



JOINT INSTITUTE  
交大密西根学院

Ve270 Introduction to Logic Design

# **Lab 1**

## **Design of a Full Adder**

### **- A Tutorial on Xilinx ISE**

UM-SJTU Joint Institute  
Shanghai Jiao Tong University  
May 2017



## 1. Objective

To become familiar with Xilinx Integrated System Environment (ISE) software for FPGA based digital system design. To draw and simulate simple logic circuits the software tool.

## 2. Background

Xilinx ISE is the world leading electronic circuit design software tool provided by Xilinx Inc. One of the features of Xilinx ISE is that logic circuits can be captured in a schematic, and inputs of the circuits can be set up and then the circuit can be run to examine the timing diagrams of the outputs. In this lab, a logic circuit – Full Adder will be built and simulated using simple logic gates. This lab also serves as a tutorial on the Xilinx ISE software.

## 3. Setup Xilinx ISE

Following the sequence to setup Xilinx ISE working environment:

- Launch Xilinx ISE by click on **Project Navigator**, and the Xilinx ISE work space screen appears.
- Create a new project for your design by **File → New Project**.
  - Select **Project Location** on the hard drive
  - Type in **Project Name**, e.g. *Lab1*. A new folder with the same name will be created in the selected project location. **Note:** Xilinx ISE tool does not like space or dash in the names or the searching path.
  - **Top-Level Source Type: Schematic**
  - Next
- Select FPGA device and design flow for the project as follows:
  - Family: **Spartan3E**
  - Device: **XC3s1200E** (or **XC3s500E**)\*
  - Package: **FG320**
  - Speed: **-4**
  - Top-Level Source Type: **Schematic**
  - Synthesis Tool: **XST (VHDL/Verilog)**
  - Simulator: **ISE Simulator (VHDL/Verilog)**
  - Preferred Language: **VHDL**
  - Next → Next → Next → Finish

**\*Note:** we have two different types of FPGA boards. The major difference is the capacity of the FPGA chip, 1200K or 500K. This information and the corresponding Device ID can be found on the FPGA chip which is mounted in the central area of the board.

- Create new source file (schematic) in the project by **Project → New Source**.
  - Select **Schematic** from the menu on the left hand side.
  - Type a File Name for the file, e.g. *Circuit1*.
  - Leave the “**Add to project**” box selected.
  - Next → Finish.
- Click through the remaining project creation options to create the project and new source file.

#### 4. Drawing Schematics in Xilinx ISE

From previous steps, a blank schematic drawing space is created and the schematic of a circuit can be drawn on it. Figure 1 shows the schematic of a simplified full adder that we will draw on the sheet.

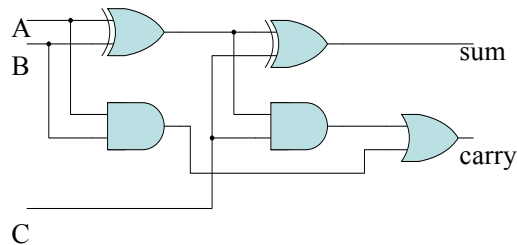


Figure 1. Full Adder Gate-Level Schematic

- Place logic gates on the blank sheet.
  - In the **Sources** panel on the left side, select the **Symbols** pad. All digital circuit parts are grouped in **Categories** and can be found in **Symbols** box. This is shown in Figure 2.

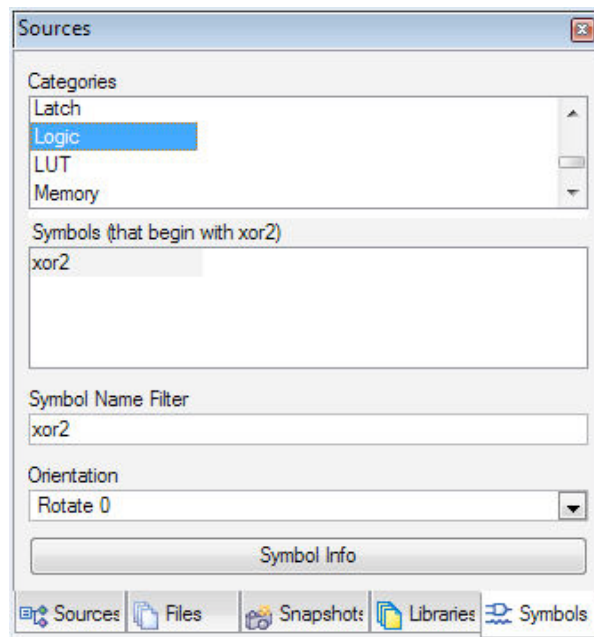


Figure 2. Sources Panel

- Click on the category **Logic**, this will bring up all the logic gates in the **Symbols** box that can be used to build a logic circuit. A symbol indicates the logic function of a gate as well as the number of its inputs. For example, symbol **and2** tells the gate is a 2-input AND gate.
- Click on the **and2** symbol once, then move the mouse over the blank sheet on the right side and click again. This will allow you to place the 2-input AND gate on the sheet. Place the **xor2** and **or2** gates on the sheet as well.
- Connect the logic gates with wires.
  - Click **Add → Wire**, then add wires between the gate terminals.
  - All the inputs and outputs should be extended with wires so that I/O pads can be added later. To extend a wire from a terminal, drag the wire outwards and then double click a blank space. This will terminate the wire.
  - To stop the wiring, right click.

NOTE: notice when each operation (e.g. add gates or add wires) is selected, an icon on the tool bar is also pressed, indicating the operation can also be activated by clicking the corresponding icon on the tool bar, as shown in Figure 3.



Figure 3. Add Wires Icon on the Tool Bar

- Create I/O ports.
  - Label each I/O wire by **Add → Net Name**. In the **Processes** panel and **Options** window on the left side, type a name for the wire in **Name** box, then click on the end of the target wire. The Process panel is as shown in Figure 4.

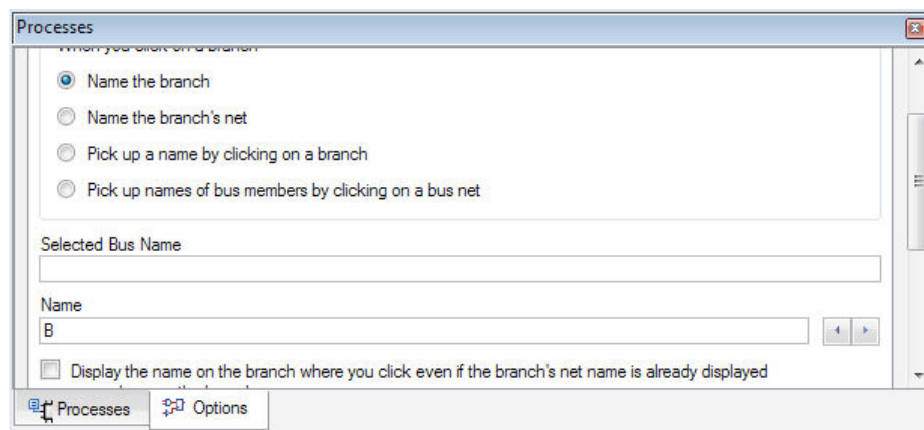


Figure 4. Processes Panel

- Add I/O ports by **Add → I/O Marker**. Using the mouse place an I/O pad at the end of each wire that is not connected to anything. Ports will be automatically labeled with the name of the connected wire.
- You may change the label of each I/O port by double clicking on an I/O name and typing a new name in the place indicated in Figure 5.

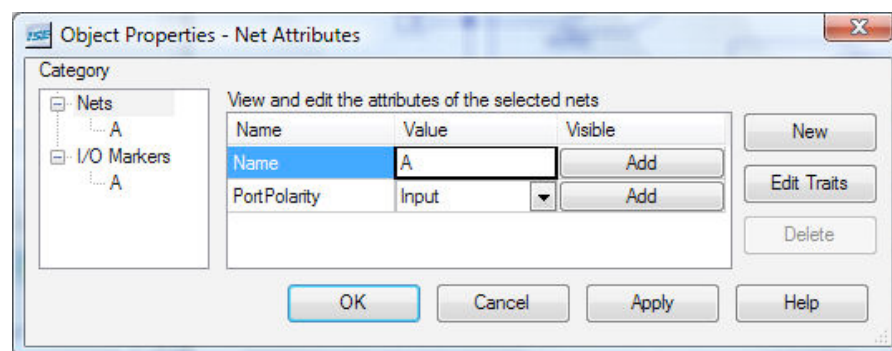


Figure 5. Change Pad Name

- The completed schematic is shown in Figure 6. Save the schematic and close the file.

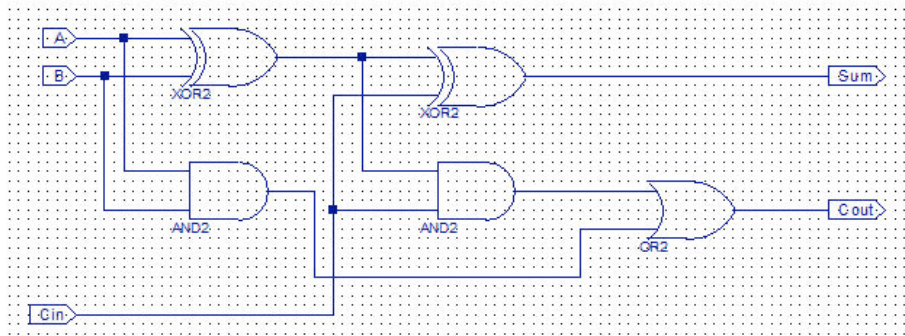


Figure 6. Completed Logic Circuit

## 5. Simulate a Circuit in Xilinx ISE

A digital circuit captured in schematics can be simulated using Xilinx ISE Simulation tools. A test bench must be created in order to simulate a designed circuit. The HDL Bench tool in Xilinx ISE is used to create a test bench which feeds stimuli for each input signal into the circuit and generate output responses. The input stimuli are created by user manually, and the output responses are generated by the tool automatically. All signals are captured in a timing diagram (waveform). Designers can analyze the timing diagram to verify the functionality of the circuit.

- Create stimuli for each input of the circuit.
  - Create a new test bench by **Project** → **New Source**.
  - In the **New Source Wizard** dialog box select **Test Bench Waveform**
  - Assign a file name
  - **Next** → **Next** → **Finish**, the Initial Timing and Clock Wizard dialog window appears.
  - Configure the test bench timing parameters as in the window shown in Figure 7.

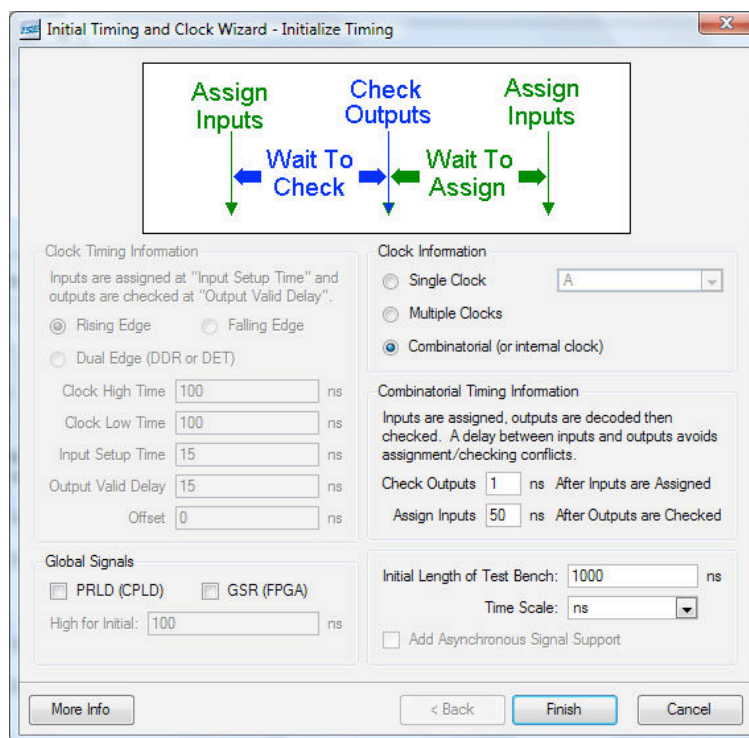


Figure 7. Initial Timing and Clock Wizard Dialog Window

- Click **Finish** then the test bench timing diagram will appear as shown in Figure 8.

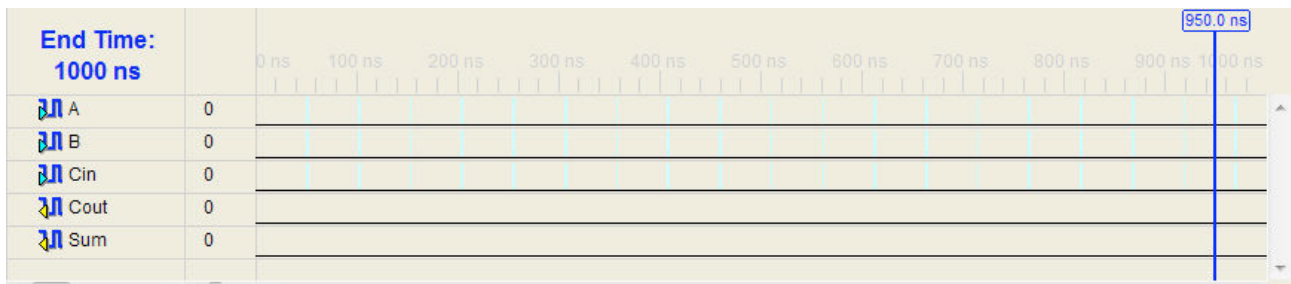


Figure 8. Test Bench Waveform

- Create stimuli for input signals. Use above truth table to create stimuli for the inputs.
  - Click on the input waveforms (the ones next to signal A and B) to change the values from 0 to 1 or 1 to 0. Perform an exhaustive verification on the circuit by testing all the possible combinations of the inputs (000, 001, 010 ... 111), as shown in Figure 9.

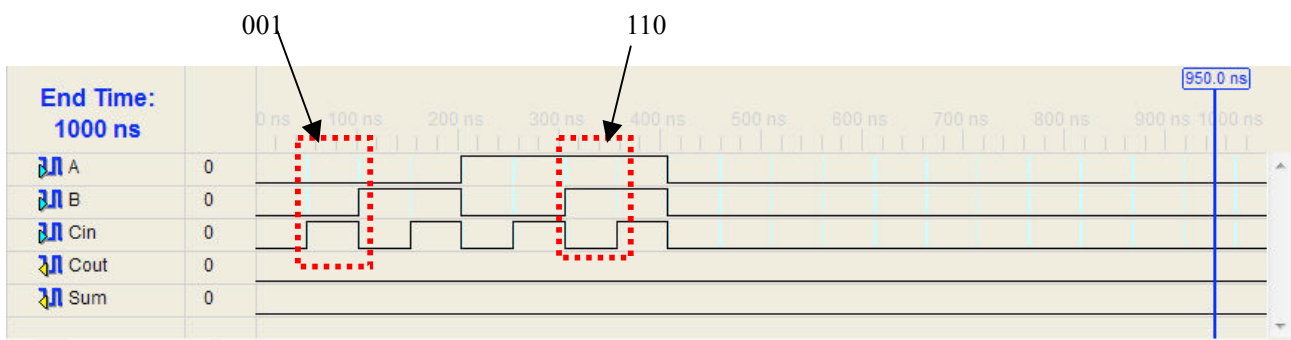


Figure 9. Test Bench Waveform with Specified Inputs

- Save the waveform. Click **Sources** panel, select **Sources for: Behavioral Simulation**, you will see the hierarchical structure of the test bench and circuit as shown in Figure 10. Notice now the Full Adder circuit is under the test bench and called UUT – Unit Under Test.

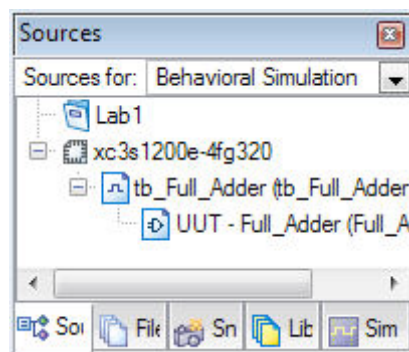
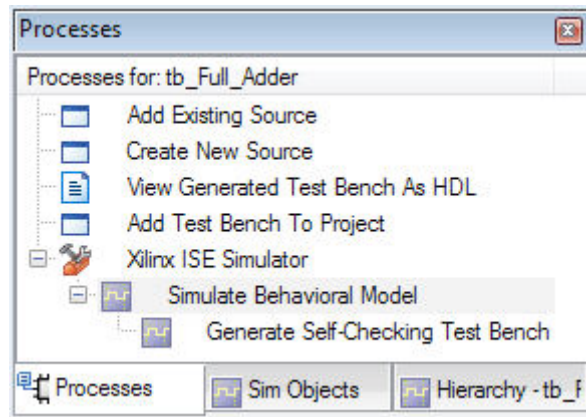


Figure 10. Test Bench in the project

- Simulate the circuit and generate output waveforms.
  - In the **Sources** window, select the test bench file (*tb\_Full\_Adder* in the example).
  - In the **Processes** panel, expand **Xilinx ISE Simulator**. Double click **Simulate Behavioral Model** to start simulation, as shown in Figure 11.



**Figure 11. Activating behavioral simulation**

- Now observe the output waveforms generated by the test bench. Complete the following truth table for the full adder circuit. Have it checked by one of the TAs before you go to the next step.

A	B	Cin	Sum	Cout

## 6. Creating Macro for Sub-Circuit

The number of logic gates used in a logic circuit may vary from less than ten to more than tens of millions. When a circuit gets bigger, it may be impossible to show all the details on the same level of abstract. In such cases, a hierarchical structure is more preferable. Generally, it is a good practice to group functionally related parts of a circuitry into manageable sub-circuits. These manageable sub-circuits are referred to as Macros in Xilinx ISE. This circuit design approach makes your circuit easier to develop and maintain. Additionally, this approach makes it possible to reuse a sub-circuit and share the Macros designed by different parties within a design group.

- Create a Macro for the Full Adder circuit.
  - Select **Tool** → **Symbol Wizard**
  - In the **Symbol Wizard** dialog window, select **Using schematic: Full\_Adder** as the **Pin name source**, select the shape of the Full\_Adder Macro that you are creating. We choose **Square** for this example.
  - Configure the layout of the I/O ports on the Macro as shown in Figure 12.



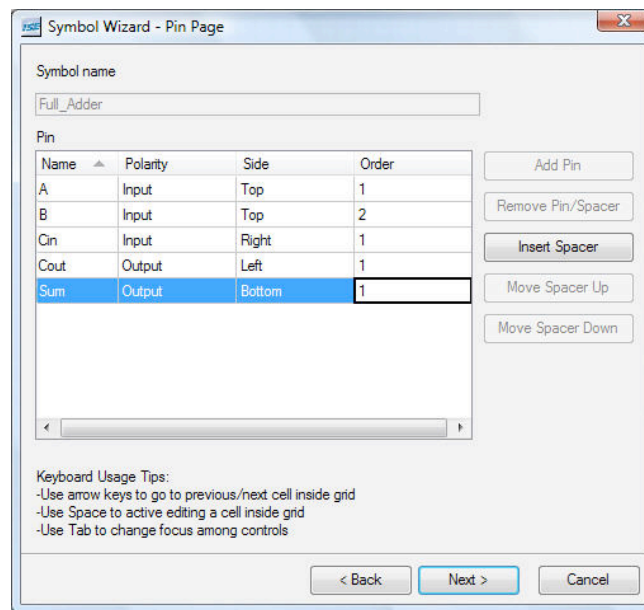


Figure 12. Configure I/O Layout for Macro

- Specify appropriate sizes for the Macro.
- Click through the remaining steps, a symbol will be created as shown in Figure 13.

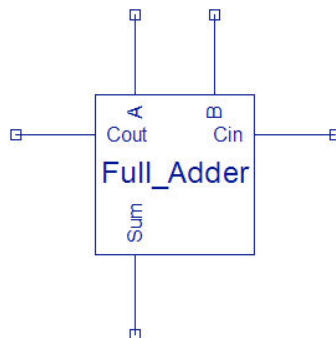
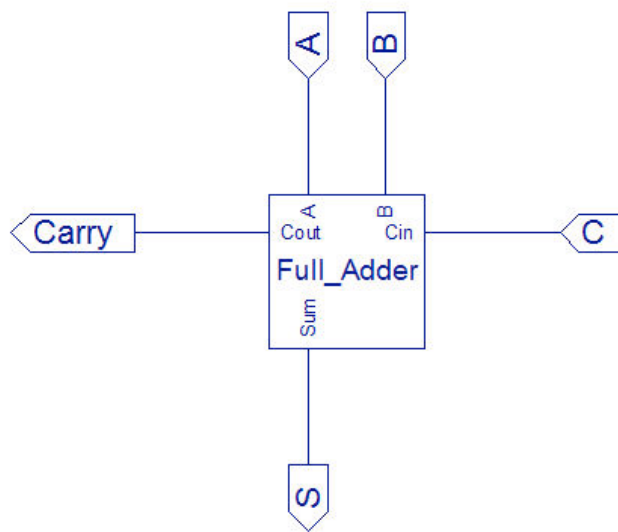


Figure 13. Symbol for the full adder Macro

- Use the Macro to draw another circuit. Now the Macro may be treat as another logic part and used to build bigger circuit.
  - Add another new schematic source file to your project in Xilinx ISE. Build a Full Adder circuit using the Full\_Adder Macro.
  - In the **Symbols** panel on the left hand side, select your working folder in **Categories**, then select Full\_Adder from the **Symbols** window.
  - Add wires and I/O Markers to finish the circuit. Note: using Macro in this circuit is actually unnecessary for this example, it's used for instructional purposes only. The completed schematic is shown in Figure 14.





**Figure 14. Full Adder circuit implemented using Full\_Adder Macros**

- Save the file with a different file name.
- Simulate the circuit.
  - Add the new schematic source file in the Xilinx project by **Project → Add Source**.
  - A hierarchical structure now can be seen in the Sources window in Xilinx ISE.
  - Simulate the circuit and verify the outputs. The simulated output waveform should be identical to the ones you obtained from the previous circuit.

## 7. Digital Logic Synthesis

The circuit you drew using Xilinx schematic capture tool is virtual. They are used to simulate and verify the functionality of the circuit. They should be converted into some real hardware circuit eventually. The process of converting a virtual design into real circuits is called synthesis. We will use Xilinx Synthesis Tool (XST) to generate a circuit that can be implemented in a *Field Programmable Gate Array* (FPGA) chip. The synthesized result is a bit-stream file that will be downloaded from the host PC into the FPGA chip through a USB connection.

The Spartan3E FPGA chip (xc3s1200e) is mounted on a *printed circuit board* (PCB), called Nexys2 board. The board also provides I/O resources for users to test/implement a digital device. It has eight *single pole double throw* (SPDT) switches and four push buttons that can be used to provide input signals to a circuit. Eight red LEDs can be used to capture the outputs. There are also four common anode seven-segment displays (SSDs). All these I/O resources are physically connected to the FPGA chip.

However, if you only download your circuit into the FPGA chip, the inputs and outputs of your circuit are not yet connected to any pins of the FPGA chip. You have to tell the FPGA which input or output of your circuit should be mapped to which FPGA chip pin. This step is called Pin Assignment. Each I/O resource is given a unique label that you may use to connect to the input and output ports of your circuit. The labels are printed on the FPGA board next to the I/O resources. For instance, switch 7 (SW7) is labeled as R17, and LED 5 (LD5) is labeled as P15\*. You may find the labels for the desired I/Os and map them to the inputs or outputs of your circuit in Pin Assignment. An example Pin Assignment for the Full Adder circuit is showing in Figure 15, where the circuit uses Switch 7 for A, Switch 6 for B, Switch 5 for Cin (C), LED 7 for Cout (Carry), and LED 6 for Sum (S).

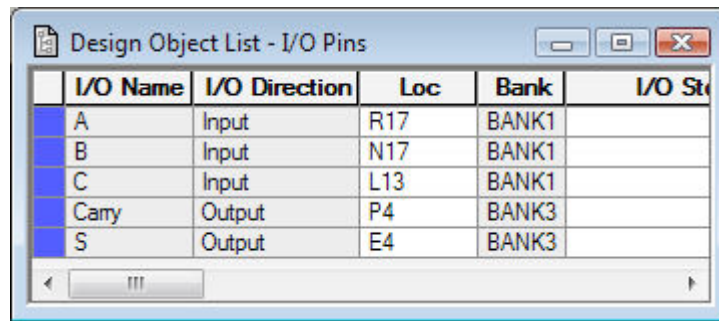



Figure 15. Pin Assignment

\*Note: we have two types of FPGA boards in our labs, XC3s1200E and XC3s500E. Other than the capacity of the chip, the FPGA pins connected to the LEDs LD4 – LD7 are also different for the two boards. The difference is captured in following table.

	XC3s1200E	XC3s500E
LD4	E16	E17
LD5	P16	P15
LD6	E4	F4
LD7	P4	R4

Follow through this sequence to synthesize the circuit you drew to a real digital circuit.

- Make pin assignment for your circuit:
  - In the **Sources** window, select **Source for: Implementation**.
  - Select the top-level circuit which is indicated by . Then **Processes** window shows the tasks involved in the synthesis process.
  - Expand the **User Constraints**. Double clicking **Floorplan IO – Pre-Synthesis** will bring up the **Xilinx PACE** window, where you will make the pin assignment for your circuit.
  - Find the **Design Object List – I/O Pins** window, assign the desired FPGA pin numbers in the Loc column as shown in Figure 15. We will use Switch 7 for A, Switch 6 for B, Switch 5 for Cin (C), LED 7 for Cout (Carry), and LED 6 for Sum (S).
  - Save the file. Use **XST Default: <>** as the **Select IO Bus Delimiter**.
  - Now you can see another file called xxxx.ucf appears under the top-level circuit. This .ucf file has the same name as your top-level circuit and tells the FPGA how to map your circuit I/Os to the FPGA pins.
- Synthesize the circuit.
  - Select the top-level circuit.
  - Right click **Generate Programming File → Properties**, on **Startup Options** panel select **JTAG Clock** for the **FPGA Start-Up Clock**. This is shown in Figure 16. Then click OK.

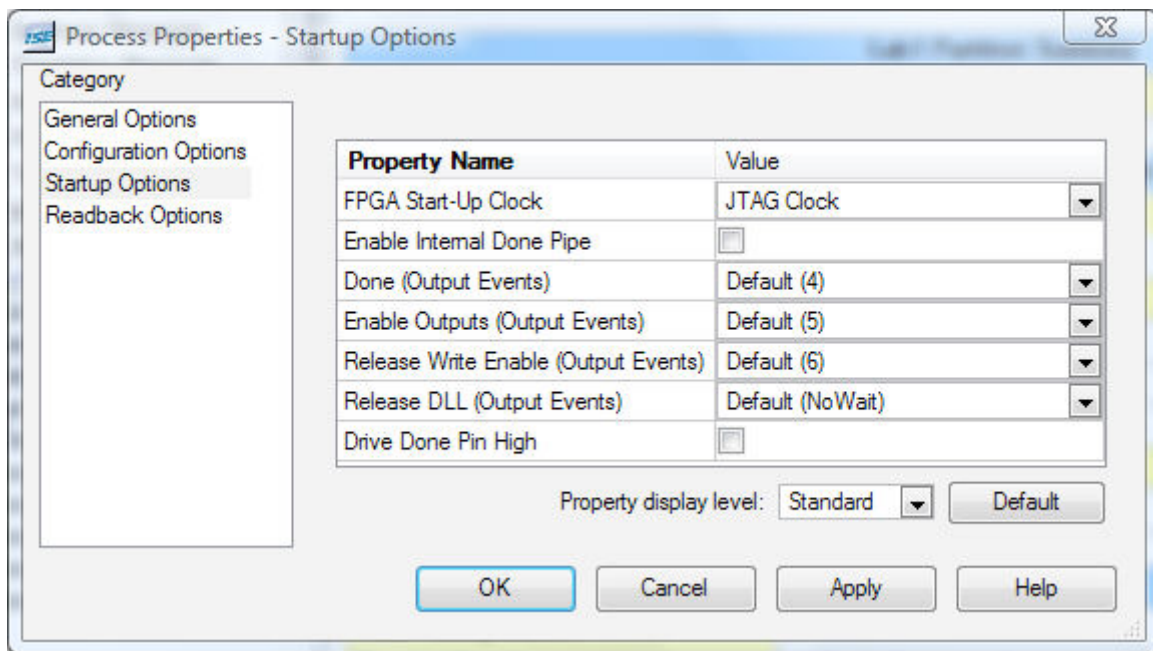


Figure 16. Options in Process Properties

- Double click on **Generate Programming File** in the **Processes** window will launch the synthesis process. The items above Generate Programming File, such as Synthesize – XST and Implement Design are steps that the synthesis process has to go through. After each step in the synthesis process is successfully completed, a green check mark will be seen. Otherwise a red cross will be given indicating errors and an exclamation mark will be shown indicating warnings.
- A binary bitstream file will be generated that captures your entire design and all user constraints and can be used to program an FPGA.
- Now we are done using the Xilinx ISE tool.

## 8. FPGA Implementation

The bitstream file generated from the synthesis can be downloaded to program an FPGA chip so that a real logic circuit can be implemented in the FPGA. There are two ways to connect a PC with the FPGA board: through a JTAG cable or through a USB cable. We will use a USB connection to program the FPGA and also to power the FPGA board. There are two ways to program the FPGA, one is to download the bitstream file from PC to the FPGA chip directly; the other is to generate a different bitstream file and download it to the on-board flash memory, then program the FPGA from the flash memory. We will program the FPGA directly from the PC.

The Xilinx ISE does not support direct PC to FPGA download through USB connection. But the board manufacturer Digilent company provides such a tool.

- Download and install the software tool **Adept** from the course website. The Adept software startup window is shown in Figure 17.

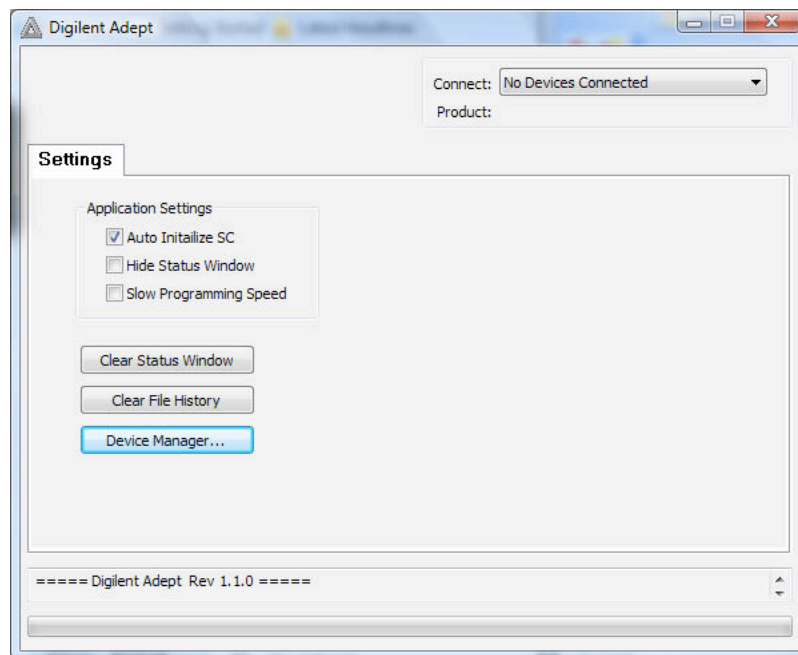


Figure 17. Startup Window of Digilent Adept Software

- Connect the FPGA board to PC using the supplied USB cable, and allow the USB driver to be installed automatically.
- On the up left corner, there is a power switch. Turn on the switch to power the board.
- In the Adept software window, change **Connect** to **Nexys2**. Then the connection will be automatically established and initialized. This is shown in Figure 18.

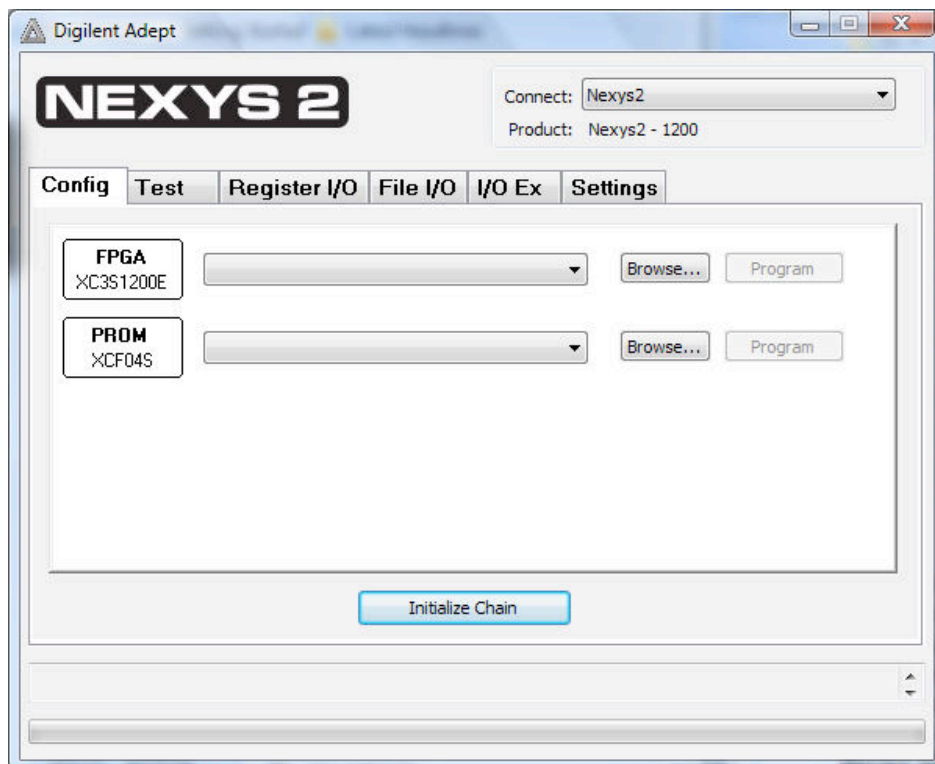


Figure 18. Adept software with Nexys2 board connected

- Browse to your Xilinx ISE project folder for the FPGA and select the .bit bitstream file. You don't have to worry about the PROM (flash memory) in this lab.
- Click Program to start programming the FPGA chip. As soon as the programming is finished, a successful message will be displayed, as shown in Figure 19.



**Figure 19. Successful message**

- Verify your circuit on the board against the truth table.

## 9. Deliverable

1. This is a 1-week lab
2. No lab report is required for this lab.
3. Demonstrate your board to the lab assistants before you sign out.

The full score for this lab is 100 points.