Overview of Xilinx ISE tools

(Creating source files in VHDL and generating test-bench waveform)

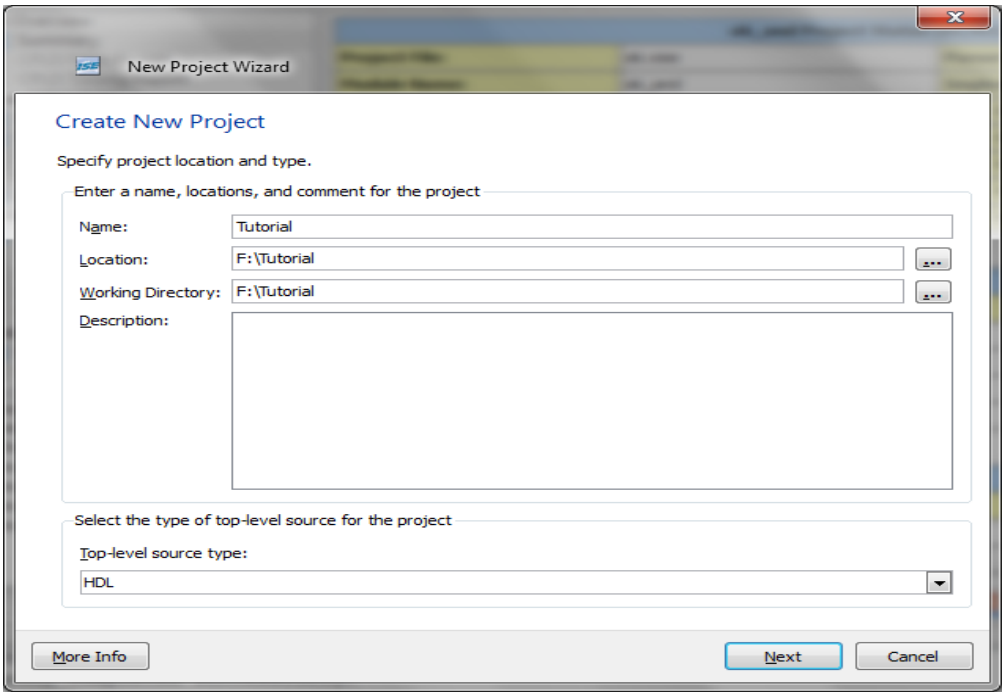
Starting the Xilinx ISE Software:

Start Xilinx ISE from the Start menu by selecting
Start →Programs → Xilinx ISE.
Note Your start-up path is set during the installation process and may differ from the one above.

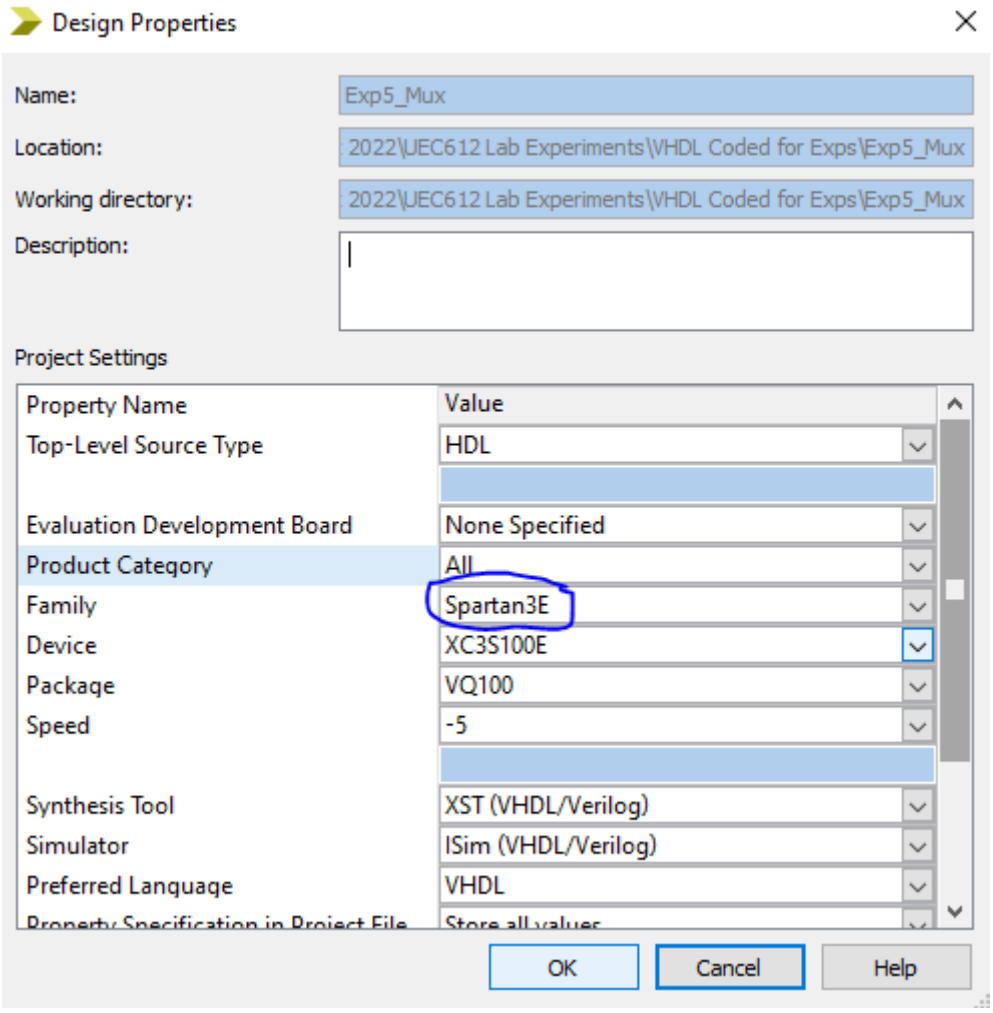
Creating a New Project

A project in ISE is a collection of all files necessary to create and download a design to the selected device. To create a new project for this tutorial:

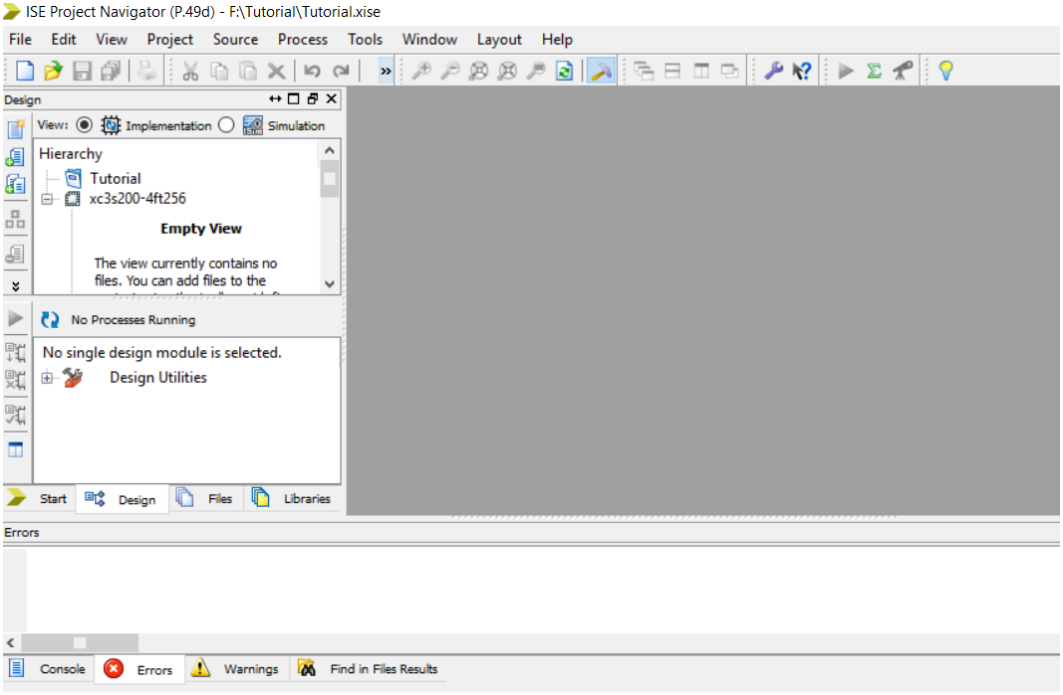
1. Select **File →New Project**.
2. In the New Project dialog box, type the desired location in the Project Location field, or browse to the directory under which you want to create your new project directory using the browse button next to the project location field.
3. Enter 'Tutorial' in the Project Name field. When you enter 'Tutorial' in the Project Name field, a Tutorial subdirectory is automatically created in the directory path in the project location field. For example, for the directory path F:\, **entering the Project Name 'Tutorial'** modifies the path as F:\Tutorial.
4. Use the pull-down arrow to select the top level source type HDL.



Then click next and select Value for each Property Name as shown in the next figure, Click in the field to access the pull down list. Change the values as follows:



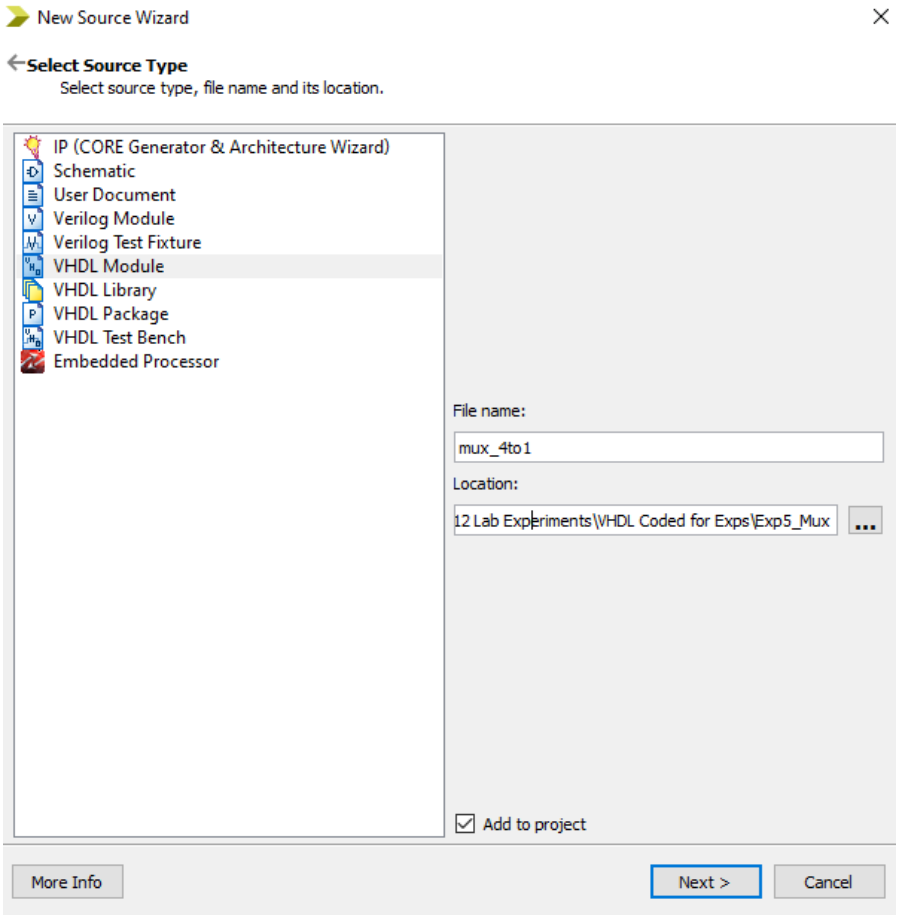
5. After that click next then click finish. Xilinx ISE creates and displays the new project in the Sources in Project window as below



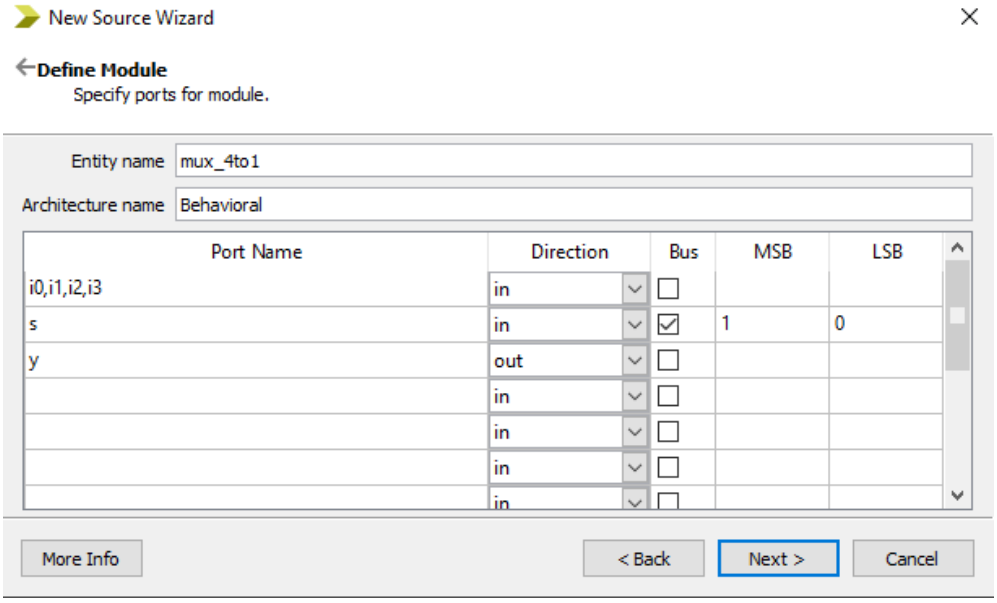
Creating a device Module:

Create a ***VHDL module*** for a digital device to create a device module:

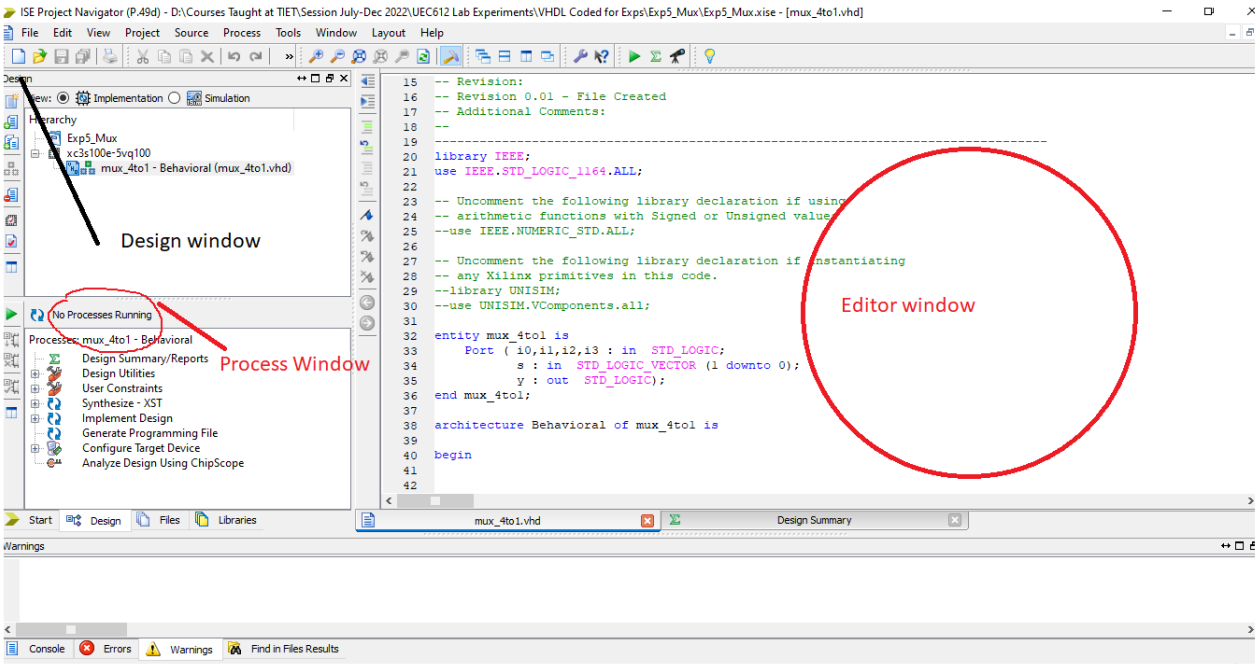
1. Right click the project name (here its name is **tutorial**) → **New Source**.
2. Select **VHDL Module** as the source type.
3. Type in the file name 'mux_4to1' or as you wish. **Click next**



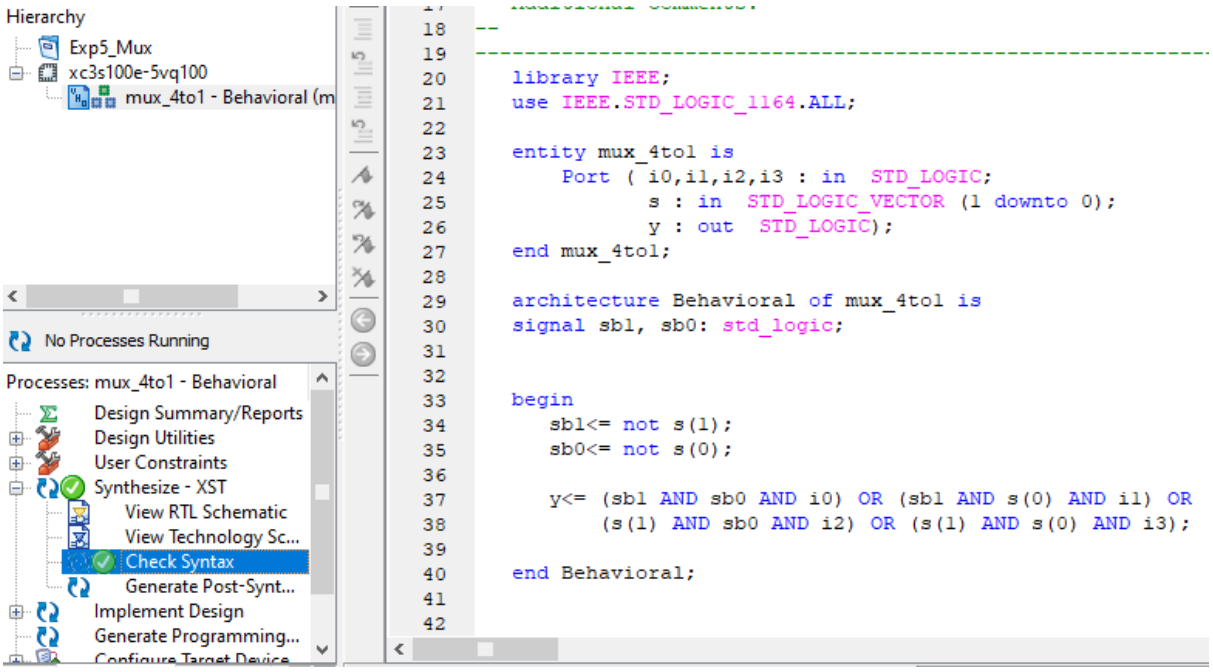
4. Enter inputs i0, i1, i2, i3, s and y for output.
5. Click **Next**.
6. Click **Finish** to complete the new source file template.



- You will get HDL Editor window, contains the library declaration and use statements along with the empty entity and architecture pair for the device you have just created. HDL Editor is a text editor designed for editing HDL source files.



- Now, write your program in HDL editor window and save it then check syntax error.



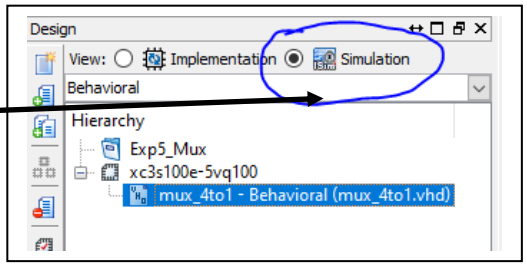
- To check your syntax errors, go through the following steps.
  Processes window→ Synthesis XST →check syntax
- After check syntax then synthesize the program.
  Processes window → Synthesis XST

If there is no error, then write the testbench for the simulation of your VHDL design.

Simulating the Behavioral Model

In this section, you will create a test bench waveform that defines the desired functionality for the device module. This test bench waveform is then used in conjunction with **ISim** to verify that the device design meets both behavioral and timing design requirements.

Creating a Test bench Waveform Source:



After going in Simulation View,

1. Right click the project name → **New Source**.
2. Select **VHDL test bench** as the source type.
3. Type in the file name 'mux_tb' or as you wish.
4. **Click next**
5. Then click next then click finish.

Note In other projects, you can associate your test bench waveform with other sources.

After clicking the finish button software displayed the HDL Editor window, contains the time scale declaration and the module declaration without port list. HDL Editor is a text editor designed for editing HDL source files.

Then write your text bench program/ put your inputs in HDL editor window and save it then check syntax error.

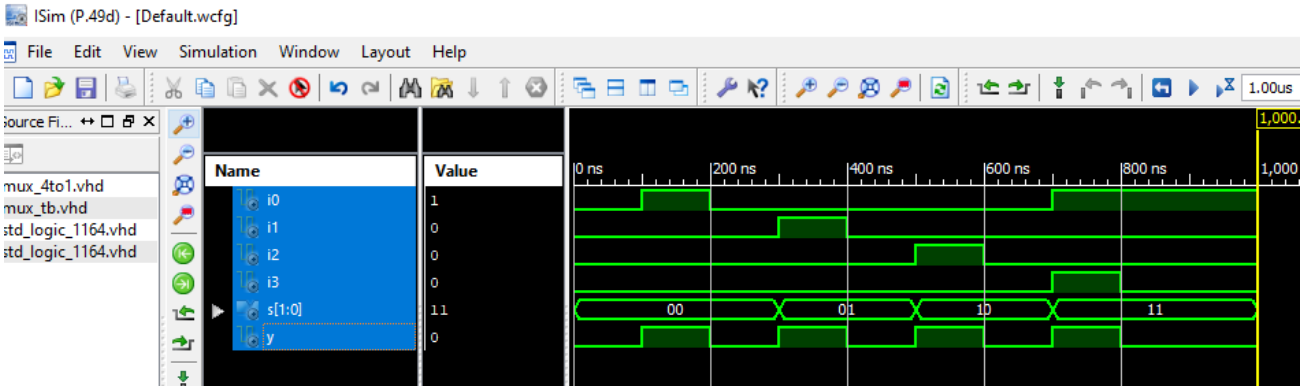
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY mux_tb IS
END mux_tb;

ARCHITECTURE behavior OF mux_tb IS

    -- Component Declaration for the Unit Under Test (UUT)

   COMPONENT mux_4to1 is
       Port ( i0,i1,i2,i3 : in  STD_LOGIC;
              s : in  STD_LOGIC_VECTOR (1 downto 0);
              y : out  STD_LOGIC);
   end COMPONENT;


   --Inputs
   signal i0 : std_logic := '0';
   signal i1 : std_logic := '0';
   signal i2 : std_logic := '0';
   signal i3 : std_logic := '0';
   signal s : std_logic_vector(1 downto 0) := (others => '0');

   --Outputs
   signal y : std_logic;
```

```
BEGIN
   -- Instantiate the Unit Under Test (UUT)
   uut: mux_4to1 PORT MAP (i0,i1,i2,i3,s,y);

   -- Stimulus process
   stim_proc: process
   begin
      -- hold reset state for 100 ns.
      wait for 100 ns;

      -- insert stimulus here to check the fucntion of multiplexer.
       s<="00";   i3<='0'; i2<='0'; i1<='0'; i0<='1'; wait for 100 ns;
       s<="00";   i3<='0'; i2<='0'; i1<='0'; i0<='0'; wait for 100 ns;

       s<="01";   i3<='0'; i2<='0'; i1<='1'; i0<='0'; wait for 100 ns;
       s<="01";   i3<='0'; i2<='0'; i1<='0'; i0<='0'; wait for 100 ns;

       s<="10";   i3<='0'; i2<='1'; i1<='0'; i0<='0'; wait for 100 ns;
       s<="10";   i3<='0'; i2<='0'; i1<='0'; i0<='0'; wait for 100 ns;

       s<="11";   i3<='1'; i2<='0'; i1<='0'; i0<='1'; wait for 100 ns;
       s<="11";   i3<='0'; i2<='0'; i1<='0'; i0<='1'; wait for 100 ns;

     wait;
   end process;
END;
```

To check your syntax errors go through the following commands.
Processes window→ ISim Simulator→ Behavioral Check Syntax

After check syntax then Simulate Behavioral Model of the program.
Processes window→ ISim Simulator → Simulate Behavioral Model

Your test bench waveform should look like as below



In the Waveform window, click **Zoom** → **Zoom Full** or click the Zoom Full icon in the toolbar.
The output waveform should look like the one in figure. Then save your test bench waveform then exit HDL test bench window.