LogiCORE™ IP Virtex®-4 FX FPGA RocketIO™ Multi-Gigabit Transceiver Wizard v1.7

Getting Started Guide

UG246 (v1.7) April 19, 2010

XILINX®

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
02/10/06	1.0	Initial Xilinx release.
06/06/06	1.1	New Wizard software revision 1.1. New v1.1 screen captures.
09/12/06	1.2	New Wizard software revision 1.2. New v1.2 screen captures. Change state of Bypass TX FIFO option. Revise Installation procedure.
11/30/06	1.3	New Wizard software revision 1.3. New v1.3 screen captures.
03/01/07	1.4	Standardize document name. New Wizard software revision 1.4. New v1.4 screen captures.
11/21/07	1.5	Wizard v1.5 documentation not released.
03/24/08	1.6	Wizard v1.6 release. Wizard v1.6 software revision and v1.6 screen captures. Major revision to Chapter 2, “Installation and Licensing” sections:“System Requirements,” and “Before You Begin.” Major revisions to Chapter 3, “Running the Wizard” in: “Creating a Directory” and “Simulating the Example Design.” Added Appendix A, “References.”
03/24/08	1.6.1	Move trademark symbol in Preface, first paragraph. Remove version numbers from referenced document titles. Change “test bench” to two words.
05/16/08	1.6.2	Correct bad URL link to DS138.
04/19/10	1.7	Wizard 1.7 release. Renamed to *Virtex-4 FX FPGA RocketIO Multi-Gigabit Transceiver Wizard Getting Started Guide*. Updated Wizard and tool versions and the screen captures.

Table of Contents

Appendix A: References

About This Guide

This guide describes the function and operation of the LogiCORE™ IP Virtex®-4 FX FPGA RocketIO Multi-Gigabit Transceiver (MGT) Wizard.

Contents

This guide contains the following chapters:

- Preface, "About this Guide" introduces the organization and purpose of this guide, a list of additional resources, and the conventions used in this document.
- Chapter 1, "Introduction"describes the wrapper core and related information, including additional resources, technical support, and submitting feedback to Xilinx.
- Chapter 2, "Installation and Licensing" provides information about installing and licensing the Virtex-4 FX FPGA RocketIO MGT Wizard.
- Chapter 3, "Running the Wizard" provides an overview of the Virtex-4 FX FPGA RocketIO MGT Wizard, and gives a step-by-step tutorial on how to generate a sample Virtex-4 FX FPGA RocketIO MGT Wizard wrapper with the CORE Generator™ tool.

Additional Resources

To find additional documentation, see the Xilinx website at:

http://www.xilinx.com/support/documentation/index.htm.

To search the Answer Database of silicon, software, and IP questions and answers, or to create a technical support WebCase, see the Xilinx website at:

http://www.xilinx.com/support/mysupport.htm.

Conventions

This document uses the following conventions. An example illustrates each convention.

Typographical

The following typographical conventions are used in this document:

Convention	Meaning or Use	Example
`Courier font`	Messages, prompts, and program files that the system displays	`speed grade: - 100`
`Courier bold`	Literal commands that you enter in a syntactical statement	**`ngdbuild`** *design_name*
Helvetica bold	Commands that you select from a menu	**File → Open**
	Keyboard shortcuts	**Ctrl+C**
Italic font	Variables in a syntax statement for which you must supply values	**`ngdbuild`** *design_name*
	References to other manuals	See the *Development System Reference Guide* for more information.
	Emphasis in text	If a wire is drawn so that it overlaps the pin of a symbol, the two nets are *not* connected.
Square brackets [ ]	An optional entry or parameter. However, in bus specifications, such as **`bus[7:0]`**, they are required.	**`ngdbuild`** [*`option_name`*] *`design_name`*
Braces { }	A list of items from which you must choose one or more	**`lowpwr =`**{**`on`**|**`off`**}
Vertical bar |	Separates items in a list of choices	**`lowpwr =`**{**`on`**|**`off`**}

Vertical ellipsis
.
.
.
	Repetitive material that has been omitted	`IOB #1: Name = QOUT'`
		`IOB #2: Name = CLKIN'`
		.
		.
		.

Horizontal ellipsis ...	Repetitive material that has been omitted	**`allow block`** *block_name loc1 loc2 ... locn;*

Online Document

The following conventions are used in this document:

Convention	Meaning or Use	Example
Blue text	Cross-reference link to a location in the current document	See the section "Additional Resources" for details. Refer to "Title Formats" in Chapter 1 for details.
Blue, underlined text	Hyperlink to a website (URL)	Go to http://www.xilinx.com for the latest speed files.

Chapter 1

Introduction

This chapter introduces the Virtex-4 FX FPGA RocketIO MGT Wizard and provides related information, including additional resources, technical support, and submitting feedback to Xilinx.

The Virtex-4 FX FPGA RocketIO MGT Wizard is a CORE Generator™ tool designed to support both Verilog and VHDL design environments. In addition, the example design delivered with the core is provided in either Verilog or VHDL.

The Wizard produces a wrapper that instantiates one or more properly configured RocketIO MGTs for custom applications (see Figure 1-1).

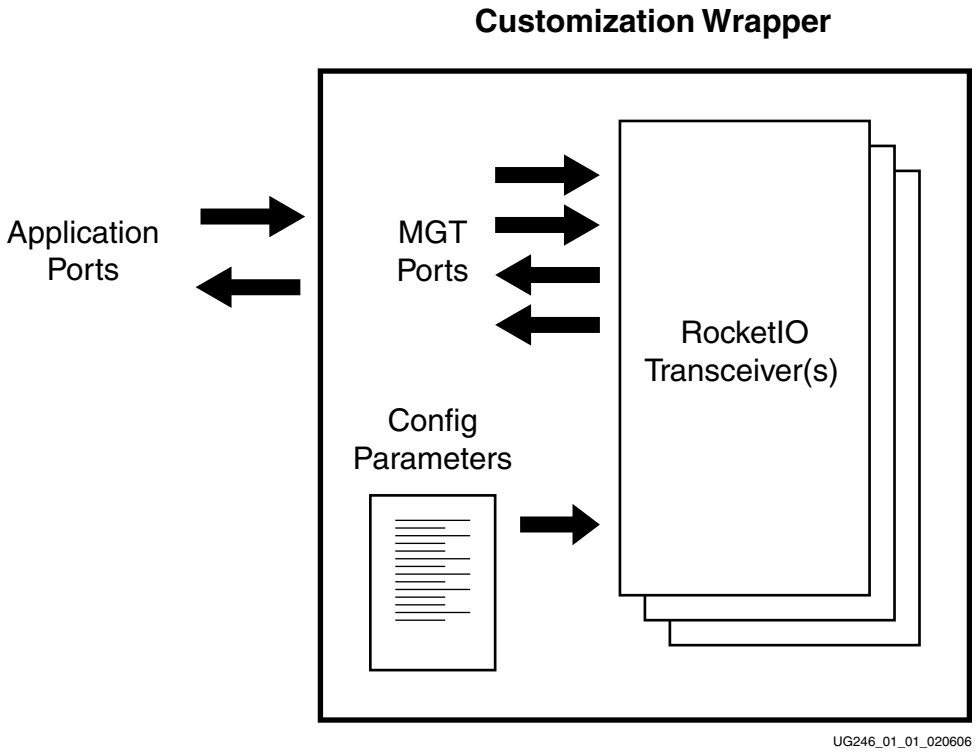

Figure 1-1: **MGT Wizard Wrapper**

About the Wizard

The Virtex-4 FX FPGA RocketIO MGT Wizard is a Xilinx CORE Generator tool, available at the Xilinx IP Center. For information about system requirements, installation, and licensing options, see Chapter 2, "Installation and Licensing."

Additional Wizard Resources

For detailed information and updates about the Virtex-4 FX FPGA RocketIO MGT Wizard, see the following documents, located at the Architecture Wizards page.

- *LogiCORE IP Virtex-4 FX FPGA RocketIO Multi-Gigabit Transceiver Wizard Data Sheet (DS138), [Ref 3]*
- *Virtex-4 FX FPGA RocketIO MGT Wizard Release Notes*

Technical Support

For technical support, go to www.xilinx.com/support. Questions are routed to a team of engineers with expertise using the Virtex-4 FX FPGA RocketIO MGT Wizard.

Xilinx provides technical support for use of this product as described in the *LogiCORE IP Virtex-4 FX FPGA RocketIO Multi-Gigabit Transceiver Wizard Getting Started Guide*. Xilinx cannot guarantee timing, functionality, or support of this product for designs that do not follow these guidelines.

Feedback

Xilinx welcomes comments and suggestions about the Virtex-4 FX FPGA RocketIO MGT Wizard and the accompanying documentation.

Virtex-4 FX FPGA RocketIO MGT Wizard

For comments or suggestions about the Virtex-4 FX FPGA RocketIO MGT Wizard, please submit a WebCase from www.xilinx.com/support. Be sure to include the following information:

- Product name
- Wizard version number
- List of parameter settings
- Explanation of your comments, including whether the case is requesting an *enhancement* (you believe something could be improved) or reporting a *defect* (you believe something isn't working correctly).

Document

For comments or suggestions about this document, please submit a WebCase from www.xilinx.com/support. Be sure to include the following information:

- Document title
- Document number
- Document revision number
- Page number(s) to which your comments refer
- Explanation of your comments, including whether the case is requesting an *enhancement* (you believe something could be improved) or reporting a *defect* (you believe something isn't documented correctly).

XILINX®

Chapter 2

Installation and Licensing

This chapter provides instructions to install the Virtex®-4 FX FPGA RocketIO MGT Wizard in the Xilinx® CORE Generator™ tool. It is not necessary to obtain a license to use the Wizard.

System Requirements

Windows

- Windows XP Professional 32-bit/64-bit
- Windows Vista Business 32-bit

Linux

- Red Hat Enterprise Linux WS v4.0 32-bit/64-bit
- Red Hat Enterprise Desktop v5.0 32-bit/64-bit (with Workstation Option)
- SUSE Linux Enterprise (SLE) desktop and server v10.1 32-bit/64-bit

Software

- ISE® 12.1
- Mentor Graphics ModelSim v6.5c and above

Check the release notes for the required Service Pack; ISE Service Packs can be downloaded from www.xilinx.com/support/download.htm.

Before You Begin

Before installing the Wizard, you must have a MySupport account and the ISE 12.1 software installed on your system. If you already have an account and have the software installed, go to "Verifying Your Installation," page 14, otherwise, do the following:

1. Click **Login** at the top of the Xilinx home page then follow the onscreen instructions to create a MySupport account.
2. Install the ISE 12.1 software. For the software installation instructions, refer to the ISE 12.1 Design Suite Release Notes and Installation Guide available in ISE software Documentation [Ref 4].

Installing the Wizard

The Virtex-4 FX FPGA RocketIO MGT Wizard is included with the ISE 12.1 software. See "Before You Begin," page 13, for installation details.

Verifying Your Installation

Use the following procedure to verify that you have successfully installed the Virtex-4 FX FPGA RocketIO MGT Wizard in the CORE Generator tool.

1. Start CORE Generator.
2. After creating a new Virtex-4 family FX project or opening an existing one, the IP core functional categories appear at the left side of the window, as shown in Figure 2-1.

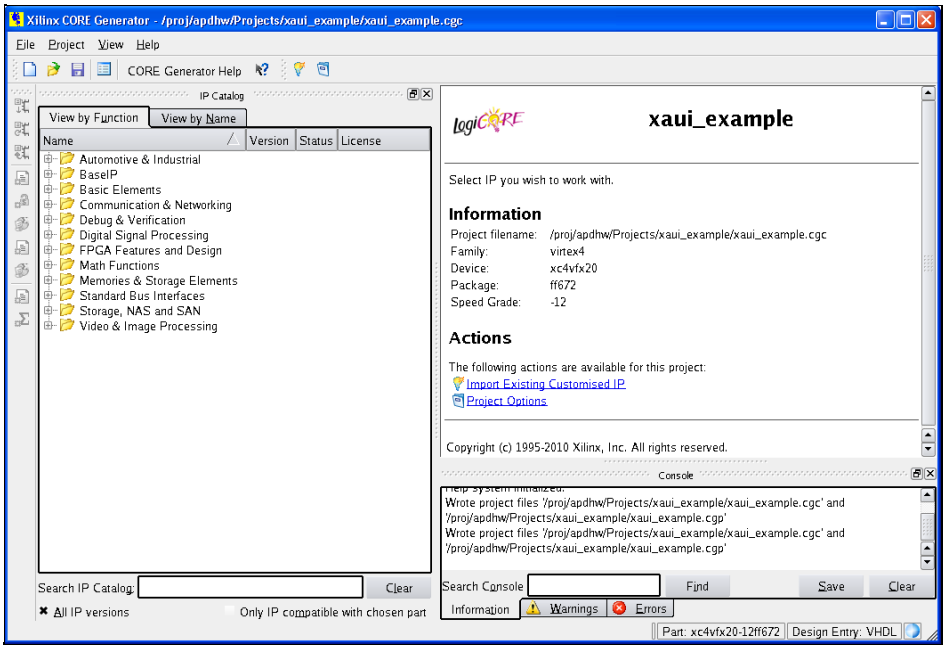

Figure 2-1: **CORE Generator Window**

3. Click to expand or collapse the view of individual functional categories, or click the **View by Name** tab at the top of the list to see an alphabetical list of all cores in all categories.
4. Determine if the installation was successful by verifying that *RocketIO Wizard v1.7* appears at the following location in the Functional Categories list:
   `/FPGA Features and Design/IO Interfaces`

   For additional assistance installing the IP Update, visit the Xilinx Support website at www.xilinx.com/support.

Chapter 3

Running the Wizard

Overview

This section provides a step-by-step procedure for generating a Virtex®-4 FX FPGA RocketIO MGT Wizard wrapper, implementing the core in hardware using the accompanying example design, and simulating the core with the provided example test bench.

The example design covered in this section is a wrapper that configures a group of RocketIO MGTs for use in a XAUI application. Guidelines are also given for incorporating the wrapper in a design and for the expected behavior in operation.

The XAUI example consists of the following components:

- A single Virtex-4 FX FPGA RocketIO MGT Wizard wrapper implementing a four-lane XAUI port using four RocketIO MGTs
- A demonstration test bench to drive the example design in simulation
- An example design providing clock signals and connecting an instance of the XAUI wrapper with modules to drive and monitor the wrapper in hardware, including optional ChipScope™ Pro modules.
- Scripts to synthesize and simulate the example design

The Virtex-4 FX FPGA RocketIO MGT Wizard example design has been tested with Synplify Pro D-2009.12 and XST 12.1 for synthesis, and ModelSim 6.5c for simulation.

Figure 3-1 shows a block diagram of the default XAUI example design

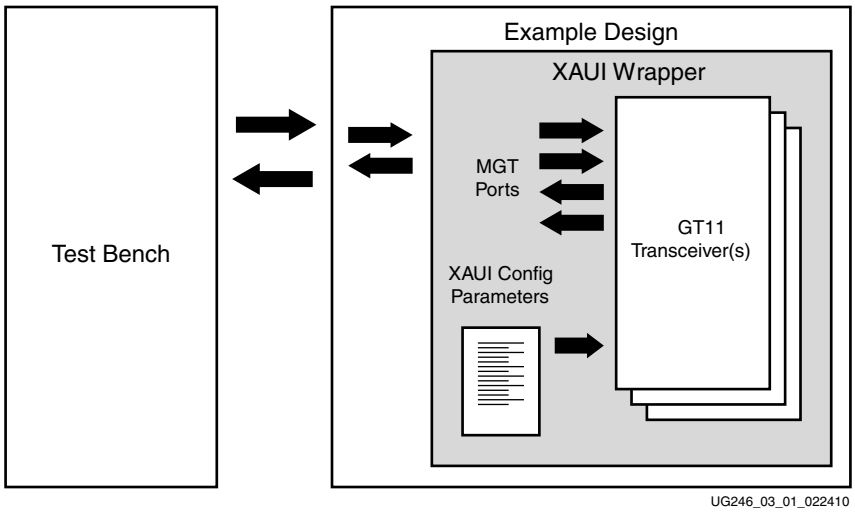

Figure 3-1: **Example Design**

Setting Up the Project

Before generating the example design, set up the project as shown in "Creating a Directory" and "Setting the Project Options."

Creating a Directory

To set up the example project, first create a directory using the following steps:

1. Change directory to the desired location. This example uses the following location and directory name:

   `/Projects/xaui_example`

2. Start the CORE Generator tool.

   For help starting and using the CORE Generator tool, see CORE Generator Help, available in ISE software documentation [Ref 4].

3. Choose **File** → **New Project** (Figure 3-2).
4. Optionally change the name of the `.cgp` file.
5. Click **Save**.

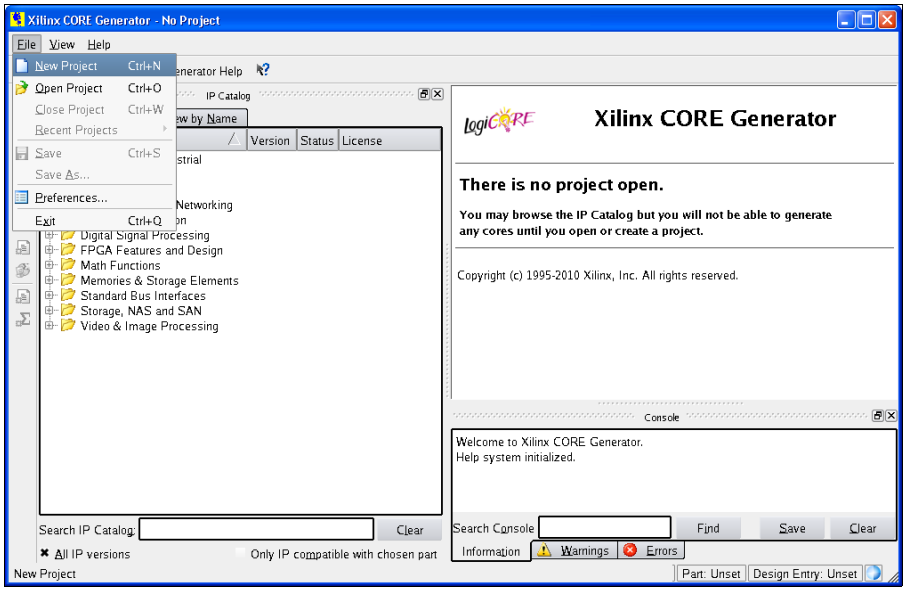

Figure 3-2: **Starting a New Project**

Setting the Project Options

Set the project options using the following steps:

1. Click the **Part** tab.
2. Select **Virtex4** from the Family list.
3. Select a device from the Device list which supports RocketIO MGTs.
4. Select an appropriate package from the Package list. This example uses the XC4VFX20 device (see Figure 3-3).

   Note: If an unsupported silicon family is selected, the Virtex-4 FX FPGA RocketIO MGT Wizard appears light grey in the taxonomy tree and cannot be customized. Only devices containing RocketIO MGTs are supported by the Wizard. See the *Virtex-4 Family Overview* [Ref 1] for a list of devices containing RocketIO MGTs.
5. Click the **Generation** tab and select either Verilog or VHDL as the output language.
6. Click **OK**.

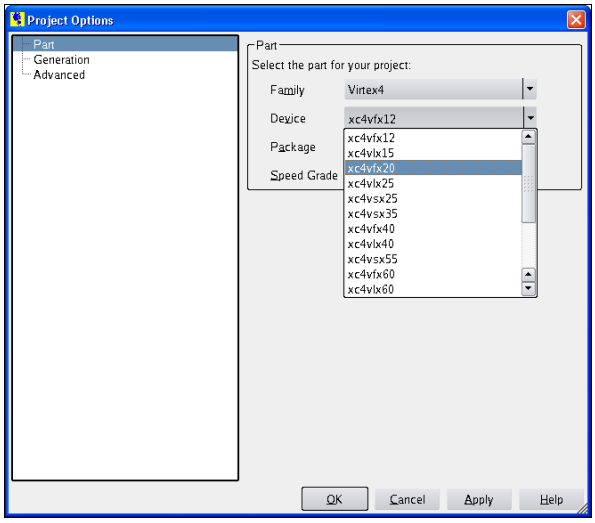

Figure 3-3: **Target Architecture Setting**

Generating the Wrapper

This section shows how to generate an example Virtex-4 FX FPGA RocketIO MGT Wizard wrapper using the default values. The core and its supporting files, including the example design, are generated in the project directory. For additional details about the example design files and directories see "Implementing the Example Design," page 39.

1. Locate the Virtex-4 FX FPGA RocketIO MGT Wizard in the taxonomy tree under `/FPGA Features & Design/IO Interfaces`. (See Figure 3-4.)
2. Double-click Virtex-4 FX FPGA RocketIO MGT Wizard to launch the Wizard.

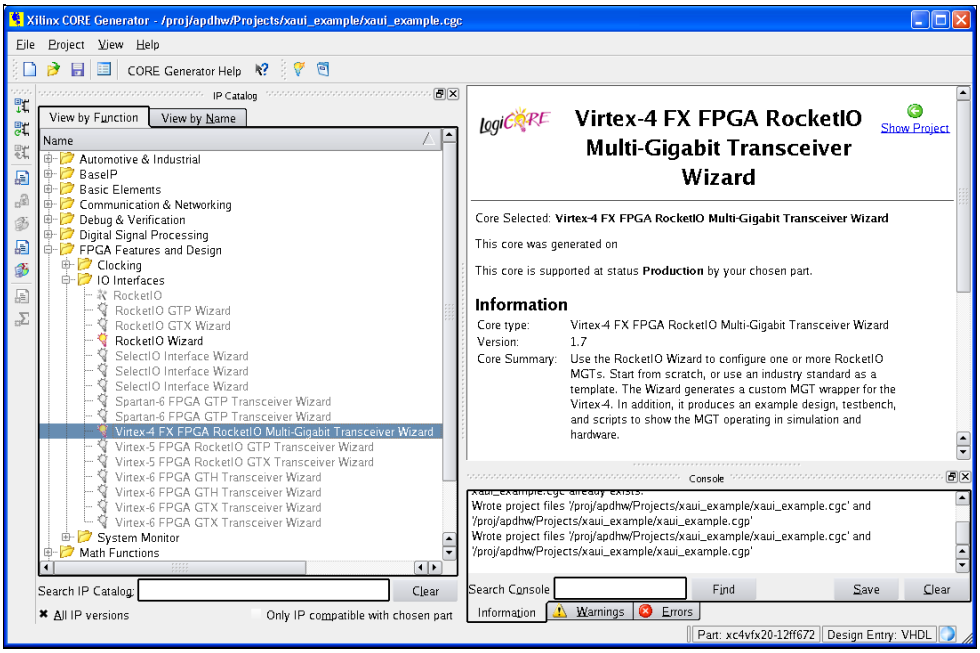

Figure 3-4: **Locating the Virtex-4 FX FPGA RocketIO MGT Wizard**

Page 1 of 9

Page 1 of the Wizard (Figure 3-5) is for selecting a component name, protocol file, and silicon revision.

1. In the Component Name field, enter a name for the core instance. This example uses the name **xaui_wrapper**.
2. Select one of the available protocols from the Protocol File drop-down list. This example uses the **xaui** protocol.
3. Select the version of the target device from the Silicon Version drop-down list. This example uses the default **PRODUCTION STEP 0 or 1**.

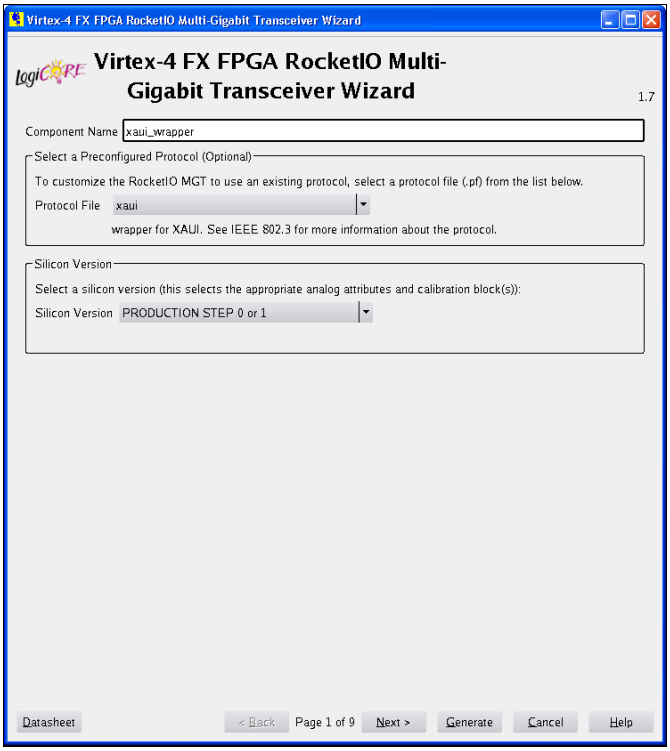

Figure 3-5: **Virtex-4 FX FPGA RocketIO MGT Wizard - Page 1**

Page 2 of 9

Page 2 of the Wizard (Figure 3-6) is for selecting the placement of the RocketIO MGTs.

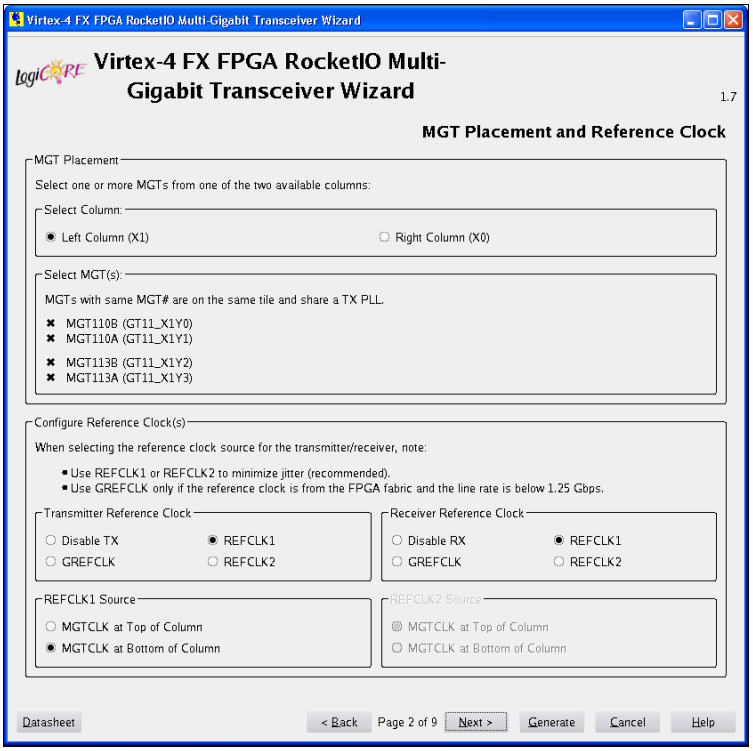

Figure 3-6: **Virtex-4 FX FPGA RocketIO MGT Wizard - Page 2**

The number of transceivers that appear depends on the target device and package that are selected. The XAUI example design uses four of the left-column transceivers by default.

Table 3-1: **MGT Placement Options**

Option	Description
Select Column	Select the column in which the transceivers are to be placed. Wrappers only include transceivers from a single column.
Select MGT(s)	Select the individual number of transceivers by location to be used in the target design.

Configure Reference Clock(s)

In addition to placing the transceivers, Page 2 is for configuring the transmitter and receiver reference clocks (Figure 3-6). Table 3-2 and Table 3-3 describe the TX/RX reference clock options. The unused settings for the example XAUI design are shaded in the tables.

Note: In the Wizard, when the transmitter and receiver use the same reference clock, they must also use the same Line Rate (see Figure 3-7, page 22).

Table 3-2: **Transmitter Reference Clock**

Option	Description
Disable TX	Disables the transmitter if it is not being used.
GREFCLK	Used only for low-speed applications.
REFCLK1	Provides a low-jitter clock (required for XAUI).
REFCLK2	Provides an alternate low-jitter clock (would also work in this XAUI example).

Table 3-3: **Receiver Reference Clock**

Option	Description
Disable RX	Disables the receiver if it is not being used.
GREFCLK	Used only for low-speed applications.
REFCLK1	Provides a low-jitter clock (required for XAUI). Additionally, it is already available because the transmitter is using it.
REFCLK2	Provides a low-jitter clock (would also work in this XAUI example).

Each column has two sets of differential pins (MGTCLK pins) that act as clock sources for REFCLK1 and REFCLK2. Either REFCLK can be driven by either set of MGTCLK pins. Table 3-4 and Table 3-5 show the selections for the REFCLK1 and REFCLK2 clock sources. To drive each REFCLK, an external clock source must be connected to the MGTCLK selected to be the REFCLK source.

Table 3-4: **REFCLK1 Source**

Option	Description
MGTCLK at Top of Column	Assigns the REFCLK1 source at the top of the column. (Used arbitrarily for this XAUI example.)
MGTCLK at Bottom of Column	Assigns the REFCLK1 source at the bottom of the column. (Could have been used in this XAUI example instead.)

The source for REFCLK2 is automatically set to the MGTCLK opposite the selected REFCLK1 source. Because *both* the transmitter and receiver use the REFCLK1 source in the example design, the REFCLK2 clock source is disabled.

Table 3-5: **REFCLK2 Source**

Option	Description
MGTCLK at Top of Column	Assigns the REFCLK2 source at the top of the column.
MGTCLK at Bottom of Column	Assigns the REFCLK2 source at the bottom of the column.

Page 3 of 9

Page 3 of the Wizard (Figure 3-7) is for selecting the datapath and line rate settings of the transmitter and receiver.

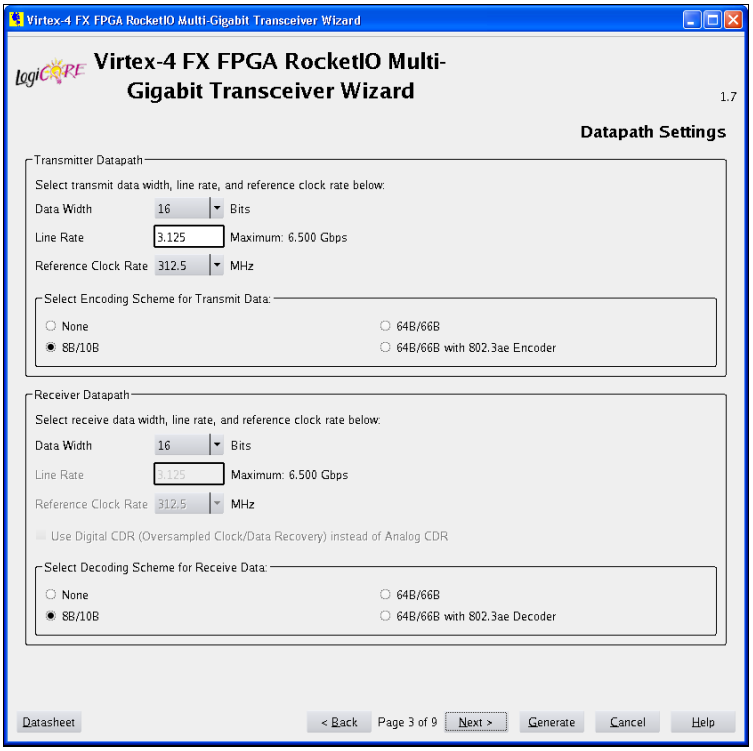

Figure 3-7: **Virtex-4 FX FPGA RocketIO MGT Wizard - Page 3**

Table 3-6, page 22 shows the options for the transmitter datapath. Because this example uses the XAUI protocol file (Figure 3-5, page 19), the data widths displayed in the drop-down list reflect the options available for an 8B/10B encoding scheme and the 8B/10B radio button is already selected.

Table 3-6: **Transmitter Datapath**

Option		Description
Data Width (Dependent on Encoding Scheme)	None	Allows data width selection of 8, 10, 16, 20, 32, 40, 64, or 80 bits.
	8B/10B	Allows data width selection of 8, 16, 32, or 64 bits. (XAUI example uses 16 bits.)
	64B/66B	Allows data width selection of 32 or 64 bits.
	64B/66B with IEEE 802.3ae Encoder	Data width is 32 bits.
Line Rate		Allows selection of a line rate up to 6.5 Gb/s. The XAUI example has a line rate of 3.125 Gb/s. Note that the maximum line rate achievable in hardware depends on the Silicon Version selected in Figure 3-5, page 19.

Table 3-6: **Transmitter Datapath *(Cont'd)***

Option		Description
Reference Clock Rate		The appropriate range for the reference clock rate is dependent upon the selected line rate and data width. Available clock rates for this example are 390.625, 312.5, 195.3125, or 156.25 MHz. The XAUI example uses a reference clock rate of 312.5 MHz.
Encoding Scheme for Transmit Data	None	Data is passed with no conversion.
	8B/10B	Data is passed to an 8B/10B encoder prior to transmission.
	64B/66B	Data is passed to a 64B/66B encoder prior to transmission.
	64B/66B with IEEE 802.3ae Encoder	Data is passed to an IEEE 802.3ae encoder. The resulting data is then passed to a 64B/66B encoder prior to transmission. This selection is appropriate for 10 Gb Ethernet.

Table 3-7 shows the options for the receiver datapath.

Table 3-7: **Receiver Datapath**

Option		Description
Data Width (Dependent on Encoding Scheme)	None	Allows data width selection of 8, 10, 16, 20, 32, 40, 64, or 80 bits.
	8B/10B	Allows data width selection of 8, 16, 32, or 64 bits. (XAUI example uses 16 bits.)
	64B/66B	Allows data width selection of 32 or 64 bits.
	64B/66B with IEEE 802.3ae Encoder	Data width is 32 bits.
Line Rate		Sets the Line Rate of the receiver. If TX and RX use the same reference clock, RX Line Rate is automatically set to match the transmitter. The XAUI example has a line rate of 3.125 Gbps.
Reference Clock Rate		Sets the reference clock rate of the receiver. If TX and RX use the same reference clock, Reference Clock Rate is automatically set to match the transmitter. The XAUI example uses a reference clock rate of 312.5 MHz.
Use Digital CDR		Digital clock/data recovery (CDR) can be used for line rates below 1.25 Gb/s, and is required for line rates that would otherwise force the RX PLL below its recommended operating range. When Digital CDR is activated, the MGT uses built in 8X oversampling to recover clock and data from the incoming serial stream. This choice is disabled because digital CDR cannot be used at 3.125 Gb/s.

Table 3-7: **Receiver Datapath *(Cont'd)***

Option		Description
Decoding Scheme for Receive Data	None	Data is passed with no conversion.
	8B/10B	Data is decoded from 8B/10B.
	64B/66B	Data is aligned using the RX Block Sync State Machine defined in IEEE 802.3, then decoded from 64B/66B.
	64B/66B with IEEE 802.3ae Decoder	Data is aligned using the IEEE 802.3ae RX Block Sync State Machine, decoded from 64B/66B, then passed to the IEEE 802.3ae Decoder. This selection is appropriate for 10 Gb Ethernet.

Page 4 of 9

Page 4 of the Wizard (Figure 3-8) is for selecting the synchronization and alignment settings of the physical coding sublayer (PCS). If 64B/66B decoding was selected on Page 3 (Figure 3-7, page 22), the comma alignment and comma detection options are disabled, because 64B/66B automatically uses the IEEE 802.3 RX Block Sync state machine.

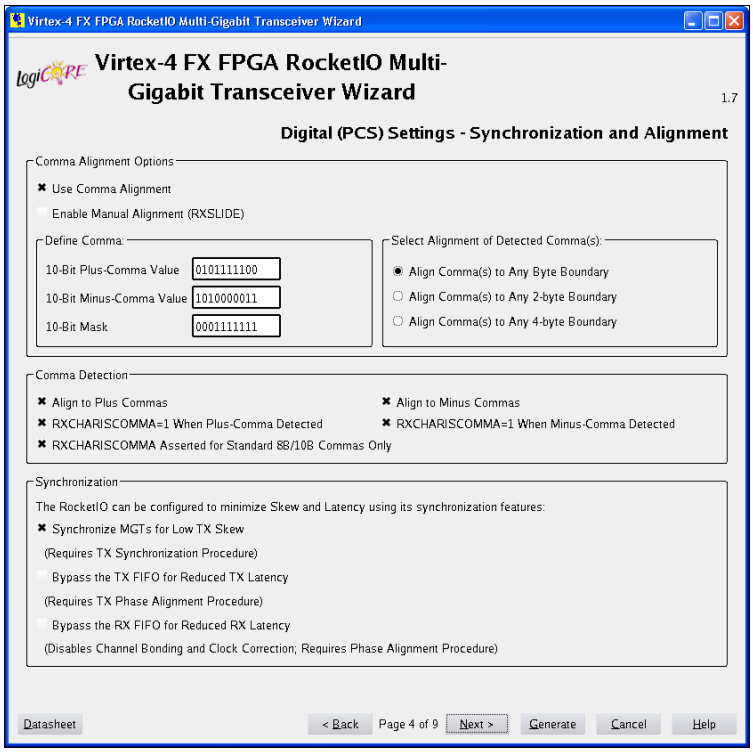

Figure 3-8: **Virtex-4 FX FPGA RocketIO MGT Wizard - Page 4**

Table 3-8 shows the comma alignment options used by the example XAUI design.

Table 3-8: **Comma Alignment Options**

Option		Description
Use Comma Alignment		Activates the comma alignment circuits. If not selected, comma alignment is bypassed. XAUI protocol uses comma alignment.
Enable Manual Alignment (RXSLIDE)		Allows the application to control alignment. See the *Virtex-4 FPGA RocketIO Multi-Gigabit Transceiver User Guide* [Ref 2] for more details on using RXSLIDE to manually align incoming data.
Define Comma	10-Bit Plus-Comma Value	XAUI uses `0101111100` (K.28.5 from the 8B/10B table).
	The 10-Bit Minus-Comma Value	XAUI uses `1010000011` **(K.28.5 from the 8B/10B table).**
	The 10-Bit Mask	XAUI uses `1111111111`, requiring all ten bits of the comma patterns to match the received data for alignment to occur.

Table 3-8: **Comma Alignment Options *(Cont'd)***

Option	Description
Select Alignment of Detected Comma(s)	Align Comma(s) to Any Byte Boundary; when a comma is found, the parallel data is shifted so the comma is aligned with the closest byte boundary.
	Align Comma(s) to Any 2-Byte Boundary; when a comma is found, the parallel data is shifted so the comma is aligned with the closest 2-byte boundary.
	Align Comma(s) to Any 4-Byte Boundary; when a comma is found, the parallel data is shifted so the comma is aligned with the closest 4-byte boundary.

Table 3-9 shows the comma detection options. The XAUI example design uses all these options.

Table 3-9: **Comma Detection Options**

Option	Description
RXCOMMADET=1 When Aligning to Plus-Comma	The RXCOMMADET signal is asserted when aligning to a Plus-Comma.
RXCOMMADET=1 When Aligning to Minus-Comma	The RXCOMMADET signal is asserted when aligning to a Minus-Comma.
RXCHARISCOMMA=1 When Plus-Comma Detected	The RXCHARISCOMMA signal is asserted when a Plus-Comma is detected.
RXCHARISCOMMA=1 When Minus-Comma Detected	The RXCHARISCOMMA signal is asserted when a Minus-Comma is detected.
RXCHARISCOMMA Asserted for Standard 8B/10B Commas Only	The RXCHARISCOMMA signal is not asserted when a custom Comma character is present.

Table 3-10, page 27 show the synchronization options. The XAUI example design uses none of these options.

Table 3-10: **Synchronization Options**

Option	Description
Synchronize MGTs	Uses TX synchronization procedure to reduce or eliminate skew between lanes. See the *Virtex-4 FPGA RocketIO Multi-Gigabit Transceiver User Guide* [Ref 2] for details.
Bypass the TX FIFO	Use TX Synchronization instead of a TX FIFO to perform the required phase alignment between the MGT's PCS and PMA clocks. Bypassing the TX FIFO can reduce TX latency, but because TX Synchronization becomes necessary, it makes the startup procedure for the MGT more complex.
Bypass the RX FIFO	In systems that don't require Clock Correction or Channel Bonding, the RX Synchronization procedure can be used in place of the RX FIFO. This can reduce RX latency, but results in a more complex MGT startup procedure.

Page 5 of 9

Page 5 of the Wizard (Figure 3-9) is for selecting the channel bonding settings of the physical coding sublayer (PCS).

Note: When using this version of the Virtex-4 FX FPGA RocketIO MGT Wizard for channel bonding designs, the bonded transceivers must not span separate columns in the device.

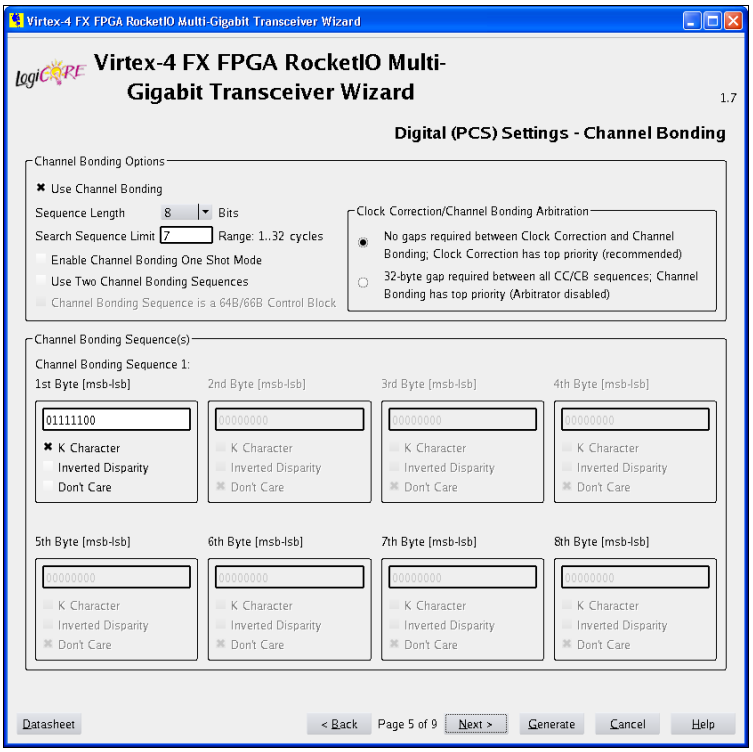

Figure 3-9: **Virtex-4 FX FPGA RocketIO MGT Wizard - Page 5**

Table 3-11 shows the channel bonding options used in the XAUI example design.

Table 3-11: **Channel Bonding Options**

Option	Description
Use Channel Bonding	Checking this box enables channel bonding and activates the remaining options.
Sequence Length	The channel bonding sequence length can be up to eight symbols long. The XAUI channel bonding sequence is **8 bits** (one symbol) long.
Search Sequence Limit	Represents the maximum number (up to 32) of symbols (cycles) that can occur between the bonding sequences detected on the master and slave channels, while still allowing for successful channel alignment. XAUI channel bonding allows 31 cycles between master and slave channels.
Enable Channel Bonding One Shot Mode	In one-shot mode, channel bonding is attempted only once after it is enabled.

Table 3-11: **Channel Bonding Options *(Cont'd)***

Option	Description
Use Two Channel Bonding Sequences	The RocketIO transceivers can use two sequences up to four symbols each or one sequence up to eight symbols. The width of the symbols (8 bits or 10 bits) depends on the width of the RX datapath and the selected decoding scheme.
Channel Bonding Sequence is a 64B/66B Control Block	If 64B/66B decoding is used, the channel bonding sequence is an 8 byte block code. If the block code is a control block, this option must be checked. XAUI uses 8B/10B encoding so this option is disabled.
Clock Correction/Channel Bonding Arbitration	The Channel Bonding/Clock Correction arbitrator handles cases where Channel Bonding and Clock Correction sequences arrive close together. When this happens, the arbitrator gives priority to Clock Correction. If Channel Bonding must have priority in a given protocol, the arbitrator can be disabled, but Channel Bonding and Clock Correction sequences must have a gap between them of at least 32 bytes. The XAUI protocol does not require the arbitrator to be disabled.

Figure 3-8, page 25 shows that up to eight symbols can be defined as a channel bonding sequence. Each byte is defined with the most-significant bit to the left. Table 3-12 describes the channel bonding sequence for the XAUI example design. The XAUI design uses a single-character sequence (one symbol), automatically disabling the 2nd through 8th bytes.

Table 3-12: **Channel Bonding Sequences**

Option	Description
Byte (Symbol)	Set each symbol to match the pattern the protocol requires. The XAUI sequence length is 8 bits: `01111100`. The other symbols are disabled because the Channel Bonding Sequence Length is set to 8.
K Character	This option is available when 8B/10B decoding is selected. When checked, the symbol is an 8B/10B K character.
Inverted Disparity	Some protocols with 8B/10B decoding use symbols with deliberately inverted disparity as channel bonding symbols. Inverted Disparity should be checked when the symbol arrives with inverted disparity in Channel Bonding Sequences. XAUI does not use Inverted Disparity.
Don't Care	Multiple-byte sequences can have *wild card* symbols by checking the **Don't Care** box. Unused bytes are automatically set to Don't Care.

Page 6 of 9

Page 6 of the Wizard (Figure 3-10) is for selecting the clock correction settings of the physical coding sublayer (PCS).

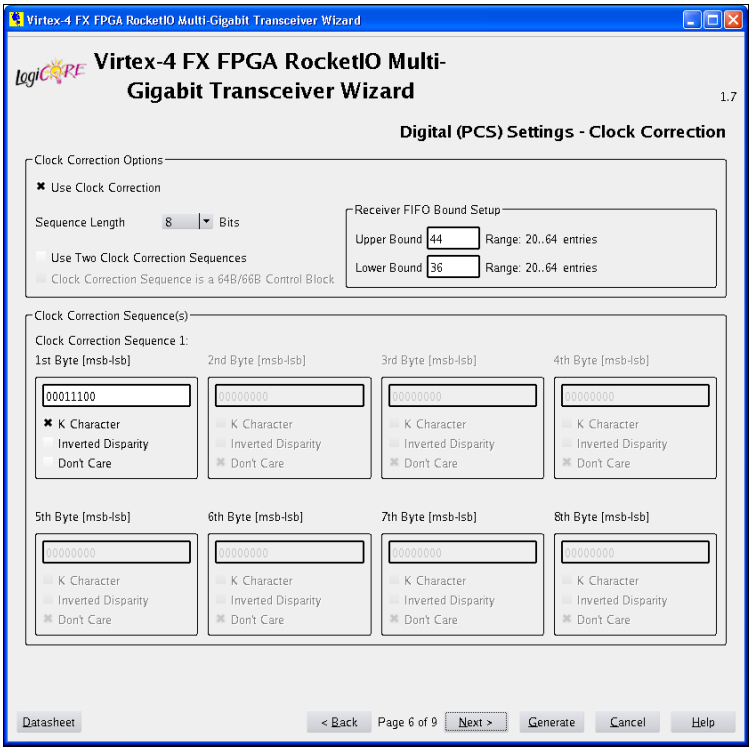

Figure 3-10: **Virtex-4 FX FPGA RocketIO MGT Wizard - Page 6**

Table 3-13 shows the clock correction options used in the XAUI example design.

Table 3-13: **Clock Correction Options**

Option	Description
Use Clock Correction	Checking this box enables clock correction and activates the remaining options.
Sequence Length	The clock correction sequence can be up to eight symbols long. The number of bits in a symbol depends on the RX datapath width and the selected decoding. The XAUI clock correction sequence is 8 bits (one symbol) long.
Use Two Clock Correction Sequences	The RocketIO transceivers can use two Clock Correction sequences up to four symbols each or one Clock Correction sequence up to eight symbols.

Table 3-13: **Clock Correction Options**

Option	Description
Clock Correction Sequence is a 64B/66B Control Block	If 64B/66B decoding is selected, the Clock Correction Sequence is an 8-byte block code. If the block code must be a control block, check this box. XAUI uses 8B/10B encoding so this option is disabled.
Receiver FIFO Bound Setup	Clock Correction depends on the number of symbols in the RX FIFO. When the count is less than the minimum bound, it replicates CC sequences to prevent FIFO underflow. When the count exceeds the maximum bound, it deletes sequences to prevent overflow. XAUI has an upper bound of 48 cycles and a lower bound of 36 cycles

Figure 3-9, page 28 shows that up to eight, 8-bit symbols can be defined as a clock correction sequence. Each byte is defined with the most-significant bit to the left. Table 3-14 describes the clock correction sequence for the XAUI example design. The XAUI design uses a one clock correction sequence, automatically disabling the 2nd through 8th bytes.

Table 3-14: **Clock Correction Sequences**

Option	Description
Byte (Symbol)	Set each symbol based on the protocol requirements. The XAUI sequence length is 8 bits: `00011100`. The remaining symbols are disabled because XAUI has a Clock Correction Sequence Length of 8 bits.
K Character	This option is available when 8B/10B decoding is selected. When checked, the symbol is an 8B/10B K character.
Inverted Disparity	Some protocols with 8B/10B decoding use symbols with deliberately inverted disparity as Clock Correction symbols. This option should be checked when symbols with this attribute appear in Clock Correction Sequences. XAUI does not use Inverted Disparity.
Don't Care	Multiple-byte sequences can have *wild card* symbols by checking the **Don't Care** box. Unused bytes are automatically set to Don't Care.

Page 7 of 9

Page 7 of the Wizard (Figure 3-11) is for selecting the analog physical media attachment (PMA) settings.

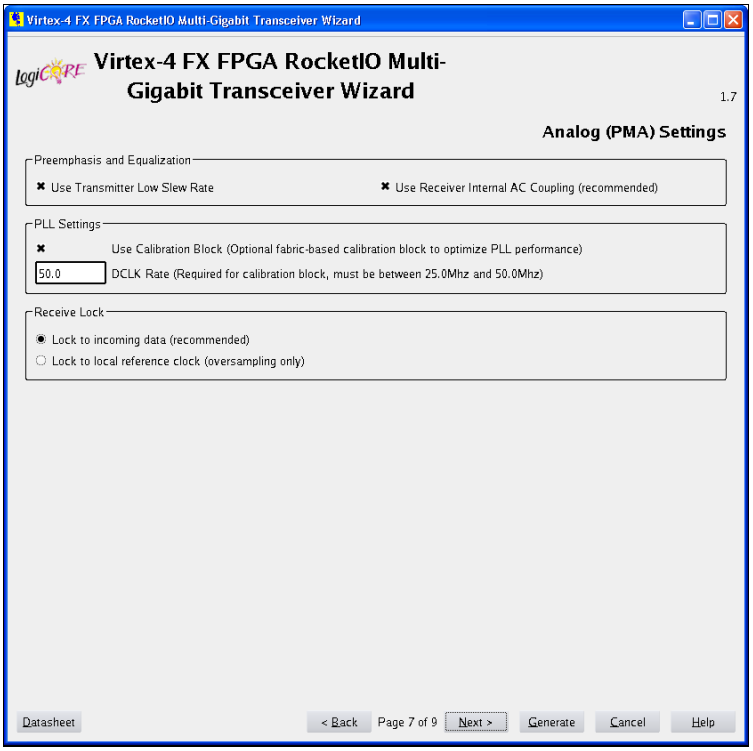

Figure 3-11: **Virtex-4 FX FPGA RocketIO MGT Wizard - Page 7**

Table 3-15 shows the PMA settings used in the XAUI example design.

Table 3-15: **Analog PMA Settings**

Option	Description
Preemphasis and Equalization	The **Use Transmitter Low Slew Rate** option decreases the output signal current thus lowering the signal slew rate.
	The **Use Receiver Internal AC Coupling** option inserts DC blocking capacitors in the serial signal path. This option should be used unless external AC coupling is provided or the application does not use DC-balanced data. XAUI uses DC-balanced 8B/10B encoded data with internal AC coupling.
PLL Settings	The **Use Calibration Block** option optimizes PLL performance. This option should be set to TRUE for all designs. The calibration block is removed automatically from silicon versions whose performance is not improved by a calibration block.
	The **DCLK Rate** is used internally by the calibration block to drive the Dynamic Reconfiguration Port and clock frequency detector circuits. The default value of 50.0 Mhz is used for this example.

Table 3-15: **Analog PMA Settings *(Cont'd)***

Option	Description
Receive Lock	The **Lock to incoming data** option turns on Clock Data Recovery, allowing data to be received without oversampling. This is the default operating mode.
	The **Lock to local reference clock** option forces the PLL to use the local reference clock that ignores phase information in the incoming data. This option is intended for oversampling applications.

Page 8 of 9

Page 8 of the Wizard (Figure 3-12) is for selecting special features for the protocol wrapper, such as CRC, out-of-band (OOB) signaling, and loopback.

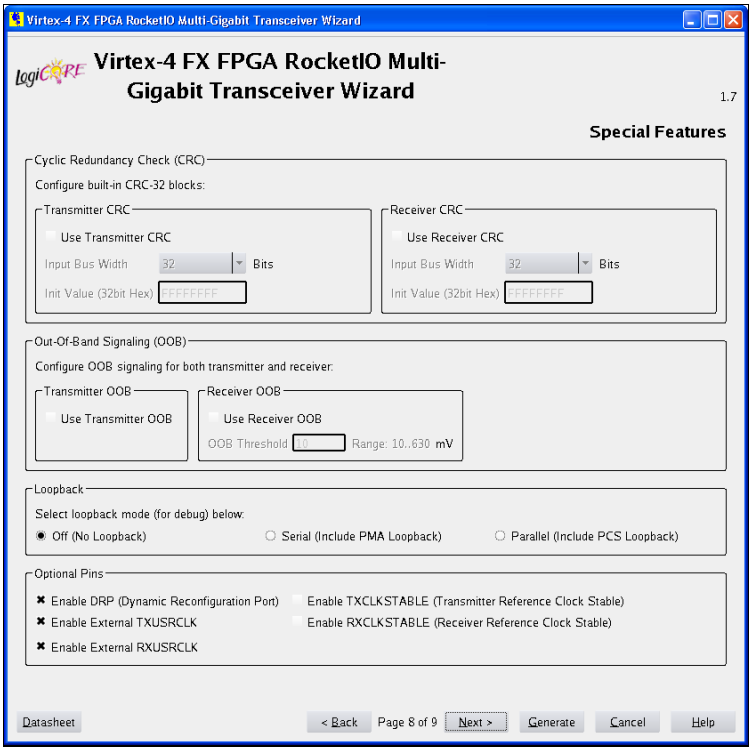

Figure 3-12: **Virtex-4 FX FPGA RocketIO MGT Wizard - Page 8**

Table 3-16 shows the cyclic redundancy check (CRC) options for the transmitter and receiver. The XAUI example design does not use the CRC options.

Table 3-16: **Cyclic Redundancy Check Options**

Option		Description
Transmitter CRC	Use Transmitter CRC	Checking this option enables internal CRC calculation on transmit data and enables the associated options.
	Input Bus Width	This value determines the maximum width of the data on which the CRC calculation is performed. It should be set to match the application data path width.
	Init Value	Protocol-specific 32-bit hexadecimal value for the initial CRC setting.

Table 3-16: **Cyclic Redundancy Check Options *(Cont'd)***

Option		Description
Receiver CRC	Use Receiver CRC	Checking this option enables internal CRC calculation on receive data and enables the associated options.
	Input Bus Width	This value determines the maximum width of the data on which the CRC calculation is performed. It should be set to match the application data path width.
	Init Value	Protocol-specific 32-bit hexadecimal value for the initial CRC setting.

Table 3-17 shows the options for OOB signaling.

Table 3-17: **Out-of-Band Signaling Options**

Option	Description
Use Transmitter OOB	Enables out-of-band signaling if the protocol in use requires this feature.
Use Receiver OOB	Enables out-of-band signaling if the protocol in use requires this feature.
Receiver OOB Threshold	Specifies the level in 10 mV steps at which the OOB signal is recognized.

Table 3-18 shows the options for loopback. Loopback is used for diagnostic testing and is disabled in normal operation. When loopback is activated, the resulting wrapper is permanently set to loopback mode.

Table 3-18: **Loopback Options**

Option	Description
Off (No Loopback)	Disables loopback function for normal operation.
Serial (Include PMA Loopback)	Enables diagnostic loopback of the serial signal through the both the PCS and PMA logic sections.
Parallel (Include PCS Loopback)	Enables diagnostic loopback through the PCS logic section, bypassing the PMA logic and serial output.

Table 3-19 shows the optional pins.

Table 3-19: **Optional Pins**

Option	Description
Enable DRP	Brings the Dynamic Reconfiguration Port signals out to logical pins on the wrapper interface.
Enable External TXUSRCLK	Brings the external TXUSRCLK input out to logical pins on the wrapper.
Enable External RXUSRCLK	Brings the external RXUSRCLK input out to logical pins on the wrapper.

Table 3-19: **Optional Pins *(Cont'd)***

Option	Description
Enable TXCLKSTABLE	This input signal is used in conjunction with external PLLs. The external PLL asserts TXCLKSTABLE when it has acquired lock and is stable.
Enable RXCLKSTABLE	This input signal is used in conjunction with external PLLs. The external PLL asserts RXCLKSTABLE when it has acquired lock and is stable.

Page 9 of 9

Page 9 of the Wizard (Figure 3-13) is a summary of the settings for the protocol wrapper core. After reviewing the settings, click Generate to exit the Wizard.

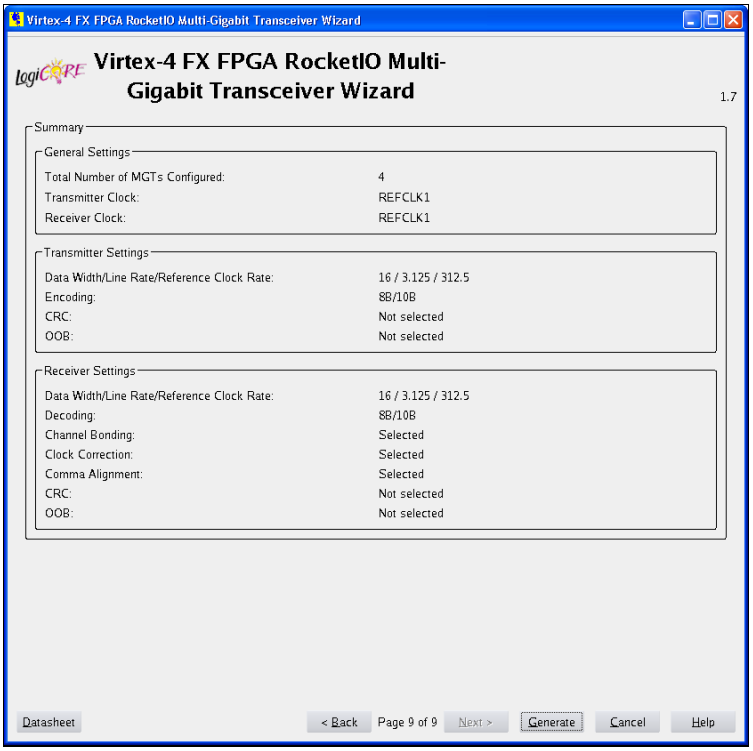

Figure 3-13: **Virtex-4 FX FPGA RocketIO MGT Wizard - Page 9**

Table 3-20 shows the summary entries for the XAUI example wrapper.

Table 3-20: **Summary Page**

Option		Setting
General Settings	Total Number of Transceivers Configured	4(1)
	Transmitter Clock	REFCLK1(1)
	Receiver Clock	REFCLK1(1)
Transmitter Settings	Data Width/Line Rate/Reference Clock Rate	16/3.125/312.5(2)
	Encoding	8B/10B(2)
	CRC	N/A(3)
	OOB	N/A(3)

Table 3-20: **Summary Page *(Cont'd)***

Option		Setting
Receiver Settings	Width/Line Rate/Reference Clock Rate	16/3.125/312.5(2)
	Decoding	8B/10B(2)
	Channel Bonding	Selected(4)
	Clock Correction	Selected(5)
	Comma Alignment	Selected(6)
	CRC	N/A(3)
	OOB	N/A(3)

Notes:

1. See Figure 3-6, page 20.
2. See Figure 3-7, page 22.
3. See Figure 3-12, page 34.
4. See Figure 3-9, page 28.
5. See Figure 3-10, page 30.
6. See Figure 3-8, page 25.

Implementing the Example Design

After wrapper generation is complete, the results can be tested in hardware. With each wrapper, the CORE Generator tool produces an example design incorporating the wrapper and additional blocks allowing the wrapper to be driven and monitored in hardware. The generated output also includes several scripts to assist in running the Xilinx software.

From the command prompt, navigate to the project directory and type the following:

For Windows

```
ms-dos> cd xaui_wrapper\scripts
ms-dos> xilperl build_script.pl
```

For UNIX

```
linux-shell% cd xaui_wrapper/scripts
linux-shell% xilperl build_script.pl
```

These commands execute a script that synthesizes, builds, maps, places, and routes the example design and produces a bitmap file. The resulting files are placed in the *scripts* directory. To learn more about the script and its available options, enter the following command:

```
xilperl build_script.pl -help
```

Simulating the Example Design

The Virtex-4 FX FPGA RocketIO MGT Wizard provides a quick way to simulate and observe the behavior of the wrapper using the provided example design and script files.

Using ModelSim

Prior to simulating the wrapper with ModelSim, the functional (gate-level) simulation models must be generated. All source files in the following directories must be compiled to a single library as shown in Table 3-21. Refer to the *Synthesis and Simulation Design Guide* for ISE® 12.1, available in the ISE Software Documentation [Ref 4], for instructions on how to compile ISE simulation libraries.

Table 3-21: **Required ModelSim Simulation Libraries**

HDL	Library	Source Directories
Verilog	UNISIMS_VER	<*Xilinx dir*>/verilog/src/unisims <*Xilinx dir*>/smartmodel/<*OS*>/wrappers/mtiverilog
VHDL	UNISIM	<*Xilinx dir*>/vhdl/src/unisims <*Xilinx dir*>/smartmodel/<*OS*>/wrappers/mtivhdl

Note: *OS* refers to the following operating systems: lin, lin64, nt, nt64

The Wizard provides a command line script for use within ModelSim. To run a VHDL or Verilog ModelSim simulation of the wrapper, use the following instructions:

1. Launch the Modelsim simulator and set the current directory to

   <*project_directory*>/xaui_wrapper/scripts.

2. Set the MTI_LIBS variable:

   modelsim> **setenv MTI_LIBS <*path to compiled libraries*>**

3. Launch the simulation script:

   modelsim> **do example_sim.do.**

The ModelSim script compiles the example design and test bench, and adds the relevant signals to the wave window.

Example Design Hierarchy

The hierarchy for the design used in this example is as follows:

```
example_tb
|___example_mgt_top
    |___xaui_wrapper
    |   |___gt11
    |   |___cal_block_v1_4_1
    |
    |___mgt_clock_module
    |   |___gt11clk
    |
    |___mgt_usrclk_source
    |___frame_gen
    |___frame_check
```

Using the Wrapper

Figure 3-14 shows the modules produced by the Virtex-4 FX FPGA RocketIO MGT Wizard.

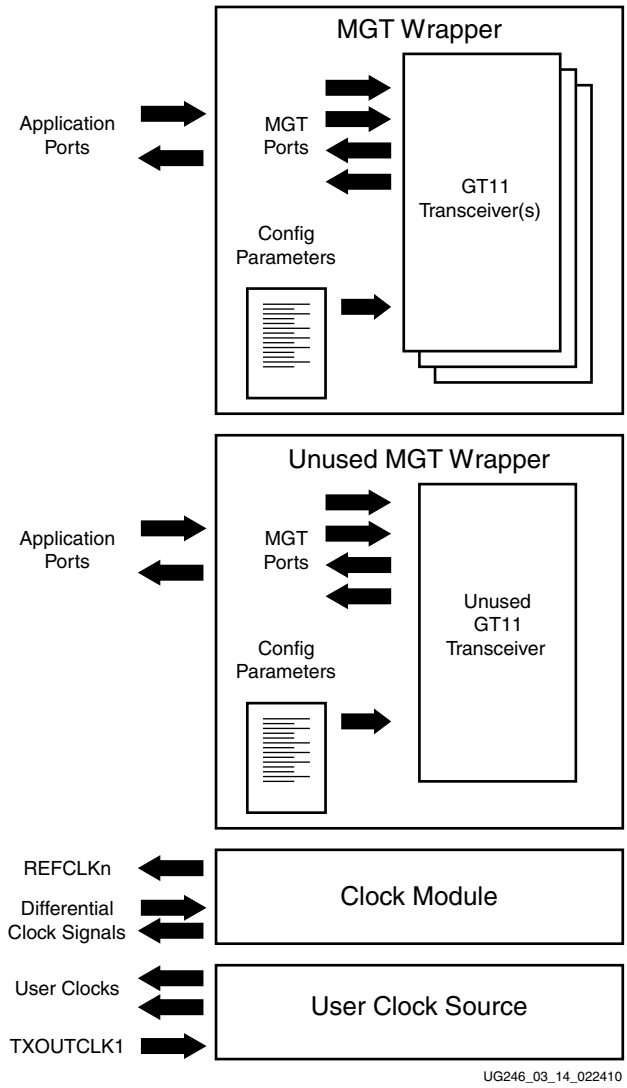

Figure 3-14: **Virtex-4 FX FPGA RocketIO MGT Wizard Wrapper Modules**

MGT Wrapper

This is the primary module which wraps one or more RocketIO GT11 primitives configured for a custom application by the settings in the Wizard. Instantiate this module in your design in place of the GT11 primitive. Individual GT11 ports are present on the top-level wrapper module. Each port name is prefixed with the transceiver number, for example **MGT0_**. Connect these as you would the ports on the GT11 primitive. See the example design for details.

Unused Transceiver Wrapper

Virtex-4 MGTs occur in pairs, with each pair occupying a tile. Since the two transceivers share components, both transceivers on a tile must be instantiated if either one is used. The wizard provides an unused transceiver wrapper to instantiate unused transceivers on a tile with a single active transceiver. A subset of the available transceiver ports are brought out at the top of the module and should be connected as appropriate. See the example design for details.

Clock Module

This module simplifies the routing of the differential reference clock source from the device pins to the REFCLK inputs of the transceiver(s). Table 3-22 describes the available ports.

Table 3-22: **Clock Module Ports**

Ports	Description
<*pos*>_MGTCLK_PAD_N_IN	The actual name of these ports is determined by the clock position selected in the Wizard. <pos> is either UPPER or LOWER. These ports receive the MGTCLK_PAD lines directly from the appropriate pins.
<*pos*>_MGTCLK_PAD_P_IN	
REFCLK*n*_OUT	The actual name of this port is determined by the reference clock input selected in the Wizard. The variable *n* is either 1 or 2. This port is connected to the appropriate reference clock input of the transceiver wrapper.

User Clock Source

This module instantiates a PMCD primitive to provide a proper source for the transceiver user clock signals. Table 3-23 describes the available ports.

Table 3-23: **User Clock Source Module Ports**

Ports	Description
USRCLK_SOURCE	These are the User Clock source signals. Connect them to the transceiver TXUSRCLK_IN and TXUSRCLK2_IN ports respectively.
USRCLK2_SOURCE	
TXOUTCLK1_IN	Connect the transceiver TXOUTCLK1_OUT signal to this port
PMCD_RESET	This is the PMCD reset input. It can be connected to the TXPMARESET line.

Running the Example Design

The build script produces a file called `example_mgt_top.bit`. This file can be used to configure the target device specified in the CORE Generator project file.

The example design includes several ChipScope™ Pro modules which drive and monitor the signals of the mgt_wrapper. If the wrapper includes several transceivers, the example design includes a multiplexer allowing the individual selection of each transceiver.

ChipScope Pro software is a Xilinx tool for logic debugging. It allows internal FPGA signals to be monitored and controlled through a configuration cable from a virtual console on a workstation or PC. The example design normally builds with ChipScope Pro modules included.

To run the design, the following steps are required:

1. Ensure that the target device is powered correctly
2. Connect the reference clock inputs to an appropriate reference clock running at the correct frequency
3. Connect the serial I/O. There are three modes available:
   a. Wrapper-to-wrapper: Configure two FPGAs with the example design and interconnect their serial I/O ports
   b. Wire loopback: Configure a single FPGA with the example design and use a set of cables to connect the TX serial I/O pins to the RX serial I/O pins
   c. Rerun the Wizard and use the Serial Loopback option
4. To see the transceiver signals:
   a. Start ChipScope Pro Analyzer version 12.1 or greater on your PC or workstation. A free evaluation copy of ChipScope Pro software is available at www.xilinx.com
   b. Configure the FPGA with the example design
   c. Load the ChipScope project file script `example_mgt_top.cpj` to automatically configure the presentation of the signals from the example design

While the example design is operating, there are several signals to watch:

1. The TX_LOCK and RX_LOCK signals are most important. If the transceiver is powered correctly, the reference clock is running at the correct rate, and the serial I/O for each transceiver is connected correctly, TX_LOCK and RX_LOCK should be High for active TX and RX transceiver datapaths.
2. Asserting PMA Reset should cause LOCK to go Low temporarily until the transceiver is able to reestablish a PLL lock.
3. The TX and RX PCS Resets are driven while LOCK is Low and continue for a few cycles following the assertion of LOCK. Buffer errors also cause assertion of TX or RX reset.
4. If the mgt_wrapper uses Alignment, the design attempts to align to the incoming data. When alignment occurs, the REALIGN signal goes High. Alignment can occur several times before settling.
5. While the RX datapath is aligning, there are usually several cycles of bad data to be flushed. This data can cause the error count of the frame checker in the example design to increment; a small number of errors detected after lock is normal.

Appendix A

References

Documents Specific to Virtex®-5 FPGAs

1. *Virtex-4 Family Overview* (DS112)

Documents Specific to RocketIO™ Transceivers

2. *Virtex-4 FPGA RocketIO Multi-Gigabit Transceiver User Guide* (UG076)
3. *LogiCORE IP Virtex-4 FX FPGA RocketIO Multi-Gigabit Transceiver Wizard Data Sheet* (DS138)

Xilinx Tools and Solutions

4. ISE Software Manuals