Experiment 2
Introduction to Altera and Schematic Programming

Prepared by: Eng. Shatha Awawdeh, Eng.Eman Abu_Zaitoun

Introduction:

This tutorial introduces the basic features of the Quartus II software. It shows how the software can be used to design and implement a circuit specified by using the means of a schematic diagram. It makes use of the graphical user interface to invoke the Quartus II commands.

Objectives:

- Creating a project.
- Design entry using schematic diagram.
- Assigning the circuit inputs and outputs to specific pins on the FPGA.
- Simulating the designed circuit.
- Programming and configuring the FPGA device.

1- Getting Started:

Each logic circuit, or sub circuit, being designed with Quartus II software is called a project. The software works on one project at a time and keeps all information for that project in a single directory (folder) in the file system.

To begin a new logic circuit design, the first step is to create a directory to hold its files. To hold the design files for this lab, we will use a directory Exp2. The running example for this Experiment is a simple circuit for Xor gate (A XOR B = (A& (~B)) | ((~A) &B)).

Start the Quartus II software. You should see a display similar to the one in Figure 1. This display consists of several windows that provide access to all the features of Quartus II software, which the user selects with the computer mouse. Most of the commands provided by Quartus II software can be accessed by using a set of menus that are located below the title bar (File, Edit, view, project…).

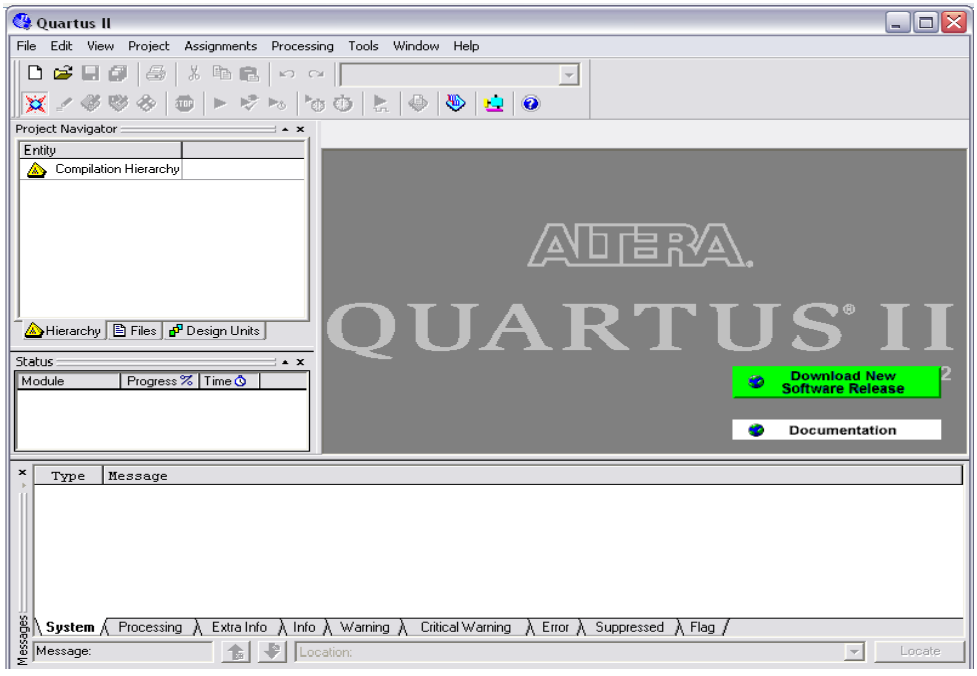


Figure 1. The main Quartus II display.

1.1 Quartus II Online Help

Quartus II software provides comprehensive online documentation that answers many of the questions that may arise when using the software. The documentation is accessed from the Help menu. To get some idea of the extent of documentation provided, it is worthwhile for the reader to browse through the Help menu. For instance, selecting Help > How to Use Help gives an indication of what type of help is provided. The user can quickly search through the Help topics by selecting Help > Search, which opens a dialog box into which keywords can be entered. Another method, context-sensitive help, is provided for quickly finding documentation for specific topics. While using most applications, pressing the F1 function key on the keyboard opens a Help display that shows the commands available for the application.

2- Starting a New Project

To start working on a new design we first have to define a new *design project*. Quartus II software makes the designer's task easy by providing support in the form of a *wizard*.

1. Create a new project; select **File > New Project Wizard** to reach the window in Figure 2b, which asks for the name and directory of the project.

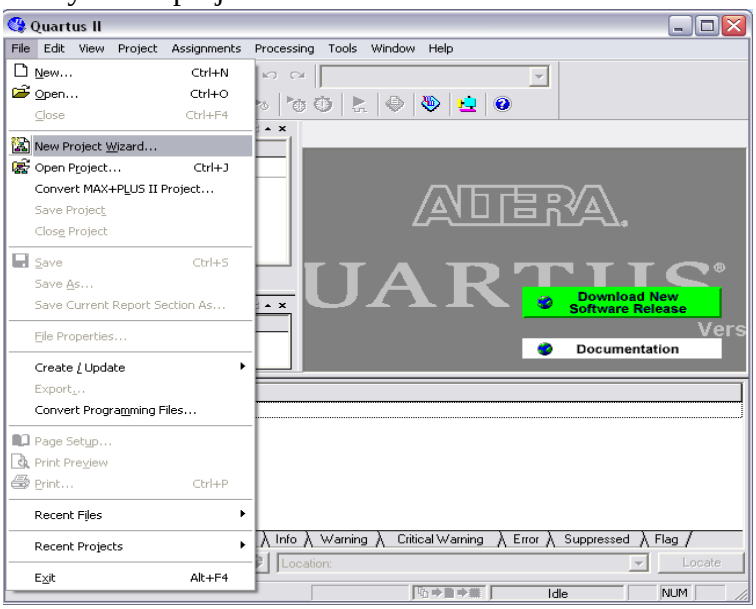


Figure 2a. Creation of a new project.

2. Set the working directory to be *Exp2*, The project must have a name, which is usually the same as the top-level design entity (schematic circuit) that will be included in the project. Choose *Xor1* as the name for both the project and the top-level entity, as shown in Figure 2b.

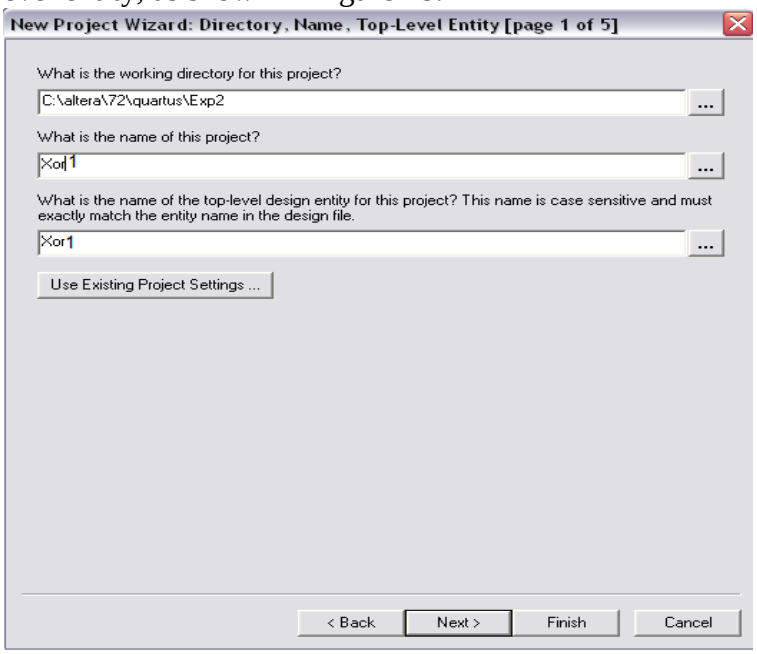


Figure 2b. Creation of a new project.

Press **Next**. Since we have not yet created the directory *Exp2*, Quartus II software displays the pop-up box in Figure 3 asking if it should create the desired directory. Click **Yes**, which leads to the window in Figure 4.

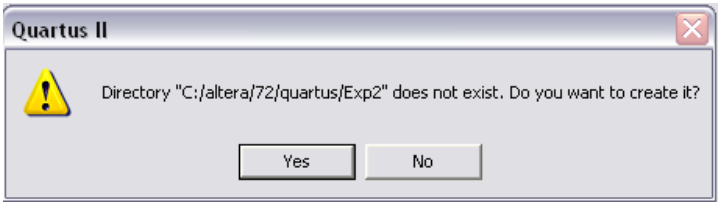


Figure 3. Quartus II software can create a new directory for the project.

3. The wizard makes it easy to specify which existing files (if any) should be included in the project. Assuming that we do not have any existing files, click **Next**, which leads to the window in Figure 5.

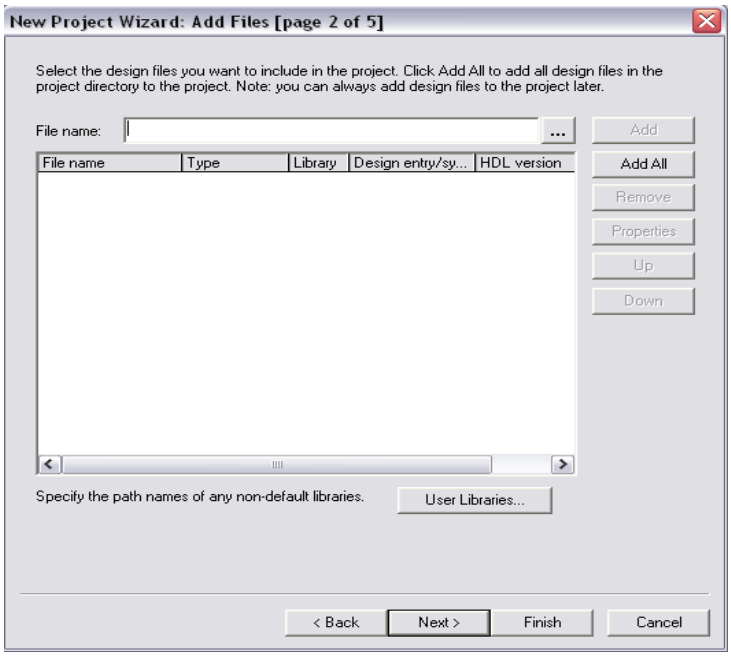


Figure 4. The wizard can include user-specified design files.

4. We have to specify the type of device in which the designed circuit will be implemented. Choose **CycloneII** as the target device family. We can let Quartus II software select a specific device in the family, or we can choose the device explicitly. We will take the latter approach. From the list of available devices, choose the device called **EP2C70F896C6** which is the FPGA used on Altera's DE2 board. Press **Next**, which opens the window in Figure 6.

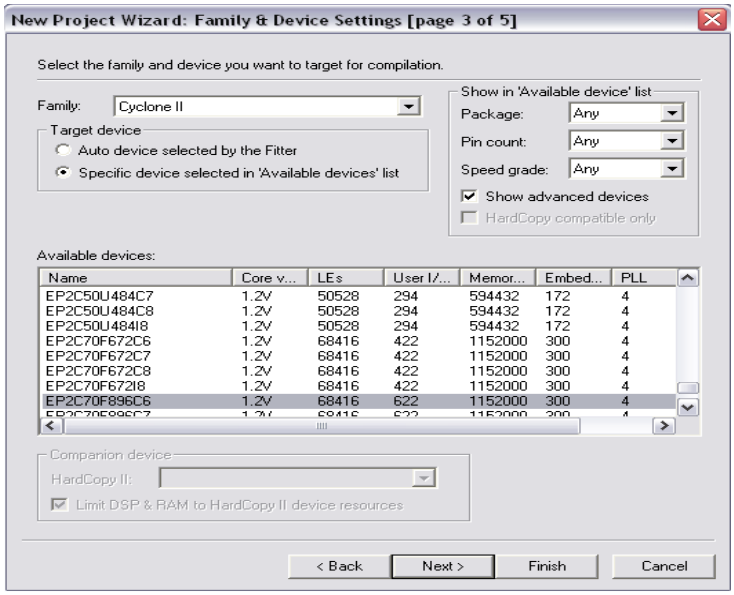


Figure 5. Choose the device family and a specific device

5. The user can specify any third-party tools that should be used. A commonly used term for CAD software for electronic circuits is EDA tools, where the acronym stands for Electronic Design Automation. This term is used in Quartus II messages that refer to third-party tools, which are the tools developed and marketed by companies other than Altera. Since we will rely solely on Quartus II tools, we will not choose any other tools. Press **Next**.

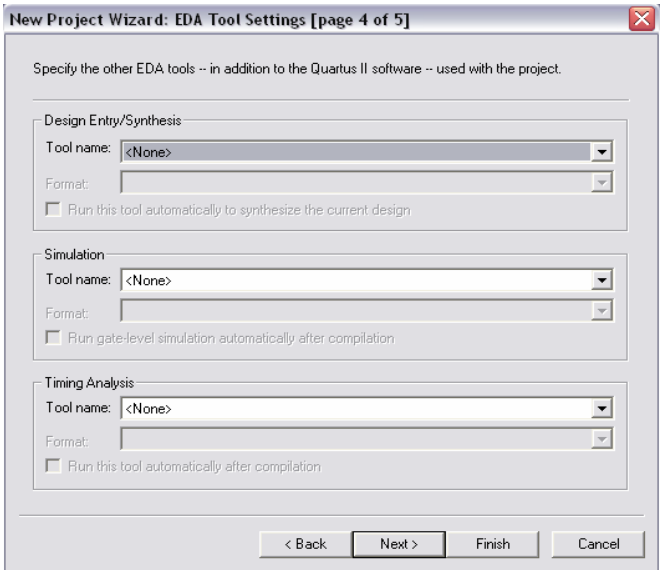


Figure 6. Other EDA tools can be specified.

6. A summary of the chosen settings appears in the screen shown in Figure 7. Press **Finish**, which returns to the main Quartus II window, but with Xor specified as the new project, in the display title bar, as indicated in Figure 8.

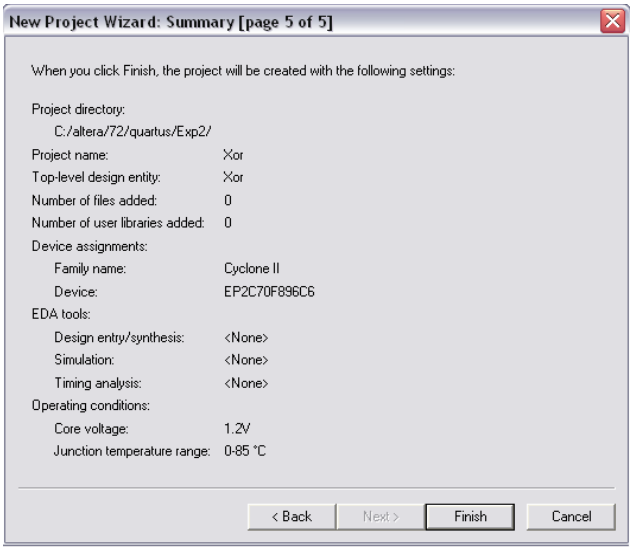


Figure 7. Summary of the project settings.

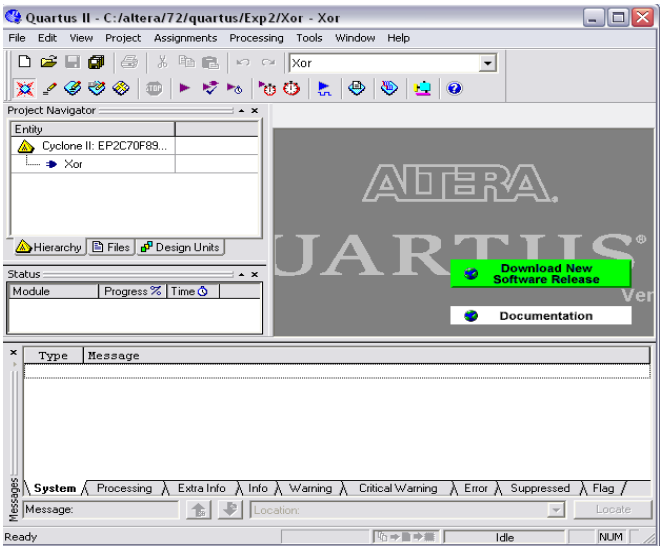
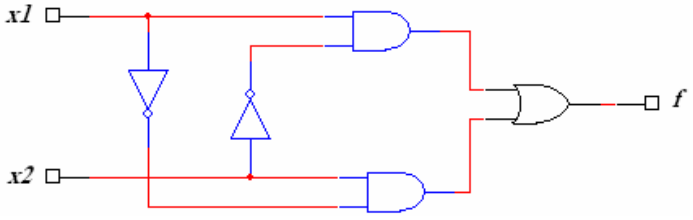


Figure 8. The Quartus II display for the created project.

3- Design Entry Using the Graphic Editor

As a design example, we will use the Xor circuit shown in Figure 9. The circuit has two input switches x1 and x2, where a closed switch corresponds to the logic value 1. The truth table for the circuit is also given in the figure.



X1	*X2*	*f*
0	0	0
0	1	1
1	0	1
1	1	0

Figure 9. The Xor function circuit.

The Quartus II Graphic Editor can be used to specify a circuit in the form of a block diagram. Select **File > New** to get the window in Figure 10, choose **Block Diagram/Schematic File**, and click **OK**. This opens the Graphic Editor window.

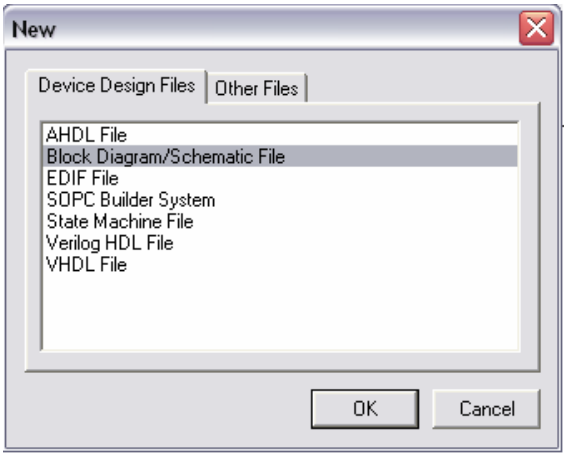


Figure 10. Choose to prepare a block diagram

The first step is to specify a name for the file that will be created. Select **File > Save As** to open the pop-up box depicted in Figure 11. In the box labeled **Save as type choose Block Diagram/Schematic File (*.bdf)**. In the box labeled File name type **Xor1**, to match the name given in Figure 2b, which was specified when the project was created. Put a checkmark in the box **Add file to current project**. Click **Save**, which puts the file into the directory Exp2 and leads to the Graphic Editor window displayed in Figure 12.

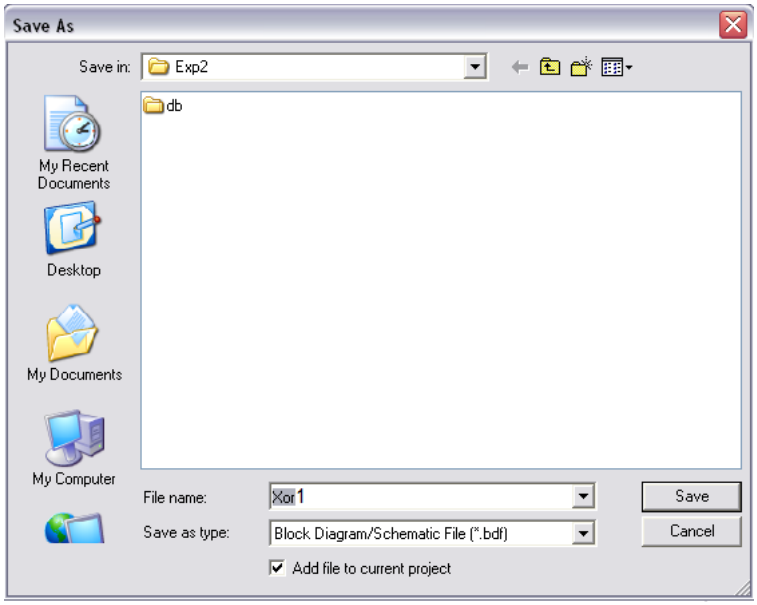


Figure 11. Name the file.

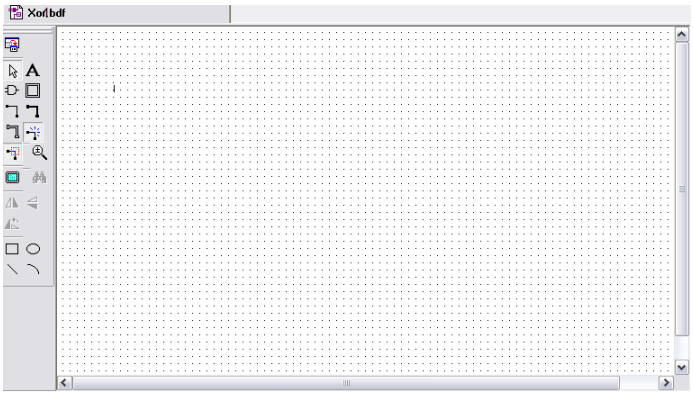


Figure 12. Graphic Editor window.

3.1 Importing Logic-Gate Symbols

The Graphic Editor provides a number of libraries which include circuit elements that can be imported into a schematic. **Double-click** on the blank space in the Graphic Editor window, or click on the icon in the toolbar that looks like an AND gate. A pop-up box in Figure 13 will appear.

Expand the hierarchy in the Libraries box as shown in the figure. First **expand libraries**, and then **expand the library primitives**, followed by **expanding the library logic** which comprises the logic gates. **Select and2**, which is a two-input AND gate, and click **OK**. Now, the AND gate symbol will appear in the Graphic Editor window. Using the mouse, move the symbol to a desirable location and **click** to place it there.

Import the second AND gate, which can be done simply by positioning the mouse pointer over the existing AND-gate symbol, right-clicking, and dragging to make a copy of the symbol. A symbol in the Graphic Editor window can be moved by clicking on it and dragging it to a new location with the mouse button pressed. Next, select or2 from the library and import the OR gate into the diagram. Then, select not and import two instances of the NOT gate. Rotate the NOT gates into proper position by using the "Rotate left 90" icon. Arrange the gates as shown in Figure 14.

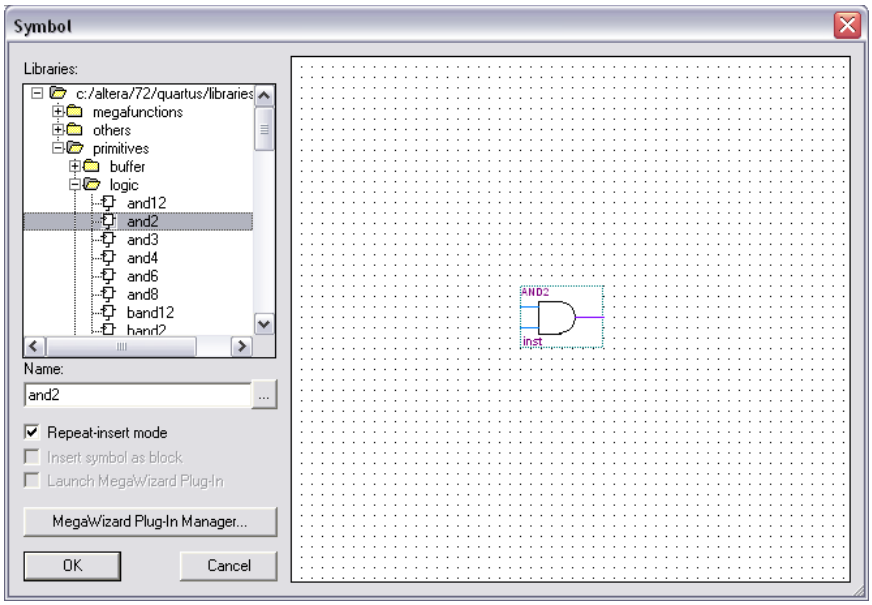


Figure 13. Choose a symbol from the library.

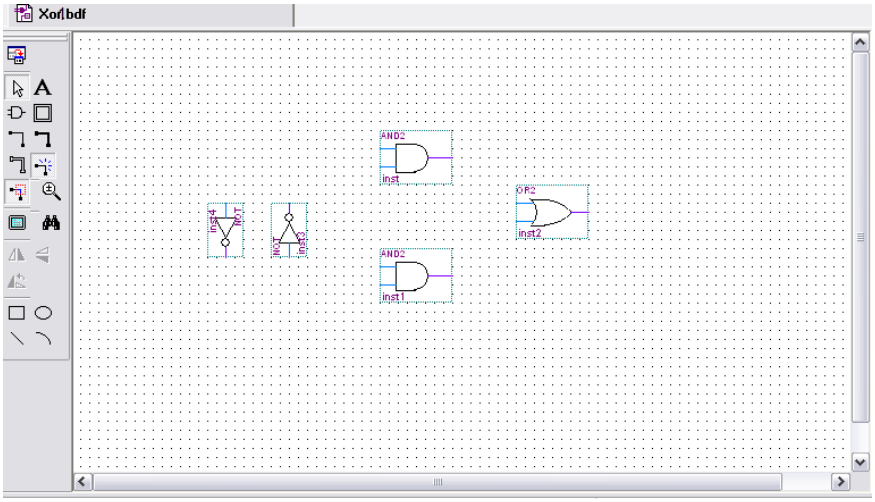


Figure 14. Import the gate symbols into the Graphic Editor window.

3.2 Importing Input and Output Symbols

Having entered the logic-gate symbols, it is now necessary to enter the symbols that represent the input and output ports of the circuit. Use the same procedure as for importing the gates, but choose the port symbols from the library primitives/pin. Import two instances of the input port and one instance of the output port, to obtain the image in Figure 15.

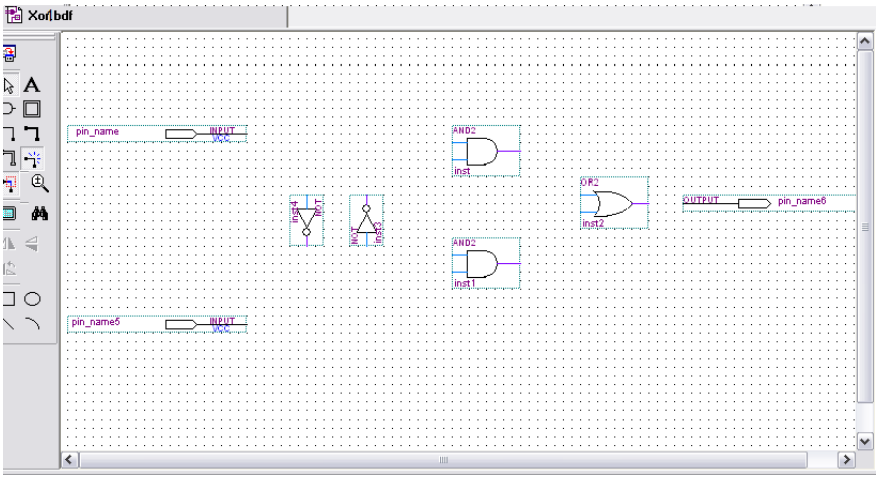


Figure 15. Import the input and output pins.

Assign names to the input and output symbols as follows. Make sure nothing is selected by clicking on an empty spot in the Graphic Editor window. **Point** to the word pin_name on the top input symbol and **double-click** the mouse. The dialog box in Figure 16 will appear. **Type the pin name**, x1, and click **OK**. Similarly, assign the name x2 to the other input and f to the output. Alternatively, it is possible to change the name of an element by selecting it first, and then double-clicking on the name and typing a new one directly.

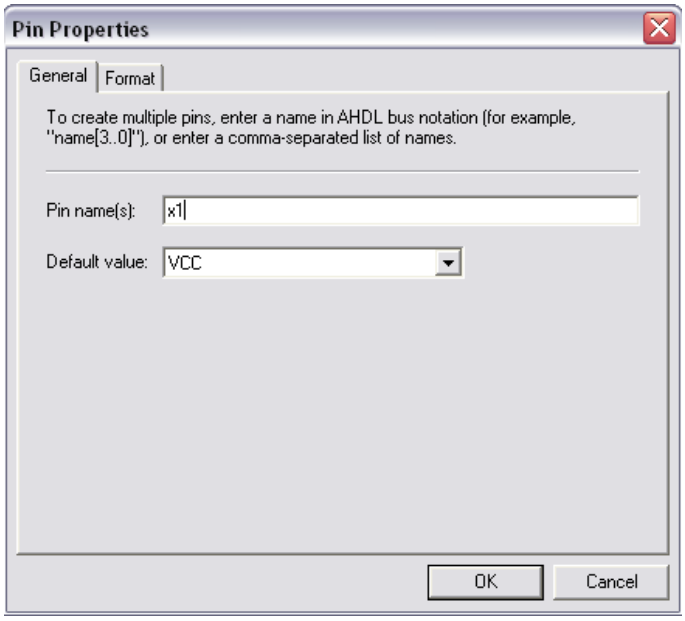


Figure 16. Naming of a pin.

3.3 Connecting Nodes with Wires

The symbols in the diagram have to be connected by drawing lines (wires). **Click on the icon** in the toolbar to activate the Orthogonal Node Tool. **Position the mouse** pointer over the right edge of the x1 input pin. **Click and hold** the mouse button and **drag** the mouse to the right until the drawn line reaches the pinstub on the top input of the AND gate. **Release** the mouse button, which leaves the line connecting the two pinstubs. Next, draw a wire from the input pinstub of the leftmost NOT gate to touch the wire that was drawn above it. Note that a dot will appear indicating a connection between the two wires.

Use the same procedure to draw the remaining wires in the circuit. If a mistake is made, a wire can be selected by clicking on it, and removed by pressing the Delete key on the keyboard. Upon completing the diagram, click on the icon , to activate the Selection Tool. Now, changes in the appearance of the diagram can be made by selecting a particular symbol or wire and either moving it to a different location or deleting it. The final diagram is shown in Figure 17; **save it.**

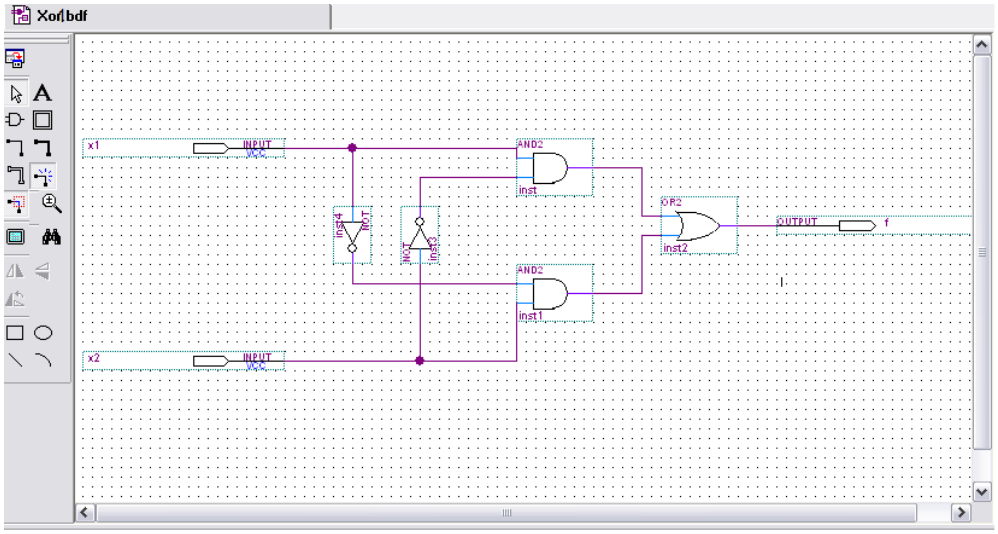


Figure 17. The completed schematic diagram.

4- Compiling the Designed Circuit

The entered schematic diagram file, Xor.bdf, is processed by several Quartus II tools that analyze the file, synthesize the circuit, and generate an implementation of it for the target chip. These tools are controlled by the application program called the Compiler.

Run the Compiler by selecting **Processing > Start Compilation**, or by clicking on the toolbar icon that looks like a **purple triangle** . As the compilation moves through various stages, its progress is reported in a window on the left side of the Quartus II display. Successful (or unsuccessful) compilation is indicated in a pop-up box.

Acknowledge it by clicking **OK**, which leads to the Quartus II display in Figure 18. In the message window, at the bottom of the figure, various messages are displayed. In case of errors, there will be appropriate messages given.

When the compilation is finished, a compilation report is produced. A window showing this report is opened automatically, as seen in Figure 18. The window can be resized, maximized, or closed in the normal way, and it can be opened at any time either by selecting **Processing > Compilation Report** or by clicking on the icon .

The report includes a number of sections listed on the left side of its window. Figure 18 displays the Compiler Flow Summary section, which indicates that only one logic element and three pins are needed to implement this tiny circuit on the selected FPGA chip.

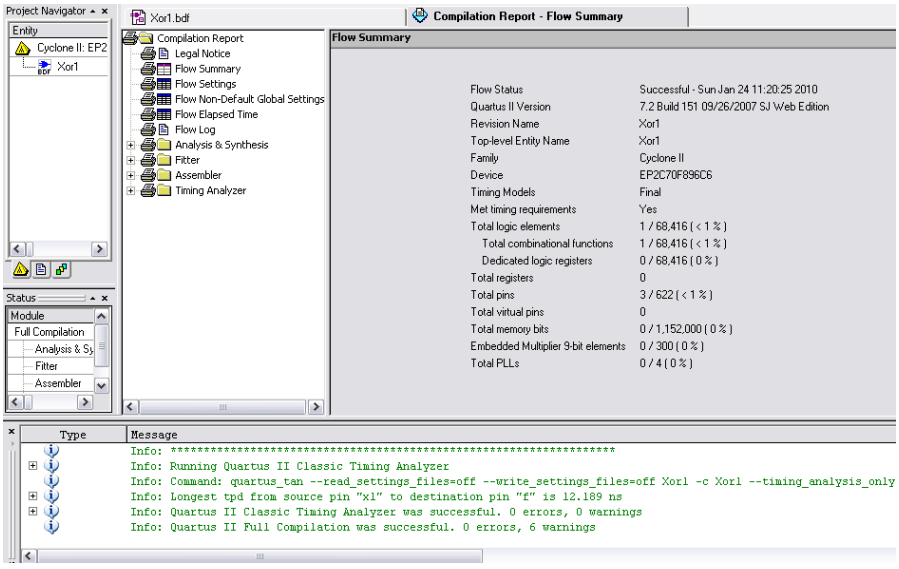


Figure 18. Display after a successful compilation.

In the case of unsuccessful compilation, Figure 19 shows the compilation report (Flow Summary).

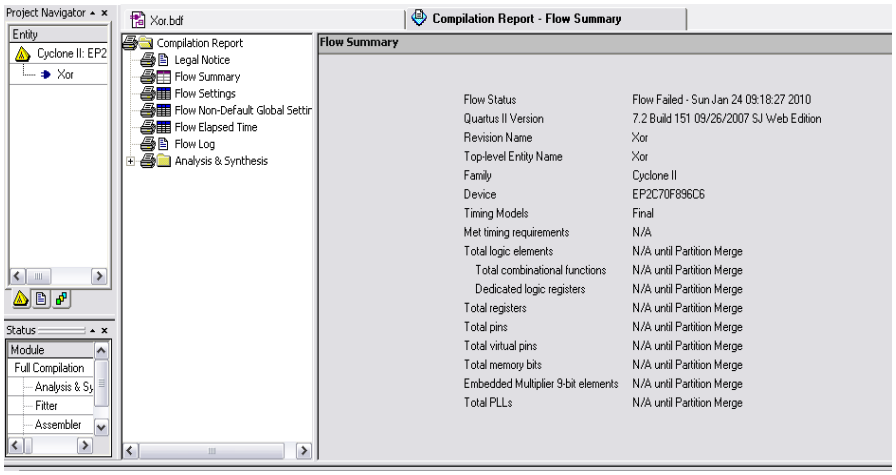


Figure 19. Compilation report for the failed design.

In the message tab, all errors will be shown, see Figure 20

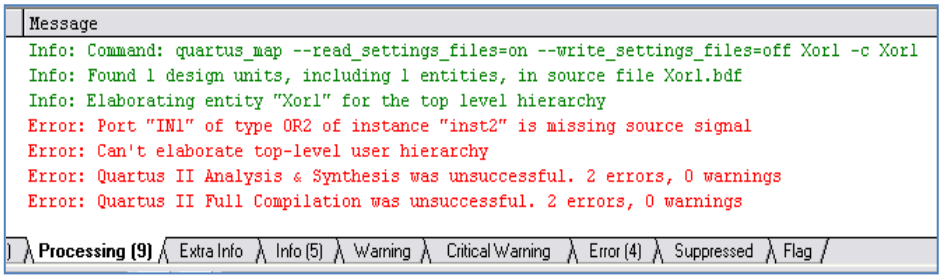


Figure 20. Error messages.

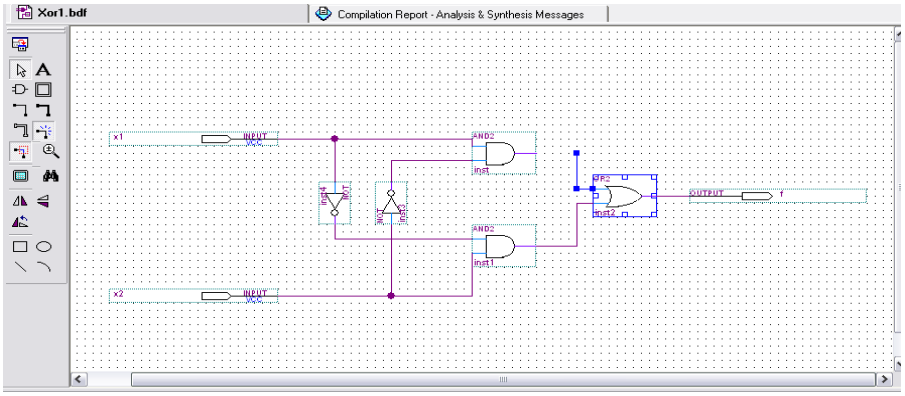


Figure 21. Identifying the location of the error.

After correcting all errors, recompile the circuit.

5- Simulating the Designed Circuit

Before implementing the designed circuit in the FPGA chip on the DE2 board, it is prudent to simulate it to ascertain its correctness. Quartus II software includes a simulation tool that can be used to simulate the behavior of a designed circuit. Before the circuit can be simulated, it is necessary to create the desired waveforms, called **test vectors**, to represent the input signals. It is also necessary to specify which outputs, as well as possible internal points in the circuit, the designer wishes to observe. The simulator applies the test vectors to a model of the implemented circuit and determines the expected response. We will use the Quartus II Waveform Editor to draw the test vectors, as follows:

1. Open the Waveform Editor window by selecting **File > New>Other Files** tab, which gives the window shown in Figure 26.Choose **Vector Waveform File** and click **OK**.

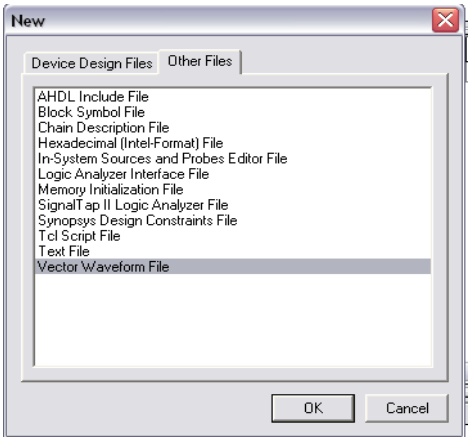


Figure 26. Need to prepare a new file.

2. The Waveform Editor window is depicted in Figure 27. **Save the file** under the name Xor.vwf; note that this changes the name in the displayed window. Set the desired simulation to run from 0 to 200 ns by selecting **Edit > End Time** and entering 200 ns in the dialog box that pops up. Selecting **View > Fit in Window** displays the entire simulation range of 0 to 200 ns in the window, as shown in Figure 28. You may wish to resize the window to its maximum size.

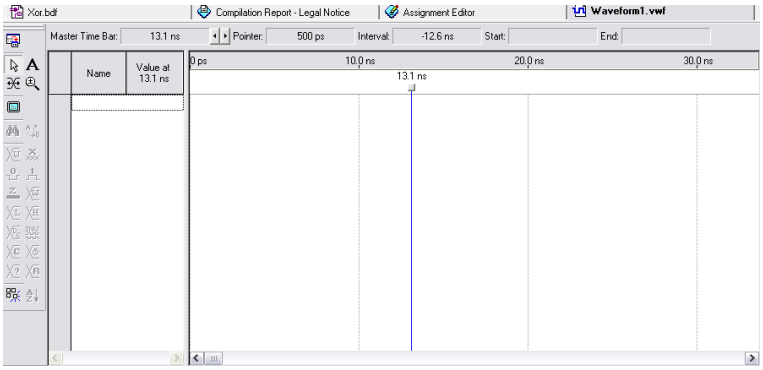


Figure 27. The Waveform Editor window.

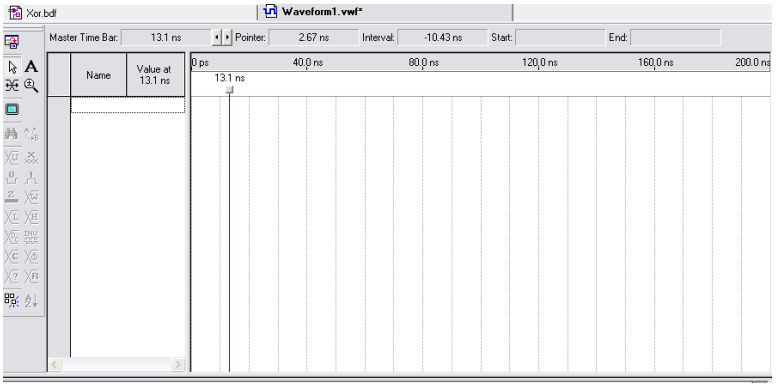


Figure 28. The augmented Waveform Editor window.

3. Next, we want to include the input and output nodes of the circuit to be simulated. Click **Edit > Insert> Insert Node or Bus** to open the window in Figure 29.

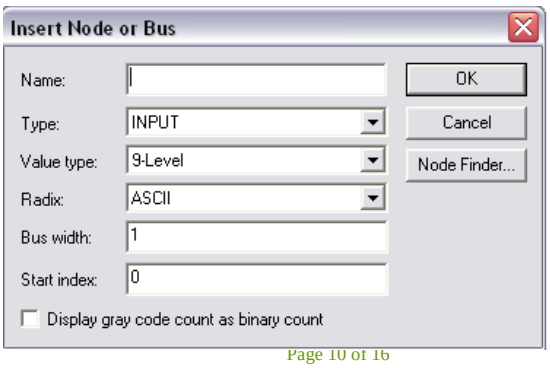


Figure 29. The Insert Node or Bus dialogue.

It is possible to type the name of a signal (pin) into the Name box, but it is easier to click on the **Node Finder** button to open the window in Figure 30. The Node Finder utility has a filter used to indicate what type of nodes are to be found. Since we are interested in input and output pins, **set the filter to Pins: all**. **Click the List button** to find the input and output nodes as indicated on the left side of the figure.

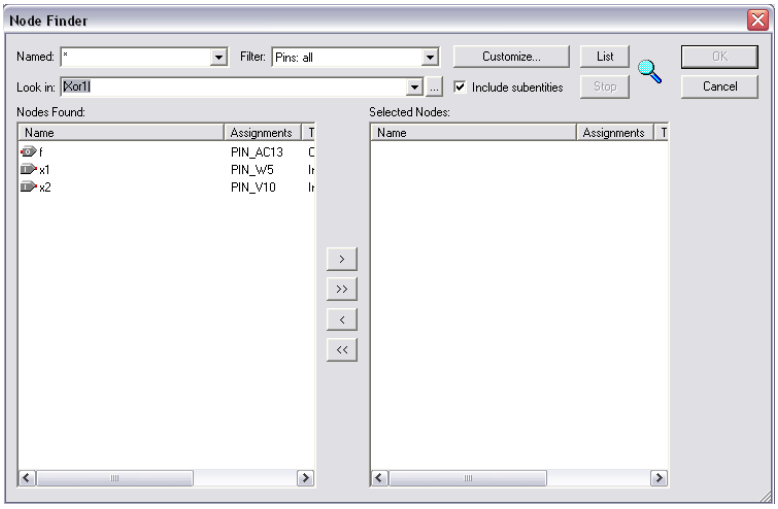


Figure 30. Selecting nodes to insert into the Waveform Editor.

Click on the x1 signal in the Nodes Found box in Figure 30, and then **click the** > **sign** to add it to the Selected Nodes box on the right side of the figure. Do the same for x2 and f. Click **OK** to close the Node Finder window, and then click **OK** in the window of Figure 29. This leaves a fully displayed Waveform Editor window, as shown in Figure 31. If you did not select the nodes in the same order as displayed in Figure 31, it is possible to rearrange them. To move a waveform up or down in the Waveform Editor window, click on the node name (in the Name column) and release the mouse button. The waveform is now highlighted to show the selection. Click again on the waveform and drag it up or down in the Waveform Editor.

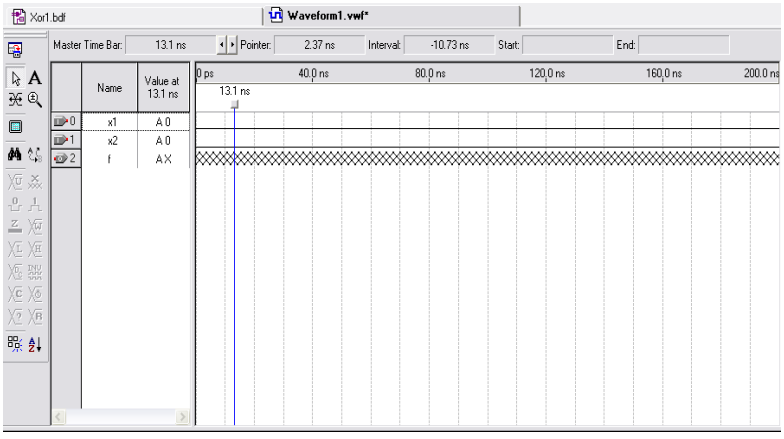


Figure 31. The nodes needed for simulation.

4. We will now specify the logic values to be used for the input signals x1 and x2 during simulation. The logic values at the output f will be generated automatically by the simulator. To make it easy to draw the desired waveforms, the Waveform Editor displays (by default) vertical guidelines and provides a drawing feature that snaps on these lines (which can otherwise be invoked by choosing **View > Snap to Grid**). Observe also a solid vertical line, which can be moved by pointing to its top and dragging it horizontally. This reference line is used in analyzing the timing of a circuit; move it to the time = 0 position.
The waveforms can be drawn using the **Selection Tool**, which is activated by selecting the icon in the toolbar, or the Waveform Editing Tool, which is activated by the icon .

To simulate the behavior of a large circuit, it is necessary to apply a sufficient number of input valuations and observe the expected values of the outputs. In a large circuit the number of possible input

valuations may be huge, so in practice we choose a relatively small (but representative) sample of these input valuations.

However, for our tiny circuit we can simulate all different combinations (00, 01, 10, 11) given in Figure 9. We will use four 50-ns time intervals to apply the four test vectors. We can generate the desired input waveforms as follows. **Click on the waveform name for the x1 node**. Once a waveform is selected, the editing commands in the Waveform Editor can be used to draw the desired waveforms. Commands are available for setting a selected signal to 0, 1, unknown (X), high impedance (Z), don't care (DC), inverting its existing value (INV), or defining a clock waveform. Each command can be activated by using the **Edit > Value** command or via the toolbar for the Waveform Editor. The Edit menu can also be opened by **right-clicking on a waveform name**.

Set x1 to 0 in the time interval 0 to 100 ns, which is probably already set by default. Next, set x1 to 1 in the time interval 100 to 200 ns. Do this by **pressing the mouse at the start of the interval and dragging it to its end**, which highlights the selected interval, and **choosing the logic value 1 in the toolbar**. Make x2 = 1 from 50 to 100 ns and also from 150 to 200 ns, which corresponds to the truth table in Figure 9. This should produce the image in Figure 32. Observe that the output f is displayed as having an unknown value at this time, which is indicated by a hashed pattern; its value will be determined during simulation. **Save the file as Xor1.vwf**.

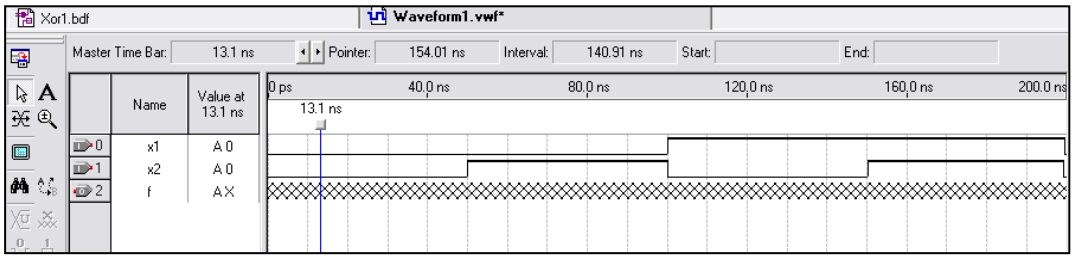


Figure 32. Setting of test values.

5.1 Performing the Simulation

A designed circuit can be simulated in two ways. The simplest way is to assume that logic elements and interconnection wires in the FPGA are perfect, thus causing no delay in propagation of signals through the circuit. This is called **Functional simulatio**n. A more complex alternative is to take all propagation delays into account, which leads to **Timing simulation**. Typically, functional simulation is used to verify the functional correctness of a circuit as it is being designed. This takes much less time, because the simulation can be performed simply by using the logic expressions that define the circuit.

5.1.1 Functional Simulation

To perform the functional simulation select **Assignments > Settings** to open the Settings window, on the left side of this window click on **Simulator Settings** to display the window in Figure 33, **choose Functional as the simulation mode**, **choose Xor1.vwf as the simulation input,** and click **OK**. The Quartus II simulator takes the inputs and generates the outputs defined in the Xor.vwf file.

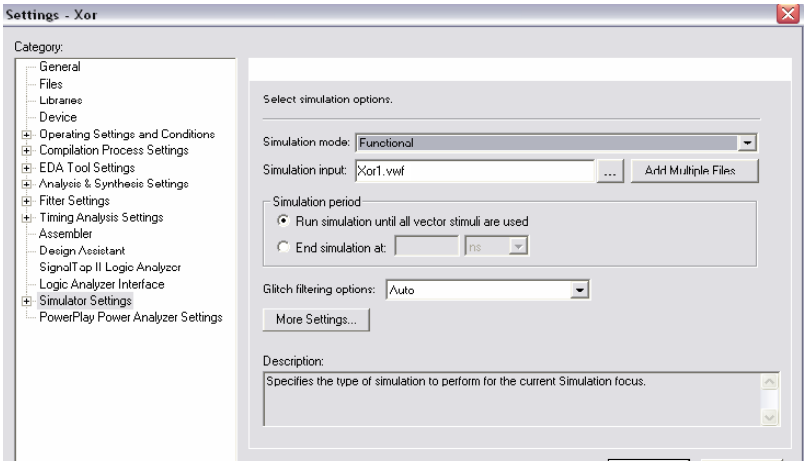


Figure 33. Specifying the simulation mode.

Before running the functional simulation it is necessary to create the required netlist; select **Processing > Generate Functional Simulation Netlist**. A simulation run is started by **Processing > Start Simulation.** At the end of the simulation, Quartus II software indicates its Successful completion and displays a **Simulation Report** illustrated in Figure 34. If your report window does not show the entire simulation time range, **click on the report window** to select it and choose **View > Fit in Window**.
Observe that the output f is as specified in the truth table of Figure 9.

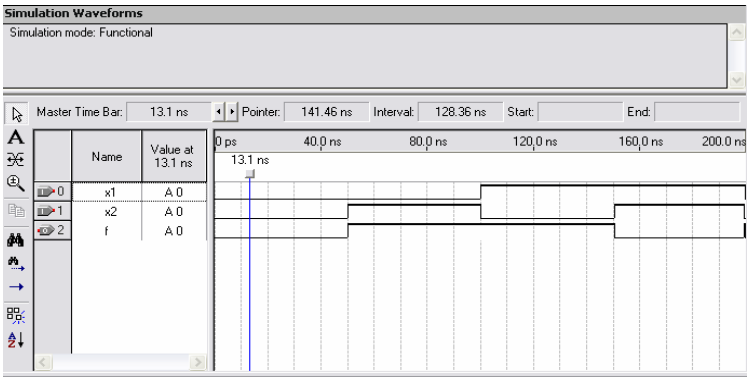


Figure 34. The result of functional simulation.

5.1.2 Timing Simulation

Having ascertained that the designed circuit is functionally correct, we should now perform the timing simulation to see how it will behave when it is actually implemented in the chosen FPGA device. Select **Assignments > Settings > Simulator Settings** to get to the window in Figure 33, **choose Timing as the simulation mode**, **choose Xor1.vwf as the simulation input,** and click **OK**. Run the simulator, which should produce the waveforms in Figure 35. Observe that there is a delay of about 6 ns in producing a change in the signal f from the time when the input signals, x1 and x2, change their values. This delay is due to the propagation delays in the logic element and the wires in the FPGA device.

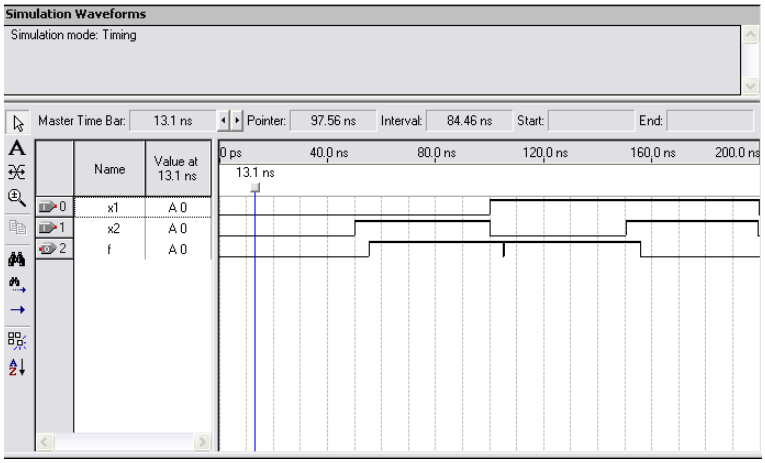


Figure 35. The result of timing simulation.

6- Pin Assignment

During the compilation above, the Quartus II Compiler was free to choose any pins on the selected FPGA to serve as inputs and outputs. However, the DE2 board has hardwired connections between the FPGA pins and the other components on the board. We will use two toggle switches, labeled **SW10** and **SW11**, to provide the external inputs, x1 and x2, to our example circuit. These switches are connected to the FPGA pins **W5** and **V10**, respectively. We will connect the output f to the red light-emitting diode labeled **LEDR10**, which is hardwired to the FPGA pin **AC13.**

Pin assignments are made by using the Assignment Editor. Select **Assignments > Assignment Editor** to reach the window in Figure 22.

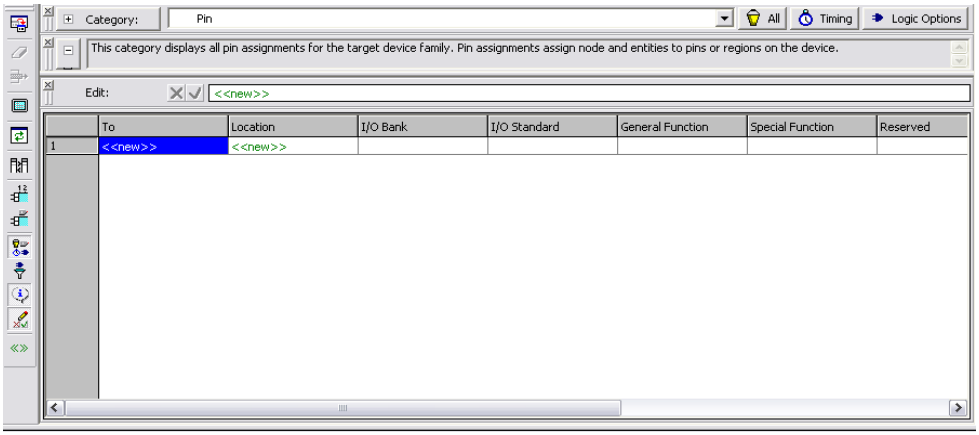


Figure 22. The Assignment Editor window.

Under Category select Pin. **Double-click on the entry <<new>>** which is highlighted in blue in the column labeled To. The drop-down menu in Figure 23 will appear. **Click on x1** as the first pin to be assigned; this will enter x1 in the displayed table. Follow this by **double-clicking** on the box to the right of this new x1 entry, in the column labeled **Location.** Now, the drop-down menu in Figure 24 appears.

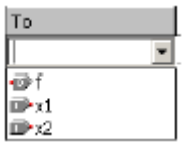


Figure 23. The drop-down menu displays the input and output names.

Scroll down and select **PIN_W5**. Instead of scrolling down the menu to find the desired pin, you can just **type the name of the pin (W5)** in the Location box. Use the same procedure to assign input x2 to pin V10 and output f to pin AC13, which results in the image in Figure 25.

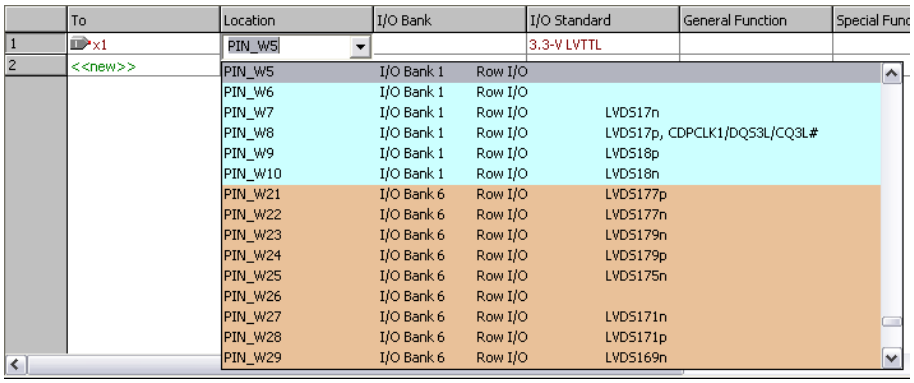


Figure 24. The available pins.

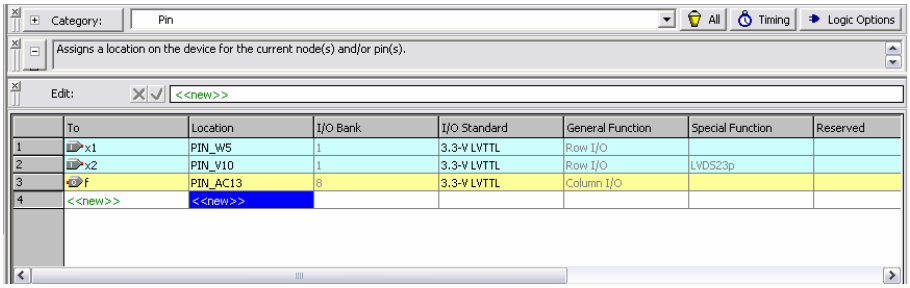


Figure 25. The complete assignment.

To save the assignments made, choose **File > Save**. You can also simply close the Assignment Editor window, in which case a pop-up box will ask if you want to save the changes to assignments; click Yes. **Recompile the circuit**, so that it will be compiled with the correct pin assignments.

7- Programming and Configuring the FPGA Device

The FPGA device must be programmed and configured to implement the designed circuit. The required configuration file is generated by the Quartus II Compiler's Assembler module. Altera's DE2 board allows the configuration to be done in two different ways, known as JTAG and AS modes. The configuration data is transferred from the host computer (which runs the Quartus II software) to the board by means of a cable that connects a USB port on the host computer to the leftmost USB connector on the board. To use this connection, it is necessary to have the USB-Blaster driver installed. Before using the board, make sure that the USB cable is properly connected and turn on the power supply switch on the board.

In the JTAG mode, the configuration data is loaded directly into the FPGA device. The acronym JTAG stands for Joint Test Action Group. This group defined a simple way for testing digital circuits and loading data into them, which became an IEEE standard. If the FPGA is configured in this manner, it will retain its configuration as long as the power remains turned on. The configuration information is lost when the power is turned off. The second possibility is to use the Active Serial (AS) mode. In this case, a configuration device that includes some flash memory is used to store the configuration data. Quartus II software places the configuration data into the configuration device on the DE2 board. Then, this data is loaded into the FPGA upon power-up or reconfiguration.

Thus, the FPGA need not be configured by the Quartus II software if the power is turned off and on. *The choice between the two modes is made by the RUN/PROG switch on the DE2 board. The RUN position selects the JTAG mode, while the PROG position selects the AS mode*.

7.1 JTAG Programming

The programming and configuration task is performed as follows. **Flip the RUN/PROG switch into the RUN position**. Select **Tools > Programmer** to reach the window in Figure 36. Here it is necessary to specify the programming hardware and the mode that should be used. If not already chosen by default, **select JTAG in the Mode box.**

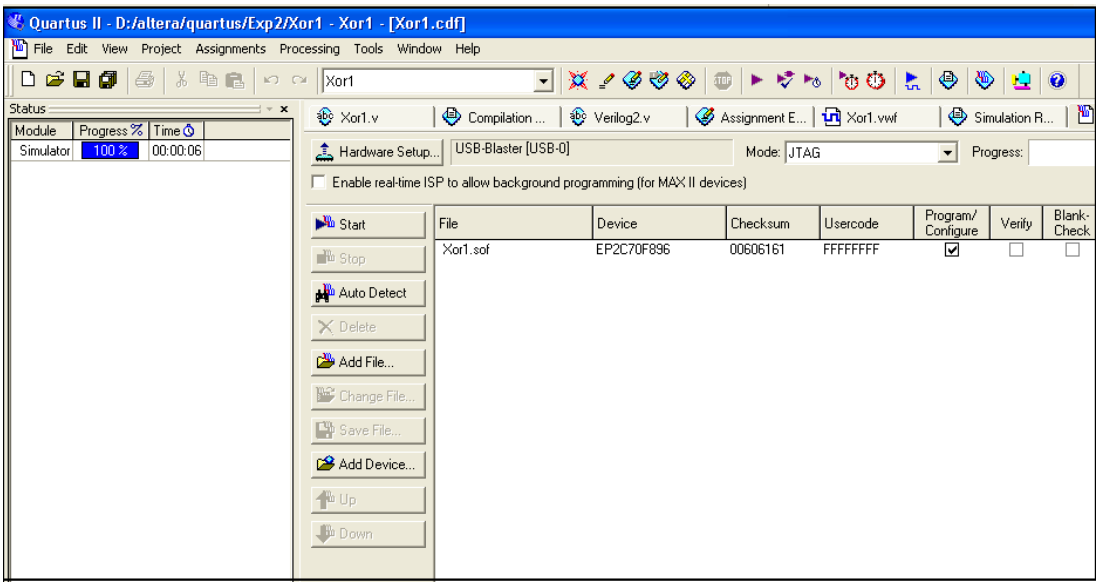


Figure 36. The Programmer window.

Also, if the USB-Blaster is not chosen by default, **press the Hardware Setup...** button and **select the USB-Blaster** in the window that pops up, as shown in Figure 37.

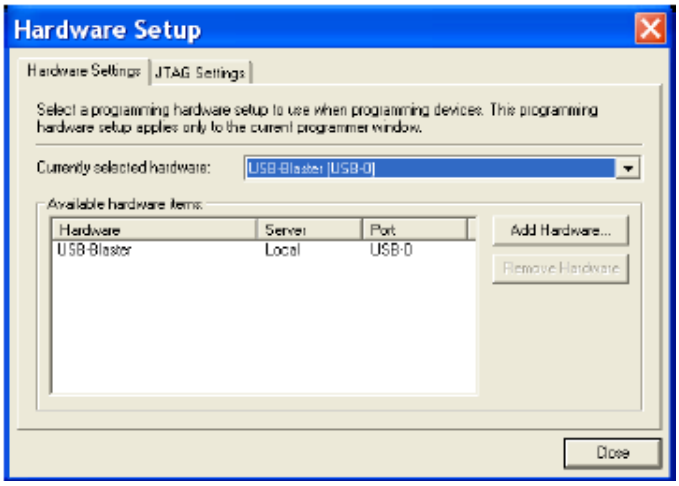


Figure 37. The Hardware Setup window.

In Figure 36, observe that the configuration file Xor1.sof is listed in the window. If the file is not already listed, then click **Add File** and select it. This is a binary file produced by the Compiler's Assembler module, which contains the data needed to configure the FPGA device. The extension .sof stands for SRAM Object File. Note also that the device selected is **EP2C70F896C6**, which is the FPGA device used on the DE2 board. **Click on the Program/Configure check box**. Now, **press Start button**. A LED on the board will light up when the configuration data has been downloaded successfully. If you see an error reported by Quartus II software indicating that programming failed, check to ensure that the board is properly powered on.