DE1-Soc

MY FIRST FPGA



G



1111

Copyright © 2003-2014 Terasic Technologies Inc. All Rights Reserved.



Chapter 1	Introduction	1
	1.1 Design Flow	1
	1.2 Before You Begin	2
	1.3 What You Will Learn	6
Chapter 2	Assign The Device	7
	2.1 Assign The Device	7
Chapter 3	Design Entry	11
	3.1 Add a PLL Megafunction	11
	3.2 Add a Multiplexer	
	3.3 Assign the Pins	
	3.4 Create a Default TimeQuest SDC File	
Chapter 4	Compile and Verify Your Design	39
	4.1 Compile Your Design	
	4.2 Program the FPGA Device	41
	4.3 Verify The Hardware	44
Chapter 5	Appendix	47
	5.1 Headquarter & Branches	47

Chapter 1

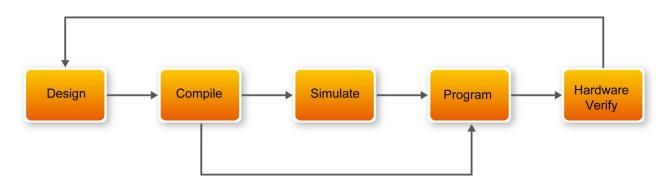


This tutorial provides comprehensive information that will help you understand how to create a FPGA design and run it on your DE1-SoC development board. The following sections provide a quick overview of the design flow, explain what you need to get started, and describe what you will learn.

1.1 Design Flow

Figure 1-1shows the FPGA design flow block diagram.

The standard FPGA design flow starts with design entry using schematics or a hardware description language (HDL), such as Verilog HDL or VHDL. In this step, you can create a digital circuit that is implemented inside the FPGA. The flow then proceeds through compilation, simulation, programming, and verification in the FPGA hardware.





This tutorial guides you through all of the steps except for simulation. Although it is not covered in this document, simulation is very important to learn, and there are entire applications devoted to simulating hardware designs. There are two types of simulation, Functional and Timing Functional simulation allows you to verify that your code is manipulating the inputs and outputs appropriately. Timing (or post place-and-route) simulation verifies that the design meets timing and functions appropriately in the device.



1.2 Before You Begin

This tutorial assumes the following prerequisites

■ You generally know what a FPGA is. This tutorial does not explain the basic concepts of programmable logic.

■ You are somewhat familiar with digital circuit design and electronic design automation (EDA) tools.

■ You have installed the Altera Quartus II 13.1 software on your computer. If you do not have the Quartus II software, you can download it from the Altera web site at www.altera.com/download.

■ You have a DE1-SoC Development Board on which you will test your project. Using a development board helps you to verify whether your design is really working.

■ You have gone through the quick start guide and/or the getting started user guide for your development kit. These documents ensure that you have:

- Installed the required software.
- Determined that the development board functions properly and is connected to your computer.

Next step you should installed the USB-BlasterII driver, Plug in the 12-volt adapter to provide power to the board. Use the USB cable to connect the leftmost USB connector on the DE1-SoC board to a USB port on a computer that runs the Quartus II software. Turn on the power switch on the DE1-SoC board.

The computer will recognize the new hardware connected to its USB port and Power on the board as shown in **Figure 1-2**, but it will be unable to proceed if it does not have the required driver already installed. The DE1-SoC board is programmed by using Altera USB-Blaster II mechanism. If the USB-BlasterII driver is not already installed, the Driver Software Installation in **Figure 1-3** will appear.Click close.





Figure 1-2 Connection Setup

U Driver Software Installation					
Device driver software wa	s not successfully installed				
Please consult with your device manufacturer for assistance getting this device installed.					
Unidentified Device X No driver found					
What can I do if my device did not	t install properly?				
	Close				

Figure 1-3 Driver Software Installation

Since the desired driver is not available on the Windows Update Web site, open the Computer Management and select the Device Manager. This leads to the window in Figure 1-4.



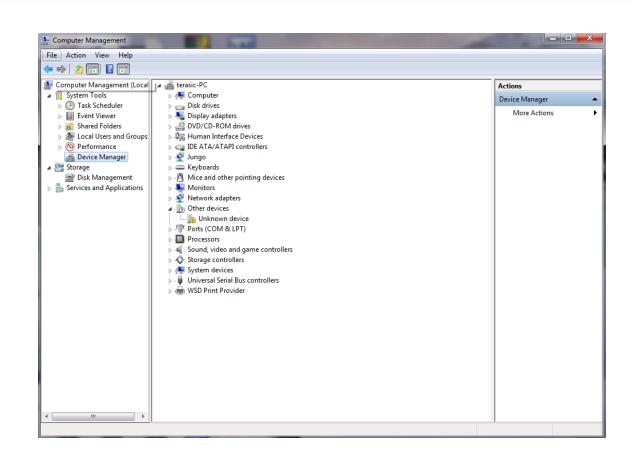
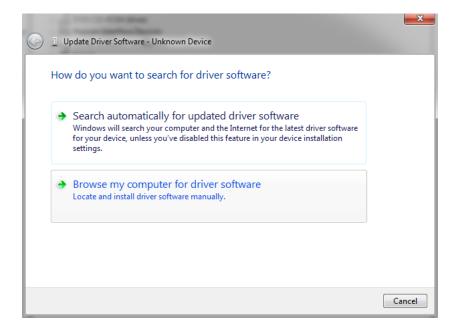


Figure 1-4 Device Manager

Right click Other devices>Unknown device and select Update Driver Software...This leads to the window in **Figure 1-5.**





4



ter aslC



The driver is available within the Quartus II software. Hence, click Browse my computer for device software to

Browse for driver software on your computer
Search for driver software in this location:
C:\altera\13.1\quartus\drivers\usb-blaster-ii
Let me pick from a list of device drivers on my computer This list will show installed driver software compatible with the device, and all driver
software in the same category as the device.

Figure 1-6.

	vse for driver software on your computer
Searc	h for driver software in this location:
C:\al	ltera\13.1\quartus\drivers\usb-blaster-ii

Figure 1-6 Specify the location of the driver

Now, click Browse to get to the pop-up box in **Figure 1-7** Find the desired driver, which is at location C:\altera\13.1\quartus\drivers\usb-blaster-ii. Click OK and then upon returning to Figure 1-6.click Next.





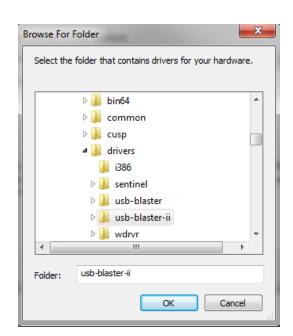
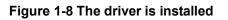


Figure 1-7 Browse to find the location

The driver will now be installed as indicated in **Figure 1-8** Click close and you can start using the DE1-SoC board.

	La reconstruction de la construction de la construc	×
\bigcirc	Update Driver Software - Altera USB-Blaster II (Unconfigured)	
	Windows has successfully updated your driver software	
	Windows has finished installing the driver software for this device:	
	Altera USB-Blaster II (Unconfigured)	
		Close



1.3 What You Will Learn

In this tutorial you will perform the following tasks:



Create a design that causes LEDs on the development board to blink at a speed that is controlled by an input key—This design is easy to create and gives you visual feedback that the design works. Of course, you can use your DE1-SoC board to run other designs as well. For the LED design, you will write Verilog HDL code for a simple 32-bit counter, add a phase-locked loop (PLL) megafunction as the clock source, and add a 2-input multiplexer megafunction. When the design is running on the board, you can press an input switch to multiplex the counter bits that drive the output LEDs.

Becoming familiar with Quartus II design tools—This tutorial will not make you an expert, but at the end, you will understand basic concepts about Quartus II projects, such as entering a design using a schematic editor and HDL, compiling your design, and downloading it into the FPGA on your DE1-SoC development board.

Develop a foundation to learn more about FPGAs—For example, you can create and download digital signal processing (DSP) functions onto a single chip, or build a multi-processor system, or create anything else you can imagine all on the same chip. You don't have to scour data books to find the perfect logic device or create your own ASIC. All you need is your computer, your imagination, and an Altera DE1-SoC FPGA development board

Chapter 2

Assign The Device

You begin this tutorial by creating a new Quartus II project. A project is a set of files that maintain information about your FPGA design. The Quartus II Settings File (.qsf) and Quartus II Project File (.qpf) files are the primary files in a Quartus II project. To compile a design or make pin assignments, you must first create a project.

2.1 Assign The Device

 In the Quartus II software, select File > New Project Wizard. The Introduction page opens. See Figure 2-1.



🗼 New Pri	oject Wizard			
Intro	duction			
The New	Project Wizard helps you create a new project and prel	iminary project settings, including the follov	ving:	
•	Project name and directory Name of the top-level design entity Project files and libraries Target device family and device			
EDA tool settings You can change the settings for an existing project and specify additional project-wide settings with the Settings command (Assignments menu). You can use the various pages of the Settings dialog box to add functionality to the project.				
📃 Don't	t show me this introduction again			
		< Back Next >	Finish Cancel Help	

Figure 2-1 New Project Wizard introduction

2. Click Next.

eras

3. Enter the following information about your project:

a. What is the working directory for this project? Enter a directory in which you will store your Quartus II project files for this design.

- b. For example, C:\My_design\my_first_fpga.
- c. File names, project names, and directories in the Quartus II software cannot contain spaces.
- d. What is the name of this project? Type my_first_fpga.

e. What is the name of the top-level design entity for this project? Type my_first_fpga. See **Figure 2-2**.

💱 New Project Wizard	×
Directory, Name, Top-Level Entity [page 1 of 5]	
What is the working directory for this project?	
C:/My_design/my_first_fpga	
What is the name of this project?	_
What is the name of the top-level design entity for this project? This name is case sensitive and must exactly match the entity name in the design file.	_
my_first_fpga	
Use Existing Project Settings	
< Back Next > Finish Cancel Help	

Figure 2-2 Project information

f. Click Next.

g. You will assign a specific FPGA device to the design and make pin assignments. See Figure 2-3.



ou can install additional device s	support with the Install I	Devices comm	and on the Tools					
Device family					able devices' list			
Family: Cyclone V (E/GX/GT/SX/SE/ST)				Package: Any				•
Devices: Cyclone V SE Mainstream				Pin count: Any			-	
			Spee	Speed grade: Any				-
Target device				-	Ally			<u> </u>
Auto device selected by th	e Fitter		Nam	e filter:				
Other: n/a								
Name	Core Voltage	ALMs	User I/Os		B Channel PMA	GXB Channel PCS	PCIe (I	P -
Name CSEMA4U23A7 (Advanced)	1.1V	15880	314	0	B Channel PMA	0	0	p -
Name CSEMA4U23A7 (Advanced) CSEMA4U23C6 (Advanced)	1.1V 1.1V	15880 15880	314 314	0	B Channel PMA	0	0	P -
Name CSEMA4U23A7 (Advanced) CSEMA4U23C6 (Advanced) CSEMA4U23C7 (Advanced)	1.1V 1.1V 1.1V	15880 15880 15880	314 314 314 314	0 0 0	B Channel PMA	0 0 0	0 0 0	-
Name CSEMA4U23A7 (Advanced) CSEMA4U23C6 (Advanced) CSEMA4U23C7 (Advanced) CSEMA4U23C8 (Advanced)	1. 1V 1. 1V 1. 1V 1. 1V 1. 1V	15880 15880 15880 15880	314 314 314 314 314	0 0 0 0 0 0	B Channel PMA	0 0 0 0	0 0 0 0 0	-
Name CSEMA4U23A7 (Advanced) CSEMA4U23C6 (Advanced) CSEMA4U23C7 (Advanced) CSEMA4U23C8 (Advanced) CSEMA4U23I7 (Advanced)	1.1V 1.1V 1.1V 1.1V 1.1V 1.1V	15880 15880 15880 15880 15880	314 314 314 314 314 314	0 0 0 0 0	B Channel PMA	0 0 0 0 0	0 0 0 0 0	-
Name CSEMA4U23A7 (Advanced) CSEMA4U23C6 (Advanced) CSEMA4U23C7 (Advanced) CSEMA4U23C8 (Advanced) CSEMA4U23I7 (Advanced) CSEMA5F31A7	1.1V 1.1V 1.1V 1.1V 1.1V 1.1V 1.1V	15880 15880 15880 15880 15880 32070	314 314 314 314 314 314 457	0 0 0 0 0 0 0	B Channel PMA	0 0 0 0 0 0 0	0 0 0 0 0 0	-
Name CSEMA4U23A7 (Advanced) CSEMA4U23C6 (Advanced) CSEMA4U23C7 (Advanced) CSEMA4U23C8 (Advanced) CSEMA4U23I7 (Advanced) CSEMA5F31A7 CSEMA5F31C6	1.1V 1.1V 1.1V 1.1V 1.1V 1.1V 1.1V 1.1V	15880 15880 15880 15880 15880 15880 32070 32070	314 314 314 314 314 314 457 457	0 0 0 0 0 0 0 0	B Channel PMA	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0	-
Name CSEMA4U23A7 (Advanced) CSEMA4U23C6 (Advanced) CSEMA4U23C7 (Advanced) CSEMA4U23C8 (Advanced) CSEMA4U23I7 (Advanced) CSEMA5F31A7 CSEMA5F31C6 CSEMA5F31C7	1.1V 1.1V 1.1V 1.1V 1.1V 1.1V 1.1V	15880 15880 15880 15880 15880 32070	314 314 314 314 314 314 457	0 0 0 0 0 0 0 0 0	B Channel PMA	0 0 0 0 0 0 0 0 0	0 0 0 0 0 0	-
	1.1V 1.1V 1.1V 1.1V 1.1V 1.1V 1.1V 1.1V 1.1V 1.1V	15880 15880 15880 15880 15880 32070 32070 32070 32070	314 314 314 314 314 314 457 457 457	0 0 0 0 0 0 0 0	B Channel PMA	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0	

Figure 2-3 Specify the Device Example

h. Click Finish.

teraslC

4. When prompted, choose Yes to create the my_first_fpga project directory. You just created your first Quartus II FPGA project. See Figure 2-4.





🚱 Quartus II 64-Bit - C/My_design/my_first_fpga/my_first_fpga - my_first_fpga	
File Edit View Project Assignments Processing Tools Window Help 🖓	Search altera.com
🗋 🖆 🖬 🦉 🕺 🖄 🖄 🥙 🔍 [my_first_fpox 🔹 🔹 💓 😰 🖌 🦃 🤯 🤣 🔘 🕨 🦻 🖄 🔞 🕫	
Project Navigator 48 x	
▲ Herarchy Image: Files J ² Design Units Y IP Comp (↓)	
A Herardhy ■ Files J ^a Design Units © IP Comp () Tasks ♀ Ø × Row Completion ▼ Oxtropice	
Task 0 ^	
V A Comple Design	
✓ ▷ ▶ Analysis & Synthesis ■ ✓ ▷ ▶ Fitter (Place & Route) ■	
✓ ▷ ▶ Fitter (Place & Route) ✓ ▷ ▶ Assembler (Generate programming files)	S View New Quartus II Information
V D TimeQuest Timing Analysis	Documentation
✓ ▷ ► EDA Netist Writer ✓ III ►	Notification Center
× A A ⊗ △ A A 	
system / Processing /	•
	0% 00:00:00

Figure 2-4 my_first_fpga project



Chapter 3 Design Entry

3.1 Add a PLL Megafunction

This section describes How to Add a PLL Megafunction

In the design entry step you create a schematic or Block Design File (.bdf) that is the top-level design. You will add library of parameterized modules (LPM) functions and use Verilog HDL code to add a logic block. When creating your own designs, you can choose any of these methods or a combination of them.

1. Choose File > New > Block Diagram/Schematic File (see **Figure 3-1** to create a new file, Block1.bdf, which you will save as the top-level design.



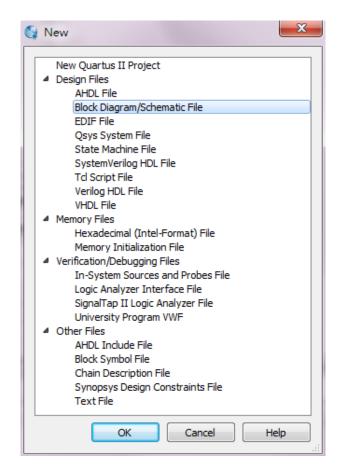


Figure 3-1 New BDF

- 2. Click OK.
- 3. Choose File > Save As and enter the following information.
 - File name: my_first_fpga
 - Save as type: Block Diagram/Schematic File (*.bdf)
- 4. Click Save. The new design file appears in the Block Editor (see Figure 3-2).





	Window Help 🗟	Search altera.com
🚰 🛃 🐊 👗 🗈 🛍 🌱 (억 🦷 my_first_fpga		
ct Navigator 🛛 🖓 🗗 🖇		
-		
Cyclone V: 5CSEMA5F31C6		
Ŗ my_first_fpga 🚈	1	
	-	
	<u> </u>	
Hierarchy 📄 Files 🧬 Design Units 📉 IP Compt ୶	»	
្រុ <u>ក</u> ្រុំ		
	▲	
Compilation Customize		
	=	
Task 🕚	â	
4 🕨 Compile Design	E	
Analysis & Synthesis		
Fitter (Place & Route)		
Assembler (Generate programming files)		
TimeOuest Timing Analysis	· ·	
4 III	() · · · · · · · · · · · · · · · · · ·	4
All 🙆 🛕 👩 💎 < <search>></search>	~	
Type ID Message		

Figure 3-2 Bank BDF

- 5. Add HDL code to the blank block diagram by choosing File > New > Verilog HDL File.
- 6. Click OK to create a new file Verilog1.v, which you will save as simple_counter.v.
- 7. Select File > Save As and enter the following information (see Figure 3-3).
 - File name: simple_counter.v
 - Save as type: Verilog HDL File (*.v, *.vlg, *.verilog)



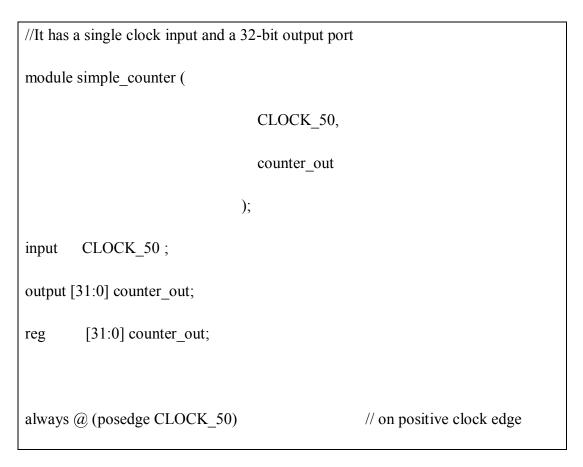


Save As						×
Save in:	\mu my_first_fpga	•	+ 🗈 💣 📰 -			
C.	Name	•	Date modified	Туре	Size	
Recent Places	퉲 db		5/7/2013 10:30 AM	File folder		
Desktop						
Libraries						
Computer						
Network						
	File name:	simple_counter.v			•	Save
	Save as type:	Verilog HDL Files (*.v *.vlg *.verilog)			•	Cancel
		Add file to current project				1.

Figure 3-3 Saving the Verilog HDL file

The resulting empty file is ready for you to enter the Verilog HDL code.

8. Type the following Verilog HDL code into the blank simple_counter.v file (see **Figure 3-4** The Verilog File of simple_counter.v).



begin	
counter_out <= #1 counter_out + 1;// increment	counter
end	
endmodule	// end of module counter

💱 Quartus II 64-Bit - C:/My_design/my_first_fpga/my_first_fpga -	my_first_fpga		X
File Edit View Project Assignments Processing Tools Windo	w Help 🗐	Search altera.com	٠
🗋 💕 🗐 🥔 👗 🖻 🛍 🔊 (*) 🕅 🕅 👘 👘			
Project Navigator 🛛 🛱 🗙	📸 my_first_fpga.bdf* 🗵 🧼 simple_counter.v 🛛		
Cydone V: SCSEMASF31C6 Cydone V: SCSEMASF31C6 Fill my_first_fpga A File J [#] Design Units File J [#] Design Unit	<pre>//It has a single clock input and a 32-bit output port CLOCK_50, CLOCK_50; CLOCK_50; CLOCK_50; counter_out; reg [\$1:0] counter_out; end le begin counter_out <= #1 counter_out + 1;// increment counter end endmodule // end of module counter </pre>		•
8			
system / Processing /			Þ
System / Processing /		0% 0	0:00:00

Figure 3-4 The Verilog File of simple_counter.v

9. Save the file by choosing File > Save, pressing Ctrl + s, or by clicking the floppy disk icon.

10. Choose File > Create/Update > Create Symbol Files for Current File to convert the simple_counter.v file to a Symbol File (.sym). You use this Symbol File to add the HDL code to your BDF schematic.

The Quartus II software creates a Symbol File and displays a message (see Figure 3-5).





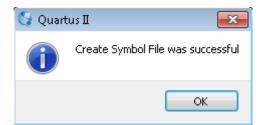


Figure 3-5 Create Symbol File was Successful

- 11. Click OK.
- 12. To add the simple_counter.v symbol to the top-level design, click the my_first_fpga.bdf tab.
- 13. Insert Symbol or click Add Symbol on the toolbar.
- 14. Double-click the Project directory to expand it.
- 15. Select the newly created simple_counter symbol by clicking it's icon.

You can also double-click in a blank area of the BDF to open the Symbol dialog box

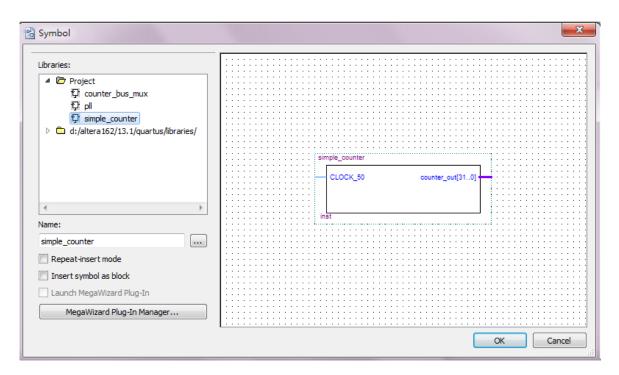


Figure 3-6 Adding the Symbol to the BDF

16. Click OK.

17. Move the cursor to the BDF grid; the symbol image moves with the cursor. Click to place the simple_counter symbol onto the BDF. You can move the block after placing it by simply clicking and dragging it to where you want it and releasing the mouse button to place it. See Figure 3-7.





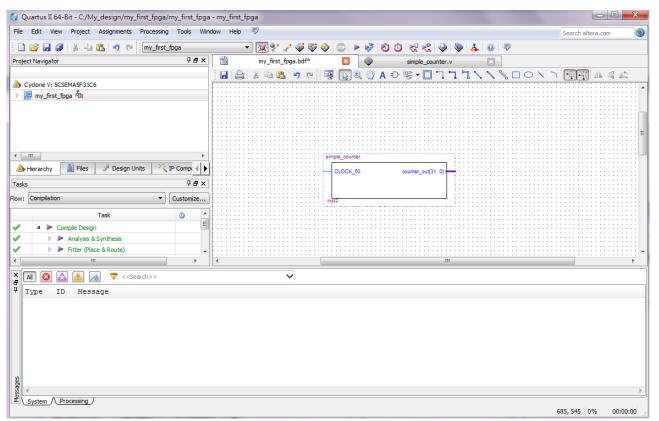


Figure 3-7 Placing the simple_counter symbol

18. Press the Esc key or click an empty place on the schematic grid to cancel placing further instances of this symbol.

19. Save your project regularly.

Using Quartus Add a PLL Megafunction

Megafunctions, such as the ones available in the LPM, are pre-designed modules that you can use in FPGA designs. These Altera-provided megafunctions are optimized for speed, area, and device family. You can increase

Efficiency by using a megafunction instead of writing the function yourself. Altera also provides more complex functions, called MegaCore functions, which you can evaluate for free but require a license file for use in production designs. This tutorial design uses a PLL clock source to drive a simple counter. A PLL uses the on-board oscillator (DE1-SoC Board is 50 MHz) to create a constant clock frequency as the input to the counter. To create the clock source, you will add a pre-built LPM megafunction named Altera PLL.

- 1. Choose Edit > Insert Symbol or click Add Symbol on the toolbar---
- 2. Click Megawizard Plug-in Manager. The MegaWizard® Plug-In Manager appears (see Figure





3-8).

🚭 MegaWizard	d Plug-In Manager [page 1]	×
*	The MegaWizard Plug-In Manager helps you create or modify design files that contain custom variations of megafunctions.	
	Which action do you want to perform?	
▲ `	Oreate a new custom megafunction variation	
	Edit an existing custom megafunction variation	
	Copy an existing custom megafunction variation	
	Copyright (C) 1991-2013 Altera Corporation	
	Cancel < Back Next > Finish	

Figure 3-8 Mega Wizard Plug-In Manager

- 3. Click Next.
- 4. In MegaWizard Plug-In Manager [page 2a], specify the following selections (see Figure 3-9):
 - a. Choose PLL> Altera PLL v13.1.

b. Under Which device family will you be using? Choose the Cyclone V for DE1-SoC development board.

c. Under Which type of output file do you want to create? Choose Verilog HDL.

d. Under What name do you want for the output file? Type pll.v at the end of the already created directory name.

e. Click Next.



Which megafunction would you like to customiz	e? Which device family will you be using? Cyclone V
Altera GPIO v13.1	Which type of output file do you want to create? AHDL VHDL VHDL Verilog HDL
MAX II/MAX V oscillator	What name do you want for the output file? C:/My_design/my_first_fpga/pll.v
 JTAG-accessible Extensions Memory Compiler PLL Altera PLL Reconfig v13.1 Altera PLL v13.1 Arria 10 FPLL v13.1 Arria 10 Transceiver ATX PLL 	 Return to this page for another create operation Note: To compile a project successfully in the Quartus II software, your design files must be in the project directory, in a library specified in the Libraries page of the Options dialog box (Tools menu), or a library specified in the Libraries page of the Settings dialog box (Assignments menu). Your current user library directories are:
Arria 10 Transceiver CMU PLL Click to Open IP MegaStore	

Figure 3-9 MegaWizard Plug-In Manager [page 2a] Selections

5. In the MegaWizard Plug-In Manager window, make the following selections (see Figure 3-10).

- a. The device speed grade choose 6 for DE1-SoC.
- b. Set the frequency of the refclk input 50 MHz.



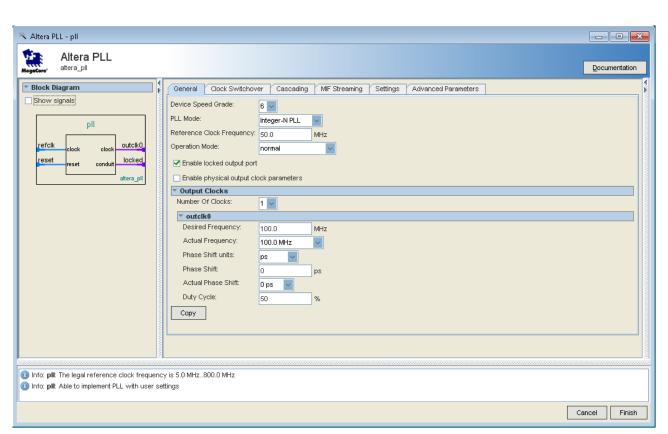


Figure 3-10 MegaWizard Plug-In Manager Selections

6. Turn off the Enable locked output port option on MegaWizard . As you turn it off, pin disappear from the PLL block's graphical preview. See Figure 3-11 for an example.



ter asiC

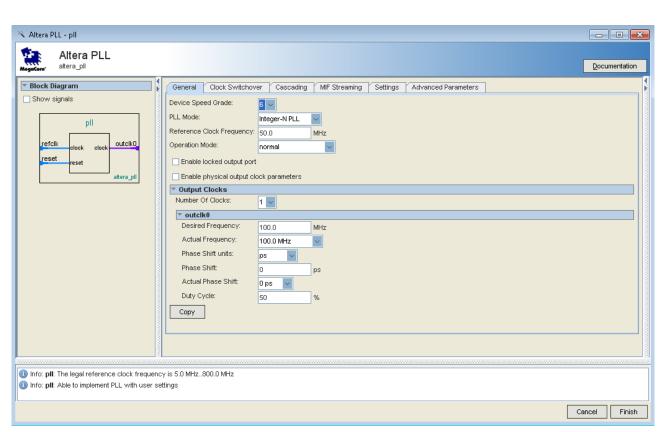


Figure 3-11 MegaWizard Plug-In Manager Selections

7. Then, make the following selection (see Figure 3-12).

Desired Frequency input 5.0

ter as IC



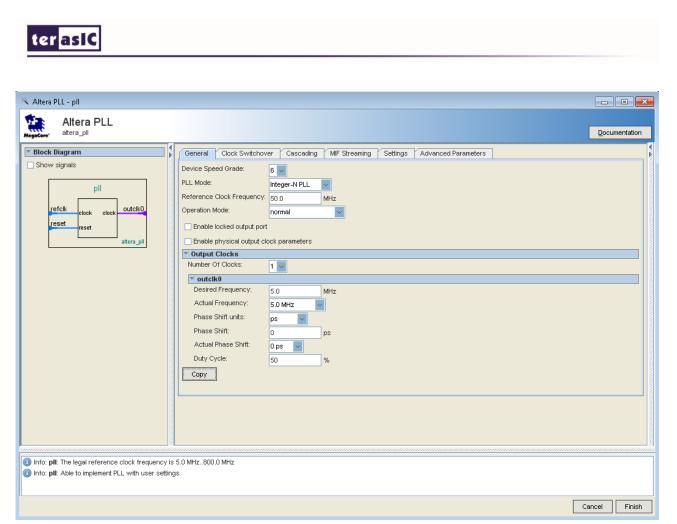


Figure 3-12 MegaWizard Plug-In Manager Selections

8. Click Finish . It leads to Figure 3-13. Click Exit.

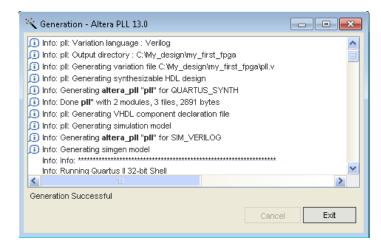


Figure 3-13 PLL Generation

The Symbol window opens, showing the newly created PLL megafunction. See Figure 3-14.



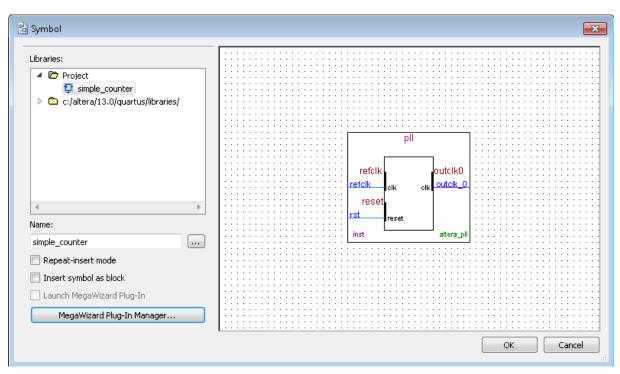


Figure 3-14 PLL Symbol

11. Click OK and place the pll symbol onto the BDF to the left of the simple_counter symbol. You can move the symbols around by holding down the left mouse button, helping you ensure that they line up properly. See **Figure 3-15**.

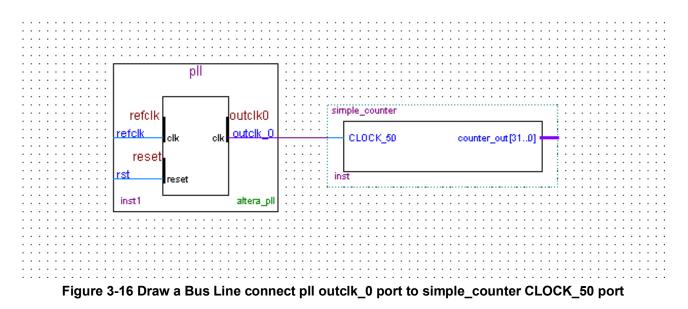
File Edit Verwinder Verwinder Verwinder	💱 Quartus II 64-Bit - C:/My_design/my_first_fpga/my_first_fpg	a - my_first_fpga		
Project Navigator 9 S X Project Navigator Cyclone V: SCSEN49F3106 Provide V: SCSEN49F3106 P	File Edit View Project Assignments Processing Tools Window Help 💎 Search altera.com 🔇			
Cyclore Y: SCSEMASF31C6 Percent SCSE	🗋 💕 🛃 🥥 👗 🗈 🛍 🌱 (*) 🕅 my_first_fpga			
 Cvdore Y: SCSMAF3165 	Project Navigator 🗘 🗗 🛪	📸 my_first_fpga.bdf* 🛛 🧼 simple_counter.v 🗵		
Image: Contract of the second seco				
Task ? ? ? * ? * ? * ? *	→ Hierarchy Files d ^{ag} Design Units × IP Comp. () Tasks ♀ ♂ ★	refcik refcik ok outcik0 reset reset reset		
Image: Constraint of the state of the st	Task Image: Comple Design ?			
System / Processing /	sebessi -		•	

Figure 3-15 Place the PLL Symbol

terasic

12. Move the mouse so that the cursor (also called the selection tool) is over the pll symbol's outclk_0 output pin. The orthogonal node tool (cross-hair) icon appears.

13. Click and drag a bus line from the outclk_0 output to the simple_counter clock input. This action ties the pll output to the simple_counter input (see Figure 3-16).



- 14. Add an input pin and an output bus with the following steps:
- a. Choose Edit > Insert Symbol.
- b. Under Libraries, select quartus/libraries > primitives > pin > input. See Figure 3-17
- c. Click OK

If you need more room to place symbols, you can use the vertical and horizontal scroll bars at the edges of the BDF window to view more drawing space.



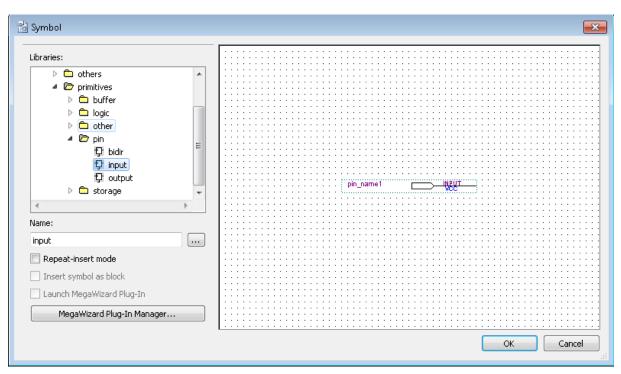


Figure 3-17 Input pin symbol

d. Place the new pin onto the BDF so that it is touching the input to the pll symbol.

e. Use the mouse to click and drag the new input pin to the left; notice that the ports remain connected as shown in Figure 3-18.

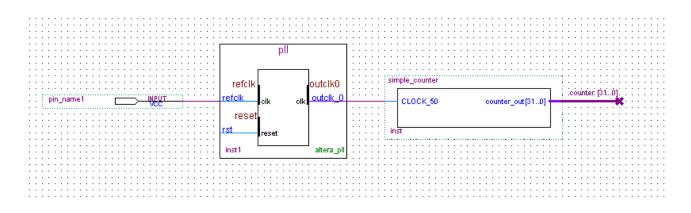


Figure 3-18 Connecting the PLL symbol and Input port

f. Change the pin name by double-clicking pin_name and typing CLOCK_50 (see **Figure 3-19**). This name correlates to the oscillator clock that is connected to the FPGA.

g. Using the Orthogonal Bus tool, draw a bus line connected on one side to the simple_counter output port, and leave the other end unconnected at about 4 to 8 grid spaces to the right of the simple_counter.



terasic



沿 Pin Properties	
General Form	nat
To create multip (For example: "r	le pins, enter a name in AHDL bus notation name[30]"), or enter a comma-seperated list of names.
Pin name(s):	CLOCK_50
Default value:	VCC 🔹
	OK Cancel Help

Figure 3-19 Change the input port name

h. Right-click the new output bus line and choose Properties.

i. Using the Orthogonal Bus tool, draw a bus line connected on one side to the simple_counter output port, and leave the other end unconnected at about 6 to 8 grid spaces to the right of the simple_counter.

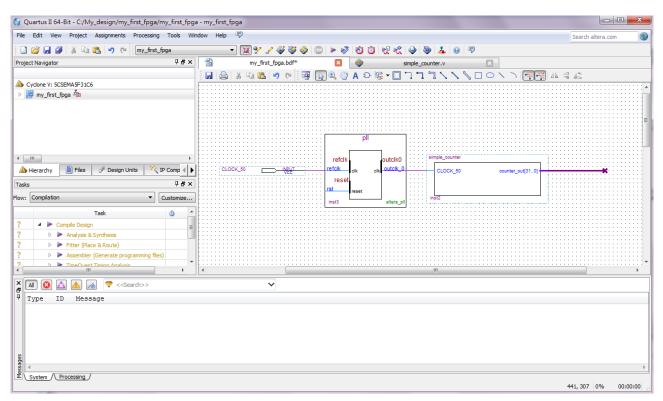
j. Type counter [31..0] as the bus name (see **Figure 3-20**). The notation [X ..Y] is the Quartus II method for specifying the bus width in BDF schematics, where X is the most significant bit (MSB) and Y is the least significant bit (LSB).

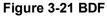
k. Click OK. Figure 3-21 shows the BDF.



💾 Bus Prop	roperties	×
General	al Font Format	
Name:	e: counter [310]	
🔳 Hide	lide name in block design file.	
	OK Cancel	Help

Figure 3-20 Change the output BUS name





3.2 Add a Multiplexer

This design uses a multiplexer to route the simple_counter output to the LED pins on the DE1-SoC development board. You will use the MegaWizard Plug-In Manager to add the multiplexer, lpm_mux. The design multiplexes two variations of the counter bus to four LEDs on the DE1-SoC development board.

- 1. Choose Edit > Insert Symbol.
- 2. Click Megawizard Plug-in Manager.
- 3. Click Next.
- 4. Choose Installed Plug-Ins > Gates > LPM_MUX.

5. Choose the device family that corresponds to the device on the development board you are using, choose Verilog HDL as the output file type, and name the output file counter_bus_mux.v (see Figure 3-22).

6. Click Next.

🌍 MegaWizard Plug-In Manager (page 2a)	
Which megafunction would you like to customize? Select a megafunction from the list below	Which device family will you be using? Cyclone V Which type of output file do you want to create? AHDL AHDL VHDL Verilog HDL Verilog HDL What name do you want for the output file? C://Wy_design/my_first_fpga/counter_bus_mux.v Output files will be generated using the classic file structure Image: Complete a project successfully in the Quartus II software, your design files must be in the project directory, in a library specified in the Libraries page of the Options dialog box (Tools menu), or a library specified in the Libraries page of the Settings dialog box (Assignments menu). Your current user library directories are: Image: Complete a project success are:
	Cancel < Back Next > Finish

Figure 3-22 Selecting Ipm_mux

7. Under How many 'data' inputs do you want? Select 2 inputs (default).



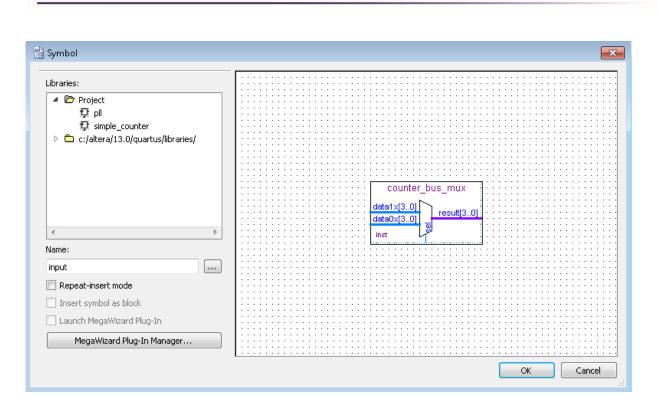
8. Under How 'wide' should the data input and result output be? Select 4 (see Figure 3-23).

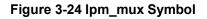
🔨 MegaWizard Plug-In Manager [page 3 of 5]
LPM_MUX About Documentation
1 Parameter Settings
Currently selected device family: Cyclone V Match project/default How many 'data' inputs do you want? 2 How wide should the 'data' input and 4 bits Do you want to pipeline the multiplexer? No Yes, I want an output latency of 1 Create an asynchronous Clear input Create a Clock Enable input
Resource Usage 1 lpm_mux Cancel < Back

Figure 3-23 lpm_mux settings

- 9. Click Next.
- 10. Click Finish twice. The Symbol window appears (see Figure 3-24 for an example).







11. Click OK

terasic

12. Place the counter_bus_mux symbol below the existing symbols on the BDF. See Figure 3-25

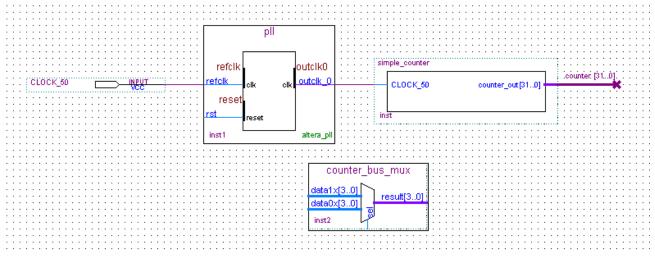


Figure 3-25 Place the lpm_mux symbol

13. Add input buses and output pins to the counter_bus_mux symbol as follows:

a. Using the Orthogonal Bus tool, draw bus lines from the data1x[3..0] and data0x[3..0]

Input ports to about 8 to 12 grid spaces to the left of counter_bus_mux.



b. Draw a bus line from the result [3..0] output port to about 4 to 8 grid spaces to the right of counter_bus_mux.

c. Right-click the bus line connected to data1x[3..0] and choose Properties.

d. Name the bus counter[26..23], which selects only those counter output bits to connect to

the four bits of the data1x input.

Because the input busses to counter_bus_mux have the same names as the output bus from simple_counter, (counter[x .. y]) the Quartus II software knows to connect these busses.

e. Click OK.

f. Right-click the bus line connected to data0x[3..0] and choose Properties.

g. Name the bus counter [24..21], which selects only those counter output bits to connect to the four bits of the data1x input.

h. Click OK. Figure 3-26 shows the renamed buses.

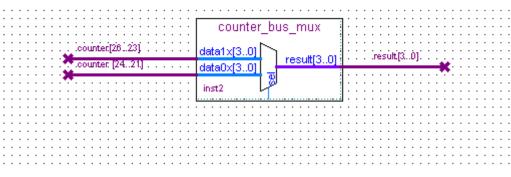


Figure 3-26 Renamed counter_bus_mux Bus Lines

If you have not done so already, save your project file before continuing.

- 14. Choose Edit > Insert Symbol.
- 15. Under Libraries, double-click quartus/libraries/ > primitives > pin > output (see Figure 3-27).



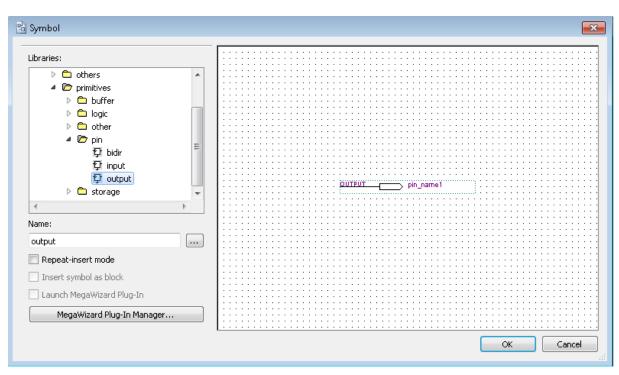


Figure 3-27 choose an output pin

16. Click OK.

erasic

- 17. Place this output pin so that it connects to the counter_bus_mux result [3..0] bus output line.
- 18. Rename the output pin as LED [3..0] as described in steps 13 c and d. (see Figure 3-28).

		::
counter_bus_mux		• •
data1×[30] result[30]	.result[30].	· · · ·
data0x[30]		: :
inst2		::
		: :
		• •

Figure 3-28 Rename the output pin

- 19. Attach an input pin to the multiplexer select line using an input pin:
- a. Choose Edit > Insert Symbol.
- b. Under Libraries, double-click quartus/libraries/ > primitives > pin > input.
- c. Click OK.
- 20. Place this input pin below counter_bus_mux.



- 21. Connect the input pin to the counter_bus_mux sel pin.
- 22. Rename the input pin as KEY [0].
- 23. Place another input pin at the left of the pll and rename it as KEY[1] (see Figure 3-29).
- 24. Choose Edit > Insert Symbol.
- 25. Under Libraries, double-click quartus/libraries/ > primitives >logic > not (see Figure 3-30).
- 26. Click OK.
- 27. Place this not pin so that it connects to KEY[1] and the pll rst (see Figure 3-31).

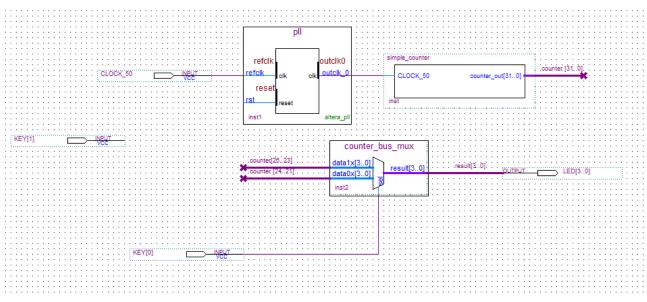
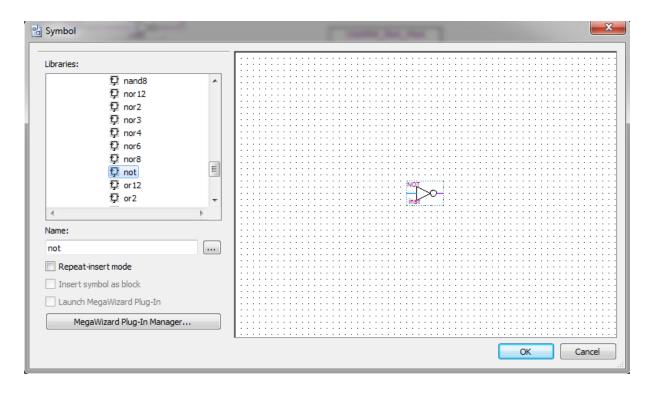
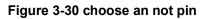


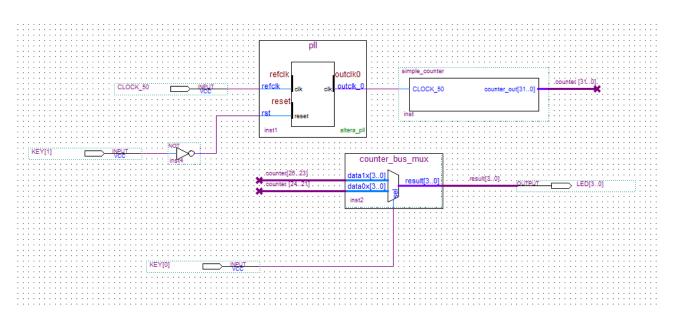
Figure 3-29 Adding the KEY [1:0] Input Pin













You have finished adding symbols to your design. You can add notes or information to the project as text using the Text tool on the toolbar (indicated with the A symbol). For example, you can add the label "OFF = SLOW, ON = FAST" to the KEY [0] input pin and add a project description, such as "My First FPGA Project."



3.3 Assign the Pins

In this section, you will make pin assignments. Before making pin assignments, perform the following steps:

1. Choose Processing > Start > Start Analysis & Elaboration in preparation for assigning pin locations.

2. Click OK in the message window that appears after analysis and elaboration completes.

To make pin assignments that correlate to the KEY[1:0] and CLOCK_50 input pins and LED[3..0] output pin, perform the following steps:

1. Choose Assignments > Pin Planner, which opens the Pin Planner, a spreadsheet-like table of specific pin assignments. The Pin Planner shows the design's six pins. See Figure 3-32.

Edit View Processing	Tools Window Helj	p 💎			Search alter	a.com
Groups		4 8 ×		Top V Ia-r - WTas Occlose V - SC SXFC 60		
Named: *		•				
Node Name	Direction	Locat 🔺		895595555656666	× • • • •	
▷ ➡ KEY[10]	Input Group		:	000000000000000 V	· · · · · · · · · · · · · · · · · · ·	
▷ 🚟 LED[30]	Output Group	-	_	BEER WEAKOW	$\nabla \nabla^{\Delta \nabla} \nabla \nabla =$	
		E.	:	HELE CONTRACTOR		
Report		₽ æ ×		HELEVEX AND A CONTRACT OF A CO	$\nabla \nabla_{\Theta X \otimes X $	
Report not available					Second Second	
Report not available				HENRY X HARA AND		
				Address System de o	66000000000000000000000000000000000000	
Named: *	Edit: 🗶 🗸				Filter: Pir	ns: all
8	Edit: 🗶 ✔	Location	I/O Bank	VREF Group	Filter: Pir	ns: all Reserved
P Node Name in_ CLOCK_50		Location	I/O Bank	VREF Group		
라 및 Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard	
Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard 2.5 V (default)	
Node Name	Direction Input Input	Location	I/O Bank	VREF Group	I/O Standard 2.5 V (default) 2.5 V (default)	
Node Name Node Name In_ CLOCK_50 In_ KEY[1] In_ KEY[0] Out LED[3]	Direction Input Input Input Input	Location	I/O Bank	VREF Group	I/O Standard 2.5 V (default) 2.5 V (default) 2.5 V (default)	
Node Name <u>in_cLock_50 [in_kEY[1] [in_kEY[0] State State (LED[3] State (LED[2] </u>	Direction Input Input Input Output Output	Location	I/O Bank	VREF Group	I/O Standard 2.5 V (default) 2.5 V (default) 2.5 V (default) 2.5 V (default) 2.5 V (default)	
Image: Node Name in	Direction Input Input Input Output Output Output	Location	I/O Bank	VREF Group	I/O Standard 2.5 V (default) 2.5 V (default) 2.5 V (default) 2.5 V (default) 2.5 V (default) 2.5 V (default)	
P Node Name inCLOCK_50	Direction Input Input Input Output Output Output Output Output Output		I/O Bank	VREF Group	I/O Standard 2.5 V (default) 2.5 V (default) 2.5 V (default) 2.5 V (default) 2.5 V (default) 2.5 V (default) 2.5 V (default)	

Figure 3-32 Pin Planner Example

2. In the Location column next to each of the seven node names, add the coordinates (pin numbers) as shown in Table 3-1 for the actual values to use with your DE1-SoC board.





Pin Name	FPGA Pin Location
CLOCK_50	AF14
KEY[1]	AA15
KEY[0]	AA14
LED[3]	V18
LED[2]	V17
LED[1]	W16
LED[0]	V16

Table	3-1	Pin	Information	Setting
-------	-----	-----	-------------	---------

Double-click in the Location column for any of the seven pins to open a drop-down list and type the location shown in the table alternatively, you can select the pin from a drop-down list. For example, if you type F1 and press the Enter key, the Quartus II software fills in the full PIN_F1 location name for you. The software also keeps track of corresponding FPGA data such as the I/O bank and VREF Group. Each bank has a distinct color, which corresponds to the top-view wire bond drawing in the upper right window. See **Figure 3-33**.

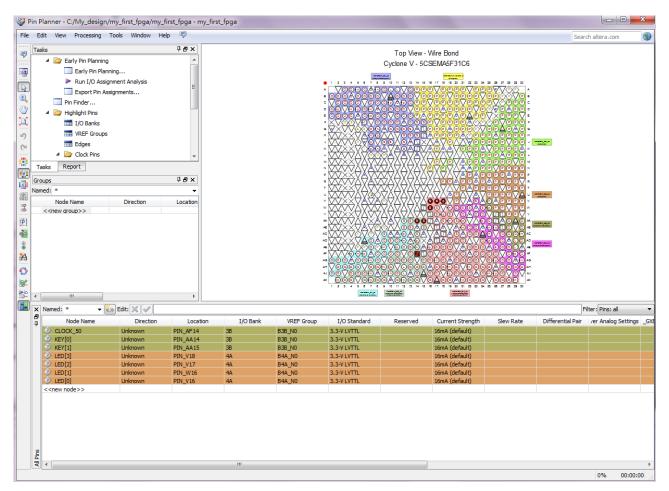


Figure 3-33 Completed Pin Planning Example

Now, you are finished creating your Quartus II design!



DE1-SoC My First FPGA

ter <mark>asIC</mark>

3.4 Create a Default TimeQuest SDC File

Timing settings are critically important for a successful design. For this tutorial you will create a basic Synopsys Design Constraints File (.sdc) that the Quartus II TimeQuest Timing Analyzer uses during design compilation. For more complex designs, you will need to consider the timing requirements more carefully.

To create an SDC, perform the following steps:

- 1. Open the TimeQuest Timing Analyzer by choosing Tools > TimeQuest Timing Analyzer.
- 2. Choose File > New SDC file. The SDC editor opens.
- 3. Type the following code into the editor:

create_clock -name "CLOCK_50" -period 20.000ns [get_ports {CLOCK_50}]

derive_pll_clocks

derive_clock_uncertainty

4. Save this file as my_first_fpga.sdc (see Figure 3-34).

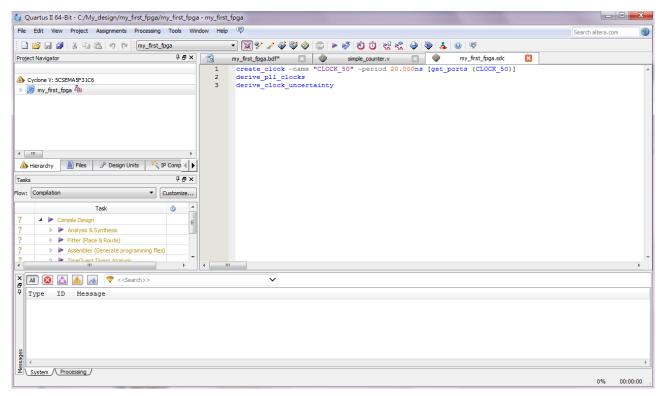


Figure 3-34 Default SDC



Naming the SDC with the same name as the top-level file except for the .sdc extension causes the Quartus II software to using this timing analysis file automatically by default. If you used another name, you would need to add the SDC to the assignments file list.





Chapter 4

Compile and Verify Your Design

After creating your design you must compile it. Compilation converts the design into a bitstream that can be downloaded into the FPGA. The most important output of compilation is an SRAM Object File (.sof), which you use to program the device. The software also generates other report files that provide information about your code as it compiles.

4.1 Compile Your Design

If you want to store .SOF in memory device (such as flash or EEPROMs), you must first convert the SOF to a file type specifically for the targeted memory device.

Now that you have created a complete Quartus II project and entered all assignments, you can compile the design.

In the Processing menu, choose Start Compilation or click the Play button on the toolbar.

If you are asked to save changes to your BDF, click Yes.

While compiling your design, the Quartus II software provides useful information about the compilation (see Figure 4-1).





Quartus II 64-Bit - C:/My_design/my_first_fpga/my_first_fpga					~
File Edit View Project Assignments Processing Tools Wir	ndow Help 💎			Search altera.com	-
🗋 🚰 🛃 🕉 🗟 🛍 🤊 (*) 🕅 my_first_fpga		े। 🕾 😤 । 🦫 🕹 । 🥹	-		
Project Navigator 🛛 🕹 🛪	📸 my_first_fpga.bdf 区 🧇	simple_counter.v	Compilation Report - my_first_fpga		
	Table of Contents	🗗 Flow Summary			
Cyclone V: 5CSEMA5F31C6	Flow Summary	Flow Status	In progress - Fri Dec 12 13:28:36 2014		
4 😸 my_first_fpga 📠	Flow Settings	Quartus II 64-Bit Version	13.1.0 Build 162 10/23/2013 SJ Full Version		
abd simple_counter:inst	Flow Non-Default Global Settings	Revision Name	my_first_fpga		
pll:inst1	Flow Elapsed Time	Top-level Entity Name Family	my_first_fpga Cvdone V		
counter_bus_mux:inst5	Flow OS Summary	Device	SCSEMA5F31C6		
	Flow Log	Timing Models	Preliminary		
	Analysis & Synthesis	Logic utilization (in ALMs)	N/A		
	Analysis & Synulesis Fitter	Total registers	27		
	Flow Messages	Total pins	7		
🛆 Hierarchy 📄 Files 🖓 Design Units 🥂 IP Comp 4 🕨	Flow Pressages Flow Suppressed Messages	Total virtual pins	0		
asks 무문×	V Plow Suppressed Messages	Total block memory bits Total DSP Blocks	0		
		Total HSSI BX PCSs	0		
ow: Compilation Customize		Total HSSI PMA RX Deserializers	0		
Task 🕚 ^		Total HSSI TX PCSs	0		
20% 4 > Compile Design 00:00:0: ==		Total HSSI TX Channels	0		
Analysis & Synthesis 00:00:0		Total PLLs	1		
0% > Fitter (Place & Route) 00:00:00		Total DLLs	0		
0% Assembler (Generate programming files) 00:00:00					
0% b TimeOuest Timing Analysis 00:00:01					
4 III III III III III III III III III I					
🖌 😰 🛆 🔷 🍺 💎 < <search>></search>	~				
Type ID Message					
Running Quartus II 64-Bit Fitt	er				
	ttings files=offwrite settings files=off	my first fpga -c my f	irst fpga		
qfit2 default script.tcl versi					
Project = my_first_fpga					
Revision = my_first_fpga					
11104 Parallel Compilation has detection	tted 8 hyper-threaded processors. However, t	the extra hyper-threade	d processors will not be used by def	ault. Parallel Compi	lat
11104 Parallel Compilation has detected					F.
System / Processing (43) /					_
itarts a new compilation				360, 415 20% 🎇 00:0	10.04

Figure 4-1 Compilation Message for project

When compilation is complete, the Quartus II software displays a message. Click OK to close the message box.

The Quartus II Messages window displays many messages during compilation. It should not display any critical warnings; it may display a few warnings that indicate that the device timing information is preliminary or that some parameters on the I/O pins used for the LEDs were not set. The software provides the compilation results in the Compilation Report tab as shown in **Figure 4-2**.





Flow Summary	
Flow Status	Successful - Fri Dec 12 13:29:23 2014
Quartus II 64-Bit Version	13.1.0 Build 162 10/23/2013 SJ Full Version
Revision Name	my_first_fpga
Top-level Entity Name	my_first_fpga
Family	Cyclone V
Device	5CSEMA5F31C6
Timing Models	Preliminary
Logic utilization (in ALMs)	16 / 32,070 (< 1 %)
Total registers	27
Total pins	7 / 457 (2 %)
Total virtual pins	0
Total block memory bits	0 / 4,065,280 (0 %)
Total DSP Blocks	0/87(0%)
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI TX Channels	0
Total PLLs	1/6(17%)
Total DLLs	0/4(0%)



4.2 Program the FPGA Device

After compiling and verifying your design you are ready to program the FPGA on the development board. You download the SOF you just created into the FPGA using the USB-BlasterII circuitry on the board. Set up your hardware for programming using the following steps:

a) Connect the power supply cable to your board and to a power outlet.

b) For the DE1-SoC board, connect the USB-BlasterII (included in your development kit) to J13 and the USB cable to the USB-BlasterII. Connect the other end of the USB cable to the host computer.

42

Refer to the getting started user guide for detailed instructions on how to connect the cables.

c) Turn the DE1-SoC board on using the on/off switch.

Program the FPGA using the following steps.

1. Choose Tools > Programmer. The Programmer window opens. See Figure 4-3.



	Device	Checksum	Usercode	Program/ Configure	Verify	Blank- Check	Examine	Security Bit	Erase	ISP CLAMP	
my_first_fpga.sof	5CSEMA5F31	00AF8727	00AF8727	\checkmark							

Figure 4-3 Programmer Window

2. Click Hardware Setup.

3. If it is not already turned on, turn on the DE1-SoC [USB-1] option under currently selected hardware. See Figure 4-4.

Hardware Setup				×
Hardware Settings JTAG Se	ettings			
Select a programming hardware hardware setup applies only to	e setup to use when prog the current programmer	ramming devices window.	s. This programming	
Currently selected hardware:	No Hardware			•
Available hardware items	·			
Hardware	Server	Port		Add Hardware
DE-SoC	Local	USB-1		Remove Hardware
				Close

Figure 4-4 Hardware Setting

- 4. Click Close.
- 5. Click Auto Detect to detect all the devices on the JTAG chain.



ter <mark>aslC</mark>

6. Select 5CSEMA5 and click OK.

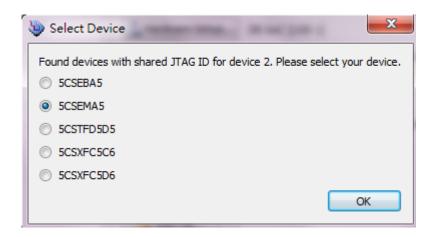


Figure 4-5 Select FPGA Device

7. Click Yes to match device list.

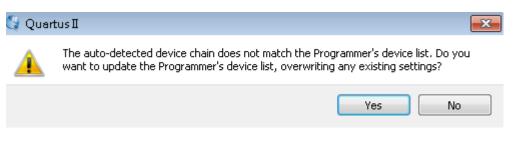


Figure 4-6 Warning for device list not match

8. Both HPS and FPGA will be list on the programmer. Select FPGA device and click Change File a .sof file.

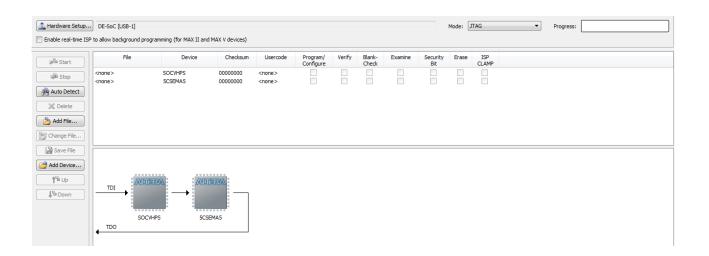






Figure 4-7 Programmer window

9. Select the my_first_fpga.sof file from the project directory (see Figure 4-8), select to Program/Configure and click Start to program .sof file into FPGA.

e ISP to allow backgrou											
File		Device	Checksum	Usercode	Program/ Configure	Verify	Blank- Check	Examine	Security Bit	Erase	ISP CLAMP
<none></none>	SOCVH	IPS	00000000	<none></none>							
my_first_fpga.se	f 5CSEM/	A5F31	00AF8727	00AF8727	V						
		AD									
		→ ×									
		→									

Figure 4-8 Downloading Complete

Congratulations, you have created, compiled, and programmed your first FPGA design! The compiled SRAM Object File (.sof) is loaded onto the FPGA on the development board and the design should be running.

4.3 Verify The Hardware

When you verify the design in hardware, you observe the runtime behavior of the FPGA hardware design and ensure that it is functioning appropriately.

Verify the design by performing the following steps:

1. Observe that the four development board LEDs appear to be advancing slowly in a binary count pattern, which is driven by the simple_counter bits [26..23].

The LEDs are active high, therefore, when counting begins all LEDs are turned off (the 0000 state).

2. Press and hold KEY [0] on the development board and observe that the LEDs advance more quickly. Pressing this KEY causes the design to multiplex using the faster advancing part of the counter (bits [24..21]).



ter <mark>asIC</mark>

3. If other LEDs emit faintness light, Choose Assignments > Device. Click Device and Pin Options. See Figure 4-9.

Device family			Show in 'A	vailable devices' list					
Family: Cyclone V (E/GX/GT/	/SX/SE/ST)		 Package: 	Any		•			
			Pin count:	Pin count: Any					
Devices: All			Fin count.	Pin count:					
Target device			Speed gra	ide: Any		•			
-			Name filte	r:					
Auto device selected by t	he Fitter								
Specific device selected in	n 'Available devices' lis	t	Show a	advanced devices					
Other: n/a									
O dici ina			Device and	Pin Options					
vailable devices:									
Name	Core Voltage	ALMs	User I/Os	GXB Channel PMA	GXB Channel PCS				
CSEBA6U2317S	1.1V	41910	314	0	0				
SCSEMA2U23A7 (Advanced)	1.1V	9430	314	0	0				
SCSEMA2U23C6 (Advanced)	1.1V	9430	314	0	0				
SCSEMA2U23C7 (Advanced)	1.1V	9430	314	0	0				
SCSEMA2U23C8 (Advanced)	1.1V	9430	314	0	0				
SCSEMA2U23I7 (Advanced)	1.1V	9430	314	0	0				
SCSEMA4U23A7 (Advanced)	1.1V	15880	314	0	0				
SCSEMA4U23C6 (Advanced)	1.1V	15880	314	0	0				
SCSEMA4U23C7 (Advanced)	1.1V	15880	314	0	0				
SCSEMA4U23C8 (Advanced)	1.1V	15880	314	0	0				
SCSEMA4U23I7 (Advanced)	1.1V	15880	314	0	0				
SCSEMA5F31A7	1.1V	32070	457	0	0				
SCSEMA5F31C6	1.1V	32070	457	0	0				
CSEMA5E31C7	1 1V	32070	457	0	0				

Figure 4-9 Device and Options

Choose unused pins. Reserve all unused pins: Choose the As input tri-stated option. See Figure 4-10.





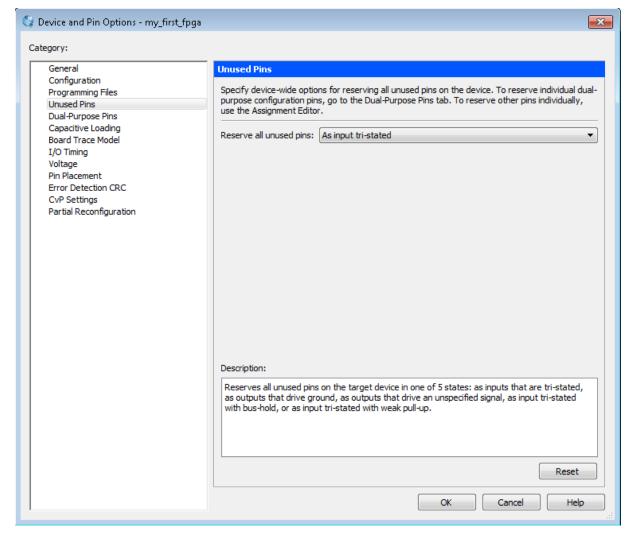


Figure 4-10 Setting unused pins

Click twice OK.

4. In the Processing menu, choose Start Compilation. After the compile, Choose Tools > Programmer. Select the my_first_fpga.sof file from the project directory. Click Start. At this time you could find the other LEDs are unlighted.





Chapter 5 *Appendix*

5.1 Headquarter & Branches

- Tel : +886-3-575-0880
- Fax : +886-3-572-6690
- Add: 9F, No.176, Sec.2, Gongdao 5th Rd, East Dist, Hsinchu City, Taiwan 300-70
- Email : sales@terasic.com / support@terasic.com

