Implementing PCIe Control Plane Design

Libero SoC v11.4 Flow Tutorial for IGLOO2 FPGA



Table of Contents

	Implementing PCIe Control Plane Design - Libero SoC v11.4 Flow Tutorial for IGLOO2 FPGA	
	Introduction	
	Associated Project Files	
	Components Used	
	Design Overview	4
	Step 1: Creating a Libero SoC Project	6
	Launching Libero SoC	6
	Instantiating the SERDESIF Component in PCIe_Demo_top SmartDesign	12
	Instantiating Debounce Logic in PCIe_Demo_top SmartDesign	16
	Instantiating Deboarie Logie in Ore_Demo_top SmartDesign	23
	Instantiating CoreAHBLSRAM in PCIe_Demo_top SmartDesign	25
	Instantiating Clock Conditioning Circuitry (CCC) in PCIe_Demo_top SmartDesign	27
	Connecting Components in PCIe_Demo_top SmartDesign	29
	Step 2: Developing the Simulation Stimulus	
	Step 3: Simulating the Design	
	Step 4: Generating the Program File	43
	Step 5: Programming the IGLOO2 Board Using FlashProment.	46
	Step 6: Connecting the Evaluation Kit to the Host PC	48
	Step 7: Running the Design	
	Running the Design on Windows	49
	Running the Design on Linux	
	Conclusion	69
А	IGLOO2 Evaluation Kit Board	. 70
В	IGLOO2 Evaluation Kit Board Setup for Laptop	. 71
С	List of Changes	. 74
_		
D	Product Support	
	Customer Service	
	Customer Technical Support Center	
	Technical Support	
	Website	
	Contacting the Customer Technical Support Center	
	Email My Cases	
	Outside the U.S.	
	ITAR Technical Support	



Introduction

This tutorial demonstrates the embedded PCI[®]express feature of IGLOO[®]2 field programmable gate array (FPGA) devices and how this can be used as a low bandwidth control plane interface. A sample design is provided to access IGLOO2 PCIe Endpoint (EP) from host PC. It can run on both Windows and RedHat Linux Operating System (OS). A GUI installer, host PC drivers for Windows OS, and a Linux PCIe application for Linux OS are provided for reading and writing to the IGLOO2 PCIe configuration and memory space. This tutorial provides a complete design flow starting from a new project to a working design on the IGLOO2 Evaluation Kit board.

After completing this tutorial, you will be able to perform the following tasks:

- Create a Libero[®] System-on-Chip (SoC) project
- Develop the Simulation Stimulus
- Simulate the design
- Generate the programming file
- Run the PCIe application

Tutorial Requirements

 Table 1 lists the hardware and software requirements of IGLOO2 PCIe Control Plane tutorial.

 Table 1 • Reference Design Requirements and Details

Reference Design Requirements and Details	Description
Hardware Requirements	
IGLOO2 Evaluation Kit	Rev C or later
 12 V adapter FlashPro4 programmer USB A to Mini-B cable 	
Host PC or Laptop with an available PCIe 2.0 Gen 1 or Gen 2 compliant slot	64-bit Windows 7 OS, 64-bit Red Hat Linux OS (Kernel Version: 2.6.18-308)
Express Card slot and PCIe Express card adapter (for Laptop only)	-
Softwa <mark>re Requi</mark> rements	
Libero [®] System-on-Chip (SoC)	v11.4
FlashPro programming software	v11.4
Host PC Drivers (provided along with the design files)	-
GUI executable (provided along with the design files)	-

Note: PCIe Express card adapter is not supplied with the IGLOO2 Evaluation Kit.



Associated Project Files

You can download the associated project files for this tutorial from the Microsemi[®] website: *http://soc.microsemi.com/download/rsc/?f=M2GL_PCIE_Control_Plane_11p4_DF.*

Design files include:

- 1. Libero project
- 2. Programming files
- 3. Host PC drivers and GUI executable for Windows OS
- 4. Host PC drivers and PCIe application for Linux OS
- 5. Readme file
- 6. Source files

Refer to the Readme.txt file provided in the design files for the complete directory structure.

Components Used

This tutorial uses the following components of the IGLOO2 device:

- Fabric clock conditioning circuitry (CCC)
- High speed serial interfaces (SERDES_IF_0)
- CoreGPIO
- CoreAHBLSRAM
- Bus interfaces CoreAHBLite, CoreAPB3, and CoreAHBTOAPB3

Design Overview

IGLOO2 FPGA devices integrate a fourth-generation flash-based FPGA fabric and high performance communication interfaces on a single chip. The IGLOO2 high speed serial interface (SERDESIF) provides a fully hardened PCIe EP implementation and is compliant with PCIe Base Specification Revision 2.0 and 1.1. For more details refer to the *IGLOO2 FPGA High Speed Serial Interfaces User Guide*.

The design helps access the IGLOO2 PCIe EP from the host PC. A GUI and Linux PCIe application are provided for read and write access to the IGLOO2 PCIe configuration and memory space of BAR0 and BAR1. The IGLOO2 PCIe BAR0 and BAR1 are configured in 32-bit mode.





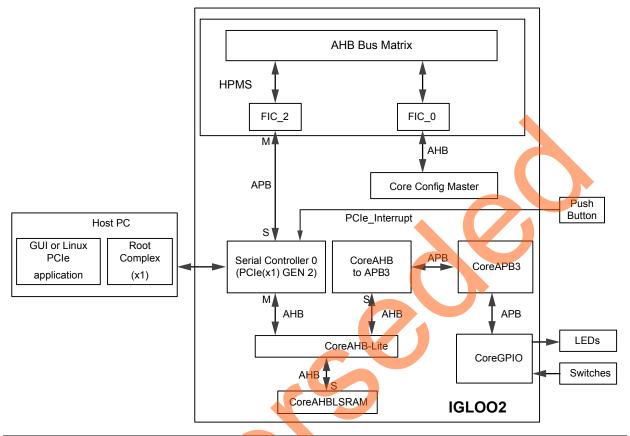


Figure 1 shows a detailed block diagram of the design implementation.

Figure 1 • PCIe Control Plane Block Diagram

The PCIe EP device receives commands from the host PC through the GUI or Linux PCIe application and performs corresponding memory writes to the IGLOO2 fabric address space.

The SERDES_IF_0 is configured for a PCIe 2.0, x1 link width with GEN2 speed. The PCIe interface to the fabric uses an AMBA High-speed Bus (AHB). The AHB master interface of SERDESIF is enabled and connected to the slaves CoreAHBLSRAM and CoreGPIO using CoreAHBLite, CoreAHBTOAPB, and CoreAPB3 bus interfaces.

SERDES_IF_0 is initialized by CoreConfig Master. This IP is configured by the System Builder.

The AXI master windows of the SERDESIF PCIe provide address translation for accessing one address space from another address space as the PCIe address is different from the IGLOO2 AHB bus matrix address space. The AXI master window 0 is enabled and configured to translate the BAR0 memory address space to the CoreGPIO address space to control the LEDs and DIP switches.

The AXI master window 1 is enabled and configured to translate the BAR1 memory address space to the CoreAHBLSRAM address space to perform read and writes from PCIe.

CoreGPIO is enabled and configured as below:

- GPIO_OUT [7:0] connected LEDs
- GPIO_IN [3:0] connected to DIP switches

The PCIe interrupt line is connected to the **SW4** push button on the IGLOO2 Evaluation Kit. The FPGA clocks are configured to run the FPGA fabric and HPMS at 100 MHz.



Step 1: Creating a Libero SoC Project

This step helps you create an IGLOO2 PCIe control plane design using the Libero tool.

Launching Libero SoC

- 1. Click Start > Programs > Microsemi Libero SoC v11.4 > Libero SoC v11.4, or click the shortcut on your desktop to open the Libero SoC Project Manager.
- Create a new project by selecting New on the Start Page tab (highlighted in Figure 2), or by clicking Project > New Project from the Libero SoC menu.



Figure 2 • Libero SoC Project Manager

- 3. Enter the information as required for the new project and the device in the **New Project** dialog box as shown in Figure 3.
 - Project
 - Name: PCIE_Demo
 - Location. Select an appropriate location (for example, D:/microsemi_prj)
 - Preferred HDL type: Select Verilog or VHDL



- Device (select the following values using the drop-down list provided):
 - Family: IGLOO2
 - Die: M2GL010T
 - Package: 484 FBGA
 - Speed: -1
 - Core Voltage: 1.2
- Operating conditions: COM

Project							
Enable Block Creation							
Name:	PCIE_Demo						
Location:	D:/Microsemi_prj				Browse		
Prefered HDL type:) Verilog 🔘 VHI	DL					
Description:							
🗼 Edit Tool Profiles							
Device							
Family:	IGLOO2 👻						
Die:	M2GL010T -						
Package:	484 FBGA 🔻						
Speed:	-1 🔻						
Core Voltage (V):	1.2 🔻	Ramp Rate:	100ms Minin	num 👻			
Operating Conditions:							
	Rar	ige	Best	Typical	,	Worst	
Junction Tempe	rature (C) COM			25	85		
Core Voltage (V	сом	+ 1.260		1.200	1.140		
System Controller Sus	spend Mode						
PLL Supply Voltage (V):	2.5 🔻						
Design Templates and Cre	ators						
📝 Use Design Tool							
		Core			Version		
Use System Builder					1.0		
						Show only la	test version

Figure 3 • Libero SoC New Project Dialog Box

- 4. Clicking **Edit Tool Profiles** (highlighted in Figure 3) displays the **Tool Profiles** window as shown in Figure 4. Check the below tool settings:
 - Synthesis: Synplify Pro ME I-2013.09M-SP1
 - Simulation: ModelSim 10.3a



- Programming: FlashPro 11.4

Tools Synthesis	Synthesis pro	files	•	_/ ×
Simulation Stimulus	Active	Name	Path	
Programming Identify Debugger	•	Synplify Pro ME	D: \Microsemi\Libero_v11.4\Synopsys\synplify_I201309MSP1-1\bin\synplify_r	oro.exe



- 5. Click **OK** on the **Tool Profiles** window.
- 6. Click OK on the New Project window. This displays the System Builder dialog box.
- 7. Enter a name for your system as shown in Figure 5.

System Builder		2 X		
Enter a name for yo	our system:	Ø		
PCIe_Demo				
Help	ОК	Cancel]	

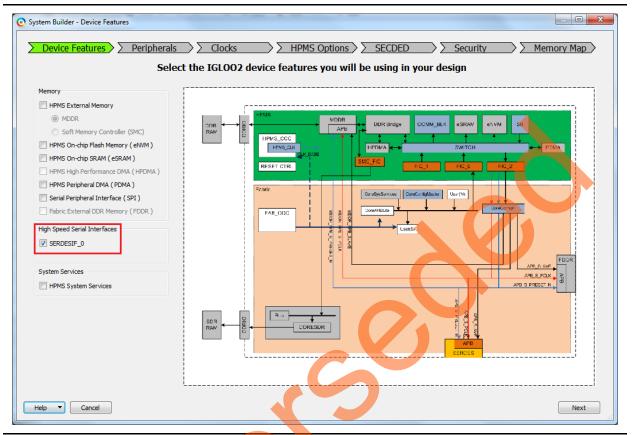
Figure 5 • Create New System Builder Dialog Box

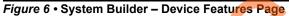
8. Enter **PCIe_Demo** as the name of the system and click **OK**. The System Builder dialog box is displayed with the **Device Features** page open by default.





9. In the System Builder – Device Features page, select SERDESIF_0 under High Speed Serial Interfaces as shown in Figure 6.





10. Click **Next**. The **System Builder – Peripherals** page is displayed. Leave all the default selections.



11. Click Next. The System Builder – Clock page is displayed, as shown in Figure 7. Select System Clock source as On-chip 25/50 MHz RC Oscillator and HPMS_CLK as 100 MHz.

		Configure your sub	osystem clocks	
lock Fabric CCC Chip	Oscillators			
System Clock				
50.0 MHz				
On-chip 25/50 MHz RC Oscilla	ator	-		DDR Bildge COMM_ELK BSRAM EVVM SI
HPMS Clock			PMS_CCC	HPDMA + AHB Bus Matrix
HPMS_CLK	= 100.00 MHz 100.00		T PN3 CER	
	- 100.00			
MDDR Clocks				
MDDR_CLK	= HPMS_CLK * 1 V		APR_0_016	
DDR/SMC_FIC_CLK	= MDDR_CLK / 1		FIC_0_C_K	
Fabric Interface Clocks				
FIC_0_CLK	= HPMS_CLK / 4 • 25.000		MS+	TIC.0
FIC_1_CLK	= HPMS_CLK / 1		Lowest frequency	
Fabric DDR Clocks			RC_0_C_K	FIC_D Solaryslam
FDDR_CLK	= 100 MHz			
FDDR_SUBSYSTEM_CLK	= FDDR_CLK / 1		USC	
		Fat	oric	

Figure 7 • System Builder – Clocks Page

- 12. Click Next. The System Builder HPMS Options page is displayed. Leave all the default selections.
- 13. Click Next. The System Builder SECDED page is displayed. Leave all the default selections.
- 14. Click Next. The System Builder Security page is displayed. Leave all the default selections.
- 15. Click Next. The System Builder Memory Map page is displayed. Leave all the default selections.
- 16. Click Finish.

The **System Builder** generates the system based on the selected options. The System Builder block is created and added to the Libero SoC project automatically, as shown in Figure 8.



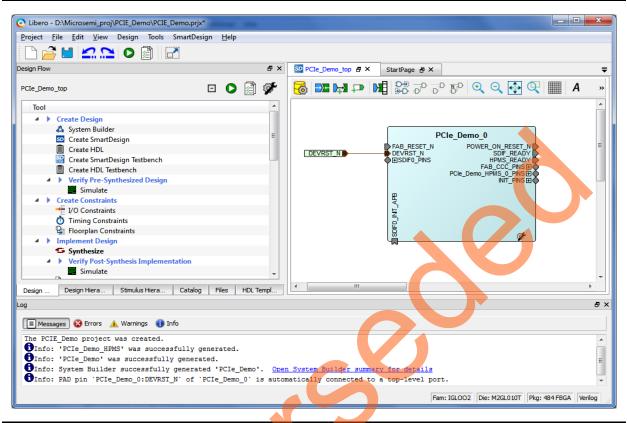


Figure 8 • IGLOO2 FPGA System Builder Generated System

The two soft cores (CoreResetP and CoreConfigP) are automatically instantiated and connected by the System Builder.

Note: CoreResetP and CoreConfigP are responsible for the reset and configuration of peripherals. In this case, they are used to reset and configure the SERDESIF module. These modules are included in the System Builder generated component.



Instantiating the SERDESIF Component in PCIe_Demo_top SmartDesign

The Libero SoC Catalog provides IP cores that can be easily dragged and dropped into the SmartDesign Canvas workspace. Many of these IPs are free to use while several require a license agreement. The SERDESIF module that supports the PCIe embedded interface is included in the catalog.

To instantiate the SERDESIF component in the **PCIe_Demo_top** SmartDesign, expand the **Peripherals** category in the Libero SoC **Catalog**.

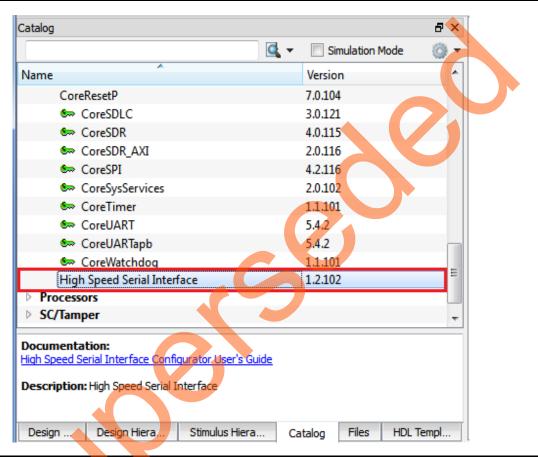


Figure 9 • IP Catalog

- 1. Drag the **High Speed Serial Interface** onto the **PCIe_Demo_top SmartDesign** canvas. If the component appears shadowed in the **Vault**, right-click the name and select **Download**.
- 2. Double-click the **SERDES_IF_0** component in the SmartDesign canvas to open the **SERDES** configurator. Configure the SERDES with the following settings as shown in Figure 10:
 - Identification
 - Simulation Level: BFM PCIe
 - Protocol Configuration
 - Protocol1: Type: PCIe
 - Protocol1: Number of Lanes: x1
 - Lane Configuration
 - Speed: Lane0: 5.0 Gbps (Gen2)
 - PCIe/XAUI Fabric SPLL Configuration
 - CLK_BASE Frequency (MHz): 100



Number of Lanes x1 -	figure PCIe Type	ber of Lanes
ne Configuration	Lane 0	Lane 1 Lane 2 Lane 3
Speed	5.0 Gbps(Gen2)	-
Reference Clock Source	REFCLK0 (Differential)	
PHY RefClk Frequency (MHz)	100	
Data Rate (Mbps)	N/A	
Data Width	N/A	
VCO Rate (MHz)	N/A	
FPGA Interface Frequency (MHz)	N/A	
CIE/XAUI Fabric SPLL Configuration	Ś	

Figure 10 • High Speed Serial Interfaces Configurator

- 3. Click **Configure PCIe** to configure the following settings as shown in Figure 11.
 - Identification Registers
 - Device ID: 0x11AA (Microsemi ID)
 - Subsystem Vendor ID: 0x11AA (Microsemi ID)
 - Fabric Interface (AXI/AHBLite)
 - Bus: select as AHBLite from the drop-down list

- Base Address Registers

- Bar 0 Width: 32-bit, Size: 1 MB (to access CoreGPIO address space)
- Bar 1 Width: 32-bit, Size: 64 KB (to access CoreAHBLSRAM memory)



Configuration	Master Interf	ace Slav	e Interface			
Identification Reg	gisters					
Vendor ID	0x11A	A	Device ID		0x11AA	
Subsystem Vendo	or ID 0x11A	A	Subsystem D	Device ID	0x0000	
Revision ID	0x0000)	Class Code	[0x0000	
Fabric Interface ((AXI/AHBLite)					
Bus AHBLite	 Interf 	face Master	•			
Base Address Re						
_	2 Bits 🔹	Size 1 N		Prefetchable		
	2 Bits 🔹		KB 👻	Freietaliable		
	one 🔻	Size	×D ·	Prefetchable		
			7	Prefetenable		
	one 🔻	Size				
Bar 4: Width	one 🔻	Size		Prefetchable		
Bar 5: Width	one 🔻	Size				
Options						
PHY Reference C	lock Slot	Slot	L2/P2			
PCIe Specificatio	n Version	Version 2. 🔻	Transmit	Swing		
Interrupts		INTx -	De-emphasis	DB 3.5	•	

Figure 11 • PCIE Configuration for Protocol 1



- 4. Click Master Interface tab to configure the PCIe master windows. The PCIe AXI master windows are used to translate the PCIe address domain to the local device address domain. In this tutorial the PCIe AXI master windows are used to translate the address of BAR0 and BAR1 to CoreGPIO address and CoreAHBLSRAM address.
 - Select Window 0 and configure the following settings:
 Size: Select as 1MB from the drop-down list
 PCIe BAR: Select as Bar0 from the drop-down list
 Local Address: Enter values as 0x40000 to translate the BAR0 address space to CoreGPIO address (0x4000_0000)
 - Select Window 1 and configure the following settings:
 Size: Select as 64KB from the drop-down list
 PCle BAR: Select as Bar1 from the drop-down list
 Local Address: Enter values as 0x20000 to translate the BAR1 address space to CoreAHBLSRAM address (0x2000_0000)

For more information on PCIe address translation, refer to the "Address Translation on the AXI Master Interface" section of the *IGLOO2 FPGA High Speed Serial Interfaces User Guide*.

Figure 12 shows the Master Interface Configuration window.

Configuration	Master Interface Sla	ave Interface	
Window 0 -			
Size	1 MB	PCIe BAR	Bar 0 🔻
Local Address	0x40000	PCIe Address	0x0000
Window 1			
Size	64 KB	PCIe BAR	Bar 1 🔻
Local Address	0x20000	PCIe Address	0x0000
Window 2			
Size	4 KB	PCIe BAR	Bar 0 💌
Local Address	0x0000	PCIe Address	0x0000
Window 3			
Size	4кв -	PCIe BAR	Bar 0 👻
Local Address	0x0000	PCIe Address	0x0000
0			

Figure 12 • Master Interface Configuration Window

- 5. Click **OK** to close the PCIE Configuration for protocol 1 window.
- 6. Click OK to save and close the High Speed Serial Interface Configurator window.

Instantiating Debounce Logic in PCIe_Demo_top SmartDesign

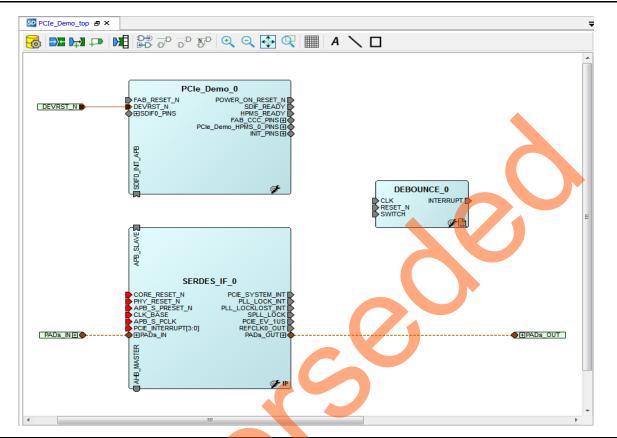
The tutorial provides a push button (**SW4**) on the IGLOO2 Evaluation Kit to send an interrupt to the host PC. This push button generates switch bounce that causes multiple interrupts to PCIe. Debounce logic is required to avoid the switch bounce.

- 1. To add the Debounce logic to the PCIe demo design, click File > Import > HDL Source files.
- Browse to the Debounce.v or Debounce.vhd file location in the design files folder: <u>M2GL_PCIE_Control_Plane_DF</u>\Source Files. Figure 13 shows the DEBOUNCE component in the Design Hierarchy window.

Design Hierarchy	E X
Show: Components	HOLE
▲ ① work	
TLOSC_FAB (osc_comps.v)	
🗎 XTLOSC (osc_comps.v)	
RCOSC_1MHZ_FAB (osc_com	ps.v)
RCOSC_1MHZ (osc_comps.v)	
▲ 1 SO PCIe_Demo_top	
PCIe_Demo	
DEBOUNCE (Debounce.v)	
COREAHBLITE_LIB	
	•
Desig Design Hier Stimulus Hie	Catalog Files

Figure 13 • DEBOUNCE Component in the Design Hierarchy Window





3. Drag the **DEBOUNCE** component from the **Design Hierarchy** into the **PCIe_Demo_top SmartDesign** canvas. Figure 14 shows Debounce in **PCIe_Demo_top**.

Figure 14 • DEBOUNCE Component in the PCIe_Demo_top SmartDesign Canvas



Instantiating the Bus Interfaces in PCIe_Demo_top SmartDesign

To instantiate the CoreAHBLite, CoreAPB3, and CoreAHBtoAPB3 in the PCIe_Demo_top SmartDesign, expand the **Bus Interfaces** category in the Libero SoC **Catalog**. Figure 15 shows the Libero IP **Catalog**.



1. Drag CoreAHBLite, CoreAHBtoAPB3, and CoreAPB3 Bus interfaces into the PCIe_Demo_top SmartDesign canvas. If the component appears shadowed in the **Vault**, right-click the name and select download. Figure 16 shows the Libero top-level design with bus interfaces.

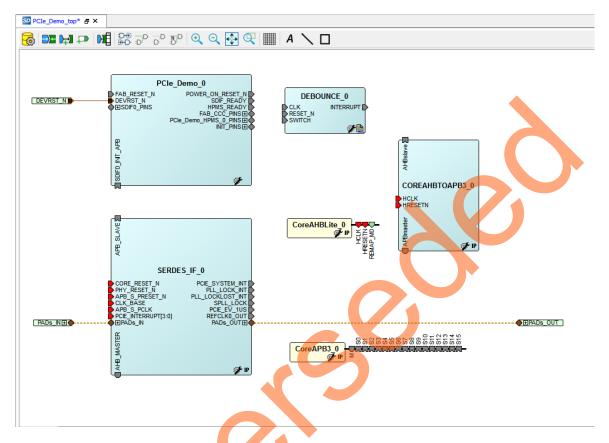


Figure 16 • CoreAHBLite, CoreAHBtoAPB3, and CoreAPB3 Bus Interfaces in the PCIe_Demo_top SmartDesign Canvas



Double-click CoreAHBLite_0 to configure it. Figure 17 shows the Configuring CoreAHBlite_0 window.

			16 slave slots, each of size 256MB	_	
		seen by slave connected to			
		0x00000000 - 0x7FFFFFF	0x8000000 - 0xFFFFFFF		
Allocate memory space to combined re	gion slave			•	
Slot 0:	1	Slot 1:	Slot 2:	Slot 3:	
Slot 4:]	Slot 5:	Slot 6:	Slot 7:	
Slot 8:		Slot 9:	Slot 10:	Slot 11:	
Slot 12:		Slot 13:	Slot 14:	Slot 15: 🔲	
Enable Master access					
M0 can access slot 0:	M1 car	access slot 0:	M2 can access slot 0:	M3 can access slot 0:	
M0 can access slot 1:	M1 car	access slot 1:	M2 can access slot 1:	M3 can access slot 1:	
M0 can access slot 2:	M1 car	access slot 2:	M2 can access slot 2:	M3 can access slot 2:	
M0 can access slot 3:	M1 car	access slot 3:	M2 can access slot 3:	M3 can access slot 3:	
M0 can access slot 4:	M1 car	access slot 4:	M2 can access slot 4:	M3 can access slot 4:	
M0 can access slot 5:	M1 car	access slot 5:	M2 can access slot 5:	M3 can access slot 5:	
M0 can access slot 6:	M1 car	access slot 6:	M2 can access slot 6:	M3 can access slot 6:	
M0 can access slot 7:	M1 car	access slot 7:	M2 can access slot 7:	M3 can access slot 7:	
M0 can access slot 8:	M1 car	access slot 8:	M2 can access slot 8:	M3 can access slot 8:	

Figure 17 • Configuring CoreAHBLite_0

5

- 3. Configure CoreAHBLite_0 with the below settings:
 - Memory Space: Select from the drop-down list as '4 GB addressable space apportioned into 16 slave slots, each of size 256 MB'.
 - Select M0 can access slot 2 to access CoreAHBLSRAM from PCIe.
 - Select M0 can access slot 4 to access CoreGPIO from PCIe.
- 4. Click OK to save and close the Configuring CoreAHBLite_0 window.



uration				-
Data Width Configuration				
	APB Master Data Bus W			
	32-bit	16-bit 🔘 8-bit		
Address Configuration				
Number of address bits drive	n by master:	28		
Position in slave address of u	inner 4 hits of master address:	[27:24] (Ignored if master add	dress width >= 32 bits)	
	pper 4 bits of master address.		aless width >= 52 bits)	
Indirect Addressing:		Not in use		
Allocate memory space to combin	ed region slave	•		=
Slot 0:	Slot 1:	Slot 2:	Slot 3:	
Slot 4:	Slot 5:	Slot 6:	Slot 7:	
Slot 8:	Slot 9:	Slot 10:	Slot 11:	
SIDE 8:	5101 9:	SIDT IU:	Slot II:	
Slot 12:	Slot 13:	Slot 14: 🔲	Slot 15:	
Enabled APB Slave Slots				
Slot 0: 📝	Slot 1:	Slot 2:	Slot 3:	
Slot 4:	Slot 5:	Slot 6:	Slot 7:	
Slot 8·	Slot 9:			
Slot 8:	Slot 9:	Slot 10:	Slot 11:	
Slot 12: 🔲	Slot 13:	Slot 14:	Slot 15: 🔲	
Testbench:		User		• •
				•

5. Double-click CoreAPB3 to configure it. Figure 18 shows the **Configuring CoreAPB3_0** window.

Figure 18 • Configuring CoreAPB3_0

- 6. Configure CoreAPB_0 with the below settings:
 - Under Data Width Configuration, select APB Master Data Bus Width as 32-bit.
 - Under Address Configuration, select Number of address bits driven by master as '28' and Position in slave address of upper 4 bits of master address as '[27:24](Ignored if master address width >=32 bits)' using the drop-down list.
 - Select Enabled APB Slave Slots as Slot 0.
- 7. Click **OK** to save and close the **Configuring CoreAPB3_0** window.



8. Figure 19 shows the PCIe_Demo_top in SmartDesign after configuring CoreAHBLite and CoreAPB3 bus interfaces.

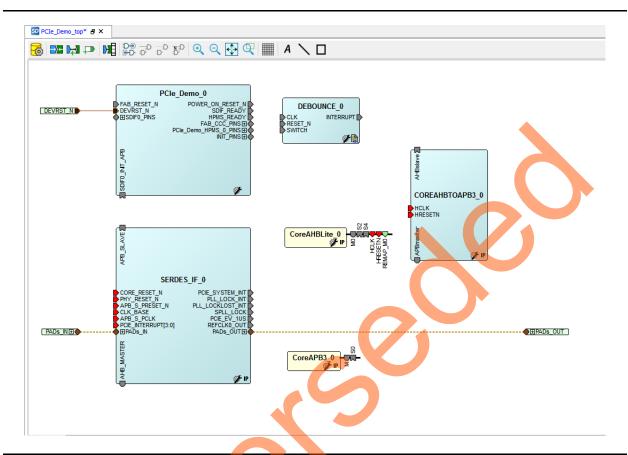


Figure 19 • CoreAHBLite and CoreAPB3 Bus Interfaces in the PCIe_Demo_top SmartDesign Canvas After Configuration



Instantiating CoreGPIO in PCIe_Demo_top SmartDesign

To instantiate CoreGPIO in the PCIe_Demo_top SmartDesign, expand the **Peripherals** category in the Libero SoC **Catalog** as displayed in Figure 20.

Catalog	5 ×
	🔍 👻 🔲 Simulation Mode 🎯 🕶
Name	Version
Arithmetic	
Bus Interfaces	
Clock & Management	
▷ DSP	
▷ I/O	
Macro Library	
Memory & Controllers	
4 Peripherals	
COREMMC	2.0.100
🖙 CORERMII	2.0.102
See Core1553BRM	4.0.101
🖙 Core1553BRT	4.0.101
🗫 Core16550	3.2.100
🖙 Core3DES	3.0.105
🖙 Core429	3,9.101
🗫 Core429_APB	3.9.101
CoreAES128	3.2.130
CoreConfigP	7.0.105
CoreDES	3.0.106
See CoreGPIO	3.0.120
CoreHPDMACtrl	2.1.103
🖙 CoreI2C	7.0.102
See CoreInterrupt	1.1.101
CoreJESD204BRX	2.4.126
CoreJESD204BTX	2.2.109
See CoreLPC	3.2.104
See CoreMBX	2.1.014 -
Documentation:	
CoreGPIO HB.pdf	
CoreGPIO RN.pdf	

Figure 20 • IP Catalog

1. Drag **CoreGPIO** into the PCIe_Demo_top SmartDesign canvas. If the component appears shadowed in the **Vault**, right-click the name and select download.



-Global Config	uration				
	APB Data Wid	:h: 32 🔻	Numbe	r of I/Os: 8 🔹	
	Single-bit inter	rupt port: Disabled 💌	Output	: enable: 🛛 Internal 🔻	
I/O bit 0					
Ouput or	n Reset: 0 🔻	Fixed Config: 📝	I/O Type: Both 💌	Interrupt Type: Disabled	
I/O bit 1					
Ouput or	n Reset: 0 🔻	Fixed Config: 🔽	I/O Type: Both 🔻	Interrupt Type: Disabled	
I/O bit 2					
Ouput or	n Reset: 0 🔻	Fixed Config: 🔽	I/O Type: Both 💌	Interrupt Type: Disabled	
I/O bit 3					
Ouput or	n Reset: 0 🔻	Fixed Config: 📝	I/O Type: Both	Interrupt Type: Disabled	•
I/O bit 4					
Ouput or	Reset: 0 🔻	Fixed Config: 📝	I/O Type: Both 🔹	Interrupt Type: Disabled	-
Volation					_
I/O bit 5					
Ouput or	n Reset: 0 🔻	Fixed Config: 🔽	I/O Type: Both	Interrupt Type: Disabled	

2. Double-click **CoreGPIO** to configure it. Figure 21 shows **Configuring CoreGPIO_0** window.

Figure 21 • Configuring CoreGPIO_0

- 3. Under Global Configuration, configure the below settings:
 - Select APB Data Width as 32.
 - Select Number of I/O's as 8.
- Select Output enable as Internal.For all I/O bits from 0 to 7, configure I/O Type as Both.
 4. Click OK to save and close the Configuring CoreGPIO_0 window.
- CoreGPIO is configured with 8 outputs connected to LED's and with four inputs connected to DIP switches.



Instantiating CoreAHBLSRAM in PCIe_Demo_top SmartDesign

To instantiate CoreAHBLSRAM in the PCIe_Demo_top SmartDesign, expand the **Memory & Controllers** category in the Libero SoC **Catalog** as displayed in Figure 22

	🔍 🔻 📄 Simulation Mode 🛛	🏐 🛨
Name	Version	
Actel Macros		
Arithmetic	· · · · · · · · · · · · · · · · · · ·	
Bus Interfaces		
Clock & Management		
DSP		
⊳ I/O		
4 Memory & Controllers		
See CoreAHBLSRAM	2.0.113	
CoreAPBLSRAM	2.0.108	
CoreEDAC	2.4.103	
🗫 CoreFIFO	2.0.101	
🗫 CoreMemCtrl	2,0.105	
🗫 CoreSDR_AXI	2.0.116	
DDR Memory Controller	1.1.100	
Dual-Port Large SRAM	1.0.100	
Micro SRAM	1.0.100	
Two-Port Large SRAM	1.0.100	
Peripherals		

Figure 22 • IP Catalog

5

1. Drag **CoreAHBLSRAM** into the **PCIe_Demo_top** SmartDesign canvas. If the component appears shadowed in the **Vault**, right-click the name and select download.



2. Double-click COREAHBLSRAM_0 to configure it. Figure 23 shows **Configuring COREAHBLSRAM_0** window.

Configuring COREAHBLSRAM_0 (COREAHBLSRAM - 2.0.113)	
Configuration	
AHB Data Width: 32	
AHB Address Width: 32	
Select SRAM Type	
LSRAM Depth	
Number of bytes of memory: 8192	
uSRAM Depth	
Number of bytes of memory: 512	
Testbench:	
RTL Ø Obfuscated	
Help CK Cancel	

Figure 23 • Configuring COREAHBLSRAM_0

- Under Configuration, select AHB Data Width and AHB Address Width as 32.
- Under Select SRAM Type, click LSRAM.
- Under LSRAM Depth, enter the Number of bytes of memory as 8192.
- 3. Click OK to save and close the Configuring COREAHBLSRAM_0 window.



Instantiating Clock Conditioning Circuitry (CCC) in PCIe_Demo_top SmartDesign

CCC supplies the clock for components instantiated in the Fabric. To instantiate CCC in the PCIe_Demo_top SmartDesign, expand the **Clock & Management** category in the Libero SoC **Catalog**. Figure 24 shows Libero Catalog.

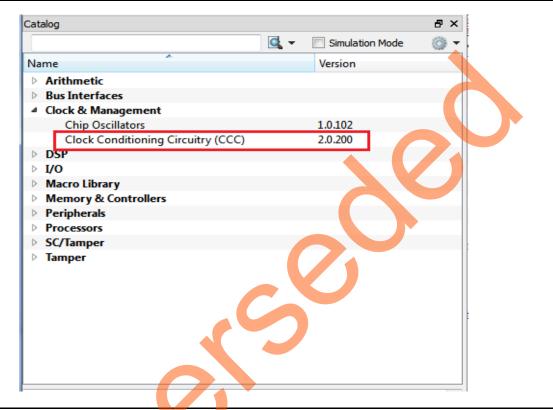


Figure 24 • IP Catalog

1. Drag CCC into the PCIe_Demo_top SmartDesign canvas. If the component appears shadowed in the **Vault**, right-click the name and select download.



2. Double-click **CCC** to configure it. Figure 25 shows the **FAB CCC Configurator** window.

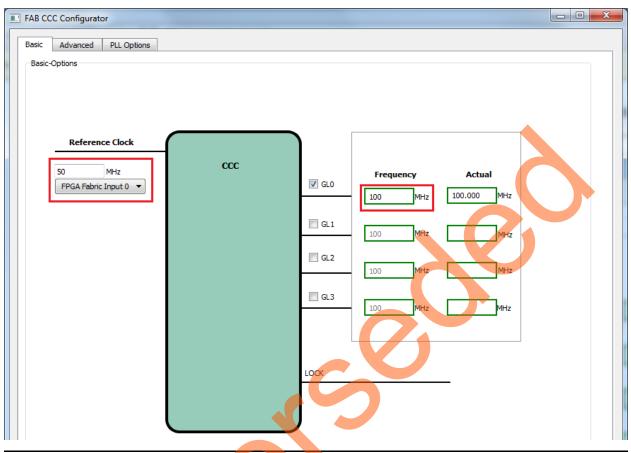
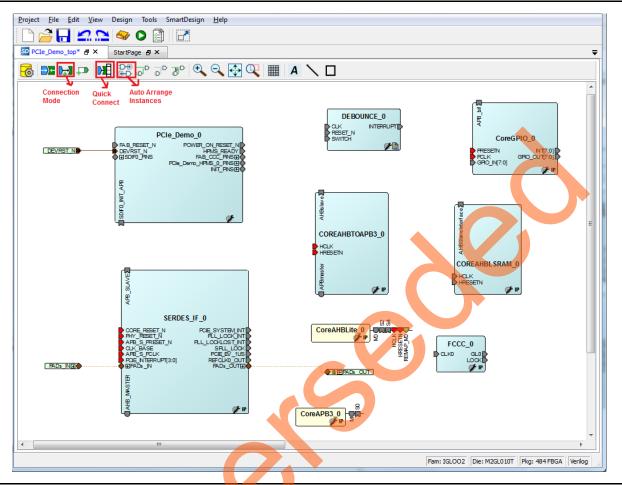


Figure 25 • Configure CCC

- Select Reference Clock as 50 MHz and FPGA Fabric Input 0 from the drop-down list.
- Select GL0 Frequency as 100 MHz.
- 3. Click **OK** to save and close the **FAB CCC Configurator** window.







4. Figure 26 shows PCIe_Demo_top in SmartDesign after configuring all components.

Figure 26 • PCIe_Demo_top in SmartDesign

Connecting Components in PCIe_Demo_top SmartDesign

There are three methods for connecting components in PCIe_Demo_top SmartDesign:

- The first method is by using the **Connection Mode** option. To use this method, change the SmartDesign to connection mode by clicking **Connection Mode** on the SmartDesign window, as shown in Figure 26. The cursor changes from the normal arrow shape to the connection mode icon shape. To make a connection in this mode, click on the first pin and drag-drop to the second pin that you want to connect.
- The second method is by selecting the pins to be connected together and selecting **Connect** from the context menu. To select multiple pins to be connected together, press down the **CTRL** key while selecting the pins. Right-click the input source signal and select **Connect** to connect all the signals together. Similarly, select the input source signal, right-click it, and select **Disconnect** to disconnect the signals already connected.
- The third method is by using the Quick Connect option. To use this method, change the SmartDesign to quick connect mode by clicking on Quick Connect mode on the SmartDesign window, as shown in Figure 26. Quick connect window will be opened.



Find the **Instance Pin** you want to connect and click to select it. In **Pins to Connect**, find the pin you wish to connect, right-click and choose **Connect** as shown in Figure 27.

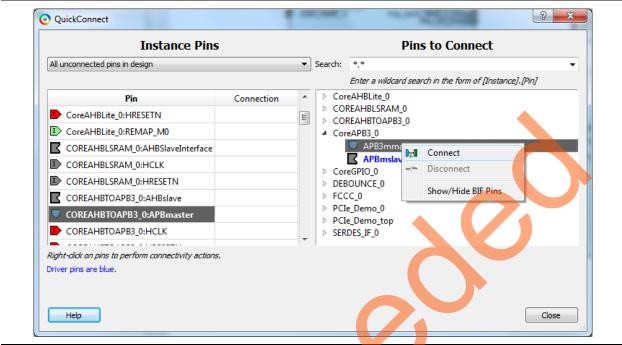


Figure 27 • Quick Connect Window

Use one of the three options described above and make the following connections:

1. Expand **SDIF0_PINS** of PCIe_Demo_0 and make connections as shown in Table 2.

Table 2 • SDIF0_PINS

From PCIe_Demo_0		To SERDES_IF_0
SDIF0_PHY_RESET_N		PHY_RESET_N
SDIF0_CORE_RESET_N		CORE_RESET_N
SDIF0_SPLL_LOCK		SPLL_LOCK

- 2. Right-click the **SDIF0_PERST_N** and **promote to top level**.
- 3. Expand INIT_PINS of PCIe_Demo_0 and make connections as shown in Table 3.

Table 3 • INIT_PINS

From PCIe_Demo_0	To SERDES_IF_0
INIT_APB_S_PCLK	APB_S_PCLK
INIT_APB_S_PRESET_N	APB_S_PRESET_N

- 4. Right-click the INIT_DONE and select mark unused.
- 5. Connect **HPMS_READY** of PCIe_Demo_0 to all resets as shown in Table 4.

Table 4 • HPMS_READY Connections

From PCIe_Demo_0	То
HPMS_READY	HRESETN of CoreAHBLite_0,COREAHBTOAPB3_0, and COREAHBLSRAM_0
	PRESETN of CoreGPIO_0
	RESET_N of DEBOUNCE_0

6. Connect **GL0** of FCCC_0 to all clocks as shown in Table 5.

Table 5 • GLU Clock Connections		
From FCCC_0	То	
GL0	HCLK of CoreAHBLite_0,COREAHBTOAPB3_0, and COREAHBLSRAM_0	
	PCLK of CoreGPIO_0	

Table 5 • GL0 Clock Connections

- 7. Expand FAB_CCC_PINS of PCIe_Demo_0:
 - Right-click the FAB_CCC_GL0 and select Mark Unused.

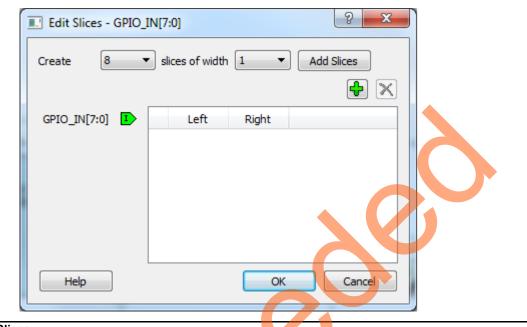
CLK BASE of SERDES IF 0

- Right-click the FAB_CCC_GL3 and select Mark Unused.
- 8. Right-click the **POWER_ON_RESET_N** of PCIe_Demo_0 and select Mark Unused.
- 9. Right-click the SDIF_READY of PCIe_Demo_0 and select Mark Unused.
- 10. Right-click the FAB_RESET_N of PCIe_Demo_0 and select Tie high.
- 11. Expand PCI_Demo_HPMS_0_PINS.
 - Right-click the COMM_BLK_INT of PCIe_Demo_0 and select Mark Unused.
 - Right-click the HPMS_INT_M2F[15:0] of PCIe_Demo_0 and select Mark Unused.
- 12. Connect SDIF0_INIT_APB of PCIe_Demo_0 and APB_SLAVE of SERDES_IF_0.
- 13. Connect Master port M0 of CoreAHBLite_0 to Master port AHB_MASTER of SERDES_IF_0
- 14. Connect Slave port S2 of CoreAHBLite_0 to Slave port AHBSIaveInterface of COREAHBLSRAM_0
- 15. Connect Slave port S4 of CoreAHBLite_0 to Slave port AHBslave of COREAHBTOAPB3_0
- 16. Connect Master port M of CoreAPB3_0 to Master port APBmaster of COREAHBTOAPB3_0
- 17. Connect Slave port S0 of CoreAPB3_0 to Slave port APB_bif of CoreGPIO_0
- 18. Right-click the CLK0 of FCCC_0 and select Promote to top level.
- 19. Right-click the LOCK of FCCC_0 and select Mark unused.
- 20. Right-click the SWITCH of DEBOUNCE_0 and select Promote to top level.
- 21. Right-click the INT[7:0] of CoreGPIO_0 and select Mark unused.
- 22. Right-click the GPIO_OUT[7:0] of CoreGPIO_0 and select Promote to top level.
- 23. This design uses 4 GPIO inputs GPIO_IN [3:0] of CoreGPIO_0 to connect DIP switches. To connect unused GPIO_IN[7:4] to logic '0' split the GPIO_IN[7:0] into two groups.





To do that, right-click the **GPIO_IN [7:0]** and select **Edit Slice**. Figure 28 displays the **Edit Slice** window.





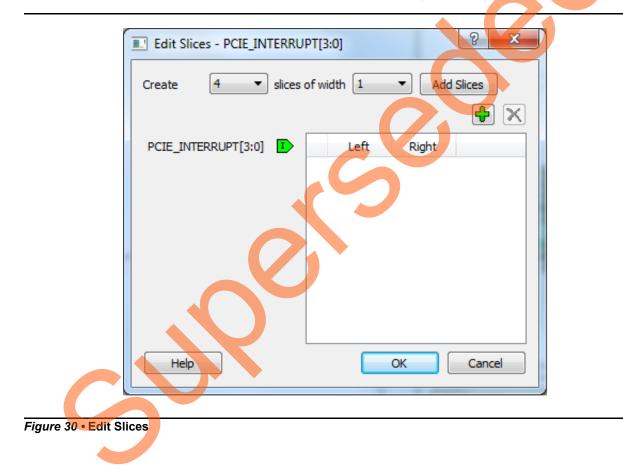
24. Select 2 slices of width 4, click Add Slices, and edit the window as shown in Figure 29.

ĺ	Edit Slices -	GPIO_IN[7	7:0]		? ×
	Create	2	slices of wid	th [4 🔹	Add Slices
					+ ×
	GPIO_IN[7:0]		Left	Right	
		1	3	0	
		2	7	4	
C					
	Help]		ОК	Cancel





- 25. Click **OK**.
- 26. Expand GPIO_IN [7:0], right-click the GPIO_IN [7:4] and select Tie low.
- 27. Right-click the GPIO_IN[3:0] and select Promote to top level.
- 28. Select the following ports of **SERDES_IF_0** by pressing down the **CTRL** key, right-click, and select **Mark Unused**.
 - PCIE_SYSTEM_INT
 - PLL_LOCK_INT
 - PLL_LOCKLOST_INT
 - PCIE_EV_1US
 - REFCLK0_OUT
- 29. The PCIe supports four interrupts. This design uses only one interrupt out of four by connecting the unused interrupts to logic '0'. To connect the unused interrupt pins to logic '0', split the interrupt pins to two groups. To do that, right-click the PCIE_INTERRUPT[3:0] of SERDES_IF_0 and select Edit Slice. The Edit Slice window is displayed as in Figure 30.





30. Click the + sign and create a slice with the Left index 0 and the Right index 0. Click + again to create a second slice with Left index 3 and Right index 1 as shown in Figure 31.

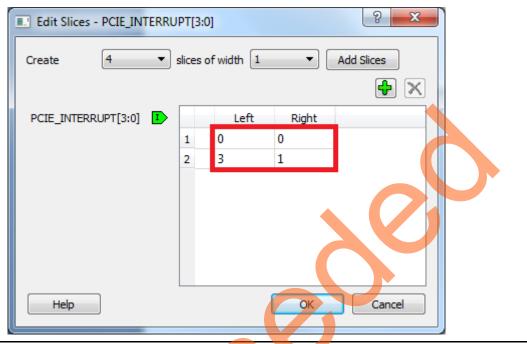
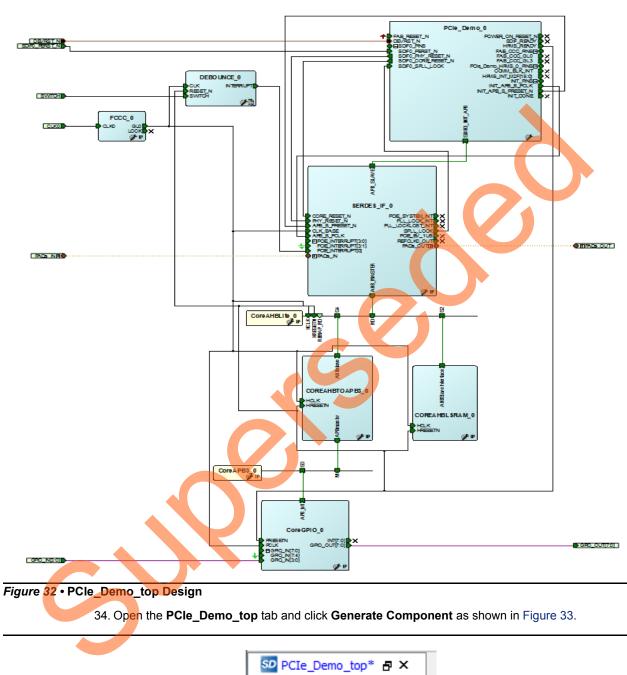
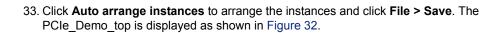


Figure 31 • Edit Slices

31. Expand PCIE_INTERRUPT[3:0], right-click the PCIE_INTERRUPT[3:1], and select Tie low. 32. Connect INTERRUPT of DEBOUNCE_0 to the PCIE_INTERRUPT[0] of SERDES_IF_0.







 PCIe_Demo_top*
 PCIe_Demo_top*

 PCIE_DEMO_top*
 PCIE_DEMO_

Figure 33 • Generate Component



35. The message "'PCIe_Demo_top' was generated" is displayed in the Libero SoC Log window if the design was generated without any errors. The Log window is displayed as shown in Figure 34 on successful component generation.

🔳 Messages 🛛 🔀 Errors 🔺 Warnings 🌒 Info	
Reading file refe_bemo_cop_rece_o_rece.v .	
Reading file 'PCIe_Demo_top_SERDES_IF_0_SERDES_IF.v'.	
Reading file 'PCIe_Demo_top_SERDES_IF_0_SERDES_IF_syn.v'.	
Reading file 'Debounce.v'.	
Reading file 'testbench.v'.	
The PCIE Demo project was opened.	
🕄 Info: 'PCIe Demo top' was successfully generated.	
Tinio: 'Pole Demo tob' was successfully deherated.	

Figure 34 • Log Window

36. The Core usage table is displayed in the Libero SoC **Cores** window. Figure 35 shows the **Cores** window that is displayed on successful component generation.

	Name	Vendor	Core Type	
	MSS	Actel	MegaCore	1.1.209
2	SERDES_IF	Actel	HW Core	1.2.102
3	COREAHBLSRAM	Actel	HW Core	2.0.113
4	FCCC	Actel	HW Core	2.0.200
5	COREAHBTOAPB3	Actel	HW Core	3.0.100
5	CoreGPIO	Actel	HW Core	3.0.120
7	CoreAPB3	Actel	HW Core	4.0.100
в	CoreAHBLite	Actel	HW Core	5.0.100

Figure 35 • Cores Window

Step 2: Developing the Simulation Stimulus

During the design process, SERDESIF was configured for the BFM simulation model. The BFM simulation model replaces the entire PCIe interface with a simple BFM that can send write transactions and read transactions over the AHBLite interface. These transactions are driven by a file and allow easy simulation of the FPGA design connected to a PCIe interface. This simulation methodology has the benefit of focusing on the FPGA design since the IGLOO2 PCIe interface is a fully hardened and verified interface. This section describes how to modify the BFM script (user.bfm) file that was generated by SmartDesign. The BFM script file simulates PCIe writing/reading to/from the Fabric CoreAHBLSRAM and CoreGPIO.



 To open the SERDESIF_0_user.bfm, go to the Files tab > Simulation folder, and double-click the SERDESIF 0 user.bfm. The SERDESIF 0 user.bfm file is displayed as shown in Figure 36.

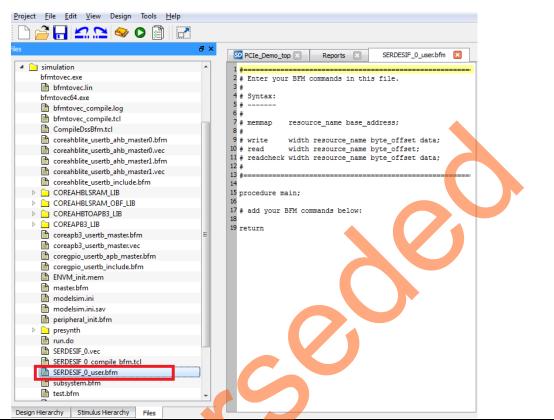


Figure 36 • SmartDesign Generated SERDESIF_0_user.bfm File

2. Modify the SERDESIF_0_user.bfm to add the following bfm commands of writing and reading:

۷.	wouny u			' _ '_'		to dud
	memmap (memmap]					
	meninap i	LSKAM	I UXZUU	100000	;	
	procedu					
	# add yo	our E	FM con	mands	below	
	wait 500)us;				
	wait 500)us;				
	write w	GPIC	0xA0	0x00;		
	write w	GPIC	0xA0	0x01;		
	write w	GPIC	0xA0	0x02;		
	write w	GPIC	0xA0	0x04;		
	write w	GPIC	0xA0	0x08;		
	write w	GPIC	0xA0	0x10;		
	write w	GPIC	0xA0	0x20;		
	write w	GPIC	0xA0	0x40;		
	write w	GPIC	0xA0	0x80;		
	write w	LSRA	M 0x0	ко 0	123456	78;
	write w	LSRA	M 0x0	4 0x	8765432	21;
	write w	LSRA	M 0x0	x0 8	9ABCDEI	F0;
	write w	LSRA	M 0x0	ОС Ох	OFEDCBA	A9;
	readcheo	ck w	LSRAM	0x00	0x12345	5678;
	readcheo	ck w	LSRAM	0x04	0x8765	54321;
	readcheo	ck w	LSRAM	80x0	0x9ABCI	DEF0;
	readcheo	ck w	LSRAM	0x0C	0x0FEI	CBA9;
	return					



BFM commands added in the SERDESIF_0_user.bfm do the following:

- Perform write to GPIO_OUT[7:0]
- Perform write to LSRAM
- Perform read-check from LSRAM
- 3. The modified BFM file appears similar to the file shown in Figure 37.

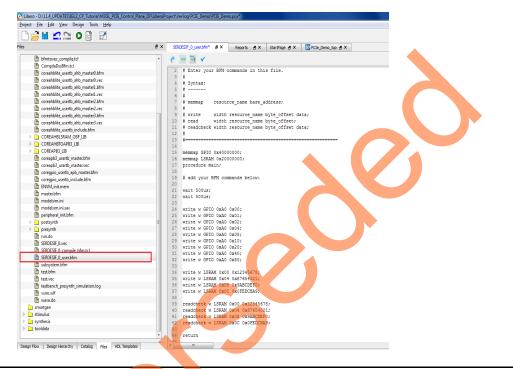


Figure 37 • Modified SERDES User BFM

Step 3: Simulating the Design

The design supports the BFM_PCIe simulation level to communicate with the High Speed Serial Interface block through the master AXI bus interface. Although no serial communication actually goes through the High Speed Serial Interface block, this scenario allows validating the fabric interface connections. The SERDESIF_0_user.bfm file under the *<Libero project>/simulation* folder contains the BFM commands to verify the read/write access to CoreGPIO and CoreAHBLSRAM. This section describes how to use the SmartDesign testbench and the BFM script file to simulate the design.

- Add the wave.do file to the PCIe demo design simulation folder by clicking File > Import > Others.
- Browse to the wave.do file location in the design files folder: M2GL_PCIE_Control_Plane_DF\Source Files. Figure 38 shows the wave.do file under simulation folder in the Files window.

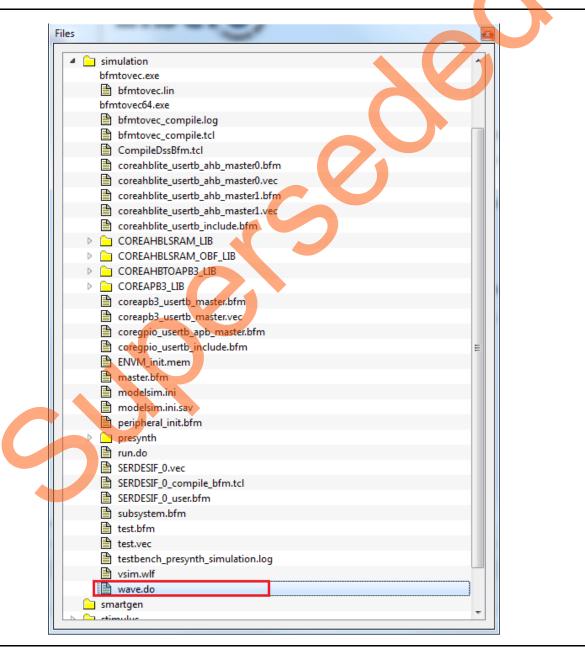


Figure 38 • Wave.do file



- 3. Open the Libero SoC project settings (Project > Project Settings).
- 4. Select **Do File** under **Simulation Options** in the **Project Settings** window. Change the **Simulation runtime** to **280us**, as shown in Figure 39.

Project Settings			? ×
Device Device I/O Settings	Use automatic DO file		Save
Preferred HDL Type	Simulation runtime:	205us ·	Restore Defaults
Design Flow Simulation Options	Testbench module name:	testbench	Incource bendunds
DO File	Top level instance name:	<top>_0</top>	
Waveforms Vsim commands	Generate VCD file		
Vsim commands A Simulation Libraries	VCD file name:	power.vcd	
IGLOO2		Select Verilog Language Syntax	
COREAHBLITE_LIB COREAHBLSRAM	Verilog 2001		
COREAHBTOAPB	System Verilog		
COREAPB3_LIB		Select VHDL Language Syntax	
	VHDL 2008		
	User defined DO file:]
	DO command parameters:		
Help			Close

Figure 39 • Project Setting – Do File Simulation Runtime Setting

5. Click Save.

S



6. Select Waveforms under Simulation Options as shown in Figure 40.

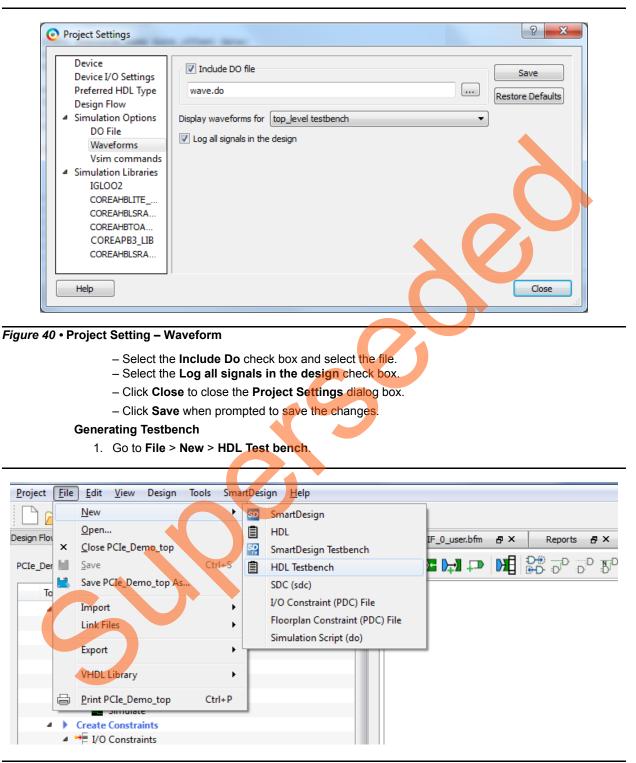


Figure 41 • HDL Testbench



2. Select HDL type as Verilog/VHDL and Give testbench as the Name.

Create New HDL Testbench File	x
HDL Type Verilog VHDL	
Name:	
testbench	
Initialize file with standard template	
Instantiate Root Design	
Set as Active Stimulus	
Help OK Car	icel

Figure 42 • Selecting HDL Type as Verilog/VHDL in HDL Testbench

3. Click OK.

To run the simulation, double-click **Simulate** under **Verify Pre-Synthesized Design** in the **Design Flow** window. ModelSim runs the design for about **205us**. The ModelSim transcript window displays the BFM commands and the BFM simulation completed with no errors, as shown in Figure 43.

Transcript	x
e Edit View Bookmarks Window Help	
Transcript	
▋-☞₩%● ጷ▣®≥⊇ ⊘-₩≝	
Time: 290850010.0ps! Instance: testbench.PCIe_Demo_top_0.CORDAHBLSRAM_0.CHTOLSRAM01.genblk1.CHTOLSRAM101.CHTOLSRAM101	
** Warning: Port A Write to and Port B Read from the same address at the same time. Read data from conflicting address is unkn	.own
Time: 290850010.0ps! Instance: testbench.FCIe_Demo_top 0.COREAHBLSRAM_0.CHTOLSRAMO1.genblk1.CHTOLSRAM001.CHTOLSRAM101	
BFM: Data Write 20000004 87654321 BFM:35:readcheck w 20000000 12345678 at 291250 ng	
series readeneeds w 20000000 12943676 at 291250 ms ** Warning: Port A Write to and Port B Read from the same address at the same time. Read data from conflicting address is unkn	OWD.
Time: 291250010.0ps! Instance: testbench.PCIe Demo to 0. CORFABLSRAM 0. CHTOLSRAM0.genblk1. CHTOLSRAMIO1. CHTOLSRAMIO1	
** Warning: Port A Write to and Port B Read from the same address at the same time. Read data from conflicting address is unkn	own
Time: 291250010.0ps! Instance: testbench.PCIe_Demo_top_0.COREAHBLSRAM_0.CHTOLSRAM01.genblk1.CHTOLSRAMI01.CHTOLSRAM101	
** Warning: Port A Write to and Port B Read from the same address at the same time. Read data from conflicting address is unkn	own
Time: 291250010.0ps! Instance: testbench.PCIe_Demo_top_0.COREAHBLSRAM_0.CHTOLSRAM01.genblk1.CHTOLSRAM101.CHTOLSRAM101	
** Warning: Port A Write to and Port B Read from the same address at the same time. Read data from conflicting address is unkn	own
Time: 291250010.0ps! Instance: testbench.PCIe_Demo_top_0.COREAHBLSRAM_0.CHTOLSRAMO1.genblk1.CHTOLSRAMo01.CHTOLSRAM101 SFM: Data Write 20000008 9abcief0	
ser, bata wire zooodoos sanchero ** Warning: Port A Write to and Rort B Read from the same address at the same time. Read data from conflicting address is unkn	OWD
Time: 291650010.0ps/ Instance: testbench.PCLE Demo top 0.COREAHBLSRAM 0.CHTOLSRAM01.embltl.CHTOLSRAMICHTOLSRAMI	
** Warning: Fort A Write to and Port B Read from the same address at the same time. Read data from conflicting address is unkn	own
Time: 291650010.0ps! Instance: testberch.PCIe_Demo_top_0.COREAHBLSRAM_0.CHTOLSRAMO1.genblk1.CHTOLSRAMIO1.CHTOLSRAMI01	
** Warning: Port A Write to and Port B Read from the same address at the same time. Read data from conflicting address is unkn	own
Time: 291650010.0ps! Instance: testbench.PCIe_Demo_top_0.COREAHBLSRAM_0.CHTOLSRAM01.genblk1.CHTOLSRAM101.CHTOLSRAM101	
** Warning: Port A Write to and Port B Read from the same address at the same time. Read data from conflicting address is unkn	own
Time: 291650010.0ps! Instance: testbench.PCIe_Demo_top_0.COREAHBLSRAM_0.CHTOLSRAMO1.genblk1.CHTOLSRAMo01.CHTOLSRAMi0i BFM: Data Write 2000000c Ofedcba9	
SFM:36 readcheck w 20000006 STEELEBAS	
STM: Data Read 2000000 12345678 MASK:fffffff at 292350.010000ns	
SEM: Data Read 20000004 87654321 at 292750.010000ng	
BFM:37:readcheck w 20000008 9abcdef0 at 292850 ns	
BFM: Data Read 20000004 87654321 MASK:ffffffff at 293150.010000ns	
SFM: Data Read 20000008 9abcdef0 at 293550.010000ns	
BFM:38:readcheck w 2000000c Ofedcba9 at 293650 ns	
SFM: Data Read 20000008 9abcdef0 MASK:fffffff at 293950.010000ns	
SFM: Data Read 2000000c Ofedcba9 at 294350.010000ns SFM:39:return	
SFM: Jata Read 2000000c Ofedoba9 MASK:fffffff at 294750.010000ms	
STM: Data Read 20000010 0xxxxxxx at 295150.010000ns	
SERDESIF_0 BFM Simulation Complete - 20 Instructions - NO ERRORS	
	-

Figure 43 • SERDES BFM Simulation



Figure 44 shows the waveform window with GPIO_OUT signals.

I+	Msgs												
/testbench/SYSCLK	0												
/testbench/NSYSRESET	1												
/testbench/PCIe_Demo_top_0/CoreGPIO_0/GPIO_OUT	00	00		01	02	04)08	10	20	40	30		
	0												
	0												
	ŏ								_				
-4 3	0												
-4 [2]	0												
	0												
	<u> </u>												

Step 4: Generating the Program File

1. Double-click **I/O Constraints** in the **Design Flow** window as shown in Figure 45. The **I/O Editor** window is displayed after completing **Synthesize and Compile**.

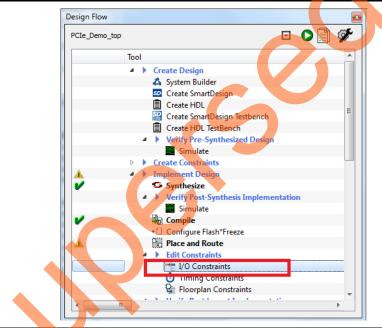


Figure 45 • I/O Constraints

2. In the I/O Editor window, make the pin assignments as shown in Table 6.

Table 6 • Port to Pin Mapping

Port Name	Pin Number
CLK0	К1
GPIO_IN[0]	L19
GPIO_IN[1]	L18
GPIO_IN[2]	K21
GPIO_IN[3]	K20
GPIO_OUT[0]	E1



Table 6 • Port to Pin Mapping (continued)

Port Name	Pin Number
GPIO_OUT[1]	F4
GPIO_OUT[2]	F3
GPIO_OUT[3]	G7
GPIO_OUT[4]	H7
GPIO_OUT[5]	J6
GPIO_OUT[6]	H6
GPIO_OUT[7]	H5
SDIF0_PERST_N	P18
SWITCH	J18

After the pins have been assigned, the I/O Editor is displayed as shown in Figure 46.



ile	<u>E</u> dit <u>V</u> iew <u>T</u> ools	; <u>H</u> elp						
		🤹 🕷 👫	12					
P	Commit and Che	eck Pins Packag	ge Viewer					
	Port Name 🕇	Direction 💌	I/O Standard 💌	Pin Number 💌	Locked 💌	Macro Cell 💌	Bank Name 💌 🖌	-
1	CLK0	Input	LVCMOS25	K1	V	ADLIB:INBUF	Bank6	
2	DEVRST_N	Input		R15	V	ADLIB:SYSRESET		
3	GPIO_IN[0]	Input	LVCMOS25	L19	V	ADLIB:INBUF	Bank2	
4	GPIO_IN[1]	Input	LVCMOS25	L18	V	ADLIB:INBUF	Bank2	
5	GPIO_IN[2]	Input	LVCMOS25	K21	V	ADLIB:INBUF	Bank2	
6	GPIO_IN[3]	Input	LVCMOS25	K20	V	ADLIB:INBUF	Bank2	
7	GPIO_OUT[0]	Output	LVCMOS25	E1	V	ADLIB:OUTBUF	Bank7	
8	GPIO_OUT[1]	Output	LVCMOS25	F4	V	ADLIB:OUTBUF	Bank7	
9	GPIO_OUT[2]	Output	LVCMOS25	F3	V	ADLIB:OUTBUF	Bank7	
10	GPIO_OUT[3]	Output	LVCMOS25	G7	V	ADLIB:OUTBUF	Bank7	
11	GPIO_OUT[4]	Output	LVCMOS25	H7	V	ADLIB:OUTBUF	Bank7	
12	GPIO_OUT[5]	Output	LVCMOS25	J6	V	ADLIB:OUTBUF	Bank7	
13	GPIO_OUT[6]	Output	LVCMOS25	H6		ADLIB:OUTBUF	Bank7	
14	GPIO_OUT[7]	Output	LVCMOS25	H5		ADLIB:OUTBUF	Bank7	
15	(P) REFCLK0_P	Input	LVDS	U1		ADLIB:INBUF_DIFF	Bank5	
16	(N) REFCLK0_N	Input	LVDS	T1		ADLIB:INBUF_DIFF	Bank5	
17	RXD0_N	Input		Y1		ADLIB:SERDESIF_0		
18	RXD0_P	Input		W1		ADLIB:SERDESIF_0		
19	RXD1_N	Input		Y3		ADLIB:SERDESIF_0		
20	RXD1_P	Input		W3	V	ADLIB:SERDESIF_0		
21	RXD2_N	Input		Y5		ADLIB:SERDESIF_0		
22	RXD2_P	Input	-	W5		ADLIB:SERDESIF_0		
23	RXD3_N	Input	-	Y7		ADLIB:SERDESIF_0		
24	RXD3_P	Input		W7	V	ADLIB:SERDESIF_0		
25	SDIF0_PERST_N	Input	LVCMOS25	P18	V	ADLIB:INBUF	Bank2	
26	SWITCH	Input	LVCMOS25	J18	V	ADLIB:INBUF	Bank2	

Figure 46 • I/O Editor



These pin assignments are for connecting the following components on the IGLOO2 Evaluation Kit:

- CLK to 50 MHz Clock Oscillator
- GPIO_OUT [0] to GPIO_OUT [7] for LEDs
- GPIO_IN [0] to GPIO_IN [3] for DIP switches
- SWITCH for SW4
- SDIF0_PERST_N is reset signal from PCIe edge connector
- 3. After updating I/O editor, click **Commit and Check**.
- 4. Close the I/O editor.
- 5. Click **Generate Bitstream** as shown in Figure 47 to complete place and route, verify timing, and generate the programming file.



Figure 47 • Generate Programming Data

Step 5: Programming the IGLOO2 Board Using FlashPro

- 1. Connect the FlashPro4 programmer to the J5 connector of the IGLOO2 FPGA Evaluation Kit.
- Connect the jumpers on the IGLOO2 FPGA Evaluation Kit as shown in Table 7.
 CAUTION: While making the jumper connections, the power supply switch SW7 on the board should be in OFF position.

Jumper	Pin (from)	Pin (to)	Comments
J22	1	2	Default
J23	1	2	Default
J24	1	2	Default
J8	1	2	Default
J3	1	2	Default

- 3. Connect the power supply to the J6 connector.
- Switch the power supply switch SW7 to ON position. Refer to the "IGLOO2 Evaluation Kit Board" section for further details.
- 5. To program the IGLOO2 device, double-click **Run Programming Action** in the **Design Flow** window as shown in Figure 48.



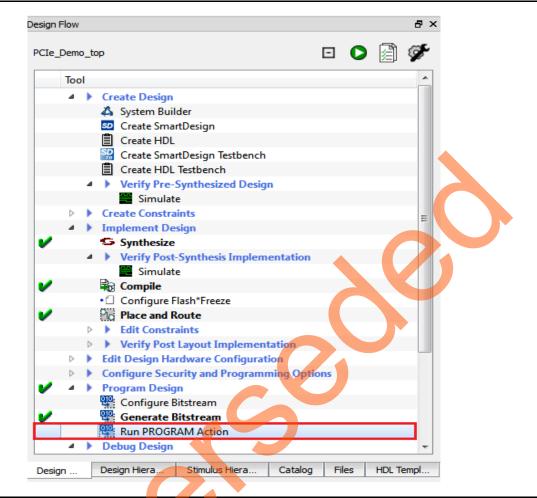


Figure 48 • Run Programming Action



Step 6: Connecting the Evaluation Kit to the Host PC

- 1. After successful programming, power off the IGLOO2 Evaluation Kit and shut down the host PC.
- Use the below steps to connect the CON1–PCIe Edge Connector either to host PC or laptop, a.Connect the CON1–PCIe Edge Connector to host PC's PCIe Gen2 slot or Gen1 slot as applicable. This tutorial is designed to run in any PCIe Gen2 compliant slot. If your host PC does not support the Gen2 compliant slot, the design switches to the Gen1 mode.
 - b.Connect the CON1–PCIe Edge Connector to the laptop PCIe slot using the express card adapter. If you are using a laptop, the express card adapters typically support only Gen1 and the design works on Gen1 mode.
- Note: Host PC or laptop should be powered OFF while inserting the PCIe Edge Connector. If you do not power off the system, the PCIe device detection and selection of Gen1 or Gen2 do not occur properly. It is recommended that the host PC or laptop should be powered off during the PCIe card insertion.
 - 3. Figure 49 shows the board setup for the host PC in which IGLOO2 Evaluation Kit is connected to the host PC PCIe slot. To connect the IGLOO2 Evaluation Kit to the Laptop using Express card adapter, refer to the "IGLOO2 Evaluation Kit Board Setup for Laptop" section.



Figure 49 • IGLOO2 Evaluation Kit Setup for Host PC

Step 7: Running the Design

This design can be run on both Windows and RedHat Linux OS.

- To run the design on Windows OS GUI, Jungo drivers are provided. Refer to "Running the Design on Windows" section on page 49.
- To run the design on Linux OS, native RedHat Linux drivers and command line scripts are provided. Refer to "Running the Design on Linux" section on page 59.



Running the Design on Windows

- 1. Switch **ON** the **SW7** power supply switch.
- 2. Power on the host PC and check the host PC Device Manager for PCIe device. It will be similar to Figure 50. If the PCIe device is not detected, power cycle the IGLOO2 Evaluation Kit board and click "scan for hardware changes" in the Device Manager.

<u>File</u> <u>A</u> ction	n <u>V</u> iew <u>H</u> elp		
🧼 🔿 📧	1 🛐 🖬 🔯		
⊿ 🚑 w7-da	onthus		
⊳ n∰ Co	omputer		
⊳ 👝 Di	sk drives		
🔋 📐 Di	splay adapters		
D۱ 🚑 ک	/D/CD-ROM drives		
> 🕼 H	uman Interface Devices		
D 📻 ID	E ATA/ATAPI controllers		
⊳	eyboards		
⊳ 🖺 M	ice and other pointing devic	es	
🛛 🕞 🔟 M	onitors		
⊳ 👰 N	etwork adapters		
a 🔂 🛛	ther devices		
	PCI Device		
	orts (COM & LPT)		
🛛 📩 🛄 Pr	ocessors		
	ound, video and game contro	ollers	
	stem devices		
	niversal Serial Bus controllers		

Figure 50 • Device Manager

- Note: If the device is still not detected, check whether or not the BIOS version in host PC is latest, and if PCI is enabled in the host PC BIOS.
 - 3. If the host PC has any other installed drivers (previous versions of Jungo drivers) for the IGLOO2 PCIe device, uninstall them. To uninstall previous versions of Jungo drivers follow step a and b.



a. To uninstall the previous Jungo drivers, go to device manager and right-click on DEVICE as shown in Figure 51. The DEVICE uninstall window is displayed.

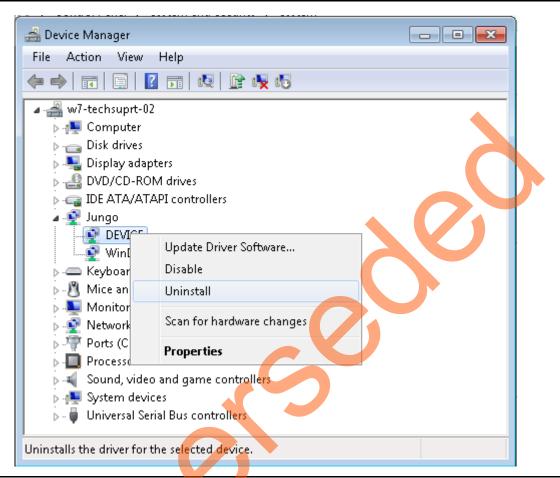


Figure 51 • Device Uninstall

b. Select the **Delete the driver software** check box for this device as shown in Figure 52. After uninstalling the previous Jungo drivers, make sure that the PCI Device is detected in the **Device** Manager window as shown in Figure 50.

Confirm Device Uninstall
DEVICE
Warning: You are about to uninstall this device from your system.
Delete the driver software for this device.
OK Cancel

Figure 52 • Confirm Device Uninstall



Installing Jungo Drivers

The PCIe tutorial uses a driver framework provided by Jungo WinDriver Pro. To install the PCIe drivers on the host PC for IGLOO2 Evaluation Kit board, use the following steps:

- Extract the PCle_Demo.rar to the C:\ drive. The PCle_Demo.rar is located in the provided design files: M2GL_PCIE_Control_Plane_DF\Windows_64bit\Drivers\PCle_Demo.rar.
- Note: Installing these drivers require host PC Administration rights.
 - 2. Run the batch file C:\PCIe_Demo\DriverInstall\Jungo_KP_install.bat.
 - 3. Click **Install** if the window is displayed as shown in Figure 53.

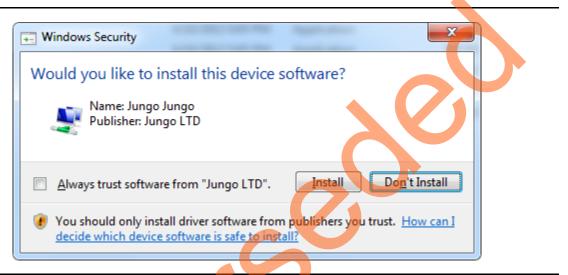


Figure 53 • Jungo Driver Installation

- Note: If the installation is not in progress, right-click on the command prompt and select **Run as** administrator. Run the batch file C:\PCIe_Demo\DriverInstall\Jungo_KP_install.bat from command prompt.
 - 4. Click Install this driver software anyway if the window appears as shown in Figure 54.

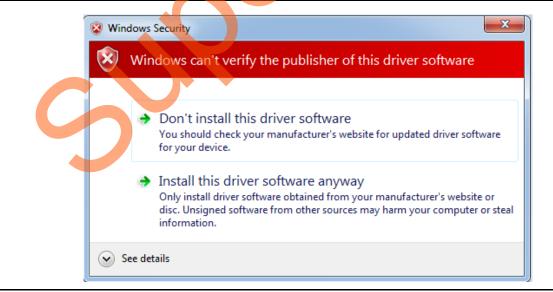


Figure 54 • Windows Security



Installing the PCIe GUI

The IGLOO2 PCIe graphic user interface (GUI) is a simple GUI that runs on the host PC to communicate with the IGLOO2 PCIe EP device. The GUI provides the PCIe link status, driver information, and demo controls. The GUI invokes the PCIe driver installed on the host PC and provides commands to the driver according to your selection. Use the following steps to install the GUI:

- 1. Extract the PCIe_Demo_GUI_Installer.rar from the provided design files: M2GL_PCIE_Control_Plane_DF\Windows_64bit\GUI.
- Double-click the setup.exe in the provided GUI installation (PCIe_Demo_GUI_Installer\setup.exe). Apply default options as shown in Figure 55.

Destination Directory Select the primary installation dire	ctory.	
	illowing locations. To install software into a button and select another directory.	~
Directory for PCIe Demo		
C:\Program Files\PCle Demo\	Browse.	
Directory for National Instruments	products	
C:\Program Files\National Instrum	nents\ Browse	

Figure 55 • GUI Installation

3. Click **Next** to complete the installation. The Installation Complete window is displayed.

PCIe Demo	1.10			
The installer has finished updating y	our system.			
		<< <u>B</u> ack	Next >>	Finish

Figure 56 • Successful GUI Installation

4. Restart the host PC.



Running the PCIe GUI

 Check the host PC Device Manager for the drivers. Make sure that the board is switched on. If the device is not detected, power cycle the IGLOO2 Evaluation Kit board and click "scan for hardware changes" in the Device Manager.

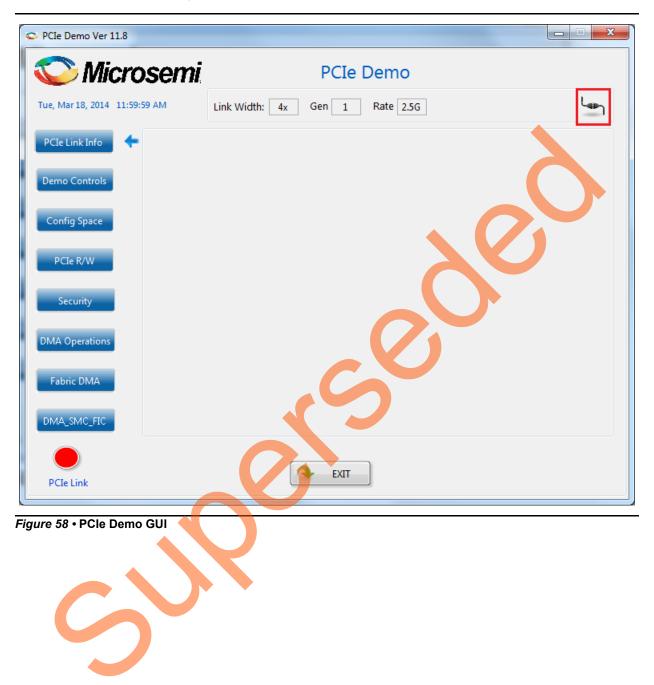
	Device Manager		
	<u>File Action View H</u> elp		
<	Þ 🔿 🖬 🛛 🗗 😡		
Г	a 📇 w7-donthus		
	Computer		
	Disk drives		
	🖻 🖳 Display adapters		
	DVD/CD-ROM drives		
L	👂 🕼 Human Interface Devices		
	IDE ATA/ATAPI controller	s	
L	🛯 📲 Jungo		
	DEVICE	\wedge	
	WinDriver		
	Keyboards		
	Mice and other pointing of the second sec	devices	
	Monitors		
	Network adapters		
L	Ports (COM & LPT) Processors		
		ontrollors	
	Sound, video and game c System devices	onuolleis	
L	Universal Serial Bus contro	ollers	
	• • • • • • • • • • • • • • • • • • •	Jileis	

Figure 57 • Device Manager - PCIe Device Detection

Note: If a warning symbol is displayed on the **DEVICE** or **WinDriver** icons in the **Device Manager**, uninstall them and start from Step1 of the "Step 7: Running the Design" section.



2. Invoke the GUI from **ALL Programs > PCIe Demo > PCIe Demo GUI**. The GUI is displayed as shown in Figure 58.





3. Click the **Connect** button at the top-right corner of the GUI. The messages are displayed on the GUI as shown in Figure 59.

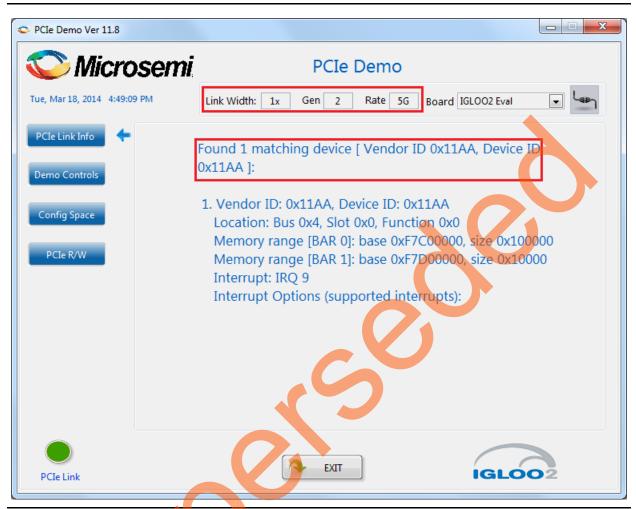


Figure 59 • Version Information

Note: If the host PC does not support GEN2 slot, then this design will run at GEN1 speed.



4. Clicking **Demo Controls** in the GUI displays the LED options and DIP switch positions as shown in Figure 60.

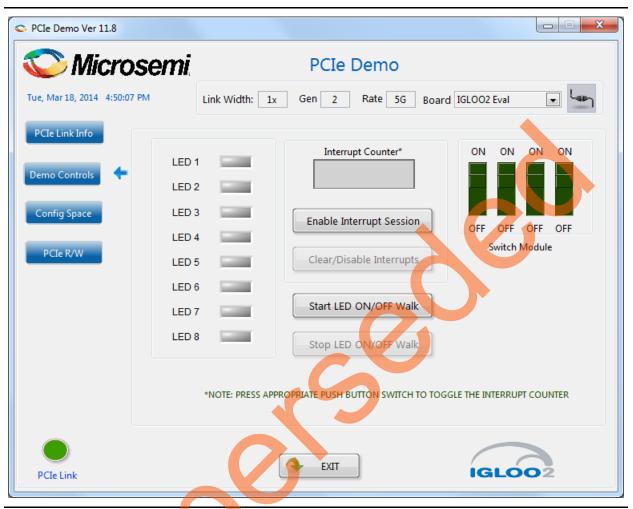


Figure 60 • Demo Controls

- 5. Click LEDs in GUI to ON/OFF the LEDs on the IGLOO2 Evaluation Kit.
- 6. Click Start LED ON/OFF Walk to make the LEDs on IGLOO2 Evaluation Kit blink.
- 7. Click Stop LED ON/OFF Walk to stop the LEDs blinking.
- 8. Change the DIP switch positions on the IGLOO2 Evaluation Kit (**SW5**) and observe the similar position of switches in the **GUI SWITCH MODULE**.
- 9. Click Enable Interrupt Session to enable the PCIe interrupt.



10. Press the push button **SW4** on the IGLOO2 Evaluation Kit and observe the interrupt count on the **Interrupt Counter** field in the GUI as shown in Figure 61.

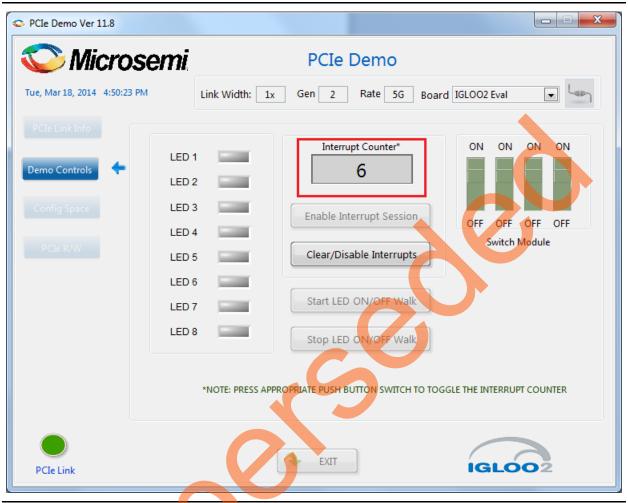


Figure 61 • Interrupt Counter

11. Click **Clear/Disable Interrupts** to clear and disable the PCIe interrupts.





12. Click **Config Space** to read details about the PCIe configuration space. Figure 62 shows the PCIe configuration space.



Figure 62 • Configuration Space



13. Click **PCIe R/W** to perform read and writes to LSRAM memory through **BAR1** space. shows the PCIe R/W window. Enter the address in the **Address** field between **0x0000** to **0x7FFC**. The **Data** field accepts a 32-bit hexadecimal value.

© PCIe Demo Ver 11.8	
🛇 Microsemi	PCIe Demo
Tue, Mar 18, 2014 4:52:54 PM	Link Width: 1x Gen 2 Rate 5G Board IGLOO2 Eval
PCIe Link Info Demo Controls	BAR 1 Memory Range
Config Space	Address 200
PCIe R/W	Data Data Read Write
PCIe Link	EXIT IGLOO2

Figure 63 • Perform Read and Write to LSRAM Using PCIe

14. Click Exit.

Running the Design on Linux

- 1. Switch **ON** the power supply switch on the IGLOO2 Evaluation Kit board.
- 2. Switch **ON** the Red Hat Linux host PC.
- 3. Red Hat Linux Kernel detects the IGLOO2 PCIe end point as Actel Device.
- 4. On Linux Command Prompt Use lspci command to display the PCIe info.
 - # lspci



	root@localhost:~	
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>T</u> erminal Ta <u>b</u> s <u>H</u> elp		
0:02.0 VGA compatible controller: Intel 0:02.1 Display controller: Intel Corpora 0:1b.0 Audio device: Intel Corporation N 0:1c.0 PCI bridge: Intel Corporation NI 0:1d.0 USB controller: Intel Corporation 0:1d.1 USB controller: Intel Corporation 0:1d.3 USB controller: Intel Corporation 0:1d.3 USB controller: Intel Corporation 0:1e.0 PCI bridge: Intel Corporation 826 0:1e.0 PGI bridge: Intel Corporation 826 0:1f.1 DE interface: Intel Corporation 0:1f.2 UE interface: Intel Corporation 0:1f.3 SMBus: Intel Corporation ND/IGH 0:1f.3 SMBus: Intel Corporation ND/IGH	Series Chipset PCI Express Root Port (rev 03) Corporation 4 Series Chipset Integrated Graphics Controller (rev 03) 410/ICH 7 Family High Definition Audio Controller (rev 01) 9/ICH 7 Family PCI Express Port 1 (rev 01) 1 N10/ICH 7 Family USB UHCI Controller #1 (rev 01) 1 N10/ICH 7 Family USB UHCI Controller #2 (rev 01) 1 N10/ICH 7 Family USB UHCI Controller #3 (rev 01) 1 N10/ICH 7 Family USB UHCI Controller #3 (rev 01) 1 N10/ICH 7 Family USB UHCI Controller #4 (rev 01) 1 N10/ICH 7 Family USB UHCI Controller #4 (rev 01) 1 N10/ICH 7 Family USB UHCI Controller #4 (rev 01) 301 PCI Bridge (rev 01) 30160/GR (ICH7 Family) LPC Interface Bridge (rev 01) 828016 (ICH7 Family) IDE Controller (rev 01) N10/ICH7 Family SATA Controller (IDE mode] (rev 01) 7 Family SMUS CONTOLER (IDE mode] (rev 01)	03)
🕸 🔲 root@localhost:~ 🛛 🛍 Starting	a Take Screenshot	

Installation

Enter the following commands in the Linux command prompt to install the PCIe drivers:

- 1. Create the igl2 directory under the home/ directory using the following command:
 - # mkdir /home/igl2
- 2. Copy the *M2GL_PCIE_Control_Plane_DF*\Linux_64bit\Drivers\PCIe_Driver folder from the Windows host PC and place it into the /home/igl2 directory of RedHat Linux host PC.
- Copy the M2GL_PCIE_Control_Plane_DF\Linux_64bit\Drivers\inc folder from the Windows host PC and place it into the /home/igl2 directory of RedHat Linux host PC. The /home/igl2 directory must contain PCIe Driver/ inc/ folders.
- 4. Execute Is command to display the contents of /home/igl2 directory.
 - # 1s
- Change to inc/ directory by using the following command: #cd /home/igl2/inc
- 6. Edit the board h file for IGLOO2 Evaluation Kit.

#vi board.h
#define IGL2

#undef SF2

	root@localhost:/home/igl2/inc	
<u>File E</u> dit <u>V</u> iew <u>T</u> erminal Ta <u>b</u> s <u>H</u> elp		
** SF2 : SmartFusion2 Board, IGL2: IGL002 #define SF2, if the hardware board is : #define IGL2, if the hardware board is / define IGL2 undef SF2	imartFusion2	
va		



- 7. To save the selected file, execute the :wq command. command
- 8. Change the PCIe Driver/directory using cd command:

```
#cd /home/igl2/PCIe_Driver
```



9. To compile the Linux PCIe device driver code, execute the make command on Linux Command Prompt.

```
#make clean [To clean any *.o, *.ko files]
#make
```

- 10. The kernel module, pci chr drv ctrlpln.ko, is created in the same directory.
- 11. To insert the Linux PCIe device driver as a module, execute insmod command on Linux Command Prompt.

```
#insmod pci_chr_drv_ctrlpln.ko
```

Note: Root privileges are required to execute <code>insmod</code> command.

Applications Places System 🔗 🏽 🏵 🗌 🐨	root@localhost:/home/igl2/PCIe_Driver	
File Edit View Terminal Tabs Help	, oorgeorgeneoselinomerigizit ete_ortet	
<pre>root@localhost PCIe_Driver]# pwd home/igl2/PCIe_Driver root@localhost PCIe_Driver]# ls lakefile pci_chr_drv_ctrlpln.c root@localhost PCIe_Driver]# make nake - C /lbb/modules/2.6.18.308.el5/build SUBD nake[1]: Entering directory /vsr/src/kernels/2 CC [M] /home/igl2/PCIe_Driver/pci_chr_drv_ct Building modules, stage 2. MODPOST CC /home/igl2/PCIe_Driver/pci_chr_drv_ct Building modules, stage 2. MODPOST CC /home/igl2/PCIe_Driver/pci_chr_drv_ct LD [M] /home/igl2/PCIe_Driver/pci_chr_drv_ct Lake[1]: Leaving directory 'vsr/src/kernels/2. root@localhost PCIe_Driver]# ls lakefile Module.symvers pci_chr_ root@localhost PCIe_Driver]# ls /dev/MS_PCI_DE 'dev/MS PCI_DE' root@localhost PCIe_Driver]# []</pre>	.6.18-308.el5-x86_64' rlpln.mod.o rlpln.ko 5.18-308.el5-x86_64' drv_ctrlpln.ko pci_chr_drv_ctrlpln.mod.o drv_ctrlpln.mod.c pci_chr_drv_ctrlpln.o v_ctrlpln.ko	
root@localhost:/home/igl2/PCle Driver		(real)

Figure 66 • PCIe Device Driver Installation

12. After successful Linux PCIe device driver installation, check /dev/MS_PCI_DEV got created by using the following command:

#ls/dev/MS_PCI_DEV

Note: /dev/MS_PCI_DEV interface is used to access the IGLOO2 PCIe end point from Linux user space.

Linux PCIe Application Compilation and PCIe Control Plane Utility Creation

- 1. Change to the /home/igl2/ directory using the following command:
 - # cd /home/igl2
- Copy the M2GL_PCIE_Control_Plane_DF\Linux_64bit\Util\PCIe_App folder from the Windows host PC and place it into the /home/igl2 directory of RedHat Linux host PC.
- Change to the /home/igl2/PCIe_App directory using the following command: #cd /home/igl2/PCIe_App
- 4. Compile the Linux user space application pcie_appln_ctrlpln.c by using gcc command.
- #gcc -o pcie_ctrlplane pcie_appln_ctrlpln.c



	0.15 PH (1)
Applications Places System System Set Toot@Jocalhost:/home//gl2/PCIe_App	8:15 PM 🌒
File Edit View Terminal Tabs Help	
<pre>[root@localhost PCIe_App]# pwd /home/igl2/PCIe_App [root@localhost PCIe_App]# ls led_blink.sh pcie_appln_ctrlpln.c pcie_config.sh [root@localhost PCIe_App]# ls led_blink.sh pcie_appln_ctrlpln.c pcie_config.sh pcie_ctrlplane [root@localhost PCIe_App]# ls</pre>	
Description: LED Control [1], Led Data [0x0-0x000000FF] Example: ./pcie_ctrlplane 1 0x000000FF	
Description: SRAM_WRITE [2], Write[1], SRAM Data [0x0-0xFFFFFFF], SRAM Offset[0x0-0x3FFF] Example: ./pcie_ctrlplane 2 1 0x12345678 0x10	
Description: SRAM_READ [3], Read[0], SRAM Offset[0x0-0x1FFF] Example : ./pcie_ctrlplane 3 0 0x10	
Description: Dip Switch Status [4] Example: ./pcie_ctrlplane 4	
Description: PCIe Device Info [5], Display PCIe Configuration Space/PCIe Device Detailed Info[1/2] Example: ./pcie_ctrlplane 5 1	
Description: PCIe Interrupt Control [6], Enable/(Clear/Disable)[1/0] Example: ./pcie_ctrlplane 6 1	
Description: PCIe Interrupt Count [7] Example: ./pcie_ctrlplane 7	
[root@localhost PCIe_App]#]	
Toot@localhost:/home/igl2/PCIe_App	9

Figure 67 • Linux PCIe Application Utility

3

- 5. After successful compilation, the Linux PCIe application utility pcie_ctrlplane is created in the same directory.
- 6. On Linux Command Prompt run the pcie_ctrlplane utility as:
 - #./pcie_ctrlplane
- 7. Help menu displays as shown in Figure 67.



Execution of Linux PCIe Control Plane Features **LED Control**

LED1 to LED8 is controlled by writing data to IGLOO2 LED control registers.

- #./pcie ctrlplane 1 0x000000FF [LED ON]
- #./pcie_ctrlplane 1 0x00000000 [LED OFF]

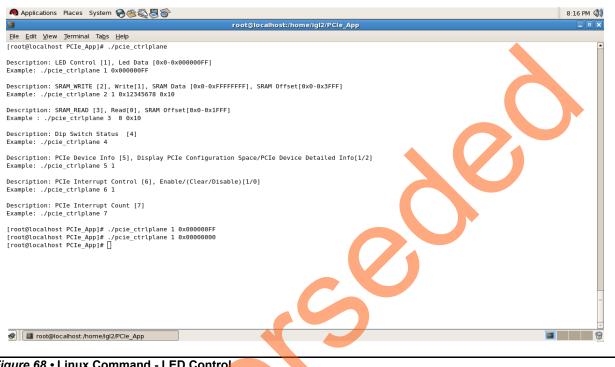


Figure 68 • Linux Command - LED Control

led blink.sh, contains the shell script code to perform LED Walk ON where as Ctrl C exits the shell script and LED Walk turns OFF.

Run the led blink, sh shell script using sh command.

#sh led blink.sh



SRAM Read/Write

32 KB SRAM is accessible for IGLOO2 Evaluation Kit.

- #./pcie_ctrlplane 2 1 0xFF00FF00 0x1000 [SRAM WRITE]
- #./pcie_ctrlplane 3 0 0x1000 [SRAM READ]

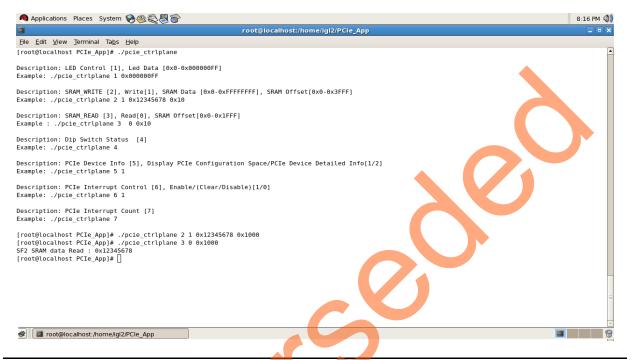


Figure 69 • Linux Command - SRAM Read/Write



DIP Switch Status

Dip switch on IGLOO2 Evaluation Kit board consists of 4 electric switches to hold the device configurations. Linux PCIe utility reads the corresponding switches (ON/OFF) state.

#./pcie_ctrlplane	4	[DIP	Switch	Status	l
-------------------	---	------	--------	--------	---

	root@localhost:/home/igl2/PCIe_App	
le Edit View Terminal Tabs Help		
oot@localhost PCIe_App]# ./pcie_ctrlplan	2	
scription: LED Control [1], Led Data [0x ample: ./pcie_ctrlplane 1 0x000000FF)-0x000000FF]	
scription: SRAM_WRITE [2], Write[1], SRA ample: ./pcie_ctrlplane 2 1 0x12345678 0	lData [0x0-0xFFFFFFF], SRAM Offset[0x0-0x3FFF] <10	
scription: SRAM_READ [3], Read[0], SRAM ample : ./pcie_ctrlplane 3 00x10)ffset[0x0-0x1FFF]	
scription: Dip Switch Status [4] ample: ./pcie_ctrlplane 4		
scription: PCIe Device Info [5], Display ample: ./pcie_ctrlplane 5 1	PCIe Configuration Space/PCIe Device Detailed Info[1/2]	
scription: PCIe Interrupt Control [6], E ample: ./pcie_ctrlplane 6 1	hable/(Clear/Disable)[1/0]	
scription: PCIe Interrupt Count [7] ample: ./pcie_ctrlplane 7		
oot@localhost PCIe_App]# ./pcie_ctrlplan P Switch Data Register Value : 0x200 1 : ON	2 4	
2 : 0FF		
3 : ON 4 : ON		
oot@localhost PCIe_App]# [
B root@localhost:/home/igl2/PCle_App		
ure 70 • Linux Command - I	DIP Switch	



PCIe Configuration Space Display

PCIe Configuration Space contains the PCIe device data such as Vendor ID, Device ID, and Base Address 0.

Note: Root Privileges are required to execute this command.

#./pcie_ctrlplane 5 1 [Read PCIe Configuration Space]

A Applications Places System 🔗 🎯 🥸 🚳 8:18 PM 🔅)))
root@localhost:/home/igi2/PCie_App	×
File Edit View Terminal Tabs Help	
[root@localhost PCIe_App]# ./pcie_ctrlplane	
Description: LED Control [1], Led Data [0x0-0x00000FF] Example: ./pcie_ctrlplane 1 0x000000FF	
Description: SRAM_WRITE [2], Write[1], SRAM Data [0x0-0xFFFFFFF], SRAM Offset[0x0-0x3FFF] Example: ./pcie_ctrlplane 2 1 0x12345678 0x10	
Description: SRAM_READ [3], Read[0], SRAM Offset[0x0-0x1FFF] Example : ./pcie_ctrlplane 3 0 0x10	
Description: Dip Switch Status [4] Example: ./pcie_ctrlplane 4	
Description: PCIe Device Info [5], Display PCIe Configuration Space/PCIe Device Detailed Info[1/2] Example: ./pcie_ctrlplane 5 1	
Description: PCIe Interrupt Control [6], Enable/(Clear/Disable)[1/0] Example: ./pcie_ctrlplane 6 1	
Description: PCIe Interrupt Count [7] Example: ./pcie_ctrlplane 7	
[root@localhost PCIe_App]# ./pcie_ctrlplane 5 1 Name Data Description	
1. VID 0x1laa Vendor Id 2. DID 0x1laa Device ID 3. CMD 0x0406 Command 4. STS 0x0010 Status 5. RID_CLCD 0x000 Revision ID & Class Code 6. SCC 0x000 Base Class Code 7. BCC 0x00 Base Class Code 8. CALN 0x10 Cache Line Size 9. LAT 0x00 Header Type 11. BIST 0x00 Built-in Self Test	
12.BADDR0 0xfe500000 Base Adress 0 13.BADDR1 0xfe4f0000 Base Adress 1 14.BADDR2 0x00000000 Base Adress 2 15.BADDR3 0x00000000 Base Adress 3 15.BADDR4 0x00000000 Base Adress 4	
17.BADDR5 0x0000000 Base Adress 5 18.CIS 0x0000000 CardBus CIS Pointer 19.SVID 0x11aa Sub-system Vendor ID 20.SUD 0x0000 Sub-System Device ID	
21.EROM 0x0000000 Expansion ROM Base Address 22.NEW_CAP 0x50 New Capabilities Pointer 23.INTLN 0x0b Interrupt Line 24.INTPIN 0x01 Interrupt Pin	
25.MINGNT 0x00 Minimum Required Burst Period 26.MAXLAT 0x00 Maximum Latency [root@localhost PCIe_App]#]	=
Toot@localhost:/home//gl2/PCIe_App	- 9

Figure 71 • Linux Command - PCle Configuration Space Display



PCIe Link Speed and Width

Note: Root Privileges are required to execute this command.

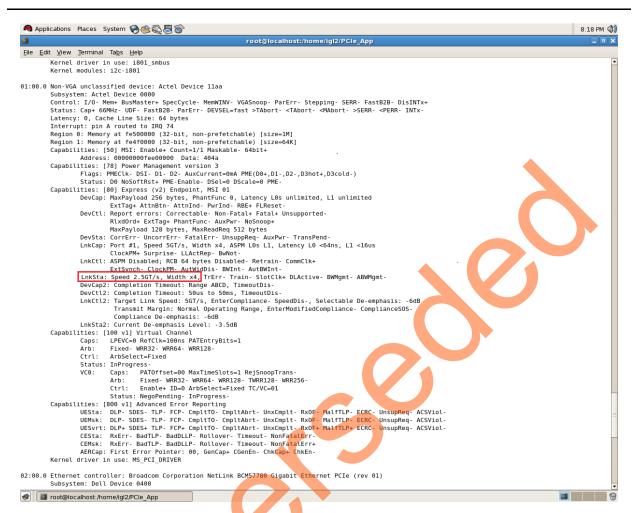
#./pcie_ctrlplane 5 2 [Read PCIe Link Speed and Link Width]

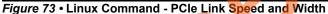
Applications Places System 🔗 🕸 🖏 🗑 🎯	8:18 PM 🜒
root@localhost:/home/igl2/PCie_App	_ • ×
Elle Edit View Terminal Tabs Help	
[root@localhost PCIe_App]# ./pcie_ctrlplane	-
Description: LED Control [1], Led Data [0x0-0x000000FF] Example: ./pcie_ctrlplane 1 0x000000FF	
Description: SRAM_WRITE [2], Write[1], SRAM Data [0x0-0xFFFFFFF], SRAM Offset[0x0-0x3FFF] Example: ./pcie_ctrlplane 2 1 0x12345678 0x10	
Description: SRAM_READ [3], Read[0], SRAM Offset[0x0-0x1FFF] Example : ./pcie_ctrlplane 3 0 0x10	
Description: Dip Switch Status [4] Example: ./pcie_ctrlplane 4	
Description: PCIe Device Info [5], Display PCIe Configuration Space/PCIe Device Detailed Info[1/2] Example: ./pcie_ctrlplane 5 1	
Description: PCIe Interrupt Control [6], Enable/(Clear/Disable)[1/0] Example: ./pcie_ctrlplane 6 1	
Description: PCIe Interrupt Count [7] Example: ./pcie_ctrlplane 7	
[root@localhost PCIe_App]# ./pcie_ctrlplane 5 2]	
	v
🕐 🔄 root@localhost:/home/igl2/PCle_App	

Figure 72 • Linux Command - PCle Link Speed and Width

3









PCIe Interrupt Control (Enable/Disable) and Interrupt Counter

IGLOO2 Evaluation Kit enable/disable the MSI interrupts by writing data to its PCIe configuration space. Interrupt counter holds the number of MSI interrupts got triggered by pressing the SW4 push button.

- #. /pcie_ctrlplane 6 0 [Disable Interrupts]
- #. /pcie_ctrlplane 6 1 [Enable Interrupts]
- #. /pcie_ctrlplane 7 [Interrupt Counter Value]

▲ Applications Places System ⊖૱등	8:20 PM 🐠
root©localhost:/home/igi2/PCIe_App	
Ele Edit View Terminal Tabs Help	
[root@localhost PCIe_App]# ./pcie_ctrlplane	-
Description: LED Control [1], Led Data [0x0-0x000000FF] Example: ./pcie_ctrlplane 1 0x000000FF	
Description: SRAM_WRITE [2], Write[1], SRAM Data [0x0-0xFFFFFFF], SRAM Offset[0x0-0x3FFF] Example: ./pcie_ctrlplane 2 1 0x12345678 0x10	
Description: SRAM_READ [3], Read[0], SRAM Offset[0x0-0x1FFF] Example : ./pcie_ctrlplane 3 0 0x10	
Description: Dip Switch Status [4] Example: ./pcie_ctrlplane 4	
Description: PCIe Device Info [5], Display PCIe Configuration Space/PCIe Device Detailed Info[1/2] Example: ./pcie_ctrlplane 5 1	
Description: PCIe Interrupt Control [6], Enable/(Clear/Disable)[1/0] Example: ./pcie_ctrlplane 6 1	
Description: PCIe Interrupt Count [7] Example: ./pcie_ctrlplane 7	
<pre>[root@localhost PCIe_App]# ./pcie_ctrlplane 6 1 [root@localhost PCIe_App]# ./pcie_ctrlplane 7 SF2 PCIE Interrupt Counter Value : 3 [root@localhost PCIe_App]# ./pcie_ctrlplane 6 0 [root@localhost PCIe_App]# ./pcie_ctrlplane 7 SF2 PCIE Interrupt Counter Value : 0 [root@localhost PCIe_App]#]</pre>	
Toot@localhost:/home//gl2/PCIe_App	9

Figure 74 • Linux Command - PCle Interrupt Control

Conclusion

This tutorial describes how to access the PCIe endpoint features of IGLOO2 and how to create a simple design. It describes the steps to verify the design with BFM simulation. This tutorial demonstrates that the host PC can easily communicate with IGLOO2 Evaluation Kit board through the provided GUI and Drivers. This tutorial also provides a Linux PCIe application for accessing the PCIe EP device through the Linux PCIe Device Driver.



A – IGLOO2 Evaluation Kit Board

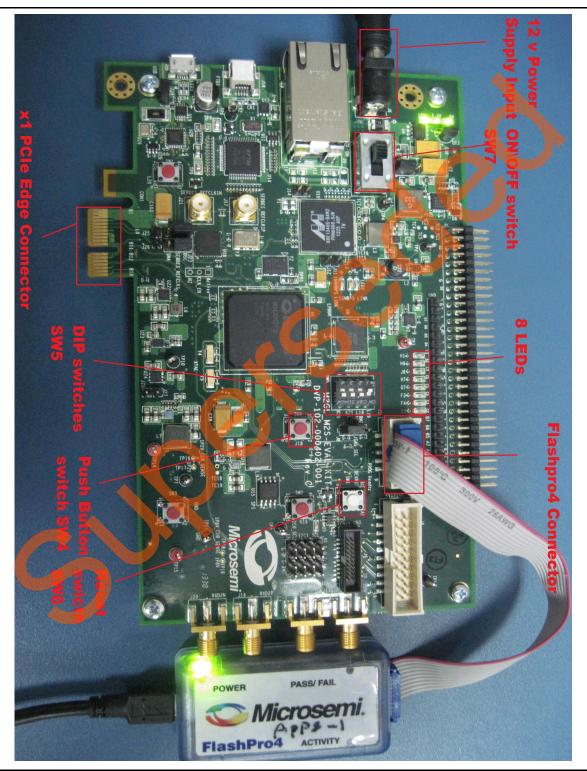


Figure 1 • IGLOO2 Evaluation Kit Board



B – IGLOO2 Evaluation Kit Board Setup for Laptop

Figure 1 shows how to line up the IGLOO2 Evaluation Kit PCIe connector with the adapter card slot.





Note: The Notch (highlighted in red) does not go into the adapter card.





Figure 2 shows IGLOO2 Evaluation Kit PCIe connector inserted into the PCIe adapter card slot.

Figure 2 • Inserting the IGLOO2 Evaluation Kit PCIe Connector



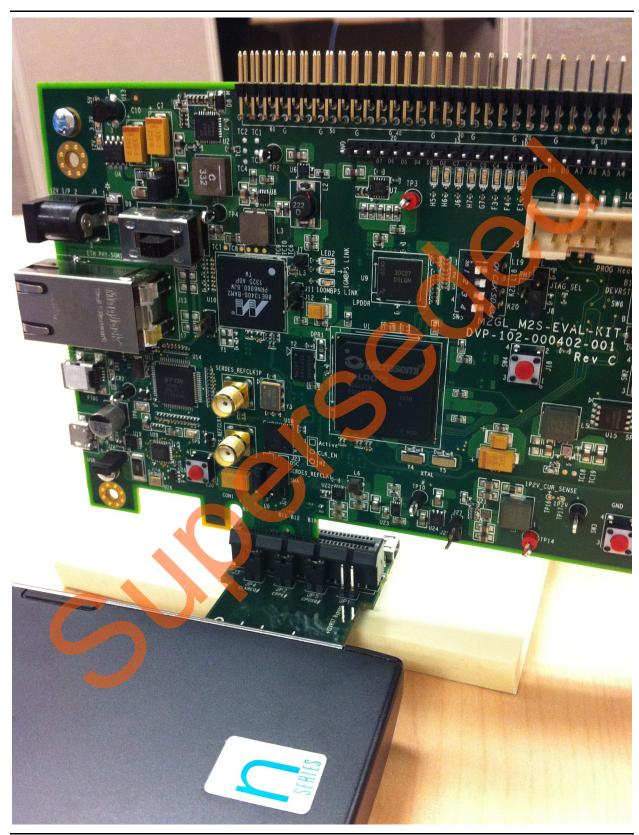


Figure 3 shows the PCIe adapter card and the IGLOO2 Evaluation Kit connected to the laptop.

Figure 3 • IGLOO2 Evaluation Kit Connected to the Laptop



C – List of Changes

3

The following table lists critical changes that were made in each revision.

Date	Changes	Page
Revision 5 (August 2014)	Updated the design files link under "Associated Project Files" section.	4
Revision 4 (July 2014)	Updated the document for Libero v11.4 software release (SAR 59562).	NA
Revision 3 (April 2014)	Updated the document for Libero v11.3 software release (SAR 55917).	NA
Revision 2 (February 2014)	Added the section "Step 7: Running the Design".	NA
Revision 1 (January 2014)	Updated the document for Libero v11.2 software release (SAR 53311).	NA
Revision 0 (November 2013)	Initial release.	NA



D – **Product Support**

Microsemi SoC Products Group backs its products with various support services, including Customer Service, Customer Technical Support Center, a website, electronic mail, and worldwide sales offices. This appendix contains information about contacting Microsemi SoC Products Group and using these support services.

Customer Service

Contact Customer Service for non-technical product support, such as product pricing, product upgrades, update information, order status, and authorization.

From North America, call 800.262.1060 From the rest of the world, call 650.318.4460 Fax, from anywhere in the world, 408.643.6913

Customer Technical Support Center

Microsemi SoC Products Group staffs its Customer Technical Support Center with highly skilled engineers who can help answer your hardware, software, and design questions about Microsemi SoC Products. The Customer Technical Support Center spends a great deal of time creating application notes, answers to common design cycle questions, documentation of known issues, and various FAQs. So, before you contact us, please visit our online resources. It is very likely we have already answered your questions.

Technical Support

Visit the Customer Support website (www.microsemi.com/soc/support/search/default.aspx) for more information and support. Many answers available on the searchable web resource include diagrams, illustrations, and links to other resources on the website.

Website

You can browse a variety of technical and non-technical information on the SoC home page, at www.microsemi.com/soc.

Contacting the Customer Technical Support Center

Highly skilled engineers staff the Technical Support Center. The Technical Support Center can be contacted by email or through the Microsemi SoC Products Group website.

Email

You can communicate your technical questions to our email address and receive answers back by email, fax, or phone. Also, if you have design problems, you can email your design files to receive assistance. We constantly monitor the email account throughout the day. When sending your request to us, please be sure to include your full name, company name, and your contact information for efficient processing of your request.

The technical support email address is soc_tech@microsemi.com.



My Cases

Microsemi SoC Products Group customers may submit and track technical cases online by going to My Cases.

Outside the U.S.

Customers needing assistance outside the US time zones can either contact technical support via email (soc_tech@microsemi.com) or contact a local sales office. Sales office listings can be found at www.microsemi.com/soc/company/contact/default.aspx.

ITAR Technical Support

For technical support on RH and RT FPGAs that are regulated by International Traffic in Arms Regulations (ITAR), contact us via soc_tech_itar@microsemi.com. Alternatively, within My Cases, select **Yes** in the ITAR drop-down list. For a complete list of ITAR-regulated Microsemi FPGAs, visit the ITAR web page.



Microsemi Corporate Headquarters One Enterprise, Aliso Viejo CA 92656 USA Within the USA: +1 (800) 713-4113 Outside the USA: +1 (949) 380-6100 Sales: +1 (949) 380-6136 Fax: +1 (949) 215-4996 E-mail: sales.support@microsemi.com Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for communications, defense and security, aerospace, and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs, and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; security technologies and scalable anti-tamper products; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, Calif. and has approximately 3,400 employees globally. Learn more at **www.microsemi.com**.

© 2014 Microsemi Corporation. All rights reserved. Microsemi and the Microsemi logo are trademarks of Microsemi Corporation. All other trademarks and service marks are the property of their respective owners.