My First FPGA for

Altera DE2-115 Board



Chapter 1	<i>Introduction</i>
	1.1 Design Flow 1
	1.2 Before You Begin
	1.3 What You Will Learn5
Chapter 2	Assign The Device 7
	2.1 Assign The Device7
Chapter 3	Design Entry 11
	3.1 Add a PLL Megafunction11
	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
	4.1 Compile Your Design
	4.2 Program the FPGA Device
	4.3 Verify The Hardware
Chapter 5	DE2-115 System Builder
	5.1 Headquarter & Branches

Chapter 1



This tutorial provides comprehensive information that will help you understand how to create a FPGA design and run it on you DE2-115 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.

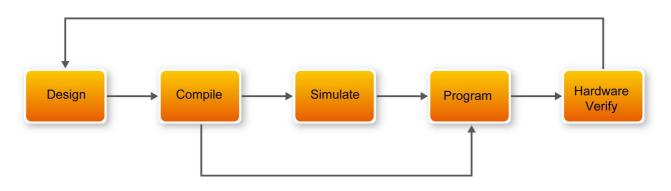


Figure 1-1 Design Flow

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 9.1 sp2 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 DE2-115 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-Blaster driver, Plug in the 12-volt adapter to provide power to the board. Use the USB cable to connect the leftmost USB connector (the one closest to the power switch) on the DE2-115 board to a USB port on a computer that runs the Quartus II software. Turn on the power switch on the DE2-115 board.

The computer will recognize the new hardware connected to its USB port, but it will be unable to proceed if it does not have the required driver already installed. The DE2-115 board is programmed by using Altera USB-Blaster mechanism. If the USB-Blaster driver is not already installed, the New Hardware Wizard in **Figure 1-2** will appear.

ter <mark>asIC</mark>



Figure 1-2 Found New Hardware Wizard

Since the desired driver is not available on the Windows Update Web site, select "No, not this time" in response to the question asked and click Next. This leads to the window in **Figure 1-3**.



Figure 1-3 The driver is found in a specific location

The driver is available within the Quartus II software. Hence, select Install from a specific location and click Next to get to Figure 1-4.



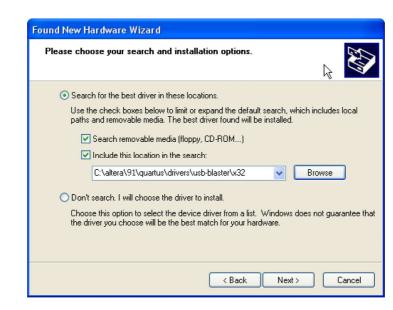


Figure 1-4 Specify the location of the driver

Now, choose Search for the best driver in these locations and click Browse to get to the pop-up box in **Figure 1-5** Find the desired driver, which is at location C:\altera\91\quartus\drivers\usb-blaster. Click OK and then upon returning to **Figure 1-4** click Next. At this point the installation will commence, but a dialog box in **Figure 1-6** will appear indicating that the driver has not passed the Windows Logo testing. Click Continue Anyway.

Browse For Folder	? 🗙
Select the folder that contains drivers for your hard	ware.
🖃 🚞 quartus	~
🗉 🧰 bin	
🖽 🛅 bin64	
🗉 🧰 common	
🖽 🧰 cusp	=
🖃 🫅 drivers	
i386	
🗉 🛅 sentinel	
🖃 🫅 usb-blaster	
🗁 x32	
🛅 x64	~
To view any subfolders, click a plus sign above.	
ОК Са	ncel

Figure 1-5 Browse to find the location



erias C





Figure 1-6 There is no need to test the driver

The driver will now be installed as indicated in **Figure 1-7** Click Finish and you can start using the DE2-115 board.



Figure 1-7 The driver is installed

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 DE2-115 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 (Please reference DE2-115 tut_quartus_intro_verilog document), 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 DE2-115 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 DE2-115 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 Project	Tizard: Introduction 🔀
The New Proje following:	ect Wizard helps you create a new project and preliminary project settings, including the
ł	Project name and directory Name of the top-level design entity Project files and libraries Target device family and device EDA tool settings
the Settings co	ge the settings for an existing project and specify additional project-wide settings with ommand (Assignments menu). You can use the various pages of the Settings dialog box hality to the project.
☐ Don't show	v me this introduction again
	< Back Next > Finish 取消







- 2. Click Next.
- 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, $E:My_design_my_first_fpga$.
- 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**.

E:\My_design\my_first_fpga				
What is the name of this project?				
my_first_fpga				
≪hat is the name of the top-level exactly match the entity n{∿ge in t		project? This na	ime is case sensit	tive and must
my_first_fpga				
	2			
Use Existing Project Settings				
Use Existing Project Settings				
Use Existing Project Settings				
Use Existing Project Settings]			
Use Existing Project Settings				
Use Existing Project Settings]			
Use Existing Project Settings]			
Use Existing Project Settings]			
Use Existing Project Settings]			
Use Existing Project Settings]			

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.





New Project Vizard:	Family	& Devic	e Sett	ings [p:	age 3 o	f 5]	X
Select the family and device y	ou want to l	target for co	mpilation.				
– Device family				-Show in 'Av	vailable dev	ice' list—	
Eamily: Cyclone IV E			_	Package:	Any		-
						_	4
Devices: All				Pin <u>c</u> ount:	Any		-
Target device				Sp <u>e</u> ed grad	le: Any	•	-
C Auto device selected by	y the Fitter			Show a	dvanced de	evices	
 Specific device selecte 		le devices'	list		py compatit		
A <u>v</u> ailable devices:							
Name	Core v	LEs	User I/	Memor	Embed	PLL	~
EP4CE115F23C8L	1.0V	114480	281	3981312	532	4	
EP4CE115F23C9L	1.0V	114480	281	3981312	532	4	
EP4CE115F23I7	1.2V	114480	281	3981312	532	4	
EP4CE115F23I8L	1.0V	114480	281	3981312	532	4	
EP4CE115F29C7 EP4CE115F29C8	1.2V	114480	529 529	3981312	532 532	4	
EP4CE115F29C8	1.2V 1.0V	114480 114480	529 529	3981312 3981312	532 532	4 4	-
EP4CE115F29C9L	1.0V 1.0V	114460	525 529	3981312	532 532	4 1	~
<						>	
Companion device							
							7
HardCopy:							<u> </u>
🔽 Limit DSP & RAM to Har	Limit DSP & RAM to HardCopy device resource						
		< Back	Next	> Fi	ni sh	取礼	肖

Figure 2-3 Specify the Device Example

h. Click Finish.

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 - E:/My_design/my_first	_fpga_review/my_first_fpga - my_first_fpga 📃 🗖 🔀
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>P</u> roject <u>A</u> ssignments Processi	ng <u>T</u> ools <u>M</u> indow <u>H</u> elp
🗋 😅 🖬 🎒 🏯 👗 🐴 💼 📾 🖂 🔤 🔤	first_fpga 🔹
X / Ø Ø Ø 💿 🕨 🗞 👦 🔿	
Project Navigator	
Entity	
Cyclone IV E: EP4CE115F29C7	
→ my_first_fpga 🖧	
Hierarchy 🖹 Files 🗗 Design Units	
Tasks ×	
Flow: Compilation	OILA D'TIC'IL
Task 🖌 🔥	OUARTUS [®] II
🗆 🕨 Compile Design	
😑 🕨 Analysis & Synthesis Edit Settings	Version 9.1
View Report	Download New
Analysis & Elaboration	Software Release
🕀 🛶 🕨 Partition Merge	
A Watlist Viaware	Ocumentation
× Type Message	
System [1] / Processing λ Extra Info λ Info λ	Warning λ Critical Warning λ Error λ Suppressed λ Flag /
Message: 1 of 1.	
Elinessage. For T	
For Help, press F1	

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.

New	×
SOPC Builder System Design Files AHDL File Block Diagram/Schematic File State Machine File SystemVerilog HDL File SystemVerilog HDL File Verilog HDL File Verilog HDL File Verilog HDL File VHDL File VHDL File VHDL File Verification/Debugging Files In-System Sources and Probes File Logic Analyzer Interface File SignalT ap II Logic Analyzer File Vector Waveform File Other Files AHDL Include File Block Symbol File Chain Description File	
- Synopsys Design Constraints File	~
OK Cance	el

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).

🐇 Quartus II - E:/My_design	/my_first_fpga/my_first	_fpga - my_first_fpga	. 🔳 🗖 🗙
📲 File Edit View Project Assign	uments Processing Tools Mindo	w <u>H</u> elp	- 8 ×
D 📽 🖬 🎒 😹 h R	my_first_fpga	•	
	a la serie d'a d'an d'an d'an	0	
11992 S S S S S S	o 🕐 😳 🚼 🕹 🕸 💆		
Project Navigator	📸 my_first_fpga.bdf		
Entity			
Cyclone IV E: EP4CE11			
my_first_fpga	► A Đ		
Ĩ <u>≜</u> ₿₽	7 + 5		
Tasks ×	€ ■		
Flow: Compilation 👻	<u> </u>		
	M		11111111
Task 🖌 🔥			
🗆 🕨 Compile Design 🥏			
Analysis & Syn			::::::
Edit Settin			
View Report			~
8	<		>
× Type Message			Ĩ
Type neorage			
System (Processing) Extra Info	λ Info λ Warning λ Critical Warning	g λ Error λ Suppressed λ Flag /	
System / Processing / Extra Info Message:	Location	<u>्र</u>	Locate
Elinear	Threedon		
For Help, press F1		�_•≥⇒₩	Idle //

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)





另存为					
保存在(工):	my_first_f	pga	• +	E 👉 📰 •	
3000 我最近的文档					
回夏					
" 一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一					
我的电脑					
S					
网上邻居	-				17
	文件名 (M):	simple_counter.v		•	保存(5)
	保存类型(工):	Verilog HDL File	(*. v; *. vlg; *. ver:	ilog 🔻	取消
		Add file to current g	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).

//It has a single clock input and a 32-bit output port

module simple_counter (

CLOCK_50,

counter_out

);

input CLOCK_50;



output [31:0] counter_out;

reg [31:0] counter_out;

always @ (posedge CLOCK_50)

// on positive clock edge

begin

counter_out <= #1 counter_out + 1;// increment counter</pre>

end

endmodule

// end of module counter

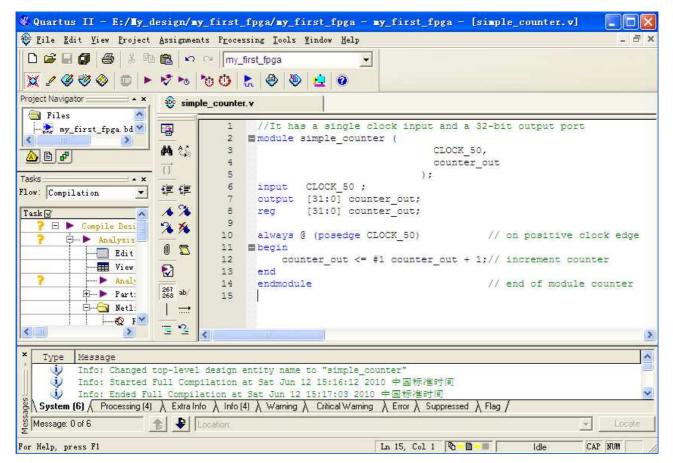


Figure 3-4 The Verilog File of simple_counter.v

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.





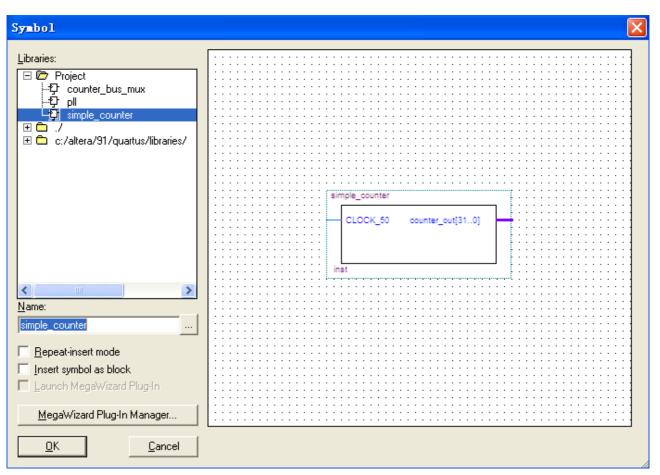
The Quartus II software creates a Symbol File and displays a message (see Figure 3-5).

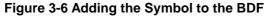


Figure 3-5 Create Symbol File was Successful

- 11. Click OK.
- 12. To add the simple_counter.v symbol to the top-level design, click the my_first_fpga.bdf tab.
- 13. Choose Edit > Insert Symbol.
- 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









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**.

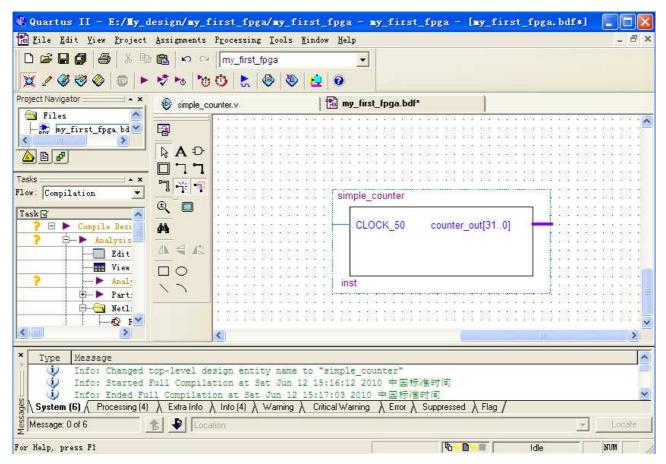


Figure 3-7 Placing the simple_counter symbol

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



16



simple counter. A PLL uses the on-board oscillator (DE2-115 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 ALTPLL.

1. Choose Edit > Insert Symbol or click Add Symbol on the toolbar---

Click Megawizard Plug-in Manager. The MegaWizard® Plug-In Manager appears (see Figure 3-8).

legaVizar	d Plug-In ∎anager [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? © <u>Create a new custom megafunction variation</u> © <u>E</u> dit an existing custom megafunction variation © Copy an existing custom megafunction variation Copyright (C) 1991-2010 Altera Corporation	
	Cancel < Back Next > Einis	h

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 I/O > ALTPLL.

b. Under Which device family will you be using? Choose the Cyclone IV E for DE2-115 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 at the end of the already created directory name.

e. Click Next.



MegaVizard Plug-In Manager [pag	e 2a]
Which megafunction would you like to customize? Select a megafunction from the list below	Which device family will you be Cyclone IV E
ALT2GXB ALT2GXB ALT2GXB_ ALT2GXB_RECONFIG ALTASMI_PARALLEL ALTCLKCTRL ALTDDIO_BIDIR ALTDDIO_BIDIR ALTDDIO_OUT ALTDDIO_OUT ALTDLL ALTDQ ALTDQ ALTDQS	Which type of output file do you want to create? ○ AHDL ○ YHDL ● Verilog HDL What name do you want for the output file? Browse E:\My_design\my_first_fpga\pll.v
ALTGX ALTGX_RECONFIG ALTGXB ALTIOBUF ALTIOBUF ALTLVDS ALTMEMPHY ALTOCT ALTPLL_RECONFIG ALTPLL_RECONFIG ALTREMOTE_UPDATE ALTREMOTE_UPDATE ALTTEMP_SENSE MAX II oscillator	Note: To compile a project successfully in the Quartus II software, your design files must be in the project directory, in the global user libraries specified in the Options dialog box (Tools menu), or a user library specified in the User Libraries page of the Settings dialog box (Assignments menu). Your current user library directories are: Project User Libraries: .\ triple_speed_ethernet-library\ megafunctions\ s4gx230_fpga_bup_maxii_interface\
	Cancel < <u>B</u> ack <u>N</u> ext > Einish

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

5. In the MegaWizard Plug-In Manager [page 3 of 14] window, make the following selections (see **Figure 3-10**).

a. Confirm that the Currently selected device family option shows the device family that corresponds to the development board you are using.

- b. The device speed grade choose 8 for DE2-115.
- c. Set the frequency of the inclock0 input 50 MHz.
- d. Click Next.



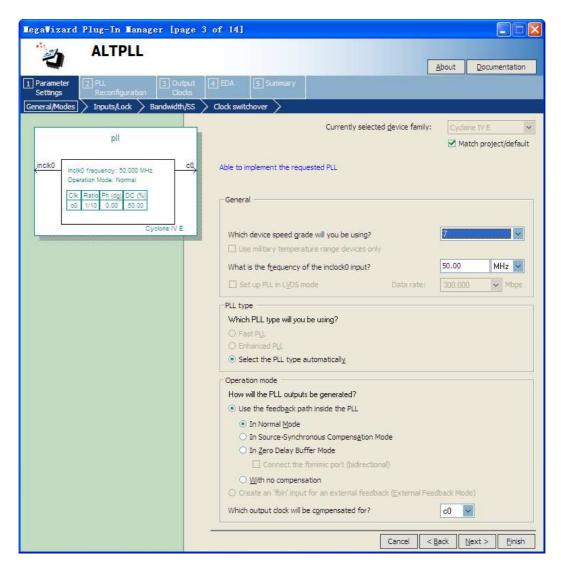


Figure 3-10 MegaWizard Plug-In Manager [page 3 of 14] Selections

6. Turn off all options on MegaWizard page 4. As you turn them off, pins disappear from the PLL block's graphical preview. See **Figure 3-11** for an example.

ter as IC

∎egaVizard Plug-In ∎anager [page	4 of 14]	
ALTPLL		
		About Documentation
Parameter 2 PLL 3 Output Settings Reconfiguration Clocks	4 EDA 5 Summary	
General/Modes Inputs/Lock Bandwidth/SS	Clock switchover	
	Able to implement the requested PLL	
pl incik0 peration Mode: Normal Cik Ratio Ph (dg) DC (%) 0 1/10 0.00 50.00 Cyclone IV E	Optional inputs Greate an 'pilena' input to selectively enable the P Create an 'areget' input to selectively enable the P Create an 'pidena' input to selectively enable the P Create an 'pidena' input to selectively enable the P Create an 'pidena' input to selectively enable the P Create an 'pidena' input to selectively enable the P Lock output Create 'locked' output Enable self-reset on loss of lock Advanced PLL parameters Using these parameters is recommended for advance Create output file(s) using the 'Advanced' PLL para- Configurations with output clock(s) that use case	the PLL phase/freq. detector

Figure 3-11 MegaWizard Plug-In Manager [page 4 of 14] Selections

7. Click Next three times.

8. At the top of the wizard, click the tab 3. Output Clocks to jump to the Output Clocks > clk c0 page

Clock Division Settings input 10 (**Figure 3-12**).

∎egaVizard Plug-In ∎anager [page 8	of 14]	
ALTPLL		
		<u>About</u> <u>D</u> ocumentation
Parameter PLL Output Settings Reconfiguration Clocks	4 EDA 5 Summary	
$\frac{1}{2}$ dk c1 \rightarrow dk c2 \rightarrow dk c3 \rightarrow dk c4 \rightarrow		
	c0 - Core/External Output Clock	
pll	Able to implement the requested PLL	
incik0 c0		
inciko frequency: 50.000 MHz	✓ Use this clock ✓ Clock Tap Settings	
Cik Ratio Ph (dg) DC (%)		Requested settings Actual settings
e0 1/10 0.00 50.00	O Enter output clock frequency:	100.00000000 MHz 5.000000
Cyclone IV E	Enter output clock parameters:	
	Clock <u>multiplication</u> factor	
	Clock <u>d</u> ivision factor	10 << Copy 10
	Clock phase shift	0.00 🚔 deg 🗸 0.00
	Phase shift step resolution(ps)	
	Clock duty cycle (%)	50.00
	More Details >>	
		Per Clock Feasibility Indicators
		c0 c1 c2 c3 c4
		Cancel < <u>B</u> ack <u>N</u> ext > <u>F</u> inish

Figure 3-12 MegaWizard Plug-In Manager [page 8 of 14] Selections

- 9. Click Finish.
- 10. The wizard displays a summary of the files it creates (see Figure 3-13). Click Finish again.

MegaVizard Plug-In Manager [page 14	of 14]	Summary
ALTPLL		
		About Documentation
1 Planameter 2 PLL 3 Output 4 Settings Reconfiguration Clocks 1 <th>EDA 5</th> <th>Summary</th>	EDA 5	Summary
Settings Reconfiguration Clocks	automatically g Finish to gener subsequent Me	es you wish to generate. A gray checkmark indicates a file that is lenerated, and a red checkmark indicates an optional file. Click ate the selected files. The state of each checkbox is maintained in gaWizard Plug-In Manager sessions. rd Plug-In Manager creates the selected files in the following my_first_fpga\ Description Variation file PinPlanner ports PPF file AHDL Include file VHDL component declaration file Quartus II symbol file Instantiation template file Verilog HDL black-box file
		Cancel < <u>B</u> ack <u>Next</u> > <u>Finish</u>
	2 42 \\/:=or	

Figure 3-13 Wizard-Created Files

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

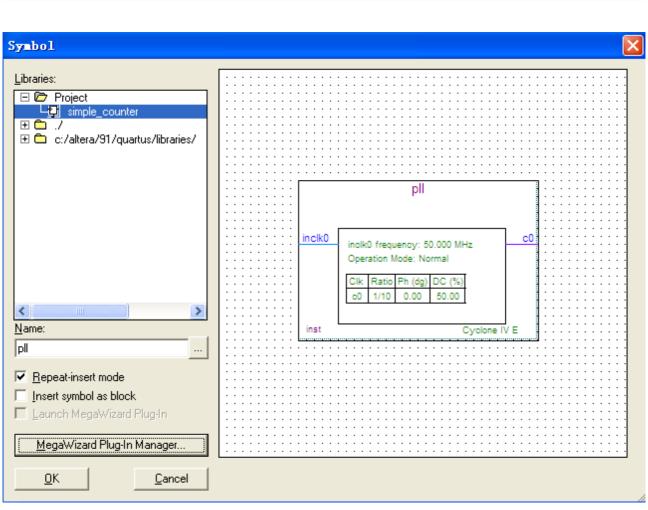


Figure 3-14 PLL Symbol

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**.

terasic

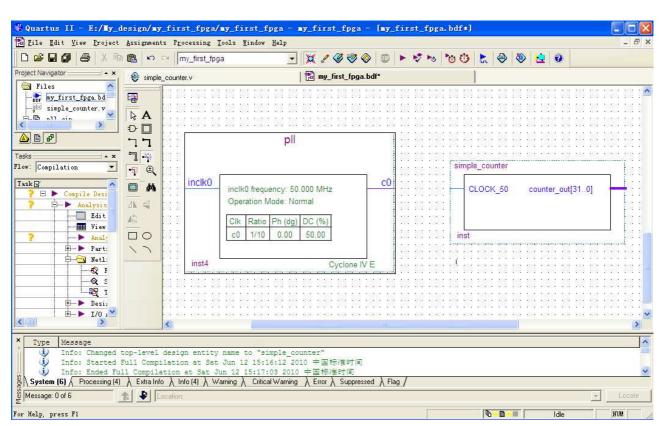


Figure 3-15 Place the PLL Symbol

12. Move the mouse so that the cursor (also called the selection tool) is over the pll symbol's c0 output pin. The orthogonal node tool (cross-hair) icon appears.

13. Click and drag a bus line from the c0 output to the simple_counter clock input. This action ties the pll output to the simple_counter input (see **Figure 3-16**).

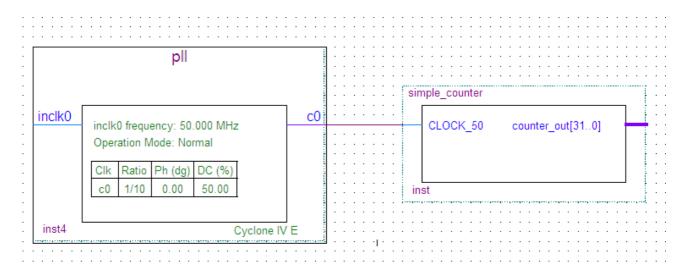


Figure 3-16 Draw a Bus Line connect pll c0 port to simple_counter CLOCK_50 port

14. Add an input pin and an output bus with the following steps:



terasiC

- 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.

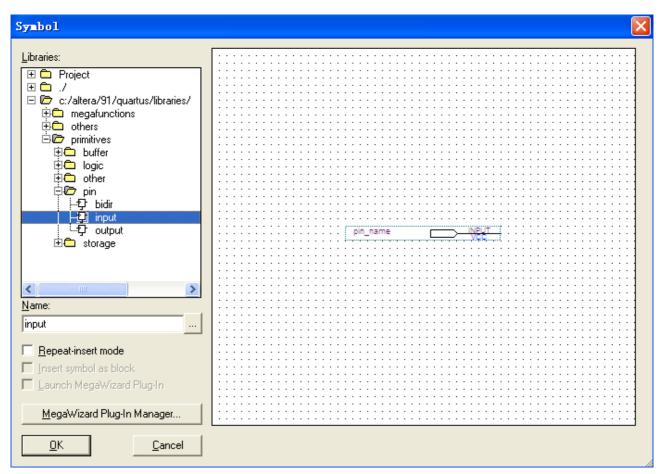


Figure 3-17 Input pin symbol

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.

	pll		
pin_name	inclk0 inclk0 frequency: 50.000 MHz Operation Mode: Normal Clk Ratio Ph (dg) DC (%) c0 1/10 0.00 50.00	CLOCK_50 counter_	_out[310]
	inst4 Cyc	one IV E	



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.

Pin Properties 🛛 🔀
General Format
To create multiple pins, enter a name in AHDL bus notation (for example, "name[30]"), or enter a comma-separated list of names.
Pin name(s): CLOCK_50
Default value: VCC
确定

Figure 3-19 Change the input port name

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

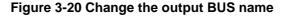
i. Using the Orthogonal Bus tool, draw a bus line connected on one side to the simple_counter output port, and leave the other end unconnected at about 6 to 8 grid spaces to the right of the simple_counter.

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 Propertie	es	$\mathbf{\Sigma}$
General Font	Format	
<u>N</u> ame:	counter[310] n block design file	ſ
	确定 取消	



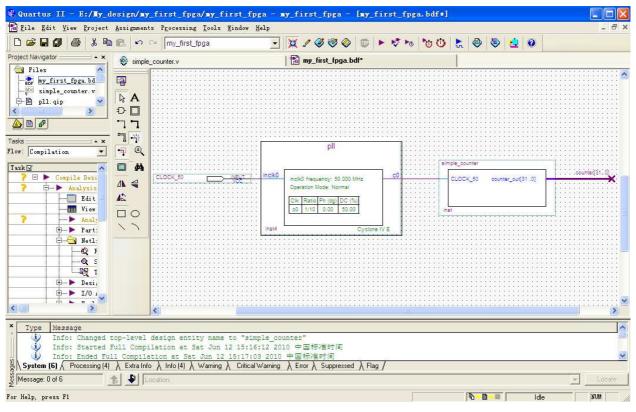


Figure 3-21 BDF

3.2 Add a Multiplexer

This design uses a multiplexer to route the simple_counter output to the LED pins on the DE2-115 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 DE2-115 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.

MegaVizard Plug-In Manager [page	2a]
LegaVizard Plug-In Lanager [page Which megafunction would you like to customize? Select a megafunction from the list below Installed Plug-Ins Altera SOPC Builder Altera SOPC Builder Communications Co	Which device family will you be using? Cyclone IV E Which type of output file do you want to create? AHDL AHDL YHDL Verilog HDL Browse What name do you want for the gutput file? Browse E:\My_design\my_first_fpga\counter_bus_mux.v 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 the global user libraries specified in the User Libraries page of the Settings dialog box (Assignments menu). Your current user library directories are: Project User Libraries: \L Tiple_speed_ethernet-library\megafunctions\
	s4gx230_fpga_bup_maxii_interface\ Cancel < <u>B</u> ack <u>N</u> ext > <u>Finish</u>

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).

∎egaVizard Plug-	in Tanager -	LPM_MUX [page 3 of {	5]
	M_MUX		About Documentation
1 Parameter 2 EDA Settings	3 Summary		
counter_bus_mux		urrently selected <u>d</u> evice family:	Cyclone IV E 🗸
data1x[30] data0x[30]	<u>301</u>	v many 'data' inputs do you want?	Match project/default
	Нои	wide should the 'data' input and result' output buses be?	4 v bits
	• N	You want to pipeline the multiplexa No Yes, I want an output latency of	er?
	[Create an asynchronous Clear Create a Clock Enable input	
Resource Usage 1 lpm_mux		Cancel < B	ack <u>N</u> ext > <u>F</u> inish

Figure 3-23 lpm_mux settings

9. Click Next.

ter <mark>asIC</mark>

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



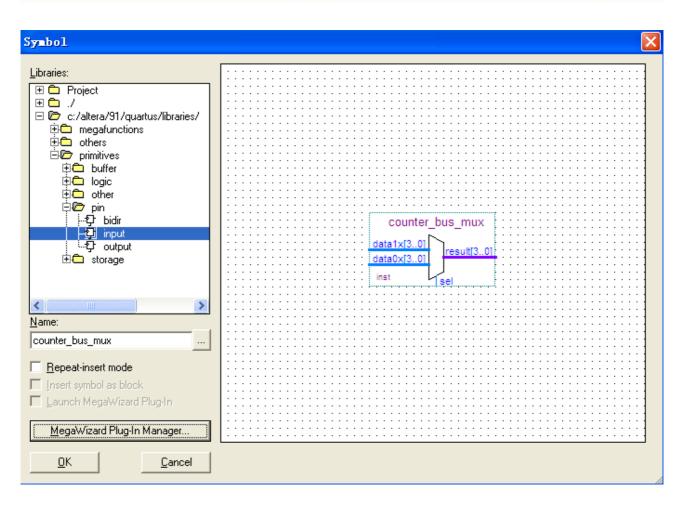


Figure 3-24 Ipm_mux Symbol

11. Click OK

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

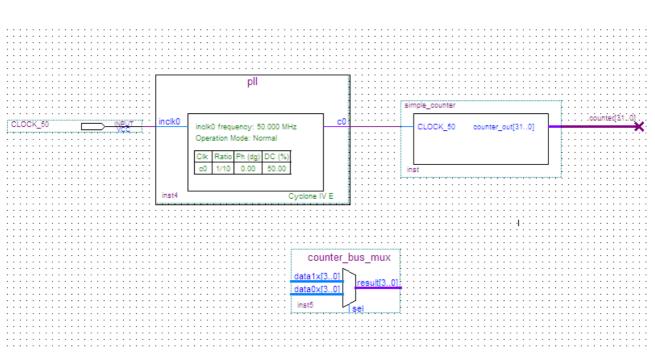


Figure 3-25 Place the lpm_mux symbol

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.

ter as IC

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.

32

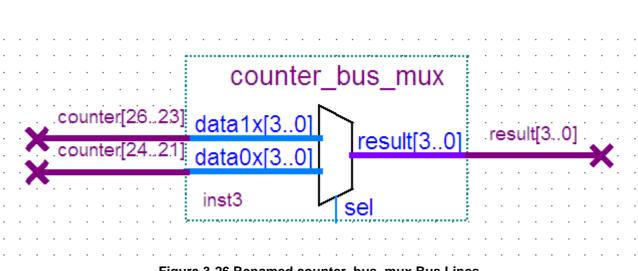


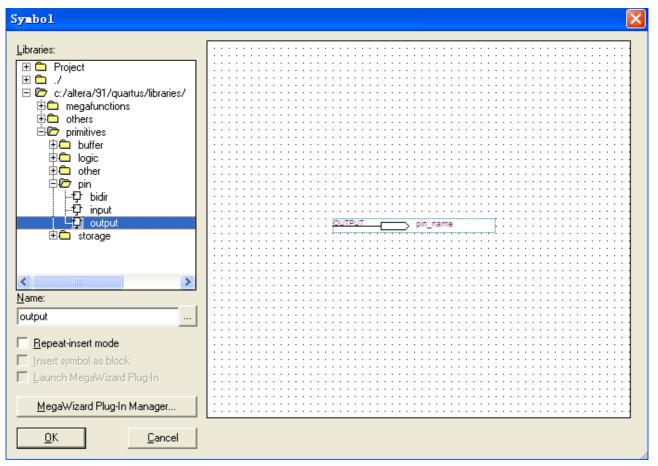
Figure 3-26 Renamed counter_bus_mux Bus Lines

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

14. Choose Edit > Insert Symbol.

ter as IC

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





16. Click OK.



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 LEDR [3..0] as described in steps 13 c and d. (see Figure 3-28).

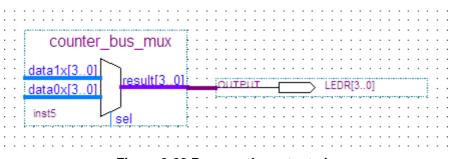


Figure 3-28 Rename the output pin

- 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] (see Figure 3-29).

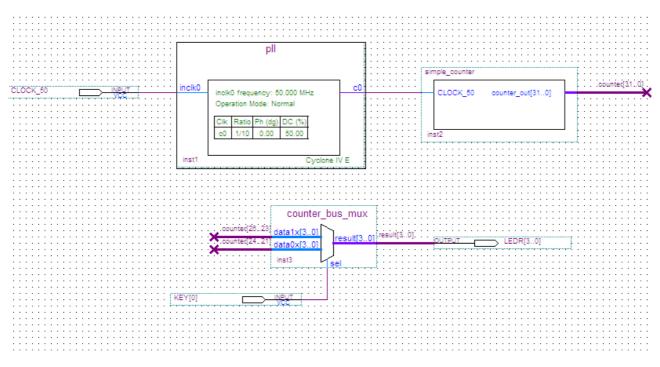


Figure 3-29 Adding the KEY [0] Input Pin 34





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 [0] and CLOCK_50 input pins and LEDR[3..0] output pin, perform the following steps:

 Choose Assignments > Pins, which opens the Pin Planner, a spreadsheet-like table of specific pin assignments. The Pin Planner shows the design's six pins. See Figure 3-30

Name	d: 📔 💽 💌	«» Edit: 🗙 🧹				Filter:	Pins: all
	Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard	Reserved
1	CLOCK_50	Input				2.5 V (default)	
2	KEY[0]	Input				2.5 V (default)	
3	LEDR[3]	Output				2.5 V (default)	
4	LEDR[2]	Output				2.5 V (default)	
5	LEDR[1]	Output				2.5 V (default)	
6	IEDR[0]	Output				2.5 V (default)	
7	< <new node=""></new>	>>					

Figure 3-30 Pin Planner Example

2. In the Location column next to each of the six node names, add the coordinates (pin numbers) as shown in **Table 3-1** for the actual values to use with your DE2-115 board.

	_
Pin Name	FPGA Pin Location
KEY[0]	M23
LEDR[3]	F21
LEDR[2]	E19
LEDR [1]	F19
LEDR [0]	G19
	35
terasiC	www.terasic.com

Table 3-1 Pin Information Setting

CLOCK_50

Y2

Double-click in the Location column for any of the six 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 3-31**.

Named:		▼ «≫ E	dit: 🗙 🗸 PIN_G19			Filte	er: Pins: all
		Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard
1		CLOCK_50	Input	PIN_Y2	2	B2_N0	2.5 V (default)
2		KEY[0]	Input	PIN_M23	6	B6_N2	2.5 V (default)
3		LEDR[3]	Output	PIN_F21	7	B7_N0	2.5 V (default)
4	0	LEDR[2]	Output	PIN_E19	7	B7_N0	2.5 V (default)
5		LEDR[1]	Output	PIN_F19	7	B7_N0	2.5 V (default)
6	•	LEDR[0]	Output	PIN_G19	7	B7_N2	2.5 V (default)
7		< <new node="">></new>					

Figure 3-31 Completed Pin Planning Example

Now, you are finished creating your Quartus II design!

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 -period 20.000 -name CLOCK_50

derive_pll_clocks

ter asiC

W



derive_clock_uncertainty

4. Save this file as my_first_fpga.sdc (see Figure 3-32)

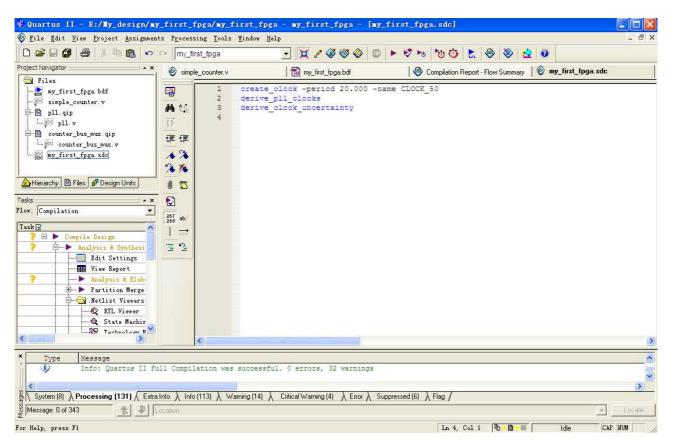


Figure 3-32 Default SDC

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.



erasic

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).

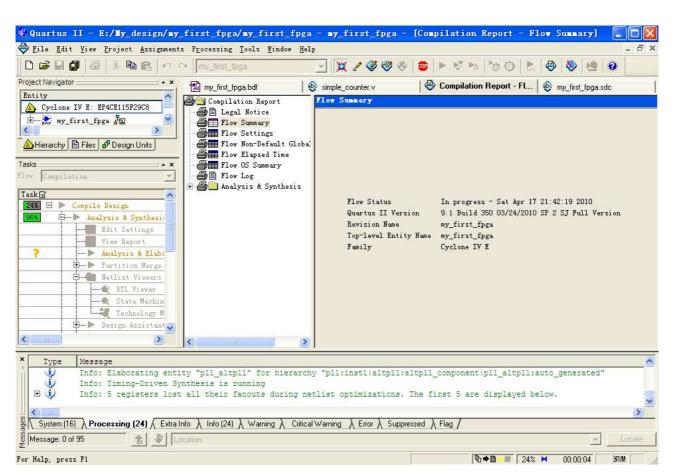


Figure 4-1 Compilation Message for project

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**.

terasic



Successful - Wed Jul 07 20:15:57 2010
9.1 Build 350 03/24/2010 SP 2 SJ Full Version
my_first_fpga
my_first_fpga
Cyclone IV E
EP4CE115F29C7
Preliminary
N/A
31 / 114,480 (< 1 %)
31 / 114,480 (< 1 %)
27 / 114,480 (< 1 %)
27
6 / 529 (1 %)
0
0 / 3,981,312 (0 %)
0 / 532 (0 %)
1 / 4 (25 %)

Figure 4-2 Compilation Report Example

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-Blaster 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 DE2-115 board, connect the USB-Blaster (included in your development kit) to J9 and the USB cable to the USB-Blaster. Connect the other end of the USB cable to the host computer.

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

c) Turn the DE2-115 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.



🔓 Hardware Seti	up USB-Blaster [US	B-0]			Mode:	JTAG		• P	rogress:	0	%	
Enable real-time	ISP to allow backgrou	ind prog	ramming (for MAX II de	vices)								
🖥 Start	File	5	Device	Checksum	Usercode	Program/ Configure	Verify	Blank- Check	Examine	Security Bit	Erase	Γ
և Stop	my_first_fpga.sof		EP4CE115F29	005666C9	FFFFFFF							
Auto Detect												
Delete												
Add File												
Change File												
Save File												
Add Device												
ն Մթ												
Down												

Figure 4-3 Programmer Window

2. Click Hardware Setup.

3. If it is not already turned on, turn on the USB-Blaster [USB-0] option under currently selected hardware. See Figure 4-4.

Hardware Setup				×
Hardware Settings JTAG Se Select a programming hardware hardware setup applies only to	e setup to use whe		levices. This pro	ogramming
Currently selected hardware: — Available hardware items: ——	USB-Blaster (L No Hardware USB-Blaster (L	-		•
Hardware	Server	Port	bbA	Hardware
USB-Blaster	Local	USB-0	Remo	ove Hardware
				Close

Figure 4-4 Hardware Setting

4. Click Close.



- 5. If the file name in the Programmer does not show my_first_fpga.sof, click Add File.
- 6. Select the my_first_fpga.sof file from the project directory (see Figure 4-5).

🖥 Quartus 🗄	II - E:/My_desi	gn/my_first_fpga/	ny_first_fp	ga - my_f ir	st_fpga ·	- [my_	first_t	fpga. cdf	1	
Zile <u>E</u> dit P <u>r</u>	ocessing <u>T</u> ools <u>W</u> in	adow								
🌲 Hardware Se	etup USB-Blaster [US	B-0]		Mode: JTA	G		 Progr 	ess:	100 %	
Enable real-tir	me ISP to allow backgrou	nd programming (for MAX II o	levices)							
🏴 Start	File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank- Check	Examine	Security Bit	Erase
🖬 Stop	my_first_fpga.sof	EP4CE115F29	005786DF	FFFFFFF						
Auto Detect										
🗙 Delete										
拳 Add File										
🗳 Change File.										
Save File										
🈕 Add Device.										
W Up										
Down										
	<									
or Help, press	s F1								ทเ	М

Figure 4-5 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 low, therefore, when counting begins all LEDs are turned on (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]).

3. If other LEDs emit faintness light, Choose Assignments > Device. Click Device and Options. See Figure 4-6.

Libraries 📉	Device								
Device ∃ Operating Settings and Conditions ↓ Voltage	Select the family and devi	ce you want to	target for c	ompilation.					
- Temperature	- Device family				Show in 'Av-	ailable dev	ices' list —		
Compilation Process Settings	Eamily: Cyclone IV E	Family: Cuclone IV E				Any	-		
Early Timing Estimate	Zaunde Laboration E				Pac <u>k</u> age: Pin count:				
	Devices: All	Devices: All				Any			
 Physical Synthesis Optimization EDA Tool Settings 						e: Any			
Design Entry/Synthesis	Target device				Show advanced devices				
- Simulation	C Auto device selecte	d hu tha Eittar			- 042 (17 The second		Second Contract		
Timing Analysis			4 4 4 W	4 Y	HardCo	by compati	ble only		
Formal Verification	<u>Specific device sele</u>	cted in 'Availat	ble devices'	list –					
Physical Synthesis	C Other n/a				Device and Pin Options				
Board-Level									
🗉 Analysis & Synthesis Settings	A <u>v</u> ailable devices:								
	Name	Core v	LEs	User I/	Memor	Embed	PLL 🔥		
Verilog HDL Input	EP4CE115F23C7	1.2V	114480	281	3981312	532	4		
Default Parameters	EP4CE115F23C8	1.2V	114480	281	27-57-225-375	532	4		
Fitter Settings	EP4CE115F23C8L	1.0V	114480	281		532	4		
Timing Analysis Settings	EP4CE115F23C9L	1.0V	114480	281	45.50.50.50.50.50	532	4		
 TimeQuest Timing Analyzer 	EP4CE115F23I7 EP4CE115F23I8L	1.2V 1.0V	114480 114480	281 281		532 532	4		
E Classic Timing Analyzer Setting	EP4CE115F29C7	1.0V	114480	529		532	4		
Classic Timing Analyzer Re	EP4CE115F29C8	1.2V	114480	529	CONTRACTOR OF A DESCRIPTION OF A DESCRIP	532	4		
Assembler	EP4CE115F29C8L	1.0V	114480	529		532	4 🗔		
- Design Assistant	EDACE11REDOCOL	1.01/	11//00	E00	0001010	E00	1		
SignalTap II Logic Analyzer	120								
- Logic Analyzer Interface	- Migration compatibility-	1.6	Companion	device			1		
Simulator Settings Simulation Verification	Marker De La	1	L I world Property	-			-		
Simulation Venncation	Migration Devices		HardCopy:	1					
PowerPlay Power Analyzer Setting	0 migration devices sele	ected	🔽 Limit DS	P & RAM to	HardCopy de	vice resol	irces		
SSN Analyzer		ospenderes (WI S WINS OF	Southern With State	ici unicen e den lheo	miniscentreessy	increases and a		

Figure 4-6 Device and Options

Choose unused pins. Reserve all unused pins: Choose the As input tri-stated option. See Figure 4-7.



Device and Pin Options 🛛 🔀
Dual-Purpose Pins Voltage Pin Placement Error Detection CRC Capacitive Loading Board Trace Model I/O Timing General Configuration Programming Files Unused Pins
Specify device-wide options for reserving all unused pins on the device. To reserve individual dual-purpose configuration pins, go to the Dual-Purpose Pins tab. To reserve other pins individually, use the Assignment Editor.
Reserve all unused pins: As input tri-stated
Description
Description: Reserves all unused pins on the target device in one of 5 states: as inputs that are tri-stated, as outputs that drive ground, as outputs that drive an unspecified signal, as input tri-stated with bus-hold, or as input tri-stated with weak pull-up.
<u>R</u> eset
确定

Figure 4-7 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

DE2-115 System Builder

5.1 Headquarter & Branches

- Tel : +886-3-5508800
- Fax : +886-3-5502860
- Add : No.356, Sec. 1, Fusing E. Rd., JhuBei City, HsinChu County Taiwan 302-64
- Email : sales@terasic.com / support@terasic.com