

# Quartus<sup>®</sup> Prime Introduction Using VHDL Designs

### For Quartus<sup>®</sup> Prime 18.1

# Contents

1	Introduction	2
2	Background	3
3	Getting Started	4
	3.1 Quartus <sup>®</sup> Prime Online Help	6
4	Starting a New Project	6
5	Design Entry Using VHDL Code	13
	5.1 Using the Quartus Prime Text Editor	14
	5.1.1 Using VHDL Templates	16
	5.2 Adding Design Files to a Project	16
6	Compiling the Designed Circuit	18
	6.1 Errors	19
7	Pin Assignment	22
8	Programming and Configuring the FPGA Device	26
	8.1 JTAG* Programming for the DE0-CV, DE0-Nano, DE10-Lite, and DE2-115 Boards	26
	8.2 JTAG* Programming for the DE0-Nano-SoC, DE1-SoC Board, DE10-Nano, and DE10-Standard	28
9	Testing the Designed Circuit	30

### 1 Introduction

This tutorial presents an introduction to the Quartus<sup>®</sup> Prime 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 Prime software. The design process is illustrated by giving step-by-step instructions for using the Quartus Prime software to implement a very simple circuit in an Intel<sup>®</sup> FPGA device.

The Quartus Prime 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 VHDL design entry method, in which the user specifies the desired circuit in the VHDL hardware description language. Three versions of this tutorial are available; one uses the Verilog hardware description language, another uses the VHDL hardware description language, and the third 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 Intel DE-series Development and Education board connected to a computer that has Quartus Prime 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 Prime version 18.1 Standard Edition; other versions of the software may be slightly different.

### 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.

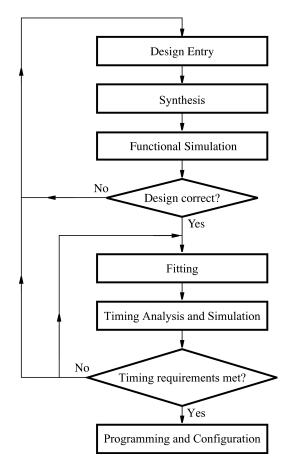


Figure 1. Typical CAD flow.

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 Prime software. It shows how the software can be used to design and implement a circuit specified by using the VHDL hardware description language. It makes use of the graphical user interface to invoke the Quartus Prime commands. Doing this tutorial, the reader will learn about:

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

# 3 Getting Started

Each logic circuit, or subcircuit, being designed with Quartus Prime 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 Prime 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 Prime software, which the user selects with the computer mouse. Most of the commands provided by Quartus Prime 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 Prime 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.

Quartus Prime Standard Edition File Edit View Project Assignments Proce	orcing Tools Window I	lala				- 🗆 X
	-	===p === ⊨ ⊨ ⊨ ⊖   ⊕ ♣ ¥				Search altera.com
Project Navigator A Hierarchy		ome 🛛			IP Catalog	- 8 × 😭
Compilation Hierarchy	Recent Projects				Device Family Cyclone V	(E/GX/GT/SX/SE/ST)
		New Project Wizard	Open Project		<ul> <li>Project Directory</li> <li>No Selection Avail</li> <li>Library</li> <li>Basic Functions</li> </ul>	lable
	Documentation	Training S	Support What's New	Notifications	DSP     Interface Protocol     Memory Interface     Processors and Pe     University Program     Search for Partner IF	s and Controllers eripherals m
Tasks     Compilation     ▼     ■ ■ ● ● ×       ✓     ▶     Compile Design     ↑       >     ▶     Analysis & Synthesis     >       >     ▶     Fitter (Place & Route)     ↓       >     ▶     Fitter (Place & Route)     ↓						
FimeQuest Timing Analysis     EDA Netlist Writer      Frit Sertinge	Close page after project			(intel)	+ Add	
All     Image: Constraint of the second	ne Intel FPGA websi	te to check for up				>
Messages						0% 00:00:00

Figure 2. The main Quartus Prime display.

	<u>N</u> ew	Ctrl+N
	Open	Ctrl+O
	Close	Ctrl+F4
-	New Project Wizard	
1	Open Project	Ctrl+J
	Save Project	
	Clos <u>e</u> Project	
	Save	Ctrl+S
	Save As	
습권	Save All	Ctrl+Shift+S
	<u>File Properties</u>	
	Create / Update	•
	Export	
	Convert Programming Files	
Ħ	Page Setup	
	Print Preview	
÷	Print	Ctrl+P
	Recent Files	•
	Recent Projects	•
	E <u>x</u> it	Alt+F4

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 the command name will be shown in the status bar at the bottom of the screen.

### 3.1 Quartus<sup>®</sup> Prime Online Help

Quartus Prime 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.

The user can quickly search through the Help topics by using the search box in the top right corner of the main Quartus display. 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 Prime 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 and click Next to reach the window in Figure 4, which asks for the name and directory of the 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 Prime 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.

New Project Wizard	×
Directory, Name, Top-Level Entity	
What is the working directory for this project?	
C:/Desktop/introtutorial	
What is the name of this project?	
light	
What is the name of the top-level design entity for this project? This name is case sensitive and must exact match the entity name in the design file.	tly
light	
Use Existing Project Settings	
< Back Next > Finish Cancel He	lp

Figure 4. Creation of a new project.

🕥 Qua	rtus Prime	×
	Directory "C:/Desktop/introtutorial" does no want to create it?	ot exist. Do you
	<u>Y</u> es	No

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

🕥 New Project Wizard	×
Project Type	
Select the type of project to create.	
Empty project	
Create new project by specifying project files and libraries, target device family and device, and EDA too settings.	ι
O Project template	
Create a project from an existing design template. You can choose from design templates installed with the Quartus Prime software, or download design templates from the <u>Design Store</u> .	
< <u>B</u> ack <u>N</u> ext > Einish Cancel <u>H</u> elp	

Figure 6. Choosing the project type.

3. The Project Type window, shown in Figure 6, allows you to choose from the Empty project and the Project template options. For this tutorial, choose Empty project as we will be creating a project from scratch, and press Next which leads to the window in Figure 7.

S New Project Wizard		×
Add Files		
Select the design files you want to include in the project. Click Add All to add all design files directory to the project. Note: you can always add design files to the project later.	in the	e project
Eile name:		<u>A</u> dd
<b>K</b>	×	Add A <u>l</u> l
File Name Type Library Design Entry/Synthesis Tool HDL Version		<u>R</u> emove
		<u>U</u> p
		<u>D</u> own
		<u>P</u> roperties
Specify the path names of any non-default libraries. User Libraries		
< <u>B</u> ack <u>N</u> ext > <u>F</u> inish Canc	el	<u>H</u> elp

Figure 7. The wizard can include user-specified design files.

4. 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 8.

Select the family and								
(ou can install addi	d device you want to	target for co	mpilation.					
i ou cun motuit duur	tional device support	with the Inst	all Devices comm	and on the Tool	s me	inu.		
Fo determine the ve	ersion of the Quartus	Prime softwa	re in which your	target device is s	supp	orted, refer to the	<u>Device Support List</u> v	vebpa
Device family				Show in 'Availal	ble d	levices' list		
Family: Cyclone V (E/GX/GT/SX/SE/ST)								
	(E/GX/G1/SX/SE/S1	)	·	Pac <u>k</u> age:		Any		•
Dev <u>i</u> ce: All			•	Pin <u>c</u> ount:		Any		•
Target device				Core sp <u>e</u> ed gra	de:	Any		-
	lected by the Fitter			Name filter:	Γ			
	-				L			
	selected in 'Available	devices' list		✓ Show advan	nced	devices		
○ <u>O</u> ther: n/a								
A <u>v</u> ailable devices:								
A <u>v</u> ailable devices: Name	Core Voltage	ALMs	Total I/Os	GPIOs	GX	B Channel PMA	GXB Channel PC	s
	Core Voltage	<b>ALMs</b> 15880	Total I/Os 314	<b>GPIOs</b> 314 0		B Channel PMA	GXB Channel PC	s
Name	-				)	(B Channel PMA		s
Name 5CSEMA4U23C8	1.1V	15880	314	314 0	)	(B Channel PMA	0	s
Name 5CSEMA4U23C8 5CSEMA4U23I7	1.1V 1.1V	15880 15880	314 314	314 0 314 0	)	(B Channel PMA	0	S
Name 5CSEMA4U23C8 5CSEMA4U23I7 5CSEMA5F31A7	1.1V 1.1V 1.1V	15880 15880 32070	314 314 457	314         0           314         0           457         0	) ) )	(B Channel PMA	0 0 0	s
Name           5CSEMA4U23C8           5CSEMA4U23I7           5CSEMA5F31A7           5CSEMA5F31C6	1.1V 1.1V 1.1V 1.1V 1.1V	15880 15880 32070 32070	314 314 457 457	314         0           314         0           457         0           457         0	) ) ) )	(B Channel PMA	0 0 0 0	:s
Name           5CSEMA4U23C8           5CSEMA4U23I7           5CSEMA5F31A7           5CSEMA5F31C6           5CSEMA5F31C7	1.1V 1.1V 1.1V 1.1V 1.1V 1.1V	15880 15880 32070 32070 32070	314 314 457 457 457	314     0       314     0       457     0       457     0       457     0	) ) ) ) )	(B Channel PMA	0 0 0 0 0	s
Name           5CSEMA4U23C8           5CSEMA4U23I7           5CSEMA5F31A7           5CSEMA5F31C6           5CSEMA5F31C7           5CSEMA5F31C8	1.1V 1.1V 1.1V 1.1V 1.1V 1.1V 1.1V 1.1V	15880 15880 32070 32070 32070 32070	314 314 457 457 457 457	314     0       314     0       457     0       457     0       457     0       457     0       457     0	) ) ) ) ) ) )	(B Channel PMA	0 0 0 0 0 0 0	s
Name           5CSEMA4U23C8           5CSEMA4U23I7           5CSEMA5F31A7           5CSEMA5F31C6           5CSEMA5F31C7           5CSEMA5F31C8           5CSEMA5F31C8           5CSEMA5F31C8	1.1V 1.1V 1.1V 1.1V 1.1V 1.1V 1.1V 1.1V	15880 15880 32070 32070 32070 32070 32070	314 314 457 457 457 457 457	314     0       314     0       457     0       457     0       457     0       457     0       457     0       457     0       457     0       457     0       457     0	) ) ) ) )	(B Channel PMA	0 0 0 0 0 0 0 0 0	s
Name           5CSEMA4U23C8           5CSEMA4U23I7           5CSEMA5F31A7           5CSEMA5F31C6           5CSEMA5F31C7           5CSEMA5F31C8           5CSEMA5F31C8           5CSEMA5F31C8           5CSEMA5F31C8           5CSEMA5F31C8	1.1V       1.1V	15880 15880 32070 32070 32070 32070 32070 32070 32070	314 314 457 457 457 457 457 314	314     0       314     0       457     0       457     0       457     0       457     0       457     0       457     0       314     0	) ) ) ) ) )	(B Channel PMA	0 0 0 0 0 0 0 0 0 0 0 0	:S

Figure 8. Choose the device family and a specific device.

5. We have to specify the type of device in which the designed circuit will be implemented. Choose the Cyclone<sup>®</sup> series device family for your DE-series board. We can let Quartus Prime 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 9.

Board	Device Name
DE0-CV	Cyclone V 5CEBA4F23C7
DE0-Nano	Cyclone IVE EP4CE22F17C6
DE0-Nano-SoC	Cyclone V SoC 5CSEMA4U23C6
DE1-SoC	Cyclone V SoC 5CSEMA5F31C6
DE2-115	Cyclone IVE EP4CE115F29C7
DE10-Lite	Max 10 10M50DAF484C7G
DE10-Standard	Cyclone V SoC 5CSXFC6D6F31C6
DE10-Nano	Cyclone V SE 5CSEBA6U2317

Table 1. DE-series FPGA device names

Tool NameFormat(s)Run Tool AutomaticallyDesign Entry/S <none><none>Simulation<none><none>Board-LevelTiming<none>Symbol<none>Signal Integrity<none>Boundary Scan<none></none></none></none></none></none></none></none></none>	S <none> <none>     Run this tool automatically to synthesize the current design       <none>     Run gate-level simulation automatically after compilation       Timing     <none>        Symbol     <none>        Signal Integrity     <none></none></none></none></none></none></none>
Simulation <none>       Run gate-level simulation automatically after compilation         Board-Level       Timing       <none>          Symbol       <none>          Signal Integrity       <none></none></none></none></none>	<none> <none>     Run gate-level simulation automatically after compilation       Timing     <none>        Symbol     <none>        Signal Integrity     <none></none></none></none></none></none>
Board-Level     Timing <none>       Symbol     <none>       Signal Integrity     <none></none></none></none>	Timing <none>       Symbol     <none>       Signal Integrity     <none></none></none></none>
Symbol <none>  Signal Integrity <none></none></none>	Symbol <none>       Signal Integrity     <none></none></none>
Signal Integrity <pre></pre> <	Signal Integrity <none></none>
Boundary Scan <none></none>	Boundary Scan <none></none>

Figure 9. Other EDA tools can be specified.

6. 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 Prime messages that refer to third-party tools, which are the tools developed and marketed by companies other than Intel. Since we will rely solely on Quartus Prime tools, we will not choose any other tools. Press Next.

7. A summary of the chosen settings appears in the screen shown in Figure 10. Press Finish, which returns to the main Quartus Prime window, but with *light* specified as the new project, in the title bar, as indicated in Figure 11.

		_
🕥 New Project Wizard	×	
Summary		
When you click Finish, the project will be created with the followi	ng settings:	
Project directory:	C:/Desktop/introtutorial	
Project name:	light	
Top-level design entity:	light	
Number of files added:	0	
Number of user libraries added:	0	
Device assignments:		
Design template:	n/a	
Family name:	Cyclone V (E/GX/GT/SX/SE/ST)	
Device:	5CSEMA5F31C6	
Board:	n/a	
EDA tools:		
Design entry/synthesis:	<none> (<none>)</none></none>	
Simulation:	<none> (<none>)</none></none>	
Timing analysis:	0	
Operating conditions:		
Core voltage:	1.1V	
Junction temperature range:	0-85 °C	
	< Back Next > Finish Cancel Help	1
		١.

Figure 10. Summary of project settings.

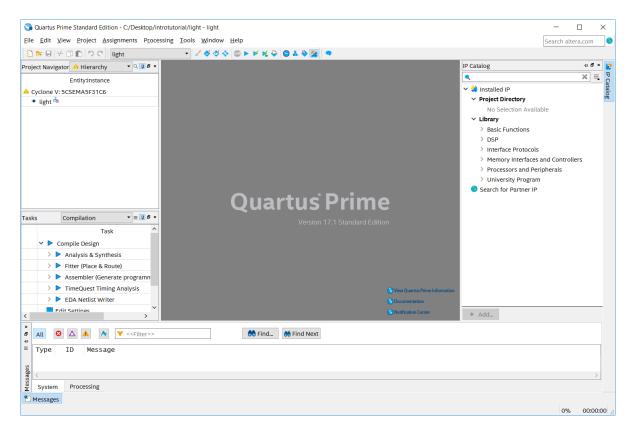


Figure 11. The Quartus Prime window for a created project.

# 5 Design Entry Using VHDL Code

As a design example, we will use the two-way light controller circuit shown in Figure 12. 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.

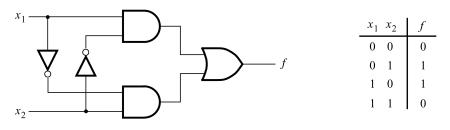


Figure 12. The light controller circuit.

The required circuit is described by the VHDL code in Figure 13. Note that the VHDL entity 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 ASCPrime files, or by using the Quartus Prime 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 VHDL entity. The file name must include the extension *vhd*, which indicates a VHDL file. So, we will use the name *light.vhd*.

```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;
ENTITY light IS
    PORT(x1, x2 : IN STD_LOGIC ;
        f : OUT STD_LOGIC);
END light ;
ARCHITECTURE LogicFunction OF light IS
BEGIN
    f <= (x1 AND NOT x2) OR (NOT x1 AND x2) ;
END LogicFunction ;
```

Figure 13. VHDL code for the circuit in Figure 11.

#### 5.1 Using the Quartus Prime Text Editor

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

Select File > New to get the window in Figure 14, choose VHDL 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 15. In the box labeled Save as type choose VHDL 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 16. Enter the VHDL code in Figure 13 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 VHDL code. First, the editor can display different types of VHDL 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.

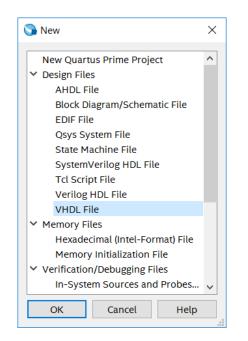


Figure 14. Choose to prepare a VHDL file.

Save As		×
$\leftarrow \rightarrow \checkmark \uparrow$	« Desktop > introtutorial > v 🖑 Se	earch introtutorial
Organize 🔻 New	/ folder	:== ▼ ?
提 Desktop	^ Name	Date modified Type
😹 Documents	db	5/10/2018 11:14 A File folder
駨 Downloads	incremental_db	5/9/2018 9:58 AM File folder
🜗 Music	output_files	5/10/2018 10:57 A File folder
hictures		
🖪 Videos		
📢 OSDisk (C:)	~ <	>
File name:	light	~
Save as type:	VHDL Files (*.vhd *.vhdl)	$\checkmark$
<ul> <li>Hide Folders</li> </ul>	Add file to current project	Save Cancel

Figure 15. Name the file.

S Quartus Prime Standard Edition - C:/Desktop/in	ntrotutorial/light - lig	ght		- 🗆 X
File Edit View Project Assignments Proce	essing Tools Win	ndow Help		Search altera.com
🗋 🗖 🖶 🗲 🗋 💼 🤊 で 🛛 light	- 🏒 🗳	🎸 🔅 💷 🕨 🗶 😓 😓 😫 🔤	•	
Project Navigator 🔺 Hierarchy 🔹 🤉 🖷 🗴		light.vhd	×	<u>*</u>
Entity:Instance	🖼 🛤 🗗 📰 🗄	💷 🖪 🗗 🎦 🖉 👅 💆 🕮 📃		P Catalog
Cyclone V: 5CSEMA5F31C6	1			
📭 light 👛				
< >>				
Tasks Compilation • = 🖪 🗗 ×				
Task ^				
✓ ► Compile Design				
Analysis & Synthesis				
Fitter (Place & Route)				
> Assembler (Generate programm)				
>      TimeQuest Timing Analysis				
EDA Netlist Writer	<			× .
×	<b>`</b>			7
<ul> <li>All</li> <li>▲</li> <li>▲</li> <li>▼ &lt;<filter>&gt;</filter></li> </ul>		💏 Find 👪 Find Next		
Type ID Message				
S .				
System Processing				>
System Processing				
				0% 00:00:00

Figure 16. Text Editor window.

#### 5.1.1 Using VHDL Templates

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

#### 5.2 Adding Design Files to a Project

As we indicated when discussing Figure 7, you can tell Quartus Prime 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 17. 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 Prime 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.vhd* file is already a part of the project and will be listed in the window in

Figure 17. Otherwise, the file must be added to the project. So, if you did not use the Quartus Prime Text Editor, then place a copy of the file *light.vhd*, which you created using some other text editor, into the directory *introtutorial*. To add this file to the project, click on the ... button next to the box labelled File name in Figure 17 to get the pop-up window in Figure 18. Select the *light.vhd* file and click Open. The selected file is now indicated in the File name box in of Figure 17. Click Add then OK to include the *light.vhd* file in the project. We should mention that in many cases the Quartus Prime software is able to automatically find the right files to use for each entity referenced in VHDL 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.

🖌 Settings - light			_	
Category:				Device/Board
General Files	Files			
Libraries V IP Settings	Select the design files project directory to th		project. Click Add All to add all design	files in the
IP Catalog Search Locatior	File name:			Add
Design Templates <ul> <li>Operating Settings and Cond</li> </ul>	•		×	Add All
Voltage Temperature	File Name	Туре	Library Design Entry/Synth	Remove
✓ Compilation Process Settings	light.vhd	VHDL File	<none></none>	Up
Incremental Compilation <ul> <li>EDA Tool Settings</li> </ul>				Down
Design Entry/Synthesis Simulation				Properties
Board-Level				
<ul> <li>Compiler Settings</li> <li>VHDL Input</li> </ul>				
Verilog HDL Input				
Default Parameters TimeQuest Timing Analyzer				
Assembler				
Design Assistant				
Signal Tap Logic Analyzer Logic Analyzer Interface				
Power Analyzer Settings				
SSN Analyzer				
	<		>	
<			OK Cancel Apply	Help
			Cancer Apply	neip

Figure 17. Settings window.

Select File			×
← → × ↑ 🖡 « [	Desktop > introtutorial >	✓ ♥ Search introtutorial	Q
Organize 🔻 New fol	der		2
This PC	Name	Date modified Ty	уре
提 Desktop	📕 db	5/10/2018 11:14 A Fi	ile folder
Documents	incremental_db	5/9/2018 9:58 AM Fi	ile folder
Downloads	output_files	5/10/2018 10:57 A Fi	ile folder
🔥 Music	🥪 light.vhd	5/10/2018 11:16 A H	lard Disk Ir
E Pictures			
Videos			
USDisk (C:)	× <		>
	ame: light.vhd	<ul> <li>✓ Design Files (*.tdf *.vhd *.vh</li> <li><u>O</u>pen</li> <li>Cana</li> </ul>	dl *. ∨

Figure 18. Select the file.

# 6 Compiling the Designed Circuit

The VHDL code in the file *light.vhd* is processed by several Quartus Prime 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 it 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 Prime display. In the message window, at the bottom of the figure, various messages are displayed throughout the compilation process. In case of errors, there will be appropriate messages given.

When the compilation is finished, a compilation report is produced. A tab showing this report is opened automatically, as seen in Figure 21. 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 O. The report includes a number of sections listed on the left side. Figure 21 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.

E       got yeer gotesting Tools Window Help       GetChattera.com         Image: Completion Report-Light       Image: Com	Quartus Prime Standard Edition - C:/Desktop/intro_tutoria	l/light - light			>
Project Navigator Project Navi	le <u>E</u> dit <u>V</u> iew <u>P</u> roject <u>A</u> ssignments P <u>r</u> ocessing <u>T</u> (	ools <u>W</u> indow <u>H</u> elp			Search altera com
Files       Files       Flow Summary         Files       Flow Summary       Flow Summary         Flow Summary       Flow Summary       Flow Summary         Flow Summary       Flow Summary       Revision Name       Light         Flow Summary       Flow Summary       Revision Name       Light         Flow Summary       Flow Summary       Revision Name       Light         Flow Summary       Flow Summary       Completernt       Summary         Flow Summary       Flow Summary       Flow Summary       Revision Name         Flow Summary       Flow Summary       Summary       Summary         Flow Summary       Flow Summary       Summary       Summary	<u>  た </u>	it 🔹 🖌 🍝	🗳 🔇 💷 Þ 🤸 🕯	ƙ � � ♣ <b>� </b> ≇ 🤊	
Image: Interview       Image: Interview <td< td=""><td>oject Navigator 📄 Files 🔹 🔍 📮 🗗 🗙</td><td>light.vhd</td><td>Compilation Re</td><td>port - light 🗵</td><td></td></td<>	oject Navigator 📄 Files 🔹 🔍 📮 🗗 🗙	light.vhd	Compilation Re	port - light 🗵	
Bit registers       File       File </td <td>Files</td> <td>Table of Contents 📃 🗗</td> <td>Flow Summary</td> <td></td> <td></td>	Files	Table of Contents 📃 🗗	Flow Summary		
Filter (Place & Route) Fitter (Place & Route) Fit	BBD light.vhd	E Flow Summary	< <filter>&gt;</filter>		
Flow Elapsed Time Revision Name light   Top-level Entity Name light   Top-level Entity Name light   Task Flow Sog   Flow Messages Compilation   Task Flow Messages   Flow Log Pow Log   Flow Messages Corplevel Entity Name   Flow Messages Corplevel Entity Name   Total registers O   Total registers O   Total block memory bits 0 / 5,662,720 (0 %)   Total block memory bits 0 / 5,662,720 (0 %)   Total HSSI PMARX Deserializers 0 / 112 (0 %)   Total HSSI PMARX Deserializers 0 / 9 (0 %)   Total HSSI PMARX Deserializers 0 / 9 (0 %)   Total HSSI PMARX Deserializers 0 / 9 (0 %)   Total HSSI PMARX Deserializers 0 / 9 (0 %)   Total HSSI PMARX Deserializers 0 / 9 (0 %)   Total HSSI PMARX Deserializers 0 / 9 (0 %)   Total HSSI PMARX Deserializers 0 / 9 (0 %)   Total HSSI PMARX Deserializers 0 / 9 (0 %)   Total HSSI PMARX Deserializers 0 / 9 (0 %)   Total HSSI PMARX Deserializers 0 / 9 (0 %)   Total HSSI PMARX Deserializers 0 / 9 (0 %)   Total HSSI PMARX Deserializers 0 / 9 (0 %)   Total HSSI PMARX Deserializers 0 / 9 (0 %)   Total HSSI PMARX Deserializers 0 / 9 (0 %)   Total HSSI PMARX Deserializers 0 / 9 (0 %)   Total HSSI PMARX Deserializers 0 / 9 (0 %)   Total HSSI PMARX Deserializers 0 / 9 (0 %)		E Flow Settings	Flow Status	Successful - Wed Feb 06 13:59:42 2019	
Top-level Entity Name Upt   Family Cyclone V   Pow Log Device   Star Compilation   Task Fitter   Flow Suppressed Messages   Flo		📰 Flow Non-Default Global Settings	Quartus Prime Version	18.1.0 Build 625 09/12/2018 SJ Standard E	dition
Findly Cyclone V Family Cyclone V Parily Cyclone V Device SCSXFC6D6F31C6 Timing Models Final Compilation Task For Suppressed Messages For Suppression For Sup		Flow Elapsed Time	Revision Name	light	
Analysis & Synthesis Fitter Fitter Flow Messages Flow Messages Flow Suppressed Message Flow Suppressed Messages Flow Suppressed Message Flow Suppressed Messages <td></td> <td></td> <td>Top-level Entity Name</td> <td>light</td> <td></td>			Top-level Entity Name	light	
asks Compilation   asks Compilation   Task   Flow Suppressed Messages   Flow Suppressed Messages   Flow Suppressed Messages   Analysis & Synthesis   Analysis & Synthesis   Fitter (Place & Route)   Fitter (Place & Route) </td <td></td> <td>Flow Log</td> <td>Family</td> <td>Cyclone V</td> <td></td>		Flow Log	Family	Cyclone V	
asks       Compilation       Image: Second S		> Analysis & Synthesis	Device	5CSXFC6D6F31C6	
Task <ul> <li>Flow Suppressed Messages</li> <li>Assembler</li> <li>Timing Analysis &amp; Synthesis</li> <li>Fitter (Place &amp; Route)</li> <li>Fitter (Place &amp; Route)</li> <li>Timing Analysis</li> <li>Total PSI RXPCSs</li> <li>O/(0%)</li> <li>Total HSSI PMA RX Descritalizers</li> <li>O/(0%)</li> <li>Total HSSI PMA RX Descritalizers</li> <li>O/(0%)</li> <li>Total HSSI PMA TX Serializers</li> <li>O/(0%)</li> </ul>		> 📙 Fitter	Timing Models	Final	
Task       Assembler <ul> <li>Comple Design</li> <li>Analysis &amp; Synthesis</li> <li>Fitter (Place &amp; Route)</li> <li>Fitter (Place &amp; Route)</li> <li>Assembler (Generate programming files)</li> <li>Fitter (Place &amp; Route)</li> <li>Timing Analysis</li> <li>Timing Analysis</li> <li>Timing Analysis</li> <li>Total HSSI RX PCSs</li> <li>O/9(0%)</li> <li>Total HSSI PMA RX Descriptions</li> <li>O/9(0%)</li> <li>Total HSSI PMA RX Descriptions</li> <li>Total HSSI PMA RX Serializers</li> <li>O/9(0%)</li> </ul>	sks Compilation 🔻 = 📮 🗗 🗙	Flow Messages	Logic utilization (in ALMs)	1 / 41,910 ( < 1 % )	
Image: Solution of the set of the s	Task	Flow Suppressed Messages	Total registers	0	
>       Analysis & Synthesis       0         >       >       Analysis & Synthesis       0         >       >       Fitter (Place & Route)       0         >       >       Assembler (Generate programming files)       0         >       >       >       Total block memory bits       0 / 5,662,720 (0 %)         Total DSP Blocks       0 / 112 (0 %)       0       0         Total HSSI RX PCSs       0 / 9 (0 %)       0       0         Total HSSI TX PCSs       0 / 9 (0 %)       0       0         Total HSSI TX PCSs       0 / 9 (0 %)       0       0         Total HSSI TX PCSs       0 / 9 (0 %)       0       0         Total HSSI TX PCSs       0 / 9 (0 %)       0       0         Total HSSI TX PCSs       0 / 9 (0 %)       0       0         Total HSSI TX PCSs       0 / 9 (0 %)       0       0         Total HSSI TX PCSs       0 / 9 (0 %)       0       0         Total HSSI TX PCSs       0 / 9 (0 %)       0       0         Total HSSI TX PCSs       0 / 9 (0 %)       0       0         Total HSSI TX PCSs       0 / 9 (0 %)       0       0         Total HSSI TX PCSs       0 / 9 (0 %)       0 <t< td=""><td></td><td>&gt; 📥 Assembler</td><td>Total pins</td><td>3 / 499 ( &lt; 1 % )</td><td></td></t<>		> 📥 Assembler	Total pins	3 / 499 ( < 1 % )	
<pre>&gt;&gt; Fitter (Place &amp; Route) &gt;&gt; Fitter (Place &amp; Route) &gt;&gt; Assembler (Generate programming files) &gt;&gt; Total HSSI PMA RX Deserializers 0/9(0%) Total HSSI RX PCSs 0/9(0%) Total HSSI TX PCSs 0/9(0%) Total HSSI TX PCSs 0/9(0%) Total HSSI TX PCSs 0/9(0%) Total HSSI PMA TX Serializers 0/9(0%) Total HSSI</pre>		🔉 📕 Timing Analyzer	Total virtual pins	0	
All C C Sign of the Constrained for setup requirements     332102 Design is not fully constrained for setup requirements     Quartus Prime Timing Analyzer was successful. 0 errors, 13 warnings     293000 Quartus Prime Full Compilation was successful. 0 errors, 13 warnings     201     10			Total block memory bits	0 / 5,662,720 ( 0 % )	
Timing Analysis EDA Netilist Writer EDA Netilist Writer Total HSSI PMA RX Deserializers 0/9(0%) Total HSSI TX PCSs 0/9(0%) Total HSSI TX Serializers 0/9(0%) Total HSSI PMA TX Serializers 0/9(0%) <p< td=""><td></td><td></td><td>Total DSP Blocks</td><td>0 / 112 ( 0 % )</td><td></td></p<>			Total DSP Blocks	0 / 112 ( 0 % )	
Image: Norther State         EDA Netlist Writer         Image: State	<ul> <li>Assembler (Generate programming files)</li> </ul>		Total HSSI RX PCSs	0/9(0%)	
All     Image: Second sec	> Timing Analysis		Total HSSI PMA RX Deserializers	0/9(0%)	
All       Image: Second s	> 🕨 EDA Netlist Writer		Total HSSI TX PCSs	0/9(0%)	
All       Image: Constraint of the setup requirements         Type       ID       Message         0       332102 Design is not fully constrained for setup requirements         0       332102 Design is not fully constrained for hold requirements         0       Quartus Prime Timing Analyzer was successful.         0       293000 Quartus Prime Full Compilation was successful.         0       293000 Quartus Prime Full Compilation was successful.         0       293000 Quartus Prime Full Compilation was successful.	>	< >	Total HSSI PMA TX Serializers	0/9(0%)	
<ul> <li>332102 Design is not fully constrained for setup requirements</li> <li>332102 Design is not fully constrained for hold requirements</li> <li>Quartus Prime Timing Analyzer was successful. 0 errors, 6 warnings</li> <li>293000 Quartus Prime Full Compilation was successful. 0 errors, 13 warnings</li> </ul>	All 🕄 🟡 🔺 🗡 🔻 < <filter>&gt;</filter>	66 Find	ind Ne <u>x</u> t		
System Processing (124)	Type ID Message 332102 Design is not fully const 332102 Design is not fully const Quartus Prime Timing Anal 293000 Quartus Prime Full Compil	rained for hold requirements yzer was successful. O errors			
System Processing (124)	· · · · · · · · · · · · · · · · · · ·				>
100% 00:0	System Processing (124)				100% 00:01:0

Figure 19. Display after a successful compilation.

#### 6.1 Errors

Quartus Prime software displays messages produced during compilation in the Messages window. If the VHDL 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 VHDL 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 VHDL 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.vhd. Remove the semicolon in the statement that defines the function

*f*, 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.vhd* file should be saved; click Yes. After trying to compile the circuit, Quartus Prime software will display error messages in the Messages window, and show that the compilation failed in the Analysis & Synthesis stage of the compilation process. The compilation report summary, given in Figure 20, 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 21. The Compilation Report can be dispayed as a separate window as in Figure 21 by right-clicking its tab and selecting Detach Window, and can be reattached by clicking Window > Attach Window. Double-click on the first error message. Quartus Prime software responds by opening the *light.vhd* file and highlighting the statement which is affected by the error, as shown in Figure 22. Correct the error and recompile the design.

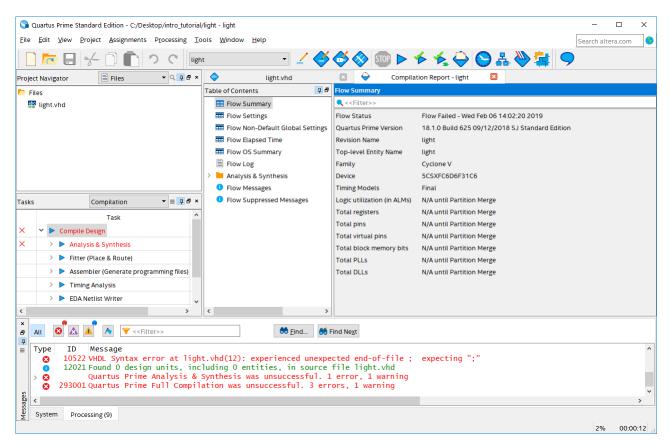


Figure 20. Compilation report for the failed design.

Compilation Report - C:/Desktop/introtutorial/light - light	-		×
<u>Eile E</u> dit <u>I</u> ools <u>W</u> indow <u>H</u> elp	Search alter	ra.com	9
Table of Contents 🛛 🖗 Analysis & Synthesis Messages			
Image: Flow Summary       Image: Flow Settings         Image: Flow Settings       All         Image: Flow Settings       Image: Flow Settings         Image: Flow Settings       Image: Flow Settings			
<ul> <li></li></ul>	ing on si letected le ; exp	hared	mach
• Messages			>
Processing (8)	2%	00.00	).30

Figure 21. Error messages.

							_	~
	Standard Edition - C:/Desktop/int						_	×
	Project Assignments Proces					Search alter	a.com	9
🗋 🗖 🗖 🚽 🖆	ا 🖍 🔿 🖓 الوالي 🖍	- 🖌 🎸 🗇 🔊 🕨	· 🗙 🤻 🔶 🚫 🔻 🔌 🐉 💊					
Project Navigator	A Hierarchy 🔹 🤉 🖛 🛪		light.vhd	×				_ 😫
À Cyclone V: 5CS	Entity:Instance	6 f : OUT ST 7 END light ; 8 ⊟ARCHITECTURE Log 10 ⊟BEGIN	<pre>jic_1164.all ; : IN STD_LOGIC ; TD_LOGIC); gicFunction OF light NOT x2) OR (NOT x1 A</pre>				^	F Catalog
×      ×      ► Comm     ×      ×      ► Comm     ×      ► Aa     →      ► Fi     →      ► Aa     →      ► Ti     →      ► EI	mpilation							
<	>	<					>	
<b>₽</b>	▲ ▲ ▼ < <filter>&gt;</filter>		💏 Find Ne <u>x</u> t					
× 10 0 12 > ⊗	030 Parallel compilation 522 VHDL Syntax error a 021 Found 0 design units	at light.vhd(12): exper ts, including 0 entitie ysis & Synthesis was un	rienced unexpected en es, in source file li nsuccessful. 1 error,	d-of-file ; expecting ";" ght.vhd 1 warning	VHDL File	2%	>	

Figure 22. Identifying the location of the error.

### 7 Pin Assignment

During the compilation above, the Quartus Prime 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 a light-emitting diode on your DE-series board. For the DE2-115 we will use a green LED:  $LEDG_0$ . On the DE0-CV, DE1-SoC, DE-10 Lite and DE10-Standard we will use  $LEDR_0$ . On the DE0-Nano and DE0-Nano-SoC, we will use  $LED_0$  The FPGA pin assignment for the LEDs can also be found in Table 2.

Component	SW <sub>0</sub>	$SW_1$	$LEDG_0, LED_0, \text{ or } LEDR_0$
DE0-CV	PIN_U13	PIN_V13	PIN_AA2
DE0-Nano	PIN_M1	PIN_T8	PIN_A1
DE0-Nano-SoC	PIN_L10	PIN_L9	PIN_W15
DE2-115	PIN_AB28	PIN_AC28	PIN_E21
DE1-SoC	PIN_AB12	PIN_AC12	PIN_V16
DE10-Lite	PIN_C10	PIN_C11	PIN_A8
DE10-Standard	PIN_AB30	PIN_AB28	PIN_AA24
DE10-Nano	PIN_Y24	PIN_W24	PIN_W15

Table 2. DE-Series Pin Assignments

<	🍑 As	signment Edito	or - C:/Desktop/	introtutorial/light - light				— C	x c
F	ile	Edit View T	ools Window	/ Help				Search altera	.com 🜖
<	<nev< td=""><td>/&gt;&gt; ▼ 🗹 Filte</td><td>er on node nar</td><td>nes: *</td><td></td><td>~</td><td>Category: All</td><td>l</td><td>•</td></nev<>	/>> ▼ 🗹 Filte	er on node nar	nes: *		~	Category: All	l	•
	tatu	From	То	Assignment Name	Value	Enabled	Entity	Comment	Tag
1		< <new>&gt;</new>	< <new>&gt;</new>	< <new>&gt;</new>					
× 5									
4	ı 🗌								
								0%	00:00:00

Figure 23. The Assignment Editor window.

Pin assignments are made by using the Assignment Editor. Select Assignments > Assignment Editor to reach the window in Figure 23 (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  $\widehat{\mathbb{M}}$  appears. Click on the button (not the drop down arrow) to reach the window in Figure 24. Click on  $\bigotimes$  and  $\bigotimes$  to show or hide more search options. 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 25 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, doubleclick 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  $x^2$  and output f to the appropriate pins listed in Table 2. An example using a DE1-SoC board is shown in Figure 26. 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.

Named: *							
Options Filter: Pins: all Customize Look in: [light] Include subentities Hierarchy view Matching Nodes: Name Assignments Iight Iight Iight Assigned Assigned X1 Unassigned X2 Unassigned Customize Nodes Found: Name Assignments Name Assignments Name Customize Nodes Found: Name Assignments Customize Nodes Found: Name Assignments Customize Nodes Found: Name Assignments Customize Nodes Found: Name Assignments Customize Name Assignments Customize Nodes Found: Name Assignments Name Assignments Customize Name Assignments Customize Name Assignments Name Assignments Customize Name Assignments Customize Name Assignments Customize	🗳 Node Fii	nder					×
Name     Assignments       light     >       Ight     >       Image: Assignments     >       Image: Assignm	Options Filter:	Pins: all		~ [	🗹 Include sube		Customize
OK Cancel	light ≌f i⊫ x1	~	Assignments Unassigned Unassigned	<ul> <li></li> <li></li></ul>	Name		Assignments
	<		>	<b>[</b> 4	<	OK	

Figure 24. The Node Finder displays the input and output names.

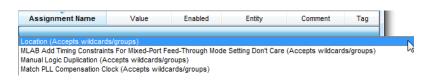


Figure 25. The available assignment names for a DE-series board.

🗳 Assignment Editor - C:/Desktop/introtutorial/light - light File Edit View Tools Window Help									
-	_		· _					Search alter	a.com
<	<new< td=""><td>/&gt;&gt; ▼ 🗹 Filte</td><td>er on node nar</td><td>nes: *</td><td></td><td>~</td><td>Category: All</td><td></td><td>•</td></new<>	/>> ▼ 🗹 Filte	er on node nar	nes: *		~	Category: All		•
	tatu	From	То	Assignment Name	Value	Enabled	Entity	Comment	Tag
1	<ul> <li>Image: A second s</li></ul>		<mark>in</mark> – x1	Location	PIN_AB12	Yes			
2	<b>~</b>		<mark>in</mark> _ x2	Location	PIN_AC12	Yes			
3	<		🌥 f	Location	PIN_V16	Yes			
4		< <new>&gt;</new>	< <new>&gt;</new>	< <new>&gt;</new>					
× P P									
								0%	00:00:00

Figure 26. The complete assignment.

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 Prime 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 Settings File (QSF)* format. The format for the file for our simple project (on a DE1-SoC board) is

```
set_location_assignment PIN_AB12 -to x1
set_location_assignment PIN_AC12 -to x2
set_location_assignment PIN_V16 -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 27. 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.

🔒 Export Assignme	nts	×
Assignments to e	(port	
File name:		
C:/Desktop/intro	tutorial/atom_netlists/light.qsf	
Export assignmer	nts hierarchy path:	
light		
Save intermediate	e synthesis results	
Save a node-l	evel netlist of the entire design into a persisten	t source file
File name:		
	OK Cancel	Help

Figure 27. Exporting the pin assignment.

You can import a pin assignment by choosing Assignments > Import Assignments. This opens the dialogue in Figure 28 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.

S Import Assignments		×					
Specify the source and categories of ass	signments to in	nport.					
File name: Categories							
Copy existing assignments into light.	qsf.bak before	Advanced					
OK	Cancel	Help					

Figure 28. 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 DE1-SoC pin assignments can be found in the  $DE1\_SoC.qsf$  file, which is available from Intel's FPGA University Program website. This file uses the names found in the  $DE1\_SoC$  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 Block Diagram/Schematic design file; namely, SW[0], SW[1] and LEDG[0] for x1, x2 and f, respectively. Since these signals are specified in the  $DE1\_SoC.qsf$  file as elements of vectors SW and LEDG, we must refer to them in the same way in our design file. For example, in the  $DE1\_SoC.qsf$  file the 10 toggle switches are called SW[9] to SW[0]. In a design file they can also be referred to as a vector SW[9..0].

# 8 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 Prime Compiler's Assembler module. Intel'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 Prime software) to the board by means of a cable that connects a USB port on the host computer to the USB-Blaster 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 Intel'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 Prime 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 Prime software if the power is turned off and on. The choice between the two modes is made by switches on the DE-series board. Consult your manual for the location of this switch on your DE-series board. The boards should be set to JTAG mode by default. This tutorial discusses only the JTAG programming mode.

#### 8.1 JTAG\* Programming for the DE0-CV, DE0-Nano, DE10-Lite, and DE2-115 Boards

For the DE0-CV, DE0-Nano, DE10-Lite, and DE2-115 Boards, the programming and configuration task is performed as follows. If using the DE1-SoC board, then the instructions in the following section should be followed. To program the FPGA chip, the RUN/PROG switch on the board must be in the RUN position. Select Tools > **Programmer** to reach the window in Figure 29. 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 30.

Nrogrammer	- C:/Desktop/introtuto	orial/light - light -	[light.cdf]			_		×
File Edit View	Processing Tools	Window Hel	р			Search	n altera.c	om 🌖
🚖 Hardware S					•	Pro	ogress:	
Enable real-	time ISP to allow back	ground program	ming when a	available				
▶ <sup>™</sup> Start	File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank- Check	Examine
🕮 Stop	output_files/light	EP4CE115F29	005629F2	005629F2	$\checkmark$			
🕦 Auto Detec								
× Delete								
붬 Add File	<							>
<sup>6</sup> Change File								
🕒 Save File		•••						
Add Device	TDI							
1 <sup>ኬ</sup> Up								
	EP4CE115	F29						
I™ Down	<b>↓</b>	]						

Figure 29. The Programmer window.

Observe that the configuration file *light.sof* is listed in the window in Figure 29. 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. Ensure the Program/Configure check box is ticked, as shown in Figure 29.

Hardware Setup					
Hardware Settings	JTAG Se	ettings			
Select a programming hardware setup appli				 devices. Th	is programming
Currently selected ha	rdware:	No Hard	ware		Ŧ
Available hardware	items				
Hardware		Server	Port		Add Hardware
USB-Blaster		Local	USB-1		Remove Hardware
					Close
					Close

Figure 30. The Hardware Setup window.

Now, press Start in the window in Figure 29. An LED on the board will light up corresponding to the programming operation. If you see an error reported by Quartus Prime software indicating that programming failed, then check to ensure that the board is properly powered on.

#### 8.2 JTAG\* Programming for the DE0-Nano-SoC, DE1-SoC Board, DE10-Nano, and DE10-Standard

For the DE0-Nano-SoC, DE1-SoC Board, DE10-Nano, and DE10-Standard boards, the following steps should be used for programming. Select Tools > Programmer to reach the window in Figure 31 (if the SOCVHPS device is missing, it can be added through the Add Device menu under the *Soc Series V* family). 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 *DE-SoC* is not chosen by default as the programming hardware, then press the Hardware Setup... button and select the *DE-SoC* in the window that pops up, as shown in Figure 32.

Nogrammer -	- C:/Desktop/introtuto	rial/light - light -	[liaht.cdf]*			_		×
	Processing Tools					Search	n altera.c	om
			r			Jocarci	T atter a.e	.0111
📥 Hardware Se	etup DE-SoC [USB	-1] Mod	de: JTAG		•	Pro	ogress:	
Enable real-t	ime ISP to allow back	ground program	ming when	available				
▶ <sup>™</sup> Start	File	Device	Checksum	Usercode			Blank-	Examine
					Configure		Check	
📲 Stop	output_files/light	5CSEMA5F31	00AF4DD1	00AF4DD1	$\checkmark$			
🟓 Auto Detec	<none></none>	SOCVHPS	00000000	<none></none>				
× Delete								
📩 Add File	<							>
Change File								
Save File								
	TDI	(intel						
Add Device		<b>→</b>						
1 <sup>1</sup> Up		.⊧ ∖⁼						
<sup>1™</sup> Down	TDO 5CSEMA5	F31 SOCVH	IPS					
	•							

Figure 31. The Programmer window.

Observe that the configuration file *light.sof* in directory *output\_files* is listed in the window in Figure 31. 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. Ensure the Program/Configure box is checked. This setting is used to select the FPGA in the Cyclone V SoC chip for programming. If the *SOCVHPS* device is not shown as in Figure 31, click Add Device > SoC Series V > SOCVHPS then click OK. Ensure that your device order is consistent with Figure 31 by clicking on a device and then clicking  $\boxed{\uparrow^{u_D}}$  or  $\boxed{\downarrow^{u_D}}$  or

Hardware Setup			
Hardware Settings JTA	G Settings		
Select a programming hard nardware setup applies only			devices. This programming
Currently selected hardware	E: DE-SoC [	[USB-1]	
Available hardware items			
Hardware	Server	Port	Add Hardware
DE-SoC	Local	USB-1	Remove Hardware
			Close
			close

Figure 32. The Hardware Setup window.

Now, press Start in the Programmer. An LED on the board will light up while the FPGA device is being programmed. If you see an error reported by Quartus Prime software indicating that programming failed, then check to ensure that the board is properly powered on.

### 9 Testing the Designed Circuit

Before implementing the designed circuit in the FPGA chip on the DE-series board, it is prudent to simulate it to ascertain its correctness. While not covered in this tutorial, users may use software such as Modelsim or other simulation environments to test the circuit in simulation. Simulation of a circuit often provides a comprehensive view of the circuit's functionality, and can help users easily find bugs within the circuit's logic without having to touch hardware.

Having downloaded the configuration data into the FPGA device, you can now test the implemented circuit. 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 12.

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

Copyright © Intel Corporation. All rights reserved. Intel, the Intel logo, Altera, Arria, Avalon, Cyclone, Enpirion, MAX, Nios, Quartus and Stratix words and logos are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries. Intel warrants performance of its FPGA and semiconductor products to current specifications in accordance with Intel's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Intel assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Intel. Intel customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

\*Other names and brands may be claimed as the property of others.