**LAB # 09**

**Verilog HDL**

**Objective**

1. To become familiar with Verilog HDL
2. Simulate & verify the Verilog code on Xilinx ISE

**Requirement**

Xilinx ISE software

**Theory**

**Verilog:**

Verilog is a Hardware Description Language (HDL) used to model digital systems. It provides the designer entry into the world of large, complex digital systems design. The Verilog language provides the digital system designer with a means of describing a digital system at a wide range of levels of abstraction, and, at the same time, provides access to computer-aided design tools to aid in the design process at these levels. It also fulfils the need for verifying the design for functionality and timing constraints like propagation delay, setup and hold times.

The module is the basic building block for modeling hardware with the Verilog HDL. The logic of a module can be described in any one (or a combination) of the following modeling styles:

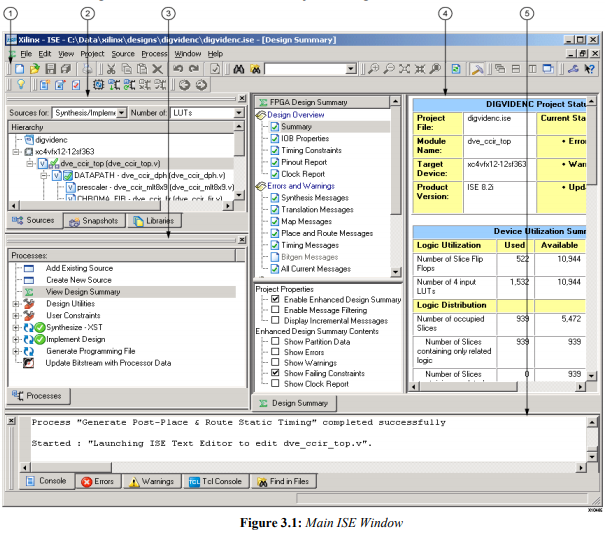
1. **Gate-level modeling** using instantiations of predefined and user-defined primitive gates.
2. **Dataflow modeling** using continuous assignment statements with the keyword assign.
3. **Behavioral modeling** using procedural assignment statements with the keyword always.

**Xilinx ISE:**

The Integrated Software Environment (ISE) is the Xilinx design software suite that allows to take design from design entry, through implementation and verification, to Xilinx device programming.

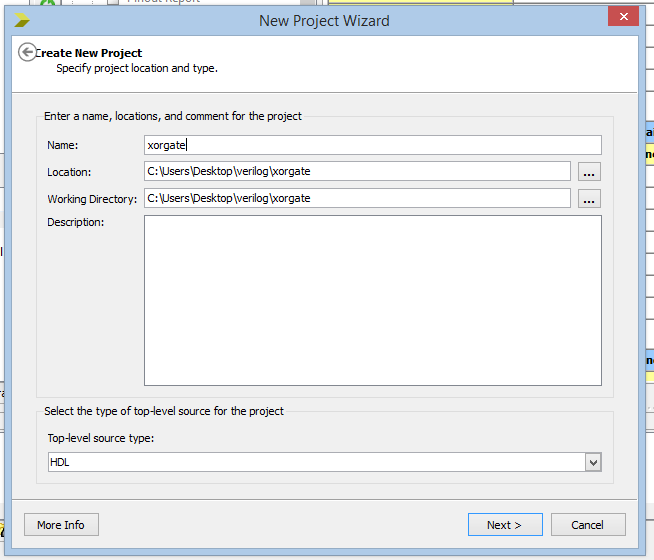
The main window of Xilinx ISE is divided into:

1. Toolbar: Toolbars provide convenient access to frequently used commands.
2. Sources window: The Sources tab in the Sources window shows the source files you create and add to your project.
3. Processes window: The Process tab shows the available processes in a hierarchical view.
4. Workspace: When you open a project source file, open the Language Templates, or run certain processes, such as viewing reports or logs, the corresponding file or view appears in the Workspace.
5. Transcript window: The Console tab of the Transcript window shows output messages from the processes you run.

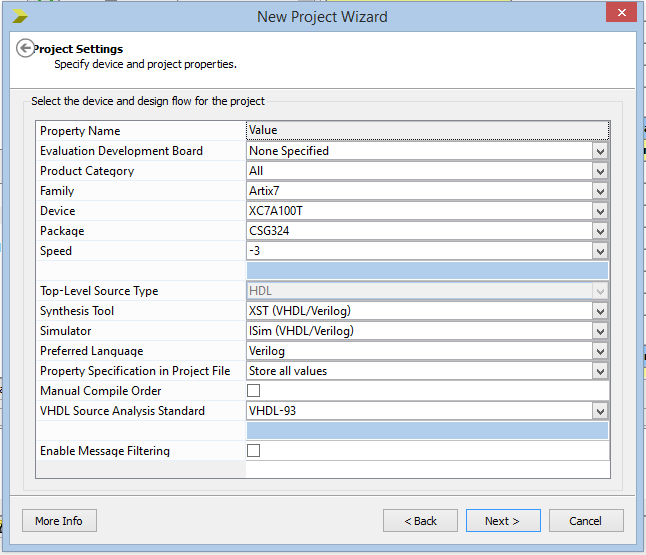


**Procedure**

1. Open the Xilinx ISE software.
2. Select **File > New Project** to launch the New Project Wizard.
3. In the [Create New Project page](file:///C:\Users\HAFSA\Desktop\dld%20new%20labs%20spring%202018\pn_db_npw_create_new_project.htm), set the name, location, and **HDL** as Top-level source type for the project, and click **Next**.



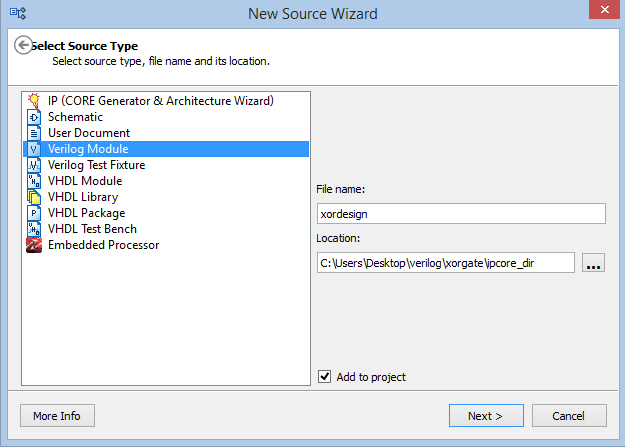
1. In the [Project Settings page](file:///C:\Users\HAFSA\Desktop\dld%20new%20labs%20spring%202018\pn_db_npw_device_properties.htm), set the device and project properties, select Verilog as preferred language & click **Next**.



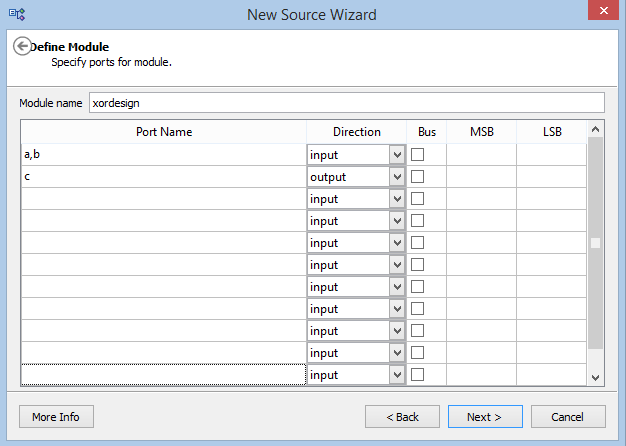
1. In the [Project Summary page](file:///C:\Users\HAFSA\Desktop\dld%20new%20labs%20spring%202018\pn_db_npw_project_summary.htm), review the information, and click **Finish** to create the project.

To create the Verilog source file for the project, do the following steps:

1. Click on the New Source icon
2. Select Verilog Module as the source type in the New Source dialog box. Type in the file name. Verify that the Add to Project checkbox is selected. Click Next.



1. Use the Define Module page to define the module for the HDL file you create i.e. declare the ports for module by filling in the port information

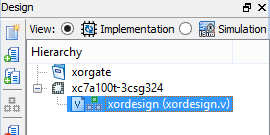


To check the syntax of the design to find errors after the source files are completed.

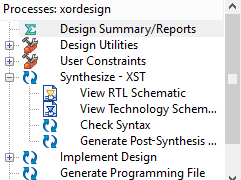
1. Check Implementation from here



1. Select the design source in the Sources window to display the related processes in the Processes window.



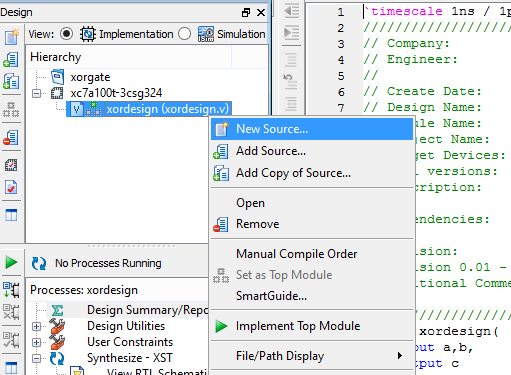
1. Click the “+” next to the Synthesize-XST process to expand the process group & double-click the Check Syntax process.



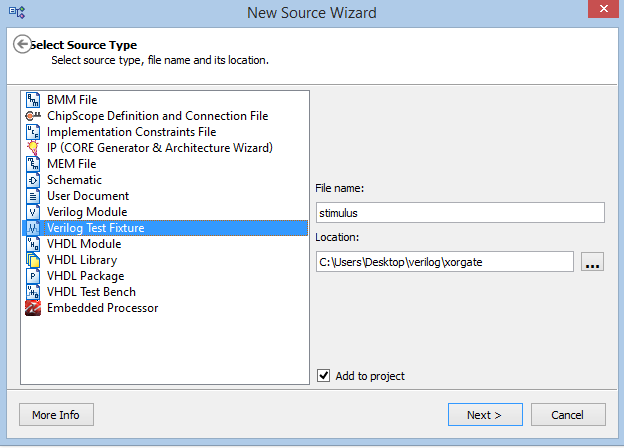
1. Check for errors in the Console tab of the Transcript window. If continue without valid syntax, design cannot be simulated or synthesize.

**Simulation**

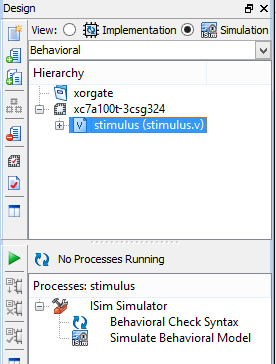
1. Create a test bench containing input stimulus to verify the functionality of the module.
2. Select the HDL file in the Sources window.
3. Create a new test bench by selecting Project → New Source



1. In the New Source Wizard, select Verilog Test Fixture & type the File name.



1. Write test bench file & save it.
2. Check simulation.



1. Select Simulation and test bench file as shown above in the fig. Click the “+” to expand the ISE Simulator & click the Simulate Behavioral Model. The ISE Simulator runs the simulation.

**4-to-1 Multiplexer code using Gate Level Modeling**

*// Module 4-to-1 multiplexer using gate level modeling*

*module mux4\_to\_1 (output out, input i0, i1, i2, i3, input s1, s0);*

*// Internal wire declarations*

*wire s1n, s0n; wire y0, y1, y2, y3;*

*// Gate instantiations*

*// Create s1n and s0n signals*

*not (s1n, s1);*

*not (s0n, s0);*

*// 3-input and gates instantiated*

*and (y0, i0, s1n, s0n);*

*and (y1, i1, s1n, s0);*

*and (y2, i2, s1, s0n);*

*and (y3, i3, s1, s0);*

*// 4-input or gate instantiated*

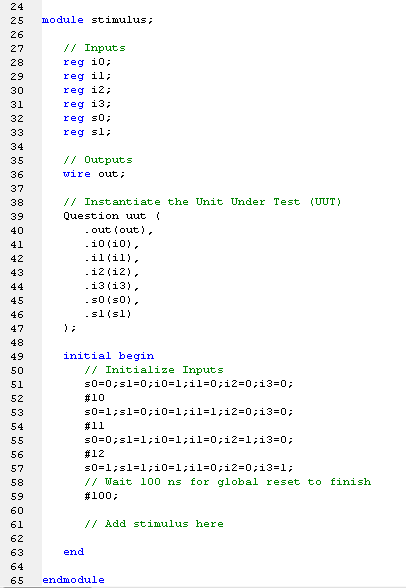
*or (out, y0, y1, y2, y3);*

*endmodule*

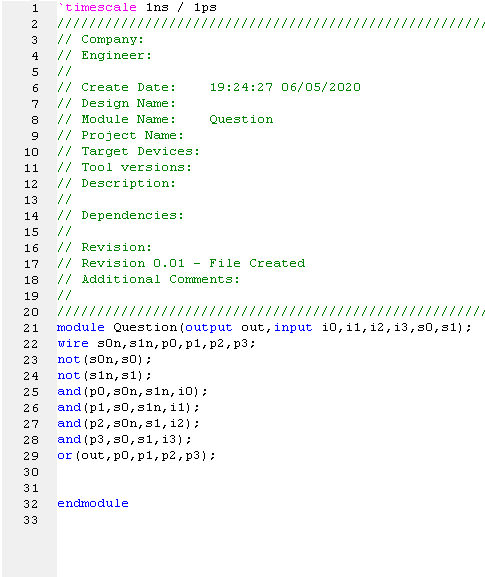
**Task**

1. Write the above code in Xilinx ISE Design Suite.
2. Check syntax of the design code.
3. Write its test bench code.
4. Show its simulation.
5. Attach snapshots of your design code, test bench code & simulation here.

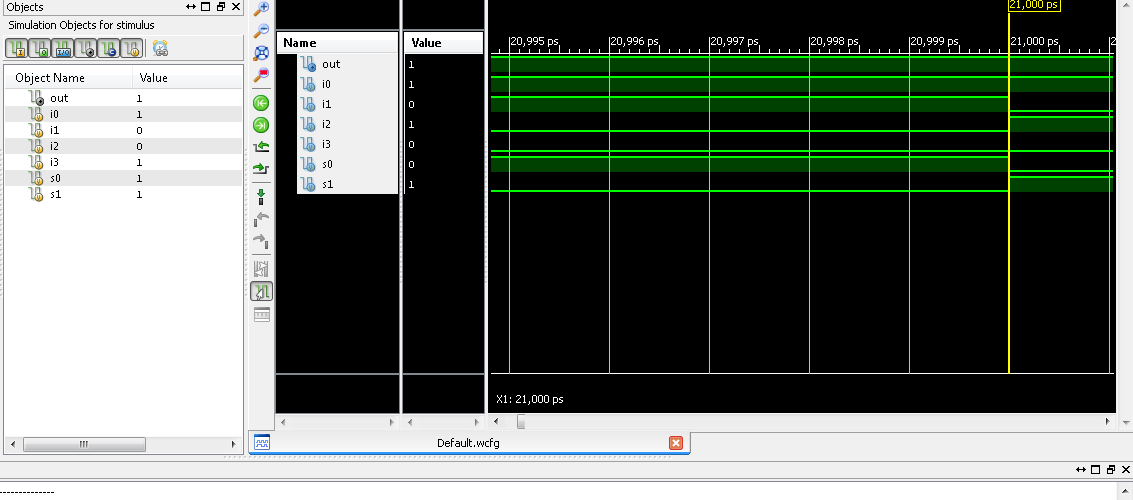
Test Bench code



Syntax Check



Simulation



**Question**

* Write design code for 2:1 Mux (in your lab manual)
* Implement 2:1 Mux on Xilinx ISE Design Suite (attach the screenshots of its design code, test bench code & simulation)

module Assignment(input s0,D0,D1,output out);

wire s0n,o1,o2;

not(s0n,s0);

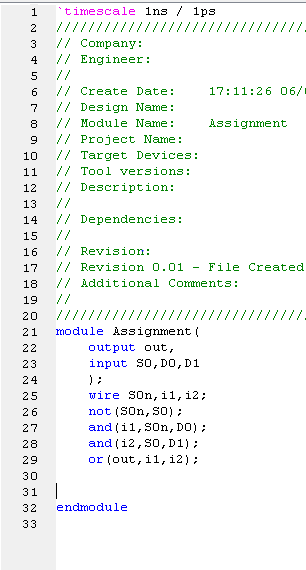
and(o1,D0,s0n);

and(o2,D1,s0);

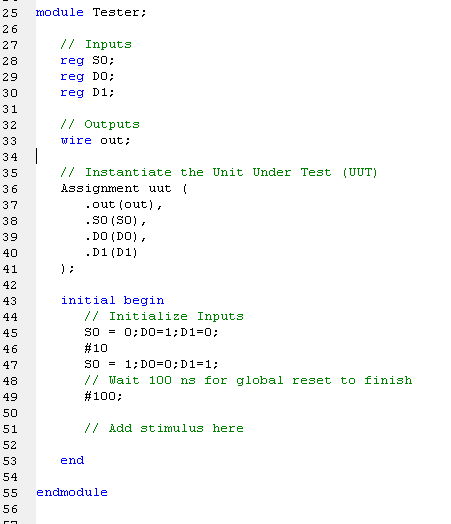
or(out,o1,o2);

endmodule

Check Syntax



Bench Test



Simulation

