HB0921
Handbook
CoreAXItoAXIConnect v2.0

Microsemi

a Microchip company

Microsemi 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
Email: sales.support@microsemi.com
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.

About Microsemi

Microsemi, a wholly owned subsidiary of Microchip Technology Inc. (Nasdaq: MCHP), offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center 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. Learn more at www.microsemi.com.

Contents

Figures

Tables

1 Revision History

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

1.1 Revision 1.0

This is the first publication of this document. Created for CoreAXItoAXIConnect v2.0

2 Acronyms

The following table contains basic acronyms used in all documents related to current hardware implementation:

Table 1 • **Acronym**

Acronym	Details
AMBA	ARM Advanced Microcontroller Bus Architecture
AXI4	Advanced eXtensible Interface v4
AXI4-Lite	Advanced eXtensible Interface v4 Lite
AXI3	Advanced eXtensible Interface v3

3 Introduction

The CoreAXItoAXIConnect, an AXI to AXI Connector that connects a slave interface of one CoreAXI4Interconnect module with the master interface of another CoreAXI4Interconnect module with no intervening logic.

This IP core is used to cascade two CoreAXI4Interconnect IP core. CoreAXI4Interconnect uses Mirrored Master and Mirrored Slave interfaces. Mirrored Slave interface can not be connected directly to the Mirrored Master interface. To cascade two CoreAXI4Interconnect IP core, CoreAXItoAXIConnect IP core implements a Master and Slave interface, and it can be used as a connector between the two CoreAXI4Interconnect IP Cores.

3.1 Key Features

This IP core is used as a connector between two CoreAXI4InterConnect module.

3.2 Core Version

This handbook is for CoreAXItoAXIConnect version 2.0.

3.3 Supported Families

The following list of families support CoreAXItoAXIConnect v2.0:

- PolarFire SoC
- PolarFire®
- SmartFusion®2
- IGLOO®2
- RTG4™

3.4 Supported Interfaces

CoreAXItoAXIConnect is available with the following interfaces:

- AXI3, AXI4, or AXI4-Lite.

3.5 Utilization and Performance

As there is no intervening logic inside the IP, and only the connections are made between the master and slave interface. Therefore, there are no logic elements in CoreAXItoAXIConnect.

Utilization and Performance table is as follows:

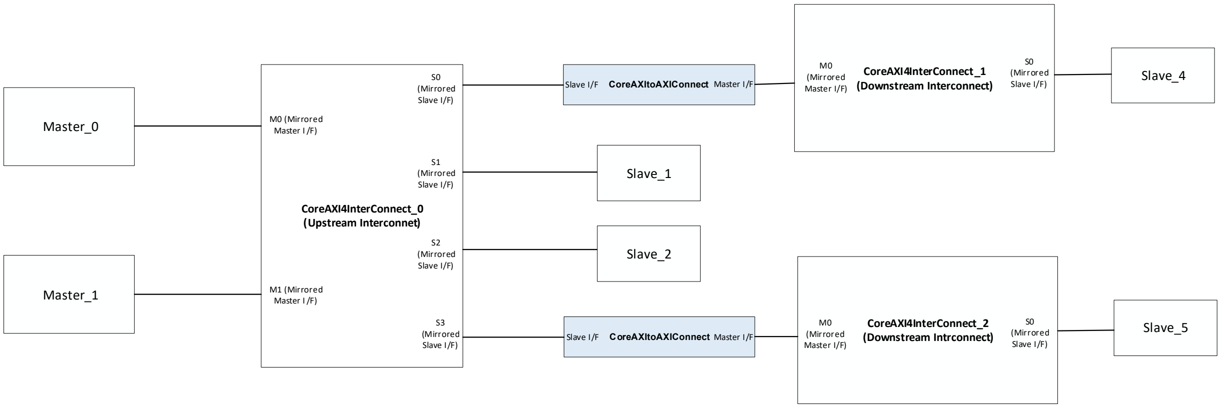
Table 2 • **Resource Utilization and Performance for different families**

		Logic Elements		
Family	**Sequential**	**Combinatorial**	**Total**	**Performance**
SmartFusion2	0	0	0	NA
IGLOO2	0	0	0	NA
RTG4	0	0	0	NA
PolarFire	0	0	0	NA
PolarFire SoC	0	0	0	NA

4 Core Description

The CoreAXItoAXIConnect connects the slave interface of one CoreAXI4Interconnect module with the master interface of another CoreAXI4Interconnect module. This IP provides the port connection points necessary to represent the connectivity in the system. There is no storage element or logic element inside the connector IP, and only the connections are made between the master interface and slave interface.

Figure 1 • **Block Diagram of CoreAXItoAXIConnect with two different blocks of CoreAXI4Interconnect.**


Master_0
Master_1
M0 (Mirrored Master I /F)
M1 (Mirrored Master I /F)
CoreAXI4InterConnect_0 (Upstream Interconnet)
S0 (Mirrored Slave I/F)
S1 (Mirrored Slave I/F)
S2 (Mirrored Slave I/F)
S3 (Mirrored Slave I/F)
Slave I/F CoreAXItoAXIConnect Master I/F
Slave_1
Slave_2
Slave I/F CoreAXItoAXIConnect Master I/F
M0 (Mirrored Master I/F)
CoreAXI4InterConnect_1 (Downstream Interconnect)
S0 (Mirrored Slave I/F)
Slave_4
M0 (Mirrored Master I/F)
CoreAXI4InterConnect_2 (Downstream Intrconnect)
S0 (Mirrored Slave I/F)
Slave_5

5 Core Interfaces

The following sections describe the parameters and I/O signals for CoreAXItoAXIConnect:

5.1 Core Parameters

The following table describes the CoreAXItoAXIConnect parameters:

Table 3 • **CoreAXItoAXIConnect Parameters**

Parameter Name	Valid Values	Default	Description
SLAVE_ID_WIDTH	1-8	1	Width of all ID signals. This parameter should be configured based on ID width of upstream interconnect slave. CoreAXI4Interconnect calculates slave ID using equation given as follows. For more Information, see **CoreAXI4InterConnect handbook** from Libero Catalog. Slave ID Width = $\log_2$[NUM_MASTERS]+ID_WIDTH For example, ID_WIDTH is configured to 2 and NUM_MASTERS used in upstream CoreAXI4Interconnect is 2 then Slave ID Width will be 3. ID_WIDTH for downstream CoreAXI4Interconnect must be configured same as SLAVE_ID_WIDTH. **Note:** As the ID_WIDTH for downstream CoreAXI4Interconnect must be same as SLAVE_ID_WIDTH, user should make sure that Slave ID Width of upstream CoreAXI4Interconnect should not be greater than 8 that is $\log_2$[NUM_MASTERS]+ID_WIDTH should not be greater than 8.
SLAVE_DATA_WIDTH	32, 64, 128, 256, 512	64	Defines the Data width of the upstream Interconnect Slave.
SLAVE_ADDR_WIDTH	16-64	32	Defines the Address width of the upstream Interconnect Slave.
SLAVE_USER_WIDTH	1-64	1	Defines the number of bits for USER signals RUSER and WUSER of the upstream Interconnect Slave.
SLAVE_TYPE[1:0]	2'b00, 2'b01, 2'b11 (2'b10 is reserved)	2'b00	Defines the type of interface for the upstream Interconnect Slave port. Valid values are: • 2'b00 - AXI4 Slave • 2'b01 - AXI4-Lite Slave • 2'b11 - AXI3 Slave

Note: User must configure same parameters for downstream interconnect master.

5.2 I/O Signals

The following table describes the port signals for the CoreAXItoAXIConnect:

Table 4 • **I/O Signals for CoreAXItoAXIConnect**

Name	Type	Description
Master Address Write Channels		
MASTER_AWID [SLAVE_ID_WIDTH-1:0]	Output	Write address ID. The identification tag for the write address group of signals.
MASTER_AWADDR [SLAVE_ADDR_WIDTH-1:0]	Output	Write address. The write address gives the address of the first transfer in a write burst transaction.
MASTER_AWLEN[7:0]	Output	Burst length. The burst length gives the exact number of transfers in a burst. This information determines the number of data transfers associated with the address. This changes between AXI3 and AXI4.
MASTER_AWSIZE[2:0]	Output	Burst size. Size of each transfer in the burst.
MASTER_AWBURST[1: 0]	Output	Burst type. The burst type and the size information, determine how the address for each transfer within the burst is calculated.
MASTER_AWLOCK[1: 0]	Output	Lock type. It provides additional information about the atomic characteristics of the transfer. This changes between AXI3 and AXI4.
MASTER_AWCACHE[3: 0]	Output	Memory type. It shows how transactions are required to progress through a system.
MASTER_AWPROT[2: 0]	Output	Protection type. It gives privilege and security level of the transaction, and whether the transaction is data access or instruction access.
MASTER_AWQOS[3:0]	Output	Quality of Service (QoS). The QoS identifier sent for each write transaction. Implemented only in AXI4.
MASTER_AWREGION [3:0]	Output	Region identifier. It permits a single physical interface on a slave, to be used for multiple logical interfaces. Implemented only in AXI4.
MASTER_AWUSER [SLAVE_USER_WIDTH-1:0]	Output	User signal. Optional user-defined signal in the write address channel. Supported only in AXI4.
MASTER_AWVALID	Output	Write address valid. Channel is signaling valid write address and control information.
MASTER_AWREADY	Input	Write address ready. Slave is ready to accept an address and associated control signals.
Master Write Data Channels		
MASTER_WID [SLAVE_ID_WIDTH-1:0]	Output	Write ID tag. This signal is the ID tag of the write data transfer. Supported only in AXI3.
MASTER_WDATA [SLAVE_DATA_WIDTH- 1:0]	Output	Write Data
MASTER_WSTRB [(SLAVE_DATA_WIDTH/8)-1:0]	Output	Write strobes. It shows which byte lanes hold valid data. There is one write strobe bit for each eight bits of the write data bus.
MASTER_WLAST	Output	Write last. Last transfer in a write burst.

Table 4 • **I/O Signals for CoreAXItoAXIConnect *(continued)***

Name	**Type**	**Description**
MASTER_WUSER [SLAVE_USER_WIDTH-1:0]	Output	User signal. Optional user-defined signal in the write data channel. Supported only in AXI4.
MASTER_WVALID	Output	Write valid. Valid write data and strobes are available
MASTER_WREADY	Input	Write ready. Slave can accept the write data.
Master Write Response Channels		
MASTER_BID [SLAVE_ID_WIDTH-1:0]	Input	Response ID tag. This signal is the ID tag of the write response.
MASTER_BRESP[1:0]	Input	Write response. Status of the write transaction.
MASTER_BUSER [SLAVE_USER_WIDTH- 1:0]	Input	User signal. Optional user-defined signal in the write response channel. Supported only in AXI4.
MASTER_BVALID	Input	Write response valid. Channel is signaling a valid write response.
MASTER_BREADY	Output	Response ready. Master can accept a write response.
Master Address Read Channels		
MASTER_ARID [SLAVE_ID_WIDTH-1:0]	Output	Read address ID. Tag for the read address group of signals.
MASTER_ARADDR [SLAVE_ADDR_WIDTH-1:0]	Output	Read address. Address of the first transfer in a read burst transaction.
MASTER_ARLEN[7:0]	Output	Burst length. Exact number of transfers in a burst. This information determines the number of data transfers associated with the address. This changes between AXI3 and AXI4.
MASTER_ARSIZE[2:0]	Output	Burst size. Size of each transfer in the burst.
MASTER_ARBURST[1: 0]	Output	Burst type. The burst type and the size information, determine how the address for each transfer within the burst is calculated.
MASTER_ARLOCK[1:0]	Output	Lock type. It provides additional information about the atomic characteristics of the transfer. This changes between AXI3 and AXI4.
MASTER_ARCACHE[3: 0]	Output	Memory type. It shows how transactions are required to progress through a system.
MASTER_ARPROT[2:0]	Output	Protection type. It gives privilege and security level of the transaction, and whether the transaction is a data access or an instruction accesses.
MASTER_ARQOS[3:0]	Output	Quality of Service, QoS. The QoS identifier sent for each read transaction. Implemented only in AXI4.
MASTER_ARREGION[3: 0]	Output	Region identifier. It permits a single physical interface on a slave to be used for multiple logical interfaces. Implemented only in AXI4.
MASTER_ARUSER [SLAVE_USER_WIDTH-1:0]	Output	User signal. Optional user-defined signal in the read address channel. Supported only in AXI4.
MASTER_ARVALID	Output	Read address valid. Channel is signaling valid read address and control information.

Table 4 • **I/O Signals for CoreAXItoAXIConnect *(continued)***

Name	Type	Description
MASTER_ARREADY	Input	Read address ready. Slave is ready to accept an address and associated control signals.
Master Read Data Channels		
MASTER_RID [SLAVE_ID_WIDTH-1:0]	Input	Read ID tag. This signal is the identification tag for the read data group of signals generated by the slave
MASTER_RDATA [SLAVE_DATA_WIDTH- 1:0]	Input	Read Data
MASTER_RRESP[1:0]	Input	Read response. Status of the read transfer.
MASTER_RLAST	Input	Read last. Last transfer in a read burst.
MASTER_RUSER [SLAVE_USER_WIDTH- 1:0]	Input	User signal. Optional user-defined signal in the read data channel. Supported only in AXI4.
MASTER_RVALID	Input	Read valid. Channel is signaling the required read data.
MASTER_RREADY	Output	Read ready. Master can accept the read data and response information.
Slave Address Write Channels		
SLAVE_AWID [SLAVE_ID_WIDTH- 1:0]	Input	Write address ID. This signal is the identification tag for the write address group of signals.
SLAVE_AWADDR [SLAVE_ADDR_WIDTH- 1:0]	Input	Write address. The write address gives the address of the first transfer in a write burst transaction.
SLAVE_AWLEN[7:0]	Input	Burst length. The burst length gives the exact number of transfers in a burst. This information determines the number of data transfers associated with the address. This changes between AXI3 and AXI4.
SLAVE_AWSIZE[2:0]	Input	Burst size. This signal indicates the size of each transfer in the burst.
SLAVE_AWBURST[1:0]	Input	Burst type. The burst type and the size information, determine how the address for each transfer within the burst is calculated.
SLAVE_AWLOCK[1:0]	Input	Lock type. It provides additional information about the atomic characteristics of the transfer. This changes between AXI3 and AXI4.
SLAVE_AWCACHE[3:0]	Input	Memory type. It shows how transactions are required to progress through a system.
SLAVE_AWPROT[2:0]	Input	Protection type. It gives privilege and security level of the transaction, and whether the transaction is a data access or an instruction access
SLAVE_AWQOS[3:0]	Input	Quality of Service, QoS. The QoS identifier sent for each write transaction. Implemented only in AXI4.
SLAVE_AWREGION[3: 0]	Input	Region identifier. Permits a single physical interface on a slave to be used for multiple logical interfaces. Implemented only in AXI4.
SLAVE_AWUSER [SLAVE_USER_WIDTH- 1:0]	Input	User signal. Optional user-defined signal in the write address channel. Supported only in AXI4.

Table 4 • **I/O Signals for CoreAXItoAXIConnect *(continued)***

Name	Type	Description
SLAVE_AWVALID	Input	Write address valid. Channel is signaling valid write address and control information
SLAVE_AWREADY	Output	Write address ready. Slave is ready to accept an address and associated control signals.
Slave Write Data Channels		
SLAVE_WID [SLAVE_ID_WIDTH - 1:0]	Input	Write ID tag. This signal is the ID tag of the write data transfer. Supported only in AXI3.
SLAVE_WDATA [SLAVE_DATA_WIDTH- 1:0]	Input	Write Data
SLAVE_WSTRB [(SLAVE_DATA_WIDTH/8)-1:0]	Input	Write strobes. It gives, which byte lanes hold valid data. There is one write strobe bit for each eight bits of the write data bus.
SLAVE_WLAST	Input	Write last. Last transfer in a write burst.
SLAVE_WUSER [SLAVE_USER_WIDTH- 1:0]	Input	User signal. Optional user-defined signal in the write data channel. Supported only in AXI4.
SLAVE_WVALID	Input	Write valid. Valid write data and strobes are available
SLAVE_WREADY	Output	Write ready. Slave can accept the write data.
Slave Write Response Channels		
SLAVE_BID [SLAVE_ID_WIDTH -1:0]	Output	Response ID tag. This signal is the ID tag of the write response.
SLAVE_BRESP[1:0]	Output	Write response. Status of the write transaction
SLAVE_BUSER [SLAVE_USER_WIDTH- 1:0	Output	User signal. Optional user-defined signal in the write response channel. Supported only in AXI4.
SLAVE_BVALID	Output	Write response valid. Channel is signaling a valid write response.
SLAVE_BREADY	Input	Response ready. Master can accept a write response.
Slave Address Read Channels		
SLAVE_ARID [SLAVE_ID_WIDTH -1:0]	Input	Read address ID. This signal is the identification tag for the read address group of signals.
SLAVE_ARADDR [SLAVE_ADDR_WIDTH- 1:0]	Input	Read address. The read address gives the address of the first transfer in a read burst transaction.
SLAVE_ARLEN[7:0]	Input	Burst length. The burst length gives the exact number of transfers in a burst. This information determines the number of data transfers associated with the address. This changes between AXI3 and AXI4.
SLAVE_ARSIZE[2:0]	Input	Burst size. Size of each transfer in the burst.
SLAVE_ARBURST[1:0]	Input	Burst type. The burst type and the size information, determine how the address for each transfer within the burst is calculated.
SLAVE_ARLOCK[1:0]	Input	Lock type. It provides additional information about the atomic characteristics of the transfer. This changes between AXI3 and AXI4.

Table 4 • **I/O Signals for CoreAXItoAXIConnect *(continued)***

Name	Type	Description
SLAVE_ARCACHE[3:0]	Input	Memory type. It shows how transactions are required to progress through a system.
SLAVE_ARPROT[2:0]	Input	Protection type. It gives privilege and security level of the transaction, and whether the transaction is a data access or an instruction access.
SLAVE_ARQOS[3:0]	Input	Quality of Service, QoS. The QoS identifier sent for each read transaction. Implemented only in AXI4.
SLAVE_ARREGION[3: 0]	Input	Region identifier. It permits a single physical interface on a slave to be used for multiple logical interfaces. Implemented only in AXI4.
SLAVE_ARUSER [SLAVE_USER_WIDTH- 1:0]	Input	User signal. Optional user-defined signal in the read address channel. Supported only in AXI4.
SLAVE_ARVALID	Input	Read address valid. Channel is signaling valid read address and control information.
SLAVE_ARREADY	Output	Read address ready. Slave is ready to accept an address and associated control signals.
Slave Read Data Channels		
SLAVE_RID[SLAVE_ID_WIDTH-1:0]	Output	Read ID tag. This signal is the identification tag for the read data group of signals generated by the slave
SLAVE_RDATA [SLAVE_DATA_WIDTH- 1:0]	Output	Read data.
SLAVE_RRESP[1:0]	Output	Read response. Status of the read transfer.
SLAVE_RLAST	Output	Read last. Last transfer in a read burst.
SLAVE_RUSER [SLAVE_USER_WIDTH- 1:0]	Output	User signal. Optional User-defined signal in the read data channel. Supported only in AXI4.
SLAVE_RVALID	Output	Read valid. Channel is signaling the required read data.
SLAVE_RREADY	Input	Read ready. Master can accept the read data and response information.

6 Tool Flows

6.1 License

The CoreAXItoAXIConnect does not require any license.

6.2 RTL

Complete RTL source code is provided for the core.

6.3 SmartDesign

CoreAXItoAXIConnect is preinstalled in the SmartDesign IP Deployment design environment. For information on using the SmartDesign to instantiate and generate cores, see *Using DirectCore in Libero SoC User Guide*.

Figure 2 • **CoreAXItoAXIConnect Instance View**

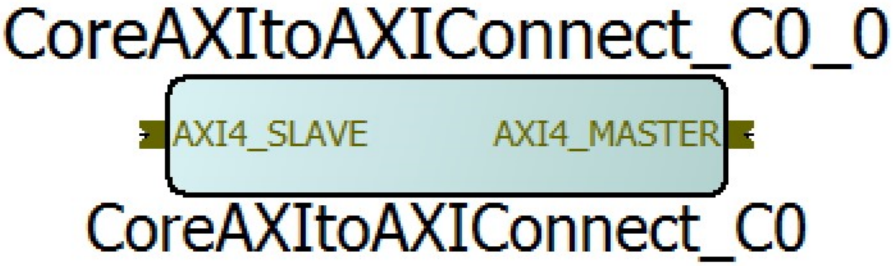
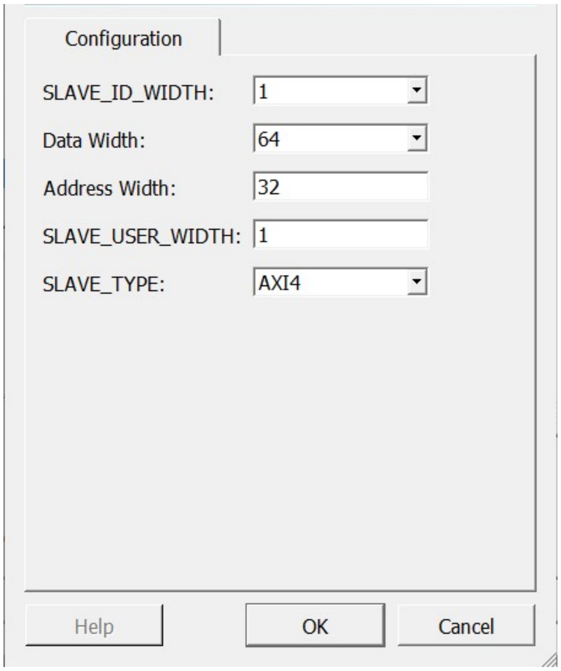


Figure 3 • **SmartDesign Configuration Window**



SLAVE_ID_WIDTH: 1

Data Width: 64

Address Width: 32

SLAVE_USER_WIDTH: 1

SLAVE_TYPE: AXI4

6.4 Synthesis in Libero SoC

After setting the design root appropriately for the design, use the following steps to run the Synthesis.

1. Click **Synthesis** in the Libero SoC software. The Synthesis window appears displaying the Synplicity project.
2. Set Synplicity to use the Verilog 2001 standard, if Verilog is used.
3. Click **Run**.

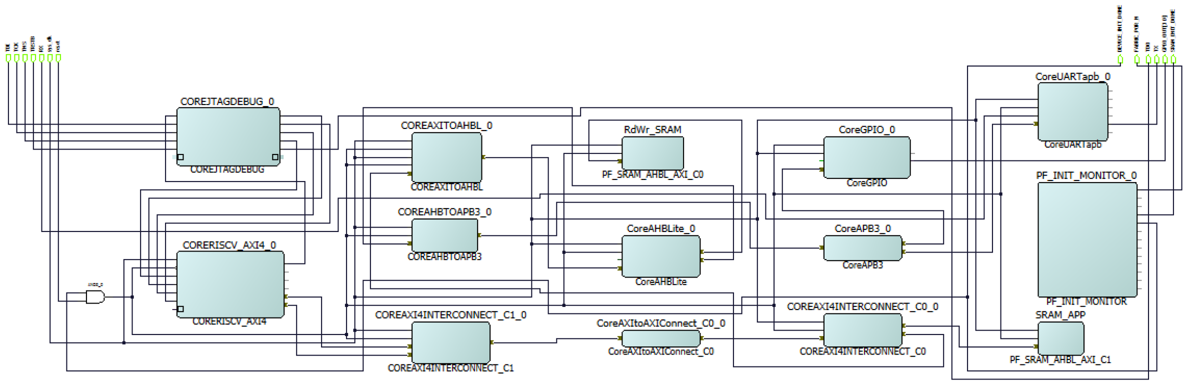
6.5 Place and Route

After setting the design route appropriately for the design and running Synthesis, click **Layout** in the Libero SoC software to invoke Designer. CoreAXItoAXIConnect does not require special place-and-route settings.

7 System Integration

This section provides an example that shows the integration of CoreAXItoAXIConnect.

Figure 4 • **CoreAXItoAXIConnect Example Design**



- The example design described in this section contains two CoreAXI4Interconnects along with CoreAXItoAXIConnect. The upstream CoreAXI4Interconnect configured as two masters and one slave and downstream CoreAXI4Interconnect configured as one master and two slaves. These two interconnects are connected through CoreAXItoAXIConnect.
- PLL_LOCK_0 of PF_CCC ANDing with Device reset on the board and DEVICE_INIT_DONE of PF_INIT_MONITOR is used for COREAXI4INTERCONNECT_C0_0 and COREAXI4INTERCONNECT_C1_0 reset "ARESETN".
- The COREAXI4INTERCONNECT_C0_0 and COREAXI4INTERCONNECT_C1_0 has ACLK which is connected to OUT0_FABCLK_0 of PF_CCC of 50Mhz.

Run the Libero flow with enabling the Timing Driven, High Effort Layout, and Driver Replication options. The example design can be obtained from the Microsemi technical support team.