COMP160, Lab 1

This lab consists of 2 parts:

- Part 1 is the introductory tutorial.
- Part 2 is the same exercise in Verilog.
- Part 3 is your assignment. It's meant to reinforce the tutorial.

Part 1. Schematic Entry Tutorial

You are going to create a 3 input combinational circuit. You will implement the following function:

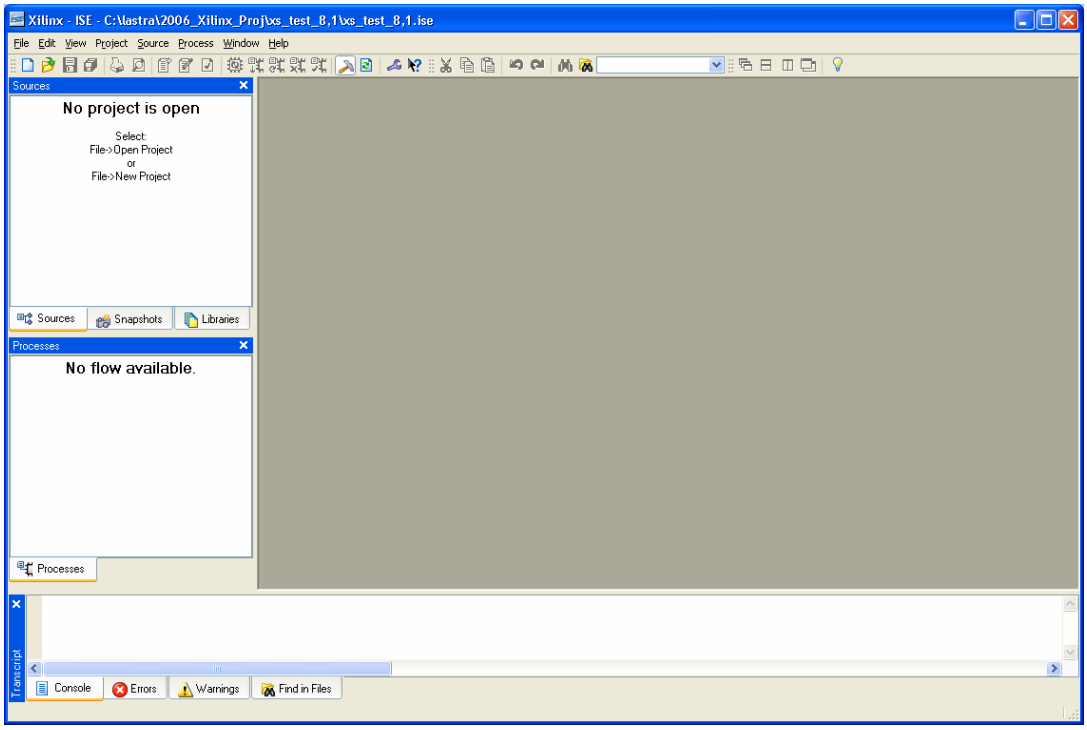
$$D = AB + \overline{C}$$

The procedure will be to

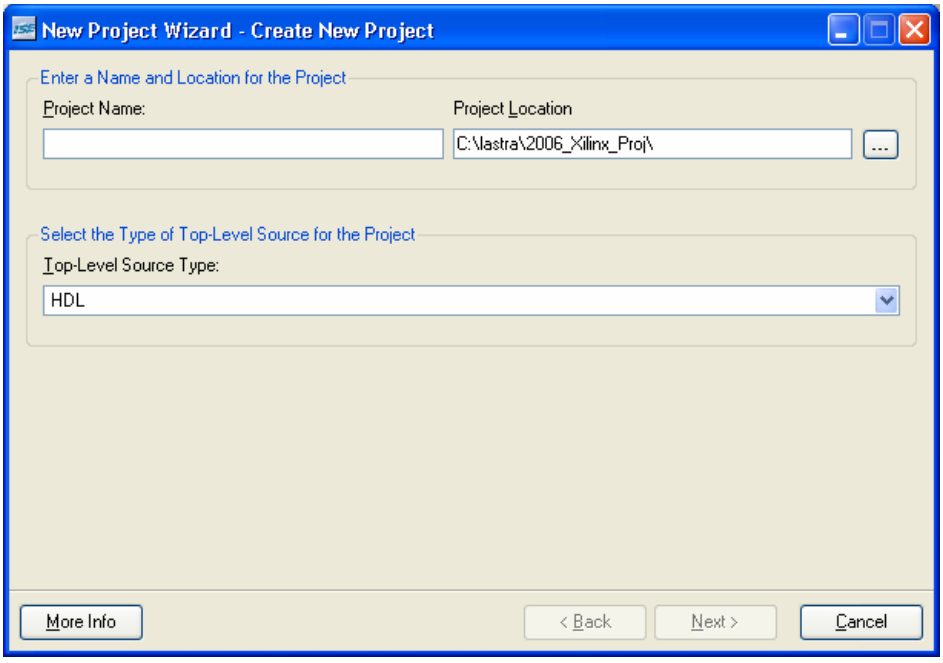
- Create a new project
- Add a schematic file
- Simulate the circuit
- Assign the A, B, C, and D inputs and outputs to chip pins.
- Download your design to the XSA board
- Test the result
- Repeat in Verilog

Creating a Project

Start by double clicking on the Project Navigator icon (on the desktop or under Xilinx in the Start Menu). This is what you will see.



To create new project, select File -> New Project, which will bring up following window.

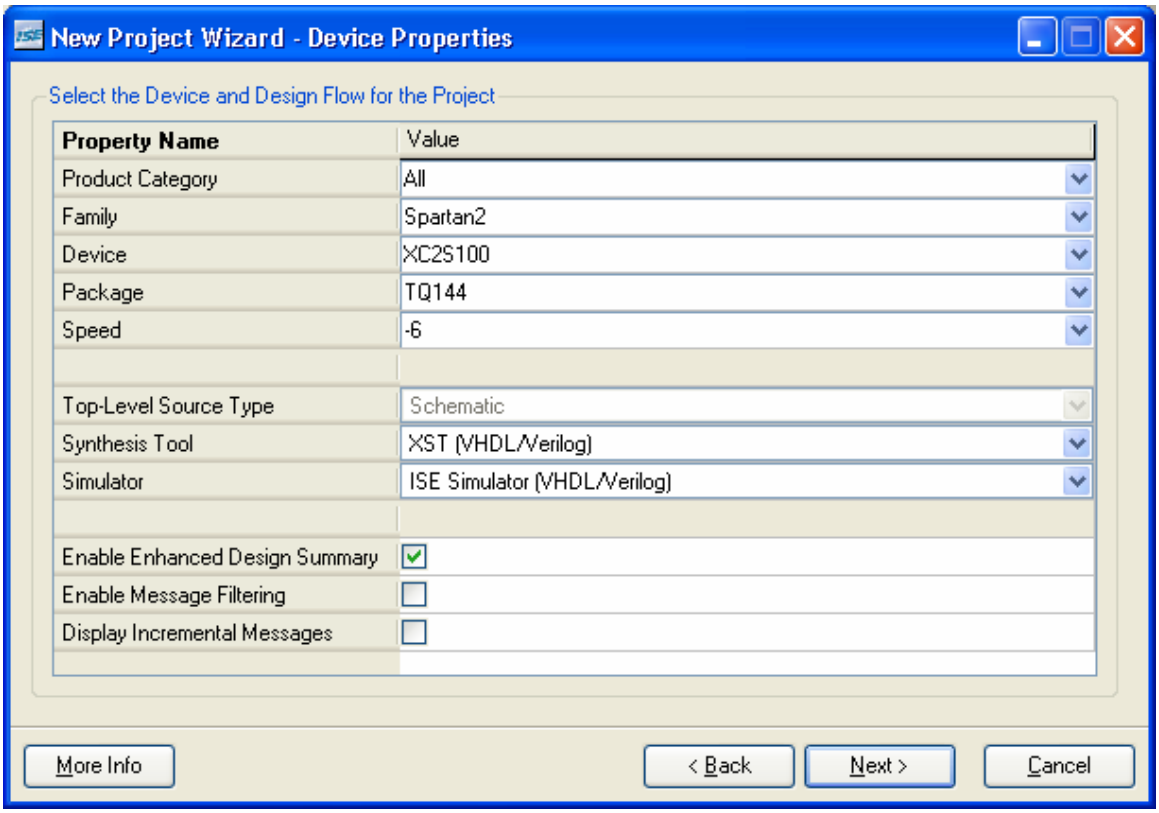


Choose a name.

Project Location: click on **…** to browse and change the folder to some convenient place. *Make sure there are spaces in the name!* I suggest making a folder in D: with your name because the C drive is small on most of the lab machines.

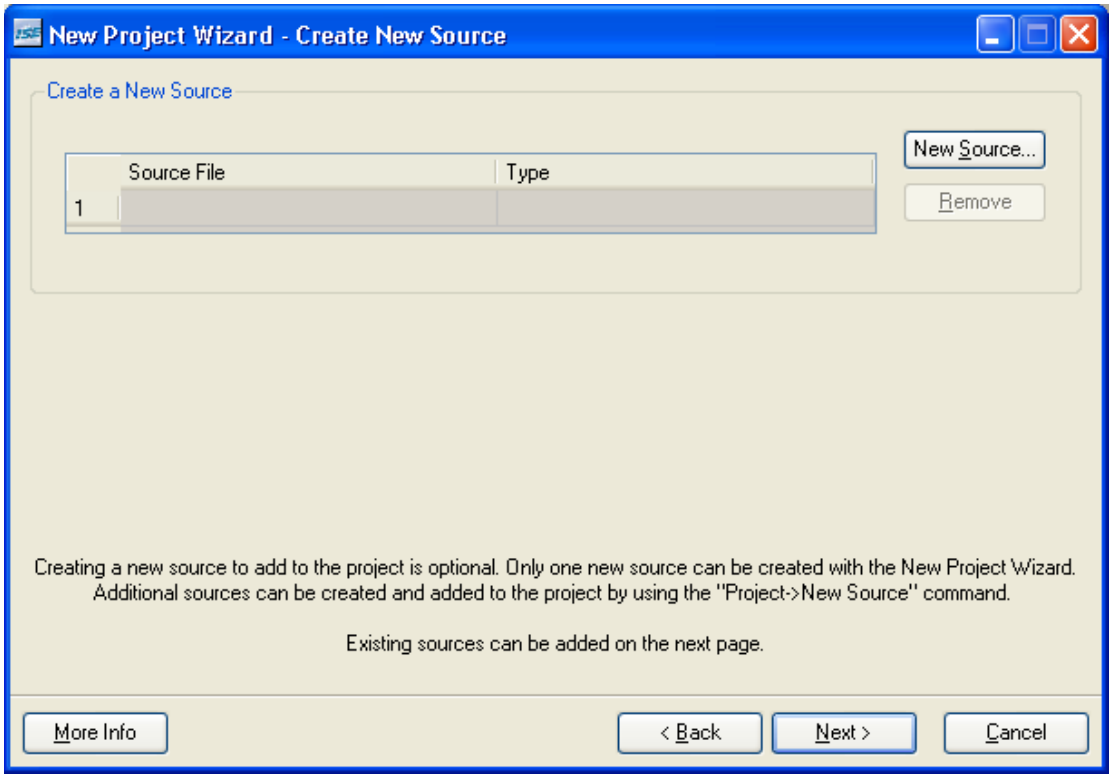
Choose *schematic* for now. Later you'll use HDL (Hardware Description Languages).

Click *Next.*

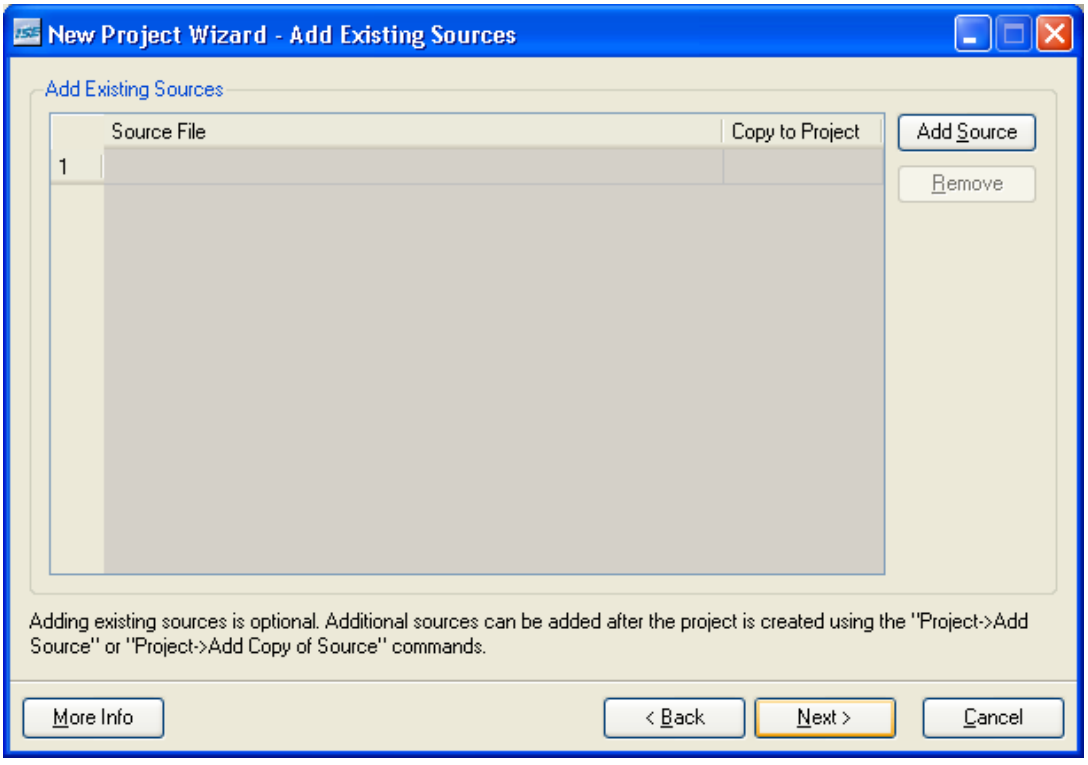


Project Device Options should be as follows:
Family: Spartan2
Device: xc2s100-5tq144
Simulator: ISE Simulator

Then click ***Next***.

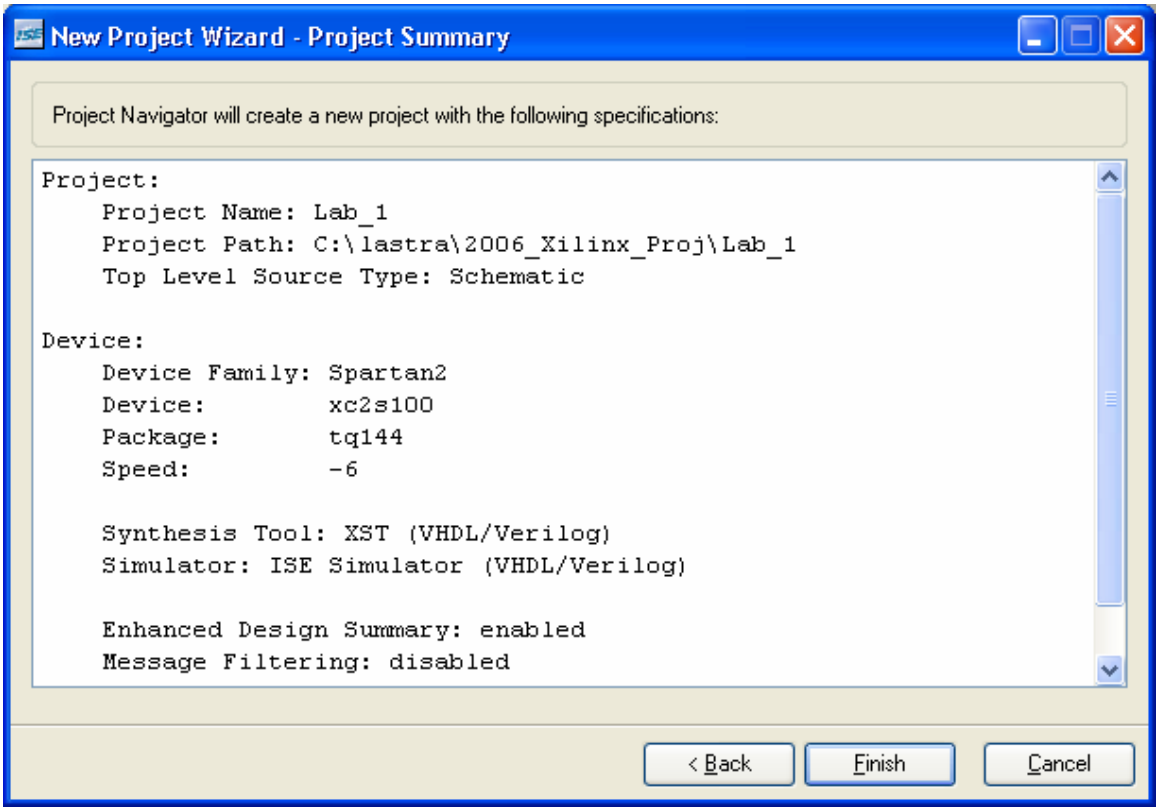


We'll add a file later, so just click ***Next.***



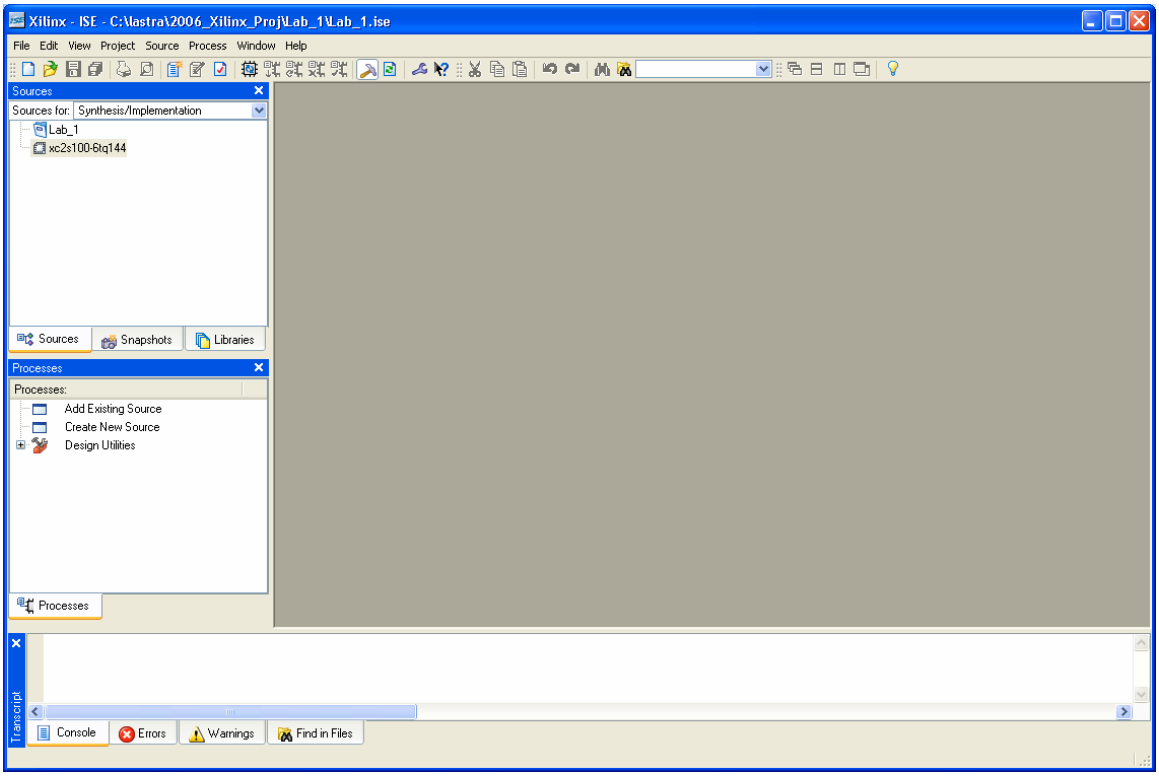
Click ***Next.***

You'll see a summary of what you've selected. Click *Finish*.



Creating a Schematic File

After you are done with this, back in Project Navigator you should have,

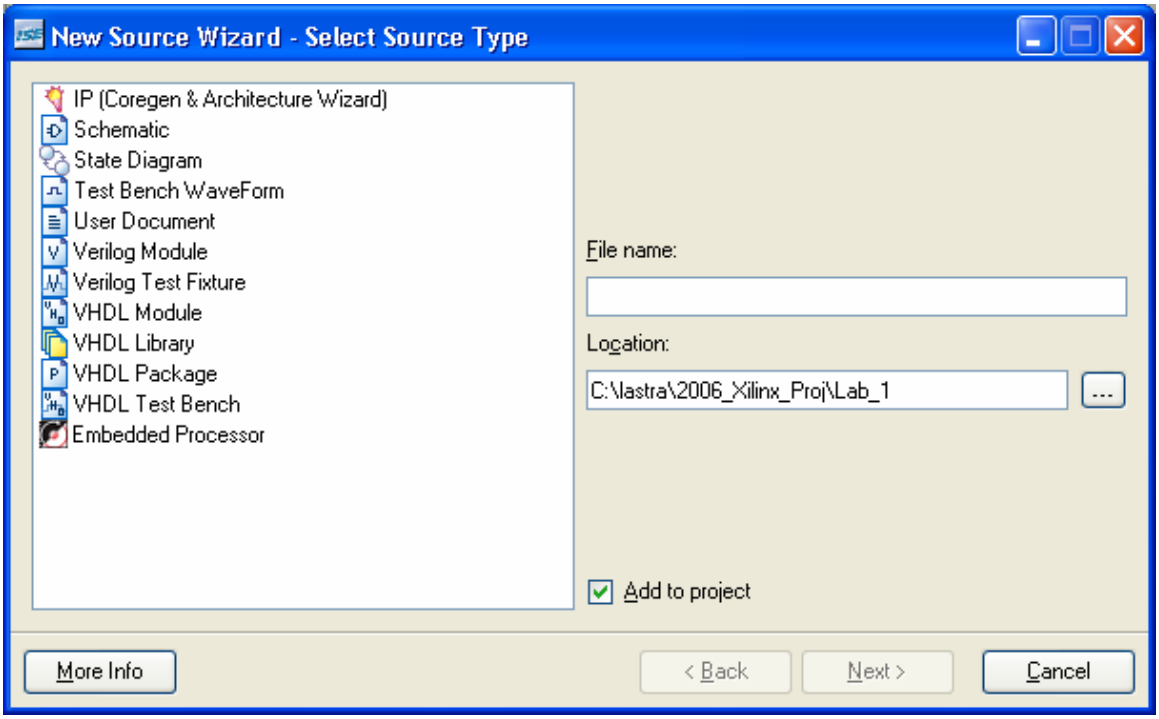


Now, either

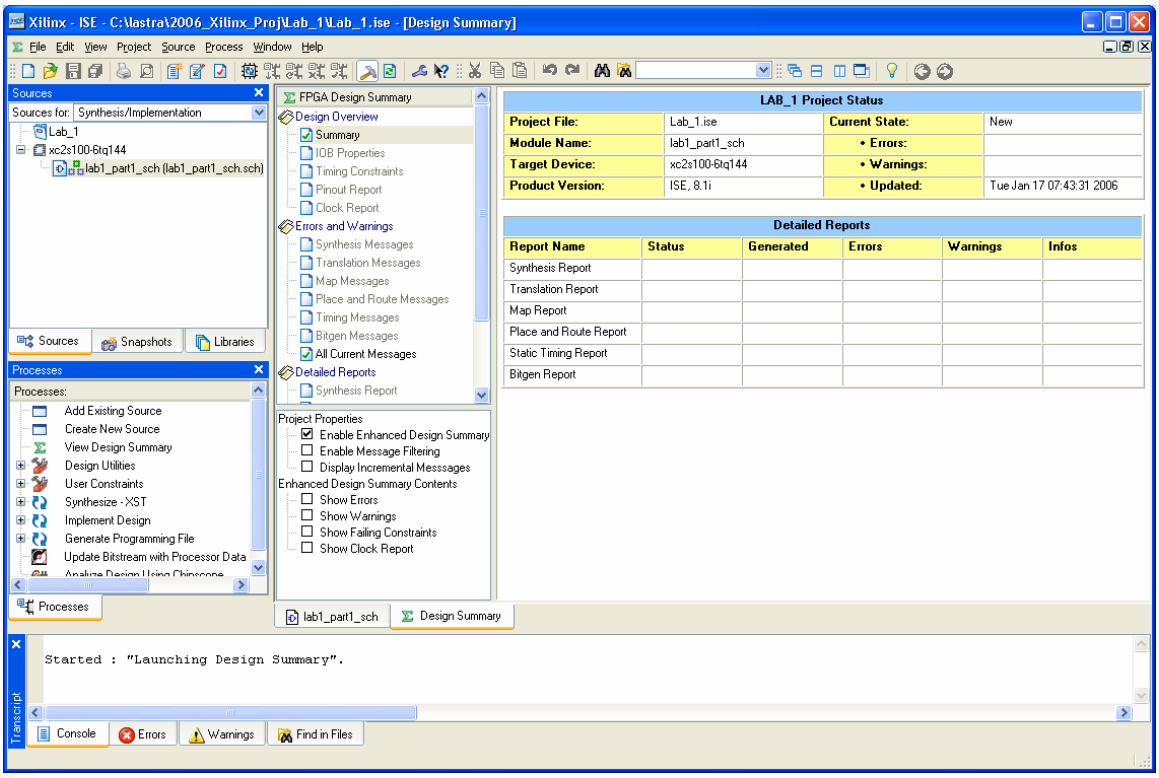
(a) highlight xc2s100-5tq144 in the Sources window and double click on *Create New Source* in the Processes window, or
(b) right-click xc2s100-5tq144 and select *New Source*.

You'll find that both of these modes of interaction are available for most actions.

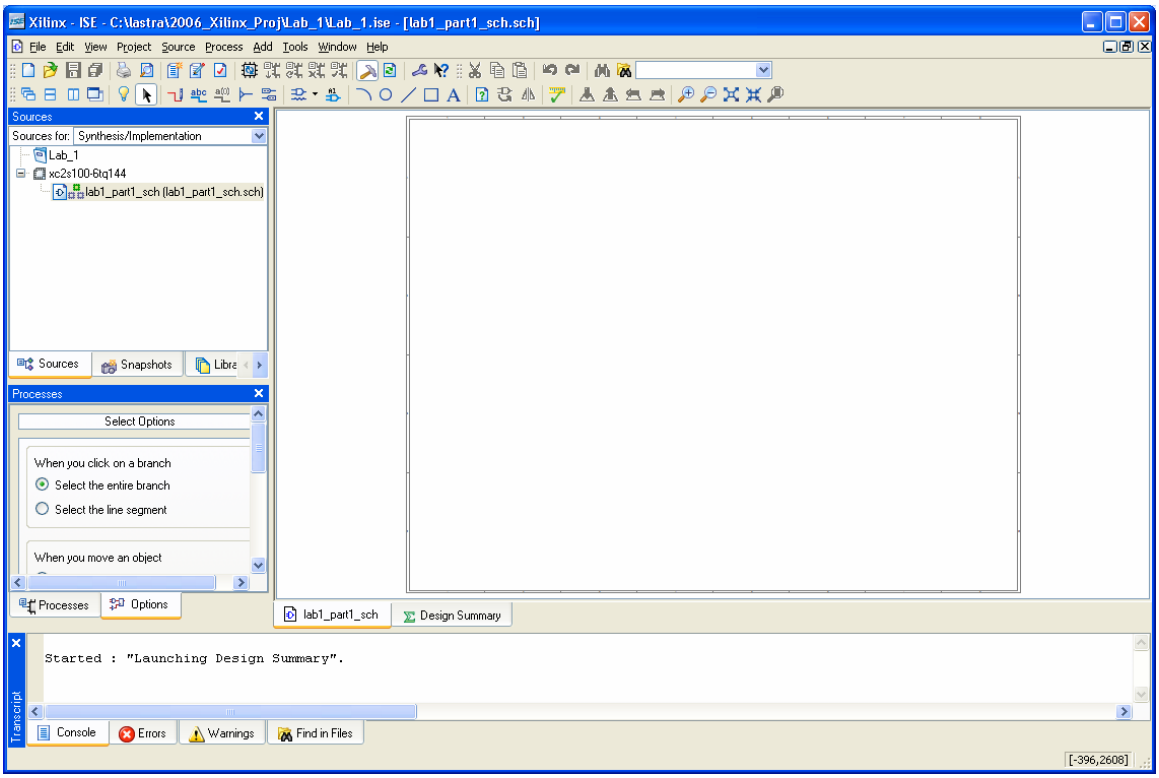
You will see the following window.



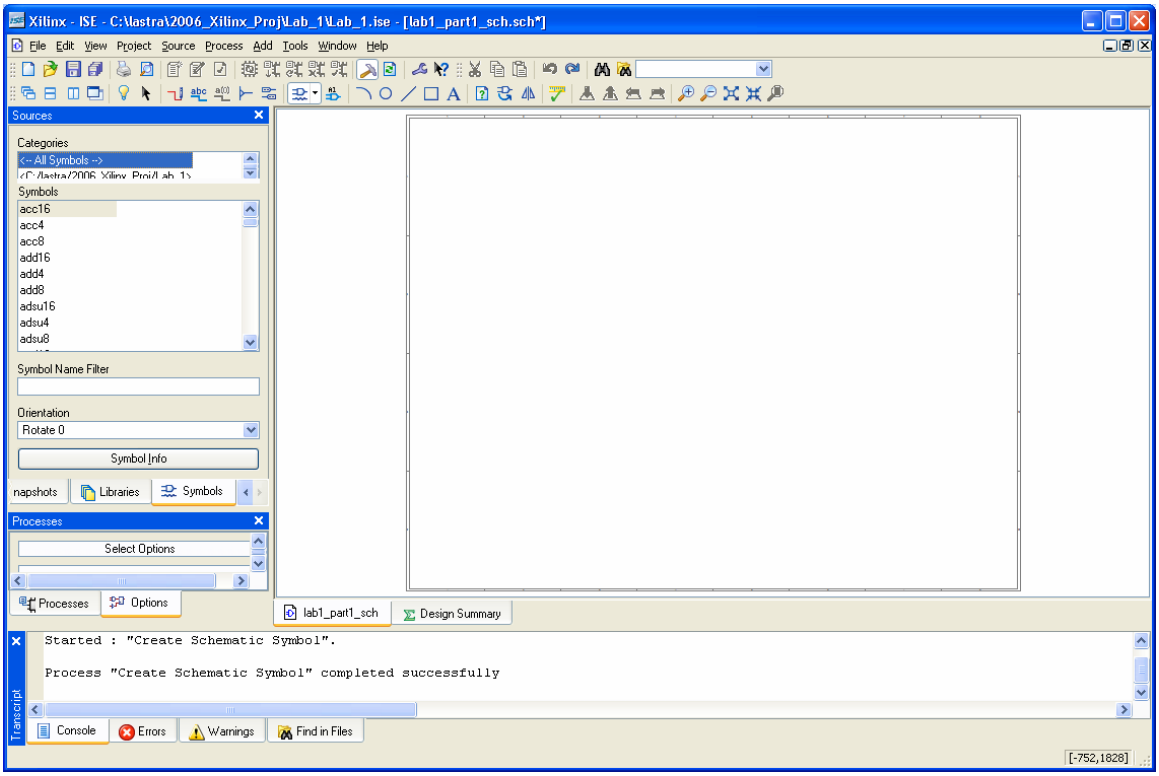
On the left panel, select *Schematic* and in the file name box, type *lab1_part1_sch* (no spaces!). Click next, and then finish.



Click on the lab1_part1_sch tab of the main window.

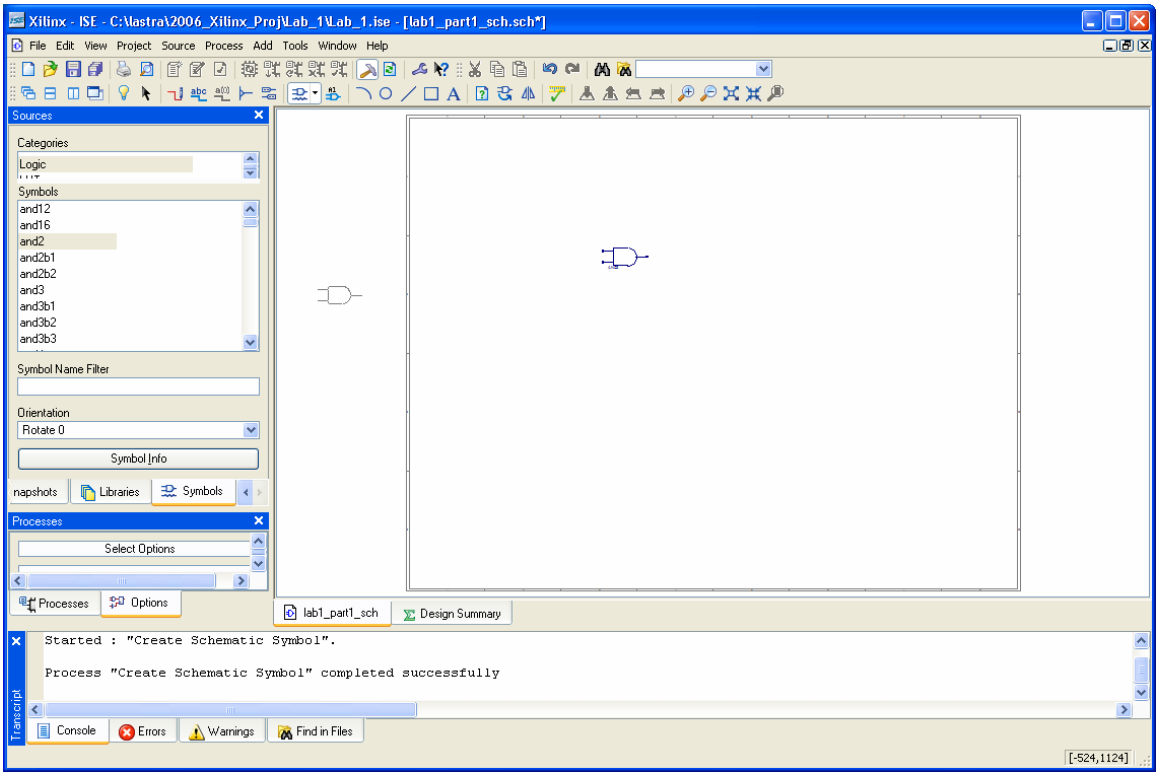


Click on the *Symbols* tab in the Sources pane. You may have to scroll the tab menu to the right. Note that you can change the sizes of the panes.

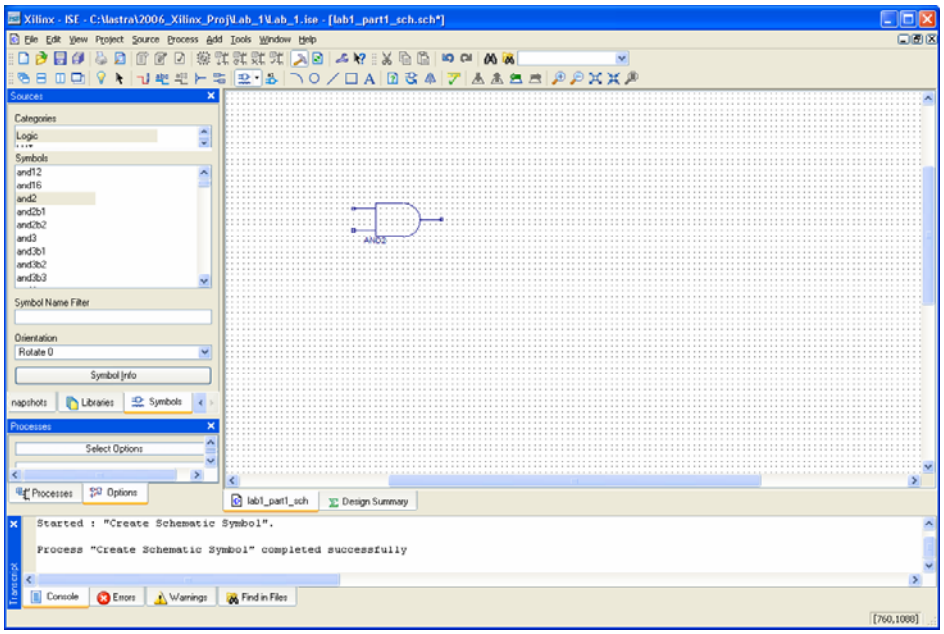


You can select a *category* in order to reduce the number of symbols shown. Select *Logic*, which will show you all of the gates.

To implement $D = AB + \overline{C}$ you will need a 2 input AND, an inverter, and a 2 input OR. Under Symbols highlight *and2*, go to the schematic window and click in it. An AND symbol will appear where you clicked. You'll see something like this.

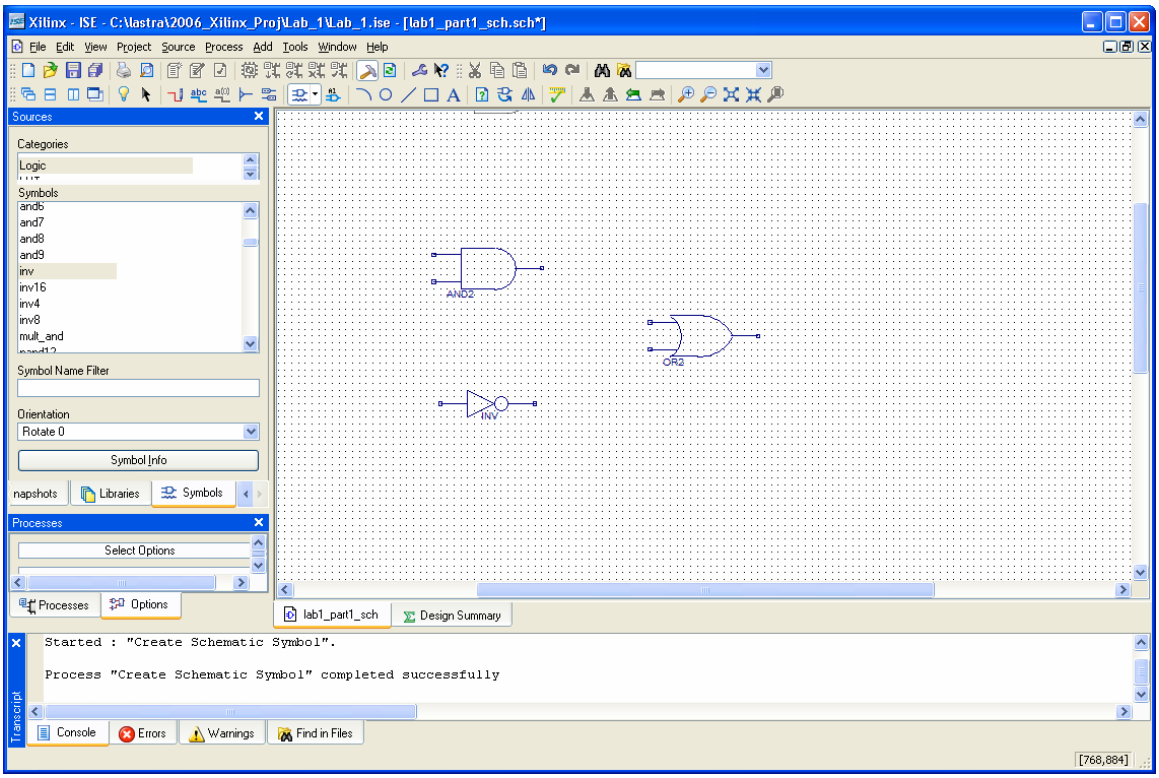


You can Zoom in or out to adjust what you see in your schematic. At some point you may see the grid (shown as dots) onto which you can place symbols.



In the symbols list find an *or2*. Highlight it and click in the schematic. Since C is inverted, you need an *inv*. Find it and repeat the above procedure.

When you are done, your schematic should look something like this.

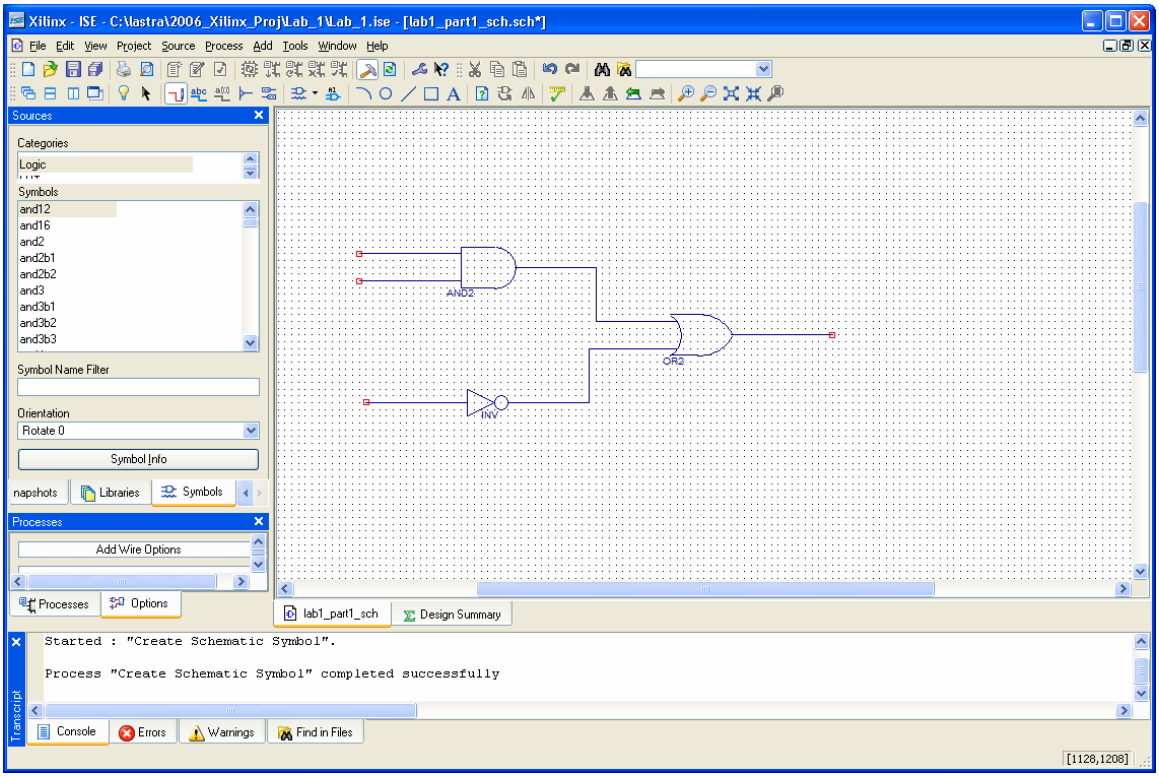


Now let's connect the symbols. On the menu bar, find a wire icon (the one with the pencil),

select it, and connect the output of the AND with one of the inputs of the OR and the INV to the other input of OR. You'll need to double click while over the terminal to stop the routing at the end of the run of wiring.

Also, extend the inputs of the AND and INV, and the output of the OR with a wire. For the latter three, since you're not connecting one end of the wire to anything, you'll have to double click on the page where you want to stop the routing.

Your schematic should look something like this.

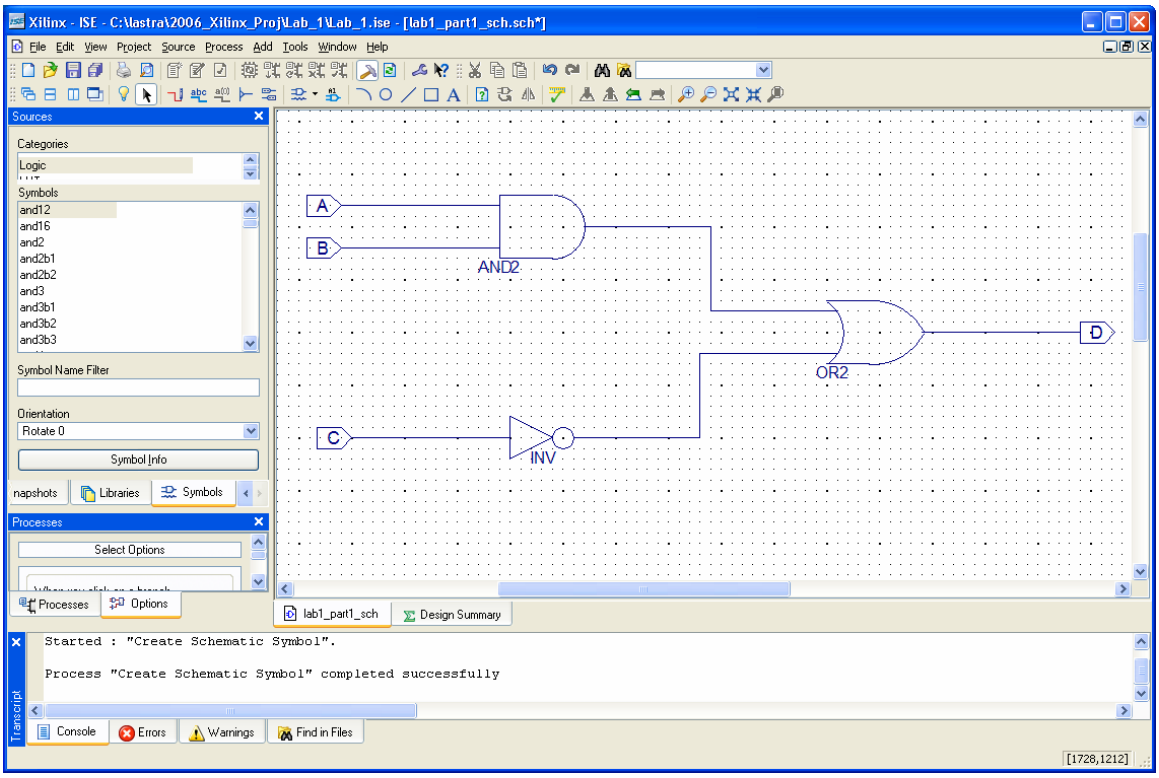


On the menu bar find the *Add I/O Marker* icon.

Add 3 input symbols, for each input of AND and the INV, and an output symbol for OR.

Switch back to the arrow. Hovering over a symbol will show you the properties. You can double click on the I/O symbols to change their properties. Rename the inputs and outputs. The inputs should be A, B and C and the output D.

On the toolbar there is a check-mark symbol. Clicking on it will tell you whether you have any errors in your schematic design. There should be none. If everything is fine, save it and close this window. Your schematic should look like this (zoomed so you can read the symbols).



Save and we'll move on to simulate the design.

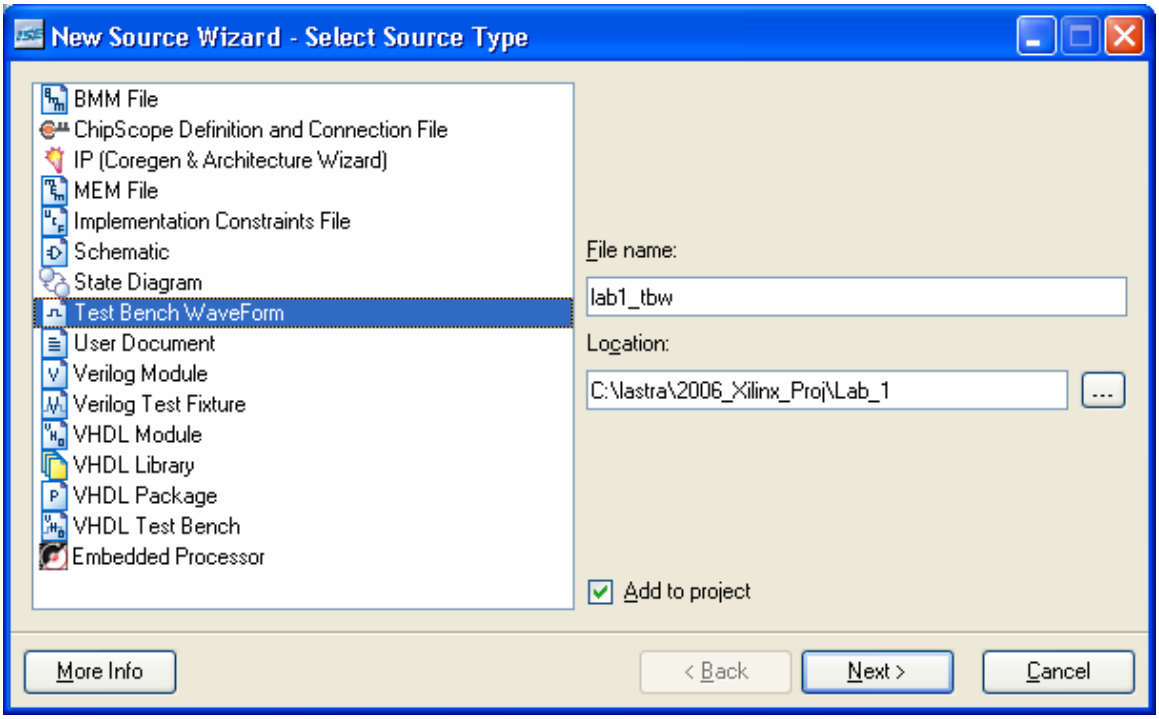
Simple Simulation

There are two simulators we can use. In this tutorial, we'll try the simpler built-in simulator. Recall that we choose the type of simulator when we started the project.

In order to simulate, we need to specify a set of *test vectors*. These are inputs to the circuit under test. We can specify these in multiple ways. Today we'll try the visual test waveform editor.

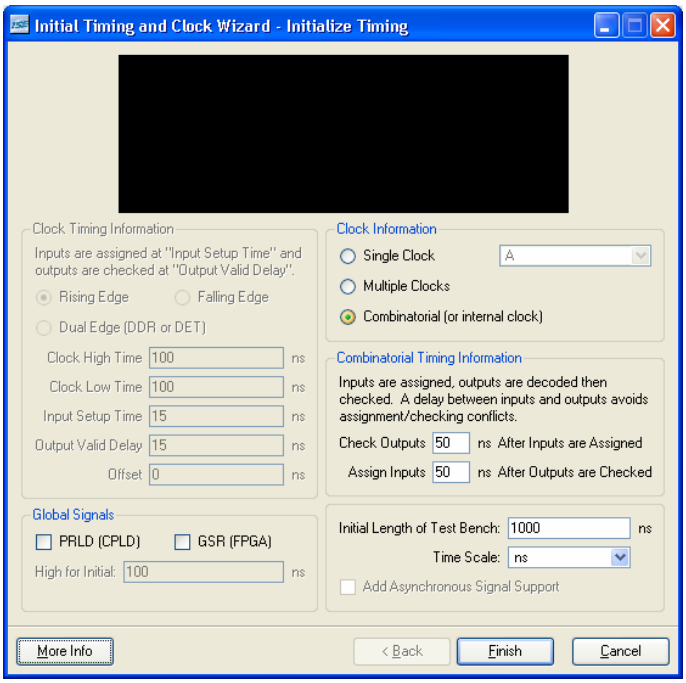
Go back to the Sources tab in the Sources window. Either select the menu item named *Project->New Source*, or right click on the sources window and select *New Source…*

In the dialog box, select *Test Bench Waveform* and name it lab1_tbw.

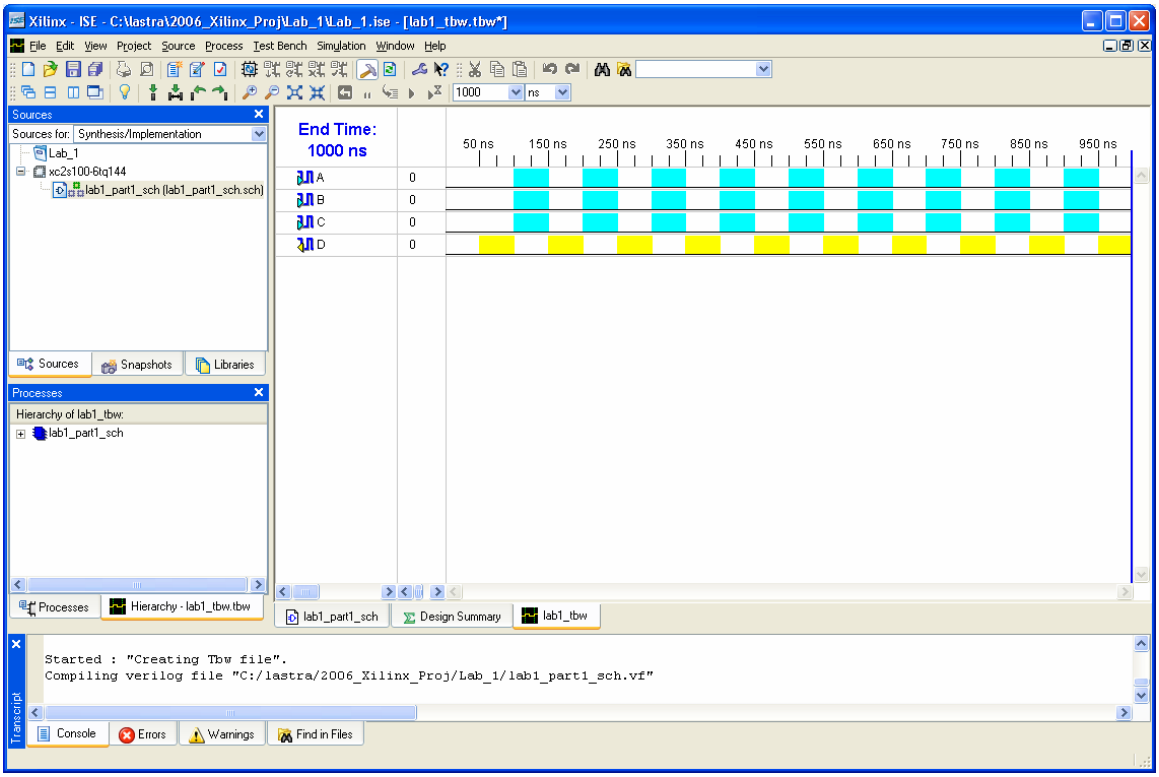


Click next. Make sure it's associated with the schematic diagram (should be no problem since it's the only design in the project at the moment). Click *Next* and *Finish*.

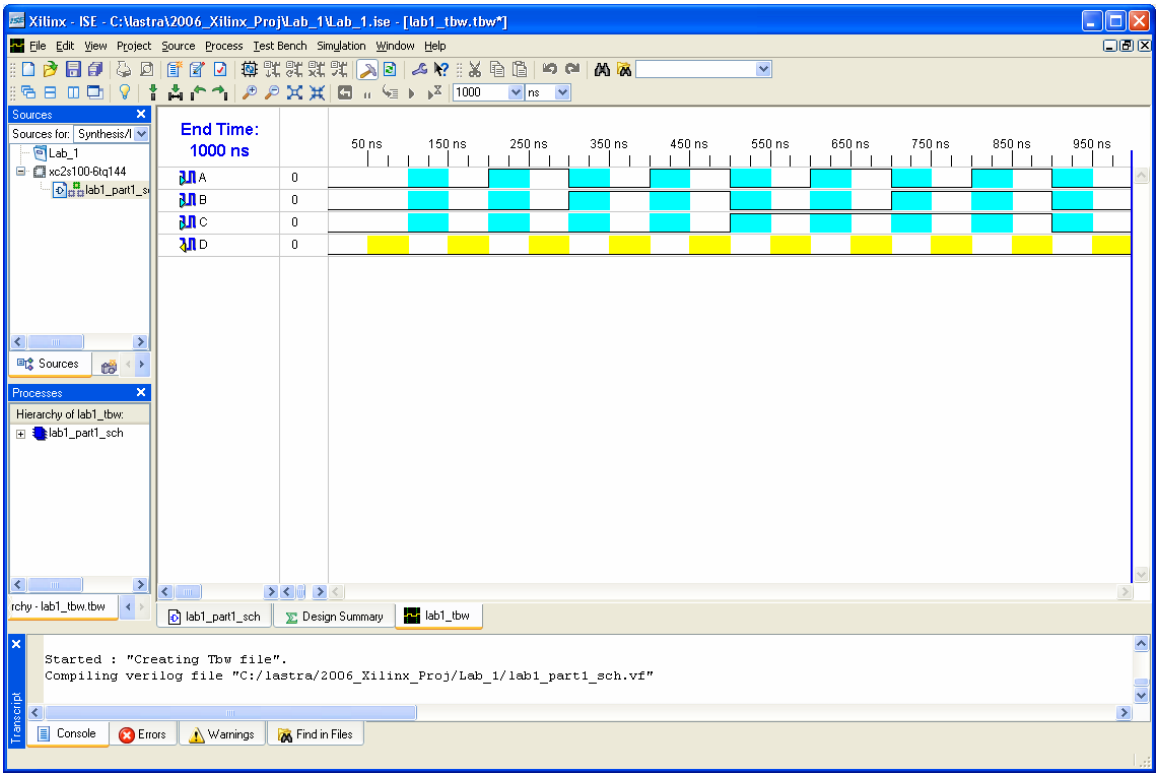
The clock timing dialog box will pop up. Select *Cimbinatorial* on the upper right panel and click *Finish*.



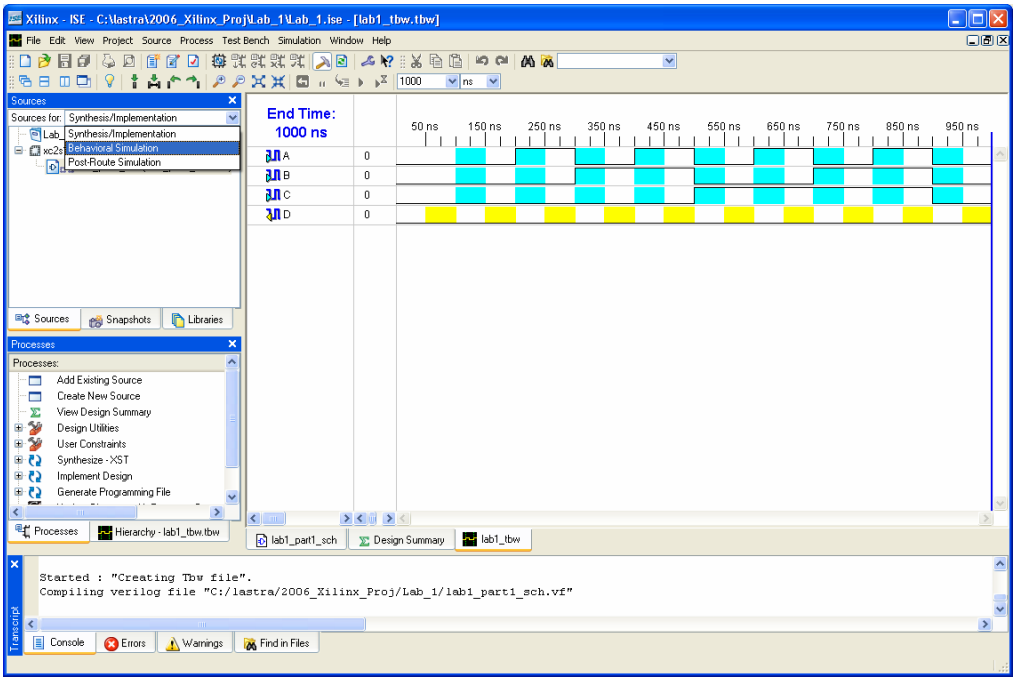
Now you'll see a waveform window.



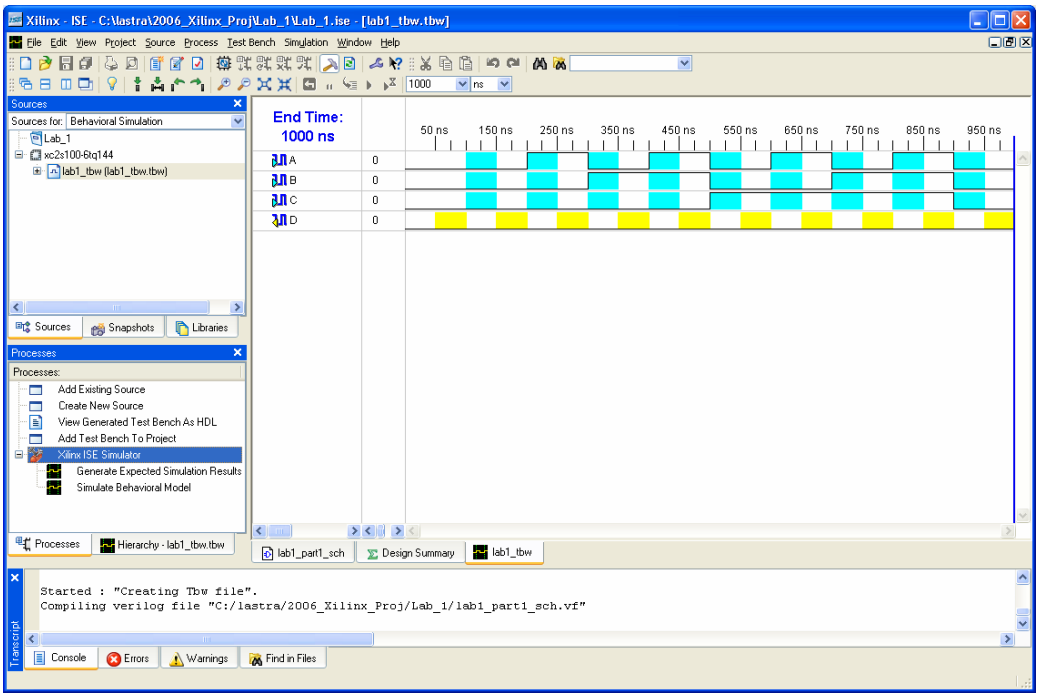
We can click on the blue boxes to toggle the binary values. Enter a set of values to test every possible combination. In the example below, I've used A as the low order of an input sequence from 0 to 7 (and moved the boundary to the left windows to increase the size of the waveform).



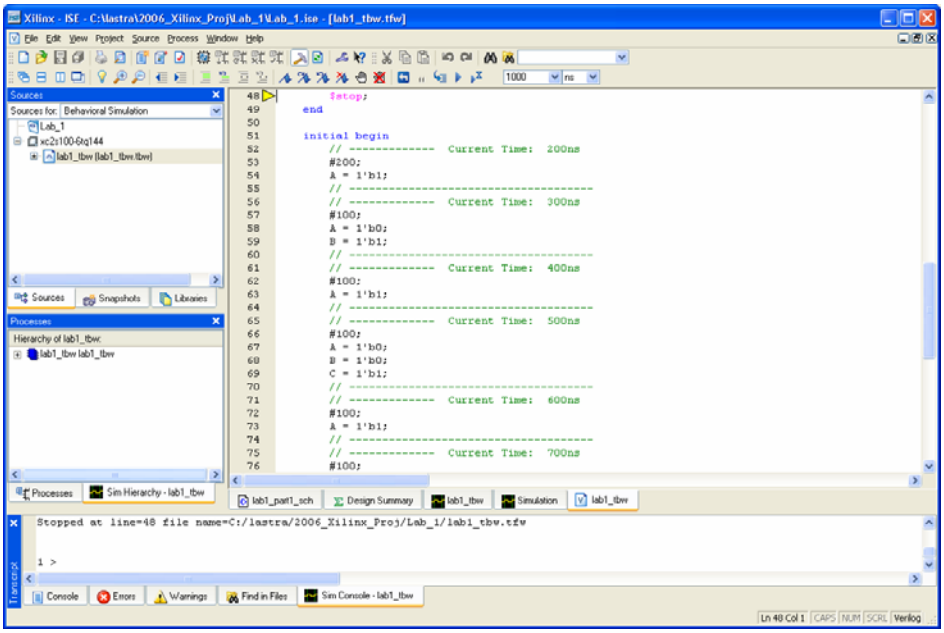
You need to change the *Sources* in order to see the simulation commands. Select the Sources tab in the Sources window and on the pull-down (below) select *Behavioral Simulation.*



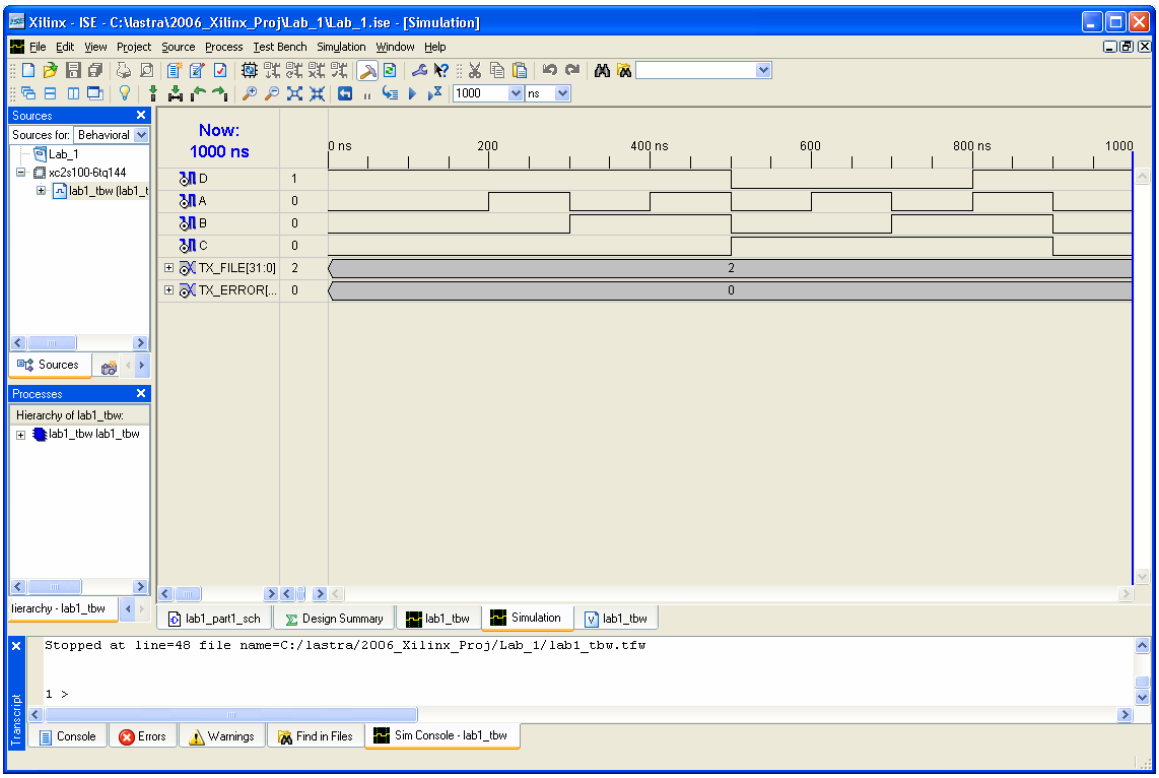
On the processes tab (lower) you'll see the simulation options. Double click the *Behavioral Simulation* item (or right click and select run).



Two new tabs will appear in the main window, one with the test bench as text (below)



and another with the results as a waveform.

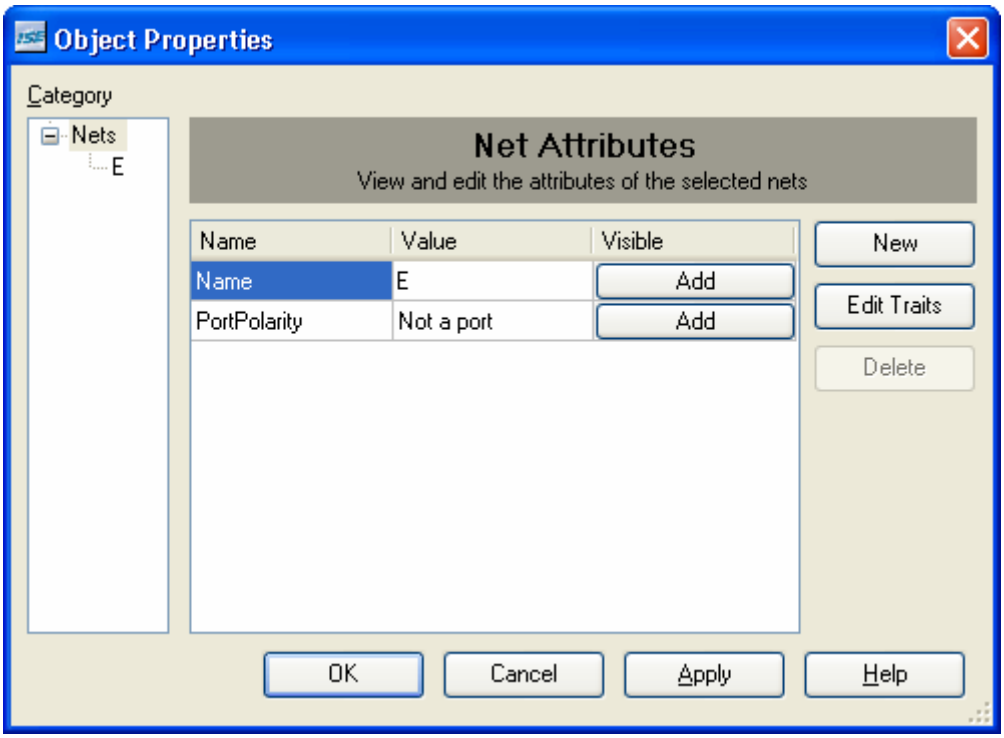


Verify the results of the simulation.

How about if we want to see the values of some internal nodes in the simulation? For example, the value at the wire between the AND gate and the OR gate.

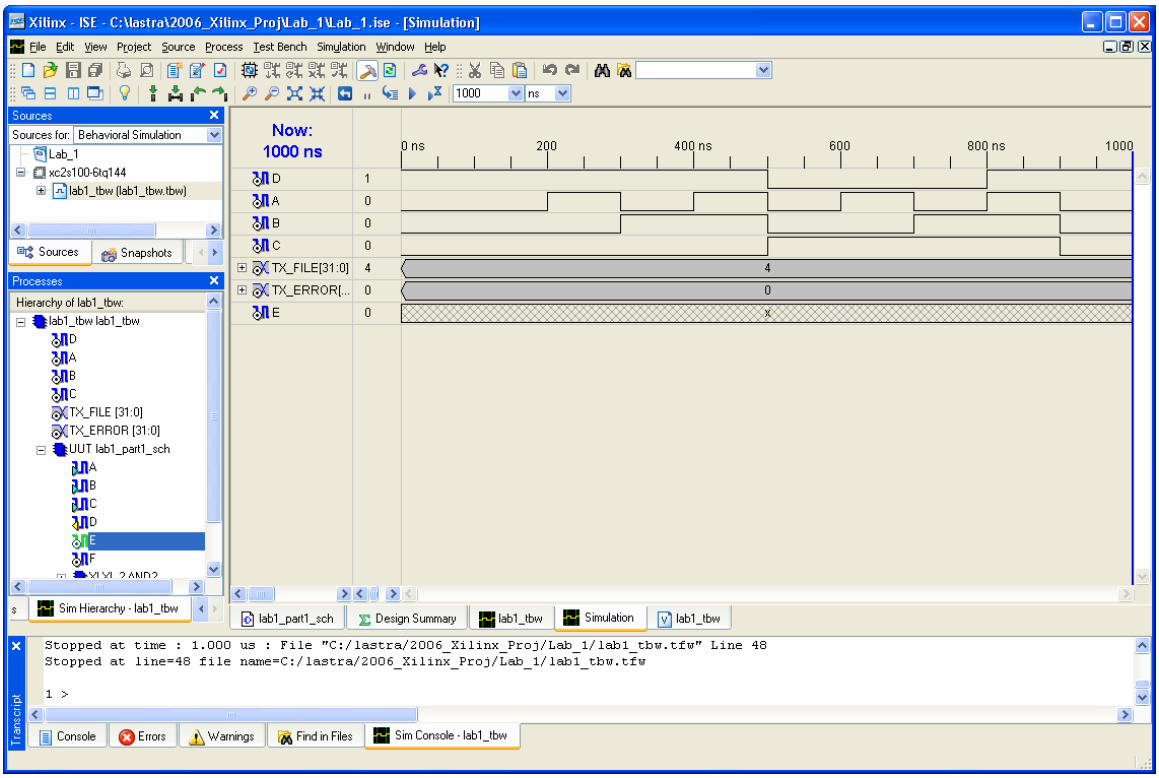
First, close the simulation window (say yes) to stop the simulation.

Now, let's name the internal wires. Go back to the schematic, and double click on the wire between the AND and OR. Name it E.

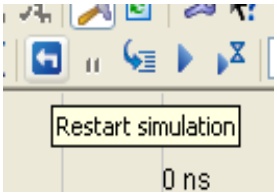


Name the wire between the inverter and the OR gate F.

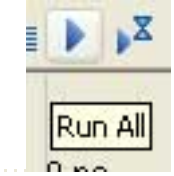
Go back and rerun the simulation. The diagram will look the same. On the processes window, click on the *Sim_Hierarchy* tab. Open the UUT (*Unit Under Test)* item in order to see the list of signals used in the design. The new E and F should be there, along with the inputs and output. Drag the E over to the signal name column of the waveform window.



Do the same with the F signal. Now you can rerun the simulation by using some of the controls on the toolbar.

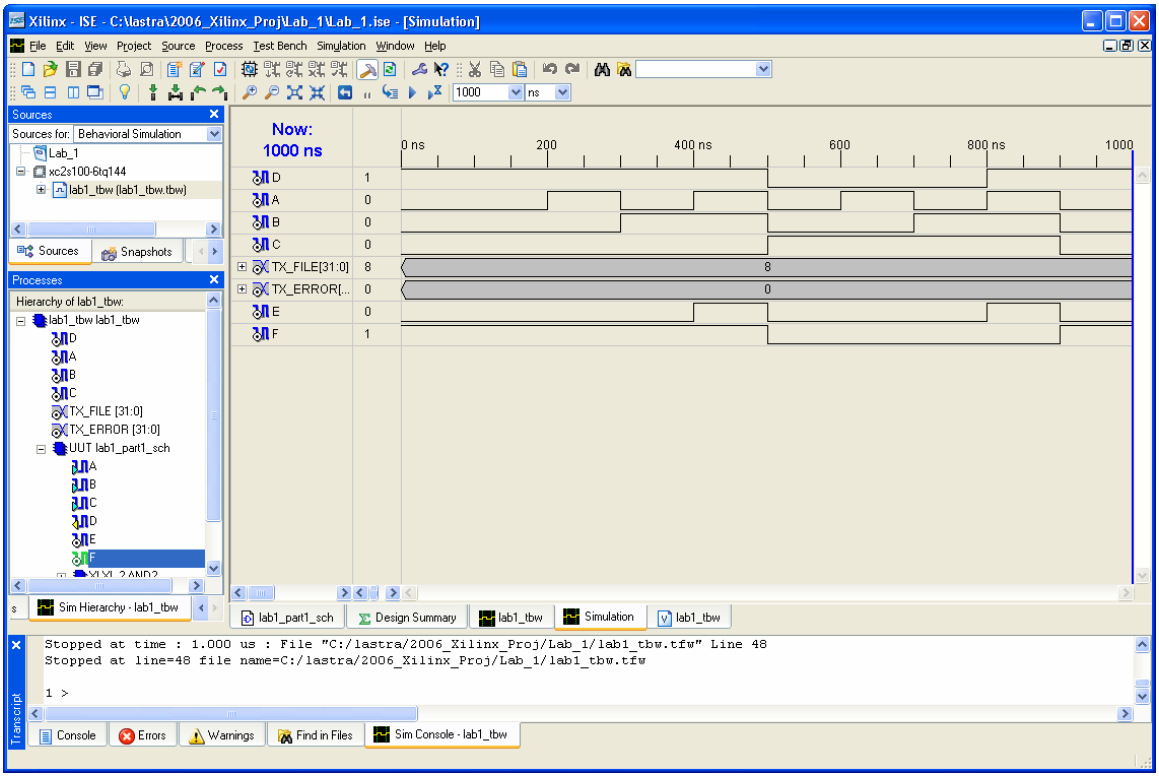


The left arrow will restart the simulation. This just clears values, it doesn't rerun the simulation.



The right triangle will *run* the simulation.

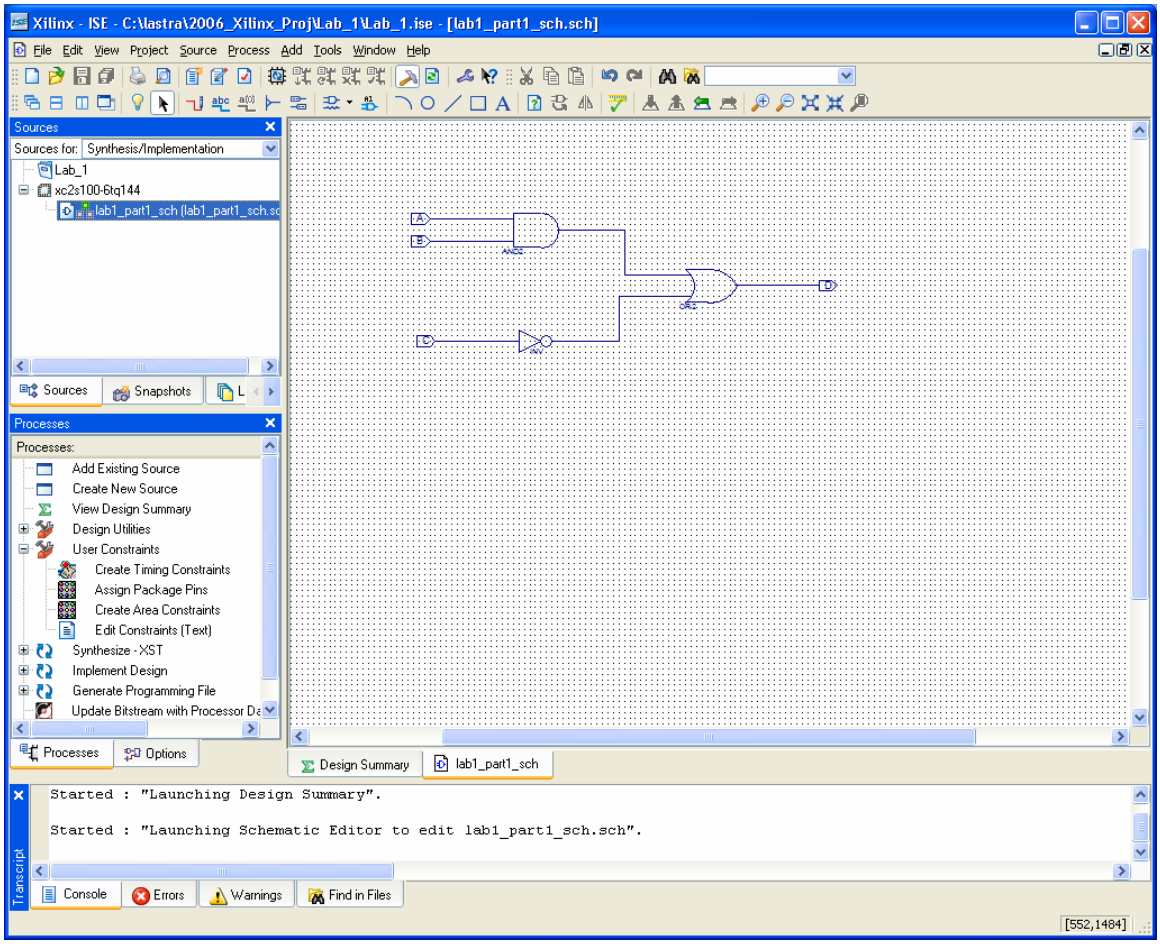
You'll see the values of E and F.



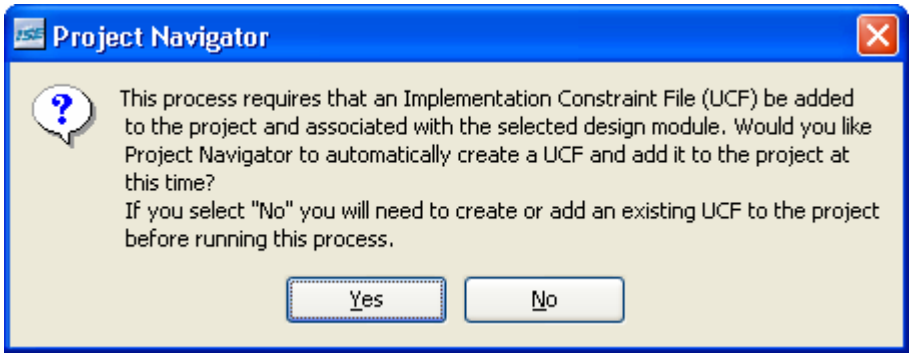
Now that you've seen how to run a basic simulation, let's move on to the steps necessary to try your design on the actual hardware.

Assigning Package Pins

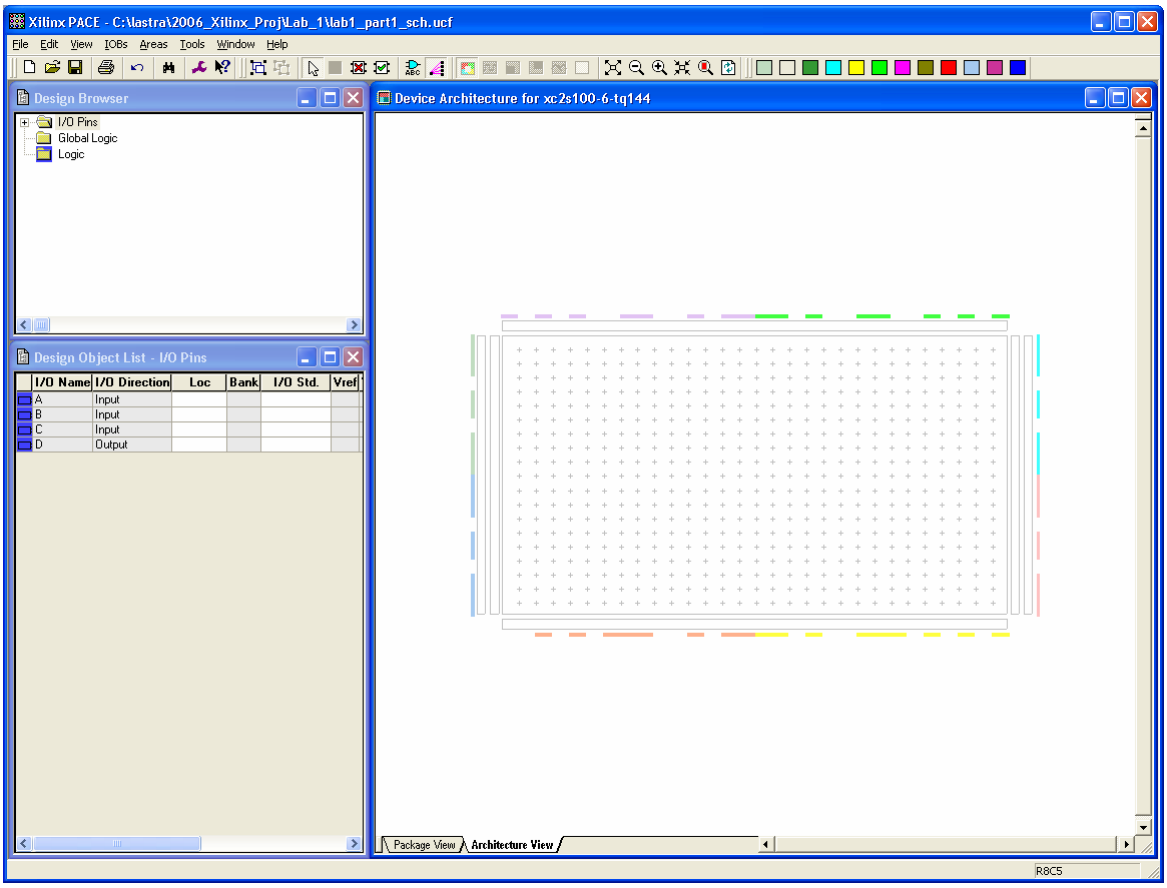
Now, go back to the Project Navigator. Your schematic file, lab1_part1_sch, should be highlighted.



Choose *Assign Package Pins* in the *User Constraints* group. This will pop up the following window.

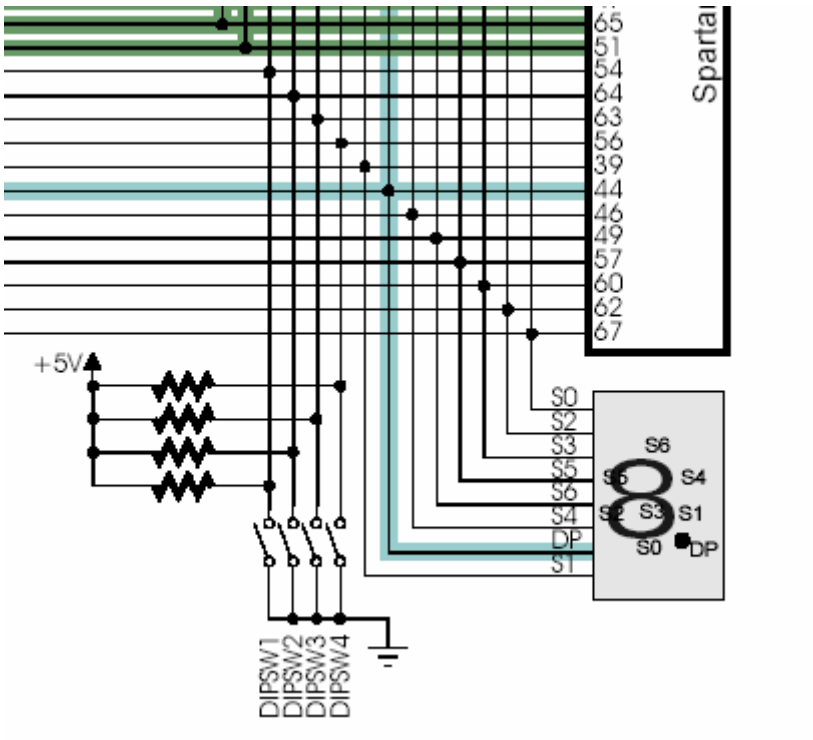


Say *yes* to add a constraints file to your design. You could also have added a constraints file using *Create New Source.* The constraints editor will open in another window.



You'll see the D, C, B, and A ports. We need to assign them to pins on the chip.

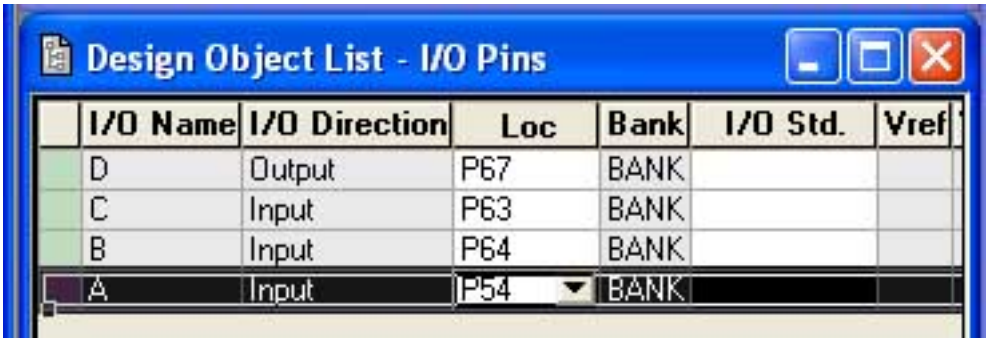
To figure out how to connect the I/O pins, refer to the schematic of the board (available in the XSA manual at http://www.xess.com/manuals/xsa-manual-v1_2.pdf or a paper copy in the lab). Connect the input pins to the DIP switches on the board and the output to the bottom bar (S0) of the seven-segment display. Below is part of a diagram from page 28 of the manual.



Assign as follows

A Pin 54
B Pin 64
C Pin 63
D Pin 67

Enter those as shown below.

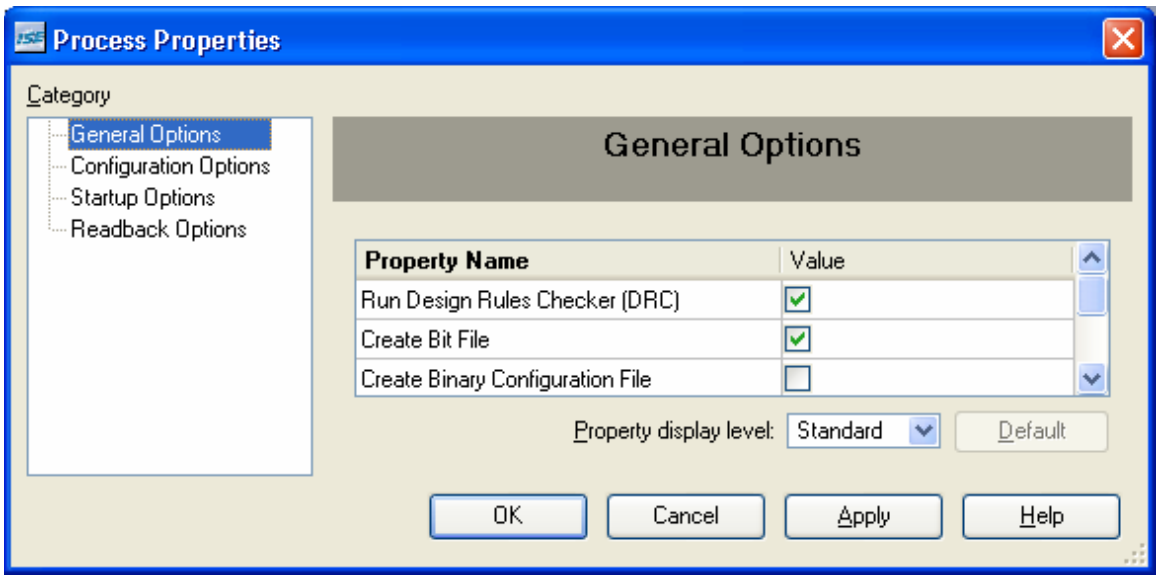


Save the pin assignments entered into the PACE tool, and exit that window. The PACE tool just manipulates a text file, so another way to enter pin assignments is to edit the text file directly using the *Edit Constraints (text)* command. *Optional:* If you double-click on Edit Constrains, you'll see a file with these contents.

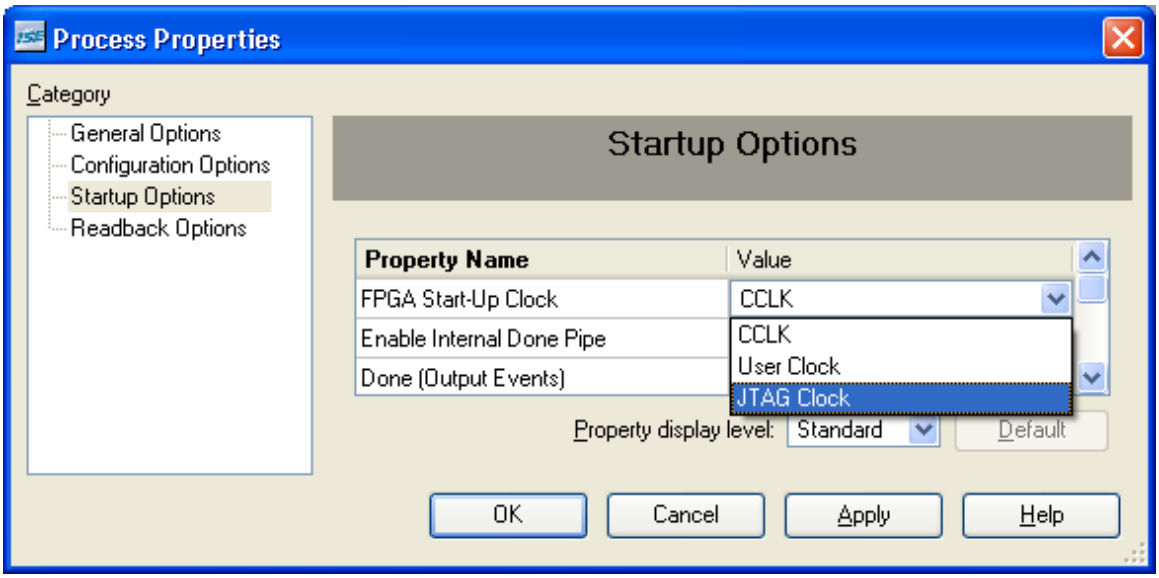
```
#PACE: Start of Constraints generated by PACE

#PACE: Start of PACE I/O Pin Assignments
NET "A"  LOC = "P54"  ;
NET "B"  LOC = "P64"  ;
NET "C"  LOC = "P63"  ;
NET "D"  LOC = "P67"  ;

#PACE: Start of PACE Area Constraints

#PACE: Start of PACE Prohibit Constraints

#PACE: End of Constraints generated by PACE
```

Creating the Programming File

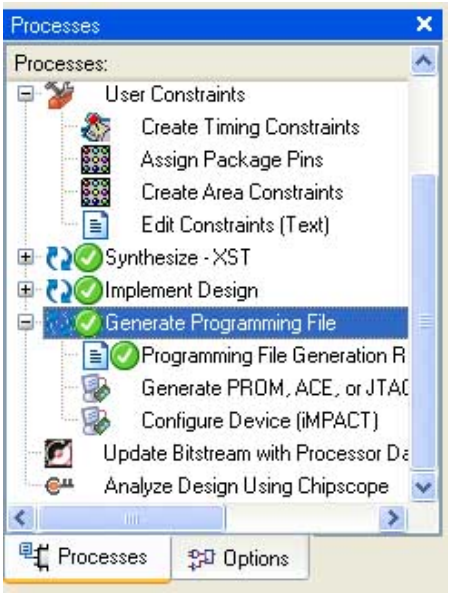
In this step we'll create a file that's downloaded to the FPGA. Go back to Project Navigator and select the schematic file in the Sources window. Then right-click *Generate Programming File* in the *Processes* tab. This window pops up.



Choose the *Startup Options* tab and choose *JTAG Clock* for the *FPGA Start-Up Clock* property. Click OK.



In the *Processes* window double-click *Generate Programming File* (or right-click and select *Run*). You should see green checkmarks (below).

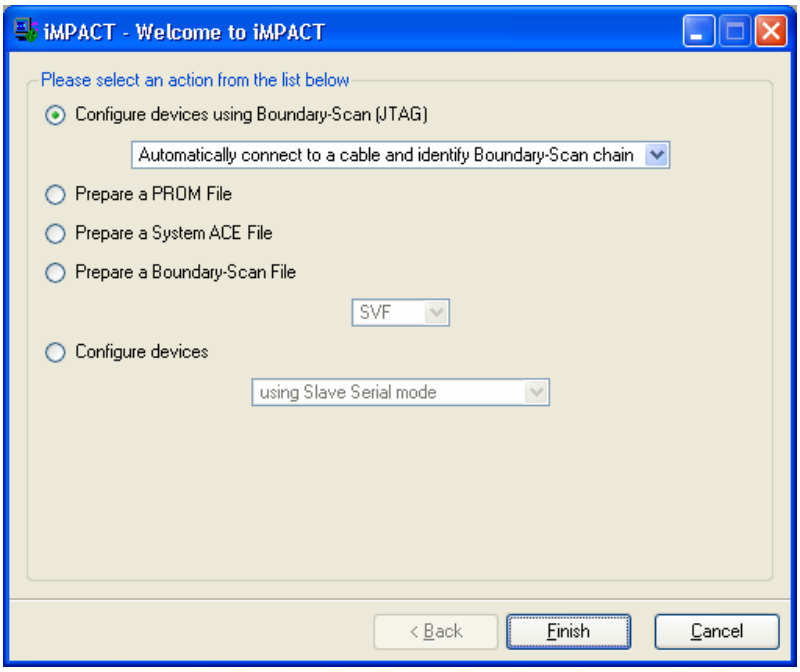


If you see any yellow questions marks, there's a problem that's detailed in the Console window. You may need to figure it out and perhaps fix the problem.

This step generates a file with a *.bit* extension that contains the programming codes for the FPGA chip. In this case it'll be *lab1_part1_sch.bit.* In the next step we'll download that file to the FPGA.

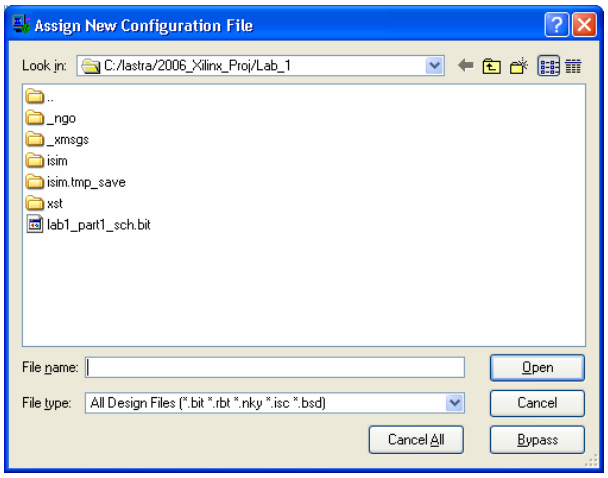
Downloading to the Board

In Project Navigator, select *Configure Device* under Generate Programming File. This will bring up the iMPACT window and the following dialog box.

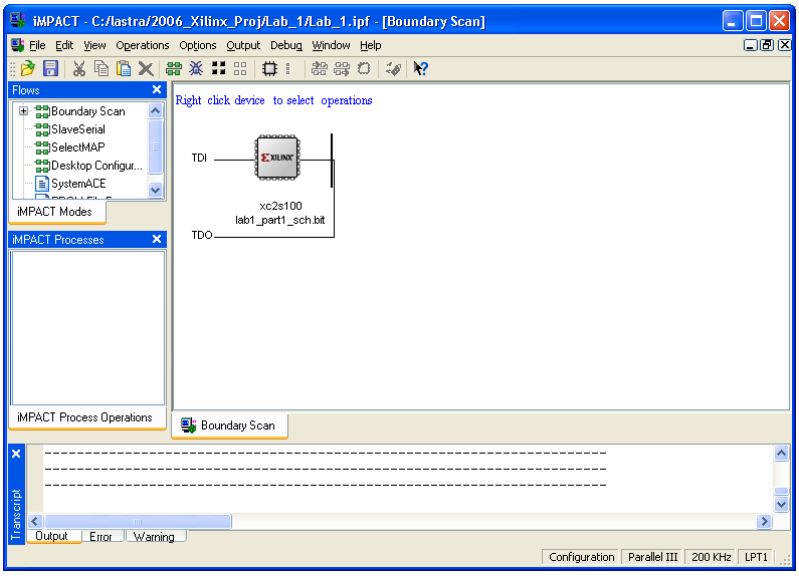


Apply power to the XSA board and make sure it's connected to the parallel port. Click Next to perform a *Boundary Scan*. This should find the FPGA chip. If it doesn't, check to see whether the board has power and the parallel cable is connected. Other possible problems are that the board is misconfigured. See the lab assistant if the board is still not found.

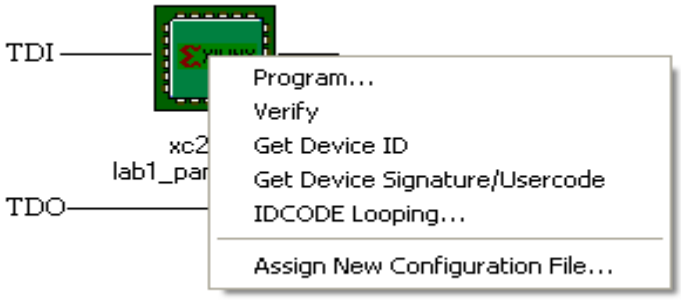
You'll be prompted to choose a configuration file. Select the one that was just generated. *Note: this dialog box will not necessarily show the folder in which you've been working. It may show the last one used to load a configuration file (maybe some other student's).*



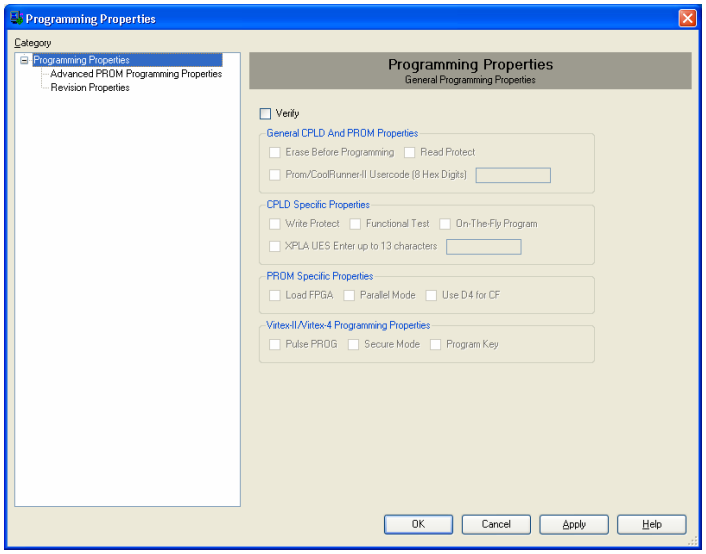
The iMPACT window should now look like this.



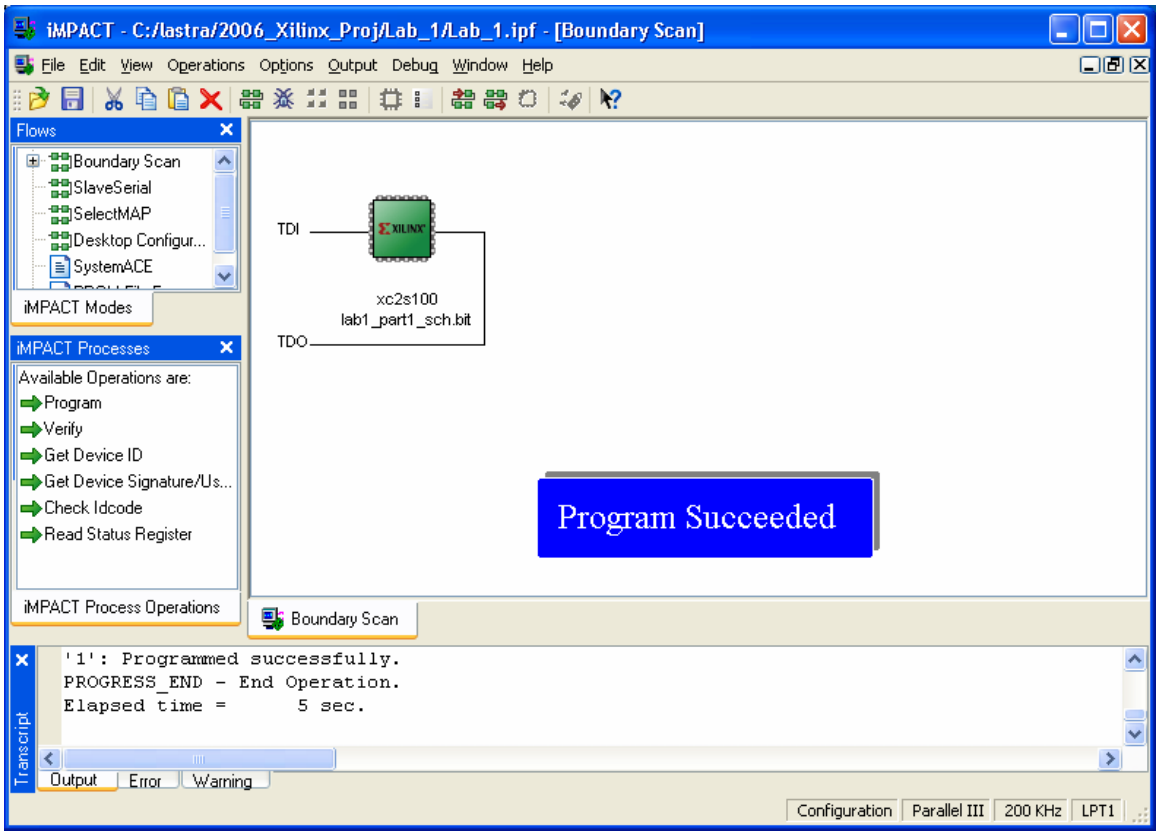
Right-click the Xilinx icon and select *Program*



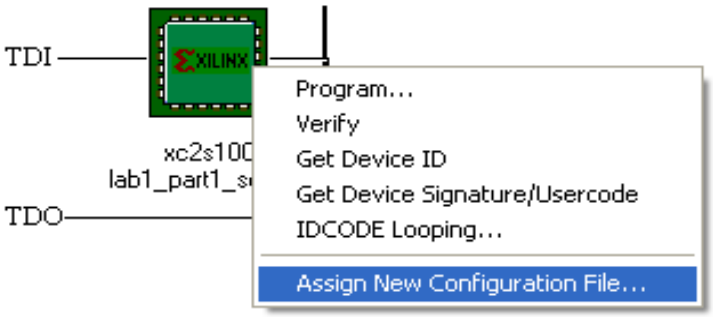
That will bring up a new dialog box (below). Just click OK.



You should get a blue confirmation that programming succeeded.



Careful. Say that you've kept the iMPACT program running but have been editing your design. Even if you change the design and a new *.bit* file has been generated, the iMPACT tool will download the old one. A way to get around this is to right-click the FPGA icon and select *Assign New Configuration File* and choose the *.bit* file again.



Testing

Flip the 1,2,3 switches and see how the output changes. Is the output what you expect it to be? Remember the switch is set up in such a way that when it is 1 it is connected to ground (low), and when it is 0 it is connected to high. Check it against table below:

A	B	C	$D = AB + \overline{C}$
1	1	1	1
1	1	0	1
1	0	1	0
1	0	0	1
0	1	1	0
0	1	0	1
0	0	1	0
0	0	0	1

Once you are ready to move on, please move all the switches back to the off position.

PART 2.

Now let's repeat the same thing but with a Verilog description.

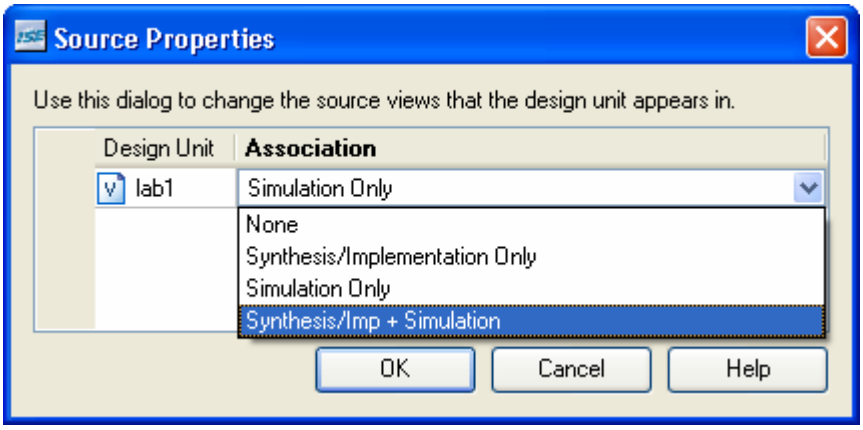
1. Create a new project.
2. This time select HDL as the top level source type.
3. Add a new source of type *Verilog module*, and call it lab1.
4. Skip the part where you can add inputs. It's easier to specify them in the source.
5. Click finish.

Now you have a new project with a Verilog file. The editor will open and show you a template for a Verilog file like this.

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
//////////////////////////
// Company:
// Engineer:
//
// Create Date:    22:21:54 01/18/2006
// Design Name:
// Module Name:    lab1
// Project Name:
// Target Devices:
// Tool versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
//////////////////////////
module lab1();


endmodule
```

Type your source code in, beginning with inputs and outputs in the declaration.

NOTE: In the current version of ise, I ran into a problem where the new Verilog file was only visible in the *Behavioral Simulation* view. I had to right click on the file name, select *Properties* and change the association to Synthesis and Simulation. See below.



PART 3.

Repeat the same steps as above for the following function:

$$E = (A \oplus B)\overline{(AC + D)}$$

Make a Word file that includes your schematic (copy using ALT-PrtSc), table of all the possible values for the inputs and results that you got when you downloaded your design to the board. You'll need one more input, the fourth switch (on pin 56). Demonstrate your results (if one of us is there) and email the file.

Things you learned:

- How to create a schematic file.
- Basic simulation.
- How to assign package pins.
- How to download the design to the board.
- Simple Verilog.

Authors: Anselmo Lastra and Aleksandra Krstic
Original: 09/23/02, updated 1/06