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

1 Revision History ......................................................... 1  
   1.1 Revision 4.0 ................................................................. 1  
   1.2 Revision 3.0 ................................................................. 1  
   1.3 Revision 2.0 ................................................................. 1  
   1.4 Revision 1.0 ................................................................. 1  

2 RTG4 FPGA SERDES EPCS Protocol Design .......................... 2  
   2.1 Design Requirements ..................................................... 2  
   2.2 Demo Design ................................................................. 3  
      2.2.1 Introduction ........................................................... 3  
      2.2.2 Description ............................................................ 5  
   2.3 Setting Up the Demo Design .............................................. 7  
      2.3.1 Setting Up the Board ............................................... 7  
      2.3.2 Programming the Demo Design .................................. 8  
      2.3.3 Installing the Demo GUI ....................................... 10  
      2.3.4 Running the Demo Design ...................................... 12  
   2.4 Conclusion ................................................................. 14  

3 Appendix: Using RTG4 for Customer Design .......................... 15  
   3.1 Transmitter Section ...................................................... 15  
   3.2 Receiver Section .......................................................... 15  

4 Appendix: Simulating the Design ....................................... 16  

5 Appendix: Verifying Timing using SmartTime .......................... 18  

6 Appendix: GUI Status Signal .............................................. 19
Figures

Figure 1  Design Files Top-Level Structure ......................................................... 3
Figure 2  RTG4 Demo Design Files Top-Level Structure ........................................ 4
Figure 3  Demo Design Block Diagram ............................................................... 5
Figure 4  TX and RX Interface RTL Blocks ......................................................... 6
Figure 5  RTG4 Development Kit Board .............................................................. 8
Figure 6  New Project Window .............................................................................. 8
Figure 7  FlashPro GUI Window .......................................................................... 9
Figure 8  Single Device Configuration Window .................................................... 10
Figure 9  Successful Programming Session .......................................................... 10
Figure 10 GUI Set Up Window ........................................................................... 11
Figure 11 GUI Setup Progress Bar ...................................................................... 12
Figure 12 EPCS Demo GUI Window ................................................................. 13
Figure 13 Sample GUI Window .......................................................................... 14
Figure 14 Replacing Demo Design with Customer Design .................................... 15
Figure 15 Organizing Simulation Testbench in Project ......................................... 16
Figure 16 Simulating the Design ....................................................................... 17
Figure 17 Simulation Waveform Window ............................................................. 17
Figure 18 Verify Timing ..................................................................................... 18
Figure 19 SmartTime ......................................................................................... 18
Figure 20 SmartTime Session ........................................................................... 18
## Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1</td>
<td>Design Requirements</td>
<td>2</td>
</tr>
<tr>
<td>Table 2</td>
<td>Jumper Settings</td>
<td>7</td>
</tr>
<tr>
<td>Table 3</td>
<td>GUI Status Signals</td>
<td>19</td>
</tr>
</tbody>
</table>
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
Updated the document for Libero v11.8 SP1 software release.

1.2 Revision 3.0
Updated the document for Libero v11.8 SP2 software release.

1.3 Revision 2.0
Updated the document for Libero v11.7 software release.

1.4 Revision 1.0
Revision 1.0 is the first publication of this document.
The RTG4™ devices have embedded high-speed SERDES blocks that can support data rates between 1 Gbps and 3.125 Gbps. The high-speed serial interface block supports several serial communication standards. The RTG4 SERDES block integrates several functional blocks to support multiple high-speed serial protocols within the FPGA.

The EPCS mode exposes the SERDES lanes directly to the fabric and configures the SERDES block in physical media attachment (PMA) only mode. In the EPCS mode, the peripheral component interconnect express (PCIe®) and ten Gigabit attachment unit interface (XAUI) PCS logic in the SERDES block is bypassed. However, the PCS logic can be implemented in the FPGA fabric, and the EPCS interface signals of the SERDES block can be connected to user protocol. This allows any user-defined high-speed serial protocol to be implemented in the RTG4 device.

In conjunction with EPCS, the available CorePCS IP module supports programmable 8B10B encoding and decoding. 8B10B is commonly used in protocols that are not included in the SERDES block by the Microsemi system-on-chip (SoC) high-speed SERDES interface. Therefore, the CorePCS IP module can be used with these protocols. It can be configured as a transmitter only, receiver only, or both transmitter and receiver. Word alignment support is included in the receiver. It can also be configured to support 10-bit or 20-bit EPCS data. For more information about this, see the CorePCS Handbook.

The SERDES blocks are completely configurable. Initial register settings of the SERDES blocks are required at run-time. This demonstration design initializes the SERDES configuration registers using the SERDES block that has a built-in initialization state machine. The state machine loads the SERDES block with the correct register settings on power up or assertion of DEVRST.

This demo describes the following:

- EPCS interface of the RTG4 device with High-Speed Serial Interface (PCIe, EPCS, XAUI) with initialization.
- Using CorePCS IP modules for customized applications

### 2.1 Design Requirements

**Table 1**, page 2 lists the design requirements.

<table>
<thead>
<tr>
<th>Design Requirements</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Hardware Requirements</strong></td>
<td></td>
</tr>
<tr>
<td>RTG4 Development Kit:</td>
<td></td>
</tr>
<tr>
<td>– USB 2.0 cable</td>
<td></td>
</tr>
<tr>
<td>– FlashPro4 programmer</td>
<td></td>
</tr>
<tr>
<td>– 12 V, 5A AC power adapter and cords</td>
<td>2- SMA Male-to-SMA Male Precision Cables, such as Pasternack Industries part number PE39429-12 (or equivalent)</td>
</tr>
<tr>
<td>SMA Male to SMA Male Loopback Cables</td>
<td></td>
</tr>
<tr>
<td>STAPL/PDB file</td>
<td>Programming file</td>
</tr>
<tr>
<td>Host PC or Laptop</td>
<td>Any 64-bit Operating System</td>
</tr>
<tr>
<td><strong>Software Requirements</strong></td>
<td></td>
</tr>
<tr>
<td>Libero® System-on-Chip (SoC)</td>
<td>v11.9 SP1</td>
</tr>
<tr>
<td>FlashPro Programming Software</td>
<td>v11.9 SP1</td>
</tr>
<tr>
<td>GUI Software</td>
<td>Provided in the design files archive file</td>
</tr>
<tr>
<td>Host PC Drivers for FlashPro5</td>
<td>USB to UART drivers</td>
</tr>
</tbody>
</table>
2.2 Demo Design

2.2.1 Introduction

The demo design files are available for download from the following path in the Microsemi website:

The demo design files include the following:
- GUI installer
- Libero SoC project
- Programming file
- Source files
- Test benches

Table 1 • Design Requirements (continued)

<table>
<thead>
<tr>
<th>Design Requirements</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Framework</td>
<td>Microsoft .NET Framework 4 client for launching demo GUI</td>
</tr>
</tbody>
</table>
This example design demonstrates transmitting a pseudo random binary sequence (PRBS) or counting pattern over the RTG4 high-speed SERDES interface. The SERDES block is configured for 2.5 Gbps operational speed. The PRBS pattern is sent over Lane 0, and a counting 8B10B encoded pattern is used with Lane 1 of the SERDES block.

- Demo 1: Lane 0 traffic is sent directly from the fabric based PRBS generator to the SERDES block and off-chip to test SMA connections. Input SMA connectors are routed to the SERDES receiver pins to bring the data back into the device to the fabric based pattern checker. External SMA cables complete the TX to RX loop back circuit.

- Demo 2: Lane 1 includes a pattern generator and checker that also utilizes the CorePCS IP module in the data path. The CorePCS IP module provides simple 8b/10b encoding and decoding functionality. This lane is routed off and on chip by looping the data on PCB trace connections.

**Note:** Lane 2 and Lane 3 are not used.
2.2.2 Description

The hardware design for the implementation includes a PRBS and count pattern generator, PRBS sequence and count pattern checker, error counter, RX and TX fabric interface blocks, delay line, UART and output select control and high-speed serial interface block connected to the RTG4 SERDES block. Each of these blocks is explained in the following sections:

- PRBS7 Generator, page 6
- Count Generator, page 6
- PRBS7 Checker, page 6
- Count Checker, page 6
- RX and TX Interface, page 6
2.2.2.1 PRBS7 Generator
The generator implements the PRBS7 polynomial \((x^7+x^6+1)\) and generates a continuous sequence of PRBS7 patterns of 10 bits each. Each 10-bit transmission from the generator occurs at a frequency of 39.3 MHz. The PRBS generator module runs at 125 MHz.

2.2.2.2 Count Generator
The count generator module implements a count pattern used to drive the CorePCS 8b10b encoder. Packets created in the count generator are separated by a K28.5 character. The payload of the packet is a simple counting pattern.

2.2.2.3 PRBS7 Checker
The PRBS7 checker checks for valid PRBS sequences. If the received sequence does not match with the one transmitted by the generator, the checker indicates an error. The checker also implements an error counter, which is incremented for each error in the received PRBS sequence.

2.2.2.4 Count Checker
The count checker module checks for a valid count pattern received from the CorePCS 8b10b decoder. The count checker checks each packet for the embedded count pattern used as the payload of the packet.

2.2.2.5 RX and TX Interface
RX and TX interfaces modules manage the timing relationships of the clock and data from the EPCS interface to the FPGA fabric when the global clocks are not used.

Figure 4 • TX and RX Interface RTL Blocks
2.2.2.6 CoreUART, FabUART, and Output Select Modules

The COREUART module communicates with UART interface on the RTG4 Development Kit. FabUART and Output select modules are glue logic modules to connect the PRBS generator and checker control and error reporting signals to the GUI that communicates to the device over UART. The Output Select block multiplexes status signals like Error, Error count, and Lock signals from both Lane 0 and Lane 1. Depending on the Lane selection, it feeds the corresponding status signals onto the UART.

2.2.2.7 SERDES

The RTG4 high-speed SERDES is a hard IP block on chip that supports rates up to 3.125 Gbps. The SERDES block offers embedded protocol support for PCIe and XAUI. The SERDES block also supports EPCS interface, which can be used for custom protocols. This Demo uses the SERDES block in the EPCS protocol. See the UG0567: RTG4 FPGA High Speed Serial Interfaces User Guide for more information on SERDES block. In this design, the SERDESIF block is configured to be 20-bit wide, 125 MHz REFCLK, and 2.5 Gbps.

2.2.2.8 Clocking

The two different types of clock domains in the EPCS demo design are:

- Control Plane Clock: Used for initialization of the SERDES block, UART, and Output Select. The control plane clock is sourced by the Fabric PLL and is passed to the control blocks at 50 MHz rate.
- EPCS interface output Clock: Each SERDES lane provides an output clock for the transmitter and the receiver. The transmit clock is used to clock epcs_tx_intf and remainder of the transmit data path. The receive clock is used to clock epcs_rx_intf and remainder of the receive path.

2.3 Setting Up the Demo Design

2.3.1 Setting Up the Board

The following steps describe how to setup the hardware demo for the RTG4 Development Kit:

1. Connect the jumpers on the board, as shown in Table 2, page 7.

The following table lists the jumper settings.

<table>
<thead>
<tr>
<th>Jumper</th>
<th>Pin (From)</th>
<th>Pin (To)</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>J11, J17, J19, J23, J26, J21, J32, J27, J28</td>
<td>1</td>
<td>2</td>
<td>Default</td>
</tr>
<tr>
<td>J16</td>
<td>2</td>
<td>3</td>
<td>Default</td>
</tr>
<tr>
<td>J33</td>
<td>1</td>
<td>2</td>
<td>Default</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

Note: Ensure that the power supply switch, SW6 is switched OFF while connecting the jumpers on the RTG4 Development Kit.

2. Connect the host PC or Laptop to the J47 connector using the USB min-B cable. This serves as both the FlashPro5 programmer interface and the UART control interface for the demo GUI.

3. Connect 12 V 6 A-power jack to the J9 power connector.

4. Ensure that the USB to UART bridge drivers are automatically detected. If USB to UART bridge drivers are not installed, download and install the drivers from: www.microsemi.com/soc/documents/CDM_2.08.24_WHQL_Certified.zip

5. Install the SMA Male to SMA Male connectors (J49 to J50 and J58 to J59).
2.3.2 Programming the Demo Design

The following steps describe how to program the demo design:

1. Download the design files from:
   http://soc.microsemi.com/download/rsc/?f=rtg4_dg0624_serdes_epcs_protocol_libero11p8sp2_df
   Programming file (STAPL) is located in the Programming_File folder.

2. Launch the FlashPro software.
3. Click New Project.
4. In the New Project dialog box, enter the Project Name, as shown in Figure 6, page 8.

Figure 6 • New Project Window
5. Select **Single device** as **Programming mode**.

6. Click **OK** to save the project. The FlashPro GUI is displayed. The Programmer List Window is updated with the programmer information.

**Figure 7 • FlashPro GUI Window**

After the project is created and the programmer is connected, the STAPL file downloaded in Step 1 is ready to be loaded.

7. Click **Configure Device** (highlighted in **Figure 7**, page 9). The **Single Device Configuration window** is displayed in FlashPro, as shown in **Figure 7**, page 9. The **Basic Mode** is used for this operation.
8. Click **Browse** to find the programming file. The **Load Programming File** dialog box is displayed.

9. Select the programming file and click **Open**. The **Single Device Configuration Window** is updated to list the Programming file information. **PROGRAM** is the default action displayed in the **Action** menu.

**Figure 8** • Single Device Configuration Window

10. Click **PROGRAM** to start programming the device. Wait until the Programmer Status is changed to **RUN PASSED** (highlighted in **Figure 9**, page 10).

**Figure 9** • Successful Programming Session

11. View the Log window and note the details about the programmed device.

### 2.3.3 Installing the Demo GUI

The following steps describe how to run the installer if the GUI is used for the first time:
1. Download the design files from:
   http://soc.microsemi.com/download/rsc/?f=rtg4_dg0624_serdes_epcs_protocol_libero11p8sp2_df
2. Open GUI_Installer > Volume > setup.exe.
3. Click Yes for any message from User Account Control. The Destination Directory window is displayed with the default locations, as shown in Figure 10, page 11.
4. Click Next.

**Figure 10 • GUI Set Up Window**

![GUI Set Up Window]

5. Follow the instructions in the GUI to start the installation.

**Note:** Accept the license agreement and click Next in the Summary dialog box.
A progress bar shows the progress of installation, as shown in Figure 11, page 12.

Figure 11 • GUI Setup Progress Bar

6. Wait for the installation to complete. After successful installation, Installation Complete message is displayed.
7. Click Finish.
8. Restart the computer before using the installed GUI.

2.3.4 Running the Demo Design

The following steps describe how to run the demo design:

1. Open Programs > EPCS_RTG4.

Figure 12, page 13 shows the GUI window.
Figure 12 • EPCS Demo GUI Window

The drop-down list for ports has the list of serial ports available on the Host PC. The working ports are enabled and the unavailable ports are grayed out.

2. Click **Connect** to connect the Host PC to the hardware through the selected port. The status signals indicate the status of the complete system operation.

3. Select the desired lane. Either Lane 0 or Lane 1 can be individually tested.

4. Click **Start** to start the EPCS Demo. The PRBS7 data (or Count data) is sent over the serial transmit link. It is then received by the receiver and checked for any errors. The status can be monitored using the status signals in the GUI at any time. For more information on the status signals, see **GUI Status Signals**, page 19.

5. Click **Stop** to stop the EPCS demo.

6. Click **Exit** to exit the GUI.
Figure 13, page 14 shows a sample GUI window during an error free operation of the SERDES EPCS demo.

Figure 13 • Sample GUI Window

2.4 Conclusion

This demo describes the EPCS interface of the RTG4 device, the use of the self-initializing SERDES block and CorePCS IP modules, and how to use them for customized applications. This demonstration design allows users to see an actual implementation of the RTG4 SERDES block on the development kit. The design shows data traffic into and out of the block. It can be used for signal quality analysis of the SERDES transmitters as well as demonstrate error free data looped between the RTG4 transmitter and receiver of the SERDES block.
Appendix: Using RTG4 for Customer Design

3.1 Transmitter Section

The PRBS7 generator in the transmitter section can be replaced with the customer data generator. The data generator is interfaced with the TX Interface block as shown in Figure 1, page 15.

3.2 Receiver Section

The PRBS7 checker in the receiver section can be replaced with the data receiver in the customer design. The data receiver takes input from the RX Interface, as shown in Figure 1, page 15.

Figure 1 • Replacing Demo Design with Customer Design

This demo is targeted for RT4G150. It is recommended to use a clock resource to reduce the clock injection time into the fabric. This is done by instantiating an RCLKINT library element on the clock outputs of SERDESIF EPCS TX_CLK and RX_CLK.

The interface blocks for the transmitter and receiver are also recommended to achieve timing closure. These blocks employ a scheme specifically designed for the RTG4 family and optimize the interface for both setup and hold when not using the global clocks. These blocks must be used exactly from this demo design into all EPCS designs. Verilog HDL is used in this tutorial. VHDL modules are available in the SOURCE Directory.
The following steps describe how to simulate the design:

1. Import the `testbench_tb.v` file to the Libero project.
2. Right-click Simulate and select Organize Input files > Organize Stimulus Files to setup the EPCS_TOP_TB.v file for simulation, as shown in Figure 1, page 16.
3. Import the `wave.do` file to the Simulation folder of the Libero project.
   
   **Note:** `wave.do` and `EPCS_TOP_TB.v` files are available in the Test benches folder of EPCS_Demo project.
4. Go to Project > Project Settings > waveforms and select Include DO file check box.
5. Change the Simulation runtime to 2 us using the Do File option under Project Settings.
6. Select vsim command under Project Settings and change the resolution to 1 ps.
7. Click Save to save the settings.
8. Right-click Simulate in the Libero Design Flow and select Open Interactively.

The design is simulated through a test bench. The testbench simulates the high-speed serial interface block in the EPCS mode. To run the simulation, double-click Simulate under Verify Pre- Synthesized Design in the Design Flow tab of the Libero project, as shown in Figure 2, page 17.
Observe the simulation results for Lane0 and Lane1.

The simulation automatically runs from the testbench and shows data on the lanes, generates and checks the results. The simulation posts messages to the log indicating the various steps of the SERDES initialization and operation.

Once the simulation moves to the operational phase, use Run -all to continue with the testbench.

After simulation, the Simulation Waveform window is displayed, as shown in Figure 3, page 17.
Appendix: Verifying Timing using SmartTime

SmartTime is a gate-level static timing analysis tool. Using SmartTime, user can perform a complete timing analysis of the design and ensure that all timing constraints are met, and the design operates at desired speed with the right amount of margin across all operating conditions. For more information, see the SmartTime for Libero SoC User’s Guide.

Double-click Verify Timing. The Libero software will perform a timing check and report any timing violations.

Figure 1 • Verify Timing

SmartTime is an interactive tool that can be invoked from the Design Flow pane. Double-click Open SmartTime.

Figure 2 • SmartTime

SmartTime window is displayed in Max Delay Analysis View as shown in Figure 3, page 18. It shows the setup or hold time violations that cause design issues in the hardware.

Figure 3 • SmartTime Session
### Appendix: GUI Status Signal

Table 1 shows the various status signals.

**Table 1 • GUI Status Signals**

<table>
<thead>
<tr>
<th>Status Signal</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host Connection</td>
<td>Indicator of COM port connection on the host PC.</td>
</tr>
<tr>
<td></td>
<td>GREEN: COM port is connected.</td>
</tr>
<tr>
<td></td>
<td>RED: COM port is disconnected.</td>
</tr>
<tr>
<td>Serial Link</td>
<td>Indicator of transmission link for serial data.</td>
</tr>
<tr>
<td></td>
<td>GREEN: Link is up and running.</td>
</tr>
<tr>
<td></td>
<td>RED: Link is down.</td>
</tr>
<tr>
<td>Rx Lock</td>
<td>Receiver lock.</td>
</tr>
<tr>
<td></td>
<td>GREEN: The receiver receives a valid and error-free data. The receiver is locked to the PRBS7 or count sequences and the subsequent transmitted sequences can be successfully received.</td>
</tr>
<tr>
<td></td>
<td>RED: The receiver receives an invalid data.</td>
</tr>
<tr>
<td>Rx Error</td>
<td>Indicates the status of the packets received.</td>
</tr>
<tr>
<td></td>
<td>GREEN: Received packets are error-free.</td>
</tr>
<tr>
<td></td>
<td>RED: A corrupted packet or any error is detected in the received PRBS7 or count sequences.</td>
</tr>
<tr>
<td>Error Count</td>
<td>Gives the count of errors detected in the received PRBS sequences.</td>
</tr>
<tr>
<td>Generate Error</td>
<td>Introduces errors in the transmission for debug purposes. Introduces error in the transmitted PRBS sequence, which increments the Error Count display.</td>
</tr>
<tr>
<td>Clear Error</td>
<td>Sets error count to zero.</td>
</tr>
</tbody>
</table>