CoreTBItoEPCS v2.2

HB0389 Handbook

Microsemi
Power Matters.™

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
www.microsemi.com

About Microsemi

Microsemi Corporation (Nasdaq: MSCC) 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. Microsemi is headquartered in Aliso Viejo, Calif., and has approximately 4,800 employees globally. Learn more at www.microsemi.com.

50200389-4/3.17

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 4.0

The following is a summary of the changes in revision 4.0 of this document (March, 2017).

- Replacement of Figure CoreTBItoEPCS SmartDesign Configuration window.
- Updates to Table 3 CoreAHBLSRAM_PF Configuration Options.
- Updates to section 7.1.1 RTL.

1.2 Revision 3.0

The following is a summary of the changes in revision 3.0 of this document (September, 2016).

- The core version was updated from v2.1 to v2.2.
- Replacement of Figure 2-3. Source Synchronous Clocking Mode Block Diagram.
- Replacement of Figure 2-4. Asynchronous Clocking Mode Block Diagram.
- Updates to Table 1-1. CoreTBItoEPCS Parameters/Generics Descriptions.
- Updates to Table 1-2. CoreTBItoEPCS I/O Signal Descriptions.

1.3 Revision 2.0

The following is a summary of the changes in revision 2.0 of this document (March, 2014).

- The core version was updated from v2.0 to v2.1.
- IGLOO2 support has been added.
- Updated "Utilization and Performance" section.
- Updated Table 1-1 CoreTBItoEPCS Parameters/Generics Descriptions.
- Added "SGMII System Level Clock Modes" section under "Design Details" chapter.
- Updated "Licensing" section and "SmartDesign" section.
- Updated Figure 3-2.
- Updated Table 4-1.

1.4 Revision 1.0

Revision 1.0 was the first publication of this document (December, 2012).

Contents

Figures

Tables

2 Preface

2.1 Purpose

This handbook provides details about the CoreTBItoEPCS DirectCore module and how to use it.

2.2 Intended Audience

FPGA designers using Libero® System-on-Chip (SoC).

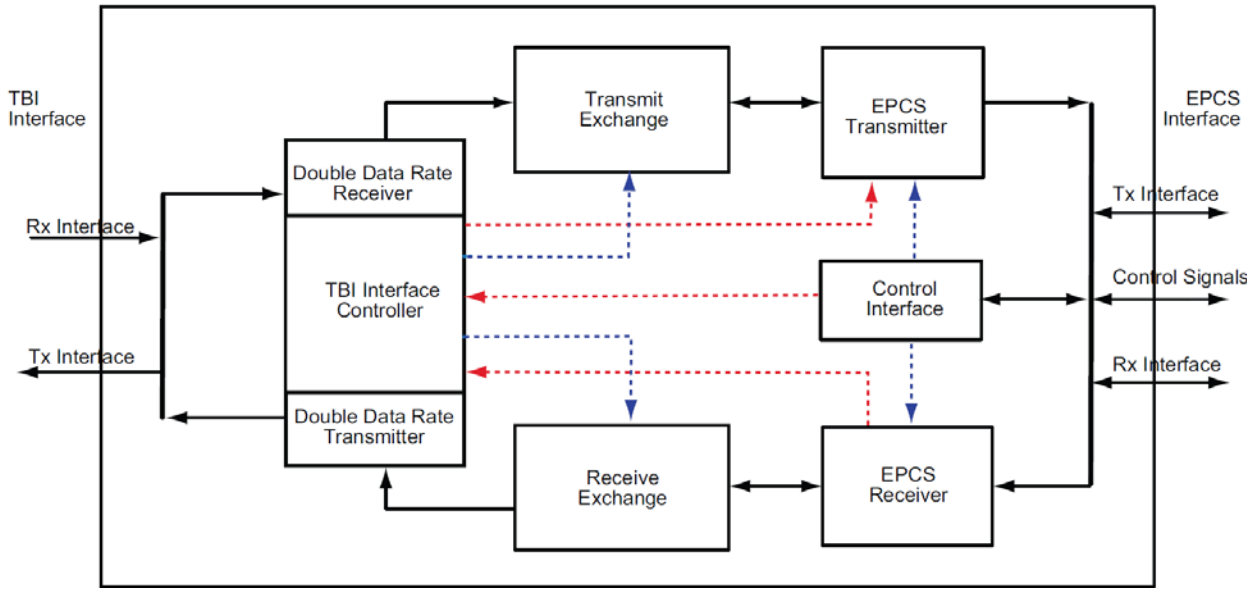
3 Introduction

3.1 Overview

A bridge exists between the ten-bit interface (TBI) and external physical coding sublayer (EPCS). It has a ten-bit transmit/receive interface on the TBI side and a 20-bit transmit/receive interface on EPCS side. It receives TBI data and transmits it toward the EPCS, and receives EPCS data and transmits it toward the TBI.

The CoreTBItoEPCS block diagram is shown in Figure 1.

Figure 1 • **CoreTBItoEPCS Block Diagram**



Only 10 bits (data bits) are used by the physical medium attachment (PMA) on the EPCS Tx/Rx bus. This block receives the 10-bit data (with double data rate) from the TBI and stores it in a transmit exchange. Then it reads the transmit exchange and transmits the data on the EPCS transmit bus. This block also receives data from the EPCS receive bus and stores it in the receive exchange. Then it reads the receive exchange and transmits the data on the 10-bit TBI transmit bus.

3.2 CoreTBItoEPCS Blocks

CoreTBItoEPCS consists of major five blocks as described below.

3.2.1 TBI Interface Controller

This block receives the data (at double data rate) on the TBI Rx bus and sends it to the transmit exchange. It also receives the data from the receive exchange and transmits it (at double data rate) on the TBI Tx bus.

3.2.2 Transmit Exchange

This block receives the data from the TBI controller and it writes the data into internal FIFO. It reads the FIFO and sends the data to the EPCS transmitter. This block acts as a write buffer.

3.2.3 Receive Exchange

This block receives the data from the EPCS receiver and it writes the data into internal FIFO. It reads the data from FIFO and sends it to the TBI controller. This block acts as a read buffer.

3.2.4 EPCS Transmitter

This block reads the data from the transmit exchange and sends it on the EPCS Tx bus when the physical layer (PHY) is ready.

3.2.5 EPCS Receiver

This block reads the data from the EPCS Rx bus and sends it to the receive exchange.

The TBI and EPCS clocks are both asynchronous. The TBI clock frequency is 62.5 MHz and EPCS clock frequency is 125 MHz. This block performs the clock-domain-crossing operation.

3.3 Features

- TBI and EPCS clocks are asynchronous to each other.
- TBI clock frequency = 62.5 MHz and EPCS clock frequency = 125 MHz.
- Data on the TBI (TCGF/RCGF) bus comes with double data rate.
- TBI data width is 10 bits.
- EPCS data width is 20 bits but only 10 bits are used, depending on the [TX|RX]_UPPER_EPCS parameter/generic.
- This bridge accepts data from the TBI only after PHY is ready (EPCS_READY), assuming that the TBI will not send data until the PHY is ready.

3.4 Core Version

This handbook is for CoreTBItoEPCS version 2.2.

3.5 Supported Families

- IGLOO®2
- SmartFusion®2

3.6 Supported Interfaces

- Ten-bit interface (TBI)
- External PCS interface

3.7 Device Utilization and Performance

Table 1 shows the utilization and performance data for CoreTBItoEPCS.

Table 1 • **CoreTBItoEPCS Device Utilization and Performance**

Family	Cells or Tiles			Utilization		Performance (MHz)
	Sequential	Combinatorial	Total	Device	Total %	
SmartFusion2	240	333	573	M2S150T	0.4	189
IGLOO2	240	333	573	M2GL150T	0.4	189

Note: Data in this table were achieved using typical synthesis and layout settings. Top-level parameters/generics were left at their default values.

4 Functional Description

4.1 Functional Blocks

CoreTBItoEPCS, shown in Figure 1 on page 9, consists of decoding logic for the double-data-rate TBI bus, write channel FIFO buffer, read channel FIFO buffer, multiple clock synchronization logic, control logic for the 10-bit TBI interface, and control logic for the 20-bit EPCS interface. CoreTBItoEPCS is activated when it receives EPCS_READY (PHY ready) from the external PCS block.

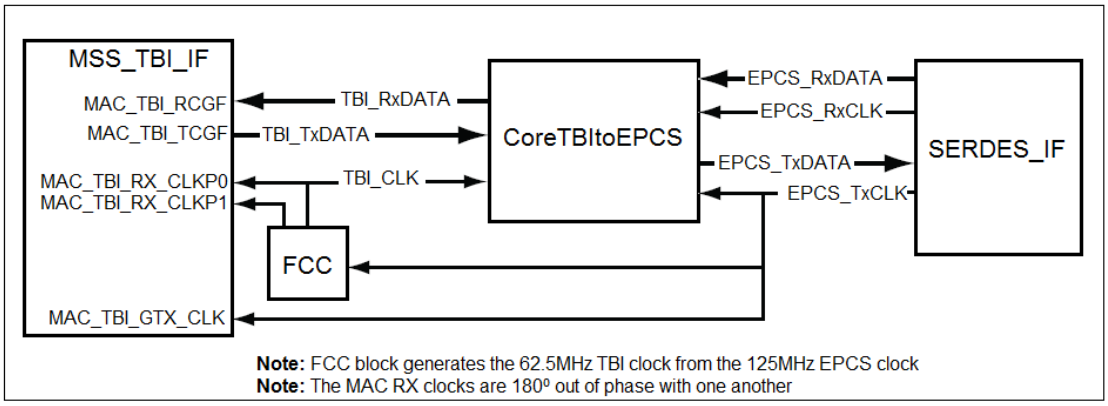
4.2 SGMII System Level Clock Modes

There are two system level clocking modes to serial gigabit media independent interface (SGMII). These are Source Synchronous and Asynchronous Clocking.

4.2.1 Source Synchronous

In this mode, a local clock is shared between both ends of the SGMII link. The SERDESIF output clocks EPCS_TxCLK and EPCS_RxCLK will have a 0ppm frequency offset with an arbitrary phase. The CoreTBItoEPCS supports this mode with a single instance using a FIFO to absorb the phase differences between the EPCS_TxCLK and EPCS_RxCLK domains.

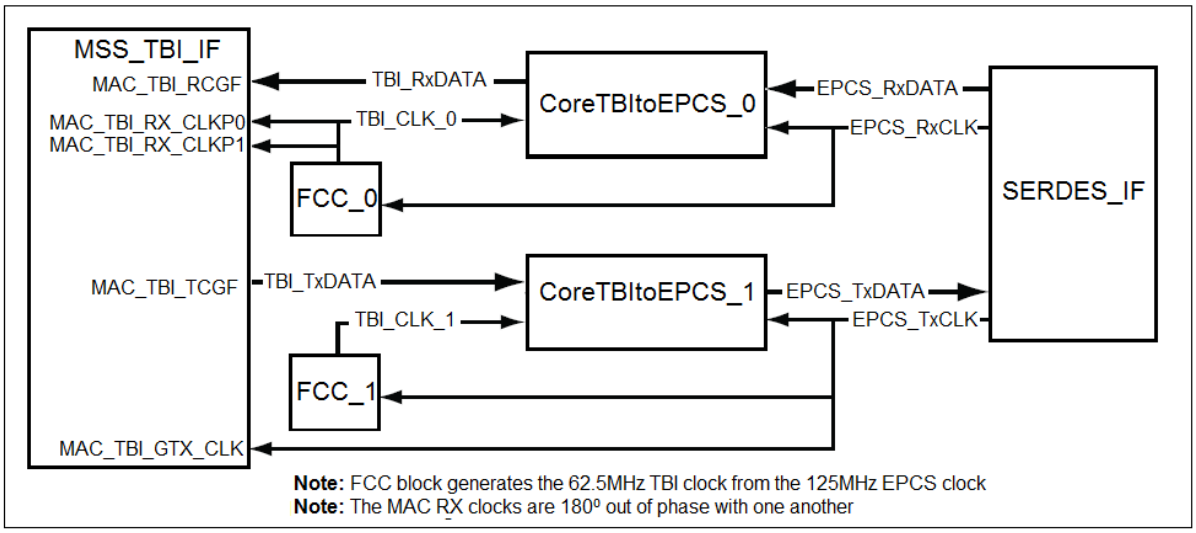
Figure 2 • **Source Synchronous Clocking Mode Block Diagram**



4.2.2 Asynchronous Clocking

In this mode, both ends of the SGMII link use their own local reference clock. In this mode, two instances of CoreTBItoEPCS are required. One instance is used for the transmit domain. The second instance is used for the receive domain. Each of the instances will only support a single data flow direction, the pins associated with the other direction should be tied off statically.

Figure 3 • **Asynchronous Clocking Mode Block Diagram**



Note: MSS_TBI_IF represents the TBI interface on the microcontroller subsystem (MSS).

5 Interface

5.1 I/O Signals

The port signals for the CoreTBItoEPCS macro are illustrated in Figure 4 and defined.

Figure 4 • **CoreTBItoEPCS I/O Signal Diagram**

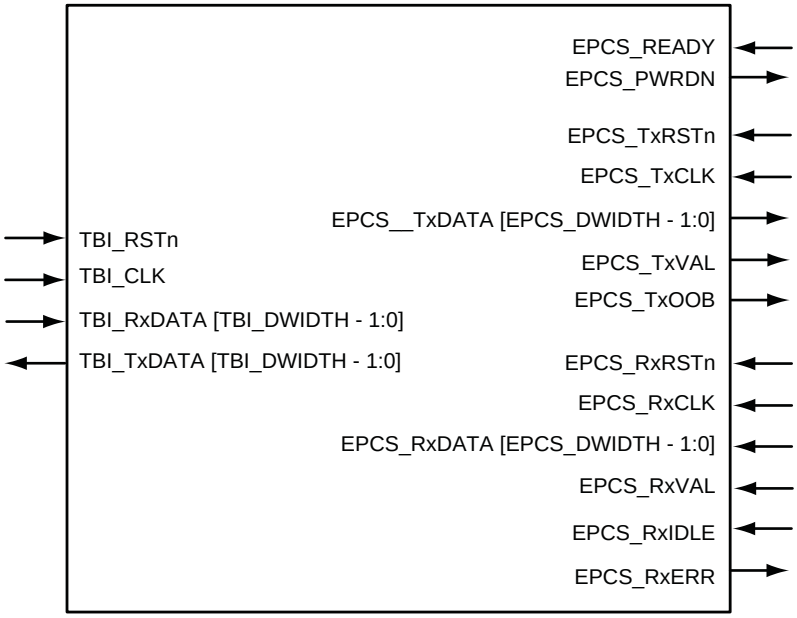


Table 2 • **CoreTBItoEPCS I/O Signal Descriptions**

Port Name	Type	Description
TBI Reset and Clock		
TBI_RSTn	In	TBI active low asynchronous reset. Asynchronous assertion and synchronous deassertion. This is used to reset TBI registers in the block.
TBI_CLK	In	TBI clock: All TBI signals are synchronous to the rising edge of this clock signal.
TBI Transmit/Receive Interface		
TBI_RxDATA [TBI_DWIDTH-1:0]	In	TBI receive data bus. This port is used to receive data with double data rate from TBI.
TBI_TxDATA [TBI_DWIDTH-1:0]	Out	TBI transmit data bus. This port is used to transmit data with double data rate to TBI.
External PCS Control Interface		
EPCS_READY	In	PHY ready signal. This signal is asserted when the PHY has completed the calibration sequence for each specific lane. This signal can be used in order to release the reset for the external PCS and controller, start transmitting data to the PMA, or any other purpose.

Table 2 • **CoreTBItoEPCS I/O Signal Descriptions**

EPCS_PWRDN	Out	PHY power-down signal. This signal is used to put the PMA in powerdown state, where Rx CDR PLL is bypassed and other low power features are applied to the PMA. When exiting power-down, no calibration is required and the link can be operational much faster than using the EPCS_TXOOB or EPCS_RSTN signals.
External PCS Transmit Interface		
EPCS_TxOOB	Out	PHY transmit out-of-band (OOB) signal. This signal is used to load Electrical Idle III in the Tx driver of the PMA macro. It can be used for SATA as part of the sequencing for transmitting very short OOB signaling.
EPCS_TxRSTn	In	PHY clean active low reset on Tx clock. This signal is a clean version of the EPCS_RSTN signal and has clean deassertion timing compared to EPCS_TXCLK.
EPCS_TxCLK	In	PHY transmit clock signal This signal is the aTxClk signal generated by the PMA macro and must be used by the external PCS logic to provide data on EPCS_TXDATA[19:0].
EPCS_TxDATA	Out	PHY transmit data signal. This signal is used to transmit data. This signal is always 20 bits per lane but the PMA macro will used only the number of bit defined by the aTxN[4:0] feature of the PMA macro. Upper or Lower active bits are set by TX_UPPER_EPCS.
EPCS_TxVAL	Out	PHY transmit valid signal. This signal is used to transmit valid data. If deasserted, the PMA macro is put in Electrical Idle I. It can be used for protocol requiring Electrical Idle (SATA) and must also be deasserted as long as EPCS_READY is not asserted. This signal must be generated one clock cycle earlier than the corresponding EPCS_TXDATA[] signals.
External PCS Receive Interface		
EPCS_RxRSTn	In	PHY clean active low reset on Rx clock. This signal is a clean version of the EPCS_RSTN signal and has clean deassertion timing compared to EPCS_RXCLK.
EPCS_RxCLK	In	PHY receive clock signal. This signal is the aTxClk signal generated by the PMA macro and must be used by the external PCS logic to provide data on EPCS_TXDATA[19:0].
EPCS_RxDATA	In	PHY receive data bus. This signal is always 20 bits per lane and the external PCS can use any number of these bits for its application based on the aRxN[4:0] feature of the PMA macro. Upper or lower active bits are set by RX_UPPER_EPCS.
EPCS_RxVAL	In	PHY receive valid data signal. This signal is used to signal receive valid data. It corresponds to the two conditions completed by the PMA control logic:: • Receiver detect incoming data (not in Electrical Idle) • CDR PLL is locked to input bitstream in fine grain state **Note:** If PMA driven mode is used by the selected protocol (see the CDRPLL_DIS register in REG00), this signal cannot be monitored and the EPCS_RXIDLE signal must be used instead.
EPCS_RxIDLE	In	PHY receive idle signal. This signal is used to signal an Electrical Idle condition detected by the PMA control logic. Note that this signal is generated on EPCS_TXCLK of the selected lane.

Table 2 • **CoreTBItoEPCS I/O Signal Descriptions**

EPCS_RxERR	Out	PHY Receive Error signal. This signal is used to report to PMA control logic that error data has been detected by the external PCS logic. This signal is used in PCS driven mode of the CDR PLL to switch back to frequency lock acquisition if too many errors are detected by the PMA control logic. This signal is unused in PMA-driven mode and can also be hardwired to zero if the PCS wants to rely only on Electrical Idle detection circuitry to switch the CDR PLL back to frequency lock state.

5.2 Configuration Parameters

5.2.1 CoreTBItoEPCS Configurable Options

There are a number of configurable options that apply to CoreTBItoEPCS as shown in Table 3. If a configuration other than the default is required, select the configuration dialog box in SmartDesign to select appropriate values for the configurable options.

Table 3 • **CoreAHBLSRAM_PF Configuration Options**

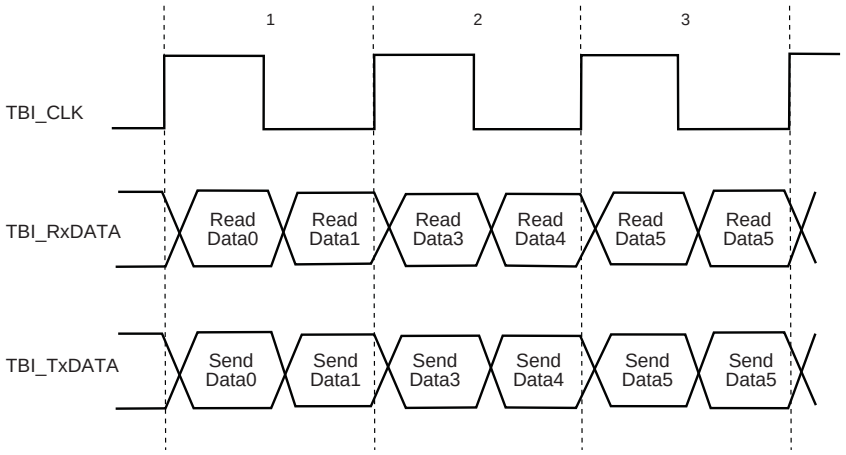
Name	Valid Range	Default	Default
FAMILY	19 or 24	19	Must be set to the required FPGA family: 19: SmartFusion2 24: IGLOO2
TX_UPPER_EPCS	0 or 1	0	This bit sets the active 10-bit Tx data bus from the 20-bit EPCS Tx data bus. 0: Lower 10 bits are active for both Tx EPCS data buses. 1: Upper 10 bits are active for both Rx/Tx EPCS data buses.
RX_UPPER_EPCS	0 or 1	0	This bit sets the active 10-bit Rx data bus from the 20-bit EPCS Rx/Tx data bus. 0: Lower 10 bits are active for both Rx EPCS data buses. 1: Upper 10 bits are active for both Rx EPCS data buses.

6 Timing Diagrams

CoreTBItoEPCS implements an Rx/Tx TBI interface and Rx/Tx External PCS interfaces.

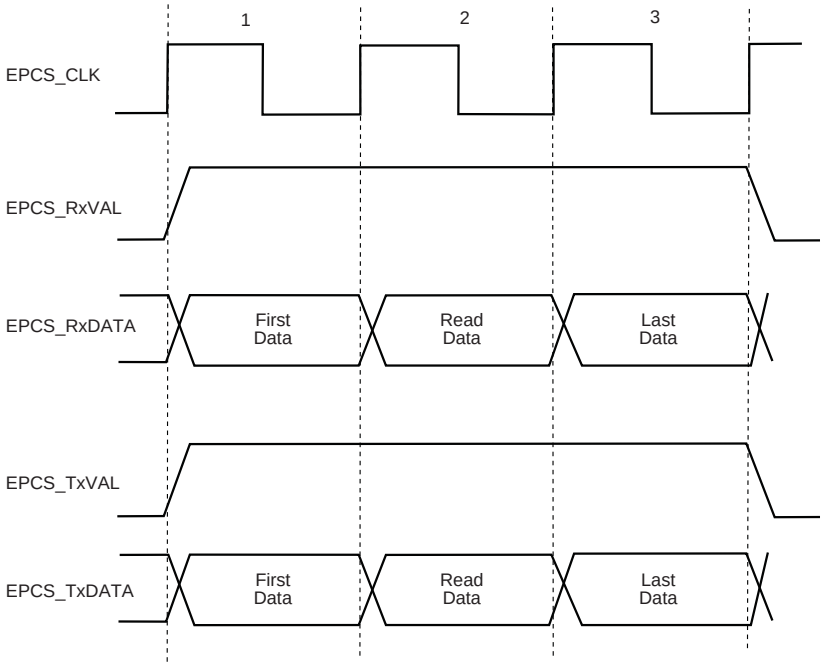
6.1 TBI Interface Timing

Figure 5 • **TBI Interface Timing Diagram**



6.2 External PCS Interface Timing

Figure 6 • **EPCS Interface Timing**



7 Tool Flow

7.1 License

CoreTBItoEPCS is licensed free.

7.1.1 RTL

Complete Verilog and VHDL RTL source code is provided for the core and testbenches.

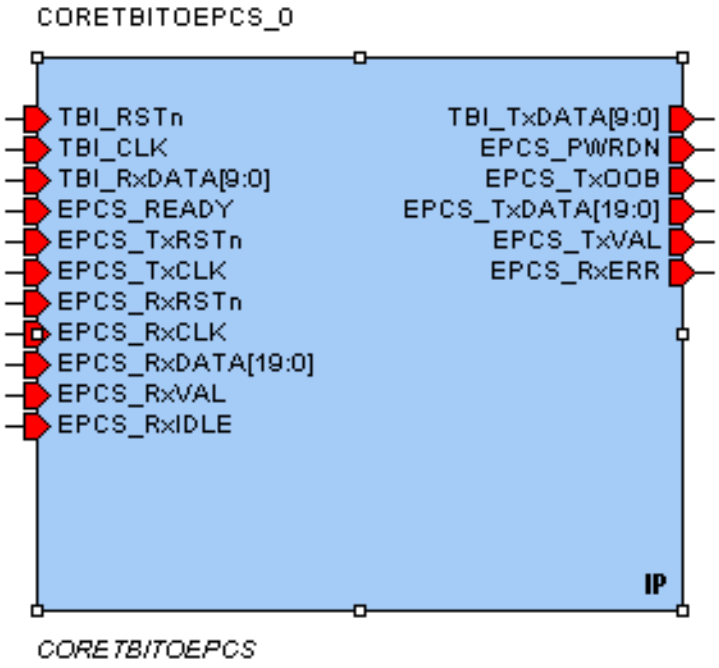
7.2 SmartDesign

CoreTBItoEPCS is available for download in the Libero IP catalog through the web repository. Once it is listed in the catalog, the core can be instantiated using the SmartDesign flow. For information on using SmartDesign to configure, connect, and generate cores, refer to the Libero online help. An example instantiated view is shown in Figure 7.

After configuring and generating the core instance, basic functionality can be simulated using the test-bench supplied with the CoreTBItoEPCS. The testbench parameters automatically adjust to the CoreTBItoEPCS configuration. The CoreTBItoEPCS can be instantiated as a component of a larger design.

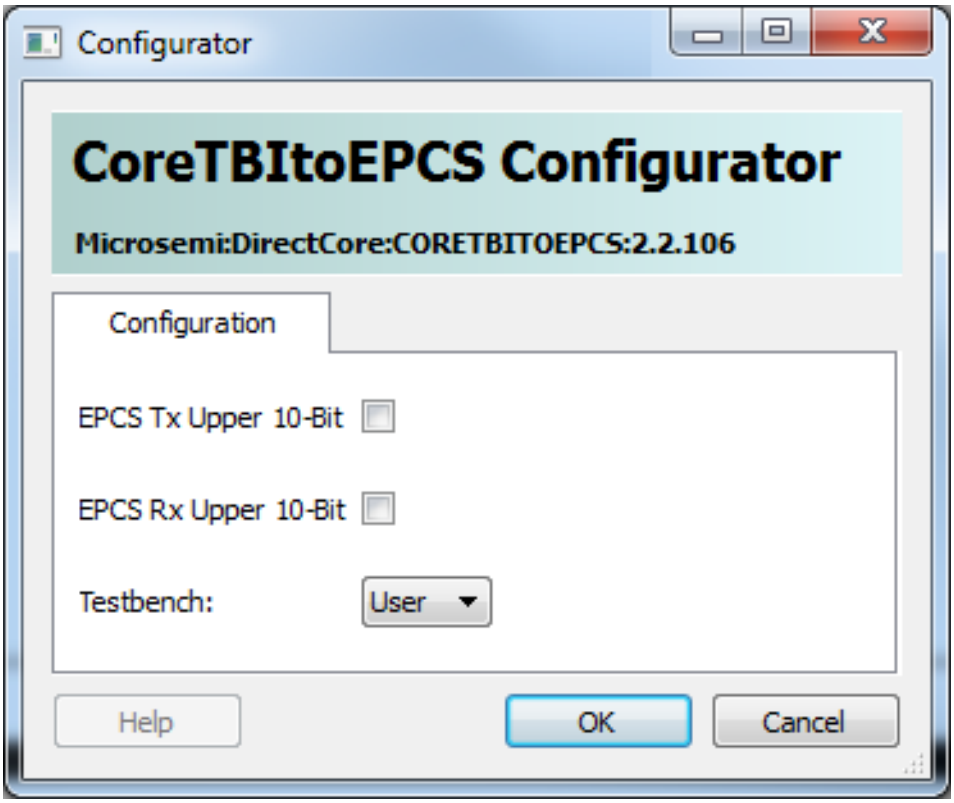
CoreTBItoEPCS is compatible with Libero SoC. For more information on using SmartDesign to instantiate and generate cores, refer to the Using DirectCore in Libero® System-on-Chip (SoC) User Guide or consult the Libero SoC online help.

Figure 7 • **CoreTBItoEPCS Full I/O View**



7.2.1 Configuring CoreTBItoEPCS in SmartDesign

Figure 8 • **CoreTBItoEPCS SmartDesign Configuration window**



7.3 Simulation Flows

The User Testbench for CoreTBItoEPCS is included in all releases.

To run simulations, select the **User Testbench** flow within SmartDesign and click **Save** and **Generate** on the **Generate** pane. The **User Testbench** is selected through the Core Testbench Configuration GUI.

When SmartDesign generates the Libero SoC project, it installs the user testbench files.

To run the **User Testbench**, set the design root to the CoreTBItoEPCS instantiation in the Libero SoC design hierarchy pane and click the **Simulation** icon in the Libero SoC design flow window. This invokes ModelSim® and automatically run the simulation.

7.4 Synthesis in Libero

Click the **Synthesis** icon in Libero SoC. The Synthesis window displays the Synplicity® project. Set Synplicity to use the Verilog 2001 standard if Verilog is being used. To run **Synthesis**, select the **Run** icon.

7.5 Place-and-Route in Libero

Click the **Layout** icon in the Libero SoC to invoke Designer. CoreTBItoEPCS requires no special place-and-route settings.

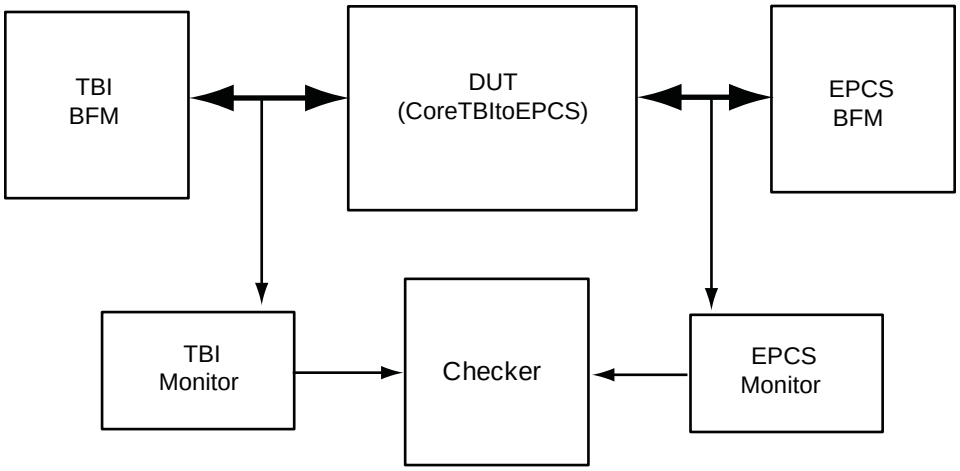
8 Testbench

A unified test-bench is used to verify and test CoreTBItoEPCS called a user test-bench.

8.1 User Test-bench

CoreTBItoEPCS's user testbench gives an example of how to use the core with a TBI and external PCS. As shown in Figure 9, the testbench instantiates a behavioral TBI bus functional model (BFM) module and EPCS bus functional model module to emulate using a TBI and external PCS.

Figure 9 • **CoreTBItoEPCS User Test-bench**



The simulation testbench shown in Figure 9 includes: an instantiation of the CoreTBItoEPCS macro, TBI bus functional model, EPCS bus functional model, TBI/EPCS monitors, and data checker. The toplevel testbench (TB_USER) includes the necessary open drain connections. The testbench utilizes a simple PHY ready function call to initialize CoreTBItoEPCS, and then transmit and/or receive TBI data to EPCS and/or EPCS data to TBI.