Quartus II Introduction Using Verilog Designs

1 Introduction

This tutorial presents an introduction to the Quartus® II CAD system. It gives a general overview of a typical CAD flow for designing circuits that are implemented by using FPGA devices, and shows how this flow is realized in the Quartus II software. The design process is illustrated by giving step-by-step instructions for using the Quartus II software to implement a very simple circuit in an Altera FPGA device.

The Quartus II system includes full support for all of the popular methods of entering a description of the desired circuit into a CAD system. This tutorial makes use of the Verilog design entry method, in which the user specifies the desired circuit in the Verilog hardware description language. Two other versions of this tutorial are also available; one uses the VHDL hardware description language and the other is based on defining the desired circuit in the form of a schematic diagram.

The last step in the design process involves configuring the designed circuit in an actual FPGA device. To show how this is done, it is assumed that the user has access to the Altera DE-series Development and Education board connected to a computer that has Quartus II software installed. A reader who does not have access to the DE-series board will still find the tutorial useful to learn how the FPGA programming and configuration task is performed.

The screen captures in the tutorial were obtained using the Quartus II version 11.0; if other versions of the software are used, some of the images may be slightly different.

Contents:

2 Background

Computer Aided Design (CAD) software makes it easy to implement a desired logic circuit by using a programmable logic device, such as a field-programmable gate array (FPGA) chip. A typical FPGA CAD flow is illustrated in Figure 1.

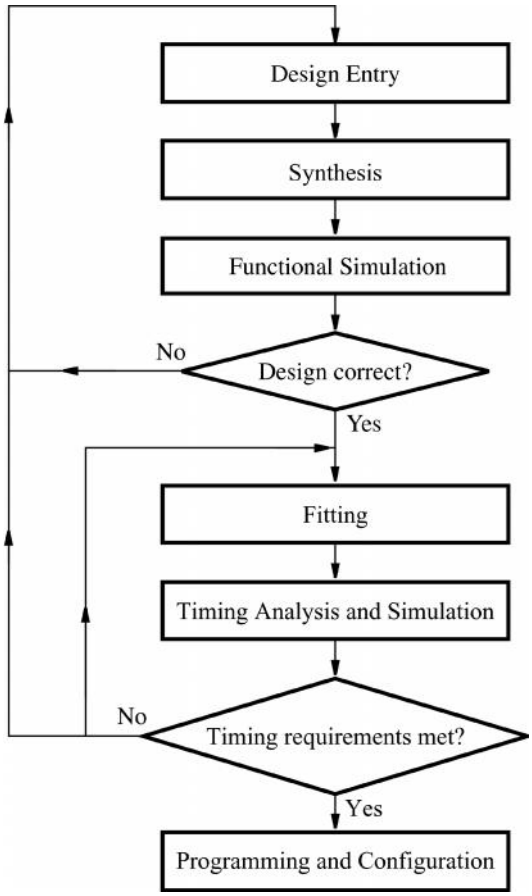


Figure 1. Typical CAD flow.

The CAD flow involves the following steps:

- Design Entry – the desired circuit is specified either by means of a schematic diagram, or by using a hardware description language, such as Verilog or VHDL
- Synthesis – the entered design is synthesized into a circuit that consists of the logic elements (LEs) provided in the FPGA chip
- Functional Simulation – the synthesized circuit is tested to verify its functional correctness; this simulation does not take into account any timing issues

- Fitting – the CAD Fitter tool determines the placement of the LEs defined in the netlist into the LEs in an actual FPGA chip; it also chooses routing wires in the chip to make the required connections between specific LEs
- Timing Analysis – propagation delays along the various paths in the fitted circuit are analyzed to provide an indication of the expected performance of the circuit
- Timing Simulation – the fitted circuit is tested to verify both its functional correctness and timing
- Programming and Configuration – the designed circuit is implemented in a physical FPGA chip by programming the configuration switches that configure the LEs and establish the required wiring connections

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 Verilog hardware description language. It makes use of the graphical user interface to invoke the Quartus II commands. Doing this tutorial, the reader will learn about:

- Creating a project
- Design entry using Verilog code
- Synthesizing a circuit specified in Verilog code
- Fitting a synthesized circuit into an Altera FPGA
- Assigning the circuit inputs and outputs to specific pins on the FPGA
- Simulating the designed circuit
- Programming and configuring the FPGA chip on Altera's DE-series board

3 Getting Started

Each logic circuit, or subcircuit, 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 tutorial, we will use a directory introtutorial. The running example for this tutorial is a simple circuit for two-way light control.

Start the Quartus II software. You should see a display similar to the one in Figure 2. 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. For example, in Figure 2 clicking the left mouse button on the menu named File opens the menu shown in Figure 3. Clicking the left mouse button on the entry Exit exits from Quartus II software. In general, whenever the mouse is used to select something, the left button is used. Hence we will not normally specify which button to press. In the few cases when it is necessary to use the right mouse button, it will be specified explicitly.

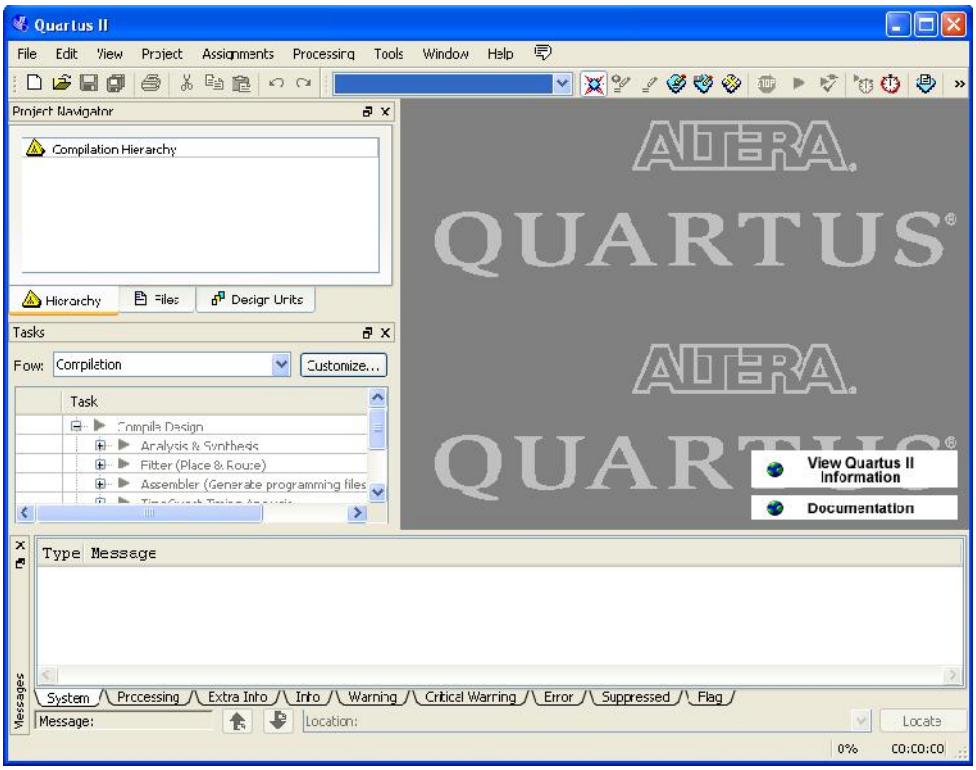


Figure 2. The main Quartus II display.

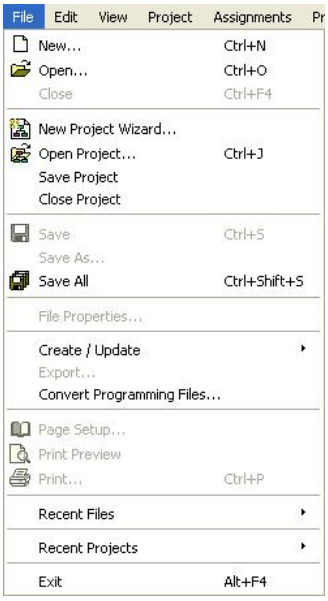


Figure 3. An example of the File menu.

For some commands it is necessary to access two or more menus in sequence. We use the convention Menu1 > Menu2 > Item to indicate that to select the desired command the user should first click the left mouse button on Menu1, then within this menu click on Menu2, and then within Menu2 click on Item. For example, File > Exit uses the mouse to exit from the system. Many commands can be invoked by clicking on an icon displayed in one of the toolbars. To see the command associated with an icon, position the mouse over the icon and a tooltip will appear that displays the command name.

3.1 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.

If no web browser is specified, Quartus will complain with an error message. To specify a web browser, go to Tools > Options... > General > Internet Connectivity. Specify a path to a web browser in the web browser field.

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.

4 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. Create a new project as follows:

1. Select File > New Project Wizard to reach the window in Figure4 , which asks for the name and directory of the project.

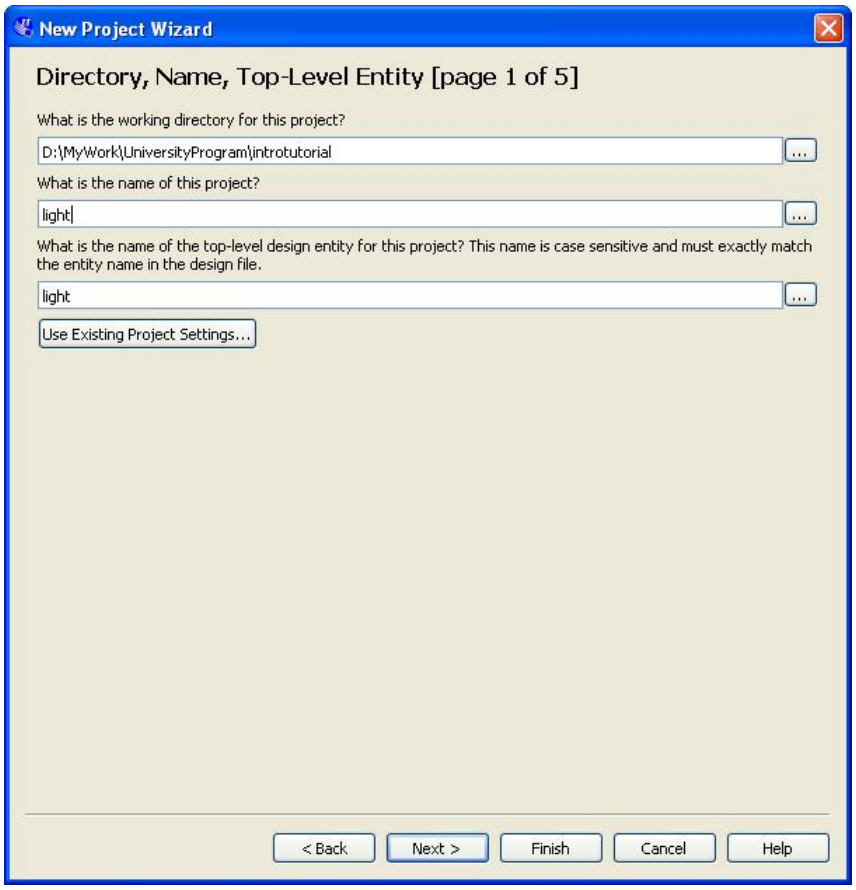


Figure 4. Creation of a new project.

2. Set the working directory to be introtutorial; of course, you can use some other directory name of your choice if you prefer. The project must have a name, which is usually the same as the top-level design entity that will be included in the project. Choose light as the name for both the project and the top-level entity, as shown in Figure 4. Press Next. Since we have not yet created the directory introtutorial, Quartus II software displays the pop-up box in Figure 5 asking if it should create the desired directory. Click Yes, which leads to the window in Figure 6.

Figure 5. Quartus II software can create a new directory for the project.

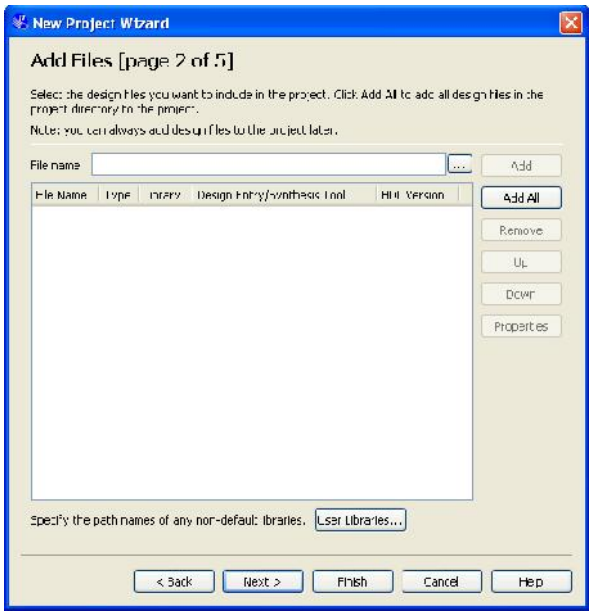


Figure 6. The wizard can include user-specified design files.

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 7.

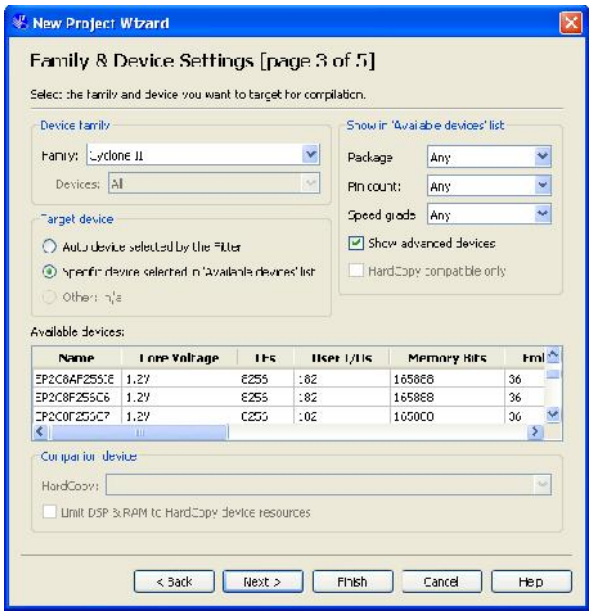


Figure 7. Choose the device family and a specific device.

4. We have to specify the type of device in which the designed circuit will be implemented. Choose the Cyclone-series device family for your DE-series board. 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 appropriate device name for your DE-series board. A list of devices names on DE-series boards can be found in Table 1. Press Next, which opens the window in Figure 8.

Board	Device Name
DE0	Cyclone III EP3C16F484C6
DE1	Cyclone II EP2C20F484C7
DE2	Cyclone II EP2C35F672C6
DE2-70	Cyclone II EP2C70F896C6
DE2-115	Cyclone IVE EP4CE115F29C7

Table 1. DE-series FPGA device names

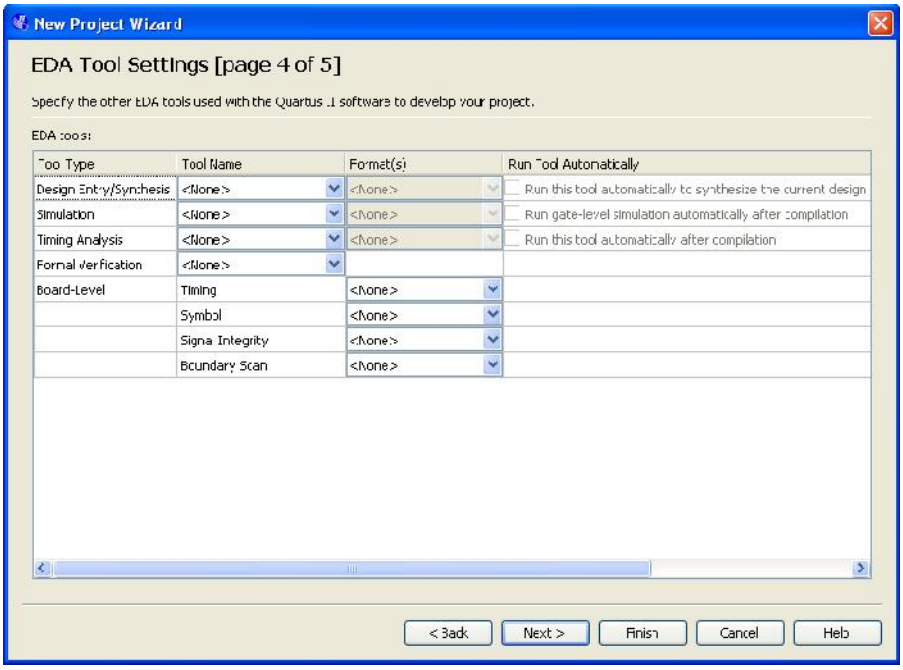


Figure 8. Other EDA tools can be specified.

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.

6. A summary of the chosen settings appears in the screen shown in Figure 9. Press Finish, which returns to the

main Quartus II window, but with light specified as the new project, in the display title bar, as indicated in Figure 10.

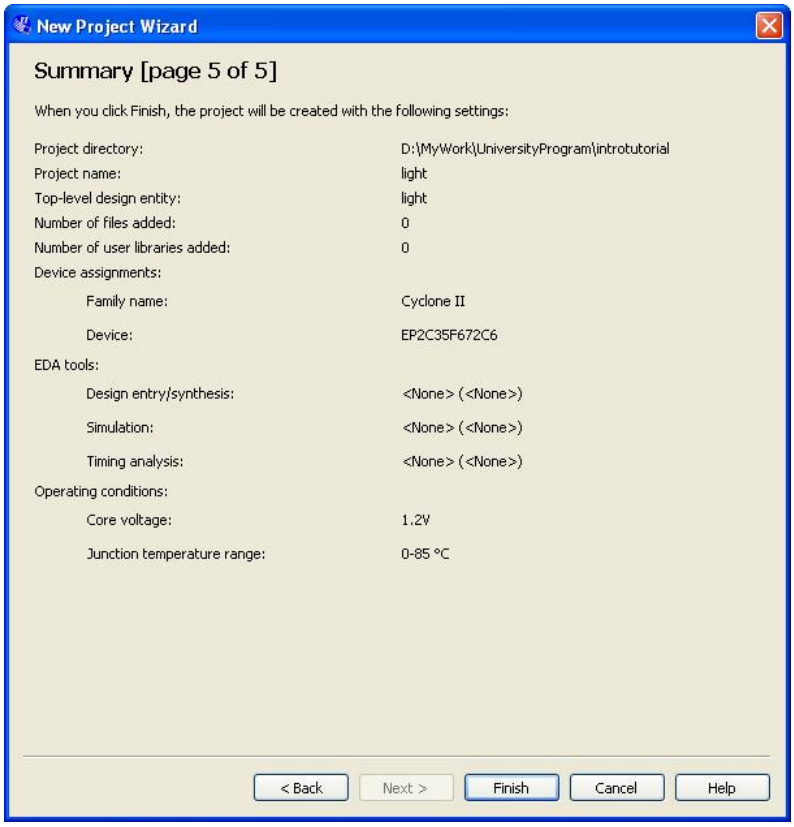


Figure 9. Example summary of a DE2 board project settings.

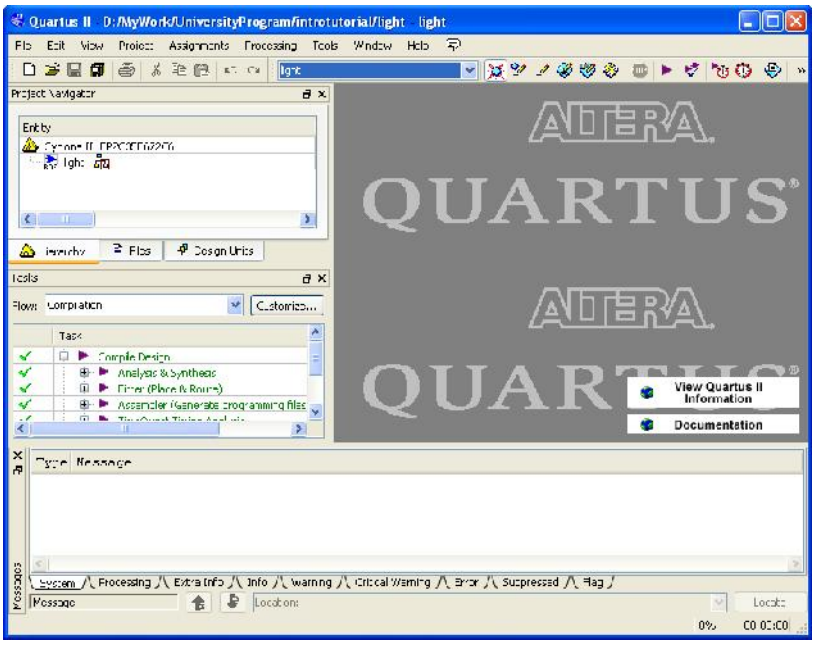
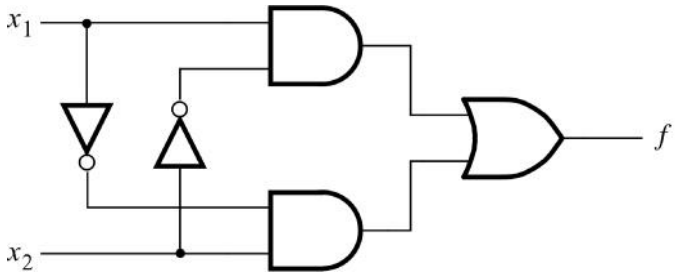


Figure 10. The Quartus II display for created project on a DE2 board.

5 Design Entry Using Verilog Code

As a design example, we will use the two-way light controller circuit shown in Figure 11. The circuit can be used to control a single light from either of the two switches, x_1 and x_2, where a closed switch corresponds to the logic value 1. The truth table for the circuit is also given in the figure. Note that this is just the Exclusive-OR function of the inputs x_1 and x_2, but we will specify it using the gates shown.



x_1	x_2	f
0	0	0
0	1	1
1	0	1
1	1	0

Figure 11. The light controller circuit.

The required circuit is described by the Verilog code in Figure 12. Note that the Verilog module is called light to match the name given in Figure 4, which was specified when the project was created. This code can be typed into a file by using any text editor that stores ASCII files, or by using the Quartus II text editing facilities. While the file

can be given any name, it is a common designers' practice to use the same name as the name of the top-level Verilog module. The file name must include the extension *v*, which indicates a Verilog file. So, we will use the name light.v.

```
module  light (x1, x2, f);
   input  x1, x2;
   output  f;
   assign  f = (x1 & ~x2) | (~x1 & x2);
endmodule
```

Figure 12. Verilog code for the circuit in Figure 11.

5.1 Using the Quartus II Text Editor

This section shows how to use the Quartus II Text Editor. You can skip this section if you prefer to use some other text editor to create the Verilog source code file, which we will name light.v.

Select File > New to get the window in Figure 13, choose Verilog HDL File, and click OK. This opens the Text Editor window. 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 14. In the box labeled Save as type choose Verilog HDL File. In the box labeled File name type light. Put a checkmark in the box Add file to current project. Click Save, which puts the file into the directory introtutorial and leads to the Text Editor window shown in Figure 15. Enter the Verilog code in Figure 12 into the Text Editor and save the file by typing File > Save, or by typing the shortcut Ctrl-s.

Most of the commands available in the Text Editor are self-explanatory. Text is entered at the insertion point, which is indicated by a thin vertical line. The insertion point can be moved either by using the keyboard arrow keys or by using the mouse. Two features of the Text Editor are especially convenient for typing Verilog code. First, the editor can display different types of Verilog statements in different colors, which is the default choice. Second, the editor can automatically indent the text on a new line so that it matches the previous line. Such options can be controlled by the settings in Tools > Options > Text Editor.

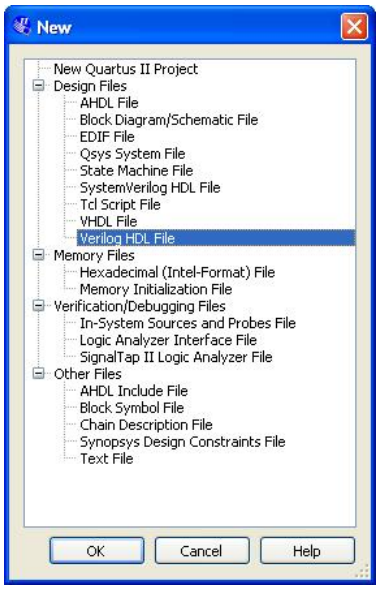


Figure 13. Choose to prepare a Verilog file.

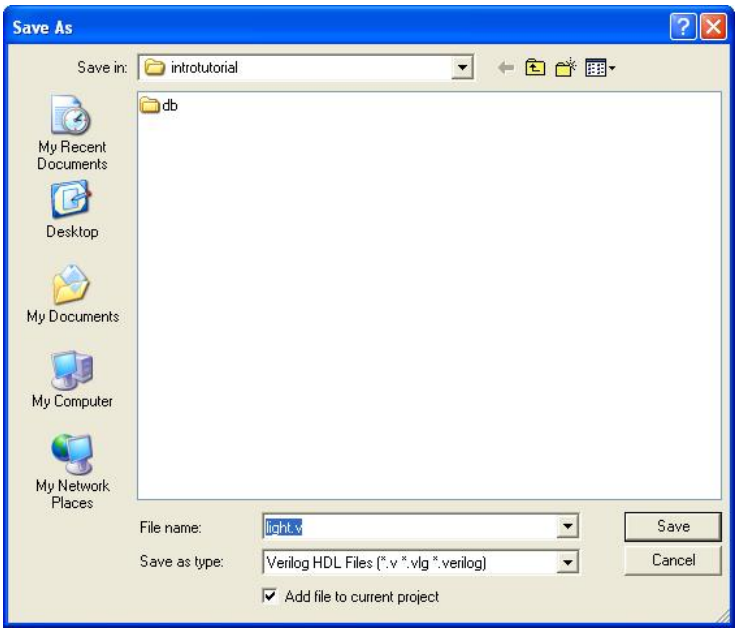


Figure 14. Name the file.

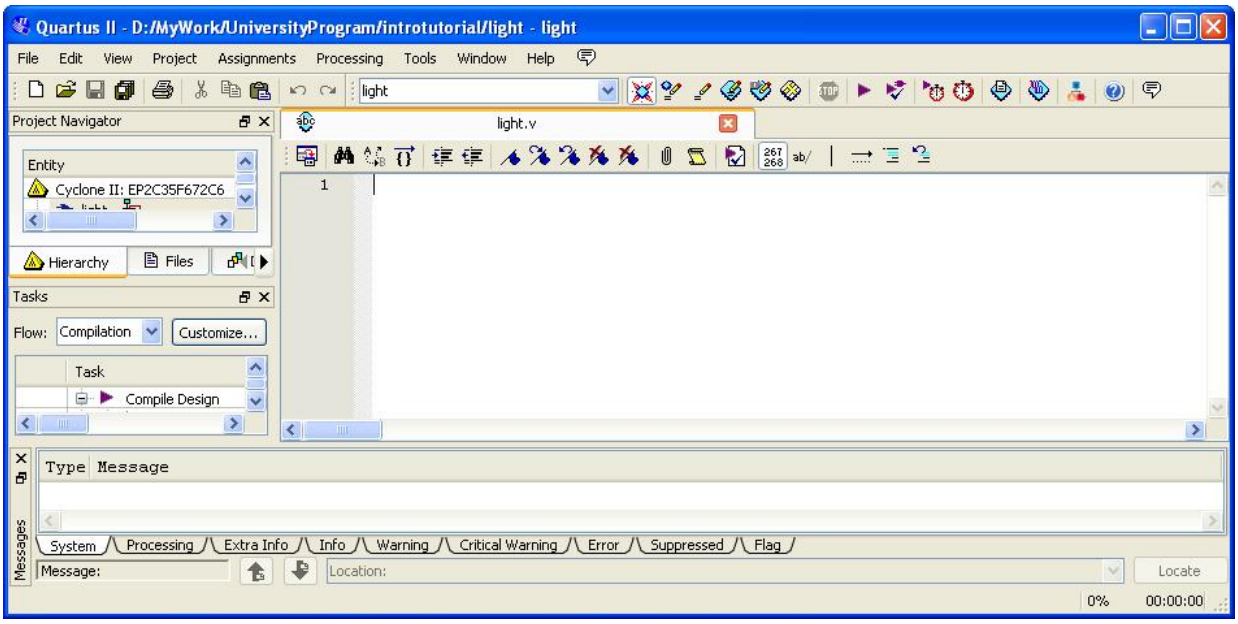


Figure 15. Text Editor window.

The syntax of Verilog code is sometimes difficult for a designer to remember. To help with this issue, the Text Editor provides a collection of Verilog templates. The templates provide examples of various types of Verilog statements, such as a module declaration, an always block, and assignment statements. It is worthwhile to browse through the templates by selecting Edit > Insert Template > Verilog HDL to become familiar with this resource.

5.2 Adding Design Files to a Project

As we indicated when discussing Figure 6, you can tell Quartus II software which design files it should use as part of the current project. To see the list of files already included in the light project, select Assignments > Settings, which leads to the window in Figure 16. As indicated on the left side of the figure, click on the item Files. An alternative way of making this selection is to choose Project > Add/Remove Files in Project.

If you used the Quartus II Text Editor to create the file and checked the box labeled Add file to current project, as described in Section 5.1, then the light.v file is already a part of the project and will be listed in the window in Figure 16. Otherwise, the file must be added to the project. So, if you did not use the Quartus II Text Editor, then place a copy of the file light.v, which you created using some other text editor, into the directory introtutorial. To add this file to the project, click on the File name: ... button in Figure 16 to get the pop-up window in Figure 17. Select the light.v file and click Open. The selected file is now indicated in the Files window of Figure 16. Click OK to include the light.v file in the project. We should mention that in many cases the Quartus II software is able to automatically find the right files to use for each entity referenced in Verilog code, even if the file has not been explicitly added to the project. However, for complex projects that involve many files it is a good design practice to specifically add the needed files to the project, as described above.

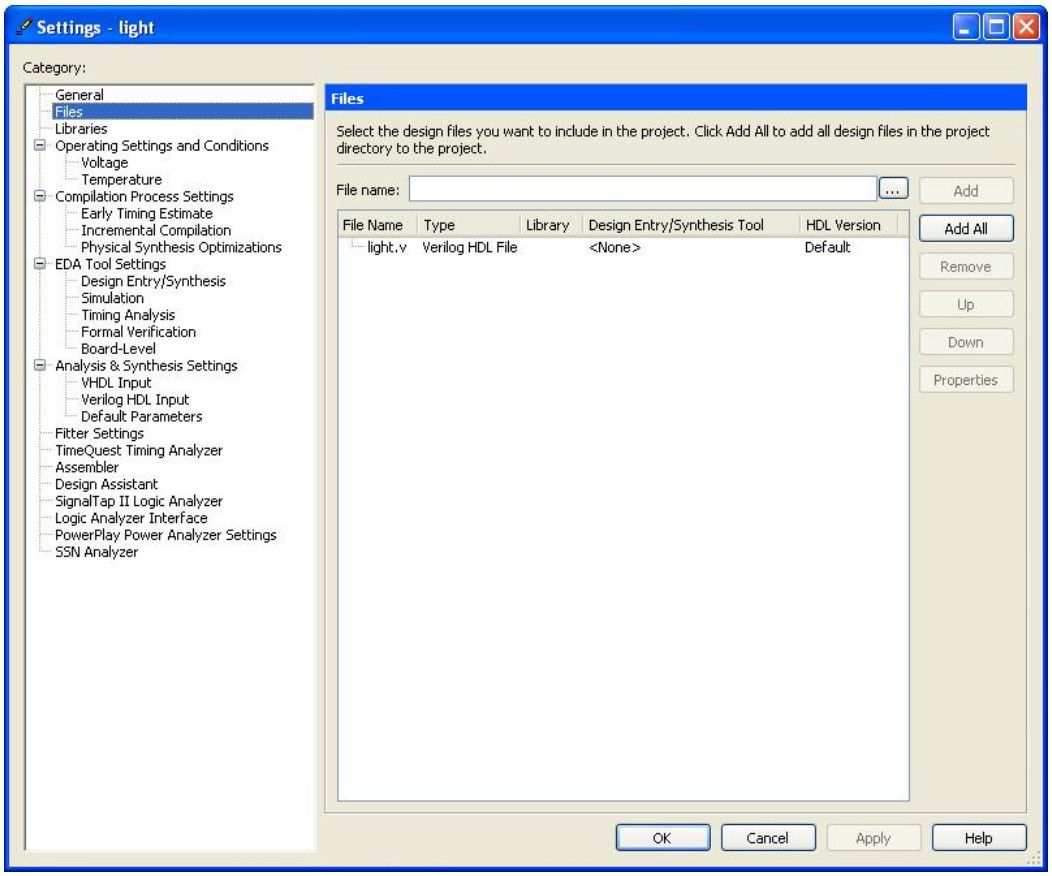


Figure 16. Settings window.

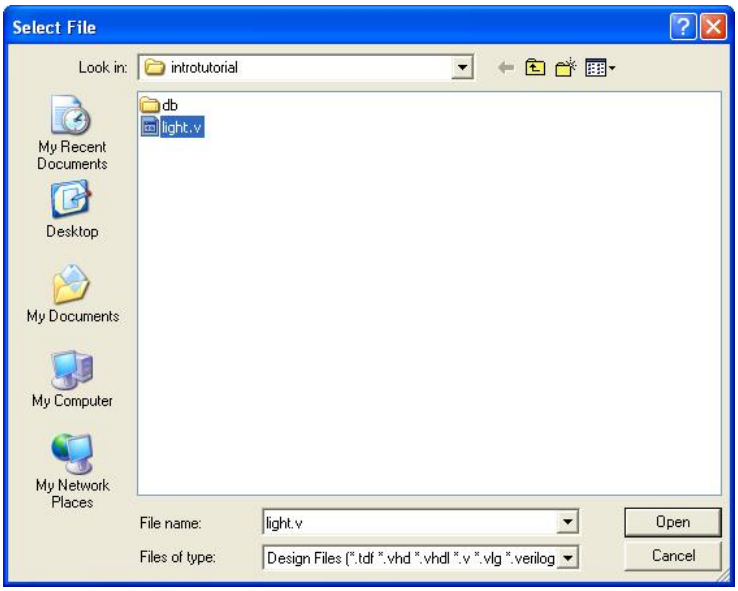


Figure 17. Select the file.

6 Compiling the Designed Circuit

The Verilog code in the file light.v is processed by several Quartus II tools that analyze the code, 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. Your project must be saved before compiling. 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.

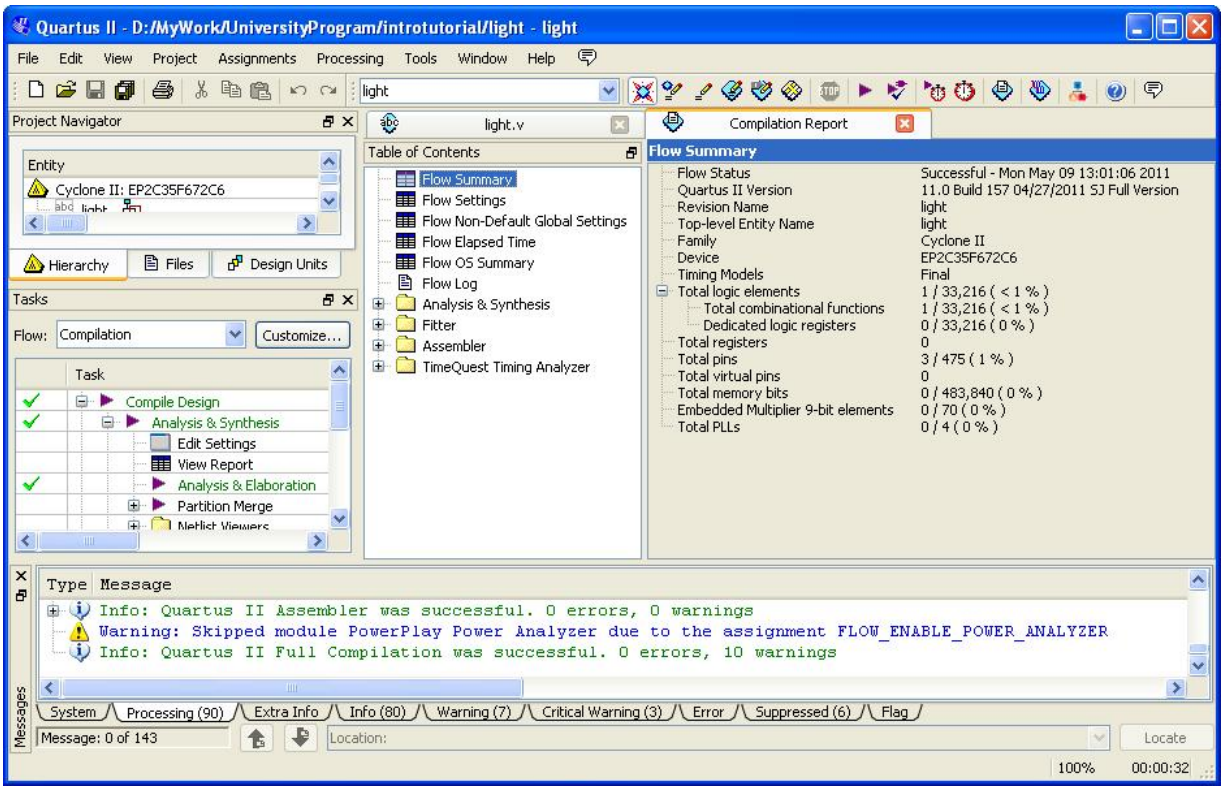


Figure 18. Display after a successful compilation.

When the compilation is finished, a compilation report is produced. A tab showing this report is opened automatically, as seen in Figure 18. The tab can be 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. 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.

6.1 Errors

Quartus II software displays messages produced during compilation in the Messages window. If the Verilog design file is correct, one of the messages will state that the compilation was successful and that there are no errors.

If the Compiler does not report zero errors, then there is at least one mistake in the Verilog code. In this case a message corresponding to each error found will be displayed in the Messages window. Double-clicking on an error message will highlight the offending statement in the Verilog code in the Text Editor window. Similarly, the Compiler may display some warning messages. Their details can be explored in the same way as in the case of error messages. The user can obtain more information about a specific error or warning message by selecting the message and pressing the F1 function key.

To see the effect of an error, open the file light.v. Remove the semicolon in the assign statement, illustrating a

typographical error that is easily made. Compile the erroneous design file by clicking on the icon. A pop-up box will ask if the changes made to the light.v file should be saved; click Yes. After trying to compile the circuit, Quartus II software will display a pop-up box indicating that the compilation was not successful. Acknowledge it by clicking OK. The compilation report summary, given in Figure 19, now confirms the failed result. In the Table of Contents panel, expand the Analysis & Synthesis part of the report and then select Messages to have the messages displayed as shown in Figure 20. The Compilation Report can be dispayed as a separate window as in Figure 20 by right-clicking its tab and selecting Detach Window, and can be reattached by clicking Window > Attatch Window. Double-click on the first error message. Quartus II software responds by opening the light.v file and highlighting the statement which is affected by the error, as shown in Figure 21. Correct the error and recompile the design.

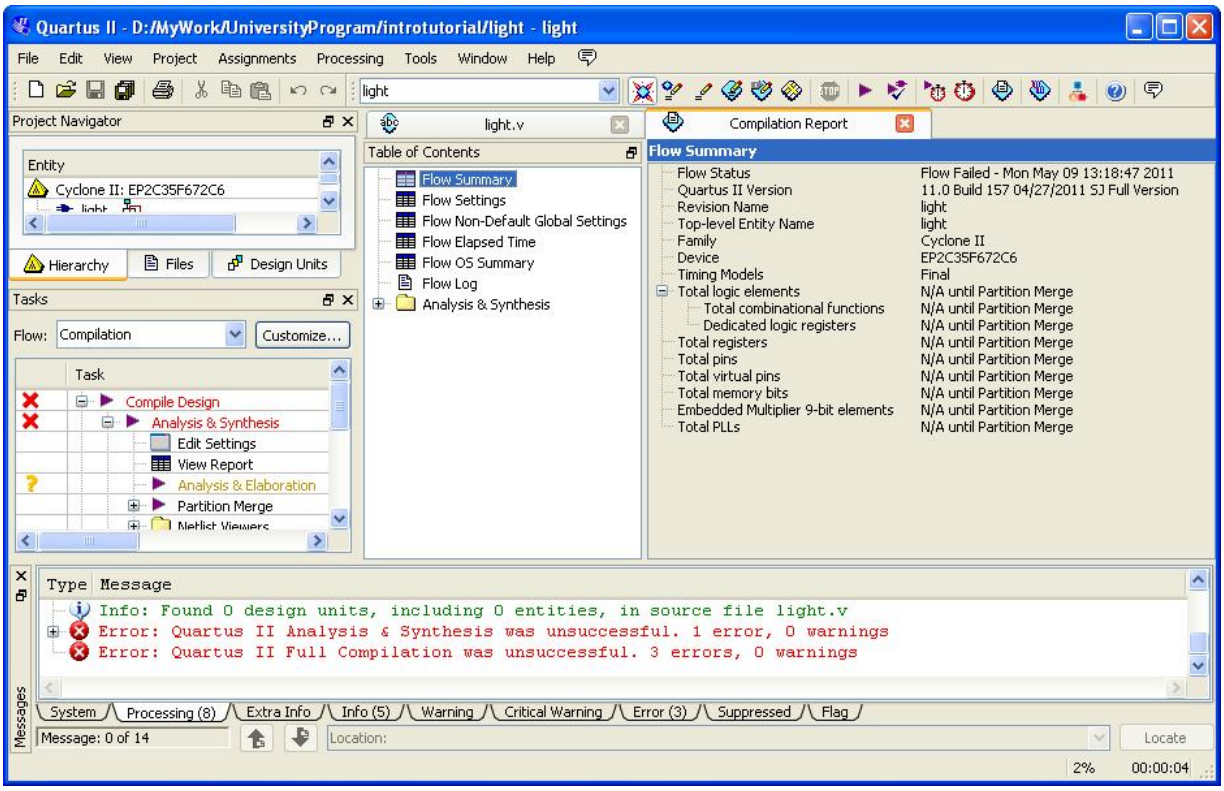


Figure 19. Compilation report for the failed design.

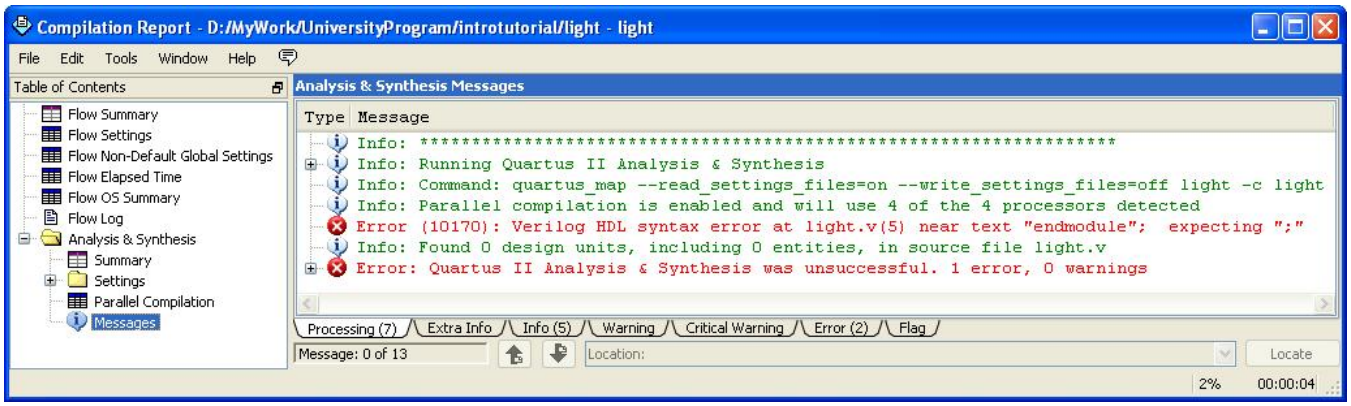


Figure 20. Error messages.

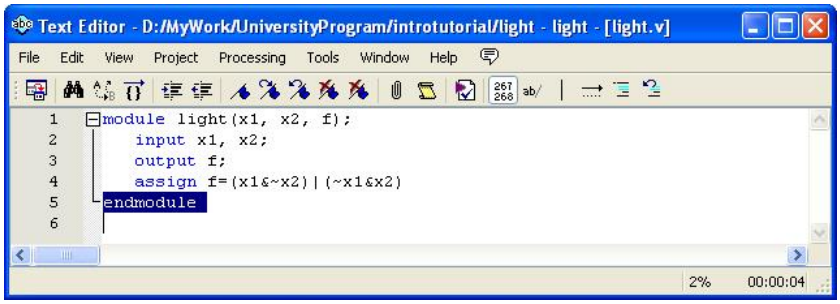


Figure 21. Identifying the location of the error.

7 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 DE-series board has hardwired connections between the FPGA pins and the other components on the board. We will use two toggle switches, labeled SW_0 and SW_1, to provide the external inputs, x_1 and x_2, to our example circuit. These switches are connected to the FPGA pins listed in Table 2. We will connect the output f to the green light-emitting diode labeled $LEDG_0$. Its FPGA pin assignment can also be found in Table 2.

Component	DE0	DE1	DE2	DE2-70	DE2-115
SW_0	PIN_J6	PIN_L22	PIN_N25	PIN_AA23	PIN_AB28
SW_1	PIN_H5	PIN_L21	PIN_N26	PIN_AB26	PIN_AC28
$LEDG_0$	PIN_J1	PIN_U22	PIN_AE22	PIN_W27	PIN_E21

Table 2. DE-Series Pin Assignments

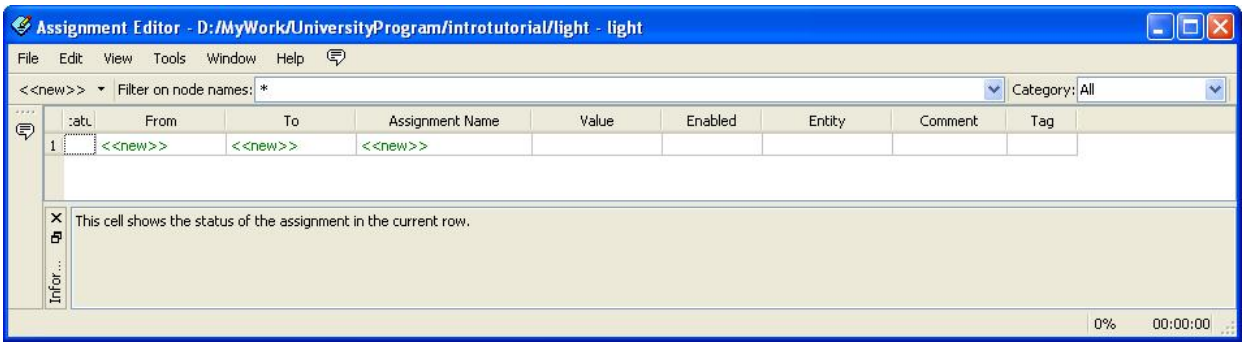


Figure 22. The Assignment Editor window.

Pin assignments are made by using the Assignment Editor. Select Assignments > Assignment Editor to reach the window in Figure 22 (shown here as a detached window). In the Category drop-down menu select All. Click on the <<new>> button located near the top left corner to make a new item appear in the table. Double click the box under the column labeled To so that the Node Finder button appears. Click on the button (not the drop down arrow) to reach the window in Figure 23. In the Filter drop-down menu select Pins: all. Then click the List button to display the input and output pins to be assigned: f, $x1$, and $x2$. Click on $x1$ as the first pin to be assigned and click the > button; this will enter $x1$ in the Selected Nodes box. Click OK. $x1$ will now appear in the box under the column labeled To. Alternatively, the node name can be entered directly by double-clicking the box under the To column and typing in the node name.

Follow this by double-clicking on the box to the right of this new $x1$ entry, in the column labeled Assignment Name. Now, the drop-down menu in Figure 24 appears. Scroll down and select Location (Accepts wildcards/groups). Instead of scrolling down the menu to find the desired item, you can just type the first letter of the item in the Assignment Name box. In this case the desired item happens to be the first item beginning with L. Finally, double-click the box in the column labeled Value. Type the pin assignment corresponding to SW_0 for your DE-series board, as listed in Table 2.

Use the same procedure to assign input $x2$ and output f to the appropriate pins listed in Table 2. An example using a DE2 board is shown in Figure 25. 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.

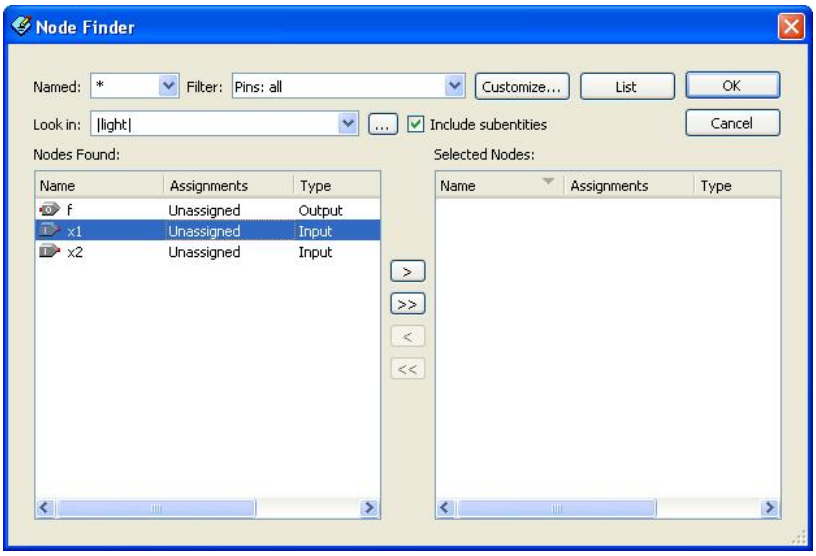


Figure 23. The Node Finder displays the input and output names.

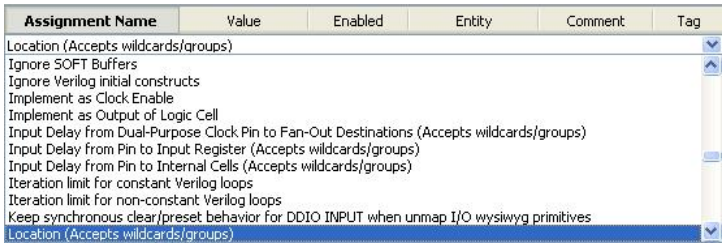


Figure 24. The available assignment names for a DE2 board.

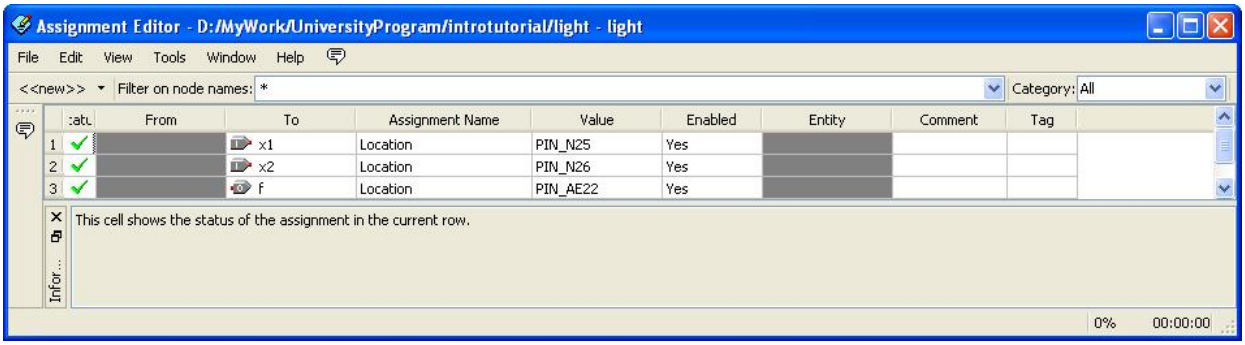


Figure 25. The complete assignment on a DE2 board.

The DE-series board has fixed pin assignments. Having finished one design, the user will want to use the same pin assignment for subsequent designs. Going through the procedure described above becomes tedious if there are many pins used in the design. A useful Quartus II feature allows the user to both export and import the pin assignments from a special file format, rather than creating them manually using the Assignment Editor. A simple file format that can be used for this purpose is the Quartus II Settings File (QSF) format. The format for the file for our simple project (on a DE2 board) is

```
set_location_assignment  PIN_N25   -to  x1
set_location_assignment  PIN_N26   -to  x2
set_location_assignment  PIN_AE22 -to  f
```

By adding lines to the file, any number of pin assignments can be created. Such qsf files can be imported into any design project.

If you created a pin assignment for a particular project, you can export it for use in a different project. To see how this is done, open again the Assignment Editor to reach the window in Figure 26. Select Assignments > Export Assignment which leads to the window in Figure 26. Here, the file light.qsf is available for export. Click on OK. If you now look in the directory, you will see that the file light.qsf has been created.

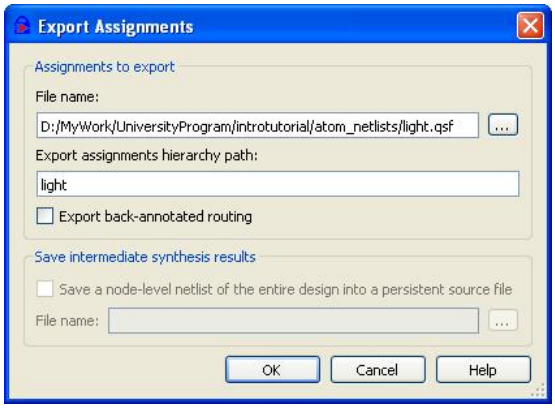


Figure 26. Exporting the pin assignment.

You can import a pin assignment by choosing Assignments > Import Assignments. This opens the dialogue in Figure 27 to select the file to import. Type the name of the file, including the qsf extension and the full path to the directory that holds the file, in the File Name box and press OK. Of course, you can also browse to find the desired file.

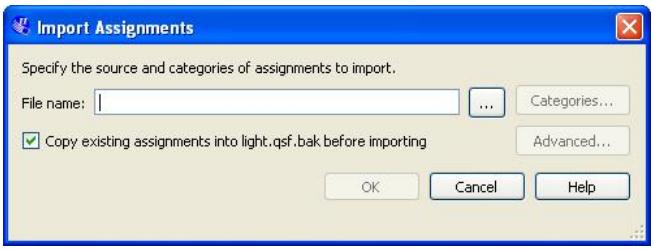


Figure 27. Importing the pin assignment.

For convenience when using large designs, all relevant pin assignments for the DE-series board are given in individual files. For example, the DE2 pin assignments can be found in the DE2_pin_assignments.qsf file, in the directory tutorials\design_files, which is included on the CD-ROM that accompanies the DE-series board and can also be found on Altera's DE-series web pages. This file uses the names found in the DE2 User Manual. If we wanted to make the pin assignments for our example circuit by importing this file, then we would have to use the same names in our Verilog design file; namely, SW[0], SW[1] and LEDG[0] for x1, x2 and f, respectively. Since these signals are specified in the qsf file as elements of vectors SW and LEDG, we must refer to them in the same way in the Verilog design file. For example, in the qsf file the 18 toggle switches are called SW[17] to SW[0]. In Verilog code, they can also be referred to as a vector SW[17:0].

8 Simulating the Designed Circuit

Omit this section

9 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 DE-series 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. If this driver is not already installed, consult the tutorial Getting Started with Altera's DE-Series Boards for information about installing the driver. 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 DE-series 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 DE-series board. The RUN position selects the JTAG mode, while the PROG position selects the AS mode.

9.1 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 38. 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. 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 39.

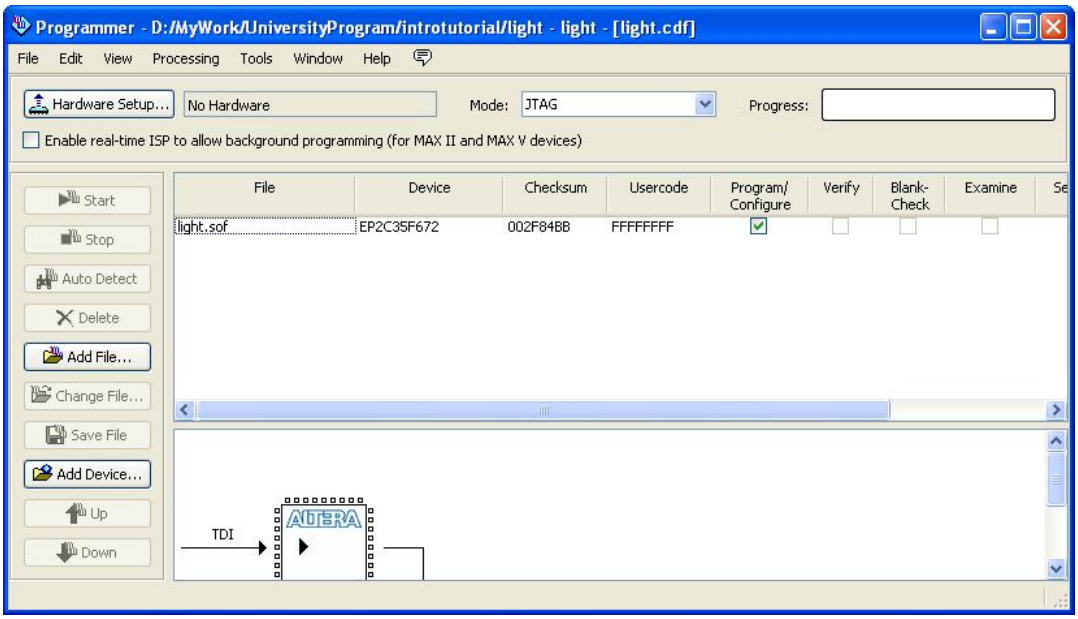


Figure 38. The Programmer window.

Observe that the configuration file light.sof is listed in the window in Figure 38. 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 EP2C35F672, which is the FPGA device used on the DE2 board. Click on the Program/Configure check box, as shown in Figure 40.

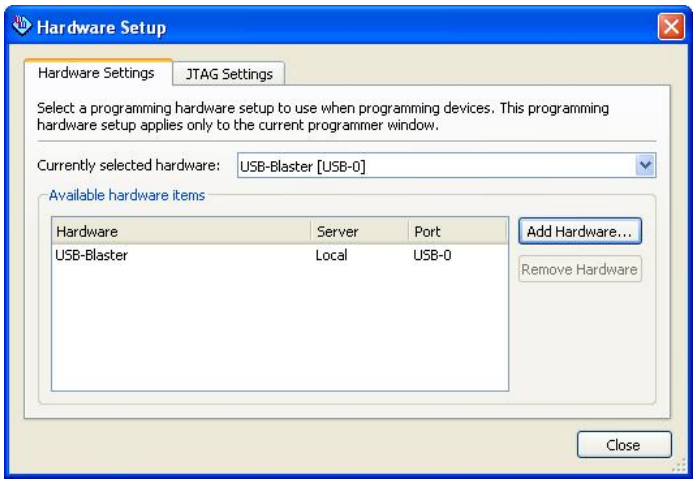


Figure 39. The Hardware Setup window.

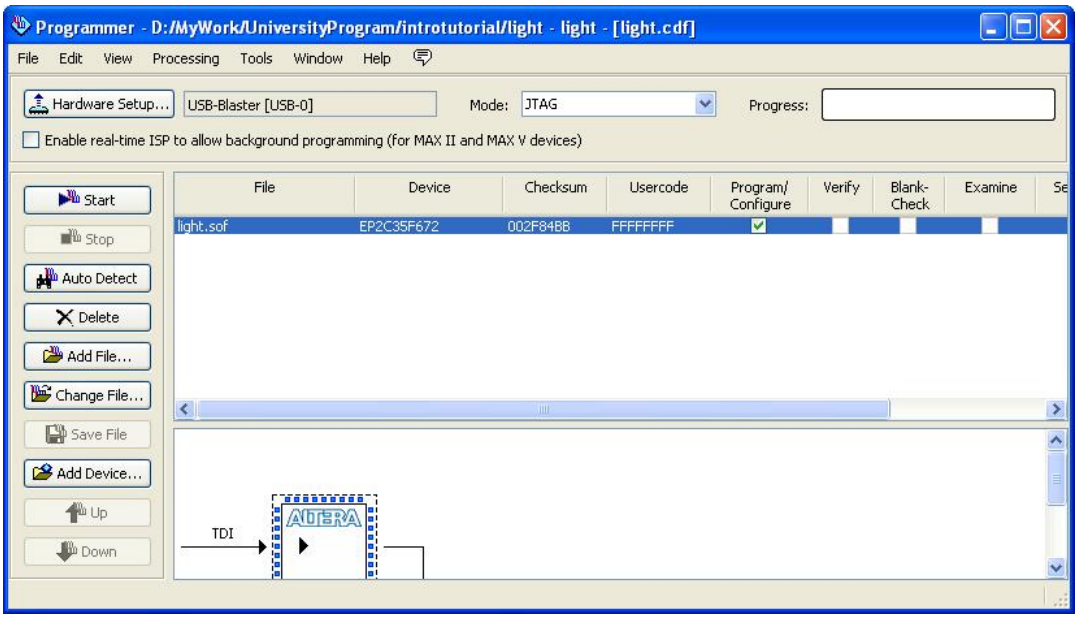


Figure 40. The updated Programmer window.

Now, press Start in the window in Figure 40. An 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, then check to ensure that the board is properly powered on.

9.2Active Serial Mode Programming

Omit this section

10 Testing the Designed Circuit

Having downloaded the configuration data into the FPGA device, you can now test the implemented circuit. Flip the RUN/PROG switch to RUN position. Try all four valuations of the input variables x_1 and x_2, by setting the corresponding states of the switches SW_1 and SW_0. Verify that the circuit implements the truth table in Figure 11.

If you want to make changes in the designed circuit, first close the Programmer window. Then make the desired changes in the Verilog design file, compile the circuit, and program the board as explained above.