RapidIO Dynamic Data Rate Reconfiguration Reference Design for Stratix IV GX Devices

AN-617-1.0 Application Note

The RapidIO dynamic data rate reconfiguration reference design demonstrates how to use the ALTGX_RECONFIG megafunction to reconfigure the RapidIO® MegaCore® function data rate using the Stratix® IV GX EP4SGX230KF40C3ES Signal Integrity Transceiver development board. The ALTGX_RECONFIG megafunction allows you to change the data rate during run time without recompiling your design.

This application note contains the following sections:

- "Functional Description"
- "Using the Reference Design" on page 11
- "Test Cases for Verification (Dual-board Connection)" on page 16

For more information about the RapidIO MegaCore function, refer to the *RapidIO MegaCore Function User Guide*.

Functional Description

The reference design uses the Nios® II embedded processor to control the dynamic reconfiguration process. You can provide a software command to the Nios II processor to process and execute data rate reconfiguration. The reconfiguration process triggers the ALTGX_RECONFIG megafunction to reconfigure the RapidIO MegaCore function transceiver with the desired data rate.

The reference design uses the Channel and clock multiplier unit (CMU) phase-locked loop (PLL) reconfiguration mode to reconfigure the transceiver in the RapidIO MegaCore function. The Channel and CMU PLL reconfiguration mode uses a Memory Initialization File (**.mif**) to change transceiver settings, such as the data rate. The reference design reconfigures the transmit (TX) and receive (RX) channels of the RapidIO MegaCore function.

101 Innovation Drive
San Jose, CA 95134
www.altera.com

© 2010 Altera Corporation. All rights reserved. ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS and STRATIX are Reg. U.S. Pat. & Tm. Off. and/or trademarks of Altera Corporation in the U.S. and other countries. All other trademarks and service marks are the property of their respective holders as described at www.altera.com/common/legal.html. Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

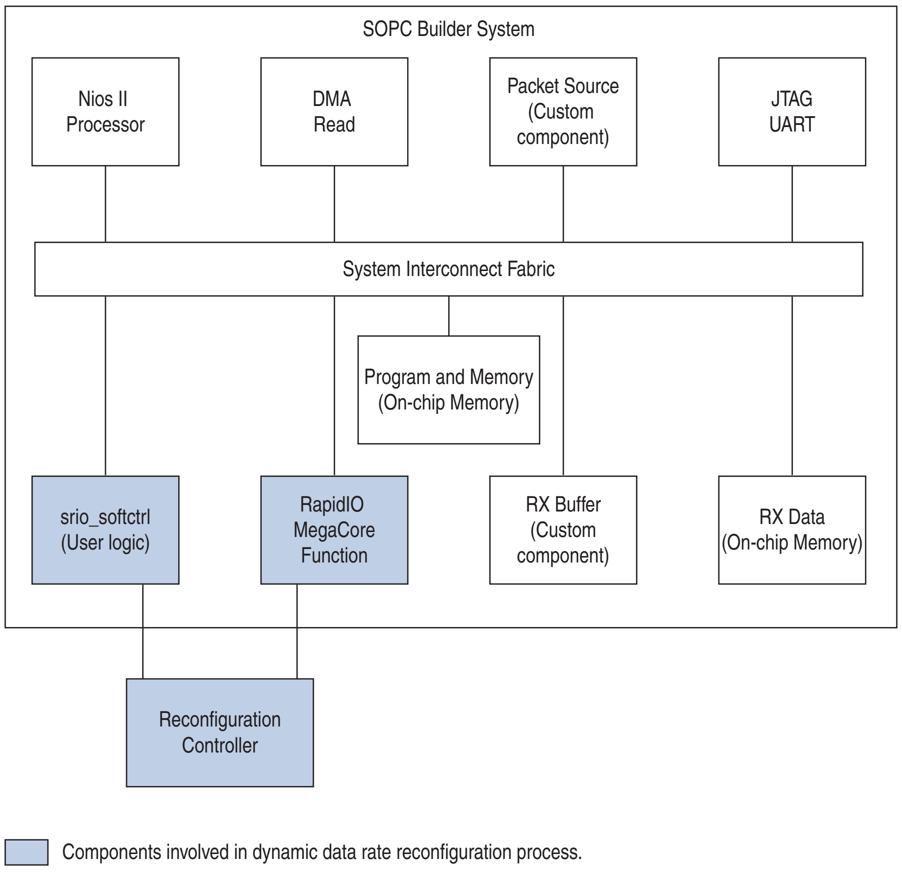
QUALITY
ISO 9001:2008
NSAI Certified

Subscribe

System Architecture

Figure 1 shows an overview of the system architecture.

Figure 1. System Architecture Overview


SOPC Builder System
Nios II Processor
DMA Read
Packet Source (Custom component)
JTAG UART
System Interconnect Fabric
Program and Memory (On-chip Memory)
srio_softctrl (User logic)
RapidIO MegaCore Function
RX Buffer (Custom component)
RX Data (On-chip Memory)
Reconfiguration Controller
Components involved in dynamic data rate reconfiguration process.

The following sections describe the roles of the main system components in the reference design.

Nios II Processor

The Nios II processor processes and executes the dynamic data rate reconfiguration process when you issue a data rate reconfiguration command.

DMA Read

The DMA read component sends read data from the I/O slave read port of the RapidIO MegaCore function to the RX data memory.

Packet Source (Custom Component)

The packet source custom component generates I/O burst transfer packets to the RapidIO MegaCore function write slave port.

JTAG UART

The JTAG UART component provides the mechanism for your device to communicate with the Nios II terminal. The Nios II terminal is the user interface to the RapidIO MegaCore function driver.

Program and Memory (On-Chip Memory)

The program and memory component stores the software program code of the Nios II processor. When you download the software program code, this component stores the code and the Nios II processor executes the code.

RX Buffer (Custom Component)

The RX buffer custom component stores the data written to and read from the write and read master ports of the RapidIO MegaCore function, respectively.

RX Data (On-Chip Memory)

The RX data stores the read data from the DMA read component.

The components responsible for dynamic data rate reconfiguration are discussed in the following section.

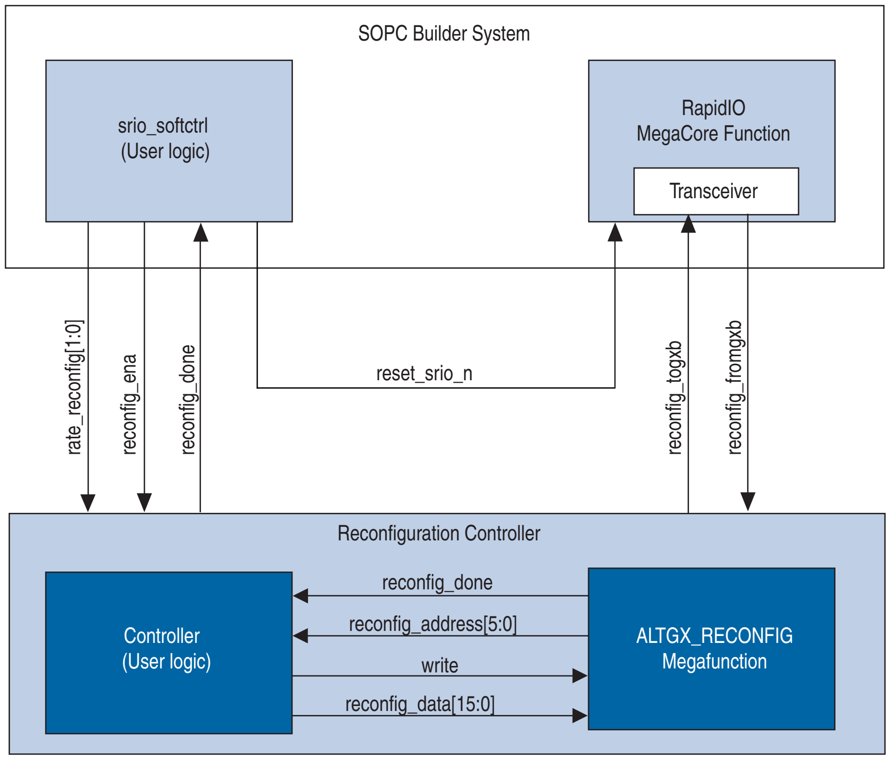
Dynamic Data Rate Reconfiguration Components

This section describes the following dynamic data rate reconfiguration components:

- srio_softctrl (User Logic)
- RapidIO MegaCore Function
- Reconfiguration Controller

Figure 2 shows the signals for the `srio_softctrl`, reconfiguration controller, and RapidIO MegaCore function components.

Figure 2. srio_softctrl, Reconfiguration Controller, and RapidIO MegaCore Function Signals



srio_softctrl (User Logic)

The `srio_softctrl` user logic sends reconfiguration instructions to the reconfiguration controller, and at the same time also sends the reset signal to reset the RapidIO MegaCore function. This user logic connects to the Avalon® Memory-Mapped (Avalon-MM) interface to deliver the reconfiguration software command from the Nios II processor to the reconfiguration controller.

To create a custom component in the SOPC Builder system, the `srio_softctrl` user logic provides the Avalon-MM interface signals that are recognized by the component editor to communicate with other components. The exported signals (except `reset_srio_n`) in the conduit connect to the reconfiguration controller. The `reset_srio_n` signal connects to the main reset of the RapidIO MegaCore function.

For more information about creating a new component based on a Verilog HDL file in the SOPC Builder system, refer to *Volume 4: SOPC Builder* of the *Quartus II Handbook*.

RapidIO MegaCore Function

The RapidIO MegaCore function establishes a RapidIO link with the link partner. It also converts the transactions presented to it on the Avalon-MM interface to the corresponding RapidIO transactions and transmits them on the RapidIO serial link. The RapidIO MegaCore function converts the RapidIO transactions from the RapidIO serial link to I/O burst transfers and presents these burst transfers to the corresponding Avalon-MM slave or master ports.

In this reference design, the RapidIO MegaCore function has the following default parameter settings:

- Initial data rate: 2.5 GBaud
- Reference clock frequency: 156.25 MHz
- Mode selection: 1× Serial

Reconfiguration Controller

The reconfiguration controller block consists of the ALTGX_RECONFIG megafunction and the controller (user logic). The reconfiguration controller block receives instruction from the `srio_softctrl` user logic and reconfigures the transceiver.

ALTGX_RECONFIG Megafunction

The ALTGX_RECONFIG megafunction reconfigures the transceiver in the RapidIO MegaCore function. To reconfigure the transceiver, the ALTGX_RECONFIG megafunction must have the correct configuration settings to send to the RapidIO MegaCore function. A **.mif** contains these configuration settings. The reference design uses the Channel and CMU PLL reconfiguration mode. The `reconfig_mode_sel` port value is set to 101.

Controller (User Logic)

The controller includes user-defined logic to obtain the contents of **.mif** files from the assigned RAMs. During reconfiguration, the controller selects a **.mif** based on the `rate_reconfig[1:0]` input signal from the `srio_softctrl` user logic. The controller loads the contents of the **.mif** it selects in the ALTGX_RECONFIG megafunction.

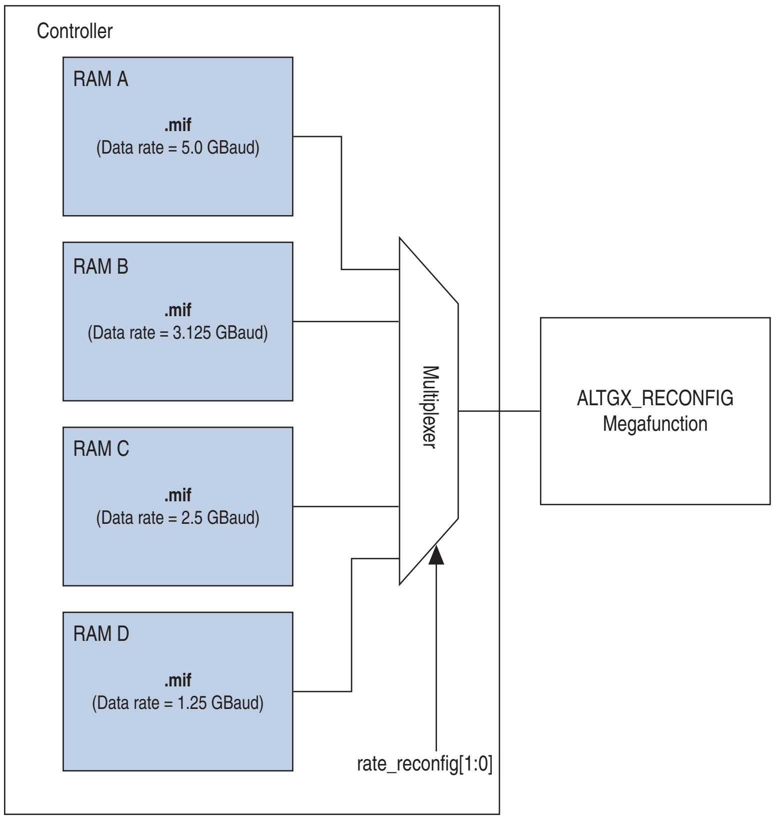
The reference design provides four **.mif** files that specify the transceiver configuration settings with different data rates—5.000, 3.125, 2.500, and 1.250 GBaud—as required by the *RapidIO Specification, 2.1*. The **.mif** files for this reference design are located in the **reconfig_mif** folder.

There are four RAMs; each RAM stores a **.mif**. When you run your design on the board, you can access the contents of the **.mif** from the RAMs.

For more information about instantiating multiple **.mif** files, refer to the *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Device Handbook*.

Figure 3 shows the architecture of the controller.

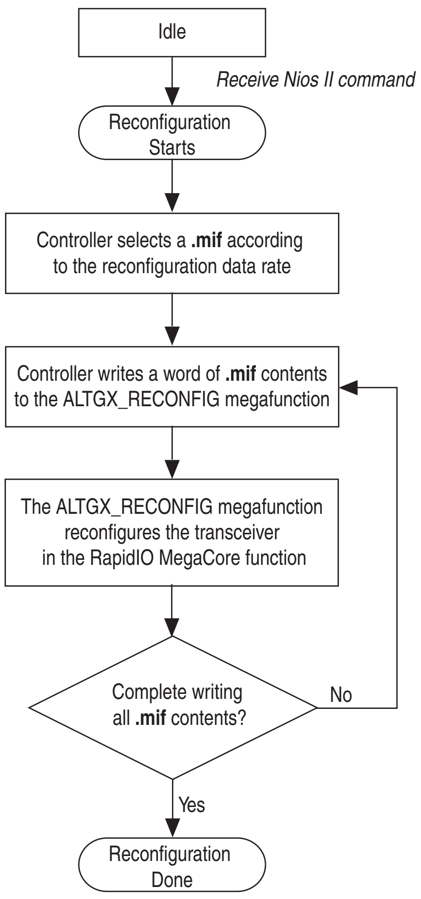
Figure 3. Controller Component Architecture



Dynamic Reconfiguration Process

Figure 4 shows the process flow for RapidIO MegaCore function dynamic data rate reconfiguration.

Figure 4. Dynamic Data Rate Reconfiguration Process Flow


Idle
Receive Nios II command
Reconfiguration Starts
Controller selects a .mif according to the reconfiguration data rate
Controller writes a word of .mif contents to the ALTGX_RECONFIG megafunction
The ALTGX_RECONFIG megafunction reconfigures the transceiver in the RapidIO MegaCore function
Complete writing all .mif contents?
No
Yes
Reconfiguration Done

When you type the `rate_reconfig` <*desired data rate*> command in the nios2-terminal, the Nios II processor sends the command to the `srio_softctrl` user logic through the Avalon-MM interface to start the reconfiguration process. When you type the software command `rate_reconfig`<*desired data rate*> in the nios2-terminal, the `srio_softctrl` user logic asserts the `reconfig_ena` signal. The `reconfig_ena` signal starts the reconfiguration process, and encodes the desired data rate in the `rate_reconfig[1:0]` signal. The `rate_reconfig[1:0]` signal is a multiplexer select signal that determines which **.mif** is selected.

After the controller selects an appropriate **.mif**, it load the contents of the **.mif** to the ALTGX_RECONFIG megafunction. The ALTGX_RECONFIG megafunction reconfigures the transceiver in the RapidIO MegaCore Function with the new transceiver settings through the `reconfig_togxb` signal.

When the controller loads the **.mif**, it asserts write signals to the ALTGX_RECONFIG megafunction. In each write cycle, the controller writes one word of the **.mif** contents to the ALTGX_RECONFIG megafunction. This process repeats until the controller completes writing all the contents to the ALTGX_RECONFIG megafunction.

When the controller completes writing the **.mif** contents, the ALTGX_RECONFIG megafunction asserts the `reconfig_done` signal to indicate that the reconfiguration process has completed successfully.

Internal Signals

Table 1 lists the internal reconfiguration signals for the dynamic data rate reconfiguration components shown in Figure 2 on page 4.

Table 1. Internal Signals for Dynamic Data Rate Reconfiguration Components

Signals	Descriptions
`rate_reconfig[1:0]`	A 2-bit control signal that indicates the data rate to which to reconfigure the RapidIO MegaCore function transceiver. 2'b00 = 5.000 GBaud 2'b01 = 3.125 GBaud 2'b10 = 2.500 GBaud 2'b11 = 1.250 GBaud
`reconfig_ena`	Enables the reconfiguration process. When the `srio_softctrl` user logic asserts this signal, the value of the `rate_reconfig` signal is valid.
`reconfig_done`	A status signal that indicates that the reconfiguration process has completed and the reconfiguration controller has returned to idle state.
`reconfig_address[5:0]`	Provides the **.mif** content address to the controller to obtain a specific word of the **.mif** contents.
`write`	A continuous pulse signal that enables a word of the **.mif** contents to be written into the ALTGX_RECONFIG megafunction in each pulse.
`reconfig_data[15:0]`	Loads reconfiguration **.mif** content data from the controller to the ALTGX_RECONFIG megafunction.
`reconfig_togxb`	Reconfiguration signal bus from the ALTGX_RECONFIG megafunction that reconfigures the RapidIO MegaCore function transceiver.
`reconfig_fromgxb`	Reconfiguration signal bus output from the RapidIO MegaCore function transceiver that provides reconfiguration data to the ALTGX_RECONFIG megafunction.
`reset_srio_n`	An active-low signal that resets the RapidIO MegaCore function when triggered by one of the following conditions: ■ Software reset (16 Avalon-MM system clock cycles) ■ Hardware reset ■ Reset during reconfiguration process (holds reset until the process is completed)

Software Application for Nios II Embedded Processor

The C programming file **srio_main_full.c** contains the software application for the Nios II embedded processor. The software application file enables you to issue software commands to the Nios II embedded processor to perform various tasks such as performing dynamic data rate reconfiguration. The **srio_main_full.c** file is located in the **software_app** folder and is downloaded to the Nios II embedded processor when you run the reference design.

Table 2 lists the software commands used for this reference design. To instruct the Nios II embedded processor and other sub-system modules to perform a particular task, type the software command in the nios2-terminal.

Table 2. Software Commands for the Nios II Embedded Processor

Software Command	Description
`h`	Lists out all available instructions.
`start`	The packet source module starts sending data packets to the RapidIO MegaCore function.
`stop`	The packet source module stops sending data packets to the RapidIO MegaCore function.
`init`	Initialization of the RapidIO internal registers. For more information about specific registers that are configured, please refer to the **srio_main_full.c** file.
`link`	Checks the status of the RapidIO link. If link is up, returned command = Link Up; if link is down, returned command = Link Down
`r`	Software reset. Resets the RapidIO MegaCore function.
`rate_reconfig`<*desired data rate*>	Starts data rate reconfiguration process. After reconfiguration, the RapidIO MegaCore function operates at desired data rate.

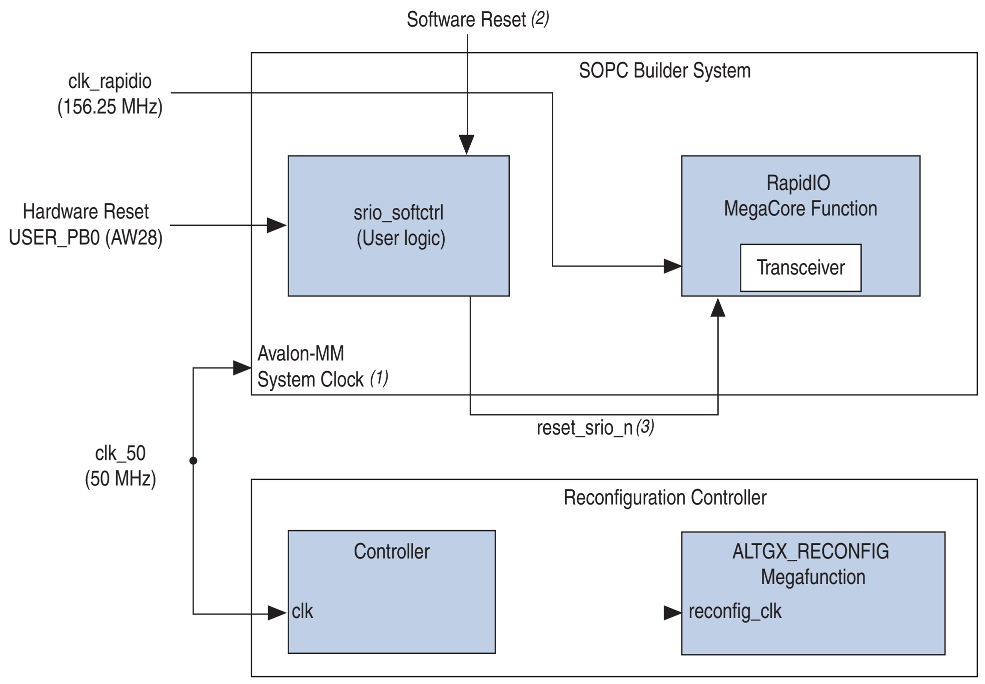
Clocking and Reset

The clocking and reset systems in the reference design include a 156.25-MHz transceiver clock and a 50-MHz frequency clock. You can obtain the clock source for these two clock frequencies from the Stratix IV GX Signal Integrity Development Board. The 156.25-MHz transceiver clock connects to a dedicated 156.25-MHz transceiver clock (G38) and the 50-MHz clock connects to a dedicated 50-MHz core clock (AR22) on the board.

Connect the 156.25-MHz transceiver clock to the transceiver in the RapidIO MegaCore function for high-speed data transmission. The Avalon-MM system clock operates with the 50-MHz frequency clock (`clk_50`), as shown in Figure 5. The `clk_50` clock connects to the `reconfig_clk` and `clk` signals.

Figure 5 shows the clocking and reset systems that support dynamic data rate reconfiguration for the RapidIO MegaCore function.

Figure 5. Clocking and Reset Systems of Dynamic Data Rate Reconfiguration



Notes to Figure 5:

(1) The Avalon-MM system clock is connected to the components in the SOPC Builder system.

(2) The software reset is the software command from the nios2-terminal.

(3) This is an active-low signal. Any of the following conditions trigger this reset signal: hardware reset, software reset, or reset following reconfiguration.

The reference design connects the clock inputs to `clk_50`. You may also connect these clocks independently to different sources for your design.

The reconfiguration process requires 7,800 clock cycles to complete. When the reconfiguration process starts, the RapidIO MegaCore function enters reset automatically and its RapidIO link goes down. During reset, the FIFO buffers in the RapidIO MegaCore function are cleared as well. The clearing of the FIFO buffers may create potential data loss if any packets are pending processing (before transmit or after receive) in the RapidIO MegaCore function.

☞ Before reconfiguration, Altera recommends that you refrain from sending data to the RapidIO MegaCore function and that you wait to receive all pending response packets from the RapidIO MegaCore function, to prevent data loss.

The reconfiguration process completes after the ALTGX_RECONFIG megafunction reasserts the `reconfig_done` signal. The reset state machine in the RapidIO MegaCore function then begins the link recovery process. The reset state machine ensures that the phase-locked loop (PLL) and frequency of the link are locked before the link is established. You can observe these reconfiguration and reset processes in the waveforms in Figure 10 through Figure 12 on page 21.

The `srio_reset_n` signal also controls the reset of the RapidIO MegaCore function following any of these events:

- Hardware reset with an external `reset_n` pin
- Software reset command

☞ These two resets reset the RapidIO MegaCore function without reconfiguring the transceiver embedded in the RapidIO MegaCore function.

Using the Reference Design

The following sections describe how to set up and use the reference design:

- "Hardware Requirements"
- "Software Requirements" on page 12
- "Installing the Reference Design" on page 12

Hardware Requirements

The reference design uses the Stratix® IV GX EP4SGX230KF40C3ES Signal Integrity Transceiver development board to demonstrate the dynamic data rate reconfiguration process. You can verify the reconfiguration process using one of the following methods:

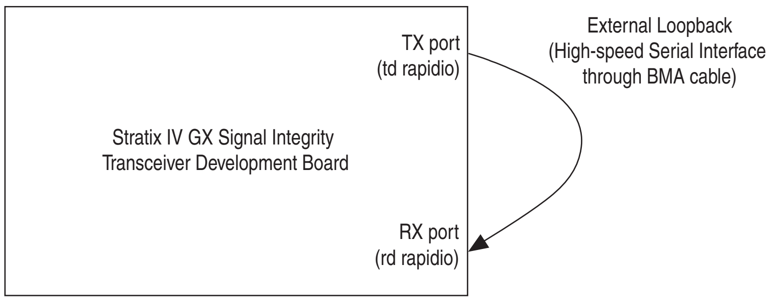
- Single-Board Connection
- Dual-Board Connection

Single-Board Connection

In the single-board connection method, the TX and RX ports connect through external loopback, and share the same clock source and transceiver channel. When the ALTGX_RECONFIG megafunction dynamically reconfigures the transceiver with the desired data rates, it changes the channel and PLL settings of the TX and RX ports. The data rate of the TX and RX ports change simultaneously. After programming the desired data rates in the transceiver, the RapidIO MegaCore function reestablishes the RapidIO link at the new operating data rate.

Figure 6 shows the single-board connection.

Figure 6. Single-Board Connection


External Loopback
(High-speed Serial Interface
through BMA cable)
TX port
(td rapidio)
Stratix IV GX Signal Integrity
Transceiver Development Board
RX port
(rd rapidio)

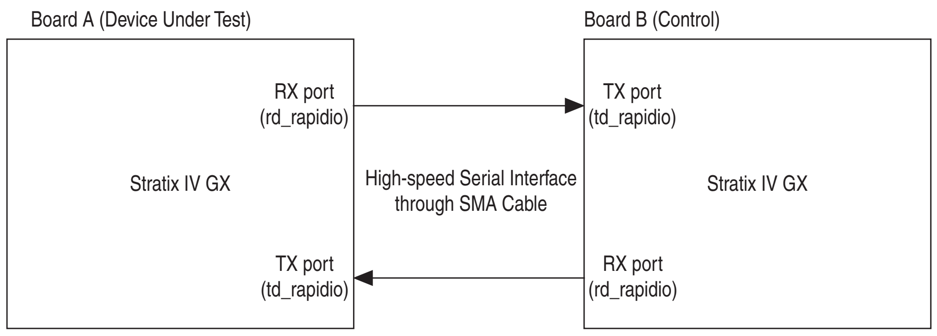
Dual-Board Connection

In the dual-board connection method, you use two Stratix IV GX EP4SGX230 Signal Integrity Transceiver development boards to demonstrate the design. The first board (board A) is a test board, and the second board (board B) is a control board.

Figure 7 shows the connection for the TX and RX ports of boards A and B. You program the same design on board A and board B. The default data rate on both boards is 2.5 GBaud. As the initial data rates of the RapidIO MegaCore functions on board A and board B are identical, the link can be established.

Figure 7 shows the dual-board connections.

Figure 7. Dual-Board Connection



You can program and download the reference design on board B with a JTAG cable. After programming and downloading the reference design on board B, unplug the JTAG cable and retain board B as a control board. Then, program and download the reference design on board A. These two boards use SMA connectors as high-speed RapidIO link. The SMA connectors are :

- TX port: GXB1 - TX4
  - `td_rapidio(p)` - AD36
  - `td_rapidio(n)` - AD37
- RX port: GXB1 - RX4
  - `rd_rapidio(p)` - AE38
  - `rd_rapidio(n)` - AE39

Software Requirements

The reference design requires the following software:

- The Quartus® II software version 9.1
- RapidIO MegaCore version 9.1
- Nios II Embedded Design Suite version 9.1

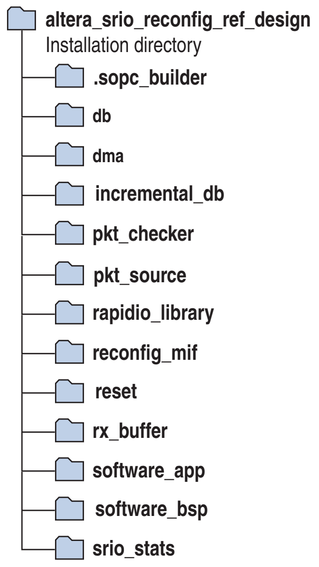
Installing the Reference Design

This section describes how to install the reference design.

The **altera_srio_reconfig_ref_design.zip** file contains all the files necessary for this reference design.

Unzip the **altera_srio_reconfig_ref_design.zip** file in the working directory you designate for this project. After you unzip the file, your working directory contains the subdirectories shown in Figure 8.

Figure 8. Directory Structure



Running the Reference Design

To run the reference design, follow these steps:

1. Connect the JTAG cable to your device. If you are using dual-board connection, connect the JTAG cable to board B.
2. To start a Nios II command shell, on the Windows Start menu, point to **All Programs** > **Altera** > **Nios II EDS** <*version_number*>, and click **Nios II** <*version_number*> **Command Shell**.
3. Navigate to the **/software_bsp** folder.
4. To execute the script that builds the Hardware Abstraction Layer (HAL) drivers required by this reference design, type the following command:

   ```
   ./create-this-bsp ↵
   ```

5. Navigate to the **/software_app** folder. This folder contains the **srio_main_full.c**, **srio_regs.h**, and **create-this-app** files.
6. To execute the script that compiles the driver software in **srio_main_full.c**, type the following command:

   ```
   ./create-this-app ↵
   ```

7. To program your device, download the software image, and start a Nios II terminal session, follow these steps:

   a. To program your device, in the command shell, type the following command:

      ```
      nios2-configure-sof -d 1 ../srio_2500_x1.sof ↵
      ```

   b. To download the software image to your device, type the following command:

      ```
      nios2-download --device=1 -g srio_test.elf ↵
      ```

   c. To communicate with the Nios II processor on your device, start a Nios II terminal session by typing the following command:

      ```
      nios2-terminal --device=1 ↵
      ```

   d. If you are using dual-board connection, now connect the JTAG cable from board B to board A. Repeat steps a to c for board A in the same Nios II command shell.

8. Start the Quartus II software. In the Tools menu, click **SignalTap II Logic Analyzer** to open the SignalTap^{TM} Embedded Logic Analyzer file **stp1.stp**. The **stp1.stp.** file contains a list of internal signals that change with the reconfiguration process. View the activity on these signals to confirm that reconfiguration occurred. You can also view these signals in Figure 10 on page 19 through Figure 12 on page 21.

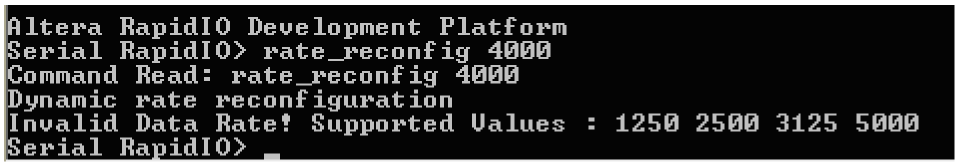
9. To reconfigure the data rate for the RapidIO MegaCore function, type the following command in the nios2-terminal:

   ```
   rate_reconfig <desired data rate> ↵
   ```

   ☞ The desired data rate can be `1250`, `2500`, `3125`, or `5000`.

If you try to reconfigure the RapidIO MegaCore function with the same data rate, the Nios II processor executes the reconfiguration and reset processes but the resulting data rate will still be the same. If you try to reconfigure the RapidIO MegaCore function with an unsupported data rate, the Nios II terminal displays the error shown in Example 1:

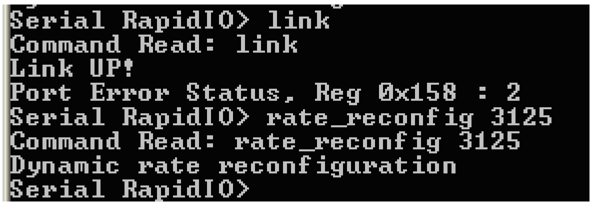
Example 1. Error



```
Altera RapidIO Development Platform
Serial RapidIO> rate_reconfig 4000
Command Read: rate_reconfig 4000
Dynamic rate reconfiguration
Invalid Data Rate! Supported Values : 1250 2500 3125 5000
Serial RapidIO> _
```

To check the current link status before and after rate reconfiguration, you can type `Link` in the nios2-terminal . If the link is established, the returned command shows `Link Up`, refer to Example 2. If the link goes down, the returned command shows `Link Down`.

Example 2. Successful Link Up



```
Serial RapidIO> link
Command Read: link
Link UP!
Port Error Status, Reg 0x158 : 2
Serial RapidIO> rate_reconfig 3125
Command Read: rate_reconfig 3125
Dynamic rate reconfiguration
Serial RapidIO>
```

If you use only one board to reconfigure, you can establish the link in all cases because the TX and RX ports are reconfigured with the same data rate. If you have two designs running on two boards with an established link, the link can be either up or down after the reconfiguration process, depending on whether the reconfigured desired data rate is the same as the data rate on the other port.

To use the software reset command, type `r` at the nios-2 terminal session.

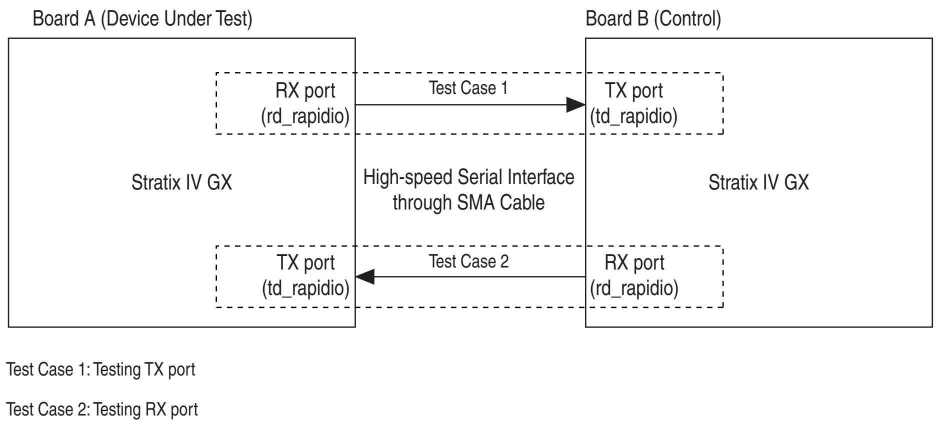
Test Cases for Verification (Dual-board Connection)

After running the reference design, you can perform test case 1 and test case 2 to verify the dynamic data rate reconfiguration process.

If the data rates of the link partners are not identical, RX port is unable to lock the frequency, and RapidIO link goes down. However, if the data rates of the link partners are identical, the RapidIO MegaCore function establishes a link successfully.

Figure 9 shows the dual-board connections for test cases 1 and 2.

Figure 9. Test Cases for Dual-board Connection



Test Case 1: Testing the TX Port

The initial data rate for test case 1 is 2.500 GBaud. The objective of this test case is to test the TX port of board A. If you reconfigure the initial data rate of board A to 1.250 GBaud, the link goes down because boards A and B are running at different data rates. The data rate at the TX port of board A is reconfigured to a desired data rate, but the data rate at the RX port of board B remains the same. Therefore, the data rates are unable to synchronize, and the link goes down.

You can monitor the test result at the `port_initialized` signal with the SignalTap II Embedded Logic Analyzer. If you reconfigure the data rate from 1.250 GBaud to 2.500 GBaud, you can monitor the reset sequence in the RapidIO MegaCore function, and observe when the corresponding link is reestablished.

Table 3 shows the test conditions for test case 1.

Table 3. Test Conditions for Test Case 1

Board	Steps	Link (port_initialized Signal)
A and B	Program with 2.500-GBaud design	Lock (1’b1)
A	Reconfigure from 2.500 GBaud to 1.2500 GBaud	Lost lock (1’b0)
A	Reconfigure from 1.250 GBaud to 2.500 GBaud	Lock (1’b1)

Test Case 2: Testing the RX Port

The initial data rate for test case 2 is 2.500 GBaud. The objective of this test case is to test the RX port. If you program the initial data rate of board B to 1.250 GBaud, board A with 2.500-GBaud data rate loses the link. Note that board B is programmed with a desired data rate and not reconfigured. You can prepare another **.sof** with a 1.250-GBaud data rate and program the **.sof** into board B. If the TX port of board B has a different data rate, the link goes down.

To reestablish the link, you must reconfigure board A with the same data rate; in this case 1.250 GBaud. If the desired data rate at the RX port of board A matches the data rate at the TX port of board B, the link is reestablished.

Table 4 shows the test conditions for test case 2.

Table 4. Test Conditions for Test Case 2

Board	Steps	Link (port_initialized Signal)
A and B	Program with 2.500-GBaud design	Lock (1'b1)
B	Program 1.250 GBaud **.sof** to board B	Lost lock (1'b0)
A	Reconfigure from 2.500 GBaud to 1.250 GBaud	Lock (1'b1)

For more information about the SignalTap II Embedded Logic Analyzer, refer to the *Design Debugging Using the SignalTap II Embedded Logic Analyzer* chapter in volume 3 of the *Quartus II Handbook*.

Waveform Examples

This section provides the SignalTap II signals and waveforms for the dynamic data rate reconfiguration process.

Table 5 describes the signals shown in Figure 10 through Figure 12 on page 21.

Table 5. Selected Signals Displayed at the SignalTap II Embedded Logic Analyzer (Part 1 of 2)

Signals	Descriptions
`port_initialized`	Indicates that the serial RapidIO initialization sequence has completed successfully. The link is established.
`reset_state_machine`	Monitors the locked status of the transceiver PLL and frequency after reset. When the RapidIO MegaCore function locks to the transceiver PLL and frequency, the RapidIO link is established.
`softctrl|reset_srio_n` *(1)*	Resets when triggered by one of the following conditions: ■ Software reset ■ Hardware reset ■ Reconfiguration process
`softctrl|sys_mnt_s_......` *(1)*	Avalon-MM signals at the `srio_softctrl` user logic, coming from the SOPC Builder system (Nios II processor).

Table 5. Selected Signals Displayed at the SignalTap II Embedded Logic Analyzer (Part 2 of 2)

Signals	Descriptions
`reconfig|reconfig_ena` *(2)*	Enables the dynamic data rate reconfiguration process. Also indicates that the `rate_reconfig[1:0]` signal is valid. Output from the `srio_softctrl` user logic and input to the reconfiguration controller.
`reconfig|rate_reconfig` *(2)*	An internal signal that indicates the current reconfiguration data rate. When the the `srio_softctrl` user logic asserts the `reconfig_ena` signal, the `rate_reconfig` value indicates the current data rate to reconfigure and the value is valid. The following values indicate the current data rates: ■ 0h: 5.000 GBaud ■ 1h: 3.125 GBaud ■ 2h: 2.500 GBaud ■ 3h: 1.250 GBaud
`reconfig|reconfig_done` *(2)*	Indicates that the reconfiguration process is complete. Returns all states and output signals to their default values.
`controller|`...... *(3)*	Data, address, and signals from the **.mif** in the controller. These signals are input to the ALTGX_RECONFIG megafunction.

Notes to Table 5:

(1) Signals from the Nios II processor to the `srio_softctrl` user logic (Avalon bus in the SOPC Builder system).

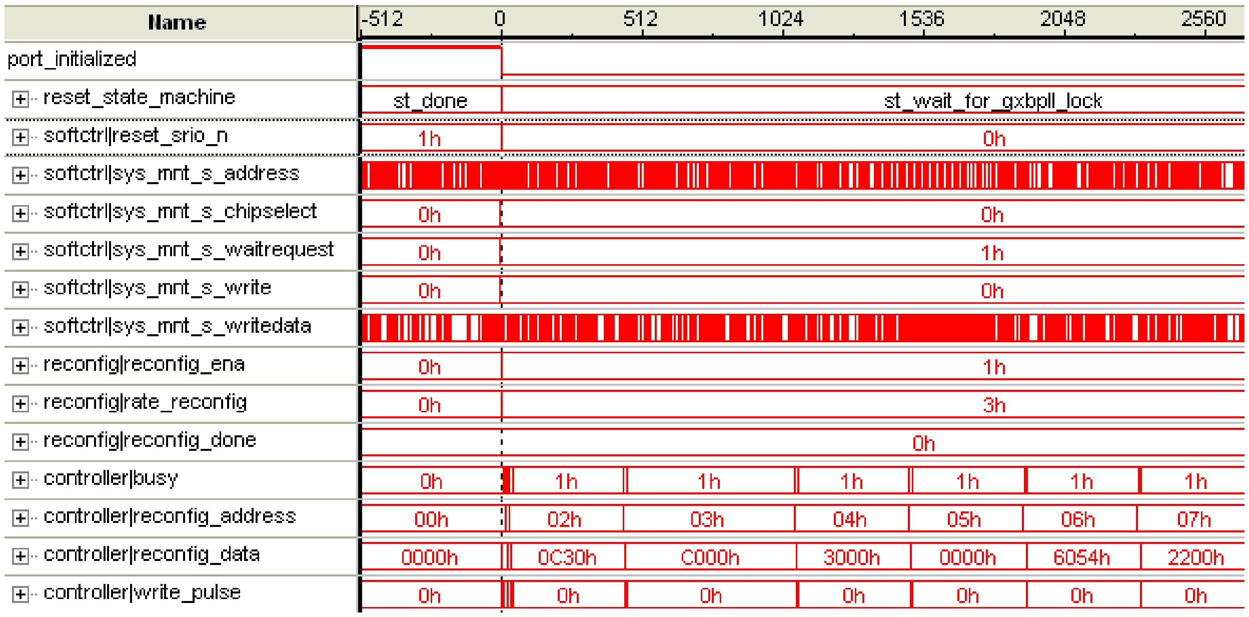
(2) Conduit signals from the `srio_softctrl` user logic (SOPC Builder system) to the controller.

(3) Signals from the controller to the ALTGX_RECONFIG megafunction.

Figure 10 through Figure 12 on page 21 show SignalTap II waveforms for dynamic data rate reconfiguration process.

Figure 10 shows the SignalTap II Logic Analyzer waveform example at the beginning of the dynamic data rate reconfiguration process. You can obtain this waveform by setting the `reconfig_ena` signal as the trigger point in the SignalTap file.

Figure 10. Beginning of the Dynamic Data Rate Reconfiguration Process ***(Note 1)***



Note to Figure 10:

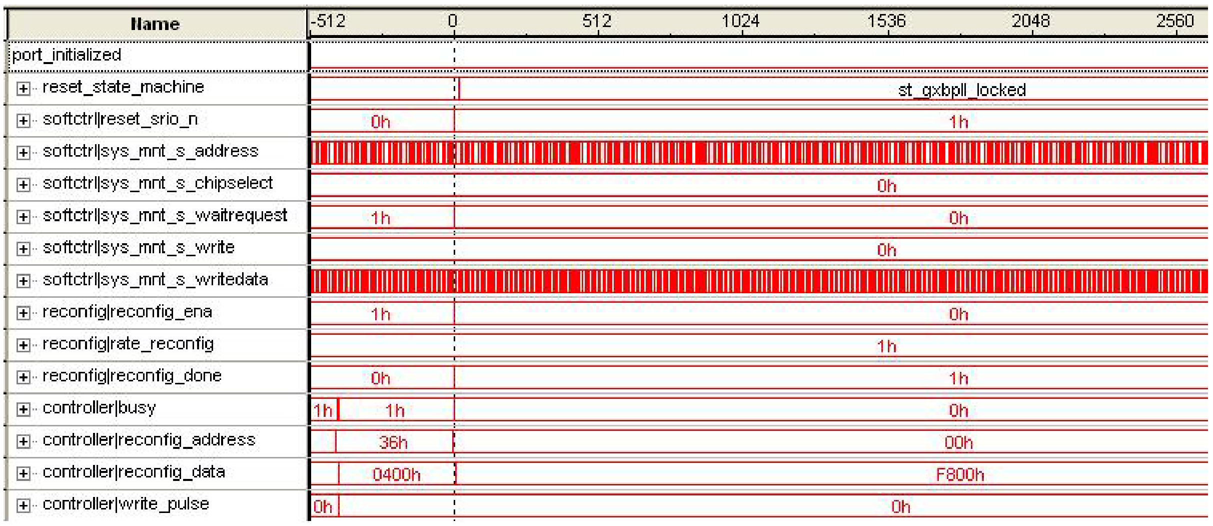
(1) Instead of representing 5.0 GBaud as stated in Table 5 on page 17, the initial value for `rate_reconfig` signal is 0h due to the register reset. The value for the `rate_reconfig` signal is only valid when the `reconfig_ena` signal (1'b1) is asserted, enabling the `rate_reconfig` signal to be sent to the controller for processing.

If you want to start the reconfiguration process, type the software command `rate_reconfig`<*desired data rate*> in the nios2-terminal. When you type software command, the Nios II embedded processor prompts the `srio_softctrl` user logic to assert the `reconfig_ena` signal. The `rate_reconfig` signal is reconfigured to 1.250 GBaud (3h). After the controller receives the `rate_reconfig` and `reconfig_ena` signals, it starts a series of write pulses to the ALTGX_RECONFIG megafunction. With every write pulse generated at the controller, the `reconfig_data` and `reconfig_address` signals load the contents of the selected **.mif** in the ALTGX_RECONFIG megafunction and reconfigure the transceiver.

When the reconfiguration process starts, the `srio_softctrl` user logic sends the reset signal to reset the RapidIO MegaCore function. The RapidIO link goes down and deasserts the `port_initialized` signal. The reset holds until the reconfiguration process completes.

Figure 11 shows a waveform example of the SignalTap II Logic Analyzer after the dynamic data rate reconfiguration process completes. You can obtain this waveform by setting the `reconfig_done` signal as the trigger point in the SignalTap file.

Figure 11. After Dynamic Data Reconfiguration Process Completes

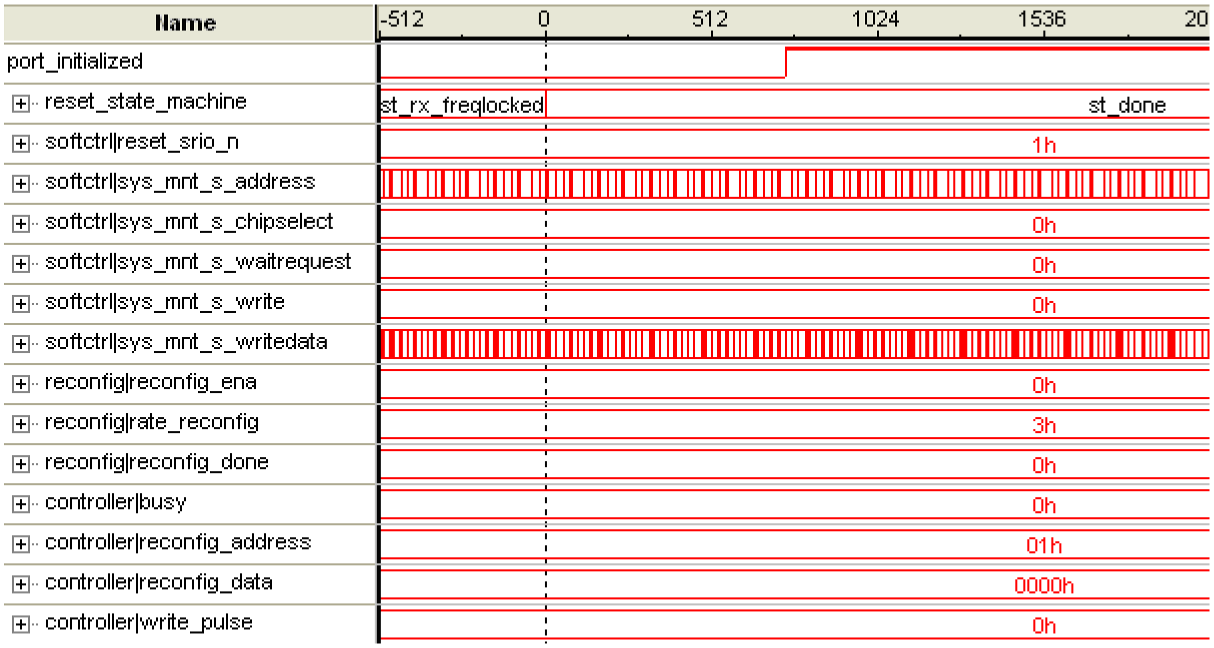


When the ALTGX_RECONFIG megafunction receives the contents of the selected **.mif** from the controller, it reconfigures the transceiver in the RapidIO MegaCore function through the `reconfig_togxb` signal. The ALTGX_RECONFIG megafunction writes all the contents in the **.mif** to complete the reconfiguration process. After the ALTGX_RECONFIG megafunction fully reconfigures the transceiver, it asserts the `reconfig_done` signal to indicate that the reconfiguration process has completed. The state machine in the `srio_softctrl` user logic deasserts the `reconfig_ena` signal, and both the `srio_softctrl` user logic and reconfiguration controller return to idle state.

When the reconfiguration process completes, the `reset_srio_n` signal releases the reset on the RapidIO MegaCore function. The RapidIO MegaCore function then tries to lock to the transceiver PLL and frequency to reestablish the link.

Figure 12 shows a waveform example of the SignalTap II Logic Analyzer when the link is reestablished. You can obtain this waveform by setting the `port_initialized` signal as the trigger point in the SignalTap file.

Figure 12. When Link is Reestablished



If the data rates are identical on the TX and RX ports, the RapidIO MegaCore function locks to the transceiver PLL and frequency to reestablish the link. The RapidIO MegaCore function asserts the `port_initialized` signal to indicate the link is established and ready. The RapidIO MegaCore function now transmits and receives data at the desired data rate.

Enabling Reconfiguration

This section provides the additional information you may need when implementing the reconfiguration process for your own design.

☞ The reference design provides an SRAM Object File (**.sof**) that includes the end result following these steps.

To reconfigure the transceiver in the RapidIO MegaCore function, follow these steps:

1. Instantiate the RapidIO MegaCore function in the SOPC Builder system.
2. Generate the SOPC Builder system.
3. Under the rapidio hierarchy, click **rapidio_riophy_gxb** to launch the ALTGX parameter editor.
4. To enable the ALTGX_RECONFIG megafunction to reconfigure the Channel and CMU PLL in the transceiver, in the **Reconfiguration settings** tab of the ALTGX_RECONFIG parameter interface, turn on **Enable Channel and Transmitter PLL reconfiguration**.

5. Click **Finish**. The transceiver file regenerates.

To instantiate the ALTGX_RECONFIG megafunction and select the Channel and CMU PLL reconfiguration mode, follow these steps:

1. On the Tools menu, click **MegaWizard Plug-In Manager**.
2. Click **Next**.
3. In the megafunctions list, click the **+** icon next to the **I/O category**, and then select **ALTGX_RECONFIG**. Click **Next**.
4. In the **Reconfiguration settings** tab of the ALTGX_RECONFIG parameter editor, turn on **Channel and TX PLL select/reconfig**.

The ALTGX_RECONFIG megafunction exchanges data with the transceiver in the RapidIO MegaCore function through the `reconfig_togxb` and `reconfig_fromgxb` signals. The other signals connect to the controller to obtain reconfiguration data from the selected **.mif**.

For more information about the ALTGX_RECONFIG megafunction, refer to the *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Device Handbook*.

If your design requires different transceiver settings (for example, if your design uses a different clock frequency than the frequency specified in the reference design), Altera recommends regenerating four **.mif** files with the desired transceiver settings for four data rates. You must regenerate the **.mif** files because the **.mif** files in the **reconfig_mif** folder contain the transceiver settings for this RapidIO MegaCore function only.

For more information about instantiating multiple **.mif** files, refer to the *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Device Handbook*.

Document Revision History

Table 6 shows the revision history for this application note.

Table 6. Document Revision History

Date	Version	Changes
December 2010	1.0	Initial release.