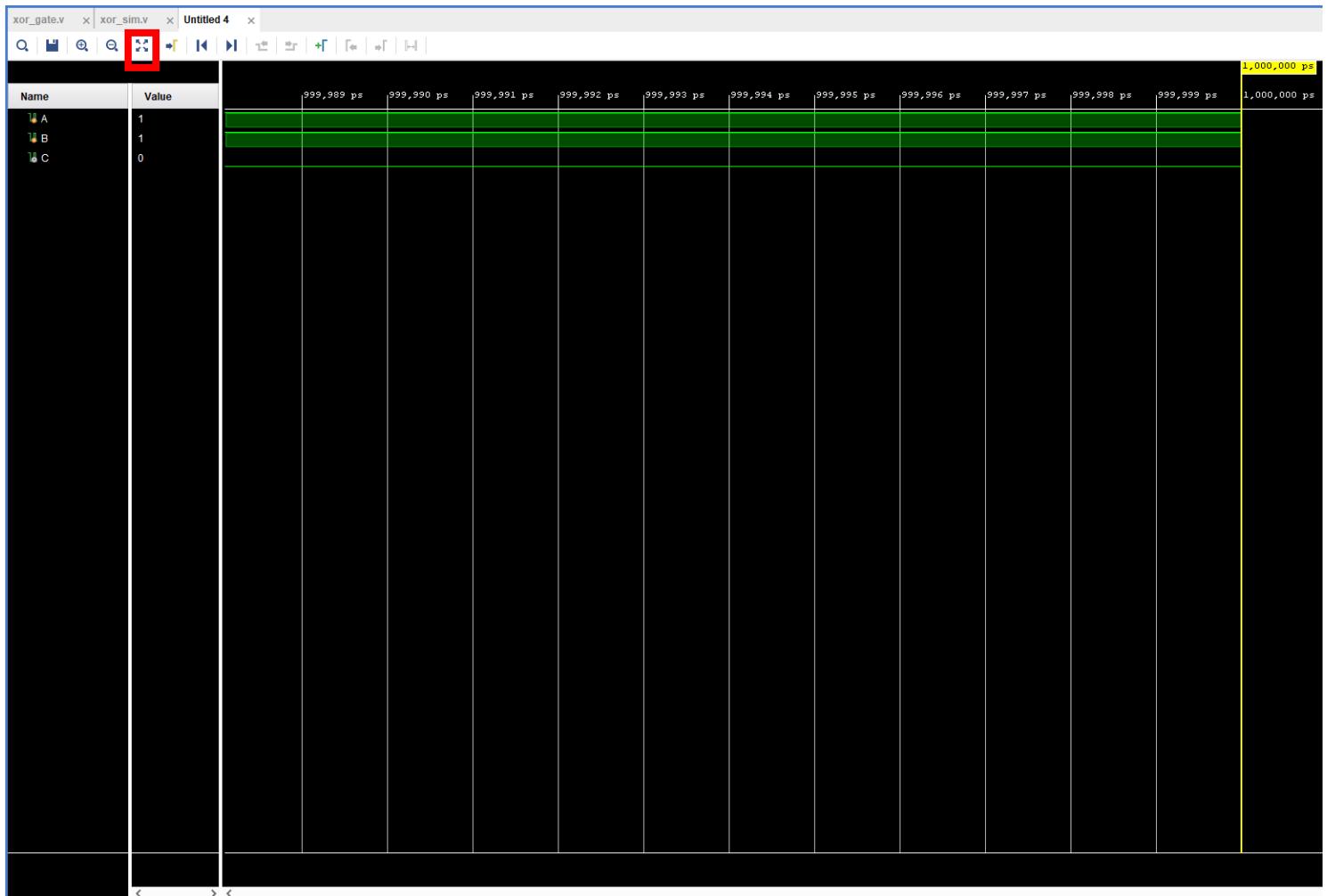
Simulation

- Click on Run Simulation
- Click on Run Behavioral Simulation



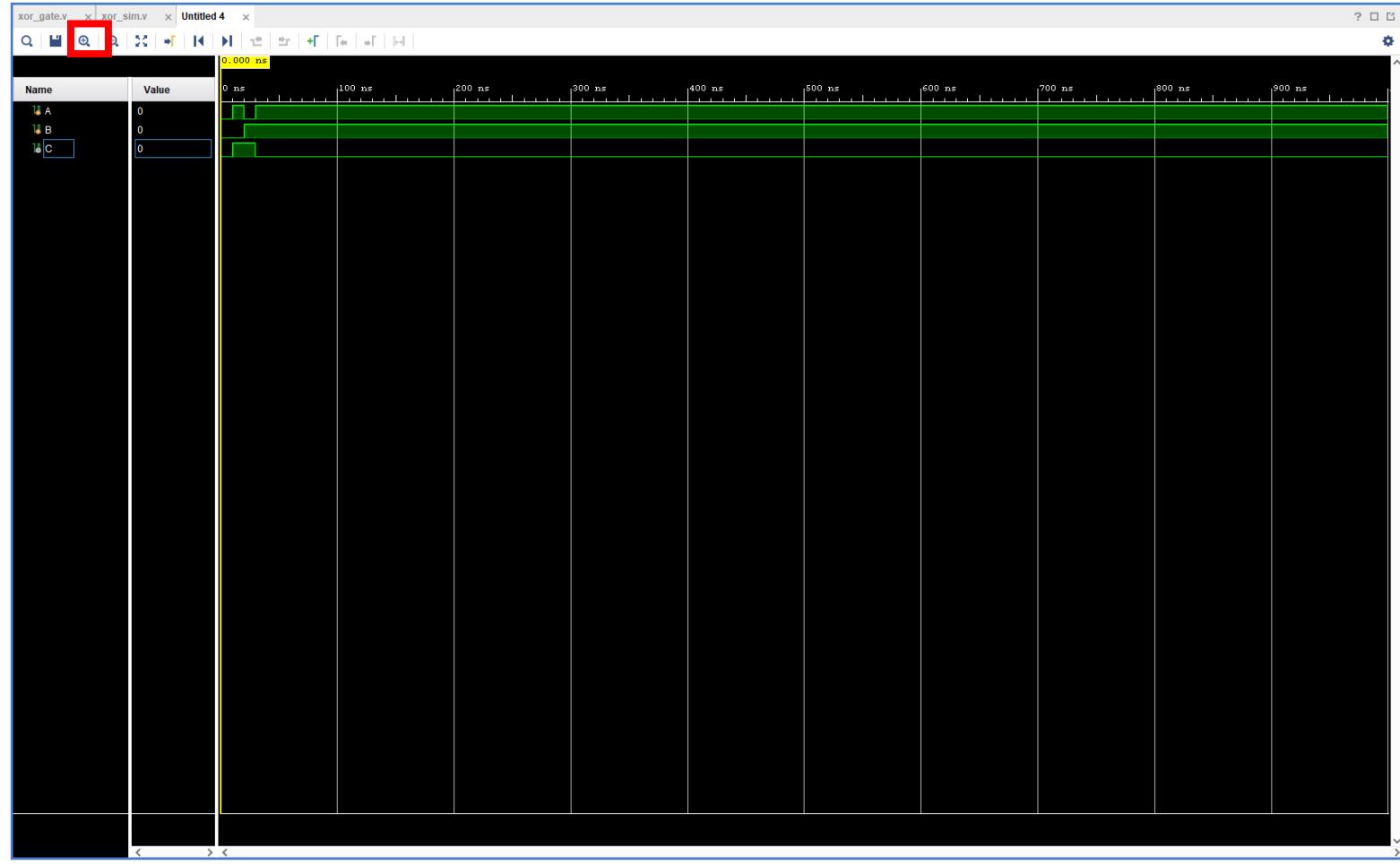
Simulation

- Simulation shows the end of the simulation time by default.
- You can see all run time by **Zoom Fit**



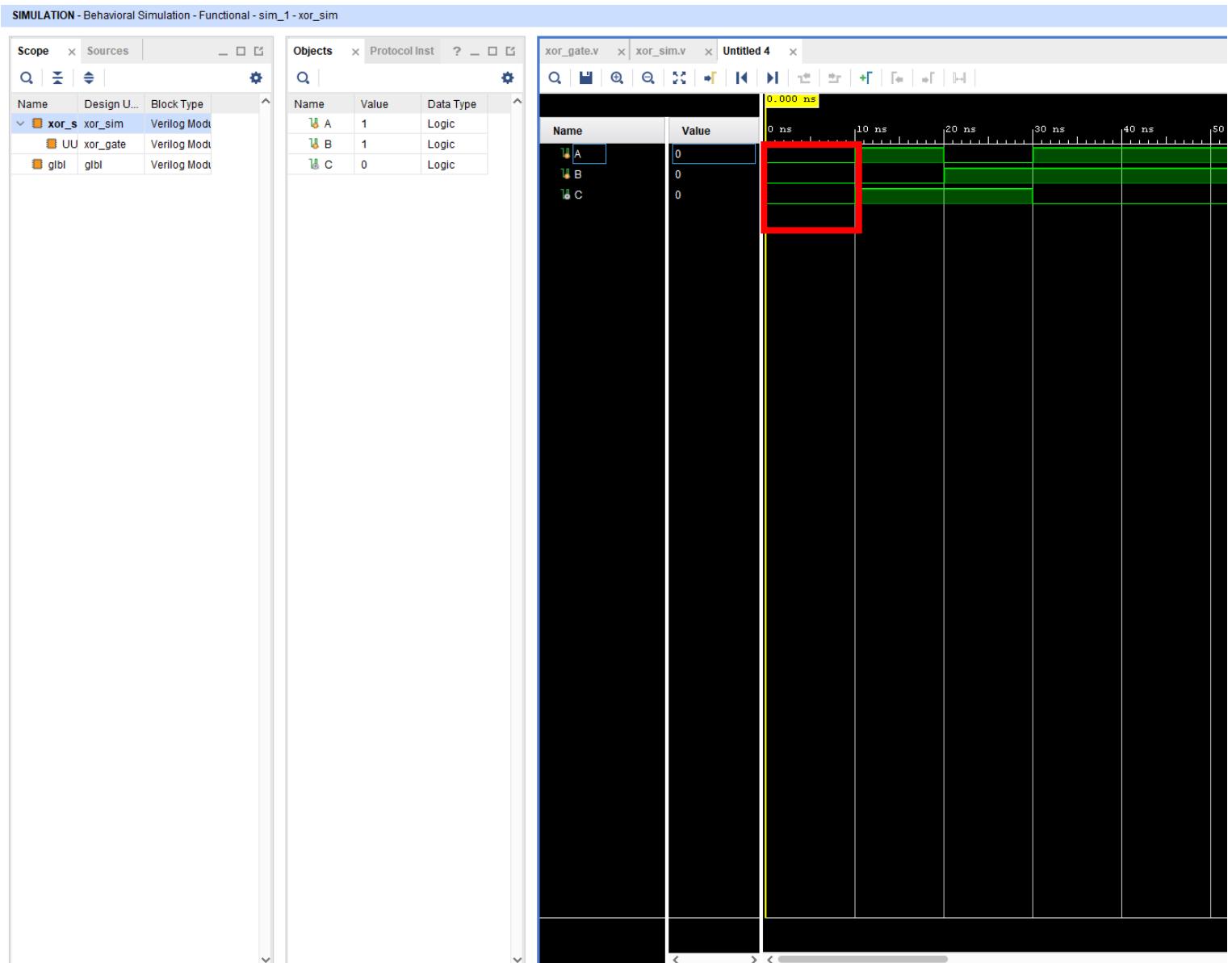
Simulation

- You can bring **yellow cursor** to the start of the simulation by clicking on the waveform.
- You can use **Zoom In** button to enlarge the area around the yellow cursor.



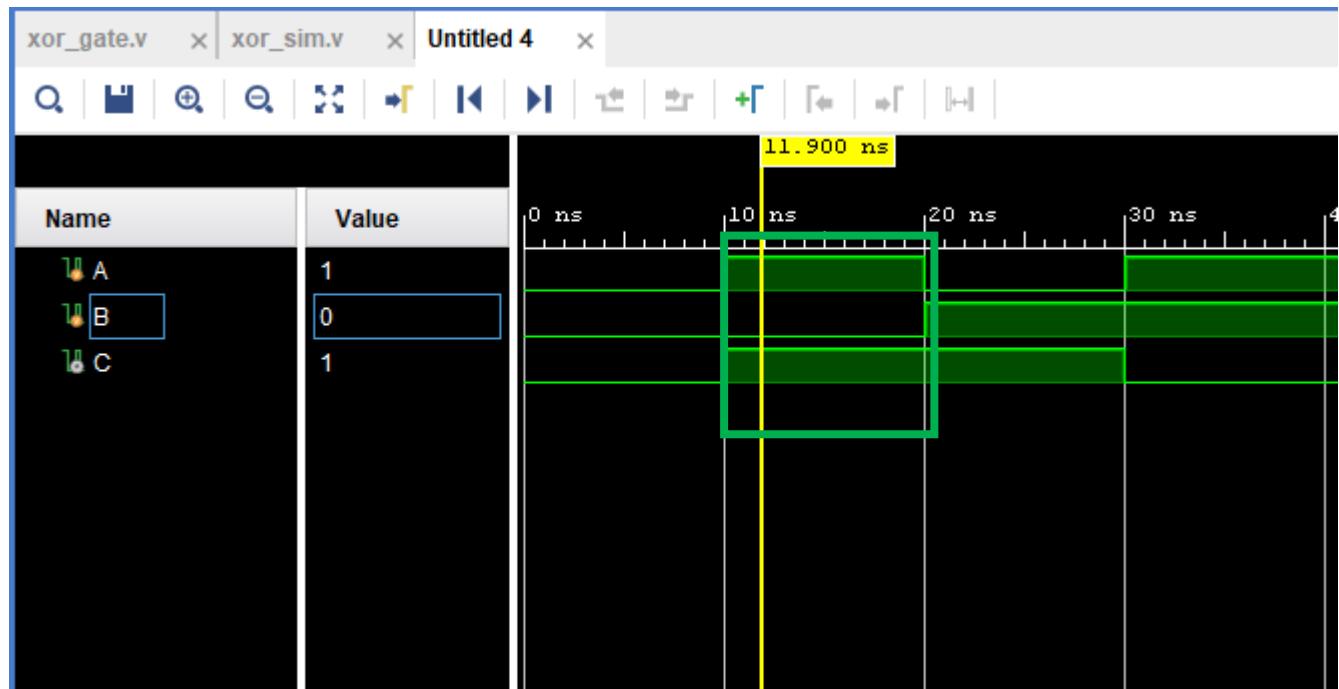
Simulation

- \oplus is XOR operation.
- $A \oplus B = C$
- $0 \oplus 0 = 0$



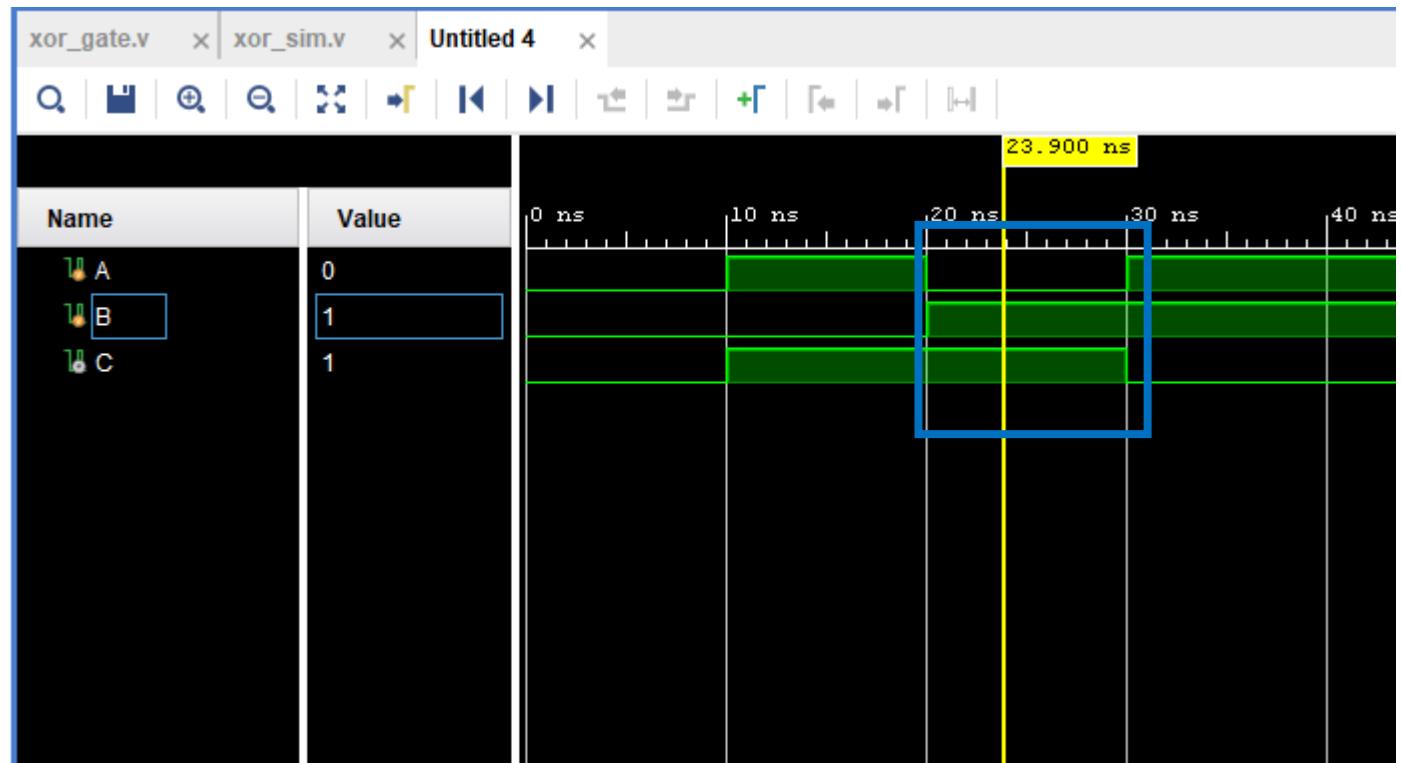
Simulation

- \oplus is XOR operation.
- $A \oplus B = C$
- $0 \oplus 0 = 0$
- $1 \oplus 0 = 1$



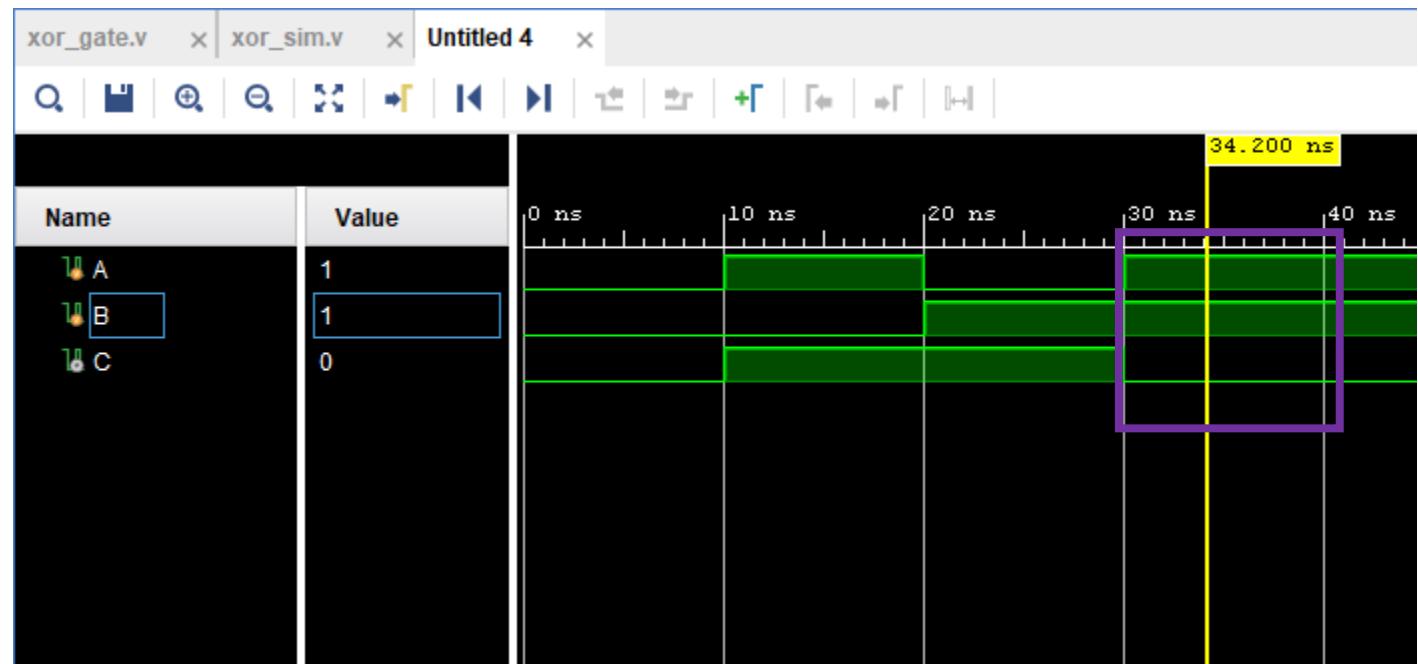
Simulation

- \oplus is XOR operation.
- $A \oplus B = C$
- $0 \oplus 0 = 0$
- $1 \oplus 0 = 1$
- $0 \oplus 1 = 1$



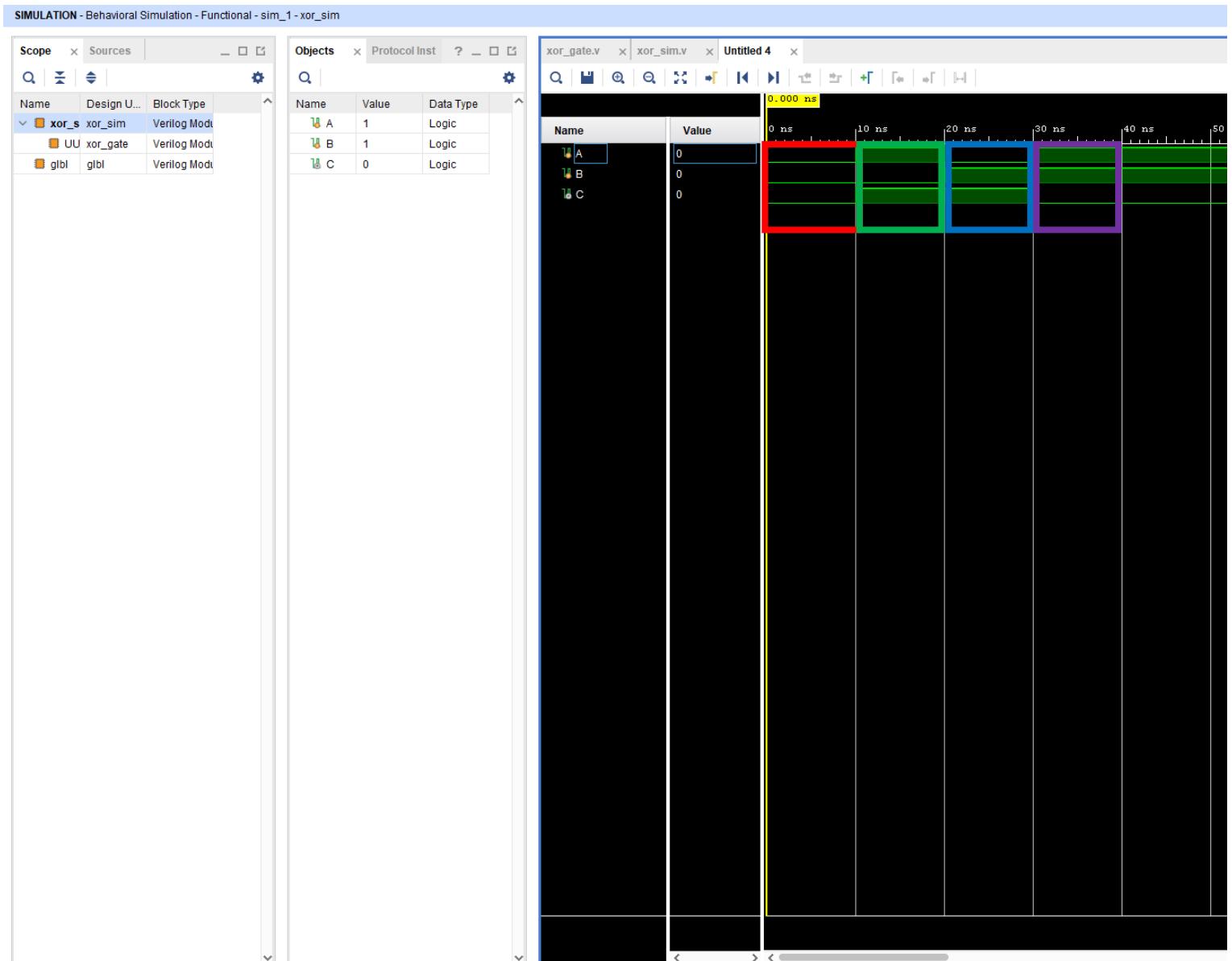
Simulation

- \oplus is XOR operation.
- $A \oplus B = C$
- $0 \oplus 0 = 0$
- $1 \oplus 0 = 1$
- $0 \oplus 1 = 1$
- $1 \oplus 1 = 0$



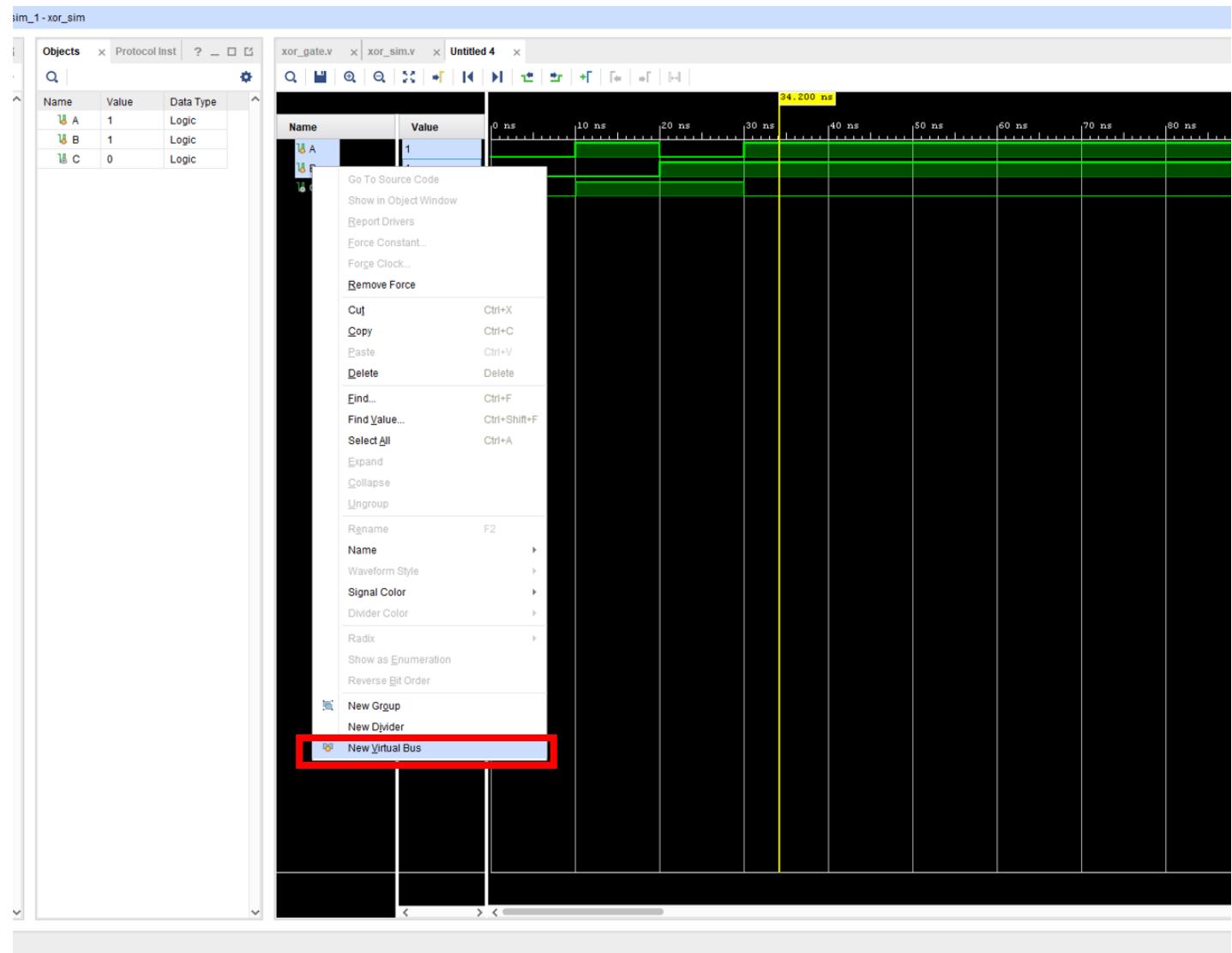
Simulation

- \oplus is XOR operation.
- $A \oplus B = C$
- $0 \oplus 0 = 0$
- $1 \oplus 0 = 1$
- $0 \oplus 1 = 1$
- $1 \oplus 1 = 0$



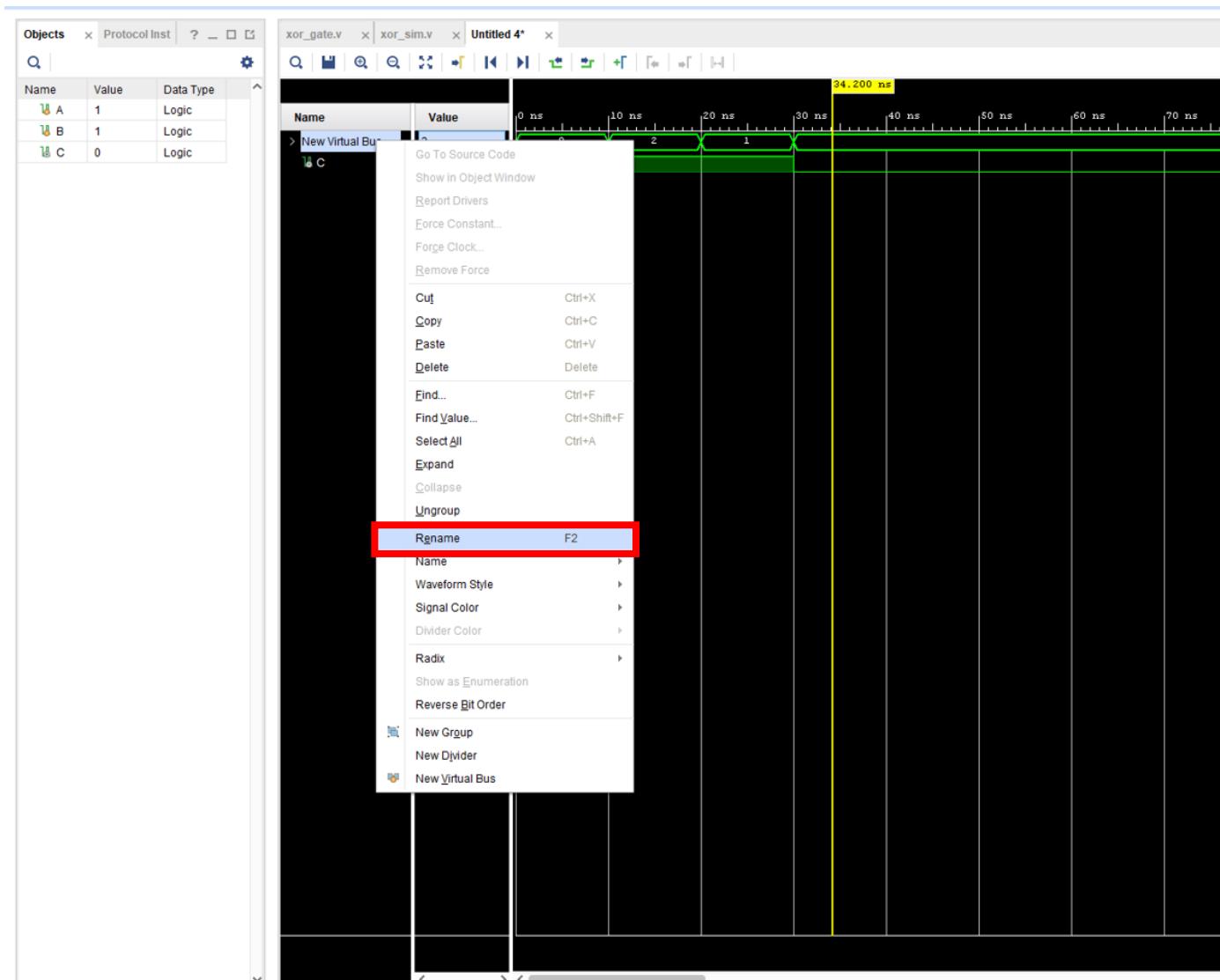
Simulation

- For better visualization, you may group inputs (and/or outputs) via **New Virtual Bus** option



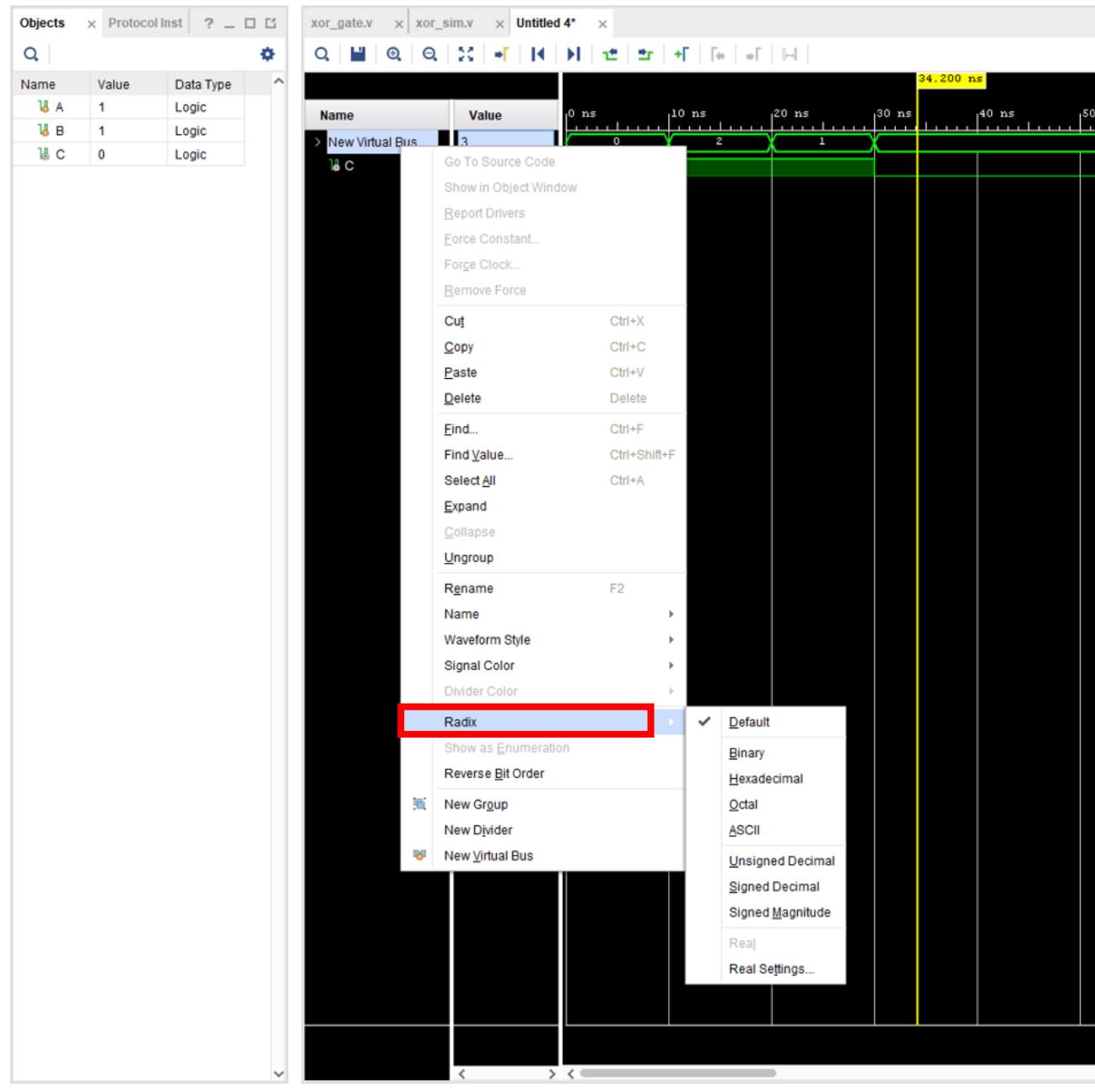
Simulation

- You may **Rename** your virtual bus



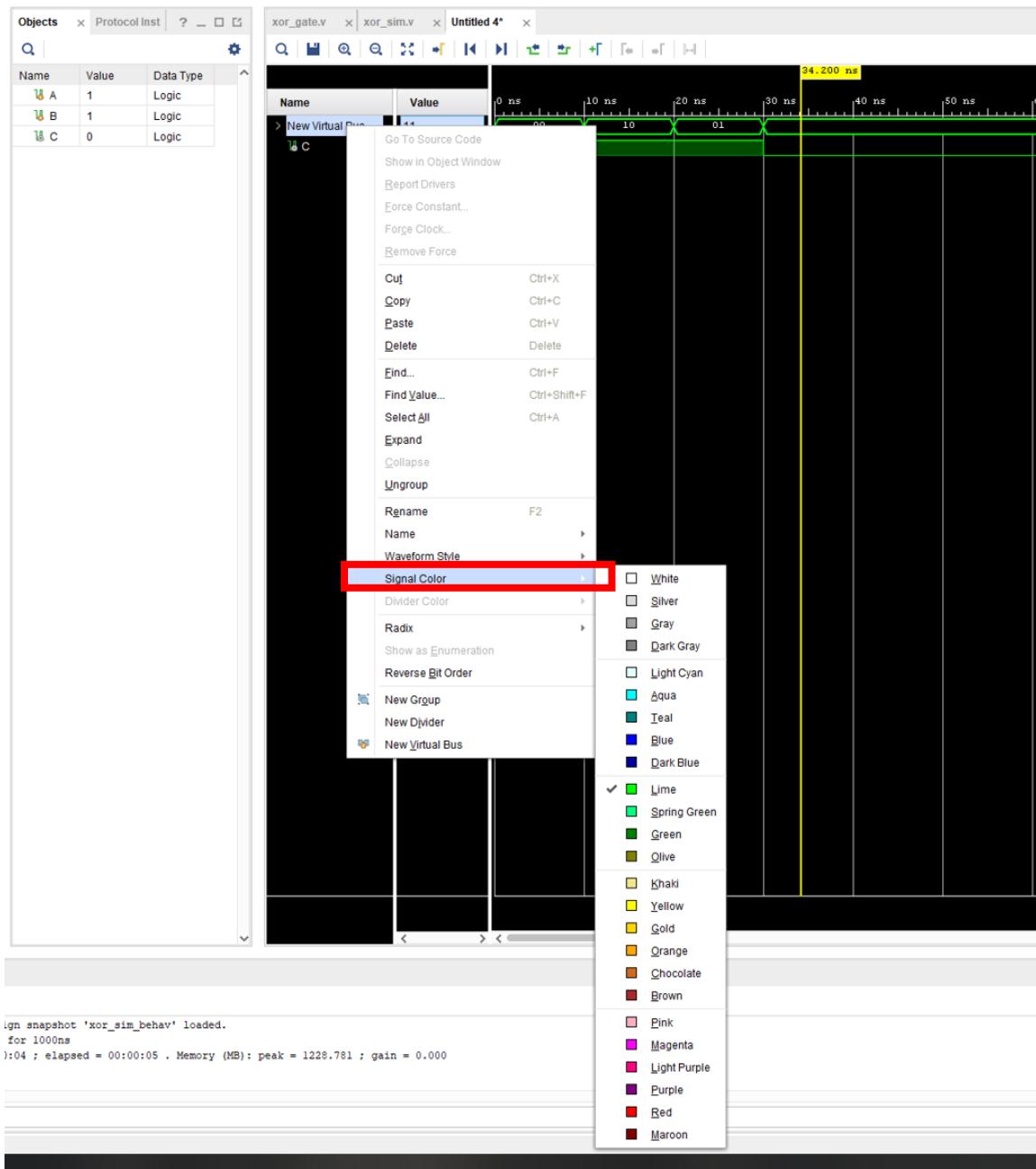
Simulation

- You may change **Radix** of virtual bus



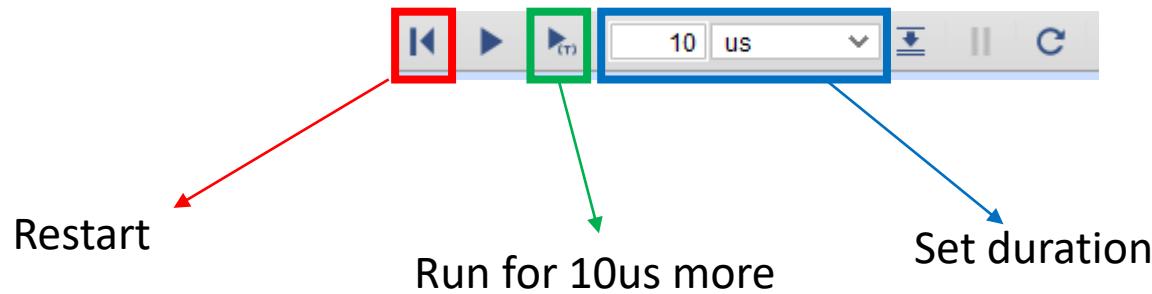
Simulation

- You may change **Signal Color** for better visualization



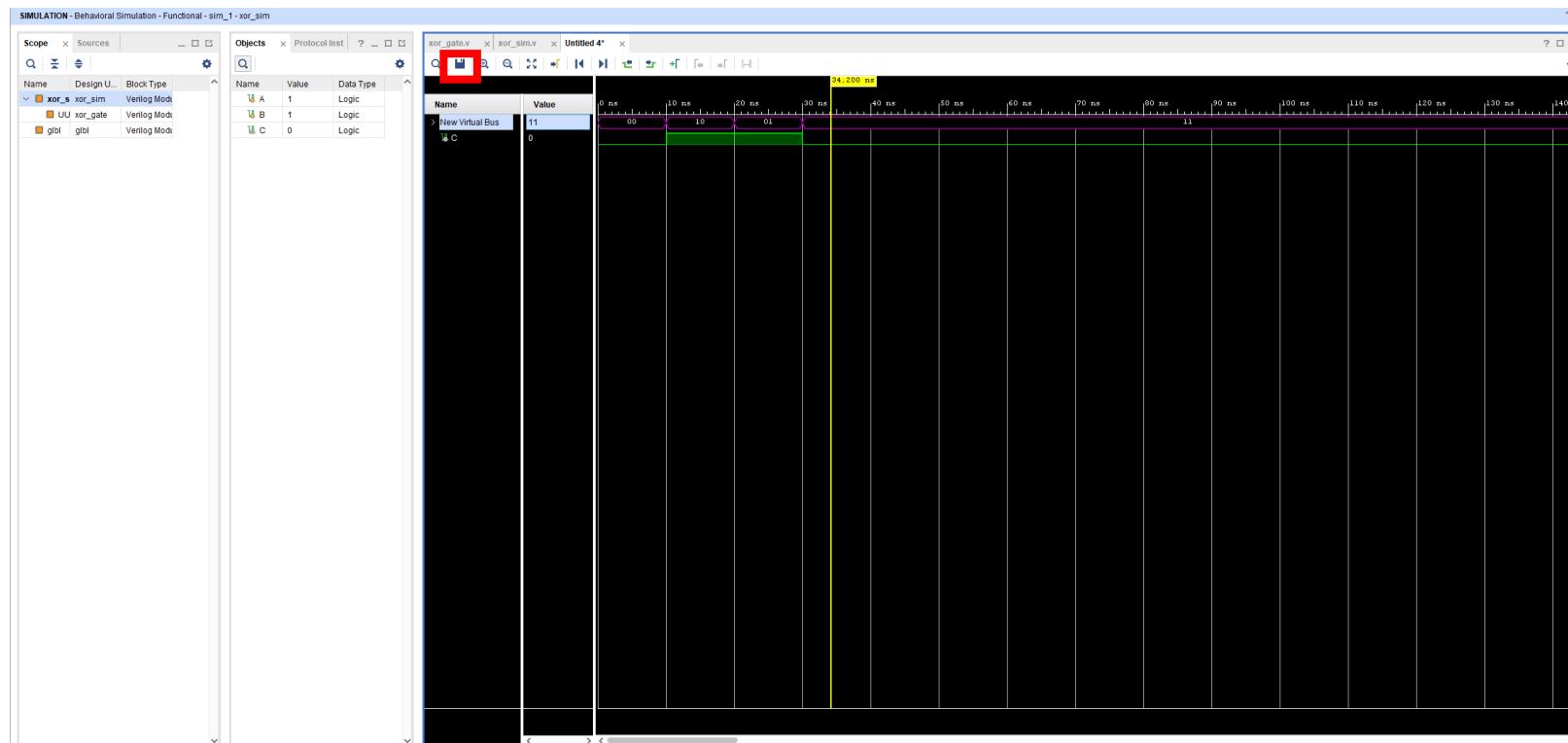
Simulation

- You can also change the duration of the corresponding simulation.



Simulation

- After you adjust the waves, you may save the waveform to use later.
- Save Waveform Configuration



Simulation

- Give a name to **wave configuration**
- **Save it**

