IGLOO2 M2GL090T/TS and SmartFusion2 M2S090T/TS Device

High Speed Serial Interface Configuration

Libero SoC v11.6 and later

Microsemi®
Power Matters.™

Table of Contents

Introduction

The High Speed Serial Interface block in the IGLOO2 M2GL090T/TS and SmartFusion2 M2S090T/TS devices (Figure 3) provides multiple high speed serial protocols, such as PCIe end-point and XAUI. In addition, it enables the FPGA fabric to connect with the External Physical Coding Sublayer (EPCS) interface and implement any user defined protocol in the fabric.

Two SERDES IP cores are available:

- High Speed Serial Interface 2 - suppports 2 protocols: 1 and 2
- High Speed Serial Interface 3 - supports 3 protocols: 1, 2 and 3

As you make selections in the core configurator, it automatically narrows down the subsequent choices and defaults. Only the relevant ports appear in the generated macro.

This document describes how to configure a High Speed Serial Interface instance and define how the signals are connected. For more details about the High Speed Serial Interface, refer to the SmartFusion2 or IGLOO2 High Speed Serial Interfaces User's Guide.

To access the High Speed Serial Interface Configurator:

1. Instantiate the High Speed Serial Interface 2 core or the High Speed Serial Interface 3 core from the Catalog into the SmartDesign Canvas, as shown in Figure 1 and Figure 2.

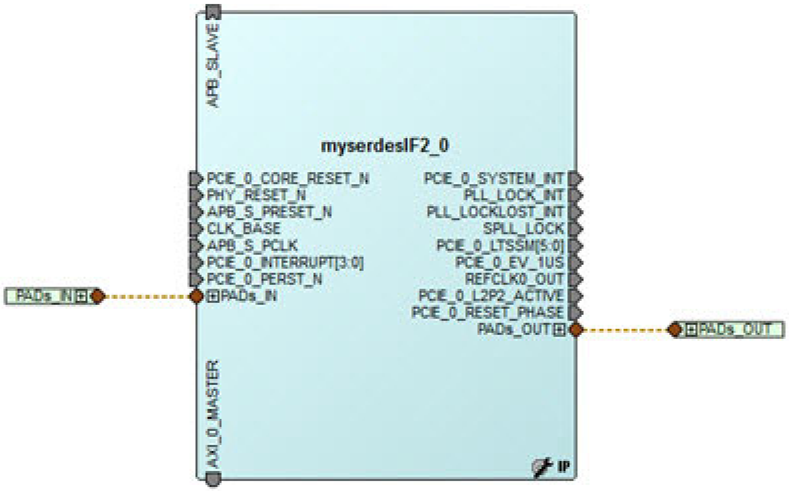


***Figure 1* • SERDES IF2 Block Instantiation on the SmartDesign Canvas**

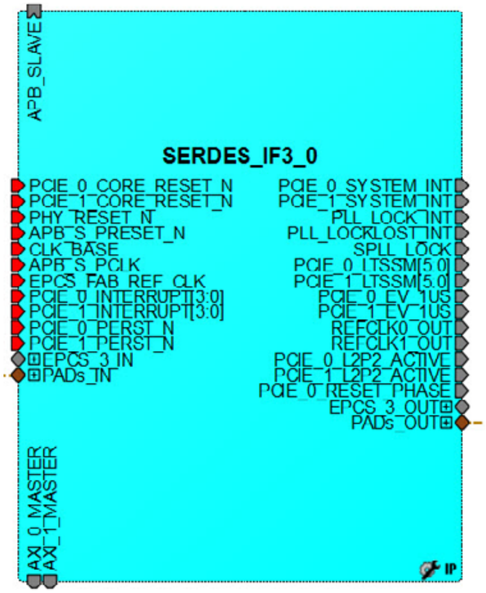
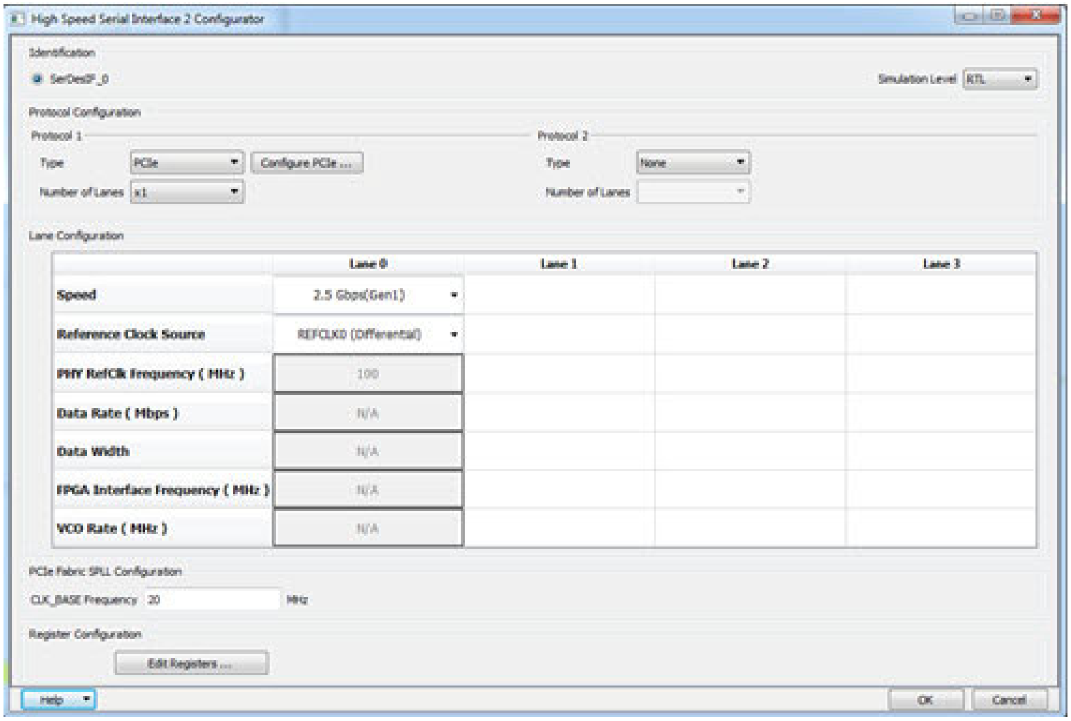


Figure 2 • **SERDES IF3 Block Instantiation on the Smart Design Canvas**

2. Double-click the SERDES block on the Canvas to open the Configurator (Figure 3 and Figure 4).



Identification

● SerDesIF_0

Simulation Level RTL

Protocol Configuration

Protocol 1

Type PCIe

Number of Lanes x1

Protocol 2

Type None

Number of Lanes

Lane Configuration

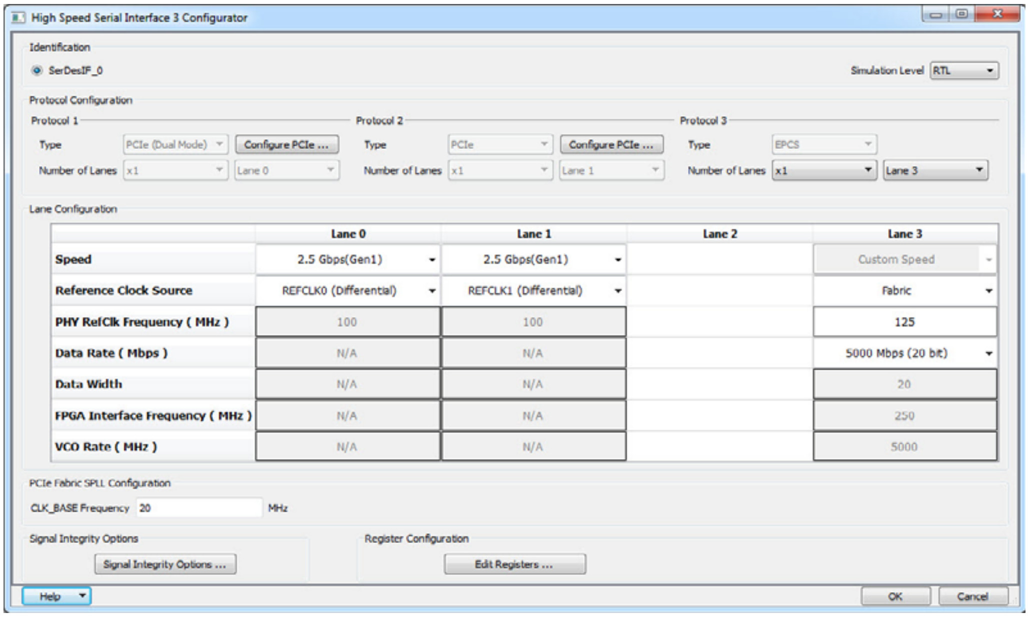
	Lane 0	Lane 1	Lane 2	Lane 3
Speed	2.5 Gbps(Gen1)			
Reference Clock Source	REFCLK0 (Differential)			
PHY RefClk Frequency (MHz)	100			
Data Rate (Mbps)	N/A			
Data Width	N/A			
FPGA Interface Frequency (MHz)	N/A			
VCO Rate (MHz)	N/A			

PCIe Fabric SPLL Configuration

CLK_BASE Frequency 20 MHz

Register Configuration

Figure 3 • **High Speed Serial Interface 2 Configurator**



High Speed Serial Interface 3 Configurator

Identification

● SerDesIF_0

Simulation Level: RTL

Protocol Configuration

Protocol 1
- Type: PCIe (Dual Mode)
- Number of Lanes: x1, Lane 0

Protocol 2
- Type: PCIe
- Number of Lanes: x1, Lane 1

Protocol 3
- Type: EPCS
- Number of Lanes: x1, Lane 3

Lane Configuration

	Lane 0	Lane 1	Lane 2	Lane 3
Speed	2.5 Gbps(Gen1)	2.5 Gbps(Gen1)		Custom Speed
Reference Clock Source	REFCLK0 (Differential)	REFCLK1 (Differential)		Fabric
PHY RefClk Frequency (MHz)	100	100		125
Data Rate (Mbps)	N/A	N/A		5000 Mbps (20 bit)
Data Width	N/A	N/A		20
FPGA Interface Frequency (MHz)	N/A	N/A		250
VCO Rate (MHz)	N/A	N/A		5000

PCIe Fabric SPLL Configuration

CLK_BASE Frequency: 20 MHz

Signal Integrity Options

Register Configuration

Figure 4 • **High Speed Serial Interface 3 Configurator**

1 – Functionality

Identification

IGLOO2 M2GL090T/TS and SmartFusion2 M2S090T/TS devices contain one High Speed Serial Interface block. Under the Identification Heading, SERDESIF_0 is selected by default to identify the SERDES block you are configuring.

Protocol Configuration

For High Speed Serial Interface 2 core, you must configure the Type and Number of Lanes for each of the two protocols.

For High Speed Serial Interface 3 core, you cannot configure the protocol type and lane configuration. They are fixed as follows;

- Protocol 1: PCIe One Lane (Lane 0), at either Gen1 or Gen2 speed
- Protocol 2: PCIe One Lane (Lane 1), at either Gen1 or Gen2 speed
- Protocol 3: EPCS One Lane (Lane 2 or Lane 3) or Two Lanes (Lane 2 and Lane 3) at Custom speed.

High Speed Serial Interface 3 core gives you three protocols but you have less flexibility to configure. High Speed Serial Interface 2 core gives you two protocols only but allows more flexibility for configuration.

Protocol 1 and Protocol 2

Select your Protocol type from the drop-down menu:

- PCIe
- PCIe (Reverse)
- XAUI
- EPCS

When you select the Protocol PCI or PCIe Reverse, you must click the Configure PCIe button to configure additional options for SERDES in PCIe mode. See "Configure PCIe" on page 7 for details.

Note: You must Configure Protocol 1 before configuring Protocol 2.

Protocol 2 Types are context sensitive; they depend on the options you have selected in Protocol 1.

Protocol 2 Type selection is disabled when you select XAUI in Protocol 1. It is activated only when PCIe, PCIe Reverse or EPCS is selected for Protocol 1 and you use less than four lanes. Refer to Table 1-1 for the Protocol 1 and Protocol 2 configuration combinations.

Number of Lanes

Select the number of lanes you wish to configure for Protocol 1 from the drop-down menu:

- X1 - Configure for 1 lane
- X2 - Configure for 2 lanes
- X4 - Configure for all 4 lanes

Note: Items in the drop-down list are context sensitive and depend on the Protocol Type.

If Protocol Type is XAUI, all four lanes are selected by default.

Protocol 1 and 2: Type, Number of Lanes, Speed

Table 1-1 shows the protocol combinations that are feasible within a single High Speed Serial Interface block.

Table 1-1 • Available Protocols

Protocol Type	Protocol #	Lane Width	Lane Assignment	Description	Speed Choices
PCIe	Protocol 1	x1	Lane 0		Gen1 (2.5 Gbps), Gen2 (5.0 Gbps)
		x2	Lane 0, Lane 1		
		x4	Lane 0, Lane 1, Lane 2, Lane 3		
	Protocol 2	x1	Lane 2	Available only when PCIe or PCIe Reverse is selected for Protocol 1	
		x2	Lane 2, Lane 3		
PCIe Reverse	Protocol 1	x1	Lane 1 or Lane 3	If Protocol 2 is used, it is Lane 1; otherwise, it is Lane 3	Gen1 (2.5 Gbps), Gen2 (5.0 Gbps)
		x2	Lane 0, Lane 1 or Lane 2, Lane 3	If Protocol 2 is used, it is Lanes 0 and 1; otherwise, it is Lanes 2 and 3	
		x4	Lane 0, Lane 1, Lane 2, Lane 3		
	Protocol 2	x1	Lane 3	Available only when PCIe or PCIe Reverse is selected for Protocol 1	
		x2	Lane 2, Lane 3		
XAUI	Protocol 1	x4	Lane 0, Lane 1, Lane 2, Lane 3		3.125 Gpbs
EPCS	Protocol 1	x1	Users can select Lane 0, 1, 2 or 3		Custom Speed
		X2	Lane 0, Lane 1		
		x4	Lane 0, Lane 1, Lane 2, Lane 3		
	Protocol 2	x1	Users can select Lane 2 or 3		
		x2	Lane 2, Lane 3		

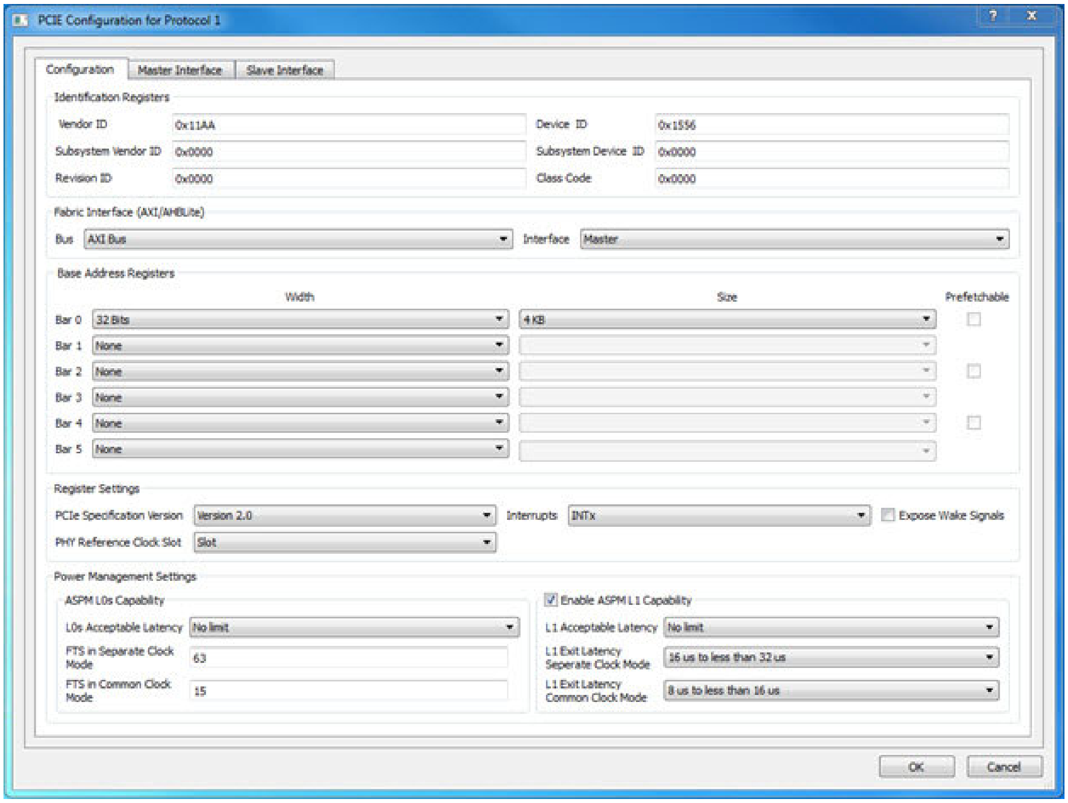
The PCIe GEN 2 (5.0 Gbps) speed is not available in STD speed-grade devices. Refer to your device datasheet (SmartFusion2 Datasheet; IGLOO2 Datasheet) for more information.

Configure PCIe

The Configure PCIe button appears only when you select the PCIe or PCI Reverse Protocol.

Configuration

The Configuration tab sets your Identification Registers, Fabric Interface, Base Address Registers and Options (Figure 1-1).



PCIE Configuration for Protocol 1

Configuration | Master Interface | Slave Interface

Identification Registers

Vendor ID	0x11AA	Device ID	0x1556
Subsystem Vendor ID	0x0000	Subsystem Device ID	0x0000
Revision ID	0x0000	Class Code	0x0000

Fabric Interface (AXI/AHBLite)

Bus: AXI Bus Interface: Master

Base Address Registers

	Width	Size	Prefetchable
Bar 0	32 Bits	4 KB	□
Bar 1	None		
Bar 2	None		□
Bar 3	None		
Bar 4	None		□
Bar 5	None		

Register Settings

PCIe Specification Version: Version 2.0 Interrupts: INTx □ Expose Wake Signals

PHY Reference Clock Slot: Slot

Power Management Settings

ASPM L0s Capability

L0s Acceptable Latency: No limit

FTS in Separate Clock Mode: 63

FTS in Common Clock Mode: 15

☑ Enable ASPM L1 Capability

L1 Acceptable Latency: No limit

L1 Exit Latency Seperate Clock Mode: 16 us to less than 32 us

L1 Exit Latency Common Clock Mode: 8 us to less than 16 us

Figure 1-1 • PCIe Configuration - Configuration Tab

Identification Registers

There are six identification registers for PCIe that you can assign 16-bit hexadecimal signatures.

- **Vendor ID** - 0x11AA is the Vendor ID assigned to Microsemi by PCI-SIG. Contact Microsemi if you would like to allocate Subsystems under the Microsemi vendor ID.
- **Subsystem vendor ID** - Card manufacturer’s ID.
- **Device ID** - Manufacturer’s assigned part number assigned by the vendor.
- **Revision ID** - Revision number, if available.
- **Subsystem Device ID** - Assigned by the subsystem vendor.
- **Class Code** - PCIe device's generic function.

Fabric Interface

Use this field to configure the Bus Standard (AXI or AHBLite) and the Interface (Master Only, Slave Only, or Master and Slave) for the PCIe protocol. You can select the Bus Standard and the Interface Mode for both Protocol 1 and Protocol 2, separately and independently. Dual-PCIe Mode operation is supported.

In PCIe mode, the SERDES block can act as an AXI or AHBLite Master.

You must instantiate a COREAXI or CoreAHBLite Bus into the SmartDesign Canvas and then connect the Master and/or Slave Bus Interface (BIF) of the SERDES to the Master and/or Slave BIF of the COREAXI bus or COREAHBLite bus.

Base Address Registers

The individual fields of the six Base Address Registers (Bar 0 through Bar 5) can be configured as follows:

- **Width** - The width on even registers can be 32 bits or 64 bits. If an even register is set to 64 bits wide, the subsequent (odd) register serves as the upper half of 64 bits. The width of odd registers is restricted to 32 bits.
- **Size** - Ranges from 4 KB to 2 GB when the Width is 32 bit. Some devices support only up to 1 GB. When the Width is 64 bit, the Size can range from 2 KB to 16 EB. Refer to your device datasheet (SmartFusion2 Datasheet; IGLOO2 Datasheet) for more information.
- **Prefetchable** - Enabled only on even registers with 64-bit width.

Options

Options enables you to configure the following:

- **PHY Reference Clock Slot** - Sets your reference clock to Slot or Independent.
- **Expose Wake Signals** - Click the checkbox to add PCIE_WAKE_N, PCIE_WAKE_REQ and PCIE_PERST_N ports.
- **PCIe Specification Version** - Sets the Specification Version to 1.0, 1.1 or 2.0.
- **Interrupt** - Sets the Interrupt to:
  - MSI 1
  - MSI 2
  - MSI 4
  - MSI 8
  - MSI 16
  - MSI 32
  - INTx

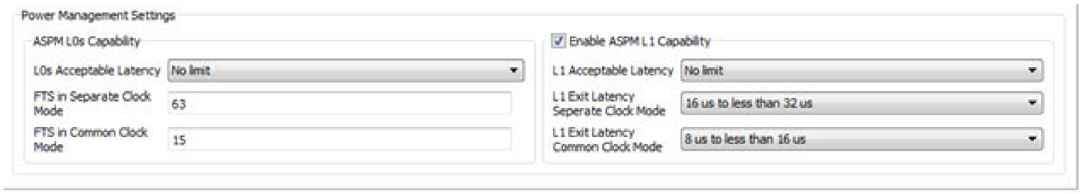
Your Interrupt selection sets bit 16 of the PCIE_MSI_CTRL_STATUS register and bits [19:17] of the PCIE_MSI_CTRL_STATUS register as shown in Table 1-2 below.

Table 1-2 • MSI and Register Settings

Interrupt Selected	Setting for Bit 16 of PCIE_MSI_CTRL_STATUS Register	Setting for Bits [19:17] of PCIE_MSI_CTRL_STATUS Register
MSI 1	1	000
MSI 2	1	001
MSI 4	1	010
MSI 8	1	011
MSI 16	1	100
MSI 32	1	101
INTx	0 (Disable MSI)	000

Power Management Settings

Use the Power Management Settings to configure the ASPM (Active State Power Management) settings. The Configurator sets the correct register values for your SERDES block based on the selections you have made.



Power Management Settings

ASPM L0s Capability

L0s Acceptable Latency: No limit

FTS in Separate Clock Mode: 63

FTS in Common Clock Mode: 15

☑ Enable ASPM L1 Capability

L1 Acceptable Latency: No limit

L1 Exit Latency Seperate Clock Mode: 16 us to less than 32 us

L1 Exit Latency Common Clock Mode: 8 us to less than 16 us

Figure 1-2 • Power Management Settings

ASPM L0s Capability

This is mandatory. Settings available are:

- **L0s Acceptable Latency** - Opens the pull-down list to choose one of the following latency values:
  - oMaximum of 64ns
  - Maximum of 128ns
  - Maximum of 256ns
  - Maximum of 512ns
  - Maximum of 1us
  - Maximum of 2us
  - Maximum of 4us
  - No Limit
- **FTS in Separate Clock Mode** - Enter the number of FTS (Fast Training Sequences) required in separate clock mode. Valid values are from 0 through 255.
- **FTS in Common Clock Mode** - Enter the number of FTS (Fast Training Sequences) required in common clock mode. Valid values are from 0 through 255.

ASPM L1 Capability

By default, the ASPM L1 Capability is enabled. Configure the settings for ASPM L1 as follows:

- **L1 Acceptable Latency** - Click the pull-down list to choose one of the of following for latency values:
  - Maximum of 1 us
  - Maximum of 2 us
  - Maximum of 4 us
  - Maximum of 8 us
  - Maximum of 16 us
  - Maximum of 32 us
  - Maximum of 64 us
  - No Limit
- **L1 Exit Latency Separate Clock/Common Clock Mode**- Click the pull-down menu to choose one of the following for the Exit Latency value of Separate/Common Clock Mode:
  - Less than 1 us
  - 1 us to less than 2 us
  - 2 us to less than 4 us
  - 4 us to less than 8 us

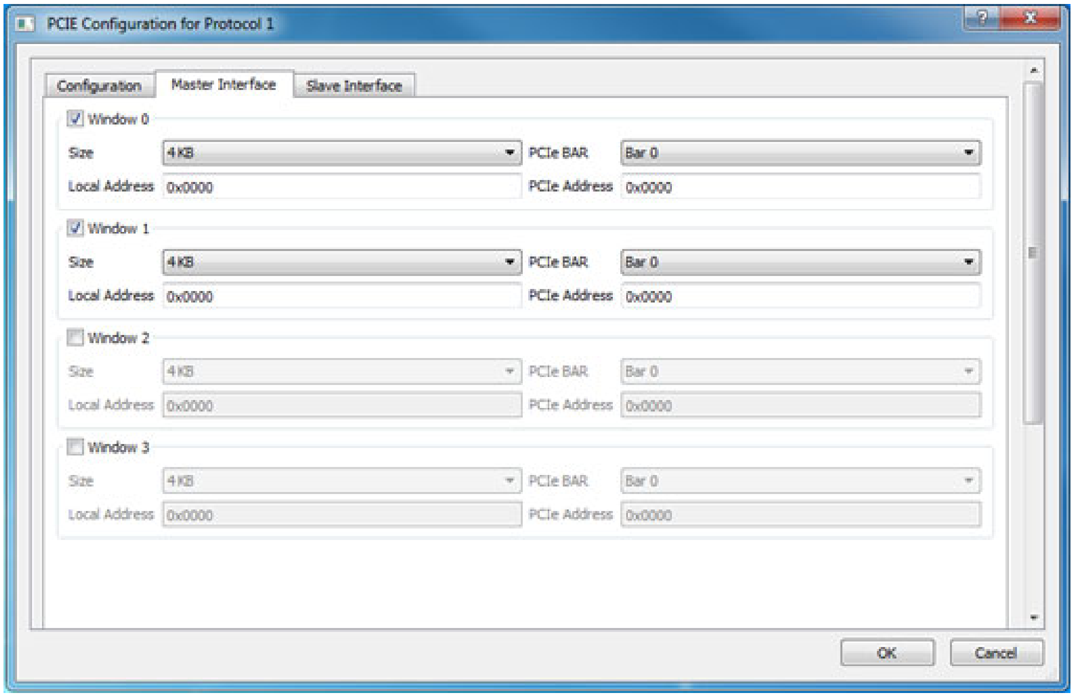
- 8 us to less than 16 us
- 16 us to less than 32 us
- 32 us to 64 us
- More than 64 us

Note: The Exit Latency value you choose for the Common Clock Mode must be smaller than the value for Separate Clock Mode.

Master Interface

PCIe/PCIe Reverse Protocol 1 or Protocol 2 enables you to use the Master Interface tab to configure up to four PCI windows (Window 0 through Window 3) with the following parameters (as shown in Figure 1-3):

- Size
- PCIe BAR (Base Address Register)
- Local Address
- PCIe Address



☑ Window 0
Size: 4 KB — PCIe BAR: Bar 0
Local Address: 0x0000 — PCIe Address: 0x0000

☑ Window 1
Size: 4 KB — PCIe BAR: Bar 0
Local Address: 0x0000 — PCIe Address: 0x0000

□ Window 2
Size: 4 KB — PCIe BAR: Bar 0
Local Address: 0x0000 — PCIe Address: 0x0000

□ Window 3
Size: 4 KB — PCIe BAR: Bar 0
Local Address: 0x0000 — PCIe Address: 0x0000

Figure 1-3 • PCIe Configuration - Master Interface Tab

Size

For each of the windows 0 through 3, select one of the available window sizes: 4KB, 8KB, 16KB, 32KB, 64KB, 128KB, 256KB, 512KB, 1MB, 2MB, 4MB, 8MB, 16MB, 32MB, 64MB, 128MB, 256MB, 512MB, 1GB and 2GB.

The default selection is 4KB. The size selected is mapped to bits [31:12] of WindowsX_1, where X can be 0, 1, 2, or 3.

PCIe BAR

Select one of the following BAR (Base Address Register):

- BAR0
- BAR1

- BAR2
- BAR3
- BAR4
- BAR5
- BAR0/1
- BAR2/3
- BAR4/5

Bar Size is mapped to bits [5:0] of WindowsX_2, where X can be 0, 1, 2 or 3, as shown in Table 1-3.

Table 1-3 • BAR Size and Corresponding Bit Settings

BAR Size	Bit Settings
BAR0, BAR0/1	0x01
BAR 1	0x02
BAR 2, BAR2/3	0x04
BAR 3	0x08
BAR 4, BAR4/5	0x10
BAR5	0x20

Local Address

Local Address is 20-bits wide and is mapped to bits [31:12] of Base address AXI Master WindowsX_0. The LSB bits [11:0] of AXI Master WindowsX_0 are reserved and the configurator will account for these bits. Do not include these reserved bits when you specify the local address.

PCIe Address

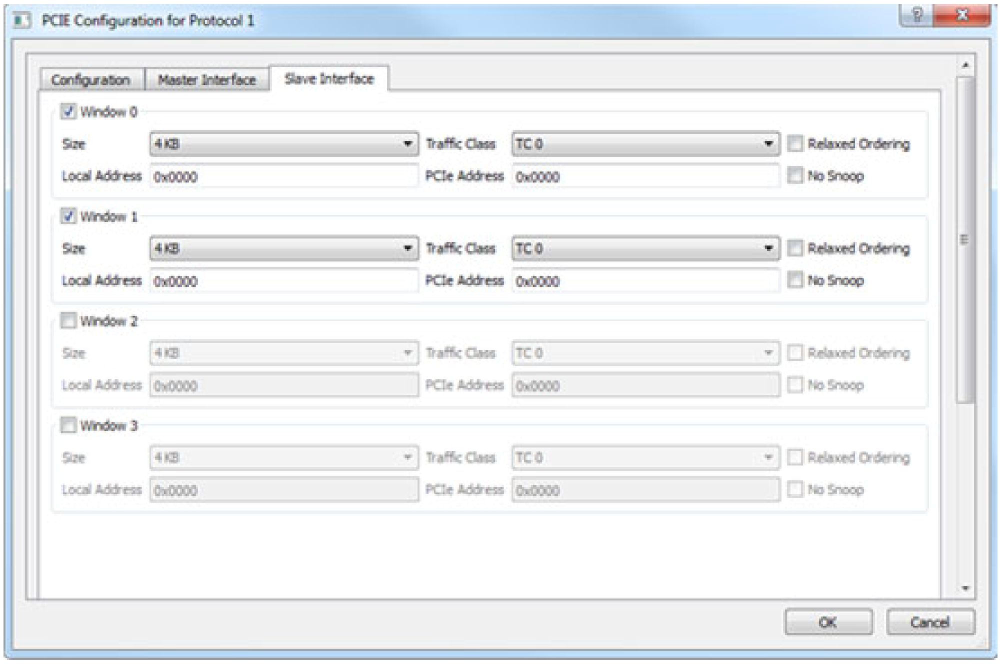
PCIe Address is mapped to bits [31:12] (LSB of Base address AXI Master WindowsX_2) and bits [31:0] (MSB of Base address AXI Master WindowsX_3).

Slave Interface

If you select PCIe or PCIe Reverse Protocol the Slave Interface tab enables you to configure up to four PCI windows, Window 0 through Window 3, with the following parameters (as shown in Figure 1-4 on page 13):

- Size
- Local Address
- PCIe Address
- Traffic Class: Selects the PCIe Traffic Class in the PCIe packet header.
- Relaxed Ordering: Enables you to generate the PCIe TLP using a selectable relaxed ordering bit.

- No Snoop: Enables you to generate the PCIe TLP using a selectable no snoop bit.



☑ Window 0
Size: 4 KB; Traffic Class: TC 0; □ Relaxed Ordering
Local Address: 0x0000; PCIe Address: 0x0000; □ No Snoop

☑ Window 1
Size: 4 KB; Traffic Class: TC 0; □ Relaxed Ordering
Local Address: 0x0000; PCIe Address: 0x0000; □ No Snoop

□ Window 2
Size: 4 KB; Traffic Class: TC 0; □ Relaxed Ordering
Local Address: 0x0000; PCIe Address: 0x0000; □ No Snoop

□ Window 3
Size: 4 KB; Traffic Class: TC 0; □ Relaxed Ordering
Local Address: 0x0000; PCIe Address: 0x0000; □ No Snoop

Figure 1-4 • PCIe Configuration - Slave Interface Tab

The Size, Local Address and PCIe Address options are the same as those for Master Interface. Refer to "Master Interface" on page 11 for more information.

Traffic Class

Traffic Class enables you to set your Traffic Class and corresponding register bits, as shown in Table 1-4. The Traffic Class value determines the relative priority of a given transaction as it traverses the PCIe link. Use this value to create a priority scheme for different packets.

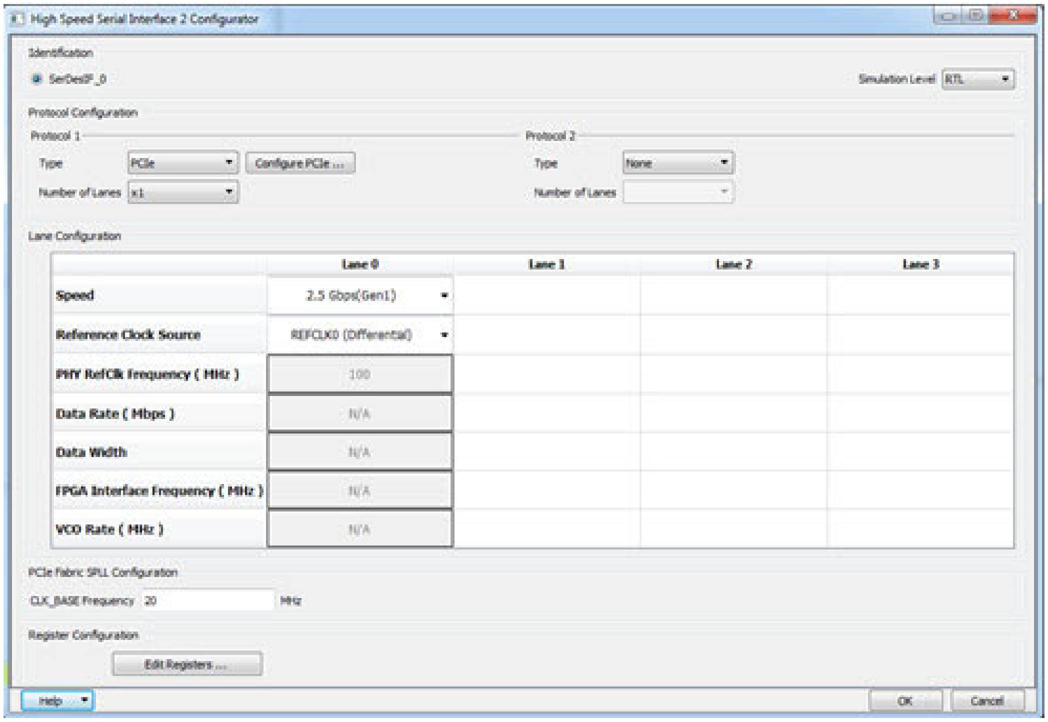
- TC 0 (Default)
- TC 1
- TC 2
- TC 3
- TC 4
- TC 5
- TC 6
- TC 7

Table 1-4 • Traffic Class and Corresponding Bit Setting

Traffic Class	Bit Setting for WindowX_2[4:2]
TC 0	000
TC 1	001
TC 2	010
TC 3	011
TC 4	100
TC 5	101
TC 6	110
TC 7	111

Lane Configuration

Use Lane Configuration to configure up to four lanes for your SERDES. The SERDES can be configured to run in dual-protocol mode. Refer to Table 1-1 on page 7 for lane configuration for dual mode operation.



High Speed Serial Interface 2 Configurator

Identification

● SerDesIF_0

Simulation Level: RTL

Protocol Configuration

Protocol 1

Type: PCIe

Number of Lanes: x1

Protocol 2

Type: None

Number of Lanes:

Lane Configuration

	Lane 0	Lane 1	Lane 2	Lane 3
Speed	2.5 Gbps(Gen1)			
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: 20 MHz

Register Configuration

Figure 1-5 • High Speed Serial Interface Configurator

Speed - Available selections depend on your selected Protocol. Refer to Table 1-1 on page 7 for the valid Speeds and available Protocols.

Reference Clock Source - Two clock sources are available: REFCLK0 and REFCLK1. Each can be differential or single-ended. You can select one of the following options for Protocol 1 and Protocol 2:

- REFCLK0 (Differential),
- REFCLK1 (Differential),
- REFCLK0 (Single-Ended),
- REFCLK1 (Single-Ended)
- Fabric (Available only for the EPCS Protocol)

Note: Lane 0 and Lane 1 share the same Reference Clock and Lane 2 and Lane 3 share the same Reference Clock. The selected Reference Clock is always available as REFCLK0_OUT or REFCLK1_OUT and can be used as clock source for logic inside Fabric.

PHY RefClk Frequency (MHz) - This is a fixed value for all protocols except EPCS Custom Speed in which case you can enter values between 100 and 160 MHz.

Data Rate (Mbps) - Read-only fixed value for all protocols except EPCS Custom Speed, in which case you can select the data rate from the drop-down list. Data Rates are computed based on the PHY RefClk Frequency.

Data Width - Read-only fixed value for all protocols except EPCS Custom Speed. For EPCS, the data width varies with Data Rate (Mbps) as follows:

- 20 bit (for 5000 Mbps and 2500 Mbps)
- 16 bit (for 4000 Mbps or 2000 Mbps)

- 10 bit (2500 Mbps or 1250 Mbps)
- 8 bit (for 2000 Mbps or 1000 Mbps)
- 5 bit for 1250 Mbps
- 4 bit (for 1000 Mbps)

The displayed value is computed and updated based on your selected PHY RefClk Frequency and Data Rate.

FPGA Interface Frequency (MHz) - Read-only fixed value for all protocols except EPCS Custom Speed, the displayed value is computed and updated based on the PHY RefClk Frequency and Data Rate you select.

VCO Rate (MHz) - Read-only fixed value for all protocols except EPCS Custom Speed. The displayed value is computed and updated based on the PHY RefClk Frequency and Data Rate you select.

PCIe/XAUI Fabric SPLL Configuration

The SPLL configuration fields are relevant only for PCIe and XAUI protocols (Figure 1-5). For the PCIe protocol, enter a valid value between 20 and 200 MHz for the CLK_BASE Frequency.

For the XAUI protocol, the CLK_BASE Frequency is read-only and fixed at 156.25 MHz.

Signal Integrity Options

For the XAUI and EPCS protocols, click S**ignal Integrity Options** to open the Signal Integrity Configuration dialog box.

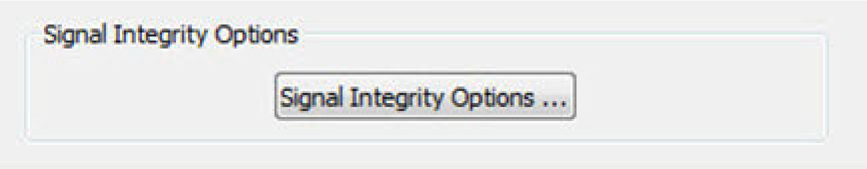
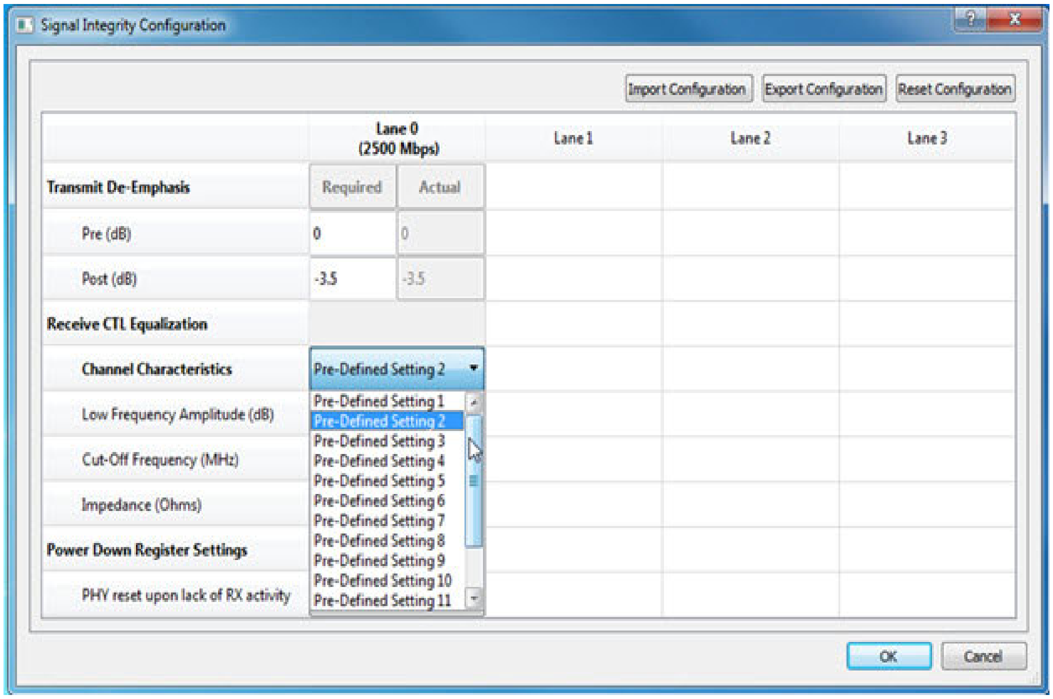


Figure 1-6 • Signal Integrity Options

The Signal Integrity Configuration dialog box gives you controls to maintain signal integrity and to mitigate signal integrity problems.



	Lane 0 (2500 Mbps)		Lane 1	Lane 2	Lane 3
Transmit De-Emphasis	Required	Actual			
Pre (dB)	0	0			
Post (dB)	-3.5	-3.5			
Receive CTL Equalization					
Channel Characteristics	Pre-Defined Setting 2				
Low Frequency Amplitude (dB)					
Cut-Off Frequency (MHz)					
Impedance (Ohms)					
Power Down Register Settings					
PHY reset upon lack of RX activity					

Figure 1-7 • Signal Integrity Configuration Dialog Box

Transmit De-Emphasis

Enter any value between 0.0 & 36.1 (in dB) in the **Required** edit box for both Pre-Transmit and Post-Transmit stage. Not all values are supported. Refer to Table 1-5 for all Actual values supported. The value you enter in the **Required** box will be matched to the closest valid Actual value and reported in the Actual box. The Configurator sets appropriate values for LANE<n>_TX_PRE_RATIO and LANE<n>_TX_PST_RATIO registers based on the Actual value.

LANE<n>_TX_PRE_RATIO and LANE<n>_TX_PST_RATIO registers are set based on the Actual value, as shown in Figure 1-5. The default value for **Required** is 0 for **Pre-Transmit** and 3.5 for **Post-Transmit**.

The LANE <n>_TX_AMP_RATIO lane register is always set to 0x80 for all lanes.

.

Table 1-5 • EPCS/XAUI Signal Integrity Configuration—Transmit De-Emphasis

Feature	Control Registers	Actual Value=value programmed in register
De-Emphasis Pre	LANE<n>_TX_PRE_RATIO	0dB = 0x0 0.1dB = 0x1 0.3dB = 0x2 0.4dB = 0x3 0.5dB = 0x4 0.7dB = 0x5 0.9dB = 0x6 1dB = 0x7 1.2dB = 0x8 1.3dB = 0x9 1.5dB = 0xa 1.6dB = 0xb 1.8dB = 0xc 2dB = 0xd 2.1dB = 0xe 2.3dB = 0xf 2.5dB = 0x10 2.7dB = 0x11 2.9dB = 0x12 3dB = 0x13 3.3dB = 0x14 3.5dB = 0x15 3.7dB = 0x16 3.9dB = 0x17 4dB = 0x18 4.3dB = 0x19 4.5dB = 0x1a 4.8dB = 0x1b 5dB = 0x1c 5.2dB = 0x1d 5.5dB = 0x1e 5.8dB = 0x1f 6dB = 0x20 6.3dB = 0x21 6.5dB = 0x22 7dB = 0x23 7.2dB = 0x24 7.5dB = 0x25 7.8dB = 0x26 8dB = 0x27 8.5dB = 0x28 9dB = 0x29 9.3dB = 0x2a 9.7dB = 0x2b 10.1dB = 0x2c 10.5dB = 0x2d 11dB = 0x2e 11.5dB = 0x2f 12dB = 0x30 12.6dB = 0x31 13.2dB = 0x32 13.8dB = 0x33 14.5dB = 0x34 15.2dB = 0x35 16.1dB = 0x36 17dB = 0x37 18dB = 0x38 19.2dB = 0x39 20.5dB = 0x3a 22.1dB = 0x3b 24dB = 0x3c 26.5dB = 0x3d 30.1dB = 0x3e 36.1dB = 0x3f
De-Emphasis Post	LANE<n>_TX_PST_RATIO	

Note: LANE<n> denotes the lane number where <n> can be 0, 1, 2 or 3.

For example, if you enter 2.4 dB in the Required box, 2.5dB (the closest match) is displayed in the Actual box and the registers are set as follows:

1. LANE<n>_TX_PRE_RATIO registers are set to 0x10
2. LANE<n>_TX_PST_RATIO registers are set to 0x10
3. LANE<n>_TX_AMP_RATIO registers are set to 0x80.

Receive CTL Equalization

There are 14 predefined settings available for user selection. Each predefined setting has the corresponding Cut-Off Frequency and Low Frequency Amplitude values preset and displayed when the predefined setting is selected. The impedance value for all predefined settings is 100 Ohms. Depending on the predefined setting, lane registers LANE<n>_RE_AMP_RATIO and LANE<n>_RE_CUT_RATIO are set to the values listed in Table 1-6.

Table 1-6 • CTL Equalization Predefined Settings and Register Values

Pre-defined Setting #	Cut-Off Frequency (MHz)	Low Frequency Amplitude (dB)	LANE<n>_RE_AMP_ RATIO (HEX)	LANE<n>_RE_CUT_ RATIO (HEX)
1	400	10.88	0x77	0x20
2	500	13.06	0x5B	0x2A
3	600	13.98	0x51	0x2F
4	700	12.04	0x4A	0x32
5	800	13.38	0x3E	0x3C
6	900	13.98	0x39	0x40
7	1000	12.57	0x70	0x4F
8	1100	11.6	0x7D	0x58
9	1200	10.88	0x37	0xFE
10	1300	9.95	0x22	0xFC
11	1400	8.52	0x52	0xFE
12	1500	7.47	0x5B	0xFE
13	1600	7.04	0x73	0xFE
14	1700	6.66	0x78	0xFE

Note: LANE<n> denotes the lane number where <n> can be 0, 1, 2, or 3.

Power Down Register Settings

This controls the Physical Reset behavior of the EPCS SERDES when there is a lack of RX activity.

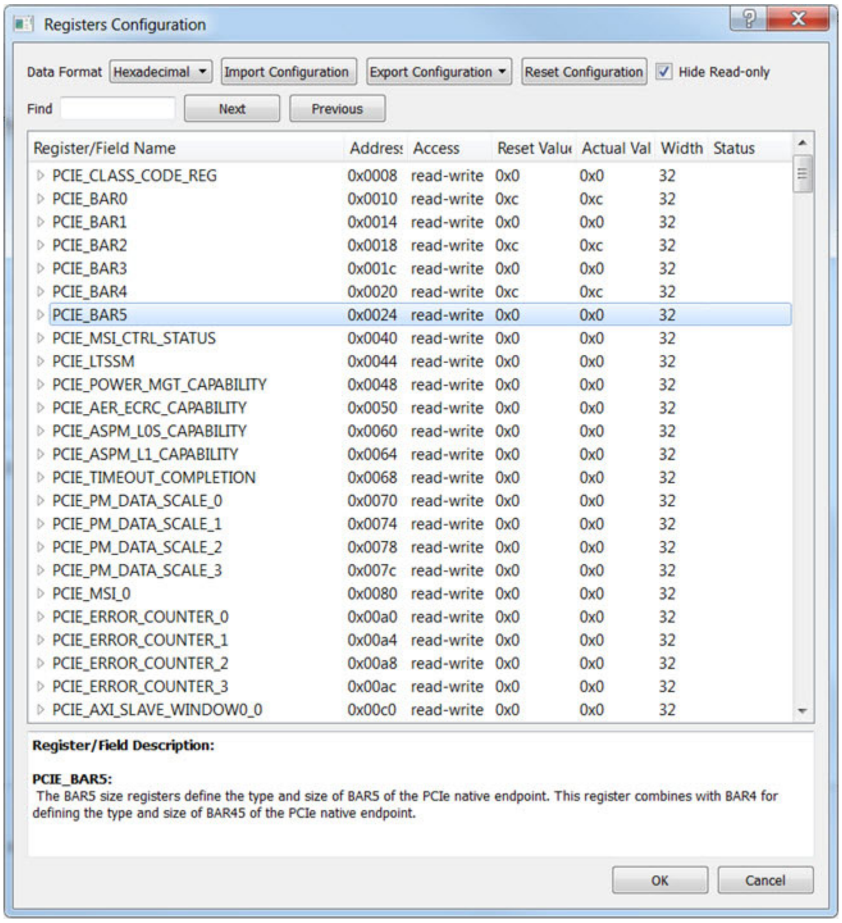
- Enabled (Physical Reset behavior is enabled)
- Disabled (Physical Reset behavior is disabled)

High Speed Serial Interface Control Registers

The High Speed Serial Interface has a set of registers that can be configured at runtime. The configuration values for these registers represent different parameters, for example, AXI BAR Window. For details about these registers, refer to the Microsemi UG0447: SmartFusion2 or IGLOO2 High Speed Serial Interfaces User's Guide.

High Speed Serial Interface Registers Configuration

To enter the High Speed Serial Interface configuration values, specify the register values when you are configuring the High Speed Serial Interface. Click **Edit Registers** in the High Speed Serial Interface Configurator (Figure 3 on page 4) to open the Registers Configuration dialog box (Figure 1-8). Data entered in this configurator is written at power up in the High Speed Serial Interface registers as described in the SmartFusion2 DDR Controller and Serial High Speed Controller Initialization Methodology document.



Data Format: Hexadecimal

☑ Hide Read-only

Register/Field Name	Addres	Access	Reset Valu	Actual Val	Width	Status
PCIE_CLASS_CODE_REG	0x0008	read-write	0x0	0x0	32	
PCIE_BAR0	0x0010	read-write	0xc	0xc	32	
PCIE_BAR1	0x0014	read-write	0x0	0x0	32	
PCIE_BAR2	0x0018	read-write	0xc	0xc	32	
PCIE_BAR3	0x001c	read-write	0x0	0x0	32	
PCIE_BAR4	0x0020	read-write	0xc	0xc	32	
PCIE_BAR5	0x0024	read-write	0x0	0x0	32	
PCIE_MSI_CTRL_STATUS	0x0040	read-write	0x0	0x0	32	
PCIE_LTSSM	0x0044	read-write	0x0	0x0	32	
PCIE_POWER_MGT_CAPABILITY	0x0048	read-write	0x0	0x0	32	
PCIE_AER_ECRC_CAPABILITY	0x0050	read-write	0x0	0x0	32	
PCIE_ASPM_L0S_CAPABILITY	0x0060	read-write	0x0	0x0	32	
PCIE_ASPM_L1_CAPABILITY	0x0064	read-write	0x0	0x0	32	
PCIE_TIMEOUT_COMPLETION	0x0068	read-write	0x0	0x0	32	
PCIE_PM_DATA_SCALE_0	0x0070	read-write	0x0	0x0	32	
PCIE_PM_DATA_SCALE_1	0x0074	read-write	0x0	0x0	32	
PCIE_PM_DATA_SCALE_2	0x0078	read-write	0x0	0x0	32	
PCIE_PM_DATA_SCALE_3	0x007c	read-write	0x0	0x0	32	
PCIE_MSI_0	0x0080	read-write	0x0	0x0	32	
PCIE_ERROR_COUNTER_0	0x00a0	read-write	0x0	0x0	32	
PCIE_ERROR_COUNTER_1	0x00a4	read-write	0x0	0x0	32	
PCIE_ERROR_COUNTER_2	0x00a8	read-write	0x0	0x0	32	
PCIE_ERROR_COUNTER_3	0x00ac	read-write	0x0	0x0	32	
PCIE_AXI_SLAVE_WINDOW0_0	0x00c0	read-write	0x0	0x0	32	

Register/Field Description:

PCIE_BAR5:
The BAR5 size registers define the type and size of BAR5 of the PCIe native endpoint. This register combines with BAR4 for defining the type and size of BAR45 of the PCIe native endpoint.

Figure 1-8 • High Speed Serial Interface Registers Configuration Dialog Box

Alternatively, you can click the Import Configuration button and import an existing configuration text file to configure the Registers.

The Registers Configuration dialog box enables you to enter High Speed Serial Interface register values using a graphical interface. The dialog box has the following features:

- **Registers Table** - Enter register values one-by-one using the Registers Table. To enter a register value, expand the register data tree (using the arrow sign), and click the **Actual Value** column to edit.
- **Import Configuration** - Import complete register configurations from text files. Register configuration syntax is shown below; Microsemi recommends using this method.
- **Export Configuration** - You can export the current register configuration data into a text file. The syntax of the exported file is the same as that of importable register configuration text files. For example:

```
PCIE_AXI_MASTER_WINDOW0_0                0x00000000
```

```
PCIE_AXI_MASTER_WINDOW0_1               0xfffff001
PCIE_AXI_MASTER_WINDOW0_2               0x00000000
PCIE_AXI_MASTER_WINDOW0_3               0x00000000
PCIE_AXI_MASTER_WINDOW1_0               0x00001000
PCIE_AXI_MASTER_WINDOW1_1               0xfffff001
PCIE_AXI_MASTER_WINDOW1_2               0x4
```

- **Reset Configuration** - Click Reset Configuration to undo any changes you have made to the register configuration. This deletes all register configuration data and you must either re-import or reenter this data. The data is reset to the hardware reset values.
- **Hide Read-Only Registers** - Enables you to show or hide the read-only registers in the Register Table. These registers are mostly status registers and do not contribute to the configuration.

When you generate your FPGA, the configuration register data entered in this configurator is used to initialize the High Speed Serial Interface simulation model when performing a BFM simulation.

Firmware (SmartFusion2 Only)

When you generate the SmartDesign, the following files are generated in the <project dir>/firmware/drivers_config/sys_config directory. These files are required for the CMSIS firmware core to compile properly and contain information regarding your current design, including peripheral configuration data and clock configuration information for the MSS. Do not edit these files manually; they are recreated every time your root design is regenerated.

- sys_config.c
- sys_config.h
- sys_config_SERDESIF_<0-3>.h - High Speed Serial Interface configuration data
- sys_config_SERDESIF_<0-3>.c - High Speed Serial Interface configuration data

Simulation Level

There are three levels of ModelSim simulation supported for the High Speed Serial Interface block depending on the selected protocol. See the SmartFusion2 FPGA High Speed Serial Interface Simulation User Guide for details.

BFM_CFG - This level provides a Bus Functional Model of only the APB configuration bus of the High Speed Serial Interface block. You will be able to write and read the different configuration and status bits from the High Speed Serial Interface block through its APB slave interface. The status bits value will not change based on the APB state; they are kept at their reset values. This simulation level is available for all protocols.

BFM_PCIe - This simulation level provides the BFM_CFG level plus the ability to communicate with the High Speed Serial Interface block through the master and slave AXI or AHB bus interfaces. Although no serial communication actually goes through the High Speed Serial Interface block, this scenario enables you to validate the fabric interface connections. This simulation level is only available for the PCIe protocol.

RTL - This simulation level enables you to fully simulate the High Speed Serial Interface block from the fabric interface to the serial data interface. This results in a longer simulation runtime. This simulation level is available for all protocols.

Simulation Files - SmartFusion2

When you generate the SmartDesign associated with your MSS, the following simulation files are generated in the <project dir>/simulation directory:

- **test.bfm** - Top-level BFM file, first executed during any simulation that exercises the SmartFusion2 MSS' Cortex-M3 processor. It executes peripheral_init.bfm and user.bfm, in that order.

- **peripheral_init.bfm** - Contains the BFM procedure that emulates the CMSIS::SystemInit() function run on the Cortex-M3 before you enter the main() procedure. It copies the configuration data for any peripheral used in the design to the correct peripheral configuration registers, and then waits for all the peripherals to be ready before asserting that the user can use these peripherals.
- **SERDESIF_<0>_init.bfm** - Contains BFM write commands that simulate writes of the High Speed Serial Interface configuration register data you entered (using the Edit Registers dialog box above) into the High Speed Serial Interface registers.
- **SERDESIF_0_PCIE_0_user.bfm** - Intended for user commands that simulate transactions being initiated off-chip (via the SERDES interface). You can simulate the datapath by adding your own commands in this file. Commands in this file will be "executed" after peripheral_init.bfm has completed. This user BFM file is for PCIe Protocol 1
- **SERDESIF_0_PCIE_1_user.bfm** - Intended for user commands that simulate transactions being initiated off-chip (via the SERDES interface). You can simulate the datapath by adding your own commands in this file. Commands in this file will be "executed" after peripheral_init.bfm has completed. This user BFM file is for PCIe Protocol 2.
- **user.bfm** - Intended for user commands. You can simulate the datapath by adding your own commands in this file. Commands in this file will be "executed" after peripheral_init.bfm has completed.
- **subsystem.bfm** - Contains the memory map. You do not have to modify this file. The Base addresses of AMBA slaves connected to the MSS via recognized AMBA buses in your design can be found here. This includes Fabric peripherals (connected to the MSS via the FICs) as well as MSS (Hard) peripherals. These files are automatically passed to ModelSim by Libero.

Using the files above, the configuration path is simulated automatically. You only need to edit the user.bfm, SERDESIF_0_PCIE_0_user.bfm and SERDESIF_0_PCIE_1_user.bfm files to simulate the datapath. Do not edit the test.bfm, peripheral_init.bfm, or SERDESIF_<0>_init.bfm files as these files are recreated every time your root design is regenerated.

Simulation Files - IGLOO2

When you generate the SmartDesign associated with your HPMS, the following simulation files are generated in the <project dir>/simulation directory:

- **ENVM_init.mem** - In IGLOO2 designs, the configuration data for any peripheral used in the design is stored in the ENVM_init.mem file. The IGLOO2 simulation library uses this file. Libero SoC creates this file for the simulation just prior to the simulation run.
- **SERDESIF_0_PCIE_0_user.bfm** - Intended for user commands that simulate transactions being initiated off-chip (via the SERDES interface). You can simulate the datapath by adding your own commands in this file. Commands in this file will be "executed" after peripheral_init.bfm has completed. This file is for PCIe Protocol 1.
- **SERDESIF_0_PCIE_1_user.bfm** - Intended for user commands that simulate transactions being initiated off-chip (via the SERDES interface). You can simulate the datapath by adding your own commands in this file. Commands in this file will be "executed" after peripheral_init.bfm has completed. This file is for PCIe Protocol 2.

High Speed Serial Interface Configuration Path - SmartFusion2

The configuration register data is used by the CMSIS::SystemInit() function compiled with your firmware application code. The SystemInit() function is run before the user main() function in your application. The Peripheral Initialization solution requires that, in addition to specifying High Speed Serial Interface configuration register values, you configure the APB configuration data path in the MSS (FIC_2). The

SystemInit() function writes the data to the High Speed Serial Interface configuration registers via the FIC_2 APB interface.

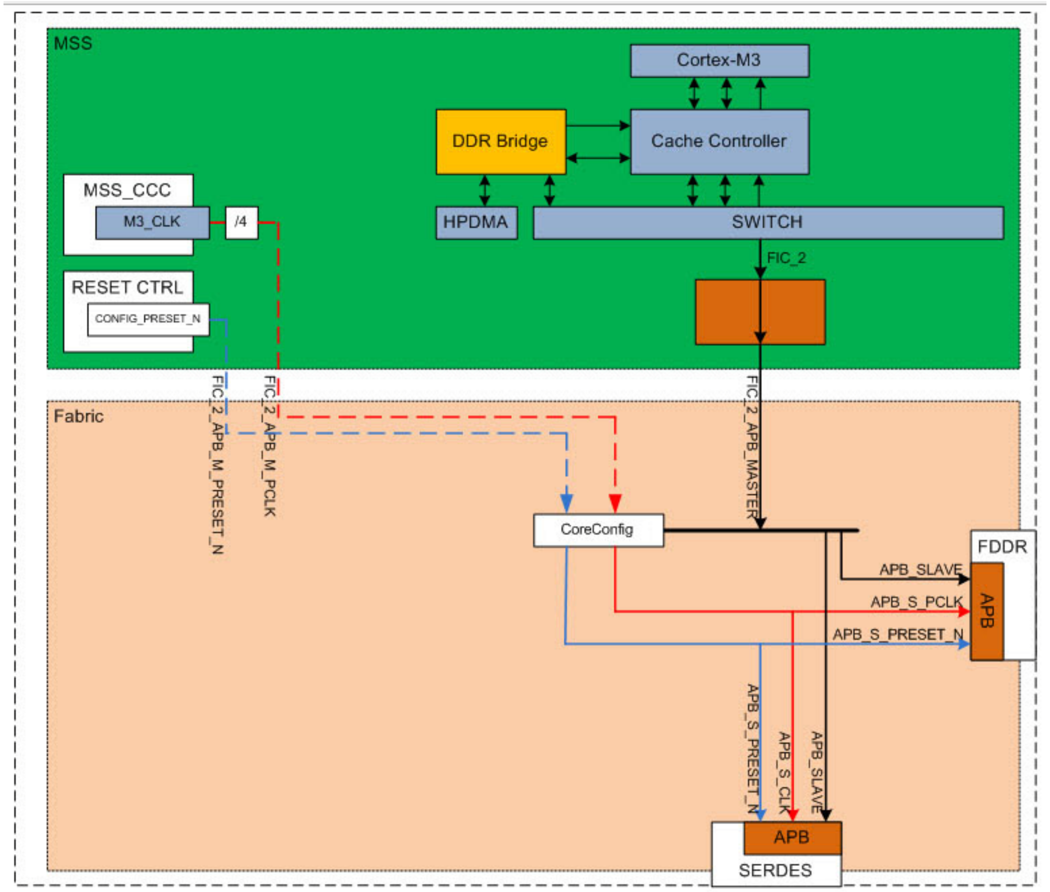


Figure 1-9 • FIC_2 Configurator Overview

To configure the FIC_2 interface:

1. Open the FIC_2 configurator dialog box (Figure 1-9) from the MSS configurator.
2. Select **Initialize peripherals using Cortex-M3**. Make sure that you have clicked the checkbox to enable **Fabric DDR and/or SERDES blocks** and the MSS DDR option (if you are using it).
3. Click **OK** to save your settings. This exposes the FIC_2 configuration ports (Clock, Reset, and APB bus interfaces), as shown in Figure 1-10.
4. Generate the MSS. The FIC_2 ports (FIC_2_APB_MASTER, FIC_2_APB_M_PCLK and FIC_2_APB_M_RESET_N) are now exposed at the MSS interface and can be connected to the CoreConfigP and CoreResetP as per the Peripheral Initialization solution specification.

For details on configuring and connecting the CoreConfigP and CoreResetP cores, refer to the SmartFusion2 DDR Controller and Serial High Speed Controller Initialization Methodology document.

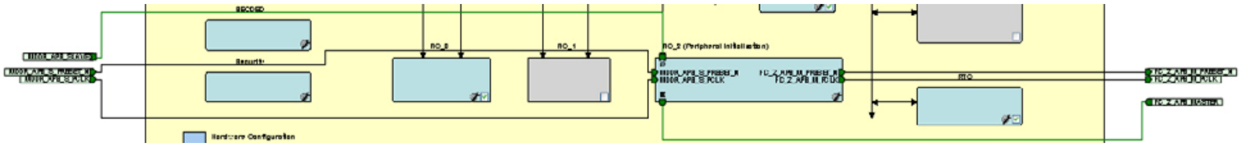


Figure 1-10 • FIC_2 Ports

High Speed Serial Interface Configuration Path - IGLOO2 Initialization

You must use System Builder for IGLOO2 designs that use the SERDES block. System Builder generates an APB bus to handle the internals of the High Speed Serial Interface Configuration Path used for the initial SERDES block configuration and initialization.

The SERDES block is always configured as an APB Slave to the System Builder Block; it cannot be configured as a master. You must configure a fabric master to use the SERDES block as a slave.

After generating your System Builder Block you must connect the SERDES configuration path signals to the System Builder Block:

- Connect the APB_SLAVE BIF of the SERDES block to the corresponding Slave BIF of the System Builder block.
- Connect the APB_S_PCLK port of the SERDES block to the INIT_APB_S_PCLK port of the System Builder block.
- Connect the APB_S_PRESET_N port of the SERDES block to the INIT_APB_S_PRESET_N port of the System Builder block.

SERDES initialization data will now be automatically loaded into the SERDES initialization registers at device bootup.

2 – Port Description

Table 2-1 **• APB Ports**

Port	Direction	Port Group
APB_S_PRDATA[31:0]	OUT	APB_SLAVE
APB_S_PREADY	OUT	
APB_S_PSLVERR	OUT	
APB_S_PADDR[14:2]	IN	
APB_S_PENABLE	IN	
APB_S_PSEL	IN	
APB_S_PWDATA[31:0]	IN	
APB_S_PWRITE	IN	
APB_S_PCLK	IN	
APB_S_PRESET_N	IN	

Table 2-2 **• PCIe_0 Ports**

Port	Direction
PCIE_0_CORE_RESET_N	IN
PHY_RESET_N	IN
CLK_BASE	IN
PCIE_0_INTERRUPT[3:0]	IN
PCIE_0_PERST_N	IN
PCIE_0_WAKE_REQ	IN
PCIE_0_SYSTEM_INT	OUT
SPLL_LOCK	OUT
PLL_LOCK_INT	OUT
PLL_LOCKLOST_INT	OUT
PCIE_0__EV_1US	OUT
PCIE_0_L2P2_ACTIVE	OUT
PCIE_0_RESET_PHASE	OUT

Table 2-2 • **PCIe_0 Ports**

Port	Direction
PCIE_0_WAKE_N	OUT
PCIE_0_LTSSM[5:0]	OUT
REFCLK<x>_OUT where x can be 0 or 1 depending on whether REFCLK0 or REFCLK1 is selected as the Reference Clock Source.	OUT

When Protocol 2 is also PCIe you will see additional PCIE_1_* ports similar to PCIE_0_*.

Table 2-3 • **PCIe_1 Ports**

Port	Direction
PCIE_1_CORE_RESET_N	IN
PHY_RESET_N	IN
CLK_BASE	IN
PCIE_1_INTERRUPT[3:0]	IN
PCIE_1_PERST_N	IN
PCIE_1_WAKE_REQ	IN
PCIE_1_SYSTEM_INT	OUT
SPLL_LOCK	OUT
PLL_LOCK_INT	OUT
PLL_LOCKLOST_INT	OUT
PCIE_1__EV_1US	OUT
PCIE_1_L2P2_ACTIVE	OUT
PCIE_1_RESET_PHASE	OUT
PCIE_1_LTSSM[5:0]	OUT
REFCLK<x>_OUT where x can be 0 or 1 depending on whether REFCLK0 or REFCLK1 is selected as the Reference Clock Source.	OUT
PCIE_1_WAKE_N	OUT

The SmartFusion2 M2S090T/TS and the IGLOO2 M2GL090T/TS devices have two sets of AXI Master/ Slave ports (one set for PCIe_0 and one set for PCIe_1). The port name convention for these ports is (AXI/AHB)_0_(M/S)_* for PCIe_0 Master/Slave interfaces and (AXI/AHB)_1_(M/S)_* for PCIe_1 Master/ Slave interfaces.

Table 2-4 • **PCIe AXI 0 Master Ports**

Port	Direction	Port Group
AXI_0_M_AWID[3:0]	OUT	AXI_0_MASTER
AXI_0_M_AWADDR[31:0]	OUT	
AXI_0_M_AWLEN[3:0]	OUT	
AXI_0_M_AWSIZE[1:0]	OUT	
AXI_0_M_AWBURST[1:0]	OUT	
AXI_0_M_AWVALID	OUT	
AXI_0_M_AWREADY	IN	
AXI_0_M_WID[3:0]	OUT	
AXI_0_M_WSTRB[7:0]	OUT	
AXI_0_M_WLAST	OUT	
AXI_0_M_WVALID	OUT	
AXI_0_M_WDATA[63:0]	OUT	
AXI_0_M_WREADY	IN	
AXI_0_M_BID[3:0]	IN	
AXI_0_M_BRESP[1:0]	IN	
AXI_0_M_BVALID	IN	
AXI_0_M_BREADY	OUT	
AXI_0_M_ARID[3:0]	OUT	
AXI_0_M_ARADDR[31:0]	OUT	
AXI_0_M_ARLEN[3:0]	OUT	
AXI_0_M_ARSIZE[1:0]	OUT	
AXI_0_M_ARBURST[1:0]	OUT	
AXI_0_M_ARVALID	OUT	
AXI_0_M_ARREADY	IN	
AXI_0_M_RID[3:0]	IN	
AXI_0_M_RDATA[63:0]	IN	
AXI_0_M_RRESP[1:0]	IN	
AXI_0_M_RLAST	IN	
AXI_0_M_RVALID	IN	
AXI_0_M_RREADY	OUT	

Table 2-5 • **PCIe AXI 1 Master Ports**

Port	Direction	Port Group
AXI_1_M_AWID[3:0]	OUT	AXI_1_MASTER
AXI_1_M_AWADDR[31:0]	OUT	
AXI_1_M_AWLEN[3:0]	OUT	
AXI_1_M_AWSIZE[1:0]	OUT	
AXI_1_M_AWBURST[1:0]	OUT	
AXI_1_M_AWVALID	OUT	
AXI_1_M_AWREADY	IN	
AXI_1_M_WID[3:0]	OUT	
AXI_1_M_WSTRB[7:0]	OUT	
AXI_1_M_WLAST	OUT	
AXI_1_M_WVALID	OUT	
AXI_1_M_WDATA[63:0]	OUT	
AXI_1_M_WREADY	IN	
AXI_1_M_BID[3:0]	IN	
AXI_1_M_BRESP[1:0]	IN	
AXI_1_M_BVALID	IN	
AXI_1_M_BREADY	OUT	
AXI_1_M_ARID[3:0]	OUT	
AXI_1_M_ARADDR[31:0]	OUT	
AXI_1_M_ARLEN[3:0]	OUT	
AXI_1_M_ARSIZE[1:0]	OUT	
AXI_1_M_ARBURST[1:0]	OUT	
AXI_1_M_ARVALID	OUT	
AXI_1_M_ARREADY	IN	
AXI_1_M_RID[3:0]	IN	
AXI_1_M_RDATA[63:0]	IN	
AXI_1_M_RRESP[1:0]	IN	
AXI_1_M_RLAST	IN	
AXI_1_M_RVALID	IN	
AXI_1_M_RREADY	OUT	

Table 2-6 • **PCIe AXI 0 Slave Ports**

Port	Direction	Port Group
AXI_0_S_AWID[3:0]	IN	AXI_0_SLAVE
AXI_0_S_AWADDR[31:0]	IN	
AXI_0_S_AWLEN[3:0]	IN	
AXI_0_S_AWSIZE[1:0]	IN	
AXI_0_S_AWBURST[1:0]	IN	
AXI_0_S_AWVALID	IN	
AXI_0_S_AWREADY	OUT	
AXI_0_S_AWLOCK[1:0]	IN	
AXI_0_S_WID[3:0]	IN	
AXI_0_S_WSTRB[7:0]	IN	
AXI_0_S_WLAST	IN	
AXI_0_S_WVALID	IN	
AXI_0_S_WDATA [63:0]	IN	
AXI_0_S_WREADY	OUT	
AXI_0_S_BID[3:0]	OUT	
AXI_0_S_BRESP[1:0]	OUT	
AXI_0_S_BVALID	OUT	
AXI_0_S_BREADY	IN	
AXI_0_S_ARID[3:0]	IN	
AXI_0_S_ARADDR[31:0]	IN	
AXI_0_S_ARLEN[3:0]	IN	
AXI_0_S_ARSIZE[1:0]	IN	
AXI_0_S_ARBURST[1:0]	IN	
AXI_0_S_ARVALID	IN	
AXI_0_S_ARLOCK[1:0]	IN	
AXI_0_S_ARREADY	OUT	
AXI_0_S_RID[3:0]	OUT	
AXI_0_S_RDATA[63:0]	OUT	
AXI_0_S_RRESP[1:0]	OUT	
AXI_0_S_RLAST	OUT	
AXI_0_S_RVALID	OUT	
AXI_0_S_RREADY	IN	

Table 2-7 • **PCIe AXI 1Slave Ports**

Port	Direction	Port Group
AXI_1_S_AWID[3:0]	IN	AXI_1_SLAVE
AXI_1_S_AWADDR[31:0]	IN	
AXI_1_S_AWLEN[3:0]	IN	
AXI_1_S_AWSIZE[1:0]	IN	
AXI_1_S_AWBURST[1:0]	IN	
AXI_1_S_AWVALID	IN	
AXI_1_S_AWREADY	OUT	
AXI_1_S_AWLOCK[1:0]	IN	
AXI_1_S_WID[3:0]	IN	
AXI_1_S_WSTRB[7:0]	IN	
AXI_1_S_WLAST	IN	
AXI_1_S_WVALID	IN	
AXI_1_S_WDATA [63:0]	IN	
AXI_1_S_WREADY	OUT	
AXI_1_S_BID[3:0]	OUT	
AXI_1_S_BRESP[1:0]	OUT	
AXI_1_S_BVALID	OUT	
AXI_1_S_BREADY	IN	
AXI_1_S_ARID[3:0]	IN	
AXI_1_S_ARADDR[31:0]	IN	
AXI_1_S_ARLEN[3:0]	IN	
AXI_1_S_ARSIZE[1:0]	IN	
AXI_1_S_ARBURST[1:0]	IN	
AXI_1_S_ARVALID	IN	
AXI_1_S_ARLOCK[1:0]	IN	
AXI_1_S_ARREADY	OUT	
AXI_1_S_RID[3:0]	OUT	
AXI_1_S_RDATA[63:0]	OUT	
AXI_1_S_RRESP[1:0]	OUT	
AXI_1_S_RLAST	OUT	
AXI_1_S_RVALID	OUT	
AXI_1_S_RREADY	IN	

Table 2-8 • **PCIe AHBLite 0 Master Ports**

Port	Direction	Ports Group
AHB_0_M_HADDR[31:0]	OUT	AHB_0_MASTER
AHB_0_M_HBURST[1:0]	OUT	
AHB_0_M_HSIZE[1:0]	OUT	
AHB_0_M_HTRANS[1:0]	OUT	
AHB_0_M_HWRITE	OUT	
AHB_0_M_HWDATA[31:0]	OUT	
AHB_0_M_HREADY	IN	
AHB_0_M_HRESP	IN	
AHB_0_M_HRDATA[31:0]	IN	

Table 2-9 • **PCIe AHBLite 0 Slave Ports**

Port	Direction	Ports Group
AHB_0_S_HSEL	IN	AHB_0_SLAVE
AHB_0_S_HADDR[31:0]	IN	
AHB_0_S_HBURST[1:0]	IN	
AHB_0_S_HSIZE[1:0]	IN	
AHB_0_S_HTRANS[1:0]	IN	
AHB_0_S_HWRITE	IN	
AHB_0_S_HWDATA[31:0]	IN	
AHB_0_S_HREADYOUT	OUT	
AHB_0_S_HRESP	OUT	
AHB_0_S_HREADY	IN	
AHB_0_S_HRDATA[31:0]	OUT	

***Table 2-10* • PCIe AHBLite 1 Master Ports**

Port	Direction	Ports Group
AHB_1_M_HADDR[31:0]	OUT	AHB_1_MASTER
AHB_1_M_HBURST[1:0]	OUT	
AHB_1_M_HSIZE[1:0]	OUT	
AHB_1_M_HTRANS[1:0]	OUT	
AHB_1_M_HWRITE	OUT	
AHB_1_M_HWDATA[31:0]	OUT	
AHB_1_M_HREADY	IN	
AHB_1_M_HRESP	IN	
AHB_1_M_HRDATA[31:0]	IN	

***Table 2-11* • PCIe AHBLite 1 Slave Ports**

Port	Direction	Ports Group
AHB_1_S_HSEL	IN	AHB_1_SLAVE
AHB_1_S_HADDR[31:0]	IN	
AHB_1_S_HBURST[1:0]	IN	
AHB_1_S_HSIZE[1:0]	IN	
AHB_1_S_HTRANS[1:0]	IN	
AHB_1_S_HWRITE	IN	
AHB_1_S_HWDATA[31:0]	IN	
AHB_1_S_HREADYOUT	OUT	
AHB_1_S_HRESP	OUT	
AHB_1_S_HREADY	IN	
AHB_1_S_HRDATA[31:0]	OUT	

Table 2-12 **• XAUI Ports**

Port	**Direction**
XAUI_RXD[63:0]	OUT
XAUI_RXC[7:0]	OUT
XAUI_RX_CLK	OUT
XAUI_VNDRESLO[31:0]	OUT
XAUI_VNDRESHI[31:0]	OUT
XAUI_MMD_MDC	IN
XAUI_MMD_MDI	IN
XAUI_MMD_MDI_EXT	IN
XAUI_MMD_MDOE_IN	IN
XAUI_MMD_PRTAD[4:0]	IN
XAUI_MMD_DEVID[4:0]	IN
XAUI_LOOPBACK_IN	IN
XAUI_MDC_RESET	IN
XAUI_TX_RESET	IN
XAUI_RX_RESET	IN
XAUI_TXD[63:0]	IN
XAUI_TXC[7:0]	IN
XAUI_MMD_MDO	OUT
XAUI_MMD_MDOE	OUT
XAUI_LOWPOWER	OUT
XAUI_LOOPBACK_OUT	OUT
XAUI_MDC_RESET_OUT	OUT
XAUI_TX_RESET_OUT	OUT
XAUI_RX_RESET_OUT[3:0]	OUT
CORE_RESET_N	IN
PHY_RESET_N	IN
SPLL_LOCK	OUT
PLL_LOCK_INT	OUT
PLL_LOCKLOST_INT	OUT
XAUI_OUT_CLK	OUT
XAUI_PMA_READY_N	OUT
REFCLK<x>_OUT where x can be 0 or 1 depending on whether REFCLK0 or REFCLK1 is selected as the Reference Clock Source.	OUT

Table 2-13 • **EPCS Ports per Lane**

Port	Direction	Ports Group
EPCS_<n>_PWRDN	IN	EPCS_<n>_IN Where n can be 0, 1, 2 or 3 depending on the number of configured lanes.
EPCS_<n>_TX_VAL	IN	
EPCS_<n>_TX_OOB	IN	
EPCS_<n>_RX_ERR	IN	
EPCS_<n>_RESET_N	IN	
EPCS_<n>_TX_DATA[<wd>:0]	IN	
EPCS_FAB_REF_CLK When Fabric is selected as the Reference Clock Source in the Configurator	IN	
EPCS_<n>_READY	OUT	EPCS_<n>_OUT Where n can be 0, 1, 2 or 3 depending on the number of configured lanes.
EPCS_<n>_TX_CLK_STABLE	OUT	
EPCS_<n>_TX_CLK	OUT	
EPCS_<n>_RX_CLK	OUT	
EPCS_<n>_RX_VAL	OUT	
EPCS_<n>_RX_IDLE	OUT	
EPCS_<n>_TX_RESET_N	OUT	
EPCS_<n>_RX_RESET_N	OUT	
EPCS_<n>_RX_DATA[<wd>:0]	OUT	
REFCLK<x>_OUT Where x can be 0 or 1 depending on whether REFCLK0 or REFCLK1 is selected as the Reference Clock	OUT	

Note: <n> indicates the lane on which EPCS is configured.

<wd> Valid values are 19,15, 9, 7, 4, and 3.

Table 2-14 • **PAD Ports**

Ports	**Direction**	**Ports Group**	**Description**
REFCLK<x>_SE	IN	PADs_IN	Where x can be 0 or 1 depending on whether REFCLK0 (Single-ended) or REFCLK1 (Single-ended) is selected as the Reference Clock Source.
RXD0_P, RXD0_N	IN		Differential input pair for lane 0 (Rx data)
RXD1_P, RXD1_N	IN		Differential input pair for lane 1 (Rx data)
RXD2_P, RXD2_N	IN		Differential input pair for lane 2 (Rx data)
RXD3_P, RXD3_N	IN		Differential input pair for lane 3 (Rx data)
REFCLK<x>_P, REFCLK<x>_N	IN		Differential input reference clock pair. This port names can be REFCLK0 or REFCLK1 depending on the user selection (refer to Figure 1 on page 3).
TXD0_P, TXD0_N	OUT	PADs_OUT	Differential output pair for lane 0 (Tx data)
TXD1_P, TXD1_N	OUT		Differential output pair for lane 1 (Tx data)
TXD2_P, TXD2_N	OUT		Differential output pair for lane 2 (Tx data)
TXD3_P, TXD3_N	OUT		Differential output pair for lane 3 (Tx data)

A – 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, **650.318.8044**

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 Microsemi SoC Products Group 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.

Visit About Us for sales office listings and corporate contacts.

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

About Microsemi

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; Enterprise Storage and Communication solutions, security technologies and scalable anti-tamper products; Ethernet 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 4,800 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.