iCEcube2 Tutorial

v1.0 - May 2010





Copyright Information

Copyright ©2008-20010 SiliconBlue Technologies Corporation. All rights reserved. SiliconBlue, iCEcube, iCEcube2, iCE65, mobileFPGA, are trademarks of SiliconBlue Technologies Corporation. Magma and Blast FPGA are trademarks Magma Design Automation, Inc. Synopsys and Synplify Pro are trademarks of Synopsys, Inc. All other trademarks are the property of their prospective owners. All specifications are subject to change without notice.

Notice of Disclaimer: This software is provided to you "as-is," without any express or implied warranty.

Contact Information

SiliconBlue Technologies Corporation

3255-7 Scott Blvd., Suite 101 Santa Clara, CA 95054 Tel: (408) 727-6101 Fax: (408) 727-6085

support@siliconbluetech.com

Revision History

The following table lists the revision history of this document.

Version	Revision
1.0	Release iCEcube2 2010.03





Preface

About this Document

The *iCEcube*2[™] *Tutorial* takes the user through the iCEcube2 development tools step-by-step using a sample design. After using the tutorial, the user will be able to set-up a project, compile a design, and run different analysis tools such as the static timing analyzer and power estimator tools.

For information on the Synopsys Synplify Pro software, please refer to the Synplify Pro documentation provided in the synpro/doc directory in the iCEcube2 software installation (<icecube2_install_dir>/synpro/doc), and on the SiliconBlue website.

For detailed information of the iCEcube2 development tools, please refer to the iCEcube2 User Guide.

Software Version

This tutorial documents the features of iCecube2 Software Version 2010.03

For more information about acquiring the iCecube2 software, please visit the SiliconBlue website: <u>http://www.siliconbluetech.com.</u>

Platform Requirements

The iCecube2 software can be installed on a platform satisfying the following minimum requirements.

A Pentium 4 computer (500 MHz) with 512 MB of RAM, running one of the following operating systems:

- Windows XP Professional
- Windows Vista
- Red Hat Enterprise Linux WS v4.0

Programming Hardware

There are three ways to program iCE FPGA devices:

- A third party programmer, using the programming files generated by the iCecube2 Physical Implementation Tools. Consult the third party programmer user manual for instructions.
- The iCEman Evaluation Board, which not only serves as a vehicle to evaluate iCE FPGAs, but also includes an integrated device programmer. This programmer can be used to program devices on the iCEman board, or it can be used to program devices in a target system. Please contact SiliconBlue Technologies for additional information on the iCEman Evaluation Board.
- Digilent USB cables



Overview

iCEcube2 Tool Suite

The figure below depicts the design flow using the iCEcube2 Tool Suite. The components in blue signify functionality supported by SiliconBlue Technologies' proprietary iCEcube2 software, and the components in purple indicate the functionality supported by Synopsys' Synplify Pro synthesis tools. The iCEcube2 software and Synopsys software together constitute the iCEcube2 Tool Suite.



Figure 0-1: The iCEcube2 Design Flow



Design Flow

The following steps provide an overview of the design flow using the iCEcube2 Tool Suite.

- 1. Create a new project in the iCEcube2 Project Navigator and specify a target device and its operating conditions. Add your HDL (Verilog or VHDL) design files and your Constraint files to the project.
- 2. Synthesize your design using the Synplify Pro design software. This software has been provided as part of the iCEcube2 Tool Suite, and can be invoked from the iCEcube2 Project Navigator. Within the Synopsys design environment, assign your Logic Synthesis, Timing and Pin constraints.
- 3. Perform Placement and Routing using the iCEcube2 place and route tools. iCEcube2 also supports physical implementation tools such as floor planning, allowing users to manually place logic cells and IOs.
- 4. Perform timing simulation of your design using an industry-standard HDL simulation tool. The files necessary for simulation are automatically generated by the iCEcube2 Physical Implementation tools, after the routing phase.
- 5. Perform Static Timing Analysis using the iCEcube2 static timing analyzer.
- 6. Generate the device programming and configuration files from the iCEcube2 Physical Implementation tools.
- 7. Program your device using the device programming hardware provided by Silicon Blue Technologies.

Tutorial

This chapter provides a brief introduction to the iCEcube2 design flow. The goal of this chapter is to familiarize the user with the fundamental steps needed to create a design project, synthesize and implement the design, generate the necessary device configuration files, and program the target device.

Detailed information on tool features and usage is provided in subsequent chapters.

Creating a Project

Starting the iCEcube2 software for the first time, you will see the following interface shown in Figure 2-1.





Figure 0-1 : Create a New Project

The first step is to create a new design project and add the appropriate design files to your project. You can create a new project by either selecting **File>New Project** from the iCEcube2 menu, or by clicking the **Create a New Project** icon as seen in Figure 2-1. The New Project Wizard GUI is displayed in Figure 2-2.



New Project	2 🛛
Project	
Project Name:	quick_start
Project Directory:	C:\SbtTools\examples\blinky\quick_start
Device	
Device Family:	iCE65
Device:	L04
Device Package:	CB284
Power Grade:	L
Range: Commerci Core Voltage(V) Voltage Tole	Best: Typical: Worst: ial 0 25 70 arance Range Best: Typical: Worst: asheet defaul 1,26 1.2 1.14
Perform timing a	O Best O Typical O Worst
Start From Synt	hesis
 Synplify 	
O Magma	
O Start From Back	End
	Next Cancel

Figure 0-2 : New Project Setup Form

- 1. **Project Name** Field: Specify a project name (*quick_start*) in the Project Name field.
- 2. **Project Directory** Field: Specify any directory where you want to place the project directory in the Project Directory field.

- 3. **Device** Fields: This section allows you to specify the SiliconBlue device you are targeting. For this example, change the Device to **L04** and change the device package to the **CB284**.
- 4. **Operating Condition** Fields: This section allows you to specify the operating conditions of the device which will be used for timing and power analysis.
- **5. Start From Synthesis:** This option allows you to specify the Synopsys Synplify synthesis tool or the Magma BlastFPGA synthesis tool for logic synthesis. For this example, select Synplify.
- 6. Click Next to go to the Add Files dialog box shown in Figure 2-3. You will be prompted to create a new project directory. Click Yes.
- 7. In the Add Files dialog box, navigate to: <iCEcube2 installation directory>/examples/blinky

Highlight the following files:

blinky.vhd

blinky_syn.sdc*

Select each file and click >> to add the selected file, or click >>> to add all the files in the open directory (files can be removed using << and <<<) to your project. Click **Finish** to create the project.

* The SDC file is a Synopsys constraint file, which contains timing constraint information.

ok in: C:/SbtTools/exampl			blinky.sdc blinky.vhd	
	blinky.sdc blinky.vhd quick_start	>>>		
Desktop		<<		
		<<<		
Home		Details		
es of type: All Files(*)		×		

Figure 0-3: New Project Wizard – Add Files dialog box

After successfully setting-up your project, you will return to the following iCEcube2 Project Navigator screen shown in Figure 2-4.





Figure 0-4: iCEcube2 Project Navigator View after Completing Project Set-up

Synthesizing the design

After a successful project setup, **Double-Click on the Synthesis Launch Synthesis Tool** icon in the project navigator window. See Figure 2-5. This will bring-up the Synopsys Synplify Pro synthesis tool's graphical user interface. See Figure 2-6.





Figure 2-5 : Launch Synthesis Tool



			arly Access 2 - [C:/SbtTools/examples/blinky/quick_star Run Analysis HDL-Analyst Options Window Tech-Support V		r_syn.pr]]						
D 🖸 👜 🗉		10					11 2 2	le.			
0 🗉 🚥		-	B ∈ A B								
Run Synplify Pro Ready											
D Open Project	_	-	quick_start_Implmnt - Silicon Blue iCE65 : iCE65L04 : CB284 : -L		Implementation Directory						
		_	🕞 📕 [quick_start_syn] - C:\SbtTools\examples\blinky\quick_start\	quick_start_syr	… r ⊂ C:\SbtTools\examples\blinky\guick_start\guick_start_Implmnt						
Close Project			VHDL Jinky.vhd [work]		i	Size	1	Modifi *			
Add File			🕀 🧭 Constraint		Name Hackup	loize	Type Directory	Modin			
B. Change Elle		-	blinkly_syn.sdc		E Coreip		Directory	17:45			
Change File	_	_	- 💭 quick_start_Implmnt		🗄 🙋 sbt		Directory	17:20			
🚑 Add Implement	ation				🗄 🙋 syntmp		Directory	17:45			
Int					🗄 🙋 xplace	The state	Directory	17:45			
Timplementation	Options				identify.log	152 bytes 119 kB	log File Edif Netlist	17:45 17:45			
Add P&R Impler	nontation				quick_start.edf	0 bytes	fse File	17:45			
Add Hork Implei	nencatio	•					rse File htm File	17:45			
View Log					guick_start.htm	348 bytes 2 kB	scf File	17:45			
A VIEW LUG					quick_start.scr Go quick_start.srd	2 KB 14 kB	Netlist	17:45			
Frequency(MHz):					guick_start.srl	6 kB	Netlist (RTL)	17:45			
1	\$				quick_start.srm	121 kB	Netlist (Gate)	17:45			
Auto Constrain					guick_start.srr	47 kB	srr File	17:45			
Muco Constrain					quick_start.srs	6 kB	Netlist (RTL)	17:45			
FSM Compiler	V	1.			guick_start.szr	17 kB	szr File	17:45			
					guick_start.tlg	869 bytes	tla File	17:45			
Resource Sharing	•				guick_start.og guick_start prepass.srd	7 kB	Netlist	17:45			
Pipelining	•	-	•		A datat_scare_propassional			4.			
Retiming	1				Contract of the second s						
quick_start_syn	.prj										
			Information 📳 🔀		Watch			e			
icense: sympl			node-locked	Log Parameter	mick	start Implmnt	-	1			
icense. sympi	rrypro	_sbc		Log . ardinocor	quick	and the second					
		_		-							
TCL Script Mess	ages			Log Watch							

Figure 2-6 : Synplify Pro Graphical User Interface

Hit the Run Button to synthesis your design. Once synthesis is complete, you will see a Done message. See Figure 2-7.



Symplify Pro I	-2009.1	25 Early Access 2 - [C:/SbtTools/examples/blinky/	/quick_sta	rt/quick_star	rt_syn.prj]			
File Edit	View Pr	oject Run Analysis HDL-Analyst Options Window Tec	ch-Support	Web Help				-8
B 🖸 🖲 🗉	1	1 🛃 😤 🗇 10 A A 🗛 🗛 🖗 🙆 😉	🥑 🚳 🗰		8 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0,0,0	日前二	4
@ I m	- 14	R R 8 8 8 8						
Run		Synplify Pro	-					
	6	Done: 0 errors, 4 warnings, 19 notes						
Dopen Project	1	quick_start_Implmnt - Silicon Blue iCE65 : iCE65LD4 :			Implementation Directory			
P	-	[auick_start_syn] - C:\SbtTools\examples\blink	VIQUICK_start	quick_start_syr	C:\SbtTools\examples\blinky\quick	start)quick sta	et Implont	
Close Project		blinky.vhd [work] ->WARNINGS: 4 ->N	OTES: 6					In real
Add File		Constraint			Name	Size	Туре	Modifi
	_	blinkly_syn.sdc			B B backup D coreip		Directory	12:42
Change File		Q quick_start_Implimit			E Stt		Directory	17:20
Add Implement	Nico				III 💋 syntmp		Directory	17:45
And turbements	300111				III 🕖 xplace		Directory	17:45
Implementation	Options				identify.log	152 bytes	log File	13:37
					quick_start.edf	119 kB	Edif Netlist	13:37
R Add P&R Imples	nentation				quick_start.fse	0 bytes	fse File	13:37
-					quick_start.htm	348 bytes	htm File	13:37
View Log					quick_start.scf	2 kB	scf File	13:37
Frequency(MHz):	-				- 😧 quick_start.srd	14 kB	Netlist	13:37
Frequency(mma):	141				quick_start.srl	6 kB	Netlist (RTL)	17:45
- 1	-				- 🕞 quick_start.srm	122 kB	Netlist (Gate)	13:37
Auto Constrain					guick_start.srr	47 kB	str File	13:37
					- 🛞 quick_start.srs	6 kB	Netlist (RTL)	17:45
FSM Compiler	•	<u> </u>			guick_start.szr	17 kB	szr File	13:37
Resource Sharing					 quick_start.tlg 	869 bytes	tig File	17:45
Pipelining				4.1	Quick_start_prepass.srd	7 kB	Netlist	13:37
Retimina				1.016				
	14.945	binkly_syn.sd: Information	6 ×		Watch			6
			•	Log Paramete	auick	start Implimit		1
BTICESS Mappe	r Compl	sted with warnings	1.1	Logi a anticica	dere			
Return Code: 1								
Completion Tim		50	- 5					
TCL Script Mess	ages			Log Watch				

Figure 2-7 : Status showing synthesis has been completed

View Timing Constraints

Double Click on the **blinky_syn.sdc** file under the Constraint folder. See Figure 2-8. It will open the timing constraints for the project shown in Figure 2-9.



Synplify Pro I	-2009.1	25 Early Access 2 - [C:/SbtTools/examples/blinky/quick_start/quick_star	t_syn.prj]									
File Edit	View Pr	oject Run Analysis HDL-Analyst Options Window Tech-Support Web Help				_ 6						
D 🖸 🖲 🗉	i 👩 V		* 8 8 0 0 0 0 0 0	9,0,0	的出生	32						
0 1 00	1 III I	ARAAA										
Run		Synplify [®] Pro										
		Ready										
B Open Protect.	_	quick_start_Implmnt - Silicon Blue ICE65 : ICE65L04 : CB284 : -L	Implementation Directory									
		🔜 回 [quick_start_syn] - C:\SbtTools\examples\blinky\quick_start\quick_start_syr	C:\SbtTools\examples\blinky\quick	starthouick star	* Implant							
Close Project		E 🔗 VHDL t blinky.vhd (work)	Name Size									
Add File		E 🧭 Constraint	🕀 🔗 backup	2026	Type Directory	Modifi *						
Change File	_	Dinkly_syn.sdc	B- Coreip		Directory	17:45						
ing change me			🕀 🥏 sbt		Directory	17:20						
🙆 Add Implement	ation		Syntmp Zolace		Directory	17:45						
Minimplementation	Onking		xplace identify.log	152 bytes	Directory log File	17:45 12:49						
Tubiemencarion	opconstru	Open the SDC file	- quick_start.edf	119 kB	Edif Netlist	12:49						
Add P&R Impler	nentation		 quick_start.fse quick_start.htm quick_start.srf quick_start.srd quick_start.srl quick_start.srm 	0 bytes 348 bytes 2 kB 14 kB 6 kB 121 kB	fse File htm File scf File Netlist Netlist (RTL) Netlist (Gate)	12:49 13:20						
A View Log						12:49						
Frequency(MHz):						12:49 17:45 12:49						
1	-											
Auto Constrain	1.41		guick_start.srr	47 kB	srr File	12:49						
Auto Consolari			- (a) quick start.srs	6 kB	Netlist (RTL)	17:45						
FSM Compiler	V	•	- quick_start.szr	17 kB	szr File	12:49						
Resource Sharing			guick_start.tlg	869 bytes	tig File	17:45						
Pipelining			quick_start_prepass.srd	7 kB	Netlist	12:49						
Retiming	-		•			4.1						
quick_start_syn			Watch									
		bt node-locked LogParameter	quick	_start_Implmnt		J						
		-	4 4									
10.7.00			1									
TCL Script Mess	ages	Log Watch										

Figure 2-8 : Open the SDC file to View Timing Constraints



F	File	Edit View Project Pup	Analysis HDL-Analyst Opti	ione Wind	ow Tech	Support Web H	eln						_8
	4 (R)			1 0	()		D. 2	15 18	000	. a.	0.0.4	2 10 11 15	k
8		PRBB	3 .5 5	41 000				9 9	* *	* *	~ ~ ~	a te an se	-
				1.	Le : L		L man	Letter	In the second		Virtual		-
	Enabled	Clock Object	Clock Alias	Frequency (MHz)	Period (ns)	Clock Group	Rise At (ns)	Fall At (ns)	Duty Cycle (%)	Route (ns)	Clock	Comment	
	•	CLK_32KHz	CLK_32KHz	1	1000	default_clkgroup_i	0	200		11			110
2	•	CLK_32MHZ	CLK_32MHZ	32	31.25	default_clkgroup_	3	1					
3	-			~			-						
				1.1.1	1	~							
5				1		Tim	ing co	nstra	ints	le reli			
5				1			1993	1.		17.1			
7	1.04			the st	10.11				5 x 14	51.0			
3		1	2										
9	1	1		1				1	1				
10		12		1					1	11.5			
11													_
12	1.1	/	 Click on Pro 	ject t	abto	return t	o Proj	ect G	01				_
13		/											
Cloc	ks Cl	ack to Clock Collections	Inputs/Outputs Registe	ers Dela	y Paths	Attributes 1/0	Standards	Compile	Points Ot	ther			
q.	ack_start	syn.prt blinkly_syn.s	dc							-			_
-	_	11	Formation			B (X)				Watch			
cer	nse: sy	mplifypro_sbt node-1	ocked			Log P	arameter			quick,	_start_Implmr	×	1
						-							
			10-1110-1110-1110-										
	-					time and the second sec							

Figure 2-9 : View Timing Constraints

Viewing Hierarchical View of Synthesis Results

Under the **HDL-Analyst menu**, **Select RTL > Hierarchical View**. You will see a hierarchical RTL view of the design just synthesized. See Figure 2-10.





Figure 2-10 : Hierarchical RTL View in HDL Analyst

If you double click on one of the blocks, it will take you to the RTL for that block. See Figure 2-11.





Figure 2-11 : Double-clicking on a block will reveal its HDL code in HDL Analyst

Select Implementation

In order to ensure that the synthesized design can be successfully imported into iCEcube2, exit the Synplify Pro GUI.

Return to the iCEcube2 Navigator and **Double-click** on **Select Implementation**. See Figure 2-10. This will tell iCEcube2 which synthesis implementation to process for place and route. If you have different synthesis implementations, you will be able to select the synthesis implementation you wish to place and route. Since we only have one implementation, **select OK** when the Select Synthesis Implementation dialog box appears.





Figure 2-12 : Select Synthesis Implementation

Importing Physical Constraints

Physical constraints such as pin assignments are stored in a .PCF file (Physical Constraint File). Add the .PCF file to your project.

In the iCEcube2 Project Navigator, **Right Click** on **Specify Additional Files**. Select **Add Files**... See Figure 2-13.

Note: For information on importing physical constraints from iCEcube to iCEcube2, please refer to the **Importing Physical Constraints from iCEcube to iCEcube2** section at the end of this quick start guide.





Figure 2-13 : Specify Additional Files for Place and Route

Navigate to the <iCEcube2 Installation Directory>/examples/blinky and **Add blinky.pcf** file. See Figure 2-14.

🗭 Add Files	2 S
Add Files	Files to add >>> >>> <<
Home File name: blinky.pcf Files of type: Constraint(*.sdc *.scf *.clb *.pcf *.mtcl) Ok. Can	Details

Figure 2-14 : Add .pcf file



Import Place & Route Input Files

The next step is to import the files for Place and Route. **Double-click** on **Import P&R Input Files** in the Project Navigator. See Figure 2-15. Once completed you will see a green check next to Import P&R Input Files. See Figure 2-16.



Figure 2-15 : Import P&R Input files





Figure 2-16: Successful Import of P&R Input Files

Place the Design

Double-click on Run Placer

Once placement is complete, a green check will appear and the Output window will show information about the placement of the design. See Figure 2-17.





Figure 2-17 : Place complete

View Floorplanner

At this point, since placement has been completed, you can view the placement of the design by opening the Floorplanner. You can open the Floorplanner by going to the menu and selecting **Tool > Floorplanner** or you can also select the Floorplanner Icon. See Figure 2-18.



roject Name: quick_start	Output	Floor Planner
Design Flow	Floorplanner Icon	8×
 Specify Synthesis Input Files Design Files 	Logic Instance Instance Type Lo	
Constraint Files	@ 🛧 IN_MUX_bfv_14_6 SB_CARRY_IN 14	, 6,8
P&R Input Files		
Select Implementation (qui quick_start.edf	⊕ ♠ IN_MUX_bfv_14_1 SB_CARRY_IN 14 ⊕ ♠ IN_MUX_bfv_14_2 SB_CARRY_IN 14	
- quick_start.scf	■ ♣ IN_MUX_bfv_14_2 5B_CARRY_IN 14	
 Specify Additional Files Import P&R Input Files 	A IN_MUX_bfv_14_4 SB_CARRY_IN 14	
✓ Place		
Route	H A IN_MUX_bfv_15_3 SB_CARRY_IN 15	, 3,8
 Bitmap Output Files 	⊞ ♠ IN_MUX_bfv_15_4 SB_CARRY_IN 15 ⊕ GONSTANT_ONE_L SB_LUT4 16	
Cuput Hes It reports		
bitmap		
 simulation netlist Device/Operating Condition 		, 8, 1 , 7, 0
Device/operating contractor	DIVIDE_32MHz_un SB_LUT4 14	, 4,2 💌
DeviceFamily ICE65	Logic GPIO Global RAM Net	Region
Device L04 Device Package CB284	World View	5×
Power Grade L	ADDRESS OF CONTRACT OF CONTRACT.	
Operating Condition		
Core Voltage(V) 1.14		
Temperature(C) 70		
	1000 C	

Figure 2-18 – Floorplanner

View the Package Viewer

You can also see how pins were placed for your design by selecting the Package Viewer. You can select the package viewer by going to the menu and selecting **Tool >> Package Viewer** or you can also select the Package Vierwer Icon. See Figure 2-19.





Figure 2-19 : Package Viewer

Route the Design

Double-click on Route in the project navigation window. Place and Route have been separated into different steps as to allow you to re-route the design after making placement modifications in the floorplanner without having to re-run the placer.

Perform Static Timing Analysis

Now that you have routed the design, you can perform timing analysis to check to see if the design is meets your timing requirements. To launch the timing analyzer, go to the menu and **select Tool > Timing Analysis**. You can also select the Timing Analysis Icon. See Figure 2-22.



Project Name: guick_start 5	0	g Analysis	icetest_sbt.rpt	Tir	ning Analyzer	
Design Flow		_	receiv_becapt		in a second second	
 Specify Synthesis Input Files Design Files blinky.vhd Constraint Files blinky.sdc 	Summary Analyze Path	s		Timing Corner: O Best	: 🕑 Typical 🔘 Worst	Save Summar
Synthesis P&R Input Files	Clock Name	Worst Slack(ps)	FMAX(MHZ)	Target Frequency(M	IH2) Failing Path #	
🗉 🥒 Select Implementation (qui	1 CLK_32KHz	991479	117.35	1.00	0	
 Specify Additional Files Constraint Files 	2 CLK_32MHZ	18696	79.65	32.00	0	
Import P&R Input Files	clock divider 1Hz	997087	343.26	1.00	a	
Place Route	³ COUNTER_inferred dock_divider_32MHz	997087	343.28	1,00	0	
Dutput Files	" COUNTER_inferred	33/08/	343.28	1,00	0.	
Device Info DeviceFamily ICE65 Device L04						
	Paths Summary(1/1)			Show More	0	Save Summary Save Detail
Operating Condition	Start Point *	Start Edge End Po	int End Edge	Slack Delay Skew	Fanout	
	1 Divider_to_1Hz_C R	ISE Divider_ti	o RISE	991479 7853 0	5	
Device L04 Device Package CB284 Power Grade L © Operating Condition	Start Point			Slack Delay Skew	Fanout	[29A6 20009A3] [25

Figure 2-22 : Timing Analysis Summary

You can see from the timing analysis that our 1MHz design is running at over 100 MHz and our 32 MHz clock is running at over 70 MHz (worst case timing). If we were not meeting timing, the timing analyzer will allow you to see your failing paths and do a more in-depth analysis. For this tutorial, we won't go into details on timing slack analysis.

Perform Power Analysis

iCEcube2 also comes with power estimator tool. To launch the power estimator, to the menu and select **Tool >> Power Estimator**. You can alternatively select the power estimator icon. Figure 2-23. There are multiple tabs in the Power Estimator tool including Summary, IO, and Clock Domain. On the Summary tab, **change the Core Vdd to 1.0V** and make sure all **IO voltages are at 2.5V**. Then hit **calculate**. The estimator will update with power information for both static and dynamic power. For more information on using the IO and Clock Domain tabs, please refer to the detailed section on the Power Estimator tool.



Core Vdd(V): 1.00 🛛 🖌 F	Power Grade: L	Dynamic Power Break	nwob
IO Voltage		Core Power(mW):	1.30151
Left Bank IO Voltage(V):	2,5	IO Power(mW):	2
Right Bank IO Voltage(V):	2.5	Power Consumption -	1415
Top Bank IO Voltage(V):	2.5	Static Power(mW): 0.013
		Dynamic Power(mW): 3.30151
Bottom Bank IO Voltage(V):	2.5	Total Power(mW): 3.31451

Figure 2-23 : Power Estimator

Programming the Device

In order to program a device, you will need to generate a programming file. In the project navigator, **double click on Bitmap**.

You are now ready to program an iCE65 mobileFPGA deice with the generated bitmap. Invoke the programmer from the **Programming** icon which is now enabled in the Project Navigator. Alternately, you may invoke it from the **Tool>Programmer** menu item.

The iCEman65 Evaluation Kit Board includes an on-board USB 2.0 programming solution to program the on-board SPI serial Flash or the iCE65 device directly. In a future iCEcube release, the utility will also support direct programming of the iCE65 device, although this is not currently supported.

Additional details on programming a device are provided in a separate section *Programming the Device* in *Chapter 5 iCEcube Physical Implementation Tools.*

Programmer	X
Programming Options	
Programming Hardware	
Eval Board	*
Eval Board	
Digilent USB Cable "DCabUsb" Digilent USB Cable "CCabUsb"	
Digilent USB Cable "BCabUsb"	
iCE Cable	
External SPI Serial Flash PROM	
m25p80	*
Image	
Image Type	
Single Image Image	File Settings
O Multiple Images	
Advanced Execute	Quit

2-24 : Programmer Graphical User Interface

Addendum:

Importing Physical Constraints from iCEcube to iCEcube2

For users who have created physical constraints using iCEcube, this section describes how to import and convert those constraints for use in iCEcube2. This section will demonstrate how to import a .MTCL file from iCEcube and save it into .PCF format used in iCEcube2.

In the iCEcube2 project navigator, **Right-click** on **Specify Additional Files.** See Figure 2-25





Figure 2-25 : Add additional constraint file

Navigate to the <iCEcube2 Installation Directory>/examples/blinky and **Add blinky.mtcl** file. See Figure 2-26.

🗭 Add Files	2 🛛
Look in: C:/SbtTools/examples/blinky	Files to add blinky_constraints.mtd >> <<<
File name: blinky_constraints.mtcl Files of type: FILES(*.sdc *.scf *.db *.pcf *.mtcl *.edf *.edif *.edn *.vqm) Ok Can	Details

Figure 2-26 : Add .mtcl file



Import Place & Route Input Files

The next step is to import the files for Place and Route. **Double-click on Import P&R Input Files** in the Project Navigator. See Figure 2-27. Once completed you will see a green check next to Import P&R Input Files. See Figure 2-28.



Figure 2-27 : Double-click on Import P&R Input files





Figure 2-28: Successful Import of P&R Input Files

Saving Physical Constraints into .pcf Format

Open the Pin Constraints Editor by going to the menu and selecting **Tool>Pin Constraints Editor** or you can also select the Pin Constraints Editor Icon. See Figure 2-29. You will se a list of pin assignments that are Locked under the locked column. **Uncheck and Recheck one of the pins under the locked column**. The save icon will now become an active icon. **Click on the Save physical constraints icon**. This will bring up a dialog box where you can save the CLB and PCF files. **Hit OK**. See Figure 2-30. The .PCF file contain physical constraints in the design used for place and route.



oject Name: quick_start	6×	Output Floor Planner	Pin Constrain	ts Editor	Package View		
Design Flow Design Flow Constraint Files Constraint Files Synthesis RR Input Files Select Implementation (qui quick_start.scf		Object List	Pin Location	Bank	10 Standard	Pull Up	Locked
	1	PMOD_B3T_339[1]	M7	Left			
	2	PMOD_B3T_339[2]	MB	Left		/	
	(qui	PMOD_831_339[3]	N7	Left			
	4	PMC0_B3T_J39[4]	™ Unche	ck and i	recheck Lock	ed Box	
Specify Additional Files	5	PMC0_R_330[1]	Y20	Bottom	A. 3. 6. 4		
 Import PAR. Input Files Place Place Place Place Bitmap Bitmap Subuston netlist Device/Operating Condition Device Info Device Tamily CE65 Device Package C8264 Power Grade Coperating Condition Core Voltage(V) 1.14 Temperature(C) Po 	6	PMO0_82R_30[2]	¥18	Bottom			2
	7	PMOD_82R_30(3)	Y19	Bottom			
	8	PMOO_B2R_J30[4] Pin Constraints	s Editor icc	Bottom			1
	9	PMOD_B0R_J13[1]	H14	Тор			
	10	PMOD_BOR_J13[2]	C14	Тор			
	11	PMOD_B0R_313[3]	C15	Тор			
	12	PMOD_B0R_J13[4]	C16	Тор			2
	13	PMOD_B3B_138[1]	P3	Left			2
	14	PMOD_B3B_138[2]	us	Left			
	15	PMOD_B38_138[3]	KB	Left			
	16	PMOD_638_138[4]	¥3	Left			1
	17	PM00_B1T_J20[1]	D20	Right			
	18	PM00_B1T_320[2]	G20	Right			
	19	PMOD_B1T_J20[3]	F18	Right			2
	20	PMOD_B1T_320[4]	316	Right	2		1
	21	PMOD_B1B_121[1]	P15	Right			
	22	PMOD_B1B_J21[2]	R20	Right			
	23	PMOD 818 J21[3]	T20	Right			

Figure 2-29 : Pin Constraints Editor.



Figure 2-30 : Pin Constraints Editor.