PDE3411 Laboratory Worksheet 1

7/10/2019

Aim

The purpose of this lab is to introduce you to the basic FPGA design and programming tools. For the purposes of this lab you will first study the useful Vivado Design Suite and then implement a 2:1 Mux circuit using FPGA design flow.

Objectives

- Verify that the Xilinx tools are up and running
- Introduce you to the Xilinx Vivado Design Suite
- Become familiar with VHDL coding and use of the Vivado Simulator
- Be able to synthesize and implement designs to FPGAs using Vivado Design Suite.

Equipment & Tools

Xilinx Vivado Suite

You can visit

http://www.xilinx.com/products/design-tools/vivado.html

and choose the version you want. You can download the Vivado Design Suite Webpack for free, but if you have trouble installing it, let us know ASAP.

Vivado Simulator

Introduction

This is a step-by-step tutorial for building a 2:1 Mux in Xilinx Vivado, a Design Suite software that provides designers with the ability to code designs in a hardware description language such as VHDL or Verilog. The Vivado Design Suite also provides the ability to apply FPGA pin and timing constraints, analyse for errors and violations, and synthesize to generate configuration bit file formats for FPGAs.

By the end of this tutorial, you should be able to:

- Create a new design by VHDL coding.
- Verify the function of a design by behavioural simulation.
- Map a design to an FPGA device through placement and routing procedures.
- Estimate the performance of the design by timing analysis.
- Use the 2:1 MUX in this tutorial to implement and simulate a MUX in FPGA.

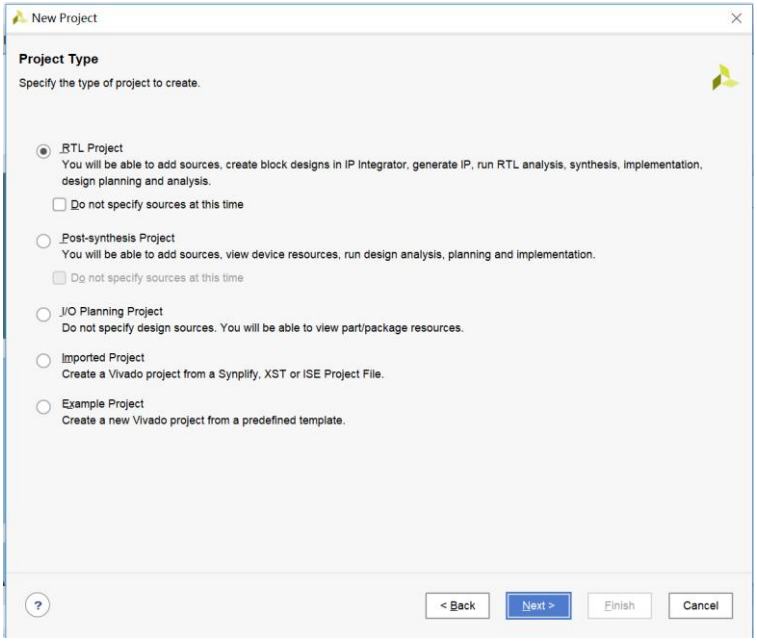
Step-by-step 2:1 MUX Design

Tasks:

(1) Create a project file in Xilinx Vivado Project Navigator.

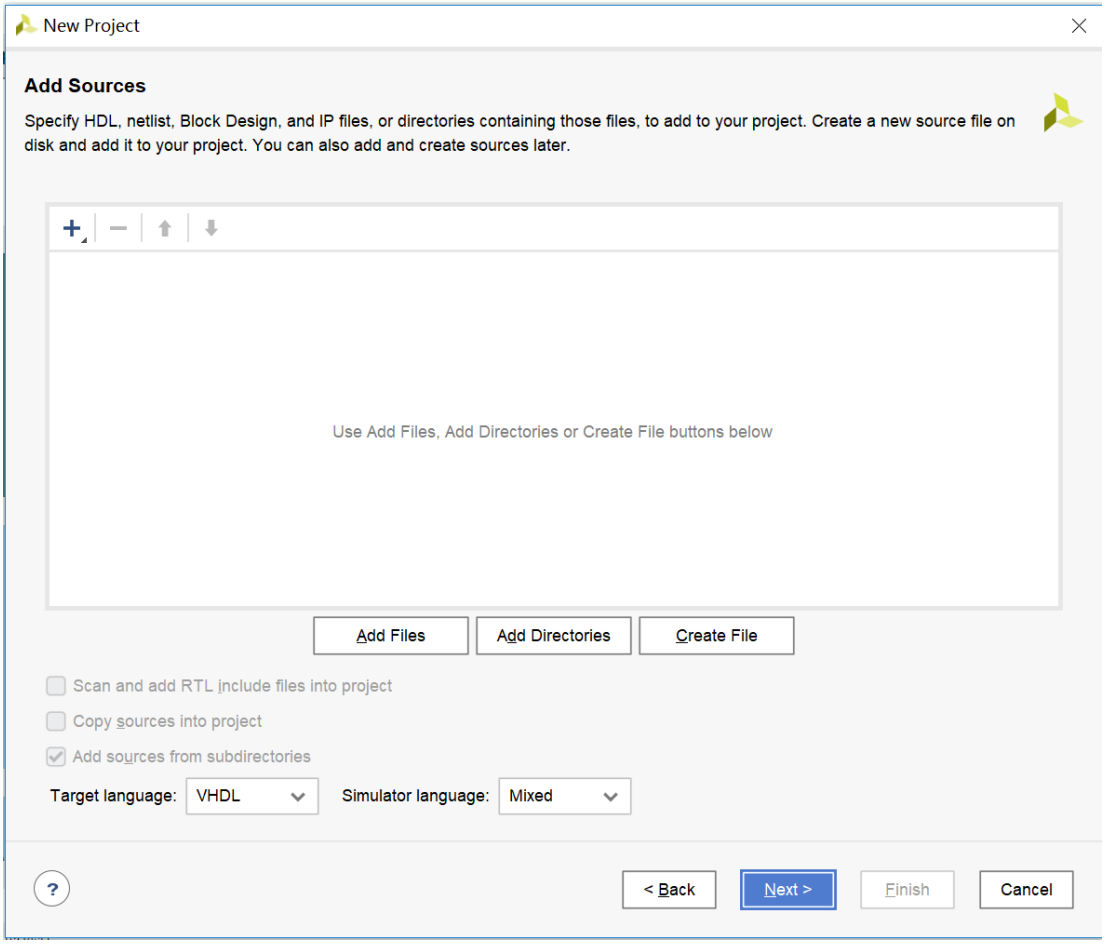
- To launch the Project Navigator run **Start → Programs → Xilinx Design Tools → Vivado 2019.1.** Or, click the following icon on desktop.

Create a new project by clicking **Create Project** in Quick Start pane. You will get a **Create a New Project** wizard. Click Next, then give the project name and location of your choice, click **Next**. For the project type, choose the default **RTL project** then click **Next** (see figure below).

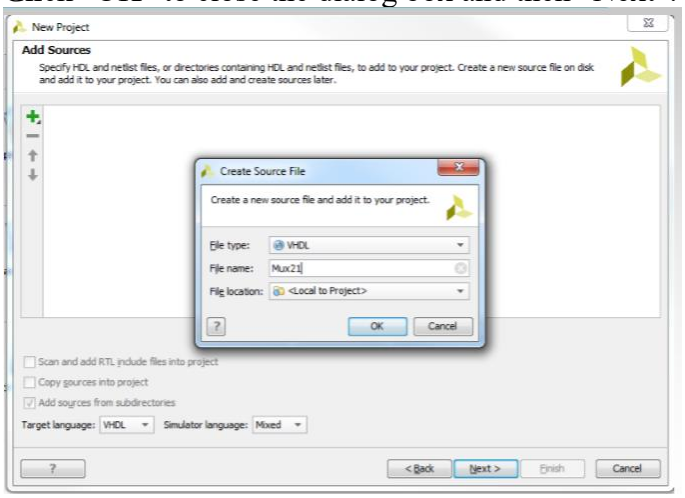


- In the Add Sources window, set the “Target Language” to VHDL and the “Simulator language” to Mixed.

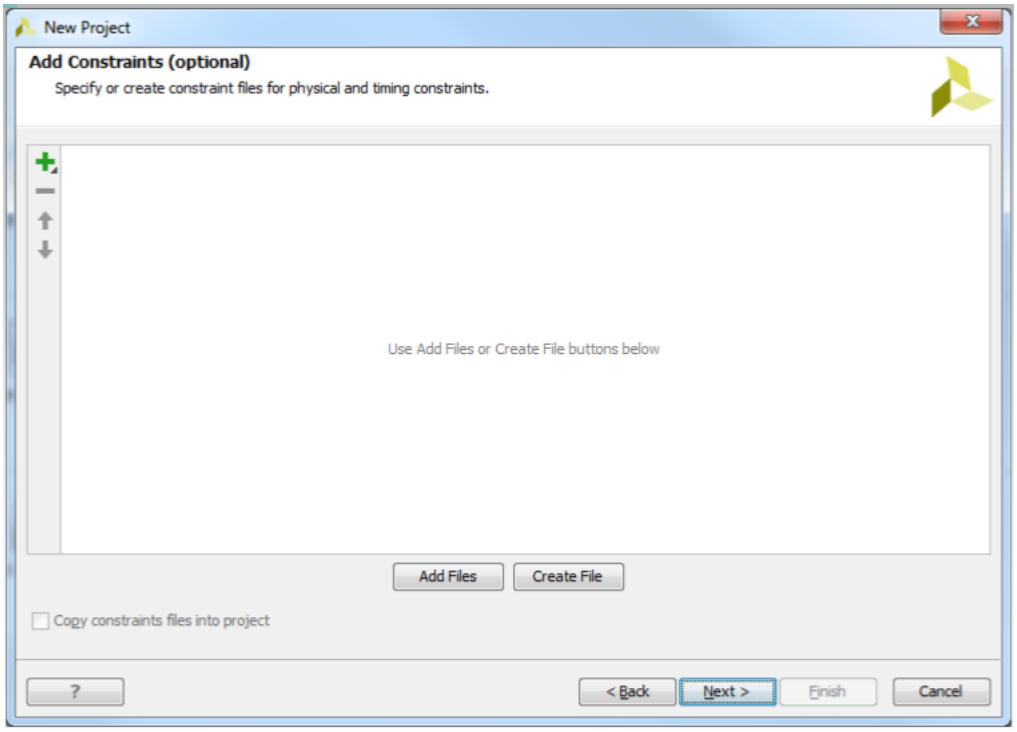
- After setting the language options, click “Create File”



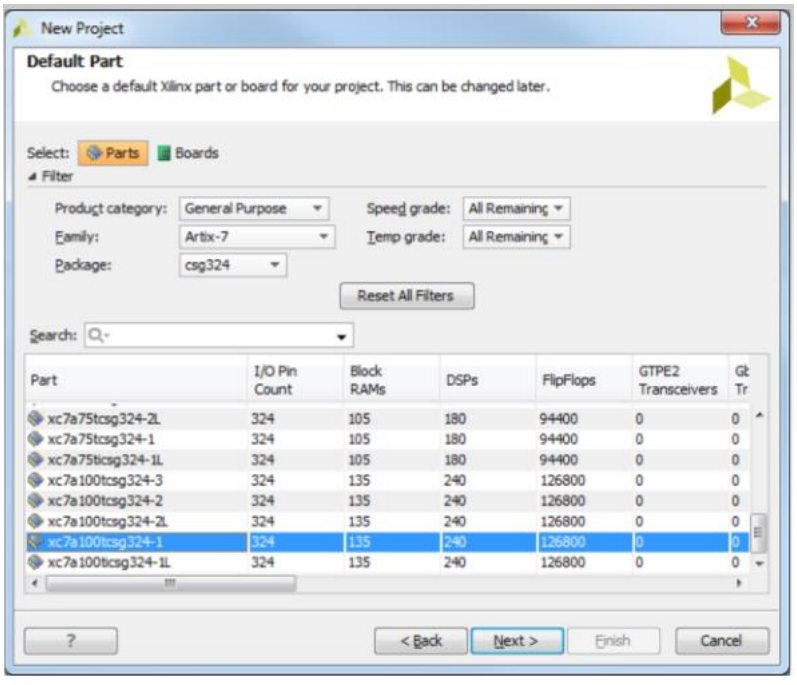
- In the Create Source File dialog box, make sure the “File type” is set to VHDL and enter the name of your file.
  - Again, it is always a good idea to use a descriptive name, in this case Mux21because this is a 2-1 multiplexor.
- Click “OK” to close the dialog box and then “Next”.



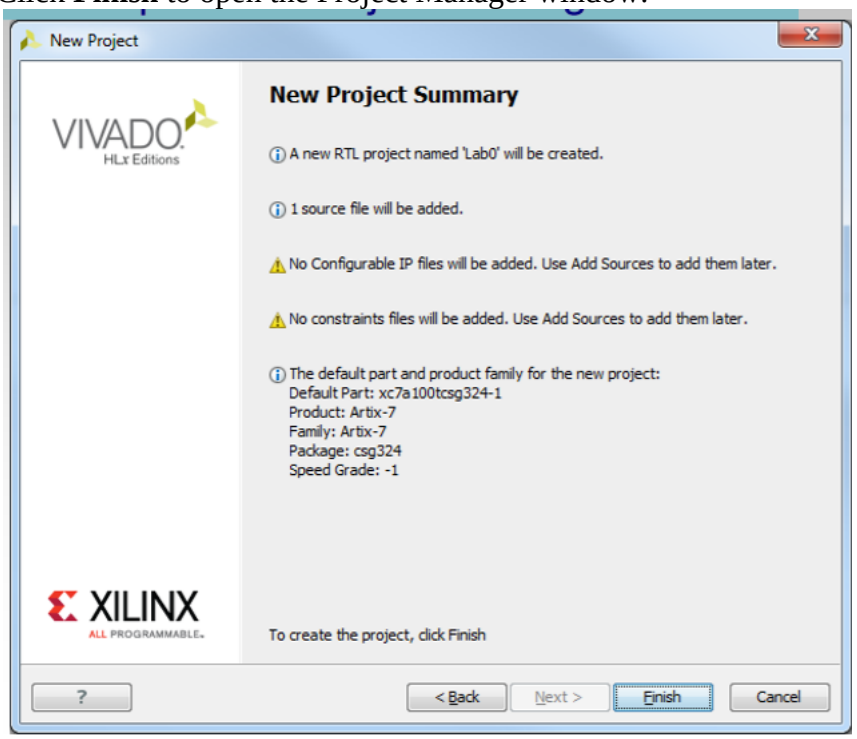
- The created file will appear in menu, click **Next.**
- We do not have any constraints yet, but in future projects, you may import and edit existing constraints files to save time. So click **Next** for now.



- Here we need to select our device. You can use the search function, filters, or just scroll until you find our device: XC7A100tcsg324-1.
  - This FPGA is from the Xilinx Artix-7 family (XC7A100T).
  - The device is contained in a 324-pin csg324 package.
  - The speed grade of the part is "-1".
- Click **Next**

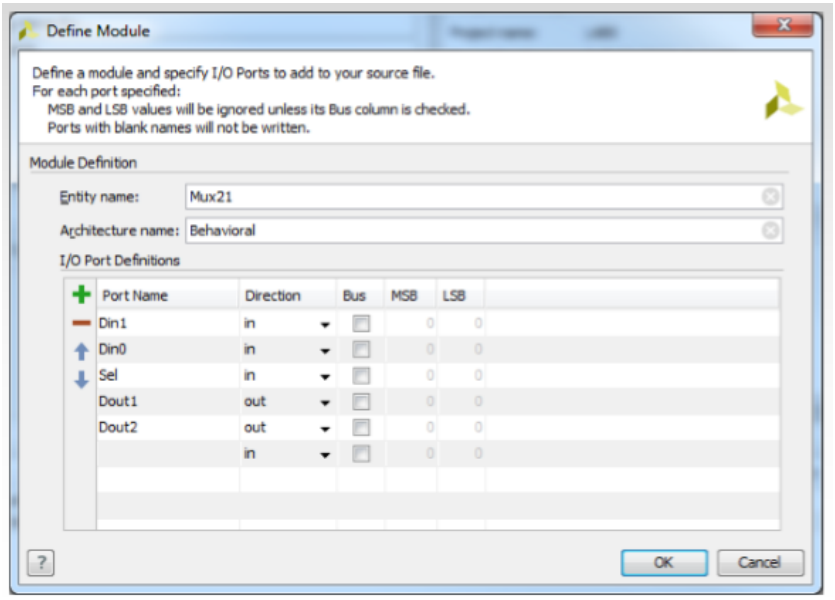


- The New Project Summary window lists information selected in the previous screens.
  • If necessary, use the **Back** button to return to previous screens to make changes/corrections.
- Click **Finish** to open the Project Manager window.

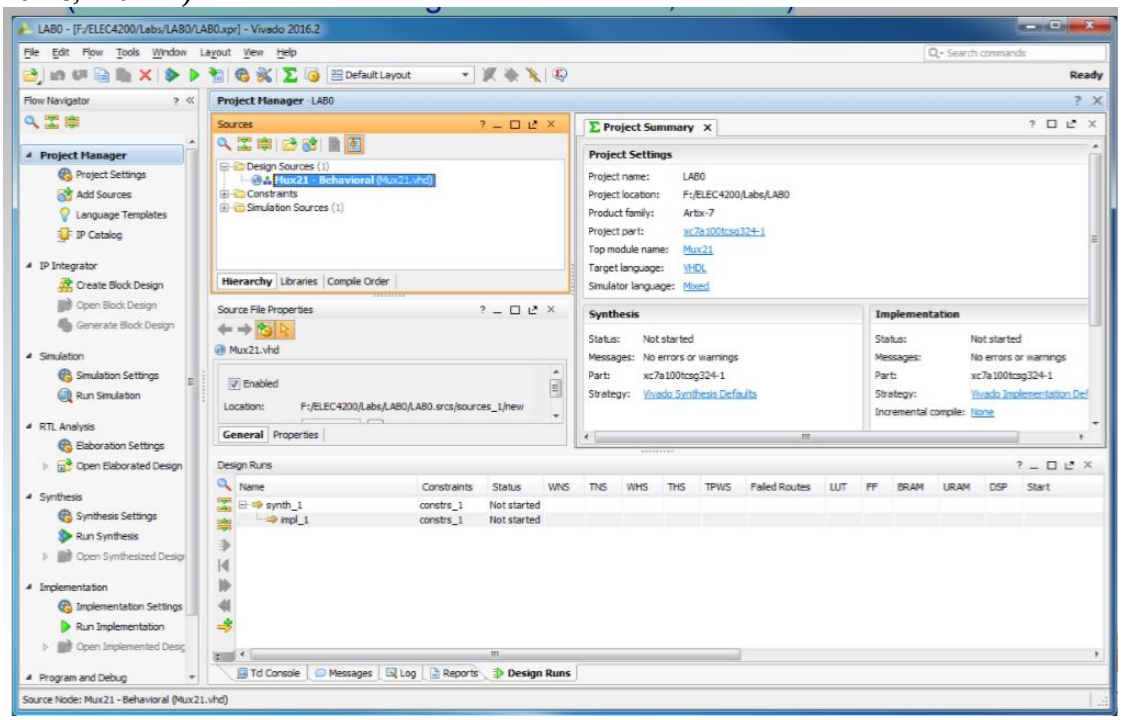


(2) Create the VHDL model

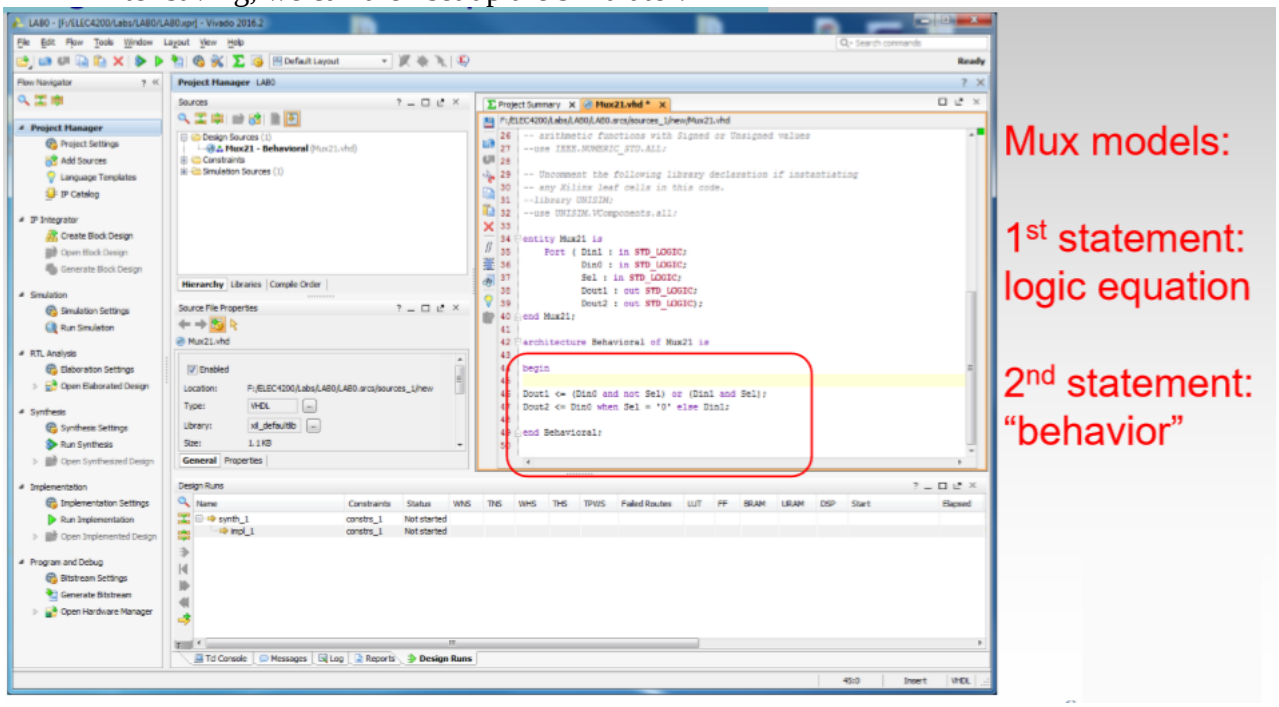
- Since we chose to create a new VHDL file, Vivado will automatically launch a wizard to assist in creating the entity and architecture structures that comprise a VHDL model. This can all be done by hand (and you can edit all of this later), but there is no reason not to take advantage of the wizard utility.
- In **Module Definition** window, leave the Entity and Architecture names as their defaults.
- Create three inputs (Direction "in"): Din0, Din1, Sel
- Create two outputs (Direction "out"): Dout1, Dout2
- Click **OK**



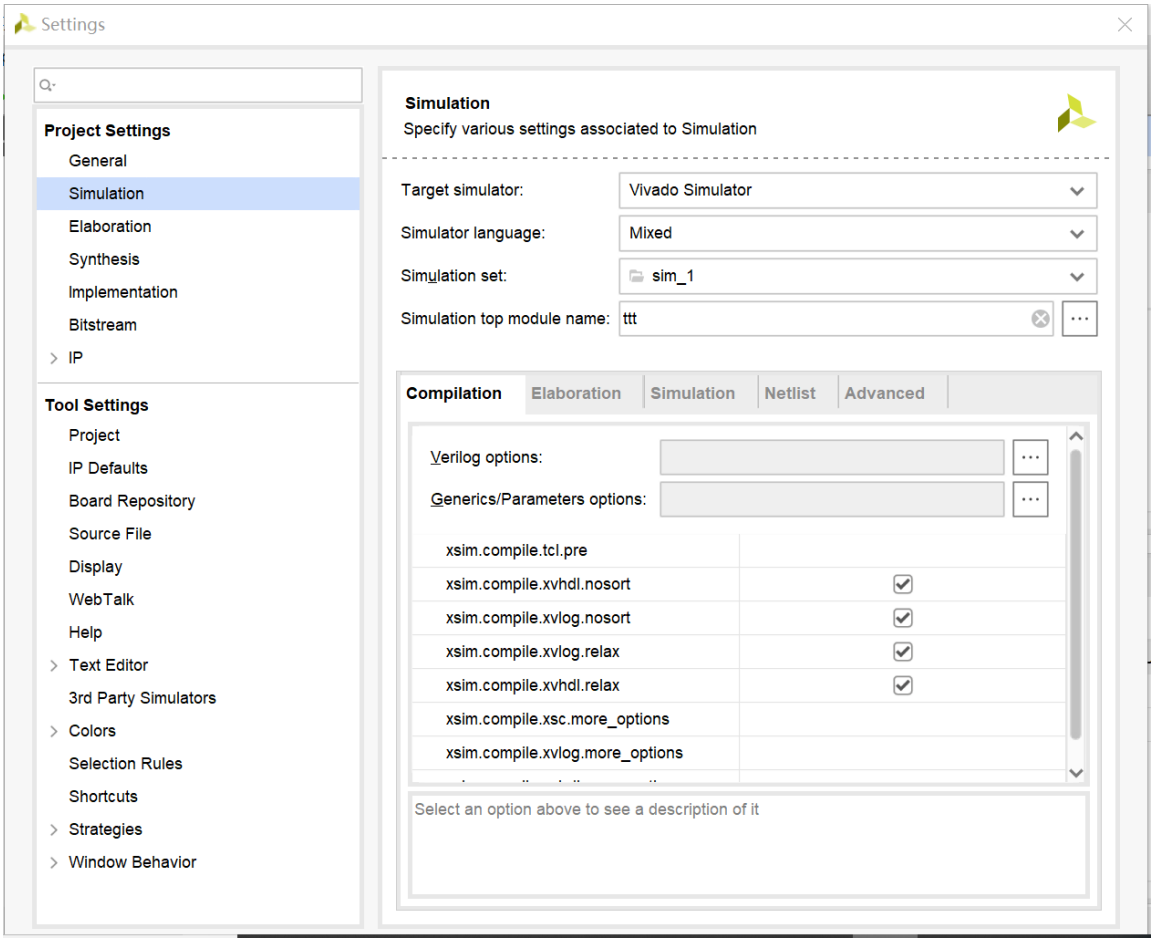
- Open your new VHDL file from the Sources window (double click on the Design Source name, Mux21)



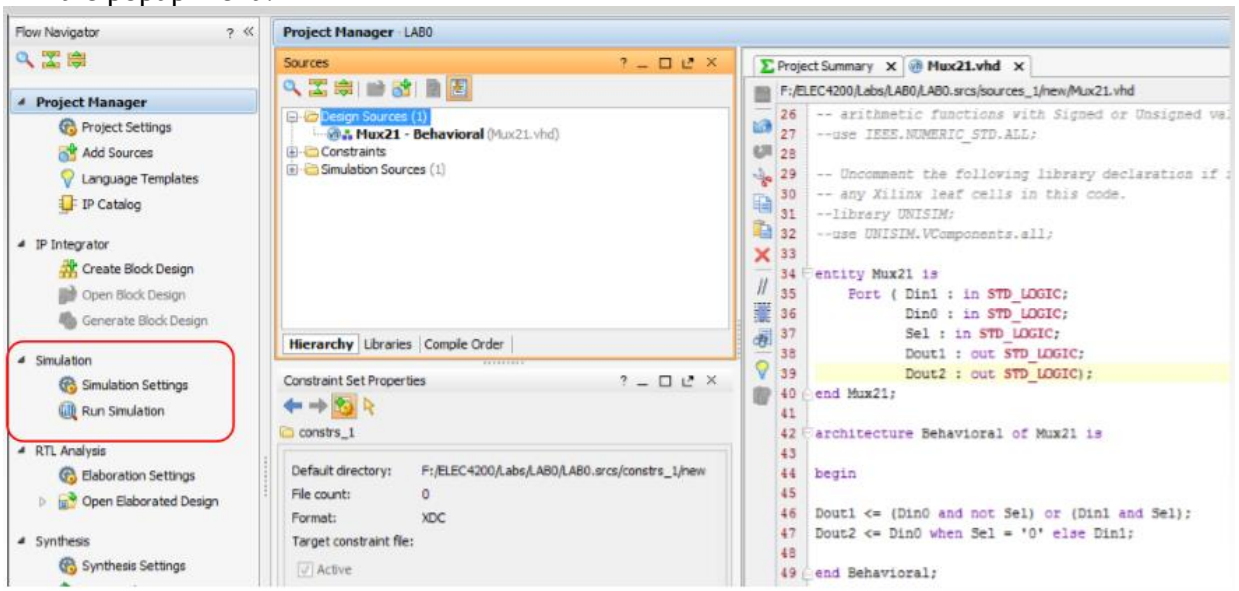
- Scroll down and add the following two lines of code between the Begin and End statements of the architecture section of the model.
  - Dout1<=(Din0 and not Sel) or (Din1 and Sel);
  - Dout2<=Din0 when Sel = '0' else Din1;
- After saving, we can then set up the simulator.



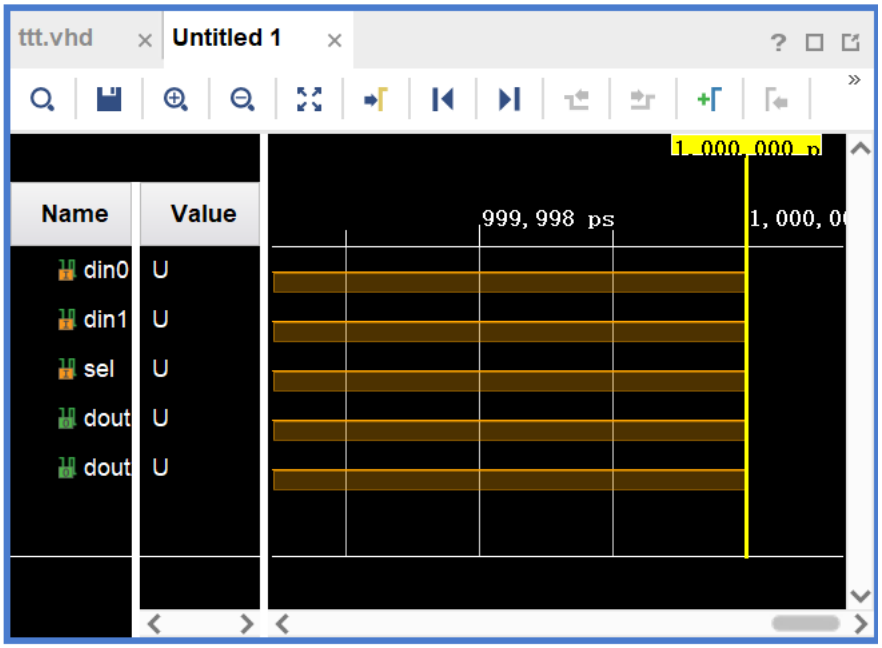
- Double click on **Settings** in **Project Manager**, in the pop out window, click on **Simulation** in **Project Setting**. In the **Target Simulator** drop-down menu, you will have choice of different simulators. We will use the default Vivado simulator.



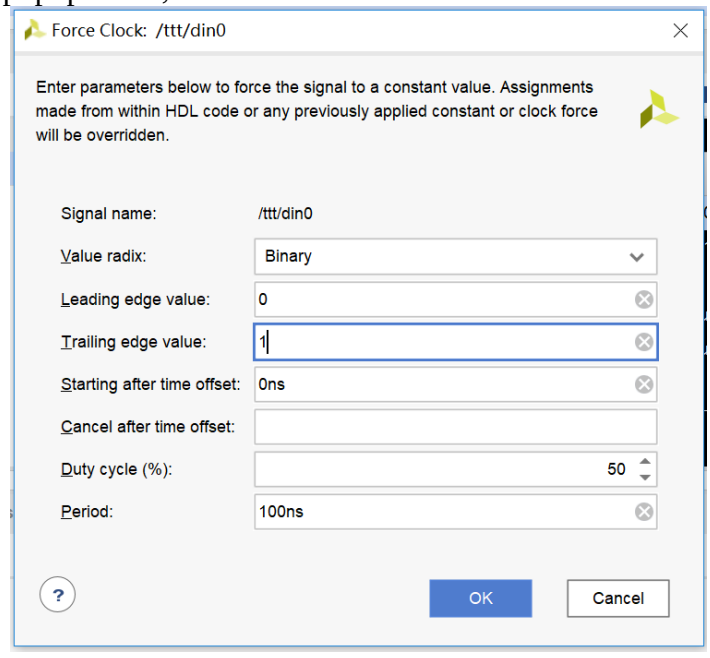
- Click on “Run Simulation" in the Flow Navigator, and then Run “Behavioral Simulation” in the popup menu.



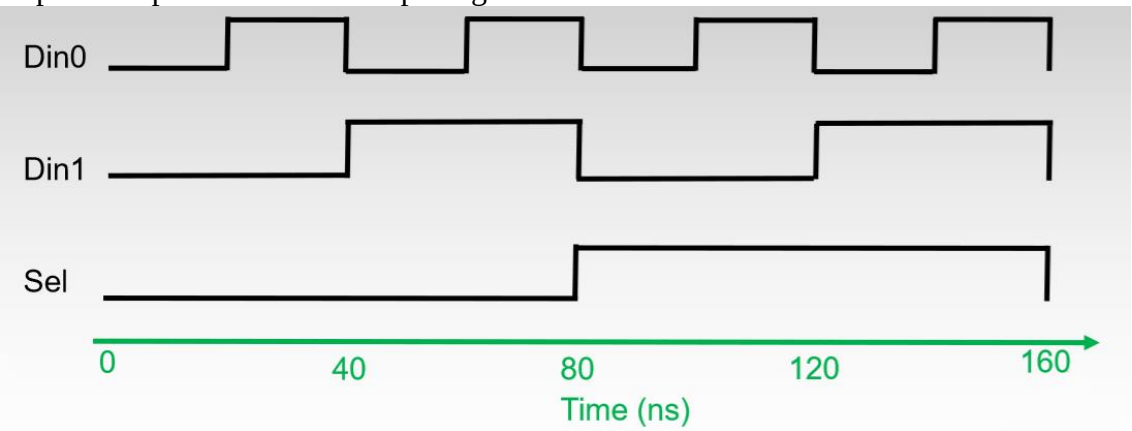
- •Note that we have not yet provided any signal inputs, so this "simulation" did not provide any useful information.
- When the simulator is finished launching you should see the following window. If you do not, call tutors to set up the proper view windows.
- We will set up the desired simulation in the next steps.



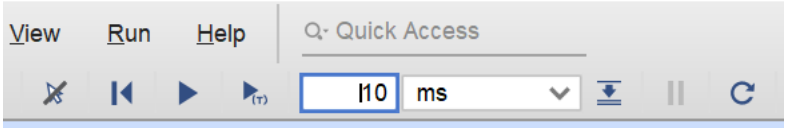
- Design verification requires that you stimulate the inputs and observe the outputs. You are to stimulate the inputs with all possible input combinations and observe each output to verify its correctness.
- This is called "exhaustive testing". It is suitable for a simple circuit (such as the mux) but is not practical for large circuits with many inputs.
- Instead of using testbench, we use an easy, alternative way to provide the repeatitive clocks to the input signals to test our design.
- Now right click on one of the input signals, in the popup menu, choose **Force Clock**. In the popup menu, set the values as shown below.

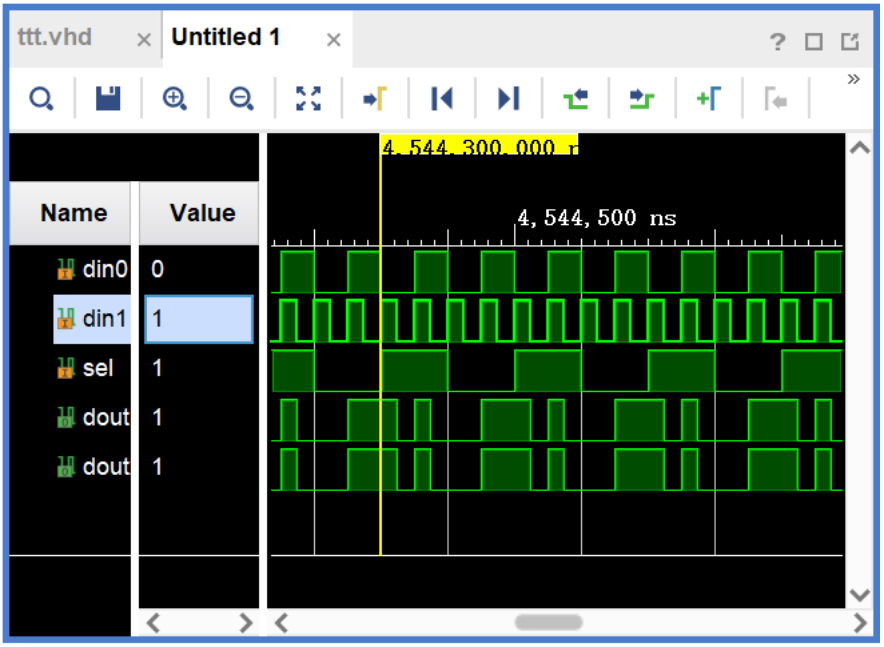


- When you set the input signal values in the above menu, you can set the period of clock of your choice.
- For the 2-1 MUX we have 3 inputs (Din0, Din1, and Sel) where each input can be either logic high (1) or logic low (0).
- • There are 2^3=8 possible input combinations. While we could stimulate each combination individually, an easier approach would be to clock stimulator, as we did just now, to generate the input timing diagrams below.
- Repeat this process to all the input signals.



- Now we need to run the simulation using the simulation controls.
- Run the simulation for 10ms by typing 10ms into the "Run Duration" box and then click "Run For". Your output should look similar to this. Use the zoom controls as needed to make the waveform fill the window.

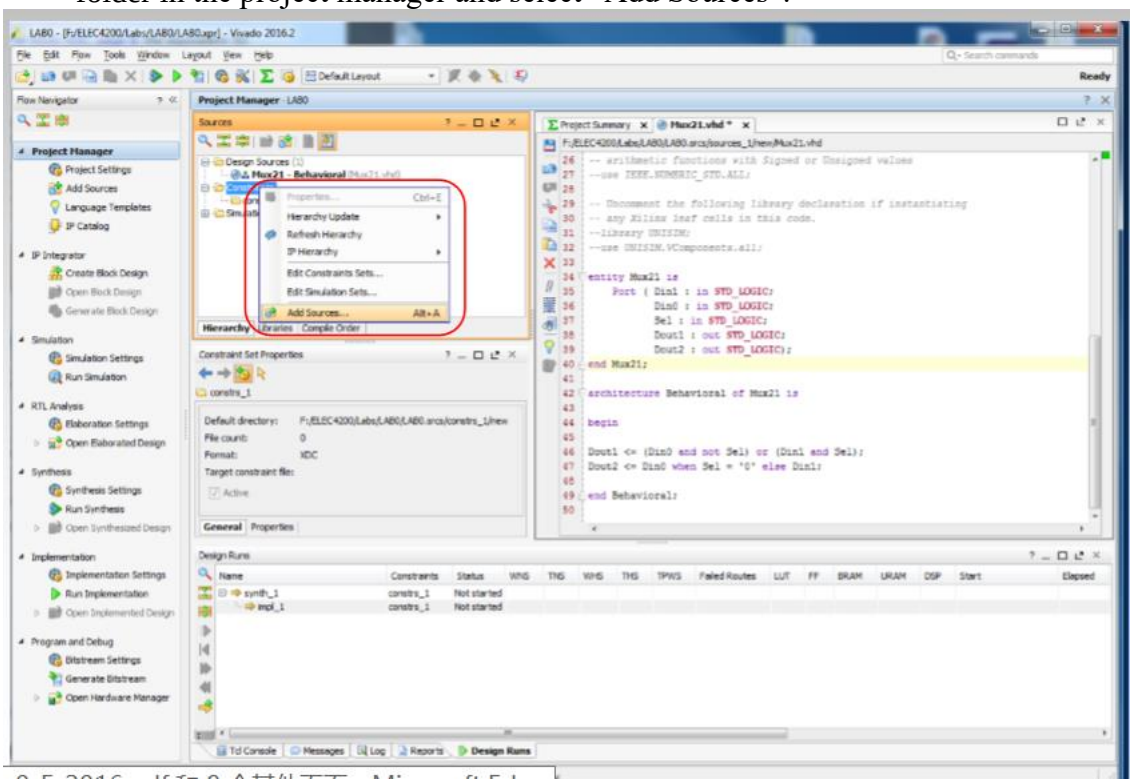




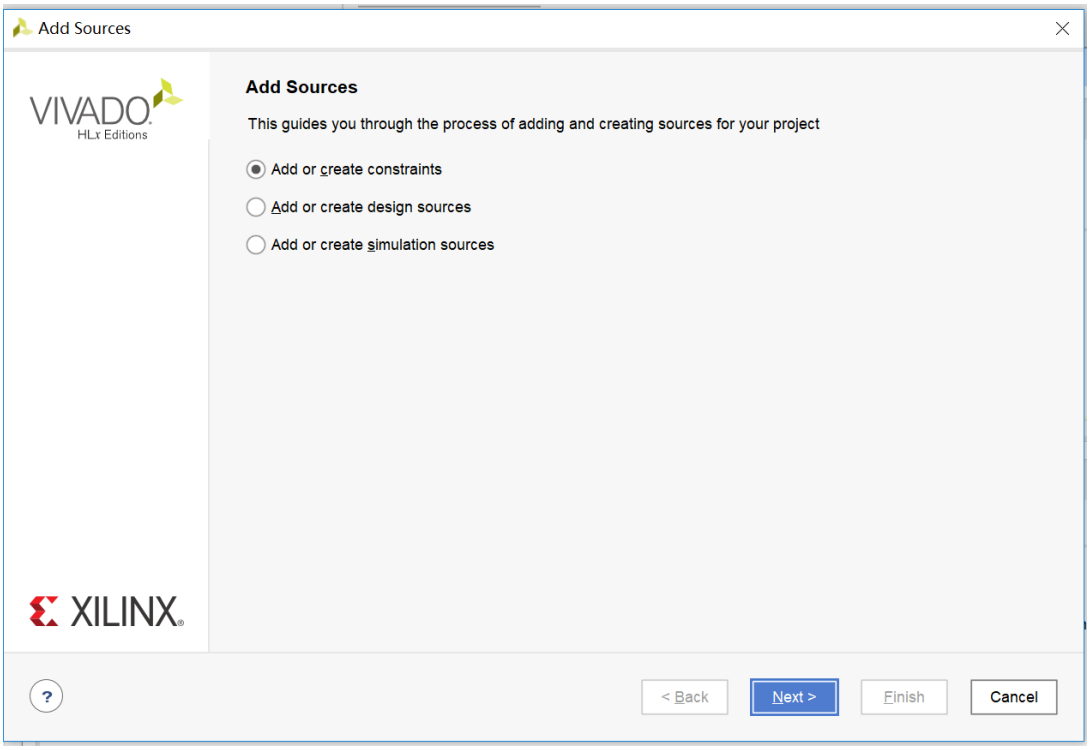
- Examine your results and ensure that they accurately match the operation of a 2-1 MUX for both Dout1 and Dout2.
- If you observe any incorrect results during simulation, go back and debug your circuit. Do not proceed any further until your simulation produces the correct results.

(3) Add constraints to your design for implementing in FPGA

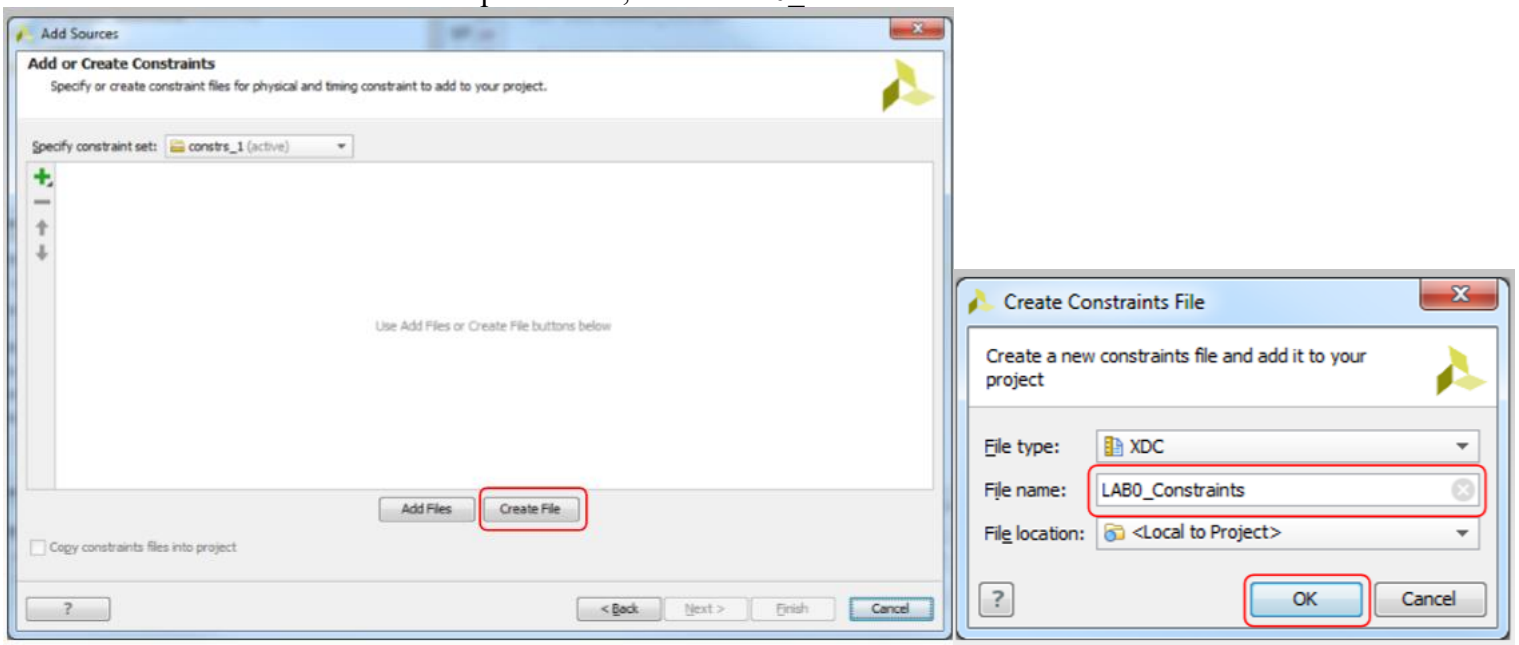
- We need to define a "constraints file" that defines which signals (Din, Dout, Sel) go to which pins on the FPGA.
- Now close the **Simulation Window** if you have not done so. Right click on the "Constraints" folder in the project manager and select "Add Sources".



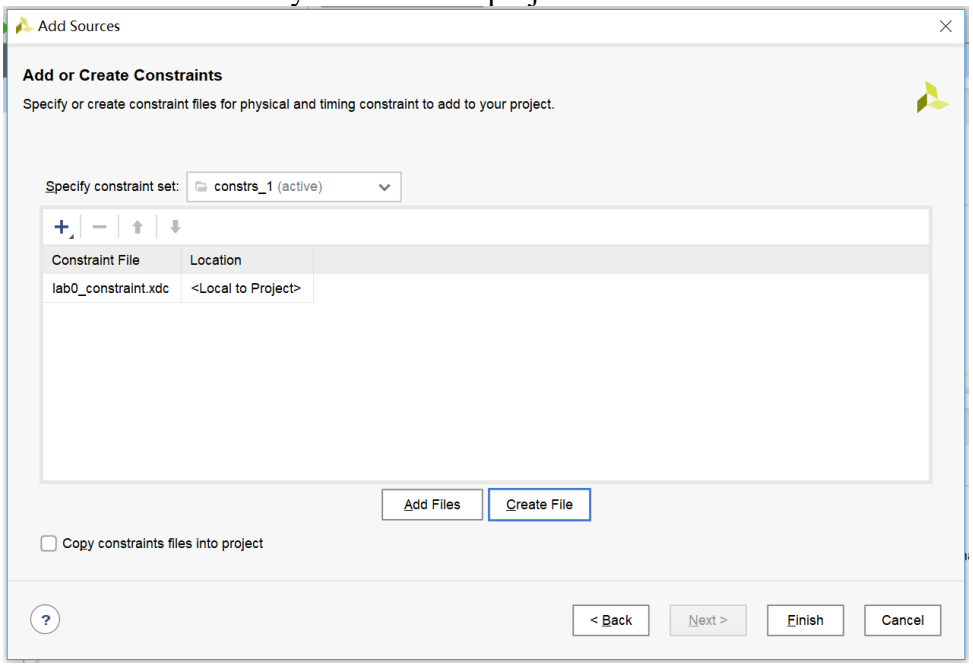
- This opens the "Add Sources Wizard". Make sure "Add or create constraints" is selected and click "Next".



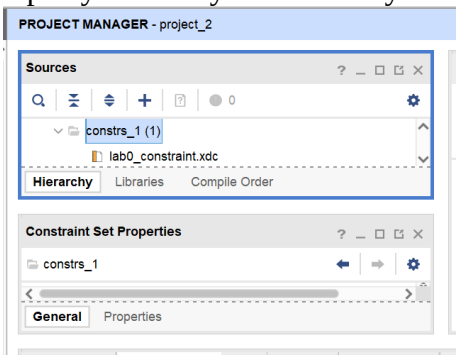
- Click “Create File” and in the dialog box that opens, name your constraints file. As always, use a descriptive name, like “LAB0_Constraints”. Then click “OK”.



- Click “Finish” to add your file to the project. Add Constraints



- Open your newly added file by double clicking on it from the Project Manager.

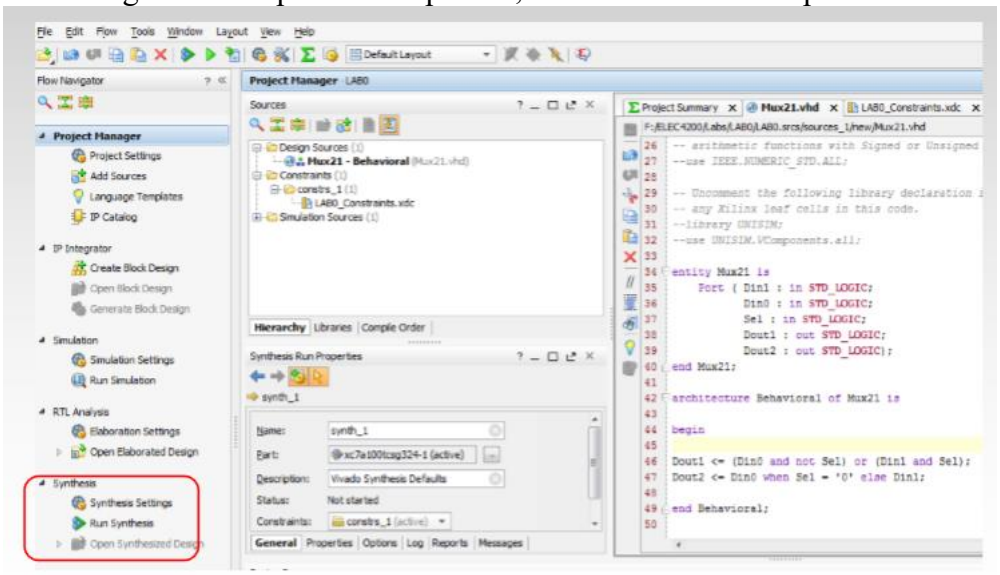


- Add the following lines in the constraints file, renaming the ports to match those in your design. Save after you are done.

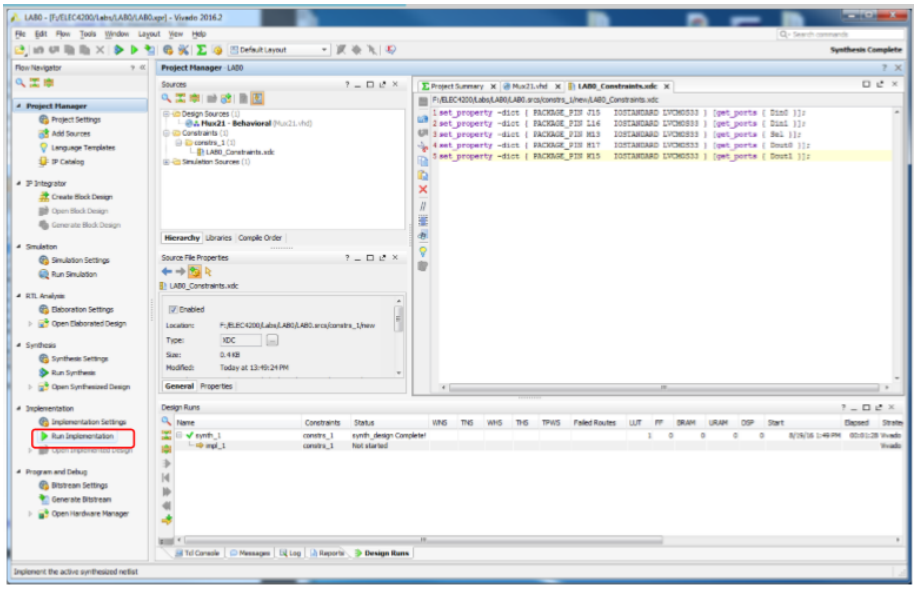
```
set_property -dict { PACKAGE_PIN J15   IOSTANDARD LVCMOS33 } [get_ports { din0}];
set_property -dict { PACKAGE_PIN L16   IOSTANDARD LVCMOS33 } [get_ports { din1}];
set_property -dict { PACKAGE_PIN M13   IOSTANDARD LVCMOS33 } [get_ports { sel}];
set_property -dict { PACKAGE_PIN H17   IOSTANDARD LVCMOS33 } [get_ports { dout0}];
set_property -dict { PACKAGE_PIN K15   IOSTANDARD LVCMOS33 } [get_ports { dout1}];
```

(4) Implement your design in FPGA

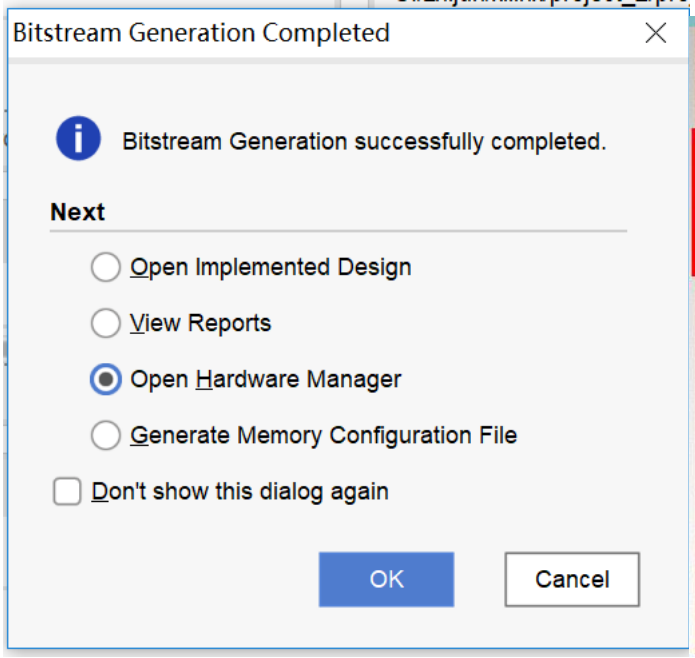
- Now you are going to have Vivado "synthesize" your design into generic digital logic components.
- In the Flow Navigator, click "Run Synthesis".
- This may take several minutes to run, so be patient.
- A dialog box will open on completion, either select "Run Implementation" or hit "Cancel".

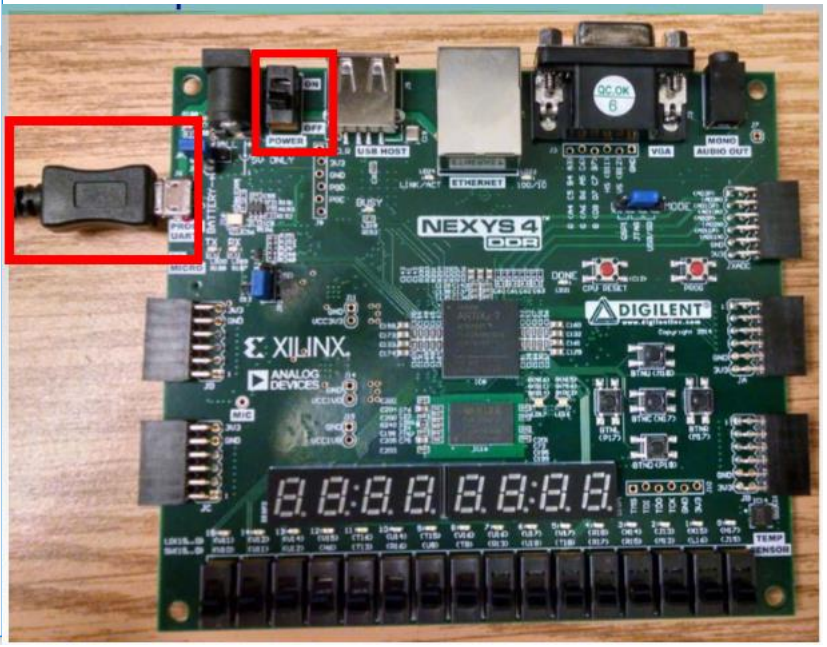


- Click on "Implement Design". This maps the synthesized design to the hardware, and routes I/O ports to the pins specified in the constraints file.
- Again, this will take several minutes to run.
- A dialog box will open on completion, either select "Generate Bitstream" or hit "Cancel".



- With the design now implemented, it is time to generate the bitstream, or the file that will be downloaded to the FPGA.
- Click “Generate Bitstream”.
- A dialog box will open on completion, either select “Open Hardware Manager” or hit “Cancel”.
- Plug in the USB cable between a PC USB port and the USB port on the FPGA board to access the JTAG programming module.
- Flip the board power switch from OFF to ON





- Click on “Open Target” => “Auto Connect” to connect to your Nexys 4 board.
- If you encounter any errors, verify that the board is plugged in and powered on.
- Click on “Program Device”, and the device name. The bitstream file will appear, then clock on “Program”.
- Verify correct operation of the circuit using the switches and observe the output on the LEDs. If you encounter any bugs, attempt to figure out what went wrong before asking for help.
- Ensure that you apply all possible input combinations, as you did in simulation, and verify that the outputs match the simulation results.
- After verifying the circuit is correct, call the tutors over and demonstrate the circuit. GOOD LUCK!

(5) Clean up!

- Turn off the board, unplug the USB cable, put them back in their box, and return the box to the tutors.
- Close Vivado, any other open programs, and save all files.
- Don’t forget to log out of your machine and take any USB drives with y