TU0509 Tutorial





Table of Contents

Introduction 3 Design Requirements 3 Project Files 4 Components Used 4 Design Overview 4 Step 1: Creating a Libero SoC Project 6 Launching Libero SoC 6 Instantiating SERDESIF Component in PCle_Demo_top SmartDesign 13 Instantiating Debounce Logic in PCle_Demo_top SmartDesign 13 Instantiating CoreAHBLSRAM in PCle_Demo_top SmartDesign 20 Instantiating CoreAHBLSRAM in PCle_Demo_top SmartDesign 27 Instantiating Clock Conditioning Circuity (CCC) in PCle_Demo_top SmartDesign 27 Instantiating the Posyment File 46 Step 3: Simulating the Design 41 Step 4: Generating the Program File 46 Step 5: Programming the IGLOO2 Board Using FlashP
Design Requirements 3 Project Files 4 Components Used 4 Design Overview 4 Step 1: Creating a Libero SoC Project 6 Launching Libero SoC 6 Instantiating SERDESIF Component in PCIe_Demo_top SmartDesign 13 Instantiating Debource Logic in PCIe_Demo_top SmartDesign 18 Instantiating CoreGPIO in PCIe_Demo_top SmartDesign 20 Instantiating CoreGPIO in PCIe_Demo_top SmartDesign 27 Instantiating CoreGPIO in PCIe_Demo_top SmartDesign 27 Instantiating CoreAHBLSRAM in PCIe_Demo_top SmartDesign 27 Instantiating Cock Conditioning Circuitry (CCC) in PCIe_Demo_top SmartDesign 27 Instantiating Cock Conditioning Circuitry (CCC) in PCIe_Demo_top SmartDesign 27 Instantiating Cock Conditioning Circuitry (CCC) in PCIe_Demo_top SmartDesign 27 Instantiating Cock Conditioning Circuitry (CCC) in PCIe_Demo_top SmartDesign 27 Instantiating Cock Conditioning Circuitry (CCC) in PCIe_Demo_top SmartDesign 27 Instantiating Cock Conditioning Circuitry (CCC) in PCIe_Demo_top SmartDesign 27 Step 3: Simulating the Design 41 Step 4: Generating the Program File 46
Design Requestion 3 Project Files 4 Components Used 4 Design Overview 4 Step 1: Creating a Libero SoC Project 6 Launching Libero SoC 6 Instantiating SERDESIF Component in PCle_Demo_top SmartDesign 13 Instantiating Bus Interfaces in PCle_Demo_top SmartDesign 13 Instantiating CoreGPIO in PCle_Demo_top SmartDesign 20 Instantiating CoreGPIO in PCle_Demo_top SmartDesign 27 Instantiating Cock Conditioning Circuitry (CCC) in PCle_Demo_top SmartDesign 27 Instantiating Cock Conditioning Circuitry (PCC) in PCle_Demo_top SmartDesign 27 Instantiating Cock Conditioning Circuitry (PCC) in PCle_Demo_top SmartDesign 29 Connecting Components in PCle_Demo_top SmartDesign 29 Connecting Components in PCle_Demo_top SmartDesign 29 Connecting Components in PCle_Demo_top SmartDesign 31 Step 2: Developing the Simulation Stimulus 38 Step 3: Simulating the Design 41 Step 4: Generating the Program File 46 Step 5: Programming the IGLOO2 Board Using FlashPro 52 Running the Design on Linux 52 Conclusio
Components Used 4 Design Overview 4 Step 1: Creating a Libero SoC Project 6 Launching Libero SoC 6 Instantiating SERDESIF Component in PCle_Demo_top SmartDesign 13 Instantiating Debounce Logic in PCle_Demo_top SmartDesign 18 Instantiating CoreGPIO in PCle_Demo_top SmartDesign 20 Instantiating CoreGPIO in PCle_Demo_top SmartDesign 27 Instantiating CoreAHBLSRAM in PCle_Demo_top SmartDesign 27 Instantiating CoreAHBLSRAM in PCle_Demo_top SmartDesign 27 Instantiating CoreAHBLSRAM in PCle_Demo_top SmartDesign 27 Instantiating CoreCPIO in PCle_Demo_top SmartDesign 27 Instantiating CoreCPIO in PCle_Demo_top SmartDesign 29 Connecting Components in PCle_Demo_top SmartDesign 29 Connecting Components in PCle_Demo_top SmartDesign 31 Step 3: Simulating the Design 41 Step 4: Generating the Program File 46 Step 5: Programming the IGLOO2 Board Using FlashPro 49 Step 6: Connecting the Evaluation Kit to the Host PC 51 Step 7: Running the Design on Windows 52 Running the Design on Linux 62 C
Design Overview 4 Step 1: Creating a Libero SoC Project 6 Launching Libero SoC 6 Instantiating SERDESIF Component in PCle_Demo_top SmartDesign 13 Instantiating BCRDESIF Component in PCle_Demo_top SmartDesign 13 Instantiating BCRDESIF Component in PCle_Demo_top SmartDesign 20 Instantiating CoreAHBLSRAM in PCle_Demo_top SmartDesign 27 Step 3: Simulation Stimulus 38 Step 4: Generating the Posign 41 <t< td=""></t<>
Step 1: Creating a Libero SoC Project 6 Launching Libero SoC 6 Instantiating SERDESIF Component in PCIe_Demo_top SmartDesign 13 Instantiating Debounce Logic in PCIe_Demo_top SmartDesign 18 Instantiating Debounce Logic in PCIe_Demo_top SmartDesign 20 Instantiating CoreGPIO in PCIe_Demo_top SmartDesign 20 Instantiating CoreGPIO in PCIe_Demo_top SmartDesign 27 Instantiating CoreGPIO in PCIe_Demo_top SmartDesign 29 Connecting Components in PCIe_Demo_top SmartDesign 29 Connecting the Simulation Stimulus 38 Step 3: Simulating the Design 41 Step 4: Generating the Program File 46 Step 5: Programming the IGLOO2 Board Using FlashPro 49 Step 6: Connecting the Evaluation Kit to the Host PC 51 Running the Design on Uninux 52 Running the Design on Linux
Launching Libero SoC 6 Instantiating SERDESIF Component in PCIe_Demo_top SmartDesign 13 Instantiating Debounce Logic in PCIe_Demo_top SmartDesign 18 Instantiating CoreGPIO in PCIe_Demo_top SmartDesign 20 Instantiating CoreAHBLSRAM in PCIe_Demo_top SmartDesign 27 Instantiating CoreAHBLSRAM in PCIe_Demo_top SmartDesign 29 Connecting Components in PCIe_Demo_top SmartDesign 29 Connecting Components in PCIe_Demo_top SmartDesign 31 Step 2: Developing the Simulation Stimulus 38 Step 3: Simulating the Design 41 Step 4: Generating the Program File 46 Step 5: Programming the IGLOO2 Board Using FlashPro 49 Step 6: Connecting the Evaluation Kit to the Host PC 51 Step 7: Running the Design on Windows 52 Running the Design on Unidows 52 Running the Design on Linux 62 Conclusion 72 A IGLOO2 Evaluation Kit Board 73 B IGLOO2 Evaluation Kit Board Setup for Laptop 74 C List of Changes 77 D Product Support 78
Instantiating SERDESIF Component in PCIe_Demo_top SmartDesign 13 Instantiating Debounce Logic in PCIe_Demo_top SmartDesign 18 Instantiating Bus Interfaces in PCIe_Demo_top SmartDesign 20 Instantiating CoreGPIO in PCIe_Demo_top SmartDesign 25 Instantiating CoreAHBLSRAM in PCIe_Demo_top SmartDesign 27 Instantiating CoreAHBLSRAM in PCIe_Demo_top SmartDesign 27 Instantiating CoreAHBLSRAM in PCIe_Demo_top SmartDesign 29 Connecting Components in PCIe_Demo_top SmartDesign 31 Step 2: Developing the Simulation Stimulus 38 Step 3: Simulating the Design 41 Step 4: Generating the Program File 46 Step 5: Programming the IGLOO2 Board Using FlashPro 49 Step 6: Connecting the Evaluation Kit to the Host PC 51 Step 7: Running the Design 51 Running the Design on Windows. 52 Running the Design on Linux 62 Conclusion 72 A IGLOO2 Evaluation Kit Board 73 B IGLOO2 Evaluation Kit Board Setup for Laptop. 74 C List of Changes 77 D Product Support 78
Instantiating Debounce Logic in PCle_Demo_top SmartDesign 18 Instantiating Bus Interfaces in PCle_Demo_top SmartDesign 20 Instantiating CoreGPIO in PCle_Demo_top SmartDesign 25 Instantiating CoreGPIO in PCle_Demo_top SmartDesign 27 Instantiating CoreAHBLSRAM in PCle_Demo_top SmartDesign 27 Instantiating CoreAHBLSRAM in PCle_Demo_top SmartDesign 29 Connecting Components in PCle_Demo_top SmartDesign 31 Step 2: Developing the Simulation Stimulus 38 Step 3: Simulating the Design 41 Step 4: Generating the Program File 46 Step 5: Programming the IGLOO2 Board Using FlashPro 49 Step 6: Connecting the Evaluation Kit to the Host PC 51 Step 7: Running the Design 52 Running the Design on Windows 52 Running the Design on Linux 62 Conclusion 72 A IGLOO2 Evaluation Kit Board 73 B IGLOO2 Evaluation Kit Board Setup for Laptop 74 C List of Changes 77 D Product Support 78
Instantiating CorreGPIO in PCIe_Demo_top SmartDesign 25 Instantiating CoreGPIO in PCIe_Demo_top SmartDesign 27 Instantiating CoreAHBLSRAM in PCIe_Demo_top SmartDesign 29 Connecting Components in PCIe_Demo_top SmartDesign 31 Step 2: Developing the Simulation Stimulus 38 Step 3: Simulating the Design 41 Step 4: Generating the Program File 46 Step 5: Programming the IGLOO2 Board Using FlashPro 49 Step 6: Connecting the Evaluation Kit to the Host PC 51 Step 7: Running the Design 51 Running the Design on Windows 52 Running the Design on Linux 62 Conclusion 72 A IGLOO2 Evaluation Kit Board 73 B IGLOO2 Evaluation Kit Board Setup for Laptop 74 C List of Changes 77 D Product Support 78
Instantiating CoreAHBLSRAM in PCle_Demo_top SmartDesign 27 Instantiating CoreAHBLSRAM in PCle_Demo_top SmartDesign 29 Connecting Components in PCle_Demo_top SmartDesign 29 Connecting Components in PCle_Demo_top SmartDesign 31 Step 2: Developing the Simulation Stimulus 38 Step 3: Simulating the Design 41 Step 4: Generating the Program File 46 Step 5: Programming the IGLOO2 Board Using FlashPro 49 Step 6: Connecting the Evaluation Kit to the Host PC 51 Step 7: Running the Design 51 Running the Design on Windows 52 Running the Design on Linux 62 Conclusion 72 A IGLOO2 Evaluation Kit Board 73 B IGLOO2 Evaluation Kit Board Setup for Laptop. 74 C List of Changes 77 D Product Support 78
Instantiating Clock Conditioning Circuitry (CCC) in PCle_Demo_top SmartDesign 29 Connecting Components in PCle_Demo_top SmartDesign 31 Step 2: Developing the Simulation Stimulus 38 Step 3: Simulating the Design 41 Step 4: Generating the Program File 46 Step 5: Programming the IGLOO2 Board Using FlashPro 49 Step 6: Connecting the Evaluation Kit to the Host PC 51 Step 7: Running the Design 51 Running the Design on Windows 52 Running the Design on Linux 62 Conclusion 72 A IGLOO2 Evaluation Kit Board 73 B IGLOO2 Evaluation Kit Board Setup for Laptop 74 C List of Changes 77 D Product Support 78
Connecting Components in PCIe_Demo_top SmartDesign 31 Step 2: Developing the Simulation Stimulus 38 Step 3: Simulating the Design 41 Step 4: Generating the Program File 46 Step 5: Programming the IGLOO2 Board Using FlashPro 49 Step 6: Connecting the Evaluation Kit to the Host PC 51 Step 7: Running the Design 51 Running the Design on Windows 52 Running the Design on Linux 62 Conclusion 72 A IGLOO2 Evaluation Kit Board 73 B IGLOO2 Evaluation Kit Board Setup for Laptop 74 C List of Changes 77 P Product Support 78
Step 2: Developing the Simulation Stimulus 38 Step 3: Simulating the Design 41 Step 4: Generating the Program File 46 Step 5: Programming the IGLOO2 Board Using FlashPro 49 Step 6: Connecting the Evaluation Kit to the Host PC 51 Step 7: Running the Design 51 Running the Design on Windows 52 Running the Design on Linux 62 Conclusion 72 A IGLOO2 Evaluation Kit Board 73 B IGLOO2 Evaluation Kit Board Setup for Laptop. 74 C List of Changes 77 P Product Support 78
Step 3: Simulating the Design 41 Step 4: Generating the Program File 46 Step 5: Programming the IGLOO2 Board Using FlashPro 49 Step 6: Connecting the Evaluation Kit to the Host PC 51 Step 7: Running the Design 51 Running the Design on Windows 52 Running the Design on Linux 62 Conclusion 72 A IGLOO2 Evaluation Kit Board 73 B IGLOO2 Evaluation Kit Board Setup for Laptop 74 C List of Changes 77 D Product Support 78
Step 4: Generating the Program File 46 Step 5: Programming the IGLOO2 Board Using FlashPro 49 Step 6: Connecting the Evaluation Kit to the Host PC 51 Step 7: Running the Design 51 Running the Design on Windows 52 Running the Design on Linux 62 Conclusion 72 A IGLOO2 Evaluation Kit Board 73 B IGLOO2 Evaluation Kit Board Setup for Laptop 74 C List of Changes 77 D Product Support 78
Step 5: Programming the IGLOO2 Board Using FlashPro 49 Step 6: Connecting the Evaluation Kit to the Host PC 51 Step 7: Running the Design 51 Running the Design on Windows 52 Running the Design on Linux 62 Conclusion 72 A IGLOO2 Evaluation Kit Board 73 B IGLOO2 Evaluation Kit Board Setup for Laptop 74 C List of Changes 77 D Product Support 78
Step 6: Connecting the Evaluation Kit to the Host PC 51 Step 7: Running the Design 51 Running the Design on Windows 52 Running the Design on Linux 62 Conclusion 72 A IGLOO2 Evaluation Kit Board 73 B IGLOO2 Evaluation Kit Board Setup for Laptop. 74 C List of Changes 77
Step 7: Running the Design 51 Running the Design on Windows 52 Running the Design on Linux 62 Conclusion 72 A IGLOO2 Evaluation Kit Board 73 B IGLOO2 Evaluation Kit Board Setup for Laptop 74 C List of Changes 77 D Product Support 78
Running the Design on Windows 52 Running the Design on Linux 62 Conclusion 72 A IGLOO2 Evaluation Kit Board 73 B IGLOO2 Evaluation Kit Board Setup for Laptop. 74 C List of Changes 77 D Product Support 78
Running the Design on Linux 62 Conclusion 72 A IGLOO2 Evaluation Kit Board 73 B IGLOO2 Evaluation Kit Board Setup for Laptop. 74 C List of Changes 77 D Product Support 78
Conclusion 72 A IGLOO2 Evaluation Kit Board 73 B IGLOO2 Evaluation Kit Board Setup for Laptop. 74 C List of Changes 77 D Product Support 78
 A IGLOO2 Evaluation Kit Board
 A IGLOO2 Evaluation Kit Board
 B IGLOO2 Evaluation Kit Board Setup for Laptop
C List of Changes
C List of Changes
D Product Support
D. Product Support
Customer Technical Support Center
Technical Support
Website
Contacting the Customer Technical Support Center
Email
Outside the U.S. 79
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 Systems (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.

The following tasks are explained in this tutorial:

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

Design Requirements

Table 1 lists the design requirements of IGLOO2 PCIe control plane tutorial.

Table 1 • Design Requirements

	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)	-
Software Requirements	
Libero SoC	v11.6
FlashPro programming software	v11.6
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.



Project Files

Download the design files from the Microsemi[®] website: http://soc.microsemi.com/download/rsc/?f=m2gl_tu0509_liberov11p6_df Design files include:

- Libero project
- Programming File
- Linux 64bit
- Windows_64bit
- Source Files
- Readme

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

The 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. Refer to the *UG0447: SmartFusion2 and IGLOO2 High Speed Serial Interfaces User Guide* for more information on SERDESIF.

The design helps accessing 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 the CoreAHBLite, CoreAHBTOAPB, and CoreAPB3 bus interfaces.

SERDES_IF_0 is initialized by CoreConfig master. The SERDES_IF_0 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.

🏷 Microsemi.

Implementing PCIe Control Plane Design in IGLOO2 FPGA

Step 1: Creating a Libero SoC Project

The following steps describe how to create an IGLOO2 PCIe control plane design using the Libero tool.

Launching Libero SoC

- 1. Choose Start > Programs > Microsemi Libero SoC v11.6 > Libero SoC v11.6, or double-click the shortcut on desktop to open the Libero SoC v11.6 Project Manager.
- 2. Create a new project using one of the following options:
 - Select New on the Start Page tab, as shown in Figure 2.
 - Click **Project > New Project** from the Libero SoC menu.



Figure 2 • Libero SoC Project Manager



- 3. Enter the following **New Project** information as shown in Figure 3 and click **Next**.
 - Project
 - Project Name: PCIE_Demo
 - Project Location: Select an appropriate location (for example, D:/microsemi_prj)
 - Preferred HDL Type: Verilog or VHDL

O New Project	
Project Details Specify Project Details	
Project Details	Project Name: PCIE_Demo
Device Selection	Project Location: D:/Microsemi_proj Browse
Device Settings	
Design Template	Preferred HDL Type: Verilog
Add HDL Sources	
Add Constraints	
Help	< Back Next > Finish Cancel

Figure 3 • Libero SoC New Project Dialog Box

S



- 4. Select the following values using the drop-down list for **Device Selection** as shown in Figure 4 and click **Next**.
 - Part Filter
 - Family: IGLOO2
 - Die: M2GL010T
 - Package: 484 FBGA
 - Speed: -1
 - Core Voltage: 1.2
 - Range: COM

evice selection Select a part for your project	from the part number list						Selected part: M20	5L010T-1F
Project Details	Part filter							
	Family: IGLOO	2	Die:	M2GL010T	Package:	484 FBGA		
Device Selection	Speed: -1		Core voitage:	1.2	• Range:	COM		
						Rese	t filters	
Device Settings	Search part:							
	Part Number	4LUT	DFF	User I/Os	USRAM 1K	LSRAM 18K	Math (18x18)	PLLs an
Design Template	M2GL0101-1FG484	12084	12084	233		4		2
Add HDL Sources								
Add HDE Sources								
Add Constraints								
-								
ibero)								
tem-on-Chip	•							,

Figure 4 • New Project - Device Selection



5. Select the **PLL supply voltage (V)** as 3.3 from the drop-down list as shown in Figure 5 and click **Next**.

New project		
Device settings Choose device settings for y	pur project	Selected part: M2GL010T-1FG484
Project Details	I/O settings Default I/O technology: LVCMOS 2.5V Please use the I/O Editor to change individual I/O attributes. Reserve pins for probes	
Device Selection		
Device Settings	Power supplies	
Design Template	PLL supply voltage (V): 3.3 Power on Reset delay : 100ms	
Add HDL Sources	System controller suspended mode	
Add Constraints		
Help	< <u>B</u> ack	Next > Finish Cancel

Figure 5 • New Project - Device Settings

6. Select Create a System Builder based design under Design Templates and Creators as shown in Figure 6.

New Project		
Design Template Choose a design template		Selected Part: M2GL010T-1FG484
Project Details	Design Templates and Creators None Orceans a System Builder based design	
Device Selection	Create a Microcontroller (MSS) based design	
Device Settings	Core	Version
Design Template		☑ Show only latest version
Add HDL Sources	Design Methodology Use Standalone Initialization for MDDR/FDDR/SERDES Peripherals	
Add Constraints		
Help		< Back Next > Einish Cancel

Figure 6 • New Project - Device Settings



7. Click **Finish** and enter **PCIe_Demo** as the name of the system in the **System Builder** dialog box as shown in Figure 7.



Figure 7 • System Builder Dialog Box

- 8. Click OK. The System Builder Device Features window is displayed.
- 9. In the System Builder Device Features tab, select the SERDESIF_0 check box under High Speed Serial Interfaces as shown in Figure 8.



Figure 8 • System Builder – Device Features Tab

 Click Next. The System Builder – Peripherals tab is displayed. Do not change the default selections.



11. Click Next. The System Builder – Clocks tab is displayed, as shown in Figure 9. Select System Clock source as On-chip 25/50 MHz RC Oscillator and HPMS_CLK as 100 MHz.

lock Fabric CCC Chip O	Iscillators		
System Clock			
50.0 MHz			
On-chip 25/50 MHz RC Oscillat	tor 🔻		
HPMS Clock		PMS_CCCC	HPDMA AHB Bus Matrix
HPMS_CLK	= 100.00 MHz 100.000		
MDDR Clocks			
MDDR_CLK	= HPMS_CLK * 1 V	40%4_0_C1 K	
DDR/SMC_FIC_CLK	= MDDR_CLK / 1 V	FIC.0.C.K	
Fabric Interface Clocks		TTK RW	
FIC 0 CLK	= HPMS_CLK / 4 = 25,000	PMS+	10,0
FIC 1 CIK	= HPMS CIK / 1	Lowest frequency	
		CCCPLL FIC_0_C_K	FIC_0 Solaryslam
Fabric DDR Clocks			
	= 100 MHZ		
FDDR_SUBSYSTEM_CLK	= FDDR_CLK / 1	OSC	
		Fabric	

Figure 9 • System Builder – Clocks Tab

- 12. Click Next. The System Builder HPMS Options tab is displayed. Do not change the default selections.
- 13. Click **Next**. The **System Builder SECDED** tab is displayed. Do not change the default selections.
- 14. Click Next. The System Builder Security tab is displayed. Do not change the default selections.
- 15. Click Next. The System Builder Memory Map tab is displayed. Do not change 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 10 on page 12.





Figure 10 • 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 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**.

🗫 CoreUART	5.5.101
🖙 CoreUARTapb	5.5.101
🖙 CoreWatchdog	1.1.101
High Speed Serial Interface	1.2.206
Processors	
SC/Tamper	
Tamper	

Figure 11 • IP Catalog

1. Drag the **High Speed Serial Interface** to 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 12:

- Identification
 - Simulation Level: BFM PCIe
- Protocol Configuration
 - Protocol1: Type: PCIe
 - Protocol1: Number of Lanes: x1
- Lane Configuration
 Speed: Lane0: 5.0 Gbps (Gen2)
- PCle Fabric SPLL Configuration
 - CLK_BASE Frequency (MHz): 100

Identification				
SerDesIF_0				Simulation Level BFM PCIe
Protocol Configuration				
Protocol 1		Protocol 2		
Type PCIe 🔻	Configure PCIe	Туре	ne 🔻	
Number of Lanes x1		Number of Lanes	T	
ana Canformation				
Lane Configuration	1 0	luci	1 2	1 2
Speed	Late 0	Lane 1	Lane Z	Lane 5
Reference Clock Source	REFCLK0 (Differential)			
PHY RefClk Frequency (MHz)	100			
Data Rate (Mbps)	N/A			
Data Width	N/A			
FPGA Interface Frequency (MHz	() N/A			
VCO Rate (MHz)	N/A			
PCIe Fabric SPLL Configuration				
CLK_BASE Frequency 100	MHz			
Register Configuration				
Edit Registers				
Later registers m				

Figure 12 • High Speed Serial Interface Configurator Window



- 3. Click **Configure PCIe** to configure the following settings as shown in Figure 13.
 - 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 Maste	er Interface Slave Inter	face	
-Identification Registers	s		
Vendor ID	0x110.0	Device ID 0x110	
Subsystem Vendor ID	0x11AA	Subsystem Device ID 0x0000	
Revision ID	0×0000	Class Code	
	0,0000		
Fabric Interface (AXI/A	AHBLite)		
Bus AHBLite		Interface Master	•
Base Address Register	rs		
	Width	Size	Prefetchable
Bar 0 32 Bits		▼ 1MB	_
Bar 1 32 Bits		• 64 КВ	•
Bar 2 None			_
Bar 3 None			
Bar 4 None			~
Bar 5 None		▼	-
Register Settings			
PCIe Specification Vers	sion Version 2.0	Interrupts MSI1	Expose Wake Signals
PHY Reference Clock S	Slot Slot	De-emphasis -3.5 dB	Transmit Swing
Power Management Se	attings		
ASPM LOs Capability		Enable ASPM L1 Capability	
LOs Acceptable Later	ncy No limit 🔻	L1 Acceptable Latency No	limit 🔹
FTS in Separate Cloc	ck Mode 63	L1 Exit Latency Separate Clock Mode 16	us to less than 32 us 🔻
ETS in Common Clock	k Mode 15	L1 Exit Latency Common Clock Mode	is to less than 16 us

Figure 13 • 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 1 MB 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 64 KB from the drop-down list
 PCIe 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 UG0447: SmartFusion2 and IGLOO2 High Speed Serial Interfaces User Guide.



Window 0	Slave Interface		
Size	[1MB ▼	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	0x9000
Window 3			
Size	4KB *	PCIe BAR	Bar 0 💌
	0v0000	PCIe Address	0x0000

Figure 14 shows the Master Interface Configuration dialog box.

Figure 14 • Master Interface Configuration Dialog Box

- 5. Click **OK** to close the PCIE Configuration for protocol 1 dialog box.
- 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. Click File > Import > HDL Source files to add the Debounce logic to the PCIe demo design.
- 2. Browse to the M2GL_PCIE_Control_Plane_11p6_DF\Source Files file location for Debounce.v or Debounce.vhd file in the design files folder. Figure 15 shows the DEBOUNCE component in the Design Hierarchy window.

Design Hierarchy	e ×
Show: Components	
▲ ① work	
TLOSC_FAB (osc_comps.v)	
TLOSC (osc_comps.v)	
RCOSC_1MHZ_FAB (osc_comps.y)	
RCOSC_1MHZ (osc_comps.v)	
▲ ① Solo PCIe_Demo_top	
PCIe_Demo	
DEBOUNCE (Debounce.v)	
COREAHBLITE_LIB	
	4
Desig Design Hier Stimulus Hie Cat	talog Files

Figure 15 • DEBOUNCE Component in Design Hierarchy Window



3. Drag the **DEBOUNCE** component from the **Design Hierarchy** to the **PCle_Demo_top SmartDesign** canvas. Figure 16 shows Debounce in **PCle_Demo_top**.



Figure 16 • DEBOUNCE Component in the PCIe_Demo_top SmartDesign Canvas



Instantiating 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 17 shows the Libero IP **Catalog**.



Figure 17 • 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 18 shows the Libero top-level design with bus interfaces.



Figure 18 • CoreAHBLite, CoreAHBtoAPB3, and CoreAPB3 Bus Interfaces in PCIe_Demo_top SmartDesign Canvas



Double-click CoreAHBLite_0 to configure it. Figure 19 shows the Configuring CoreAHBlite_0 window.

guration						
Memory space					_	
	Memory sp	ace: 4GB addressable space	apportioned int	o 16 slave slots, each of size 256MB	•	
	Address	ange seen by slave connecte	d to huge (2GB)	slot interface:		
		Ox0000000 - 0x7FFF	FFFF	0x8000000 - 0xFFFFFFF		
Allocate memory space to combined 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:	
Slot 12:		Slot 13: 🕅		Slot 14: 🔲	Slot 15:	
Enable Master access						
M0 can access slot 0:	M	1 can access slot 0:		M2 can access slot 0:	M3 can access slot 0:	
M0 can access slot 1:	M	1 can access slot 1:		M2 can access slot 1:	M3 can access slot 1:	
M0 can access slot 2:	M	1 can access slot 2:		M2 can access slot 2:	M3 can access slot 2:	
M0 can access slot 3:	M	1 can access slot 3:		M2 can access slot 3:	M3 can access slot 3:	
M0 can access slot 4:	M	1 can access slot 4:		M2 can access slot 4:	M3 can access slot 4:	
M0 can access slot 5:	M	1 can access slot 5:		M2 can access slot 5:	M3 can access slot 5:	

Figure 19 • Configuring CoreAHBLite_0

- 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.



Configuring CoreAPB3_0 (CoreAPB3	- 4.1.100)		
onfiguration			^
Data Width Configuration	ADD Marshar Data Due Mild	14	
	APB Master Data Bus Wid	th	
	32-bit	16-bit 🔘 8-bit	
Address Configuration			
Number of address bits driver	by master:	28	
Position in slave address of u	oper 4 bits of master address:	[27:24] (Ignored if master add	tress width >= 32 bits)
Indirect Addressing:	(Not in use	
Allocate memory space to combine	d region slave		
Slot 0:	Slot 1:	Slot 2:	Slot 3:
Slat 4:	Slot 5:	Slot 6:	Slot 7:
30(4.	500.5.	Side d.	
Slot 8:	Slot 9:	Slot 10:	Slot 11:
Slot 12:	Slot 13:	Slot 14	Slot 15:
300 12.	301 13.	30014	SIGT 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 10:	Slot 11:
Slot 12:	Slot 13:	Slot 14:	Slot 15:
Help 🔻			OK Cancel

5. Double-click **CoreAPB3** to configure it. Figure 20 shows the **Configuring CoreAPB3_0** window.

Figure 20 • 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.

Figure 21 shows the PCIe_Demo_top in SmartDesign after configuring CoreAHBLite and CoreAPB3 bus interfaces.



Figure 21 • CoreAHBLite and CoreAPB3 Bus Interfaces in PCIe_Demo_top SmartDesign Canvas After Configuration



Instantiating CoreGPIO in PCIe_Demo_top SmartDesign

To instantiate CoreGPIO in the PCIe_Demo_top SmartDesign,

1. Expand the Peripherals category in the Libero SoC Catalog as displayed in Figure 22.



Figure 22 • IP Catalog

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



guration Global Configuration			
APB Data Win	dth: 32	Numbe Output	r of I/Os: 8
I/O bit 0			
Ouput on Reset: 0 💌	Fixed Config: 📝	I/O Type: Both 💌	Interrupt Type: Disabled
I/O bit 1			
Ouput on Reset: 0 💌	Fixed Config: 🔽	I/O Type: Both 🔻	Interrupt Type: Disabled
I/O bit 2			
Ouput on Reset: 0 💌	Fixed Config: 🔽	I/O Type: Both 💌	Interrupt Type: Disabled
I/O bit 3			
Ouput on Reset: 0 💌	Fixed Config: 📝	I/O Type: Both 🔻	Interrupt Type: Disabled
I/O bit 4			
Ouput on Reset: 0 💌	Fixed Config: 📝	I/O Type: Both 💌	Interrupt Type: Disabled
I/O bit 5			
Ouput on Reset: 0 💌	Fixed Config: 🔽	I/O Type: Both 👻	Interrupt Type: Disabled
I/O bit 6			
Ouput on Reset: 0 💌	Fixed Config: 🔽	I/O Type: Both 💌	Interrupt Type: Disabled
I/O bit 7			
Ouput on Reset: 0 💌	Fixed Config: 📝	I/O Type: Both 🔻	Interrupt Type: Disabled

3. Double-click **CoreGPIO** to configure it. Figure 23 shows **Configuring CoreGPIO_0** window.

Figure 23 • Configuring CoreGPIO_0

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



Instantiating CoreAHBLSRAM in PCIe_Demo_top SmartDesign

To instantiate CoreAHBLSRAM in the PCIe_Demo_top SmartDesign,

1. Expand the **Memory & Controllers** category in the Libero SoC **Catalog** as displayed in Figure 24.



Figure 24 • IP Catalog

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



3. Double-click COREAHBLSRAM_0 to configure it. Figure 25 shows **Configuring COREAHBLSRAM_0** window.

Configuring COREAHBLSRAM_0 (C	OREAHBLSRAM - 2.1.102)
Configuration	
AHB Data Width:	32 •
AHB Address Width:	32 🔹
Select SRAM Type	
LSRAM	O uSRAM
LSRAM Depth	
Number of bytes of men	nory: 32768
uSRAM Depth	
Number of bytes of men	10ry: 128
Testbench:	User
Help	OK Cancel

Figure 25 • 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 32768.
- 4. 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,

1. Expand the **Clock & Management** category in the Libero SoC **Catalog**. Figure 26 shows Libero Catalog.



Figure 26 • Catalog

2. Drag Clock Conditioning Circuit (CCC) to the PCIe_Demo_top SmartDesign canvas. If the component appears shadowed in the **Vault**, right-click the name and select download.



3. Double-click CCC to configure it. Figure 27 shows the FAB CCC Configurator window.



- Select Reference Clock as 50 MHz and FPGA Fabric Input 0 from the drop-down list.

– Select GL0 Frequency as 100 MHz.

4. Click OK to save and close the FAB CCC Configurator window.





Figure 28 shows PCIe_Demo_top in SmartDesign after configuring all components.

Figure 28 • 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 28. 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 28. 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 29.



Figure 29 • Quick Connect Window

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

1. Expand SDIF0_PINS of PCIe_Demo_sb_0 and make connections as shown in Table 2.

Table 2 • SDIF0_PINS

From PCIe_Demo_sb_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_sb_0 and make connections as shown in Table 3.

Table 3 • INIT_PINS

From PCIe_Demo_sb_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_sb_0 to all resets as shown in Table 4.

Table 4 •	HPMS	READY	Connections
14010 1			••••••••

From PCIe_Demo_sb_0	То
HPMS_READY	HRESETN of CoreAHBLite_0,COREAHBTOAPB3_0, and COREAHBLSRAM_0
	PRESETN of CoreGPIO_0

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

Table 5 • GL0 Clock Connections

From FCCC_0	То			
GL0				
	PCLK of CoreGPIO_0			
	CLK of DEBOUNCE_0			
	CLK_BASE of SERDES_IF_0			

- 7. Expand FAB_CCC_PINS of PCIe_Demo_sb_0:
 - Right-click the FAB_CCC_GL0 and select Mark Unused.
 - Right-click the FAB_CCC_GL3 and select Mark Unused.
 - Right-click the FAB_CCC_LOCK and select Mark Unused.
- 8. Connect POWER_ON_RESET_N of PCIe_Demo_sb_0 to RESET_N of DEBOUNCE_0.
- 9. Right-click the SDIF_READY of PCIe_Demo_sb_0 and select Mark Unused.
- 10. Right-click the FAB_RESET_N of PCIe_Demo_sb_0 and select Tie high.
- 11. Expand PCI_Demo_HPMS_0_PINS.
 - Right-click the **COMM_BLK_INT** of PCIe_Demo_sb_0 and select **Mark Unused**.
 - Right-click the HPMS_INT_M2F[15:0] of PCIe_Demo_sb_0 and select Mark Unused.
- 12. Connect SDIF0_INIT_APB of PCIe_Demo_sb_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 AHBSlaveInterface 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 30 displays the **Edit Slice** window.





24. Select 2 slices of width 4, click Add Slices, and edit the window as shown in Figure 31.







- 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 32.





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 33.

Edit Slices - PCIE_INTERRU	IPT[3:0]			
Create 4 • slices of width 1 • Add Slices				
PCIE_INTERRUPT[3:0]	Left Right			
	1 0 0			
	2 3 1			
Help	ОК	Cancel		



3

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.




33. Click **Auto arrange instances** to arrange the instances and click **File > Save**. The PCIe_Demo_top is displayed as shown in Figure 34.

Figure 35 • Generate Component



35. The message PCIe_Demo_top was generated is displayed in the Libero SoC Log window if the design is generated without any errors. The Log window is displayed as shown in Figure 36 on successful component generation.

g	đΧ	Message
🗏 Messages 😵 Errors 🗼 Warnings 🌒 Info		🔳 Messages 😵 Errors 🔺 Warnings info 🗏 Manage suppressed messages
Uncir PAD pin "SERDES IF_0:FAD=QUT" of "SERDES IF_0" is automatically connected to a top-level port. Øinfo: The connection is being dropped from the net "CLK0_FAD_0" because the pin "CLK0_FAD" does not exist. Øinfo: The connection is being dropped from the net "CLK0_FAD" because the pin "CLK0_FAD" does not exist. Øinfo: "FCL0_Emen" was successfully generated. Øinfo: "FCL0_Emen" was successfully generated. Øinfo: "FCL0_Emen" was successfully generated. Øinfo: "FCL0_Emen" mainfest file 'D:/Kicrosemi_proj/1/FCLE_Demo/component/work/FCLe_Demo/PCLe_Demo_manifest.txt' was successfully generated.	*	Message Message ID Source Location Log Location ⊡ Design Entry ⊟ SmartDesign Check

Figure 36 • Log Window

Step 2: Developing the Simulation Stimulus

During the design process, SERDESIF is 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 as the IGLOO2 PCIe interface is a fully hardened and verified interface. This section describes how to modify the BFM script (user.bfm) file that is 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 37.



Figure 37 • SmartDesign Generated SERDESIF_0_user.bfm File



2. Modify the **SERDESIF** 0 user.bfm to add the following bfm commands of writing and reading:

memmap GPIO 0x4000000; memmap LSRAM 0x2000000; procedure main; # add your BFM commands below: wait 500us; wait 500us; write w GPIO 0xA0 0x00; write w GPIO 0xA0 0x01; write w GPIO 0xA0 0x02; write w GPIO 0xA0 0x04; write w GPIO 0xA0 0x08; write w GPIO 0xA0 0x10; write w GPIO 0xA0 0x20; write w GPIO 0xA0 0x40; write w GPIO 0xA0 0x40;

write w LSRAM 0x00 0x12345678; write w LSRAM 0x04 0x87654321; write w LSRAM 0x08 0x9ABCDEF0; write w LSRAM 0x0C 0x0FEDCBA9; readcheck w LSRAM 0x00 0x12345678; readcheck w LSRAM 0x04 0x87654321; readcheck w LSRAM 0x08 0x9ABCDEF0; readcheck w LSRAM 0x0C 0x0FEDCBA9; 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



The modified BFM file appears similar to the file shown in Figure 38.





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 <code>SERDESIF_0_user.bfm</code> file under the *<Libero project>/simulation* folder contains the BFM commands to verify the read/write access to CoreGPIO and CoreAHBLSRAM. The following steps describe how to use the SmartDesign testbench and the BFM script file to simulate the design:

1. 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_11p6_DF\Source Files. Figure 39 shows the wave.do file under simulation folder in the Files window.



Figure 39 • 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 **205µs**, as shown in Figure 40.

Device selection Device settings		Save
Design flow	Simulation runtime: 205us	Restore Defaults
Analysis operating conditions Simulation options	Testbench module name: testbench	
DO file	Top level instance name: <top>_0</top>	
Waveforms	Generate VCD file	
Vsim commands Timercale	WCD file name:	
 Simulation libraries 	Select Verilon Januare syntax	
IGLO02	In Norther 2001	
COREAHBLITE_LIB		
COREAHBLORAMI_LIB	System remog	
COREAPB3_LIB	Select VHDL language syntax	
	VHDL 2008	
	User defined DO file:	
	DO command parameters:	

Figure 40 • Project Setting – Do File Simulation Runtime Setting

- 5. Click Save.
- 6. Select Waveforms under Simulation Options as shown in Figure 41.

Project settings	-				• X
Device selection Device settings Design flow Analysis operating conditions Simulation options DO file Weveforms Vsim commands Timescale Simulation libraries I GIOO2 COREAHBLITE LIB COREAHBLISRAM_LIB COREAHBLISRAM_LIB COREAHBLISRAM_LIB COREAHBLISR	✓ Include D0 file wave.do Display waveforms for top_level testbench ✓ Log all signals in the design			Restore D	e Defaults
Help	_				Close

Figure 41 • Project Setting – Waveform

- Select the Include Do check box and select the file.
- Select the Log all signals in the design check box.
- 7. Click Close to close the Project Settings dialog box.



8. Click Save when prompted to save the changes.

Generating Testbench

1. Go to File > New > HDL Test bench.



Figure 42 • HDL Testbench

2. Select HDL Type as Verilog or VHDL and enter testbench as the Name.

Create New HDL Testbench File
HDL Type Verilog VHDL Name: testbench Clock Period (ns) : 100
 ✓ Initialize file with standard template ✓ Instantiate Root Design ✓ Set as Active Stimulus Help OK Cancel

Figure 43 • Create New HDL Testbench File

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 **205µs**. The ModelSim transcript window displays the BFM commands and the BFM simulation completed with no errors, as shown in Figure 44.



Figure 44 • SERDES BFM Simulation

Figure 45 shows the waveform window with GPIO_OUT signals.



Figure 45 • Simulation Result with GPIO_OUT Signals



Step 4: Generating the Program File

The following steps describe how to generate the program file:

- 1. Click File > Import > Timing Constraints (SDC) files to add the Timing Constraints file to the PCIe demo design.
- 2. Browse to the PCIe_Demo_New.sdc file location in the design files folder: M2GL_PCIE_Control_Plane_11p6_DF\Source Files.
- 3. Click Yes in Information window as shown in Figure 46.

Information	x
Do you want to organize constraint file(s) for your o	current root (PCIe_Demo) for (Compile)?
Yes No	
Figure 46 • New Figure Information window	



4. Double-click **I/O Constraints** in the Design Flow window as shown in Figure 47. The I/O Editor window is displayed after completing Synthesize and Compile.



Figure 47 • I/O Constraints

5. 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	K1			
GPIO_IN[0]	L19			
GPIO_IN[1]	L18			
GPIO_IN[2]	K21			
GPIO_IN[3]	K20			
GPIO_OUT[0]	E1			
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			



e	<u>E</u> dit <u>V</u> iew <u>T</u> ools	<u>H</u> elp						
-"		👳 👫 👫	1A					
_	Commit and Che	ck						
P	Dert Name +	Direction	I/O Standard	Din Number 💌	Locked 💌	Magra Call	Pank Nama 💌	
-		Input	IVCMOS25	Fin Number				
	DEVRST N	Input		R15			Daliko	
	GPIO INI01	Input	IVCM0S25	119	V		Bank2	
	GPIO IN[1]	Input	IVCMOS25	118			Bank2	
	GPIO IN[2]	Input	LVCMOS25	K21			Bank2	-
	GPIO IN[3]	Input	LVCMOS25	K20		ADLIB:INBUF	Bank2	
	GPIO OUTIOI	Output	LVCMOS25	El	V	ADLIB:OUTBUF	Bank7	
	GPIO OUT[1]	Output	LVCMOS25	F4	v	ADLIB:OUTBUF	Bank7	-
	GPIO OUT[2]	Output	LVCMOS25	F3	v	ADLIB:OUTBUF	Bank7	-
0	GPIO_OUT[3]	Output	LVCMOS25	G7	V	ADLIB:OUTBUF	Bank7	
1	GPIO_OUT[4]	Output	LVCMOS25	H7	V	ADLIB:OUTBUF	Bank7	=
2	GPIO_OUT[5]	Output	LVCMOS25	J6		ADLIB:OUTBUF	Bank7	
3	GPIO_OUT[6]	Output	LVCMOS25	H6		ADLIB:OUTBUF	Bank7	
4	GPIO_OUT[7]	Output	LVCMOS25	H5		ADLIB:OUTBUF	Bank7	
5	(P) REFCLK0_P	Input	LVDS	U1	V	ADLIB:INBUF_DIFF	Bank5	
6	(N) REFCLK0_N	Input	LVDS	TI		ADLIB:INBUF_DIFF	Bank5	
7	RXD0_N	Input		Y1		ADLIB:SERDESIF_0		
8	RXD0_P	Input		W1		ADLIB:SERDESIF_0		
9	RXD1_N	Input	-	Y3		ADLIB:SERDESIF_0		
0	RXD1_P	Input	-7	W3		ADLIB:SERDESIF_0		
1	RXD2_N	Input		Y5		ADLIB:SERDESIF_0		
2	RXD2_P	Input		W5		ADLIB:SERDESIF_0		
3	RXD3_N	Input		Y7		ADLIB:SERDESIF_0		
4	RXD3_P	Input		W7	V	ADLIB:SERDESIF_0		
5	SDIF0_PERST_N	Input	LVCMOS25	P18	V	ADLIB:INBUF	Bank2	
6	SWITCH	Input	LVCMOS25	J18	V	ADLIB:INBUF	Bank2	

After assigning the pins, the **I/O Editor** is displayed as shown in Figure 48.

Figure 48 • 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
- 6. After updating the I/O editor, click **Commit and Check**.
- 7. Close the I/O editor.
- 8. Click Verify Timing to complete place and route, and verify timing.
- 9. Click Generate Bitstream as shown in Figure 49 to generate the programming file.



Figure 49 • Generate Programming Data

Step 5: Programming the IGLOO2 Board Using FlashPro

The following steps describe how to program the IGLOO2 board using Flashpro:

- 1. Connect the FlashPro4 programmer to the J5 connector of the IGLOO2 Evaluation Kit board.
- 2. Connect the jumpers on the IGLOO2 FPGA Evaluation Kit board as shown in Table 7. **CAUTION:** Ensure to switch off **SW7** on the board While connecting the jumpers.

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

Table 7 • IGLOO2 FPGA Evaluation Kit Jumper Settings

- 3. Connect the power supply to the **J6** connector.
- Power ON the power supply switch, SW7. Refer to the "IGLOO2 Evaluation Kit Board" section for further details.
- 5. To program the IGLOO2 device, double-click **Run PROGRAM Action** in the **Design Flow** window as shown in Figure 50 on page 50.





Figure 50 • Run PROGRAM Action



Step 6: Connecting the Evaluation Kit to the Host PC

The following steps describe how to connect the IGLOO2 Evaluation Kit to the host PC:

- 1. After successful programming, power off the IGLOO2 Evaluation Kit board and shut down the host PC.
- 2. Use the following 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 use a laptop, the express card adapters typically support only Gen1 and the design works on Gen1 mode.
- Note: Host PC or laptop must 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 must be powered off during the PCIe card insertion.

Figure 51 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 51 • 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 1-52.
- 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 1-62.



Running the Design on Windows

The following steps describe how to run the design on Windows:

- 1. Switch **ON** the **SW7** power supply switch.
- Power on the host PC and check the host PC Device Manager for PCIe device as shown in Figure 52. If the PCIe device is not detected, power cycle the IGLOO2 Evaluation Kit board and click scan for hardware changes in the Device Manager.

📇 Devic	e Manager		
<u>F</u> ile <u>A</u>	ction <u>V</u> iew <u>H</u> elp		
<	🖬 🛛 🖬 👪		
⊿ 🚑 v	v7-donthus		
Þ	퇻 Computer		
Þ	🗃 Disk drives		
Þ	🖥 Display adapters		
Þe	DVD/CD-ROM drives		
⊳ 0	🛱 Human Interface Devices		
Þ	IDE ATA/ATAPI controllers		
⊳⊘	Keyboards		
Þ	Mice and other pointing device	s	
⊳ I	Monitors		
Þ	Network adapters		
a - [Other devices		
	PCI Device		
⊳.×	Ports (COM & LPT)		
Þ	Processors		
⊳∎	Sound, video and game control	lers	
<u>⊳ d</u>	System devices		
⊳ (Universal Serial Bus controllers		

Figure 52 • Device Manager

Note: If the device is still not detected, check if 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 53. The DEVICE uninstall window is displayed.

- Device Manager		
Eile Action View	Halp	
A NI - I		
🖉 👍 w7-techsuprt-0	2	
👂 🚛 Computer		
👂 🧫 Disk drives		
👂 🖳 Display ada	pters	
p 🔮 DVD/CD-RC)M drives	
D Construction IDE ATA/AT	API controllers	
Jungo		
DEVI	Update Driver Software	
⊳ - Keyboar	Disable	
🕨 🖉 Mice an	Uninstall	
Monitor	Scan for hardware changes	
Network	Scarror naraware enanges	
Ports (C	Properties	
Sound vide	o and game controllers	
System devi	ices	
🔰 🖡 🗍 Universal Se	rial Bus controllers	
Uninstalls the driver for	the selected device.	

Figure 53 • Device Uninstall

b. Select the **Delete the driver software** for this device check box as shown in Figure 54. After uninstalling the previous Jungo drivers, ensure that the PCI Device is detected in the **Device Manager** window as shown in Figure 54.



Figure 54 • 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 PCIe_Demo.rar to the C:\ drive. The PCIe_Demo.rar is located in the provided design files: M2GL_PCIE_Control_Plane_11p6_DF\Windows_64bit\Drivers\PCIe_Demo.rar.
- Note: Installing these drivers require the 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 55.



Figure 55 • 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 56.



Figure 56 • 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. Download the PCIe_Demo_GUI_Installer.rar from the below link:

http://soc.microsemi.com/download/rsc/?f=PCIe_Demo_GUI_Installer

 Double-click the setup.exe in the provided GUI installation (*PCIe_Demo_GUI_Installer\setup.exe*). Apply default options as shown in Figure 57.

PCIe Demo	
Destination Directory Select the primary installation directory.	
All software will be installed in the following locations. To install software into a different locations, click the Browse button and select another directory. Directory for PCIe Demo C:\Program Files\PCIe Demo\ Browse	
Directory for National Instruments products C:\Program Files\National Instruments\ Browse	
<< <u>Back</u> Next >> <u>Cancel</u>	

Figure 57 • GUI Installation

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

PCIe	Demo			
Ch	The installer has finished updating your system.			
		<< <u>B</u> ack	<u>N</u> ext >>	Finish

Figure 58 • Successful GUI Installation

4. Restart the host PC.



Running the PCIe GUI

 Check the host PC Device Manager for the drivers. Ensure 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	
🔺 📇 w7-donthus	
> 📲 Computer	
🖻 🥁 Disk drives	
🔈 📲 Display adapters	
DVD/CD-ROM drives	
🔈 🥼 Human Interface Devices	
De ATA/ATAPI controllers	
🖉 🔛 Jungo	
DEVICE	
> Can Keyboards	
Mice and other pointing devices	
Monitors	
Network adapters	
Ports (COM & LPT)	
Sound, video and game controllers	
System devices	
🖉 💭 💭 Universal Serial Bus controllers	

Figure 59 • 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 60.

© PCIe Demo Ver 11.8		
🛇 Microsemi.	PCIe Demo	
Tue, Mar 18, 2014 11:59:59 AM	Link Width: 4x Gen 1 Rate 2.5G	L
PCIe Link Info 🗧 🔶		
Demo Controls		
Config Space		2
PCIe R/W		
Security		
DMA Operations		
Fabric DMA		
DMA_SMC_FIC		
PCIe Link	EXIT	
igure 60 • PCle Demo GUI		
\sim		



3. Click the **Connect** button at the top-right corner of the GUI. The messages are displayed on the GUI as shown in Figure 61.



Figure 61 • Version Information

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



4. Clicking **Demo Controls** in the GUI displays the LED options and DIP switch positions as shown in Figure 62.

> PCIe Demo Ver 11.8		
📀 Microsem	Ī	PCIe Demo
Tue, Mar 18, 2014 4:50:07 PM	Link Width: 1x	Gen 2 Rate 5G Board IGLOO2 Eval
PCIe Link Info		
	D 1	Interrupt Counter* ON ON ON ON
Demo Controls	D 2	
Config Space LE	D 3	Enable Interrupt Session
LE	D 4	OFF OFF OFF OFF
PCIe R/W LE	D 5	Clear/Disable Interrupts
LE	D 6	
LE	D 7	Start LED ON/OFF Walk
LE	D 8	Stop LED ON/OFF Walk
	*NOTE: PRESS APPR	ROPRIATE PUSH BUTTON SWITCH TO TOGGLE THE INTERRUPT COUNTER
PCIe Link		EXIT IGLOO2

Figure 62 • Demo Controls

- 5. Click LEDs in GUI to ON/OFF the LEDs on the IGLOO2 Evaluation Kit board.
- 6. Click Start LED ON/OFF Walk to make the LEDs on the IGLOO2 Evaluation Kit board blink.
- 7. Click Stop LED ON/OFF Walk to stop the LEDs blinking.
- 8. Change the DIP switch positions on the IGLOO2 Evaluation Kit board (**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 board and observe the interrupt count on the **Interrupt Counter** field in the GUI as shown in Figure 63.



Figure 63 • 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 64 shows the PCIe configuration space.



Figure 64 • Configuration Space



Click PCle R/W to perform read and writes to LSRAM memory through BAR1 space. Figure 65 shows the PCle R/W window. Enter the address in the Address field between 0x0000 to 0x7FFC. The Data field accepts a 32-bit hexadecimal value.



Figure 65 • Perform Read and Write to LSRAM Using PCIe

14. Click Exit.

Running the Design on Linux

The following steps describe how to run 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 ${\tt lspci}$ command to display the PCIe info.
 - # lspci



	root@localhost:~	
le <u>E</u> dit <u>V</u> iew <u>T</u> erminal Ta <u>b</u> s <u>H</u> elp		
oot@localhost ~]# lspci		
:00.0 Host bridge: Intel Corporation 4 Series Chipset D	RAM Controller (rev 03)	
:01.0 PCI bridge: Intel Corporation 4 Series Chipset PC	E Express Root Port (rev 03)	
:02.0 VGA compatible controller: Intel Corporation 4 Se	ries Chipset Integrated Graphics Controller (rev 03)	
:02.1 Display controller: intel corporation 4 Series Ch	Lpset Integrated Graphics Controller (rev 03)	
10.0 Audio device: Intel Corporation N10/ICH / Family N	Figh Definition Audio Controller (rev 01)	
:1d & USP controller: Intel Corporation N10/ICH 7 Family FC.	(USB_UHCT_Controllor #1 (rov 01)	
:1d 1 USB controller: Intel Corporation N10/ICH 7 Family	/ USB UHCI Controllor #2 (rev 01)	
:1d 2 USB controller: Intel Corporation N10/ICH 7 Family	/ USB UHCI Controller #2 (rev 01)	
:1d.3 USB controller: Intel Corporation N10/ICH 7 Family	/ USB UHCI Controller #4 (rev 01)	
:1d.7 USB controller: Intel Corporation N10/ICH 7 Family	/ USB2 EHCT Controller (rev 01)	
:1e.0 PCI bridge: Intel Corporation 82801 PCI Bridge (r	ev el)	
:1f.0 ISA bridge: Intel Corporation 82801GB/GR (ICH7 Fa	nily) LPC Interface Bridge (rev 01)	
:1f.1 IDE interface: Intel Corporation 82801G (ICH7 Fam:	ily) IDE Controller (rev 01)	
:1f.2 IDE interface: Intel Corporation N10/ICH7 Family	SATA Controller [IDE mode] (rev 01)	
:1f.3 SMBus: Intel Corporation N10/ICH 7 Family SMBus C	ontroller (rev 01)	
:00.0 Non-VGA unclassified device: Actel Device 11aa		
:00.0 Ethernet controller: Broadcom Corporation NetLink	BCM57780 Gigabit Ethernet PCIe (rev 01)	
oot@localhost ~]#		
		_

Figure 66 • PCIe Device Detection

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_11p6_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_11p6_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

- 5. 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

Applications Places System 🔗 🛞 🖉 🋜	🕢 5:43 PM 🐠
root@localhost:/home/igi2/inc	_ • ×
Elle Edit View Terminal Tabs Help	
/** * SF2: SmartFusion2 Board, IGL2: IGL002 Board * #define SF2, if the hardware board is SmartFusion2 * #define IGL2, if the hardware board is IGL002 */ #define IGL2 #undef SF2	
🕐 🔲 root@localhost:/home/igl2/inc 🛛 🕲 Outlook Web App - Mozilla Firefox	

Figure 67 • Edit board.h file



- 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 insmod command.



Figure 68 • 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_11p6_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.

- 3. 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



	root@l	iocalhost:/home/igi2/PCie_App		
Eile Edit ⊻iew Terminal	a <u>b</u> s <u>H</u> elp			
[root@localhost PCIe_App /home/ig12/PCIe_App [root@localhost PCIe_App led_blink.sh pcie_appln [root@localhost PCIe_App [root@localhost PCIe_App [root@localhost PCIe_App [root@localhost PCIe_App	<pre># pwd # ls ctrlpln.c pcie_config.sh # <u>gcc -o pcie_ctrlplane pcie_appln_ctrlplr</u> # ls ctrlpln.c pcie_config.sh pcie_ctrlplane # _/pcie_ctrlplane</pre>	1.C		
Description: LED Control Example: ./pcie_ctrlplan	[1], Led Data [0x0-0x000000FF] 1 0x000000FF		•	
Description: SRAM_WRITE Example: ./pcie_ctrlplan	2], Write[1], SRAM Data [0x0-0xFFFFFFF], 2 1 0x12345678 0x10	SRAM Offset[0x0-0x3FFF]		
Description: SRAM_READ [Example : ./pcie_ctrlpla], Read[0], SRAM Offset[0x0-0x1FFF] e 3 0 0x10			
Description: Dip Switch : Example: ./pcie_ctrlplan	tatus [4] 4			
Description: PCIe Device Example: ./pcie_ctrlplan	Info [5], Display PCIe Configuration Space 5 1	e/PCIe Device Detailed Info[1/2]		
Description: PCIe Interr Example: ./pcie_ctrlplan	pt Control [6], Enable/(Clear/Disable)[1/0 6 1	9]		
Description: PCIe Interr Example: ./pcie_ctrlplan	pt Count [7] 7			
[root@localhost PCIe_App	# []			
剩 🔲 root@localhost:/hom	e/ial2/PCIe App			

Figure 69 • 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 69.



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 70 • 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 board.

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

A Applications Places System 🔗 🚳 🚭 🎯	8:16 PM 🕥)
root@localhost:/home/igl2/PCle_App	_ • ×
Elle Edit View Jerminal 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 2 1 0x12345678 0x1000 [root@localhost PCIe_App]# ./pcie_ctrlplane 3 0 0x1000 SF2 SRAM data Read : 0x12345678 [root@localhost PCIe_App]#]</pre>	
	Ξ
Toot@localhost:/home/igl2/PCle_App	

Figure 71 • Linux Command - SRAM Read/Write

3



DIP Switch Status

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

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

	root@localhost:/home/igl2/PCle_App	
le <u>E</u> dit ⊻iew <u>T</u> erminal Ta <u>b</u> s <u>H</u> elp		
oot@localhost PCIe_App]# ./pcie_ctrlplane		
scription: LED Control [1], Led Data [0x0-0x06 ample: ./pcie_ctrlplane 1 0x000000FF	00000FF]	
scription: SRAM_WRITE [2], Write[1], SRAM Data ample: ./pcie_ctrlplane 2 1 0x12345678 0x10	a [0x0-0xFFFFFFFF], SRAM Offset[0x0-0x3FFF]	
<pre>scription: SRAM_READ [3], Read[0], SRAM Offset ample : ./pcie_ctrlplane 3 0 0x10</pre>	[0x0-0x1FFF]	
scription: Dip Switch Status [4] ample: ./pcie_ctrlplane 4		
scription: PCIe Device Info [5], Display PCIe ample: ./pcie_ctrlplane 5 1	Configuration Space/PCIe Device Detailed Info[1/2]	
scription: PCIe Interrupt Control [6], Enable, ample: ./pcie_ctrlplane 6 1	(Clear/Disable)[1/0]	
scription: PCIe Interrupt Count [7] ample: ./pcie_ctrlplane 7		
oot@localhost PCIe_App]# ./pcie_ctrlplane 4 P Switch Data Register Value : 0x200 1 : ON		
2 : OFF		
4 : ON		
oot@localhost PCIe_App]# []		
	6	
I a root@localhost:/home/igl2/PCle_App		
ure 72 • Linux Command - DIP	Switch	



PCIe Configuration Space Display

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

Root Privileges are required to execute this command.

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

Applications Places System 🔗 🎯 🍣 👔 🚳	: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 [0x0-0x00000FF] 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	
<pre>1.VID 0x1laa Vendor Id 2.DID 0x1laa Device ID 3.CM0 0x0406 Command 4.STS 0x0010 Status 5.RID_CLO 0x0000 Revision ID & Class Code 6.SCC 0x00 Sub Class Code 7.RCC 0x00 Base Class Code 8.CALN 0x10 Cache Line Size 9.LAT 0x00 Latency Timer 10.HDR 0x00 Header Type 11.BIST 0x00 Base Adress 0 12.BADDR0 0x7e500000 Base Adress 1 14.BADDR1 0x7e470000 Base Adress 1 14.BADDR5 0x00000000 Base Adress 3 15.BADDR4 0x00000000 Base Adress 3 16.BADDR4 0x00000000 Base Adress 5 18.CIS 0x00000000 Base Adress 5 18.CIS 0x00000000 Base Adress 5 18.CIS 0x00000000 Expansion ROM Base Adress 4 17.BADDR5 0x00000000 Expansion ROM Base Adress 5 18.CIS 0x0000000 Expansion ROM Base Adress 7 22.REW_CAP 0x50 New Capabilities Pointer 23.INTIN 0x00 Interrupt Line 24.INTPIN 0x01 Interrupt Line 25.MINGMT 0x00 Maximu Latency [root@localbost PCLE_App]# □</pre>	
💌 🔄 root@localhost:/home/igi2/PCIe_App	9

Figure 73 • 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//gl2/PCle_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]	_
	-
See Toot@localhost:/home/igl2/PCle_App	9

Figure 74 • Linux Command - PCle Link Speed and Width

3



	root@localhost:/home/igl2/PCle_App	
e <u>E</u> dit ⊻iew	Terminal Tabs Help	
Kernel	driver in use: i801_smbus	
Kernel	modules: 12c-1801	
AA A Non VG	A unclassified device: Actel Device 1122	
Subsyst	A unclassified device: Actet Device find	
Control	(; I/O- Mem+ BusMaster+ SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR- FastB2B- DisINTx+	
Status	: Cap+ 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <tabort- <mabort-="">SERR- <perr- intx-<="" td=""><td></td></perr-></tabort->	
Latency	y: 0, Cache Line Size: 64 bytes	
Interru	upt: pin A routed to IRQ 74	
Region	0: Memory at feedbook (32-bit, non-prefetchable) [512e=1M]	
Capabil	1. Memory at revision (32-51), non-prefericitate, [512=04K]	
	Address: 00000000fee00000 Data: 404a	
Capabi	lities: [78] Power Management version 3	
	Flags: PMEClk- DSI- D1- D2- AuxCurrent=0mA PME(D0+,D1-,D2-,D3hot+,D3cold-)	
	Status: D0 NoSoftRst+ PME-Enable- DSel=0 DScale=0 PME-	
Capabi	lities: [80] Express (v2) Endpoint, MSI 01	
	Devia): Maxayload 250 bytes, maintrunc 0, Latency Los untimited, Li untimited Exitage AttnRin, AttnInd, BREF El Resol.	
	DevCl: Report errors: Correctable- Non-Fatal+ Fatal+ Unsupported-	
	RlxdOrd+ ExtTag+ PhantFunc- AuxPwr- NoSnoop+	
	MaxPayload 128 bytes, MaxReadReq 512 bytes	
	DevSta: CorrErr- UncorrErr- FatalErr- UnsuppReq- AuxPwr- TransPend-	
	LnkCap: Port #1, Speed 5GT/s, Width x4, ASPM L0s L1, Latency L0 <64ns, L1 <16us	
	CLOCKPM+ Surprise- LLACTREP- BWNOt-	
	EINCLL ASH DISADLED, RED dy bytes Disabled- Retrain- community	
	LnkSta: Speed 2.5GT/s, Width x4, TrErr- Train- SlotClk+ DLActive- BWMgmt- ABWMgmt-	
	DevCap2: Completion Timeout: Range ABCD, TimeoutDis-	
	DevCtl2: Completion Timeout: 50us to 50ms, TimeoutDis-	/
	LnkCtl2: Target Link Speed: 5GT/s, EnterCompliance- SpeedDis-, Selectable De-emphasis: -6dB	
	Transmit Margin: Normal Operating Range, EnterModifiedCompliance- ComplianceSOS-	
	Compliance Detemphasis, -oub	
Capabi	Lindstall (190 v1) Virtual Channel	
	Caps: LPEVC=0 RefClk=100ns PATEntryBits=1	
	Arb: Fixed- WRR32- WRR64- WRR128-	
	Ctrl: ArbSelect=Fixed	
	Status: InProgress-	
	VUC: Caps. FAIOIISEL=00 MAXIMESICUS=1 REJSHOOPTAHS-	
	Ctrl: Enable+ ID=0 ArbSelect=Fixed TC/VC=01	
	Status: NegoPending- InProgress-	
Capabi	lities: [800 vl] Advanced Error Reporting	
	UESta: DLP- SDES- TLP- FCP- CmpltTO- CmpltAbrt- UnxCmplt- RxOF- MalfTLP- ECRC- UnsupReq- ACSViol-	
	UEMSK: DLP- SDES- TLP- FCP- CmpltT0- CmpltAbrt- UNXCmplt- RXOF- MalTLP- ECRC- UnsupReq- ACSVIOL-	
	UESVIC: DEFF SDESF IEFF FCF (mpttto- unpttable on Kumpte RKOF Matthewetck) on Suppred- ACSVIC: (FSta: RXFr- BadTle BadTle Ballover. Timeout- NonFatalFr-	
	CEMsk: RXErr- BadTLP- BadDLLP- Rollover- Timeout- NonFata/Err+	
	AERCap: First Error Pointer: 00, GenCap+ CGenEn- ChkCap+ ChkEn-	
Kernel	driver in use: MS_PCI_DRIVER	
0.0 Etherne	et controller: Broadcom Corporation NetLink BCM57780 Gigabit Ethernet PCIe (rev 01)	

Figure 75 • Linux Command - PCle Link Speed and Width



PCIe Interrupt Control (Enable/Disable) and Interrupt Counter

IGLOO2 Evaluation Kit board enables or disables 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]



Figure 76 • Linux Command - PCle Interrupt Control

Conclusion

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


Appendix A: IGLOO2 Evaluation Kit Board



Figure 1 • IGLOO2 Evaluation Kit Board



Appendix 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.



Figure 1 • Lining up the IGLOO2 Evaluation Kit Board

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



TU0509: Implementing PCIe Control Plane Design in IGLOO2 FPGA - Libero SoC v11.6



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



List of Changes

3

The following table shows the important changes made in this document for each revision.

Date	Changes	Page
Revision 7 (October 2015)	Updated the document for Libero v11.6 software release (SAR 72421).	NA
Revision 6 (February 2015)	Updated the document for Libero v11.5 software release (SAR 63980).	NA
Revision 5 (August 2014)	Updated the design files link under "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



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

For Microsemi SoC Products Support, visit http://www.microsemi.com/products/fpga-soc/design-support/fpga-soc-support

Website

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



TU0509: Implementing PCIe Control Plane Design in IGLOO2 FPGA - Libero SoC v11.6

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. Visit About Us for sales office listings and corporate contacts.

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@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

© 2015 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.

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for communications, defense & 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; Etherent Solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, Calif., and has approximately 3,600 employees globally. Learn more at **www.microsemi.com**.

Microsemi makes no warranty, representation, or guarantee regarding the information contained herein or the suitability of its products and services for any particular purpose, nor does Microsemi assume any liability whatsoever arising out of the application or use of any product or circuit. The products sold hereunder and any other products sold by Microsemi have been subject to limited testing and should not be used in conjunction with mission-critical equipment or applications. Any performance specifications are believed to be reliable but are not verified, and Buyer must conduct and complete all performance and other testing of the products, alone and together with, or installed in, any end-products. Buyer shall not rely on any data and performance specifications or parameters provided by Microsemi. It is the Buyer's responsibility to independently determine suitability of any products and to test and verify the same. The information provided by Microsemi hereunder is provided "as is, where is" and with all faults, and the entire risk associated with such information is entirely with the Buyer. Microsemi does not grant, explicitly or implicitly, to any party any patent rights, licenses, or any other IP rights, whether with regard to such information itself or anything described by such information. Information provided in this document is proprietary to Microsemi, and Microsemi reserves the right to make any changes to the information in this document or to any products and services at any time without notice.