

PART I: Theory

WHAT IS AN FPGA?

Field-Programmable Gate Arrays

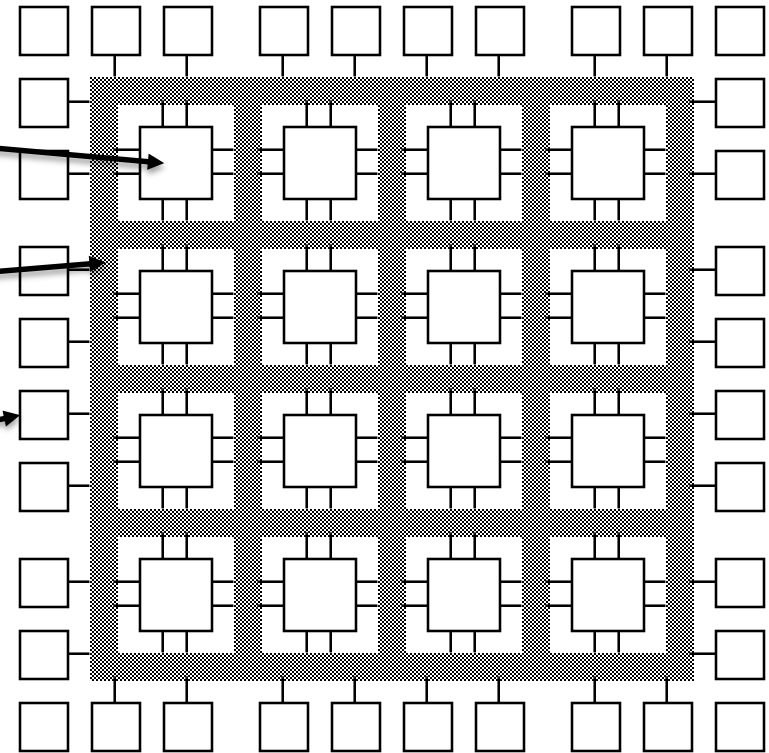
- A field-programmable gate array (FPGA) is an integrated circuit designed to be configured by a customer or a designer after manufacturing.
- In short, programmable circuit.



Image Courtesy: [https://upload.wikimedia.org/wikipedia/commons/4/4e/Xilinx_Spartan-3E_\(XC3S500E\).jpg](https://upload.wikimedia.org/wikipedia/commons/4/4e/Xilinx_Spartan-3E_(XC3S500E).jpg)

Field-Programmable Gate Arrays

- Logic blocks
 - to implement combinational and sequential logic
- Interconnect
 - wires to connect inputs and outputs to logic blocks
- I/O blocks
 - special logic blocks at periphery of device for external connections



Slide Courtesy: <http://courses.cs.washington.edu/courses/cse467/00wi/lectures/ppt/FPGAIntro/FPGAIntro.ppt>.

Applications of FPGA

- Aerospace and Defense
 - Communications, Missiles, Mars Rovers
- ASIC Prototyping
- Consumer Electronics
 - Digital displays, digital camera
- Data Centers
 - Servers, Routers
- High Performance Computing
- ...

Plan

1. The tool: Verilog HDL
2. Some theory: FPGA Design and Implementation
3. Some practice: Lab 0

FPGA DESIGN AND IMPLEMENTATION FUNDAMENTALS

FPGA Design Fundamentals

- Step 1 – Design
 - Know what it is that you want to implement, e.g. an adding machine, or a traffic controller
 - Module-level diagrams and interactions between modules
 - Control logic and state machine drawings
 - Understand how your FPGA design will interact with the physical world, e.g. Ethernet, VGA, LCD.
 - Plan **everything** out before writing a single line of code! Explain the plan to someone else.

FPGA Design Fundamentals

- Step 2 – Implementation
 - Translate your plan to source code!
 - Express each module in HDL source code
 - Connect the modules in hierarchical order like building LEGO blocks. You should end up with a single top-level file.
 - Use any text editor (even Notepad or Wordpad will do) as long as the file name ends with “.v”

FPGA Design Fundamentals

- Step 3 – Simulation
 - Simulation is the single most important debugging tool you will ever use in a FPGA design
 - You will have access to real-time debugging tools (e.g. chipScope) but simulation is far easier to find and fix the bugs.

FPGA Design Fundamentals

- Step 4 – Logic Synthesis
 - Once the bugs are out, a logic synthesis tool analyzes the design and generates a netlist with common cells available to the FPGA target
 - The netlist should be functionally equivalent to the original source code.
 - We will use ISE's XST to synthesize the project

FPGA Design Fundamentals

- Step 5 – Technology Mapping
 - The synthesized netlist is mapped to the device-specific libraries.
 - The result is another netlist that's closer to the final target device.
 - On ISE this is performed by NGDBUILD

FPGA Design Fundamentals

- Step 6 – Cell Placement
 - The cells instantiated by final netlist are placed in the FPGA layout, i.e. each cell is assigned a physical location on the target device.
 - Can be a time-consuming process depending on the size of the design and complexity of timing and physical constraints.
 - On ISE this process is done by the program MAP (i.e. map to physical location)

FPGA Design Fundamentals

- Step 7 – Route
 - Often referred to as “Place-and-Route” in combination with cell placement.
 - In this process, the placement tool determines how to connect (“route”) the cells in the device to match the netlist
 - Can be a time-consuming process depending on the size of the design and complexity of timing and physical constraints.
 - Done by program PAR on ISE.

FPGA Design Fundamentals

- Step 8 – Bitstream Generation
 - A placed and routed design can be used to produce a programming file to program the FPGA.
 - The programming file is called a “bitstream.” It contains everything there is about how to configure the cells and connecting them.
 - Done by program BITGEN on ISE.
 - Now you have a “compiled” FPGA design.

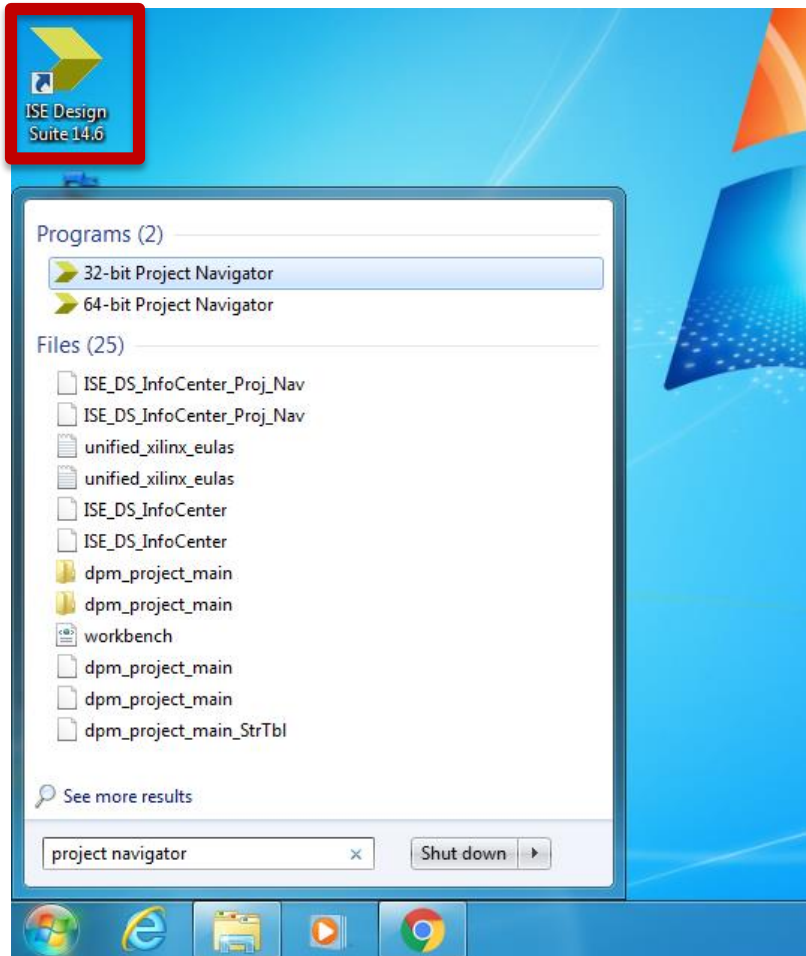
Tools of Trade

- Text editor of choice
- Simulator
 - ISE Webpack provides ISIM
 - Alternatively use free Modelsim PE
- Synthesis
 - ISE Webpack provides XST
 - Alternatively use Synplify Pro (evaluation version)
- Map, Place-and-Route
 - ISE Webpack

PART II: Practice

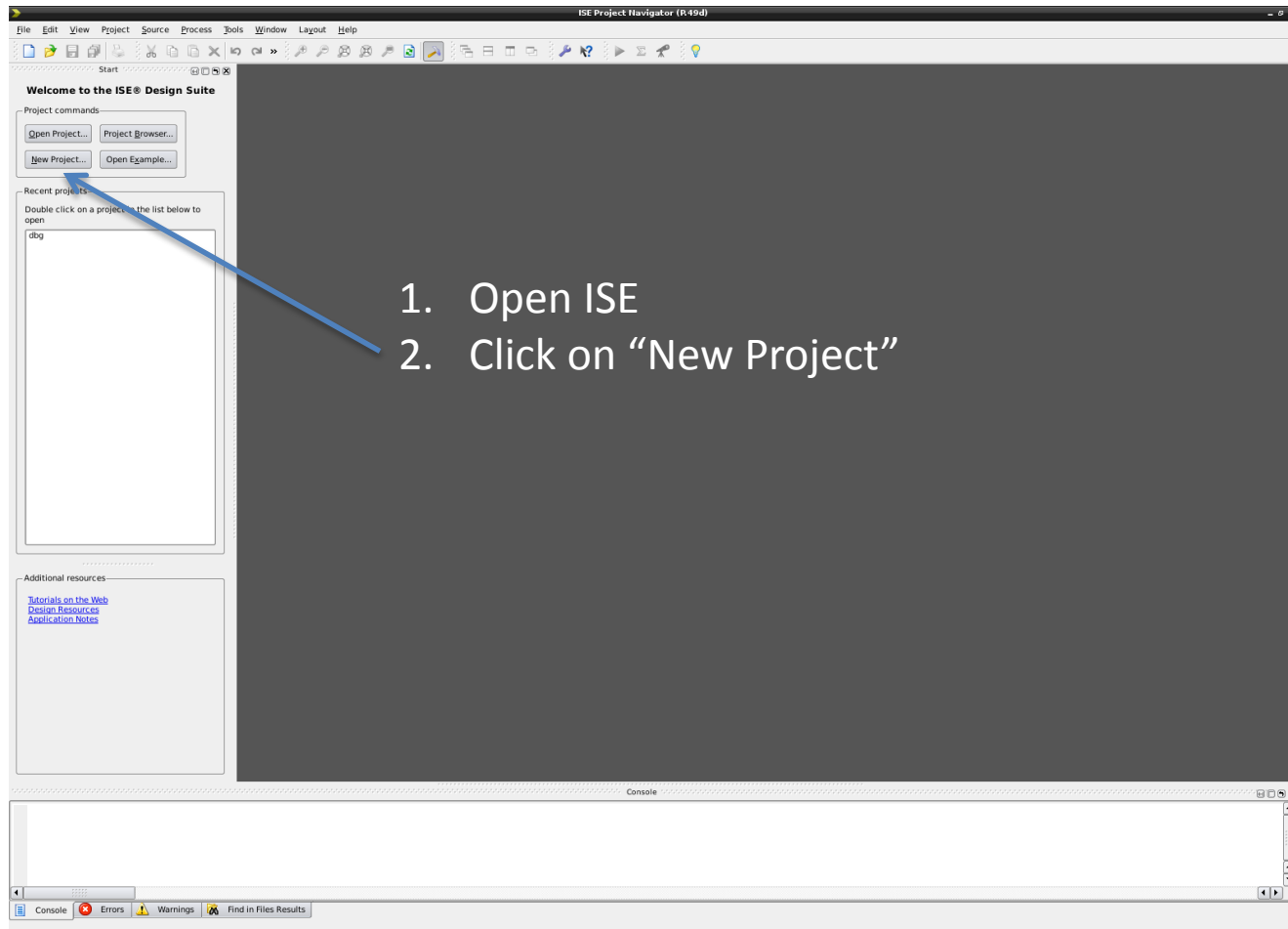
EXAMPLE PROJECT IMPLEMENTATION

Open the Xilinx ISE



- Click the ISE Design Suite 14.6 icon
- Or search for “project navigator” in the start menu

Create an ISE Project



New Project Dialogue

Create New Project
Specify project location and type.

Enter a name, locations, and comment for the project:

Name:

Location:

Working Directory:

Description:

Select the type of top-level source for the project:

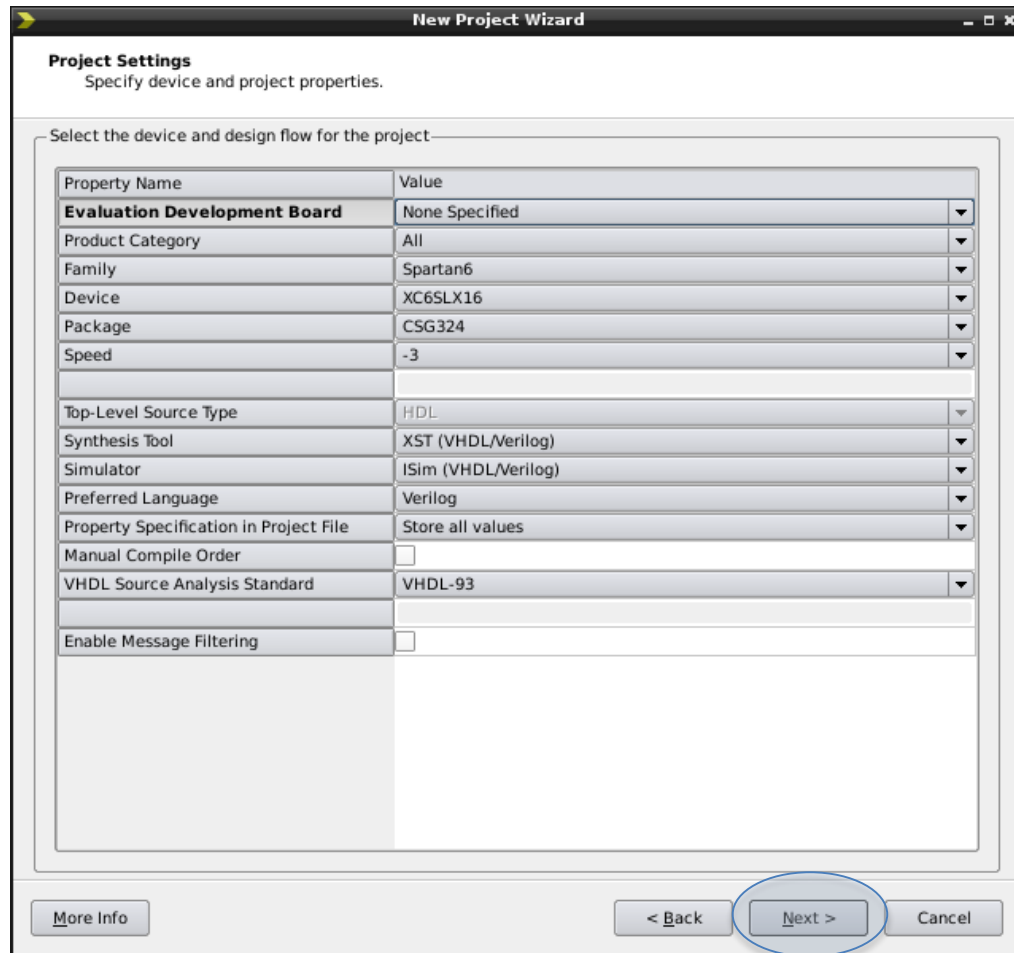
Top-level source type:

• No spaces in the path!
• Choose “HDL” project

More Info Next > Cancel

Device Properties

Make sure the fields match what you see here

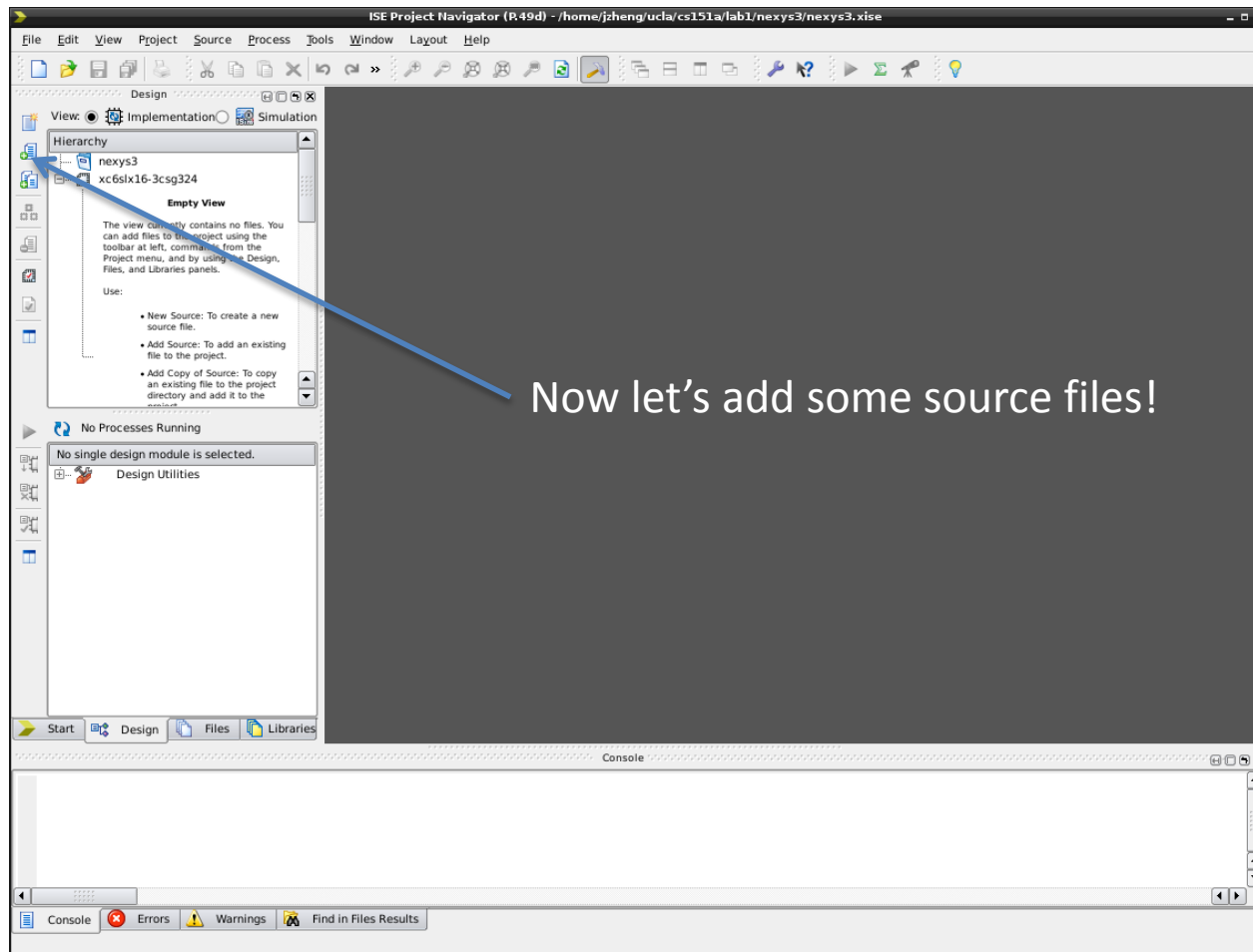


The image shows a screenshot of the 'New Project Wizard' dialog box, specifically the 'Project Settings' tab. The dialog is titled 'New Project Wizard' and has a subtitle 'Specify device and project properties.' Below the subtitle, there is a section titled 'Select the device and design flow for the project'. This section contains a table with two columns: 'Property Name' and 'Value'. The table lists various project settings, including device selection, synthesis tool, and language. The 'Next >' button at the bottom right is circled in blue.

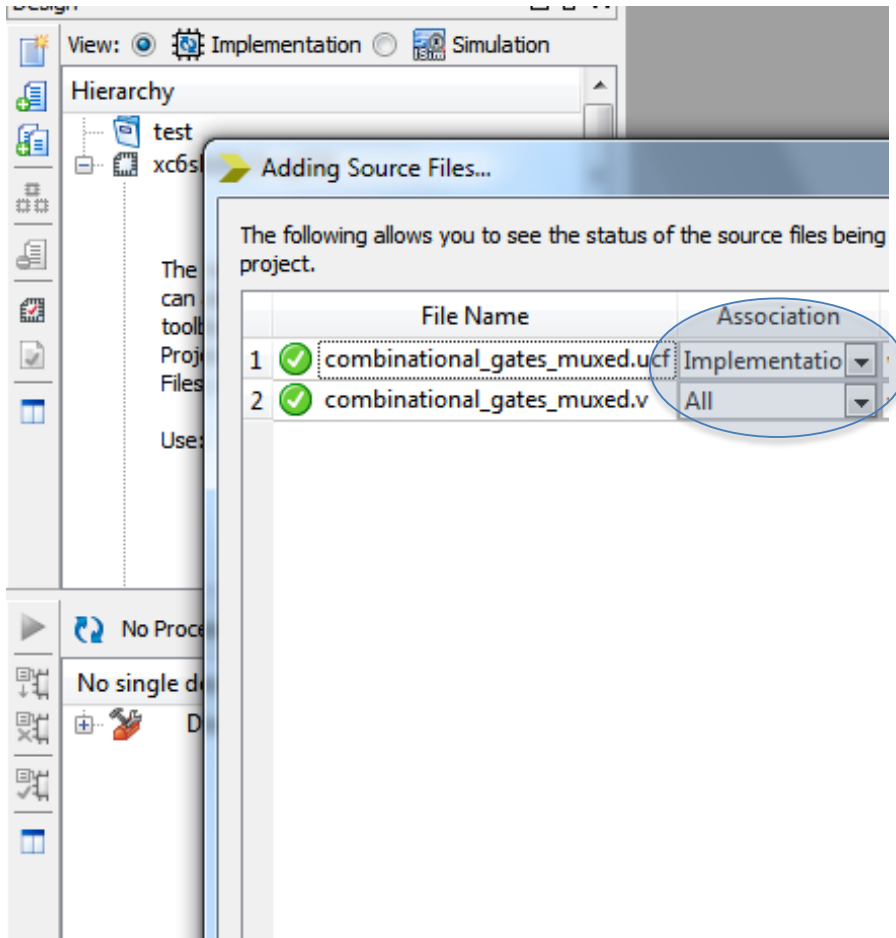
Property Name	Value
Evaluation Development Board	None Specified
Product Category	All
Family	Spartan6
Device	XC6SLX16
Package	CSG324
Speed	-3
Top-Level Source Type	HDL
Synthesis Tool	XST (VHDL/Verilog)
Simulator	ISim (VHDL/Verilog)
Preferred Language	Verilog
Property Specification in Project File	Store all values
Manual Compile Order	<input type="checkbox"/>
VHDL Source Analysis Standard	VHDL-93
Enable Message Filtering	<input type="checkbox"/>

More Info < Back **Next >** Cancel

Project Created

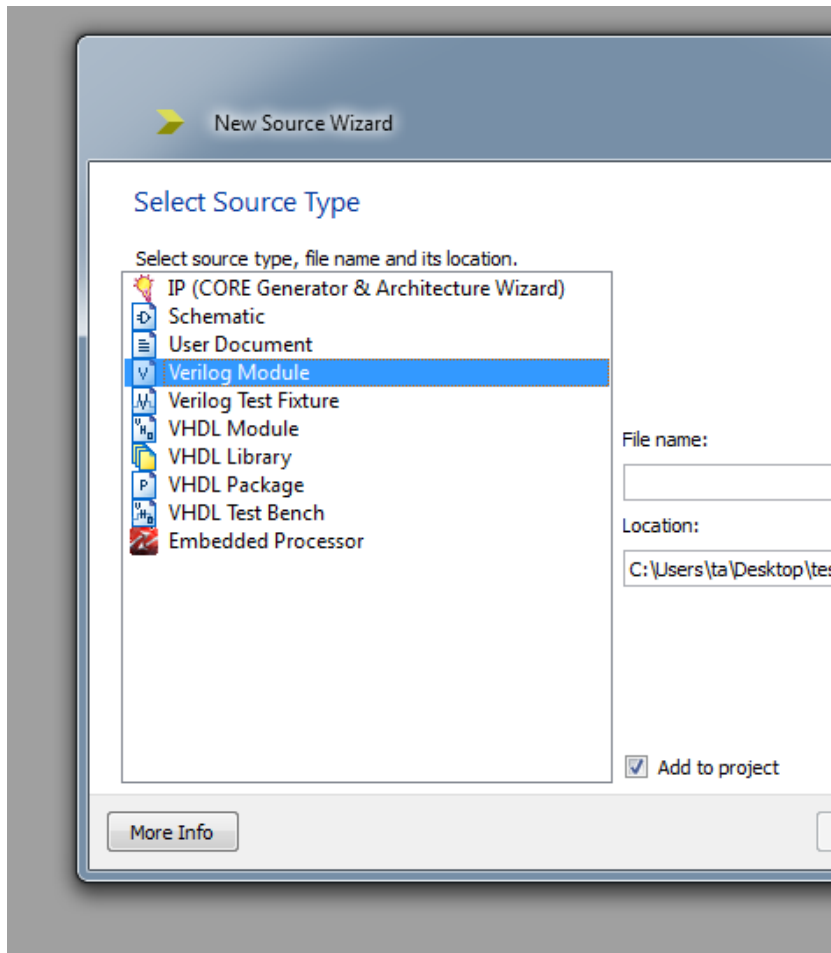


Add Source Files



1. Click on “Add sources”
2. Select
combinational_gates_m
uxed.v,
combinational_gates_m
uxed.ucf
3. Make sure the file
association is correct

New Source Files



- Alternatively if you want to write your own code, click on “New Source”
- Select “Verilog Module” then follow the instructions

Source Files

- .v files are Verilog source code
- .ucf files are *User Constraint Files*

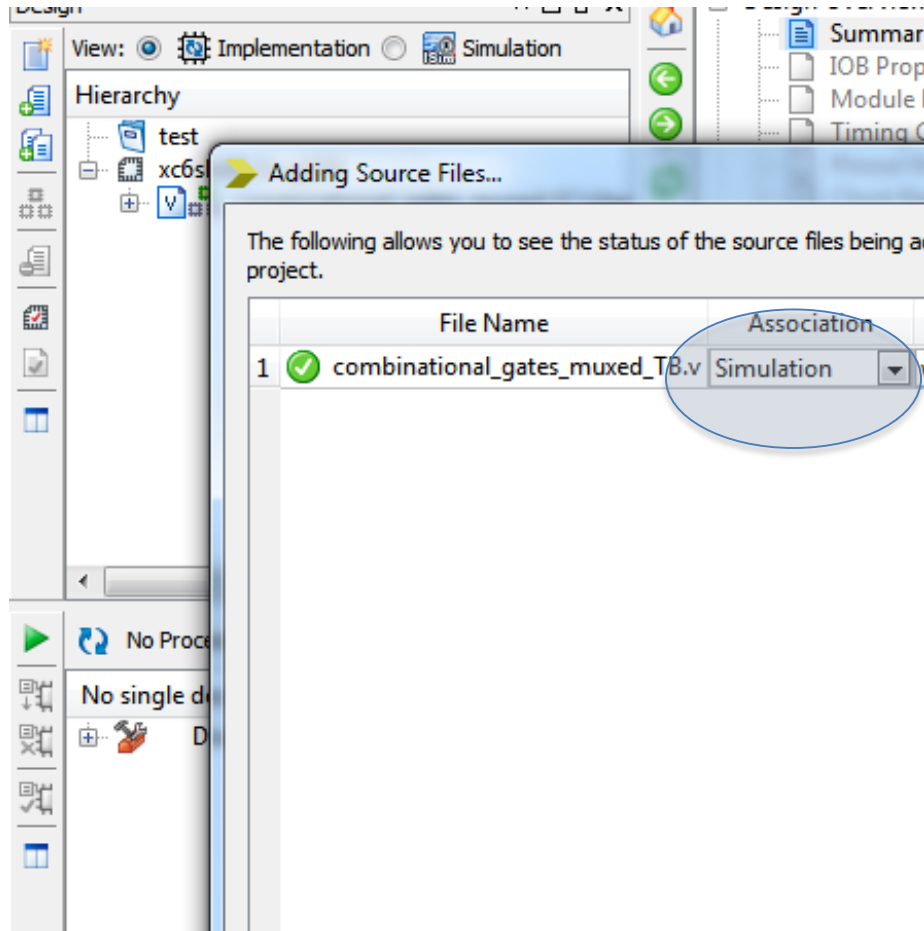
UCF lists all the available pin mappings in the FPGA in the following format:

```
Net "your_signal_name<bit_index>" LOC = XX | IOSTANDARD =  
    LVCMOS33; # More details about the pin
```

For example:

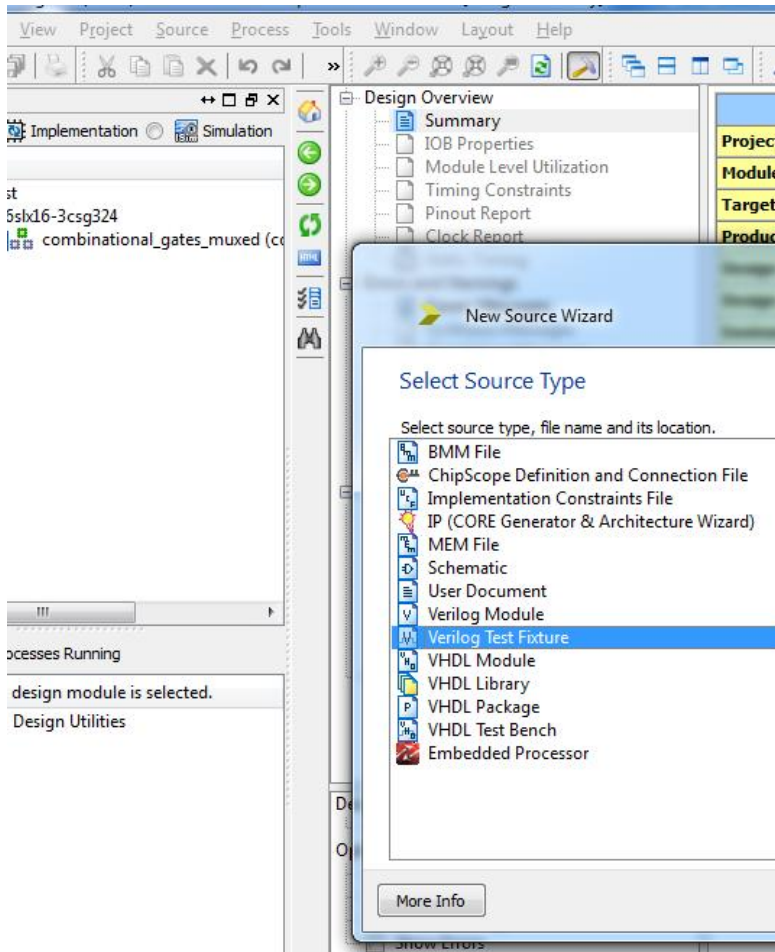
```
Net "sw<0>" LOC = T10 | IOSTANDARD = LVCMOS33; #Bank = 2,  
    pin name = IO_L29N_GCLK2, Sch name = SW0
```

Add Testbench Code



1. Click on “Add sources” again
2. Select the combinational_gates_muxed_TB.v
3. Make sure the file association is correct

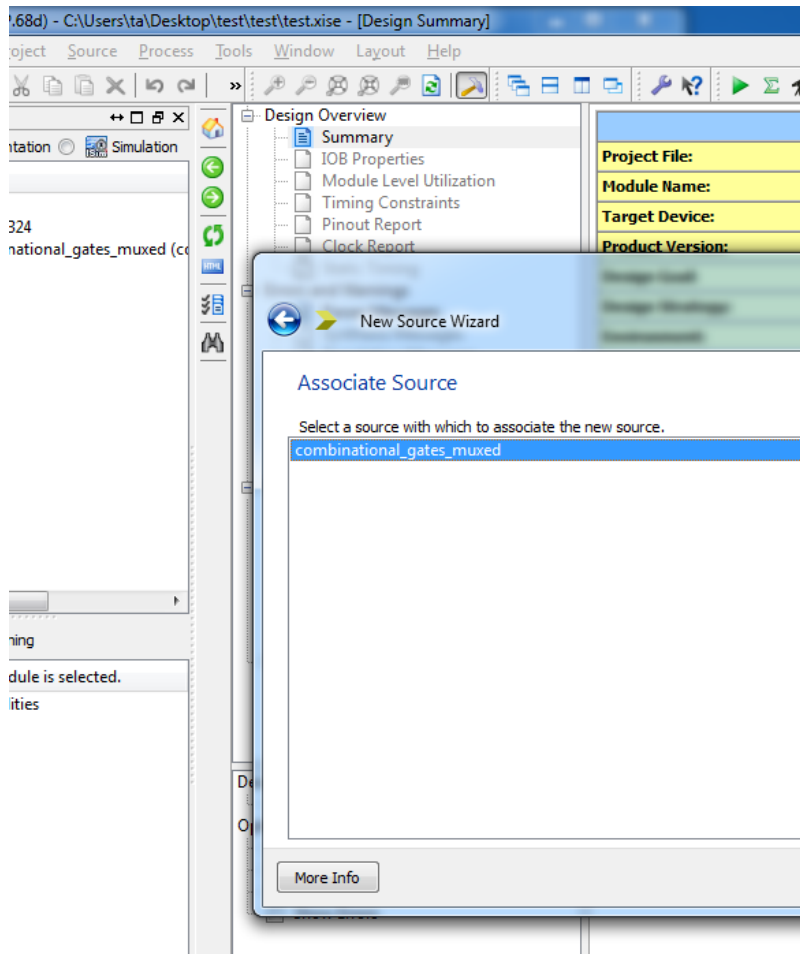
Create Testbench Code



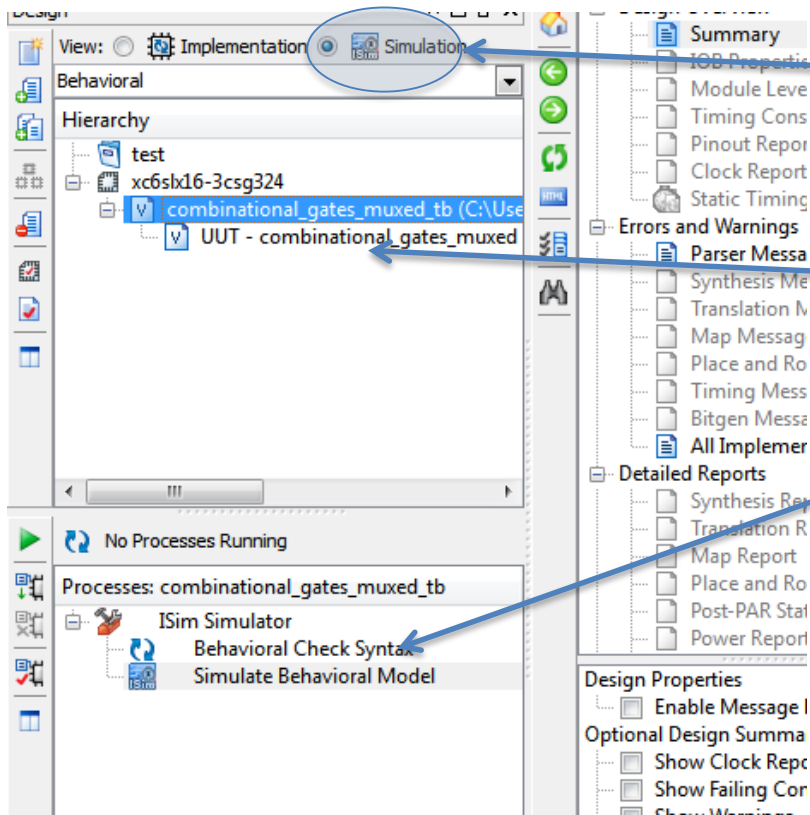
- Alternatively if you want to write your own testbench, click on “New Source”
- Select “Verilog Test Fixture”

Create Testbench Code

- Select the module to test and then follow the instructions



Almost Ready for Simulation!



1. Switch to simulation view
2. Select ...TB.v from Hierarchy view
3. Right click on “Simulate Behavioral Model” in process view
4. Click on “Process Properties”

ISIM Process Properties

Process Properties - ISim Properties

Switch Name	Property Name	Value
	Use Custom Simulation Command File	<input type="checkbox"/>
	Custom Simulation Command File	...
	Run for Specified Time	<input checked="" type="checkbox"/>
	Simulation Run Time	1000 ns
	Waveform Database Filename	/home/jzheng/ucla/cs151a/lab1/nexys3/tb_isim_beh.wdb
	Use Custom Waveform Configuration File	<input type="checkbox"/>
	Custom Waveform Configuration File	...
	Specify Top Level Instance Names	work.tb
	Load glbl	<input checked="" type="checkbox"/>

Uncheck "Run for Specified Time"
Our simulation is terminated by making a "\$finish" system task call

Property display level: Standard ☐ Display switch names Default

OK Cancel Apply Help

Launch ISIM

- Right click on “Simulate Behavioral Model” again, this time choose “run all”
- ISIM will be launched
- ISIM is the simulation environment where you can dynamically debug the circuit, much like a software debugger
- Your main focus should be on the console window and the waveform window

ISIM Main Window

Hierarchical
View

Signal
View

Waveform

The screenshot displays the ISIM Main Window with three primary views: Hierarchical View, Signal View, and Waveform View. The Hierarchical View on the left shows the project structure, including the testbench file 'combinational_gates_muxed_TB.v'. The Signal View in the center shows the simulation objects for 'Initial_49_3', with a table listing 'led_T' and 'sw_T[4:0]'. The Waveform View on the right shows the Verilog code for the testbench, including the instantiation of the 'combinational_gates_muxed' module and the initialization of the 'sw_T' register. The Console at the bottom shows the simulation status, including a warning about the software subscription period and the simulation stopping at 160 ns.

Object Name	Value
led_T	0
sw_T[4:0]	11111

```
18 reg [4:0] sw_T;
19
20 // Outputs in the module to be tested will be port mapped to wire variables
21 wire led_T;
22
23 // Instantiation of the design module to be verified by the testbench
24 // Use named portmapping to map inputs to register variables and outputs to
25 // wires
26 combinational_gates_muxed UUT (.sw(sw_T),
27                                .led(led_T));
28
29 // Used for saving Value Change Dump (.vcd) file that records the waveforms of
30 // the simulation. Not needed while using Xilinx ISIM simulator.
31 initial
32 begin
33     $dumpfile("combinational_gates_muxed.vcd");
34     $dumpvars(2, combinational_gates_muxed_TB UUT);
35 end
36
37 // IMPORTANT: Initialize all inputs. Otherwise the default value of register
38 // will be don't care (x).
39 initial
40 begin
41     sw_T = 5'h0;
42 end
43
44 // Use an always block to generate all the test cases
45 always
46     #5 sw_T = sw_T + 1'b1;
47
48 // Code to terminate simulation after all the test cases have been covered.
49 initial
50     #160 $finish; // After 160 timeunits, terminate simulation.
51
52 endmodule
```

Console

ISim P.68d (signature 0x7708f090)

WARNING:Security:42 - Your software subscription period has lapsed. Your current version of Xilinx tools will continue to function, but you no longer qualify for Xilinx software updates or new releases.

This is a Full version of ISim.
Time resolution is 1 ps
Simulator is doing circuit initialization process.
Finished circuit initialization process.
Stopped at time : 160 ns : [File "C:/Users/152/Desktop/test/test/combinational_gates_muxed_TB.v" Line 50](#)
ISim>

Console

Post Simulation Examination

The screenshot shows a digital logic simulation tool interface. The 'Simulation Objects for UUT' panel lists various objects and their values. A blue arrow points from the 'led' object in the list to the waveform view, indicating the process of adding signals to the waveform.

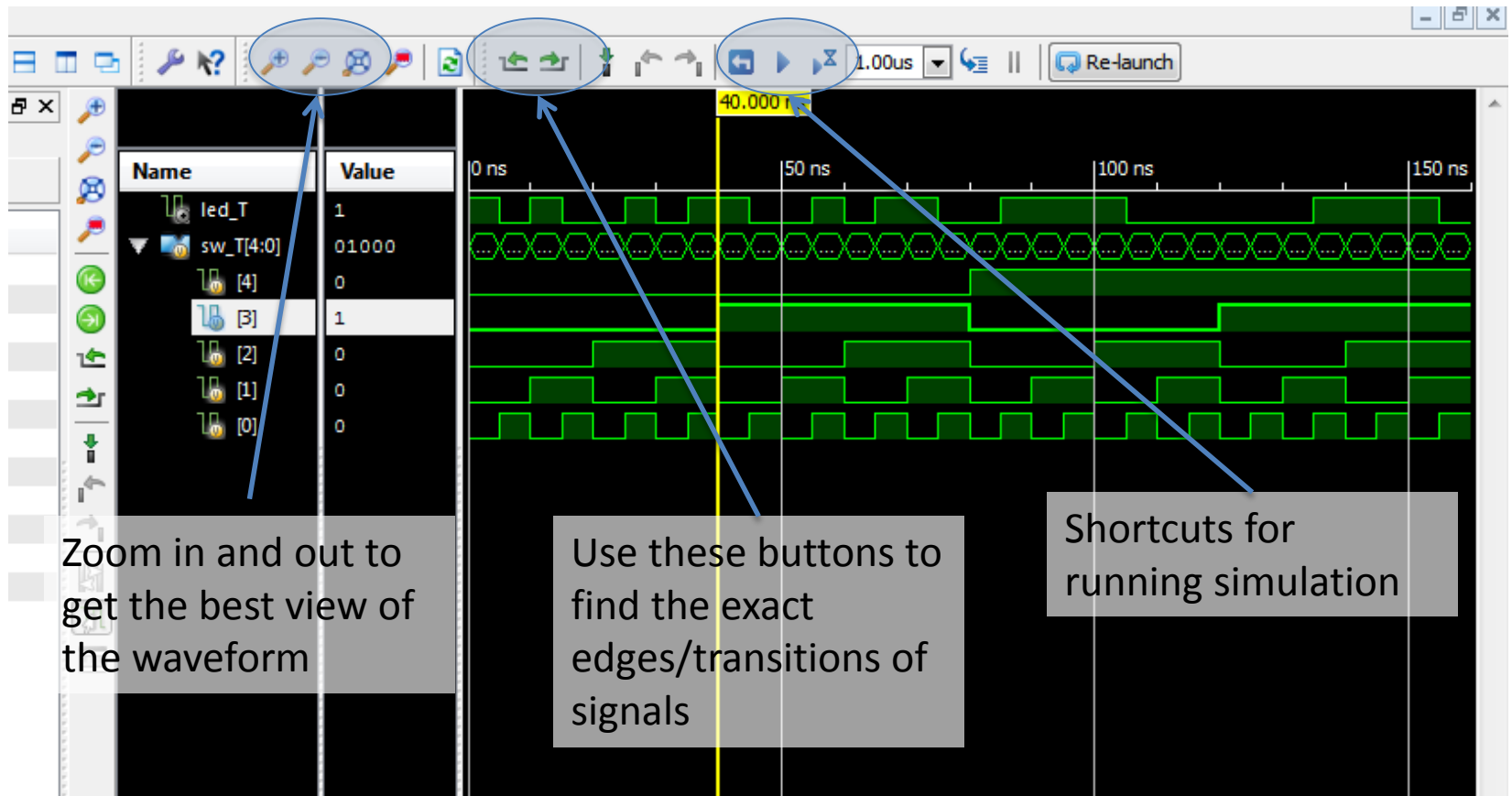
Object Name	Value
sw[4:0]	11111
nand_out	0
and_out	1
nor_out	0
or_out	1
xor_out	0
xnor_out	1
buff_out	1
not_out	0
Muxn[7:0]	01010110
SelectIn[2:0]	111
led	0

To debug, you can add any signals to the waveform view by right click and select add

Name	Value
led_T	0
sw_T[4:0]	11111

Waveform view showing signals at 159,997 ps and 159,998 ps. The signal 'sw_T[4:0]' is highlighted in green.

Post Simulation Examination



Ready for Synthesis

Switch back to implementation view if you haven't already

Double click on synthesis to start. You can also view the auto generated schematics here

combinational_gates_muxed Project Status

Project File:	test.xise	Parser Errors:
Module Name:	combinational_gates_muxed	Implementation State:
Target Device:	xc6slx16-3csg324	• Errors:
Product Version:	ISE 14.6	• Warnings:
Design Goal:	Balanced	• Routing Results:
Design Strategy:	Xilinx Default (unlocked)	• Timing Constrai
Environment:		• Final Timing Sco

Detailed Reports

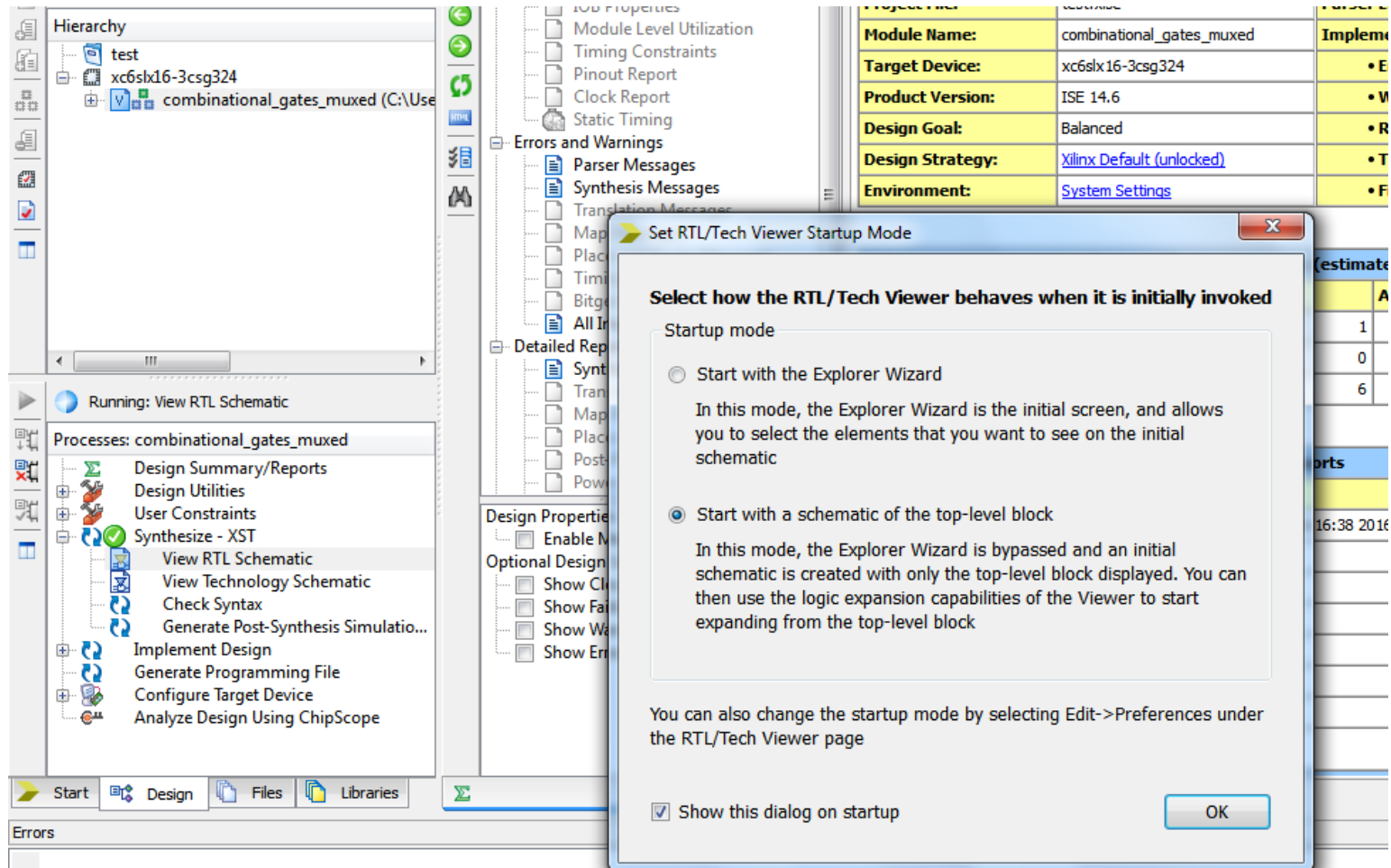
Report Name	Status	Generated	Errors	Warn
Synthesis Report				
Translation Report				
Map Report				
Place and Route Report				
Power Report				
Post-PAR Static Timing Report				
Bitgen Report				

Secondary Reports

Report Name	Status	Generated
-------------	--------	-----------

Date Generated: 09/28/2016 - 16:10:44

View Schematics



Place-n-Route and Bitstream

The screenshot shows the Xilinx ISE software interface during the 'Implement Design' step. The 'Design Overview' pane on the right lists various reports, with 'Place and Route Report' selected. The 'Design Properties' pane below it shows 'Generate Programming File' checked. The 'Processes' pane on the left shows the 'Implement Design' button highlighted. A blue arrow points to this button. The 'Errors' pane at the bottom is empty.

combinational_gates_muxed

Project File:	test.xise
Module Name:	combinational_gates_muxed
Target Device:	xc6slx16-3csg324
Product Version:	ISE 14.6
Design Goal:	Balanced
Design Strategy:	Xilinx Default (unlocked)
Environment:	System Settings

Device Utilization Summary

Logic Utilization	Used
Number of Slice LUTs	
Number of fully used LUT-FF pairs	
Number of bonded IOBs	

Detailed Reports

Report Name	Status	Generated
Synthesis Report	Current	Wed Sep 28
Translation Report		
Map Report		
Place and Route Report		
Power Report		
Post-PAR Static Timing Report		
Bitgen Report		

Design Properties

- ☐ Enable Message Filtering
- Optional Design Summary Contents**
 - ☐ Show Clock Report
 - ☐ Show Failing Constraints
 - ☐ Show Warnings
 - ☐ Show Errors

Processes: combinational_gates_muxed

- Design Summary/Reports
- Design Utilities
- User Constraints
- Synthesize - XST
- Implement Design**
- Generate Programming File
- Configure Target Device
- Analyze Design Using ChipScope

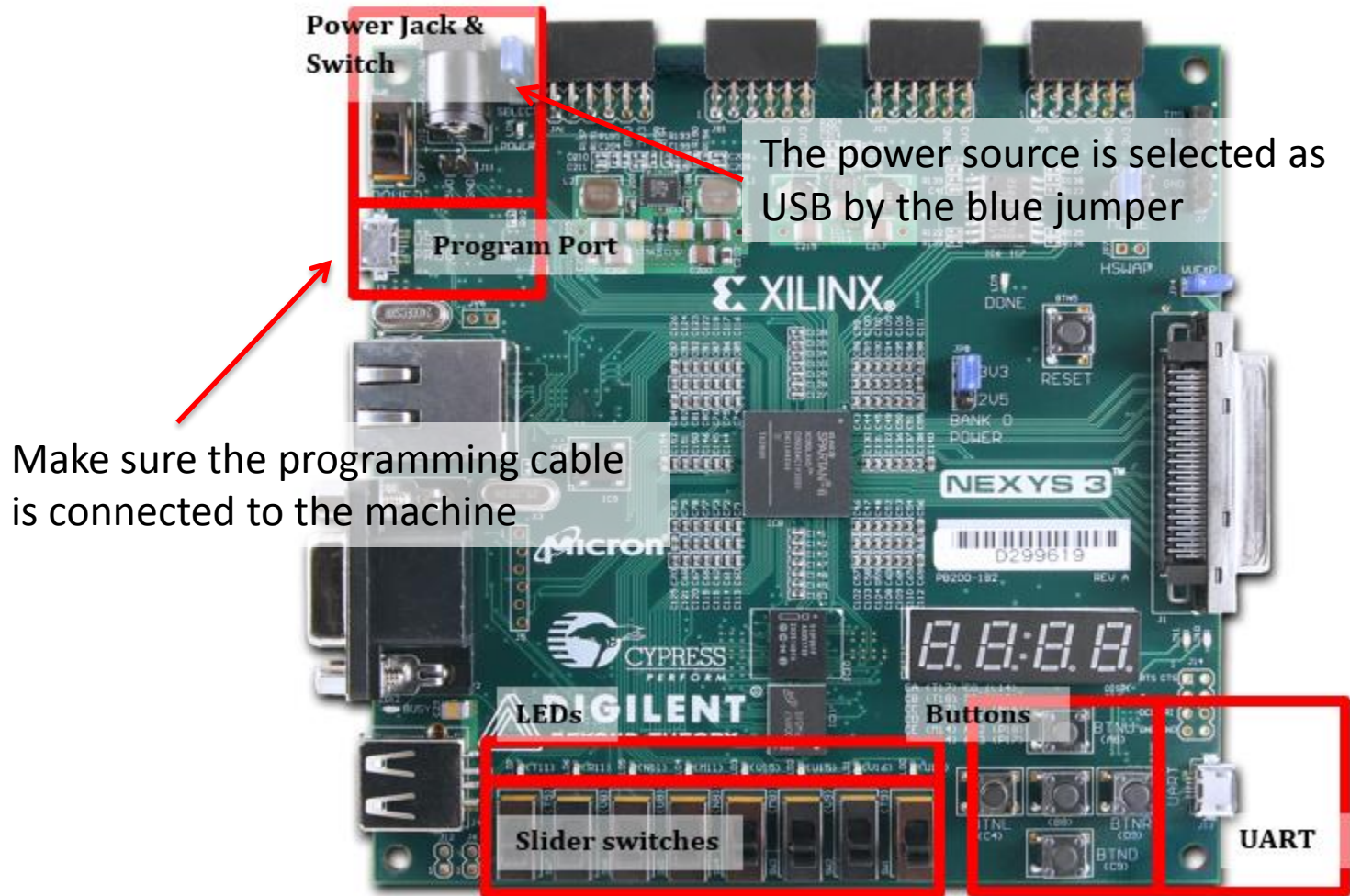
Start Design Files

Double click on "Implement Design" and then "Generate Programming File"

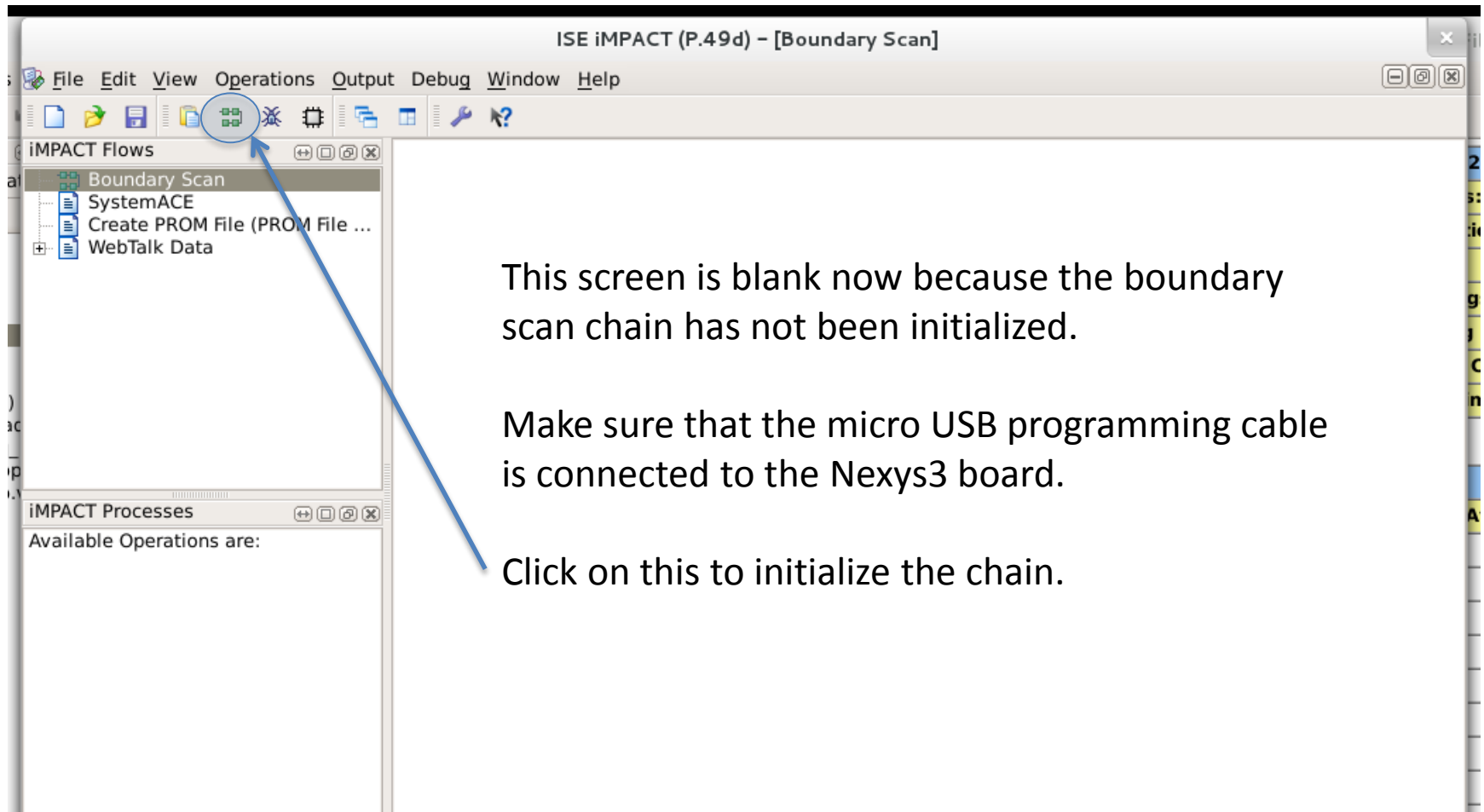
Download Bitstream to FPGA

- By now you should have a `top_module_name.bit(combinationa_gate_muxed.bit)` file generated in the project folder
- You will now program the FPGA using this file.
- Click on “Configure Target Device” to open the Impact program.

Connect the board



ISE Impact



Scan Chain Initialization

ISE iMPACT (P.49d) - [Boundary Scan]

File Edit View Operations Output Debug Window Help

iMPACT Flows

- Boundary Scan
 - SystemACE
 - Create PROM File (PROM File ...)
 - WebTalk Data

iMPACT Processes

Available Operations are:

Right click device to select operations

TDI

TDO

xc6slx16
nexys3.bit

SPI/BPI

A properly initialized chain should contain a single FPGA (xc6slx16).

Assign the bit file to this FPGA

Do not assign any SPI flash programming files.

Click on the FPGA symbol to show the available options.

Program FPGA

ISE iMPACT (P.49d) - [Boundary Scan]

File Edit View Operations Output Debug Window Help

iMPACT Flows

- Boundary Scan
- SystemACE
- Create PROM File (PROM File ...)
- WebTalk Data

iMPACT Processes

Available Operations are:

- Program
- Get Device ID
- Get Device Signature/Usercode
- Read Device Status
- One Step SVF
- One Step XSVF
- Read Device DNA

Double-click on the “Program” action to program the FPGA.

Wait for the programming to finish.

Diagram showing a device (XILINX) connected to TDI and TDO pins. A file path is shown: File : /home/jzheng/ucla/cs151a/lab1/nexys3/nexys3.bit

Play Time

- Did you see the rightmost LED light up?
 - If yes, the board is programmed!
- Can you use the switches to control the LED?
 - Study code to understand how this is done