

My First FPGA - Quartus II 15.0(64-bit)



Copyright © 2003-2015 Terasic Inc. All Rights Reserved.



Chapter 1	Introduction	1
1.1 Design F	low	
1.2 Before Ye	ou Begin	
1.3 What You	ı Will Learn	
Chapter 2	Assign The Device	7
2.1 Assign Tl	ne Device	7
Chapter 3	Design Entry	12
3.1 Add a PL	L Megafunction	
3.2 Add a Mu	ıltiplexer	
3.3 Assign th	e Pins	
3.4 Create a l	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 Th	e Hardware	
Chapter 5	MAX10 NEEK Technical Support	44
5.1 Headquar	ter & Branches	



ii

Chapter 1



This tutorial provides comprehensive information that will help you understand how to create a FPGA design and run it on you MAX10 NEEK 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 begins 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.



MAX10 Neek My First FPGA Manual

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 15.0 (win7 64bits) 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 MAX10 NEEK 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 II driver, Plug in the 5-volt adapter to provide power to the board. Use the mini USB cable to connect the mini USB connector J8 (type B) on the MAX10 NEEK board to a USB port on a computer that runs the Quartus II software. Turn on the power switch on the MAX10 NEEK 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 MAX10 NEEK board is programmed by using Altera USB-Blaster II mechanism. If the USB-Blaster II driver is not already installed, the Driver Software Installation window in

Figure 1-2 will appear.





Figure 1-2 Driver Installation window

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 1-3.



Figure 1-3 The USB Blaster II in device manager

Right click Other devices ->Unknown device and select Update Driver Software then selecte Browse my computer for drive software and Click Next to get to Figure 1-4.



O I Update Driver Software - USB-Blaster	× ×
Browse for driver software on your computer	
Search for driver software in this location:	
C:\altera\15.0\quartus\drivers\usb-blaster	
 Include subfolders 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. 	
Next	Cancel

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\15.0\quartus\drivers\ usb-blaster-ii. 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 Install Anyway.



Browse For F	older	×
Select the	folder that contains drivers for your hard	lware.
	⊳ 퉬 bin64	*
	🖻 퉲 common	
	🖻 퉲 cusp	
	4 鷆 drivers	
	퉬 i386	
	Image: Sentinel Image: Sentinel Image: Sentinel Image: Sentinel Image: Sentinel Image: Sentine Image: Sentin	
	🖻 퉬 usb-blaster	
	🖻 퉲 usb-blaster-ii	
	⊳ 퉲 wdrvr	-
•		•
Folder:	usb-blaster-ii	
	ОК Са	incel

Figure 1-5 Browse to find the location

Windows Security
Would you like to install this device software? Name: Altera USB-Blaster Device Driver Package Publisher: Delaware Altera Corporation
Always trust software from "Delaware Altera Corporation".
You should only install driver software from publishers you trust. <u>How can I decide which</u> <u>device software is safe to install?</u>

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 MAX10 NEEK 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 a visual feedback that the design works. Of course, you can use your MAX10 NEEK 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 MAX10 NEEK 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 MAX10 NEEK 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 Wizard
Introduction
The New Project Wizard helps you create a new project and preliminary project settings, including the following:
 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 show me this introduction again
< Back Next > Finish Cancel Help

Figure 2-1 New Project Wizard introduction

- 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**.



💱 New Project Wizard
Directory, Name, Top-Level Entity
What is the working directory for this project?
E:/My_design/my_first_fpga
What is the name of this project?
my_first_fpga
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.



	on of the Quartus II soft	ware in wh			n the Tools n rice is support		<u>e Support List</u> webpa	age
Device family				Show i	n 'Available d	evices' list		
Family: MAX 10 (DA/DF/DC/SA/SF/SC)				Package: Pin count:		Any		
0	ted by the Fitter lected in 'Available devi	ces' list		Name		Any devices		
Other: n/a								
· ·	Core Voltage	LEs	Tota	l I/0s	GPIOs	Memory Bits	Embedded m	•
Available devices: Name	Core Voltage	LEs 49760	Tota 178	l I/Os	GPIOs 178	Memory Bits 1677312	Embedded m	-
Name 10M50DAF256C8GES				l I/Os		-		-
Name Name 10M50DAF256C8GES 10M50DAF256I7G	1.2V	49760	178	l I/Os	178	1677312	288	•
Name Name 10M50DAF256C8GES 10M50DAF25617G 10M50DAF484C6GES	1.2V 1.2V	49760 49760	178 178	l I/Os	178 178	1677312 1677312	288 288	•
Available devices:	1.2V 1.2V 1.2V	49760 49760 49760	178 178 360	l I/Os	178 178 360	1677312 1677312 1677312	288 288 288 288	

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**.





Figure 2-4 my_first_fpga project



Chapter 3



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).

🔇 Quartus II 64-Bit - E:/My_design/r	ny_first_fpga/my_first_fpga - my_first_fpga	×			
File Edit View Project Assignments Processing Tools Window Help Image: March altera.com Search altera.com					
🗋 😂 🖬 🥔 👗 🛍 🕲 🕫	🛛 🕅 my_first_fpga 💦 😵 🖌 🍑 💱 🕨 🕨 🤌 🕸	🕴 👸 👯 🤻 🔹			
Project Navigator Project Navigator	📲 my_first_fpga.bdf 🛛				
۹. X	🖷 🕞 🔍 A Ð 😪 🗕 🗆 🦳 🥄 🔪 🖉 🗛	> □ □ □ ×			
Entity					
MAX 10: 10M50DAF484C6GES					
▶ my_first_fpga					
		_			
▲ Hierarchy 🖹 Files 4		=			
Tasks ##×					
Flow: Compilation Customize					
▲ ► Compile Design					
Analysis & Synthesis		-			
4	III	4			
	< <filter>> A Find A Find Next</filter>				
Type ID Message					
	vice 10M08DAF484C7G is automatically selected f	or the device fa			
System (1) / Processing /	III	•			
System (1) / Processing /]			
	9	, 228 0% 00:00:00 <u>.</u>			

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:	🗼 my_first_fpga	-	▼ 📰 🌥 🛋 🕈	
Ca	Name	•		Di
Recent Places	📕 db			25
Desktop				
Libraries				
Computer				
Network				
	•	111		4
	File name:	simple_counter	•	Save
	Save as type:	Verilog HDL Files (*.v *.vlg *.verilog	g) 🔽	Cancel
		Add file to current project		

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.

- 12. To add the simple_counter.v symbol to the top-level design, click the my_first_fpga.bdf tab.
- 13. Double-click my_first_fpga.bdf Blank area
- 14. Double-click the Project directory to expand it.



15. Select the newly created simple_counter symbol by clicking its icon.

You can also double-click in a blank area of the BDF to open the Symbol dialog box (see **Figure 3-5**).



Figure 3-5 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 3-6**.



🔇 Quartus II 64-Bit - E:/My_design/r	ny_first_fpga/my_first_fpga - my_first_fpga	- • ×
File Edit View Project Assignments	Processing Tools Window Help 🗟	Search altera.com
🗋 😂 🖬 🗿 🕺 🛍 🖄 🔹	🛛 🕅 my_first_fpga 💦 😵 🔮 💱 🔶 🚥 🕨 🦻	🖄 🙆 👸 👯 🤹 🔹
Project Navigator Project Navigator	🛗 my_first_fpga.bdf* 🛛 🔶 simple_counter.v 🖂	
۹. ×	· · · · · · · · · · · · · · · · · · ·	ヽ ヽ !!! » ! »
 Files my_first_fpga.bdf simple_counter.v 	simple_counter	
▲ Hierarchy 🖹 Files	CLOCK_5@ounter_out[310]	=
Tasks # # ×	inst	
Flow: Compilation Customize		
Task	- m	-
X All O A 7 Type ID Message	< <filter>></filter>]
System / Processing /		► 300, 164 0% 00:00:00 .::

Figure 3-6 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 (MAX10 NEEK 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. Tools>IP Catalog>Library >Basic Functions> clocks;PLLsand Resets

2. Click PLL There will be a ALTPLL. The IP Catalog appears (see Figure 3-7)



IP Catalog	џ	₽×
0,	×	₹
4 🕌 Installed IP		
Project Directory		
No Selection Available		
▲ Library		
Basic Functions		
Arithmetic		
Bridges and Adaptors		
Clocks; PLLs and Resets		
📉 ALTCLKCTRL		
▲ PLL	_	
📉 ALTPLL		
X ALTPLL_RECONFIG		
🔆 Altera IOPLL		
🔨 Altera PLL		
🔨 Altera PLL Reconfig		
Configuration and Programming		
▷ I/O		
Miscellaneous		
On Chip Memory		
Simulation; Debug and Verification	n	
▷ Bitec		
DSP		
Interface Protocols		
▷ Low Power		
Memory Interfaces and Controllers		
Processors and Peripherals		
University Program		
Search for Partner IP		

Figure 3-7 IP Catalog

- 3 Double click on the ALTPLL
- 4. In Sve IP Variation, specify the following selections (see Figure 3-8)
 - a. Choose ALTPLL.
 - b. Under Which type of output file do you want to create? Choose Verilog

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





Figure 3-8 Save IP Variation Selections

d. Click OK.

5. In the MegaWizard Plug-In Manager [page 1 of 12] window, make the following selections (see Figure 3-9).

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

- b. The device speed grade is choosed 7 for MAX10 NEEK.
- c. Set the frequency of the inclock0 input 50 MHz.
- d. Click Next.





Figure 3-9 MegaWizard Plug-In Manager [page 1 of 12] 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 3-10** for an example.





Figure 3-10 MegaWizard Plug-In Manager [page2 of 12] 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 3-11).



NegaWizard Plug-In Manager [page 6 of	12]	? 💌
altpll		About Documentation
ParameterPLLOutputSettingsReconfigurationClocks	EDA EDSummary	
$\fbox{clk c0}$ $>$ $ m clk c1$ $>$ $ m clk c2$ $>$ $ m clk c3$ $>$	> dk c4 >	
pll inclk0 inclk0 frequency: 50.000 MHz Operation Mode: Normal Cik RatioPh (dg) DC (%) c0 1/10 0.00 50.00 MAX*10	CO - Core/External Output Clock Able to Implement the requested PLL Use this clock Clock Tap Settings Requested Settings Enter output clock frequency 0.00000000 [MHz - Clock multiplication factor 1 Clock division factor 10 Clock duty cycle (%) 50.00 Note: The displayed internal 50.00 settings of the PLL is Primary clock VCO frequenc 6 * recommended for use by advanced users only Per Clock Feasibility Indicator c0 c1 c2 c3 c4	
	Cancel	< Back Next > Finish

Figure 3-11 MegaWizard Plug-In Manager [page 6 of 12] Selections

- 9. Click Finish.
- 10. The wizard displays a summary of the files it creates (see Figure 3-12). Click Finish again.





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

Double-click my_first_fpga.bdf blank area and open the Symbol window, showing the newly created PLL megafunction. See Figure 3-13.



Figure 3-13 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 3-14.



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 3-15).



Figure 3-15 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:
 - a. Double-click my_first_fpga.bdf blank area
 - b. Under Libraries, libraries > primitives > pin >input. See Figure 3-16



c. Click OK

If you need more space 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 3-16 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 3-17.



Figure 3-17 Connecting the PLL symbol and Input port

f. Change the pin name by double-clicking pin_name and typing CLOCK_50 (see **Figure 3-18**). This name correlates to the oscillator clock that is connected to the FPGA.



MAX10 Neek My First FPGA Manual

	Pin Properties		X
Γ	General Fo	rmat	
		iple 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 3-18 Change the input port name

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.

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

i. Type counter [31..0] as the bus name (see **Figure 3-19**). 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).

j. Click OK. Figure 3-20 shows the BDF.



Ľ	Bus Properties
	General Font Format
	Name: counter [310]
	Hide name in block design file.
	OK Cancel Help

Figure 3-19 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 MAX10 NEEK 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 MAX10 NEEK development board

- 1. Choose Library
- 2. Click Basic Functions.



3. Click Miscellaneous

4. Click LPM_MUX

5. Choose the device family that corresponds to the device on the development board you are using, choose Verilog as the output file type, and name the output file counter_bus_mux.v (see Figure 3-21)



Figure 3-21 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 3-22).



🔨 MegaWizard Plu	ug-In Manager [page 1 of 3]
👌 LPM_	MUX <u>About</u> <u>Documentation</u>
Parameter Settings	DA I Summary
counter_bus data1x[3] faj data0x[3] faj data0x[3] faj	Currently selected device family: MAX 10 - Match project/default
	How many 'data' inputs d 2 now wide should the 'data' input and the 4 trocult output busics bo2
	Do you want to pipeline the multiplexer?
	No
	○ Yes, I want an output latency of 1 ⊂ clock cycles
	Create an asynchronous Clear input Create a Clock Enable input
Resource Usage	
1 lpm_mux	Cancel < Back Next > Finish

Figure 3-22 Ipm_mux settings

- 9. Click Next.
- 10. Click Finish twice. The Symbol window appears (see Figure 3-23 for an example).





Figure 3-23 Ipm_mux Symbol

11. Click OK

12. Place the counter_bus_mux symbol below the existing symbols on the BDF. See Figure 3-24.



Figure 3-24 Place the Ipm_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 3-25 shows the renamed buses.

			•		•		•			• •		•			•					•			• •		•												•			•			•			•	•	
•	• •	•	•	• •	•		•	•		• •		•	•		•	• •	•	• •	•	•	•		• •		•	•				•	•		•	•		•	•		•	•	•	•	•	•	•	•	•	• •
•			•		•	• •	•	•		• •			•		•	• •						•	• •		٠	•											•		•	•	•	•	•	•		•	•	• •
•			•		٠.			•		• •					•	• •						•	• •		•	•											•			۰.	•		•	•		•	•	
•	• •		•		•		•	•							•	• •		• •		·		•	• •		•	•											•		•	•			•	•		•	•	
 					•														_	_	_	_	_	_	_	_	_	_	_	_	_									•			•					
 																.	~	o		-	4.	~		- 1			~			-		х.																
 																.	C	υ	u	п	L	e	I		J	u	s		г	п	u	2	ι.															
 																.	- 1	Ξ.				Ξ.	-	_				_				۰.	۰.															
 				C.C	M.	m	ė	r٢	21	ε.	- 0	2	1			.	d	at	а	1	νĪ	3	γ	A1			_	_		_	. :									0								
 			۰.	U.	10												u	u	u	1.	~]	9	t d	2	C	u b	ŧΓ	2		U.			re	20	11	ıtt		2		01	ι.							
				CO	M.	und	0	r I	[2]	Λ		2	11				d	at	a	n,	νĪ	2	64	ΥÐ	9	u	ч	<u> </u>		<u>u</u>	:	-	1	-	20	In		0		0			-	÷.				
		· · .		υu	лu		C	r	[4	4		2	r j			· I	u	αι	α	U.	~1										•												- N.	•				
		-7	÷																					P										•		•	·	•	•	·	•		÷			•		
		2	1		÷		÷	÷				÷	÷			•	i	าร	t2)				,								÷	-	;	÷	÷	;	ļ	ì	:	;	2	ŗ	÷		:	: .	
		2			:	: :	÷	÷			Ĵ	ł	÷	1		:	i	าร	t2	2			4	7								ļ	ļ		Ì	Ì	÷	i	i	:	÷		ľ	÷	i.			
 		2			÷		÷	Ì			ļ	÷	Ì	÷		:	İ	۱s	t2	2			4	2								ļ				Ì	:	:	:	:	-	-	ļ					
 		2			:		:	-				:	Ì	-		-	i	ns	t2	2			4	7		<u></u>	<u>.</u>										:		:	•		-	-					
								-					-	-			i	ns	t2	2	:		4	7		<u>.</u>															-						-	
 		2					•						-	-			İ	ns	t2	2			4	7																			-				-	
														-			i	ns	t2				4	;;																							-	
 																	i	ns	t2	2			4									* * * * * * * *																
 																	i	ns	t2	-			4												********	* * * * * * * * * *		* * * * * * * * * *	* * * * * * * * * *								-	

Figure 3-25 Renamed counter_bus_mux Bus Lines

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

- 14. . Double-click my_first_fpga.bdf Blank area
- 15. Under Libraries, double-click quartus/libraries/ > primitives > pin > output (see Figure 3-26).





Figure 3-26 choose an output pin

- 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 3-27).

	counter bus mux
counter[26, 22]	data 1v[2][0]
Counter[2020]	data 1/10 Meutto 13 01 result (3.01
Converter [04, 04]	
Counter [24.:21]	
* counter [24.:21]	Uataoxio. 201701 Leok [30]
~	inst2
	inst2
	inst2 🛱
*	inst2 🛱
	inst2 🛱
	inst2 🛱
*	inst2 🛱

Figure 3-27 Rename the output pin

- 19. Attach an input pin to the multiplexer select line using an input pin:
- a. Double-click the my_first_fpga.bdf Blank area.
- $b. \quad Under \ Libraries, \ double-click \ quartus/libraries/> primitives > pin > input.$
- 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 3-28).





Figure 3-28 Adding the KEY [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 [0] and CLOCK_50 input pins and LEDR[3..0] output pin, perform the following steps:

1. Choose Assignments > Pins 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 3-29





Figure 3-29 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 MAX10 NEEK board.

Pin Name	FPGA Pin Location
KEY[0]	T22
LEDR[3]	C3
LEDR[2]	A3
LEDR [1]	B3
LEDR [0]	C2
CLOCK_50	N5

Table 3-1 Pin Information Setting

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 3-30**.





Figure 3-30 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:



derive_pll_clocks

derive_clock_uncertainty

4. Save this file as my_first_fpga.sdc (see Figure 3-31)



Figure 3-31 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).



	My_design/my_first_fpga/my_first_fpga Assignments Processing Tools Window		Search alte	
	my_first_fpga	<u>_ X 2 4 4 8 9 0 0 1</u>		🔶 🧼 🔺 »
Project Navigator 🛛 🖓 🖉 ×	Compilation Report - my_first_fpga	🗵 🔡 my_first_fpga.bdf 🗵	🔷 my_first_fpga.sdc 🗵	IP Cat ₽₽×
Q X	Table of Contents	Flow Summary		Q X ⊒
Files Files my_first_fpga.bdf simple_counter.v pll.qip a counter_bus_mux.qir my_first_fpga.sdc rchy Files Files Fow: (Customize 6% 4 Compile De > Analysis 0% > Fitter (PI	Flow Summary Flow Settings Flow Non-Default Global Settings Flow Elapsed Time Flow CC Summary	Flow Status Quartus II 64-Bit Version Revision Name Top-level Entity Name Family Device Timing Models Total logic elements Total combinational functions Dedicated logic registers Total registers Total pins Total virtual pins Total virtual pins Total memory bits Embedded Multiplier 9-bit elements Total PLLs UFM blocks ADC blocks	In progress - Thu Jun 25 13:5 15.0.0 Build 145 04/22/2015 5 my_first_fpga MAX 10 10M50DAF484C6GES Preliminary 31 31 27 27 6 0 0 0 1 1 0 0	and arrocarro
0% > Assembl -				
		• []]	•	🕆 Add
↓ 14896 Fi	<pre></pre>		nd Next because it is not s	supported
System Processin	<u>g (61)</u> /		269	6 🐡 00:00:28 🖽

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 - Thu Jun 25 13:58:08 2015
Quartus II 64-Bit Version	15.0.0 Build 145 04/22/2015 SJ Full Version
Revision Name	my_first_fpga
Top-level Entity Name	my_first_fpga
Family	MAX 10
Device	10M50DAF484C6GES
Timing Models	Preliminary
Total logic elements	32 / 49,760 (< 1 %)
Total combinational functions	32 / 49,760 (< 1 %)
Dedicated logic registers	27 / 49,760 (< 1 %)
Total registers	27
Total pins	6 / 360 (2 %)
Total virtual pins	0
Total memory bits	0 / 1,677,312 (0 %)
Embedded Multiplier 9-bit elements	0 / 288 (0 %)
Total PLLs	1 / 4 (25 %)
UFM blocks	0/1(0%)
ADC blocks	0 / 2 (0 %)

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 II 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 MAX10 NEEK board, connect the USB-Blaster II (included in your development kit) to J9 and the USB cable to the USB-Blaster II. 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 MAX10 NEEK 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.



	- E:/My_design/m			ja - my_fir	st_fpga -	[Chain]	1.cdf]					
File Edit View	Processing Tools	Window Hel	p 🛡							Search	altera.con	ו 🚯
🔔 Hardware Se	tup NEEK10 [U	SB-1]		Mode:	JTAG			•	Progress	:		
Enable real-ti	ime ISP to allow ba	ckground progra	amming wh	en available	9							
Start	File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank- Check	Examine	Security Bit	Erase	isp Clamf	
🔎 Stop												
Auto Dete												
× Delete												
🌥 Add File												
Change File												
🖼 Save File												
Add Device												
t [™] Up												
J [™] Down												

Figure 4-3 Programmer Window

2. Click Hardware Setup.

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

Hardware Setup				×
Hardware Settings	JTAG Settings			
Select a programming l hardware setup applies			devices. Th	is programming
Currently selected hard Available hardware ite	No Har	0 [USB-1] dware 0 [USB-1]		·
Hardware	Server	Port		Add Hardware
NEEK10	Local	USB-1		Remove 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 dir ectory (see Figure 4-5).

🖐 Programme	r - E:/My_design/m	ny_first_fpga/m	ny_first_fpg	ga - my_fir	st_fpga -	[Chain]	1.cdf]*					
File Edit View	Processing Tools	Window Hel	p 💎							Searc	h altera.co	om 🔇
🔔 Hardware S	etup NEEK10 [U	SB-1]		Mode:	JTAG			•	Progress	:	100% <mark>(Su</mark>	ccessful)
Enable real-	time ISP to allow ba	ckground progra	amming wh	en available	9							
Start	File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank- Check	Examine	Security Bit	Erase	ISP CLAMF	
\mu Stop	output_files/my	10M50DAF4	00273104	00273104	✓							
Auto Dete												
X Delete												
b Add File												
Change File												
Save File												
Add Device	10M50DAF4	84ES										
1 [™] Up												
J [™] Down												
												H.

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.

	he Quartus II software in	which your	target device is su	pported, refer	to the <u>Device Suppor</u>	<u>t List</u> webp	page			
Device family			Show in 'Availa	ble devices' lis	t					
Family: MAX 10 (DA/DF/D	C/SA/SF/SC)	•	Package:	Any			•			
Devices: All		-	Pin count:	Any			•			
Target device			Core Speed grade: Any							
 Auto device selected by 	the Fitter		Name filter:							
, ,	in 'Available devices' list		Show adva	nced devices						
Other: n/a			Device and Pin (Options						
Available devices:										
Name	Core Voltage	LEs	Total I/Os	GPIOs	Memory Bits	Emb	e ^			
10M50DAF484C6GES	1.2V	49760	360	360	1677312	288	Ξ			
10M50DAF484C7G	1.2V	49760	360	360	1677312	288	-			
4	1					•				

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 - my_	first_fpga
Category:	
Category: General Configuration Programming Files Unused Pins Dual-Purpose Pins Capacitive Loading Board Trace Model I/O Timing Voltage Pin Placement Error Detection CRC CvP Settings Partial Reconfiguration	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: 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 bushold, or as input tri-stated with weak pull-up.
	OK Cancel Help

Figure 4-7 Setting unused pins

Click twice OK.

4. In the Processing menu, choose Start Compilation. After the compilation, 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

MAX10 NEEK Technical Support

5.1 Headquarter & Branches

- Tel : +886-3-575-0880
- Add : China/406, Jingfeng Bld. B, Intl Business Center, Special No. 1, Guang Gu Rd., Wuhan, China, 430074
- Email : sales@terasic.com.cn / support@terasic.com.cn

