



ELEC 204 Digital System Design  
Laboratory Manual

---

**Experiment #1**  
**Introductory Tutorial For Xilinx ISE v14.7**  
**& Implementing XOR Gate**

---

DEPARTMENT OF ELECTRICAL & ELECTRONICS  
ENGINEERING  
COLLEGE OF ENGINEERING  
KOÇ UNIVERSITY

©2017

## Contents

<b>1 Objectives</b>	<b>2</b>
<b>2 Equipment &amp; Software</b>	<b>2</b>
<b>3 Procedure</b>	<b>2</b>
<b>4 Assessment</b>	<b>3</b>
<b>5 Introductory Tutorial for Xilinx ISE Foundation v14.7</b>	<b>4</b>
5.1 Design	4
Step 1 Design a logic circuit such as the sample circuit in Fig. 1.	4
5.2 Design Entry	4
5.2.1 Creating New Project:	4
Step 2 Launching Xilinx ISE Design Tools	4
Step 3 Creating project in Xilinx ISE Design Tools	5
Step 4 Specifying FPGA device properties	6
Step 5 Viewing project summary	7
5.2.2 Adding VHDL Source File:	8
Step 6 Adding new source file	8
Step 7 Specifying input and output ports	9
Step 8 Writing VHDL code in the newly created source file	10
5.3 Simulating Device Behavior	13
Step 9 Creating a new test bench waveform source file	13
Step 10 Associating the new test bench waveform file with the VHDL code	14
Step 11 Adjusting simulation parameters	15
Step 12 Run simulation	16
Step 13 Analyze simulation	17
5.4 Synthesize and Map Design	18
Step 14 Creating a new implementation constraints file	18
Step 15 Setting the name for implementation constraints file	19
Step 16 Assigning Pins to Inputs and Outputs	20
Step 17 Generating Programming File	22
5.5 Program Hardware	23
Step 18 Launching Prometheus-Fireprog Software	23
Step 19 Programming the FPGA Board	24
5.6 Test Hardware	25
Step 20 Testing the Design	25
<b>6 Implementing XOR Gate</b>	<b>26</b>

# 1 Objectives

- Becoming familiar with the Xilinx ISE software package:
  - Creating project using Xilinx ISE Project Navigator,
  - Writing VHDL code for Prometheus FPGA board,
  - Simulating VHDL code behavior.
- Becoming familiar with Prometheus FPGA board:
  - Loading (flashing) your designed project to the FPGA board,
  - Testing your design on the FPGA board.
- Implementing the exclusive “OR” (XOR) gate in VHDL.

# 2 Equipment & Software

- IBM compatible PC with Windows 7 operating system,
- Xilinx ISE v14.7 & Prometheus software packages,
- Prometheus FPGA (Xilinx Spartan 3A) board,
- USB cable for programming.

# 3 Procedure

In this first experiment, we'll define general steps of the digital design and exemplify steps of this design procedure. Throughout the semester we will be following the same design procedure for all our lab work. The digital design procedure includes the following 6 tasks:

1. **Design:** Define inputs, outputs and design the logic from inputs to outputs. Design task is exemplified in section 5.1 using the logic circuit with inputs X and Y, and output Z.
2. **Design Entry:** Enter gates and blocks to build your design using hardware description language in to the software tool. Design entry task is exemplified in section 5.1 using Xilinx ISE Design Tools through steps 2 to 8.
3. **Design Simulation:** Test your design in computer simulation. Revise your design if necessary by turning back to the design task. Design simulation task is exemplified in section 5.3 through steps 9 to 13.

4. **Synthesize and Map Design:** After a successful simulation, map inputs and outputs of the design to I/O pins of the hardware. Synthesize and map design task is exemplified in section 5.4 for the targeted Prometheus hardware through steps 14 to 17.
5. **Program Hardware:** Upload the final synthesized bit file to the hardware using a USB cable. Program hardware task is exemplified in section 5.5 for the targeted Prometheus hardware through steps 18 to 19.
6. **Test Hardware:** Verify the hardware functionality for your design. Revise your design if necessary by turning back to the design task. Test hardware task is exemplified in section 5.6 with step 20.

In the experiment 1, you are expected to exercise the above design procedure by

- Reading the introductory tutorial given in this document for Xilinx ISE software,
- Going through the steps of the tutorial and implement the given logic function, and
- Implementing the exclusive “OR” (XOR) gate in VHDL.

## 4 Assessment

1. Write a report summarizing the exact procedure you used to test your circuit.
2. Describe any problems you encountered, how you knew that the circuit was functioning properly.
3. Present the VHDL Code of your design.
4. Make your comments and state your conclusions for the experiment.

## 5 Introductory Tutorial for Xilinx ISE Foundation v14.7

This section describes design procedure of the digital circuits.

### 5.1 Design

In this tutorial we take a sample digital circuit design as shown in Fig. 1. We will follow the design procedure to implement this circuit using Xilinx ISE v10.2 software and deploy the final design in to the Digilent Prometheus Spartan 3A FPGA board.

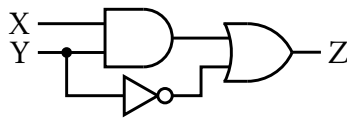


Figure 1: A sample logical circuit.

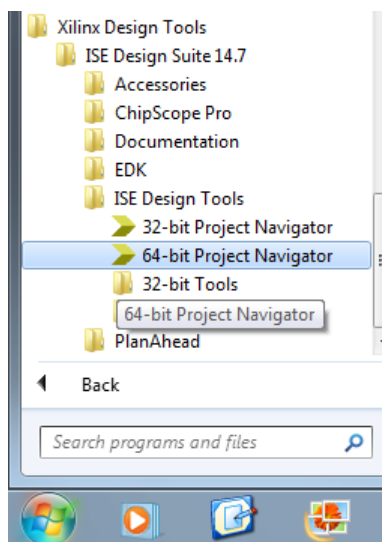
**Step 1 :** Design a logic circuit such as the sample circuit in Fig. 1.

### 5.2 Design Entry

#### 5.2.1 Creating New Project:

**Step 2 :** Launching Xilinx ISE Design Tools

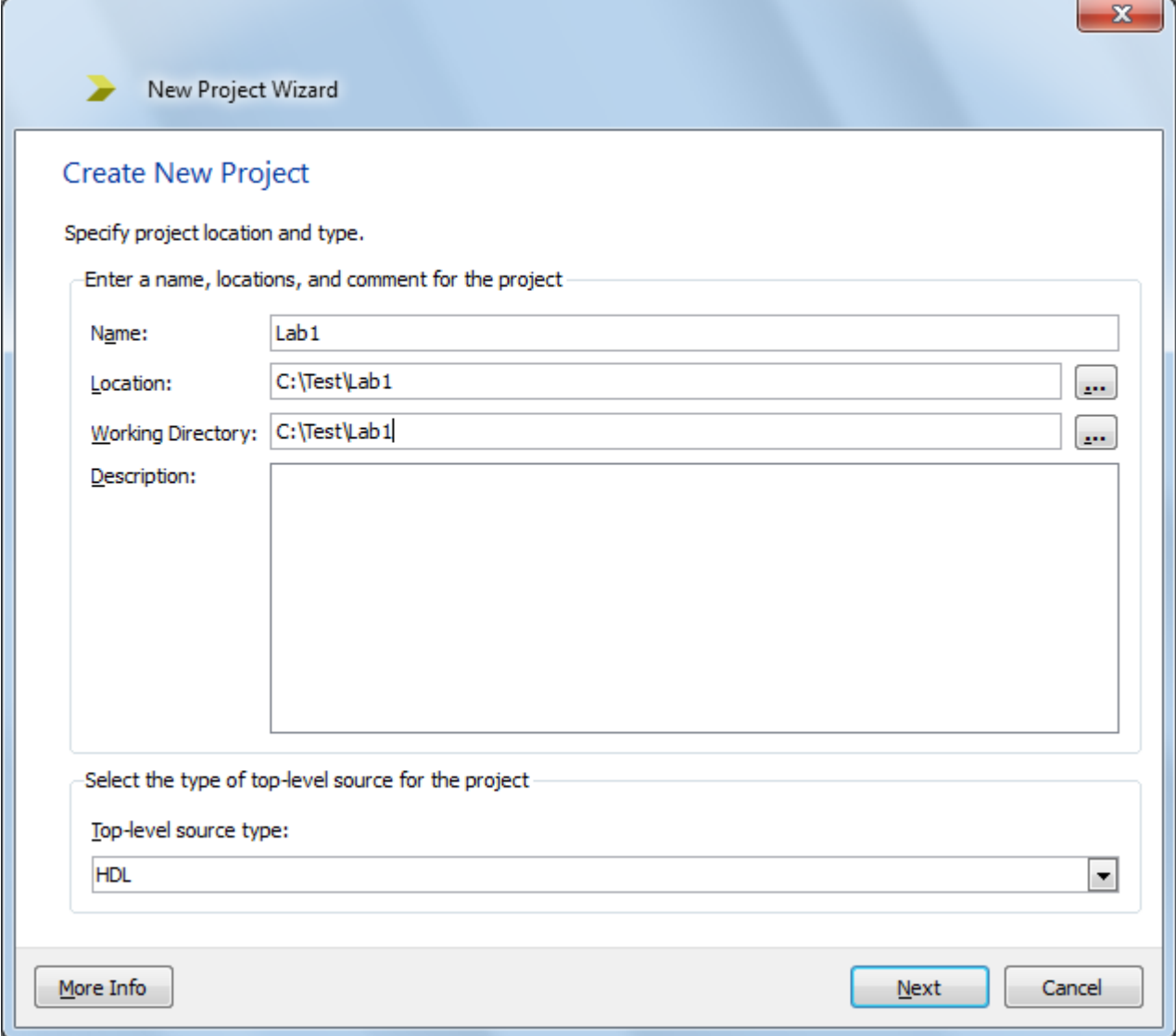
From the **Start** menu, launch Xilinx ISE Design Tools Project Navigator.



**Step 3 : Creating project in Xilinx ISE Design Tools**

From **File** menu, select **New Project** and hit **OK**.

Name the new project “Lab1”, then select Top-level source type “HDL” and click on **Next** button.



The image shows the 'New Project Wizard' dialog box in Xilinx ISE. The title bar says 'New Project Wizard'. The main heading is 'Create New Project'. Below it, the instruction is 'Specify project location and type.' There are two sections. The first section, 'Enter a name, locations, and comment for the project', contains four fields: 'Name:' with the value 'Lab1', 'Location:' with the value 'C:\Test\Lab1', 'Working Directory:' with the value 'C:\Test\Lab1', and 'Description:' which is an empty text area. The second section, 'Select the type of top-level source for the project', contains a 'Top-level source type:' dropdown menu with 'HDL' selected. At the bottom, there are three buttons: 'More Info', 'Next', and 'Cancel'. The 'Next' button is highlighted with a blue border.

New Project Wizard

Create New Project

Specify project location and type.

Enter a name, locations, and comment for the project

Name: Lab1

Location: C:\Test\Lab1

Working Directory: C:\Test\Lab1

Description:

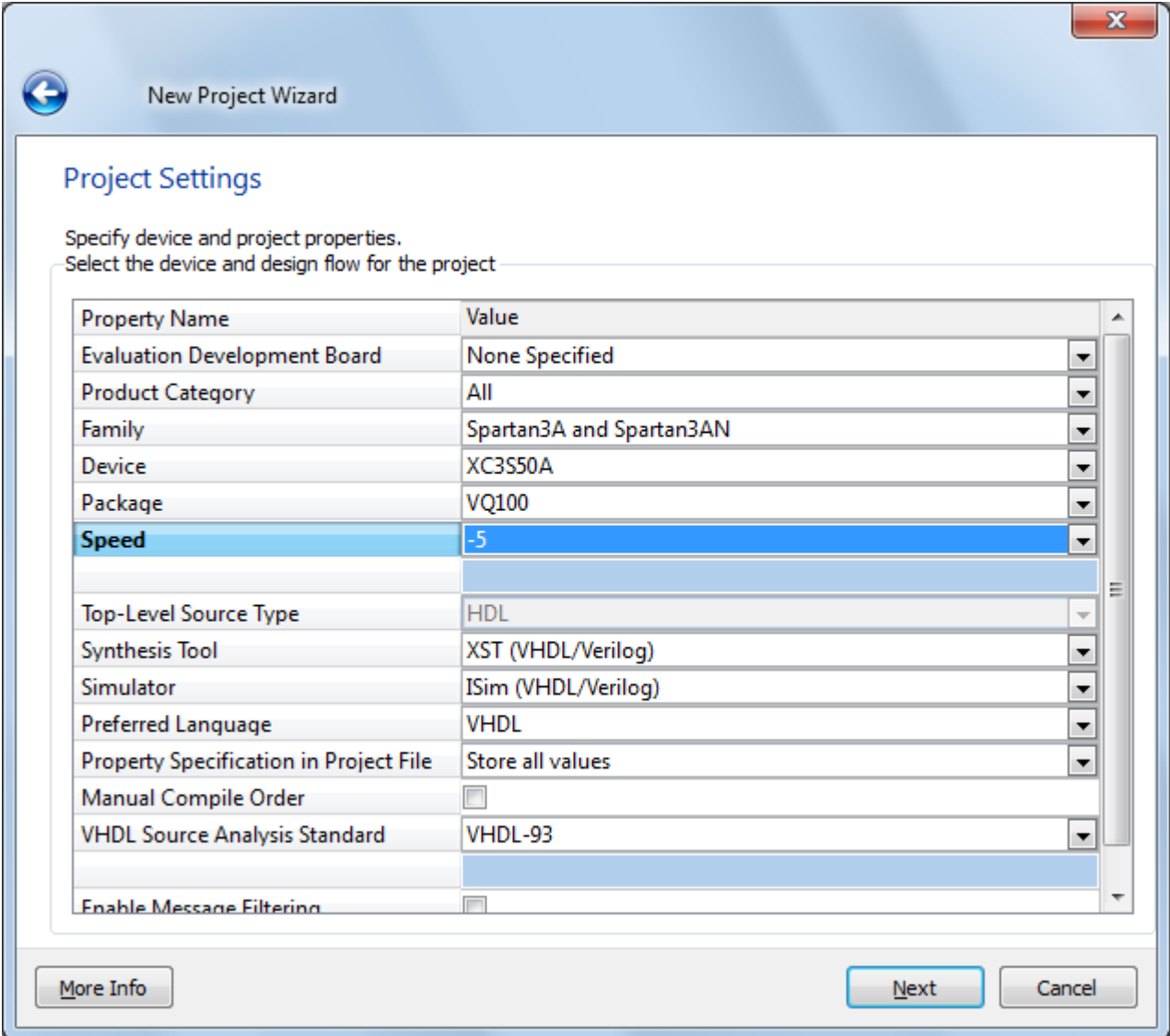
Select the type of top-level source for the project

Top-level source type: HDL

More Info Next Cancel

**Step 4 : Specifying FPGA device properties**

Change the contents of the boxes as shown below and click on **Next** button.

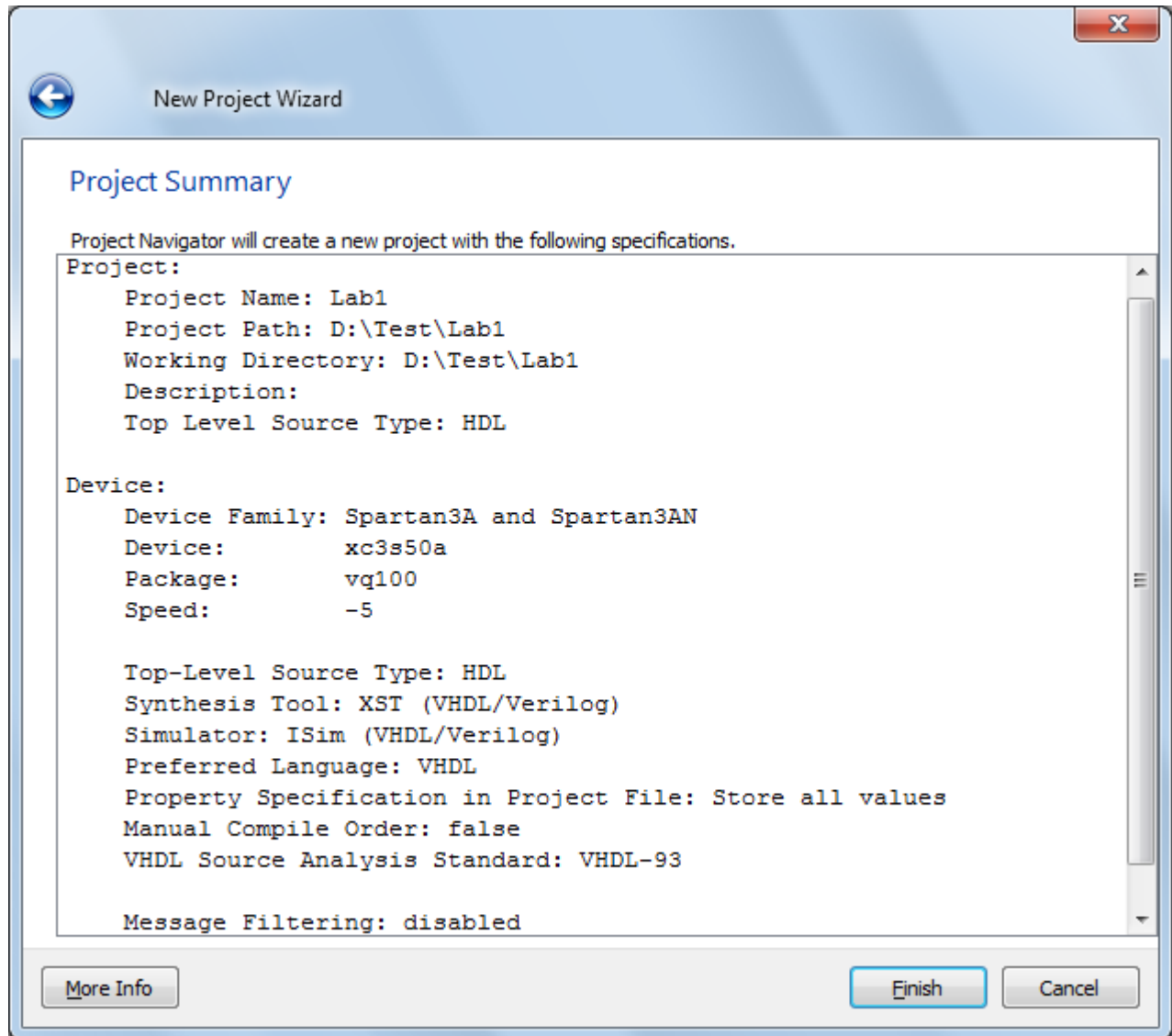


The image shows the 'New Project Wizard' dialog box, specifically the 'Project Settings' tab. The dialog has a title bar with a back arrow, the text 'New Project Wizard', and a close button. Below the title bar, the 'Project Settings' section is titled 'Specify device and project properties. Select the device and design flow for the project'. It contains a table with two columns: 'Property Name' and 'Value'. The 'Speed' property is highlighted in blue. At the bottom of the dialog, there are three buttons: 'More Info', 'Next', and 'Cancel'.

Property Name	Value
Evaluation Development Board	None Specified
Product Category	All
Family	Spartan3A and Spartan3AN
Device	XC3S50A
Package	VQ100
<b>Speed</b>	<b>-5</b>
Top-Level Source Type	HDL
Synthesis Tool	XST (VHDL/Verilog)
Simulator	ISim (VHDL/Verilog)
Preferred Language	VHDL
Property Specification in Project File	Store all values
Manual Compile Order	<input type="checkbox"/>
VHDL Source Analysis Standard	VHDL-93
Enable Message Filtering	<input type="checkbox"/>

**Step 5 : Viewing project summary**

Check the project summary and click on **Finish** button.

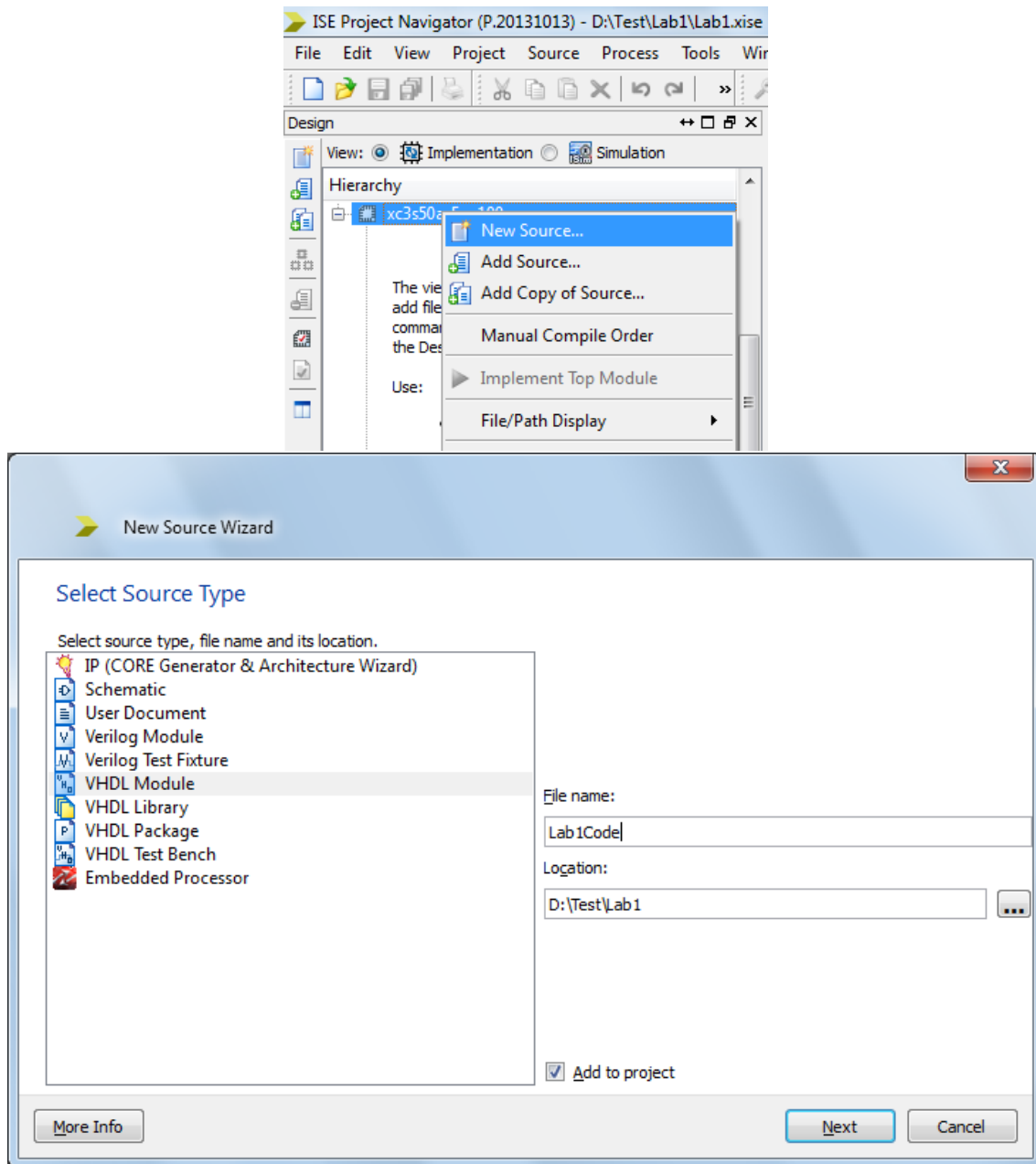




### 5.2.2 Adding VHDL Source File:

Right-click on **xc3s50a-5vq100** device and press “New Source”. Select “VHDL module” in the popped up window.

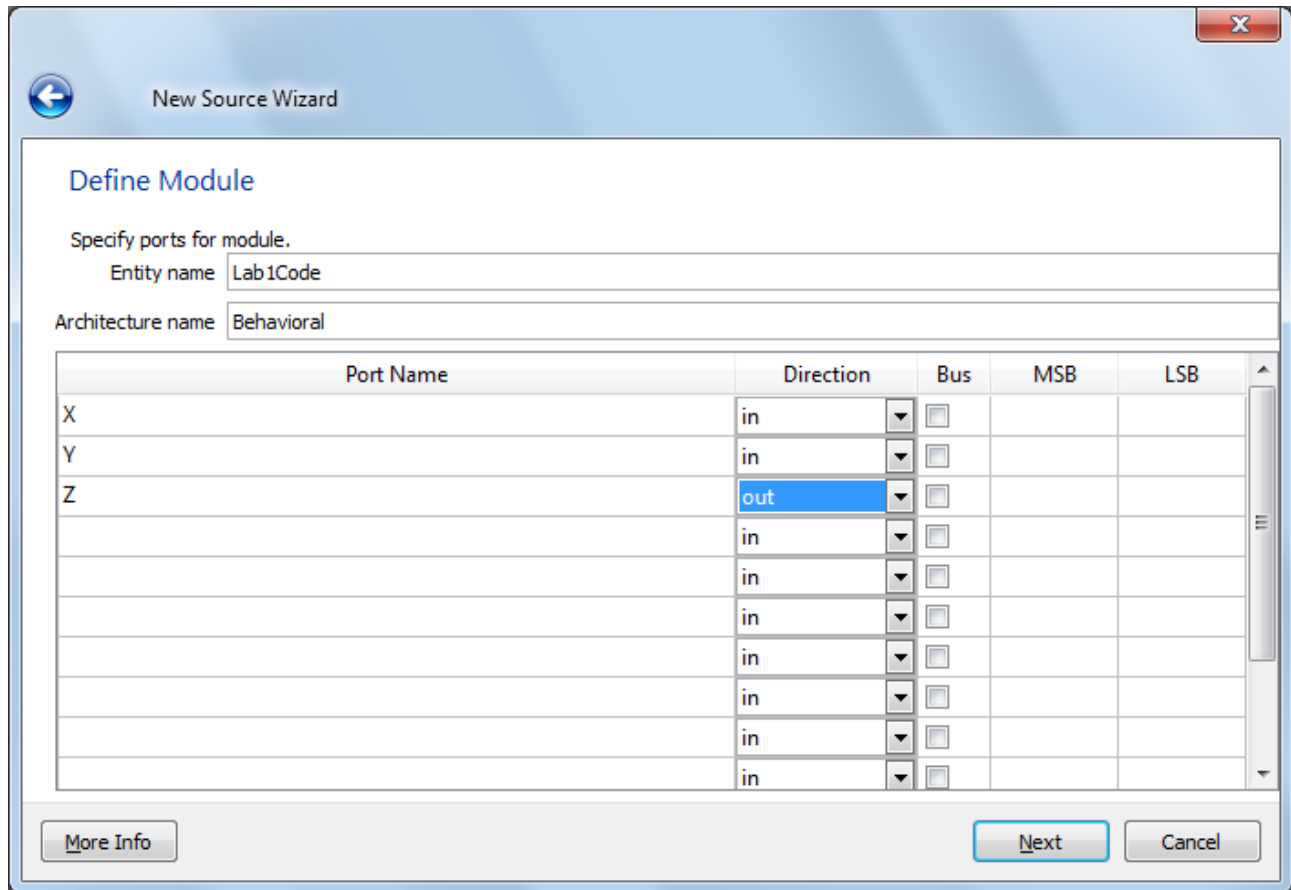
**Step 6 :** Adding new source file



**Step 7 : Specifying input and output ports**

Give names to ports which will be used in the implementation. Specify their type (input-output) using pop-up menus. Then click on the **Next** button.

If you are not sure about the number of input or output ports, you can skip this step. Then, you will write the corresponding VHDL lines into your code to specify the names and number of input-output ports.



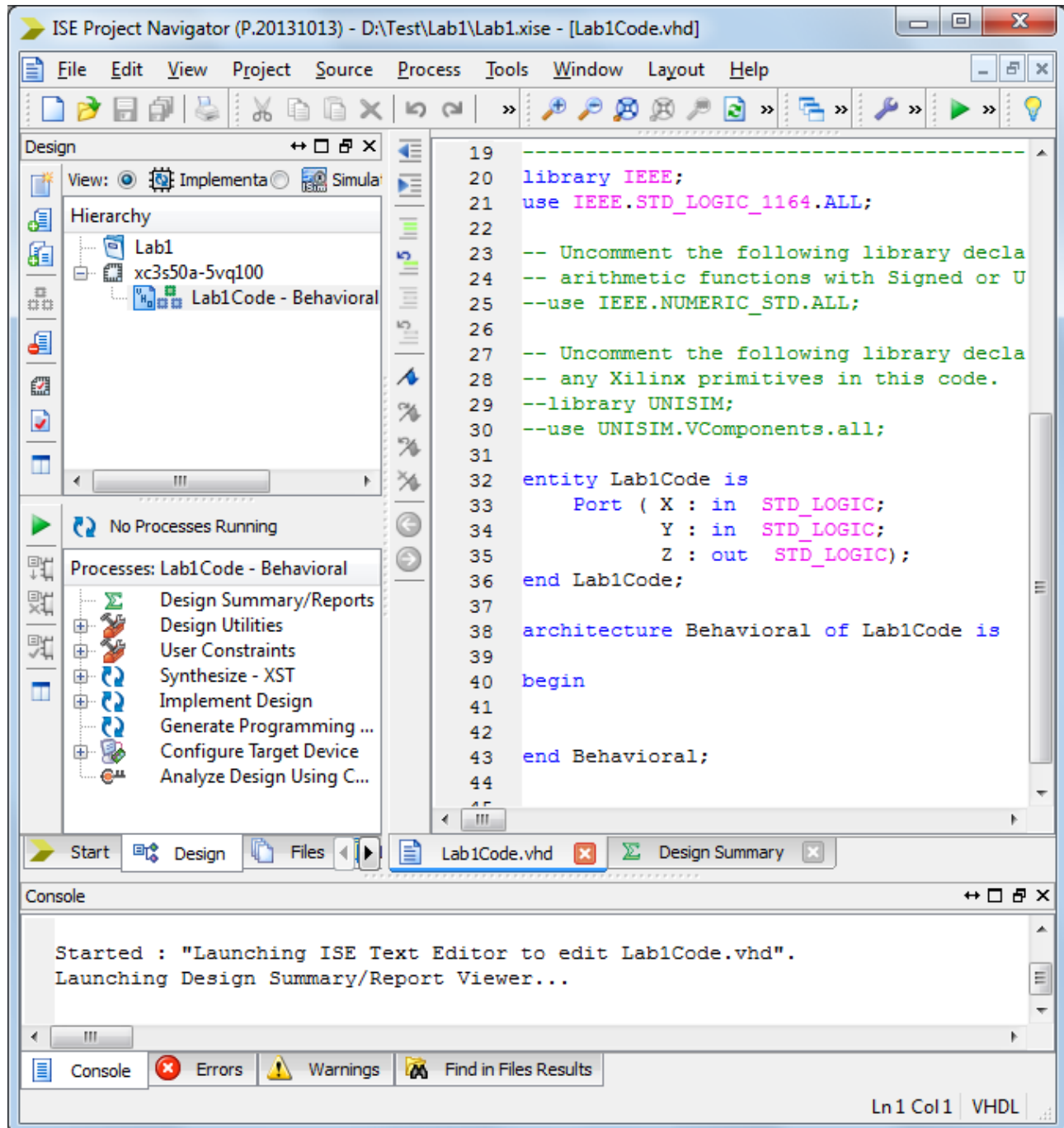
The image shows a screenshot of the "New Source Wizard" dialog box, specifically the "Define Module" step. The dialog has a title bar with a back arrow icon and the text "New Source Wizard". Below the title bar, the text "Specify ports for module." is displayed. There are two text input fields: "Entity name" with the value "Lab1Code" and "Architecture name" with the value "Behavioral". Below these fields is a table with five columns: "Port Name", "Direction", "Bus", "MSB", and "LSB". The table contains several rows. The first row has "X" in the "Port Name" column, "in" in the "Direction" column, and an unchecked checkbox in the "Bus" column. The second row has "Y" in the "Port Name" column, "in" in the "Direction" column, and an unchecked checkbox in the "Bus" column. The third row has "Z" in the "Port Name" column, "out" in the "Direction" column, and an unchecked checkbox in the "Bus" column. The "out" text is highlighted in blue. Below the table, there are three buttons: "More Info", "Next", and "Cancel". The "Next" button is highlighted in blue.

Port Name	Direction	Bus	MSB	LSB
X	in	<input type="checkbox"/>		
Y	in	<input type="checkbox"/>		
Z	out	<input type="checkbox"/>		
	in	<input type="checkbox"/>		
	in	<input type="checkbox"/>		
	in	<input type="checkbox"/>		
	in	<input type="checkbox"/>		
	in	<input type="checkbox"/>		
	in	<input type="checkbox"/>		
	in	<input type="checkbox"/>		

**Step 8 :** Writing VHDL code in the newly created source file

In the **Sources** window, under **Implementation** choose configured target device (**xc3s50a-5vq100**). Then double click on the file name **Lab1Code.vhd**.

Analyze the circuit shown in Figure 2. Follow the steps regarding signal definition in VHDL on the next page.



As shown in Figure 2, the circuit has two input signals {X,Y}, two intermediate signals {XY, NY} and one output signal Z. Corresponding VHDL code for the circuit is shown in Listing 1.

Input and output signals are defined in the **Entity** section of the code (lines 30-34 of Listing 1). STD\_LOGIC corresponds to a single bit signal.

Intermediate signals do not have input and output ports. These signals facilitate code writing by partitioning the problem into subproblems, usually describing outputs of numerous logic gates. Intermediate signals are defined in the **Architecture** section (lines 36-38 of Listing 1).

Actual values of intermediate signals are defined in the **Behavioral** section of the code (lines 40-44 of Listing 1). For example, node XY is an output of a binary AND gate fed by signals X and Y. Corresponding VHDL representation is given below:

$$XY \leq X \text{ and } Y;$$

Note the “<=” symbol defining the signal and the “;” symbol at the end of the statement.

Inverter gate has only one input and it is used as follows:

$$NY \leq \text{not } Y;$$

Output signal Z is an addition of signals XY and NY. Corresponding VHDL representation is given below:

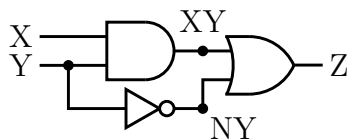
$$Z \leq XY \text{ or } NY;$$


Figure 2: A sample logical circuit with intermediate signals.

Be careful with syntax of the VHDL code and use template code generated by Xilinx. Now write the VHDL code as shown in Listing 1 and continue with the next steps.

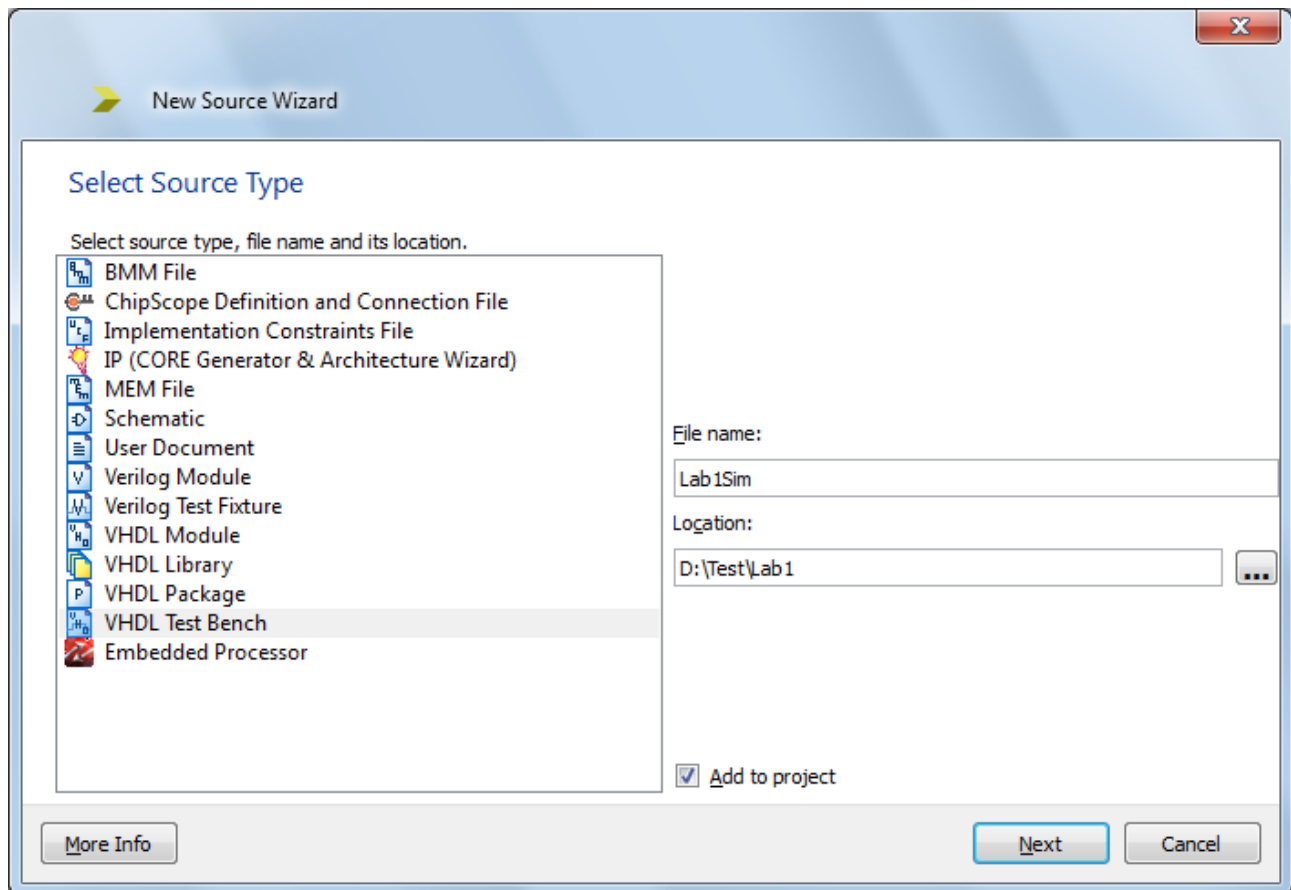
Listing 1: VHDL code for Experiment #1.

```
1  -----
2  -- Company:
3  -- Engineer:
4  --
5  -- Create Date:      12:10:08 02/09/2013
6  -- Design Name:
7  -- Module Name:      Lab1Code - Behavioral
8  -- Project Name:
9  -- Target Devices:
10 -- Tool versions:
11 -- Description:
12 --
13 -- Dependencies:
14 --
15 -- Revision:
16 -- Revision 0.01 - File Created
17 -- Additional Comments:
18 --
19 -----
20 library IEEE;
21 use IEEE.STD_LOGIC_1164.ALL;
22 use IEEE.STD_LOGIC_ARITH.ALL;
23 use IEEE.STD_LOGIC_UNSIGNED.ALL;
24
25 ---- Uncomment the following library declaration if instantiating
26 ---- any Xilinx primitives in this code.
27 --library UNISIM;
28 --use UNISIM.VComponents.all;
29
30 entity Lab1Code is
31     Port ( X : in  STD_LOGIC;  -- Input signal
32           Y : in  STD_LOGIC;  -- Input signal
33           Z : out STD_LOGIC); -- Output signal
34 end Lab1Code;
35
36 architecture Behavioral of Lab1Code is
37     signal XY : STD_LOGIC; -- Intermediate signal
38     signal NY : STD_LOGIC; -- Intermediate signal
39
40 begin
41     XY <= X and Y; -- Intermediate signals are defined here
42     NY <= not Y;
43     Z <= XY or NY; -- Output signal
44 end Behavioral;
```

### 5.3 Simulating Device Behavior

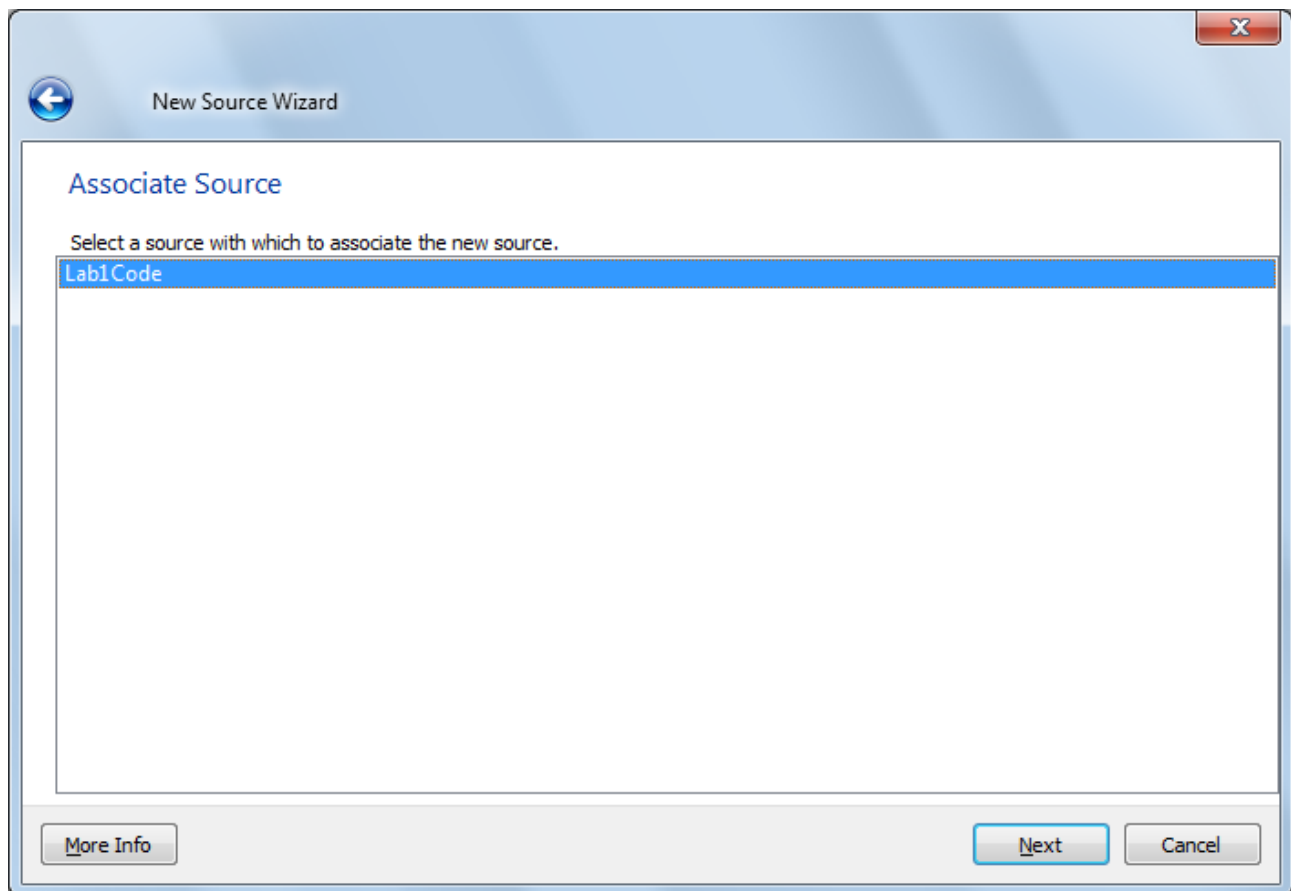
**Step 9 :** Creating a new test bench waveform source file

Using menus on the top of the main window, select **Project** → **New Source**. In the dialog box, select **VHDL Test Bench** as the source type.



**Step 10 :** Associating the new test bench waveform file with the VHDL code

Select previously created **Lab1Code.vhd** file to be associated with the new test bench waveform file.



**Step 11** : Adjusting simulation parameters

Modify the VHDL Test Bench file ("Lab1Sim.vhd") with the following content:

Listing 2: VHDL Test Bench Simulation for Experiment #1.

```

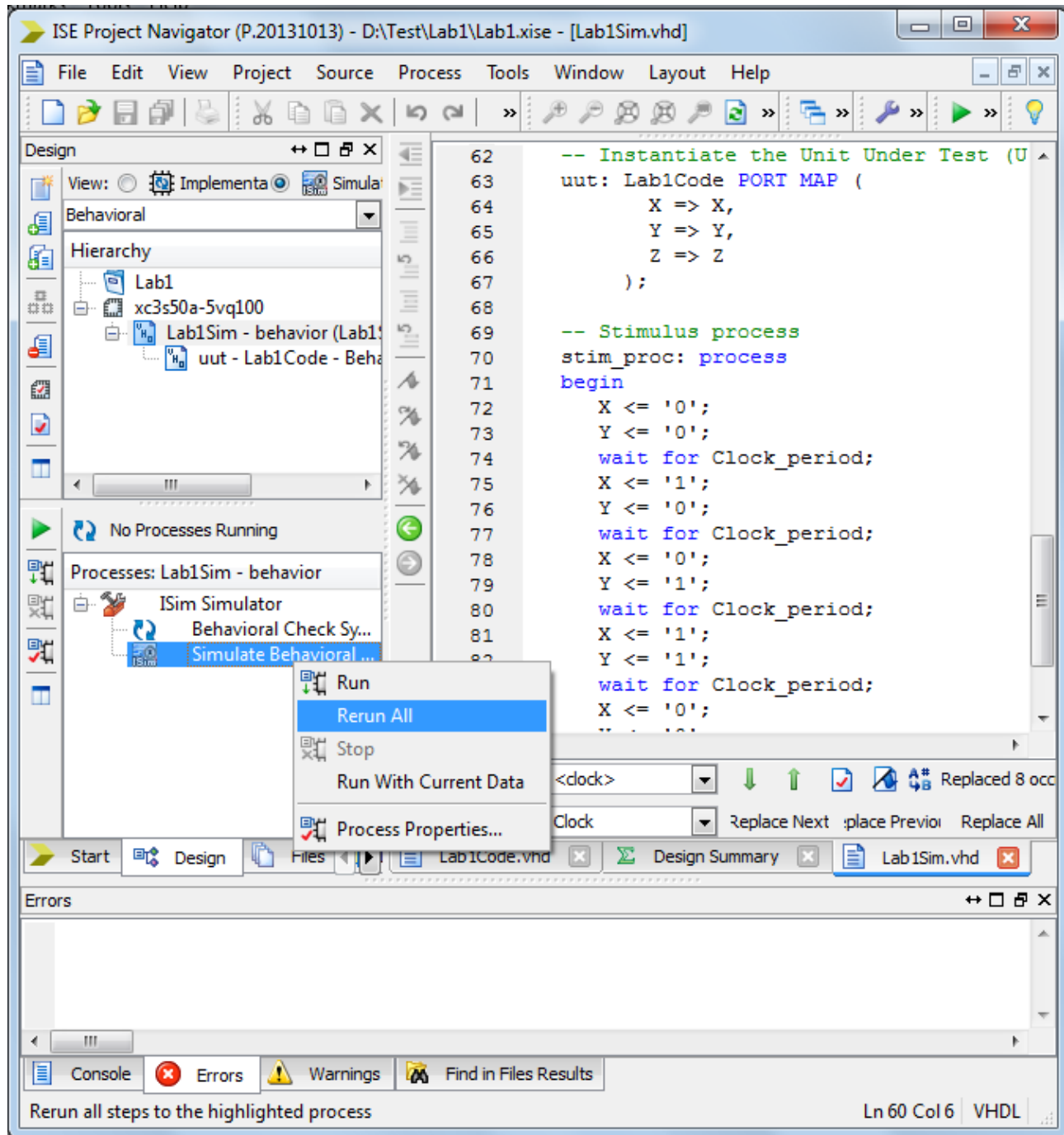
1  LIBRARY ieee ;
2  USE ieee.STD_LOGIC_1164.ALL;
3
4  ENTITY Lab1Sim IS
5  END Lab1Sim;
6
7  ARCHITECTURE behavior OF Lab1Sim IS
8      -- Component Declaration for the Unit Under Test (UUT)
9      COMPONENT Lab1Code
10     PORT(X : IN  STD_LOGIC;
11          Y : IN  STD_LOGIC;
12          Z : OUT STD_LOGIC);
13     END COMPONENT;
14     --Inputs
15     signal X : STD_LOGIC := '0';
16     signal Y : STD_LOGIC := '0';
17     --Outputs
18     signal Z : STD_LOGIC;
19     constant Clock_period : time := 10 ns;
20 BEGIN -- Instantiate the Unit Under Test (UUT)
21     uut: Lab1Code PORT MAP (
22         X => X,
23         Y => Y,
24         Z => Z);
25     stim_proc: process -- Stimulus process
26     begin
27         X <= '0'; Y <= '0';
28         wait for Clock_period;
29         X <= '1'; Y <= '0';
30         wait for Clock_period;
31         X <= '0'; Y <= '1';
32         wait for Clock_period;
33         X <= '1'; Y <= '1';
34         wait for Clock_period;
35         X <= '0'; Y <= '0'; -- Return to initial value
36         wait;
37     end process;
38 END;
```

The test bench has all possible values of the inputs. There are two inputs: X and Y, hence  $N = 2$  and number of possible combinations of X, Y is equal to  $2^N = 2^2 = 4$ , i.e.  $(X, Y) \in \{00, 01, 10, 11\}$ .



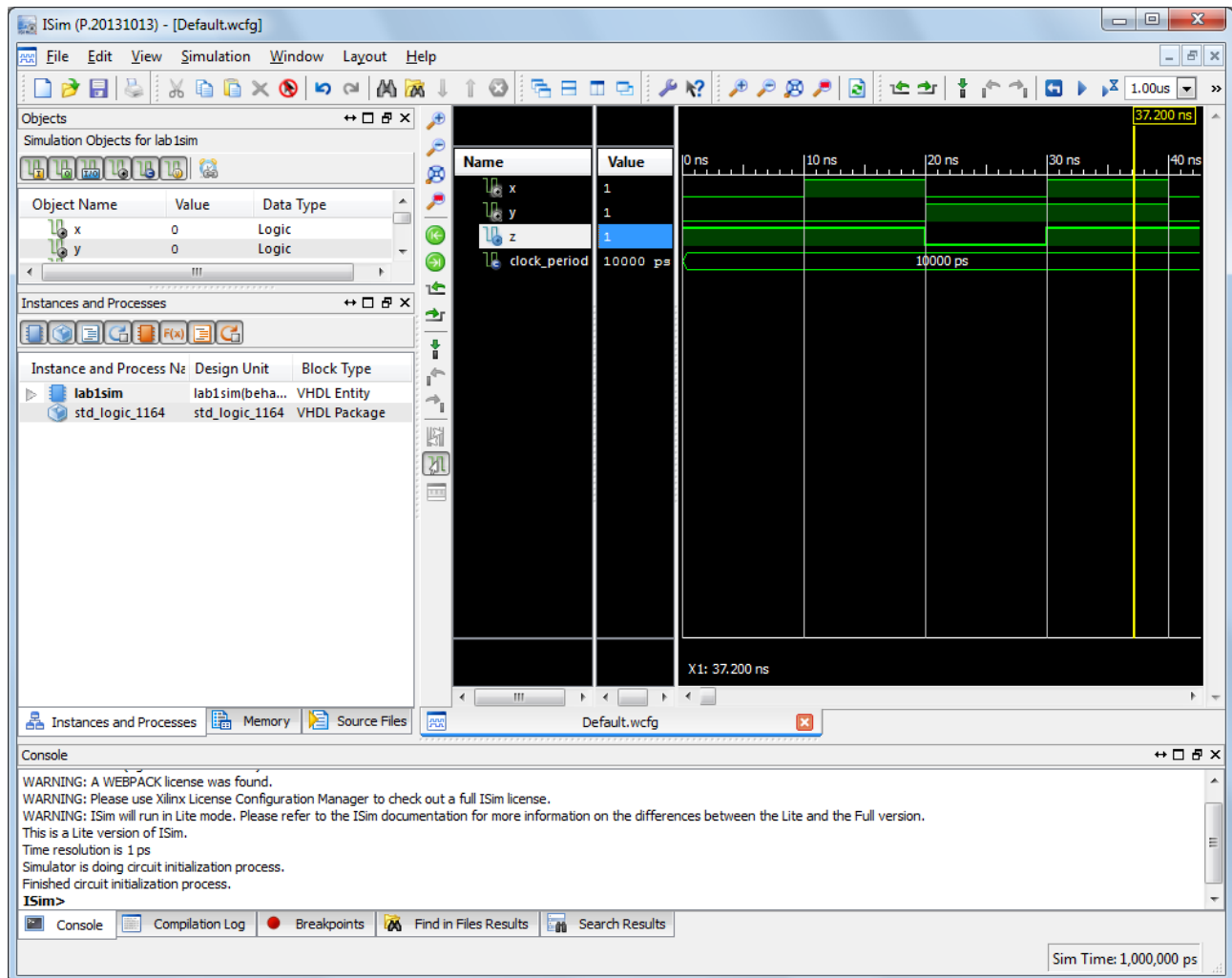
**Step 12 : Run simulation**

Go to **Processes** → **ISim Simulator**. Double click on the **Simulate Behavioral Model**. Wait until simulation ends.



**Step 13 : Analyze simulation**

Important! The scale of the simulation waveform is usually not suitable for displaying the signals. Use “Ctrl” key and scroll wheel of the mouse to zoom the simulation waveform. Also move horizontal slider so that a suitable range for the simulation waveform is achieved and all four combinations of the inputs are visible: Check whether for all four possible

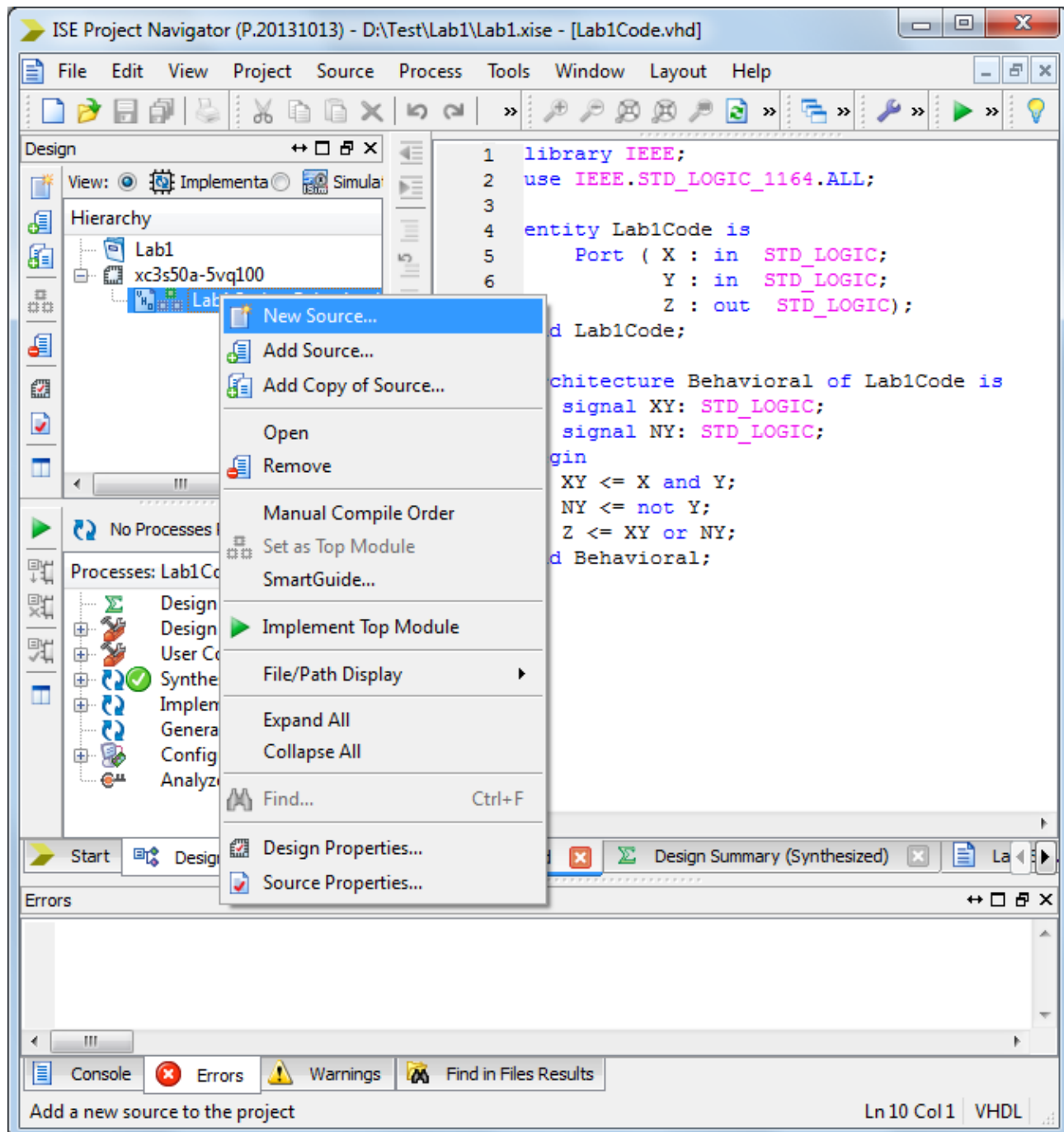


inputs of X and Y the output Z gives correct results.

## 5.4 Synthesize and Map Design

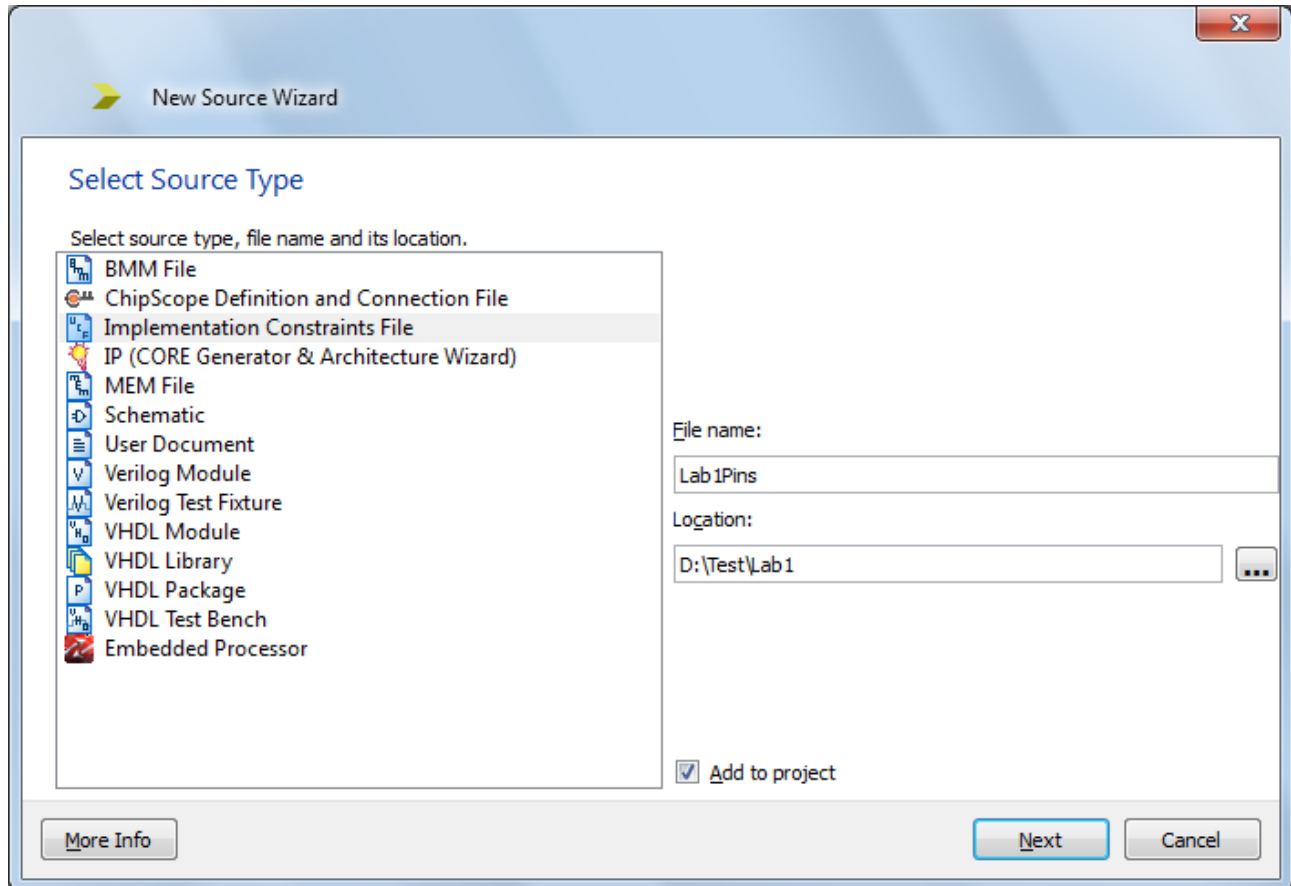
**Step 14 :** Creating a new implementation constraints file

Using menus on the top of the main window, select **Project** → **New Source**.



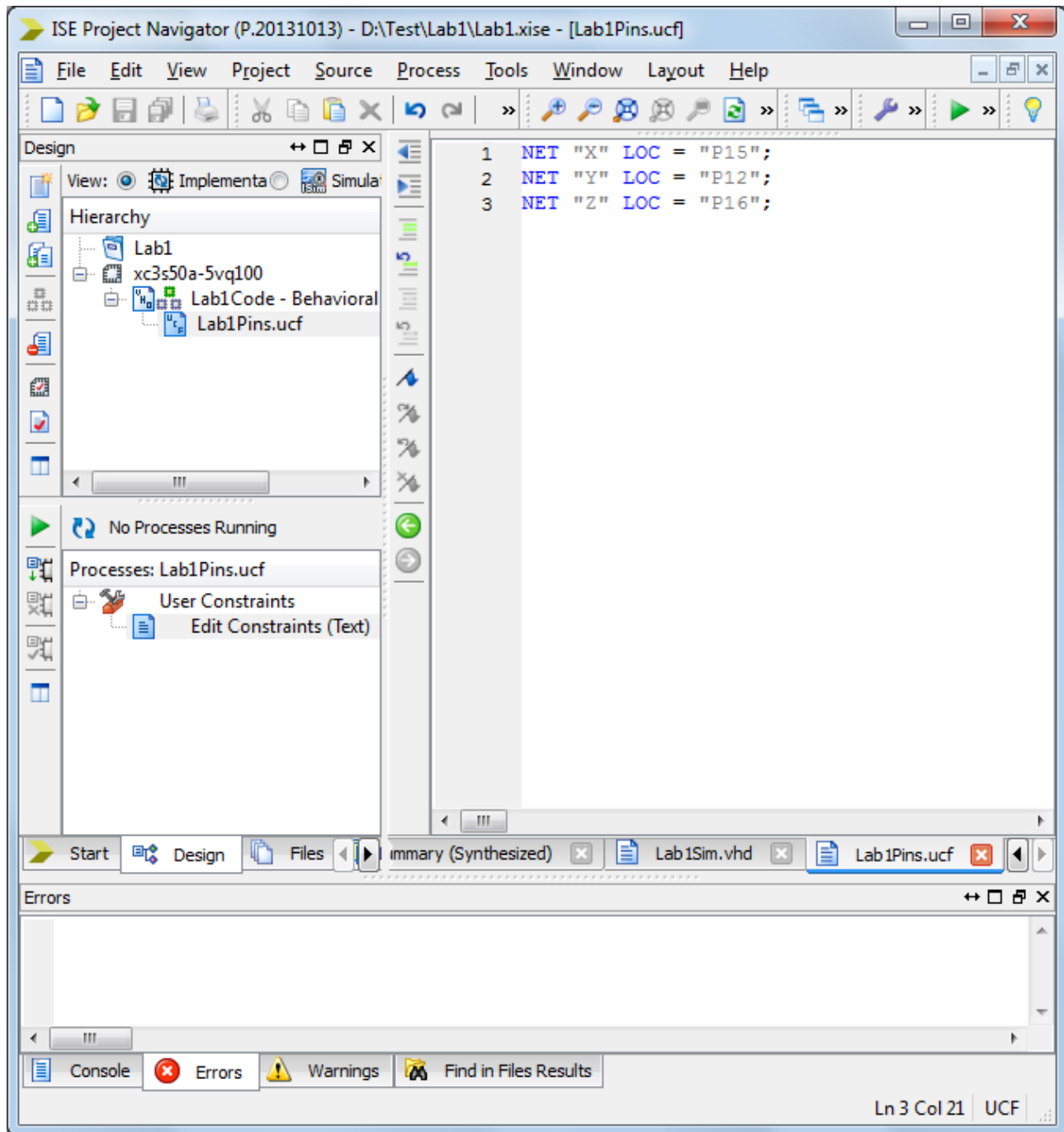
**Step 15 :** Setting the name for implementation constraints file

In the dialog box, select **Implementation Constraints File** as the source type. Set the file name to **Lab1Pins.ucf**.



**Step 16 : Assigning Pins to Inputs and Outputs**

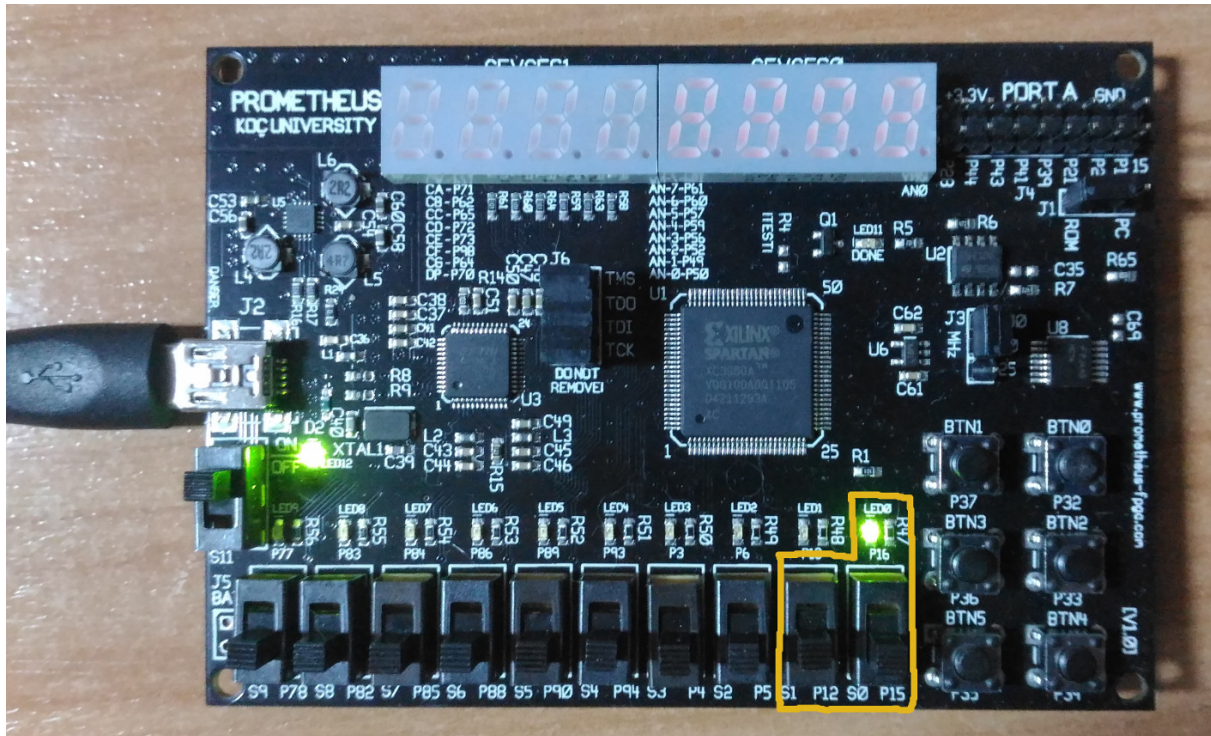
Implementation Constraint File (UCF) is especially important because it contains all pin-outs of the input and output signals of the board. A circuit synthesized without this file cannot communicate with an outside world (i.e. LEDs, switches etc.) Now you can set input and output pins to your ports X, Y and Z in the "Lab1Pins.ucf" file:



Physical appearance of the switches and the LEDs and their pins is provided below.

Component	Abbreviation	Pin #
Switch "0"	SW0	P15
Switch "1"	SW1	P12
Led "0"	LD0	P16

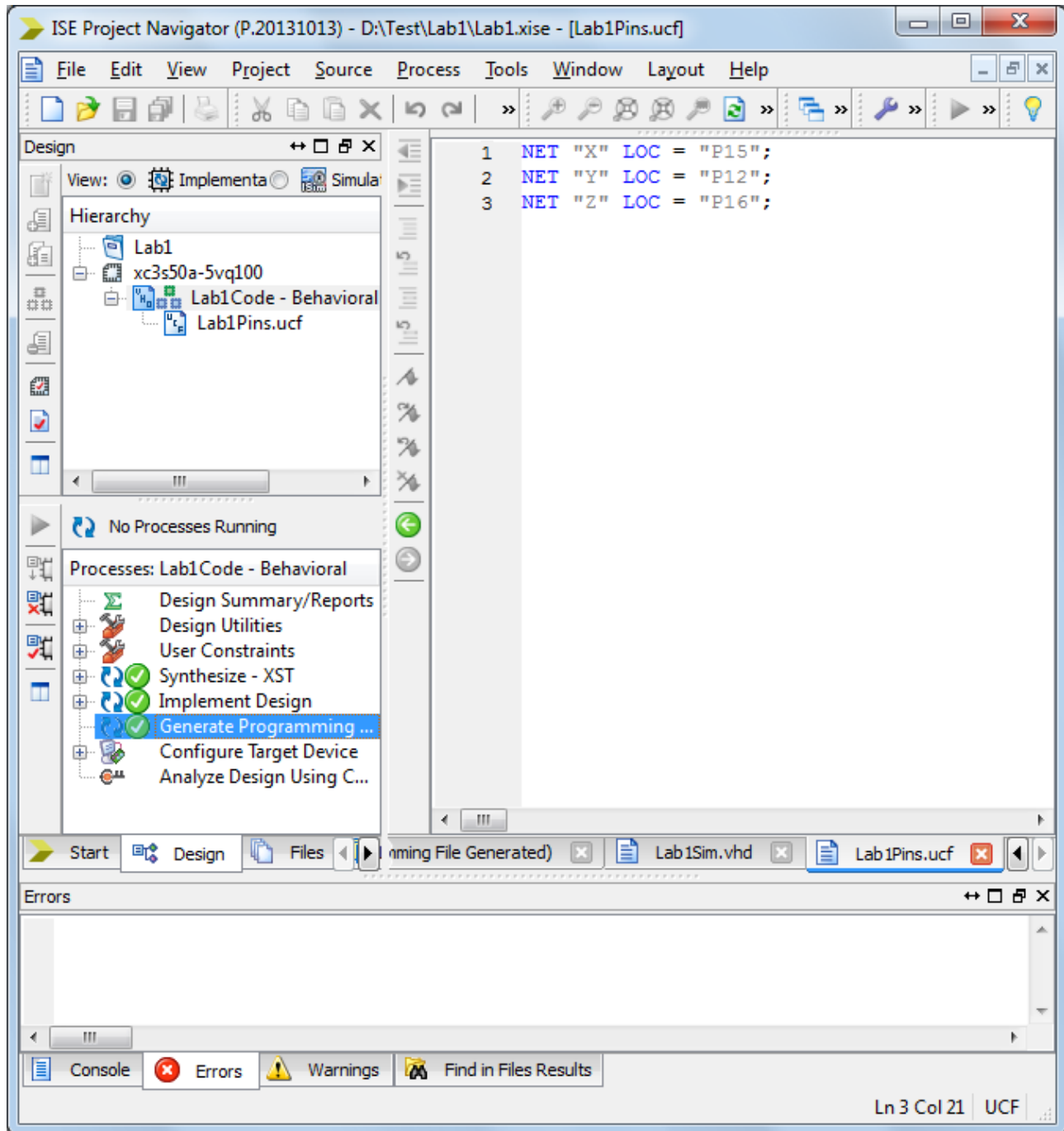
Table 1: Pin-outs of the board.





**Step 17 : Generating Programming File**

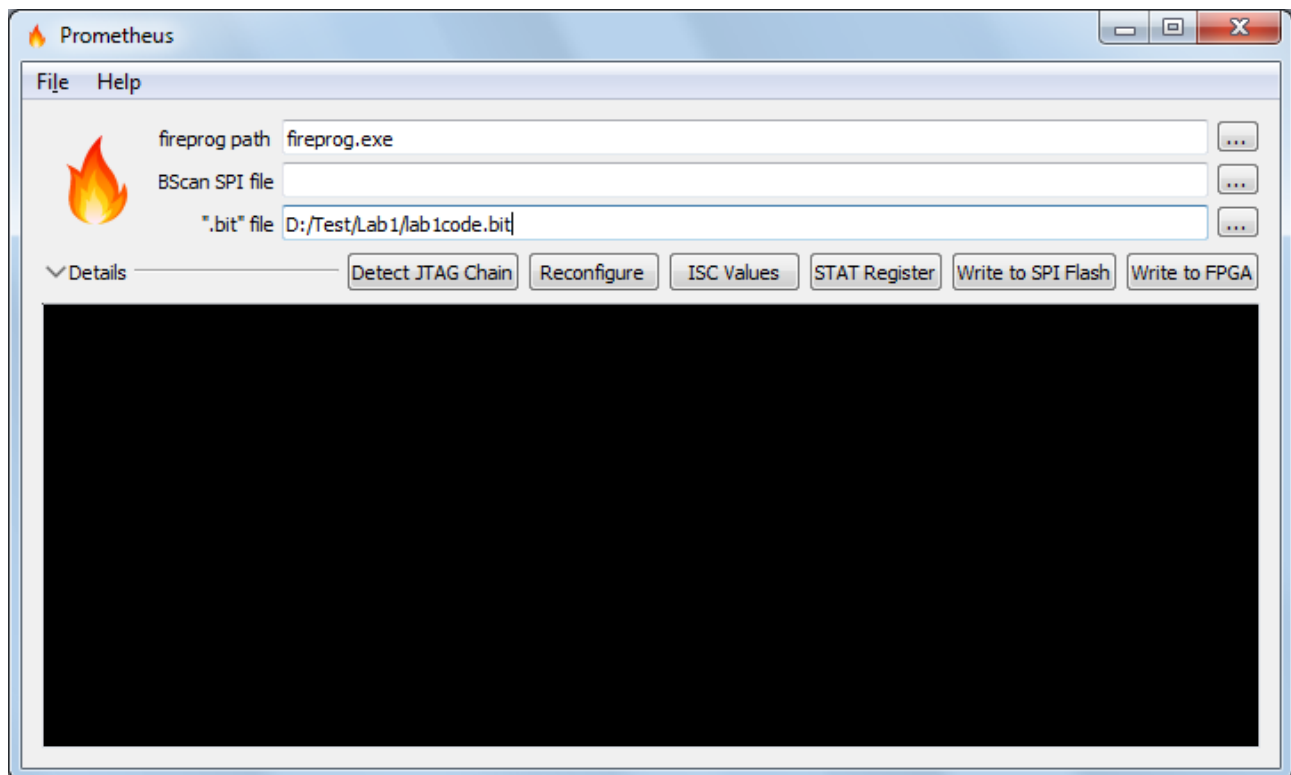
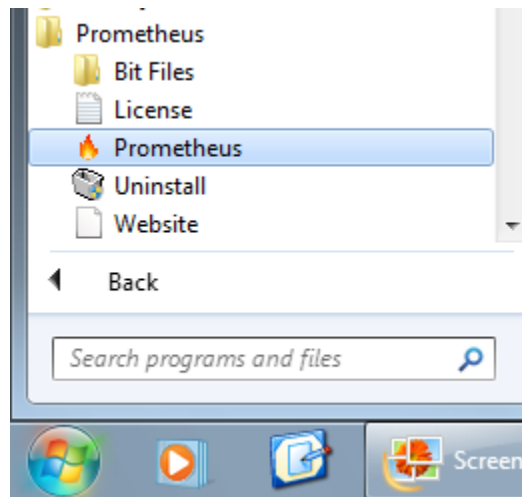
Go to **Processes**. Double click on the **Generate Programming File** option. Wait until the bit file is generated.



## 5.5 Program Hardware

### Step 18 : Launching Prometheus-Fireprog Software

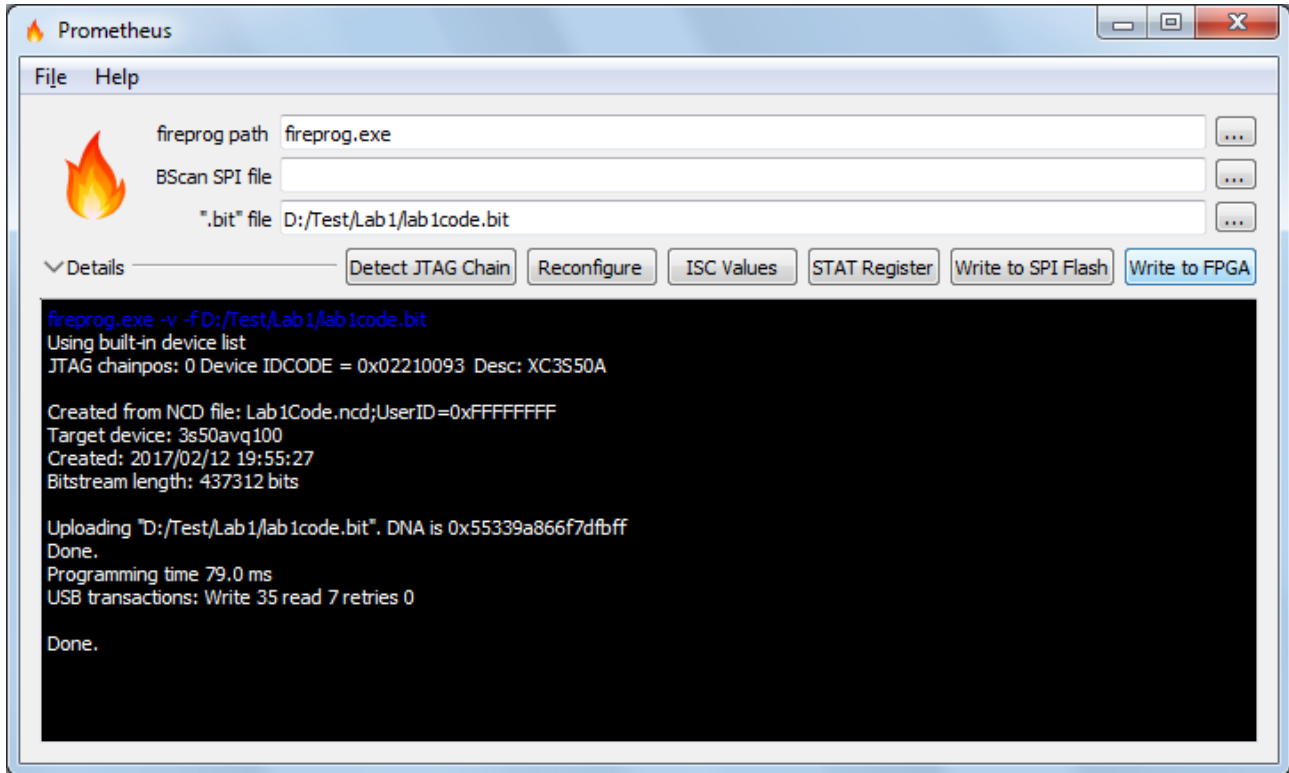
First, connect Prometheus FPGA board to the computer using USB cable. Turn on the board by setting switch (“S11”) on the board to position “on”. Go to **Start Menu**. Launch **Prometheus** program. Choose the “\*.bit” file name with the full path:





**Step 19 : Programming the FPGA Board**

Click on the **Write to FPGA** button:



## 5.6 Test Hardware

### Step 20 : Testing the Design

Now you can test the Boolean function using switches S0 and S1 and observing the output on LED0.

## 6 Implementing XOR Gate

Now you are asked to implement exclusive “OR” gate (XOR) shown in Figure 3 using VHDL.

1. Write VHDL code for XOR gate.
2. Simulate the circuit behavior using Xilinx ISE simulator.
3. Deploy compiled bit code to the Digilent Prometheus Spartan 3A board.
4. Test your deployed circuit by verifying truth table.
5. Demonstrate circuit performance to the teaching assistants.

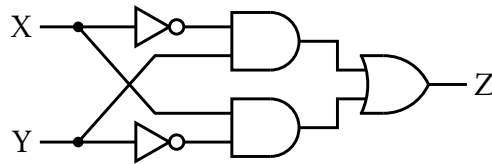


Figure 3: XOR gate structure.